T-Engine Forum
Technical Report

TEF040-S104-1.00.01/en
2006-06-06

T-Format (4): Binary Code Format

Number: TEF040-S104-1.00.01/en
Title: T-Format (4): Binary Code Format

Status: [ ] Working Draft, [ ] Final Draft for Voting, [X] Technical Report
Date: 2003/02/28 First Edited
2003/08/01 Updated to 1.00.00
2006/06/06 Updated to 1.00.01

Table of Contents

Foreword

The T-Engine Project has been established to offer an open, real-time standardized development environment with the aim of achieving a ubiquitous computing environment where everything has a computer incorporated in it and is connected to a network. T-Engine is trying to offer an efficient development environment for the development of portable information devices, home electronic appliances and other network devices in a short period of time.

T-Engine employs network security architecture called eTRON whose architecture is intended to prevent tapping, falsification, and disguise of malicious users so that electronic information can be safely delivered to the other party through insecure network channels.

To support efficient development, T-Engine standardizes hardware (T-Engine board) and real-time kernel (T-Kernel), and especially encourages distribution of middleware. T-Engine is also aimed at smoothing cooperation among semiconductor makers, hardware makers, software makers and system manufacturers, encouraging mutual business dealings, reducing development time and cost, thus enabling high value added product offerings in a short period of time. The combination of advanced semiconductors, implementation and software technologies in T-Engine makes it suitable and unrivaled for the development of advanced application products.

Scope

T-Format specifies the code format of middleware and application software that run on T-Engine/T-Kernel. T-Format includes the following three specifications:

1. Source-code style guideline
   Source-code style format for middleware and application that run on T-Engine/T-Kernel. By using the format, source codes written by various vendors can be combined and compiled/linked as one program.

2. Standard binary format
   Standard executable format for distributing, in binary code, middleware and application software that run on T-Engine/T-Kernel. Executable code format and debugger symbol format are defined.

3. Standard documentation format
   Type and format of documentation that is attached to distributed middleware and application software.

This document provides guidelines for the binary code style for distribution as part of T-Format. It does not cover debugging information such as debugger symbol format, because debugging information is distributed in source code.

Normative References

[1] T-Engine Forum. "T-Format (3) : Global Symbol Naming Rule in C Language", TEF-040-S103, 2002.

[2] Free Software Fundation. "GCC, the GNU Compiler Collection". http://gcc.gnu.org/

[3] Free Software Fundation. GNU. http://gnu.org/

[4] Tool Interface Standard Commitee. "Executable and Linkable Format (ELF)". Specification Version 1.1, 1993.

Terms and Definitions

Middleware See "T-Format(3)[1] Terms and Definitions."

1. T-Engine Middleware Models

See “T-Format(3)[1] T-Engine Middleware Models.”

2. Binary Format for Distribution

The T-Format Binary Format meets the following two conditions:

1. Meet the ELF (Executable and Linking Format)[4] specification. (requirement)
2. The ELF specification does not fully cover CPU operations, since part of them is dependent on compiler, etc. Such part shall follow the implementation of GNU[3]-based reference development environment that the T-Engine Forum separately defines.

For the reasons described below, there is no specification of binary format that does not depend on implementation and runs properly with any CPU currently implemented on T-Engine, as well as no implementation to run every binary format used for T-Engine. We have therefore reached a conclusion that as a practical solution, we should define the binary code compiled by GCC (GNU Compiler Collection)[2] of a reference environment as the standard binary code format.

2.1 Compiler Difference

ELF (Executable and Linking Format) could be considered a candidate for the standard binary format, since many compilers support it and it is the major binary format of the GNU development environment. ELF, however, could have compiler-dependent information, and an object file in ELF format that runs properly with one compiler might not run as expected with another. The ELF specification alone, therefore, does not satisfy what T-Format requires for distributing software in binary format.

2.2 ABI (Application Binary Interface) of GCC

The ABI of GCC is basically the same regardless of the version of GCC, but not without exception. The ABI of one version of GCC could be partially different from another. For example, the ABI of C++ is changed in the GCC3.1 from the GCC3.0. Moreover, GCC is distributed in source code, and each user builds it in his/her own environment. Depending on their building option, one user's ABI could be different from another's. Therefore, the ABI of a binary code compiled by the same version of GCC could be different from one user to another. For example, one option adds "_" to C language's global symbols, but another option does not.

For these reasons, specifying the release version of GCC does not define the binary format that satisfies the conditions that the T-Format requires of distribution of software

in binary format, either.