

---

# TN8000.06

## Technical Note

---

### ***XE8000 interleaved interrupts handling***

Author : Miguel Luis  
For further information, please contact

**XEMICS SA**  
E mail: [info@xemics.com](mailto:info@xemics.com)  
Web: [www.xemics.com](http://www.xemics.com)

## Introduction

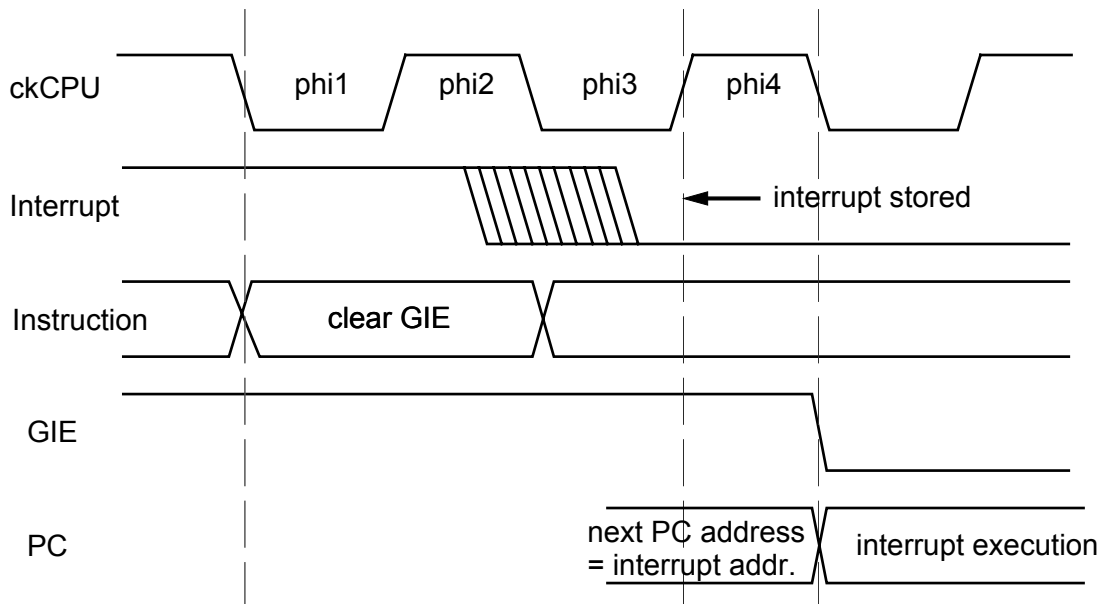
This application note concerns all the XE8000 products with software using interleaved interrupts. CoolRIDE C-compiled software does not include interleaved interrupts.

## Abstract

The clearing of the General Interrupt Enable (GIE) bit in the status register may not be executed properly whenever an interrupt occurs at the same time the clear instruction is executed. This may lead to inconsistent behavior. This application note provides a way to overcome the issue by software.

## Situation Description

The following timing diagram illustrates the issue:



Interrupt requests are taken into account at the next rising edge of the clock, provided the GIE flag is set. The clearing of the GIE flag to disable any interrupt needs some time to take effect, typically GIE will be zero at the falling edge of the phi4 phase and it is the time when the interrupt routine starts to execute. Interrupt requests may therefore be taken into account before the instruction that clears the GIE actually takes effect.

Other concerned situations are when interleaved interrupts are used or when some critical part of an interrupt code has to be protected from interrupts. In this case we have a sequence like the following:

```

_int:
    ... ; GIE is set, interrupts may occur
    _non_critical_section:
        ... ; some instructions
        CLRB stat, #5 ; clear GIE
    _critical_section:
        ... ; interrupts should be disabled
    
```

A problem occurs when another interrupt request occurs during the time the “clear GIE” instruction is executed. The interrupt request is anyway validated as the GIE flag is not yet cleared. As a result, when returning from this interrupt with the RETI instruction, the GIE flag is set. The critical section of the first interrupt is therefore no more protected and the end behavior may be different from the case when the second interrupt occurs later.

## Solution

It is suggested to wait until the GIE flag is actually cleared before handling the critical section. Here is an assembly code fragment that illustrates the workaround:

```

_int:
    SETB          stat, #5    ; enable GIE
    ...
    _noncritical_section:
    ...
    _critical_section:
    _check_GIE:
        CLRB      stat, #5    ; clear GIE
        TSTB      stat, #5    ; check GIE
        JZC       _check_GIE
    ...
    RETI

```

## C code

Note that a C program do not have interleaved interrupts, unless assembly instructions are explicitly included with the asm instruction. In that case, the workaround still applies.