### Packet Structure

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Offset | Octet | 0 | | | | | | | | 1 | | | | | | | | 2 | | | | | | | | 3 | | | | | | | |
| Octet | Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| 0 | 0 | Destination MAC Address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 32 | Source MAC Address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 64 | Checksum | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | 96 | Sequence Number | | | | | | | | | | | | | | | | Type | | | | | | | | Length | | | | | | | |
| 16 | 128 |  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |  | Data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |  |  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

### MAC Address

|  |  |  |  |
| --- | --- | --- | --- |
| DC | xx | xx | xx |

Byte 0 = DC (Langlo devices; Other prefixes will be allocated as required)

Bytes 1-3 = Last three bytes of default device MAC address or similar

### Checksum

|  |
| --- |
|  |

32-bit checksum calculated on the remainder of the packet (from Byte 12 to end-of-packet)

### Sequence Number

|  |
| --- |
| n |

n = 1..65535

Sequence number simply wraps after reaching 65535, if it ever comes to that.

### Type

|  |
| --- |
| n |

n = 1..255

Packet payload Type will define the structure of the date provided in the payload. At this point, it will probably define the length, making the Length byte redundant, but who knows what we might need this for in the future.

For the moment, the following packet payload Types are defined:

Power (0x00) — Module battery, solar panel (if relevant) and load voltage (uint16\_t in millivolts) and current (uint16\_t in milliamps) status, each as uint16\_t (12 bytes in total).

Voltage (0x01) — Module supply voltage as uint16\_t (2 bytes)

Tank Level (0x11)— Single byte (uint8\_t) pressure reading. Tank unit could quite easily also report on atmospheric temperature, pressure and humidity if there were any perceived benefit.

Pump Status (0x12) — Single bit on/off status (but currently coded in a whole byte—uint8\_t). Will need to report this periodically, to confirm off-state under normal conditions. Will also need to report more frequently (Every 10 seconds? More often?) when running, once again to verify state.

Weather Report (0x20) — Should ultimately include temperature (uint16\_t, as 10 x temp), pressure (uint16\_t), humidity (uint8\_t), rainfall (uint16\_t, as 10 x rainfall), wind direction (uint16\_t) and wind speed (uint8\_t) (10 bytes in total). Must decide on how often these reports will be issued.

Atmosphere Report (0x21) — A subset of the Weather Report, including atmospheric temperature (uint16\_t), pressure (uint16\_t) and humidity (uint8\_t) only (5 bytes in total).

Rainfall Report (0x22) — A subset of the Weather Report, including rainfall (uint16\_t) only (2 bytes).

Wind Report (0x23) — A subset of the Weather Report, including wind direction (uint16\_t) and wind speed (uint8\_t) only (3 bytes in total).

Sprinkler Controller (0x30) — Reports available per API, via WiFi, so may not even need to be included here, but structure undefined at this point.

All reports could also include an equipment status report, including battery condition, uptime and anything else that seems like a good idea, but the mechanism for including these is yet to be defined.

### Length

|  |
| --- |
| n |

n = 1..48

Packet payload length. In line with the whole LoRa philosophy, this should never get anywhere near maxing out. Packet payload length will, for the moment at least, be defined by the packet payload Type.