USER INTERRUPT COMPILER GUIDE

H.J. Lu

IAGS/OISA/DSE/CE

March 2021

Introduction

User Interrupt (UINTR)

- Interrupt can be delivered to interrupt handler in user space.
- UINTR handler has a different calling convention:
  - Hardware defined stack frame
  - All registers must be preserved.
  - “UIRET” must be used to return from UINTR handler.

Compiler Options

- -muintr
  - Enable UINTR handlers.
  - Enable UINTR intrinsics.
- -mgeneral-regs-only
  - Generate code that uses only the integer registers.
- -minline-all-stringops
  - Inline memcpy, memmove, memset and memcmp to avoid vector register usage in library functions.

Interrupt Attribute

Use this attribute to indicate that the specified function is a UINTR handler. The compiler generates function entry and exit sequences suitable for use in a UINTR handler when this attribute is present. The ‘UIRET‘ instruction, instead of the 'RET' instruction, is used to return from UINTR handlers. All registers, except for the RFLAGS register which is restored by the ‘UIRET' instruction, are preserved by the compiler. '-mgeneral-regs-only -minline-all-stringops' must be used to compile UINTR handlers.

```
#include <x86gprintrin.h>     /* For struct __uintr_frame  */
 __attribute__ ((interrupt))
 void f (struct __uintr_frame *frame, unsigned long long uirrv)
 {
 }
```

User Interrupt Frame

Interrupt state can be accessed in UINTR handler with a pointer argument to

```
struct __uintr_frame
{
  unsigned long long rip;
  unsigned long long rflags;
  unsigned long long rsp;
};
```

__attribute__((target("general-regs-only")))

Use this attribute to indicate that only integer registers should be used when generating code for this function. If the function explicitly uses non-integer code, then the compiler gives an error.

no_caller_saved_registers Attribute

Use this attribute to indicate that the specified function has no caller-saved registers. That is, all registers are callee-saved. For example, this attribute can be used for a function called from a UINTR handler. The compiler generates proper function entry and exit sequences to save and restore any modified integer registers, except for the RFLAGS register.

UINTR Intrinsics

#include <x86gprintrin.h>

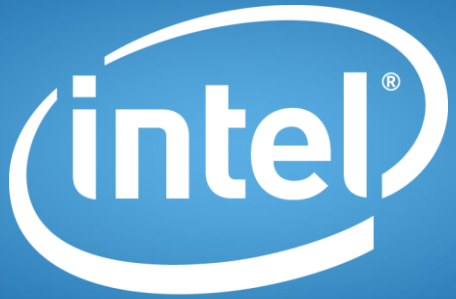
Instruction	Intrinsic
clui	extern void _clui (void);
stui	extern void _stui (void);
testui	extern unsigned char _testui (void);
senduipi r64	extern void _senduipi (unsigned long long);

UINTR Programming

- UINTR handlers must be marked with interrupt attribute.
  - Interrupt state can be accessed with the pointer argument.
- UINTR handlers and all functions called by UINTR handlers must be compiled separately with “-muintr -mgeneral-regs-only -minline-all-stringops” compiler options.
  - memcpy, memmove, memset nor memcmp can’t be used since in they won’t preserve vector registers.
  - __attribute__((target("general-regs-only", "inline-all-stringops"))) can be used on functions called by UINTR handlers if “-muintr -mgeneral-regs-only -minline-all-stringops” can’t be used.
- UINTR intrinsics can be used with “–muintr” compiler option.

UINTR Code Recommendations

- Place UINTR handlers and all functions calledd by UINTR handlers in separate files.
  - Compile them with “-muintr -mgeneral-regs-only -minline-all-stringops” compiler options.
  - Verify there are no references to memcpy, memmove, memset nor memcmp.
- Mark functions called by UINTR handlers with no_caller_saved_registers attribute to avoid unnecessary register save and restore.
- Avoid complex operations in UINTR handlers:
  - Better compiler optimization.
  - Faster UINTR processing.



Software