Intel® System Debugger 2018
Intel® Debug Extensions for WinDbg* 2018
Installation Guide and Release Notes

Installation Guide and Release Notes for Windows* host

26 February 2018

Contents

1 Introduction

The Intel® Debug Extensions for WinDbg* 2018 is a component of the Intel® System Debugger 2018 and is an add-on to the Microsoft WinDbg* debugger to support Intel® Processor Trace. The extension allows for easy setup of Intel® Processor Trace (Intel® PT) by abstracting hardware configuration and then reconstructing and displaying execution flow from the collected trace data. It integrates with other WinDbg* features like symbolization and high-level source display.

Intel® Processor Trace is a new technology for low-overhead execution tracing. It facilitates debugging a program by exposing an accurate and detailed trace of the program's activity, and its triggering and filtering capabilities help identifying and isolating the relevant program executions.

1.1 Technical Support and Documentation

The default installation directory of the Intel® System Studio including the Intel® System Debugger, in the following also called `<INSTALLDIR>`, is

```
C:\Program Files (x86)\IntelSWTools\
```

This installation directory can be specified by the user with a 'Custom' installation. The product however is being installed in a fixed directory structure below `<INSTALLDIR>`.

All information about the Intel® Debug Extensions for WinDbg* can be found under `<INSTALLDIR>\documentation_2018\en\debugger\system_studio_2018\windbg-ext\`

If you did not register your System Studio license yet, please do so at the Intel® Software Development Products Registration Center. Registration entitles you to free technical support, product updates and upgrades for the duration of the support term.

To submit issues related to this product please visit the Online Service Center webpage and submit issues under the product **Intel® System Studio**.

Additionally you may submit questions and browse issues in the Intel® System Studio User Forum.

For information about how to find Technical Support, product documentation and samples, please visit http://software.intel.com/en-us/intel-system-studio.

Note: If your distributor provides technical support for this product, please contact them for support rather than Intel.

1.2 Product Contents

- Intel® System Debugger 2018 including
- Intel® Debug Extensions for WinDbg* for Intel® Processor Trace

2 What's New for the Intel® Debug Extension for WinDbg* 2018 Update 1

- Support event-based breakpoints to debug ACPI Machine Language (AML)
- Collect BSOD information with the get_bsod_info script

3 System Requirements

3.1 Host System Requirements

Microsoft* Windows* 7, 8 and 10 64-bit host system.
Windows* Driver Kit* (WDK) 10*, available under the download link:
https://msdn.microsoft.com/en-us/windows/hardware/dn913721.aspx

3.2 Target System Requirements

- The Intel® Debug Extensions for WinDbg* for Intel® Processor Trace feature requires a 6th generation Intel® Core Processor (Skylake)
- The Intel® Debug Extensions for WinDbg* for IA JTAG debugging feature is tested on 6th generation Intel® Core Processor (Skylake) and Intel® Atom™ Processor Z36xx, Z37xx - 2 cores (Baytrail / MinnowBoard MAX) (see chapter 5 for details). The feature should work on any supported platform running windows but these are the platforms where functionality is verified.
- The Intel® Debug Extensions for WinDbg* for IA JTAG debugging feature is supported on Intel® Atom™ Processors N4200, N3350, x7-E3950, x5-39xx (Apollo Lake), Intel® Xeon™ Scalable Processor (Skylake-SP) / Intel C620 Series Chipset (Lewisburg) and 7th generation Intel® Core™ Processor as well.

4 Installation

4.1 Pre-Installation Steps

Intel® Debug Extensions for WinDbg* for IA JTAG debugging feature requires the Microsoft Windows* Software Development Kit* - Windows 10* to operate properly (see "System Requirements”).

4.2 Product Installation

The Intel® Debug Extensions for WinDbg* 2018 on Windows* host is included in the Intel® System Studio 2018 package.

Double-click on the executable file `system_studio_2018.0.xxx_windows_target.exe` or `system_studio_2018.0.xxx_windows_target_online.exe` to begin installation and follow the on-screen instructions.

The default installation directory of the WinDbg Extension component is:

```
<INSTALLDIR>\system_debugger_2018\windbg-ext\
```

5 Some Usages of Intel® Debug Extensions for WinDbg*

5.1 Event-based debugging using breakpoints

Intel® Debug Extensions for WinDbg* doesn't have an agent-based solution and cannot be notified from OS Windows that messages got sent. In order to capture this omission, we introduce the event-based debugging using breakpoints. The user could set conditional breakpoints on kernel function and print required fields using the command line like: ***bp ACPI!ACPIInterruptServiceRoutine ".echo ACPI Interrupt!; g"***.

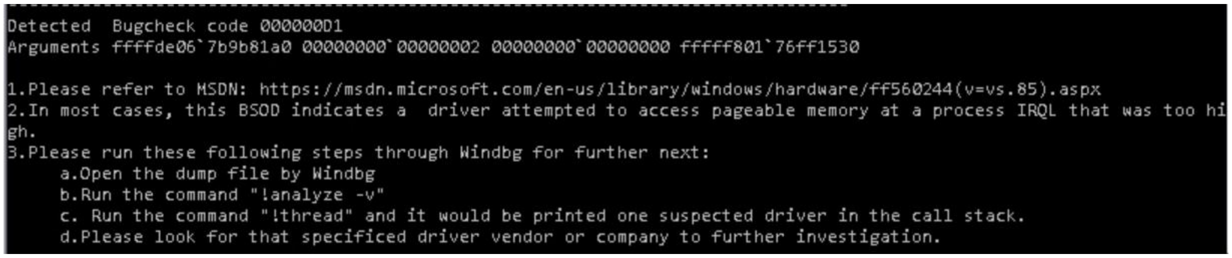
- ***bp ACPI!ACPIInterruptServiceRoutine*** is for setting a breakpoint at the ACPI interrupt function and the process will be paused when the function is invoked. This function is for dealing with the ACPI interrupts. That's why we call it as the event-base debugging.
- ***".echo ACPI Interrupt!; g"*** combines two actions which will be executed when the process breaks at the function ACPIInterruptServiceRoutine(). The command "*.echo ACPI Interrupt!*" to output a message "ACPI Interrupt!" in the kernel debug view on WinDbg* and the command *".echo ACPI Interrupt!*" and the command "*g*" is to make the process go back running.

We can also get the raw data of registers by the command like: ***bp nt!DbgPrintEx "da r8; g"***.

- ***bp nt!DbgPrintEx*** is for setting a breakpoint to halt the CPU when the Windows function DbgPrintEx() is invoked.
- ***"da r8; g"*** is also for displaying the content of register *r8* by *da* when the nt!DbgPrintEx gets hit

5.2 Collecting BSOD information by get_bsod_info

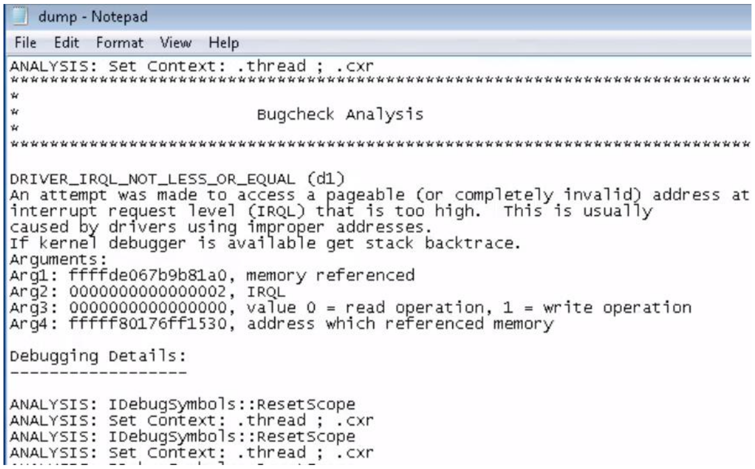
A smart script get_bsod_info is developed for helping users to collect useful data based on what error code it gets from Windows BSOD. When BSOD happens, the user just need to launch Intel® Debug Extensions for WinDbg* and type in the command ***forensic.get_bsod_info(r“<filename>”)#***. Then the script will help to do the corresponding commands and output the data into the designated file for analysis. For example:



```
--------------------------------------------------------------------------------
Detected  Bugcheck code 000000D1
Arguments ffffde06`7b9b81a0 00000000`00000002 00000000`00000000 fffff801`76ff1530

1.Please refer to MSDN: https://msdn.microsoft.com/en-us/library/windows/hardware/ff560244(v=vs.85).aspx
2.In most cases, this BSOD indicates a  driver attempted to access pageable memory at a process IRQL that was too hi
gh.
3.Please run these following steps through Windbg for further next:
     a.Open the dump file by Windbg
     b.Run the command "!analyze -v"
     c. Run the command "!thread" and it would be printed one suspected driver in the call stack.
     d.Please look for that specificed driver vendor or company to further investigation.
```

In this case, the script detects the error code belong to DRIVER_IRQL_NOT_LESS_OR_EQUAL (bug check 0xD1). So, it helps to run the commands "!analyze -v" and "!thread" for collecting necessary data. Once it's done, the user can read the log file for analysis like



```
ANALYSIS: Set Context: .thread ; .cxr
***************************************************************************
*
*                               Bugcheck Analysis
*
***************************************************************************

DRIVER_IRQL_NOT_LESS_OR_EQUAL (d1)
An attempt was made to access a pageable (or completely invalid) address at
interrupt request level (IRQL) that is too high.  This is usually
caused by drivers using improper addresses.
If kernel debugger is available get stack backtrace.
Arguments:
Arg1: ffffde067b9b81a0, memory referenced
Arg2: 0000000000000002, IRQL
Arg3: 0000000000000000, value 0 = read operation, 1 = write operation
Arg4: fffff80176ff1530, address which referenced memory

Debugging Details:
------------------

ANALYSIS: IDebugSymbols::ResetScope
ANALYSIS: Set Context: .thread ; .cxr
ANALYSIS: IDebugSymbols::ResetScope
ANALYSIS: Set Context: .thread ; .cxr
```

6 Known Limitations

The following limitations apply only to the Intel® Debug Extensions for WinDbg* for IA JTAG debugging feature.

- A data breakpoint can be set on the first thread only.
- If connection to target fails, process "dllhost" is not killed automatically. The process must be killed manually to be able to try connection again.
- Breakpoint skip count is not supported.
- Execution HW breakpoints is not supported.
- Feature has been tested on
  - Intel® Atom™ Processor Z36xx, Z37xx - 2 cores (Baytrail / MinnowBoard MAX) under 32- and 64-bit Windows Embedded* 8.1
  - 6th Gen Intel® Core™ Processor based system and 64-bit Windows 10 preview

7 Troubleshooting

7.1 Error message "Unable to read debugger data block header" in WinDBG*

The error message "Unable to read debugger data block header" indicates that on the target the kernel debugging is not activated.
Activate kernel debugging by executing the command "`bcdedit /debug on`" on the target in a command prompt.

8 Change History

Below are features listed for older versions of the Intel® System Debugger releases.

8.1 Intel® System Debugger 2018 Initial Release

No changes

8.2 Intel® Debug Extension for WinDbg* 2018 Beta

- WinDbg* support Windows Driver Kit (WDK) version 1703. Added support for a new eXDI callback (DBGENG_EXDI_IOCTL_V3_GET_NT_BASE_ADDRESS_VALUE) to locate windows key structure KdVersionBlock.
- Extended Intel® Debug Extensions for WinDbg* for Intel® Processor Trace plugin to support Windows public symbol information.
- Extended Intel® Debug Extensions for WinDbg* for Intel® Processor Trace plugin to support ring 3 tracing.
- Extended Intel® Debug Extensions for WinDbg* for Intel® Processor Trace plugin to support decoding Intel® Processor Trace data from crash dump.

8.3 Intel® Debug Extension for WinDbg* 2017 Update 3

No change

8.4 Intel® Debug Extension for WinDbg* 2017 Update 2

- Installation improvements especially regarding Python* setup
- Added a WinDBG* theme
- Improved error messaging and reduced number of verbose messages during start.
- Print a warning if Kernel Debug on the target is not enabled.
- Add checks is Windows running for Windows debug use cases
- Deleting breakpoints works again for all targets.
- Improved error handling
- New Intel® Debug Extensions for WinDbg* welcome banner
- Updated user guide

8.5 Intel® Debug Extension for WinDbg* 2017 (Update 1)

No changes

8.6 Intel® Debug Extension for WinDbg* 2017 (Initial Release)

- Installation improvements especially regarding Python* setup
- Added a WinDBG* theme
- Fixed issues regarding debugging with Intel® Processor Trace
- The Intel® Debug Extensions for WinDbg* comprise the following features:
  - Intel® Debug Extensions for WinDbg* for IA JTAG debugging
    The Intel® Debug Extensions for WinDbg* for IA JTAG debugging (IA JTAG) enables the connection of WinDbg* to a target over the Joint Test Action Group (JTAG). The server acts as a mediator and forwards the calls from WindDbg* to the IPC interface and back.
  - Intel® Debug Extensions for WinDbg* for Intel® Processor Trace

The Intel® Debug Extensions for WinDbg* for Intel® Processor Trace (Intel® PT) is designed to help WinDbg* users by extending their debugging tool set with execution tracing. The extension allows for easy setup of Intel® PT by abstracting hardware configuration and then reconstructing and displaying execution flow from the collected trace data. It will integrate with other WinDbg* features like symbolization and high-level source display.

9 Attributions

Portions of this software were originally based on the following:
- software copyright (c) 1999, IBM Corporation., http://www.ibm.com.
- software copyright (c) 1999, Sun Microsystems., http://www.sun.com.
- the W3C consortium (http://www.w3c.org) ,
- the SAX project (http://www.saxproject.org)
- voluntary contributions made by Paul Eng on behalf of the
Apache Software Foundation that were originally developed at iClick, Inc.,
software copyright (c) 1999.

This product includes updcrc macro,
Satchell Evaluations and Chuck Forsberg.
Copyright (C) 1986 Stephen Satchell.

This product includes software developed by the MX4J project
(http://mx4j.sourceforge.net).

This product includes ICU 1.8.1 and later.
Copyright (c) 1995-2006 International Business Machines Corporation and others.

Portions copyright (c) 1997-2007 Cypress Semiconductor Corporation.
All rights reserved.

This product includes XORP.
Copyright (c) 2001-2004 International Computer Science Institute

This product includes software from the book
"Linux Device Drivers" by Alessandro Rubini and Jonathan Corbet,
published by O'Reilly & Associates.

This product includes hashtab.c.
Bob Jenkins, 1996.

10 Disclaimer and Legal Information

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at Intel.com, or from the OEM or retailer.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

Intel, the Intel logo, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.

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