Reducing the Standby Power Consumption of the S3 State for PCs

Te HUANG††, Member, Ying-Wen BAI††, and Po-Yang HSU††, Nonmembers

SUMMARY Most research projects with respect to energy saving are trying to improve power efficiency and are using software to manage the power systems in the power on mode; but in our design, we modify the original Suspend to RAM mode-S3 state, which is the 3rd system state as defined by the ACPI specification, in order to reduce power consumption. We’ve redesigned the control circuit to save power while a PC is in the standby mode. First, we re-examine the entire circuit in the standby mode, and clarify which chip is used both to wake up the system and to turn off all unnecessary standby power previously used by the chips. Secondly, we redesign the power sequence and use an additional chip to control the system power supply, to allow a PC’s normal system’s operation to turn off the unnecessary power control chips. Third, in order to save power supply in the standby mode, we have simplified the multiple remote wake-up mechanism to control the remote boot device. The improvement shows that our design reduced power consumption to 0.21W from the original 0.56W while all the remote wake-up functions are disabled; and consumes 0.42W when using multiple remote wake-up functions. We implement the above modification from the legacy S3 state, and obtain lower power consumption. In order to distinguish the standby states, we name the modified S3 state as Deep S3 state.

key words: power control, standby power, power system management, energy saving, power consumption

1. Introduction

Modern personal computers (PC) usually maintain some wake-up functions for the user’s convenience. They consume standby power in the “System off state”, and return to the working state as soon as possible. We use the Suspend to RAM mode, rather than the System off mode, which not only can quickly recover, but also consumes less standby power [1]. To strengthen environmental protection, the international environmental agencies of all countries have established the standard currently used to saving power. One of the provisions stipulates that when PCs are in the “System off state”, the power consumption must be less than 1W, and less than 1.7W in the sleep state [2].

Many methods are available to reduce the dynamic power consumption of these systems. These methods use hardware both to reduce the clock-swing of the processor and decrease the leakage current, or software to monitor and predict the power consumption [3], [4]. If a computer system is in the idle state or does not require high performance, dynamic scaling can lower both the operation voltage and the clock rate, thereby decreasing the total leakage current to save dynamic power consumption [5]–[7]. On the other hand, reducing the static power consumption of a system not only increases the efficiency of the power supply module, but also saves the power consumption if the system is idle [8], [9]. Some experts have even considered estimating and minimizing energy leakage in an IC design at the component level by inserting control points [10], [11]. Some systems can also be controlled remotely to save the standby power supply to efficiently manage the standby power [12], [13]. The most useful method is to turn on the power supply of an electric home appliance only when we want to use it [14]–[16]. Based on this state control mechanism, we turn off the standby power supply of a system by redesigning its power circuit and sequence, until the users wake up the system by means of either a PS/2 interface, a Universal Serial Bus (USB), a Local Area Network (LAN) or a power switch [17], [18].

We should aim to reduce the environmental impact, and this includes energy consumption. The two principal international environmental protection agencies which establish the rules for energy consumption are the US Environmental Protection Agency with its “ENERGY STAR Program Requirements for Computers” [19] and the European Commission with its “Eco-Design Requirements for Energy-Using Products Directive” [20].

Table 1 shows that according to the ENERGY STAR program requirements for computers, the power consumption of desktop PCs must be less than 1W when the system is turned off.

Since January 6, 2010, according to the Energy Using Products Directive (EuP) Lot 6 for electronic home appliances with a standby power supply, which includes desktop

<table>
<thead>
<tr>
<th>Product Type</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Desktop, Integrated Desktop, and Notebook Computers</td>
<td>Off Mode: ≤1.6W, Sleep Mode: ≤1.7W, Idle State: ≤10.0W</td>
</tr>
<tr>
<td>Workstations</td>
<td>Off Mode: ≤2.0W, Sleep Mode: ≤4.0W, Idle State: ≤80.0W, Max. Power: ≤180.0W</td>
</tr>
</tbody>
</table>

Table 1 ENERGY STAR program requirements for computers

Manuscript revised March 13, 2015.
††The authors are with the Graduate Institute of Applied Science and Engineering, Fu Jen Catholic University, New Taipei City, 24205 Taiwan.
†The author is with the Department of Electrical Engineering, Fu Jen Catholic University, New Taipei City, 24205 Taiwan.

E-mail: 497598030@mail.fju.edu.tw
DOI: 10.1587/eie.98.C.952

Copyright © 2015 The Institute of Electronics, Information and Communication Engineers
PCs, the standby power consumption must be less than 1W in the “System off” state. The standby power consumption should be less than 2W, if electronic home appliances support either display, remote control or Wake-on-LAN (WOL) functions. In addition, the standby power must be reduced to 0.5W after January 6, 2013 and to 1W for electronic home appliances supporting either display, remote control or WOL functions. In order to reduce the standby power consumption, there has been much research focusing on the circuit design, on modifying the architecture of the power consumption, or on increasing the power factor to improve the efficiency of the AC-to-DC power regulator [21]-[26]. There are some other ways that can be used to improve the efficiency of the Pulse Width Modulation (PWM) of the DC-to-DC power regulator, such as by using either the Zero Voltage Switching (ZVS) or the Zero Current Switching (ZCS) methods. Both methods focus on the circuit modification to improve the ZCS and ZVS to reduce switching losses [27], [28].

Table 2 shows the comparison of the power consumption and the standby resume times of the standby states of a quad core system.

Table 2 System standby status comparison

<table>
<thead>
<tr>
<th>Status</th>
<th>S0</th>
<th>S1</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resume (sec)</td>
<td>N/A</td>
<td>&lt;1s</td>
<td>3s-5s</td>
<td>20s-40s</td>
<td>40s-90s</td>
</tr>
<tr>
<td>Power (watt)</td>
<td>65W-95W</td>
<td>20W-75W</td>
<td>2W-5W</td>
<td>1-2W</td>
<td>1W-2W</td>
</tr>
<tr>
<td>Advantage</td>
<td>N/A</td>
<td>Fastest</td>
<td>Fast</td>
<td>Saves</td>
<td>Saves</td>
</tr>
<tr>
<td>Disadvantage</td>
<td>N/A</td>
<td>Wastes Power</td>
<td>Saves Power</td>
<td>Slow</td>
<td>Slowest</td>
</tr>
</tbody>
</table>

In Sect. 2, we analyze the major chips that consume standby power and measure the power consumption of each. In Sect. 3, we summarize the results when we cut off the unnecessary power supply of the chips and redesign the power sequence to ensure that the system is able to boot normally. In Sect. 4, we show the results of the measurement. In Sect. 5, we draw the conclusion and provide a look at future possibilities.

2. Analysis and Measurement of the Power Consumption of Major Chips of a PC

2.1 Standard Power State Transition of ACPI

According to the Advanced Configuration and Power Interface (ACPI) specifications, there are five system states (Sx states) in the standby mode: S0/S1/S3/S4/S5. These states may be described as follows [29]:
S0: “System on” is a PC fully turned on power state utilized under normal working condition, when the power consumption is at its highest.
S1: “Power on Suspend” is the first standby mode. In this state, the system turns off some unnecessary peripheral devices like the monitor or the hard disk to reduce power consumption.
S3: “Suspend to RAM” is the third standby mode. In this state, the memory is still powered, although it is almost the only component in use. In this state, the OS, all applications and the opened documents are stored in the main memory, which consumes standby power to maintain these stored data.
S4: “Suspend to Disk” is the fourth standby mode. In this state, we save all the content of the main memory into the hard drive, thus preserving the state of the OS, all applications and any opened documents. The power dissipation is equal to that of a shutdown as there is no extra power needed to maintain the machine state.
S5: “System off” is the system turned off state, when the power consumption is at its lowest.

The ACPI specification has also defined five global system states (Gx states) of the entire system: G0/G1/G2/G3/G5. These states may be described as follows:
G0: The “Working State” is the normal working condition which is the same as the S0 state.
G1: The “Sleeping State” is the standby state when a computer consumes a small amount of power; it includes the S1-S4 states.
G2/G5: The “Soft off” is the shutdown state which is the same as S5 state. The system maintains the standby power for resuming the system; it consumes a minimal amount of power.
G3: The “Mechanical off” is the full power off state which is accomplished by turning off the AC mechanical switch or by removing the AC power cord from the AC outlet. The system maintains the battery power for the Real-Time Clock (RTC) only; the power consumption is zero, except RTC.

Table 3 shows that the system standby state relies on the ACPI controller and the OS. When the users set the standby mode in both the OS and the Basic Input/Output System (BIOS), the OS requests a standby command from the ACPI controller, and the ACPI manager generates the S3#, S5# and Power OK (POK) signals to control the system standby state.

Figure 1 reveals the transition in the ACPI standby states and indicates when the system initially starts up from
Table 3  System standby status signals

<table>
<thead>
<tr>
<th>Status</th>
<th>S0</th>
<th>S1</th>
<th>S3</th>
<th>S4</th>
<th>S5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal S3#</td>
<td>High</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Signal S5#</td>
<td>High</td>
<td>High</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
</tr>
<tr>
<td>Signal POK</td>
<td>High</td>
<td>High</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
</tr>
</tbody>
</table>

Fig. 1  Transition of standby states in ACPI

2.2 Standard Power Sequence of ACPI

Figure 2 shows the previous power sequence design of the S3 state. When the battery of the RTC (VBAT) has been installed, the RTC reset (RTCRST#) goes to a high level after a 200ms delay to wait for the power to stabilize, and then the system enters the initial G3 state. When the adapter plug that provides DC 19V and the 3VSB is ready, the RSMRST# goes into a high level, and then the system enters the S5 state.

When the user presses the power switch, the power switch input signal (PSIN#) becomes low, and after its de-bounce delay, the standby power control unit sends the power switch output signal (PSOUT#) to wake up the system power manager which is inside the South Bridge chip. The South Bridge chip sends the S3# Low signal. The S5# does not change and maintains both the V_RAM and the RAM_RST# to transition the power state from S0 to S3.

2.3 System Standby Power Consumption in the S3 State

The system power status manager, the standby power control unit and the remote wake-up devices consume standby power to wake up PCs from the standby state. The system power status of the system is defined by the South Bridge chip, and it provides both the S3# and the S5# signals to the standby power control unit to control the power supplies.

Figure 3 shows a typical power consumption map in
the S3 state [31]. The PWM2 and the PWM3 are voltage regulators which are used to change the adapter power (DC 19V) to both 3VSB and 5VSB for standby devices; and the PWM1 is used to control both the RAM and the CPU memory controller. First, the RAM and CPU memory controller consume 247mW, which is the highest consumption in the whole standby state. Second, the power consumption of the LAN is 115.9mW, which is one of the wake-up devices. Third, the power consumption of the South Bridge chip is 47.5mW, which is one of the consuming chips in the S3 state. The South Bridge chip integrates a lot of I/O functions, such as the USB, the LAN and the power status manager which manages the system power status. The fourth part is the remote wake-up function, which is typically implemented by the peripheral devices in the earlier design. These include the USB keyboard and the mouse, which together consume 22.8mW, and an infrared receiver (IR), which consumes 5.7mW. Fifth, the power consumption of the LED is 19mW which is an indicator of the system’s status, when in the S3 state. The wake-up devices wake the system by asserting the wake-up signal (WAKE#) to the system power status manager inside of the South Bridge chip. Finally, the standby power control unit is a combination by means of some discrete logical gate (TTL) and operational amplifier (OP-Amp.), and it consumes 11.4mW, which is less than other chips. There are some other miscellaneous circuits in the system which consumes 36.7mW. PWM1, PWM2 and PWM3 consume 25.4mW, 18.2mW and 16.6mW respectively for each; The total drained current of 19V is 29.8mA, which is approximately 566mW.

3. Design and Implementation

As the South Bridge chip, which is the power state manager, is not allowed to cut off the power supply once it becomes a power manager, we need to use a different chip to manage the power state.

3.1 Turning off the South Bridge Chip, the USB, the LAN and the IR Receiver Power Supply

We use a low-power Application-specific integrated circuit (ASIC) as the ACPI controller to manage the system power state instead of the legacy standby power control unit, and we also use the two MOSFET as a switch to turn off the system standby power (V_SB) to the South Bridge and the LAN chips; the I/O standby power (V_IO) to the USB and the IR receiver as well; but not to the ACPI controller and LED indicator. We also change the WAKE# signal from the South Bridge chip to the ACPI controller, in order to maintain the remote wake-up functions. The standby power system modification is shown by the grey boxes in Fig. 4.

3.2 Improvement of the RAM Power Circuits

We assume the Memory controller of CPU is not working in the S3 state, thus we propose to turn off the power supply of CPU to reduce the power consumption. We redesign the power supply circuit of RAM, and use the RAM_EN signal to manage the PWM1. We kept the V_RAM both to continue to maintain the system and to enter the Suspend to RAM mode. The new design uses the CPU_EN# signal to control the power supply of the CPU memory controller (V_CPU). The CPU_EN# controls the MOSFET switch to determine whether to turn the V_CPU on or off. Thus the V_RAM is separated into two power rails. The RAM is kept in the original V_RAM power rail as shown in Fig. 5. Because we use the ACPI controller to turn off the V_CPU, the memory reset signal (RAM_RST#) will not be maintained in a high state since the CPU memory controller is not operational. As the stored data in the RAM will be cleared if the RAM_RST# signal becomes low, we use a MOSFET as a switch to isolate the memory reset signal. We use the RST_EN# signal to control the MOSFET switch, and the RAM_RST# signal has been separated as DRAM_RST# for the memory side. If the RST_EN# signal is not set correctly, the data stored in the RAM will be lost. Then the...
system will not be able to resume from the S3 state. When the system goes into the S3 state, the RST\_EN\# becomes low in order to isolate the low state from the RAM\_RST\# to DRAM\_RST\#; the DRAM\_RST\# will remain high because of the pull-high resistor connected to the V\_RAM.

### 3.3 System Power State Control of the ACPI Controller

Figure 6 shows the block diagram of the ACPI controller. The ACPI controller is powered by a 3VSB power source which is a non-controlled voltage regulator (PWM3) from the 19V. The ACPI state controller begins to work when the 3VSB reaches 2.8V. It is the main control unit in the ACPI controller and monitors both the 3VSB and V\_SB power states to obtain the current system state, and it also monitors the S3\# and S5\# signals to obtain the current system state.

The power supply of the wake-up devices (LAN, IR receiver or keyboard, mouse connected through USB ports) controlled by the ACPI state controller through the power enable signals (X\_EN\#) is latched in the output buffer. The wake-up (WAKE\#) signals allow the wake-up devices to assert an interrupt event to switch the ACPI state controller into the S3 state from the Deep S3 state. The wake-up conditions are programmable through the System Management Bus (SMBUS) and are recognized in the internal battery powered SRAM [32].

The RSMRST\# asserts when the V\_SB comes and reaches more than 2.8V with a 100ms delay, and also when the system is controlled by the ACPI state controller. The RSMRST\# signal is de-asserted by the ACPI state controller while the system is in the Deep S3 state. It re-generates depending on the V\_SB reaching 2.8V from the Deep S3 to the normal S3 state.

The PSOUT\# signal comes through a de-bounce circuit as a result of turning on the PSIN\# power switch. The ACPI state controller monitors the PSIN\# signal and controls the buffer depending on the power state. The PSOUT\# buffer passes through into the normal S3 state and is disconnected in the Deep S3 state to re-regenerate the PSOUT\# signal to wake up the system.

### 3.4 Deep S3 State

After shutting down the system standby power, the power state turns into a new power state which is similar to the G3 state. But this new state is not exactly the G3 state. The ACPI controller, which is still working in this state, still requires standby power to wake up the system. We define the new power state as Deep S3 which is not defined in the ACPI specifications. Figure 7 shows the transition of the standby state of the Deep S3. All we need do is to cut off all standby power to reduce the power consumption in the Deep S3 state. In this state the RAM is still powered in order to maintain the stored data. Therefore as the unused standby devices including the power status manager of the South Bridge chip, can be turned off, the standby power of the entire system is greatly reduced [33].

We know that the system power status is dominated by the South Bridge chip, which sends both S3\# and S5\# signals to the ACPI controller. The system power status manager cannot work anymore in the Deep S3 state, because the power for the South Bridge chip has been turned off. We use the ACPI controller to replace the power status manager of the South Bridge chip and to record the system power state. The major difference is to cut off the standby power which is reached within the 4 to 6 seconds’ time limit into the Deep S3 state, and the ACPI controller maintains the power status until the next WAKE\# occur. As the Deep S3 power state which we defined is not subject to the ACPI specifications, we have to redesign the system power sequence to make the Deep S3 work as a normal system power sequence.

We have modified the power sequence as shown in Fig. 8. The power sequence is same as the original one from...
initial G3 to S5, then S0, as shown in Fig. 2. While the system state goes into the S3 state, we will trigger the timer in the ACPI controller to start counting down. When the time is up, the ACPI controller turns off the standby power of the wake-up devices. The power remains only in the ACPI controller and the VRAM, and the system turns into the Deep S3 state which we have previously defined. The modified sequence is marked as dotted lines in Fig. 8.

We also modify the wake-up sequence, as shown in Fig. 9. When the ACPI controller receives the wake-up events (PSIN# or WAKE#) in the Deep S3 state, the controller first activates the V_SB, and as a result the RSMRST# is ready. Meanwhile the system goes into the S3 state from the Deep S3 state. The ACPI controller then sends the PSOUT# to wake up the South Bridge chip, and the system resumes with the S0 after a delay of 200ms by means of the S3# signal which goes to a high level. The multiple remote wake-up functions have been disabled in the Deep S3 state, because the modified standby power scheme supports only the power switch for power-on, whilst the USB keyboard, the mouse, the LAN and the South Bridge chip are not able to wake up the system in the Deep S3 state without any power supply.

We redesign the standby power scheme and use an ACPI controller to control the USB, the LAN and the South Bridge chip power supplies respectively. Thus we can turn off the power of both the South Bridge and LAN chips which consume the most standby power. The modified signals and power rails have been marked as dotted lines in Fig. 9.

3.5 Multiple Wake-Up Support in Deep S3 State

We use the independent power enable (X_EN#) signals to control the power supply of the wake-up devices. We use the LAN_EN# and SB_EN# signals to control the MOSFET switch of the LAN and the South Bridge chip. As both the USB_EN# and the IR_EN# control the MOSFET switch of the IR receiver and the USB ports, the previous listed power supply can be turned off independently.

We modify the multiple wake-up functions which are marked as gray boxes and with dotted lines in Fig. 10. The PWM1 is for the VRAM power, and the PWM3 is for the ACPI controller; they have to be kept on in the Deep S3 state. The PWM2 is for the 5VSB devices. It can be turned off while the USB and IR receiver are disabled. Thus we add a 5VSB_EN signal to turn off the PWM2.

4. Results of Measurement

4.1 Power Sequence Measurement of Our Design

Figure 11 shows the actual power sequence control waveform when entering the Deep S3 state from the S3 state. Once the user switches the computer system into the S3 state, the PSIN# first toggles a low pulse. Then the S3#
goes to a low level when the system transitions to the legacy S3 state. Next the ACPI controller starts to count down for 6 seconds, and then it turns off the MOSFET switch to cut off the V_SB when the time is up. The system then goes into the Deep S3 state that we defined.

The S3 to Deep S3 transition delay is designed for the capacitor discharge, which can avoid the wrong power sequence while the system is transition. The working power of 3.3V (VCC3) and 5V (VCC5) must be turned off and fully discharged to ensure that the system enters into the S3 state; then the standby power (V_SB) can be turned off. Figure 12 shows the capacitor discharge waveform. When we tested some different AC/DC adaptors, we found the output bulk capacitor of different manufacturers is not the same, and thus the discharge time is not quite the same. Our test result is: in general, 2-3 seconds discharge time is enough for most AC/DC adaptors. Thus we set the discharge time longer to avoid any compatibility issue.

Figure 13 shows the wake-up power sequence of Deep S3. PSIN# toggles a low pulse first while the user presses the power switch. The ACPI controller then turns on the V_SB, and asserts the RSMRST# after 100ms delays while the V_SB prepare for the next step. The ACPI controller has a 50ms delay to ensure that the South Bridge chip is ready, and then asserts the PSOUT# to the South Bridge chip to wake up the system. The PSOUT# signal is kept low for 220ms to ensure that South Bridge chip receives the PSOUT# signal. This procedure is just like the power sequence shown in Fig. 9. It is about 140ms delay from the first PSIN# falling edge to the PSOUT# falling edge; in other words, the South Bridge chip receive the power on signal with a 140ms delay when the user presses the power switch.

We also compare the wake-up delay between the legacy S3 and the Deep S3, as shown in Fig. 14 and Fig. 13. In the legacy S3 state, the PSOUT# goes low with a 60ms de-bounce delay after the PSIN# signal; it does not need to wait for the V_SB and RSMRST# to become ready. The result is approximately 80ms more for the modified power sequence. This 80ms delay is designed to avoid any problems caused by unstable power; and compared to the S3 resume time of 3–5 seconds, it is negligible to the users.
4.2 Power Control Signals Measurement of Our Design

Figure 15 shows the actual waveform of the V_RAM, the V_CPU power supply and the S3#, and the PSIN# control signals for both the Deep S3 and the legacy S3 state.

The CPU_EN# signal controls the power supply of the V_CPU from the V_RAM. Once the user presses the power switch to change the system into the S3 state, the PSIN# toggles at a low level first, and the S3# goes to a low level when the system goes into the legacy S3 state. The ACPI controller cuts off the V_CPU power supply by setting the CPU_EN# signal at a high level. The V_RAM maintains the power supply to keep the data stored in the RAM; and cuts off the power supply to the CPU (V_CPU) to save current consumption. The system goes into the Deep S3 state that we designed.

When the user presses the power switch again to wake-up the system from the Deep S3 state, the CPU_EN# goes to a low level to turn on the V_CPU power supply. The system first goes into the legacy S3 state, and then goes into the S0 state instantly.

Figure 16 shows the actual waveform of the RAM_RST# and the DRAM_RST# signals in both the legacy S3 and in the Deep S3 state.

In the Deep S3 state, the RAM_RST# becomes low because as the V_CPU power supply was turned off, the CPU memory controller can’t continue to maintain the RAM_RST# signal state. Thus the ACPI controller cuts off the RAM_RST# to DRAM_RST# signal path by means of the RST_EN# signal. The DRAM_RST# connects back to the RAM_RST# after the V_CPU is ready. The DRAM_RST# signal control method reverts back to the CPU memory controller.

4.3 Standby Power Measurement of Our Design

Figure 17 shows the provision for measuring the standby power consumption. The power source of the 19V is from the AC/DC adaptor. The 19V connects to a high-precision digital multi-meter (DMM1) to observe the input current (I_IN) first and then to another DMM2 to observe the input voltage (V_IN). The second multi-meter then connects the 19V to the DC jack of the system. The oscilloscope is used to measure both the control signals and the related power state.

Figure 18 is the actual measurement environment of Fig. 17.

First, we set up the wake-up condition and boot the system into the OS. Second, we wait for 30 seconds after the system has booted into the OS to prevent the system from becoming unstable. Third, we allow the system to enter the Deep S3 state and wait for 10 seconds till the capacitors are discharged. Fourth, we check the input current and calculate the power consumption. Fifth, we record the measurement results into each power consumption table as a comparison. With this new implementation the user can choose any method to wake up the system. We have measured the standby power consumption in terms of all wake-up functions. A record of the test results is shown in Table 4, which summarizes the power consumption while using different wake-up methods.
Table 4  Power consumption of multiple wake-up methods

<table>
<thead>
<tr>
<th>V-CPU</th>
<th>USB Wake-up</th>
<th>LAN Wake-up</th>
<th>IR Wake-up</th>
<th>Power Switch Wake-up</th>
<th>Consumption (Watt)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Enable</td>
<td>Enable</td>
<td>Enable</td>
<td>Enable</td>
<td>0.56W</td>
</tr>
<tr>
<td>2</td>
<td>Disable</td>
<td>Enable</td>
<td>Enable</td>
<td>Enable</td>
<td>0.42W</td>
</tr>
<tr>
<td>3</td>
<td>Disable</td>
<td>Disable</td>
<td>Enable</td>
<td>Enable</td>
<td>0.35W</td>
</tr>
<tr>
<td>4</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
<td>Enable</td>
<td>0.24W</td>
</tr>
<tr>
<td>5</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
<td>Disable</td>
<td>0.21W</td>
</tr>
</tbody>
</table>

Item 1 is an original configuration whereby the system supports both the V_CPU and the USB interface by means of a keyboard and a mouse wake-up, a LAN wake-up, an IR remote controller wake-up and a power switch wake-up. This system consumes a total of 0.56W which is inclusive of every wake-up function.

Item 2 to item 5 are the improvement result of the Deep S3. The power consumption depends on the specific wake-up function that is enabled.

Item 2 is the improvement result of Fig. 5. As we turn off the power supply for the V_CPU, the power consumption is thereby reduced to 0.42W.

With item 3, we turn off the V_CPU, disable both the USB keyboard and the mouse wake-up function and turn off the power supply for the USB port as well as for the South Bridge chip power. The power consumption is now 0.35W.

With item 4, we disable and turn off the V_CPU, USB, the South Bridge chip and the LAN power supply. The power consumption is now 0.24W.

With item 5, we disable and turn off the V_CPU, USB, the South Bridge chip, the LAN, the IR receiver and PWM2, so the system allows only the power switch to wake up the system. The total power consumption is 0.21W.

In addition, we have measured other reference boards with the same South Bridge chip like ours, and we have made a graph as shown in Fig. 19 to show a comparison of the standby mode power consumption of the various designs.

As also shown in Fig. 19, on the board we used “Product A”, “Product B”, “Product C”, as well as “Our Design”, before and after improvement. We found that “Product A” will not cut off the standby power in the S3 state even if the wake-up function is disabled. As “Product B” and “Product C” support fewer wake-up functions than others, thus these products have lower standby power consumption.

In our design, we can not only individually disable the wake-up function, but also turn off each standby power individually while the wake-up function is disabled. In the meantime the user can use any wake-up function individually in our design. The power consumption will increase by 0.07W for USB devices, by 0.11W for LAN devices and by 0.01W for IR devices.

5. Conclusion

To reduce the standby power consumption of the PC, we redesigned the system standby circuit and modified the previous design of the power sequence for the legacy S3 state. We not only disabled all wake-up functions, but also cut off the system standby power via the MOSFET switches, except for the power switch, in order to obtain the lowest power consumption in the Deep S3 state. We also redesigned the system’s power sequence to ensure that the system allows the power-on function to be switched from the Deep S3 state to a normal S3 state. Moreover we redesigned the circuit to allow the ACPI controller to dominate the power of the USB, LAN and IR receiver separately.

When comparing the legacy S3 and our Deep S3 design, we use an ACPI controller to replace the standby power control unit. Then we need to add some more MOSFET as a switch to control the power supply of the remote wake-up device. The more the components, the more the cost can be seen by comparing this design to the previous design. When we compare the power consumption of the legacy standby power control unit and the ACPI controller, the ACPI controller is increased from 11.4mW to 19.6mW, due to the more logic inside. Even if the ACPI controller increases the system power by 8.2mW, the controller reduces the system power overall by 351mW.

Because our design is a modification based on the ACPI specifications, it can generally be applied to ACPI compatible platforms. PCs are currently using a standard ACPI power control method. No matter what commercial GUI OS or open source OS has been installed, our design can be used to reduce the standby power consumption of the S3 state easily. A fully utilized system boots into the OS in 3 seconds, with a standby power consumption of less than 0.21W. Our design complies with the 2010 EuP Lot6.
standard which stipulates that PC or home appliances in the standby state should have a consumption of less than 1 Watt and also complies with the even stricter 2013 EuP Lot6 that requires a power consumption of less than 0.5W in both the S3 and the S5 states.

References


Te Huang received the M.S. degree in Electrical Engineering from Fu Jen Catholic University, Taiwan, in 2007. His research focuses on PC main-board design. He is currently working toward the Ph.D. degree in Graduate Institute of Applied Science and Engineering at Fu Jen Catholic University, Taiwan.

Ying-Wen Bai received the M.S. and Ph.D. degrees in electrical engineering from Columbia University, New York, in 1991 and 1993, respectively. Between 1993 and 1995, he worked at the Institute for Information Industry. His research focuses on mobile computing and microcomputer system design. He is a professor in the Department of Electrical Engineering at Fu Jen Catholic University, Taiwan.

Po-Yang Hsu received the B.S. degree in Electrical Engineering from Ching-Yun University, Taiwan, in 2003. His research focuses on PC main-board design. He is currently working toward the M.S. degree in Electrical Engineering at Fu Jen Catholic University, Taiwan.