C-Morse: Cross-technology Communication with Transparent Morse Coding

Zhimeng Yin[1] Wenchao Jiang[1] Song Min Kim[2] Tian He[1]

[1]Department of Computer Science and Engineering, University of Minnesota, U.S.
[2]Department of Computer Science, George Mason University, U.S.
{yinxx283, jiang832}@umn.edu, song@gmu.edu, tianhe@cs.umn.edu

***Abstract*—Recent research on CTC (cross-technology communication) demonstrates the viability of direct coordination among heterogeneous devices (e.g., WiFi and ZigBee) with incompatible physical layers. Although encouraging, current solutions suffer from either severe inefficiency in channel utilization or low throughput using limited beacons. To address these limitations, this paper presents C-Morse, which leverages all traffic (such as through data packets, beacons and other control frames) to achieve a high cross-technology communication throughput. The key idea of C-Morse is to slightly perturb the transmission timing of existing WiFi packets to construct recognizable radio energy patterns without introducing noticeable delays to upper layers. At the receiver side, ZigBee captures such patterns by sensing the RSSI value, and then decodes the transmitted symbols. C-Morse also introduces a novel timing-based multiplexing technique to allow the coexistence of multiple C-Morse access points and reject other interference, showing a reliable symbol delivery ratio. As a result, C-Morse achieves a free side-channel, whose CTC throughput is as much as 9× of the present state of the art, while maintaining the through traffic within a negligible delay that goes unnoticed by applications and end-users.**

I. INTRODUCTION

The rapid development of wireless technologies over the last decade has added great convenience to our lives. To accommodate different application requirements for wireless performance (e.g., range, throughput, reliability, timeliness, and energy), a wide range of wireless technologies have been proposed. Many of these technologies, such as Wi-Fi and ZigBee, operate in the public spectrum (e.g., ISM bands), where devices must compete with each other to access spectrum resources. The uncoordinated competition among incompatible wireless technologies leads to considerable spectrum inefficiencies and low reliability [7].

To address this issue, CTC (cross-technology communication) has been recently introduced to establish direct communication channels among different wireless technologies. In a nutshell, CTC is built upon the key idea that wireless devices working in the public spectrum (e.g., ISM bands) require sensing the existence of signals via channel energy detection (e.g., RSSI). This channel energy detection is both available and mandatory in current popular technologies such as WiFi and ZigBee, despite of their inability to decode packets due to different and incompatible physical layers. In other words, CTC is achieved by generating unique and universally detectable energy patterns across technologies.

Although the coexistence of heterogeneous devices is inevitable given the increasing popularity of wireless technologies, the research on CTC (cross-technology communication) is still limited and insufficient. There are a few pioneering designs [5], [23], [12] exploring the possibility of CTC, which either suffer from severe inefficiency in channel utilization [5], [23] by incurring disruptive traffic, or have constrained and low side-channel throughput by utilizing only beacon packets [12].

In this paper, we propose C-Morse, which modulates the transmission timing of all packets (e.g., through data traffic and control frames) to achieve the full potential of CTC. By constructing special patterns via changing the transmission timing of through packets, our design is free from additional overhead. Admittedly the timing-based modulation is nothing new. However, the new and challenging task is to perturb the timing of through data traffic and/or control frames in a non-intrusive and standard-compatible manner, so that the introduced method does not affect mechanisms in the upper-layers, such as TCP timeout in the transport layer and video streaming jitters in the application layer. Our design minimizes the impact upon network application with a bounded delay. The practicality of C-Morse is further improved via a novel timing-based multiplexing technique that enables concurrent C-Morse communication and noise/interference rejection. In summary, this paper offers the following key contributions:

- To the best of our knowledge, C-Morse is the first work that leverages *all existing* traffic (such as through data packets and control frames) to achieve the cross-technology communication between heterogeneous devices.
- C-Morse is transparent by only perturbing the transmission of existing packets within a bounded amount, without requiring hardware modification or dedicated traffic. In addition, C-Morse's novel timing-based multiplexing enables the noise rejection, concurrent transmission, and adaptation to various kinds of traffic simultaneously.
- The implementation on both ZigBee and WiFi platforms demonstrates C-Morse's practicality and reliability in the real environment. Results suggest that the CTC throughput of C-Morse is as much as 9× of FreeBee (asynchronous version) [12], without introducing dedicated data traffic, while being transparent to legacy networks.

978-1-5090-5336-0/17/$31.00 ©2017 IEEE

The paper is organized as follows. Section II presents the motivation of this paper. C-Morse design and advanced features are introduced in Sections III and IV. Section V analyzes the theoretical performance of C-Morse, while section VI evaluates its performance empirically via a physical testbed. A summary of related work is introduced in section VII, followed by a conclusion in Section VIII.

II. Motivation

This section demonstrates various benefits of CTC, followed by the experimental insight into the opportunity of achieving CTC, under a typical everyday environment.

A. CTC Benefits

• **Cost reduction for smart homes**: CTC establishes direct Device-to-Device (D2D) link between heterogeneous devices, without dual-radio gateways which are costly (>100 USD [3]) and rarely available in practice. Further, CTC reduces home maintenance cost through context-aware services. For example, the home WiFi AP collects rain forecast from the Internet, which is fed directly to ZigBee-enabled smart sprinklers to adaptively control the water consumption.

• **Energy savings in mobile devices**: The power-hungry WiFi network interface card (NIC) is one of the main causes of power consumption in portable devices, where significant amount of energy is drained by continuously searching for accessible WiFi APs. This issue can be addressed by letting WiFi APs to broadcast their SSID via CTC, and adopting a low-power radio (e.g., ZigBee) in mobile devices to capture the CTC symbols (i.e., SSID). This allows significant energy savings by turning on the WiFi NIC only when detecting APs of interest.

• **Spectrum efficiency under unlicensed bands**: Unlicensed bands (e.g, 2.4GHz ISM) today are overly-crowded with heterogeneous and incompatible wireless technologies. The severe competition over the spectrum has led to excessive cross-technology interference [8], [19], [21], which results in unfair and inefficient channel utilization. CTC provides a fundamental solution to this issue, by enabling direct channel access negotiation and allocation (e.g., TDMA and FDMA) among heterogeneities.

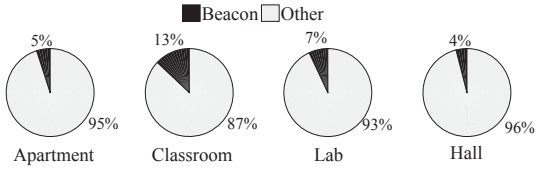


Fig. 1: The percentage of beacons in WiFi traffic

B. CTC Opportunities

Embedding CTC symbols on the mandatory beacons to carry CTC symbols has recently been introduced in FreeBee [12], which resolves the issue of spectrum inefficiency suffered by other state-of-the-art CTC techniques [5], [23]. However, we argue that CTC opportunity is significantly under-explored, leaving notable room for further improvement. That is, the full potential of CTC is reached by utilizing *all* existing traffic (unlike FreeBee which is confined to beacons).

We experimentally reveal the potential by collecting WiFi traces at four different settings (an apartment, a university hall, a research lab, a classroom) via Wireshark [2]. Figure 1 depicts the percentage of WiFi beacons at these four locations, where the beacon packets only account for less than 7% of the entire WiFi traffic on average. Therefore, the current technique based on beacons significantly underutilizes the opportunities for CTC, calling for a generic solution to fully utilize all the WiFi traffic.

III. Design

This section first presents design challenges and an overview of C-Morse, followed by a detailed description of its modulation and demodulation methods. For the sake of clarity and conciseness, we only introduce the communication design from WiFi to ZigBee, while the reverse direction has been implemented in a similar way.

A. Design Goals

Existing CTC solutions rely on either control beacons or dummy packets, therefore they do not need to concern about the impacts on the passing-through traffic. In contrast, a key design goal of C-Morse is transparency, where C-Morse should not noticeably introduce extra packet delay, decrease throughput, or trigger unnecessary TCP-timeouts.

To explore the relationship between the incurred delay and its impacts on the upper-layer performance, we have conducted experiments on the WARP 802.11 reference design [1], which is an FPGA based wireless research platform with open source for modification. Acting as a wireless AP, it receives packets from the Ethernet, and then transmits them to WiFi devices via the 802.11g standard. In each experiment, we provide WARP with either UDP or TCP at different rate, with increasing delay.

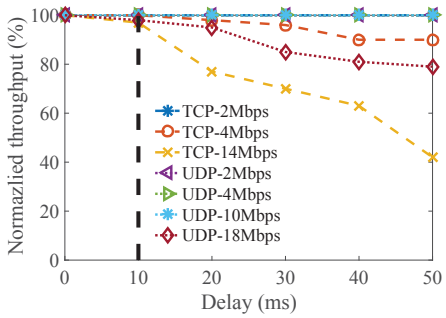


Fig. 2: The normalized throughput with delay

Fig. 2 studies the relationship between legacy throughput and the delay. It is clear that when the delay is small, such as 10ms marked with the dash line, the throughput degradations of both TCP and UDP are less than 4%. With a larger delay, the throughput losses of both UDP and TCP start to increase. When the delay reaches 30ms, the TCP suffers from more than 30% throughput loss, due to the congestion control based on dynamic network conditions. These findings suggest that C-Morse has to be careful when delaying the existing data packets for the purpose of transparency.

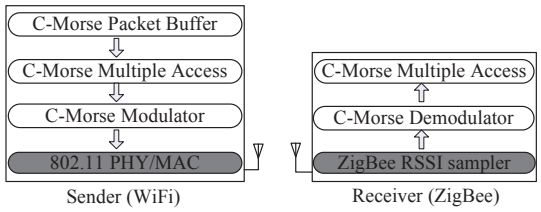


Fig. 3: Overview of C-Morse

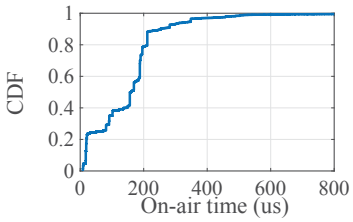


Fig. 4: The packet on-air time distribution

B. System Overview

Fig. 3 offers an overview of C-Morse's communication from the WiFi sender to the ZigBee receiver, while the other direction can be achieved in the same way. The shaded boxes represent the existing standards, while the black boxes are the components in C-Morse. Note that C-Morse is built upon existing standards, providing full compatibility with the existing billions of devices.

The WiFi sender modulates its symbols by only perturbing the transmission timing of through data packets and control frames within a negligible delay. In this way, C-Morse constructs the statistically recognizable energy patterns in the ISM band in a transparent way. On top of the modulator, C-Morse has its multiple access control, and a packet buffer to opportunistically utilize the existing data packets, avoiding the need of generating dummy packets.

Due to incompatible PHY layers, ZigBee radios cannot decode the WiFi packets directly but instead can utilize the RSSI sensing capability to sense the transmitted energy patterns. Such RSSI sensing is mandatory for many MAC standards including CSMA, and is provided in the widely used ZigBee and WiFi technologies. In addition, C-Morse's multiple access enables the ZigBee receivers to decode CTC symbols from several senders at the same time, which will be discussed in Section IV.

C. Modulation

This section describes the basic version of C-Morse's modulation techniques. For the sake of clear presentation, we here introduce this design under an ideal setting in which C-Morse has buffered enough packets for CTC modulation and there is no wireless interference. We will address these issues with advanced features in Section IV.

For the purpose of transparency, C-Morse (1) avoids the packet fragmentation, (2) does not rely on RTS/CTS injection, (3) does not change transmission rate, (4) and does not require synchronization between senders and receivers. As a result, C-Morse just postpones the existing data packets within a negligible delay.

To have a direct view of the existing WiFi traffic, we collected WiFi traces with the same setting mentioned in Section II. Fig. 4 demonstrates the CDF for on-air time, where more than 80% of the data packets have the on-air time less than 200us, because of the high transmission rates used today.

As WiFi traffic is highly dynamic, it is impossible for C-Morse to send existing data packets with fixed durations [5], [23] for constructing CTC. Instead, C-Morse borrows the idea of using a combination of long (dash) and short (dot) from International Morse Code, which can be constructed from the existing data packets by slightly perturbing their transmission timing. Note that the transmission timing of dots and dashes can be set differently under different wireless configurations (e.g., 802.11a/b/n/g). Without loss of generality, here a dot is defined as a WiFi energy pattern that lasts less than 3 ZigBee samples (384 us[1]); a dash is defined as a WiFi energy pattern with a duration longer than 4 ZigBee samples (512 us). There are two symbols, '1'and '0', which are represented by different and unique combinations of dashes and dots.

• **Dot**: To send a dot, C-Morse simply transmits one existing WiFi data packet from the head of the buffer. As confirmed in Fig 4, a data packet's duration is less than 384us with a high probability of 97%. We note that in practice, a data packet may have a small chance (1%) to have the on-air time will be longer than 512 us, which violates the definition of dot. Although we can reduce the air-time of a long packet by fragmenting it or reorder the packets in the buffer, these approaches are intrusive and unnecessary. Our simple design is based on the fact that each symbol is represented by a unique sequence of dots and dashes with redundant information. Therefore, demodulation can be largely successful with a small chance of error. In fact, C-Morse's advanced feature with timing-based correlation can further avoid such errors, which will be discussed later in Section IV.

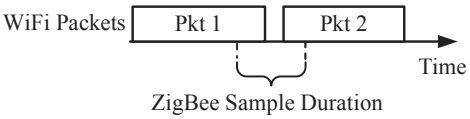


Fig. 5: Concatenating two packets.

• **Dash**: In contrast to a dot, a dash is too long to be represented by one single WiFi packet. Instead of lowering the transmission rate, C-Morse leverages the asymmetric sensing power between the ZigBee and WiFi, as depicted in Fig. 5. Each of the two packets is too short to represent a dash, so C-Morse transmits these two packets as close as possible while following 802.11 standard. As a result of the low RSSI sampling capability (128us for one ZigBee sample) [5], [23], ZigBee cannot detect the short idle periods between two data packets and can only sense one continuous dash instead of two dots. For example, in 802.11 standards, the short interframe

[1]Note: one Zigbee sample takes 128us

space (10us) [9] and the DCF interframe space (50us or 28us) [9] are too short to be observed by ZigBee radios.

More specifically, there are three ways to construct the dash, while following the current IEEE 802.11 standard.

1) Send out WiFi packets closely with a small interval [9] while following the DCF the standard. To reduce the impact caused by the random backoff, a ZigBee receiver utilizes a small window to detect the idle period between consecutive packets. If the period is small, the ZigBee receiver views them as one consecutive energy pattern.
2) Utilize the block ACK [9], where the successive data packets are separated by an interval of SIFS. By controlling the number of data packets in a block, C-Morse is able to construct a dash.
3) Leverage the A-MPDU [9] standard in 802.11, which enables aggregation of several data packets from the same source-destination pair into one longer data packet.

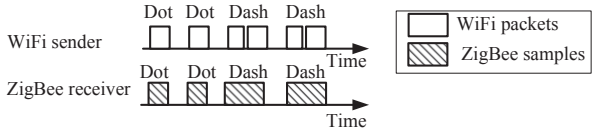


Fig. 6: Modulation and demodulation of C-Morse

Fig.6 depicts the process of transmitting symbol 1, which is represented by the Morse sequence {dot, dot, dash, dash}. C-Morse sender will first buffer data packets and then send the dot, dot, dash, and dash sequentially, where the dots and dashes are constructed by the method mentioned above.

D. Demodulation

To demodulate the CTC symbols, ZigBee collects the RSSI samples at a default sampling rate of 7.8KHz, and compares the collected RSSI values against the threshold (-85dBm). If the RSSI value is greater than the threshold, it is quantized to 1; otherwise, this RSSI value is quantized to 0. Fig. 6 depicts the corresponding received energy pattern at the ZigBee receiver, when a C-Morse AP transmits the Morse sequence {dot, dot, dash, dash}. With the quantized RSSI values, the ZigBee receiver knows that it has received the Morse sequence of {dot, dot, dash and dash} sequentially.

To compute the cross-correlation, the ideal sequence is denoted as $M = \{m_1, m_2, m_3, m_4, \ldots, m_L\}$, where $m_i = -1$ for a dot, and $m_i = 1$ for a dash. The received sequence is denoted as $\widetilde{M} = \{\widetilde{m}_1, \widetilde{m}_2, \widetilde{m}_3, \widetilde{m}_4, \ldots, \widetilde{m}_L\}$. The correlation as the similarity between the transmitted signal and the ideal case is computed in the following way:

$$correlation = \langle \widetilde{M}, M \rangle = \sum_{i=1}^{L} m_i \times \widetilde{m}_i \qquad (1)$$

In Fig. 6, the correlation between the received sequence and the sequence of symbol '1' is 4, while the correlation between the received sequence and the symbol '0' (denoted by {dot, dash, dash, dot}) is 0. After considering the correlation for symbol '0' and symbol '1' C-Morse's receiver recognizes that it has received symbol '1'.

IV. Advanced Features

For illustration purposes, Section III presents the C-Morse's basic modulation and demodulation in an ideal setting. Since the ISM band is crowded with heterogeneous devices, the wireless interference will affect the received energy patterns inevitably. In this section, we demonstrate how C-Morse's noise rejection method ensures its accuracy, while still following the existing 802.11 standard. In addition, we also introduce additional advanced features, such as concurrent transmission, and dynamic traffic adaptation.

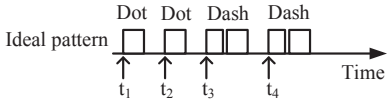


Fig. 7: Send out a CTC symbol at critical time points

A. Noise Rejection

To deal with the dynamic and unpredictable noise, C-Morse sends out WiFi packets at *critical time points*, instead of sending them out as quickly as possible. Fig. 7 demonstrates the way C-Morse sends the Morse sequence {dot, dot, dash,dash} at critical time sequence $T = \{t_1, t_2, t_3, t_4\}$ (note all t_i are relative). C-Morse first sends out a dot at time t_1, and then waits until time t_2 to send out another dot, while it repeats this process until completing its CTC symbol. Note that this procedure still follows the existing DCF standard for the CSMA purpose in 802.11, and we will talk about how to deal with the noises happening at these critical time points. By doing this, a ZigBee receiver is able to utilize the correlation at the critical time points for the denoise purpose.

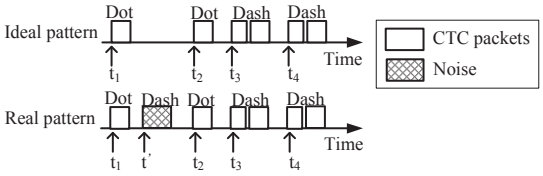


Fig. 8: Noise rejection using timing correlation

Fig. 8 depicts the way that C-Morse deals with random wireless noises. At time t' between t1 and t2, another AP sends out a packet that does not belong to the ongoing CTC symbol. Since $t' \neq t_i$, where $i = 1, 2, 3, 4$, C-Morse's receiver realizes that t' is not at the critical time point and hence can be ignored. After that, it will remove the dot at t' and recognize that it has received a dot at time t_1, a dot at t_2, a dash at t_3, and a dash at t_4. The correlation value is still 4, as in the ideal setting, even in the presence of wireless interruption.

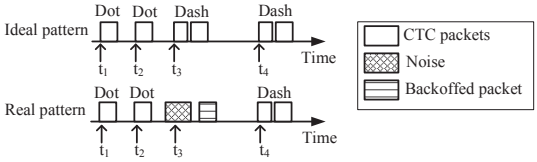


Fig. 9: Noises at critical time points

However, the interruption/noise can happen at any time, even at the critical time points. At time $t3$ in Fig. 9, for

example, C-Morse wants to send a dash by concatenating two data packets. Due to the existence of the noises, C-Morse needs to follow the DCF standard in 802.11, and delays its current transmission with the random backoff. To alleviate the impact of random noises, C-Morse utilizes several packets to construct one CTC symbol, so that the redundant information can be used for symbol restoration.

Similar to the basic design, the ideal and received Morse sequences are denoted as $M = \{m_1, m_2, \ldots, m_L\}$ and $\widetilde{M} = \{\widetilde{m}_1, \widetilde{m}_2, \ldots, \widetilde{m}_K\}$, respectively. Unlike the basic design, we introduce the ideal and received time sequences as $T = \{t_1, t_2, \ldots, t_L\}$, and $\widetilde{T} = \{\widetilde{t}_1, \widetilde{t}_2, \ldots, \widetilde{t}_K\}$, respectively. It is worth noting that the time sequences T and $\widetilde{T}$ are all relative sequences corresponding to their first timestamps t_1 and $\widetilde{t}_1$, and as a result C-Morse does not require the time synchronization between the sender and receiver.

The correlation function now is enhanced as follows: $correlation = \max_\tau\{\sum_{i=1}^{L} \delta(t_i + \tau)\}$, where

$$\delta(t_i + \tau) = \begin{cases} m_i \times \widetilde{m_j}, & \exists\, \widetilde{t}_j \in \widetilde{T} \text{ s.t. } t_i + \tau = \widetilde{t}_j \\ 0, & otherwise \end{cases} \quad (2)$$

The computed correlation measures the similarity between the ideal pattern and the received pattern, which is affected by the random wireless interferences. If the correlation is higher than a threshold, C-Morse's receivers treat the received signal as the transmitted CTC symbol.

B. Concurrent C-Morse

In the basic version, we assume that the time sequence $T = \{t_1, t_2, \ldots, t_L\}$ is the same for all C-Morse APs. By extending the idea of correlation, C-Morse supports concurrent transmission among multiple C-Morse APs. Specifically, C-Morse assigns different time sequences to different APs, so that the information from different C-Morse APs can be correctly differentiated.

Let us take the example of two APs operating at the same time. The first sender uses the timing sequence of $T^1 = \{t_1^1, t_2^1, \ldots, t_L^1\}$, while the second sender uses the time sequence of $T^2 = \{t_1^2, t_2^2, \ldots, t_L^2\}$. To ensure successful de-multiplexing, we choose two timing sequences T_1 and T_2 with a small time correlation. Formally, the correlation between timing sequences T_1 and T_2 can be calculated as $c = \max_\tau\{\sum_{i=1}^{L} \delta(t_i^1 + \tau)\}$, where

$$\delta(t_i^1 + \tau) = \begin{cases} 1, & \exists\, t_j^2 \in T^2 \text{ s.t. } t_i^1 + \tau = t_j^2 \\ 0, & otherwise \end{cases}$$

Fig. 10 depicts the concurrent transmission of two APs, where the first sender's time sequence is $T^1 = \{t_1^1, t_2^1, t_3^1, t_4^1\}$ and the second sender's time sequence is $T^2 = \{t_1^2, t_2^2, t_3^2, t_4^2\}$. Since the correlation between T^1 and T^2 is small, the receiver can use the time sequence T^1 to decode CTC symbols from the AP1, and vice versa. This is because only the packets sent at the critical time points can potentially lead to high correlation, while packets sent at non-critical time points will be considered as noises and then discarded. In this way, C-Morse achieves both multiplexing and the implicit addressing of multiple senders. We note that our time-based multiplexing creates orthogonal channels in the time domain, similar to the orthogonal CDMA channels in the code domain and FDMA in the frequency domain.

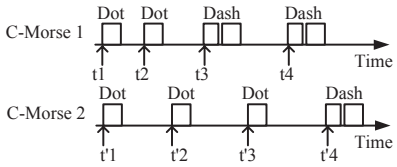


Fig. 10: Multiplexing in C-Morse

In reality, the orthogonal time sequences can be pre-computed and stored with the WiFi APs for avoiding computation overhead. One example could be that each AP chooses the interval between their two consecutive time points to be one unique co-prime number, or the time sequences can be set manually. We leave the auto generation of the critical time sequences for future work. To ensure that two different APs choose different time sequences, they listen before talk, and choose only unoccupied time sequences stored previously. In addition, the coordination can be accomplished via Ethernet to avoid the hidden terminal problems between the WiFi senders.

In addition, one C-Morse sender can boost its throughput by using multiple time sequences at the same time. In the previous discussion, one C-Morse sender has only one critical time sequence, and transmits one CTC symbol during its symbol duration. By using k different critical time sequences, C-Morse achieves a throughput gain of $k - 1$ times.

C. Traffic Adaptation

In Section III, we assume that C-Morse has buffered enough packets, and only transmits packets at the critical time points. In reality, however, WiFi through traffic is generated by network applications, while C-Morse cannot control neither the volume nor the rate of the through traffic. This leads to two possible scenarios: 1) excessive traffic, and 2) insufficient traffic.

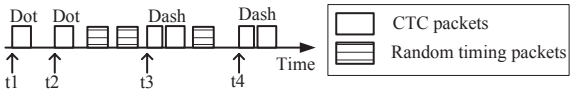


Fig. 11: Reduce the accumulated delay

Excessive traffic: Fig. 11 explains how C-Morse adapts to excessive packets. Instead of sending out the packets only at critical time points, C-Morse also sends out buffered packets at non-critical time points to reduce the delay of buffered packets. Since these packets do not occupy critical time points, they are considered as the random noises, which will be discarded by the specific time sequence in the demodulation process. In this way, C-Morse reduces the number of buffered packets and can be adapted to applications with intensive data traffic. In reality, the number of packets sent at non-critical time points is dynamic, which depends on the arrival rate of through traffic.

Insufficient traffic: On the other hand, it is also possible that C-Morse's AP might not have enough data packets to construct the CTC, since C-Morse is fully opportunistic and does not resort to generating dummy packets. To alleviate this issue, C-Morse utilizes the mandatory control frames, such as the beacons, to offer a low-bound bit rate. This can be done by using multiple time sequences within one C-Morse sender. For example, the C-Morse sender utilizes one critical time sequence for the data packets, while it adopts another critical time sequence for the control frames, such as beacons, where the gap between the successive critical time points can be as large as 100ms. By doing this, C-Morse maintains a lower-bound bit rate with mandatory control frames, when there is no passing through traffic. On top of that, C-Morse achieves a higher bit rate by utilizing passing-through data packets whenever available.

V. Analytics

In this section, we provide the theoretical analysis of the reliability and bit rate of C-Morse.

A. Reliability

To send out a CTC symbol, C-Morse sends out dots and dashes at the critical time points, while the noises will have the following impacts:

- The noise happens to occur at the critical time point, leading to an unsuccessful transmission of the dot or dash, at a probability of P_n.
- Right after the successful transmission of the dot, the random noises may concatenate after the dot, changing the received energy pattern to be dash, with a probability of P_n

Considering the impacts of noises, the successful reception probability of the dot is $P_{dot} = (1 - P_n) * (1 - P_n)$, while the probability is $P_{dash} = (1 - P_n)$ for a dash. As a result, the success reception of the dot or dash is denoted as $P_{rece} = (1 - P_n) * (1 - P_n)$, for a lower bound analysis of the symbol reliability. For a CTC symbol with length L to be successfully demodulated, more than $Thres$ dots and dashes need to be correctly received. As a result, the symbol reliability P_r is

$$P_r = \sum_{i=Thres}^{L} \binom{L}{i} (P_{rece})^i \times (1 - P_{rece})^{(L-i)}$$

Fig. 12 depicts the demodulation accuracy with different settings on the noise level and the symbol duration. It is clear that the demodulation accuracy decreases with an increasing noise ratio, which can be addressed by increasing the symbol duration.

B. Bit Rate

To be transparent, C-Morse delays the existing data packets within the delay bound Δ for the CTC opportunity. If C-Morse has buffered more than sufficient packets while satisfying the delay bound Δ, then it starts to transmit the CTC symbol. Reaching the delay bound, these packets are transmitted directly to ensure the transparency.

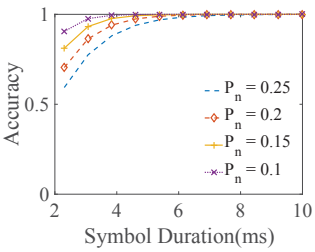


Fig. 12: Accuracy

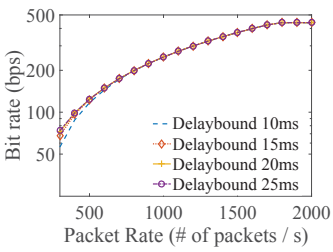


Fig. 13: Bit rate

Fig. 13 depicts the CTC bit rate with different settings at the log scale. At 300 packets/s, C-Morse benefits more from the traffic with the 25ms delay bound, since it can store unused data packets for future CTC opportunities. With the increasing packet rate, C-Morse manages to generate more bit rate, demonstrating C-Morse's capability of constructing a high CTC throughput with sufficient input traffic. With the gap between two critical time points to be 768us, the bit rate saturates to 430bps at the packet rate of 1800 packets/s.

VI. Evaluation

In this section, we evaluate C-Morse's performance across different aspects, including the CTC throughput, transparency, reliability, concurrency and the receiver cost.

TABLE I: Experiment settings

Communication Direction	Tx CH.	Tx power	Rx Ch.	Dist.
WiFi to ZigBee	1	12dBm	11-14	5m
ZigBee to WiFi	11-14	0dBm	1	3m

A. Experiment Settings

To evaluate C-Morse's performance, we have implemented C-Morse on the following platforms: (1) WARP with the open source 802.11 reference design [1], which follows the 802.11 standards. (2) commodity laptops, (3) and ZigBee compliant MICAz sensors. The implementation on commodity laptops is to verify C-Morse's compatibility with common off-the-shelf devices, while the following experiment results are carried on the WARP and MICAz sensors, following the settings in Table I. This is because the WARP platform enables fine-granularity control on the transmission time of the packets, when it follows the existing standards. On the contrary, the commodity NICs on laptops suffer from the intervention of other running applications.

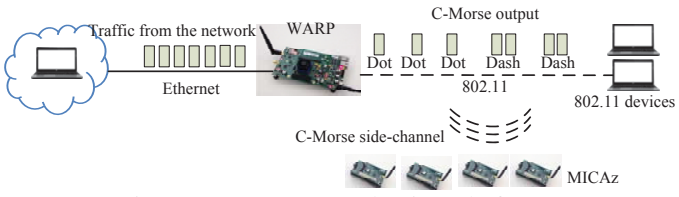


Fig. 14: C-Morse's evaluation platform

Fig. 14 depicts the C-Morse's conceptual evaluation platform, while Fig. 15a and Fig. 15b represent the evaluation settings. Connected to the data source using Ethernet, the

(a) Indoor settings (b) Outdoor settings

Fig. 15: Evaluation settings in C-Morse

WARP modulates the CTC symbols via transmitting these existing data packets to 802.11 devices. At the receiver side, the ZigBee-compliant MICAz sensors continuously sense the channel, and demodulate the transmitted symbols.

TABLE II: Throughput comparison

Direction	Throughput (bps)		
	C-Morse	FreeBee	Esense
WiFi to ZigBee	936	560	326
ZigBee to WiFi	215	146	NA

B. Throughput

This section presents the throughput comparison between C-Morse and previous methods.

1) Maximum experimental throughput: Following the setting in Fig. 14 and Table I, the WARP is provided with sufficient number of data packets. It sends out {dot,dot} with the 320us gap between two critical time points for representing symbol '1', while it transmits {dot,dash} with the 320us and then 704us gaps between two critical time points for representing symbol '0'. Following this setting, C-Morse has an average symbol duration of 832us, and achieves a maximum throughput of 936bps, as shown in Table II. Since the synchronization is an unrealistic assumption for heterogeneous devices, we compare C-Morse with the asynchronous FreeBee from now on. Compared with the 560bps in FreeBee (asynchronous version) [12] and 326bps in Esense [5], C-Morse's throughput is 1.6× of FreeBee and 2.8× of Esense, demonstrating its potential capability of future applications.

As for ZigBee sender, it sends out a Morse sequence of {dot, dot, dash} for symbol '1', and {dash, dot, dot} for symbol '0'. To boost the CTC throughput, it utilizes 3 different time sequences with an average duration of 12ms, following Section IV. In this way, C-Morse achieves a maximum throughput of 215bps for the ZigBee sender, a throughput gain of 47% compared with FreeBee. Note that the WiFi receiver utilizes a small time window for recognizing two ZigBee packets with a small idle period to be a dash.

2) Ideal achievable throughput: We demonstrate the effectiveness of C-Morse by its *theoretical* performance under the *ideal* conditions, following the *same* settings used in Esense and FreeBee: the RSSI sampling rate of 32KHz; there is no wireless noise; the inter-frame duration is at least 90us; and the maximum WiFi transmissions rate is 54Mbps.

As the *theoretical* scenario, a dot with the duration of 30.5us represents the symbol '0', while the dash of 61us represents the symbol '1'. Supplied with enough 100-byte packets, C-Morse utilizes two critical time sequences with inter-frame duration to be either 90us or 120us. With an ideal theoretical throughput of 13.2kbps, C-Morse offers a 2.6× throughput of FreeBee's 5.1Kbps (asynchronous version) and Esense's 5.13Kbps.

C. Transparency

This section evaluates C-Morse's transparency, given different volumes and kinds of data traffic.

• **Transparent throughput:** Following Fig. 14, we change the amount of through traffic per second for evaluating C-Morse's impacts on legacy networks.

The dark regions in Fig. 17b and Fig. 16b demonstrate the loss of legacy throughput because of C-Morse. When the original legacy throughput is small, such as 1Mbps, C-Morse only leads to less than 1% of the throughput loss, negligible for upper-layer mechanisms. With an increasing amount of through traffic, the legacy throughput loss starts to increase slightly, while C-Morse still manages to maintain the loss ratio under 7% for all the tested cases, and only results in a throughput loss of 5% for the 10Mbps legacy traffic, suggesting C-Morse's capability of transparency.

In addition to transparency, Fig. 16a and Fig. 17a provide the CTC throughput while maintaining transparency. With 100 packets/s, C-Morse achieves a throughput of 12bps, and manages to boost its CTC throughput to 137bps at 800 packets/s. For a fair comparison, the CTC sender avoids the generation of dedicated packets (e.g. beacons or data packets), and does not require synchronization. In the literature, only FreeBee (asynchronous version) meets these conditions with a 14.6bps throughput. The 9× throughput gain is achieved by utilizing all kinds of existing WiFi traffic, instead of the limited beacons. Note that the CTC throughput discrepancy between UDP and TCP is due to the different tolerable delay bounds (10ms for TCP and 15ms for UDP).

When there are no WiFi data packets, C-Morse utilizes the mandatory control frames, such as beacons, to provide a 3bps side-channel lower bound bit rate. Similar to the WiFi sender, the ZigBee sender constructs the CTC symbols via the existing ZigBee packets. With one ZigBee packet every 75ms [16], C-Morse's ZigBee sender achieves a 21bps throughput, with 8 critical time sequences on one ZigBee sender.

• **Transparency with traffic adaptation:** To reveal the impacts of C-Morse's traffic adaptation, we increase the through traffic rate to 18Mbps. After deploying C-Morse, the traffic adaptation (TA) module is turned off for the first five seconds, and is then switched on. Fig. 18 depicts the legacy throughput with time, where the C-Morse without traffic adaptation (C-Morse w/o TA) results in a throughput loss of 34%. However, adopting traffic adaptation limits the throughput loss to 6%, demonstrating C-Morse's transparency even under the condition of a very heavy through traffic.

• **Applications:** In addition, we evaluate C-Morse with the following applications.

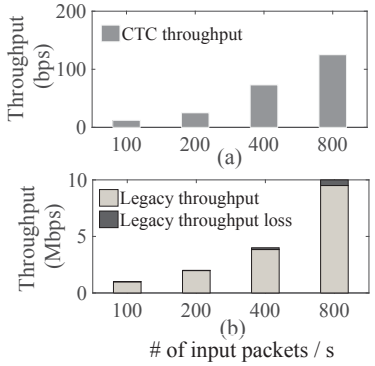


Fig. 16: Transparency with UDP

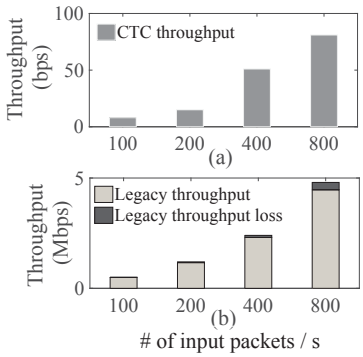


Fig. 17: Transparency with TCP

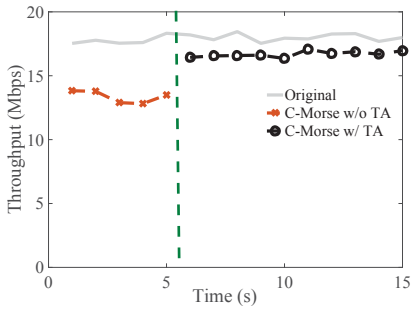
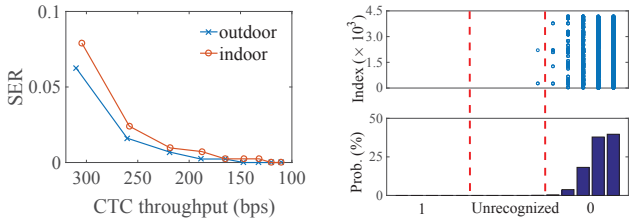


Fig. 18: Throughput in comparison with legacy networks

1) FTP file transfer. We measure the time needed to download files with different sizes from 10MBytes to 100MBytes. For all these tests, the normalized additional downloading time is bounded to 10%.
2) Audio streaming. We deploy an audio streaming server and a client. There are five volunteers participating in this experiment, while none of them have observed jitters or glitches when listening to the music.

All these results of real applications further demonstrate that C-Morse's transparency while it delays the existing data traffic to create CTC opportunities.



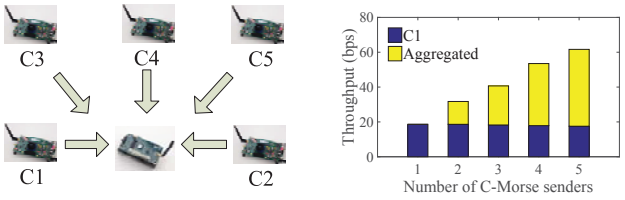
(a) SER with throughput (b) Constellation

Fig. 19: Symbol error rate.

D. Reliability

This section presents the reliability of C-Morse, by evaluating its symbol error rate (SER). Following the settings in Fig. 14 and Table I, the WARP is provided with sufficient data packets from the data source, while the gap between two critical time points is 756us. The WiFi sender (WARP) sends out the symbol '0' and symbol '1' repeatedly 4,300 times, when the ZigBee sensors continuously collect the RSSI samples and demodulate. To reveal the impacts of noise level, the experiment is carried multiple times at one outdoor place 15b and an indoor residential apartment 15a.

Fig. 19a provides the symbol error rate with different CTC throughput. With a longer duration, C-Morse achieves better symbol accuracy, and the SER reaches below 1% for both indoor and outdoor scenarios at a CTC throughput of 180bps. Note that the outdoor SER is lower, because of the lower wireless noises. To provide a detailed view, Fig. 19b illustrates the constellation and the corresponding probability of transmitted



(a) Experiment settings (b) Aggregated throughput

Fig. 20: Settings and aggregated throughput

symbol '0' 4,300 times, at a symbol duration of 5.3ms. It is clear that more than 99.7% of all the symbols reside in the right area. The false positive (FP) demonstrates the possibility of demodulating the noises as transmitted symbols. Since the noises are random, the ZigBee sensors can detect them with an accuracy of more than 99%.

E. Multiplexing

In this section, we evaluate the scenario of five concurrent C-Morse senders, as depicted in Fig. 20a. We turn on these five senders one by one to observe the interference between multiple ongoing C-Morse senders. As shown in Fig. 20b, the CTC throughput drops from 18.6bps with one C-Morse sender to 18.2bps when there are five concurrent senders. The small CTC throughput variation demonstrates the reliability of the timing-based multiplexing, where CTC symbols sent from different senders can be successfully distinguished.

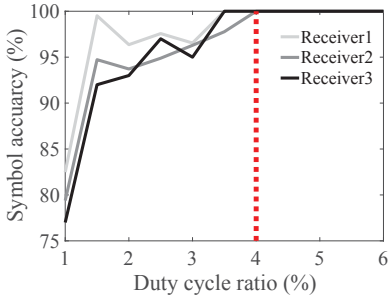


Fig. 21: Low-duty-cycle demodulation error with sampling rate

F. Receiver Cost

C-Morse is light-weight for commodity ZigBee devices, such as MiCAz and TelosB sensors, in terms of the storage cost, energy cost, and computation cost.

• **Storage cost**: Withe current CC2420 radios, the storage cost of quantized RSSI samples within one second is 7.8kbits, which only takes less than less than 0.3% of the available storage space provided by MICAz's on-board flashes.

• **Energy cost**: As the dominating factor of the energy cost [10] on ZigBee sensors, the ZigBee radio utilizes the low-duty-cycle to reduce the energy cost. Fig. 21 depicts the decoding accuracy of the control messages. At a duty cycle ratio of 4%, C-Morse's receivers can successfully detect the control messages, turn to the full sampling mode, and return to the low-duty-cycle to save the energy.

• **Computation cost**: To reduce the computation cost, C-Morse calculates the maximum possible correlation in a way similar to KMP [14] for avoiding unnecessary computations. By doing this, C-Morse's receivers can save 95% of the computation, when the channel is empty.

VII. Related Work

Many recent works, such as [15], [17], [24], [13], [18], have been proposed to improve the performance of cross-platforms. However, there is only limited work for the direct communication across heterogeneous systems. Although promising, recent studies such as [4], [11], [22] require dedicated hardware, resulting in incompatibility with existing billions of devices.

Another line of researches, such as Esense [5], Howies [23], Interconnecting [20] and FreeBee [12], focus on building the direct cross-technology communication (CTC), with existing hardwares and protocols. For example, Esense [5], Howies [23], and Interconnecting [20] modulate the CTC information by sending out additional and dedicated packets, and suffer from the inevitable low channel efficiency. As a remedy for this, FreeBee [12] modulates by shifting the positions of existing mandatory beacons, but suffer from the low throughput constrained by the number of available beacons. In addition, the recent work [6] builds a one-way communication from BLE to WiFi.

In contrast to previous works, C-Morse has the following features:(i) the first design that utilizes all existing WiFi traffic for constructing CTC, while maintaining transparency; (ii) a timing-based correlation for filtering out noises and concurrent communication.

VIII. Conclusion

In this paper, we design and implement C-Morse, a cross-technology communication (CTC) design that supports D2D communication among *heterogeneous* devices with different PHY layers. C-Morse modulates the timing of the packets passing through WiFi APs to construct recognizable radio energy patterns within negligible delay. Furthermore, C-Morse introduces an innovative timing-based multiplexing design that supports noise rejection, concurrent transmission, and delay-sensitive traffic adaptation simultaneously. Our extensive experiment results have demonstrated that C-Morse's reliability, transparency, and compatibility for low-cost wireless devices.

Acknowledgements: This work was supported in part by US NSF Grant CNS-1525235 and CNS-1444021. We sincerely thank the anonymous reviewers for their valuable feedback and suggestions.

References

[1] Wireless open-access research platform. http://warpproject.org/trac.
[2] Wireshark. https://www.wireshark.org/.
[3] Xbee. http://www.digi.com/products/xbee-rf-solutions/gateways/xbee-gateway.
[4] D. Bharadia, K. R. Joshi, M. Kotaru, and S. Katti. Backfi: High throughput wifi backscatter. In *Proceedings of ACM SIGCOMM 2015*.
[5] K. Chebrolu and A. Dhekne. Esense: communication through energy sensing. In *Proceedings of ACM MobiCom 2009*.
[6] Z. Chi, Y. Li, H. Sun, Y. Yao, Z. Lu, and T. Zhu. B2w2: N-way concurrent communication for iot devices. In *Proceedings of the ACM Sensys 2016*.
[7] R. G. Garroppo, L. Gazzarrini, S. Giordano, and L. Tavanti. Experimental assessment of the coexistence of wi-fi, zigbee, and bluetooth devices. In *Proceedings of IEEE WoWMoM 2011*.
[8] S. Gollakota, F. Adib, D. Katabi, and S. Seshan. Clearing the rf smog: making 802.11n robust to cross-technology interference. In *Proceedings of ACM SIGCOMM 2011*.
[9] W. L. W. Group. Ieee standard part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications. In *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, March 2012.
[10] S. Guo, L. He, Y. Gu, B. Jiang, and T. He. Opportunistic flooding in low-duty-cycle wireless sensor networks with unreliable links. *IEEE Transactions on Computers*, 63(11):2787–2802, 2014.
[11] B. Kellogg, A. Parks, S. Gollakota, J. R. Smith, and D. Wetherall. Wi-fi backscatter: internet connectivity for rf-powered devices. In *Proceedings of SIGCOMM 2014*.
[12] S. M. Kim and T. He. Freebee: Cross-technology communication via free side-channel. In *Proceedings of ACM MobiCom 2015*.
[13] S. M. Kim, S. Wang, and T. He. cetx: Incorporating spatiotemporal correlation for better wireless networking. In *Proceedings of ACM Sensys 2015*.
[14] D. E. Knuth, J. H. Morris, Jr, and V. R. Pratt. Fast pattern matching in strings. *SIAM journal on computing*, 6(2):323–350, 1977.
[15] W. Li, Y. Zhu, and T. He. Wibee: Building wifi radio map with zigbee sensor networks. In *Proceedings of IEEE INFOCOM 2012*.
[16] C.-J. M. Liang, N. B. Priyantha, J. Liu, and A. Terzis. Surviving wi-fi interference in low power zigbee networks. In *Proceedings of the ACM Sensys 2010*.
[17] R. Mahindra, H. Viswanathan, K. Sundaresan, M. Y. Arslan, and S. Rangarajan. A practical traffic management system for integrated lte-wifi networks. In *Proceedings of ACM MobiCom 2014*.
[18] S. Wang, S. M. Kim, Y. Liu, G. Tan, and T. He. Corlayer: A transparent link correlation layer for energy efficient broadcast. In *Proceedings of the ACM Mobicom 2013*.
[19] Y. Yan, P. Yang, X. Li, Y. Tao, L. Zhang, and L. You. Zimo: building cross-technology mimo to harmonize zigbee smog with wifi flash without intervention. In *Proceedings of ACM MobiCom 2013*.
[20] S. Yin, Q. Li, and O. Gnawali. Interconnecting wifi devices with ieee 802.15. 4 devices without using a gateway. In *Proceedings of IEEE Dcoss 2015*.
[21] X. Zhang and K. G. Shin. Enabling coexistence of heterogeneous wireless systems: case for zigbee and wifi. In *Proceedings of ACM MobiHoc 2011*.
[22] X. Zhang and K. G. Shin. Gap sense: Lightweight coordination of heterogeneous wireless devices. In *Proceedings of IEEE INFOCOM 2013*.
[23] Y. Zhang and Q. Li. Howies: A holistic approach to zigbee assisted wifi energy savings in mobile devices. In *Proceedings of IEEE INFOCOM 2013*.
[24] R. Zhou, Y. Xiong, G. Xing, L. Sun, and J. Ma. Zifi: wireless lan discovery via zigbee interference signatures. In *Proceedings of ACM MobiCom 2010*.