Enable Flexible Spectrum Access with Spectrum Virtualization

Kun Tan, Haichen Shen, Jiansong Zhang, Yongguang Zhang
Microsoft Research Asia

***Abstract*—Enabling flexible spectrum access (FSA) in existing wireless networks is challenging due to the limited *spectrum programmability – the ability to change spectrum properties of a signal to match an arbitrary frequency allocation.* This paper argues that *spectrum programmability* can be separated from general wireless physical layer (PHY) modulation. Therefore, we can support flexible spectrum programmability by inserting a new *spectrum virtualization layer (SVL)* directly below traditional wireless PHY, and enable FSA for wireless networks without changing their PHY designs.**

SVL provides a *virtual baseband* abstraction to wireless PHY, which is static, contiguous, with a desirable width defined by the PHY. At the sender side, SVL reshapes the modulated baseband signals into waveform that matches the dynamically allocated physical frequency bands – which can be of different width, or non-contiguous – while keeping the modulated information unchanged. At the receiver side, SVL performs the inverse reshaping operation that collects the waveform from each physical band, and reconstructs the original modulated signals for PHY. All these reshaping operations are performed at the signal level and therefore SVL is *agnostic* and *transparent* to upper PHY. We have implemented a prototype of SVL on a software radio platform, and tested it with various wireless PHYs. Our experiments show SVL is flexible and effective to support FSA in existing wireless networks.

I. Introduction

Traditional wireless network works on a fixed set of predefined channels. This, however, becomes inefficient as the frequency bands are shared by more and more heterogeneous wireless networks. For example, in 2.4GHz ISM band, transmissions on a narrow band wireless network (*e.g.*, ZigBee) will interfere with a coexising wide-band wireless network (*e.g.*, 802.11n) and cause it to backoff, wasting a large portion of wireless frequency. To improve the spectrum efficiency, the future wide-band devices should support *flexible spectrum access* (FSA) [2]. Instead of contending with the narrow-band wireless with a large contiguous channel, FSA allows the wide-band device to utilize the available spectrum segments leftover by narrow-band networks and avoid mutual interference.

While FSA is desirable, enabling it in existing wireless networks remains a challenging task due to the lack of *spectrum programmability – the ability to change spectrum properties of a signal to match an arbitrary frequency allocation*. First, conventional wireless standards are primarily designed for static and monolithic spectra. Therefore, they have very limited spectrum- and bandwidth- agility. For example, WhiteFi [3] can down-convert and vary the channel width of 802.11 signals, based on the half-/quarter-clocked modes, to fit a TV whitespace. But it is limited to operating on only contiguous frequency bands. Second, future wireless PHY may support more flexible spectrum programmability by adopting non-contiguous OFDM (NC-OFDM) schemes [12], [14]. But supporting NC-OFDM operation significantly complicates the PHY design. Different combinations of non-contiguous frequency segments may result in different preamble types and pilot placements, each of which may require special treatment. Therefore, the implementation effort of such a NC-OFDM PHY grows proportionally with the number of NC patterns supported. Indeed, the latest wireless standards support only limited NC-OFDM configurations. For example, 802.11ac specifies only one non-contiguous 80+80 MHz channel bonding [1].

In this paper, we argue that spectrum programmability can be separated from the general PHY modulation design. Therefore, we can support flexible spectrum programmability by adding a new layer below the existing PHY that dynamically shapes the modulated PHY signals to match the real available frequency segments, before sending them to the radio-frequency (RF) front-end. This spectrum reprogramming layer abstracts away the underly spectrum dynamics and provide the PHY a fixed contiguous *virtual spectrum band* with a *desired width* defined by the PHY itself. Thus, the conventional PHY designs, including preambles, modulation, and pilot placements, etc., can be seamlessly supported. We call this approach as *spectrum virtualization* and the intermediate layer we proposed as *spectrum virtualization layer (SVL)*. Figure 1 shows the location of SVL in the network architecture view. Since SVL sits between the physical layer (or baseband processing) and the RF front-ends, we also term it *Layer 0.5*.

The key function of SVL is signal reshaping and spectrum enforcement. SVL maintains a mapping between the virtual spectrum band and the set of available physical frequency segments. It receives modulated signals from PHY and performs reshaping operations – a set of digital signal processing (DSP) algorithms that filter, shift, decomposition and recomposition signals (detailed in Section IV) – to transform them into waveform suitable to transmit in the real frequency segments, while keeping the modulated information unchanged. At the receiver side, waveform from each frequency segment is combined at SVL and the original modulated signals are reconstructed and fed to wireless PHY. Further, SVL also maintains the transmission power and mask regulations of each physical frequency segment, and ensures the waveform transmitted in each frequency segment compliant to these requirements.

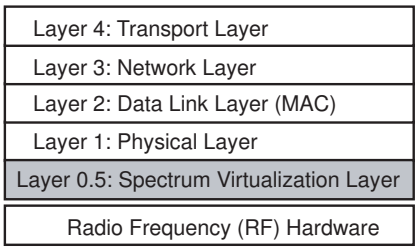

Fig. 1. Layer 0.5: Spectrum Virtualization Layer.

SVL decouples wireless PHY from the RF front-ends. Therefore, it can map multiple virtual spectrum bands to the same RF front-ends and enables multiple PHYs to seamlessly share the same RF hardware. We call this property as *radio virtualization*. Radio virtualization provides convenient means for multi-radio integration on mobilde devices.

We have implemented a prototype of SVL based on the Sora software radio platform [13] and successfully add FSA function to multiple wireless PHYs without changing their designs. Specifically, we test SVL with three different PHYs, namely ZigBee, 802.11b, and 802.11a, presenting two typical wireless technologies used today: ZigBee and 802.11b are based on single carrier modulation with spectrum spreading; while 802.11a uses multi-carrier modulation of OFDM. Our results verify that SVL is indeed a generic design: Each of our tested wireless PHY can flexibly bond arbitrary frequency segments and get similar throughput as long as the sum of the channel width of all usable frequency bands (excluding guard-bands) remains the same. This implies our reshaping operations inside SVL causes no additional distortion (or signal-to-noise ratio, SNR, loss) to the original signal.

In summary, the paper makes following contributions:

1) We propose to separate the spectrum programmability from the general wireless PHY design and promote a new Spectrum Virtualization Layer to support flexible spectrum access. SVL is located at Layer 0.5 and enables FSA function to conventional PHY without changing the PHY design itself.
2) We design the signal reshaping process for SVL that can flexibly map virtual spectrum band to (non-contiguous) physic spectrum band(s) without causing additional distortion. Our signal reshaper also does not need the specific knowledge of the modulation scheme used by the upper layer PHY.
3) We present the novel way to multiplex multiple PHY on single RF front-end.
4) We demonstrate the feasibility of SVL with our implementation on a software radio platform, and evaluate its performance.

The rest of the paper is organized as follows. Section II provides some background on radio transceiver designs and our motivation. We present the SVL architecture in Section III. The detailed designs are discussed in Section IV. After describing the implementation of a SVL prototype using a software radio platform in Section V, we describe describes some applications based on SVL in Section VI. Section VII presents the performance evaluation. Finally, Section VIII discusses related work and Section IX concludes.

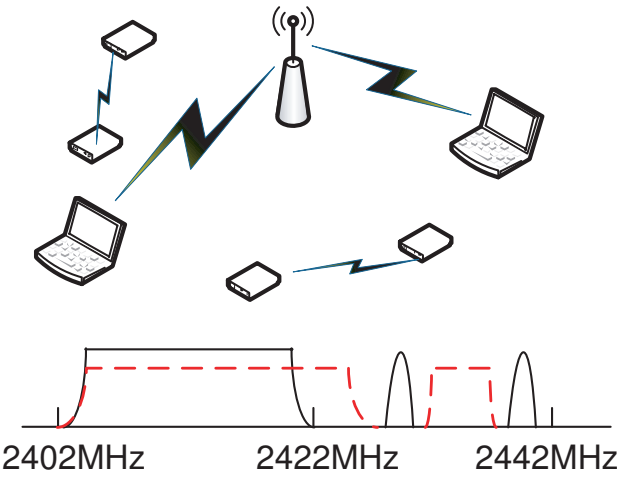

Fig. 2. Dynamic spectrum access. Communication frequency bands may change dynamically according to the spectrum availability as well as application requirements.

II. Background and Motivation

A. Flexible Spectrum Access

Current wireless network usually involves a pre-assigned channel for communication. Such a channel covers a continuous frequency band with a pre-defined central frequency and bandwidth. Such static spectrum management may cause frequency spectrum fragmentation and lower efficiency in spectrum utilization. For example, Figure 2 illustrates a common scenario for a 802.11 WLAN network coexists with two ZigBee networks in a same spectrum band. The two ZigBee networks are configured at channel 16 (central freq. 2430MHz) and 18 (central freq. 2440MHz), which overlap with the 802.11 channel 5 (central freq. 2432MHz). As shown in Figure 2, these two ZigBee networks have separated the spectrum band from 2.4GHz to 2.442GHz into two segments. Although modern 802.11n can exploit the entire 40MHz frequency band, it may result low throughput due to the narrow-band interference from two ZigBee networks, as any ongoing ZigBee transmission will block the entire 40MHz 11n channel. In current wireless network, the best you can do is to configure the 11n network to only the lower half 20MHz band, as shown by the black solid line in Figure 2, which is still inefficient. In contrast, with FSA, the radio can make full use of 30MHz available spectrum for communication, as indicated by the read dash line in Figure 2. Therefore, the overall spectrum utilization is greatly improved.

FSA removes the concept of pre-defined channel. Frequency bands that a radio is operating on are dynamically allocated and managed, depending on the spectrum availability and the application requirements. To efficiently utilize these varied RF bands, it requires wireless PHY to generate matched baseband waveform as well. Although new wireless PHY based on NC-OFDM [12], [14] can support such frequency agility, NC-OFDM usually requires much complicated PHY design. With NC-OFDM PHY, the number of available subcarriers

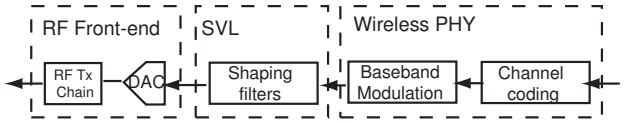

Fig. 3. The architecture of a wireless transmitter with SVL.

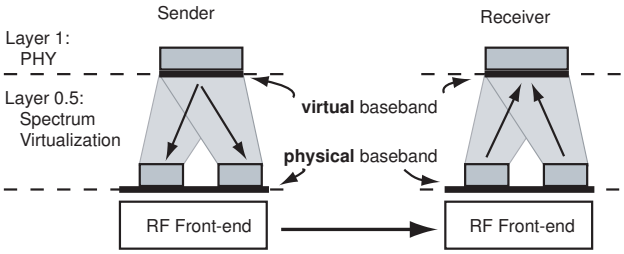

Fig. 4. Spectrum virtualization.

as well as their positions change with the available frequency bands. As a consequence, the entire PHY modulation structure needs to change as well. For example, for each frequency availability pattern, the NC-OFDM may need a different matching preamble. Also, pilot subcarrier placement may need to be adjusted based on the subcarrier availability. Effectively, each combination of available frequency bands may create a *configuration* for NC-OFDM PHY, and it is a non-trivial effort to support a large set of such configurations.

B. Spectrum Virtualization

In this paper, we propose a new way to support FSA in wireless networks by adding a *Spectrum Virtualization Layer* (SVL) below the traditional PHY (baseband processing). Figure 3 illustrates a transmitter architecture with SVL. Since SVL is located between the wireless PHY and the analog RF front-end, we term it as *Layer 0.5*.

The goal of SVL is to bridge the gap between the traditional wireless PHY, which is designed to use a static frequency band with pre-defined bandwidth, and the desirable dynamic baseband under FSA, which can have any time and space varying spectrum configuration, often over a wider band. SVL decouples the tight connection between wireless PHY and RF front-end and presents yet another indirection. It provides a *virtual baseband* to wireless PHY that is contiguous with a desired bandwidth. Then, in runtime, SVL dynamically reshapes the *virtual baseband signals* to waveforms that matches the real available frequency bands, and transmit them through the RF front-end. At the receiver side, the inverse reshaping operation is performed inside SVL and waveform from each real frequency band is collected and combined to reconstruct the original baseband signals. Figure 4 illustrates these operations. In Figure 4, a wide band virtual spectrum signal is reshaped into two narrower waveforms that are transmitted in two frequency bands. The receiver, after collecting these two waveforms, reconstructs the virtual spectrum signal for the wireless PHY.

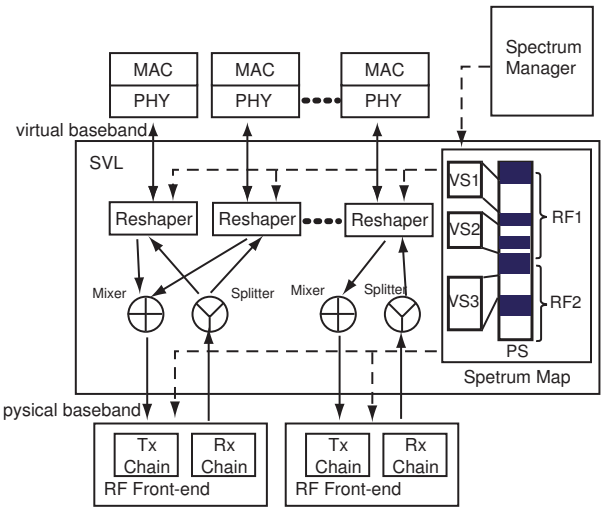

Fig. 5. Architecture of Spectrum Virtualization Layer. SVL maintains a mapping between virtual baseband and physical spectrum bands, and dynamically reshapes signals between virtual and physical spectrum. Gray portion of the spectrum map shows the unavailable spectrum bands. The solid arrows show the data paths; while the dashed arrows show the control paths.

SVL is a nature extension to the signal shaping function in most wireless PHY designs [1]. While traditional shaping function is only used for limiting the effective bandwidth of a transmission, SVL adds more general *spectrum programmability* in radio and provides an architecture to systematically support FSA for any wireless PHY.

The core challenge of SVL is how to design a flexible and effective signal shaper. In the next section, we overview the SVL architecture and principles. Section IV discusses the detailed designs.

III. Overview and Architecture

The overview architecture of SVL is shown in Figure 5. SVL provides a *virtual baseband* to each PHY. The width of the virtual baseband is specified by each PHY during the initialization stage. Hereafter in this paper, when we call *baseband*, we refer to this the virtual baseband; We also use the term "*physical spectrum band*" (or simply "*physical band*") to denote a portion of spectrum on a RF front-end's *physical baseband* that is allocated to the PHY.

SVL maintains a *spectrum map* between virtual baseband and physical spectrum bands. The mapping is quite flexible in SVL. For example, it can map the baseband to a physical spectrum band with the same width (*e.g.*, *VS1* in the figure). Alternatively, it can map the baseband to a narrower contiguous physical band, or several non-contiguous physical bands (*e.g.*, *VS2* or *VS2* in the figure). How spectrum is allocated is controlled by the *Spectrum Manager*.

The function of the spectrum manager is to monitor the current spectrum usage (*e.g.*, by sensing or querying a database), allocate free physical spectrum bands for a PHY based on various policies, and update the spectrum map in SVL.

[1] Also known as pulse shaping.

There is already abundant work on the dynamically spectrum management, like [3], [5], [8], [14]. In this paper, we do not explicitly address how spectrum should be allocated among different PHYs - which is still an active research area. Instead, we assume SVL already has such a spectrum map, and our focus is how to enforce the *spectrum access* of baseband signals to match the physical spectrum specification.

The core and challenging part of SVL is to design the *signal reshaper* that translates signals from baseband to physical bands, and vice versa. The reshaper must satisfy the following requirements.

First, the reshaper should be *PHY agnostic*. To support heterogenous wireless PHY, the reshaper must perform signal translation without knowing the specific modulation scheme. It implies that the reshaper can only adopt general digital signal processing algorithms that operate on general baseband waveform.

Second, the reshaper should be *transparent* to upper layer PHY. That means, although the reshaping operation may change the baseband waveform in some way, the upper layer PHY should not be able to tell whether this distortion is coming from the reshaping operation or it is due to a natural wireless channel fading. This is to ensure the distortion caused by reshaping can be modeled by an equivalent multi-path fading channel, and therefore can be handled with the equalization mechanisms in current PHY.

Finally, the reshaper should be as *simple* as possible, introducing minimal overhead. For example, it should not try to equalize and recover exactly the original transmitted baseband waveforms, because of following two reasons: (1) It may add considerable overhead by sending training symbols; (2) It largely overlaps with similar functions in PHY. Therefore, it can be either redundant or premature, since only the PHY knows the best way to handle fading, based on its modulation scheme as well as application requirements.

We will elaborate our reshaper design in Section IV-A that satisfies all above three requirements. Our reshaper reveals all distortion of baseband waveform to upper layer PHY and relies on their own ability to correct them.

After reshaping, baseband signals are converted to physical baseband signals. Physical baseband signals from multiple PHY may be mixed (added) together before sending to the RF front-end. When receiving, the incoming signals are passed to a *splitter*, which contains a matched filter for each PHY based on its spectrum map. The filtered physical band signals are fed to the reshaper again, where the inverse operations are performed to recover the baseband signals. These baseband signals are then sent to wireless PHY.

Conceptually, SVL also *virtualizes* RF front-ends for each wireless PHY, with the *mixing* and *splitting* operations. As illustrated in Figure 6, SVL can flexibly map different PHY to different RF front-ends; Also, it multiplexes several PHYs onto single RF front-end hardware. *RF front-end virtualization* is particularly helpful in FSA networks, as it allows multiple PHY to share a common powerful wide-band agile RF front-end. Thereby, it reduces the required resource for multi-radio integration in terms of space, energy as well as price, on mobile devices.

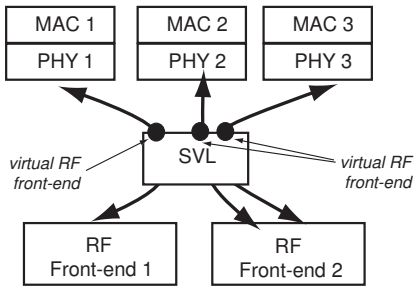

Fig. 6. RF front-end virtualization.

IV. SVL Operations

A. Signal Reshaping

The reshaper is the core functional module in SVL that performs digital signal processing to translate signals between virtual and physical basebands. Figure 7 shows the processing blocks inside the reshaper.

There are three main operations: *signal decomposition/recomposition*, *bandwidth and sampling rate adjustment* and *frequency band shifting*. Signal decomposition may split a virtual baseband signal into several sub-streams, each of which can be mapped to a separated physical band. At the receiver side, these sub-streams are collected by *recomposition* and converted back to virtual baseband signals. The signal decomposition and recomposition are two core operations to support non-contiguous bonding of physical bands. The *bandwidth and sampling rate adjustment* may change the signal bandwidth and match the signal sampling rate to the RF front-end. This operation allows a virtual baseband to be mapped to a physical band with different width. Finally, the *frequency band shifting* module will move the central frequency of baseband signal ($0Hz$) to that of the physical band when transmitting, and move it back to zero when receiving.

In following discussion, we refer a spectrum band, B, as a set of frequency, which can be represented by its central frequency f and pass-band width b. Therefore, we denote $B(f, b)$ as a frequency band, whose range is $(f-\frac{b}{2}, f+\frac{b}{2})$. We also denote $\Theta = \{B_{p,i}(f_i, b_i)|i = 1..k\}$, the set of allocated physical bands to the PHY.

We often use b_s to denote the *aggregated bandwidth* of all allocated physical bands in Θ, *i.e.*,

$$b_s = \sum b_i.$$

And we denote $B_{span}(f_{span}, b_{span})$ *the span* of Θ, which is defined as the spectrum band between the highest and lowest frequency of all physical bands. Formally, $B_{span} = (f_{low}, f_{high})$, where

$$\begin{aligned} f_{high} &= \max f, f \in B_{p,i}, B_{p,i} \in \Theta; \\ f_{low} &= \min f, f \in B_{p,i}, B_{p,i} \in \Theta. \end{aligned}$$

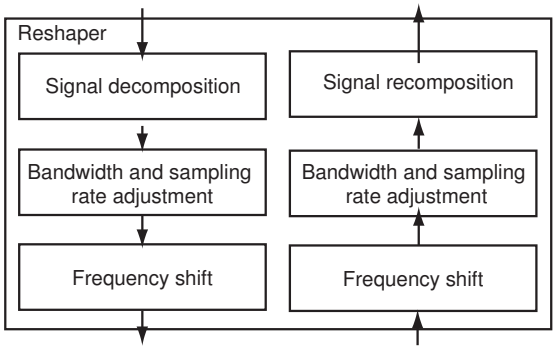

Fig. 7. Processing blocks of the reshaper in SVL.

1) Signal Decomposition and Recomposition: There could be several ways to split a baseband signal into multiple sub-streams. For example, one naive way is the *time-domain decomposition*, which may cut signal samples into segments, *i.e.* PHY *symbols*, and map each symbol to a physical band alternatively, as shown in Figure 8(a). However, such time-domain decomposition violates the *transparency* principle as discussed in Section III. This is because symbols passing different spectrum bands may experience completely different multi-path fading, and be distorted in different ways. This will defeat the channel estimation and equalization of existing PHY, as they always assume that all samples should pass a same *coherent* wireless medium. The erroneous channel estimation and equalization will severely reduce the demodulation performance and cause frame losses.

Instead, in this paper, we propose a frequency-domain signal decomposition based on FFT/iFFT, as shown in Figure 8(b). The core idea is to perform an M-point FFT on every PHY symbol passed to SVL. This FFT operation decomposes the time-domain signals into M frequency-domain components. We can re-assign each of the M frequency component to a sub-carrier corresponding the allocated physical bands. After re-assignment, it performs an N-point iFFT to convert the frequency components back into time-domain samples. Here, the aggregated bandwidth of physical bands should be no less than virtual baseband width. Thus, N is no less than M. At the receiver side, the inverse operation is performed. This time, it will first perform an N-point FFT to convert the incoming samples into frequency components. Next, M sub-carriers corresponding to the allocated physical bands are collected. Then, an M-point iFFT is performed to regenerate the baseband symbols, as shown in Figure 8(c).

The frequency-domain decomposition/recomposition retrains the transparency property of SVL. Intuitively, it ensures the components at the same frequency of different PHY symbols always pass the same sub-carrier in a physical spectrum band. Thus, it would appear to upper layer PHY as a coherent wireless channel, although the multi-path fading property may be changed by the reshaping operation. Consequently, existing channel estimation and equalization mechanisms can be used to handle this multi-path fading effect. Later in Section IV-A4, we will give a more formal explanation.

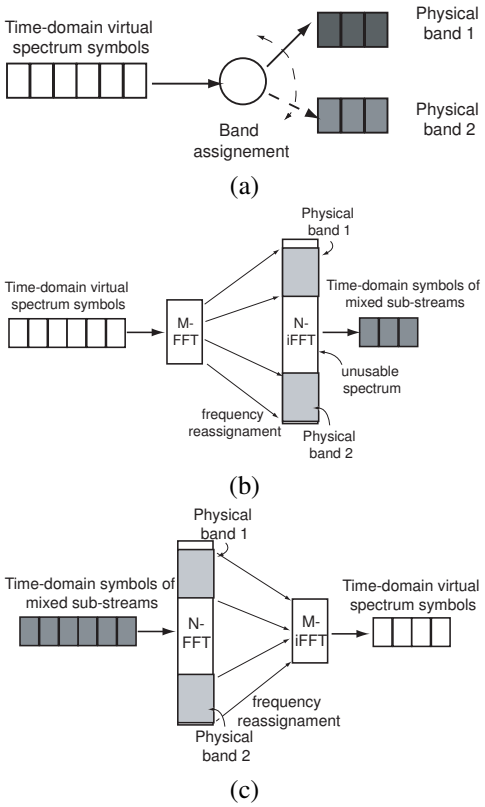

Fig. 8. Signal decomposition. (a) Time-domain decomposition; (b) Frequency-domain decomposition; (c) Frequency-domain recomposition.

There are several detailed issues that need to be considered.

a) How to choose M? The value M determines the resolution of our frequency decomposition. It should not be too small. For example, for a multi-carrier modulated PHY with C sub-carriers, M should not be less than C. Otherwise, we will introduce inter-carrier interference during our reshaping operations. On the other hand, if M is too large, the FFT/iFFT operation may create much overhead as the computation complex of FFT increases as $O(M \log M)$. In this paper, we take the following rule-of-thumb to pick up a proper M for frequency decomposition,

$$M = \max(C, M_{min}),$$

where M_{min} specifies the minimal resolution. We choose $M_{min} = 64$ in this paper.

b) How to choose N? We should choose N sub-carriers that are large enough to cover all the physical bands (maybe non-contiguous). We denote b_v the width of virtual baseband, b_s the aggregated bandwidth, and b_{span} the width of the span of the allocated physical bands. We only consider the case where $b_v = b_s$. Later, we will discuss the case when $b_v > b_s$ [2]. Then, N should satisfy the following condition,

$$N \geq M \frac{b_{span}}{b_v}. \quad (1)$$

[2]Note that SVL shall not allocate more physical bands than required by the virtual baseband.

To ease the computation of FFT, we choose N as the smallest power of 2 that satisfies Equation 1.

c) How to map N sub-carriers to physical bands? Let f_{span} be the central frequency of the span (B_{span}) of allocated physical bands. We will first shift each physical band, $B_{p,i}(f_i, b_i) \in \Theta$, by $(-f_{span})$ to be $\hat{B}_{p,i}(f_i - f_{span}, b_i)$. Then, a subcarrier is *available* if it is covered by any $\hat{B}_{p,i}$. An available sub-carrier can be mapped to a frequency component of a baseband signal. It is easy to see there should be at least M available sub-carriers if Equation 1 is satisfied.

d) What if the number of samples in a symbol is different from M? It is a common case that the samples in one PHY symbol, *i.e.* K, is different from M. For single-carrier PHY, K is usually smaller than M; while for multi-carrier PHY, K may be greater due to the use of cyclic-prefix (CP). By perform M-point FFT and N-point iFFT, the decomposition enlarges the signal bandwidth by a factor of $\beta = \frac{N}{M}$. At the same time, it converts K samples of a symbol to βK samples accordingly.

If $K \leq M$, we simply pad zero to the K sample before perform M-point FFT. As we know from DSP theory, zero padding in time-domain samples does not change the frequency response of a signal. Then, after signal re-arrangement and N-point iFFT, we will obtain N time-domain samples, with which only the first βK samples are significant. So, we output only these βK samples and truncate the rest.

On the other hand, if $K > M$, we perform FFT as follows. We perform M-point FFT for every M samples until to the tail of the symbol where the remaining samples, say L samples, are less than M. Next, we perform an additional M-point FFT from the $(K-M)$th to Kth sample. Since we artificially shift FFT window by $(M-L)$ samples, it will cause a phase rotation in the frequency domain. Thus, we need to compensate it before performing N-point iFFT. The compensation is done by rotating the phase of values on corresponding sub-carriers. Specifically, if a frequency component i has been assigned to sub-carrier j, then we multiply the sample at sub-carrier j by a factor $e^{j2\pi \frac{M-L}{M}(j-i)}$. After N-point iFFT, this last group of samples should overlap with its previous group by $\beta(M-L)$ samples. Then, we take an average for these $\beta(M-L)$ samples as output.

Similar operations are performed by recomposition at the receiver side. But with recomposition, it will reduce the signal bandwidth by β. Accordingly, it takes K samples from the physical bands and regenerates $\frac{K}{\beta}$ virtual baseband samples.

e) What if the aggregated bandwidth (b_s) of allocated physical bands is less than the virtual bandwidth (b_v)? In previous discussion, we assume the aggregated bandwidth of allocated physical bands, b_s, equals to b_v. Here, we consider the case when $b_s < b_v$. We artificially scale the frequency of these physical bands by a factor of $\alpha = b_v/b_s$. Thus, the aggregated bandwidth of the scaled physical bands, say $\hat{b}_s$, is equal to b_v. The same decomposition/recompoistion operations are performed with these scaled physical bands. Later, the *bandwidth adjustment* module will compensate this back as discussed in Section IV-A2.

2) Bandwidth and Sampling Rate Adjustment: This module has two main tasks. First, as aforementioned, signal decomposition may scale the physical bandwidth by α to facilitate the signal splitting. This should be compensated here by reducing the signal bandwidth by α. It is done by interpolation. To understand how it works, let's assume a signal s has a bandwidth b with a sampling rate f_s. To reduce the bandwidth of s, we can add more samples to s (interpolation). For example, if we interpolate twice more samples to s, with the same sampling rate f_s, the bandwidth of s is reduced by half.

Similar, to reduce the signal bandwidth by α, we should add α times more samples. This is achieved by interpolation and decimation. Without loss generality, we assume $\alpha = k/l$, k and l are integers. This is done with following three steps:

1) Zero padding. For every sample, it pads $(k-1)$ zeros.
2) Low-pass filtering. A low-pass filter is applied the the zero-padded samples to remove the high-frequency signal image.
3) Decimation. We pick up a sample for every l samples to get the final signal.

The second task is to adjust the sampling rate to match the RF front-end. This is accomplished by re-sampling the signal with the real sampling rate of the RF front-end. The re-sampling operation is again achieved by interpolation and decimation. Let f_s be the sampling rate after the bandwidth adjustment, and f_r is the real sampling rate of the RF front-end. Let f_{LCM} be the least common multiple of both f_s and f_r. To re-sampling, we first increase the sampling rate of s to f_{LCM} by interpolation, then perform similar low pass filtering and decimation to get the final signal with desired sampling rate. Note that both bandwidth and sampling rate adjustment are actually involving same DSP operations of interpolation and decimation. So they can be combined together to save computation.

At the receiver side, the inverse operations are performed. The received signal is sampled at f_r, and it should be adjusted to match the sampling rate of the virtual baseband before sent to the *recomposition* module.

3) Frequency Band Shifting: The signal generated by N-point iFFT after signal decomposition is centered at zero. As discussed earlier, when we map N sub-carriers to the physical bands, we artificially shift the physical bands by $-f_{span}$. Here, we should compensate this back to make the signal pass the real physical bands. Shifting a signal s is to multiply a factor of $e^{j2\pi f_h i}$ to each digital sample $\{x_i\}$ of that signal. At the sender side, we shift the signal by f_{span} before sending it to the *mixer*; Similarly, at the receiver side, we shift the signal from the *splitter* by $-f_{span}$ before sending it to *bandwidth and sampling rate adjustment*.

4) Understanding Reshaping: While bandwidth adjustment, re-sampling and frequency band shifting are common used techniques for digital signal processing and may be well-understood [11], it is not obvious on the impact of signal decomposition and recomposition. In this section, we present a

simple mathematical model on this. For the sake of simplicity, we only consider the case where a virtual baseband is split to two separated physical bands, say B_1 and B_2. We note the similar technique can be also applied to analyze the case with more separated bands, but we omit it due to the space limitation. Without losing generality, we assume one band is centered at f_h and the other one is centered at $-f_h$[3]. We assume $b_v = b_{p,1} + b_{p,2}$ in our analysis. Thus, the bandwidth adjustment and frequency shifting operations can be ignored.

We denote $x(t)$ as the baseband signal, and $X(f)$ is the FFT of $x(t)$. Let $X_1(f)$ contain all sub-carriers that are mapped to B_1 and $X_2(f)$ contain other sub-carriers mapped to B_2. Then, according to the linearity of FFT, we have

$$x(t) = x_1(t) + x_2(t), \tag{2}$$

where $x_1(t)$/$x_2(t)$ is the iFFT of $X_1(f)$/$X_2(f)$. To shift the frequency of a signal by f, we multiply its time-domain samples by $e^{j2\pi ft}$. Thus, the transmitted signal $x^T(t)$ is

$$x^T(t) = x_1(t)e^{j2\pi f_h t} + x_2(t)e^{-j2\pi f_h t}. \tag{3}$$

The received signal $y^R(t)$ can be modeled as

$$y^R(t) = h * x^T(t - t_0)e^{j2\pi f_\delta t}, \tag{4}$$

where h models the multi-path fading channel, t_0 models the timing synchronization error, and f_δ models the frequency synchronization error.

Plugging in Equation 3 into Equation 4, we have

$$\begin{aligned} y^R(t) &= y_1^R(t) + y_2^R(t) \\ &= h * x_1(t - t_0)e^{j2\pi f_h(t-t_0)}e^{j2\pi f_\delta t} + \\ &\quad + h * x_2(t - t_0)e^{-j2\pi f_h(t-t_0)}e^{j2\pi f_\delta t}. \end{aligned} \tag{5}$$

The recomposition operation will shift frequency back for $y_1^R(t)$ and $y_2^R(t)$ to recover the baseband signal. Then, we get

$$\begin{aligned} y(t) &= y_1^R(t)e^{-j2\pi f_h t} + y_2^R(t)e^{j2\pi f_h t} \\ &= h * x_1(t - t_0)e^{-j2\pi f_h t_0}e^{j2\pi f_\delta t} + \\ &\quad + h * x_2(t - t_0)e^{j2\pi f_h t_0}e^{j2\pi f_\delta t} \\ &= h * x'(t - t_0)e^{j2\pi f_\delta t}, \end{aligned} \tag{6}$$

where $x'(t - t_0) = x_1(t - t_0)e^{-j2\pi f_h t_0} + x_2(t - t_0)e^{j2\pi f_h t_0}$.

From Equation 6, we can see a number of points. First, in the recovered baseband signal $y(t)$, all wireless channel fading and the frequency offset are preserved. This is the desire behavior of SVL. As we discussed in Section III, we believe only the PHY itself knows the best way to handle channel distortions.

Second, the recovered signal contains a multi-path version of the original transmitted signal, *i.e.* different portions of the signal have different fading coefficients. In other words, the decomposition/recomposition may create additional multi-path fading for non-contiguous spectrum bonding compared to the contiguous case. We note such self-generated multi-path effect is essential since when using separated spectrum bands, we indeed let different portions of a signal pass different paths. This multi-path effect reduces with the decrease of f_h and t_0. When either reduces to zero, this multi-path effect is completely removed[4]. This means an accurate timing synchronization is desired. As we will show later, we achieve this by adding Layer 0.5 synchronization sequences.

Finally, we note that for some PHY that are designed to handle multi-path channel, such self-generated multi-path effect may have less adverse impacts. For example, for OFDM, the constant multi-path coefficients for $x_1(t-t_0)$ and $x_2(t-t_0)$ may just result a constant phase-shift on a sub-carrier, which can be easily handled by equalization without losing performance. As we will show later with our experiments, spectrum spreading PHY (*e.g.* 802.11b 1/2Mbps and Zigbee) can also handle multi-path channel well. But for other PHY, like CCK in 802.11b 5.5/11Mbps, the demodulation performance will be significantly reduced with multi-path fading [16]. Thus, for such PHY, we would suggest to disable non-contiguous spectrum bond by specifying *SVL_VS_NO_SPLIT* policy when registering to SVL.

B. Protocol Timing

When SVL maps a virtual baseband to a physical band with a narrower width, it will take more time to transfer baseband signals than a PHY would imagine. For example, if an 802.11a PHY with $20MHz$ baseband is mapped to a $10MHz$ physical band, it may take SVL actually $8\mu s$ to send a symbol instead of $4\mu s$ as expected by the virtual PHY. These changes in timing may impact the correctness of the wireless protocols (*e.g.* MAC *etc.*) if they rely on absolute time information. For example, Network Allocation Vector (NAV) and ACK timeout would be expired pre-maturely if the transmitting time of PHY signal is extended.

In this paper, we use *virtual clock* to address this issue. SVL provides each virtual PHY a *virtual clock*, whose ticking rate is adaptive according to the actual allocated physical spectrum bands. Specifically, let b_s be the aggregated bandwidth of allocated physical bands, and b_v be the width of virtual baseband. We adjust the ticking rate by a factor of b_s/b_v. Since virtual PHY/MAC always measure the timing using the virtual clock, the protocol behavior automatically adapts to the current spectrum allocation, and there is no need to modify the protocol implementation itself.

C. RF Front-end Multiplexing

SVL supports multiple PHY multiplexing on the same wide-band RF front-end as long as the width of the RF front-end can cover the span of physical bands allocated to these PHY. SVL uses *mixer* and *splitter* to support multiple PHY multiplexing.

The mixer sits in the transmitting chain, which collects the physical band signals of each PHY (after reshaping), scales the signal amplitude according to each PHY's power mask and then adds (mix) them up before sending to DAC of the

[3]This condition can always be satisfied by proper shifting the central frequency of these two bands.

[4]When $f_h = 0$, it means a contiguous physical channel.

RF front-end. On the other hand, the splitter contains a set of band-pass filters, each of which matches a physical band that has been allocated to a PHY. For PHY that has non-contiguous physical bands, filters to all bands are combined to form a single band-selective filter. The splitter applies a matched band-selective filter for each PHY and the filtered signal samples are fed to the corresponding reshaper to the PHY.

It should be noted that if a device has only one half-duplex RF front-end, multiplexing multiple PHY may require careful scheduling, since such a RF front-end can only be either receiving or transmitting. Thus, different PHY should be scheduled to transmit and receive simultaneously. It is easy for SVL to deploy such a scheduler to synchronize the behaviors of multiple PHY. SVL uses buffers to temporarily hold baseband samples when the RF front-end is receiving, and defers the transmissions until the receiving is done (*i.e.* detecting no signal power). And SVL can also hide this buffering latency from PHY by subtracting it from the *virtual time*. However, when working in a network of several nodes with different PHY, there may be a potential *"blind node"* problem. When one PHY is transmitting on a spectrum band, it will put the RF front-end into transmitting mode. This will prevent other PHY mapped to the same RF front-end from receiving any incoming signals, even though they are on an orthogonal band. Such a problem is in general difficult and needs further studies.

On the other hand, RF front-end multiplexing would work best if RF transmitting chain and receiving chain can work simultaneously in a full-duplex mode. It can be achieved with a full-duplex RF front-end, or attaching two half-duplex RF front-ends to SVL. Full-duplex RF front-ends are quite common today for FDD cellular systems. We believe for future DSA systems, it should enable both TX and RX chains of a wide-band agile RF front-end simultaneously. It is because wireless PHY may use only portion of RF bandwidth for sending and therefore it could receive at same time on a separated band.

It should be noted that the full-duplex mode discussed here is different from that in [7]. In [7], Choi *et. al.*, tried to enable both sending and receiving in the *same* spectrum band, which is far more challenging. In our case, the sending and receiving bands are orthogonal and analog notch (band-stop) filters can be applied to filter out self-transmitted signals to prevent the receiving chain from being saturated.

V. Implementation

We have implemented a prototype SVL on Sora, a high-performance and fully programmable software radio platform based on general-purpose PC architectures [13].

A. Implementation architecture

Figure 9 illustrates the software architecture. For programming convenience, we place the entire networking stack in user-space. Each instance (represented only by PHY here) is implemented as a user-mode thread. Each PHY uses memory mapping to directly access the kernel DMA buffers that store the incoming digital samples (*Rx Buf*), and allocates a local buffer for outgoing samples (*Tx Buf*). SVL is linked into each PHY as a library, providing all the necessary signal processing functions, such as reshaper and splitter. The mixer function, which collects all the generated samples from each PHY's Tx Buf and mixes them into the *Tx DMA buffer*, is implemented as a separated user-level thread.

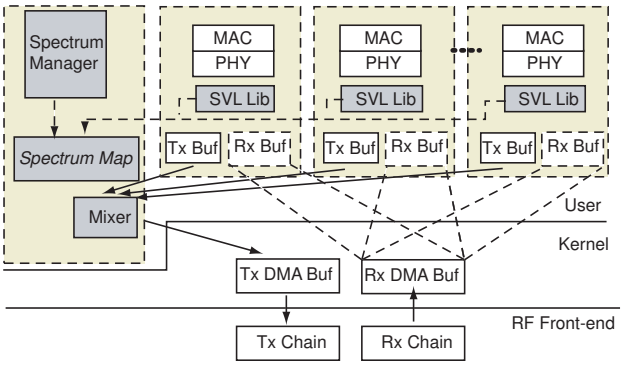

Fig. 9. Implementation architecture of SVL on Sora.

The spectrum manager is a separate program. It manages a database called *spectrum map* for all PHY instances, and dynamically configures the reshaper/splitter parameters at their SVL library to yield the desired virtualization effects. The actual spectrum management function is however rudimentary in the current version and serves only as a proof-of-concept: it reads a sequence of frequency allocation changes from a configuration file or from the command line. Sophisticated spectrum management algorithms can be implemented later as DSA research progresses.

B. Layer 0.5 Synchronization

Since we use FFT/iFFT to decompose a signal to support non-contiguous spectrum bonding, we need to perform N-point FFT on the aligned time-domain samples at the receiver side to correctly recompose the signal. To achieve this, we need to add some special sequence (sync-sequence) at Layer 0.5 to synchronize the symbol boundaries.

Sync-sequence is generated as follows. Using a long-enough master pseudo-random noise (PN) sequence, we take the first M samples (the same M as in M-point FFT used by the PHY for decomposition) and map them to the corresponding sub-carriers as if they were the frequency components of a PHY symbol. Then we perform N-point iFFT to generate the time-domain samples of the sync-sequence. Note that when sender and receiver sides agree on the same physical band allocation, the sync-sequence between them is also then determined.

At the sender side, this sync-sequence is inserted as header of a block of PHY symbols (*e.g.* a whole PHY frame). The overhead is one PHY symbol per frame, or less than 1% for a 1500-byte frame in 802.11g with 36Mbps data rate. At the receiver side, the recomposition module will continuously look out for this sync-sequence. The symbol boundary is then marked whenever such a sync-sequence is detected.

We note the synchronization is needed only when the PHY uses non-contiguous bonding. If the baseband is mapped to a contiguous physical band, there is no need for the decomposition and recomposition operations, and hence no need for keeping FFT boundary information.

VI. SVL Applications

By adding spectrum programmability, SVL is a powerful platform to build many applications. One instance is that we can readily build a TV White-space network using existing 802.11 technology with SVL. White-Space networking allows opportunistic use of unused TV channels for data communications. In United States, the White-Space spectrum is 512-698Mhz and each TV channel occupies 6Mhz. We have designed an immediately deployable White-Space network based on SVL and the very proven 802.11g PHY. We use SVL to virtualize a contiguous 20Mhz baseband over the 512-698Mhz band. As long as there is a suitable RF front-end, such as USRP WBX, we can establish a WiFi network on any number of spare TV channels. While SVL maps one or more spare TV channels (contiguous or non-contiguous) into a 20Mhz virtual baseband, the 802.11g PHY will establish links as if it were in 2.4Ghz band. This solution differs from the WhiteFi project [3] in following two important ways: 1) WhiteFi has only a limited set of channel width configurations, *i.e.*, 5/10/20MHz. Therefore, it may not be able to fully utilize the available TV Whitespace. For example, when there is a whitespace crossing three TV channels, WhiteFi may use only 10MHz, but the available frequency band is 18MHz. In contrast, SVL can make full use of all available frequency due to its flexible spectrum programmability. 2) WhiteFi supports only contiguous spectrum bands, while SVL can utilize both contiguous and non-contiguous frequency bands.

Another application of SVL is to facilitate the integration and co-existing of heterogeneous wireless systems. As mentioned earlier, there is an increasing number of different wireless standards, each of which is designed to fit a specific application need. It is a common case that multiple devices with heterogeneous wireless standards are simultaneously used in a vicinity. For example, at home, a user may simultaneously talk over a cordless phone while browse pages using WiFi; while home appliance may communicate or sense with ZigBee and RFID. Currently, it needs to setup multiple Access Points (AP) or Network Controllers (NC) for each wireless standard, adding considerable management overhead and hardware cost. Further, it is also a complex issue to avoid mutual interference from one another.

With SVL, we have built *Multiple Purpose Access Point (MPAP)* that deploys multiple heterogeneous wireless standards on the same wide-band RF front-end hardware [9]. The benefit of MPAP is three-folds: First, it consolidates multiple wireless devices into a single hardware platform and thus reduces the maintenance cost. Second, based on software radio, it is flexible and extensible to support future wireless standards. Finally, SVL enables *flexible spectrum access* for all wireless PHY deployed. Thus, they can use spectrum much efficiently to achieve better coexistence.

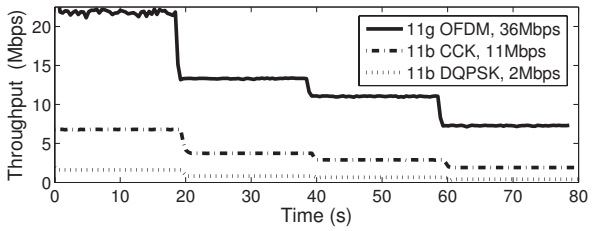

Fig. 10. Measured throughput of various PHY with variable contiguous physical spectrum allocation.

We believe, through our experience, SVL is a powerful abstraction to provide spectrum programmability. For example, both aforementioned projects require only a few days to link wireless PHY code into SVL libraries, without the need to change their original implementations. In next section, we evaluate SVL with detailed experiments.

VII. Evaluation

In this section, we evaluate the performance of SVL using our prototype implementation on four Sora stations. Each Sora station is a PC with Intel core i7-980X CPU (3.33GHz, six-core) and 3GB memory. We run several popular wireless PHYs, including 802.11b, 802.11g, and ZigBee. For 802.11b and 802.11g PHY, we use the source code from Sora SDK. We wrote ZigBee PHY ourselves because it is relatively easy. We use two kinds of 2.4GHz RF front-ends in our experiment: one with 40MHz bandwidth and half-duplex (either receive or transmit at a time), and the other with 20MHz bandwidth but full-duplex, *i.e.*, separated TX and RX chains capable of simultaneous transmitting and receiving on different spectrum bands. Although we conduct our experiments in 2.4GHz ISM band, we fully expect similar results in other spectrum bands (*e.g.* TV White-Space).

A. Flexible Spectrum Bonding

First, we evaluate SVL's ability to support flexible spectrum bonding. We measure the network throughput with different physical spectrum allocations, and validate if SVL can indeed reshape baseband signals to match the spectrum specification. We conduct the experiment under three different types of PHY, namely 802.11g (OFDM, 36Mbps), 802.11b (CCK, 11Mbps), and 802.11b (DQPSK, 2Mbps), all operating on 20MHz virtual baseband. At every 20 seconds, we change the SVL spectrum map to reduce the allocated width of the physical band from 20MHz to 10MHz, then to 8MHz, and finally to 5MHz. All PHY frames are 1000-byte long, with the exception of 600-byte for 802.11b (DQPSK) case due to a sample buffer length limitation on Sora platform.

Figure 10 shows the measured throughput under each of these three PHYs. We can see SVL can reshape virtual baseband signals to varied physical band very well. The link throughput changes proportionally in each case. For example,

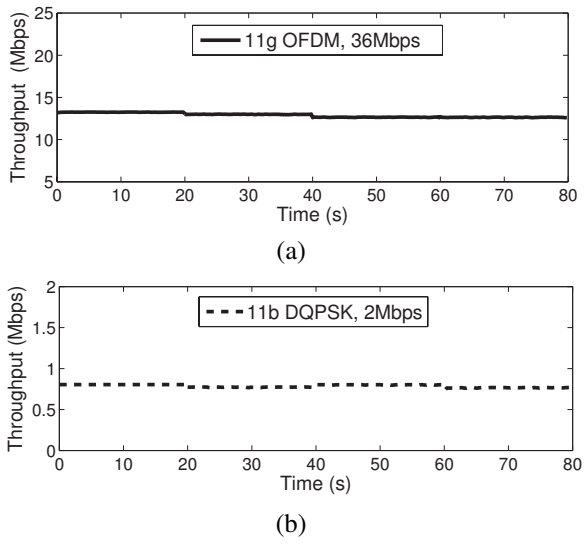

Fig. 11. Non-contiguous spectrum bonding. The sum of bandwidth of two physical band is 10MHz. (a) Measured throughput of 802.11g (OFDM); (b) Measured throughput of 802.11b (DQPSK).

when the physical bandwidth reduced from 20MHz to 10MHz, the throughput under each PHY also drops to half. This result validates our expectation that varying spectrum allocation width can be properly supported in SVL.

Next, we measure for the non-contiguous spectrum bonding case. We allocate two disjoint bands with a total bandwidth of 10MHz. At every 20 seconds, we enlarge the gap between these two bands from 0Hz (contiguous case) to 5MHz, to 8MHz, and to 10MHz. Figure 11 shows the measured throughput under two types of PHY, 802.11g (OFDM) and 802.11b (DQPSK). The results imply that neither the existence of a spectrum hole nor its size seems to affect the throughput, validating that the non-contiguous spectrum bonding can also be properly supported.

B. DSA Networking

We now further evaluate SVL in a simple DSA network. There are three different types of wireless PHY used in this network, namely 802.11g (OFDM), 802.11b (CCK), and ZigBee, all sharing the same spectrum band. We perform the experiments based on the following scenario (Figure 12(d)). At the beginning, there is only one 802.11g flow in the network and it occupies a 20MHz band as normal. At 20th second, a ZigBee flow starts, which usually takes 5MHz in the same band as in 802.11g. To avoid interference, the DSA manager intervenes and moves the allocation for the 802.11g flow away from the spectrum used by ZigBee. The new allocation now occupies two disjoint spectrum bands, with a hole in the middle (occupied by ZigBee). The sum of these two disjoint bands is still 20MHz. At 40th second, an 802.11b flow starts. The DSA manager intervenes again and assigns half of the 802.11g flow's allocation to the new 802.11b flow. Finally at 60th second, the ZigBee flow ends and at the same time the 802.11g flow's traffic requirement reduces significantly.

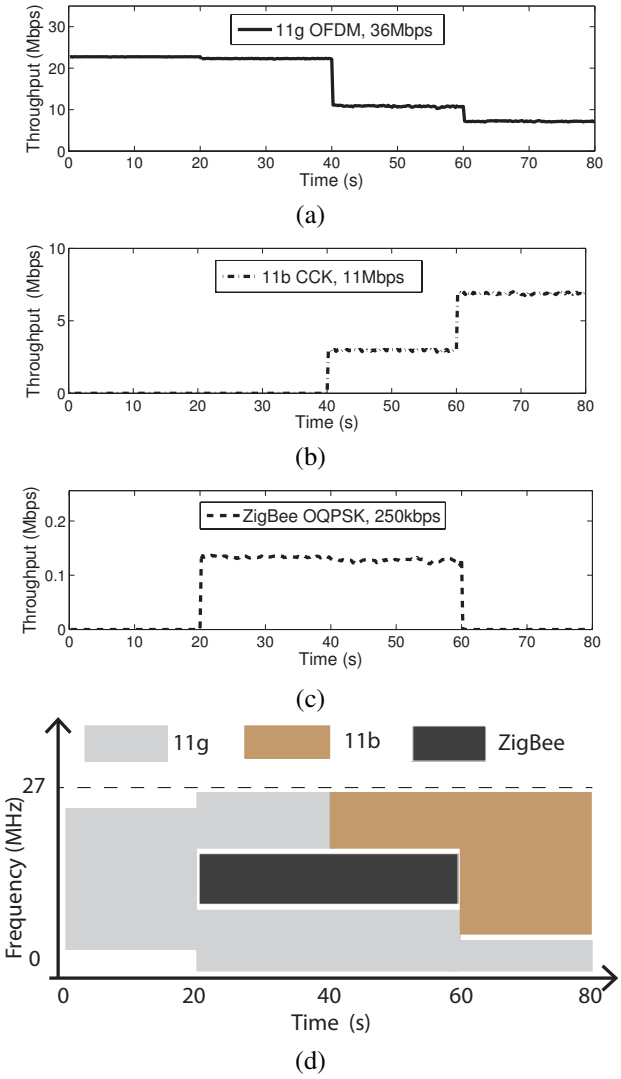

Fig. 12. Measured throughput of different flows in an example DSA network. (a) 11g(OFDM, 36Mbps); (b) 11b(CCK, 11Mbps); (c) ZigBee(250kbps); (d) The spectrum allocation.

The DSA manager then re-allocates spectrum accordingly and assigns a 20MHz band to the 802.11b flow. Figure 12(d) shows this spectrum allocation change over time.

We use an out-of-band control mechanism to coordinate all nodes in this DSA network to follow the above schedule. The frame size is 1000-byte for both 802.11b and 802.11g flows, and 30-byte for the ZigBee flow. Figure 12(a)–(c) show the measured throughput in each flow. The results largely match our prediction. The 802.11g flow receives a throughput of 22.7Mbps when it uses 20MHz spectrum in contiguous or non-contiguous bands. Then it drops to 10.8Mbps after yielding 10MHz band to the 802.11b flow at time 40. It further reduces to 7Mbps after yielding even more bandwidth at time 60. Similarly, the 802.11b flow doubles its throughput from 3Mbps to 6.9Mbps when its bandwidth increases from 10MHz to 20MHz at time 60. Since ZigBee does not change its spectrum allocation, the throughput remains the same at 130Kbps.

This experiment also demonstrates the benefit of DSA to improve spectrum usage efficiency in wireless networks. In a traditional wireless network, a ZigBee transmission would prevent 802.11g from using the same channel, even though ZigBee only occupies a small portion of the spectrum. Here in a DSA network, 802.11g can still make efficient use of the remaining bandwidth and the wireless spectrum is not wasted.

C. RF Front-end Multiplexing

SVL supports RF front-end multiplexing, *i.e.*, to multiplex multiple separate wireless networks, hence multiple PHY instances, on one full-duplex RF front-end. That is, the node can simultaneously communicate on multiple separate wireless networks (such as on different frequency bands), each via a different PHY. Since we don't have a full-duplex RF front-end yet, we use two radios in our experiment. One radio is used for transmitting and the other is for receiving. We run two instances of 802.11g PHY (OFDM, 18Mbps) on one node. Each PHY is allocated with a 5MHz physical band and is expected to have a throughput around 4Mbps.

In the experiment, we turn on both PHYs and put one in receive mode and the other in transmit mode. We are particularly interested in this case because it is easier otherwise (both in the same receiving or transmitting mode). We vary the "distance" by which the two physical bands are separated from each other, and measure the throughput of both PHYs. The result is listed in Table I. It clearly shows that, as long as the two physical bands are orthogonal (e.g., Distance ≥ 0), each PHY can achieve the same throughput independent of the other PHY, as the SVL's splitter filters and the guard-bands employed by PHY can already handle the cross-channel interference very well. This indeed validate that a full-duplex RF front-end can be easier to support two simultaneous PHYs, if these two signals occur on different spectrum bands.

TABLE I
THROUGHPUT MEASURED OF TWO PHY MULTIPLEXING ON A SINGLE FULL-DUPLEX RF FRONT-END.

	Distance			
	0MHz	**2MHz**	**4MHz**	**8MHz**
PHY1	3.97Mbps	3.98Mbps	3.98Mbps	3.99Mbps
PHY2	4.06Mbps	4.06Mbps	4.06Mbps	4.06Mbps

D. Reshaping Operation Precision

SVL employs sophisticated digital signal processing (DSP) to reshape the baseband signals, which will introduce additional processing errors. To quantify these errors, we first reshape baseband signals to physical band signals, and then immediately reshape them back into baseband. Then we compute the mean square error (MSE) between the original signals and the "reshape-back" signals. Table II summarizes MSE values normalized to the signal energy when we reshape a 20MHz baseband to two 5MHz non-contiguous physical bands and back. Rows marked "11g" show the results when 802.11g PHY (OFDM, 36Mbps) is used and rows marked "11b" are the results when 802.11b PHY (DQPSK, 2Mbps) is used. The first column labels the bit-width of our fix-point data in DSP algorithms. We also vary the choice of FFT points (N). As we have discussed in Section IV, a larger N means a wider separation of these two non-contiguous physical bands.

We can see that the processing error increases with FFT points. With 128 point FFT and using 16-bit fix-point DSP, the average error is about -17dB below the signal energy, which is acceptable for many PHYs like 802.11b and 802.11g at a rate of 36Mbps or below. If we want to support higher data rate, or if we want to bond to widely separated bands with large N, we will need to increase our computation precision by using 32-bit fix-point DSP. At this point, we believe 32-bit will be good enough because the error introduced is below -40dB (Table II), which is much lower than noise from a common wireless channel.

TABLE II
ERROR INTRODUCED BY SVL DSP OPERATIONS (DB).

Int-Width	**PHY**	**N-Point**			
		128	256	512	1024
16b	11g	-17	-15.2	-13.3	-8.3
	11b	-17.8	-15.9	-13.6	-10.7
32b	11g	-46.4	-43.9	-41	-38.1
	11b	-48	-45.4	-42.6	-39.8

VIII. RELATED WORK

Dynamic spectrum access (DSA) has received a lot of attentions in the research community in recent years [2]. Several DSA systems have been implemented [3], [12], [14], [15]. For example, White-Fi [3] uses UHF White Space for data communication. It takes a commodity Atheros Wi-Fi card and down-converts the Wi-Fi signals to UHF band and vice versa. It further supports multiple contiguous TV channels with variable channel width of 5, 10 and 20MHz [6]. SWIFT [12] enables wide-band wireless devices to co-exist with narrow-band users in unlicensed band. It customizes the OFDM PHY to avoid the sub-areas in the spectrum band that are used by narrow-band users. Jello [14], another OFDM PHY based DSA system, is similar to SWIFT in the sense that they both support non-contiguous spectrum bands. They differ from each other in the way they senses the channel and manages the spectrum allocation. Unlike these DSA works that proposed new cognitive PHY, our goal is to devise a network architecture so that DSA can be provided as a general function to all wireless PHY. Nevertheless, these work can also be complementary to SVL because SVL requires a separate spectrum management function and it can adopt the spectrum sensing and management approaches from these works. Picasso [10] designs a programmable frequency division full-duplex radio that enables simultaneously transmitting and receiving on separated available frequency bands on the same RF hardware. Each available frequency band is operated separated, and non-contiguous frequency bonding is not supported. The Picasso

full-duplex radio can be one of radio platforms to implement SVL.

The term *Virtual Radio* was proposed by Bose, *et. al.*, about a decade ago [4], to refer to a software radio running on general purpose PC architecture. With many recent efforts (like [13]), software radio is increasingly recognized as a powerful enabler for cognitive and DSA systems. Indeed, SVL can be implemented on a software radio platform.

IX. Conclusion and Future Work

In this paper, we argue that *spectrum programmability* is an independent property that can be separated from the general PHY modulation design. We propose to add a new *spectrum virtualization layer (SVL)*, at *Layer 0.5*, to support flexible spectrum programmability, and enable flexible spectrum access for general wireless networks.

SVL provides a *virtual baseband* abstraction to wireless PHY, which is static, contiguous, with a desirable width defined by the PHY. At the sender side, SVL performs real-time DSP operations to reshape the modulated baseband signals into waveform that matches the dynamically allocated physical bands, while keeping the modulated information unchanged. At the receiver side, SVL performs inverse reshaping operation that collects the waveform from each physical band, and reconstruct the original modulated signals for the PHY. All these reshaping operations are performed at the signal level without the detailed knowledge on baseband modulation.

Beside, SVL further provides the flexibility to map wireless PHY to RF front-ends. It can map different PHY to different RF front-ends, and it is also able to multiplex multiple PHY onto single RF front-end. Thereby, it enables multiple PHY to share a common powerful wide-band RF hardware and simplifies the integration of multi-radio in one mobile device and reduces the cost.

We have implemented a prototype of SVL based on a software radio platform. Our experiments show that our SVL design are flexible and effective to support DSA for general wireless designs with affordable overheads.

There are several future directions we are going in: First, SVL can facilitate the takeoff of TV Whitespace networking by adopting existing well-understood wireless designs into whitespace with little efforts. Second, SVL is flexible to deploy various dynamical spectrum management algorithms. So we can evaluate and compare these algorithms with a single common platform. Finally, we will continue optimizing our reshaper design in software and explore the possibility to accelerate its processing using hardware.

References

[1] IEEE tgac draft amendement for IEEE 802.11 wireless lan.

[2] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty. Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey. *COMPUTER NETWORKS JOURNAL (Elsevier)*, 2006.

[3] P. Bahl, R. Chandra, T. Moscibroda, R. Murty, and M. Welsh. White space networking with wi-fi like connectivity. In *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, SIGCOMM '09, pages 27–38, New York, NY, USA, 2009. ACM.

[4] V. Bose, M. Ismert, M. Welborn, and J. Guttag. Virtual Radios. *IEEE Journal on Selected Areas in Communications*, 17(4), April 1999.

[5] D. Cabric, A. Tkachenko, and R. W. Bordersen. Experimental study of spectrum sensing based on energy detection and network cooperation. In *ACM 1st International Workshop on Technology and Policyy for Accessing Spectrum*, 2006.

[6] R. Chandra, R. Mahajan, T. Moscibroda, R. Raghavendra, and P. Bahl. A case for adapting channel width in wireless networks. *SIGCOMM Comput. Commun. Rev.*, 38(4):135–146, 2008.

[7] J. I. Choi, K. Srinivasan, M. Jain, P. Levis, and S. Katti. Achieving single channel, full duplex wireless communication. In *ACM Mobicom*, 2010.

[8] FCC press release. FCC adopts rules for unlicensed use of television white spaces. 2008.

[9] Y. He, J. Fang, J. Zhang, H. Shen, K. Tan, and Y. Zhang. MPAP: Virtualization Architecture for Heterogenous Wireless APs. *CCR Online (Best demo in SIGCOMM 2010)*, 2010.

[10] S. Hong, J. Mehlman, and S. Katti. Picasso: Full duplex signal shaping to exploit fragmented spectrum. In *HotNets'11*, 2011.

[11] A. V. Oppenheim, R. Schafer, and J. Buck. *Discrete-Time Signal Processing (2nd Ed)*. Prentice-Hall, 1999.

[12] H. Rahul, N. Kushman, D. Katabi, C. Sodini, and F. Edalat. Learning to share: narrowband-friendly wideband networks. *SIGCOMM Comput. Commun. Rev.*, 38(4):147–158, 2008.

[13] K. Tan, J. Zhang, J. Fang, H. Liu, Y. Ye, S. Wang, Y. Zhang, H. Wu, W. Wang, and G. M. Voelker. Sora: High performance software radio using general purpose multi-core processors. In *NSDI 2009*.

[14] L. Yang, W. Hou, L. Cao, B. Y. Zhao, and H. Zheng. Supporting demanding wireless applications with frequency-agile radios. In *In Proc. of NSDI (2010)*, 2010.

[15] L. Yang, B. Y. Zhao, and H. Zheng. The spaces between us: setting and maintaining boundaries in wireless spectrum access. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking*, MobiCom '10, pages 37–48, New York, NY, USA, 2010. ACM.

[16] P. Yang and D. Noneaker. Performance analysis of cck modulation in a multi-path channel. In *Research Report at UC Berkeley*, 2002.