LOONGSON HOBBYISTS
COMMUNITY
Collaborate / Architect / Forward

LoongArch Biweekly

Session #1 (May 13, 2026)

Discord Server

@LOONGSON_USERS

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Before We Start

- This meeting is organised by Loongson Hobbyists' Community, a 3rd-party organisation unaffiliated with Loongson Technology or any of the companies, entities, and institutions mentioned in this document.
- Please limit discussion to open information only - we will not answer questions for products and designs currently under development or testing.
- Please refrain from making political statements or commentary for the duration of the meeting.
- Opinions expressed in this meeting are strictly of the individual speakers.

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

How This Meeting Works

- Most of the upstream updates comes from the Chinese biweekly, which took place last Sunday.
  - With additional updates from the past three days.
- If you would like to present, please contact for editing permissions.
  - Telegram: @JeffBai, Discord: @mingcongbai
  - Or simply shout out in the group chats!
  - You may edit until the sessions begin.
- Presentations will be in English, but Russian translation/clarification is available per request. We will have a short Q&A after each segment.
  - If you would like to help in other languages, please let us know.
- Meetings will be live streamed on YouTube…
  - … and will be recorded and made available on YouTube, VK, and Bilibili.

LOONGSON HOBBYISTS
COMMUNITY
Collaborate / Architect / Forward

Upstream Development

Binutils

- Zhou Zhao fixed an issue where the GNU Assembler (gas) may not align LoongArch instructions when alignment was not explicitly specified.
  - The issue was addressed by forcing alignment by 4 bytes (word-aligned).
  - This fix was needed as unaligned instructions will cause instruction fetch errors on LoongArch (AArch64 requires alignment by 4 bytes, and RISC-V (with the C extension) expects 2 bytes).
  - Technically the fix is not strictly needed as the compiler always emits a .align directive for the .text section, and experienced developers have also learned to write it. But a “Segmentation Fault” may surprise new developers.
  - On the other hand, having an assembly file working “fine” with gas >= 2.44 but triggering “Segmentation Fault” with gas <= 2.43 will be still surprising :(
- Qinggang Meng then added a test case insn_align_4.s to verify this fix.

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

GCC (16.x)

- GCC 16.1 has been released, the following should land in 16.2.
  - However, we do recommend that distros backport the following fixes accordingly.
- Lulu Cheng:
  - Fixed PR 125057: An ICE (Internal Compiler Error) caused by incomplete split conditions in loongarch_split_vector_move - GCC 16 added an vector move __lasx_cast_128 (moving from 128-bit to 256-bit vector), which was not accounted for in RTL, causing it to be treated as a move from GPR (General Purpose Register) to FPR (Floating-Point Register).
- xry111 (Xi Ruoyao):
  - Fixed PR 125049: The lifetime for the true value and the address in which it was stored in stack canary (as part of the implementation for stack protector in GCC) was too long, making it possible for attackers to read the value residing in the register or, when the register pressure is high, the attacker may overwrite the address spilled onto the stack, and thereby rendering the protector null.
  - Allow using %c placeholders to output symbol names in inline assembly (the extended top-level assembly introduced in GCC 16 is likely to make use of this).

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

GCC (16.x, issue)

- Yang Yanjun (setarcos) reported that GCC 16.1 crashes (with an ICE) whilst compiling VTK 9.6.1
  - Not reproducible on the latest releases/gcc-16 branch.
  - Bisection pinpointed r16-8886 as the commit which made the ICE go away.
  - The commit is the fix of the target-independent bug PR 125185, so this issue highly likely a case of PR 125185 (not confirmed by the reporter yet).

GCC (17.x)

- xry111 (Xi Ruoyao):
  - Eliminated an extraneous sign extension instruction (slli.w …, 0) after the ctz.w instruction, which extends 32-bit integer CTZs.
  - Introduced spaceship operation, which optimizes (a == b) ? 0 : (a < b) ? -1 : 1 as two slt[u] instructions and a subtraction instruction.
    - Lulu Cheng pointed out in her review that the SRP_SIGNED_AND_UNSIGNED marker for the result may be incorrect and needed revising.
  - Introduced an optimization (mimicking RISC-V zbb), revising a ^ b ^ (a | c) as (c & ~a) ^ b, in order to make use of the andn (AND NOT) instruction
    - Lulu Cheng and Waterman pointed out that the commit message contained errors and needed revising.

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

LLVM

- lrzlin (Runze Lin):
  - Introduced custom lowering for LSX vector sign extension - by combining vslti and vilvl/vilvh instructions, LLVM should now be able to generate more efficient instruction sequences.
  - Added test cases for LSX/LASX extension operations for sitofp and uitofp, covering multiple bitwidths and vector lengths.
    - CI checks pointed out that the test case uses deprecated undef statements, revision needed.
- heiher (Rui Wang):
  - Introduced vector add/subtract (ADD/SUB) support for vNi128 types. LLVM currently supports ADD/SUB for both v1i128 (one 128-bit integer in a 128-bit vector) and v2i128 (two 128-bit integers in a 256-bit vector) - doing so will allow LLVM to generate native vector ADD/SUB instructions with Q-sized elements.: VADD.Q, VSUB.Q (LSX), and XVADD.Q, XSUB.Q (LASX).
- CSharperMantle (Rong Bao):
  - Implemented stack clash protection (-fstack-clash-protection) for LoongArch (with references to RISC-V backend implementation), with the same same allocation-unrolling strategy for const-sized allocations.

Rust

- heiher (Rui Wang):
  - Mostly internal changes.
  - Refactored the portable SIMD helper naming and intrinsic paths, renaming SimdL as SimdExt, and categorizing code paths as is:: (internal intrinsics), cs:: (core SIMD modules), and ls:: (native LoongArch SIMD helpers).
  - Refactored implementations for SIMD bit operations, such as vbitclr, vbitrev, vbitset, etc., they all now use crate::intrinsics::simd.

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Linux Kernel

- Platform Support
  - **Huacai Chen:** Check for ACPI PCIH markers for each PCIe root bridge, skipping unneeded memory resource fixup (revision 1)
  - **Hongliang Wang:** Add clock properties and adjust bus frequency for i2c-ls2x (Loongson 2 Family I2C controller) (revision 3)
  - **Qunqin Zhao:** Make Security Engine check for interrupts on node0 (revision 2)
  - **Binbin Zhou:** Loongson CAN-FD bus (revision 1)
- KVM Subsystem
  - **Tao Cui:** Simplify _kvm_pte_init() with memset64() (revision 1, rejected for potential performance regression)
  - **Qiang Ma:** Cap KVM_CAP_NR_VCPUS (recommended max number of vCPUs) with KVM_MAX_VCPUS (max vCPU for the platform), preventing invalid configurations (revision 2)
  - **Bibo Mao:** Move some variable declarations into paravirt.h to prevent compiler warnings when !CONFIG_SMP, as asm/qspinlock.h is not included on such configurations (revision 1)
  - **Bibo Mao:** Remove unneeded interrupt injection when software timer expires (revision 1)

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Linux Kernel

- Other Code or Functional Fixes
  - **Huacai Chen:** Enable CONFIG_64BIT by default, as v7.1 introduces 32BIT as an option, causing existing configurations (defconfig's) to break (revision 1)
  - **Huacai Chen:** Specify -m32/-m64 with cc-option to fix build on Clang (revision 1)
  - **Huacai Chen:** Fix SYM_SIGFUNC_START definition on 32-bit builds (revision 1)
  - **Wentao Guan:** Fix an issue where the loongson_gpu_fixup_dma_hang() quirk may cause ADE access exceptions (revision 3)
  - **Rui Wang:** Move KASLR logic into EFI stub, mitigating memory segment overlaps between initrd and boot services/kernel (which may cause boot failures) (revision 4)
- Other Code Refactoring and Clean-ups
  - **Qiang Ma:** Remove unused cpu_has_perf definition (revision 1)
  - **Qiang Ma:** Simplify show_cpuinfo() logic (revision 1, rejected as invalid)
  - **Qiang Ma:** Fix definition for CSR_CPUID_COREID{,_WIDTH} (revision 1, rejected as invalid)

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Box64

- v0.4.2 was released, quickly followed by a hotfix release v0.4.3-1.
- Continued work to improve to the unit testing infrastructure.
- Fixed several instruction translation errors in the LoongArch JIT.
  - … with help from the improved unit test infrastructure.
- Optimized the performance of the C-based PCLMUL emulation function.
  - LoongArch lacks a carryless multiplication instruction, and the previous software emulation performed poorly.
  - References OpenSSL's implementation and adopts a 4-bit lookup table method to improve performance.
- Fixed minor typos in library wrapping.
- Added Chinese translation of the environment variable usage documentation.
  - Groundwork for the upcoming graphical configuration tool, for which we plan to add multilingual support.
- Fixed several issues with the code file caching mechanism.
  - Goal: Polish this experimental feature and enable it by default before the next release.
- Changed the deferred EFLAGS computation from a C function to JIT inlining.
  - This function can be a hot spot in certain games.
  - JIT inlining reduces context switch overhead.
  - Makes full use of the LoongArch LBT (Loongson Binary Translation) extension to further improve performance.

UEFI (EDK II)

Loongson's UEFI firmware is based on EDK II, generic platform support are gradually being upstreamed by Loongson engineers.

- kilaterlee (Chao Li) fixed and optimized PC related relocation, in order to support cases where HI is not adjacent to LO and where one HI corresponds to multiple LOs
  - He also added with this pull request support for CLANGDWARF compilers, making it possible to build the UEFI firmware with Clang. This should enable UEFI builds with Link-Time Optimization enabled, with potential benefits in binary size.

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Other Updates (Feature Request/Issues)

This segment is dedicated to communication of potential feature requests and may help find your next task ;-)

- bh1xaq (XiaoTan) requested for full LoongArch support in BambuStudio, a slicing software for 3D printers. The application currently builds on LoongArch but lacks support for the binary-only network plugin.
  - It is then requested that the upstream would help support it.
  - bh1xaq also pointed out that there is an open source alternative (open-bambu-networking).
- wojiushixiaobai (Xiaobai Wu) reported an issue to python-build-standalone, where their pre-compiled Python binaries will segfault on launch, as the ELF alignment (by 4KiB pages) does not fit the more common 16KiB pages.
  - **Upstream fixed this issue as of last week by updating patchelf to 0.14 and setting the default page alignment to 64KiB, as is default with recent Binutils versions.**

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Other Updates (Contributions)

- MarsDoge and yetist fixed an issue where gnu-efi may be compiled with LSX/LASX instructions, which is not supported by Loongson's UEFI firmware and will cause an assertion error on boot.
  - This issue was fixed by specifying the -mno-lsx and -mno-lasx compiler flags.
  - **This patch is currently under review and could use test results.**
- phprus (Vladislav Shchapov) submitted LoongArch support for zlib-ng's /delta CI workflow.
  - The workflow has now been merged, reports for compression ratio, binary sizes, symbol sizes, etc. will also be reported for LoongArch by maintainer request to help with code review.
- zevorn (Chao Liu) added LoongArch support for machina, a full-system emulator with JIT written in Rust. The emulator is capable of booting Linux kernels (MMU enabled) and supports peripheral devices.


ホーム
POINT
30/30 (100.0%) Battery: 1%
リズムゲーム
キャラクターセレクト
DIVAルーム
ショップ
リズムゲームをプレイすることができます。
:移動
:選択
SELECT:キャラ / カメラ位置リセット
:キャラ回転 / ズーム

LOONGSON HOBBYISTS
COMMUNITY
Collaborate / Architect / Forward

Distro/OS Updates

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Tips for Maintainers

- Consider backporting the aforementioned fix for GCC PR 125049 (fix will be backported to 16.2, 15.3, 14.4, and 13.5, distros using GCC 12 will have to handle the backport locally).
  - **Affected packages must be rebuilt to get hardened.**
- A commit in Linux 6.19 removed a *temporary* workaround for FPU exception in dml21_map_dc_state_into_dml_display_cfg().
  - The *final* fix is now part of Linux 7.1 (currently in the RC phase), meaning that AMD RDNA 4 has been broken on 6.19 and 7.0.
  - This could have been caused by a mix-up when AMD engineers submitted internal patches to the upstream.
  - Ruoyao Xi is considering re-introducing the temporary workaround or, alternatively, backport the final fix currently in 7.1.

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

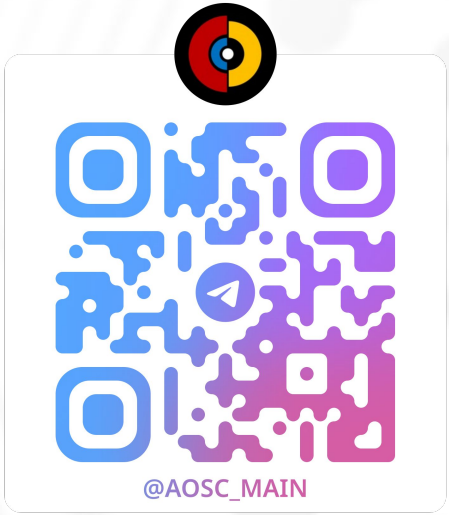
Linux From Scratch

- A custom LFS 12.0 build was made for Chips & Cheese to build LoongArch toolchain binaries for SPEC CPU2026 (and potentially submission to SPEC for inclusion in the next update).
  - We chose LFS 12.0 as it represents some of LoongArch's oldest upstream baselines - glibc 2.38 and GCC 13.2, allowing for the pre-built binaries to run on most distros.
  - Please note that this toolchain does not affect benchmarks - it contains tools to run the benchmark on the target device.
- Added clarification on bootloader/firmware support, distinguishing it by boot protocol and not simply the concept of "new/old worlds", avoiding conflation between ABIs and boot/firmware implementations.

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

AOSC OS

- Linux 6.18.27 was pushed to stable, containing security fixes.
  - **Dirty Frag (two LPE vulnerabilities) and Copy Fail (LPE).**
  - Please update at your earliest convenience (AOSC OS installed on server, multi-user, and shared devices should be updated immediately).
- Linux 7.0 is now available for testing.
  - Contains latest LoongArch features such as optimisation for 128-bit atomic operations.
  - Known issues: Intel GPUs do not work (blank screen with a flood of GPU operation timeout errors), AMD cards may show graphical artifacts under higher CPU loads (?!).
  - **oma topics --opt-in linux-kernel-7.0.3**
- OpenJDK 25 is now available as the default OpenJDK version.
  - Significant performance uplift observed in Minecraft and Ghidra.
- **19** security advisories has been released in the past two weeks, **15** of which contains high or critical severity vulnerabilities!



LOONGSON HOBBYISTS
COMMUNITY
Collaborate / Architect / Forward

Community News & Events

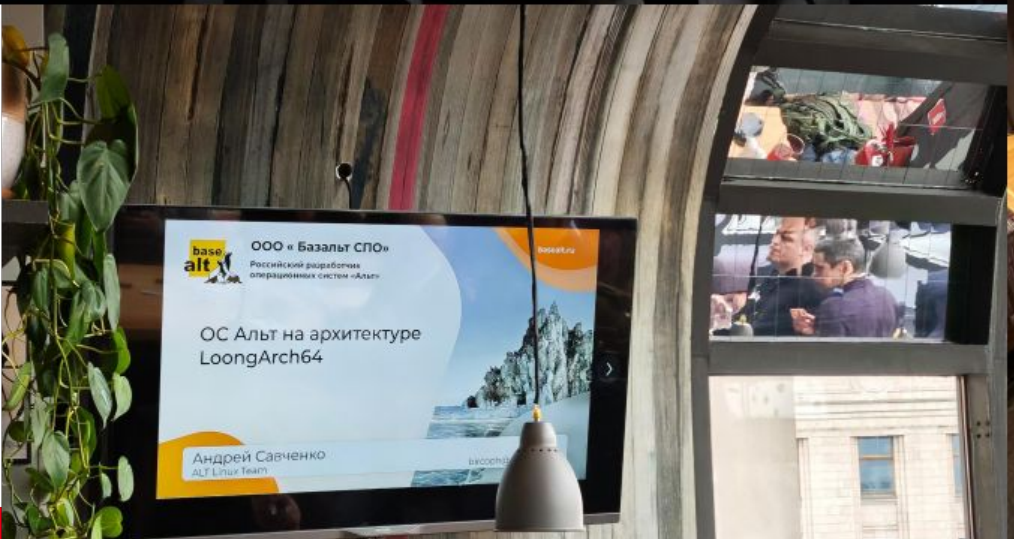
LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Recap: Trip to Moscow

- April 12-18, our community staff made a trip to Moscow.
  - We visited Tramplin Electronics and supported their stall at Expo Electronica - we built an Irtysh C616 gaming machine with Box64.
  - We gave two talks at Irtysh Meetup ‘26 on community building and binary translation.
  - Lots of new friends and new information (and to-dos)!
- Many of our colleagues in Russia are enthusiastic about LoongArch…
  - … But they needed more technical support to make better use of LoongArch devices.
  - **We need more technical exchange sessions like the biweekly meetings!**
  - ALT Linux and ROSA has good knowledge on LoongArch and have successfully built some of the earliest upstream-based LoongArch distros. **But they lacked knowledge on non-upstream fixes and platform-specific quirks.**
  - Manufacturers, likewise, needed information on peripheral support and firmware updates.
  - Box64 was a highlight, attracting attention from hobbyists, manufacturers, and media.


Трамплин
Электроникс

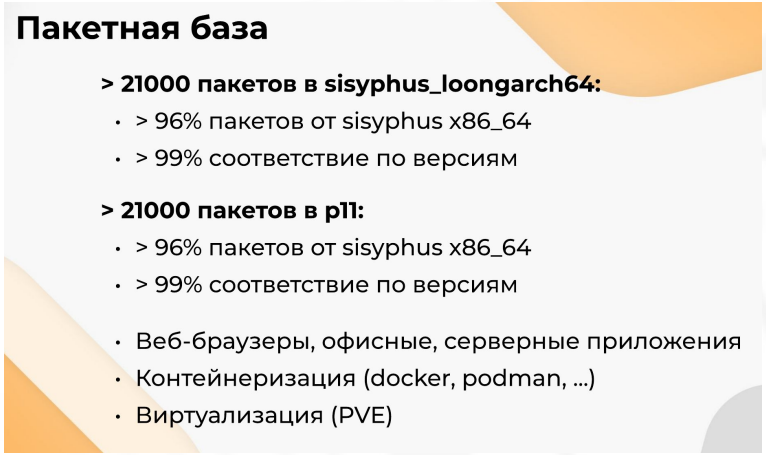

иртыш


ООО « Базальт СПО»
Российский разработчик операционных систем «Альт»
ОС Альт на архитектуре LoongArch64
Андрей Савченко
ALT Linux Team

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

ALT Linux's Feedback

- Porting work started in 2023, LoongArch has performed well and has now reached **> 95% package coverage in their repositories**.
- However, LoongArch devices is still 2-3x slower than build servers for the "established ports" (x86-64, i586, and AArch64).
  - Andrei mentions that ALT Linux was using 3C5000L servers, which are known not to scale too well with multi-core workloads. Tramplin has sponsored Irtysh C632 servers, which should have been (much) more performant, but they lacked the RAM modules to put it into use (the 4-Rank modules they own are not supported).
- LoongGPU has been a trouble for being closed-source and buggy.
- LoongArch is currently (therefore) considered one of the "catch-up ports" along with riscv64 and e2k (Elbrus), effectively tier 2 architectures.



LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Feedback from Other Manufacturers

- Support for Mellanox NICs seems iffy, would be good to have official words on compatibility and other notes.
- VPP (Virtual Packet Processing) port is sorely needed for software-based router and firewall solutions - the current port builds but lacks LoongArch SIMD optimisations.
- Firmware for motherboards, especially those from ASUS, are hard to find.
- Database solution company XSquare is optimistic about LoongArch but hopes for more attention on runtime SIMD/assembly optimisation.
- It would help to have more consolidated channels for technical information and latest updates… ***we are working on it!***

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

Shameless Plug

Please visit [loongfans.cn](), our community portal! French, German, and Russian translation forthcoming…

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

How to Read and Understand UEFI Logs

...and Why It Will Save Your Life

(Speaker: Maria)

LOONGSON HOBBYISTS COMMUNITY
Collaborate / Architect / Forward

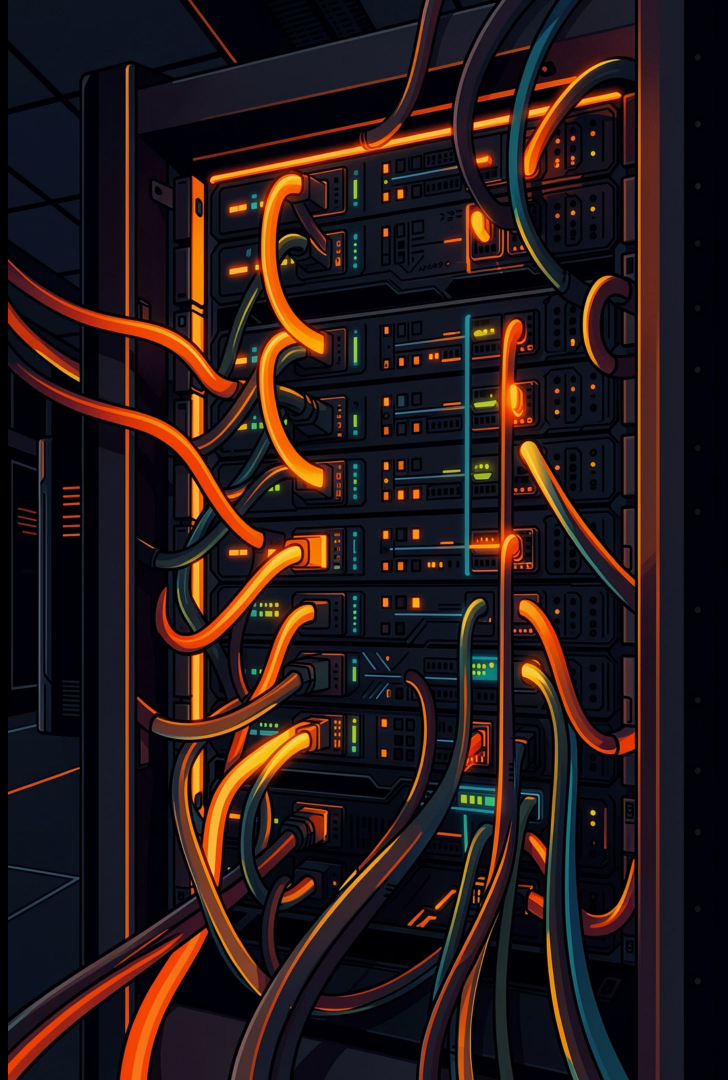
Executive Summary

- This presentation teaches embedded systems and firmware developers how to read and understand UEFI logs output through a serial port (UART) from the very first microseconds after power-on. Using a real Loongson platform example (the putty.log file), we walk through the entire boot path: from cold start and first POST codes, through CPU core wake-up, high-speed link training, problematic PCIe port detection, and chipset initialization. The key takeaway: instead of guessing why a board won't boot, you can find the root cause in the log in 5 minutes and calmly go have some tea.

How to Read and Understand UEFI Logs

And Why It Will Save Your Life

FULL BREAKDOWN USING THE LOONGSON PLATFORM AS AN EXAMPLE



What This Talk Is About and Why You Need It

The Problem

As a developer of embedded systems, firmware, or low-level software, you often face: **"The board won't boot."** LEDs blink, fans spin, screen is blank. Guess? Swap the board? Re-solder the processor?

The Solution

The only reliable way to understand what is happening in the earliest boot stages is to read the **UEFI logs via a serial port (UART)**.

The Goal

1. Understand the structure of a UEFI boot log on the Loongson platform
2. Identify signs of problems and understand their causes
3. Connect log messages to specific files in the EDK2 source code
4. Apply this knowledge for debugging, optimization, and development

Instead of guessing for hours, you can determine from the log in **5 minutes** that the board hung due to a poorly seated memory module — and go have some tea instead of re-soldering the chipset.

Toolkit: How to Get the Log

1.1 UART

Any serious embedded board has a **UART** (Universal Asynchronous Receiver-Transmitter) connector. It works from the very first microseconds after power-on - long before video, USB, and network are initialized.

What You'll Need

- A USB-UART adapter (e.g., CP2102 or FT232 chip)
- Three wires: TX, RX, GND
- Terminal program: PuTTY, minicom, screen
- Settings: 115200 baud, 8 bits, 1 stop bit, no parity

1.2 POST Codes: Language of a "Silent" System

Before video initialization, UEFI outputs POST codes to port `0x80` and duplicates them on UART. The last POST code tells you exactly where the system hung.

Range	Phase
0x00–0x0F	SEC Phase (very early init)
0x10–0x1F	PEI Phase (before memory init)
0x20–0x3F	PEI Phase (after memory init)
0x40–0x4F	DXE Phase
0x50–0x5F	BDS Phase

Log Anatomy: SEC Phase - A World Before Memory

We trace the **entire boot path** from the first processor instruction to the BIOS Setup menu, analyzing `putty(1).log` line by line.

Stage 0: Cold Boot and First Bytes

```
=~= PuTTY log 2026.03.18 11:03:46 =~=
Shut down slave cores done!
```

The message `Shut down slave cores done!` is the **tail of a previous boot** that ended with a software reset - this is the second boot iteration.

SEC Phase Highlights

```
POSTCODE=<0x00>
CPU Type: 3C6000/D
AVS: Get Vddn value is: 0000044c
CPU CLK SEL : 00000004
```

- **CPU Detection:** Firmware identifies the processor model to select the correct init code path
- **AVS:** Adaptive Voltage Scaling reads factory-calibrated values incorrect values cause instability
- **CLK SEL:** Frequency multiplier indexes, converted to real MHz later

Cache-as-RAM (CAR): The Critical Moment

During SEC, main DRAM is **not yet initialized**. C code needs a stack - so the system uses the **processor cache as temporary RAM**.

```
POSTCODE=<0x06>
Copy Sec code to SCache-As-Ram.
Copy PEI code to SCache-As-Ram.
POSTCODE=<0x07>
Entering C environment
```

If CAR setup fails, you will **see nothing**. No UART, no POST codes. This is the "black screen of death" at the firmware level - the hardest problem to diagnose.

PEI Phase: Waking the System

The most complex and dense stage: processor cores, inter-socket links, and the memory controller are initialized, and the chipset is discovered.

1

PEI Module Dispatcher

Finds **0x16 PEI FFS files** in the Firmware Volume. Each module is identified by a GUID - e.g., `9B3ADA4F...` = `PcdPeim.efi`. Use the GUID to locate the module in the EDK2 source tree.

2

Waking All Cores

`PreCpuMpPei.efi` wakes all cores: 4 nodes × 16 cores = 64 physical cores. With SMT: **128 logical processors**. A missing line in the log means a core failed to wake.

3

PHY Configuration

`PhyConfigPpi.efi` handles electrical parameters of high-speed links, dumping large PHY parameter tables for all lanes. Parameters may come from code, EEPROM, or previous training runs.

4

Inter-Socket Link Training

`retry to gen4 ..No.3 times link up GEN4 SUCCESSFULLY!` - The link failed on the first attempt. Only on the **third try** did Gen4 (~16 GT/s) succeed. Retries indicate marginal signal quality or suboptimal power/temperature.

PCIe Failures and the Saving Bridge

PCIe Disaster: Dead Ports

```
init pcie 2
ltssm 0x0
error!!! not link up!
```

`ltssm = 0x0` means **Detect.Quiet** - the receiver detects no signal. There is physically no device on the other end.

⚠ This is **not a firmware bug** - it's a feature of the TD622E0 board. Not all PCIe lanes are routed to slots. However, each failed attempt wastes **tens to hundreds of milliseconds**, adding seconds to boot time. A simple fix: add a list of disabled ports to skip them.

The Saving Bridge: 7A2000 Chipset

```
bridge init start ...
device linked in X8 mode GEN1
device linked in X8 mode GEN3
sub 7A version 0x0
```

After a series of failures, the **CPU-to-7A2000 Southbridge** link comes up: first at x8 Gen1, then transitions to Gen3.

This is one of the most important positive moments in the log. If this step had failed, we would see **nothing further** - no SATA, no USB, no video.

The Detective Story: Why Did the Reboot Occur?

The initialization sequence appears **twice** in the log. Here is the full chronology of events that explains why.

1. **Flash JEDEC ID Read**
   System reads the flash chip identifier during first boot.
2. **CMOS Cell 54 = 0x0**
   Cold boot from G3 (Mechanical Off) is confirmed. `Resume from G3. Boot with Full cfg.`
3. **Critical: Bit 10 is Set**
   `CmosRestoreFactoryInit` detects **bit 10 = 0x800** in CMOS 54 - interpreted as "load factory defaults
4. **Factory Reset Performed**
   All UEFI variables are reset. `Restore Factory Done, Start Reset!!!` software system reset is triggered.
5. **Second Boot Begins**
   The second half of the log is the clean boot with freshly reset settings. Mystery solved.

The cause of the set bit could be: a user action in the BIOS menu, a variable layout mismatch after a firmware update, or an **RTC power glitch**.

Conclusion and Practical Takeaways

1

UART is Primary

The UART log is your primary diagnostic tool when the system is silent. It works from the very first moment after power-on.

2

POST Codes

Instantly localize a hang to a specific phase: SEC, PEI, DXE, or BDS. The last code tells you everything.

3

CAR Failure = Silence

Cache-as-RAM failure means absolute silence. This is the hardest problem to diagnose — no UART, no POST codes.

4

GUIDs and PPIs

Trace any log message directly to a specific driver in the EDK2 source tree using the module GUID.

5

Links and Retries

Analyze "retry" and "not link up" messages - they point to hardware issues or simple boot-time optimization potential.

6

CMOS and NVRAM

Can cause mysterious reboots. Always check for reset flags if you see a repeating initialization sequence in the log.

> Instead of blindly re-soldering, you can now spend **5 minutes with the log**, find the root cause, and go have your tea. **Thank you.**

LOONGSON HOBBYISTS
COMMUNITY
Collaborate / Architect / Forward

Q&A

LOONGSON HOBBYISTS
COMMUNITY
Collaborate / Architect / Forward

Next Meeting

May 28th, 2026 @ 13:00 (UTC)

Discord Server

@LOONGSON_USERS

LOONGSON HOBBYISTS COMMUNITY

Collaborate / Architect / Forward