

Bypass Chip Problems on C166 & ST10 Devices with Keil C166 Development Tools

Application Note 141
Version 4

Revised October 9, 2001 Munich, Germany

by Reinhard Keil, Keil Elektronik GmbH rk@keil.com ++49 89 456040-13

Keil Software works closely with silicon vendors to ensure support for all 166 and ST10 variants. Thus the Keil toolchain has directives to create save code for bypassing chip problems in new 166 & ST10 derivatives.

Especially when you are using new 166 or ST10 devices you should check the chip errata sheet that is available from the silicon vendor to ensure correct settings for the Keil C166 compiler. The silicon vendors publish in up-to-date errata sheets directly the required directives to bypass the chip bugs.

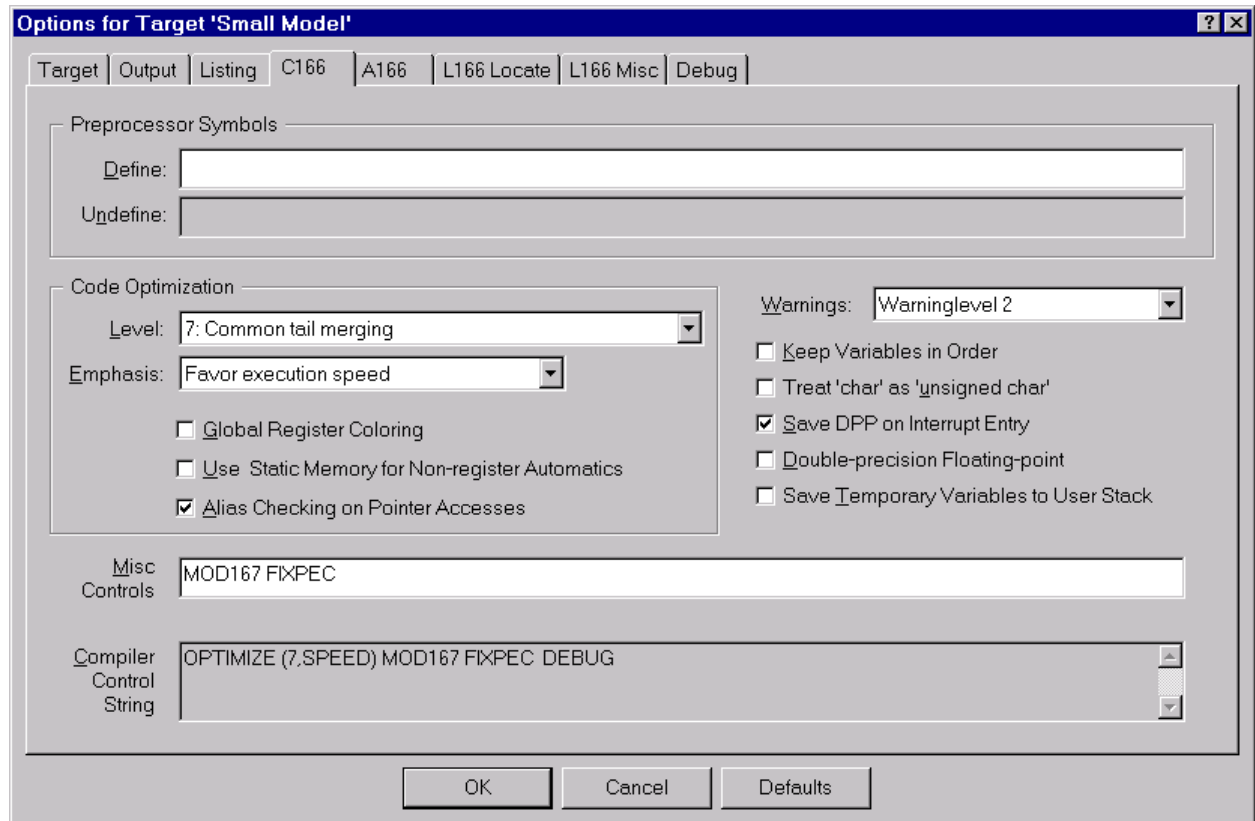
Enclosed is a list of all bypass directives found in the Keil C166 development tools.

| C166 Directive | Description |
|----------------|---|
| FIX166 | Before RETI instruction the interrupt level is set to 15. This directive was required to bypass chip problem A17 of the original 80C16x device -not needed on current device steps. |
| FIX167 | Ensure that the instruction MOV Rn.[Rm+#d16] is not at the end of EXTP/EXTS sequences. Required for early versions of the C167 device -not needed on current device steps. |
| FIXPEC | Insert NOP to avoid JMP - JMP constructs. Use this if you compile your program with OPTIMIZE(7) and if you are using the PEC. It is required to ensure correct operation of the PEC. The chip bug was present at press time of this Application Note in several devices. Check carefully the chip errata. |
| FIXROM | Avoid the MOV [Rn].mem instruction for accessing constants. Is required to bypass CPU.16 problem that is present in some FLASH variants. Check carefully the chip errata. |
| FIX32F | Some early variants of Flash devices with 256KB Flash Memory have problems with the MOV Rx,[Ry+const16] addressing. This sequences needs to be replaced with <pre>[ATOMIC #3] ;; in case of Ry == R0 ADD Ry,#const16 MOV Rx,[Ry] SUB Ry,#const16</pre> <p>Note: If you are using a device with this chip problem you need also a modified C166 run-time library. This run-time library can be found in the folder Keil\C166\LIB\FIX. You can enter this library path in <i>µVision2</i> under Project – File Extensions, Books, Enviornment – Environment – LIB Folder.</p> |
| FIXMDU | In some early variants of Flash devices the MUL/DIV instruction needs an ATOMIC #1 instruction in front of the instruction. If you enable FIXMDU the C-166 compiler inserts devices with 256KB Flash Memory have problems with the MOV Rx,[Ry+const16] addressing. This sequences needs to be replaced with <pre>[ATOMIC #3] ;; in case of Ry == R0 ADD Ry,#const16 MOV Rx,[Ry] SUB Ry,#const16</pre> <p>Note: If you are using a device with this chip problem you need also a modified C166 run-time library. This run-time library can be found in the folder Keil\C166\LIB\FIX. You can enter this library path in <i>µVision2</i> under Project – File Extensions, Books, Enviornment – Environment – LIB Folder.</p> |
| FIX272 | The FIX272 directive allows you to generate save code for the ST10R272L device. It bypasses the chip problems Kfm_BR04 and Kfm_BR05 described in the Errata Sheet dated 15. Sep. 1998 |

| C166 Directive | Description |
|-----------------|--|
| FIXBFLD | <p>The FIXBFLD directive bypasses the CPU.21 problem documented by Infineon. The FIXBFLD directive ensures that the Compiler encloses BFLDL and BFLDH instructions with ATOMIC sequences. The FIXBFLD directive inserts ATOMIC #1 instructions before each BFLDL/BFLDH instruction. If the BFLD instruction is used to access ESFR registers, the EXTR sequence is not combined with other EXTR sequences (exception: this is not the case for the SYSCON1/SYSCON2/SYSCON3 SFR's; required by the hardware for UNLOCK sequences).</p> <p>If you are using the <code>_bfd_</code> intrinsic function within <code>_atomic_ (0); _endatomic_ ();</code> blocks, the Compiler does not modify the code, since such sequences usually do not require any bypass. Please check such code blocks carefully and insert if required <code>_nop_ ()</code> calls before the <code>_bfd_ ()</code> intrinsic call.</p> |
| FIX_BR03 | <p>The FIX_BR03 directive bypasses the Kfm_BR03 CPU problem documented by ST. This problem exists only in some early versions of the ST10. Please check carefully the errata sheet, to verify if this bypass directive is required or not. The FIX_BR03 directive inserts a NOP instruction after the MAC coSTORE instruction.</p> |

Use the FIXxxx Directives in the μ Vision2 IDE

You can enter the FIX directives in the μ Vision2 IDE in the dialog box Options - C166 under Misc Controls as shown in the following.



Bypassing ST10-272 Problem BR06

To bypass the Kfm_BR06 problem of the chip (JMPS/PEC) broken use the following work-around:

- Do not enable Optimize (7)
- Change the instruction "JMP FAR main" at the end of the START167.A66 file to "CALL FAR main"

NOTE

The Keil C166 compiler will be extended to support new devices. If you are using new 16x/ST10 derivatives, check the Release Notes of the 166 tools and the chip errata sheet to make sure that you have correct C166 compiler settings.
