This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing

It Can Drain out Your Energy: An Energy-Saving Mechanism against Packet Overhearing in High Traffic Wireless LANs

Tao Xiong, Junmei Yao, Jin Zhang, Student Member, IEEE, Wei Lou, Member, IEEE

Abstract—Energy efficiency is a critical issue of wireless devices. As the packets are broadcast to the devices in the wireless transmission medium, all active neighboring devices have to spend their energy receiving the packets though the packets are not addressed to them, which is called as the *packet overhearing problem*. The real-world traffic trace analysis reveals that the energy cost on the packet overhearing accounts for the majority of the devices' energy inefficiency in high traffic wireless local area networks (WLANs). In this paper, we propose a novel sample-address sample-duration (SASD) scheme to solve the energy inefficiency of the packet overhearing problem. By adding a new SASD header, which contains the critical information, in front of the data packet at the PHY layer, the SASD enables the devices to discern the required information in the energy-saving downclocking mode. Consequently, the non-destination devices of the packet can switch to the sleeping mode to avoid the packet overhearing problem. We demonstrate the feasibility of the SASD through hardware experiments and evaluate its energy-saving performance through ns-2 simulations. The results show that the SASD can greatly outperform the existing approaches in the high traffic WLAN scenario.

Index Terms—Energy efficient, packet overhearing problem, downclocking, wireless LANs.

1 Introduction

Wireless networks are becoming an indispensable part of people's daily life due to the low cost in the deployment of networking infrastructures and the high availability to mobile users. Various types of mobile devices, such as smartphones, tablets and laptops, can easily access the Internet through wireless local area networks (WLANs) nowadays. However, as mobile devices are mostly powered by batteries which can be rapidly drained out by wireless communications, it remains a challenging issue about how to achieve the energy efficiency of mobile devices in WLANs.

Generally speaking, due to the broadcast nature of wireless transmission media, WiFi protocols would employ a CSMA mechanism to sense the channel, which inevitably causes energy inefficiency of *channel sensing* and *packet overhearing*. The continuously channel sensing for sending packets or waiting for incoming packets would cause energy inefficiency of mobile devices, because the energy cost of the channel sensing is comparable to that of the packet reception [1], [2]. Moreover, when a packet is broadcast in a wireless channel, all WiFi devices which overhear the packet transmission would receive and decode the packet even if they are not the intended receiver of the packet. Despite the packet would be eventually dropped at the MAC layer after the receiver checks the packet's receiver MAC address, the energy spent on receiving this packet has already been wasted, and this energy inefficiency of the unnecessary packet overhearing can be even worse if the wireless devices are placed in a high traffic WLAN [3]–[6].

Some well-known solutions to alleviate the energy inefficiency are to build MAC-layer power saving mode (PSM) protocols [3], [4], [7]–[10] which schedule WiFi stations to periodically wake up to exchange control packets with an access point (AP). Then, the stations decide whether to stay active to receive data packets from the AP or to switch to the sleeping mode to save their energy. The PSM protocols can classify the network traffic and reduce the energy inefficiency of the channel sensing for waiting for incoming packets. However, they cannot reduce the energy inefficiency caused by the channel sensing for sending packets and overhearing unnecessary packets [3], [6].

Recently, another energy-efficient approaches have conducted at the PHY layer. As the power consumption of a wireless device is known to be proportional to its voltage-square and sampling rate, it can significantly save the energy by either putting the device in a low voltage or downclocking its sampling rate during the channel sensing and packet overhearing [2], [5], [6], [11]–[13]. A state-of-the-art work–E-MiLi [6] was proposed to let the WiFi chipsets work at the downclocking mode during the channel sensing and packet overhearing, and restore to the fullclocking mode when detecting the intended packets. Although E-MiLi alleviates the energy consumption on the packet overhearing, it still needs to spend extra energy sampling unnecessary packets continuously. The energy performance of the combination of the PSM and E-MiLi is even worse than the E-MiLi as the stations have to further spend extra energy processing the control packets used in the PSM protocols.

To effectively reduce the energy consumption on channel sensing and packet overhearing, in this paper we propose a sample-address sample-duration (SASD) scheme, which leverages the advantages of both the downclocking mode and sleep-

- *Tao Xiong, Junmei Yao, Jin Zhang and Wei Lou are with the Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong and The Hong Kong Polytechnic University Shenzhen Research Institute, Shenzhen, P. R. China. E-mail: {cstxiong, csjyao csjzhang, csweilou}@comp.polyu.edu.hk.*
- *This work was supported in part by Hong Kong RGC GRF grant PolyU-521312, HKPU grants 4-BCB6 and G-YBJU, and NSFC grant No. 61272463. Tao Xiong and Junmei Yao are co-primary authors. Wei Lou is the corresponding author.*

Tao Xiong与Junmei Yao为共同第一作者，Wei Lou为通信作者。

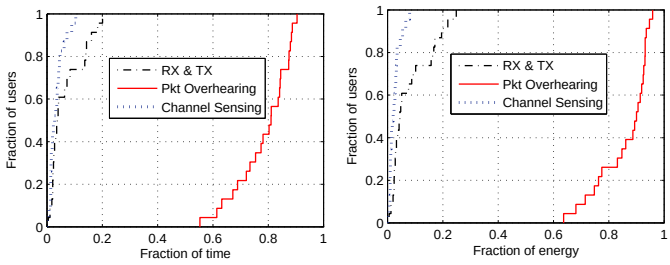
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing

ing mode to save the energy consumption of wireless devices. The SASD can not only put a device in the downclocking mode during the channel sensing as E-MiLi does, but also turn the device into the sleeping mode when it detects that the transmitted packet is not addressed to it, so as to further save its energy consumption, especially in the high traffic WLAN scenario. To achieve this, two critical packet information, the packet's receiver MAC address and transmission duration, shall be well carried in the packet frame at the PHY layer and can be correctly detected without the whole packet being received, specifically in the downclocking mode. The SASD implements a MAC address mapping method that uses a local unique fixed-length bit sequence, called as *sample-address* (SA), to represent the receiver's MAC address. The destined receiver, when receiving a packet header, can detect its corresponding SA, then switches to the fullclocking mode to receive the incoming packet. On the other hand, to reduce the energy cost on the packet overhearing, the SASD adopts a catalogued duration time method that maps all possible packet transmission durations into different catalogues, and uses different global unique fixed-length sample sequences, called as *sample-durations* (SDs), to represent these transmission duration catalogues. Stations which cannot detect the SA would turn to detect and identify the SD in the downclocking mode, switch to the sleeping mode for the catalogued duration time to avoid the packet overhearing. As the SASD is mainly implemented at the PHY layer, it requires no change of the data packets at the MAC layer, but just adds an SASD header in front of the data packet at the PHY layer. Thus, it is compatible with existing MAC-layer energy-saving protocols.

The contributions of this paper are summarized as follows:

- We introduce the SASD scheme, which can deliver the packet's receiver MAC address and transmission duration information in one shot at the PHY layer. Moreover, the SASD scheme leverages the advantages of both the downclocking mode and sleeping mode for achieving energy efficiency.
- We design, implement, evaluate and analyze the SASD scheme on a 3-node testbed with USRP2/GNURadio. The results demonstrate the feasibility of the SASD to solve the packet overhearing problem in the downclocking mode.
- To reveal the performance improvement, we conduct simulations for various network topology scenarios in ns-2. The results show that our SASD can outperform the existing approaches significantly in term of energy usage with slight negative influence on the network throughput in various WLAN scenarios.

The rest of this paper is organized as follows: In Section 2, we motivate the SASD scheme by exploring the energy inefficiency of the packet overhearing problem. In Section 3, we detail the architecture design of the SASD mechanism. In Section 4, we reveal the hardware implementation details, along with the experiment results. In Section 5, we give the SASD's ns-2 simulation results under different network scenarios. In Section 6, we briefly introduce prior related work. Lastly, in Section 7 we give the conclusion of the paper.



(a) Time (b) Energy

Fig. 1. Time and energy characteristics of mobile devices in a high traffic wireless scenario (The data come from the traces given in [14]).

2 Motivation

In this section, we motivate the SASD by exploring the energy inefficiency of the packet overhearing problem, especially in high traffic networks. We detail the time and energy costs on the channel sensing and packet overhearing through real world WiFi traces, respectively.

Generally, among the list of a wireless device's energy cost, the idle listening accounts for the major cause of the energy inefficiency [6]. Furthermore, the energy cost on the idle listening can be divided into two parts [15]: (1) the energy cost on packet overhearing, and (2) the energy cost on channel sensing for sending packets or waiting for incoming packets. We describe the energy cost on the three aspects as below, and analyze the energy cost on these aspects by using the real world WiFi traces [14], respectively:

- **Energy cost on transmitting (TX) and receiving (RX)**: the energy cost on transmitting or receiving packets.
- **Energy cost on packet overhearing**: the energy cost on receiving and decoding packets which are not addressed to the station. The energy spent on this aspect is similar to that on TX & RX. However, the packets will be eventually dropped at the MAC layer, and this kind of energy cost is considered as energy inefficiency.
- **Energy cost on channel sensing for sending packets or waiting for incoming packets**: the energy cost on sensing the wireless channel before sending packets or monitoring the channel for unpredictable incoming packets. Although the channel sensing is necessary in wireless networks to avoid the packet collision, we consider this energy cost on the channel sensing as energy inefficiency.

Note that we neglect the energy cost when the station is in the sleeping mode, as it is quite small compared with the three aspects in high traffic WLANs.

Fig. 1(a) shows the normalized fraction of time in the three aspects discussed above. More than 90% of the devices (the fraction of users is from 0.1 to 1.0) spend over 63% of the total time on the packet overhearing while more than 80% of the devices (the fraction of users is from 0 to 0.8) spend less than 5% of the total time on the channel sensing. We believe that it is because the traces were taken from a busy network. In contrast, around 70% stations (the fraction of users is from 0 to 0.7) spend less 10% time on sending and receiving packets.

With respect to the energy cost on the three aspects, we adopt a typical Atheros card's energy profile measured by [16]: TX (1550mW), RX (1350mW), channel sensing (900mW). Fig. 1(b) shows that in a high traffic WLAN, 80% stations

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing

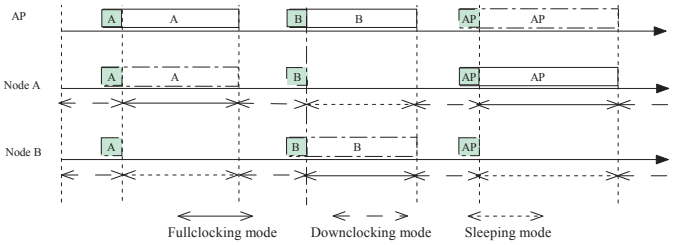


Fig. 2. An overview of the SASD mechanism. (The full-line box denotes the packet sending and dot-dash-line box denotes the packet receiving.)

(the fraction of users is from 0.2 to 1.0) spend more than 70% energy on the packet overhearing. This significant energy inefficiency of the packet overhearing would rapidly drain out mobile devices' battery power.

The above analysis reveals that the packet overhearing accounts for the majority of wireless devices' energy cost in busy networks. Admittedly, in lightly loaded networks, the stations will spend less energy on the packet overhearing but more energy on the carrier sensing. However, we believe that the packet overhearing is quite common in current WLANs due to the rapid growth of data traffic. If this energy inefficiency of the packet overhearing can be effectively eliminated, the energy efficiency of wireless devices will be clearly improved in current high traffic WLANs.

3 Architecture and Design

3.1 Overview of the SASD Scheme

Different from the PSM protocols or other PHY layer energy saving mechanisms, the SASD tries to enable the stations to detect and identify the packet's receiver MAC address and transmission duration information in one shot without receiving the whole packet at the PHY layer. It leverages the advantages of both the sleeping mode and downclocking mode: (1) the stations by default work in the downclocking mode on the packet overhearing to save the energy cost; (2) the stations which are the transmitter or the intended receiver will work in the active fullclocking mode; (3) the stations which are not involved in the packet transmitting or receiving can turn to the sleeping mode to further save the energy cost on the packet overhearing. Please note that the radio only needs several μs to switch to the sleeping mode [16], [17], the value is negligible comparing with the packet transmission duration, which is in the *ms* level. The energy cost for the switching process from sleep to active or inverse is also negligible in this paper. We should clarify that SASD is only designed for client stations in WLANs, APs still work in the fullclocking mode as they always have power supply and the energy cost is not a problem to these APs. Meanwhile, SASD cannot be applied to save the energy cost for hotspots in WLANs, these hotspots act as APs and should work in the fullclocking mode.

As shown in Fig. 2, both nodes A and B work in the downclocking mode to listen to the channel. When there is a data transmission from AP to node A, node A turns to the fullclocking mode to detect the packet, while node B turns to the sleeping mode to save energy. At the end of this data transmission, both A and B return to the downclocking mode for packet overhearing and prepare for the next data

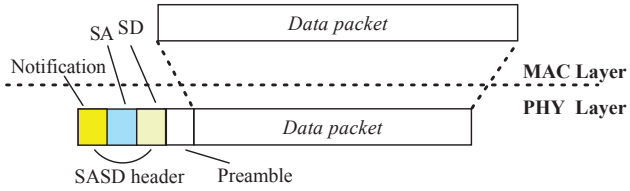


Fig. 3. The SASD data frame at the PHY layer.

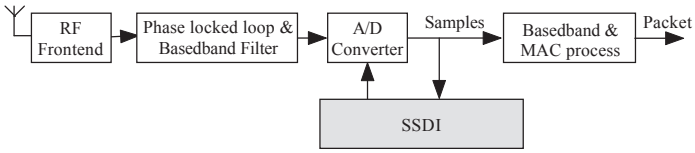


Fig. 4. The architecture of the SASD scheme at the receiver side. Only the gray SSDI block is a new component.

transmission. As the stations can save considerable energy consumptions when sensing the channel in the low-power downclocking mode and turning their states into the sleeping mode whenever possible, the SASD scheme can significantly increase the energy efficiency of the entire wireless network.

To achieve this energy efficiency goal, the main challenges are how to carry and discern the two key MAC layer information, the packet's receiver MAC address and transmission duration, at the PHY layer without requiring the receiver to fully receive and decode the whole packet, specifically in the downclocking mode.

The SASD does not change the data frame's structure at the MAC layer, but adds an SASD header in front of the data packet at the PHY layer. As shown in Fig. 3, the SASD header contains three fixed-length fields: the notification field, the sample-address (SA) field and the sample-duration (SD) field. The notification field is used to notify all wireless devices that there is an incoming packet even when the devices work in the downclocking mode. The SA and SD fields are used to carry the two MAC layer information which are mentioned above.

At the transmitter side, the transmitter will generate the SASD header in front of each packet and send them together. At the receiver side, the receiver adopts the cross-correlation to detect and identify the three fields in the downclocking mode. To decode the information carried in the SASD header, the receiver employs a new decoder, called *SASD detection and identification decoder* (SSDI) (the gray block in Fig. 4), to obtain the SA and SD information from the received packet at the PHY layer and then to inform the A/D converter to switch to the fullclocking mode or sleeping mode accordingly: When the device detects a notification of an incoming packet in the downclocking mode, it uses the SSDI to detect and identify the SA field to determine whether the incoming packet is addressed to it or not. If yes, it restores to the fullclocking mode to receive the packet. Otherwise, it turns to detect and identify the SD field and changes to the sleeping mode to save the energy cost on the packet overhearing. Please note that the SASD header will be attached in front of each data packet even if it is retransmitted, the receiver has the same SSDI procedure for the retransmitted packet. The SSDI procedure is listed in Algorithm 1.

In the following subsections, we first detail the detection and identification process in the fullclocking mode, then describe

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing

Algorithm 1 SSDI Procedure

Input: Incoming samples from the A/D converter in the downclocking mode.

1: **while** detect and identify the notification field in the incoming samples **do**
2: Cut the samples in the fixed-length SA field.
3: **if** the MAC address information extracted from the samples in the SA field matches the receiver's own address **then**
4: Inform the A/D converter to switch to the fullclocking mode to receive the packet.
5: **else**
6: Cut the samples in the fixed-length SD field.
7: **if** the packet transmission duration information can be extracted from the samples in the SD field **then**
8: Inform the station to switch to the sleeping mode to save the energy cost on the packet overhearing.
9: **end if**
10: **end if**
11: **end while**
12: Go to Step 1.

the process in the downclocking mode.

3.2 Detection and Identification in the Fullclocking Mode

When a bit sequence is transmitted, the transmitter maps this bit sequence to a series of baseband data symbols. These symbols will be further resampled as a sequence of baseband samples that can be represented as:

$$x[n] = A[n]e^{j\phi[n]}, \tag{1}$$

where $A[n]$, $\phi[n]$ are the magnitude and the angle of the n^{th} resampled sample. Then, these samples are gone to the D/A converter and carried by the analog signal. It is noted that the resampled samples are closely related to the input symbol. When the input symbol is changed, the corresponding resampled samples are also changed.

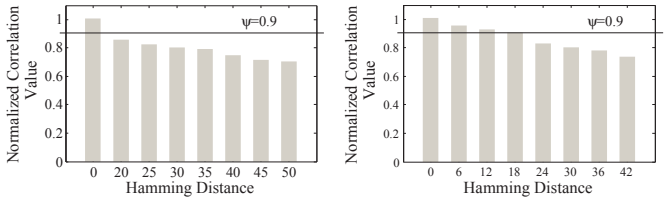
When the signal is transmitted over the wireless channel, at the receiver side, the RF down-converter samples the incoming signal, and the A/D converter derives a stream of discrete complex baseband samples. However, in practice, due to the hardware constraint and wireless channel effect, e.g. frequency offset, sampling offset, and inter-symbol interference, the received samples differ from the transmitted samples. Mathematically, the received complex baseband samples can be represented as:

$$y[n] = Hx[n]\cdot e^{j2\pi\Delta f\Delta t} + w[n]. \tag{2}$$

Here, H is a complex number representing the channel coefficient between the transmitter and receiver, $w[n]$ is the Gaussian noise at the n^{th} sample, and $e^{j2\pi\Delta f\Delta t}$ denotes the effect caused by the frequency offset Δf between the transmitter and receiver over time Δt.

Let $x_{kn}[i]$ denote the i^{th} baseband sample resampled from the known bit sequence and $\overline{x_{kn}[i]}$ be the complex conjugate of $x_{kn}[i]$. To detect the known bit sequence, the receiver can apply the cross-correlation to calculate the correlation value $C(l)$ as:

$$C(l) = \left|\sum_{i=0}^{l-1} \overline{x_{kn}[i]} \cdot y_{in}[i]\right|. \tag{3}$$



(a) The conventional correlation. (b) The enhanced correlation.

Fig. 5. The Hamming distance requirements for two detection and identification methods. (bit sequence size = 160 bit, $SINR \approx 0dB$)

Here, l is the length of known samples which are resampled from the known bit sequence and $y_{in}[i]$ is the received sample that contains the known sample $x_{kn}[i]$, interference sample $y_{int}[i]$ and white noise $w[i]$, i.e., $y_{in}[i] = H \cdot x_{kn}[i] + y_{int}[i] + w[i]$. As the known sample sequence is contained in the incoming signal, by compensating the frequency offset [18], Eq. (3) can be written as:

$$\begin{aligned} C(l) &= \left|\sum_{i=0}^{l-1} \overline{x_{kn}[i]} \cdot (H \cdot x_{kn}[i] + y_{int}[i] + w[i])\right| \\ &= \left|\sum_{i=0}^{l-1} H \cdot |x_{kn}[i]|^2 + O(l)\right|. \end{aligned} \tag{4}$$

Here,

$$O(l) = \sum_{l=0}^{l-1} \overline{x_{kn}[i]} \cdot y_{int}[i] + \sum_{l=0}^{l-1} \overline{x_{kn}[i]} \cdot w[i].$$

Since $x_{kn}[i]$ is independent of either the interference sample $y_{int}[i]$ or the noise $w[i]$, as long as l is large enough, $O(l) \approx 0$ [18]. The $C(l)$ can be written as:

$$C(l) \approx |H| \cdot \sum_{i=0}^{l-1} |x_{kn}[i]|^2. \tag{5}$$

The correlation value $C(l)$ is compared with the threshold β_{kn}: If $C(l) \geq \beta_{kn}$, the known sample sequence is presented; otherwise, the sequence is considered absent. Normally, threshold β_{kn} can be defined as $\beta_{kn} = \psi \cdot l \cdot RSSI_{kn}$, where ψ is a constant and $RSSI_{kn}$ is the estimated received signal strength indicator of the known sequence signal. Thus, the presence of the sequence is shown as:

$$NC(l) = \frac{C(l)}{l \cdot RSSI_{kn}} \geq \psi. \tag{6}$$

We call $NC(l)$ as normalized correlation value.

It is known that the detection and identification would cause two kinds of errors, false negative error and false positive error. From Ineq. (6), lowering the ψ can minimize the false negative error rate to 0. On the other hand, increasing the Hamming distance between two known bit sequences can decrease the false positive error rate. As the ψ and false positive/negative errors will influence the SASD's detection and identification performance, we detail the two errors in Section 4 through hardware experiments, while ψ can be set to $0.55 \sim 0.95$. For example, we conduct cross correlation between the transmitted known bit sequence and any one of other known bit sequences with various Hamming distances on the USPR2 platform, the results in Fig. 5(a) show that, with the threshold $\psi = 0.9$, once the Hamming distance between two known bit sequences is larger than 20, the correlation process can correctly detect and identify the transmitted known bit sequence, making the false positive error rate close to 0 (when $SINR \approx 0dB$).

The notification field is a global unique known bit sequence. One node needs to continuously conduct cross correlation

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing

between this sequence and the received signal to determine whether there is an incoming packet. If there is, it will detect the information in the SA and SD fields using the SSDI procedure. We will give detailed design for the SA and SD fields in the following parts.

3.2.1 How to process the receiver MAC address?

In SASD, the AP would maintain a *local* unique known bit sequence pool and a MAC address mapping table. When a new station joins the AP's wireless network, during the association process, the AP randomly selects an unused known bit sequence from the pool to represent the new station's MAC address and puts them in the MAC address mapping table, the AP then informs this station both its known bit sequence and AP's known bit sequence. Please note that we make the stations work in the fullclocking mode to accomplish the association process; after that, they work by default in the downclocking mode to save energy.

Once the AP wants to send a data packet to the station, the AP puts the associated known bit sequence of the station in the SA field of the SASD header. Also, if the station wants to send a packet to the AP, it puts AP's known bit sequence in the SA field. The station would use the SSDI procedure to check whether the incoming packet is addressed for it or not. If yes, the station switches to the fullclocking mode for receiving the packet. Otherwise, it tries to detect and identify the SD field in the downclocking mode.

Here we want to discuss the scenario when there are multiple APs in a network, which may always exist in actual networks. When two stations associated to different APs are assigned the same bit sequence, one station may mistakenly turn to the fullclocking mode if the packet is addressed to another one, thus consume more energy. We consider that two methods can be used to solve this problem: (1) Each AP maintains a different set of bit sequences, and assigns each client a unique bit sequence independently. (2) As APs are always connected through a backbone network [19], they can share a bit sequence pool and exchange information with each other about whether a bit sequence has been used before allocating it. The first method is simple and easy to deploy; the second one is more flexible to suit different scenarios. They can be selected according to the actual network's characteristics.

3.2.2 How to process the packet transmission duration?

One of the SASD's main challenge pertains to the delivery of the packet transmission duration at the PHY layer, which can enable those non-receivers to switch to the sleeping mode to avoid the packet overhearing and to switch back to the downclocking mode after the packet transmission is completed. In this part, we will first illustrate how to design known sequences to represent the packet transmission duration, then illustrate how to convey the transmission duration through known sequences.

A) How to represent the transmission duration?

The packet transmission duration has a maximum value according to the 802.11 standard: The duration (PLCP frame size) cannot exceed $5.484ms$ in both 802.11n and 802.11ac [20], while the duration in the 802.11a is related with the packet length L_p and the transmission rate R. As $L_p \leq 4095bytes$ and R has eight values from $6Mbps$ to $54Mbps$ in 802.11a [7], the maximum transmission duration is about $\frac{4095\times8}{6M} = 5.46ms$. We set the maximum packet transmission duration $T_{max} = 5.484ms$, then divide T_{max} into N catalogues, and map each catalogued duration time T^i_{cata} to a *global* unique known bit sequence. T^i_{cata} can be calculated as follows:

$$T^i_{cata} = \frac{T_{max}}{N} \cdot i, \tag{7}$$

where i is the catalogued duration time index of the global unique known bit sequence and $1 \leq i \leq N$. As the actual packet transmission duration could fall in any catalogue, the AP and stations must preload all the global unique known bit sequences for generating or discerning the correct catalogued duration time index.

To prove that Eq. (7) is reasonable, here we will give detailed analysis to show that the packet transmission durations can be uniformly divided within $[0, T_{max}]$, while the analysis will be given from 802.11ac, 802.11n and 802.11a, respectively.

According to 802.11ac, all the data frames should be transmitted through the aggregate MPDU (A-MPDU), even if the A-MPDU has only one frame in it. The length of an A-MPDU is $2^{13+Exp} - 1$ bytes, where $Exp = 0, \ldots, 7$, which allow the A-MPDU length to range from $8K$ to $1M$ bytes [20]. Meanwhile, the 802.11ac has a series of data transmission rates, which are determined by the MCS, the bandwidth, and the number of spatial streams (SS), as shown in Table 1. That means, the 802.11ac supports up to 312 categories' data rates from $7.2Mbps$ to $6.933Gbps$.

MCS value	20MHz data rate (n SS)	40MHz data rate (n SS)	80MHz data rate (n SS)	160MHz data rate (n SS)
MCS0	7.2·n	15.0·n	32.5·n	65.0·n
MCS1	14.4·n	30.0·n	65.0·n	130.0·n
MCS2	21.7·n	45.0·n	97.5·n	195.0·n
MCS3	28.9·n	60.0·n	130.0·n	260.0·n
MCS4	43.3·n	90.0·n	195.0·n	390.0·n
MCS5	57.8·n	120.0·n	260.0·n	520.0·n
MCS6	65·n	135.0·n	292.5·n	585.0·n
MCS7	72.2·n	150.0·n	325.0·n	650.0·n
MCS8	86.7·n	180.0·n	390.0·n	780.0·n
MCS9	/	200·n	433.3·n	866.7·n

TABLE 1
The data rate matrix in 802.11ac ($Mbps$). $n = 1, \ldots, 8$.

The 802.11n also supports frame aggregation, and the maximum PSDU (the PLCP payload size) is 65535 bytes [20], which is denoted as L_{pmax} = 65535bytes here. Comparing with the 802.11ac data rates in Table 1, it only supports MCS0~MCS8, 20MHz/40MHz channel bandwidths, and up to 4 spatial streams. That means, the 802.11n supports up to 72 categories' data rates from $7.2Mbps$ to $720Mbps$.

The 802.11a recommends that the maximum PSDU is 4095 bytes (L_{pmax} = 4095bytes), and recommends eight data transmission rates: 6,12, 18, 24, 36, 48 and 54 $Mbps$.

To get the distribution of the possible packet transmission durations for each standard, we generate 8 packet lengths for 802.11ac when $Exp = 0, \ldots, 7$; we also generate $N_L = \lceil \frac{L_{pmax}}{100} \rceil$ packet lengths for 802.11n and 802.11ac, each packet has the length $j \cdot 100$, where $j = 1, \ldots, N_L$; we then get the transmission durations for all the combinations of the packet length and transmission rate in each standard. The cumulative functions of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing

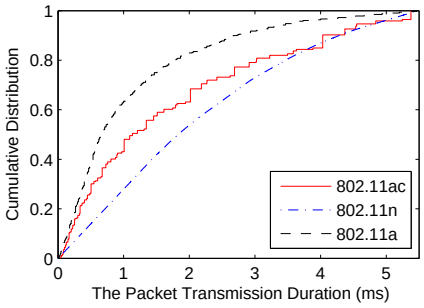


Fig. 6. The cumulative distribution of the transmission duration.

the transmission durations are shown in Fig. 6, which indicates that the packet transmission durations of each standard can be uniformly divided within $[0, 5.484ms]$; thus, it is rational to calculate the catalogued duration time through Eq. (7).

B) How to convey the transmission duration?

When the AP (or station) prepares to send a packet, it obtains the packet transmission duration T_D, calculates the catalogued duration time index $i = \left\lfloor \frac{N \cdot T_D}{T_{max}} \right\rfloor$. After that, the AP (or station) puts the corresponding known bit sequence in the SD field. Note that the catalogued duration time function could keep the stations sleeping slightly shorter than the actual data transmission duration, so as to avoid unfairness to the network. However, this design may in turn slightly increase SASD's energy cost, enlarging the catalogue value N can alleviate this cost clearly.

When the station receives the packet, the SASD adopts the enhanced correlation approach (Algorithm 2), which not only considers that the correlation value must excess the threshold but also compares the correlation values between different catalogued duration time indexes to detect and identify the correct catalogued duration time index. First, the station correlates the incoming samples with N different known sample sequences corresponding to the N catalogued duration time indexes in parallel and picks up the normalized correlation results which exceed the threshold ψ_{SD} as the candidates. After that, it picks up the largest correlation result among the candidates and maps it to the catalogued duration time index i. The station gets the index i, calculates the catalogued duration time T^i_{cata}, and derives the sleeping time T_{sleep} by adding T^i_{cata} with a SIFS time (T_{SIFS}) and an ACK duration (T_{ACK}). Then, the station switches to the sleeping mode during the sleeping time T_{sleep}.

Algorithm 2 Enhanced Correlation Approach

Input: Samples in the SA field.

1: Correlate the samples with N different known sample sequences corresponding to the N catalogued duration time indexes; pick up the normalized correlation values which exceed ψ_{SD} as the candidates.
2: Compare these candidates and pick up the maximum value $NC_i(l)$.
3: Get the catalogued duration time index i of the selected one; calculate the catalogued duration time T^i_{cata}; set the sleeping time $T_{sleep} = T^i_{cata} + T_{SIFS} + T_{ACK}$.
4: Switch to the sleeping mode for T_{sleep}.

We also conduct the enhanced correlation process between the transmitted known bit sequence and any one of other known bit sequences with various Hamming distances on the USRP2 platform, the results in Fig. 5(b) show that, with the threshold $\psi = 0.9$, the transmitted known bit sequence (the Hamming distance is 0) can be correctly identified as it has the maximum correlation value, although the correlation values of the other known sequences with the Hamming distance of 6 and 12 also exceed the threshold ψ.

3.3 Detection and Identification in the Downclocking Mode

In Section 3.2, we show that when the A/D converter works in the fullclocking mode, the receiver MAC address and packet transmission duration information can be detected and identified at the PHY layer. In this section, we explore that, even if the A/D converter works in the downclocking mode, these approaches are still functional to generate and discern the receiver MAC address and packet transmission duration information at the PHY layer.

3.3.1 Is the SASD still functional in the downclocking mode?

At the receiver side, according to the Nyquist-Shannon sampling theorem, to fully reconstruct the signal that was sent out by the transmitter, the receiver has to use twice of the transmitting bandwidth to resample the incoming signal. Thus, the A/D sampling frequency is $f_r = 2B$, where B is bandwidth.

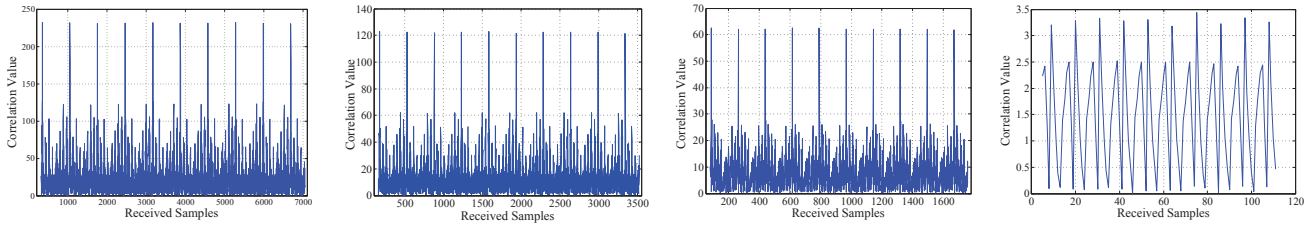
When the A/D converter reduces the sampling rate in order to save the energy, it also decreases the number of the samples that the SASD can use to calculate the correlation value. Based on Eq. (5), the correlation value after downclocking would be:

$$C(\frac{l}{\tau}) \approx \frac{1}{\tau} \cdot |H| \cdot \sum_{i=0}^{l-1} |x_{kn}[i]|^2, \tag{8}$$

where τ is the downclocking rate of the A/D converter. Remember that Eq. (5) can only be true when l is large enough to make $O(l) \approx 0$. Thus, Eq. (8) also needs to meet the requirement that $\frac{l}{\tau}$ is large enough to make $O(\frac{l}{\tau}) \approx 0$. Fig. 7 shows the relationship between the correlation value $C(\frac{l}{\tau})$ and downclocking rate τ from the USRP2 hardware experiment. The known bit sequence is transmitted for 10 times. Another traffic flow with randomly generated bits is also transmitted simultaneously to make the sequence's SINR$\approx 0dB$. It is shown that when $\tau \leq 4$, $C(\frac{l}{\tau}) \approx \frac{C(l)}{\tau}$. When $\tau = 64$, due to the reason that $\frac{l}{\tau}$ cannot make $O(\frac{l}{\tau}) \approx 0$, Eq. (8) cannot be held anymore. However, 10 correlation spikes can be detected in Fig. 7(d). It evidently shows that by using the cross-correlation, even when the A/D converter works in the downclocking mode, the correlation value $C(\frac{l}{\tau})$ can still be used to detect the known bit sequence. Consequently, the SASD can use the known bit sequences to deliver the SA and SD information in the downclocking mode.

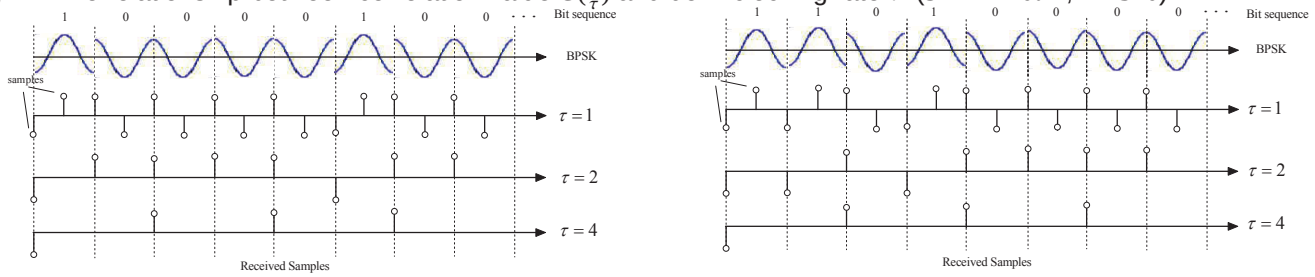
As we know, the difference between two bit sequences can be measured by their Hamming distance. In the fullclocking mode, every different bit in these two different bit sequences is fully sampled into different sample sequences, which guarantee the difference of their correlation values in the fullclocking mode. However, this difference cannot be guaranteed when the A/D converter works in the downclocking mode, i.e., the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing



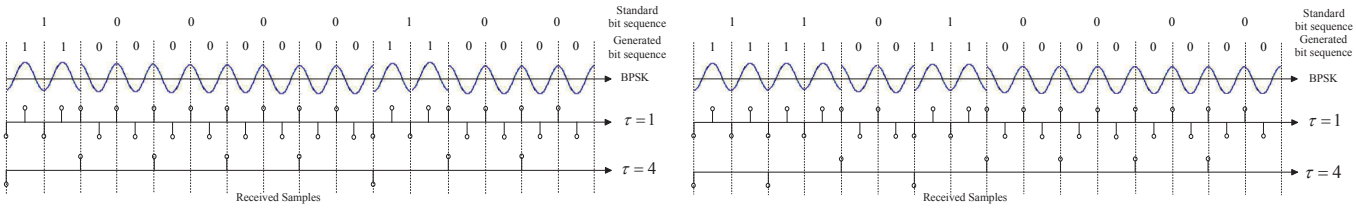
(a) $\tau = 1$ (b) $\tau = 2$ (c) $\tau = 4$ (d) $\tau = 64$

Fig. 7. The relationship between correlation value $C(\frac{l}{\tau})$ and downclocking rate τ. ($SINR \approx 0dB$, $l = 320$)



(a) The bit sequence "10000100". (b) The bit sequence "11010000".

Fig. 8. The received samples of two bit sequences under different A/D's downclocking rate τ. Here, we assume the A/D starts sampling at the first place.



(a) The generated bit sequence "1100000000110000". (b) The generated bit sequence "1111001100000000".

Fig. 9. The received samples of two generated bit sequences under A/D's downclocking rate.

different samples which the A/D converter samples in the fullclocking mode would be lost. Thus, different bit sequences may lead to the same sample sequence in the downclocking mode. Consequently, the identification approach, especially the enhanced correlation approach would fail to distinguish these various known bit sequences.

As $f_r = 2B_s$[1] and the samples are closely related to the incoming symbol, when the A/D works at the downclocking rate $\tau = 2$, the receiver can still get different sample sequences to help the SSDI identify the known bit sequences. As shown in Fig. 8, two bit sequences ("10000100" and "11010000") are sampled into different sample sequences when the A/D works at the fullclocking rate. When the A/D uses the downclocking rate $\tau = 2$, the resultant samples of two bit sequences are still different samples. However, if the A/D goes to the downclocking rate $\tau = 4$, the resultant samples of two bit sequences would be the same. At that time, the SASD scheme fails to identify these two bit sequences.

3.3.2 How to generate known bit sequences at a higher downclocking rate?

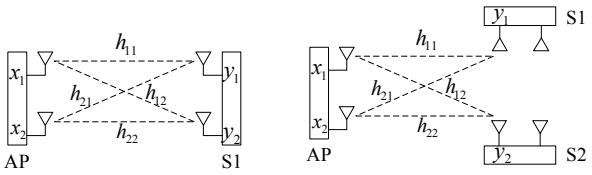
To generate the known bit sequences at a higher downclocking rate, e.g. $\tau = 4$, we employ the interpolation method to generate a new known bit sequence from each fullclocking standard bit sequence. As the higher downclocking rate will cause some bits to miss sampled, the interpolation will duplicate the last sampled bit to fill the missed bits to guarantee the difference of the received samples at the higher downclocking rate. E.g., in Fig. 9, when $\tau = 4$, every two bits will only generate one sample, which means that one bit is missed sampled. For each bit in the two standard bit sequences ("10000100" and "11010000"), the interpolation method will duplicate it to two bits, resulting in two new bit sequences "1100000000110000" and "1111001100000000". Apparently, the new bit sequences can keep the difference of the received samples when $\tau = 4$, but with the cost that the bit sequence's length has $\frac{\tau}{2}$ times increasing. Therefore, a higher τ can increase stations' energy efficiency in the packet overhearing, but will in turn lead to $\frac{\tau}{2}$ times more overhead for the cross correlation process and for transmitting the sequences, which may reduce the network throughput to some extent.

3.3.3 How to identify the first sampling place?

When the A/D converter works in the downclocking mode, the A/D cannot guarantee to sample the bit sequence at the first sampling place, which inevitably forces the SSDI to check all possible downclocking sample sequences with the incoming sample sequence to locate the first sampling place of the sequence. By choosing the downclocking sample sequence with the largest correlation value, the SSDI is aware of the first sampling place of the sequence in the downclocking mode.

1. The relationship between the bandwidth and symbol rate can be written as $B = B_s(1 + \alpha)$, where B_s is the symbol rate and α is the roll-off factor of a low-pass filter. Normally, $\alpha \geq 0$. In this paper, we assume the bandwidth is fully used, which means we take $\alpha = 0$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing



(a) The SU-MIMO scenario. (b) The MU-MIMO scenario.

Fig. 10. The example of SASD in a MIMO system.

From what we have discussed above, the SASD scheme can work in the downclocking mode when the difference requirement of the sample sequences can be guaranteed. Particularly, when the A/D takes the downclocking rate $\tau = 2$, the known bit sequences which are used at the fullclocking mode can be directly employed in the downclocking mode with downclocking rate $\tau = 2$. Moreover, if the A/D wants to work at a higher downclocking rate, the new known bit sequences can be generated from the fullclocking standard bit sequence by using the interpolation method.

3.4 Apply SASD to MIMO

Nowadays, the new standards such as 802.11n and 802.11ac are already widely deployed in WLANs to increase the data transmission rate through wider bandwidth, MIMO (Multiple-Input-Multiple-Out) technology and so on. Especially, MU-MIMO (Multi-User MIMO) is proposed by the 802.11ac to make one AP communicates with multiple client stations simultaneously to further improve the network performance. In this part, we will demonstrate that SASD can still work in typical MIMO scenarios: the SU-MIMO (Single-User MIMO) scenario and the MU-MIMO scenario.

3.4.1 The SU-MIMO scenario

In the SU-MIMO systems, a transmitter equipped with multiple antennas communicates with a receiver that has multiple antennas. Consider a simple 2×2 MIMO example as shown in Fig. 10(a), two data packets x_1 and x_2 are transmitted from AP to a station S1 through the antenna array. x_1 is transmitted from AP's first antenna while x_2 is transmitted from the second one. The received signals y_1 and y_2 at S1 are:

$$\begin{aligned} y_1 &= h_{11}x_1 + h_{21}x_2, \\ y_2 &= h_{12}x_1 + h_{22}x_2, \end{aligned} \tag{9}$$

where $h_{11}, h_{12}, h_{21}, h_{22}$ are the channel coefficients, which are all complex numbers and can be obtained in the training phase. S1 can get y_1 and y_2 through solving the two equations, and then get x_1 and x_2 after demodulation.

Note that the station contains two demodulation modules, each of which is shown in the white blocks in Fig. 4. We name the module in S1 to demodulate y_1 as MODULE1, and name the other one as MODULE2. When applying SASD in this scenario, S1 should perform SASD at both MODULE1 and MODULE2. Meanwhile, AP adds the SASD headers that contain three new fields, notification, SA which represents S1's address, and SD in front of both x_1 and x_2. Then the process at S1 works as follows: both modules by default work in the downclocking mode for packet overhearing; when MODULE1 or MODULE2 performs SSDI and identifies that the address information in the SA field matches S1's address, it switches to the fullclocking mode; after receiving both the data packets y_1 and y_2 at the fullclocking mode, S1 can calculate x_1 and x_2 through the standard MIMO process. Similarly, when a neighboring station performs SSDI and identifies that this packet is not addressed to it, it continues to identify the SD field and switches to the sleeping mode in this duration to save energy.

3.4.2 The MU-MIMO scenario

In this part, we intend to use the scenario in Fig. 10(b) to demonstrate that SASD can also be applied in the MU-MIMO scenario.

As shown in Fig. 10(b), the AP intends to simultaneously transmit two data packets, x_1 and x_2, to S1 and S2 respectively. Before transmitting, AP should perform precoding to maximize the received signal's SINR at each station. We set the transmitted signals at the two antennas be $w_1 \cdot x_1$ and $w_2 \cdot x_2$, where w_1 and w_2 are the precoding weights. Then the received signals y_1 at S1 and y_2 at S2 are:

$$\begin{aligned} y_1 &= h_{11}w_1x_1 + h_{21}w_2x_2, \\ y_2 &= h_{12}w_1x_1 + h_{22}w_2x_2. \end{aligned} \tag{10}$$

According to the precoding algorithm, the part $h_{11}w_1x_1$ will dominate the signal strength of y_1, while $h_{22}w_2x_2$ will dominate the signal strength of y_2. Therefore, S1 and S2 can get x_1 and x_2 respectively through the normal demodulation process. When applying SASD to the system, AP adds the SASD headers that contain these three fields in front of x_1 and x_2, while the SA in x_1 indicates S1's address, and the SA in x_2 indicates S2's address. Then the processes at S1 and S2 work as follows: both S1 and S2 work by default in the downclocing mode; when S1 performs SSDI and identifies that the address information in the SA field of y_1 matches its address, it switches to the fullclocking mode to decode x_1. Similarly, after identifying the SA field of y_2, S2 switches to the fullclocing mode to decode x_2. The process of a neighboring node that receives these signals is the same as that in the SU-MIMO scenario. Please note that, even in the MU-MIMO scenario, the 802.11ac requires the concurrently transmitted signals x_1 and x_2 have the same packet length, which can be achieved through frame padding [20]. Thus, the SD fields in front of both x_1 and x_2 will be filled with the same known bit sequence.

We consider both scenarios in the 2×2 MIMO system can be applied to a more general $m \times n$ system. Thus, SASD can be applied to the latest 802.11 standards to improve the stations' energy efficiency.

4 Hardware Experiments

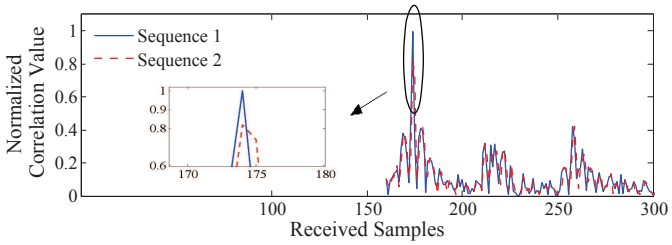
In this section, we reveal the hardware implementation and experimental methodology. As the three fields (notification field, SA field and SD field) in the SASD header are separately detected and identified, each field would have different requirements for the detection and identification.

4.1 Hardware Implementation and Experimental Methodology

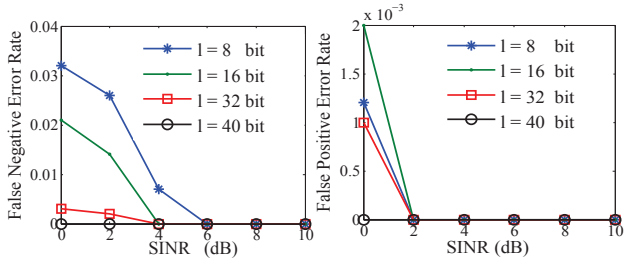
4.1.1 Hardware implementation

We have implemented the SASD scheme on a 3-node GNURadio/USRP2 testbed. Each node is a commodity PC connected

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing



(a) Find the A/D's first sampling place.



(b) The false negative error rate. (c) The false positive error rate.

Fig. 11. Notification detection performance ($\tau = 2$, and l = 8, 16, 32, 40 bits. Here, $SINR \approx RSSI_{SASD} - RSSI_{INT}$).

to a Universal Software Radio Peripheral 2 (USRP2) [21] with RFX2400 daughter-board. The RFX2400 operates at the 2.4GHz frequency band. All PCs are installed Ubuntu 10.04 and GNURadio [22]. As the USRP2 could not use the external clock to downclock the incoming signal, we have emulated the A/D downclocking procedure by adjusting the decimation rate. Please note that changing the decimation rate would not save much energy, but it would give the same outcome of baseband samples as that from the real A/D downclocking procedure. Also, we have considered the interference model, which is more common in real-world WLANs. We denote $RSSI_{SASD}$ and $RSSI_{INT}$ as the received signal strength indicators of the SASD header and interference, respectively. Thus, the SINR is calculated as $SINR \approx RSSI_{SASD} - RSSI_{INT}$.

The SASD uses the BPSK modulation/demodulation module, which is commonly used in the 802.11 standard. We have used the default GNURadio configuration for the SASD evaluation, i.e., at the transmitter side, the DAC rate is 400M samples/s, the interpolation rate is 100 (interpolation rate in the DAC chip is 4 and interpolation rate controlled by GNURadio is 25), and the number of samples per symbol is 2; at the receiver side, the ADC rate is 100M samples/s, and we set the decimation rate be 50 to emulate the scenario when A/D works at the downclocking rate 2. All the results in this section are based on the GNURadio/USRP2 testbed.

4.1.2 Experimental methodology

USRP2 has hardware delays in transmitting samples from the RF front-end to its connected commodity PC, and GNURadio also incurs some artificial software delay to process these samples. Thus, it is difficult to conduct a real time evaluation of the SASD in high bit rates. Hence, we have resorted to the trace-based evaluation [6], [18], where each node would save all the outgoing and incoming samples for off-line processing.

4.2 Notification Field's Detection and Identification

Before the SA and SD fields' detection and identification, the receiver has to sense if there is an incoming packet by correlating the notification field in the downclocking mode. Unfortunately, in the downclocking mode, the SSDI has to test τ possible known sample sequences to locate the first sampling place of the known sample sequences (Section 3.3). In our experiment, as we take the downclocking rate $\tau = 2$, the total number of possible downclocking sample sequences would be 2. After that, the SSDI is aware of the A/D's first sampling place, and it can directly calculate the right downclocking sample sequences of SA and SD during the SSDI procedure. Fig. 11(a) shows the normalized correlation values between the received samples and the two known downclocking sample sequences (sequence 1 and sequence 2) which are generated from the same known fullclocking sample sequence. By choosing the sequence with the largest normalized correlation value, the SSDI is aware of A/D's first sampling place when working in the downclocking mode.

In the SSDI procedure (Algorithm 1), the detection of the notification field takes the key place because it is the foundation of the detection and identification of the SA and SD fields. The false negative error is more related to the SINR, known bit sequence's length and threshold ψ. To minimize the false negative error, we took $\psi_N = 0.85$ during the detection of the notification field. Fig. 11(b) gives the false negative error rates for different l-length known bit sequences under the interference model. As shown in Table 2, for the false positive error, we can decrease this error rate by enlarging the Hamming distance between the data and known sample sequence. In our experiment, we have taken 40-bit length as the length of the notification field.

Hamming Distance	2	3	6	7
FPE Rate	0.0012	0.0020	0.0010	0
Sequence Length l (bit)	8	16	32	40

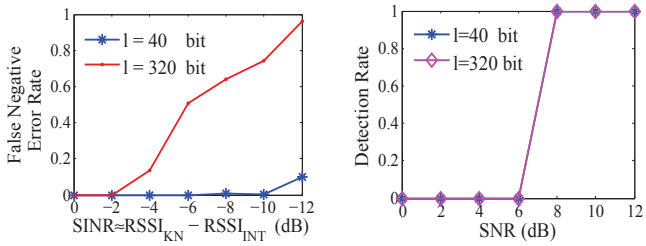
TABLE 2
The relationship between Hamming distance and false positive error (FPE) rate with various lengths of known bit sequence. (SINR = $0dB$, $\psi_N = 0.85$)

4.3 SA's Detection and Identification

In the SA field's detection and identification, as the SSDI has detected the notification field and identified the A/D's starting downclocking position, the SSDI can directly calculate the station's own SA known downclocking sample sequence for the SA field's detection and identification.

One of the key design issue of the SA field is to have a large set of sample sequences to support enough wireless stations. Fortunately, the SSDI can store all the sample sequences that represent the SA information. We raise the threshold ψ_{SA} for the SA's detection and identification. In our experiment, we set $\psi_{SA} = 0.9$ and SA's length to be 40 bits, the false negative error rate is nearly 0 even when $SINR = 0dB$. At the same time, when the Hamming distance reaches 5, the false positive error rate is also minimized to 0 ($SINR = 0dB$). However, this would cause the SSDI to extract the SA information from an incoming packet, switch to the fullclocking mode for receiving the whole packet, even if the SINR is not high enough to correctly decode the packet. However, because the packet is addressed to the station, we do not consider this booting-up and unnecessary packet receiving as the packet overhearing problem.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing



(a) The false negative error rate of the known bit sequences under the low SINR. ($\psi = 0.55, \tau = 2$)

(b) The detection rate of known bit sequences under the low SNR. ($\psi = 0.85, \tau = 2$)

Fig. 12. The detection and identification performance of known bit sequences with two sizes (40 bits, 320 bits) under the low SINR/SNR scenarios.

Considering all the factors above, we can design more than 40 different bit sequences for the SA field. We believe these known sequences are enough for the normal usage, even if the AP reserves some sequences as the broadcast and multicast addresses. Moreover, if the AP wants to support more stations, one method is to increase the length of the SA field to generate more bit sequences. Beyond that, the *Minimum-Cost Address Sharing* mechanism proposed in E-MiLi [6] can also be used to make the same sequence length support more stations, thus reducing the sequence transmission overhead.

4.4 SD's Detection and Identification

The station that needs to switch to the fullclocking mode after the SA field's detection and identification would use the transmission time of SD field to switch its mode. This switching time is various in the case of different devices, normally from 9.5 μs to 128 μs [6], which is enough to transmit the SD field in the SASD header. In our experiment, we have chosen 320-bit length as the SD field size to test the SASD scheme. Note that the SASD scheme is not restricted to this length, as the SD field can be set to different lengths to fit for the A/D's switching time. Thus, stations have to restore all possible lengths for the SD field's detection and identification. Table 3 gives the transmission times for various lengths of the SD field in standard 802.11 WLANs.

SD Field Length (bit) / Standards	80	160	240	320
802.11a/g	6.67	13.3	20	26.6
802.11b	7.27	14.55	21.82	29.09
	Time (μs)			

TABLE 3
Channel occupation time overhead.

Same as the SA field's detection and identification, the SASD can directly "cut" the samples in the SD field as the input for the enhanced correlation approach, and those samples are thought to carry the SD information only. Moreover, compared with the conventional correlation detection and identification, which only allows one correlation value to exceed the threshold, the enhanced correlation approach would enable several correlation values to exceed the threshold $\psi_{SD} = 0.9$ which significantly relaxes the Hamming distance requirements (as shown in Table 4). Consequently, the SASD can design more known bit sequences to minimize the catalogue cost in the enhanced correlation approach. In our USRP2 experiment, we have designed more than 320 different bit sequences for the 320-bit length SD field. Thus, the catalogue cost is below 6.3 μs in IEEE 802.11a. Also, considering all the factors above, the false negative/positive error rates of 320-bit length sequences are already close to 0 when $SINR = 0dB$.

SD Field Length (bit) / Method	80	160	240	320
Conventional Correlation	10	20	30	40
Enhanced Correlation	4	6	12	16
	Required Hamming Distance			

TABLE 4
The Hamming distance requirements between the conventional correlation and enhanced correlation approaches. (SINR = $0dB$, $\psi_{SD} = 0.9$)

4.5 Aggressive Model - Low SINR and Low SNR Scenarios

The above sections (4.2∼ 4.4) only reveal the scenario that the $RSSI_{SASD}$ of the SASD header is larger than the $RSSI_{INT}$ of the interference. However, the high interference is an inevitable situation in the high traffic WLANs [3]. Also, due to the signal fading effect, the SASD scheme may detect an SASD header even if the SNR is not high enough for decoding a packet. In this subsection, we discuss the aggressive model, i.e., how the SASD works under the low SINR and low SNR scenarios.

One critical issue caused by the low SINR is the rising of the false negative error rate, which is highly related to the SINR, known bit sequence's length l and threshold ψ. To minimize the false negative error rate, we use $\psi = 0.55$ in the low SINR scenario to evaluate the performance. Fig. 12(a) shows the false negative error rate under the low SINR for the known bit sequences with two sizes (40 bits, 320 bits and $\tau = 2$) used in the hardware experiment. Even though we have decreased the threshold ψ to 0.55, the missing rate of 40-bit known sequence would rise to 74% ($SINR = -8dB$), which means that the SASD would miss most of the chances to detect the notification field. Although the SASD can increase the length of the notification field to decrease the false negative error rate, the overall overhead of the SASD header would be increased. Considering the SASD works in the downclocking mode, this missing detection would not cause much energy waste on the packet overhearing, comparing with that in the fullclocking mode.

Another critical issue is the false positive error rate of the notification field. It would cause the unnecessary booting-up to the fullclocking mode due to the detection and identification of the SA and SD. To deal with this problem, we modify the correlation value normalization in Ineq. (6) as:

$$\frac{C(l)}{l \cdot RSSI} \geq \psi. \tag{11}$$

Here, we use the $RSSI$ of the incoming signal instead of the estimated $RSSI_{SASD}$. In the low SINR scenario, as $RSSI \approx RSSI_{INT} > RSSI_{SASD}$, the normalized correlation value would be much smaller under the low SINR. When the data meets the Hamming distance requirement in Table 2, the false positive error rate of the notification field can be minimized to 0.

Different from the low SINR scenario, the low SNR scenario would cause the SSDI to detect and identify the SA field, and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing

make the station switch to the fullclocking mode to decode the packet even if the SNR is not high enough to correctly decode the packet. To deal with this issue, we set a minimum detectable SNR_{min} [6] and modify Ineq. (11) as:

$$\frac{C(l)}{l \cdot Max(RSSI, SNR_{min} + RSSI_{en})} \geq \psi. \quad (12)$$

Here, $RSSI_{en}$ is the environment noise strength (typically, $-98 \sim -95dBm$). Fig. 12(b) gives the detection rate of known bit sequences with two sizes (40 bits, 320 bits and $\tau = 2$) when $SNR_{min} = 8dB$. Note that the modified Ineq. (12) can also be applied to the low SINR environments, in which situation the inequation is simplified to Ineq. (11). Meanwhile, Ineq. (12) does not change the detection performance of the decoder when the SINR/SNR is high, i.e., when $RSSI_{SASD} \gg RSSI_{INT} > (SNR_{min} + RSSI_{en})$, $RSSI \approx RSSI_{SASD}$. Also, in the low SNR scenario, as $RSSI < (SNR_{min} + RSSI_{en})$, Ineq. (12) would use $SNR_{min} + RSSI_{en}$ to normalize the correlation value.

4.6 Computational Overhead

The SSDI will introduce extra computational overhead to conduct the cross-correlation of the incoming signal sample by sample to identify the SASD's notification, SA and SD information. To analyze the computational overhead of the cross-correlation process in this paper, we let t_{MUL} denote the time spent for one multiplication step, and let t_{ADD} denote the time spent for one addition step. Then the time spent for each $C(l)$ calculation through Eq. (5) is $t_C = 4 \cdot l \cdot t_{MUL} + (4 \cdot l - 2) \cdot t_{ADD}$. For the notification field's detection and identification, one node should spend $\tau \cdot t_C$ at each sample, thus the overall computational overhead in this field is $\tau \cdot t_C \cdot N_s$, where N_s is the number of samples before the notification. This value is comparable with that in the process of the preamble detection and synchronization, which also deploys the cross-correlation at each sample to detect and synchronize the preamble (the overall overhead is $t_C \cdot N_s$). After detecting the notification field, as the SA and SD fields have fixed positions, the SSDI can directly calculate their correlation values without sample by sample, and the time spent for the SA and SD fields are $l \cdot t_C$ and $l \cdot t_C \cdot N$, respectively, both values will be significantly less than that of the preamble synchronization. As the overhead of the preamble synchronization is known to be acceptable [6], [18], [23], [24] and can be applied in real networks, we consider the SSDI process can also be implemented in current wireless devices.

5 Performance Evaluation

In this section, we give ns-2 simulation results that show the effectiveness of our SASD scheme to save the energy cost on the packet overhearing problem. We have implemented and tested the SASD in ns-2.34 under two different network scenarios: single AP with multiple stations and multiple APs with multiple stations. The energy profile used in the simulations is listed in Table 5, which is based on a commercial 802.11ac chipset Qualcomm Atheros 9880 (QCA9880), whose energy consumption has been tested by [16]. We make the energy cost in half sampling (W_{HS}) half of that in the channel sensing (W_{CS}). As the 802.11 standard recommends two mechanisms in the constantly awake mode, including PCS (stations use physical carrier sense to access the channel) and VCS (virtual carrier sense, stations exchange RTS and CTS before the data transmission), we compare our SASD with PCS, VCS, PCS+PSM, and VCS+PSM, the four combinations in the standard. We also compare SASD with E-MiLi and μPM, the current related works. In addition, as the energy cost that the mobile device spends on the wireless transmission part is much greater than that on the computation part, we have considered the energy cost on receiving the SASD header but ignored the energy cost introduced by the SSDI block in the SASD scheme. We have modified ns-2's source code at the PHY layer to support the SSDI procedure by using the hardware experiment results. We have also considered the time overhead caused by the SASD header as well as the catalogue cost of the SD field in the simulations.

Energy Mode	Value (mW)
TX	1550
RX	1350
Sleeping (SP)	1.8
Channel Sensing (CS)	900
Half Sampling (HS)	450

TABLE 5
Energy profile for ns-2 simulation.

As the above energy profile $(W_{TX}, W_{RX}, W_{SP}, W_{CS}, W_{HS})$ is used, the total energy usage per station can be calculated as:

$$E = W_{TX} \cdot T_{TX} + W_{RX} \cdot T_{RX} + W_{SP} \cdot T_{SP} + W_{CS} \cdot T_{CS} + W_{HS} \cdot T_{HS}, \quad (13)$$

where $(T_{TX}, T_{RX}, T_{SP}, T_{CS}, T_{HS})$ are the time durations the station spends in different energy modes. In all the simulations, we have set a constant 5-minute time duration to evaluate the energy usage, that is, $T_{TX} + T_{RX} + T_{SP} + T_{CS} + T_{HS} = 5mins$. Please note that the real value W_{HS} may be more than half of W_{CS} according to the analysis in [6]. However, according to this equation, a higher W_{HS} will make SASD have even more energy saving comparing with E-MiLi.

Table 6 lists the parameter configurations used in our simulations. We evaluate the performance of the five protocols under the 802.11ac standard. The 802.11ac recommends A-MPDU for the data packet transmission, and its length L_P is $2^{13+Exp} - 1$ bytes, where $Exp = 0, \ldots, 7$ [20]. That means, the packet length L_P is from $8K$ to $1M$ bytes. Meanwhile, 802.11ac has a series of data rates, as listed in the Appendix, here we just select one as an example in our simulations.

Parameter	Value	Parameter	Value
Preamble	$32\mu s$	SIFS	$16\mu s$
SASD header[2]	$33.35\mu s$	DIFS	$34\mu s$
Catalogue cost	$6.3\mu s$	CWmax	$1023\mu s$
Data rate	1.56Gbps	CWmin	$15\mu s$
Packet size	$8K \sim 1M$bytes	Time slot	$9\mu s$

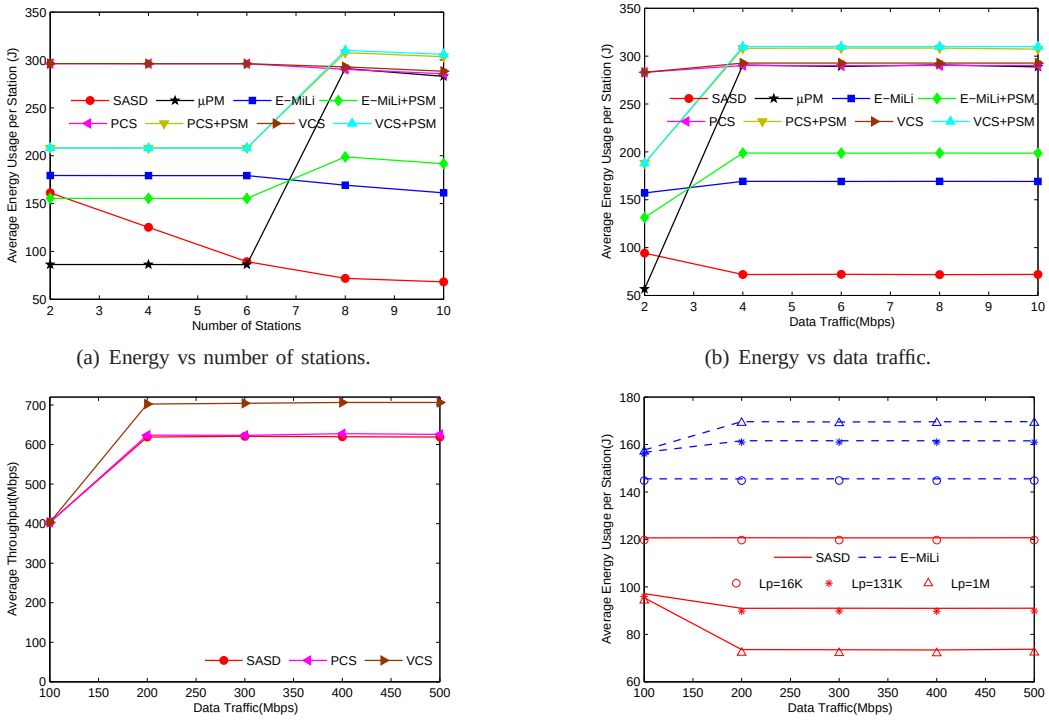
TABLE 6
Parameter configurations for ns-2 simulation.

5.1 Single AP with Multiple Stations Scenario

In the single AP with multiple stations scenario, we have set up a WLAN that consists of one AP with multiple stations. The stations are randomly deployed within the AP's transmission range. First, we study the energy saving performance by increasing the number of stations. We use the fixed data rate

2. The SD field can be different among different devices, which causes different SASD headers. Here, we choose the maximum SD field length which is used in our hardware experiment.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing



(a) Energy vs number of stations.

(b) Energy vs data traffic.

(c) Throughput vs data traffic.

(d) Energy vs packet size.

Fig. 13. Energy saving performance comparison in the single AP with multiple stations scenario.

(200$Mbps$, 1M-byte packet size) between the AP and each station. Fig. 13(a) shows that, the energy performance of SASD increases along with the increase of the number of stations, that is because, when the network density increases and all stations want to communicate with the AP, each station has more chances to switch to the sleeping mode to save energy. The energy cost of SASD is lower than the other protocols except μPM, which makes nodes have more chances to go to sleep in the lightly loaded scenarios when the number of stations is below 8. However, when the number of stations increases to 8 , the energy cost of μPM increases dramatically as stations always have packets waiting to be sent out, thus cannot go to sleep. This phenomenon is also observed in PSM.

To evaluate the energy performance under the full workload, we have fixed the number of stations to 8 and tested the energy cost performance under different data traffic loads. We have fixed the total data traffic loads to different levels and randomly generated the data traffic load on each station. Fig. 13(b) shows that, with various data traffic loads, the SASD can achieve a better energy saving performance than the PCS (75.2%), VCS (75.4%), PCS+PSM (76.7%), VCS+PSM (76.8%), E-MiLi (57.5%), E-MiLi+PSM (63.7%) and μPM (75.1%) when the data traffic load reaches 200$Mbps$. The energy usage of each station in the SASD is decreasing when the data traffic loads increase from 100$Mbps$ to 200$Mbps$ due to the increase of T_{SP} in Eq. (13). However, the energy cost in PSM (PCS+PSM or VCS+PSM) and μPM is increasing in this situation, due to the increased data packets in the stations' waiting lists.

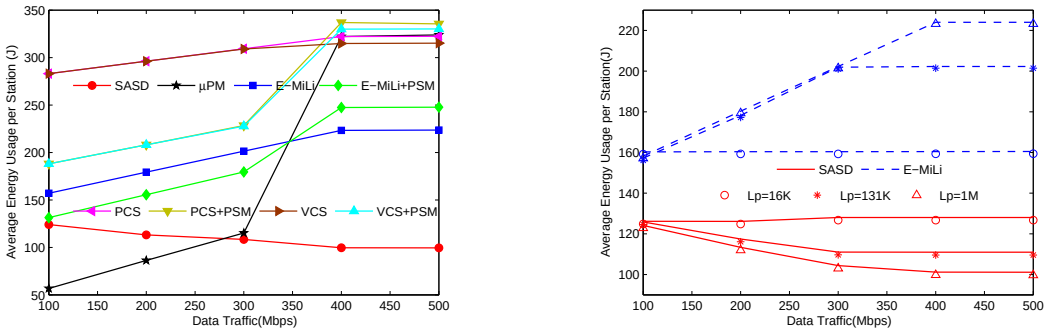
Fig. 13(c) shows that, for the network throughput, the SASD has about 1% throughput degradation comparing with PCS, due to the overhead induced by the SASD header. Both SASD and PCS have about 11% throughput degradation comparing with VCS, as VCS can combat the hidden terminal problem existed in this scenario through the exchange of RTS and CTS. Please note that PCS and VCS have similar energy consumption according to Fig. 13(a) and Fig. 13(b), as packet retransmissions in PCS due to hidden terminals will also consume stations' energy; moreover, stations uninvolved in packet transmission still consume energy on carrier sensing or packet overhearing.

Finally, we have varied the packet size to investigate its impact on the SASD's performance. Fig. 13(d) shows that, the energy cost of SASD decreases while that of E-MiLi increases with the increase of the packet size. That is because, when the packet size increases, the transmission overhead (such as backoff, the control frame transmissions and so on) decreases, leading to a larger T_{TX} in Eq. (13). For E-MiLi, a larger T_{TX} leads to a smaller T_{HS}, thus leads to a higher energy cost when the packet size increases. For SASD, a larger T_{TX} will lead to a larger T_{SP}, that is why its energy cost decreases when the packet size increases.

5.2 Multiple APs with Multiple Stations Scenario

To evaluate the SASD's scalability in a more general scenario, we have generated a multiple APs with multiple stations scenario that consists of 3 APs and 18 stations. Specifically, we have setup 3 WLANs, each of which has 1 AP and 6 stations and the stations are randomly deployed around the AP. We make each AP maintain a different set of bit sequences, and assign each client a unique bit sequence independently. In the simulation, we first vary the data traffic loads from 100$Mbps$ to 500$Mbps$ with 1M-byte packet size, the results in Fig. 14(a) show that, under high traffic loads (≥400$Mbps$), the SASD can outperform the PCS (68.3%), VCS (67.7%), PCS+PSM (70.1%), VCS+PSM (69.3%), E-MiLi (53.2%), E-MiLi+PSM (58.8%) and μPM (68.1%) on the energy efficiency. We have also varied the packet size to investigate its impact on the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing



(a) Energy vs data traffic.

(b) Energy vs packet size.

Fig. 14. Energy saving performance comparison in the multiple APs with multiple stations scenario.

SASD's performance in this scenario, the results in Fig. 14(b) show that the energy cost of SASD also decreases while that of E-MiLi increases with the increase of the packet size.

The above simulation experiments reveal that the SASD scheme can detect and identify the SASD header in the down-clocking mode and switch the device to the sleeping mode to save the energy cost on the packet overhearing. Therefore, the SASD can solve the energy inefficiency of the packet overhearing and achieve better energy efficiency performances in different WLAN environments.

6 Related Work

The energy efficiency of WiFi devices has been a popular research topic in the research community of wireless networks. Most solutions to the energy inefficiency problem work at the MAC layer. A well-known power saving mode (PSM) protocol [7] in 802.11 standard requests the AP to buffer all the packets for the stations which work at the power saving mode and the stations to periodically wake up to receive a Traffic Indication Map (TIM) notice in the AP's beacon messages. If the station's corresponding TIM field is set to 1, it sends back a PS-POLL packet to the AP and prepares to receive the buffered packet from the AP. The PSM would cause a long response delay that particularly affects the QoS of the time sensitive applications [25]. Consequently, most WiFi devices would implement the adaptive PSM [8] that devices can switch between the constantly awake mode (CAM) and PSM based on certain mechanisms, so that the QoS would not be degraded in the adaptive PSM. However, it cannot prevent the stations from receiving unnecessary packets when it switches to the CAM.

NAPman [9] improves the energy efficiency of the above PSM protocols by isolating the traffic of PSM clients, so that stations can reduce the time staying in the high power consumption CAM. Based on NAPman, Manweiler and Choudhury [10] proposed a TDMA manner mechanism that isolates the traffic from different WLANs, the stations would wake up in each scheduled time and receive the packets. μPM [17] proposed to save the energy by aggressively switching the stations to the sleeping mode in a very short interval. It uses a prediction mechanism to exploit short idle intervals and does not need any special support from the AP. However, it is only effective in very lightly loaded networks.

Recent studies exploited a new form of energy efficiency at the PHY layer. As the main energy consumption of a wireless card is caused by the modern wireless digital circuits and is proportional to $V_d^2 f$, where V_d is the supply voltage and f is the sampling rate [2], [12], [26], by reducing the voltage of radio circuits or decreasing the sampling clock-rate, the power efficiency of whole wireless system can be achieved [11].

The wake-on-wireless [13] employs a low power radio circuit to detect the packet, and switches to the full power circuit to receive the incoming packets. The system would keep in a low energy consumption in channel sensing. But the devices need a secondary low power circuit to implement the packet detection, and also, it cannot solve the packet overhearing problem.

Kim et al. [27] proposed a similar solution as [13], instead of using the secondary low power circuit to detect packets, it deploys the low power sensor, which is called accelerometer, to sense the channel. As a result, the overall energy cost at the PHY layer can be reduced. However, as the sensor is using the energy-based detection to sense the channel, the false negative/positive error cannot be controlled. Thus, it would cause the unfair transmission in busy networks and the QoS cannot be guaranteed.

R. Chandra et al. [2] conducted the hardware experiment on the energy cost of Atheros NIC cards. It revealed that the energy consumption of the wireless card can be reduced when the A/D clocking-rate (channel bandwidth) goes down. However, this paper is more focusing on the adaptive channel bandwidth usage, and does not discuss the packets' detection method under the downclocking rate.

Most recently, Zhang and Shin [6] proposed a state-of-the-art PHY layer energy saving mechanism–E-MiLi. Similar as [2], it linearly decreases the A/D converter's sampling clocking-rate, and adds a M-preamble field to notify the station whether the incoming packet is addressed to it. The stations which are not involved in the packets receiving can still in the downclocking mode to save the energy on the idle listening. E-MiLi needs no extra circuits compared with wake-on-wireless [13] or sensors [27], and because it applies at the PHY layer, it can combine with other MAC layer PSM protocols. By deploying the A/D converter's downclocking rate 2, E-MiLi can save 36% energy cost of wireless NIC card. After that, Zhang and Shin [28] proposed the Gap Sensing which uses the same method of E-MiLi to detect packets and save the energy between heterogeneous wireless devices.

However, all these MAC layer or PHY layer power saving protocols cannot avoid the energy consumption on packet overhearing in the high dense and traffic WLANs [3], [6]. To avoid this significant energy inefficiency of packet overhearing, Biswas and Datta [4] proposed a RTS/CTS based mechanism that forces the stations switch to the sleeping mode if the packet is not addressed to it. The approach relies on the receiving of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMC.2016.2604805, IEEE Transactions on Mobile Computing

the RTS/CTS packets at the MAC layer. It cannot take the advantage of the PHY layer energy saving. Also, as the high overhead of the RTS/CTS mechanism, it would also degrade the QoS at the clients side.

Xiong et al. [29] proposed a new form of RTS/CTS messages. It adopts the symbols-level detection to carry the packet transmission time, which allows the packets transmission time to be directly delivered at the PHY layer without decoding the whole packets at the MAC layer. However, this kind of symbols-level information delivery and detection method should work in the fullclocking mode.

Comparing to the previous works, the proposed SASD does not need expensive hardware such as extra sensors to leverage the advantages of both the sleeping mode and downclocking mode for energy savings. Although it introduces the overheads of the SASD header and catalogue cost, it is a practical solution to the packet overhearing problem, which can improve the energy efficiency of the wireless devices.

7 Conclusions

In this paper, we have proposed a novel SASD scheme to enable the wireless devices to discern the required information under the energy-saving downclocking mode. As the SASD scheme effectively eliminates the energy waste on the packet overhearing, it can significantly improve the energy efficiency of wireless devices. Moreover, we show the feasibility and performance improvement of the SASD through both hardware implementation and software simulation. All these efforts demonstrate the SASD to be a practical solution in real world network scenarios.

References

[1] K. Y. Jang, S. Hao, A. Sheth, and R. Govindan, "Snooze: Energy management in 802.11n wlans," in *ACM SIGCOMM*, 2011.

[2] R. Chandra, R. Mahajan, T. Moscibroda, R. Raghavendra, and P. Bahl, "A case for adapting channel width in wireless networks,," in *ACM SIGCOMM*, 2008.

[3] F. Ashraf, "Survival guide for dense networks," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2013.

[4] S. Biswas and S. Datta, "Reducing overhearing energy in 802.11 networks by low-power interface idling," in *IEEE International Conference on Performance, Computing, and Communications*, 2004.

[5] K.-H. Kim, A. W. Min, D. Gupta, P. Mohapatra, and J. P. Singh, "Improving energy efficiency of wi-fi sensing on smartphones," in *IEEE INFOCOM*, 2011.

[6] X. Zhang and K. G. Shin, "E-MiLi: Energy-minimizing idle listening in wireless networks," in *ACM MobiCom*, 2011.

[7] IEEE Computer Society. 802.11, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," 2007.

[8] R. Krashinsky and H. Balakrishnan, "Minimizing energy for wireless web access with bounded slowdown," *Wireless Networks*, vol. 11, no. 1-2, pp. 135–148, 2005.

[9] E. Rozner, V. Navda, R. Ramjee, and S. Rayanchu, "Napman: Network-assisted power management for wifi devices," in *ACM MobiSys*, 2010.

[10] J. Manweiler and R. Roy Choudhury, "Avoiding the rush hours: Wifi energy management via traffic isolation," in *ACM MobiSys*, 2011.

[11] F. Lu, G. M. Voelker, and A. C. Snoeren, "Slomo: Downclocking wifi communication." in *NSDI,USENIX*, 2013.

[12] K. Flautner, S. Reinhardt, and T. Mudge, "Automatic performance setting for dynamic voltage scaling," in *ACM MobiCom*, 2001.

[13] E. Shih, P. Bahl, and M. J. Sinclair, "Wake on wireless: An event driven energy saving strategy for battery operated devices," in *ACM MobiCom*, 2002.

[14] S. Yang, J. Kurose, and B. N. Levine, "Disambiguation of residential wired and wireless access in a forensic setting," in *IEEE INFOCOM Mini-Conference*, 2013.

[15] C. C. Enz, A. El-Hoiydi, J.-D. Decotignie, and V. Peiris, "WiseNET: An ultralow-power wireless sensor network solution," *Computer*, vol. 37, no. 8, pp. 62–70, 2004.

[16] O. Lee, J. Kim, and S. Choi, "Wizizz: Energy efficient bandwidth management in ieee 802.11ac wireless networks," in *IEEE SECON*, 2015.

[17] J. Liu and L. Zhong, "Micro power management of active 802.11 interfaces," in *ACM MobiSys*, 2008.

[18] S. Sen, R. R. Choudhury, and N. Santhapuri, "CSMA/CN: Carrier sense multiple access with collision notification," in *ACM MobiCom*, 2010.

[19] T. Bansal, B. Chen, P. Sinha, and K. Srinivasan, "Symphony: Cooperative packet recovery over the wired backbone in enterprise WLANs," in *Proc. of ACM MOBICOM*, 2013.

[20] M. S. Gast, "802.11ac: A survival guide," *O'Reilly Media*, 2013.

[21] Ettus Inc, "Universal Software Radio Peripheral," http://ettus.com.

[22] GNU Radio, "GNU Radio - The Open Source Software Radio Project," http://gnuradio.squarespace.com/.

[23] K. A. Jamieson, "The SoftPHY Abstraction: From packets to symbols in wireless network design," Ph.D. dissertation, MIT, 2008.

[24] S. Gollakota and D. Katabi, "Zigzag decoding: Combating hidden terminals in wireless networks," in *ACM SIGCOMM*, 2008.

[25] Y. Agarwal, R. Chandra, A. Wolman, P. Bahl, K. Chin, and R. Gupta, "Wireless wakeups revisited: Energy management for voip over wi-fi smartphones," in *ACM MobiSys*, 2007.

[26] W. R. Dieter, S. Datta, and W. K. Kai, "Power reduction by varying sampling rate," in *ACM/IEEE ISLPED*, 2005.

[27] K.-H. Kim, A. W. Min, D. Gupta, P. Mohapatra, and J. P. Singh, "Improving energy efficiency of wi-fi sensing on smartphones," in *IEEE INFOCOM*, 2011.

[28] X. Zhang and K. G. Shin, "Gap sense: Lightweight coordination of heterogeneous wireless devices," in *IEEE INFOCOM*, 2013.

[29] T. Xiong, J. Zhang, J. Yao, and W. Lou, "Symbol-level detection: A new approach to silencing hidden terminals," in *IEEE ICNP*, 2012.

Tao Xiong received the Ph.D. degree in the Department of Computing, The Hong Kong Polytechnic University, Hong Kong in 2015. He also received the B.E. degree in Computing Science from China University of Geoscience, China in 2003 and the M.E. degree in Computer Science and Engineering from University of New South Wales, Australia in 2008. His research interest focuses on cross-layer protocol design and implementation for wireless networks.

Junmei Yao received the bachelor degree in Communication Engineering from Harbin Institute of Technology, China in 2003, the Master degree in Communication and Information System from Harbin Institute of Technology, China in 2005. After working as an engineer at Huawei for five years, she is currently working toward the PhD degree in the Department of Computing at The Hong Kong Polytechnic University, Hong Kong. Her research interests include wireless networks, wireless communications and RFID systems.

Jin Zhang received his Ph.D. degree in the Department of Computing, The Hong Kong Polytechnic University, Hong Kong in 2016. He also received his B.E. and M.E. degrees in Applied Mathematics from Harbin Institute of Technology, China in 2006 and 2008, respectively. His current research interests are in the areas of peer-to-peer streaming, mobile cloud computing, and game theory.

Dr. Wei Lou is currently an associate professor in the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China. He received the B.E. degree in Electrical Engineering from Tsinghua University, China in 1995, the M.E. degree in Telecommunications from Beijing University of Posts and Telecommunications, China in 1998, and the Ph.D. degree in Computer Engineering from Florida Atlantic University, USA in 2004. Dr. Lou's current research interests are in the areas of wireless networking, mobile ad hoc and sensor networks, peer-to-peer networks, and mobile cloud computing. He has worked intensively on designing, analyzing and evaluating practical algorithms with the theoretical basis, as well as building prototype systems. His research work is supported by several Hong Kong GRF grants and Hong Kong Polytechnic University ICRG grants.