MOTOROLA

Order this document by
has not been assigned
Rev. 1.6, 03/2003

DSP56F800 Flash programming via JTAG/OnCE using the Parallel Command Converter

Description of DOS/Windows-based command-line application

version Epsilon 0.2

Draft for review

Digital DNA™ from Motorola

About this Document

This reference manual describes a windows-based command line application for programming of on-chip flash memories of the DSP56F800 family of DSPs using the Parallel Command Converter.

Audience

This document is intended for application developers who develop software for the Motorola DSP56F800 V1 family of DSPs.

Organization

This manual is organized into four chapters and three appendixes.

- **Chapter 1, "Introduction,"** provides an overview of the application.
- **Chapter 2, "Application description,"** describes, in detail the operation of the application.
- **Chapter 3, "Operation under Windows NT and Windows 2000 environments,"** describes additional requirements for running the application under Windows NT and Windows 2000 systems.
- **Chapter 4, "License,"** provides the license required to use this product.
- **Appendix A, "Generating the S-record file using the CodeWarrior development tool,"** contains details about generating S-Record files using the CodeWarrior IDE.
- **Appendix B, "Description of flash configuration file,"** Describes contents of the flash configuration file.
- **Appendix C, "Parallel Command Converter,"** Provides details about the Parallel Command Converter.

Conventions

This document uses the following notational conventions:

Typeface, Symbol or Term	Meaning	Examples
`Courier Monospaced Type`	Code examples	`//Process command for line flash`
Italic	Directory names, project names, calls, functions, statements, procedures, routines, arguments, file names, applications, variables, directives, code snippets in text	...and contains these core directories: *applications* contains applications software... ...CodeWarrior project, *3des.mcp* is... ...the *pConfig* argument.... ...defined in the C header file, *aec.h*....
Bold	Reference sources, paths, emphasis	...refer to the **Targeting DSP56F80x Platform** manual.... ...see: **C:\Program Files\Motorola\Embedded SDK\help\tutorials**
Blue Text	Linkable on-line	...refer to **Chapter 4**, License....
Number	Any number is considered a positive value, unless preceded by a minus symbol to signify a negative value	3V -10 DES^{-1}
ALL CAPITAL LETTERS	# defines/ defined constants	# define INCLUDE_STACK_CHECK
Brackets [...]	Function keys	...by pressing function key [F7]
Quotation marks, “...”	Returned messages	...the message, “Test Passed” is displayed.... ...if unsuccessful for any reason, it will return “NULL”...

Definitions, Acronyms, and Abbreviations

The following list defines the acronyms and abbreviations used in this document. As this template develops, this list will be generated from the document. As we develop more group resources, these acronyms will be easily defined from a common acronym dictionary. Please note that while the acronyms are in solid caps, terms in the definition should be initial capped ONLY IF they are trademarked names or proper nouns.

DSP	Digital Signal Processor or Digital Signal Processing
FFT	Fast Fourier Transforms
FIR	Finite Impulse Response
I/O	Input/Output
IDE	Integrated Development Environment
IIR	Infinite Impulse Response
LSB	Least Significant Bit
MAC	Multiply/Accumulate
MIPS	Million Instructions Per Second
MSB	Most Significant Bit
OnCE™	On-Chip Emulation
OMR	Operating Mode Register
PC	Personal Computer
SDK	Software Development Kit
SP	Stack Pointer
SPI	Serial Peripheral Interface
SR	Status Register
SRC	Source

References

The following sources were referenced to produce this manual:

1. *DSP56800 Family Manual*, DSP56800FM/D
2. *Programming On-Chip Flash Memories of DSP56F80x DSPs Using the JTAG/OnCE Interface*, AN1935
3. *DSP56F80x User's Manual*, DSP56F801-7UM/D

Chapter 1
Introduction

This application provides a means of programming the on-chip Flash memories of the DSP family. The flash memory blocks are programmed “in circuit”, using the on-chip JTAG/OnCE interface. The application is suitable for both a low-volume production environment as well as for in-field firmware upgrades.

The application runs primarily under a 32-bit Windows environment on PC-compatible computers, however 16-bit & 32-bit DOS versions of the application are also available in the distribution package for user's convenience (please see **Chapter 2.3** for details). A Parallel Command Converter is needed to provide the interface between the JTAG/OnCE interface of the target system and the parallel/printer port of the computer. The connection is outlined in **Figure 1-1**. Details on the Parallel Command Converter are provided in **Appendix C**.

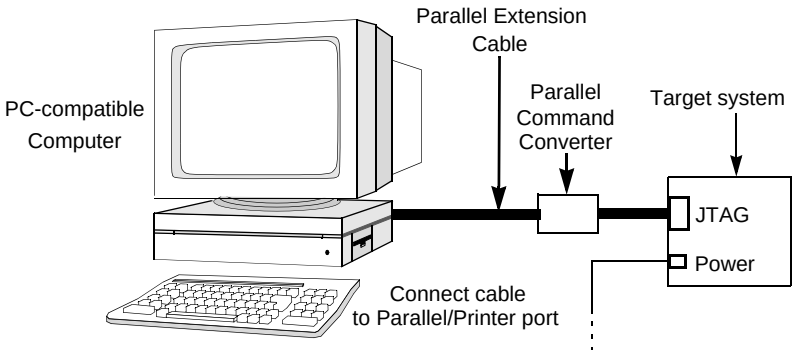

Figure 1-1. Connecting the Target System to the PC

The application is command-line oriented to allow easy integration into scripts. It accepts two data files on its input. The first file contains parameters of the Flash memory on the target DSP; the second file contains data which will be programmed into the Flash memory.

The first file, the *flash config file*, has a specific format compatible with the CodeWarrior development tool; its internal structure is described in **Appendix B**.

The second file, the *S-record file*, contains the code and data of the target application, which will be written into the Flash memory blocks defined by the *flash config file*. The procedure for generating the *S-record file* using the CodeWarrior development tool is provided in **Appendix A**.

1.1 Software Development

This is the recommended procedure for target application development:

1. Development of algorithms and benchmarking using the DSP56F8xxEVM platform
2. Development and debugging of the final application on the user-specific target platform
3. Generation of an S-record file containing code and constant data of the target application; this step is outlined in **Appendix A** for users of the CodeWarrior development environment
4. Low-volume test production using the application described in this document
5. High-volume production using high-speed programming methods

Chapter 2
Application description

The application runs under 32-bit Windows operating systems. However a special driver is needed under Windows NT and Windows 2000 environments; see **Chapter 3** for details.

If you have installed the software from the SDK, the application software can be found in the following file:

..\src\x86\win32\applications\flash_jtag_loader

If you have installed the software from a zip file, it will be located in the directory or folder into which you extracted it.

2.1 Parameters

The application is written as a command line tool. The command line interface is very effective when the application needs to be integrated into scripts or executed by other programs which control the manufacturing process.

All the mandatory and optional parameters required for operation of the application are provided on the command line. The syntax has the two following variants:

```
flash_over_jtag <flash config file> <S-record file> [<opts>]
flash_over_jtag <flash config file> [<opts>]
```

Length of lines in the S-record file should be limited to 300 characters (including control characters like CR and LF) - see **Appendix A**.

In the second variant of syntax no S-record file is supplied to the application. In such cases the application performs a performance test which can be used to determine the speed of operation.

2.2 Options

Position of options on the command line is not fixed, they can be specified before, between or after the config and S-record files.

2.2.1 -p<PORT>

Parameter -pPORT is optional and is used to define the base address of the parallel/printer port to which the Parallel Command Converter is attached. The default value is 0x378 (LPT1). The addresses are dependant on particular computer brand and model. Please consult the manufacturer's documentation for details. Typical values are shown in **Table 2-1**.

Table 2-1. Typical addresses assigned to LPT ports on IBM-PC machines

LPT #	Base address	Address range
LPT1	0x378I	0x378 - 0x37F
LPT2	0x278	0x278 - 0x27F
LPT3	0x3BC	0x3BC - 0x3C3

Example: `flash_over_jtag flash801.cfg my_appl.S -p0x278`

2.2.2 -page

The optional parameter -page specifies that page erase mode should be used instead of mass erase mode. This parameter is useful for in-field upgrades where parameters stored in on-chip flash must be preserved. In this case, only (and all) pages containing addresses referenced by the S-record file are erased.

Example: `flash_over_jtag flash801.cfg my_appl.S -page`

2.2.3 -l<log file>

The optional parameter -l<log file> forces creation of a log file. All messages shown on the screen will also be written into the log file. If the file already exists, it will be overwritten.

Example: `flash_over_jtag flash801.cfg my_appl.S -lmylog.txt`

2.2.4 -t<S-record file>

The optional parameter -t<S-record file> specifies, that additional S-record file should be processed. Contents of this file will be merged with the primary S-record file. The additional S-record file is always processed after the primary S-record file. Therefore should contents of the additional S-record file conflict with the primary S-record file (both files contain data for the same address), then the additional S-record file has precedence. This option is useful for creating timestamps and serial numbers in flash memory of the target device; the primary S-record file contains the application itself and the additional S-record file contains the serial number and date code.

Example: `flash_over_jtag flash801.cfg my_appl.S -tserial.S`

2.2.5 -ml,D

The application supports programming of a target DSP in situations where the DSP is daisy-chained with other devices. The application measures the length of the JTAG chain automatically, but needs information about the position of the DSP in the chain. There are two types of information transferred across the JTAG chain: JTAG instructions and data. The chain length is different for instructions and data and it is therefore necessary to specify the position of the target DSP in both instruction and data chains.

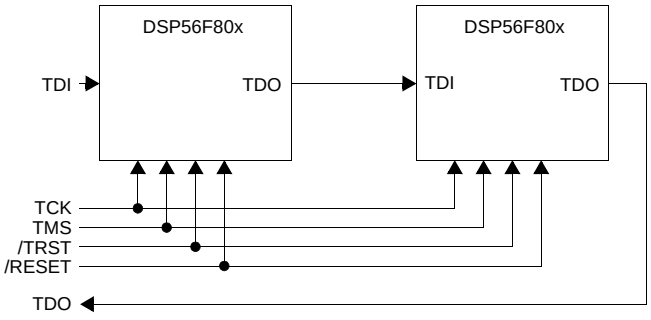

Figure 2-1. Two DSP56F80x DSPs in JTAG daisy-chain

After initialisation all bits of the instruction chain are filled with a binary value of ‘1’. This is equal to the JTAG instruction “BYPASS”. The position of the target DSP in the chain is numbered from 0 onwards (0 means that the DSP is positioned at the beginning of the chain).

Figure 2-1 shows two DSP units connected in daisy-chain. Every DSP56F80x has an instruction chain length of 4 bits and bypass data chain length of 1 bit. Therefore the address of the first unit is -m0,0 and the address of the second unit is -m4,1.

Where the DSP is connected in daisy-chain with another JTAG device, please consult the documentation of the other devices to determine their instruction and bypass data lengths.

Note: Please note that both instruction and bypass data chain lengths are limited to 256 bits. Long JTAG chains result in slow operation of the application and unacceptably long programming times.

Example: `flash_over_jtag flash801.cfg my_appl.S -m4,1`

2.2.6 -info

Optional parameters -info forces the application to access the Flash Information Blocks instead of Flash Main Blocks. The Flash Information Blocks can be used for storage of serial numbers or date codes.

Note that when Information Blocks access is used together with Mass Erase mode, both Flash Information Blocks and Flash Main Blocks are erased. When only Flash Information Blocks should be erased, Page Erase mode should be used.

Example: `flash_over_jtag flash801.cfg my_appl.S -info`

2.2.7 -r<mem><start>:<end>

When the optional parameter -r<mem><start>:<end> is used, contents of DSP memory is dumped to the S-record file specified. When the ouput file exists, it is overwritten. “<mem>” specifies whether Data (“x”) or Program (“p”) memory should be dumped. “<start>” and “<end>” addresses are specified in hexadecimal.

Examples:
```
flash_over_jtag flash801.cfg readout_x.S -rx0x0:0x1000
flash_over_jtag flash801.cfg readout_p.S -rp0x0:0x1000
```

2.2.8 -v<mem><start>:<end>

When the optional parameter -v<mem><start>:<end> is used, contents of DSP memory is dumped to the screen. “<mem>” specifies whether Data (“x”) or Program (“p”) memory should be dumped. “<start>” and “<end>” addresses are specified in hexadecimal.

Examples:
```
flash_over_jtag flash801.cfg -vx0x0:0x1000
flash_over_jtag flash801.cfg -vp0x0:0x1000
```

2.2.9 -w

This option will force the tool to wait until the DSP comes out of Reset state. This is useful in situations where there is external circuitry on user‘s proprietary target hardware which prolongs duration of Reset. This option is also useful when the DSP is powered down and it is neccessary to make sure that it will be programmed as soon as the target hardware is powered up.

Example:
```
flash_over_jtag flash801.cfg my_appl.S -w
```

2.2.10 -s

Normally the tool reports an error for each word of data or code found in the specified S-record file which does not fit into any of the flash blocks defined in the flash config file. In "silent" mode enabled by this option the tool will only report that some data were ignored, but will not print any further details.

Example:
```
flash_over_jtag flash801.cfg my_appl.S -s
```

2.2.11 -d

Normally the tool resets the target after programming is concluded. This option forces the tool to leave the target in debug mode - i.e. the downloaded code is not executed immediately after programming.

Example:
```
flash_over_jtag flash801.cfg my_appl.S -d
```

2.3 Limitations of the 16-bit DOS version

The 16-bit (Real Mode) DOS version of the application is provided for user's convenience. The 16-bit DOS mechanism for memory allocation introduces certain limitations however. The maximum size of allocated memory block is 65520 bytes, what equals to 32760 words. Therefore 32760 words is the maximum size of any given flash block specified in the flash config file (see **Appendix B** for details). This limitation applies for example to program flash block #1 of the DSP56F807 (size 32764 words):

```
0 0x0004 0x7fff 1 0x1340 0x0002 0x0006 0x001A 0x0033 0x0066 0x001A 0x019A 0x0006
```

The workaround is to split the flash block into two blocks of smaller size:

```
0 0x0004 0x7000 1 0x1340 0x0002 0x0006 0x001A 0x0033 0x0066 0x001A 0x019A 0x0006
0 0x7001 0x7fff 1 0x1340 0x0002 0x0006 0x001A 0x0033 0x0066 0x001A 0x019A 0x0006
```

or to limit size of the flash block to 32760 words, what will make the last 4 words in the block unusable.

This limitation does not apply to Win32 and 32-bit DOS versions.

2.4 Operation

The application outputs progress and status messages to standard output. Where the application is called from a script or another program, the output messages can be redirected to a file for later inspection by the operator. For easy integration into scripts, the final status of the programming operation is also provided through error level (see **Section 2.6** for details). When the amount of code and data in the S-record file is large, programming and verification may take longer. For user convenience, a progress indication is provided during programming and verification (one character per 512 words). Typical error-free output of the application can be seen in **Listing 2-1**.

Listing 2-1. Sample of application messages

```
DSP56F800 Flash loader. Compiled on Mar 10 2002, 14:26:15.
version Delta 0.1
(c) Motorola 2001 - 2002, MCSL
JTAG IR path length: 4
JTAG DR path length: 1 (BYPASS)
IDCode status: 0x9
Jtag ID: 0x11f2501d
Debug Request status: 0xd
Enable OnCE status: 0xd, polls left: 10
Enable OnCE successful, target chip is in Debug mode
4 flash blocks defined in the config file.
S-record ID: PROGRAM&DATA
Initialising FIU at address: 0xf40
FIU (0xf40) initialisation done.
Flash (0xf40) mass erase done.
Flash (0xf40) programming done. 0xdb words written.
Initialising FIU at address: 0xf80
FIU (0xf80) initialisation done.
Flash (0xf80) mass erase done.
Flash (0xf80) programming done. 0x4 words written.
Initialising FIU at address: 0xf80
FIU (0xf80) initialisation done.
Mass erase skipped.
Flash (0xf80) programming done. 0x4 words written.
Initialising FIU at address: 0xf60
FIU (0xf60) initialisation done.
Flash (0xf60) mass erase done.
Flash (0xf60) programming done. 0x2 words written.
The target was reset, the application is running
```

2.5 Error Messages

The application will generate error messages in various situations. The error messages are:

- S-record file corruption:

```
Hex2dec conversion error: %character%
S-record file line does not start with "S"
S-record checksum error
Data @ 0x#address# ignored
```

- Operating system related errors:

```
Memory allocation error for flash block #number#
Cannot open file <filename>
```

- Parallel Command Convertor related:

```
Command Converter not connected or disabled!
```

- Target chip related:

```
Command Converter not connected or disabled!
Target chip refused to enter Debug mode!
FIU initialisation failed, BUSY bit is set.
Flash mass erase failed, BUSY bit is set.
Verification error at addr: #addr#, wr: #data#, rd: #data#
```

2.6 Observing result through error level

Error level exit status is provided for easier integration of the application into scripts and manufacturing process control systems. This status gives the calling script information about the programming result. Possible error level values with their descriptions are provided in **Table 2-2**.

Table 2-2. Error level exit codes

Error level	Description
0	Programming finished successful
1	Flash configuration file not found
2	S-record file not found or corrupted
3	Command converter not found
4	Failed to bring the target DSP into Debug Mode
5	Flash verification error
6	Incorrect parameters
7	Performed speed test
8	System error

2.7 Performance

The performance of the application is limited by the speed of the parallel/printer port. In most IBM-PC compatible computers, the parallel/printer port is implemented as an ISA-bus device and its throughput is therefore limited.

Speed of operation is also dependant on the parallel/printer port mode. The fastest results are achieved in ECP mode.

Where faster operation is required, higher throughput may be achieved by using PCI parallel/printer port card.

A typical programming time of a whole program flash on DSP56F803 or DSP56F805 (32250 words) with an ISA-bus-based parallel/printer port in ECP mode is 45 seconds. In the case that only a portion of the flash memory space is occupied by the target application, the programming time will be proportionally shorter.

Chapter 3
Operation under Windows NT and Windows 2000 environments

Windows NT and Windows 2000 operating systems prevent applications from accessing the hardware of the host computer directly. However this application needs direct access to the parallel/printer port for performance reasons. A special driver which enables the application to access the parallel/printer port is therefore needed under these operating systems. Please note that installation of drivers requires administrator privileges.

3.1 Driver description

This application uses the ZLPORT driver created by Alexander Zloba (zal@specosoft.com) to gain access to I/O ports. File “zlportio.sys” must be present in the same directory as the exe file when the Win32 based version of the application is executed for the first time under Windows NT or Windows 2000 operating systems. The driver can be removed from the system at any time by running the “driver_remove” utility.

Chapter 4
License

4.1 Limited Use License Agreement

LIMITED USE LICENSE AGREEMENT

PLEASE READ THIS AGREEMENT CAREFULLY BEFORE USING THIS SOFTWARE. BY USING OR COPYING THE SOFTWARE, YOU AGREE TO THE TERMS OF THIS AGREEMENT.

The software in either source code form ("Source") or object code form ("Object") (cumulatively hereinafter "Software") is provided under a license agreement ("Agreement") as described herein. Any use of the Software including copying, modifying, or installing the Software so that it is usable by or accessible by a central processing unit constitutes acceptance of the terms of the Agreement by the person or persons making such use or, if employed, the employer thereof ("Licensee") and if employed, the person(s) making such use hereby warrants that they have the authority of their employer to enter this license agreement,. If Licensee does not agree with and accept the terms of this Agreement, Licensee must return or destroy any media containing the Software or materials related thereto, and destroy all copies of the Software.

The Software is licensed to Licensee by Motorola Incorporated ("Motorola") for use under the terms of this Agreement. Motorola retains ownership of the Software. Motorola grants only the rights specifically granted in this Agreement and grants no other rights. Title to the Software, all copies thereof and all rights therein, including all rights in any intellectual property including patents, copyrights, and trade secrets applicable thereto, shall remain vested in Motorola.

For the Source, Motorola grants Licensee a personal, non-exclusive, non-assignable, revocable, royalty-free right to use, copy, and make derivatives of the Source solely in a development system environment in order to produce object code solely for operating on a Motorola semiconductor device having a central processing unit ("Derivative Object").

For the Object and Derivative Object, Motorola grants Licensee a personal, non-exclusive, non-assignable, revocable, royalty-free right to copy, use, and distribute the Object and the Derivative Object solely for operating on a Motorola semiconductor device having a central processing unit.

Licensee agrees to: (a) not use, modify, or copy the Software except as expressly provided herein, (b) not distribute, disclose, transfer, sell, assign, rent, lease, or otherwise make available the Software, any derivatives thereof, or this license to a third party except as expressly provided herein, (c) not remove obliterate, or otherwise defeat any copyright, trademark, patent or proprietary notices, related to the Software (d) not in any form export, re-export, resell, ship or divert or cause to be exported, re-exported, resold, shipped, or diverted, directly or indirectly, the Software or a direct product thereof to any country which the United States government or any agency thereof at the time of export or re-export requires an export license or other government approval without first obtaining such license or approval.

THE SOFTWARE IS PROVIDED ON AN "AS IS" BASIS AND WITHOUT WARRANTY OF ANY KIND INCLUDING (WITHOUT LIMITATION) ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL MOTOROLA BE LIABLE FOR ANY LIABILITY OR DAMAGES OF ANY KIND INCLUDING, WITHOUT LIMITATION, DIRECT OR INDIRECT OR INCIDENTAL OR CONSEQUENTIAL OR PUNITIVE DAMAGES OR LOST

PROFITS OR LOSS OF USE ARISING FROM USE OF THE SOFTWARE OR THE PRODUCT REGARDLESS OF THE FORM OF ACTION OR THEORY OF LIABILITY (INCLUDING WITHOUT LIMITATION, ACTION IN CONTRACT, NEGLIGENCE, OR PRODUCT LIABILITY) EVEN IF MOTOROLA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. THIS DISCLAIMER OF WARRANTY EXTENDS TO LICENSEE OR USERS OF PRODUCTS AND IS IN LIEU OF ALL WARRANTIES WHETHER EXPRESS, IMPLIED, OR STATUTORY, INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR PARTICULAR PURPOSE.

Motorola does not represent or warrant that the Software is free of infringement of any third party patents, copyrights, trade secrets, or other intellectual property rights or that Motorola has the right to grant the licenses contained herein. Motorola does not represent or warrant that the Software is free of defect, or that it meets any particular requirements or need of the Licensee, or that it conforms to any documentation, or that it meets any standards.

Motorola shall not be responsible to maintain the Software, provide upgrades to the Software, or provide any field service of the Software. Motorola reserves the right to make changes to the Software without further notice to Licensee.

The Software is not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Software could create a situation where personal injury or death may occur. Should Licensee purchase or use the Software for any such unintended or unauthorized application, Licensee shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the Software.

The term of this Agreement is for as long as Licensee uses the Software for its intended purpose and is not in default of any provisions of this Agreement. Motorola may terminate this Agreement if Licensee is in default of any of the terms and conditions of this Agreement.

This Agreement shall be governed by and construed in accordance with the laws of the State of Arizona and can only be modified in a writing signed by both parties. Licensee agrees to jurisdiction and venue in the State of Arizona.

By using, modifying, installing, compiling, or copying the Software, Licensee acknowledges that this Agreement has been read and understood and agrees to be bound by its terms and conditions. Licensee agrees that this Agreement is the complete and exclusive statement of the agreement between Licensee and Motorola and supersedes any earlier proposal or prior arrangement, whether oral or written, and any other communications relative to the subject matter of this Agreement.

Appendix A
Generating the S-record file using the CodeWarrior development tool

This appendix describes how to set up the CodeWarrior development tool to generate an S-record file accepted by the application.

The S-record file is generated based on linker set up. The correct linker set up is shown in **Figure A-1**.

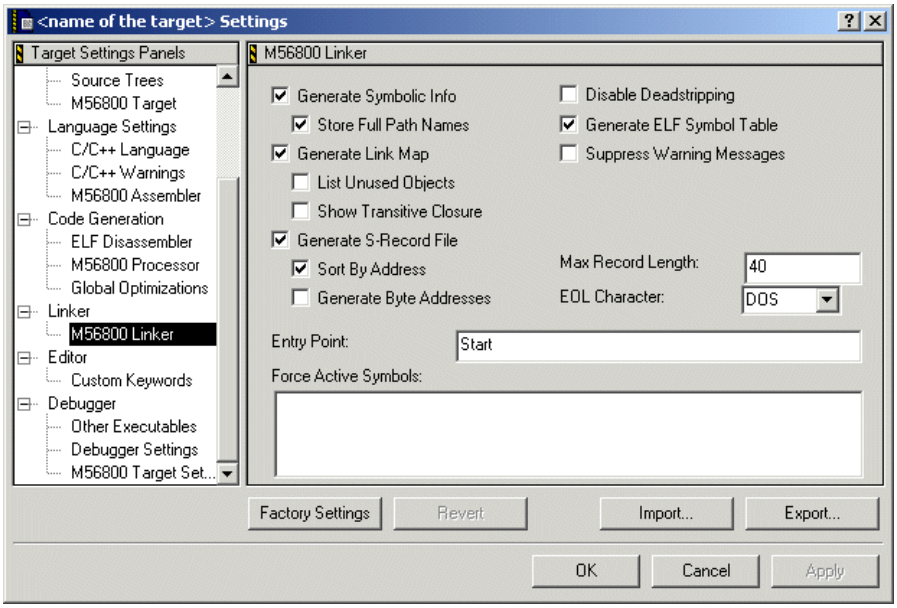

Figure A-1. Linker configuration

The linker will generate three different S-record files:

- **output_file.p.S** - contains data and code to be stored in program memory locations
- **output_file.x.S** - contains data and code to be stored in data memory locations
- **output_file.S** - combination of the two previous files

In the combined S-record file, data memory locations are distinguished from program memory locations by having addresses greater than 65535. The combined file is the accepted format for the application.

Appendix B
Description of flash configuration file

The flash configuration file serves as a description of the internal memory structure of the DSP. It describes the memory map of the flash memories, base addresses of their programming interfaces and timing information required for correct programming.

During programming, the chip operates at the frequency of the crystal attached to it and it is the responsibility of the user to provide the correct values for the flash timing registers at that particular frequency.

The format of the flash configuration file is compatible with the format required by the CodeWarrior development tool.

Each line in the flash configuration file describes one block of flash memory in the memory map. The only exception are lines starting with a "#" character, which are considered to be comments and are ignored. An example of a flash configuration file is shown in **Listing B-1**.

Listing B-1. Sample Flash Configuration File

```
#base  start   end     P  reg    terase  tme     tnvs    tpgs    tprog   tnvh    tnvh1   trcv    fdiv
0      0x0004  0x7dff  1  0x0f40 0x0002  0x0006  0x001A  0x0033  0x0066  0x001A  0x019A  0x0006  0x000f
0      0x0000  0x0003  1  0x0f80 0x0002  0x0006  0x001A  0x0033  0x0066  0x001A  0x019A  0x0006  0x000f
0      0x8000  0x87ff  1  0x0f80 0x0002  0x0006  0x001A  0x0033  0x0066  0x001A  0x019A  0x0006  0x000f
0      0x1000  0x1fff  0  0x0f60 0x0002  0x0006  0x001A  0x0033  0x0066  0x001A  0x019A  0x0006  0x000f
```

base	This column is ignored
start	Address of the beginning of the flash block in the memory map
end	Address of the end of the flash block in the memory map
P	1 = program memory, 0 = data memory
reg	Base address of the Flash programming interface
terase	Value to be stored in the *Terasel* register during programming
tme	Value to be stored in the *Tmel* register during programming
tnvs	Value to be stored in the *Tnvsl* register during programming
tpgs	Value to be stored in the *Tpgsl* register during programming
tprog	Value to be stored in the *Tprogl* register during programming
tnvh	Value to be stored in the *Tnvhl* register during programming
tnvh1	Value to be stored in the *Tnvh1l* register during programming
trcv	Value to be stored in the *Trcvl* register during programming
fdiv	Value to be stored in the *Fiu_Divisor* register during programming

The last column (Fiu_Divisor) can be omitted. In this case the register value defaults to 15 (0x000f). When the S-record file specifies data to be written to a flash location outside any of the blocks, the error message, *Data @ 0x#address# ignored*, is generated.

Appendix C
Parallel Command Converter

The Parallel Command Converter acts as simple voltage-level translator interface. It accepts +5V TTL levels on the parallel/printer port side and +3.3V CMOS levels on the JTAG/OnCE interface side. The only exception is the $\overline{\text{RESET}}$ signal, which features a higher current pull-down transistor with 4.7 kΩ pull-up resistor.

Only four JTAG signals (TCK, TDI, TDO and TMS) plus a ground connection are required for correct operation of the application. This is useful where space or number of pins of the JTAG header is limited on the target platform.

The Parallel Command Converter is provided on board the EVM for the convenience of the user. Therefore external command converter is not needed for programming EVMs.

The Parallel Command Converter can be purchased from Motorola (order code “DSPCOMMPARALLEL”) or a third party. Alternatively, the user can build the Parallel Command Converter on his own.

The Parallel Command Converter schematic is provided on the next page.

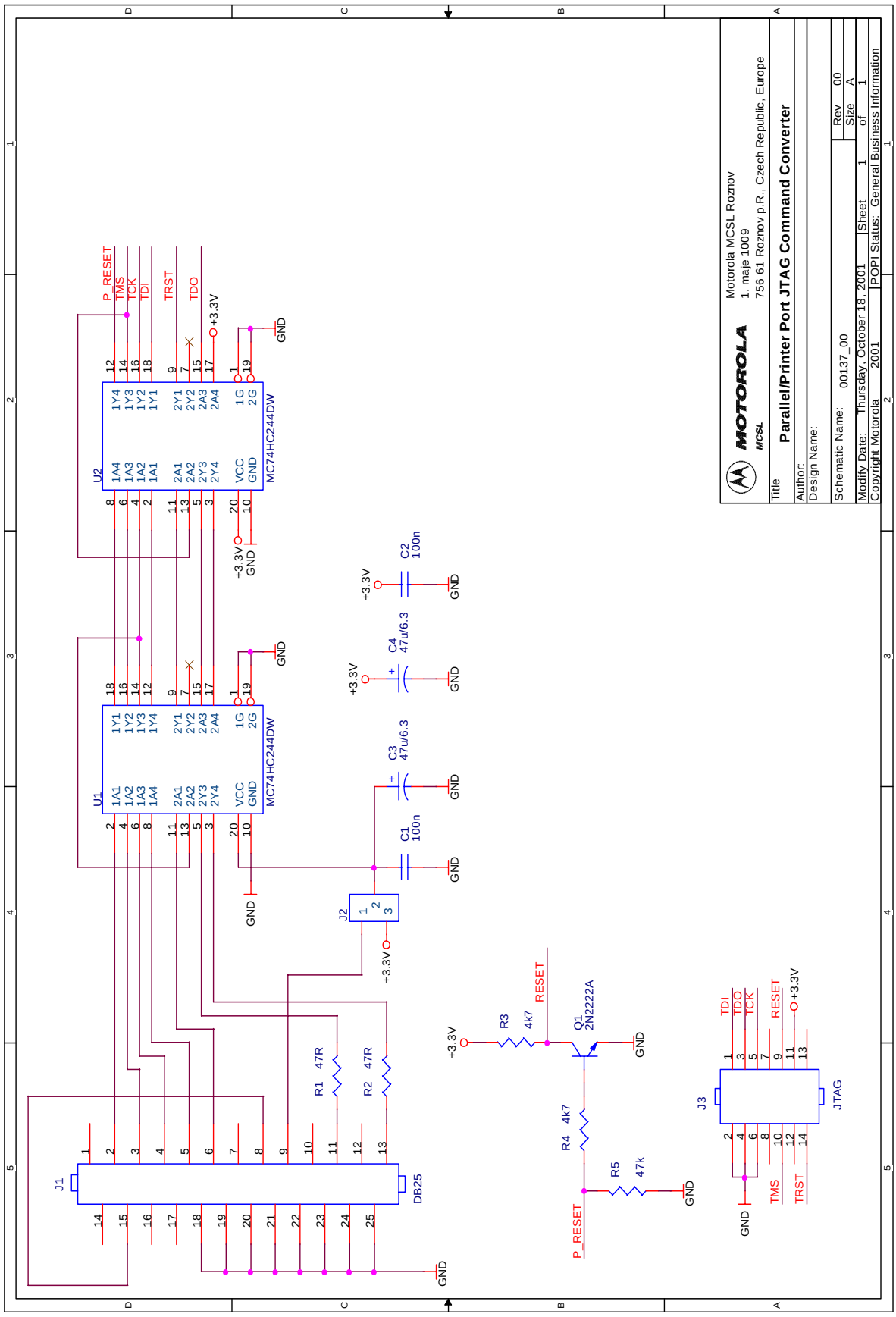

Figure C-1. Parallel Command Converter schematic

OnCE™ is a registered trademark of Motorola, Inc.

How to reach us:
USA/EUROPE/Locations Not Listed: Motorola Literature Distribution; P.O. Box 5405, Denver, Colorado 80217. 1–303–675–2140 or 1–800–441–2447

JAPAN: Motorola Japan Ltd.; SPS, Technical Information Center, 3–20–1, Minami–Azabu. Minato–ku, Tokyo 106–8573 Japan. 81–3–3440–3569

ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Centre, 2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong. 852–26668334

Technical Information Center: 1–800–521–6274

HOME PAGE: http://www.motorola.com/semiconductors/

MOTOROLA