MULTI® Integrated Development Environment

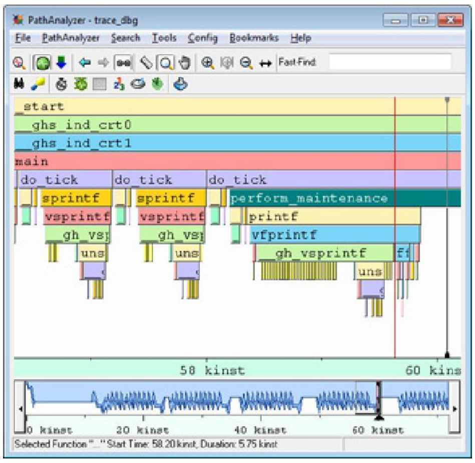
- Find and fix bugs faster
- Make sense of complex systems
- Boost productivity
- Debug embedded Linux
- Maximize performance
- Benefit from an expert
- Lower your production cost
- Functional safety confidence

After more than three decades of customer use and continuous enhancement, the MULTI IDE is unmatched in the embedded software industry. Developers know they can rely on MULTI to help them produce high-quality code and get their devices to market faster. Whether pinpointing a hard-to-find bug, resolving a memory leak, or maximizing system performance—MULTI consistently works.

» Download MULTI datasheet (PDF)
» Download What's New in MULTI 7 datasheet (PDF)

Find and fix bugs faster

Every feature of our revolutionary **Debugger** is designed to quickly solve problems that stump traditional tools. It often takes weeks or months to track down problems like inter-task corruptions, missed real-time requirements, and external hardware events. Green Hills' **TimeMachine tool suite** helps you solve the same problems in hours or even minutes. The TimeMachine tool suite automatically captures program execution data, combining the MULTI Debugger interface with innovative replay debugging capabilities. When debugging with TimeMachine, you can step, run, and debug forwards and backwards through code to pinpoint the root cause of the problem.



An invaluable part of the TimeMachine tool suite, the PathAnalyzer provides a view of your application's call stack over time. Click for larger view

At any point in the TimeMachine capture you can:

- Debug task interactions at the source level
- Find specific data points by using powerful execution and data breakpoints
- Examine RTOS data structures
- Debug virtual memory applications

DoubleCheck, Green Hills' integrated static analyzer, saves you time and money by allowing you to identify programming errors even before running your program. Because DoubleCheck is part of the Green Hills Compilers, there's no setup cost to enabling it. Simply set an option and DoubleCheck reviews source code each time it's compiled. Additionally, DoubleCheck runs much faster than traditional static analysis tools, which means that every developer can have it on all the time.

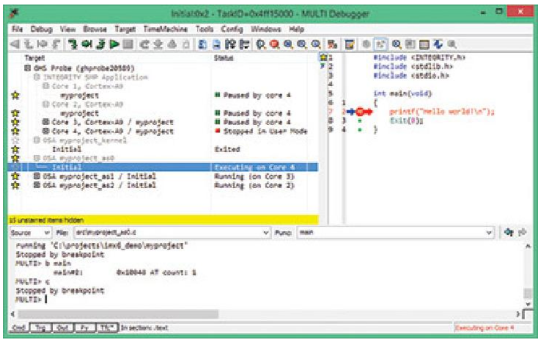
Run-time error checking complements DoubleCheck by finding bugs that cannot be identified by static analysis alone. Especially heinous bugs remain silent for a long time before failure. With run-time error checking, you're alerted to the cause of the problem when it happens. Fixing a bug is trivial when you're looking directly at it.

top

Make sense of complex systems

Debugging complex systems becomes a much simpler task when the information needed to understand system status is at your fingertips. We present important system information in intuitive ways to help you find problems faster:

- The **OSA Explorer** provides comprehensive views into the status of kernel objects, tasks, and resources.
- The **Debugger** allows complete postmortem application debugging that includes traversing the stack, browsing objects, and viewing memory and registers.
- The **EventAnalyzer** shows a timeline of system events occurring at intervals only microseconds apart.


Initial0x2 - TaskID=0x4ff15000 - MULTI Debugger
File Debug View Browse Target TimeMachine Tools Config Windows Help
Source
File:
Func:
Cmd Trg Out Py TK*

Target	Status
GHS Probe (ghprobe20580)	
INTEGRITY SMP Application	
Core 1, Cortex-A9	
myproject	Paused by core 4
Core 2, Cortex-A9	
myproject	Paused by core 4
Core 3, Cortex-A9 / myproject	Paused by core 4
Core 4, Cortex-A9 / myproject	Stopped in User Mode
OSA myproject_kernel	
Initial	Exited
OSA myproject_as0	
Initial	Executing on Core 4
OSA myproject_as1 / Initial	Running (on Core 3)
OSA myproject_as2 / Initial	Running (on Core 2)

15 unstarred items hidden

File: src\myproject_as0.c Func: main

```
#include <INTEGRITY.h>
#include <stdlib.h>
#include <stdio.h>

int main(void)
{
    printf("Hello world!\n");
    Exit(0);
}
```

```
running 'C:\projects\imx6_demo\myproject'
Stopped by breakpoint
MULTI> b main
    main#2:        0x10048 AT count: 1
MULTI> c
Stopped by breakpoint
MULTI> |
```

In sections .text   Executing on Core 4

MULTI's synchronous run control halts all cores in a multicore system simultaneously. This enables you to debug one core without worrying that operations running on another core will affect shared memory. Click here for a larger view.

Synchronous run control for multicore

Green Hills Software was one of the first companies to offer a probe and debugger solution for multicore debugging. MULTI now offers features that make debugging multicore systems as straightforward as single core systems.

Key among these features is synchronous run control, which halts all cores as a unit when any core encounters a debugging condition. For instance, when a core hits a breakpoint the target list clearly shows:

- which core(s) have hit the breakpoint
- what thread was executing on the core at the time
- what the other cores in the system were doing

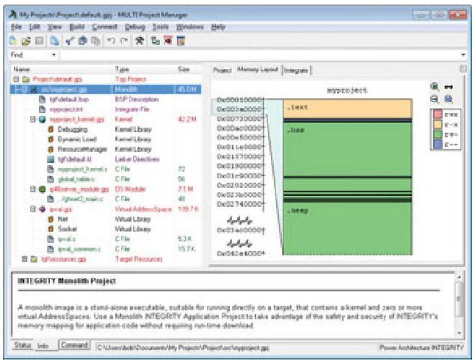
Downloading code to a multicore target can be a complex and tedious process. MULTI can automatically handle downloading INTEGRITY to these targets. For other types of operating systems and applications a simple configuration file can be used to automate the download and setup process.

top

Boost productivity

Clean code is less likely to contain errors and is easier to test, understand, and modify—all factors that contribute to fewer bugs and greater reliability. The Green Hills Compilers enable enforcement of clean coding conventions via industry standards such as **MISRA 2004** as well as **GHS Standard Mode**, which is a collection of Compiler warnings and errors that enforces a stricter coding standard than regular C and C++ . GHS Standard mode is based on our long history of successfully deployed software and on our collaboration with customers who have achieved the same high level of reliability.

We've simplified building so that you can spend more of your valuable time designing your next product. With the MULTI **Builder**, there's no complicated build infrastructure to maintain. You can change your build configuration easily. The Builder automatically analyzes the dependencies of your entire project and compiles and links as many files in parallel as possible, taking full advantage of modern multicore systems.



Name	Type	Size
Project\default.gpj	Top Project	
src\myproject.gpj	Monolith	45.0M
tgt\default.bsp	BSP Description	
myproject.int	Integrate File	
myproject_kernel.gpj	Kernel	42.2M
Debugging	Kernel Library	
Dynamic Load	Kernel Library	
ResourceManager	Kernel Library	
tgt\default.ld	Linker Directives	
myproject_kernel.c	C File	72
global_table.c	C File	56
ip4server_module.gpj	OS Module	7.1M
../ghnet2_main.c	C File	48
ipval.gpj	Virtual AddressSpace	105.7K
Net	Virtual Library	
Socket	Virtual Library	
ipval.c	C File	5.3K
ipval_common.c	C File	15.7K
tgt\resources.gpj	Target Resources	

INTEGRITY Monolith Project

A monolith image is a stand-alone executable, suitable for running directly on a target, that contains a kernel and zero or more virtual AddressSpaces. Use a Monolith INTEGRITY Application Project to take advantage of the safety and security of INTEGRITY's memory mapping for application code without requiring run-time download.

C:\Users\bob\Documents\My Projects\Project\src\myproject.gpj

The MULTI Project Manager simplifies how you build and update your application. Click here for a larger view.

The Builder and other sophisticated features significantly reduce the overhead of project development. A seamlessly integrated project manager, editor, flash programmer, instruction set simulator, and more enable you to jumpstart development and work more efficiently.

top

Debug embedded Linux

For engineers developing software for embedded Linux systems, MULTI for Linux brings the comprehensive debugging capabilities needed to dramatically improve productivity and bring a more reliable product to market faster.

Traditionally, Linux engineers had to rely on debugging tools that offered only limited visibility and usefulness. GDB—and its related front-ends such as Eclipse—are cumbersome to set up, difficult to use, and not always trustworthy. As a result, developers often resort to the hit-and-miss effectiveness **printf** or **printk** commands. Unfortunately, these provide only a very small window into complex system execution.

With MULTI for Linux, embedded engineers can now visualize and debug all aspects of an embedded Linux system—including the Linux kernel, kernel threads, interrupt services routines, as well as user mode threads and processes—faster and more easily than ever before.

Linux debugging capabilities

- Comprehensive Linux kernel visualization—including shared memory regions, semaphores, message queues, and kernel modules
- Support for SMP multicore debugging
- Debug from kernel through user space with one debugger
- Debug native Linux and virtualized Linux (on INTEGRITY
- Multivisor virtualization technology)
- Debug loadable kernel modules and device drivers
- Easily debug process startup code
- Set and thread-specific breakpoints in kernel code
- Debug shared objects in virtual address spaces
- Handle breakpoints with minimum performance cost
- Debug multi-threaded interactions across processes

top

Maximize performance

Generate faster, smaller code

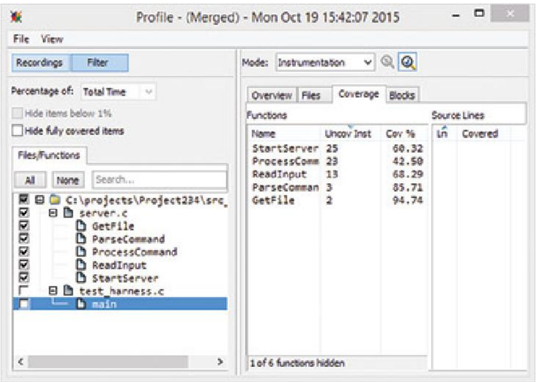
Green Hills optimizing compilers are the best on the market. On EEMBC benchmarks—the most widely accepted benchmarks in the embedded industry—the Green Hills Compilers consistently outperform competing compilers to generate the fastest and smallest code for 32- and 64-bit processors.

The Green Hills Compilers use hundreds of the most advanced optimizations to significantly increase program execution speed and decrease size. For example, advanced whole-program optimizations span function call boundaries to allow the Green Hills Compilers to operate on entire source bases. As a result, the Compilers are able to eliminate unnecessary code and variables, inline across modules, and better analyze interprocedural aliases. You can fine-tune Compiler output by enabling different combinations of optimizations and by configuring optimizations on a per-project, per-file, per-function, or even per-source-line basis. On most programs, the Green Hills Compilers offer a 20% improvement in speed over the GNU Compiler.

Pinpoint performance bottlenecks

Using Green Hills' exceptional PathAnalyzer, Profiler, and EventAnalyzer, you can easily identify and eliminate performance bottlenecks in your code. The result is a faster, more streamlined application:

- The truly innovative **PathAnalyzer** provides a high-level, time-based view of every function executed in your program, making it easy to identify patterns in program flow and see where time is being spent. Instead of showing you only a shadow of your program's execution, the PathAnalyzer provides the entire picture.
- For system optimization and analyses, the powerful **Profiler** provides unrivaled insight into program behavior. The profiler shows how much processor time is consumed by each task, function, source line, and assembly language instruction, making it easy to:
    - find performance problems
    - find coverage gaps and track test coverage of specific code
    - share profile data with other uses
    - collect and merge profile data from several tests or profile recordings
- The **EventAnalyzer** displays the length and frequency of INTEGRITY RTOS events, making it quickly apparent what operations take the most time and where optimization efforts should be focused.



Mode: Instrumentation

Percentage of: Total Time

□ Hide items below 1%

□ Hide fully covered items

☑ C:\projects\Project234\src_
☑ server.c
☑ GetFile
☑ ParseCommand
☑ ProcessCommand
☑ ReadInput
☑ StartServer
□ test_harness.c
□ main

Name	Uncov Inst	Cov %
StartServer	25	60.32
ProcessComm	23	42.50
ReadInput	13	68.29
ParseComman	3	85.71
GetFile	2	94.74

Ln	Covered

1 of 6 functions hidden

The Profiler's interface streamlines analyses, showing how often and for how long different tasks, functions, instructions, etc were called so you can pinpoint where to target optimization and testing efforts. *Click here for a larger view.*

top

Benefit from an expert

Green Hills software engineers use MULTI and our compilers every day. Our team of productivity consultants can draw on this firsthand experience to ensure that you know how to realize the full potential of your Green Hills products as well. They can

- configure our tools to meet your specific requirements
- train you in the best use of our products
- evalutate your use and identify ways in which you can work more productively

And to obtain the highest possible performance, you can also turn your program over to our performance-tuning experts. For three decades, we've been using our in-depth knowledge and best-in-class tools to beat every competitor on published benchmarks for program speed. We can apply the same techniques that we use internally to make your application run faster. We use our comprehensive knowledge to select optimal compiler optimizations for each part of your program.

Our performance-tuning service is not a commodity, it's a custom service. We'll custom-write compiler optimizations based on our evaluation of your application and use our expertise in embedded systems to make your program run faster. It's not uncommon for our customers to see 300% or more improvement after working with us.

top

Lower your production cost

Reduce your processor costs

You don't always need a faster processor to meet increasing performance requirements. Our superior compiler optimizations are tuned to get the most out of whatever chip you use. Additionally, our performance analysis tools such as the PathAnalyzer, Profiler, and EventAnalyzer give you a bird's-eye view of where your program is spending its time, helping you to focus your optimization efforts and speed up your application. How much money would you save if you could use a lower-cost processor to meet the same real-time requirements? Or if you could release a new product without having to produce a new hardware platform to run it on?

Not only will improving your application's performance save you money on the processor itself, lower-cost processors operate at lower clock rates and require less power and cooling. Longer battery life makes your product better and cheaper at the same time.

Reduce your memory costs

Green Hills Compiler optimizations reduce your memory costs by reducing the size of your executable. Most programs see at least 10%

MULTI Support

Green Hills Software is continually adding support for new processors and operating systems. If yours is not listed here, please contact Green Hills Software for the latest availability information.

Target processors

- ARM/XScale
- Power Architecture
- Intel Architecture
- ColdFire
- 68K
- MIPS
- RH850 & V850
- Tricore

Development hosts

- Windows
- Linux

Operating systems

- INTEGRITY RTOS
- *µ-velOSity* RTOS
- Windows
- Linux
- ThreadX
- VxWorks
- OSE
- Bare board
- In-house

improvement relative to the GNU Compiler. Additionally, DoubleCheck finds memory leaks at compile time, and MULTI's Memory Allocations tool searches for them dynamically at run time.

Maximize savings
Our team of cost reduction experts can help you to reduce your memory and processor requirements without sacrificing features. We have reduced customer memory usage by over 30% and have improved performance by over 300%.

- Solaris

Third-party integrations

- Eclipse
- Rhapsody
- Emacs
- vi
- Subversion
- ClearCase
- CVS
- SourceSafe
- DWARF/Stabs compilers

Languages

- C
- Ada (w/AdaMULTI)
- C++
- EC++

top

Functional safety confidence

The MULTI toolchain has been certified to meet the highest levels of tool qualification and C/C++ runtime certification:

- IEC 61508:2010 (Industrial), EN 50128:2011 (Railway) and ISO 26262:2011 (Automotive) functional safety standards
- Certificates received from both TÜV NORD and exida
- Satisfies both SIL 4 (Safety Integrity Level) and ASIL D (Automotive Safety Integrity Level) tool qualification and C/C++ runtime certification requirements—the highest levels achievable
- Supported on a broad set of target processors

The benefits to customers are simple and profound:

- lower cost and time-to-certification
- reduced product time-to-market
- reduced certification maintenance after product release
- application development support at highest SIL/ASIL levels

top