Traveo™ Family Graphics MCUs - How to Set Up Video Output

Author: Waqar Saleem

Associated Part Family: S6J3200

Related Documents: Click Here

This application note describes how to get a video output from Traveo family S6J3200 series graphics MCUs. It describes related subsystems that are involved in setting up the video output. It also provides example code snippets for actually implementing it using the Cypress evaluation board environment.

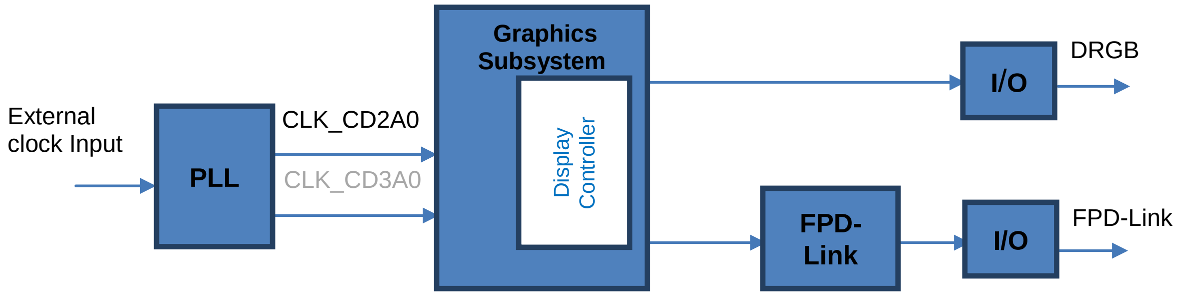
1 Introduction

Cypress offers the Traveo family of ARM® Cortex®-R5 based MCUs for instrument cluster applications, including those that need graphics rendering, from low-end to high-end.

To support the display function in instrument clusters, Traveo MCUs have a graphics subsystem. It consists of a 2.5D engine and an optional 3D engine. The display controller is an interface for connecting the rendering engine with the displays and collects all graphics content to transmit over one of the supported video output interfaces.

One of the very first steps in setting up the graphics subsystem is to set up the video output interface in the display driver. This application note describes the setup for various hardware modules needed for the output and provides code examples. This application note only covers Digital RGB (DRGB; parallel interface) and FPD-Link (sometimes referred to as LVDS, a differential interface with up to four data channels and clock) interfaces.

Figure 1. Clock Input, Graphics Subsystem, and Video Outputs



Note: CLK_CD2A0 is for display reference; CLK_CD3A0 is for graphics engine

2 Setting Up the Video Output

To get the video data generated by the graphics subsystem (output at the MCU pins), there are a few hardware blocks in the graphics driver and non-graphics subsystems that need to be configured. They are described as follows.

2.1 Display Controller Settings

Display controller settings are divided into two categories: those common to both DRGB and FPD-Link interfaces, and those specific to the FPD-Link interface.

2.1.1 Common Settings Between DRGB and FPD-Link Outputs

- **Display Reference Clock:** The clock used as the display reference clock in the graphics subsystem is CLK_CD2A0 (assigned name). PLL1 is the default source to generate this clock. The maximum frequency for this clock is 400 MHz.

- **Clock Divisor:** The pixel or dot-clock for the display is generated from CLK_CD2A0 by a divider in the graphics subsystem.

  The graphics driver is usually used to configure and use the graphics subsystem. API calls of the driver are used to specify the display reference clock (mmlGdcSysInitializeDriver) and dot-clock values (mmlGdcDispOpenDisplay). The division ratio to convert CLK_CD2A0 to the pixel clock is calculated and set automatically by this API call.

- **Display Timing Parameters:** To generate the video output with the proper timing, various parameters of horizontal and vertical display times must be set up. These parameters, for example the number of vertical and horizontal pixels, must match the display requirements. These parameters are set up using the graphics driver.

- **MapBit Setting:** DRGB video signals generated by the graphics subsystem can be mapped to MCU pins. This is achieved by configuring MapBit registers to set up a signal-to-pin relationship.

  For the DRGB output, only color signals can be configured for remapping. For the FPD-Link output, sync signals can also be remapped. Sync signals are transmitted as data on the FPD-Link data output lanes.

2.1.2 Settings for FPD-Link Output

- **TCON Setting:** TCON (Timing Controller) is used when the FPD-Link output is required. TCON enables generating video output signals in a flexible way. Some additional parameters must be set up so that TCON can generate the required SYNC and control signals.

The example in Section **Error! Reference source not found.** shows a sample TCON configuration.

2.2 Additional Setup

In addition to the graphics subsystem setup, two other modules must be configured.

2.2.1 I/O Port Setting for DRGB Output Case: I/O Port Setup

There are parameters that must be set up for each port or pin of the MCU such as I/O direction, drive strength, pull resistor, and port function. This is done by setting up registers in the I/O Port module depending on the software environment as follows:

1. Direct register access
2. Software development based on a software library
3. MCAL configuration (in the case of AUTOSAR)

The approach used depends on your preference. The first two options may be suitable for an evaluation. Cypress provides a Peripheral Driver Library (PDL) for testing and evaluation. For the actual project development by Tier 1 companies, the MCAL configuration with AUTOSAR is almost always used. Cypress develops and provides MCAL as a product for use along with its Traveo family MCUs.

For the DRGB Output, up to 28 ports can be used per display; parameters must be set up for each port.

2.2.2 FPD-Link: Initialization Sequence

The FPD-Link module converts the video output from the graphics subsystem to an FPD-Link-compatible data stream. It consists of up to four differential data signals and one differential clock.

Once the graphics subsystem is configured, the FPD-Link module must be activated and deactivated in a specific sequence as listed below.

Activation sequence

1. Power ON
2. Input to TXCLK, TXIN (internal connection between the graphics subsystem and FPD-Link)
3. Set CH_SEL [1], CTRL1 [1], TXn_CONF [1]
4. PWD33 release
5. Set disp_ClockEnable [1] = H ; After more than 20 cycles of TXCLK, set PWD33=L
6. More than 20 ns later, PWD12 [1] = L
7. More than 20 ns later, ENABLE [1] = H
8. After ENABLE = H, set PLLRST [1] = L
9. After PWD12 [1] = L, more than 20ns + 10 ms later, RST [1] = L

Deactivation sequence

1. RST [1] = H
2. More than 100 ns later, video stream from graphics subsystem turned OFF
3. ENABLE [1] = L, PWD12 = H [1], PLLRST [1] = H
4. More than 20 ns later, PWD33 [1] = H
5. Display termination process
6. Power OFF

3 Software Example

The following example code is from a software project that was developed and tested on the Cypress S6T3J200281281A208A2 evaluation board. The code shows setting up the video output using the FPD-Link interface. For DRGB output only, FPD-Link module initialization will not be necessary and additional I/O port settings will be needed.

Key components of the test environment are mentioned below.

- **Hardware Setup:** S6J328CK MCU; Sharp DRGB WVGA Display LQ070Y3DG05; TI DS90CF384 FPD-Link Receiver
- **Software Setup:** S6J3200 2D Graphics Driver, Green Hills Multi

See the S6J3200 2D Graphics Driver User Guide for more information on the API used.

1. Display Reference Clock Setting using the Driver API

```
MML_GDC_SYSINIT_INFO pDriverInitInfo = {0, 400000000};
```

2. TCON Parameters, MapBit, and Display Timing Settings

```
/* TCON parameters for FPD-Link; needed only for FPD-Link */
static MML_GDC_DISP_TCON_PROPERTIES    tcon_values_0[37] =
{
    /* Offset Address,  Value */
    { 0x0410, 0x00000100}, // TCON_CTRL

    // define HSYNC: 0x338...0x380 - 824 to 896
    { 0x0450, 0x03380000}, // SPG0PosOn
    { 0x0454, 0x0000FFFF}, // SPG0MaskOn
    { 0x0458, 0x03800000}, // SPG0PosOff
    { 0x045C, 0x0000FFFF}, // SPG0MaskOff

    // define VSYNC: 0x1E3...0x1EA - 483 to 490
    { 0x0460, 0x000001e3}, // SPG1PosOn
    { 0x0464, 0x7FFF0000}, // SPG1MaskOn
    { 0x0468, 0x000001ea}, // SPG1PosOff
    { 0x046C, 0x7FFF0000}, // SPG1MaskOff

    // define DataEnable:  0x000...0x320 & 0x000...0x1E0 - 0...800 & 0...480
    { 0x0470, 0x00000000}, // SPG2PosOn
    { 0x0474, 0x00007FFF}, // SPG2MaskOn
    { 0x0478, 0x03200000}, // SPG2PosOff
    { 0x047C, 0x00007FFF}, // SPG2MaskOff
    { 0x0480, 0x00000000}, // SPG3PosOn
    { 0x0484, 0x7FFF0000}, // SPG3MaskOn
    { 0x0488, 0x000001E0}, // SPG3PosOff
    { 0x048C, 0x7FFF0000}, // SPG3MaskOff

    { 0x0510, 0x00000002}, // SMx0Sigs
    { 0x0514, 0xFFFFFFFD}, // SMx0FctTable
    { 0x0518, 0x00000018}, // SMx1Sigs
    { 0x051C, 0xFFFFFFFB}, // SMx1FctTable
    { 0x0520, 0x0000002C}, // SMx2Sigs
    { 0x0524, 0x00000008}, // SMx2FctTable

    /* define MapBit settings according to DRGB or LVDS stream definition */
    { 0x0418, 0x03040508}, // MapBit3_0   ( R3, R4, R5, G0)
    { 0x041C, 0x11000102}, // MapBit7_4   ( B1, R0, R1, R2)
    { 0x0420, 0x0B0C0D10}, // MapBit11_8  ( G3, G4, G5, B0)
    { 0x0424, 0x191A090A}, // MapBit15_12 ( VSYNC, EN, G1, G2)
    { 0x0428, 0x13141518}, // MapBit19_16 ( B3, B4, B5, HSYNC)
    { 0x042C, 0x16171F12}, // MapBit23_20 ( B6, B7, RES, B2)
    { 0x0430, 0x06070E0F}, // MapBit27_24 ( R6, R7, G6, G7)

    { 0x0434, 0x03020100}, // MapBit3_0_Dual
    { 0x0438, 0x07060504}, // MapBit7_4_Dual
    { 0x043C, 0x0B0A0908}, // MapBit11_8_Dual
    { 0x0440, 0x0F0E0D0C}, // MapBit15_12_Dual
    { 0x0444, 0x13121110}, // MapBit19_16_Dual
    { 0x0448, 0x17161514}, // MapBit23_20_Dual
    { 0x044C, 0x1B1A1918}, // MapBit27_24_Dual

};

#define NUM_OF_TCON_SETTING_0   sizeof(tcon_values_0)/sizeof(tcon_values_0[0])

/* set Display Timing Parameters (for WVGA) */
/* Derived from display specification*/

static MML_GDC_DISPLAY        display;

mode_line.pixelClock                = 30.0f;/* MHz */

mode_line.horDisplayPeriod          = 800;  /* pixel */
mode_line.horPulseStart             = 824;
mode_line.horPulseEnd                  = 896;
mode_line.horTotal                     = 927;


mode_line.vertDisplayPeriod            = 480;  /* lines */
mode_line.vertPulseStart               = 483;
mode_line.vertPulseEnd                 = 490;
mode_line.vertTotal                    = 524;

mode_line.DCKDelay                     = 0;
mode_line.DCKInvertEnable              = MML_GDC_DISP_DCK_INVERT_OFF;
mode_line.syncPolarity                 = MML_GDC_DISP_DE_HIGH;

mode.outputController           = MML_GDC_DISP_CONTROLLER_0;
mode.fcvm                       = 1;
mode.pDISP_TCON_PROPS           = tcon_values_0;
mode.modeLine                   = &mode_line;
mode.xResolution                = 800;
mode.yResolution                = 480;
mode.countTconProps             = NUM_OF_TCON_SETTING_0;

mmlGdcDispOpenDisplay(&mode, &display);
mmlGdcDispSetAttribute(display, MML_GDC_DISP_ATTR_BACKGROUND_COLOR, 0x00ff0000);
mmlGdcDispCommit(display);
```

3. FPD-Link Module Initialization

```
//-------
// Channel Configuration - for FPD-Link
//-------

FPDLCNV_UNLOCK           = 0xB3B0BcB4; // unlock

FPDLCNV_TX0_CONF_RX_NUM = 0x1;         // TxDOUT0
FPDLCNV_TX1_CONF_RX_NUM = 0x2;          // TxDOUT1
FPDLCNV_TX2_CONF_RX_NUM = 0x3;          // TxDOUT2
FPDLCNV_TX3_CONF_RX_NUM = 0x4;          // TxDOUT3
FPDLCNV_TX4_CONF_RX_NUM = 0x0;          // TxCLK

FPDLCNV_CH_SEL_CH       = 0x0;         //disp0

//-------
// Initialization of FPD-Link module
//-------
Disp_ClockEnable = 1;                         //Pseudo code - Director write
                                              //to Graphics Engine register
Read_Disp_ClockEnable = Disp_ClockEnable;   // Pseudo code - Dummy Read
FPDLCNV_CTRL0_PWD33     = 0x0;
wait (T);                                    // T = at least 20 nsec
FPDLCNV_CTRL0_PWD12     = 0x0;
FPDLCNV_CTRL0_ENABLE    = 0x1;
FPDLCNV_CTRL0_PLLRST    = 0x0;
FPDLCNV_CTRL1_LFCTRL    = 0x0;               // Example value only
FPDLCNV_CTRL1_FRANGE    = 0x0;               // Example value only
wait(T);                                     //T = at least 20ms + 10nsec
FPDLCNV_CTRL0_RST       = 0x0;
```

4 References

- S6J3200 Series Datasheet
- S6J3200 Series Hardware Manual
- S6J3200 2D Graphics Driver User Guide (Provided under NDA only)

Document History

Document Title: AN214034 – Traveo™ Family Graphics MCUs - How to Set Up Video Output

Document Number: 002-14034

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5531805	MUSA	11/24/2016	Initial release

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Lighting & Power Control	cypress.com/powerpsoc
Memory	cypress.com/memory
PSoC	cypress.com/psoc
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless/RF	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corporation. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com