Simple KernelCI Labs with Labgrid

Jan Lübbe – jlu@pengutronix.de
Agenda

• labgrid introduction
• scenarios with KernelCI
• example & discussion
labgrid

• python-based lab control framework
• automated testing (pytest)
• interactive developer access
• low-level bootstrap (incl. bootloader)
Layers

Protocol

Driver

Resource

CommandProtocol

Bootloader Driver | Shell Driver

Console Driver

/dev/ttyUSB4

Serial
Client-Side Configuration

YAML
● describes “Targets” with
● “Resources”
● “Drivers”
● HW/SW-specific parameters

⇒ The “Environment”
Test-Cases with pytest

- Test execution, selection and reporting is provided by pytest
- Fixtures provide access at different levels (command, strategy, target, env)
- pytest (and Python libs) makes it easy to prepare test data and analyze results

- Easy to integrate into other tools and CI

```python
def test_hwclock_rate(command):
    """Test that the hardware clock rate is not too inaccurate."""
    result = command.run_check('hwclock -c | head -n 3')
    hw_time, sys_time, freq_offset_ppm, tick = result[-1].strip().split()
    assert abs(int(freq_offset_ppm)) < 1000
```
KernelCI & labgrid

use KernelCI builds on hardware in local lab to
• find regressions (features & performance)
• give earlier feedback for upstream changes

add incentive to get defconfigs to work on custom HW
Proposed Workflow

• filter KernelCI builds and download artifacts
• run tests/benchmarks in the lab
• report results / compare with baseline
Example on BeagleBone Black
Example on BeagleBone Black

- upload kernel+dtb+initramfs via fastboot
- boot to shell
- run test cases
<table>
<thead>
<tr>
<th>Version</th>
<th>Configuration</th>
<th>Mean</th>
<th>Baseline</th>
</tr>
</thead>
<tbody>
<tr>
<td>v5.4-20187-g5734bb3b197e</td>
<td>staging-stable-20220909.0</td>
<td>70.417</td>
<td>100.0%</td>
</tr>
<tr>
<td>v5.4-20187-gac1844124765</td>
<td>staging-stable-20220910.0</td>
<td>70.819</td>
<td>100.6%</td>
</tr>
<tr>
<td>v6.0-rc3-4736-ge47eb90a0a9a</td>
<td>next-20220901</td>
<td>34.798</td>
<td>49.4%</td>
</tr>
<tr>
<td>v6.0-rc4-206-gbfb8bcb45634</td>
<td>staging-mainline-20220909.8</td>
<td>34.659</td>
<td>49.2%</td>
</tr>
<tr>
<td>v6.0-rc4-286-gb5021cff88d3</td>
<td>staging-mainline-20220910.0</td>
<td>34.612</td>
<td>49.2%</td>
</tr>
<tr>
<td>v6.0-rc4-304-g516609f26401</td>
<td>staging-mainline-20220911.0</td>
<td>34.630</td>
<td>49.2%</td>
</tr>
<tr>
<td>v6.0-rc4-5671-g35e245acbcdf</td>
<td>staging-next-20220907.0</td>
<td>34.590</td>
<td>49.1%</td>
</tr>
<tr>
<td>v6.0-rc4-5854-g7172196f4de0</td>
<td>staging-next-20220908.0</td>
<td>35.167</td>
<td>49.9%</td>
</tr>
</tbody>
</table>
Open Questions

• Should these results be submitted to KernelCI?
• Which use cases/features would be useful for the KernelCI community?
Links

GitHub: https://github.com/labgrid-project/labgrid
Docs: https://labgrid.readthedocs.io/en/latest/
Chat: #labgrid on libera.chat, bridged to #labgrid:matrix.org
YouTube tutorials:
https://youtube.com/playlist?list=PLPy6BX4pJosCEq7CIU06bt2WM7lFAy1CF
Linux Plumbers Conference

Dublin, Ireland  September 12-14, 2022