LuxSenz communication protocol

LuxSenz uses visible ambient light to wirelessly transmit data from device A (transmitter) to device B (receiver). Both devices should not move. In order to establish a data connection, the receiver must be pointed towards the transmitter. This document explains the communication protocol and explains

# Physical layer

The LuxSenz signal is created by changing the intensity of reflected light at the transmitter. These fluctuations in intensity are picked up by the receiver using a light-sensitive sensor.

The intensity changes should not create health risks or visible flicker. Therefore, the LuxSenz transmitter has to obey the modulation level requirements stated in [IEEE Recommended Practices for Modulating Current in High-Brightness LEDs] to ensure the low-risk region for all (induced) frequency components in the signal. The flicker-index of the signal should be kept low for all frequencies.

LuxSenz uses FSK (Frequency Shift Keying) to transmit streams of bits. A baud rate R is first selected (e.g. R = 80 b/s). Then two integer numbers **k, l** are chosen with **k** < **l**, and usually **k** + 1 = **l** (e.g. **k** = 7,  **l** = 8). The two frequencies **f1, f0** of the FSK signal are then determined as follows: **f1** = **k·**R, **f2** = **l·**R ((e.g. **f1**=560 Hz, **f2**=640 Hz with **k** = 7, **l** = 8, R = 80).

f1 is used to transmit a bit **1** and f2 is used to transmit a bit **0**. A bit **1** consists of **k** (integer) full periods of the light intensity modulated at f1. A bit **0** consists of **l** (integer) periods of the light intensity modulated at f2. Integer numbers **k**, **l** are chosen such that the data rate is constant, i.e. f1/**k** = f2/**l** (the values of f1, f2 may be adjusted to satisfy this condition). Commonly, **k** and **l** are chosen between 4 and 10 with a minimal difference.

f1, f2 are usually chosen around 600 Hz, but all these parameters can be chosen differently, depending on the characteristics of hardware filters on the receiver side and expected interfering light sources (LED lights often blink at 400 Hz). A goal is always to avoid flickering. We must therefore look at the bandwidth as approximated by Carson’s rule. The bandwidth of the FSK signal is then given by the range [**f1**-R, **f2**+R] (e.g. 480-720 Hz with previously mentioned settings). To avoid flickering, the full bandwidth must be above 200 Hz (thus f1 > 200 + R), and not overlap with the frequency of any nearby oscillating light source (fluorescent lights at 50/100 Hz, or LED oscillating at 300/400/500 Hz, or any installed light source).

The waveform shape of the transmitted signal doesn’t need to follow a clear sine wave, but it should always consist of two parts: a part (taking between 20% and 80% of the total time) in which the intensity of reflected light increases and a part (other 20% to 80%) in which the intensity of reflected light decreases. Usually, the control signal has a duty cycle of 50%, and the generated light waveform will be like a sawtooth signal (50% of the time rising edge, 50% of the time falling edge). The average light intensity must be kept constant and at the same level for f1 and f2.

# Data layer

The data link layer of the LuxSenz protocol is based on ASCII code. Data is grouped in bytes consisting of 8 bits, generally with the most significant bit first.

In the idle state, the transmitter continuously sends a SYN (Synchronous Idle, 00010110). A data frame is preceded by STX (Start of Text, 00000010) and followed by ETX (End of Text, 00000011) and ETB (End of Transmission Block, 00010111). Transmission of a data frame can be aborted by sending CAN (Cancel, 00011000).

The default maximum frame length is 128 bytes, but may be altered.

# Network, transport and session layers

LuxSenz only allows two devices per channel with a continuous session.

# Presentation layer

No translation of data encoding is required in current applications.

# Application layer: LuxSenz demo software

This section describes the application layer used for demo purposes.

The following message types exist:

* Text messages, consisting of ASCII text in the data frame
* Control messages, data frame starts with DC2 (Device Control 2, 00010010), then followed by a message

There is no checksum or error correction included in the messages.

A Text message has a maximum data frame length of 64 bytes, consisting of ASCII characters. Text in the data frame should be displayed on the receiver after the complete frame has been received.

The following control messages exist:

* Perform reset: DC2 followed by the ASCII char R or the full word Reset.
* Update displayed sensor value. DC2, then 1 uppercase char indicating the sensor type, then the new sensor value encoded as an ASCII string. The following sensor indication chars are implemented:
  + H Humidity sensor
  + T Temperature sensor
  + P Pressure sensor
  + C CO2 concentration sensor
  + G Gas pollution concentration (Total Volatile Organic Compounds)
* Show/hide a sensor value. DC2 followed by the lower case character indicating the sensor type (see list update displayed sensor value). The sensor indicator should be followed by an x to hide the sensor value, or by an s to show the sensor value.

# Example message: data layer to application layer

As an example, the message details are given for a message that can be sent to update the displayed temperature to 20.3’C.

|  |  |  |
| --- | --- | --- |
| **Binary** | **ASCII Character** | **Description** |
| … | … |  |
| 00010110 | SYN | Idle |
| 00010110 | SYN | Idle |
| 00000010 | STX | Start of message |
| 00010010 | DC2 | This data frame contains a control message |
| 01010100 | T | Temperature sensor update |
| 00110010 | 2 | Part of string to display: 2 |
| 00110000 | 0 | Part of string to display: 0 |
| 00101110 | . | Part of string to display: . |
| 00110011 | 3 | Part of string to display: 3 |
| 01100000 | ‘ | Part of string to display: ` (might be displayed as °C) |
| 01000011 | C | Part of string to display: C |
| 00000011 | ETX | End of data frame |
| 00010111 | ETB | End of message |
| 00010110 | SYN | Idle |
| … | … |  |

# Mitigating clock drift

Frequencies will often deviate a bit from the desired frequencies, because of clock drift or rounding errors. Both the sample frequency at the receiver and the output frequencies at the transmitter can be adjusted. The receiver software is able to correct for small clock drifts, but the quality of the communication channel increases when clock drift is made smaller.

## Correcting the sample frequency at the receiver/transceiver

Before setting up a communication channel, one must ensure that the sample frequency of the receiver is correct. See the section *Correcting the sample frequency* in the document *LuxSenz receiver software*.

## Correcting the output frequencies of the transmitter board

In main.c, the function UpdateFrequencies is implemented. For different duty cycles, the timer periods are always set by dividing a constant number by the frequency value passed to this function. Often, 1 should be subtracted to set the timer frequencies accurately. For a duty cycle of 70% and frequencies of 560 Hz, 640 Hz subtraction is only needed to the timer setting for the timer values counting 70% of the duty cycle before resetting the output signal (ticks\_off\_f\_low and ticks\_off\_f\_high). In this case, the measured frequencies were 559,99 Hz and 640,00 Hz.

Corrections to the output frequencies can be applied by subtracting/adding (small) integer values to ticks\_off\_f\_low, ticks\_off\_f\_high, ticks\_on\_f\_low and ticks\_on\_f\_high, in the function UpdateFrequencies. An output frequency can be measured by connecting an oscilloscope to the output pins of the transmitter board (where the shutters are connected to). In main.h both the frequencies F\_LOW and F\_HIGH should be set to the frequency that should be measured an TICKS\_COMPENSATION should be set to 0 (or the setting for TICKS\_COMPENSATION should be commented, to set the value to 0).

# Maintaining constant light level (signal balancing)

As stated in the section Physical Layer, the average light intensity must be kept constant and at the same level for f1 and f2. To check if this requirement is met, one should measure and compare the average light level for f1 and f2. The LuxSenz receiver can be adjusted to not filter out the average light level, but it is not recommended to use such a LuxSenz receiver, because experiments showed significant variations in the received values. It is easier to use any light sensor connected to, for example, a BeagleBone and take a sample trace containing repeatedly long periods of f1 and f2.

## Preparing the transmitter for signal balancing

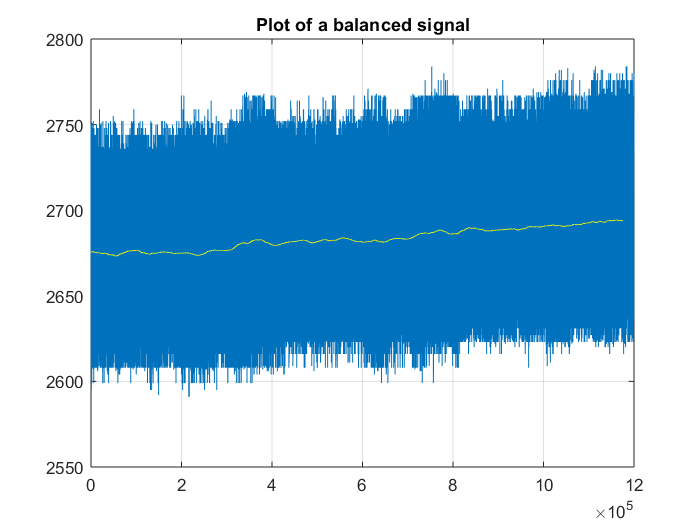
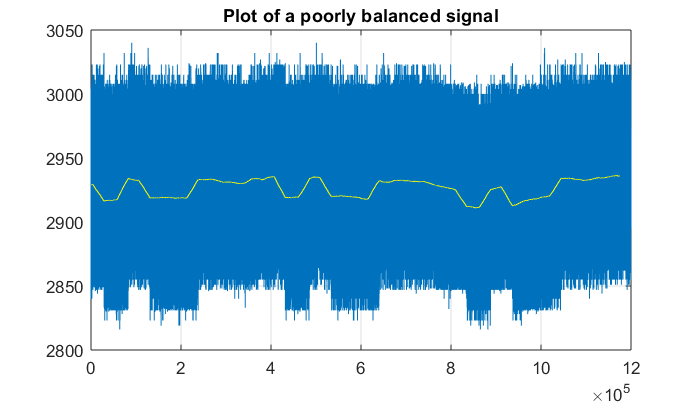
In main.h, select a (new) settings profile on the desired signal frequencies f1 (F\_LOW) and f2 (F\_HIGH), defined in Hz. Set the duty cycle as desired (probably between 50 and 75 %). Set FSK\_REPEAT to a value between f1 and f2, such that every transmitted bit will take approximately one second. Upload the code to the microcontroller of the transmitter.

## Performing a measurement with the BeagleBone

Run the PRUADC program to take a sample trace of the connected light sensor circuit. To save the sample trace to a file, run: ./mem2file 1200000 > filename.dat

Copy the file to a PC (by using SFTP) and import it in your favourite toolbox. Plot the original signal and the signal with a moving average filter (window of 500ms) applied. For MATLAB, a script (balancing.m) is provided to show this plot.

Below two example plots are shown: in the left example the signal is poorly balanced, as there is a significant difference in the average light level (yellow line) for f1 and f2. In the right image, fluctuations in the average level are diminished and the signal is properly balanced. Often, one can still see a different amplitude for f1 and f2. That should not lead to major flickering issues.



## Tune the signal to improve the balancing

By changing the value of TICKS\_COMPENSATION on the transmitter (can be defined in the settings profile in main.h), the signal can be balanced. This value influences the duty cycle of f1 and therefore the average light level when f1 is active. Play around with the value and perform measurements to see what value should be used.