binho

应用笔记

在 Microchip dsPIC33AK 上构建 CMIS I3C 目标器件并使用 Binho Supernova 进行测试

AN0019

文档信息

关键词	AN0019, Supernova, CMIS, OIF, I3C, MIPI I3C Basic, SDR, I3C 目标器件, dsPIC33AK, dsPIC33AK256MPS306, MCC Melody, DMA, ENTDAA, IBI, GETSTATUS, 光模块
摘要	CMIS 光模块作为 I3C 目标器件应答其主机，而主机读取它的方式与 I2C 主机相同：写入字节地址、重复 START、读取。 在 dsPIC33AK256MPS306 的 I3C 外设上，该写操作的中断在读头已被应答之后才到达，因此由该中断装载应答的目标器件 会对每一次此类读取回以 NACK。本应用笔记说明字节地址在何处能够及时获得、固件如何改为由接收 DMA 进行装载， 以及在 Binho Supernova 上测得的数据：4 到 128 字节的读取 60 次中 60 次成功，开启带内中断时 300 次中 300 次成功， 复位后同样如此。本笔记提供主机端测试脚本，并列出移植时必须预先考虑的外设和驱动程序行为。

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

修订版 1.0
6 NOVEMBER 2026

> **测试范围**。第 8 节中的每项结果均在第 3 节所述的测试台上测得。目标器件固件不在资产压缩包中，如需获取请联系 techsupport@binho.io。

1 简介

CMIS 模块提供一个 256 字节的窗口，由低端存储区（Lower Memory）加上高端存储区（Upper Memory）的一个 128 字节页组成，页由 `00h:127` 处的页选择字节选定。通过 I2C，主机先写入字节地址，然后以重复 START 进行读取。 正在为 CMIS 定义的 I3C 管理接口保留了这种访问方式，因此基于 I3C 目标器件外设构建的模块必须应答这样一个读取： 其字节地址在同一次传输中仅早几个位时间到达。

如果外设在每个字节到达时就将其交给固件，这很容易实现。而 dsPIC33AK256MPS306 的 I3C 外设则基于命令和响应： 接收到的传输只以响应的形式报告一次，读取由一个发送命令应答，而该命令必须在读头到达时就已经有效。 本应用笔记讨论的就是如何让这两者衔接起来。

1.1 范围

- dsPIC33AK 上的 CMIS 寄存器模型：分页、第 8.2.15 节的不支持页覆盖、附录 B.1.2 规定的 128 字节段内回绕， 以及携带 Flags Summary 字节 `00h:4` 的 IBI。
- 在 2.5 MHz 推挽模式下应答写入 + Sr + 读取（组合读取）以及写入、STOP、读取（分离读取）。
- 动态地址分配（DAA）、RSTDAA 和 Direct GET CCC。
- 一个从 Supernova 运行本笔记中所有检查的主机脚本。

不涉及：CDB、固件下载、HDR 模式，以及 CMIS 控制寄存器在模块端的含义。除页选择和主机写入的字节外， 寄存器映像是静态的。

1.2 适用器件

表 1. 适用器件

系列	示例器件	标识	依据
带 I3C 的 dsPIC33AK MPS	dsPIC33AK256MPS306（EV02E57A DIM）	PID `02 36 43 4D 49 53`	已测试
不带 I3C 的 dsPIC33AK MP	dsPIC33AK512MPS506 及其系列	-	不适用：无 I3C 外设
具有相同 I3C 外设的其他器件	-	-	未测试

1.3 相关文档

表 2. 相关文档

文档	标题
OIF-CMIS-05.3	Common Management Interface Specification，修订版 5.3
MIPI I3C Basic v1.1.1	MIPI I3C Basic Specification，版本 1.1.1
DS70005539	dsPIC33AK MPS 系列数据手册，第 24 节（I3C）
AN0009	使用 Binho Supernova 通过 I3C 管理 CMIS 光模块

2 系统概述

Supernova 是 I3C 控制器，也是总线上唯一的控制器。dsPIC33AK 是目标器件，静态地址为 `0x50`，即 CMIS 模块地址。控制器通过 ENTDAA 为其分配动态地址（测试台分配 `0x08`），然后以 SDR 对其进行读写。

决定固件设计的唯一事实是：组合读取的字节地址何时变得可见。在关闭 I3C 中断的情况下，以 100 MHz 定时器为基准 轮询外设，并与逻辑分析仪对齐，测得：

表 3. 2.5 MHz 推挽下的组合读取，从 START 起算的时间

事件	时间
字节地址的 T 位	22.3 µs
字节地址进入接收 FIFO	23.8 µs
重复 START	24.1 µs
读头的 ACK 时隙	~27.4 µs
写操作的响应中断	ACK 时隙之后

由写操作的响应中断装载（arm）的发送命令总是太迟，目标器件会对读取回以 NACK。字节地址本身在 ACK 时隙之前 3.6 µs 即可获得，而装载一个命令需要 1.15 到 1.26 µs。因此固件由接收 DMA 进行装载：将接收到的字搬运到 RAM 的通道每个字产生一次中断，其优先级高于 I3C 中断，该中断根据第一个字节装载发送命令。

其余一切都源于该外设及其 MCC Melody 驱动程序的三个特性：

- 一个发送命令应答一次读取。固件装载从字节地址到段末尾的 128 字节数据串；控制器会提前结束读取。
- 发送缓冲区可容纳 8 个字（32 字节）。数据串的其余部分由主循环供给。任何使主循环离开的时间超过发送 32 字节 所需时间（约 115 µs）的操作，都会使读取在 32 字节处被截断。
- 重新装载接收会复位接收 FIFO，同时也会丢弃已装载的发送命令。接收仅在读取之后才重新装载。

3 硬件设置

3.1 所需设备

- Binho Supernova，固件 4.6.2，I3C HV 端口。
- Microchip EV74H48A Curiosity 平台，配 EV02E57A dsPIC33AK256MPS306 DIM。其 PKoB4 用于对器件编程并承载 UART。
- 可选：连接 SCL 和 SDA 的逻辑分析仪。本笔记中的捕获使用 Saleae Logic 以 12 MS/s 采集。

3.2 连接

表 4. Supernova 与目标器件之间的连接

Supernova I3C 端口	dsPIC33AK DIM	是否必需
SCL	I3CSCL1（RC5）	是
SDA	I3CSDA1（RC4）	是
GND	GND	是
I3C 电源轨，3.3 V	不连接，DIM 由 Curiosity 板供电	-
-	UART1 TX（RD0），经 PKoB4 虚拟 COM 端口输出	可选

3.3 总线注意事项

总线工作在 3.3 V。推挽 2.5 MHz 是 Supernova 提供的最低速率，也是本文唯一测量过的速率。更快的速率会缩短表 3 中的 3.6 µs 窗口：在 12.5 MHz 时，该窗口短于装载一个命令所需的时间。

4 软件设置

目标器件固件：

- MPLAB X v6.35 及 MPLAB IPE、XC-DSC v4.00、dsPIC33AK-MP DFP 1.4.260。
- MCC Melody I3C 目标器件驱动程序，含第 5.6 节所述的本地修改。

主机：

- Python 3 及 pycosmicsdk（最新版本），以及本笔记资源中的 `cmis_target_check.py`。

```
pip install --upgrade pycosmicsdk
```

5 目标器件固件设计

5.1 寄存器模型

低端存储区和高端存储区页 `00h` 取自 CMIS 5.3 附录 C 的示例映像。`00h:127` 处的页选择接受 `00h` 和 `01h`； 任何其他值都会被替换为 `00h`（第 8.2.15 节）。每个段首尾相接存储两次，另加偏移一到三个字节的副本， 因此从任何字节地址开始的 128 字节数据串都是一个对齐的缓冲区，段内的 B.1.2 回绕无需任何代码。

5.2 标识

表 5. 控制器看到的标识

项目	值
静态地址	`0x50`
PID	`02 36 43 4D 49 53`（MIPI 制造商 `0x11B`，器件 `"CMIS"`）
BCR	`0x17`：支持 IBI，IBI 携带一个字节，有最大数据速率限制
DCR	`0x00`

5.3 装载读取

1. 接收 DMA 通道将每个接收到的字搬运到 `rxbuf`，其完成中断已使能，优先级为 5，高于优先级为 4 的 I3C 中断。
2. 在写操作的第一个字到达时，DMA 中断将 `rxbuf[0]` 作为字节地址，并装载从该地址开始的数据串的发送命令。
3. 当写操作的响应到达且其中只携带了地址时，命令保持装载状态，接收在读取完成后重新装载。
4. 当写操作携带了数据时，通过复位扩展发送缓冲区来取消该命令，并立即重新装载接收。
5. 如果读头到达时仍未找到已装载的命令，外设会发出读请求。固件丢弃任何迟到装载的命令，并通过 `I3C1_Target_ReceiveCancel()` 重新装载接收，因为在这种情况下写操作的响应永远不会到来。

5.4 供给长读取

发送缓冲区预先填入 8 个字。`I3C1_Target_TxRefill()` 从主循环供给其余部分，在其执行的少量指令期间屏蔽 I3C 和 DMA 中断。调试固件的 UART 状态行默认关闭：其在 9600 波特率下的阻塞式 `printf` 每秒占用主循环约 110 ms， 使 128 字节的读取在 32 字节处被截断。

5.5 事件

在 IBI 使能期间，IBI 每秒发出一次，以 `00h:4` 作为其必需的数据字节。外设复位后 IBI 处于使能状态，这符合 I3C Basic 的规定，并且外设不会保持写入该使能位的 0：DISEC 才是关闭 IBI 的方法。

5.6 对 Melody 驱动程序的修改

Table: `mcc/` 中的本地修改，以 `binho:` 标记

修改	原因
`BufferTransmit` 预先填入 8 个字；`I3C1_Target_TxRefill()` 供给其余部分	超过 8 个字时，DMA 路径将第 7 个字写了两次且不再继续，因此命令被拒绝
`I3C1_Target_ReceiveCancel()`	在写入 + Sr + 被 NACK 的读头之后，写操作不会有响应；驱动程序一直将其接收标记为进行中，并拒绝之后的每一次接收
在完成回调之前清除 `rxInProgress`	使回调能够重新装载接收
静态地址 `0x50`	CMIS 模块地址

6 操作步骤

步骤 1：在固件文件夹中构建目标器件固件并编程：

```
make flash
```

构建以 MPLAB IPE 输出的 `Program Succeeded.` 结束。

步骤 2：按表 4 将 Supernova I3C 端口连接到 DIM。

步骤 3：以写入 + Sr + 读取方式读取 128 字节，共 60 次：

```
python cmis_target_check.py reads --length 128 --count 60
```

```
init bus: FW_I3C_BUS_INIT_NACK_RSTDAA
target DA 0x08, PID 02 36 43 4d 49 53
............................................................
reads 60/60  (combined, 128 B, IBI off)
```

`init bus` 行表示复位后的第一次 RSTDAA 被 NACK（第 8.3 节）；脚本会再次初始化。

步骤 4：在使能 IBI 的情况下执行相同操作（读取之前发送 ENEC）：

```
python cmis_target_check.py reads --length 128 --count 300 --ibi
```

```
reads 299/300  (combined, 128 B, IBI on)
```

点行中的 `F` 表示出现错误的读取，`B` 表示返回错误数据的读取。

步骤 5：数据写入与读取混合：每一轮写入 `7Fh=00h` 和 `7Eh=00h`，然后读取 `7Fh`（1 字节）和 `80h`（16 字节），并与映像进行比对：

```
python cmis_target_check.py mixed --count 60
```

```
rounds 60/60
```

步骤 6：地址复位和重新分配：

```
python cmis_target_check.py daa --cycles 10
```

```
0: RSTDAA SUCCESS, GETSTATUS at old DA FW_I3C_NACK_ADDRESS, read after ENTDAA ok
...
cycles 10/10
```

步骤 7：标识和 Direct GET CCC：

```
python cmis_target_check.py ccc --getstatus 100
```

```
GETBCR   SUCCESS               17
GETDCR   SUCCESS               00
GETPID   SUCCESS               02 36 43 4d 49 53
GETMXDS  SUCCESS               00 00 00 00 01
GETMRL   FW_I3C_NACK_ADDRESS
GETMWL   SUCCESS               ff
GETSTATUS 100/100  {'SUCCESS': 100}
```

Direct GET 可能因 `7Eh` 头被 NACK 而失败（第 8.3 节）；此处 GETMRL 即失败了。请再次运行。

7 参考

Table: `cmis_target_check.py` 命令

命令	功能
`reads --length N --count N [--split] [--ibi]`	依次在 `00h`、`10h`、`7Ch`、`80h`、`A0h`、`FCh` 处读取并与映像比对。除非指定 `--split`，否则为组合读取；`--ibi` 先发送 ENEC，否则发送 DISEC
`mixed --count N`	向 `7Fh` 和 `7Eh` 写入数据，然后对 `7Fh` 和 `80h` 进行组合读取并校验

命令	功能
`ccc [--getstatus N]`	GETBCR、GETDCR、GETPID、GETMXDS、GETMRL、GETMWL，然后执行 GETSTATUS N 次
`daa --cycles N`	RSTDAA，检查旧地址是否 NACK，ENTDAA，检查一次读取
`i2c`	在任何地址分配之前，以 I2C 模式在 `0x50` 处读取 `00h:0-3` 和 `00h:46-48`

通用选项：`--serial`、`--push-pull`（默认 `PUSH_PULL_2_5_MHZ_DC_25`）、`--voltage-mv`（默认 3300）、`--image`（默认为脚本旁的 `cmis_image.h`）。每条命令都以 RSTDAA 和 ENTDAA 开始，并通过 PID 中的 `"CMIS"` 找到目标器件。

8 测量结果

8.1 测试配置

表 6. 测试配置

项目	版本
Supernova	序列号 4F970418，固件 4.6.2
目标器件	EV74H48A 上的 EV02E57A DIM
工具链	XC-DSC v4.00、DFP 1.4.260、MPLAB X v6.35
主机	Ubuntu 26.04；会话 1 和 2：Python 3.13、binhosupernova 4.2.0；会话 3：Python 3.14.4、pycosmicsdk 1.9.1
总线	3.3 V，推挽 2.5 MHz，开漏 1 MHz

8.2 结果

共三个会话。会话 2 在编程（一次复位）之后开始；会话 3（2026 年 11 月 6 日）在每项测试前都对目标器件编程， 主机使用 pycosmicsdk：

表 7. 结果

测试	会话 1	会话 2	会话 3
组合读取，4 / 16 / 32 / 128 B，IBI 关闭	各 60/60	60/60（4 和 128 B）	各 60/60
组合读取，128 B，IBI 开启	300/300	300/300	299/300
分离读取，4 / 128 B	各 60/60	60/60（128 B）	60/60（128 B）
数据写入和组合读取，数据经校验	60/60 轮	60/60 轮	60/60 轮
RSTDAA，旧地址 NACK，ENTDAA，读取	10/10	10/10	10/10
GETSTATUS	97/100	94/100	100/100

在采用接收 DMA 装载之前，同样的组合读取为 60 次中 0 次成功。

8.3 主机应预期的行为

- **目标器件复位后的第一次传输会被 NACK**。在 5 次复位中的每一次之后，第一个 `7Eh` 头都被 NACK，第二个被 ACK。 DS70005539 第 24.4.5 节描述了该模块只有在检测到 SCL 时钟后才退出复位。只探测一次并将 NACK 视为 “无 I3C 目标器件”的主机将会漏掉此模块；请重试探测。
- **部分 Direct GET CCC 会因 `7Eh` 被 NACK 而失败**。在一个会话中，Direct GET 之后的 `7Eh` 头约每 7 个中有 1 个被 NACK（1200 次 GET 中有 173 次失败，全部 173 次在线路上均可见为 `7E/W NACK`）；在另一个会话中， 500 次中 500 次通过。私有读取不受影响。请重试因 NACK 而失败的 GET。
- **GETMRL 报告 255，且 SETMRL 不生效**。固件设置为 128；外设自行以 255 应答 GETMRL。执行 SETMRL 16 之后， 32 字节的读取仍返回 32 字节。

8.4 地址分配之前的 I2C 读取

在 ENTDAA 之前，目标器件在 `0x50` 处应答 I2C。从 Supernova 的 I2C 端口（一个普通 I2C 控制器，3.3 V 下使用 2.2 kOhm 上拉电阻）进行读取，每次运行前都对目标器件编程，在 `00h`、`10h`、`7Ch`、`80h`、`A0h` 和 `FCh` 处读取 3、4、16 和 128 字节，在 100 kHz 和 400 kHz 下均为各 30/30，数据与映像比对一致。从 I2C 模式下的 I3C 端口（每次传输都以 `7Eh` 头开始）进行读取时，会话 1 中 30 次有 8 次正确；在会话 3 中，复位后的第一次 读取被 NACK，第二次从 `00h:46-48` 返回 `80 00 00`。

9 故障排除

表 8. 故障排除

现象	可能原因及措施
每次组合读取都以 `FW_I3C_NACK_ADDRESS` 失败	命令装载太迟：检查 DMA 中断优先级是否高于 I3C 中断，以及其中是否运行了耗时的操作
128 字节读取失败，分析仪显示 308 个 SCL 时钟	读取在 32 字节处被截断：主循环没有供给发送缓冲区。移除阻塞式 `printf` 及类似操作
读取失败后目标器件停止应答，直到复位	接收一直被标记为进行中；固件需要 `I3C1_Target_ReceiveCancel()`
上电后的第一次初始化以 `FW_I3C_BUS_INIT_NACK_RSTDAA` 失败	复位后的第一次传输被 NACK；请再次初始化
MPLAB IPE 报告 "Connection Failed" 或 `Transmission on endpoint 2 failed (err = -150)`	需要重新插拔 PKoB4 的 USB

10 缩略语

表 9. 缩略语

术语	含义
BCR	Bus Characteristics Register（总线特性寄存器）
CCC	Common Command Code（通用命令码）
CMIS	Common Management Interface Specification（通用管理接口规范）
DA	Dynamic Address（动态地址）
DCR	Device Characteristics Register（器件特性寄存器）
DMA	Direct Memory Access（直接存储器访问）
ENTDAA	Enter Dynamic Address Assignment（进入动态地址分配）
IBI	In-Band Interrupt（带内中断）
MRL	Max Read Length（最大读取长度）
PID	Provisioned ID（预置 ID）
RSTDAA	Reset Dynamic Address Assignment（复位动态地址分配）
SDR	Single Data Rate（单数据速率）
Sr	Repeated START（重复 START）

11 关于本文档中源代码的说明

本文档中展示并随附存档中提供的示例代码仅供评估和参考。其依据 MIT 许可证发布，许可证文本包含在存档中。

存档 `AN0019-assets.zip` 包含 `cmis_target_check.py` 和 `cmis_image.h`。

12 修订历史

表 10. 修订历史

修订版	日期	说明
0.1	29 October 2026	供内部审阅的草稿。
0.2	6 November 2026	主机脚本迁移到 pycosmicsdk；会话 3 的结果；地址分配之前的 I2C 读取改用普通 I2C 控制器测量。
1.0	6 November 2026	正式发布。

法律信息（英文）

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. **I3C** 和 **MIPI** 是 MIPI Alliance, Inc. 的商标或注册商标。 **CMIS** 和 **OIF** 是 Optical Internetworking Forum 的商标或注册商标。 **Microchip**、**dsPIC**、**MPLAB** 和 **Curiosity** 是 Microchip Technology Incorporated 的商标或注册商标。**Saleae** 和 **Logic** 是 Saleae, Inc. 的商标。