**TINY CONTROLS PRIVATE LIMITED**
Advanced CNC Systems

# TIO-0808

## Mixed Signal I/O Card

*with RS-485 Modbus RTU Interface*

## OPERATION MANUAL

| | |
|---|---|
| **Document No.** | TIO-0808-21 |
| **Document Rev.** | 2.0 |
| **Product Rev.** | 1.0 |
| **Firmware** | 3.00 |
| **Date** | August 2026 |

Tiny Controls Private Limited | B-17/A, Nishat Park, Kakrola, New Delhi, India – 110078
www.tinycontrols.com | anil@tinycontrols.com | +91-991-119-3210

# Table of Contents

# Chapter 1 — Safety Information

Read all safety warnings before installing or operating the TIO-0808.

## 1.1 Warning Levels

| Level | Symbol | Meaning |
|---|---|---|
| DANGER | Red border | Imminently hazardous — will result in death or serious injury if not avoided. |
| WARNING | Orange border | Potentially hazardous — could result in death or serious injury. |
| CAUTION | Yellow border | May result in minor injury or equipment damage. |
| NOTE | Blue border | Important information for correct operation. No injury risk. |

## 1.2 Electrical Safety

> ⚠ **WARNING**
> The TIO-0808 operates from 24 V DC. Always disconnect the power supply before making or changing any wiring connections.

> **CAUTION**
> The analog output (DAC) requires a separate isolated 10 V supply on its dedicated input terminal. Do not connect this terminal to the 24 V supply — doing so will damage the card.

> **CAUTION**
> The analog input (ADC) and PWM output are non-isolated and share GND with the 24 V supply. Ensure signal grounds are correctly referenced to avoid damaging the card or connected equipment.

## 1.3 Motion Safety

> ⚠ **WARNING**
> Seek commands (5, 6) run until the target input asserts — no distance limit, no timeout. If the input is not wired, not powered, or already passed, the axis runs forever. Set a master-side timeout and send command 0 (Stop) to end it.

> **CAUTION**
> Confirm the axis is stopped (register 0x0304, bit 0 clear) before power-off or Save (command 8). Save is refused while moving.

> **NOTE**
> Command 0 (Stop) ramps down — not a hard stop.

## 1.4 General Safety Rules

- Always disconnect 24 V power before changing DIP switch settings.
- Never wire RS-485, signal, and power cables in the same conduit.
- Ensure no two devices on the RS-485 bus share the same address.
- Always terminate the RS-485 bus at the last device with a 120 Ω resistor.
- Do not exceed relay contact or output current ratings.
- Use flyback diodes across all inductive loads connected to the outputs.
- Never issue a Seek command without a master-side timeout — the axis will not stop by itself.
- Verify the moving bit (state word bit 0) is clear before saving parameters or removing power from a card driving an axis.

# Chapter 2 — Product Overview

The TIO-0808 is a general-purpose mixed-signal I/O card with Modbus RTU over RS-485. It also drives a step/direction motion axis, using the same register map as the Tstep-087 stepper drive.
The card has 8 opto-isolated digital inputs, 6 outputs (2 relay, 4 open-collector), an isolated 0–10 V analog output, a 0–10 V analog input, a variable PWM output, and a step/direction axis. Up to 8 cards share one RS-485 bus (addresses 17–24); Tstep-087 drives use 1–16, so both can share the same bus.

## 2.1 Key Features

- Modbus RTU protocol over RS-485 — function codes 0x03, 0x04, 0x06 and 0x10.
- 8 opto-isolated digital inputs.
- 6 outputs (OUT1–OUT6): 2 relay (N/O) and 4 open-collector, for load switching.
- Step/direction motion axis — acceleration, speed, distance, position, seek-to-input homing.
- 1 × 0–10 V isolated analog output (DAC). Requires a separate isolated 10 V supply input.
- 1 × 0–10 V non-isolated analog input (ADC) — returns voltage in millivolts.
- 1 × PWM output: open-collector, non-isolated, frequency 100–10,000 Hz, duty 1–99.9%.
- Indicator LED with programmable blink rate via Modbus register.
- CPU temperature readable over Modbus.
- Sequence-numbered motion commands (register 0x020B) for safe retry after a timeout.
- RS-485 bus: up to 8 TIO-0808 cards (addresses 17–24) per bus, coexisting with Tstep-087 drives (addresses 1–16).
- Address (S1–S3) and baud rate (S4) set via 4-position DIP switch.
- 24 V DC powered, < 100 mA with all outputs off.

## 2.2 Technical Specifications

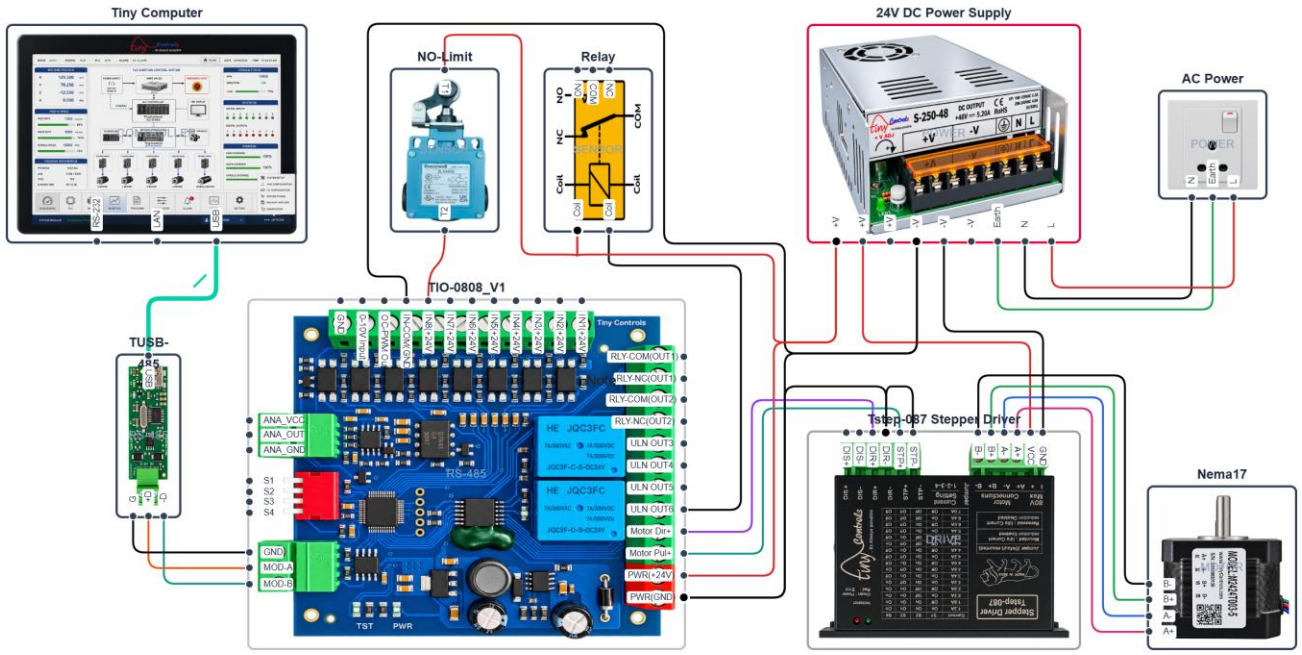
| Parameter | Specification |
|---|---|
| Supply Voltage | 24 V DC |
| Current Consumption | < 100 mA (all outputs off) |
| Digital Inputs | 8 × opto-isolated, 12–24 V compatible |
| Outputs | 6 channels (OUT1–OUT6): 2 relay (N/O) and 4 open-collector, via 6-bit output port register |
| Motion Axis Outputs | OUT7, OUT8 dedicated to step/direction signals |
| Analog Output (DAC) | 0–10 V isolated. Requires separate isolated 10 V supply. Range: 1–99% of supply. |
| Analog Input (ADC) | 0–10 V non-isolated. Returns voltage in mV. |
| PWM Output | Open-collector, non-isolated. Frequency: 100–10,000 Hz. Duty: 1–99.9%. |
| Indicator LED | Programmable blink rate 0–100 per sec (0 = Off, 100 = Solid On) |
| Max Step Rate | 100,000 steps/s |
| Communication Protocol | Modbus RTU — function codes 0x03, 0x04, 0x06, 0x10 |

| | |
|---|---|
| RS-485 Address Range | 17–24 (S1–S3 selectable, 8 unique addresses) |
| Baud Rates | 19,200 (S4 = OFF) or 115,200 (S4 = ON) |
| Max Operating Temp. | 70 °C |

# Chapter 3 — Installation and Wiring

> ⚠ **WARNING**
> Disconnect 24 V power before making any wiring connections.

## 3.1 Board Wiring Diagram




> **NOTE**
> The HMI panel, power supply, AC power connection, relay, limit switch, RS-485 adapter, stepper driver, and stepper motor shown above are external components used to illustrate a typical wiring setup — none of them are included with the TIO-0808_V1 board itself.

## 3.2 Power Supply

Connect regulated 24 V DC to the power input terminals. Current draw is < 100 mA with all outputs off. Allow extra margin for relay coil and output load current.

> **NOTE**
> The DAC output needs a separate isolated 10 V supply on the dedicated isolated supply input terminal, electrically isolated from the 24 V rail. A dedicated 10 V isolated DC-DC converter is recommended.

## 3.3 Digital Inputs (IN1–IN8, Opto-isolated)

All 8 digital inputs are opto-isolated and compatible with 12–24 V signals.

- Connect the signal positive to the input terminal (IN1–IN8).
- Connect the signal GND or return to the input common terminal.

- Read all 8 inputs at once from register 0x0500 (Input Port); bit n = input n+1.
- Inputs 1 and 2 are also available debounced at register 0x0305, bits 0–1 — this is what seek commands (5, 6) use.

## 3.4 Outputs (OUT1–OUT6): Relay and Open-Collector

OUT1–OUT6 are 2 on-board relay outputs (N/O contacts) and 4 open-collector transistor outputs.

- Relay outputs: connect the load across the relay's common and N/O contact terminals.
- Open-collector outputs: connect the load between the positive supply voltage and the OUTx terminal — the transistor sinks current when the output is ON.

**CAUTION**
Always use a flyback diode across inductive loads (solenoids, coils) connected to the open-collector outputs. Failure to do so can damage the output transistors.

**NOTE**
The output port register 0x0405 is six bits wide (bit n = output n+1). Writing bit 6 or 7 returns exception 03 — those lines belong to the motion axis, section 3.5.

## 3.5 Step/Direction Axis Outputs (OUT7, OUT8)

Outputs 7 and 8 are dedicated to the step/direction axis and are not general-purpose outputs — writing bit 6 or 7 of register 0x0405 returns exception 03.
Wire OUT7 and OUT8 to the STEP and DIR inputs of an external stepper driver, referenced to the card's GND. See Chapter 6 (registers) and Chapter 7 (commands).

**CAUTION**
This card has no driver-enable line or driver-fault input. Commands 17, 18 and 19 exist on the Tstep-087 but are refused here (exception 03).

## 3.6 Analog Output — DAC (0–10 V Isolated)

The DAC provides an isolated 0–10 V analog output voltage, set as a percentage of the isolated supply voltage.

1. Connect an isolated 10 V supply to the isolated supply input terminal.
2. Write register 0x0401 = 1 to enable the DAC output.
3. Write register 0x0400 (1–99) to set the output voltage.

**NOTE**
Output voltage = (Duty% ÷ 100) × isolated supply voltage. For example, 50% with a 10 V supply gives 5.0 V output.

## 3.7 Analog Input — ADC (0–10 V Non-isolated)

The analog input accepts 0–10 V signals. Read the current voltage from register 0x0502, in millivolts (e.g. 5000 = 5.000 V).

**CAUTION**
The analog input is non-isolated and shares GND with the 24 V supply. Ensure the signal source GND is at the same potential. Do not apply more than 10 V to this input.

## 3.8 PWM Output (Non-isolated)

The PWM output is open-collector and non-isolated, with programmable frequency and duty cycle. It shares GND with the 24 V supply.

1. Write register 0x0402 to set the PWM frequency (100–10,000 Hz).
2. Write register 0x0403 to set the PWM duty cycle (1–999, representing 0.1%–99.9%).
3. Write register 0x0404 = 1 to enable the PWM output.

## 3.9 RS-485 / Modbus Connection

| Terminal | Connection |
|---|---|
| G | RS-485 signal ground — connect to host controller GND or cable shield. |
| D+ | RS-485 data positive (also called A or +). |
| D– | RS-485 data negative (also called B or –). |

**NOTE**

Use twisted-pair cable for D+/D–. 8N1 (8 data bits, no parity, 1 stop bit). Terminate the last device with a 120 Ω resistor. Up to 8 cards per bus (addresses 17–24); Tstep-087 uses 1–16. No auto-baud — set the master to match switch S4.

# Chapter 4 — DIP Switch Configuration

The 4-position DIP switch (S1–S4) sets the card address and baud rate. Always disconnect power before changing switch settings.

> **CAUTION**
> Always disconnect power before changing DIP switch settings.

## 4.1 Card Address — S1, S2, S3

Each TIO-0808 on the bus must have a unique address. Addresses 17–24 are reserved for this card: Address = 17 + (4·S3 + 2·S2 + 1·S1).

| Address | S1 | S2 | S3 |
|---|---|---|---|
| 17 | OFF | OFF | OFF |
| 18 | ON | OFF | OFF |
| 19 | OFF | ON | OFF |
| 20 | ON | ON | OFF |
| 21 | OFF | OFF | ON |
| 22 | ON | OFF | ON |
| 23 | OFF | ON | ON |
| 24 | ON | ON | ON |

> **NOTE**
> Switches are read once at power-up — change a switch, then power-cycle the card. Address range 17–24 is specific to the TIO-0808; Tstep-087 uses a different range.

## 4.2 Baud Rate — S4

| Baud Rate | S4 |
|---|---|
| 19,200 | OFF |
| 115,200 | ON |

> **NOTE**
> All devices on the RS-485 bus must be set to the same baud rate.

# Chapter 5 — Modbus RTU Protocol Fundamentals

The card communicates using standard Modbus RTU. This chapter covers the wire protocol. Register map: Chapter 6. Commands: Chapter 7.

## 5.1 Serial Port Settings

| Parameter | Value |
|---|---|
| Baud Rate | 19,200 (S4 = OFF) or 115,200 (S4 = ON) |
| Data Bits | 8 |
| Parity | None |
| Stop Bits | 1 |
| Flow Control | None |

## 5.2 Quick Start — Reading the Identity Block

1. Set the card's address on switches S1–S3 — all off is address 17.
2. Set the baud rate on switch S4 — off is 19,200, on is 115,200.
3. Wire D+ and D– to the bus, and G to the card's ground reference.
4. Power the card and read registers 0x0000–0x0003 (function 0x03). A working card answers with its product ID 0x0808 and the address set on the switches.

```
->  11 03 00 00 00 04 46 99      read identity from the card at address 17
<-  11 03 08 08 08 03 00 00 11 00 1F 58 8F
```

**NOTE**
Matching this exact exchange confirms your master's framing, byte order, and CRC are all correct — everything else in this manual follows the same pattern.

## 5.3 Frame Format

Every request and reply follows the same layout: [address][function][data …][CRC low][CRC high].

- Frames are separated by silence, not a terminator — 3.5 character times of no traffic marks end of frame.
- Send a whole frame in one write call — a pause inside it is read as end-of-frame and fails CRC on both halves.
- Maximum frame length is 64 bytes. Anything longer is discarded.

## 5.4 CRC-16/MODBUS

Checksum: CRC-16, polynomial 0xA001 (reflected), initial value 0xFFFF, no final XOR — standard Modbus CRC. Computed over address through last data byte (not the CRC bytes).

⚠ **WARNING**
The CRC is sent low byte first. Every other multi-byte value (addresses, quantities, register values) is big-endian, high byte first. Forgetting to byte-swap the CRC is the most common error in master implementations.

```
uint16 t crc16 modbus(const uint8 t *data, int length) {
    uint16_t crc = 0xFFFF;
    for (int i = 0; i < length; i++) {
        crc ^= data[i];
        for (int bit = 0; bit < 8; bit++) {
            if (crc & 1) crc = (crc >> 1) ^ 0xA001;
            else         crc = crc >> 1;
        }
    }
    return crc;
}
// appended low byte first: frame[n]=crc&0xFF; frame[n+1]=crc>>8;
```

Worked check for frame 01 03 00 00 00 04 (address 1, read holding registers, start 0x0000, quantity 4): CRC = 0x0944, on the wire low byte first = 44 09. Complete frame = 01 03 00 00 00 04 44 09. Standard CRC-16/MODBUS check value for "123456789" is 0x4B37 — use it to test an implementation independently.

**NOTE**

If framing and CRC are correct, the worked examples in Chapter 10 reproduce byte for byte — check against them before using real hardware.

## 5.5 Function Codes and 32-bit Values

| Code | Name | Limit |
|---|---|---|
| 0x03 | Read Holding Registers | 8 registers per request |
| 0x04 | Read Input Registers | identical behaviour to 0x03 |
| 0x06 | Write Single Register | — |
| 0x10 | Write Multiple Registers | 4 registers per request |

Any other function code returns exception 01. This includes Read Device Identification (0x2B/0x0E) — use the identity registers instead.

Motion registers (speed, distance, position) are 32-bit: two consecutive registers, even start address, high word first — value = (register[n] << 16) | register[n+1]. Fixed, not configurable. Position is signed; everything else unsigned.

The 4-register write limit means one transaction carries two 32-bit values. The card’s own I/O registers (Chapter 6) are all single 16-bit values — only motion uses pairs.

# Chapter 6 — Register Map

## 6.1 Identity Registers (Read Only)

| Register | Name | Value |
|---|---|---|
| 0x0000 | Product ID | 0x0808 |
| 0x0001 | Firmware version | 0x0300 = version 3.00 |
| 0x0002 | Unit address | 17–24, as set by the switches |
| 0x0003 | Capabilities | 0x001F — see below |

| Capability Bit | Meaning |
|---|---|
| 0x0001 | Analog output (DAC) |
| 0x0002 | PWM output |
| 0x0004 | Analog input |
| 0x0008 | 8 digital inputs, 6 digital outputs |
| 0x0010 | Step/direction axis |

**NOTE**
Capability bits are product-specific — the Tstep-087 uses bit 0 for motion, this card for the DAC. Check the product ID first.

## 6.2 Motion Parameters (Read/Write, 0x0200–0x020B)

Identical to the Tstep-087 drive.

| Registers | Name | Units | Min | Max |
|---|---|---|---|---|
| 0x0200–0x0201 | Acceleration | steps/s² | 1 | 1,000,000 |
| 0x0202–0x0203 | Start speed | steps/s | 1 | 100,000 |
| 0x0204–0x0205 | Cruise speed | steps/s | 1 | 100,000 |
| 0x0206–0x0207 | Move distance | steps | 0 | 2,147,483,647 |
| 0x0208–0x0209 | Dwell | ms | — | not implemented |
| 0x020A | Command | — | — | writing this executes |
| 0x020B | Command sequence | — | — | see Chapter 9 |

Parameters are staged, not applied — writing one does nothing until a command is written to 0x020A. This is why a 32-bit value is safe to write as two registers: a half-updated value just sits in staging.

Values are clamped, not refused — an out-of-range value is silently clamped at execute time; read it back to check. Acceleration is honoured exactly in range. Start speed is where a ramp begins and ends — deceleration returns to it, not to zero.

**NOTE**
Max step rate is 100,000 steps/s.

⚠ **WARNING**
Dwell registers (0x0208–0x0209) do nothing — accepted and read back, but unused. State bit 0x0004 is never set. Time any pause yourself.

Command 8 saves acceleration, start speed, cruise speed and distance to flash; restored at power-up. Before any save, the card uses built-in defaults. Dwell is not saved.

**CAUTION**
Save is refused while moving (exception 03). Stop the axis (state bit 0 clear), then save. Takes ~35 ms — allow for it in the timeout. Save a finished profile, not on every change.

## 6.3 Motion Status (Read Only, 0x0300–0x0305)

Identical to the Tstep-087 drive.

| Registers | Name | Units |
|---|---|---|
| 0x0300–0x0301 | Current speed | steps/s |
| 0x0302–0x0303 | Position | steps, signed |
| 0x0304 | State word | bitmask, below |
| 0x0305 | Seek inputs | bit 0 = IN1, bit 1 = IN2 |

Six contiguous registers — one 0x03 read of quantity 6 returns the axis's entire state. Position counts every step in every mode.
Register 0x0305 is the debounced state of inputs 1–2 — what seek logic uses. The raw 8-input port at 0x0500 may briefly differ during contact bounce.

| Bit | Meaning |
|---|---|
| 0x0001 | Moving |
| 0x0002 | Direction is clockwise |
| 0x0004 | Waiting on a dwell — never set |
| 0x0008 | Analog input is driving the speed (pot mode) |
| 0x0010 | Driver fault — never set on this card, no driver error input |

## 6.4 Card Control (Read/Write, 0x0400–0x0406)

| Register | Name | Range | Units |
|---|---|---|---|
| 0x0400 | DAC level | 0–100 | % |

| | | | |
|---|---|---|---|
| 0x0401 | DAC enable | 0 or 1 | — |
| 0x0402 | PWM frequency | 100–10000 | Hz |
| 0x0403 | PWM duty | 1–999 | per mille |
| 0x0404 | PWM enable | 0 or 1 | — |
| 0x0405 | Output port | 0–63 | bit n = output n+1, OUT1–OUT6 only |
| 0x0406 | Status LED | 0–100 | % |

A value outside its range returns exception 03 and leaves the register unchanged.

> ⚠ **WARNING**
> Output port is six bits now, not eight — OUT7/OUT8 are the direction and step pins. Writing bit 6 or 7 returns exception 03.

## 6.5 Card Status (Read Only, 0x0500–0x0503)

| Register | Name | Units |
|---|---|---|
| 0x0500 | Input port | bit n = input n+1, all eight |
| 0x0501 | Output port readback | bitmask, six bits |
| 0x0502 | Analog input | mV |
| 0x0503 | CPU temperature | °C |

These four registers are adjacent — one 0x03 read of quantity 4 fetches the entire I/O state.

## 6.6 Unmapped Registers

0x0004–0x01FF, 0x020C–0x02FF, 0x0306–0x03FF, 0x0407–0x04FF and 0x0504 upward are unmapped. Reading one returns exception 02, never zero.

# Chapter 7 — Motion Commands (Write to 0x020A)

Same command numbers as the Tstep-087. This card implements a subset — unsupported numbers are refused, never reused.

## 7.1 Command Table

| Value | Command | Uses |
|---|---|---|
| 0 | Stop — ramps down, never a hard stop | — |
| 1 | Move clockwise | distance, speeds, acceleration |
| 2 | Move anticlockwise | distance, speeds, acceleration |
| 3 | Run clockwise until stopped | speeds, acceleration |
| 4 | Run anticlockwise until stopped | speeds, acceleration |
| 5 | Seek input 1 | speeds, acceleration |
| 6 | Seek input 2 | speeds, acceleration |
| 7 | Reset position counter to zero | — |
| 8 | Save motion parameters to flash | refused while moving — exception 03 |
| 13 / 14 | Pot mode enable / disable | analog input as speed source |
| 17 / 18 | Driver enable / disable | not on this card — exception 03 |
| 19 | Clear a latched driver fault | not on this card — exception 03 |
| 20 | Apply the staged cruise speed | retargets a move already running |

An unrecognised command returns exception 03. Commands 17, 18 and 19 exist on the Tstep-087, not here — this card has no driver-enable line or driver-error input, so they are refused rather than silently ignored.

## 7.2 Waiting for a Move to Finish

Poll register 0x0300 quantity 6 and watch bit 0 of the state word. Polling every 50–100 ms is ample.

## 7.3 Changing Speed While Moving

Stage a new cruise speed in 0x0204–0x0205, then write command 20 — the axis ramps to it at the configured acceleration.
Command 20 is refused in pot mode — the analog input would overwrite the new target within milliseconds.

## 7.4 Changing Direction While Moving

Command 3 or 4 during a continuous run is a reversal — the axis winds down, reverses at zero speed, and ramps back up. No separate stop needed; position tracks correctly.
A reversal during a finite move is refused (exception 03). Stop the move, then command the new direction.

## 7.5 Pot Mode — Commands 13 and 14

In pot mode, the analog input scales the cruise speed register — the knob spans zero to the configured maximum and can never exceed it. The staged cruise speed itself is never overwritten.
With pot mode off, the analog input is a plain input again. It stays readable at 0x0502 in both modes.
A knob move goes through the normal ramp, like any speed change.

> ⚠ **WARNING**
> Turning the knob to its minimum does not stop the axis — it commands 1 step/s and the moving bit stays set. Only command 0 stops a move.

> **NOTE**
> Disabling pot mode does not restore the staged cruise speed. The axis stays at whatever the knob last asked for until command 20 or the next move applies the staged value.

## 7.6 Seeks Have No Distance Limit or Timeout

> ⚠ **WARNING**
> Commands 5/6 run until the target input asserts — nothing else stops them. If not wired, not powered, or already past, the axis runs forever. Set your own timeout or travel limit and send command 0 to end it. Confirm register 0x0305 actually changes before relying on a seek.

# Chapter 8 — Exceptions

## 8.1 Exception Codes

| Code | Meaning | When |
|---|---|---|
| 01 | Illegal function | Any function outside the four listed |
| 02 | Illegal data address | An unmapped register, or a range that runs into one |
| 03 | Illegal data value | Quantity of zero, over the per-function limit, a value outside a register's range, a command this card does not implement, or a byte count contradicting the quantity |

- A multi-register write touching anything unwritable writes nothing — the whole range is checked first, so there's never a partial write.
- An unmapped register is an exception, never a zero.

## 8.2 When the Card Does Not Answer

Silence is deliberate for:

- a frame addressed to another unit;
- a frame with a bad CRC — the address can't be trusted, so replying risks the wrong card answering;
- any broadcast (address 0) — executed but never acknowledged;
- a malformed frame: wrong length for its function, or longer than 64 bytes.

> **NOTE**
> Silence means "not for me", "corrupted", or "nothing there". Treat a timeout as a retryable communications failure, not a device state.

# Chapter 9 — Reliable Communication

## 9.1 Recommended Master Settings

| Setting | Value | Why |
|---|---|---|
| Response timeout | 100 ms | About 20× the measured round trip |
| Retries | 2, then report | The card is silent on CRC failure, so a corrupted frame looks like a timeout |
| Inter-frame delay | ≥ 5 ms | Must exceed the card's 3 ms end-of-frame gap |
| Frame transmission | One write call | A pause inside a frame splits it, and both halves fail CRC |

## 9.2 Retrying a Move — the Sequence Register

Retrying I/O register writes has always been safe — writing a value twice leaves the same result as once.
Motion is different — a lost reply looks like a lost command, so blindly resending a move runs it twice. Register 0x020B solves this. Before each command, write a sequence number; then write the command:

```
1.  0x06  0x020B  value N     the sequence number for this command
2.  0x06  0x020A  value cmd   the command
```

If the sequence was already executed, the card acknowledges but does not repeat it. A retry is just resending both writes — the result is exactly one execution either way.
Reading 0x020B returns the last executed sequence — check it after a timeout without risking a second move.

- The number is opaque — only ensure two different commands carry different numbers.
- A refused command does not claim its sequence, so it stays retryable once you fix the cause.
- Not using the register at all is legitimate and means "always execute".
- The guard only applies to a sequence written since the last command — forgetting to update it risks a double execution at worst, never a lost command.

## 9.3 Probing at the Wrong Baud Rate

A master may scan both baud rates to find a card. Wrong-rate traffic just looks like noise and is discarded; the next correctly framed request is answered normally.

## 9.4 Performance

| Transaction | Round Trip |
|---|---|
| Read 4 registers | 5.03 ms |
| Read 1 register | 4.55 ms |
| Write single register | 4.60 ms |

Measured over USB-RS485 at 115,200, 200 transactions each. Batch reads — most of a transaction is fixed overhead: four single reads cost 18.2 ms vs 5.0 ms batched (3.6× faster).
Budget ~5 ms per transaction at 115,200, ~16 ms at 19,200, shared across the bus. Polling faster than ~50 ms per card just saturates the bus.

> **NOTE**
> Check your USB adapter’s latency timer — FTDI defaults to 16 ms, which dominates the figures above. Set it to 1 ms before measuring anything.

# Chapter 10 — Worked Examples

Captured from a real card at address 17, firmware 3.00. Reproduce these frames byte for byte to confirm your CRC and framing.

## 10.1 Read the Identity Block

```
->  11 03 00 00 00 04 46 99
<-  11 03 08 08 08 03 00 00 11 00 1F 58 8F
```

## 10.2 Read the Motion Parameter Block

```
->  11 03 02 00 00 08 47 24
<-  11 03 10 00 00 07 D0 00 00 00 C8 00 00 03 20 00 00 06 40 AD AD
    (acceleration 2000, start speed 200, cruise speed 800, distance 1600)
```

## 10.3 Stage Cruise Speed and Distance

```
->  11 10 02 04 00 04 08 00 00 03 20 00 00 06 40 D3 5F
<-  11 10 02 04 00 04 83 23          start address and quantity echoed, not
the data
```

## 10.4 Read the Staged Values Back

```
->  11 03 02 04 00 04 06 E0
<-  11 03 08 00 00 03 20 00 00 06 40 42 B3
```

## 10.5 Command a Move with a Sequence Number

```
->  11 06 02 0B 00 4D 3B 15      sequence = 77
<-  11 06 02 0B 00 4D 3B 15
->  11 06 02 0A 00 01 6B 20      command 1, move clockwise
<-  11 06 02 0A 00 01 6B 20      request echoed — that is the
acknowledgement
```

## 10.6 Poll the Axis Until It Stops

```
->  11 03 03 00 00 06 C7 1C
<-  11 03 0C 00 00 00 00 00 00 06 40 00 02 00 00 23 15
    (speed 0, position 1600, state 0x0002 — stopped, clockwise, no seek
inputs asserted)
```

## 10.7 Confirm Which Command Executed

```
->  11 03 02 0B 00 01 F6 E0
<-  11 03 02 00 4D B9 B2          sequence 77 — the move above
```

## 10.8 Read the Card's I/O State

```
->  11 03 05 00 00 04 46 55
<-  11 03 08 00 00 00 00 00 00 20 C0 CF
    (inputs 0x0000, outputs 0x0000, analog 0 mV, CPU 32 °C)
```

## 10.9 Write the Output Port

```
->  11 06 04 05 00 3F DA 7B      all six outputs on
<-  11 06 04 05 00 3F DA 7B
->  11 06 04 05 00 FF DA 2B      0xFF asks for eight
<-  11 86 03 03 A4                code 03 — OUT7 and OUT8 are the axis
```

## 10.10 Set the Status LED

```
->  11 06 04 06 00 2A EB B4      42 %
<-  11 06 04 06 00 2A EB B4
```

## 10.11 Pot Mode

```
->  11 06 02 0A 00 0D 6B 25      command 13, pot mode on
<-  11 06 02 0A 00 0D 6B 25
->  11 06 02 0A 00 14 AA EF      command 20 while pot mode owns the speed
<-  11 86 03 03 A4                code 03 — the pot would overwrite it
->  11 06 02 0A 00 0E 2B 24      command 14, pot mode off
<-  11 06 02 0A 00 0E 2B 24
```

## 10.12 Saving Motion Parameters

Staged with two writes, because Write Multiple Registers is capped at four registers:

```
->  11 10 02 00 00 04 08 00 00 10 E1 00 00 00 DE 9F EA
<-  11 10 02 00 00 04 C2 E2       accel 4321, start 222
->  11 10 02 04 00 04 08 00 00 0D 05 00 01 56 66 73 AC
<-  11 10 02 04 00 04 83 23       cruise 3333, steps 87654
->  11 06 02 0A 00 08 AB 26      command 8, save to flash
<-  11 06 02 0A 00 08 AB 26       accepted, ~35 ms
```

Refused while the axis is moving:

```
->  11 06 02 0A 00 03 EA E1      command 3, run clockwise
<-  11 06 02 0A 00 03 EA E1
->  11 06 02 0A 00 08 AB 26      command 8 while moving
<-  11 86 03 03 A4                code 03 — stop the axis first
```

## 10.13 Unimplemented Commands and Exceptions

```
->  11 06 02 0A 00 11 6A EC      command 17, driver enable
```

```
<-  11 86 03 03 A4                code 03
```

```
->  11 03 99 99 00 01 78 29      an unmapped register
<-  11 83 02 C1 34                function 0x03 | 0x80, code 02
->  11 03 00 00 00 09 87 5C      nine registers, over the limit of eight
<-  11 83 03 00 F4                code 03
->  11 2B 0E 01 00 00 74 74      Read Device Identification
<-  11 AB 01 9F 35                code 01, not supported
->  11 06 05 00 00 01 4A 56      writing to a read-only status register
<-  11 86 02 C2 64                code 02
```

# Chapter 11 — Not Supported

Be explicit about these when writing a master — none of them will work:

- Any function code other than 0x03, 0x04, 0x06, 0x10.
- Read Device Identification (0x2B/0x0E) — use registers 0x0000–0x0003.
- Reads over 8 registers, writes over 4 registers.
- Frames longer than 64 bytes.
- Any reply to a broadcast.
- Changing the address or baud rate over the bus — both are switch-only, read at power-up.
- Outputs 7 and 8 as general-purpose outputs — they are the direction and step pins.
- Saving anything other than the four motion parameters — I/O state, pot mode and dwell are not stored.
- Saving while the axis is moving — stop it first.
- A dwell — the registers exist but nothing acts on them.
- Driver enable/disable and fault clearing — no such signals on this card.
- Selectable word order for 32-bit values — high word first, always.

# Chapter 12 — Troubleshooting

Use this table to diagnose and resolve common problems. If the issue persists, contact Tiny Controls support.

| Problem | Possible Cause | Solution |
|---|---|---|
| No response from card | Wrong address in the request; baud mismatch; bus not terminated; cable fault | Verify the request address (0x11–0x18) matches S1–S3. Check S4 baud (19,200/115,200) matches the host. Add a 120 Ω terminator at the last device. Use twisted-pair cable for D+/D–. |
| Every request times out even though wiring and baud look correct | CRC computed but sent high-byte-first | The CRC is the only field sent low byte first — verify the byte order of the last two bytes of the frame. |
| A move appears to run twice after a timeout and retry | Master resent the move command blindly | Use the sequence register 0x020B (section 9.2) so a retried command executes at most once. |
| Axis never stops during a seek | Target input not wired, not powered, or already passed | Add a master-side timeout or travel budget and send command 0. Confirm register 0x0305 changes at all. |
| Command 8 (save) returns exception 03 | Axis was moving when the save was requested | Stop the axis, confirm state word bit 0 is clear, then save. |
| Read returns exception 02 | Register address is unmapped, or a multi-register read runs into a hole | Check the register map in Chapter 6; unmapped registers never return zero. |
| Write returns exception 03 | Value outside range, quantity over the per-function limit, or an unimplemented command number | Check the range/limit tables in Chapters 4, 6 and 7. |
| DAC analog output has no voltage | Isolated 10 V supply not connected; DAC not enabled | Connect the isolated 10 V supply. Write register 0x0401 = 1 before setting the duty. |
| PWM output not present | PWM not enabled, or frequency/duty not set | Write 0x0402 (frequency), then 0x0403 (duty), then 0x0404 = 1, in that order. |

# Chapter 13 — Warranty and Support

## 13.1 Warranty

Tiny Controls Private Limited warrants the TIO-0808 against defects in materials and workmanship for 12 months from the date of purchase under normal use conditions.
The warranty does not cover:

- Damage from connecting the isolated supply input to the 24 V supply.
- Damage from applying more than 10 V to the analog input.
- Damage from exceeding relay contact or output current ratings.
- Damage from miswiring an external stepper driver's STEP/DIR inputs to OUT7/OUT8.
- Physical damage including bent terminals, cracked PCB, or liquid ingress.
- Damage from unauthorised modifications or repairs.

## 13.2 Obtaining Support

Before contacting support, have the following ready:

- Product model: TIO-0808
- Firmware version (register 0x0001)
- DIP switch settings (S1–S4) and selected address
- The Modbus RTU requests being sent and the responses received (in hex)
- A description of the problem

| Contact Method | Details |
|---|---|
| Website | www.tinycontrols.com (download demo software here) |
| Email | anil@tinycontrols.com |
| Phone | +91-991-119-3210 / +91-83840-24665 |
| Address | Tiny Controls Private Limited, B-17/A, Nishat Park, Kakrola, New Delhi, India – 110078 |

End of Document