Open Source 西门子

Beijing · 2026

Oct. 29
Industrial Open Source

Oct. 30
Open Source Robotics

Agenda

从移植到性能优化：2026 年 Chromium V8 在 RISC-V 平台上的技术进展



创新求实 · 永竞一流

陆亚涵
yahan@iscas.ac.cn

SECTION

01

第 1 / 4 章

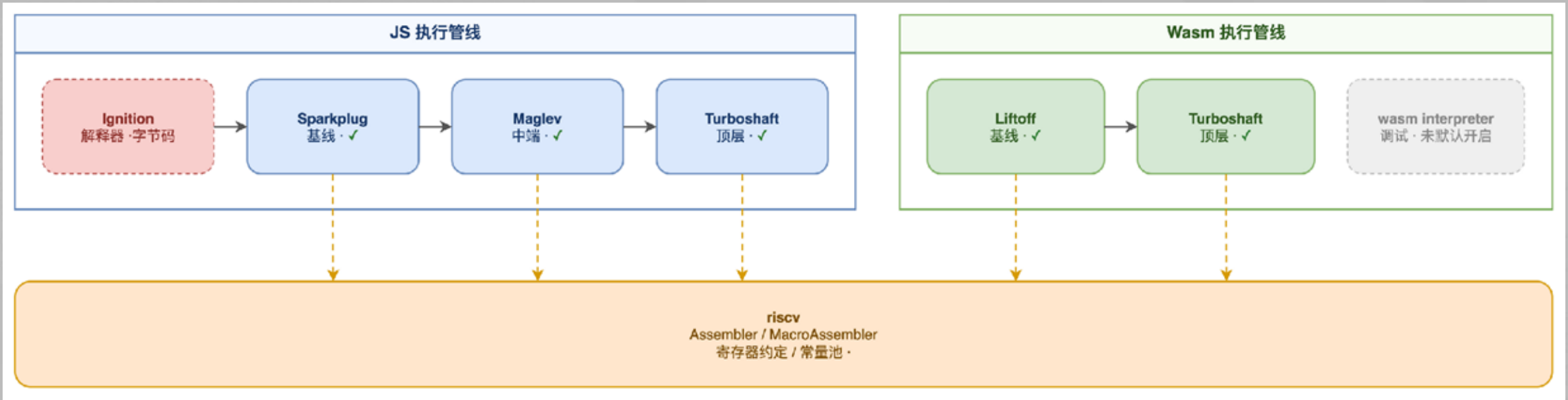
认识 V8

V8是什么

V8 是 Google 开源的高性能 JavaScript / WebAssembly 引擎：内嵌于 Chrome 浏览器，负责网页 JavaScript 的解析与执行，支撑动态交互与异步逻辑——页面渲染由 Blink 完成，V8 是渲染背后的“执行心脏”。同时，V8 也是 Node.js 的运行时内核，广泛服务于服务器端开发与前端工具链生态。

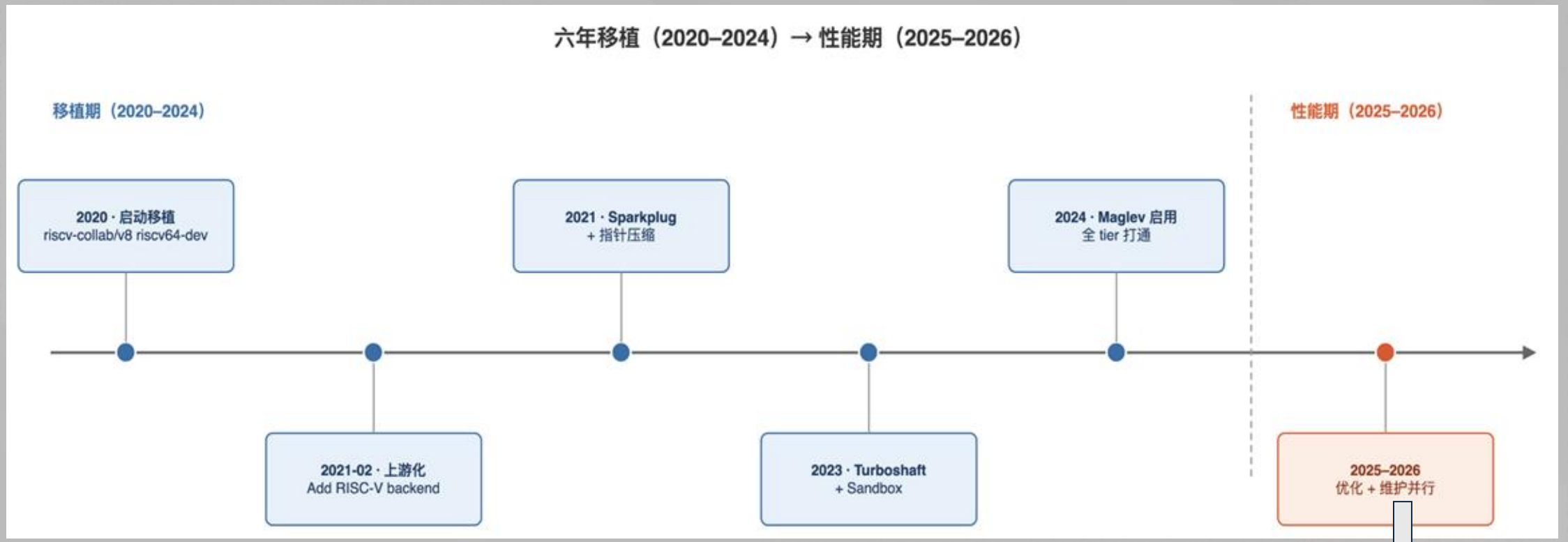


V8 执行管线



作为动态语言虚拟机，V8 采用多层次执行架构：解释器（Ignition）负责快速启动，多层 JIT 编译器（Sparkplug、Maglev、TurboFan）逐级优化热点代码，并配合精确式垃圾回收实现内存自动管理；启动快照（Snapshot）机制通过预序列化上下文加速冷启动。

功能移植结束，性能优化与日常维护双线并行



SECTION

02

第 2 / 4 章

RVA23 for 2026

认识 V8
RVA23
后端优化
性能验证

RVA23 for 2026

- 硬件 Hardware

  SpacemiT K3　　2026 H1 出货

- 内核

  Linux v7.0-rc1　RVA23 强制覆盖率 68% → 100%

- 发行版 Distribution

  Ubuntu 26.04 LTS

  首个 RVA23 LTS 发行版 2026-04发布



V8 RISCV 扩展矩阵支持总览

基础浮点	FPU	ZFH	ZFA
位操作（B）	ZBA	ZBB	ZBS
标量增强	ZICOND	ZCB	RVC
控制流完整性	ZICFISS		
向量	RVV		

目标基线
RVA23U64

SECTION

03

第 3 / 4 章

RISC-V 后端优化

优化一 启用 rva23u64：Builtin 体积减小

Profile Impact on Builtin Size

口径：d8 --print-builtin-size 逐行求和 · 2446 个 builtin · release · riscv64 模拟器（riscv_use_rvv=true）

Builtin 总体积 (byte)	指针压缩 关	指针压缩 开
rv64gcv	2,620,396	2,762,304
rva23u64	2,567,556	2,690,052
差值	**−52,840 (−2.02%)**	**−72,252 (−2.62%)**

类型	rv64gcv·关指针压缩	rva23·关指针压缩	Δ	rv64gcv·开指针压缩	rva23·开指针压缩	Δ
ASM	24,904	24,608	−296	26,452	26,064	−388
BCH	455,736	440,100	−15,636	475,644	452,436	−23,208
CPP	11,280	11,280	±0	11,280	11,280	±0
TFC	658,432	643,644	−14,788	694,844	675,716	−19,128
TFH	426,688	420,900	−5,788	452,412	443,296	−9,116
TFJ	884,496	869,556	−14,940	934,172	916,812	−17,360
TFS	158,860	157,468	−1,392	167,500	164,448	−3,052

优化二：Sub32 优化word32相等判断

RISC-V 无条件码寄存器 · Word32 Eq/Ne 只看低 32 位 ·

Before · 旧路径

```
slli t0, a, 32
slli t1, b, 32
beq  t0, t1, <target>
```

问题：两次 slli 只为把 32 位值挪到高位——对「只看低 32 位」的相等判断是纯开销

After · 新路径

```
subw t0, a, b
beq  t0, zero, <target>
```

优化思路：subw 自带符号扩展

subw 会对 32 位结果做符号扩展；
所以 subw t, a, b 结果为 0
⇔ a、b 低 32 位相等。

用算术的符号扩展，替代显式符号扩展。

优化后效果:
builtin size减少：−66,760 B（−2.42%）

优化三：sext.w 消除

32 位 ALU / 窄加载天然自带符号扩展 · 3f7ae316feb 等 6 提交

背景

addw / subw / sllw / sraw / srlw
andiw/ lw
结果天然按 32 位符号扩展

提交	Before	After
3f7ae3	add32(sext(a), sext(b))	add32(a, b)
3f7ae3	sext(add32(x, y))	add32(x, y)
0a62f7	sext(trunc(x))	x
0a62f7	cmp0_32(trunc(x))	cmp0_64(x)
25f050	cmp0_32(and32(a, b))	cmp0_64(and32(a, b))
ccde78	store32(p, trunc(x))	store32(p, x)
ccde78	sar32(trunc(x), n)	sar32(x, n)
ccde78	cmp0_32(load32(p))	cmp0_64(load32(p))
035307	mulovf32(sext(a), sext(b))	mulovf32(a, b)
02bf58	mulovf32(a, sext(K))	mulovf32(a, K')*

* K'：li 时已完成 32 位符号扩展

效果

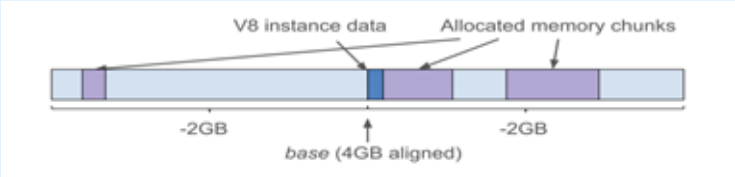
builtin size减少：−16.4 KB(−0.59%)
826 个 builtin 受影响

优化原因

指针压缩 + 静态 root → V8 的类型判断大量退化为 32 位比较

① 指针压缩 Pointer Compression

ptr = base+index



64 位指针拆分成基地址base 和 32 位index：内存base只保留一份, 对象存储时只存 index

代价: 寻址范围限制在 4gb

② 静态 Static Roots

v8中常用根对象固定在内存中固定偏移地址, undefined固定偏移为0x11

x === undefined → beq x, 0x11

(undefined / null / true / false 同理)

③ RISC-V 特点

① 无条件码寄存器

② 32 位值符号扩展到 64 位寄存器

③ 32 位 ALU / 加载自带符号扩展

优化四 立即数溢出检查：不算溢出，比较大小

cf146b30c24 · 7 条 → 3 条 ·

BEFORE · 旧路径 — 7 条

```
li      t3, 1
addw    a4, a3, t3
xor     s3, a4, a3
xor     t3, a4, t3
and     s3, s3, t3
sext.w  s3, s3
blt     s3, zero_reg, overflow
```

旧路径走通用溢出检查：真的把溢出位算进 kScratchReg，再判符号

AFTER · 新路径 — 3 条

```
addiw   a4, a3, 1
sext.w  s3, a3
bge     s3, a4, overflow
```

sext.w 不可省：先把 x 规整成 32 位符号扩展形，有符号比较才成立

关键

x ± imm 溢出 ⇔ result 与 x 的大小关系反转 → 不用算溢出位，直接比较即可

add · imm > 0	add · imm < 0	sub · imm > 0	sub · imm < 0
溢出 → blt	溢出 → bgt	溢出 → bgt	溢出 → blt
result < x	result > x	result > x	result < x

−1,608 B（−0.06%）· 59 个 builtin 受影响 · add / sub × imm 正负 四种形态全部 7 → 3

优化五 常量池：不用搬，放到代码末尾

3ef9d4cc32d → 092658c7072 → ba6172ecda9 · auipc ±2GB：池不必就近，全代码末尾发一次

BEFORE · 内联发池

```
...code...
auipc   t0, entry
ld      t0, t0(t0)
...code...
j       L            ← 每池 1 条 j
auipc   x0, #words
j       0            ← 池守卫
pool:   .quad ...
L:      ...code...
```

AFTER · 末尾单池

```
...code...
auipc   t0, entry
ld      t0, t0(t0) .
..code...
FinishCode():
auipc   x0, #words
j       0            ← 池守卫
pool:   .quad ...
```

常量池池一旦 emit 就关闭写入：后续常量再也进不去

常量池位置	可能在代码中间	**代码末尾**
跳过池的 j	每池 1 条（4 B）	**0 条**
汇编器检查	每 128 条指令一次	**不再检查**
去重范围	单个池内	**整个 code object**

SECTION

04

第 4 / 4 章

RISC-V · V8 2026

性能验证

性能验证

① RV64GC vs RVA23

SpacemiT K3

套件	RV64GC		RVA23	提升
Octane 分↑	11,495.0	→	**11,804.9**	**+2.70%**
SunSpider ms↓	426.0	→	**417.2**	**+2.11%**
Kraken ms↓	3,038.4	→	**2,572.8**	**+18.10%**

同二进制、同板；正值 = RVA23 更快

② 2025-01 vs 2026.10

K3 · 2025.01 构建 → 2026.10 RVA23

套件	2025-01		RVA23	**提升**
Octane 分↑	10,480.2	→	**11,804.9**	**+12.6%**
SunSpider ms↓	473.4	→	**417.2**	**+13.5%**
Kraken ms↓	4,114.8	→	**2,572.8**	**+59.9%**

同一块 K3；ms 套件的提升 = 旧耗时 ÷ 新耗时 − 1

③ Kraken 分析

k3 · 原始耗时 ms，2025-01 → 当前 RVA23

分组	2025-01		RVA23	**提速（倍）**
4 个 crypto	1,862.9	→	**496.8**	**×3.75**
其余 10 项	2,251.9	→	**2,076.0**	**×1.08**

总结：从「能跑」到「跑得快」

① 移植已完成：五级执行管线全链路打通

2020 riscv-collab/v8 riscv64-dev → 2021-02 上游化 → 2024 Maglev 启用

Ignition → Sparkplug → Maglev → Turboshaft → Liftoff

② 扩展已接入 13 项

浮点 / 位操作 / 标量增强 / 控制流完整性 / 向量 —— 与 RVA23 profile 逐项对齐

FPU　FP64FPU　RVV　ZBA　ZBB　ZBS　ZFA　ZFH　ZVFH　ZICOND　ZICFISS　RVC　ZCB

③ 优化明显：编译期瘦身 + 真机提速

-2.0~2.6%
builtins 体积 · kRVA23U64

+2.1%
SunSpider 测试集

+2.7%
Octane 测试集

+18.1%
Kraken 测试集· crypto +64.5%

移植收官只是入场券——13 项扩展与五级管线就位后，收益取决于负载能否吃到扩展

ISCAS
中国科学院软件研究所
Institute of Software Chinese Academy of Sciences

谢谢！



Open Source 西门子

Beijing · 2026

Oct. 29
Industrial Open Source

Oct. 30
Open Source Robotics

Agenda