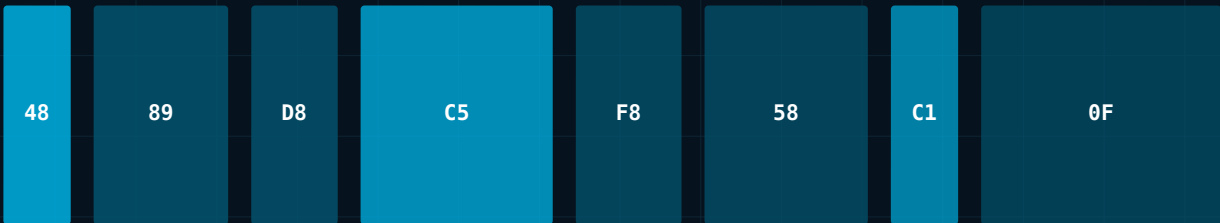


The x86-64 Architecture Atlas

Registers, Instruction Encoding, System Programming, Memory, Interrupts, SIMD, and ABI

X64



GPR	RFLAGS	ENC	PAGING	IDT	AVX	ABI
-----	--------	-----	--------	-----	-----	-----

A3 PORTRAIT · VISUAL SYSTEMS REFERENCE · FIRST EDITION

REFERENCE SCOPE

- Variable-length encoding
- Legacy-to-modern register state
- Paging and privilege
- SSE / AVX / AVX-512 / AMX

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

64 interior pages · English edition · Official-reference baseline

Independent technical publication — no vendor affiliation implied.

The x86-64 Architecture Atlas

Registers, Instruction Encoding, System Programming, Memory, Interrupts, SIMD, and ABI

X64

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

EDITION

First Edition · 2026

FORMAT

A3 Portrait · 297 × 420 mm

SCOPE STATEMENT

This volume is a dense visual reference for software-visible x86-64 architecture. It is designed for operating-system developers, compiler and JIT engineers, debugger authors, reverse engineers, firmware developers, and low-level programmers.

PUBLICATION NOTE

Official architecture specifications remain normative. This atlas compresses programming-visible concepts into traceable diagrams, tables, and cross-references; it does not replace formal instruction pseudocode or vendor-specific implementation manuals.

How to Read This Volume

A consistent editorial language for dense technical information

REG

ARCHITECTURE STATE

Registers, flags, system state, and privilege-visible control.

ENC

ENCODING FIELD

Instruction bits, prefixes, opcode maps, and immediate reconstruction.

MEM

MEMORY EFFECT

Address formation, ordering, cache/TLB effect, and translation behavior.

EXC

EXCEPTION PATH

Fault, trap, interrupt, privilege transition, and return mechanism.

STS

STATUS MARKER

Required, optional, legacy, deprecated, ratified, or implementation-defined.

SRC

SOURCE CODE

Compact official-reference key printed at the foot of each technical page.

STATUS LANGUAGE

REQUIRED

OPTIONAL

LEGACY

DEPRECATED

RATIFIED

IMPLEMENTATION-DEFINED

TABLE READING ORDER

1 Name and purpose

3 Architectural state affected

5 Exceptions, caveats, and source

2 Operands and width

4 Privilege and memory ordering

Detailed Contents I

64 interior pages · covers excluded

01 Half Title

02 Edition and Scope

03 How to Use This Atlas

04 Symbols and Status Markers

05 Contents I

06 Contents II

07 Lineage, Modes, and Compatibility

08 Programmer-Visible Architecture Map

09 General-Purpose Registers

10 Subregister Overlap and Write Semantics

11 RIP and RFLAGS

12 Segment Registers and Base State

13 Control Registers

14 XCRs, EFER, and Selected MSRs

15 Privilege Rings and Descriptor Tables

16 GDT, LDT, and TSS

17 Effective Address Formation

18 ModR/M Byte

19 SIB Byte

20 Legacy Prefixes

21 REX, VEX, and EVEX

22 Instruction-Length and Decode Flow

23 Data Transfer Instructions

24 Address Generation and Stack Operations

25 Integer Arithmetic

26 Multiplication and Division

27 Logic and Bit Operations

28 Shifts and Rotates

29 Compare and Conditional State

30 Branches, Calls, and Returns

31 System Call Entry and Return

32 x87 Floating-Point State

33 SSE Register and Instruction Model

Detailed Contents II

64 interior pages · covers excluded

34 AVX, AVX2, and FMA

35 AVX-512 and Opmask State

36 AMX Tile State

37 Cryptographic and Bit-Manipulation Extensions

38 CPUID and Feature Discovery

39 Floating-Point Environment

40 XSAVE-Managed Extended State

41 Canonical Addresses

42 Four-Level Paging

43 Five-Level Paging

44 Page-Table Entry Fields

45 TLB Invalidation and PCID

46 Protection Keys, CET, and Control-Flow Protection

47 Architectural Exceptions

48 IDT and Gate Formats

49 Interrupt Routing and APIC Context

50 Privilege Transitions and Fast System Calls

51 Virtualization Overview: VMX and SVM

52 Atomic Operations and LOCK

53 Memory Ordering and Fences

54 Cache and Memory-Control Operations

55 Context-Switch State Checklist

56 Debug Registers and Breakpoints

57 Performance Monitoring and Time Sources

58 System V AMD64 ABI

59 Microsoft x64 ABI

60 Stack Frames and Unwinding

61 ELF, PE/COFF, and Relocations

62 Assembler Syntax and Toolchain Notes

63 Quick Reference Tables

64 Official English References

65 Index, Errata, and Revision Record

x86-64 Programmer-Visible Architecture Map

Editorial sample: architecture state, privilege, memory, and extension layers

ARCHITECTURAL STATE

RAX-R15

RIP

RFLAGS

CR0 / CR2 / CR3 / CR4

XCR0 / EFER

XMM-YMM-ZMM

K0-K7 / TILECFG

Descriptor and interrupt state

MODE AND PRIVILEGE

Long mode · Rings 0-3 · GDT/IDT/TSS

INSTRUCTION MODEL

1-15 bytes · prefixes · opcode maps · ModR/M · SIB

MEMORY SYSTEM

Canonical VA · 4/5-level paging · PCID · TLB

EVENT DELIVERY

Exceptions · IDT gates · APIC · IRETQ

MODERN COMPUTE

SSE · AVX2 · AVX-512 · AMX · crypto

Every full-volume technical page maps back to one or more layers shown here.

Register State and Instruction Encoding

High-density A3 portrait layout with vector tables and concise explanations

REGISTER / STATE MAP

RAX	RBX	RCX	RDY
RSI	RDI	RBP	RSP
R8	R9	R10	R11
R12	R13	R14	R15

ENCODING / FIELD MODEL

Prefixes	0-4+ bytes	Legacy / REX / VEX / EVEX
Opcode	1-3+ bytes	Primary and extended maps
ModR/M	1 byte	mod · reg/opcode · r/m
SIB	0 or 1 byte	scale · index · base
Displacement	0/1/2/4/8 bytes	Address contribution
Immediate	0/1/2/4/8 bytes	Constant or relative target

Variable length requires ordered prefix and field decoding; architecturally valid length is at most 15 bytes.

Official English References

The complete volume uses section-level source keys on every technical page

PRIMARY NORMATIVE SOURCES

- 01 Intel® 64 and IA-32 Architectures Software Developer’s Manual, Volumes 1-4.
- 02 AMD64 Architecture Programmer’s Manual, Volumes 1-5, Revision 4.10.
- 03 System V Application Binary Interface: AMD64 Architecture Processor Supplement.
- 04 Microsoft x64 Calling Convention and PE/COFF Specification.
- 05 DWARF Debugging Information Format, Version 5.

REFERENCE-KEY EXAMPLE

INTEL-SDM-V1 §3.4 · INTEL-SDM-V3A §4 · AMD-APM-V2 §5

Revision and access-date metadata are stored in the bibliography and frozen for each published edition. Corrections are tracked through a public errata register.

EDITORIAL COMMITMENT

- No unsourced register maps or bit positions.
- No mnemonic equivalence without behavior verification.
- Architecture revision shown for optional and recent features.
- Official manuals remain normative when ambiguity exists.

A reference built for the moment when a diagram must be faster than a manual search — without sacrificing traceability.

DESIGNED FOR

- Operating-system and kernel developers
- Compiler, assembler, linker, and JIT engineers
- Firmware and hypervisor developers
- Debugger, profiler, and reverse-engineering specialists
- Advanced low-level programmers and architecture educators

SERIES PRINCIPLE

One architecture per volume. One dedicated comparison volume. A unified visual grammar across registers, instruction encodings, privilege, virtual memory, exceptions, atomics, vectors, ABI, and binary interfaces.

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

THE PROCESSOR ARCHITECTURE ATLAS SERIES

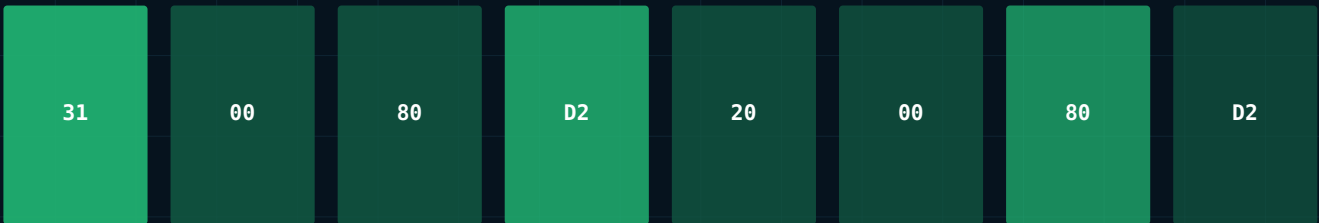
Independent technical reference · English edition · A3 portrait

ISBN / BARCODE AREA

The Arm AArch64 Architecture Atlas

Registers, A64 Encoding, Exception Levels, Memory Translation, SVE/SME, Barriers, and ABI

A64



A3 PORTRAIT · VISUAL SYSTEMS REFERENCE · FIRST EDITION

REFERENCE SCOPE

- Fixed 32-bit A64 encoding
- Exception levels and security states
- Stage 1 / Stage 2 translation
- NEON / SVE / SVE2 / SME

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

56 interior pages · English edition · Official-reference baseline

Independent technical publication — no vendor affiliation implied.

The Arm AArch64 Architecture Atlas

Registers, A64 Encoding, Exception Levels, Memory Translation, SVE/SME, Barriers, and ABI

A64

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

EDITION

First Edition · 2026

FORMAT

A3 Portrait · 297 × 420 mm

SCOPE STATEMENT

This volume is a dense visual reference for software-visible Arm AArch64 architecture. It is designed for operating-system developers, compiler and JIT engineers, debugger authors, reverse engineers, firmware developers, and low-level programmers.

PUBLICATION NOTE

Official architecture specifications remain normative. This atlas compresses programming-visible concepts into traceable diagrams, tables, and cross-references; it does not replace formal instruction pseudocode or vendor-specific implementation manuals.

How to Read This Volume

A consistent editorial language for dense technical information

REG

ARCHITECTURE STATE

Registers, flags, system state, and privilege-visible control.

ENC

ENCODING FIELD

Instruction bits, prefixes, opcode maps, and immediate reconstruction.

MEM

MEMORY EFFECT

Address formation, ordering, cache/TLB effect, and translation behavior.

EXC

EXCEPTION PATH

Fault, trap, interrupt, privilege transition, and return mechanism.

STS

STATUS MARKER

Required, optional, legacy, deprecated, ratified, or implementation-defined.

SRC

SOURCE CODE

Compact official-reference key printed at the foot of each technical page.

STATUS LANGUAGE

REQUIRED

OPTIONAL

LEGACY

DEPRECATED

RATIFIED

IMPLEMENTATION-DEFINED

TABLE READING ORDER

1 Name and purpose

3 Architectural state affected

5 Exceptions, caveats, and source

2 Operands and width

4 Privilege and memory ordering

Detailed Contents I

56 interior pages · covers excluded

01 Half Title

02 Edition and Scope

03 How to Use This Atlas

04 Symbols and Status Markers

05 Contents I

06 Contents II

07 A-Profile Architecture Overview

08 AArch64 Programmer-Visible State

09 X and W General-Purpose Registers

10 SP, PC, XZR, and Register Aliases

11 PSTATE and NZCV

12 System Register Naming and Access

13 Key EL1 System Registers

14 Key EL2 and EL3 System Registers

15 A64 Instruction-Encoding Map

16 Immediate and Register Fields

17 Address Generation: ADR and ADRP

18 Load/Store Addressing Modes

19 Pair, Literal, and Exclusive Accesses

20 Integer Arithmetic

21 Logical, Shift, and Bitfield Operations

22 Compare and Conditional Select

23 Branches, Calls, and Returns

24 Floating-Point Register Model

25 NEON Advanced SIMD

26 SVE Scalable Vectors

27 SVE2 Capability Map

28 SME and ZA Matrix State

Detailed Contents II

56 interior pages · covers excluded

29 Exception Levels EL0-EL3

30 Security States and World Transitions

31 Stage 1 Translation Regimes

32 Stage 2 Translation

33 Granules, Levels, Blocks, and Pages

34 Descriptor and Permission Fields

35 Memory Attributes and Shareability

36 Synchronous Exceptions

37 IRQ, FIQ, and SError

38 Vector Table and Exception Return

39 GIC Relationship and Interrupt Routing

40 AArch64 Memory-Ordering Model

41 DMB, DSB, and ISB

42 Exclusive and LSE Atomics

43 Cache Maintenance

44 TLB Maintenance

45 Debug and Watchpoint State

46 PMU and Trace Overview

47 PAC, BTI, and MTE

48 Feature Identification Registers

49 AAPCS64 Register Convention

50 Stack Frames and Unwinding

51 ELF, Relocations, and TLS

52 GNU and LLVM Assembly Syntax

53 Compiler Intrinsics and OS Patterns

54 Quick Reference Tables

55 Official English References

56 Index, Errata, and Revision Record

Arm AArch64 Programmer-Visible Architecture Map

Editorial sample: architecture state, privilege, memory, and extension layers

ARCHITECTURAL STATE

X0-X30

SP / PC

PSTATE / NZCV

SCTLR_ELx

TTBR / TCR / MAIR

V0-V31

Z / P / ZA state

Exception and vector state

MODE AND PRIVILEGE

AArch64 · EL0-EL3 · Security states

INSTRUCTION MODEL

Fixed 32-bit A64 · major decode groups

MEMORY SYSTEM

Stage 1/2 · 4K/16K/64K granules · ASID

EVENT DELIVERY

Sync · IRQ · FIQ · SError · ERET

MODERN COMPUTE

NEON · SVE · SVE2 · SME · crypto

Every full-volume technical page maps back to one or more layers shown here.

Register State and A64 Encoding

High-density A3 portrait layout with vector tables and concise explanations

REGISTER / STATE MAP

X0	X1	X2	X3	X4	X5	X6	X7
X8	X9	X10	X11	X12	X13	X14	X15
X16	X17	X18	X19	X20	X21	X22	X23
X24	X25	X26	X27	X28	X29	X30	SP

ENCODING / FIELD MODEL

Major op	Bits 31:25	Top-level A64 decode group
sf / size	Variant field	Operation width or element size
Rm / Rn / Rd	5 bits each	Source and destination registers
imm fields	Variable width	Immediate value reconstruction
shift / extend	Subfields	Operand transformation
cond / op	Control fields	Condition or operation selector

A64 instructions are 32 bits; operation classes are recognized from high-order encoding fields.

Official English References

The complete volume uses section-level source keys on every technical page

PRIMARY NORMATIVE SOURCES

- 01

Arm® Architecture Reference Manual for A-profile Architecture.
- 02

Procedure Call Standard for the Arm 64-bit Architecture (AAPCS64).
- 03

ELF for the Arm 64-bit Architecture (AAELF64).
- 04

Arm® System Registers Reference.
- 05

DWARF for the Arm 64-bit Architecture.

REFERENCE-KEY EXAMPLE

ARM-ARM D1.3 · AAPCS64 §6 · AAELF64 §4

Revision and access-date metadata are stored in the bibliography and frozen for each published edition. Corrections are tracked through a public errata register.

EDITORIAL COMMITMENT

- No unsourced register maps or bit positions.
- No mnemonic equivalence without behavior verification.
- Architecture revision shown for optional and recent features.
- Official manuals remain normative when ambiguity exists.

A reference built for the moment when a diagram must be faster than a manual search — without sacrificing traceability.

DESIGNED FOR

- Operating-system and kernel developers
- Compiler, assembler, linker, and JIT engineers
- Firmware and hypervisor developers
- Debugger, profiler, and reverse-engineering specialists
- Advanced low-level programmers and architecture educators

SERIES PRINCIPLE

One architecture per volume. One dedicated comparison volume. A unified visual grammar across registers, instruction encodings, privilege, virtual memory, exceptions, atomics, vectors, ABI, and binary interfaces.

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

THE PROCESSOR ARCHITECTURE ATLAS SERIES

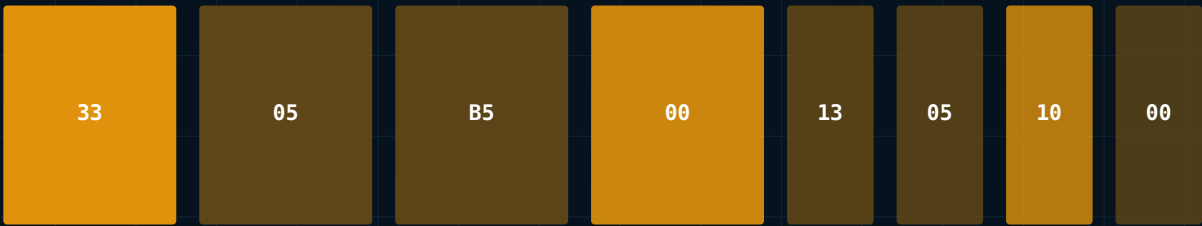
Independent technical reference · English edition · A3 portrait

ISBN / BARCODE AREA

The RISC-V Architecture Atlas

Base ISAs, Extensions, Encoding, Privilege, CSRs, Virtual Memory, Vectors, Atomics, and psABI

RV



x0-x31	CSR	FMT	U/S/M	SV48	RVV	psABI
--------	-----	-----	-------	------	-----	-------

A3 PORTRAIT · VISUAL SYSTEMS REFERENCE · FIRST EDITION

REFERENCE SCOPE

- Modular ISA and extension naming
- R/I/S/B/U/J instruction formats
- Privilege modes and CSRs
- RVV vector-length-agnostic design

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

56 interior pages · English edition · Official-reference baseline

Independent technical publication — no vendor affiliation implied.

The RISC-V Architecture Atlas

Base ISAs, Extensions, Encoding, Privilege, CSRs, Virtual Memory, Vectors, Atomics, and psABI

RV

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

EDITION

First Edition · 2026

FORMAT

A3 Portrait · 297 × 420 mm

SCOPE STATEMENT

This volume is a dense visual reference for software-visible RISC-V architecture. It is designed for operating-system developers, compiler and JIT engineers, debugger authors, reverse engineers, firmware developers, and low-level programmers.

PUBLICATION NOTE

Official architecture specifications remain normative. This atlas compresses programming-visible concepts into traceable diagrams, tables, and cross-references; it does not replace formal instruction pseudocode or vendor-specific implementation manuals.

How to Read This Volume

A consistent editorial language for dense technical information

REG

ARCHITECTURE STATE

Registers, flags, system state, and privilege-visible control.

ENC

ENCODING FIELD

Instruction bits, prefixes, opcode maps, and immediate reconstruction.

MEM

MEMORY EFFECT

Address formation, ordering, cache/TLB effect, and translation behavior.

EXC

EXCEPTION PATH

Fault, trap, interrupt, privilege transition, and return mechanism.

STS

STATUS MARKER

Required, optional, legacy, deprecated, ratified, or implementation-defined.

SRC

SOURCE CODE

Compact official-reference key printed at the foot of each technical page.

STATUS LANGUAGE

REQUIRED

OPTIONAL

LEGACY

DEPRECATED

RATIFIED

IMPLEMENTATION-DEFINED

TABLE READING ORDER

1 Name and purpose

3 Architectural state affected

5 Exceptions, caveats, and source

2 Operands and width

4 Privilege and memory ordering

Detailed Contents I

56 interior pages · covers excluded

01 Half Title

02 Edition and Scope

03 How to Use This Atlas

04 Symbols and Status Markers

05 Contents I

06 Contents II

07 The Modular RISC-V Architecture

08 ISA String and Extension Naming

09 Integer Register Atlas

10 ABI Register Names and Roles

11 Privilege Modes and Execution Context

12 CSR Addressing and Access Rules

13 Machine-Mode CSRs

14 Supervisor-Mode CSRs

15 R, I, and S Instruction Formats

16 B, U, and J Instruction Formats

17 Immediate Reconstruction

18 Compressed Instruction Formats

19 RV32I and RV64I Base Instructions

20 Loads, Stores, and Alignment

21 Branches, JAL, and JALR

22 PC-Relative Address Sequences

23 M Extension: Multiply and Divide

24 A Extension: LR/SC and AMOs

25 Floating-Point Register State

26 F, D, and Q Extensions

27 C Extension and Code Density

28 B Bit-Manipulation Extensions

29 Vector Register Model

Detailed Contents II

56 interior pages · covers excluded

30 VTYPE, VL, SEW, and LMUL

31 Vector Loads, Stores, and Masks

32 Vector Arithmetic and Reductions

33 **Scalar and Vector Cryptography**

34 Z Extension Capability Map

35 Trap Entry and Cause State

36 Direct and Vectored Trap Handling

37 Interrupt Sources and Delegation

38 PLIC and AIA Context

39 Timer and Software Interrupts

40 Sv39 Virtual Memory

41 **Sv48 and Sv57 Virtual Memory**

42 Page-Table Entry Fields

43 ASIDs, TLBs, and SFENCE.VMA

44 PMP and Physical Memory Attributes

45 RVWMO Memory Model

46 FENCE, Acquire, and Release

47 Atomic Ordering and Synchronization

48 Cache-Block Operations

49 **Hypervisor Extension Overview**

50 Debug Mode and Triggers

51 Counters and Performance Monitoring

52 Feature Discovery and Profiles

53 RISC-V psABI

54 ELF, Relocations, and Pseudo-Instructions

55 Quick Reference Tables

56 Official English References

57 **Index, Errata, and Revision Record**

RISC-V Programmer-Visible Architecture Map

Editorial sample: architecture state, privilege, memory, and extension layers

ARCHITECTURAL STATE

x0-x31

PC

f0-f31

v0-v31

mstatus / sstatus

satp / hgatp

Trap CSRs

Debug and counter state

MODE AND PRIVILEGE

U · S · M · optional H extension

INSTRUCTION MODEL

R/I/S/B/U/J · compressed · modular extensions

MEMORY SYSTEM

Sv39/Sv48/Sv57 · ASID · PMP/PMA

EVENT DELIVERY

Traps · delegation · direct/vectored entry

MODERN COMPUTE

M/A/F/D/C/B/V · scalar/vector crypto

Every full-volume technical page maps back to one or more layers shown here.

Register ABI Map and Base Encoding Formats

High-density A3 portrait layout with vector tables and concise explanations

REGISTER / STATE MAP

x0/zero	x1/ra	x2/sp	x3/gp	x4/tp	x5/t0	x6/t1	x7/t2
x8/s0	x9/s1	x10/a0	x11/a1	x12/a2	x13/a3	x14/a4	x15/a5
x16/a6	x17/a7	x18/s2	x19/s3	x20/s4	x21/s5	x22/s6	x23/s7
x24/s8	x25/s9	x26/s10	x27/s11	x28/t3	x29/t4	x30/t5	x31/t6

ENCODING / FIELD MODEL

opcode	Bits 6:0	Base operation class
rd	Bits 11:7	Destination register
funct3	Bits 14:12	Minor operation selector
rs1	Bits 19:15	Source register 1
rs2 / imm	Bits 24:20	Source register or immediate fragment
funct7 / imm	Bits 31:25	Major selector or immediate fragment

Base instructions are 32 bits; the low-order opcode bits distinguish standard-length encoding classes.

Source: RISC-V Unprivileged ISA, base instruction formats

Official English References

The complete volume uses section-level source keys on every technical page

PRIMARY NORMATIVE SOURCES

- 01

The RISC-V Instruction Set Manual, Volume I: Unprivileged Architecture, Version 20260120.
- 02

The RISC-V Instruction Set Manual, Volume II: Privileged Architecture, Version 20260120.
- 03

RISC-V ABIs Specification.
- 04

RISC-V Debug Specification.
- 05

RISC-V Advanced Interrupt Architecture Specification.

REFERENCE-KEY EXAMPLE

RISCV-UNPRIV §2 · RISCV-PRIV §3 · RISCV-PSABI §1

Revision and access-date metadata are stored in the bibliography and frozen for each published edition. Corrections are tracked through a public errata register.

EDITORIAL COMMITMENT

- No unsourced register maps or bit positions.
- No mnemonic equivalence without behavior verification.
- Architecture revision shown for optional and recent features.
- Official manuals remain normative when ambiguity exists.

A reference built for the moment when a diagram must be faster than a manual search — without sacrificing traceability.

DESIGNED FOR

- Operating-system and kernel developers
- Compiler, assembler, linker, and JIT engineers
- Firmware and hypervisor developers
- Debugger, profiler, and reverse-engineering specialists
- Advanced low-level programmers and architecture educators

SERIES PRINCIPLE

One architecture per volume. One dedicated comparison volume. A unified visual grammar across registers, instruction encodings, privilege, virtual memory, exceptions, atomics, vectors, ABI, and binary interfaces.

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

THE PROCESSOR ARCHITECTURE ATLAS SERIES

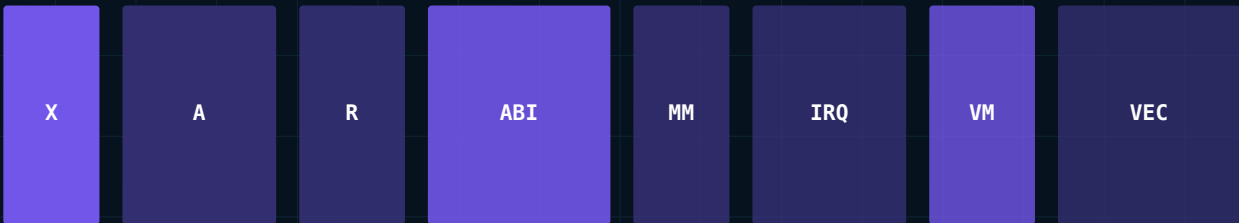
Independent technical reference · English edition · A3 portrait

ISBN / BARCODE AREA

The Cross-Architecture Systems Atlas

x86-64 · Arm AArch64 · RISC-V — Direct Comparisons for OS, Compiler, JIT, and Low-Level Development

X · A · R



A3 PORTRAIT · VISUAL SYSTEMS REFERENCE · FIRST EDITION

REFERENCE SCOPE

- **Instruction and register equivalence**
- ABI and binary interface comparison
- Memory models and atomics
- Porting patterns and backend design

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

48 interior pages · English edition · Official-reference baseline

Independent technical publication — no vendor affiliation implied.

The Cross-Architecture Systems Atlas

x86-64 · Arm AArch64 · RISC-V — Direct Comparisons for OS, Compiler, JIT, and Low-Level Development

X · A · R

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

EDITION

First Edition · 2026

FORMAT

A3 Portrait · 297 × 420 mm

SCOPE STATEMENT

This volume is a dense visual reference for software-visible Cross-Architecture architecture. It is designed for operating-system developers, compiler and JIT engineers, debugger authors, reverse engineers, firmware developers, and low-level programmers.

PUBLICATION NOTE

Official architecture specifications remain normative. This atlas compresses programming-visible concepts into traceable diagrams, tables, and cross-references; it does not replace formal instruction pseudocode or vendor-specific implementation manuals.

How to Read This Volume

A consistent editorial language for dense technical information

REG

ARCHITECTURE STATE

Registers, flags, system state, and privilege-visible control.

ENC

ENCODING FIELD

Instruction bits, prefixes, opcode maps, and immediate reconstruction.

MEM

MEMORY EFFECT

Address formation, ordering, cache/TLB effect, and translation behavior.

EXC

EXCEPTION PATH

Fault, trap, interrupt, privilege transition, and return mechanism.

STS

STATUS MARKER

Required, optional, legacy, deprecated, ratified, or implementation-defined.

SRC

SOURCE CODE

Compact official-reference key printed at the foot of each technical page.

STATUS LANGUAGE

REQUIRED

OPTIONAL

LEGACY

DEPRECATED

RATIFIED

IMPLEMENTATION-DEFINED

TABLE READING ORDER

1 Name and purpose

3 Architectural state affected

5 Exceptions, caveats, and source

2 Operands and width

4 Privilege and memory ordering

Detailed Contents I

48 interior pages · covers excluded

01 Half Title

02 Edition and Scope

03 How to Use This Atlas

04 Symbols and Comparison Rules

05 Contents I

06 Contents II

07 Architecture Philosophy at a Glance

08 Programmer-Visible Register Models

09 Status Flags and Condition State

10 Integer Data Types and Width Semantics

11 Instruction Length and Decode Strategy

12 Encoding Fields Compared

13 Addressing Modes Compared

14 Move, Load, and Store Equivalence

15 Arithmetic and Logical Equivalence

16 Shifts, Rotates, and Bit Operations

17 Compare and Conditional Selection

18 Branches, Calls, and Returns

19 Stack Models and Frame Construction

20 Calling Convention Matrix

21 System Call Entry and Return

22 Privilege Models Compared

23 Virtual Address Translation

24 Page-Table Structures and Page Sizes

Detailed Contents II

48 interior pages · covers excluded

25 Protection and Memory Attributes

26 Exceptions and Trap Classification

27 Interrupt Entry and Routing

28 Context-Switch State Matrix

29 Atomic Read-Modify-Write Operations

30 Memory-Ordering Models

31 Fences and Barriers

32 Cache and TLB Maintenance

33 Floating-Point State Compared

34 SIMD and Scalable Vector Models

35 Cryptographic Extensions

36 Feature Discovery

37 Debug and Performance Monitoring

38 ELF, PE/COFF, and Mach-O Context

39 Relocations and Position-Independent Code

40 Assembly Syntax and Operand Order

41 Compiler Backend Mapping

42 JIT Code Generation Patterns

43 Operating-System Porting Checklist

44 Virtualization Models

45 Security and Control-Flow Protection

46 Quick Translation Tables

47 Official English References

48 Index, Errata, and Revision Record

Cross-Architecture Programmer-Visible Architecture Map

Editorial sample: architecture state, privilege, memory, and extension layers

ARCHITECTURAL STATE

Integer registers

Program counter

Condition state

Privilege state

Translation state

Vector state

Exception state

ABI-visible state

REGISTER MODEL

Overlapping vs orthogonal vs ABI aliases

INSTRUCTION MODEL

Variable vs fixed vs modular encoding

MEMORY SYSTEM

Page-table geometry and ordering rules

EVENT DELIVERY

Vectors, causes, privilege entry, return

MODERN COMPUTE

SIMD, scalable vectors, matrices, crypto

Every full-volume technical page maps back to one or more layers shown here.

Direct Architecture Comparison Matrix

High-density A3 portrait layout with vector tables and concise explanations

Property	x86-64	Arm AArch64	RISC-V
Instruction length	1-15 bytes	32-bit A64	32-bit base; 16-bit C; longer encodings reserved
Integer GPRs	16 × 64-bit	31 × 64-bit + SP/ZR views	32 × XLEN; x0 fixed zero
Condition state	RFLAGS	NZCV + conditional select	Compare-and-branch; no global integer flags
Addressing	Base + index×scale + disp; RIP-relative	Base + immediate/register; pre/post-index; literal	Base + signed immediate; AUIPC sequences
Privilege	Rings 0-3	EL0-EL3	U/S/M (+ H)
Memory model	Strong / TSO-oriented	Weakly ordered	RVWMO; optional Ztso
Vector model	SSE/AVX/AVX-512 fixed widths	NEON fixed + SVE scalable	RVV scalable via VLEN/VL
Fast syscall	SYSCALL/SYSRET	SVC + exception return	ECALL + trap return

Editorial rule: equivalent behavior is compared by architectural effect, not by mnemonic similarity.

Sources: official architecture and ABI manuals

Official English References

The complete volume uses section-level source keys on every technical page

PRIMARY NORMATIVE SOURCES

- 01

Official Intel®, AMD, Arm®, and RISC-V architecture manuals.
- 02

System V AMD64 ABI, AAPCS64, and RISC-V psABI.
- 03

Microsoft PE/COFF and x64 ABI documentation.
- 04

DWARF Debugging Information Format, Version 5.
- 05

IEEE Standard for Floating-Point Arithmetic.

REFERENCE-KEY EXAMPLE

INTEL-SDM · ARM-ARM · RISCV-UNPRIV · ABI-SPECS

Revision and access-date metadata are stored in the bibliography and frozen for each published edition. Corrections are tracked through a public errata register.

EDITORIAL COMMITMENT

- No unsourced register maps or bit positions.
- No mnemonic equivalence without behavior verification.
- Architecture revision shown for optional and recent features.
- Official manuals remain normative when ambiguity exists.

A reference built for the moment when a diagram must be faster than a manual search — without sacrificing traceability.

DESIGNED FOR

- Operating-system and kernel developers
- Compiler, assembler, linker, and JIT engineers
- Firmware and hypervisor developers
- Debugger, profiler, and reverse-engineering specialists
- Advanced low-level programmers and architecture educators

SERIES PRINCIPLE

One architecture per volume. One dedicated comparison volume. A unified visual grammar across registers, instruction encodings, privilege, virtual memory, exceptions, atomics, vectors, ABI, and binary interfaces.

Prepared by Ayman Alheraki

Sponsored by <https://ForgeVM.org> Project

THE PROCESSOR ARCHITECTURE ATLAS SERIES

Independent technical reference · English edition · A3 portrait

ISBN / BARCODE AREA