binho

アプリケーションノート

Binho Supernova による CMIS ファームウェア更新時間の測定

AN0011

文書情報

キーワード	AN0011, Supernova, Pulsar, CMIS, OIF, CDB, ファームウェア更新, LPL, EPL, tWRITE, tCDBF, tBPC, Nmax, 光モジュール, ホールドオフ, I2CMCI, I3C, SPIMCI, ブロックあたりのトランザクション数
概要	光モジュールのファームウェア更新には時間がかかり、すぐに思いつく対策はより高速な管理バスです。 CMIS のアクセス制限と Table 10-4 のタイミングパラメータを順に計算していくと、時間はワイヤ上で 費やされているのではないことがわかります。1 MiB のイメージでは、バスクロックが占めるのは全体の ごく一部であり、大部分はブロックごとのホールドオフ（hold-off）が占めます。本アプリケーションノートは 仕様から時間の見積もりを導出し、どの数値が算術でありどの数値が実測を要するかを明示し、実際の モジュールのアクセスフェーズを仕様上の上限と比較して計時するコマンドを提供します。これにより、 ホストは自分のモジュールが実際にどのホールドオフを取るのかを確認できます。

Binho Inc.
42 Dore Street, Unit A, San Francisco, CA 94103

改訂 1.0
6 NOVEMBER 2026

> **検証範囲**。セクション 4 の見積もりは仕様から導出した算術であり、仕様と本シリーズが 扱う 3 つの管理インターフェースすべて、すなわち I2CMCI、SPIMCI、および Binho による導出としての I3C を 含むようになりました。セクション 5 の `timing` の出力は、光モジュールではなくファームウェアで実装した ターゲットに対する実際の実行結果であり、ターゲットではなくホスト経路を測定していることが判明しています。 セクション 5.2 では 2 つのホストトランスポートで 2 つのアダプタファームウェアリビジョンを測定し、 どちらもこの下限を動かさないことを確認しています。セクション 5.3 と 5.4 ではその下限をアダプタの USB ディスクリプタまでたどり、ホストがそのうちどれだけを隠せるかを示します。セクション 7 では未完了の 事項を述べます。

1 はじめに

I3C のシングルデータレート（single data rate）は、ワイヤ上で I2C の Fast-mode より約 31 倍高速です。 モジュールのファームウェア更新が遅いことに直面したホストエンジニアが、より高速な管理バスによって更新が 約 31 倍速くなると期待するのはもっともなことです。

CMIS のアクセス制限に沿って算術を進めると別の答えが得られ、その違いは実際の対策につながります。 どのバスを選ぶかよりも、ペイロードをどの Common Data Block（共通データブロック）コマンドで運ぶかの 選択のほうが重要です。

本ノートは見積もりを導出し、特定のモジュールが仕様上許容される場所ではなく、実際にどこで時間を 費やしているかを確認する方法を提供します。

> **本ドキュメントは実測スループットを示しません。** その理由と、それを示すために何を実行する必要が あるかはセクション 7 で説明します。

1.1 適用範囲

本ドキュメントは CMIS ファームウェアダウンロードの算術、すなわちアクセス長の制限、2 つの Write Firmware Block コマンドのブロックサイズ、および Table 10-4 のホールドオフパラメータを扱います。 モジュールの読み出し、ページ切り替え、Common Data Block アクセスの各フェーズを、それらの上限と 比較して計時するコマンドを提供します。

本ドキュメントはモジュールにファームウェアを書き込み**ません**。また、提供するユーティリティにも それを行うコマンドはありません。ファームウェア更新ステートマシン、イメージ署名、Run と Commit の シーケンスについては、そのタイミングに触れる以上のことは扱いません。レジスタモデルは AN0008 で 扱っており、本ドキュメントの前提となります。

1.2 対象デバイス

表 1. 対象デバイス

項目	役割	根拠
Binho Supernova	ホストアダプタ	検証済み、セクション 7
Binho Pulsar	ホストアダプタ	未検証
CMIS 管理対象の任意のモジュール	ターゲット	**未検証。** 測定にはターゲットモードのアダプタを使用しました

> **セクション 4 の見積もりは算術であり、測定ではありません。** OIF-CMIS-05.3 のセクション 5.2.2 と 第 9 章、および Table 10-4 から導出したもので、`python cmis_mci.py budget` で再現できます。 どの行もモジュールに対して確認されていません。

1.3 関連資料

表 2. 関連資料

資料	内容
OIF-CMIS-05.3	セクション 5.2.2.1 と 5.2.2.2、CDB については第 9 章、タイミングについては Table 10-4
AN0008	Binho Supernova による I2C 経由の CMIS 光モジュールの立ち上げ
AN0009	Binho Supernova による I3C 経由の CMIS 光モジュールの管理
AN0010	Binho Pulsar による SPI 経由の CMIS 光モジュールの立ち上げ

2 必要な機器とソフトウェア

表 3. 必要な機器

項目
Binho Supernova または Binho Pulsar USB ホストアダプタ（`timing` コマンド用）
CMIS 管理対象の光モジュール（`timing` コマンド用）
`budget` コマンドには何も必要ありません

```
pip install binhosupernova
python cmis_mci.py budget
```

> **ホストアダプタは 2 つのトランスポートを提供でき、どちらが使われるかはホストソフトウェアによって、 通知なしに決まります。** Binho アダプタファームウェア 4.5.0 以降は、従来の USB HID 経路に加えて ベンダー固有のバルクエンドポイントでも転送を行います。バルクはオプトインです。ホストがそれを ネゴシエートしないアダプタは HID 経路を従来どおり提供し、エラーも警告も出しません。これにより、 このファームウェアは互換性を壊さないリリースになっています。本ノートとシリーズの他のノートで使用する `binhosupernova` は HID のみであり、1.5.0 より前の CosmicSDK も同様です。インストールしたリリースから 推測するのではなく、オープンしたデバイスにトランスポートが利用可能かどうかを問い合わせてください。 それをサポートしない SDK は古い数値を出し、その間も何の異常も報告しないためです。セクション 5.2 では 両方を測定しており、この違いは数値よりも結論にとって重要です。

3 見積もりの根拠となる 4 つの事実

3.1 Local Payload 経由のファームウェアブロックは 116 バイト

Common Data Block は Page 9Fh を占めます。その Local Payload（ローカルペイロード）は 120 バイトで、 そのうち `Write Firmware Block LPL`（コマンド `0103h`）がブロックオフセットに 4 バイトを使います。 したがって、ブロックあたりのファームウェアは **116 バイト** です。

Extended Payload（拡張ペイロード）はページ A0h から AFh までの 128 バイトのページ 16 枚で、 `Write Firmware Block EPL`（コマンド `0104h`）によりブロックあたり **2048 バイト** になります。

3.2 WRITE が運べるのは最大 8 バイト

セクション 5.2.2.2 は次のように直接述べています。"A successful WRITE writes a sequence of up to eight given byte values"（成功した WRITE は、与えられた最大 8 個のバイト値の列を書き込む）。 `FullPageReadSupported` は**読み出し**の `Nmax` を 128 に引き上げますが、書き込みには何の影響もありません。

唯一の例外は直後の注記です。"A WRITE with more than 8 bytes may be allowed when specified explicitly in chapter 7.8"（第 7.8 章で明示的に規定されている場合、8 バイトを 超える WRITE が許可されることがある）。これは Common Data Block の Extended Payload 領域のことです。

したがって、116 バイトの Local Payload ブロックは `ceil(116 / 8)` = **15 回の個別の書き込み トランザクション** になり、それぞれが独自のアドレスフェーズを持ちます。

3.3 その後、モジュールは応答を止める

表 4. ホールドオフパラメータ、CMIS Table 10-4 の最大値

パラメータ	最大値	適用されるタイミング
tREAD	500 マイクロ秒	レジスタ読み出しの後
tWRITE	10 ms	揮発性メモリへの書き込みの後
tWRITENV	80 ms	不揮発性メモリへの書き込みの後
tBPC	10 ms	バンクまたはページの切り替えの後
tCDBF	4960 ms	フォアグラウンド CDB コマンドの後。デフォルトは 80 ms
tMgmtInit	2000 ms	リセットの後、ModuleLowPwr に到達するまで

これらは最大値です。モジュールはこれ以下の任意の値を通知でき、セクション 5 は特定のモジュールが実際に どれだけかかるかを確認するためのものです。

3.4 CMIS メモリのほぼすべては揮発性

仕様がそう明言しているため、**10 ms の tWRITE が一般的なケース** であり、楽観的なケースではありません。

ホールドオフは、トランザクションごとではなく**ブロックごとに 1 回** 発生します。これが唯一のレバーです。 1 MiB のイメージで、2048 バイトのブロックでは 512 回、116 バイトのブロックでは 9040 回発生します。

4 見積もり

```
python cmis_mci.py budget --image-kib 1024
```

```
Firmware update budget, 1024 KiB image, tWRITE 10 ms per block
  path                            blocks   per block       bus    hold-off     total
  I2CMCI 400 kHz, LPL 0103h         9040       15 txn   36.61 s     90.40 s   127.01
s
  I3C SDR 12.5 Mb/s, LPL 0103h      9040       15 txn    1.17 s     90.40 s    91.57
s
  SPIMCI 1 MHz, LPL 0103h           9040       15 txn   15.19 s     90.40 s   105.59
s
  I2CMCI 400 kHz, EPL 0104h          512       16 txn   24.33 s      5.12 s    29.45
s
  I3C SDR 12.5 Mb/s, EPL 0104h       512       16 txn    0.78 s      5.12 s     5.90
s
  SPIMCI 1 MHz, EPL 0104h            512        1 txn    8.41 s      5.12 s    13.53
s
  SPIMCI 50 MHz, EPL 0104h           512        1 txn    0.17 s      5.12 s     5.29
s

  Bus clock alone, 400 kHz to 12.5 Mb/s, about 31 times the wire speed: 1.39 times
faster.
  Payload size alone, LPL to EPL at 400 kHz: 4.3 times faster.
  Both together: 21.5 times faster.
  SPIMCI at 50 MHz, 125 times the I2C wire speed and one transaction per block: 24.0
times faster.
```

最初の 2 行を比べてください。ワイヤ速度を 31 倍にしても削減されるのは 35 秒です。元の低速クロックのまま、 ペイロードを運ぶ Common Data Block コマンドを変えると 97 秒削減されます。

4.1 3 つの管理インターフェース、すべてから同じ答え

SPIMCI の行は、本シリーズ全体が行おうとしている比較を担っています。SPIMCI は CMIS が実際に規定している 唯一のもう 1 つの管理インターフェースだからです。Appendix B.3.7.1 はすべてのトランザクションを、 4 バイトの制御ワード、`N` バイトのフロー制御バイト、最大 **2048 バイト** のペイロードとしてフレーミングし、 `00h:27.3-0` の `MciSpeedConfiguration` は 1 MHz から **50 MHz** までの範囲です。これは I2C クロックの 125 倍であり、ページ型インターフェースが行える最大アクセスの 16 倍です。

これにより、I2CMCI の 21.5 倍に対して 24.0 倍の高速化が得られます。そこから切り分けておくべき事実が 2 つあります。

モジュールが実際に起動する速度では、SPIMCI はワイヤ上で I3C より遅くなります。 MgmtInit を抜けた時点の `MciSpeedConfiguration` の初期値は常に 1 MHz であり、1 MHz では Extended Payload 経路は ワイヤ上で 8.41 s を費やします。I3C では 0.78 s です。クロックは、モジュールの通知を読み取ってその フィールドに書き込むホストが、意図的に引き上げる必要があります。

どの速度でも SPIMCI が変えるのはトランザクション数です。 Extended Payload ブロックは、どちらも 1 回のアクセスを 1 ページに制限する I2CMCI と I3C では 16 トランザクション、SPIMCI では

1 トランザクション です。この列が帯域幅の数値に現れることはありませんが、セクション 5 で示すように、ホストが体感するのは この列です。各トランザクションはホストを経由するラウンドトリップのコストを伴い、ベンチ上では、バスが 何をしていてもそのラウンドトリップは約 2 ミリ秒です。

つまり、3 つのインターフェースはワイヤ上では 125 倍、合計では 24 倍異なりますが、時間がどこで費やされるかに ついては完全に一致します。ファームウェア更新は帯域幅の問題ではありません。

表 5. 見積もりモデルに含まれるものと含まれないもの

含まれるもの	含まれないもの
ブロックあたりのトランザクション数（アクセス長の制限から算出）	ホストの USB レイテンシ（バスの特性ではない）
トランザクションあたりのバス時間（4 バイトのフレーミングとペイロード、各 9 ビットとして算出）	tCDBF（モジュールとコマンドにより異なる）
ブロックごとに 1 回の tWRITE	Run Image と Commit Image のステップ
	アクセスが拒否された後のリトライ

> **バス時間の列はワイヤ占有の推定値であり、実測スループットではありません。** これは意図的に大まかな ものです。その唯一の役割は、ホールドオフの列と比べて小さいことを示すことだからです。より精密なモデルに しても結論は変わらず、実測値はバスと同程度にホストの USB 経路を特徴づけることになります。

5 実際のモジュールの測定

```
python cmis_mci.py timing
```

このコマンドはファームウェアを書き込みません。更新を構成するアクセスフェーズ、すなわち 1 バイトの読み出し、 書き込み制限と同じ 8 バイトの読み出し、モジュールが通知している場合はフルページの読み出し、双方向の ページ切り替え、Common Data Block ページへの到達を計時し、それぞれを Table 10-4 の上限と並べて出力します。

```
$ python cmis_mci.py timing
phase                                  median        max    CMIS max
single byte read                     2.058 ms    3.400 ms      0.5 ms
8 byte read, the WRITE limit         1.995 ms    2.064 ms           -
page change 00h to 01h to 00h        7.988 ms    9.019 ms     20.0 ms
select CDB page 9Fh                  4.043 ms    5.913 ms     10.0 ms

One 116 byte LPL firmware block is 15 write accesses, about 29.9 ms of bus time,
against a tWRITE hold-off of up to 10 ms that the module pays once per block.
```

これはターゲットではなく、ホストの測定として読んでください。 どの行も USB ラウンドトリップ 1 回あたり 約 2 ミリ秒です。1 回のアクセスで約 2 ms、CDB ページ選択の 2 回のレジスタ書き込みで約 4 ms、双方向の ページ切り替えの 4 回の書き込みで約 8 ms です。1 バイトの読み出しは tREAD の 4 倍であり、8 バイトの 読み出しは 1 バイトの読み出しより遅くありません。これは、計時しているのがワイヤ上のバイトでない場合に しか起こり得ません。

この実行のターゲットは I2C ターゲットモードのアダプタで、即座に応答し、バスをホールドオフすることは ありません。したがって、これらの数値が示しているのは、モジュールが何も寄与する前にホスト経路が課す 下限であり、この構成ではその下限はアクセスあたり約 2 ms です。結論はセクション 7.2 で述べます。

結果で注目すべき点は次のとおりです。

中央値が仕様の最大値を大きく下回る 場合、モジュールは許容されているよりも高速であり、書き込みのたびに 最大値だけ待つホストは、更新時間の大部分を無駄にしています。これは、この測定から得られる最も価値のある 情報です。

ページ切り替えの最大値が仕様の上限に近い 場合、モジュールは実際にときどき tBPC を必要としています。 中央値を前提にしたホストは間欠的に失敗し、これはデバッグが最も難しい故障モードです。

1 バイトの読み出しが tREAD を超える 場合、それはモジュールのホールドオフではなく、ホスト側または アダプタ側のオーバーヘッドです。tREAD は I2C のクロックストレッチを抽象化したものであり、クロック ストレッチはバスキャプチャ上では見えますが、ホスト側のタイマーではほかの要因と区別できません。

5.1 I3C CMIS ターゲットに対する同じ測定

上記の実行は、ターゲットモードのアダプタに対する I2CMCI 経由のものです。フラットなメモリではなく レジスタモデルを実装した AN0009 の CMIS ターゲットファームウェアに対して I3C 経由で繰り返しても、 重要なものは何も変わりません。

```
$ python cmis_mci.py timing --transport i3c --address 0x09 --push-pull
PUSH_PULL_2_5_MHZ_25_DC
phase                                  median        max    CMIS max
single byte read                     2.059 ms    3.008 ms      0.5 ms
8 byte read, the WRITE limit         2.062 ms    2.760 ms           -
page change 00h to 01h to 00h        7.997 ms    9.170 ms     20.0 ms
select CDB page 9Fh                  3.998 ms    4.120 ms     10.0 ms

One 116 byte LPL firmware block is 15 write accesses, about 30.9 ms of bus time,
against a tWRITE hold-off of up to 10 ms that the module pays once per block.
```

31 倍高速なバスと、フラットな領域ではなくファームウェアから応答するターゲットを使っても、どの行も 元の位置のままです。これは見積もりが予測するのと同じ結果に、反対方向から到達したものです。

5.2 下限はホストスタックに属し、このスタックではそこから抜け出せない

Supernova ファームウェア 4.5.0 はアダプタの転送をベンダー固有のバルクエンドポイントに移し、転送の上限を 引き上げ、USB 経路を作り直しています。本ノートが提供するユーティリティで測定すると、何も変わりません。

表 6. 2 つのアダプタファームウェアリビジョンでの同じ 4 フェーズ、3 回の実行の中央値

フェーズ	4.4.0	4.5.0
1 バイトの読み出し	2.006 ms	2.036 ms
8 バイトの読み出し（WRITE の制限）	2.003 ms	2.036 ms
ページ切り替え 00h から 01h、そして 00h へ	7.993 ms	8.061 ms
CDB ページ 9Fh の選択	3.987 ms	4.035 ms

ボードは 4.4.0 から 4.5.0 へ、再び 4.4.0 へ戻し、もう一度アップデートし、それぞれの状態で測定しました。 再書き込みをまたいだ 1 回だけの比較では、ファームウェアの変更とほかに変化した要因とを区別できない ためです。

上の両方とも互換経路です。 新しいトランスポートはホスト側でのオプトインです。ホストがそれをネゴシエートしないアダプタは古い経路を従来どおり提供し、これによりこのファームウェアは互換性を壊さない リリースになっています。本ノートとシリーズ全体が動作している SDK である `binhosupernova` 4.2.0 は USB HID で転送し、ベンダー固有のバルクエンドポイントのコードはまったく含んでいません。したがって、 この組は互換経路が退行していないことを示しますが、それ単独では 4.5.0 が追加したトランスポートについて 何も示しません。

それをネゴシエートするホストスタックで同じ 4 フェーズを繰り返すと、表では答えられない問いに答えられます。 フェーズは変わりません。

表 7. 互換トランスポートとバルクトランスポートでの同じ 4 フェーズ、いずれもファームウェア 4.5.0

フェーズ	USB HID	バルクトランスポート
1 バイトの読み出し	2.036 ms	2.009 ms
8 バイトの読み出し（WRITE の制限）	2.036 ms	2.006 ms
ページ切り替え 00h から 01h、そして 00h へ	8.061 ms	8.019 ms
CDB ページ 9Fh の選択	4.035 ms	3.993 ms

右側の列は、トランスポートが自身を利用可能と報告し、32768 バイトの上限を通知している状態で取得したもので あり、名前を変えただけの 2 回目の互換経路の実行ではありません。このトランスポートは CosmicSDK 1.5.0 で 導入されました。古い SDK は何の異常も報告せずに左側の列の結果を出します。これこそ、この測定が前提とする のではなく排除しなければならなかった失敗です。確認方法はオープンしたデバイスに対する機能問い合わせであり、 リリース番号ではありません。ただし、これはトランスポートについての否定的な結果ではありません。同じ セッションで、トランスポートは設計どおりの働きをしているからです。

表 8. SPI からのバイト出力、同じアダプタとセッション、転送サイズ別

転送	リクエスト経路	バルク経路
8 バイト	2.05 ms	2.00 ms
1024 バイト	35.0 ms	5.0 ms
32768 バイト	利用不可	114.8 ms、285 kB/s

幅の広いトランスポートはペイロードのレバーであり、レイテンシのレバーではありません。 1 キロバイトでは 7 倍高速です。8 バイトでは同じ速度です。8 バイトの転送のコストはラウンドトリップ 1 回であり、パイプが 広くなってもラウンドトリップは短くならないためです。

これはセクション 4 全体をホスト側から言い直したものであり、2 つのセクションが一致する理由です。 `5.2.2.2` は WRITE を 8 バイトに、アクセスを 1 ページに制限するため、**ページ型の管理インターフェース上の CMIS は、これが役立つほど大きな転送を決して発行しません。** そのような転送を発行するインターフェースは 2048 バイトの SPIMCI であり、セクション 4 でブロックあたり 16 トランザクションを 1 つにまとめている行です。 このベンチには SPIMCI ターゲットが存在しないため、その行は算術のままです。

5.3 2 ミリ秒はどこから来るのか

独立した 2 つのホストスタックが 3 桁まで同じ数値で一致するのは偶然ではなく、原因はどちらのスタックでもなく アダプタの USB ディスクリプタにあります。

表 9. アダプタのエンドポイント、コンフィギュレーションディスクリプタから

インターフェース	エンドポイント	転送タイプ	最大パケット	インターバル
Human Interface Device	0x01 OUT, 0x81 IN	インタラプト	64 バイト	1 ms
ベンダー固有	0x02 OUT, 0x82 IN	バルク	64 バイト	なし

コマンドプレーンはインタラプトエンドポイントのペアであり、インタラプトエンドポイントはスケジュールに 従ってポーリングされます。 ここでのインターバルは 1 ミリ秒であるため、リクエストは次のフレームより前に 送出できず、その応答はさらに次のフレームより前には届きません。これが 2 ミリ秒の大部分を説明しており、 どの SDK が行っている処理でもありません。バスが転送の発生を許すタイミングなのです。残りはアダプタ ファームウェアのターンアラウンドとホストスタックであり、この測定ではそれらを分離していません。

バルクのペアはコマンドではなくペイロードを運びます。これが先の結果のすべてです。ペイロード経路を 広げてもコマンドのラウンドトリップは元のままなので、コマンド 1 つと応答 1 つだけでほかに何もない 8 バイトのアクセスは速くなりようがありません。

同じモデルはもう一方の列も予測でき、これこそ行う価値のある確認です。リクエスト経路での 1024 バイトの 転送はコマンド自体の中で 64 バイトずつ運ばれるため、ペイロードの 16 フレームにコマンドのラウンドトリップを 加えたものになります。約 34 ms で、実測は 35.0 ms です。バルク経路では、ペイロードはスケジュールから 完全に外れ、コマンドのラウンドトリップだけが残ります。約 2 ms に転送時間を加えたもので、実測は 5.0 ms です。

つまり、この下限には名前があり、ホストに何をすべきかを教えてくれます。 それを動かすのはコマンドの数と、 同時にいくつのコマンドが待機しているかです。どちらも、コマンドが通るパイプのサイズではありません。

5.4 下限の半分は応答であり、ホストはそれを隠せる

コマンドとその応答がそれぞれ 1 フレームのコストを持つなら、次のコマンドを送る前にすべての応答を待つ ホストは両方を支払い、コマンドをインフライト（in flight）に保つホストは前者だけを支払うはずです。これは 検証できます。同じ 1 バイトの読み出しをノンブロッキングインターフェース経由で 200 回発行し、1 回は各応答を 待ち、もう 1 回は待たずに実行しました。

表 10. 同じ 200 回の読み出し、各応答を待つ場合とインフライトに保つ場合の比較

	合計	コマンドあたり
1 つずつ	400.0 ms	2.000 ms
すべてインフライト	200.9 ms	1.005 ms

パイプライン化によって半分になり、取り除かれる半分は応答です。 残るのはフレームあたり 1 コマンドで、 これは送信方向のポーリングインターバルであり、ホストはコマンドの数を減らさない限りこれを下回れません。 200 回すべてに応答があったため、これはコマンドが破棄されたことを測定したものではありません。

それを望むホストにとって、2 つの帰結があります。削減は実在しますが限界があります。Extended Payload ブロックのホスト時間のコストを約 32 ms から約 16 ms に下げますが、tWRITE は 10 ms なので、ホストは ブロックを支配しなくなるものの、無視できるほどにはなりません。また、これはアクセスが互いに依存しない 場合にのみ利用でき、1 つのファームウェアブロック内では依存しません。同じ Extended Payload 領域への 連続した書き込みだからです。

はっきり述べておくべきこととして、**本ノートが提供するユーティリティはこれを行いません。** ユーティリティは ブロッキングインターフェースを使い、1 度に 1 コマンドずつ処理します。このため、セクション 5 のすべての 数値は 2 ミリ秒の列になっています。上記の測定はノンブロッキングインターフェースに対する別のハーネスで あり、ツールを説明するためではなく、この選択肢の価値を見積もるためのものです。

これはセクション 4 と並べて見る価値があります。Extended Payload ブロックは 16 回のアクセスであり、 10 ms の tWRITE に対してホスト時間は約 32 ms なので、このベンチではブロックを支配しているのはモジュールでは なくホストです。パイプライン化はそれを約 16 ms にし、**SPIMCI はそれを 16 トランザクションではなく 1 トランザクションにします**。これは、本ノートの中でセクション 5 が測定する項に届く最大の変更です。自身の ラウンドトリップを無視できると前提して構築されたホストはここでは誤っており、ページ型の管理インターフェース 上でできる最善はそれを半分にすることです。

これが、セクション 4 と 5 が反対方向から同じ場所に到達する理由です。

6 この結果をどう活かすか

ホストソフトウェア向け。 ダウンロードの開始時に `Firmware Management Features`（CDB コマンド `0041h`）を 1 回問い合わせ、モジュールがサポートしている場合は `0103h` よりも `Write Firmware Block EPL`（`0104h`）を優先してください。どちらであるとも決めつけないでください。そのうえで、ブロック内のアクセスは各応答を待たずに発行してください。セクション 5.4 はそれによりホストの コストが半分になることを測定しています。これが可能なのは 1 つのブロック内のアクセスが互いに依存しない ためであり、必要なのはより高速なインターフェースではなく、ノンブロッキングなインターフェースです。

モジュールファームウェア向け。 守ることのできるホールドオフ時間を通知してください。tWRITE は最大値で あって目標値ではなく、セクション 5 の測定は、残された余裕をホストが見つけ出す方法です。

新しい MCI バインディングを策定するすべての方へ。 SPIMCI はすでに 1 トランザクションで最大 2048 バイトを 運べ、その見積もりはセクション 4.1 で示しています。また I3C は SETMWL、SETMRL、GETMXDS によって同じことを ネイティブにネゴシエートします。I2CMCI の 8 バイトの書き込み制限を引き継ぐバインディングは、この問題全体を そのまま引き継ぎ、どのクロックで動作しても引き継ぎます。

> **Run Image ステップは別に見積もってください。** CDB コマンド `0109h` は完全なモジュールの再起動と同じだけの 時間がかかり、"mainly governed by the tMgmtInit timing parameter"（主に tMgmtInit タイミングパラメータに よって決まる）とされているため、CDB のタイムアウトではなく約 2 秒を見込んでください。

7 測定結果

7.1 テスト構成

表 11. テスト構成、セクション 5 の I2CMCI の実行

項目	値
コントローラ	Binho Supernova、シリアル `1912DEB5…79FA`、ファームウェア 4.4.0、ハードウェアリビジョン C
ターゲット	Binho Supernova、シリアル `ED7BF8F3…F7BB`、ファームウェア 4.4.0、アドレス 0x50 の I2C ターゲットとして設定
ターゲットの内容	CMIS 5.3 イメージ、Appendix C の 400G QSFP-DD の作例
SDK	`binhosupernova` 4.2.0、Python 3.10.12
ホスト	Ubuntu 22.04、x86-64
バス	3.3 V、400 kHz
繰り返し	フェーズごとに 20 回、中央値と最大値を報告
日付	2026 年 9 月 1 日

表 12. テスト構成、セクション 5.1 と 5.2 の I3C の実行

項目	値
コントローラ	Binho Supernova、シリアル `4F970418…1111`、ファームウェア 4.4.0 および 4.5.0
ホストトランスポート	USB HID 経由の `binhosupernova` 4.2.0、およびベンダー固有のバルクエンドポイント経由の CosmicSDK 1.5.0（機能問い合わせにより利用可能であることを確認、32768 バイトの上限を通知）
ターゲット	AN0009 の CMIS ターゲットファームウェアを実行する NXP FRDM-MCXN947、ダイナミックアドレス 0x09
ターゲットの内容	CMIS 5.3 イメージ、Appendix C の 400G QSFP-DD の作例
SDK	`binhosupernova` 4.2.0、Python 3.14
ホスト	Arch Linux、x86-64
バス	3.3 V、I3C SDR、プッシュプル 2.5 Mbit/s・デューティ比 25%、オープンドレイン 1 Mbit/s
繰り返し	フェーズごとに 20 回、中央値と最大値を報告、ファームウェアごとに 3 回の実行
セクション 5.4	ノンブロッキングインターフェースに対する別のハーネス、各条件 200 回の読み出し、3 回の実行
日付	2026 年 9 月 3 日

2 つの構成は、ベンチ、ホスト、Python のバージョンがすべて異なります。本ノートのどこでも互いに比較しては おらず、セクション 5.2 のファームウェア比較に使用しているのは 2 つ目の構成だけです。そこではホストと ターゲットを固定し、アダプタファームウェアだけを変えています。

7.2 結果

導出結果。 セクション 4 の見積もりは提供するユーティリティで再現でき、セクション 3 の仕様の引用だけを 根拠としています。これは算術であり、本ドキュメントはそれ以外のものとして提示していません。

実測結果、そしてそれはホストを測定しています。 セクション 5 の実行は、このホスト、アダプタ、SDK で、 それ自身のホールドオフを課さないターゲットを相手に、レジスタアクセスあたり約 2 ms の下限を確立しています。 この数値はバスでもモジュールでもなく USB 経路に属します。1 バイトの読み出しが tREAD の 4 倍を超え、 8 バイトの読み出しが 1 バイトの読み出しより遅くないことは、いずれもワイヤ時間が支配的な項でない場合にしか 起こり得ません。

これが、**本ドキュメントのどこにもエンドツーエンドのスループットの数値が現れない** 理由です。この方法で 測定した数値はホストを特徴づけるものであり、それをバスやモジュールの特性として引用すると、後から 発見しにくい形で誤ることになります。

ただし、これは本ノート自身の主張を弱めるのではなく、むしろ鋭くします。セクション 4 は、ファームウェア 更新の時間はワイヤ時間に費やされていないと主張しています。測定はさらに、ホストのラウンドトリップコストにも 費やされていないことを加えます。どちらもブロックごとのホールドオフと比べれば小さく、ブロックが大きくなると 同じ理由で縮小します。**アクセスが減り、ブロックが減るからです。** アクセスあたり 2 ms の下限は Local Payload ブロックあたり 30 ms、Extended Payload ブロックあたり 32 ms のコストになりますが、Extended Payload ブロックは 17 倍多くのファームウェアを運びます。

新しいアダプタファームウェアと新しいトランスポートで再測定しても、動きませんでした。 Supernova ファームウェア 4.5.0 は 4 つのフェーズすべてを 4.4.0 のときと同じ位置に保ち、そのファームウェアが追加した バルクトランスポートも、トランスポートが自身を利用可能と報告している状態で測定して同様でした。同じ セッションで、そのトランスポートは 1 キロバイトの転送では 7 倍高速なので、これはセットアップの失敗ではなく 限界のある結果です。幅の広いトランスポートはペイロードのレバーであってレイテンシのレバーではなく、 8 バイトのアクセスはどちらでもラウンドトリップ 1 回のコストがかかります。

これはセクション 4 を反対側から言い直したものです。`5.2.2.2` は WRITE を 8 バイトに、アクセスを 1 ページに 制限するため、ページ型の管理インターフェース上の CMIS は、幅の広いパイプが役立つほど大きな転送を決して 発行しません。この表に届くレバーは、ラウンドトリップの数（ブロックあたりのトランザクション数の列が数えて いるもので、SPIMCI が変えるもの）と、そのうちいくつが同時に未完了であるかです。

その 2 つ目のレバーは半分の価値があり、今日から利用できます。 セクション 5.4 は、各応答を待たずに 同じアクセスを 200 回発行し、2.000 ms に対して 1.005 ms を測定しています。すべてのコマンドに応答が ありました。残るのは USB フレームあたり 1 コマンドです。したがって、ファームウェアブロックを書き込むホストは、 アクセスが独立しているブロック内で待たないことで自身の寄与を半分にでき、その後はページ型インターフェース上 ではもうレバーがありません。

これは、セクション 4 のブロックあたりのトランザクション数の列こそが読むべき列である理由でもあります。 SPIMCI は、ページ型インターフェースが 16 トランザクションを必要とする Extended Payload ブロックを 1 回の トランザクションで運び、トランザクションあたり約 2 ミリ秒のホストコストでは、バスが何かを行う前の時点で ブロックあたり 2 ms 対 32 ms になります。

ハードウェア上、モジュールに対して。 ここには何もありません。リビジョン 1.0 時点で未完了の事項は次の とおりです。実際のモジュールに対する `timing` コマンドの実行（モジュールが実際に取るホールドオフを確立する 唯一の手段）、Local Payload と Extended Payload の両経路での完全なファームウェアダウンロード（エンドツーエンド の数値を裏付ける唯一の手段）、そして SPIMCI での何らかの測定（このベンチには SPIMCI ターゲットが存在しない） です。セクション 4 の SPIMCI の数値はすべて Appendix B.3 からの算術であり、そのように明記しています。

それが実行されるまで、本ノートが裏付ける主張は、**時間がどこで費やされることが許容されているか** と、 モジュールが何も寄与する前にホストがどれだけのコストを要するかについてのものです。

8 トラブルシューティング

表 13. トラブルシューティング

症状	考えられる原因と対処
`timing` が tREAD を大きく上回る 1 バイトの読み出しを報告する	モジュールではなく、ホストまたはアダプタのオーバーヘッドです。バスキャプチャと比較してください
すべてのフェーズが上限に近い	モジュールが本当に遅いか、別のコントローラがバスを競合している可能性があります
出力に 94 バイトの読み出しがない	モジュールがフルページ読み出しを通知していないため、`Nmax` が 8 のままです
実行ごとに中央値が大きくばらつく	ほかの何かが USB 経路を使っています。ほかのホストツールを閉じて再実行してください
新しいアダプタファームウェアでも数値が同じに見える	想定どおりであり、セクション 5.2 と 5.3 で説明しています。また、どのトランスポートを使っているかも確認してください。SDK がバルクトランスポートをネゴシエートしない場合、アダプタはエラーなしで HID 経路を提供します。バージョン番号から推測せず、SDK に問い合わせてください
すべてのアクセスに約 2 ms かかり、何を変えても改善しない	その大部分は USB のスケジュールです。コマンドプレーンは 1 ms でポーリングされるインタラプトエンドポイントであるため、コマンドとその応答は 2 フレームにまたがります。ディスクリプタはセクション 5.3 にあります。コマンドのサイズではなく数を減らしてください
ページ切り替えが tBPC の 2 倍を超える	モジュールが仕様で許容されるよりも長い時間をかけています。ホストを疑う前にバスをキャプチャしてください

9 略語

表 14. 略語

用語	意味
CDB	Common Data Block（共通データブロック）
CMIS	Common Management Interface Specification（共通管理インターフェース仕様）
EPL	Extended Payload、CDB ページ A0h から AFh
LPL	Local Payload、CDB ページ 9Fh 内
MCI	Management Communication Interface（管理通信インターフェース）
Nmax	1 回のレジスタアクセスでの最大バイト数
OIF	Optical Internetworking Forum
SDR	シングルデータレート（Single data rate）

10 本ドキュメント中のソースコードに関する注意

本ドキュメントに示し、付属のアーカイブで提供するサンプルコードは、評価および参考のために提供するものです。 MIT ライセンスの下でリリースされており、その本文はアーカイブに含まれています。

表 15. アセットアーカイブの内容

ファイル	説明
`AN0011.md`	本ドキュメントの Markdown ソース
`assets/cmis_mci.py`	リファレンスユーティリティ、AN0008 から AN0011 で共通
`assets/LICENSE`	サンプルコードに適用される MIT ライセンス

11 改訂履歴

表 16. 改訂履歴

リビジョン	日付	説明
0.1	2026 年 9 月 1 日	社内レビュー用のドラフト。見積もりは仕様から導出。測定は未実施。
0.2	2026 年 9 月 1 日	Supernova の I2C ターゲットフィクスチャに対して `timing` を実行。出力を掲載し、モジュールではなくホスト経路の測定として報告。
0.3	2026 年 9 月 2 日	AN0010 から番号を変更。測定と見積もりに変更なし。
0.4	2026 年 9 月 3 日	見積もりを SPIMCI に拡張し、初期値の 1 MHz と `MciSpeedConfiguration` の上限である 50 MHz を追加。ブロックあたりのトランザクション数の列を、ホストが体感する列として明示。セクション 5 に、AN0009 の CMIS ターゲットに対する I3C 経由の同じ測定と、2 つのホストトランスポートにわたる Supernova ファームウェア 4.4.0 と 4.5.0 の比較（どちらも下限を動かさない）を追加し、同じセッションでの SPI 転送サイズのスイープにより、幅の広いトランスポートが 1 キロバイトで 7 倍高速、8 バイトでは同一であることを示した。トランスポートが CosmicSDK 1.5.0 に導入された後、両トランスポートの実行をそれに対して繰り返し、変化なし。新しいセクション 5.3 で、2 ミリ秒の下限をコマンドプレーンが使う 1 ms のインタラプトエンドポイントまでたどり、そのモデルを転送サイズのスイープで確認。ユーティリティは 1.1。
0.5	2026 年 9 月 3 日	新しいセクション 5.4 で、コマンドをインフライトに保つことの価値を見積もり、すべてのコマンドに応答がある状態でコマンドあたりのコストが半分になることを測定。セクション 6 は、ホストにブロック内でそれを行うよう指示するようになった。残る下限はフレームあたり 1 コマンド。
1.0	2026 年 11 月 6 日	公開。

法的情報（英語）

Definitions

Draft

A document marked draft is under internal review and subject to formal approval. Binho Inc. makes no representation or warranty as to its accuracy or completeness and accepts no liability arising from its use.

Disclaimers

Limited warranty and liability

Information in this document is believed to be accurate and reliable. Binho Inc. makes no representation or warranty, express or implied, as to its accuracy or completeness, accepts no liability for the consequences of its use, and takes no responsibility for content originating outside Binho Inc.

In no event shall Binho Inc. be liable for any indirect, incidental, punitive, special or consequential damages, including without limitation lost profits, lost savings, business interruption, or costs of removing or replacing any product, whether or not based on tort, warranty, breach of contract or any other legal theory. Notwithstanding any damages a customer might incur, the aggregate and cumulative liability of Binho Inc. is limited in accordance with its terms and conditions of commercial sale.

Right to make changes

Binho Inc. reserves the right to change the information in this document, including specifications and product descriptions, at any time and without notice. This document supersedes all information supplied prior to its publication.

Suitability for use

Binho Inc. products are not designed, authorized or warranted for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure of a Binho Inc. product can reasonably be expected to result in personal injury, death, or severe property or environmental damage. Any such inclusion or use is at the customer's own risk and Binho Inc. accepts no liability for it.

Applications

Applications described here are illustrative only. Binho Inc. makes no warranty that they are suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their own applications and products, for appropriate design and operating safeguards, and for determining whether a Binho Inc. product is suitable and fit for their purpose. Binho Inc. accepts no liability for assistance with applications or customer product design.

Third-party products and specifications

This document refers to products, boards, specifications and documentation supplied by third parties, which remain the responsibility of their respective owners. Binho Inc. makes no warranty regarding them and their behavior may change without notice. Register maps, protocol details and device identifiers reproduced here were observed on the specific hardware and firmware versions stated in this document and must be confirmed against the vendor's own documentation for the reader's part.

Example code

Source code supplied with this document is provided as an example, for evaluation and reference. It is not qualified for production use, has not been developed under a functional safety or security process, and is licensed under the terms accompanying it in the distribution archive.

Terms and conditions of commercial sale

Binho Inc. products are sold subject to the general terms and conditions of commercial sale published by Binho Inc., unless otherwise agreed in a valid written individual agreement, in which case only the terms of that agreement apply.

Export control

This document and the items it describes may be subject to export control regulations. Export may require prior authorization from the competent authorities.

Security

All products may be subject to unidentified vulnerabilities or may support security standards with known limitations. The customer is responsible for the design and operation of its applications and products throughout their lifecycle to reduce the effect of such vulnerabilities, and for compliance with all legal, regulatory and security requirements concerning its products.

Trademarks

All referenced brands, product names, service names and trademarks are the property of their respective owners. **Binho** and the Binho logo are trademarks of Binho Inc. **CMIS** および **OIF** は Optical Internetworking Forum の商標または登録商標です。 **I3C** および **MIPI** は MIPI Alliance, Inc. の商標または登録商標です。