BCA CPE

Power Management

Application Note

Table of Contents

1 Introduction 4
2 Architecture 4
3 Compile Instructions and Options 5
3.1 Configuring Power Management 5
3.2 Configuring Other Blocks 5
3.3 Board Parameters 5
4 Shell Utility 6
4.1 pwr config Command 6
4.1.1 Power Profiles 6
4.2 pwr show Command 7
5 Energy Saving Feature Description 8
5.1 Disk: USB and SATA 8
5.2 Wi-Fi Suspend 8
5.3 PCIe ASPM 9
5.4 UBUS DCM 9
5.5 CPU Off Module 9
5.6 CPU Wait Command 10
5.7 CPU Speed Command 10
5.7.1 CPUFREQ Framework 11
5.7.2 CPUIDLE Commands 11
5.8 XRDP Clock Gating 12
5.9 Net Down 12
5.10 PHY Down 12
5.11 PHY EEE 13
5.12 PHY APD 13
5.13 StarFighter2 Deep Green Mode 14
5.14 DRAM SR 14
5.15 Adaptive Voltage Scaling 14
6 Temperature Throttling and Tracking 15
7 Wake On LAN (WOL) Mode 16
7.1 Enabling Magic Packet Detection (MPD) (Optional) 16
7.2 Entering WOL Mode 17
7.3 Exiting WOL Mode 17
7.4 WOL Example Script 17
Revision History **18**

1 Introduction

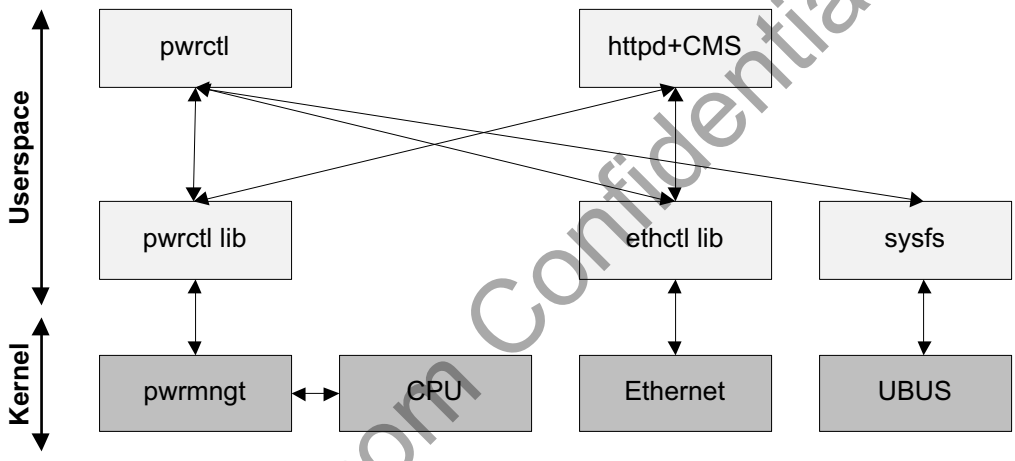
This document describes how to control the various power management features available in Broadcom CPE Gateway software development kit (SDK).

2 Architecture

Power management is implemented in a distributed fashion in the CPE SDK. Each driver is responsible for handling the low-level power management features supported by its hardware. The driver typically provides control through a *sysfs* or an *ioctl* interface. A userspace application called *pwrctl* and associated library are made available to provide a human interface to control the features. Some of the power management features can also be controlled from the Web interface.

A power management driver also exists to interface with other drivers that currently do not support a *sysfs* or *ioctl* interface.

Figure 1: Power Management Architecture



Power management features can be divided in two categories: dynamic features and static features.

Dynamic power management features are mostly implemented in hardware and only configured by software. Once enabled by software, the hardware is responsible for detecting idle periods and for placing some of its components in a sleep or suspended mode, thus saving power. The hardware is also responsible for detecting the end of the idle period and for waking up the necessary components to accomplish some work. When software disables the feature, the hardware can work in full capacity regardless of the idle periods. Dynamic power saving features are designed and tuned to avoid affecting the end-user experience. Some examples of such feature are Energy Efficient Ethernet and Ethernet Auto-Power-Down.

Static power management features do not have hardware to detect idle periods. An external decision is made to place some hardware in a disabled or suspended state and to return to a normal state. Software directly controls the state of the hardware. Due to their nature, static power saving features may have an impact on the end-user experience and may only be useful under certain conditions where the affected components are not required. Some examples of such feature are Wi-Fi suspend and Network power down.

3 Compile Instructions and Options

Most power management features are already enabled in the software releases. They are controlled through *menuconfig*.

3.1 Configuring Power Management

When *make menuconfig* is run, the following settings are found under the menu option “Other features”. The example provided here is for profile 963158GW:

```
<M> PWRMNGT Driver
<dynamic>    PWRCTL
[*] Ethernet Auto Power Down and Sleep
[*] Energy Efficient Ethernet
[*] Ethernet Deep Green Mode
[*] UBUS4 dynamic clock module
[*] DDR Self-Refresh Power Saving
[*] UBUS4 Dynamic Clock Management
[*] XRDP Dynamic Clock Gating
[ ] Automated idle CPU power-down
[ ] PCIe L1 Active State Power Management
```

Note that the last two features listed are not enabled on profile 963158GW. This document will explain why in the following sections.

3.2 Configuring Other Blocks

There are other blocks that can be configured in *menuconfig* that will have an impact on power consumption. For example, if USB is compiled in, the USB module will turn on clocks related to the USB hardware and will increase the chip power consumption. If USB is not used in your product, it is recommended to disable USB in *menuconfig*. Similarly for SATA and Wi-Fi, if any of these are unused in the product, they should be compiled out in order to save some power, although the difference can sometimes be small.

3.3 Board Parameters

In some cases, you may want to use the same image on multiple products and therefore you want to have the USB, SATA, or Wi-Fi module compiled in but still want to take advantage of the power savings available on the boards that don’t offer USB, SATA, or Wi-Fi. In such case, you can leave the module compiled in and instead add one of the following board parameters to the board definition in *boardparms.c*:

```
…
{bp_usUsbDis,                .u.us = 1 },
{bp_usPciDis,                .u.us = 1 },
{bp_usSataDis,               .u.us = 1 },
…
```

Note that using this method will not allow USB or SATA to be powered up at a later stage.

4 Shell Utility

An application called *pwrctl* (short name: pwr) is available to interact with the various power saving features. To see the available commands, type:

```
# pwr
Usage:
       pwr config ...
                 --all        on|off|wol|idle      : All power saving features
                 --disk       on|off               : Disk Suspend: USB, SATA etc.
                 --wifi       on|off               : WIFI suspend
                 --pci        on|off               : PCI ASPM: Active State Power Management
                 --ubus       on|off               : UBUS DCM
                 --cpuoff     on|off               : CPU Off
                 --cpuwait    on|off               : CPU Wait
                 --cpuspeed   on|off [speed]       : CPU Speed
                 --xrdp       on|off               : XRDP Clock Gating
                 --net        on|off [ifname]      : Network device down
                 --phy        on|off [ifname]      : PHY Power Down
                 --eee        on|off [ifname]      : PHY EEE: Energy Efficient Ethernet
                 --apd        on|off [ifname]      : PHY APD: Auto Power Down
                 --dgm        on|off               : SF2 DGM: Deep Green Mode
                 --sr         on|off               : DRAM SR: Self Refresh
                 --avs        on|off               : AVS: Adaptive Voltage Scaling
       pwr show
       pwr help
```

4.1 pwr config Command

The command “pwr config” allows enabling or disabling each of the power saving features individually as documented in the help menu. The first argument to this function indicates which power saving feature you want to enable or disable.

The next argument specifies “on” or “off”. A word of caution is necessary here. “On” means that the power saving feature is enabled and “Off” means that the power saving feature is disabled. This can lead to confusion for certain features, for example in the case of USB and SATA power savings, setting the “--disk” to “On” means that the components are suspended and power saving activated. Re-enabling the USB and SATA components is done by setting the “--disk” power saving feature to “Off”.

Some of the features support additional, optional arguments. For example, a number of features accept an interface name as an argument (these names are shown in “ifconfig”). One command, detailed later in this document, accepts a speed argument which is only used for debugging purposes and should be avoided.

4.1.1 Power Profiles

When used with the “--all” option, the “pwr config” command will attempt to configure all of the options according to a power profile associated with each of the On/Off/Idle settings. At boot time, the default setting is “Idle”. The “On” profile could be considered for example, when a system runs on battery and one wants all possible power savings. The “Off” profile is mostly for testing purposes as it ensures that all power saving features are disabled.

The *pwr config* command is used as follows:

- For normal operation:
  `pwr config –all idle`
- To disable all power savings, generally for testing:
  `pwr config –all off`

- To bring the chip to the lower possible power state, for example when on batteries:
  `pwr config –all on`
- To bring the chip in a power state appropriate for Wake On LAN:
  `pwr config --all wol`

The following table describes the pre-compiled profiles used by this command. The “category” column identifies if the feature is dynamic or static as described in the Architecture section.

Table 1: Predefined Power Profiles

Feature	Description	Category	On Profile	Idle Profile	Off Profile	WOL Profile
disk	Disk Suspend: USB, SATA	Static	Suspended	Resumed	Resumed	Suspended
wifi	Wi-Fi Suspend	Static	Suspended	Resumed	Resumed	Suspended
pci	PCIe ASPM (Note 1)	Dynamic	Enabled	Disabled	Disabled	Enabled
ubus	UBUS DCM	Dynamic	Enabled	Enabled	Disabled	Enabled
cpuoff	CPU Off module	Dynamic	Enabled	Enabled	Disabled	Enabled
cpuwait	CPU Wait	Dynamic	Enabled	Enabled	Disabled	Enabled
cpuspeed	CPU Speed	Dynamic	Enabled	Enabled	Disabled	Enabled
xrdp	Runner XRDP Clock Gating	Dynamic	Enabled	Enabled	Disabled	Enabled
net	Network Device Down	Static	Down	Up	Up	Up
phy	Ethernet PHY Power Down	Static	Down	Up	Up	Up
eee	Energy Efficient Ethernet	Dynamic	Enabled	Enabled	Disabled	Enabled
apd	Ethernet Auto-Power-Down	Dynamic	Enabled	Enabled	Disabled	Enabled
dgm	SF2 Switch Deep-Green-Mode	Dynamic	Enabled	Enabled	Disabled	Enabled
sr	DRAM Self-Refresh	Dynamic	Enabled	Enabled	Disabled	Enabled
avs	Adaptive Voltage Scaling	Dynamic	Enabled	Enabled	Note 2	Enabled

NOTE1: PCIe ASPM is not enabled in idle mode by default. To enable it, the code needs to be recompiled.

NOTE2: The AVS feature can only be disabled in the BCM63268, where AVS is optional. In subsequent chips, AVS is not optional and cannot be disabled at run time by the pwr command. See Adaptive Voltage Scaling for details.

NOTE3: Even though the Wake On LAN (WOL) power profile is available for all platforms, actual system may or may not support WOL mode. Refer to the data sheet for details.

The OFF, ON, IDLE and WOL profiles are pre-defined in the *pwrctl.c* application, under the names `profile_full_power`, `profile_power_save`, `profile_idle`, and `profile_wol` respectively. It is possible to modify these profiles, or add additional ones, by changing the code.

4.2 pwr show Command

The “*pwr show*” command is used to display the current status of the power saving features. If a feature is not compiled in because it is not wanted or because it is not supported by the chip, then the status will show as N/A.

The example below is for the BCM63158.

```
# pwr show
Power Management Configuration
WIFI Off            Disabled
DISK Off            Disabled
PCI ASPM            Disabled
UBUS DCM            Enabled
```

```
CPU Off             N/A
CPU Wait            Enabled
CPU Speed           N/A
XRDP Gate           Enabled
NET Down            Disabled
PHY Down            Disabled
PHY EEE             Enabled
PHY APD             Enabled
SF2 DGM             Enabled
DRAM SR             Enabled
AVS                 Enabled
```

In the example shown, two features show N/A; the CPU Off module is compiled out and CPU Speed control is not necessary on BCM63158. More on this in the next section.

5 Energy Saving Feature Description

This section describes the various power saving features available.

5.1 Disk: USB and SATA

By invoking the `pwr config --disk on/off` command, you can suspend and resume the USB and SATA interfaces. This command actually powers down the externally attached device and also shuts down the internal USB and SATA components in the chip, saving as much power as possible. This feature supports the following commands:

- To suspend USB and SATA:
  `pwr config --disk on`
  The above command invokes the following command:
  `/etc/init.d/disk.sh suspend`
- To resume USB and SATA:
  `pwr config --disk off`
  The above command invokes the following command:
  `/etc/init.d/disk.sh resume`

5.2 Wi-Fi Suspend

By invoking the `pwr config --wifi on/off` command, you can suspend and resume all the Wi-Fi interfaces. This command uninstalls the wireless modules and shuts down the PCIe interfaces. This actually stops the Wi-Fi chips to save as much power as possible.

This feature supports the following commands:

- To suspend Wi-Fi:
  `pwr config --wifi on`
  The above command invokes the following command:
  `/etc/init.d/wifi.sh suspend`
- To resume Wi-Fi:
  `pwr config --wifi off`
  The above command invokes the following command:
  `/etc/init.d/wifi.sh resume`

If you see this error message when invoking the script:

```
rmmod: can't unload 'bcm63xx_pcie': unknown symbol in module, or unknown parameter
```

It is because the module that handles the PCIe power control was not compiled in. The module is included in the build by using “make menuconfig”, under “Kernel Configuration Selection”, under “Misc Drivers”, select "PCI Express repower module".

5.3 PCIe ASPM

PCIe ASPM is a standard power saving technique available on most PCIe devices and hosts. Find more about this technology on the Web. While Broadcom chips also support PCIe ASPM, it is not typically enabled by default in our released builds. This feature requires proper board layout and significant testing and support when enabled. It should only be considered when other power savings features are not sufficient to reach your goals. This feature supports the following commands:

- To enable ASPM:
  ```
  pwr config --pci on
  ```
  The above command invokes the following command:
  ```
  echo l1_powersave > /sys/module/pcie_aspm/parameters/policy
  ```
- To disable ASPM:
  ```
  pwr config --pci off
  ```
  The above command invokes the following command:
  ```
  echo default > /sys/module/pcie_aspm/parameters/policy
  ```

5.4 UBUS DCM

UBUS is the main internal bus in the Broadcom PON/DSL chips. This bus has the capacity to support a very large throughput of data, but when the system is idle, this capacity is not necessary. DCM is a dynamic clock management feature that slows down the bus clock when no transaction is pending and immediately speeds up the bus clock on the first incoming transaction. This feature supports the following commands:

- To enable UBUS DCM:
  ```
  pwr config --ubus on
  ```
  The above command invokes the following command:
  ```
  echo 1 > /sys/module/ubus4_dcm/parameters/enable
  ```
- To disable UBUS DCM:
  ```
  pwr config --ubus off
  ```
  The above command invokes the following command:
  ```
  echo 0 > /sys/module/ubus4_dcm/parameters/enable
  ```

5.5 CPU Off Module

PON/DSL chips that include a Brahma53 ARM-8 compatible CPU, generally support the ability to completely power down three of the four cores. Powering down three cores on such a chip provides additional power savings beyond the standard clock control features such as CPU Wait (WFI). A module called *cpu_off* was implemented by Broadcom and is available through *menuconfig* under **Automated Idle CPU Power-Down** to monitor the usage of the cores and power them down when unused. This module is not compiled in by default as it could have impact on system performance. It is available for customers to enable it and experiment with its power savings. Core 0 is never powered down, but the additional cores are controlled by this module. Thresholds are made available to control how sensitive the system must be when usage drops or resumes. More documentation on this feature can be found in a readme file called *bcm_cpuoff.txt* saved with the module C code.

When the CPU OFF module is compiled in, this feature supports the following commands:

- To enable CPU OFF:
  `pwr config --cpuoff on`
  The above command invokes the following command:
  `echo 1 > /sys/module/bcm_cpuoff/parameters/enable`
- To disable CPU OFF:
  `pwr config --cpuoff off`
  The above command invokes the following command:
  `echo 0 > /sys/module/bcm_cpuoff/parameters/enable`

5.6 CPU Wait Command

Both MIPS and ARM CPUs support the ability to gate internal clocks when no process is running. An operating system knows when there is no process that needs to run when the code enters the OS idle loop. If the CPU does not implement power savings in the idle loop, the OS looks through its structures constantly until a process has work to perform. Ultimately, the trigger that activates processes is always an interrupt. While waiting for an interrupt to occur, the CPU can save power by stopping most of its internal clocks and can stop looking for work. The OS informs the CPU that it has no work to do by invoking the WFI (Wait For Interrupt) instruction in ARM, or the WAIT instruction in MIPS.

The CPU Wait power saving feature is a thoroughly tested feature that can always remain enabled. However, for testing purposes, the `pwr config --cpuwait` command is made available to compare power consumption with and without this feature. The command invokes a library API to enable or disable this feature. There is no equivalent setting anywhere else in the user interface.

- To enable CPU Wait:
  `pwr config --cpuwait on`
- To disable CPU Wait:
  `pwr config --cpuwait off`

5.7 CPU Speed Command

To save additional power when the system is idle, it is possible to automatically adjust the CPU frequency based on the amount of work it has. The power that is saved when scaling the frequency of a CPU overlaps with the power that is saved through the CPU WAIT feature. Thus, enabling both features simultaneously only provides a small additional power saving benefit over enabling only one of the two features. In some cases, as in the BCM63158, the CPU WAIT feature is improved to the point where there is no additional benefit in slowing down the CPU. For this reason, frequency scaling is only made available in chips where power savings are seen beyond what the CPU WAIT feature provides.

Linux supports two frameworks that can control CPU frequency. The first framework is called CPUFREQ and uses a governor to monitor the CPU demand and adjust the frequency accordingly. In some cases the on-demand governor does not react very quickly to demand. A second option is the interactive governor that can be ported from Android. The BCM6858XX, the BCM63138, and the BCM63148 are using this framework.

Some Broadcom chips, such as the BCM4908, BCM6836X, and BCM6856X, have the ability to very quickly adjust CPU frequency on the fly, without having to coordinate between all cores. In this case, it is possible to use the CPUIDLE framework in Linux, which invokes the Broadcom CPU frequency handler each time the system enters an idle period, during which the frequency can be lowered. The frequency is then readjusted to its maximum speed as soon as an interrupt occurs. The BCM63268 also uses a similar design as the BCM4908, where frequency can be adjusted on the fly, but the implementation does not use the Linux CPUIDLE framework.

In all cases, the following commands are used to turn on or off the feature:

- To enable Automated CPU Frequency Scaling:
  `pwr config --cpuspeed on`
- To disable Automated CPU Frequency Scaling:
  `pwr config --cpuspeed off`

What the command does depends on which of the frameworks is enabled in the device.

5.7.1 CPUFREQ Framework

The CPUFREQ framework is a standard Kernel feature that is well documented in Linux. Below is a summary of the available settings in this CPUFEQ framework. When using the `pwr config --cpuspeed` command, the CPUFREQ system switches between “interactive” and “userspace” governors as described below.

- Display the current governor used by the system:
  `cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor`
- List available governors in the system:
  `cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors`
- Select the userspace governor (invoked when doing pwr config --cpuspeed off):
  `echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor`
- List available CPU frequencies (in kHz):
  `cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies`
- Show current frequency (in kHz):
  `cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_cur_freq`
- Set CPU frequency to 750000 kHz:
  `echo 750000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed`
- Select the interactive governor (invoked when doing pwr config --cpuspeed on):
  `echo interactive > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor`
- Display time spent in each of the available frequencies:
  `cat /sys/devices/system/cpu/cpu0/cpufreq/stats/time_in_state`

To use the interactive governor, it is necessary to disable the Kernel Linux feature called Scheduler Runtime Sharing (BRCM_SCHED_RT_SHARE). Control to this setting is provided through *menuconfig*:

make menuconfig > Kernel Configuration Selection > Scheduler RT RQ sharing

By disabling this setting, realtime processes that are tied to a CPU will not be able to block non-realtime processes. For more information about `cpufreq` and governors, consult the Linux documentation found in the folder *kernel/linux-3.4rt/Documentation/cpu-freq* in the CPE SDK software release.

5.7.2 CPUIDLE Commands

The CPUIDLE framework, as implemented in the SDK, does not need any configuration commands. By design, it does not impact performance and therefore does not need any threshold. It can only be enabled or disabled with the `pwr config --cpuspeed` command previously discussed. This command invokes an *ioctl* API to control the CPU speed.

5.8 XRDP Clock Gating

The Runner Network Processor found in the latest chips such as BCM63158 and BCM6858XX has the ability to gate the clocks inside its cores when they are unused. When there is little or no traffic to process, Runner cores automatically detect inactivity and gate some of the clocks to save power. This feature supports the following commands:

- To enable XRDP clock gating:
  `pwr config --xrdp on`
  The above command invokes the following command:
  `/bin/bs /b/c system clock_gate=yes`
- To disable XRDP clock gating:
  `pwr config --xrdp off`
  The above command invokes the following command:
  `/bin/bs /b/c system clock_gate=no`

5.9 Net Down

To stop traffic from being processed in Linux, it is possible to bring the interfaces down. This can be useful for example when running on battery. This feature supports the following commands:

- To bring all network interfaces down:
  `pwr config --net on`
  Or, for a specific interface:
  `pwr config --net on <ifname>`
  The above command invokes the following command:
  `ifconfig <ifname> down`
- To bring back all network interfaces up:
  `pwr config --net off`
  Or, for a specific interface:
  `pwr config --net off <ifname>`
  The above command invokes the following command:
  `ifconfig <ifname> up`

5.10 PHY Down

When running on battery, it may be desired to force an Ethernet PHY down even if the link is currently up. This feature supports the following commands:

- To enable full power savings for all Ethernet PHYs (to power down the PHYs)
  `pwr config --phy on`
- To disable full power savings for all Ethernet PHYs (to power up the PHYs)
  `pwr config --phy off`
- To enable full power savings for a single Ethernet PHY (to power down the PHY)
  `pwr config --phy on [ifname]`
  The above command is equivalent to executing:
  `ethctl <ifname> phy-power down`
- To disable full power savings for a single Ethernet PHY (to power up the PHY)
  `pwr config --phy off [ifname]`
  The above command is equivalent to executing:
  `ethctl <ifname> phy-power up`

5.11 PHY EEE

Energy Efficient Ethernet is a well-documented standard allowing power saving when no packets are being transmitted on an Ethernet interface. This feature supports the following commands:

- To enable EEE for all interfaces
  `pwr config --eee on`
- To disable EEE for all interfaces
  `pwr config --eee off`
- To enable EEE for a single interface
  `pwr config --eee on [ifname]`
  The above command is equivalent to using
  `ethctl <ifname> eee on`
- To disable EEE for a single interface
  `pwr config --eee off [ifname]`
  The above command is equivalent to using
  `ethctl <ifname> eee off`

After connecting an Ethernet cable to a partner, it is possible to determine if the link has successfully negotiated with EEE. Generally, this happens when both ends have EEE enabled. The following command allows the user to determine if both ends have agreed on enabling EEE:

- To read EEE resolution
  `ethctl <ifname> eee-resolution`

5.12 PHY APD

Broadcom Ethernet PHY hardware have the ability to automatically power down if there is no cable connected, if there is no link partner. Approximately every three seconds, the PHY awakes for a very short time to determine if a link partner has attempted to connect, and to signify to its presence.

This feature supports the following commands:

- To enable APD for all interfaces
  `pwr config --apd on`
- To disable APD for all interfaces
  `pwr config --apd off`
- To enable APD for a single interface
  `pwr config --apd on [ifname]`
  The above command is equivalent to using
  `ethctl <ifname> apd on`
- To disable APD for a single interface
  `pwr config --apd off [ifname]`
  The above command is equivalent to using
  `ethctl <ifname> apd off`

5.13 StarFighter2 Deep Green Mode

The StarFighter2 Ethernet switch uses internal clocks that can be slowed down when all the LAN ports are down. This feature is implemented in software in the Ethernet driver under the name Deep-Green-Mode (DGM). If DGM power saving is enabled, when all of the LAN ports of the switch are down (generally by disconnecting the Ethernet cables) the software detects the transition and activates DGM. When a first LAN port is brought up by connecting an Ethernet cable, the software also detects the transition and deactivates DGM.

- To enable DGM:
  `pwr config --dgm on`
  The command directly invokes an *ioctl* call to the Ethernet driver.
- To disable DGM:
  `pwr config --dgm off`

5.14 DRAM SR

DDR memories require constant refreshing in order to retain data. Normally, the DDR performs refresh when the memory controller periodically sends the Auto-Refresh command. This mode of operation causes pending read or write accesses to queue while the refresh takes place. However, when there are no pending read or write requests, the memory controller can chose to send a different command called Self-Refresh. This command tells the DDR to handle the periodic refresh cycles autonomously. The DDR Self-Refresh mode allows both the DDR and the Memory controller to save power by disabling or slowing down internal clocks and to stop driving the bus lines.

In more recent chips such as the BCM63158, the memory controller chooses which mode it wants to operate in (Auto-Refresh or Self-Refresh) by monitoring read and write requests and switching between the two modes based on activity. In previous chips, software was responsible for identifying idle periods and telling the memory controller to enter Self-Refresh. In all DSL and PON chips, the memory controller is capable of exiting the Self-Refresh mode autonomously on the first read or write request with little latency.

The DDR Self-Refresh feature is only enabled on a few chips. Only consider using this feature if it is enabled in the most recent software release for the chip you are using.

- To enable DDR Self-Refresh:
  `pwr config --sr on`
  The command directly invokes an *ioctl* call to the power management driver.
- To disable DDR Self-Refresh:
  `pwr config --sr off`

5.15 Adaptive Voltage Scaling

Adaptive Voltage Scaling (AVS) is a power saving technique that reduces the input voltage to the digital logic of the chip without affecting its performance. This document is not meant to describe AVS and the following paragraphs only provide a summary that can be used to understand the software commands associated with AVS. For more details about AVS, consult the chip's data sheet.

During development, chips are designed to meet setup and hold timing requirements of the digital logic. During manufacturing, variations in the process produces chips that very easily meet these timing requirements (fast parts) and chips that simply meet the requirements (slow parts). Since all parts for a given chip are clocked at the same set of frequencies, the fast parts have a lot of timing tolerance while slow parts have less timing tolerance.

AVS uses internal circuitry to measure the amount of tolerance of each part at run-time. If the part is found to have a lot of tolerance, AVS will set the digital logic to run at a lower voltage, thus reducing the tolerance closer to specification. If the part is found to have less tolerance, the voltage is reduced less.

The AVS feature is not something that can be turned on or off in a product. This is because the maximum heat dissipation of the parts is calculated assuming that AVS is used. Disabling AVS would increase the maximum heat dissipation of the part.

The command “`pwr show`” reports if AVS is enabled and will normally report “On”. It is not possible to disable AVS using the “pwr” commands, except for the BCM63168/63268 where AVS was optional.

6 Temperature Throttling and Tracking

The temperature throttling feature keeps the chip temperature within operating range by adjusting the chip performance at high temperature. The temperature tracking feature forces the chip to reset if it exceeds certain thresholds. Temperature tracking is supported through the PMC firmware and the PMC Linux driver while temperature throttling is implemented using the bcm_thermal Linux driver and the WLAN (Wireless LAN) driver.

It should be noted that the temperature readings of the chip must not be used to estimate the overall system temperature. There are many components on the PCB that contribute to the system temperature. To properly track system temperature, using a calibrated external temperature sensor is required.

On BCM63158, the PMC firmware monitors the chip temperature; if the temperature goes beyond 125°C, the PMC firmware sends an interrupt to the PMC Linux driver, that tries to orderly reboot the board. If the reboot does not occur within three seconds, the PMC firmware will issue a chip soft reset. On BCM63178 and BCM47622, the PMC firmware directly enables the hardware over-temperature warning and reset using 130°C and 133°C as the respective thresholds; if the chip temperature goes beyond 130°C, the hardware generates a warning interrupt to the PMC Linux driver that tries to orderly reboot the board. If the reboot does not occur and the chip temperature goes beyond 133°C, the hardware will do a chip soft reset. For test and other experimental purposes, CFE provides the command "pmc tracktemp [status|on|off]" to show or set the status of this feature in the PMC firmware.

The bcm_thermal Linux driver utilizes the Linux thermal framework to track both the chip temperature and the ARM CPU temperature; if the maximum of these two temperatures goes beyond 115°C, the bcm_thermal Linux driver will take the one CPU offline (CPU3 for BCM4908, BCM63158, BCM47622, and BCM6755, CPU2 for CM63178 and BCM6750). If the temperature goes beyond 125°C, the bcm_thermal Linux driver will take all the possible CPUs except CPU0 offline. When the temperature returns below 110°C (accounting 5°C for hysteresis), the bcm_thermal Linux driver will return all the offline CPUs back online. For test and other experimental purposes, the Linux shell command "rmmod -w bcm_thermal" can be used to disable this feature by unloading the bcm_thermal Linux driver module from the running kernel.

On BCM63178, BCM6750, BCM47622, and BCM6755, the WLAN driver switches the Wi-Fi radio off (reducing 50% of chains) when the Wi-Fi core temperature goes beyond 110°C.

Table 2 summarizes the temperature throttling thresholds, actions and minimum SDK versions for BCM4908, BCM63158, BCM63178, BCM6750, BCM47622, and BCM6755.

N: CPU Core ID starting from 0

N=2 for BCM63178 and BCM6750

N=3 for BCM4908, BCM63158, and BCM47622

Table 2: Broadcom xDSL Thermal Throttling - Degraded Performance and Hardware Protection

	Threshold 1 Tj > 110°C	Threshold 2 Tj > 115°C	Threshold 3 Tj > 125°C	Threshold 4 Tj > 130°C	Threshold 5 Tj > 133°C
WLAN rate limiting, 50% chain redux	5.02L.07 for 63178/47622/6750/6755	–	–	–	–
ARM CPU Core N shutdown	–	5.02L.05 for 4908 5.02L.07 for 63158/63178/47622/6750/6755	–	–	–
ARM CPU Cores 1..N-1 shutdown	–	–	5.02L.05 for 4908 5.02L.07 for 63158/63178/47622/6750/6755	–	–
PMC orderly reset	–	–	5.02L.06 for 63158	5.02L.07 for 63178/47622/6750/6755	–
PMC forced reset	–	–	5.02L.06 for 63158 (if orderly reset fails after three seconds)	–	5.02L.07 for 63178/47622/6750/6755 (fully HW-based)

7 Wake On LAN (WOL) Mode

Wake On LAN (WOL) mode is supported on certain platforms. Certain modules can operate in low power, or suspended state while operating in WOL mode. Refer to the WOL power profile outlined in Table 1, where Disk (USB and SATA) and Wi-Fi are placed into power saving mode as an example.

There are two sources of interrupt that can take the system out of WOL mode:

- Receiving an ARP packet. (Enabled by default)
- Receiving a Magic Packet.

Currently WOL mode is only supported on the BCM47622 and BCM6755 platforms.

7.1 Enabling Magic Packet Detection (MPD) (Optional)

To enable Magic Packet as one of the wake sources in WOL mode, the user must configure the target MAC address in the magic packet. One of the following two methods can be used:

- Specify a target MAC address
  `archerctl mpd_cfg -int <interface> xx:xx:xx:xx:xx:xx`
  Where <interface> is the network interface where MPD should be enabled, for example, eth0.1.
- Retrieve the MAC address of a specific interface for MPD
  `archerctl mpd_cfg -int <interface>`

7.2 Entering WOL Mode

Before entering WOL mode, the user must put the system in an appropriate power configuration to save as much power as possible:

```
pwr config --all wol
```

The user can use the following command to put a network interface in WOL mode:

```
archerctl wol_enter -int <interface>
```

While one network interface enters WOL mode, the other interface can be placed in power down mode for low power operation mode.

7.3 Exiting WOL Mode

While in WOL mode, a userspace application can monitor the WOL status by retrieving it from `/proc/driver/archer/wol_status`.

When the status is no longer WOL, the system has exited WOL. At this point, the system can be returned to its normal "idle" power state:

```
pwr config --all idle
```

7.4 WOL Example Script

A script in `userspace/private/apps/scripts/pwr/wol.sh` implements all the steps to enter and exit WOL properly as described in the previous subsections.

Revision History

CPE-AN3601, May 28, 2019

- Updated Table 1, Predefined Power Profiles
- Added Temperature Throttling and Tracking
- Added Wake On LAN (WOL) Mode

CPE-AN3600, October 28, 2018

- Initial release