Payload Codec Proto - Spec Completeness Audit

Generated: 2026-02-19 11:05

Summary

Metric	Count
Features Complete	31/31
Features Partial	0
Features Missing	0
Requirements Traced	70
Requirements Tested	70
Total Tests	401

Feature Status

Feature	Status	Tests
integer_types	PASS	13
float_types	PASS	13
bool_type	PASS	8
bytes_type	PASS	10
string_type	PASS	9
enum_type	PASS	7
mult_modifier	PASS	6
div_modifier	PASS	4
add_modifier	PASS	2
lookup_modifier	PASS	5
polynomial	PASS	3
compute	PASS	15
guard	PASS	2
transform	PASS	2
ref	PASS	6
switch_match	PASS	20
flagged	PASS	7
bitfield	PASS	7
bitfield_string	PASS	4
byte_group	PASS	3
tlv	PASS	6
nested_object	PASS	7
repeat	PASS	10
ports	PASS	2
definitions	PASS	2
var	PASS	4
skip	PASS	3
endian	PASS	17
unit	PASS	2
semantic	PASS	6
formula	PASS	10

Note: Test counts are approximate due to pattern-matching. Some tests cover multiple features.

Requirements Traceability

Data Types

- REQ-Unsigned-I-001: Unsigned integer types
- REQ-Signed-Int-002: Signed integer types
- REQ-Type-Alia-003: Type aliases (uint8, int8, etc.)
- REQ-Float-Typ-007: 32-bit float
- REQ-Float-Typ-008: 64-bit float
- REQ-Boolean-Ty-016: Boolean type
- REQ-Bytes-Type-017: Raw bytes type
- REQ-String-Typ-018: String type
- REQ-ASCII-Type-019: ASCII string type
- REQ-Hex-Type-020: Hex string output
- REQ-Skip-Type-021: Skip/padding type
- REQ-Base64-Typ-057: Base64 encoding

Bitfields

- REQ-Bitfield–009: Bitfield extraction
- REQ-Bitfield–010: Bitfield range syntax
- REQ-Bitfield–011: Bitfield from multi-byte
- REQ-Bitfield–012: Bitfield signed extraction
- REQ-Bitfield–013: Bitfield consume behavior
- REQ-Bitfield–014: Bitfield edge cases
- REQ-Bitfield-String-069: Bitfield string output
- REQ-Consume-Fi-015: Consume field behavior
- REQ-Byte-Group-037: Byte group construct

Modifiers

- REQ-Arithmeti-022: Arithmetic modifiers
- REQ-Modifier-O-023: Modifier ordering
- REQ-Mult-Modi-024: Multiply modifier
- REQ-Add-Modif-025: Add modifier
- REQ-Div-Modif-026: Divide modifier
- REQ-Lookup-Ta-027: Lookup table modifier
- REQ-Transform-001: Transform pipeline

Computed Fields

- REQ-Formula-Fi-028: Formula field (deprecated)
- REQ-Polynomial-001: Polynomial evaluation
- REQ-Compute-001: Compute operations
- REQ-Guard-001: Guard conditions
- REQ-Encode-Formula-063: Formula encoding

Conditional Parsing

- REQ-Match-Cond-030: Match conditional
- REQ-Match-Sing-032: Match single value
- REQ-Match-List-033: Match value list

- REQ-Match-Rang-034: Match value range
- REQ-Match-Defa-035: Match default case
- REQ-Match-Defa-036: Match fallthrough

Structures

- REQ-Nested-Obj-029: Nested objects
- REQ-Enum-Type-038: Enum type
- REQ-Enum-Value-039: Enum value mapping
- REQ-Definitio-051: Definitions section
- REQ-Ref-Field-052: $ref field references
- REQ-Variables-070: Variable storage

Repeat/Arrays

- REQ-Repeat-Coun-064: Count-based repeat
- REQ-Repeat-Byte-065: Byte-length repeat
- REQ-Repeat-Unti-066: Until-end repeat
- REQ-Repeat-Mini-067: Min/max constraints
- REQ-Repeat-Vari-068: Variable-based count

Endianness

- REQ-Endiannes-004: Big endian default
- REQ-Endiannes-005: Little endian option

Output Formats

- REQ-Output-For-040: Output formatting
- REQ-IPSO-Outpu-041: IPSO/LwM2M output
- REQ-SenML-Outp-042: SenML output
- REQ-TTN-Output-043: TTN v3 output
- REQ-Unit-Annot-071: Unit annotations

Encoding

- REQ-Encoder-Re-044: Encoder requirements
- REQ-Roundtrip-045: Decode/encode roundtrip

Schema Document

- REQ-Document-S-SHAL-003: Schema format
- REQ-Document-S-REQU-004: Required fields
- REQ-Document-S-REQU-008: Test vectors

Special Features

- REQ-Timestamp-ISO-060: ISO timestamp
- REQ-Timestamp-Elapsed-061: Elapsed time
- REQ-Version-String-062: Version string parsing

Implementation Coverage

Python Interpreter (`tools/schema_interpreter.py`)

- All 31 features implemented

- 401 passing tests
- Supports decode and encode
- Full repeat/array support added

Go Interpreter (`go/schema/schema.go`)

- All features implemented
- Includes repeat, TLV, match, flagged
- Production-ready performance

Notes

- **PASS**: Feature is implemented, documented, and tested
- **PARTIAL**: Feature exists but missing documentation or tests
- **FAIL**: Feature not implemented

All requirements are traced to test cases via `REQ-*` tags in test docstrings.