HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

Euclidean rhythm sequencer made for 800 yen - modular synth self-made

71

HAGIWO
September 1, 2021 12:14 PM

I used the Arduino nano to build my own Euclidean rhythm sequencer, a modular synthesizer, so I wrote a memoir of that.

[$7] DIY eurorack modular synth Euclidean rhythm...

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

wn modular synth.

hms may be one of the hallmarks of modular synths. Rhythm machines out there don't have Euclidean rhythm sequencers. However, Eurolac has a lineup of many Euclidean rhythm sequencers.

In order to create a rhythm different from dance music, we diverted the hardware of the "6CH trigger sequencer" created in the past to create a Euclidean rhythm sequencer.

800円で作る6CH トリガーシーケンサー w/SSD1306 0.96 inch OLED

Arduinoプログラミングに挑戦しつつ、モジュラーシンセサイザー 6CHトリガーシーケンサーモジュールを自作したので、その備忘録。 背景コードの書けないシステムエンジニア脱却のために始めたプログラミングの9作...

♡ 14

HAGIWO/ハギヲ
2021/02/13 22:17

Specs of the production

Eurorack Standard 3U 6HP Power

Supply: Operates from a single standby 37mA (at5V or 12V) output 100mA (at 5V or 12V)

5V supply. Or can operate from a single 12V supply.

size

eter selection

:er change/decision

-5V) trigger input: 1CH (0-5V)

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

Two types of modes

Manual mode: Each parameter of each CH is arbitrarily set and played.

The selectable parameters are as follows:

HITS: The number of outputs in 16 steps.

OFFSET: **LIMIT** to shift

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

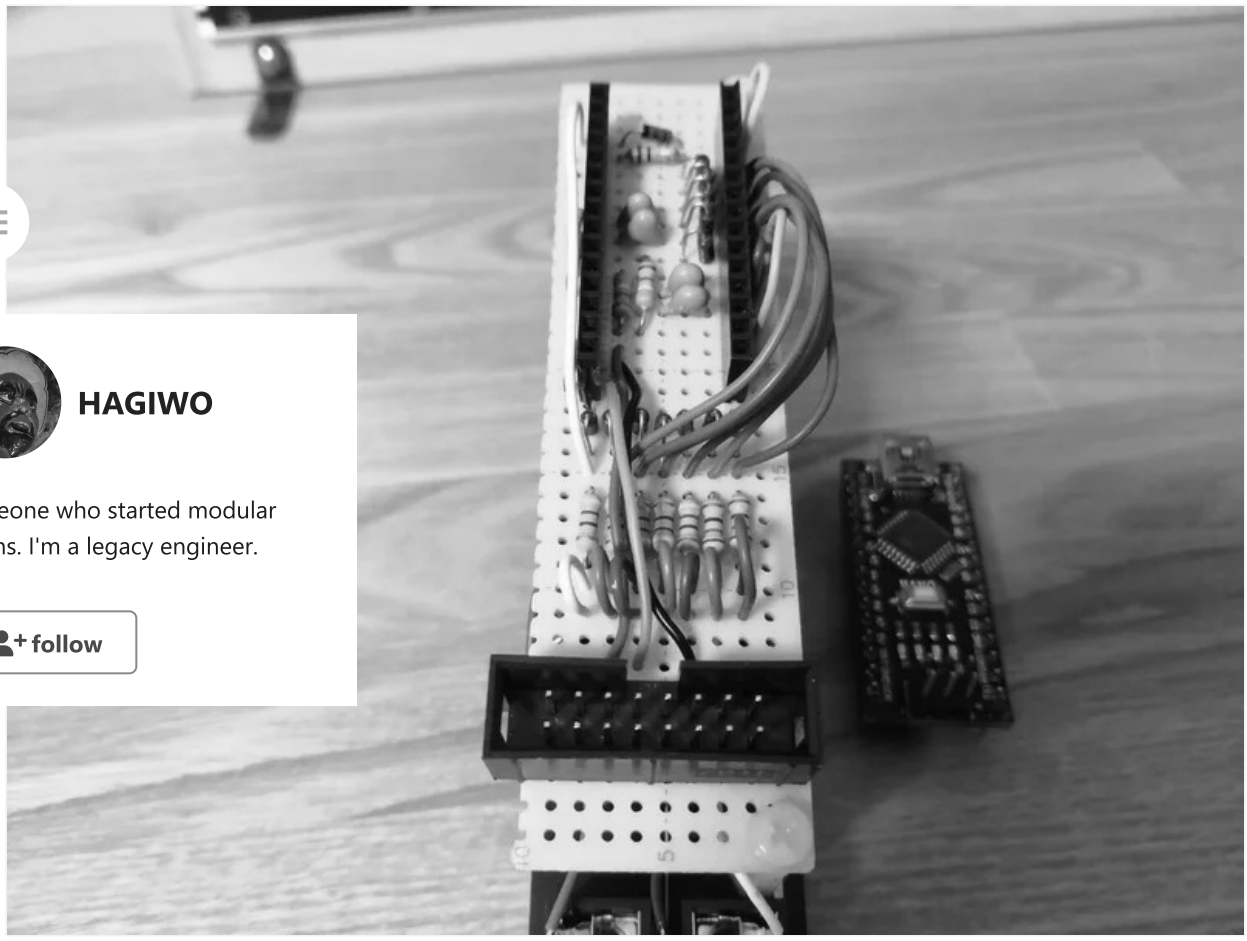
ı by an offset minute: When there is a trigger input of
ɔf times, return to the first step. For example, if you set
to the first step after 1 ~ 5 step output. It can be used in
r.

MUTE: Eliminates the trigger output of the selected CH.

RESET: Press the button to return the playback step of all CH to the first step.

Random Mode: Each parameter of each CH switches randomly every time the specified beat is reached. There is a tendency to be completely random rather than a value that is chosen by each CH.

OCCURRENCE: When the number of STEPS specified here is reached, each parameter is randomly switched. The choices are "2,4,8,16" and can be checked in the bar at the bottom left of the screen.



HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

Production cost

Total amount about 800 yen---------- Arduino nano 200 yen OLED (SSD1306) 180 yen panel, 150 yen rotary encoder 80 yen

, etc.

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

programming

About Euclidean sequences:

Euclidean rhythm is obtained by advanced calculations, but not programmatically. If it is limited to 16 steps, there are only 17 types of rhythm patterns (0 hit ~ 16 hit), so the rhythm patterns are stored in the table.

```
const static byte euc16[17][16] PROGMEM = {//euclidian rythm
 {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
 {1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
 {1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0},
 {1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0},
 {1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0},
 {1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0},
 {1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0},
 {1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0},
 {1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0},
```

```
{1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0},
{1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1},
{1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1},
{1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1},
{1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1},
{1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1},
{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0},
{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}
};
```

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

of OLED display:

ed on the screen, latency occurs in the trigger output. update of the OLED is performed immediately after the trigger output ends. It is inconvenient because the display of the screen is not updated unless there is a trigger input, but you can increase the frequency of screen update by enabling commentout on the source.

```
//    disp_reflesh = 1;//Enable while debugging.
```

OLED figure display:

Displays polygons connecting the vertices of the step where the hit is valid. If it's a 3hit, it's a triangle, if it's a 4hit, it's a square. The program that displays this polygon is as follows.

```
for (k = 0; k <= 5; k++) { //ch count
 buf_count = 0;
 for (m = 0; m < 16; m++) {
   if (offset_buf[k][m] == 1) {
     line_xbuf[buf_count] = x16[m] + graph_x[k];//store active step
     line_ybuf[buf_count] = y16[m] + graph_y[k];
     buf_count++;
   }
```

```
  }

  for (j = 0; j < buf_count - 1; j++) {
    display.drawLine(line_xbuf[j], line_ybuf[j], line_xbuf[j + 1], line_ybuf[j + 1], WHI
  }
  display.drawLine(line_xbuf[0], line_ybuf[0], line_xbuf[j], line_ybuf[j], WHITE);
}
for (j = 0; j < 16; j++) {//line_buf reset
  line_xbuf[j] = 0;
```


HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

ex coordinates in a buffer for drawing lines.

connecting the coordinates, delete the buffer.

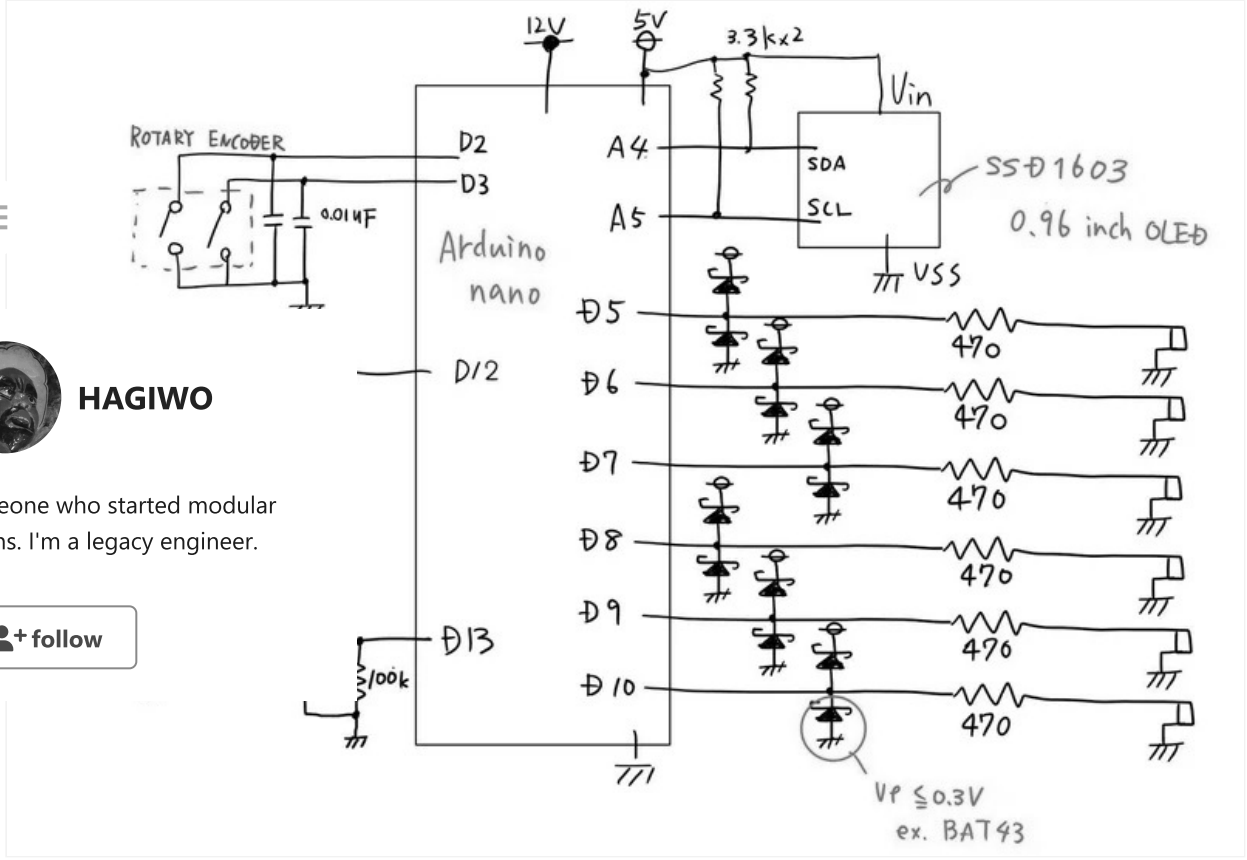
It is processed by repeating it at 1 ~ 6 CH.

Memory usage:

On the Arduino IDE, it shows that it is only using 31% of the RAM area, but if you make the program process heavier any further, the operation becomes unstable.

For example, if the number of characters displayed in OLED is increased or complicated conditional branching by the if is performed, the Arduino often does not accept operations.

Was it incompatible with the library? The cause is unknown.



HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

hardware

The circuit is the same as the 6ch trigger sequencer.

By rewriting the software, the functionality can be changed.

Publicity: Please support open source projects

In order to continue the open source project of DIY modular synths, we are recruiting patrons with a service called patreon.

I would be happy if you could help me with a cup of coffee.

We also distribute patron-only content.

HAGIWO is creating DIY eurorack modular synth esizer | Patreon

Become a patron of HAGIWO today: Get access to exclusive cont
www.patreon.com

source code

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

ere is a bad point, it will be a learning experience if you

ut if you're creating a new product based on this source

ou included a link to this blog (or YouTube).

euclid_pub.ino

12.1 KB

About file download

ダウンロード

```
//Encoder setting
#define  ENCODER_OPTIMIZE_INTERRUPTS //countermeasure of encoder noise
#include <Encoder.h>

//Oled setting
#include<Wire.h>
#include<Adafruit_GFX.h>
#include<Adafruit_SSD1306.h>

#define OLED_ADDRESS 0x3C
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

//rotery encoder
Encoder myEnc(3, 2);//use 3pin 2pin
int oldPosition  = -999;
int newPosition = -999;
int i = 0;

//push button
bool sw = 0;//push button
```

```
bool old_sw;//countermeasure of sw chattering
unsigned long sw_timer = 0;//countermeasure of sw chattering

//each channel param
byte hits[6] = { 4, 4, 5, 3, 2, 16};//each channel hits
byte offset[6] = { 0, 2, 0, 8, 3, 9};//each channele step offset
bool mute[6] = {0, 0, 0, 0, 0, 0}; //mute 0 = off , 1 = on
byte limit[6] = {16, 16, 16, 16, 16, 16};//eache channel max step
```

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow


```
16[17][16] PROGMEM = {//euclidian rythm
, 0, 0, 0, 0, 0, 0, 0, 0, 0},
, 0, 0, 0, 0, 0, 0, 0, 0, 0},
, 0, 1, 0, 0, 0, 0, 0, 0, 0},
{1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0},
{1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0},
{1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0},
{1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0},
{1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0},
{1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0},
{1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0},
{1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1},
{1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1},
{1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1},
{1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1},
{1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1},
{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0},
{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1}
};
bool offset_buf[6][16];//offset buffer , Stores the offset result

bool trg_in = 0;//external trigger in H=1,L=0
bool old_trg_in = 0;
byte playing_step[6] = {0, 0, 0, 0, 0, 0}; //playing step number , CH1,2,3,4,5,6
unsigned long gate_timer = 0;//countermeasure of sw chattering

//display param
byte select_menu = 0;//0=CH,1=HIT,2=OFFSET,3=LIMIT,4=MUTE,5=RESET,
byte select_ch = 0;//0~5 = each channel -1 , 6 = random mode
bool disp_reflesh = 1;//0=not reflesh display , 1= reflesh display , countermeasure of di

const byte graph_x[6] = {0, 40, 80, 15, 55, 95};//each chanel display offset
const byte graph_y[6] = {0, 0,  0,  32, 32, 32};//each chanel display offset

byte line_xbuf[17];//Buffer for drawing lines
byte line_ybuf[17];//Buffer for drawing lines

const byte x16[16] = {15,  21, 26, 29, 30, 29, 26, 21, 15, 9,  4,  1,  0,  1,  4,  9};//\
const byte y16[16] = {0,  1,  4,  9,  15, 21, 26, 29, 30, 29, 26, 21, 15, 9,  4,  1};//Ve
```

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

```
//random assign
byte hit_occ[6] = {0, 10, 20, 20, 40, 80}; //random change rate of occurrence
byte off_occ[6] = {10, 20, 20, 30, 40, 20}; //random change rate of occurrence
byte mute_occ[6] = {20, 20, 20, 20, 20, 20}; //random change rate of occurrence
byte hit_rng_max[6] = {0, 14, 16, 8, 9, 16}; //random change range of max
byte hit_rng_min[6] = {0, 13, 6, 1, 5, 10}; //random change range of max

unt 16 steps, the bar will increase by 1.
 4, 8, 16} ;//selectable bar
/selected bar
ount 16 steps, the bar will increase by 1.

6_SWITCHCAPVCC, 0x3C);
1);
(WHITE);
OLED_display();

 //pin mode setting
 pinMode(12, INPUT_PULLUP); //BUTTON
 pinMode(5, OUTPUT); //CH1
 pinMode(6, OUTPUT); //CH2
 pinMode(7, OUTPUT); //CH3
 pinMode(8, OUTPUT); //CH4
 pinMode(9, OUTPUT); //CH5
 pinMode(10, OUTPUT); //CH6
}

void loop() {
 old_trg_in = trg_in;
 oldPosition = newPosition;
 //-----------------Rotery encoder read----------------------
 newPosition = myEnc.read();

 if ( newPosition   < oldPosition ) {//turn left
   oldPosition = newPosition;
   //    disp_reflesh = 1;//Enable while debugging.
   if (select_menu != 0) {
     select_menu --;
   }
 }

 else if ( newPosition    > oldPosition ) {//turn right
   oldPosition = newPosition;
   //    disp_reflesh = 1;//Enable while debugging.
   select_menu ++;
 }
 if (select_ch != 6) { // not random mode
   select_menu = constrain(select_menu, 0, 5);
 }
 else  if (select_ch == 6) { // random mode
```

```
  select_menu = constrain(select_menu, 0, 1);
}

//-----------------push button----------------------

sw = 1;
if ((digitalRead(12) == 0 ) && ( sw_timer + 300 <= millis() )) { //push button on ,Logic
  sw_timer = millis();
  sw = 0;
                    = 1;//Enable while debugging.
```

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

```
                  h button on
                  u) {
                   chanel

                  >= 7) {
                  0;
    break;

  case 1: //hits
    if (select_ch != 6) { // not random mode
      hits[select_ch]++ ;
      if (hits[select_ch] >= 17) {
        hits[select_ch] = 0;
      }
    }
    else  if (select_ch == 6) { // random mode
      bar_select ++;
      if (bar_select >= 4) {
        bar_select = 0;
      }
    }
    break;

  case 2: //offset
    offset[select_ch]++ ;
    if (offset[select_ch] >= 16) {
      offset[select_ch] = 0;
    }
    break;

  case 3: //limit
    limit[select_ch]++ ;
    if (limit[select_ch] >= 17) {
      limit[select_ch] = 0;
    }

    break;

  case 4: //mute
    mute[select_ch] = !mute[select_ch] ;
    break;
```

```
    case 5: //reset
      for (k = 0; k <= 5; k++) {
        playing_step[k] = 0;
      }
      break;
  }
}
```

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

```
ffset setting----------------------
k++) { //k = 1~6ch
; i <= 15; i++) {
- offset[k]] = (pgm_read_byte(&(euc16[hits[k]][i])));


fset[k]; i++) {
 - offset[k] + i] = (pgm_read_byte(&(euc16[hits[k]][i])));
```

```
//-----------------trigger detect & output----------------------
trg_in = digitalRead(13);//external trigger in
if (old_trg_in == 0 && trg_in == 1) {
  gate_timer = millis();
  for (i = 0; i <= 5; i++) {
    playing_step[i]++;      //When the trigger in, increment the step by 1.
    if (playing_step[i] >= limit[i]) {
      playing_step[i] = 0;  //When the step limit is reached, the step is set back to 0.
    }
  }
  for (k = 0; k <= 5; k++) {//output gate signal
    if (offset_buf[k][playing_step[k]] == 1 && mute[k] == 0) {
      switch (k) {
        case 0://CH1
          digitalWrite(5, HIGH);
          break;

        case 1://CH2
          digitalWrite(6, HIGH);
          break;

        case 2://CH3
          digitalWrite(7, HIGH);
          break;

        case 3://CH4
          digitalWrite(8, HIGH);
          break;

        case 4://CH5
          digitalWrite(9, HIGH);
          break;

        case 5://CH6
```

```
          digitalWrite(10, HIGH);
          break;
        }
      }
    }
    disp_reflesh = 1;//Updates the display where the trigger was entered.If it update it c

    if (select_ch == 6) {// random mode setting
      step_cnt ++;
                        16) {
```

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow


```
                        bar_max[bar_select]) {

                        e();
  if  (gate_timer + 10 <= millis()) { //off all gate , gate time is 10msec
    digitalWrite(5, LOW);
    digitalWrite(6, LOW);
    digitalWrite(7, LOW);
    digitalWrite(8, LOW);
    digitalWrite(9, LOW);
    digitalWrite(10, LOW);
  }


  if (disp_reflesh == 1) {
    OLED_display();//reflesh display
    disp_reflesh = 0;
  }
}

void Random_change() { // when random mode and full of bar_now ,
  for (k = 1; k <= 5; k++) {

    if (hit_occ[k] >= random(1, 100)) { //hit random change
      hits[k] = random(hit_rng_min[k], hit_rng_max[k]);
    }

    if (off_occ[k] >= random(1, 100)) { //hit random change
      offset[k] = random(0, 16);
    }

    if (mute_occ[k] >= random(1, 100)) { //hit random change
      mute[k] = 1;
    }
    else if (mute_occ[k] < random(1, 100)) { //hit random change
      mute[k] = 0;
    }
  }
```

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

```
}

void OLED_display() {
 display.clearDisplay();
 //------------------------euclidean circle display-----------------
 //draw setting menu
 display.setCursor(120, 0);
 if (select_ch != 6) { // not random mode
   display.print(select_ch + 1);
                          = 6) { //random mode
                          ;

                          0, 9);
                          { // not random mode
                          ;

                          = 6) { //random mode
                          ;
 }
 display.setCursor(120, 18);
 if (select_ch != 6) { // not random mode
   display.print("O");
   display.setCursor(0, 36);
   display.print("L");
   display.setCursor(0, 45);
   display.print("M");
   display.setCursor(0, 54);
   display.print("R");
 }


 //random count square
 if (select_ch == 6) { //random mode
   //        display.drawRect(1, 32, 6, 32, WHITE);
   //    display.fillRect(1, 32, 6, 16, WHITE);
   display.drawRect(1, 62 - bar_max[bar_select] * 2, 6, bar_max[bar_select] * 2 + 2, WHI
   display.fillRect(1, 64 - bar_now * 2 , 6, bar_max[bar_select] * 2, WHITE);
 }
 //draw select triangle
 if ( select_menu == 0) {
   display.drawTriangle(113, 0, 113, 6, 118, 3, WHITE);
 }
 else  if ( select_menu == 1) {
   display.drawTriangle(113, 9, 113, 15, 118, 12, WHITE);
 }

 if (select_ch != 6) { // not random mode
   if ( select_menu == 2) {
     display.drawTriangle(113, 18, 113, 24, 118, 21, WHITE);
   }
   else  if ( select_menu == 3) {
     display.drawTriangle(12, 36, 12, 42, 7, 39, WHITE);
   }
```

```
  else  if ( select_menu == 4) {
    display.drawTriangle(12, 45, 12, 51, 7, 48, WHITE);
  }
  else  if ( select_menu == 5) {
    display.drawTriangle(12, 54, 12, 60, 7, 57, WHITE);
  }
}

//draw step dot
                    k++) { //k = 1~6ch
                    imit[k] - 1; j++) { // j = steps
                    l(x16[j] + graph_x[k], y16[j] + graph_y[k], WHITE);


                    ~16hits
                    k++) { //ch count


                    ; m++) {
    if (offset_buf[k][m] == 1) {
      line_xbuf[buf_count] = x16[m] + graph_x[k];//store active step
      line_ybuf[buf_count] = y16[m] + graph_y[k];
      buf_count++;
    }
  }

  for (j = 0; j < buf_count - 1; j++) {
    display.drawLine(line_xbuf[j], line_ybuf[j], line_xbuf[j + 1], line_ybuf[j + 1], WHI
  }
  display.drawLine(line_xbuf[0], line_ybuf[0], line_xbuf[j], line_ybuf[j], WHITE);
}
for (j = 0; j < 16; j++) {//line_buf reset
  line_xbuf[j] = 0;
  line_ybuf[j] = 0;
}

//draw hits line : 1hits
for (k = 0; k <= 5; k++) { //ch count
  buf_count = 0;
  if (hits[k] == 1) {
    display.drawLine(15 + graph_x[k], 15 + graph_y[k], x16[offset[k]] + graph_x[k], y16[
  }
}

//draw play step circle
for (k = 0; k <= 5; k++) { //ch count
  if (mute[k] == 0) { //mute on = no display circle
    if (offset_buf[k][playing_step[k]] == 0) {
      display.drawCircle(x16[playing_step[k]] + graph_x[k], y16[playing_step[k]] + graph
    }
    if (offset_buf[k][playing_step[k]] == 1) {
      display.fillCircle(x16[playing_step[k]] + graph_x[k], y16[playing_step[k]] + graph
    }
  }
```

HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow

```
    }
    display.display();
  }
```


HAGIWO

Someone who started modular synths. I'm a legacy engineer.

follow