Symmetricom XLi GPS Week-Rollover Remediation

Author: zombu2
Date: 2026-02-11
Version: 2.0 (full technical rewrite)

Abstract

This document is a complete technical record of the XLi GPS rollover remediation effort. It covers firmware reverse engineering, runtime instrumentation, flash integrity behavior, failed approaches, validated recovery paths, and the final working deployment pattern. The report is based on two persistent evidence sources: an engineering runbook and durable session memory. It is written as an operator/engineering handoff intended to be reproducible on comparable units.

1. Objective and success criteria

1.1 Objective

Correct GPS-related calendar date regression (notably year 2006 or 1970 states) on XLi while preserving unit stability and maintaining a deterministic rollback path.

1.2 Success criteria

1. Date conversion path yields correct year under real operating conditions.
2. Unit remains serviceable (NI shell, web UI/auth, burn interface functional).
3. Procedure is reproducible and auditable.
4. Recovery to stock state is always possible.

2. Evidence corpus and reproducibility baseline

2.1 Primary firmware artifacts and hashes

- `192-8001V1_106.bin`
  SHA-256 `15765c562ed4dce60fbe7d5eeb63153f5531d43c06690e6d25148744425bafbb`
- `192-8000V1_106.bt`
  SHA-256 `2bcfdefdf12b12a2093a6b403416710ec037c249666c008ae46d747b3766232c`
- `184-8000V65.bin`
  SHA-256 `76f70a64982546d44c1e7b426404ade665193091a59a63b3d3f92956c509434b`
- `192-8002V1_106.fs`
  SHA-256 `f67bb2f1d99431697681c7104b26be15ec0597f197d44f2cfcd76cd2de9ec3e6`

2.2 Final patch artifact hash

- `192-8001-branchstub-17da20-lencrc.bin`
  SHA-256 `ace2ccc1ea9a72d76a2f454363a84dcee8477c7a97a0c1576cbfbd6b0b29fa33`

2.3 Runtime translation rule (critical)

For the app handler family used in this project:

- `live_runtime_addr = decompressed_app_offset + 0x1000`

Validated repeatedly at multiple patch points (gpsTime and rollover paths).

3. System architecture findings

3.1 Flash partition map (live authoritative)

From live `showFlashSizes()` and mapped flash reads:

- Bootloader: start `0x10000000`, size `0x000A0000`
- FPGA: start `0x100A0000`, size `0x00060000`
- Application: start `0x10100000`, size `0x004E0200`

- Filesystem: start `0x105E0200`, size `0x00200000`
- NVRAM bank B: start `0x107E0000`, size `0x00010000`
- NVRAM bank A: start `0x107F0000`, size `0x00010000`
- Total flash size: `0x00800000` (8 MiB)

Key implication: all partitions are memory mapped and can be read directly, so full forensics and backup are possible without raw programmer access.

3.2 Firmware packaging and decompression

Both software (`.bin`) and boot (`.bt`) images contain primary zlib streams at offset `0xA801`.

- App decompressed length: `2,809,624`
- Boot decompressed length: `1,178,200`

This explains why runtime signatures are not found by naive raw-image scans. Reverse-engineering and patch-coordinate derivation must be done against decompressed payloads.

3.3 Filesystem reality (critical correction)

`192-8002V1_106.fs` has OEM `VXEXT1.0` and parsed root entries showing only directories (`config`, `etc`, `web`) with cluster 33 and zero sizes in direct VXEXT parse.

However, deeper signature carving and backup FS analysis proved rich web content exists in live filesystem images. Therefore:

- The static parser result (“empty dirs only”) is incomplete for total content extraction.
- HTML/UI behavior is not safely corrected by simplistic `.fs` edits alone without matching live FS layout and integrity model.

3.4 FPGA image classification

`184-8000V65.bin` is a bit-swapped Xilinx bitstream (sync at offset `0x4` after reversal). It is configuration data, not VxWorks application logic. GPS rollover logic was not addressed in this file.

4. Reverse-engineering methodology

4.1 Static tooling pipeline

A repeatable static pipeline was built around decompression, inventory, callpath mining, binwalk recursion, and signature carving.

Core tools generated:

- `xli_firmware_autopsy.py`
- `xli_firmware_tool_inventory.py`
- `xli_firmware_deep_mine.py`
- `xli_firmware_runtime_compare.py`
- `xli_reverse_callpath_map.py`

Supporting outputs included JSON/CSV/Markdown reports for symbol inventory, address maps, and extraction manifests.

4.2 Dynamic validation pipeline

Live validation emphasized read-first introspection:

- `lkup` symbol discovery
- mapped flash reads (`d 0x10xxxxxx,...`)
- GPS wrapper probes (`GPSBp*`, `SA_M12_*`)
- NI burn command confirmation (`BH`, `bu`, `bf`)
- status and date-buffer byte validation

4.3 Correlation process

Cross-correlation linked static markers to live addresses and runtime behavior:

1. Locate candidate functions in decompressed app payload.
2. Translate by `+0x1000` into live address space.
3. Confirm exports in live `lkup` when available.
4. Validate behavior with controlled runtime patch or read-only probes.

5. High-value hidden/quirk findings

5.1 FTP fetch-name quirk

XLi FTP fetches required colon-prefixed filename staging on this unit.

Practical requirement:

- stage both `192-8001.bin` and `:192-8001.bin`

Without this, fetch behavior became inconsistent.

5.2 Telnet session-cap failure mode

Concurrent telnet sessions can exhaust NI session capacity and block control operations, especially around reboot/inject loops.

Operational rule adopted:

- one active burn/control session only

5.3 Bootline startup-script unreliability

`bootChange` startup-script persistence (`s=`) was not deterministic for this patch path on this firmware.
It cannot be the sole persistence mechanism.

5.4 Flash integrity gate behavior

Modified software images encountered repeat failures near sector 31 with post-program verification (`ERR 4`) even when simple trailer CRC looked valid.

Interpretation:

- integrity acceptance is stronger than one trailing CRC field
- image/partition validation logic enforces deeper consistency

5.5 Inline patch non-equivalence

An inline persistent patch at live `0x000b6d70` (`inline1024`) booted but produced incorrect date behavior (`1970`), proving semantic non-equivalence to the known-good runtime branch+stub fix.

6. GPS path reverse engineering

6.1 Confirmed control surfaces

Exposed and exercised:

- `GPSBpSendMsg`
- `GPSBpGetMsg`
- `GPSBpStatus`
- `GPSBpLocked`
- `GPSBpOutputEnable`
- `SA_M12_Set_Get_GPS_Mode`

- `SA_M12_GetRcvrIDdata`
- `SA_M12_Clear_Nav`

6.2 Direct module interrogation

Read-only module identification calls succeeded (software/FPGA identity strings returned), proving live backplane communication.

6.3 GPS firmware write feasibility conclusion

No explicit module firmware erase/program API was identified in exposed `SA_M12`/`GPSBp` wrapper surface. Direct module reflashing remained unproven in this effort.

7. Patchpoint mapping and caller recovery

7.1 Concrete caller recovery

Recovered caller chains included:

- `FUN_00070d9c -> FUN_00035b68/FUN_00036108 -> GPSBpSendMsg`
- direct mode call site to `SA_M12_Set_Get_GPS_Mode`

7.2 Boot/mode hook map

Pinned patchpoint family (payload to live via `+0x1000`) included:

- dispatch pointer and handler entry for `FUN_00070d9c`
- mode argument/mode source instructions
- direct call to `SA_M12_Set_Get_GPS_Mode`
- candidate hook slot for injected call path

This mapping constrained safe experiments and reduced blind patching.

8. Experimental campaigns and outcomes

8.1 Runtime-only date correction (success, non-persistent)

Runtime branch/stub and equivalent RAM patch paths corrected date conversion in-session and validated root-cause path.

Limitation: state reverted on reboot.

8.2 Startup-script persistence attempt (failure)

On-box startup script + boot param edits were attempted for automatic reapply. Behavior was non-deterministic and at times destabilized network reachability.

Conclusion: rejected as canonical persistence method.

8.3 Direct hint injection campaign (transport success, functional failure)

A fresh 17-byte `@@Gb` hint payload was generated and injected repeatedly via:

- `/tyCo/1` serial writes
- `GPSBpSendMsg` backplane path

Transport evidence confirmed end-to-end command delivery, but date remained in rollover-error behavior for this unit state.

Conclusion: transport path valid, standalone hint injection insufficient.

8.4 Persistent software burn attempt with two-byte payload modifications (failure/side effects)

A modified software image was burned via corrected `burnHost/burnFile` call pattern. Side effects included web login loop in at least one run.

Recovery required stock software and, where needed, stock filesystem re-burn.

Conclusion: not production-safe.

8.5 Sector31/ERR4 gate encounter (failure)

Full rebuilt images repeatedly failed around sector 31 with integrity error. This established a hard boundary against arbitrary repack without satisfying hidden validation constraints.

8.6 Final branch-stub persistent image path (working)

A branch-at-site model with stub at live `0x0017da20` became the canonical accepted path on the validated unit state.

Canonical site details:

- patch site live `0x000b6d78`
- stock word `0xe51b3040`
- patched branch word `0xea031b28`
- stub returns to `0x000b6d80`
- embedded offset constant `0x24EA0000`

Reported working status:

- `site_is_branch=true`
- `site_branch_target=0x0017da20`
- `stub_pattern_ok=true`
- date path showed corrected year behavior under lock path transitions

9. Recovery engineering (what prevented permanent outage)

Reliable recovery baseline:

1. Re-burn stock software image (`BH` + `bu`).
2. If UI/auth damaged, re-burn stock filesystem (`BH` + `bf`).
3. Reboot (`BASET AUTO`).
4. Verify NI shell + web reachability + date behavior.

This baseline converted high-risk experiments into controlled, recoverable operations.

10. Canonical deployment procedure

Canonical package:

- patched software image (`branchstub-17da20`)
- stock software rollback image
- stock filesystem rollback image
- deploy automation script
- checksum manifest

Canonical control flow:

1. Verify checksums.
2. Stage both FTP names (`normal` and `colon-prefixed`).
3. Burn via NI in single session.
4. Reboot.
5. Validate patch markers + date behavior + UI.
6. Roll back immediately if any integrity/behavior regression appears.

11. Hard rules derived from evidence

1. Do not bypass flash protection unless sector ownership and free-space status are proven.
2. Do not run multi-session telnet burn workflows.
3. Do not treat “FLASH SUCCESSFULLY PROGRAMMED” as sufficient by itself.
4. Do not use inline `0x000b6d70` persistence variant for this issue.
5. Do not rely on boot startup script path alone for persistence.
6. Always keep stock rollback artifacts staged before mutation.

12. Limitations

1. A universal, integrity-clean repacker for all software-image variants was not finalized in this campaign.
2. A direct GPS module firmware programming path was not proven from exposed APIs.
3. Some behavior depends on firmware build and runtime service state; validation must remain per-unit.

13. Final conclusion

The final success came from combining disciplined reverse engineering with strict operational controls:

- static decomposition and callpath mapping
- live read-first probing
- explicit failure logging and rollback discipline
- selection of a patch model that matched proven runtime semantics

The central outcome is a reproducible engineering method, not just a single binary artifact. That method is what made repeated recovery and eventual success possible.

Appendix A — Core reverse-engineering commands (static)

```bash
python3 <TOOLS_DIR>/xli_firmware_autopsy.py \
  --ftp-dir <FTP_DIR> \
  --mem-dir <LIVE_DUMP_DIR> \
  --out-json <TOOLS_DIR>/xli_firmware_autopsy.json \
  --out-md <TOOLS_DIR>/xli_firmware_autopsy.md \
  --out-zlib-dir <TOOLS_DIR>

python3 <TOOLS_DIR>/xli_firmware_tool_inventory.py \
  --app-dec <TOOLS_DIR>/192-8001V1_106.bin.zlib_payload.bin \
  --boot-dec <TOOLS_DIR>/192-8000V1_106.bt.zlib_payload.bin \
  --out-json <TOOLS_DIR>/xli_firmware_tool_inventory.json \
  --out-csv <TOOLS_DIR>/xli_firmware_tool_inventory.csv \
  --out-md <TOOLS_DIR>/xli_firmware_tool_inventory.md

python3 <TOOLS_DIR>/xli_firmware_deep_mine.py \
  --app-dec <TOOLS_DIR>/192-8001V1_106.bin.zlib_payload.bin \
  --boot-dec <TOOLS_DIR>/192-8000V1_106.bt.zlib_payload.bin \
  --out-json <TOOLS_DIR>/xli_firmware_deep_mine.json \
  --out-md <TOOLS_DIR>/xli_firmware_deep_mine.md

python3 <TOOLS_DIR>/xli_firmware_runtime_compare.py \
  --app-bin <FTP_DIR>/192-8001V1_106.bin \
  --boot-bin <FTP_DIR>/192-8000V1_106.bt \
  --mem-dir <LIVE_DUMP_DIR> \
  --out-json <TOOLS_DIR>/xli_fw_runtime_compare.json \
  --out-md <TOOLS_DIR>/xli_fw_runtime_compare.md

binwalk -E -M <FTP_DIR>/192-8001V1_106.bin > <TOOLS_DIR>/binwalk_192-8001V1_106_E_M.txt
binwalk -E -M <FTP_DIR>/192-8000V1_106.bt  > <TOOLS_DIR>/binwalk_192-8000V1_106_E_M.txt
binwalk -E -M <FTP_DIR>/184-8000V65.bin    > <TOOLS_DIR>/binwalk_184-8000V65_E_M.txt
```

Appendix B — Recursive extraction and carve commands

```bash
mkdir -p <TOOLS_DIR>/binwalk_me_test
cd <TOOLS_DIR>/binwalk_me_test
cp <FTP_DIR>/192-8001V1_106.bin ./input.bin
binwalk -Me --run-as=root input.bin > run.log 2>&1

strings -a -n 4 <TOOLS_DIR>/192-8001V1_106.bin.zlib_payload.bin | \
  rg -i 'SCAN_FOR_OPT_CARD|INSTALL_SMART_OPTIONS|OPTION BAY|BoardAdmin|IRIG|1PPS|FTM|N8|N1'

strings -a -n 4 <TOOLS_DIR>/192-8001V1_106.bin.zlib_payload.bin | \
  rg -o '[A-Za-z0-9_./-]+\.(html|shtml|gif|jpg|jpeg|css|js)' | sort -u

strings -a -n 4 <TOOLS_DIR>/192-8001V1_106.bin.zlib_payload.bin | \
  rg '^F[0-9]{3}\b' | sort -u
```

Appendix C — Live read-only probe commands

```bash
# flash map and base checks
showFlashSizes()
SizeofFlash()
get_flash_memptr()

# mapped flash reads
# boot / fpga / app / fs / nvram
# (sample)
d 0x10000000,32,1
d 0x100a0000,32,1
d 0x10100000,32,1
d 0x105e0200,32,1
d 0x107e0000,32,1

# symbol probes
lkup "scan_for_opt_card"
lkup "install_smart_options"
lkup "show_opt_table"
lkup "GPSBp"
lkup "SA_M12"
lkup "gpsTimeToUnixEpoch"
lkup "gpsTimeNormalize"
lkup "leapSecGpsWeekDayToGpsDate"

# GPS direct interrogation
SA_M12_Sw_Id_Version(0x30006000,sw)
SA_M12_FPGA_Version(0x30006000,fv)
SA_M12_GetRcvrIDdata(0x30006000,id1,id2,id3)
GPSBpStatus(0x30006000)
GPSBpLocked(0x30006000)
```

Appendix D — Burn/deploy/recovery command set

```bash
# stage files (include colon-prefixed variant)
cp -f <PATCHED_IMAGE> <FTP_DIR>/192-8001.bin
cp -f <PATCHED_IMAGE> <FTP_DIR>/:192-8001.bin
```

NI burn commands
software

F100 BH:<FTP_HOST>,192-8001.bin
F100 bu

filesystem rollback

F100 BH:<FTP_HOST>,192-8002V1_106.fs
F100 bf

reboot

F100 BASET AUTO
```

### Appendix E — Verified canonical branch-stub parameters

- Patch site live address: `0x000b6d78`
- Patched branch word: `0xea031b28`
- Stub live address: `0x0017da20`
- Return address from stub: `0x000b6d80`
- Embedded offset constant: `0x24EA0000`
- Expected patch verifier fields:
  - `site_is_branch=true`
  - `site_branch_target=0x0017da20`
  - `stub_pattern_ok=true`

### Appendix F — Chronological attempt ledger (persistent memory)

This appendix is derived from durable session memory and preserves the technical progression, including failures and near-misses.

- `2026-02-10T02:02:27+00:00` | `success` | `b70c8413-137f-42a6-8933-9e3b61d82033`
  - summary: Observed success signal: front panel year now shows 2026 after RAM patch sequence.
  - lesson: RAM patch path confirms root-cause path; next step is persistent firmware patch for reboot survival.
- `2026-02-10T02:02:27+00:00` | `failure` | `dbe80f45-3ebd-46ab-a27b-2e12b4e5aa78`
  - summary: Flashburn trigger path toggles doNvRamFlashBurn but does not emit expected FTP fetch when invoked via CLI symbols.
  - lesson: Need exact call signature/selector mapping for ProcFlashFile*; manual ftp copy path works and confirms server reachability.
- `2026-02-10T02:10:10+00:00` | `near_miss` | `6fd20685-e9db-407f-bde7-a2d70095bc74`
  - summary: Device became unreachable after VxWorks shell bootline probing during telnet session.
  - lesson: Do not call potentially blocking bootline/NVRAM boot functions from interactive tShell without offline signature verification and rollback path.
- `2026-02-10T02:13:29+00:00` | `success` | `94c52eea-78fd-42d0-adfc-053283b96280`
  - summary: VxWorks FTP/netDrv behavior on XLi: client prepends ':' to requested filenames and uses active-mode PORT; anonymous login works against local vsftpd root <PATH>
  - lesson: Keep both normal and colon-prefixed filenames in FTP root for fetches (example: 192-8002.fs and :192-8002.fs).
- `2026-02-10T02:13:30+00:00` | `success` | `3d00ee41-d60b-4d4d-86c0-48d55bb5c737`
  - summary: Known-good telnet access sequence: connect <TARGET_IP>:23, send username immediately, then password at prompt; max-session exhaustion returns explicit refusal message.
  - lesson: When login fails with concurrent-session message, clear stale sessions before retrying; keep one interactive shell.
- `2026-02-10T02:13:30+00:00` | `success` | `1cb16d06-962b-4843-a4e0-2f7c0c5c4732`
  - summary: Filesystem image 192-8002V1_106.fs verified as placeholder-only content: root shows config/etc/web directories with no patchable files.
  - lesson: Use SHA-256 f67bb2f1d99431697681c7104b26be15ec0597f197d44f2cfcd76cd2de9ec3e6 to confirm known empty image; do not claim HTML patch success from this fs.
- `2026-02-10T02:13:30+00:00` | `success` | `9ceb9e71-8bc9-4cc6-98e8-db939bd128f3`
  - summary: Working RAM rollover fix path: patch GetTimeCOPS_TOR flow by branching at 0x000b6d78 to stub that adds +0x24EA0000 seconds before localtime conversion; restore year-only instruction at 0x000b6f88 to
```

original +0x0c when full fix is active.
  - lesson: Year-only patch can produce wrong calendar date; full +1024-week seconds-offset fix corrects year/month/day but is RAM-only until persistence method is implemented.
- `2026-02-10T02:13:30+00:00` | `failure` | `7f80d08d-2bb7-42fb-9024-604afe8cf321`
  - summary: ProcFlashFile* invocation is signature-sensitive; one attempted call produced FTP RETR <TARGET_IP> (host string treated as filename), confirming argument mismatch despite apparent function success returns.
  - lesson: Do not trust return code alone; validate flashburn with packet capture and explicit RETR of expected filename.
- `2026-02-10T02:13:30+00:00` | `near_miss` | `1067f562-2a74-4195-ba7a-83d15b350005`
  - summary: Interactive bootline/NVRAM probing in tShell can destabilize access; a bootline probe path was followed by host-down state and required recovery.
  - lesson: Block getNvRamBootLine()/unknown bootline calls in live shell unless user gives explicit risk acceptance and recovery path is in place.
- `2026-02-10T03:33:44+00:00` | `failure` | `30dc8f8f-e3aa-4c15-9e97-d9d17bdc0ba6`
  - summary: After reboot, startup script path remained configured but patch did not auto-apply; code at 0x000b6d70 reverted to stock and date came up 1970/2006.
  - lesson: This firmware path does not execute startup script in the expected shell context at boot; cannot rely on bootChange s= for automatic RAM patching.
- `2026-02-10T03:33:44+00:00` | `success` | `18ae41f4-5463-4034-8f07-7bc4971b0163`
  - summary: Removed local launchd watchdog; moved all patch artifacts onto XLi under /flash0/config and configured bootChange startup script field to /flash0/config/gpspatch.start.
  - lesson: On-device storage and boot param write are working; use long wait on TTFlash writes to avoid zero-length files.
- `2026-02-10T03:33:44+00:00` | `success` | `899bf5e9-5070-4c76-ad85-ed689632622b`
  - summary: Recovered from bad interim patch states (1970/1989) by restoring known-good branch+stub patch: stub at 0x00d5ac98 and branch block at 0x000b6d70 with 0xea328fc6.
  - lesson: Keep known-good restore block ready; always verify with XLiUserHome_time_date and d 0x0031a40c after any patch attempt.
- `2026-02-10T03:50:38+00:00` | `success` | `7bbe706f-8c03-42aa-b98b-1353b10d5555`
  - summary: Enabled runtime WDB exposure from F100 shell using wdbConfig(); confirmed tWdbTask active and UDP/17185 reachable from LAN probe.
  - lesson: Use wdbConfig() as low-impact runtime debug entry; verify via task list before using external debugger tooling.
- `2026-02-10T03:50:38+00:00` | `success` | `cd1826f5-6195-4c78-af79-741930b1b030`
  - summary: Reinitialized FTP daemon after adding users; inbound FTP accepts sessions and supports absolute-path RETR/STOR to on-box filesystem.
  - lesson: If LIST fails, test RETR/STOR with absolute paths directly; behavior differs per directory.
- `2026-02-10T03:50:38+00:00` | `near_miss` | `72243412-2752-48e9-bc45-54a9a7f7f3a5`
  - summary: Interactive getstartup() call caused Data abort and shell restart.
  - lesson: Avoid unknown startup helper function calls in interactive shell; stick to lkup, d, i and explicit known-safe routines.
- `2026-02-10T03:52:40+00:00` | `success` | `a8908e35-90fa-4efe-8581-670eb80b1811`
  - summary: Verified inbound FTP can write to /flash0/config and read back the written bytes; confirms direct remote modification path to persistent storage.
  - lesson: Use absolute /flash0 paths for deterministic writes; validate with immediate RETR readback.
- `2026-02-10T03:52:40+00:00` | `near_miss` | `4ac9d3ad-653c-462f-aa6a-a2ed1cf85dc0`
  - summary: FTP DELETE and QUIT can time out on this target even when operation likely succeeded.
  - lesson: After timeout, verify state with fresh session and RETR/ls before retrying destructive ops.
- `2026-02-10T04:00:54+00:00` | `failure` | `0c75b565-a6aa-4737-b674-0582d148f70e`
  - summary: Verified no built-in SSH daemon capability in current XLi firmware: lkup for ssh/sshd/dropbear/scp/sftp returned no symbols.
  - lesson: Do not spend cycles trying to flip an SSH enable flag; requires custom daemon/firmware integration.
- `2026-02-10T04:00:54+00:00` | `success` | `2f8cb874-d692-483c-8616-9ec2a017ef65`
  - summary: Confirmed front Network Interface telnet login is separate from VxWorks shell login DB; loginUserAdd does not bypass NI gate.
  - lesson: Treat NI login and shell login as different controls when planning exposure/persistence changes.
- `2026-02-10T04:00:54+00:00` | `success` | `fbe4bcd0-464e-49ac-ac9b-b303c0bf49fb`
  - summary: WDB task remains active; repeat wdbConfig() returns bind failed when already running. UDP/17185 remains reachable from LAN.
  - lesson: Use tWdbTask/port probe as truth; bind failed on wdbConfig can indicate already active service.
- `2026-02-10T04:05:25+00:00` | `success` | `e35224db-7acd-4954-b3ef-68ab7495b549`
  - summary: Preparing persistent exposure change via boot flags + startup script

(/flash0/config/expose.start) to force WDB and FTP at boot.
  - lesson: Capture current bootline and verify post-write before reboot.
- `2026-02-10T04:11:19+00:00` | `failure` | `d018982e-388e-47d9-b939-fe2b8c591aee`
  - summary: Post-reboot check showed date reverted to 01/01/1970 and gps patch stub at 0x00d5ac98 was erased (0xEE fill).
  - lesson: Current gps patch path is RAM-only; requires true firmware binary patch/flash for persistence.
- `2026-02-10T04:11:19+00:00` | `failure` | `601e5576-dc8e-4456-8965-9818042b5eb3`
  - summary: Bootline modifications done through bootChange did not survive reboot; sysBootLine returned to default without f=/s= fields.
  - lesson: Do not rely on bootChange for persistent startup hooks on this unit; persistence likely sourced elsewhere or overwritten at boot.
- `2026-02-10T04:11:19+00:00` | `near_miss` | `f04a15c7-6419-402c-9b74-d4cd1dce1d1b`
  - summary: After reboot, UDP/17185 remained reachable but FTP auth reopened to 530 failures for all tested creds.
  - lesson: Service exposure state is partially reset on reboot; need firmware-level init patch for deterministic WDB/FTP behavior.
- `2026-02-10T04:25:43+00:00` | `success` | `bbc64d88-9f24-4de4-bdcb-16649e0baea8`
  - summary: Enumerated flash primitives and burn pipeline symbols (flash_write*, erase, TTFlash, ProcFlashFile, XLiFlashburnAdmin_*).
  - lesson: Classify flash_write/erase/TTFlash/ProcFlashFile/XLiFlashburnAdmin_Execute as destructive; require backup and rollback before use.
- `2026-02-10T04:25:43+00:00` | `success` | `04e0353c-48f4-4bc2-83db-e841ad60603e`
  - summary: Mapped full flash layout via showFlashSizes and confirmed direct memory-mapped access at 0x10000000..0x107FFFFF.
  - lesson: Use mapped flash reads for partition verification before any write attempt; addresses are authoritative from showFlashSizes().
- `2026-02-10T04:25:43+00:00` | `near_miss` | `3190bab6-a99e-46da-97f0-f9c00518e638`
  - summary: Direct nvAmdFlashRead call with unknown signature destabilized shell/session.
  - lesson: Do not call unknown-signature flash APIs interactively; prefer validated wrappers and read-only probes.
- `2026-02-10T04:26:18+00:00` | `success` | `84fb4383-c207-4b2f-bb85-4701a5b160ce`
  - summary: XLi flash map captured from showFlashSizes(): bootloader 0x10000000 size 0x000A0000; FPGA 0x100A0000 size 0x00060000; application 0x10100000 size 0x004E0200; filesystem 0x105E0200 size 0x00200000; NVRAM bank B 0x107E0000 size 0x00010000; NVRAM bank A 0x107F0000 size 0x00010000.
  - lesson: Treat these partition boundaries as authoritative for backup and patch planning; never write outside partition bounds.
- `2026-02-10T04:26:18+00:00` | `success` | `e32deb70-2996-4d1c-b1be-55151ab49c33`
  - summary: Verified flash is memory-mapped: get_flash_memptr()=0x10000000, SizeofFlash()=0x800000; direct reads via d succeeded at 0x10000000 (boot), 0x100A0000 (FPGA), 0x10100000 (app), 0x105E0200 (VXEXT FS boot sector), 0x107E0000 (NVRAM).
  - lesson: Use read-only mapped-address dumps first for verification and backups before any erase/write operation.
- `2026-02-10T04:26:18+00:00` | `success` | `963afbf6-b1b9-46b9-8655-588e8cfa7941`
  - summary: Flash primitive symbols present: flash_write, flash_write_word, flash_write_string, flash_sector_erase, flash_chip_erase, flash_read_word, flash_get_status/device_id/manuf_code/sector_size/numsectors, flash_sector_protect_verify, flash_erase_resume, flash_reset_ub, nvAmdFlashRead. Higher-level burn path symbols: TTFlashInit/TTFlash, ProcFlashFile/ProcFlashFileWithStringCheck, RestoreFlashImage, Crc32FlashImage, nvRamFlashBurnTask, and XLiFlashburnAdmin_* handlers.
  - lesson: Classify flash_write*/erase*/TTFlash/ProcFlashFile/XLiFlashburnAdmin_Execute as destructive and require backup + rollback plan before use.
- `2026-02-10T04:26:18+00:00` | `success` | `94c2bada-1ed0-44de-9e65-d841e165c928`
  - summary: Burn-related globals observed: doNvRamFlashBurn @0x0027e754 = 0, burnFlag @0x0027f38c = 0, burnName @0x0027f390 = 'bootrom_uncmp.bin', burnIP @0x002ecaac zeroed. GetFlashStatus() returned 0. flash_get_device_id()/flash_get_manuf_code() returned 1 on this unit.
  - lesson: Use global-state checks before invoking any burn path to confirm expected idle state and selected target image name.
- `2026-02-10T04:26:18+00:00` | `near_miss` | `4699c91c-2297-4022-93cd-1692596bd3e9`
  - summary: Near-miss: direct interactive call to nvAmdFlashRead with unknown signature caused shell/session instability and temporary service loss; unit later recovered.
  - lesson: Never invoke unknown-signature flash APIs interactively; prefer known wrappers, read-only probes, and controlled recovery path.
- `2026-02-10T04:53:49+00:00` | `success` | `08e5cd18-64bb-4f23-9ed8-4fed4677ab0e`
  - summary: Completed full autopsy of 192-8001V1_106.bin and 192-8000V1_106.bt: both have primary zlib

payload at offset 0xA801; decompressed app len 2809624 and boot len 1178200. Runtime GPS patch probe signatures map into decompressed app payload with stable rule runtime_addr = dec_offset + 0x1000.
  - lesson: Persistent patch prep must target decompressed app payload coordinates (runtime-0x1000), not raw .bin offsets.
- `2026-02-10T04:53:49+00:00` | `success` | `f067cb76-e5de-4e0c-a892-935e1d922332`
  - summary: Autopsy confirmed 184-8000V65.bin is bit-swapped Xilinx bitstream (sync at 0x4, FDRI 24374 words) and 192-8002V1_106.fs is VXEXT1.0 image with root dirs config/etc/web only (cluster 33, size 0) and no patchable web files.
  - lesson: Do not spend time patching web UI via 192-8002V1_106.fs; use runtime web extraction and application firmware path for GPS logic changes.
- `2026-02-10T04:53:49+00:00` | `success` | `e6caa6e5-2d94-429e-8ff0-2fcde79e7d6c`
  - summary: Created repeatable autopsy tool <PATH> producing <PATH> and .md plus decompressed payload artifacts.
  - lesson: Run the autopsy script first in new sessions to re-establish known-good firmware facts before live patching.
- `2026-02-10T05:04:08+00:00` | `success` | `68760542-ee3c-4985-9826-33c74904f150`
  - summary: Extracted deep tool inventory from decompressed app/boot payloads using xli_firmware_tool_inventory.py. App payload contains GPSBp*, gpsTime*, ProcFlashFile/TTFlash, flash_write/erase, wdbConfig, ftpdInit/Delete, tftpGet/Put, loginUserAdd/Delete, bootChange, sysBootLine, /tyCo/*, /flash0, /ramdrv markers.
  - lesson: Use inventory artifacts to select correct subsystem and avoid blind probing. Key files: xli_firmware_tool_inventory.json/.md/.csv.
- `2026-02-10T05:04:08+00:00` | `success` | `5ae8cbc2-7498-42e0-9efd-063d306a5612`
  - summary: Searched decompressed app/boot payloads for embedded GPS firmware artifacts (Intel HEX, Motorola S-record, Furuno/GT-8031/GT-8036/GT-8736, HCS12, MSP430 markers). No standalone GPS firmware blob identified in local firmware files.
  - lesson: Direct GPS manipulation path currently available is host-side transport/control (GPSBp* and /tyCo/*), not an in-bundle GPS module firmware image.
- `2026-02-10T05:04:08+00:00` | `success` | `9adde1d6-b153-41f7-b6c3-e3e7dfe459b2`
  - summary: Confirmed decompressed payload mapping rule remains stable: runtime_addr = app_dec_offset + 0x1000 across GPS patch probe points; both app and boot bundles carry primary zlib stream at raw offset 0xA801.
  - lesson: Persistent patch work must operate on decompressed app payload offsets (runtime-0x1000), then repack; raw .bin offsets are misleading.
- `2026-02-10T05:10:00+00:00` | `success` | `703a9be1-6797-4d81-b009-85ef400519f2`
  - summary: Completed deep firmware mine on 192-8001 app and 192-8000 boot decompressed payloads; extracted deterministic startup/bootline/exposure/gps marker map with runtime translation rule runtime=dec+0x1000; updated RUNBOOK section 45 with read-only validation sequence.
  - lesson: Use symbol/address map first and validate with lkup before any live patch; avoid guessed addresses.
- `2026-02-10T05:10:00+00:00` | `failure` | `a10f2671-d4a6-4db9-a3dc-4ad6b7e2124e`
  - summary: persistent-memory CLI path on this machine is <MEMORY_TOOL>; /root/.analysis-tooling path failed.
  - lesson: Always resolve persistent-memory script path from installed skill directory before running memory/env commands.
- `2026-02-10T05:10:15+00:00` | `success` | `b2c10392-7c7d-47a4-a3ec-02861a5b5773`
  - summary: Deep mine confirms extensive option-bay/databus symbol surface in app payload (BackPlSetup, slot/card selectors, XLi*BoardAdmin handlers, GPSBp* path). Card control appears host-mediated via backplane APIs and web dispatchers rather than discrete standalone card firmware blobs inside app/boot payloads.
  - lesson: For bay/card debugging, pivot first to GPSBp*/slot/card symbols and dispatcher handlers; do not assume separate recoverable card firmware blob in host images.
- `2026-02-10T05:27:42+00:00` | `success` | `917dbf7a-0bd6-45ef-a37d-5a8684d5c09f`
  - summary: Completed full live flash backup using chunked RAMDRV->FTP extraction. Reassembled bootloader/app/filesystem images plus captured FPGA and both NVRAM banks. Direct large RAMDRV dumps were unreliable due RAM limits; chunk size 0x20000 worked.
  - lesson: Use chunked extraction for large regions; verify file counts/sizes and sha256 after host-side reassembly before risky changes.
- `2026-02-10T05:27:42+00:00` | `near_miss` | `c4e1633a-b557-40cf-b764-9dfb4ed2df07`
  - summary: Unit became unreachable during bootline probe sequence involving getNvRamBootLine() in F100 SHELL. Required reboot to recover LAN access.
  - lesson: Avoid direct bootline/NVRAM function probing in live shell unless absolutely necessary; keep local reboot access and backup in place.
- `2026-02-10T05:28:19+00:00` | `success` | `b1610b7e-74dd-458c-b404-66873a0d3235`
  - summary: Live symbol discovery confirmed bootline/bootparam and startup-related functions are present

(getNvRamBootLine, bootChange, bootParamsShow/Prompt, build_boot_line, setNvRamDwLd, nvRamFlashBurnTask, getstartup, wdbConfig, ftpdInit). setNvRamBootLine/setstartup are not exported via lkup.
  - lesson: Use exported boot parameter paths first; avoid raw unexported setter assumptions and avoid direct bootline pointer probes.
- `2026-02-10T06:00:40+00:00` | `near_miss` | `7c27e2aa-504e-4bf3-b874-26897892f1a0`
  - summary: Persistent exposure attempt changed boot params to startup script /flash0/expose.start and flags read back as 0x50 (entered 0x80). After delayedReboot unit became unstable/unreachable on LAN; WDB 17185 did not come up stably.
  - lesson: Rollback first: set flags back to 0x0 and clear startup script via bootChange before further experiments. Avoid unknown boot-flag permutations without out-of-band recovery.
- `2026-02-10T06:04:49+00:00` | `success` | `1553e943-b994-4741-a6df-72fb798bfb7c`
  - summary: Post-power-cycle verification before rollback: bootChange showed flags=0x0 and startup script blank. Manual wdbConfig() succeeded and task list showed tWdbTask, but no evidence of persistent auto-WDB startup yet.
  - lesson: Use this as clean baseline: safe boot params + manual WDB path confirmed.
- `2026-02-10T06:14:54+00:00` | `success` | `bf99c117-df37-49f6-beb2-5d5dffb37821`
  - summary: Completed full local XLi firmware teardown pass across app/boot/fpga/fs with decompression, runtime mapping, deep mine, tool inventory, binwalk entropy scans, and artifact hashing; consolidated in RUNBOOK section 51.
  - lesson: Use <MEMORY_TOOL> path. Stable runtime-to-decompressed-app delta is 0x1000. 192-8002V1_106.fs remains placeholder (config/etc/web only). Live web handlers exist in decompressed app payload.
- `2026-02-10T06:25:45+00:00` | `success` | `6792bf28-a8a7-4a37-9dec-f9762b25ab5e`
  - summary: Checked M12M online command/control capabilities over serial bus. Documented config/query commands and data injection paths (almanac/ephemeris), but no documented firmware dump/readback command in command list.
  - lesson: Bus supports control and navigation data maintenance, not proven firmware extraction. Use GPSBpSendMsg/GPSBpGetMsg for command bridge and focus on date/time hint and mode controls.
- `2026-02-10T06:30:13+00:00` | `success` | `99e2fb8a-90de-495d-8208-1f88dfce799f`
  - summary: Non-official source sweep found GT-8031 proprietary PFEC command list and operational hacks (boot-time date register set/MITM), but no credible public hidden command for full GPS firmware dump/readback.
  - lesson: Leaked GT-8031 protocol indicates config/status/almanac upload-download via PFEC sentences; no firmware extraction command surfaced. Most reported permanent fixes patch host firmware/HCS12 or replace module.
- `2026-02-10T06:36:23+00:00` | `success` | `09b39a30-504e-45e4-8263-d841b6bba84b`
  - summary: Live shell confirmed dumpGpsEmib and GPSBp/SA_M12 functions are exported/callable. dumpGpsEmib returns telemetry fields only, not firmware bytes.
  - lesson: Use lkup dumpGps/GPSBp/SA_M12 to verify surfaces quickly; treat dumpGpsEmib as status dump only. No direct module firmware dump symbol found yet.
- `2026-02-10T06:47:26+00:00` | `success` | `ed39f257-dd4f-429f-9c2f-40c190b817ba`
  - summary: Completed kitchen-sink firmware expansion across 265 artifacts with hash/signature/entropy/marker/correlation coverage; runbook section 54 added with outputs and caveats.
  - lesson: Earlier VXEXT parser conclusion ('empty-only fs') was incomplete: signature+string scans show rich embedded web/UI content in vendor and backup FS images. Treat current fs parser as partial; use alternative extraction/index reconstruction.
- `2026-02-10T06:53:41+00:00` | `success` | `4f33c599-5b81-49ca-9d99-53dff16a3496`
  - summary: Ran recursive binwalk extraction across vendor and backup images with --run-as=root and expanded analysis of extracted A801 payloads plus FS signature carving.
  - lesson: A801 payloads are identical between vendor and backup for app/boot; A801.zlib length differences come from trailing bytes. FS/flash images need signature-based/manual carving for deep extraction.
- `2026-02-10T06:57:16+00:00` | `success` | `b09737e5-e093-4eff-bcb6-39a64e9bebd0`
  - summary: Post-extraction full-surface pass completed: option bay/card scan strings, web asset inventory in app payload, F100 marker set in app+boot, runbook section 56 added with repeatable commands.
  - lesson: Use app zlib payload as primary source of option/web/control surfaces; do not treat empty FS parser result as final. Always start from runbook sections 52, 54.6, 55, 56.
- `2026-02-10T07:03:27+00:00` | `success` | `0e786259-033a-4257-88f8-ec34e0eaea5a`
  - summary: Built consolidated option-bay and GPS callpath mapper and generated address-level patch anchor report; updated RUNBOOK section 57 with exact symbols/offsets/checklist.
  - lesson: Use xli_reverse_callpath_map.py outputs as the primary map for bay/gps debugging. Runtime address rule remains dec+0x1000. Part-number evidence in payloads currently exposes 87-8043 only; do not assume card part numbers are all embedded as plain strings.
- `2026-02-10T07:05:17+00:00` | `near_miss` | `e54342ab-eb92-4a75-81fe-7ad99a0de171`
  - summary: Corrected callpath addressing semantics: deep-mine offsets are marker/string anchors, not executable entry points; switched report to marker+live-lkup model.

  - lesson: Do not derive executable patch addresses from marker string offsets. Always confirm function addresses using live lkup or disassembly at verified code regions.
- `2026-02-10T07:45:15+00:00` | `success` | `0e94d981-426c-4197-87d4-83f71c1def76`
  - summary: Recovered and documented exact payload/live boot patchpoints for FUN_00070d9c mode path, including direct SA_M12_Set_Get_GPS_Mode callsite and hook slot candidate; appended to RUNBOOK section 60 with artifacts.
  - lesson: For this handler family use live=payload+0x1000; patchpoint set is stable at 0x70df8/0x70dfc/0x70e04/0x70e78/0x70e7c and should be validated with code-cave register-preserving stub before any runtime flash write.
- `2026-02-10T07:55:44+00:00` | `success` | `a2ea715e-98a1-4dd4-a72c-72043a5c04a2`
  - summary: Confirmed direct GPS interrogation through SA_M12 wrappers (version strings read), but no explicit GPS firmware flash/update wrapper found in SA_M12/GPSBp + FUN_000490c8 command surface; host patch remains primary persistence path.
  - lesson: Use SA_M12_Sw_Id_Version/FPGA_Version/GetRcvrIDdata on 0x30006000 for non-destructive proof of direct path; treat GPS firmware reflashing as unproven until a concrete erase/program command or module debug interface is identified.
- `2026-02-10T08:19:32+00:00` | `near_miss` | `9ced786d-1030-4482-9901-0426f3020829`
  - summary: Built patched 192-8001 software image (payload offsets 0x588ac/0x588b0) and invoked burn via burnHost/burnFile with corrected signature; sector write progress observed, then NI command channel became non-responsive pending reboot verification.
  - lesson: burnFile signature is burnFile(filename, output, type) with host set separately by burnHost; dry-run with NOSUCH.bin is safe for signature verification. After load burn, perform power-cycle before judging command-plane health and persistence.
- `2026-02-10T09:08:40+00:00` | `near_miss` | `067eb429-7b71-499e-82ec-ae58784b80ea`
  - summary: Patched 192-8001 software image causes web login failure (logon always loops to XLi Login Page) while NI/telnet remains functional.
  - lesson: Do not deploy current GPS rollover software patch to production image; validate full HTTP login path after each burn, not only telnet/shell availability.
- `2026-02-10T09:08:40+00:00` | `success` | `fd483242-693d-4c14-92e8-4788e170b250`
  - summary: Recovered unit from web/app auth failure by restoring stock 192-8001V1_106.bin (and 192-8002V1_106.fs when needed), then rebooting.
  - lesson: Canonical recovery sequence: BH+bu stock software, BH+bf stock filesystem, BASET AUTO reboot, verify 192-8001 banner + F100 SHELL + TCP/80.
- `2026-02-10T11:16:55+00:00` | `success` | `c5772f0e-2fbc-4f57-a8d4-bc12361e4c19`
  - summary: Delivered fresh UTC @@Gb hint (17-byte payload) to XLi via both /tyCo/1 write and GPSBpSendMsg after FTP pull of :hint.bin; transport path confirmed end-to-end.
  - lesson: Transport path works: FTP RETR :hint.bin, copy to /ramdrv, serial write returns 17, GPSBpSendMsg returns 0/1. This proves command delivery but not rollover correction.
- `2026-02-10T11:16:55+00:00` | `failure` | `8e5b8956-8d68-41de-835d-97362a51d9c8`
  - summary: After single and repeated (~30 cycles) direct hint injections, XLi date buffer remained 06/27/2006 and GPSBpStatus/GPSBpLocked stayed 0.
  - lesson: On this firmware/module state, runtime hint injection alone does not fix WNRO. Required next path is persistent host firmware patch in GPS date conversion path (or module replacement/firmware-level module fix).
- `2026-02-10T15:06:21+00:00` | `near_miss` | `1f17642c-4146-44bc-9dfc-97305d169c3d`
  - summary: Executed requested sequence: reboot, then pre-lock @@Cf + repeated @@Gb injector over /tyCo/1 and GPSBp paths. Initial injector blocked by telnet session-cap; second controlled attempt led to host becoming unreachable.
  - lesson: Telnet session cap on NI can block early-boot automation. If automation churn occurs, unit may drop off network and require manual power-cycle. Use single-session injector only after prompt stability check.
- `2026-02-10T15:12:55+00:00` | `success` | `1406128e-8ce9-4ef7-9853-130e987c550b`
  - summary: Selected persistence strategy: firmware-level one-shot startup hook on XLi to generate/send GPS hint internally at boot and self-terminate; avoid bootChange s= due unreliability.
  - lesson: For this unit, boot script persistence is unreliable; implement internal startup call path using existing GPSBpSendMsg/GPSBpLocked symbols and bounded retry loop.
- `2026-02-10T22:14:27+00:00` | `success` | `af7b7d43-d98a-4515-b375-cb8faaebbfe6`
  - summary: Hard rule adopted: only write confirmed-used partitions/sectors; never write unknown/reserved sectors. Modified app images hit sector31/CRC ERR4 gate and were rejected.
  - lesson: Treat flash integrity gate as authoritative. If sector-program/CRC verification fails, stop and restore stock image; do not force writes into protected regions.
- `2026-02-10T22:14:27+00:00` | `success` | `6859f6d8-f899-49c2-91dd-1f2f0c1f3a8a`
  - summary: Recovered unit from bootloader mode by burning stock 192-8001V1_106.bin, then rebooting to NI 192-8001 with shell available.

  - lesson: Stock app burn is reliable recovery baseline when modified image integrity checks fail.
- `2026-02-10T22:16:08+00:00` | `success` | `5919a9af-5be1-4c82-b0c8-2ca045ce2554`
  - summary: Active rule: do not bypass write protection unless sector usage/free partition space is proven safe; do not destroy OS.
  - lesson: When sector ownership is uncertain, treat as protected and do not write. Only write to confirmed unused or explicitly free partition space.
- `2026-02-10T22:23:47+00:00` | `success` | `e9eca08b-2678-4548-ac1c-fb9cb09279d7`
  - summary: Unofficial-doc sweep captured VxWorks startup/loader/WDB quirks and XLi operational constraints; runbook section 68 added with source links and safe persistence strategy.
  - lesson: Preflight before planning: read last 500 lines of RUNBOOK and latest persistent-memory memory export; avoid relying solely on bootline s= startup script for persistence; keep <=2 telnet sessions; preserve stock burn recovery path.
- `2026-02-10T22:31:06+00:00` | `success` | `b2a42546-3c5f-418c-acb5-75535882c29e`
  - summary: Live probe: date buffer shows 02/10/2026 while GPSBpStatus=0 and GPSBpLocked=0; unit currently not GPS-locked, but runtime patch/date path active.
  - lesson: 1970 can be transient before lock. 2006 indicates rollover bug path; with patch, date may show corrected year even when lock remains 0. Validate lock separately from displayed date.
- `2026-02-10T22:34:15+00:00` | `success` | `337d6086-c802-472b-b4be-9a788a60762b`
  - summary: Reapplied runtime rollover patch successfully: patcher reported patched_date=02/10/2026 with valid branch/stub. Raw date buffer bytes at 0x0031a40c decode to 02/10/2026.
  - lesson: Telnet may close transiently under session churn; retry in single controlled session. Parser can miss date due control chars, so trust raw buffer bytes when needed.
- `2026-02-10T22:45:17+00:00` | `success` | `af7caf99-2787-40a2-be8e-6235bf208d63`
  - summary: Reapplied runtime GPS rollover patch after reboot regression to 2006; patch tool returned patched_date=02/10/2026 and date buffer bytes decode to 02/10/2026.
  - lesson: Patch is still RAM-only and will revert on reboot. GPS lock/status currently report 0 while date conversion path is corrected.
- `2026-02-10T22:53:13+00:00` | `near_miss` | `8da5d124-2a4a-4d1f-b927-5b63644096ed`
  - summary: Persistent burn attempt blocked: NI returns 'Maximum number of concurrent telnet sessions exceeded' for all connection attempts; no login slot available to execute F100 bu.
  - lesson: When session-cap is stuck, clear sessions via power-cycle/front-panel reboot before burn workflow. Keep exactly one automation session afterward.
- `2026-02-10T23:11:59+00:00` | `failure` | `5da8c959-9a82-449e-8acc-1d62b1b214e2`
  - summary: Tried full rebuilt 192-8001.bin with corrected trailer CRC32; burn still fails at sector 31 with same readback mismatch and ERR4 on all three attempts.
  - lesson: Sector31 gate is not only file trailer CRC. Verifier enforces deeper integrity/auth data (or per-sector scheme) tied to modified image bytes.
- `2026-02-10T23:25:54+00:00` | `failure` | `f671c723-cd91-4db7-936c-6bfcc8c74334`
  - summary: After reboot, date regressed to 06/27/2006 and runtime patch markers were absent (site 0x000b6d78 = 0xe51b3040, no branch/stub).
  - lesson: Current GPS rollover fix is still RAM-only; always verify site_branch/stub state after reboot before assuming persistence.
- `2026-02-10T23:25:54+00:00` | `success` | `58686ad4-289b-4f8a-9793-4e5effc76ffe`
  - summary: Single-session runtime recovery succeeded with xli_patch_rollover.py; patched_date=02/10/2026 and branch_target=0x00d71eac stub_ok=1.
  - lesson: Fast recovery command is stable: run xli_patch_rollover.py then verify d 0x0031a40c,16,1. Keep one telnet session only.
- `2026-02-10T23:43:03+00:00` | `failure` | `76fa476c-1307-4614-b484-288912f0ca65`
  - summary: Persistent inline1024 firmware patch booted but produced date 02/02/1970 on cold boot (site word 0xe50b2040 at 0x000b6d78, no branch stub).
  - lesson: Inline replacement at 0x000b6d70 is not semantically equivalent to working runtime branch+stub path; do not reuse for persistence.
- `2026-02-10T23:43:03+00:00` | `success` | `1e62cafa-e2f2-41d7-85b5-99afb86874fe`
  - summary: Recovered from 1970 state by reapplying xli_patch_rollover runtime branch+stub; date bytes at 0x0031a40c now show 02/10/2026.
  - lesson: Fast recovery remains: apply runtime patch and verify raw date bytes. This is still RAM-only and must be reapplied after reboot.
- `2026-02-11T00:16:21+00:00` | `success` | `95ea4cc7-59d1-4170-a55b-c4a7852aa09d`
  - summary: Added canonical persistent GPS rollover patch procedure to RUNBOOK section 71 with exact build/stage/burn/verify/rollback steps and hard do-not-do rules.
  - lesson: Use only branch-stub 0x0017da20 workflow from stock image with single-session BH+bu burn and post-burn status-only verification. Reject inline patch at 0x000b6d70 and reject cave 0x0027c600.
- `2026-02-11T00:20:00+00:00` | `success` | `f7d2615f-3795-4b4b-a39b-12df844c1466`
  - summary: Created ready-to-go deployment bundle at <PATH> with patched image, staged filenames, stock

rollback images, SHA256 manifest, and 3-page PDF instructions.
  - lesson: Use bundle as canonical handoff package for matching 192-8001V1_106 devices; verify SHA256 before every burn.
- `2026-02-11T00:23:25+00:00` | `success` | `bfe9c2e8-41bf-40a6-85c9-84b714e82dfb`
  - summary: Created ready-to-go patch bundle with xli_autopatch_deploy.py (configurable --target-ip and --ftp-host), added AUTOPATCH_USAGE.md, added XLi_Autopatch_Deploy_Quickstart.pdf, refreshed SHA256SUMS, and documented section 72 in <PATH>
  - lesson: Use single telnet burn session; keep write protection unless sector usage is proven; verify hashes before deploy.

Appendix G — Low-level patch command history (runtime and persistent campaigns)

The following command patterns were used in byte-level experiments. They are included to preserve complete procedural history.

G.1 Runtime patch byte writes (RAM-only campaigns)
```text
d 0x547c0,8,1
m 0x547c0,1
01 2B 82 E2 00 20 83 E5

m 0x167598,1 ; BA 3C 82 E2
m 0x167790,1 ; BA 3C 82 E2
m 0x1679b0,1 ; BA 3C 82 E2
m 0x167b98,1 ; BA 3C 82 E2

m 0x000598ac ; DB 25 81 E2
m 0x000598b0 ; FF 28 82 E2

m <stub_addr>
00 40 A0 E1 00 20 94 E5 01 2B 82 E2 00 20 84 E5 98 02 CC EA
m 0x00059194 ; 61 FD 33 EA

m 0x0017d50c ; 15 20 83 E2
m 0x000b6f88 ; 20 20 82 E2  (year-only test)

m 0x000b6d78 ; <branch to stub>
m 0x000b6f88 ; 0x200c 0xe282  (restore +0x0c add)
```

G.2 Direct GPS hint injection command pattern
```text
copy "ftp:hint.bin","/ramdrv/hint.bin"
buf=malloc(64)
fd=open("/ramdrv/hint.bin",0,0)
read(fd,buf,64)
fd2=open("/tyCo/1",2,0)
write(fd2,buf,17)
GPSBpOutputEnable(0x30006000,&one,1)
GPSBpSendMsg(0x30006000,buf,17)
GPSBpStatus(0x30006000)
GPSBpLocked(0x30006000)
```

G.3 Burn API / NI burn command history
```text
burnHost("<FTP_IP>",1)
burnFile("192-8001.bin",1,1)

F100 BH:<FTP_IP>,192-8001.bin
F100 bu
F100 BH:<FTP_IP>,192-8002V1_106.fs
F100 bf
F100 BASET AUTO
```

```

### G.4 Read-only safety probes before any mutation
```text
showFlashSizes()
SizeofFlash()
get_flash_memptr()
lkup "GPSBp"
lkup "SA_M12"
lkup "gpsTimeToUnixEpoch"
lkup "gpsTimeNormalize"
d 0x10000000,32,1
d 0x100a0000,32,1
d 0x10100000,32,1
d 0x105e0200,32,1
d 0x107e0000,32,1
```
```