nowECC Generation Tool
Version 2.22

User's Guide

Literature Number: SPNU491D
April 2014

Contents

List of Tables

nowECC Generation Tool

TI Microcontroller devices developed for safety-critical applications contain a module embedded in the flash wrapper for correction of one bit error and detection of two bit errors called the SECDED (Single Error Correction Double Error Detection). This is done using the ECC algorithm. This ECC data needed to do the correction and detection is programmed along with executable data.

The nowECC tool generates the ECC data corresponding to the executable program data and as per the required memory mapping.

This document describes the nowECC tool user commands and options.

1 Introduction

The TI microcontroller family contains as part of the embedded Flash module a circuit that provides, the capability to detect and correct memory faults. This module known as Single Error Correction and Double Error Detection circuit (SECDED) needs 8 Error correction check bits to be programmed along with every 64 bit of data programmed in the memory.

The purpose of the nowECC tool is to generate these ECC check bits needed for the embedded SECDED to recognize and correct the memory faults.

The tool described in this document is a command line tool which executes in 32-bit mode on a Microsoft Windows PC.

The detailed description of the functionality of the embedded Flash module and the SECDED can be found in respective Flash Module specifications.

2 nowECC - ECC Generation Tool

The nowECC tool generates the data to be programmed into the ECC memory locations of a TMS470/TMS570 Platform device with ECC Flash memory. This section describes the user commands and options applicable for the nowECC tool.

2.1 Command Line Format

The command line interface of the nowECC tool is defined as follows:

```
nowecc -i <input> <memory_map> [OPTIONS] [-o <output>]
```

Input file: The input file is the mandatory input that is required for the nowECC tool to operate on.

The input file should be in one of the following formats:

- Extended Tektronix
- Intel Hex
- ASCII Hex
- Motorola S
- COFF
- ELF

Memory map: The user has to specify the memory map of the TI microcontroller for which the ECC needs to be generated.

The supported memory maps are:

- F05 Legacy
- LF035 Legacy
- F035
- F021 8MB
- F021 16 MB
- F021 16 MB FPGA
- F28M35x_C28
- F28M35x_M3

For more information refer to sections Section 2.2.11 to Section 2.2.18.

Options: The following lists the user options that could be provided to the nowECC tool.

- **Output file:** The tool generates the output file with the name provided by the user. If the output file is not specified, then the output file is generated as "<input>_ECC". The output file is of the same format as the input file.
- **Return value:** Return value '0' indicates that no error has occurred. In case of an error, a non-zero return value and corresponding error message is displayed. For the list of error codes, see Appendix A.

2.2 Command Line Options

The command line options are not case-sensitive and can be specified in any order.

The parameters shown in Table 1 are used in the command line descriptions.

Table 1. Command Line Options

Command Line	Description
<ip_file>	Specifies Input file name
<memory_map>	Specifies the required memory map.
[op_file]	Specifies the output file name
[value]	Specifies a numeric value. All numbers are assumed to be Decimal unless prefixed by "0x" or "0X", then they are hexadecimal.
[address]	Specifies an address. All numbers are assumed to be decimal unless prefixed by "0x" or "0X", then they are hexadecimal.
[address_begin]	Specifies the start address of a section.
[address_end]	Specifies the end address of a section.

2.2.1 '-i' Command - To Specify Input file

The '-i' command can be used to specify the input file. By default, any argument without a switch is considered an input file.

```
nowecc -i  <ip_file> <memory_map>
```

or

```
nowecc <ip_file> <memory_map>
```

2.2.2 '-o' Command - Specify Output File

With the - o command option, an output file can be specified.

```
nowecc -i  <ip_file> <memory_map> -o [op_file]
```

2.2.3 '-a' Command - Append ECC

With the append option, the ECC data is appended to the end of the input file. A new section that contains the ECC data is created. If no output file name is specified, the output file name would be <ip_file>_ECC.

```
nowecc <ip_file> <memory_map> -a -o [op_file]
```

2.2.4 '-f' Command - Fill Gaps

The fill option specifies the byte value to be used for filling holes when the input file sections do not begin or end on a 64-bit boundary. The default fill value is 0xFF. The fill value has to be specified in hexadecimal format.

```
nowecc -i  <ip_file> <memory_map> -f [value] -o [op_file]
```

2.2.5 '-r' Command - Range

If one or more range options are given, the ECC data is calculated for only the given data ranges. The default condition is to use all of the initialized data in the input file for calculating the ECC data.

```
nowecc -i <ip_file> <memory_map> -r [address_begin] [address-end] -o [op_file]
```

2.2.6 '-x' Command - Exclude

The exclude option specifies one or more ranges of the input data to be ignored during calculation of the ECC data.

```
nowecc -i <ip_file> <memory_map> -x [address_begin] [address_end] -o [op_file]
```

NOTE: The range option and the exclude option cannot be specified at the same time.

2.2.7 '-l' Command - Generate linkable output

This option generates an output that contains relocation information and is linkable (i.e., the F_RELFLG and F_EXEC in the file header are 0).

```
nowecc -l -i  <ip_file> <memory_map> -o [op_file]
```

2.2.8 -r4' Command - ECC in CPU feature of Cortex-R4

The -r4 option calcuates ECC based on the internal ECC logic in the Cortex-R4 CPU.

```
nowecc - i <ip_file> <memory_map> -o [op_file] -r4
```

2.2.9 -q' Command - Quiet Mode

This option suppresses the display messages. The error messages and warnings are still issued.

```
nowecc -q -i <ip_file> <memory_map> -o [op_file]
```

2.2.10 '-h' Command - Display Help

The display help option causes the tool to display a short description of the tool and a help for every single command that can be used together with the tool on the command line.

```
nowecc -h
```

2.2.11 '-f05' Command - Memory Map F05

```
nowecc -f05 -i <ip_file>-o [op_file]
```

-F05 specifies the F05 flash memory mapping.

For the F05 memory map, the data can exist in any of the memory regions shown in Table 2.

Table 2. F05 Memory Regions

Memory Region	Hex Address	ECC Code Memory Mapping
Normal memory space	0x00000000 - 0x003FFFFF	ECC code starts at 0x00400000

NOTE: ECC for OTP memory is not supported for the F05 model.

2.2.12 '-f035' Command - Memory Map F035

```
nowecc -i <ip_file> -f035 -o [op_file]
```

For F035 memory map, 8 bit of ECC is generated for every 64 bit data and 19 address bits.

Data can exist in any of the memory regions shown in Table 3.

Table 3. F035 Memory Regions

Memory Region	Hex Address	ECC Code Memory Mapping
Normal memory space	0x00000000 - 0x003FFFFF	ECC code starts at 0x00400000
Customer OTP space	0x00600000 - 0x00603FFF	ECC code starts at 0x00608000
TI OTP space	0x00604000 - 0x00607FFF	ECC code starts at 0x0060C000
TLB Space	0xFD000000 - 0xFD1FFFFF	ECC code starts at 0xFD200000

2.2.13 -lf035' Command - Memory Map LF035

```
nowecc -i <ip_file> -lf035 -o [op_file]
```

For LF035 memory map, 8 bit of ECC is generated for every 64 bit of data.

Data can exist in any of the memory regions shown in Table 4.

Table 4. LF035 Memory Regions

Memory Region	Hex Address	ECC Code Memory Mapping
Normal memory space	0x00000000 - 0x003FFFFF	ECC code starts at 0x00400000
Customer OTP space	0x00600000 - 0x00603FFF	ECC code starts at 0x00608000
TI OTP space	0x00604000 - 0x00607FFF	ECC code starts at 0x0060C000
TLB Space	0xFD000000 - 0xFD1FFFFF	ECC code starts at 0xFD200000

2.2.14 '-f021 16M' Command - Memory Map F021 - 16MB

For F021 16MB memory map there are two configurations:

1. **-f021 16M_NOADD**: 8 bit of ECC calculated for every 64 bit data (No address bits used).

   ```
   nowecc -i <ip_file> -f021 16M_NOADD -o [op_file]
   ```

   Example:

   ```
   nowecc -i file.out -f021 16M_NOADD -o file_ecc.out
   ```

2. **-f021 16M_ADD**: 8 bit of ECC calculated for every 64 bit data and 19 address bits.

   ```
   nowecc -i <ip_file> -f021 16M_ADD -o [op_file]
   ```

   Example:

   ```
   nowecc -i file.out -f021 16M_ADD -o file_ecc.out
   ```

Data can exist in any of the memory regions shown in Table 5.

Table 5. F021 - 16MB Memory Regions

Memory Region	Hex Address	ECC Code Memory Mapping
Normal memory space	0x00000000 - 0x00FFFFFF	ECC code starts at 0xF0400000
Customer OTP space	0xF0000000 - 0xF000FFFF	ECC code starts at 0xF0040000
TI OTP space	0xF0080000 - 0xF008FFFF	ECC code starts at 0xF00C0000
EEPROM Space	0xF0200000 - 0xF03FFFFF	ECC code starts at 0xF0100000
TLB Space	0xFD000000 - 0xFD1FFFFF	ECC code starts at 0xFD200000

2.2.15 '-f021 8M' Command - Memory Map F021 - 8MB

```
nowecc -i <ip_file> -f021 8M -o [op_file]
```

For F021 memory map, 8 bit of ECC is calculated for every 64 bit data (no address bits used).

Data can exist in any of the memory regions shown in Table 6.

Table 6. F021 - 8MB Memory Regions

Memory Region	Hex Address	ECC Code Memory Mapping
Normal memory space	0x00000000 - 0x00FFFFFF	ECC code starts at 0xF0300000
Customer OTP space	0xF0000000 - 0xF000FFFF	ECC code starts at 0xF0200000
TI OTP space	0xF0080000 - 0xF008FFFF	ECC code starts at 0xF0280000
TLB Space	0xFD000000 - 0xFD1FFFFF	ECC code starts at 0xFD200000

2.2.16 '-f021 16M FPGA' Command - Memory Map F021 - 16MB FPGA

For F021 16MB FPGA memory map there are two configurations:

1. **-f021 16M_NOADD_FPGA**: 8 bit of ECC calculated for every 64 bit data (No address bits used).

   ```
   nowecc -i <ip_file> -f021 16M_NOADD_FPGA -o [op_file]
   ```

   Example:

   ```
   nowecc -i file.out -f021 16M_NOADD_FPGA -o file_ecc.out
   ```

2. **-f021 16M_ADD_FPGA**: 8 bit of ECC calculated for every 64 bit data and 19 address bits.

   ```
   nowecc -i <ip_file> -f021 16M_ADD_FPGA -o [op_file]
   ```

   Example:

   ```
   nowecc -i file.out -f021 16M_ADD_FPGA -o file_ecc.out
   ```

Data can exist in any of the memory regions shown in Table 7.

Table 7. F021 - 16MB FPGA Memory Regions

Memory Region	Hex Address	ECC Code Memory Mapping
Normal memory space	0x60000000 - 0x603FFFFF	ECC code starts at 0x60400000

2.2.17 '-F021 Concerto_M3' Command - Memory Map F28M35x M3

```
nowecc -i <ip_file> -f021 Concerto_M3 -o [op_file]
```

For F28M35x M3 memory map, 8 bit of ECC is calculated for every 64 bit data.

Data can exist in any of the memory regions shown in Table 8.

Table 8. F28M35x M3 Memory Regions

Memory Region	Hex Address	ECC Code Memory Mapping
Normal memory space	0x00200000 - 0x0027FFFF	ECC code starts at 0x00600000
OTP Bank1 - TI Manufacturing	0x00680000 - 0x006807FF	ECC code starts at 0x00700000
OTP Bank1 - TI Application use	0x00680800 - 0x00680FFF	ECC code starts at 0x00700100

2.2.18 '-F021 Concerto_C28' Command - Memory Map F28M35x C28

```
nowecc -i <ip_file> -f021 Concerto_C28 -o [op_file]
```

For F28M35x C28 memory map, 8 bit of ECC is calculated for every 64 bit data.

Data can exist in any of the memory regions shown in Table 9.

Table 9. F28M35x C28 Memory Regions

Memory Region	Hex Address	ECC Code Memory Mapping
Normal memory space	0x00100000 - 0x0013FFFF	ECC code starts at 0x00200000
OTP Bank2 - TI Manufacturing	0x00240000 - 0x002403FF	ECC code starts at 0x00280000
OTP Bank2 - TI Application use	0x00240400 - 0x002407FF	ECC code starts at 0x00280080

2.2.19 '-e' Command - Exchange ECC

```
nowecc -i <ip_file> -e -o [op_file] [new mapping]
```

This exchange option could be used if the input already has the ECC section (of any unknown mapping) along with the data section to generate ECC for the required mapping.

Example:

```
nowecc -i file.out -e -o ECC_file_8Mf021.out -F021 8M
nowecc -i file.out -e -o ECC_file_16Mf021.out -F021 16M_ADD
```

NOTE: If the input does not have any ECC section this '-e' option would be ignored by the tool and would continue to generate ECC section as usual.

The user should use the -a option if the ECC section has to be appended along with the data section.

Example:

```
nowecc -e -a -i file.out -o file_8Mf021ECC.out -F021 8M
nowecc -e -a -i file.out -o file_16Mf021ECC.out -F021 16M_ADD
```

2.2.20 '-s' Command - Force ECC code

```
nowecc -s [ecc_address+offset] [ecc_value] -i <ip_file> <memory_map> -o [op_file]

nowecc -s [ecc_address] [ecc_value] -i <ip_file> <memory_map> -o [op_file]
```

The calculated ECC at the location <ecc_address+offset> is substituted by the 8-bit <ecc_value>.

The meaning of the parameters is given below:

<ecc_address>	Starting address or a symbolic address for an ECC byte. If this value is not an address in the ECC memory range, the tool displays an error message.
<offset>	The offset value in hex from the <ecc_address>. The offset is an optional parameter. No space is allowed in this option.

<ecc_value>	An 8-bit value that is substituted for the calculated ECC value at that location.

2.2.21 '-s1' Command - Create One-Bit Error

```
nowecc -s1 [address] [bit-number1] -i <ip_file> <memory_map> -o [op_file]
nowecc -s1 [address+offset] [bit-number1] -i <ip_file> <memory_map> -o [op_file]
```

The meaning of the parameters is given below:

<address>	Starting address or a symbolic address of a 64-bit field in the normal or OTP memory space. This value must be aligned to eight bytes (64 bits). If this value is not aligned, the tool displays an error message.
<offset>	The offset value, in hex, from the starting address or the symbolic address. The offset is an optional value. No space is allowed in this option.
<bit-number1>	The number (decimal) of the bit that is assumed to be inverted. Range = 0 ... 90; 0 = LSB, 63 = MSB, 64 - 71 LSB - MSB of ECC. 72 - 90 Address bits [3:21] If number < 64, the numbered bit is inverted in data before calculating ECC. If number ≥ 64 and ≤ 71, the numbered bit in the ECC is inverted before written to output file. In other words, if the bit number is from 0 to 63, the tool calculates the ECC value as if the original data bit had been inverted. If number ≥ 72, the corresponding address bit is inverted before calculating ECC.

2.2.22 '-s2' Command - Create Two-Bit Error

```
nowecc -s2 [address] [bit-number1] [bit-number2] -i <ip_file> <memory_map> -o [op_file]
nowecc -s2 [address+offset] [bit-number1] [bit-number2] -i <ip_file> <memory_map> -o [op_file]
```

The meaning of the parameters is given below:

<address>	Starting address or a symbolic address of a 64-bit field in the normal or OTP memory space. This value must be aligned to eight bytes (64 bits). If this value is not aligned, the tool displays an error message.
<offset>	The offset value, in hex, from the starting address or the symbolic address. The offset is an optional value. No space is allowed in this option.
<bit-number1>	The number (decimal) of the bit that is assumed to be inverted. Range = 0 ... 90; 0 = LSB, 63 = MSB, 64 - 71 LSB - MSB of ECC. 72 - 90 Address bits [3:21] If number < 64, the numbered bit is inverted in data before calculating ECC. If number ≥ 64 and ≤ 71, the numbered bit in the ECC is inverted before written to output file. In other words, if the bit number is from 0 to 63, the tool calculates the ECC value as if the original data bit had been inverted. If number ≥ 72, the corresponding address bit is inverted before calculating ECC.
<bit-number2>	The number (decimal) of the bit that is assumed to be inverted. Range = 0 ... 90; 0 = LSB, 63 = MSB, 64 - 71 LSB - MSB of ECC. 72 - 90 Address bits [3:21] If number < 64, the numbered bit is inverted in data before calculating ECC. If number ≥ 64 and ≤ 71, the numbered bit in the ECC is inverted before written to input file. In other words, if the bit number is from 0 to 63, the tool calculates the ECC value as if the original data bit had been inverted. If number ≥ 72, the corresponding address bit is inverted before calculating ECC.

2.2.23 '-v' Command - Display Version

The -v would display the tool version information.

```
nowecc -v
```

2.2.24 '-n' Command - ECC Section Root Name

```
nnowecc -i <ip_file> <memory_map> -o [op_file] -n root_name
```

The ECC Section Root Name option is used to specify the root name for the ECC sections created by the tool for ELF/COFF file.

The root name should not exceed more than six characters. If this option is used for any file other ELF/COFF file, the tool ignores this option.

2.2.25 '-c' Command - Command File

```
nowecc -c [command file>] [options] -i <ip_file> <memory_map> -o [op_file]
```

The -c option can be used to specify a command file. The command file can contains one or more command line options with parameters. Each line should contain only one option. Comments can be included in the command file. Comments should start with the character '#' and end at the end of the line. The options -r, -x, -s, -s1, and -s2 can be put more than one time on the command line or the command file.

2.2.26 '-fpga' Command - ECC Implementation on Platform Board

```
nowecc -i <ip_file> <memory_map> [options] -o [op_file] -fpga
```

This option can be used to calculate ECC using a different mirroring technique to support the ECC implementation on the platform board. A different mirroring technique is needed since the new FWM implementation in bitstream considers only the last byte of every 32-bit pattern of 64-bit flash data. This option is to be used in conjunction with the appropriate memory map. The different mirroring technique is explained below:

- Original mirroring of ECC:
  0x01230123,
  0x01230123,
  0x45674567,
  0x45674567.
- Mirroring after using the '-fpga' option:
  0x01010101,
  0x23232323,
  0x45454545,
  0x67676767.

2.2.27 '-d' Command - Ignore Data Lying in Invalid Memory Regions

```
nowecc -i <ip_file> <memory_map> [options] -o [op_file] -d
```

This option is used to ignore data lying in invalid memory regions in the input before calculating ECC. The valid memory regions are dependent on the memory map option used.

The data lying only in the valid memory regions is considered for calculating ECC. The tool displays a message indicating the sections ignored for calculating ECC.

2.2.28 -r5' Command - ECC in CPU feature of Cortex-R5

The -r5 option calcuates ECC based on the internal ECC logic in the Cortex-R5 CPU.

```
nowecc - i <ip_file> <memory_map> -o [op_file] -r5
```

3 Return Values - Error Codes

The nowECC tool terminates with a return value of zero upon successful completion. It returns with an error code number and an error message upon the detection of any error condition.

All error codes returned by the nowECC tool are listed in Appendix A.

Appendix A nowECC Error Codes

Table 10 is a list of all error codes returned by the nowECC tool.

Table 10. List of Error Codes for nowECC

Code	Description
01	Error 01 - Not enough input arguments specified
02	Error 02 - Input file is not specified
03	Error 03 - Output file is not specified
04	Error 04 - Fill value is not specified
05	Error 05 - Fill value should be between 0x00 and 0xff
06	Error 06 - No range values specified for the "-r" option
07	Error 07 - No end address specified for the range option
08	Error 08 - End address cannot be less than the start address for the range option
09	Error 09 - No exclude values specified for the "-x" option
10	Error 10 - No end address specified for the exclude option
11	Error 11 - End address cannot be less than the start address for the exclude option
12	Error 12 - No address or symbol specified for the "-s" option
13	Error 13 - No ECC value specified for the "-s" option
14	Error 14 - ECC value for the "-s" option cannot be greater than 0xff
15	Error 15 - No address or symbol specified for the "-s1" option
16	Error 16 - No bit number specified for the "-s1" option
17	Error 17 - Bit value for the "-s1" option should be between 0 to 90
18	Error 18 - No address or symbol specified for the "-s2" option
19	Error 19 - No bit number specified for the "-s2" option
20	Error 20 - First bit value for the "-s2" option should be between 0 to 90
21	Error 21 - Two bit values should be specified for the "-s2" option.
22	Error 22 - Second bit value for the "-s2" option should be between 0 to 90
23	Error 23 - Illegal option found in the command line
24	Error 24 - Range and Exclude options cannot be specified simultaneously
25	Error 25 - Two memory maps cannot be specified simultaneously
26	Error 26 - Invalid Input file format
27	Error 27 - Invalid Tek input, no termination record
28	Error 28 - Invalid Tek input, first character not '%'
29	Error 29 - Invalid Tek input, checksum failure
30	Error 30 - Malloc Error
31	Error 31 - Invalid Tek input, Only 32-bit addresses supported
32	Error 32 - Invalid Tek input, illegal record type in line.
33	Error 33 - Address specified for the "-s1" option is invalid
34	Error 34 - Address specified for the "-s2" option is invalid
35	Error 35 - Input file has too many sections.
36	Error 36 - Invalid Motorola input, first character not 'S'
37	Error 37 - Invalid record type found in Motorola input
38	Error 38 - Checksum failure on Motorola input
39	Error 39 - Invalid Intel input, first character not ':'
40	Error 40 - Invalid record type found in Intel input
41	Error 41 - Checksum failure on Intel input
42	Error 42 - COFF file not for TMS470
43	Error 43 - Problem in reading section in the Input file

Table 10. List of Error Codes for nowECC (continued)

Code	Description
44	Error 44 - Address specified for the -s option is invalid
45	Error 45 - Illegal character found in line of input
46	Error 46 - Invalid range value - Start and End Address are invalid
47	Error 47 - Invalid range value - Start address is invalid
48	Error 48 - Invalid range value - End address is invalid
49	Error 49 - Invalid exclude value - Start and End Address are invalid
50	Error 50 - Invalid exclude value - End Address is invalid
51	Error 51 - Invalid exclude value - Start Address is invalid
52	Error 52 - Input file not found
53	Error 53 - ELF file not for ARM
54	Error 54 - No valid sections found in the input for calculating ECC
55	Error 55 - Both the bit numbers cannot be same for the "-s2" option
56	Error 56 - The address for the "-s1" and "-s2" options cannot be the same
57	Error 57 - Address should be aligned on 64 bit boundary
58	Error 58 - Data exists in invalid memory region.
59	Error 59 - Command file is not specified
60	Error 60 - Command file not found
61	Error 61 - Range values specified overlap
62	Error 62 - Fill command specified more than once
63	Error 63 - Input file specified more than once.
64	Error 64 - Exclude values specified overlap
65	Error 65 - Overlapping memory region in the input
66	Error 66 - Offset value not specified for the "-s1" command
67	Error 67 - Offset value not specified for the "-s2" command
68	Error 68 - Offset not specified for "-s" command.
69	Error 69 - Output file specified more than once
70	Error 70 - Root name not specified
71	Error 71 - Root Name specified more than once
72	Error 72 - Root name should not be longer than 6 characters
73	Error 73 - Symbol not found in the input
74	Error 74 - Symbol table section not found in file
75	Error 75 - Symbol cannot be specified for this file format
76	Error 76 - Input file already contains ECC data
77	Error 77 - Little endian option should not be used. Please remove this option from the command line. This option is unnecessary for Little endian devices.
78	Error 78 - No memory map specified
79	Error 79 - Incorrect F021 option specified
80	Error 80 - Cannot specify -r4 and -r5 options at the same time

Appendix B Revision History

Table 11 lists the changes made since the previous revision of this document.

Table 11. Document Revision History

Reference	Additions/Modifications/Deletions
Section 2.2.14	Changed second list item.

Tool Revision History lists the tool changes made since the previous revision of this document.

Tool Revision History

User's Guide Version	Tool Version	Release Date	Author	Comment
2.15.1	2.15	07/14/2010	Siddharth Deshpande	Initial version. Drafted out from a basic version of TI's ECC generation tool user specification. -Added Little endian support (-le user option) for ARM Cortex -R4 devices.
2.15.2	2.15	02/10/2011	Pratip Kumar	-x option syntax corrected.
2.16.1	2.16	03/01/2011	Pratip Kumar	Cosmetic reformatting Included error 78 and 79. The LF035 and F021 memory map have to be explicitly specified, the default support option removed.
2.17	2.17	08/01/2011	John Hall	Updated version number for new installer.
2.18	2.18	09/05/2011	Sapthagiri Gudihalli	Updated the tool to add support for F021 16M FPGA memory map.
2.19	2.19	10/31/2011	Sapthagiri Gudihalli	Updated the tool to add support for F28M35x M3 and C28 memory maps
2.20	2.20	09/10/2012	Siddharth Deshpande	Updated the tool to add support for Cortex-R5 devices
2.21	2.21	06/12/2013	Siddharth Deshpande	Updated the tool to fix issues : 1. Data in EEPROM range for Intel hex files, 2. nowECC creates 2 bytsof ECC data for an S-record with 8 bytes of Flash data
2.22	2.22	03/31/2014	Siddharth Deshpande	Updated the tool to fix issues : 1. Incorrect ECC address generated for Intel hex files. 2. Removed "-le" option since this option should not be used for Little Endian devices.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265