SIEMENS

Microcomputer Components
Technical Support Group Munich
HL DC AT

Errata Sheet

June 18, 1998 / Release 1.1

Device :	**SAB-C163-16F25F**
Stepping Code / Marking :	**ES-BB x, BB x**

The C163-**16F** is the **128 Kbyte Flash** version of the C163.

This Errata Sheet describes the functional problems and restrictions known in this step. Problem classification and numbering is performed relative to modules, where the C167 AC-step has been taken as the reference. Since most problems of earlier steps have already been fixed in this step of the C163-16F, problem numbering is not necessarily consecutive.

The SAB-C163-16F25F devices are mounted in a 100-pin Plastic Thin Quad Flat Pack (P-TQFP-100-1) package.

Note: devices which are marked as **ES-BB** are **engineering samples**. Specific test conditions/restrictions may apply to these engineering samples which may differ from the standard test conditions and specifications. These are described in a separate **Status Sheet**.

Changes from Errata Sheet Rel. **1.0** for devices with stepping code/marking **ES-BA x** to this Errata Sheet Rel. **1.0** for devices with stepping code/marking **ES-BB x, BB x:**

- Interrupted Multiply/Divide Instructions in Internal Flash (CPU.18)

Functional Problems and Limitations:

The following problems and limitations are known in this step:

Pin (Vpp/)OWE

This pin is internally not connected (NC). An internal pull-down disables the oscillator watchdog.

CPU.16: Data read access with MOVB [Rn], mem instruction to internal Flash

When the MOVB [Rn], mem instruction (opcode 0A4h) is executed, where

1. mem specifies a direct 16-bit byte operand address in the internal Flash memory,

AND

2. [Rn] points to an **even** byte address, while the contents of the word which includes the byte addressed by mem is **odd,**
   OR
   [Rn] points to an **odd** byte address, while the contents of the word which includes the byte addressed by mem is **even**

the following problem occurs:

a) when [Rn] points to **external** memory or to the **X-Peripheral** (SSP) address space, the data value which is written back is always 00h

b) when [Rn] points to the **internal** RAM or SFR/ESFR address space,
- the (correct) data value [mem] is written to [Rn]**+1**, i.e. to the **odd** byte address of the selected word in case [Rn] points to an **even** byte address,
- the (correct) data value [mem] is written to [Rn]**-1**, i.e. to the **even** byte address of the selected word in case [Rn] points to an **odd** byte address.

Workaround:
When mem is an address in internal Flash memory, substitute instruction

```
        MOVB [Rn], mem    e.g. by         MOV   Rm, #mem
                                          MOVB  [Rn], [Rm]
```

Note: the Keil C166 Compiler V3.10 has been extended by the directive FIXROM which avoids accesses to 'const' objectes via the instruction MOVB [Rn], mem.

<u>CPU.17:</u> Arithmetic Overflow by DIVLU instruction

For specific combinations of the values of the dividend (MDH,MDL) and divisor (Rn), the Overflow (V) flag in the PSW may not be set for **unsigned divide** operations, although an overflow occured.

E.g.:

```
        MDH   MDL      Rn         MDH  MDL
        F0F0  0F0Fh : F0F0h  = FFFF FFFFh, but no Overflow indicated !
                               (result with 32-bit precision: 1 0000h)
```

The same malfunction appears for the following combinations:

```
        n0n0 0n0n : n0n0
        n00n 0nn0 : n00n
        n000 000n : n000
        n0nn 0nnn : n0nn    where n means a Hex Digit between 8 ...  F
```

i.e. all operand combinations where at least the most significat bit of the dividend (MDH) and the divisor (Rn) is set.

In the cases where an overflow occurred after DIVLU, but the V flag is not set, the result in MDL is equal to FFFFh.

Workaround:

Skip execution of DIVLU in case an overflow would occur, and explicitly set V = 1.

```
E.g.:                CMP Rn, MDH
                     JMPR cc_ugt, NoOverflow ; no overflow if Rn > MDH
                     BSET V                  ; set V = 1 if overflow would occur
                     JMPR cc_uc, NoDivide    ; and skip DIVLU
      NoOverflow:    DIVLU Rn
      NoDivide: ...                ; next instruction, may evaluate correct V flag
```

Note:

- the KEIL C compiler, run time libraries and operating system RTX166 do not generate or use instruction sequences where the V flag in the PSW is tested after a DIVLU instruction.

- with the TASKING C166 compiler, for the following intrinsic functions code is generated which uses the overflow flag for minimizing or maximizing the function result after a division with a DIVLU:

```
        _div_u32u16_u16()
        _div_s32u16_s16()
        _div_s32u16_s32()
```

Consequently, an incorrect overflow flag (when clear instead of set) might affect the result of one of the above intrinsic functions but only in a situation where no correct result could be calculated anyway. These intrinsics first appeared in version 5.1r1 of the toolchain.

Libraries: not affected

CPU.18: Interrupted Multiply/Divide Instructions in internal Flash

When a multiply (MUL, MULU) or divide (DIV, DIVU, DIVL, DIVLU) instruction which is executed in internal Flash is interrupted, incorrect results may occur under the following conditions:

(1) the multiply/divide instruction and the RETI instruction of the interrupt service routine which has interrupted the multiply/divide operation are **both** located in different code segments in **internal Flash** according to the following table:

s_{RETI} s_{MD}	0	1	2
2	c	c	ok
1	ok	ok	c
0	ok	ok	c

- combinations marked as **ok** will not lead to problem
- combinations marked as **c** ('critical') will lead to a problem when the word at a specific location *ca* ('critical address') in internal Flash represents the opcode of an instruction which operates on data type **BYTE** (typically the 4LSBs of these opcodes are odd hex numbers from 1 .. 9). The critical address *ca* depends on the 16-bit intra-segment address i_{MD} of the multiply/divide instruction and the code segment i_{RETI} in which the RETI instruction is executed:

$ca = s_{RETI}:i_{MD}$	always when Flash is mapped to segment 1, or when Flash is mapped to segment 0 and $s_{RETI} = 2$
$ca = 1:i_{MD}$	when $s_{RETI} = 0$ and $i_{MD} \geq 8000h$ and Flash mapped to segment 0
$ca = 0:i_{MD}$	when $s_{RETI} = 1$ and $i_{MD} \leq 7FFEh$ and Flash mapped to segment 0

(2) the multipy/divide instruction is interrupted by a PEC **byte** data transfer

Workaround:

Avoid interrupts or PEC transfers during execution of multiply/divide instructions e.g. by placing ATOMIC #1 in front of every MULx/DIVx instruction.

KEIL offers a special version of its C compiler V3.12 with a directive FIXMDU which automatically inserts ATOMIC #1 in front of every MULx/DIVx instruction.

PWRDN.1: Execution of PWRDN Instruction while pin NMI# = high

When instruction PWRDN is executed while pin NMI# is at a high level, power down mode should not be entered, and the PWRDN instruction should be ignored. However, under the conditions described below, the PWRDN instruction may not be ignored, and no further instructions are fetched from external memory, i.e. the CPU is in a quasi-idle state. This problem will only occur in the following situations:

a) the instructions following the PWRDN instruction are located in external memory, and a **muliplexed bus** configuration **with memory tristate waitstate** (bit MTTCx = 0) is used, or

b) the instruction preceeding the PWRDN instruction **writes** to external memory or an XPeripheral (XRAM, CAN), and the instructions following the PWRDN instruction are located in external memory. In this case, the problem will occur for any bus configuration.

Note: the on-chip peripherals are still working correctly, in particular the Watchdog Timer will reset the device upon an overflow. Interrupts and PEC transfers, however, can not be processed. In case NMI# is asserted low while the device is in this quasi-idle state, power down mode is entered.

Workaround:
Ensure that no instruction which writes to external memory or an XPeripheral preceeds the PWRDN instruction, otherwise insert e.g. a NOP instruction in front of PWRDN. When a muliplexed bus with memory tristate waitstate is used, the PWRDN instruction should be executed out of internal RAM or XRAM.

Deviations from DC/AC Specification:

Test Conditions for I_{PD} (Power Down Mode Current)

The test limit has been set to 300 µA.

Test Condition for I_{ALEH} (ALE inactive current)

This parameter is only tested at Vcc = 5.0 V

Notes:

1) Pin **READY**# has an internal pullup (all C166 derivatives). This will be documented in the next revision of the Data Sheet.

2) During **reset**, the **internal pullups on P6.[4:0]** are active, independent whether the respective pins are used for CS# function after reset or not.