

midOSC: a Gumstix-based MIDI-to-OSC converter

Sébastien Schiesser

Institute for Computer Music and Sound Technology

Zurich University of the Arts

Baslerstrasse 30, 8048 Zurich, Switzerland

sebastien.schiesser@zhdk.ch

Abstract

The Gumstix embedded computers provide interesting and modular platforms to develop embedded applications. Several features are already built-in in the base image of the operating system and a development framework allows to customize it for specific needs.

A MIDI-to-OSC converter has been developed with these products. It currently shows competitive timing characteristics for short MIDI messages. Furthermore, this tool will be developed to a more generic sensors-to-OSC converter and used for artistic purposes or quick interfacing of gesture acquisition devices.

Keywords: MIDI, Open Sound Control, converter, gumstix

1. Introduction

The Institute for Computer Music and Sound Technology (ICST) has been working for many years with Ambisonics: a set of recording and replay techniques for multichannel audio [6]. It has been active in the B-format encoding definition, wrote spatialization tools for Max/MSP [9], supports concerts and organizes residences for composers who want to use its Ambisonics facilities.

The ICST has developed a mobile Ambisonics equipment (mAe) for concert venues, which is able to play sound files or process live audio in a setup with up to 64 channels [2]. Due to the important size of this equipment and the fan noise caused by the processing computers, many devices are situated outside a concert hall and have to be remote-controlled via MIDI [7].

To avoid having an excessive amount of MIDI cables and overcome their limitations in length¹, control signals are converted into the Open Sound Control (OSC) format [11][12], sent to the remote-controlled devices location and converted back to MIDI. Until now, this has been done at each conversion point through a Max/MSP patch running

¹ The MIDI Manufacturers Association recommends a maximum length for DIN cables of 15 meter, while OSC works with twisted pairs ethernet cables, which can be up to 100 meter long.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

© 2009 Carnegie Mellon University

on a computer connected to a MIDI interface. This way of doing is very demanding in terms of hardware: in the backstage system of the mAe, one computer is dedicated to conversion purposes only. And if MIDI devices are present on stage, one supplementary laptop with interface is required.

The mAe is intended to be modular and to support several “I/O hubs”, where audio and data are collected and dispatched. In order to avoid to depend on a converting computer at each hub, it seemed interesting to use a dedicated converter, which can run in standalone, be stacked in a rack and be drastically less expensive than a computer with MIDI interface.

Several devices with this capability already exist. Most of them can do much more than a MIDI-to-OSC conversion, thus are too expensive [5][3] or not commercially available [1]. A project of the Upper Austria University of Applied Sciences was a dedicated MIDI-to-OSC converter, but it had some drawbacks (no DHCP capabilities, no maintained drivers) and has never been completed [4].

2. midOSC v1

Different possibilities have been envisaged to build a dedicated MIDI-to-OSC converter: microcontrollers, field-programmable gate arrays (FPGA) or embedded computers. The Gumstix embedded computers² offer the required features in terms of connectivity, as well as a versatile framework, which would allow to develop other tools beyond this specific application. Furthermore, working with a maintained operating system (OS) give access to regular updates of drivers or communication protocols, which makes the product easier to use and to develop.

2.1. Setup

Gumstix embedded computers are all based on the same principle: a motherboard with given CPU speed, installed OS and connectivity features can be connected to a bunch of extension cards for specific purposes: network, sound, Global Positioning System (GPS), touchscreen, sensors... Many drivers are already installed in the base image and the OS (a Linux distribution, but Windows CE exists as third-party software) can be accessed and configured over serial or ethernet connection.

² <http://www.gumstix.com>

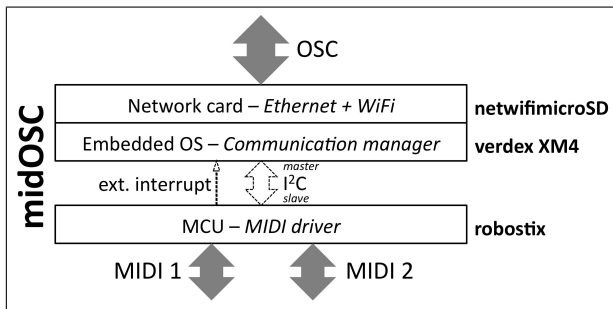


Figure 1. Gumstix cards and communication layers used in the midOSC converter

The cross-compile environment OpenEmbedded (OE)³ is used as a development platform to create and setup custom packages. A developer site, as well as a mailing list provide the necessary user documentation⁴.

The midOSC converter is based on a Gumstix *verdex XM4* motherboard with a 400 MHz PXA270 processor, connected to two extension cards: *netwifimicroSD* on one side for the Ethernet connection and *robostix* on the other side for serial communication at custom baud rate. The robostix works independently with a Atmel ATmega 128 microcontroller unit (MCU) and communicates with the motherboard as a slave on a Inter-Integrated Circuit (I²C) bus at 400 kHz (see Figure 1).

Since the ATmega 128 provides only two Universal Asynchronous Receiver/Transmitters (UARTs), the first midOSC version works with two MIDI ports. A second version is also envisaged with a custom developed serial interface, based on a ATmega 1280 MCU, which provides four UARTs (more details about developments in Section 4).

Connectivity is provided on the OSC side by the network card with a standard ethernet socket, and on the MIDI side with a custom board, connected with a flexible flat cable to the robostix UART pins. The complete setup is shown on Figure 2.

Basically, two main programs run on the midOSC converter: a MIDI driver on the robostix and a communication manager (CM) on the verdex. The MIDI driver is interrupt-driven, either by a UART (i.e. MIDI) reception or by a I²C call, when the motherboard received some OSC data and wants to forward them to MIDI. The communication manager consists of two threads, which react to an incoming OSC message, respectively to an interrupt on one of the verdex General Purpose In/Out (GPIO) pins, signifying that the robostix needs to send some MIDI data. Since the I²C protocol works on a master-slave configuration, the robostix (slave) cannot directly interrupt the motherboard. It also has to use an external interrupt on a GPIO line to ask the CM to start a bytes transfer.

³ <http://www.openembedded.org>

⁴ Developer site: <http://www.gumstix.net>; mailing-list: <http://www.nabble.com/Gumstix-f22543.html> [2009 Jan 23]

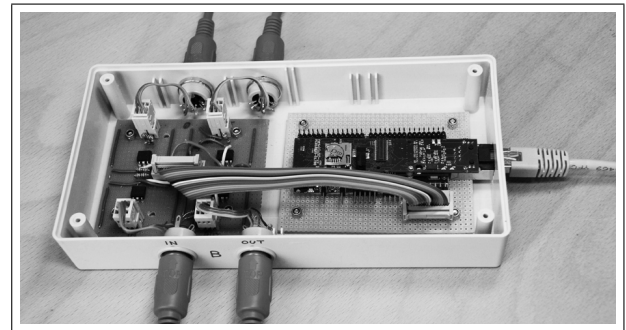


Figure 2. Packaging of the midOSC v1 converter with the Gumstix (right) and the custom electronic board

Several other applications run in the OS background, like the network and I²C drivers, the GPIO interrupt module or a web server, which is used to call a configuration interface. These applications are provided as standard packages in the Gumstix distributions and used as is.

2.2. Communication

midOSC devices can work in two different network configuration: peer-to-peer or node. In the peer-to-peer configuration, only two converters can be connected to each other and a 1:1 routing is made. This can be useful, when MIDI has to be transmitted over long distances (e.g. from the stage to the control desk).

In the node configuration, all the midOSC devices are clients of a router/DHCP server. Each midOSC has a MIDI port offset value, assigned by a rotary switch, which tells the device what its first MIDI port is.

A basic communication scheme of the midOSC converter consists of the following sequence:

1. A MIDI message is received on a UART and the communication manager is interrupted by the robostix on a GPIO line
2. The communication manager send an I²C read request to the robostix, which sends back its MIDI message
3. The CM creates a OSC message containing the MIDI port number and data, and send it to the network
4. The OSC message is received by the other devices, interpreted and – if needed – sent to their robostix, which directly push it to their corresponding UART

Figure 3 shows the standard configuration of a port-following transmission: an incoming MIDI message is received on port 1 of device 1 and is broadcasted with its port number to the OSC network. Every midOSC receive the data, but only those providing the same MIDI port number forward them.

Other types of routing can be achieved: broadcasting (every ports of every devices are addressed), device broadcasting (every ports of a specific device) or strict routing (one

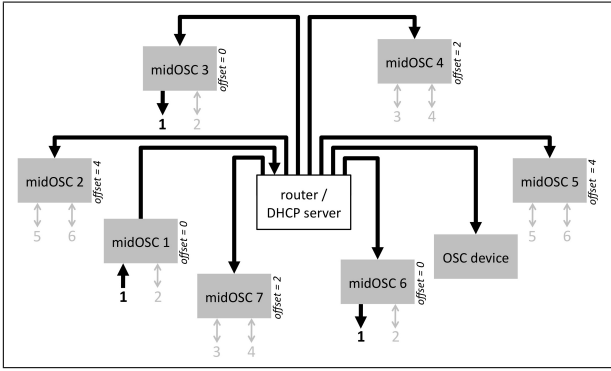


Figure 3. Port-following communication on a midOSC network: MIDI data is received on port 1, broadcasted and forwarded by the other devices providing a MIDI port 1

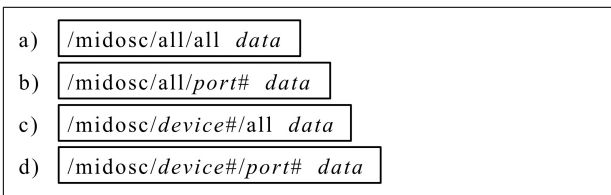


Figure 4. OSC address patterns: a) broadcasting b) port-following c) device broadcasting d) strict routing

specific port of a given device). The different OSC address patterns are shown in Figure 4. Naturally, other OSC-enabled devices can communicate in a midOSC network, assuming that they match the corresponding patterns.

3. Timing characteristics

One of the most critical features of a real-time system is its latencies. Therefore, timing measurements have been completed as the first setup characterization.

For timing measurements, a Max/MSP patch is used to generate two-, three-byte and system-exclusive (SysEx) MIDI messages and send them to a MIDI interface, where a probe measures the input signal. From the MIDI interface, the signal is sent to a first midOSC connected as shown in Figure 3 in a port-following mode. The output signal is measured on the same MIDI port of a second midOSC device.

Scope results of measurements (see Figure 5) show a communication sequence from the incoming message in a first device to the outgoing in a second one. One observes a large time offset induced by the I²C transfers on both sides, the GPIO interrupt on the motherboard and the network communication (both not visible in scope). This offset value is about 4 ms for a two-byte message and increases to about 30 μ s – the time needed for a I²C byte to be transferred – for each supplementary byte.

M. Nelson [8] and J. Wright [10] have both defined and measured latency in real-time MIDI setups. According to their definitions, a midOSC system has a latency of 5 ms,

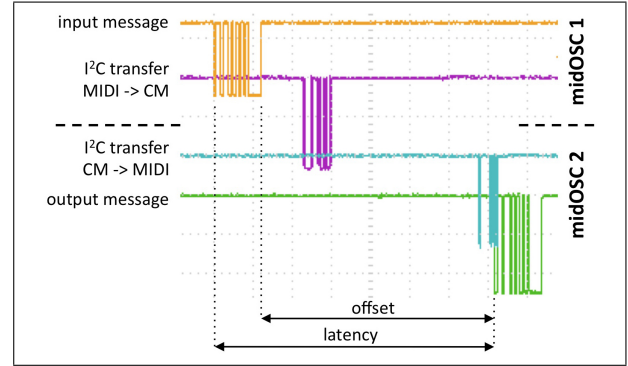


Figure 5. Timing sequence of a two-byte MIDI message: incoming bytes (yellow line) and I²C transfer to the motherboard (pink) on the first device. On the second device: I²C transfer from the motherboard to the robostix (blue) and outgoing MIDI bytes (green).

with a jitter of 0.4 ms and a standard deviation of 0.1 ms for a three-byte message.

One drawback of this setup is that the I²C transfer between the robostix and the motherboard on the first device takes place first when the complete MIDI message has been received, thus causing big latencies for long SysEx messages and has buffer size limitations. Some tests have been made in order to send each byte directly after its reception. The GPIO interrupt timing, as well as the I²C baud rate limitation (400 kHz) do not allow tight timing and stretch the outgoing message – i.e. insert delays between single MIDI bytes – of about 30 %.

Nevertheless, timing performances of the midOSC converter are clearly below the generally accepted limit values [8] for standard MIDI messages and some improvements discussed in Section 4.2 will allow to get rid of high latencies for SysEx messages.

4. Discussion

The midOSC converter shows a very interesting potential to be used as a standalone MIDI-to-OSC converter, allowing not to depend on computers to perform this specific task.

It is envisaged to make it available as open-source electronics and share schematics and source codes. For this purpose, price reduction, as well as improvements in connectivity, timings and range of use can be achieved.

4.1. Costs

Based on the prices given on the gumstix website on January 23, 2009, the midOSC components cost \$248.–⁵. With supplementary parts for connectivity and packaging, and without volume order price reduction, a midOSC converter costs about \$270.–.

⁵ With compatible new featured products: verdex pro, netpro-vx (without wifi) and robostix

In order to reduce price, as well as improve connectivity, the robstix expansion card will be replaced by a custom board based on a ATmega 1280 MCU, which gives access to four UARTs. This would reduce the midOSC costs to about \$220.–

4.2. Timing improvements

The timing characteristics highlighted that the system latency can become too high for long MIDI messages. An elegant solution to overcome this limitation would be to transfer each byte directly after reception without delaying the system. And that for all available MIDI ports. Assuming the use of a four-UART MCU, this implies that each byte has to be transferred within less than 80 μ s.

In order to achieve such timings, a faster communication protocol than I²C has to be used. Both the ATmega MCUs and the PXA270 microprocessor support the Serial Peripheral Interface (SPI) bus, which has a much higher throughput than I²C (e.g. up to 8 MHz baud rate for a 16 MHz MCU clock frequency). In addition, the currently used GPIO interrupt has to be temporally accurately controlled or replaced by another communication scheme.

When both tasks are implemented, the system latency should remain constant and is estimated to circa 3 ms. The buffer size as well will not be a concern anymore.

4.3. Perspectives

Beyond MIDI-to-OSC conversion, the Gumstix embedded systems offer a great range of possibilities to develop custom applications. Furthermore, the robstix expansion card has many I/O pins exposed on its headers⁶ smartly ordered with one supply and one ground for each channel. This allows standardized connection possibilities for in- and output devices.

Based on the midOSC framework, generic modules will be developed to enable low-latency bidirectional conversion between sensors data and OSC. This tool could be used as data-collecting and normalizing node for network performances, interactive installations or exhibitions.

A pedagogical aspect can also be emphasized since this system could be used as a quick prototyping tool to interface gesture acquisition devices, thus enable a time and resource-saving experimentation platform.

5. Conclusion

A MIDI-to-OSC converter has been developed on an Gumstix embedded computer. Despite currently not optimized timing characteristics, this device shows very good performances for short MIDI messages and will be optimized to a very competitive real-time converter. It shows also a great potential as a generic OSC conversion tool.

⁶ Eight 10-bit analog-to-digital, at least 16 digital and several pulse-width modulation (PWM) pins

The midOSC base setup will be developed in the one hand to a low-latency four-port MIDI-to-OSC, and in the other hand to a sensors-to-OSC converter, taking advantage of the available robstix connectivity to be used as a quick prototyping tool for pedagogical and artistic purposes.

6. Acknowledgements

Thanks to Jasch for fruitful conversations and improvement ideas. Thanks also to Peter Faerber for the introduction to the mobile Ambisonics equipment and the whole ICST staff for support, brainstorming and office sharing.

References

- [1] R. Avizienis, A. Freed, T. Suzuki, and D. Wessel. "Scalable Connectivity Processor for Computer Music Performance Systems," in *Proc. of the International Computer Music Conference (ICMC)*, Berlin, 2000
- [2] P. Faerber. "The Mobile Ambisonics Equipment", 2006 [2009 Jan 23], Available: <http://www.icst.net/index.php?show=163>
- [3] A. Fraietta. "The Smart Controller Workbench," in *Proc. of the Conference on New Interfaces for Musical Expression (NIME)*, Vancouver, 2005, pp. 46–49. See also [2009 Jan 23] <http://www.smartcontroller.com.au/>
- [4] G. Gessert. "mOSCito: multiple OSC performance controller," [Web site] 2006 [2009 Jan 23] Available: <http://www.hsse.fh-hagenberg.at/Projekte/mOSCito/>
- [5] S. Kartadinata. "the gluion. advantages of an FPGA-based sensor interface," in *Proc. of the Conference on New Interfaces for Musical Expression (NIME)*, Paris, 2006, pp. 93–96. See also [2009 Jan 23] <http://www.glui.de/>
- [6] D. G. Malham. "Ambisonics - A Technique for Low Cost, High Precision Three-Dimensional Sound Diffusion," in *Proc. of the International Computer Music Conference (ICMC)*, Glasgow, 1990, pp. 118–120. See also [2009 Jan 23] <http://www.ambisonic.net/>
- [7] MIDI manufacturers association. "The complete MIDI 1.0 detailed specifications," [Web site] 2008[2009 Jan 23] Available: <http://www.midi.org/techspecs/index.php>
- [8] M. Nelson, and B. Thom. "A Survey of Real-Time MIDI Performance," in *Proc. of the Conference on New Interfaces for Musical Expression (NIME)*, Hamamatsu, 2004, pp. 35–38
- [9] J. C. Schacher, P. Kocher. "Ambisonic Spatialization Tools for Max/MSP," in *Proc. of the International Computer Music Conference (ICMC)*, New Orleans, 2006
- [10] J. Wright. "System-Level MIDI Performance Testing," in *Proc. of the International Computer Music Conference (ICMC)*, Havana, 2001
- [11] M. Wright. "The Open Sound Control 1.0 Specifications," [Web site] 2002, [2009 Jan 23], Available: <http://opensoundcontrol.org/spec-1.0>
- [12] M. Wright, A. Freed, and A. Momeni. "OpenSound Control: State of the Art 2003," in *Proc. of the Conference on New Interfaces for Musical Expression (NIME)*, Montreal, 2003, pp. 153–159.