Internet of Things
class 12

LCD Display

OLED Display Module

- 1.3 Inch 7Pin OLED (128x64)
  - Driver IC: SH1106
  - Voltage: 3.3 ~ 5V
  - Interface: 4Wire-SPI

<Pin Configuration>

GND

VCC

CLK -- Clock

MOSI -- ESP32 -> OLED Module Data/Command

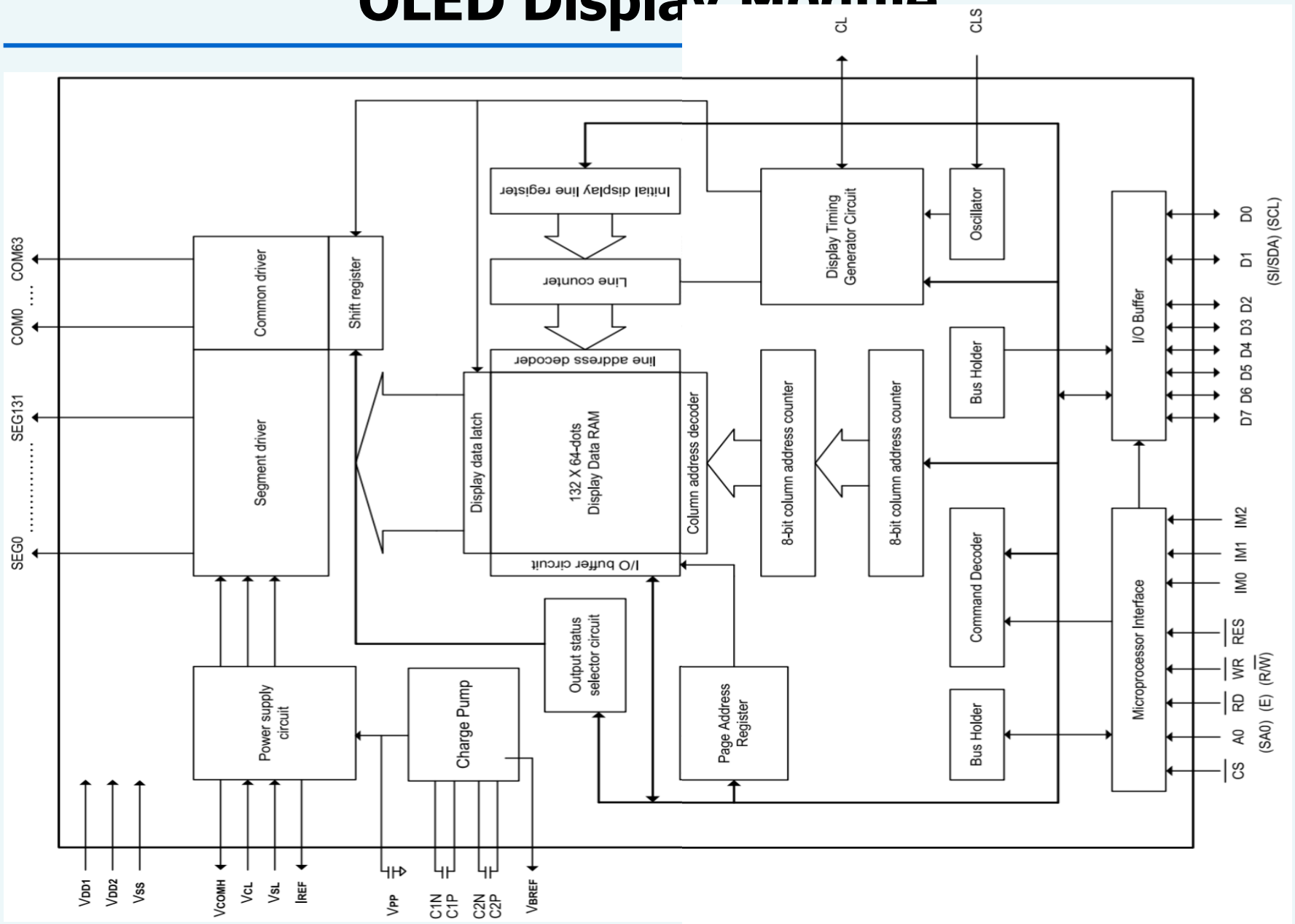
RES -- Reset

DC --- Data / Command (A0)

CS --- Chip Select

C.B.Choi gen1223@kau.ac.kr

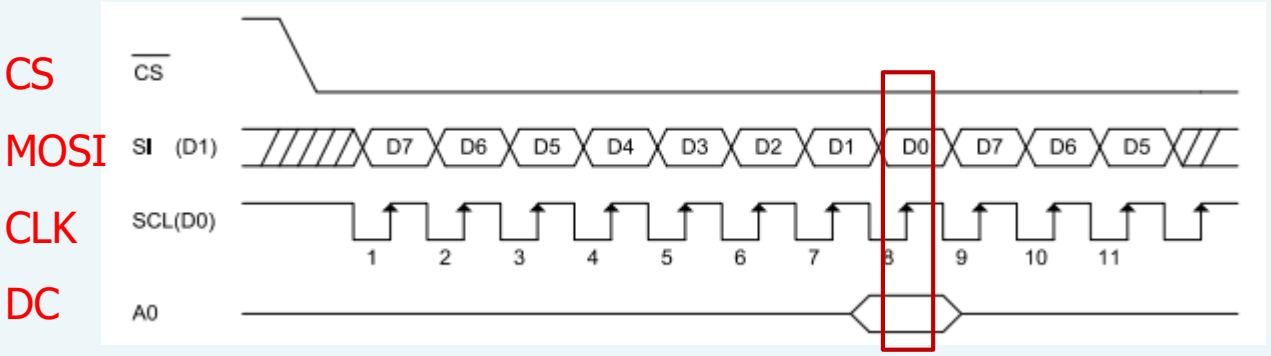
OLED Display Module

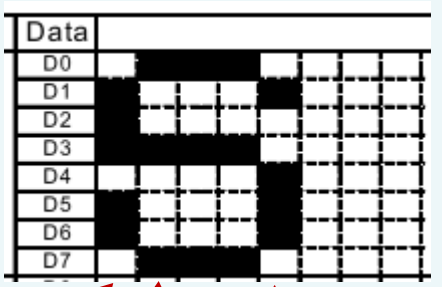


C.B.Choi gen1223@kau.ac.kr

OLED Display Module

- ## 4Wire Serial Interface
  - Initialized when CS is high
  - A Falling edge on CS enables Data Transmission
  - SI is shifted into 8bit Shift Register on every rising edge of SCL (D7..D0)
  - A0 is sampled on every 8th clock
  - Data byte in shift register is written to Display RAM (A0 = 1)
  - or to Command Register (A0 = 0)



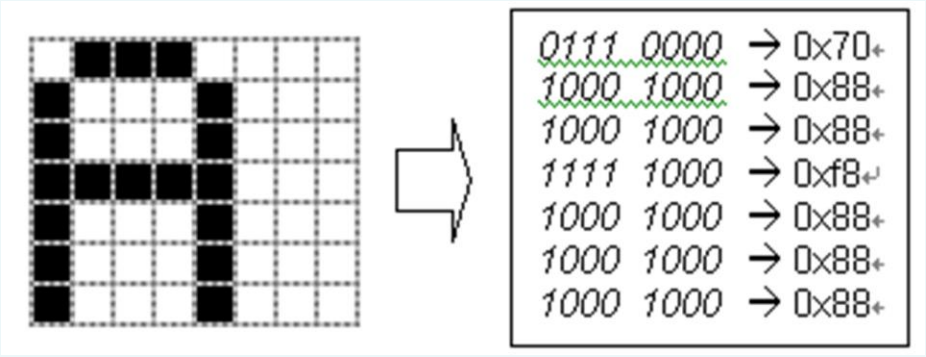


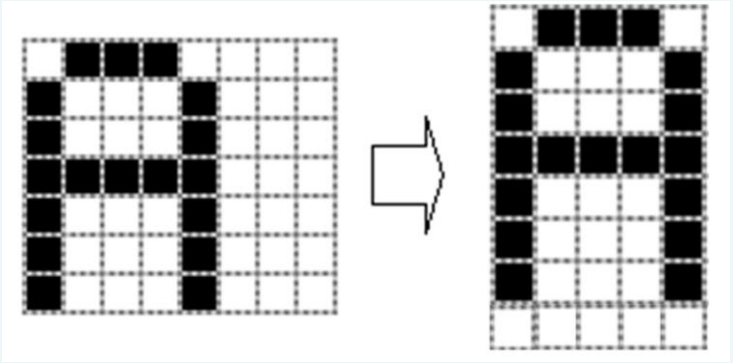
01101110 10001001 .. 01110010

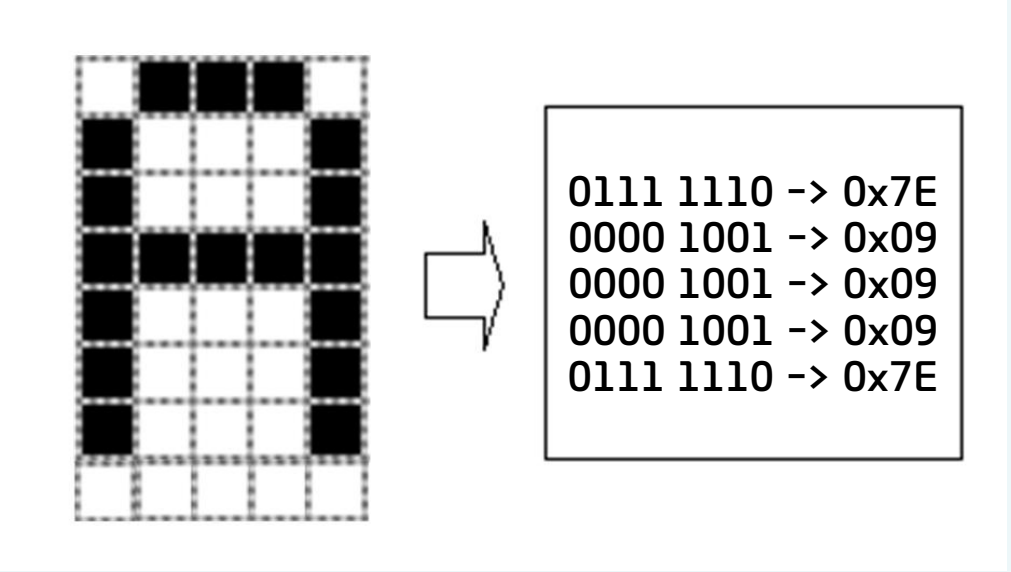
C.B.Choi gen1223@kau.ac.kr

OLED Display Module

- ## Handling Fonts
  - Example: 5 x 7 pixel

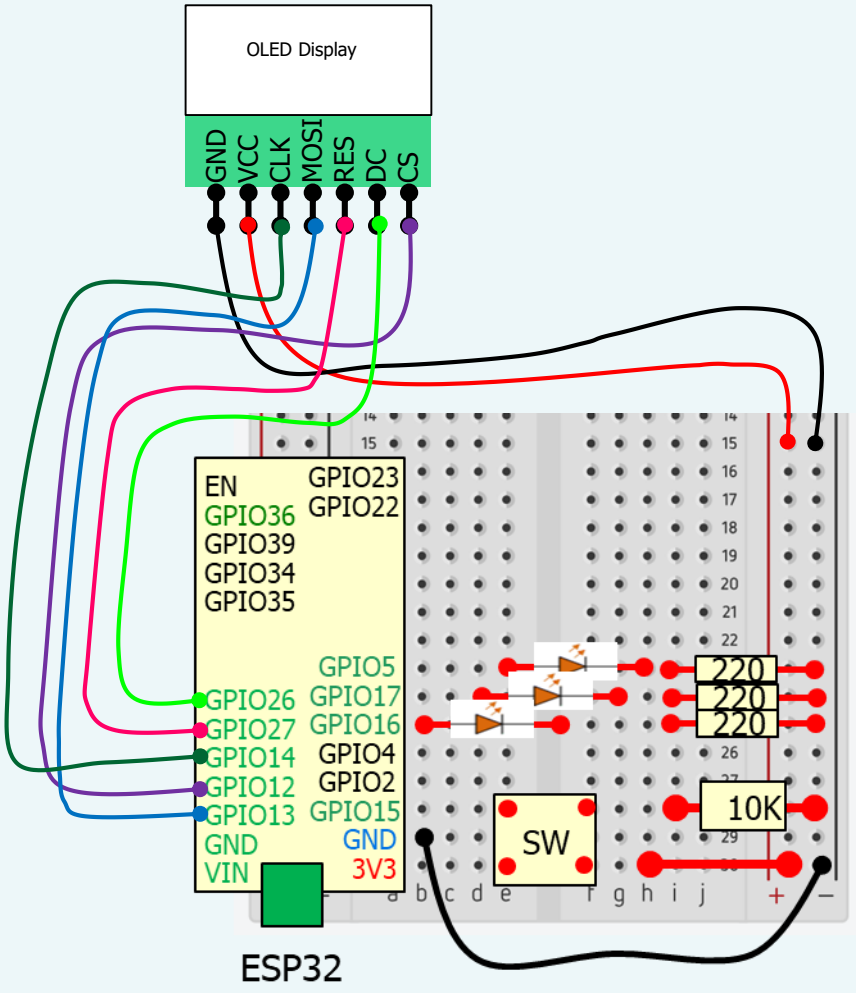




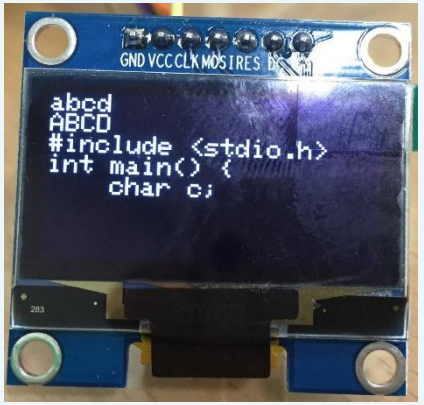


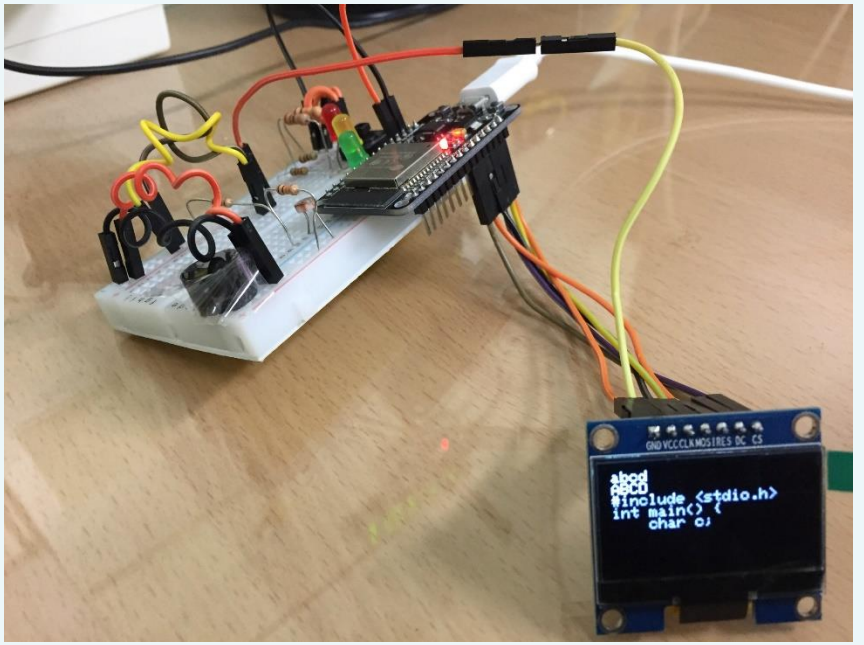
OLED Display Module

- Using SW SPI of ESP32



MOSI: GPIO13
CLK: GPIO14
DC: GPIO26
CS: GPIO12
RESET: GPIO27





C.B.Choi gen1223@kau.ac.kr

OLED Display Module

- Install Libraries
  - Adafruit GFX Library:

    https://github.com/adafruit/Adafruit-GFX-Library
  - SH1106 OLED Library:

    https://github.com/nhatuan84/esp32-sh1106-oled

OLED Display Module- Test Code

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SH1106.h>

/* If using software SPI (the default case): */
#define OLED_MOSI  13 // 9
#define OLED_CLK   14 //10
#define OLED_DC    26 //11
#define OLED_CS    12 //12
#define OLED_RESET 27 //13
Adafruit_SH1106 display(OLED_MOSI, OLED_CLK, OLED_DC, OLED_RESET, OLED_CS);


/* Uncomment this block to use hardware SPI
#define OLED_DC     26
#define OLED_CS     15      // HSPI
#define OLED_RESET  27
Adafruit_SH1106 display(OLED_DC, OLED_RESET, OLED_CS); */

#define NUMFLAKES 10
#define XPOS 0
#define YPOS 1
#define DELTAY 2
```

OLED Display Module- Test Code

```
#define LOGO16_GLCD_HEIGHT 16
#define LOGO16_GLCD_WIDTH  16
static const unsigned char PROGMEM logo16_glcd_bmp[] =
{ B00000000, B11000000,
  B00000001, B11000000,
  B00000001, B11000000,
  B00000011, B11100000,
  B11110011, B11100000,
  B11111110, B11111000,
  B01111110, B11111111,
  B00110011, B10011111,
  B00011111, B11111100,
  B00001101, B01110000,
  B00011011, B10100000,
  B00111111, B11100000,
  B00111111, B11110000,
  B01111100, B11110000,
  B01110000, B01110000,
  B00000000, B00110000 };

#if (SH1106_LCDHEIGHT != 64)
#error("Height incorrect, please fix Adafruit_SH1106.h!");
#endif
```

OLED Display Module- Test Code

```
void setup()   {
  Serial.begin(115200);

  // by default, we'll generate the high voltage from the 3.3v line internally! (neat!)
  display.begin(SH1106_SWITCHCAPVCC);

  // Show image buffer on the display hardware.
  // Since the buffer is intialized with an Adafruit splashscreen
  // internally, this will display the splashscreen.
  display.display();
  delay(2000);

  // Clear the buffer.
  display.clearDisplay();

  // draw a single pixel
  display.drawPixel(10, 10, WHITE);
  // Show the display buffer on the hardware.
  // NOTE: You _must_ call display after making any drawing commands
  // to make them visible on the display hardware!
  display.display();
  delay(2000);
  display.clearDisplay();
```

OLED Display Module- Test Code

```
// draw many lines
testdrawline();
display.display();
delay(2000);
display.clearDisplay();

// draw rectangles
testdrawrect();
display.display();
delay(2000);
display.clearDisplay();

// draw multiple rectangles
testfillrect();
display.display();
delay(2000);
display.clearDisplay();

// draw mulitple circles
testdrawcircle();
display.display();
delay(2000);
display.clearDisplay();
```

OLED Display Module- Test Code

```
// draw a white circle, 10 pixel radius
display.fillCircle(display.width()/2, display.height()/2, 10, WHITE);
display.display();
delay(2000);
display.clearDisplay();

testdrawroundrect();
delay(2000);
display.clearDisplay();

testfillroundrect();
delay(2000);
display.clearDisplay();

testdrawtriangle();
delay(2000);
display.clearDisplay();

testfilltriangle();
delay(2000);
display.clearDisplay();
```

OLED Display Module- Test Code

```
// draw the first ~12 characters in the font
testdrawchar();
display.display();
delay(2000);
display.clearDisplay();

// draw scrolling text
/*testscrolltext();
delay(2000);
display.clearDisplay();*/

// text display tests
display.setTextSize(1);
display.setTextColor(WHITE);
display.setCursor(0,0);
display.println("Hello, world!");
display.setTextColor(BLACK, WHITE); // 'inverted' text
display.println(3.141592);
display.setTextSize(2);
display.setTextColor(WHITE);
display.print("0x"); display.println(0xDEADBEEF, HEX);
display.display();
delay(2000);
```

OLED Display Module- Test Code

```
  // miniature bitmap display
  display.clearDisplay();
  display.drawBitmap(30, 16,  logo16_glcd_bmp, 16, 16, 1);
  display.display();

  // invert the display
  display.invertDisplay(true);
  delay(1000);
  display.invertDisplay(false);
  delay(1000);

  // draw a bitmap icon and 'animate' movement
  testdrawbitmap(logo16_glcd_bmp, LOGO16_GLCD_HEIGHT, LOGO16_GLCD_WIDTH);

  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,0);

}
```

OLED Display Module- Test Code

```
void loop() {
      char ch;
      if (Serial.available() > 0) {
        ch = Serial.read();
        if (ch == '0') {
            display.clearDisplay();
            display.setCursor(0,0);
        }
        else if (ch >= '1' && ch < '9')
            display.setTextSize(ch-'0');
        else
            display.print(ch);
        display.display();
    }
}
```

OLED Display Module- Test Code

```
void testdrawbitmap(const uint8_t *bitmap, uint8_t w, uint8_t h) {
  uint8_t icons[NUMFLAKES][3];

  // initialize
  for (uint8_t f=0; f< NUMFLAKES; f++) {
    icons[f][XPOS] = random(display.width());
    icons[f][YPOS] = 0;
    icons[f][DELTAY] = random(5) + 1;

    Serial.print("x: ");
    Serial.print(icons[f][XPOS], DEC);
    Serial.print(" y: ");
    Serial.print(icons[f][YPOS], DEC);
    Serial.print(" dy: ");
    Serial.println(icons[f][DELTAY], DEC);
  }

//  while (1) {
    for (int i = 0; i < 20; i++) {
      // draw each icon
      for (uint8_t f=0; f< NUMFLAKES; f++) {
        display.drawBitmap(icons[f][XPOS], icons[f][YPOS], logo16_glcd_bmp, w, h, WHITE);
      }
      display.display();
      delay(200);
```

OLED Display Module- Test Code

```
        // then erase it + move it
      for (uint8_t f=0; f< NUMFLAKES; f++) {
       display.drawBitmap(icons[f][XPOS], icons[f][YPOS],  logo16_glcd_bmp, w, h, BLACK);
       // move it
       icons[f][YPOS] += icons[f][DELTAY];
       // if its gone, reinit
       if (icons[f][YPOS] > display.height()) {
                        icons[f][XPOS] = random(display.width());
                        icons[f][YPOS] = 0;
                        icons[f][DELTAY] = random(5) + 1;
       }
      }
    }
}


void testdrawchar(void) {
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0,0);

  for (uint8_t i=0; i < 168; i++) {
    if (i == '\n') continue;
    display.write(i);
    if ((i > 0) && (i % 21 == 0))
      display.println();
  }
  display.display();
}
```


C.B.Choi gen1225@kau.ac.kr

OLED Display Module- Test Code

```
void testdrawcircle(void) {
  for (int16_t i=0; i<display.height(); i+=2) {
    display.drawCircle(display.width()/2, display.height()/2, i, WHITE);
    display.display();
  }
}

void testfillrect(void) {
  uint8_t color = 1;
  for (int16_t i=0; i<display.height()/2; i+=3) {
    // alternate colors
    display.fillRect(i, i, display.width()-i*2, display.height()-i*2, color%2);
    display.display();
    color++;
  }
}

void testdrawtriangle(void) {
  for (int16_t i=0; i<min(display.width(),display.height())/2; i+=5) {
    display.drawTriangle(display.width()/2, display.height()/2-i,
                  display.width()/2-i, display.height()/2+i,
                  display.width()/2+i, display.height()/2+i, WHITE);
    display.display();
  }
}
```

OLED Display Module- Test Code

```
void testfilltriangle(void) {
  uint8_t color = WHITE;
  for (int16_t i=min(display.width(),display.height())/2; i>0; i-=5) {
    display.fillTriangle(display.width()/2, display.height()/2-i,
                         display.width()/2-i, display.height()/2+i,
                         display.width()/2+i, display.height()/2+i, WHITE);
    if (color == WHITE) color = BLACK;
    else color = WHITE;
    display.display();
  }
}

void testdrawroundrect(void) {
  for (int16_t i=0; i<display.height()/2-2; i+=2) {
    display.drawRoundRect(i, i, display.width()-2*i, display.height()-2*i, display.height()/4, WHITE);
    display.display();
  }
}

void testfillroundrect(void) {
  uint8_t color = WHITE;
  for (int16_t i=0; i<display.height()/2-2; i+=2) {
    display.fillRoundRect(i, i, display.width()-2*i, display.height()-2*i, display.height()/4, color);
    if (color == WHITE) color = BLACK;
    else color = WHITE;
    display.display();
  }
}
```

OLED Display Module- Test Code

```
void testdrawrect(void) {
  for (int16_t i=0; i<display.height()/2; i+=2) {
    display.drawRect(i, i, display.width()-2*i, display.height()-2*i, WHITE);
    display.display();
  }
}

void testdrawline() {
  for (int16_t i=0; i<display.width(); i+=4) {
    display.drawLine(0, 0, i, display.height()-1, WHITE);
    display.display();
  }
  for (int16_t i=0; i<display.height(); i+=4) {
    display.drawLine(0, 0, display.width()-1, i, WHITE);
    display.display();
  }
  delay(250);

  display.clearDisplay();
  for (int16_t i=0; i<display.width(); i+=4) {
    display.drawLine(0, display.height()-1, i, 0, WHITE);
    display.display();
  }
  for (int16_t i=display.height()-1; i>=0; i-=4) {
    display.drawLine(0, display.height()-1, display.width()-1, i, WHITE);
    display.display();
  }
  delay(250);
```


C.B.Choi gen1225@kau.ac.kr

OLED Display Module- Test Code

```
  display.clearDisplay();
  for (int16_t i=display.width()-1; i>=0; i-=4) {
    display.drawLine(display.width()-1, display.height()-1, i, 0, WHITE);
    display.display();
  }
  for (int16_t i=display.height()-1; i>=0; i-=4) {
    display.drawLine(display.width()-1, display.height()-1, 0, i, WHITE);
    display.display();
  }
  delay(250);

  display.clearDisplay();
  for (int16_t i=0; i<display.height(); i+=4) {
    display.drawLine(display.width()-1, 0, 0, i, WHITE);
    display.display();
  }
  for (int16_t i=0; i<display.width(); i+=4) {
    display.drawLine(display.width()-1, 0, i, display.height()-1, WHITE);
    display.display();
  }
  delay(250);
}
```