

USAGE: TRANSACTIONAL MEMORY SUPPORT FOR GCC

This document describes the usage of the preliminary implementation for TM in GCC. It contains the sections covering the install, usage, how to program with transactions, and features of the release. Since this is a pre-release not all features are fully implemented and thoroughly tested. If you encounter problems, discover bugs or simply want to give some feedback, please write an email to schindew@ira.uka.de. Thank you very much for your interest in this project.

1. INSTALL

Please download version 4.3.2 of GCC from one of the GCC mirrors: <http://gcc.gnu.org/mirrors.html>. If you want to test and install the original GCC, follow the instructions given on

<http://gcc.gnu.org/install/>. If you want to start with TM patched GCC do the following: unpack the GCC sources and the TM patch. While unpacking the source directory (referred to as $\$(SRC_DIR)$) of GCC is created. Apply the patch using the program `patch`. Since the patch only covers the `gcc` subdirectory of the source directory, the syntax should look like:

```
 $\$(SRC\_DIR):patch -p0 < GnuTM_patch_0.94.txt$ 
```

Then create the GCC build directory (also referred to as $\$(OBJ_DIR)$) and configure and make your patched GCC. Distinct $\$(SRC_DIR)$ and $\$(OBJ_DIR)$ directories are preferable to set up GCC (see <http://gcc.gnu.org/install/configure>) for details. We recommend to use the following configure options:

```
 $\$(OBJ\_DIR):../\$(SRC\_DIR)/configure --enable-checking=all$   
 $--disable-bootstrap --prefix=\$(OBJ\_DIR) --program-suffix=$   
 $-4.3.2gtm --enable-languages=c$ 
```

When the configure returned successfully, start the make process with:

```
make CFLAGS="-g3 -O0"
```

and then install:

```
make install
```

Since building GCC from scratch can take a while, we suggest to install TinySTM meanwhile. Go to <http://tinystm.org/tinystm/> and download and install **version 0.9.0b1**. Please notice: this is not the latest version of TinySTM. The GTM builtins used to interface with TinySTM are not yet adjusted to the newest version. Then unpack and install TinySTM following the instructions delivered with the package. Please run the test cases to make sure that TinySTM is working correctly. When the patched GCC and TinySTM are installed successfully, you may move on to the next section.

2. USAGE

The GCC enhanced with support for Transactional Memory should now be accessible in:

```
$(OBJ_DIR)/bin/gcc-4.3.2gtm
```

Simply invoke it directly or add the directory to the search path of your environment. The options used to enable transactions are the following:

```
bin/gcc-4.3.2gtm -O1 -fgnu-tm -Wall -o test test.c
-L$(TinySTM_Lib_Path) -ltstm -lpthread
```

The `-O1` enables optimisations and is necessary to make the GCC transform the program into SSA form. Our check-pointing pass works on SSA. Thus, if the switch is omitted the check-pointing will not be done, resulting in not correctly synchronised programs. `-fgnu-tm` is the switch that enables the TM passes. Without the switch the `#pragma tm atomic` will be ignored and neither instrumentation nor check-pointing are performed. If you want to use OpenMP to parallelise your code, you will have to give the `-fopenmp` switch to the compiler.

3. PROGRAMMING WITH TRANSACTIONS

Our current release only supports bare transactions. This means that variables used inside of a transaction that are potentially shared should be instrumented with calls to the STM run-time. In our case the STM run-time refers to TinySTM. A transaction is marked with the `#pragma tm atomic` the following block will be treated differently from the rest of the code and should execute in isolation with respect to other transactions. At run-time isolation and consistency are assured by the underlying STM. If you encounter misbehaving programs, please send us some feedback including a description of the problem and the code. Thank you very much for your cooperation.

4. FEATURES

The implementation supports the extension with the additional pragma for the programming language C. For now Gnu-TM just supports transactions with flat nesting. Meaning that if transactions are nested, the inner ones are collapsed into the outer most one. Thus, only one transaction remains and no partial rollbacks are performed. Function cloning and the use of different attributes for functions are on our list for future work, but not yet implemented. The `tm_abort`-keyword is not implemented. The cases where (`break`, `return`, `continue`, or `goto`) leave the scope of a transaction should be instrumented with a call to `stm_commit` and, thus, be handled correctly in a transactional context.

E-mail address: schindew@ira.uka.de