Application Note

For

Glamo 3365 Series

The Multimedia co-Processor for Mobile

Preliminary

V.0.92

March 20, 2007

This specification is subject to change without notice. SMedia Technology Corporation assumes no responsibility for any errors contained herein.

Copyright by SMedia Technology Corporation, all rights reserved.
### History

<table>
<thead>
<tr>
<th>VERSION</th>
<th>DATE</th>
<th>AUTHOR</th>
<th>APPROVAL MANAGER</th>
<th>APPROVAL DATE</th>
<th>NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.85</td>
<td>Apr. 1, 2006</td>
<td>Luke Lin</td>
<td>Chin-Tu Wu</td>
<td>Apr. 1, 2006</td>
<td>1</td>
</tr>
<tr>
<td>0.86</td>
<td>Jul. 13, 2006</td>
<td>Gavin Yen</td>
<td>Chin-Tu Wu</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>0.87</td>
<td>Jul. 13, 2006</td>
<td>Luke Lin</td>
<td>Chin-Tu Wu</td>
<td></td>
<td>3, 4</td>
</tr>
<tr>
<td>0.88</td>
<td>Aug. 23, 2006</td>
<td>Gavin Yen</td>
<td>Chin-Tu Wu</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>0.89</td>
<td>Sep. 13, 2006</td>
<td>Gavin Yen</td>
<td>Chin-Tu Wu</td>
<td></td>
<td>6</td>
</tr>
<tr>
<td>0.90</td>
<td>Sep. 19, 2006</td>
<td>Gavin Yen</td>
<td>Chin-Tu Wu</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>0.91</td>
<td>Jan. 31, 2007</td>
<td>Gavin Yen</td>
<td>Chin-Tu Wu</td>
<td></td>
<td>8</td>
</tr>
<tr>
<td>0.92</td>
<td>Mar. 20, 2007</td>
<td>Gavin Yen</td>
<td>Chin-Tu Wu</td>
<td>Mar. 16, 2007</td>
<td>9</td>
</tr>
</tbody>
</table>
# Notes

<table>
<thead>
<tr>
<th>NUMBER</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>
| 2      | Chapter 2: Reference Circuit  
          Update reference circuit to have more clear graph and fonts  
          Add Notice: Notice about non-used pins of 3365 if only using some functions of 3365  
          Chapter 7: Modify the Layout Rule of 3365  
          Appendix.: Add TV Encoder Reference Schematic (Reference Only)  
          Add Notice: Notice about the LCD & TV Dual Display of 3365 |
| 3      | Modify Sensor reset and power down pin assignment in Page 13. |
| 4      | Add chapter 11 Quick Boot Concept. |
| 5      | Update Power Consumption of Glamo 3365 Table |
| 6      | Add 3365 Lead-free reflow profile suggestion for SMT |
| 7      | Add Notice about Indirect Addressing 8-bit type of Host Bus  
          Update Initial Glamo3365 script (page 50) |
| 8      | Modify description about Terminate non-used Function (page 13)  
          Update Initial Glamo3365 script (page 49) |
| 9      | Add a Troubleshooting Page  
          1.About Samsung 2443 Host Bus Connection (Page 60) |
8. POWER DELIVERY

8.1. POWER CONSUMPTION ................................. 45
8.2. POWER ON/OFF PROCEDURE ......................... 47
8.3. POWER SAVING ........................................... 48
8.4. CLOCK TREE ............................................. 48

9. HOW TO INITIATE ........................................... 50

9.1. INITIATE GLAMO 3365 AND HOST BUS .......... 50
9.2. INITIATE LCD ............................................. 52
  9.2.1. Fill Data in Glamo 3365 Registers .................. 52
  9.2.2. Command Data for Initial Panel ................. 52
9.3. INITIATE CCD/CMOS SENSOR ....................... 55
9.4. INITIATE AUDIO CODEC .............................. 55

10. NO-WAIT MEMORY BUS READ/WRITE TIMING FOR GLAMO 3365 ....... 58

11. QUICK BOOT CONCEPT .................................. 59

12. TROUBLESHOOTING ..................................... 60

APPENDIX: ....................................................... 61

PART A: LCD MODULE REFERENCE SCHEMATICS .......... 61
  LCM: SONY / ACX506AKU .................................. 61
  LCM: SONY / ACX515AKM-7 ............................... 62
  LCM: CASIO / COM18T1170 ............................... 63
  LCM: CASIO / CMG12MS8 .................................. 64
  LCM: EPSON / L2F50032 .................................... 65

PART B: IMAGE SENSOR REFERENCE SCHEMATICS .......... 66
  TOSHIBA TCM8210MD ....................................... 66
  SANYO IGT99268F .......................................... 67
  SAMSUNG SSX433CA ........................................ 68
  PixArt PAS302 ................................................ 69
  OMNIVISION OV9640 ....................................... 70
  OMNIVISION OV7648 ....................................... 71
  BIOMORPHIC Bi8831A ....................................... 72
  AGILENT ADCM-2700-0000 ............................... 73

PART C: TV ENCODER (TV OUT) INTERFACE REFERENCE SCHEMATICS .......... 74
  CH7023/CH7024 .............................................. 74
  IMAGIS ISE2110B ............................................ 75
Figures

FIGURE 1.1-1 MOBILE PHONE SYSTEM ARCHITECTURE ...........................................................................................................9
FIGURE 1.2-1 GLAMO 3365 MULTIMEDIA CO-PROCESSOR BLOCK DIAGRAM ..............................................................................10
FIGURE 1.3-1 GLAMO 3365 MULTIMEDIA CO-PROCESSOR MEMORY MAPPED I/O ADDRESS ....................................................11
FIGURE 1.4-1 SAMSUNG 2410 MEMORY MAP AND GLAMO 3365 BASE ADDRESS SETTING ON EVB .............................................12
FIGURE 3.-1 GLAMO 3365 HOST BUS ....................................................................................................................................17
FIGURE 3.1.-1 TYPE 1 CONNECTION BETWEEN GLAMO 3365 HOST BUS AND BASEBAND CPU BUS ..........................................19
FIGURE 4.1-1 CONNECTION OF GLAMO 3365 TO 80-TYPE CPU INTERFACE LCM .................................................................26
FIGURE 4.1-2 CONNECTION OF GLAMO 3365 TO 68-TYPE CPU INTERFACE LCM ..................................................................27
FIGURE 4.1-3 CONNECTION OF GLAMO 3365 TO RGB WITH SPI INTERFACE LCM .................................................................28
FIGURE 4.1-4 CONNECTION OF GLAMO 3365 TO RGB WITHOUT SERIAL INTERFACE LCM ..........................................................31
FIGURE 4.2-1 CONNECTION OF GLAMO 3365 TO 80-TYPE CPU INTERFACE LCM IN BYPASS MODE ..........................................32
FIGURE 4.2-2 CONNECTION OF GLAMO 3365 TO 68-TYPE CPU INTERFACE LCM IN BYPASS MODE ..........................................33
FIGURE 5.3-1 CONNECTION OF GLAMO 3365 TO SENSOR (MODE 1) .......................................................................................35
FIGURE 5.3-2 CONNECTION OF GLAMO 3365 TO SENSOR (MODE 2) .......................................................................................35
FIGURE 6-1 I2S AUDIO INTERFACE ........................................................................................................................................38
FIGURE 6-2 CONNECTION OF GLAMO 3365 TO AUDIO CODEC .................................................................................................38
FIGURE 7.-1 GLAMO 3365 LAYOUT RULE ...................................................................................................................................44
FIGURE 8.2-1 GLAMO 3365 POWER ON/OFF PROCEDURE ......................................................................................................47
FIGURE 8.4-1 GLAMO 3365 CLOCK TREE ....................................................................................................................................49
FIGURE 12-1 MEMORY INTERFACE DIFFERENCE FROM SAMSUNG 2440 ...................................................................................60
FIGURE :SURFACE MOUNT TECHNOLOGY (SMT) REFLOW PROFILES ......................................................................................79
## Tables

<table>
<thead>
<tr>
<th>Table Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLE 3. -1 GLAMO 3365 HOST BUS INTERFACE TO SAMSUNG 2410 PIN DESCRIPTIONS</td>
<td>18</td>
</tr>
<tr>
<td>TABLE 4-1 LCD INTERFACE PIN DESCRIPTIONS</td>
<td>23</td>
</tr>
<tr>
<td>TABLE 5-1 CAMERA INTERFACE PIN DESCRIPTIONS</td>
<td>34</td>
</tr>
<tr>
<td>TABLE 6-1 AUDIO INTERFACE PIN DESCRIPTIONS</td>
<td>38</td>
</tr>
</tbody>
</table>
1. Glamo 3365 System Overview

This chapter describes how to build up Glamo 3365 in your mobile phone hardware system and connect LCD/CMOS module to Glamo 3365.

1.1. Mobile Phone System Architecture

This figure shows mostly mobile (feature/smart) phone architecture, some DBB supports multimedia function and LCD/CCD interface, but without 2D/3D and MPEG engine now. Some hi-end BB supports game/video function by CPU power, but it's not good as hardware implement. Now, Glamo 3365 provides total hardware solution for multimedia application and it is easy to connect to mostly baseband by SRAM-like bus.
1.2. **Glamo 3365 Block Diagram**

Figure 1.2.-1 Glamo 3365 Multimedia co-processor Block Diagram

Mobile phone baseband IC use asynchronous SRAM-like interface to communicate outside device. The outside devices include SRAM, FLASH and the Multimedia chip. Most of the interface only support slave mode, that is, CPU driven mode. And some of the interface may support wait state insertion.

Glamo 3365 host bus supports 16 bits SRAM-like interface and it supports Intel 80 type direct addressing mode and others direct/indirect addressing mode.

Glamo 3365 supports 8/9/16/18 bits CPU/RGB interface and by-pass mode for LCD module. Glamo 3365 supports both that with/without memory LCD modules (LCM). The programmable interface timing is designed to fit most LCMs. There are 4 types as below.

- Type 1 80-Type CPU Interface
- Type 2 68-Type CPU Interface
- Type 3: RGB with Serial Interface
- Type 4: RGB with Direct Control Interface

For LCD signal by-pass mode, Glamo 3365 by-pass Baseband LCD data to LCM directly, and it supports 80 and 68 Type interface only.

Video capture (CCD/CMOS) interface supports most sensor module and it supports 2 modes now, one is sensor provides a clock for pixel clock, the other is pixel clock come from Glamo 3365 PLL.

1.3. **Glamo 3365 Memory Mapped I/O Address**

![Glamo 3365 Multimedia co-processor Memory Mapped I/O Address](image)

- Video Playback
- MPEG Engine
- JPEG Engine
- Video Capture/Image Processor
- SRAM (Memory Controller)
- Host Bus
- General
- Not Used
- Staked RAM
- Not Used
- 3D Engine
- 2D Engine
- Audio Engine (RISC)
- Command Queue
- Micro Processor
- LCD Controller
Glamo 3365 stacked 2M/4M bytes on-chip frame buffer memory, and Glamo 3365 on-chip memory is allocated on 0x80_0000 to 0xC0_FFFF. Glamo 3365 addressing memory space up to 16MB, first 8MB space (0x00_0000 to 0x7F_FFFF) is for register that is including general setting, host bus, SRAM (memory), capture, JPEG engine, MPEG engine, playback engine, LCD controller, uP, Command Queue, 2D engine and 3D engine. And second 8MB space (0x80_0000 to 0xFF_FFFF) for stacked memory. Special highlight that you must set Glamo 3365 MMIO in non-cacheable and non-buffered memory space. And please do not connect Glamo 3365 and others device in the same bank (use same chip select pin).

1.4. System Memory Map based on Glamo 3365 EVB

Figure 1.4.-1 Samsung 2410 Memory Map and Glamo 3365 base address setting on EVB

Glamo MMP base address is 0x0800_0000 based on EVB. Base address depend on hardware circuit defined. 0x0800_0000 to 0x08FF_FFFF is used by Glamo MMP.
Since host CPU is Samsung 2410 on Glamo 3365 EVB, so the base address is addressing to 0x0800_0000. After determine baseband and then find out a free memory space for Glamo 3365 in your mobile phone system, Glamo 3365 needs a total 16M space for operating. Once your hardware memory map is not same as Glamo 3365 EVB, please remember change Glamo 3365 base address in boot up firmware (Initial Code), detail see chapter How to initial for host bus initiate setting.

Memory-mapped I/O subsystems and I/O-mapped subsystems both require the CPU to move data between the peripheral device (Glamo 3365, audio chip…) and main memory (SDRAM, Flash...). For example, to input one byte from Glamo 3365 and store the byte into SDRAM, the CPU must read the value and store it into memory.

Following chapters will show the way for connect each bus. When everything settles down, you can start to initiate and test it by firmware (initial procedure), about the procedure please see the chapter How to Initial.

### 1.5. Hardware Circuit AP Note

#### 1.5.1. Modify circuit in HWR# and HRD#

Please put 20pF capacitor in HWR# and HRD#.

![Circuit Diagram](image)

Put near on Glamo 3365 chip.

#### 1.5.2. CFG [2:3] Pin Function Description

<table>
<thead>
<tr>
<th>Pin Number</th>
<th>Pin Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>CFG2</td>
<td>I</td>
<td>Clock source 0: Use internal PLL 1: Use Oscillator</td>
</tr>
<tr>
<td>N2</td>
<td>CFG3</td>
<td>I</td>
<td>Clock frequency 0: 32.768KHz 1: 13MHz</td>
</tr>
</tbody>
</table>
1.5.3. Terminate non-used Function

Notice about non-used pins of 3365 if only using some functions of 3365:

The pins of camera interface: please pull low 100K ohm resistor to Ground (GND) if the default value of the pin is input or hi impedance (Hi-Z) after reset. (Refer to datasheet Chapter 2) · Output pins could be floating.

Other pins: please directly pull low to Ground (GND) if the default value of the pin is input or hi impedance (Hi-Z) after reset. (Refer to datasheet Chapter 2) · Output pins could be floating.

2. Reference Circuit

please assign DGPIO14 & DGPIO15 as sensor's reset & standby pins respectively. GPIO13 & GPIO14 should be reserved for sensor data bus (data0 & data1 for 10 bits data bus).
Note: Please reserve PAD on all of float pins for debug issue.

- MOVDD 1.8V -5%/+5%
- Note: Please reserve PAD on all of float pins for debug issue.
- HOVDD 1.8 to 3.3V depend on BB(CPU) I/O voltage
- CSGPO1 for Sensor D1, CD0 for Sensor D2, CD1 for Sensor D3 addressing (MTK 621X, TI OMAP, Intel 270, Samsung 24xx, Philips Dragonfly)
- COVDD 1.8 to 3.3V depend on Camera I/O voltage
- CD7 for Sensor D9
- TOVDD 1.8 to 3.3V depend on MMC I/O voltage
- UOVDD 1.8 to 3.3V depend on Audio codec I/O voltage

- CFG2=0, Internal PLL(Default). CFG2=1, Oscillator
- CFG3=0, OSCI input 32.768KHz. CFG3=1, OSCI input 13MHz
2.1. Notice about Indirect Addressing 8-bit type of Host Bus

When using indirect addressing 8-bit type of host bus, please add the referred circuit to your schematic.

P.S NC7SZ125 also can be replaced by OR cell and no need pull high resistor
3. Host Bus Interface

Glamo 3365 connect to baseband by SRAM-like bus, but it does not support page mode and burst mode that support by some type SRAM. It supports 16 bit direct addressing and 8/16 bit indirect addressing mode. The address bus is 24 bits width, so it addressing up to 16MB. Lower 8MB space for controller registers setting, upper 8MB for stacked memory (Frame Buffer). Glamo 3365 provides 2MB, 4MB and 8MB stacked memory for customer needs, and it is stacked on die, without any H/W board level effort.

Host bus design procedure:

- Check Baseband’s (AP-processor) SRAM/Memory Bus
- Make sure what type used (Recommend Intel 80 type)
- Connect Data and Address bus
- Connect control signal
- Set CFG [1:0]
- Set PLL Clock
- Initial and test

About detail read and write timing diagram, please see datasheet. And host bus contains total 46 pins that included 23 for addressing, 16 bit for data and 7 pins for bus controller.

Figure 3.1 Glamo 3365 Host Bus

There are 8 types of bus protocol supported by Glamo 3365 host interface. Hardware trapping CFG [1:0], HCLK, HADVn and HA10 to decides the bus protocol of the host interface. **It’s recommend use Intel 80 type 1 direct addressing mode for host (BB, CPU) access.**
Since Glamo 3365 host bus is SRAM-Like bus, please check baseband’s datasheet for SRAM connection method before connect Glamo 3365 host bus to system baseband (co-processor) SRAM bus. The only different is RESET and WAIT signal pins; please connect it to baseband CPU memory bus controller. And set the CFG[1:0] in right setting. Special note that you must turn on WAIT function in host CPU BUS WIDTH & WAIT CONTROL REGISTER (BWSCON) setting, if CPU with wait (Ready) pin.

Table 3. -1 Glamo 3365 Host Bus Interface to Samsung 2410 Pin Descriptions

<table>
<thead>
<tr>
<th>Pin Name</th>
<th>Description</th>
<th>Pin Number</th>
<th>Pin Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSTO</td>
<td>For external device reset control</td>
<td>D6</td>
<td>RST#</td>
<td>I</td>
<td>Chip asynchronous reset signal</td>
</tr>
<tr>
<td>CSn</td>
<td>General Chip Select</td>
<td>B5</td>
<td>HCS#</td>
<td>I</td>
<td>Host interface chip select signal</td>
</tr>
<tr>
<td>WRn</td>
<td>nWE (Write Enable)</td>
<td>D5</td>
<td>HWR#</td>
<td>I</td>
<td>Host interface write command signal</td>
</tr>
<tr>
<td>RDn (nOE)</td>
<td>nOE (Output Enable)</td>
<td>C5</td>
<td>HRD#</td>
<td>I</td>
<td>Host interface read command signal</td>
</tr>
<tr>
<td>UBE0 (DQM1)</td>
<td>nBE0:nWBE0:DQM1</td>
<td>A3</td>
<td>HUB#</td>
<td>I</td>
<td>Host interface upper byte enable</td>
</tr>
<tr>
<td>LBE0 (DQM0)</td>
<td>nBE1:nWBE1:DQM1</td>
<td>C4</td>
<td>HLB#</td>
<td>I</td>
<td>Host interface lower byte enable</td>
</tr>
<tr>
<td>A[23:1]</td>
<td>Address Bus</td>
<td>HA[23:1]</td>
<td>I</td>
<td></td>
<td>Host interface address bus and allow up to 16 MB addressable space</td>
</tr>
<tr>
<td>D[15:0]</td>
<td>Data Bus</td>
<td>HD[15:0]</td>
<td>I/O</td>
<td></td>
<td>Host interface bi-directional data bus</td>
</tr>
<tr>
<td>WAITn</td>
<td>nWAIT requests to prolong a current bus cycle. As long as nWAIT is L, the current bus cycle cannot be completed.</td>
<td>A4</td>
<td>HWAIT#</td>
<td>O</td>
<td>Host interface wait cycle output</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A6</td>
<td>OSCI</td>
<td>I</td>
<td>32 K/13MHz crystal input or 32K/13MHz system clock input</td>
</tr>
</tbody>
</table>
3.1. **80 Type 1 Direct Addressing Mode Implement**

Direct addressing mode has separate bus for address and data, address bus is up to A23 for 16MB memory space, and data bus is 16 bit in/out in one read/write cycle.

80 Type 1 host interface has separate signals for write and read cycle control (WR and RD). In this mode, please set CFG0 = 0, CFG1 = 0, HADVN = 0 and HCLK = 0.

---

**Figure 3.1.-1 Type 1 Connection between Glamo 3365 Host Bus and Baseband CPU Bus**

![Diagram of Type 1 Connection Between Glamo 3365 Host Bus and Baseband CPU Bus](image-url)
3.2. Host Bus PLL Setting

Due to baseband CPU clock is adjustable cause memory bus timing may not match with Glamo 3365 host bus timing, so need to adjust Glamo 3365 PLL and HWAIT# stats to match baseband bus timing. If Glamo 3365 read/write is not stable, please check PLL setting and HWAIT# timing first. You can set wait state timing by baseband side also, and you can get the matched timing in the memory bus. For system performance issue, please set the memory bus as fast as possible.

For example: Baseband CPU Samsung 2410 on Glamo 3365 EVB:

Clock signals
FCLK – for CPU
HCLK – for host bus (AHB) devices, e.g. Memory cont., interrupt, cont.
PCLK – for peripherals bus (APB) devices, e.g. I2C, I2S, UART...

If Samsung 2410 PLL set at 100MHz, bus ratio = 1:1:2, then SDRAM and Glamo 3365 bus is also 100MHz.
See below for ratio setting.

<table>
<thead>
<tr>
<th>HDIVN</th>
<th>PDIVN</th>
<th>FCLK</th>
<th>HCLK</th>
<th>PCLK</th>
<th>Divide Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>FCLK</td>
<td>FCLK</td>
<td>FCLK</td>
<td>1 : 1 : 1 (Default)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>FCLK</td>
<td>FCLK</td>
<td>FCLK/2</td>
<td>1 : 1 : 2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>FCLK</td>
<td>FCLK/2</td>
<td>FCLK/2</td>
<td>1 : 2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>FCLK</td>
<td>FCLK/2</td>
<td>FCLK/4</td>
<td>1 : 2 : 4 (recommended)</td>
</tr>
</tbody>
</table>
### 3.3. 80 Type 1 Default Read/Write Timing Setting

The following timing chart is Intel 80 type write timing, the timing setting is for reference only, and the satiable timing depends on board level design. If your CPU support wait pin then you can skip write active time setting. Don't care HLB# and HUB# pin.

Write Glamo 3365 Step:

1. Address bus ready and CS pin falling.
2. Setup time 30nS.
3. Write pin falling and wait pin output falling.
4. Write active time = 150nS. (If CPU supports wait function, you can skip this setting and CPU will take it automatic.)
5. Hold time 30nS.
6. Turnaround 30nS for next read/write.

---

**Diagram:**

- **HCS#**
- **HA[23:1]**
- **HRD#**
- **HWR#**
- **HWAIT#**
- **HD[15:0]**

<table>
<thead>
<tr>
<th></th>
<th>Setup 30nS</th>
<th>Hold 30nS</th>
<th>Setup 30nS</th>
<th>Turnaround 30nS</th>
</tr>
</thead>
<tbody>
<tr>
<td>HCS#</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HA[23:1]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HRD#</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HWR#</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HWAIT#</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HD[15:0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The following timing chart is Intel 80 type read timing, the timing setting is for reference only, and the satiable timing depends on board level design. If your CPU support wait pin then you can skip read active time setting. Don’t care HLB# and HUB# pin.

Read Glamo 3365 Step:

7. Address bus ready and CS pin falling.
8. Setup time 30nS.
9. Read pin falling and wait pin output falling.
10. Write active time = 280nS. (If CPU supports wait function, you can skip this setting and CPU will take it automatic.)
11. Hold time 30nS.
12. Turnaround 30nS for next read/write.
4. LCD Interface

Support maximum two panels in Glamo 3365 LCD interface (LCD1/Video-encoder (Main) and LCD2 (Sub) cannot refreshed simultaneously). It’s recommend use RGB + SPI interface 16/18 bit W240 x H320 LCD panel for your cell phone, which RGB is for display content(data) and SPI for LCD panel control command. There are some video encoder supports RGB/CPU interface, so Glamo 3365 support TV out also. If you need TV out and LCD out simultaneously, LCD module timing and resolution should same as video encoder’s.

Case 1: There is only one LCD on board:

There are 3 types connection method as below.

- RGB interface 6, 9, 16, 18 bits output panel (with/without RAM integrated)
- CPU interface 8, 9,16, 18 bits output panel (RAM integrated)

Case 2: There are 2 LCD on board:

There are 2 types connection method as blow.

LCD1: CPU interface 8, 9, 16, 18 bits output panel (RAM integrated)
LCD2: CPU interface 8, 9, 16, 18 bits output panel (RAM integrated)

or

LCD1: RGB interface 6, 9, 16, 18 bits output panel (RAM integrated)
LCD2: CPU interface 8, 9, 16, 18 bits output panel (RAM integrated)

Glamo 3365 supported 2 modes interface for LCD module:

- Mode 0 : CPU Interface
- Mode 1 : RGB Interface

Following pins description table as below for pin function description.

**Table 4-1 LCD Interface Pin Descriptions**

<table>
<thead>
<tr>
<th>Ball Location</th>
<th>Pin Name</th>
<th>Type</th>
<th>Default (Reset)</th>
<th>Default (Sleep)</th>
<th>Power</th>
<th>Description</th>
</tr>
</thead>
</table>
| M8            | LCS0#    | I/O  | O(1)           | O(1)            | LOVDD | Mode 0: Chip select output for LCD1
|               |          |      |                 |                 |       | Mode 1: Serial interface enable
|               |          |      |                 |                 |       | This pin can be programmed as GPIO4. |
| H10           | LCS1#    | I/O  | O(1)           | O(1)            | LOVDD | Mode 0: Chip select output for LCD2
|               |          |      |                 |                 |       | Mode 1: No used
|               |          |      |                 |                 |       | This pin can be programmed as GPIO5. |
| N8            | LDCLK    | I/O  | O(0)           | O(0)            | LOVDD | Mode 0: No used
|               |          |      |                 |                 |       | Mode 1: Clock output
|               |          |      |                 |                 |       | This pin can be programmed as GPIO6. |
| K9            | LDE#     | I/O  | O(1)           | O(1)            | LOVDD | Mode 0: Read enable for 80CPU
|               |          |      |                 |                 |       | Read or Write enable for 68 CPU
|               |          |      |                 |                 |       | Mode 1: Data enable output
<p>|               |          |      |                 |                 |       | This pin can be programmed as GPIO7. |
| N10           | LD0      | O    | O(U)           | O(U)            | LOVDD | B0 data output |</p>
<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
<th>Power</th>
<th>Truth Value</th>
<th>Supply</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>L11</td>
<td>LD1</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>M11</td>
<td>LD2</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>N11</td>
<td>LD3</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>M12</td>
<td>LD4</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>N12</td>
<td>LD5</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>M13</td>
<td>LD6</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>L13</td>
<td>LD7</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>L12</td>
<td>LD8</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>K13</td>
<td>LD9</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>K12</td>
<td>LD10</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>K11</td>
<td>LD11</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>J13</td>
<td>LD12</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>J12</td>
<td>LD13</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>J11</td>
<td>LD14</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>J10</td>
<td>LD15</td>
<td>O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>J9</td>
<td>LD16</td>
<td>I/O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>H11</td>
<td>LD17</td>
<td>I/O</td>
<td>O(U)</td>
<td>O(U)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>L9</td>
<td>LHSYNC</td>
<td>O</td>
<td>O(1)</td>
<td>O(1)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>M9</td>
<td>LVSYNC</td>
<td>O</td>
<td>O(1)</td>
<td>O(1)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>N9</td>
<td>LSCK</td>
<td>I/O</td>
<td>O(1)</td>
<td>O(1)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>L10</td>
<td>LSDA</td>
<td>I/O</td>
<td>O(0)</td>
<td>O(0)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>M10</td>
<td>LSA0</td>
<td>I/O</td>
<td>O(0)</td>
<td>O(0)</td>
<td>LOVDD</td>
</tr>
<tr>
<td>H9</td>
<td>TVCLK</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>LOVDD</td>
</tr>
</tbody>
</table>

### 4.1. LCD Interface Implementation

#### The Highlight for LCD Panel Implement

1. **Hardware Notes**
   - a. IO power, Internal Logic power, System Power supply range.
   - b. Back Light operating Power supply (Current/Voltage)
   - c. Hardware trapping pin
   - d. Reset active low or active high.
   - e. R[5..0] connect to D[17..12], G[5..0] connect to D[11..6], B[5..0] connect to D[5..0] (18 bits, RGB IF)
   - f. R[4..0] connect to D[15..11], G[5..0] connect to D[10..5], B[4..0] connect to D[4..0] (16 bits, RGB IF)

2. **Software Notes:**
   - There are 2 parts in script file. One is fill data to Glamo 3365 registers, the others is command / data to initialize LCD panel.

2-1 Fill data in GLAMO 3365 registers:
a. Command type: LCD command fire (Reg. 11A0[15..14]=00)
b. Display width, height and pitch.
c. Display base address

2-1-1 CPU Interface
a. CS, XWR, A0(XRES) polarity setting
b. Setting DCLK divide ratio to meet panel’s bus cycle time. (DCLK divide ratio: Reg 0036 bits[7..0])
c. Program CS, XWR waveform (CS: Reg 11B0; XWR: Reg 11B2) to meet panel’s timing characteristics.
d. Setting data command header (Reg 114A)

2-1-2 RGB Interface
a. Serial interface data type: 8bits/ 9 bits/ 24bits.
b. Vsync, Hsync, DCLK, A0, SCLK polarity setting
c. Setting DCLK divide ratio to meet panel’s bus cycle time. (DCLK divide ratio: Reg 0036 bits[7..0])
d. Programming Vsync, Hsync timing to meet panel’s timing characteristics. (Reg. 111C-112C for Hsync, Reg. 1130-1140 for Vsync.)

2-2 Command data for initializing panel:
2-2-1 CPU Interface:
a. Command type: Parallel command mode (Reg. 11A0[15..14]=01)
b. Setting command format (Reg. 11A0[13..9]) to meet panel’s “write accesses to internal registers.” flowchart.
c. Fire command data step by step to LCD by following panel’s power-on sequence.

2-2-2 RGB Interface
a. Command type: Serial command mode (Reg. 11A0[15..14]=10)
b. Command data width: 8 bits/ 9 bits/ 24bits.
c. Setting command format (Reg. 11A0[13..9]) to meet panel’s “write accesses to internal registers.” flowchart.
d. Fire command data step by step to LCD by following panel’s power-on sequence.

4.1.1. 80-Type CPU Interface Implementation

The figure below illustrates the implementation for interfacing the Glamo 3365 to an 80-type CPU interface LCM.

Pins from panel: WR, RS, RD, CS, Data[X : 0](X for CPU bus)

Pins from Glamo 3365: LVSYNC, LHSYNC, LDEN, LCS0, LD[X : 0](X for CPU bus)

Before firing LCD controller, User must set the relative pins’ polarity.

LVSYNC (WR): Users can set its polarity from Register 110A : D[6 : 4] → DGRWPLTY
LHSYNC (RS): Users can set its polarity from Register 110A : D[13 : 12] → DGXRESPLTY
    00: Low for data
    01: High for data
    10: force low
11: force high

LDEN (RD): Users can set its polarity from Register 110A : D[15 : 14] → DGDEnPLTY
   00: Low active
   01: high active
   10: force low
   11: force high

LCS*(CS): Users can set its polarity from Register 110A : D[10 : 8] → DGCS0PLTY
   Users can set its polarity from Register 1102 : D[10 : 8] → DGCS1PLTY

Figure 4.1-1 Connection of Glamo 3365 to 80-type CPU interface LCM

4.1.2. 68-Type CPU Interface Implementation

The figure below illustrates the implementation for interfacing the Glamo 3365 to a 68-type CPU interface LCM.
LVSYNC (E): Users can set its polarity from Register 110A : D[6 : 4] → DGRWPLTY
LHSYNC (RS): Users can set its polarity from Register 110A : D[13 : 12] → DGXRESPLTY
   00: Low for data
   01: High for data
10: force low
11: force high

LDEN (R/W): Users can set its polarity from Register 110A : D[15 : 14] \(\rightarrow\) DGDEnPLTY

00: Low active
01: high active
10: force low
11: force high

LCS*(CS): Users can set its polarity from Register 110A : D[10 : 8] \(\rightarrow\) DGCS0PLTY

Users can set its polarity from Register 1102 : D[10 : 8] \(\rightarrow\) DGCS1PLTY

**Figure 4.1-2 Connection of Glamo 3365 to 68-type CPU interface LCM**

### 4.1.3. RGB with SPI Interface Implementation

The figure below illustrates the implementation for interfacing the Glamo 3365 to a RGB with serial interface LCM.

LVSYNC (Vsync): Users can set its polarity from Register 1100 : D[10] \(\rightarrow\) DGVsynPLTY

0: low active
1: high active

LHSYNC (Hsync): Users can set its polarity from Register 1100 : D[9] \(\rightarrow\)
DGHsyncPLTY
0: low active
1: high active

LDCLK (Dotclk): Users can set its polarity from Register 110A : D[11] →
DGDCLKPLTY
0: DCLK Rising Edge Latch data
1: DCLK Falling Edge Latch data

LDEN (DE): Users can set its polarity from Register 110A : D[15 : 14] → DGEnPLTY
00: Low active
01: high active
10: force low
11: force high

LD[17 : 0] (Data[X : 0] ): relative registers setting
1102 : D[3] → DGOutPosMode
0: Align LSB
1: Align MSB
1102 : D[2] → DGNoUseBDef
0: output 0 for no use bits
1: output 1 for no use bits

LCS*(CS): Users can set its polarity from Register 110A : D[10 : 8] → DGCS0PLTY
Users can set its polarity from Register 1102 : D[10 : 8] → DGCS1PLTY

LSA0 (A0): Users can set its polarity from Register 110A : D[7] → DAG0PLTY
0: low for data
1: high for data

LSDA (SI): data pin

LSCK (SCK): Users can set its polarity from Register 110A : D[1] → DGsCLKPLTY
0: SCLK Rising Edge Latch data
1: SCLK Falling Edge Latch data

Figure 4.1-3 Connection of Glamo 3365 to RGB with SPI interface LCM
4.1.4. RGB with Direct control Interface Implementation

The figure below illustrates the implementation for interfacing the Glamo 3365 to a RGB with direct control interface LCM. Some panels with RGB interface just have several signals to control its behavior. For these panels, GLAMO 3365 support direct control to control this pin, user can decide Ctl* pins’ polarity.

LVSYNC (Vsync): Users can set its polarity from Register 1100 : D[10] → DGVsynPLTY
  0: low active
  1: high active

LHSYNC (Hsync): Users can set its polarity from Register 1100 : D[9] → DGHsyncPLTY
  0: low active
  1: high active

LDCLK (Dotclk): Users can set its polarity from Register 110A : D[11] → DGDCLKPLTY
  0: DCLK Rising Edge Latch data
  1: DCLK Falling Edge Latch data
LDEN (DE): Users can set its polarity from Register 110A : D[15 : 14] \rightarrow DGDEnPLTY

00: Low active
01: high active
10: force low
11: force high

LD[17 : 0] (Data[X : 0]) : relative registers setting

1102 : D[3] \rightarrow DGOutPosMode
0: Align LSB
1: Align MSB

1102 : D[2] \rightarrow DGNoUseBDef
0: output 0 for no use bits
1: output 1 for no use bits

Direct Control

LCS1(Ctrl0) LCS0(Ctrl1) LSA0(Ctrl2) LSDA(Ctrl3) LSCK(Ctrl4)

Register 11A0

<table>
<thead>
<tr>
<th>D15</th>
<th>D14</th>
<th>D13</th>
<th>D12</th>
<th>D11</th>
<th>D10</th>
<th>D9</th>
<th>D8</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Ctrl0</td>
<td>Ctrl1</td>
<td>Ctrl2</td>
<td>Ctrl3</td>
</tr>
</tbody>
</table>

Users can decide which value should be assigned to Ctrl*.
Figure 4.1-4 Connection of Glamo 3365 to RGB without serial interface LCM

<table>
<thead>
<tr>
<th>LCD Controller</th>
<th>LCD panel0</th>
</tr>
</thead>
<tbody>
<tr>
<td>LVSYNC</td>
<td>Vsync</td>
</tr>
<tr>
<td>LHSYNC</td>
<td>Hsync</td>
</tr>
<tr>
<td>LDCLK</td>
<td>Dot clk</td>
</tr>
<tr>
<td>LDEN</td>
<td>DE</td>
</tr>
<tr>
<td>LD[17:0]</td>
<td>LD[17:0]</td>
</tr>
<tr>
<td>LCS0</td>
<td>Clt1</td>
</tr>
<tr>
<td>LCS1</td>
<td>Clt0</td>
</tr>
<tr>
<td>LSA0</td>
<td>Clt2</td>
</tr>
<tr>
<td>LDSA</td>
<td>Clt3</td>
</tr>
<tr>
<td>LSCK</td>
<td>Clt4</td>
</tr>
</tbody>
</table>
4.2. LCD Bypass Implementation

Glamo 3365 also support bypass mode, which is BB can send it's waveform to LCD panel without through LCD controller. In this mode, all signals sent to Glamo 3365 will be sent to LCD panel. Bypass LCD controller. Set the bit by setting write register = 0002E0h. Clear the bit by setting write register = 0002F0h.

4.2.1. 80-Type CPU Bypass Implementation

The figure below illustrates the implementation for interfacing the Glamo 3365 to an 80-type CPU interface LCM in bypass mode.

![Figure 4.2-1 Connection of Glamo 3365 to 80-type CPU interface LCM in bypass mode](image)

<table>
<thead>
<tr>
<th>Glamo Input</th>
<th>Bypass to</th>
<th>LCD Panel</th>
</tr>
</thead>
<tbody>
<tr>
<td>HWR</td>
<td>LVSYNC</td>
<td>WR</td>
</tr>
<tr>
<td>HA[1]</td>
<td>LHSYNC</td>
<td>RS</td>
</tr>
<tr>
<td>HRD</td>
<td>LDEN</td>
<td>RD</td>
</tr>
<tr>
<td>HCS</td>
<td>LCS0</td>
<td>CS</td>
</tr>
<tr>
<td>HA[2]</td>
<td>LCS1</td>
<td></td>
</tr>
</tbody>
</table>
4.2.2. **68-Type CPU Bypass Implementation**

The figure below illustrates the implementation for interfacing the Glamo 3365 to a 68-type CPU interface LCM in by pass mode.

**Figure 4.2-2 Connection of Glamo 3365 to 68-type CPU interface LCM in bypass mode**

<table>
<thead>
<tr>
<th>Glamo Input</th>
<th>Bypass to</th>
<th>LCD Panel</th>
</tr>
</thead>
<tbody>
<tr>
<td>HLB</td>
<td>LVSYNC</td>
<td>E</td>
</tr>
<tr>
<td>HA[1]</td>
<td>LHSYNC</td>
<td>RS</td>
</tr>
<tr>
<td>HRD</td>
<td>LDEN</td>
<td>R/W</td>
</tr>
<tr>
<td>HCS</td>
<td>LCS0</td>
<td>CS</td>
</tr>
<tr>
<td>HA[2]</td>
<td>LCS1</td>
<td></td>
</tr>
</tbody>
</table>
5. Video Capture Interface Description

5.1. Introduction

This chapter describes and lists Camera interface pings on Glamo 3365. Glamo 3365 support 8 bit data bus only, check MSB/LSB output data swap and 10 bit to 8 bit connection from camera datasheet before connect camera bus to Glamo 3365.

For camera output data format, Glamo 3365 provides 2 kinds data format for input, one is Raw RGB 8 bit (Bayer pattern) type, the other is YUV4:2:2 format. Since Glamo 3365 provides powerful ISP and supporting higher resolution camera sensor, recommend Bayer pattern for input format, Maximum supporting resolution as below.

5.2. Camera Interface Pin Definition

Camera interface contains total 18 pins. Detail pin information is presented in Table 4.2-1.

<table>
<thead>
<tr>
<th>Ball Location</th>
<th>Pin Name</th>
<th>Type</th>
<th>Default (Reset)</th>
<th>Default (Sleep)</th>
<th>Power</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E13</td>
<td>PCLK</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td>Pixel clock from sensor</td>
</tr>
<tr>
<td>D11</td>
<td>SCLK</td>
<td>O</td>
<td>highz</td>
<td>highz</td>
<td>COVDD</td>
<td>Output clock to sensor</td>
</tr>
<tr>
<td>D12</td>
<td>CSEN</td>
<td>O</td>
<td>highz</td>
<td>highz</td>
<td>COVDD</td>
<td>Serial interface enable</td>
</tr>
<tr>
<td>D13</td>
<td>CSCL</td>
<td>O</td>
<td>highz</td>
<td>highz</td>
<td>COVDD</td>
<td>Serial interface clock</td>
</tr>
<tr>
<td>C12</td>
<td>CSDA</td>
<td>I/O</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td>Serial interface data input/output</td>
</tr>
<tr>
<td>E9</td>
<td>CSGPO0</td>
<td>I/O</td>
<td>O(0)</td>
<td>I/Oper</td>
<td>COVDD</td>
<td>Sensor data input D[0] Serial interface general purpose output This pin can be programmed as GPIO13.</td>
</tr>
<tr>
<td>D9</td>
<td>CSGPO1</td>
<td>I/O</td>
<td>O(0)</td>
<td>I/Oper</td>
<td>COVDD</td>
<td>Sensor data input D[1] Serial interface general purpose output This pin can be programmed as GPIO14.</td>
</tr>
<tr>
<td>C13</td>
<td>CHSYNC</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td>Horizontal sync signal</td>
</tr>
<tr>
<td>B12</td>
<td>CVSYNC</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td>Vertical sync signal</td>
</tr>
<tr>
<td>B13</td>
<td>CD0</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td>Sensor data input D[9:2]</td>
</tr>
<tr>
<td>A12</td>
<td>CD1</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td></td>
</tr>
<tr>
<td>B11</td>
<td>CD2</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td></td>
</tr>
<tr>
<td>A11</td>
<td>CD3</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td></td>
</tr>
<tr>
<td>D10</td>
<td>CD4</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td></td>
</tr>
<tr>
<td>C10</td>
<td>CD5</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td></td>
</tr>
<tr>
<td>B10</td>
<td>CD6</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td></td>
</tr>
<tr>
<td>A10</td>
<td>CD7</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>COVDD</td>
<td></td>
</tr>
<tr>
<td>C9</td>
<td>FLCTL</td>
<td>I/O</td>
<td>O(0)</td>
<td>O(0)/Oper</td>
<td>COVDD</td>
<td>Flash light control output This pin can be programmed as GPIO15.</td>
</tr>
</tbody>
</table>
5.3. Video Capture Interface Implementation

Video Capture supports two modes for connecting sensor

- 1. When sensor provides a main clock input for sensor operation and a pixel clock out for latch pixel data, Glamo 3365 supports mode 1. (Such as Omnivision OV9640/7648, Agilent ADM-2700-0000, PixArt PAS302, Samsung S5X433CA, Sanyo IGT99268F, Toshiba TCM8210MD)

- 2. When sensor provides only one clock input for either sensor operation or latch pixel data, Glamo 3365 supports mode 2. (Such as Bimorphic Bi8831A)

---

**Figure 5.3-1 Connection of Glamo 3365 to Sensor (Mode 1)**

**Figure 5.3-2 Connection of Glamo 3365 to Sensor (Mode 2)**
5.4. The Highlight of Image Sensor Note

1. **HW Settings for Power on sequence**
   a. Power supply voltage
   b. VDDIO, VDDIN, RESET sequence
   c. RESET, STDBY, PWRDN active level

2. **SW Settings for Power on sequence**
   a. Serial interface protocol
      ①. Slave address
      ②. CMD and DATA bus width
      ③. With or Without register block switch
      ④. 3 wire or 2 wire interface
   b. Serial bus timing specification
      ①. SCL clock frequency
      ②. Hold time (repeated) START condition.
      ③. After this period, the first clock pulse is generated.
④. Low period of the SCL clock
⑤. HIGH period of the SCL clock
⑥. Set-up time for a repeated START condition
⑦. Data hold time. For I2C-bus device
⑧. Data set-up time
⑨. Set-up time for STOP condition
⑩. Bus free time between a STOP and START

c. MCLK for input
d. Output format
e. Output size
f. HSYNC, VSYNC polarity
g. Order of recommended initial registers

Specific register for update
6. Audio Interface Description

Glamo 3365 connect audio codec by I2S serial bus and control audio codec by I2C serial bus. Driver default programming DGPIO0 be I2C’s SCLK function and DGPIO1 be I2C SDA function, and Smedia will provide I2C sample code to customers, customers have to control audio codec by themselves. Please see below for evaluation board reference connection and the on EV board codec is Wolfson WM8978. Hard wired Glamo 3365 and audio codec for voice and multimedia function. When voice mode, BaseBand controls codec, Glamo 3365 I2S bus is tri-state. When multimedia mode, Glamo 3365 controls audio codec, baseband I2S bus is tri-state.

<table>
<thead>
<tr>
<th>Ball Location</th>
<th>Pin Name</th>
<th>Type</th>
<th>Default (Reset)</th>
<th>Default (Sleep)</th>
<th>Power</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>N6</td>
<td>AMCLK</td>
<td>O</td>
<td>O(0)</td>
<td>O(0)</td>
<td>UOVDD</td>
<td>Audio master clock</td>
</tr>
<tr>
<td>M4</td>
<td>ACLK</td>
<td>I/O</td>
<td>O(0)</td>
<td>O(0)</td>
<td>UOVDD</td>
<td>I^2S audio-in clock</td>
</tr>
<tr>
<td>N5</td>
<td>AWS</td>
<td>I/O</td>
<td>O(0)</td>
<td>Oper</td>
<td>UOVDD</td>
<td>I^2S audio-in word select</td>
</tr>
<tr>
<td>M5</td>
<td>ASDIN</td>
<td>I</td>
<td>-</td>
<td>-</td>
<td>UOVDD</td>
<td>I^2S audio-in serial data</td>
</tr>
<tr>
<td>M6</td>
<td>ZCLK</td>
<td>I/O</td>
<td>O(0)</td>
<td>O(0)</td>
<td>UOVDD</td>
<td>I^2S audio-out clock</td>
</tr>
<tr>
<td>L6</td>
<td>ZWS</td>
<td>I/O</td>
<td>O(0)</td>
<td>Oper</td>
<td>UOVDD</td>
<td>I^2S audio-out word select</td>
</tr>
<tr>
<td>K6</td>
<td>ZSDOUT</td>
<td>I/O</td>
<td>O(0)</td>
<td>O(0)/Oper</td>
<td>UOVDD</td>
<td>I^2S audio-out serial data</td>
</tr>
</tbody>
</table>

This pin can be programmed as GPIO20.

The LRC signal is low for the left channel time slot and high for the right channel time slot, and the LRC signal is synchronous to the falling edge of the BCLK. Serial data is latching on the rising edge of BCLK. There is a one BCLK cycle delay from the edge of the LRC before the MSB of the data is latching for both the left channel and the right channel.

For the I2S mode of the codec port interface, there is a 16-bit transmit and a 16-bit receive shift register for each SDOUT and SDIN signal. The interface automatically fills the unused bits with zeros. Serial data is transmitted in twos-complement with the MSB first.

The LRC signal is low for the left channel time slot and high for the right channel time slot, and the LRC signal is synchronous to the falling edge of the BCLK. Serial data is latching on the rising edge of BCLK. There is a one BCLK cycle delay from the edge of the LRC before the MSB of the data is latching for both the left channel and the right channel.

For the I2S mode of the codec port interface, there is a 16-bit transmit and a 16-bit receive shift register for each SDOUT and SDIN signal. The interface automatically fills the unused bits with zeros. Serial data is transmitted in twos-complement with the MSB first.

Figure 6-1 I2S Audio Interface

Figure 6-2 Connection of Glamo 3365 to Audio Codec
7. **Layout and PCB**

1. About the PCB Format of the layer, ball, VIA and trace. The following is the illustration:

<table>
<thead>
<tr>
<th>PCB Layers</th>
<th>Trace Width / Space</th>
<th>Ball Pad</th>
<th>VIA Pad Ring/Hole</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 Layer</td>
<td>5/4 mil</td>
<td>9 mil</td>
<td>2/8 mil</td>
</tr>
</tbody>
</table>

**M3365 PCB Design Rule**

* (0.5 mm Ball Pitch)

<table>
<thead>
<tr>
<th>Signal</th>
<th>Layer-1</th>
<th>Some of VIA use the blind hold via on layer 1 and 2.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signal</td>
<td>Layer-2</td>
<td></td>
</tr>
<tr>
<td>GND</td>
<td>Layer-3</td>
<td></td>
</tr>
<tr>
<td>Signal</td>
<td>Layer-4</td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td>Layer-5</td>
<td></td>
</tr>
<tr>
<td>Signal</td>
<td>Layer-6</td>
<td></td>
</tr>
</tbody>
</table>

Figure: Six Layer Routing Scheme

2. Show the M365 ball and VIA of the placement with some detailed explanation.(Drawing for top view):
Note: The placement includes the blind hole on the VIA.

Detail “A”

PAD  VIA for Blind Hole  VIA for Through Hole

Detail “B”

3.5 mil
3. The PCB standard rout for every layers (All drawings for top view).

A: Layer 1 (Top Layer)

B: Layer 2 (Signal layer)
C: Layer 3 (GND Layer)

D: Layer 4 (Signal Layer)
E: Layer 5 (Power Layer—IVDD & MOVDD)

![MOVDD power plan for M365 on the area](image)

![IVDD power plan for M365 on the area](image)

F: Layer 6 (Bottom Layer)

![PCB Power Layer (Layer 5)](image)

![PCB Signal Layer (Layer 6, Bottom Layer)](image)

Figure 7.1 Glamo 3365 Layout Rule
8. Power Delivery

8.1. Power Consumption

The typical value for reference only and it were measured on Glamo 3365 EVB v1.0. (Test Result V1.4-20060728)

<table>
<thead>
<tr>
<th>Item</th>
<th>Application</th>
<th>Core (µA) 1.6V</th>
<th>SRAM (µA) 1.8V</th>
<th>Total (µW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Deep Power Down mode</td>
<td>58.5</td>
<td>2.1</td>
<td>97</td>
</tr>
<tr>
<td>2</td>
<td>LCD Bypass mode</td>
<td>58.9</td>
<td>75.7</td>
<td>231</td>
</tr>
<tr>
<td>3</td>
<td>Standby mode (no LCD refresh)</td>
<td>58.4</td>
<td>75.6</td>
<td>230</td>
</tr>
<tr>
<td>5</td>
<td>Full Static Display mode</td>
<td>6100</td>
<td>4100</td>
<td>17140</td>
</tr>
<tr>
<td>6</td>
<td>2D Game application (2D+LCD+SRAM)</td>
<td>9500</td>
<td>7670</td>
<td>29006</td>
</tr>
<tr>
<td>7</td>
<td>3D Game application (3D+2D+LCD+SRAM)</td>
<td>24892</td>
<td>16110</td>
<td>68825</td>
</tr>
<tr>
<td>8</td>
<td>Camera preview only with QVGA 15 f/s</td>
<td>16018</td>
<td>7810</td>
<td>39687</td>
</tr>
<tr>
<td>9</td>
<td>MPEG4 CIF 15 f/s encoder</td>
<td>28039</td>
<td>11030</td>
<td>64716</td>
</tr>
<tr>
<td>10</td>
<td>MPEG4 CIF 30 f/s encoder</td>
<td>39017</td>
<td>16360</td>
<td>91875</td>
</tr>
<tr>
<td>11</td>
<td>MPEG4 QCIF 15 f/sencoder</td>
<td>26111</td>
<td>9200</td>
<td>58338</td>
</tr>
<tr>
<td>12</td>
<td>MPEG4 QCIF 30 f/s encoder</td>
<td>35058</td>
<td>14560</td>
<td>82301</td>
</tr>
<tr>
<td>13</td>
<td>MPEG4 CIF 15 f/s decode and playback</td>
<td>18902</td>
<td>11190</td>
<td>50385</td>
</tr>
<tr>
<td>14</td>
<td>MPEG4 CIF 30 f/s decode and playback</td>
<td>23641</td>
<td>19490</td>
<td>72908</td>
</tr>
<tr>
<td>15</td>
<td>MPEG4 QCIF 15 f/s decode and playback</td>
<td>18023</td>
<td>10370</td>
<td>47503</td>
</tr>
<tr>
<td>16</td>
<td>MPEG4 QCIF 30 f/s decode and playback</td>
<td>21704</td>
<td>15530</td>
<td>62680</td>
</tr>
<tr>
<td>17</td>
<td>MPEG4 CIF 15 f/s encoder + AMR record</td>
<td>37533</td>
<td>9600</td>
<td>77333</td>
</tr>
<tr>
<td>18</td>
<td>MPEG4 CIF 30 f/s encoder + AMR record</td>
<td>45012</td>
<td>13940</td>
<td>97111</td>
</tr>
<tr>
<td>19</td>
<td>MPEG4 QCIF 15 f/s encoder + AMR record</td>
<td>30957</td>
<td>6830</td>
<td>61825</td>
</tr>
<tr>
<td>20</td>
<td>MPEG4 QCIF 30 f/s encoder + AMR record</td>
<td>37652</td>
<td>9530</td>
<td>77397</td>
</tr>
<tr>
<td>21</td>
<td>MPEG4 CIF 15 f/s decode + AMR decode</td>
<td>35666</td>
<td>16780</td>
<td>87270</td>
</tr>
<tr>
<td>22</td>
<td>MPEG4 CIF 30 f/s decode + AMR decode</td>
<td>37389</td>
<td>20460</td>
<td>96650</td>
</tr>
<tr>
<td>23</td>
<td>MPEG4 CIF 15 f/s decode + AAC decode</td>
<td>34603</td>
<td>16680</td>
<td>85389</td>
</tr>
<tr>
<td>24</td>
<td>MPEG4 CIF 30 f/s decode + AAC decode</td>
<td>36313</td>
<td>19750</td>
<td>93651</td>
</tr>
<tr>
<td>25</td>
<td>MP3 playback + No Display</td>
<td>22688</td>
<td>8500</td>
<td>51601</td>
</tr>
<tr>
<td>26</td>
<td>MP3 playback + Partial Display</td>
<td>23883</td>
<td>8830</td>
<td>54107</td>
</tr>
<tr>
<td>27</td>
<td>MP3 playback + Display (60 fps)</td>
<td>23762</td>
<td>10110</td>
<td>56217</td>
</tr>
<tr>
<td>Item</td>
<td>Application</td>
<td>Core (uA)</td>
<td>SRAM (uA)</td>
<td>Total (uW)</td>
</tr>
<tr>
<td>------</td>
<td>---------------------------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.6V</td>
<td>1.8V</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>AAC playback + No Display</td>
<td>24692</td>
<td>5460</td>
<td>49335</td>
</tr>
<tr>
<td>29</td>
<td>AAC playback + Partial Display</td>
<td>25833</td>
<td>5870</td>
<td>51899</td>
</tr>
<tr>
<td>30</td>
<td>AAC playback + Display (60 fps)</td>
<td>25875</td>
<td>7750</td>
<td>55350</td>
</tr>
<tr>
<td>31</td>
<td>MIDI playback + No Display</td>
<td>21835</td>
<td>8340</td>
<td>49948</td>
</tr>
<tr>
<td>32</td>
<td>MIDI playback + Partial Display</td>
<td>22953</td>
<td>8650</td>
<td>52295</td>
</tr>
<tr>
<td>33</td>
<td>MIDI playback + Display (60 fps)</td>
<td>23150</td>
<td>9970</td>
<td>54986</td>
</tr>
<tr>
<td>34</td>
<td>AMR playback + No Display</td>
<td>23417</td>
<td>3910</td>
<td>44505</td>
</tr>
<tr>
<td>35</td>
<td>AMR playback + Partial Display</td>
<td>24629</td>
<td>4340</td>
<td>47218</td>
</tr>
<tr>
<td>36</td>
<td>AMR playback + Display (60 fps)</td>
<td>24841</td>
<td>6070</td>
<td>50672</td>
</tr>
<tr>
<td>37</td>
<td>AMR record + No Display</td>
<td>22300</td>
<td>6040</td>
<td>46552</td>
</tr>
<tr>
<td>38</td>
<td>AMR record + Partial Display</td>
<td>23486</td>
<td>6450</td>
<td>49188</td>
</tr>
<tr>
<td>39</td>
<td>AMR record + Display (60 fps)</td>
<td>23500</td>
<td>8080</td>
<td>52144</td>
</tr>
<tr>
<td>40</td>
<td>AMR duplex + No Display</td>
<td>22842</td>
<td>6860</td>
<td>48895</td>
</tr>
<tr>
<td>41</td>
<td>AMR duplex + Partial Display</td>
<td>23975</td>
<td>7240</td>
<td>51392</td>
</tr>
<tr>
<td>42</td>
<td>AMR duplex + Display (60 fps)</td>
<td>24172</td>
<td>8850</td>
<td>54605</td>
</tr>
</tbody>
</table>
8.2. **Power on/off Procedure**

When power on, supply all of VDD to each block and make sure the voltage level is correctly and stable.

After voltage stable, you must reset Glamo 3365 hardware by RST# pin.

PLL and CLK setting have to match host CPU timing.

Glamo 3365 S/W driver included power on/off control procedure, customers don’t have to detail control it.
8.3. Power Saving

Clock power consumption is classified to 5 levels:

- Level 1: Supply normal clock signals.
- Level 2: Reduce clock frequency but continue to supply clock signals.
- Level 3: Gate off clock signals after the clocks are generated from clock dividers.
- Level 4: Disable clock divider but PLL is still working.
- Level 5: Disable PLL.

- **Gating clock:**
  - Level 1 to level 5 can be controlled by SW drivers.
  - For saving power more efficiently at level 3, it is better to dynamically turn on/off clock by HW.

Glamo 3365 S/W driver included power saving control procedure, customers don’t have to detail control it.

8.4. Clock Tree

<table>
<thead>
<tr>
<th>Clock Name</th>
<th>Description</th>
<th>Clock Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCLK</td>
<td>Host bus clock</td>
<td>CCLK</td>
<td>Image signal processor 1 clock</td>
</tr>
<tr>
<td>DCLK</td>
<td>LCD controller clock</td>
<td>DCLK</td>
<td>LCD controller high frequency clock</td>
</tr>
<tr>
<td>DMCLK</td>
<td>LCD controller middle frequency clock</td>
<td>E0CLK</td>
<td>3D engine front clock</td>
</tr>
<tr>
<td>E1CLK</td>
<td>3D engine back clock</td>
<td>E1CLK</td>
<td>2D engine clock</td>
</tr>
<tr>
<td>HCLK</td>
<td>Host bus clock for iBurst mode</td>
<td>ICLK</td>
<td>Micro processor 0 clock</td>
</tr>
<tr>
<td>HCLK</td>
<td>Image signal processor 2 clock</td>
<td>JCLK</td>
<td>JPEG engine clock</td>
</tr>
<tr>
<td>KCLK</td>
<td>System clock for timer in microprocessor</td>
<td>M0CLK</td>
<td>Memory clock in host bus controller</td>
</tr>
<tr>
<td>M1CLK</td>
<td>Memory clock in memory controller</td>
<td>M2CLK</td>
<td>Memory clock in image signal processor 1 and 2</td>
</tr>
<tr>
<td>M3CLK</td>
<td>Memory clock in JPEG engine</td>
<td>M4CLK</td>
<td>Memory clock in MPEG engine</td>
</tr>
<tr>
<td>M5CLK</td>
<td>Memory clock in LCD controller</td>
<td>M6CLK</td>
<td>Memory clock in command queue</td>
</tr>
<tr>
<td>M7CLK</td>
<td>Memory clock in 2D engine</td>
<td>M8CLK</td>
<td>Memory clock in 3D engine</td>
</tr>
<tr>
<td>M9CLK</td>
<td>Memory clock in MMC controller</td>
<td>MCLKD</td>
<td>Delayed memory clock to sample write data</td>
</tr>
<tr>
<td>MCLKSTROBE</td>
<td>Delayed memory clock to strobe read data</td>
<td>MOCACLK</td>
<td>Memory output clock for SDRAM</td>
</tr>
<tr>
<td>SCLK</td>
<td>Clock to sensor</td>
<td>TCLK</td>
<td>MMC controller clock</td>
</tr>
</tbody>
</table>
Figure 8.4-1 Glamo 3365 Clock Tree

[Diagram showing the clock tree with various components and labels such as PLL1, PLL2, OSCI, B0PLL, B0PLLYBypass2, MUX, DIV, Gating, CCLK, SCLK, DCLK, DMCLK, DHCLK, TCLK, FCLK, BCLK, KCLK, and HCLK.]
9. How to initiate

All of initial sequence for hardware level testing and make sure component workable only, after SDK driver porting successful, driver will take all of hardware low-level control, and customer control Glamo 3365 by API that Smedia provided.

9.1. Initiate Glamo 3365 and Host Bus

Please make sure baseband (co-processor) host bus is working well first, before start initial Glamo 3365. Check Host can read/write system SRAM/DRAM successful to make sure system host bus circuit, clock, bus timing… are correct. Then you can identify which one is problem, if Glamo 3365 is not workable or something wrong.

Application Example

Following is a sequence of register-write for initiate Glamo 3365, you can integrate it in Boot loader for initiate Glamo. After PCB board ready, please read device ID register 0x0002 for bus testing to check circuit correct or not, when you read back 0x3365 that means Glamo 3365 workable and the bus read successful, you don’t need to initial Glamo 3365 first when read Glamo chip ID. The physical address is base address + register address, after initial chip and LCD panel, you will see gray pattern on LCD panel.

```
//Register_write( register address , data)
//Reset Module
//Initial Glamo3365 script (v3)
//0xfffe, x --> means delay x ms
static unsigned short init365[] ={
  0x12, 0x300a, 0x14, 0x10aa, 0x16, 0x100a, 0x18, 0x32aa, 0x1A, 0x100a, 0x1C, 0x302a, 0x1E, 0x302a, 0x16, 0x3aaa, 0x20, 0x12aa, 0x22, 0x2a, 0x24, 0x3aaa, 0x26, 0x12aa, 0xfffe, 5, //0x10, 0x0008,
  0x10, 0x000d, 0x12, 0xa, 0x14, 0xee, 0x16, 0xa, 0x18, 0x2aa, 0x1A, 0xa, 0x1C, 0x2a, 0x1E, 0x2a, 0x16, 0x3aaa, 0x22, 0x2a, 0x24, 0x0aaa, 0x26, 0x2aa, 0xfffe, 5, 0x40, 0x0588, //PLL1=36Mhz OSCI = 13MHz For 3365
  0x44, 0x0a27, //PLL2=66Mhz OSCI = 13MHz For 3365
```

//0x40, 0x0462, //PLL1=36Mhz OSCl = 32kHz For 3365
//0x44, 0x0809, //PLL2=66Mhz OSCl = 32kHz For 3365

//0x40, 0x0762, //PLL1=48Mhz OSCl = 13MHz For 3362
//0x44, 0x0C4E, //PLL2=80Mhz OSCl = 13MHz For 3362

//0x40, 0x05DB, //PLL1=48Mhz OSCl = 32kHz For 3362
//0x44, 0x09C3, //PLL2=80Mhz OSCl = 32kHz For 3362

0xfffe, 5,
0x30, 0x18ff,
0x32, 0x051f,
0x0034, 0x2000,
0x0036, 0x0105,
0x0038, 0x0100,
0x003c, 0x0017,
0x003e, 0x0017,
0x200, 0x0e00,
0x202, 0x07ff,
0x0206, 0x0080,
0x2008, 0x0244,
0x200a, 0x0600,
0x0216, 0xff0e,
//0x0300, 0x0874, //VRAM 8M
//0x0300, 0x0873, //VRAM 4M
0x0300, 0x0972, //VRAM 2M
0x0302, 0xafaf,
0x0304, 0x0108,
0x0306, 0x0010,
0x0308, 0x0000,
0x030a, 0x0000,
0x030c, 0x0000,
0x030e, 0x0000,
0x0310, 0x0000,
0x0312, 0x1002,
0x0314, 0x6006,
0x0316, 0x00ff,
0x0318, 0x0001,
0x031a, 0x0020,
0x031c, 0x0000,
0x334, 0x00,
0xfffe, 1,
0x334, 0x1000,
0xfffe, 1,
0x334, 0xe100,
0x336, 0x01d6,
9.2. **Initiate LCD**

There are 2 parts in setting procedure. One is fill data to Glamo 3365 LCD controller registers, the other is command data to initialize LCD panel that recommend control it by I2C or SPI bus. The detail setting procedure is shown in example.

### 9.2.1. Fill Data in Glamo 3365 Registers

1. Command type: LCD command fire (Reg. 11A0[15..14] = 00)
2. Display width, height and pitch.
3. Display base address

#### 3.1. CPU Interface

1. CS, XWR, A0(XRES) polarity setting
2. Setting DCLK divide ratio to meet panel’s bus cycle time. (DCLK divide ratio: Reg 0036 bits[7..0])
3. Program CS, XWR waveform (CS: Reg 11B0; XWR: Reg 11B2) to meet panel’s timing characteristics.
4. Setting data command header (Reg 114A)

#### 3.2. RGB Interface

1. Serial interface data type: 8bits/ 9 bits/ 24 bits.
2. Vsync, Hsync, DCLK, A0, SCLK polarity setting.
3. Setting DCLK divide ratio to meet panel’s bus cycle time. (DCLK divide ratio: Reg 0036 bits[7..0])
4. Programming Vsync, Hsync timing to meet panel’s timing characteristics. (Reg. 111C-112C for Hsync, Reg. 1130-1140 for Vsync.)

### 9.2.2. Command Data for Initial Panel

#### 1. CPU Interface

1.1. Command type: Parallel command mode (Reg. 11A0[15..14]=01)
1.1.2. Setting command format (Reg. 11A0[13..9]) to meet panel’s “write accesses to internal registers.”

**Flowchart.**

1.1.3. Fire command data step by step to LCD by following panel’s power-on sequence.

#### 1.2. RGB Interface

1.2.1. Command type: Serial command mode (Reg. 11A0[15..14]=10)
1.2.2. Command data width: 8 bits/ 9 bits/ 24 bits.
1.2.3. Setting command format (Reg. 11A0[13..9]) to meet panel’s “write accesses to internal registers.”

**Flowchart.**

1.2.4. Fire command data step by step to LCD by following panel’s power-on sequence.

### Application Example

© Example: SONY 506

<!--CPU IF 565-->

#This is a demo of initiate script file

#Example → write(11a0, 0011) write data(0011h) to 0x11a0(physical address = base address + 11a0 = 0800_11a0 on EVB)

step on

# Control procedures

write(11a0, 0011) # command fire
write(1100, 2004) # mode setting
write(1102, 0020) # single buffer
write(1104, 0230) # CPU IF, RGB565, 16bits
write(1106, 00B0) # width 176
write(1108, 00DC) # height 220
write(110a, C000) # AC characteristic setting
write(110c, 0000) #allocated address
write(110e, 8001) #allocated address
write(1118, 0160) # display pitch (176x2)

write(11b0, 010c) # CS timing setting
write(11b2, 0206) # RWtiming setting

frame_write(test.out,0) # write BMP file to BASE A
wait(200)# wait 200 ms

# LCD panel initialization
write(11a0, 4c14) # Command to LCD panel; Software reset.
write(11a0, 4801) # Data to LCD panel
write(11a0, 4c14) # Command to LCD panel; Release reset.
write(11a0, 4800) # Data to LCD panel
write(11a0, 4c11) # Command to LCD panel; Set Async mode
write(11a0, 4800) # Data to LCD panel
write(11a0, 4c12) # Command to LCD panel; Define bus mode, 64k color.
write(11a0, 4801) # Data to LCD panel
write(11a0, 4c0d) # Command to LCD panel; Selector activator disabled.
write(11a0, 4810) # Data to LCD panel
write(11a0, 4c11) # Command to LCD panel; Awake mode.
write(11a0, 4880) # Data to LCD panel
wait(120)       # Wait for 120msec.
write(11a0, 0000) # LCD display fire
step off
<RGB IF 8bits>
step on
write(11a0, 0015)  # command fire
write(1100, 2004) # mode setting
write(1102, 0020) # single buffer
write(1104, 0b40) # RGB IF, RGB666, 18bits
write(1106, 00B0) # width 176
write(1108, 00DC)  # height 220
write(110a, 0200) # AC characteristic setting
write(1118, 0160) # display pitch (176x2)
write(111c, 00B8) # total 184
write(1120, 0000) # HR start 0
write(1124, 0003) # HR end 3(sync width 3)
write(1128, 0004) # HD start 6(Back porch 1)
write(112c, 00B4) # HD end 180(H 176 pixels)  # display range
write(1130, 00E7) # V total 231
write(1134, 0000) # VR start 0
write(1138, 0003) # VR end 3(sync width 3)
write(113C, 0009) # VD start 9(Back porch 6)
write(1140, 00E5) # VD end 229(V 220 pixels)

frame_write(test.out,0) # write BMP file to BASE A
wait(200) # wait 200 ms

# LCD panel initialization
write(11a0, 8c14) # Command to LCD panel; Software reset.
write(11a0, 8801) # Data to LCD panel
write(11a0, 8c14) # Command to LCD panel; Release reset.
write(11a0, 8800) # Data to LCD panel
write(11a0, 8c11) # Command to LCD panel; Set Sync mode.
write(11a0, 8801) # Data to LCD panel
write(11a0, 8c0b) # Command to LCD panel; Set H back porch.
write(11a0, 8801) # Data to LCD panel
write(11a0, 8c0c) # Command to LCD panel; Set V back porch.
write(11a0, 8806) # Data to LCD panel
write(11a0, 8c0d) # Command to LCD panel; Selector activator disabled
write(11a0, 8810) # Data to LCD panel
write(11a0, 0001) # Command to LCD panel; LCD display fire. Start sync signal.
write(11a0, 8c11) # Command to LCD panel; Awake mode.
write(11a0, 8881) # Data to LCD panel
wait(120)

write(11a0, 0001)
write(11a0, 0000)
#write(1153, 6000)
9.3. **Initiate CCD/CMOS Sensor**

There are 2 parts in setting procedure. One is fill data to Glamo 3365 registers for capture data setting, after this sequence, there is a preview window is created in LCD panel. The other is command data to initialize CMOS/CCD sensor, you have to initial it by I2C bus first. The detail setting procedure is shown in example.

```c
// Create preview window (This procedure for RAW data input only, if there is nothing in preview window, please modify register 0x406 to 0004 or 0008 or 000c)
Register_write(0x400,0002)
Register_write(0x402,100f)
Register_write(0x404,00f1)
Register_write(0x406,0000)  // b3: Invert the Hsync signal.
Register_write(0x408,0000)
Register_write(0x40A,0000)
Register_write(0x40C,0000)  // Horizontal data start 0
Register_write(0x40E,013f)  // Horizontal data end 319
Register_write(0x410,0002)  // Vertical data start 2
Register_write(0x412,00f1)   // Vertical data end 241
Register_write(0x414,0000)
Register_write(0x416,0000)
```

wait(200)

#frame_read(test.out,0)

#wait(100)
Register_write(0x418,0000)
Register_write(0x41A,0000)
Register_write(0x41C,6002)
Register_write(0x41E,0004)
Register_write(0x420,0d7a)
Register_write(0x422,0006)
Register_write(0x424,00cc)
Register_write(0x426,0000)
Register_write(0x428,0000)
Register_write(0x42A,00ec)
Register_write(0x42C,0134)
Register_write(0x42E,0010)
Register_write(0x430,0000)
Register_write(0x432,0198)
Register_write(0x434,0000)
Register_write(0x436,0000)
Register_write(0x438,4a52)
Register_write(0x43A,4040)
Register_write(0x43C,091b)
Register_write(0x43E,0925)
Register_write(0x440,0000)
Register_write(0x442,0048)
Register_write(0x444,010c)
Register_write(0x446,020c)
Register_write(0x448,02a8)
Register_write(0x44A,0370)
Register_write(0x44C,03b8)
Register_write(0x44E,03dc)
Register_write(0x450,03e8)
Register_write(0x452,0000)
Register_write(0x454,0000)
Register_write(0x456,091b)
Register_write(0x458,0925)
Register_write(0x45A,0000)
Register_write(0x45C,0048)
Register_write(0x45E,0100)
Register_write(0x460,0208)
Register_write(0x462,02a8)
Register_write(0x464,0370)
Register_write(0x466,03b8)
Register_write(0x468,03dc)
Register_write(0x46A,03c8)
Register_write(0x46C,0000)
Register_write(0x46E,0000)
Register_write(0x470,091b)
Register_write(0x472,0925)
Register_write(0x474,0000)
Register_write(0x476,0048)
Register_write(0x478,0100)
Register_write(0x47A,0208)
Register_write(0x47C,02a8)
Register_write(0x47E,0370)
Register_write(0x480,03b8)
Register_write(0x482,03dc)
Register_write(0x484,03c8)
Register_write(0x486,0000)
Register_write(0x488,0000)
Register_write(0x48A,091b)
Register_write(0x48C,0925)
Register_write(0x48E,0000)
Register_write(0x490,0048)
Register_write(0x492,010c)
Register_write(0x494,0210)
Register_write(0x496,02b0)
Register_write(0x498,0374)
Register_write(0x49A,03b8)
Register_write(0x49C,03c8)
Register_write(0x49E,03e8)
Register_write(0x4A0,0000)
Register_write(0x4A2,0048)
Register_write(0x4A4,0000)
Register_write(0x4A6,0000)
Register_write(0x4A8,0000)
Register_write(0x4AA,0000)
Register_write(0x4AC,0000)
Register_write(0x4AE,0000)
Register_write(0x4B0,0000)
Register_write(0x4B2,0000)
Register_write(0x4B4,0000)
Register_write(0x4B6,0000)
Register_write(0x4B8,0000)
Register_write(0x4BA,0000)
Register_write(0x4BC,0000)
Register_write(0x4BE,0000)
Register_write(0x4C0,0000)
Register_write(0x4C2,0000)
Register_write(0x4C4,0000)
Register_write(0x4C6,0000)
Register_write(0x4C8,0000)
Register_write(0x4CA,0000)
Register_write(0x4CC,0000)
Register_write(0x4CE,0000)
Register_write(0x4DO,0000)
Register_write(0x4D2,0000)
Register_write(0x4D4,0000)
Register_write(0x4D6,0000)
Register_write(0x4D8,0000)
Register_write(0x4DA,0000)
Register_write(0x4DC,0000)
Register_write(0x4DE,0000)
Register_write(0x4E0,0000)
Register_write(0x4E2,0000)
Register_write(0x4E4,0000)
Register_write(0x4E6,0000)
Register_write(0x4E8,0000)
Register_write(0x4EA,0000)
Register_write(0x4EC,0000)
Register_write(0x4EE,0000)
Register_write(0x4F0,0000)
Register_write(0x4F2,0000)
Register_write(0x4F4,0000)
Register_write(0x4F6,0030)
Register_write(0x4F8,0000)
Register_write(0x4FC,0001)
Register_write(0x4FE,0001)
Register_write(0x502,0000)
Register_write(0x504,0040)
Register_write(0x506,0000)
Register_write(0x508,c004)
Register_write(0x50A,0008)
Register_write(0x50C,aae4)
Register_write(0x50E,000A)
Register_write(0x510,0000)
Register_write(0x512,0000)
Register_write(0x514,0000)
Register_write(0x516,0000)
Register_write(0x518,0000)
Register_write(0x51A,0000)
Register_write(0x51C,0000)
Register_write(0x51E,0198)
Register_write(0x520,00f6) //ISP input height (12:0) = 240+6
Register_write(0x522,0160) //ISP input width (12:0) = 320+32
Register_write(0x524,0000)
Register_write(0x526,0000)
Register_write(0x528,0000)
Register_write(0x52A,0000)
Register_write(0x52C,8080)
Register_write(0x52E,2526)
Register_write(0x530,2020)
Register_write(0x532,9023)
Register_write(0x534,AFE2)
Register_write(0x536,0000)
Register_write(0x538,0000)
Register_write(0x53A,0000)
Register_write(0x53C,8100)
Register_write(0x53E,0000)
Register_write(0x540,8660)
Register_write(0x542,0000)
Register_write(0x544,00F5)
Register_write(0x546,2896)
Register_write(0x548,0000)
Register_write(0x54A,7810)
Register_write(0x54C,0000)
Register_write(0x54E,0000)
Register_write(0x550,0000)
Register_write(0x552,0000)
Register_write(0x554,0000)
Register_write(0x556,0000)
Register_write(0x558,0000)
Register_write(0x55A,0000)
Register_write(0x55C,019C)
Register_write(0x55E,1FAc)
Register_write(0x560,1FB8)
Register_write(0x562,1F6E)
Register_write(0x564,01AD)
Register_write(0x566,1FE5)
Register_write(0x568,1FAA)
Register_write(0x56A,1F05)
Register_write(0x56C,0251)
Register_write(0x56E,0002)
Register_write(0x570,0000)
Register_write(0x572,0000)
Register_write(0x574,0000)
Register_write(0x576,0000)
Register_write(0x578,0000)
Register_write(0x57A,0000)
Register_write(0x57C,0000)
Register_write(0x57E,0000)
Register_write(0x580,0000)
Register_write(0x582,0000)
Register_write(0x584,0011)
Register_write(0x586,0004)
Register_write(0x588,0005) //15:0] To LCD frame buffer A start addr.
Register_write(0x58A,0804) //22:16] To frame buffer A start addr.
Register_write(0x58C,0008) //15:0] To LCD frame buffer B start addr.
Register_write(0x58E,00f0) //22:16] To frame buffer B start addr.
Register_write(0x590,00b4) //Output width of port 1 : 240x2
Register_write(0x592,01e0) //Output height of port 1 : 180
Register_write(0x594,0000)
Register_write(0x596,0000)
Register_write(0x598,0000)
Register_write(0x59A,0000)
Register_write(0x59C,0000)
Register_write(0x59E,0000)
Register_write(0x5A0,0000)
Register_write(0x5A2,0000)
Register_write(0x5A4,0000)
Register_write(0x5A6,0000)
Register_write(0x5A8,0000)
Register_write(0x5AA,0000)
Register_write(0x5AC,0000)
Register_write(0x5AE,0000)
Register_write(0x5B0,0000)
Register_write(0x5B2,0000)
Register_write(0x5B4,0000)
Register_write(0x5B6,0000)
Register_write(0x5B8,0000)
Register_write(0x5C0,0000)
Register_write(0x5C2,0000)
Register_write(0x5C4,0000)
Register_write(0x5C6,0000)
Register_write(0x5C8,0000)
Register_write(0x5CA,0000)
Register_write(0x5CC,0000)
Register_write(0x5CE,0000)
Register_write(0x5D0,0000)
Register_write(0x5D2,0000)
Register_write(0x5D4,0000)
Register_write(0x5D6,0000)
Register_write(0x5DA,0000)
Register_write(0x5DE,0000)
Register_write(0x5E0,0000)
Register_write(0x5E2,0000)
Register_write(0x5E4,0000)
Register_write(0x5E6,0000)
Register_write(0x5E8,0000)
Register_write(0x5EA,0000)
Register_write(0x5E0,0000)
Register_write(0x5F0,0000)
Register_write(0x5F2,0000)
Register_write(0x5F4,0000)
Register_write(0x5F6,0000)
Register_write(0x5F8,0000)
Register_write(0x5FA,0000)
Register_write(0x5FC,0000)
Register_write(0x5FE,0000)
Register_write(0x600,0000)
Register_write(0x602,0000)
Register_write(0x604,0000)
Register_write(0x606,0000)
Register_write(0x608,0000)
Register_write(0x60A,0000)
Register_write(0x60C,0000)
Register_write(0x60E,0000)
Register_write(0x610,0000)
Register_write(0x612,0000)
Register_write(0x614,0000)
Register_write(0x616,0000)
Register_write(0x618,0000)
Register_write(0x61A,0000)
Register_write(0x61C,0000)
Register_write(0x61E,0000)
Register_write(0x620,0000)
Register_write(0x622,0000)
Register_write(0x624,0000)
Register_write(0x626,0000)
Register_write(0x628,0000)
Register_write(0x62A,0000)
Register_write(0x62C,0000)
Register_write(0x62E,0000)
Register_write(0x630,0000)
Register_write(0x632,0000)
Register_write(0x634,0000)
Register_write(0x636,0000)
Register_write(0x638,0000)
Register_write(0x63A,0000)
Register_write(0x63C,0000)
Register_write(0x63E,0000)
Register_write(0x640,0000)
Register_write(0x642,0000)
Register_write(0x644,0000)
Register_write(0x646,0000)
Register_write(0x648,0000)
Register_write(0x64A,0000)
Register_write(0x64C,0000)
Register_write(0x64E,0000)
Register_write(0x650,0000)
Register_write(0x652,0000)
Register_write(0x654,0000)
Register_write(0x656,0000)
Register_write(0x658,0000)
Register_write(0x65A,0000)
Register_write(0x65C,0000)
Register_write(0x65E,0000)
Register_write(0x660,0000)
Register_write(0x662,0000)
Register_write(0x664,0000)
Register_write(0x666,0000)
Register_write(0x668,0000)
Register_write(0x66A,0000)
Register_write(0x66C,0000)
Register_write(0x66E,0000)
Register_write(0x670,0000)
Register_write(0x672,0000)
Register_write(0x674,0000)
Register_write(0x676,0000)
Register_write(0x678,0000)
Register_write(0x67A,0000)
Register_write(0x67C,0000)
Register_write(0x67E,0000)
Register_write(0x680,0000)
Register_write(0x682,0000)
Register_write(0x684,0000)
Register_write(0x686,0000)
Register_write(0x688,0000)
Register_write(0x68A,0000)
Register_write(0x68C,0000)
Register_write(0x68E,0000)
Register_write(0x690,0000)
Register_write(0x692,0000)
Register_write(0x694,0000)
Register_write(0x696,0000)
Register_write(0x698,0000)
Register_write(0x69A,0000)
Register_write(0x69C,0000)
Register_write(0x69E,0000)
Register_write(0x6A0,0000)
Register_write(0x6A2,0000)
Register_write(0x6A4,0000)
Register_write(0x6A6,0000)
Register_write(0x6A8,0000)
Register_write(0x6AA,0000)
Register_write(0x6AC,0000)
Register_write(0x6AE,0000)
Register_write(0x6B0,0000)
Register_write(0x6B2,0000)
Register_write(0x6B4,0000)
Register_write(0x6B6,0000)
Register_write(0x6B8,0000)
Register_write(0x6BA,0000)
Register_write(0x6BB,0000)
Register_write(0x6BC,0000)
Register_write(0x6BD,0000)
Register_write(0x6BF,0000)
Register_write(0x6C0,0000)
Register_write(0x6C2,0000)
Register_write(0x6C4,0000)
Register_write(0x6C6,0000)
Register_write(0x6C8,0000)
Register_write(0x6CA,0000)
Register_write(0x6CE,0000)
Register_write(0x6D0,0000)
Register_write(0x6D2,0000)
Register_write(0x6D4,0000)
Register_write(0x6D6,0000)
Register_write(0x6D8,0000)
Register_write(0x6DA,0000)
Register_write(0x6DC,0000)
Register_write(0x6DE,0000)
Register_write(0x6E0,0000)
Register_write(0x6E2,0000)
Register_write(0x6E4,0000)
Register_write(0x6E6,0000)
Register_write(0x6E8,0000)
Register_write(0x6EA,0000)
Register_write(0x6EC,0000)
Register_write(0x6EE,0000)
Register_write(0x6F0,0000)
Register_write(0x6F2,0000)
Register_write(0x6F4,0000)
Register_write(0x6F6,0000)
Register_write(0x6F8,0000)
Register_write(0x6FA,0000)
Register_write(0x6FC,0000)
Register_write(0x6FE,0000)
Register_write(0x6B8,0000)
Register_write(0x6BA,0000)
Register_write(0x6BC,0000)
Register_write(0x6BE,0000)
Register_write(0x6C0,0000)
Register_write(0x6C2,0000)
Register_write(0x6C4,0000)
Register_write(0x6C6,0000)
Register_write(0x6C8,0000)
Register_write(0x6CA,0000)
Register_write(0x6CC,0000)
Register_write(0x6CE,0000)
Register_write(0x6D0,0000)
Register_write(0x6D2,0000)
Register_write(0x6D4,0000)
Register_write(0x6D6,0000)
Register_write(0x6D8,0000)
Register_write(0x6DA,0000)
Register_write(0x6DC,0000)
Register_write(0x6E0,0000)
Register_write(0x6E2,0000)
Register_write(0x6E4,0000)
Register_write(0x6E6,0000)
Register_write(0x6E8,0000)
Register_write(0x6EA,0000)
Register_write(0x6EC,0000)
Register_write(0x6EE,0000)
Register_write(0x6F0,0000)
Register_write(0x6F2,0000)
Register_write(0x6F4,0000)
Register_write(0x6F6,0000)
Register_write(0x6F8,0000)
Register_write(0x6FA,0000)
Register_write(0x6FC,0000)
Register_write(0x6FE,0000)

Register_write(0x500, 0x000A) //5:4]= 00:GR/BG, 01:RG/GB, 10:BG/GR, 11:GB/RG [1]= 1: Enable Capture
fire ISP mode
Register_write(0x4FA,0001)    //b0=1: Enable the ISP
Capture module.

9.4. Initiate Audio codec

There are 3 parts code for testing audio codec, first is initial audio codec by I2C bus that used DGPIO0/1, second is setting Glamo 3365 Audio Engine (openRISC) for MP3 decode, last is music file that is MP3 format. Use these file to play mp3 file to check audio codec circuit correct or not.

Following setting for initial Wolfson audio codec WM8978 by I2C bus. For others audio codec, customers have to prepare the setting by themselves.

SetAudioCodecRegister(0 , 0x000);  
SetAudioCodecRegister(1 , 0xDD);  
SetAudioCodecRegister(2 , 0xBF);  
SetAudioCodecRegister(3 , 0xEF);  
SetAudioCodecRegister(4 , 0x010);  

//R05(0)= 1 WM8978 test itself 
SetAudioCodecRegister(5 , 0x000);  
//R06=0x9 WM8978 master/ 0x0 WM8978 slave 
SetAudioCodecRegister(6 , 0x000);  
SetAudioCodecRegister(7 , 0x000);  
SetAudioCodecRegister(8 , 0x000);  
SetAudioCodecRegister(9 , 0x000);  
SetAudioCodecRegister(10 , 0x000);  
SetAudioCodecRegister(11 , 0x1FF);  
SetAudioCodecRegister(12 , 0x1FF);  
SetAudioCodecRegister(14 , 0x1FF);  
SetAudioCodecRegister(15 , 0x1FF);  
SetAudioCodecRegister(16 , 0x1FF);  
SetAudioCodecRegister(18 , 0x1C);  
SetAudioCodecRegister(19 , 0x02C);  
SetAudioCodecRegister(20 , 0x02C);
SetAudioCodecRegister(21, 0x02C);
SetAudioCodecRegister(22, 0x02C);
SetAudioCodecRegister(24, 0x032);
SetAudioCodecRegister(25, 0x000);
SetAudioCodecRegister(27, 0x000);
SetAudioCodecRegister(28, 0x000);
SetAudioCodecRegister(29, 0x000);
SetAudioCodecRegister(30, 0x000);
SetAudioCodecRegister(32, 0x192);
SetAudioCodecRegister(33, 0x10f);
SetAudioCodecRegister(34, 0x0aa);
SetAudioCodecRegister(35, 0x000);
SetAudioCodecRegister(36, 0x008);
SetAudioCodecRegister(37, 0x00C);
SetAudioCodecRegister(38, 0x093);
SetAudioCodecRegister(39, 0x0E9);
SetAudioCodecRegister(41, 0x000);
SetAudioCodecRegister(43, 0x011);
SetAudioCodecRegister(44, 0x003);
SetAudioCodecRegister(45, 0x1BF);
SetAudioCodecRegister(46, 0x010);
SetAudioCodecRegister(47, 0x000);
SetAudioCodecRegister(48, 0x000);
SetAudioCodecRegister(49, 0x002);
SetAudioCodecRegister(50, 0x021);
SetAudioCodecRegister(51, 0x021);
SetAudioCodecRegister(52, 0x1BF);
SetAudioCodecRegister(53, 0x1BF);
SetAudioCodecRegister(54, 0x1BF);
SetAudioCodecRegister(55, 0x1BF);
SetAudioCodecRegister(56, 0x001);
SetAudioCodecRegister(57, 0x001);

Following Glamo 3365 register write is for audio engine setting.

//start
Register_write(0x40, 0x050a);
wait(20)
Register_write(0x44, 0x050a);
wait(20)
Register_write(0x300, 0x0973);
Register_write(0x336, 0x0041);
Register_write(0x334, 0x0000);
wait(20)
Register_write(0x334, 0xe060);
Register_write(0x312, 0xe00b);
Register_write(0x314, 0xe0fb);
Register_write(0x302, 0x00ff);
Register_write(0x304, 0x0108);
Register_write(0x306, 0x0010);

//#ORSIC_Init
//#3365 slave
//#Register_write(0x1692, 0x0041);
//#3365 Master
Register_write(0x1692, 0x0000);
wait(20)

//#ZCLK_Init 44100hz
Register_write(0x44,0x0535);
wait(20)
Register_write(0x32,0x021f);
wait(20)
Register_write(0x3c,0xB);
wait(20)
Register_write(0x3e,0xB);
wait(20)
Register_write(0x16ae,0x1F);
wait(20)
Register_write(0x16b0,0x1F);
wait(20)
//end

Smedia provides mp3 sample that convert to special format that Glamo 3365 needs for testing, you don’t have to prepare it by yourself.
10. No-wait Memory Bus Read/Write Timing for Glamo 3365

Since some baseband doesn’t provide wait function pin for I/O device, Smedia provides this non-wait signal read/write table for reference. Default PLL (main clock) is 36MHz in Glamo 3365, when BCLK divide ratio is 1(default), then operation frequency is 36MHz, so the worst case is 270ns for signal read from 3D registers. It’s useful for setting baseband read/write waiting timing.

<table>
<thead>
<tr>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Write to Registers for all Registers</td>
<td>90 ns</td>
</tr>
<tr>
<td>Single Read from Registers except ISP, JPEG, MPEG and 3D Registers</td>
<td>90 ns</td>
</tr>
<tr>
<td>Single Read from ISP Registers</td>
<td>120 ns</td>
</tr>
<tr>
<td>Single Read from JPEG Registers</td>
<td>210 ns</td>
</tr>
<tr>
<td>Single Read from MPEG Registers</td>
<td>210 ns</td>
</tr>
<tr>
<td>Single Read from 3D Registers</td>
<td>240 ns</td>
</tr>
<tr>
<td>MicroP for AE/AWB Write to MMIO Registers Penalty</td>
<td>60 ns</td>
</tr>
<tr>
<td>MicroP for AE/AWB Read from MMIO Registers Penalty</td>
<td>150 ns</td>
</tr>
<tr>
<td>Open RISC Write to MMIO Registers Penalty</td>
<td>60 ns</td>
</tr>
<tr>
<td>Open RISC Read from MMIO Registers Penalty</td>
<td>120 ns</td>
</tr>
<tr>
<td>Consecutive Write to Registers Penalty</td>
<td>30 ns</td>
</tr>
<tr>
<td>Consecutive Read from Registers Penalty</td>
<td>30 ns</td>
</tr>
<tr>
<td>Write to Registers after Read Penalty</td>
<td>30 ns</td>
</tr>
<tr>
<td>Read from Registers after Write Penalty</td>
<td>90 ns</td>
</tr>
</tbody>
</table>
11. Quick Boot Concept

Since mobile phone needs to play music and shows boot image when loading OS for booting, Smedia provides a procedure for quick initiation of Glamo 3365, panel, and audio codec to display BMP images and play MP3 files. The procedure involves a write register sequence similar to chapter 9 but packaged all sequences in one package. Currently, support for BMP images and MP3 music only.

There are 11 files in this package as below:

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>main.c</td>
<td>Sample code for showing image and playing MP3.</td>
</tr>
<tr>
<td>365.h</td>
<td>Include file</td>
</tr>
<tr>
<td>init365.c</td>
<td>The procedure for initializing Glamo 3365.</td>
</tr>
<tr>
<td>InitPanel.c</td>
<td>Example code for turning on panel.</td>
</tr>
<tr>
<td>Audio_WM8978_IIC.c</td>
<td>Example code for initializing audio codec.</td>
</tr>
<tr>
<td>mp3boot.c</td>
<td>The procedure of MP3 decode by script.</td>
</tr>
<tr>
<td>mp3decoder.dat</td>
<td>Array of Glamo RISC program.</td>
</tr>
<tr>
<td>AngelTheme_32kbps.dat</td>
<td>Array of MP3 which the customer wants to play quickly.</td>
</tr>
<tr>
<td>S240x320P04.h</td>
<td>Array of image which the customer wants to display on the screen.</td>
</tr>
<tr>
<td>convert.exe</td>
<td>Convert 24bpp BMP to array. Image size depends on panel size.</td>
</tr>
<tr>
<td>convarray.exe</td>
<td>Convert MP3 to array. Due to array size issue, recommend 8k bit rate MP3 file.</td>
</tr>
</tbody>
</table>

Customers integrate main.c in the boot loader or any entry, and the entry should be after initiating the baseband, before OS boot. The real entry in the target system must be determined by the customer self.

Follow these steps to implement this function:

1. Prepare the BMP image you want and convert it to an array.
2. Prepare the MP3 music file and convert it to an array.
3. Modify Glamo 3365 base address in 365.h, the base address depends on the H/W circuit. Default “M365_Base” is 0x8000000 based on 3365’s evaluation board.
4. Modify initial panel code in InitPanel.c. Default “panel type” is Sony 342 based on 3365's evaluation board.
5. Modify initial audio codec code in Audio_WM8978_IIC.c. Default “Audio codec type” is Wolfson WM8978 based on 3365 evaluation board.
6. Due to audio codec I2C read/write speed depending on CPU speed, perhaps you need to modify “IIC_PULSE_DELAY” in Audio_WM8978_IIC.c for codec needs.
7. Follow the procedure in main.c to implement it in the entry.
8. Test and debug.
12. Troubleshooting

Q1: Why is the address pin connection between Samsung 2443 and 3365 series different from Samsung 2440 series?

Ans: According to Samsung 2443 application note, S3C2443 memory interface is different with S3C2440 series memory controller. Regardless of data bus width, A0 of memory pad must be connected to A0 of S3C2443, please refer to the information below, please check this with your Samsung™ CPU provider about detail if you have more question.

6.1.1 Differences with others (S3C2440, S3C2413)

<table>
<thead>
<tr>
<th>Function</th>
<th>S3C2440</th>
<th>S3C2413</th>
<th>S3C2443</th>
</tr>
</thead>
<tbody>
<tr>
<td>mSDR</td>
<td>Normal SDR</td>
<td>support</td>
<td>Support</td>
</tr>
<tr>
<td>mDDR</td>
<td>x</td>
<td>support</td>
<td>Support</td>
</tr>
<tr>
<td>Addr. connection</td>
<td>8bit : A0 → A0</td>
<td>8bit : A0 → A0</td>
<td>8bit : A0 → A0</td>
</tr>
<tr>
<td></td>
<td>16bit : A1 → A0</td>
<td>16bit : A0 → A0</td>
<td>16bit : A0 → A0</td>
</tr>
<tr>
<td>32bit : A2 → A0</td>
<td>32bit : A0 → A0</td>
<td>32bit : A0 → A0</td>
<td></td>
</tr>
<tr>
<td>EMRS command</td>
<td>Not support</td>
<td>support</td>
<td>support</td>
</tr>
<tr>
<td>Data bus signal</td>
<td>Shared mSDR/ROM</td>
<td>Shared mSDR/ROM</td>
<td>Isolated mSDR and ROM</td>
</tr>
</tbody>
</table>

Figure 12-1 Memory interface difference from Samsung 2440
(Source from Samsung™ Electronics SMDK2443_AppDev_Note_SDRAM_070125_rev1)

Smedia processor and Samsung 2440 Series

---

Smedia processor is word addressable, not byte addressable, so you can find read/write port of all related registers are even.
Appendix:

Part A: LCD Module Reference Schematics

LCM: SONY / ACX506AKU

SONY-ACX506AKU

Pin location

For LCD I/O Power

Glamo 3360 LCD IF
LCM: CASIO / CMG12MS8

For LCD power

CASIO CMG12MS8

For LCD power

CASIO CMG12MS8
Part B: Image Sensor Reference schematics

Toshiba TCM8210MD

MM3360 Camera Interface

Toshiba / TCM8210MD

---

Multimedia in Every Mobile

66 of 83
Sanyo IGT99268F

Sanyo / IGT99268F

MM3360 Camera Interface

- C11: PCLK
- C10: SCLK
- A12: CSCL
- C9: CSCL
- A11: CSDA
- A8: CSSPO1
- C6: CSSPO0
- C8: MCHSYNC
- C9: CVSYNC
- A9: DVDD
- E7: CD0
- D8: CD1
- B8: CD2
- A8: CD3
- D7: CD4
- C7: CD5
- B6: CD6
- D6: CD7
- B5: FLCT

Pin connections include:
- CON1: DVDD 1
  - VDD1
  - VDD2
  - VDD3
  - CS
  - REGRES
  - HREF
  - VREF
  - MCK
  - D7
  - D6
  - D5
  - D4
  - D3
  - D2
  - D1
  - D0

- R1: 4.7K
- R2: 4.7K

Capacitors:
- C3: 0.1μF
- C2: 0.1μF
- C1: 0.1μF

Ground connections:
- GND
- GND
- GND
- GND
- GND
- GND
Part C: TV Encoder (TV Out) Interface Reference Schematics

CH7023/CH7024

Diagram of TV Encoder (TV Out) Interface Reference Schematics for CH7023/CH7024.
Notice about the LCD & TV Dual Display solution of 3365

Because 3365 have only one LCD controller, if you need the LCD & TV Dual display solution, please notice following recommendations:

- Please find the LCM with 4:3 (Landscape mode) resolution ratio (ex: 320x240, because TV is 4:3 ratio mode)
- Please confirm that the I/O spec. and timing of the LCM must be the same as the spec. of TV encoder
- Please enhance the driving ability of all connected pins of 3365 LCD Interface

When using landscape LCM, please use RGB Interface as control interface. (Recommendations)
If customers want to use portrait LCM, please use CPU Interface as control interface.
Video Mode in portrait LCM (video)

Note: The portrait LCM must have memory buffer to keep video frame, besides, the solution has performance suffer, because raster engine of 3365 must update LCM and TV by turns.

Customers can implement the choosing items for end-users to choose. EX:

1. TV display only (no performance suffer)
2. LCM display only (no performance suffer)
3. TV + LCM display. (have performance suffer)
Part D: 3365 Lead-free reflow profile suggestion for SMT

Notice: all information is for reference, the real situation must be based on the SMT reflow environment.

1.) Follow: IPC/JEDEC J-STD-020 C

2.) Condition:
   Average ramp-up rate (217°C to peak): 1~2°C/sec max.
   Preheat: 150~200°C, 60~180 seconds
   Temperature maintained above 217°C: 60~150 seconds
   Time within 5°C of actual peak temperature: 20 ~ 40 sec.
   Peak temperature: 260+0/-5°C
   Ramp-down rate: 3°C/sec. max.
   Time 25°C to peak temperature: 8 minutes max.
   Cycle interval: 5 minutes

Figure: Surface Mount Technology (SMT) Reflow Profiles