```
#include <MIDI.h>
#include "Controller.h"

int val1 = 0;
int lastVal1 = 0;
int val2 = 0;
int lastVal2 = 0;
int LED = 12; //the pin we connect the LED

MIDI_CREATE_DEFAULT_INSTANCE();

//***SET THE NUMBER OF CONTROLS USED*************************
//---How many buttons are connected directly to pins?---------
byte NUMBER_BUTTONS = 1;
//---How many potentiometers are connected to a multiplexer?--
byte NUMBER_MUX_POTS = 15;
//***********************************************************

Mux M1(A1, 16, true); //Analog multiplexer on Arduino analog pin A1
//*****************************************************************

//***DEFINE DIRECTLY CONNECTED BUTTONS*****************************
//Button (Pin Number, Command, Note Number, Channel, Debounce Time)
//** Command parameter 0=NOTE  1=CC  2=Toggle CC **

Button BU8(8, 2, 65, 1, 5 );
//*****************************************************************
//Add buttons used to array below like this->  Button *BUTTONS[] {&BU1, &BU2,
&BU3, &BU4, &BU5, &BU6, &BU7, &BU8};
Button *BUTTONS[] {&BU8};
//*****************************************************************


//***DEFINE POTENTIOMETERS CONNECTED TO
MULTIPLEXER*******************
//Pot::Pot(Mux mux, byte muxpin, byte command, byte control, byte channel)
//**Command parameter is for future use**

Pot MPO1(M1, 1, 0, 13, 1);
Pot MPO2(M1, 2, 0, 14, 1);
Pot MPO3(M1, 3, 0, 18, 1);
Pot MPO4(M1, 4, 0, 20, 1);
Pot MPO5(M1, 5, 0, 74, 1);
Pot MPO6(M1, 6, 0, 71, 1);
Pot MPO7(M1, 7, 0, 22, 1);
Pot MPO8(M1, 8, 0, 23, 1);
Pot MPO9(M1, 9, 0, 24, 1);
Pot MPO10(M1, 10, 0, 26, 1);
Pot MPO11(M1, 11, 0, 73, 1);
Pot MPO12(M1, 12, 0, 75, 1);
```

```
Pot MPO13(M1, 13, 0, 27, 1);
Pot MPO14(M1, 14, 0, 72, 1);
Pot MPO15(M1, 15, 0, 3, 1);
//Pot MPO16(M1, 0, 0, 55, 1);
//*****************************************************************
//Add multiplexed pots used to array below like this->  Pot *MUXPOTS[] {&MPO1,
&MPO2, &MPO3, &MPO4, &MPO5, &MPO6, &MPO7, &MPO8, &MPO9, &MPO10,
&MPO11, &MPO12, &MPO13, &MPO14, &MPO15};
Pot *MUXPOTS[] {&MPO1, &MPO2, &MPO3, &MPO4, &MPO5, &MPO6, &MPO7,
&MPO8, &MPO9, &MPO10, &MPO11, &MPO12, &MPO13, &MPO14, &MPO15};
//*****************************************************************


void setup() {
  MIDI.begin(MIDI_CHANNEL_OFF);
  pinMode(LED, OUTPUT); //set the LED pin as OUTPUT
}

void loop() {
  if (NUMBER_BUTTONS != 0) updateButtons();
  if (NUMBER_MUX_POTS != 0) updateMuxPots();
  val1 = analogRead(4)/8;      // Divide by 8 to get range of 0-127 for midi
  if (val1 != lastVal1) // If the value does not = the last value the following command is
made.
  {
  MIDI.sendControlChange(5,val1,1);}         // 176 = CC command, 1 = Which
Control, val = value read from Potentionmeter
  lastVal1 = val1;

  val2 = analogRead(5)/205;     // Control point 2
  if (val2 != lastVal2)
  {
  MIDI.sendControlChange(29,val2,1);}
  lastVal2 = val2;
}

//*****************************************************************
void updateButtons() {

  // Cycle through Button array
  for (int i = 0; i < NUMBER_BUTTONS; i = i + 1) {
    byte message = BUTTONS[i]->getValue();

    //  Button is pressed
    if (message == 0) {
      switch (BUTTONS[i]->Bcommand) {
        case 0: //Note
          MIDI.sendNoteOn(BUTTONS[i]->Bvalue, 127, BUTTONS[i]->Bchannel);
          break;
        case 1: //CC
```

```
        MIDI.sendControlChange(BUTTONS[i]->Bvalue, 127, BUTTONS[i]-
>Bchannel);
        break;
      case 2: //Toggle
        if (BUTTONS[i]->Btoggle == 0) {
          MIDI.sendControlChange(BUTTONS[i]->Bvalue, 127, BUTTONS[i]-
>Bchannel);
          digitalWrite(LED, HIGH); //write 1 or HIGH to led pin
          BUTTONS[i]->Btoggle = 1;
        }
        else if (BUTTONS[i]->Btoggle == 1) {
          MIDI.sendControlChange(BUTTONS[i]->Bvalue, 0, BUTTONS[i]->Bchannel);
          digitalWrite(LED, LOW); //write 0 or LOW to led pin
          BUTTONS[i]->Btoggle = 0;
        }
        break;
    }
   }

   //  Button is not pressed
   if (message == 1) {
     switch (BUTTONS[i]->Bcommand) {
       case 0:
         MIDI.sendNoteOff(BUTTONS[i]->Bvalue, 0, BUTTONS[i]->Bchannel);
         break;
       case 1:
         MIDI.sendControlChange(BUTTONS[i]->Bvalue, 0, BUTTONS[i]->Bchannel);
         break;
     }
   }
  }
}
//*****************************************************************
void updateMuxPots() {
  for (int i = 0; i < NUMBER_MUX_POTS; i = i + 1) {
    MUXPOTS[i]->muxUpdate();
    byte potmessage = MUXPOTS[i]->getValue();
    if (potmessage != 255) MIDI.sendControlChange(MUXPOTS[i]->Pcontrol,
potmessage, MUXPOTS[i]->Pchannel);
  }
}
```