系统消费电力简介

2014/6/23
renyl

1 介绍

整个系统的电力消耗主要分为以下几类：
1） CPU
2） GPU
3） 内存
4） 磁盘
5） 网卡
6） 显示器
7） 其它设备

接下来，对这些设备如何消耗电力以及如何节电进行分别介绍。

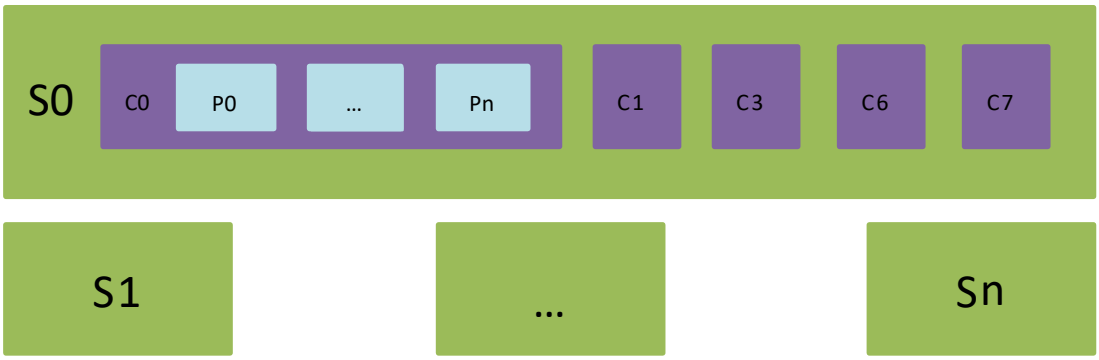
2 电力消耗

2.1 CPU

CPU 的电力消耗占系统整体电力消耗中的大部分，因此对 CPU 的有效节电能够明显降低系统整体电力消耗。

CPU 有多种状态，常见的有如下几种：
1) S-States（Sleeping states）：系统睡眠状态，当手动点击“睡眠”，或者达到一定的待机时间便进入睡眠状态。
2) C-States（CPU Power states）：CPU 电源状态，CPU 进入睡眠的深度，睡眠越深，电力消费越少，越节电。
3) P-States（CPU Performance states）：CPU 性能状态，表示 CPU 的核心频率，频率越低，电力消费越少，越节电。
4) T-States（thermal states）：CPU 热状态，表示 CPU 的温度，温度越低，电力消耗越少，越节电。

其中，S-States、C-States 和 P-States 之间的关系如下图所示：



说明：
S-States 中的 S0 指非睡眠状态，包含了系统正常运作状态以及待机状态。这意味着只有在 S0 状态下，C-States 才会存在。同样，C0 代表正常工作状态，而 P-States 正是处理器正常运行时的状态，所以 P-States 只存在 C0 状态下。

关于 CPU 的节电目前主要其中在 C-States 和 P-States。

C-States 有多种状态，根据 CPU 架构的不同会略有差异，但在 X86 架构中以下这几种状态基本都存在且表示的含义基本一致。
1） C0：CPU 处于正常的运行状态下。

2） C1\C1E：CPU 处于一个不执行任何指令的状态，但是能够随时执行工作且几乎没有延时。这种状态并不是一个低电力消耗的状态。

3） C2：CPU 的内部时钟会停止，但是仍然保持其 register 和 Cache 的完整状态。当内部时钟再次启动时，CPU 能够立即进入工作状态。该状态处于一个相对低电力消耗的状态。

4） C3：CPU 会进入 Sleep 状态并且不会对 Cache 的内容进行更新。该状态处于一个比 C2 状态更加省电的状态。

当然，不同的 CPU 还有更多的 C-States 状态，如 C6，C7，…Cn。C-States 进入的睡眠状态越深，关闭的功能也就越多，其电力消耗的也就越少。但是 CPU 若睡眠越深，唤醒其进入工作状态的时间也就越长，即延时也就越大。

可以使用命令“cpupower idle-info”来查看当前系统 CPU 所支持的 C-states 以及每种状态唤醒时的延时，如下所示：

```
[root@localhost renyl]# cpupower idle-info
CPUidle driver: intel_idle
CPUidle governor: menu

Analyzing CPU 0:
```

```
Number of idle states: 6
Available idle states: POLL C1-SNB C1E-SNB C3-SNB C6-SNB C7-SNB
POLL:
Flags/Description: CPUIDLE CORE POLL IDLE
Latency: 0
Usage: 1
Duration: 8
C1-SNB:
Flags/Description: MWAIT 0x00
Latency: 2
Usage: 6107
Duration: 785034
C1E-SNB:
Flags/Description: MWAIT 0x01
Latency: 10
Usage: 8626
Duration: 2827982
C3-SNB:
Flags/Description: MWAIT 0x10
Latency: 80
Usage: 1219
Duration: 760964
C6-SNB:
Flags/Description: MWAIT 0x20
Latency: 104
Usage: 32
Duration: 64528
C7-SNB:
Flags/Description: MWAIT 0x30
Latency: 109
Usage: 8888
Duration: 338383141
```

由于 CPU 在性能和电力消耗上处于一个相互矛盾的位置。因此，需要在性能和电力消耗之间的选择上做一个平衡。

P-States 支持多种的 CPUfreq Governor，根据 CPU 架构的不同，支持的模式也不同，主要有如下几种：

1) performance：这种模式下的 CPU 时钟频率将一直维持在一个尽可能的最高值，系统性能最好，但不利于节电。

2) powersave:在这模式下的 CPU 时钟频率将一直维持在一个尽可能的最低值,利于节电，但损失了系统的性能。

需要注意的是，如果在 powersave 模式下 CPU 处于满负载运行，但在 performance 模式下，CPU 并没有处于满负载运行，在这种情况下，powersave 模式不仅比 performance 模式的性能差，电力消费可能也更多。

因此，powersave 模式更适合于 CPU 处于 low activity 的时候。

3) ondemand：这模式下的 CPU 时钟频率将会根据 CPU 的负载情况来自动地动态调整，当 CPU 负载高时，CPU 时钟频率将会调整到最高值，当 CPU 负载低时，CPU 时钟频率将会调整到最低值。

需要注意的是，由于 CPU 频率的切换会产生延时，因此如果 CPU 的时钟频率切换太过频繁的话，过多的延时就会抵消掉 ondemand 模式所带来的高性能和低电力消耗优势。所以，ondemand 模式适合于 CPU 要么长时间处于 high workload，要么长时间处于 idle 状态。对 CPU 一会处于 high workload，一会处于 idle 这样的情况，ondemand 模式并不适合。

4) userspace：这种模式可根据用户的配置来设置 CPU 的频率，它能提供最好的平衡在性能和电力消耗之间。

5) conservative：这种模式与 ondemand 类似，也会根据 CPU 的负载情况来动态调整 CPU 的频率。与 ondemand 模式不同的是，ondemand 调整的 CPU 频率要么在最大值，要么在最小值，而 conservative 模式会根据 CPU 的负载情况来逐步调整 CPU 的频率。

主要注意的是，conservative 模式比 ondemand 模式更加节电，但是由于频繁的切换时钟频率，会导致更大的延时产生。

可以通过文件“/sys/devices/system/cpu/cpu1/cpufreq/scaling_available_governors”或者命令“cpupower frequency-info --governors”来查看当前系统的 CPU 所支持的 CPUfreq Governor 模式，如下所示：

```
[root@localhost/]#cat /sys/devices/system/cpu/cpu1/cpufreq/scaling_available_governors
performance powersave
[root@localhost /]# cpupower  frequency-info  --governors
analyzing CPU 0:
performance powersave
[root@localhost /]#
```

通过命令“echo[governor]>/sys/devices/system/cpu/cpuN/cpufreq/scaling_governor”或者 “cpupower frequency-set －governor [governor]”来设置 CPU 的 CPUfreq Governor 模式，如下所示：

```
[root@localhost /]# cpupower  -c 1 frequency-set  --governor performance
Setting cpu: 1
[root@localhost/]#echo performance >/sys/devices/system/cpu/cpu1/cpufreq/scaling_governor
[root@localhost renyl]#
```

2.2 GPU

GPU 内部的多个时钟管理着其内部各种各样的组件，减少 GPU 内部时钟的 cycles 数能够降低 GPU 的电力消耗。

目前 Linux 系统没有给用户提供接口来控制 GPU 内部时钟的频率。

但是 RHEL7 的内核在 GPU 处于 idle 时，会自动减小其内部时钟的 cycles 数，达到节电目的。当 GPU 需要工作时，内核会自动提高其内部时钟的 cycles 数。

另外，RHEL7 会自动检测当前系统是否有 monitor 连接到机器上，如果没有的话，将会自动关闭 GPU 的所有功能，达到节电目的。

2.3 内存

操作系统的内存是由 DRAM（Dynamic Random Access Memory）组成的，DRAM 为了保存数据使用电容进行存储，然而 DRAM 只能将数据保存很短的时间，所以必须每隔一段时间刷新一次，如果存储单元没有被刷新，存储的信息就会丢失。

DRAM 的刷新频率是由内存控制器来控制的，且 DRAM 每隔一段时间就要刷新一次。因此，DRAM 和内存控制器是需要消耗电力的。

目前 Linux 系统没有给用户提供接口来控制 DRAM 和内存控制器的电力消耗。

然而，RHEL7 系统对 Intel Grphics Adapters 提供了优化机制，当内存控制器是 idle 时，系统会触发 DRAM 内部的定时器产生一个 self-refresh cycles，从而达到节电目的。

2.4 磁盘

磁盘控制器控制着磁盘的读写操作，磁盘控制器和磁盘的操作都需要消耗电力。

RHEL7 针对磁盘提供了 ALPM（Aggressive Link Power Managenment）技术来进行节电。当磁盘处于 idle 状态（没有 I/O 操作）时，系统会自动设置一个 SATA link 使磁盘进入 low-power 状态，从而进行节电。当磁盘需要进行 I/O 操作时，ALPM 会自动设置磁盘到正常活动状态。

ALPM 技术能够对磁盘进行节电，但是会让对磁盘的操作产生延时。

RHEL7 系下，ALPM 技术默认是开启的，ALPM 有三种模式，如下所述：

1) min_power：这种模式当磁盘没有 I/O 操作时，会设置磁盘进入最省电力状态。因此，这种模式适合磁盘长时间处于 idle 状态。

2) medium_power：这种模式当磁盘没有 I/O 操作时，会设置磁盘进入较省电力状态。对磁盘的性能影响较小。

3) max_performance：这种模式下相当于禁用 ALPM 技术，即磁盘在没有 I/O 操作时，也不会使磁盘进入任何节电状态。这种模式对磁盘的性能最好，同时也是最消耗电力的。

需要注意的是，ALPM 技术目前只适合于 SATA 控制器，可以通过检查文件/sys/class/scsi_host/host*/link_power_management_policy 是否存在来判断当前系统上磁盘是否支持 ALPM 技术。

为了改变 ALPM 模式，只需向/sys/class/scsi_host/host*/link_power_management_policy 文件写入 ALPM 模式值即可。

2.5 网卡

网络控制器控制着网卡的数据发送与接收，网络控制器与网卡的操作都需要消耗电力。

目前，RHEL7 并没有针对网卡的节电提出新技术。在网卡接口速度没有称为性能瓶颈的前提下，RHEL7 仅提供了可以通过限制网卡的接口速度来进行节电。

可以使用工具 ethtool 来调整网卡的接口速度。

2.6 显示器

显示器上的画面是由系统传送 pixel 信息到 screen 所产生的。图形控制器控制着 pixel 信息的发送频率。

RHEL7 系统提供了一种新技术 LVDS(Low-voltage differential signaling)，当 screen 没有变化时，会自动降低图形控制器发送 pixel 信息的频率，从而达到节电目的。

2.7 其它设备

2.7.1 PCIe

RHEL7 系针对 PCIe（Peripheral Component Interconnect Express）设备提供了 ASPM（Active-State Power Management）技术来进行节电。ASPM 技术能够进行节电，但同时也会使设备产生延时。

ASPM 提供了 3 种节电模式，如下所述：
1) default：这是 ASPM 的默认模式，由系统的 firmware 进行设定（如 BIOS 里设定）。

2) powersave：这种模式不管性能的损失，尽可能的进行节电。

3) performance:这种模式相对于禁用 ASPM 技术,尽可能的发挥最大性能,不用进行节电。

主要注意的是，不是所有的 PCIe 设备都支持 ASPM 技术，通过命令“dmesg |grep aspm”可以查看当前系统的 PCIe 设备是否支持 ASPM 技术。

ASPM 的节电模式通过修改文件“/sys/module/pcie_aspm/parameters/policy”来操作。

2.7.2 无线设备

许多操作系统会提供了一些具有无线传输功能的设备，如 Wi-Fi、Bluetooth、3G、4G 等设备。这些设备都需要消耗电力。

RHEL7 系通过 RFKill 来提供一个接口操作无限传输功能，让用户查询、激活和关闭某项具有无限功能的设备。通过这些接口来关闭一些不需要的设备从而达到节电目的。

使用工具 rfkill 可以完成具有无限功能设备的查询、开启与关闭

3 电力调优

REHL7 系提供的 tuned 服务默认自带多个 profile，每个 profile 调优的侧重点不同，从而对系统进行不同的调优,。各 profile 的简要描述如下所示：

1) powersave：最佳节电模式。
2) throughput-performance：最佳性能模式。
3) network-throughput：基于 throughput-performance 模式，针对网络的吞吐量专门优化的模式。
4) latency-performance：最小延时模式。
5) network-latency：基于 latency-performance 模式，针对网络延时专门优化的模式。

通过使用不同的 profile，对 TPC-E（cache 模式，agent=8）进行测定，相对于默认模式的测定结果如下所示：

番号	profile	Throughput	电力消耗
1	throughput-performance(默认模式)	-	-
2	powersave	×	↓9%
3	network-throughput	×	×
4	latency-performance	↑6%	×
5	network-latency	×	↑8%

注：

×：表示基本没有变化

↓：表示下降。

↑：表示上升。

通过测定结果可以发现，使用 powersave 模式相比默认模式可以在性能基本不变的情况下，使得系统电力消耗下降约 9%。

为了实现更好的节电效果，在 tuned 服务之外，尝试手工对一些参数进行调优，从而达到系统节电目的。根据上述对系统整个电力消费的各个组成部分的研究，结合本次的调查背景，发现可以用来进行节电优化的部分只有 CPU、磁盘和网卡。

在 tuned 服务关闭的情况下，针对 CPU、磁盘和网卡分别做了如下节电优化尝试，TCP-E 的测定结果如下表所示：

类型	方法	说明	Throughput	电力消费
CPU	x86_energy_perf_policy powersave	设置 CPU 进入最佳节能模式	×	×
	nohz_full=1-11 with grub.cof	设置 CPU 1-11 为 dynamic tickless	×	×
磁盘	echo min_power >/sys/class/scsi_host/host*/link_power_management_policy	使用 ALPM 技术设置磁盘为 min_power 模式	×	×
	mount -o noatime /dev/sd* /dir	修改文件系统元数据的修改模式	×	×
	echo 3000 >/proc/sys/vm/dirty_writeback_centisecs	修改脏数据写到磁盘的时间间隔为 30 秒	×	×
	echo 20 > /proc/sys/vm/dirty_background_ratio	修改脏数据达到内存的百分比为 20%时才异步回写脏数据到磁盘。	×	×
网卡	ethtool -s eth0 advertise 0x002	设置网卡 eth0 的接口速度为 10Mb/s	×	×
	ethtool -C eth0 rx-usecs 1000	将 eth0 的 rx queue 的 timer 设为 1000us	↓20%	×
	ifconfig eth0 txqueuelen 5000	将 eth0 的 txqueuelen 设为 5000	×	×

注：

×：表示基本没有变化

↓：表示下降。

4 总结与展望

1） 使用 tuned 服务的 powersave 可以在性能基本不变的情况下，使得系统的电力消耗降低约 9%。

2） 上述的节电调优方法都没有使系统的整体电力消耗下降，可能跟测试工具 TPC-E 运行模式有关。根据不同的调优方法选择特定的测试工具，这些节电调优方法或许能够启动明显的节电效果。

3） 由于 TPC-E 的测定时间较短（10 分钟），这些节电调优方法没有起到累积效果，从而导致系统节电效果不明显。或许上述的这些节电调优方法在系统长时间运行的情况下因累积效应可以体现出节电效果。

4） 现在的 CPU C-State 状态无法对单个 Core 或者 Socket 进行设置。在下一代 CPU 中，或许能够对单个 Core 或者 Socket 的 C-States 状态进行设置，这样能够在完全不影响性能的基础上达到节电效果。

5 参考文献

https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Power_Management_Guide/index.html