TriMedia Technologies Inc. SDE v2.2

Release Notes

Printed in the United States of America

SDE v2.2 Release Notes

What's New?

Welcome to Version 2.2! This is the newest release of the TriMedia Software Development Environment and it provides extra improvements in functionality, optimization and several user requested bug fixes. This section will review major changes and new additions.

For more information and pointers to related information in the documentation, please read the Migration Guide (Chapter 5 of Book 1: Getting Started), and the Important Points to Remember section of the Troubleshooting Chapter (Chapter 6 of Book 1: Getting Started).

The V2.2 release of the TriMedia Software Development Environment (SDE) includes a complete set of program compilation, simulation, debugging and execution tools. It also includes device libraries for TriMedia devices and example applications which use the device libraries. The distribution includes complete documentation in Adobe (TM) Acrobat (TM) format.

pSOS 2.5

As of version 2.2 the SDE has moved to pSOS v2.5 (from pSOS v2.1). The following list gives some of the highlights of pSOS 2.5. For additional information on these changes refer to the pSOSystem System Concepts and the pSOSystem System Calls manuals.

* A new pSOS+ object, mutex, has been added to support mutual exclusion among tasks. The kernel has added support for the following mutex operations: `mu_create`, `mu_delete`, `mu_ident`, `mu_lock`, `mu_unlock`, and `mu_setceil`.
* A new pSOS+ object, condition variable, has been added to achieve complex synchronization operations among tasks, when used in conjunction with mutexes. The kernel has added support for the following condition variable operations: `cv_create`, `cv_delete`, `cv_ident`, `cv_wait`, `cv_signal`, and `cv_broadcast`.
* A new pSOS+ object, called the task-specific data object, has been added to the pSOS+ kernel for allocation, easy access and management of task-specific data. The kernel provides new system calls: `tsd_create`, `tsd_delete`, `tsd_ident`, `tsd_setval`, and `tsd_getval` to manage task-specific data.
* Support has been added for task variables which will facilitate the porting of products that are process based to become task based. The kernel provides new system calls, `t_addvar` and `t_delvar` for this purpose.
* Support has been added to obtain information about kernel objects, as well as other system information. The new system calls provided for this purpose are: `pt_info`, `q_info`, `q_vinfo`, `rn_info`, `sm_info`, `t_info`, `cv_info`, `mu_info`, `tsd_info`, `ob_roster`, and `sys_info`.
* Support has been added for peek functionality for message queues, so that the content of a message item can be retrieved without removing the message from the queue. The `q_receive` and `q_vreceive` system calls have been augmented for this purpose.

* Support has been added for a pSOS+ task to wait on a combination of message queues, semaphores, asynchronous signals, signal events and time-out, at the same time. The new system calls provided to register a task and a set of events to notify it of an appropriate operation, are `sm_notify, as_notify, q_notify`, and `q_vnotify`. The task must wait at the `ev_receive` call for notification of any event. The mechanisms are described in detail in the pSOSystem System Concepts manual, and the API is defined in the pSOSystem System Calls manual.
* Support has been added for a per-task time-slice quantum used in the round-robin scheduling of kernel tasks. The quantum can be modified with the help of a new system call, `t_tslice`.
* A new pSOS+ system call, `tm_getticks`, has been added to obtain the number of system ticks elapsed since the system startup.
* Support has been added for bounded and counting semaphore objects. You can create a binary semaphore, which is created with a count of 1.

There are some release to release migration concerns when moving to the new pSOS. An overview of these concerns is given in the Backward Compatibility section of this document. For a full description of changes please refer to the Migration Guide chapter of Book 1 of the docmentation set.

The Compiler Tool Chain

There were many small changes to the compiler tool chain during this release including changes to the profiling code. If you currently compile your code using profile directed feedback, you will need to rebuild your profiles with the 2.2 compiler.

HP-UX

SDE v2.2 no longer supports HP-UX as a compilation environment.

tmcc

The `-noloadspec` option has been added to **tmcc**. This option tells the compiler tool chain not to do load speculation. This option is a more general implementation of the tmsched option `-noloadspec`. Using this new tmcc option will signal **tmcfe**, **tmccom** and **tmsched** not to generate speculative loads.

64 bit Floating Point Emulation -fp64

SDE release v2.1 included the **tmcc** option `-uselongdouble64` which changed the representation of the `long double` C type to use IEEE double precision format (64 bits). SDE v2.2 moves to a faster fully ANSI C compliant option called `-fp64.`

Previously, TCS represented C types `float, double and long double` using IEEE single precision format (32 bits), and the C/C++ compiler generated TriMedia hardware floating point operations for all floating point arithmetic operations.

When the `-fp64` option is used, TCS represents the C types `double` and `long double` using IEEE double precision format (64 bits); the `float` C type is still represented using IEEE single precision format (32 bits). With `-fp64` single precision floating point computations are still done in the hardware but double precision floating point operations are done using a software emulation library.

Because the size of `double and long double` objects is different with the `-fp64` option, the TCS distribution includes two separate versions of the standard C and C++ libraries and tmcc links with the appropriate version. Users should not mix object files compiled with and without the `-fp64` option.

The standard C library normally defines only double versions of the standard `<math.h>` mathematics library functions, such as

```
extern  double cos(double d);
```

The `-fp64` version of the standard C library also defines corresponding float routines, such as

```
extern float _f_cos(float f);
```

for users want access to the hardware performance of 32 bit floating point while also having available the greater precision provided for doubles and long doubles with `-fp64`. See the Library Function chapter of the C Users Guide for details.

If an application developer intends to use -fp64 but wishes to do some single precision (i.e. C type float) arithmetic on hardware to aid in performance, floating point code must be written with care. In addition to using the float specific functions mentioned above, users should note that unsuffixed floating point constants are of type double, not float; floating point constants suffixed with 'F' (e.g. 1.234F) are of type float. If you neglect to put the 'F' suffix on constants that you intend to be single precision, you may unintentionally bring the 64 bit floating point emulation library into use on your calculation.

Migration Note: The `-uselongdouble64` scenario available in 2.1 (C types float and double are 32 bits, C long double is 64 bits) is not available in SDE v2.2. To aid in migration, the _ld_* functions that were available in SDE v2.1 are also available in SDE v2.2. Please note that these compatibility routines will be phased out in a future release.

The Debugger

The 2.2 release of **tmdbg** uses Windows registry entries corresponding to TriMedia Technologies instead of Philips Semiconductors. If you start up the new debugger you may notice that several settings such as screen geometry, previously loaded files, jtagbase address, etc. will have been lost. It is fine to continue without the old settings, but if you

would like the new debugger to use the old settings, please refer to the Migration Issues section of this document.

Debugging PSOS applications

SDE v2.2 supports pSOS 2.5. **tmdbg** for SDE v2.2 does not support debugging pSOS 2.1 applications.

JTAG Driver API

SDE v2.2 comes with a new API to allow customers to write their own JTAG drivers and thereby allow **tmdbg** to support more than the current JTAG cards. A new example `jtagdll` has been added to aid in developing for this new interface.

Debugging hosted applications over JTAG

The 2.2 debugger supports debugging hosted applications over JTAG. There is a special version of the TriMedia Debug Monitor, `libmon_j.o` which forces debug communication to be "tmNoHost" always, even if the application was compiled with -host WinNT, for example.

Re-compiling the application is not necessary, only re-linking. The steps required to link and download your program are as follows:

1. Link the application with the special version of the debug monitor libmon_j.o. This can be accomplished by using the tmcc option "-libmon={libmon file name}". The complete path must be specified. For example: `tmcc -g -libmon=${TCS}/lib/{el,eb}/libmon_j.o -host WinNT hello.c`
2. Download the trimedia `.out` file from your host as you would normally, with TMGMON/TMRUN, etc.
3. Launch the **tmdbg** debugger.
4. The debugger will automatically detect an application is already running, connect to the debug monitor there, and collect all debug information from the target. The debug session is now ready.

The Execution Hosts - tmman

As of SDE v2.2 **tmman** uses a registry entry corresponding to TriMedia Technologies, instead of Philips Semiconductors. Please refer to the Migration Issues section following this one for information on how to move your current registry entries to work with SDE v2.2.

A new registry option has been added to SDE v2.2 that allows the user to exclude the TM-PCI cards based on SubSystemID and SubVendorID. Sometimes users have many TM-PCI cards with different SubSystemIDs and SubVendorIDs. If they wish only to use

those TM-PCI cards that match a desired SubsystemID and SubVendorID (excluding any other TM-PCI cards) they can now specify the acceptable SubSystemIDs or SubVendorIDs in the registry. Please note that this new option should be used very carefully. If errors are made while entering the IDs in the registry, then **tmman** may not recognize any TM-PCI card in the system.

SDE v1.1 to SDE v2.2 Performance Comparisons

The Component Manager and the Registry add code size and additional startup overhead to every executable. This should be taken into account when comparing SDE v1.1 and SDE v2.2 executables. For **tmsim** hosted executables, users should use the **tmsim** `-ns` option to eliminate the counting and profiling of startup overhead in v2.2 binaries. For TM executables, v2.2 by default links in Component Manager, the registry and (when not `-host tmsim`) several different board support packages. This allows the same executable to run on different TM boards, but at the cost of added code size due to multiple board support packages. For meaningful size comparisons with SDE v1.1 executables, users should use an explicit **tmcc** `-board=` option to specify just the board they need when building a SDE v2.2 executable. For performance comparisons of **tmsim** hosted executables vs. TM executables, note that **tmsim** does not simulate SDRAM refresh cycles. This can cause up to 4% difference between **tmsim** and the TM hardware execution times. Also, remember that **tmsim** hosted executables are linked with different host communication files than TM executables, which can result in data residing at different locations and therefore can produce differences in execution time due to cache effects.

Device and Applications Libraries

For details on Device Library and Application Library changes, please refer to the Migration Guide in Book 1 of the documentation set.

Documentation

As in 2.1, the documentation has been reorganized into nine books:

* Book 1: *Getting Started*
  — This book contains the new Migration Guide (chapter 5) as well as the Important Points to Remember section added to the Troubleshooting Guide. We recommend you read both of these sections before you start designing or porting an application to work with SDE v2.2
* Book 2: *Cookbook*
* Book 3: *Software Architecture*
* Book 4: *Software Tools*
* Book 5: *System Utilities*

* Book 6: *Audio Support Libraries*
* Book 7: *Video Support Libraries*
* Book 8: *Graphics Libraries*
* Book 9: *Communications Libraries*

These books include documentation for components that do not appear on the SDE v2.2 CDs. For the most up-to-date documentation, refer to the softcopy documentation on the CD or the on-line documentation available at our website. For further information on the website, please see the *Support* section of this document.

This release is documented using Acrobat 4. Please ensure that the search plug-in is installed!

Important Fixes in SDE v2.2

* Fixes have been provided for two compiler assertions reported when compiling at `-O2` or higher.
* A fix has been provided for a range analysis code generation bug. Most occurrences of this problem happened when one looped through an array backwards.
* A code generation bug that occurred only with `-g` when you had unused locals has now been fixed.
* Many problems that occur when using the TCS pragmas to change the `opt` level for a given function have been fixed.. This does not remove the restriction against using TCS_O4 or TCS_O5 if your programs command line opt level is -O3 or lower.
* A memory leak that occurred because `freopen` had failed to free a file descriptor has now been fixed.
* The debugger now supports DP commands even if the `.out` file contains uncached, or locked data sections.
* The debugger also supports dumping DP while the target is running.
* Extensive enhancements to the C++ support have also been done in **tmdbg**.
* Some internal errors issued by the **tmdbg** GUI when visiting tasks have been fixed.
* **tmman** WinNT SGBuffer functions no longer corrupt NonPaged pool headers.
* Fixes were added to **tmman** to resolve a hanging problem that occurred on dual-pentium NT systems.
* An error in processor initialization for NT `tmman.sys` was also fixed.
* **tmcc** and **tmdbg** now recognizes files with the `.cxx` extension as C++.

Migration Issues

Migrating pSOS Applications

The TriMedia SDE v2.2 release includes the TriMedia port of pSOS v2.5. The new version is backward compatible with the pSOS v2.1 port except for the following changes that will be required if you have your own `drv_conf.c`:

* a change in InstallDriver which now has ten arguments instead of nine.
* the function formerly called `slow_entry` should now be called `_psos_slow_entry.`

If you use your own `sysinit.c` file you will also have to migrate over the latest changes.

There have also been several new defines added to the `sysconf.h` file.

For further information please refer to the Migration Guide chapter of Book 1 of the documentation set.

Migrating Registry Entries

SDE 2.2 is the first release of the TriMedia Software Development Environment from TriMedia Technologies Inc. With this release many of the software tools which use the windows registry will now be looking at a registry entry corresponding to TriMedia Technologies Inc., instead of Philips Semiconductors. This will cause you to lose some defaults and settings. If you installed SDE 2.2 from the CD the following registry entries under
`HKEY_LOCAL_MACHINE\\SOFTWARE\\PhilipsSemiconductors\\Trimedia\\TMMan` were automatically migrated:

```
HostTraceBufferSize HostTraceLevelBitmap HostTraceType
TargetTraceBufferSize TargetTraceLevelBitmap TargetTraceType
MemorySizeMailboxCount ChannelCount VIntrCount MessageCount
EventCount StreamCount NameSpaceCount MemoryCount SGBufferCount
SpeculativeLoadFix PCIInterruptNumber MMIOInterruptNumber MapSDRAM
TMRunWindowSize DefaultEndianness TMGMonDDraw TMCRTDebug TCSPath
DLLPath
```

The following device specific subkeys from
`HKEY_LOCAL_MACHINE\\SOFTWARE\\PhilipsSemiconductors\\TriMedia\\TMMAN` were also copied for the first 32 devices:

```
ClockSpeed CacheOption SystemBaseAddress SDRAMBaseAddress
MMIOBaseAddress
```

The following section gives instructions on how to copy your old registry entries for use with SDE v2.2.

tmman

tmman keeps several user settings in the registry. The following steps explain how to copy your current tmman information to the new location.

1. Go to the "Start" menu's "Run" option, and type "regedit". This will start the Windows Registry Editor.
2. Go to the entry:
   HKEY_LOCAL_MACHINE\SOFTWARE\PhilipsSemiconductors\Trimedia
3. Choose the "Registry" menu's "Export Registry File" option. The SelectedBranch radio button should have been automatically selected, with the branch name filled in (as specified above: HKEY_LOCAL_MACHINE\SOFTWARE\PhilipsSemiconductors). Save it into a temporary file, say D:\mytmman.reg.
4. Using an ascii text editor, like Notepad, open the registry file saved in the previous step.
5. Change every instance in the file of,
   [HKEY_LOCAL_MACHINE]\SOFTWARE\PhilipsSemiconductors\Trimedia
   into:[HKEY_LOCAL_MACHINE]\SOFTWARE\TrimediaTechnologies
6. Save the .reg file.
7. Go back to the Registry Editor and import the updated file by choosing the "Registry" menu's "Import Registry File" option. Select the registration file which was just modified above (D:\mytmman.reg in our example).
8. You should get a dialog box saying: Information in D:\mytmman.reg has been successfully entered into the registry.

tmdbg

If you start up the v2.2 debugger, you may notice that several settings such as screen geometry, previously loaded files, jtag base address, etc. will have been lost. It is fine to continue without the old settings, but if you would like the new debugger to use these the old settings, you must perform the following steps:

1. Go to the "Start" menu's "Run" option, and type "regedit". This will start the Windows Registry Editor.
2. Go to the entry:
   HKEY_CURRENT_USER\Software\PhilipsSemiconductors\Trimedia\TMDbg
3. Choose the "Registry" menu's "Export Registry File" option. The SelectedBranch radio button should have been automatically selected, with the branch name filled in (as specified above:
   HKEY_CURRENT_USER\Software\PhilipsSemiconductors\TriMedia\TMDbg). Save it into a temporary file, say D:\mytmdbg.reg.

4. Using an ascii text editor, like Notepad, open the registry file saved in the previous step.
5. Change line 3 of the .reg file,
   from:[HKEY_CURRENT_USER\Software\PhilipsSemiconductors\Trimedia\TMDbg
   into:[HKEY_CURRENT_USER\Software\TrimediaTechnologies\TMDbg]
6. Save the .reg file.
7. Go back to the Registry Editor and import the updated file by choosing the "Registry" menu's "Import Registry File" option. Select the registration file which was just modified above (D:\mytmdbg.reg in our example).
8. You should get a dialog box saying: Information in D:\mytmdbg.reg has been successfully entered into the registry.

Backward Compatibility

If you use profile directed feedback when compiling your applications you should generate new profiles when you move to a new level of compiler.

SDE v2.1 to SDE v2.2 Compatibility

Source Level Compatibility

With the exception of two changes in pSOS, SDE v2.2 is completely source level compatible with SDE 2.1. If you developed for pSOS 2.1 you will find that the `InstallDriver` function now has an extra parameter, the `slow_entry` function has been renamed `_psos_slow_entry`, there is some new initialization code in `sysinit.c` that must be ported, and several new defines have been added to the `sysconf.h` file. For more details refer to the migration guide in Book 1 of the documentation set.

SDE 2.1 source modified to work with the `-useld64` option will work with SDE 2.2's -fp64 option; however, computations involving doubles will now have the larger 64 bit precision.

Object Level Compatibility

All SDE v2.1 objects can be linked and run with SDE v2.2 with the following exception:

* Due to changes in the calling conventions for 64bit floating point, any object files that were compiled using `-uselongdouble64` or `-useld64` on version 2.1 must be recompiled (with -fp64) before they can be linked with v2.2 -fp64 object.

Module Level Compatibility

In general SDE v2.2 is binary compatible with SDE v2.1. There have been no tmman interface changes.

Although all attempts have been made to make code compiled on SDE 2.1 with `-useld64` or `-uselongdouble64` compatible with SDE 2.2, it is recommended that the user recompile with -fp64.

The SDE v2.1 debugger is not able to debug applications built with SDE v2.2. The SDE v2.2 debugger is able to debug v2.2, v2.1 and v2.0 applications. However the SDE v2.2 debugger cannot debug pSOS 2.1 applications.

SDE v2.0 to SDE v2.2 Compatibility

Source Level Compatibility

With the exception of two changes in pSOS, SDE v2.2 is completely source level compatible with SDE 2.0. If you developed for pSOS 2.1 you will find that the `InstallDriver` function now has an extra parameter, the `slow_entry` function has been renamed `_psos_slow_entry`, there is some new initialization code in `sysinit.c` that must be included and several new defines have been added to the `sysconf.h` file. For more details refer to the migration guide in Book 1 of the documentation set.

SDE 2.2 source modified to work with the -fp64 option will generate numbers of different precision when compiled with SDE v2.0 (which does not support 64-bit floating point emulation).

Object Level Compatibility

All SDE v2.0 objects can be linked and run with SDE v2.2 with the following exceptions:

* Objects that are compiled with -fp64 should not be linked with SDE v2.0 objects that use double or long double.
* There have been several important fixes to the debug stabs generated for C++. If you are debugging C++ programs you should recompile with SDE v2.2.
* Due to changes in the debugger interface to pSOS, the pSOS and debug monitor used when debugging must be from the same level of product. That is, if you are using the v2.0 version of pSOS you must link in the v2.0 version of the debug monitor. The results of using the v2.0 pSOS with the v2.1 or v2.2 debug monitor are undefined.

Module Level Compatibility

In general SDE v2.2 is binary compatible with SDE v2.0. There have been no tmman interface changes.

The SDE v2.0 debugger is not able to debug applicationss built with SDE v2.2. The SDE v2.2 debugger is able to debug v2.2, v2.1 and v2.0 applications. However the SDE v2.2 compiler cannot debug pSOS 2.1 applications (pSOS 2.1 was shipped with SDE v2.0).

SDE v1.1 to SDE v2.2 Compatibility

Source Level Compatibility

In general both the 2.0 compiler and the 2.2 compiler are source level compatible with the v1.1 compiler. The following are the known exceptions:

* If you developed for pSOS 2.1 you will find that the `InstallDriver` function now has an extra parameter the `slow_entry` function has been renamed `_psos_slow_entry`, there is some new initialization code in `sysinit.c` that must be ported, and several new defines have been added to the `sysconf.h` file. For more details refer to the migration guide in Book 1 of the documentation set.
* The SDE v2.0 Final/v2.1/v2.2 compilers do not support pragma statements (e.g. `pragma TCS_break_dtree`), only preprocessor #pragma directives (e.g. #pragma `TCS_break_dtree`).
* The SDE v2.0 Final, v2.1 and v2.2 compilers use natural alignment by default in all cases. In some cases the SDE v1.1 Final compiler would use four byte alignment for external char * or structs. If you are casting to a type that requires stronger than default natural alignment you should investigate the `TCS_align pragma` or `-Xalign` compiler option.
* The pSOS sys_conf.h file has changed. Unused fields have been removed, new fields have been added.
* With the advent of component manager, board support packages have changed. If you have written your own BSP, you may have to do slight modifications for 2.2.
* As described in What's New? and the Migration Guide many of the applications interfaces have been expanded and slightly modified. e.g. the redefinition of *in-place component*.
* The old Win95 tmman interface has now been removed. The tmman interface for Win95 and WinNT is now consistent. If you used the old tmman interface for Win95, you will have to change your source code.
* Due to host communication changes between SDE v1.1 Final and SDE v2.0 Final, developers who used the WinNT tmman interface are recommended to recompile their source. No code changes should be required.

Object Level Compatibility

* Due to a calling convention inconsistency, all SDE v2.0 alpha compiled code must be recompiled with SDE v2.0 Final or later.

* While SDE v2.2 is backward compatible with all releases back to SDE v2.0 beta, some bugs were fixed in 2.0 Final that helped the debugger's ability to do stack traces. You may wish to recompile your code with 2.2 if you are trying to debug with **tmdbg**.
* If you compiled with -host Win95 or -host WinNT and did not use the **tmman** interface, your code will need to be relinked due to C library changes.

Module Level Compatibility

Modules compiled -host nohost which are source level compatible should be completely binary compatible.

A bug was found in the DLL compatibility mechanism. All DLLs will need to be relinked with SDE v2.0 Final or a later release.

SDE v2.2 Coexistence with other Releases

SDE v2.2 contains parts with the same names as SDE v2.1, SDE v2.0 and SDE v1.1 and therefore cannot reside in the same directory.

On Solaris, both products can be installed but only one can be in the PATH. You can move between releases by changing your PATH or always specifying the full pathname on your TCS calls.

Because of changes in the PC host execution tools, only one version of the device drivers can be installed on a given operating system instantiation. This means that only one version of the complete development environment can be active at a given time. You can still use the compiler from another version of the SDE by changing the path, but only one set of drivers may be installed.

Installation and System Requirements

Installation

The TriMedia SDE v2.2 release includes the TriMedia Compilation System (TCS) and the TriMedia Applications System (TAS). Additional application libraries are available as separate products from TriMedia Technologies. SDE v2.2 is shipped on two CD-ROMs.

Disc 1 of the TriMedia SDE v2.2 distribution contains a version of the TriMedia Software Development Environment for Solaris 2.6, 2.7 and 2.8. This disc contains the installation script `SETUPSOL`. The setup program is located in the SOLARIS directory under the root directory of Disc 1.

Disc 2 of the SDE v2.2 distribution contains versions for Windows 95 4.0 SR2 and Windows NT 4.0 as well as the Metrowerks CodeWarrior plug-ins for MacOS 7.0 or later, Windows 95, Windows 98 and Windows NT. The Installation program for Windows 95 and Windows NT is called `setup.exe` and is located in the `WINDOWS` directory under the root directory of the CD. The file `Metrowerks/ReleaseNotes.txt`, on Disc 2, contains information about the Metrowerks distribution and its installation.

Some of the newer TriMedia reference boards place the TriMedia behind an opaque PCI bridge. This is the case when DTV reference boards 3 and 4 (TM2700) are installed in a hosted system. When the TriMedia is behind such an opaque bridge, the normal install procedure will not work correctly, and you will have to use a manual procedure. This procedure is described in `WINDOWS\drivers\bridge\readme.txt` on Disc 2.

The SDE disk contains a sampling of supplementary libraries available from the TriMedia product group. These are installed in an `apps` directory by the installer. If you are also using the DTV library disk, or the Application library disk, it is common to install the libraries over the existing `apps` directory. The contents of the library disk always updates the `apps` directory on the SDE. This is the common usage. The DTV library disk must be installed after the SDE, installing the SDE after the DTV library disk may result in the loss of the DTV specific libraries.

Please refer to the *Getting Started* Book of the documentation and the README in the root directory for each platform for detailed installation instructions.

Supported Environments

Hardware Compatibility

SDE v2.2 is intended for use with TM-1S1.1 (that is, TM-1000 0.35 micron V1.1), TM-1100, or TM-1300 only. It is not guaranteed to perform as expected with earlier versions of TM-1000 hardware. The v2.2 software includes workarounds for bugs in earlier

hardware revisions, but some bugs which are fixed in TM-1S1.1 can lead to unpredictable program behavior on older chips.

Compilation/Debugging Hosts

The following are the supported operating system levels for compilation /debugging hosts and the associated version of SDE v2.2:

* Windows 95 (use Windows version of SDE v2.2 on Disc 2)
* Windows NT (use Windows version of SDE v2.2 on Disc 2)
* Solaris 2.6,2.7 or 2.8 (Use Solaris version of SDE v2.2 on Disc 1)

Execution Hosts

The v2.2 release supports the following execution hosts:

* Windows 95
* Windows NT 4.0
* Windows 98
* Windows 2000
* Windows CE 2.11

These hosts are collectively referred to as Windows Platforms throughout this document.

Board Support Packages

SDE v2.2 ships with the Board Support Packages for the following boards:

— IREF
— DTV-REF2
— DTV-REF3
— DTV-REF4
— DTV-REF5
— DTV-TTR
— DVR_REF1
— Debug-TM1000
— Debug-TM1100
— Debug-TM1300
— Debug-TM2700

System Requirements

All versions of the TriMedia SDE V2.2 release require at least 64MB of processor RAM.

Win95 and WinNT

The compact Windows installation requires about 225MB of disk space, not including documentation. A full installation including documentation requires 450MB of disk space. The typical Windows installation installs the entire Win95 or WinNT SDE, which allows host compilation and execution. Users can also custom install an execution-only environment, which allows execution on PC-hosted TriMedia hardware.

Solaris

The Solaris installations require about 200 MB of disk space, including documentation.

Metrowerks (MacOS, Win95 and WinNT)

The compact Metrowerks CodeWarrior TriMedia installation requires about 100MB of disk space, not including documentation. A full installation including documentation requires about 180 MB. Metrowerks CodeWarrior Professional (PowerPC for MacOS 7.0 or later, x86 for Win95/98/NT) Release 4, or later is required.

Other Included Software

The Solaris and Windows versions of the Install includes a copy of a searchable Adobe(tm) Acrobat(tm) 4.0.

Warning

Demonstration programs: TriMedia provides all example programs and sample applications for demonstration purposes only. All other usage is prohibited.

Support

For general information about TriMedia Products and services and questions about the SDE v2.2 release, please visit the TriMedia Technologies website:

`http://www.trimedia.com`

For version 2.2 support issues please email: `ttihelp@trimedia.com`

Known Problems, Workarounds and Other Issues

The following sections document the known problems in various tools and suggested workarounds, if any. You should also read the "Troubleshooting Guide" section of the Getting Started book.

Other Issues To Be Aware of

Application Libraries

* The following application library examples are not supported in big endian mode: exalVtransICP, exolVtransICP, exolAmixSimple, and exolArendAO.

Demonstration Programs

* The `icptest` example program no longer supports default values for the PCI video card parameters. The user must supply the PCI video display address, the display stride and the number of bytes per pixel explicitly using the command arguments `-d`, `-s` and `-b` respectively. The `-h` option displays the program parameter requirements.
* Demonstration program `post.out` is a Power On Self Test. It is designed to run after power on. It is not intended to be run after other demonstration programs and it may report spurious failures when run after other programs.

pSOS+

* The C++ front end **tmcfe** performs special processing of the function `main()` which results in the execution of static constructors when the program begins execution. Because the root task in a pSOS+ program is `root()` rather than `main()`, a pSOS+ root task written in C++ may not compile as expected and may not execute static constructors as expected at start-up. Workaround: add declarations `extern "C" void _main(void); and` add a call to `_main`() at the start of `root`(). Note that `_main()` and `root`() have C linkage. The user must also make sure that the link command which builds the executable includes the C++ library, for example by compiling with **tmCC** or **tmcpp** rather than with **tmcc**.

Interrupt Service Routines

* Using calls to exit or to `tmAssert` in interrupt service routines is not recommended. When used in conjunction with pSOS, they might cause the system to hang. `tmAssert` is perfectly safe if the assert condition is not triggered. Both functions work as expected outside ISRs.

* The PCI device library provides a function `pciMemoryCopy()` which can be used to copy memory between SDRAM and HOST(PC) even with the PCI memory window disabled. This function uses synchronous DMA calls that depend on the scheduler being active and hence cannot be called from interrupt handlers.

tmccom - Compiler

* The TCS_O4 and TCS_O5 pragmas are not supported unless the command line optimzation level was -O4 or -O5.

tmld - Linker

* The `-bcompact` linker optimization (on by default with **tmcc** unless you link with `-g`) may reorder dtrees at link time. If your code is order dependent, please link with the **tmcc** `-nocompact` option.

C++ Compiler and Library

* The C++ compiler is not completely at ANSI level yet. Please check the "Ongoing Standardization Issues" section in chapter 8 of the C Users Guide for a full list of the limitations of our current C++ implementation.
* Due to a C++ compiler restriction the Standard Template Library is missing some operator->() member functions in iterator classes. Please use (*iterator).function() instead of iterator->function() as a workaround.

tmdbg - Debugger

* Task mode can not be entered unless stopped at the beginning of the application (i.e. before or at the beginning of the `root`() function).
* The debugger cannot debug linker optimized code. This includes targets compiled with any of the following linker options: `-bcompact`, `-bremoveunusedcode`, or `-bfoldcode.`
* The debugger does not utilize source file/target time stamps. If a source file is modified after the target has been built, the debugger does not warn the user that the source file is out of date.
* The class browser does not display protection and some other C++ specific attributes.
* In C++ the debugging of unnamed types (including anonymous unions), unnamed enums, unnamed members and unnamed namespaces is not currently supported.
* There is no support in the debugger to dynamically cast a variable to or from a complex type (struct, union, class) from the debugger command line while you are debugging.
* The debugger cannot debug header files.

tmprof - Program Profiler

* Profiling introduces an overhead which may affect the execution time of the program due to increased cache usage. The accuracy also depends on statistical significance.

tmsim - Simulator

* **tmsim** cycle counts do not take memory refresh into account (see *TM-1000 Data Book*).
* **tmsim** uses the underlying operating system to implement system calls. Differences in the implementation of system calls in the underlying operating systems (HP-UX vs. SunOS vs. Win95 vs. WinNT) and differences in the underlying hardware (e.g. blocksize) can cause small differences in reported simulated cycles on different execution host platforms.
* **tmsim** uses the floating point of the host machine to simulate TM-1000 floating point operations. In a few special cases, the behavior of floating point operations involving NaNs and denormals under **tmsim** may differ from the behavior on TM-1000 hardware or on **tmsim** on a different host platform.
* **tmsim** simulation of JTAG, PCI, SSI and VLD peripherals requires the `-mm` option because the peripherals will not work correctly with the `-nomm` option. In general, simulation of all peripheral blocks requires the `-mm` option. The `-mm` option is true by default.
* **tmsim** simulation of VLD differs slightly from VLD hardware operation. Several counters (including but not limited to the DMA output counters) count differently in TM hardware operation than in **tmsim** simulation.

tmman - Execution Hosts

* On systems that have a secondary PCI bus, if the TriMedia device and a PCI VGA card co-exist on the same (non primary) bus, then data read and written by the TriMedia device from main host memory may be corrupted. This is due to a bug in the PCI bridge chip that causes memory reads/writes to addresses whose last 10 bits are the same as VGA ISA I/O Port addresses (0x3b0-0x3bb) to be blocked from going to main memory. To workaround this problem, place the TriMedia device and the VGA card on different busses; or, if they must be on the same bus, put both cards on the primary PCI bus (Bus #0).
* If the WDM driver (tmman.sys) is installed on a system running Windows98, and if the file tmman.vxd is present in the Windows\System directory, then running any Host Tools will cause a PC crash. To work around this problem, delete or rename the tmman.vxd file from the Windows\System directory.
* Currently the TriMedia device drivers support up to 32 TriMedia devices in a system under Windows platforms. However, due to Windows virtual memory restrictions, SDRAM mapping may need to be disabled to ensure that all the devices can be handled

by the driver. Note that if SDRAM Mapping is disabled system calls made from the Trimedia processor in a hosted environment are not supported.

* Configuration settings for the ref3 card have to be specified in the windows registry for `SystemBaseAddress` (default 0x10000000), `SDRAMBaseAddress` (default 0x00000000), and `MMIOBaseAddress` (default 0xef000000). They must be defined as DWORD values under `HKEY_LOCAL_MACHINE\SOFTWARE\TrimediaTechnologies\TMMan\DeviceX` where X is 0, 1, 2 ...31 depending on the order of card detection. For example, in the case of a system with a single ref3 board, the values must be defined under `HKEY_LOCAL_MACHINE\SOFTWARE\TriMediaTechnologies\TMMan\Device0`. If these registry values are not used, the default value, mentioned above, will be used.