D/AVE 2D

Data Sheet

D/AVE 2D-TL, D/AVE 2D-TS

Author	:	Christian Sehnke
Document Number	:	TD-240201SH DS
Security	:	Public
Version	:	3.0
Date	:	2020-03-12
Status	:	Approved
Released From	:	TES Electronic Solutions GmbH
Authorization	:	Christian Sehnke
Copy List	:	Look ‘n’ Feel Team Hamburg
Commentators	:	Look ‘n’ Feel Team Hamburg
Reviewers	:	Bastian Zuehlke, Anton Schwab, Christoph ten Haaf, Christian Sehnke, Andy Strzeletz, Michael Reusch

PRELIMINARY REMARK:

TABLE OF CONTENTS

CHANGE HISTORY

Date	Person	Version	Reason
14.08.06	Bastian Zuehlke	0.1	First version
21.08.06	Christian Sehnke	0.2	Minor corrections, added memory and gate count information
24.08.06	Christian Sehnke	0.3	Corrections after review
24.08.06	Bastian Zuehlke	1.0	Authorization, set status to 'Approved'
31.10.06	Christian Sehnke	1.1	Corrected memory sizes
16.11.06	Bastian Zuehlke	1.2	Minor changes
19.01.07	Christian Sehnke	1.3	New toplevel interface + AMBA interface
26.02.07	Bastian Zuehlke	1.4	Set status to 'Approved'
19.06.07	Christian Sehnke	1.5	Added BUS_ERROR interrupt
01.10.07	Andreas Schibilla	1.6	Minor changes in AHB feature description
11.02.08	Andreas Schibilla	1.7	Merge files (added ASIC synthesis results)
06.03.08	Christian Sehnke	1.8	Clarification of fbcache memory type
29.04.08	Christian Sehnke	1.9	Added chapter "Limitations"
09.06.08	Michael Reusch	2.0	Added RLE, CLUT256, subbyte formats and color keying
16.07.08	Andy Strzeletz	2.0.1	Added new performance counter events
12.03.09	Michael Reusch	2.1	corrected limitations
28.10.09	Michael Reusch	2.2	added alpha write to image 3.5
23.03.10	Michael Reusch	2.3	new Alpha Channel Blending
28.01.11	Christian Sehnke	2.4	clarified chapter 'Framebuffer limitations'
10.02.11	Christian Sehnke	2.5	added chapter 'Driver Memory Footprint'
19.07.11	Michael Reusch	2.6	added alpha4,2,1 texture formats
20.10.11	Christian Sehnke	2.7	added synthesis results for Xilinx
13.04.12	Christian Sehnke	2.8	detailed the differences between D/AVE 2D-TL and -TS
01.08.12	Christian Sehnke	2.9	added note in chapter 6.3, that register writes are not allowed while the DLR is active
12.03.20	Michael Reusch	3.0	Added AXI, removed Avalon, CLUT single ported Ram

1. INTRODUCTION

The D/AVE 2D is a VHDL RTL IP developed for sophisticated vector based graphic applications. D/AVE perfectly fits into the requirements of embedded systems.

Feature highlights:

- Low resource usage:
  - Low gate count
  - Small memory bandwidth consumption
- Offers a wide range of graphical primitives, e.g.
  - Lines
  - Triangles
  - Quadrangles
  - Circles
- Supports anti-aliasing and sub pixel accurate rendering

The D/AVE 2D core is available in two different versions. A ‘standard’ version named D/AVE 2D-TS and ‘light’ version named D/AVE 2D-TL. The major differences between both cores are:

- D/AVE 2D-TL needs 4 cycles per pixel while D/AVE 2D-TS needs only 1 cycle (except in case of texture filtering). Since this peak performance can not be fully reached in most use cases anyway due to bus bandwidth and latency limitations, the performance difference from application point of view is in the range of a factor of 2-3 instead of 4.
- D/AVE 2D-TL features simpler caches and does not provide prefetching of reads on the framebuffer interface.
- Sub-byte texture formats (1,2 and 4bpp) as well as CLUT formats are not available in the current version of D/AVE 2D-TL.
- Color-keying is not available in the current version of D/AVE 2D-TL.
- RLE textures are not available in the current version of D/AVE 2D-TL.
- Alpha channel blending in the blend unit is not available in the current version of D/AVE 2D-TL.

Both cores are available as ALTERA SOPC-Builder™ ready component using an Avalon bus interface, tested on the device families Cyclone II and Stratix I.
An AMBA AHB/APB bus interface is also available.

1.1 Definitions and Abbreviations

ARGB	:	Alpha, Red, Green, Blue (color components)
BLIT	:	Block Image Transfer
bpp	:	Byte per pixel
C/S	:	Configuration/Status Interface
DLR	:	Display List Reader
FB	:	Framebuffer
MBA	:	Master Bus Adaptor
SBA	:	Slave Bus Adaptor
U/V	:	Sub coordinates inside texture
Texel	:	Texture pixel
RLE	:	Run Length Encoding
a8	:	8 bit alpha
b32	:	32 bit pattern
c24	:	24 bit color 24
ca32	:	32 bit color and alpha
f16	:	16 bit fixed point fractional part
fx32	:	32 bit fixed point
i16	:	16 bit integer

2. SIGNAL DESCRIPTION

The D/AVE IP core component is a single clock-domain pipeline structure having four external interfaces:

- a single token slave interface for configuration writes and status reads
- a read-only master interface for display list reads
- a read-only master interface for texel reads
- a read/write interface for framebuffer pixel data

Internal caches are used to transform the single-token accesses for texel and pixel data to burst accesses for better performance on the bus.

2.1 Block Diagram

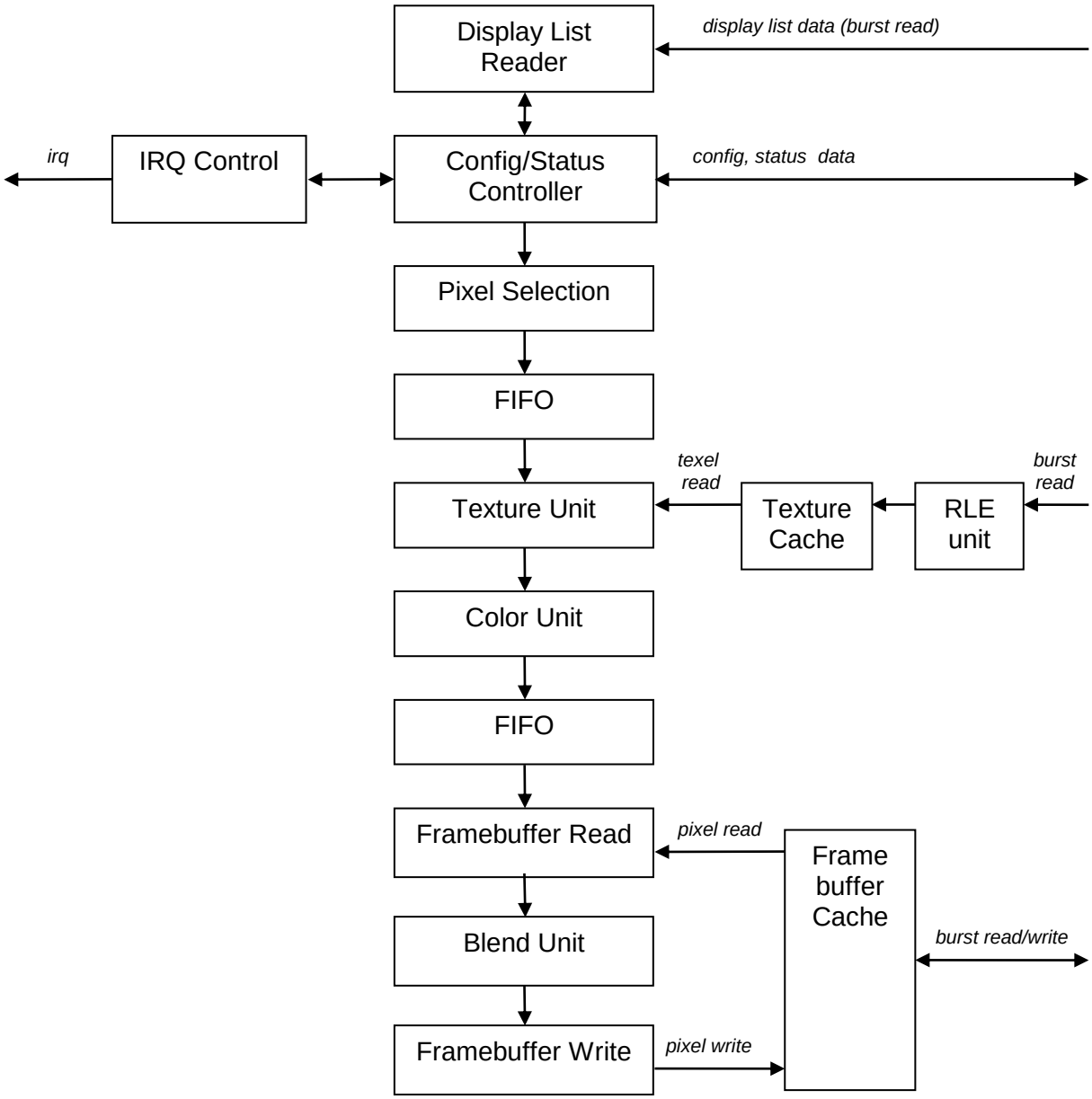


Image 2-1 Block diagram

2.2 Generic Toplevel

The generic D/AVE toplevel offers generalized bus interfaces that can be interfaced to any bus interface using special bus adaptor modules.

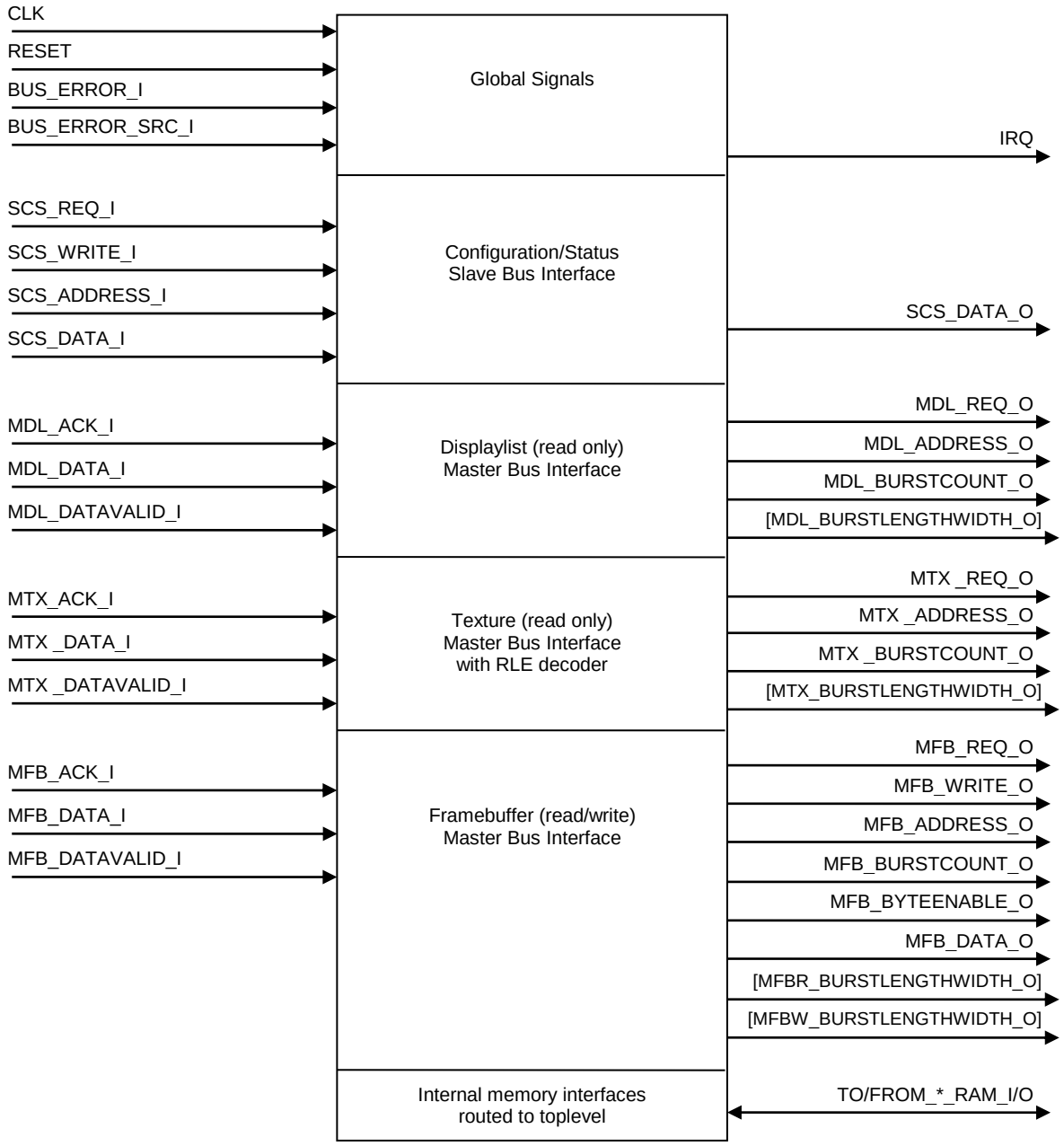


Image 2-2 Generic Toplevel

2.2.1 General Signals

CLK	*Single clock, active edge is rising edge*
RESET	*Asynchronous reset, active high*
IRQ	*Interrupt request, active high*
BUS_ERROR_I	*Bus error signal that can generate an interrupt (coming from bus adaptor modules)*
BUS_ERROR_SRC_I	*Indication which bus adaptor caused an error*

Table 2-1 Toplevel general signals

2.2.2 Configuration/Status Slave Bus Interface

SCS_REQ_I	*Read request if SCS_WRITE_I is low, write request otherwise.*
SCS_WRITE_I	*Request is write request if high. Valid when SCS_REQ_I is high.*
SCS_ADDRESS_I [5..0]	*Configuration register address. Valid when SCS_REQ_I is high.*
SCS_DATA_I [31..0]	*Configuration data in. Valid when SCS_REQ_I is high.*
SCS_DATA_O [31..0]	*Status data out. Valid one clock cycle after SCS_REQ_I was high.*

Table 2-2 Toplevel configuration/status interface signals

2.2.3 Displaylist Master Bus Interface

MDL_REQ_O	*Read request.*
MDL_ACK_I	*Acknowledge for MDL_REQ_O.*
MDL_VALID_I	*Data from bus is valid.*
MDL_ADDRESS_O [default: 31..0]	*Read address of display list. Valid when MDL_REQ_O is high.*
MDL_BURSTCOUNT_O[default: 4..0]	*Number of data words requested minus one. Starting at MDL_ADDRESS_O. Valid when MFB_REQ_O is high.*
MDL_DATA_I [default: 31.. 0]	*Display list data. Valid when MDL_VALID_I is high.*
MDL_BURSTLENGTHWIDTH_O[2:0]	*Optional output for burst length limit*

Table 2-3 Toplevel Displaylist interface signals

2.2.4 Texture Master Bus Interface

MTX_REQ_O	*Read request.*
MTX_ACK_I	*Acknowledge for MTX_REQ_O.*
MTX_VALID_I	*Data from bus is valid.*
MTX_ADDRESS_O [default: 31..0]	*Read address of texel. Valid when MTX_REQ_O is high.*
MTX_BURSTCOUNT_O[default: 4..0]	*Number of data words requested minus one. Starting at MTX_ADDRESS_O. Valid when MFB_REQ_O is high.*
MTX_DATA_I [default: 31.. 0]	*Texel data. Valid when MTX_VALID_I is high.*
MTX_BURSTLENGTHWIDTH_O[2:0]	*Optional output for burst length limit*

Table 2-4 Toplevel Texture interface signals

2.2.5 RLE unit Master Bus Interface

This interface is shared with the Texture Master Bus Interface

RLE_REQ_O	*Read request.*
RLE_ACK_I	*Acknowledge for RLE_REQ_O.*
RLE_VALID_I	*Data from bus is valid.*
RLE_ADDRESS_O [default: 31..0]	*Read address of RLE code. Valid when RLE_REQ_O is high.*
RLE_BURSTCOUNT_O[default: 4..0]	*Number of data words requested minus one. Starting at RLE_ADDRESS_O. Valid when RLE_REQ_O is high.*
RLE_DATA_I [default: 31.. 0]	*RLE code data. Valid when RLE_VALID_I is high.*

Table 2-5 Toplevel RLE unit interface signals

2.2.6 Framebuffer Master Bus Interface

MFB_REQ_O	Read request if MFB_REQ_O is low, write request otherwise.
MFB_WRITE_O	*Request is write request if high. Valid when MFB_REQ_O is high.*
MFB_ACK_I	*Acknowledge for MFB_REQ_O.*
MFB_VALID_I	*Pixel data input is valid.*
MFB_ADDRESS_O [default: 31..0]	*Read address of display list. Valid when MFB_REQ_O is high.*
MFB_BURSTCOUNT_O[default: 4..0]	*Number of data words requested minus one. Starting at MFB_ADDRESS_O. Valid when MFB_REQ_O is high.*
MFB_BYTEENABLE_O[default: 3..0]	*Byte enable. All bits high during read, only used for writes.*
MFB_DATA_I [default: 31.. 0]	*Pixel data input. Valid when MFB_VALID_I is high.*
MFB_DATA_O [default: 31.. 0]	*Pixel data output. Valid when MFB_WRITE_O is high.*
MFBR_BURSTLENGTHWIDTH_O[2:0]	*Optional output for read burst length limit*
MFBW_BURSTLENGTHWIDTH_O[2:0]	*Optional output for write burst length limit*

Table 2-6 Toplevel Framebuffer interface signals

2.2.7 Available bus adaptors

TES has available bus adaptors for the ARM AMBA busses (AMBA APB and AXI4lite for slave interface, AMBA AHB and AXI for master interfaces). Other bus interfaces can be implemented by the customer or provided by TES as design service.

To support busses, which don't offer a byte-enable functionality for writes, TES has available a special implementation of the framebuffer cache which does not use this features, but does a read/modify/write instead.

To reduce the number of external interfaces, a 3-to-1 arbiter is also available, that reduces the number of external master bus interfaces to a single read/write interface for all supported bus architectures.

2.3 AMBA AHB/APB Toplevel

TES is offering a configurable ARMAMBA2 AHB/APB interface for D/AVE 2D.

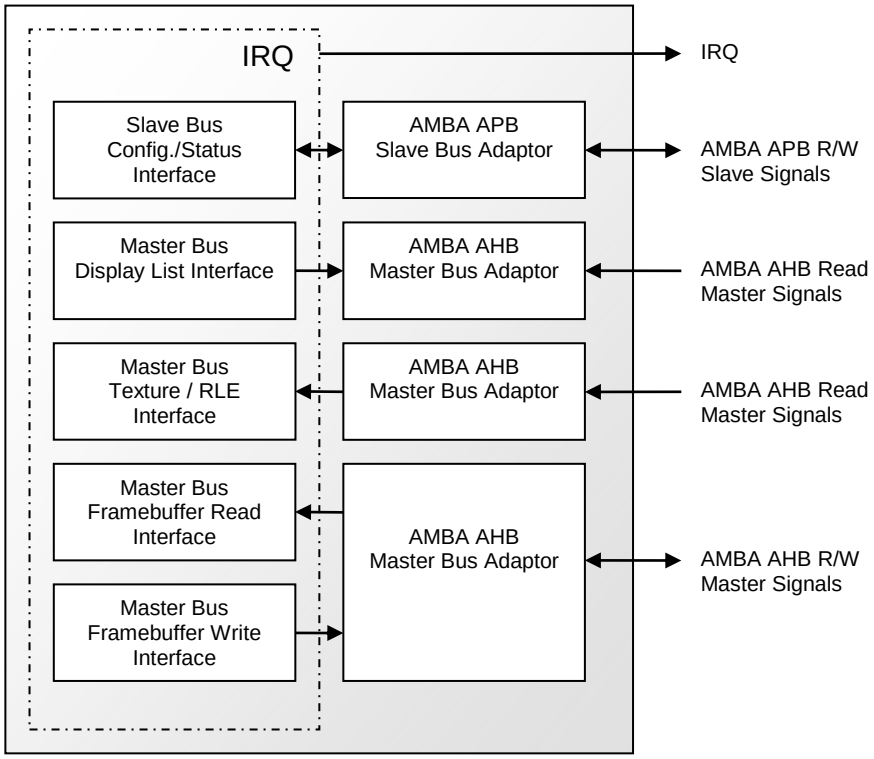


Image 2-3 AMBA AHB Toplevel

Configurable features are:

- Three vs. single AHB master interface
- Data width of AHB interface
- Support locked transfers
- Supported transfer sizes:
  Cache on: WORD (depending on bus width)
  Cache off: WORD, HALFWORD, BYTE (depending on frame buffer format)

The AHB master bus adaptor implements the AHB interface with the following constraints:

- No BUSY cycles generated
- No wrapping bursts or incrementing bursts of unspecified length performed
- No SPLIT and RETRY responses supported, ERROR responses generate interrupts
- Narrow writes on wide bus (in case the cache is disabled)
- Use of locking mode can be enabled.

The complete AMBA 2 specific signal description is not part of this document.

2.4 AMBA AXI Toplevel

TES is offering a configurable ARM AMBA 3 AXI and APB or AXI4Lite interface for D/AVE 2D.

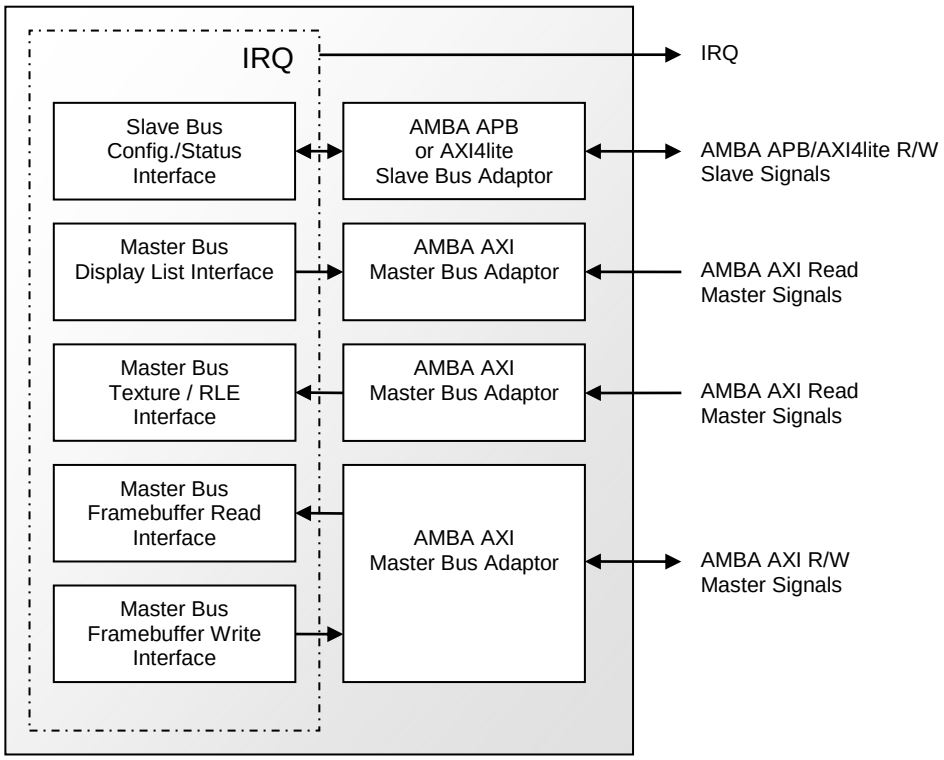


Image 2-4 AMBA AXI Toplevel

Configurable features are:

- Three vs. two or single AXI master interface
- Data width of AXI interface
- Supported transfer sizes:
  Cache on: WORD (depending on bus width)
  Cache off: WORD, HALFWORD, BYTE (depending on frame buffer format)

The complete AMBA AXI specific signal description is not part of this document.

3. SYSTEM ARCHITECTURE

The D/AVE 2D IP has four major interfaces:

- Configuration/Status Interface
  - Used to configure D/AVE and to get current status
  - Interrupt Request
  - Used for CPU and D/AVE synchronization
- Display List Reader Interface
  - Read the display list out of the memory
- Texture Interface
  - Read texture pixels out of the memory
- Framebuffer Interface
  - Read/write framebuffer pixels from/to the memory

3.1 High-Level System Architecture

In common systems, D/AVE 2D is connected via a high performance bus to system memory or to dedicated memory. The configuration itself is normally done via a peripheral bus. For synchronization between CPU and D/AVE 2D, we highly recommend using interrupts, but polling is still an option. The next image shows a typical system architecture.

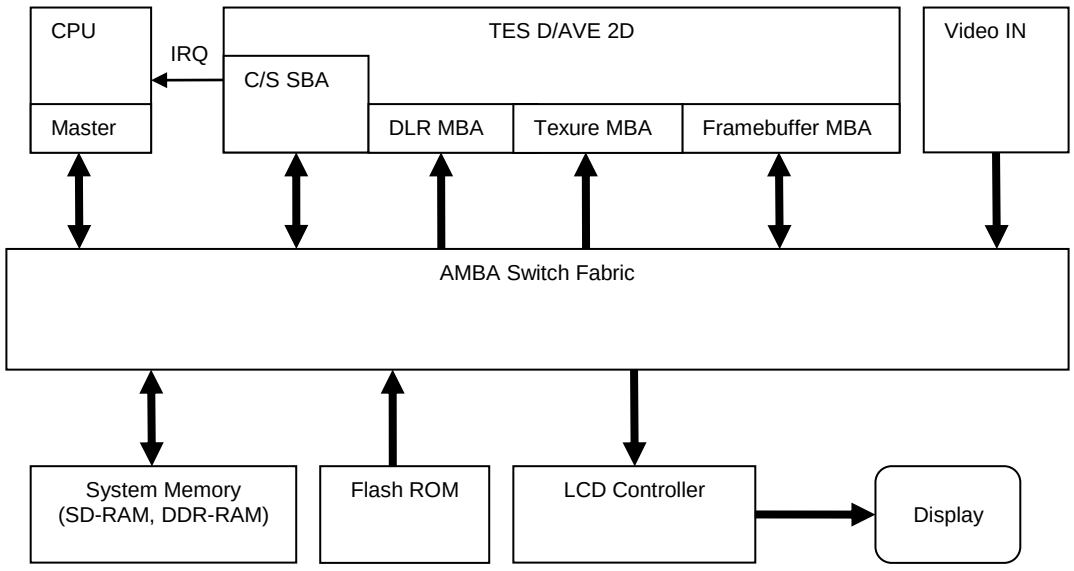


Image 3-1 Typical high level system architecture

3.2 Configuration/Status Interface

This interface is used to read and write the D/AVE 2D registers. A register set description can be found in chapter 4.

3.2.1 Interrupt Request

On an interrupt request, the IRQ signal is asserted until the CPU resets it by clearing the respective interrupt control register bits. A detailed description of the interrupt handling register can be found in chapter 4.1.4.

3.3 Display List Reader Interface

The Display list reader reads a memory block containing instructions on how to set the D/AVE 2D control registers and does these control register writes accordingly.

Using a display list (also called 'render buffer') is a way to allow a fully asynchronous operation of CPU and D/AVE 2D. It also gives the best possible system performance. The following image illustrates this. The display list format is described in chapter 5.

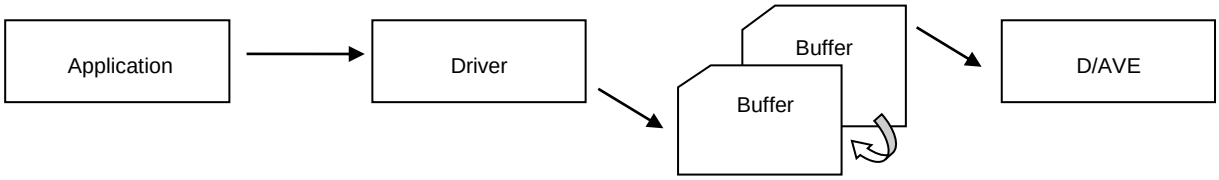


Image 3-2 Display List mechanism

3.4 Texture Interface

For BLIT and texture operations this interface provides D/AVE 2D with the needed pixel data. Pixel formats widths 8, 16 and 32 bits are supported as well as run length encoded textures.
Pixel formats with 4, 2 and 1 bits per pixel are supported with the restriction that these formats must be byte packed.

3.5 Framebuffer Interface

This interface allows D/AVE to read and write to/from the frame buffer. Pixel format widths 8, 16 and 32 bits are supported.

3.6 2D Render Feature List

This chapter gives a brief overview of all available render features of the D/AVE IP. To get a more detailed description please refer to chapter 3.7.

- Supported graphic primitives
  - BLIT
    - Direct and stretch
  - Box
  - Circle
    - Filled or empty (ring)
  - Convex Polygon
  - Line
    - Supported caps:
      - Butt
      - Round
      - Square
    - Supported line joins:
      - Bevel
      - Miter
      - Round
    - Supporting different start and end widths
  - Quad
  - Triangle
  - Triangle Fan
  - Triangle List
  - Triangle Stripe
  - Wedge
    - Filled or empty

- Supported attributes for graphic primitives
  - Anti-Aliasing
  - Blend Modes
  - Color
  - Edge Blur
  - Linear Alpha gradient
  - Pattern
  - Texture
    - U/V Clamp, Repeat support
    - No-, Linear-, Bilinear- Filtering support
    - RLE

- Supported color formats
  - Input
    - ARGB8888, RGB565, ARGB4444, ARGB1555, ALPHA8, AI44, RGBA8888, RGBA4444, RGBA5551, I8, I4, I2, I1, ALPHA4, ALPHA2, ALPHA1
  - Output
    - ARGB8888, RGB565, ARGB4444, ALPHA8, RGBA8888, RGBA4444

3.7 2D Render Capabilities

The D/AVE IP offers a wide range of render capabilities, which are explained in this chapter.

3.7.1 Anti-Aliasing

D/AVE offers primitive anti-aliasing. In case of a triangle, the anti-aliasing can be turned on or off for every single edge. This mechanism is useful to render polygons with correct outline anti-aliasing.

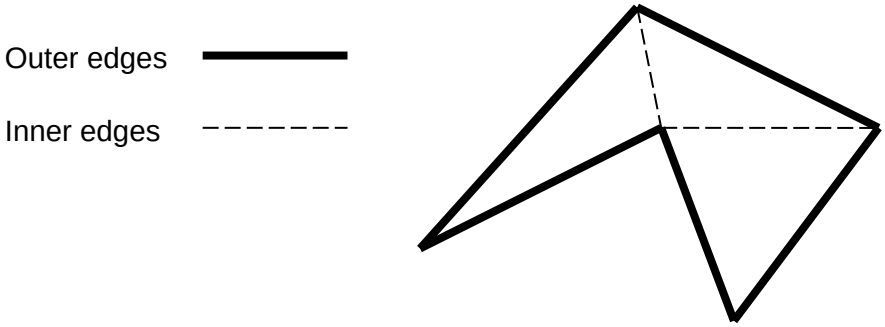


Image 3-3 Polygon subdivided into triangles

3.7.2 Basic Graphic Primitives

The D/AVE IP offers a wide range of basic graphic primitives. These are geometries which D/AVE can render in one pass. One pass supported primitives are:

- BLIT
- Box
- Circle
- Line
- Quad
- Triangle
- Wedge

3.7.3 Blend Modes

D/AVE offers a wide range of color calculation operations. The color calculation is done in the Color Unit and Blend Unit.

3.7.3.1 Color Unit

The following operations can be applied to each color channel (a,r,g,b) individually :

- REPLACE c
- COPY x
- INVERT 1 - x
- MULTIPLY x * c
- INVMULTIPLY (1-x) * c
- BLEND x * c + (1-x) * d

x is the color channel value (a, r, g or b), c (color 1) and d (color 2) are user specified constants.

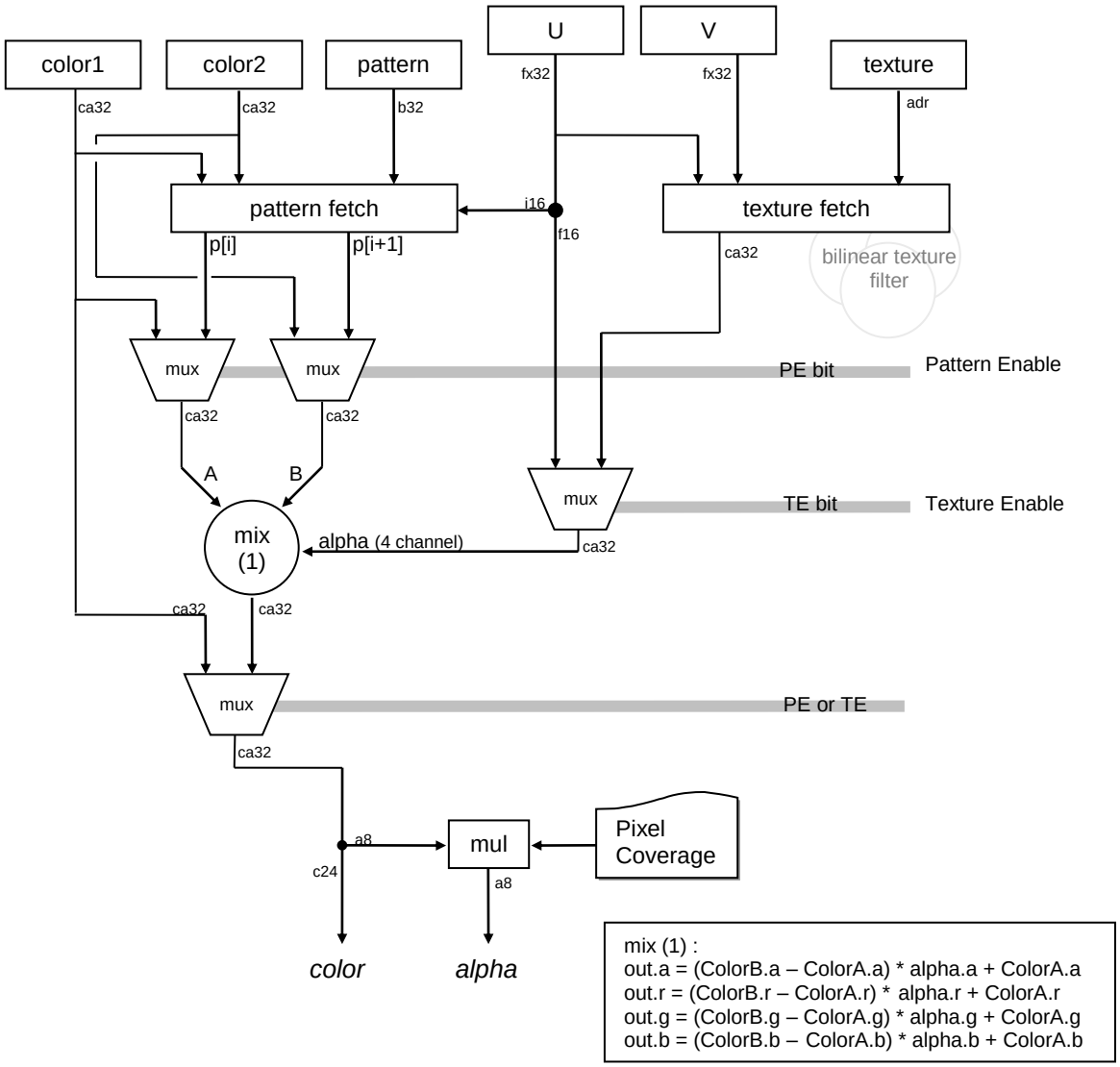


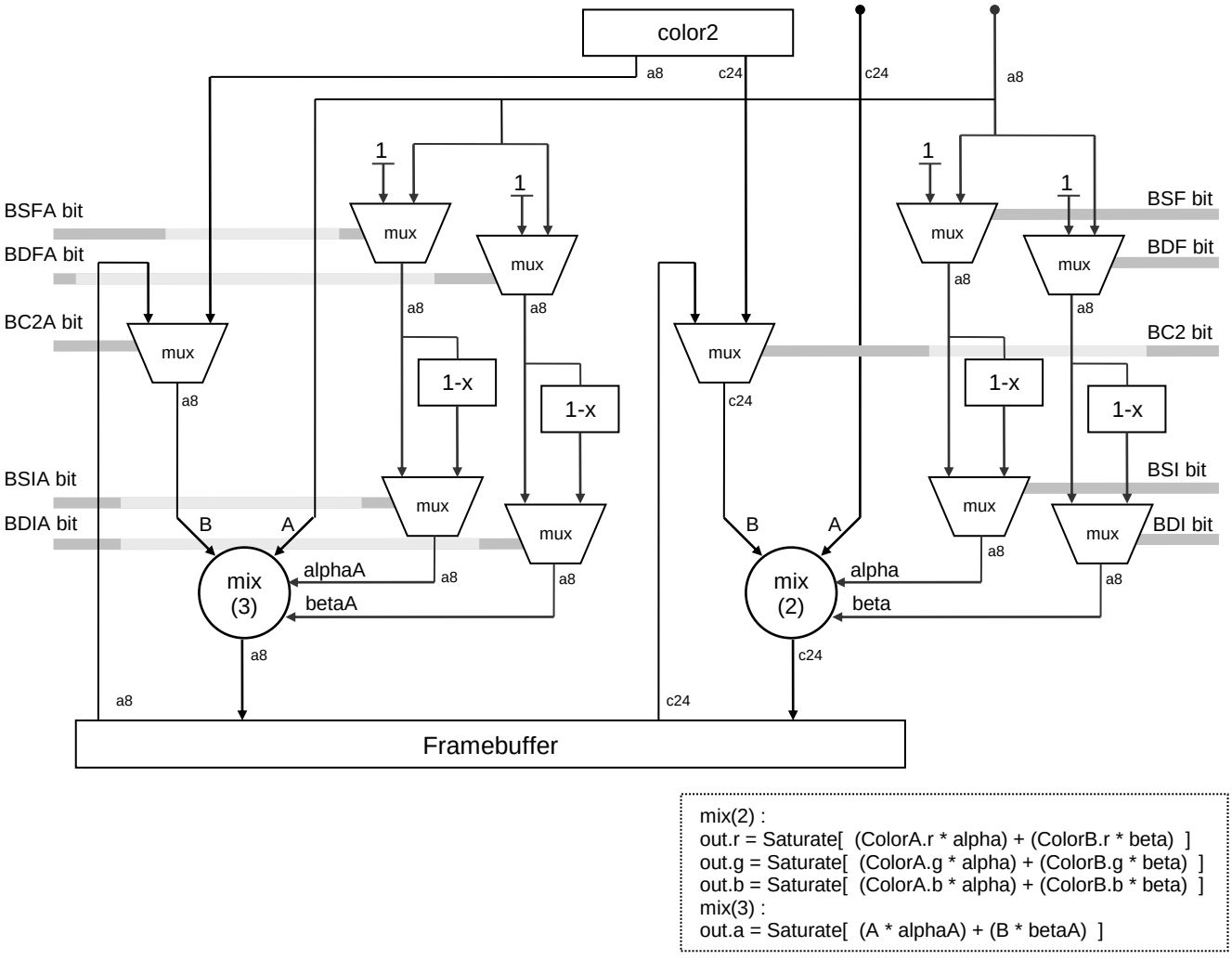
Image 3-4 Color Unit Block Diagram

3.7.3.2 Blend Unit

The following blend modes are directly supported as source and destination factors separately for color and alpha channels:

- ZERO
- ONE
- ALPHA
- ONE_MINUS_ALPHA

Instead of the color/alpha value read back from the framebuffer, a constant (color2) can be used as destination color/alpha for the blend unit.



mix(2) :
out.r = Saturate[(ColorA.r * alpha) + (ColorB.r * beta)]
out.g = Saturate[(ColorA.g * alpha) + (ColorB.g * beta)]
out.b = Saturate[(ColorA.b * alpha) + (ColorB.b * beta)]
mix(3) :
out.a = Saturate[(A * alphaA) + (B * betaA)]

Image 3-5 Blend Unit Block Diagram

3.7.4 Texture

D/AVE supports U/V clamp and U/V repeat. Also filtering for both axes separately is supported. The next table demonstrates these different options.

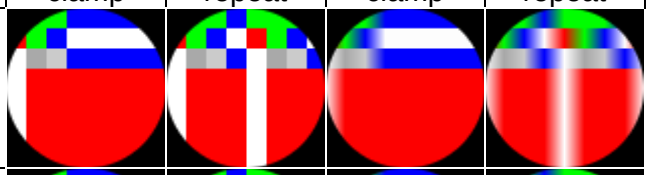
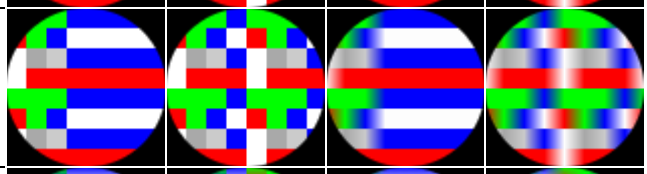
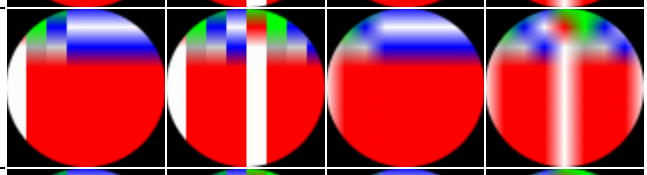
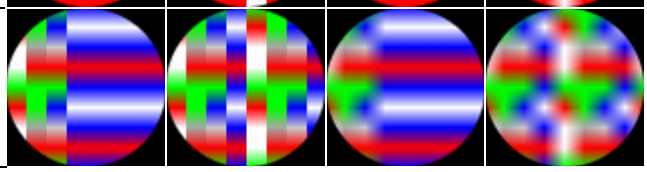
			x			
			no filter		filter	
			clamp	repeat	clamp	repeat
y	no filter	clamp				
		repeat				
	filter	clamp				
		repeat				

Table 3-1 Texture examples

4. REGISTER DESCRIPTION

Not all registers are described in full detail here as the interface to D/AVE 2D is considered to be the SW driver and not the register interface itself.

4.1 Control Registers

Write:

D2_CONTROL	-	Geometry control register
D2_CONTROL2	-	Surface control register
D2_IRQCTL	-	Interrupt control register
D2_CACHECTL	-	Cache control register

Read:

D2_STATUS	-	Status control register
D2_HWREVISION	-	Hardware version and feature set ID

4.1.1 D2_CONTROL

Offset:	0x00
Default Value	0x00000000
Access	Write Only
Description	This register controls the pixel enumeration and selection units, deciding which pixels are part of the geometry. A detailed description would not be understandable without deep knowledge of the internal rendering mechanism of D/AVE 2D and is therefore not provided.

Table 4-1 Register: D2_CONTROL

4.1.2 D2_CONTROL2

Offset:	0x01	
Default Value	0x00000000	
Access	Write Only	
Description	This register controls the colorization and blending units, deciding what color a pixel should have.	
Name	**Bits**	**Description**
D2C_RLE_PIXEL_WIDTH	31:30	Width of a texel for the RLE unit: 0..3 = 1-4 bytes per texel.
D2C_BDIA	29	If D2C_USE_ ACB = 1: 0 = use blend factor as specified by D2C_BDFA 1 = invert destination blend factor (1-x)
D2C_BSIA	28	If D2C_USE_ ACB = 1: 0 = use blend factor as specified by D2C_BSFA 1 = invert source blend factor (1-x)
D2C_CLUTFORMAT	27	0 = ARGB8888 (default) 1 = RGB565
D2C_COLKEY_ENABLE	26	Enable the color keying unit.
D2C_CLUT_ENABLE	25	Enables the CLUT. If CLUT is not enabled for an

		indexed format, the index will be directly put on the internal R,G,B channels.
D2C_RLE_ENABLE	24	enables the RLE unit.
D2C_WRITEALPHA	23:22	If D2C_USE_ ACB = 0: Set the 'alpha source' for the framebuffer. 00 = use alpha from color2 01 = use source alpha (pixel coverage) 10 = use 0.0 as alpha 11 = use alpha from framebuffer If D2C_USE_ ACB = 1: 00 = BC2A is 1. Use destination alpha from color2 01, 10, 11 = BC2A is 0. Use destination alpha
D2C_WRITEFORMAT	8,21:20	Pixel format of the framebuffer 000 = alpha8 (1 bpp) 001 = rgb565 (2 bpp) 010 = argb8888 (4 bpp) 011 = argb4444 (2 bpp) 110 = rgba8888 (4 bpp) 111 = rgba4444 (2 bpp)
D2C_READFORMAT	5,4,19:18	Pixel format of the texture buffer 0000 = alpha8 (1 bpp) 0001 = rgb565 (2 bpp) 0010 = argb8888 and rgb888 (4 bpp) 0011 = argb4444 and rgb444 (2 bpp) 0100 = argb1555 and rgb555 (2 bpp) 0101 = ai44: 4 bit alpha, 4 bit indexed color (1 bpp) 0110 = rgba8888 (4 bpp) 0111 = rgba4444 (2 bpp) 1000 = rgba5551 (2 bpp) 1001 = i8, 8bit indexed color (1 bpp) 1010 = i4, 4bit indexed color (2 ppb) 1011 = i2, 2bit indexed color (4 ppb) 1100 = i1, 1bit indexed color (8 ppb) 1101 = alpha4 (2ppb) 1110 = alpha2 (4ppb) 1111 = alpha1 (1ppb)
D2C_TEXTUREFILTERY	17	0 = no filtering on texture v axis 1 = linear filtering on texture v axis
D2C_TEXTUREFILTERX	16	0 = no filtering on texture u axis 1 = linear filtering on texture u axis
D2C_TEXTURECLAMPY	15	0 = mask texture v-coordinate 1 = clamp texture v-coordinate
D2C_TEXTURECLAMPX	14	0 = mask texture u-coordinate 1 = clamp texture u-coordinate
D2C_BC2	13	Blend color2 instead of framebuffer pixel 0 = use pixel from framebuffer as destination 1 = use color2 as destination

D2C_BDI	12	0 = use blend factor as specified through D2C_BDF 1 = invert destination blend factor (1-x)
D2C_BSI	11	0 = use blend factor as specified through D2C_BSF 1 = invert source blend factor (1-x)
D2C_BDF	10	0 = use 1.0 as destination blend factor 1 = use alpha as destination blend factor
D2C_BSF	9	0 = use 1.0 as source blend factor 1 = use alpha as source blend factor
D2C_WRITEFORMAT3	8	Bit 3 of the framebuffer buffer format. See D2C_WRITEFORMAT above for description.
D2C_BDFA	7	If D2C_USE_ ACB = 1: 0 = use 1.0 as destination blend factor for alpha channel 1 = use alpha as destination blend factor for alpha channel
D2C_BSFA	6	If D2C_USE_ ACB = 1: 0 = use 1.0 as source blend factor for alpha channel 1 = use alpha as source blend factor for alpha channel
D2C_READFORMAT4	5	Bit 4 of the texture buffer format. See D2C_READFORMAT above for description.
D2C_READFORMAT3	4	Bit 3 of the texture buffer format. See D2C_READFORMAT above for description.
D2C_USE_ACB	3	0 = use WRITEALPHA mode (see D2C_WRITEFORMAT) 1 = use full alpha channel blending mode (see D2C_BSFA, D2C_BDFA, D2C_BSIA, D2C_BDIA, D2C_WRITEFORMAT) see also ALPHACHANNELBLENDING bit in D2_HWREVISION
D2C_PATTERNSOURCEL5	2	Limiter 5 is used as pattern index instead of the default U-Limiter
D2C_TEXTUREENABLE	1	Pixel source is read from texture and used as an alpha to blend between D2_COLOR1 and D2_COLOR2. 0 = disable texture 1 = enable texture
D2C_PATTERNENABLE	0	Pixel source is a pattern color (blend of D2_COLOR1 and D2_COLOR2 depending on D2_PATTERN and pattern index) 0 = disable pattern 1 = enable pattern

Table 4-2 Register: D2_CONTROL2

4.1.3 D2_CONTROL3

Offset:	0x02	
Default Value	0x04040404	
Access	Write Only	
Description	This register sets the burst length limit for the 3 master bus interfaces	
Name	**Bits**	**Description**
BURSTLENGTH_MFBR	2:0	Log2 of the burst length limit for MFB read 0 : 2^0 = single transfer 1 : 2^1 = max burst length 2 2 : 2^2 = max burst length 4 3 : 2^3 = max burst length 8 4 : 2^4 = max burst length 16
BURSTLENGTH_MFBW	10:8	Log2 of the burst length limit for MFB write
BURSTLENGTH_MTX	18:16	Log2 of the burst length limit for MTX read
BURSTLENGTH_MDL	26:24	Log2 of the burst length limit for MDL read

Table 4-3 Register: D2_CONTROL3

4.1.4 D2_IRQCTL

Offset:	0x30	
Default Value	0x00000000	
Access	Write Only	
Description	This register enables interrupts and clears the interrupt status.	
Name	**Bits**	**Description**
D2IRQCTL_CLR_BUS_ERROR	5	0 = leave bus error interrupt untouched 1 = clear bus error interrupt
D2IRQCTL_ENABLE_BUS_ERROR	4	0 = disable bus error interrupt 1 = enable bus error interrupt
D2IRQCTL_CLR_FINISH_DLIST	3	0 = leave displaylist interrupt untouched 1 = clear displaylist interrupt
D2IRQCTL_CLR_FINISH_ENUM	2	0 = leave enumeration interrupt untouched 1 = clear enumeration interrupt
D2IRQCTL_ENABLE_FINISH_DLIST	1	0 = disable displaylist finished interrupt 1 = enable displaylist finished interrupt
D2IRQCTL_ENABLE_FINISH_ENUM	0	0 = disable enumeration finished interrupt 1 = enable enumeration finished interrupt

Table 4-4 Register: D2_IRQCTL

4.1.5 D2_CACHECTL

Offset:	0x31	
Default Value	0x00000000	
Access	Write Only	
Description	Internal caches can be enabled / disabled and flushed using this register.	
Name	**Bits**	**Description**
D2C_CACHECTL_FLUSH_TX	3	0 = do not flush the texture cache 1 = flush the texture cache
D2C_CACHECTL_ENABLE_TX	2	0 = disable the texture cache 1 = enable the texture cache
D2C_CACHECTL_FLUSH_FB	1	0 = do not flush the framebuffer cache 1 = flush the framebuffer cache
D2C_CACHECTL_ENABLE_FB	0	0 = disable the framebuffer cache 1 = enable the framebuffer cache

Table 4-5 Register: D2_CACHECTL

4.1.6 D2_STATUS

Uses the same address offset as **D2_CONTROL** for reading access.

Offset:	0x00
Default Value	0x00000000
Access	Read Only
Description	The current dave status can be polled by reading this register.

Name	Bits	Description
D2C_IRQ_BUS_ERROR	6	0 = no bus error occurred or interrupt disabled 1 = bus error interrupt triggered
D2C_IRQ_DLIST	5	0 = displaylist not finished or interrupt disabled 1 = displaylist finished interrupt triggered
D2C_IRQ_ENUM	4	0 = enumeration not finished or interrupt disabled 1 = enumeration finished interrupt triggered
D2C_DLISTACTIVE	3	0 = displaylist reader is idle 1 = displaylist reader busy, no direct access to registers
D2C_CACHE_DIRTY	2	0 = Framebuffer cache is not dirty 1 = Framebuffer cache is dirty, frame should not be flipped
D2C_BUSY_WRITE	1	0 = Framebuffer writeback finished 1 = Framebuffer writeback busy, framebuffer type can not be changed
D2C_BUSY_ENUM	0	0 = enumeration unit idle 1 = enumeration unit busy, new primitive can not be started

Table 4-6 Register: D2_STATUS

4.1.7 D2_HWREVISION

Uses the same address offset as **D2_CONTROL2** for reading access.

Offset:	0x01
Default Value	0x001e000a (D/AVE 2D-TS) 0x000e100a (D/AVE 2D-TL)
Access	Read Only
Description	Read this (constant) register to identify the present hardware revision and feature set

Name	Bits	Description
BURSTSPLITTING	28	0 = burst length limit is not available 1 = burst length limit is available, bursts will be split if requested length exceeds limit
ALPHACHANNELBLENDING	27	0 = ACB is not available 1 = ACB is available
HILIMITERPRECISION	26	0 = QLimiter use standard 16.16 precision 1 = QLimiter can be switched to high precision
COLORKEY	25	0 = no color keying 1 = color keying available
TEXCLUT256	24	0 = CLUT size is 16 entries 1 = CLUT size is 256 entries
RLEUNIT	23	0 = no RLE unit 1 = RLE unit available
PERFCOUNT	20	0 = no performance counter 1 = two performance counters available
TXCACHE	19	0 = no texture cache 1 = Texture cache available

FBCACHE	18	0 = no framebuffer cache 1 = framebuffer cache available
DLR	17	0 = no Display list reader 1 = Display list reader available
SWDAVE	16	0 = hardware D/AVE 1 = software D/AVE
D/AVE Type	15:12	0 = D/AVE 2D-TS 1 = D/AVE 2D-TL
Revision number	11:0	Hardware version 11:8 = branch number 7:0 = revision number

Table 4-7 Register: D2_HWREVISION

4.2 Color Registers

All color registers are write only, reading will return undefined results. When using textures the two color registers (D2_COLOR1 and D2_COLOR2) are still used to modify the color value read from texture memory.

The equation applied to each of the four color channels (when doing texture mapping) is:

(c2-c1)*tex+c1.

In order to keep the unmodified texture data c1 has to be 0 and c2 has to be 1, meaning 0x00000000 and 0xffffffff for D2_COLOR1 and D2_COLOR2 respectively.

Write:

D2_COLOR1 - Base color register
D2_COLOR2 - Secondary color register
D2_PATTERN - Pattern register

4.2.1 D2_COLOR1

Offset:	0x19	
Default Value	0x00000000	
Access	Write Only	
Description	Base color register (constant color 1)	
Name	**Bits**	**Description**
COLOR1_ALPHA	31:24	Alpha channel of color1 (0x0=transparent, 0xff=opaque)
COLOR1_RED	23:16	Red channel of color1
COLOR1_GREEN	15:8	Green channel of color1
COLOR1_BLUE	7:0	Blue channel of color1

Table 4-8 Register: D2_COLOR1

4.2.2 D2_COLOR2

Offset:	0x1A

Default Value	0x00000000	
Access	Write Only	
Description	Secondary color register (constant color 2)	
Name	**Bits**	**Description**
COLOR1_ALPHA	31:24	Alpha channel of color2 (0x0=transparent, 0xff=opaque)
COLOR1_RED	23:16	Red channel of color2
COLOR1_GREEN	15:8	Green channel of color2
COLOR1_BLUE	7:0	Blue channel of color2

Table 4-9 Register: D2_COLOR2

4.2.3 D2_PATTERN

Offset:	0x1C	
Default Value	0x00000000	
Access	Write Only	
Description	Each bit in the pattern register is interpreted as a reference to one of the two color registers (‘0’ = D2_COLOR1 ; ‘1’ = D2_COLOR2).	
Name	**Bits**	**Description**
PATTERN	7:0	Bitmap of the pattern

Table 4-10 Register: D2_PATTERN

4.3 Geometry Registers

The geometry registers mainly describe the shape of the graphical primitive. All geometry registers are write only, reading will return undefined results.

A detailed description would not be understandable without deep knowledge of the internal rendering mechanism of D/AVE 2D and is therefore not provided in this document.

Offset:	0x04-0x17
Default Value	0x00000000
Access	Write Only
Description	Geometry registers

Table 4-11 Register: Geometry registers

4.4 Texture Registers

All texture registers are write only, reading will return undefined results. Patterns and textures require a 1D or 2D index (UV Coordinate) at every pixel. Specialized texture mapping units are available to generate these.

Write:

D2_TEXORIGIN - Texture base address
D2_TEXPITCH - Texels per texture line
D2_TEXMASK - U/V Texture mask
D2_TEXCLUT - Write access to Color Lookup Table for index texture format.

4.4.1 D2_ TEXORIGIN

Offset:	0x2F	
Default Value	0x00000000	
Access	Write Only	
Description	Address of the upper left corner texel. Addresses inside the framebuffer can be used as valid texture origin as well.	
Name	**Bits**	**Description**
TEXORIGIN	31:0	Texture base address

Table 4-12 Register: D2_ TEXORIGIN

4.4.2 D2_ TEXPITCH

Offset:	0x2D	
Default Value	0x00000000	
Access	Write Only	
Description	Texels per texture line. Pitch is equal or bigger than texture width.	
Name	**Bits**	**Description**
TEXPITCH	31:0	Texels per texture line

Table 4-13 Register: D2_ TEXPITCH

4.4.3 D2_ TEXMASK

Offset:	0x2E	
Default Value	0x00000000	
Access	Write Only	
Description	Texture size or texture address mask	
Name	**Bits**	**Description**
TEXUMASK	31:11	V mask
TEXVMASK	10:0	U mask

Table 4-14 Register: D2_ TEXMASK

4.4.4 D2_ TEXCLUT

Offset:	0x36	
Default Value	0x00000000	
Access	Write Only	
Description	Color Lookup Table for the indexed texture format.	
Name	**Bits**	**Description**
CLUT_INDEX	31:24	Index of CLUT-Entry to write
CLUT_ENTRY	23:0	RGB888 CLUT Entry

Table 4-15 Register: D2_ TEXCLUT

4.4.5 D2_ TEXCLUT_ADDR

Offset:	0x37	
Default Value	0x00000000	
Access	Write Only	
Description	Write starting address of Color Lookup Table for the indexed texture format if CLUT size is 256.	
Name	**Bits**	**Description**
	31:8	not used
CLUT_ADDR	7:0	write address

Table 4-16 Register: D2_ TEXCLUT_ADDR

4.4.6 D2_ TEXCLUT_DATA

Offset:	0x38	
Default Value	0x00000000	
Access	Write Only	
Description	Write data to CLUT_ADDR of Color Lookup Table if CLUT size is 256. After each data write CLUT_ADDR is incremented by 1.	
Name	**Bits**	**Description**
CLUT_DATA	31:0	CLUT data (format = ARGB8888)
CLUT_DATA	15:0	CLUT data (format = RGB565) for index*2
CLUT_DATA	31:16	CLUT data (format = RGB565) for index*2+1

Table 4-17 Register: D2_ TEXCLUT_DATA

4.4.7 D2_ TEXCLUT_OFFSET

Offset:	0x39	
Default Value	0x00000000	
Access	Write Only	
Description	Offset to the index for the indexed texture formats i8, i4, i2 and i1	
Name	**Bits**	**Description**
CLUT_OFFSET	7:0	CLUT offset

Table 4-18 Register: D2_ TEXCLUT_OFFSET

4.4.8 D2_ COLKEY

Offset:	0x40	
Default Value	0x00000000	
Access	Write Only	
Description	The R, G and B components of the internal color representation of a texel is compared with the color key. If the values are equal then A, R, G and B are set to 0 (transparent), else A is set to 1.	
Name	**Bits**	**Description**
COLORKEY	23:0	RGB888 color value for color keying

Table 4-19 Register: D2_ COLKEY

4.4.9 Texture mapping registers

A detailed description would not be understandable without deep knowledge of the internal rendering mechanism of D/AVE 2D and is therefore not provided.

Offset:	0x24-0x2B
Default Value	0x00000000
Access	Write Only
Description	Texture mapping registers

Table 4-20 Register: Texture mapping registers

4.5 Misc. Registers

Write:

D2_SIZE	-	Bounding box dimension
D2_PITCH	-	Frame buffer pitch and spanstore delay
D2_ORIGIN	-	Address of the first pixel in framebuffer
D2_DLISTSTART	-	Display list start address
D2_PERFTRIGGER	-	Performance counter control register

Read/Write:

D2_PERFCOUNT1,2	-	Performance counters 1 and 2

4.5.1 D2_ SIZE

Offset:	0x1E	
Default Value	0x00000000	
Access	Write Only	
Description	Bounding box dimension	
Name	**Bits**	**Description**
SIZEY	31:16	Height of the bounding box
SIZEX	15:0	Width of the bounding box

Table 4-21 Register: D2_ SIZE

4.5.2 D2_ PITCH

Offset:	0x1F	
Default Value	0x00000000	
Access	Write Only	
Description	Framebuffer pitch and spanstore delay	
Name	**Bits**	**Description**
SSD	31:16	Spanstore delay
PITCH	15:0	Pitch of the framebuffer. A negative width can be used to render bottom-up instead of top-down.

Table 4-22 Register: D2_ PITCH

4.5.3 D2_ ORIGIN

Offset:	0x20	
Default Value	0x00000000	
Access	Write Only	
Description	Address of the first pixel in framebuffer. Writing to D2_ORIGIN will trigger D/AVE to start rendering.	
Name	**Bits**	**Description**
ORIGIN	31:0	Address of the first pixel in framebuffer

Table 4-23 Register: D2_ ORIGIN

4.5.4 D2_ DLISTSTART

Offset:		0x32
Default Value		0x00000000
Access		Write Only
Description	Display list start address. Setting a new display list base address triggers execution of the new dlist.	
Name	**Bits**	**Description**
DLISTSTART	31:0	Display list start address

Table 4-24 Register: D2_ DLISTSTART

4.5.5 D2_ PERFTRIGGER

Offset:		0x35
Default Value		0x00000000
Access		Write Only
Description	Performance counters control register	
Name	**Bits**	**Description**
PERFTRIGGER2	31:16	Select the internal event that will increment D2_PERFCOUNT2 register. 0 = Disable performance counter (D2PC_NONE) 1 = D/AVE active cycles (D2PC_DAVECYCLES) 2 = Framebuffer read access (D2PC_FBREADS) 3 = Framebuffer write access (D2PC_FBWRITES) 4 = Texture read access (D2PC_TXREADS) 5 = Invisible pixels (enumerated but selected with alpha 0%) (D2PC_INVPIXELS) 6 = Invisible pixels while internal fifo is empty (lost cycles) (D2PC_INVPIXELS_MISS) 7 = Display list reader active cycles (D2PC_DLRCYCLES) 8 = Framebuffer read hits (D2PC_FBREADHITS) 9 = Framebuffer read misses (D2PC_FBREADMISSES) 10 = Framebuffer write hits (D2PC_FBWRITEMISSES) 11 = Framebuffer write misses (D2PC_FBWRITEMISSES) 12 = Texture read hits (D2PC_TEXREADHITS) 13 = Texture read misses (D2PC_TEXREADMISSES) 17 = display list reader burst reads (D2_PC_DLRBURSTREADS) 18 = display list reader words read (D2_PC_DLRWORDSREAD) 20 = RLE rewind count (D2PC_RLEREWIND) 21 = texture cache burst reads (D2_PC_TEXBURSTREADS) 22 = texture cache words read (D2_PC_TEXWORDSREAD) 23 = framebuffer cache burst reads

		(D2_PC_FBBURSTREADS) 24 = framebuffer cache words read (D2_PC_FBWORDSREAD) 25 = framebuffer cache burst writes (D2_PC_FBBURSTWRITES) 26 = framebuffer cache words written (D2_PC_FBWORDSWRITTEN) 31 = Every clock cycle (D2PC_CLKCYCLES)
PERFTRIGGER1	15:0	Same as above, but for performance counter 1.

Table 4-25 Register: D2_ PERFTRIGGER

4.5.6 D2_ PERFCOUNT1

Offset:	0x33	
Default Value	0x00000000	
Access	Read/Write	
Description	Performance Counter 1. D/AVE will increment the counter on every event selected by D2_PERFTRIGGER1	
Name	**Bits**	**Description**
PERFCOUNT1	31:0	Counter Value

Table 4-26 Register: D2_ PERFCOUNT1

4.5.7 D2_ PERFCOUNT2

Offset:	0x34	
Default Value	0x00000000	
Access	Read/Write	
Description	Performance Counter 2. D/AVE will increment the counter on every event selected by D2_PERFTRIGGER2	
Name	**Bits**	**Description**
PERFCOUNT1	31:0	Counter Value

Table 4-27 Register: D2_ PERFCOUNT2

5. DISPLAY LIST FORMAT DESCRIPTION

Display lists are stored using direct register to value mappings. There is almost no abstraction, so the structure of display list is quite simple. As D/AVE registers are always 32 bit wide, list entries had to be at least of this size to be efficient. Because register indices are much smaller (6 to 7 bits) the storage format uses a packed notation to reduce the size.

5.1 Overview

- The list is stored in memory as a stream of 32-bit words.
- It contains two different types of words: ‘address words’ and ‘data words’
- The stream always starts with an ‘address word’
- Each ‘address word’ can specify up to 4 (one byte each) register indices
- The indices are read and interpreted lsb to msb (lowest byte first)
- If less then 4 indices are required, remaining entries are filled with 0x80
- An ‘address word’ is followed by 0-4 ‘data words’, one for each register index

5.2 Example

```
DWORD  0x11223344     // start of list 'address word'
DWORD  0x00000000     // data word 1 (for register index 0x44)
DWORD  0xffffffaa     // data word 2 (for register index 0x33)
DWORD  0x00000001     // data word 3 (for register index 0x22)
DWORD  0x00ce0f20     // data word 4 (for register index 0x11)
DWORD  ...            // next 'address word'
```

This stream of dwords does essentially this :

- write 0x00000000 into hw register 0x44
- write 0xffffffaa into hw register 0x33
- write 0x00000001 into hw register 0x22
- write 0x00ce0f20 into hw register 0x11

5.3 Special Cases

There are a few additional operations besides filling registers necessary to drive the D/AVE core, for example waiting for different busy signals. All special cases are encoded using address word entries equal to or above 0x80 (bit 7 set). There are no data words for such and the following indices, the stream continues with the next address word.

The behavior is unspecified if an address word contains ‘gaps’ e.g. i3, 0x80, i2, i1.
If special indices are used, no normal index must follow after them in the same address word.

If the first index contains the special index 0xff, the following (second) index is interpreted as follows:

bit 0 set : display list end
bit 1 set : wait for pipeline write to framebuffer cache complete then issue a full pipeline flush (texture cache and framebuffer cache) and wait for framebuffer cache write to framebuffer complete (necessary before flip)

bit 2 set : wait for pipeline write to framebuffer cache complete (necessary before fb format change and alpha channel blending change)

6. OPERATION MODES

6.1 Synchronization

The synchronization between CPU and D/AVE can either be done by polling or by interrupt. For best system performance it is recommended to use interrupt synchronization.

6.2 Register based

In the register based operation mode the CPU is setting all registers over the peripheral bus. To start a new render process the CPU has to wait until D/AVE is finished. Then the next render process can be initiated. It is recommended to use this mode for debug reasons only.

6.3 Display list based

In this mode the CPU creates a display list in memory. Such a display contains a bundle of render operations (basically just a list of register settings). Then the D/AVE DLR will execute the display list in parallel to the CPU. It is a requirement that the memory which stores the display list is accessible from CPU and D/AVE. Note that while the DLR is active, no register writes must be done via the slave bus interface, as they will conflict with the commands given by the DLR to the D/AVE rendering core.
A more detailed description of the display list structure can be found in chapter 5.

7. INTEGRATION OVERVIEW

This chapter gives a brief overview about how to integrate the IP.

7.1 Configurable constants

The following constants can be configured (default values are denoted in brackets).

DAVE2_MBA_ADDRESSWIDTH	Master address width in bits [32].
DAVE2_MBA_DATAWIDTH	Master data bus width in bits [32].
DAVE2_MBA_BURSTCOUNTWIDTH	Master burst count width in bits [5]. E.g. 5 means 2^5=32 words maximum burst length (must be consistent with cache line length).
DAVE2_FBCACHE_LINE_SEL_WIDTH	Width of selector signal for framebuffer cache lines [2]. E.g. 2 means 2^2=4 cache lines in framebuffer cache.
DAVE2_FBCACHE_WORD_SEL_WIDTH	Width of selector signal for framebuffer cache word inside a line [4]. E.g. 4 means 2^4=16 words per cache line.
DAVE2_TEXCACHE_LINE_SEL_WIDTH	Width of selector signal for texture cache lines [2]. E.g. 2 means 2^2=4 cache lines in texture cache.
DAVE2_TEXCACHE_WORD_SEL_WIDTH	Width of selector signal for texture cache word inside a line [5]. E.g. 5 means 2^5=32 words per cache line.

Table 7-1 Configurable constants

7.2 Clock

D/AVE is running in a single clock domain. The active edge is the rising edge. The IP is not using gated clocks and does not have any multi cycle paths.

7.3 Reset

The reset is fully asynchronous and active high. When the reset signal goes to low, the IP is ready for work after the next clock cycle.

7.4 Power Management

No known restrictions.

7.5 Memory Description

The following numbers apply for the default configuration of D/AVE 2D-TS. They can be fine-tuned using parameter settings.

RAM module name	Memory type	Number of memory blocks	Memory width (bit)	Memory depth (bit)	Memory size (bit)	Total memory size (bit)
Texture cache	dual port (1r / 1w)	1	32	128	4096	4096
Framebuffer cache	True dual port (2 r/w)	8	9	32	288	2304
DLR FIFO	dual port (1r / 1w)	1	32	32	1024	1024
Pixel selection FIFO	dual port (1r / 1w)	1	107	128	13696	13696
Color unit FIFO	dual port (1r / 1w)	1	102	128	13056	13056
RLE unit FIFO	dual port (1r /1w)	1	32	32	1024	1024
CLUT 256	single port (1 r/w)	1	32	256	8192	8192

Table 7-2 Memory blocks

7.6 Gate Count

7.6.1 FPGA synthesis results (Altera)

Tool version: Altera Quartus II 9.0 SP1
Used chip: Cyclone III EP3C25F324C8
Constraints: 66MHz / optimized for area

	D/AVE 2D-TS	**D/AVE 2D-TL**
Number of logic cells:	11k logic elements	7k logic elements
Number of RAM blocks:	16 M9K blocks	11 M9K blocks
Number of DSP blocks:	17 multipliers	7 multipliers

7.6.2 FPGA synthesis results (Xilinx)

Tool version: Xilinx ISE 13.2
Used chip: Kintex 7 xc7k70t-fbg676-1
Constraints: 100 MHz

	D/AVE 2D-TS	**D/AVE 2D-TL**
Number of occupied Slices:	2716 slices	n.a.
Number of DSP48E1s	7	n.a.
Number of RAMB36E1s	3	n.a.
Number of RAMB18E1s	7	n.a.

7.6.3 ASIC synthesis results

As the ASIC gate count varies depending on the process and the synthesis tool used, these are only estimates based on available ASIC synthesis results.

	D/AVE 2D-TS	**D/AVE 2D-TL***
Gate count	< 100k gates	~60k gates
Memory bits (see 7.5)	< 35k memory bits	<20k memory bits

* estimates based on FPGA synthesis results

7.7 Driver Memory Footprint

The static code size of the D/AVE 2D driver is in the range of 90 to 100kBytes, depending on the platform and the compiler.
The dynamic memory requirements strongly depend on the actual application.

8. LIMITATIONS

8.1 Framebuffer limitations

D/AVE is verified for resolutions up to 1024 x 1024 pixels so far.
The hardware interface could support resolutions up to 32767 x 65535 pixels in theory, but big primitives covering larger screen areas are likely to have accuracy problems (especially quadratic primitives like circles etc.)
From the software side, the value range for coordinates going into the driver is limited to an even smaller range of values between -2048.0 and 2047.9375, so for larger resolutions, a driver change is required.
Another relevant limitation is the fill rate: Fast full-screen animations on large screens won't be possible due to the high number of pixels to be rendered. However, HMI like applications which only update small parts of the screen per frame or applications with a low fps requirement will be possible.

8.2 Texture limitations

D/AVE handles textures with a maximum width of 2047 pixels and a maximum height of 1024 pixels.
If necessary, D/AVE can be changed to allow for bigger textures.

IMAGES

TABLES