```
#include "mbed.h"
//OLED Display library
#include "Adafruit_SSD1306.h"
//GUI images compiled as cpp files
#include "sine.cpp"
#include "saw.cpp"
#include "square.cpp"
#include "tent.cpp"

//Switch input definition
#define SW0_PIN p21
#define SW1_PIN p22
#define SW2_PIN p23
#define SW3_PIN p24

//Sampling period for the switch oscillator (us)
#define SW_PERIOD 20000


//Display interface pin definitions
#define D_MOSI_PIN p5
#define D_CLK_PIN p7
#define D_DC_PIN p8
#define D_RST_PIN p9
#define D_CS_PIN p10

//an SPI sub-class that sets up format and clock speed
class SPIPreInit : public SPI
{
public:
   SPIPreInit(PinName mosi, PinName miso, PinName clk) : SPI(mosi,miso,clk)
   {
      format(8,3);
      frequency(2000000);
   };
};

//Interrupt Service Routine prototypes (functions defined below)
void sedge0();
void sedge1();
void sedge2();
void sedge3();
void tout();
void numselect(volatile uint32_t &freq);
int cursorselect();
void sinepicker();
void direction();
void paint();



//Output for the alive LED
//DigitalOut alive(LED1);
PwmOut wave(p25);
AnalogOut aout(p18);

//External interrupt input from the switch oscillator
InterruptIn swin0(SW0_PIN);
InterruptIn swin1(SW1_PIN);
InterruptIn swin2(SW2_PIN);
InterruptIn swin3(SW3_PIN);

//Switch sampling timer
Ticker swtimer;
Ticker sinetimer;



//Registers for the switch counter, switch counter latch register and update flag
volatile uint16_t scounter0=0;
```

```
volatile uint16_t scount0=0;
volatile uint16_t swstate0=0;
volatile uint16_t scounter1=0;
volatile uint16_t scount1=0;
volatile uint16_t swstate1=0;
volatile uint16_t scounter2=0;
volatile uint16_t scount2=0;
volatile uint16_t swstate2=0;
volatile uint16_t scounter3=0;
volatile uint16_t scount3=0;
volatile uint16_t swstate3=0;
volatile uint32_t freq=0;
volatile int8_t cursor=0; //multiplier
volatile uint16_t update=0;
volatile uint16_t change0=0;
volatile uint16_t change1=0;
volatile uint16_t change2=0;
volatile uint16_t change3=0;
volatile uint16_t freqchange=0;
volatile uint16_t mode=0;
volatile float period = 0;
volatile uint16_t cnt = 0;
volatile uint16_t sampadj = 1;
volatile uint32_t SINE_PERIOD = 20000;
float lookup [64] =
{0.54900857,0.597545161,0.645142339,0.691341716,0.735698368,0.777785117,0.817196642,0.853553391,0.886505227,0.915734806,0.940960632,0.961939766,
0.978470168,0.99039264,0.997592363,1,0.997592363,0.99039264,0.978470168,0.961939766,0.940960632,0.915734806,0.886505227,0.853553391,0.817196642,
0.777785117,0.735698368,0.691341716,0.645142339,0.597545161,0.54900857,0.5,0.45099143,0.402454839,0.354857661,0.308658284,0.264301632,0.22221488
3,0.182803358,0.146446609,0.113494773,0.084265194,0.059039368,0.038060234,0.021529832,0.00960736,0.002407637,0,0.002407637,0.00960736,0.02152983
2,0.038060234,0.059039368,0.084265194,0.113494773,0.146446609,0.182803358,0.222214883,0.264301632,0.308658284,0.354857661,0.402454839,0.45099143
,0.5};
float lookupsin [64] =
{0.54900857,0.597545161,0.645142339,0.691341716,0.735698368,0.777785117,0.817196642,0.853553391,0.886505227,0.915734806,0.940960632,0.961939766,
0.978470168,0.99039264,0.997592363,1,0.997592363,0.99039264,0.978470168,0.961939766,0.940960632,0.915734806,0.886505227,0.853553391,0.817196642,
0.777785117,0.735698368,0.691341716,0.645142339,0.597545161,0.54900857,0.5,0.45099143,0.402454839,0.354857661,0.308658284,0.264301632,0.22221488
3,0.182803358,0.146446609,0.113494773,0.084265194,0.059039368,0.038060234,0.021529832,0.00960736,0.002407637,0,0.002407637,0.00960736,0.02152983
2,0.038060234,0.059039368,0.084265194,0.113494773,0.146446609,0.182803358,0.222214883,0.264301632,0.308658284,0.354857661,0.402454839,0.45099143
,0.5};
float lookupsaw [64]={0.0, 0.015873015873015872, 0.031746031746031744, 0.047619047619047616, 0.06349206349206349, 0.07936507936507936,
0.09523809523809523, 0.1111111111111111, 0.12698412698412698, 0.14285714285714285, 0.15873015873015872, 0.1746031746031746,
0.19047619047619047, 0.20634920634920634, 0.2222222222222222, 0.23809523809523808, 0.25396825396825395, 0.2698412698412698,
0.2857142857142857, 0.30158730158730157, 0.31746031746031744, 0.3333333333333333, 0.3492063492063492, 0.36507936507936506,
0.38095238095238093, 0.3968253968253968, 0.4126984126984127, 0.42857142857142855, 0.4444444444444444, 0.4603174603174603, 0.47619047619047616,
0.49206349206349204, 0.5079365079365079, 0.5238095238095238, 0.5396825396825397, 0.5555555555555556, 0.5714285714285714, 0.5873015873015873,
0.6031746031746031, 0.6190476190476191, 0.6349206349206349, 0.6507936507936508, 0.6666666666666666, 0.6825396825396826, 0.6984126984126984,
0.7142857142857143, 0.7301587301587301, 0.746031746031746, 0.7619047619047619, 0.7777777777777778, 0.7936507936507936, 0.8095238095238095,
0.8253968253968254, 0.8412698412698413, 0.8571428571428571, 0.873015873015873, 0.8888888888888888, 0.9047619047619048, 0.9206349206349206,
0.9365079365079365, 0.9523809523809523, 0.9682539682539683, 0.9841269841269841, 1.0};
float lookuptent [64]={0.0, 0.03225806451612903, 0.06451612903225806, 0.0967741935483871, 0.12903225806451613, 0.16129032258064516,
0.1935483870967742, 0.22580645161290322, 0.25806451612903225, 0.2903225806451613, 0.3225806451612903, 0.3548387096774194, 0.3870967741935484,
0.41935483870967744, 0.45161290322580644, 0.4838709677419355, 0.5161290322580645, 0.5483870967741935, 0.5806451612903226, 0.6129032258064516,
0.6451612903225806, 0.6774193548387096, 0.7096774193548387, 0.7419354838709677, 0.7741935483870968, 0.8064516129032258, 0.8387096774193549,
0.8709677419354839, 0.9032258064516129, 0.9354838709677419, 0.967741935483871, 1.0, 1.0, 0.967741935483871, 0.9354838709677419,
0.9032258064516129, 0.8709677419354839, 0.8387096774193549, 0.8064516129032258, 0.7741935483870968, 0.7419354838709677, 0.7096774193548387,
0.6774193548387096, 0.6451612903225806, 0.6129032258064516, 0.5806451612903226, 0.5483870967741935, 0.5161290322580645, 0.4838709677419355,
0.45161290322580644, 0.41935483870967744, 0.3870967741935484, 0.3548387096774194, 0.3225806451612903, 0.2903225806451613, 0.25806451612903225,
0.22580645161290322, 0.1935483870967742, 0.16129032258064516, 0.12903225806451613, 0.0967741935483871, 0.06451612903225806,
0.03225806451612903, 0.0};
volatile uint16_t xpos=0;
volatile uint16_t ypos=0;
volatile uint8_t colour=1;

//Initialise SPI instance for communication with the display
SPIPreInit gSpi(D_MOSI_PIN,NC,D_CLK_PIN); //MOSI,MISO,CLK

//Initialise display driver instance
Adafruit_SSD1306_Spi gOled1(gSpi,D_DC_PIN,D_RST_PIN,D_CS_PIN,64,128); //SPI,DC,RST,CS,Height,Width

int main() {
  //Initialisation
```

```
gOled1.setRotation(2); //Set display rotation

//Attach switch oscillator counter ISR to the switch input instance for a rising edge
swin0.rise(&sedge0);
swin1.rise(&sedge1);
swin2.rise(&sedge2);
swin3.rise(&sedge3);

//Attach switch sampling timer ISR to the timer instance with the required period
swtimer.attach_us(&tout, SW_PERIOD);
sinetimer.attach_us(&sinepicker, SINE_PERIOD);


//Write some sample text
//gOled1.printf("%ux%u OLED Display\r\n", gOled1.width(), gOled1.height());
gOled1.clearDisplay();

//gOled1.invertDisplay(true);

//Main loop
while(1)
{
  //Has the update flag been set?
  if (update) {
    //Clear the update flag
    update = 0;


    gOled1.clearDisplay();

    if(cursor==-5){
      gOled1.setTextCursor(0,0);
      if(mode==0){
        gOled1.drawBitmap(0, 0, Square, 128, 36, 1);
      }
      if(mode==1){
        gOled1.drawBitmap(0, 0, Sine, 104, 48, 1);
      }
      if(mode==2){
        gOled1.drawBitmap(0, 0, Saw, 128, 40, 1);
      }
      if(mode==3){
        gOled1.drawBitmap(0, 0, Tent, 128, 48, 1);
      }
    }
    else{
      gOled1.setTextCursor(5,0);

      gOled1.printf("\n\n <");
      gOled1.printf("    >");
      gOled1.printf("    ^");
      gOled1.printf("    v");

      //Set text cursor
      gOled1.setTextCursor(5,0);

      //Write the latest switch osciallor count
      //gOled1.printf("\n%05u  ",scount0);
      gOled1.printf("%01u  ",swstate0);
      //gOled1.printf("\n%05u  ",scount1);
      gOled1.printf("   ");
      gOled1.printf("%01u  ",swstate1);
      //gOled1.printf("\n%05u  ",scount2);
      gOled1.printf("   ");
      gOled1.printf("%01u  ",swstate2);
      //gOled1.printf("\n%05u  ",scount3);
      gOled1.printf("   ");
      gOled1.printf("%01u  ",swstate3);

    }
```

```
        gOled1.setTextCursor(15,30);
        gOled1.printf("Frequency");
        gOled1.setTextCursor(15,43);
        gOled1.printf("%09u  ",freq);

        gOled1.setTextCursor(87,30);
        gOled1.printf("Mode");
        gOled1.setTextCursor(93,43);
        gOled1.printf("%01u  ",mode);

        gOled1.setTextCursor((63-(6*cursor)),50);
        gOled1.printf(" ");


        numselect(freq); //CHECK ************************************************************

//          period = (1.0/((float)freq + 20.0));


        if(freqchange==1){

           period = (1.0/((float)freq));
           wave.period(period);
           wave.write(0.50f);
           if((mode==1)||(mode==2)){

              if(mode==1){
                  memcpy(lookup, lookupsin, 64);
              }
              if(mode==2){
                  memcpy(lookup, lookupsaw, 64);
              }
              if(mode==3){
                  memcpy(lookup, lookuptent, 64);
              }

              SINE_PERIOD = (int)floor((1000000*period*sampadj)/64);
              sinetimer.attach_us(&sinepicker, SINE_PERIOD);
              if(freq<900){
                 sampadj=1;
              }
              else if(freq<1400){
                 sampadj=2;
              }
              else if(freq<2700){
                 sampadj=3;
              }
              else if(freq<3100){
                 sampadj=4;
              }
              else{
                 sampadj=5;
              }
           }
           else{
              SINE_PERIOD = 20000;
              sinetimer.attach_us(&sinepicker, SINE_PERIOD);
           }

        }

        gOled1.setTextCursor((63-(6*cursor)),50);
        gOled1.printf("-");

        //Copy the display buffer to the display
        gOled1.display();
```

```
        //Toggle the alive LED
//          alive = !alive;


      }



      if(swstate0==1 && swstate3==1){
         gOled1.clearDisplay();
         while(!(swstate0==1 && swstate1==1 && swstate2==1 && swstate3==1)){
         //Has the update flag been set?
         if (update) {
            //Clear the update flag
            if(swstate3==1 && swstate2==1){
               gOled1.clearDisplay();
            }
            if(swstate0==1 && swstate1==1 && change0==0 && change1==1){
               if(colour==1){
                  colour=0;
               }
               else{
                  colour=1;
               }
            }
            update = 0;

            direction();
            paint();

            gOled1.display();
         }

         }
      }
   }
}


//Interrupt Service Routine for rising edge on the switch oscillator input
void sedge0() {
   //Increment the edge counter
   scounter0++;
}
void sedge1() {
   //Increment the edge counter
   scounter1++;
}
void sedge2() {
   //Increment the edge counter
   scounter2++;
}
void sedge3() {
   //Increment the edge counter
   scounter3++;
}

//Interrupt Service Routine for the switch sampling timer
void tout() {

   //Read the edge counter into the output register
   scount0 = scounter0;
   scount1 = scounter1;
   scount2 = scounter2;
   scount3 = scounter3;

   if(scount0 < 1350){
      swstate0 = 1;
   }
```

```
    if(scount0 > 1400){
      swstate0 = 0;
    }
    if(scount1 < 1450){
      swstate1 = 1;
    }
    if(scount1 > 1500){
      swstate1 = 0;
    }
    if(scount2 < 1350){
      swstate2 = 1;
    }
    if(scount2 > 1400){
      swstate2 = 0;
    }
    if(scount3 < 1550){
      swstate3 = 1;
    }
    if(scount3 > 1600){
      swstate3 = 0;
    }
    //Reset the edge counter
    scounter0 = 0;
    scounter1 = 0;
    scounter2 = 0;
    scounter3 = 0;
    //Trigger a display update in the main loop
    update = 1;
}

void numselect(volatile uint32_t &freq){

    if(swstate0==1 && change0==0){
      if((cursor != 8) && (cursor != -5)){
        cursor++;
        change0 = 1;
      }
      if(cursor == -5){
        cursor = 0;
        change0 = 1;
      }
    }
    if(swstate1==1 && change1==0){
      if((cursor != 0) && (cursor != -5)){
        cursor--;
        change1 = 1;
      }
      else if(cursor == 0){
        cursor = -5;
        change1 = 1;
      }
    }
    if(swstate2==1 && change2==0){
      if(cursor==-5){
        mode++;
        change2 = 1;
        if((mode==0)||(mode==1)){
          freq = 0;
        }
      }
      else{
        freq = freq + cursorselect();
        change2 = 1;
      }
    }
    if(swstate3==1 && change3==0){
      if(cursor==-5){
        mode--;
        change3 = 1;
        if(mode==0){
```

```
            freq = 0;
          }
        }
        else{
          freq = freq - cursorselect();
          change3 = 1;
        }
      }
      if(mode>3){
        mode=0;
      }
      if(freq>999999999){
        freq = 0;
      }
      if(swstate0==0){
        change0=0;
      }
      if(swstate1==0){
        change1=0;
      }
      if(swstate2==0){
        change2=0;
      }
      if(swstate3==0){
        change3=0;
      }
      if((change0 || change1 || change2 || change3) == 1){
        freqchange=1;
      }
      else{
        freqchange=0;
      }
    }

    int cursorselect(){
      if(cursor==0){
        return 1;
      }
      if(cursor==1){
        return 10;
      }
      if(cursor==2){
        return 100;
      }
      if(cursor==3){
        return 1000;
      }
      if(cursor==4){
        return 10000;
      }
      if(cursor==5){
        return 100000;
      }
      if(cursor==6){
        return 1000000;
      }
      if(cursor==7){
        return 10000000;
      }
      if(cursor==8){
        return 100000000;
      }
      return 0;
    }

    void sinepicker(){
      if(cnt<(64-sampadj)){
        cnt=cnt+sampadj;
      }
      else{
```

```
    cnt=0;
  }
  aout=lookup[cnt];
}


void direction(){
  if(swstate0==1){
    if(xpos==0){
      xpos=126;
    }
    xpos--;
    change0 = 1;

  }
  if(swstate1==1){
    xpos++;
    change1 = 1;
    if(xpos==126){
      xpos=0;
    }
  }
  if(swstate2==1){
    if(ypos==0){
      ypos=64;
    }
    ypos--;
    change2 = 1;

  }
  if(swstate3==1){
    ypos++;
    change3 = 1;
    if(ypos==64){
      ypos=0;
    }
  }

  if(swstate0==0){
    change0=0;
  }
  if(swstate1==0){
    change1=0;
  }
  if(swstate2==0){
    change2=0;
  }
  if(swstate3==0){
    change3=0;
  }


}

void paint(){

  gOled1.drawPixel(xpos, ypos, colour);
}
```