MAX Objects for Media Integration

Adrian Freed, David Wessel

Center for New Music and Audio Technologies (CNMAT), University of California
1750 Arch Street, Berkeley, CA 94709
adrian@cnmat.berkeley.edu, wessel@cnmat.berkeley.edu

David Zicarelli

Opcode Systems Inc., 3641 Haven Drive, Suite A, Menlo Park, CA 94025

Introduction

The MAX language has grown considerably since its origins as a MIDI scheduler [Puckette 1986] with a graphical programming interface [Puckette 1988]. The most important facility added as MAX was brought to its commercial form [Puckette and Zicarelli 1990, Schulz 1991, Zicarelli 1990] is the external object interface, which allows programmers to create new MAX objects in the C programming language. These objects are bound dynamically to MAX during initialization or when required. A similar dynamic linking facility is available in Apple's HyperCard (XCMD's) and Macromind Director (XObjects). MAX is unique among these environments in having a general-purpose, flexible real-time scheduler and a visual programming paradigm.

This paper introduces new MAX objects for media integration of interest to musicians and multimedia artists. They were created in C with the MAX externals interface. Design issues and applications of these objects will be discussed.

Random-access Media: Handshaking and Parallel Execution

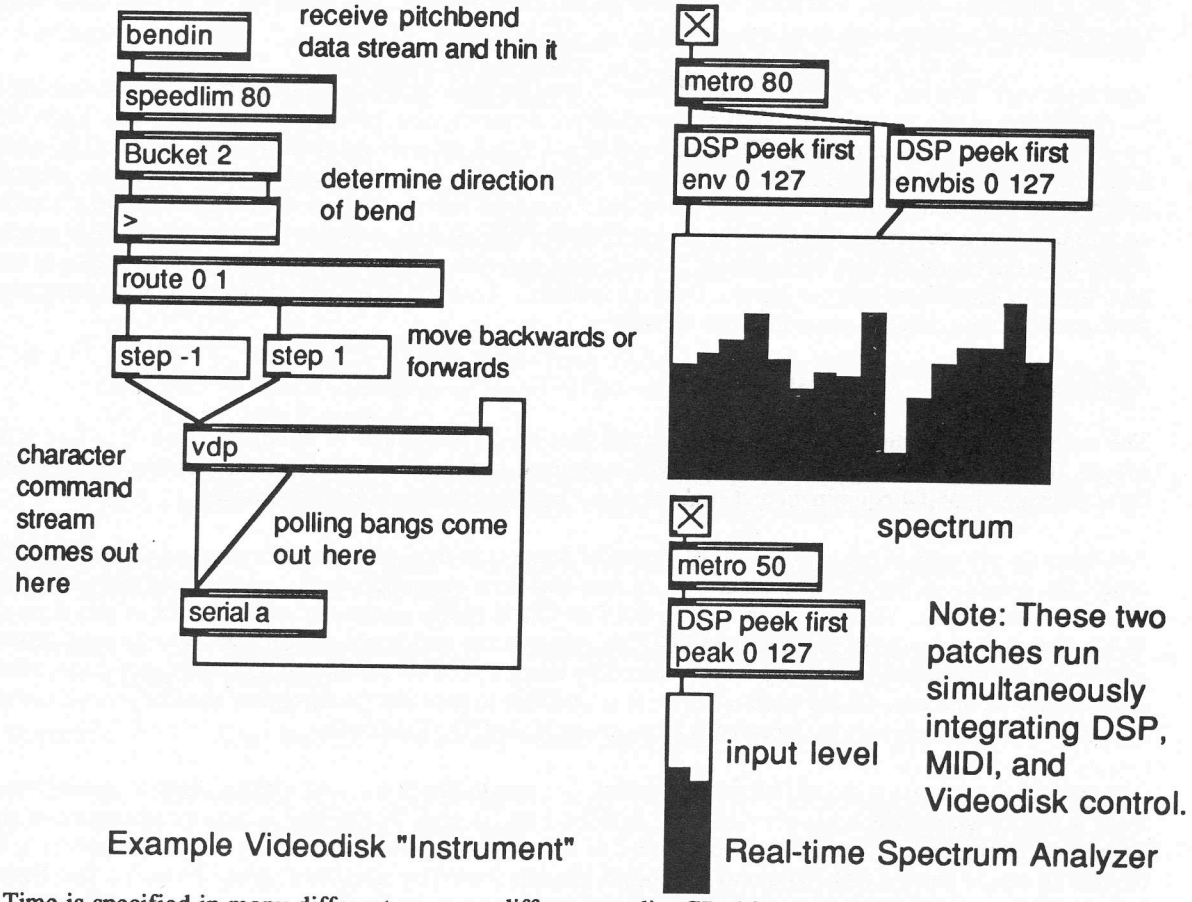
The objects to be described perform a similar function to drivers in operating systems – they provide an interface to the functionality of specialized hardware or software subsystems while hiding the details of the control and supervisory functions these systems require. The general structure is to receive messages in the MAX object's inlets and issue control information to a device or software subsystem. Status information obtained by polling or from an interrupt is packaged and sent out of the object's outlet. MAX runs as a single Macintosh thread, so permitting objects to "busy wait" for a device is inconsistent with real-time performance. Implementing certain handshaking schemes in MAX is therefor harder than in environments such as HyperCard, but the benefit is that many devices may be controlled simultaneously.

The **vdp** object, for controlling laser videodisk players, illustrates these difficulties. It must communicate with a simple processor inside the videodisk player which holds one line of received command characters and issues an acknowledgement when ready to receive the next line. Failure to follow this protocol leads to disaster. The first design decision is to separate the details of Macintosh serial port communication from the videodisk control protocol. This is achieved by connecting the **vdp** object to the general-purpose serial port object. After sending a command to **serial**, **vdp** begins sending bang messages every 20 milliseconds using a MAX timer facility. If any characters have been received by the **serial** object, a bang will cause them to be sent out its outlet. The outlet of the **serial** object is connected to a special inlet of the **vdp** object. When **vdp** sees the special acknowledge character sequence, it sends out the pending command and restarts the polling. There is no waiting in this scheme and other processes can execute in the middle of a command/acknowledge cycle, which may take as long as one second.

Random-access Media: Timing, Synchronization and Triggering

Random-access media devices such as compact disk and videodisk players are not as responsive to control signals as MIDI instruments. There are unavoidable delays triggering sounds or searching to specific locations within a medium. In addition, these devices often have idiosyncrasies that limit the effective

bandwidth of continuous control. Compensating for idiosyncrasies is often inseparable from the performer's interactive control task, so the MAX device control objects are designed to offer low level services. This moves the burden of resolving the idiosyncrasies and developing performance control structures to higher level patches. For example, to "scrub" a series of videodisk frames by moving a MIDI controller such as a pitchbend wheel, the MAX **speedlim** object is used to thin out the stream of commands sent to the **vdp** object. The maximum command rate is a function of the communication data rate and access time of the videodisk player in hand. Adjustments to the responsiveness of a videodisk "instrument" can thus be made without changing the **vdp** object itself.



Time is specified in many different ways on different media. CD drives use minutes, seconds, and blocks (1/75 of a second). NTSC Videodisks use 30 frames per second. There is inherent uncertainty in synchronizing devices, since Macintosh device drivers do not tag transactions with a physical time reference. It is possible to poll a device for the current frame or block being read with the caveat that there could be uncertainty about when the device made its determination.

Audio Signal Processing

DSP object: This object encapsulates the functions of a Digital Signal Processor driver [Freed and Gordon, 1990]. It supports the Motorola DSP56001's designed into the Polysonic Reson8, Digidesign AudioMedia and SoundAccelerator, and the Studer EdiTech MacMix Excelerator. This object responds to messages to configure and load code into DSP nodes and read and write vectors of parameters from and to the nodes. These reads and writes are done atomically to avoid, for example, the dribbling in of critical parameters like filter coefficients. The example above illustrates how users can combine the DSP object and the user interface object **multislider** (the bar graph) into a useful application.

Gestural Interfaces (non-MIDI controllers)

It is simple to create objects for gestural controllers that are interfaced to the Macintosh via MIDI using MAX's extensive set of MIDI objects. The following controllers are interfaced using more exotic methods.

Glove: The Mattel Power Glove is a controller for the Nintendo Entertainment Systems (NES). The Gold Brick [Transfinite 1991] interfaces NES controllers to the Macintosh ADBport. The **glove** object communicates with the Gold Brick to provide eight gestural variables from the Power Glove, including X, Y and Z location, rotation, and bend of the thumb and three fingers. The glove object reports these values every time it receives a bang message.

Radio Drum: The Mathews/Boie drum [Mathews and Schloss 1989] is a spatial controller consisting of two radio frequency transmitting sticks moved above a rectangular, planar array of receivers. Each stick transmits on a different frequency. The amplitude of the signal received by each antenna at one of the stick's frequencies is a non-linear function of the distance of that stick from the antenna and its distance from the other stick. These non-linear functions have been mapped into the position of each stick in Cartesian coordinates using a neural network [Lee, Freed, Wessel 1991]. A Motorola 68020 coprocessing card running a real-time operating system implements the mapping and also measures the velocity of the sticks as they pass through predefined trigger planes. Outputs available from the MAX drum object include continuous stick position data, trigger plane hits and velocity.

Applications and Conclusions

The external objects discussed here demonstrate that MAX is capable of handling more than just MIDI events. The ability to use MIDI and non-MIDI devices in the same environment can serve as a bridge between existing MIDI equipment and equipment not intended for musical performance.

Applications abound. One application of practical interest to the computer music community is to bring more interactivity to the existing repertoire of non-real-time computer music, particularly pieces for tape and live instruments. Transfer from tape to DAT or CD is easily accomplished and once in this form the MAX objects that control CD-ROM and DAT players can cue and begin sections in a timely way. Such a computer-controlled playback system is enhanced by using a MIDI controlled audio mixer such as the Niche Audio Control Module. Using such a device it is possible to provide performance based dynamic control including tightly synchronized cross-fades between multiple CD-ROM drives.

One would then create a list of "hit points" which are sent to the MAX **past** object. As the time reference from a device goes past a hit point a bang is issued and a new "hit point" is set. Applications of this include sounding MIDI events or sequences tied to sound from a CD or displaying videodisk frames to the rhythm of music from a CD. The way videodisk players search by quickly playing through a few frames before the target frame allows actors in a movie to appear to be dancing to music on a CD. Another interesting application that relies on careful synchronization is to put copies of the same CD in two or more drives and overlap playing from one with searches on the other. The resulting real-time seemless edits from hours of audio is much cheaper to achieve than from hard disks. Combining this with MIDI-controlled effects processors or direct digital feeds to DSP cards all under MAX control provides an even richer performance environment.

An important goal in the successful integration of multimedia devices into MAX is a coherent control style which lends itself to real-time performance. Performers would like to see these devices as new instruments. Unfortunately, the current generation of devices were not designed with this in mind. The MAX objects described go a long way towards integrating these diverse devices and we are encouraged by the interest manufacturers are demonstrating in these new applications and the new hardware, protocols and drivers they are creating specifically for computer control, e.g., NEC's PC-VCR and Sony's Computer Video Drive (CDV-1000).

Acknowledgements

This work was supported in part by grant INV9004-011 from Apple Computer, Inc.

References

Puckette, M. (1986), "Interprocess Communication and Timing in Real-Time Computer Music Performance", *Proceedings of the12th International Computer Music Conference, The Hague,1986,* Computer Music Association.

Puckette, M. (1988), The Patcher, *Proceedings of the 14th International Computer Music Conference,* Köln,1988, Feedback Studio Verlag, available from Computer Music Association.

Freed, A., Gordon, K., (1990), "DSP Driver Software for Performance-Oriented Music Synthesis Systems", *Proceedings of the 16th International Computer Music Conference, Glasgow 1990,* Computer Music Association.

Lee, M. Freed, A., Wessel, D. (1991), Real–Time Neural Network Processing of Gestural and Acoustic Signals, *Proceedings of the 17th International Computer Music Conference, 1991,* Computer Music Association.

Mathews, M., Schloss, A. [1989], "The Radio Drum as a Synthesizer Controller", *Proceedings of the 15th International Computer Music Conference, 1989.* Computer Music Association.

Puckette, M., Zicarelli, D. (1990), *MAX – An Interactive Graphic Programming Environment,* Opcode Systems, Menlo Park, CA, 1990.

Shultz C. (1991), "Opcode MAX: Interactive Graphic Programming Environment (Macintosh)", *Keyboard Magazine,* May 1991.

Transfinite [1991], *Gold Brick User's Manual,* Transfinite Systems, Cambridge, MA, 1991.

Wessel, D. "Improvisation with Highly Interactive Real-Time Performance Systems", *Proceedings of the 17th International Computer Music Conference, 1991,* Computer Music Association, 1991.

Zicarelli, D. "Writing External Objects for MAX," Opcode Systems, Menlo Park, CA, 1990.