ADVANCED MICRO DEVICES, INC.
END USER LICENSE AGREEMENT

IMPORTANT-READ CAREFULLY: DO NOT COPY OR USE THE SPECIFICATION DOCUMENTATION (AS DEFINED BELOW), OR ANY PORTION THEREOF, (COLLECTIVELY "DOCUMENTATION") UNTIL YOU HAVE CAREFULLY READ AND AGREED TO THE FOLLOWING TERMS AND CONDITIONS. THIS IS A LEGAL AGREEMENT ("AGREEMENT") BETWEEN YOU (EITHER AN INDIVIDUAL OR AN ENTITY) ("YOU") AND ADVANCED MICRO DEVICES, INC. ("AMD").

IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, DO NOT USE THIS DOCUMENTATION. BY COPYING OR USING THE DOCUMENTATION YOU AGREE TO ALL THE TERMS AND CONDITIONS OF THIS AGREEMENT.

1. DEFINITIONS.

a) "**Documentation**" means online or electronic documentation associated, included, or provided in connection with the ISA specification, or any portion thereof.
b) "**Intellectual Property Rights**" means all copyrights, trademarks, trade secrets, patents, mask works, and all related, similar, or other intellectual property rights recognized in any jurisdiction worldwide, including all applications and registrations with respect thereto.

2. LICENSE. Subject to the terms and conditions of this Agreement, AMD hereby grants You a non-exclusive, royalty-free, revocable, non-transferable, limited, copyright license to use the Documentation.

3. RESTRICTIONS. Except for the limited license expressly granted in Section 2 herein, You have no other rights in the Documentation, whether express, implied, arising by estoppel or otherwise. Further restrictions regarding Your use of the Documentation are set forth below. You may not:

a) modify or create derivative works of the Documentation;
b) distribute, publish, display, sublicense, assign or otherwise transfer the Documentation;
d) alter or remove any copyright, trademark or patent notice(s) in the Documentation; or
e) use the Documentation to: (i) develop inventions directly derived from Confidential Information to seek patent protection; (ii) assist in the analysis of Your patents and patent applications; or (iii) modify existing patents.

4. FEEDBACK. You have no obligation to give AMD any suggestions, comments or other feedback ("Feedback") relating to the Documentation. However, AMD may use and include any Feedback that it receives from You to improve the Documentation or other AMD products, software and technologies. Accordingly, for any Feedback You provide to AMD, You grant AMD and its affiliates and subsidiaries a worldwide, non-exclusive, irrevocable, royalty-free, perpetual license to, directly or indirectly, use, reproduce, license, sublicense, distribute, make, have made, sell and otherwise commercialize the Feedback in the Documentation or other AMD products, software and technologies. You further agree not to provide any Feedback that (a) You know is subject to any Intellectual Property Rights of any third party or (b) is subject to license terms which seek to require any products incorporating or derived from such Feedback, or other AMD intellectual property, to be licensed to or otherwise shared with any third party.

5. OWNERSHIP AND COPYRIGHT OF DOCUMENTATION. The Documentation, including all Intellectual Property Rights therein, is and remains the sole and exclusive property of AMD or its licensors, and You shall have no right, title or interest therein except as expressly set forth in this Agreement.

6. WARRANTY DISCLAIMER: THE DOCUMENTATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. AMD DISCLAIMS ALL WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, NON-INFRINGEMENT, OR WARRANTIES ARISING FROM CUSTOM OF TRADE OR COURSE OF USAGE. THE ENTIRE RISK ASSOCIATED WITH THE USE OF THE DOCUMENTATION IS ASSUMED BY YOU. Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to You.

7. LIMITATION OF LIABILITY AND INDEMNIFICATION: AMD AND ITS LICENSORS WILL NOT, UNDER ANY CIRCUMSTANCES BE LIABLE TO YOU FOR ANY PUNITIVE, DIRECT, INCIDENTAL, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING FROM USE OF THE DOCUMENTATION OR THIS AGREEMENT EVEN IF AMD AND ITS LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. In no event shall AMD's total liability to You for all damages, losses, and causes of action (whether in contract, tort (including negligence) or otherwise) exceed the amount of $100 USD. You agree to defend, indemnify and hold harmless AMD and its licensors, and any of their directors, officers, employees, affiliates or agents from and against any and all loss, damage, liability and other expenses (including reasonable attorneys' fees), resulting from Your use of the Documentation or violation of the terms and conditions of this Agreement.

8. EXPORT RESTRICTIONS: You shall adhere to all applicable U.S. import/export laws and regulations, as well as the import/export control laws and regulations of other countries as applicable. You further agree to not export, re-export, or transfer, directly or indirectly, any product, technical data, software or source code received from AMD under this license, or the direct product of such technical data or software to any country for which the United States or any other applicable government requires an export license or other governmental approval without first obtaining such licenses or approvals; or in violation of any applicable laws or regulations of the United States or the country where the technical data or software was obtained. You acknowledge the technical data and software received will not, in the absence of authorization from U.S. or local law and regulations as applicable, be used by or exported, re-exported or transferred to: (i) any sanctioned or embargoed country, or to nationals or residents of such countries; (ii) any restricted end-user as identified on any applicable government end-user list; or (iii) any party where the end-use involves nuclear, chemical/biological weapons, rocket systems, or unmanned air vehicles. For the most current Country Group listings, or for additional information about the EAR or Your obligations under those regulations, please refer to the U.S. Bureau of Industry and Security's website at http://www.bis.doc.gov/.

9. NOTICE TO U.S. GOVERNMENT END USERS. The Documentation is considered "commercial items", as that term is defined at 48 C.F.R. §2.101, consisting of "commercial computer software" and "commercial computer software documentation", as such terms are used in 48 C.F.R. §12.212 and 48 C.F.R. §227.7202, respectively. Consistent with 48 C.F.R. §12.212 or 48 C.F.R. §227.7202-1 through 227.7202-4, as applicable, the commercial computer software and commercial computer software documentation are being licensed to U.S. Government end users (a) only as commercial items and (b) with only those rights as are granted to all other end users pursuant to the terms and conditions set forth in this Agreement. Unpublished rights are reserved under the copyright laws of the United States.

10. TRANSMISSION. The parties agree the Documentation will be transmitted by remote telecommunication with the receipt of no tangible personal property other than Documentation.

11. TERMINATION OF LICENSE. This Agreement will terminate immediately without notice from AMD or judicial resolution if (1) You fail to comply with any provisions of this Agreement, or (2) You provide AMD with notice that You would like to terminate this Agreement. Upon termination of this Agreement, You must delete or destroy all copies of the Documentation. Upon termination or expiration of this Agreement, all provisions survive except for Section 2.

12. GOVERNING LAW. This Agreement is made under and shall be construed according to the laws of the State of Texas, excluding conflicts of law rules. Each party submits to the jurisdiction of the state and federal courts of Travis County and the Western District of Texas for the purposes of this Agreement. You acknowledge that Your breach of this Agreement may cause irreparable damage and agree that AMD shall be entitled to seek injunctive relief under this Agreement, as well as such further relief as may be granted by a court of competent jurisdiction.

13. GENERAL PROVISIONS. You may not assign this Agreement without the prior written consent of AMD and any assignment without such consent will be null and void. The parties do not intend that any agency or partnership relationship be created between them by this Agreement. Each provision of this Agreement shall be interpreted in such a manner as to be effective and valid under applicable law.

However, in the event that any provision of this Agreement becomes or is declared unenforceable by any court of competent jurisdiction, such provision shall be deemed deleted and the remainder of this Agreement shall remain in full force and effect.

14. ENTIRE AGREEMENT. This Agreement sets forth the entire agreement and understanding between the parties with respect to the Documentation and supersedes and merges all prior oral and written agreements, discussions and understandings between them regarding the subject matter of this Agreement. No waiver or modification of any provision of this Agreement shall be binding unless made in writing and signed by an authorized representative of each party.

Preliminary Information—Subject to Change

AMD Supervisor Entry Extensions

Publication # **57115** Revision: **0.50**
Issue Date: **February 2021**

Advanced Micro Devices

The information contained herein is for informational purposes only, and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale. Any unauthorized copying, alteration, distribution, transmission, performance, display or other use of this material is prohibited.

Trademarks

AMD, the AMD Arrow logo, AMD AllDay, AMD Virtualization, AMD-V, PowerPlay, Vari-Bright, and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Reverse engineering or disassembly is prohibited.

USE OF THIS PRODUCT IN ANY MANNER THAT COMPLIES WITH THE MPEG ACTUAL OR DE FACTO VIDEO AND/OR AUDIO STANDARDS IS EXPRESSLY PROHIBITED WITHOUT ALL NECESSARY LICENSES UNDER APPLICABLE PATENTS. SUCH LICENSES MAY BE ACQUIRED FROM VARIOUS THIRD PARTIES INCLUDING, BUT NOT LIMITED TO, IN THE MPEG PATENT PORTFOLIO, WHICH LICENSE IS AVAILABLE FROM MPEG LA, L.L.C., 6312 S. FIDDLERS GREEN CIRCLE, SUITE 400E, GREENWOOD VILLAGE, COLORADO 80111.

Contents

List of Figures **5**
List of Tables **5**
Revision History **6**
Chapter 1 Introduction **7**
1.1 SYSCALL/SYSRET 7
1.2 Re-Entrant Exceptions 7
Chapter 2 Enhanced SYSCALL/SYSRET (ESC) **8**
2.1 Determining Support for Enhanced SYSCALL/SYSRET 8
2.2 Enabling Enhanced SYSCALL/SYSRET 8
2.3 New MSRs 8
2.4 Enhanced SYSCALL Behavior 8
2.5 Enhanced SYSRET Behavior 9
Chapter 3 Reserved Supervisor Shadow Stacks (RSSS) **10**
3.1 Determining Support for RSSS 10
3.2 Enabling RSSS 10
3.3 RSSS Behavior 10
3.3.1 WRMSR PL[0,1,2]_SSP 10
3.3.2 Stack Switching 10
Chapter 4 Re-Entrancy Protection **11**
4.1 Determining Support for Re-Entrancy Protection 11
4.2 Enabling Re-Entrancy Protection 11
4.3 New MSRs 11
4.4 New Behavior 11
4.4.1 IDT Extension 11
4.4.2 Exception Processing 11
4.4.3 Interrupt Shadow Tracking 12
4.4.4 IRET 12
4.4.5 New Exception Frame 12
4.5 NMI Mask 13
4.6 Additional Information 13

4.6.1 Re-Entrancy Protection 13
4.6.2 RPE and NMI 13
4.6.3 Machine Check (#MC) 13
Chapter 5 Additional Features 14
5.1 Loading of RSP on IRET/RET FAR 14
5.2 Suppression of #DB on Interrupts/Exceptions and Other Transfers 14
5.3 Partial FS/GS Loading 14
Appendix A Pseudo-Code 15
A.1 SYSCALL 15
A.2 SYSRET 17

List of Figures

Figure 1. Exception Frames (No Error Code and With Error Code) 12

List of Tables

Table 1. ExcpInfo Byte Definition..13

Revision History

Date	Revision	Description
February	0.50	Initial preliminary release for public comment. ***Note:*** *This content in this document is subject to change.*

Chapter 1 Introduction

In the x86 architecture, system software must handle certain types of interrupts and/or exceptions that occur at inopportune times. There are two scenarios in particular that result in significant software complexity to properly deal with circumstances that in practice are rare but cannot be ignored.

Note: *The material contained in this document is preliminary and many bit definitions, MSR numbers, offsets, etc. are undefined and noted as to be determined (TBD). These values will be specified prior to the finalization of this specification. Please send comments and feedback on this specification to* dl.ISAEXT.feedback@amd.com.

1.1 SYSCALL/SYSRET

The SYSCALL and SYSRET instructions do not atomically switch all CPU state necessary to transition a modern OS between user and supervisor mode. After SYSCALL entry, a typical OS requires performing a SWAPGS instruction and loading a kernel stack pointer, and potentially a shadow stack pointer, before the kernel can process interrupts and exceptions. While SYSCALL clears IF blocking normal interrupts, it is still possible for maskable interrupts (NMIs) to occur. Additionally, certain types of x86 exceptions may occur during this time including #DB, #HV, #SX, etc. A typical 64-bit OS uses the IST to set up a kernel stack when such exceptions occur.

1.2 Re-Entrant Exceptions

There is no architectural protection against being in the handler for an exception and receiving the same exception again. For example, a #SX could occur during the #SX handler. When exceptions use the IST mechanism to establish a stack, taking the same exception again can lead to an infinite loop as the exception frame on the stack gets overwritten.

The instruction set architecture (ISA) extension proposed here is designed to address both challenges and to limit the need for using the IST mechanism. This is accomplished by enhancing the SYSCALL/SYSRET behavior to swap additional state and preventing unintentional re-entrancy by tracking in-progress exceptions. Additional enhancements included in this specification reduce the software complexity required to handle various corner cases in the AMD64 architecture.

To support fast SYSCALL/SYSRET paths when supervisor shadow stacks are enabled, a new ISA extension is proposed that improves the handling of busy bits. This extension is expected to be enabled when shadow stacks are used alongside the enhanced SYSCALL/SYSRET instructions.

Chapter 2 Enhanced SYSCALL/SYSRET (ESC)

2.1 Determining Support for Enhanced SYSCALL/SYSRET

CPUID Fn<TBD> bit <TBD> indicates support for enhanced system call instruction behavior.

2.2 Enabling Enhanced SYSCALL/SYSRET

Software may enable the enhanced SYSCALL/SYSRET behavior by setting EnhancedSystemCallEnable (ESCE) in EFER (MSR C000_0080h) bit <TBD>.

The enhanced SYSCALL and SYSRET behavior is only enabled when the CPU is in long mode.

2.3 New MSRs

The new MSR STSTAR (TBD) specifies the target RSP value to be loaded upon kernel entry. The same value is used regardless of whether the calling software is in 64-bit mode or in compatibility mode.

The STSTAR MSR is saved and restored from VMCB save area offset <TBD>. For SEV-ES guests, this MSR is saved and restored from VMSA offset <TBD>.

2.4 Enhanced SYSCALL Behavior

When the ESC extension is enabled and SYSCALL is executed, the processor performs most of the actions associated with a normal SYSCALL and then takes the following additional actions:

1. The user GS.base value and KernelGSBase values are swapped, as is done in the SWAPGS instruction.
2. The RSP register is loaded with the value in STSTAR.
3. If S_CET.SH_STK_EN=1, the SSP is loaded with the value from PL0_SSP.

   Note: *The shadow stack busy bit is NOT set. Software is expected to enable Reserved Supervisor Shadow Stacks (RSSS). For more information about RSSS, see Chapter 3 on page 10.*
4. A standard exception frame is created on the new stack that contains the next-RIP, CS, RFLAGS, RSP, and SS at the time that SYSCALL was executed. Additionally, if SYSCALL is executed within an interrupt shadow, any previously pending debug traps are dropped.

2.5 Enhanced SYSRET Behavior

When the ESC extension is enabled and SYSRET is executed, the processor returns to user mode. User mode state is restored from the stack, although the CS and SS selectors are ignored. The GDT/LDT are not referenced when loading the segment registers, and the segment information is instead loaded with fixed values for both CS and SS. The GS base values are again swapped.

Note: *The busy bit on the supervisor mode shadow stack is* ***not*** *cleared.*

Chapter 3 Reserved Supervisor Shadow Stacks (RSSS)

The Reserved Supervisor Shadow Stacks (RSSS) feature is an enhancement that marks a supervisor shadow stack as busy when the PL[0,1,2]_SSP MSR is written. This effectively reserves the shadow stack for future use and allows for faster switching to this shadow stack.

3.1 Determining Support for RSSS

CPUID Fn<TBD> bit<TBD> indicates support for reserved supervisor shadow stacks. The RSSS feature is supported on all CPUs that support ESC.

3.2 Enabling RSSS

Software enables the reserved supervisor shadow stacks by setting ReservedSupervisorShadowStackEnable (RSSSE) in the S_CET MSR bit TBD. It is recommended that software enable RSSS before the supervisor shadow stacks are initially configured.

3.3 RSSS Behavior

3.3.1 WRMSR PL[0,1,2]_SSP

When RSSS is enabled, the shadow stack busy bits are modified when a WRMSR to PL[0,1,2]_SSP occurs. If the previous value of the MSR was non-0, the busy bit corresponding to the previous shadow stack pointer value is cleared (similar to CLRSSBSY). The busy bit corresponding to the new shadow stack pointer value is then set (similar to SETSSBSY), if the new value is non-0.

If an error occurs attempting to clear the old busy bit, the WRMSR fails with an appropriate exception, and the MSR contents are not modified. If the old busy bit was already 0, this is not considered an error, and the bit will not be changed.

If an error occurs while attempting to set the new busy bit, the WRMSR fails with a #CP(SETSSBSY) exception, and the MSR contents is set to 0.

3.3.2 Stack Switching

On CPL transitions when supervisor shadow stacks are enabled, new supervisor shadow stack pointers are loaded from the appropriate MSR, but no busy bits are checked by hardware. This improves the performance of CPL transitions.

Chapter 4 Re-Entrancy Protection

4.1 Determining Support for Re-Entrancy Protection

CPUID Fn<TBD> bit <TBD> indicates support for exception re-entrancy protection.

4.2 Enabling Re-Entrancy Protection

Software may enable re-entrancy protection by setting ReEntrantProtectionEn (RPE) in EFER (MSR C000_0080h) bit TBD. Re-entrancy protection is only observed when the CPU is in long mode.

4.3 New MSRs

The new MSR EXCP_IN_PROG (TBD) contains a bit vector corresponding to x86 exception vectors 0-31. Out of reset, this MSR is set to 0h. If bit N is set in EXCP_IN_PROG, it means that exception N is in progress (in some stage of being handled by software).

The EXCP_IN_PROG MSR is saved and restored from VMCB save area offset <TBD>. For SEV-ES guests, this MSR is saved and restored from VMSA offset <TBD>.

4.4 New Behavior

4.4.1 IDT Extension

In 64-bit mode, a new bit called Re-EntrantProtect (RP) is defined in the IDT interrupt and trap gate descriptors. The RP bit is bit 7 in the second DWORD of the IDT descriptor. If RPE is set to 1 and RP is set to 1, it indicates that the IDT vector is not re-entrant and the busy bit should be set on exception entry.

Note: *The RP bit is only defined for vectors 0-31.*

4.4.2 Exception Processing

When the CPU is in long mode and an x86 exception occurs when RPE is set to 1 and the RP bit of the IDT entry is set to 1, the CPU performs all standard exception handling functions with two additions:

1. The EXCP_IN_PROG MSR bit corresponding to the new exception is checked.
   a. If this bit is 1, then a double fault (#DF) is generated.
      If the new exception was already a #DF, then a SHUTDOWN is generated.
   b. If this bit is 0, then it is set to 1 and normal exception processing resumes.

2. The 1-byte exception vector is written to the exception frame in the new ExcpVec field. Additionally, the ExcpValid bit in the new ExcpInfo field is set to 1.

4.4.3 Interrupt Shadow Tracking

When the CPU is in long mode with RPE set to 1 and an x86 exception occurs while in an interrupt shadow, the CPU sets the new IntShadow bit in ExcpInfo in the exception frame. This bit is later used on IRET to suppress interrupt checking after returning from the exception handler. If an exception occurs while in an interrupt shadow, a pending unmasked interrupt is recognized by the processor before execution of the exception handler begins.

4.4.4 IRET

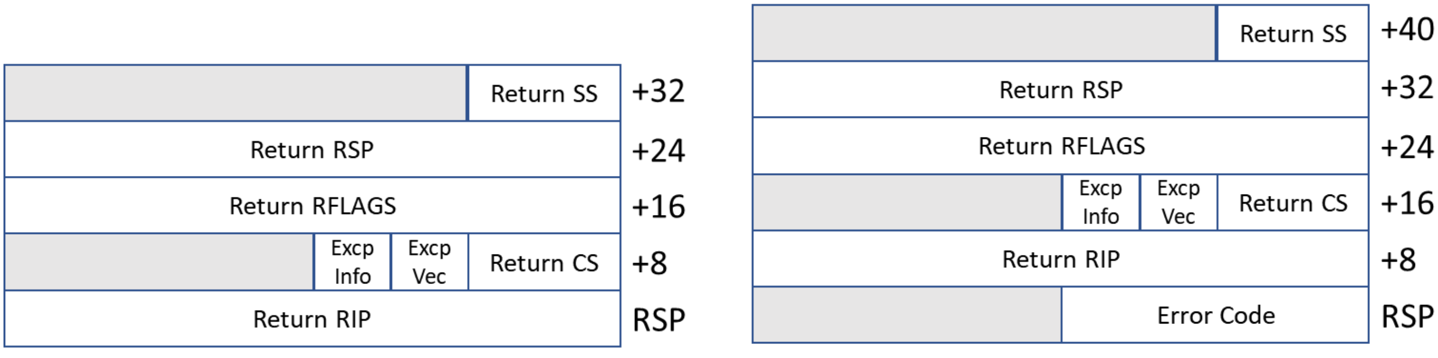
When IRET is executed in long mode and RPE is set to 1, the CPU also performs the following actions:

1. The ExcpValid bit on the stack is read.
2. If this bit is 1 then the exception vector being finished is read from the stack and the corresponding bit in EXCP_IN_PROG is cleared to 0
3. The IntShadow bit on the stack is read.
   If this bit is a 1, the processor enters an interrupt shadow after completion of the IRET.

Note: *If software wishes to perform an IRET without clearing the exception busy bit, it must clear the exception vector valid bit in the exception stack frame prior to executing IRET.*

4.4.5 New Exception Frame

The exception frame layout when RPE is set to 1 is shown in Figure 1, and the ExcpInfo byte definition is shown in Table 1.



Exception frame (no error code)

Exception frame (with error code)

Figure 1. Exception Frames (No Error Code and With Error Code)

Table 1. ExcpInfo Byte Definition

7:2	1	0
(Reserved)	IntShadow	ExcpValid

4.5 NMI Mask

When RPE is set to 1, bit 2 of the EXCP_IN_PROG MSR is treated as the processor's NMI mask. Additional NMIs are held pending while EXCP_IN_PROG[2] is set to 1 and are only taken when that bit is cleared to 0. Software may explicitly mask/unmask NMIs by modifying this MSR bit.

In guest mode, EXCP_IN_PROG[2] only affects the handling of NMI exceptions within the guest and does not control the physical processor's NMI mask.

4.6 Additional Information

4.6.1 Re-Entrancy Protection

Re-entrancy protection ensures that double faults are generated if a second exception of the same type occurs before software is ready to handle it. The exception-in-progress bit is naturally cleared on the IRET at the end of the exception handler; however, the exception-in-progress bit may be cleared earlier through a WRMSR to EXCP_IN_PROG.

4.6.2 RPE and NMI

When RPE is used, IRET only clears the processor's NMI mask if the stack frame indicates a return from an NMI handler. This implies that other exceptions may be taken during NMI handling; however, the NMI mask is only cleared when the IRET of the NMI handler (as opposed to the IRET of any exception handler) is executed.

4.6.3 Machine Check (#MC)

The machine check (#MC) handling is already partially non-reentrant because the processor sets the MCIP (Machine Check in Progress) bit in MSR MCG_STAT when an #MC is taken. If another #MC is taken while MCIP=1, then a SHUTDOWN is generated. However, there is a time window in the #MC handler after MCIP is cleared and before IRET when another machine check could occur. For this reason, software may wish to configure the #MC handler with RP=1 to ensure that a second machine check that occurs in this window results in a #DF.

Chapter 5 Additional Features

5.1 Loading of RSP on IRET/RET FAR

In the AMD64 architecture, an IRET or RET FAR instruction, which loads a new stack pointer when returning to a 16-bit stack segment, only modifies bits 15:0 of the stack pointer (SP). Therefore, when returning to user-mode code, supervisor-mode software must ensure that no sensitive information leaks in the unmodified upper bits of the stack pointer.

The new CPUID Fn<TBD> bit<TBD> indicates that when executed in 64-bit mode, IRET and RET FAR load the entire 64-bit RSP register from the stack regardless of the stack segment size that is being returned to. This behavior is unconditional on CPUs that set this CPUID bit.

5.2 Suppression of #DB on Interrupts/Exceptions and Other Transfers

When CPUID Fn<TBD> bit <TBD> is set, it indicates that any exceptions, software interrupts (INT N, INT3, INTO, etc.), call gate transfers, or conforming CALL FARs that occur during an interrupt shadow cause any pending debug exceptions to be dropped. This ensures that a debug breakpoint that occurred in user code is not taken at the start of a supervisor-mode exception handler.

5.3 Partial FS/GS Loading

MSR PARTIAL_FS_LOAD (TBD) and PARTIAL_GS_LOAD (TBD) enable 64-bit supervisor mode software to load a new FS or GS segment respectively; however, they do not modify the cached segment base. These MSRs are available on all processors that support ESC and are intended for use by supervisor code to load an FS or GS segment on behalf of a user program.

To use the PARTIAL_FS_LOAD or PARTIAL_GS_LOAD MSRs, software writes the new segment selector to bits 15:0 of the MSR. The processor proceeds to perform standard segment checks and, if successful, modifies only the selector, attribute, and limit fields of the segment in the segment cache. RDMSR of either MSR or a WRMSR, when not in 64-bit mode, results in #GP(0).

Appendix A Pseudo-Code

A.1 SYSCALL

```
SYSCALL_START:
  IF (MSR_EFER.SCE == 0) // Check if syscall/sysret are enabled.
    EXCEPTION [#UD]

  IF (LONG_MODE)
    SYSCALL_LONG_MODE
  ELSE // (LEGACY_MODE)
    SYSCALL_LEGACY_MODE

SYSCALL_LONG_MODE:
  IF (!EFER.ESCE) {
    RCX.q = next_RIP
    R11.q = RFLAGS // with rf cleared
  }

  IF (64BIT_MODE)
    temp_RIP.q = MSR_LSTAR
  ELSE // (COMPATIBILITY_MODE)
    temp_RIP.q = MSR_CSTAR

  IF (ShadowStacksEnabled at current CPL)
      PL3_SSP = SSP

  IF (EFER.ESCE) {
    Swap GS.base and KernelGSBase

    oldRSP = RSP
    RSP.q = MSR_STSTAR

    oldCS = CS.sel
    oldSS = SS.sel


  // Busy bit already has been set at the time the MSR was written (through RSSS feature)
  IF (ShadowStacksEnabled at CPL0)
      SSP = PL0_SSP
  }

  CS.sel = MSR_STAR.SYSCALL_CS AND 0xFFFC
  CS.attr = 64-bit code,dpl0 // Always switch to 64-bit mode in long mode.
  CS.base = 0x00000000
  CS.limit = 0xFFFFFFFF

  SS.sel = MSR_STAR.SYSCALL_CS + 8
  SS.attr = 64-bit stack,dpl0
  SS.base = 0x00000000
  SS.limit = 0xFFFFFFFF

  IF (EFER.ESCE) {
    // Create exception frame
    PUSH.q oldSS
    PUSH.q oldRSP
    PUSH.q RFLAGS
    PUSH.q oldCS
    PUSH.q next_RIP
  }


  RFLAGS = RFLAGS AND ~MSR_SFMASK
  RFLAGS.RF = 0

  CPL = 0

  RIP = temp_RIP
  EXIT

SYSCALL_LEGACY_MODE:
  RCX.d = next_RIP
  temp_RIP.d = MSR_STAR.EIP

  CS.sel = MSR_STAR.SYSCALL_CS AND 0xFFFC
  CS.attr = 32-bit code,dpl0 // Always switch to 32-bit mode in legacy mode.
  CS.base = 0x00000000
  CS.limit = 0xFFFFFFFF

  SS.sel = MSR_STAR.SYSCALL_CS + 8
  SS.attr = 32-bit stack,dpl0
  SS.base = 0x00000000
  SS.limit = 0xFFFFFFFF

  RFLAGS.VM,IF,RF=0

  CPL = 0
  RIP = temp_RIP
EXIT
```

A.2 SYSRET

```
SYSRET_START:
  IF (MSR_EFER.SCE == 0) // Check if syscall/sysret are enabled.
    EXCEPTION [#UD]

  IF ((!PROTECTED_MODE) || (CPL != 0))
    EXCEPTION [#GP(0)] // SYSRET requires protected mode, cpl0

  IF (64BIT_MODE)
    SYSRET_64BIT_MODE
  ELSE // (!64BIT_MODE)
    SYSRET_NON_64BIT_MODE

SYSRET_64BIT_MODE:
  IF (EFER.ESCE)
  {
    Swap GS.base and KernelGSBase

    // Read exception frame
    POP.q temp_RIP
    POP.q temp_CS // not used
    POP.q temp_RFLAGS
    POP.q temp_RSP
    POP.q temp_SS // not used
  }

  IF (OPERAND_SIZE == 64) // Return to 64-bit mode.
  {
    CS.sel = (MSR_STAR.SYSRET_CS + 16) OR 3
    CS.base = 0x00000000
    CS.limit = 0xFFFFFFFF
    CS.attr = 64-bit code,dpl3
    IF (!EFER.ESCE)
      temp_RIP.q = RCX
  }
  ELSE // Return to 32-bit compatibility mode.
  {
    CS.sel = MSR_STAR.SYSRET_CS OR 3
    CS.base = 0x00000000
    CS.limit = 0xFFFFFFFF
    CS.attr = 32-bit code,dpl3
    IF (!EFER.ESCE)
      temp_RIP.d = RCX
  }

  IF (!EFER.ESCE)
  {
      SS.sel = MSR_STAR.SYSRET_CS + 8 // SS selector is changed,
    // SS base, limit, attributes unchanged.
  }
  ELSE
  {
    // Set SS base/limit/attributes
    SS.sel = (MSR_STAR.SYSRET_CS + 8) OR 3
    SS.base = 0x00000000
    SS.limit = 0xFFFFFFFF
    SS.attr = read-write data,dpl3
  }


  IF (!EFER.ESCE)
    temp_RFLAGS = R11

  // Set shadow stack pointer if enabled in user mode
  IF (ShadowStacksEnabled at CPL3)
    SSP = PL3_SSP

  // Restore user stack pointer
  IF (EFER.ESCE)
      RSP.q = temp_RSP

  RFLAGS.q = temp_RFLAGS // RF=0,VM=0
  CPL = 3

  RIP = temp_RIP
  EXIT

SYSRET_NON_64BIT_MODE:
  CS.sel = MSR_STAR.SYSRET_CS OR 3 // Return to 32-bit legacy protected mode.
  CS.base = 0x00000000
  CS.limit = 0xFFFFFFFF
  CS.attr = 32-bit code,dpl3

  temp_RIP.d = RCX

  SS.sel = MSR_STAR.SYSRET_CS + 8 // SS selector is changed.
  // SS base, limit, attributes unchanged.
  RFLAGS.IF = 1
  CPL = 3

  RIP = temp_RIP
  EXIT
```