

**Application programming
Interface for C (C-API)**

© **Fujitsu Microelectronics Europe GmbH**
Am Siebenstein 6-10
63303 Dreieich-Buchsschlag, Germany

History

Revision	Date	Comment
V1.0	01.06.01	New Document
V1.1	07.06.01	Update midlevel functions
V1.2	08.06.01	Update midlevel functions GDC_CMD_PtWD(void)
V1.3	09.07.01	Change overview description

Warranty and Disclaimer

To the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH restricts its warranties and its liability for **all products delivered free of charge** (e.g. software include or header files, application examples, target boards, evaluation boards, engineering samples of IC's etc.), its performance and any consequential damages, on the use of the Product in accordance with (i) the terms of the License Agreement and the Sale and Purchase Agreement under which agreements the Product has been delivered, (ii) the technical descriptions and (iii) all accompanying written materials. In addition, to the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH disclaims all warranties and liabilities for the performance of the Product and any consequential damages in cases of unauthorised decompiling and/or reverse engineering and/or disassembling. **Note, all these products are intended and must only be used in an evaluation laboratory environment.**

1. Fujitsu Microelectronics Europe GmbH warrants that the Product will perform substantially in accordance with the accompanying written materials for a period of 90 days from the date of receipt by the customer. Concerning the hardware components of the Product, Fujitsu Microelectronics Europe GmbH warrants that the Product will be free from defects in material and workmanship under use and service as specified in the accompanying written materials for a duration of 1 year from the date of receipt by the customer.
2. Should a Product turn out to be defect, Fujitsu Microelectronics Europe GmbH entire liability and the customer's exclusive remedy shall be, at Fujitsu Microelectronics Europe GmbH sole discretion, either return of the purchase price and the license fee, or replacement of the Product or parts thereof, if the Product is returned to Fujitsu Microelectronics Europe GmbH in original packing and without further defects resulting from the customer's use or the transport. However, this warranty is excluded if the defect has resulted from an accident not attributable to Fujitsu Microelectronics Europe GmbH, or abuse or misapplication attributable to the customer or any other third party not relating to Fujitsu Microelectronics Europe GmbH.
3. To the maximum extent permitted by applicable law Fujitsu Microelectronics Europe GmbH disclaims all other warranties, whether expressed or implied, in particular, but not limited to, warranties of merchantability and fitness for a particular purpose for which the Product is not designated.
4. To the maximum extent permitted by applicable law, Fujitsu Microelectronics Europe GmbH and its suppliers' liability is restricted to intention and gross negligence.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES

To the maximum extent permitted by applicable law, in no event shall Fujitsu Microelectronics Europe GmbH and its suppliers be liable for any damages whatsoever (including but without limitation, consequential and/or indirect damages for personal injury, assets of substantial value, loss of profits, interruption of business operation, loss of information, or any other monetary or pecuniary loss) arising from the use of the Product.

Should one of the above stipulations be or become invalid and/or unenforceable, the remaining stipulations shall stay in full effect.

1. Overview

The MB87J2120 & MB87P2020 Series Application programming interface (the “API”) is a set of commands written in C language to assist graphics application programs utilizing the 2D Graphics Display Controller. The “API” is to interface between application programs and hardware. By using this “API”, application programs can be made without concerning the code to access to hardware registers. This specification describes about the interface between application program and the “API”. The components of the GDC can be initialized directly with the API.

The initialization of Clock Unit must be done before the first usage of all other parts on the chip, to insure proper chip operation. ClkPdR (Clock Power Down Register) of Clock Unit is a set of enable bits for the clocks provided to the dedicated GDC components (Low Power Management).

The SDRAM controller must be initialized before the first application of PictureMemory and GPU in order to assure right demonstration-service. The OnChip SDRAM Interface holds a microstatemaschine, which has to be setup by downloading a small microprogramm (performance optimization).

The Video RAM has to be partitioned into memory resources called layers. Each layer can be independent configured as a part of the video memory with it’s own size and color space/depth.

The Lavender supports up to 16 predefined Layers and handles up to four layers in one scenario.

The area of a given Layer, which may appear on the resulting display (Window), can be defined as a part of the whole Layer individual for each Layer.

The initialization of the graphic controller is executed in the right sequence, if the function GDC_INIT_GPU () in the API used.

Special drawing functions ('Midlevel functions') are summarized in a module (gdc_drw.c). This midlevel functions use the commands of "API". You can see this module in the structure below.

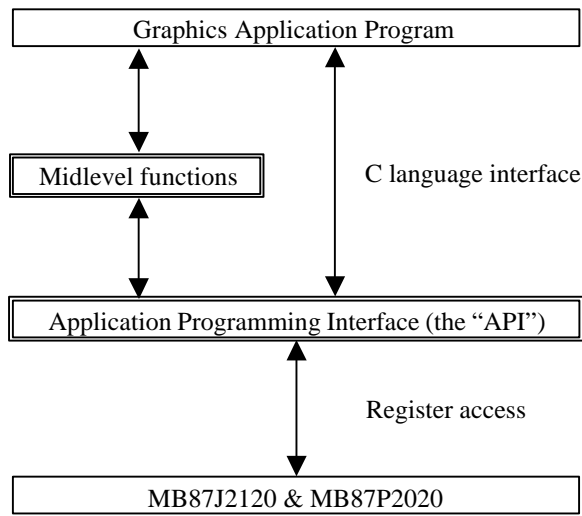


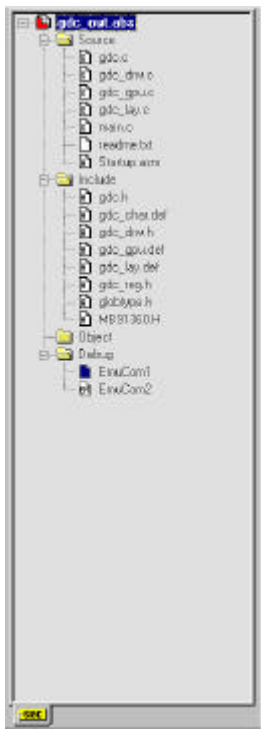
Figure 1-1: Diagram of API

Graphic commands are to interface to the "API" from application programs. There are approximately 87 API functions are supported, depending on each function, such as various primitive draw, display control, and so on. Application programs are able to use various hardware functions featured by the MB87J2120 & MB87P2020, by calling these "API" commands.

2. Creating projects using the API

To create a new project for graphic controller application, it is recommended to use the provided example projects and templates. These projects and templates can be easily modified for own applications. The provided software is written for Fujitsu MB9136x controllers. The library and include files in this installation cannot be used, if you are using a different CPU.

The following figure shows the project-setup to see the necessary files and to learn about the project-settings.



- `gdc.c` : contains all API low-level functions; directed register-access to write colors, coordinates, Layer number, commands... into the corresponding registers
- `gdc_drw.c` : contains API midlevel functions; these functions use several low-level drawing functions from `gdc.c`
- `gdc_gpu.c` : contains API midlevel functions; for setup the GPU master timing, DIPA windows and clock unit
- `gdc_lay.c` : contains API midlevel functions; for setup the GPU display partitioning, merging records and color look up table
- `main.c` : contains the actual source code for the application
- `readme.txt` : text description of the project
- `Startup.asm` : required for the CPU startup

- `gdc.h` : contains all API low-level functions prototypes
- `gdc_char.def` : contains the definition of the characters for the text drawing; ASCII code in the format 8x12 pixels
- `gdc_drw.h` : contains all API midlevel functions prototypes
- `gdc_gpu.def` : GPU timing definition for resolution
- `gdc_lay.def` : GDC display partitioning for resolution
- `gdc_reg.h` : contains all GDC register definitions
- `globtype` : definition of the data type applied in the API
- `MB91360.H` : CPU IO registers
- `EmuCom1` : Emulator setting for Com1
- `EmuCom2` : Emulator setting for Com2

Figure 2-1: Project setup

Figure 2-2 shows a flow chart of graphic controller command execution and a C-example of a graphic controller command (DrawRectangle). For this example the C-API for Lavender and Jasmine is used. Before a new command can be written to command register the flag `FLNOM_CWEN` should be checked. Afterwards the command can be written to the graphic controller to store values for next command in a separate register (e.g. rectangle color for the next DrawRectangle command).

The next step within command execution is to send data to input FIFO. The application has to take care that no FIFO overrun occurs. Therefore it should watch the FIFO flag `FLNOM_IF`. The input FIFO will be completely flushed when the next command is sent to graphic controller. The next command can be written to command register after all data have been sent to input FIFO. In order to keep the code simple and readable a NoOperation command is written to the graphic controller.

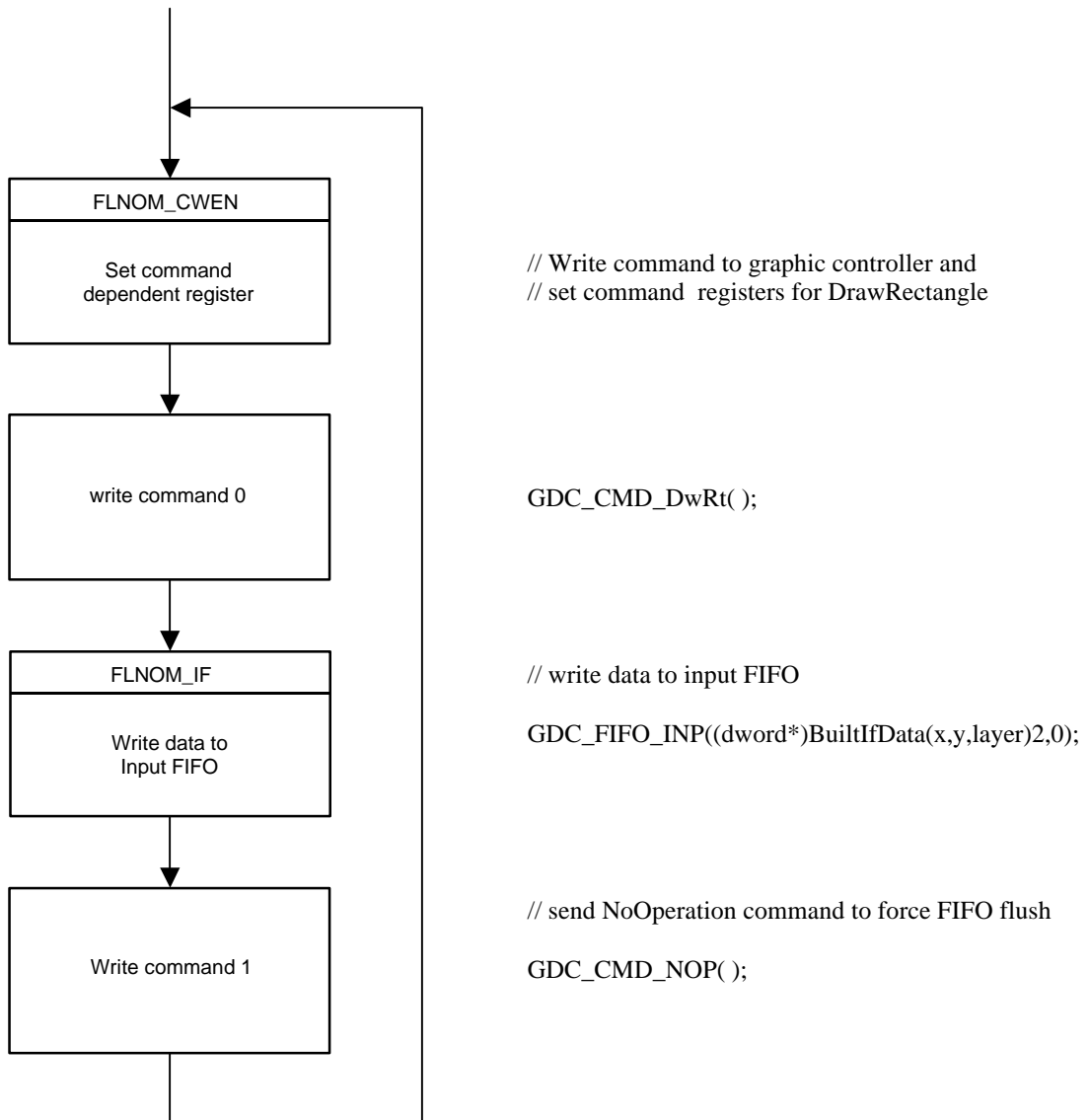


Figure 2-2: Command flow for graphic controller

To create your own projects for graphical applications using the Lavender or Jasmine devices, it is recommended to use and modify the templates.

The template ”_Template” can be easily modified for own applications using appropriate resolution.

3. API Commands

System Control Commands

Command name	Function
GDC_CMD_WAIT	Command Wait Enable flag control
GDC_CMD_NOP	No operation
GDC_CMD_SwRs	Reset by software

Address Decoding Commands

Command name	Function
ULB_ADR_DCD	Address mapping
ULB_ADR_DCD_EN	Enable direct memory access
ULB_ADR_DCD_DIS	Disable direct memory access
ULB_DPA_WR	Function for direct memory access
GDC_CMD_PtPA	Put physical address (FIFO)
GDC_CMD_GtPA	Get physical address (FIFO)
DPA_PHA_CTL	DIPA priority
GDC_GPU_MTODD	Master timing, odd field
GDC_GPU_SCAN	Set scan mode
GDC_SPG_ON	SPG switch on
GDC_SPG_OFF	SPG switch off
GDC_SPG_MIX	Set sync mixer
GDC_SPG_FCT	Set sync mixer output signal

Pixel Commands

Command name	Function
GDC_CMD_PtPx	Draw pixel
GDC_CMD_PxFC	Draw pixel (fixed color)
GDC_CMD_PtWD	Draw dword data
GDC_CMD_PtBM	Draw uncompressed Bit-Map
GDC_CMD_PtCP	Draw compressed Bit-Map
GDC_CMD_GtPx	Get pixel
GDC_CMD_XcPx	Exchange pixel
GDC_CMD_DwLn	Draw line
GDC_CMD_DwPy	Draw polygon
GDC_CMD_DwRt	Draw rectangle
GDC_CMD_TxBM	Draw uncompressed pixel
GDC_CMD_TxCP	Draw compressed pixel
GDC_CMD_MmCy	Memory copy

Functions Supporting Commands

Command name	Function
GDC_FIFO_INP	Feed input FIFO
GDC_FIFO_OUT	Feed output FIFO
GDC_FIFO_IOS	Feed input and output FIFO
GDC_FIFO_Bsy	FIFO is not prepared
ULB_IFO_LIM	Input FIFO limit
ULB_OFO_LIM	Output FIFO limit
DPA_PHA_IFO	Input FIFO limits for DIPA
DPA_PHA_OFO	Output FIFO limits for DIPA
ULB_DMA_HDG	DMA handling
ULB_FLG_ERR	Error codes

Graphic Controller Initialization Commands

Command name	Function
GDC_INIT_GPU	Initialization of graphic controller
GDC_GPU_IDSP	Initialize sync. Timing
GDC_SDC_ISEQ	Initialize SDC RAM controller
GDC_ULB_IWND	Initialize DIPA windows
GDC_GPU_ILDR	Initialize layer
GDC_GPU_IMDR	Initialize merging records
GDC_GPU_ILUT	Initialize color look up table
GDC_GPU_MTON	Master timing ON
GDC_GPU_MTOFF	Master timing OFF

Display Setup Commands

Command name	Function
GPU_DIR_PS	Physical size of display
GPU_DIR_FC	Format of output stream
GPU_DIR_SM	Scan mode for display
GPU_DIR_AS	Pin assignment of signals
GPU_DIR_IP	Display interrupt position

Layer Setup Commands

Command name	Function
GPU_LDR_PA	Layer start address
GPU_LDR_DZ	Layer domain size
GPU_LDR_FP	Set first pixel position
GPU_LDR_WS	Window size
GPU_LDR_WO	Window offset
GPU_LDR_CC	Set color space code
GPU_LDR_TC	Set transparency color
GPU_LDR_BC	Set blink color
GPU_LDR_AC	Set alternative blink color
GPU_LDR_BR	Set blinkrate
GPU_MDR_BK	Blink enable flag
GPU_LDR_TE	Transparence enable flag
GPU_LDR_LD	Line doubling flag
GPU_MDR_ZO	Priority of displayed layers
GPU_MDR_ZOP	Assign layer to plane
GPU_MDR_ZOD	Enable the plane

Color Control Commands

command name	Function
GPU_MDR_CC	Intermediate color space code
GPU_MDR_TC	Intermediate transfer code
GPU_MDR_CO	CLUT offset
GPU_MDR_PL	Duty ratio modulation
GPU_MDR_CLUT	Set CLUT entries
GPU_MDR_CLUTGET	Get CLUT entries
GPU_MDR_CLLD	Set CLUT color entry of multiple color
GPU_MDR_BCEN	Background color enable
GPU_MDR_BC	Background color
GPU_DIR_CC	Physical color space code
GPU_DIR_CCL	Limits of color key

Chip Operation

Command name	Function
GDC_CU_INITi	Clock unit control information
GDC_PWR_Up	Switch modules on
GDC_PWR_Dwn	Switch modules off
GDC_PWR_Chk	Check module status

Anti Aliasing Commands

Command name	Function
PXP_AAF_THE	Enable anti aliasing filter 2x2
PXP_AAF_THO	Enable anti aliasing filter 4x4
PXP_AAF_THR	Boundary values for red
PXP_AAF_THG	Boundary values for green
PXP_AAF_THB	Boundary values for blue

4. Midlevel Commands

Command name	Function
gdc_lib_pform	Generate pixel coordinates
DrawRect	Draw rectangular
DrawFrame	Draw frame
DrawLine	Draw line
DrawPoint	Draw point
out_long	Character conversion
DrawChar	Draw character
DrawText	Draw text
PlotChar	Draw character vertical
PlotText	Draw text vertical
MemCopy	Copy rectangular
DrawPicture	Draw picture
DrawPictMask	Draw picture with ignore color
Draw8Pix	Draw 8 pixels
DrawCircle	Draw circle
GetPix	Get pixel color
SetAAFLevel	Boundary values for AAF
SetLayDomain	Set layer domain
SetLayWindow	Set layer window
SetLayCForm	Set layer color format
SetLayTrans	Set layer transparent color
SetLayBlink	Set layer blink color

5. Data Format

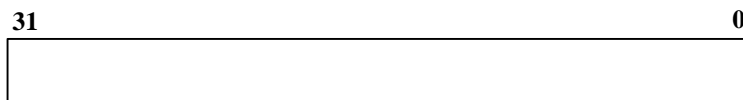
Data Type

Format	Description
byte	unsigned char – 8 bit
word	unsigned short – 16 bit
dword	unsigned long – 32 bit

These data types are for the Fujitsu FR-compiler only

Data String Format

physical byte address

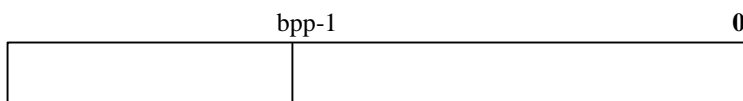


Note: Depending on video memory size not all addresses are used

data word

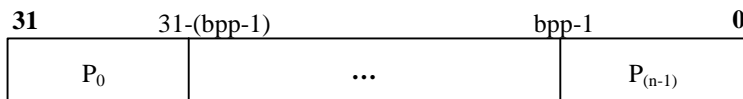


single color data (LSB aligned)



bpp...24, 16, 8, 4, 2, 1 bit

color data



bpp...24, 16, 8, 4, 2, 1 bit
n...count of pixel per word

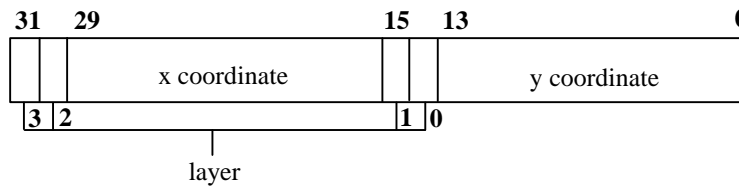
bpp	n
24	1
16	2
8	4
4	8
2	16
1	32

color enable data



- (....) source or target FIFO data
- [....]n deliver data in '[....]' n times
- [....]+ deliver data in '[....]' at least one time

Pixel coordinates



6. API Command Reference

System Control Commands

GDC_CMD_WAIT [Command Wait Enable flag control]

Format void GDC_CMD_WAIT (void)

Parameter None

Return value None

Description Wait until Command Write Enable flag is set

GDC_CMD_NOP [No operation]

Format word GDC_CMD_NOP (void)

Parameter None

Return value 0 - no error codes

Description No Operation – used to stop infinite commands and to closed up last drawing command

GDC_CMD_SwRs [Reset by software]

Format word GDC_CMD_SwRs (void)

Parameter None

Return value 0 - no error codes

Description Software Reset – stop command mode of Pixel Engine and reset FIFO addressing

Address Decoding Commands

ULB_ADR_DCD [Address mapping]

Format word ULB_ADR_DCD (byte hwnd_vram, dword gdc_offs, dword gdc_size, dword sdc_offs)

Parameter	hwnd_vram	window-handle - hwnd_sdram (0,1)
	gdc_offs	MCU offset for SDRAM window 'hwnd_vram'
	gdc_size	size of SDRAM window 'hwnd_vram'
	sdc_offs	SDRAM offset for SDRAM window 'hwnd_vram'

Return value 0 - no error codes

Description Lavender Address mapping
2 windows of the Lavender-SD-Ram (Lavender1:8MB;Lavender2:0.5MB) can be mapped to the memory space of the controller. 2MB of memory is reserved for the Lavender; the lower memory space of the Lavender memory is reserved for register access (0x00 .. 0x3FFFF); upon this address the memory is free to be defined for direct access via windows! The windows are selected by a window-handle (hwnd_sdram) and have to be enable after both are defined; to enable the windows (both) call ULB_ADR_DCD_EN(), to disable call ULB_ADR_DCD_DIS(); make sure, that none of the windows are pointing to the register map!

ULB_ADR_DCD_EN [Enable direct memory access]

Format	void ULB_ADR_DCD_EN (void)
Parameter	None
Return value	0 - no error codes
Description	Enables the direct memory access to SDRAM memory via the windows set with ULB_ADR_DCD

ULB_ADR_DCD_DIS [Disable direct memory access]

Format	void ULB_ADR_DCD_DIS (void)
Parameter	None
Return value	0 - no error codes
Description	Disable the direct memory access to SDRAM memory via the windows set with ULB_ADR_DCD

ULB_DPA_WR [Function for direct memory access]

Format	void ULB_DPA_WR (dword *adr, dword data)	
Parameter	*adr data	address from controller memory space data to write to adr
Return value	0 - no error codes	
Description	function for direct memory access via windows	

GDC_CMD_PtPA [Put physical address (FIFO)]

Format	word GDC_CMD_PtPA (dword phy_adr)	
Parameter	phy_adr	physical address of graphic controller memory map
Return value	0 - error codes	
Description	Put Physical Address - store data in VRAM direct with address auto-increment	

GDC_CMD_GtPA [Get physical address (FIFO)]

Format	word GDC_CMD_GtPA (dword phy_adr, dword wcnt)	
Parameter	phy_adr wcnt	physical start address of graphic controller memory map number (wcnt) of data (dwords) to be read from SDRAM
Return value	number of dword which have been read	
Description	Get data from physical address load data from VRAM with address auto-increment, stop after n dwords	

DPA_PHA_CTL [DIPA priority]

Format	word DPA_PHA_CTL (word mode)	
Parameter	mode	bits [2: 0] IPA priority for SDC access bits [18:16] DPA priority for SDC access
Return value	0 - no error codes	
Description	Set priority for IPA and DPA for SDC access into DIPA control register RAM internal; optimization for speed	

GDC_GPU_MTODD [Master timing, odd field]

Format	void GDC_GPU_MTODD (word xs, word ys, word xe, word ye)	
Parameter	xs	x start
	ys	y start
	xe	x stop
	ye	y stop
Return value	None	
Description	Sets the number of scan points in each direction, surround the area with visible pixels on the physical screen display	

GDC_GPU_SCAN [Set scan mode]

Format	void GDC_GPU_SCAN (word scan, word ofs)	
Parameter	scan	physical size in scan clocks (pixel*bpp/bits per scan clock)
	ofs	dual scan Y offset
Return value	None	
Description	Determines the scan mode of the display, i.e. the number and layout of pixel streams written in parallel to the physical display.	

GDC_SPG_ON [SPG switch on]

Format	void GDC_SPG_ON (byte spg, word xs, word ys, word xm, word ym)			
Parameter	spg			
	xs			
	ys			
	xm			
	ym			
Return value	None			
Description	Setup front edge of sync pulse generator – sync pulse generator position to switch on			

GDC_SPG_OFF [SPG switch off]

Format	void GDC_SPG_OFF (byte spg, word xs, word ys, word xm, word ym)			
Parameter	spg			
	xs			
	ys			
	xm			
	ym			
Return value	None			
Description	Setup back edge of sync pulse generator – sync pulse generator position to switch off			

GDC_SPG_MIX [Set sync mixer]

Format	void GDC_SPG_ON (byte spg, byte s0, byte s1, byte s2, byte s3, byte s4)			
Parameter	spg			
	s0			
	s1			
	s2			
	s3			
	s4			
Return value	None			
Description	Setup the mixer for the sync generator are used			

GDC_SPG_FCT [Set sync mixer output signal]

Format void GDC_SPG_FCT (byte spg, dword ftab)

Parameter spg
 ftab

Return value None

Description Setup the sync mixer for the output signal

Pixel Commands

GDC_CMD_PtPx [Draw pixel]

Format word GDC_CMD_PtPx (void)

Parameter None

Return value 0 - no error codes

Description Put pixel – store pixel data with color in VRAM

GDC_CMD_PxFC [Draw pixel (fixed color)]

Format word GDC_CMD_PxFC (dword color)

Parameter color single color data (right aligned)

Return value 0 - no error codes

Description Put pixel – store pixels with same color in VRAM
 color is passed by parameter, therefore a smaller data set for FIFO is required!

GDC_CMD_PtWD [Draw dword data]

Format word GDC_CMD_PtWD (void)

Parameter None

Return value 0 - no error codes

Description Put dword data – store pixel word data in VRAM word count = pixel count / bpp

GDC_CMD_PtBM [Draw uncompressed Bit-Map]

Format	word GDC_CMD_PtBM (word xmin, word xmax, word ymin, word ymax, dword ign_col, byte ign_ena, byte dir, byte mirror, byte layer)	
Parameter	xmin	start position x
	xmax	stop position x
	ymin	start position y
	ymax	stop position y
	ign_col	color which is to be ignored if ign_ena is set
	ign_ena	enable/disable ignore color 'ign_col' 0: disable 1: enable
	dir	draw direction 0: horizontal 1: vertical
	mirror	mirror function 00: none 01: x-mirror 10: y-mirror 11: xy- mirror
	layer	layer select
Return value	0 - no error codes	
Description	Store uncompressed Bit-Map data in VRAM. Start at {xmin, ymin}, stop at {xmax, ymax} Bit-Map data: $n = (xmax - xmin) * (ymax - ymin) * bpp / 32$	

GDC_CMD_PtCP [Draw compressed Bit-Map]

Format	word GDC_CMD_PtCP (word xmin, word xmax, word ymin, word ymax, dword ign_col, byte ign_ena, byte dir, byte mirror, byte layer)	
Parameter	xmin	start position x
	xmax	stop position x
	ymin	start position y
	ymax	stop position y
	ign_col	color which is to be ignored if ign_ena is set
	ign_ena	enable/disable ignore color 'ign_col' 0: disable 1: enable
	dir	draw direction 0: horizontal 1: vertical
	mirror	mirror function 00: none 01: x-mirror 10: y-mirror 11: xy- mirror
	layer	layer select
Return value	0 - no error codes	
Description	Store compressed Bit-Map data (run length coded) in VRAM; start at {xmin, ymin}, stop at {xmax, ymax} Bit-Map data: $n = (xmax - xmin) * (ymax - ymin) * \text{compressed factor}$	

GDC_CMD_GtPx [Get pixel]

Format word GDC_CMD_GtPx (void)
Parameter None
Return value 0 - no error codes
Description Get pixel – load pixel data from VRAM

GDC_CMD_XcPx [Exchange pixel]

Format word GDC_CMD_XcPx (void)
Parameter None
Return value 0 - no error codes
Description Store single pixel data into VRAM and load old pixel data in output FIFO

GDC_CMD_DwLn [Draw line]

Format word GDC_CMD_DwLn (dword line_col)
Parameter line_col logical line color
Return value 0 - no error codes
Description Draw a line with 'line_col'

GDC_CMD_DwPy [Draw polygon]

Format word GDC_CMD_DwPy (dword poly_col)
Parameter poly_col logical polygon color
Return value 0 - no error codes
Description Draw a polygon with 'poly_col'

GDC_CMD_DwRt [Draw rectangle]

Format word GDC_CMD_DwRt (dword rect_col)
Parameter rect_col logical rectangle color
Return value 0 - no error codes
Description Draw a colored rectangle with 'rect_col'

GDC_CMD_TxBM [Draw uncompressed pixel]

Format	word GDC_CMD_TxBM (word xmin, word xmax, word ymin, word ymax, dword frg_col, dword bkg_col, byte bkg_ena, byte dir, byte mirror, byte layer)	
Parameter	xmin	start position x
	xmax	stop position x
	ymin	start position y
	ymax	stop position y
	frg_col	foreground (text) color
	bkg_col	background color if bkg_ena is set
	bkg_ena	enable/disable background color 'bkg_col' 0: disable 1: enable
	dir	draw direction 0: horizontal 1: vertical
	mirror	mirror function 00: none 01: x-mirror 10: y-mirror 11: xy- mirror
	layer	layer select
Return value	0 - no error codes	
Description	Draw uncompressed pixel with fixed foreground and background color transfer 1 bit per pixel (32 pixel per dword) via FIFO	

GDC_CMD_TxCP [Draw compressed pixel]

Format	word GDC_CMD_TxCP (word xmin, word xmax, word ymin, word ymax, dword frg_col, dword bkg_col, byte bkg_ena, byte dir, byte mirror, byte layer)	
Parameter	xmin	start position x
	xmax	stop position x
	ymin	start position y
	ymax	stop position y
	frg_col	foreground (text) color
	bkg_col	background color if bkg_ena is set
	bkg_ena	enable/disable background color 'bkg_col' 0: disable 1: enable
	dir	draw direction 0: horizontal 1: vertical
	mirror	mirror function 00: none 01: x-mirror 10: y-mirror 11: xy- mirror
	layer	layer select
Return value	0 - no error codes	
Description	<p>Draw compressed pixel (RLE) with fixed foreground and background color</p> <p>RLE data stream are a continuous data stream with 1 command byte and n data bytes</p> <p>RLE compressed command byte (bit 7...0) : 1NNNNNNC 1 : compression sign NNNNNN : n bits colored with C C : 1 = foreground color : 0 = background color if enable</p> <p>RLE uncompressed command byte (bit 7...0) : 0NNNNNNNN 0 : uncompressing sign NNNNNNN : n pixels with uncompressed color information (NNNNNNN+1 mod 8)bytes with color information (CCCCCCCC) the last byte may be incomplete</p>	

GDC_CMD_MmCy [Memory copy]

Format	word GDC_CMD_MmCy (void)
Parameter	None
Return value	0 - no error codes
Description	Memory copy from one layer to another/same layer

Functions Supporting Commands

GDC_FIFO_INP [Feed input FIFO]

Format	word GDC_FIFO_INP (dword *p_arr, word pcnt, byte dma_ena)
Parameter	*p_arr array of pixel coordinates (IFIFO) pcnt number of pixel coordinates dma_ena DMA enable/disable
Return value	0 - no error codes
Description	Feed the graphic controller input FIFO from an user array

GDC_FIFO_OUT [Feed output FIFO]

Format	word GDC_FIFO_OUT (dword *p_arr, word pcnt, byte dma_ena)
Parameter	*p_arr array of pixel coordinates (OFIFO) pcnt number of pixel coordinates dma_ena DMA enable/disable
Return value	0 - no error codes
Description	Feed an user array from the graphic controller output FIFO

GDC_FIFO_IOS [Feed input and output FIFO]

Format	word GDC_FIFO_IOS (dword *p_arr, dword *q_arr, word pcnt, byte dma_ena)
Parameter	*p_arr array of pixel coordinates (IFIFO) *q_arr array of pixel coordinates (OFIFO) pcnt number of pixel coordinates dma_ena DMA enable/disable
Return value	0 - no error codes
Description	Feed the graphic controller input FIFO from an user array, while feed an user array from the graphic controller output FIFO

GDC_FIFO_Bsy [FIFO is not prepared]

Format	word GDC_FIFO_Bsy (void)
Parameter	None
Return value	0 - no error codes
Description	Signal the user that the FIFO routine is not prepared to be used

ULB_IFO_LIM [Input FIFO limit]

Format	word ULB_IFO_LIM (byte iffo_low, byte iffo_high)	
Parameter	iffo_low iffo_high	IFIFO low limit IFIFO high limit
Return value	0 - no error codes	
Description	Set input FIFO limits	

ULB_OFO_LIM [Output FIFO limit]

Format	word ULB_OFO_LIM (byte offo_low, byte offo_high)	
Parameter	offo_low offo_high	OFIFO low limit OFIFO high limit
Return value	0 - no error codes	
Description	Set output FIFO limits	

DPA_PHA_IFO [Input FIFO limits for DIPA]

Format	word DPA_PHA_IFO (word min, word max)	
Parameter	min max	min. block size for data transfer to the VRAM : bits [15: 0] max. block size for data transfer to the VRAM : bits [31:16]
Return value	0 - no error codes	
Description	Set input FIFO limits for DIPA which is necessary for flags	

DPA_PHA_OFO [Output FIFO limits for DIPA]

Format	word DPA_PHA_OFO (word min, word max)	
Parameter	min max	min. block size for data transfer from the VRAM : bits [15: 0] max. block size for data transfer from the VRAM : bits [31:16]
Return value	0 - no error codes	
Description	Set output FIFO limits for DIPA which is necessary for flags	

ULB_DMA_HDG [DMA handling]

Format	word ULB_DMA_HDG (byte offo_low, byte offo_high, word modes, word block, int direction)	
Parameter	offo_low offo_high modes block direction	OFIFO low limit OFIFO high limit transfer mode setting 00: block/step transfer mode 01: burst transfer mode 10: demand transfer mode block transfer size transfer direction 0: output FIFO 1: input FIFO
Return value	0 - no error codes	
Description	Settings for DMA handling for graphic controller and MCU block/step and burst mode is the same for graphic controller	

ULB_FLG_ERR [Error codes]

Format	dword ULB_FLG_ERR (void)
Parameter	None
Return value	0 - no error codes
Description	Error codes for graphic controller err_cmd_code (0) '1' : wrong command instruction code detected err_cmd_overflow (1) '1' : graphic controller command pipeline overflow (too many commands) err_data_underflow (2) '1' : input FIFO data underflow for current command (too few data for command) err_mcp_bpp (3) '1' : color depth of source and target layer for 'MemCp' command was unequal -> MCP stops command execution

Graphic Controller Initialization Commands

GDC_INIT_GPU [Initialization of graphic controller]

Format	word GDC_INIT_GPU (word mode)
Parameter	mode decide which blocks of the graphic controller is initialized
Return value	0 - no error codes
Description	Initialize the whole GPU for customer display

GDC_GPU_IDSP [Initialize sync. Timing]

Format	word GDC_GPU_IDSP (void)
Parameter	None
Return value	0 - no error codes
Description	Sync timing for customer display (internal sub function of GDC_INIT_GPU)

GDC_SDC_ISEQ [Initialize SDC RAM controller]

Format	word GDC_SDC_ISEQ (void)
Parameter	None
Return value	0 - no error codes
Description	Setup the sync sequencer (internal sub function of GDC_INIT_GPU)

GDC_ULB_IWND [Initialize DIPA windows]

Format	word GDC_ULB_IWND (void)
Parameter	None
Return value	0 - no error codes
Description	Setup the DIPA memory window for direct MCU access of the display memory (internal sub function of GDC_INIT_GPU)

GDC_GPU_ILDR [Initialize layer]

Format word GDC_GPU_ILDR (void)

Parameter None

Return value 0 - no error codes

Description Initialize the layer description records for customer display
(internal sub function of GDC_INIT_GPU)

GDC_GPU_IMDR [Initialize merging records]

Format word GDC_GPU_IMDR (void)

Parameter None

Return value 0 - no error codes

Description Initialize the merging description records for customer display
(internal sub function of GDC_INIT_GPU)

GDC_GPU_ILUT [Initialize color look up table]

Format word GDC_GPU_ILUT (void)

Parameter None

Return value 0 - no error codes

Description Initialize the CLUT description records for customer display
(internal sub function of GDC_INIT_GPU)

GDC_GPU_MTON [Master timing ON]

Format word GDC_GPU_MTON (void)

Parameter None

Return value 0 - no error codes

Description Switch master timing ON (internal sub function of GDC_INIT_GPU)

GDC_GPU_MTOFF [Master timing OFF]

Format word GDC_GPU_MTOFF (void)

Parameter None

Return value 0 - no error codes

Description Switch master timing OFF

Display Commands

GPU_DIR_PS [Physical size of display]

Format	word GPU_DIR_PS (word PX, word PY)	
Parameter	PX	display width in pixels
	PY	display height in lines
Return value	0 - no error codes	
Description	Set physical size of display setting the dimension of the viewable area of the display	

GPU_DIR_FC [Format of output stream]

Format	word GPU_DIR_FC (word format)	
Parameter	format	code number stream format
		Code Depth
		0: 1 bpp
		1: 2 bpp
		2: 4 bpp
		3: 6 bpp
		4: 8 bpp
		5: 9 bpp
		6: 12 bpp
		7: 18 bpp
		8: 24 bpp
		9: 24 bpp analog output
		other codes are not useable
Return value	0 - no error codes	
Description	Setting the format of the physical output stream	

GPU_DIR_SM [Scan mode for display]

Format	word GPU_DIR_SM (byte scan_mode)	
Parameter	scan_mode	scan mode number
		Code Mode
		0 normal
		1 dual scan
		2 zigzag mode
		other modes are not useable
Return value	0 - no error codes	
Description	Setting the special scan mode if necessary for display	

GPU_DIR_AS [Pin assignment of signals]

Format	word GPU_DIR_AS (dword pin_asset)	
Parameter	pin_asset	assignment for pins bit[28] for DAC bit[27] for ColKey bit[26] for Csync bit[25] for Vsync bit[24] for Hsync bit[23:16] for R7...R0 bit[15: 8] for G7...G0 bit[7: 0] for B7...B0
Return value	0 - no error codes	
Description	Set pin assignment for signal setting the output enable for the physical pins	

GPU_DIR_IP [Display interrupt position]

Format	word GPU_DIR_IP (dword irq_dispos)	
Parameter	irq_dispos	display interrupt position
Return value	0 - no error codes	
Description	Setting the display interrupt position	

Layer Commands

GDC_LDR_PA [Layer start address]

Format	word GDC_LDR_PA (word Layer, dword Phys_adr)	
Parameter	Layer	layer select
	Phys_adr	physical address of the first pixel for this layer domain this value must be aligned to 2^{10}
Return value	0 - no error codes	
Description	Setting the physical start address in the VRAM for a layer	

GPU_LDR_DZ [Layer domain size]

Format	word GDC_LDR_DZ (word Layer, word dx, word dy)	
Parameter	Layer	layer select
	dx	domain width in pixels
	dy	domain height in lines
Return value	0 - no error codes	
Description	Setting the domain size of a layer it defines the amount of memory which can be used by a window inside this domain	

GPU_LDR_FP [Set first pixel position]

Format	word GDC_LDR_FP (word Layer, word fx, word fy)	
Parameter	Layer	layer select
	fx	left origin in pixel
	fy	upper origin in lines
Return value	0 - no error codes	
Description	Setting the first pixel position of a window in its domain	

GPU_LDR_WS [Window size]

Format	word GDC_LDR_WS (word Layer, word wx, word wy)	
Parameter	Layer	layer select
	wx	window width in pixels
	wy	window height in lines
Return value	0 - no error codes	
Description	Setting the size of a window	

GDC_LDR_WO [Window offset]

Format	word GDC_LDR_WO (word Layer, word ox, word oy)	
Parameter	Layer	layer select
	ox	left origin in pixels
	oy	upper origin in lines
Return value	0 - no error codes	
Description	Setting the offset for a window in the display	

GPU_LDR_CC [Set color space code]

Format	word GDC_LDR_CC (word Layer, word cspc)	
Parameter	Layer	layer select
	cspc	code number of the pixel format for a layer domain useful codes are: Code Col.Depth Out path 0 : 1 bpp B/W CLUT useable 1 : 2 bpp Col4 CLUT useable 2 : 4 bpp Col16 CLUT useable 3 : 8 bpp Col256 CLUT useable 6 : 24 bpp RGB888 direct specialized codes are : 4 : 16 bpp RGB555 direct / aligned 5 : 16 bpp RGB565 direct / aligned Very specialized codes are : 7 : 16 bpp YUV422 matrix 8 : 24 bpp YUV444 matrix unusable Codes are : 9..14 reserved codes are: 15: 0 bpp -/- no output
Return value	0 - no error codes	
Description	Setting the color space code for a layer this defines the color depth for a domain, which is called the logical color space it belongs to the overall amount of memory	

GPU_LDR_TC [Set transparency color]

Format	word GDC_LDR_TC (word Layer, dword transcol)	
Parameter	Layer transcol	layer select logical color of this layer, which defines the transparent pixels
Return value	0 - no error codes	
Description	Setting the transparency color of a layer each pixel in this layer which carries this color will go transparent if the transparency enable flag for this layer is set	

GPU_LDR_BC [Set blink color]

Format	word GDC_LDR_BC (word Layer, dword blinkcol)	
Parameter	Layer blinkcol	layer select logical color of this layer, which defines the blinking pixels
Return value	0 - no error codes	
Description	Setting blink color of a layer each pixel in this layer which carries this color will go blinking if the blink rate for this layer is set	

GPU_LDR_AC [Set alternative blink color]

Format	word GDC_LDR_AC (word Layer, dword alterncol)	
Parameter	Layer alterncol	layer select logical color of this layer, which defines the 2nd color of the blinking pixels when the blinking color disappear
Return value	0 - no error codes	
Description	Setting alternate blink color of a layer each pixel in this layer which is blinking, changes its color from blink color to alternate blink color	

GPU_LDR_BR [Set blink rate]

Format	word GDC_LDR_BR (word Layer, byte blink_on, byte blink_off)	
Parameter	Layer blink_on blink_off	layer select frame count -1 for pixels be displayed with its blink color frame count -1 for pixels be displayed with its alternate blink color
Return value	0 - no error codes	
Description	Setting blink rate in frame counts of the display if the values for blink_on and blink_off are different from zero, the blinking of the appropriate pixels is ongoing if the values for blink_on and blink_off are zero, the blinking of the appropriate pixels is stopped	

GPU_MDR_BK [Blink enable flag]

Format	word GDC_MDR_BK (word Layer, byte blink_enabled)	
Parameter	Layer blink_enabled	layer select enables/disable the appropriate pixels in this layer to be blinking 1: enable 0: disable
Return value	0 - no error codes	
Description	Setting blink enable/disable flag for this layer	

GPU_LDR_TE [Transparence enable flag]

Format	word GDC_LDR_TE (word Layer, byte transparent)	
Parameter	Layer transparent	layer select enable/disable the appropriate pixels in this layer to be transparent 1: enable 0: disable
Return value	0 - no error codes	
Description	Setting transparence enable/disable flag each pixel in this layer which carries this color will be transparent	

GPU_LDR_LD [Line doubling flag]

Format	word GDC_LDR_LD (word Layer, byte line_doubling)	
Parameter	Layer line_doubling	layer select enable/disable line doubling for this layer what's used with the video input capture device
Return value	0 - no error codes	
Description	Setting line doubling flag for this layer	

GPU_MDR_ZO [Priority of displayed layers]

Format	word GDC_MDR_ZO (word zrg, word idl)	
Parameter	zrg	layer select bit[15:12] layer select for plane 3 (topmost) bit[11: 8] layer select for plane 2 bit[7: 4] layer select for plane 1 bit[3: 0] layer select for plane 0
	idl	display enable/disable flag bit[3] plane3 bit[2] plane2 bit[1] plane1 bit[0] plane0
Return value	0 - no error codes	
Description	Setting the planes with layer number and enable/disable the planes to be displayed	

GPU_MDR_ZOP [Assign layer to plane]

Format	word GDC_MDR_ZOP (word plane, word layer)	
Parameter	plane layer	plane select (3-0) layer select
Return value	0 - no error codes	
Description	Assign a layer to a plane	

GPU_MDR_ZOD [Enable the plane]

Format	word GDC_MDR_ZOD (word plane, byte enable)	
Parameter	plane enable	plane select (3-0) enable/disable the plane
Return value	0 - no error codes	
Description	Enable/disable the plane which assigned with 'GPU_MDR_ZOP'	

Color Control Commands

GPU_MDR_CC [Intermediate color space code]

Format word GPU_MDR_CC (word cspc)

Parameter cspc code number of the pixel format

Code	Col.Depth	Out path
0 :	1 bpp	B/W
1 :	2 bpp	Col4
2 :	4 bpp	Col16
3 :	8 bpp	Col256
6 :	24 bpp	RGB888
specialized codes are :		
4 :	16 bpp	RGB555
5 :	16 bpp	RGB565
Very specialized codes are :		
7 :	16 bpp	YUV422
8 :	24 bpp	YUV444
9 :	2 bpp	RGB111
10 :	6 bpp	RGB222
11 :	9 bpp	RGB333
12 :	12 bpp	RGB444
13 :	18 bpp	RGB666
reserved codes are:		
14-15		0 bpp -/-

Return value 0 - no error codes

Description Setting the color space code (CSC) for the intermediate display stream
 this defines the color depth for the first common merged color, which is called the intermediate color space
 all CSC of all layers that have to be displayed were transferred into this color format;
 further this color format will be transferred to the physically color format in most cases as 1:1
 in a few cases it will be transferred by using a duty ratio modulator or a multiplexer to get the physically color format
 hint: note the intermediate CSC is the resulting color format of all merged layers
 so the CSC of the layer with the highest resolution of color depth should be,
 but only not more then the display may be able to display

GPU_MDR_TC [Intermediate transfer code]

Format word GPU_MDR_TC (byte idl, byte itc)

Parameter idl layer select
 itc transfer code
 0: transfer normal
 1: transfer alternative

Return value 0 - no error codes

Description Setting the transfer code for conversation from layer to intermediate CSC

GPU_MDR_CO [CLUT offset]

Format	word GPU_MDR_CO (word idl, word cl_ofs)	
Parameter	idl	layer select
	cl_ofs	offset in CLUT
Return value	0 - no error codes	
Description	Offset for CLUT setting up the first CLUT entry offset for a layer hint: there exists 256 CLUT entries; for a layer with (for example) 16 colors, you can use color number cl_ofs...cl_ofs+15 as the representing sub-CLUT	

GPU_MDR_PL [Duty ratio modulation]

Format	word GPU_MDR_PL (byte pseudo, byte prdl)	
Parameter	pseudo	index of table entry
	prdl	modulation value
Return value	0 - no error codes	
Description	Set pseudo ratios of duty levels setting up of one from 14 possible modulation values	

GPU_MDR_CLUT [Set CLUT entries]

Format	word GPU_MDR_CLUT (word col_num, dword rgb_col)	
Parameter	col_num	index of the color number
	rgb_col	color value for this color number
Return value	0 - no error codes	
Description	Set CLUT color entry of one color	

GPU_MDR_CLUTGET [Get CLUT entries]

Format	word GPU_MDR_CLUTGET (word col_num)	
Parameter	col_num	index of the color number
Return value	G0CLUT(col_num)	CLUT entry at the place col_num
Description	Get color from CLUT	

GPU_MDR_CLLD [Set CLUT color entry of multiple color]

Format	word GPU_MDR_CLLD (word offset, word col_cnt, dword *rgb_col)	
Parameter	offset	index of the first entry to be changed
	col_cnt	number of entries to be changed
	*rgb_col	pointer from data source array of n color values
Return value	0 - no error codes	
Description	Set CLUT color entry of multiple color setting up multiple of the 256 possible CLUT entries	

GPU_MDR_BCEN [Background color enable]

Format word GPU_MDR_BCEN (byte encl)

Parameter encl enable/disable background color
1: enable
0: disable

Return value 0 - no error codes

Description Enable/disable the background color for layer less display

GPU_MDR_BC [Background color]

Format word GPU_MDR_BC (dword lite)

Parameter lite color value normalized in the intermediate CSC for the display

Return value 0 - no error codes

Description Set background color

GPU_DIR_CC [Physical color space code]

Format word GPU_DIR_CC (word col_code)

Parameter col_code code number of the pixel format

Code	Col.Depth	Out path
0 :	1 bpp	B/W
9 :	2 bpp	RGB111
2 :	4 bpp	Col16
11 :	9 bpp	RGB333
12 :	12 bpp	RGB444
13 :	18 bpp	RGB666
6 :	24 bpp	RGB888

other codes are not useable:

Return value 0 - no error codes

Description Setting the physical color space code (CSC)

GPU_DIR_CCL [Limits of color key]

Format word GPU_DIR_CCL (dword llck, dword ulck)

Parameter llck lower limits of color key
ulck upper limits of color key

Return value 0 - no error codes

Description Setting the limits of the color key

Chip Operation

GDC_CU_INITi [Clock unit control information]

Format	word GDC_CU_INITi (void)
Parameter	None
Return value	0 – no error codes
Description	The clock unit (CU) provides all necessary clocks to graphic display controller clock unit register should be configured before all other graphic controller setup information

GDC_PWR_Up [Switch modules on]

Format	word GDC_PWR_Up (tCLKPDRSTR mask)
Parameter	mask
Return value	0 – no error codes
Description	Switch graphic controller internal modules on for preparing operation

GDC_PWR_Dwn [Switch modules off]

Format	word GDC_PWR_Dwn (tCLKPDRSTR mask)
Parameter	mask
Return value	0 – no error codes
Description	Switch graphic controller internal modules off for power saving

GDC_PWR_Chk [Check module status]

Format	word GDC_PWR_Chk (tCLKPDRSTR mask)
Parameter	mask
Return value	CLKPDR clock power down register
Description	Check graphic controller internal module status

Anti aliasing Commands

PXP_AAF_THE [Enable anti aliasing filter 2x2]

Format	word PXP_AAF_THE (byte ena)	
Parameter	ena	anti aliasing 2x2 enable flag 0: AAF 2x2 = off 1: AAF 2x2 = on
Return value	0 - no error codes	
Description	Set anti aliasing filter 2x2 active or inactive	

PXP_AAF_THO [Enable anti aliasing filter 4x4]

Format	word PXP_AAF_THE (byte opti)	
Parameter	opti	anti aliasing 4x4 enable flag 0: AAF 4x4 = off 3: AAF 4x4 = on
Return value	0 - no error codes	
Description	Set anti aliasing filter 4x4 active or inactive	

PXP_AAF_THR [Boundary values for red]

Format	word PXP_AAF_THR (word low, word high)	
Parameter	low high	lower bound of threshold value upper bound of threshold value
Return value	0 - no error codes	
Description	setting the boundary values for one separate color channel, which is red	

PXP_AAF_THG [Boundary values for green]

Format	word PXP_AAF_THG (word low, word high)	
Parameter	low high	lower bound of threshold value upper bound of threshold value
Return value	0 - no error codes	
Description	setting the boundary values for one separate color channel, which is green	

PXP_AAF_THB [Boundary values for blue]

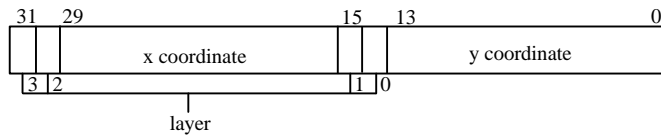
Format	word PXP_AAF_THB (word low, word high)	
Parameter	low high	lower bound of threshold value upper bound of threshold value
Return value	0 - no error codes	
Description	setting the boundary values for one separate color channel, which is blue	

7. Midlevel Drawing Library Function

The midlevel functions are in the module `gdc_drw.c` and use the commands of “API”.

`gdc_lib_pform` [Generate pixel coordinates]

Format	<code>dword gdc_lib_pform (int layer, int x, int y)</code>	
Parameter	<code>layer</code>	layer select
	<code>x</code>	pixel coordinate x
	<code>y</code>	pixel coordinate y
Return value	<code>PIX</code>	pixel and layer information for IFIFO
Description	Setting layer number and pixel coordinates in dword the data format is show in the figure below	



`DrawRect` [Draw rectangular]

Format	<code>void DrawRect (int layer, int xs, int ys, int xe, int ye, long color)</code>	
Parameter	<code>layer</code>	layer select
	<code>xs</code>	start position x
	<code>ys</code>	start position y
	<code>xe</code>	stop position x
	<code>ye</code>	stop position y
	<code>color</code>	rectangular color
Return value	None	
Description	Draws a filled rectangular plane with upper left corner <code>xa, ya</code> and lower right corner <code>xe, ye</code>	

`DrawFrame` [Draw frame]

Format	<code>void DrawFrame (int layer, int xs, int ys, int xe, int ye, long color)</code>	
Parameter	<code>layer</code>	layer select
	<code>xs</code>	start position x
	<code>ys</code>	start position y
	<code>xe</code>	stop position x
	<code>ye</code>	stop position y
	<code>color</code>	frame color
Return value	None	
Description	Draws a rectangular frame with upper left corner <code>x1, y1</code> and lower right corner <code>x2, y2</code>	

DrawLine [Draw line]

Format void DrawLine (int layer, int xs, int ys, int xe, int ye, long color)

Parameter	layer	layer select
	xs	start position x
	ys	start position y
	xe	stop position x
	ye	stop position y
	color	line color

Return value None

Description Draws a line from point xa, ya to point xe, ye

DrawPoint [Draw point]

Format void DrawPoint (int layer, int xs, int ys, long color)

Parameter	layer	layer select
	xs	x position
	ys	y position
	color	point color

Return value None

Description Draws a single point

out_long [Character conversion]

Format void out_long (unsigned long x, int d, char *dl)

Parameter	x	to formatting numerical value
	d	sum of output signs
	*dl	array with character representation

Return value None

Description Conversion numerical values in character representation

DrawChar [Draw character]

Format void DrawChar (int layer, int xs, int ys, char d, long color)

Parameter	layer	layer select
	xs	x position
	ys	y position
	d	character
	color	character color

Return value None

Description Draws a character

DrawText [Draw text]

Format void DrawChar (int layer, int xs, int ys, char *s, long color)

Parameter	layer	layer select
	xs	x position
	ys	y position
	*s	string array
	color	character color

Return value None

Description Draws a string (use the function DrawChar)

PlotChar [Draw character vertical]

Format void PlotChar (int layer, int xs, int ys, char d, long color)

Parameter	layer	layer select
	xs	x position
	ys	y position
	d	character
	color	character color

Return value None

Description Draws a character vertical

PlotText [Draw text vertical]

Format void PlotText (int layer, int xs, int ys, char *s, long color)

Parameter	layer	layer select
	xs	x position
	ys	y position
	*s	string array
	color	character color

Return value None

Description Draws a string vertical (use the function PlotChar)

MemCopy [Copy rectangular]

Format void MemCopy (int lay_src, int xs, int ys, int xe, int ye, int lay_tgt, int xt, int yt)

Parameter	lay_src	source layer select
	xs	source layer start position x
	ys	source layer start position y
	xe	source layer stop position x
	ye	source layer stop position y
	lay_tgt	target layer select
	xt	target layer start position x
	yt	target layer start position y

Return value None

Description Memory copy with upper left corner x1, y1 and lower right corner x2, y2 from source layer to upper left corner x3, y3 target layer

DrawPicture [Draw picture]

Format void DrawPicture (int layer, int xs, int ys, int xw, int yh, long *picadr, long size)

Parameter	layer	layer select
	xs	start position x
	ys	start position y
	xw	picture width
	yh	picture height
	*picadr	array of pixel coordinates (IFIFO)
	size	number of pixel coordinates

Return value None

Description Put bit-map without ignore color
store bit-map data in VRAM; start at x0, y0, stop at x0+xw, y0+h

DrawPictMask [Draw picture with ignore color]

Format void DrawPictMask (int layer, int xs, int ys, int xw, int yh, long *picadr, long size, long xcolor)

Parameter	layer	layer select
	xs	start position x
	ys	start position y
	xw	picture width
	yh	picture height
	*picadr	array of pixel coordinates (IFIFO)
	size	number of pixel coordinates
	xcolor	ignore color

Return value None

Description Put bit-map with color wish is to be ignored
store bit-map data in VRAM; start at x0, y0, stop at x0+xw, y0+h

Draw8Pix [Draw 8 pixels]

Format void Draw8Pix (int layer, int xs, int ys, int xr, int yr, long color)

Parameter	layer	layer select
	xs	start position x
	ys	start position y
	xr	radius x
	yr	radius y
	color	pixel color

Return value None

Description Draws 8 points

DrawCircle [Draw circle]

Format void DrawCircle (int layer, int xs, int ys, int r, long color)

Parameter	layer	layer select
	xs	circle-center x
	ys	circle-center y
	r	circle radius
	color	circle color

Return value None

Description Draws circle with algorithm (use the function Draw8Pix)

GetPix [Get pixel color]

Format	long GetPix (int layer, int xs, int ys)	
Parameter	layer	layer select
	xs	pixel coordinate x
	ys	pixel coordinate y
Return value	PixCol	pixel color from OFIFO
Description	Load pixel data from VRAM in variable PixCol	

SetAAFLevel [Boundary values for AAF]

Format	void SetAAFLevel (int RL, int RH, int GL, int GH, int BL, int BH, int ena)	
Parameter	RL	lower boundary values for red
	RH	upper boundary values for red
	GL	lower boundary values for green
	GH	upper boundary values for green
	BL	lower boundary values for blue
	BH	upper boundary values for blue
	ena	AAF enable flag
Return value	None	
Description	Setting the boundary values for all three colors RGB	

SetLayDomain [Set layer domain]

Format	void SetLayDomain (int layer, long adr, int dxw)	
Parameter	layer	layer select
	adr	physical address of the first pixel for this layer domain
	dxw	domain width in pixels
Return value	None	
Description	Setting the physical start address in the VRAM and the domain size of a layer	

SetLayWindow [Set layer window]

Format	void SetLayWindow (int layer, int xs, int ys, int xe, int ye, int x, int y)	
Parameter	layer	layer select
	xs	left position in the display
	ys	upper position in the display
	xe	right position in the display
	ye	lower position in the display
	x	left position in the domain
	y	upper position in the domain
Return value	None	
Description	Setting the offset for a window in the display, the size of window and the first pixel position of a window in its domain	

SetLayCForm [Set layer color format]

Format	void SetLayCForm (int layer, int vCSC, int offlut, int dbling, int acnv)	
Parameter	layer	layer select
	vCSC	color space code for a layer
	offlut	line doubling flag
	dbling	transfer code
	acnv	offset in CLUT
Return value	None	
Description	Defines the color depth for a domain, set line doubling flag, transfer code for conversation and The first CLUT entry offset for a layer	

SetLayTrans [Set layer transparent color]

Format	void SetLayTrans (int layer, long color, int ena)	
Parameter	layer	layer select
	color	transparent color
	ena	transparent enable flag
Return value	None	
Description	Each pixel in this layer which carries this color will go transparent, if ena =1	

SetLayBlink [Set layer blink color]

Format	void SetLayBlink (int layer, long color1, long color2, int rate1, int rate2, int ena)	
Parameter	layer	layer select
	color1	blink color
	color2	alternative blink color
	rate1	blink rate for blink color
	rate2	blink rate for alternative blink color
	ena	blink enable flag
Return value	None	
Description	Set blink color, alternative blink color and blink rate for the corresponding color	