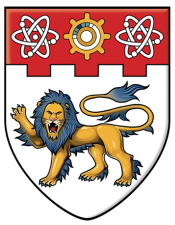

NANYANG
TECHNOLOGICAL
UNIVERSITY

# POWER OPTIMISATION FOR ADAPTIVE EMBEDDED WIDEBAND RADIOS

A Thesis submitted to
the School of Computer Engineering
of Nanyang Technological University

by

**Pham Hung Thinh**

in fulfillment of the requirement for
the Degree of Doctor of Philosophy of
for Computer Engineering

March 18, 2015

# List of Figures

# List of Tables

# Chapter 1

# Research Introduction

# Chapter 2

# Background Literature

# Chapter 3

# Multiplierless Correlator Design for low-power systems

# Chapter 4

# A Method for OFDM Timing Synchronisation

## 4.1 Introduction

OFDM performance is sensitive to receiver synchronisation [3]. Frequency offset causes inter-subcarrier interference, and errors in timing synchronisation can lead to inter-symbol interference. Therefore, accurate synchronisation is critical to the performance in OFDM systems. This section summarizes conventional synchronisation methods used for OFDM systems, discussing their merits and drawbacks, and then the following sections in this chapter presents a new and efficient method that is robust to large frequency offset, obtains accurate and robust synchronisation and requires relatively low complexity computation. It is suitable for hardware implementation on reconfigurable systems.

The method presented in this chapter has also been discussed in:

- T. H. Pham, I. V. McLoughlin, and S. A. Fahmy, "Robust and Efficient OFDM Synchronisation for FPGA-Based Radios," in *Circuits, Systems, and Signal Processing, vol. 33, no. 8, pp. 2475 - 2493, Aug. 2014, Springer* [4].

## 4.2 Related Work

The autocorrelation-based method is commonly preferred for implementing synchronisation because of its low computation requirements. In [1], timing metrics are defined as follows: the normalised power measure $M[d]$:

$$M[d] = \frac{|P[d]|^2}{(R[d])^2}, \tag{4.1}$$

where $d$ denotes a time index corresponding to the first sample in a search space comprising $2L$ samples of received signal $r$. The power measure $R$, which represents the energy of the second half of the receiver search window, is defined as:

$$R[d] = \sum_{m=0}^{L-1} |r[d+m+L]|^2, \tag{4.2}$$

and the normalisation parameter $P$ computes the correlation between two periodic halves in the search window as:

$$P[d] = \sum_{m=0}^{L-1} (r^*[d+m]r[d+m+L]), \tag{4.3}$$

The metric $M[d]$ for a typical channel is illustrated in Fig. 4.1, where it can be seen to form a distinct plateau when the preamble is presented within a received window. Clearly, $M[d]$ accurately detects the preamble, however the nature of the plateau having a flat top presents an uncertainty of positioning, and hence degrades the accuracy of determining the exact start of the frame in practice.

In order to improve the accuracy of synchronisation, a scheme is often used as in [5–9] based on combining the above metric for detecting frame, estimated coarse STO and fractional CFO, and additional cross-correlation operations for computing fine STO and integer CFO.

### 4.2.1 Coarse STO and Fractional CFO Estimation

First, the plateau of the $M[d]$ metric is used to detect a frame start and estimate coarse STO by comparing the magnitude of $M[d]$ to a given threshold. After this, the CFO is estimated using the $P$ metric in (4.3). We now assume that we are receiving a signal $s[d]$ which has a normalised carrier frequency offset $\xi$ with respect to $r[d]$:

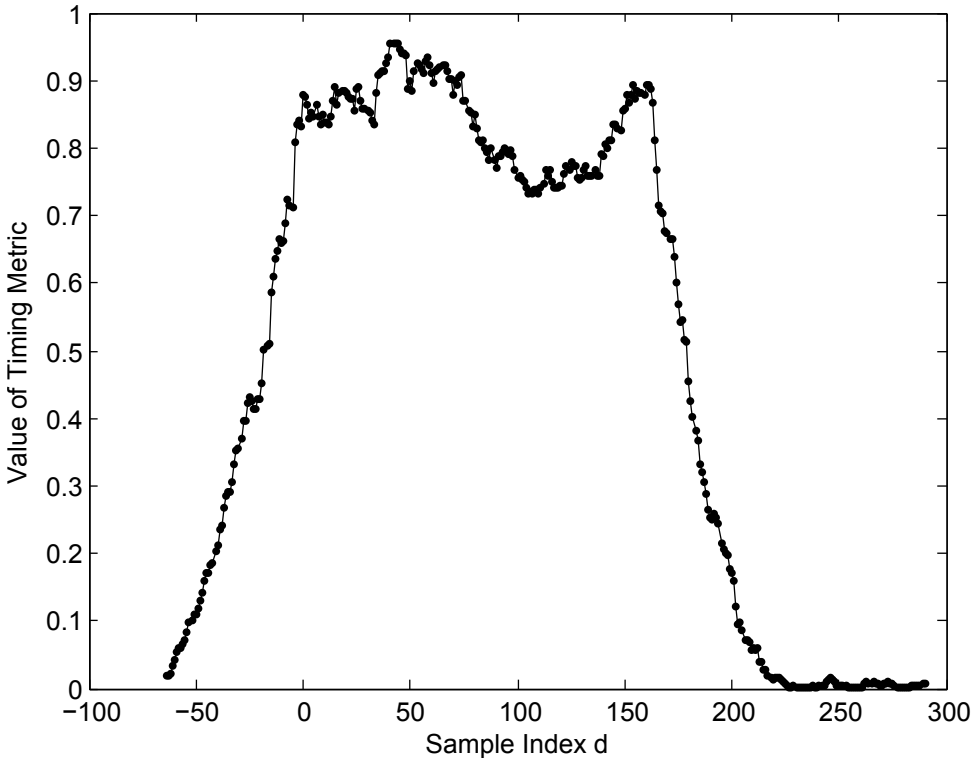


Figure 4.1: The timing metric in [1] applied to IEEE 802.16-2009 preamble in the AWGN channel (SNR = 10dB)

$$s[d] = e^{j2\pi\xi\frac{d}{N}} * r[d]. \tag{4.4}$$

where $N$ is the numbers of subcarriers of OFDM symbol. At the plateau of $M[d]$, the received samples in the window will be periodic:

$$r[d] = r[d + L]. \tag{4.5}$$

Substituting (4.4) and (4.5) in (4.3)

$$\begin{aligned} P[d] &= \sum_{m=0}^{L-1}(s^*[d+m]s[d+m+L]) \\ &= e^{j2\pi\xi\frac{L}{N}}\sum_{m=0}^{L-1}|r[d+m]|^2 \end{aligned} \tag{4.6}$$

The CFO can be determined as follows;

$$\xi = \frac{\angle P[d] + 2\pi z}{2\pi\frac{L}{N}} \tag{4.7}$$

where $\angle P[d]$ is the angle of $P[d]$ within the range -$\pi$ to $\pi$. The CFO consists of 2 parts: the fractional part, $\hat{\lambda} = \frac{\angle P[d]}{2\pi\frac{L}{N}}$, can be estimated replied on $\angle P[d]$ and the integer part (IFO) is represented by an integer $z$, $\hat{\epsilon} = \frac{zN}{L}$. The fractional CFO estimation has a limitation as follows:

$$-\frac{N}{2L} < \hat{\lambda} < \frac{N}{2L} \tag{4.8}$$

Clearly, the range of estimating fractional CFO can be increased by decreasing the length of period $L$. However, because of channel noise, a smaller value of $L$ may degrade the accuracy of estimating the fractional CFO $\hat{\lambda}$. In the case of the IEEE 802.16-2009 preamble [10], the number of subcarriers $N$ and length of period $L$ are commonly chosen to be 512 and 64 samples, respectively [11]. So the fractional CFO estimation range is within -2 to +2 subcarrier spacings. And the value of IFO parts $\hat{\epsilon}$ is estimated as an integer that is multiple of 4.

This method is fast and robust for estimating coarse STO and fractional CFO. But it has some drawbacks. First, the coarse STO is estimated by comparing the metric to a threshold. It

is not certain that the samples for estimating the fractional CFO are within the plateau of $M[d]$ in which the received samples in the window will be periodic. This degrades the performance of fractional CFO estimation. Second, the CFO estimation only performs correctly in the limited range of the fractional CFO shown in (4.8). If the CFO is outside this range, it is necessary to estimate the integer CFO separately.

### 4.2.2 Fractional CFO Compensation

Before fine STO estimation is performed using cross-correlation, that is sensitive to CFO, the estimated fractional CFO must be used for frequency offset compensation by phase de-rotating the received samples in the time domain:

$$\widehat{r[d]} = e^{-j2\pi\xi\frac{d'}{N}} * s[d]; \quad d' = d + t_{off}. \tag{4.9}$$

$\widehat{r[d]}$ and $s[d]$ are the compensated samples and frequency offset samples, respectively. $d'$ is the coarse estimated timing index that differs from the correct timing index $t_{off}$ samples. Substituting (4.4) in (4.9), we get:

$$\begin{aligned} \widehat{r[d]} &= e^{-j2\pi\xi\frac{(d+t_{off})}{N}} * e^{j2\pi\xi\frac{d}{N}} * r[d] \\ &= e^{-j2\pi\xi\frac{t_{off}}{N}} * r[d]. \end{aligned} \tag{4.10}$$

When compensating the fractional CFO, the fine STO has still not been estimated. This results in a common phase error within the coarse CFO compensated samples as shown in (4.10).

### 4.2.3 Fine STO Estimation

The purpose of timing synchronisation is to find the correct starting point for receiver demodulation. This starting point will generally mark the beginning of a discrete Fourier transform (DFT) window. Coarse time synchronisation is commonly based on auto-correlation as mentioned above, leading to relatively simple hardware implementation. However, with purely coarse STO, it is almost impossible to achieve sufficient accuracy to correctly detect the starting

point. Thus, fine STO estimation is necessary to refine the accuracy of timing synchronisation. To obtain more accurate fine STO estimation, cross-correlation is usually performed between the received samples and known transmitted preamble. The peak cross-correlation occurs when the received samples match to the known preamble. Fine STO can thus be found by locating this peak within a search window. In order to reduce the overhead of cross-correlation, a sign bit multiplier can be used instead of a real multiplier to calculate cross-correlation. However, this is much more sensitive to CFO in practice [5]. Some researchers employ enhanced methods based on cross-correlation for the time synchronisation. For example, Kishore and Reddy [2] presented an algorithm using cross-correlation between the known transmitted and received preamble symbols. The timing metrics for synchronisation using this method perform the normalised $M(d)$ calculation, and use the same denominator $R$ as in (4.1) and (4.2), however they refine the numerator $P$ as follows;

$$P[d] = \sum_{m=0}^{L-1} (r[d+m]a[m])^*(r[d+m+L]a[m]), \tag{4.11}$$

where $d$ again denotes the time index corresponding to the first sample in a window of $2L$ samples of received signal $r$ and superscript "$*$"again denotes complex conjugation. In (4.11) the samples $a$ are now the known, transmitted, time domain preamble samples.

The cross correlation now produces distinct peaks at the times when received samples match to the known transmitted samples of the preamble. Fig. 4.2 illutrates this by plotting metrics $M$ versus the sample index $d$ for an example channel in AWGN with SNR = 10 dB. A frame is detected when $M$ crosses a threshold and the start of frame is found by searching the peaks of $M$. This can accurately determine the start of frame even at low SNRs. Moreover, this method is robust to large CFO for time synchronisation. Simulations [2] show that time synchronisation can be performed with CFO = 10.5 sub-carrier spacings. However, the cross-correlation operation requires complex computation. Although the complexity of cross-correlation can be reduced using multiplierless correlation [12], the multiplierless correlator degrades the precision of metric $P$, used for estimating frequency offset. In general, this method is appropriate for time synchronisation but requires significant hardware resources for cross-correlation when implemented.

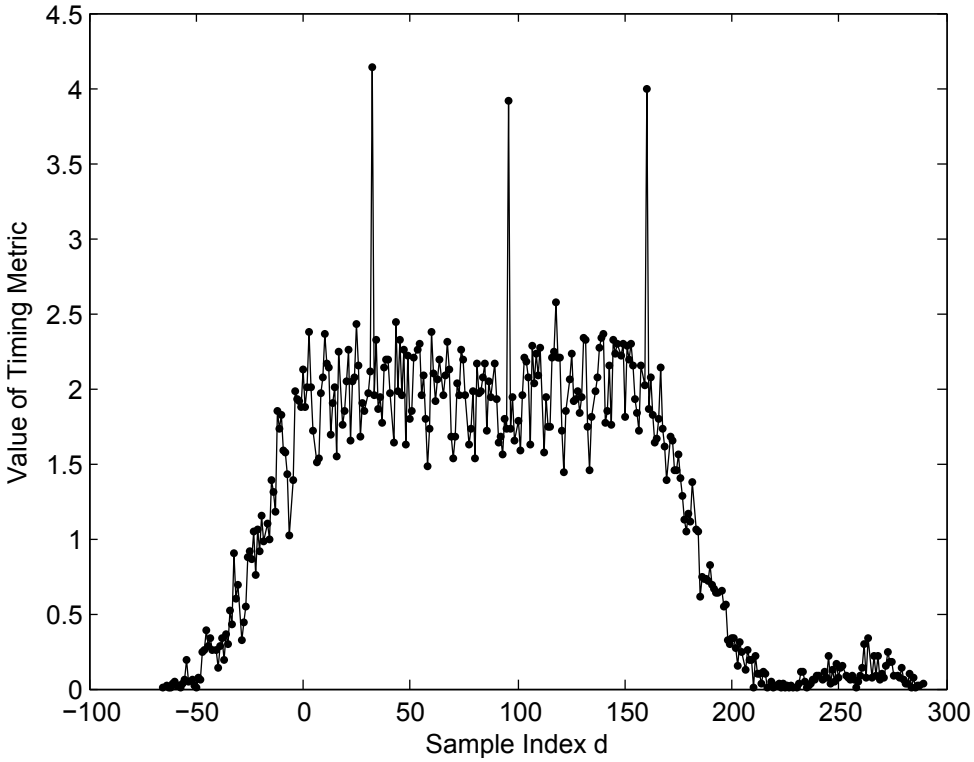


Figure 4.2: The timing metric in [2] apply to IEEE 802.16 preamble in the AWGN channel (SNR = 10dB)

## 4.3 Proposed Fractional CFO Estimation and Synchronisation

Considering the limitations of previous work, we propose new timing metrics to take advantages of preamble characteristics such as period and energy distribution. Again, the proposed method is illustrated for the specific case of the IEEE 802.16-2009 preamble, as shown in Fig.4.4.

Firstly, we define the autocorrelation between the two halves of the receiver window for normalisation purposes;

$$P'[d] = \sum_{m=0}^{M-1} (r^*[d+m]r[d+m+L]) \tag{4.12}$$

where $d$ denotes a time index of received signal $r$ that now corresponds to the first sample in a window of $L + M$ samples of received signal, $L$ being the length of preamble period (64 for the IEEE802.16 preamble) and $M$ being the length of received samples for estimation, respectively.

Next, a power measure, $R'$ is proposed which takes account of both recevied and transmitted symbol power;

$$R'[d] = \sum_{m=0}^{M-1} |r[d+m+L]|^2 |a[m]|^2 \tag{4.13}$$

In common with the metric developed by Schmidl and Cox [1], our $P'(d)$ detects the periodic characteristic of the preamble and can thus be used for estimating fractional CFO. $P'$ forms a plateau is evident in Fig.4.3 when the preamble is present within the receiver window. $R'$ is then used to find the starting point of the preamble based on its energy distribution. $R'$ causes peaks shown in Fig.4.3 at the point where the energy distribution of the received samples matches that of the transmitted preamble. Since the $R'$ metric that is used to estimate time synchronisation is computed on the square of the amplitude of received samples, the time synchronisation is insensitive to the CFO that effects the phase of the received samples. Moreover, computing on amplitude squared, a real number, requires less resources than using complex numbers as in alternative approaches such as [2]. In addition, the time synchronisation is performed based on the peak value of $R'$ rather than its absolute value. Therefore $R'$ is a good

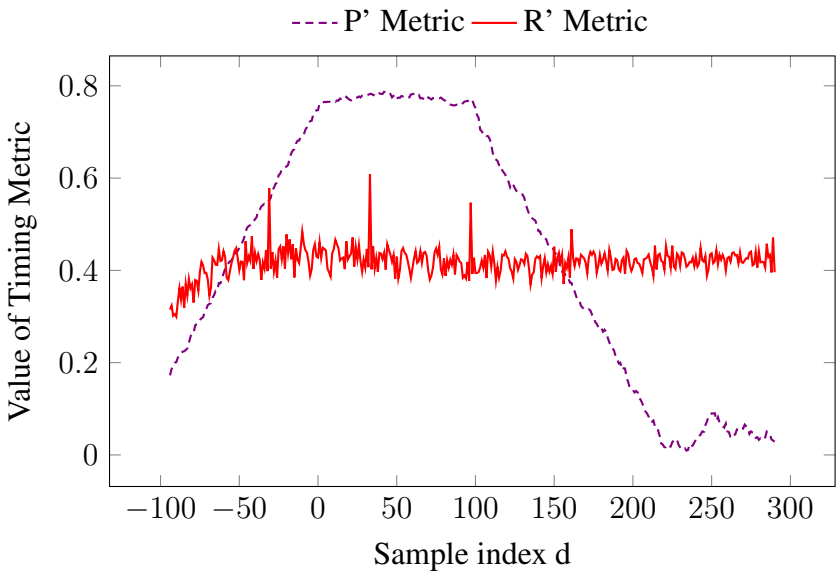


Figure 4.3: Proposed timing metrics applied to the IEEE 802.16 preamble in AWGN (SNR = 10 dB, CFO = 10.5).

candidate for implementation using a multiplierless correlator to reduce computational complexity. Fig.4.4 illustrates the flow of the proposed synchronisation according to the received samples of the preamble, compared to that of the conventional scheme. As can be seen in the conventional scheme, the coarse STO and fractional CFO estimation are performed in the two first periods of the short training symbol from (i) to (iii), and then the estimated fractional CFO is used for compensation. The fine STO is estimated in the last period of the short training symbol from (iv) to (vi), and integer CFO should be computed in the long training symbol from (vii) to (viii). By contrast, the proposed method performs STO synchronisation and fractional CFO estimation in the three first periods of the short training symbol from (i) to (iv), and the start of frame will be detected at (ii). Then the fractional CFO compensation is computed, and integer CFO estimation is performed in the long training symbol from (vii) to (viii).

The method used for synchronisation based upon these metrics is as follows:

### 4.3.1 Frame Synchronisation and Fractional CFO Estimation

First, frame detection is performed by comparing the metric $P'$ to $R'$ with a threshold $thr$ as shown in (4.14)

$$|P'[d]| > thr * R'[d]. \tag{4.14}$$

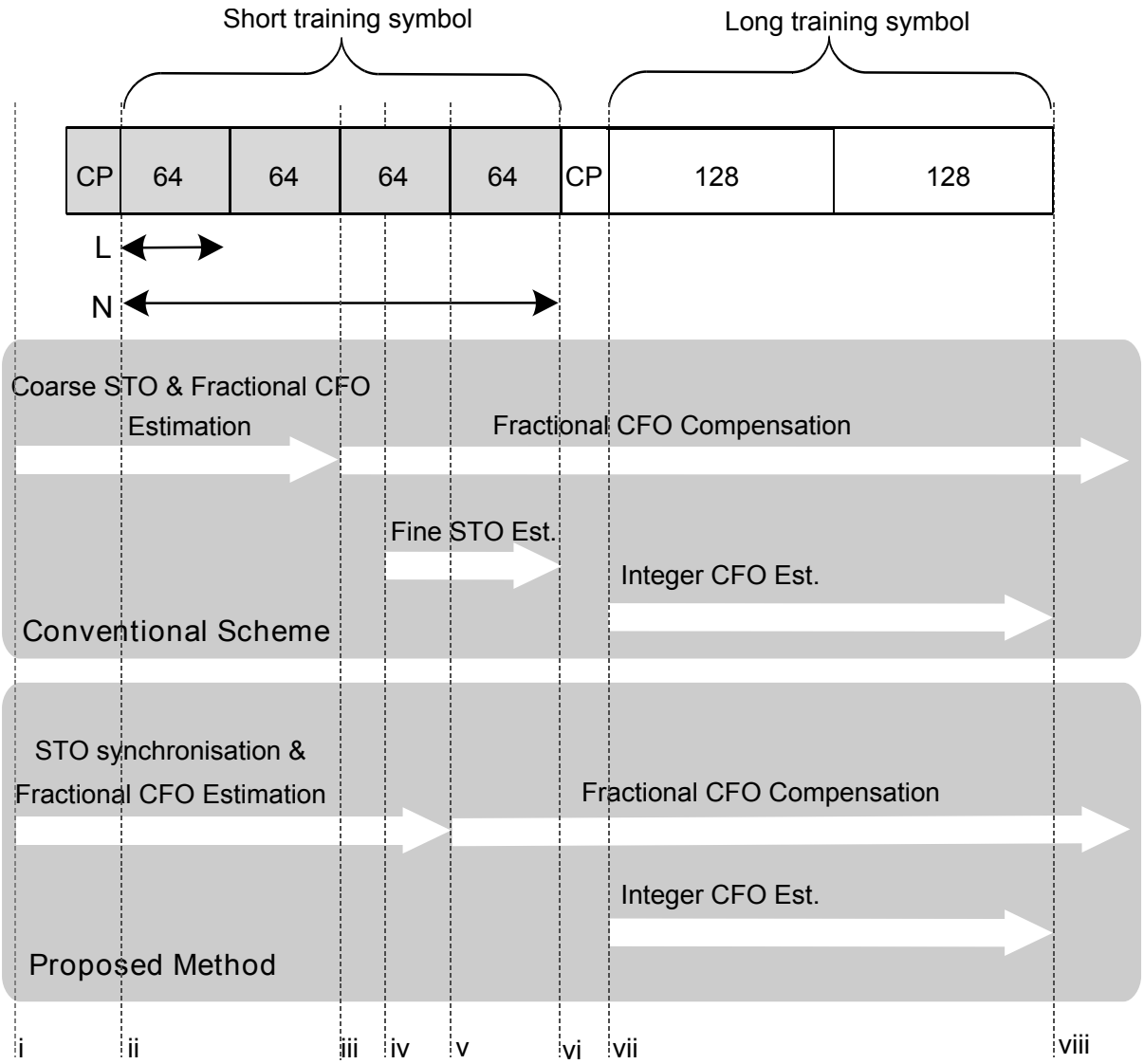


Figure 4.4: The synchronisation flow according to the received samples within the preamble showing its packet format above the conventional synchronisation scheme flow, and proposed scheme below.

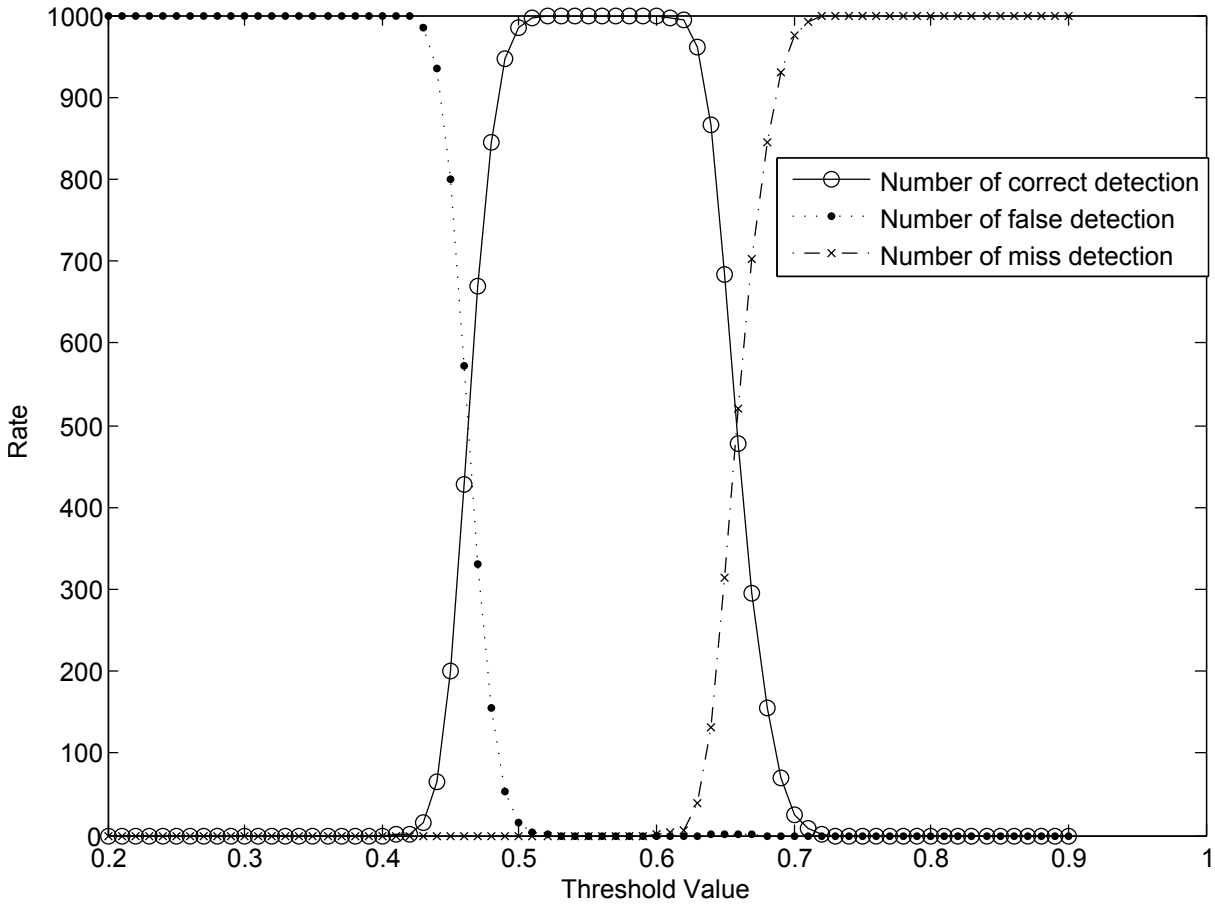


Figure 4.5: Performance of the frame synchronisation versus selecting threshold in the AWGN channel (SNR =10dB)

The threshold is different for each channel and needs to be determined empirically by simulation (in common with other systems such as [2]). Fig.4.5 shows the performance of the frame synchronisation in the given channel for different threshold values. With each threshold value, 1000 frame detections are simulated. The number of correct detections indicates how often the time synchronisation has been found correctly at the start of the frame. The number of false detections indicate that the time synchronisation gets the wrong start of the frame. Otherwise, the frame is not detected (i.e. $|P'[d]|$ can be not greater than $thr * R'[d]$), and a miss detection is declared. From the results plotted, the optimum value can be seen to be around 0.55. If the threshold is too small, noise may cause a failure of detection. On the other hand, when the threshold is large, the frame detection may be missed if the noise reduces the amplitude of the timing metric, which consequentially does not cross the threshold. Assuming the channel

is known, the threshold can be selected from a look-up table. In the simulations reported in this chapter, each channel threshold has been pre-computated, based upon the current channel model and conditions.

After a frame is detected, the starting point of the short preamble is found by searching for the peaks in $R'[d]$. As can be seen in Fig. 4.3, two peaks in $R'[d]$ will bracket the transition to the plateau in $P'[d]$ shown at sample indices of -31 and 33, respectively. The second peak is used as the starting point for the DFT window. This is accomplished for the maximum values of $R'[d] + R'[d-L]$ over the next $L$ samples after frame detection. Second, the fractional CFO estimation is presented in (4.15) based on the $P'[d]$ metric similarly to other work such as [1, 5–9];

$$\hat{\lambda} = \frac{\angle P'[d]}{2\pi \frac{L}{N}} \tag{4.15}$$

where $\hat{\lambda}$ is the estimated fractional CFO, $\angle P'[d]$ denotes the angle of P'[d] and $N$ is the number of subcarriers. In this proposed method, the fractional CFO is estimated at the starting point of the preamble to guarantee that the correct angle of $P'[d]$ is taken for estimation. Using the conventional techniques, the inherent uncertainty in the coarse STO does not allow this to be achieved in practice. Thus, the estimated fractional CFO in the proposed method will, on average, tend to be more accurate. In addition, this method separates the length of preamble period $L$ and the length of received samples for estimation $M$, and because of that $L$ can be small to extend the range of fractional CFO, and $M$ can be longer to increase the robustness to channel noise. For the IEEE802.16 preamble, $M$ is set equal to $2L$ to improve the overall precision of synchronisation.

### 4.3.2 Fractional CFO Compensation

Compensating the Fractional CFO is performed in the traditional way by phase de-rotating the received samples in the time domain;

$$\begin{aligned} \widehat{r[d]} &= e^{-j2\pi\Delta f_c d T_s} * s[d]. \\ &= e^{-j2\pi\xi\frac{d}{N}} * e^{j2\pi\xi\frac{d}{N}} * r[d] \\ &= r[d]. \end{aligned} \tag{4.16}$$

However, in this proposed method, the frame synchronisation is achieved before performing fractional CFO compensation, therefore, $t_{off}$ is removed, and thus, the common phase errors caused by fractional CFO compensation are missing. The compensated received samples are then demodulated using the DFT to obtain the received symbols in the frequency domain.

### 4.3.3 Simulation Results and Discussion

We evaluate the performance of the proposed synchronization method, applied to the IEEE 802.16-2009 downlink preamble, in MATLAB under both AWGN channels as well as the more realistic SUI channels that are widely used in the research literature [2, 11].

The Stanford University Interim (SUI) channel model [13] is used to simulate a frequency selective channel and takes into account many wireless channel effects including delay spread, Doppler spread, phase noise, and channel interference. In addition, the value of metrics computed in MATLAB are later verified with corresponding values from the FPGA simulation, to ensure practical functional equivalence.

In total, 100,000 OFDM frames, preceded by noise with randomly seeded AWGN and followed by preamble and data symbols, are used to evaluate synchronization performance for each method. The proposed method is compared to the state of the art method, in terms of accuracy of both time synchronization and fractional CFO estimation. The performance of STO estimation is measured in terms of failure rate (%), and the accuracy of CFO estimation is evaluated in terms of mean square error (MSE). We separately evaluate the robustness of time synchronization against large CFO for each method. Three versions of the proposed approach are constructed by varying the length, $C$, of received samples for estimation based on (4.12). These are investigated to determine the tradeoff between accuracy and computational cost, with $C$ defined as follows in each case:

- *Prop 1*: $C = L$
- *Prop 2*: $C = 2L$
- *Prop 3*: $C = 3L$

These are compared to the state of the art method (denoted as *SoA*) [9, 14], and the method of Kishore and Reddy [2] (denoted as *K&R*) for a number of evaluation scenarios. First, the

performance of each method is found for AWGN channels beginning with CFO = 0.5, then AWGN channels with CFO varying from -10 to +10 times carrier spacing, then SUI1, and finally SUI2 channels.

**Performance in AWGN**

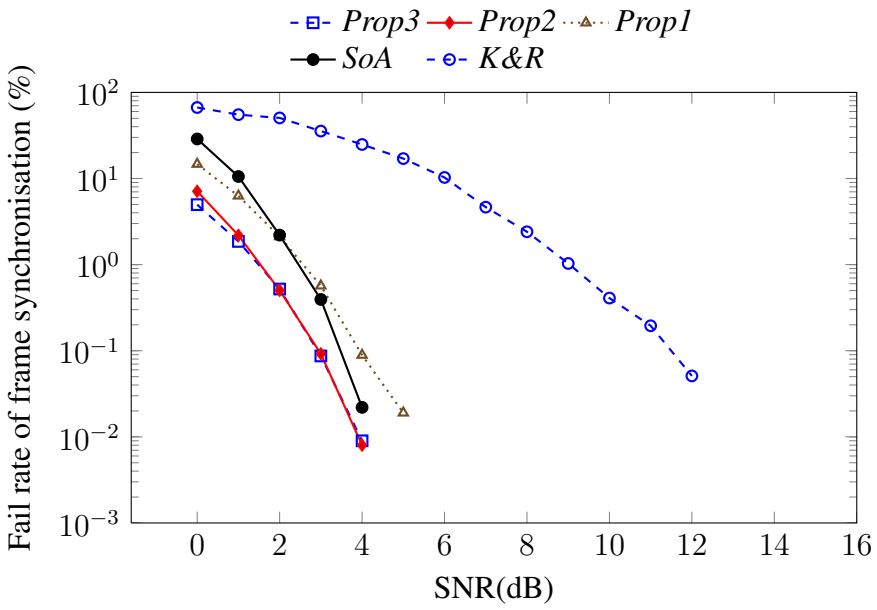


Figure 4.6: Performance of time synchronization in AWGN channels with a frequency offset of 0.5 time subcarrier spacing.

Fig. 4.6 and Fig. 4.7 plot the performance results of STO and CFO estimation in AWGN with a frequency offset of 0.5 subcarrier spacings, respectively. *SoA* and the proposed methods have much better performance than *K&R* in these tests, achieving perfect synchronization when SNR exceeds 5 dB. *Prop1*'s STO estimation has slightly better accuracy with SNR below 3 dB but is worse at higher SNR than *SoA*. Increasing the length of received samples for estimation, i.e., setting $C = 2L$, allows *Prop2* to obtain a remarkable improvement in STO estimation, clearly better than the estimation achievable by *SoA*. *Prop3*, with $C = 3L$, demonstrates decreasing gains: it is not able to enhance accuracy as much as *Prop2*, despite a considerable hardware cost incurred when increasing $C$. In addition, the CFO of *Prop3* and *Prop2* achieve significant improvement compared to the other methods in Fig. 4.7, while the

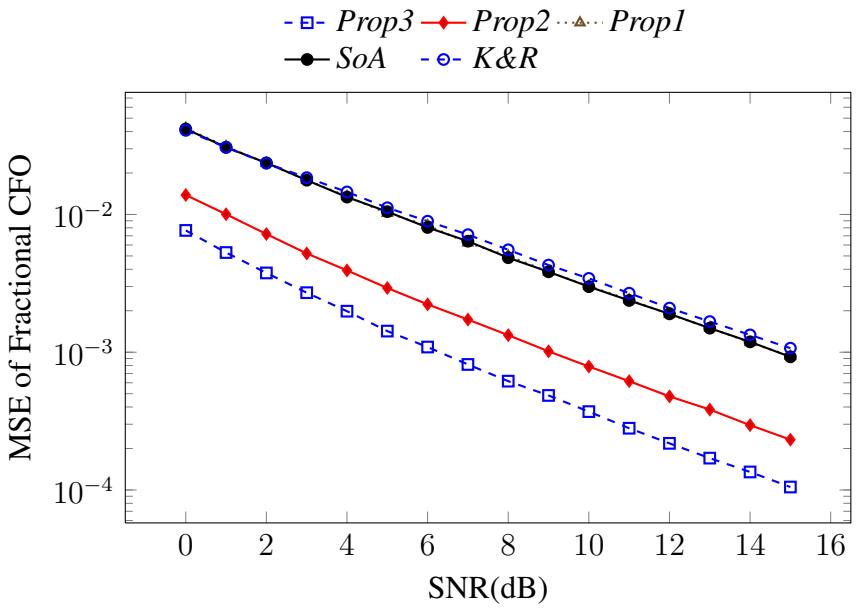


Figure 4.7: Performance of fractional frequency offset estimation in AWGN channels.

accuracy of *Prop1* and *SoA* are identical (the curve for *Prop1* is hidden behind the curve for *SoA* as a result).

The gap between *Prop1* and *Prop2* is much larger than that between *Prop2* and *Prop3*, again demonstrating decreasing gains as $C$ is extended. Thus, increasing $C$ from $L$ to $2L$ is a more effective improvement than extending $C$ from $2L$ to $3L$. The accuracy of CFO estimation in *Prop2* is improved by about 5 dB in comparison to *SoA* and *K&R*. This improvement is as a result of the increased length of received samples for estimation, $C$. These results show the proposed method to be competitive with state of the art methods.

### Performance in Fading Channels

Fig. 4.8 and Fig. 4.9 present the performance results of STO estimation in SUI1 and SUI2 channels, respectively. The proposed methods are seen to achieve much better accuracy than the *K&R* method. Compared to *SoA*, Fig. 4.8 reveals that the estimation of *Prop1*, *Prop2* and *Prop3* is more accurate when SNR is below 3 dB. However for higher SNRs, the accuracy of *SoA* is slightly better than that of the proposed method. Increasing the length of received samples for estimation achieves an improvement when SNR is below about 5 dB, although the

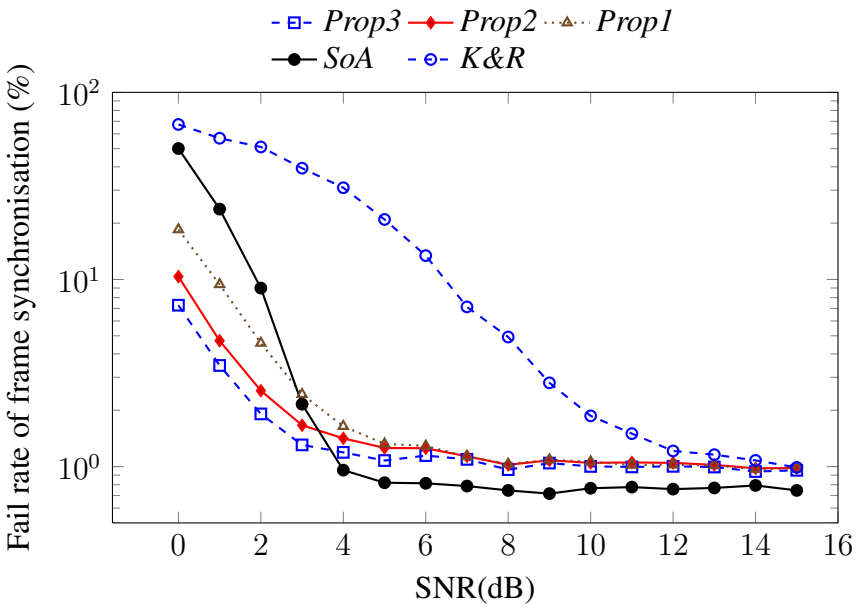


Figure 4.8: Frame synchronization performance of various methods in an SUI1 channel with respect to SNR.

results for *Prop1*, *Prop2* and *Prop3* saturate and become almost identical at higher SNRs, as does *K&R*.

**Performance with Large Frequency Offset**

The proposed method is designed to work even with large frequency offsets. Fig. 4.10 explores performance over 100,000 tests where the frequency offset is chosen randomly (with uniform distribution) from -10 to +10 times the subcarrier spacing for each test, in an AWGN channel. This experiment is specifically designed to investigate the robustness of STO estimation in large CFO conditions and shows that the proposed methods still maintain good performance. *K&R* exhibits some accuracy degradation, however all methods are seen to outperform *SoA*. The proposed methods are therefore seen to offer robustness of STO estimation against large CFO. This robustness against large CFO is because the proposed metric is computed on magnitude values that are insensitive to phase errors. CFO estimation is not evaluated here because large CFO estimation requires an integer CFO estimator that is investigated in the subsequent section.

In summary, simulation results show that *Prop3* and *Prop2* have better STO and CFO estimation accuracy compared to other methods. *Prop3* enjoys just a small improvement in terms

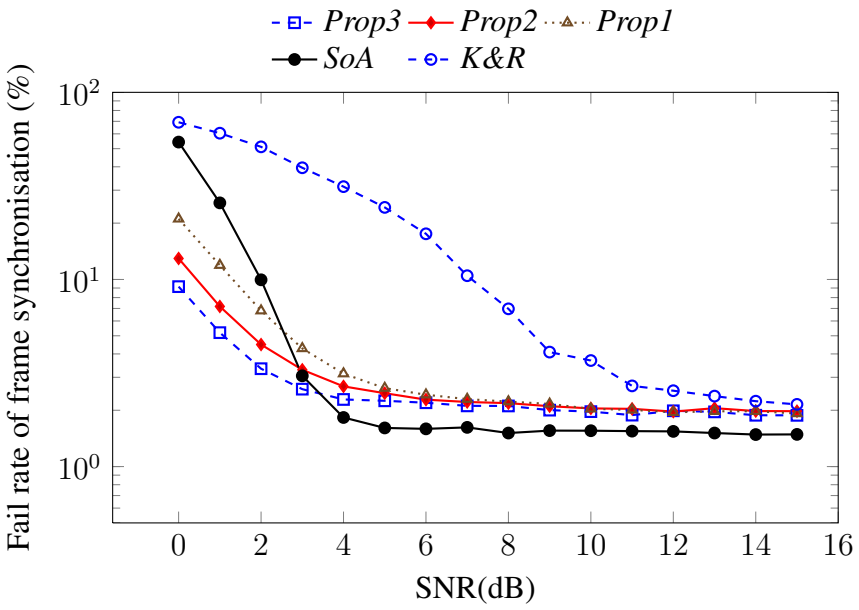


Figure 4.9: Frame synchronization performance of various methods in an SUI2 channel with respect to SNR.

of STO estimation compared to *Prop2* but this improvement incurs a significant hardware cost because the length of received samples for estimation must increase from $2L$ to $3L$. Given the results described in this sub-section, *Prop2* is selected as an implementation candidate in the subsequent sub-section, where the trade-off between accuracy and hardware cost is explored in more detail.

### 4.3.4 Hardware Implementation

This sub-section discusses the implementation of, and investigates the hardware optimization of, the proposed method in terms of word size. The target FPGA is a low-power Xilinx Spartan-6 XC6SLX45 device, with ISE 13.2 used to evaluate both hardware resource and power consumption. The results illustrate the trade-off between hardware consumption and the accuracy of the proposed method. The above sub-section already revealed that the performance of *K&R* is generally worse than the other methods, moreover it requires many complex multiply operations to compute the cross-correlation for the $P$ Metric. For this reason, *K&R* is not implemented, only the proposed method and *SoA* are compared here, in terms of hardware resources, power consumption, and accuracy. As mentioned above we set $C = 2L$ since *Prop2* consis-

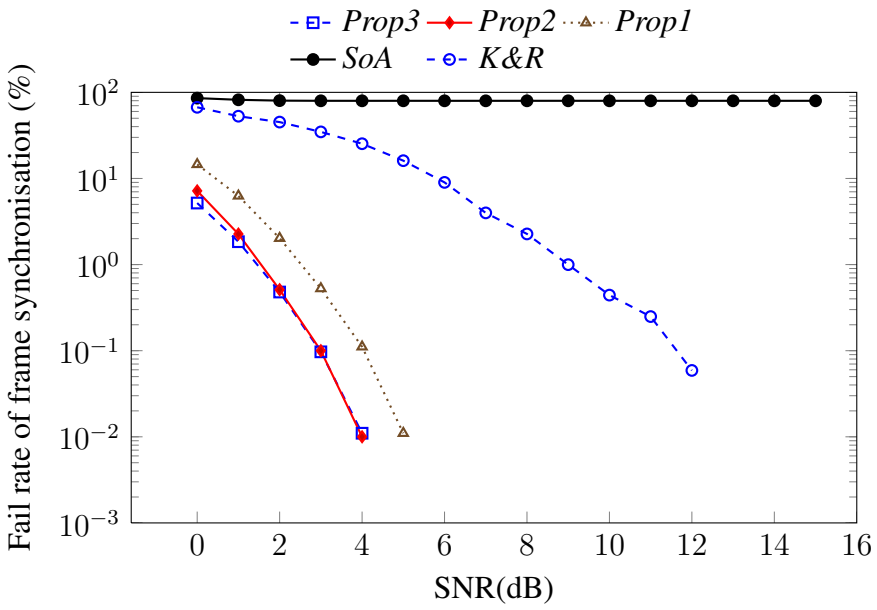


Figure 4.10: Performance of frame synchronization in an AWGN channel with uniform random frequency offset varying from -10 to 10 times carrier spacing, with respect to SNR.

tently showed performance close to *Prop3*, and significantly better than *Prop1*. It therefore offers a good balance between increased hardware cost and performance.

### Implementation of Conventional Synchronizer

First, let us consider a conventional synchronizer. We have implemented this as shown in Fig. 4.11, following the efficient implementation methods presented in [6, 7, 14, 15]. The $P$ and $R$ timing metrics in (4.3) and (4.2), respectively are computed using delay and summation, whilst a very efficient signed bit multiplier [5] is used to significantly reduce the computational overhead of the cross correlation for fine timing synchronization.

We assume a 16-bit two's complement fixed point representation with 15 fractional bits (i.e. Q1.15 format in Q-notation). The auto-correlation and squared amplitude of received samples are computed with a complex multiplier IP core that uses DSP slices. Results are scaled to Q1.15 to reduce resource usage in subsequent pipeline stages. Since this synchronizer is for the IEEE 802.16 preamble, the length of the delay is 64 samples. Each element is scaled to be less than 1, guaranteeing that the final summation result is less than 64, needing just 7 bits for representation in two's complement. Hence, the $P$ and $R$ values are represented in Q7.15

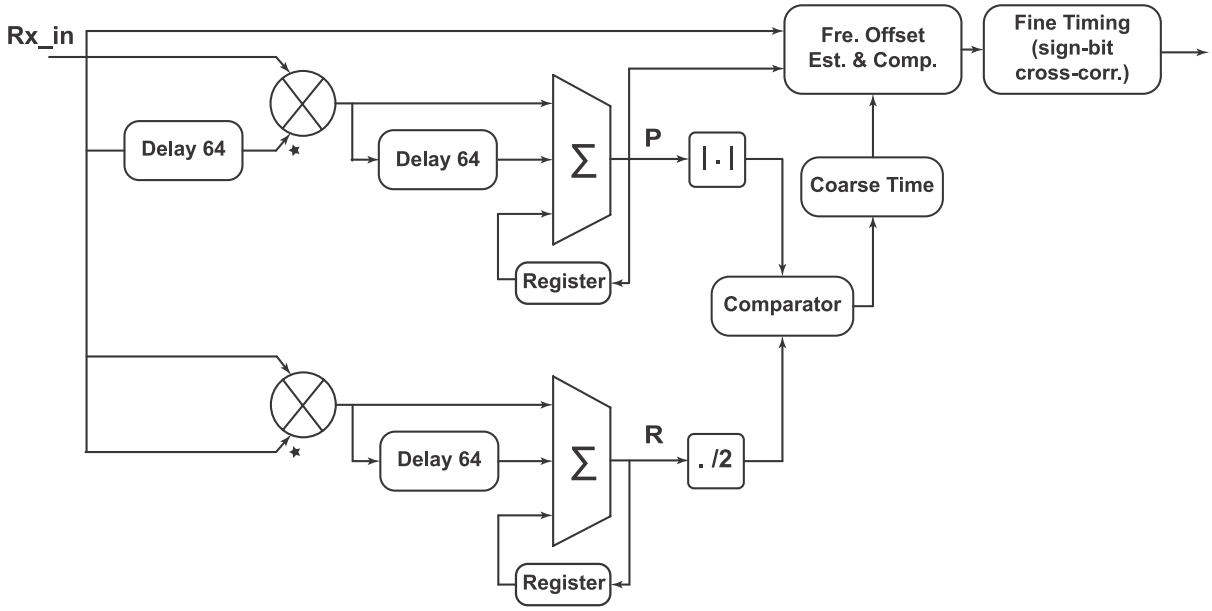


Figure 4.11: Architecture of the conventional synchronization FPGA implementation.

format. For the $|P|$ metric, the auto-correlation uses a complex multiplier IP core to multiply the current received sample and the 64th delayed sample. The magnitude of the $P$ metric is approximated to reduce hardware complexity as per [14]. For the $R$ metric, another complex multiplier is used to compute the squared magnitude of the received sample. The threshold is commonly chosen to be 0.5, which can be implemented using a right shift by 1 bit [11] instead of using a multiplier. After the frame is detected, the $P$ metric is used to estimate and correct the fractional CFO. A CORDIC IP core is used to determine the phase of the $P$ metric, and to derive the estimated fractional CFO. The received samples are then compensated using phase accumulation and phase rotation. These compensated samples are now used to determine fine timing synchronization.

### Implementation of Proposed Synchronizer

The architecture for our proposed method is shown in Fig. 4.12.

The format of received samples is similar to the conventional design. The $R'$ metric is determined using an energy correlator as illustrated in Fig. 4.13. The number of samples used to compute it is 128 and since the 128 samples of the short preamble are arranged in two identical spans of 64 samples each, the correlator only needs 64 taps. Eqn. (4.17) shows the

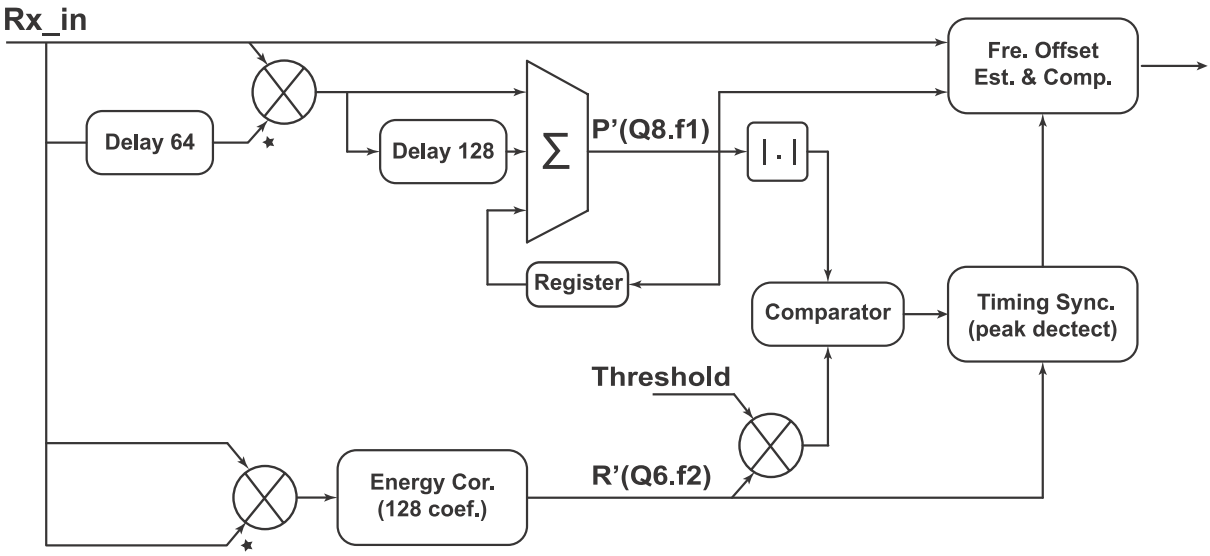


Figure 4.12: Architecture for the proposed synchronization method implemented on FPGA.

derived equations of this optimization:

$$\begin{aligned} R'(z) &= I(z)A_{127} + I(z)z^{-1}A_{126} + ... + I(z)z^{-63}A_{64} \\ &\quad + I(z)z^{-64}A_{63} + ... + I(z)z^{-127}A_0, \\ &= I(z)A_{63} + I(z)z^{-1}A_{62} + ... + I(z)z^{-63}A_0 \\ &\quad + I(z)z^{-64}A_{63} + ... + I(z)z^{-127}A_0], \\ &= (I(z) + I(z)z^{-64})A_{63} + ... \\ &\quad + (I(z) + I(z)z^{-64})z^{-63}A_0, \\ &= I(z)(1 + z^{-64})A_{63} + z^{-1}(I(z)(1 + z^{-64})A_{62} \\ &\quad + z^{-1}(... + z^{-1}I(z)(1 + z^{-64})A_0))), \end{aligned} \tag{4.17}$$

where $I$ is the squared amplitude of the received sample and $A_n$ denotes the normalised squared amplitude of known preambles. Following this, a multiplierless correlator, as described in detail in [16], is used to compute the output, shown in block diagram form in Fig. 4.13.

The values of $A_n$ are quantised to 0.5 and a shift/multiplex operation replaces the multiplication. $R'$ is always positive and smaller than twice the sum of all $A_n$ elements, i.e., 63. So the $R'$ metric requires just 6 bits to represent its integer part.

The $P'$ metric is computed in an identical way to the conventional one, but the number of samples used in the computation is 128 instead of 64 since $C = 2L$. So, the moving summation

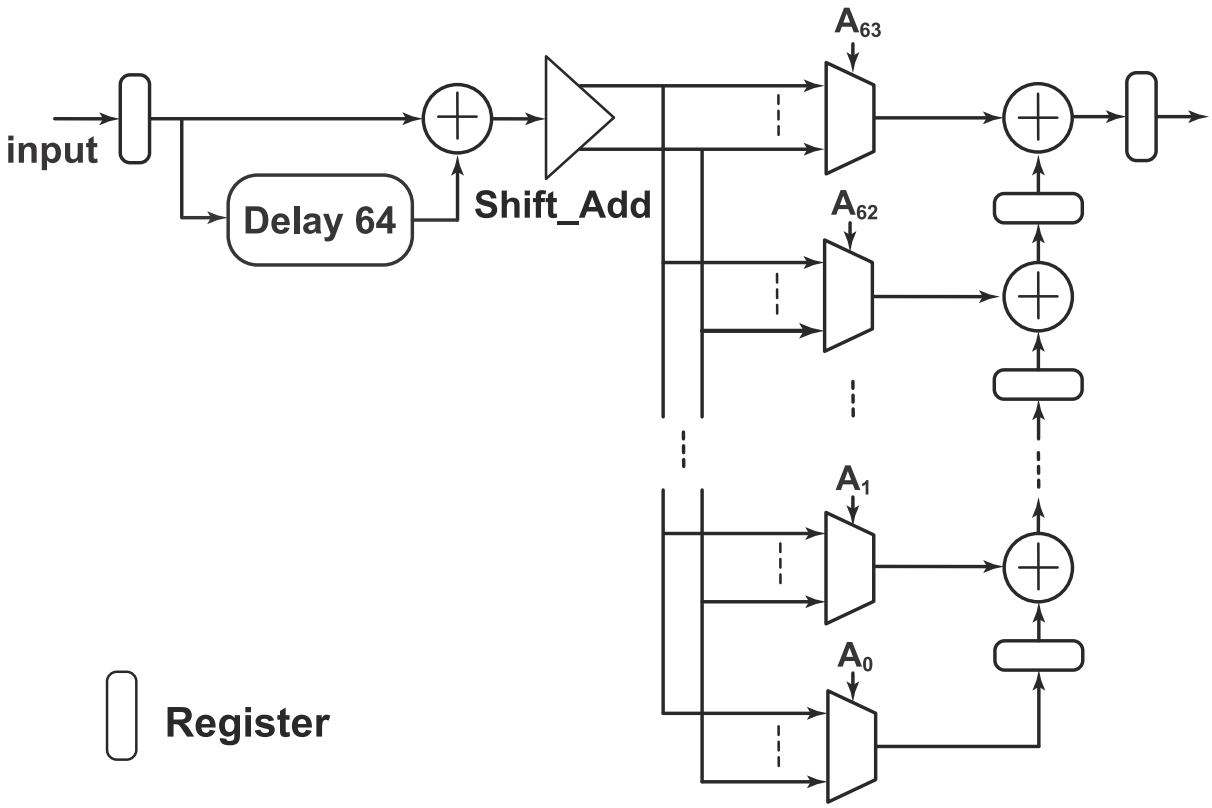


Figure 4.13: Implementation of energy correlator on FPGA.

uses a delay buffer of 128 samples, and the result value now requires 8 bits to represent the integer part.

**Effect of Reduced Precision**

Let $f1$ and $f2$ be the number of bits representing the fractional part for computing $P'$ and $R'$ respectively. Thus $P'$ has fixed point format Q8.$f1$ and $R'$ has fixed point format Q6.$f2$. The effects of reducing the number of bits used to represent these fractional components of $P'$ and $R'$ will now be investigated, with the aim of optimising the reduced precision against hardware savings.

Recall the results of CFO estimation in the simulation sub-section where *Prop2* exhibited a significant improvement in CFO estimation accuracy compared to *SoA* thanks to an increase in the evaluation window size obtained by setting $C = 2L$. This requires more multiplications, leading to increased hardware cost to compute the $P'$ metric. Reducing $f1$ allows a reduction in this extra hardware cost by making each individual computation simpler. The question

is to determine how much reduction in $f1$ can be sustained without losing the performance advantage enjoyed by *Prop2*.

Table 4.1: Resources required for computing $P'$ on FPGA with different word lengths, $Q1.f1$.

| $f1$ | FF | LUT | BRAM | DSP |
|---|---|---|---|---|
| *Prop-15b* | 304 | 427 | 96 | 3 |
| *Prop-7 b* | 240 | 323 | 64 | 3 |
| *Prop-6 b* | 232 | 311 | 60 | 3 |
| *Prop-5 b* | 224 | 297 | 56 | 3 |
| *Prop-4 b* | 216 | 269 | 52 | 3 |

To understand precisely how $f1$ reduction can save hardware, Table 4.1 details the hardware resource required for implementing five representative word sizes. Meanwhile, Fig. 4.14 plots CFO performance curves for the corresponding sizes of $f1$. It is clear from the graph that all degraded precision computations perform well – even the lowest Q1.4 precision computation (*Prop-4b*) can outperform *SoA* below about 7dB. The optimal choice for this range of SNRs is probably $f1 = 7bits$ (*Prop-7b*) which suffers just a slight decrease in accuracy compared to the 'full length' 15 bit version (*Prop-15b*). This allows a reduction of 21%, 24%, and 33% in the number of flipflops (FF), Look-up-tables (LUT), and BRAM blocks, respectively. Moreover, *Prop-7b* still achieves excellent performance when compared to the state of the art method, *SoA*.

Table 4.2: Resources required for computing $R'$ on FPGA with different word lengths, $Q1.f2$.

| $f2$ | FF | LUT | BRAM | DSP |
|---|---|---|---|---|
| *Prop-15b* | 1404 | 1017 | 16 | 2 |
| *Prop-7 b* | 884 | 633 | 8 | 2 |
| *Prop-6 b* | 818 | 589 | 7 | 2 |
| *Prop-5 b* | 753 | 537 | 6 | 2 |
| *Prop-4 b* | 689 | 504 | 5 | 2 |

Similarly, the optimized tradeoff between the accuracy of STO estimation and hardware usage for computation of the $R'$ metric is obtained based on reducing $f2$. In this case, Table 4.2 reveals the corresponding reduction achieved in computation resources and Fig. 4.15 plots the frame synchronization fail rate with SNR for several values of $f2$. The performance of the proposed method with $f2 = 6bit$, *Prop-6b*, can be seen to be almost identical to the 'full

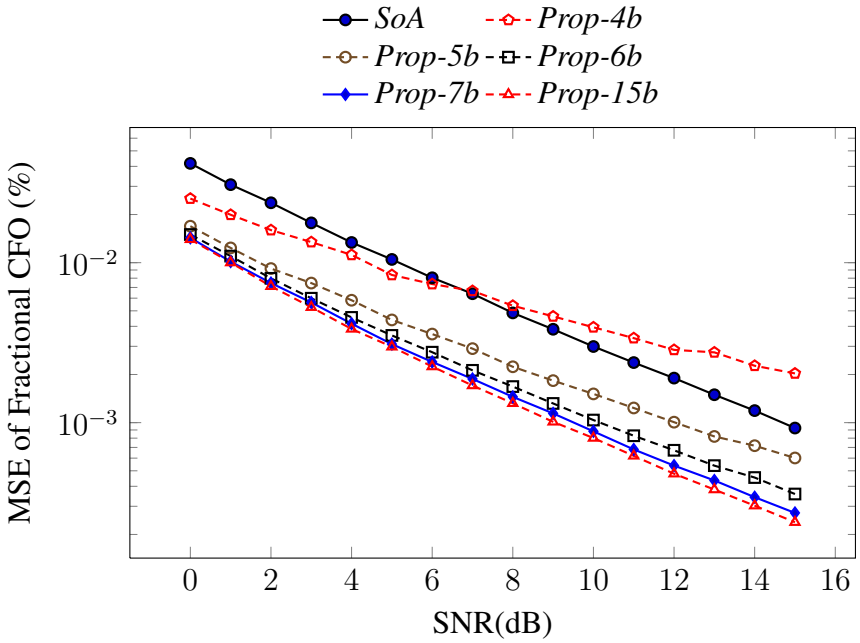


Figure 4.14: Performance of CFO estimation in an AWGN channel against SNR, with different numbers of fractional bits used in the computation of $P'$.

length' computation using 15 fractional bits, *Prop-15b*. Overall, *Prop-6b* achieves much more accurate estimation compared to the state of the art method, *SoA*. Reducing $f2$ to 6 bits allows a reduction of 41%, 42%, and 56% in the number of FFs, LUTs, and BRAM blocks.

### Optimized Alternatives

The preceding results are now used to define four alternative implementations of the proposed method to compare against the state-of-the-art method, *SoA* which uses full length Q1.15 arithmetic. These alternatives are namely:

- *Prop-A1*: a non-optimized instance of the proposed method with both $f1$ and $f2$ set to 15.
- *Prop-A2*: only $P'$ is optimized with $f1 = 7$ while $f2$ remains set to15.
- *Prop-A3*: only $R'$ is optimized with $f2 = 6$ while $f1$ remains set to 15.
- *Prop-A4*: both $P'$ and $R'$ are optimized by setting $f1 = 7$ and $f2 = 6$.

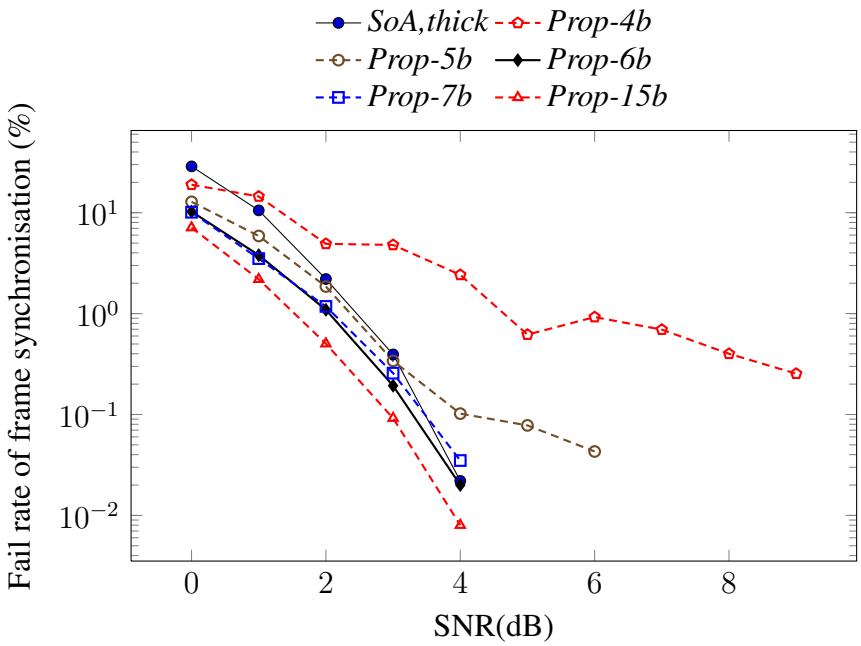


Figure 4.15: Performance of frame synchronization in an AWGN channel against SNR, with different numbers of fractional bits used in the computation of $R'$.

Table 4.3: Total resources consumed by a full word length implementation of *SoA* and four reduced complexity instances of the proposed method. Dynamic (Dpwr) and quiescent power (Qpwr) consumption are reported in mA. Maximum frequency is reported in MHz.

| | Slices | BRAM | DSP | Qpwr | Dpwr | Frequency |
|---|---|---|---|---|---|---|
| *SoA* | 930 | 112 | 13 | 37 | 41 | 121 |
| *Prop-A1* | 1000 | 118 | 14 | 37 | 43 | 133 |
| *Prop-A2* | 923 | 86 | 14 | 37 | 39 | 142 |
| *Prop-A3* | 869 | 109 | 14 | 37 | 38 | 137 |
| *Prop-A4* | 777 | 77 | 14 | 37 | 35 | 142 |

Table 4.3 reports the overall hardware resources required for these instances of the synchronizer, as well as detailing the power consumption of each. It should be noted that the CFO estimation and frame synchronization performance of these instances can be seen by choosing the corresponding word length from plots of $P'$ & $R'$ in Figs. 4.14 & 4.15 respectively. In other words, all instances have been simulated and reported in the previous plots. From the table, it is evident that reducing word length can yield a significant reduction in both hardware requirement and power consumption. The fully optimized alternative, *Prop-A4*, achieves a reduction of 16.4%, 31.2%, and 14.6% in the number of occupied slices, BRAMs, and in

dynamic power consumption, when compared to *SoA*. The maximum frequencies of the SoA and proposed method implementations are also reported. The required frequency for baseband processing in IEEE 802.16 ranges from from 5.6 to 22.4 MHz, and this requirement is easily met by all tested implementations.

Table 4.4 details the comparison between *SoA* and *Prop-A4* in terms of their constituent building block resources (where the function names in this table correspond to the block diagrams of Figs. 4.11 and 4.12).

Table 4.4: Detailed Resource Comparison between two synchronization methods.

| | Function | FF | LUT | BRAM | DSP |
|---|---|---|---|---|---|
| *SoA* | $\|P\|$ metric | 303 | 427 | 64 | 3 |
| | $R$ metric | 168 | 186 | 16 | 2 |
| | CFO comp | 1478 | 1517 | 0 | 8 |
| | Coarse time | 7 | 33 | 0 | 0 |
| | Fine time | 942 | 1009 | 32 | 0 |
| | **Total** | **2898** | **3172** | **112** | **13** |
| *Prop--A4* | $\|P'\|$ metric | 240 | 323 | 64 | 3 |
| | $R'$ metric | 818 | 600 | 7 | 2 |
| | CFO comp | 1467 | 1515 | 0 | 8 |
| | Time sync | 66 | 98 | 6 | 1 |
| | **Total** | **2591** | **2536** | **77** | **14** |

The function named 'CFO comp', which performs frequency offset estimation and compensation, clearly consumes the largest amount of hardware in both methods. The $R'$ metric computation in *Prop-A4* uses more hardware than the computation of $R$ in *SoA*. However fine timing estimation, 'Fine time', takes a large proportion of the total hardware cost in *SoA* while the alternative in the proposed method (timing synchronization, known as 'Time sync'), requires much less hardware.

## 4.4 Summary

Although the state of the art synchronisation methods achieve good performance when the CFO is in the range of fractional CFO estimation, they can not work with larger CFO. Some methods employ cross-correlation for the time synchronisation; these method are robust to large CFO and can obtain acceptable performance at low SNR. However, the much higher computational

resources needed for cross-correlation tend to make such methods unsuitable for hardware implementation. The methods have been presented to improve upon these drawbacks of previous reported works. The method takes the advantage of period and energy distribution characteristics of the preamble to perform time synchronisation. The synchronisation performance results, obtained through simulation, demonstrates good performance and the robustness to large CFO. Although the method just estimates and compensates the fractional CFO, the method still performs well with the larger CFO may presenting the interger CFO and an enhanced OFDM synchronisation method that provides an efficient and low cost IFO estimation will be presented in the next chapter.

# Chapter 5

# A CFO Estimation Method for OFDM Synchronisation

# Chapter 6

# A Spectrum Efficient Shaping Method

# Chapter 7

# A Novel Architecture for Multiple Standard Cognitive Radios

# Chapter 8

# Conclusion and Future Work

# References


[1] T. Schmidl and D. Cox, "Robust frequency and timing synchronization for OFDM," *IEEE Transactions on Communications*, vol. 45, pp. 1613 –1621, Dec. 1997.

[2] C. N. Kishore and V. U. Reddy, "A frame synchronization and frequency offset estimation algorithm for OFDM system and its analysis," *EURASIP Journal on Wireless Communications and Networking*, vol. 2006, pp. 1–16, 2006.

[3] L. Hanzo and T. Keller, *OFDM and MC-CDMA : A Primer*. Wiley-IEEE Press, 2006.

[4] T. H. Pham, I. V. McLoughlin, and S. A. Fahmy, "Robust and Efficient OFDM Synchronisation for FPGA-Based Radios," *Circuits, Systems, and Signal Processing*, vol. 33, pp. 2475–2493, Aug. 2014, Springer.

[5] L. Schwoerer, "VLSI suitable synchronization algorithms and architecture for IEEE 802.11a physical layer," in *IEEE International Symposium on Circuits and Systems, 2002. ISCAS 2002*, vol. 5, pp. V–721, 2002.

[6] F. Manavi and Y. Shayan, "Implementation of OFDM modem for the physical layer of IEEE 802.11a standard based on Xilinx Virtex-II fpga," in *Vehicular Technology Conference, 2004. VTC 2004-Spring. 2004 IEEE 59th*, vol. 3, pp. 1768 – 1772, May 2004.

[7] J. Guffey, A. Wyglinski, and G. Minden, "Agile radio implementation of OFDM physical layer for dynamic spectrum access research," in *IEEE Global Telecommunications Conference, 2007. GLOBECOM '07*, pp. 4051 –4055, Nov. 2007.

[8] Z. Huang, B. Li, and M. Liu, "A proposed timing synchronization method for 802.16e downlink," in *International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS), 2010*, pp. 1 –4, Dec. 2010.

[9] A. Recio and P. Athanas, "Physical layer for spectrum-aware reconfigurable OFDM on an FPGA," in *13th Euromicro Conference on Digital System Design: Architectures, Methods and Tools (DSD), 2010*, pp. 321 –327, Sept. 2010.

[10] IEEE std.802.16-2009, *IEEE Standard for Local and Metropolitan Area Networks Part16: Air Interface for Fixed Broadband Wireless Access Systems*.

[11] T.-H. Kim and I.-C. Park, "Low-power and high-accurate synchronization for IEEE 802.16d systems," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, pp. 1620 –1630, Dec. 2008.

[12] K.-W. Yip, Y.-C. Wu, and T.-S. Ng, "Design of multiplierless correlators for timing synchronization in IEEE 802.11a wireless LANs," *IEEE Transactions on Consumer Electronics*, vol. 49, pp. 107 – 114, Feb. 2003.

[13] V. Erceg, K. V. S. Hari, and M. S. Smith, "Channel models for fixed wireless applications," *Tech. Rep. IEEE 802.16a-03/01*, Jul. 2003.

[14] Q. Liu, G. Constantinides, K. Masselos, and P. Cheung, "Data-reuse exploration under an on-chip memory constraint for low-power FPGA-based systems," *IET Computers & Digital Techniques*, vol. 3, no. 3, pp. 235–246, 2009.

[15] K. Wang, J. Singh, and M. Faulkner, "FPGA implementation of an OFDM-WLAN synchronizer," in *IEEE International Workshop on Electronic Design, Test and Applications (DELTA)*, pp. 89 – 94, Jan. 2004.

[16] T. H. Pham, S. A. Fahmy, and I. V. McLoughlin, "Low-power correlation for IEEE 802.16 OFDM synchronisation on FPGA," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, pp. 1549–1553, August 2013.