TINY CONTROLS PRIVATE LIMITED
Advanced CNC Systems

TSTEP-087

Stepper Drive Controller

Modbus RTU over RS-485

USER MANUAL

Document No.	TSTEP-087-MODBUS
Document Rev.	2.0
Product Rev.	1.0
Firmware	2.00
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 this chapter before wiring, configuring, or operating the Tstep-087.

1.1 Warning Levels

This manual flags hazards and important operating notes at four levels, shown throughout in the same visual format:

Level	Meaning
DANGER	A hazard that will cause death or serious injury if not avoided.
WARNING	A hazard that can cause death or serious injury if not avoided.
CAUTION	A hazard that can cause minor injury or equipment damage if not avoided.
NOTE	Operational information carrying no injury hazard.

1.2 Electrical Safety

> ⚠ **WARNING**
> Disconnect AC power at its source before wiring the drive, the motor, or the DC supply. Do not work on live connections.

The on-board driver is powered from an external DC supply — the wiring diagram in Chapter 3 shows a 48V DC SMPS as a typical example — never from AC mains directly.

> ⚠ **WARNING**
> Bond the DC supply's earth terminal and the AC supply's protective earth before applying power. An unearthed supply is a shock hazard.

> **CAUTION**
> Set the phase current switches (S1–S3, Chapter 4) to the connected motor's rated current or below. A setting above the motor's rating overheats it.

The drive latches an alarm on over-current, over-temperature, or short-circuit at the on-board driver (Chapter 11). This protects the driver, not the operator — treat a latched fault as a signal to de-energize the system and inspect the motor and wiring before resuming.

> **NOTE**
> Connect the RS-485 shield or signal common to the drive's ground reference and terminate the bus at both physical ends (section 3.1).

1.3 Motion Safety

> **DANGER**
> A stop sent over Modbus is not a safety function. It travels over a wire, through a UART, into a polled loop, and depends on the drive still being able to respond. The hardwired emergency-stop chain is what makes a machine safe — see Chapter 12 for the full

discussion.

The drive executes a valid command as soon as it receives it — there is no separate confirmation step. Keep clear of the motor shaft and any attached mechanism whenever the system is powered.

> ⚠ **WARNING**
> Seeks have no distance limit and no timeout. If the seek input is not wired, not powered, or the axis is already past it, the motor runs indefinitely. Wire and verify every seek input before relying on it.

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

> **NOTE**
> The position counter starts at zero on every power-up; it is not stored across a power cycle. Re-establish absolute position (for example, with a seek to a limit) after every power loss before relying on it.

A direction reversal during a finite move is refused, not queued — the motor keeps running in its original direction. Stop the move first, then command the new direction.

1.4 General Safety Rules

- Read this manual, in particular Chapter 4 (Configuration Switches) and Chapter 8 (Register Map), before wiring or commissioning the drive.
- Only qualified personnel should wire, configure, or service the drive.
- Do not bypass or defeat limit switches or the emergency-stop chain to work around a fault — find and fix the cause instead.
- Verify the configuration switches (address, baud rate, phase current) match the intended installation before connecting a motor.
- Keep this manual accessible to anyone commissioning, operating, or servicing the drive.
- After any driver fault, inspect the wiring and motor condition before clearing the alarm and resuming motion.

Chapter 2 — Overview

The Tstep-087 is a Modbus-controlled stepper drive. The motion controller and the stepper driver are on one board: it receives commands over RS-485 using the Modbus RTU protocol and drives the motor directly.

Feature	Specification
Control interface	Modbus RTU over RS-485, 2-wire half duplex
Baud rate	19200 or 115200, switch selected
Unit address	1–16, switch selected
Motor output	On-board stepper driver, bipolar
Phase current	0.6–7.0 A in 8 steps, switch selected
Resolution	10 microsteps — 2000 steps per revolution, fixed
Maximum step rate	100,000 steps/s (3000 rev/min)
Acceleration range	1–1,000,000 steps/s²
Move distance range	1–2,147,483,647 steps
Position counter	Signed 32-bit, steps
Digital inputs	IN1, IN2 — active low, internally pulled up
Analog input	Potentiometer, speed control
Indicator	Single status LED
Parameter storage	On-board flash, reloaded at power-up

Motion types supported: finite moves of a commanded distance, continuous runs, and seeks that runs until a digital input triggers. Speed and direction can both be changed while the motor is running.

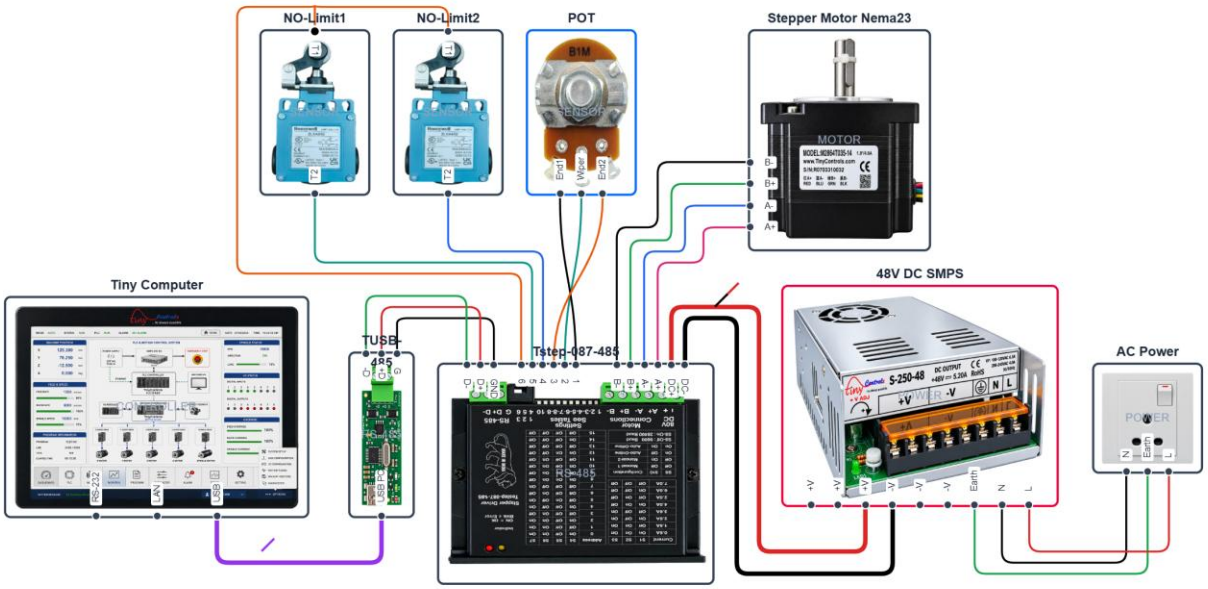
Chapter 3 — Installation and Wiring

3.1 Connections

Signal	Direction	Function
D+, D-	Bidirectional	RS-485 differential pair.
A+ A- B+ B-	Output	Stepper motor phases, driven by the on-board driver.
DC+ DC-	Input	External DC supply to the on-board driver (see Figure 3.1).
IN1	Input	General-purpose input, active low, internally pulled up.
IN2	Input	General-purpose input, active low, internally pulled up.
POT	Input	Potentiometer wiper, speed control.

NOTE
Connect the RS-485 shield or signal common to the drive's ground reference. Terminate the bus at both physical ends.

Figure 3.1 — Electrical Connection Diagram



3.2 Resolution

The on-board driver is fixed at 10 microsteps, giving 2000 steps per revolution with a standard 1.8° motor. Every distance, speed, and position on the bus is expressed in these steps:

Quantity	Conversion

Distance	Revolutions × 2000 = steps
Speed	Rev/min × 33.33 = steps/s
Position	Steps ÷ 2000 = revolutions

One revolution is 2000 steps, 60 rev/min is 2000 steps/s, and the 100,000 steps/s ceiling corresponds to 3000 rev/min. The speed a motor actually reaches without stalling depends on the motor and the load.

3.3 Digital Inputs

IN1 and IN2 are pulled up internally and read as asserted when driven low — by a switch, a sensor output, or a limit flag. Both are debounced for 11 ms; a pulse shorter than that is not seen.
Their live state is reported in register 0x0305, bit 0 for IN1 and bit 1 for IN2, where 1 means asserted. Commands 5 and 6 use them as seek targets.

3.4 Potentiometer

The potentiometer is sampled continuously and averaged, and the result is applied every 100 ms. It is only acted on while potentiometer control is enabled with command 13.
While enabled it scales the cruise speed register:

```
live speed = cruise speed × potentiometer / 1023
```

The potentiometer reading ranges from 1 to 1023, so the commanded speed never falls below 1 step/s.

3.5 Status LED

LED	Meaning
On, steady	Normal operation
Flashing, 5 Hz	Driver fault latched — see Chapter 11

3.6 Power-Up Behaviour

1. Loads the saved parameters from flash.
2. Reads the configuration switches.
3. Enables the driver — the motor is energised and holding.
4. Waits 1 second, then begins answering the bus.

> ⚠ **WARNING**
> The drive does not respond to Modbus requests for approximately 1 second after power-up or reset. A master polling during that window will time out.

> **NOTE**
> The position counter starts at zero on every power-up. It is not stored in flash.

Chapter 4 — Configuration Switches

Switches are listed left to right, in the order they appear on the switch block.

Switches	Function	When a Change Takes Effect
S1–S3	Motor current	Immediately
S4–S7	Unit address	At the next power-up
S8	Baud rate	At the next power-up
S9–S10	Unused	—

4.1 Motor Current — S1 to S3

The three switches select the phase current delivered to the motor. A switch turned off adds its weight, and S1 carries the lowest weight.

S1	S2	S3	Current
on	on	on	0.6 A
off	on	on	1.6 A
on	off	on	2.6 A
off	off	on	3.6 A
on	on	off	4.0 A
off	on	off	5.0 A
on	off	off	6.0 A
off	off	off	7.0 A

NOTE

S1 to S3 take effect immediately. No power cycle is needed, and a change made while the motor is running alters the phase current at once. Set the current with the motor stopped.

CAUTION

Set the current to the motor’s rated phase current or below. A setting above the motor’s rating overheats it.

⚠ **WARNING**

All three switches off is 7.0 A, the maximum. Set the current before connecting a motor rated below that.

4.2 Address — S4 to S7

Turning a switch on adds its weight:

```
Address = 1 + (1·S4 + 2·S5 + 4·S6 + 8·S7)
```

Address	S4	S5	S6	S7
1	off	off	off	off
2	on	off	off	off
3	off	on	off	off
4	on	on	off	off
5	off	off	on	off
6	on	off	on	off
7	off	on	on	off
8	on	on	on	off
9	off	off	off	on
10	on	off	off	on
11	off	on	off	on
12	on	on	off	on
13	off	off	on	on
14	on	off	on	on
15	off	on	on	on
16	on	on	on	on

All four off is address 1. All four on is address 16.

4.3 Baud Rate — S8

S8	Baud
off	19200
on	115200

4.4 S9 and S10

Unused. Their position has no effect.

4.5 Changing the Address or Baud Rate

> **NOTE**
> S4 to S8 are read once at power-up. Change a switch, then power-cycle the drive. Neither setting can be changed over the bus.

Chapter 5 — Modbus RTU in Brief

Modbus RTU is a request-response protocol on a shared serial line.

- One master, many slaves. The master sends a request; the addressed slave replies. Slaves never transmit unless asked. This drive is always a slave.
- Each slave has a unique address, 1 to 247. Address 0 is the broadcast address: every slave executes the request and none replies.
- Data is organised as 16-bit registers, each with a 16-bit address. A request names a starting register and a quantity.
- A function code selects the operation — read registers, write one register, write several.
- Every frame is checksummed with a CRC. A slave that receives a frame with a bad CRC sends nothing at all.
- Frames are separated by silence. There is no start or end marker; a gap of at least 3.5 character times on an otherwise idle line ends a frame.
- A slave that cannot carry out a request replies with an exception: the function code with its top bit set, followed by a one-byte code.

Exception	Meaning	Raised When
01	Illegal function	The function code is not supported
02	Illegal data address	The register does not exist, or is read-only
03	Illegal data value	The value or quantity is out of range, or the command was refused

NOTE

Modbus carries no timing or priority. A request is served when it arrives; there is no queue and no way to pre-empt one transaction with another.

Chapter 6 — Frame Format and CRC

Every request and reply has the same layout:

```
[ address ][ function ][ data ... ][ CRC low ][ CRC high ]
```

- Maximum frame length is 64 bytes. Longer frames are discarded.
- Send each frame in a single write call. A pause of more than 3 ms inside a frame ends it early and both fragments fail the CRC check.
- Leave at least 5 ms between frames.

6.1 CRC-16/MODBUS

Polynomial 0xA001 (reflected), initial value 0xFFFF, no final XOR. Computed over every byte from the address through the last data byte, excluding the CRC itself.

> ⚠ **WARNING**
> The CRC is the one field sent low byte first. Register addresses, quantities, and register values are all big-endian, high byte first.

```
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;
}

// appending it to a frame buffer of length n (address .. last data byte)
uint16_t crc = crc16_modbus(frame, n);
frame[n]     = (uint8_t)(crc & 0xFF);   // low byte first
frame[n + 1] = (uint8_t)(crc >> 8);
```

For the frame 01 03 00 00 00 04 the CRC value is 0x0944, sent as 44 09, giving the complete frame 01 03 00 00 00 04 44 09.

> **NOTE**
> The standard CRC-16/MODBUS check value for the ASCII string "123456789" is 0x4B37 — use it to test an implementation independently.

Chapter 7 — Function Codes

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

> **NOTE**
> Any other function code returns exception 01.

7.1 32-Bit Values

Steps, speeds, and positions are 32-bit. Each occupies two consecutive registers, starting on an even address, high word first:

```
value = (register[n] << 16) | register[n+1]
```

The word order is fixed and not configurable. Position is signed, two’s complement across the pair; every other 32-bit value is unsigned.

> **NOTE**
> The 4-register write limit means one 0x10 transaction carries two 32-bit values.

Chapter 8 — Register Map

8.1 Identity Registers (Read Only)

Register	Name	Value
0x0000	Product ID	0x0087
0x0001	Firmware version	0x0200 = version 2.00
0x0002	Unit address	1–16, as set by the switches
0x0003	Capabilities	0x0007 — bit 0 motion, bit 1 potentiometer, bit 2 IN1/IN2

8.2 Motion Parameters (Read/Write)

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 13

NOTE
Parameters are staged. Writing one changes nothing about the motor. Staged values are applied when a command is written to 0x020A.

NOTE
Values are clamped, not refused. A parameter outside its range is brought into range when a command is executed. Read the register back to see what was accepted.

Acceleration applies to both ramping up and ramping down. The start speed is the speed at which a ramp begins and ends; deceleration returns to the start speed, not to zero. If the start speed is set above the cruise speed, the axis runs flat at the cruise speed.

⚠ **WARNING**
The dwell registers do nothing. 0x0208–0x0209 accept, clamp, and read back a value, but nothing acts on it, and state word bit 0x0004 is never set. A master needing a pause between moves must time it itself.

8.3 Live Status (Read Only)

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

NOTE
Six contiguous registers in polling order — one 0x03 request with quantity 6 returns the drive's entire live state.

Current speed is the instantaneous ramp speed, and reads 0 when idle. Position counts every step in every mode: finite moves, continuous runs, and seeks alike. It increments clockwise and decrements anticlockwise.

Bit	Meaning
0x0001	Moving
0x0002	Direction is clockwise
0x0004	Waiting on a dwell — never set
0x0008	Potentiometer speed control enabled
0x0010	Driver fault latched — see Chapter 11

NOTE
While bit 0x0010 is set, no other bit is reported.

8.4 Unmapped Registers

0x0004–0x01FF, 0x020C–0x02FF, and 0x0306 upward return exception 02. Writing to a read-only register also returns exception 02.

Chapter 9 — Commands — Write to 0x020A

Value	Command	Parameters Used
0	Stop — ramps down to the start speed, then halts	—
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 parameters to flash	—
13	Potentiometer control enable	—
14	Potentiometer control disable	—
17	Driver enable	—
18	Driver disable	—
19	Clear a latched driver fault	—
20	Apply the staged cruise speed	cruise speed

NOTE
An unrecognised command returns exception 03. A command is acknowledged by echoing the request. An exception 03 means the command was refused and nothing changed.

9.1 When Commands Are Refused

Command	Refused with Exception 03 When
1, 2	The motor is moving, a fault is latched, or the distance is 0
3, 4	A finite move is running, or a fault is latched
5, 6	The motor is moving, a fault is latched, or the target input is already asserted
8	The motor is moving
19	The driver is disabled, or the fault is still present
20	Potentiometer control is enabled

Commands 0, 7, 13, 14, 17, and 18 are always accepted.

Chapter 10 — Operation

10.1 Commanding a Move

5. Write the acceleration, start speed, cruise speed, and distance.
6. Write the command to 0x020A.
7. Poll 0x0300 quantity 6 and watch bit 0 of the state word.

> **NOTE**
> Polling every 50–100 ms is sufficient.

10.2 Changing Speed While Moving

Stage a new cruise speed in 0x0204–0x0205, then write command 20. The drive ramps to the new speed at the configured acceleration. This works during finite moves and continuous runs alike, and sets the speed for the next move when idle.

> **NOTE**
> Command 20 is refused while potentiometer control is enabled.

10.3 Changing Direction While Moving

Sending command 3 or 4 during a continuous run in the opposite direction reverses the axis: the drive ramps down to the start speed, changes direction at zero speed, and ramps back up. No stop command is needed, and position tracks correctly throughout.

> ⚠ **WARNING**
> A reversal during a finite move is refused. Stop the move first, then command the new direction.

10.4 Stopping

Command 0 ramps down at the configured acceleration and halts. There is no hard stop.

10.5 Seeks

Commands 5 and 6 run continuously until IN1 or IN2 respectively asserts, then halt immediately. A seek is refused if its target input is already asserted.

> ⚠ **WARNING**
> Seeks have no distance limit and no timeout. Nothing other than the input stops them. If the input is not wired, not powered, or the axis is already past it, the motor runs indefinitely. A master issuing a seek must impose its own timeout or travel budget and send command 0 when it expires.

10.6 Potentiometer Control

Command 13 enables potentiometer control, command 14 disables it. While enabled, the potentiometer sets the live speed target every 100 ms, scaled against the cruise speed register. State word bit 0x0008 reports that it is enabled.

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

10.7 Driver Enable and Disable

Command 18 disables the driver and stops any move in progress. The motor is left without holding torque. Command 17 re-enables it. Fault detection is suspended while the driver is disabled and for 100 ms after it is re-enabled.

10.8 Saving Parameters — Command 8

Stores the acceleration, start speed, cruise speed, and move distance to flash. They are reloaded at every power-up. The dwell registers and the position counter are not stored.

> **CAUTION**
> Refused with exception 03 while the motor is moving.

> ⚠ **WARNING**
> Allow 50 ms for the reply to this command, against 5 ms for every other transaction. A master with a shorter timeout will report a failure for a save that succeeded.

Values in flash are range-checked at power-up. A stored value above its maximum is replaced: acceleration and start speed revert to their minimum, cruise speed and move distance to their maximum. On a drive that has never been saved, this gives:

Parameter	Value at First Power-Up
Acceleration	1 step/s²
Start speed	1 step/s
Cruise speed	100,000 steps/s
Move distance	2,147,483,647 steps

> **NOTE**
> Write and save a working set of parameters before commanding the first move.

Chapter 11 — Driver Faults

The drive monitors the on-board driver for faults — over-current, over-temperature, and short circuit. When a fault is reported continuously for 50 ms the drive latches an alarm and sets state word bit 0x0010.
While the alarm is latched:

- Motion is locked out. Commands 1 to 6 return exception 03.
- Reads continue to work. Position, parameters, and status remain available.
- The status LED flashes at 5 Hz.
- No other state word bit is reported.

> **NOTE**
> Fault detection is armed only while the driver is enabled, and only after a 100 ms settling period following an enable. Disabling the driver with command 18 does not raise an alarm.

Clear the alarm with command 19. The command is refused with exception 03 while the fault is still present or while the driver is disabled. A successful clear means the driver has stopped reporting the fault.

> **CAUTION**
> When this bit is set, check the motor supply, the motor wiring, and the ambient temperature before resuming.

Chapter 12 — Safety

- The bus is the only source of commands. The drive is always online.
- The drive answers while it is moving. Position reads and stop commands work mid-move.

> ⚠ **WARNING**
> A stop over the bus is not a safety function. It travels over a wire, through a UART, into a polled loop. The hardwired emergency-stop chain remains what makes a machine safe.

> **NOTE**
> Modbus has no priority. Send a stop as its own transaction rather than queued behind parameter writes.

> ⚠ **WARNING**
> Seeks and potentiometer control both run without limits. See the warnings in Chapter 10.

Chapter 13 — Master Implementation

13.1 Recommended Settings

Setting	Value
Response timeout	100 ms — 500 ms for command 8
Retries	2, then report the failure
Inter-frame delay	≥ 5 ms
Frame transmission	One write call per frame

> **NOTE**
> The drive is silent on a CRC failure, so a corrupted frame is indistinguishable from a timeout.

13.2 Retrying a Command — the Sequence Register

> ⚠ **WARNING**
> A retried move executes twice unless the sequence register is used. A lost reply cannot be distinguished from a lost command.

Register 0x020B prevents 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
```

A command carrying a sequence number the drive has already executed is acknowledged and not carried out again. A retry is both writes sent again; whichever was lost, the command executes exactly once.

> **NOTE**
> Reading 0x020B returns the sequence number of the last command executed, which confirms after a timeout whether a command got through.

- The number is opaque. Only ensure that two different commands carry different numbers.
- A refused command does not claim its sequence number and stays retryable.
- Not using the register at all is legitimate and means "always execute".
- The guard engages only for a sequence number written since the last command.

13.3 Performance

Measured at 115200 baud, 200 transactions each:

Transaction	Round Trip
Read 6 registers — the whole live state	5.12 ms
Read 1 register	4.28 ms
Write single register	4.35 ms

Budget 5 ms per transaction at 115200, or 16 ms at 19200, shared across every device on the segment. Polling faster than 50 ms per drive gains nothing. Batch reads where possible — most of a transaction is fixed overhead.
Motion does not slow the bus: 16.01 ms idle, 16.18 ms at 300 steps/s, 16.16 ms at 8669 steps/s. The exception is command 8, which takes about 42 ms.

NOTE

Traffic at the wrong baud rate is discarded, and the next correctly framed request is answered normally.

⚠ **WARNING**

Check the USB adapter’s latency timer. FTDI-based adapters default to 16 ms, which dominates every figure above. Set it to 1 ms before measuring.

Chapter 14 — Worked Examples

Frames for a drive at address 1.

14.1 Identify

```
->  01 03 00 00 00 04 44 09
<-  01 03 08 00 87 02 00 00 01 00 07 ...
                00 87    0200    0001    0007
                product  firmware  address   capabilities
```

14.2 Stage Cruise Speed 2000 and Distance 6000 in One Write

```
->  01 10 02 04 00 04 08 00 00 07 D0 00 00 17 70 8E 86
<-  01 10 02 04 00 04 81 B3          start address and quantity echoed, not
the data
```

14.3 Read the Staged Values Back

```
->  01 03 02 04 00 04 04 70
<-  01 03 08 00 00 07 D0 00 00 17 70 5B A6
```

14.4 Command a Clockwise Move

```
->  01 06 02 0A 00 01 69 B0
<-  01 06 02 0A 00 01 69 B0          request echoed
```

14.5 Poll the Whole Live State

```
->  01 03 03 00 00 06 C5 8C
<-  01 03 0C 00 00 00 00 00 00 00 00 00 00 00 93 70
```

> **NOTE**
> An idle drive: speed 0, position 0, state 0x0000, no inputs asserted.

14.6 Stop

```
->  01 06 02 0A 00 00 A8 70
<-  01 06 02 0A 00 00 A8 70
```

14.7 Exceptions

```
->  01 2B 0E 01 00 00 76 E4      Read Device Identification
<-  01 AB 01 9E F0                code 01, function not supported
```

```
->  01 03 99 99 00 01 7A B9      an unmapped register
<-  01 83 02 C0 F1                code 02

->  01 03 00 00 00 09 85 CC      nine registers, over the limit of eight
<-  01 83 03 01 31                code 03

->  01 06 02 0A 00 63 E8 59      command 99, which does not exist
<-  01 86 03 02 61                code 03
```

Chapter 15 — Not Supported

- 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, baud rate, motor current, or microstep resolution over the bus.
- Dwell — the registers exist but nothing acts on them.
- Selectable word order for 32-bit values.
- Storing the position counter through a power cycle.

End of Document