```
url: "https://seimei.do/ja/blog/vmware-fusion-windows-on-arm-devenv/"
title: "Mac の上に Windows on ARM 検証環境を作る — VMware Fusion + Windows 11 ARM64"
description: "特許の本人出願に使う特許庁のインターネット出願ソフトは Windows 専用で、Mac 版は提供終了済
  ↪ み。Mac からの出願ルートはないので、Apple Silicon Mac の上に VMware Fusion で Windows 11 ARM64 環
  ↪ 境を作った。Fusion は 2024 年 11 月から商用含め無償、Windows 11 ARM64 の ISO も Microsoft 公式か
  ↪ ら入手できる。セットアップ最大の関門である Microsoft アカウント回避、ドライバー開発の検証環境として効く
  ↪ 設定（Secure Boot 無効化・USB パススルー・スナップショット）まで、実際に使っている構成（Fusion 26.0.0
  ↪ / M3 Max）を記録する。"
pubDate: 2026-08-10
tags: ["windows", "arm64", "vmware", "mac"]
series: ""
related: ["windows-on-arm-rcs380-driver"]
graphLabel: "Fusion WoA 検証環境"
graphWeight: 1
lang: ja
draft: false
ogImage: "/blog/vmware-fusion-windows-on-arm-devenv/og.jpg"
```

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

この環境を作った動機は、特許の本人出願である。特許庁への電子出願に使うインターネット出願ソフトは Windows 専用で、Mac 版は 2021 年 9 月に提供終了している。つまり、Mac からの出願ルートは存在しない。手元の機材は Mac だけ。それなら Windows を仮想マシンで用意すればいい。

出願専用というわけでもない。公証役場の電子確定日付（電子公証）のような Windows 前提の行政系システムにも、この環境はそのまま使える。

ARM64 版 Windows の実機を持っていなくても、Apple Silicon Mac が 1 台あれば、Windows on ARM の環境は仮想マシンで用意できる。この記事はその構築記録である。手元の実構成は次のとおり。

```
MacBook Pro（Apple M3 Max, RAM 128 GB）
  → VMware Fusion 26.0.0
  → Windows 11 Pro ARM64（24H2, build 26100）ゲスト
      vCPU 2 / RAM 4 GB / ディスク実使用 約 41 GB / NAT / vTPM
  → USB パススルー → RC-S380（実デバイス）
```

ゲストへの割り当ては vCPU 2 個・メモリ 4 GB と控えめだが、MSVC の ARM64 ツールチェーン (約 12 GB) を入れてドライバーをビルドする用途で足りている。検証機は贅沢でなくてよい。

VMware Fusion はいまや無償である

まず費用の話から。VMware Fusion は Broadcom 買収後の 2024 年 5 月に個人利用が無償化され、2024 年 11 月からは商用・教育・個人を問わずライセンスキー不要の無償になった。業務利用でも費用はかからない。Apple Silicon 対応も 13.x 系からの正式機能で、現行版まで継続している。

仮想化ソフトの選択肢としては Parallels Desktop(有償サブスクリプション) や UTM(無償・QEMU ベース)

もある。本記事の構成で Fusion を使うのは、無償で商用利用でき、スナップショットと USB パススルーが揃っているためである。

余談: Parallels を買った大学時代の話

Mac 上の Windows VM には個人的な因縁がある。大学指定パソコンの OS は Windows だったが、サカナクションのファンだったので Mac を買った。併用のため大学入学時に買ったのが Parallels Desktop である。

それで特に困ることはなかった。強いて言えば、情報の授業の Office スイートくらいか。Windows 版と Mac 版はデザインや機能が微妙に異なり、微妙に下位互換の Mac 版は教科書どおりに進められないことがあった。

大学 1 年の頃、テザリングで Windows 8.1 の ISO をダウンロードしたのは強烈な記憶である。学生特典の無料ライセンス 1 本 (Windows 10) をなぜかエラーで無駄にし、ダウンロードし直しの利く 8.1 を使った。

学生の頃は VR で遊ぶために結局あとから Windows のデスクトップも買った。それはいい経験だった。

要するに、好きな OS を選べばええ、という話である。自作 er やゲーマーなら Windows のほうがいいかもしれない。ただ自分については、あのとき Mac を選ばなかったら、ここまで技術が好きにならなかった気もする。

Windows 11 ARM64 の ISO は公式から取れる

かつて ARM 版 Windows の入手は Insider Preview 頼みだったが、現在は Microsoft が Windows 11 ARM64 の ISO を公式配布している。今回使ったのは `Windows11_26100.4349_Professional_ja-jp_arm64.iso`(24H2) である。

ダウンロードした ISO を Fusion の新規 VM 作成ダイアログにドラッグし、OS として「Windows 11 64-bit Arm」を選べばよい。

Windows 11 のインストール要件である TPM は、Fusion 13.5 以降なら vTPM が自動で構成される。VM 作成時に仮想ディスクの暗号化方式 (高速/完全) を聞かれるのは vTPM の要件のためで、高速 (構成ファイルのみ暗号化) で足りる。今回の VM もこの構成である。

インストール後に暗号化や vTPM デバイスを外すと Windows が起動しなくなるので、ここは触らないこと。

セットアップ最大の関門: Microsoft アカウントの回避

インストール自体は x64 版 Windows と変わらないが、OOBE(初期セットアップ) には関門がある。Windows 11 は Microsoft アカウントでのサインインを事実上強制してくる。検証機はローカルアカウントで作りたい。スナップショットで巻き戻し、壊しては作り直す使い捨ての機体に、個人アカウントを紐付けたくないからである。

やっかいなのは、回避手段が Microsoft とのいたちごっこになっていることだ。定番だった `BYPASSNRO` スクリプトは 2025 年 3 月に Microsoft が削除を発表して塞がれた。

手元では最初、もうひとつの定番である「ネットワークに繋がなければ Microsoft アカウントを要求されない」方式を使った。Fusion で VM のネットワークアダプタを切断したまま OOBE を進める手だ。ローカルアカウントで OOBE は完了した。だが、そこからが地獄だった。

デスクトップ到達の直前に「Microsoft エクスペリエンスのロックを解除する」という全画面が現れ、閉じるボタンがない。タスクマネージャーから該当アプリを終了させると全画面は消えたが、今度はタスクバーとデスクトップアイコンが出ない。

explorer.exe の再起動でも直らず、VM を再起動したら Windows 回復環境に落ちた。OOBE の内部状態が壊れていたらしい。個別修復を諦めて、VM ごと削除して作り直した。

作り直しで使ったのが `ms-cxh:localonly` 方式である。今度はネットワークを切らず、Microsoft アカウントのサインイン画面まで通常どおり進む。

1. 「Microsoft アカウントを追加しましょう」の画面まで進む
2. `Shift + F10` でコマンドプロンプトを開く
3. `start ms-cxh:localonly` を実行する
4. ローカルアカウントの作成ダイアログが直接開く

こちらは何の引っかかりもなくデスクトップまで到達した。ネットワーク切断方式は OOBE の想定外の経路を通るぶん内部状態を壊しやすく、`ms-cxh:localonly` は想定内フローに近いぶん安定している——というのが一度壊した後の理解である。

なお 25H2 の新しいビルドではこの `ms-cxh:localonly` も塞がれたという報告が出ている。この記事の手順も賞味期限付きと思ったほうがよい。個別の回避手段より、「検証機を作る前に、その時点で通る回避手段を確認する」という習慣のほうが長持ちする知見である。

ドライバー開発の検証環境として効く設定 3 つ

素の Windows VM を「ドライバー検証機」にするための設定である。

1. Secure Boot を VM 設定で無効化する

自作ドライバーをテスト署名で動かすには testsigning を有効にする必要があり、その前提として Secure Boot を無効化する。

実機ではファームウェア設定に入る手間と「日常使いの機体の防御を落とす」ためらいが伴うが、VM なら Fusion の詳細設定で UEFI Secure Boot のチェックを外すだけで済む。使い捨てられる検証機だからこそ、ためらいなく落とせる。

仮想マシン > 設定 > 詳細 >「UEFI セキュア ブートを有効にする」を外す

そのうえでゲスト内で `bcdedit /set testsigning on` を実行して再起動すれば、テスト署名ドライバーがロードできる状態になる。

2. USB **パススルーで実デバイスをゲストに渡す**

Mac に挿した USB デバイスは、Fusion のメニューまたは接続ダイアログからゲスト側へ切り替えられる。RC-S380 を渡すと、ゲストの Windows からは物理マシンに直接挿したのと同じ USB デバイス (`usb:054c:06c1`) として見える。

ドライバー開発の文脈で重要なのは、切断イベントを気軽に起こせることである。「USB 切断からの自動復帰」のテストを、手元では物理的な挿抜で行った。Fusion の接続切り替えでも同種の切断は起こせるはずだが、そちらは試していない。

3. **スナップショットを撮ってからドライバーを入れる**

ドライバーのインストールは OS の状態を汚す操作で、失敗すれば起動不能まであり得る。VM ならインストール前にスナップショットを 1 枚撮っておくだけで、何が起きても数十秒で戻れる。`pnputil` で登録したパッケージの掃除に悩むより、戻してやり直すほうが早い。

テスト署名の証明書ストア、testsigning フラグ、ドライバーパッケージの登録状態。この 3 つの組み合わせを崩したくないとき、「動いている状態」のスナップショットが保険になる。

運用トラブル: ゲストのネットワークだけが死ぬ日

運用を始めてから、何度か同じ持病を踏んでいる。ある瞬間からゲストのネットワークだけが通らなくなり、VMware Tools の応答も途絶える。Fusion のログにはこう出る。

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

一見「ゲスト側で Tools が死んだ」ように見える。ところがログを時刻で遡ると、その何時間も前から別のメッセージが流れ続けていた。

```
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.   ← 以後この行が延々と続く
```

起点はホスト側の事象である。Mac の優先ネットワークインターフェースが切り替わった瞬間にゲストの仮想 NIC(VMXNET3) の受信処理が噛み合わなくなり、同じエラー行が流れ続ける。手元のログを遡ると、多い世代では同一行が 1 万行を超えていた。GuestRpc のタイムアウトはその下流の症状にすぎず、Tools は被害者であって犯人ではない。

手元では VM を再起動して戻している (ログ上も再起動で一旦収まっている) が、ホストのネットワークが変われば再発しうる持病として付き合っている。

その後、この持病の「落とす側」も実測することになった。ある再発時、エラー行を 8 千行以上流しながら

CPU を 190% 近く焼いたまま丸 2 日稼働していた VM を落とすことになった。

やってみると、正常シャットダウンの経路が全滅している。`vmrun stop <vmx> soft` は `VIX_E_TOOLS_NOT_RUNNING` で弾かれる。`runProgramInGuest` でゲスト内の `shutdown.exe` を叩く手も、同じく Tools 経由なので最初から通らない。

Fusion メニューの「シャットダウン」(この VM の設定では ACPI 電源ボタン相当で、Tools 不要のはずの経路) も反応しないまま。結局 `vmrun stop <vmx> hard` でようやく落とせた。

つまりこの持病は、Tools を道連れにすることで正常シャットダウンの選択肢を外側から一つずつ奪っていく。Tools 経由の 2 経路は即死、ACPI も CPU 焼きの下では通らないことがある。

hard は電源断と同じだが、証明書ストアや testsigning の設定はとっくにディスクに確定済みで、ネットワーク受信経路が焼けているだけならディスクへの書き込み中でもない。そう見立てて引いた。

この hard 停止は手元で何度も繰り返しているが、次の起動はいずれも問題なく通っている。電源断と同じ操作である以上すすめはしないものの、この持病に限っていえば実害は出ていない。

教訓はドライバー開発で踏んだ地雷群と同じ構図で、目立つ症状 (GuestRpc timeout) と真因 (ホスト網の変化 → 仮想 NIC) が別の場所にある。ログは目立つ 1 行を精読するより先に、同じ行が何千回出ているかを数えるほうが早い。

VM で検証したことの限界も書いておく

この構成でのテストは「Windows 11 ARM64 の VM + USB パススルー」で行っている。Snapdragon 搭載の実機 Windows PC では検証していない。ARM64 ネイティブバイナリなので原理上は同じに動くはずだが、実機の USB コントローラーやファームウェアの差までは潰せていない。実機で試した方がいれば結果を聞きたい。

この環境の上で何を作ったのか。ベンダーが x64 版ドライバーしか出していない USB デバイスを Windows on ARM で動かした話は、次の記事で書いた。