

PBEnvelope

What is the PBEnvelope?

The `PBEnvelope` is a wrapper message defined in `components/common/envelope.proto` in the ERC-Protobufs repository. Every protobuf message sent over UDP on this rover — whether from the Basestation, the Jetson, or a hardware microcontroller — is wrapped inside a `PBEnvelope` before transmission.

When a raw UDP datagram arrives, the receiver has no way of knowing what type of message is inside without some kind of type identifier. The `PBEnvelope` solves this by using proto3's `oneof` field, which encodes the message type directly into the serialized bytes. The receiver calls `payload_case()` on the deserialized envelope to determine the message type before extracting and deserializing the actual payload.

Structure

The `PBEnvelope` uses a `oneof payload` block where each field corresponds to one registered message type. Only one field can be set at a time — that is the nature of `oneof`. When serialized, proto3 encodes which field is set, allowing the receiver to call `envelope.payload_case()` and get back an enum value like `PBEnvelope::kImuInfo` or `PBEnvelope::kControlMode`.

Registered Message Types

Below is the full table of currently registered payload cases, their field numbers, and their communication direction:

Field Number	Message Type	Direction
1	SensorBoardPHInfo	Hardware → ROS2
2	ArmBoardControlSignals	ROS2 → Hardware
3	ArmBoardDiagnostics	Hardware → ROS2
4	ArmBoardMovementFeedback	Hardware → ROS2
5	ArmBoardActualPositions	Hardware → ROS2
6	ArmBoardTargetMovement	ROS2 → Hardware

7	ArmBoardObstructions	Hardware → ROS2
8	DrivingBoardDiagnostics	Hardware → ROS2
9	DrivingBoardMotorMessage	ROS2 → Hardware
10	DrivingBoardMotorPeriodicProgress	Hardware → ROS2
11	SensorBoardDiagnostics	Hardware → ROS2
12	SensorBoardGPSInfo	Hardware → ROS2
13	SensorBoardIMUInfo	Hardware → ROS2
14	SensorBoardLoadCellInfo	Hardware → ROS2
15	SensorBoardPressureInfo	Hardware → ROS2
16	BasestationControlMode	BS → ROS2
17	BasestationDetectedObject	ROS2 → BS
18	BasestationObjectSelection	BS → ROS2
19	BasestationRockMeasureRequest	BS → ROS2
20	BasestationRockMeasureResult	ROS2 → BS
21	BasestationRoverLocalization	ROS2 → BS

Important: Field numbers in a `oneof` block are permanent. Once a field number is assigned to a message type, it must never be reused for a different type — even if the original message is removed. This would break deserialization for any system still using an older version of the envelope. Always use the next available field number when adding a new entry.

A note on BasestationDetectedObject

Unlike most messages which carry a complete snapshot of state, `BasestationDetectedObject` sends **one detected object per message**. This is because Embedded requires fixed-size protobuf messages and cannot handle repeated fields (which are dynamically sized). The Basestation reconstructs the full frame by collecting all messages sharing the same `frame_id` until it has received `total_count` of them. See `components/basestation/detected_object.proto` for the full definition.

A note on message_types.proto

TLDR: I don't remember what we were doing with this file.

The ERC-Protobufs repo contains a file called `message_types.proto` which defines a `PBMessageType` enum. This file is a leftover from an earlier architecture where a manual type ID system was planned. It is explicitly excluded from the build in `CMakeLists.txt` because its enum values collide with message names in the global proto3 namespace. It is unused, do not reference it or add new entries to it.

Trailing Zero Bytes

The `envelope.proto` file includes an important note: all `PBEnvelopes` sent over the network must have trailing zero bytes removed before transmission. Proto3 serializes unset fields as zero bytes, and stripping them reduces network traffic. The receiver must pad the received datagram back to the full `PBEnvelope` size before deserializing. This is handled by the protobuf library's `SerializeToString` and `ParseFromArray` functions when used correctly.

Adding a New Message Type

To add a new message type to the `PBEnvelope`, see the dedicated [Adding a new message](#) page for the full step-by-step procedure.

Revision #3

Created 2026-04-16 13:30:40 UTC by Andrei Badea

Updated 2026-04-17 13:42:14 UTC by Andrei Badea