

A FILE FORMATS

Overview

The development tools support many file formats, in some cases several for each development tool. This appendix describes the formats of files that you prepare as input for the tools and points out features of files produced by the tools. The three types of file formats that you can learn about in this appendix are as follows:

- [“Source Files” on page A-2](#)
- [“Build \(Processed\) Files” on page A-5](#)
- [“Debugger Files” on page A-7](#)

Most of the development tools use industry standard file formats. Sources that describe these formats appear in [“Format References” on page A-8](#).

Source Files

This section describes the following types of input file formats:

- “C/C++ Source Files (.C, .CPP, .CXX)” on page A-2
- “Assembly Source Files (.ASM)” on page A-3
- “Assembly Initialization Data Files (.DAT)” on page A-3
- “Header Files (.H)” on page A-4
- “Linker Description Files (.LDF)” on page A-4
- “Linker Command-Line Files (.TXT)” on page A-4

C/C++ Source Files (.C, .CPP, .CXX)

These are text files containing C and/or C++ code, compiler directives, possibly a mixture of assembler code and directives, and (typically) pre-processor commands.

Several “dialects” of C code are supported: pure (portable) ANSI C, and at least two subtypes^{*} of ANSI C with ADI extensions. These extensions include memory type designations for certain data objects, and segment directives, used by the linker to structure and place executable files.

For information on using the C/C++ compiler and associated tools, as well as a definition of ADI extensions to ANSI C, see the *C/C++ Compiler & Library Manual for ADSP-21xx DSPs*.

For information on specifying the C/C++ dialect and general C code handling within VisualDSP++, see the *VisualDSP++ 2.0 User’s Guide for ADSP-21xx DSPs*.

^{*} With and without builtin function support; a minimal differentiator. There are others.

Assembly Source Files (.ASM)

Assembly source files are text files containing assembly instructions, assembler directives, and (optionally) preprocessor commands. For information on assembly instructions, see the user's manual for the corresponding DSP.

The DSP's instruction set is supplemented with assembler directives. Preprocessor commands control macro processing and conditional assembly or compilation.

For information on the assembler and preprocessor, see the *Assembler and Preprocessor Manual for ADSP-21xx DSPs*.

Assembly Initialization Data Files (.DAT)

These are text files containing fixed-point or floating-point data. These files can provide the initialization data for an assembler `.VAR` directive or serve in other tool operations. When a `.VAR` directive uses a `.DAT` file for initialization data, the assembler reads the data file and initializes the buffer in the output object file (`.DOJ`). Data files have one data value per line and may have any number of lines.

The `.DAT` extension is merely explanatory or mnemonic. A directive to `#include <file>` can of course take any filename (or extension) as an argument.

For all numeric bases, the assembler uses 16-bit words for data storage; 24-bit data is for PM only. The largest word in the buffer determines the size for all words in the buffer. If you have some 8-bit data in a 16-bit wide buffer, the assembler loads the equivalent 8-bit value into the most significant 8 bits into the 8-bit memory location and zero-fills the lower eight bits.

Fixed-point values (integers) in data files may be signed, and they may be decimal-, hexadecimal-, octal-, or binary-base values. The assembler uses

Source Files

the prefix-conventions in [Table A-1](#) for identifying a fixed-point value's numeric base.

Table A-1. Fixed-Point Values in Data Files

Convention	Description
<i>0xnumber</i> , <i>Hnumber</i> <i>hnumber</i>	Hexadecimal number
<i>number</i> , <i>Dnumber</i> <i>dnumber</i>	("D", "d", or <i>no</i> prefix.) decimal number
<i>Onumber</i>	octal number
<i>Bnumber</i> , <i>bnumber</i>	binary number

Header Files (.H)

Header files are text files that contain macros or other preprocessor commands that the preprocessor substitutes into source files. For information on macros or other preprocessor commands, see the *Assembler and Preprocessor Manual for ADSP-21xx DSPs*.

Linker Description Files (.LDF)

The linker's .LDF files are ASCII text files that contain commands for the linker in the linker's scripting language. For information on this scripting language see ["Linker Description File Reference" on page 2-41](#).

Linker Command-Line Files (.TXT)

The linker's command line files are ASCII text files that contain command line input for the linker. For more information on the linker's command line, see ["Linker Command-Line Reference" in Chapter 2](#).

Build (Processed) Files

Build files are files that the development tools produce when building your VisualDSP project. This section describes the following types of build file formats:

- [“Assembler Object Files \(.DOJ\)” on page A-5](#)
- [“Archiver Archive Files \(.DLB\)” on page A-5](#)
- [“Linker Executable Files \(.DXE, .SM, .OVL, .dlb\)” on page A-5](#)
- [“Linker Memory Map Files \(.MAP\)” on page A-6](#)

Assembler Object Files (.DOJ)

The assembler’s output object files are binary, Executable-Linkable-File (ELF) format. Object files contain relocatable code and debugging information for your program’s segments. The linker processes your object files into an executable file. For information on the ELF format used for object files, see the [“Format References” on page A-8](#).

Archiver Archive Files (.DLB)

The archiver’s output archive files are binary, Executable-Linkable-File (ELF) format. Archive files contain one or more object files, called Archive Elements. The linker searches through archive files for any archive elements that your code uses. For information on the ELF format used for executable files, see the [“Format References” on page A-8](#).

Linker Executable Files (.DXE, .SM, .OVL, .dlb)

The linker’s output executable files are binary, Executable-Linkable-File (ELF) format. Executable files contain your program’s code and debugging information. The linker may fully or partially resolve addresses in

Build (Processed) Files

executable files, depending on the options given on the linker's command line and in your linker description file. For information on the ELF format used for executable files, see the TIS Committee texts cited in [“Format References” on page A-8](#).

Linker Memory Map Files (.MAP)

The linker's memory map files are ASCII text files that contain memory and symbol information for your executable file(s). The map contains a summary of memory that you define with MEMORY{} commands in your linker description file, and provides a listing of the absolute addresses of all symbols.

Debugger Files

Debugger files provide input to the debugger to define support for simulation or emulation of your program. The debugger supports all the executable file types produced by the linker (.DXXE, .SM, .OVL, .DLB). To simulate I/O, the debugger also supports the data file formats (.DAT) from the assembler.

The standard hexadecimal format for a SPORT data file is a single integer value per line. Hex numbers do not require the 0x prefix to indicate hexadecimal. A value can have any number of digits, but are read as follows

- The hexadecimal number is converted to binary
- The number of binary bits read in matches the word size set for the SPORT, which starts reading from the LSB. The SPORT then fills with zeros values that are shorter than the word size or, conversely, truncates any bits beyond the word size on the MSB end.

Example: a SPORT is set for 20-bit words and the data file contains hex numbers. The simulator converts these hex numbers to binary, then fills/truncates to match the SPORT word size.

Format References

In the following table, the number A5A5 is filled and the number 123456 is truncated

Table A-2. SPORT Data File Example

Hex Number	Binary Number	Truncated/Filled
A5A5A	1010 0101 1010 0101 1010	1010 0101 1010 0101 1010
FFFF1	1111 1111 1111 1111 0001	1111 1111 1111 1111 0001
A5A5	1010 0101 1010 0101	0000 1010 0101 1010 0101
5A5A5	0101 1010 0101 1010 0101	0101 1010 0101 1010 0101
11111	0001 0001 0001 0001 0001	0001 0001 0001 0001 0001
123456	0001 0010 0011 0100 0101 0110	0010 0011 0100 0101 0110

Format References

The following texts define industry standard file formats supported by VisualDSP:

- (1993) Executable and Linkable Format (ELF) V1.1 from the Portable Formats Specification V1.1, Tools Interface Standards Committee {available from <ftp://ftp.intel.com/pub/tis>}
- (1993) Debugging Information Format (DWARF) V1.1 from the Portable Formats Specification V1.1, UNIX International, Inc. {available from <ftp://ftp.intel.com/pub/tis>}