Tiny4kOLED

This document provides an overview of the `Tiny4kOLED` library, which is specifically designed for use with small `OLED` displays. The library enables to interact with `OLED` screens, displaying text, graphics, and more, with minimal code. It is particularly useful in `Arduino` projects where an efficient, easy-to-use library is required for visual outputs.

This document will firstly explore the core functionalities of the `Tiny4kOLED` library, including the key features it offers for controlling `OLED` displays. After that, two practical code examples will be illustrated that showcase how to use the library for different tasks.

Index:

Functionalities

- Initialization

  `oled.begin()`

  Initializes the `OLED` screen.

- Clear the screen

  `oled.clear()`

  Clears the display, removing all content from previous operations.

- Turning display On/Off

  `oled.on()`

  Turns on the `OLED` display.

  `oled.off()`

  Turns off the `OLED` display.

- Display rotation

  `oled.setRotation()`

  Adjusts the orientation of the display. There are two supported orientations, the `oled.begin()` method sets the orientation to 1, it is possible to flip it 180 degrees by switching it to 0.

- Frames

  `oled.switchRenderFrame()`

  Changes the memory area where new data for the screen is being written.

  `oled.switchFrame()`

  Does the same as `oled.switchRenderFrame()` and additionally changes the memory area that is being displayed.

- Font

  `oled.setFont()`

Sets the font that is going to be used to display text. The two supported font sizes are 8X16 and 6X8.

- Displaying text

  `oled.setCursor(x, y)`

  Sets the position where the text is going to be printed. The vertical alignment can only be set to 0, 1, 2 or 3. This is because the screen is separated into four 8px rows.

  X is in pixels and Y is one of the four available rows.

  `oled.print("Text")`

  Prints the text given at the current cursor position.

Millisecond counter

Here is a simple example using the `Tiny4kOLED` library that can be found on the `GitHub` page of the library: GitHub - datacute/Tiny4kOLED: Library for an ATTiny85 to use an SSD1306 powered, double buffered, 128x32 pixel OLED, over I2C

```
#include <Wire.h>
#include <Tiny4kOLED.h>

void setup() {
  oled.begin();

  oled.setFont(FONT8X16);

  oled.clear();

  oled.on();

  oled.switchRenderFrame();
}

void loop() {
  updateDisplay();
  delay(50);
}

void updateDisplay() {
  oled.clear();

  oled.setCursor(0, 1);

  oled.print(F("ms: "));

  oled.print(millis());

  oled.switchFrame();
}
```

This `Arduino` sketch uses the `Tiny4kOLED` library to display the number of milliseconds that have passed since the program started. The value is continuously updated on the screen.

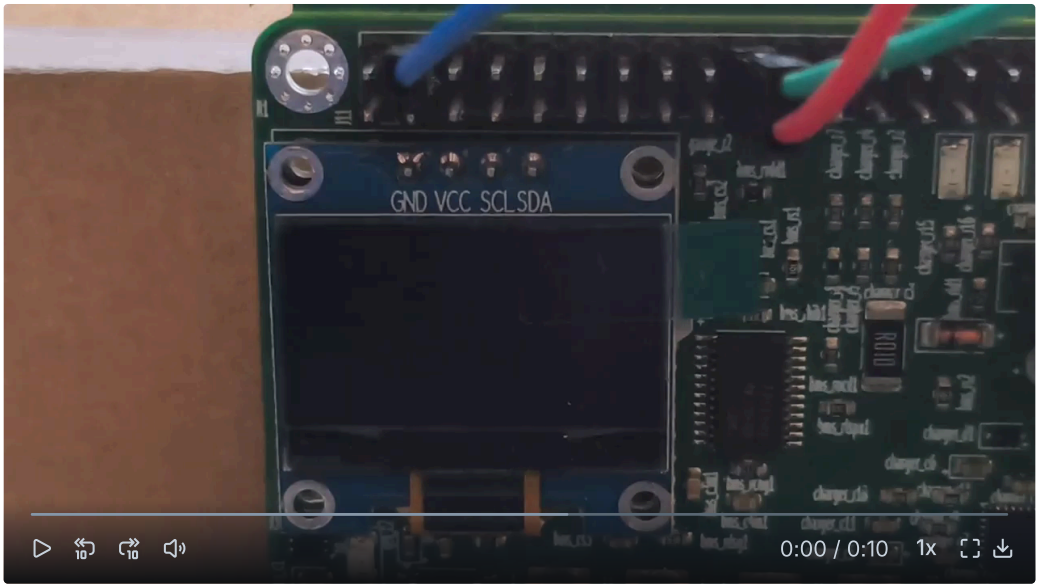
Main Functionality

- **Display Milliseconds**
  The main feature of the code is to display the **elapsed time** in milliseconds on the `OLED` screen, starting from when the program begins. The time updates every 50 milliseconds.

Loop Behaviour

- The `loop()` function repeatedly calls the `updateDisplay()` function, which updates the `OLED` display with the current time in milliseconds.

This is what should be displayed once the code is uploaded:



Accelerated clock with time reminders

Here is another example. In this case, it displays the current date and, at certain hours, a reminder appears on the screen:

```
#include <Wire.h>
#include <Tiny4kOLED.h>

const int screenWidth = 128;
const int screenHeight = 32;
const int margin = 4;
const int charWidth = 6;
const int charHeight = 8;
const int maxCharPerLine = ((screenWidth - (2 * margin)) / charWidth);

unsigned long startMillis;
int hours = 0;
int minutes = 0;

bool justShownReminder = false;
void setup() {
  Wire.begin();
  oled.begin();
  oled.setFont(FONT6X8);
  oled.clear();
  oled.on();
  startMillis = millis(); // store the time when the program starts
  oled.switchRenderFrame();
}

void loop() {
  updateDisplay(); // update the display with the current time and any reminders
```

```
  delay(100);
}

void updateDisplay() {
  unsigned long elapsedTime = millis() - startMillis; // calculate the time elapsed since the program started
 unsigned long fastTime = elapsedTime * 100; // accelerate the time

  updateTime(fastTime);
  oled.clear();
  displayTime();
  oled.switchFrame(); // update the OLED screen with the new display content

  if (!justShownReminder) {
    checkReminder();
  }
}

// function to calculate the accelerated minutes and hours
void updateTime(unsigned long fastTime) {
  minutes = (fastTime / 60000) % 60;
  hours = (fastTime / 3600000) % 24;
}

//function to display the time with the proper formatting in the second row of the screen
void displayTime() {
  oled.setCursor(0, 1);
  oled.print(F("Time: "));
  if (hours < 10) oled.print("0"); // add leading zero for single-digit hours
  oled.print(hours);
  oled.print(F(":"));
  if (minutes < 10) oled.print("0"); // add leading zero for single-digit minutes
  oled.print(minutes);
}

// function to check if it's time to show a reminder
void checkReminder() {
  if (hours == 6 && minutes == 0) {
    showReminder("Time to wake up");
  } else if (hours == 12 && minutes == 0) {
    showReminder("It's lunchtime!");
  } else if (hours == 18 && minutes == 0) {
    showReminder("Time for exercise");
  } else if (hours == 21 && minutes == 0) {
    showReminder("Time for dinner");
  } else if (hours == 23 && minutes == 0) {
    showReminder("Time to sleep");
  } else {
    justShownReminder = false;
  }
}

void showReminder(const char* message) {
  oled.clear();

  // adjust the message to fit within a single line
  char line[maxCharPerLine + 1] = {0}; // create an empty character array for the line
  strncpy(line, message, maxCharPerLine); // copy the message into the line array

```

```
  int lineLen = strlen(line); // get the length of the message after it has been copied

  // calculate the position to centre the text on the screen horizontally
  int x = margin + (screenWidth - 2 * margin - (lineLen * charWidth)) / 2;

  oled.setCursor(x, 1);
  oled.print(line);

  oled.switchFrame(); // update the OLED screen with the reminder message

  justShownReminder = true;
  delay(2000); // show the reminder for 2 seconds before moving on to the next update
}
```

This `Arduino` sketch simulates a real-time clock that shows the current time on a 128x32 `OLED` display and periodically displays reminders at specific times of the day. The time is accelerated by a factor of 100 to simulate faster passage of time.

Main Functionality

- **Real-Time Clock:**

  The time starts at 00:00, and minutes and hours are updated in real-time (though accelerated by a factor of 100). The time is displayed on the `OLED` screen in the format `HH:MM`.

- **Reminder Messages:**

  The sketch checks the current time every loop. When the time matches one of the specified reminder times (6:00, 12:00, 18:00, 21:00, 23:00), a message is displayed, such as "Time to wake up," "It's lunchtime!" etc.

  The message is displayed for 2 seconds, and then the screen refreshes to show the time again.

- **Accelerated Time:**

  The `fastTime` variable speeds up the time by multiplying the elapsed time by 100. This causes the clock to update faster, showing time as if it passes 100 times quicker than real-time.

- **Display:**

  The time is updated and shown on the second row of the `OLED` screen.

  When a reminder is triggered, the message is displayed in the centre of the screen.

This is what should be displayed once the code is uploaded: