LOCOSTO Program

IMAGING Application Note

Rev:1.0

Document Number:	APN212
Version:	1.0
Status:	Released
Creation Date:	2005-Nov-07
Last changed:	2006-Jan-06
File Name:	APN212.doc

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

HISTORY

Version	Date	Author	Notes
Ver: 0.1	07-Nov-2005	CL, KA	1
Ver 1.0	06-Jan-2006	CL, KA	2
			3
			4

NOTES:

1. Creation
2. First release
3. ……..
4.

REFERENCE DOCUMENTS

[1] ADCM-2700 Technical Reference Manuel – Agilent
[2] HD66772 LCD source driver – Hitachi
[3] locosto_registers_13_01_02_02663 Locosto Register specification –TI
[4] Locosto BUM EMIF chapter –TI
[5] Locosto BUM DMA chapter – TI
[6] Locosto BUM LCD interface BUM chapter – TI
[7] Locosto BUM CLK_RST_PW chapter – TI
[8] Locosto BUM CAMERA INTERFACE chapter – TI
[9] Locosto Sanity check APN206 – TI
[10] Locosto Debug features APN211 – TI
[11] LPH8754-I Data Manuel – Phillips
[12] Locosto_ballout_13_01_02_02664 – TI

GLOSSARY

APLL	**A**nalog **P**hase **L**ock **L**oop
DBB	**D**igital **B**ase **B**and
BPP	**B**yte **P**er **P**ixel
CCP	**C**ompact **C**amera **C**ore
DPLL	**D**igital **P**hase **L**ock **L**oop
DSP	**D**igital **S**ignal **P**rocessor
EMIF	**E**xternal **M**emory **I**nterface
FSM	**F**inite **S**tate **M**achine
FPS	**F**rame **P**er **S**econd
IF	**I**nter**F**ace
I2C	**I**nter-**I**ntegrated **C**ircuit Bus. A two-wire, synchronous, serial interface designed primarily for communication between intelligent IC devices
JPEG	**J**oint **P**hotographic **E**xperts **G**roup (standard method of image compression)
LCD	**L**iquid **C**rystal **D**isplay
MCU	**M**icro **C**ontroller **U**nit
OCP	**O**pen **C**ore **P**rotocol
QCIF	**Q**uarter **C**ommon **I**ntermediate **F**ormat
TCS	**T**I **C**hipset **S**oftware
VGA	**V**ideo **G**raphics **A**rray (computer display standard)
YUV	YUV model defines a color space in terms of one luminance and two chrominance components. **Y** stands for the luminance component, and **U** and **V** are the chrominance components
WS	**W**ait **S**tate

1 Introduction

The purpose of this document is to provide an example for camera application with Locosto.

This document describes a stand-alone[1] camera "view-Finder" QCIF image and "snap-shot" VGA image system from data capture to image display using the on-chip Camera I/F, DMA and LCD I/F. This implementation can be advantageously used for software benchmarking (system control, image processing, data flow performances…). No processing (rotation, border frame, zoom, etc) is to be done on the image to be displayed

The camera system implementation example is based on the following hardware:

- Locosto Debug/Emulator board (named Calli-conso: Used for hardware system validation purpose) with JTAG connector
- Locosto chip.
- The Agilent ADCM-2700 sensor
- The Phillips LPH8754-I LCD Controller & Panel
- The Spansion S7INS128JC0: Multi chip product (S29NS-J Flash & Cellular RAM type PSRAM)

[1] This document does not address the integration of Camera sub-system within a GSM/GPRS system … OS constraint is not considered &/ evaluated as well…

2 Camera "View-Finder" system overview

The Figure 1 shows the video-data path from the camera sensor connected to the Camera I/F up to the external LCD Controller attached to the dedicated Locosto LCD I/F.

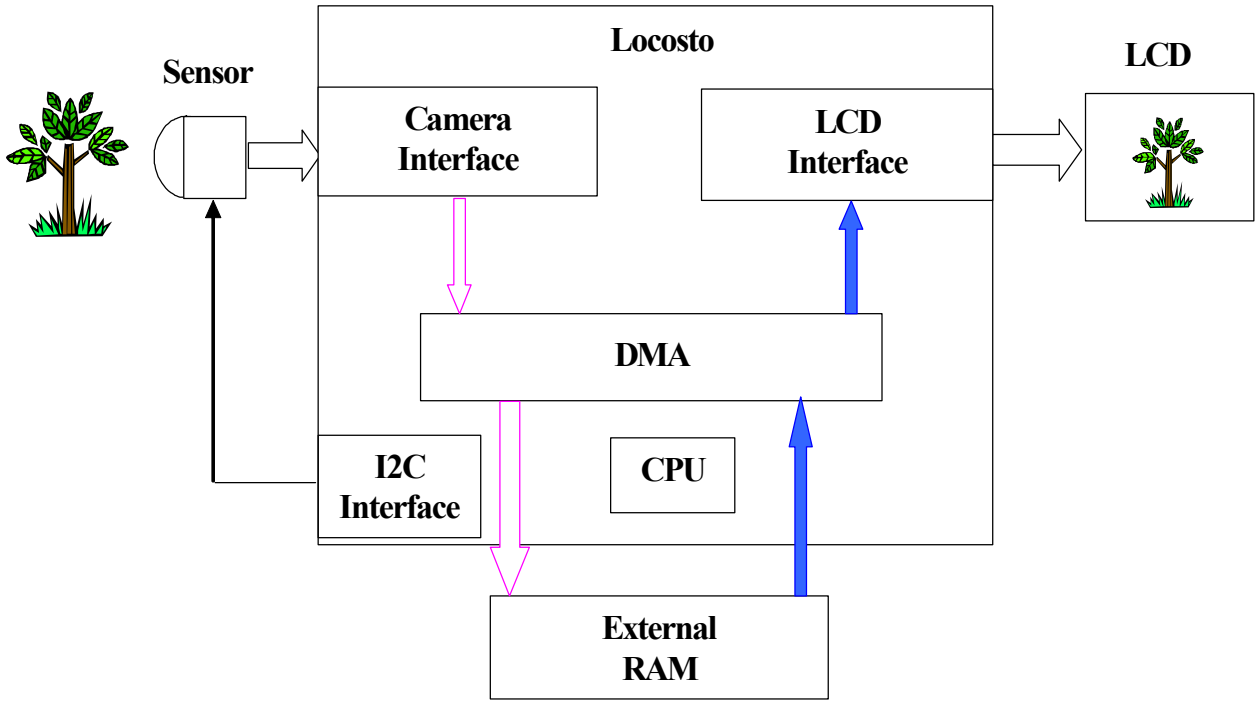

Figure 1 Camera Data Path Overview view finder

- The Camera sensor is configured via the I2C serial link to get a color pre-processed (e.g. gamma correction…) QCIF image RGB 5.6.5 format in video mode and VGA YUV:4.2.2 in still mode
- The data captured from the sensor is transferred (32 bits) from FIFO of the camera interface to an external RAM frame-buffer through the DMA.
- The frame Buffer is sized at least to receive the entire VGA input data (~80 K-bytes)
- CPU processes is not needed for this use case as the sensor output 5.6.5 RGB format and the external LCD is configured to receive 5.6.5 RGB format.
- Then, the DMA takes care for feeding from the frame-buffer to the FIFO of the LCD interface which finally loads the external LCD controller & panel.

2.1 Hardware system diagram

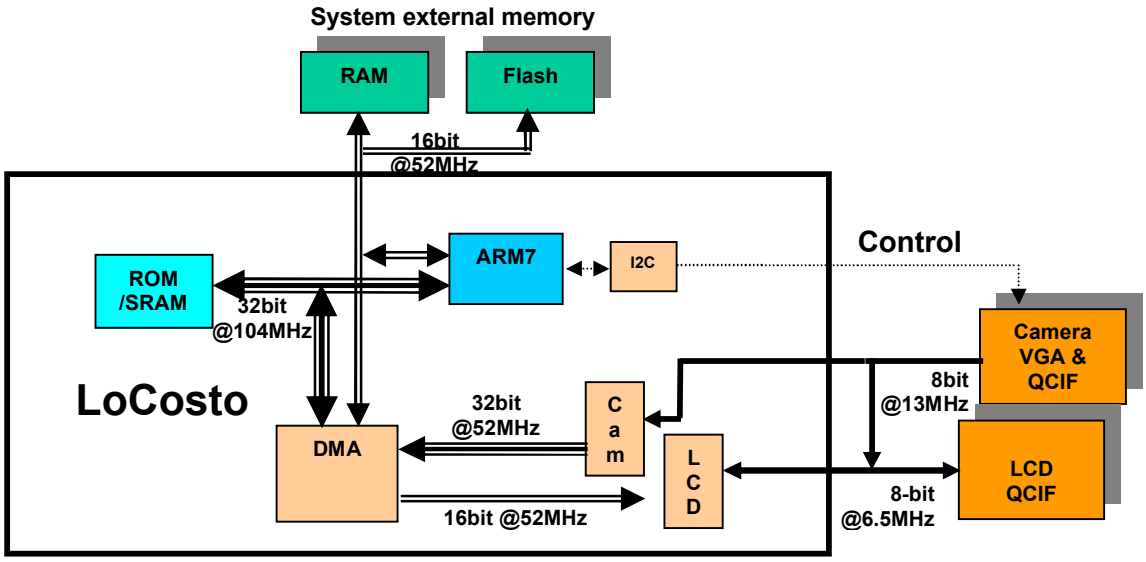

Figure 2 Hardware system diagram

The Table 1 summarizes the bus, port and interface maximum bandwidth.
Note the in the use case LCD clock is set to 6.5 MHz and the pixel clock is 13 MHz instead the 48 MHz (APLL clock).

Another thing to keep in mind is that, in real conditions with operating system, layer1, modem activity … not all the bandwidth of each bus or interface is available.

Bus, port, interface		Data width	Frequency	Theoretical peak rate
Camera external port		8 bit	48 MHz	384 Mbps
		1 bit	208 MHz	208 Mbps
External Memory I/F (EMIF)	burst	16 bit	52 MHz	832 Mbps
	asynchronous		11.5 MHz	184 Mbps
I^2C external port		1 bit	0.4 MHz	0.4 Mbps
LCD external port		8 bit	13 MHz	104 Mbps
			3.25 MHz	26 Mbps
Camera, EMIF and LCD internal bus I/F		32 bit	52 MHz	1664 Mbps
I^2C internal bus interface		16 bit	52 MHz	832 Mbps

Table 1 Buses, ports and interfaces bandwidths summary

2.2 Camera core Interface

The camera core module interfaces externally with an image sensor, bufferises the pixel data in a FIFO and generates DMA requests. Due to IO pin multiplexing Locosto is only able to interface Camera-Sensor's 8-bit video output data port CAM_D_[7:0].

- **Interfaces features**
  - Serial interface: compact camera port (CCP) interface. Maximum clock of 208 MHz.
  - Parallel interface: 8, 10 or 12 bits. Maximum clock of 96 MHz for 8-bit data and 48 MHz for 10 or 12-bit data. Supports two operating modes:
    - BT 656: ITU-R BT656 compatible parallel interface
    - NOBT 656: general parallel interface with vertical and horizontal synchronization signals.**(Used for the current application note)**

Caution:

On Locosto Platform there is a limitation to 8 bits for the bus width and the maximum clock rate is 48 Mhz.

- **Interrupt**

The interrupt line is asserted (active low) when one the following events, takes place:

- FIFO_UF – FIFO underflow
- FIFO_OF – FIFO overflow
- FIFO_THR – FIFO threshold
- FIFO_FULL – FIFO full
- FIFO_NOEMPTY – FIFO not empty

The following interrupts are not used in NOBT mode.

- SSC_ERR – Shifted Synchronization Code
- FSC_ERR – False Synchronization Code
- FW_ERR – Frame Width Error
- FS – Frame Start
- LE – Line End
- LS – Line Start
- FE – Frame End

Only the FIFO underflow and Overflow are enabled on the use case. There is no reason to set others Interrupt sources for better performances on imaging application.

Caution:

If an overflow or underflow event occurs you must reset all dataflow paths in order to restart the system in clean way (for example the DMA controller).

2.3 LCD Interface

The Locosto LCD Interface is an 8 bits bi-directional parallel port that can be configured for interfacing 6800 or 8086 operating mode LCD. It includes a 128 x 16-bit FIFO that can be accessed either by the CPU or the DMA in 1 cycle up to 52MHz, via a 16-bit data bus.

- **Interfaces**
    - Parallel interface 8 bits with transfer rate up to 13 M-bytes per second. Supports two operating modes:
        - 6800
        - 8086 **(Used for the current application note)**
- **Interrupt**

The interrupt line is asserted (active low) when one of the following events, takes place:

- FIFO EMPTY
- READ EVENT

There is no reason to set Interrupts management for the Imaging application (viewfinder, snap-shoot …).

2.4 Camera I/F to Sensor connections

2.4.1 Sensor module constraints

The role of the camera module is to:

- Deliver preview quality frames during “viewfinder” sub-sampled resolution (QCIF for instance)
- Deliver final quality frame during “snapshot” full resolution (VGA for instance)

Here are the requirements for a camera module used with LOCOSTO:

1. Picture size: The camera module must be able to deliver pictures in the range QCIF / 176x144 up to VGA / 640x480.

2. Camera module must implement the ISP (3A algorithm, etc)

3. Frame rate: The targeted frame rate in viewfinder mode is 15 fps.

4. Data format: The output formats that need to be supported natively by the camera module are the following:
   a. RGB565 16bpp
   b. YUV 4:2:2 16bpp

5. VGA camera module must be connectable to LOCOSTO parallel camera core I/F, or to LOCOSTO serial CCP camera core I/F,

6. Tristate capability: In the case where a parallel data interface camera is used, because of LOCOSTO I/O balls assignment, camera signals will be multiplexed with other functions signals (e.g. IrDA). This obliges the camera module to be able to put its data lines into High-Z state.

7. Control interface: the control commands will be send through I^2C in the case of a directly connected parallel camera. This interface can run at 100 KHz or 400 KHz (fast mode).

8. I/O voltage: 1.8 V. **This means the camera module component has to support 1.8V I/O for connection. A solution to support a traditional 2.8V I/O camera module is to use external level shifters on the board.**

2.4.2 Agilent ADCM features

The Agilent ADCM-2700 VGA CMOS camera module contains a single integrated circuit with an image sensor and an image processor.
Powered at 2.8 Volt and clocked from the On-chip oscillator or from an external master source 8 up to 24 MHz.

Type: CMOS sensor,
A/D: 8 bits per pixel.
Interface: Serial port, Parallel port,
Resolution: CIF, VGA, QCIF, QVGA, QQCIF, QQVGA,
Decimation: full-frame, sub-sampled, windowing,
Frame rate: 15fps max at VGA resolution,
Output formats: YUV, RGB,
Functions: Horizontal mirroring, Vertical mirroring,
RGB configuration: RGB888, RGB666, RGB565, RGB444, RGB332,
YUV configuration: YUV4:4:4, YUV4:2:2,
Algorithms: AEC, AWB.
Power supply: 2.8 V (Level shifter required on the use case).

The sensor is configured through a serial port (2 wires) interface which is compliant with the Slave I2C protocol. Therefore the sensor's serial port is connected to the Locosto I2C port.

2.4.3 Hardware connection

For reduced ball count, many I/O signals of LOCOSTO are multiplexed on the same balls. Depending on LOCOSTO configuration, one signal or another is present on one given ball at a time.
For the imaging domain, the signals of interest are the camera I/F signals, as well as the LCD interface signals. The Table 2 details the multiplexing for those signals:

Pin	Ball	Mode 0	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5
2	M6		Dcd_txir	Cam_d_1			
142	E9	Lcd_data_0	Cam_d_0		Ndf0		
143	B8	Lcd_data_1	Cam_d_1		Ndf1		
144	C8	Lcd_data_2	Cam_d_2		Ndf2		
145	E8	Lcd_data_3	Cam_d_3		Ndf3		
147	B7	Lcd_data_4	Cam_d_4		Ndf4		
148	D8	Lcd_data_5	Cam_d_5		Ndf5		
150	C7	Lcd_data_6	Cam_d_6		Ndf6		
151	B6	Lcd_data_7	Cam_d_7		Ndf7		
153	F8		Cam_hs	Cam_d_3			
154	E7		Cam_vs	Cam_d_3			
155	A5		Cam_lclk				
156	C6		Cam_xclk	Cam_d_3			
163	C4		Spi_ncs2	Ndf7	Cam_d_7		
165	G7		Bu	Ndf6	Cam_d_6		
166	B3		Lt3	Ndf5	Cam_d_5		
168	E5	Nd_nwp	Ndf4	Cam_d_4			
178	F5		Ncs2	Ndf3			
179	D2		Ncs1	Ndf2			
180	H7		Add_23	Ndf1			
181	E3	Ncs0		Lt1	Tspact_7	Ndf0	
182	G6		nfwp	Tspact_7	Lt1	Ndf2	Cam_d_2
183	G5		Add_22	Ndf0			
229	N5		Dsr_rxir	Cam_d_0			

Other	GPIO / test	Camera	Nand Flash	LCD

Table 2 Locosto Multiplexing pin

As signals are multiplexed together, one cannot avoid losing some functionalities while others are in use.

The first information is that it is not possible to have camera, LCD and Nand Flash at the same time. Either LCD or Nand will share the same bus, or camera and Nand.

Also, even if the camera bus can be isolated from the LCD bus completely, cam_d_3 has then to be multiplexed with another camera control signal.

The possible configurations are:

1. Calypso+ like configuration. LCD multiplexed with Nand. Camera on a separate bus. Cam_d_3 multiplexed with either cam_vs or cam_xclk.

2. Denser configuration. LCD multiplexed camera. Nand either multiplexed with both LCD and camera, or on a separate bus.

2.4.3.1 Configuration 1 details

Configuration number 1 is the preferred configuration.
For the remaining parts of this document, the use case will focus on this configuration 1.

For this configuration, either the vertical synchronization signal cam_vs or the external clock output cam_xclk cannot be used any more because of the multiplexing.

If cam_vs is not used (As done on the use case), the software will need to synchronize on DMA end of transfer interrupt rather than vertical synchronization

If cam_xclk is not used (As done on the use case), the camera module will need to take its reference clock from another source (e.g. Locosto 13 MHz clock output, ckout_13M) or from it internal PLL.

2.4.3.1.1 Configuration 1 without VSYNC signal

The Table 3 summarizes the inter-connections between the Agilent ADCM-2700 sensor and the Locosto device without the usage of sensor VSYNC signal.

ADCM-2700 CMOS sensor	LOCOSTO
SDA	I2c_scl
SCL	I2c_sda
D0	gpio_47/cam_d_0
D1	Gpio_0/cam_d_1
D2	gpio_7/cam_d_2
D3	gpio_20/cam_d_3
D4	Nd_nwp/cam_d_4
D5	gpio_30/cam_d_5
D6	gpio_29/cam_d_6
D7	gpio_28/cam_d_7
VCLK	gpio_21/cam_lclk
HSYNC	gpio_19/cam_hs
VSYNC	Not connected
MCLK	gpio_22/cam_xclk or **ckout_13M**
RESET (not existing for this sensor)	Use any GPIO

Table 3 ADCM-2700 CMOS sensor to Locosto connection without VSYNC signal

- **I2C connections are tied to the power-supply through pull-up resistor (I2C requirement).**
- **In case of real need of Vertical synchronization interruption a GPIO in interrupt mode can be used and connected though sensor VSYNC pin .**

See CAMERA_IOPinMuxing4() function in annex for IO CONFIGURATION registers value.

Note:

In case of real need of vertical Interrupt is needed synchronization GPIO can be set as interrupt trigger.

2.4.3.1.2 Configuration 1 with VSYNC signal

ADCM-2700 CMOS sensor	LOCOSTO
SDA	I2c_scl
SCL	I2c_sda
D0	gpio_47/cam_d_0
D1	Gpio_0/cam_d_1
D2	gpio_7/cam_d_2
D3	gpio_22/cam_d_3
D4	Nd_nwp/cam_d_4
D5	gpio_30/cam_d_5
D6	gpio_29/cam_d_6
D7	gpio_28/cam_d_7
VCLK	gpio_21/cam_lclk
HSYNC	gpio_19/cam_hs
VSYNC	Gpio_20/cam_vs
MCLK	Ckout_13Mhz
RESET (not existing for this sensor)	Use any GPIO

Table 4 ADCM-2700 CMOS sensor to Locosto connection with VSYNC signal

See CAMERA_IOPinMuxing3() function in annex for IO CONFIGURATION registers value.

2.4.3.2 Configuration 2 details

Configuration 2 finds its roots in the buses bandwidths estimations.

During viewfinder configuration, bandwidth resources for LCD and camera buses are not fully used. This is also the case during snapshot mode, for camera and Nand flash buses, which leads to a variation of configuration 2, where even the Nand flash signals are multiplexed with the camera and LCD signals.

This configuration leads to a more aggressive multiplexing than configuration 1 and will not be more discussed in the present document.

2.5 LCD I/F to LCD Controller connections

2.5.1 LCD module constraints

The LCD is the main user interface of a GSM phone. Its role, in the context of camera connectivity, is to display…

- Preview images, when in “viewfinder” mode
- Decimated version of the “snapshot”
- Other images (e.g. images from the WAP)

The requirements for the LCD are:

1. LCD size: a typical LCD for the camera application will be QCIF / QCIF+ (i.e. 220x170 pixels).

2. Data format: the LCD needs to accept data in RGB565 (two bytes) format and/or RGB888 (three bytes) format.

3. LCD Interface: Locosto comprises an 8 bit large LCD interface that can run at a fraction of 13 MHz in {1, 2, 4, and 8}.

4. **I/O voltage: due to Locosto C027 process, Locosto LCD interface’s I/O support 1.8V. The LCD component has to support 1.8V I/O for proper connection. Other wise level shifter must be used.**

2.5.2 Phillips LPH8754-I features

The Phillips LPH8754-I is a 5.7cm (2.2”) MD_TFD Liquid Crystal Display module with built-in oscillating circuit and 6 x 540 x 288 (933120 bits) RAM capacity. The controller is an HD66772 from Hitachi.

Type: TFT liquid Crystal,
Mode: 8086-series standard
Interface: Parallel port **8**/9/16 or 18 bits,
Resolution: 176 (W) x 220 (H).
Input formats: RGB
Functions: Moving picture, area selecting/scrolling & column automatic incrementing …
RGB configuration: RGB666, RGB565, RGB444,
Power supply: 2.8 V (Level shifter required on the use case).

2.5.3 Hardware connection

The Table 5 summarizes the inter-connections between the LPH8754-I LCD and the Locosto device

LPH8754-I LCD	LOCOSTO
CS	GPIO_13/LCD_NCS0
Reset	LCD_RS
WR	LCD_RNW
RD	LCD_ESTRB
MD8	LCD_D_7
MD7	LCD_D_6
MD6	LCD_D_5
MD5	LCD_D_4
MD4	LCD_D_3
MD3	LCD_D_2
MD2	LCD_D_1
MD1	LCD_D_0

Table 5 LPH8754-I LCD to Locosto connection

Note:

On Table 2 LCD_DATA IO pins are multiplexed with Nand Flash IO pins. In order to switch in one shot from LCD to Nand Flash and vice versa use LCD/Camera/Nand flash configuration register bit 1:0.

E.g. Switch directly all LCD_DATA_0 to 7 into mode 2 (NAND_D_[7:0]) by setting “10” into bit 1:0 of LCD/Camera/ Nand flash configuration register (CONF_LCD_CAM_ND).

3 Software Description

The software hereafter described gives an example of Locosto peripheral programming as well as data flow control to get image sequences from the Agilent ADCM-2700 CMOS sensor (QCIF RGB 5.6.5) and display them on the Phillips LPH8754-I LCD (176 x 220, RGB 5:6:5 format).

The basic demonstration program is structured in two parts:

- System Initialization & Configuration (clock system & memory interface, I2C Interface, Agilent ADCM-2700 CMOS, CCP Interface, DMA sub-system, LCD Interface & Phillips LPH8754-I LCD)
- Data Flow control as shown figure 3.

3.1 Image processing Flow

The Diagram below highlights the phases of the video processing flow…The flow chart (*see figure 4*) details the CPU system control; later-on, *the Section 3.4.1* provides values regarding the data capture and data display execution times.

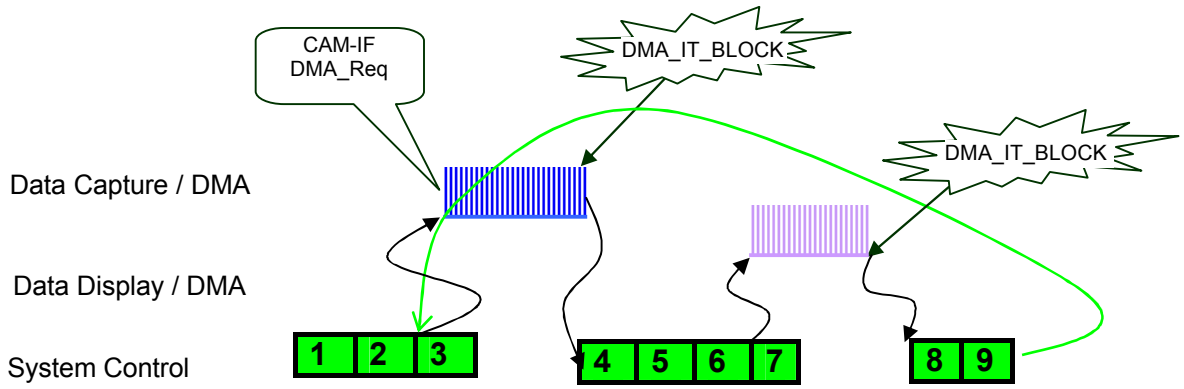

Figure 3 Image frame sequence with parallel acquisition and display

- DMA Sub-system is in charge for moving data-in (CAM I/F -> Frame-Buffer) and data-out (Frame-Buffer-> LCD I/F); 2 DMA channels are configured to support these data transfers.
- The CPU handles the system control flow from external flash.
- **The CPU activity is considered null during the transfer view finder flow.**

3.1.1 System Control Flow chart

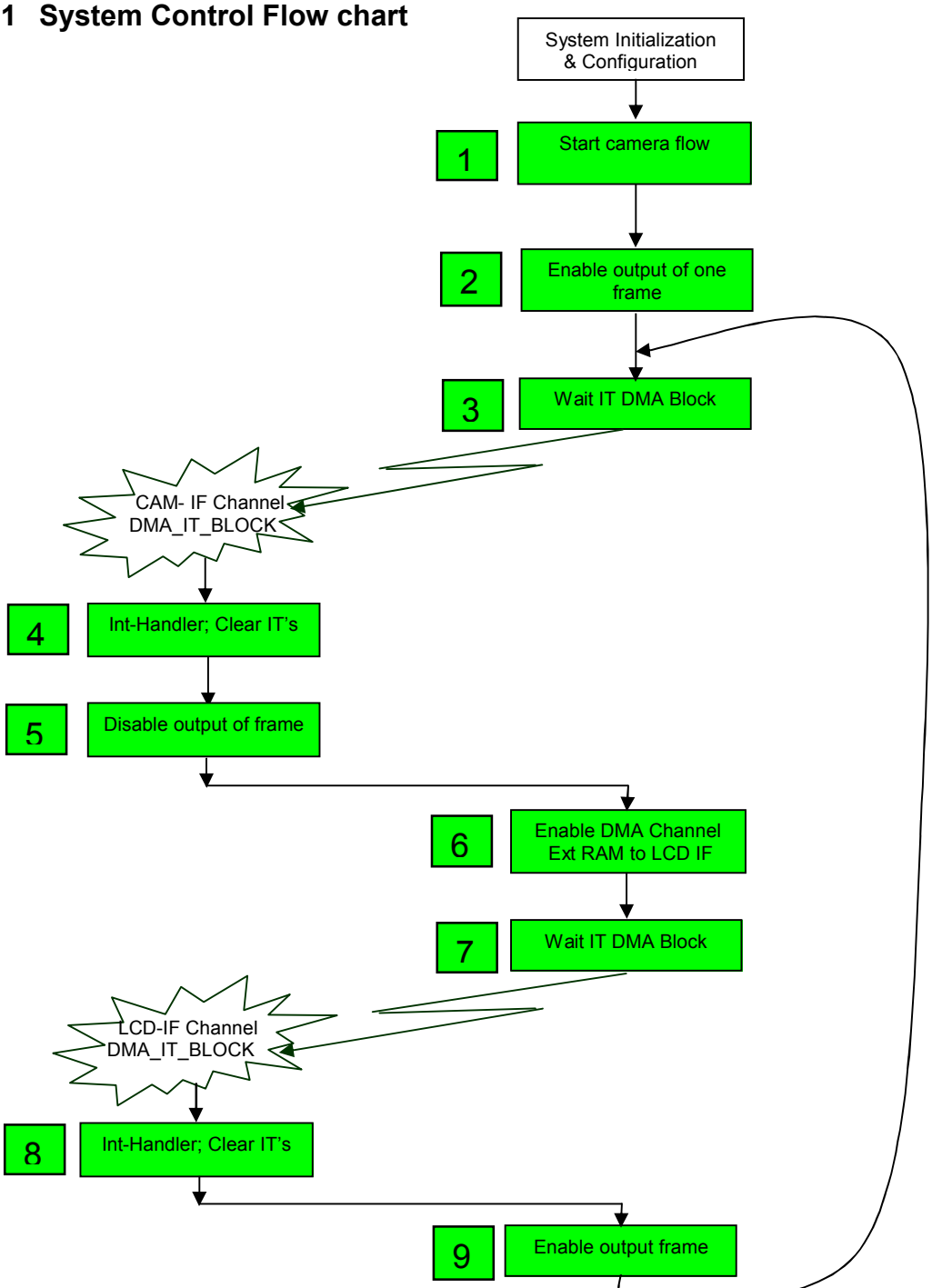

Figure 4 Frame sequence flow chart with parallel acquisition and display

- **System Initialization and configuration** This step consist in the initialization by reset and configuration of following modules:

  - Camera Interface
  - LCD Interface
  - DMA channels
  - Sensor & LCD display

- **Step 1** Start camera flow:

  - Reset FIFO pointers and internal camera core state machines, by writing "1" to the CC_RST bit 18 of control register (CC_CTRL) of camera core module.

  - Reset by setting RST_CHANNEL1 bit 1 of soft reset register (DMA_SRR) to "1"
  - Enable by setting EN bit 7 of channel 1 control register (DMA_CCR) to "1". The DMA channel is ready for moving received data to the external buffer. DMA channel is in auto-Init mode.

  - Enable DMA hardware request when the FIFO of camera core interface reach the threshold setting (if not already done on initialization of Camera Interface or if has been disable before) by setting DMA_EN bit 8 of control DMA register (CC_CTRL_DMA).

  - Enable the video mode in the sensor by writing 0x0001 to control register of the sensor.

  - Set the CC_EN bit 16 of control register to "1", to enable the data flow from the image sensor (CC_CTRL).

- **Step 2** Enable output of one frame. Set bit 0 of CMD_2 register of the sensor to "1".
  Setting this bit has an effect if bit 12 (FR_OUT Frame output control) of OUTPUT_CTRL register is set to "0"

- **Step 3** Waiting of DMA end-of-block interrupt. The end of the image acquisition is signaled by the DMA end-of-block interrupt.

- **Step 4** (ISR) Processes the Interrupt Handler; mainly clears IT status…

- **Step 5** (ISR) Disable output of one frame. Set bit 0 of CMD_2 register of the sensor to "0".
  When the output is re-enabled (Step 9) the sensor continue with a new frame generation; this insuring the system frame synchronization.

- **Step 6** Enable the DMA channel ready for moving data from the external buffer to LCD display.
  (Before enabling the DMA channel for LCD display set the LCD pointer address to 0)

- **Step 7** Waiting of DMA end-of-block interrupt. The end of the image transfer to the LCD is signaled by the DMA end-of-block interrupt.

- **Step 8** (ISR) processes the Interrupt Handler; mainly clears IT status… and restart step 1 for a new image capture…..

- **Step 9** Enable output of one frame. Set bit 0 of CMD_2 register of the sensor to "1".
  Since the output frame has been stopped at the end of the previous frame (step 5), re-enabling output insures a proper Frame synchronization (i.e. the first pixel of the new picture is the first data received and captured within the camera I/F FIFO).
  (The DMA transfers the data from camera interface to external RAM until the complete QCIF image is captured).

For the section code see CAM_start_viewfinder() function in ANNEX.

Note:

- **The "Vertical Sync" (Frame End) signal from the camera-sensor is not monitored for this application; instead the end of the image-data transfer is detected; the image capture completion is indicated by a DMA's IT_BLOCK when DMA take care of the transfer.**
- The step 5 and 9 can be bypassed in case of use Vertical synchronization.
- In the application ckout_13M signal clock is used as input clock of the sensor but APLL_CLK can also be used though CAM_XCLK pin to increase pixel clock (up to 48 Mhz). The configuration of the clock divider is programmable by setting the external clock register of camera interface module (CC_CTRL_XCLK).
  By default the APLL clock is disabled. To Turn on APLL clock set bit 0 of APLL control register (CNTL_APLL_DIV_CLK)

Caution:

Never stop the ckout_13M or CAM_XCLK (Clock supply for the sensor) in video mode. The sensor needs it for processing image.

3.1.2 Memory foot print

For RGB565 16bpp sensors, it is expected working with one single buffer (frame buffer), called the video plane.

	RGB565 format	Value (in KB)
1	QCIF buffer containing RGB565 data, 2 bytes/pixel	49.5

Table 6 QCIF image memory footprint

3.1.3 "View-Finder" plot measurement

The following figure shows a plot during a view-finder mode. It illustrates the different steps described in the flow chart.

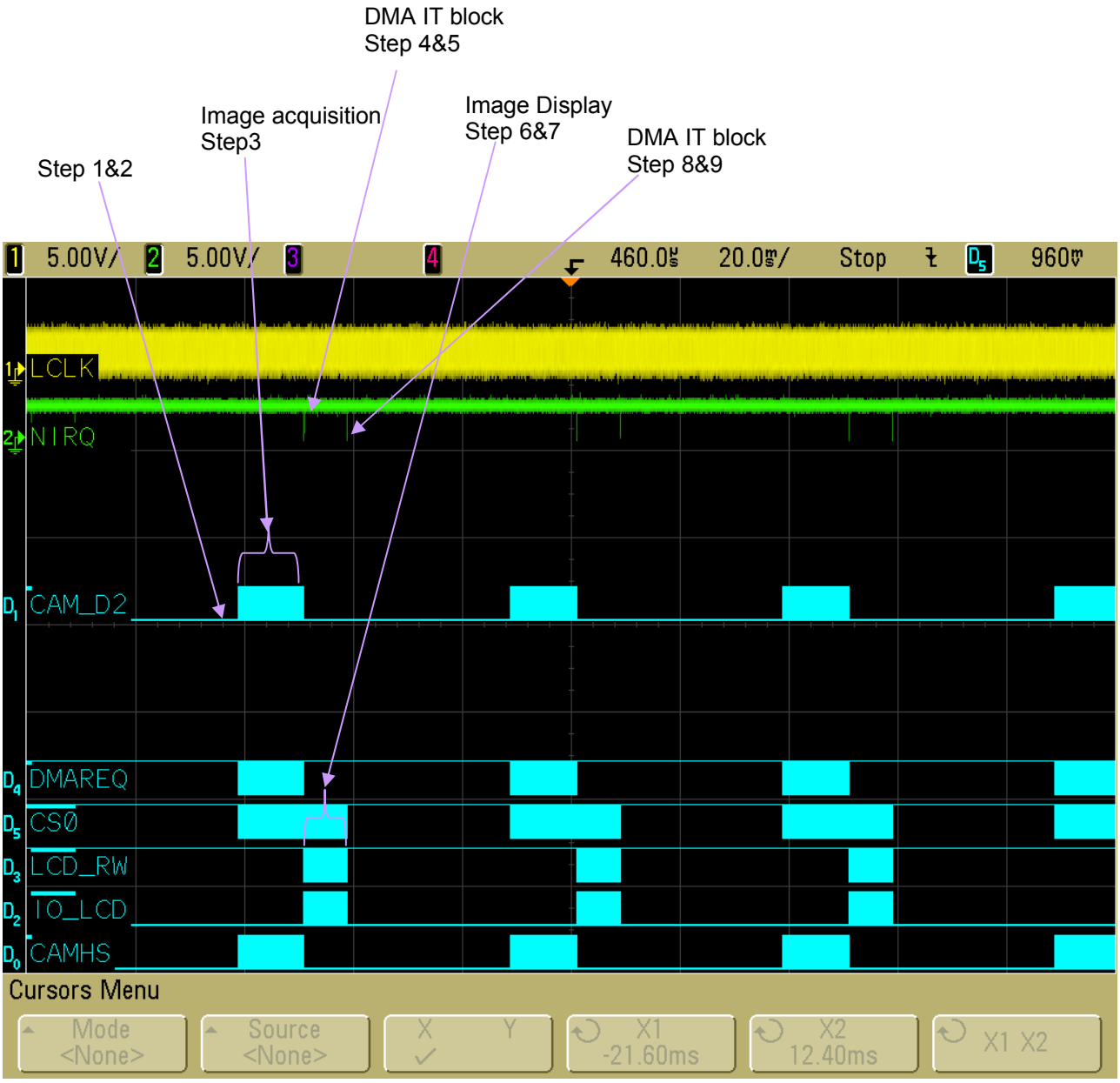

Figure 5 Frame sequence with parallel acquisition and display plot

3.2 Camera "View-Finder" System Configuration

This section describes the on chip as well as external components configuration.

3.2.1 Generic platform setting & control

The system configuration consists of:

- ARM system clock is set @ 104Mhz,
- External memory CS3 (Flash) is accessed in burst mode with Prefetch buffer enabled.
- External memory CS0 (External RAM where is located the frame buffer) is accessed in burst mode at 52 MHz.
- Interrupt Handler setting (IT DMA and Camera Interface are enabled).

3.2.2 I2C Configuration & Control (ADCM sensor control)

Refer to document Reference **Error! Reference source not found.** for more detail.

Three registers are programmed after the module has been reset (SOFTWARE_RESET /I2C System configuration register bit 1). Others registers can keep the reset value.

- SCL Low Time Register: (SCL value=0x00)
  - SCLL (bit 15:0).This register determines the serial clock time low pulse duration value when master; 400 kb/s.

Register value set: 0x0009;

- SCL High Time Register: (SCL value=0x00)
  - SCLH (bit 15:0).This register determines the serial clock time high pulse duration value when master; 400 kb/s.

Register value set: 0x0009;

- Configuration Register: (I2C_CON= 00xx xxxx 0000 xx00)
  - STT (bit 0). Remains to its reset value. No action or start condition generated; "0"
  - STP (bit 1). Remains to its reset value. No action or stop condition generated; "0"
  - XA (bit 8). Remains 7-bit address mode; "0"
  - TRX (bit 9). Remains receive mode; "0"
  - MST (bit 10). Mater mode; "0"
  - STB (bit 11). Remains normal mode; "0"

- BE (bit 14). Remains Little endian mode; “0”
- I2C_EN (bit 15). I2c module enabled; “1”

Register value set: 0x8000

Note:

The Agilent Sensor I2C slave address is “0x53”.

3.2.3 Camera Core Interface Configuration & Control

Refer to document Reference [8] for more detail.

The Camera interface is used in conjunction with a DMA to capture a QCIF image from the sensor. The input data rate is 13 M-bytes/s clocked from the Sensor pixel clock in the application.

Camera Interface input clock

Camera gets either 48 MHz clock or 52 MHz from clock management module for source clock.

- Set “CAMERA_DIV_FACTOR” bit 3:2 of camera control clock register (CNTL_CAMERA_DIV_CLK) to “10“defines camera source clock as 52 Mhz DPLL clock.
- Set “CAMERA_ENABLE” bit 0 of CNTL_CAMERA_DIV_CLK register to “1“ enables the camera interface module

Camera control clock register is part of clock management module.

Note:

Set “CAMERA_DIV_FACTOR” bit 3:2 of camera control clock register to “01“defines camera source clock as APLL clock.
The camera source clock can be modified on the fly when camera module is enabled.

Four registers are programmed after the module has been reset. Others registers must keep the reset value.

The reset is done by:

1. Set “SOFTRESET” bit 1 of system configuration register (CC_SYSCONFIG)to “1”
2. Read “RESETDONE” bit 0 of system status register (CC_SYSSTATUS) to check if it is “1”, meaning the reset took place.

If after 5 reads RESETDONE still returns 0, it can be assumed that an error occurred during the reset stage.

- Control Register: (CC_CTRL; Reset value=0x00001001)
  - CCP_MODE (bit 0). CCP mode is disabled; “1“
  - PAR_MODE (bit 3:1). Set parallel NO-BT 8 bit mode; “000“
  - NOBT_VS_POL (bit 8). Set CAM_P_VS active high; “0“
  - NOBT_HS_POL (bit 9). Set CAM_P_HS active high; “0“
  - PAR_CLK_POL (bit 10). Data sampled on rising edge; “0”
  - PAR_ORDER_CAM (bit 11). Swapped is enabled; “1”
  - BT_CORRECT (bit 12). Not used on NO-BT mode, Remains to its reset value; “0”
  - NoBT_CONTROL (bit 13). Acquisition starts when Vertical synchronization is high, Remains to its reset value; “0”
  - CC_EN (bit 16). Enables the sensor interface of the camera core module; “1”
    - By writing "1" to this field, the module is enabled.

- By writing "0" to this field, the module is disabled at the end of the frame if CC_FRAM_TRIG =1 and is disabled immediately if CC_FRAM_TRIG = 0

- CC_FRAME_TRIG (bit 17). Not used, Remains to its reset value; “0”

- CC_RST (bit 18). Remains to its reset value; “0”
- CC_IF_SYNCHRO (bit 19). Remains to its reset value; “0”

Register value set: 0x00010800

Note:

- **In order to reduce the power consumption of the module, the user must select the auto-idle capability by setting ”AUTOIDLE” bit 0 of system configuration register(CC_SYSCONFIG) to “1”.**

- **The sensor is configured in RGB 5.6.5 mode. Each 24 bit RGB pixel has the lower two bits of green truncated. Red and Blue have the lower three bits truncated. The pixel is packed into two bits as following with bytes 1 in MSB and Byte 2 in LSB.**

Byte #	Bit 7	Bit 6	Bit 5	Bit 5	Bit 3	Bit 2	Bit 1	Bit 0
1	R7	R6	R5	R4	R3	G7	G6	G5
2	G4	G3	G2	B7	B6	B5	B4	B3

Table 7 Pixel package

PAR_ORDER_CAM bit is set to 1 in order to swap LSB and MSB. On the frame buffer data are correctly packed in RGB 5.6.5 according to LCD format.

- There are additional pixel times between rows that represent a blanking period. The active pixels are identified by a combination of two additional timing signals: horizontal synchronization (CAM_P_HS) and vertical synchronization (CAM_P_VS). During the image sensor readout, these signals define when a row of valid data begins and ends and when a frame starts and ends. A bit field sets the CAM_P_HS polarity (NOBT_HS_POL) and CAM_P_VS polarity (NOBT_VS_POL). **In case of not using CAM_P_VS data are valid only according polarity setting.**

➢ Control DMA Register: (CC_CTRL_DMA;Reset value=0x00000087)

- FIFO_TRESHOLD (bit 6:0). Sets the FIFO threshold in 32 bits. For QCIF image, pixels row number is 176. In order to have the synchronization with the sensor FIFO threshold is set to 176/2. So FIFO threshold value is set to 87. “0x57”

- DMA_EN (bit 8). Enable DMA request; “1“

Register value set: 0x00000157

The FIFO_TRESHOLD is set to rows pixel number in 32 bits -1 when the image size enables it.

E.g. In the case of QCIF image when an entire row (176 pixels 16 bits) is written to the FIFO a DMA request is sent. The transfers of the data on the external memory are done during the horizontal blanking period.
There is one DMA request per line. This reduces DMA load.

- Generic Parameters Register: (CC_GENPAR;Reset value=0x00000006)
    - CC_FIFO_DEPTH (bit 6:0). Sets the FIFO size to $2^{CC\_FIFO\_DEPTH}$ 32-bit words =128. "7"

  Register value set: 0x00000007

- Interrupt Enable Register: (CC_IRQEN;Reset value=0x00000000)
    - Only FIFO underflow and Overflow are enabled on the current use case.

Register value set: 0x00000003

The interrupt is cleared by writing "1" to the CC_IRQSTATUS register bit corresponding to the interrupt occurred.

Note:

In case of use of CAM_XCLK clock (APLL clock), the setting of the clock divisor value for CAM_XCLK generation based on APLL clock (48 Mhz) is done though External Clock Control register.

3.2.4 LCD Interface Configuration & Control

Refer to document Reference [6] for more detail.

A DMA channel (*see 3.2.5.2 DMA LCD channel*) is also used to load the FIFO of the LCD Interface. LCD I/F supplies the Phillips LCD with a QCIF image 50688-bytes block (176 x 144 x 2) for each image. The output data rate is set to 6.5 M-bytes/s.

The set-up of the LCD interface is achieved via one register: control register (CNTL_REG). Others registers remain to their reset value.

- Interface Control Register: (CNTL_REG; Reset value=0x3801)
    - SOFT_NRST (bit 0). Software Reset. This bit is activated (set to 0) before setting the configuration. It returns high after LCD I/F has been reset; "1"
    - CLOCK13_EN (bit 1). Module clock is enabled; "1"
    - TX_CLOCK_DIV (bit 3:2). 13MHz/2 is selected; "10"
    - RX_CLOCK_DIV (bit 5:4). Remains to its reset value (Not used); "00"
    - FIFO_EMPTY_IT_EN (bit 6). Interrupt remains disabled; "0"
    - LCD_READ_EVENT_IT_EN (bit 7). Interrupt remains disabled; 0x0
    - DMA_EN (bit 8). DMA capability is enabled; "1"
    - MODE (bit 9). 8086-series standard is set; "1"
    - FLIP_BYTES (bit 10). MSB is read/write first; "0"
    - SUSPEND_EN (bit 11). Emulation use only; "1"

- MIN_FRAME_SIZE (bit 13:12).Minimum number of transmitted words (16 bits) between two DMA request set to 64 words 16 bits; “10”
- N_DUMMY (bit 15 & 14). Set the 2 dummy read cycle (Not used); “10”

Register value set: 0xAB0B

Note:

In this example, the CLOCK13_EN bit of control register is set during the system configuration step for as long as the system runs. Chip consumption may be reduced by switching this bit in the system control loop, each time the LCD I/F is not activated. (Step 1, 2,3,4,5 & 9 on *figure 4*)

3.2.5 DMA Sub-System Configuration & Control

Refer to document Reference [5] for more detail.

Two DMA Sub-system channels are configured to support data transfer from Camera I/F to frame-buffer and frame-buffer to LCD I/F.

DMA Global registers setting apply to the six available DMA channels. Global Control, Soft-Reset & Arbiter registers only are programmed, others keep the reset value.

- Global Control Register:(DMA_GCR; Reset value = xxxx xxxx xxxx 01xx)
  - FREE (bit 2). On going transfers are suspended according to the CPU suspend signal (emulation use only); “0”
  - AUTOGATING_ON (bit3). Allows the DMA to cut off its clocks according to its activity (**This setting allows to save power consumption**); “1”

Register value set: 0x0008

- Soft Reset Register:(DMA_SSR; Reset value = xxxx xxxx xxx0 0000)
  - Soft_rst_ch_(5:0) (bit 5:0). Bit 4 & 5 are set to 1; returns to 0 after reset;

Register value set: 0x0003

- Arbiter Register:(DMA_AR; Reset value = xxxx xxxx 1001 1100)
  - IMIF_PRIO (bit 2:0). Access priority on IMIF between ARM and DMA. This field defines number of consecutive cycles that are allocated to the CPU before DMA steals one cycle. DMA bandwidth granted 20%; “0x4”
  - RHEA_PRIO (bit 3). Access granted to DMA when ARM and DMA perform simultaneous access on RHEA bus; “0x0”

- API_PRIO (bit 3). Access granted to DMA when ARM and DMA perform simultaneous access on API memory; “0x0”

- IPERIPH_PRIO (bit 4). Access priority on I-PERIPH between ARM and DMA. This field defines number of consecutive cycles that are allocated to the CPU before DMA steals one cycle. DMA bandwidth granted 100%; “0x0”

Register value set: 0x0004

Note:

The code is running in flash and frame buffer is located in the external memory. DMA and CPU access priority is managed by the EMIF. *Refer to document Reference [4] for more detail.*

The EMIF priority register allows the EMIF to give consecutive access to a host in Least Recently Used (LRU) priority arbitration. MPU could have from 1 to 8 consecutive accesses whereas DMA could have 1 to 15 consecutive accesses (**1 cycle allowed to the ARM and 1 to the DMA in the application)**

3.2.5.1 DMA Camera Interface Channel

The DMA channel 1 is dedicated to the Camera I/F. It is configured to transfer 32bit data from the Camera I/F page-buffer (IPERIPH bus) to the external RAM (frame buffer).

Each Channel has its own set of “Regular channel” registers.

- Channel Source Destination Parameters Register :(DMA_CSDP; Reset value= 0x0000)

  - DATA_TYPE (bit 1 & 0). 32 bits; “10”
  - SRC (bit 5:2). Transfer source I-PERIPH ; “0011”
  - SRC_PACK (bit 6). Remains to its reset value; Source packing not used; “0”
  - SRC_BURST_EN (bit 8 & 7). Burt access enable; “10”
  - DST (bit 12:9). Transfer destination EMIF (External RAM); “0100”
  - DST_PACK (bit 13). Remains to its reset value; Source packing not used; “0”
  - DST_BURST_EN (bit 15 & 14). Burst access enable; “10”

Register value set: 0x890E

- Channel Control Register :(DMA_CCR; Reset value= 0000 x0x0 00x0 0000)

  - SYNC (bit 4:0). The DMA transfer start is synchronized to the hardware request number “0x15 –> 10101”
  - PRIO (bit 6). Remains to its reset value, low priority level; “0”
  - EN (bit 7). This bit enables the channel and starts the transfer according to the channel configuration. This bit is not activated during the configuration; it is set during the system control flow (Step 1). “0”
  - AUTO_INIT (bit 8). The channel is enabled at the end of the transfer, auto-init ON; “1”
  - FIFO_FLUSH (bit 10). Enable FIFO flush; “1”
  - SRC_AMODE (bit 13 & 12). Constant address; “00”
  - DST_AMODE (bit 15 &14). Post incremented address; “01”

Register value set: 0x4515

- Channel Interrupt Control Register:(DMA_CICR;Reset value= xxxx xxxx x00x 0x11)
  - TOUT_IE (bit 0). Timeout interrupt enable; 0x1
  - DROP_IE (bit 1). Synchronization event drop interrupt enable; 0x1
  - FRAME_IE (bit 3). Frame interrupt remains disabled; 0x0
  - BLOCK_IE (bit 5). Block interrupt enable; 0x1
  - HALF_BLOCK_IE (bit 6). Half-Block interrupt remains disabled; 0x0

Register value set: 0x0023

- Channel Source start address, **lowers bits** register :(DMA_CSSA_L; Reset value undefined)
  - SOURCE_START_ADDRESS (bit 15:0). "Camera Interface FIFO DATA address:0x0970004C; "0x004C"
- Channel Source start address, **uppers bits** Register:(DMA_CSSA_U; Reset value undefined)
  - SOURCE_START_ADDRESS (bit 15:0). ""Camera Interface FIFO DATA address:0x0970004C; "0x0970"
- Channel Destination start address, **lowers bits** Register:(DMA_CDSA_L; Reset value undefined)
  - DESTINATION_START_ADDRESS (bit 15:0). "Frame Buffer address 0x00400000". 0x0000
- Channel Destination start address, **uppers bits** Register:(DMA_CDSA_U; Reset value undefined)
  - DESTINAION_START_ADDRESS (bit 15:0). "Frame Buffer address 0x00400000". 0x0040
- Channel Element Number Register :(DMA_CEN; Reset value undefined)
  - CHANNEL ELEMENT NUMBER (bit 15:0). 88
- Channel Frame Number Register :(DMA_CFN; Reset value undefined)
  - CHANNEL FRAME NUMBER (bit 15:0). 144

Note:

- **The DMA element number must be equal to threshold level fixed on Control DMA Register bit (6:0).**
- The DMA element & frame numbers are set to up-load a total block (QCIF RGB 5.6.5 image) of 50688 bytes (176 x 144 x 2). The frame number is then deducted from 50688 / 88 word 32 bits = 144 words 32 bits.

3.2.5.2 DMA LCD Interface Channel

The DMA channel 0 is dedicated to the LCD Interface. It is configured to transfer 16 bits data from the External RAM (frame buffer) to the transmit FIFO (RHEA bus) of the LCD interface.

- Channel Source Destination Parameters Register :(DMA_CSDP; Reset value= 0x0000)
  - DATA_TYPE (bit 0 & 1). 16 bits; “01”
  - SRC (bit 5:2). Transfer source EMIF (Internal RAM); “0100”
  - SRC_PACK (bit 6). DMA Packing; “1”
  - SRC_BURST_EN (bit 8 & 7). Burst enable; “10”
  - DST (bit 12:9). Transfer destination RHEA (TX FIF0 LCD Interface); “0001”
  - DST_PACK (bit 13). Remains to its reset value; Source packing not used; “0”
  - DST_BURST_EN (bit 15 & 14). Remains to its reset value, no burst mode; “00”

Register value set: 0x0351

- Channel Control Register :(DMA_CCR; Reset value= 0000 x0x0 00x0 0000)
  - SYNC (bit 4:0). The DMA transfer start is synchronized to the hardware request number 0x03; “00011”
  - PRIO (bit 6). Remains to its reset value, low priority level; “0”
  - EN (bit 7). Remains to its reset value during configuration phase; “0” This bit is set during the system control flow (Step 6).
  - AUTO_INIT (bit 8). The channel is disabled at the end of the transfer, auto-init off; “0”
  - FIFO FLUSH (bit 10). Enable FIFO flush; “1”
  - SRC_AMODE (bit 13 & 12). Post incremented address; “01”
  - DST_AMODE (bit 15 &14). Remains to its reset value. Constant address;”00”

Register value set: 0x1443

- Channel Interrupt Control Register :(DMA_CICR; Reset value= xxxx xxxx x00x 0x11)
  - TOUT_IE (bit 0). Timeout interrupt enable; “1”
  - DROP_IE (bit 1). Synchronization event drop interrupt enable; “1”
  - FRAME_IE (bit 3). Frame interrupt remains disabled; “0”
  - BLOCK_IE (bit 5). Block interrupt Enable; “1”
  - HALF_BLOCK_IE (bit 6). Half-Block interrupt remains disable; “0”

Register value set: 0x00023

- Channel Source start address, **lowers bits** register:(DMA_CSSA_L; Reset value undefined)
  - SOURCE_START_ADDRESS (bit 15:0). “Frame buffer address 0x00400000”. 0x0000

- Channel Source start address, **uppers bits** register:(DMA_CSSA_U; Reset value undefined)
    - SOURCE_START_ADDRESS (bit 15:0). “Frame buffer address 0x00400000”. 0x0040
- Channel Destination start address, **lowers bits** register:(DMA_CDSA_L; Reset value undefined)
    - DESTINATION_START_ADDRESS (bit 15:0). “LCD FIFO address 0xFFFFA006”. 0xA006
- Channel Destination start address, **uppers bits** Register:(DMA_CDSA_U; Reset value undefined)
    - DESTINAION_START_ADDRESS (bit 15:0). “LCD FIFO address 0xFFFFA006”. 0xFFFF
- Channel Element Number Register :(DMA_CEN; Reset value undefined)
    - CHANNEL ELEMENT NUMBER (bit 15:0). 88
- Channel Frame Number Register :(DMA_CFN; Reset value undefined)
    - CHANNEL FRAME NUMBER (bit 15:0). 288

3.2.6 Setting the ADCM-270 Sensor

Refer to document Reference [1] for more detail.

In the basic application proposed, only few registers of the ADCM-270 sensor are modified from their reset values.
After the ADCM-270 sensor has been reset (software reset), the following configuration and control is set:

Simple control registers

Output Format Register: (OUTPUT_FORMAT; Reset value = 0x0000)
Configure the sensor to output RGB 5.6.5 in viewfinder mode and YUV: 4.2.2 in still mode.

OUTPUT_FORMAT = 0x0803.

Image Size and Orientation Register: (SIZE; Reset value = 0x0000)
Configure QCIF format in viewfinder mode and VGA in still mode.
Allow sub sampling mode in video mode.
Disable right and down flipping in video and still mode.
Disallow sub sampling mode in still mode.

OUTPUT_FORMAT = 0x0681.

Frame rate Register: (FRAME_RATE; Reset value = 0x96)
Frame rate in 1/10 per second-default is 150 = 15 frames per seconds. Set to 50 frames per second.

FRAME_RATE =0x190;

Set bit 2 (CONFIG bit) of control register to "1" in order to update the simple control register.

Expert Hardware registers

Initial clock divider Register: (I_CLK_DIV; Reset value = 0x0001)
Divider value for MCLK prescaler set to "1". (MCLK is the clock provided by Locosto to the sensor, 13 Mhz in the application)

I_CLK_DIV = 0x0000

Clock Divisor Video mode Register: (CLK_DIV_V; Reset value = 0x0000)
Sensor and Image pipeline clock divider set to 0.

CLK_DIV_V =0x0000;

Clock Divisor Still mode Register: (CLK_DIV_S; Reset value = 0x0000)
Sensor and Image pipeline clock divider set to 0.

CLK_DIV_S =0x0000;

Parallel Control output Video mode Register: (PARALLEL_CTRL_V; Reset value = 0x0003)
Divider value for VCLK (Image Pipeline clock per byte) prescaler set to "00000000" in video mode. (VCLK is the clock pixel provided by the sensor to Locosto, 13 MHz in the application)

PARALLEL_CTRL_V =0x0000;

Parallel Control output Still mode Register: (PARALLEL_CTRL_S; Reset value = 0x0003)
Divider value for VCLK (Image Pipeline clock per byte) prescaler set to "00000000" in still mode. (VCLK is the clock pixel provided by the sensor to Locosto, 13 MHz in the application)

PARALLEL_CTRL_S =0x0000;

Note:

- If APLL_DIV_CLK for MCLK clock is used instead ckout_13M clock the value of Input Clock frequency register (CLK_FREQ) set by default at 13 MHz must be changed.
  E.g. If MCLK is 12 MHz→ CLK_FREQ = 0x2EE0.

- If instead QCIF image in view-finder mode is chosen Image Size and Orientation Register must be changed to value 0x3637 (Bit 2:0 set to "111") and to determine window size is done through Sensor width Video (SENSOR_WID_V) & Height (SENSOR_HGT_V) registers and Output width Video (OUTPUT_WID_V) & Height (OUTPUT_HGT_V) register

E.g. for 176 x 224 image size:
SENSOR_WID_V = 2 x 176
SENSOR_HGT_V = 2 x 224
OUTPUT_WID_V = 176
SENSOR_HGT_V = 224

If the image size is updated, three parameters must be changed:
- **DMA block size value for LCD channel (If Display size is also updated)**
- **DMA block size value for Camera channel**
- **FIFO threshold level value in Camera Interface.**

(*See Camera interface and DMA camera channel setting 3.2.3 & 3.2.5.1*)

- **For saving power consumption all clocks of sensor module can be cut when idle mode through CLK_GATE_DIS.**

To start View-finder write 0x0001 to control register of the sensor.
In order to control the output frame of the sensor update Output Control Register as described here-below

Output Control Register: (OUTPUT_CTRL; Reset value = 0x8013)
Configure the sensor to enable the output frame through CMD_2 bit 0 register. Set bit 12 (FR_OUT) to "0"
This is done after write 0x0001 to control register.

OUTPUT_CTRL = 0x8013.

3.2.7 Setting the Epson L2D22002TB301 LCD

Refer to document Reference [2] for more detail.

The LCD is configured through LCD interface; the following instruction commands are set after:

- Reset of the LCD
- Start of oscillation
- Power on sequence
- Turn Display On sequence

Frame Cycle Control Register: (FRAME_FREQ)
Set the division ratio of clock for internal operation to 4 (DIV 1-0 set to “10”)

FRAME_FREQ = 0x0025;

Display Control 1 Register: (DISP_CTRL1)
Set Source Output in display mode and HD67772 & Gate driver control signals in operate mode (D1:0 bits set to “11”.
Select the inversion of the display of all characters and graphics (REV bit set to “1”)
The Vgon will be Vgoff (GON bit set to “1”). Operation mode (DTE bit set to “1”)

DISP_CTRL1 = 0x0037;

Display Control 2 Register: (DISP_CTRL2)
Set the number of raster period in the back porch to 3 (BP3:0 set to “11”).
Set the number of raster period in the front porch to 3 (FP3:0 set to “11”).

DISP_CTRL2 = 0x0503;

Entry mode Register: (ENTRY_MODE)
The address counter is automatically incremented by 1 after data is written to the GRAM (I/D 1:0 bits set to “11”).
Data can be written the GRAM at high speed (HVM bit set to “1”).
BGR bit set to “1” IN CASE OF 18 bits interface.

ENTRY_MODE = 0x1230;

Drive Output Control Register: (DRV_OUT_CTRL)
Set the number of raster-row to be driven at 528 x 224 dots. (NL4:0 set to “11011”)

DRV_OUT_CTRL = 0x011B;

LTPS Interface Control Register: (LTPS_IF_CTRL)
Set the low pulse wide of CL1 signal to 8 internal clocks. (CLW2:0 set to “111”)

LTPS_IF_CTRL = 0x0700;

3.3 Camera "Snap-Shot" capture System flow

3.3.1 Sensor/DMA/Camera Interface Configuration

The purpose of this chapter is to explain how the snap shot capture image is done in the current application.

Concerning the sensor all the settings for VGA picture in still mode have been already programmed (*See* 3.2.6 Setting the ADCM-270 Sensor*)*
Upon view finder application when any keyboard is pressed an interrupt occurs.
Upon IT keyboard following step are done:

- Start still mode on camera. Write 0x0002 to control register.
- Disable DMA channel allocated to the camera interface by setting EN bit 7 of channel 1 control register (DMA_CCR) to "0".
- Change DMA Block length for Camera channel for VGA YUV:4.2.2 image

  Channel Element Number Register :(DMA_CEN)

  - CHANNEL ELEMENT NUMBER (bit 15:0). 80

  Channel Frame Number Register :(DMA_CFN)

  - CHANNEL FRAME NUMBER (bit 15:0). 1920.
- Change FIFO threshold level for VGA YUV:4.2.2 image at middle level of FIFO capacity

  Control DMA register: FIFO_TRESHOLD (bit 6:0). 80
- Enable DMA channel allocated to the camera interface by setting EN bit 7 of channel 1 control register (DMA_CCR) to "1".
- Waiting Block DMA interrupt

Note:

- As mentioned on *3.2.5.1DMA Camera Interface Channel setting* the element number shall be equal to FIFO threshold fixed on control DMA register of camera Interface.
- The DMA element & frame numbers are set to up-load a total block (VGA YUV 4.2.2 image) of 614400 bytes (649 x 480 x 2). The frame number is then deducted from 614400 / 80 word 32 bits = 1920 words 32 bits.

The FIFO threshold is set to 80 word 32 bits while an entire row is equal to 640 pixels 16 bits.
There are 4 DMA requests per line.

Caution:

Before parameters update on DMA it is necessary to stop the DMA.

Never stop the sensor between video mode and still mode. This increases by 5 the shutter time.

3.3.2 Memory footprint

For YUV 422 16bpp sensors, it is expected working with one single buffer.

	YUV422 format	Value (in KB)
1	VGA buffer containing YUV422 data, 2 bytes/pixel	600

Table 8 VGA Image memory footprint

3.3.3 Snap shot plot measurement

The following figure shows a plot during a snap-shot mode. It illustrates the change of mode from View-finder to Snap-shot mode

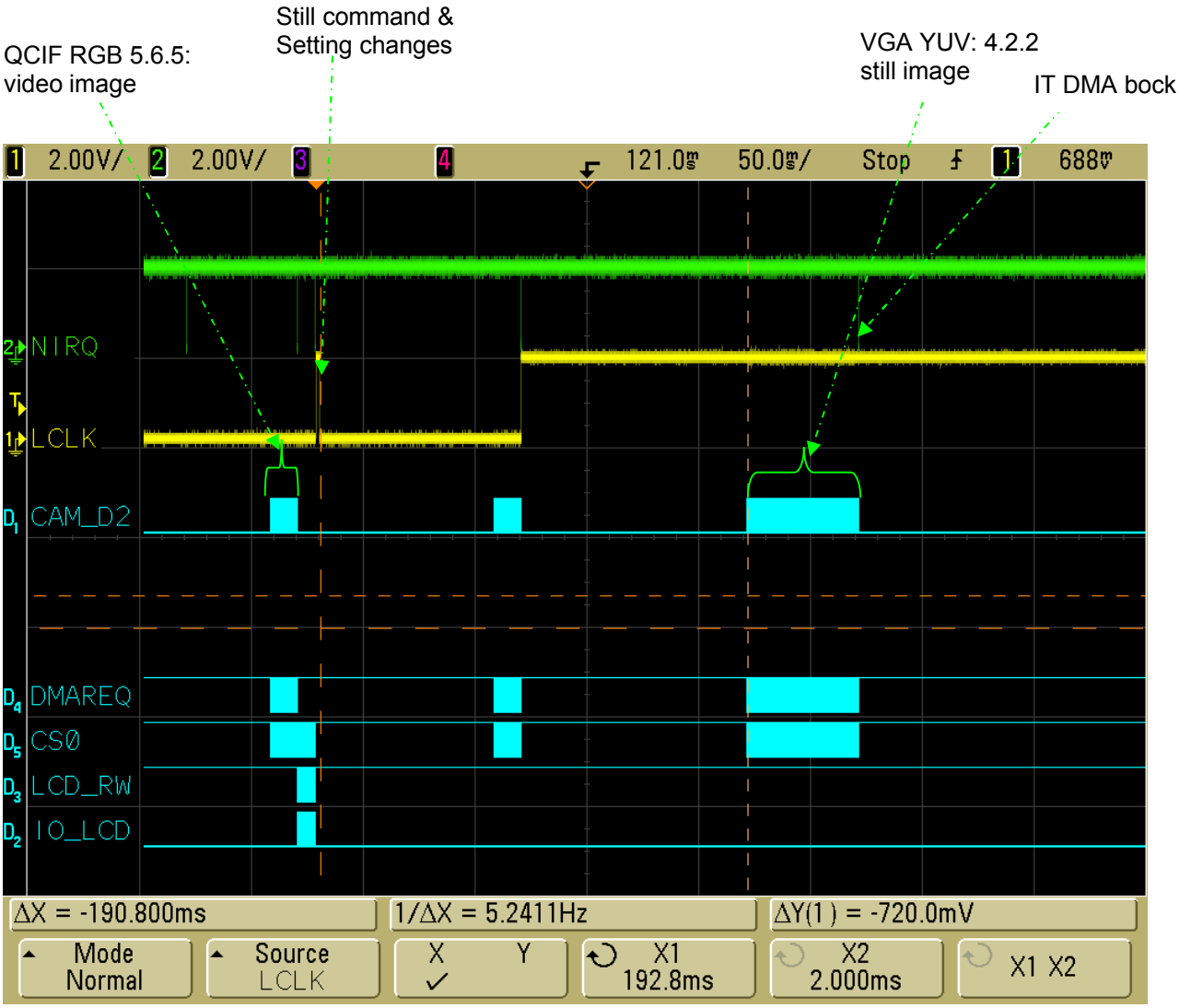

Figure 6 Snap Shot sequence with parallel acquisition plot

Caution:

Between still command sent to the sensor and the still image provided there is a QCIF video image. This is due that the sensor has already a video frame in the pipeline and needs to output it before the still image. The software has to manage it.

3.4 Camera "Snap-Shot" complete System flow

In first situation, snapshot picture is captured then some image processing treatments may done on the picture, then a thumbnail image is generated, and the picture is JPEG encoded by software on LOCOSTO side (JPEG Hw solution is outside the scope of LOCOSTO C027).

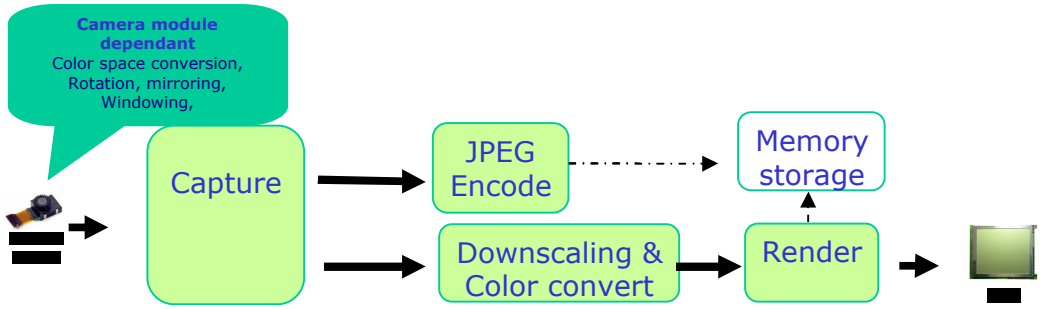

Figure 7 Still image capture flow

After capturing the image the following processing are done:

- JPEG encoding in volatile memory the captured image
- Downscaling image from VGA to QCIF format
- Color convert from YUV 422 to RGB 565 (It is expected sensor outputs in YUYV format and the software JPEG encoder embedded inside LOCOSTO takes YUYV input format)
- Displaying the result on the screen or rendering

Note:

Some sensors can directly output JPEG format (Not the case with the sensor use)

3.4.1 Memory foot print

For YUYV sensors, it is expected working with a ping pong buffer for the capture (up to VGA size 16bpp, windowing provided by the sensor) and a temporary buffer for making the color space conversion, plus another buffer called the video plane for the rendering.

	YUYV format	Value (in KB)
1	VGA ping buffer containing YUYV data 2 bytes/pixel	600
2	VGA pong buffer containing YUYV data 2 bytes/pixel	600
3	QCIF buffer containing YUYV data 2 bytes/pixel.	49.5
4	QCIF buffer containing RGB data 2 bytes/pixel.	49.5
	Total	**1299**

Table 9 Complete Snap Shot system memory footprint

4 System performances

4.1 Definition

Capture time: Capture time is defined as the time needed by the system to output in external volatile memory data coming from the sensor on the shutter action from the user.

Display time: Or rendering time is defined as the time needed by the system to display the image captured on the LCD display.

Preview Time: Preview time is defined as the time needed by the system to grab one frame from the sensor and to render it onto the screen with eventually some processing requested by the done by software. Depending if the sensor is configured in continuous mode for preview or at a fixed value, the frame rate will be best as possible or fixed by the sensor configuration.

Shutter lag time: Shutter lag time is defined as the time needed by the sensor to output the still image after still command is sent.

Snap Shot time: Snap Shot time is defined as the time needed by the sensor to capture the still image since still command is sent. This time include shutter time and capture time.

Camera shot to shot delay: Shot to shot delay is defined as the time needed by the system from the shutter action from the user (not including the shutter lag) for:

1- Capturing the still image from the sensor (Capture time),
2- And displaying the result on the screen (Display time),
3- And JPEG encoding in volatile memory the captured image.

The time needed for storing the picture in non volatile memory is never included in shot to shot delay as this feature is application implementation dependant (non volatile memory support on which the picture is to be stored, storage time masking to the user, etc).

4.2 Speed Performances view finder

The timing diagram below gives the values measured for the acquisition & display time. Pixel-clock and LCD-clock are fixed to 13 MHz and 6.5 MHz respectively while the Locosto on-chip system clock is 104MHz.

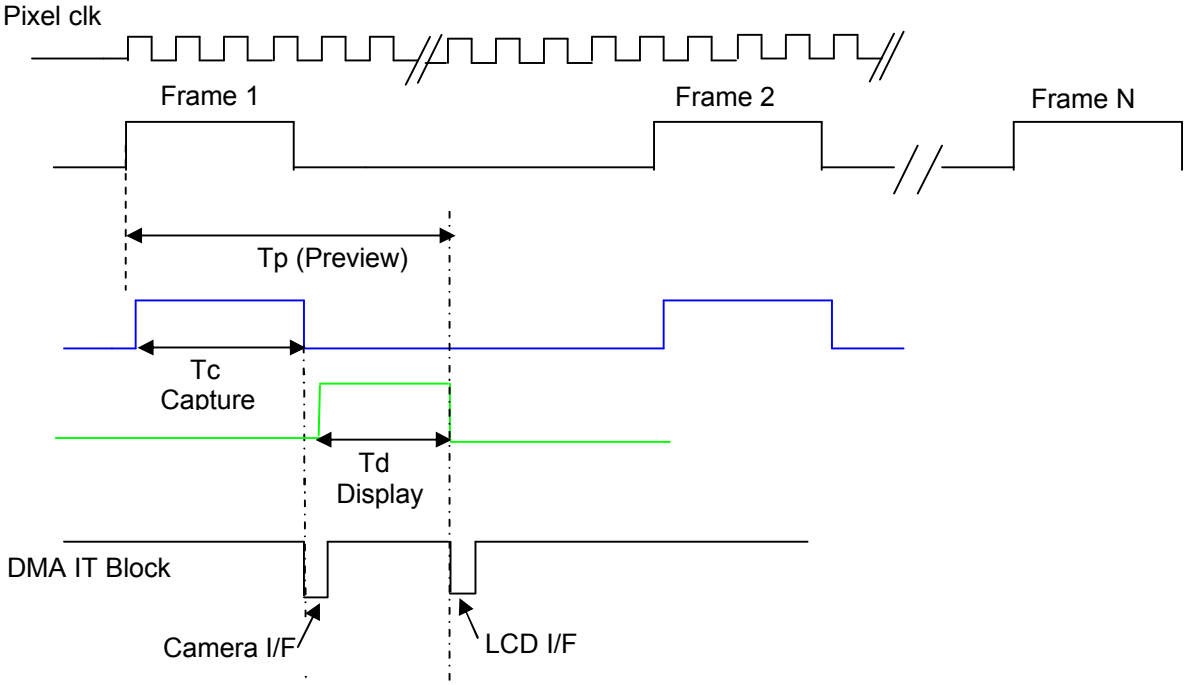

Figure 8 Timing from acquisition to display

Timings	QCIF RGB 5.6.5 view finder mode	176 x 224 [1] RGB 5.6.5 view finder mode
Tc	12 ms	8.02 ms
Td	8.048 ms	12.096 ms
Tp	20.048 ms	20.116 ms

Table 10 View-Finder table measured performances

[1] **176 x 224 RGB 5.6.5 is set on the sensor instead of 176 x 220 because columns row number must be a multiple of 8 according sensor requirement. However 176 x 220 RGB 5.6.5 images are displayed.**

4.3 Speed Performances Snap Shot

The timing diagram below gives the values measured for the acquisition of VGA image.

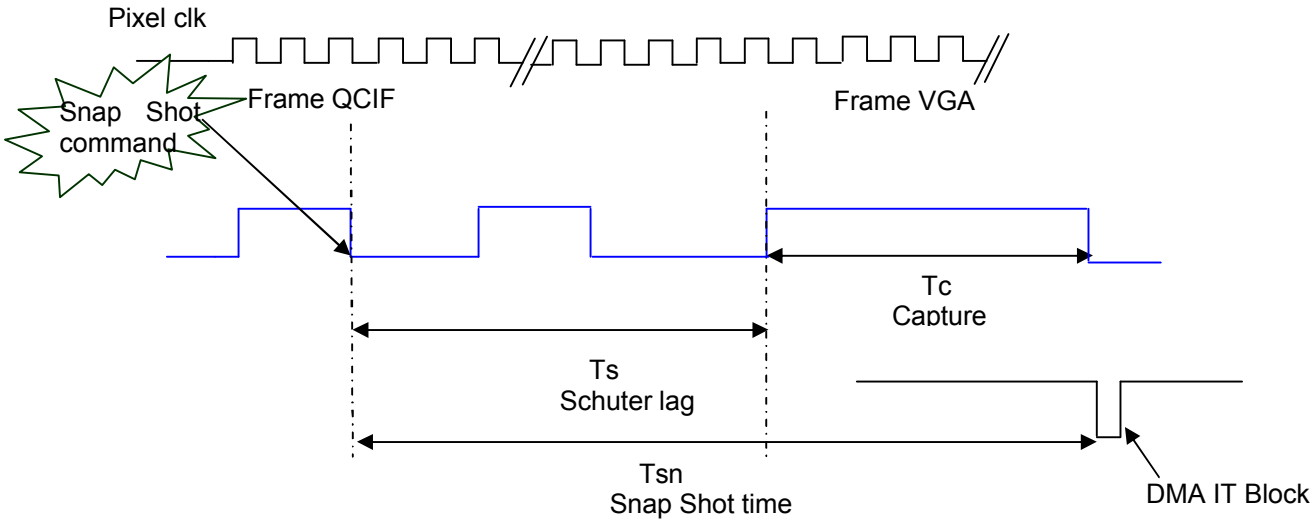

Figure 9 Timing for snap shot

Timings	YUV 4.2.2 Still mode
Tc	50 ms
Ts	190.8 ms
Tsn	240.8 ms

Table 11 Snap Shot table measured performances

4.4 Calculated /expected times

4.4.1 QCIF View-Finder

Capture time:

- Pixel clock = 13 MHz
- Number of pixel (QCIF RGB 5.6.5): 176 x 144 x 2 = 50688 bytes
- Horizontal synchronization duration: 176 x 2 x 1/13Mhz = **27.07 us (27 us measured)**
- **Horizontal Blanking Time measured: 56.8 us**
- DMA write access of the last burst (8 x 16 bits) to external RAM in burst mode at 52 MHz: (6 Wait-State for the first access and 7 others accesses are done at 52 MHz → 13 cycles at 52 MHz) = 250 ns (250 ns measured). Considered negligible.

Tc = (27.07 + 56.8) x 144 = **12.077 ms (12 ms measured)**

The capture time computed above is consolidated with the value of the Horizontal Blanking period that can greatly impact the final frame time capture.

For a QCIF image there is about 52% of blanking period.

Display time:

- LCD clock =6.5 MHz
- Number of pixel (QCIF RGB 5.6.5): 176 x 144 x 2 = 50688 bytes

Td = 50688 x 1/6.5 MHz = 7.8 ms **(8.048 ms measured)**

The difference of 248 us between the measured and expected value is due to the command sent to the LCD to set the address pointer at 0 adding to it the time to enable the DMA transfer.

Preview Time:

Tp = Capture Time + Display Time = 12.365 + 8.048 = **20.412ms**

~ 50 images per second RGB 5.6.5 QCIF image can be reached from hardware point of view!

Sensor frame rate is set to 40 frames per seconds. Note that this frame rate is reached in case of very good brightness condition. In case of dark condition the maximum exposure time is added and so decrease frame rate performances.

4.4.2 Windowing View-Finder

For a windowing format (176 * 224 RGB 5.6.5), the preview time is the same than for a QCIF image. This is explained by the fact that horizontal blanking period is smaller than QCIF image.

Capture time:

- Pixel clock = 13 MHz
- Number of pixel (QCIF RGB 5.6.5): 176 x 224 x 2 = 78848 bytes
- Horizontal synchronization duration: 176 x 2 x 1/13Mhz = **27.07 us (27 us measured)**
- **Horizontal Blanking Time measured: 8.6 us**

Tc = (27.07 + 8.6) x 224 = **7.990 ms (8.02 ms measured)**

For a 176*224 image there is 31 % of blanking period.

Display time:

- LCD clock =6.5 MHz
- Number of pixel (QCIF RGB 5.6.5): 176 x 220 x 2 = 77440 bytes

Td = 77440 x 1/6.5 MHz = 11.91 ms **(12 ms measured)**

For a 176 * 220 image the command sent to the LCD to set the address pointer at 0 is not necessary as the screen controller set automatically the address pointer to 0 when the last address is reached.

Preview Time:

Tp = Capture Time + Display Time = 7.990 + 11.91 = **19.9 ms**

~ 50 images per second RGB 5.6.5 QCIF image can be reached from hardware point of view!

4.4.3 VGA snap shot

Capture time:

- Pixel clock = 13 MHz
- Number of pixel (VGA YUV 4.2.2): 640 x 480 x 2 = 600 Kbytes
- Horizontal synchronization duration: 640 x 2 x 1/13Mhz = **98.46 us (98.4 us measured)**
- **Horizontal Blanking Time measured: 4.8 us**
- Shutter lag time **=** 190.8 ms

Tc = (98.4 + 4.8) x 480 = **49.5 ms (50 ms measured)**

For a VGA image there is 4.8 % of blanking period.

Snap shot time

Tsn = Capture time + Shutter time = 240.3 ms

Note:

The shutter lag time is depends on the brightness condition. In dark environment auto exposure time is added and increases the shutter time.

4.4.4 Shot to Shot delay

Shot to Shot delay:

Tss = Tcapture + Tprocessing + Tdispay + Tjpeg_compression

Before display the image processing is necessary:
- Downscaling Image from VGA to QCIF:
- Color Converting QCIF picture from YUV420 to RGB 565

Jpeg compression is required for image storage.

Operation		Value (in ms)
1	Capture VGA Image: DMA transfers form FIFO camera IF to external memory including shutter time	240.3
2	JPEG decoding VGA resolution YUV420 format, Encode 10:1, from ext. RAM to ext. RAM.	390
3	Downscaling picture format from QVGA to QCIF	59
4	Color Converting QCIF picture from YUV420 to RGB 565	22.5
5	DMA transfer 176*144 pixels, 16bpp (RGB565 format), at 6.5Mhz on an 8 bits bus from external RAM memory to LCD.	8.048
	Total	719.8

Table 12 Shot to shot table performances

Shot to Shot delay time

Tss = 719.8 ms

Note:

- **On the shot to shot delay capture image timing is depends on:**
  - **Horizontal and Vertical low level time (4.8 % in the use case)**
  - **Shutter lag time (190.8 ms in the use case).**

  These parameters depend on the sensor configuration and dark or bright condition.

- JPEG decoding, Downscaling picture and Color Converting processing timings are theoretical timing. The services functions are located in Internal ROM running at 104 MHz with 0 WS and 9 WS for data located in external memory.

5 Constraints

5.1 CPU load & Real time constraints

The CPU is 0% loaded during the view finder, it is fully available during the capture and display phase. There is no software processing in the application. (E.g. YUV to RGB conversion, rotation, zoom …)

With phone activity about 30 % of the bandwidth of the CPU is necessary.

Real-time constraint is not judged very high since it slows down the Fps ratio (display quality is degraded) however, does not lock the system.

5.2 DMA Constraints

The strongest DMA constraint is associated to the Capture phase where both CPU and DMA have to access the external memory bus. If the DMA bandwidth is not properly calibrated, input data is lost hence image cannot be built & displayed…

Two arbitrations must be considered:

- **CPU / DMA concurrent access to:**
  - **The Internal Peripheral bus**.

    The Camera interface FIFO is loaded in (88 x 4) 352 cycles @13MHz pixel-clock for QCIF image that is equal to 1408 cycles@52MHz. During these 1408 cycles the DMA must perform 88[1] accesses @ 52MHz. Therefore a minimum of 88/1408 cycles (6.25%) must be reserved to the DMA on the I-PERIPH bus to insure a proper capture operation.

    The input data rate has a direct impact on DMA activity; a pixel clock down to 6.5 MHz will reduce the necessary DMA bandwidth to 3.125%; Fps (Frames per second) ratio will also be degraded.

  - **External Memory Bus**.

    The Camera interface FIFO is loaded in 1408 cycles @52MHz. During these 88 accesses EMIF must perform 22 x 13 cycles at 52 MHz (286 cycles at 52 MHz). Therefore a minimum of 286/1408 cycles (20 %) must be reserved to the DMA on the EMIF bus to insure a proper capture operation.

[1] **The FIFO is accessed by an internal 32 bits data bus at 52 MHz.**

- **DMA channels concurrence**:

  This depends on system integrated; no concurrence between Camera interface and LCD interface channels…

6 Debug Purpose

Refer to APN 211. (Locosto debug feature application note).

7 ANNEXE

C-code Source

```
// ---------------------------------------------------------------------------
// CAM_start_viewfinder() : View finder sequence
// ---------------------------------------------------------------------------
void CAM_start_viewfinder(void)
{

dma_enh_csr_block_cam =0;
dma_enh_csr_block_lcd =0;

CAM_start_flow();
LCD_load_with_DMA((UWORD32)pic_buffer, LCD_CHANNEL);
CAMERA_Write(0x00,0x02,0x0001);      // start sensor in viewfinder mode

CAMERA_Write(0x20,0x08, 0x8013); // FR_OUT  of OUTPUT_CTRL register to 0
CAMERA_Write(0x20,0x04, 0x0003);    // Enable output of one frame (Set bit 0 0f CMD_2 to 1)

printf("Viewfinder started\n\r");

do{

        while(dma_enh_csr_block_cam == 0);    // IT block camera channel
        dma_enh_csr_block_cam=0;

        LCD_Instruction(RAM_ADD_SET,0x0);    //Set address pointer of LCD to 0
        LCD_Set_Index(GRAM_ACCESS);

        DMA_EnableChannel(LCD_CHANNEL); //Enable LCD DMA transfer

        while(dma_enh_csr_block_lcd == 0);  // IT block LCD channel
        dma_enh_csr_block_lcd=0;

        CAMERA_Write(0x20,0x04, 0x0003); // Enable output of one frame (Set bit 0 0f CMD_2 to 1)

  }while(snap_shot==0); //While snap shot flag set
}

// Snap shot sequence upon Press Key
DMA_DisableChannel(CAM_CHANNEL); //Disable DMA camera channel
*(UWORD16 *)(0x09700044) = 0x14F; // Change Threshold value
*(UWORD16 *)(0xFFFFE850) = 80; //Update element number for VGA image
*(UWORD16 *)(0xFFFFE852) = 1920; //Update frame number for VGA image
DMA_EnableChannel(CAM_CHANNEL); //Enable DMA camera channel
CAMERA_Write(0x00,0x02,0x0002);//Snap shot command
```

```
// ---------------------------------------------------------------------------
// CAM_start_flow() : Camera start flow sequence
// ---------------------------------------------------------------------------
void CAM_start_flow(void)
{
        CAM_stop_flow();  //Stop camera flow
        *(UWORD16 *)(0x09700044) &= ~(0x0100);  //Stop DMA request
        CAMCOR_SmReset(CC_ADD); // reset & flush FIFO
        CAM_CORE.irq_status.all = 0xFFFFFFFF; // reset FIFO status
        Dma_reset_channel(CAM_CHANNEL);     // reset DMA channel
        DMA_EnableChannel(CAM_CHANNEL); // enable DMA camera channel
        *(UWORD16 *)(0x09700044) |= 0x0100;  //Enable DMA request
        CAMCOR_EnableDisable(CC_ADD,CAMCOR_SET);     // Enable FIFO
}

// ---------------------------------------------------------------------------
// CAM_stop_flow() : Camera stop flow sequence
// ---------------------------------------------------------------------------
void CAM_stop_flow(void)
{
        CAMERA_Write(0x00,0x02,0x0000);      // stop sensor
        CAMCOR_EnableDisable(CC_ADD,CAMCOR_CLEAR); // disable FIFO
        DMA_DisableChannel(CAM_CHANNEL); // stop DMA camera channel
}
/*---------------------------------------------------------------------------
 * LCD_load_with_DMA() : displays a 16-bit RGB frame using DMA
 *---------------------------------------------------------------------------*/
ReturnCode_t LCD_load_with_DMA(UWORD32 pic_addr, UWORD16 channel_number)
{
  union DMA_CSDP_REG dummy_CSDP;
// Configure LCD I/F & controller
Wait_LCD_FIFO_free(); //Check and wait FIFO empty
LCD_DmaDisableM;  //Disable DMA request
LCD_Instruction(RAM_ADD_SET,0x0);  //Set address pointer of LCD to 0
LCD_Set_Index(GRAM_ACCESS);

// Configure DMA
DMA.CHANNEL[channel_number].CSSA_L = (UWORD16)pic_addr; // Set DMA source address
DMA.CHANNEL[channel_number].CSSA_U = (UWORD16)(pic_addr>>16);

dummy_CSDP.all = DMA.CHANNEL[channel_number].CSDP.all; // Set source port
if(pic_addr >= 0xFFFE0000)      dummy_CSDP.bit.SRC = DMA_RHEA;
else if   (pic_addr >= 0xFFD00000)       dummy_CSDP.bit.SRC = DMA_API;
else if   (pic_addr >= 0x09000000)       dummy_CSDP.bit.SRC = DMA_IPERIPH;
else if   (pic_addr >= 0x08000000)       dummy_CSDP.bit.SRC = DMA_IMIF;
else if   (pic_addr >= 0x00400000)       dummy_CSDP.bit.SRC = DMA_EMIF;
DMA.CHANNEL[channel_number].CSDP.all = dummy_CSDP.all;

LCD_DmaEnableM; //Enable DMA request
return RET_OK;
```

```
}

/*-------------------------------------------------------
 * LCD_IF_Config  : Configure LCD interface
 *-------------------------------------------------------*/
void LCD_IF_Config(void)
{
LCD_ResetInterface();    // Reset LCD interface

LCD_INTERFACE_CNTL_REG =
(1|  //Enable Interface
(LCD_CLOCK13_EN << LCD_INTERFACE_CNTL_REG_CLOCK13_EN_POS)| //Enable 13 Mhz Clock
(LCD_TX_CLOCK_DIV2 << LCD_INTERFACE_CNTL_REG_TX_CLOCK_DIV_POS)| //6.5 Mhz TX clock
(LCD_RX_CLOCK_DIV8 << LCD_INTERFACE_CNTL_REG_RX_CLOCK_DIV_POS) |
(LCD_FIFO_EMPTY_IT_DIS << LCD_INTERFACE_CNTL_REG_FIFO_EMPTY_IT_EN_POS)|
(LCD_READ_EMPTY_EVENT_IT_DIS<<LCD_INTERFACE_CNTL_REG_LCD_READ_EVENT_IT_EN_
POS) | // All IT disabled
(LCD_DMA_DIS << LCD_INTERFACE_CNTL_REG_DMA_EN_POS) | //DMA enable
(LCD_INTERFACE_8086<<LCD_INTERFACE_CNTL_REG_MODE_POS)| //Select 8086 mode
(LCD_WRITE_MSB_FIRST<< LCD_INTERFACE_CNTL_REG_FLIP_BYTES_POS) | //MSB first
(LCD_SUSPEND_ENABLED<< LCD_INTERFACE_CNTL_REG_SUSPEND_EN_POS)
(LCD_MIN_FRAME_SIZE_64W << LCD_INTERFACE_CNTL_REG_MIN_FRAME_SIZE_POS)| // FIFO
threshold is set to 64 words 16 bits;
(LCD_2_DUMMY<< LCD_INTERFACE_CNTL_REG_N_DUMMY_POS));

LCD_INTERFACE_LCD_CNTL_REG =
(LCD_NCS0_SEL|   // Set LCD controller register (mainly to select CS0)
(LCD_DATA<< LCD_INTERFACE_LCD_CNTL_REG_LCD_RS_POS) |
(LCD_NO_READ<< LCD_INTERFACE_LCD_CNTL_REG_LCD_START_READ_POS) |
(LCD_RESET_CTRL<< LCD_INTERFACE_LCD_CNTL_REG_LCD_NRESET_POS) |
(LCD_NCS1_UNSEL<< LCD_INTERFACE_LCD_CNTL_REG_LCD_NCS1_POS));
}

// -------------------------------------------------------
// CAM_IF_init() : Camera Initialization
// -------------------------------------------------------
static UWORD32 CAM_IF_config(void)
{
*(UWORD16*)0xFFFFFD08 = 5; //Select the DPLL and enable clock in CNTL_CAMERA_DIV_CLK
register of CKLM module

CAMCOR_EnableDisable(CC_ADD,CAMCOR_CLEAR); // disable acquisition in FIFO
CAMCOR_SmReset(CC_ADD); // reset FIFO
CAMCOR_SoftReset(CC_ADD); // reset module

CAMCOR_SetAutoIdleMode(CC_ADD,CAMCOR_AUTOIDLE_ENABLE); //Auto Idle ON

CAMCOR_ParCamConfigure(CC_ADD, // base Address
CAMCOR_PARMODE_NOBT_8BIT,      //No BT 8 bit mode selected
CAMCOR_PARMODE_FIFO_TEST,
0, // nobt VS signal Polarization active high;
```

```
0, // nobt HS signal Polarization active high;
0, // Data sampled on rising edge
1, // Swapped is enabled
0, // BT Correction not used in noBT mode
0); // NoBt Synchro  0 since VS not available

CAMCOR_FrameTrigSetClear(CC_ADD,CAMCOR_SET); // no sync since VS not available

CAMCOR_IrqMaskAll(CC_ADD); //Mask all IT
// interrupts (Underflow and Overflow set)
CAMCOR_IrqMaskUnmask(CC_ADD,CAMCOR_FIFO_OF_IRQ_EN,CAMCOR_SET);
CAMCOR_IrqMaskUnmask(CC_ADD,CAMCOR_FIFO_UF_IRQ_EN,CAMCOR_SET);

// enable CAM_CORE IT line
PE_InterruptArray[INTH_IT_CAMERA_AL] = CAM_default_isr; //Camera ISR routine
INTH_InitLevel   (INTH_IT_CAMERA_AL, INTH_IRQ,1, INTH_FALLING_EDGE_SENSITIVE);
INTH_EnableOneIT (INTH_IT_CAMERA_AL);

CAMCOR_DmaRequestConfigure(CC_ADD // baseAddress,
                                        cam_packet-1, // fifoThreshold - 1
                                        1);      // enable DMA request


// start clock to camera (useless if XCLK pin is multiplexed instead VS)
CAMCOR_ClockDivider(CC_ADD,4); // Divider by source clock by 4

return CAM_CORE.rev;
}
```

```
//-------------------------------------------------------------------------------
//   Dma_init      : reset + configuration DMA module
//-------------------------------------------------------------------------------
void DMA_init(void)
{
  UWORD16 ChNumber;

  // Allocate channels to ARM
  INT_SetSupervisor();  //Set Supervisor mode
  DMA_Allocation(DMA_CHANNEL_0,(BOOL)DMA_CAR_CH_ARM_ALLOCATION);  //Use for LCD IF
  DMA_Allocation(DMA_CHANNEL_1,(BOOL)DMA_CAR_CH_ARM_ALLOCATION); //Use for camera IF
  DMA_Allocation(DMA_CHANNEL_2,(BOOL)DMA_CAR_CH_ARM_ALLOCATION);
  DMA_Allocation(DMA_CHANNEL_3,(BOOL)DMA_CAR_CH_ARM_ALLOCATION);
  DMA_Allocation(DMA_CHANNEL_4,(BOOL)DMA_CAR_CH_ARM_ALLOCATION);
  DMA_Allocation(DMA_CHANNEL_5,(BOOL)DMA_CAR_CH_ARM_ALLOCATION);
  INT_SetUser(); //Set User mode

  // Set priority
  DMA_PriorityAccess(   DMA_AR_IMIF_PRIO_4 // IMIF priority 4
                        (BOOL)DMA_AR_RHEA_PRIO_DMA,    // DMA Priority on rhea bus
                        (BOOL)DMA_AR_API_PRIO_DMA);      // DMA Priority on API bus


  // Enable Interrupt
  for (ChNumber = 0; ChNumber < DMA_NUMBER_OF_CHANNEL; ChNumber++) {

          DMA_ItStatusReg(ChNumber);  // clear any pending interrupt

          // init IT counters
          DmaCsrTout[ChNumber] = 0;
          DmaCsrDrop[ChNumber] = 0;
          DmaCsrFrame[ChNumber] = 0;
          DmaCsrBlock[ChNumber] = 0;
          DmaCsrHalfBlock[ChNumber] = 0;

          // hook default isr to DMA_inth
          DMA_inth[ChNumber].tout = DMA_tout_default_isr;
          DMA_inth[ChNumber].drop = DMA_drop_default_isr;
          DMA_inth[ChNumber].frame = DMA_frame_default_isr;
          DMA_inth[ChNumber].block = DMA_block_default_isr;
          DMA_inth[ChNumber].half = DMA_half_default_isr;
   }
  // enable DMA IT line
  PE_InterruptArray[INTH_IT_DMA] = DMA_isr;  //DMA ISR routine
  INTH_InitLevel   (INTH_IT_DMA, (INTH_InterruptKind_t)INTH_IRQ,1, INTH_LOW_LEVEL_SENSITIVE);
  INTH_EnableOneIT (INTH_IT_DMA);
}
```

```
//---------------------------------------------------------------------------
//   DMA_config_LCD_channel   : Set up a DMA channel for LCD Tx
//---------------------------------------------------------------------------
DMA_config_LCD_channel(UWORD16 channel_number)
{
 DMA_ChannelDescriptor_t Regular; // Shadow regular register

Dma_reset_channel(channel_number);  //Reset channel

Regular.Csdp.SrcPort              = DMA_IMIF;   // Source port
Regular.Cssa.SrcAdd               = (UWORD32) NULL; //Set on LCD_load_with_DMA() function
Regular.Csdp.DestPort             = DMA_RHEA;  // Destination Port
Regular.Cdsa.DestAdd              = (UWORD32) &LCD_IF.WR.all; // Destination Address
Regular.Ccr.SrcAddressMode    = DMA_ADD_POSTINC;   // Source Post-incrementation mode
Regular.Ccr.DestAddressMode  = DMA_ADD_CONSTANT; //Destination address constant
Regular.Csdp.SrcPack              = DMA_PACKING; //Source Packing
Regular.Csdp.DestPack              = DMA_NO_PACKING; // Destination No Packing
Regular.Csdp.SrcBurst             = DMA_BURST_4;        //Source Burst
Regular.Csdp.DestBurst            =DMA_SINGLE_BURST; // Destination no Burst
Regular.Ccr.Fifoflush             = DMA_FORCE_FLUSH;
Regular.Ccr.Priority              = DMA_LOW_PRIORITY;
Regular.Ccr.Autoinit              =DMA_AUTOINIT_OFF; //auto initialization OFF
Regular.Ccr.SyncNumb              = DMA_SYNC_LCD_TX; // Synchronization on DMA Request LCD TX
Regular.Cicr.TimeoutIntEnable  = DMA_ENABLE; // Timeout IT Enable
Regular.Cicr.DropIntEnable        = DMA_ENABLE; // Drop IT Enable
Regular.Cicr.FrameIntEnable     = DMA_DISABLE; // Frame IT Disable
Regular.Cicr.BlockIntEnable       = DMA_ENABLE; // Block IT Enable
Regular.Cicr.HalfBlockIntEnable=DMA_DISABLE; //Half Block IT Disable
Regular.Cfn.FrameNumber       = LCD_CUSTOM_COLUMNS*2; // Frame Number
Regular.Csdp.TypeSize             = DMA_TYPE_16_BITS; // Element Size: 16 bits
Regular.Cen.EltNumber             = LCD_CUSTOM_ROWS/2; // Element Number

DMA_SetupChannel(channel_number, &Regular);
}
```

```
//---------------------------------------------------------------------------
//   DMA_config_CAM_channel   : Set up a DMA channel for CAMERA module
//---------------------------------------------------------------------------
void DMA_config_CAM_channel(UWORD16 channel_number, UWORD32 dest_addr)
{
 DMA_ChannelDescriptor_t Regular; // Shadow regular register
 UWORD16 dest_port;

Dma_reset_channel(channel_number);

//Set source port
if(dest_addr >= 0xFFFE0000)    dest_port = DMA_RHEA;
else if   (dest_addr >= 0xFFD00000)      dest_port = DMA_API;
else if   (dest_addr >= 0x09000000)      dest_port = DMA_IPERIPH;
else if   (dest_addr >= 0x08000000)      dest_port = DMA_IMIF;
else if   (dest_addr >= 0x00400000)      dest_port = DMA_EMIF;

 Regular.Csdp.SrcPort    = DMA_IPERIPH;   //Destination port
 Regular.Cssa.SrcAdd   =     (UWORD32)     (CC_ADD+CAMC_CC_FIFODATA_OFFSET);//Destination
aadress
Regular.Csdp.DestPort            = dest_port;
Regular.Cdsa.DestAdd             = dest_addr;     // Destination address
Regular.Ccr.SrcAddressMode    = DMA_ADD_CONSTANT; // Source address mode
Regular.Ccr.DestAddressMode =   DMA_ADD_POSTINC;   //   Destination   Post-incrementation   mode
Regular.Csdp.SrcPack             = DMA_NO_PACKING; //Source No Packing
Regular.Csdp.DestPack            = DMA_NO_PACKING;  //Destination No Packing
Regular.Csdp.SrcBurst            = DMA_BURST_4; //Source Burst mode
Regular.Csdp.DestBurst           = DMA_BURST_4; // Destination Burst
Regular.Ccr.Fifoflush            = DMA_FORCE_FLUSH; // Fifo Flush
Regular.Ccr.Priority             = DMA_LOW_PRIORITY; // Priority
Regular.Ccr.Autoinit             = DMA_AUTOINIT_ON;  // auto initialization On
Regular.Ccr.SyncNumb = DMA_SYNC_CAMERA_FIFO;// Synchronization on DMA Request CAM FIFO
Regular.Cicr.TimeoutIntEnable  = DMA_ENABLE; // Timeout IT Enable
Regular.Cicr.DropIntEnable       = DMA_ENABLE; // Drop IT Enable
Regular.Cicr.FrameIntEnable     = DMA_DISABLE; // Frame IT Disable
Regular.Cicr.BlockIntEnable      = DMA_ENABLE; // Block IT Enable
Regular.Cicr.HalfBlockIntEnable=DMA_DISABLE; //Half Block IT Disable
Regular.Cfn.FrameNumber        = CAM_CUSTOM_PIXELS/(2*cam_packet); // Frame Number
Regular.Csdp.TypeSize            = DMA_TYPE_32_BITS; // Element Size: 32 bits
Regular.Cen.EltNumber            = cam_packet; //Element size

DMA_SetupChannel(channel_number, &Regular);
}
```

```
/*-------------------------------------------------------------------------------
 * LCD_Controller_Config() : Recommended boot up sequence for LCD controller
 *-------------------------------------------------------------------------------*/
void LCD_Controller_Config()
{
        // 20ms reset pulse
        LCD_ResetController();
        Delay_us(TIMER1,20000,NULL);
        LCD_EndResetController();

        // Start oscillation
        LCD_Instruction(START_OSC, 0x0);
        LCD_Instruction(START_OSC, 0x1);
        Delay_us(TIMER1,20000,NULL);


/*-----------------------------------------------
 *      Boot sequence from
 *-----------------------------------------------*/
        // Power on sequence
        LCD_Instruction(POW_CTRL2,0x8406);
        LCD_Instruction(POW_CTRL3,0x0004);
        LCD_Instruction(POW_CTRL4,0x0602);
        LCD_Gate_Drv_Serial_Xfer(0x01);
        LCD_Instruction(POW_CTRL5,0x161F);
        LCD_Gate_Drv_Serial_Xfer(0x02);
        LCD_Instruction(POW_CTRL1,0x2064);
        LCD_Gate_Drv_Serial_Xfer(0x00);
        LCD_Instruction(POW_CTRL5,0x361F);
        LCD_Gate_Drv_Serial_Xfer(0x00);
        LCD_Instruction(POW_CTRL4,0x0612);
        LCD_Gate_Drv_Serial_Xfer(0x01);

        // Other settings
        LCD_Instruction(DRV_OUT_CTRL,0x011B);
        LCD_Instruction(GATE_SCAN_POS,0x0000);
        LCD_Gate_Drv_Serial_Xfer(0x06);

        LCD_Instruction(LCD_DRV_WAVE,0x0C00);     // field-AC waveform / n-raster row AC waveform
        LCD_Gate_Drv_Serial_Xfer(0x07);

        LCD_Instruction(GAMA_CTRL1,0x0000);
        LCD_Instruction(GAMA_CTRL2,0x0604);
        LCD_Instruction(GAMA_CTRL3,0x0107);
        LCD_Instruction(GAMA_CTRL4,0x0500);
        LCD_Instruction(GAMA_CTRL5,0x0006);
        LCD_Instruction(GAMA_CTRL6,0x0300);
        LCD_Instruction(GAMA_CTRL7,0x0707);
        LCD_Instruction(GAMA_CTRL8,0x0005);

        LCD_Instruction(FRAME_FREQ_ADJ,0x4000);
```

```
        LCD_Instruction(VERT_SCROL_CTRL,0x0000);
        LCD_Instruction(FST_SCR_DRV_POS,0xDB00);
        LCD_Instruction(SEC_SCR_DRV_POS,0x0000);
        LCD_Instruction(HOR_RAM_ADDR_POS,0xAF00);
        LCD_Instruction(VER_RAM_ADDR_POS,0xDB00);

        LCD_Instruction(LTPS_IF_CTRL,0x0700);
        LCD_Instruction(RAM_ADD_SET,0x0000);
        LCD_Instruction(ENTRY_MODE,0x1230);

        // Display on
        LCD_Instruction(DISP_CTRL1,  0x0001);
        LCD_Instruction(DISP_CTRL1,  0x0021);
        LCD_Gate_Drv_Serial_Xfer(0x00);
        LCD_Instruction(DISP_CTRL1,  0x0023);
        LCD_Instruction(DISP_CTRL1,  0x0037);
        LCD_Instruction(DISP_CTRL2,  0x0503);
}

// ------------------------------------------------------------------------
// CAM_sensor_config() : Sensor configuration
// ------------------------------------------------------------------------
void CAM_sensor_config(void)
{
// Intialisation of I2C interface
I2C_InitConnection();

// Power on sequence
IO_CONF_GPIO_17_REG = 0x18; // gpio + pull up => disable signals to daughter board
Delay_us(TIMER1,20000,NULL);
IO_CONF_GPIO_17_REG = 0x8; // gpio + pull down => enable signals to daughter board
Delay_us(TIMER1,300000,NULL); // FIXME: I2C crashes with 150ms
While ((CAMERA_Read_16reg(0x00,0x04) & 0x0004) != 0){}; // wait for module ready

// configure camera "simple registers"
CAMERA_Write(0x00 (Block),0x02 (Offset),0x0000 (Value)); // stop sensor
CAMERA_Write(0x00,0x02,0x0080); // Clear error status register
CAMERA_Write(0x00,0x02,0x0008); // reset camera sensor
while(CAMERA_Read_16reg(0x00,0x02) != 0x0000) {};   // wait for reset done

CAMERA_Write(0x00,0x0A,0x0803); // display mode is RGB656 in Video mode and YUV 422 in still
mode

CAMERA_Write(0x00,0x08,0x0681);  //QCIF format is video mode and VGA format in still mode
CAMERA_Write(0x00,0x12, 0x0190);  //Set frame rate to 40 fps

CAMERA_Write(0x20,0x08, 0x8019);  //Set FR_OUT  of OUTPUT_CTRL register to 0
```

```
CAMERA_Write(0x00,0x02,0x0004);      // enable the changes
while(CAMERA_Read_16reg(0x00,0x02) != 0x0000) {};

CAMERA_Write(0x01,0x00, 0x0000);   // set prescaler divider to /1 (CLK_DIV register)
CAMERA_Write(0x02,0x18,  0x0000);  //set main image pipeline divider to /1 (CLK_DIV_V register)
CAMERA_Write(0x02,0x38, 0x0000);   // set main image pipeline divider to /1 (CLK_DIV_S register)
CAMERA_Write(0x02,0x1a, 0x0000);   //set V_clk_div_v divider to 0 parallel_ctrl_v register*/
CAMERA_Write(0x02,0x3a, 0x0000);   // set V_clk_div_s divider to 0 parallel_ctrl_s register*/
}

// ---------------------------------------------------------------------------
// CAMERA_IOPinMuxing4: IO configuration without use of VSYNC signal
// ---------------------------------------------------------------------------
void CAMERA_IOPinMuxing4(void)
{

  IO_CONF_GPIO_28_REG = IO_CONF_GPIO_28_REG | 0x0003; // To be used as a camera data pin 7
  IO_CONF_GPIO_29_REG = IO_CONF_GPIO_29_REG | 0x0003; // To be used as a camera data pin 6
  IO_CONF_GPIO_30_REG = IO_CONF_GPIO_30_REG | 0x0003; // To be used as a camera data pin 5
  IO_CONF_ND_NWP_REG = IO_CONF_ND_NWP_REG  | 0x0002;//To be used as a camera data pin 4
  IO_CONF_GPIO_20_REG = IO_CONF_GPIO_20_REG | 0x0002; // To be used as a camera data pin 3
  IO_CONF_NFWP_REG  = IO_CONF_NFWP_REG    | 0x0005; // To be used as a camera data pin 2
  IO_CONF_GPIO_0_REG = IO_CONF_GPIO_0_REG  | 0x0002; // To be used as a camera data pin 1
  IO_CONF_GPIO_47_REG = IO_CONF_GPIO_47_REG | 0x0002; // To be used as a camera data pin 0

  IO_CONF_GPIO_21_REG = IO_CONF_GPIO_21_REG | 0x0001; // To be used as a camera LCLK
  IO_CONF_GPIO_22_REG = IO_CONF_GPIO_22_REG | 0x0001; // To be used as a camera XCLK
  IO_CONF_GPIO_19_REG = IO_CONF_GPIO_19_REG | 0x0001; // To be used as a camera HSYNC
}

// ---------------------------------------------------------------------------
// CAMERA_IOPinMuxing3: IO configuration with use of VSYNC signal
// ---------------------------------------------------------------------------
void CAMERA_IOPinMuxing3(void)
{

  IO_CONF_GPIO_28_REG = IO_CONF_GPIO_28_REG | 0x0003; // To be used as a camera data pin 7
  IO_CONF_GPIO_29_REG = IO_CONF_GPIO_29_REG | 0x0003; // To be used as a camera data pin 6
  IO_CONF_GPIO_30_REG = IO_CONF_GPIO_30_REG | 0x0003; // To be used as a camera data pin 5
  IO_CONF_ND_NWP_REG = IO_CONF_ND_NWP_REG | 0x0002;//To be used as a camera data pin 4
  IO_CONF_GPIO_20_REG = IO_CONF_GPIO_20_REG | 0x0001; // To be used as a camera VSYNC
  IO_CONF_NFWP_REG    = IO_CONF_NFWP_REG    | 0x0005; // To be used as a camera data pin 2
  IO_CONF_GPIO_0_REG   = IO_CONF_GPIO_0_REG  | 0x0002; // To be used as a camera data pin 1
  IO_CONF_GPIO_47_REG = IO_CONF_GPIO_47_REG | 0x0002; // To be used as a camera data pin 0

  IO_CONF_GPIO_21_REG = IO_CONF_GPIO_21_REG | 0x0001; // To be used as a camera LCLK
  IO_CONF_GPIO_22_REG = IO_CONF_GPIO_22_REG | 0x0002; // To be used as a camera data pin 3
  IO_CONF_GPIO_19_REG = IO_CONF_GPIO_19_REG | 0x0001; // To be used as a camera HSYNC
}
```