

# Autonomous CPU performance selection

## A new CPUFreq governor?

Sumit Gupta <[sumitg@nvidia.com](mailto:sumitg@nvidia.com)>



LINUX  
PLUMBERS  
CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026

# Why consider hw\_autonomous?

## ■ Load-based governors still calculate performance targets

Even with Autonomous Selection enabled, schedutil, ondemand and conservative continue evaluating CPU demand and sending Desired hints where supported.

## ■ performance does not guarantee MAX frequency

With Autonomous Selection enabled, the performance governor's request for maximum performance is a hint. The actual frequency may be below the policy maximum.

## ■ An explicit choice for each CPUFreq policy

Use hw\_autonomous to let hardware select a performance level without Desired requests. Other policies can keep their existing governors.

Fewer governor calculations and Desired writes may reduce overhead.

PCC systems may also save firmware mailbox commands. Potential savings not yet tested.



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



# Governors and Autonomous Selection

With `cpuc_freq`, existing governors can operate with Autonomous Selection disabled or enabled.

| Governor                            | Autonomous Selection | Desired performance                     | Meaning                                                                             |
|-------------------------------------|----------------------|-----------------------------------------|-------------------------------------------------------------------------------------|
| Existing governor<br>e.g. schedutil | Disabled<br>(Direct) | Valid value within<br>MIN / MAX         | OS requests a performance level                                                     |
|                                     | Enabled<br>(Guided)  | Nonzero                                 | OS supplies an optional<br>performance hint                                         |
| hw_autonomous<br>(proposed)         | Enabled              | Zero when the<br>register is<br>present | Platform selects performance<br>level for the workload without a<br>Desired request |

On platforms without a Desired register, Autonomous Selection is fixed on.  
Neither an existing governor nor hw\_autonomous can send Desired requests.



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



# CPUFreq driver models and scope

- Keep the `target()` based model so each policy can choose `hw_autonomous` or an existing CPUFreq governor.
- This requires new core/driver coordination to prepare performance controls and update policy limits.

| Driver                           | Mode             | API                      | Proposed scope           |
|----------------------------------|------------------|--------------------------|--------------------------|
| cppc_cpufreq                     | Direct / guided  | <code>target()</code>    | Initial driver           |
| intel_cpufreq                    | Passive with HWP | <code>target()</code>    | Potential future support |
| amd-pstate                       | Guided           | <code>target()</code>    | Potential future support |
| intel_pstate /<br>amd_pstate_epp | Active           | <code>setpolicy()</code> | Outside this proposal    |



# Proposed callbacks in struct cpufreq\_driver

| Driver callback        | When it runs                                     | Use in cppc_cpufreq                                                                                                                                                                                             |
|------------------------|--------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| prepare_perf_control() | With previous governor requests stopped.         | <b>For hw_autonomous</b> <ul style="list-style-type: none"><li>- Enable selection with no Desired request.</li><li>- Apply MIN / MAX limits.</li><li>- Apply EPP and Activity Window where supported.</li></ul> |
|                        | Before the selected governor starts or restarts. | <b>For existing governors</b> <ul style="list-style-type: none"><li>- Prepare the configured direct or guided mode.</li></ul>                                                                                   |
| set_policy_limits()    | After start and on policy MIN / MAX changes.     | Update MIN / MAX from the policy limits.                                                                                                                                                                        |

hw\_autonomous requires Autonomous Selection support and writable MIN / MAX.



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



# Switching to and from hw\_autonomous

Proposed sequence when preparation and governor start succeed

## SELECT HW\_AUTONOMOUS

01

Stop / exit previous governor  
Initialize hw\_autonomous

### prepare\_perf\_control()

02

Set MIN / MAX limits  
Keep Desired valid while  
Autonomous Selection is off

03

Enable selection & apply preferences  
Clear writable Desired to 0

04 · Start hw\_autonomous & apply limits

## SWITCH TO ANOTHER GOVERNOR

01 **prepare\_perf\_control()**

Prepare configured mode for selected governor

### Direct Mode

- Set valid Desired
- Disable selection

### Guided / Always On

- Keep selection on
- Set valid Desired if writable

02 Stop / exit hw\_autonomous  
03 Init / start governor & apply limits

Preparation runs first because hw\_autonomous issues no workload requests.



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



# Proposed auto\_select compatibility

auto\_select configures direct or guided operation for existing governors.

| Selected governor                     | Write auto_select = 0                         | Write auto_select = 1                            |
|---------------------------------------|-----------------------------------------------|--------------------------------------------------|
| Existing governor<br>(e.g. schedutil) | Select direct control.<br>Keep this governor. | Select guided control.<br>Keep this governor.    |
| hw_autonomous                         | Refused. Select<br>another governor first.    | Valid. Keep selection on.<br>Normally no change. |

Platforms with permanently enabled Autonomous Selection reject auto\_select = 0.



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



# Proposed CPPC EPP interface

- Prefer energy\_performance\_preference for names and numbers.
- Keep energy\_performance\_preference\_val for backward compatibility.
- Both update the same user preference, retained across governor changes.

| Accepted input      | Value  | Meaning                               |
|---------------------|--------|---------------------------------------|
| performance         | 0x00   | Favor performance                     |
| balance_performance | 0x80   | Intermediate preference               |
| balance_power       | 0xBF   | Stronger energy preference            |
| power               | 0xFF   | Favor energy efficiency               |
| default             | Varies | Firmware value captured at probe      |
| Numeric input       | 0–255  | Any value, even without a preset name |

The four named presets match amd-pstate.



LINUX

PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



# Discussion

- 01 Is a new governor the right CPUFreq interface for autonomous selection without governor-generated Desired requests?
- 02 If so, are separate preparation and policy-limits callbacks the right core/driver interface?
- 03 How should governor selection coexist with writable `auto_select` while preserving existing userspace behavior?



LINUX  
PLUMBERS  
CONFERENCE

PRAGUE, CZECHIA  
OCT. 5-7, 2026

# Thank you



**LINUX  
PLUMBERS  
CONFERENCE**

PRAGUE, CZECHIA  
OCT. 5-7, 2026

# Proposed CPPC mode reporting in cpupower

Combine existing information to explain who controls CPU performance.

| Interface                        | Information available                                                                      |
|----------------------------------|--------------------------------------------------------------------------------------------|
| Existing cpupower frequency-info | Driver, governor and policy limits.<br>EPP where the driver supports it.                   |
| Existing CPPC sysfs              | auto_select reports selection as 0 or 1.<br>cpupower does not read this field today.       |
| Proposed cpupower summary        | For CPPC, use driver, governor and auto_select to explain direct, guided or hw_autonomous. |

