All Places > **STM32 MCUs Community** > **STM32 MCUs Forum** > **Discussions**

Log in to create and rate content, and to follow, bookmark, and share content with other members.

ANSWERED

port.c(483): error: A1586E: Bad operand types (UnDefOT, Constant) for operator (

Question asked by **Gharibi.Zaher** on Oct 28, 2017
Latest reply on Nov 9, 2017 by lifeland

Like • 0 Comment • 14

I'm trying to compile a project generated using STM32CubeMX for MDK-ARM but the compiler gives the following error:

> port.c(483): error: A1586E: Bad operand types (UnDefOT, Constant) for operator (

everytime I enable the "FreeRTOS" option in CubeMX. Initially, and after generating the code from CubeMX, everything compiles without an issue, but as soon as I start to add some source files to the project, the compiler starts to complain about that error in port.c file. Removing the newly added sources with their include paths doesn't help solve the issue. It's just like if something corrupts the FreeRTOS port sources and it won't go away until I disable the "FreeRTOS" option then re-enable it and generate the code again.

Upgrading to the latest STM32CubeMx with the latest firmware packages for my device did not help, as well!

eggcar
Nov 6, 2017 9:15 AM

Correct Answer

It seems that the stm32f407xx.h(or other models) constant definition is the culprit.

```
/**
  * @brief Configuration of the Cortex-M4 Processor and Core
Peripherals
  */
#define __CM4_REV                 0x0001U  /*!< Core revision
r0p1                           */
#define __MPU_PRESENT             1U       /*!< STM32F4XX provides
```

```
an MPU                     */
#define __NVIC_PRIO_BITS          4U        /*!< STM32F4XX uses 4
Bits for the Priority Levels */
#define __Vendor_SysTickConfig    0U        /*!< Set to 1 if
different SysTick Config is used  */
#define __FPU_PRESENT             1U        /*!< FPU
present                                */
```

Just remove the 'U' behind these constants, it then works well. Seems that armcc assemblier does not recognize C-format constant representation.

See the reply in context

No one else had this question

Visibility: STM32 MCUs Forum • **288 Views**

Last modified on Oct 28, 2017 9:35 PM

Tags: stm32f4 freertos port.c(483): error: a1586e: ...

0

14 Replies

Turvey.Clive.002
Oct 28, 2017 9:40 PM

Guess you'd want to look closely at the line in question, and ask yourself why the *assembler* is throwing an error on it.

What version of Keil/MDK-ARM ?
Present the surrounding lines, and any #ifdef/#endif around them.

Actions Like • 0

Gharibi.Zaher
@ Turvey.Clive.002 on Oct 28, 2017 9:52 PM

I'm on version 5.15

And here is the assembler block in question:

> ```
> __asm void xPortPendSVHandler( void )
> {
> extern uxCriticalNesting;
> extern pxCurrentTCB;
> extern vTaskSwitchContext;
> PRESERVE8
> mrs r0, psp
> isb
> /* Get the location of the current TCB. */
> ldr r3, =pxCurrentTCB
> ldr r2, [r3]
> /* Is the task using the FPU context? If so, push high vfp registers. */
> tst r14, #0x10
> it eq
> vstmdbeq r0!, {s16-s31}
> /* Save the core registers. */
> stmdb r0!, {r4-r11, r14}
> /* Save the new top of stack into the first member of the TCB. */
> str r0, [r2]
> stmdb sp!, {r3}
> mov r0, #configMAX_SYSCALL_INTERRUPT_PRIORITY
> msr basepri, r0
> dsb
> isb
> bl vTaskSwitchContext
> mov r0, #0
> msr basepri, r0
> ldmia sp!, {r3}
> /* The first item in pxCurrentTCB is the task top of stack. */
> ldr r1, [r3]
> ldr r0, [r1]
> /* Pop the core registers. */
> ldmia r0!, {r4-r11, r14}
> /* Is the task using the FPU context? If so, pop the high vfp registers
> too. */
> tst r14, #0x10
> ```

```
it eq
vldmiaeq r0!, {s16-s31}
msr psp, r0
isb
#ifdef WORKAROUND_PMU_CM001 /* XMC4000 specific errata */
#if WORKAROUND_PMU_CM001 == 1
push { r14 }
pop { pc }
nop
#endif
#endif
bx r14
nop
nop
}
```

I guess it's something related to memory allocation on the stack? If I was not wrong on my assumption, this happens everytime I try to add some source files from the demo package that allocate memory on the CCM 0x10000000 region.

** Just updated the block!

Actions

Like • 0

Turvey.Clive.002

@ Gharibi.Zaher on Oct 28, 2017 10:17 PM

So assuming an STM32F4 part?
Which is the offending line here?
Is the FPU enabled in the target options dialog?

Actions

Like • 0

Gharibi.Zaher

@ Turvey.Clive.002 on Oct 28, 2017 10:25 PM

Yes, it is an STM32F407, and the FPU is enabled, as well! I'm sorry, I should have mentioned that from the beginning.
The offending line is:

> mov r0, configmax_syscall_interrupt_priority

I believe I was wrong on my assumption. It has nothing to do with memory allocation on the CCM region. I removed that piece of code and the assembler still throws the error.

Actions

Like • 0

Turvey.Clive.002
@ Gharibi.Zaher on Oct 28, 2017 10:29 PM

perhaps add the line in the .C file in one of the other functions

printf("%d\n", configMAX_SYSCALL_INTERRUPT_PRIORITY);

Or mouse over **configMAX_SYSCALL_INTERRUPT_PRIORITY** and see if you can right click to the definition, and it is numeric.

Actions

Like • 0

Gharibi.Zaher
@ Turvey.Clive.002 on Oct 28, 2017 10:42 PM

I have disabled Browse Info in the target options for faster compile time. I just re-enabled it and now compiling the project. I can also debug that numeric. I have ITM Trace Support for debugger. Will get back with that numeric shortly...

Actions

Like • 0

Gharibi.Zaher
@ Turvey.Clive.002 on Oct 28, 2017 11:04 PM

It took it more than 15 minutes to compile with Browse Info enabled in target options. Here is what I got so far:

> #define configMAX_SYSCALL_INTERRUPT_PRIORITY (configLIBRARY_MAX_SYSCALL_INTERRUPT_PRIORITY << (8 - configPRIO_BITS))

Actions

Like • 0

Gharibi.Zaher
@ Turvey.Clive.002 on Oct 28, 2017 11:20 PM

I have commented the offending line, added the printf() function in main, right after all initializations and before the scheduler starts, and the value it returns is 80. Isn't that way more than the **MAX_PRIORITIES** defined in the config of FreeRTOS?

Actions Like • 0

Turvey.Clive.002
@ Gharibi.Zaher on Oct 28, 2017 10:18 PM

Code cannot execute from CCM RAM, so that can be a problem if any code is moved there, but the assembler doesn't understand that limit so hard to see why it would error.

Actions Like • 0

Gharibi.Zaher
@ Turvey.Clive.002 on Oct 28, 2017 10:29 PM

I don't think the code was moved to CCM, but it was a buffer used for memory allocation in the CCM region:

> #define MEM_BASE 0x10000000
> mem_TypeDef memory_pool __attribute__((at(MEM_BASE)));

Actions Like • 0

eggcar
@ Turvey.Clive.002 on Nov 6, 2017 9:15 AM

It seems that the stm32f407xx.h(or other models) constant definition is the culprit.

```
/**
  * @brief Configuration of the Cortex-M4 Processor and
Core Peripherals
  */
#define __CM4_REV                0x0001U  /*!< Core
revision r0p1                             */
```

```
#define __MPU_PRESENT             1U        /*!< STM32F4XX
provides an MPU                       */
#define __NVIC_PRIO_BITS          4U        /*!< STM32F4XX
uses 4 Bits for the Priority Levels */
#define __Vendor_SysTickConfig    0U        /*!< Set to 1
if different SysTick Config is used  */
#define __FPU_PRESENT             1U        /*!< FPU
present                                    */
```

Just remove the 'U' behind these constants, it then works well. Seems that armcc assemblier does not recognize C-format constant representation.

Actions Like • 0

Gharibi.Zaher
@ eggcar on Nov 6, 2017 12:31 PM

Thank you. I will try it if I had the same error again. For now, it's already solved as stated earlier. Like you said, it's an issue with the ARM Assembler as I have never seen this issue with gcc. I have almost an identical copy of the project built for AC6 and it compiles and works just fine!

Actions Like • 0

Gharibi.Zaher
Oct 29, 2017 10:56 AM

I believe I have just found what causes that error and my project compiles now with all the modules added without any issue. It had nothing to do with memory allocation on CCM, interrupt priorities, or anything I have thought of before. It was just an include file. One of the modules I have used from the demo package includes the main.h file, which in turn includes the stm32f4xx_hal.h file. Replacing that file with the device header file, stm32f4xx.h, has solved the issue and now the project compiles with FreeRTOS enabled and all the other sources I had initially!

Not sure if that was a permanent solution to the problem, but I'm really afraid of getting the same error again as I add more and more sources to the project. Up to this moment, everything seems to be working flawlessly!

Zaher

Actions Like • 0

lifeland
Nov 9, 2017 9:02 AM

It's a new bug of CUBEMX V4.23, and you just need remove this 'U'   #define __NVIC_PRIO_BITS   4U ,that's will be OK.

Actions Like • 0

Related Content

freertos stm32f1 lcd interface problem

stm32f103vc freertos fsmc error

STM32F1, CubeMX and FreeRTOS: undefined reference to pxCurrentTCB and vTaskSwitchContext

CubeMX Freertos C++

STM32L4 CubeMX FreeRTOS issue

Recommended Content

2018 STM32 Wish List

Configure USB Bulk IN/OUT packet size 256 bytes

STM32F769I-DISCO: No audio output from wm8994

Cosmic STM8 sprintf

Efficiently use DMA with UART RX on STM32

Incoming Links

Re: freertos stm32f1 lcd interface problem