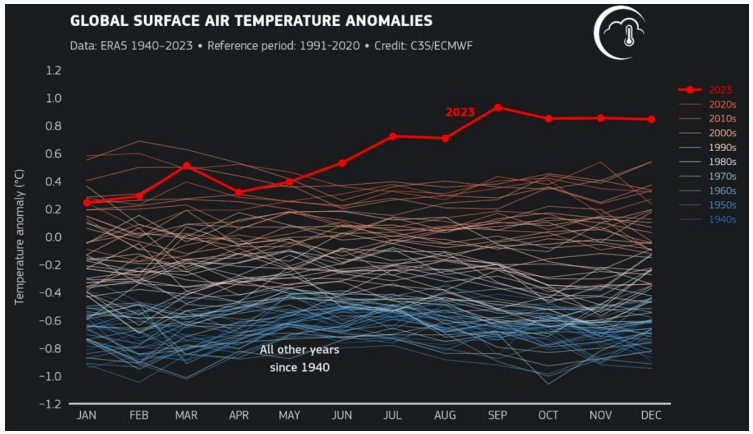
Linux Power Management Analysis for Embedded Systems

Hagen Paul Pfeifer <hagen@jauu.net>

Relevance of Power Efficiency in Embedded Systems

In addition to *environmental protection*, power efficiency is generally an important aspect in the embedded sector:

- **Battery Life**, Autonomy
- **Heat Dissipation**, especially in industrial environments, fan vs. fanless
- **Cost Savings**, reduced operational costs
- **Regulatory and Compliance Requirements**
- **Integration Densities**, product packaging



https://climate.copernicus.eu

I have decided not to generate a random dall-e3, midjourney placeholder illustration. Instead show a randomly picked climate anomaly illustration

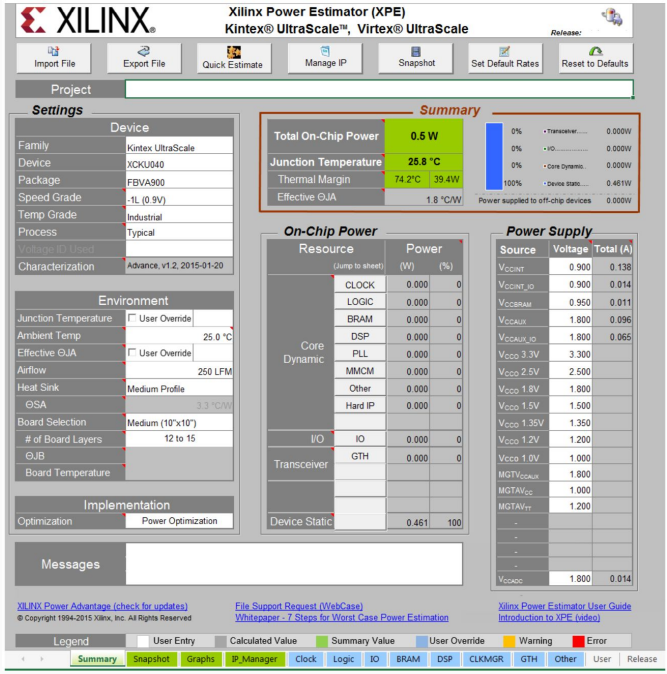
Reduce, Refine, Reassess: Power Tactics

To reduce power consumption:

- turn things off (e.g. idle cores, peripherals)
- turn things down (e.g. core and memory frequency)
- … do something more efficient (e.g. optimize algorithms, use energy-efficient programming languages)

Use vendor tools to create min-max design conceptualizations

- E.g. Xilinx Power Estimator illustrates very clearly how switching things off or down has a strong effect on power consumption. From DDR RAM clock rates over FPGA toggle rates to CPU core counts and so on



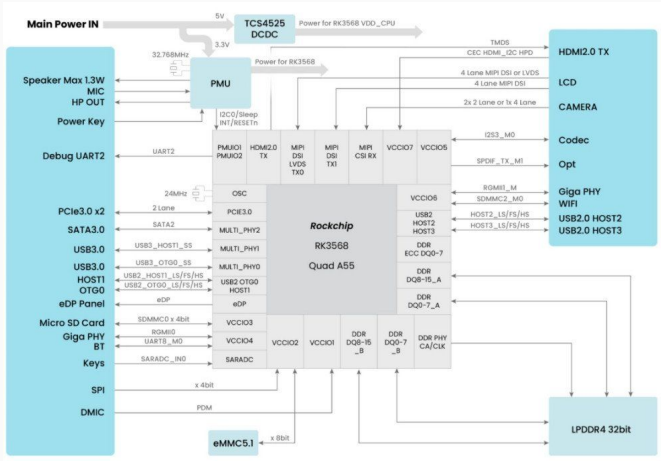
Xilinx Power Estimator Spreadsheet

What this talk does not address - Foundations

If the hardware has been poorly designed or selected, the software stack cannot overcome this

In product design, there are many decisions across many hardware levels

- **Architecture**: x86_64 vs. ARM Neoverse for cloud usage (e.g. „GFlops per Watt“)
- **CPU Supported Power Management Features**: sleep modes, DVFS , …
- **Deep Learning Interconnects**: PCIe vs NVLink vs CXL
- **Processing Platform:** Xilinx Ultrascale vs Intel Stratix for embedded products, FPGA vs ASIC
- **Type and configuration of memory**: DRAM, SRAM, Flash, etc.
- **Networking:** 802.3az, Energy Efficient Ethernet (EEE)
- **Converter:** Efficiency of a voltage converter
- **Cooling:** the higher the temperature, the higher the leakage currents



To simplify things: from here on, we assume that the hardware department has done a very good job! Finger crossed ;-)

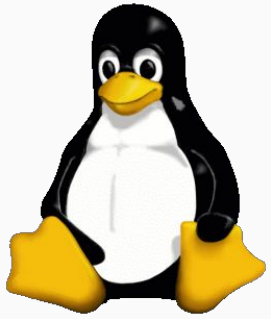
Toolset Turbocharge

There are a number of important tools & sources for the analysis:

powertop(1)
- dynamic real-time view of a system's power usage
- suggests optimizations that can lead to power saving

perf(1)
- Allows very detailed insights
- Not only pm events, but also task-relevant properties

sysfs(5)
- Virtual filesystem exports information about devices and drivers
- Standard interface to deal with many Linux PM subsystems



... and a number of specialized tools and custom scripts to handle the amount of data!

Task Characteristics

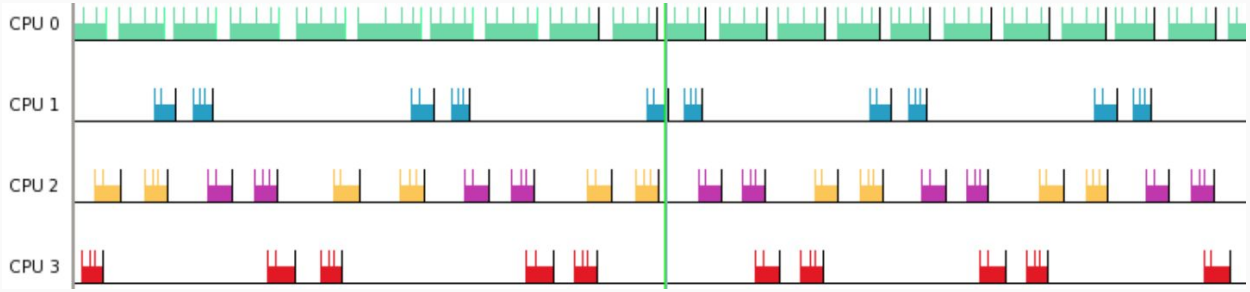
Jobs are executed as tasks on a CPU - developing an exact understanding of this is crucial

Task Characteristics

- How distributed is the load of specific cores, can affinity support for PM effects?
- Are energy aware scheduling optimizations for big.LITTLE systems possible?
- What is the ratio of executed and sleeping time at task level?
- Are task wake-ups globally time-aligned or completely random?
- Does task often wake up and only calculate for a short time?
- What does the task actually do during this time?
- ...

Appropriate measures can lead to significant optimizations at system level for power management!

Kernelshark visualizing scheduler tracepoints



Processor Frequency Scaling

Dynamic Voltage and Frequency Scaling (DVFS) is a power management technique where the voltage and frequency of a processor are adjusted dynamically

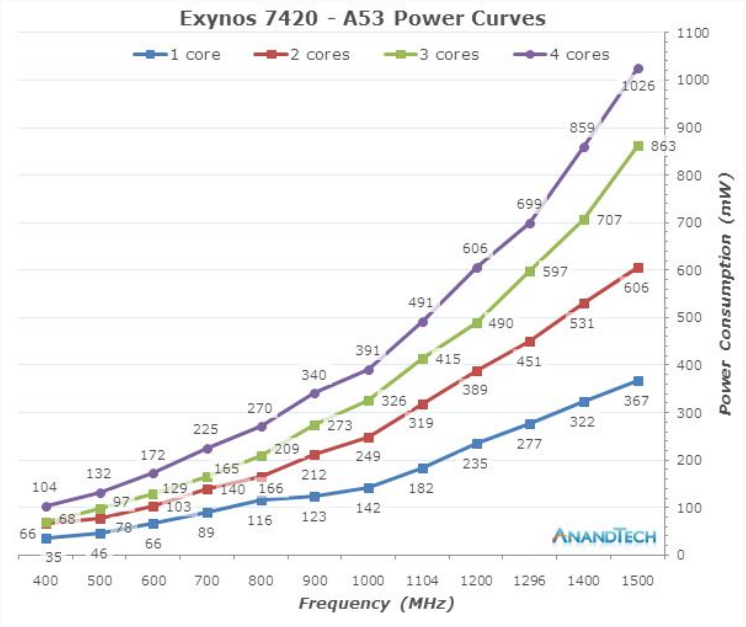
Subsystem divided into:

- CPUFreq Driver - interact with hardware
- CPUFreq Governor - set policies

Governors:

- Performance
- Powersave
- Userspace
- Ondemand
- Conservative
- Schedutils

Note: depending on the ARM implementation, a multicore system may share the same power and clock domain. Core individual controlling may not possible



Source: AnandTech

Controlling DVFS

CPUFreq subsystem is controlled via sysfs file system

```
# CPU Model: Armv8 Cortex-A76; Release: 6.6.20
# grep -r . /sys/devices/system/cpu/cpu0/cpufreq/
/sys/devices/system/cpu/cpu0/cpufreq/scaling_min_freq:1500000
/sys/devices/system/cpu/cpu0/cpufreq/scaling_available_governors:conservative ondemand userspace powersave performance schedutil
/sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_cur_freq:1500000
/sys/devices/system/cpu/cpu0/cpufreq/scaling_governor:ondemand
/sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_max_freq:2400000
/sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies:1500000 1600000 1700000 [...] 2200000 2300000 2400000
/sys/devices/system/cpu/cpu0/cpufreq/related_cpus:0 1 2 3
/sys/devices/system/cpu/cpu0/cpufreq/scaling_cur_freq:1500000
/sys/devices/system/cpu/cpu0/cpufreq/affected_cpus:0 1 2 3
/sys/devices/system/cpu/cpu0/cpufreq/scaling_max_freq:2400000
/sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_transition_latency:4294967295
/sys/devices/system/cpu/cpu0/cpufreq/scaling_driver:cpufreq-dt
/sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_min_freq:1500000
```

Note: tools like cpufreq-info(1) are wrapper around sysfs. Eye-candy included, use if you prefer

Monitoring DVFS

Kernel option CONFIG_CPU_FREQ_STAT provides basic overall statistics via sysfs

Use perf(1) to get detailed insight into all DFVS commanding with exact switching time, new target frequency on core basis

Record data

```
# perf record -a -e power:cpu_frequency -- sleep 10
[ perf record: Woken up 1 times to write data ]
[ perf record: Captured and wrote 0,110 MB perf.data (8 samples) ]
```

Note: Intel HWP ("Speed Shift") cannot be monitored by perf(1)

Analyze data

```
# perf script
 kworker/1:0-mm_      23 [001]  4615.651204: power:cpu_frequency: state=1800000 cpu_id=0
 kworker/1:0-mm_      23 [001]  4615.651208: power:cpu_frequency: state=1800000 cpu_id=1
 kworker/1:0-mm_      23 [001]  4615.651208: power:cpu_frequency: state=1800000 cpu_id=2
 kworker/1:0-mm_      23 [001]  4615.651209: power:cpu_frequency: state=1800000 cpu_id=3
 kworker/2:2-eve    1310 [002]  4615.747314: power:cpu_frequency: state=2400000 cpu_id=0
 kworker/2:2-eve    1310 [002]  4615.747317: power:cpu_frequency: state=2400000 cpu_id=1
 kworker/2:2-eve    1310 [002]  4615.747317: power:cpu_frequency: state=2400000 cpu_id=2
 kworker/2:2-eve    1310 [002]  4615.747317: power:cpu_frequency: state=2400000 cpu_id=3
```

Energy Aware Scheduling - big.LITTLE Systems

Increasing number of big.LITTLE systems require a suitable process scheduler

- Samsung Exynos 9 Serie 8895: 4 x ARM Cortex-A73, 4 x ARM Cortex-A53
- Google Tensor G3: 1 x ARM Cortex-X3, 4 x ARM Cortex-A715, 4 x ARM Cortex-A510

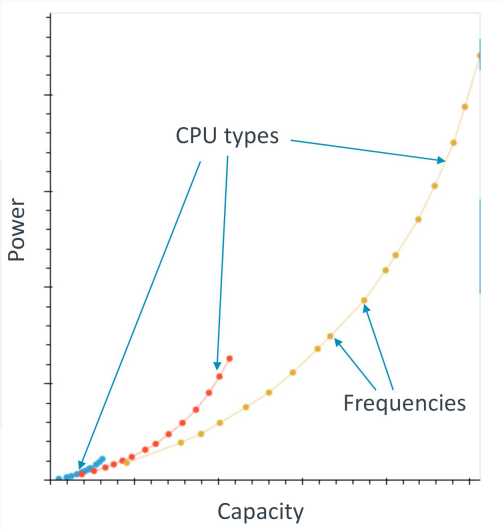
Linux support Energy Aware Scheduling (EAS), without big.LITTLE systems would be pointless

- CONFIG_SCHEDUTIL
- CONFIG_ENERGY_MODEL
- CONFIG_CPU_FREQ_GOV_SCHEDUTIL

Capacity: "amount of work it can absorb when running at its highest frequency compared to the most capable CPU of the system"

```
# record scheduling data
$ perf record -e sched:sched_switch,sched:sched_migrate_task -a -- sleep 10

# recorded, analyze and verify EAS decisions
$ perf script
```

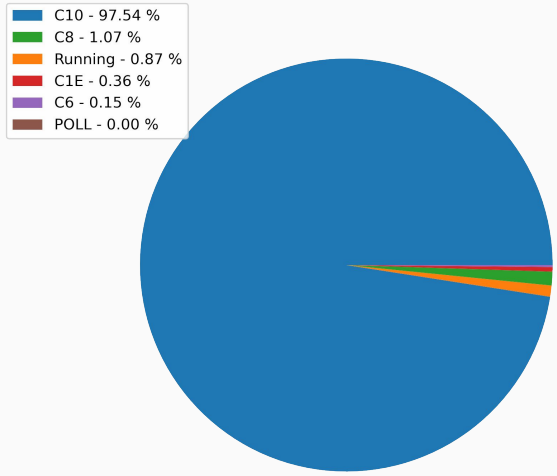


CPU Idle Subsystem

The CPUIdle subsystem on Linux is designed to manage idle CPU states to help reduce power consumption when the processor is not actively executing tasks

Drivers:

- Hardware interface: communicate with the hardware to apply changes in power states as decided by the governors
- Examples: acpi idle, intel idle



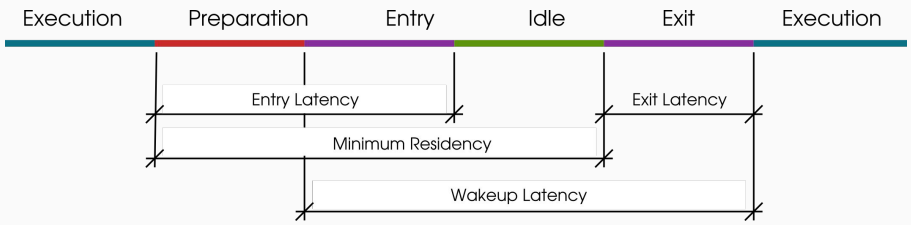
c-state distribution, idle system

Governors:

- Decision Makers: Determine when and how to change power states based on policies
- Examples: menu, teo, ladder & haltpoll

CPU Idle - Energetically Beneficial

Latency values within the device tree files makes the platform specific values known to the operating system



DTS file for Xilinx ZynqMP

```
idle-states {
   entry-method = "psci";

   CPU_SLEEP_0: cpu-sleep-0 {
     compatible = "arm,idle-state";
     arm,psci-suspend-param = <0x40000000>;
     local-timer-stop;
     entry-latency-us = <300>;
     exit-latency-us = <600>;
     min-residency-us = <10000>;
   };
 };
```

DTS file for Freescale i.MX8 Mini

```
idle-states {
   entry-method = "psci";

   cpu_pd_wait: cpu-pd-wait {
     compatible = "arm,idle-state";
     arm,psci-suspend-param = <0x0010033>;
     local-timer-stop;
     entry-latency-us = <1000>;
     exit-latency-us = <700>;
     min-residency-us = <2700>;
   };
 };
```

https://www.kernel.org/doc/Documentation/devicetree/bindings/arm/idle-states.txt

CPU Idle - Commanding & Status

Both CPUIdle Governors configuration and statistics can be queried via sysfs

Query statistics:

```
$ grep . /sys/devices/system/cpu/cpu4/cpuidle/state*/{name,time} | sort -n
/sys/devices/system/cpu/cpu4/cpuidle/state0/name:POLL
/sys/devices/system/cpu/cpu4/cpuidle/state0/time:83041750
/sys/devices/system/cpu/cpu4/cpuidle/state1/name:C1E
/sys/devices/system/cpu/cpu4/cpuidle/state1/time:488597510
/sys/devices/system/cpu/cpu4/cpuidle/state2/name:C6
/sys/devices/system/cpu/cpu4/cpuidle/state2/time:179749418
/sys/devices/system/cpu/cpu4/cpuidle/state3/name:C8
/sys/devices/system/cpu/cpu4/cpuidle/state3/time:1310566555
/sys/devices/system/cpu/cpu4/cpuidle/state4/name:C10
/sys/devices/system/cpu/cpu4/cpuidle/state4/time:80860014998
```

CPU Idle - Granular Tracking

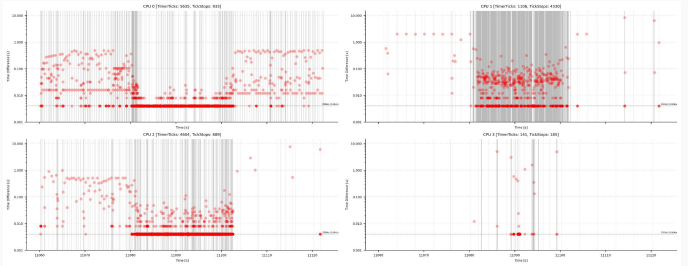
Individual C State transitions can be tracked with Perf

Command :

```
$ perf record -a -e power:cpu_idle -- sleep 60
$ perf script --gen-script python
$ vim perf-script.py
$ perf script -s ./perf-script.py
    2 36002.863851506 state=4294967295, cpu_id=2
   18 36002.863873513 state=4, cpu_id=18
    2 36002.864113542 state=4, cpu_id=2
   18 36002.864140040 state=4294967295, cpu_id=18
    2 36002.864272308 state=4294967295, cpu_id=2
   16 36002.864349085 state=4294967295, cpu_id=16
   18 36002.864381491 state=4, cpu_id=18
    0 36002.864396402 state=4294967295, cpu_id=0
   12 36002.864403081 state=4294967295, cpu_id=12
    0 36002.864415912 state=4, cpu_id=0
   12 36002.864447521 state=4, cpu_id=12
    2 36002.864472649 state=4, cpu_id=2
   16 36002.864490568 state=4, cpu_id=16
```

Note: state numbers are not C states! Numbers are kernel array index into states. Use sysfs for mapping information, see previous slide

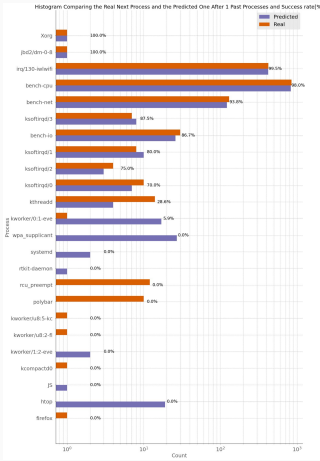
Scratching the Surface - There is More



Covered: task behavior, DFVS & CPU idle analyzation

But there is a lot more:

- Power management quality of service (PM QoS) & latency impact for (RT) workloads
- IRQ, kerneltasks and timer behavior is crucial for proper PM
- Thermal management, the cooler the package, the fewer leakage currents
- Suspend to ram, hibernate, hybrid sleep
- Regulator framework and PMIC drivers
- Generic power domain for managing multiple devices (power rails)
- ...



Research activities (ongoing, two master students attending *Eco-Compute 2024*):

- **eBPF based CPUIdle Governor**, for dynamic, userspace, targeted governor algorithms
- **Machine Learning based Task Prediction**, foundation for targeted (EAS) scheduling

Thank You!

Energy well spent! Thanks for sticking with me — questions?

For late questions or comments: hagen@jauu.net

Legal Disclaimer

This presentation is for informational purposes only and is not intended as an endorsement or promotion of the products or brands mentioned herein. The trademarks, logos, and service marks (collectively the “Trademarks”) displayed in this presentation are registered and unregistered Trademarks of their respective owners.

No affiliation or endorsement by the trademark owners is intended or should be inferred. The use of these Trademarks in this presentation does not imply any affiliation with or endorsement by their respective owners.

The views and opinions expressed in this presentation are those of the presenter and do not necessarily reflect the official policy or position of the brands mentioned. This presentation is not sponsored, endorsed by, or associated with any of the brands whose products are mentioned.

All information in this presentation is provided on an “as is” basis with no guarantees of completeness, accuracy, usefulness, or timeliness, and without any warranties of any kind whatsoever, express or implied.