Manufatura e programação de controladores: possibilidades de desenvolvimento para aplicação em sistemas musicais interativos

José Henrique Padovani[1]
Mestrando/Instituto de Artes da Unicamp
padovani@iar.unicamp.br

Sérgio Freire
Departamento de Teoria Geral da Música/Escola de Música da UFMG
sfreire@musica.ufmg.br

Sumário:
Neste artigo descrevemos os resultados e as perspectivas de estudo decorrentes da pesquisa voltada à manufatura e à programação de controladores especialmente desenhados para a utilização em música interativa (principalmente, nos programas Pure Data e Max/MSP). Dois controladores manufaturados são apresentados. O primeiro, construído a partir do microcontrolador BasicStamp2sx e o segundo baseado na plataforma livre de computação física Arduino. Também são discutidos aspectos mais gerais como pertinência de se investir esforços na elaboração de novos controladores no contexto de pesquisas voltadas à criação musical com sistemas interativos.

Palavras-Chave: música e tecnologia, sistemas musicais interativos, controladores, computação física

1. Interação algorítmica e interação física

É possível perceber um grande desenvolvimento das atividades artísticas que exploram a interatividade através de recursos computacionais nos últimos dez anos. Ambientes de programação como Pure Data[2] e Max/MSP[3], primeiramente limitados, por força de restrições técnicas, à comunicação via protocolo MIDI com módulos externos dedicados à síntese e ao processamento do som, passaram a possibilitar o processamento digital de sinais em tempo real e, mais tarde, expandiram suas possibilidades para a síntese e o processamento de imagens.

Vários algoritmos implementados nesses ambientes são capazes de processar digitalmente o som sem necessitar de um grande nível de interação, isto é, de alteração de seus parâmetros por um usuário. Entretanto, em grande parte das situações de concerto em que sistemas musicais interativos são utilizados é comum que o programador/usuário/compositor precise auxiliar o intérprete, enviando comandos ao computador. Tal situação é freqüente quando faz-se necessário alterar certos parâmetros dos algoritmos em questão, trocar os processos de manipulação algorítmica utilizados ou mesmo possibilitar uma improvisação ou um ajuste de certos parâmetros (como ganho, espacialização, reverberação) no momento da performance. Nestas situações, quando a interação determina de maneira imediata o resultado musical, as interfaces de controle tradicionais dos computadores pessoais (limitados, geralmente, a teclados e mouses) se mostram extremamente insuficientes. De fato, o teclado do computador é pouco gestual se comparado aos instrumentos musicais e o mouse reduz nossos 10 dedos a um único ponteiro[4].

Observa-se assim um aumento considerável das possibilidades algorítmicas de interação, enquanto as interfaces físicas geralmente permanecem limitadas ao convencional. Nesse artigo apresentamos duas

[1] Bolsista da Fundação de Amparo do Estado de São Paulo. Parte da pesquisa aqui apresentada foi realizada durante a graduação na Escola de Música da UFMG, com o apoio da Fundação de Amparo a Pesquisa do Estado de Minas Gerais.

[2] Cf. http://www.puredata.info

[3] Cf. http://www.cycling74.com

[4] Cf. O'Sullivan e Igoe, 2004: xix.

interfaces semelhantes construídas sobre plataformas diferentes, ressaltando as possibilidades de cada uma no desenvolvimento de controladores musicais destinados à utilização no contexto de música interativa.

2. Montagem de interfaces de controle: para além da mimese instrumental

Grande parte dos controladores existentes consiste em artefatos que procuram imitar instrumentos tradicionais como teclados, superfícies de tambores ou instrumentos de sopro. A lógica de sua construção é aquela de reproduzir, com maior fidelidade possível, aspectos físicos e interativos de instrumentos tradicionais[5]. Contudo, tanto a configuração física destes instrumentos quanto o tratamento algorítmico dos impulsos recebidos costuma oferecer resistência à utilização em contextos específicos de interação, o que é natural, já que eles não são construídos para funcionar em situações específicas, mas em situações generalizadas.

As interfaces aqui apresentadas destinam-se, prioritariamente, ao controle de parâmetros unidimensionais contínuos e discretos através de potenciômetros deslizantes e giratórios e botões do tipo *switch*. Contudo, os recursos técnicos estudados possibilitam a fácil implementação de interfaces mais integradas à técnica tradicional dos instrumentistas, o que permite um controle musicalmente mais refinado de parâmetros de síntese e processamento de áudio e de dados musicais a serem tratados algoritmicamente. O desenvolvimento de novos controladores para utilização em sistemas musicais interativos cria um tipo de pesquisa composicional que vai desde uma *luteria* digital aos problemas específicos da composição e da performance, agregando a potencialidade gestual da música instrumental à versatilidade sonora da música eletroacústica[6].

3. A transformação de uma mesa de som em um controlador MIDI

O primeiro projeto realizado consistiu na transformação de uma mesa analógica de mixagem de áudio TEAC 2A em um controlador MIDI. Para a digitalização da informação transmitida através de 8 potenciômetros (4 deslizantes e 4 giratórios) foi utilizado um circuito construído a partir das funções de digitalização do microcontrolador BasicStamp 2Sx (BS2sx), da fabricante Parallax.

O BS2sx possui recursos nativos de medição de resistências variáveis através de circuitos RC (resistor/capacitor) e da função RCTIME. Um capacitor ligado à resistência variável é carregado pelo microcontrolador que, em seguida, mede o tempo de descarga da tensão armazenada no capacitor através da resistência do potenciômetro. O código para ler um potenciômetro ligado ao pino 7 do *chip* através do circuito exibido na figura 1 seria escrito desta maneira na linguagem PBASIC[7]:

```
RC PIN 7
x     VAR Word
Main:
DO
   HIGH RC          ' carrega o pino 7
   PAUSE 1          ' pausa por 1 ms
   RCTIME RC, 1, x  ' atribui à variável x o valor lido
LOOP
END
```

O comando HIGH é responsável por carregar o pino, o comando PAUSE estabelece 1 ms de pausa para que o capacitor se carregue totalmente e o comando RCTIME faz uma contagem de quanto tempo leva para que o pino vá do estado 1 (carregado) a 0 (descarregado) e, em seguida, atribui o valor lido à variável *x*. LOOP faz com que o processo ocorra continuamente. Vale observar que a função RCTIME é um contador que opera a ciclos de 0.8 µs e faz uma contagem até 65535.

Se o capacitor se descarrega imediatamente (a resistência do potenciômetro é nula), a pausa de 1ms do comando PAUSE é suficiente para que o capacitor se descarregue e a função RCTIME levará apenas 1 ciclo (0.8 µs) para atribuir à variável *x* o valor 1. Contudo, na medida que a resistência aumentar, maior será o tempo demandado para que o valor seja atribuído (já que o capacitor levará mais tempo para se

[5] Cf. Wanderley e Depalle, 1999: 153.

[6] Cf. Collins, 2006: 192-193.

[7] Cf. Parallax, 2005: p. 7.

descarregar). O valor atribuído será igual ao número de ciclos contados. Se a contagem de 65535 chegar ao final e o pino não mudar de estado[8], a função RCTIME levará 54,428 ms (65535 x 0.8 µs) para retornar à variável *x* o valor 0.

Figura 1. Mesa TEAC conectada ao módulo BS2sx

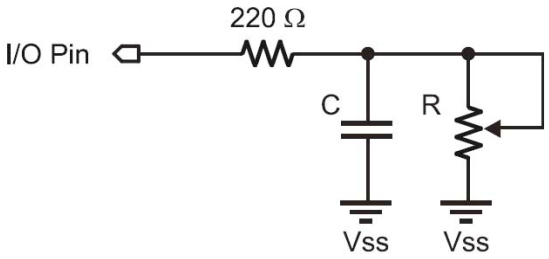

Figura 2. Circuito RC

Para a formatação da informação recebida no protocolo MIDI utilizamos o comando SEROUT (Serial Out). Este comando possui 4 variáveis. A primeira se refere ao pino que envia informações seriais. O segundo, denominado *BaudMode*, é uma constante que define informações sobre *baud rate* codificadas em 16 bits. A variável 3, *Pace*, define um *delay* opcional entre cada byte enviado. A última variável, *OutputData* envia os bytes (separados por vírgulas). Para receber as informações dos potenciômetros mantendo a resolução de bits alta, enviamos as informações pelo comando *Pitch Wheel Change*, que aceita 14 bits de informação (divididos em 2 bytes com 7 bits de informação, cada)[9]. Como o protocolo MIDI reserva apenas um *Pitch Wheel Change* por canal, é utilizado um canal MIDI para cada potenciômetro.

Abaixo está transcrito o código para receber a informação referente a um potenciômetro, ligado por circuito RC ao pino 0 do módulo:

```
Pot0 VAR Word
Main:
DO
      HIGH 0
      PAUSE 1
      RCTIME 0,1,Pot0
      Pot0 = Pot0<<1
      SEROUT 15, 60, 0, [224,Pot0.LOWBYTE>>1, Pot0.HIGHBYTE]
LOOP
END
```

[8] O que significa que não há resistência variável conectada ao circuito ou que o valor dela é demasiado alto para permitir que o capacitor se descarregue.

[9] Cf. http://www.midi.org/about-midi/table1.shtml

4. Um controlador baseado na interface Arduino NG

O segundo projeto consistiu na montagem de um controlador com 24 potenciômetros (12 deslizantes e 12 giratórios) e 8 botões do tipo *switch*. A conversão das informações analógicas, a programação da interface e a comunicação com o computador se dão de maneira diferente que no projeto da mesa TEAC. No lugar do BS2sx, é utilizada a interface de computação física Arduino NG[10]. Trata-se de um circuito baseado no microcontrolador Atmega168 acompanhado de um *bootloader* que o permite ser programado em uma linguagem derivada da *Wiring*[11] em um ambiente de programação desenvolvido inicialmente para a linguagem de programação *Processing*[12].

Diferentemente do projeto anterior, este utiliza comunicação USB, já que Arduino NG possui um conversor Serial USB integrado (FTDI232R). Arduino também possui uma versão *Bluetooth*, o Arduino BT, que permite que os mesmos circuitos e o mesmo código utilizado para programar o microcontrolador sejam utilizados para se criar interfaces que se comunicam com o computador sem a utilização de cabos.

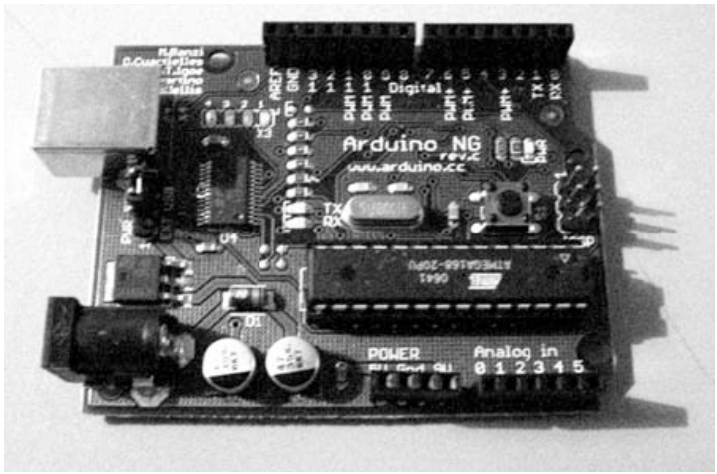

Figura 3. Arduino NG

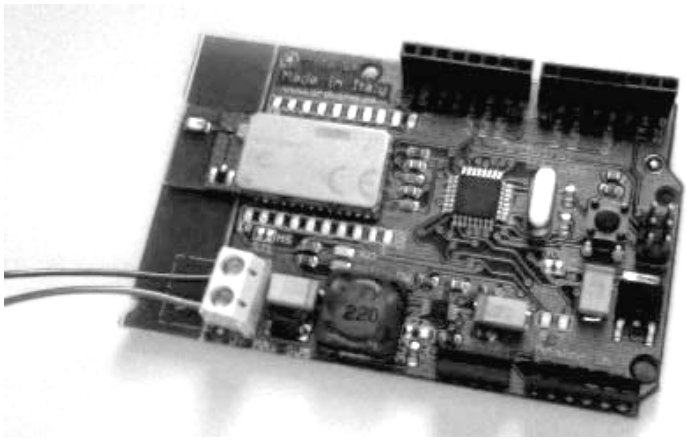

Figura 4. Arduino BT

Diferentemente do BS2sx, Arduino é capaz de medir variações de tensão elétrica, não sendo necessária a utilização de circuitos RC. Assim, basta conectar uma das pontas do potenciômetro ao pino de 5 volts da placa e, a outra ponta a um fio ligado ao pino GND (terra) da placa. No pino central do potenciômetro conectamos um fio que se liga a um dos pinos analógicos da placa, que realiza a leitura da voltagem. O microcontrolador escala os valores lidos a 10 bits (0 a 1023).

Como Arduino NG possui apenas 6 entradas/saídas analógicas e 13 entradas/saídas digitais (sendo que duas ficam reservadas à comunicação serial com o computador), utilizamos três multiplexadores

[10] Cf. http://www.arduino.cc/

[11] Cf. http://www.wiring.org.co/

[12] Cf. http://processing.org/

analógicos de 16 canais CD74HC4067. Dois deles fazem com que as 24 resistências referentes aos potenciômetros possam ser medidas através de apenas dois pinos analógicos da interface (16 canais no pino 0 e 8 canais no pino 1). O terceiro multiplexador faz com que as informações referentes ao estado de cada um dos 8 *switches* possam ser transmitidos através de uma única entrada digital. Para comandar os multiplexadores, 4 pinos digitais do Arduino são utilizados. A combinação dos estados possíveis desses 4 pinos (HIGH ou LOW) faz com que o circuito CD74HC4067 leia cada uma de suas 16 entradas.

Para a formatação das informações da placa e seu envio ao computador, utilizamos a biblioteca *SimpleMessageSystem*[13], que converte as informações lidas para a codificação ASCII (*American Standard Code for Information Interchange*) antes de enviar o dados através da porta serial. A vantagem dessa formatação é que a informação recebida pode ser facilmente lida em ambientes de programação como Max/MSP e Puredata, que utilizam de tal codificação para representar dados numéricos e de controle[14].

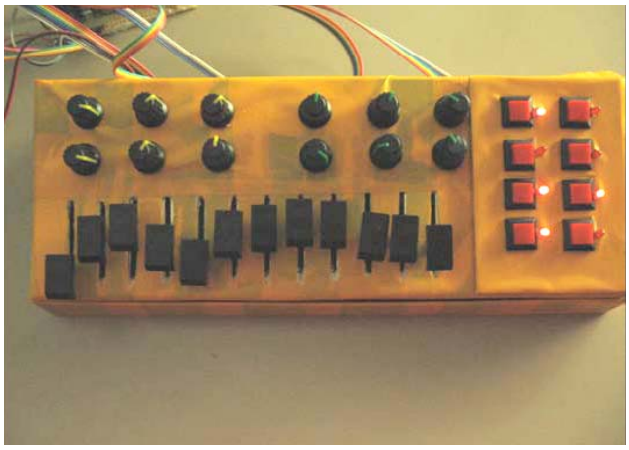

Figura 5. Interface baseada em 24 potenciômetros e 8 botões

Figura 6. Comunicação com um patch programado no aplicativo Puredata/GEM

5. Comparação técnica e perspectivas de pesquisa

Embora em ambos os projetos o objetivo tenha sido aquele de construir interfaces do mesmo tipo, a metodologia utilizadas na manufatura e programação de cada um deles é bastante diferente. Enquanto na mesa TEAC, a medição das resistências se dá por circuito RC, no protótipo desenvolvido sobre a plataforma Arduino, isso é feito através de recursos do microcontrolador Atmega168 de medição da própria tensão

[13] Cf. http://www.arduino.cc/playground/Code/SimpleMessageSystem

[14] Mais detalhes sobre o controlador baseado na plataforma Arduino podem ser obtidos na página: http://www.padovani.googlepages.com.

variável. Na mesa TEAC, a comunicação com o computador se dá através do protocolo MIDI a uma *baud rate* de 31250 Bd. No segundo caso, pode-se utilizar uma *baud rate* de até 115200 Bd, a comunicação com o computador pode ser realizada sem fios e a formatação dos dados é realizada com o auxílio da biblioteca *SimpleMessageSystem*, que codifica os dados computados em caracteres ASCII (o que amplia o número de bits enviados mas reduz o trabalho de programação nos ambientes que recebem os dados). Deve-se mencionar que a linguagem de programação *Arduino* é muito mais estruturada e que seu formato *open-source* facilita imensamente a obtenção de informações, crucial no desenvolvimento de novos controladores num contexto de pesquisa interdisciplinar[15].

Nos protótipos desenvolvidos o interesse era aquele de criar uma interface semelhante a uma mesa de mixagem. No entanto, a complexidade técnica em programação e eletrônica é a mesma daquela que seria necessária para projetos envolvendo sensores e componentes de controle a serem utilizados por um instrumentista ou *performer* para, ele mesmo, controlar parâmetros e variáveis dos algoritmos de interação, gerando assim *hiperinstrumentos*[16], destinados à utilização em composições e contextos específicos - que é o foco dos nossos atuais interesses de pesquisa.

Referências bibliográficas

Collins, Nicolas. (2006). *Handmade Electronic Music*. New York: Routledge.

O'Sullivan, D. e Igoe, T. (2004). *Physical Computing: Sensing and Controlling the Physical World with Computers.* Boston, MA: Thomson.

Parallax (2005). *BASIC Stamp Syntax and Reference Manual v.2.2*. Disponível em: http://www.parallax.com/Portals/0/Downloads/docs/prod/stamps/web-BSMv2.2.pdf. Acessado em 31 de Maio de 2008.

Rowe, Robert (1994). Interactive Music Systems. Cambridge: MIT Press.

Wanderley, Marcelo e Depalle, Philippe (1999). "Contrôle gestuel de la synthèse sonore" in *Interfaces homme-machine et création musicale.* Org.: H. Vinet e F. Delalande. Paris: Hermès. p.145-164.

[15] No sítio http://www.arduino.cc existem vários tutoriais e um fórum onde é possível obter informações que vão desde a manuais sobre técnicas básicas de soldagem até guias relacionados à utilização de componentes e sensores variados ou a formatação de dados em protocolos específicos.

[16] Cf. Rowe, 1993: 74-78.