

Data Race Detection on GNU* Project Debugger

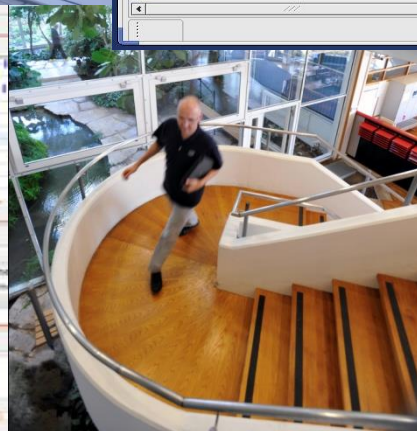
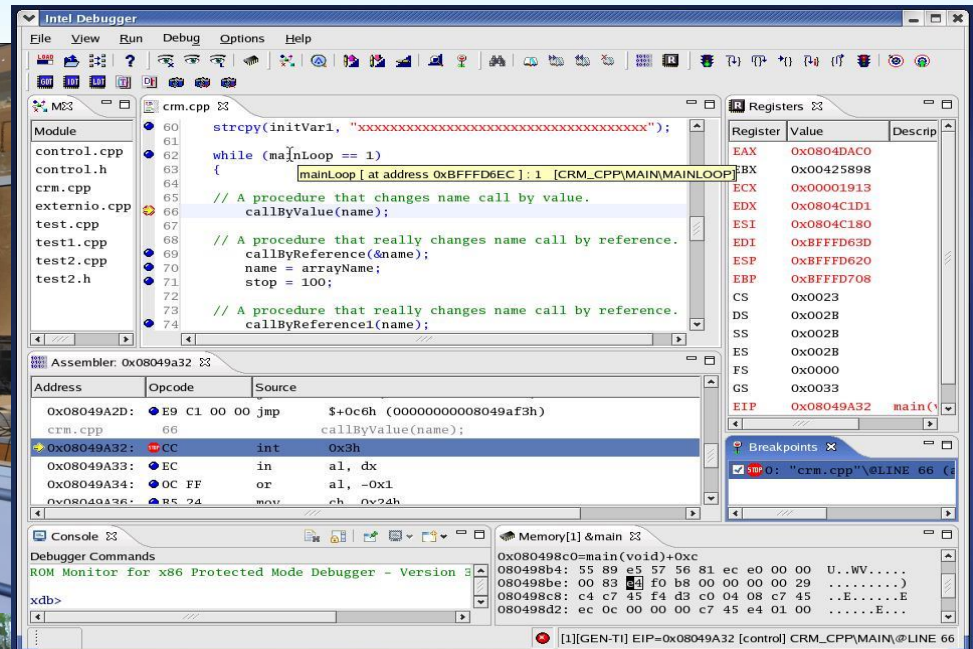
Dr. Walfred Tedeschi

Code the Future

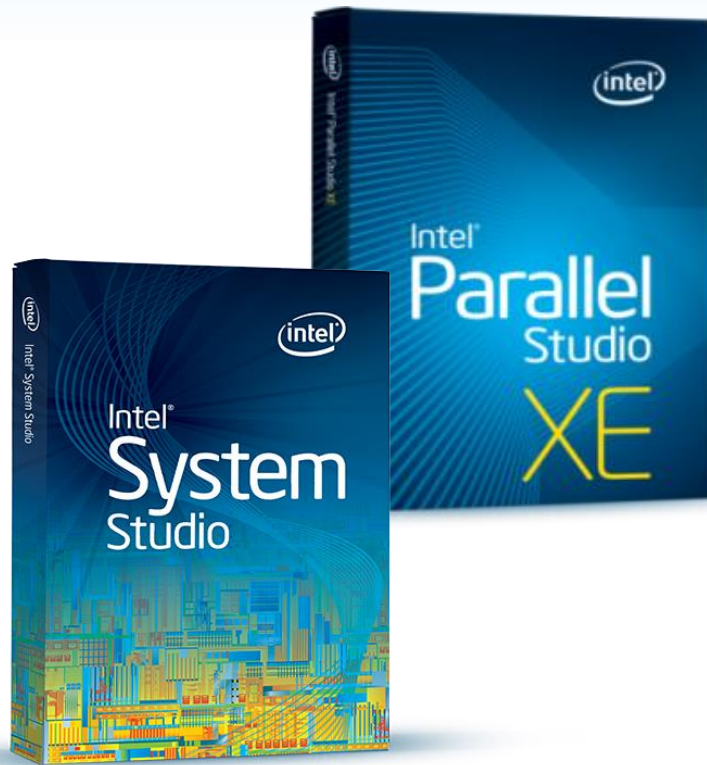


Intel GmbH - Ulm Site

- The Competence Center for Debugging Solutions



Projects



- **Owner of Intel® System Studio** software development tools for embedded IA
- **Development of Debuggers** - including GDB - for all Intel development products like **Intel® Parallel Studio**
- Unique Value:
The **Ulm** site is the world wide **Intel Competence Center for Debugging on IA**

Code the Future

Introduction: GDB

Features added / worked on by Intel:

- Data Race Detection
- Intel® Many integrated Core Architecture
- Intel® Memory Protection Extensions
- Intel® Advanced Vector Extensions 512
- Fortran support like Variable Length Array
- Pointer Checker

Code the Future

Agenda

- Data Race Conditions
- Usage
- Configuration Options
- Filter (Set) Options
- Overhead Management
- Selective Enabling
- Summary

Code the Future

Data Race Conditions

$x = 0$

$a = 1$

$b = 2$

Thread 1

Thread 2

$x = a + b$

$b = 42$

$x = 3 \text{ or } 43 ?$

Execution order is not guaranteed unless synchronization constructs are used

Code the Future

Data Race Conditions

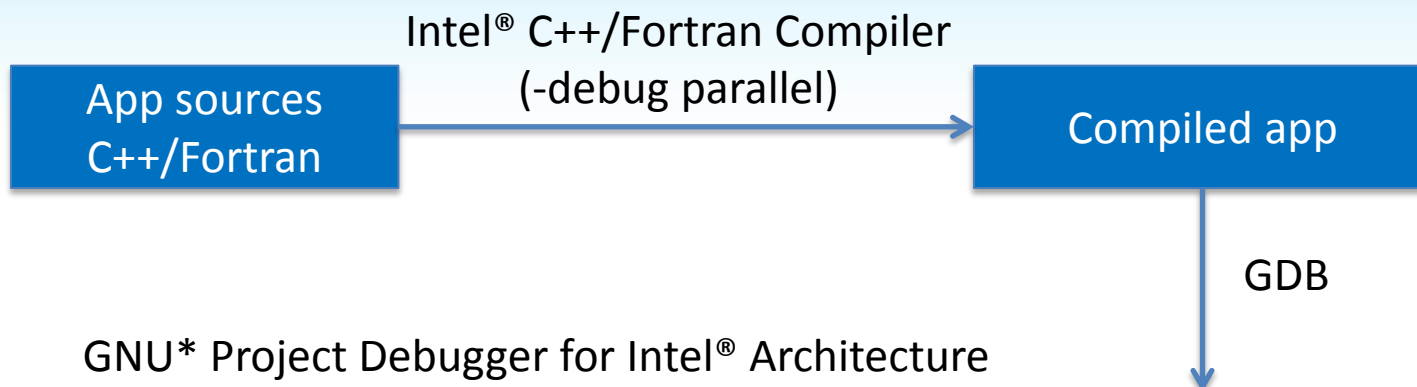
Usual Symptoms:

- Random clobbered results
- Unpredictable lost updates
- Sporadic memory corruption

How to debug this kind of random behavior?

Code the Future

Usage



```
(gdb) pdbx enable
(gdb) run
data race detected
1: write b, 4 bytes from foo.c:36
3: read b, 4 bytes from foo.c:40
Breakpoint, 0x401515 in foo () at foo.c:36
36 b = 42;
(gdb)
```

Code the Future

Usage

Pre requisites

- Intel® C++ or Fortran Compiler
 - Compile with *-debug parallel* option
 - Link with data race analyzer library
- Supported threading models: POSIX* threads and OpenMP*
- Supported processors: Intel® Core™, Xeon®, Atom™ and Xeon Phi™
- Python* 2.6+

Workflow

- Start application in Intel® GDB
- Configure data race analysis
- Debugger breaks in the context of a data race

Code the Future

Usage

```
$ icc -g -debug parallel -o foo foo.c
```

Code the Future

Usage

```
$ icc -g -debug parallel -o foo foo.c
```

```
$ gdb foo
```

```
GNU gdb (GDB) ...
```

```
Reading symbols from foo...done.
```

```
(gdb) pdbx enable
```

Code the Future

Usage

```
$ icc -g -debug parallel -o foo foo.c
```

```
$ gdb foo
```

```
GNU gdb (GDB) ...
```

```
Reading symbols from foo...done.
```

```
(gdb) pdbx enable
```

```
(gdb) run
```

```
data race detected
```

```
1: write b, 4 bytes from foo.c:36
```

```
3: read b, 4 bytes from foo.c:40
```

```
Breakpoint -11, 0x401515 in foo () at foo.c:36
```

```
36 b = 42;
```


Configuration Options

- Fine-tune the data race analysis through Filters
 - Ignore specified regions (suppress)
 - Consider only specified regions (focus)
 - Ignore all read accesses
- Filters can be added/modified at any time
- Setting filters similar to breakpoints/watchpoints
- Filters are organized in filter sets

Code the Future

Filter Operations

- `pdbx filter line foo.c:36` # Line
 - `pdbx filter code 0x40518-0x40524` # Code range
 - `pdbx filter var answer` # Variable
 - `pdbx filter data 0x60f48-0x60f50` # Data range
 - `pdbx filter reads` # All reads
-
- Filter defines code or data region in debuggee
 - Filter expressions evaluated when filter is set

Code the Future

Filter Set Operations

- `pdbx fset new my_fset` # set creation
- `pdbx fset suppress/focus` # filter meaning

- `pdbx fset list` # list filter sets
- `pdbx fset show` # list filters in set
- `pdbx fset select other_fset` # switch between sets

- `pdbx fset remove 4` # remove filter number 4
- `pdbx fset disable 2-6` # disable filters from 2 to 6
- `pdbx fset enable 8` # enable 8th filter
- `pdbx fset evaluate` # re-evaluate filter expr

- `pdbx fset import other:3-8` # import

Code the Future

Example

```
(gdb) pdbx enable
```

```
(gdb) run
```

```
data race detected
```

```
1: write b, 4 bytes from foo.c:36
```

```
3: read b, 4 bytes from foo.c:40
```

```
Breakpoint -11, 0x401515 in foo () at foo.c:36
```

```
36 b = 42;
```

```
(gdb)
```


Example

```
(gdb) pdbx enable
(gdb) run
data race detected
1: write b, 4 bytes from foo.c:36
3: read b, 4 bytes from foo.c:40
Breakpoint -11, 0x401515 in foo () at foo.c:36
36 b = 42;
(gdb) pdbx filter var b
(gdb) continue
Program exited normally.
```

Code the Future

Overhead Management

- Manage overhead effectively through
 - Focus filter sets
 - Global filter on reads
 - Selective enabling
- Allows fast and focused debugging of data race bugs
- Allows data race detection under resource constraints

Interactive data race debugging

Code the Future

Selective Enabling

```
(gdb) pdbx disable
```

```
(gdb) pdbx reset # discards memory access logs
```

```
(gdb) break foo
```

```
Breakpoint 1 at 0x4006e8: file foo.c, line 30
```

```
(gdb) continue
```

```
Breakpoint 1, foo () at foo.c:30
```

```
(gdb) pdbx enable
```

Selective Enabling

```
(gdb) continue
```

```
data race detected
```

```
1: write b, 4 bytes from foo.c:36
```

```
3: read b, 4 bytes from foo.c:40
```

```
Breakpoint -11, 0x401515 in foo () at foo.c:36
```

```
36 b = 42;
```

Code the Future

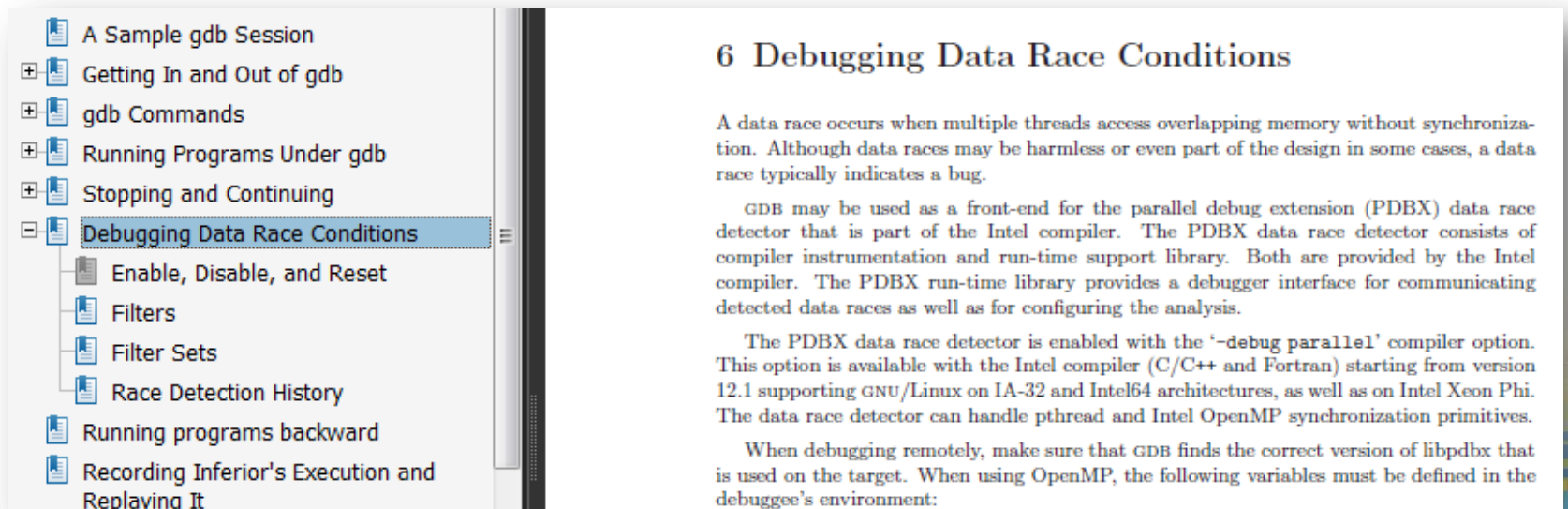
Summary

- Interactive debugging of data races in GDB
- Configurable analysis scope to manage performance
- Use well known GDB syntax

Code the Future

Learn More

- Learn more: [GDB - The GNU* Project Debugger for Intel® Architecture](#)
- The GNU* Project Debugger for Intel® Architecture Documentation:



Code the Future

Where to find (Linux)

For Intel® Atom™:

[Intel® System Studio](#)

For Intel® Core™, Atom™ and Xeon Phi™:

[Intel® Composer XE 2013 SP1](#)

[Intel® Parallel Studio XE 2013 SP1](#)

Code the Future

Thank You

Code the Future



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm>

Any software source code reprinted in this document is furnished under a software license and may only be used or copied in accordance with the terms of that license. <http://software.intel.com/en-us/articles/intel-sample-source-code-license-agreement>

This document contains information on products in the design phase of development.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804