Engineer-to-Engineer Note EE-334

Using Blackfin® Processor Hibernate State for Low Standby Power

Contributed by Jeff Sondermeyer, Joe Beauchemin, and Hiren Desai *Rev 1 – May 5, 2008*

Introduction

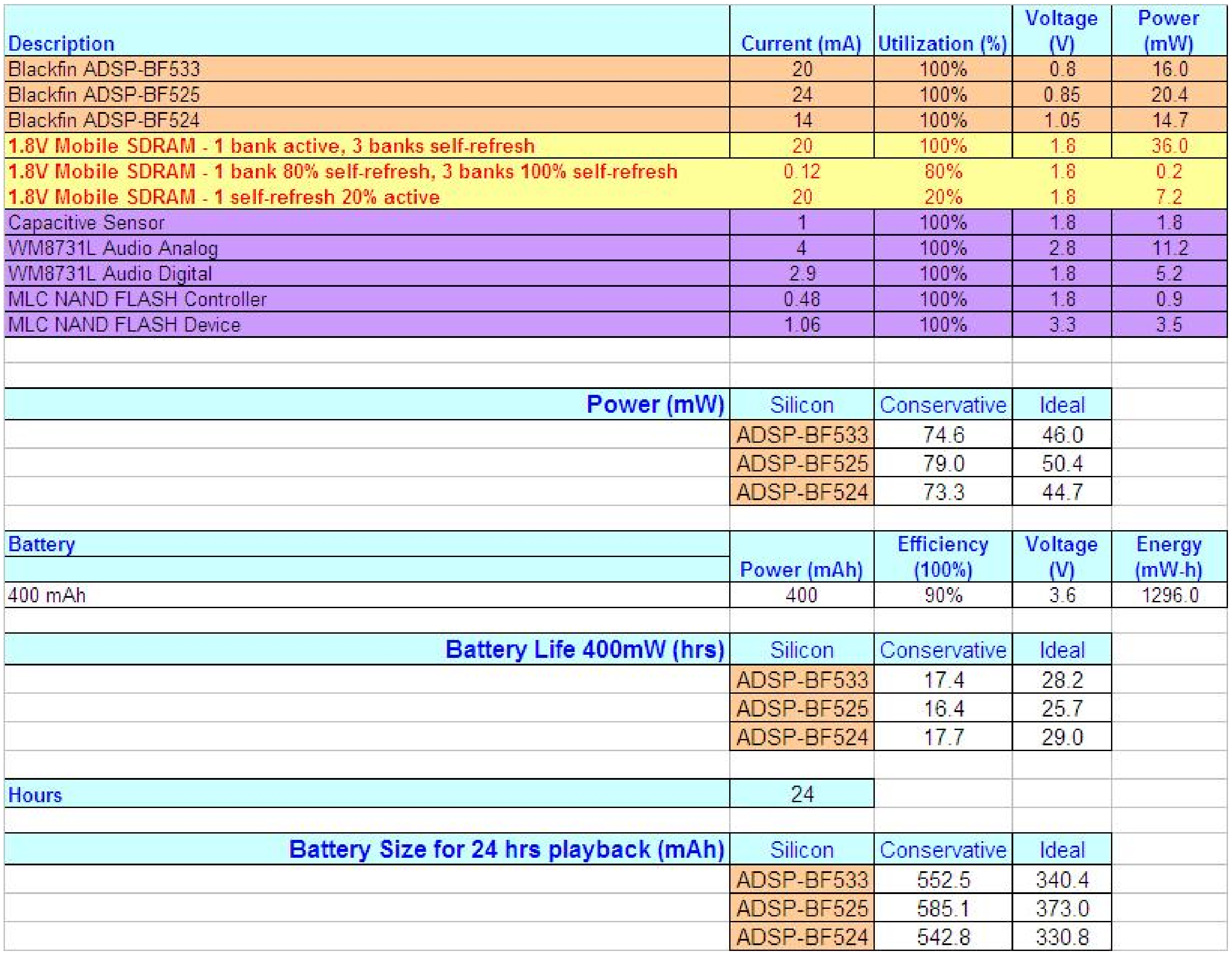
A key advantage of Blackfin® processors is low active power per MIP (~0.16 mW/MIP). However, due to the physics of high-speed (~600 MHz) memories, the processor can have standby current (static leakage current) on the order of 1-10 mA, depending on the Blackfin processor. Portable media applications have standby current requirements that are usually no greater than a few hundred µA. Depending on the Blackfin processor, a 1-10 mA standby current is roughly an order of magnitude higher than what the application calls for. Portable media players (PMPs) also typically have high MIPs requirements (≥ 400 MHz). Looking at processors available in the market today, a general observation can be made: low MIPs processors usually have low standby power, and high MIPs processors usually have high standby power. There is an indirect relationship between standby power and process technology – as process technology gets smaller, standby power goes up. So, how do you get both high MIPs with low active power and low standby power?

This EE-Note addresses this issue by using a Blackfin processor feature called the hibernate state. By definition, hibernate state provides the ability to remove core power while keeping the I/O power applied. However, when core power is removed, the processor's internal SRAM context is lost. Most Blackfin processors have a real-time clock (RTC) that can be used to wake the core from the hibernate state on a periodic basis, and several Blackfin processors allow other sources to take the processor out of the hibernate state as well. This EE-Note describes a solution for maintaining the application's context through a hibernate cycle utilizing the RTC wakeup event.

The Need for Low Standby Power

With the advent of PMPs, applications have an increasing need to operate longer on batteries, thus requiring multiple days of consistent use between charges. Adding to this challenge, the form factor of these devices is shrinking, forcing batteries to be smaller (i.e., the battery's mA-h rating is decreasing). For example, a portable MP3 player (see Figure 1) would like 24 hours of playing time from a 400 mA-h battery on a single charge. See Table 1 for a typical power breakdown for an MP3 player.

Not only is active power important for these applications, but standby power is also critical to battery life. For the sake of this discussion, *standby power* is the power required to keep the processor in a ready standby state. The processor will spend a majority of its time in this state and should be able to respond to some user interaction (such as a push-button) and resume normal operation.



Description	Current (mA)	Utilization (%)	Voltage (V)	Power (mW)
Blackfin ADSP-BF533	20	100%	0.8	16.0
Blackfin ADSP-BF525	24	100%	0.85	20.4
Blackfin ADSP-BF524	14	100%	1.05	14.7
1.8V Mobile SDRAM - 1 bank active, 3 banks self-refresh	20	100%	1.8	36.0
1.8V Mobile SDRAM - 1 bank 80% self-refresh, 3 banks 100% self-refresh	0.12	80%	1.8	0.2
1.8V Mobile SDRAM - 1 self-refresh 20% active	20	20%	1.8	7.2
Capacitive Sensor	1	100%	1.8	1.8
WM8731L Audio Analog	4	100%	2.8	11.2
WM8731L Audio Digital	2.9	100%	1.8	5.2
MLC NAND FLASH Controller	0.48	100%	1.8	0.9
MLC NAND FLASH Device	1.06	100%	3.3	3.5

Power (mW)	Silicon	Conservative	Ideal
	ADSP-BF533	74.6	46.0
	ADSP-BF525	79.0	50.4
	ADSP-BF524	73.3	44.7

Battery	Power (mAh)	Efficiency (100%)	Voltage (V)	Energy (mW-h)
400 mAh	400	90%	3.6	1296.0

Battery Life 400mW (hrs)	Silicon	Conservative	Ideal
	ADSP-BF533	17.4	28.2
	ADSP-BF525	16.4	25.7
	ADSP-BF524	17.7	29.0

Hours	24

Battery Size for 24 hrs playback (mAh)	Silicon	Conservative	Ideal
	ADSP-BF533	552.5	340.4
	ADSP-BF525	585.1	373.0
	ADSP-BF524	542.8	330.8

Table 1. ADSP-BF53x and ADSP-BF52x power for a MP3 player

The challenge is in maintaining the application's context while providing the ability to quickly recover from standby state. This solution can be applied to any portable application that has these requirements:

- High MIPs (≥ 400 MHz), low active power (≤ 0.16 mW/MIP), and low standby power (≤300 µW).
- Adjustable core frequency and/or voltage for maximum task efficiency. Blackfin processors allow dynamic changes to core/system clocks and voltage (in 50 mV increments) in software.
- High efficiency regulator(s) (≥ 90%).
- Interface to 1.8 V low-power peripherals: flash, SDRAM, audio codecs, etc.
- Quick response to user interface commands and/or peripheral wakeups (≤ 4 msec).
- Support for multi-threaded RTOS/OS environments or standalone mode of operation.
- Boot from serial EEPROM/flash (parallel NOR flash is too large and costs too much).
- Periodic wakeup from an RTC (~100 msec).

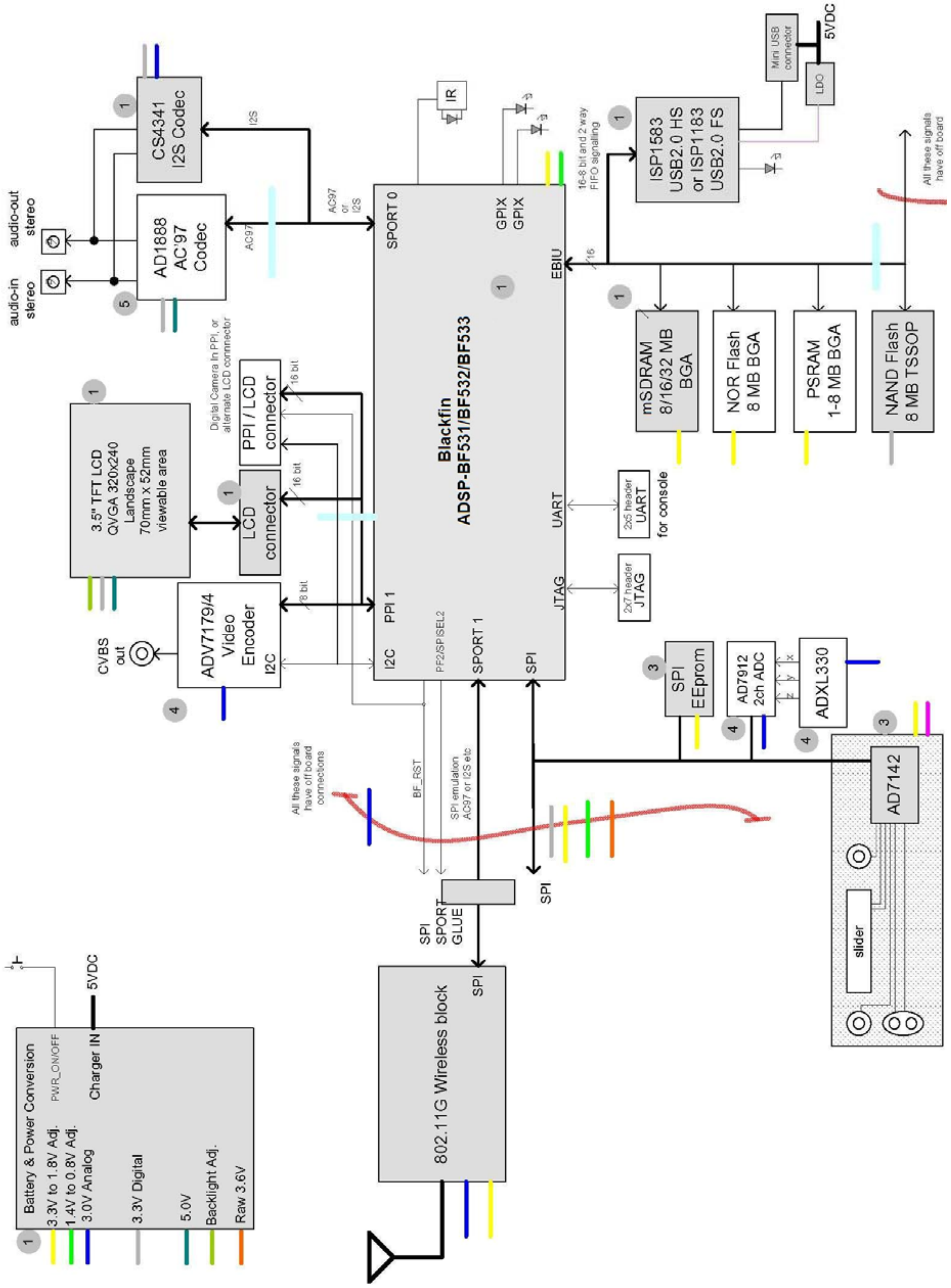


Figure 1. ADSP-BF531 portable media reference design

Using Hibernate State

Hibernate state on Blackfin processors is achieved by shutting off the internal core supply, which disables both the core clock (`CCLK`) and system clock (`SCLK`). When the core is powered down, the internal supply voltage (`VDDINT`) is set to 0 V, which results in the internal state of the processor being lost. Therefore, any critical information stored internally (e.g., memory contents, register contents, etc.) must be written to a non-volatile storage device prior to removing power. In the context of this EE-Note, this non-volatile memory is mSDRAM.

Removing V_{DDINT} eliminates most of the leakage currents from the processor, thus drastically reducing power dissipation in the system. The mSDRAM memory (L3) is placed into self-refresh mode during hibernate state, which preserves its contents. On ADSP-BF531/BF532/BF533 Blackfin processors, hibernate state can be exited by asserting the `/RESET` pin or via a Real-Time Clock (RTC) wakeup event, which then causes a hardware reset to occur.

An optimal Blackfin solution, which is particularly useful for the PMP market, has several advantages:

- With the Blackfin processor in hibernate state, it draws ~50 µA of current.
- External mSDRAM (1.8V) has very low standby current while in self-refresh mode. For example, when the Micron MT48H4M16LF - 1M x 16 x 4-bank SDRAM device[1] is in self-refresh mode with four banks open, the current drawn is ~100 µA.
- It quickly restores content (warm reboot) to the processor's internal memory from mSDRAM once core power is turned on (~96 kB in less than 2 msec). A slower cold reboot is not required.

Hardware and software requirements must be considered when implementing such a system, as described in the following sections.

Software Considerations

Two VisualDSP++® examples are used to demonstrate the concepts discussed in this EE-Note. One example runs on the ADSP-BF533 EZ-KIT Lite® evaluation board. The other example was used to prove that full context save/restore around use of hibernate state works with the additional hardware necessary for a portable media application. A PMP reference design that utilizes 1.8 V mSDRAM and a high-efficiency external core voltage regulator was used as the hardware platform. This portable media reference design (Figure 1) is not publicly available, but all the pertinent hardware details are discussed herein.

Both examples were developed using VisualDSP++ 4.5. Because the assumption was made that portable applications prefer to utilize a multi-threaded RTOS/OS environment, the VisualDSP++ Kernel (VDK) is used. The complete projects, `BF533_EZKIT_Hibernation_Example.dpj` and `PMP_Hibernation_Example.dpj`, are included in the associated `.ZIP` file.

The `BF533_EZKIT_Hibernation_Example.ldr` image must be programmed into the flash on the ADSP-BF533 EZ-KIT Lite board. An initialization block (`init_code.dxe`) is required to restore context coming out of hibernate state.

Code Overview

The following is an overview of the code operation. The code implements four threads, one interrupt service routine, and two semaphores:

Threads

1. `Main` (priority 5) – Configures the peripherals and creates the `Hibernate` and `BlinkLED` threads.
2. `BlinkLED` (priority 6) – Simply blinks LED4 15 times on the ADSP-BF533 EZ-KIT Lite board.
3. `Hibernate` (priority 7) – Shuts down peripherals, saves the system state into SDRAM, and puts the processor into hibernate state. Before the thread goes into hibernate state, the RTC is configured and SDRAM is placed into self-refresh mode. Upon wakeup, the processor returns to this thread and the peripherals are reconfigured.
4. `Idle` (lowest priority thread) – default VDK thread. The processor executes this thread when there in nothing to do.

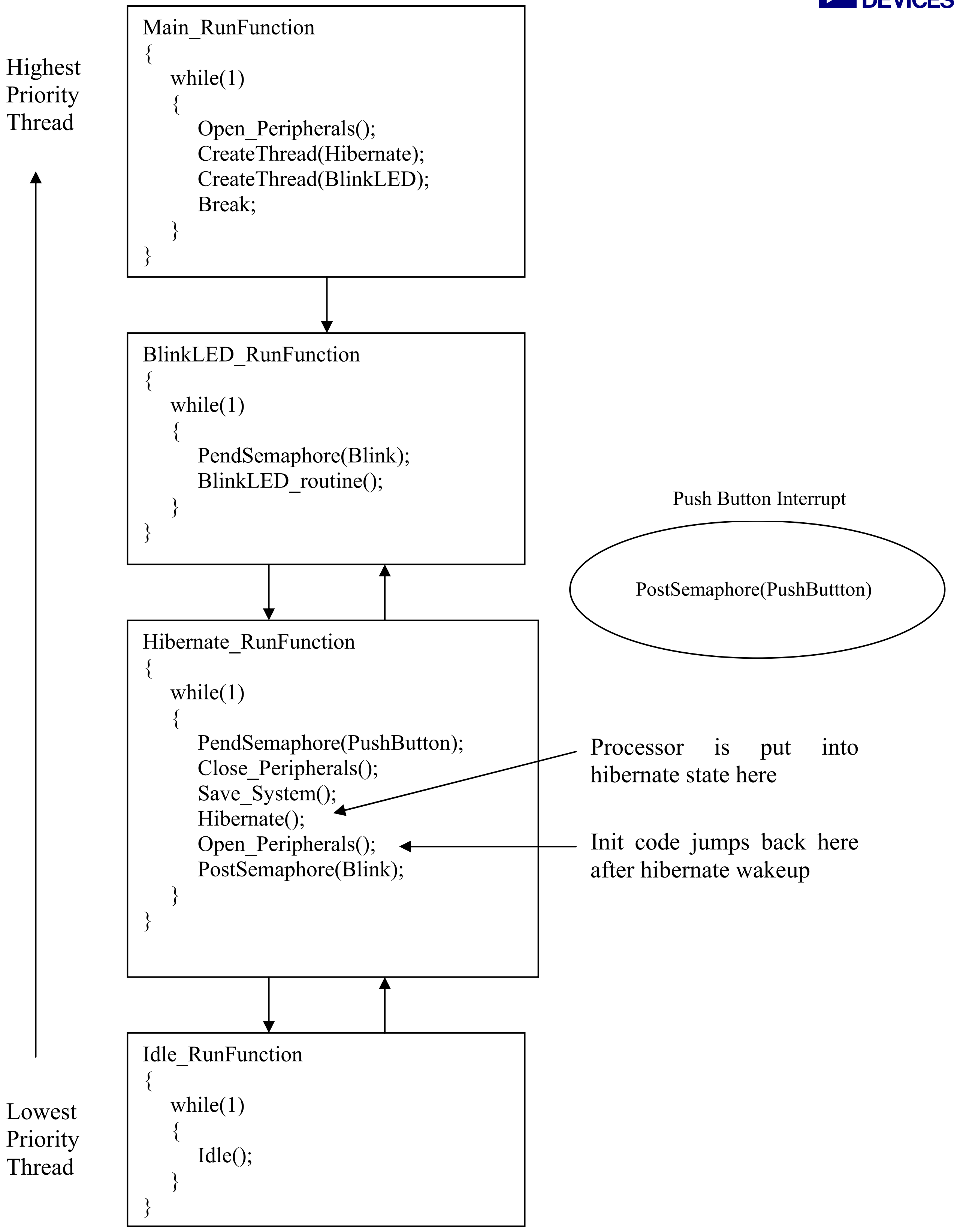
Interrupts

1. Hibernation – A push button is used to put the processor into hibernate state. Push button SW6 (PF10) is set as an input. When SW6 is pressed, the processor vectors to the `EVT_IVG9_Entry` interrupt handler (found in `Hibernate_IVG9.c`). This interrupt service routine simply posts the `PushButton` semaphore.

Semaphores

1. `PushButton` – When the user presses SW6 (PF10), this semaphore is posted. The `Hibernate` thread pends on this semaphore upon entry. If the semaphore is available (user has pushed SW6), the thread continues operation and the processor is put into hibernate state. If the semaphore is not available (user has not pushed SW6), the `Hibernate` thread blocks and the scheduler places the processor into the `Idle` thread.
2. `Blink` – This semaphore is posted at the end of the `Hibernate` thread when the processor wakes up from hibernate state. The `Blink` thread pends on this semaphore before going into the blink routine. If the semaphore is available, LED4 blinks 15 times. If the semaphore is not available, the `Blink` thread blocks and the scheduler places the processor in the `Hibernate` thread.

The code operation is shown in Figure 2.


Highest Priority Thread
Push Button Interrupt
PostSemaphore(PushButtton)
Processor is put into hibernate state here
Init code jumps back here after hibernate wakeup
Lowest Priority Thread

```
Main_RunFunction
{
    while(1)
    {
        Open_Peripherals();
        CreateThread(Hibernate);
        CreateThread(BlinkLED);
        Break;
    }
}
```

```
BlinkLED_RunFunction
{
    while(1)
    {
        PendSemaphore(Blink);
        BlinkLED_routine();
    }
}
```

```
Hibernate_RunFunction
{
    while(1)
    {
        PendSemaphore(PushButton);
        Close_Peripherals();
        Save_System();
        Hibernate();
        Open_Peripherals();
        PostSemaphore(Blink);
    }
}
```

```
Idle_RunFunction
{
    while(1)
    {
        Idle();
    }
}
```

Figure 2. Code operation

SDRAM Usage

Before the processor is placed into hibernate state, the contents of on-chip memory are saved into SDRAM memory. Table 2 shows what contents are being saved, where they are saved in SDRAM memory, and the method used to store the data.

On-Chip Memory Resources	**SDRAM Address**	**Method**
Key (value = 0x12345678) - used by Init Code	0x03FEC	Core R/W
Stack Pointer (SP)	0x03FF0	Core R/W
Frame Pointer (FP)	0x03FF4	Core R/W
L3 to L1 Descriptor List - used by Init Code	0x03FF8	Core R/W
Return Address (RETS) - used by Init Code	0x03FFC	Core R/W
L1 Instruction Memory (0xFFA00000)	0x04000	MemDMA
L1 Data Bank A (0xFF800000)	0x0C000	MemDMA
L1 Data Bank B (0xFF900000)	0x14000	MemDMA
Scratchpad (0xFFB00000)	0x1C000	Core R/W
Event Vector Table	0x1D000	Core R/W
IMASK	0x1D010	Core R/W

Table 2. Memory resources saved before hibernate

RTC Wakeup from Hibernate

Before the processor is put into hibernate state, the RTC is configured with the stopwatch feature enabled to generate an interrupt. The RTC is clocked by a 32.768 kHz crystal, and the stopwatch is set to 80/32768 (~2.44 ms). After the 2.44 ms timer expires, the RTC invokes an internal reset, which brings the part back out of hibernate state and forces the regulator to re-apply voltage to the core and L1 memory.

The RTC wakeup was only tested in the `BF533_EZKIT_Hibernation_Example.dpj` project.

Upon a hibernate wake-up event, the processor boots depending on the state of the `BMODE` pins. In this example, an initialization block (`init_code.dxe`) is pre-pended to the boot stream, which is explained in *ADSP-BF533 Blackfin Booting Process (EE- 240)*[2]. The init code configures SDRAM memory and determines whether the processor is coming out of hibernate state. It does this by checking location 0x3FEC in SDRAM memory, where `Key` is stored. If the 0x3FEC location does not contain the value of 0x12345678, the processor is coming out of a non-hibernate hardware reset (cold start) and a full boot is required. For a full boot, the initialization code simply restores the context and returns back to the boot ROM for the full boot of the application. If the key is set to 0x12345678, the processor is coming out of hibernate (warm start) and a full boot is not required. In this case, the init code uses the pointer to the list of DMA descriptors at location 0x3FF8 to set up a memDMA transfer from L3 memory, mSDRAM, back to L1 memory. It also restores the rest of the on-chip memory resources listed in Table 2.

The linker description file (`.ldf`) used to generate the memory map for the init code must place this code at the end of the L1 code space (i.e., 0xFFA14000 minus the init code size). This is done to ensure that when the run-time code is DMAed back to location 0xFFA00000 in L1 from the mSDRAM, it does not overwrite the code that is performing that task.

Finally, the init code jumps back to the `RETS` address stored at 0x3FFC, which takes the processor back to the `Hibernate` thread (the last thread running before the processor was put into hibernate state). This initialization code is listed in Appendix A.

Wakeup times will vary, depending on how much memory content needs to be restored. Table 3 lists various L1 memory restores and their corresponding wakeup times.

Amount of L1 Memory Saved/Restored	**Wakeup Time**
8 kBytes of L1 Instruction 8 kBytes of L1 Data Bank B	318 µs
56 kBytes of L1 Instruction 16 kBytes of L1 Data Bank A 24 kBytes of L1 Data Bank B	1.01 ms
32 kBytes of L1 Instruction 32 kBytes of L1 Data Bank A 32 kBytes of L1 Data Bank B	1.81 ms

Table 3. Wakeup times after hibernate state

Wakeup times were measured from the time voltage was applied to the `VROUT` pin (pin 4 of the U32 MOSFET on the ADSP-BF533 EZ-KIT Lite board) until all the on-chip memory resources were restored (right before the `Hibernate` thread is re-entered). This period includes the time voltage was applied (which includes the RTC-invoked reset), the time for the boot ROM to load/execute the initialization code, and the time required for on-chip memory to be restored by the initialization code.

Hardware Considerations

For portable applications, several common hardware blocks are necessary to implement a full context save/restore around use of the hibernate state. Note that not all Blackfin processors require *all* the external circuits discussed here. As the Blackfin processor family matures, several advancements have been made to incorporate more sophisticated hibernate functionality. Refer to the appropriate *Hardware Reference* manual for your Blackfin device to understand these advanced features. Let's take a look at the major subsystems used in conjunction with hibernate. Specifically, we will start with a review of three external buck regulator design options.

Option #1 - External Variable Buck Design with No RTC Wakeup

Blackfin processors have a built-in buck regulator to supply V_{DDINT}. In less power-sensitive applications, this buck regulator and the associated external components minimize the bill of materials. However, as seen in PMP applications, power efficiency is critical. The Blackfin regulator has an efficiency of ~75%, and it is not supported when V_{DDEXT} = 1.8 V. So, for low-power systems requiring V_{DDEXT} = 1.8 V and ~90%

efficiency from the regulator design, an external regulator must be used. The structure of the internal voltage regulator in Blackfin processors is based on a simple voltage controller. It does not include current-limiting functionality, synchronous rectification, or soft start capability. An external buck design can be used in conjunction with the Blackfin processor internal voltage controller to:

- Improve dynamic performance,
- Improve efficiency,
- Improve stability,
- Reduce PCB area required,
- Reduce rating required for components, and
- Allow variable voltage control via software.

This design allows software to write the `VR_CTL` register to utilize the hibernate state and to adjust V_{DDINT} in 100 mV intervals. However, the internal voltage regulator controller will not function if $V_{DDEXT} < 2.5$ V, and the RTC cannot be used to wake the core from the hibernate state.

This topic has already been covered in a previous paper, *Using an External Switching Regulators to Supply an Adjustable Internal Voltage to Blackfin Processors*[3], which is included in the `.ZIP` file associated with this EE-Note. This design uses the `VROUT` outputs from the Blackfin processor's on-chip regulator controller, so core voltage is software-adjustable in ~100-mV steps. See Figure 3 for the required external circuit.

Option #2 – External Fixed Buck Design with RTC Wakeup

If the Blackfin RTC needs to wake the core from hibernate state, this external buck design may be the preferred choice. Typically, a portable system must be woken up quickly (≤ 10 msec) at a predetermined interval, perform a specific task, and then must go back to the low-power standby state. If the RTC is used to periodically wake the core, a signal must alert the external buck regulator to restore power to the core. One signal that can be used to accomplish this task is common among all Blackfin processors, `VROUT`. Typically, this signal is generated by the internal voltage controller and is used with other passive components to switch the external FET and drive V_{DDINT}. However, in this case, a transition on `VROUT` is used to alert an external regulator to reapply power to the core.

An inexpensive circuit is shown in Figure 4 to accomplish this task. This topic is described in detail in *External Circuitry for Hibernate Wakeup Function on Blackfin Processor with a Fixed Voltage External Voltage Regulator*[4], which is included in the associated `.ZIP` file.

This design allows an RTC wakeup to take the processor out of the hibernate state and works even when $V_{DDEXT} < 2.5$ V. However, the internal voltage regulator controller is not programmable, meaning that V_{DDINT} is fixed.

In the circuit in Figure 4, the Blackfin `GPIO` pin (PF6) is used to initiate the hibernate process. Therefore, software must write a 1 to PF6 when entering hibernate state and write a 0 upon exiting hibernate state.

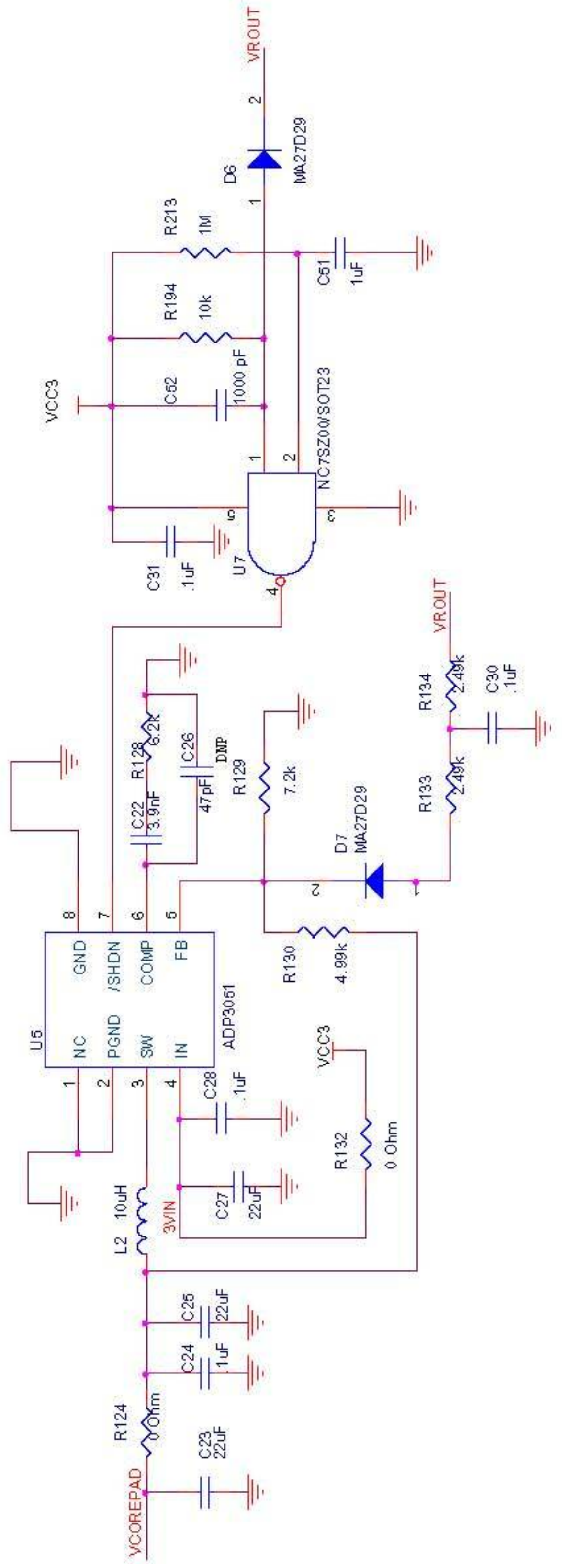


Figure 3. External variable buck design with no RTC wakeup

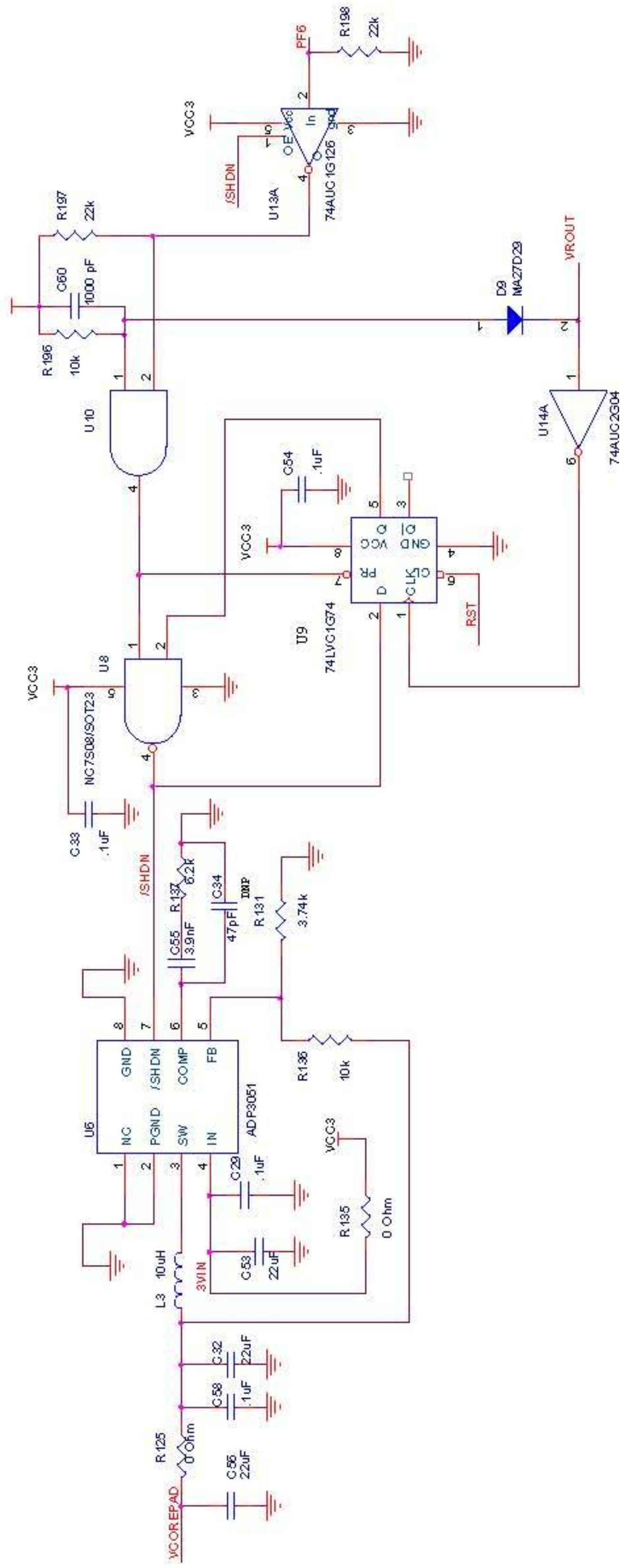
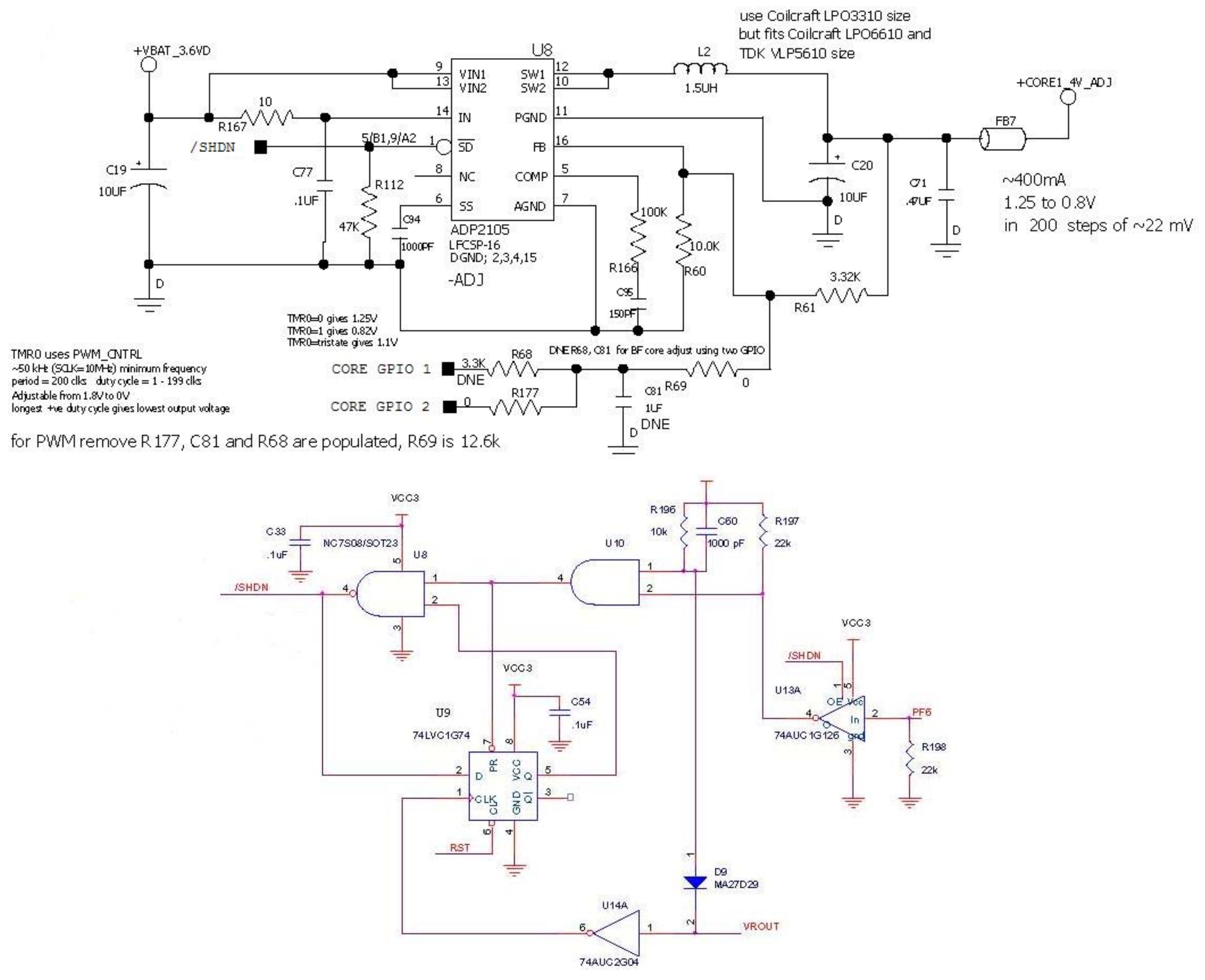


Figure 4. External fixed buck design with RTC wakeup

Option #3 – External Variable Buck Design with RTC Wakeup

Features from Option #1 and Option #2 can be combined to get an optimal solution. Figure 5 shows an ADP2105 buck regulator along with the logic in Figure 4 that allows for a RTC wakeup. This solution allows four core voltage settings using two GPIOs. Or, `Core_GPIO_1` in Figure 5 can be connected to a Blackfin timer output in PWM mode to provide unlimited voltage steps.

This design allows an RTC wakeup to wake the external buck regulator after hibernate state and works when V_{DDEXT} is 3.3 V, 2.5 V, or 1.8 V. It also provides variable core voltage adjustments in 100-mV intervals in software. However, the internal voltage regulator controller is only programmable to four discrete values using two GPIO pins.



use Coilcraft LPO3310 size but fits Coilcraft LPO6610 and TDK VLP5610 size
+VBAT_3.6VD
U8
ADP2105
LFCSP-16
DGND; 2,3,4,15
-ADJ
L2
1.5UH
+CORE1_4V_ADJ
FB7
~400mA
1.25 to 0.8V
in 200 steps of ~22 mV
TMR0=0 gives 1.25V
TMR0=1 gives 0.82V
TMR0=tristate gives 1.1V
TMR0 uses PWM_CNTRL
~50 kHz (SCLK=10MHz) minimum frequency
period = 200 clks duty cycle = 1 - 199 clks
Adjustable from 1.8V to 0V
longest +ve duty cycle gives lowest output voltage
DNE R68, C81 for BF core adjust using two GPIO
CORE GPIO 1
CORE GPIO 2
for PWM remove R177, C81 and R68 are populated, R69 is 12.6k
/SHDN
VCC3
NC7S08/SOT23
U10
U9
74LVC1G74
U13A
74AUC1G126
PF6
RST
D9
MA27D29
U14A
74AUC2G04
VROUT

Figure 5. External Variable Buck Design with RTC wakeup

Power On Reset – Core Only

Once the core power is stabilized, a core reset must be applied, which then causes the processor to boot. This requirement applies to all three core buck designs above.

For this core only, power-on reset (POR) is required for hibernate state and is in addition to the system POR.

The vast majority of power on reset (POR) chips requires a power supply ≥ 1.8 V. In this case, once the Blackfin processor V_{DDINT} reaches a stable ~1.1 V, a clean core reset must be applied (`/RESET` must be asserted low for a minimum of 275 nsec, per the *ADSP-BF531/ADSP-BF532/ADSP-BF533 Blackfin Embedded Processor Data Sheet* [4]). Thus, a special POR is needed, which will allow an outboard adjustable voltage reference and an adjustable reset timeout (the smallest is typically 1 msec). The Maxim MAX6896 POR chip (see Figure 6) can provide this functionality. The Analog Devices ADP2105 buck regulator has a soft-start feature that allows the product to produce a stable Vout in a minimum of 750 µsec. So, the minimal startup time from the application of core power to actually booting code is:

750 µsec (for the buck to stabilize) + 1 msec (minimum for the reset pulse width) = 1.75 msec

Obviously, as stated earlier, depending on how much context is being saved and restored, this takes an additional 2 msec. Thus, there is a 4 msec startup delay (total) from the initial application of power to when full context is restored and code is running.

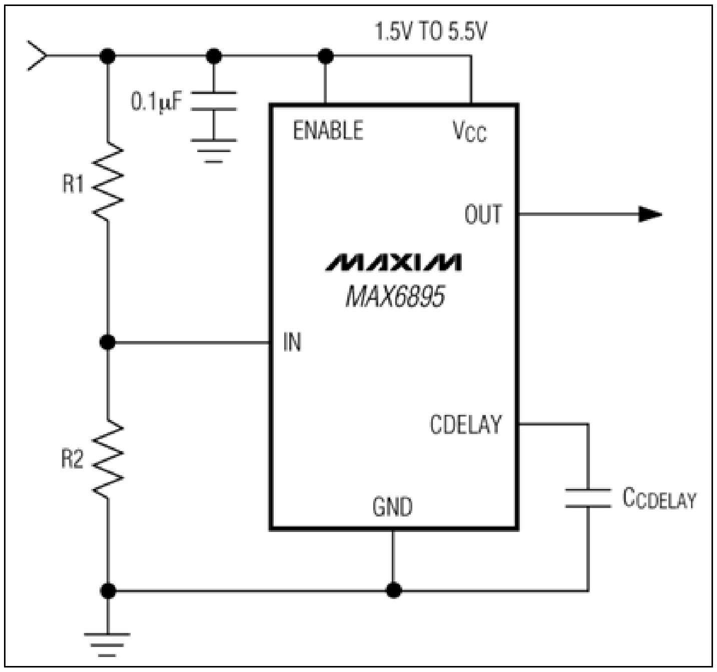


Figure 6. POR with external voltage ref and adjustable reset pulse length

One additional consideration is that a PMP typically wants to wake the core (i.e., restore power) from a user interface (UI) command or button press. This usually comes in the form of a UI interrupt (such as an interrupt from a Wi-Fi module). This UI interrupt would have to be tied into U8 or U10 in Figure 5 if it were to be implemented.

Entering Hibernate Under Software Control

How is hibernate state entered? All PMP systems must have this feature under core control (i.e., software control). Since the regulator is external, a mechanism is needed to disable the core power. In a portable application that uses the hibernate state, the designer will want to keep external components (mSDRAM, flash, etc.) powered while the core is shut off.

The core request that the external buck regulator shut down (`/SHDN`) needs to be able to override any other signal produced by the RTC circuit in Figure 4 or Figure 5.

Depending on how the Blackfin processor boots, the state of the general-purpose flag outputs (GPIOs) may be indeterminate after core power is removed. During the transition of the core power (on and off), the state of these GPIOs must not change. Therefore, in the PMP reference design shown in Figure 7, an externally powered latch (U35 – 74LVC374A) and a D flip-flop (U16 – 74AUC1G74) were added. These two low-cost components are powered by the 1.8-V external voltage rail and remain powered through the entire hibernate process. We must simply write to these latches as part of our configuration/setup stage after a warm or cold boot. Then, to enter hibernate state, we make a 0-to-1 transition on the appropriate latch bit (`SD5`), which turns off the external regulator. This has the added benefit of increasing the number of GPIOs available on the processor. Note that the system power-on switch is connected to the `/SET` pin on the D flip-flop so that the core buck regulator is forced to an "on" state when the PMP is first turned on.

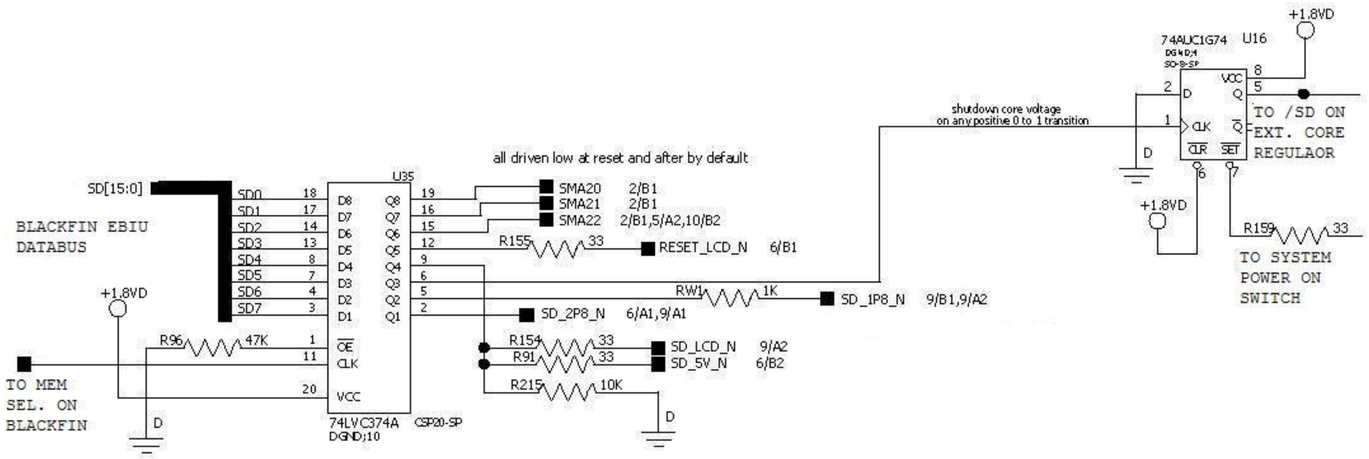


Figure 7. External latch with D flip flop to turn external buck regulator on/off

Three-Stating SDRAM Clock Enable (SDCKE)

When the core goes into the hibernate state, the control signals for the mSDRAM will be driven to their off states when the reset sequence that follows hibernate state is executed. If we look at how SDRAM works, we will find that if the SDRAM clock enable (`SDCKE`) signal is driven high by the host while the mSDRAM is in self-refresh mode, it causes the part to exit self-refresh. Earlier Blackfin processors like the ADSP-BF531/BF532/BF533 did not provide flexibility to maintain the state of the `SCKE` signal during the boot process (which connects to the mSDRAM device's `SDCKE` pin). The unfortunate side-effect of this is the mSDRAM would prematurely exit self-refresh and lose it contents while the Blackfin processor executed its boot process. Therefore, the PMP reference design adds a three-state buffer (U36 - 74LVC1G125 in Figure 8) where the output enable (`/OE`) is tied to the external latch (74LVC374A - U35). Just after we put the mSDRAM in self-refresh prior to the hibernate state, we three-state `SDCKE`. Later, when the core power is restored, `SDCKE` is left undisturbed until after the core does a warm reboot (i.e., the part stays in self-refresh). As a part of the setup and configuration phase, we re-enable `SDCKE`.

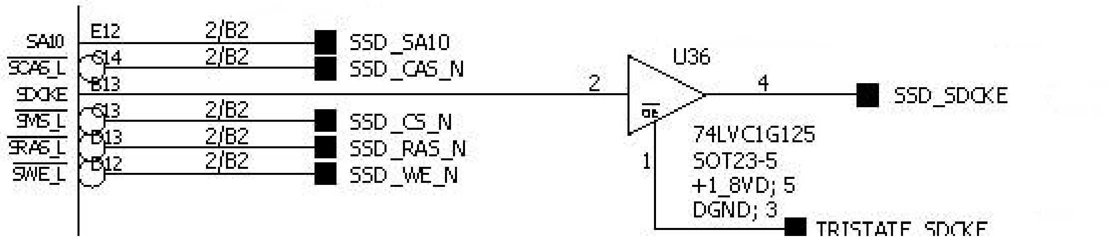


Figure 8. SDCKE three-state buffer

SDRAM Retaining Contents Long after Power Down

With the PMP reference design, it was noted that, depending on the amount of capacitance on the printed circuit board, mSDRAM would retain its contents long after system power was removed (on the order of several minutes). You will recall that we store a key (value = 0x12345678) into the mSDRAM. If this key is present, a warm boot occurs. But in this case, even when full power is removed, the system still attempts a warm reboot (i.e., restore contents from mSDRAM). There is a real risk here for a corrupted boot to occur, depending on the current depletion level of capacitance in the system. Therefore, for a full-context preservation through hibernate state to work properly, the system must realize when a cold boot versus a warm boot has occurred. Possibly the easiest way to do this in hardware is to tie the system reset to the preset pin (`/SET`) on an external D flip flop (74AUC1G74). The Blackfin processor can interrogate this latch at boot time to determine whether the processor is coming up from a full cold boot (flash boot) or a warm boot (hibernate boot). Obviously, a full system reset will not occur for a warm boot, but only for core reset. During the init stage of the boot process, the Blackfin processor can monitor this latch. If it is 1, the processor does a full cold boot from flash. During the context save stage, the Blackfin processor can write out a 0 to this latch. Then, during a warm boot, the processor checks this bit and, if it is a 0, does a hibernate context restore.

The D flip-flop must be powered by the I/O supply (1.8 V) so as to not be impacted by the core voltage reset circuit.

Conclusions

All hibernate operations discussed in this EE-Note are necessary for a system that requires full context save and restore capabilities. For systems that can tolerate the time to perform a full cold reboot, context save and restore operations are not necessary (along with much of the hardware discussed). However, where these features are necessary (such as a PMP that must wake up periodically and quickly), this application is critical. This is not to say that just removing core power is not valuable. It certainly is in some applications. But, the focus herein is to provide a Blackfin solution that provides three major advantages:

- Low active power (~0.16 mW/MIP)
- Low standby power ($\leq$ 300 μW)
- Fast wakeup from a processor standby state (4 msec or less)

Appendix A – Init Block Source Code

```
#include "defbf533.h"
#define OFFSET_( x ) ((x) & 0x0000FFFF)
#define UPPER_( x ) (((x) >> 16) & 0x0000FFFF)
#define LOWER_( x ) ((x) & 0x0000FFFF)
#define SYSMMR_BASE 0xFFC00000
#define COREMMR_BASE 0xFFE00000

.section program;
Start_of_Code:
    [--SP] = ASTAT; [--SP] = RETS; [--SP] = (r7:0); [--SP] = (p5:0);

    P1.H = UPPER_(SYSMMR_BASE);     // P1 Points to top of SYSTEM MMR Space
    P1.L = LOWER_(SYSMMR_BASE);
    R0 = 0x10(z);
    W[P1+OFFSET_(SPI_BAUD)] = R0;   // Speed up SPI rate.

CONFIG_SDRAM:
    R0 = 0x0817(Z);
    W[P1+OFFSET_(EBIU_SDRRC)] = R0;       // SDRAM Refresh Rate Control Register
    R0 = 0x0013(Z);
    W[P1+OFFSET_(EBIU_SDBCTL)] = R0;      // SDRAM Memory Bank Control Register
    R0.H = 0x0091; R0.L = 0x998D;
    [P1+OFFSET_(EBIU_SDGCTL)] = R0;       // SDRAM Memory Global Control Register
    SSYNC;

PLL_change:
    P0.L = lo(SIC_IWR); P0.H = hi(SIC_IWR);
    R1 = [P0];          // Save current SIC_IWR
    R0 = 0x1;
    [P0] = R0;          // Enable only PLL relock to break idle

    P1.H = hi(PLL_CTL); P1.L = lo(PLL_CTL);
    R0.L = 0x2C00;      // CLKIN = 27MHz for the ADSP-BF533 EZ-KIT Lite board
    w[P1] = R0;         // MSEL = 010110 = 22X = 22 X 27 = 594 MHz
    idle;               // Required for PLL relock
    [P0] = R1;          // Restore previous SIC_IWR

    P1.L = lo(PLL_DIV); P1.H = hi(PLL_DIV);
    R0.L = 0x5;   // Default for ADSP-BF533 is 0x0005 -> CCLK/5 = 594/5 = 118.8
    w[P1] = R0;

CHECK_KEY:
    P2.H = 0x0000; P2.L = 0x3FEC;   // Key is located at address 0x3FEC
    R0.H = 0x1234; R0.L = 0x5678;   // Expected key value is 0x12345678
    R1 = [P2++];
    CC = R0 == R1;
    IF CC JUMP RESTORE;       // If key matches, don't do a full boot
FULL_BOOT:
    (p5:0) = [SP++]; (r7:0) = [SP++]; RETS = [SP++]; ASTAT = [SP++];
    RTS;                //If key doesn't match, return back to bootrom for full boot

RESTORE:
    P1.H = UPPER_(SYSMMR_BASE);     // P1 Points to top of SYSTEM MMR Space
    P1.L = LOWER_(SYSMMR_BASE);
    SP = [P2++]; FP = [P2++];       // Restore Stack and Frame Pointers

// Restore L1 Memory via DMA
    R0 = [P2++];        // P2 will point to the Return Address after this update
    [P1+OFFSET_(MDMA_S0_CURR_DESC_PTR)] = R0;
    R0 += 0x20;
    [P1+OFFSET_(MDMA_D0_CURR_DESC_PTR)] = R0;
    R0 = (FLOW-0x3000)|(NDSIZE-0x400)|RESTART|WDSIZE_32|DMAEN(z);
    W[P1+OFFSET_(MDMA_S0_CONFIG)] = R0;
    R0 = (FLOW-0x3000)|(NDSIZE-0x400)|RESTART|WDSIZE_32|WNR|DMAEN(z);
    W[P1+OFFSET_(MDMA_D0_CONFIG)] = R0;

// Restore Scratchpad Memory via Core reads/writes
    P3.H = 0x0001; P3.L = 0xC000;         //P3 = SDRAM Memory
    P4.H = 0xFFB0; P4.L = 0x0000;         //P4 = Scratchpad Memory
    P5.H = 0x0000; P5.L = 0x0400;         //Restore 4096 bytes
    LSETUP(L3_TO_SCRATCH_begin, L3_TO_SCRATCH_end) LC0 = P5;
         L3_TO_SCRATCH_begin:       R0 = [P3++];
         L3_TO_SCRATCH_end:         [P4++] = R0;

// Restore EVT Table via Core reads/writes
    P4.H = 0xFFE0; P4.L = 0x2000;         //P4 = Event Vector Table
    P5.H = 0x0000; P5.L = 0x0010;         //Restore 16 entries
    LSETUP(L3_TO_EVT_begin, L3_TO_EVT_end) LC0 = P5;
         L3_TO_EVT_begin:     R0 = [P3++];
         L3_TO_EVT_end:       [P4++] = R0;

    P4.H = 0xFFE0; P4.L = 0x2104;         //Restore IMASK
    R0 = [P3++];
    [P4] = R0;

TST_RUN_BIT:                              // Wait until L1 Memory DMA is complete
    R0 = W[P1+OFFSET_(MDMA_D0_IRQ_STATUS)](z);
    CC = BITTST(R0,3);
    IF CC JUMP TST_RUN_BIT;               //Wait until DMA is complete

/**** For profiling purposes ****/
    R0 = 0x0080;
    W[P1+OFFSET_(FIO_DIR)] = R0;          //Enable PF7 as output
    W[P1+OFFSET_(FIO_FLAG_S)] = R0;       //Set PF7 high
/********************************/
    P5 = [P2];
JUMP(P5);
```

References

[1] *MT48H4M16LF – 1 Meg x 16 x 4 banks Mobile SDRAM Data Sheet.* Rev C, October 2007. Micron Technology, Inc.

[2] *ADSP-BF533 Blackfin Booting Process (EE- 240).* Rev 3, January 11, 2005. Analog Devices, Inc.

[3] *Using External Switching Regulators to Supply an Adjustable Internal Voltage to Blackfin Processors.* Rev 3, January 30, 2007. Analog Devices, Inc. (included in `.ZIP` file)

[4] *External Circuitry for Hibernate Wakeup Function on Blackfin Processor with a Fixed Voltage External Voltage Regulator*. March 2007, Analog Devices, Inc. (included in `.ZIP` file)

[5] *ADSP-BF531/ADSP-BF532/ADSP-BF533 Blackfin Embedded Processor Data Sheet.* Rev E, July 2007. Analog Devices, Inc..

[6] *ADSP-BF533 Blackfin Processor Hardware Reference*. Rev 3.2, July 2006, Analog Devices, Inc.

Document History

Revision	**Description**
Rev 1 – May 5, 2008 by J. Sondermeyer, Joe Beauchemin, and Hiren Desai	Initial Release