Level meter module

REV A

Publication Date: 2013/11/12

XMOS

Table of Contents

1 Overview 3
1.1 Features . . . 3
1.2 Memory requirements . . . 3
1.3 Resource requirements . . . 3

2 Hardware requirements 4

3 API 5

4 API 6

5 Programming guide 7
5.1 Includes and configuration . . . 7
5.2 Programming . . . 7
5.3 Software requirements . . . 7

6 Example applications 8
6.1 app_display_spectrum . . . 8
6.1.1 Getting Started . . . 8
6.2 app_display_spectrum_from_adc . . . 8
6.2.1 Getting Started . . . 8

1 Overview

IN THIS CHAPTER

- Features
- Memory requirements
- Resource requirements

The level meter module is used to create a level meter display of a data array on LCD. The rendered image is stored in SDRAM.

1.1 Features

- Non-blocking SDRAM management.
- Real time rendering.
- Color selection for the display.
- No real time constraints on the application.

1.2 Memory requirements

Resource	Usage
Data	1520 bytes
Program	2160 bytes

1.3 Resource requirements

Resource	Usage
Channels	1
Timers	0
Clocks	0
Cores	1

2 Hardware requirements

This module may be evaluated using the sliceKIT Modular Development Platform, available from digikey. Required board SKUs are:

- XP-SKC-U16 (slicekit U16 Core Board) plus xTAG-2
- XA-SK-SCR480 (which includes a 480x272 color touch screen)
- XA-SK-SDRAM

To build a project including the `module_level_meter` the following modules are required:

- `module_display_controller`
- `module_sdram` in `sc_sdram_burst` which handles the SDRAM
- `module_lcd` in `sc_lcd` which handles the LCD

The section below details the configuration defines and the APIs used in the application.

The color palette to be used for the level meter display can be configured via the header `level_meter_conf.h`. The defines are:

LEVEL_METER_NCOLORS
This defines the number of color used for the level meter display.

LEVEL_METER_COLORS
This gives the colors used. The colors can be picked from the list given in `level_meter.h`.

4 API

The `module_level_meter` functionality is defined in

- `level_meter.xc`
- `level_meter.h`

The level_meter API is:

```
void level_meter(chanend c_dc,
                 unsigned frBufNo,
                 unsigned data[],
                 unsigned N,
                 unsigned maxData)
```

This function renders the level meter display of a given data sequence.

It connects to display controller for storing the rendered image frame.

This function has the following parameters:

`c_dc`	channel connecting display controller.
`frBufNo`	index of frame buffer to be updated.
`data`	array containing magnitude spectrum values.
`N`	number of data values to be displayed as bars.
`maxData`	Maximum possible data value

5 Programming guide

IN THIS CHAPTER

- Includes and configuration
- Programming
- Software requirements

This section provides information on how to create an application using `level_meter` API.

5.1 Includes and configuration

The application needs to include `level_meter.h`. A configuration file `level_meter_conf.h` should be placed in the application directory. The color palette for the level meter display is defined in the configuration file.

5.2 Programming

The level meter module uses the APIs of display controller module. A simple application function that uses `level_meter` API is given below.

```
void app(c_dc, data, N)
{
  unsigned frBuf;

  // Create frame buffer
  frBuf = display_controller_register_image(c_dc, LCD_ROW_WORDS, LCD_HEIGHT
    ↪ );

  // Render level meter display frame and commit
  level_meter(c_dc, frBuf, data, N);
  display_controller_frame_buffer_commit(c_dc, frBuf);
}
```

`c_dc` is the channel connecting display controller. `data` is the array of unsigned data values to be displayed. `N` is the number of data values.

5.3 Software requirements

The module is built on xTIMEcomposer version 13 The module can be used in version 13 or any higher version of xTIMEcomposer.

6 Example applications

IN THIS CHAPTER

- app_display_spectrum
- app_display_spectrum_from_adc

This tutorial describes the demo applications that uses the level meter module. Section §2 describes the required hardware setup to run the demos.

6.1 app_display_spectrum

This application uses display controller and other modules to create a level-meter kind of spectral display on an LCD for a simulated signal. This application demonstrates real-time rendering and display of spectrum by taking short-time fourier transform.

6.1.1 Getting Started

1. Connect XA-SK-SDRAM Slice Card to the XP-SKC-U16 Slicekit Core board using the connector marked with `SQUARE`.
2. Connect XA-SK-SCR480 Slice Card with LCD to the XP-SKC-U16 Slicekit Core board using the connector marked with `DIAMOND`.
3. Select `app_display_spectrum`. Build the project and run.

The spectra of segments of mixed signal of two simulated chirp waveforms are displayed on LCD.

6.2 app_display_spectrum_from_adc

This application uses `module_usb_tile_support` along with display controller and other modules to create a level-meter kind of spectral display on an LCD for an analog audio input. This application showcases the use of multichannel ADC in an xCORE-USB series XMOS device to sample the analog input and real-time rendering and display of spectrum by taking short-time fourier transform.

6.2.1 Getting Started

1. Connect XA-SK-SDRAM Slice Card to the XP-SKC-U16 Slicekit Core board using the connector marked with `SQUARE`.
2. Connect XA-SK-SCR480 Slice Card with LCD to the XP-SKC-U16 Slicekit Core board using the connector marked with `DIAMOND`.

3. Connect XA-SK-MIXED SIGNAL Slice Card to the XP-SKC-U16 Slicekit Core board using the connector marked with A.

4. Give the two channels of audio input from a PC or a mobile to pins 1 and 2 of J2 on the mixed signal slice card using a suitable cable. The ground is connected to pin 4 of J3.

5. Select `app_display_spectrum_from_adc`. Build the project and run.

6. Play an audio in the PC or mobile.

The spectra of segments of mixed signal of two audio channels are displayed on LCD.