Introduction to the Jam™ Device Programming & Test Language Developer's Kit

Overview

What is Jam?

Jam is an interpreted language optimized for programming PLDs via the standard IEEE 1149.1 TAP Controller (JTAG). This method of programming is called In-System Programming (ISP). The syntax of the Jam language is similar to BASIC, with additional features and instructions to directly support a JTAG interface. The basic Jam flow is shown in Figure 1.

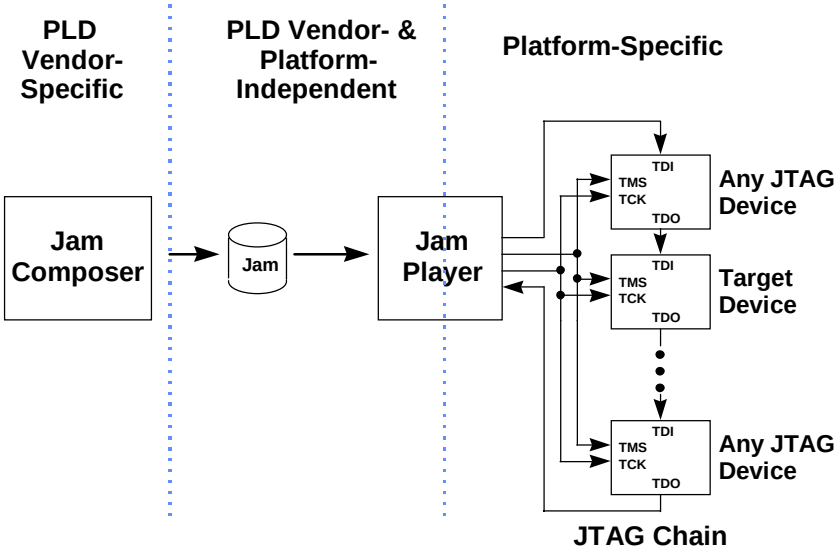

Figure 1. Basic Jam Flow

The Jam Composer is a software program, written by a PLD vendor, that creates a Jam File to program one of the vendor's PLDs. The Jam file contains both the programming data and the programming algorithm necessary to program a device (no other information is required). The Jam Player is software written by a device programmer or tester vendor that interprets the Jam file and manipulates the JTAG chain to actually program the device. Embedded system developers can also develop a Jam Player to program devices in their systems. Most of the source code required for the Jam Player is contained within this kit. The only software routines that must be provided to complete the Jam Player are those to access the JTAG chain.

Benefits of Jam

1. Vendor Independence

Any Jam Player can read any Jam file created by any Jam Composer. This vendor independence means that device programmer vendors do not need to add software to support new PLDs. See Figure 2.

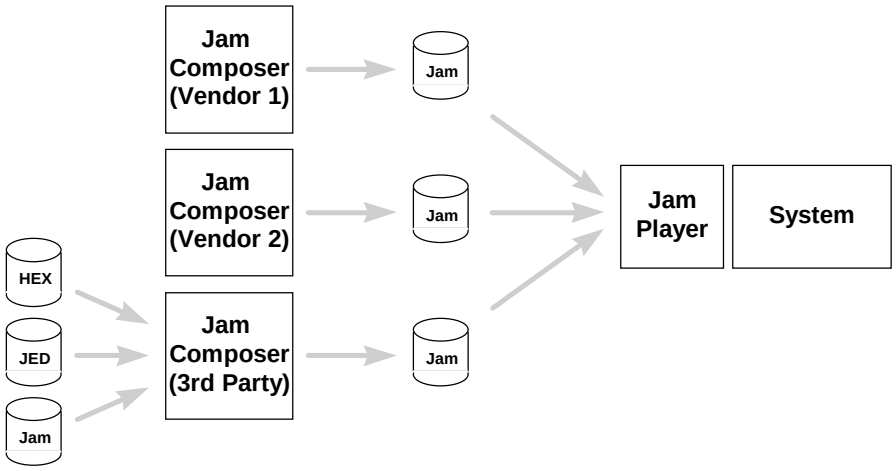

Figure 2. Vendor Independence

2. Platform Independence

Any Jam file is compatible with any Jam Player running on any platform. This platform independence frees PLD vendors from the burden of generating different programming files for different programming platforms. See Figure 3.

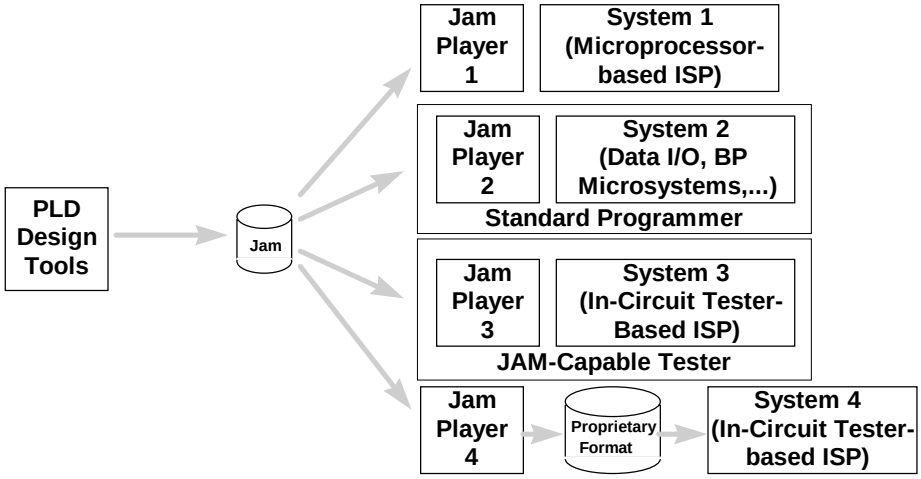

Figure 3. Platform Independence

3. Small File Size

Although Jam files are in ASCII format and contain both programming data and a programming algorithm, they are very compact. This compactness is critical for ISP via embedded processors because it allows the whole file to fit on a FLASH card. It is also important in tester environments because larger files take longer to process. Two key approaches are used to reduce the size of the Jam file. First, repetitive sequences of JTAG waveforms are described algorithmically; second, data compression is used for the programming

data. The Jam language supports looping, branching, and subroutine instructions such as CALL/RETURN and FOR/NEXT. It also supports run-length encoding and advanced data compression.

4. Faster Programming Times

Due to manufacturing tolerances, the programming pulse length required by PLDs varies over time. Because the Jam language supports branching instructions, the Jam Player can use shorter programming pulses if a particular device can use them. This capability is supported by reading the programming pulse length from the device prior to programming. Without this capability, the worst-case programming pulse length must be used for all devices. This pulse is often an order of magnitude longer than the typical programming pulse.

5. Supports Existing and Future Products

The Jam language is a completely generic programming language with no knowledge of any PLD architecture. Any programming algorithm can be described in a Jam file. This flexibility ensures that all existing and future devices can be supported by the Jam language.

What is the Jam Device Programming & Test Language Developer's Kit?

The Jam Device Programming & Test Language Developer's Kit (JDK) allows you to develop software and hardware that supports the Jam Language. It contains the documentation and software necessary to develop both Jam Composer and Jam Player software.

Package Contents

This package contains the following items:

- Jam Device Programming & Test Language Developer's Kit CD-ROM Version 1.0
- ByteBlaster parallel port download cable

System Requirements

The minimum system requirements for successful use of the JDK are as follows:

- 133 MHhz Pentium PC
- 24 Mbytes RAM
- 10 Mbytes Disk Space
- Windows 95 or Windows NT 4.0
- Parallel Port

Installation

It is not necessary to install the JDK onto your hard disk. Files may be used directly from the CD-ROM. However, if you wish to install the contents of the JDK CD-ROM onto your hard disk, type the following command at a system prompt:

```
xcopy f:\ c:\jam /s <Enter>
```

The command above assumes that your CD-ROM is drive `f:` and you wish to install in `c:\jam`.

To install the ByteBlaster parallel port download cable, simply attach it to an available parallel port. For more information, refer to `\doc\byteblaster.pdf` on the JDK CD-ROM.

CD ROM Contents

Note: All documentation on the CDROM was created in Adobe Acrobat version 3.0. You can download the latest version of the Adobe Reader from `http://www.adobe.com/prodindex/acrobat`

Directory	Filename	Description
\doc	jamplayer.pdf	Jam Player User Guide. Contains the information necessary to create a Jam Player from the source code provided.
	jamlanguage.pdf	Jam Language Specification. Complete description of the Jam Device Programming & Test Language.
	byteblaster.pdf	ByteBlaster parallel port download cable Data Sheet.
	intro.pdf	This document.
\exe	jam.exe	PC Win95/Windows NT 4.0 Jam Player executable. This program was built from source code in `\source\jamplayer` and works with the ByteBlaster parallel port download cable. `jam.exe` is used to test Jam files.
	jamdata.exe	PC Win95/Windows NT 4.0 Jam Data executable. This program was built from the source code in `\source\jamdata`. Use this program to generate compressed data for inclusion in Jam files.
\source\jamplayer	*.c *.h	Source code for `jam.exe`. This source code is needed to create a Jam Player.
	makefile.mak	Makefile for `jam.exe`.
\source\jamdata	jamdata.c	Source code for `jamdata.exe`. This source code is needed to create a Jam Composer.
\examples\isp_demo_board	*.jam	Example Jam files for use with the Altera ISP Demo Board.
\examples\jam_demo_board	*.jam	Example Jam files for use with the Altera Jam Demo Board. Contact Altera via e-mail at `jam@altera.com` for information about obtaining a Jam Demo Board.
\examples\language_spec	*.jam	Example Jam files shown in the Jam Language Specification.
\examples\how_to_use_jam	*.jam	Example Jam files that show basic use of the Jam Language. These files do not use the JTAG interface.
	*.out	Output from running `jam.exe` on each of the Jam files in this directory. There is one `.out` file for each `.jam` file.

Technical Support

For technical support, send e-mail to `jam@altera.com`.