IN THIS DOCUMENT

1 sw_display_controller demo : Quick Start Guide

- XP-SKC-L16 sliceKIT
- XA-SK-SCR480 Slice Card,
- XA-SK-SDRAM Slice Card,
- module_sdram,
- module_lcd,
- module_display_controller,
- module_i2c_master,
- module_touch_controller_lib,
- module_slicekit_support,

together to create an interactive display on LCD. This application showcases some of the key software features and serves as an example on how to use an LCD without the real-time constraint of having to update the LCD line buffer and how to use the touch screen for interactive display.

1.1 Hardware Setup

The XP-SKC-L16 sliceKIT Core board has four slots with edge connectors: `SQUARE`, `CIRCLE`, `TRIANGLE` and `STAR`.

To setup up the system:

1. Connect XA-SK-SDRAM Slice Card to the XP-SKC-L16 sliceKIT Core board using the connector marked with the `STAR`.
2. Connect XA-SK-SCR480 Slice Card to the XP-SKC-L16 sliceKIT Core board using the connector marked with the `TRIANGLE`.
3. Connect the xTAG Adapter to sliceKIT Core board, and connect xTAG-2 to the adapter.
4. Connect the xTAG-2 to host PC. Note that the USB cable is not provided with the sliceKIT starter kit.
5. Set the `xCONNECT LINK` to `OFF` on the xTAG Adapter(XA-SK-XTAG2).

Publication Date: 2013/11/12 REV A

6. Ensure the jumper on the XA-SK-SCR480 is bridged if the back light is required.

7. Switch on the power supply to the sliceKIT Core board.

1.2 Import and Build the Application

1. Open xTIMEcomposer and check that it is operating in online mode. Open the edit perspective (Window->Open Perspective->XMOS Edit).

2. Locate the `'Display Controller Demo'` item in the xSOFTip pane on the bottom left of the window and drag it into the Project Explorer window in the xTIMEcomposer. This will also cause the modules on which this application depends to be imported as well.

3. Click on the app_display_controller_demo item in the Explorer pane then click on the build icon (hammer) in xTIMEcomposer. Check the console window to verify that the application has built successfully.

4. There will be quite a number of warnings that `bidirectional buffered port not supported` These can be safely ignored for this component.

For help in using xTIMEcomposer, try the xTIMEcomposer tutorial, which you can find by selecting Help->Tutorials from the xTIMEcomposer menu.

Note that the Developer Column in the xTIMEcomposer on the right hand side of your screen provides information on the xSOFTip components you are using. Select the module_display_controller component in the Project Explorer, and you will see its description together with API documentation. Having done this, click the *back* icon until you return to this quickstart guide within the Developer Column.

1.3 Run the Application

Now that the application has been compiled, the next step is to run it on the sliceKIT Core Board using the tools to load the application over JTAG (via the xTAG-2 and xTAG Adapter card) into the xCORE multicore microcontroller.

1. Select the file `app_display_controller_demo.xc` in the `app_display_controller_demo` project from the Project Explorer.

2. Click on the `Run` icon (the white arrow in the green circle).

3. At the `Select Device` dialog select `XMOS xTAG-2 connect to L1[0..1]` and click `OK`.

4. Wait until the images have loaded over the xTAG connector from the host, this should take approximately 21 seconds.

5. There should be a series of 6 images for transition from one to another.

6. Once the first image is displayed, a message is displayed on the console to prompt the user to touch any of the corners or the center of LCD screen for watching different transition effects.

1.4 Next Steps

1. Try changing the files that are loaded from the host. To do this, generate an image of 480 by 272 pixels, save it in `tga` format uncompressed in "top left" format ("bottom left" will also work but the image will have to be upside-down). Save the file(s) into `the app_display_controller_demo` directory within your workspace. Now, increment the `IMAGE_COUNT` define to 7 and add the name of your new image to the array `images`. Ensure the filename is less than 30 characters long.

2. Each transition has a frame count that configures the speed of the transition, try adjusting them and observe the results. To do this take a look at the API for the display controller. Note how each of the transition effects have a `frame_count` parameter. This parameter specifies how many frames the transition should take.

3. Try writing an exciting transition effect. To do this, begin with the template shown below and refer to the Display Controller API documentation.

```
static void transition_exciting_impl(chanend server, unsigned
  ↪ next_image_fb,
   unsigned image_from, unsigned image_to, unsigned line) {
   //insert code here
}
unsigned transition_exciting(chanend server, unsigned frame_buf[2],
  unsigned from, unsigned to, unsigned frames, unsigned cur_fb_index) {
  unsigned next_fb_index;
  for (unsigned frame = 0; frame < frames; frame++) {
    next_fb_index = (cur_fb_index + 1) & 1;
    for (unsigned line = 0; line < LCD_HEIGHT; line++)
      transition_exciting_impl(server, frame_buf[next_fb_index], from,
        ↪ to, line);
    frame_buffer_commit(server, frame_buf[next_fb_index]);
    cur_fb_index = next_fb_index;
  }
  return cur_fb_index;
}
```