The Scalability of WiFi for Mobile Embedded Sensor Interfaces

Johnty Wang
Input Devices and Music Interaction Lab/CIRMMT
McGill University, Canada
johnty.wang@mail.mcgill.ca

Eduardo Meneses
Input Devices and Music Interaction Lab/CIRMMT
McGill University, Canada
eduardo.meneses@mcgill.ca

Marcelo M. Wanderley
Input Devices and Music Interaction Lab/CIRMMT
McGill University, Canada
marcelo.wanderley@mcgill.ca

ABSTRACT

In this work we test the performance of multiple ESP32 microcontrollers used as WiFi sensor interfaces in the context of real-time interactive systems. The number of devices from 1 to 13, and individual sending rates from 50 to 2300 Hz are tested to provide examples of various network load situations that may resemble a performance configuration. The overall end-to-end latency and bandwidth are measured as the basic performance metrics of interest. The results show that a maximum message rate of 2300 Hz is possible on a 2.4 GHz network for a single embedded device and decreases as the number of devices are added. During testing it was possible to have up to 7 wireless devices transmitting at 100 Hz to a wireless receiver via a router while attaining less than 10 ms of end-to-end latency. Performance however degrades with increasing sending rates and number of devices. Additionally, performance can also vary significantly from day to day depending on network usage in a crowded environment.

Author Keywords

WiFi, Sensor Interfaces, Latency

CCS Concepts

•**Hardware** → **Sensor devices and platforms;** •**Applied computing** → *Sound and music computing; Performing arts;*

1. INTRODUCTION

One of the key benefits of wireless sensor technology in the context of interactive systems and Digital Musical Instruments (DMIs) is that it is possible to create networked ensemble configurations consisting of multiple, untethered devices. Microcontrollers such as the ESP32[1] allow portable, wireless input controllers to be implemented in a cost effective manner via a variety of widely compatible protocols like MIDI over Bluetooth Low Energy or WiFi, OpenSoundControl (OSC) via WiFi that can interface with a variety of standard multimedia software and audio synthesis platforms. Coupling the low cost of such wireless interfaces with efficient battery and sensor packages, not only is it easier to build wireless interfaces, but it is also possible to consider implementing "smart", standalone sensors that can work independently as part of a larger network configuration[9]. However, in both cases the scaling in terms of number and overall transmission rate of devices become an issue.

In this paper we present a test configuration that attempts to provide a better understanding of the issue by looking at two variables: the impact of device message transmission rate and number of devices on the end-to-end latency of the system, such as when such a wireless device is used as a sensor interface for a DMI.

2. RELATED WORK

Most existing research surrounding latency of sensor interfaces measure the performance of a single device at a time. An end-to-end (signal in to audio out) latency measurement setup is described in [5] and results from a number of wired and wireless interfaces and synthesis systems are tested. Utilizing this test platform, similar tests were performed in a comparison focusing on different protocols implemented on the same hardware device [10]. In the context of selecting an embedded processing platform, the same testing methodology was yet again applied [6]. While all of these aforementioned tests involved only a single sender/receiver pair, they do provide a consistent testing platform that can be used to compare any new work and the open source nature of the test system supports easy reproduction and extension of the tests. The results show that it should be possible to achieve end-to-end latencies of less than 10ms for a WiFi device, a value which is often used as a standard in the community[11].

In terms of tests involving multiple devices, the overall bandwidth capacity of multiple 2.4 GHz WiFi devices was measured at over 4000 messages/second for up to 15 devices, with one-way latencies of a single device was as low as 3ms [7], but the effect of additional devices on latency was not measured. In another work, eight to ten devices were tested for a system implemented using low-power XBee radios [1] with results were in the order of tens of milliseconds, well beyond our threshold of interest for instrumental interactions.

3. TEST CONFIGURATION

For the present test, we build upon the hardware and software configuration used in the end-to-end latency tests [5]. A MacBook Pro (2.5 GHz i7, 16GB RAM) running OSX 10.14 was connected to a D-Link DIR-600 Router as the receiver. The audio synthesis Max/MSP patch used in [5] was modified to receive OSC messages and trigger the audio output as well as count the number of messages received by all senders on the network. A simple counter was also implemented in the synthesis patch to verify that the correct

[1] `https://www.espressif.com/en/products/hardware/esp32/overview`

NIME'20, July 21-25, 2020, Royal Birmingham Conservatoire, Birmingham City University, Birmingham, United Kingdom.

number of messages was received every second. The sending devices emit OSC messages at constant intervals with a 2 character name followed by a single integer, 0 except for one device under test that emits a 1 when it is triggered by the test jig. To make sure the test triggers were emitted as fast as possible and avoid the latency associated with the sampling rate itself, an extra message with the 1 (synth on) trigger was transmitted immediately. This eliminates the additional latency due to the sampling rate, which would be higher for lower sampling rates. For a single latency measurement, the test jig toggles a pin on one sender device which then transmits a 1 to the receiving synthesis patch on the computer. An audio output is then triggered, and the time difference logged by the test jig. Figure 1 shows the test setup, where the final end-to-end latency is calculated by the time difference between the trigger output and audio input.

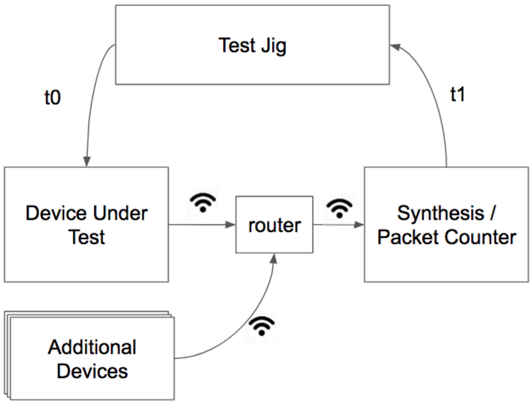

Figure 1: latency test setup.

We employed a wireless access point with no encryption (as suggested for higher performance [7]), and selected a channel that had the least amount of frequency usage measured by a MikroTik[2] router. This router was used for the sole purpose of network monitoring. The tests were performed in an office environment on a weekend where there is relatively little network traffic. A single test was also carried out during a work day to compare the difference in performance when more potential network congestion was encountered.

#	Frequency (MHz)	Usage	Noise Floor
0	2412	24.1	-103
1	2417	21.2	-104
2	2422	7.0	-104
3	2427	0.0	-104
4	2432	6.5	-105
5	2437	12.8	-101
6	2442	4.8	-103
7	2447	1.0	-102
8	2452	3.4	-103
9	2457	12.2	-106
10	2462	9.3	-105

Figure 2: Screen capture of the MikroTik's "Frequency Usage" tool, showing WiFi channel activity.

[2]www.mikrotik.com

4. RESULTS

In the first test, we employed a single sending device and altered the message transmission rate. Figure 3 shows the average end-to-end latency of 1000 trigger samples sent at 200 ms intervals. The results show that up to 1000 Hz, an average latency of below 10 ms was achievable. We reached a maximum rate of 2300 Hz on the ESP32 when running the processing loop without attempting to throttle the send rate.

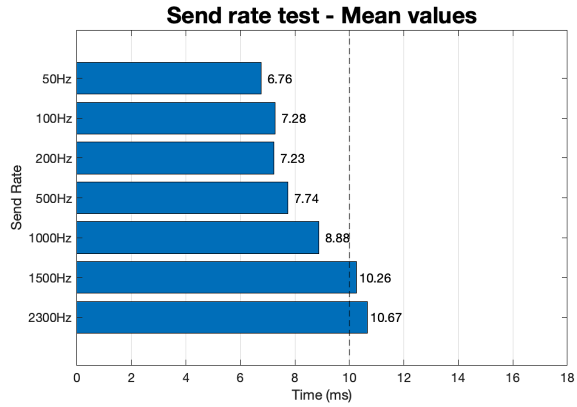

Figure 3: Latency test using a single device with different send rates

In the second test, we kept each device transmitting at 100 Hz, but incrementally added the number of devices.

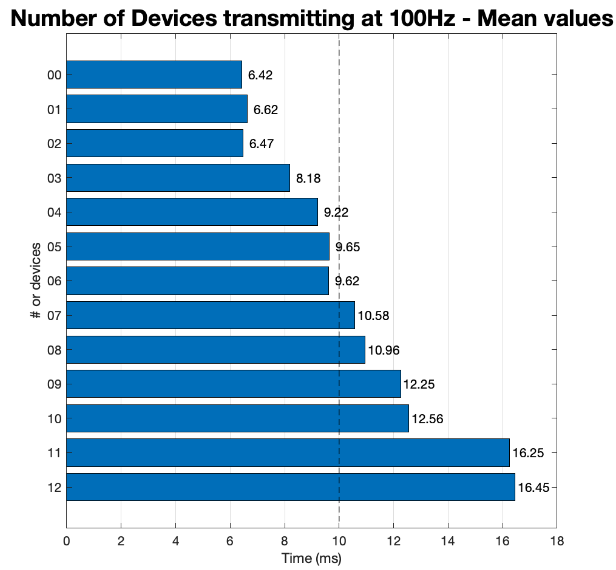

Figure 4: Latency test using multiple devices at a fixed 100 Hz send rate.

In both cases, latency values above 100 ms were removed as outliers. Even though this value is somewhat arbitrary, the decision is based on research that suggest that latency values above 70 ms between trigger action and sonic response can no longer be perceived as audiotactile simultaneous [4]. Discarding values above 100 ms guarantees removing outliers without removing a large amount of measurements. Table 1 shows the percentage of such outliers, and reveal that such values were present in most test cases.

Similar to previous work [5, 10], we found noticeable jitter in the latency measurements. Instead of looking at just the min/max bounds of latency or jitter, a more revealing

Table 1: Percentage of latency values above 100ms

Test	%	Test	%
1 @ 50 Hz	0.1	4 @ 100 Hz	0
1 @ 100 Hz	0.7	5 @ 100 Hz	0.6
1 @ 200 Hz	0.2	6 @ 100 Hz	0.6
1 @ 500 Hz	0	7 @ 100 Hz	0
1 @ 1000 Hz	0.8	8 @ 100 Hz	0.7
1 @ 1500 Hz	1.3	9 @ 100 Hz	0.7
1 @ 2300 Hz	0	10 @100 Hz	0.5
1 @ 100 Hz	0	11 @ 100 Hz	0.6
2 @ 100 Hz	0.6	12 @ 100 Hz	1.4
3 @ 100 Hz	0.5	13 @ 100 Hz	2

method is to look at the latency distribution via the Empirical Cumulative Density Function (ECDF) as employed by [6], which displays the number of cumulative latency values up to each point on the X-axis. Figures 5 and 6 show the ECDF for the single and multiple devices tests, respectively. Note that for clarity of presentation, intermediate number of devices in Figure 6 were omitted to reduce the number of lines drawn since they fell relatively evenly in between the previous and subsequent values.

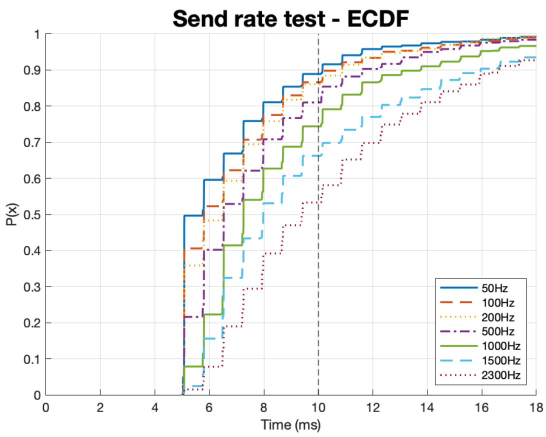

Figure 5: Empirical Cumulative Density Function (ECDF) of the send rate latency test.

By plotting a vertical line on the X-axis at a key point (such as 10 ms, for example), we can easily observe what portion of latency values falls below this threshold. The slope of the function shows how spread out the values are, and clearly shows larger jitter as the number of devices or send rate increases. Interestingly, the horizontal step size of the ECDF is quantized at 0.7 ms intervals, which can be best explained by the output audio vector size of 32 samples that was used (at 44 100 Hz sample rate).

Finally, when we conducted the single device 100 Hz test during working hours on a weekday, we found a significant difference in measured performance compared to previous tests. In this situation, we found the latency to be more than double the tests ran over a weekend. Figure 7 shows the difference in performance between less busy and more congested networks.

5. DISCUSSION AND FUTURE WORK

In this section we first discuss the consequence of the results, and then present some ways in which the tests can be improved. Although the scope of the test can be expanded in a number of ways, the results we have obtained is also

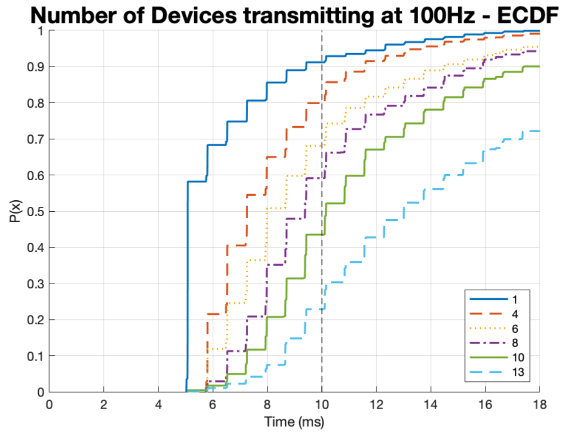

Figure 6: Empirical Cumulative Density Function (ECDF) of the multiple devices latency test.

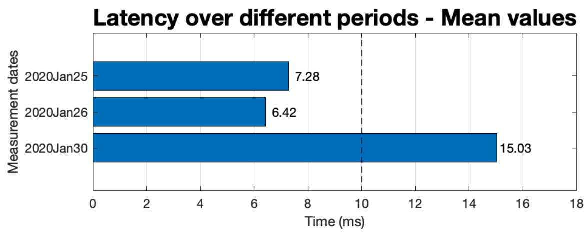

Figure 7: Mean values of the 100Hz single-device latency test on different days.

quite revealing in terms of clear issues and limitations posed by the utilization of WiFi.

5.1 Observed Outcomes

First, as expected, there is a clear pattern that latency increases as the number of devices and transmission rates increase on the network. Based on our results for the same total number of messages transmitted, the addition of devices add extra latency. For example, a single device transmitting at 500 Hz (Figure 3) appears to perform better than five devices at 100 Hz (Figure 4). This implies that fewer devices sending the same data at higher rates are preferable over more devices at lower rates. Consequently, for optimal performance it would be preferable to aggregate sensor inputs where possible.

Second, jitter is present in WiFi transmissions even for relatively ideal situations, and while it is possible to reduce jitter via the addition of latency [2], for the values we observe it would mean overall latencies of considerably greater than 10ms. The subsequent issue then, is to determine the acceptable amount of latency and jitter. From existing work it appears that such thresholds may be context dependent and also vary between individuals [4].

As a practical example, imagine a wireless device such as a the T-Stick [8] where real-time sensor data is being transmitted to a receiver for processing and synthesis. Our results show that, at least for our test configuration, that the 10 ms latency limit will be exceeded with less than ten devices are operating at 100 Hz. This suggests that careful planning of the connection topology, as well as extensive testing, should be performed when deploying systems involving wireless devices.

Finally, based on the presence of large latency values in most of the test cases, if strict timing requirements are to be met, it appears that WiFi may not be a reliable solution due to the potential for drastically degraded performance under congestion, which can be beyond the user's control depending on the environment.

5.2 Limitations and Improvements

As shown in Figure 8, the tests presented here span a relatively small portion of the potential space of variables. Additionally, while the main tests were performed under "ideal" situations (of an empty office on a weekend), there are a number of access points active in this environment which may affect performance slightly, even if it is not as obvious as the congested situation as measured during working hours. A better baseline measurement could be done in an RF isolated or absent location such as an anechoic chamber, or a rural location with little network activity, respectively. However, it should be noted that such measurements would only provide a "best-case" scenario, which may not be practical in actual application settings such as performance environments.

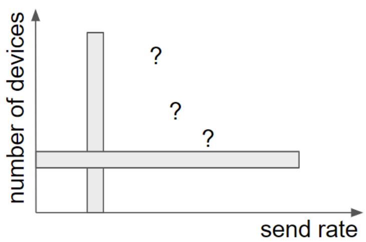

Figure 8: Test Coverage in terms of potential space of variables.

A more informed method of obtaining environmental data is through the use of wireless diagnostics tools. The features offered by the RouterOS found on relatively affordable MikroTik-based routers is an accessible way to gain a better understanding of the operating wireless environment, and can be used for planning decisions when deploying such systems. While we made cursory use of the frequency utilization measurements, a tighter coupling between the network traffic monitoring and latency measurements could provide a more accurate correlation between network congestion and recorded performance values.

The routing performance is potentially different between various router hardware as well as network configurations such as Ad-hoc/AP modes, encryption, etc. are thus potential variables of interest as well. Additionally, a particular system may consist of more than one receiver device, and/or bi-directional communication as well. One other important consideration is whether the receiving device is required to be wireless or not, since in the latter case there is effectively twice the potential bandwidth available.

Finally, embedded devices are starting to emerge with 5 GHz WiFi, which should provide much better performance due to the higher amount of bandwidth and larger number of channels available. The performance of 5 GHz systems should be tested as soon as they are available and may provide significant improvements.

6. CONCLUSION

In this work we documented the process of testing the scaling performance of a network of wireless embedded microcontrollers. By increasing the number of devices and send rates, we have established a first look at the general bounds of network performance when attempting to scale beyond a single device. The fact that WiFi performance can vary drastically from day to day due to external network load conditions is of concern in performance settings unless the environment can be well controlled. While WiFi, one of the most commonly used wireless interfaces have been demonstrated to operate well under certain situations, the results we have obtained, in conjunction with previous work in the literature, still suggest that wired connections are preferable for timing critical applications, of which DMIs most often fall under [3].

7. REFERENCES

[1] M. A. J. Baalman, V. De Belleval, J. Malloch, J. Thibodeau, C. Salter, and M. M. Wanderley. Sense/Stage-Low Cost, Open Source Wireless Sensor Infrastructure For Live Performance And Interactive, Real-Time Environments. In *Proc. Int. Computer Music Conf. (ICMC)*, 2010.

[2] E. Brandt and R. B. Dannenberg. Low-latency music software using off-the-shelf operating systems. In *Proc. Int. Computer Music Conf. (ICMC)*, 1998.

[3] P. Cook. Re-Designing Principles for Computer Music Controllers: A Case Study of SqueezeVox Maggie. In *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2009.

[4] R. H. Jack, A. Mehrabi, T. Stockman, and A. McPherson. Action-sound latency and the perceived quality of digital musical instruments. *Music Perception*, 36(1), 2018.

[5] A. McPherson, R. Jack, and G. Moro. Action-Sound Latency: Are Our Tools Fast Enough? In *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2016.

[6] E. Meneses, J. Wang, S. Freire, and M. M. Wanderley. A Comparison of Open-Source Linux Frameworks for an Augmented Musical Instrument Implementation. In *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2019.

[7] T. Mitchell, S. Madgwick, S. Rankine, G. Hilton, A. Freed, and A. Nix. Making the Most of Wi-Fi: Optimisations for Robust Wireless Live Music Performance. In *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2014.

[8] A. Nieva, J. Wang, J. Malloch, and M. M. Wanderley. The t-stick: Maintaining a 12 year-old digital musical instrument. In *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2018.

[9] L. Turchet, C. Fischione, G. Essl, D. Keller, and M. Barthet. Internet of musical things: Vision and challenges. *IEEE Access*, 6, 2018.

[10] J. Wang, A. Mulder, and M. M. Wanderley. Practical Considerations for MIDI over Bluetooth Low Energy as a Wireless Interface. In *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2019.

[11] D. Wessel and M. Wright. Problems and Prospects for Intimate Musical Control of Computers. *Computer Music journal*, 26(3), 2002.