X-Cast: Cross-Technology Broadcasts for Enhanced Spectrum Utilization in Low-Power Networks

Rainer Hofmann, Carlo Alberto Boano, and Kay Römer

Institute of Technical Informatics, Graz University of Technology, Austria

Email: rainer.hofmann@student.tugraz.at, cboano@tugraz.at, roemer@tugraz.at

***Abstract*—Cross-technology interference is becoming a major challenge for reliable wireless communication. For instance, the 2.4 GHz ISM band accommodates various technologies such as Wi-Fi, Bluetooth Low Energy (BLE), and IEEE 802.15.4, often leading to significant coexistence issues. Devices such as Wi-Fi routers dominate the shared spectrum due to their greater power and bandwidth, negatively impacting reliability, increasing latency, and raising power consumption of low-power networks based on BLE or IEEE 802.15.4. Existing coexistence schemes primarily focus on efficiently utilizing overlapping radio channels by adjusting transmissions rather than preventing interference by smartly (re-)allocating radio channels. This paper introduces X-Cast, a novel approach for cooperatively reallocating radio channels among heterogeneous wireless networks to utilize the frequency spectrum more efficiently, reduce interference, and foster coexistence. X-Cast employs cross-technology broadcasts to share information about the radio channels a network uses with nearby appliances. Based on the underlying wireless technology, X-Cast autonomously determines appropriate actions, such as switching to a different channel or blocklisting a specific channel. We seamlessly integrate X-Cast into the Zepyhr operating system and evaluate its performance using off-the-shelf Wi-Fi, BLE, and ZigBee devices. Our experiments demonstrate the effectiveness of X-Cast, showing up to 18% and 46% improvements in link-layer packet reception for BLE and ZigBee devices, respectively. Moreover, X-Cast reduces the maximum communication latency by up to 3x for BLE devices and up to 10x for ZigBee devices while lowering their power consumption by as much as 25%.**

***Index Terms*—Coexistence; Cross-technology communication; Cross-technology interference; Channel coordination; Wi-Fi; Bluetooth Low Energy; IEEE 802.15.4; ZigBee; X-Burst.**

I. Introduction

Due to the growing popularity of Internet of Things (IoT) applications, an increasing number of wireless devices is operating in the same industrial, scientific, and medical (ISM) bands, leading to severe cross-technology interference (CTI) that negatively affects communication performance. This is especially problematic in the unlicensed 2.4 GHz ISM band, where multiple technologies, including Wi-Fi, Bluetooth Low Energy (BLE), and IEEE 802.15.4, operate on the same frequencies. Figure 1 shows the radio channels of these three technologies and potential areas of interference. Consequently, standard-compliant devices must compete for medium access, turning the radio spectrum into an expensive resource [1], [2].

Traditional approaches are insufficient. To efficiently utilize the shared frequency spectrum, collision avoidance strategies such as TDMA [3] and CSMA [4] are often used. While the former specifies different timeslots for exchanging data, the latter uses energy sensing to determine if the shared medium is used and adjusts its transmissions accordingly. However, when

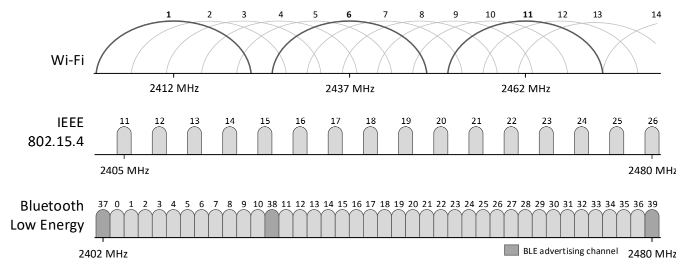


Fig. 1: Overlapping radio channels between Wi-Fi, IEEE 802.15.4, and BLE in the 2.4 GHz ISM band.

devices using different wireless technologies are co-located, *neither TDMA nor CSMA is suitable* to enable coexistence. TDMA schemes require a common communication standard, and hence cannot be used in the context of heterogeneous devices using incompatible physical layers (PHYs). CSMA approaches, instead, suffer from asymmetries in transmission power and channel bandwidth among devices employing different PHYs. For instance, because of such asymmetries, an IEEE 802.15.4 device may detect transmissions from a Wi-Fi device but not vice versa. Consequently, IEEE 802.15.4 devices can experience a reduction in packet reception rates by up to 85% due to CTI from nearby Wi-Fi traffic [5], [6].

In contrast, Bluetooth Low Energy (BLE) employs a technique called adaptive frequency hopping (AFH) to reduce the effects of CTI. This method uses multiple channels across the entire frequency spectrum according to a well-known hopping sequence for communication. Consequently, ongoing communications from Wi-Fi or IEEE 802.15.4 will only affect a portion of the BLE channels, reducing the impact on the overall reliability. However, studies have shown that even when using AFH, the reliability of BLE can drop by up to 70% at the link layer [7], [8]. Blocklisting poor-quality channels, i.e., excluding them from the hopping sequence, can enhance the packet reception ratio (PRR). However, *classifying and managing these channels* is not trivial, and no standardized classification techniques are established, often resulting in the absence of blocklisting approaches in off-the-shelf devices [9].

The need for cooperative spectrum allocation. To improve the coexistence of heterogeneous devices and enhance spectrum utilization, new coordination methods are necessary. Cross-technology communication (CTC) can help address this challenge by allowing direct communication between different technologies. Although numerous approaches for implementing CTC among various devices and technologies have been proposed over the past decade [10], only a few studies have explored its use for improving the coexistence of heterogeneous devices. For instance, ECC [11] utilizes CTC to allow Wi-Fi devices to notify IEEE 802.15.4 devices about available Wi-Fi white spaces. This method enables IEEE 802.15.4 devices to transmit messages without interference during these white spaces. Additionally, BiCord [12] extends ECC by incorporating bidirectional communication, which allows IEEE 802.15.4 devices to request these white spaces actively. However, previous studies have primarily concentrated on adjusting transmission timing for overlapping radio channels without considering the possibility of *explicitly reallocating channels* to prevent such adjustments altogether.

Keeping BLE in the loop. Previous research on coexistence strategies has mainly concentrated on Wi-Fi and IEEE 802.15.4 devices, overlooking BLE (often under the assumption that its AFH scheme gives sufficient reliability [8]). However, *CTI also significantly affects BLE*, often due to poor AFH configuration [13]. Providing BLE devices with information about channels with interfering communications would, thus, enhance their performance. This proactive approach would be more effective than traditional reactive methods, as channels could be excluded from the hopping sequence even before a certain number of transmission failures have occurred. This is particularly advantageous for applications with strict timing requirements. Additionally, this strategy would also *minimize the effects of BLE on Wi-Fi and IEEE 802.15.4* devices, promoting better coexistence among these technologies.

Our contributions. In this paper, we introduce X-Cast, a novel approach for cooperatively allocating radio channels to efficiently utilize the frequency spectrum among nearby heterogeneous devices, i.e., among devices using incompatible PHYs. In particular, X-Cast leverages *cross-technology broadcast messages* to enable *off-the-shelf devices* to simultaneously notify surrounding appliances about their presence and share detailed information about the radio channels they are using. Upon receiving such broadcasts, X-Cast autonomously takes appropriate actions, such as staying on the current radio channel, switching to a different channel, or blocklisting a channel, depending on the underlying wireless technology. To the best of our knowledge, *X-Cast is the first channel reallocation scheme* that autonomously adjusts radio channels among heterogeneous devices for improved spectrum utilization.

After reviewing related work in Sect. II, this paper makes the following contributions:

- We introduce X-Cast, a novel channel reallocation scheme for heterogeneous wireless devices, emphasizing its key design principles (Sect. III).
- We explain X-Cast's modular architecture and the primary purpose of each module (Sect. IV).
- We present and explain a proof-of-concept of X-Cast designed for BLE, ZigBee, and Wi-Fi devices, demonstrating its working principle (Sect. V).
- We describe how X-Cast is seamlessly integrated into the Zephyr OS and implemented on off-the-shelf devices using Wi-Fi, BLE, and IEEE 802.15.4 (Sect. VI).
- We evaluate X-Cast in real-world applications, showing improved reliability, reduced latency, and lower energy consumption with minimal memory increase (Sect. VII).

After discussing open challenges and offering an outlook on future work (Sect. VIII), we conclude the paper (Sect. IX).

II. Related Work

We analyze next related work on conventional coexistence approaches, cross-technology communication, and the use of CTC to enhance coexistence among heterogeneous devices.

Conventional coexistence approaches. To avoid cross-technology interference, IEEE 802.15.4 networks often employ energy sensing techniques like *CSMA/CA*. These methods help identifying when the channel is busy and delay transmissions accordingly [14], [15]. Another technique called *adaptive frequency hopping*, which is mandatory for BLE and is also used by IEEE 802.15.4 (TSCH) [16], utilizes multiple channels in a well-known hopping sequence to reduce the effects of CTI. Additionally, crowded or low-quality channels can be excluded from the hopping sequence to enhance performance further [8]. However, detecting nearby devices can be difficult, particularly for Wi-Fi, as it often fails to recognize ongoing IEEE 802.15.4 or BLE communications due to differences in bandwidth and transmission power, raising the need for direct communication.

Cross-technology communication. To enable devices to recognize each other and facilitate direct information exchange, cross-technology communication can be utilized. A substantial amount of research has already explored how CTC can be achieved among various technologies, with a particular emphasis on devices operating in the 2.4 GHz ISM band, such as those based on Wi-Fi, BLE, and IEEE 802.15.4 [10]. Early studies primarily focused on a general method known as *packet-level modulation*, which involves encoding data within transmission power [17], [18], beacon intervals [19], duration [20]–[23], or the gaps between legitimate data packets [24]. In contrast, more recent research has explored advanced concepts such as *PHY emulation* [25]–[27] and *cross-decoding* [28]–[30], which significantly enhance the data rates of CTC. Additionally, investigations have been conducted on the feasibility of CTC between devices operating beyond the 2.4 GHz ISM band [31], [32]. The latest approaches also employ neural networks to facilitate cross-technology communication [33].

Cross-technology coexistence. Currently, only a few works have explored the use of CTC to improve coexistence among heterogeneous devices. Yin et al. [11] presented a unidirectional approach in which Wi-Fi devices utilize CTC to notify ZigBee devices about available Wi-Fi white spaces[1]. Yu et al. [12] enhanced this method, allowing ZigBee devices to request these white spaces actively. Later approaches utilize bidirectional CTC between Wi-Fi and ZigBee devices to allocate white spaces more accurately [34] or to assign

[1]ZigBee devices can transmit without interference in these white spaces.

specific transmission times [35]. Existing work has mainly focused on Wi-Fi and ZigBee and adjusting transmissions; in contrast, X-Cast also includes BLE and emphasizes avoiding interference by cooperatively *reallocating* radio channels.

III. X-CAST: DESIGN RATIONALE

In this section, we explain the rationale behind the design of X-Cast and discuss the challenges related to cooperatively allocate radio channels in heterogeneous wireless networks.

Lightweight and decentralized approach. When designing X-Cast, we primarily aimed to keep it lightweight and to minimize the introduced overhead. For this reason, we have decided against a centralized approach where a single device would manage the allocation of radio channels for multiple heterogeneous networks. Such a method would quickly become complex, raising questions like: Who would be the managing device? Additionally, networks would need to request and agree on the use of specific radio channels, which further complicates the process in distributed settings. Another disadvantage of a centralized approach is that it would not scale well with an increasing number of surrounding networks, leading to bottlenecks and significantly increased traffic, as all network-related changes would need to be routed toward a central entity. Therefore, we designed X-Cast as a decentralized approach, meaning no managing device is required; instead, *each network manages its operations individually*.

The need for cross-technology broadcasts. To accomplish this, heterogeneous networks must be able to exchange information directly. Cross-technology communication enables this; however, to facilitate an efficient exchange of information without negatively impacting CTI by introducing excessive overhead – such as additional messages and distinct CTC schemes for each technology pair – it is crucial to *broadcast this information to nearby networks simultaneously*. While a few existing studies have focused on CTC broadcast transmissions, there has been no investigation into utilizing this feature to share information to cooperatively allocate the shared frequency spectrum efficiently [23], [36]. Consequently, X-Cast utilizes CTC broadcasts to *periodically share information* about channel usage across heterogeneous networks.

Not every device must broadcast. To prevent worsening CTI due to excessive broadcasting, not every device needs to broadcast. Depending on the network's (physical) scale, only a few devices should be broadcasting; ideally, as little as one device per network[2] to limit the overhead introduced by X-Cast. Furthermore, as transmitting data via CTC can be quite time-consuming, depending on the specific scheme used, it is essential to *keep the broadcast information to a minimum*.

Which information to share. As technologies interpret radio channels differently, e.g., the frequency of Wi-Fi channel 1 is different from that of BLE channel 1, each CTC broadcast must include information about the *technology type* and the specific *radio channels* being used. This information will enable the recipient to calculate the affected frequencies. Additionally, each CTC broadcast will contain a *network ID* that links the received broadcast to its corresponding network[3].

[2]This device does not necessarily have to be the ZigBee Coordinator or the BLE Central. Based on hardware capabilities and resource constraints, one should select devices that can communicate with the most nearby networks.

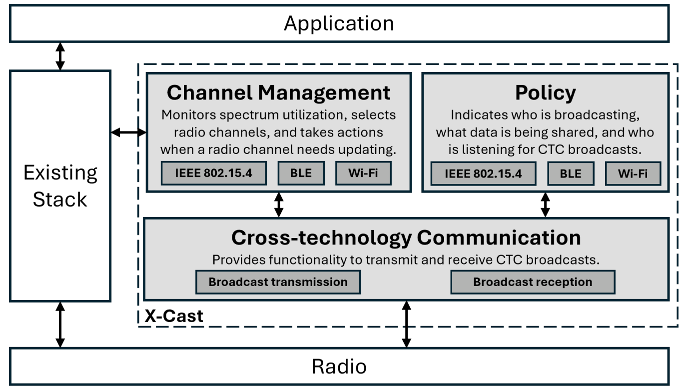


Fig. 2: X-Cast's modular architecture.

Identifying optimal channels. Nearby networks listen to these broadcast messages and adjust their behavior based on the information received. To allow X-Cast to allocate the most suitable radio channel, each network must have a comprehensive overview of the channel allocation across the entire frequency spectrum. Therefore, *X-Cast creates a network table* that includes an entry for every detected nearby network, detailing its type and the channels it is using. Furthermore, utilizing the network ID, X-Cast can determine whether an existing network has recently changed its radio channel or if a new network has appeared. Additionally, if a CTC broadcast from a network is not received within a specified time, the network is considered inactive, and its entry removed. Keeping the network table updated provides each network with an *overview of the radio channel utilization in its surroundings*.

Updating the network. Once CTI is detected, X-Cast will take appropriate actions, such as reallocating the network's radio channels. If the device running X-Cast is not responsible for updating network parameters it will send a request to the responsible network coordinator – such as the BLE Central, ZigBee Coordinator, or Wi-Fi access point – using its standard protocols. Upon receiving the request, the *coordinator will initiate the necessary procedures to update the network parameters* accordingly. For example, ZigBee has a feature called frequency agility that allows to reallocate channels at runtime. As multiple devices within a network can run X-Cast, the coordinator may receive multiple requests. In this case, the coordinator will adjust network settings based on all requests.

IV. X-CAST: ARCHITECTURE

To accommodate the flexibility of IoT networks, we designed X-Cast modularly, functioning alongside the existing protocol stack. Figure 2 sketches its overall architecture, which follows the design rationale outlined in Section III.

[3]In this work, we assume that one device belongs to only one network, but X-Cast does not prevent devices from being part of multiple networks.

A *policy module* defines how X-Cast behaves based on the underlying wireless technology. It determines whether CTC broadcasts should be transmitted, the information being broadcast, and whether the device should listen for incoming CTC broadcasts from nearby networks. For instance, the policy might state that BLE networks only listen for broadcasts.

A *channel management module* is responsible for selecting appropriate channels when radio frequencies are reallocated. It further monitors the frequency spectrum utilization and takes necessary actions if CTI is detected, depending on the underlying wireless technology. For example, suppose an IEEE 802.15.4 network identifies a nearby Wi-Fi network operating on the same frequency. In that case, the module will choose a new channel based on its objectives and initiate a procedure to update the network's channel.

A *cross-technology communication module* provides the CTC scheme for transmitting and receiving CTC broadcasts.

V. X-CAST: A PROOF-OF-CONCEPT

This section presents a proof-of-concept for X-Cast, outlining an example policy and channel management scheme for devices that utilize Wi-Fi, BLE, and ZigBee technologies[4].

A. Policy

Wi-Fi does not reallocate radio channels. Reallocating the radio channel during operation can have serious consequences for nearby devices that operate on the same frequencies. This concern is essential when the network reallocating channels has a high bandwidth, as it could interfere with multiple radio channels of other technologies. Thus, we do not reallocate radio channels on Wi-Fi networks, as this would affect at least 10 BLE or 4 IEEE 802.15.4 channels at once. It is also worth noting that Wi-Fi networks are typically managed in industrial or public settings and cannot switch the channel autonomously.

BLE only listens. Unlike Wi-Fi, BLE with its AFH operates across several narrow bandwidth channels. Broadcasting this information involves two considerations. First, it requires exchanging additional data compared to devices operating on a single channel. Second, and more importantly, if the BLE network uses all its channels within the 2.4 GHz ISM band, a nearby network cannot mitigate CTI as it cannot switch to a different non-overlapping channel. Consequently, BLE networks do not broadcast; instead, they only listen for CTC broadcasts and adjust their hopping sequence accordingly.

Keep cross-technology broadcast packets short. We let devices broadcast only the essential data required for X-Cast, as outlined in Sect. III. The technology type and the radio channel being used are represented using 1 byte: the two most significant bits indicate the technology (0 for Wi-Fi and 1 for IEEE 802.15.4), while the remaining 6 bits specify the actual channel number. For the network-ID, X-Cast utilizes the last two bytes of the network coordinator's MAC address, resulting in 3-bytes large cross-technology broadcast packets.

[4]Note that the goal of this work is to demonstrate the feasibility of explicitly reallocating channels using CTC broadcasts rather than designing a complex framework with all possible policies and channel management schemes.

B. Channel Management Scheme

Detecting overlapping CTI. Networks periodically broadcast their network type and the radio channel they are using. X-Cast uses this data, along with a lookup table, to identify whether a nearby network operates on overlapping frequencies.

Adjusting BLE's hopping sequence. In BLE, adding (allowlisting) or removing (blocklisting) specific radio channels from the hopping sequence is possible. We utilize these features to adjust BLE's hopping sequence as needed. However, blocklisting every channel with CTI may worsen the communication. To avoid this, X-Cast specifies a minimum number of channels that must remain active. Once this minimum has been reached, X-Cast will start categorizing channels based on the type of interference they experience, such as from Wi-Fi or IEEE 802.15.4 traffic. Given that Wi-Fi typically has a more significant impact, X-Cast prioritizes avoiding interference from nearby Wi-Fi devices when reaching the minimum number of radio channels left in the hopping sequence.

Reallocating IEEE 802.15.4 radio channels. Different actions are taken depending on the type of CTI detected, i.e., caused by Wi-Fi or IEEE 802.15.4 traffic. Since Wi-Fi has typically a more significant impact, the objective is to switch to a channel unaffected by Wi-Fi. If two IEEE 802.15.4 networks operate on the same channel and there are unused channels in the spectrum, the device with the lower network ID will switch to a different channel. Selecting an appropriate channel, however, will have an impact on nearby BLE networks. Thus, X-Cast categorizes the available channels into two groups to minimize the effect of channel reallocation on nearby BLE networks. The first group, which includes channels 12, 14, 16, 18, 20, 22, and 24, identifies *preferred channels*. These channels, in fact, overlap with only one BLE channel. The second group, which is composed of channels 11, 13, 17, 18, 21, 23, and 25, identifies *non-preferred channels*. These channels partially overlap with two BLE channels, requiring the BLE network to blocklist two channels instead of one.

To enhance connection-less BLE communication and BLE's device discovery, X-Cast avoids using IEEE 802.15.4 channels 15 and 26, as these overlap with BLE's advertising channels.

C. Working Principle

Following the policy and channel management scheme outlined above, we explain next the working principle of X-Cast using the example illustrated in Figure 3.

According to our policy, the Wi-Fi network only broadcasts periodically, the BLE network only listens, and the ZigBee network both listens to and sends CTC broadcasts. For simplicity, we assume that the Wi-Fi network was already operating in the surroundings, while the BLE and ZigBee networks are initiated simultaneously ❶. Initially, the Wi-Fi network sends a CTC broadcast ❷ indicating that it operates on Wi-Fi channel 6. Upon receiving this information, the BLE network begins blocklisting overlapping channels, specifically channels 11 through 21 ❸. Meanwhile, the ZigBee network detects a conflict since it operates on channel 16, and switches to

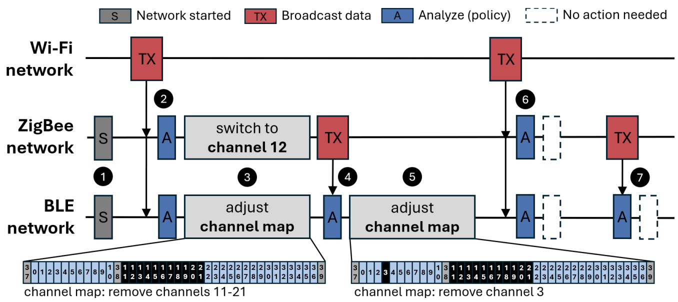


Fig. 3: Illustrative scenario showing the working principle of our proof-of-concept for X-Cast.

channel 12 ❸. Once the reallocation is completed, the ZigBee network informs nearby networks about the change ❹. The BLE network then updates its hopping sequence again by also blocklisting channel 3 ❺. After receiving the next CTC broadcast sent by the Wi-Fi network ❻, neither the ZigBee nor the BLE network needs to take any actions, as no conflicts are identified. The same holds true when the BLE network receives the next CTC broadcast from the ZigBee network ❼.

VI. Implementation

We seamlessly integrate X-Cast into Zephyr, an open-source, real-time operating system designed for resource-constrained embedded devices. Initially, we port X-Burst [23] – a packet-level CTC framework that encodes data into the duration of energy bursts and decodes information through high-frequency sampling of the received signal strength indicator (RSSI) – into Zephyr to enable data exchange among heterogeneous devices. Subsequently, we add support for Wi-Fi devices and ensure that CTC messages can be exchanged without impacting the regular operations of a device. Finally, we enhance X-Burst for efficient broadcasting, implement routines for adjusting the hopping sequence in BLE networks, and enable channel-switching in ZigBee networks.

Hardware platforms. We implement X-Cast on *state-of-the-art*, *off-the-shelf* IoT devices equipped with radios from Nordic Semiconductor. Specifically, we use the Nordic nRF52840 DK as BLE and ZigBee devices, and the Nordic nRF7002 DK as Wi-Fi device. It is important to note that X-Cast is not limited to these radios, as its fundamental principles (i.e., conducting CTC concurrently with a device's regular operations) have been demonstrated to be effective on radios from different manufacturers [37], [38].

Energy bursts. To generate the energy bursts required by X-Burst on ZigBee devices, we set the radio to *transmit mode*, where it emits an unmodulated radio signal. This approach differs from the original X-Burst method [23], which relies on actual data packets to create energy bursts, but allows for a more precise generation of energy bursts, improving the performance of CTC. For Wi-Fi devices, we utilize *Nordic's raw packet transmit* feature. This feature allows for transmitting of unmodified IEEE 802.11 packets at various data rates, facilitating the generation of energy bursts of different lengths.

RSSI sampling. To detect and decode energy bursts on ZigBee and BLE devices, we set the device's radio to *receive mode* and directly read the RSSI from the `RSSISAMPLE` register.

Periodic timeslots. To simultaneously perform CTC along a device's regular operations, X-Cast makes use of dedicated radio utilization timeslots [38]. During these timeslots, CTC can be carried out without impacting the device's usual radio activities. For duty-cycled devices, we leverage *Nordic's Multiprotocol Service Layer (MPSL)* work queue library, which supports multiprotocol implementations. This means that whenever the radio is in idle mode, it can be allocated a specific amount of time to perform CTC. This way, regular communication is not significantly impacted by X-Cast. For always-on devices, we have developed a routine that periodically requests radio time to perform CTC. Thus, a timeslot for using the radio is reserved in advance. This timeslot will only be used if there are no packets to transmit or receive; if there are packets, the reservation will be canceled.

Channel reallocations. In order to adjust the hopping sequence on BLE devices (i.e., adding or excluding channels), we use existing primitives within the Zephyr operating system. To switch the channel on ZigBee devices we use a custom routine based on IEEE 802.15.4 primitives[5]. Therefore, we enhance the existing IEEE 802.15.4 MAC commands by adding an extra command to *request a channel switch* and another one to *confirm the channel switch*. The managing device, such as the ZigBee Coordinator or ZigBee Router, sends a channel switch request to each network participant or child node. Following this, all devices switch to the new channel, sending a confirmation message to the managing device on the new channel, and they wait for the final confirmation message from the managing device. No data packets are transmitted during this waiting period to prevent data loss. Once the managing device has received a confirmation message from each network participant, it responds with a final confirmation message indicating that the channel switch was successful. If a confirmation message is not received from a specific network participant, the managing device returns to the original channel and repeats the switching routine with the missing devices.

VII. Evaluation

We evaluate X-Cast experimentally. First, we showcase the performance of a BLE and ZigBee network, both with and without X-Cast, by measuring each network's *packet-reception ratio*, *latency*, and *channel reallocation time*. Thereafter, we analyze X-Cast's memory footprint and conclude our evaluation by analyzing the power consumption of different devices operating with and without X-Cast.

Experimental setup. We set up two networks as depicted in Figure 4. We use two Nordic nRF52840 DK to establish a connection-based BLE network and three Nordic nRF52840 DK to establish a ZigBee network. The BLE network resembles a heart rate monitoring application, where the Peripheral

[5]ZigBee has an optional feature called *frequency agility* that allows to change the channel at runtime, but it is currently not supported in Zephyr.

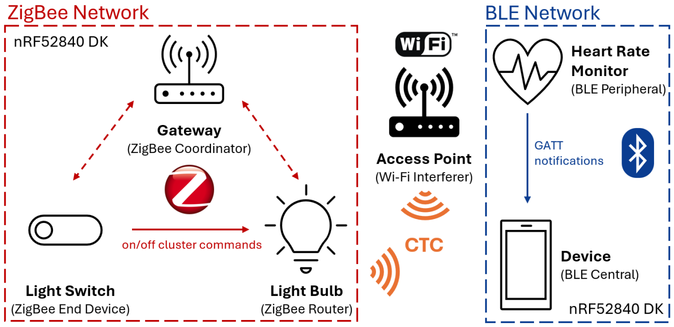


Fig. 4: Experimental setup used to evaluate X-Cast.

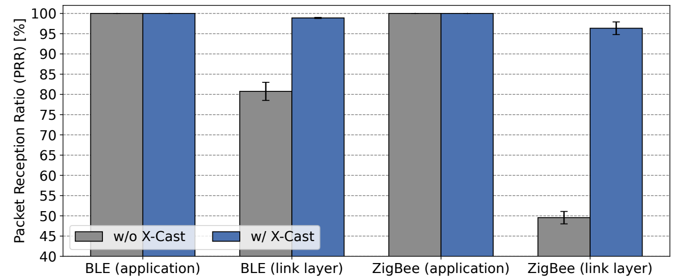
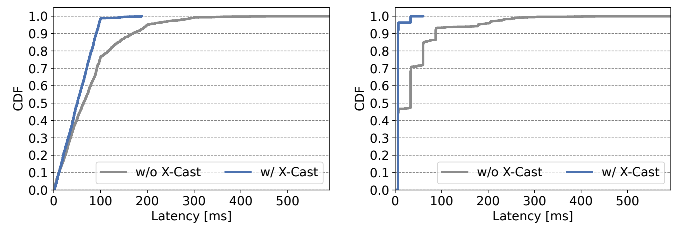


Fig. 5: Packet-reception ratio with and without X-Cast.



(a) BLE network (b) ZigBee network

Fig. 6: CDF of the latency with and without X-Cast.

device (representing the heart rate monitor) periodically transmits a simulated heart rate to the Central device (representing the smartphone) using the GATT heart rate service. The ZigBee network resembles a home automation application for controlling a light bulb, consisting of a ZigBee Coordinator (representing the gateway), a ZigBee Router (representing the light bulb), and a ZigBee End Device (representing the light switch). In addition, we utilize one Nordic nRF7002 DK for pseudo-randomly transmitting raw Wi-Fi packets with a transmission power of 20 dBm, mimicking the behavior of a Wi-Fi access point. Unless otherwise specified, all experiments are conducted in an office environment, with the devices placed at 1m distance and a transmission power setting of 0 dBm. During all evaluations, X-Cast operates alongside regular operations on one device per network: the BLE Central, the ZigBee Coordinator, and the Wi-Fi device. Devices exchange CTC broadcasts on the same frequencies: channel 13 for Wi-Fi, channel 24 for ZigBee, and channel 32 for BLE.

A. Network Performance

We begin by analyzing the performance of each network by measuring the PRR and latency, both with and without X-Cast, resembling the example shown in Figure 3. The BLE devices use the 1M PHY, with a connection interval of 100 ms, and exchange a new heart rate value (i.e., 2 bytes) every 500 ms across all 37 data channels. The ZigBee End Device transmits on/off cluster commands (resulting in IEEE 802.15.4 packets of 64 bytes) directly to the ZigBee Router on IEEE 802.15.4 channel 16 every 500 ms. Before transmitting IEEE 802.15.4 packets, a clear channel assessment (CCA) is performed to prevent collisions with ongoing transmissions. After 10 seconds, the Wi-Fi device starts transmitting 1500-bytes packets with pauses lasting between 1 to 5 ms, at a data rate of 6 Mbit/s on Wi-Fi channel 6, interfering with IEEE 802.15.4 channel 16. The evaluation lasts for 10 minutes and is repeated 5 times.

1) Packet Reception Ratio: To assess the PRR in the BLE network, we use *Nordic's quality of service report* to gain insights into the communication at the link layer. For the ZigBee network, we have enhanced the IEEE 802.15.4 driver by measuring and reporting information about the data exchange at the link layer. Figure 5 illustrates the PRR for each network at both the application and link layer levels.

As can be seen in Figure 5, the PRR at the application level is 100% for both networks. For the BLE network this is expected, as a message is retransmitted (at the next connection event) until it is received and acknowledged by the receiver. To make a fair comparison, we also had the ZigBee device transmit a message (using a backoff time of approx. 30 ms) until it was successfully received, resulting in a 100% PRR at the application level. At the link layer, the impact of X-Cast is evident. Under Wi-Fi interference, the average PRR of the BLE network drops to 80% (and even below 40% for specific channels). However, using X-Cast, the average PRR could be improved to almost 99% by avoiding interfered channels. The benefit of X-Cast is even more significant for the ZigBee network, with the average link-layer PRR rising from roughly 50% to over 95%. In particular, this could be achieved by letting the ZigBee network switch from channel 16 to channel 12, and letting the BLE network blocklist channel 3 (interfering with IEEE 802.15.4 channel 12) and channels 11–21 (interfering with Wi-Fi channel 6), as illustrated in Figure 3. It is important to remark that the benefit of X-Cast is significant here: although no data is lost at the application level, the loss of several link-layer packets may have huge implications on the responsiveness of the application, as packets may need to be re-transmitted several times, introducing large delays.

2) Latency: Next, we show the latency within each network when transmitting application data. When a message is sent, a timer starts on the transmitting device. Upon receiving the message, the recipient toggles a GPIO pin connected via cable to the transmitter, which stops the timer. Figure 6 presents the cumulative distribution function (CDF) of the latency, for each network. Table I provides the latency in more detail.

As can be seen in Figure 6 and Table I, X-Cast significantly improves the communication latency in both networks. While the average latency of the BLE network is not very expressive[6], the maximum latency becomes more important.

[6] In a perfect link scenario, the application latency will always range between 0 and the connection interval due to the periodic behavior of BLE.

Network	Mean	75%	90%	99%	Max
BLE	78 ms	75 ms	169 ms	296 ms	589 ms
BLE (w/ X-Cast)	51 ms	75 ms	91 ms	121 ms	188 ms
ZigBee	41 ms	60 ms	87 ms	260 ms	593 ms
ZigBee (w/ X-Cast)	7 ms	6 ms	6 ms	33 ms	60 ms

TABLE I: Application data latency for both networks.

Device	RAM / ROM (kB)		
	w/o X-Cast	w/ X-Cast	**X-Cast only**
BLE Central	41.9 / 163.8	44.5 / 168.7	**2.75 / 4.92**
ZigBee Coord.	73.3 / 321.2	73.8 / 324.9	**0.56 / 3.67**
Wi-Fi device	305.3 / 556.9	305.6 / 560.8	**0.28 / 4.00**

TABLE II: Memory footprint with and without X-Cast.

With X-Cast, the maximum latency decreases by a factor of 3, from 589 ms to 188 ms, and the maximum number of retransmissions is reduced from 5 to 1. The impact of X-Cast on latency for the ZigBee network is even more noticeable. With X-Cast, most messages could be sent almost instantaneously, while without X-Cast, due to the RF interference congesting the channel, it took up to 13 retransmissions to successfully transmit a message, reducing the maximum latency by a factor of almost 10, from 593 ms to 60 ms.

3) Channel Reallocation Time: The time required for a network to detect and reallocate its channels is heavily influenced by how quickly it can identify nearby networks. This, in turn, depends on the duration a network spends monitoring CTC broadcasts and the frequency at which nearby networks are broadcasting [38]. During our evaluation, the Wi-Fi device broadcasts at 1-second intervals, the ZigBee Coordinator broadcasts every 500 ms and scans for incoming broadcasts for 200 ms, whereas the BLE Central listens for 50 ms during each connection interval. As a result, the ZigBee network took less than 4 seconds to detect the Wi-Fi device and to reallocate the radio channel, whereas the BLE network required at most 16 seconds to detect both the ZigBee network and Wi-Fi device and to perform two channel map updates.

B. Memory Footprint

Next, we quantify X-Cast's memory footprint using the nRF Connect extension for Visual Studio Code. Table II presents the memory footprint of X-Cast in terms of RAM and ROM usage for the BLE Central, the ZigBee Coordinator, and the Wi-Fi device. The table also compares the memory footprint of the entire Zephyr application with and without X-Cast.

We can observe that the BLE Central has the highest memory usage: this is because X-Cast uses the MSPL to schedule CTC between regular BLE communications. The smaller memory usage on the Wi-Fi device is because it only needs to transmit CTC messages, does not need to keep track of surrounding devices, and does not implement a channel management functionality (e.g., blocklisting or switching the radio channel). Overall, with a memory footprint below 3 kB of RAM and 5 kB of ROM among all platforms, X-Cast is clearly well-suited for resource-constrained IoT devices.

C. Power consumption

We conclude our evaluation by analyzing the power consumption of the BLE Central, along with the energy-constrained devices in both networks: the BLE Peripheral and the ZigBee End Device[7]. Since ZigBee Coordinators and Routers are always on, X-Cast did not significantly affect their power consumption. To calculate the average power consumption of the devices, we utilize *Nordic's Power Profiler Kit II* and measure the average current consumption at a supply voltage of 3.3 volts. First, we establish a baseline by running all devices without X-Cast, and then repeat the measurements with X-Cast enabled to identify any differences in power consumption. Table III shows our experimental results.

Device	Average power consumption [mW]		
	w/o X-Cast	w/ X-Cast	Δ
BLE Central	21.45	33.06	**+ 54.12%**
BLE Peripheral	20.83	20.79	**- 00.19%**
ZigBee End Device	34.19	25.58	**- 25.18%**

TABLE III: Power consumption with and without X-Cast.

As expected, X-Cast increases the power consumption on the BLE Central, due to the additional radio activity introduced by sending and listening to CTC broadcasts. The BLE Peripheral, instead, does not exhibit a significant variation in its power consumption when running X-Cast: this is because it needs to wake up and exchange mandatory link layer packets in every connection event, regardless of whether application data must be transmitted[8]. In general, it is recommended to activate X-Cast on the BLE device having the least resource constraints (usually the BLE Central), so to minimize the impact of the additional overhead introduced by X-Cast. Table III shows that activating X-Cast is advantageous to the ZigBee End Device in our setup. This is the result of operating in a less congested channel, allowing the device to switch to low-power mode earlier and reducing its average power consumption by more than 25%. This proves the suitability of X-Cast in low-power wireless networks involving battery-operated devices.

VIII. Discussion and Future Work

In this section, we outline key open challenges and provide a preview of our plans to address them in future work.

Device selection. Our proof-of-concept focused on a scenario in which all heterogeneous devices can detect each other, which allowed each network to have a single device running X-Cast. When networks are spread across large areas, it is necessary that multiple devices run X-Cast. Developing methods to choose which devices run X-Cast and how to coordinate their activities will be investigated in future work.

Running out of channels. When a large amount of networks operates in the same area, it may happen that no channel is completely free from interference, resulting in the need for multiple networks to operate on the same frequencies. To cope with this, X-Cast could be enriched with a more advanced channel management scheme categorizing channels based on their utilization. For example, information about the amount of data generated could be integrated into CTC broadcasts, enabling other networks to pick the least-utilized channels.

[7]Those are typically battery-operated and should sustain a long battery life.

[8]During our evaluation, the Peripheral Latency was set to 0.

Networks without X-Cast. Some networks may not support X-Cast, which means that devices within those networks cannot communicate their presence to nearby appliances. However, it is still possible to detect these networks by either (i) analyzing any sudden drops in network performance or (ii) actively scanning the noise floor across the entire frequency spectrum by sampling the RSSI. BLE is a suitable option for the second method, as it provides good resolution in the 2.4 GHz ISM band by utilizing 40 channels. When detecting RF noise that cannot be matched to the activities of known nearby networks, a new source of CTI has been identified. This information can then be distributed to nearby devices by extending CTC broadcasts with a 5-byte channel bitmap. We will enhance X-Cast with this feature in future work.

IX. Conclusions

This work introduces X-Cast, a novel approach allowing to cooperatively reallocate radio channels among off-the-shelf devices with incompatible PHYs, and foster a more efficient utilization of the frequency spectrum. To this end, X-Cast employs cross-technology broadcasts to inform nearby devices about the presence of a network and its employed frequencies. We seamlessly integrate X-Cast into Zephyr and demonstrate its benefits for BLE and ZigBee networks experimentally. X-Cast allows to improve the communication reliability and latency of co-located low-power wireless networks while preserving the energy efficiency of resource-constrained devices.

References

[1] G. Zhou *et al.*, "Crowded Spectrum in Wireless Sensor Networks," *Proc. of IEEE Workshop on EmNets*, 2006.

[2] R. Natarajan *et al.*, "Analysis of Coexistence between IEEE 802.15.4, BLE and IEEE 802.11 in the 2.4 GHz ISM Band," in *Proc. of the Annual Conference of the IEEE Industrial Electronics Society (IECON)*, 2016.

[3] A. Sgora *et al.*, "A Survey of TDMA Scheduling Schemes in Wireless Multihop Networks," *ACM Computing Surveys*, 2015.

[4] L. Sanabria-Russo *et al.*, "A High Efficiency MAC Protocol for WLANs: Providing Fairness in Dense Scenarios," *IEEE/ACM Transactions on Networking*, vol. 25, 2017.

[5] C. A. Boano *et al.*, "JamLab: Augmenting Sensornet Testbeds with Realistic and Controlled Interference Generation," in *Proc. of the* 10^{th} *Conference on Information Processing in Sensor Networks (IPSN)*, 2011.

[6] A. Hithnawi *et al.*, "Understanding the Impact of Cross Technology Interference on IEEE 802.15.4," in *Proc. of the* 9^{th} *ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization (WiNTECH)*, 2014.

[7] B. Pang *et al.*, "Bluetooth Low Energy Reliability and Throughput under Wi-Fi Interference," in *Scientific Conference Electronics (ET)*, 2022.

[8] M. Spörk *et al.*, "Improving the Reliability of Bluetooth Low Energy Connections," in *Proc. of the International Conference on Embedded Wireless Systems and Networks (EWSN)*, 2020.

[9] B. Pang *et al.*, "Bluetooth Low Energy Interference Awareness Scheme and Improved Channel Selection Algorithm for Connection Robustness," *Sensors*, vol. 21, 2021.

[10] Y. He *et al.*, "Cross-Technology Communication for the Internet of Things: A Survey," *ACM Computing Surveys*, 2022.

[11] Z. Yin *et al.*, "Explicit Channel Coordination via Cross-Technology Communication," in *Proc. of the* 16^{th} *International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2018.

[12] Z. Yu *et al.*, "BiCord: Bidirectional Coordination among Coexisting Wireless Devices," in *Proc.s of the* 41^{st} *International Conference on Distributed Computing Systems (ICDCS)*, 2021.

[13] M. Spörk *et al.*, "Improving the Timeliness of Bluetooth Low Energy in Dynamic RF Environments," *Transactions on Internet of Things*, 2020.

[14] O. Ali *et al.*, "Adaptive Clear Channel Assessment (A-CCA): Energy Efficient Method to Improve Wireless Sensor Networks (WSNs) Operations," *Journal of Electronics and Communications*, vol. 131, 2021.

[15] T. Sparber *et al.*, "Mitigating Radio Interference in Large IoT Networks through Dynamic CCA Adjustment," *Open Journal of Internet Of Things (OJIOT)*, vol. 3, 2017.

[16] T. Watteyne *et al.*, "Reliability Through Frequency Diversity: Why Channel Hopping Makes Sense," in *Proc. of the* 6^{th} *ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN)*, 2009.

[17] Z. Chi *et al.*, "B2W2: N-way Concurrent Communication for IoT Devices," in *Proc. of the* 14^{th} *ACM International Conference on Embedded Network Sensor Systems (SenSys)*, 2015.

[18] X. Guo *et al.*, "Wizig: Cross-technology Energy Communication over a Noisy Channel," in *Proc. of the* 36^{th} *IEEE International Conference on Computer Communications (INFOCOM)*, 2017.

[19] S. M. Kim *et al.*, "FreeBee: Cross-Technology Communication via Free Side-Channel," in *Proc. of the* 21^{st} *International Conference on Mobile Computing and Networking (MobiCom)*, 2015.

[20] K. Chebrolu *et al.*, "Esense: Communication through Energy Sensing," in *Proc. of the* 15^{th} *International Conference on Mobile Computing and Networking (MobiCom)*, 2009.

[21] Y. Zhang *et al.*, "HoWiES: A Holistic Approach to ZigBee Assisted WiFi Energy Savings in Mobile Devices," in *Proc. of the* 32^{nd} *International Conference on Computer Communications (INFOCOM)*, 2013.

[22] S. Yin *et al.*, "Interconnecting WiFi Devices with IEEE 802.15.4 Devices without Using a Gateway," in *Proc. of the* 15^{th} *International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2015.

[23] R. Hofmann *et al.*, "X-Burst: Enabling Multi-Platform Cross-Technology Communication between Constrained IoT Devices," in *Proc. of the* 16^{th} *IEEE International Conference on Sensing, Communication and Networking (SECON)*, 2019.

[24] X. Zhang *et al.*, "Gap Sense: Lightweight Coordination of Heterogeneous Wireless Devices," in *Proc. of the* 32^{nd} *International Conference on Computer Communications (INFOCOM)*, 2013.

[25] Z. Li *et al.*, "WEBee: Physical-Layer Cross-Technology Communication via Emulation," in *Proc. of the* 23^{rd} *International Conference on Mobile Computing and Networking (MobiCom)*, 2017.

[26] W. Jiang *et al.*, "BlueBee: a 10,000x Faster Cross-Technology Communication via PHY Emulation," in *Proc. of the* 15^{th} *ACM International Conference on Embedded Network Sensor Systems (SenSys)*, 2017.

[27] S. Wang *et al.*, "Networking Support For Physical-Layer Cross-Technology Communication," in *Proc. of the* 26^{th} *IEEE International Conference on Network Protocols (ICNP)*, 2018.

[28] W. Jiang *et al.*, "Achieving Receiver-Side Cross-Technology Communication with Cross-Decoding," in *Proc. of the* 24^{th} *IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 2018.

[29] X. Guo *et al.*, "LEGO-Fi: Transmitter-Transparent CTC with Cross-Demapping," in *Proc. of the IEEE Conference on Computer Communications (InfoCom)*, 2019.

[30] Y. Chen *et al.*, "Reliable physical-layer cross-technology communication with emulation error correction," *IEEE/ACM Trans. on Net.*, 2020.

[31] P. Gawłowicz *et al.*, "Enabling Cross-technology Communication between LTE Unlicensed and WiFi," in *Proc. of the* 37^{th} *IEEE International Conference on Computer Communications (INFOCOM)*, 2018.

[32] D. Xia *et al.*, "WiRa: Enabling Cross-Technology Communication from WiFi to LoRa with IEEE 802.11ax," in *Proc of the IEEE Conference on Computer Communications (INFOCOM)*, 2022.

[33] H. Wang *et al.*, "Physical layer cross-technology communication via explainable neural networks," *Transactions on Mobile Computing*, 2024.

[34] S. Kim, "Enabling WLAN and WPAN Coexistence via Cross-Technology Communication," *Sensors*, vol. 22, 2022.

[35] D. Gao *et al.*, "Spectrum Efficient Communication for Heterogeneous IoT Networks," vol. 9, 2022.

[36] H. Brunner *et al.*, "Leveraging Cross-Technology Broadcast Communication to build Gateway-Free Smart Homes," in *Proc. of the* 17^{th} *Conf. on Distributed Computing in Sensor Systems (DCOSS)*, 2021.

[37] ——, "Cross-Technology Broadcast Communication between Off-The-Shelf Wi-Fi, BLE, and IEEE 802.15.4 Devices," in *Proc. of the* 17^{th} *International Conference on Embedded Wireless Systems and Networks (EWSN), Demo Session*, 2020.

[38] R. Hofmann *et al.*, "SERVOUS: Cross-Technology Neighbour Discovery and Rendezvous for Low-Power Wireless Devices," in *Proc. of the* 18^{th} *Conference on Embedded Wireless Systems and Networks (EWSN)*, 2021.