JTAG-Booster for

Analog Devices Blackfin

Forth-Systeme GmbH

P.O: Box 1103
Kueferstrasse 8
Tel. +49 (7667) 908-0
sales@fsforth.de

- D-79200 Breisach, Germany
- D-79206 Breisach, Germany
- Fax +49 (7667) 908-200
- http://www.fsforth.de

FS FORTH-SYSTEME GmbH
Postfach 1103, D-79200 Breisach, Germany

Release of Document: February 28, 2005
Author: Dieter Fögele
Filename: JTAG_ADSP-BLACKFINa.doc
Program Version: 4.xx

Table of Contents

1. General 5
   1.1. Ordering Information 6
   1.2. System Requirements 6
   1.3. Contents of Distribution Disk 7
   1.4. Connecting your PC to the target system 8
   1.5. First Example with Analog Devices ADSP-BF531/BF532/BF533 10
   1.6. First Example with Analog Devices ADSP-BF561 12
   1.7. Trouble Shooting 14
   1.8. Error Messages 15
   1.9. Initialization file JTBFxxx.INI 20
   1.10. Supported flash devices 31

2. JTBFxxx Parameter Description 32
   2.1. Program a Flash Device 35
   2.2. Read a Flash Device to file 40
   2.3. Verify a Flash Device with file 42
   2.4. Dump target memory 44
   2.5. Program a Serial Device (I²C/SPI/MicroWire) 46
   2.6. Read a Serial Device to file (I²C/SPI/MicroWire) 49
   2.7. Verify a Serial Device with file (I²C/SPI/MicroWire) 51
   2.8. Dump a Serial Device (I²C/SPI/MicroWire) 53
   2.9. Toggle CPU pins 55
   2.10. Polling CPU pins 56
   2.11. Polling CPU pins while the CPU is running 57
   2.12. Show status of all CPU pins while the CPU is running 58

3. Implementation Information 60
   3.1. Implementation Information Analog Devices ADSP-BF531/BF532/BF533 61
   3.2. Implementation Information Analog Devices ADSP-BF561 62

4. Converter Program HEX2BIN.EXE 63

5. Support for Windows NT, Windows 2000 and Windows XP 65
   5.1. Installation on a clean system 65
   5.2. Installation with already installed version 5.x/6.x of Kithara 65

5.3. Installation with already installed version 4.x of Kithara 65
5.4. De-Installation version 5.x/6.x: 66

1. General

The programs JTBF531.EXE and JTBF561.EXE use the IEEE 1149.1 JTAG port of the Analog Devices Blackfin microcontrollers in conjunction with the small JTAG-Booster:

- to program data into flash memory
- to verify and read the contents of a flash memory
- to make a memory dump
- to access a serial device (I²C/SPI/MicroWire)
- to test CPU signals

All functions are done without any piece of software running in the target. No firmware or BIOS must be written. Bootstrap software may be downloaded to initially unprogrammed memories.

As this tool uses boundary scan, it is extremely simple and very powerful. It assists you in bringing-up new hardware. Even if there are essential bugs in the hardware (i.e. RAM not reliable working, soldering problems with the BGA package), in many cases you are able to load small test programs into flash, which helps you to analyze hardware problems. Or if you have a flash memory, which is not connected correctly to the, we can support you with a special adapted version of the JTAG-Booster.

The JTAG-BOOSTER' s software is highly optimized to the JTAG chain of a specific target CPU. To give support for all processors of the Analog Devices Blackfin family, there are two different programs on the distribution disk:

- JTBF531.EXE : Tool for Analog Devices ADSP-BF531/BF532/BF533
- JTBF561.EXE : Tool for Analog Devices ADSP-BF561

Please contact us, if you need support for other members of the Analog Devices Blackfin family.

For latest documentation please refer to the file README.TXT on the distribution disk.

1.1. Ordering Information

The following related products are available

- 9062 JTAG-Booster Analog Devices Blackfin
  Analog Devices ADSP-BF531/BF532/BF533
  DOS/Win9x/WinNT/Win2000/WinXP,
  delivered with adapter type 285
  and cable with single strands type TK02206

1.2. System Requirements

To successfully run this tool the following requirements must be met:

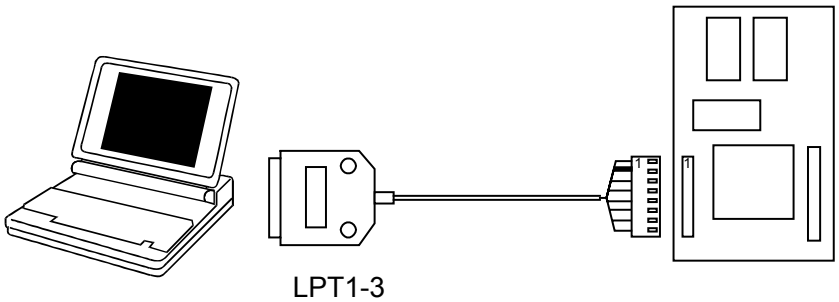
- MSDOS, WIN3.x, WIN9x, WinNT, Win2000 or WindowsXP
  (WinNT/Win2000/WindowsXP is supported with an additional tool, see chapter 5 “Support for Windows NT, Windows 2000 and Windows XP”)
- Intel 80386 or higher
- 205 kByte of free DOS memory
- Parallel Port

1.3. Contents of Distribution Disk

- JTBF531.EXE
  JTBF531.OVL — Tool for Analog Devices ADSP-BF531/BF532/BF533
- JTBF531.INI — Template configuration file for Analog Devices ADSP-BF531/BF532/BF533. See chapter 1.9 "Initialization file JTBFxxx.INI"
- JTBF561.EXE
  JTBF561.OVL — Tool for Analog Devices ADSP-BF561
- JTBF561.INI — Template configuration file for Analog Devices ADSP-BF561. See chapter 1.9 "Initialization file JTBFxxx.INI"
- WinNT.zip — Support for Windows NT, Windows 2000 and Windows XP. See chapter 5 "Support for Windows NT, Windows 2000 and Windows XP"
- JTAG_V4xx_FLASHES.pdf — List of all supported Flash devices
- README.txt — Release notes, new features, known problems

1.4. Connecting your PC to the target system

The JTAG-Booster can be plugged into standard parallel ports (LPT1-3) with a DB25-Connector.



LPT1-3

The target end of the cable has the following reference:

1	2*	3	4	5	6	7	8
TCK	GND	TMS	TRST#	NC	TDI	TDO	+3.3V

*PIN 2 can be detected by the thick cable.

To connect your design to the JTAG-BOOSTER you need a single row berg connector with a spacing of 2.54mm on your PCB. The names refer to the target: Pin 7 is the target's TDO pin and is connected to the JTAG-Booster's TDI pin.

The 3.3V version of the JTAG-Booster (FS part number 285) is delivered together with this package. Don't use the 5V version of the JTAG-Booster (FS part number 227) with a 3.3V target. **Don't apply 5V to the 3.3V version of the JTAG-Booster!**

Your target must be able to power the JTAG-Booster, it draws about 100mA.

Before you start the program, the JTAG-BOOSTER must be plugged to a parallel interface of your PC and to the 8 pin JTAG connector on the target.

The utility is started with the general command line format: JTAGxxx

JTBFxxx /function [filename] [/option_1] ... [/option_n].

Note that the function must be the first argument followed (if needed) by the filename.

If you want to cancel execution of JTBFxxx, press CTRL-Break-Key.

On any error the program aborts with an MSDOS error level of one.

1.5. First Example with Analog Devices ADSP-BF531/BF532/BF533

In the following simple example it is assumed that the JTAG-Booster is connected to LPT1 of your PC and target power is on.

Typing

```
JTBF531  /P  MYAPP.BIN
```

at the DOS prompt results in the following output:

```
JTBF531 --- JTAG utility for Analog Devices ADSP-BF531/BF532/BF533
Copyright © FS FORTH-SYSTEME GmbH, Breisach
Version 4.xx of mm/dd/yyyy

(1)   Configuration loaded from file JTBF531.INI
(2)   Target: Generic Target
(3)   Using LPT at I/O-address 0378h
(4)   JTAG Adapter detected

(5)   1 Device detected in JTAG chain
        Device 0: IDCODE=227A50CB  Analog Devices ADSP-BF531/BF532/BF533, Revision 2
(6)   Sum of instruction register bits  : 5
(7)   CPU position                      : 0
(8)   Instruction register offset       : 0
(9)   Length of boundary scan reg       : 197

      Looking for a flash device with known JEDEC ID...
(10)  STM 29W320B, 3.3V, Boot Block Bottom  detected
(11)  Bus size is 16 Bit
(12)  Erasing Flash-EPROM Block #:0  1  2  3
      Programming File MYAPP.BIN
      65536 Bytes programmed successfully

      Erase Time          :      0.8 sec
      Programming Time    :     27.0 sec
```

(1) The initialization file JTBF531.INI was found in the current directory.

(2) The target identification line of the initialization file is printed here.

(3) The resulting I/O-address of the parallel port is printed here. With WinNT/Win2000/WinXP you must specify the option /LPT2 to access to the standard address 378h.

(4) A JTAG-Booster is found on the parallel port

(5) The JTAG chain is analyzed. There may be several parts in the JTAG chain. The chain is analyzed and all parts except the Analog Devices ADSP-BF531/BF532/BF533 are switched to bypass mode.

(6) The length of all instruction registers in the JTAG chain are added.

(7) The position of the Analog Devices ADSP-BF531/BF532/BF533 in the JTAG chain is assumed to be zero, if not specified in the command line (see option /CPUPOS=).

(8) The position of the JTAG instruction register of the Analog Devices ADSP-BF531/BF532/BF533 is assumed to be zero, if not specified in the command line (see option /IROFFS=).

(9) The real length of the boundary scan register is displayed here and compared with the boundary scan register length of a Analog Devices ADSP-BF531/BF532/BF533.

(10) A flash STM 29W320B selected with AMS0# was found.

(11) The resulting data bus size is printed here.

(12) In this example 4 blocks must be erased.

1.6. First Example with Analog Devices ADSP-BF561

In the following simple example it is assumed that the JTAG-Booster is connected to LPT1 of your PC and target power is on.

Typing

```
JTBF561  /P  MYAPP.BIN
```

at the DOS prompt results in the following output:

```
JTBF561 --- JTAG utility for Analog Devices ADSP-BF561
Copyright © FS FORTH-SYSTEME GmbH, Breisach
Version 4.xx of mm/dd/yyyy

(1)   Configuration loaded from file JTBF561.INI
(2)   Target: Generic Target
(3)   Using LPT at I/O-address 0378h
(4)   JTAG Adapter detected

(5)   1 Device detected in JTAG chain
        Device 0: IDCODE=027BB0CB  Analog Devices ADSP-BF561, Revision 0
(6)   Sum of instruction register bits  : 3
(7)   CPU position                      : 0
(8)   Instruction register offset       : 0
(9)   Length of boundary scan reg       : 355

      Looking for a flash device with known JEDEC ID...
(10)  STM 29W320B, 3.3V, Boot Block Bottom  detected
(11)  Bus size is 16 Bit
(12)  Erasing Flash-EPROM Block #:0  1  2  3
      Programming File MYAPP.BIN
      65536 Bytes programmed successfully

      Erase Time          :      0.8 sec
      Programming Time    :     48.7 sec
```

(1) The initialization file JTBF561.INI was found in the current directory.

(2) The target identification line of the initialization file is printed here.

(3) The resulting I/O-address of the parallel port is printed here.

(4) A JTAG-Booster is found on the parallel port

(5) The JTAG chain is analyzed. There may be several parts in the JTAG chain. The chain is analyzed and all parts except the Analog Devices ADSP-BF561 are switched to bypass mode.

(6) The length of all instruction registers in the JTAG chain are added.

(7) The position of the Analog Devices ADSP-BF561 in the JTAG chain is assumed to be zero, if not specified in the command line (see option /CPUPOS=).

(8) The position of the JTAG instruction register of the Analog Devices ADSP-BF561 is assumed to be zero, if not specified in the command line (see option /IROFFS=).

(9) The real length of the boundary scan register is displayed here and compared with the boundary scan register length of a Analog Devices ADSP-BF561.

(10) A flash STM 29W320B selected with AMS0# was found.

(11) The resulting data bus size is printed here.

(12) In this example 4 blocks must be erased.

1.7. Trouble Shooting

Avoid long distances between your Host-PC and the target. If you are using standard parallel extension cable, the JTAG-BOOSTER may not work. Don't use Dongles between the parallel port and the JTAG-BOOSTER.

Switch off all special modes of your printer port (EPP, ECP, ...) in the BIOS setup. Only standard parallel port (SPP) mode is allowed.

If there are problems with autodetection of the flash devices use the /DEVICE= option. To speed up autodetection specify one of the options /8BIT /16BIT or /32BIT.

Don't use hardware protected flash memories.

The used chip selects must be defined as output and inactive in the initialization file (see chapter 1.9 “Initialization file JTBFxxx.INI”). Also the address bits must be defined as output.

Use the option /NOWRSETUP to speed up flash programming.

If you have problems using the option /CFI (Common Flash Interface) use the option /CFIDEBUG instead and redirect the program's output into a file. Sending us this file helps in analyzing problems.

1.8. Error Messages

- **80386 or greater required**
  The JTAG-BOOSTER does not work on a 8088/8086 or a 80286 platform.

- **Cable not connected or target power fail**
  The JTAG-Booster (or one of the simple Parallel Port JTAG adapters selected with the options /LATTICE /WIGGLER /PLS) wasn't found. Please check connection to parallel port and connection to target. Check target power. Check the command line options. Check your BIOS-Setup. If you are using this program with WinNT, Win2000 or WinXP you must specify /LPT2 or /LPT-BASE=378 to get access to the standard printer port.

- **Can't open x:\yyy\zzz\JTBFxxx.OVL**
  The overlay file JTBFxxx.OVL must be in the same directory as JTBFxxx.EXE.

- **Configuration file XYZ not found.**
  The file specified with the option /INI= wasn't found.

- **Device offset out of range**
  The value specified with the option /OFFSET= is greater than the size of the detected flash device.

- **Disk full**
  Writing a output file was aborted as a result of missing disk space.

- **Do not specify option /NOCS with any other chip select**
  There is a conflict in the command line.

- **Do not specify option /BYTE-MODE. *Flash device* does not have a byte mode pin.**
  The flash device specified with the option /DEVICE= does not support switching between 16 (or 32) bit mode and 8 bit mode. In practice it does not have a pin with the name BYTE#

- **Error creating file:**
  The output file could not be opened. Please check free disk space or write protection.

- **Error: *Pin-Name* is an output only pin**
  The specified pin cannot be sampled. Check the command line. Check the initialization file.

- **Error: *Pin-Name* is an input only pin**
  The specified pin cannot be activated. Check the command line. Check the initialization file.

- **Error: *Pin-Name* may not be read back**
  The specified pin can be switched to tristate, but cannot be read back. Check the command line.

- **illegal function:**
  The first parameter of the command line must be a valid function. See chapter 2 “JTBFxxx Parameter Description” for a list of supported functions.

- **illegal number:**
  The specified number couldn’t be interpret as a valid number. Check the relevant number base.

- **illegal option:**
  See chapter 2 “JTBFxxx Parameter Description” for a list of supported options.

- **illegal Pin Type:**
  The name specified with the option /PIN= must be one of the list of chapter 1.9 ”Initialization file JTBFxxx.INI”

- **illegal Flash Type:**
  The name specified with the option /DEVICE= must be one of the list of chapter 1.10 ”Supported flash devices”.

- **Input file not found:**
  The specified file cannot be found

- **Input file is empty:**
  Files with zero length are not accepted

- **" " is undefined**
  Please check the syntax in your configuration file. (See chapter 1.9 “Initialization file JTBFxxx.INI”).

- **LPTx not installed**
  The LPT port specified with /LPTx cannot be found. Please check the LPT port or specify a installed LPT port. Check your BIOS setup. If you are using this program with WinNT, Win2000 or WinXP you 1st must install the WinNT support package as described in chapter 5 "Support for Windows NT, Windows 2000 and Windows XP"

- **missing filename**
  Most functions need a filename as second parameter.

- **missing option /SERCLK=**
  Some functions need the option /SERCLK= to be defined.

- **missing option /SERDAT=**
  Some functions need the option /SERDAT= or the options /SERDATO= and /SERDATI= to be defined.

- **missing option /SERCS=**
  Some functions need the option /SERCS= if the option /SPI or the option /MWIRE is specified.

- **missing option /LENGTH=**
  Some functions need the option /LENGTH= to be defined.

- **missing option /PIN=**
  Some functions need the option /PIN= to be defined.

- **More than 9 devices in the JTAG chain or TDO pin stuck at low level**
  The JTAG chain is limited to 9 parts. Check target power. Check the target’s TDO pin.

- **No devices found in JTAG chain or TDO pin stuck at high level**
  A stream of 32 high bits was detected on the pin TDO. TDO may stuck at high level. Check the connection to your target. Check the target power. Check the target’s TDO pin.

- **Option /CPUPOS= out of range**
  The number specified with the option /CPUPOS= must be less or equal to the number of parts minus 1.

- **Option /IROFFS= out of range**
  Please specify a smaller value

- **Part at specified position is not a Analog Devices Blackfin**
  The option /CPUPOS= points to a part not a Analog Devices Blackfin

- **Pins specified with /SERCLK= and /SERDAT= must have different control cells**
  The pin specified with the option /SERDAT= must be able to be switched to high impedance while the pin specified with option /SERCLK= is an active output. See chapter 1.9 “Initialization file JTBFxxx.INI”.

- **Pins specified with /SERCLK= and /SERDATI= must have different control cells**
  The pin specified with the option /SERDATI= must be able to be switched to high impedance while the pin specified with option /SERCLK= is an active output. See chapter 1.9 “Initialization file JTBFxxx.INI”.

- **Pins specified with /SERDATO= and /SERDATI= must have different control cells**
  The pin specified with the option /SERDATI= must be able to be switched to high impedance while the pin specified with option /SERDATO= is an active output. See chapter 1.9 “Initialization file JTBFxxx.INI”.

- **Specify only one of these options:**
  Some options are exclusive (i.e. /8BIT and /16BIT). Don't mix them.

- **Sum of instruction register bits to low. Should be at least 5 bits for a Analog Devices Blackfin**
  The sum of all instruction register bits in the JTAG chain does not fit to the Analog Devices Blackfin. Check the target connection. Check the target CPU type. Check the settings for /IROFFS= and /CPUPOS= , if there are several parts in the JTAG chain.

- **Target no longer connected**
  There is a cyclic check of the JTAG chain. Check target power. Check target connection.

- **There are unknown parts in the JTAG chain. Please use the option /IROFFS= to specify the instr. reg. offset of the CPU.**
  If there are unknown parts in the JTAG chain, the program isn't able to determine the logical position of the CPU's instruction register.

- **There is no Analog Devices Blackfin in the JTAG chain**
  No Analog Devices Blackfin was found in the JTAG chain. Check the target power. Try with option /DRIVER=4 again.

- **Value of option /FILE-OFFSET out of range**
  The value of the option /FILE-OFFSET= points behind end of file.

- **wrong driver #**
  The value specified with the option /DRIVER= is out of range.

- **Wrong Flash Identifier (xxxx)**
  No valid identifier found. Check the specified chip select signal and the bus width. Try with the option /DEVICE= . Use the option /8BIT or /16BIT or /32BIT to specify the correct data bus size.

- **Wrong length of boundary scan register. Should be 197 for a Analog Devices ADSP-BF531/BF532/BF533. (Should be 355 for a Analog Devices ADSP-BF561.)**
  The length of the boundary scan register of the selected part (if there are more than one in the chain) does not fit to the Analog Devices Blackfin. Check the target connection. Check the target CPU type. Check the settings for /IROFFS= and /CPUPOS= , if there are several parts in the JTAG chain.

1.9. Initialization file JTBFxxx.INI

This file is used to define the default direction and level of all CPU signals. This file **must be carefully adapted** to your design with the Analog Devices Blackfin. The Target-Entry is used to identify your design which is displayed with most commands.

When the program JTBFxxx.EXE is started it scans the current directory for an existing initialization file named JTBFxxx.INI. If no entry is found the default values are used. You may also specify the initialization file with the option /INI= . If the specified file isn't found, the program aborts with an error message.

The CPU pins can also be used with the functions /BLINK (chapter 2.9), /PIN? (chapter 2.10) and /SAMPLE (chapter 2.11) to test the signals on your design.

The sample file below represents the values which are used for default initialization when no initialization file could be found in the current directory and no initialization file is specified with the option /INI=.

Changes to the structure of the file could result in errors. Remarks can be added by using //.

Sample File JTBF531.INI:

```
// Description file for NetSilicon NS9750/ns9775
// Description file for Analog Devices ADSP-BF531/BF532/BF533
Target: Generic Target, 2005/02/14
// All chip select signals are set to output and inactive.
// All signals should be defined. Undefined signals are set to their defaults.
// Pin names are defined in upper case.
// Low active signal are signed with a trailing #.

// Group 196: All pins in this group must be set to the same direction
//                  This pins are bidirectional
//                  During flash programming these pins are switched between
//                  input/inactive and output/active.
//                  For Flash programming and other memory accesses
//                  these pins should be set to Input
DATA0               Inp     // Data Bus
DATA1               Inp     //
DATA2               Inp     //
DATA3               Inp     //
DATA4               Inp     //
DATA5               Inp     //
DATA6               Inp     //
DATA7               Inp     //
DATA8               Inp     //
DATA9               Inp     //
DATA10              Inp     //
DATA11              Inp     //
DATA12              Inp     //
DATA13              Inp     //
DATA14              Inp     //
DATA15              Inp     //

// Group 17: All pins in this group must be set to the same direction
//                  These pins are tristateable outputs but can not be read back
//                  For Flash programming and other memory accesses
//                  these pins must be configured as output
ADDR1               Out,Lo  // Address Bus
ADDR2               Out,Lo  //
ADDR3               Out,Lo  //
ADDR4               Out,Lo  //
ADDR5               Out,Lo  //
ADDR6               Out,Lo  //
```

```
ADDR7           Out,Lo  //
ADDR8           Out,Lo  //
ADDR9           Out,Lo  //
ADDR10          Out,Lo  //
ADDR11          Out,Lo  //
ADDR12          Out,Lo  //
ADDR13          Out,Lo  //
ADDR14          Out,Lo  //
ADDR15          Out,Lo  //
ADDR16          Out,Lo  //
ADDR15          Out,Lo  //
ADDR18          Out,Lo  //
ADDR19          Out,Lo  //
ABE0#           Out,Lo  // SDQM0, Byte Enable
ABE1#           Out,Lo  // SDQM1, Byte Enable

// Group 27: All pins in this group must be set to the same direction
//              These pins are tristateable outputs but can not be read back
//              For Flash programming and other memory accesses
//              these pins must be configured as output
AWE#            Out,Hi  // Write Enable
ARE#            Out,Hi  // Read Enable
AOE#            Out,Hi  // Output Enable
AMS0#           Out,Hi  // Bank Select
AMS1#           Out,Hi  // Bank Select
AMS2#           Out,Hi  // Bank Select
AMS3#           Out,Hi  // Bank Select

// Group 39: All pins in this group must be set to the same direction
//              These pins are tristateable outputs but can not be read back
CLKOUT          Out,Lo  // SDRAM Clock Output
SCKE            Out,Hi  // SDRAM Clock Enable
SMS#            Out,Hi  // SDRAM Bank Select
SRAS#           Out,Hi  // SDRAM Row Address Strobe
SCAS#           Out,Hi  // SDRAM Column Address Strobe
SWE#            Out,Hi  // SDRAM Write Enable
SA10            Out,Lo  // SDRAM A10 Pin
```

```
// The following pins are complete bidirectional pins.
// The direction of each pin can be set independent of the other pins.
// Each pin can be used as input.
PPI0            Inp     //
PPI1            Inp     //
PPI2            Inp     //
PPI3            Inp     //
PF0             Inp     // SPISS#
PF1             Inp     // SPISEL1/TMRCLK
PF2             Inp     // SPISEL2
PF3             Inp     // SPISEL3/PPI_FS3
PF4             Inp     // SPISEL4/PPI15
PF5             Inp     // SPISEL5/PPI14
PF6             Inp     // SPISEL6/PPI13
PF7             Inp     // SPISEL7/PPI12
PF8             Inp     // PPI11
PF9             Inp     // PPI10
PF10            Inp     // PPI9
PF11            Inp     // PPI8
PF12            Inp     // PPI7
PF13            Inp     // PPI6
PF14            Inp     // PPI5
PF15            Inp     // PPI4
RSCLK0          Inp     // SPORT0 Receive Serial Clock
RFS0            Inp     // SPORT0 Receive Frame Sync
TSCLK0          Inp     // SPORT0 Transmit Serial Clock
TFS0            Inp     // SPORT0 Transmit Frame Sync
RSCLK1          Inp     // SPORT1 Receive Serial Clock
RFS1            Inp     // SPORT1 Receive Frame Sync
TSCLK1          Inp     // SPORT1 Transmit Serial Clock
TFS1            Inp     // SPORT1 Transmit Frame Sync
TMR0            Inp     // Timer 0
TMR1            Inp     // PPI_FS2
TMR2            Inp     // PPI_FS2
SCK             Inp     // SPI Clock
MISO            Inp     // SPI Master In Slave Out
MOSI            Inp     // SPI Master Out Slave In
```

```
// The following pins are tristateable outputs.
// These pins are tristateable outputs but can not be read back.
// Each pin can be disabled independent of the other pins.
DT0PRI          Inp     // SPORT0 Transmit Data Primary
DT0SEC          Inp     // SPORT0 Transmit Data Secondary
DT1PRI          Inp     // SPORT1 Transmit Data Primary
DT1SEC          Inp     // SPORT1 Transmit Data Secondary

// The following pins are output only pins.
// Setting to input (tristate) one of these pins results in an error.
BG#             Out,Hi  // Bus Grant
BGH#            Out,Hi  // Bus Grant Hang
TX              Out,Hi  // UART Transmit

// The following pins are input only.
// Setting to output of one of these pins results in an error.
// Declaration of the direction of these pins is optional.
ARDY            Inp     // Hardware Ready Control
BR#             Inp     // Bus Request
RESET#          Inp     // Reset Input
NMI             Inp     // Non Maskable Interrupt
PPI_CLK         Inp     // PPI Clock Input
DR0PRI          Inp     // SPORT0 Receive Data Primary
DR0SEC          Inp     // SPORT0 Receive Data Secondary
DR1PRI          Inp     // SPORT1 Receive Data Primary
DR1SEC          Inp     // SPORT1 Receive Data Secondary
RX              Inp     // UART Receive
BMODE0          Inp     // Boot Mode Strap
BMODE1          Inp     // Boot Mode Strap
TEST            Inp     // ???
```

Sample File JTBF561.INI:

```
// Description file for Analog Devices ADSP-BF561
Target: Generic Target, 2005/02/22
// All chip select signals are set to output and inactive.
// All signals should be defined. Undefined signals are set to their defaults.
// Pin names are defined in upper case.
// Low active signal are signed with a trailing #.

// Group 221: All pins in this group must be set to the same direction
//                  This pins are bidirectional
//                  During flash programming these pins are switched between
//                  input/inactive and output/active.
//                  For Flash programming and other memory accesses
//                  these pins should be set to Input
DATA0               Inp     // Data Bus
DATA1               Inp     //
DATA2               Inp     //
DATA3               Inp     //
DATA4               Inp     //
DATA5               Inp     //
DATA6               Inp     //
DATA7               Inp     //
DATA8               Inp     //
DATA9               Inp     //
DATA10              Inp     //
DATA11              Inp     //
DATA12              Inp     //
DATA13              Inp     //
DATA14              Inp     //
DATA15              Inp     //

// Group 254: All pins in this group must be set to the same direction
//                  This pins are bidirectional
//                  During flash programming these pins are switched between
//                  input/inactive and output/active.
//                  For Flash programming and other memory accesses
//                  these pins should be set to Input
DATA16              Inp     // Data Bus
DATA17              Inp     //
DATA18              Inp     //
DATA19              Inp     //
DATA20              Inp     //
```

```
DATA21          Inp     //
DATA22          Inp     //
DATA23          Inp     //
DATA24          Inp     //
DATA25          Inp     //
DATA26          Inp     //
DATA27          Inp     //
DATA28          Inp     //
DATA29          Inp     //
DATA30          Inp     //
DATA31          Inp     //

// Group 219: All pins in this group must be set to the same direction
//              These pins are tristateable outputs but can not be read back
//              For Flash programming and other memory accesses
//              these pins must be configured as output
ADDR2           Out,Lo  // Address Bus
ADDR3           Out,Lo  //
ADDR4           Out,Lo  //
ADDR5           Out,Lo  //
ADDR6           Out,Lo  //
ADDR7           Out,Lo  //
ADDR8           Out,Lo  //
ABE0#           Out,Lo  // SDQM0, Byte Enable
ABE1#           Out,Lo  // SDQM1, Byte Enable
ABE2#           Out,Lo  // SDQM2, Byte Enable
ABE3#           Out,Lo  // SDQM3, Byte Enable

// Group 175: All pins in this group must be set to the same direction
//              These pins are tristateable outputs but can not be read back
//              For Flash programming and other memory accesses
//              these pins must be configured as output
ADDR9           Out,Lo  //
ADDR10          Out,Lo  //
ADDR11          Out,Lo  //
ADDR12          Out,Lo  //
ADDR13          Out,Lo  //
ADDR14          Out,Lo  //
ADDR15          Out,Lo  //
ADDR16          Out,Lo  //
ADDR15          Out,Lo  //
ADDR18          Out,Lo  //
ADDR19          Out,Lo  //
```

```
ADDR20          Out,Lo  //
ADDR21          Out,Lo  //
ADDR22          Out,Lo  //
ADDR23          Out,Lo  //
ADDR24          Out,Lo  //
ADDR25          Out,Lo  //

// Group 189: All pins in this group must be set to the same direction
//              These pins are tristateable outputs but can not be read back
//              For Flash programming and other memory accesses
//              these pins must be configured as output
AWE#            Out,Hi  // Write Enable
ARE#            Out,Hi  // Read Enable
AOE#            Out,Hi  // Output Enable
AMS0#           Out,Hi  // Bank Select
AMS1#           Out,Hi  // Bank Select
AMS2#           Out,Hi  // Bank Select
AMS3#           Out,Hi  // Bank Select

// Group 194: All pins in this group must be set to the same direction
//              These pins are tristateable outputs but can not be read back
SCLK0           Out,Lo  // SDRAM Clock Output
SCLK1           Out,Lo  //
SCKE            Out,Hi  // SDRAM Clock Enable
SMS0#           Out,Hi  // SDRAM Bank Select
SMS1#           Out,Hi  //
SMS2#           Out,Hi  //
SMS3#           Out,Hi  //
SRAS#           Out,Hi  // SDRAM Row Address Strobe
SCAS#           Out,Hi  // SDRAM Column Address Strobe
SWE#            Out,Hi  // SDRAM Write Enable
SA10            Out,Lo  // SDRAM A10 Pin

// The following pins are complete bidirectional pins.
// The direction of each pin can be set independent of the other pins.
// Each pin can be used as input.
PF0             Inp     // SPISS#/TMR0
PF1             Inp     // SPISEL1/TMR1
PF2             Inp     // SPISEL2/TMR2
PF3             Inp     // SPISEL3/TMR3
PF4             Inp     // SPISEL4/TMR4
PF5             Inp     // SPISEL5/TMR5
PF6             Inp     // SPISEL6/TMR6
```

```
PF7             Inp     // SPISEL7/TMR7
PF8             Inp     //
PF9             Inp     //
PF10            Inp     //
PF11            Inp     //
PF12            Inp     //
PF13            Inp     //
PF14            Inp     //
PF15            Inp     // EXTCLK

RSCLK0          Inp     // PF28
RFS0            Inp     // PF19
DR0PRI          Inp     //
DR0SEC          Inp     // PF20
TSCLK0          Inp     // PF29
TFS0            Inp     // PF16
DT0PRI          Inp     // PF18
DT0SEC          Inp     // PF17
RSCLK1          Inp     // PF30
RFS1            Inp     // PF24
DR1PRI          Inp     //
DR1SEC          Inp     // PF25
TSCLK1          Inp     // PF31
TFS1            Inp     // PF21
DT1PRI          Inp     // PF23
DT1SEC          Inp     // PF22
TX              Inp     // PF26
RX              Inp     // PF27

PPI0D0          Inp     //
PPI0D1          Inp     //
PPI0D2          Inp     //
PPI0D3          Inp     //
PPI0D4          Inp     //
PPI0D5          Inp     //
PPI0D6          Inp     //
PPI0D7          Inp     //
PPI0D8          Inp     // PF32
PPI0D9          Inp     // PF33
PPI0D10         Inp     // PF34
PPI0D11         Inp     // PF35
PPI0D12         Inp     // PF36
PPI0D13         Inp     // PF37
```

```
PPI0D14         Inp     // PF38
PPI0D15         Inp     // PF39
PPI0SYNC1       Inp     // TMR8
PPI0SYNC2       Inp     // TMR9
PPI0SYNC3       Inp     //

PPI1D0          Inp     //
PPI1D1          Inp     //
PPI1D2          Inp     //
PPI1D3          Inp     //
PPI1D4          Inp     //
PPI1D5          Inp     //
PPI1D6          Inp     //
PPI1D7          Inp     //
PPI1D8          Inp     // PF40
PPI1D9          Inp     // PF41
PPI1D10         Inp     // PF42
PPI1D11         Inp     // PF43
PPI1D12         Inp     // PF44
PPI1D13         Inp     // PF45
PPI1D14         Inp     // PF46
PPI1D15         Inp     // PF47
PPI1SYNC1       Inp     // TMR10
PPI1SYNC2       Inp     // TMR11
PPI1SYNC3       Inp     //

SCK             Inp     // SPI Clock
MISO            Inp     // SPI Master In Slave Out
MOSI            Inp     // SPI Master Out Slave In

// The following pins are output only pins.
// Setting to input (tristate) one of these pins results in an error.
BG#             Out,Hi  // Bus Grant
BGH#            Out,Hi  // Bus Grant Hang
SLEEP           Out,Lo  //
```

```
// The following pins are input only.
// Setting to output of one of these pins results in an error.
// Declaration of the direction of these pins is optional.
ARDY            Inp     // Hardware Ready Control
BR#             Inp     // Bus Request
RESET#          Inp     // Reset Input
BY_PASS         Inp     // PLL Bypass Control
NMI0            Inp     // Non Maskable Interrupt Core A
NMI1            Inp     // Non Maskable Interrupt Core B
PPI0CLK         Inp     //
PPI1CLK         Inp     //
BMODE0          Inp     // Boot Mode Strap
BMODE1          Inp     // Boot Mode Strap
TEST            Inp     // ???
```

1.10. Supported flash devices

Type JTBFxxx /LIST [optionlist]

to get a online list of all flash types which could be used with the /DEVICE= option. In addition newer flash devices are supported by using the option /CFI.

See separate file JTAG_V4xx_FLASHES.pdf to get a complete list of supported flash types.

2. JTBFxxx Parameter Description

When you start JTBFxxx.EXE without any parameters the following help screen with all possible functions and options is displayed:

```
JTBFxxx --- JTAG utility for Analog Devices Blackfin
Copyright © FS FORTH-SYSTEME GmbH, Breisach
Version 4.xx of mm/dd/yyyy

Programming of Flash-EPROMs and hardware tests on targets with the
Analog Devices Blackfin.

The JTAG-Booster is needed to connect the parallel port of the PC
to the JTAG port of the Analog Devices Blackfin.

Usage: JTBFxxx /function [filename] [/option_1] ... [/option_n]
Supported functions:
        /P          : Program a Flash Device
        /R          : Read a Flash Device to file
        /V          : Verify a Flash Device with file
        /DUMP       : Make a target dump
        /PSER       : Program an I2C/SPI/MicroWire Device with file
        /RSER       : Read an I2C/SPI/MicroWire Device to file
        /VSER       : Verify an I2C/SPI/MicroWire Device with file
        /DUMPSER    : Make a dump of an I2C/SPI/MicroWire Device
        /BLINK      : Toggle a CPU pin
        /PIN?       : Test a CPU pin
        /SAMPLE     : Test a CPU pin while the CPU is running
        /SNAP       : Test all CPU pins while CPU is running
        /LIST       : Print a list of supported Flash devices
```

Supported Options:

/CS0	/CS1	/CS2	/CS3	/NOCS
/NOWRSETUP	/TOP	/BYTE-MODE	/BM	/CFI
/CFIDEBUG	/PAUSE	/P	/NODUMP	/NOERASE
/ERASEALL	/LATTICE	/LPT1	/LPT2	/LPT3
/LPT-BASE=	/32BIT	/16BIT	/8BIT	/NOMAN
/LENGTH=	L=	/FILE-OFFSET=	/FO=	/OFFSET=
/O=	/DELAY=	/DEVICE-BASE=	/DB=	/DRIVER=
/IROFFS=	/CPUPOS=	/DEVICE=	/PIN=	/SERCS=
/SERCLK=	/SERDAT=	/SERDATI=	/SERDATO=	/SERBUFF=
/SERBIG	/SPI	/MWIRE	/LSB1ST	/SPIERA
/WATCH=	/OUT=	/INI=	/REP	

The following options are valid for most functions:

/DRIVER=x with x = 1,2,3,4
A driver for the interface to the JTAG-BOOSTER on the parallel port may be specified. /DRIVER=1 selects the fastest available driver, /DRIVER=4 selects the slowest one. Use a slower driver if there are problems with JTAG-BOOSTER.
Default: /DRIVER=3

/INI=file
An initialization file may be specified. By default the current directory is searched for the file JTBFxxx.INI. If this file is not found and no initialization file is specified in the command line, default initialization values are used (see also chapter 1.9 “Initialization file JTBFxxx.INI”).
Note: The initialization file is not loaded for the functions /SAMPLE (chapter 2.11) and /SNAP (chapter 2.12).
Default: /INI=JTBFxxx.INI

/LATTICE
For demonstration purposes this software works with the Lattice ispLSI-Adapter, too. With the option /LATTICE you can simulate the speed achievable with the simple ispLSI-Adapter.

/LPT1 /LPT2 /LPT3
A printer port may be specified where the JTAG-Booster resides. If you are using this program with WinNT, Win2000 or WinXP you must specify /LPT2 or /LPT-BASE=378 to get access to the standard printer port.
Default: /LPT1

/LPT-BASE
The physical I/O-Address of printer port may be specified instead of the logical printer name. Useful option, if you work with WinNT or Win2000, because the standard printer port is mapped as LPT2 here. Use the option /LPT-BASE=378 to get a command line which works independent of the operation system.

/OUT=file_or_device
All screen outputs are redirected to the specified file or device. Note that you can't redirect to the same parallel port where the JTAG-Booster resides.
Default: /OUT=CON

/PAUSE
With the option /PAUSE you can force the program to stop after each screen. Please do not use this option if you redirect the output to a file.
Abbreviation: /P

/WATCH=
With the option /WATCH= a pin can be specified, which is toggled twice per second, while the program is active. This pin may be the trigger of a watchdog. This pin must be specified as output in the initialization file.

/IROFFS=
Specifies the position of the Analog Devices Blackfin instruction register within the JTAG chain. In most cases this option is not needed.
Default: /IROFFS=0

/CPUPOS=
Specifies the position of the Analog Devices Blackfin within the JTAG chain.
Default: /CPUPOS=0

2.1. Program a Flash Device

Usage: JTBFxxx /P filename [optionlist]

The specified file is programmed into the flash memory. The flash status is polled after programming of each cell (cell=8, 16 or 32 bit, depending on current data bus width). In case of a programming error, the contents of the flash memory is written to a file with the extension DMP.

If you want a complete verify after programming, please use an additional command line with the verify function. See chapter 2.3 “Verify a Flash Device with file”. In most cases this additional verify step is not needed.

The type of the flash device is normally detected by the software. When autodetection fails you should use the /DEVICE= option together with /8BIT or /16BIT or /32BIT to set the right flash device and configuration. The known flash devices are shown in chapter 1.10 “Supported flash devices”. Use the option /CFI if the flash is not in the list of known devices.

Options:

/DEVICE=devicename
The flash device is detected automatically by switching to autoselect mode. In case of trouble you should select the flash device by using this parameter to avoid autodetection. Combine this option with one of the following options which specify the data bus width and the option /BYTE-MODE if applicable.

/CFI
To be prepared for future flash chips, the JTAG-Booster integrates support for flashes which contain the CFI (Common Flash Interface) information structure. The CFI support is activated by simply adding the option /CFI to the command line. The JTAG-Booster then automatically searches in all available bus widths for all possible flash types and configurations after searching for the JEDEC identification code.
In case of an error use the command line option /CFIDEBUG instead of /CFI and redirect the program's output into a file. Sending us this file helps in analyzing problems.

/8BIT /16BIT /32BIT
Specifies the data bus width to the target flash device. You can speed up autodetection, if you specify the correct data bus size. You need this option together with the option /DEVICE= to explicit specify a specific flash configuration.
Note: Option /32BIT is not available for Analog Devices ADSP-BF531/BF532/BF533.

/BYTE-MODE
If there is a flash device connected to the CPU which does have a byte mode pin (8 bit and 16/32 bit bus mode), you can force it to be used as 8 bit mode with the option /BYTE-MODE. In most cases this option will not be needed.
Abbreviation: /BM

/NOMAN
If you use a flash device which is identical to one of the supported parts, but is from a different manufacturer, with this option you can suppress the comparison of the manufacturer identification code. We recommend to use this option together with the /DEVICE= option to avoid failures in autodetection.

/DEVICE-BASE=hhhhhh[1]
Here you can specify a flash device starting address. In most cases, where the flash device is selected with one of the CPUs chip select pins, this parameter is not needed. But if there is any decoding logic in your hardware, this option will be needed. Especially, if there are several flash banks connected to one chip select and a sub decoding logic generates chip selects for these flash banks, this option can be used to select a specific flash bank.
Default: /DEVICE-BASE=0
Abbreviation: /DB=

/OFFSET=hhhhhh
The programming starts at an offset of hhhhhh relative to the start address of the flash device. If the offset is negative, the offset specifies an address relative to the end of the flash device. See also option /TOP
Default: /OFFSET=0
Abbreviation: /O=

/TOP
If the option /TOP is used the option /OFFSET= specifies the address where the programming ends (plus one) instead of the starting address. This option is very important for Intel CPU architectures, because target execution always starts at the top of the address space.

/FILE-OFFSET=hhhhhh
If FILE-OFFSET is specified, the first hhhhhh bytes of the file are skipped and not programmed to target.
Default: /FILE-OFFSET=0
Abbreviation: /FO=

/LENGTH=hhhhhh
The number of programmed bytes may be limited to LENGTH. If no LENGTH is specified the whole file is programmed.
Default: /LENGTH=4000000 (64 MByte)
Abbreviation: /L=

/NODUMP
In case of a verify error the contents of the flash memory is written to a file with the extension .DMP. With /NODUMP you can suppress this feature.

[1]hhhhhh=number base is hex

/ERASEALL
Erase the whole flash device. If this option isn't set, only those blocks are erased where new data should be written to.

/NOERASE
This option prevents the flash device from being erased.

/CS0 /CS1 /CS2 /CS3
This options may be used to specify one or more chip select signals to the flash memory. The used chip selects must be defined as output and inactive in the initialization file. (See chapter 1.9 “Initialization file JTBFxxx.INI”.)
Default: /CS0

/NOCS
Use this option to switch off all chip select signals. This may be necessary if the device's chip select is generated via a normal decoder instead of using the Analog Devices Blackfin chip select unit.

/NOWRSETUP
By default write cycles to the Flash EPROM are realized with three steps: 1. set address/data 2. write strobe active 3. write strobe inactive. **In most cases** it is possible to set the write strobe coincident with setting of address and data by specifying the option /NOWRSETUP. **This increases the programming speed by 50%.**

Examples:

JTBFxxx /P ROMDOS.ROM /L=20000 /TOP
This example programs up to 128 Kbytes of the file ROMDOS.ROM (with i.e. 512 Kbytes) to the top of the boot flash memory.

JTBFxxx /P CE.ROM /32BIT /CS1
This example programs the file CE.ROM to the 32 Bit Flash-EPROM connected to AMS1#.

2.2. Read a Flash Device to file

Usage: JTBFxxx /R filename [optionlist]

The contents of a flash device is read and written to a file.

The type of the flash device is normally detected by the software. When autodetection fails you should use the /DEVICE= option together with /8BIT or /16BIT or /32BIT to set the right flash device and configuration. The known devices are shown in chapter 1.10 “Supported flash devices”. Use the option /CFI if the flash is not in the list of known devices.

Options:

/DEVICE=devicename
See function /P (Chapter 2.1)

/CFI
See function /P (Chapter 2.1)

/8BIT /16BIT /32BIT
See function /P (Chapter 2.1)

/BYTE-MODE
See function /P (Chapter 2.1)

/NOMAN
See function /P (Chapter 2.1)

/DEVICE-BASE=hhhhhh[2]
See function /P (Chapter 2.1)

[2]hhhhhh=number base is hex

/OFFSET=hhhhhh
Reading of the flash memory starts at an offset of hhhhhh relative to the start address of the flash device. If the offset is negative, the offset specifies a address relative to the end of the flash device.
See also option /TOP.
Default: /OFFSET=0
Abbreviation: /O=

/TOP
If the option /TOP is used the option /OFFSET= specifies the address where reading ends (plus one) instead of the starting address.

/LENGTH=hhhhhh
The number of read bytes may be limited to LENGTH. If no LENGTH is specified the whole flash device is read (if no offset is specified).

/CS0 /CS1 /CS2 /CS3
See function /P (Chapter 2.1)

/NOWRSETUP
See function /P (Chapter 2.1)
Please note: In the function /R write cycles are needed to detect the type of the flash memory.

Example:

JTBFxxx /R BIOS.ABS /L=10000 /TOP
This example may be used to read the upper most 64 Kbyte of the flash memory to the file BIOS.ABS.

2.3. Verify a Flash Device with file

Usage: JTBFxxx /V filename [optionlist]

The contents of a flash device is compared with the specified file. If there are differences the memory is dumped to a file with the extension DMP.

The type of flash device is normally detected by the software. When autodetect fails you should use the /DEVICE= option together with /8BIT or /16BIT or /32BIT to set the right flash device and configuration. The known devices are shown in chapter 1.10 “Supported flash devices”. Use the option /CFI if the flash is not in the list of known devices.

Options:

/DEVICE=devicename
See function /P (Chapter 2.1)

/CFI
See function /P (Chapter 2.1)

/8BIT /16BIT /32BIT
See function /P (Chapter 2.1)

/BYTE-MODE
See function /P (Chapter 2.1)

/NOMAN
See function /P (Chapter 2.1)

/DEVICE-BASE=hhhhhh
See function /P (Chapter 2.1)

/OFFSET=hhhhhh
See function /P (Chapter 2.1)

/TOP
See function /P (Chapter 2.1)

/FILE-OFFSET=hhhhhh
See function /P (Chapter 2.1)

/LENGTH=hhhhhh
See function /P (Chapter 2.1)

/NODUMP
See function /P (Chapter 2.1)

/CS0 /CS1 /CS2 /CS3
See function /P (Chapter 2.1)

/NOWRSETUP
See function /P (Chapter 2.1)
Please note: In the function /V write cycles are needed to detect the type of the flash memory.

Example:

JTBFxxx /V ROMDOS.ROM /L=20000 /TOP
This example may be used to verify the upper most 128 Kbytes of the flash memory with the file ROMDOS.ROM (with i.e. 512 Kbytes).

2.4. Dump target memory

Usage: JTBFxxx /DUMP [optionlist]

A Hex-Dump of the target memory is printed on the screen, if not redirected to file or device.

Options:

/8BIT /16BIT /32BIT
Default: /16BIT (Analog Devices ADSP-BF531/BF532/BF533)
Default: /32BIT (Analog Devices ADSP-BF561)

/OFFSET=hhhhhh
The memory dump starts at an offset of hhhhhh plus the device start address (see option /DEVICE-BASE=).
Default: /OFFSET=0
Abbreviation: /O=

/DEVICE-BASE=hhhhhh[3]
The device start address is used as an additional offset. This gives the function /DUMP the same behavior as function /P /V and /R.
Default: /DEVICE-BASE=0
Abbreviation: /DB=

/TOP
If the option /TOP is used the option /OFFSET= specifies the address where the dump ends (plus one) instead of the starting address

/LENGTH=hhhhhh
Default: /LENGTH=100
Abbreviation: /L=

/CS0 /CS1 /CS2 /CS3
See function /P (Chapter 2.1)
Default: /CS0

[3]hhhhhh=number base is hex

Example:

JTBFxxx /DUMP
This example makes a memory dump of the first 256 bytes of the Boot-EPROM.

2.5. Program a Serial Device (I²C/SPI/MicroWire)

Usage: JTBFxxx /PSER filename [/SERBIG] [optionlist]

The specified file is programmed to a serial device (i.e. EEPROM) connected to pins of the CPU. Finally a complete verify is done. If the verify fails, the contents of the serial device is written to a file with the extension DMP.

For an I²C device there are two different methods how to connect it to the CPU. The first method uses two CPU pins, one pin for clock output (SERCLK) and one pin for serial data input/output (SERDAT). The second method uses one pin for clock output (SERCLK), one for serial data input (SERDATI) and one for serial data output (SERDATO).

Connecting a SPI/MicroWire device needs four different CPU pins: SERCS is the chip select output of the CPU, SERCLK is the clock output of the CPU, SERDATO is the serial data output of the CPU and must be connected to the SI input at the SPI/MicroWire device and SERDATI is the serial data input to the CPU and must be connected to the SO output of the SPI/MicroWire device.

Options:

/SERBIG
Specify this option if there is a device which needs a three byte address instead of a two byte address. For SPI devices this option is normally needed for devices with more than or equal to 64 kBytes. For I²C devices this option is normally needed for devices with more than 2 kByte.
This option must be the first option after the filename.

/SPI
Specify this option, if there is a SPI device connected instead of an I²C device.

/MWIRE
Specify this option, if there is a MicroWire device connected instead of an I²C device.
Please Note: Actually only the M93C06 and the M93C46 and only in 16 Bit mode are supported.

/DEVICE-BASE=hhhhhh
This option specifies an I²C device starting address. The default values are chosen to access a serial EEPROM. By changing the device starting address different devices can be selected. As SPI/MicroWire devices are selected by the chip select signal instead of an address, this option does not make sense for SPI/MicroWire devices.
Default: /DEVICE-BASE=5000 (if option /SERBIG omitted)
Default: /DEVICE-BASE=500000 (if option /SERBIG specified)
Default: /DEVICE-BASE=0 (for SPI/MicroWire devices)

/OFFSET=hhhhhh
The programming starts at an offset of hhhhhh relative to the start address of the serial device.
Default: /OFFSET=0
Abbreviation: /O=

/FILE-OFFSET=hhhhhh
If FILE-OFFSET is specified, the first hhhhhh bytes of the file are skipped and not programmed to target.
Default: /FILE-OFFSET=0
Abbreviation: /FO=

/LENGTH=hhhhhh
The number of programmed bytes may be limited to LENGTH. If no LENGTH is specified the whole file is programmed.
Abbreviation: /L=

/NODUMP
In case of a verify error the contents of the I²C-Device is written to a file with the extension .DMP. With option /NODUMP you can suppress this feature.

/SERCS=pin_name (SPI/MicroWire mode only)
Specifies the CPU pin used to select the serial device.
In SPI mode SERCS is treated as low active.
In MicroWire mode SERCS is treated as high active.

/SERCLK=pin_name
Specifies the CPU pin used for serial clock output.

/SERDAT=pin_name (I²C only)
Specifies the CPU pin used for serial data input and output for an I²C device. Pin_name must specify a bidirectional pin otherwise an error message occurs. Instead of one bidirectional pin one pin for serial data input and one for serial data output may be used. See option /SERDATO= and /SERDATI= .

/SERDATO=pin_name
Specifies the CPU pin used for serial data output. Pin_name must specify a output pin otherwise an error message occurs. This pin must be connected to the serial data **input** of a SPI/MicroWire device.

/SERDATI=pin_name
Specifies the CPU pin used for serial data input. Pin_name must specify a input pin otherwise an error message occurs. This pin must be connected to the serial data **output** of a SPI/MicroWire device.

/SERBUFF= hhhhhh
For I²C and SPI devices the write page mode can be activated by specifying the option /SERBUFF=. Using this feature increases the programming performance. Please note: Some SPI devices do not support single byte write mode. For these devices the option /SERBUFF= must be specified in the command line.

/LSB1ST
Some devices need the least significant data bit sent/received first (i.e. Altera ECS1 configuration device for FPGAs). Addresses are still sent/received most significant bit first. This option does not affect the behavior of accessing I²C devices.

/SPIERA
Some SPI devices need to be erased before programming. Add option /SPIERA to the command line to perform a chip erase procedure before programming.

Example:

JTBFxxx /PSER EEPROM.CFG /SERCLK=FLAG0 /SERDAT=FLAG1
This example loads the file EEPROM.CFG to a I²C EEPROM connected to the pins FLAG0 and FLAG1 of the Analog Devices Blackfin

2.6. Read a Serial Device to file (I²C/SPI/MicroWire)

Usage: JTBFxxx /RSER filename [/SERBIG] /L=hhhhhh [optionlist]

The contents of a serial device (i.e. EEPROM) is read and written to a file. The option /LENGTH= must be specified.

Options:

/SERBIG
This option must be the first option after the filename.
See function /PSER (Chapter 2.5)

/SPI
Specify this option, if there is a SPI device connected instead of a I²C device.

/MWIRE
Specify this option, if there is a MicroWire device connected instead of an I²C device.

/DEVICE-BASE=hhhhhh
See function /PSER (Chapter 2.5)

/OFFSET=hhhhhh
Reading of the serial device starts at an offset of hhhhhh relative to the start address of the serial device.
Default: /OFFSET=0
Abbreviation: /O=

/LENGTH=hhhhhh
The number of read bytes must be specified otherwise an error message occurs.
Abbreviation: /L=

/SERCS=pin_name
See function /PSER (Chapter 2.5)

/SERCLK=pin_name
See function /PSER (Chapter 2.5)

/SERDAT=pin_name
See function /PSER (Chapter 2.5)

/SERDATO=pin_name
See function /PSER (Chapter 2.5)

/SERDATI=pin_name
See function /PSER (Chapter 2.5)

/LSB1ST
See function /PSER (Chapter 2.5)

Example:

JTBFxxx /RSER EEPROM.CFG /SERCLK=GP26 /SERDAT=GP27 /L=100
This example reads 256 bytes from a I²C EEPROM to the file EEPROM.CFG. The serial EEPROM is connected to the pins CP26 and GP27 of the Analog Devices Blackfin.

2.7. Verify a Serial Device with file (I²C/SPI/MicroWire)

Usage: JTBFxxx /VSER filename [/SERBIG] [optionlist]

The contents of a serial device (i.e. EEPROM) is compared with the specified file. If there are differences the contents of the I²C -Device is written to a file with the extension DMP.

Options:

/SERBIG
This option must be the first option after the filename.
See function /PSER (Chapter 2.5)

/SPI
Specify this option, if there is a SPI device connected instead of a I²C device.

/MWIRE
Specify this option, if there is a MicroWire device connected instead of an I²C device.

/DEVICE-BASE=hhhhhh
See function /PSER (Chapter 2.5)

/OFFSET=hhhhhh
See function /PSER (Chapter 2.5)

/FILE-OFFSET=hhhhhh
See function /PSER (Chapter 2.5)

/LENGTH=hhhhhh
See function /SER (Chapter 2.5)

/NODUMP
See function /PSER (Chapter 2.5)

/SERCS=pin_name
See function /PSER (Chapter 2.5)

/SERCLK=pin_name
See function /PSER (Chapter 2.5)

/SERDAT=pin_name
See function /SER (Chapter 2.5)

/SERDATO=pin_name
See function /PI2C (Chapter 2.5)

/SERDATI=pin_name
See function /PI2C (Chapter 2.5)

/LSB1ST
See function /PSER (Chapter 2.5)

Example:

JTBFxxx /VSER EEPROM.CFG /SERCLK=GP26 /SERDAT=GP27
This example verifies 256 bytes from a serial EEPROM with the file EEPROM.CFG. The serial EEPROM is connected to the pins CP26 and GP27 of the Analog Devices Blackfin.

2.8. Dump a Serial Device (I²C/SPI/MicroWire)

Usage: JTBFxxx /DUMPSER [/SERBIG] [optionlist]

A Hex-Dump of serial device (i.e. EEPROM) is printed on the screen, if not redirected to file or device.

Options:

/SERBIG
This option must be the first option.
See function /PSER (Chapter 2.5)

/SPI
Specify this option, if there is a SPI device connected instead of a I²C device.

/MWIRE
Specify this option, if there is a MicroWire device connected instead of an I²C device.

/DEVICE-BASE=hhhhhh
See function /PSER (Chapter 2.5)

/OFFSET=hhhhhh[4]
The memory dump starts at an offset of hhhhhh.
Default: /OFFSET=0
Abbreviation: /O=

/LENGTH=hhhhhh
Default: /LENGTH=100
Abbreviation: /L=

/SERCS=pin_name
See function /PSER (Chapter 2.5)

/SERCLK=pin_name
See function /PSER (Chapter 2.5)

[4]hhhhhh=number base is hex

/SERDAT=pin_name
See function /SER (Chapter 2.5)

/SERDATO=pin_name
See function /PI2C (Chapter 2.5)

/SERDATI=pin_name
See function /PI2C (Chapter 2.5)

/LSB1ST
See function /PSER (Chapter 2.5)

Example:

JTBFxxx /DUMPSER /SERCLK=FLAG0 /SERDAT=FLAG1
This example makes a memory dump of the first 100h bytes of a I²C EEPROM connected to the CPU.

2.9. Toggle CPU pins

Usage: JTBFxxx /BLINK /PIN=pinname [optionlist]

This command allows to test the hardware by blinking with LEDs or toggling CPU signals. Faster signals can be generated by setting the delay option to zero. This can be a very helpful feature to watch signals on an oscilloscope.

The signal on the defined pin has an duty cycle of 1/2: The level is 67% high and 33% low.

Please Note: Not every pin of the Analog Devices Blackfin may be specified as an output pin.

Options:

/PIN=pin_name
CPU pin to toggle. If the option /PIN= is not specified an error message occurs. Most pins of the list in chapter 1.9 “Initialization file JTBFxxx.INI” can be used. If you type /PIN= without any pin declaration a list of the CPU pins is displayed.

/DELAY=dddddd[5]
Time to wait to next change of signal. This option can be adjusted to get optimum signals for measures with the oscilloscope.
Default: /DELAY=10000

Example:

JTBFxxx /BLINK /PIN=FLAG3 /DELAY=0
This example toggles the FLAG3 pin very fast which can be followed by the use of an oscilloscope.

[5]dddddd=number base is decimal

2.10. Polling CPU pins

Usage: JTBFxxx /PIN? /PIN=pinname [optionlist]

This command allows to test the hardware by polling CPU signals.

Please Note: Not every pin of the Analog Devices Blackfin may be specified as an input pin.

Options:

/PIN=pin_name
CPU pin to poll. If the option /PIN= is not specified an error message occurs. Most pins of the list in chapter 1.9 “Initialization file JTBFxxx.INI” can be used. If you type /PIN= without any pin declaration a list of the CPU pins is displayed.

Example:

JTBFxxx /PIN? /PIN=RESET#
This example samples the reset pin of the Analog Devices Blackfin.

2.11. Polling CPU pins while the CPU is running

Usage: JTBFxxx /SAMPLE /PIN=pinname [optionlist]

This command is similar to the function /PIN?. But with this function any pin can be observed, independent of the pin direction. Furthermore the CPU remains in normal operation.

Options:

/PIN=pin_name
CPU pin to poll. If the option /PIN= is not specified an error message occurs. All pins of the list in chapter 1.9 “Initialization file JTBFxxx.INI” can be used. If you type /PIN= without any pin declaration a list of the CPU pins is displayed.

Example:

JTBFxxx /SAMPLE /PIN=FLAG3
This example samples the state of the port pin FLAG3 while the Analog Devices Blackfin is running.

2.12. Show status of all CPU pins while the CPU is running

Usage: JTBFxxx /SNAP [optionlist]

This function is similar to the function /SAMPLE, but displays the status of all CPU pins on the screen. The CPU remains in normal operation.

The behavior of the function /SNAP depends on the option /REP: With this option specified, the JTAG-Booster samples and displays the state of the CPU pins repetitive. Without this option the status of the pins is displayed only once.

Options:

/PAUSE
Use this option to stop the output after each displayed screen. Don't use this option together with the option /REP or if the output is redirected to a file.
Abbreviation /P

/REP
If this option is specified the status of the pins is sampled and displayed repetitive. In case of many signals the display is separated into several screens. Therefor we recommend to use a video mode with 43 or 50 lines. Use the '+' and the '-' key to switch between different screens. Any other key terminates the program.

Sample output:
This is a sample output for a Analog Devices ADSP-BF531/BF532/BF533

| 1 BG# | 1 BGH# | 0 ADDR19 | 0 ADDR18
| 0 ADDR17 | 0 ADDR16 | 0 ADDR15 | 0 ADDR14
| 0 ADDR13 | 0 ADDR12 | 0 ADDR11 | 0 ADDR10
| 0 ADDR9 | 0 ADDR8 | 1 ADDR7 | 0 ADDR6
| 0 ADDR5 | 1 ADDR4 | 1 ADDR3 | 1 ADDR2
| 1 ADDR1 | 0 ABE1# | 0 ABE0# | 1 AWE#
| 1 ARE# | 1 AOE# | 1 AMS3# | 1 AMS2#
| 1 AMS1# | 1 AMS0# | 1 ARDY | 1 BR#
| 1 SA10 | 1 SWE# | 1 SRAS# | 1 SCAS#
| 0 CLKOUT | 1 SMS# | 1 SCKE | 1 RESET#
| 0 NMI | 1 PPI_CLK | 1 PPI0 | 1 PPI1
| 1 PPI2 | 1 PPI3 | 1 PF15 | 1 PF14
| 0 PF13 | 1 PF12 | 0 PF11 | 0 PF10
| 0 PF9 | 0 PF8 | 0 PF7 | 0 PF6
| 0 PF5 | 0 PF4 | 0 PF3 | 0 PF2
| 1 PF1 | 1 PF0 | 0 SCK | 0 MISO
| 0 MOSI | 0 DT1SEC | 0 DT1PRI | 0 TFS1
| 0 TSCLK1 | 0 DR1SEC | 0 DR1PRI | 0 RFS1
| 0 RSCLK1 | 0 DT0SEC | 0 DT0PRI | 0 TFS0
| 0 TSCLK0 | 0 DR0SEC | 0 DR0PRI | 0 RFS0
| 0 RSCLK0 | 0 TMR2 | 0 TMR1 | 0 TMR0
| 1 TX | 1 RX | 0 BMODE1 | 0 BMODE0
| 0 TEST | 1 DATA15 | 1 DATA14 | 1 DATA13
| 1 DATA12 | 0 DATA11 | 0 DATA10 | 0 DATA9
| 0 DATA8 | 1 DATA7 | 1 DATA6 | 1 DATA5
| 1 DATA4 | 0 DATA3 | 0 DATA2 | 0 DATA1
| 0 DATA0

3. Implementation Information

This chapter summarizes some information about the implementation of the JTAG-Booster and describes some restrictions.

- The JTAG-Booster currently uses Boundary Scan to perform Flash programming. The Debug Interface of the Analog Devices Blackfin is not used.

3.1. Implementation Information Analog Devices ADSP-BF531/BF532/BF533

- The software assumes the following scheme for connecting the (NOR-) Flash-EPROM to the Analog Devices ADSP-BF531/BF532/BF533. Please contact us, if you have used a different method.

Analog Devices ADSP-BF531/BF532/BF533 signal	8 Bit Flash	16 Bit Flash
AMS0# AMS1# AMS2# AMS3#	CS#	CS#
AOE#	OE#	OE#
AWE#	WE#	WE#
ABE0#	A0	-
ADDR1..19	A1..19	A1..19
DATA0..7	D0..7	-
DATA0..15	-	D0..15

1.) Connecting a 8 bit flash device is shown for reference only. Please consult the documentation of the Analog Devices ADSP-BF531/BF532/BF533 if and how 8 bit flash devices are really supported.

2.) The signal ARE# is set to low for read accesses and set high for write accesses.

3.) All other signals are hold static during flash programming. The state of these signals is defined in the Initialization file.

4.) As the highest available address line is A19, the flash size is limited to 1 Mbyte per chip select. If more space in the flash device is needed, a banking logic must be implemented.

3.2. Implementation Information Analog Devices ADSP-BF561

- The software assumes the following scheme for connecting the (NOR-) Flash-EPROM to the Analog Devices ADSP-BF561. Please contact us, if you have used a different method.

Analog Devices ADSP-BF561 signal	**8 Bit Flash**	**16 Bit Flash**	**32 Bit Flash**
AMS0# AMS1# AMS2# AMS3#	CS#	CS#	CS#
AOE#	OE#	OE#	OE#
AWE#	WE#	WE#	WE#
ABE0#	A0	-	-
ABE2#	A1	A1	-
ADDR2..25	A2..25	A2..25	A2..25
DATA0..7	D0..7	-	-
DATA0..15	-	D0..15	-
DATA0..31	-	-	D0..31

1.) Connecting a 8/16 bit flash device is shown for reference only. Please consult the documentation of the Analog Devices ADSP-BF561 if and how 8/16 bit flash devices are really supported.

2.) The signal ARE# is set to low for read accesses and set high for write accesses.

3.) All other signals are hold static during flash programming. The state of these signals is defined in the Initialization file.

4.) As the highest available address line is A25, the flash size is limited to 64 Mbyte per chip select.

4. Converter Program HEX2BIN.EXE

Since the JTAG-Booster software is not able to handle Intel-HEX or Motorola S-Record files, an separate converter tool is delivered with this product package.

Five types of HEX formats can be converted to BIN file:

- I : INTEL HEX format (BYTE oriented)
- D : Digital Research
- M : MOTOROLA S HEX format (BYTE oriented)
- T : TEKTRONICS HEX format (BYTE oriented)
- H : Intel HEX-32

A 4th parameter for starting address can be specified to skip out the leading garbage and you will maintain a small size of output binary file.

If you start the HEX2BIN without any additional parameter all necessary parameters will be asked for in a prompt mode:

```
HEX2BIN
Input HEX file name: MYAPP.H86
Output BIN file name[MYAPP.BIN]:
HEX file format
<I>ntel  /<M>otorola  /<D>igital Research  /<T>ektronics  /[H] Intel HEX-32[I] : H
Input CODE segment start address[0000000]: 10000
Input CODE segment end address[FFFFFFF]:
Unused bytes will be <1>00  <2>FF   [1]  : 2
```

Instead of using the prompt mode, you can directly specify all necessary parameters in the command line. This is essential for making batch files:

```
HEX2BIN  MYAPP.H86  MYAPP.BIN  H  0010000  FFFFFFF  2
```

It is very important to fill unused bytes with 0xFF, because this are simply skipped by the JTAG-Boosters software and so it speeds up the programming performance.

Please Note: **"CODE segment start address"** is interpreted as a Intel x86 architecture segment address: You have to specify a start address of 10000 to start the conversion at 1 MByte.

This converter is a relatively old DOS tool and therefor it has problems with non DOS compliant file and directory names. Avoid names with spaces, limit names to eight characters. Otherwise the converter does not convert the input file, without any error message!!

5. Support for Windows NT, Windows 2000 and Windows XP

A configured run time version of the "Kithara DOS Enabler, Version 6.x" is used to give support for some of our DOS based tools (like the JTAG-Booster) for Windows NT, Windows 2000 and Windows XP. After installation of the "DOS Enabler" the accesses to the LPT ports are allowed for the all programs listed in file Readme_WinNT.txt

Note: Accesses to the ports are only allowed for the programs listed in file Readme_WinNT.txt. If you rename one of our tools, the DOS Enabler does not work.

Important: You need administrator rights to install or de-install this program.

5.1. Installation on a clean system

If you have a clean system without having installed a previous version of the "Kithara Tool Center", this tool is really simple to install. Extract the ZIP file to a new folder and start KSETUP.EXE. Everything is done within a few seconds. No additional input is needed. Now reboot your PC.

5.2. Installation with already installed version 5.x/6.x of Kithara

If you have already installed an older WinNT support (Kithara Version 5.x or 6.x), you have to de-install it 1st as described in chapter 5.4.

After rebooting your PC you can install the Kithara 6.x as described above.

5.3. Installation with already installed version 4.x of Kithara

Important!! If you have already installed an older WinNT support, you have to deinstall it completely!!!

- Start kcenter
- Select Register "Einstellungen" (=Settings)
  and deactivate "VDD benutzen"
  and "speziellen seriellen Treiber benutzen".
- Stop Kernel