BCMSDR: Breaking the Cost Barrier of High-Rate SDRs Using Commercial Wi-Fi Hardware

Jakob Link
TU Darmstadt
Darmstadt, Germany
jlink@seemoo.tu-darmstadt.de

Stefano Fontana*
University of Brescia/CNIT
Brescia, Italy
stefano.fontana@unibs.it

Denis Festa*
University of Brescia/CNIT
Brescia, Italy
d.festa004@unibs.it

Matthias Hollick
TU Darmstadt
Darmstadt, Germany
IMDEA Networks Institute
Madrid, Spain
matthias.hollick@tu-darmstadt.de

Francesco Gringoli*
University of Brescia/CNIT
Brescia, Italy
francesco.gringoli@unibs.it

Abstract

Software-defined radios (SDRs) are a key tool for wireless systems research, but platforms that support high sampling rates and wide bandwidths remain costly and are hard to scale beyond small lab deployments. At the same time, modern commercial off-the-shelf (COTS) Wi-Fi devices already contain capable radio front-ends and baseband hardware, but expose only packet-based interfaces.

In this paper, we show how to repurpose selected Wi-Fi devices into low-cost SDR platforms with direct access to raw IQ samples. We design and implement a host-to-baseband streaming architecture that enables raw IQ sample transmission and reception using commercial Wi-Fi hardware. The proposed platform, BCMSDR, supports continuous sending or receiving at 20 MS/s. Additionally, it supports burst-based operation with timed, non-continuous modes at 40 MS/s, 80 MS/s and 160 MS/s. Our platform works in the 2.4 GHz, 5 GHz and 6 GHz Wi-Fi bands on selected `BCM43` chipsets. We provide a comprehensive experimental characterization, including IQ imbalance and RF impairments, validating the suitability of the platform for advanced physical layer research. To the best of our knowledge, this work presents the first SDR solution that offers up to 160 MS/s sampling rates and continuous streaming at 20 MS/s on COTS Wi-Fi hardware at a cost below 350 USD.

*Also with National Inter-University Consortium for Telecommunications.



This work is licensed under a Creative Commons Attribution 4.0 International License.
WiNTECH '26, Austin, TX, USA
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2879-2/26/10
https://doi.org/10.1145/3831662.3844169

CCS Concepts

• **Networks** → **Network performance analysis**; **Network measurement**; **Network experimentation**; • **Security and privacy** → **Software reverse engineering**.

Keywords

Software Define Radio (SDR), Wi-Fi Chip, Repurpose.

ACM Reference Format:
Jakob Link, Stefano Fontana, Denis Festa, Matthias Hollick, and Francesco Gringoli. 2026. BCMSDR: Breaking the Cost Barrier of High-Rate SDRs Using Commercial Wi-Fi Hardware. In *20th ACM Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization (WiNTECH '26), October 26–30, 2026, Austin, TX, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3831662.3844169

1 Introduction

The ubiquity of Wi-Fi technology, with billions of devices installed worldwide, has made it the primary medium for local wireless communication. However, the vast majority of these devices utilize chipsets that operate as closed-source Ethernet-to-Wi-Fi bridges. This architectural design abstracts the physical layer (PHY) and medium access control (MAC) layers from the host system, drastically restricting the ability of researchers to implement custom protocols or interact directly with raw baseband (BB) signals on commercial off-the-shelf (COTS) devices [16]. Prior research and practical frameworks like Nexmon have provided vendor firmware insights and enabled patching to unlock features such as monitor mode, frame injection, PHY feature extraction, and basic raw IQ sample transmission and reception [9, 11, 15, 16]. Still, direct BB use has been limited in these approaches due to slow or complex proprietary memory interfaces and restricted sample buffer lengths [11, 15].

In this paper, we present a novel implementation of a half-duplex software-defined radio (SDR) on commercial Broadcom chipsets that bypasses the traditional Wi-Fi architecture logic entirely. Unlike previous attempts that rely on limited IQ buffers, our work introduces a novel reverse-engineered memory interface capable of data streaming between host device kernel space and Wi-Fi chip BB on selected Broadcom `BCM43` chipsets. Larger buffer sizes and memory accesses significantly faster than any previous public implementation enable targeted hardware to sustain a continuous streaming interface at 20 MS/s. Direct interaction with the digital BB and the radio frequency front-end (RFFE) bypasses IEEE 802.11 MAC and PHY completely and allows full control over wireless low-level experiments where prior approaches fail. We also support legacy duration-limited one shot or looped operation with rates up to 160 MS/s and a maximum bandwidth of 160 MHz. Furthermore, we expose control over front-end configurations. This includes arbitrary frequency tuning, transmit and receive gain control, selecting filter configurations, and using bandwidths outside the rigid definitions of the IEEE 802.11 standard. By deactivating hardware-controlled power loops and manually configuring programmable gain amplifier (PGA) and power amplifier (PA), our tool, BCMSDR, can achieve a maximum power output of 30 dBm [5].

These enhanced capabilities provide an ideal foundation for low-level wireless experimentation. One popular field of application is cross-technology communication (CTC). Traditional CTC methods on COTS devices often rely on complex signal emulation—where Wi-Fi payloads are meticulously chosen to mimic the waveforms of technologies like LoRa or ZigBee [13, 18]. Our implementation, instead, allows for direct injection of natively modulated signals into the analog front-end and even enables reception of CTC signals. This allows devices to act as universal gateways that can communicate directly with diverse Internet of Things (IoT) ecosystems, such as industrial sensors or smart home units, without specialized intermediary hardware [15].

By providing all project sources freely accessible in a build-ready state[1], we enable lower-cost COTS devices to be used as flexible wireless research platforms by a large community.

2 Background and Related Work

While specialized hardware like USRP, LimeSDR, and HackRF provide flexibility for PHY research, their cost and features shown in Table 1 limit large-scale mobile deployments.

Several projects have shown that low-cost hardware can provide parts of the functionality commonly associated with SDRs. The best known example is RTL-SDR, which repurposes DVB-T USB receivers based on the RTL2832U chipset to access raw IQ samples in software [1]. RTL-SDR made spectrum access widely available at very low cost, but it is receive-only, offers only a few MS/s, and common tuner variants do not reach the 2.4 GHz and 5 GHz bands where most modern wireless communications happens. Other low-cost projects target the opposite direction. `Rpitx` uses the Raspberry Pi clock hardware as a general RF transmitter and can generate signals without an external radio front-end, except for filtering [4]. This makes it useful for experiments and simple transmissions, but it does not provide a receiver and relies on a general-purpose single-board computer rather than an actual wireless transceiver. A different example is `cm-sdr`, which extracts complex samples from a cable modem's receiver path [17]. This demonstrates that unexpected COTS devices can expose useful signal data, but the resulting system is a receive-only prototype with strong limits on continuous streaming and practical use as a general SDR. These systems show the potential of repurposing cheap hardware, but they usually provide only one direction, are limited in bandwidth or frequency range.

Commodity Wi-Fi hardware itself has also been used to obtain SDR-like receiver functionality. SDR-Lite tricks unmodified Wi-Fi receivers into decoding an ambient signal as if it were the payload of a Wi-Fi frame [10]. The resulting bit stream is then used to reconstruct the underlying waveform. This approach is attractive because it does not require firmware or hardware modifications and works with common Wi-Fi devices. However, the Wi-Fi PHY and MAC have to be carefully tricked through an emulated header, and the received signal is reconstructed after Wi-Fi demodulation rather than streamed as direct BB samples. It is therefore a receive-only system and well suited for tasks such as spectrum monitoring, fingerprinting, or cross-technology reception, but it does not provide direct sample access.

Another line of work is based on Nexmon, a firmware patching framework for Broadcom Wi-Fi chips [16]. Nexmon started by exposing Wi-Fi related functionality such as monitor mode and frame injection on mobile devices, and later enabled access to more hidden low-level features. On the receive side, the Nexmon CSI extractor exposes per-frame channel state information (CSI) on Broadcom chips [8, 9]. While this is valuable for sensing, localization, and measurement applications, CSI is computed once per received Wi-Fi frame and is therefore not equivalent to continuous raw IQ sample streaming. Other Nexmon-based receive work exposes signal strength sampling for CTC. Broadcast CTC uses such measurements to receive energy bursts from heterogeneous devices and thereby enables communication between Wi-Fi, BLE, and ZigBee devices [2]. This is sufficient for protocols that encode information in packet lengths or energy bursts, but it only provides coarse power measurements. It

[1] BCMSDR sources available at http://github.com/seemoo-lab/bcmsdr.

Table 1: Frequency range, bandwidth, and transmission power of popular SDRs and two BCMSDR Asus devices.

Model	Frequency Range	Max Bandwidth	Max TX Power	Costs[d]
Ettus X310	DC → 6 GHz[a]	160 MHz	10 dBm[a]	11 400 USD
Ettus B206mini	70 MHz → 6 GHz	56 MHz	16 dBm	1820 USD
LimeSDR Mini	10 MHz → 3.5 GHz	40 MHz	10 dBm	600 USD
HackRF One	1 MHz → 6 GHz	20 MHz[b]	15 dBm	300 USD
Asus GT-AXE11000 (`BCM43684c0`)	2.4, 5, and 6 GHz Wi-Fi bands	160 MHz[c]	30 dBm [6]	370 USD
Asus RT-AX86U (`BCM43684b0`)	5 GHz Wi-Fi band	160 MHz[c]	30 dBm [5]	350 USD

[a] Daughterboard dependent. [b] Half duplex. [c] 20 MHz streaming, 160 MHz one-shot use. [d] Market prices from official distributors at the time of writing.

cannot replace BB access when phase, frequency, or waveform details are required.

On the transmit side, prior Nexmon-based work showed that Broadcom Wi-Fi chips can play arbitrary IQ samples from internal buffers. Massive reactive smartphone-based jamming uses this capability on Nexus 5 smartphones to generate jamming waveforms stored in IQ sample buffers and transmit them in reaction to received Wi-Fi frames [14]. Shadow Wi-Fi extends this direction by transmitting raw signals from a larger chip-internal buffer and combining this with per-frame CSI extraction to build a PHY-layer covert channel [15]. These works demonstrate that smartphone Wi-Fi chips can transmit custom BB waveforms in the 2.4 GHz and 5 GHz bands. Still, they do not constitute a full SDR. Transmission is limited by chip-internal sample memories and by slow refill paths from the host. As a result, signals are either one-shot, frame-based, or looped, but samples are not continuously streamed from the host into the Wi-Fi chip.

Pair-Fi recently combined refined raw-sample transmission with raw IQ sample reception on modern smartphone Wi-Fi chips [11]. It uses this functionality to implement a fast integrity-code based out-of-band channel for secure device pairing. Pair-Fi therefore moves closer to bidirectional SDR-like access than previous Nexmon-based systems, as it demonstrates both arbitrary waveform transmission and raw IQ sample reception on commodity smartphones. However, the system is designed for short pairing signals and still relies on chip-internal sample memories and application-specific timing logic. Continuous streaming of IQ samples across the chip-host boundary is not provided.

In summary, existing low-cost SDR projects and prior Wi-Fi-based systems provide partial access to required functionality. However, no open source system provides a low-cost repurposed COTS device that can act as a practical streaming SDR with continuous raw IQ sample transfer in both directions. We fill this gap with BCMSDR.

3 SDR Device Design

Prior work has successfully demonstrated transmission and reception of raw signals in form of IQ samples on low-cost COTS devices featuring Broadcom `BCM43` Wi-Fi system-on-chips (SoCs) [11, 14, 15]. However, usage is limited to one-shot and looping operations over fixed durations because the chip-internal sample memory, which is typically dedicated to a proprietary media access controller (D11), is exposed through slow register accesses, preventing continuous streaming over the chip-host-boundary even at the lowest sampling rates.

While reverse engineering a proprietary CSI extractor by Broadcom, named CSIMON, we discovered that recent high-end `BCM43` SoCs that target access points (APs) have a slightly different architecture where the sample memory is part of the SoC system memory. Further, we found that the same chipsets implement a simplified address-based direct memory access (DMA) mechanism that can be instantiated to move data between the Wi-Fi chip sample memory and reserved host memory areas at speeds sufficient to continuously stream IQ samples at 20 MS/s over peripheral component interconnect express (PCIe).

In this work, we design and implement a system around these new insights that converts selected COTS devices into streaming-capable SDRs. Section 3.1 introduces the system architecture, which is visualized in Figure 1. For integration of our system into existing SDR applications, we create a USRP hardware driver™ (UHD) interface that is described in Section 3.2.

3.1 Internal Architecture

Our system, BCMSDR, consists of three connected parts that together form the continuous sample path shown in Figure 1. A local host runs the user application, a remote Linux host contains the target Wi-Fi chip, and a Nexmon-patched firmware on the Wi-Fi SoC controls the chip internal sample path. The local host can be any UNIX system. It communicates over Gigabit Ethernet (GbE) with a remote device that features one of the supported `BCM43` SoCs, currently `BCM43684b0` and `BCM43684c0`. We use Asus RT-AX86U and Asus GT-AXE11000 devices as development platforms, while numerous access points from Asus, Netgear, and TP-Link contain the same chipsets.

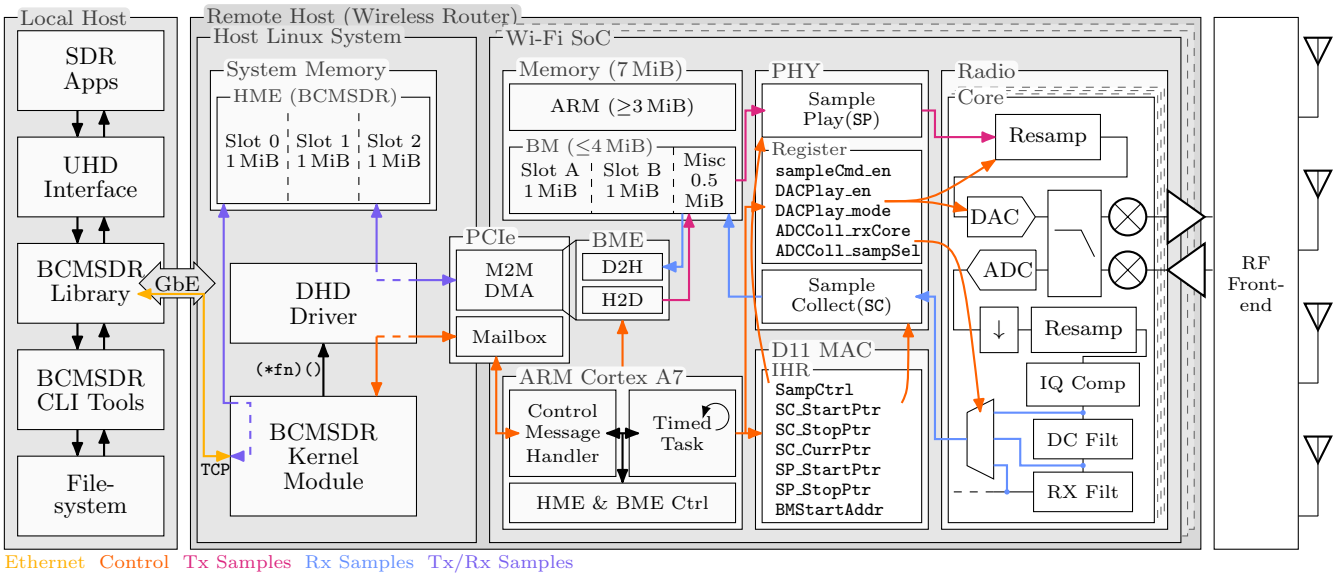


Figure 1: BCMSDR system overview, layout, and integration into the architecture of selected BCM43 chips.

On the local host, our BCMSDR library implements the logic required to control the remote system and stream samples in both directions. It sends control messages, receives samples from the remote host, and forwards transmit samples in the opposite direction. In transmit mode, the library throttles the outgoing stream to prevent overruns of the limited intermediate memories on the remote host. Control messages and samples are transported over TCP. This reuses the existing network stack for buffering, ordering, and retransmission. The samples are complex signed shorts, which means four bytes per IQ sample and thus a minimum rate of 640 Mbit/s is required at 20 MS/s. The GbE link leaves enough headroom for protocol overhead. Besides the UHD interface described in Section 3.2, BCMSDR provides command line tools that can stream samples from or to the filesystem for a fixed duration or until manually interrupted.

On the remote host, BCMSDR is split between a Linux kernel module and firmware running on the Wi-Fi SoC ARM core. The kernel module terminates the TCP connection to the local host and moves sample blocks between this connection and memory that is visible to the Wi-Fi chip. We do not modify the original Wi-Fi driver, named DHD, but use it as a provider of platform specific state. In particular, DHD allocates and manages host memory extension (HME) memory during chip initialization based on requirements announced by the Wi-Fi firmware. BCMSDR obtains the corresponding addresses and helper functions during runtime through `kallsyms` symbol lookup. The same mechanism is used to fetch and commit PCIe mailbox messages. Our kernel module directly subscribes to PCIe interrupts.

The Wi-Fi SoC provides a Broadcom memory-to-memory (M2M) DMA system that can copy data between host memory and chip memory over the PCIe bus. M2M provides endpoint pairs that can be used as DMA channels. A simpler abstraction named byte move engine (BME) is layered on top of these channels. Instead of preparing descriptor rings, a BME transfer is instantiated with a host memory address, a Wi-Fi memory address, and a copy direction, either dongle-to-host (D2H) or host-to-dongle (H2D), where "dongle" refers to the Wi-Fi SoC. Multiple BME channels can be used in parallel and polled for completion. We use this to reach the throughput required for continuous sample streaming. On our target platform, copying 1 MiB takes 3639 µs in D2H direction and 10 375 µs in H2D direction, which is faster than the 13 107 µs that it takes to cover 1 MiB at 20 MS/s.

The host memory address used by BME must point into HME, which is reserved host memory logically owned by the Wi-Fi SoC. The firmware supports up to eight HME users, who can request memory in 4 KiB blocks and up to one block less than 4 MiB. We repurpose the firmware entry originally used by Broadcom's CSIMON component and replace it with a BCMSDR HME user that receives 3 MiB. The BME Wi-Fi memory address can point anywhere into the SoC memory, which is 7 MiB large on the `BCM43684c0`. For IQ samples, however, only the buffer memory (BM) partition is usable. By default, this area is only 1.5 MiB large. We can grow the BM up to 4 MiB by patching the ARM firmware to use less memory and by modifying a D11 register.

Sample input and output is handled by two PHY level engines. For transmission, the sample play (SP) engine reads

IQ samples from BM and feeds them into the radio transmit path. For reception, the sample collect (SC) engine writes IQ samples into BM. Both engines are controlled through D11 registers that configure enable state, start and stop pointers, and whether the engine stops or loops once the stop pointer is reached. The SC engine additionally reports its current write pointer. In transmit direction, further PHY registers select the sample rate and whether the samples read by SP are actually connected to the transmission path. Calibration routines that we reverse engineered configure the rate to twice the selected Wi-Fi channel bandwidth, which would imply at least 40 MS/s, but we found that the rate can also be set to 20 MS/s. This is required for BCMSDR, as H2D copies are not fast enough to sustain 40 MS/s without underruns. In receive direction, the relevant PHY configuration enables and selects the tap point of SC. It can collect samples after an IQ compensator, after a DC filter, or after a receive low pass filter. The sample rate follows the selected Wi-Fi channel bandwidth, resulting in 20 MS/s on a 20 MHz channel. We use the DC filter output by default.

To stream samples without losses with the available BME speeds, BCMSDR uses a ping pong scheme on both sides of the DMA boundary. We split the BM into two consecutive 1 MiB sample slots, over which the active engine loops continuously. A periodic task in the Wi-Fi firmware checks whether the engine has moved from one slot to the other. The slot not currently used by the engine is then safe to refill in transmit mode, or safe to drain in receive mode. Because SP does not expose a current pointer, transmit mode runs SC in parallel on an additional 0.5 MiB BM region at the same rate. The SC pointer thereby acts as a virtual SP pointer and reveals when SP has switched slots. Once a free BM slot is detected, the firmware starts a BME copy either from HME to BM or the other way round. On the host side, we split HME into three 1 MiB slots. Two slots would be sufficient to alternate ownership between the Wi-Fi SoC and the Linux kernel module, but the third slot relaxes timing requirements and absorbs short network jitter. Slot states and ownership are tracked by the ARM firmware and synchronized with the kernel module through PCIe mailbox commands.

In receive mode, a mailbox command starts the streaming engine. Whenever SC fills a BM slot, the firmware copies it into the next free HME slot and assigns that slot to the host. The kernel module immediately forwards the slot over TCP and notifies the firmware once the memory can be reused. Transmit mode works the same way in the opposite direction, but with one additional initialization step. At first, all HME slots are owned by the host. The local host fills them at the desired throttled rate, the kernel module announces completed slots to the firmware, and the firmware copies samples into BM. Before enabling SP, both BM slots are filled with valid samples so that the periodic refill logic can take over without an initial underrun.

3.2 Interoperability

To enable interoperability with existing SDR workflows, we developed an extension to the UHD library [3] that enables UHD-based applications to communicate with the modified devices. UHD is widely used as a hardware abstraction layer for SDR platforms, providing standardized interfaces for device discovery, configuration, streaming, and synchronization. A large body of measurement and PHY-research tooling is built directly on top of it. By extending UHD to support the BCMSDR platform, the modified Wi-Fi device can be exposed to host applications through an interface that resembles conventional SDR hardware, while the underlying implementation remains tailored to the embedded communication protocol and wireless chipset constraints. Targeting UHD also yields indirect support for higher-level frameworks such as GNU Radio [7] and SoapySDR [12], which can bind to UHD devices through their existing backend. Our UHD extension implements the host-side logic required to discover the modified devices, establish communication with the embedded kernel extension, configure streaming parameters, and exchange sample or payload data during continuous operation. UHD acts as the bridge between SDR applications and the custom device-specific transport layer. We extended the library with a new UHD device family that seamlessly plugs into UHD's device discovery and factory mechanism.

The controlling host maintains an active data path to the device, allowing sustained exchange of waveform-related data or transmission payloads. Sample transfer is handled by dedicated receive and transmit streamers through a TCP data connection where raw interleaved 16-bit IQ values are transmitted. This on-the-wire representation lets us reuse UHD's optimized conversion routines to translate to the application's requested host format without additional implementation cost.

The UHD interface can be further improved by implementing optional features. One such feature that we realize is target frequency control. By directly changing the device channel allocation table, we are able to implement 1 MHz carrier frequency tuning, removing the constraint defined by the Wi-Fi channel allocation of the underlying firmware.

4 Evaluation Methodology

Initial development and validation were performed on an Asus RT-AX86U platform equipped with a Broadcom `BCM43684b0` 5 GHz band Wi-Fi chip. This device was selected as the first development target due to platform experience and ease of patching. At this stage, all experiments were limited to the 5 GHz Wi-Fi band, and later the system was ported, and

development continued, on the Asus GT-AXE11000 featuring dedicated chips for the 2.4 GHz, 5 GHz and 6 GHz bands with a newer revision of the chipset (`BCM43684c0`).

For measurements, one RF port corresponding to a single radio core of the device under test was connected directly to an Ettus B210 USRP SDR using a coaxial cable. An attenuator chain was inserted between the two devices to reduce the signal level before it reached the USRP front-end to prevent overloading conditions. This setup minimizes external propagation effects, uncontrolled interference, antenna placement sensitivity, and multipath fading.

We designed a validation experiment to test whether the implemented transmit and receive paths could preserve a known sample sequence without discontinuities. Figure 2 shows the corresponding block-flow representation. A deterministic waveform with time-varying characteristics was generated on the host using MATLAB. The same waveform generation procedure was used for transmit-side validation and receive-side validation, so that the received samples could be compared against a known reference sequence. Initially, a single tone varying in amplitude was used, to simplify interpretation of received data and identification of errors in the received or transmitted signal.

For transmit validation, the MATLAB-generated sequence was transferred from the host to the modified Wi-Fi device and emitted through the selected RF chain. The B210 captured the resulting conducted signal and forwarded the received samples back to the host for offline analysis.

For receive validation, the signal flow was reversed: The B210 generated the known waveform, and the modified Wi-Fi device captured the corresponding IQ samples. In both directions, the captured sample stream was analyzed. Results are shown in Section 5. The analysis focused on detecting failures such as buffer underruns, buffer overruns, repeated blocks, dropped sample regions, and stream discontinuities.

To make such failures observable, the waveform length was deliberately chosen so that it was not an integer multiple of the internal buffer length. This choice prevents buffer-periodic artifacts from being hidden by the periodic structure of the test sequence. If a buffer segment is repeated, skipped, or incorrectly wrapped, the resulting discontinuity appears at a non-periodic position in the reconstructed waveform and can therefore be detected by comparing the received sequence with the expected reference. This property is important for validating streaming correctness. A test signal aligned with the buffer size could otherwise mask exactly the type of errors the experiment is intended to reveal.

A second validation experiment was performed to evaluate the effective bandwidth behavior of the platform. For this experiment, a chirp waveform was generated on the host spanning all the available bandwidth, and transmitted through the same setup. A chirp was selected because its

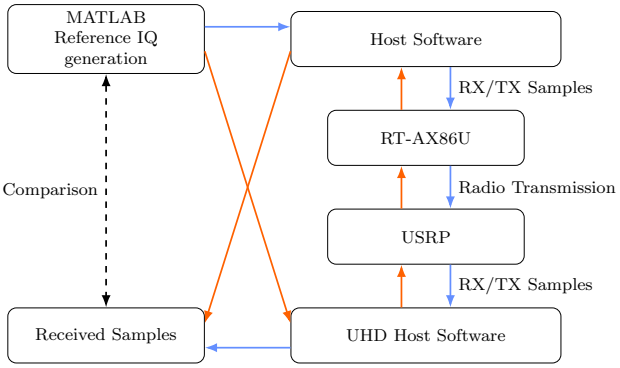


Figure 2: Block-flow representation of a conducted validation experiment. A deterministic single-tone amplitude-varying signal is used as a known reference. Received samples are checked for missing samples, repeated data, misalignment, and stream discontinuities.

instantaneous frequency sweeps over a defined bandwidth, making it suitable for observing whether the transmit and receive chain can preserve energy across the intended frequency span. Compared with a single-tone test, this provides a broader view of the usable bandwidth and can expose attenuation, filtering, or frequency-dependent distortions introduced by the baseband processing, RF frontend configuration, or external measurement chain.

The setup was later extended by replacing the B210 with an Ettus X310 USRP in order to verify one-shot transmission at 160 MS/s. In this configuration, the X310 provides the measurement bandwidth required to observe the full buffered transmission mode of the modified Wi-Fi platform. The purpose of this experiment was to validate operation at the maximum one-shot transmission bandwidth supported by the device and to confirm that the buffered baseband path can be operated at 160 MS/s.

5 Experimental Results

This section presents the experimental validation of the proposed SDR-like operation. The results focus on three aspects of the platform: sample-stream correctness, amplitude-domain repeatability, and bandwidth configurability. Experiments were executed on an Asus RT-AX86U router and an Asus GT-AXE11000 router. The latter was the only viable option to test the 2.4 GHz band. USRP results were only gathered using the RT-AX86U due to the lack of external SMA connectors on the other device.

To validate if a known baseband sequence can be transmitted and received without dropping samples, a deterministic single-tone amplitude-varying waveform was generated on the host and used as a reference signal. Figure 3 shows the received amplitude sequence when a BCMSDR device

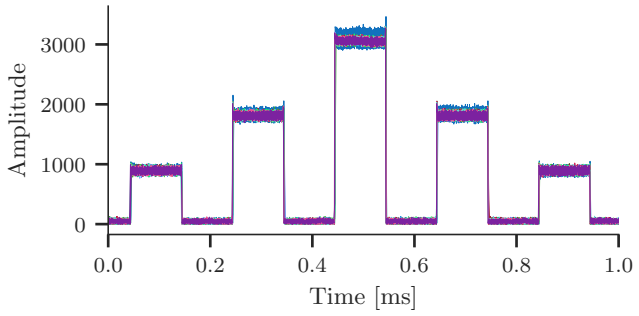


Figure 3: The figure shows overlaid chunks of 1 ms duration of a single-tone amplitude-varying signal that was repeatedly transmitted by a BCMSDR and received on a USRP over five seconds. Matching of the overlaid chunks indicates fine timing precision and lossless sample streaming at the transmitter.

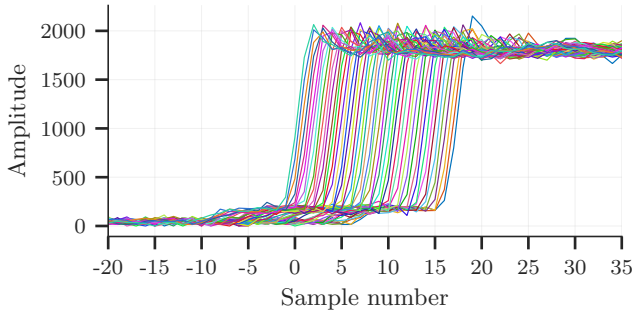


Figure 4: Zoom in on Figure 3 with focus on a single amplitude transition. There is a constant drift between each represented chunk, totaling in a cumulative drift of 15 samples over the span of a five second experiment duration.

transmits the prepared waveform and a USRP captures the resulting RF signal. The capture duration is approximately five seconds. For analysis, the acquired stream is divided into chunks of the reference signal's length of 1 ms. All chunks are then overlaid. The near-perfect masking of the overlaid chunks indicates that the transmitted stream preserves the expected sample ordering over the whole capture interval. No missing regions, duplicated blocks, or buffer-wrap discontinuities occur.

The same validation was performed in the opposite direction. In that configuration, the USRP transmits the waveform and the BCMSDR device captured the corresponding IQ samples. The resulting captures exhibited the same overlap behavior as in the transmit-side experiment, showing that transmit and receive paths both preserve the deterministic sample sequence under the tested conditions.

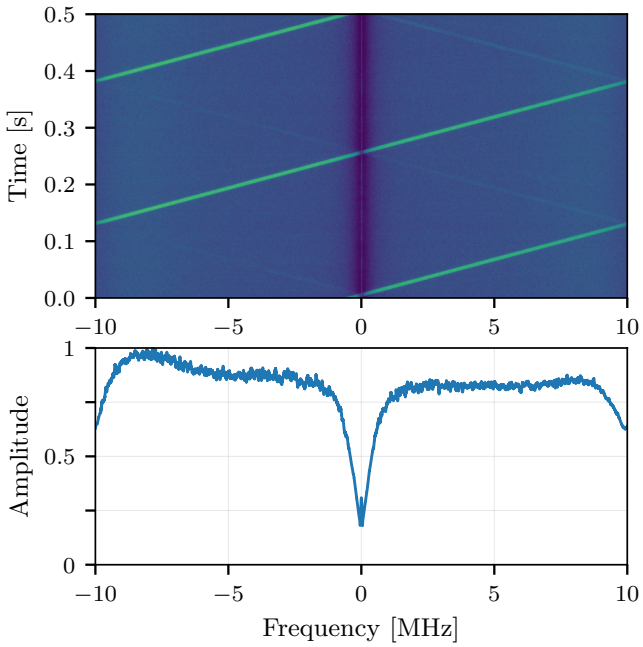


Figure 5: Spectrum of chirp received on a BCMSDR device. The upper plot shows the frequency content of the signal over time, with the chirp clearly visible. The lower plot shows the respective overall spectrum occupation, highlighting effects of receive chain filtering and DC removal.

A zoomed-in excerpt of the plot in Figure 3 is shown in Figure 4, revealing details of a single amplitude transition. The plot highlights the rising edge of the periodically repeated waveform across several chunks. While the transition shape is almost equal for all chunks, a small relative time displacement can be observed between consecutive rising edges. The figure shows a drift of about 15 samples between the first and last chunk. This corresponds to 150 ns of drift for each second of transmission, which can be attributed in the setup's sampling frequency offset (SFO), quantifiable at 150 ppb. The drift of the sampling clocks is natural because the devices were not clock-locked and the test setup did not feature a common external reference.

To evaluate the frequency-domain behavior of the reception path, a chirp waveform was transmitted with a USRP and received on a BCMSDR device. Figure 5 shows a spectrogram and an accumulated spectrum plot of the received samples. The spectrogram confirms that the chirp sweep is preserved over the expected bandwidth. The accumulated spectrum shows the influence of the configured receive filtering near the edges of the band pass. In particular, attenuation becomes visible close to the bandwidth limits, as expected from the receive filtering stages.

We also assessed the quality of the transmitted signal by capturing its spectrum. The signal followed the expected behavior but showed a small residual IQ imbalance. This impairment does not influence the functionality of the system, but it may affect specific applications. For this reason, dedicated host-side calibration routines may be required to compensate for hardware-dependent distortions and further improve output signal quality.

6 Conclusion

This work demonstrates that commercial Wi-Fi hardware can be successfully repurposed beyond its intended role as a standards-compliant network interface to serve as a practical platform for high-rate SDR experimentation.

By exposing a direct interface to the baseband and the RF front-end, the proposed platform provides the low-level hardware control required for advanced wireless research. This approach provides a compelling and highly scalable alternative to laboratory-grade hardware, particularly for experiments that require low-cost and scalable deployment.

The ability to reliably transmit and receive continuous raw baseband IQ samples at 20 MS/s enables experiments with non-Wi-Fi waveform generation, CTC, reactive transmissions, and rapid prototyping of custom PHY protocols on commodity hardware, providing flexibility where required.

One shot and looped operation modes at sampling rates up to 160 MS/s and bandwidths up to 160 MHz further enable high-precision experiments that even lower-end commercial SDRs can not provide, albeit limited.

Control over RF front-end and radio properties, such as transmit and receive power, filter properties, and target frequency allows experts to fine-tune their setups.

The UHD interface allows for integration of BCMSDR into a broad range of popular SDR applications and hides most of the architectural complexity from end-users, providing a simple-to-use yet powerful SDR experience.

Acknowledgments

This work was partially supported at CNIT by project 6G-INTENSE (Intent-driven NaTive AI architecturE supporting Compute-Network abstraction and Sensing at the Deep Edge), Horizon Europe SNS JU Grant Agreement N. 101139266; and by project PAISES-6G (Pioneering AI-enhanced SEcure 6G Services framework), Horizon Europe SNS JU Grant Agreement N. 101292896.

References

[1] 2026. RTL-SDR (RTL2832U). https://www.rtl-sdr.com/.

[2] Hannah Brunner, Rainer Hofmann, Markus Schuß, Jakob Link, Matthias Hollick, Carlo Alberto Boano, and Kay Römer. 2021. Leveraging Cross-Technology Broadcast Communication to build Gateway-Free Smart Homes. In *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. 1–9.

[3] Ettus Research. 2026. *USRP Hardware Driver (UHD) Software*. https://github.com/EttusResearch/uhd Accessed July 06, 2026.

[4] F5OEO. 2026. *rpitx*. https://github.com/F5OEO/rpitx

[5] FCC.report. 2026. FCC ID MSQ-RTAXI600: ASUSTeK Computer Inc. Wireless Router Filing. https://fcc.report/FCC-ID/MSQ-RTAXI600. Accessed May 27, 2026.

[6] FCC.report. 2026. FCC ID MSQ-RTAXJF00: ASUSTeK Computer Inc. Wireless-AXE11000 Tri-band Gigabit Router Filing. https://fcc.report/FCC-ID/MSQ-RTAXJF00. Accessed May 27, 2026.

[7] GNU Radio Project. 2026. *GNU Radio: The Free and Open Software Radio Ecosystem*. https://www.gnuradio.org Accessed July 06, 2026.

[8] Francesco Gringoli, Marco Cominelli, Alejandro Blanco, and Joerg Widmer. 2021. AX-CSI: Enabling CSI Extraction on Commercial 802.11ax Wi-Fi Platforms. In *Proceedings of the 15th ACM Workshop on Wireless Network Testbeds, Experimental Evaluation & CHaracterization*.

[9] Francesco Gringoli, Matthias Schulz, Jakob Link, and Matthias Hollick. 2019. Free Your CSI: A Channel State Information Extraction Platform For Modern Wi-Fi Chipsets. In *Proceedings of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*.

[10] Woojae Jeong, Jinhwan Jung, Yuanda Wang, Shuai Wang, Seokwon Yang, Qiben Yan, Yung Yi, and Song Min Kim. 2020. SDR receiver using commodity wifi via physical-layer signal reconstruction. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*.

[11] Jakob Link, Florentin Putz, and Matthias Hollick. 2026. Pair-Fi: Integrity Code Protected Secure Device Pairing via SDR-Enabled Wi-Fi Chips on Smartphones. In *Proceedings of the 19th ACM Conference on Security and Privacy in Wireless and Mobile Networks*.

[12] Pothosware. 2026. *SoapySDR: Vendor and Platform Neutral SDR Library*. https://github.com/pothosware/SoapySDR Accessed July 06, 2026.

[13] Sascha Rösler, Anatolij Zubow, and Falko Dressler. 2025. Mute the Immutable - Making WiFi-to-LoRa CTC Robust Against WiFi Selective Jamming. In *44th IEEE International Conference on Computer Communications*.

[14] Matthias Schulz, Francesco Gringoli, Daniel Steinmetzer, Michael Koch, and Matthias Hollick. 2017. Massive reactive smartphone-based jamming using arbitrary waveforms and adaptive power control. In *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*.

[15] Matthias Schulz, Jakob Link, Francesco Gringoli, and Matthias Hollick. 2018. Shadow Wi-Fi: Teaching Smartphones to Transmit Raw Signals and to Extract CSI to Implement Practical Covert Channels over Wi-Fi. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*.

[16] Matthias Schulz, Daniel Wegemer, and Matthias Hollick. 2018. The Nexmon firmware analysis and modification framework: Empowering researchers to enhance Wi-Fi devices. *Computer Communications* 129 (2018).

[17] stdw. 2026. *cm-sdr*. https://stdw.github.io/cm-sdr/

[18] Dan Xia, Xiaolong Zheng, Fu Yu, Liang Liu, and Huadong Ma. 2022. WiRa: Enabling CTC from WiFi to LoRa with IEEE 802.11ax. In *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*. IEEE.