Permanent Frequency Offset Compensation

By Torgeir Sundet

Keywords

- *Frequency Offset Compensation*
- *Component Tolerance/Accuracy*
- *Permanent Compensation*
- *Compensation Span*
- *Centre Frequency*
- *Receive*
- *CC1100*
- *CC1100E*
- *CC1101*
- *CC1110*
- *CC1111*
- *CC2500*
- *CC2510*
- *CC2511*

1 Introduction

Due to component inaccuracy (e.g. crystal drift/tolerance) an RF link might suffer a certain frequency offset between transmitter and receiver, which again could cause degraded sensitivity and link performance/range. The CC11xx/CC25xx counteracts this by using a built-in frequency offset compensation algorithm. Under normal operation, the receiver demodulator compensates for the offset between the transmitter and receiver frequency, within certain limits, by estimating the centre frequency of the received data. This value is available in the [FREQEST] status register. The tracking range of the algorithm is selectable as fractions of the receive channel bandwidth, using the [FOCCFG.FOC_LIMIT] configuration register. However, this compensation is not permanent, so the next time the CC11xx/CC25xx enters RX the synthesizer will not be aligned with the previously received centre frequency.

If the initial frequency offset is large, and close to the edge of the built-in compensation span, the ability of the CC11xx/CC25xx to fully compensate for any further frequency drift is limited. That is, if the RX and TX frequencies drift too far apart, the receiver eventually fails to fully compensate, because the offset between the programmed RX/TX frequency and the actual RX/TX frequency exceeds the span of the built-in algorithm.

In order to regain the ideal compensation span centre it is possible to make the frequency offset compensation permanent. This is done by reading the [FREQEST] status register and writing the accumulated value back to the [FSCTRL0.FREQOFF] register. As a result the next time CC11xx/CC25xx enters RX or TX the frequency synthesizer is adjusted to align with the previously received centre frequency. Hence this method basically introduces a fixed offset into the built-in algorithm, which in turn relocates the compensation centre, increases the available compensation range, and finally improves the sensitivity the next time RX is entered. Please refer to Figure 1 for illustration of sensitivity loss as a result of frequency offset.

Applying permanent frequency offset compensation makes the receiver regain its optimum sensitivity and thus improves the corresponding link range. However, note that permanent frequency offset compensation relies very much upon the accuracy of the measured frequency offset. A noisy/inconsistent measurement (made by the chip itself) is more likely to degrade the performance than improve it. So it might be wise to filter (software) the measured frequency offset in order to suppress noisy samples. Generally the application should also implement a safety/recovery feature, which enables the receiver to locate the actual transmitter frequency. Otherwise the application will suffer permanent RF link degradation.

Table of Contents

2 Abbreviations

EB	Evaluation Board, e.g. SmartRF®04EB.
EM	Evaluation Module, e.g. CC1100EM.
MCU	Micro Controller Unit
RX	Receive
TX	Transmit

3 Sensitivity versus Frequency Offset

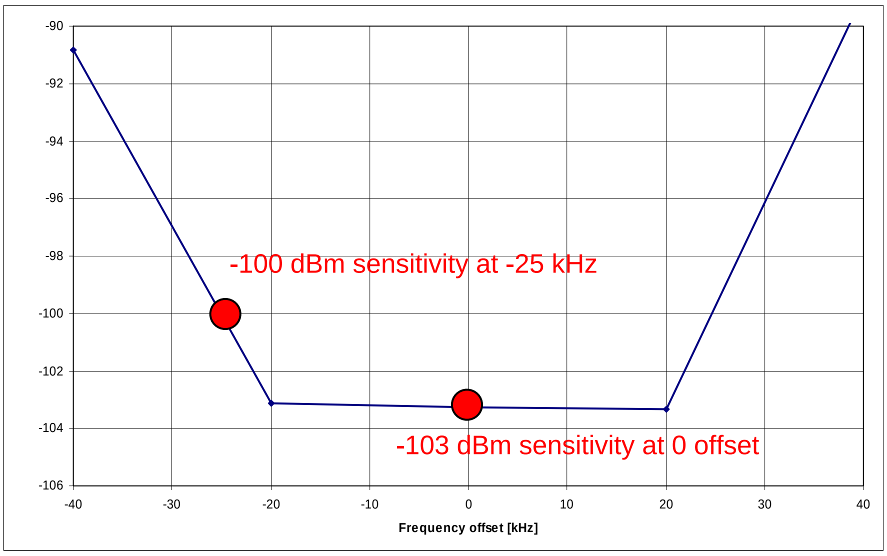


Figure 1. Sensitivity versus Frequency Offset

Figure 1 shows that an RF link suffers loss of sensitivity when the receiver and transmitter frequency drifts apart. Unless properly counteracted using permanent frequency offset compensation, further frequency drift towards the edge of the compensation span can lead to permanent range reduction and thus degraded link performance.

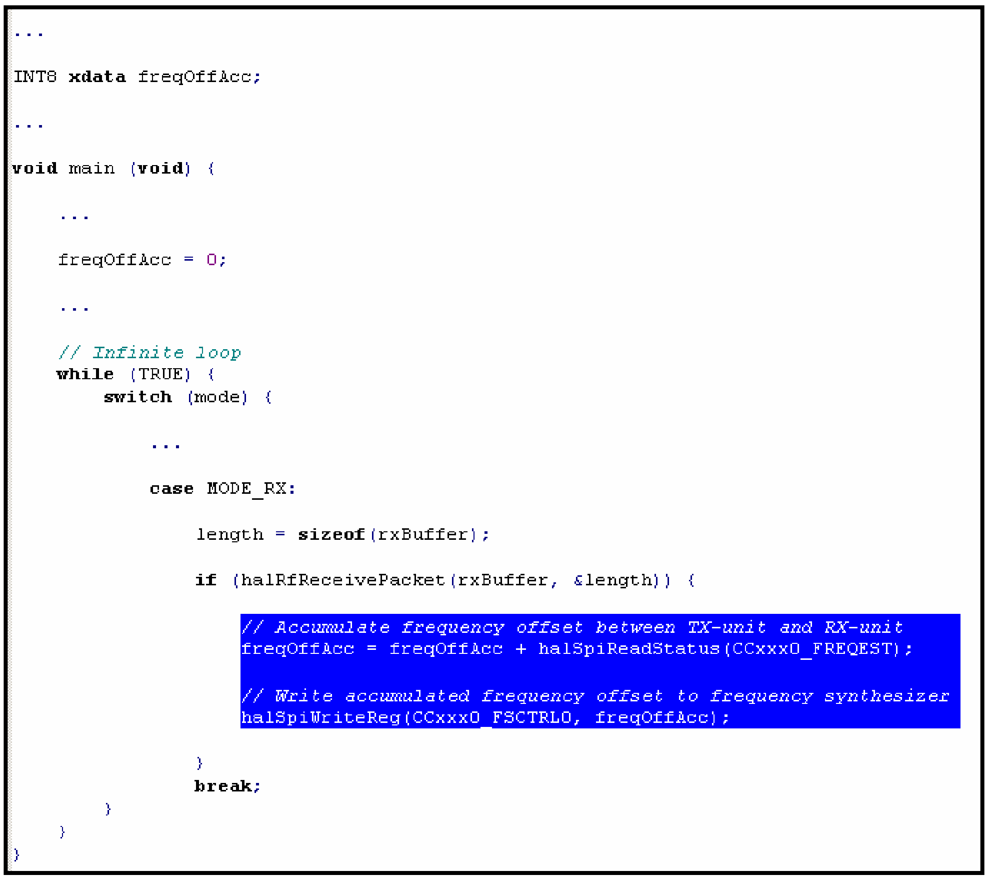
4 Implementation of Permanent Frequency Offset Compensation

Permanent frequency offset compensation can easily be tested using e.g. a standard CC1100 Development Kit, which includes two SmartRF® 04EBs and a number of CC1100EMs.

In this design note the transmitter board was programmed with an unmodified version of the “ex_link” example (available in the CC1100 CC1101 CC2500 Examples Libraries). The receiver board used a slightly adapted version of it, and the essential part of this software is shown in Figure 2. The required receiver software for permanent frequency offset compensation is simple and can be illustrated with the following sequence:

1. Allocate e.g. an MCU memory location to store the accumulated frequency offset value and initialize it to zero (assuming zero frequency offset at the very beginning).
2. Await a valid incoming RF packet.
3. In CC11xx/CC25xx **IDLE** state, (typically right after an RF packet has been received) read the current frequency offset from [FREQEST.FREQOFF_EST] and add this value to the previously sampled value in MCU memory.
4. Write the accumulated frequency offset from MCU memory into [FSCTRL0.FREQOFF].

Note that the receiver software must apply the accumulated value to [FSCTRL0.FREQOFF], not just the instant [FREQEST.FREQOFF_EST] value. It is also important that the CC11xx/CC25xx is in **IDLE** state when reading [FREQEST.FREQOFF_EST], to ensure that the [FREQEST.FREQOFF_EST] value links to the last received RF packet. Furthermore, noisy [FSCTRL0.FREQOFF] samples can be suppressed by filtering it (e.g. low-pass filter), but this has not been done in this design note. Finally, note that in an RF network application each receiver should memorize the frequency offset value associated with each transmitting node in order to apply the correct permanent offset when communicating with different nodes.



```
...

INT8 xdata freqOffAcc;

...

void main (void) {

    ...

    freqOffAcc = 0;

    ...

    // Infinite loop
    while (TRUE) {
        switch (mode) {

            ...

            case MODE_RX:

                length = sizeof(rxBuffer);

                if (halRfReceivePacket(rxBuffer, &length)) {

                    // Accumulate frequency offset between TX-unit and RX-unit
                    freqOffAcc = freqOffAcc + halSpiReadStatus(CCxxx0_FREQEST);

                    // Write accumulated frequency offset to frequency synthesizer
                    halSpiWriteReg(CCxxx0_FSCTRL0, freqOffAcc);

                }
                break;
        }
    }
}
```

Figure 2: Software for Permanent Frequency Offset Compensation

5 General Information

5.1 Document History

Revision	Date	Description/Changes
SWR159A	2009.03.12	Added CC1100E.
SWR159	2007.10.22	Initial release.