PowerPC™

Power-Managing the PowerPC 601™ CPU

Application Note AN-PPC-004

PowerPC 601 and 601v can be power-managed by entering soft-stop mode and then stopping the CPU clock.

MPRA04APU-01
Keith Braithwaite and Dale Elson
August, 1995

IBM®

Power-Managing the PowerPC 601™ CPU

Keith Braithwaite and Dale Elson

PowerPC 601 and 601v can be power-managed by entering soft stop mode and then stopping the CPU clock.

1 Introduction

A significant portion of the power consumed by a computer is dedicated to the CPU. The quiescent power consumption of both the PowerPC 601 and PowerPC 601v™ is low compared to the dynamic power consumption, which is directly proportional to the operating clock frequency. This application note presents a high level discussion of implementing 601 power management by soft stopping the 601 and then stopping the 601 PCLK_EN#. A hardware and software algorithm is described which allows automatic reduction of power during periods when the CPU is not in use. The methods described herein are suitable for both the 601 and 601v processors, both of which are refered to herein as 601.

Power management (PM) of other parts of the system can also be achieved, but these issues are beyond the scope of this application note. Likewise, multi-processor power management issues are not addressed herein. The system designer can incorporate the 601 power management information contained herein with the unique requirements of the target system to design a complete system solution.

The PM design contained in this application note was implemented on a PowerPC Reference Platform 2.0/2.1 compliant system (the system) which is similar to the IBM PowerPC 603/604 Reference Design (reference design). Using the techniques described in this note, the system achieved a 25x decrease in the power consumption of the CPU during sleep mode.

The power management strategy presented herein covers 6 major topics:

1. Introduction.
2. Detecting a system idle condition.
3. Soft stopping the 601.
4. Stopping the 601 PCLK_EN#.
5. Monitoring for wake up calls.
6. Awakening the 601.
7. Using RISCWatch with power management.

Table 1 shows the measured typical power consumption of the 601v in a power-managed system while running and while PCLK_EN# is stopped. The system supplies $Vdd_{I/O}$ = 5v and Vdd_{INT} = 2.7v. Processor speed is 120MHz and bus speed is 60MHz.

Table 1. 601v Power Consumption

Processor State	Measured 601 Power Consumption
Running	4 W
PCLK_EN# stopped	.162 W

The major hardware blocks involved in power management are shown in Figure 1. The power management controller (PMC) controls the various power management components. The PMC function can be implemented by a combination of devices. The clock control logic (CCL) contains control registers and a serial interface to the MPC970 clock generator. The MPC970 stops and restarts PCLK_EN# in an orderly manner in response to commands received via the serial interface.

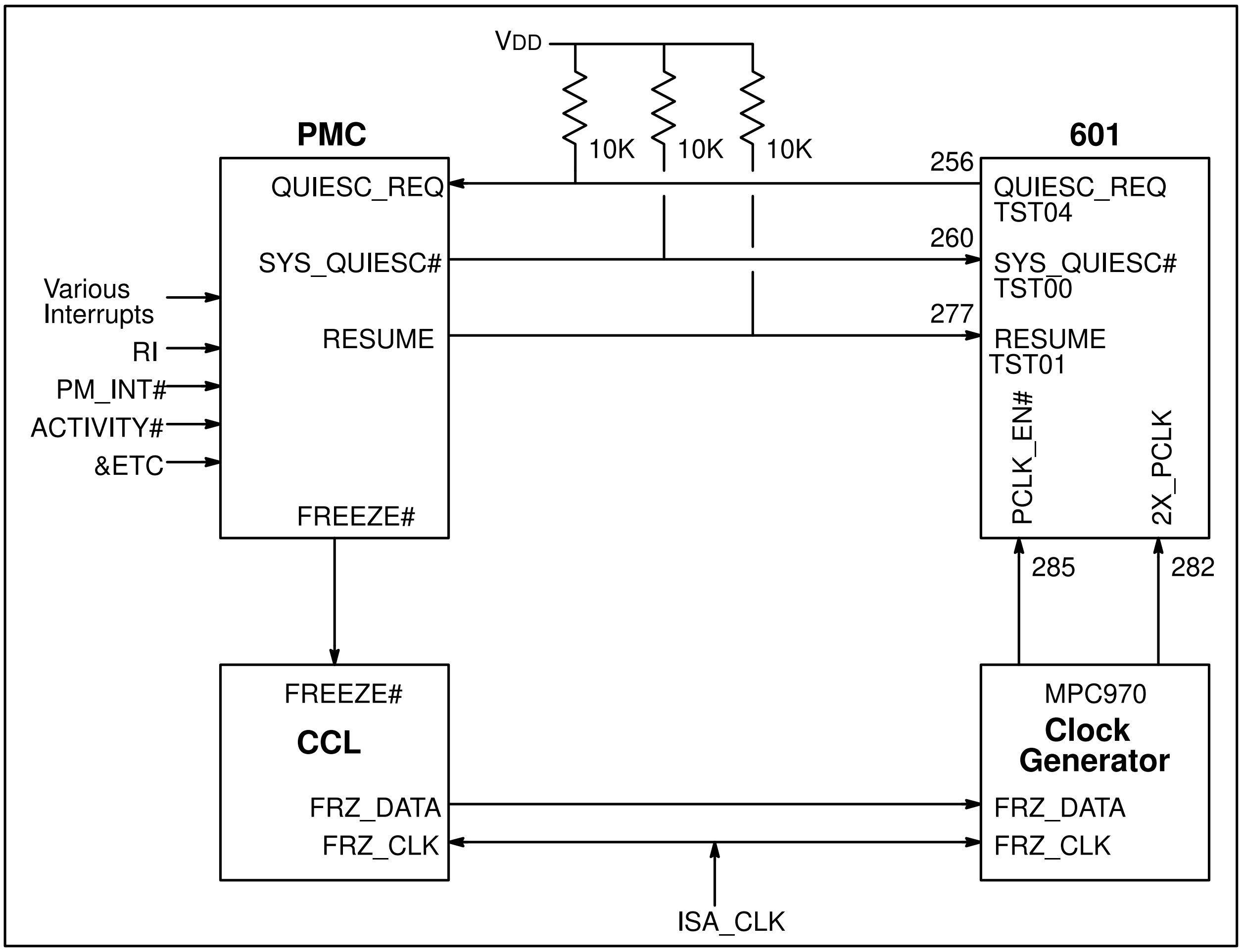

Figure 1. Power Management System Block Diagram

The general algorithm for putting the 601 into low power mode is shown in Figure 2. Once the operating system (OS) detects a system idle condition, it (if required) instructs all of the CPU and PCI bus agents not to access system memory until further notice, but to issue an interrupt if a system memory access is required.

The OS then executes the 601 soft stop algorithm, during which it asserts QUIESC_REQ. The PMC then asserts SYS_QUIESC#, causing the 601 to enter the soft stop state. At the same time, the PMC asserts FREEZE# to the CCL, and the CCL sends the freeze command to the clock generator over a serial link. The clock generator then stops PCLK_EN# to the 601. At this point the 601 is in low power mode.

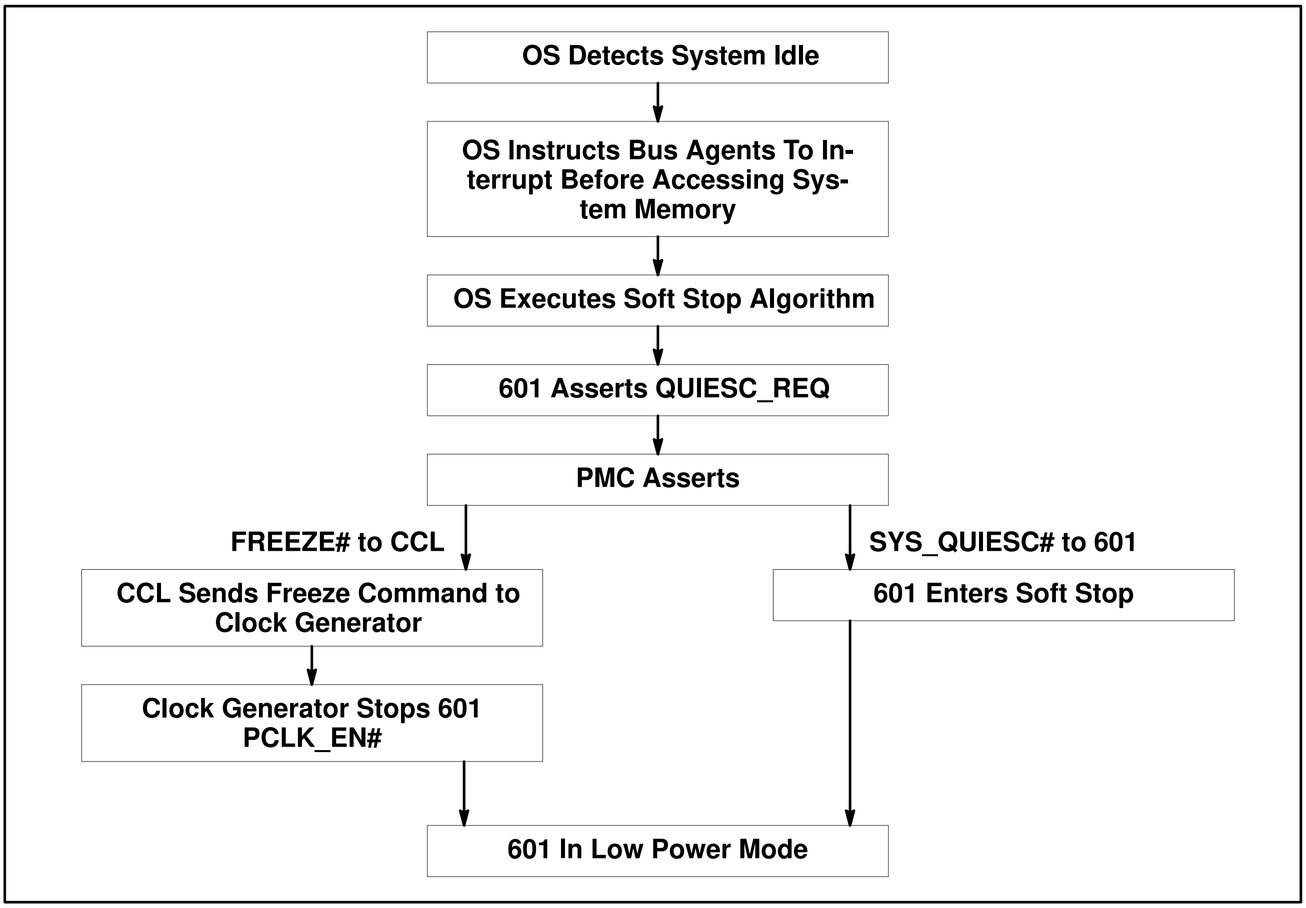

Figure 2. Putting the 601 uP Into Low Power Mode

2 Detecting System Idle

For the purposes of this application note, detecting a system idle condition is defined as an OS task. The system is considered to be idle when no useful CPU tasks are pending, and no action is currently required of the CPU. Different systems will impose differing constraints on the definition of system idle.

3 Soft Stopping the 601

Soft stopping the 601 CPU is described in the *PowerPC 601 RISC Microprocessor User's Manual* (MPR601UMU–02) in section 9.7.4. The 601 can be put into the reduced-power stop mode by either a hard stop or a soft stop. A hard stop is an immediate transition from running to stopped, and may leave instructions and bus transfers partially completed. Thus once a 601 is hard stopped, it must be hard reset before resuming operation.

A soft stop allows all outstanding instructions and bus transfers to complete. The OS initiates a soft stop by executing the series of instructions shown in Table 2. This sequence flushes CPU bus buffers, synchronizes the CPU bus agents, and causes the 601 to assert QUIESC_REQ. Note that the overall power management strategy may call for the OS to disable the assertion of INT to the 601 while the 601 is executing the soft stop algorithm code.

In response to QUIESC_REQ, the PMC quiesces all required system activity in an orderly manner, as determined by the system designer. This allows the system to complete any activities that might otherwise be adversely affected by stopping the CPU clock. The PMC then asserts SYS_QUIESC# to the 601.

On receipt of SYS_QUIESC#, the 601 enters stop mode (stops). While the 601 is stopped the internal clocks are disabled, the real time clock is not running, the decrementer is not running, and the 601 does not monitor the bus interface or INT#. The 601 does monitor RESUME, the assertion of which is the only mechanism provided to restart a stopped 601.

Table 2. Soft Stop Algorithm for the 601

Step	Description	Note
1	Execute the *isync* instruction followed by the *sync* instruction.	Allows all previous instructions and all previous data stores to complete before proceeding.
2	Load the value 0X60400000 into the HID1 register which is the debug mode register.	Sets the debug mode to *full instruction address compare* and causes a soft stop when address compare occurs.
3	Execute an *isync* instruction.	
4	Execute the *BL* instruction with parameters *$+4*.	Causes the address of the following instruction to be placed into the link register.
5	Store the contents of the link register into a general purpose register.	
6	Add the value 0x18 to this register.	
7	Execute an *isync* instruction.	
8	Move contents of the general purpose register to the HID2 register, which is the instruction address breakpoint register.	
9	Perform an *isync* instruction.	
10	Execute 2 *nop* instructions.	Address of the 2nd *nop* is the address pointed to by the HID2 register. Execution of this *nop* causes the 601 to soft stop.

4 Stopping the 601 Clock

As the PMC asserts SYS_QUIESC# to the 601, it also asserts FREEZE# to the clock control logic (CCL). In response to the assertion of FREEZE#, the CCL sends a clock control word to the Motorola MPC970 clock generator to stop the 601 clock. The Motorola MPC970 clock generator features the ability to stop 13 of the 15 clock outputs (see the Motorola MPC970 data sheet MPC970/D for more details).

4.1 Controlling the Clock Generator

The CCL sends a clock control command word to the MPC970 via a two wire serial interface. The FRZ_DATA signal carries the data, and is clocked by the rising edge of the FRZ_CLK signal. The command word contains a start bit and 13 clock control bits. The OS programs the clock control command word into the CCL registers. Other designs might hard-wire this data.

4.2 Stopping PCLK_EN#

Upon receipt of the final bit in the serial data stream, the MPC970 waits for the 601 PCLK_EN# to next enter the low state, and then halts it there. This is done in an orderly manner in order to avoid producing glitches on the clock line.

In this design, FRZ_CLK is driven by the ISA clock at about 8.77 MHz. Thus the MPC970 will not stop the 601 PCLK_EN# sooner than about 1.6us (14 bits * .114us/b) after the assertion of FREEZE# (SYS_QUIESC# to the 601). This delay ensures that the 601 has time to complete its internal clock disable operations before PCLK_EN# is stopped.

4.3 Effects of Stopping PCLK_EN#

Once the 601 is stopped, the only functional effect of stopping PCLK_EN# to the 601 is to prevent the 601 from responding to RESUME. Otherwise, the functionality of a stopped 601 is not affected by whether or not PCLK_EN# is running.

The electrical effect of stopping PCLK_EN# is to significantly further reduce the amount of power consumed by the 601, because the internal clock tree of the 601 is no longer being toggled.

4.4 PCLK_EN# vs 2X_PCLK

The input element of the internal clock tree of the 601 is functionally a D-flop, which is clocked by 2X_PCLK and whose D input is PCLK_EN#. This structure allows the 601 internal clock tree to be stopped by stopping either 2X_PLCK or PCLK_EN#. The effect on the power consumption of the 601 is identical in either case. This design stops PCLK_EN# because it is electrically the simpler option.

5 Monitoring for Wakeup Requests

The power management design was developed for systems which include a PCI bus, an ISA bus, a look-aside L2, and possibly other CPU bus agents. While the 601 is stopped, the rest of the system is allowed to keep running. This requires that the system not present any situation to the 601 that it will not be able to react to appropriately while PCLK_EN# is stopped.

In general, the PM hardware and the OS work together to cause all situations that require the CPU to wake up to be reported to the PMC via interrupt lines and special purpose system monitor lines. Upon recognizing a wakeup request, the PMC works with the CCL and the clock generator to awaken the CPU.

5.1 A System Note

Various PowerPC to PCI bridges support power management in different ways. The Motorola MPC105 PCI Bridge/Memory Controller implements several modes of power management support. For more information, see the *MPC105 PCI Bridge/Memory Controller User's Manual* (MPC105UM/AD). The power management design described in this application note implements power management externally to the bridge.

5.2 Interrupts

System interrupts are routed to the PMC, which recognizes them as wakeup requests. Keyboard, mouse, hard drive, SCSI, and Ethernet adaptors are possible interrupt sources.

The PMC also contains various timers which can be programmed to signal a wakeup request under various conditions. Note that 601 real time clock and decrementer interrupts will not occur while the 601 is stopped, because these functions are not running.

In general, there is no need to disable INT assertion to the 601 while it is stopped, since the 601 ignores INT while it is stopped. As soon as the 601 is awakened, it resumes monitoring INT normally.

5.3 Other Wakeup Request Sources

Additional hardware signals can be used to detect or modify wakeup requests:

- The serial port ring indicator signal can be monitored for COM port activity.
- Hard drive LED signal can be monitored for hard drive activity.
- The power button indicates the requested state of the power supply.
- The power management interrupt, PM_INT#.

5.4 CPU Bus Traffic

Since the 601 is unable to snoop CPU bus transactions while stopped, no CPU bus traffic is allowed while the 601 is stopped. CPU traffic can originate as:

- Snoop cycles broadcast on the CPU bus by the PCI bridge while a PCI agent is accessing system memory.
  - In some systems, the PCI bridge can not be programmed to awaken a stopped CPU before allowing PCI to memory accesses. In these systems, PCI busmasters are required to receive permission from the OS before beginning memory transactions.
  - The OS can program the PCI agents to issue an interrupt if a system memory access is required. The interrupt instructs the PMC to awaken the CPU.
  - This constraint on the PCI agents can be either left in effect constantly, or only imposed during stop mode, at the designer's option.
- CPU bus cycles initiated by another CPU bus master, such as a special purpose CPU bus device. These agents can be controlled in a manner analogous to that used with the PCI agents.

6 Awakening the 601

As shown in Figure 3, when the PMC detects a wakeup request, it first negates FREEZE#. This causes the CCL to send a serial Thaw command to the clock generator, which then restarts PCLK_EN# to the 601 in an orderly manner.

Meanwhile, the PMC waits 100uS for PCLK_EN# to start and for the 601 to stabilize, and then asserts RESUME to the 601. RESUME may be asserted asynchronously and must remain asserted for at least three PCLK_ENABLE# cycles. As soon as the 601 completes its response to RESUME (and resumes executing instructions – see Table 2), the OS is free to enable the interrupt controller and to service the interrupt that caused the wakeup request. If the interrupt was issued by a PCI busmaster to request system memory access, the OS would then enable the PCI busmaster.

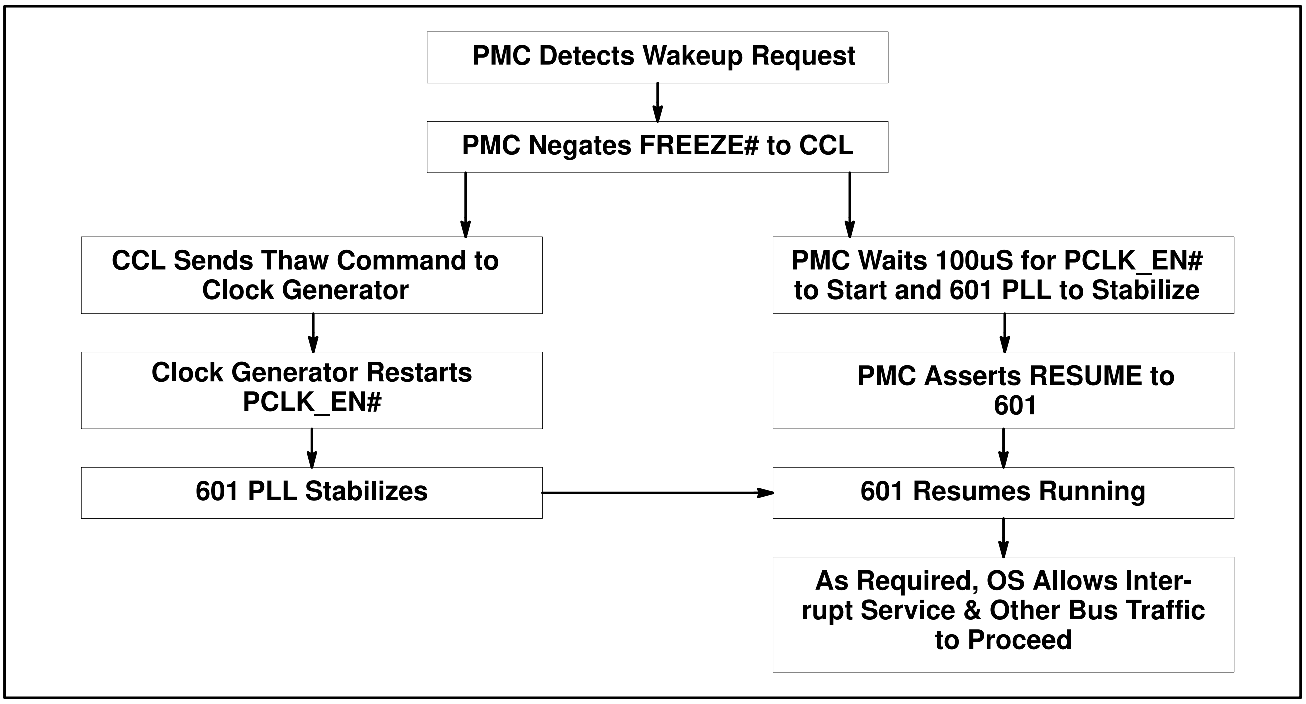

Figure 3. Awakening the 601

7 Using RISCWatch With Power Management

RISCWatch is a diagnostic and debug tool used in PowerPC systems. Use of RISCWatch requires the TST00 and TST01 pins of the 601 to be constantly low. However, the power management design uses the 601 soft stop protocol, which actively uses these two pins. To enable the use of RISCWatch and power management with the same design, add an input to the PMC which, when driven low, disables the power management features and causes the PMC to drive SYS_QUIESC# and RESUME high. This signal is shown as EN_PM (ENable Power Management) in Figure 4.

Slightly modify the plug in RISCWatch module by shorting the connector pin 9 to pin 11. While the RISCWatch connector is unplugged, power management operates normally. While the RISCWatch connector is plugged in, EN_PM is driven low, disabling power management and causing the PMC to drive SYS_QUIESC# and RESUME high.

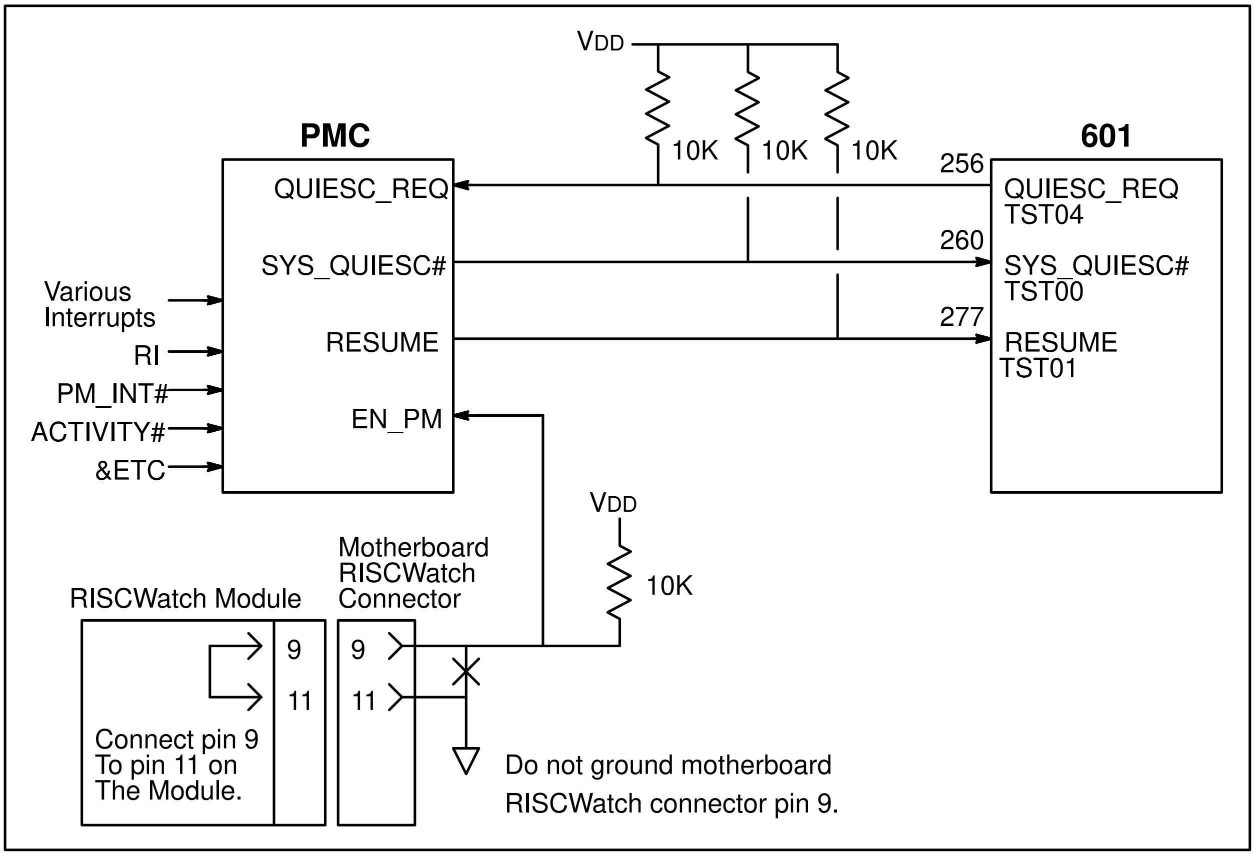

Figure 4. Modifications to Allow Use of RISCWatch With Power Management

PowerPC

USA and Canada:
IBM Microelectronics Division
1580 Route 52, Bldg. 504
Hopewell Junction, NY 12533-6531
Tel: (800) PowerPC
Fax: (800) PowerFax

Europe:
IBM
La Pompignane BP 1021
34006 Montpellier, France
Tel: (33) 6713–5757 (Français)
(33) 6713–5756 (Italiano)
Fax: (33) 6713–5750
(from Paris add 16)

Europe:
IBM
Informations Systeme GmbH
Laatzener Str. 1
30539 Hannover, Germany
Tel: (49) 511–516–3444 (English)
(49) 511–516–3555 (Deutsche)
Fax: (49) 511–516–3888

Japan:
IBM
800, Ichimiyake
Yasu–cho, Yasu–gun
Shiga–ken, Japan 520–23
Tel: (81) 775–87–4745
Fax: (81) 775–87–4735

Document Number: MPRA04APU–01