Contribution ID: **22** Type: **not specified**

Don’t bake your Graphics cards!

Thursday, 17 September 2020 19:00 (20 minutes)

Well, gaming enthusiasts might have tried baking their graphics card while tinkering with their old graphics hardware but doing so is not a great idea though some people might still try it!

https://www.reddit.com/r/pcmasterrace/comments/6zfrf1/should_you_bake_your_graphics_card/

On the contrary, this document describes a power saving, thermal efficient feature known as AMD Zero Power Technology. Though most people like its synonym, BACO!

BACO - Bus Alive Chip Off

BACO is an idle state of the dGPU which is employed in idle scenarios for long idle power requirements. BACO is entered when dGPU has been idle for some time and display has gone blank or when there is no compute work load. Driver support is required to save the video memory and other required information as part of BACO entry sequence. More on it later.
There are other related features as well, similar to BACO.

BOCO: Bus Off, Chip Off. For Notebooks that support legacy S3 sleep. (ACPI)

BAMACO: Bus Alive, Memory Alive, Chip Off. For Desktops that support Modern Standby or Linux suspend to idle state or active S0ix states. (Connected Modern Standby)

BOMACO: Bus Off, Memory Alive, Chip Off. For Notebooks that support Modern Standby special case for deep S0ix states (Disconnected Modern Standby)

The purpose of this abstract is to outline the design and implementation details of AMD Zero Core power technology with focus on BACO.

BACO refers to a hardware state that allows the GPU to save as much power as possible on the graphics chip when it is not being used. The main purpose for this state is to keep power consumption in the dGPU as low as possible when it is not being used while keeping the PCIe configuration space alive. Keeping PCIe configuration space alive maintains device presence for the Linux kernel. The System Management Unit (SMU) implements the actual entry-exit sequence algorithm and it needs inputs from amdgpu driver to detect idleness and trigger entry and exit events. When in BACO, SMU turns power off for as many IP blocks, as possible and gates most PLLs such as SPLL, DPLL etc, but keeps for the bus interface intact. BIF maintains PCIe configuration space and OS configuration requests. BIF switches its clock to PCIe ref clk while GPU is in BACO state. In that state, only a part of the SOC logic remains on such as Thermal, SMBUS interface, DFX and most part of NBIO.

amdgpu driver supports Linux runtime power management framework already and the efforts to integrate goodness of BACO with it were going on from quite some time now https://lists.freedesktop.org/archives/amd-gfx/2019-February/031552.html . AMDKFD driver which is a part of amdgpu driver and is used for compute and machine learning workloads, didn’t implement support for runtime power management but with recent kernels that dependency is resolved. https://lists.freedesktop.org/archives/amd-gfx/2020-February/045504.html

Here’s is a plausible scenario that explains how BACO fits into standard runtime pm framework and this should also apply to suspend-to-idle flow w.r.t amdgpu driver.

When there is idleness in the system i.e there is no CPU bound work load the CPUs tend to go to deeper sleep states a.k.a C-states and when the devices such as GPU don’t have any graphics or compute kernel to act upon they go to device idle states called D-states. D-state can have sub-states like D0, D1, D2, D0i3, D3Hot, D3Cold, and D3 with D0 as active state and D3 as most power efficient state. Devices that support runtime pm can announce their capabilities in their PCIe config space and they write 1 to their PMCSR bits when in a D state like D3. How a device implements its sleep state is up to the device and one such power optimization is BACO that works with D3Hot / D0i3 states.

SMU IP Block

Platform Security Engine (PSP)
Power Management Engine (PMU)
Integrated Sensor Hub

SMU IP block contains three important sub units and one of those is known as PMU that implements BACO entry / exit sequencing.

BACO Entry Sequence at a high level

Amdgpu driver notify SMU it would like to enter BACO.
If SMU detects GPU is actually not at idle, it doesn't respond to the driver, hence no trigger for BACO entry.
If SMU finds GPU to be idle, it issues an interrupt to driver to start BACO entry process, followed by below steps.
driver take below steps as the real start of BACO entry :
FB (Frame Buffer) content saving
Put THERMAL PHY at low power mode to save BACO mode power
For any scenarios if enabled displaypll, can put displaypll at low power mode as well
Enable doorbell_monitor. More details on it below.
program Dstate change to use bypass mode; hereafter, any Dstate change would be handled by BIF.
driver signals BACO Entry Event to SMU firmware
SMU firmware take below steps and many more :
disables all BACO domain IP's related rSMU interrupt
ramp downs / gates PLLs.
turns off voltage rails to quiescence the device.
Since the BACO state cuts the power on video memory, we have to make sure all contents allocated in the video memory will be saved / restored properly. When changing BACO state, we need to check if audio device is busy for the following scenarios:

When audio device is busy, the GPU shouldn't enter BACO even the video device is idle.
When the GPU is in BACO, it should exit BACO if audio device starts working
BACO Exit Sequence:

For wake up sequence, the doorbell mechanism is used. The GPU doorbell mechanism that was introduced to the Volcanic Island family provides an application or driver to indicate GPU engine that it requires work on the HW. These doorbells can be issued from the software running on the CPU or on the GPU. The hardware supports doorbell mechanism by implementing a watch I/O memory mechanism that is programmed to recognize when a write happens to a special range of address. For BACO, this presents a new problem since these doorbell accesses cannot be detected by amdgpu driver if they originated from the software running on the CPU. Attempting to access the ASIC while it is in the BACO state will result in a hang. In order to prevent this from happening, there are two major design considerations:

The first is driver will now have to wait for ASIC idle status from SMU before entering BACO. This will prevent the driver from attempting to enter BACO when there is outstanding doorbell access. See above for entry sequence details.
The second is SMU will transfer control of monitoring doorbell activities to BIF when in BACO mode. This will allow BIF to detect any doorbell transaction and initiates an interrupt to the driver to exit BACO.
As BIF detects an incoming configuration cycle, it asserts a GPIO to wake-up the off power rails and the rest of the dGPU. A PCIe link training is not required after normal BACO exit. The doorbell monitor control is already transferred to BIF before ASIC entered BACO. While ASIC is in BACO, amdgpu driver gets notification of any doorbell activities from BIF via an interrupt. This triggers an event to wake ASIC up from BACO. Rest of the steps are nothing but unwinding steps of entry sequence described above.

MISC:

GPU reset using BACO.
https://github.com/RadeonOpenCompute/ROCK-Kernel-Driver/blob/7b4f1de71ea335e965fba590f4d030de52644137/drivers/gpu/drm/amd
/sys/kernel/debug/dri/N/amdgpu_gpu_recover can be used to manually trigger a gpu reset at the next fence wait and internally it may use BACO if applicable.
There are challenges with virtualization. Power management features such as BACO with Doorbell are not enabled with PCIe-SRIOV because of BIF ring buffer issues which is used for doorbell.
Important functions to look for:
bool amdgpu_device_supports_boco(struct drm_device *dev);*
*bool amdgpu_device_supports_baco(struct drm_device*dev);
bool amdgpu_device_is_peer_accessible(struct amdgpu_device *adev,*
*struct amdgpu_device*peer_adev);
int amdgpu_device_baco_enter(struct drm_device *dev);*
*int amdgpu_device_baco_exit(struct drm_device*dev);
int smu_baco_get_state(struct smu_context *smu, enum smu_baco_state*state);
int smu_baco_enter(struct smu_context *smu);*

*int smu_baco_exit(struct smu_context*smu);
bool smu_baco_is_support(struct smu_context *smu);
*bool amdgpu_dpm_is_baco_supported(struct amdgpu_device*adev)

GSoC, EVoC or Outreachy

No

Code of Conduct

Yes

Primary authors: Mr BHARDWAJ, Rajneesh (AMD); Mr DEUCHER, Alexander (AMD)

Session Classification: Main Track

Track Classification: Talk (half slot) (Closed)