ORIGIN ACADEMY · COURSE SYLLABUS

x86-64 Developer

Master the x86-64 platform from the inside: assembly, the System V ABI, paging, privilege rings, SIMD and virtualisation.

20 weeks · about 12 hours a week · live cohort of up to 10
Curriculum last updated 18 November 2026

Availability

This course opens soon. Join the waitlist on the course page to hear first about dates and prices.

Before you start

- C programming (our C Programming course)
- Comfort with the Linux command line
- English good enough to follow technical documentation and live sessions: about a credit (C6 or better) in WASSCE or NECO English Language, or an equivalent such as KCSE or IGCSE English (roughly CEFR level B1). You don't need the certificate: Phase 0 is free, so you can check for yourself
- A laptop running Linux, macOS, or Windows with WSL

How the course works

- A live cohort of up to 10 students, with a mentor.
- At least half your time is hands-on: every module ends with a lab.
- Live sessions are recorded, so you can rewatch them or catch up.
- Questions between sessions are answered within 24 hours on weekdays.
- A mentor reviews every phase project, like a pull request at work.

Phase 0 is free for everyone, before you enroll.

Curriculum

PHASE 0 · WEEKS 1–2 · FREE

x86-64 Architecture

Registers, operating modes, the instruction encoding and how CPUs execute it.

- Registers and operating modes
- Instruction encoding
- How CPUs execute instructions
- Reading the Intel and AMD manuals

You build: Disassembler for a subset of instructions

PHASE 1 · WEEKS 3–6

Assembly & the ABI

x86-64 assembly, the System V calling convention, stack frames and linking.

- x86-64 assembly basics
- The System V ABI and calling convention
- Stack frames and the red zone
- Linking assembly with C

You build: Assembly library linked into a C program

PHASE 2 · WEEKS 7–9

Memory & Paging

Segmentation history, 4-level paging, TLBs and memory-mapped I/O.

- Segmentation: history and what remains
- 4-level paging
- The TLB
- Memory-mapped I/O

You build: Page-table walker and visualiser

PHASE 3 · WEEKS 10–12

Privilege, Interrupts & Syscalls

Rings, the IDT, exceptions, APIC interrupts and the syscall path.

- Privilege rings
- The IDT and exceptions
- The APIC and interrupts
- The syscall path

You build: Trace a syscall from user space to the kernel

PHASE 4 · WEEKS 13–14

Debugging & Profiling

GDB, perf, hardware performance counters and reading compiler output.

- GDB at the instruction level
- perf and performance counters
- Reading compiler output

You build: Profile and fix a real performance bug

PHASE 5 · WEEKS 15–16

SIMD: SSE & AVX

Vector registers, intrinsics and auto-vectorisation.

- Vector registers
- SSE and AVX intrinsics
- Auto-vectorisation

You build: Vectorised image-processing kernel

PHASE 6 · WEEKS 17–18

Virtualisation Basics

VT-x, VMCS, EPT and how hypervisors like KVM work.

- VT-x and the VMCS
- EPT
- How KVM works

You build: Tiny hypervisor experiment with KVM

PHASE 7 · WEEKS 19–20

Capstone

Write a small x86-64 kernel that boots under QEMU.

- Booting under QEMU
- Paging and interrupts
- A tiny shell
- Demo and review

You build: Kernel with paging, interrupts and a shell

Your certificate

You earn it by doing two things:

- Attend every live session. If you miss one, make it up by watching the recording and submitting that week's lab within 7 days. You can do this up to 2 times per course.
- Score more than 50% on the final evaluation at the end of the course.

Your Enterprise Readiness Rating is separate: it scores your projects, capstone and assessments so employers can see how job-ready you are.

Questions

Email academy@originclouds.io, or see the course online: https://academy.originclouds.io/courses/x86-64-developer