JTAG-Booster for

ARM7TDMI

Forth-Systeme GmbH

P.O: Box 1103 ● D-79200 Breisach, Germany
Kueferstrasse 8 ● D-79206 Breisach, Germany
Tel. +49 (7667) 908-0 ● Fax +49 (7667) 908-200
sales@fsforth.de ● http://www.fsforth.de

Release of Document:	April 30, 2003
Author:	Dieter Fögele
Filename:	JTGarm7a.doc
Program Version:	4.xx

Table of Contents

1. General

The program JTAGARM7.EXE uses the JTAG port of the ARM7TDMI embedded microprocessor 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 stop the target and show the register configuration

This tool uses the EmbeddedICE macro cell of the ARM7TDMI core. If there is no target RAM available, 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 into initially unprogrammed memories.

If there is target RAM available, the flash programming algorithm is loaded into target RAM to speed up programming performance. In this case the flash is programmed with about **100kByte/sec**.

This tool is a generic tool for all ARM7TDMI based systems. By adapting the configuration file customers can adapt the tool to their own CPU and hardware design.

This tool does not support boundary scan, even if the used chip does have a boundary scan chain. If you need a tool, which combines usage of the boundary scan chain of a specific chip **and** the EmbeddedICE macro cell of the ARM7TDMI in one program, please contact us.

For latest documentation and a list of explicit supported and tested target CPUs please refer to the file README.TXT on the distribution disk.

1.1. Ordering Information

The following related products are available

- 9005 JTAG-Booster ARM7TDMI, 3.3V,
  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

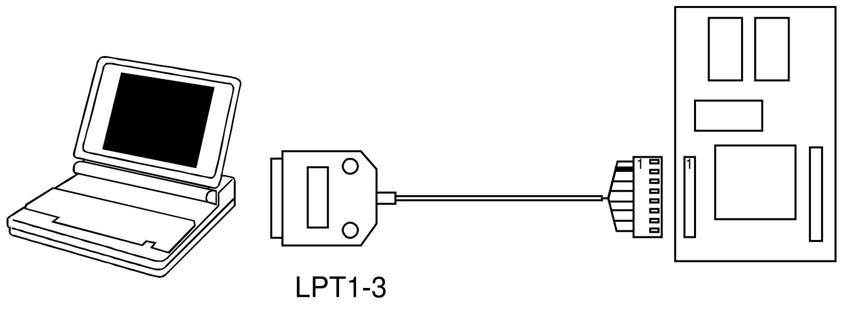
1.3. Contents of Distribution Disk

- JTAGARM7.EXE JTAGARM7.OVL — Tool for ARM7TDMI
- 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, Windows 2000 and Windows XP. See chapter 5 "Support for Windows NT, Windows 2000 and Windows XP"
- JTAG_V4xx_FLASHES.pdf — List of all supported Flash devices
- README.txt — Release notes, new features, known problems
- TARGETS — This subdirectory holds the configuration file and batch files for some specific target CPUs. Actually the following CPUs are supported:

NS7520	: NetSilicon NS7520, little and big endian
NET40	: NetSilicon NET+40, little and big endian
NET50	: NetSilicon NET+50, little and big endian

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 white thick cable.

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

The standard version of the JTAG-BOOSTER for this target type operates at 3.3V (Ordering number 285). If your target operates at a different voltage, please contact us.

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:

```
JTAGARM7 /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 JTAGARM7, press CTRL-Break-Key.

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

1.5. First Example

In the following simple example it is assumed that the JTAG-Booster is connected to LPT1 of your PC, target power is on and the target runs a "good" program.

Typing

```
        JTAGARM7  /P  MYAPP.BIN /CFG=NS7520l.cfg
```

at the DOS prompt results in the following output:

```
        JTAGARM7 --- JTAG utility for ARM7TDMI
        Copyright © FS FORTH-SYSTEME GmbH, Breisach
        Version 4.xx of mm/dd/yyyy

(1)     Using LPT at I/O-address 0378h
(2)     JTAG-Booster detected

(3)     1 Device detected in JTAG chain
(4)       Device 0: IDCODE=3F0F0F0F  ARM7TDMI, Revision 3
(5)     Sum of instruction register bits  : 4
(6)     CPU position                      : 0
(7)     Instruction register offset       : 0

(8)     Target is running
(9)     Target stopped by debug request at 0800028C
(10)    Configuration loaded from file NS7520L.CFG
(11)    Target: NetSilicon NS7520, Little Endian, 2003/04/29
(12)    GEN_SSR (0xFFB00004) = 288007FF
(13)    CPU is a NetSilicon NS7520-1
(14)    Target is Little Endian
        Last Reset caused by External Reset
        Last Reset caused by PLL updated
(15)    SYSCLK=55.296Mhz @ FXTAL=18.432MHz
(16)    Setting Chip Selects
        CS0 at 0x00000000
        CS1 at 0x02000000, use option /DEVICE-BASE=02000000 to activate CS1
        CS2 at 0x04000000, use option /DEVICE-BASE=04000000 to activate CS2
        CS3 at 0x06000000, use option /DEVICE-BASE=06000000 to activate CS3
        CS4 at 0x08000000, use option /DEVICE-BASE=08000000 to activate CS4
        Looking for a known flash device. Please wait..
(17)    Dual STM 29W800B detected
(18)    Bus size is 32 bit
(19)    Loading Flash Algorithm (328 Bytes)
(20)    Erasing Flash-EPROM Block #:0  1  2
        Programming File MYAPP.BIN (65536 Bytes)
        65536 Bytes programmed
(21)    Calculating checksum  0031B460
        Programming successful

        Erase Time          :       0.2 sec
        Programming Time    :       1.9 sec
```

If this examples works successfully on your target add the options /DRIVER=1 and /FAST or /FAST= to speed up the programming performance. (In the example above these options reduce the programming time down to 0.5 sec!!)

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

(2) A JTAG-Booster connected to the parallel port was found.

(3) The JTAG chain is analyzed. There may be several parts in the JTAG chain. The chain is analyzed and all parts except the ARM7TDMI are switched to bypass mode.

(4) Actually only the Revision 3a of the ARM7TDMI debug interface is supported.

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

(6) The position of the ARM7TDMI in the JTAG chain is checked.

(7) The position of the JTAG instruction register of the ARM7TDMI is checked

(8) In this example the target is regularly running when starting the tool JTAGARM7.EXE.

(9) For using the debug interface and flash programming the target execution must be stopped.

(10) The configuration file NS7520L.CFG is opened and the commands in this file are interpreted. The lines (11) to (16) are generated while executing the commands of the configuration file.

(11) This line is used to identify the used configuration file

(12) The NetSilicon's NET+ARM CPUs do have an identification register (mapped to the standard memory space, instead of being accessible via the coprocessors interface). The raw register value is printed here.

(13) The value of the identification register is checked and printed as a readable CPU type.

(14) The endianess of the target is checked and printed here.

(15) The value of the PLL configuration register is read and translated to a readable text.

(16) One of the most important tasks of the configuration file is to configure the chip select unit of the microcontroller to give access to all available chip selects.

(17) Two Flash-EPROMs STM 29W800B selected by chip select CS0# are found.

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

(19) The flash programming algorithm is loaded to target memory.

(20) In this example three blocks must be erased.

(21) A 32 bit checksum over the programmed flash address range is calculated and compared with the estimated value.

1.6. 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.

1.7. 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 /PPJARM /PLS /UNCBAS) 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\JTAGARM7.OVL**
  The overlay file JTAGARM7.OVL must be in the same directory as JTAGARM7.EXE.

- **Configuration file XYZ not found.**
  The file specified with the option /CFG= 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 /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 loading Flash Algorithm. Check Target RAM and CFG-File.**
  For higher programming performance the flash programming algorithm is loaded into target memory, if the keyword RAM_BASE is available in the configuration file. If there is a problem with the target RAM, this error message is printed.

- **illegal function:**
  The first parameter of the command line must be a valid function. See chapter 2 “JTAGARM7 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 “JTAGARM7 Parameter Description” for a list of supported options.

- **illegal Flash Type:**
  The name specified with the option /DEVICE= must be one of the list of chapter 1.9 ”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 1.8 “Configuration file JTAGARM7.CFG”).

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

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

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

- **Missing parameter**
  There is a syntax error in the configuration file. Please check your configuration file. (See chapter 1.8 “Configuration file JTAGARM7.CFG”

- **More than 9 devices in the JTAG chain or TDI 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 TDI pin stuck at high level**
  A stream of 32 high bits was detected on the pin TDI. TDI may stuck at high level. Check the connection to your target. Check the target power. Check the target’s TDO pin. Check the target's reset status.

- **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 ARM7TDMI**
  The option /CPUPOS= points to a part not a ARM7TDMI

- **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 4 bits for a ARM7TDMI**
  The sum of all instruction register bits in the JTAG chain does not fit to the ARM7TDMI. 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 is in Thumb mode. Not supported.**
  Thumb mode is actually not supported. If the target runs in thump mode, the target can not be stopped.

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

- **Target out of Sync. Try again**
  During flash programming the communication between host and target has broken. This may be a result of using the option /FAST or /FAST=. Please omit this options or increase the value of the option /FAST=. Another reason could be an unreliable target RAM. To overcome this problem you may specify the option /NORAM to avoid using the target's RAM. The programming performance is reduced dramatically in this case.

- **Tests Aborted**
  During loading the configuration file (specified by the option /CFG= or the file JTAGARM7.CFG) there was an error condition (a line with the command MEM_TESTxx"). The succeeding line with the command TEST_STOP, forces a program abort.

- **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 ARM7TDMI in the JTAG chain**
  No ARM7TDMI was found in the JTAG chain. Check the target power. Try with option /DRIVER=4 again.

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

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

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

1.8. Configuration file JTAGARM7.CFG

The configuration file is intended to configure the target, especially the target's memory configuration. Furthermore the configuration file can be used to do hardware tests and to analyse the target's current register settings.

This file **must be carefully adapted** to your design with the ARM7TDMI. Use the examples in the subdirectory \TARGETS as a template.

When the program JTAGARM7.EXE is started it scans the current directory for an existing configuration file named JTAGARM7.CFG. You may also specify the configuration file with the option /CFG= . If the specified file isn't found, the program aborts with an error message.

1.8.1. Valid Keywords

Keyword	**Function**
//	The remainder of the line is treated as an comment
TARGET: target: Target:	Prints an identification line, inclusive the keyword **Target:**
INFO: info: Info:	Similar as keyword TARGET:, but the keyword itself is not printed
MEM_WRITE32 MEM_WRITE16 MEM_WRITE8	Write to memory. Use read- modify-write if necessary.
MEM_INFO32" MEM_INFO16" MEM_INFO8"	Read memory and mask it with the and mask. Print a message, if the resulting bit pattern is identical with the specified.
MEM_TEST32" MEM_TEST16" MEM_TEST8"	Read memory and mask it with the and mask. Print an error message, if the resulting bit pattern is different to the specified. Set an error status. Stop execution at the next TEST_STOP keyword.
MEM_PRT32"	Read memory and print the read value together with a message
TEST_STOP	The interpretation of the configuration file is stopped and the program is aborted, if the error status was set by an previous error condition.
RAM_BASE	Specifies the target address, where the flash programming algorithm should be loaded to.
DEBUGPC_RESET	Reset the program counter during debug. This keyword is needed to fix a problem with the NetSilicon's debugger implementation.

1.8.2. General Syntax rules

- All keywords are case sensitive.
- Number base is hexadecimal.
- 32 bit constants must identified by a trailing dot.
- Letters within numbers must be in upper case.
- All keywords and numbers must be separated by an space or an end of line.

1.9. Supported flash devices

Type JTAGARM7 /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. JTAGARM7 Parameter Description

When you start JTAGARM7.EXE without any parameters the following help screen with all possible functions and options is displayed:

```
JTAGARM7 --- JTAG utility for the ARM7TDMI
Copyright © FS FORTH-SYSTEME GmbH, Breisach
Version 4.xx of mm/dd/yyyy

Programming of Flash-EPROMs and hardware tests on targets with the
ARM7TDMI.

The JTAG-Booster is needed to connect the parallel port of the PC
to the JTAG port of the ARM7TDMI.

Usage: JTAGARM7 /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
        /LIST       : Print a list of supported Flash devices

Supported Options:
/BIG         /NOMAN       /NORESET       /FAST           /FAST=
/TOP         /BYTE-MODE   /BM            /PAUSE          /P
/NODUMP      /NOERASE     /ERASEALL      /LATTICE        /PPJARM
/PLS         /UNCBAS      /LPT1          /LPT2           /LPT3
/LPT-BASE=   /32BIT       /16BIT         /8BIT           /NOMAN
/LENGTH=     /L=          /FILE-OFFSET=  /FO=            /OFFSET=
/O=          /DELAY=      /DEVICE-BASE=  /DB=            /DRIVER=
IROFFS=      /CPUPOS=     /DEVICE=       /OUT=           /CFG=
```

The following options are valid for most functions:

/BIG
This option switches the byte ordering to big endian mode. This option must fit to the target's endianess. Normally the target is configured to the right endianess after reset. In some cases, the endianess can be changed within the configuration file.
Default: Little Endian

/CPUPOS=
Specifies the position of the ARM7TDMI within the JTAG chain.
Default: /CPUPOS=0

/CFG=file
An configuration file may be specified. By default the current directory is searched for the file JTAGARM7.CFG. If this file is not found and no configuration file is specified in the command line, the target's memory is left unchanged, i.e. the target's register remain in the reset state (see also chapter 1.8 “Configuration file JTAGARM7.”).
Default: /CFG=JTAGARM7.CFG

/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

/IROFFS=
Specifies the position of the ARM7TDMI instruction register within the JTAG chain. In most cases this option is not needed.
Default: /IROFFS=0

/LATTICE /PPJARM /PLS /UNCBAS
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 the printer port may be specified instead of the logical printer name. Useful option, if you work with WinNT, Win2000 or WinXP, 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.

/NORESET
Normally the program forces a target reset at program termination. (Since the JTAG-Booster does not have a pin to be connected to the target's reset, the user is asked to press the target's reset button instead.)

/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

2.1. Programming a Flash Device

Usage: JTAGARM7 /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, since there is a 32 bit checksum calculated after programming and the result is compared with the estimated value.

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 1.9 “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=hhhhhh[1]
Here you can specify a flash device starting address. Since the different chip selects of the CPU are normally all mapped to different address ranges by the configuration file (See chapter 1.8 "Configuration file JTAGARM7.CFG"), with this option you can switch between the different chip selects.

Default:	/DEVICE-BASE=0
Abbreviation:	/DB=

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

Default:	/OFFSET=0
Abbreviation:	/O=

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

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

Default:	/FILE-OFFSET=0
Abbreviation:	/FO=

[1]hhhhhh=number base is hex

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

Default:	/LENGTH=4000000 (64 MByte)
Abbreviation:	/L=

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

/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.

/NORAM
To speed up the programming performance the flash programming algorithm is loaded into target RAM, when the keyword RAM_BASE is specified in the configuration file. If there are problems during programming, you should avoid using the RAM by adding this option.

/FAST /FAST=
If you specify the option /FAST the host sends the data to be programmed to the target without polling the status of the communication interface. This increases the programming performance dramatically. But, when the target is slow or the host is very fast, this can result in a data lost and the program aborts with the message: "**Target out of Sync. Try again".** You can overcome this problem by using the option /FAST= which does add some extra delay before sending the next programming data to the target**.**

Examples:

```
JTAGARM7  /P  ROMDOS.ROM  /L=20000
```
This example programs up to 128 Kbytes of the file ROMDOS.ROM (with i.e. 512 Kbytes) to the bottom of the flash memory.

2.2. Read a Flash Device to file

Usage: JTAGARM7 /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 1.9 “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[2]
See function /P (Chapter 2.1)

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

Default:	/OFFSET=0
Abbreviation:	/O=

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

Example: JTAGARM7 /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: JTAGARM7 /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 1.9 “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=hhhhhh
See function /P (Chapter 2.1)

/NODUMP
See function /P (Chapter 2.1)

Example: `JTAGARM7 /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: JTAGARM7 /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=hhhhhh
The memory dump starts at an offset of hhhhhh plus the device start address (see option /DEVICE-BASE=).
Default: /OFFSET=0
Abbreviation: /O=

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

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

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

Example: JTAGARM7 /DUMP /CFG=NS7520B.CFG /BIG

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

[3]hhhhhh=number base is hex

2.5. List of supported Flash Devices

Usage: JTAGARM7 /LIST [optionlist]

This command lists all supported flash devices to screen if not redirected to file or device. Flash devices signed with ´*´ are not yet tested.

3. Implementation Information

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

- Actually only the revision 3a (Identifier: 3F0F0F0F) of the ARM7TDMI EmbeddedICE is supported. The revision 4 (Identifier: 40700F0F) is not tested and may not work.

- The flash programming algorithm is loaded into target memory. Actually this algorithm has a size of less than 512 bytes. This size may increase in further implementations to up to 1kByte.

- High speed flash programming is actually not available for Intel. Use the option /NORAM if you want to program this flash types.

- The function /V (verify flash) and /R (read flash) are much slower than the function /P. This is the result of the extremely optimized programming function, but the programming is the most important function.

- Thumb mode is actually not supported. If the target runs in thump mode, the target can not be stopped. An error message is printed.

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

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

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

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

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

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

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

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

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

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

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

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

5.1. Installation on a clean system

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

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

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

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

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

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

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

- 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