REVISTA CIENTÍFICA DA UMC

PERIFÉRICO DE CONTROLE MIDI COMO FERRAMENTA DE AUXÍLIO AO ENGENHEIRO DE MIXAGEM

MIDI CONTROL PERIPHERAL AS A TOOL TO SUPPORT THE MIXING ENGINEER

Everson Dressler Araújo, Christian Álvaro de Camargo, Marcelo Teixeira de Azevedo, Sidney Longo, Paulo Vitorino

Resumo:
Nesta pesquisa é proposta uma ferramenta para facilitar a rotina de trabalho de um engenheiro de mixagem, através de um hardware capaz de controlar as principais funções dos softwares que são utilizados atualmente em *studios* e *home studios*. Tal abrangência é possível por conta da padronização da comunicação musical em vias eletrônicas denominada MIDI (*Musical Instrument Digital Interface*), em português "*Interface Digital de Instrumentos Musicais*". A construção feita com base no microcontrolador da *Microchip*, o PIC18F4553, com comunicação USB que torna a conexão com o computador bastante pratica. O engenheiro de mixagem ou estudante de mixagem que fizer uso deste periférico em seu trabalho sentira conforto, realizara as tarefas com mais agilidade, em controles amplamente intuitivos e com aparência amigável e realista.

Palavras-chave: MIDI; DAW; *Studio*; *Home Studio*; Mixagem; PIC.

Abstract:

In this research, a tool is proposed to facilitate the work routine of a mixing engineer, through a hardware capable of controlling the main functions of the software that are currently used in studios and home studios. Such scope is possible due to the standardization of musical communication in electronic ways called MIDI (Musical Instrument Digital Interface), in Portuguese "Digital Interface of Musical Instruments". The construction based on Microchip's microcontroller, the PIC18F4553, with USB communication that makes connecting to the computer very practical. The mixing engineer or mixing student who makes use of this peripheral in his work will feel comfortable, perform tasks more quickly, in largely intuitive controls and with a friendly and realistic appearance.

Keywords: MIDI; DAW; Studio; Home Studio; Mixing; PIC.

INTRODUÇÃO

A necessidade de controlar a DAW (*Digital Audio Workstation*), em português "Estação de Trabalho de Áudio Digital", é a necessidade de todos *studios* que utilizam sistema de gravação digital, mesmo os grandes estúdios, com capacidade de

trabalhar com periféricos analógicos, fazem uso de DAW e também fazem uso de ferramentas digitais em suas consoles.

As DAWs nativas são controladas pelos periféricos que fazem parte da composição do computador, que são: o teclado e mouse. Mas a utilização destes periféricos torna o processo cansativo, além de não ter ligação visual com a virtualização apresentada pelo software.

Além dos protocolos proprietários feitos para comunicação fechada entre equipamentos de uma mesma empresa, o protocolo MIDI (*Musical Instrument Digital Interface*), em português “Interface Digital de Instrumentos Musicais”, traz o MIDI CC, ou (MIDI *Continuous Controllers*), em português “Controles Contínuos MIDI”, que se trata de um protocolo reconhecido e utilizado por qualquer dispositivo que utilize comunicação MIDI.

O problema em um controlador MIDI para controle de plugins em DAWs é o número elevado de fabricantes e aplicações da ferramenta, tornando impossível cobrir todos os modelos usuais do mercado, como solução, o controlador utilizado neste projeto é um dos mais populares em estúdios, além de ter controles com distribuição genérica, tornando a utilização natural mesmo em plugins diferentes.

Métodos

Os conceitos que torna possível a comunicação entre periférico e computador, possibilitando o controle de softwares, através do microcontrolador são: comunicação e protocolo USB (*Universal Serial Bus*), Protocolo MIDI, Protocolo MIDI CC e Protocolo *MIDI Mackie Universal Control*.

Para este projeto o microcontrolador utilizado é o PIC18F4553 fabricado pela Microchip. O PIC18F4553 possui 12 bits de conversão A/D.

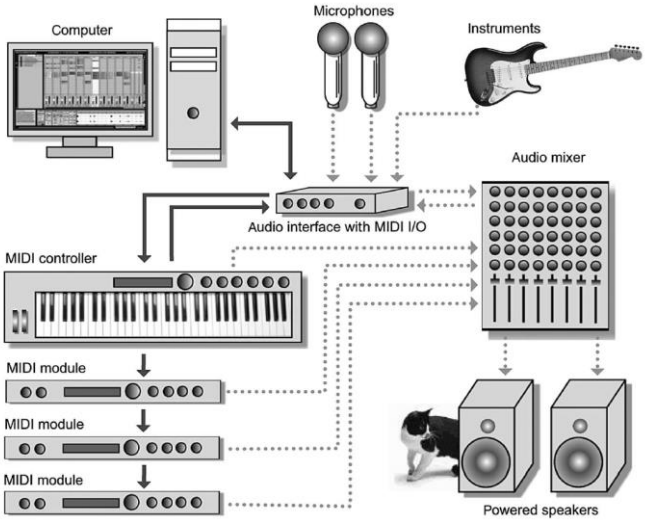
O que torna este microcontrolado apropriado para a aplicação necessária neste projeto é; em primeiro lugar a comunicação USB, necessário para ligar o computador ao periférico; em segundo o número de entradas de conversão de sinais analógicos para sinais digitais, que é o tipo de sinal manipulável pelo microcontrolador; em terceiro, a disponibilidade de biblioteca direcionada a comunicação USB MIDI. Além

de toda capacidade que torna o microcontrolador totalmente aplicável neste projeto, o formato do encapsulamento do microcontrolador facilita a prototipagem, o formato PTH (*Pin Througt Hole*), pelo seu tamanho e formato de pinagem, disponibiliza soquetes tornando a montagem de giga de teste bastante fácil e prático. E para projetos definitivos, o fabricante fornece encapsulamento SMD (*Surface Mounted Device*), reduzindo o tamanho final do projeto.

No final da década de 1970, os instrumentos musicais eram comuns e acessíveis na América do norte, Europa e Japão, o controle destes instrumentos era feito a partir da condução elétrica em suas teclas, os fabricantes utilizavam esta condução elétrica para controlar um ou mais dispositivos, este era um sistema ineficiente para controlar sintetizadores digitais e polifônicos mais recentes. Alguns fabricantes criaram sistemas de interligação aos seus próprios equipamentos, o que tornava incompatível a ligação de equipamentos de fabricantes diferentes. Em 1981, o fundador da Roland Corporation, propôs a ideia de padronização da comunicação entre equipamentos, e empresas como *Oberheim Electronics, Sequential Circuits*, Yamaha, Korg e Kawai aceitaram o desafio. Nos próximos dois anos, o padrão de comunicação foi discutido e modificado pelos representantes das empresas envolvidas. O desenvolvimento do MIDI tornou-se público na edição de outubro de 1982 da revista *Keyboard*, em janeiro de 1983 foi feito uma demonstração de uma conexão MIDI entre um sintetizador e um teclado Roland JP-6. A especificação MIDI foi publicada em agosto de 1983 (MIDI, 2021).

De forma simples, o MIDI é uma linguagem de comunicação digital que permite a instrumentos eletrônicos controle e desempenho, também permite que computadores e outros dispositivos relacionados se comuniquem entre si por meio de uma rede conectada. O protocolo de comunicação MIDI é usado para traduzir eventos relacionados à performance e controle, tais como tocar um teclado, variar uma roda de modulação, disparar um efeito visual encenado, a partir da transmissão de mensagens digitais para outro dispositivo MIDI controlando parâmetros de desempenho. Uma mensagem MIDI pode ser gravada em um hardware ou Software conhecidos como sequenciadores, nos sequenciadores é possível editar e transmitir a outros instrumentos ou dispositivos, a fim de criar ou controlar qualquer número de parâmetros (HUBER,2007).

Figura 1: Exemplo de um Sistema MIDI típico com as conexões de áudio.



Fonte: Próprio Autor, 2020.

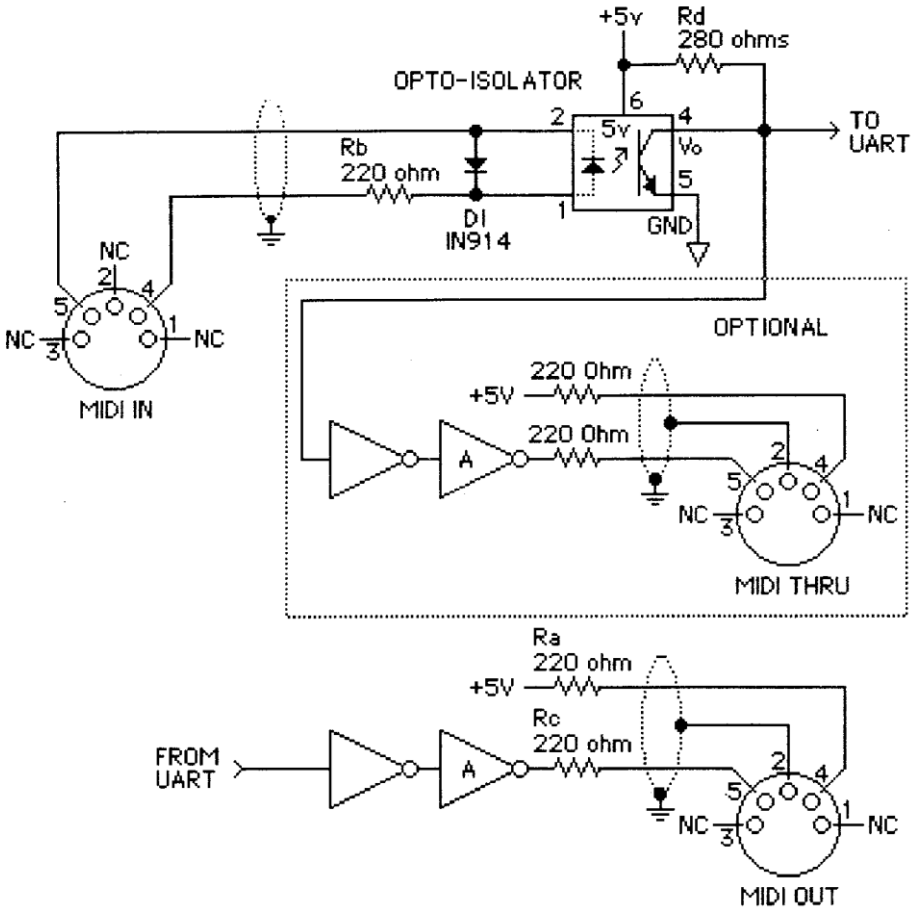
MIDI é uma interface de controle remoto digital para sistemas musicais. Um equipamento controlado por MIDI normalmente baseia-se em controle do microprocessador, com a interface MIDI formando uma porta de I/O. Por sua popularidade como meio de controle, a comunicação MIDI foi adotada para outros sistemas de áudio visuais, incluindo a automação de consoles de mixagem, controle de equipamentos externos do estúdio, controles de sistemas de iluminação e outras máquinas de estúdio. Apesar de seus comandos, por padrão, serem direcionados à música, é possível adaptar seus comandos musicais para fins não musicais, ou usar comandos para métodos alternativos de controle. A escolha do padrão serial para MIDI foi concebida considerando economia e prática, tendo previsão de que fosse possível que a interface instalada em equipamentos relativamente baratos, sendo disponível para uma ampla gama de usuários. A simplicidade e facilidade de instalação dos sistemas MIDI, foram responsáveis por sua rápida aceitação como padrão internacional. O sistema MIDI integra comandos de controle de tempo e sistema com comandas de disparo de tom e nota, desta forma informações podem ser transportadas no mesmo formato pelo mesmo fio. O protocolo MIDI permite controlar instrumentos musicais em tempo pseudo-real, ou seja, a velocidade de transmissão é tal que atrasos na transferência dos comandos de execução não são audíveis na maioria dos casos. Além de ser possível endereçar vários dispositivos de recepção separados em um único fluxo de dados MIDI, permitindo que um dispositivo de

controle determine o destino de um comando (RUMSEY, F.; MCCORMICK, T., 2006).

A comunicação MIDI é uma comunicação serial, o que muda é o formato das informações e a aplicação de uma velocidade própria de transição de dados. A interface de hardware MIDI opera a 31,250 Kbaud (bits/s), assíncrona, contendo 1 bit de início, 8 bits de dados (D0 a D7) e 1 bit de parada, tendo um total de 10 bits em um período de 320ms por byte serial. O bit de início da comunicação é o 0 lógico e o bit de parada é um 1 lógico. Os bytes são enviados com o bit menos significativo (LSB) primeiro. Os circuitos do transmissor e receptor são separados internamente por opto-isolador. Os cabos devem ter o comprimento máximo de 15 metros, e devem utilizar conectores DIN de 5 pinos em suas extremidades. O cabo deve ser par trançado blindado, com a blindagem ligada ao pino 2 em ambos conectores. Caso necessário a utilização de uma cópia dos dados transmitidos, uma saída Thru pode ser adicionada (ASSOCIATION, 1995).

Na Figura 2 está disposto o esquema elétrico de um hardware de entrada e saída MIDI.

Figura 2: *Hardware* padrão MIDI



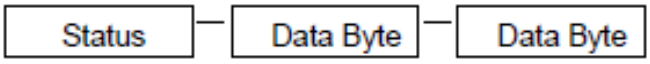
Fonte: ASSOCIATION (1995, p.2).

Mensagem MIDI é formada por grupos de palavras de 8 bits, ou seja, 1 byte, que são transmitidas em série formando instruções para dispositivos MIDI dentro do sistema. Dois tipos de bytes são definidos pela especificação MIDI: o byte de status e o byte de dados. O byte de status identifica o tipo de função MIDI será executada pelo dispositivo, usado também como codificador dos dados do canal, o byte de dados é usado para associar um valor ao evento fornecido pelo byte de status que acompanha (HUBER,2007).

Uma mensagem MIDI é composta por um byte de status seguido por um ou dois bytes de dados. São vários os tipos de mensagens MIDI. No momento em que uma tecla é pressionada em um instrumento MIDI, o controlador envia uma mensagem de Note ON na porta de saída MIDI, o dispositivo pode ser configurado para transmitir em qualquer canal entre os 16 canais MIDI disponíveis, o byte de status de mensagem envia informações de Note ON e o número do canal selecionado, este byte é seguido

por dois bytes de dados, que indicam o número da tecla pressionada e a velocidade (com que força a tecla foi pressionada) (ASSOCIATION, 2014).

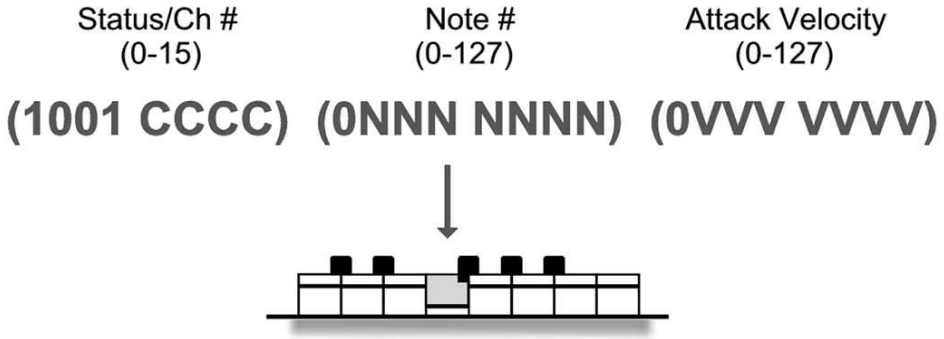
Figura 3: Palavra MIDI em *bytes.*



Fonte: ASSOCIATION (2014, p.4).

A Figura 3 expõe a palavra MIDI em bytes enquanto a Figura 4 expõe o uso dos bits da palavra MIDI.

Figura 4: Palavra MIDI em *bits.*



Fonte: HUBER (2007, p.23).

Para a transmissão de uma palavra MIDI completa são necessários três bytes, o primeiro nibble do primeiro byte é responsável por informar o status da mensagem, por exemplo o valor 0b1001 em binário, 0x9 em hexadecimal, ou seja, valor nove, indica que a nota está sendo ligada (0x9 = NOTE ON), o segundo nibble do primeiro byte representa o canal que será utilizado, são 16 canais disponíveis para comunicação, então no primeiro byte contendo 0x90 informa que o status é NOTE ON no canal 1. O segundo e terceiro byte utiliza 7 dos 8bits disponíveis, a escala de variação em MIDI varia de 0 a 127 utilizando assim apenas 7bits (0bX1111111), sendo

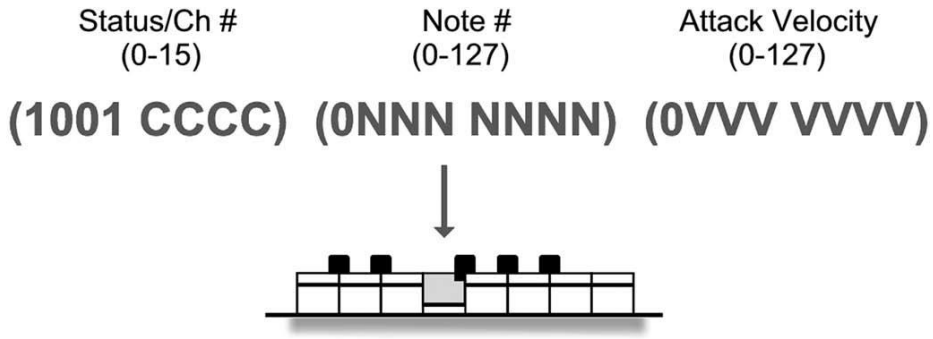
o número de 128 notas possíveis e sensibilidade em escala de 128.

O segundo byte carrega o número que indica qual nota está sendo acionada.

O terceiro byte carrega informação da intensidade de execução da nota.

A dinâmica na execução das notas em uma comunicação MIDI é apresentada na Figura 5.

Figura 5: Intensidade de execução.



Fonte: HUBER (2007, p.23).

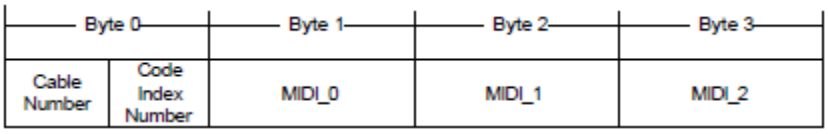
USB é uma opção de comunicação MIDI muito comum além da conexão de 5 pinos do conector DIN. Quando o USBMIDI foi desenvolvido revolucionou a maneira com que o MIDI era transmitido, aumentando a velocidade e a largura de banda do formato, também permite que dispositivos sejam alimentados pelo barramento, tirando a necessidade de uma fonte externa. Atualmente os sistemas operacionais incluem drivers que permitem ao dispositivo USBMIDI funcionar sem a necessidade de software adicional, os hardwares são projetados compatíveis com a classe, tendo seu funcionamento imediatamente após a conexão (MCGUIRE, 2020).

Ao implementar USBMIDI utiliza-se 32 bits em uma palavra MIDI, isso significa dizer que agora a palavra MIDI passou de 3 bytes para 4 bytes, tendo incluso 1 byte no início da palavra MIDI, esta inclusão trouxe largura de banda, em um cabo é possível transmitir várias linhas MIDI ao mesmo tempo. O primeiro byte é responsável pela informação de qual cabo virtual de comunicação será usado, por exemplo: um expansor MIDI pode ser montado para enviar várias linhas de informações MIDI ao

computador.

O nibble mais significativo do primeiro byte é responsável por carregar informações de canal/cabo utilizado para comunicação, são 16 canais de cabo possíveis de endereçamento. O nibble menos significativo do primeiro byte é responsável por carregar um código índice, o número do código índice indica a classificação dos bytes 1, 2 e 3 (MIDI10, 1999). A palavra MIDI representada na Figura 6 contendo informações implementada pelo protocolo USB.

Figura 6: Palavra USBMIDI.



Fonte: MIDI10 (1999, p.16).

Implementações do protocolo USB para comunicação MIDI apresentadas no Quadro 1.

Quadro 1: Classificação de número do código índice.

CIN	Tamanho do MIDI	Descrição
0x0	*1, 2 ou 3*	Diversos códigos de função. Reservado para extensões futuras.
0x1	*1, 2 ou3*	Eventos de cabo. Reservado para expansão futura.
0x2	*2*	Mensagens comuns do sistema de dois bytes, como MTC, SongSelect, etc.
0x3	*3*	Mensagens comuns do sistema de três bytes, como SPP, etc.
0x4	*3*	SysEx inicia ou continua
0x5	*1*	Mensagem comum do sistema de byte único ou SysEx termina com o seguinte byte único.
0x6	*2*	SysEx termina com os dois bytes seguintes.
0x7	*3*	SysEx termina com os três bytes seguintes.
0x8	*3*	Desliga Nota
0x9	*3*	Liga Nota
0XA	*3*	Poly-Key precionado
0XB	*3*	Mudança de controle
0XC	*2*	Mudança de programa
0XD	*2*	Pressão do Canal
0XE	3	Mudança de Pitch Bend
0XF	1	Byte único

Fonte: MIDI10 (1999, p.16).

Em uma gravação em instrumento virtual via MIDI, as teclas de um teclado funcionam como um teclado alfanumérico comum, no momento que a tecla é acionada informações são transmitidas ao computador que registra o acontecimento, no caso do teclado MIDI, são transmitidos: indicação de tecla e indicação de intensidade de acionamento da tecla, a DAW que recebe estas informações registra em uma time line: o acionamento da tecla, a intensidade de acionamento e o momento do acionamento, assim, dentro do período da gravação o computador saberá o momento exato para referenciar as informações a um determinado som, com o controle do volume e tempo de duração do referido.

Com base nestas informações, foram feitos testes utilizando comunicação MIDI com equipamentos como: teclado Yamaha SHS-10, bateria eletrônica Alesis D4 e DM5, controladora M-Audio ProjectMix IO e Hehringer BCF2000, controladora para DJ Behringer BCD3000. Varios softwares foram utilizados para testes de recepção de comandos MIDI como as DAWs: Cakewalk, Pro Tools, Reaper, Reason; e plugins como: Guitar Rig, Baterry, EZ Drummer, Waves Audio, Fabfilter.

Os testes direcionaram o projeto, partindo de controles ineficientes, passando por controles parciais até chegar em controle total.

As funções mais usadas em um estúdio são as de equalização, funções que ajustem volumes de entrada e saída, liga e desliga atuação do software e equalização, em equalização é necessário contemplar ajuste de frequência, ganho de frequência e fator Q de atuação da equalização. O equalizador escolhido para servir como controlador foi o G-EQ da Solid State Logic, muito popular nos consoles analógicos em grandes estúdios, virtualizado pela Waves Audio que fornece em formato digital entre seus plugins. O plugin do equalizador SSL G-EQ é demonstrado na Figura 7.

Figura 7: *Plugin* da Waves Audio, SSL G-EQ.

Fonte: WAVES (2021).

A escolha deste plugin foi feita com méritos de generalidade, a partir da disponibilidade de tais controles é possível fazer manipulações em vários outros modelos que, apesar de serem diferentes possuem mesmas disposições de ajustes. Este plugin contém acionamento de simulação de console analógico; acionamento e controle do filtro Low Cut de HP (Passa Alta); controle do filtro Low Shelf de LF (Frequências Graves) contendo varredura de frequência e ajuste de ganho de frequência; controle do filtro Bell de LMF (Frequência Médio Grave) contendo varredura de frequência, ajuste de ganho de frequência e fator Q de frequência, o fator Q indica o quanto o ajuste afeta as frequências ao redor da frequência ajustada, a região LMF conta com chave de controle de sensibilidade de ajuste de frequência. Com os mesmos recursos de ajustes da região LMF o G-EQ controla a região HMF (Frequência Médio Agudo), controle de High Shelf de HF (Frequências Altas) contendo varredura e ajuste do ganho de frequência; controle de liga e desliga atuação do plugin; inversão de fase do áudio; controle de ganho de saída do áudio.

A Microchip disponibiliza uma biblioteca para uso no microcontrolador PIC18F4550 com as programações básicas para uma comunicação USB, a biblioteca é totalmente funcional.

Resultados

Para que haja comunicação USB do periférico com o computador utilizando protocolo USB de áudio para comunicação MIDI, o descritor USB deve ser configurado. O descritor USB dispõe de poucos campos personalizáveis, a maioria das informações são configurações da comunicação USB, alguns dos campos que podem ser personalizados são identificação do vendedor, identificação do produto, número de série do produto e o nome do dispositivo reconhecido pelo computador.

O CHANNEL VOICE MESSAGES, conforme ASSOCIATION 1995 é formado pelos seguintes valores em hexadecimal:

0x8n – Note Off

0x9n – Note On

0xAn – Poly Key Pressure

0xBn – Control Change

0xCn – Program Change

0xDn – Channel Pressure

0xEn – Pitch Bend

Os códigos do Channel Voice Messages compõem a maior parte das informações em uma comunicação MIDI. Incluem comandos de liga e desliga nota, mudança de programa, mudança de passo no andamento, pressão pós toque e alterações do controlador (ASSOCIATION, 1995).

No primeiro byte a ser enviado ao computador usado neste projeto é o código 0xB0 que se refere ao control change (Mudança no controle), já que será feito ajustes nos controles analógicos, o segundo byte carrega o referencial ao canal a ser controlado, no caso de ser 0x00, será enviado comando para controle do canal um, se enviado 0x01, refere-se ao controle do canal dois, assim sucessivamente até o valor máximo de 120 canais sendo enviado o valor 0x78. O terceiro byte carrega informações de intensidade, no caso de acionamento de uma nota no piano, o terceiro byte informaria a intensidade em que a nota foi tocada, no caso deste projeto, o terceiro byte carregara informações de controle, sendo de valores para mínimo, intermediário e máximo, ou seja, uma varredura de 0% a 100%, no caso da mensagem MIDI, a varredura vai de 0 a 127, sendo 7 bits do terceiro byte que carregaram o comando de controle do periférico ao computador.

A rotina de programação é baseada na função de conversão ADC, a função main retem a função “leitura_analogico();”, responsável pelo tratamento de conversão ADC, na própria função fica o envio para o fluxo de entrada e saída de dados USB.

```
Void leitura analógico(void){
        static unsigned char debouce [8];
        Char LO, HI, AD, mensagem[4];

        ADCON0bits.CHS = canal;
        ADCON0bits.GO = 1;
        while(ADCON0bits.DONE);

HI = ADRESH>>1;

        if(AD != ADC_NOTA[canal]){
            if(debouce[canal] == 60){
                mensagem[0] = 0x09;
                mensagem[0] = 0xB0;
                mensagem[0] = canal;
                mensagem[0] = HI;

                data_Out(mensagem,sizeof(mensagem));

                ADC_NOTA[canal] = AD;
            }
            else{
                if(debouce[canal<60) debouce[canal]++;
```

```
                    else debouce[canal] = 0;
                }
            }
            else debouce[canal] = 0;

            if(canal >= 13) canal = 0;
            else canal++;
        }
```

Para cada controle virtual de um potenciômetro no plugin SSL-EQ, foi acrescentado um potenciômetro físico. O controle do software acontece da seguinte forma, caso o potenciômetro tenha sido ajustado para o meio de seu curso, o conversor ADC obterá o valor de 0x7FF, este valor é rotacionado três vezes para a esquerda para que o valor mais significativo esteja encaixado nos 7 bits menos significativo do byte que será enviado, ignorando o byte menos significativo da conversão ADC, com isto o resultado a ser enviado será de 0x3F para o potenciômetro em meio curso. Tendo em vista que o controle feito será do primeiro potenciômetro referenciado, a palavra MIDI enviada é, 0xB0 0x00 0x3F, que o software interpretará a informação como sendo acionamento de controle, do controle um, com o valor 63.

Como a comunicação MIDI será feita via USB, não pode ser deixado de fora o acréscimo feito para o protocolo USB, que pede uma referência de Cable Number (número de cabo) que serve para identificar qual cabo está fazendo a comunicação, este uso é necessário em um multiplicador de MIDI, equipamento capas de receber vários cabos MIDI e fazer comunicação apenas por um USB, também é acrescentado o Code Index Number (Número de índice de código), usado para direcionar o código recebido para o departamento de tratamento de código adequado, com finalidade de agilizar a comunicação. A aplicação do Cable Number e do Code Index Number no protocolo USB requer um acréscimo de um único byte. Cada informação está disposta em um nibble do byte, da forma apresentada na Figura 6.

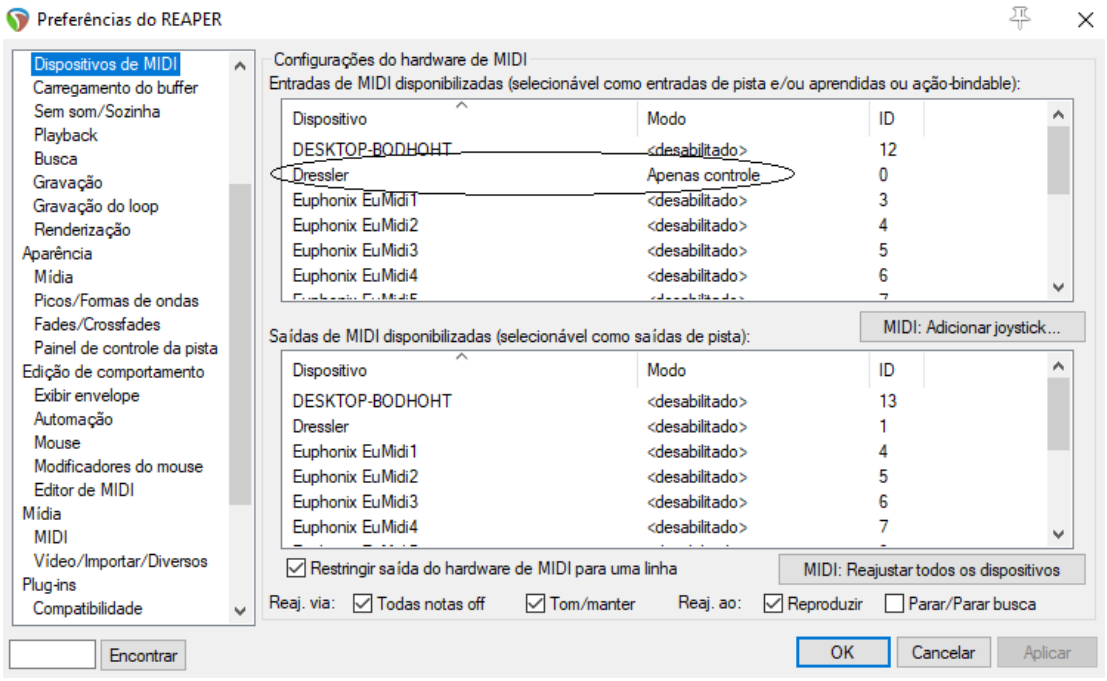
A transmissão do mesmo código ficou 0x0B 0xB0 0x00 0x3F, neste caso o primeiro byte indica que o número do cabo é um e índice de código refere-se ao control change.

Considerações Finais

O software que é totalmente controlável é o REAPER fabricado por Cockos.

A adesão do controlador ao software não é automática, requer configurações. Tendo conectado o controlador ao computador, próximo passo é conectar ao software, o REAPER reconhece todas as portas MIDI conectadas ao computador e as deixa em suspenso esperando confirmação de qual será utilizada para comunicação, as portas configuradas para comunicação recebem status de Habilitado, em nosso caso, recebe status de Apenas Controle para comunicação de entrada, não é necessário configurar a saída MIDI do REAPER para o controlador, não será tratado recebimento de dados pelo controlador. É possível ver as configurações do REAPER na Figura 8.

Figura 8: Configurações do REAPER.

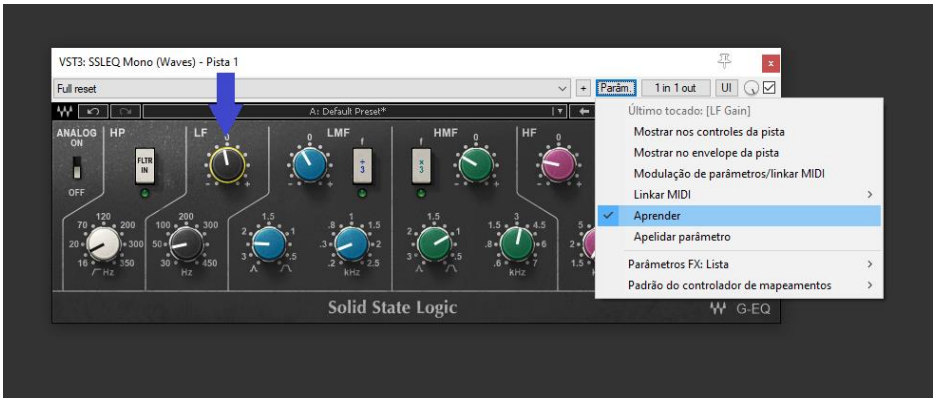


Fonte: Próprio Autor, 2021.

Após a configuração da porta MIDI de entrada para controle no REAPER, basta levar o software a aprender os comandos atribuindo controle as funções, para fazer REAPER aprender controles é preciso realizar em todos os controles individualmente o mesmo comando.

Após ter aberto o plugin que será controlado, segue-se os seguintes passos, primeiro seleciona-se o controle virtual a que será endereçado o comando, seleciona-se o a guia “Parâm” no lado superior direito do plugin e seleciona-se a função aprender, como pode ser visto na Figura 9.

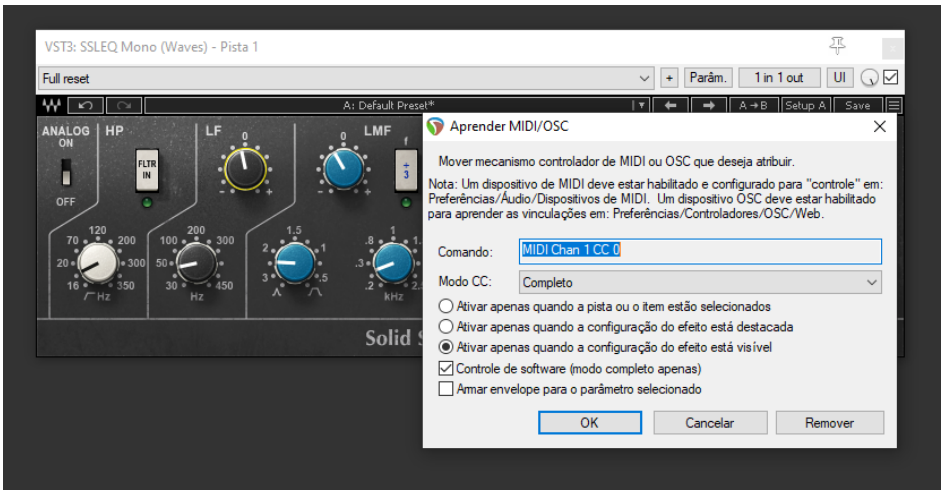
Figura 9: Comando Aprender.



Fonte: Próprio Autor, 2021.

Após ter selecionado a função aprender o REAPER abre uma aba de configuração, enquanto a aba de configuração estiver aberta o REAPER analisa qual comando do controlador é acionado para referenciar ao knob selecionado, como mostrado na Figura 10.

Figura 10: Guia de identificação de comando.



Fonte: Próprio Autor, 2021.

No caso da Figura 9, o REAPER identificou como entrada MIDI o canal de entrada 1, indicando que o control change é igual a 0xB0 e o código CC 0, indicando que o canal a ser controlado é igual a 0x00.

Repetindo estas instruções em outros plugins, o software criara um banco de dados com todas as aprendizagens feitas, no momento do uso de cada plugin as configurações estarão prontas para a utilização, sem necessidade de refazer a memorização dos comandos novamente.

Com os recursos aplicados neste projeto é possível ter controle total sobre a maioria do plugins utilizados atualmente nos estúdios, deixando-o bastante abrangente em aplicabilidade.

REFERÊNCIAS BIBLIOGRÁFICAS

ASSOCIATION, T. M. M. MIDI 1.0 Detailed Specification: Los Angeles: 1995

ASSOCIATION, T. M. M. The Complete MIDI 1.0 Detailed Specification: 3.ed. Los Angeles: 2014.

COMPAC; HEWLETT-PACKARD; INTEL; LUCENT; MICROSOFT; NEC PHILIPS. Universal serial bus specification. Revisão 2.0, 27 de abril de 2000. Disponível em: <https://www.usb.org/sites/default/files/usb_20_20211008.zip>. Acesso em: 30 Out. 2021.

HUBER, D. M. The MIDI Manual: A practical Guide to MIDI in the Project Studio. 3.ed. USA: Focal Press, 2007.

MCGUIRE, S. Modern MIDI: Sequencing and Performing Using Traditional And Mobile Tools. 2.ed. New

York: 2020.

MICROCHIP, PIC18F2458/2553/4458/4553 Data Sheet. EUA: Microchip, 2007. Disponível em: <http://ww1.microchip.com/downloads/en/devicedoc/39887b.pdf>. Acesso em: 28 Out. 2021.

MICROCHIP, MPLAB® X IDE. EUA: Microchip, 2021. Disponível em <https://www.microchip.com/en-us/development-tools-tools-and-software/mplab-x-ide#>. Acesso em 30 Out. 2021.

MICROCHIP, PICkit™ 3 Programmer/Debugger User's Guide. EUA: Microchip, 2010. Disponível em: <https://ww1.microchip.com/downloads/en/DeviceDoc/51795B.pdf> Acesso em: 30 Out. 2021.

MIDI. IN: WIKIPÉDIA: a enciclopédia livre. Disponível em: <https://pt.wikipedia.org/wiki/MIDI>. Acesso em: 31 Out. 2021.

MIDI10, Universal Serial Bus Device Class Definition for MIDI Device: EUA: 1999. Disponível em: <https://www.usb.org/sites/default/files/midi10.pdf> Acesso em: 02 Nov. 2021.

MIYADAIRA, A. N. Microcontroladores PIC18: aprenda e programe em linguagem C. 4.ed. São Paulo: Érica, 2013.

MOTA, D. M. Utilização de comunicação USB em sistemas embutidos: 2007. Monografia (Engenharia de Controle e Automação) – Universidade Federal de Ouro Preto, Ouro Preto, 2007.

RUMSEY, F.; MCCORMICK, T. Sound and Recording: An Introduction. 5.ed. USA: Focal Press, 2006.

RSPE, Avid Pro Tools | S6 M40 32-9-D Control Surface: EUA: 2021. Disponível em <https://www.rspeaudio.com/Avid-Pro-Tools-S6-M40-32-9-D-p/avid-s6-m40-32-9-d.htm> Acesso em: 27 Nov. 2021.

WAVES, SSL G-Equalizer: EUA: 2021. Disponível em <https://img.wavescdn.com/1lib/images/products/plugins/full/ssl-g-equalizer-v2.jpg> Acesso em: 14 Nov. 2021.