Boot ROM Layout — Release 1

Kenneth J Hamer-Hodges

May 2026

Boot ROM Layout

v1.0 — 2026-04-29 CONFIDENTIAL

The Boot ROM is a 1024-word (4 KiB) read-only instruction memory, defined in `hardware/boot_rom.py` and instantiated as the `BootRom` class. The address bus is 10 bits wide (`addr[9:0]`), data bus is 32 bits (`data[31:0]`), and read latency is one clock cycle (registered output).

IMEM Map

```
 Word Index      Byte Address     Region
──────────────────────────────────────────────────────
[  0 : 255 ]    0x000 – 0x3FC    BOOT_PROGRAM  (256 words)
[256 : 511 ]    0x400 – 0x7FC    NUC_PROGRAM   (256 words)
[512 : 680 ]    0x800 – 0xAA4    SLIDERULE_CODE (169 words)
[681 : 1023]    0xAA8 – 0xFFC    (reserved, zero-filled)
──────────────────────────────────────────────────────
```

---

1. BOOT_PROGRAM `[0:255]`

Secure-boot firmware. 13 real instructions, remainder zero-padded to 256.

Word	Instruction	Phase	Purpose
0	`CHANGE CR12, CR12, #1`	B:02 INIT_THRD	Switch to thread context
1	`LOAD CR1, CR6[0]`	B:03 INIT_ABSTR	Code/constants R|X GT → CR1
2	`LOAD CR2, CR6[1]`	B:03	Boot code X GT → CR2
3	`TPERM CR2, #X`	B:03	Restrict to X only
4	`LAMBDA CR2`	B:03	Enter boot code — **seal checkpoint 1**
5	`LOAD CR0, CR6[4]`	B:04 LOAD_NUC	Salvation E-GT → CR0
6	`TPERM CR0, #E`	B:04	Restrict to E only
7	`CALL CR0, CR0`	B:04	Enter user abstraction — **seal checkpoint 2**
8	`LOAD CR7, CR6[1]`	Epilogue	Reload boot code GT
9	`TPERM CR7, #X`	Epilogue	Restrict to X
10	`LAMBDA CR7`	Epilogue	Re-enter boot finalisation
11	`RETURN`	Epilogue	Boot complete; bare RETURN (mask field not implemented)
12	`SAVE CR6, CR1, #2`	Epilogue	Persist Thread GT to c-list[2]

2. NUC_PROGRAM `[256:511]`

LED0 blink demo — the Salvation abstraction (NS Slot 4). 17 instructions, padded to 256 words.

```
Register use:
  DR0 = 0 (hardwired zero)
  DR1 = 1 (on value, set once)
  DR2 = inner delay counter   (16383 iterations)
  DR3 = outer delay counter   (380 iterations)
  CR3 = LED_DEV capability    (loaded from c-list[8])

Timing at 50 MHz:
  inner × outer = 16383 × 380 = 6,225,540 loop iterations
  ~24.9 M cycles per phase ≈ 0.498 s → ~1 Hz blink
```

Word	Instruction	Purpose
0	`LOAD CR3, CR6[8]`	LED_DEV capability into CR3
1	`IADD DR1, DR0, #1`	DR1 = 1
2	`DWRITE CR3[0], DR1`	LED0 = ON
3–8	delay loop (ON phase)	inner/outer ISUB+BRANCH
9	`DWRITE CR3[0], DR0`	LED0 = OFF
10–15	delay loop (OFF phase)	inner/outer ISUB+BRANCH
16	`BRANCH AL, #-14`	Jump back to word 2

Lump Header

At DMEM word 255 (byte `0x3FC`):

```
magic = 0x1F   n_minus_6 = 0   cw = 17   cc = 0   typ = 0
```

---

3. SLIDERULE_CODE `[512:680]`

NS Slot 16 — Layer 3 Mathematics. E-perm, chainable. 169 words total: 16-word dispatch table + 153 words of method bodies.

Dispatch Convention

Caller sets **DR3 = method index** before CALL. The first 16 words form a dispatch table of 8 × (ISUB + BRANCH EQ) pairs:

```
ISUB  DR2, DR3, #0     ; compare with method 0
BRANCH EQ, +offset_0   ; jump to Add body
ISUB  DR2, DR3, #1     ; compare with method 1
BRANCH EQ, +offset_1   ; jump to Sub body
  ...                   ; etc. for all 8 methods
```

Method Table

Index	Method	Words	Description
0	Add	5	DR1 + DR2 → DR1
1	Sub	4	DR1 − DR2 → DR1
2	Mul	35	DR1 × DR2 → DR1 (shift-add)
3	Div	40	DR1 ÷ DR2 → DR1 (long division)
4	Sqrt	38	√DR1 → DR1 (Newton’s method, 20 iterations)
5	Pow	29	DR1 ^ DR2 → DR1 (square-and-multiply)
6	ToDegrees	1	Stub (RETURN)
7	ToRadians	1	Stub (RETURN)

Total: 16 dispatch + 153 body = 169 words

Lump Header

At DMEM word 511 (byte `0x7FC`):

```
magic = 0x1F   n_minus_6 = 2   cw = 169   cc = 0
Header word = 0xF902A400
```

---

4. Reserved `[681:1023]`

343 words, zero-filled. Available for future abstractions.

DMEM Structures

These are loaded into data memory at boot, not part of IMEM.

DEMO_NAMESPACE (19 slots × 3 words = 57 words)

Each NS entry is 3 words: `word0_location`, `word1_w2` (limit + gt_seq), `word2_w3` (CRC-16 seal).

Slot	Name	Perms	Location	Notes
0	Boot.NS	R|W	NS_TABLE_BASE	NS root, 64 words
1	Boot.Thread	R|W	0x0100	Thread lump, threadLumpWords
2	(free/null)	—	follows Thread	Free heap region, 64 words; Boot.Abstr director eliminated (Task #247)
3	Boot.Abstr	E	follows slot 2	Boot Abstraction lump, abstractionLumpWords (default 256)
4	Salvation	E	0x03FC	NUC_PROGRAM, 64 words
5	Navana	E	0x0500	Namespace controller
6	Mint	E	0x0600	Capability minting
7	Memory	E	0x0700	Memory management
8	Scheduler	E	0x0800	Thread scheduling
9	Stack	E	0x0900	LIFO stack
10	DijkstraFlag	E	0x0A00	Synchronisation
11	UART	R|W	0x40000014	MMIO: TX, STATUS, RX
12	LED	R|W	0x40000000	MMIO: LED0–LED4
13	Button	R	0x40000028	MMIO: button bitmask
14	Timer	R|W	0x4000002C	MMIO: TICKS, TOD, ALARM
15	Display	E	0x0F00	Reserved for HDMI
16	SlideRule	E	0x07FC	256 words, cw=169
17	(empty)	—	—	Reserved
18	Constants	R	0x1200	Read-only math constants

DEMO_CLIST (17 Golden Tokens at physical end of Boot.Abstr)

Initial c-list for Boot.Abstr (NS Slot 3). CR6 points here after boot. Slot 2 (free/null) has no c-list; the Boot.Abstr director indirection was eliminated in Task #247 — B:04 LOAD_NUC loads slot 3 directly.

Idx	Perms	Slot	Name	Notes
0	R|W	0	—	Boot-internal: memory-manager GT (NS slot 0)
1	—	1	—	Boot-internal: Boot.Thread GT
2	—	—	—	(free/null — Task #247)
3	E	3	—	Boot-internal: Boot.Abstr self-reference E-GT (same slot)
4	E	4	Salvation	First user abstraction
5	E	5	Navana	Namespace controller
6	E	6	Mint	Capability minting
7	E	7	Memory	Memory management
8	R|W	12	LED[0]	MMIO, b_flag=1
9	R|W	12	LED[1]	MMIO, b_flag=1
10	R|W	12	LED[2]	MMIO, b_flag=1
11	R|W	12	LED[3]	MMIO, b_flag=1
12	R|W	12	LED[4]	MMIO, b_flag=1
13	R|W	12	LED[5]	MMIO, b_flag=1
14	R|W	11	UART	MMIO, b_flag=1 (3 regs)
15	R	13	Button	MMIO, b_flag=1 (1 reg)
16	R|W	14	Timer	MMIO, b_flag=1 (5 regs)

Indices 0–3 are boot-internal (used by BOOT_PROGRAM firmware only). Indices 4–7 are user abstraction GTs (Salvation, Navana, Mint, Memory). Indices 8–16 are hardware device GTs (6 LED channels, UART, Button, Timer).

Slot 2 has no c-list (free/null — Task #247). The boot indirection via a director c-list[3] E-GT has been eliminated; B:04 loads Boot.Abstr (slot 3) directly.

The `b_flag=1` on device GTs marks them as IDE-bound to a physical peripheral. The b_flag bit is excluded from the CRC seal input so the runtime can clear it on un-bind without recomputing the NS entry seal.

See Also

- **Lump-Architecture.md — Lump object structure, Header Word encoding, and zone layout**

  Confidential — Kenneth Hamer-Hodges — April 2026