# Intel SOC DVFS

Very high overview from an Intel guy.
(so you'll get where I'm coming from)

# Intel SOC's structure

- North complex DVFS implemented in P-unit
  - I.E.cpu AND graphics are controlled by P-unit FW
- South complex DVFS, clock and power gating is implemented (mostly) by SCU firmware.

- Kernel can send hints to P-unit and SCU but they make sure SOC always operate in valid voltage and frequency configurations.

## This simplifies OS enabling

- Freq and clocks are kept in valid ranges at all time.
- Complicated low power modes and dependencies are taken care of for the OS.
- OS developers don't need to know about very low level aspects of the chip.
- FW processing is more RT that the kernel and can do higher sampling rates of SOC performance counters than the OS can get away with while keeping power low.
  - This is how we keep our chips from smoking themselves.

## It works great until it doesn't

- Sometimes sub optimal when the governing FW makes choices based on CPU only.
  - GPU work loads that let CPU go idle may throttle memory bandwidth or let CPU go to a low frequency resulting in buffer under run in the graphics pipeline.
  - Security IP block is working while CPU is idle and the memory self refresh gets kicked to high latencies.
  - Critical sections where clock or power gating will make “fabric errors”
- Most work arounds are done using pm_qos to block some low power states.

## We love (and hate) our firmware.

- Its seen as a differentiator WRT OS enabling and hardware robustness.
  - The OS can try to wedge the CPU but the FW is supposed to make sure that never succeeds
  - Enabling low power states is simpler for OS developers.
  - Protect CPU from SW induced dammage
- We have plans to add more features to our FW for future SOC's.
- Yet its hard to get right and full testing seems to need a full OS running on top of it.