Bound-T timing analysis tool

Technical Note

Intel® Hex as input to Bound-T

TR-TN-IHEX-001
Version 1
2007-09-27
Tidorum Ltd.

Tidorum Ltd
www.tidorum.fi
Tiirasaarentie 32
FI-00200 Helsinki
Finland

This document was written and is maintained by Niklas Holsti at Tidorum Ltd.

Document reference:	TR-TN-IHEX-001
Document issue:	Version 1
Document issue date:	2007-09-27
Bound-T version:	Various, depending on target processor.
Web location:	*http://www.bound-t.com/tech_notes/tn-ihex.pdf*

Credits:
This document was created with the free OpenOffice.org software, *http://www.openoffice.org/*.

Relevant Bound-T source-file versions:

File	*Revision*
formats-intel_hex.adb	1.2
formats-intel_hex.ads	1.1
formats-intel_hex-opt.ads	1.1
formats-intel_hex-text.adb	1.1
formats-intel_hex-text.ads	1.1

The information in this document is believed to be complete and accurate when the document is issued. However, Tidorum Ltd. reserves the right to make future changes in the technical specifications of the product Bound-T described here. For the most recent version of this document, please refer to the web address *http://www.tidorum.fi/*.

If you have comments or questions on this document or the product, they are welcome via electronic mail to the address *info@tidorum.fi*, or via telephone, fax or ordinary mail to the address given below.

Please note that our office is located in the time-zone GMT + 2 hours (+3 hours in the summer) and office hours are 9:00 -16:00 local time.

Cordially,

Tidorum Ltd.

Telephone: +358 (0) 40 563 9186
Web: http://www.tidorum.fi/
http://www.bound-t.com/
Mail: info@tidorum.fi

Post: Tiirasaarentie 32
FI-00200 HELSINKI
Finland

Credits

The Bound-T tool was first developed by Space Systems Finland Ltd (*http://www.ssf.fi*) with support from the European Space Agency (ESA/ESTEC). Free software has played an important role; we are grateful to Ada Core Technology for the Gnat compiler, to William Pugh and his group at the University of Maryland for the *Omega* system, to Michel Berkelaar for the *lp-solve* program, to Mats Weber and EPFL-DI-LGL for Ada component libraries, and to Ted Dennison for the *OpenToken* package. Call-graphs and flow-graphs from Bound-T are displayed with the *dot* tool from AT&T Bell Laboratories.

Contents

1 INTRODUCTION

1.1 Purpose and Scope

Bound-T is a tool for computing bounds on the worst-case execution time of real-time programs; see reference [1]. Bound-T applies static analysis to the machine-code program in its compiled, linked and executable form. Bound-T must therefore start by reading in the machine-code program from a file, for example a file in the ELF format. Bound-T can read and understand several program-file formats. In addition to the basic machine code (memory load image) a program file usually also contains symbolic debugging information for the program (and this is usually the complex part of the format).

There are different versions of Bound-T for different target processors. Each version supports a set of program formats that are commonly used with that target processor and its cross-compilers. The Bound-T Application Note for each target explains which formats are supported.

This Technical Note supplements the Bound-T User Manual [1] and the target-specific Application Notes by describing how Bound-T reads and models programs represented in the *Intel® Hexadecimal Format* (Intel Hex) as defined by Intel [].

This Technical Note describes Bound-T Intel Hex support in a generic way, without considering a particular target processor. When a version of Bound-T for a target processor supports Intel Hex, the Application Note for that target processor may give additional target-specific details, for example on the mapping of numerical addresses to particular address spaces in the processor.

1.2 Overview

The reader is assumed to be familiar with the general principles and usage of Bound-T, as described in the Bound-T User Manual [1]. The user manual also contains a glossary of terms, some of which may be used in this Technical Note. You may also may find it useful to first read the Bound-T Application Note for your target processor.

The remainder of this document is structured as follows:

- Chapter 2 describes the main features of the Intel Hex format and how they relate to the functions of Bound-T. The chapter also gives an overview of how Bound-T reads and uses an Intel Hex file.
- Chapter 3 explains the warning messages that Bound-T may emit if it finds some problems or unsupported features in an Intel Hex file.
- Chapter 4 explains the possible error messages similarly.

1.3 References

[1] Bound-T User Manual.
Tidorum Ltd, Doc. ref. TR-UM-001.
http://www.bound-t.com/user-manual.pdf.

[2] Hexadecimal Object File Format Specification.
Intel Corporation, Revision A, January 6, 1988.
For example, *http://microsym.com/editor/assets/intelhex.pdf*.

[3] CDB File Format.
Lenny Story, SDCC Development Team, 2003-03-21.

[4] ASxxxx Assemblers and ASLINK Relocating Linker.
Alan R. Baldwin, Kent State University, Version 2.0, August 1998.

1.4 Typographic Conventions

We use the following fonts and styles to show the role of pieces of the text:

-option	A command-line option for Bound-T or other tools.
symbol	A mathematical symbol or variable.
`text`	Text quoted from a text / source file or command.
Record Field	The name of an Intel Hex record (type), or the name of a field in an Intel Hex record.

1.5 Abbreviations and Acronyms

See also reference [1] for abbreviations specific to Bound-T and Appendix B of [] for a glossary of Intel Hex terms.

Hex	Hexadecimal
WCET	Worst-Case Execution Time

2 THE INTEL HEX FORMAT

2.1 Intel Hex Features

Intel Hex

The Intel Hexadecimal Format, briefly Intel Hex, is a representation of the memory contents of a processor. Typically these memory contents represent the complete machine code of a compiled and linked program for some target processor, but it might also be just a part of that code, or part of data.

No symbolic debugging information

An Intel Hex file contains no symbolic debugging information. If you analyse a program with Bound-T using only an Intel Hex representation of the program, you must give all input in machine-code terms. For example, you must select the root subprogram by its entry address, not by its name. Moreover, the output from the analysis will give only machine addresses, not source-code identifiers or source-line numbers.

For a given target processor Bound-T may accept additional input files that provide the symbolic debugging information, for example CDB files from the SDCC compiler [3]. Please refer to the Application Note for your target processor.

Intel Hex records and record types

Intel Hex as Intel defined it in [2] is a textual format that represents binary data using hexadecimal numbers (base 16). An Intel Hex file is a sequence of *records* where each record is a line of text and vice versa. The first character in each record/line is a colon ':'. All the other characters are hexadecimal digits ('0' to '9' and 'a' to 'f' or 'A' to 'F'). These hexadecimal digits are interpreted two at a time and each such pair represents the value of an 8-bit octet (there is always an even number of hex digits in the record). Thus, ignoring the initial colon, an Intel Hex record represents a string of octets. In the following we take this view of an Intel Hex record

There are several *types* of record. In each type the octets in the record are grouped into data *fields* in different ways. However, all Intel Hex records start with the following fields:

- a one-octet *record length* field, RECLEN,
- a two-octet *address* field, LOAD OFFSET,
- a one-octet *record type* field, RECTYP.

The rest of the fields in the record depend on the record type. The last octet in each record is a *check-sum* of the record's contents – the CHKSUM field.

The Intel document [2] defines six record types. However, four types can occur only in Intel Hex files for 16- or 32-bit processors. Bound-T currently supports Intel Hex only for 8-bit processors with 16-bit addresses (eg. [4]) which means that only two record types are supported:

- Data Record: Defines the memory contents at a given memory address (LOAD OFFSET).

- End Of File Record: Ends the file and (in some variants of Intel Hex) may define the starting address of the program (as the LOAD OFFSET).

Overall Intel Hex structure

As Bound-T only supports the two types of record listed above, the overall structure of an Intel Hex is simply some Data Records followed by an End Of File Record. However, the general Intel Hex reader in Bound-T allows any sequence of records and leaves it to the target-specific parts of Bound-T to decide what to do with each record, for example, whether to accept more Data Records after an End Of File Record.

2.2 How Bound-T Reads an Intel Hex File

Auto-detecting Intel Hex format

A given version of Bound-T can often read several forms of program files. If the user does not specify the file-format with a command-line option Bound-T will try to determine the format from the given program file itsef. This is called “auto-detecting” the format.

To determine if a given file is an Intel Hex file Bound-T tries to read one Intel Hex record from the start of the file. Bound-T considers the file to be Intel Hex if the first line of the file is a valid Intel Hex record.

At this point Bound-T reads the file using a binary access method. The “first line” is defined as the octet string from the first octet in the file up to but not including the first octet that contains the CR or LF control-code.

Reading and loading an Intel Hex file

Bound-T reads an Intel Hex file as a text file in one pass from the first line/record (usually a Data Record) to the last line/record (usually an End Of File Record). The file is expected to have the standard line terminators for text files on the current host system.

While reading the file, for each record/line Bound-T:

- checks that the record consists of a ':' followed by an even number of hex digits, for a total of at least 11 characters,
- checks that the RECLEN field matches the actual length of the record (line),
- checks that the RECTYP field is a known value (indicates a Data Record or an End Of File Record),
- checks that the CHKSUM field contains the actual check-sum of the record, and
- passes the record to further target-specific processing.

The target-specific processing of a Data Record typically loads the code/data from the record into a memory-image data-structure that Bound-T later uses to analyse the program.

2.3 Target-Specific Parts of Intel Hex Processing

The process described above for reading and loading an Intel Hex file passes each record to further processing that is specific to the target processor for which the Intel Hex program is compiled and linked. The actions that can be defined in a target-specific way are the following:

- Mapping the LOAD OFFSET value to a memory address in the target processor.
- Interpreting the contents of a Data Record using the word size, endianness, and other properties of the target processor.

These target-specific steps in Intel Hex processing should be explained in the Bound-T Application Note for the relevant target processor, as should be the warning or error messages that may issue from these steps.

3 INTEL HEX WARNING MESSAGES

The following table lists the warning messages that Bound-T may emit to highlight some problems or unsupported features in an Intel Hex file. The messages are listed in alphabetical order, perhaps slightly altered by variable fields in the message; such fields are indicated by *italic* text. The Bound-T User Manual [1] explains the general form of warning messages. The Bound-T Application Note for the relevant target processor may describe additional warning messages relating to the target-specific steps in Intel Hex processing.

The probable reason for any of these warnings is either a damaged Intel Hex file, or a file that uses a version of Intel Hex that Bound-T does not support. To correct the problem you should obtain an undamaged program file, in a supported version of Intel Hex, or in some other format that Bound-T supports for your target processor.

Table 1: Warning messages for Intel Hex

Warning Message	*Meaning*
Intel-Hex file is empty: *name*	The file by this *name* seems to be an empty file. Note that auto-detection never classifies an empty file as an Intel Hex file, so for this warning to appear there must be an explicit option or other user input that makes Bound-T assume Intel Hex format for this file.

4 INTEL HEX ERROR MESSAGES

The following table lists the error messages that Bound-T may emit to highlight severe problems or unsupported features in an Intel Hex file. The messages are listed in alphabetical order, perhaps slightly altered by variable fields in the message; such fields are indicated by *italic* text. The Bound-T User Manual [1] explains the general form of error messages. The Bound-T Application Note for the relevant target processor may describe additional error messages relating to the target-specific steps in Intel Hex processing.

The probable reason for any of these errors is either that the Intel Hex file is damaged or that the file uses a version of Intel Hex that Bound-T does not support. To correct the problem you should obtain an undamaged program file, in a supported version of Intel Hex, or in some other format that Bound-T supports for your target processor.

Table 2: Error messages for Intel Hex

Error Message	*Meaning*
Intel-Hex: Computed checksum *S* differs from record value *R*	The check-sum computed from the octets in the record (*S*, in hex) is not the same as the value in the CHKSUM field (*R*, in hex).
Intel-Hex: Record cannot contain a '*C*' character	The body of the record (after the initial ':') contains the character *C* which is not a hexadecimal digit.
Intel-Hex: Record cannot have a length of *L* characters	The length *L* (decimal) of the record (the number of characters in the line) is less than the minimum length (11 characters) or is even.
Intel-Hex: Record cannot start with '*C*'	The first character in the record is *C*, not ':' as required.
Intel-Hex: Record of length *L* characters cannot have *D* data octets	The RECLEN field claims that the record contains data for *D* (in decimal) octets, which requires 2*D* hexadecimal digits, but this conflicts with the overall length of the record which is *L* (in decimal) characters.
Intel-Hex: Record too long; at most *L* characters allowed	The text line is too long to be an Intel Hex record. (The 8-bit RECLEN field sets an upper limit on the length.)
Intel-Hex: Record type *T* is not understood	The RECTYP field has a value *T* (in hex) that does not correspond to a supported record type (0 = Data Record, 1 = End Of File Record).
Intel-Hex: The characters "*S*" are not a 16-bit hexadecimal number	A 2-octet record field *S* contains some non-hexadecimal characters. (This error message should never appear because the characters are checked earlier and a different error message is emitted.)
Intel-Hex: The characters "*S*" are not an 8-bit hexadecimal number	A 1-octet record field *S* contains some non-hexadecimal characters. (This error message should never appear because the characters are checked earlier and a different error message is emitted.)
Intel-Hex: The error is in record number *N*	This is an auxiliary message that may follow one of the other error messages. If the number of the erroneous record/line did not appear in the locus fields [1] of the earlier error message, the line number (*N*) is given in this message. The first line in an Intel Hex file is line number 1.

Tidorum Ltd

Tiirasaarentie 32
FI-00200 Helsinki, Finland
www.tidorum.fi
Tel. +358 (0) 40 563 9186
VAT FI 18688130