Valtoris.

Integration Guide: Polling Deye / Sunsynk Inverters via Modbus RTU

Resolving the S16 Two's Complement Sign Trap for Bidirectional Power Flows

Document No:	AN-2026-08
Product Series:	Valtoris VT-DTU500 / Edge Gateways
Target Devices:	Deye, Sunsynk, Sol-Ark Hybrid Inverters
Date:	June 2026 (Rev. 1.2)

1. Executive Summary

Deye hybrid inverters (including OEM variants like Sunsynk and Sol-Ark) are highly capable energy storage systems. However, the factory-supplied Wi-Fi loggers route data through remote cloud servers with significant latency, making sub-second zero-export controls or local SCADA integrations impossible.

For deterministic performance, system integrators must poll the internal control processor via its dedicated RS485 Modbus RTU interface. This document outlines the physical wiring requirements and provides critical guidance on decoding the inverter's bidirectional signed integer (S16) payloads.

2. Physical Layer: RS485 Terminal Wiring

Most Deye inverters expose their Modbus RTU interface through an RJ45 port labeled BMS, Meter, or RS485. Establish a 3-wire differential loop with your Master Gateway.

Inverter RJ45 Connector Pins	RS485 Master Terminal	Signal Function
Pin 7 (or Pin 1 on specific models)	**B-** (Data-)	Inverted Differential Signal Line
Pin 8 (or Pin 2 on specific models)	**A+** (Data+)	Non-Inverted Differential Signal Line
Pin 6	**GND**	Reference Signal Ground (Crucial for noise immunity)

> **Warning: Pinout Variations**
>
> Pin definitions vary slightly between single-phase residential units and commercial three-phase models. If initial polling attempts yield timeout errors, verify the specific RS485 A/B pinout in your inverter's installation manual, as they may utilize Pins 1 & 2 instead of 7 & 8.

3. Core Holding Registers

Configure your communications node to: **Baud Rate: 9600, Data Bits: 8, Stop Bits: 1, Parity: None**. The default target Slave ID is 1. Deye maps operational metrics as **Holding Registers (Function Code 0x03)**.

Register (Dec)	Data Format	Parameter Description	Multiplier / Unit
590	S16 (Signed)	Total Active Grid Power (Negative = Exporting)	1 W
586	U16 (Unsigned)	Battery State of Charge (SOC)	1 %
591	U16 (Unsigned)	Battery Voltage Baseline	0.01 V
592	S16 (Signed)	Battery Current Flow (Negative = Discharging)	0.01 A
502	U16 (Unsigned)	Daily PV Generation Accumulator	0.1 kWh

4. Deep Dive: The Two's Complement Sign Trap

Because energy flows bidirectionally (e.g., charging vs. discharging, importing vs. exporting), Deye represents these parameters using **Signed 16-bit integers (S16)** utilizing the two's complement binary system. This is a common source of integration failure.

Hexadecimal Frame Breakdown

Assume the inverter is actively exporting 10 Watts to the utility grid. You poll Total Grid Power (Register 590 / `0x024E`).

```
Master Request (Tx): 01 03 02 4E 00 01 E5 9E

Meter Response (Rx): 01 03 02 FF F6 XX XX

Data Payload: FF F6
```

- **The Error:** If your SCADA software incorrectly parses `0xFFF6` as a standard Unsigned Integer (U16), it converts it to **65,526 Watts**. This massive false spike will instantly trigger alarms or break zero-export control loops.
- **The Solution:** The hex value `0xFFF6` must be explicitly cast as a Signed Integer (S16), which correctly evaluates to **-10 Watts** (indicating export).

5. Integration Architecture: DIY vs. Edge Hardware

A. Software Parsing (Python/Node-RED)

If utilizing a Raspberry Pi with a generic USB-RS485 adapter, your code must programmatically cast the 16-bit register into a signed value.

B. Protocol Normalization (Edge Gateway)

Industrial architectures utilize dedicated Edge Gateways (e.g., Valtoris VT-DTU500) to shift the protocol translation burden to silicon.

- **Native S16 Translation:** Enter register `590` in the configuration UI and select

```
import struct

# Deye exporting 10W returns 0xFFF6
# Python modbus client may return
65526
raw_register_val = 65526

# Pack as unsigned short (H),
# unpack as signed short (h)
power = struct.unpack('h',
struct.pack('H', raw_register_val))
[0]

print(f"Grid Power: {power} W") #
Outputs: -10 W
```

Note: Managing OS-level serial polling across distributed microgrid sites introduces significant scripting maintenance overhead.

"INT16". The gateway inherently handles the two's complement conversion.

- **Cloud Ready JSON:** The hardware continuously polls the inverter and pushes clean, pre-formatted JSON (e.g., `{"GridPower_W": -10}`) directly to your MQTT broker via Ethernet or LTE.

Industrial Microgrid Deployment

To secure deterministic response times and bypass custom scripting maintenance traps, learn more about the protocol bridging engine inside the Valtoris Edge Gateway Series.