

# JTAG-Booster for Intel XScale



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>

Copyright © 1995..2003:

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

Release of Document: September 12, 2003  
Author: Dieter Fögele  
Filename: JTAG\_XScaleb.doc  
Program Version: 4.xx

All rights reserved. No part of this document may be copied or reproduced in any form or by any means without the prior written consent of FS FORTH-SYSTEME GmbH.

**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 Intel PXA210/250.....	10
1.6. First Example with Intel PXA255/26x.....	13
1.7. First Example with Intel IOP321.....	15
1.8. First Example with Intel IXP425.....	17
1.9. Trouble Shooting .....	19
1.10. Error Messages.....	20
1.11. Initialization file JTAGxxx.INI.....	25
1.12. Supported flash devices.....	56
2. JTAGxxx Parameter Description.....	57
2.1. Program a Flash Device .....	60
2.2. Read a Flash Device to file.....	64
2.3. Verify a Flash Device with file.....	66
2.4. Dump target memory.....	68
2.5. Program an I <sup>2</sup> C-Device.....	70
2.6. Read an I <sup>2</sup> C-Device to file.....	72
2.7. Verify an I <sup>2</sup> C-Device with file.....	74
2.8. Dump an I <sup>2</sup> C-Device .....	76
2.9. Toggle CPU pins .....	78
2.10. Polling CPU pins .....	79
2.11. Polling CPU pins while the CPU is running .....	80
2.12. Show status of all CPU pins while the CPU is running .....	81
3. Implementation Information .....	84
4. Converter Program HEX2BIN.EXE.....	86
5. Support for Windows NT, Windows 2000 and Windows XP .....	88
5.1. Installation on a clean system.....	88
5.2. Installation with already installed version 5.x/6.x of Kithara.....	88
5.3. Installation with already installed version 4.x of Kithara .....	88

5.4. De-Installation version 5.x/6.x: ..... 89

## **1. General**

The programs JTAG250.EXE, JTAG255.EXE, JTAG321.EXE and JTAG425 use the JTAG port of the Intel XScale processors 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 an I<sup>2</sup>C Device
- 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.

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 Intel XScale family, there are three different programs on the distribution disk:

- JTAG250.EXE : Tool for Intel PXA210/250
- JTAG255.EXE : Tool for Intel PXA255/26x
- JTAG321.EXE : Tool for Intel IOP321
- JTAG425.EXE : Tool for Intel IXP425

Please contact us, if you need support for other members of the Intel XScale 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

- 9003 JTAG-Booster Intel XScale, 3.3V, PXA210/250, PXA255/26X, IOP321, PXA425  
DOS/Win9x/WinNT/Win2000/WinXP,  
delivered with adapter type 285

### **1.2. System Requirements**

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

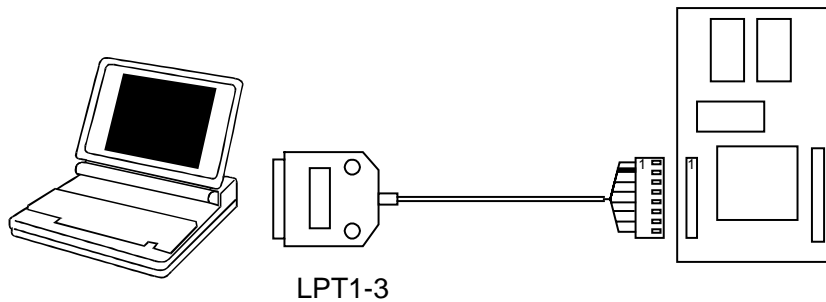
- MSDOS, WIN3.x, WIN9x, WinNT, Win2000 or WindowsXP  
(WinNT/Win2000/WindowsXP is supported with an additional tool, see  
chapter 5 "Support for Windows NT, Windows 2000 and Windows XP")
- Intel 80386 or higher
- 205 kByte of free DOS memory
- Parallel Port

### **1.3. Contents of Distribution Disk**

- JTAG250.EXE      Tool for Intel PXA210/250  
  JTAG250.OVL
- JTAG250.INI      Template configuration file for Intel PXA210/250. See  
  chapter 1.11 "Initialization file JTAGxxx.INI"
- JTAG255.EXE      Tool for Intel PXA255/26x  
  JTAG255.OVL
- JTAG255.INI      Template configuration file for Intel PXA255/26x. See  
  chapter 1.11 "Initialization file JTAGxxx.INI"
- JTAG321.EXE      Tool for Intel IOP321  
  JTAG321.OVL
- JTAG321.INI      Template configuration file for Intel IOP321. See  
  chapter 1.11 "Initialization file JTAGxxx.INI"
- JTAG425.EXE      Tool for Intel IXP425  
  JTAG425.OVL
- JTAG425.INI      Template configuration file for Intel IXP425. See  
  chapter 1.11 "Initialization file JTAGxxx.INI"
- HEX2BIN.EXE      Converter program to convert Intel HEX and Motorola  
  S-Record files to binary. See chapter 4 "Converter  
  Program HEX2BIN.EXE"
- WinNT.zip        Support for Windows NT and Windows 2000. See  
  chapter 5 "Support for Windows NT, Windows 2000"
- JTAG\_V4xx\_FLAS  
  HES.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.



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 an input on the JTAG-Booster's side.

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

JTAGxxx /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 JTAGxxx, press CTRL-Break-Key.

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

### **1.5. First Example with Intel PXA210/250**

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

```
JTAG250 /P MYAPP.BIN
```

at the DOS prompt results in the following output:

```
JTAG250 --- JTAG utility for Intel PXA210/250
Copyright © FS FORTH-SYSTEME GmbH, Breisach
Version 4.xx of mm/dd/yyyy

(1) Configuration loaded from file JTAG250.INI
(2) Target: Generic Target
(3) Using LPT at I/O-address 0378h
(4) JTAG Adapter detected

(5) 1 Device detected in JTAG chain
(6)   Device 0: IDCODE=39264013 Intel XScale PXA210/250
(7) Sum of instruction register bits : 5
(8) CPU position                    : 0
(9) Instruction register offset     : 0
(10) Length of boundary scan reg    : 385

Looking for a known flash device. Please wait..
(11) Dual Intel 28F128 StrataFlash, 3.3V detected
(12) Bus size is 32 Bit
(13) Erasing Flash-EPROM Block #:0
Programming File MYAPP.BIN
65536 Bytes programmed successfully

Erase Time       :      0.8 sec
Programming Time :     28.1 sec
```

- (1) The initialization file JTAG250.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 Intel PXA210/250 are switched to bypass mode.
- (6) The revision (1<sup>st</sup> digit of the ID) is relevant for the PXA2xx family:  
Revision 0 -> PXA250 A0  
Revision 1 -> PXA250 A1  
Revision 2 -> PAA250 B0  
Revision 3 -> PXA250 B1  
Revision 4 -> PXA250 B2  
Revision 5 -> PXA26x B0, use JTAG255.EXE  
Revision 6 -> PXA255 A0, use JTAG255.EXE
- (7) The length of all instruction registers in the JTAG chain are added.
- (8) The position of the Intel XScale in the JTAG chain is assumed to be zero, if not specified in the command line (see option /CPUPOS=).
- (9) The position of the JTAG instruction register of the Intel XScale is assumed to be zero, if not specified in the command line (see option /IROFFS=).
- (10) The real length of the boundary scan register is displayed here and compared with the boundary scan register length of a Intel PXA210/250.
- (11) Two Flashes Intel 28F128 selected with CS0# where found.
- (12) The resulting data bus size is printed here.

(13) In this example one block must be erased.

## **1.6. First Example with Intel PXA255/26x**

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

```
JTAG255 /P MYAPP.BIN
```

at the DOS prompt results in the following output:

```
JTAG255 --- JTAG utility for Intel PXA255/26x
Copyright © FS FORTH-SYSTEME GmbH, Breisach
Version 4.xx of mm/dd/yyyy

(1) Configuration loaded from file JTAG255.INI
(2) Target: Generic Target
(3) Using LPT at I/O-address 0378h
(4) JTAG Adapter detected

(5) 1 Device detected in JTAG chain
(6)   Device 0: IDCODE=69264013 Intel XScale PXA255/26x
(7) Sum of instruction register bits : 5
(8) CPU position                    : 0
(9) Instruction register offset     : 0
(10) Length of boundary scan reg    : 410

Looking for a known flash device. Please wait..
(11) Dual Intel 28F128 StrataFlash, 3.3V detected
(12) Bus size is 32 Bit
(13) Erasing Flash-EEPROM Block #:0
Programming File MYAPP.BIN
65536 Bytes programmed successfully

Erase Time      :      0.8 sec
Programming Time :     29.9 sec
```

- (1) The initialization file JTAG255.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 Intel PXA255/26x are switched to bypass mode.
- (6) The revision (1<sup>st</sup> digit of the ID) is relevant for the Intel PXA2xx family:  
Revision 0 -> PXA250 A0, use JTAG250.EXE  
Revision 1 -> PXA250 A1, use JTAG250.EXE  
Revision 2 -> PAA250 B0, use JTAG250.EXE  
Revision 3 -> PXA250 B1, use JTAG250.EXE  
Revision 4 -> PXA250 B2, use JTAG250.EXE  
Revision 5 -> PXA26x B0  
Revision 6 -> PXA255 A0
- (7) The length of all instruction registers in the JTAG chain are added.
- (8) The position of the Intel XScale in the JTAG chain is assumed to be zero, if not specified in the command line (see option /CPUPOS=).
- (9) The position of the JTAG instruction register of the Intel XScale is assumed to be zero, if not specified in the command line (see option /IROFFS=).
- (10) The real length of the boundary scan register is displayed here and compared with the boundary scan register length of a Intel PXA255/26x.
- (11) Two Flashes Intel 28F128 selected with CS0# where found.
- (12) The resulting data bus size is printed here.
- (13) In this example one block must be erased.

## **1.7. First Example with Intel IOP321**

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

```
JTAG321 /P MYAPP.BIN
```

at the DOS prompt results in the following output:

```
JTAG321 --- JTAG utility for Intel IOP321
Copyright © FS FORTH-SYSTEME GmbH, Breisach
Version 4.xx of mm/dd/yyyy

(1) Configuration loaded from file JTAG321.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=09267013 Intel XScale IOP321, Revision 0
(6) Sum of instruction register bits : 5
(7) CPU position                    : 0
(8) Instruction register offset     : 0
(9) Length of boundary scan reg    : 356

Looking for a known flash device. Please wait..
(10) Dual Intel 28F128 StrataFlash, 3.3V detected
(11) Bus size is 32 Bit
(12) Erasing Flash-EEPROM Block #:0
    Programming File MYAPP.BIN
    65536 Bytes programmed successfully

Erase Time       : 0.8 sec
Programming Time : 26.0 sec
```

- (1) The initialization file JTAG321.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 Intel IOP321 are switched to bypass mode.
- (6) The length of all instruction registers in the JTAG chain are added.
- (7) The position of the Intel XScale 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 Intel XScale 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 Intel IOP321.
- (10) Two Flashes Intel 28F128 selected with PCE0# where found.
- (11) The resulting data bus size is printed here.
- (12) In this example one block must be erased.



## **1.8. First Example with Intel IXP425**

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

```
JTAG425 /P MYAPP.BIN
```

at the DOS prompt results in the following output:

```
JTAG425 --- JTAG utility for Intel IXP425
Copyright © FS FORTH-SYSTEME GmbH, Breisach
Version 4.xx of mm/dd/yyyy

(1) Configuration loaded from file JTAG425.INI
(2) Target: Generic Target with IXP425
(3) Using LPT at I/O-address 0378h
(4) JTAG Adapter detected

(5) 1 Device detected in JTAG chain
(6)   Device 0: IDCODE=09274013 Intel XScale IXP425, Revision 0
(7) Sum of instruction register bits : 7
(8) CPU position                    : 0
(9) Instruction register offset     : 0
(10) Length of boundary scan reg   : 498

Looking for a known flash device. Please wait..
(11) Intel 28F128 StrataFlash, 3.3V detected
(12) Bus size is 16 Bit
(13) Erasing Flash-EPROM Block #:0
Programming File MYAPP.BIN
65536 Bytes programmed successfully

Erase Time      :      0.8 sec
Programming Time :     72.6 sec
```

- (1) The initialization file JTAG425.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 Intel IXP425 are switched to bypass mode.
- (6) There are different ID codes for the Intel IXP425:  
x9274013 -> Intel IXP425, 533 MHz  
x9275013 -> Intel IXP425, 400 MHz  
x9276013 -> Intel IXP425, ??? MHz (reserved)  
x9277013 -> Intel IXP425, 266 MHz  
Note: "x" identifies the silicon stepping
- (7) The length of all instruction registers in the JTAG chain are added.
- (8) The position of the Intel XScale in the JTAG chain is assumed to be zero, if not specified in the command line (see option /CPUPOS=).
- (9) The position of the JTAG instruction register of the Intel XScale is assumed to be zero, if not specified in the command line (see option /IROFFS=).
- (10) The real length of the boundary scan register is displayed here and compared with the boundary scan register length of a Intel IXP425.
- (11) One Flash Intel 28F128 selected with EX\_CS0# was found.
- (12) The resulting data bus size is printed here.
- (13) In this example one block must be erased.

## **1.9. 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 1.11 "Initialization file JTAGxxx.INI"). Also the address bits must be defined as output.

Use the option /NOWRSETUP to speed up flash programming.

### **1.10. 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 /TRITON) 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\JTAGxxx.OVL**  
The overlay file JTAGxxx.OVL must be in the same directory as JTAGxxx.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 "JTAGxxx 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 "JTAGxxx 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 1.11 "Initialization file JTAGxxx.INI"
- **illegal Flash Type:**  
The name specified with the option /DEVICE= must be one of the list of chapter 1.12 "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 1.11 "Initialization file JTAGxxx.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 1<sup>st</sup> 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 /I2CCLK=**  
Some functions need the option /I2CCLK= to be defined.
- **missing option /I2CDAT=**  
Some functions need the option /I2CDAT= or the options /I2CDATO= and /I2CDATI= to be defined.
- **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 Intel XScale**  
The option /CPUPOS= points to a part not a Intel XScale
- **Pins specified with /I2CCLK= and /I2CDAT= must have different control cells**  
The pin specified with the option /I2CDAT= must be able to be switched to high impedance while the pin specified with option /I2CCLK= is an active output. See chapter 1.11 "Initialization file JTAGxxx.INI".
- **Pins specified with /I2CCLK= and /I2CDATI= must have different control cells**  
The pin specified with the option /I2CDATI= must be able to be switched to high impedance while the pin specified with option /I2CCLK= is an active output. See chapter 1.11 "Initialization file JTAGxxx.INI".
- **Pins specified with /I2CDATO= and /I2CDATI= must have different control cells**  
The pin specified with the option /I2CDATI= must be able to be switched to high impedance while the pin specified with option /I2CDATO= is an active output. See chapter 1.11 "Initialization file JTAGxxx.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 Intel XScale (7 bits for the IXP425)**  
The sum of all instruction register bits in the JTAG chain does not fit to the Intel XScale. 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 Intel XScale in the JTAG chain**  
No Intel XScale 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= .
- **Wrong length of boundary scan register. Should be 385 for a Intel PXA210/250. (Should be 410 for a Intel PXA255/26x/Should be 356 for a Intel IOP321/ Should be 498 for a Intel IXP425.)**  
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 Intel XScale. 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.



### **1.11. Initialization file JTAGxxx.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 Intel XScale. The Target-Entry is used to identify your design which is displayed with most commands.

When the program JTAGxxx.EXE is started it scans the current directory for an existing initialization file named JTAGxxx.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 JTAG250.INI:**

```
// Description file for Intel PXA210/250
Target: Generic Target
// Adapt this file carefully to your design!!
// All chip select signals are set to output and inactive.
// The chip selects for the external PC-Card 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 #.

// 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 an input.
GPIO0      Inp    //
GPIO1      Inp    // GP_RST#, GPreset input
GPIO2      Inp    //
GPIO3      Inp    //
GPIO4      Inp    //
GPIO5      Inp    //
GPIO6      Inp    // MMCCLK, MMC clock output
GPIO7      Inp    // 48MHZ, 48MHz clock output
GPIO8      Inp    // MMCCS0, MMC chip select 0
GPIO9      Inp    // MMCCS1, MMC chip select 1
GPIO10     Inp    // RTCCLK, RTC 1Hz output
GPIO11     Inp    // 3.6MHZ, 3.6MHz clock output
GPIO12     Inp    // 32KHZ, 32kHz clock output
GPIO13     Inp    // MBGNT, memory controller grant
GPIO14     Inp    // MBREQ, memory controller alternate bus master request
GPIO15     Out,Hi // CS1#, chip select 1
GPIO16     Inp    // PWM0, PMW0 output
GPIO17     Inp    // PWM1, PWM1 output
GPIO18     Inp    // RDY, ext. bus ready input
GPIO19     Inp    // DREQ1, ext. DMA request 1
GPIO20     Inp    // DREQ0, ext. DMA request 0
GPIO21     Inp    //
GPIO22     Inp    //
GPIO23     Inp    // SSPCLK, SSP clock output
GPIO24     Inp    // SSPFRM, SSP frame
GPIO25     Inp    // SSPTXD, SSP transmit data
GPIO26     Inp    // SSPRXD, SSP receive data
GPIO27     Inp    // SSPEXTCLK, SSP ext. clock input
GPIO28     Inp    // BITCLK, AC97 bit clock input
```

		// BITCLK, AC97 bit clock output
		// BITCLK, I2S bit clock input
		// BITCLK, I2S bit clock output
GPIO29	Inp	// SDATA_IN0, AC97 sdata input 0
		// SDATA_IN, I2S sdata input
GPIO30	Inp	// SDATA_OUT, AC97 sdata output
		// SDATA_OUT, I2S sdata output
GPIO31	Inp	// SYNC, AC97 sync output
		// SYNC, I2S sync output
GPIO32	Inp	// SDATA_IN1, AC97 sdata input 1
		// SYSCLK, I2S system clock output
GPIO33	Out,Hi	// CS5#, chip select 5
GPIO34	Inp	// FFRXD, FFUART receive data
		// MMCCS0, MMC chip select 0
GPIO35	Inp	// FFCTS#, FFUART clear to send
GPIO36	Inp	// FFDCD#, FFUART data carrier detect
GPIO37	Inp	// FFDSR#, FFUART data terminal ready
GPIO38	Inp	// FFR1#, FFUART ring indicator
GPIO39	Inp	// FFTXD, FFUART transmit data
		// MMCCS1, MMC chip select 1
GPIO40	Inp	// FFDTR#, FFUART data terminal ready
GPIO41	Inp	// FFRTS#, FFUART request to send
GPIO42	Inp	// BTRXD, BTUART receive data
GPIO43	Inp	// BTTXD, BTUART transmit data
GPIO44	Inp	// BTCTS#, BTUART clear to send
GPIO45	Inp	// BTRTS#, BTUART request to send
GPIO46	Inp	// RXD, STD_UART receive data
		// IRRXD, IR receive data
GPIO47	Inp	// TXD, STD_UART transmit data
		// IRTXD, IR transmit data
GPIO48	Inp	// POE#, Card Space output enable
GPIO49	Inp	// PWE#, Card Space write enable
GPIO50	Inp	// PIOR#, Card Space I/O read
GPIO51	Inp	// PIOW#, Card Space I/O write
GPIO52	Out,Hi	// PCE1#, Card Space card enable 1, !!!
GPIO53	Out,Hi	// PCE2#, Card Space card enable 2, !!!
		// MMCCLK, MMC clock output
GPIO54	Inp	// PSKTSEL#, Card Space socket select
		// MMCCLK, MMC clock output
GPIO55	Inp	// PREG#, Card Space register select
GPIO56	Inp	// PWAIT#, Card Space wait input
GPIO57	Inp	// IOIS16#, Card Space I/O bus width select
GPIO58	Inp	// L_DD0, LCD data pin 0

```

GPIO59      Inp      // L_DD1, LCD data pin 1
GPIO60      Inp      // L_DD2, LCD data pin 2
GPIO61      Inp      // L_DD3, LCD data pin 3
GPIO62      Inp      // L_DD4, LCD data pin 4
GPIO63      Inp      // L_DD5, LCD data pin 5
GPIO64      Inp      // L_DD6, LCD data pin 6
GPIO65      Inp      // L_DD7, LCD data pin 7
GPIO66      Inp      // L_DD8, LCD data pin 8
                // MBREQ, memory controller alternate bus master request
GPIO67      Inp      // L_DD9, LCD data pin 9
                // MMCCS0, MMC chip select 0
GPIO68      Inp      // L_DD10, LCD data pin 10
                // MMCCS1, MMC chip select 1
GPIO69      Inp      // L_DD11, LCD data pin 11
                // MMCCLK, MMC clock output
GPIO70      Inp      // L_DD12, LCD data pin 12
                // RTCCLK, RTC 1Hz output
GPIO71      Inp      // L_DD13, LCD data pin 13
                // 3.6MHZ, 3.6MHz clock output
GPIO72      Inp      // L_DD14, LCD data pin 14
                // 32KHZ, 32kHz clock output
GPIO73      Inp      // L_DD15, LCD data pin 15
                // MBGNT, memory controller grant
GPIO74      Inp      // L_FCLK, LCD frame clock
GPIO75      Inp      // L_LCLK, LCD line clock
GPIO76      Inp      // L_PCLK, LCD pixel clock
GPIO77      Inp      // L_BIAS, LCD AC bias drive
GPIO78      Out,Hi   // CS2#, chip select 2
GPIO79      Out,Hi   // CS3#, chip select 3
GPIO80      Out,Hi   // CS4#, chip select 4
MMDAT      Inp      // MMC data pin
MMCMD      Inp      // MMC command pin
CS0#       Out,Hi   // Boot Chip Select

```

// Group 85: All pins in this group must be set to the same direction

// These pins are bidirectional

```

MD0        Inp      // Memory Data Bus
MD1        Inp      // Memory Data Bus
MD2        Inp      // Memory Data Bus
MD3        Inp      // Memory Data Bus
MD4        Inp      // Memory Data Bus
MD5        Inp      // Memory Data Bus
MD6        Inp      // Memory Data Bus

```

```
MD7      Inp    // Memory Data Bus
MD8      Inp    // Memory Data Bus
MD9      Inp    // Memory Data Bus
MD10     Inp    // Memory Data Bus
MD11     Inp    // Memory Data Bus
MD12     Inp    // Memory Data Bus
MD13     Inp    // Memory Data Bus
MD14     Inp    // Memory Data Bus
MD15     Inp    // Memory Data Bus

// Group 84: All pins in this group must be set to the same direction
//          These pins are bidirectional
MD16     Inp    // Memory Data Bus
MD17     Inp    // Memory Data Bus
MD18     Inp    // Memory Data Bus
MD19     Inp    // Memory Data Bus
MD20     Inp    // Memory Data Bus
MD21     Inp    // Memory Data Bus
MD22     Inp    // Memory Data Bus
MD23     Inp    // Memory Data Bus
MD24     Inp    // Memory Data Bus
MD25     Inp    // Memory Data Bus
MD26     Inp    // Memory Data Bus
MD27     Inp    // Memory Data Bus
MD28     Inp    // Memory Data Bus
MD29     Inp    // Memory Data Bus
MD30     Inp    // Memory Data Bus
MD31     Inp    // Memory Data Bus

// Group 205: All pins in this group must be set to the same direction
//          These pins are tristateable, but can not be read back
RD/WR#   Out,Hi // Read/Write#
DQM2     Out,Lo //
DQM3     Out,Lo //

// Group 207: All pins in this group must be set to the same direction
//          These pins are tristateable, but can not be read back
WE#      Out,Hi //
OE#      Out,Hi //
```

```
// Group 208: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
SDCLK1      Out,Lo //
SDCS0#      Out,Hi //
DQM0        Out,Lo // SDRAM DQM0
DQM1        Out,Lo // SDRAM DQM1
SDCAS#      Out,Hi // SDRAM CAS#
SDRAS#      Out,Hi // SDRAM RAS#
MA0         Out,Lo //
MA1         Out,Lo //
MA2         Out,Lo //
MA3         Out,Lo //
MA4         Out,Lo //
MA5         Out,Lo //
MA6         Out,Lo //
MA7         Out,Lo //
MA8         Out,Lo //
MA9         Out,Lo //
MA10        Out,Lo //
MA11        Out,Lo //
MA12        Out,Lo //
MA13        Out,Lo //
MA14        Out,Lo //
MA15        Out,Lo //
MA16        Out,Lo //
MA17        Out,Lo //
MA18        Out,Lo //
MA19        Out,Lo //
MA20        Out,Lo //
MA21        Out,Lo //
MA22        Out,Lo //
MA23        Out,Lo //
MA24        Out,Lo //
MA25        Out,Lo //

// Group 81: All pins in this group must be set to the same direction
//           These pins are bidirectional
USB_N       Inp    //
USB_P       Inp    //
```

---

```
// The following pins are output only pins.
// Setting to input (tristate) one of these pins results in an error.
PWR_EN      Out,Hi // Enable ext. power supply
RESET_OUT#  Out,Lo // Reset output
ACRESET#    Out,Lo // AC97 audio port reset output
SDCLK0      Out,Lo // SDRAM clock output 0
SDCLK2      Out,Lo // SDRAM clock output 2
SDCKE0      Out,Lo // SDRAM clock enable 0
SDCKE1      Out,Lo // SDRAM clock enable 1
SDCS1#      Out,Hi // SDRAM chip select 1
SDCS2#      Out,Hi // SDRAM chip select 2
SDCS3#      Out,Hi // SDRAM chip select 3

// 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.
TEST        Inp    // Test Mode input, must be grounded
TESTCLK     Inp    // Test clock input, should be grounded
VDD_FAULT#  Inp    // VDD fault
BATT_FAULT# Inp    // Battery fault
BOOT_SEL0   Inp    // boot programming select pin
BOOT_SEL1   Inp    // boot programming select pin
BOOT_SEL2   Inp    // boot programming select pin
RESET#      Inp    // hard reset input
```

**Sample File JTAG255.INI:**

```
// Description file for Intel PXA255/26x
Target: Generic Target
// Adapt this file carefully to your design!!
// All chip select signals are set to output and inactive.
// The chip selects for the external PC-Card 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 #.

// 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 an input.
GPIO0      Inp    //
GPIO1      Inp    // GP_RST#, GPreset input
GPIO2      Inp    //
GPIO3      Inp    //
GPIO4      Inp    //
GPIO5      Inp    //
GPIO6      Inp    // MMCCLK, MMC clock output
GPIO7      Inp    // 48MHZ, 48MHz clock output
GPIO8      Inp    // MMCCS0, MMC chip select 0
GPIO9      Inp    // MMCCS1, MMC chip select 1
              // USB_RCV, USB client single-ended interface RCV
GPIO10     Inp    // RTCCLK, RTC 1Hz output
GPIO11     Inp    // 3.6MHZ, 3.6MHz clock output
GPIO12     Inp    // 32KHZ, 32kHz clock output
GPIO13     Inp    // MBGNT, memory controller grant
GPIO14     Inp    // MBREQ, memory controller alternate bus master request
GPIO15     Out,Hi // CS1#, chip select 1
GPIO16     Inp    // PWM0, PMW0 output
GPIO17     Inp    // PWM1, PWM1 output
GPIO18     Inp    // RDY, ext. bus ready input
GPIO19     Inp    // DREQ1, ext. DMA request 1
GPIO20     Inp    // DREQ0, ext. DMA request 0
GPIO21     Inp    //
GPIO22     Inp    //
GPIO23     Inp    // SSPCLK, SSP clock output
GPIO24     Inp    // SSPFRM, SSP frame
GPIO25     Inp    // SSPTXD, SSP transmit data
GPIO26     Inp    // SSPRXD, SSP receive data
GPIO27     Inp    // SSPEXTCLK, SSP ext. clock input
```



GPIO28	Inp	// BITCLK, AC97 bit clock input // BITCLK, AC97 bit clock output // BITCLK, I2S bit clock input // BITCLK, I2S bit clock output
GPIO29	Inp	// SDATA_IN0, AC97 sdata input 0 // SDATA_IN, I2S sdata input
GPIO30	Inp	// SDATA_OUT, AC97 sdata output // SDATA_OUT, I2S sdata output
GPIO31	Inp	// SYNC, AC97 sync output // SYNC, I2S sync output
GPIO32	Inp	// SDATA_IN1, AC97 sdata input 1 // SYSCLK, I2S system clock output // USB_VP, USB client single-ended interface VP
GPIO33	Out,Hi	// CS5#, chip select 5
GPIO34	Inp	// FFRXD, Full Function UART receive data // MMCCS0, MMC chip select 0 // USB_VM, USB client single-ended interface VM
GPIO35	Inp	// FFCTS#, Full Function UART clear to send
GPIO36	Inp	// FFDCD#, Full Function UART data carrier detect
GPIO37	Inp	// FFDSR#, Full Function UART data terminal ready
GPIO38	Inp	// FFRI#, Full Function UART ring indicator
GPIO39	Inp	// FFTXD, Full Function UART transmit data // MMCCS1, MMC chip select 1 // USB_VPO, USB client single-ended interface VPO
GPIO40	Inp	// FFDTR#, Full Function UART data terminal ready
GPIO41	Inp	// FFRTS#, Full Function UART request to send
GPIO42	Inp	// BTRXD, Bluetooth UART receive data // HWRXD, Hardware UART receive data
GPIO43	Inp	// BTTXD, Bluetooth UART transmit data // HWTXD, Hardware UART transmit data
GPIO44	Inp	// BTCTS#, Bluetooth UART clear to send // HWCTS#, Hardware UART clear to send
GPIO45	Inp	// BTRTS#, Bluetooth UART request to send // HWRTS#, Hardware UART request to send
GPIO46	Inp	// RXD, STD_UART receive data // IRRXD, IrDA receive data
GPIO47	Inp	// TXD, STD_UART transmit data // IRTXD, IrDA transmit data
GPIO48	Inp	// POE#, Card Space output enable // HWTXD, Hardware UART transmit data
GPIO49	Inp	// PWE#, Card Space write enable // HWRXD, Hardware UART receive data
GPIO50	Inp	// PIOR#, Card Space I/O read

```

GPIO51      Inp    // HWCTS#, Hardware UART clear to send
              // PIOW#, Card Space I/O write
              // HWRTS#, Hardware UART request to send
GPIO52      Out,Hi // PCE1#, Card Space card enable 1, !!!
GPIO53      Out,Hi // PCE2#, Card Space card enable 2, !!!
              // MMCCLK, MMC clock output
GPIO54      Inp    // PSKTSEL#, Card Space socket select
              // MMCCLK, MMC clock output
GPIO55      Inp    // PREG#, Card Space register select
GPIO56      Inp    // PWAIT#, Card Space wait input
              // USB_VMO, USB client single-ended interface VMO
GPIO57      Inp    // IOIS16#, Card Space I/O bus width select
              // USB_OE#, USB client single-ended interface OE#
GPIO58      Inp    // L_DD0, LCD data pin 0
GPIO59      Inp    // L_DD1, LCD data pin 1
GPIO60      Inp    // L_DD2, LCD data pin 2
GPIO61      Inp    // L_DD3, LCD data pin 3
GPIO62      Inp    // L_DD4, LCD data pin 4
GPIO63      Inp    // L_DD5, LCD data pin 5
GPIO64      Inp    // L_DD6, LCD data pin 6
GPIO65      Inp    // L_DD7, LCD data pin 7
GPIO66      Inp    // L_DD8, LCD data pin 8
              // MBREQ, memory controller alternate bus master request
GPIO67      Inp    // L_DD9, LCD data pin 9
              // MMCCS0, MMC chip select 0
GPIO68      Inp    // L_DD10, LCD data pin 10
              // MMCCS1, MMC chip select 1
GPIO69      Inp    // L_DD11, LCD data pin 11
              // MMCCLK, MMC clock output
GPIO70      Inp    // L_DD12, LCD data pin 12
              // RTCCLK, RTC 1Hz output
GPIO71      Inp    // L_DD13, LCD data pin 13
              // 3.6MHZ, 3.6MHz clock output
GPIO72      Inp    // L_DD14, LCD data pin 14
              // 32KHZ, 32kHz clock output
GPIO73      Inp    // L_DD15, LCD data pin 15
              // MBGNT, memory controller grant
GPIO74      Inp    // L_FCLK, LCD frame clock
GPIO75      Inp    // L_LCLK, LCD line clock
GPIO76      Inp    // L_PCLK, LCD pixel clock
GPIO77      Inp    // L_BIAS, LCD AC bias drive
GPIO78      Out,Hi // CS2#, chip select 2
GPIO79      Out,Hi // CS3#, chip select 3

```

```
GPIO80      Out,Hi // CS4#, chip select 4
GPIO81      Inp   // NSSPCLK, Network synchronous serial port (output)
GPIO82      Inp   // NSSPSFRM, Network synchronous serial frame signal (output)
GPIO83      Inp   // NSSPTXD, Network synchronous serial port transmit (output)
GPIO84      Inp   // NSSPRXD, Network synchronous serial port receive (input)
GPIO85      Inp   //
GPIO86      Out,Hi // SDCS2#, SDRAM chip select 2
GPIO87      Out,Hi // SDCS3#, SDRAM chip select 3
GPIO88      Out,Hi // RD/WR#, Read/Write#
GPIO89      Out,Lo // ACRESET#, AC97 audio port reset output
MMDAT       Inp   // MMC data pin
MMCMD       Inp   // MMC command pin
CS0#        Out,Hi // Boot Chip Select
```

```
// Group 85: All pins in this group must be set to the same direction
```

```
//           These pins are bidirectional
MD0         Inp   // Memory Data Bus
MD1         Inp   // Memory Data Bus
MD2         Inp   // Memory Data Bus
MD3         Inp   // Memory Data Bus
MD4         Inp   // Memory Data Bus
MD5         Inp   // Memory Data Bus
MD6         Inp   // Memory Data Bus
MD7         Inp   // Memory Data Bus
MD8         Inp   // Memory Data Bus
MD9         Inp   // Memory Data Bus
MD10        Inp   // Memory Data Bus
MD11        Inp   // Memory Data Bus
MD12        Inp   // Memory Data Bus
MD13        Inp   // Memory Data Bus
MD14        Inp   // Memory Data Bus
MD15        Inp   // Memory Data Bus
```

```
// Group 84: All pins in this group must be set to the same direction
```

```
//           These pins are bidirectional
MD16        Inp   // Memory Data Bus
MD17        Inp   // Memory Data Bus
MD18        Inp   // Memory Data Bus
MD19        Inp   // Memory Data Bus
MD20        Inp   // Memory Data Bus
MD21        Inp   // Memory Data Bus
MD22        Inp   // Memory Data Bus
MD23        Inp   // Memory Data Bus
```

```

MD24      Inp    // Memory Data Bus
MD25      Inp    // Memory Data Bus
MD26      Inp    // Memory Data Bus
MD27      Inp    // Memory Data Bus
MD28      Inp    // Memory Data Bus
MD29      Inp    // Memory Data Bus
MD30      Inp    // Memory Data Bus
MD31      Inp    // Memory Data Bus

```

```

// Group 205: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
DQM2      Out,Lo //
DQM3      Out,Lo //

```

```

// Group 207: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
WE#       Out,Hi //
OE#       Out,Hi //

```

```

// Group 208: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
SDCLK1    Out,Lo //
SDCS0#    Out,Hi //
DQM0      Out,Lo // SDRAM DQM0
DQM1      Out,Lo // SDRAM DQM1
SDCAS#    Out,Hi // SDRAM CAS#
SDRAS#    Out,Hi // SDRAM RAS#
MA0       Out,Lo //
MA1       Out,Lo //
MA2       Out,Lo //
MA3       Out,Lo //
MA4       Out,Lo //
MA5       Out,Lo //
MA6       Out,Lo //
MA7       Out,Lo //
MA8       Out,Lo //
MA9       Out,Lo //
MA10      Out,Lo //
MA11      Out,Lo //
MA12      Out,Lo //
MA13      Out,Lo //
MA14      Out,Lo //
MA15      Out,Lo //

```

```
MA16      Out,Lo //
MA17      Out,Lo //
MA18      Out,Lo //
MA19      Out,Lo //
MA20      Out,Lo //
MA21      Out,Lo //
MA22      Out,Lo //
MA23      Out,Lo //
MA24      Out,Lo //
MA25      Out,Lo //

// Group 81: All pins in this group must be set to the same direction
//           These pins are bidirectional
USB_N     Inp    //
USB_P     Inp    //

// The following pins are output only pins.
// Setting to input (tristate) one of these pins results in an error.
PWR_EN    Out,Hi // Enable ext. power supply
RESET_OUT# Out,Lo // Reset output
SDCLK0    Out,Lo // SDRAM clock output 0
SDCLK2    Out,Lo // SDRAM clock output 2
SDCKE0    Out,Lo // SDRAM clock enable 0
SDCKE1    Out,Lo // SDRAM clock enable 1
SDCS1#    Out,Hi // SDRAM chip select 1

// 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.
TEST      Inp    // Test Mode input, must be grounded
TESTCLK   Inp    // Test clock input, should be grounded
VDD_FAULT# Inp   // VDD fault
BATT_FAULT# Inp  // Battery fault
BOOT_SEL0 Inp    // boot programming select pin
BOOT_SEL1 Inp    // boot programming select pin
BOOT_SEL2 Inp    // boot programming select pin
RESET#    Inp    // hard reset input
```

**Sample File JTAG321.INI:**

```
// Description file for Intel IOP321
Target: Generic Target
// Adapt this file carefully to your design!!
// 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 39: All pins in this group must be set to the same direction
//           These pins are bidirectional
PCE0#      Out,Hi // PBI Peripheral Chip Enable 0
             // RST_MODE# configuration pin latch at pos. edge of P_RST#
PCE1#      Out,Hi // PBI Peripheral Chip Enable 1
             // RETRY configuration pin latch at pos. edge of P_RST#
PCE2#      Out,Hi // PBI Peripheral Chip Enable 2
             // 32BITPCI# configuration pin latch at pos. edge of P_RST#
PCE3#      Out,Hi // PBI Peripheral Chip Enable 3
             // P_BOOT16# configuration pin latch at pos. edge of P_RST#
PCE4#      Out,Hi // PBI Peripheral Chip Enable 4
             // PBI66MHZ# configuration pin latch at pos. edge of P_RST#
PCE5#      Out,Hi // PBI Peripheral Chip Enable 5
             // PBI100MHZ# configuration pin latch at pos. edge of P_RST#

// Group 40: All pins in this group must be set to the same direction
//           These pins are bidirectional
AD24      Inp   // PBI Address/Data Bus
AD25      Inp   // PBI Address/Data Bus
AD26      Inp   // PBI Address/Data Bus
AD27      Inp   // PBI Address/Data Bus
AD28      Inp   // PBI Address/Data Bus
AD29      Inp   // PBI Address/Data Bus
AD30      Inp   // PBI Address/Data Bus
AD31      Inp   // PBI Address/Data Bus

// Group 41: All pins in this group must be set to the same direction
//           These pins are bidirectional
AD16      Inp   // PBI Address/Data Bus
AD17      Inp   // PBI Address/Data Bus
AD18      Inp   // PBI Address/Data Bus
AD19      Inp   // PBI Address/Data Bus
AD20      Inp   // PBI Address/Data Bus
```

```
AD21      Inp    // PBI Address/Data Bus
AD22      Inp    // PBI Address/Data Bus
AD23      Inp    // PBI Address/Data Bus

// Group 42: All pins in this group must be set to the same direction
//          These pins are bidirectional
AD8       Inp    // PBI Address/Data Bus
AD9       Inp    // PBI Address/Data Bus
AD10      Inp    // PBI Address/Data Bus
AD11      Inp    // PBI Address/Data Bus
AD12      Inp    // PBI Address/Data Bus
AD13      Inp    // PBI Address/Data Bus
AD14      Inp    // PBI Address/Data Bus
AD15      Inp    // PBI Address/Data Bus

// Group 43: All pins in this group must be set to the same direction
//          These pins are bidirectional
AD0       Inp    // PBI Address/Data Bus
AD1       Inp    // PBI Address/Data Bus
AD2       Inp    // PBI Address/Data Bus
AD3       Inp    // PBI Address/Data Bus
AD4       Inp    // PBI Address/Data Bus
AD5       Inp    // PBI Address/Data Bus
AD6       Inp    // PBI Address/Data Bus
AD7       Inp    // PBI Address/Data Bus

// Group 51: All pins in this group must be set to the same direction
//          These pins are tristateable, but can not be read back
A2        Out,Lo // PBI Demultiplexed Address
A3        Out,Lo // PBI Demultiplexed Address

// Group 52: All pins in this group must be set to the same direction
//          These pins are tristateable, but can not be read back
WIDTH0    Out,Lo // PBI
WIDTH1    Out,Lo // PBI
```

```
// Group 53: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
BE0#        Out,Lo // PBI Byte Enable for AD0..7 for 32/16Bit
              // PBI Address Bit A0 for 8Bit
BE1#        Out,Lo // PBI Byte Enable for AD8..15 for 32Bit
              // PBI Address Bit A1 for 16/8Bit
BE2#        Out,Lo // PBI Byte Enable for AD16..23 for 32Bit
BE3#        Out,Lo // PBI Byte Enable for AD24..31 for 32Bit
              // PBI Byte Enable for AD8..15 for 16Bit

// Group 111: All pins in this group must be set to the same direction
//           These pins are bidirectional
P_AD32      Inp    // PCI Data upper bits
P_AD33      Inp    //
P_AD34      Inp    //
P_AD35      Inp    //
P_AD36      Inp    //
P_AD37      Inp    //
P_AD38      Inp    //
P_AD39      Inp    //
P_AD40      Inp    //
P_AD41      Inp    //
P_AD42      Inp    //
P_AD43      Inp    //
P_AD44      Inp    //
P_AD45      Inp    //
P_AD46      Inp    //
P_AD47      Inp    //
P_AD48      Inp    //
P_AD49      Inp    //
P_AD50      Inp    //
P_AD51      Inp    //
P_AD52      Inp    //
P_AD53      Inp    //
P_AD54      Inp    //
P_AD55      Inp    //
P_AD56      Inp    //
P_AD57      Inp    //
P_AD58      Inp    //
P_AD59      Inp    //
P_AD60      Inp    //
P_AD61      Inp    //
P_AD62      Inp    //
```



```
P_AD63      Inp    //

// Group 112: All pins in this group must be set to the same direction
//           These pins are bidirectional
P_AD0       Inp    // PCI Address/Data
P_AD1       Inp    //
P_AD2       Inp    //
P_AD3       Inp    //
P_AD4       Inp    //
P_AD5       Inp    //
P_AD6       Inp    //
P_AD7       Inp    //
P_AD8       Inp    //
P_AD9       Inp    //
P_AD10      Inp    //
P_AD11      Inp    //
P_AD12      Inp    //
P_AD13      Inp    //
P_AD14      Inp    //
P_AD15      Inp    //
P_AD16      Inp    //
P_AD17      Inp    //
P_AD18      Inp    //
P_AD19      Inp    //
P_AD20      Inp    //
P_AD21      Inp    //
P_AD22      Inp    //
P_AD23      Inp    //
P_AD24      Inp    //
P_AD25      Inp    //
P_AD26      Inp    //
P_AD27      Inp    //
P_AD28      Inp    //
P_AD29      Inp    //
P_AD30      Inp    //
P_AD31      Inp    //

// Group 113: All pins in this group must be set to the same direction
//           These pins are bidirectional
P_CBE0#     Inp    // PCI Command and Byte Enable
P_CBE1#     Inp    // PCI Command and Byte Enable
P_CBE2#     Inp    // PCI Command and Byte Enable
P_CBE3#     Inp    // PCI Command and Byte Enable
```

```
// Group 114: All pins in this group must be set to the same direction
//           These pins are bidirectional
P_CBE4#    Inp  // PCI Byte Enable
P_CBE5#    Inp  // PCI Byte Enable
P_CBE6#    Inp  // PCI Byte Enable
P_CBE7#    Inp  // PCI Byte Enable

// Group 224: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
M_CK0      Out,Lo // DDR Memory Clock
M_CK0#     Out,Hi // DDR Memory Clock

// Group 225: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
M_CK1      Out,Lo // SDRAM Memory Clock
M_CK1#     Out,Hi // SDRAM Memory Clock

// Group 226: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
M_CK2      Out,Lo // SDRAM Memory Clock
M_CK2#     Out,Hi // SDRAM Memory Clock

// Group 227: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
SCS0#      Out,Hi // SDRAM Chip Select
SCS1#      Out,Hi // SDRAM Chip Select
SRAS#      Out,Hi // SDRAM Row Address Strobe
SCAS#      Out,Hi // SDRAM Column Address Strobe
SWE#       Out,Hi // SDRAM Write Enable
M_RST#     Out,Lo // SDRAM Reset Output
RCVENO#    Out,Hi // SDRAM Receive Enable Out
SCKE0      Out,Lo // SDRAM Clock Enable Output
SCKE1      Out,Lo // SDRAM Clock Enable Output

// Group 228: All pins in this group must be set to the same direction
//           These pins are tristateable, but can not be read back
SBA0       Out,Lo // SDRAM Bank Address
SBA1       Out,Lo // SDRAM Bank Address
SA0        Out,Lo // SDRAM Memory Address
SA1        Out,Lo // SDRAM Memory Address
SA2        Out,Lo // SDRAM Memory Address
SA3        Out,Lo // SDRAM Memory Address
```

---

```
SA4      Out,Lo // SDRAM Memory Address
SA5      Out,Lo // SDRAM Memory Address
SA6      Out,Lo // SDRAM Memory Address
SA7      Out,Lo // SDRAM Memory Address
SA8      Out,Lo // SDRAM Memory Address
SA9      Out,Lo // SDRAM Memory Address
SA10     Out,Lo // SDRAM Memory Address
SA11     Out,Lo // SDRAM Memory Address
SA12     Out,Lo // SDRAM Memory Address

// Group 229: All pins in this group must be set to the same direction
//          SDQM0..3 are tristateable, but can not be read back
//          DQ0..31 are bidirectional
SDQM0    Inp    // SDRAM Data Mask
SDQM1    Inp    // SDRAM Data Mask
SDQM2    Inp    // SDRAM Data Mask
SDQM3    Inp    // SDRAM Data Mask
DQ0      Inp    // SDRAM Data
DQ1      Inp    // SDRAM Data
DQ2      Inp    // SDRAM Data
DQ3      Inp    // SDRAM Data
DQ4      Inp    // SDRAM Data
DQ5      Inp    // SDRAM Data
DQ6      Inp    // SDRAM Data
DQ7      Inp    // SDRAM Data
DQ8      Inp    // SDRAM Data
DQ9      Inp    // SDRAM Data
DQ10     Inp    // SDRAM Data
DQ11     Inp    // SDRAM Data
DQ12     Inp    // SDRAM Data
DQ13     Inp    // SDRAM Data
DQ14     Inp    // SDRAM Data
DQ15     Inp    // SDRAM Data
DQ16     Inp    // SDRAM Data
DQ17     Inp    // SDRAM Data
DQ18     Inp    // SDRAM Data
DQ19     Inp    // SDRAM Data
DQ20     Inp    // SDRAM Data
DQ21     Inp    // SDRAM Data
DQ22     Inp    // SDRAM Data
DQ23     Inp    // SDRAM Data
DQ24     Inp    // SDRAM Data
DQ25     Inp    // SDRAM Data
```

```

DQ26      Inp    // SDRAM Data
DQ27      Inp    // SDRAM Data
DQ28      Inp    // SDRAM Data
DQ29      Inp    // SDRAM Data
DQ30      Inp    // SDRAM Data
DQ31      Inp    // SDRAM Data

// Group 232: All pins in this group must be set to the same direction
//          SDQM4..7 are tristateable, but can not be read back
//          DQ32..63 are bidirectional
SDQM4     Out,Lo // SDRAM Data Mask
SDQM5     Out,Lo // SDRAM Data Mask
SDQM6     Out,Lo // SDRAM Data Mask
SDQM7     Out,Lo // SDRAM Data Mask
DQ32      Inp    // SDRAM Data
DQ33      Inp    // SDRAM Data
DQ34      Inp    // SDRAM Data
DQ35      Inp    // SDRAM Data
DQ36      Inp    // SDRAM Data
DQ37      Inp    // SDRAM Data
DQ38      Inp    // SDRAM Data
DQ39      Inp    // SDRAM Data
DQ40      Inp    // SDRAM Data
DQ41      Inp    // SDRAM Data
DQ42      Inp    // SDRAM Data
DQ43      Inp    // SDRAM Data
DQ44      Inp    // SDRAM Data
DQ45      Inp    // SDRAM Data
DQ46      Inp    // SDRAM Data
DQ47      Inp    // SDRAM Data
DQ48      Inp    // SDRAM Data
DQ49      Inp    // SDRAM Data
DQ50      Inp    // SDRAM Data
DQ51      Inp    // SDRAM Data
DQ52      Inp    // SDRAM Data
DQ53      Inp    // SDRAM Data
DQ54      Inp    // SDRAM Data
DQ55      Inp    // SDRAM Data
DQ56      Inp    // SDRAM Data
DQ57      Inp    // SDRAM Data
DQ58      Inp    // SDRAM Data
DQ59      Inp    // SDRAM Data
DQ60      Inp    // SDRAM Data

```

```
DQ61      Inp    // SDRAM Data
DQ63      Inp    // SDRAM Data

// Group 233: All pins in this group must be set to the same direction
//           SCB0..7 are bidirectional
//           SDQM8 is tristateable, but can not be read back
SCB0      Inp    // SDRAM ECC Check
SCB1      Inp    // SDRAM ECC Check
SCB2      Inp    // SDRAM ECC Check
SCB3      Inp    // SDRAM ECC Check
SCB4      Inp    // SDRAM ECC Check
SCB5      Inp    // SDRAM ECC Check
SCB6      Inp    // SDRAM ECC Check
SCB7      Inp    // SDRAM ECC Check
SDQM8     Inp    // SDRAM Data Mask

// Group 230: All pins in this group must be set to the same direction
//           These pins are bidirectional
DQS0      Inp    // SDRAM Data Strobe
DQS1      Inp    // SDRAM Data Strobe
DQS2      Inp    // SDRAM Data Strobe
DQS3      Inp    // SDRAM Data Strobe

// Group 231: All pins in this group must be set to the same direction
//           These pins are bidirectional
DQS4      Inp    // SDRAM Data Strobe
DQS5      Inp    // SDRAM Data Strobe
DQS6      Inp    // SDRAM Data Strobe
DQS7      Inp    // SDRAM Data Strobe

// 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 an input.
NC0       Inp    // not connected
NC1       Inp    // not connected
NC2       Inp    // not connected
GPIO0     Inp    // General Purpose Input/Output
GPIO1     Inp    // General Purpose Input/Output
GPIO2     Inp    // General Purpose Input/Output
GPIO3     Inp    // General Purpose Input/Output
GPIO4     Inp    // = SDA1
GPIO5     Inp    // = SCL1
GPIO6     Inp    // = SDA0
```

```

GPIO7      Inp    // = SCL0
RDYRCV#    Out,Hi // PBI Read/Recover
P_ACK64#    Inp    // PCI Bus Acknowledge 64 bit Transfer
P_REQ64#    Inp    // PCI Bus Request 64 bit
P_PAR64     Inp    // PCI Bus Upper DWORD Parity
P_SERR#     Inp    // PCI Bus System Error
P_PERR#     Inp    // PCI Parity Error
P_PAR       Inp    // PCI Bus Parity
P_STOP#     Inp    // PCI Bus Stop
P_DEVSEL#   Inp    // PCI Bus Device Select
P_TRDY#     Inp    // PCI Bus Target Ready
P_FRAME#    Inp    // PCI Bus Initiator Ready
DQS8        Inp    // SDRAM Data Strobe

```

// The following pins are tristateable, but can not be read back

```

TXD        Inp    // Serial Transmit Data
SFRM       Inp    // Serial Frame
SSCKO      Inp    // Serial Port Clock Out
ALE         Out,Lo // PBI Address Latch Enable
ADS#       Out,Hi // PBI Address Strobe
W/R#       Out,Hi // PBI Write/Read
FWE#       Out,Hi // PBI Flash Write Enable
DEN#       Out,Hi // PBI Data Enable
BLAST#     Out,Hi // PBI Burst Last
PB_RST#    Out,Hi // PBI Peripheral Bus Reset
HOLDA      Out,Lo // PBI Hold Acknowledge
PB_CLK     Out,Lo // PBI Peripheral Bus Clock
P_REQ#     Inp    // PCI Bus Request
P_INTA#    Out,Hi // PCI Interrupt Output
P_INTB#    Out,Hi // PCI Interrupt Output
P_INTC#    Out,Hi // PCI Interrupt Output
P_INTD#    Out,Hi // PCI Interrupt Output

```

// 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.

```

HPI#       Inp    // High Priority Interrupt
PWRDELAY   Inp    // Power Fail Delay
RXD        Inp    // Serial Receive Data
SSCKI      Inp    // Serial Port Clock In
XINT0#     Inp    // External Interrupt Request
XINT1#     Inp    // External Interrupt Request
XINT2#     Inp    // External Interrupt Request

```

XINT3#	Inp	// External Interrupt Request
HOLD	Inp	// PBI HOLD
P_M66EN	Inp	// PCI 66MHz enable
P_GNT#	Inp	// PCI Bus Grant input
P_RST#	Inp	// PCI Reset Input
P_IDSEL	Inp	// PCI Initialization Device Select
P_CLK	Inp	// PCI Clock Input
RCVENI#	Inp	// SDRAM Receive Enable Input

**Sample File JTAG425.INI:**

```
// Description file for Intel IXP425
Target: Generic Target
// Adapt this file carefully to your design!!
// 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 #.

// The following pins are complete bidirectional pins
// The direction of each pin can be set independent of the other pins
// For Flash Programming these pins must be set to output
EX_ADDR23      Out,Lo // Expansion Bus Address
EX_ADDR22      Out,Lo //
EX_ADDR21      Out,Lo //
EX_ADDR20      Out,Lo //
EX_ADDR19      Out,Lo //
EX_ADDR18      Out,Lo //
EX_ADDR17      Out,Lo //
EX_ADDR16      Out,Lo //
EX_ADDR15      Out,Lo //
EX_ADDR14      Out,Lo //
EX_ADDR13      Out,Lo //
EX_ADDR12      Out,Lo //
EX_ADDR11      Out,Lo //
EX_ADDR10      Out,Lo //
EX_ADDR9       Out,Lo //
EX_ADDR8       Out,Lo //
EX_ADDR7       Out,Lo //
EX_ADDR6       Out,Lo //
EX_ADDR5       Out,Lo //
EX_ADDR4       Out,Lo //
EX_ADDR3       Out,Lo //
EX_ADDR2       Out,Lo //
EX_ADDR1       Out,Lo //
EX_ADDR0       Out,Lo //
```



// The following pins are complete bidirectional pins  
 // These pins are switched between output/active and input/tristate during  
 // programming of Flash-EPROMs

EX_DATA15	Inp	// Expansion Bus Data
EX_DATA14	Inp	//
EX_DATA13	Inp	//
EX_DATA12	Inp	//
EX_DATA11	Inp	//
EX_DATA10	Inp	//
EX_DATA9	Inp	//
EX_DATA8	Inp	//
EX_DATA7	Inp	//
EX_DATA6	Inp	//
EX_DATA5	Inp	//
EX_DATA4	Inp	//
EX_DATA3	Inp	//
EX_DATA2	Inp	//
EX_DATA1	Inp	//
EX_DATA0	Inp	//

// The following pins are tristateable, but can not be read back  
 // These pins are modified during flash programming

EX_CS7#	Out,Hi	// Expansion Bus Chip Select
EX_CS6#	Out,Hi	//
EX_CS5#	Out,Hi	//
EX_CS4#	Out,Hi	//
EX_CS3#	Out,Hi	//
EX_CS2#	Out,Hi	//
EX_CS1#	Out,Hi	//
EX_CS0#	Out,Hi	//
EX_RD#	Out,Hi	// Expansion Bus Read Strobe
EX_WR#	Out,Hi	// Expansion Bus Write Strobe

// 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 an input.

HSS_TXFRAME1	Inp	// High Speed Serial Transmit Frame 1
HSS_TXCLK1	Inp	// High Speed Serial Transmit Clock 1
HSS_RXFRAME1	Inp	// High Speed Serial Receive Frame 1
HSS_RXCLK1	Inp	// High Speed Serial Receive Clock 1
HSS_TXFRAME0	Inp	// High Speed Serial Transmit Frame 0
HSS_TXCLK0	Inp	// High Speed Serial Transmit Clock 0
HSS_RXFRAME0	Inp	// High Speed Serial Receive Frame 0

HSS_RXCLK0	Inp	// High Speed Serial Receive Clock 0
PCI_GNT0#	Inp	// PCI Arbitration Grant
PCI_REQ0#	Inp	// PCI Arbitration Request
PCI_SERR#	Inp	// PCI System Error
PCI_PERR#	Inp	// PCI Parity Error
PCI_DEVSEL#	Inp	// PCI Device Select
PCI_STOP#	Inp	// PCI Stop
PCI_TRDY#	Inp	// PCI Target Ready
PCI_IRDY#	Inp	// PCI Initiator Ready
PCI_FRAME#	Inp	// PCI Cycle Frame
PCI_PAR	Inp	// PCI Parity
PCI_CBE3#	Inp	// PCI Command/Byte Enable
PCI_CBE2#	Inp	//
PCI_CBE1#	Inp	//
PCI_CBE0#	Inp	//
PCI_AD31	Inp	// PCI ADdress/Data
PCI_AD30	Inp	//
PCI_AD29	Inp	//
PCI_AD28	Inp	//
PCI_AD27	Inp	//
PCI_AD26	Inp	//
PCI_AD25	Inp	//
PCI_AD24	Inp	//
PCI_AD23	Inp	//
PCI_AD22	Inp	//
PCI_AD21	Inp	//
PCI_AD20	Inp	//
PCI_AD19	Inp	//
PCI_AD18	Inp	//
PCI_AD17	Inp	//
PCI_AD16	Inp	//
PCI_AD15	Inp	//
PCI_AD14	Inp	//
PCI_AD13	Inp	//
PCI_AD12	Inp	//
PCI_AD11	Inp	//
PCI_AD10	Inp	//
PCI_AD9	Inp	//
PCI_AD8	Inp	//
PCI_AD7	Inp	//
PCI_AD6	Inp	//
PCI_AD5	Inp	//
PCI_AD4	Inp	//

---

PCI_AD3	Inp	//
PCI_AD2	Inp	//
PCI_AD1	Inp	//
PCI_AD0	Inp	//
SDM_DATA31	Inp	// SDRAM Data
SDM_DATA30	Inp	//
SDM_DATA29	Inp	//
SDM_DATA28	Inp	//
SDM_DATA27	Inp	//
SDM_DATA26	Inp	//
SDM_DATA25	Inp	//
SDM_DATA24	Inp	//
SDM_DATA23	Inp	//
SDM_DATA22	Inp	//
SDM_DATA21	Inp	//
SDM_DATA20	Inp	//
SDM_DATA19	Inp	//
SDM_DATA18	Inp	//
SDM_DATA17	Inp	//
SDM_DATA16	Inp	//
SDM_DATA15	Inp	//
SDM_DATA14	Inp	//
SDM_DATA13	Inp	//
SDM_DATA12	Inp	//
SDM_DATA11	Inp	//
SDM_DATA10	Inp	//
SDM_DATA9	Inp	//
SDM_DATA8	Inp	//
SDM_DATA7	Inp	//
SDM_DATA6	Inp	//
SDM_DATA5	Inp	//
SDM_DATA4	Inp	//
SDM_DATA3	Inp	//
SDM_DATA2	Inp	//
SDM_DATA1	Inp	//
SDM_DATA0	Inp	//
USB_D-	Inp	//
USB_D+	Inp	//
GPIO15	Inp	// CLKOUT
GPIO14	Inp	//
GPIO13	Inp	//
GPIO12	Inp	//
GPIO11	Inp	//

```

GPIO10      Inp    //
GPIO9       Inp    //
GPIO8       Inp    //
GPIO7       Inp    //
GPIO6       Inp    //
GPIO5       Inp    //
GPIO4       Inp    //
GPIO3       Inp    //
GPIO2       Inp    //
GPIO1       Inp    //
UTP_IP_ADDR4 Inp    // Utopia Receive PHY Address
UTP_IP_ADDR3 Inp    //
UTP_IP_ADDR2 Inp    //
UTP_IP_ADDR1 Inp    //
UTP_IP_ADDR0 Inp    //
UTP_OP_ADDR4 Inp    // Utopia Transmit PHY Address
UTP_OP_ADDR3 Inp    //
UTP_OP_ADDR2 Inp    //
UTP_OP_ADDR1 Inp    //
UTP_OP_ADDR0 Inp    //

// The following pins are tristateable, but can not be read back
HSS_TXDATA1 Inp    // High Speed Serial Transmit Data
HSS_TXDATA0 Inp    //
PCI_INTA#   Inp    // PCI Interrupt
PCI_GNT3#   Inp    // PCI Arbitration Grant
PCI_GNT2#   Inp    //
PCI_GNT1#   Inp    //
SDM_CLKOUT  Out,Lo // SDRAM Clock
SDM_CKE     Out,Lo // SDRAM Clock Enable
SDM_WE#     Out,Hi // SDRAM Write Enable
SDM_RAS#    Out,Hi // SDRAM Row Address Strobe
SDM_CAS#    Out,Hi // SDRAM Column Address Strobe
SDM_BA1     Out,Lo // SDRAM BAnk Address
SDM_BA0     Out,Lo //
SDM_DQM3    Inp    // SDRAM Data Bus Mask
SDM_DQM2    Inp    //
SDM_DQM1    Inp    //
SDM_DQM0    Inp    //
SDM_CS1#    Out,Hi // SDRAM Chip Select
SDM_CS0#    Out,Hi //
SDM_ADDR12  Out,Lo // SDRAM Address
SDM_ADDR11  Out,Lo //

```

---

```

SDM_ADDR10    Out,Lo //
SDM_ADDR9     Out,Lo //
SDM_ADDR8     Out,Lo //
SDM_ADDR7     Out,Lo //
SDM_ADDR6     Out,Lo //
SDM_ADDR5     Out,Lo //
SDM_ADDR4     Out,Lo //
SDM_ADDR3     Out,Lo //
SDM_ADDR2     Out,Lo //
SDM_ADDR1     Out,Lo //
SDM_ADDR0     Out,Lo //
EX_ALE        Out,Lo // Expansion Bus Address Latch Enable
RTS1#         Inp    //
TXDATA1       Inp    //
RTS0#         Inp    //
TXDATA0       Inp    //
UTP_IP_FCO    Inp    // Utopia Receive Data Flow Control Output
UTP_OP_SOC    Inp    // Utopia Transmit Start of Cell
UTP_OP_DATA7  Inp    // Utopia Transmit Data
UTP_OP_DATA6  Inp    //
UTP_OP_DATA5  Inp    //
UTP_OP_DATA4  Inp    //
UTP_OP_DATA3  Inp    //
UTP_OP_DATA2  Inp    //
UTP_OP_DATA1  Inp    //
UTP_OP_DATA0  Inp    //
UTP_OP_FCO    Inp    // Utopia Transmit Data Flow Control Output
ETH_TXDATA1_3 Inp    //
ETH_TXDATA1_2 Inp    //
ETH_TXDATA1_1 Inp    //
ETH_TXDATA1_0 Inp    //
ETH_TXEN1     Inp    //
ETH_TXDATA0_3 Inp    //
ETH_TXDATA0_2 Inp    //
ETH_TXDATA0_1 Inp    //
ETH_TXDATA0_0 Inp    //
ETH_TXEN0     Inp    //

```

// 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.

```

HSS_RXDATA1   Inp    //
HSS_RXDATA0   Inp    //

```

```

PCI_CLKIN      Inp  //
PCI_REQ3#     Inp  //
PCI_REQ2#     Inp  //
PCI_REQ1#     Inp  //
PCI_IDSEL     Inp  //
EX_RDY3#     Inp  //
EX_RDY2#     Inp  //
EX_RDY1#     Inp  //
EX_RDY0#     Inp  //
EX_CLK       Inp  //
EX_IOWAIT#   Inp  //
CTS1#        Inp  //
RXDATA1      Inp  //
CTS0#        Inp  //
RXDATA0      Inp  //
UTP_IP_DATA7 Inp  // Utopia Receive Data
UTP_IP_DATA6 Inp  //
UTP_IP_DATA5 Inp  //
UTP_IP_DATA4 Inp  //
UTP_IP_DATA3 Inp  //
UTP_IP_DATA2 Inp  //
UTP_IP_DATA1 Inp  //
UTP_IP_DATA0 Inp  //
UTP_IP_SOC   Inp  // Utopia Receive Start of Cell
UTP_IP_FCI   Inp  // Utopia Receive Data Flow Control Input
UTP_IP_CLK   Inp  // Utopia Receive Data Clock Input
UTP_OP_FCI   Inp  // Utopia Transmit Data Flow Control Input
UTP_OP_CLK   Inp  // Utopia Transmit Data Clock Input
RESET_IN#    Inp  //
BYPASS_CLK   Inp  //
ETH_COL1     Inp  //
ETH_CRS1     Inp  //
ETH_RXCLK1   Inp  //
ETH_RXDATA1_3 Inp  //
ETH_RXDATA1_2 Inp  //
ETH_RXDATA1_1 Inp  //
ETH_RXDATA1_0 Inp  //
ETH_RXDV1    Inp  //
ETH_TXCLK1   Inp  //
ETH_COL0     Inp  //
ETH_CRS0     Inp  //
ETH_RXCLK0   Inp  //
ETH_RXDATA0_3 Inp  //
    
```

ETH_RXDATA0_2	Inp	//
ETH_RXDATA0_1	Inp	//
ETH_RXDATA0_0	Inp	//
ETH_RXDV0	Inp	//
ETH_TXCLK0	Inp	//

### **1.12. Supported flash devices**

Type JTAGxxx /LIST [optionlist]

to get a online list of all flash types which could be used with the /DEVICE= option.

See separate file JTAG\_V4xx\_FLASHES.pdf to get a complete list of supported flash types.



## **2. JTAGxxx Parameter Description**

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

```
JTAGxxx --- JTAG utility for Intel XScale
Copyright © FS FORTH-SYSTEME GmbH, Breisach
Version 4.xx of mm/dd/yyyy
```

Programming of Flash-EPROMs and hardware tests on targets with the Intel XScale.

The JTAG-Booster is needed to connect the parallel port of the PC to the JTAG port of the Intel XScale.

Usage: JTAGxxx /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
/PI2C      : Program an I2C Device with file
/RI2C      : Read an I2C Device to file
/VI2C      : Verify an I2C Device with file
/DUMPI2C   : Make a dump of an I2C 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	/CS4
/CS5	/NOCS	/NOWRSETUP	/TOP	/BYTE-MODE
/BM	/PAUSE	/P	/NODUMP	/NOERASE
/LATTICE	/ERASEALL	/LPT1	/LPT2	/LPT3
/LPT-BASE=	/32BIT	/16BIT	/8BIT	/NOMAN
/LENGTH=	L=	/FILE-OFFSET=	/FO=	/OFFSET=
/O=	/DELAY=	/DEVICE-BASE=	/DB=	/DRIVER=
/DATA-MASK=	/DM=	/IROFFS=	/CPUPOS=	/DEVICE=
/PIN=	/I2CCLK=	/I2CDAT=	/I2CDATI=	/I2CDATO=
I2CBIG	/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 `JTAGxxx.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 1.11 "Initialization file `JTAGxxx.INI`").

Note: The initialization file is not loaded for the functions `/SAMPLE` (chapter 2.11) and `/SNAP` (chapter 2.12).

Default: `/INI=JTAGxxx.INI`

`/LATTICE /WIGGLER /PLS /TRITON`

Besides the standard JTAG-Booster interface there are several simple "Parallel-Port-JTAG" interfaces supported. With this interfaces the programming performance, of course, is reduced.

**/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, Win2000 or WindowsXP, 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 Intel XScale instruction register within the JTAG chain. In most cases this option is not needed.

Default: /IROFFS=0

**/CPUPOS=**

Specifies the position of the Intel XScale within the JTAG chain.

Default: /CPUPOS=0

## **2.1. Program a Flash Device**

**Usage:** JTAGxxx /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 1.12 "Supported flash 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.

/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.

/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.

---

**/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=hhhhh<sup>1</sup>**

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=hhhhh**

The programming starts at an offset of hhhhh 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=hhhhh**

If `FILE-OFFSET` is specified, the first hhhhh bytes of the file are skipped and not programmed to target.

Default: `/FILE-OFFSET=0`

Abbreviation: `/FO=`

<sup>1</sup>hhhhh=number base is hex

**/LENGTH=hhhhh**

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.

**/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 /CS4 /CS5**

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 1.11 "Initialization file JTAGxxx.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 Intel XScale 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:**

JTAGxxx /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.

JTAGxxx /P CE.ROM /32BIT /CS1

This example programs the file CE.ROM to the 32 Bit Flash-EPROM connected to CS1#.

## **2.2. Read a Flash Device to file**

**Usage:** JTAGxxx /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 1.12 "Supported flash devices".

### **Options:**

/DEVICE=devicename

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=hhhhh<sup>2</sup>

See function /P (Chapter 2.1)

/OFFSET=hhhhh

Reading of the flash memory starts at an offset of hhhhh 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=

<sup>2</sup>hhhhh=number base is hex



**/TOP**

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

**/LENGTH=hhhhh**

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 /CS4 /CS5**

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:**

`JTAGxxx /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:** JTAGxxx /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 1.12 "Supported flash devices".

**Options:**

/DEVICE=devicename

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=hhhhh`

See function `/P` (Chapter 2.1)

`/NODUMP`

See function `/P` (Chapter 2.1)

`/CS0 /CS1 /CS2 /CS3 /CS4 /CS5`

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:**

`JTAGxxx /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:** JTAGxxx /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: /32BIT

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

/DEVICE-BASE=hhhhh<sup>3</sup>  
 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=hhhhh  
 Default: /LENGTH=100  
 Abbreviation: /L=

/CS0 /CS1 /CS2 /CS3 /CS4 /CS5  
 See function /P (Chapter 2.1)  
 Default: /CS0

<sup>3</sup>hhhhh=number base is hex

**Example:**

JTAGxxx /DUMP

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

## **2.5. Program an I<sup>2</sup>C-Device**

**Usage:** JTAGxxx /PI2C filename [/I2CBIG] [optionlist]

The specified file is programmed to an I<sup>2</sup>C-Device (i.e. a serial EEPROM) connected to pins of the CPU. Finally a complete verify is done. If the verify fails, the contents of the I<sup>2</sup>C-Device is written to a file with the extension DMP.

Two methods to connect the I<sup>2</sup>C-Device to the CPU are supported. The first method is to use two CPU pins, one pin for clock output (I2CCLK) and one pin for serial data input and output (I2CDAT). The second method is to use one pin for clock output (I2CCLK), one for serial data input (I2CDATI) and one for serial data output (I2CDATO).

### **Options:**

**/I2CBIG**

Specify this option if there is a device which needs a three byte address instead of a two byte address.

**This option must be the first option after the filename.**

**/DEVICE-BASE=hhhhhh**

This option specifies an I<sup>2</sup>C device starting address. The default values are chosen to access an serial EEPROM.

Default: /DEVICE-BASE=5000 (if option /I2CBIG omitted)

Default: /DEVICE-BASE=500000 (if option /I2CBIG specified)

**/OFFSET=hhhhhh**

The programming starts at an offset of hhhhhh relative to the start address of the I<sup>2</sup>C-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=hhhhh`

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 I<sup>2</sup>C-Device is written to a file with the extension .DMP. With option `/NODUMP` you can suppress this feature.

`/I2CCLK=pin_name`

Specifies the CPU pin used for serial clock output.

`/I2CDAT=pin_name`

Specifies the CPU pin used for serial data input and output. 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 `/I2CDATO=` and `/I2CDATI=` .

`/I2CDATO=pin_name`

Specifies the CPU pin used for serial data output. Pin\_name must specify a output pin otherwise an error message occurs.

`/I2CDATI=pin_name`

Specifies the CPU pin used for serial data input. Pin\_name must specify a input pin otherwise an error message occurs.

**Example:**

`JTAGxxx /PI2C EEPROM.CFG /I2CCLK=FLAG0 /I2CDAT=FLAG1`

This example loads the file EEPROM.CFG to a serial EEPROM connected to the pins FLAG0 and FLAG1 of the Intel XScale

## **2.6. Read an I<sup>2</sup>C-Device to file**

**Usage:** JTAGxxx /RI2C filename [/I2CBIG] /L=hhhhhh [optionlist]

The contents of an I<sup>2</sup>C-Device (i.e. a serial EEPROM) is read and written to a file. The option /LENGTH= must be specified.

### **Options:**

/I2CBIG

**This option must be the first option after the filename.**

See function /PI2C (Chapter 2.5)

/DEVICE-BASE=hhhhhh

See function /PI2C (Chapter 2.5)

/OFFSET=hhhhhh

Reading of the I<sup>2</sup>C-Device starts at an offset of hhhhhh relative to the start address of the I<sup>2</sup>C-Device.

Default: /OFFSET=0

Abbreviation: /O=

/LENGTH=hhhhhh

The number of read bytes must be specified otherwise an error message occurs.

Abbreviation: /L=

/I2CCLK=pin\_name

See function /PI2C (Chapter 2.5)

/I2CDAT=pin\_name

See function /PI2C (Chapter 2.5)

/I2CDATO=pin\_name

See function /PI2C (Chapter 2.5)

/I2CDATI=pin\_name

See function /PI2C (Chapter 2.5)



**Example:**

JTAGxxx /I2C EEPROM.CFG /I2CCLK=GP26 /I2CDAT=GP27 /L=100  
This example reads 256 bytes from a serial EEPROM to the file EEPROM.CFG.  
The serial EEPROM is connected to the pins CP26 and GP27 of the Intel XScale.

## **2.7. Verify an I<sup>2</sup>C-Device with file**

**Usage:** JTAGxxx /I2C filename [/I2CBIG] [optionlist]

The contents of an I<sup>2</sup>C-Device (i.e. a serial EEPROM) is compared with the specified file. If there are differences the contents of the I<sup>2</sup>C -Device is written to a file with the extension DMP.

### **Options:**

/I2CBIG

**This option must be the first option after the filename.**

See function /PI2C (Chapter 2.5)

/DEVICE-BASE=hhhhhh

See function /PI2C (Chapter 2.5)

/OFFSET=hhhhhh

See function /PI2C (Chapter 2.5)

/FILE-OFFSET=hhhhhh

See function /PI2C (Chapter 2.5)

/LENGTH=hhhhhh

See function /PI2C (Chapter 2.5)

/NODUMP

See function /PI2C (Chapter 2.5)

/I2CCLK=pin\_name

See function /PI2C (Chapter 2.5)

/I2CDAT=pin\_name

See function /PI2C (Chapter 2.5)

/I2CDATO=pin\_name

See function /PI2C (Chapter 2.5)

/I2CDAT=pin\_name

See function /I2C (Chapter 2.5)

**Example:**

JTAGxxx /I2C EEPROM.CFG /I2CCLK=GP26 /I2CDAT=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 Intel XScale.

## **2.8. Dump an I<sup>2</sup>C-Device**

**Usage:** JTAGxxx /DUMPI2C [/I2CBIG] [optionlist]

A Hex-Dump of an I<sup>2</sup>C-Device is printed on the screen, if not redirected to file or device.

### **Options:**

/I2CBIG

**This option must be the first option.**

See function /PI2C (Chapter 2.5)

/DEVICE-BASE=hhhhhh

See function /PI2C (Chapter 2.5)

/OFFSET=hhhhh<sup>4</sup>

The memory dump starts at an offset of hhhhh.

Default: /OFFSET=0

Abbreviation: /O=

/LENGTH=hhhhh

Default: /LENGTH=100

Abbreviation: /L=

/I2CCLK=pin\_name

Specifies the CPU pin used for serial clock output.

/I2CDAT=pin\_name

Specifies the CPU pin used for serial data input and output. 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 /I2CDATO= and /I2CDATI= .

/I2CDATO=pin\_name

Specifies the CPU pin used for serial data output. Pin\_name must specify a output pin otherwise an error message occurs.

<sup>4</sup>hhhhh=number base is hex

`/I2CDAT=pin_name`

Specifies the CPU pin used for serial data input. Pin\_name must specify a input pin otherwise an error message occurs.

**Example:**

`JTAGxxx /DUMPI2C /I2CCLK=FLAG0 /I2CDAT=FLAG1`

This example makes a memory dump of the first 100h bytes of a serial EEPROM connected to the CPU.

## **2.9. Toggle CPU pins**

**Usage:** JTAGxxx /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 Intel XScale 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 1.11 "Initialization file JTAGxxx.INI" can be used. If you type /PIN= without any pin declaration a list of the CPU pins is displayed.

/DELAY=dddddd<sup>5</sup>

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:**

JTAGxxx /BLINK /PIN=FLAG3 /DELAY=0

This example toggles the FLAG3 pin very fast which can be followed by the use of an oscilloscope.

<sup>5</sup>dddddd=number base is decimal

## **2.10. Polling CPU pins**

**Usage:** JTAGxxx /PIN? /PIN=pinname [optionlist]

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

Please Note: Not every pin of the Intel XScale 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 1.11 "Initialization file JTAGxxx.INI" can be used.

If you type /PIN= without any pin declaration a list of the CPU pins is displayed.

### **Example:**

JTAGxxx /PIN? /PIN=RESET#

This example samples the reset pin of the Intel XScale.

## **2.11. Polling CPU pins while the CPU is running**

**Usage:** JTAGxxx /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 1.11 "Initialization file JTAGxxx.INI" can be used. If you type /PIN= without any pin declaration a list of the CPU pins is displayed.

### **Example:**

JTAGxxx /SAMPLE /PIN=FLAG3

This example samples the state of the port pin FLAG3 while the Intel XScale is running.



**2.12. Show status of all CPU pins while the CPU is running****Usage:**            JTAGxxx /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 Intel PXA210/250

0 GPIO0	1 GPIO1	1 GPIO2	1 GPIO3
1 GPIO4	1 GPIO5	1 GPIO6	1 GPIO7
1 GPIO8	1 GPIO9	1 GPIO10	0 GPIO11
1 GPIO12	1 GPIO13	1 GPIO14	1 GPIO15
1 GPIO16	1 GPIO17	1 GPIO18	1 GPIO19
1 GPIO20	1 GPIO21	0 GPIO22	1 GPIO23
0 GPIO24	1 GPIO25	1 GPIO26	1 GPIO27
1 GPIO28	1 GPIO29	1 GPIO30	1 GPIO31
1 GPIO32	1 GPIO33	1 GPIO34	1 GPIO35
1 GPIO36	1 GPIO37	1 GPIO38	1 GPIO39
1 GPIO40	1 GPIO41	1 GPIO42	1 GPIO43
1 GPIO44	1 GPIO45	1 GPIO46	1 GPIO47
1 GPIO48	1 GPIO49	1 GPIO50	1 GPIO51
1 GPIO52	1 GPIO53	0 GPIO54	1 GPIO55
1 GPIO56	0 GPIO57	0 GPIO58	0 GPIO59
1 GPIO60	1 GPIO61	1 GPIO62	1 GPIO63
1 GPIO64	1 GPIO65	1 GPIO66	1 GPIO67
1 GPIO68	1 GPIO69	1 GPIO70	1 GPIO71
1 GPIO72	1 GPIO73	1 GPIO74	1 GPIO75
1 GPIO76	1 GPIO77	1 GPIO78	1 GPIO79
1 GPIO80	1 SCL	1 SDA	0 USB_N
0 USB_P	1 MMDAT	1 MMCMD	0 MD0
0 MD1	0 MD2	0 MD3	1 MD4
1 MD5	0 MD6	0 MD7	1 MD8
1 MD9	0 MD10	0 MD11	1 MD12
1 MD13	0 MD14	0 MD15	0 MD16
0 MD17	0 MD18	0 MD19	0 MD20
0 MD21	0 MD22	0 MD23	0 MD24
0 MD25	0 MD26	0 MD27	0 MD28
0 MD29	0 MD30	0 MD31	1 PWR_EN
1 RESET_OUT#	0 ACRESET#	1 RD/WR#	0 SDCLK0
1 SDCLK1	0 SDCLK2	0 SDCKE0	1 SDCKE1
1 SDCS0#	1 SDCS1#	1 SDCS2#	1 SDCS3#
0 DQM0	0 DQM1	0 DQM2	0 DQM3
1 SDCAS#	1 SDRAS#	1 WE#	1 OE#
1 CS0#	0 MA0	0 MA1	0 MA2
0 MA3	0 MA4	0 MA5	0 MA6
0 MA7	0 MA8	0 MA9	0 MA10
0 MA11	0 MA12	0 MA13	1 MA14
1 MA15	0 MA16	1 MA17	1 MA18
0 MA19	1 MA20	0 MA21	0 MA22

0 MA23	0 MA24	0 MA25	0 TEST
0 TESTCLK	1 VDD_FAULT#	1 BATT_FAULT#	0 BOOT_SEL0
0 BOOT_SEL1	0 BOOT_SEL2	0 RESET#	

### **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 XScale Onchip debugger is not used.
- The software assumes the following scheme for connecting the Flash-EPROM to the Intel XScale. Please contact us, if you have used a different method.

<b>PXA210/250, PXA255/26x signal</b>	<b>8 Bit Flash</b>	<b>16 Bit Flash</b>	<b>32 Bit Flash</b>
CS0# GPIO15/CS1# GPIO78/CS2# GPIO79/CS3# GPIO80/CS4# GPIO33/CS5#	CS#	CS#	CS#
OE#	OE#	OE#	OE#
WE#	WE#	WE#	WE#
MA0	A0	-	-
MA1	A1	A1	-
MA2..25	A2..25	A2..25	A2..25
MD0..7	D0..7	-	-
MD0..15	-	D0..15	-
MD0..31	-	-	D0..31

<b>IOP321 signal</b>	<b>8 Bit Flash</b>	<b>16 Bit Flash</b>	<b>32 Bit Flash</b>
PCE0# PCE1# PCE2# PCE3# PCE4# PCE5#	CS#	CS#	CS#
W/R#	OE#	OE#	OE#
FWE#	WE#	WE#	WE#
BE0#	A0	-	-
BE1#	A1	A1	-
A2, A3, AD4..25	A2..25	A2..25	A2..25
AD0..7	D0..7	-	-
AD0..15	-	D0..15	-
AD0..31	-	-	D0..31

<b>IXP425 signal</b>	<b>8 Bit Flash</b>	<b>16 Bit Flash</b>	<b>32 Bit Flash</b>
EX_CS0#..EX_CS1# EX_CS2#..EX_CS3# EX_CS4#..EX_CS5# EX_CS6#..EX_CS7#	CS#	CS#	CS#
EX_RD#	OE#	OE#	OE#
EX_WR#	WE#	WE#	WE#
EX_ADDR0	A0	-	-
EX_ADDR1	A1	A1	-
EX_ADDR2..23	A2..23	A2..23	A2..23
EX_DATA0..7	D0..7	-	-
EX_DATA0..15	-	D0..15	-
EX_DATA0..31	-	-	D0..31

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

#### **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

Maximum conversion size is 256 kBytes. A 4<sup>th</sup> 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[FFFFFFFF]:
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 FFFFFFFF 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 1<sup>st</sup> 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



- exit the kcenter program
- Now you can deinstall the Kithara Package with:  
Settings - Control Panel.  
All unused parts must be removed.
- Reboot your PC
- Now you can install the Kithara 6.x as described above.

#### **5.4. De-Installation version 5.x/6.x:**

For deinstallation of the runtime version of the "Kithara DOS-Enabler Version 5.x/6.x":

- use: Settings - Control-Panel - Add/Remove Programs  
and remove the  
"FS FORTH-SYSTEME WinNT Support"  
and/or  
"WinNT Support for JTAG-Booster and FLASH166"
- Reboot your PC