```yaml
url: "https://seimei.do/en/blog/vmware-fusion-windows-on-arm-devenv/"
title: "Building a Windows on ARM Test Environment on a Mac: VMware Fusion + Windows 11 ARM64"
description: "The Japan Patent Office e-filing software is Windows-only, and the Mac build was
  ↪ discontinued — so there is no filing route from a Mac. I built a Windows 11 ARM64
  ↪ environment on an Apple Silicon Mac with VMware Fusion instead. Fusion has been free
  ↪ (commercial use included) since November 2024, and the Windows 11 ARM64 ISO is available
  ↪ from Microsoft directly. This covers the biggest setup hurdle, getting past the Microsoft
  ↪ account requirement, plus the settings that make it a driver test bench (disabling Secure
  ↪ Boot, USB passthrough, snapshots), on the actual setup in use (Fusion 26.0.0 / M3 Max)."
pubDate: 2026-08-10
tags: ["windows", "arm64", "vmware", "mac"]
series: ""
related: ["windows-on-arm-rcs380-driver"]
graphLabel: "Fusion WoA test env"
graphWeight: 1
lang: en
draft: false
ogImage: "/blog/vmware-fusion-windows-on-arm-devenv/og-en.jpg"
```

https://seimei.do/en/blog/vmware-fusion-windows-on-arm-devenv/

The reason I built this environment is to file patents myself. The Internet e-filing software for the Japan Patent Office is Windows-only, and the Mac build was discontinued in September 2021. In other words, there is no filing route from a Mac. All I have on hand is a Mac. So the answer is to bring up Windows in a virtual machine.

It is not only for filing. Windows-dependent administrative systems, such as a notary office' s electronic certified-date service (electronic notarization), run in this same environment.

You do not need a physical ARM64 Windows machine. With a single Apple Silicon Mac, a Windows on ARM environment fits in a virtual machine. This article is the build record. The actual setup on hand is below.

```
MacBook Pro (Apple M3 Max, 128 GB RAM)
  → VMware Fusion 26.0.0
  → Windows 11 Pro ARM64 (24H2, build 26100) guest
      2 vCPU / 4 GB RAM / ~41 GB disk in use / NAT / vTPM
  → USB passthrough → RC-S380 (real device)
```

The allocation is modest — 2 vCPUs and 4 GB of memory — but it is enough to install the MSVC ARM64 toolchain (about 12 GB) and build the driver. A test bench does not have to be lavish.

VMware Fusion is free now

Cost first. After the Broadcom acquisition, VMware Fusion went free for personal use in May 2024, and since November 2024 it has been free for commercial, educational, and personal use with no license key. There is no cost even for business use. Apple Silicon support has been an official feature since the 13.x line and continues in the current release.

Other options exist: Parallels Desktop (paid subscription) and UTM (free, QEMU-based). I use Fusion here because it is free for commercial use and has both snapshots and USB passthrough.

An aside: buying Parallels back in university

Mac-hosted Windows VMs have a personal history for me. The university-specified PC ran Windows, but I bought a Mac because I was a Sakanaction fan. To run both, I bought Parallels Desktop when I started university.

It caused no real trouble. If anything, it was the Office suite in the information-literacy class. The Windows and Mac versions differed slightly in design and features, and the slightly-less-capable Mac version sometimes could not follow the textbook step for step.

A vivid memory from my first year is downloading the Windows 8.1 ISO over a phone tether. I somehow wasted the one free Windows 10 license from the student benefit on an error, and used 8.1 instead, which could be re-downloaded any number of times.

Back then I also ended up buying a Windows desktop later, for VR. That turned out to be a good experience.

So pick whichever OS you like. Builders and gamers may be better off on Windows. For myself, though, if I had not chosen a Mac back then, I doubt I would have grown to like technology this much.

The Windows 11 ARM64 ISO is available officially

Getting ARM Windows used to mean relying on Insider Preview, but now Microsoft distributes the Windows 11 ARM64 ISO officially. I used `Windows11_26100.4349_Professional_ja-jp_arm64.iso` (24H2).

Drag the downloaded ISO onto Fusion' s new-VM dialog and pick "Windows 11 64-bit Arm" as the OS.

Windows 11 requires a TPM, and Fusion 13.5 and later configure a vTPM automatically. When VM creation asks for a disk encryption mode (fast/full), that is for the vTPM requirement, and fast (config-file-only encryption) is enough. My VM uses this too.

Removing the encryption or the vTPM device after installation stops Windows from booting, so leave them alone.

The biggest setup hurdle: getting past the Microsoft account

Installation is no different from x64 Windows, but the OOBE (initial setup) has a gate. Windows 11 all but forces sign-in with a Microsoft account. I want the test bench on a local account, because I do not want a personal account tied to a disposable machine that gets rolled back by snapshot and rebuilt.

The awkward part is that the workaround is a cat-and-mouse game with Microsoft. The old standby, the `BYPASSNRO` script, was announced for removal by Microsoft in March 2025 and closed off.

My first attempt used the other classic: "no network, no Microsoft account demanded" . I disconnected the VM' s network adapter in Fusion and walked through OOBE that way. OOBE did complete with a local account. Then things went to hell.

Just before reaching the desktop, a full-screen "Unlock your Microsoft experience" appeared, with no close button. Killing the offending app from Task Manager cleared the screen — and then the taskbar and desktop icons never came back.

Restarting explorer.exe did nothing, and rebooting the VM dropped it into the Windows Recovery Environment. The OOBE internal state had apparently been corrupted. I gave up on spot repairs and deleted the VM to rebuild it.

The rebuild is where `ms-cxh:localonly` came in. This time, leave the network connected and proceed normally all the way to the Microsoft account sign-in screen.

1. Advance to the "Let' s add your Microsoft account" screen
2. Open a command prompt with `Shift + F10`
3. Run `start ms-cxh:localonly`
4. The local-account creation dialog opens directly

This route reached the desktop without a single snag. The network-disconnect method walks OOBE down a path it does not expect and corrupts state easily; `ms-cxh:localonly` stays close to the intended flow and holds up — that is my understanding, arrived at after breaking one install.

Note that `ms-cxh:localonly` has reportedly been closed off in newer 25H2 builds as well. Treat the procedure here as having a shelf life. More durable than any single workaround is the habit of checking, before building a test machine, which workaround still works at that moment.

Three settings that make it a driver test bench

Settings that turn a plain Windows VM into a "driver test bench."

1. Disable Secure Boot in the VM settings

Running a homemade driver under test signing requires enabling testsigning, and that in turn requires disabling Secure Boot.

On real hardware this means entering firmware settings, plus the hesitation of lowering the defenses on a machine you use daily. On a VM it is just unchecking UEFI Secure Boot in Fusion's advanced settings. Precisely because it is a disposable test bench, you can lower it without hesitation.

```
Virtual Machine > Settings > Advanced > uncheck "Enable UEFI Secure Boot"
```

Then run `bcdedit /set testsigning on` inside the guest and reboot, and test-signed drivers will load.

2. Hand the real device to the guest via USB passthrough

A USB device plugged into the Mac can be switched to the guest from Fusion's menu or the connection dialog. Hand over the RC-S380 and the guest Windows sees it as the same USB device (`usb:054c:06c1`) it would if plugged in physically.

What matters in a driver-development context is that disconnect events are cheap to produce. I tested "automatic recovery from USB disconnect" by physically unplugging and replugging the device. Fusion's connection toggle should produce the same kind of disconnect, but I have not tried that route.

3. Take a snapshot before installing the driver

Installing a driver is an operation that dirties the OS state, and a failure can go as far as an unbootable machine. On a VM, taking one snapshot before installation means you can be back in tens of seconds no matter what happens. In many cases, rolling back and redoing beats agonizing over cleaning up packages registered with `pnputil`.

The test-signing certificate store, the testsigning flag, the registered driver package — when you do not want to disturb that combination, a snapshot of the "working state" is your insurance.

The limits of what the VM verified

The tests here were done on "a Windows 11 ARM64 VM plus USB passthrough." I have not verified on a physical Snapdragon Windows PC. The driver is an ARM64-native binary, so in principle it should behave the same, but I have not ruled out differences in real USB controllers or firmware. If anyone has tried it on real hardware, I would like to hear the results.

An operational quirk: the day the guest's network dies

Since putting this into use, I have hit the same recurring ailment several times. At some point the guest's network alone stops working, and VMware Tools stops responding too. Fusion's log shows this:

```
GuestRpc: app toolbox's second ping timeout; assuming app is down
```

It reads as though Tools died inside the guest. But tracing the log back by timestamp, a different message had been streaming for hours before that.

```
VNET: MACVNetPortVirtApiPrimaryIfaceChanged: Global state changed
VNET: MACVNetLinkStateEventHandler: 'ethernet0' state from 4 to 6.
VNET: MACVNetLinkStateTimerHandler: 'ethernet0' state from 6 to 1.
VMXNET3 hosted: Cannot retrieve the buffer descriptors per rx packet.   ← this line then
  ↪repeats endlessly
```

The origin is host-side. The moment the Mac's primary network interface switches, the guest's virtual NIC (VMXNET3) receive path stops meshing, and the same error line streams on. Tracing my own logs, the busiest generation had this one line over ten thousand times. The GuestRpc timeout is just a downstream symptom; Tools is the victim here, while the real culprit is elsewhere.

I recover by restarting the VM (the log, too, settles once after a restart), but I live with it as a recurring ailment that can return whenever the host network changes.

Later I got to measure the shutdown side of this ailment too. During one recurrence, I had to bring down a VM that had been running for two straight days, burning nearly 190% CPU while streaming over eight thousand copies of that error line.

When I tried, every graceful shutdown path was dead. `vmrun stop <vmx> soft` bounces with `VIX_E_TOOLS_NOT_RUNNING`. Driving `shutdown.exe` inside the guest via `runProgramInGuest` goes through Tools as well, so it never had a chance.

Fusion's menu "Shut Down" (which for this VM's settings amounts to an ACPI power button, a path that should not need Tools) got no response either. In the end only `vmrun stop <vmx> hard` brought it down.

So this ailment, by taking Tools down with it, strips away your graceful shutdown options one by one from the outside. The two Tools-based paths are dead on arrival, and even ACPI can fail under a burning CPU.

A hard stop equals pulling the plug. But the certificate store and the testsigning flag had long been committed to disk, and if only the network receive path is burning, nothing is mid-write to disk either. That was the reasoning when I pulled it.

I have repeated this hard stop many times by now, and the next boot has come up fine every time. It is still the same act as pulling the plug, so I would not recommend it — but for this particular ailment it has cost me nothing so far.

The lesson has the same shape as the landmines in the driver work: the loud symptom (GuestRpc timeout) and the real cause (host network change → virtual NIC) sit in different places. With logs, counting how many thousands of times one line appears is faster than poring over the loud one first.

What did I build on top of this environment? The story of getting a USB device — one whose vendor ships only an x64 driver — working on Windows on ARM is in the next article.