

B30. Thermal Sensor May Assert SMBALERT# Incorrectly

Problem: The Mobile Pentium II processor and the Pentium II Processor Mobile Module have a thermal sensor that monitors the processor core's temperature. Please note that desktop systems could have a similar thermal device. The thermal sensor asserts SMBALERT# if the processor temperature exceeds the temperature limits set in the Alarm Threshold Registers (T_{HIGH} , T_{LOW}). It also sets the corresponding Status Register bits to identify the cause of the interrupt. Figure 1 gives one example of the how the SMBALERT# signal could be used in a system.

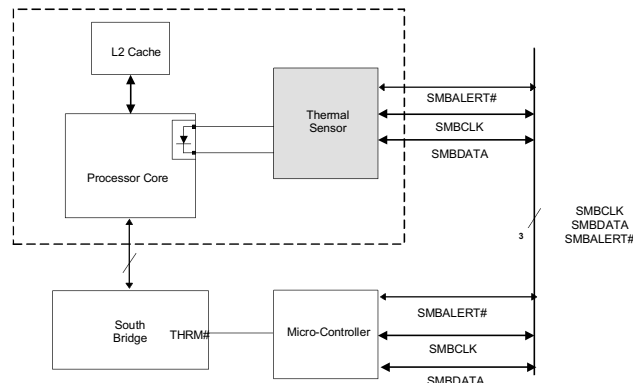


Figure 1. An Example of Microcontroller Driven Thermal Management

Under the conditions described below, the thermal sensor incorrectly generates the SMBALERT# interrupt. All of the following conditions must be met to trigger a false interrupt:

1. The thermal sensor must be in auto-convert mode.
2. The absolute value of the difference between the current temperature reading and the T_{HIGH} or T_{LOW} limit value must be less than or equal to $8^{\circ}C$.
3. The current temperature reading must be different from the previous reading.

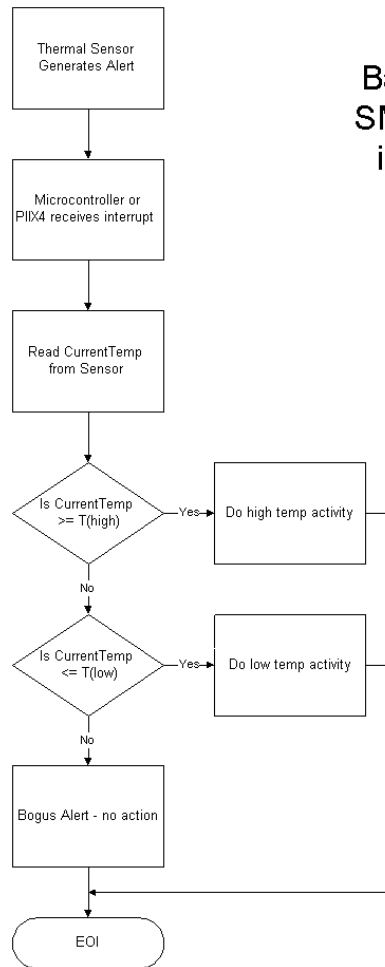
With a false assertion of SMBALERT#, the corresponding bit in the Status Register (L_{HIGH} , L_{LOW} , R_{HIGH} , and R_{LOW}) also will be incorrect.

Implication: There is no system impact from this erratum if temperature polling is used for processor thermal management. If the SMBALERT# interrupt is employed to manage processor thermal sensing, then servicing the false interrupt may result in premature system action depending on the software and hardware implementations used. The rate of the false interrupts is less than the auto-convert rate of the thermal sensor.

Workaround: Three different (mutually exclusive) workarounds are possible:

1. Before servicing an interrupt from the thermal sensor, read and compare the processor thermal reading with the threshold limits (T_{HIGH} or T_{LOW}). Figures 2 and 3 provide basic flowcharts for the implementation of this workaround in an interrupt driven system.
2. If the firmware implemented polls the Status Register only, then before taking any action, re-read the temperature register and do a comparison with the alarm threshold limits (T_{HIGH} or T_{LOW}) to determine if the value is actually still within the temperature window.

3. Use a temperature polling scheme to monitor the processor temperature.



Basic flowchart for SMBus Alert driven implementations

Figure 2. Workaround Flowchart: SMBALERT#-Driven System

Basic flowchart for
system interrupt
driven
implementations

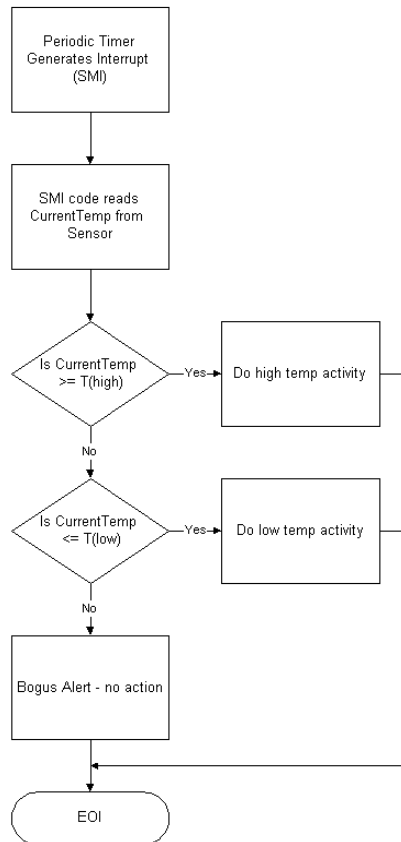


Figure 3. Workaround Flowchart: SMI#-Driven System

Status: For the steppings affected see the *Summary of Changes* at the beginning of this section.



SPECIFICATION CLARIFICATIONS

The Specification Clarifications listed in this section apply to the following documents:

- *Mobile Pentium® II Processor at 233 MHz, 266 MHz, and 300 MHz* datasheet
- *Intel® Pentium® II Processor Mobile Module : Mobile Module Connector 1 (MMC-1)* datasheet
- *Intel® Pentium® II Processor Mobile Module : Mobile Module Connector 2 (MMC-2)* datasheet
- *Mobile Pentium® II Processor in Mini-Cartridge Package at 400 MHz, 366 MHz, 333 MHz, 300PE MHz, and 266PE MHz* datasheet
- *Mobile Pentium® II Processor in Micro-PGA and BGA Packages at 400 MHz, 366 MHz, 333 MHz, 300PE MHz, and 266PE MHz* datasheet
- *Intel® Pentium® II Processor Mobile Module With On-Die Cache: Mobile Module Connector 1 (MMC-1)* datasheet
- *Intel® Pentium® II Processor Mobile Module With On-Die Cache: Mobile Module Connector (MMC-2)* datasheet
- *P6 Family of Processors Hardware Developer's Manual*

All Specification Clarifications will be incorporated into a future version of the appropriate Mobile Pentium II processor documentation.

B1. PWRGOOD Inactive Pulse Width

Footnote 10 of Table 3.11 in the *Mobile Pentium® II Processor at 233 MHz, 266 MHz, and 300 MHz* datasheet, should read as follows:

10. When driven inactive or after V_{CC} , V_{CCP} , V_{CC3} , and BCLK become stable, PWRGOOD must remain below $V_{IL,max}$ from Table 3.7 until all the voltage planes meet the voltage tolerance specifications in Table 3.4 and BCLK has met the BCLK AC specifications in Table 3.8 for at least 10 clock cycles. PWRGOOD must rise glitch-free and monotonically to 2.5 V.

B2. Thermal Sensor Setpoint

The thermal sensor in the Mobile Pentium II processor and mobile module implements the SMBALERT# signal described in the SMBus specification. SMBALERT# is always asserted when the temperature of the processor core thermal diode or the thermal sensor internal temperature exceeds either the upper or lower temperature thresholds. SMBALERT# may also be asserted if the measured temperature equals either the upper or lower threshold.



B3. Thermal Sensor Configuration Register—RUN/STOP Bit

(Refer to *Mobile Pentium® II Processor and Pentium® II Processor Mobile Module Thermal Sensor Interface Specifications*, Rev.1.0. AP-825)

The configuration register of the thermal sensor in Mobile Pentium II processor based systems controls the operating mode (Auto-convert vs. Standby) of the device. Since the processor temperature varies dynamically during normal operation, auto-convert mode should be used exclusively to monitor processor temperature. Table 1 shows the format of the configuration register. If the RUN/STOP bit is low, then the thermal sensor enters auto-conversion mode. If the RUN/STOP bit is set high, then the thermal sensor immediately stops converting and enters Standby mode. The thermal sensor will still perform temperature conversions in Standby mode when it receives a one-shot command. However, the result of a one-shot command during auto-convert mode is not guaranteed. Intel does not recommend using the one-shot command to monitor temperature when the processor is active—only auto-convert mode should be used.

Table 1. Thermal Sensor Configuration Register

Bit	Name	Reset State	Function
7 (MSB)	MASK	0	Masks SMBALERT# when high.
6	RUN/STOP	0	Standby mode control bit. If low, the device enters auto-convert mode. If high, the device immediately stops converting, and enters standby mode where the one-shot command can be performed.
5 – 0	RFU	0	Reserved for future use.

NOTE:

All RFU bits should be written as “0” and read as “don’t care” for programming purposes.