```
//
// MUSES72320制御プログラム Ver. 1.2
//
// 0dB ~ -60dB 2dB step
//
// Arduino Nano Every SPI Pin                      MUSES72320 Pin
//   Controller In Peripheral Out : CIPO D12(30)
//   Controller Out Peripheral In : COPI D11(29)  DATA  : 18
//   SCK                          : SCK  D13(1)   CLOCK : 19
//   Chip Select Pin              : CS   D8(26)   LATCH : 20
//
// sppedMaximum : 125000    MUSES72320の最小幅4us ... 1/8^-6=125kHz
// dataOrder    : MSBFIRST
// dataMode     : SPI_MODE2
//   Clock Polarity(CPOL) : 1 アイドリング状態のクロック信号はhigh
//   Clock Phase(CPHA)    : 0 サンプリングは立ち下がりエッジ
// Chip Address : 0000      ADR2~0 : Low
//
// Lchボリュームコントロール(Att.)
//            MSB                   LSB
//            --Data--        Ctrl Chip
//    0.0dB : 00010000 =  16  0000 0000
//   -2.0dB : 00010100 =  20  0000 0000
//                      ...
//  -58.0dB : 10000100 = 132  0000 0000
//  -60.0dB : 10001000 = 136  0000 0000
//     Mute : 11111111 = 255  0000 0000
//
// Rchボリュームコントロール(Att.)
//                           Ctrl Chip
//                           0010 0000
//
// Lchボリュームコントロール(Gain)
//            MSB                   LSB
//            --Data--        Ctrl Chip
//    0.0dB : 00000000 =   0  0001 0000
//
// Rchボリュームコントロール(Gain)
//                           Ctrl Chip
//                           0011 0000
//
#include <SPI.h>
const int CS  = 8;                      // CS は D8 に接続
const int VOL = 7;                      // VOLは A7 に接続
const int LED = 2;                      // LEDは D2 に接続
int dVol, _dVol, __dVol;                // MUSES72320に送信するボリューム値，前回，
前々回
unsigned int time, _time;               // ボリューム設定時間，前回

void setup() {
  SPI.begin();                          // SPIを初期化
  SPI.beginTransaction(SPISettings(125000, MSBFIRST, SPI_MODE2));
  spiSend(0b00000000, 0b01000000);      // Lch，Rch独立制御　ゼロクロス検出回路ON
  spiSend(0b11111111, 0b00000000);      // Lchボリュームコントロール(Att.) Mute
  spiSend(0b11111111, 0b00100000);      // Rchボリュームコントロール(Att.) Mute
  spiSend(0b00000000, 0b00010000);      // Lchボリュームコントロール(Gain) 0dB
  spiSend(0b00000000, 0b00110000);      // Rchボリュームコントロール(Gain) 0dB

  pinMode(CS, OUTPUT);                  // CSを出力に設定
  pinMode(LED, OUTPUT);                 // LEDを出力に設定

  Serial.begin(9600);                   // シリアル通信を9600bpsで初期化
  Serial.println("Setup complete.");    // シリアルモニタに送信
```

```
}

void loop() {
  int aVol;

  aVol = analogRead(VOL);                 // 0~5Vの電圧を読み込み0~1023で代入
  dVol = (31 - aVol / 32) * 4 + 16;

  if (_dVol != dVol) {                    // dVolが前回の値と違う場合
    time = millis();
                                          // dVolが前々回の値と違う場合 又は 1秒以上
経過している場合
    if ((__dVol != dVol) || (time - _time) > 1000 ) {
      if (dVol > 136) {                   // dVolが136を超えている場合 Mute
        spiSend(0b11111111, 0b00000000); // Lchボリュームコントロール(Att.) Mute
        spiSend(0b11111111, 0b00100000); // Rchボリュームコントロール(Att.) Mute
        Serial.println("Volune: Mute");  // シリアルモニタに送信
      } else {
        spiSend(dVol, 0b00000000);        // Lchボリュームコントロール(Att.)
        spiSend(dVol, 0b00100000);        // Rchボリュームコントロール(Att.)
        Serial.print("Volune: ");         // シリアルモニタに送信
        Serial.print(-(dVol - 16) / 2);
        Serial.println("dB");
      }
      digitalWrite(LED, HIGH);            // 送信表示
      __dVol = _dVol;                     // 前回の値を保存
      _dVol = dVol;                       // 今回の値を保存
      _time = time;                       // 今回の時間を保存
      digitalWrite(LED, LOW);
    }
  }
}

// SPIで2byteを送信
void spiSend(byte msb, byte lsb) {
  digitalWrite(CS, LOW);                  // 通信開始の通知
  delayMicroseconds(4);                   // CLOCKセットアップ時間 min 1.6us < 4us
  SPI.transfer(msb);                      // 最初の1byteを送信
  SPI.transfer(lsb);                      // 最後の1byteを送信
  delayMicroseconds(8);                   // LATCH立ち上がりホールド時間 min 4us <
8us
  digitalWrite(CS, HIGH);                 // 通信終了の通知
}
```