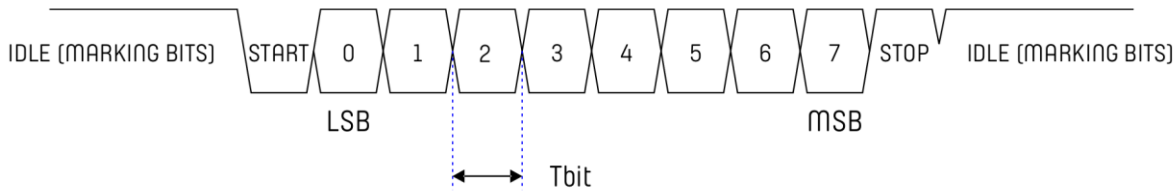
Plensor installation & instruction manual

Embedded sensor

Physical Layer

Communication speed is 921600 baud with 1 start bit (‘0’), 8 data bits and 1 stop bit (‘1’). All data bits are shifted out with the LSB (Least Significant Bit) first. These settings are commonly referred to as 921600, 8, N, 1. One bit time (Tbit) is 1 / 921600 ≈ 1.085 µS. No horizontal parity is used since vertical parity is implemented in the Data Link Layer.



Data Link Layer

The Data Link Layer detects errors that may occur in the physical layer. It defines a mechanism to establish and terminate a connection between two physically connected devices. It also defines the protocol for flow control between them. In normal operation, the sensor devices (TIE-Finder - Slave) are in sleep mode to reduce power consumption. At regular intervals, the Master will wake up specific sensors using a sensor-specific ID. Waking up the Slave is done by sending a Break character followed by a small waiting period and then 6 hexadecimal characters. A Break character consists of a start bit, at least 8 data bits that are set to zero, and an (inverted) stop bit ('0'). This is a violation of a normal byte received by a UART since a stop bit must be a logic high and is therefore considered a Break condition. To avoid character errors or character fragments during a wake-up event, the wake-up data bits must all be zeros ('0'). The number of data bits can be increased as long as their value is zero ('0'). 13 data bits are recommended, similar to a LIN-bus break condition.

For a software bit-bang UART, the falling edge of the RXD signal can be used to wake up a microcontroller. On microcontrollers with a UART peripheral, the break feature is often implemented in hardware. The time it takes to wake up depends on how wake-up from sleep is implemented in the Slave's microcontroller. Crystal oscillators have a certain Oscillator Start-up Time (OST), and some microcontrollers will wait for several Oscillator periods (Tosc) before executing code.

Message Format

The frame format is identical for both the Master and the Slave. The frame consists of 4 sections:

- Message Transmission Start (SOH): \x01 (Pi) / \x01 (Slave)
- Message Number of Frames (MLOF, 16 bits): <number of frames> (Pi) / <number of frames> (Slave)

  Frame Level Loop:

  - Start Byte (STX): \x02 (Pi) / \x02 (Slave)

    - Length of Frame (LOF): … Usually 3840 bits, should be 16 bits long (Pi) / … Usually 3840 bits, should be 16 bits long (Slave)
    - Payload (PYL): … Usually 3840 bits (Pi) / … Usually 3840 bits (Slave)
    - Checksum (CHK): ACK (\x06) or NAK (\x15) Send to Sensor (Pi) / ACK (\x06) or NAK (\x15) Received from Pi (Slave)
- Message Level Signal for End of Receiving Frames (ETX): \x03 (Pi) / \x03 (Slave)
- Message Frame Checksum (MCHK): ACK (\x06) or NAK (\x15) Received from Sensor (Pi) / ACK (\x06) or NAK (\x15) Received from Pi (Slave)
- Message Transmission End (EOT): \x04 (Pi) / \x04 (Slave)

All bytes in a frame are in binary format. The start byte, frame length, and checksum are single bytes. Only the payload can vary in size. The total length of the frame should be less than or equal to 3840 bytes. The LOF is the total number of bytes in the frame, including the start, length, and checksum byte.

Payload description

Within a frame, the payload is the actual data that is to be transmitted. The payload is transmitted in binary format. The Master can send simple commands or extended commands with additional data. Simple commands only contain a command byte (CMD) in the payload and are used for requesting information like sensor data, product ID, production date or a serial number. Extended commands are used to transfer additional data to and from the Slave. Extended commands have additional data bytes after the CMD byte. The additional data can therefore hold a pointer to a memory address, the length of the data and the data itself, in case of a Write EEPROM command. Other examples are calibration parameters, calibration date, product ID or serial number.

The Slave will reply with a simple ACK or NAK (Acknowledge (0x06) or No-Acknowledge (0x15)), followed optionally by the data itself. If a Slave receives a valid frame, but the command is not supported, the Slave must reply with a NAK.

Master example of a simple command: A simple command will have only one command byte in the payload.

STF | LOF | CMD | CHK

Slave reply simple command: In case the command is supported and the data is valid, the Slave will reply with an ACK. Additional data bytes may follow, depending on the command.

STF | LOF | ACK | (DATA) | (DATA) | … | CHK

In case of an error or unsupported command, the Slave will issue a NAK.

STF | LOF | NAK | CHK

Master example of an extended command: An extended command will hold one or more additional data bytes after the command byte

STF | LOF | CMD | DATA | DATA | … | CHK

Slave reply extended command: In case the command is supported and the data is valid, the Slave will reply with an ACK. Additional data bytes may follow, depending on the command.

STF | LOF | ACK | (DATA) | (DATA) | … | CHK

In case of an error or unsupported command, the Slave will issue a NAK.

STF | LOF | NAK | CHK

Glossary of codes

Break B: \x00 (HEX: 0x00)

Message start MS: \x01 (HEX: 0x01)

Frame start FS: \x02 (HEX: 0x02)

End of receiving frames EF: \x03 (HEX: 0x03)

Message end ME: \x04 (HEX: 0x04)

Not acknowledge NAK: \x05 (HEX: 0x05)

Acknowledge ACK: \x06 (HEX: 0x06)

End of line: \x07 (HEX: 0x07)

Sine command: \x08 (0x08)

Block command:\x09 (0x09)

Pulse command:\x0A (0x0A)

Temperature command:\x0B (0x0B)

Glossary of terms used

FLOF: length of frame/payload

MLOF: number of frames

FCHK: Frame XOR based checksum

MCHK: Message # frames checksum

PYL: payload / the frame