

**TMS320AV7110**  
**Integrated Digital Set-top Box Decoder**  
**Functional Specification**

Last update: 07/06/98  
Revision 3.1

## Table of Contents

<b>1. INTRODUCTION</b> .....	<b>7</b>
1.1 FEATURES .....	7
<b>2. PIN FUNCTIONS</b> .....	<b>8</b>
2.1 PIN TYPES .....	11
<b>3. ARCHITECTURE</b> .....	<b>12</b>
3.1 FUNCTIONAL BLOCK DIAGRAM.....	12
3.2 OVERVIEW .....	12
3.2.1 <i>Transport bit-stream processing</i> .....	14
3.2.2 <i>Audio and Video data processing</i> .....	14
3.3 SOFTWARE SYSTEM OVERVIEW .....	14
<b>4. ARM CPU</b> .....	<b>16</b>
4.1 FEATURES .....	16
4.2 BLOCK DIAGRAM.....	16
4.3 RESOURCE MANAGEMENT.....	17
<b>5. TRAFFIC CONTROLLER (TC)</b> .....	<b>18</b>
5.1 FEATURES .....	18
5.2 SDRAM INTERFACE.....	19
5.3 PRIORITIZED INTERRUPTS .....	23
5.4 MANAGING DMA TRANSFER .....	23
5.5 VIDEO/AUDIO BUFFER MONITORING SUPPORT.....	23
5.6 TRANSFER OF VIDEO/AUDIO DATA FROM THE HIGH SPEED DATA INTERFACE (HSDI) TO SDRAM .....	24
<b>6. MPEG TRANSPORT DECODER (TPP)</b> .....	<b>25</b>
6.1 THE TPP MODULE .....	25
6.1.1 <i>Features</i> .....	25
6.1.2 <i>Description</i> .....	25
6.1.3 <i>Conditional access and Descrambling processing</i> .....	26
6.1.4 <i>Transport Parser Input Interface</i> .....	26
6.2 HARDWARE SECTION FILTERING.....	27
6.2.1 <i>Memory Map</i> .....	28
6.2.2 <i>Filter hardware definition</i> .....	28
6.2.3 <i>Filter Match Data Register definition</i> .....	29
6.2.4 <i>Match Result Register Definition</i> .....	30
6.2.5 <i>Status Register Word Definition</i> .....	30
6.2.6 <i>Filter Reset Control Register</i> .....	31
6.2.7 <i>Typical software process flow</i> .....	31
6.2.8 <i>CPU load assessment</i> .....	32
6.3 HARDWARE CRC CIRCUIT .....	33
<b>7. VIDEO DECODER</b> .....	<b>34</b>
7.1 FEATURES .....	34
7.2 DESCRIPTION .....	34
7.3 TRICK MODE.....	35
7.4 HIGH-LEVEL COMMANDS .....	35
7.5 VIDEO DECIMATION AND SCALING .....	35
7.6 FREE-RUN AND VBV-BASED DISPLAY SYNCHRONIZATION .....	37
7.7 MEMORY USAGE REDUCTION .....	38
7.7.1 <i>Vsync Reset</i> .....	38
7.7.2 <i>Split Video Input Buffer</i> .....	39
7.8 CONFIGURATION OF THE VIDEO DECODER.....	39
7.8.1 <i>Configuration of video display format</i> .....	39

7.8.2	<i>True Size Display Mode</i> .....	39
7.8.3	<i>Video display synchronization mode</i> .....	40
7.8.4	<i>Vsync reset and split VBV buffer</i> .....	40
<b>8.</b>	<b>OSD &amp; GRAPHICS ACCELERATION</b> .....	<b>41</b>
8.1	OSD .....	41
8.1.1	<i>Description</i> .....	41
8.1.2	<i>OSD data storage</i> .....	42
8.2	SETTING UP AN OSD WINDOW .....	42
8.2.1	<i>CAM Memory</i> .....	42
8.2.2	<i>Window Attribute Memory</i> .....	43
8.2.3	<i>Color Look Up Table</i> .....	43
8.2.4	<i>Blending and Transparency</i> .....	43
8.2.5	<i>Hardware Cursor</i> .....	45
8.2.6	<i>Output Channels</i> .....	45
8.3	THE BITBLT HARDWARE.....	46
8.3.1	<i>Sources and Destinations</i> .....	47
8.3.2	<i>Source and Destination Window Formats</i> .....	47
8.3.3	<i>Transparency</i> .....	47
<b>9.</b>	<b>VIDEO OUTPUT INTERFACES</b> .....	<b>48</b>
9.1	ANALOG VIDEO OUTPUT - NTSC/PAL ENCODER MODULE .....	48
9.2	TELETEXT AND WIDE SCREEN SIGNALING (WSS) INSERTION (PAL MODE ONLY).....	48
9.2.1	<i>Teletext insertion into PAL CVBS</i> .....	48
9.2.2	<i>WSS insertion into CVBS</i> .....	49
9.2.3	<i>Insertion mode</i> .....	49
9.3	CLOSED CAPTION, EXTENDED DATA SERVICES, AND VIDEO ASPECT RATIO IDENTIFICATION SIGNAL INSERTION (NTSC MODE ONLY) .....	49
9.4	DIGITAL VIDEO OUTPUT .....	50
9.4.1	<i>PAL Mode Digital Video Output</i> .....	50
9.4.2	<i>NTSC Mode Digital Video Output</i> .....	50
<b>10.</b>	<b>AUDIO DECODER</b> .....	<b>52</b>
10.1	FEATURES .....	52
10.2	PCM AUDIO OUTPUT.....	53
10.2.1	<i>Using an External Audio PLL</i> .....	55
10.3	PCM BYPASS.....	55
10.4	ELEMENTARY STREAM PLAYBACK .....	56
10.5	SPDIF AUDIO OUTPUT .....	56
<b>11.</b>	<b>SYNCHRONIZATION</b> .....	<b>57</b>
11.1	SYSTEM TIME CLOCK .....	57
11.2	STARTUP SYNCHRONIZATION.....	57
11.3	RUNTIME SYNCHRONIZATION .....	57
11.4	AUDIO SYNCHRONIZATION.....	58
11.5	VIDEO SYNCHRONIZATION.....	58
11.6	AUDIO-VIDEO SYNCHRONIZATION (LIP-SYNC).....	59
<b>12.</b>	<b>EXTENSION BUS INTERFACE (EBI)</b> .....	<b>60</b>
12.1	ADDRESS RANGE AND WAIT STATE OF CHIP SELECT .....	60
12.2	EBI READ AND WRITE CYCLES .....	61
12.3	INTERRUPTS .....	61
12.4	THE EXTWAIT SIGNAL .....	62
12.5	THE EXTENSION BUS DRAM.....	62
12.6	BYTE ORDERING ON THE EXTENSION BUS .....	63
<b>13.</b>	<b>HIGH SPEED DATA INTERFACE (HSDI)</b> .....	<b>65</b>
13.1	IEEE 1394 INTERFACE .....	66
13.1.1	<i>The 'AV7110 reads data from the MPEG2Lynx</i> .....	67

13.1.2	The 'AV7110 writes data to the MPEG2Lynx.....	68
13.1.3	The 'AV7110 Internal Data Path for 1394 .....	68
13.2	EXTERNAL DMA INTERFACE.....	70
13.3	IEEE 1284 INTERFACE .....	72
13.4	COMBINING THE 1394, 1284 AND EXTERNAL CONTROLLER INTERFACES.....	73
<b>14.</b>	<b>COMMUNICATION PROCESSOR.....</b>	<b>75</b>
14.1	FEATURES .....	75
14.2	SMART CARD INTERFACE .....	75
14.3	TIMERS .....	77
14.4	UARTS.....	78
14.5	IR INPUT PORT .....	78
14.6	IR OUTPUT PORT .....	79
14.7	GENERAL PURPOSE I/Os .....	80
14.8	I <sup>2</sup> C INTERFACE .....	81
<b>15.</b>	<b>DEVICE CONTROL INTERFACES.....</b>	<b>82</b>
15.1	RESET .....	82
15.2	JTAG PINS .....	82
<b>16.</b>	<b>REGISTER MAPS.....</b>	<b>83</b>
16.1	NTSC/PAL REGISTERS - BASE ADDRESS 0x72000000.....	83
16.2	BitBLT REGISTERS - BASE ADDRESS 0x70000000.....	85
16.3	CCP REGISTERS - BASE ADDRESS 0x6E000000.....	86
16.4	AUDIO DECODER REGISTERS - BASE ADDRESS 0x6C000000 .....	94
16.5	VIDEO DECODER REGISTERS - BASE ADDRESS 0x6A000000 .....	96
16.6	OSD REGISTERS - BASE ADDRESS 0x68000000.....	99
16.7	TC REGISTERS - BASE ADDRESS 0x66000000.....	101
16.8	TPP REGISTERS - BASE ADDRESS 0x64000000.....	105
<b>17.</b>	<b>TIMING DIAGRAMS AND DC PARAMETERS.....</b>	<b>108</b>
17.1	DRAM INTERFACE TIMING .....	109
17.2	EBI INTERFACE TIMING.....	111
17.3	SDRAM INTERFACE TIMING .....	115
17.4	VIDEO OUTPUT TIMINGS .....	121
17.5	HSDI TIMING DIAGRAMS .....	123
17.6	EXTERNAL DMA .....	125
17.7	INPUT STREAM TIMING .....	127
17.8	DC CHARACTERISTICS.....	128
17.9	SDRAM MEMORY MAP FOR FIRST SILICON FIRMWARE .....	129
17.9.1	PAL mode Memory Map .....	129
17.9.2	NTSC mode Memory Map.....	130

### List of Tables

TABLE 1. THE 'AV7110 PINS.....	8
TABLE 2. SDRAM SELECTION REQUIREMENTS.....	19
TABLE 3. DECODED BITSTREAM INFORMATION.....	21
TABLE 4. ERROR STATISTICS AND MONITOR INFORMATION .....	22
TABLE 5. FLAGS AND COMMANDS.....	22
TABLE 6. GPDMA SOURCES AND DESTINATIONS .....	23
TABLE 7. TRANSPORT PACKET INPUT INTERFACE PIN DESCRIPTION .....	27
TABLE 8. SUPPORTED VIDEO RESOLUTIONS FOR AUTOMATIC UP-SAMPLING .....	34
TABLE 9. VIDEO DECODER COMMANDS .....	35
TABLE 10. SCALING FACTORS FOR 4:3 MONITOR FOR PAL FORMAT. ....	37
TABLE 11. SCALING FACTORS FOR 16:9 MONITOR FOR PAL FORMAT. ....	37
TABLE 12. SCALING FACTORS FOR 4:3 MONITOR FOR NTSC FORMAT.....	37
TABLE 13. SCALING FACTORS FOR 16:9 MONITOR FOR NTSC FORMAT.....	37
TABLE 14. THE 'AV7110 MEMORY USAGE .....	38
TABLE 15. STORAGE OF B FRAMES.....	38
TABLE 16. TARGET OSD MEMORY SPACE IN SDRAM (IN BYTES).....	39
TABLE 17. SDRAM OSD WINDOW SIZE.....	42
TABLE 18. BLENDING LEVELS .....	44
TABLE 19. BLENDING AND TRANSPARENCY.....	44
TABLE 20. OSD MODULE OUTPUT CHANNEL CONTROL .....	46
TABLE 21. SOURCE AND DESTINATION MEMORIES FOR BITBLT .....	47
TABLE 22. ALLOWABLE BITBLT WINDOW FORMATS .....	47
TABLE 23. MULTIPLEXED VIDEO OUTPUT DEFINITIONS .....	48
TABLE 24. DIGITAL OUTPUT CONTROL.....	51
TABLE 25. VARIS CODE SPECIFICATION.....	51
TABLE 26. THREE BYTE VARIS CODE.....	51
TABLE 27. CODING OF ASPECT RATIO IN THE VARIS CODE.....	51
TABLE 28. AUDIO MODULE REGISTERS .....	52
TABLE 29. PCMCLK FREQUENCIES .....	55
TABLE 30. SYSTEM TIME CLOCK MODEL.....	57
TABLE 31. EXTENSION BUS CHIP SELECT ASSIGNMENTS .....	61
TABLE 32. HIGH SPEED DATA INTERFACE SIGNAL PIN ASSIGNMENT .....	65
TABLE 33. TYPES OF DMA TRANSFERS ALLOWED FOR DATA PORTS .....	65
TABLE 34. 1394 INTERFACE SIGNALS.....	67
TABLE 35. 1394 CONTROL LINES DESCRIPTION .....	67
TABLE 36. EXTERNAL DMA INTERFACE SIGNALS.....	70
TABLE 37. EDMA REGISTER DEFINITION.....	71
TABLE 38. IEEE 1284 INTERFACE SIGNALS .....	73
TABLE 39. SMART CARD PIN DESCRIPTION.....	75
TABLE 40. GROUP 1 F VALUES .....	76
TABLE 41. GROUP 2 F VALUES .....	76
TABLE 42. TIMER CONTROL AND STATUS REGISTERS .....	77
TABLE 43. I/O CONTROL/STATUS REGISTERS, IOCSRN.....	81
TABLE 44. GPIO_IRQ BIT DEFINITIONS .....	81

## List of Figures

FIGURE 1. THE 'AV7110 BLOCK DIAGRAM.....	12
FIGURE 2. THE 'AV7110 MEMORY MAP.....	13
FIGURE 3. SOFTWARE BLOCK DIAGRAM.....	15
FIGURE 4. ARM CORE DATA PATH.....	17
FIGURE 5. TRAFFIC CONTROLLER DATA FLOW.....	18
FIGURE 6. DEFAULT MEMORY ALLOCATION OF SDRAM (NTSC).....	20
FIGURE 7. DEFAULT MEMORY ALLOCATION OF SDRAM (PAL).....	21
FIGURE 8. EXAMPLE CIRCUIT FOR 27 MHz CLOCK GENERATION.....	26
FIGURE 9. FEC INPUT INTERFACE TO TPP TIMING.....	27
FIGURE 10: HARDWARE FILTER LAYOUT.....	28
FIGURE 11. DISPLAY FORMATS FOR THE 'AV7110.....	36
FIGURE 12. OSD MODULE BLOCK DIAGRAM.....	42
FIGURE 13. USING THE OSD_ACTIVE SIGNAL FOR EXTERNAL ANALOG VIDEO.....	45
FIGURE 14. OSD OUTPUT CHANNELS MATRIX.....	46
FIGURE 15. EXTERNAL INSERTION OF TELETEXT SIGNAL.....	49
FIGURE 16. DIGITAL VIDEO OUTPUT TIMING.....	50
FIGURE 17. PCM OUTPUT TIMING (16-BIT PCM FORMAT).....	54
FIGURE 18. PTS TRACKING DIAGRAM.....	58
FIGURE 19. EXAMPLES OF DRAM CONNECTIONS TO 16-BIT AND 32-BIT EXTENSION BUSES.....	63
FIGURE 20. BYTE ORDERING ON THE EXTENSION BUS.....	64
FIGURE 21. FUNCTIONAL BLOCK DIAGRAM OF HIGH SPEED DATA INTERFACE.....	66
FIGURE 22. 1394 INTERFACE.....	66
FIGURE 23. 1394 INTERFACE READ SEQUENCE.....	68
FIGURE 24. 1394 INTERFACE WRITE SEQUENCE.....	68
FIGURE 25. 1394 DATA FLOW BLOCK DIAGRAM.....	69
FIGURE 26. INTERFACING THE 'AV7110 TO A SCSI CHIP.....	70
FIGURE 27. PROGRAMMABLE DELAY FOR REGISTER ACCESSES ON THE EDMA PORT.....	71
FIGURE 28. PROGRAMMABLE DELAY FOR DMA ACCESSES ON THE EDMA PORT.....	72
FIGURE 29. INTERFACING THE 'AV7110 TO AN IEEE-1284 BUS.....	72
FIGURE 30. COMBINED MPEG2LYNX, 1284 AND SCSI CONTROLLER.....	74
FIGURE 31. THE 'AV7110'S INTERFACE TO TWO SMART CARDS.....	76
FIGURE 32. READ ACCESS FROM DRAM.....	109
FIGURE 33. WRITE ACCESS FROM DRAM.....	109
FIGURE 34. READ-MODIFY-WRITE ACCESS FOR 32 BIT DRAM.....	110
FIGURE 35. READ-MODIFY-WRITE ACCESS FOR 32 BIT DRAM.....	110
FIGURE 36. CAS BEFORE RAS DRAM REFRESH TIMING.....	111
FIGURE 37. EXTENSION BUS SINGLE ACCESS READ TIMING (4 WAIT STATES).....	111
FIGURE 38. EXTENSION BUS WRITE TIMING (4 WAIT STATES).....	112
FIGURE 39. EXTENSION BUS TIMING FOR MULTI-ACCESS MODE (3 WAIT STATES).....	112
FIGURE 40. READ WITH EXTWAIT ACTIVE.....	113
FIGURE 41. WRITE WITH EXTWAIT ACTIVE.....	114
FIGURE 42. SDRAM READ CYCLE.....	115
FIGURE 43. SDRAM WRITE CYCLE.....	116
FIGURE 44. SDRAM MULTI READ CYCLE.....	117
FIGURE 45. SDRAM MULTI WRITE CYCLE.....	118
FIGURE 46. SUCCESSIVE SDRAM OPERATIONS.....	119
FIGURE 47. VARIS CODE OUTPUT TIMING.....	121
FIGURE 48. DIGITAL VIDEO OUTPUT TIMING.....	122
FIGURE 49. 1394 READ CYCLE TIMING.....	123
FIGURE 50. 1394 WRITE CYCLE TIMING.....	124
FIGURE 51. EXTERNAL DMA CYCLE TIMING.....	125
FIGURE 52. EXTERNAL DMA "REGISTER ACCESS" TIMING.....	126
FIGURE 53. INPUT INTERFACE TIMINGS.....	127

## 1. Introduction

The TMS320AV7110 Integrated Set-top Box Decoder is the major component of a Digital Video Broadcast (DVB) Settop Box. It incorporates: an ARM CPU and a transport packet parser (TPP) with an integrated European Common Descrambler (ECD), an MPEG-2 video decoder, an MPEG-1 audio decoder, an NTSC/ PAL video encoder, an on screen display (OSD) controller to mix graphics and video, a configurable high speed data interface, three UART serial data interfaces, programmable infra red (IR) input and output ports, a Smart Card interface, and an extension bus to connect peripherals such as: additional RS232 ports, display and control panels, and extra ROM, DRAM, or EPROM memory. External program and data memory expansion allows the IC to support a range of set-top boxes from low end to high end.

### 1.1 Features

- fully functional decoder using a single 16 Mbit external SDRAM
- accepts transport bit-streams of up to 72.8Mbps burst rate (60 Mbps average over one transport packet)
- on-chip European Common Descrambler hardware
- hardware CRC accelerator for SI/PSI information processing
- video decoder that decodes MPEG-1 and MPEG-2 Main Profile@Main Level bit-streams
- audio decoder that decodes MPEG-1 Layer I and II and the basic stereo channels from MPEG-2 Multi-channel bit-streams
- audio output in both PCM and SPDIF formats
- OSD processor that enables mixture of OSD and video data with transparency
- BitBLT hardware that accelerates memory block move
- 32/16 bit ARM/Thumb processor that removes the need of another CPU in the set-top box
- firmware that controls device operation and provides application access to hardware resources
- on-chip NTSC/PAL encoder that incorporates MacroVision logic for anti-taping protection
- analog Y/C or RGB, and Composite video outputs with 9-bit precision
- internally or externally generated video sync signals
- on-chip SDRAM controller for 16, 20, or 32 Mbit SDRAM
- general purpose 16-/32-bit Extension Bus Interface (EBI)
- a configurable high speed data interface to connect to either an IEEE 1394 link device, an IEEE 1284 interface, or an external DMA device like SCSI that supports up to 16 Mbps data rate
- two 4-wire UART data ports supporting up to 115.2Kbps rate, and one 2-wire UART data port supporting up to 9.6Kbps rate
- ISO 7816-3/NDC Smart Card interface
- I<sup>2</sup>C master/slave interface
- a programmable IR input port
- a programmable IR output port
- four dedicated general purpose I/O pins and five multiplexed pins which can be configured to be general purpose I/O pins
- 2.5 volt device with 3.3 volt I/O

## 2. Pin Functions

The 'AV7110 is packaged in a 272 pin BGA. Table 1 is a list of pin names and their descriptions. Multiplexed pins (muxed pins) are those whose definitions can be selected by the user using APIs. For example, if there is no need for the SCVccDetect signal for the smart card interface on the set top box, the pin SCVccDetect can be configured to become general purpose IO pin IO3.

**Table 1. the 'AV7110 Pins**

Pin Name	#	I/O	Description
<b>Transport Parser</b>			
DATAIN[7:0]	8	I	Data Input. Bit 7 is the first bit in the transport stream
DCLK	1	I	Input Data Clock. (Note 1). The maximum frequency is 7.5 MHz.
PACCLK	1	I	Packet Clock. Indicates valid packet data on DATAIN.
BYTE_START	1	I	Byte Start. Indicates the first byte of a transport packet for DVB.
DERROR	1	I	Data Error, active high. Indicates an error in the input data. Tie low if not used.
<b>Extension Bus</b>			
EXTRW	1	O	Extension Bus Read/Write. Selects read when high, write when low.
EXTOE/EXTACTIVE	1	O	Active low, user programmable EXTOE: default mode, asserted only during read cycles EXTACTIVE: asserted for read/write cycles
EXTWAIT	1	I	Extension Bus Wait Request, active low
EXTADDR[24:0]	25	O	Extension Address bus: byte address
EXTDATA[31:16]	16	I/O	Extension Bus Data (16-bit or 32-bit EBI)
EXTDATA[15:0]	16	I/O	Extension Bus Data in 32-bit EBI mode (muxed with other signal pins in 16-bit EBI mode)
EXTINT[2:0]	3	I	External Interrupt requests (IRQ) , active low.
EXTACK[2:0]	3	O	External Interrupt request acknowledge, active low
IO1	1	I/O	General Purpose I/O
IO2	1	I/O	General Purpose I/O
IO3	1	I/O	General Purpose I/O (muxed with SCVccDetect)
IO4	1	I/O	General Purpose I/O
IO5	1	I/O	General Purpose I/O
IO6	1	I/O	General Purpose I/O (muxed with EXTDATA[15])
IO7	1	I/O	General Purpose I/O (muxed with EXTDATA[14])
IO8	1	I/O	General Purpose I/O (muxed with EXTDATA[13])
IO9	1	I/O	General Purpose I/O (muxed with EXTDATA[12])
CLK40	1	O	40.5 MHz Clock output for the extension bus and HSDI interface
CS1	1	O	Chip Select 1. Selects EPROM
CS2	1	O	Chip Select 2.
CS3	1	O	Chip Select 3.
CS4	1	O	Chip Select 4.
CS5	1	O	Chip Select 5.
CS6	1	O	Chip Select 6.
RAS1	1	O	DRAM Row Address Strobe 1
RAS2	1	O	DRAM Row Address Strobe 2 (for 32-bit support)
RAS3	1	O	DRAM Row Address Strobe 3 (for 2 <sup>nd</sup> DRAM module at a fixed address partition).



Pin Name	#	I/O	Description
UCAS	1	O	DRAM Column address strobe for upper byte
LCAS	1	O	DRAM Column address strobe for lower byte
<b>High Speed Parallel Data Interface (See Table 32 in Section 13 for details)</b>			
HSDI_DATA[7:0]	8	I/O	Data Bus
HSDI_SIG1	1	O	
HSDI_SIG2	1	O	
HSDI_SIG3	1	I or O	
HSDI_SIG4	1	I/O, I or O	
HSDI_SIG5	1	I/O, I or O	
HSDI_SIG6	1	I/O or O	
HSDI_SIG7	1	I	
HSDI_SIG8	1	O	
HSDI_SIG9	1	O	
HSDI_STATUS[1:0]	2	O	Signals to indicate which of the three device interfaces (1394, External DMA, or 1284) is active
<b>Smart Card Interface</b>			
SCDET1	1	I	Smart Card Detect Card 1, (programmable polarity)
SCDET2	1	I	Smart Card Detect Card 2, (programmable polarity)
SCRESET	1	O	Smart Card Reset.
SCCLK	1	O	Smart Card Clock.
SCVppEN	1	O	Smart Card Vpp enable (programmable polarity)
SCVccEN	1	O	Smart Card Vcc enable (programmable polarity).
SCDATAIO	1	I/O	Smart Card data Input / Output.
SCSEL	1	O	Smart Card selection signal: Low = Smart Card 1 Selected, High = Smart Card 2 Selected.
SCVccDetect	1	I	Smart Card Vcc detect input signal (for NDC mode). (muxed with IO3)
<b>Communications Processor</b>			
IRIN	1	I	Infra-Red sensor input
IROUT	1	O	Infra-Red sensor output.
UART_DI1	1	I	UART port 1, Data Input
UART_DO1	1	O	UART port 1, Data Output
UART_RTS1	1	O	UART port 1, RTS.
UART_CTS1	1	I	UART port 1, CTS.
UART_CK16	1	O	UART port 1's x16 UART clock output
UART_DI2	1	I	UART port2, Data Input
UART_DO2	1	O	UART port2, Data Output
UART_RTS2	1	O	UART port 2, RTS.
UART_CTS2	1	I	UART port 2, CTS.
UART_DI3	1	I	UART port3, Data Input
UART_DO3	1	O	UART port3, Data Output
IICS_SDA	1	I/O	I <sup>2</sup> C Interface Serial Data, open drain
IICS_SCL	1	I/O	I <sup>2</sup> C Interface Serial Clock, open drain
<b>SDRAM Interface</b>			
SDATA[15:0]	16	I/O	SDRAM Data bus.
SADDR[11:0]	12	O	SDRAM Address bus.
SRAS	1	O	SDRAM Row Address Strobe
SCAS	1	O	SDRAM Column Address Strobe

Pin Name	#	I/O	Description
SWE	1	O	SDRAM Write Enable
SDOMU	1	O	SDRAM Data Mask Enable, Upper byte.
SDOML	1	O	SDRAM Data Mask Enable, Lower byte.
SCLK	1	O	SDRAM Clock
SCKE	1	O	SDRAM Clock Enable
SCS1	1	O	SDRAM Chip Select 1
SCS2	1	O	SDRAM Chip Select 2
<b>Audio Interface</b>			
PCMDATA	1	O	PCM Data audio output.
LRCLK	1	O	Left/Right Clock for output PCM audio data.
PCMCLK	1	I/O	PCM Clock (for Input - Note 1).
ASCLK	1	O	Audio Serial Data Clock
SPDIF	1	O	SPDIF audio output
APLL_LOCK	1	O	Remains active while the Internal Audio PLL is stable.
<b>VCXO</b>			
CLK27	1	I	27 MHz Clock input from an external VCXO (Note 1)
VCXO_CTRL	1	O	Digital pulse output for external VCXO Control.
CLK_SEL	1	I	Clock select: Low = 27 MHz input clock, High = 81 MHz input clock.
<b>Digital Video Interface</b>			
YCOUT[7:0]	8	O	4:2:2 digital video output. (muxed with EXTDATA[7:0])
YCCLK	1	O	27 MHz digital video output clock. (muxed with EXTDATA[8])
YCCTRL[1:0]	2	O	Digital video output control signals (muxed with EXTDATA[11] and EXTDATA[10] respectively)
TEXTEN	1	O	Teletext window signal - active during Teletext lines. (muxed with EXTDATA[9])
<b>NTSC/PAL Encoder Interface</b>			
VSYNC	1	I/O	Vertical synchronization signal
HSYNC	1	I/O	Horizontal synchronization signal
OSD_ACTIVE	1	O	Active when OSD data is being output on video out (active high)
RC_OUT	1	O	Red or C video signal Output.
BIAS_RC	1	I	Red or C D/A Bias terminal.
GY_OUT	1	O	Green or Y video signal Output.
BIAS_GY	1	I	Green or Y D/A Bias terminal.
B_OUT	1	O	Blue video signal Output.
BIAS_B	1	I	Blue D/A Bias terminal.
Y/COMP_OUT	1	O	Composite or Y video signal Output.
BIAS_COMP	1	I	Composite Bias terminal.
COMP[3:0]	4	I	Compensation-capacitor terminals. [BGRC=3:0]
VREF	1	I	Voltage reference.
<b>Device Control</b>			
RESET	1	I	Reset, active low
TDI	1	I	JTAG Data Input. Can be tied high or left floating.
TCK	1	I	JTAG Clock. Must be tied low for normal operation.
TMS	1	I	JTAG Test Mode Select Can be tied high or left floating.
TRST	1	I	JTAG Test Reset, active low. Must be tied low or connected to RESET for normal operations.
TDO	1	O	JTAG Data Output
test	4	I	Reserved for Test, tie to ground when not in use

Pin Name	#	I/O	Description
<b>Power</b>			
VDD2_5V			2.5 v supply to core of chip. $\pm 5\%$ tolerance.
VDD3_3V			3.3 v Digital supply to IO of chip. $\pm 5\%$ tolerance.
VSS			Digital Ground.
AVCC_CPLL	1		3.3 v Analog supply. $\pm 5\%$ tolerance. - Clock PLL
AGND_CPLL	1		Analog Ground - Clock PLL
AVCC_APLL1	1		3.3 v Analog Supply - Audio PLL1
AGND_APLL 1	1		Analog Ground - Audio PLL1
AVCC_APLL 2	1		3.3 v Analog Supply - Audio PLL2
AGND_APLL 2	1		Analog Ground - Audio PLL2
AVCC_GY	1		3.3 v Analog Supply - DAC for GY_OUT
AGND_GY	1		Analog Ground - DAC for GY_OUT
AVCC_RC	1		3.3 v Analog Supply - DAC for RC_OUT
AGND_RC	1		Analog Ground - DAC for RC_OUT
AVCC_B	1		3.3 v Analog Supply - DAC for B_OUT
AGND_B	1		Analog Ground - DAC for B_OUT
AVCC_COMP	1		3.3 v Analog Supply - DAC for COMP_OUT
AGND_COMP	1		Analog Ground - DAC for COMP_OUT

Note 1: These signals are Schottky inputs

## 2.1 Pin Types

- All I/O pins power up in input mode.
- All outputs are 3.3V output and will require external level shifters to interface with 5V devices.
- All digital inputs/outputs comply with normal TTL standard:  $V_{ih} = 2.0V_{max}$ ,  $V_{il} = 0.8V_{min}$ ,  $V_{oh} = V_{cc} - 0.4V_{min}$ ,  $V_{ol} = 0.4V_{max}$ , for the specified source/sink currents.

### 3. Architecture

#### 3.1 Functional Block Diagram

Figure 1 shows a functional block diagram of the TMS320AV7110.

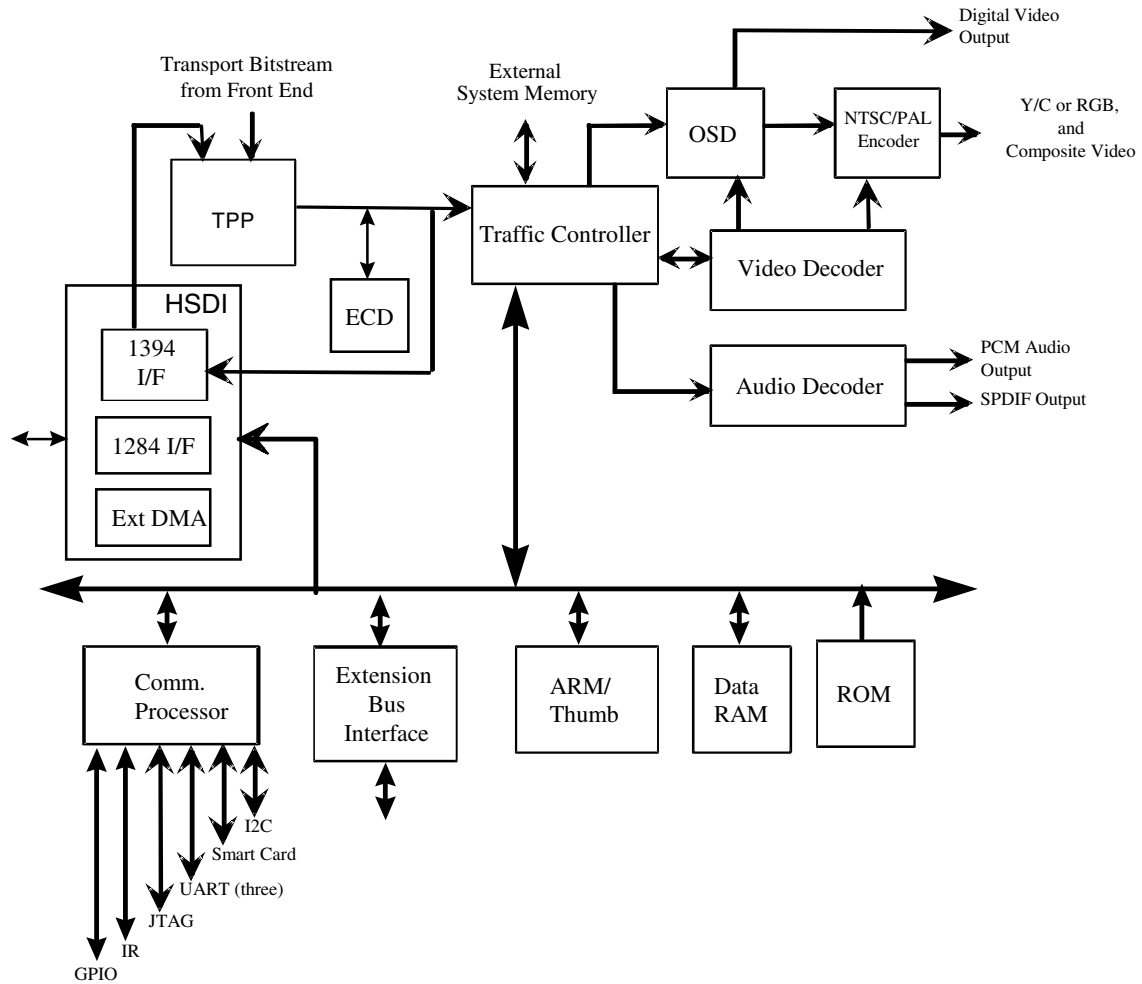


Figure 1. The 'AV7110 Block Diagram

#### 3.2 Overview

Figure 2 shows the overall memory map for the 'AV7110.

Comments	Address	Content	General Description
	FFFF FFFF	NOT USED	
Unprotected but with some => Supervisor only access sub-regions	CE00 0000	SDRAM (4 MBytes)	* See SDRAM Memory Map Section
	CDFF FFFF		
	CC00 0000	NOT USED	
	CBFF FFFF	NOT USED	
	A200 0000	Reserved	On Chip SRAM
	A1FF FFFF		
	A000 0C00	SRAM (CPU)	
FIQ/BIT access only (protected) =>	A000 0BFF		
	A000 0000	Reserved	
	9FFF FFFF		
	9E00 0D34	SRAM (HW)	
FIQ/BIT access only (protected) =>	9E00 0D33		
	9E00 0000	NOT USED	
	9DFF FFFF		
	7400 0000	NTSC/PAL Encoder	Hardware Registers & Memory Region  (See Register descriptions in Section 16)
Supervisor access only (protected) =>	73FF FFFF		
	7200 0000	BitBit	
Supervisor access only (protected) =>	71FF FFFF		
	7000 0000	CCP	
Supervisor access only (protected) =>	6FFF FFFF		
	6E00 0000	Audio Decoder	
FIQ/BIT access only (protected) =>	6DFF FFFF		
	6C00 0000	Video Decoder	
FIQ/BIT access only (protected) =>	6BFF FFFF		
	6A00 0000	OSD	
Supervisor access only (protected) =>	69FF FFFF		
	6800 0000	TC	
FIQ/BIT access only (protected) =>	67FF FFFF		
	6600 0000	TPP	
FIQ/BIT access only (protected) =>	65FF FFFF		
	6400 0000	HDSI	
FIQ/BIT access only (protected - but with => some User accessible sub-regions)	63FF FFFF		
	6200 0000	NOT USED	
	61FF FFFF		
	3A00 0000	CS6	EBI Space (Section 12)
USER Access (un-protected) =>	39FF FFFF		
	3800 0000	CS5	
USER Access (un-protected) =>	37FF FFFF		
	3600 0000	CS4	
USER Access (un-protected) =>	35FF FFFF		
	3400 0000	CS3	
USER Access (un-protected) =>	33FF FFFF		
	3200 0000	CS2	
USER Access (un-protected) =>	31FF FFFF		
	3000 0000	DRAM	
USER Access (un-protected) =>	2FFF FFFF		
	2E00 0000	CS1	
USER Access (un-protected - but some => Supervisor only sub-regions and no 8-bit write access)	2DFF FFFF		
	2C00 0000	NOT USED	
	2BFF FFFF		
	0200 0000	NOT USED	

**Figure 2. The 'AV7110 Memory Map**

### 3.2.1 Transport bit-stream processing

The 'AV7110 accepts a DVB transport bit-stream from the output of a Forward Error Correction (FEC) device with a maximum (burst) throughput of 72.8 Mbits/s or 9.1 Mbytes/s (60Mbps average over one transport packet). The Transport Packet Parser (TPP) in the 'AV7110 processes the header of each packet and determines whether the packet should be discarded, further processed by ARM CPU, or if the packet only contains relevant data and needs to be stored without intervention from the ARM. The TPP sends all packets requiring further processing or containing relevant data through the Traffic Controller (TC) to the internal RAM. The TPP also activates or deactivates the decryption engine (ECD) based on the content of the individual packet. The conditional access keys are stored in a local key table and managed by special firmware running on the ARM CPU. The data transfer from TPP to the Data RAM is done via DMA.

Further processing on the packet is done by the ARM firmware, which is activated by interrupt from the TPP after the completion of the packet data transfer. Two types of transport packets are stored in the Data RAM and managed as a FIFO. One is for pure data which is routed to SDRAM without intervention from the ARM, and the other is for packets that need further processing. Within the interrupt service routine, the ARM checks the FIFO for packets that need further processing, performs necessary parsing, removes the header portion, and establishes DMA for transferring payload data from the Data RAM to the SDRAM. The TC re-packs the data and gets rid of the voids created by the header removal.

Together with the ARM, the TPP also handles Program Clock Reference (PCR) recovery with an external VCXO. The TPP will latch and transfer to the ARM its internal system clock upon the arrival of any packet which contains a PCR. After further processing on the packet and identifying the system clock, the ARM calculates the difference between the system clock from a bit-stream and the actual system clock at the time the packet arrives. Then, the ARM filters the difference and sends it through an 8-bit Sigma-Delta DAC to control the external VCXO. The ARM will drive the VCXO to its center frequency during start-up or, if enabled by an API, after channel change when there is no incoming PCR.

The TPP hardware is capable of detecting packets lost from the transport stream. With error concealment by the audio and the video decoders the 'AV7110 minimizes the effect of lost data.

### 3.2.2 Audio and Video data processing

After removing packet headers and other system related information, both audio and video data is stored in the external SDRAM. The video and audio decoders then read the bit-streams from the SDRAM and process them according to the ISO standards. The chip decodes MPEG-1 and MPEG-2 main profile at main level for video and Layer I and II MPEG-1 and MPEG-2 stereo for audio. This includes the abnormal cases such as discontinuity of reference clock or packet continuity counters as well as splicing. Both Video and Audio decoders synchronize their presentation of reconstructed data using the transmitted Presentation Time Stamps (PTS) and their local system clock. The local system clock is continuously updated by the ARM.

## 3.3 Software System Overview

Software on the 'AV7110 is divided into three sections: Firmware, API, and user application software. The first section, firmware, is masked into the internal ROM of the 'AV7110. This software is supplied by Texas Instruments and is used for low level control of the hardware and bit-stream de-multiplexing in the 'AV7110. The user application software is resident in external memory and contains all the software written by the application programmer. The user application software communicates with the firmware through Application Programming Interfaces (APIs). These APIs are contained in external ROM. API's are written by TI and provided in the form of a Run Time Support Library (RTSL).

The 'AV7110 firmware comprises a collection of interrupt (FIQ) service routines and supervisor mode run-time support programs. This software isolates the application software from any direct interaction with the hardware.

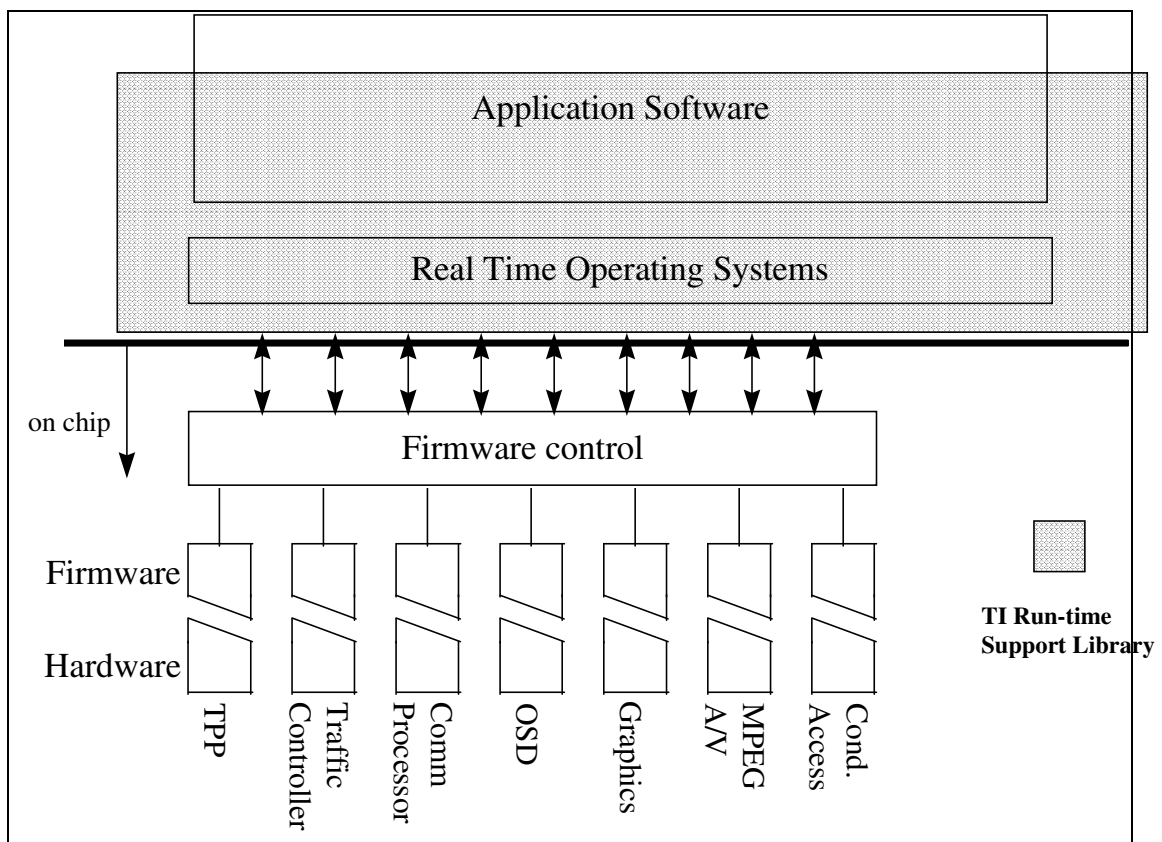
Figure 3 shows the high level block diagram of the complete software system. At the lowest level, each hardware component has a firmware module associated with it. A module may consist of:

- Initialization code - executed at processor reset
- Interrupt Service Routines - executed from interrupt by the associated hardware module

The run-time support library provides an application programming interface for the user application software. All run-time modules are written to be invoked from supervisory mode. This prevents software running in user mode from interfering with the built-in firmware.

Functionally, the firmware consists of several software processes that are self contained at runtime and operate for the most part, independently and asynchronously. Because of the nature of the processing of a complex input stream, the processes are interrupt driven.

The interrupt priorities determine the processing sequence, and the de-multiplexed input stream packet information determines the data path through the system. There is only minimal inter-process dependency, which eliminates the need for a software control-flow process. Any inter-process dependencies can be handled by proper structuring of the interrupt process in terms of priorities and multi-level processing.



**Figure 3. Software block diagram**

## 4. ARM CPU

### 4.1 Features

- Runs at 40.5 MHz
- Supports byte (8-bit), half-word (16-bit), and word (32-bit) data types
- Reads instructions from on-chip ROM or from the Extension Bus
- Can switch between ARM (32-bit) or Thumb (16-bit) instruction mode
- 32-bit data and 32-bit address lines
- 7 processing modes
- Two interrupts, FIQ and IRQ

The 32-bit ARM processor running at 40.5 MHz and its associated firmware provide the following:

- initialization and management of all hardware modules
- service for selected interrupts generated by hardware modules and I/O ports
- application program interface (API) for users to develop their own applications

The on-chip SRAM (also referred to as Internal Data RAM in this document) provides the space necessary for the 'AV7110 to properly decode transport bit-streams without losing any packets. The run-time support library (RTSL) and all user application software are located outside the 'AV7110. Details of the firmware and RTSL are provided in a separate software specification document.

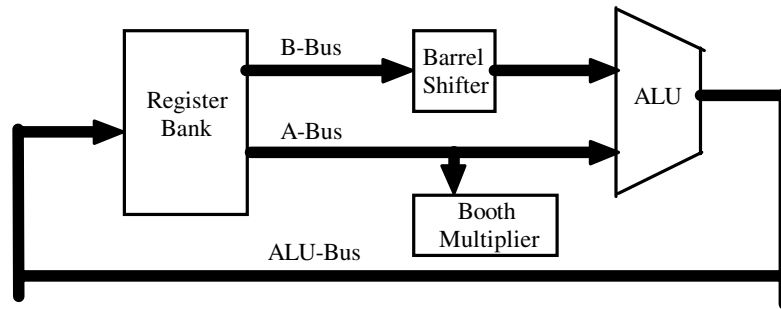
The CPU in the 'AV7110 is a 32-bit RISC processor, the ARM7TDMI/Thumb, which has the capability to execute instructions in 16- or 32-bit format at a clock frequency of 40.5 MHz. The regular ARM instructions are exactly one word (32-bit) long, and the data operations are only performed on word quantities. The LOAD and STORE instructions however, can transfer either byte, half-word or word quantities.

The Thumb uses the same 32-bit architecture with a 16-bit instruction set. That is, it retains the 32-bit performance but reduces the code size with 16-bit instructions. With 16-bit instruction, the Thumb still gives 70 - 80% of the performance of the ARM running ARM instructions from 32-bit memory. ARM and Thumb are used interchangeably, throughout this document.

### 4.2 Block Diagram

ARM uses a LOAD and STORE architecture; i.e. all operations are on the registers. ARM has 7 different processing modes with 16, 32-bit registers visible in user mode. In the Thumb state, there are only 8 registers available in user mode. The high registers may still be accessed through special instructions in this case. The instruction pipeline is three stage, fetch → decode → execute, and most instructions only take one cycle to execute. Figure 4 shows the data path of ARM processor core.





**Figure 4. ARM Core Data Path**

### **4.3 Resource management**

The ARM CPU is responsible for managing all the hardware and software resources in the 'AV7110. At power up the ARM verifies the existence and size of external memory. Following that, it initializes all the hardware modules by setting up control registers and tables and then resetting data pointers. It then executes the default firmware from internal ROM and transfers control to the application software. A set of run-time library routines provide the access to the firmware and hardware for user application programs. The application programs are stored in external memory attached to the Extension Bus.

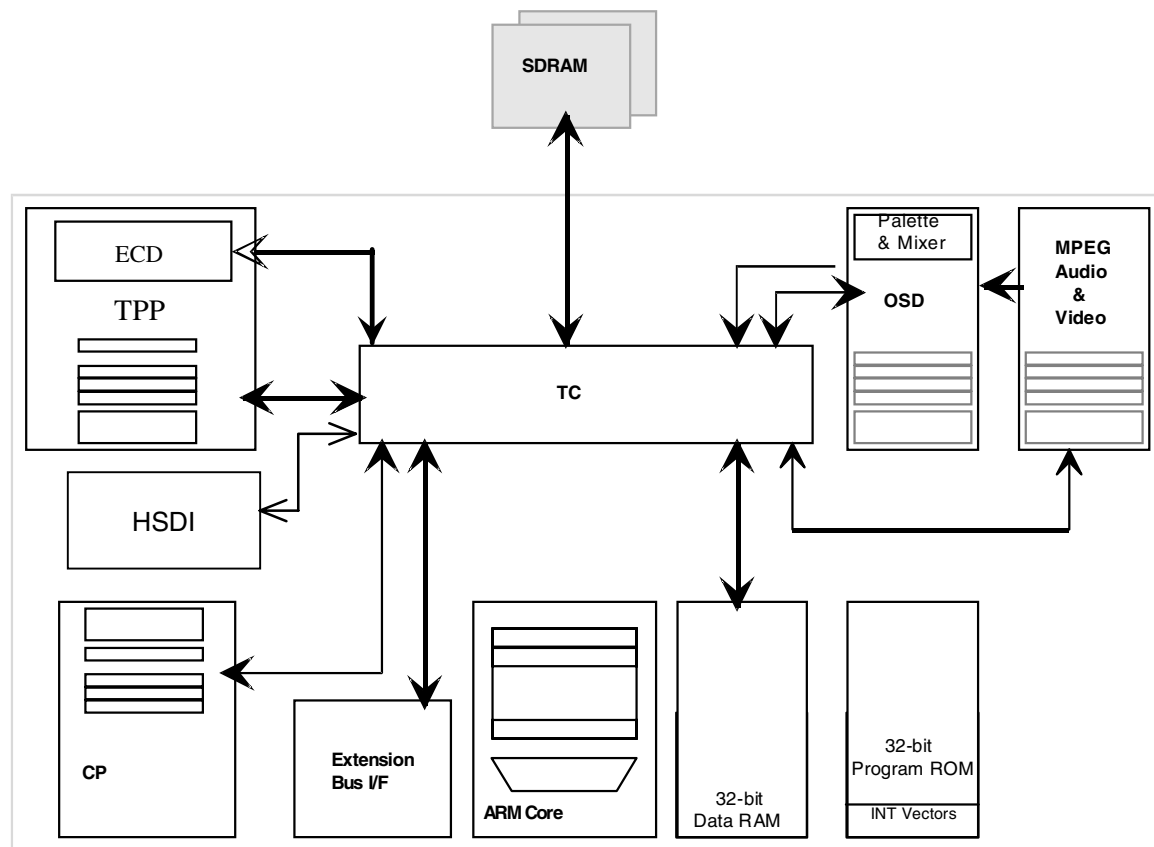
During normal operation, the ARM constantly responds, based on a programmable priority, to FIQ interrupt requests from any of the hardware modules and devices on the Extension Bus. The types of interrupt services include transport packet parsing, program clock recovery, traffic controller and OSD service requests, service or data transfer requests from the Extension Bus and Communication Processor, and service requests from the Audio/Video decoder.

## 5. Traffic Controller (TC)

### 5.1 Features

- Manages interrupt requests
- Authorizes and manages DMA transfers
- Provides SDRAM interface
- Manages the Extension Bus
- Provides memory access protection
- Manages the data flow between processors and memories
  - TPP to / from internal Data RAM
  - Data RAM to/from Extension Bus
  - SDRAM to OSD
  - OSD to / from internal Data RAM
  - Audio / Video Decoder to / from SDRAM
  - SDRAM to / from internal Data RAM
  - High Speed Data Interface to / from internal Data RAM
- Generates chip selects (CS) for all internal modules and devices on the Extension Bus
- Generates programmable wait states for devices on the Extension Bus
- Provides 3 breakpoint registers and a 64x32-bit patch RAM space

Figure 5 depicts the data flow managed by the TC.



**Figure 5. Traffic Controller Data Flow**

## 5.2 SDRAM Interface

The SDRAM must be 16-bits wide. The 'AV7110 provides control signals for up to two SDRAMs. Any combination of 4 or 16 Mbit SDRAMs may be used, provided they total at least 16 Mbits. Other supported sizes and configurations are:

- 16 Mbit → one 16 Mbit SDRAM
- 20 Mbit → one 16 Mbit and one 4 Mbit SDRAM
- 32 Mbit → two 16 Mbit SDRAM

The SDRAM must operate at an 81 MHz clock frequency. After power up reset, the 'AV7110 issues a Mode Register Set (MRS) Cycle to configure the SDRAM. The format is fixed and automatic. It does this by holding SRAS, SCAS, SWE, A11 and A10 Low while placing the input mode word value on A9-A0 (SADDR[11:0] = 0000 0011 0111). This forces a Read Latency of 3, Serial operation and sets the burst length to the maximum. In choosing SDRAM, the parameters of Table 2 represent some of the critical requirements found to vary the most with SDRAM manufacturers. The 16 Mbit TI TMS626162 SDRAM is recommended for use with the 'AV7110 and meets these requirements.

**Table 2. SDRAM Selection Requirements**

PARAMETER (Note 1)		MIN	MAX	UNIT
t <sub>AC</sub>	Access time, CLK↑ to data out		10	ns
t <sub>RC</sub>	REFR command to ACTV, MRS, REFR, or SLFR command; ACTV command to ACTV, MRS, REFR, or SLFR command; Self-refresh exit to ACTV, MRS, REFR, or SLFR command	110		ns
t <sub>RAS</sub>	ACTV command to DEAC or DCAB command	72		ns
t <sub>RP</sub>	DEAC or DCAB command to ACTV, MRS, SLFR, or REFR command	40		ns
t <sub>RCD</sub>	ACTV command to READ or WRT command	35		ns
t <sub>RRD</sub>	ACTV command for one bank to ACTV command for the other bank	24		ns
t <sub>RWL</sub>	Final data in to DEAC or DCAB command	24		ns

Note 1: Command mnemonics

REFR	Auto refresh	DEAC	Bank deactivate (precharge)
ACTV	Bank activate	DCAB	Deactivate all banks
MRS	Mode-register set	READ	Column-address entry/read operation
SLFR	Self-refresh entry	WRT	Column-address entry/write operation

Timing diagrams for various cycles are given in Section 17.2. These represent the more detailed timing parameters for the SDRAM interface of the 'AV7110.

The access to the SDRAM can be by byte, half word, single word, continuous block, video line block, or 2D macroblock. The interface also supports decrement mode for bitBLT block transfers.

The two chip selects correspond to the following address ranges:

$$\overline{SCS1} \rightarrow 0xCC00\ 0000 - 0xCC1F\ FFFF$$

$$\overline{SCS2} \rightarrow 0xCC20\ 0000 - 0xCDFF\ FFFF$$

During DVB decoding, the 'AV7110 allocates the SDRAM for NTSC & PAL modes according to the default layouts shown in Figure 6 & Figure 7, respectively. Address pointers for the Audio Buffer, Video Buffer, B-Frame and primary OSD Space in the SDRAM partitioning (Shaded regions of Figure 6 and Figure 7) are configurable via API; hence, the actual layout can be customized. Note that more OSD space can be obtained by: (1) synchronizing the video decoder and the VSYNC signal thus reducing the Video Buffer, (2) storing the decoded B pictures in reduced resolution to reduce B-Frame storage (the figures show a 0.6 B Frame reduction), (3) using DRAM as an overflow portion of the video input buffer which would reduce the Video Buffer size. Refer to Section 7.7 for more details.

Address	Content	Size *	Comments
CDFF FFFF	Reserved		
CC40 0000	Expansion Memory Region (OSD/User Data)	2 MB	
CC3F FFFF			
CC20 0000	Audio Buffer	65,536 b	or 8 kB **
CC1F FFFF			
CC1F E000	Video Buffer	2,748,416 b	or 335.5 kB **
CC1F DFFF			
CC1A A200	OSD or other use	356,608 B	or 348.25 kB
CC1A A1FF			
CC15 3100	B Frame	311,040 B	or 303.75 kB
CC15 30FF			
CC10 7200		(0.6 Frame Size)	
CC10 71FF	Second Reference Frame	518,400 B	or 506.25 kB
CC08 8900	First Reference Frame	518,400 B	or 506.25 kB
CC08 88FF			
CC00 A000	Video Microcode	36,864 B	or 36 kB
CC00 9FFF			
CC00 1000	Tables and FIFOs	3 kB	* See Table 3 through Table 5
CC00 0FFF			
CC00 0400	Pointers	1 kB	
CC00 03FF			
CC00 0000			

\* B = Bytes, b = Bits

\*\* This is the default size and may vary based on the application

**Figure 6. Default Memory Allocation of SDRAM (NTSC)**

Address	Content	Size *	Comments
CDFF FFFF	Reserved		
CC40 0000	Expansion Memory Region (OSD/User Data)	2 MB	
CC3F FFFF			
CC20 0000	Audio Buffer	65,536 b or 8 kB	
CC1F FFFF			
CC1F E000	Video Buffer	2,748,416 b or 335.5 kB **	
CC1F DFFF			
CC1A A200	OSD or other use	87,040 B or 85 kB **	
CC19 C1FF			
CC19 4E00	B Frame	373,248 B or 364.5 kB	
CC19 4DFF			
CC13 9C00		(0.6 Frame Size)	
CC13 9BFF	Second Reference Frame	622,080 B or 607.5 kB	
CC0A 1E00	First Reference Frame	622,080 B or 607.5 kB	
CC0A 1DFF			
CC00 A000	Video Microcode	36,864 B or 36 kB	
CC00 9FFF			
CC00 1000	Tables and FIFOs	3 kB	* See Table 3 through Table 5
CC00 0FFF			
CC00 0400	Pointers	1 kB	
CC00 03FF			
CC00 0000			

\* B = Bytes, b = Bits

\*\* This is the default size and may vary based on the application

**Figure 7. Default Memory Allocation of SDRAM (PAL)**

The address range from 0xCC000000 to 0xCC000FFF contains decoded bitstream information, error statistics and monitor information, and Commands and Flags that provide a communication channel between the ARM CPU and the Video Decoder. The relative addresses for these locations are defined in Table 3 through Table 5.

**Table 3. Decoded Bitstream Information**

Byte Address	Name	Usage
<b>Sequence Header as defined in the ISO/IEC 13818-2</b>		
0x000C00	horizontal_size_value	
0x000C04	vertical_size_value	
0x000C08	aspect_ratio_information	
0x000C0C	frame_rate_code	
0x000C10	bit_rate_value	
0x000C18	vbv_buffer_size_value	
0x000C1C	constrained_parameters_flag	
0x000C20	load_intra_quantizer_matrix	
0x000C24	load_non_intra_quantizer_matrix	
<b>Sequence Extension as defined in the ISO/IEC 13818-2</b>		
0x000C60	profile_and_level_indication	

Byte Address	Name	Usage
0x000C64	progressive_sequence	
0x000C68	chroma_format	
0x000C6C	low_delay	
0x000C70	frame_rate_extension_n	
0x000C74	frame_rate_extension_d	
<b>Sequence Display Extension as defined in the ISO/IEC 13818-2</b>		
0x000C80	video_format	
0x000C84	colour_description	
0x000C88	colour_primaries	
0x000C8C	transfer_characteristics	
0x000C90	matrix_coefficients	
0x000C94	display_horizontal_size	
0x000C98	display_vertical_size	
<b>Group of Pictures Header as defined in the ISO/IEC 13818-2</b>		
0x000CA0	the first 12 bits of time_code	stores time_code_hours, time_code_minutes
0x000CA4	the rest 12 bits of time_code	stores time_code_seconds, time_code_pictures
0x000CA8	closed_gop and broken_link	
<b>Picture header as defined in the ISO/IEC 13818-2</b>		
0x000CC0	temporal_reference	
0x000CC4	vbv_delay	

**Table 4. Error Statistics and Monitor Information**

Byte Address	Name	Usage
0x000510	SCR_Monitor	SCR value at the time of picture decode
0x000514	RATE_BUF_RPTR	read pointer of VBV buffer at the time of picture decode
0x000518	RATE_BUF_WPTR	write pointer of VBV buffer at the time of picture decode
0x000544	TOTAL_SKIP_PICS	total skipped pictures since last channel switch or power up
0x000548	TOTAL_REPEAT_PICS	total repeated pictures since last channel switch or power up
0x000184	ERRORS_IN_SLICE	accumulated error in slice layer; reset at power up or channel change
0x000188	ERRORS_NOT_IN_SLICE	accumulated error outside of slice layer; reset at power up or channel change
0x000194	PIC_ERROR_COUNT	accumulated error within a picture; reset at each picture
0x000198	SEQ_ERROR_COUNT	accumulated error within a sequence; reset at each sequence
0x00019C	VLD_ERROR_COUNT	accumulated error in the variable length decode; reset at power up or channel change

**Table 5. Flags and Commands**

Byte Address	Name	Usage
0x000FA8	DISABLE_USER_SYNC	0: default mode; use PTS given by application for video synchronization 1: disable synchronization using PTS
0x000FC4	DISABLE_PAN_SCAN	0: default mode; apply Pan_Scan to 16:9 source video 1: disable Pan_Scan on 16:9 source video
0x000400	CMD_FIFO_START	FIFO for normal commands like Play, ½ Decimate etc...
0x0004FC	CMD_FIFO_END	End of Video Command FIFO
0x000500	HIGH_PRIORITY_CMD	command like Reset and NewChannel has higher priority and should use this location
0x000F00	CMD_FIFO_RPTR	read pointer of Command FIFO
0x000F04	CMD_FIFO_WPTR	write pointer of Command FIFO

Byte Address	Name	Usage
0x000FA0	DECIMATION_CMD	Designated location for decimation commands

### 5.3 Prioritized Interrupts

Interrupt requests are generated from internal modules like the TPP, OSD, A/V decoder, Communication Processor, and devices on the Extension Bus. Some of the requests are for data transfers to internal RAM, whereas others are true interrupts to the ARM CPU. The TC handles data transfers, and the ARM provides services to true interrupts. The interrupts are grouped into FIQs and IRQs. The firmware uses FIQs; the application software uses IRQs. The priorities for FIQs are managed by the firmware while those of IRQs are managed by the application software.

### 5.4 Managing DMA transfer

The TC in the 'AV7110 provides DMA capability to facilitate large block transfers between memories and buffers. The burst length (in multiple of 4 bytes) of the DMAs are independently selectable via 6-bit registers with an API. This provides a burst value from 4 to 188 bytes.

The SDRAM is used to store system-level tables, video and audio bit streams, reconstructed video images, OSD data, video decoding codes, tables, and FIFOs. The internal Data RAM stores temporary buffers, OSD window attributes, keys for conditional access, and other tables and buffers for firmware.

The TC manages two types of DMA transfers, but only the General Purpose DMA (GPDMA) is accessible to the application software. DMAs initiated by the TPP, the video decoder, the audio decoder, and the OSD module are inaccessible. The GPDMA includes ARM-generated and bitBLT-generated DMAs. The TC can accept up to 4 GPDMA at any given time. DMA from the Data RAM to the Extension Bus (and vice versa) is byte aligned. Table 6 describes the allowable GPDMA transfers.

**Table 6. GPDMA Sources and Destinations**

DMA Transfers Allowed by Hardware				
	SDRAM	Data RAM	Extension Bus	HSDI
SDRAM	NO	YES	NO	NO
Data RAM	YES	NO	YES	YES
Extension Bus	NO	YES	NO	NO
HSDI	NO	YES	NO	NO

Note that while there is no direct DMA transfer between the Extension Bus memories, HSDI or the SDRAM. The user can use two general DMAs and the internal Data RAM as an intermediate step for this process. Alternatively, an API is provided to transfer data between these interfaces. This is accomplished using this internal Data RAM and two DMAs.

### 5.5 Video/Audio Buffer Monitoring support

The TC continuously monitors the fullness of the video and audio input buffers. If overflow or underflow occurs, the firmware is alerted so that corrective action can be taken in a timely manner. Alternatively, the application software can periodically inspect (read access) the read/write pointers of the circular video and audio input buffers and take preemptive actions if overflow/underflow is deemed imminent.

Moreover, the TC also keeps the number of bytes of video data being sent to the video decoder in a wrap-around counter. This byte-count is used by the firmware for accurate video PTS synchronization.

## **5.6 Transfer of Video/Audio Data from the High Speed Data Interface (HSDI) to SDRAM**

With APIs, it is possible to use the general purpose DMA (GPDMA) on the 'AV7110 to feed audio/video data from the HSDI to the audio/video circular buffers in the SDRAM. Although it is not recommended, it is possible to do this from the application software as well. Given restricted traffic through the Transport Packet Parser (TPP, see Section 6.1), it is possible to transfer low-frequency Elementary Stream Video Data using the appropriate API. Likewise, similar Audio PES or Elementary Stream data can be transferred. No PTS processing can be done in this mode. Also, the application software manages data types coming into the HSDI. Management of the audio and video circular buffers will however be handled automatically by the TC hardware in this case.

When there is other TPP Audio/Video data traffic, the transfer mechanism is handled differently. When audio or video is played from the HSDI, the same type of data cannot be demultiplexed from the incoming DVB stream via the TPP hardware. The ARM sets up a transfer for the data from the HSDI via an API called from the application software.



## 6. MPEG Transport Decoder (TPP)

### 6.1 The TPP module

#### 6.1.1 Features

- Parses transport bit streams
- Accepts bit stream either from the transport parser input or from the 1394 interface
- Provides filtered decrypted or encrypted packets directly to the 1394 interface

Maximum Input bit rate (peak) Transport parser input .....	72.8Mbps
Maximum Input bit rate (average over one transport packet) Transport parser input .....	60Mbps
Maximum Input bit rate through the 1394 interface.....	64.8Mbps
Max Video bit rate .....	15Mbps
Max audio bit rate .....	1.13Mbps
Minimum time between two transport packets (providing input rate is less than average rate) .....	0 Sec
Maximum number of PIDs that can be filtered .....	32
Maximum number of PIDs that can be descrambled.....	32
Maximum number of pairs of keys for the descrambler.....	16
Recording mode through 1394 interface	
• full transport stream without descrambling	
• full transport stream with up to 32 PIDs descrambled	
• up to 32 descrambled PIDs	

#### 6.1.2 Description

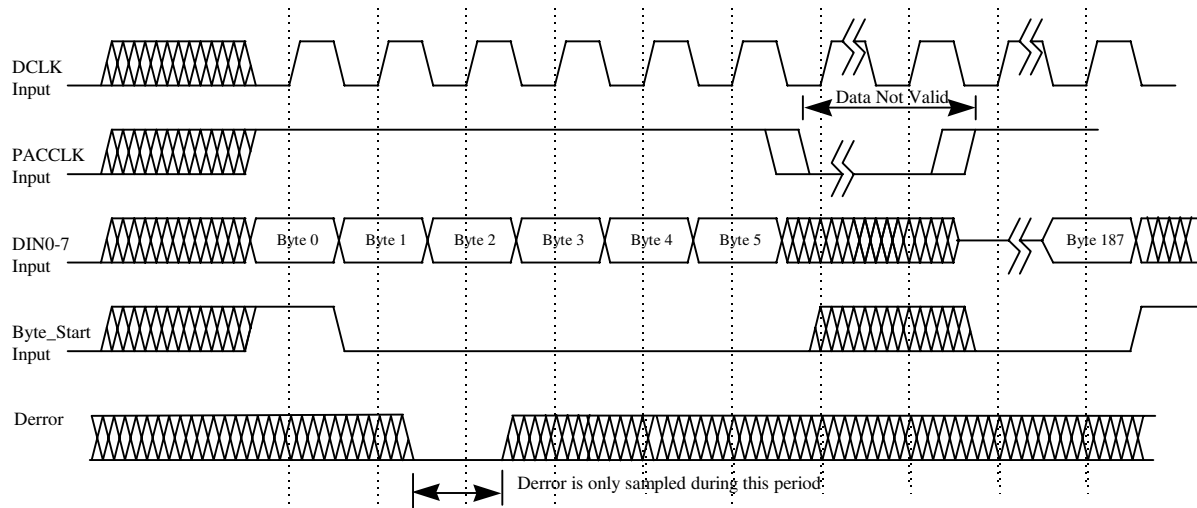
The Transport Packet Parser (TPP) in the 'AV7110 filters the header of each packet according to the PID and decides whether the packet should be discarded, further processed by the ARM CPU, or stored without intervention from the CPU. At start up, the TPP discards all data until the firmware enables reception. Individual encrypted channel data is then ignored until the first control word is received or until the firmware indicates that data is acceptable. The TPP can filter up to 32 different PIDs at any one time. All packets requiring further processing or containing relevant data are sent by the TPP to the internal Data RAM. The TPP also activates or deactivates the European Common Descrambler (ECD) hardware based on the content of individual packets or under the CPU control. Up to 16 pairs of Conditional Access (CA) keys are stored in the TPP. The data transfer from TPP to the internal Data RAM is done via DMA automatically.

Transport packets containing only audio or video payloads are automatically transferred to the proper external SDRAM memory buffer space via DMA. Other packets that require further processing are transferred to the CPU buffer, and an FIQ is issued to the CPU. Within the interrupt service routine, the firmware checks the FIFO that contains packets for further processing. It then performs the necessary parsing, removes the header portion, and establishes a DMA for transferring payload data from internal Data RAM to the appropriate destination.

The TPP also detects packets lost from the transport stream. Together with error concealment by the audio/video decoder, the TPP minimizes the effect of lost data.

Along with the CPU, the TPP handles Program Clock Reference (PCR) recovery to control the external VCXO. The TPP latches and transfers to the CPU its internal system clock upon the arrival of any packet containing a PCR. Firmware, through an interrupt handling routine (FIQ), performs minimal filtering of the PCRs to control the external 27MHz VCXO. Application software can perform more extensive filtering. Figure 8 shows an example circuit for the external VCXO. The output from the 'AV7110 is a pulse width modulated signal with a resolution of 256 levels.





**Figure 9. FEC Input Interface to TPP Timing**

DCLK is edge programmable; and PACCLK, BYTE\_START, and DERROR are level programmable. Figure 9 is shown with the default levels. The maximum burst bit rate for the transport parser interface is 72.8Mbps. A more detailed timing diagram is included in Section 17.7 for reference.

**Table 7. Transport packet Input interface pin description**

DATAIN[0..7]	Parallel data input
DCLK	Data clock. Runs at the rate at which bytes are sent to the ‘AV7110 on D0-D7. (max 9.1MHz) Programmable, edge triggered; default mode is rising edge triggered.
PACCLK	Packet clock. Level programmable, default is active high. Indicates valid data bytes on D0-D7. All bytes of a transport packet may be consecutive, in which case PACCLK is high for the whole duration of the transport packet. Certain clocking strategies adopted may require there to be gaps of one or more byte times between some consecutive bytes within and/or between transport packets. In this case PACCLK can go low for one or more byte times to indicate data bytes that should be ignored.
BYTE_START	Valid during the first byte of each transport packet on D0-D7. The edge of this signal is synchronous with that of DATAIN. Level programmable, default is active high.
DERROR	This pin is sampled by the ‘AV7110 during the 3rd byte of the header. The signal has the same meaning as the Transport Error bit in the transport packet header. If the transport enable error bit in the TPP status register is not set when this signal is detected, the ‘AV7110 discards the packet. During recording, such a packet is always be transferred to the 1394 port with the Transport Error bit in its header set to signal the error. This DERROR pin must be detected High at Byte 3 for an error to be detected.

### 6.2 Hardware Section Filtering

The hardware section filtering module comprises the following elements:

- 32 x 96 bit (12 byte) filters
- 32 x 96 bit (12 byte) mask
- 2 x 32 bit Match Data registers
- 2 x 32 bit Match Result Registers
- 1 x 32 bit Reset Control Register
- 1 x 32 bit Status register
- 1 x 32 bit Start Filter Enable
- 1 x 32 bit Stop Filter Enable

All comparisons, irrespective of the number of active filters, require 32 machine cycles with the result available in the 33<sup>rd</sup> cycle.

### 6.2.1 Memory Map

The hardware section filtering module uses the following quick reference memory mapped locations:

Name	Description
HW_FILT_CONTROL	Write of 1 to this register causes initialization of HW
HW_FILT_STATUS	Indicates result of the search, contains done, found bits and index of first match
MATCHDATA	Register to write 8 bytes of data to compare against filter values using masking
MATCH_RESULT_REG	Bitmapped index of matches of data in the filter set
HW_FILT_BASE	Starting address of the filter members (32 x 8 bytes)
HW_MASK_BASE	Starting address of the filter masks (32 x 8 bytes)

### 6.2.2 Filter hardware definition

The filter hardware consists of a set of tables containing the filter to compare against and the mask to apply to both the data to be compared and the filter. In addition the hardware engine contains an attribute word for each element of the filter table.

Filter values, filter masks and attributes are all related in the tables and are addressed by the same index in the respective tables as shown in Figure 10.

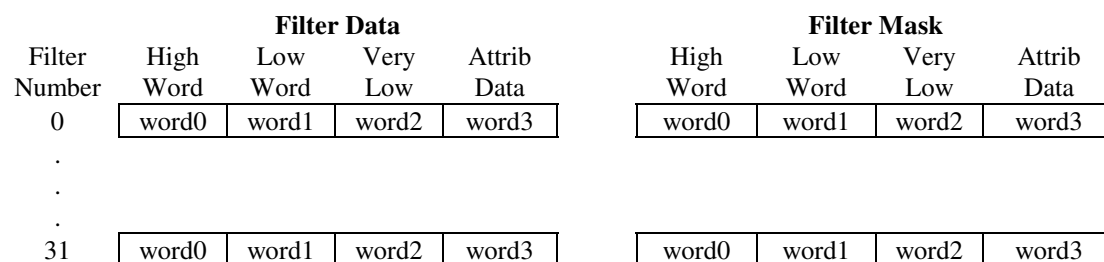


Figure 10: hardware filter layout

#### 6.2.2.1 Filter Data Memory Addressing

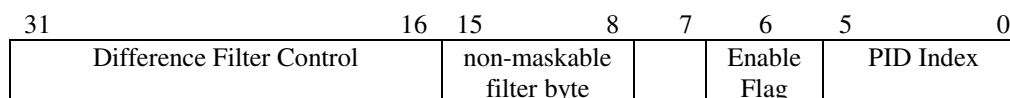
The 32 filter entries are comprised of three WORDs of filter data plus one WORD of attribute information.

Storage of the entries is memory mapped as follows. The starting word address of the filter is:

```
#define HW_FILT_BASE (ulong *)
```

The high word, or upper 4 bytes of the filter data, is stored in word 0. The remaining 4 bytes of the filter data are word 1. Word 2 contains the attribute data, right justified. The attribute data is the 6 bit PID field, 1 bit enable and a non-maskable filter. Word 3 is empty.

#### 6.2.2.2 Filter Attribute Bit Definition



[5:0] PID Index

Six bit index into the PID filter table (PID CAM) that allows the application to attach each filter entry to a given active PID. The PID index is 6 bits, or 64 entries. Currently, the AV7110 hardware only supports 32 separate PIDs. The additional PID indexes are available for potential future AV7XXX implementations using this filter.

[6] Enable Flag:

- '0' indicates filter is inactive. No comparisons are performed on the associated filter data.
- '1' indicates filter is enabled (active). Filtering will be performed on each valid section.

[15:8] Non-Maskable Filter Byte

Eight bit field that can be loaded with filter data. These eight bits cannot be masked.

[31:16] Difference filter control bits. When difference filtering is enabled for a byte, that byte is a match if any bits in the filter are different from the match register. Bit 31 enables difference filtering for byte 15, bit 30 enables byte 14, etc.

### 6.2.2.3 Filter Mask Memory Addressing

The 32 mask entries are memory mapped as follows. The starting word address of the filter is defined as:

```
#define HW_MASK_BASE (ulong *)
```

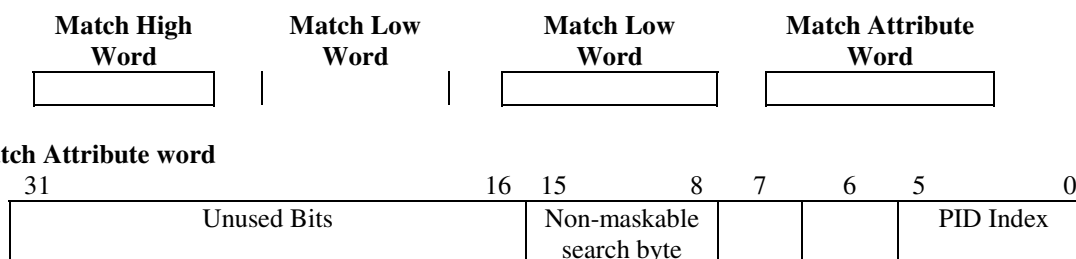
A mask entry corresponds to each filter. The mask is applied to both the filter memory words and the input match data words before they are compared. A '1' in the mask bit position indicates that the corresponding bits in the filter and input match data words should be compared. A '0' in the mask bit position indicates that the corresponding bits in the filter and input match data words are ignored for the comparison.

### 6.2.3 Filter Match Data Register definition

Two sets of input match data registers are available. The two sets of registers enable the search to be taking place at the same time that the next search data is readied. The switching of the Filter Match Data Register from one register to the other is automatic. After the Attribute word is written to the match data register, subsequent writes to the Filter Match Data Register will load the other register. The Match Data Attribute should be the last word written because a write to this register initiates a search.

The match data registers contain 12 bytes of data that is loaded by the CPU and an attribute match word. The match data is normally extracted from bytes 1,2 & 5-10 of the PSI/SI section header. Data extraction is performed by the firmware, from the Transport Packet stored into one of the internal SRAM TPP buffers.

The attribute is the value of the PID index of the Transport Packet, from which the section data is extracted. Bits 8 to 15 of the match attribute word contains the one byte of non-maskable search data.



[0:5] PID Index

Six bit index. The PID Index in the Match attribute is compared with the PID Index in the filter table to indicate which filter entries to perform the comparison.

[8:15] Non-maskable search data

These search bits will be compared against bits 8 to 15 in each filter attribute word. These 8 bits in the Match Attribute Word and the Filter Attribute Word are not maskable.

[6:7,16:31] not used, must be written as all 0's.

### 6.2.3.1 The Filter Match Data Register Memory Addressing

The two sets of registers enable the search to be taking place at the same time that the next search data is readied. After the Attribute word is written to the match data register, subsequent writes to the Filter Match Data Register will load the other register.

```
#define MATCHDATA (ulong *)
```

The matching operation performed by the hardware is as follows:

```
For i=1,32 {
MATCH[i] == if enable_bit[i] {
                (filter_PID_index[i] == match_data_PID_index)
                &((filter_data[i] & mask[i])==(match_data & mask[i])) }
```

Although one match data register is mapped into the address space of the CPU, in reality there are two of them operating in "flip-flop." While one is used by the matching hardware to compare the match data with each entry of the filter memory, the other is available to the CPU for loading new match data. The full comparison duration of 33 CPU cycles ( @ 40.5 MHz) can be "masked" by the actual loading of the data.

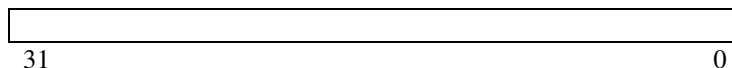
The software must ensure that a second search is not started by writing to the Match Data Attribute Register until the current search is done. The filter status word contains the done flag.

## 6.2.4 Match Result Register Definition

Depending on the type of filtering being performed it is possible to have either one or multiple filter matches for a given PID\_Index. Multiple matches are caused when the application generates overlapping definitions, or separate tasks request exactly the same filter data.

The result of the Match Register will be represented as a 32 bit result, where any matches in the search will be indicated by a '1' in the corresponding bit position. For example, if Filter 0 matched, bit 0 of the result register is set to a '1' (0x0000 0001).

The bit map of the match register are as follows:



### 6.2.4.1 Match Result Memory Addressing

The Match Result register is mapped as:

```
#define MATCH_RESULT (ulong *)
```

## 6.2.5 Status Register Word Definition

The status register can be polled to indicate the completion of a search. One bit is defined in the status register. The most significant byte always contains the index of the first filter match. This is specifically used for cached filters, where there will be ONLY one match per section. In this case the index will be read instead of the bit pattern stored in the Match Result Register.

When a search is initiated on an input match register, done is set to '0'. Upon the completion of the search, done is set to a '1'.

31	25	24	1	0
Filter Index	not used		found bit	done bit

[0] done bit. Indicates completion of a filtering operation.

[1] found bit. If '1', indicates that the Filter Index is valid. Also indicates that at least one bit in the result vector is set.

[2:24] not used.

[25:31] Contains the index of the first filter match found

### 6.2.5.1 Status Register Memory Addressing

```
#define HW_FILT_STATUS (ulong *)
```

### 6.2.6 Filter Reset Control Register

A filter reset is achieved by writing a '1' to bit 0 of the control register. This operation resets only the hardware filter circuitry. The hardware filter only needs to be reset in the event of a catastrophic system failure or software crash.

Bit 1 of the control register contains the "escape on first match" flag. When this control bit is set to '1', the filtering operations stops on the first match. The hardware does not go through the rest of the filter and search for further matches. The Status Register is valid after the first match. When the escape on first match bit is set to '0', all 32 locations of the filter are searched. In both cases, the Match Result register is valid.

Bit 2 resets the Start/Stop filter settings

Bit 3 controls the difference mode

'1' Local mode: use each filter attributes for difference byte control

'0' Global mode: use control register for difference byte control

31	16	3	2	1	0
global difference control		difference mode	start/stop reset	escape on first match	filter reset

#### 6.2.6.1 Filter Reset Control Register Memory Addressing

The filter reset control register is mapped as:

```
#define HW_FILT_CONTROL (ulong *)
```

The writing of a '1' to bit 0 pulses an internal reset. Following the reset, a write to the Match Data Attribute register will initiate a search..

### 6.2.7 Typical software process flow

The software algorithm for accessing the hardware filter needs to take in account the time it takes to complete the search (approx. 30+ cycles). Polling the completion bit in the status register is not acceptable at the firmware level, where processing of the end of packet (EOP) interrupt is extremely time critical. Completion of the interrupt process cannot extend over 25 microseconds in the general case. Therefore it is important to use the search time in the hardware filter to pre-load the second match to be done.

The parallel processing will continue until the last data to compare is loaded from the firmware. At that time the wait for the last completion can be used to pre-setup some of the DMA parameters. This parallelism allows a zero time loss in case the DMA needed to be done. In case the DMA was not needed, there is no processing penalty.

A typical payload processing sequence using this mechanism could be:

```

Reset Hardware                // Initialize the hardware filter state machine
Load section data            ; // load first data into the hardware filter
Write attributes;            // start the first compare
while( ptr <= payload end)   // Do for the whole payload
{
    move ptr to start of next section; // setup the next data
    load section data;           // load into the data compare register
    set next flag;              // don't start the search yet

    while (filter not complete) ; // wait for previous search to complete

    read result;                // see if match

    if (next flag)             // if new search pending
    {
        write attributes;       // start the hardware engine
        reset next flag;        // free up for next round
    }

    if(result == match)        // process the previous result here
        process result;         // setup DMA if needed
}
while (filter not complete) ; // wait for previous search to complete
read last result              // finish up the last search
process last result           // and process it

```

Note: The above algorithm is intended to be a functional example only.

## 6.2.8 CPU load assessment

### 6.2.8.1 Introduction

The CPU load for filtering of SI/PSI data will be depending on the following factors:

- Bit-rate of incoming data
- Number of sections per payload
- Number of channels to perform filter on

Sample of current broadcast requirements are:

- Maximum bit-rate = 5 Mb/s (3486 packets of 188 bytes each)
- Maximum 4 sections per payload
- Maximum 4 filters per PID
- Only 1 PID for carrying electronic program guide information

Expected increases in the above requirements in the (near) future:

- Maximum 6 sections per payload (including “start/stop” filters)
- Possible up to 8 PID carrying this type of data



In the Texas Instruments implementation of the hardware assisted filter, there is no time/CPU load difference between using 1 filter per PID or 32 filters per PID. Therefore in the assessment below the number of filters per PID will not be taken into account.

### 6.2.8.2 Assumptions

- Maximum rate of filter data does not exceed 5 Mb/s
- Assessments are based on 100% CPU load available for processing (High priority FIQ)

### 6.2.8.3 CPU load assessment

	Data in one PID only		Data in 8 PID	
	4 sections	6 sections	4 sections	6 sections
Packets/second	3324	3324	26596	26596
Processing time/packet	15 usec	25 usec	15 usec	25 usec
Processing time	49.86 msec	83.1 msec	398.94 msec	664.9 msec
% CPU	5.0	8.3	39.9	66.5

As is clear from the above information, at a load of 8 PID with 6 sections each, there is significant impact to the average available CPU for application use.

## 6.3 Hardware CRC circuit

A CRC-32 hardware module is available that allows an API to compute or verify checksums (for example, the CRC-32 in PSI/SI data). The CRC is calculated with the following polynomial:

$$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1.$$

Transfer of data from the DRAM to the hardware CRC circuit is transparent to the application software. At completion of the CRC processing, an IRQ interrupt can be issued if requested by an API call.

## 7. Video Decoder

### 7.1 Features

- Real-time decoding of MPEG-2 Main Profile@Main level and MPEG-1 video bit-streams
- Error detection and concealment
- Internal 90 KHz/27 MHz System Time Clock
- Sustained input rate of 15 Mbps
- Supports Trick Mode with full size trick mode picture
- Provides 1/4 and 1/16 decimated size picture
- Extracts Closed Caption and other picture user data from the bit-stream
- 3:2 pulldown in NTSC mode
- Provides read access to video input buffer's read/write pointers
- Pan-and-scan for 16:9 and 20:9 source material according to MPEG syntax
- Letterbox support
- High level command interface
- Synchronization using Presentation Time Stamps (PTS); also supports VBV delay based and free run (no synchronization) video playback
- Supports automatic upsampling of the display format with polyphase horizontal resampling and vertical chrominance filtering (see Table 8). Others can be done automatically by use of an API call.

**Table 8. Supported Video Resolutions for Automatic Up-Sampling**

NTSC (30 Hz)		PAL (25 HZ)	
Source	Display	Source	Display
720 x 480	720 x 480	720 x 576	720 x 576
704 x 480	720 x 480	704 x 576	720 x 576
544 x 480	720 x 480	544 x 576	720 x 576
480 x 480	720 x 480	480 x 576	720 x 576
352 x 480	720 x 480	352 x 576	720 x 576
352 x 240	720 x 480	352 x 288	720 x 576

### 7.2 Description

The Video Decoder module receives a video bit-stream from the SDRAM. It also uses the SDRAM as its working memory to store tables, buffers, and reconstructed images. The decoding process is controlled by a RISC engine which accepts high level commands from the ARM. In that fashion, the ARM is acting as an external host to initialize and control the Video Decoder module. The output video is sent to the OSD module for further blending with OSD data.

The 'AV7110 synchronizes the presentation of video with the audio using the transmitted Presentation Time Stamps (PTS) extracted by the ARM from the PES packets. It compares the PTS with the local system time clock (STC) and performs synchronization recovery if the difference is outside of a user programmable threshold range given as follows:

$$\text{STC} - \text{threshold} < \text{PTS} < \text{STC} + \text{threshold}$$

If synchronization recovery is needed, the video decoder either redisplay or skips a frame, depending on the PTS value. If the PTS lags, that is, the time for displaying the current picture has already passed, the video decoder discards the following B pictures without decoding them until the PTS catches up with the STC. If the PTS leads, that is, the time for displaying the current picture has not arrived yet, the video decoder pauses the decoding and continuously displays the last picture. See Section 11 for more details.

### 7.3 Trick Mode

When decoding a picture from a digital recorder, the decoder can handle trick modes (decode and display I frame only). The limitation is that the data has to be a whole picture instead of several intra-slices. Random bits are allowed in between trick mode pictures. It is important to note that if the random bits emulate a start code, unpredictable decoding and display errors will likely occur. The trick mode is activated by the scan command (see Table 9).

During trick mode decoding, the video decoder repeats the following steps:

- searches for a sequence header followed by an I picture
- ignores the video buffer under-flow error
- continuously displays the decoded I frame.

### 7.4 High-Level Commands

The Video decoder accepts the high-level commands detailed in Table 9.

**Table 9. Video Decoder Commands**

Play	Normal decoding
Freeze	Normal decoding but continue to display the last I or P picture (The choice of Top field only or Full Frame is dependent on the data in the stream)
Freeze1	Normal decoding but continue to display the last picture (Top field)
Freeze2	Normal decoding but continue to display the last picture (Full frame)
Stop	Stops the decoding process. The display continue with the last picture
Scan	Searches for the first I picture, decodes it, displays it, then continues searching for next I picture and repeats
FastForward	Decode and display only the I and P pictures.
SingleStep	Decode and display the next picture, then stop. Applicable only when user has control of the input bit-stream.
SlowMotion (N)	Decode pictures and display each picture N-frame times. Applicable only when user has control of the input bit-stream.
FindIndex (N)	Generate interrupt when N value is reached. N may be: PTS value, or time_code field in the GOP layer.
FreezeIndex (N)	Freeze on I picture at index value N. Index value is same as in FindIndex.
NewChannel	For channel change, the video decoder first finishes the current display picture. Then, based on settings provided through an API in user software, selects what to display: a pre-setup OSD window or blank background.
Reset	Halts execution of the current command. The bit-stream buffer is flushed and the video decoder performs an internal reset
Decimate1/2	Continue normal decoding and displaying of a 1/2 x 1/2 decimated picture (controlled by API)
Decimate1/4	Continue normal decoding and displaying of a 1/4 x 1/4 decimated picture (controlled by API)
TS_Display_On	Turn on true size display mode (controlled by API).
TS_Display_Off	Turn off true size display mode (controlled by API).



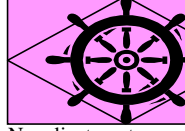


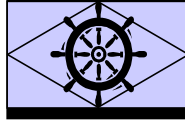
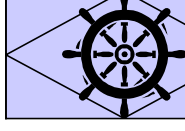



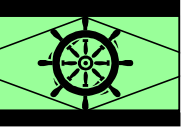

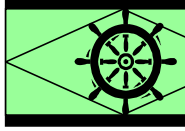
### 7.5 Video Decimation and Scaling

The video decoder contains a polyphase filter and a vertical interpolation filter for performing horizontal resampling and luma/chroma vertical interpolation to support decimation and scaling of pictures. In operation, the decoder first loads picture data from the video storage space in the SDRAM to the internal buffer of the video decoder. Vertical and horizontal filters are then applied to the video data to produce the 4:2:2 image.

The video decoder is capable of producing decimated pictures using 1/2 or 1/4 decimation per dimension, which results in reduced areas of 1/4 or 1/16. Decimation is achieved by using field data out of a frame, skipping lines, and performing vertical filter to smooth out the decimated image. However this decimation mode does not apply when the picture is displayed in the letterbox format. The

decimated picture can be viewed in real time. Such a decimated picture can be positioned anywhere on the screen by inserting it into an OSD window and positioning the window at the desired location.

The picture can also be scaled in the vertical and horizontal direction to allow for changes to the display format (see Figure 11). Various compression ratios in the vertical direction are supported to handle vertical scaling required for PAL (Table 10 and Table 11) and NTSC (Table 12 and Table 13).

Transmitted Format	4:3 Monitor	16:9 Monitor
 <p>4:3 Aspect Ratio</p>	 <p>No adjustment</p>	 <p>No adjustment</p>
 <p>16:9 Aspect Ratio</p>	 <p>Pan &amp; Scan</p>  <p>Vertical Compression</p> <p style="text-align: center;"><b>'AV7110</b> Display Formats</p>	 <p>No adjustment</p>
 <p>2.21:1 Aspect Ratio (PAL only)</p>	 <p>Pan &amp; Scan</p>  <p>Vertical Compression</p>  <p>Vertical Compression</p>	 <p>Pan &amp; Scan</p>  <p>Vertical Compression</p>

**Figure 11. Display Formats for the 'AV7110**

For full size pictures, the pan & scan mode is supported up to 1/16-pel resolution (*i.e.*, full-pel, 1/2-pel, 1/4-pel and 1/8-pel). In the decimated picture the pan & scan resolution is 1/8-pel for 1/2 decimated picture and 1/4-pel for 1/4 decimated picture. For non-full size pictures, the pan & scan mode is supported at full-pel resolution only.

**Table 10. Scaling Factors for 4:3 monitor for PAL format.**

4 : 3 Monitor		Source Luminance resolution Horz x vert.				
Source Aspect ratio	Display Type	720 x 576	544 x 576	480 x 576	352 x 576	352 x 288
4 : 3	Full Screen	H 1/1 V 1/1	H 4/3 V 1/1	H 3/2 V 1/1	H 2/1 V 1/1	H 2/1 V 2/1
16: 9	Pan & Scan	H 4/3 V 1/1	H 16/9 V 1/1	H 2/1 V 1/1	H 8/3 V 1/1	H 8/3 V 2/1
16 : 9	Letterbox	H 1/1 V 3/4	H 4/3 V 3/4	H 3/2 V 3/4	H 2/1 V 3/4	H 2/1 V 3/2
2.21:1	Letterbox	H 1/1 V 3/5	H 4/3 V 3/5	H 3/2 V 3/5		
2.21:1	Pan & Scan	H 5/3 V 1/1				
2.21 : 1	Letterbox and Pan and Scan	H 5/4 V 4/5	H 5/3 V 4/5	H 16/9 V 4/5		

**Table 11. Scaling Factors for 16:9 monitor for PAL format.**

16 : 9 Monitor		Source Luminance resolution Horz x vert.				
Source Aspect ratio	Display Type	720 x 576	544 x 576	480 x 576	352 x 576	352 x 288
4 : 3	Stretched	H 1/1 V 1/1	H 4/3 V 1/1	H 3/2 V 1/1	H 2/1 V 1/1	H 2/1 V 2/1
16: 9	Full Screen	H 1/1 V 1/1	H 4/3 V 1/1	H 3/2 V 1/1	H 2/1 V 1/1	H 2/1 V 2/1
2.21:1	Pan & Scan	H 5/4 V 1/1	H 5/3 V 1/1	H 16/9 V 1/1		
2.21 : 1	Letterbox	H 1/1 V 4/5	H 4/3 V 4/5	H 3/2 V 4/5		

**Table 12. Scaling Factors for 4:3 monitor for NTSC format.**

4 : 3 Monitor		Source Luminance resolution Horz x vert.				
Source Aspect ratio	Display Type	720 x 480	544 x 480	480 x 480	352 x 480	352 x 240
4 : 3	Full Screen	H 1/1 V 1/1	H 4/3 V 1/1	H 3/2 V 1/1	H 2/1 V 1/1	H 2/1 V 2/1
16: 9	Pan & Scan	H 4/3 V 1/1	H 16/9 V 1/1	H 2/1 V 1/1	H 8/3 V 1/1	H 8/3 V 2/1
16:9	Letterbox	H 1/1 V 3/4	H 4/3 V 3/4	H 3/2 V 3/4	H 2/1 V 3/4	

**Table 13. Scaling Factors for 16:9 monitor for NTSC format.**

16 : 9 Monitor		Source Luminance resolution Horz x vert.				
Source Aspect ratio	Display Type	720 x 480	544 x 480	480 x 480	352 x 480	352 x 240
4 : 3	Stretched	H 1/1 V 1/1	H 4/3 V 1/1	H 3/2 V 1/1	H 2/1 V 1/1	H 2/1 V 2/1
16: 9	Full Screen	H 1/1 V 1/1	H 4/3 V 1/1	H 3/2 V 1/1	H 2/1 V 1/1	H 2/1 V 2/1

## 7.6 Free-run and VBV-based Display Synchronization

Typically, display of decoded pictures is synchronized using the transmitted PTS and the internal system time clock (see Section 11 for more details). An API is also provided for the application software to instruct the video decoder on the 'AV7110 to display pictures as soon as possible (the free-run mode) or to wait for the duration specified in the VBV-delay field in each picture header before

decoding the picture. In the VBV-based display mode no synchronization is applied at display time. Note that the change in display synchronization mode takes effect only when a new channel is selected.

## 7.7 Memory Usage Reduction

In the minimum memory configuration, the 'AV7110 is designed to work with two memory devices: a single 16Mbit SDRAM device for system data storage (Audio, video data, etc.), and a 4Mbit DRAM for local data storage (private data, SI information etc.). Table 14 shows what the two memories are used for, in bytes. The amount of memory available for OSD applications depends upon the mode of operation of the 'AV7110. For example, if the letterbox display format is used, more memory is required to store B frames and hence less OSD space is available. Table 15 shows the size of the B frame buffer for different display formats at normal (full) resolution.

**Table 14. The 'AV7110 Memory Usage**

	SDRAM (bytes)	DRAM (bytes)
Video Input Buffer (theoretical + decoder delay + display synchronization = 229,376 + 52,000 + 75,000) - can be split between SDRAM and DRAM	107,000 to 356,376	249,376 to 0
Audio Lip-sync buffer	8,192	
MPEG I and P frames	1,244,160	
Video decoder micro-code	36,864	
Internal Tables FIFO's and Scratch space	4,096	
B frames	See Table 15	
User application, SI data private data etc.		393,216
OSD	See Table 16	

**Table 15. Storage of B frames**

Display Format	Normal resolution (1.0 B Frame = 622,080 bytes)	
	Portion of each B-Frame to be stored	Memory Required (bytes)
Full Screen	0.60	373,248
Letterbox 3/4 ratio	0.72	447,898
Letterbox 3/5 ratio	0.78	485,223

The amount of available OSD space can be significantly improved by reducing SDRAM space usage during video decoding. A combination of the following options is available: Vsync reset, and split Video Input Buffer.

### 7.7.1 Vsync Reset

The video input buffer, which is located in the SDRAM, is made up of three components: a theoretical rate control buffer whose size is specified in the MPEG-2 video standard (229,376 bytes); a buffer space to compensate for the decoder delay (52,000 bytes); and additional storage space to synchronize the video decoder timing with that of the NTSC/PAL encoder Vsync timing (75,000 bytes). On the 'AV7110, an API is provided to allow user applications to synchronize the video sync pulse to the video decoder timing at channel change. Doing so can potentially recover *up to* 75,000 bytes (15Mbps x 40msecs) of storage space for OSD usage. When Vsync reset is activated, the display during channel change is always blank.

## 7.7.2 Split Video Input Buffer

It is also possible to split the video input buffer between DRAM and SDRAM to increase the amount of SDRAM space available for OSD display. When this mode is selected, the DRAM portion is only used as an overflow buffer. When the allocated buffer space in the SDRAM is full, the incoming data is written directly to the DRAM. As soon as memory space is available in the SDRAM, the data is transferred from the DRAM to the SDRAM via DMA automatically. Once the overflow buffer in the DRAM has been emptied, incoming data is written directly to the SDRAM again. The video decoder always takes data from the SDRAM. The size of the buffers in each area of memory is under API control. At least 107 KBytes of the Video Input Buffer must be located in the SDRAM for the video decoder to operate properly. The minimum total VBV buffer size required from combined SDRAM and DRAM is 356,376 Bytes. The split should be changed only when the decoder is inactive to avoid loss of data. A dedicated DMA channel is used for the automatic transfer of data from the DRAM to SDRAM to implement this split buffer scheme. Table 16 shows the maximum amount of memory (in bytes) that can be made available for OSD applications when the video input buffer is split off to the DRAM (1Mbits and 1.5Mbits shown) on the extension bus.

**Table 16. Target OSD Memory Space in SDRAM (in Bytes)**

Decoder Options			Max OSD space with a 16Mbit SDRAM		
internal display sync	Video Input Buffer split		Letterbox 3/5 ratio 0.78 B frame	Letterbox 3/4 ratio 0.72 B frame	Normal Display 0.6 B frame
	SDRAM	DRAM			
NO	356,376	0	N/A	N/A	74,216
YES	356,376	0	37,241	74,566	149,216
YES	225,304	131,072	168,313	205,638	280,288
YES	159,768	196,608	233,849	271,174	345,824

## 7.8 Configuration of the Video Decoder

### 7.8.1 Configuration of video display format

The 'AV7110 processes video with aspect ratios of 4:3, 16:9, and 2.21:1. It also supports the display aspect ratio of 4:3 and 16:9. Figure 10 details the various display formats supported by the 'AV7110, and the application can select the specific format using API.

The following 3 flags, controlled by the API, are used to define the specific display format.

**16:9\_DISPLAY:** This flag can only be set during power up; because setting this flag causes an automatic hardware reset.  
0: 4:3 display device, default value  
1: 16:9 display device

**DIS\_PAN\_SCAN:** This flag can be set at any time and it takes effect at the next sequence start.  
0: [default value] Apply Pan\_Scan mode when the source video is 2.21:1, or when the source video is 16:9 and the display device is specified as 4:3.  
1: [Letterbox mode] Selects letterbox display when the source video is 16:9 and the display is 4:3 or when the source video is 2.21 and the display is 16:9. If the source video is 2.21 and the display is 4:3, the specific display depends on the flag LB\_PAN\_SCAN

**LB\_PAN\_SCAN:** This flag only applies to the case where the source is 2.21:1 and the output is 4:3  
0: letterbox display only, default value  
1: apply pan and scan followed by letterbox display

### 7.8.2 True Size Display Mode

The MPEG-2 bit stream contains two types of image size information; the encoded source image size and the active display size. Typically, these two sizes are equal. The size information specified by the

output display device is usually equal to the largest allowable source image, 720 pixels by 576 lines, as defined by the MPEG-2 main profile main level. In the AV7110, the application software must specify whether the display should follow active display size, the true size display mode, or the full screen output display size, the default mode. An API is available for setting the TS\_DISP flag to indicate true size display mode, the specified OSD window to host true size video, and the location of the true size display (X Y). The X value indicates left, right, or middle justification, whereas the Y value indicates top, middle, or bottom. The true size display API takes effect at the start of the next video sequence.

In full screen output display size, the default mode, the active display size information is ignored. The video decoder automatically adjusts the encoder source image size to the full screen of the display device. During the parsing of the active display size, the video decoder always checks the TS\_DISP flag. If it is set, then the following sequence of actions occurs:

- A. The video decoder compares the active display size against the previous display size; which normally is the full screen size. If they are different, issue a FIQ interrupt to the ARM with interrupt status register indicating that the active display size is different from previous display size. If they are the same continue decoding.
- B. The video FIQ interrupt service routine (ISR) reads the active display size from its fixed SDRAM location. If the active display size equals the full screen size, the ISR issues TS\_display\_off to the video decoder and disables the corresponding OSD window. If the size is different, then the ISR sets up an OSD window with the active display size and adjusts the window to the location specified by the (X, Y) pair defined above.
- C. The interrupt service routine then issues the command, TS\_display\_on, to the video decoder to start the True Size display mode.
- D. Upon receiving the command, the video decoder turns off the automatic size conversion process and only sends the number of pixels equivalent to the active display size to the OSD window starting from the next available picture. The video decoder also converts aspect ratio, if needed, with the active display size as the target size.

### 7.8.3 Video display synchronization mode

A 2-bit flag, SYNC\_MODE, is used to specify the synchronization method for the video decoder. This flag can be set by API as part of the channel switch initialization process. The meaning of this flag is defined as follows:

SYNC\_MODE Set by API during channel change

- 00: default value, apply PTS based synchronization
- 01: free run, disable all synchronization
- 11: synchronization based on VBV delay of each picture and disable PTS synchronization
- 10: reserved

### 7.8.4 Vsync reset and split VBV buffer

The VSYNC\_RESET and the VBV\_SPLIT flags control the on or off status of vsync reset and split VBV buffer modes, respectively. These two flags are specified by APIs during the hardware reset initialization. The value "1" indicates the mode is ON, and the value "0" indicates OFF. The default of these flags is "0". When the VSYNC\_RESET is ON, the screen always goes blank for 2 or 3 frames during channel change. If the VBV\_SPLIT mode is selected, the same API should also provide the size allocated in the SDRAM, which should not be smaller than 107 Kbytes.



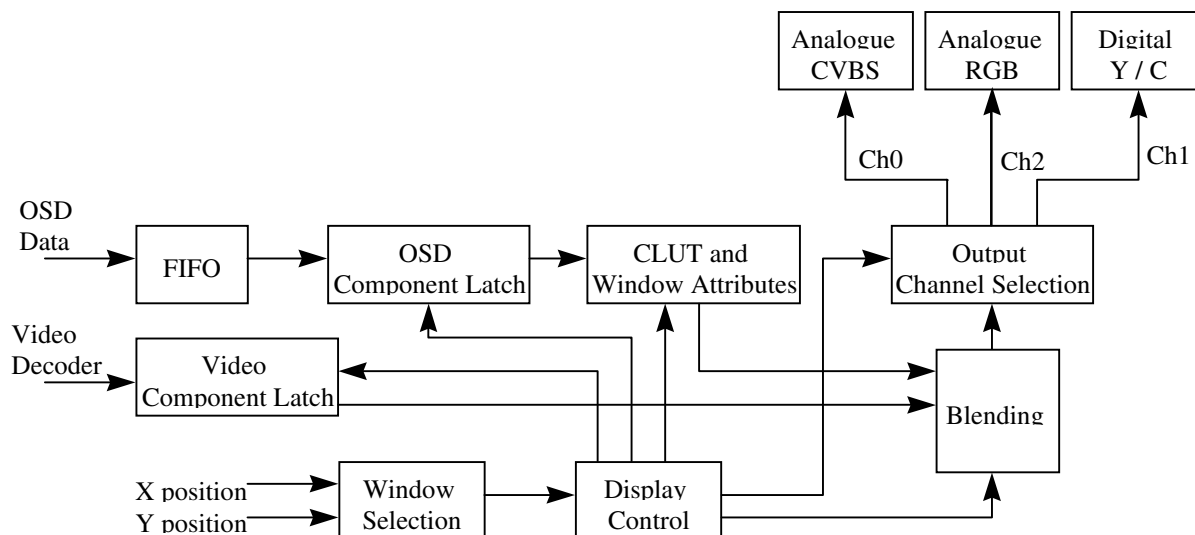
## 8. OSD & Graphics Acceleration

### 8.1 OSD

- Supports up to eight hardware windows, one of which can be used for a cursor
- Displays the nonoverlapped windows simultaneously
- Displays overlapped windows obstructively with the highest priority window on top
- Provides a hardware window-based cursor with programmable size, shape, color (two colors) and blinking frequency
- Provides a programmable background color, which defaults to black
- Supports four window formats:
  - empty window for decimated video
  - bitmap
  - YCrCb 4:4:4 graphics component
  - YCrCb 4:2:2 CCIR 601 component
- Supports blending of bitmap, YCrCb 4:4:4, or YCrCb 4:2:2 with motion video and with an empty window
- Supports window mode and color mode blending
- Provides a programmable, Color Look Up table (CLUT) with 256 entries.
- Outputs motion video or mixture with OSD in a programmable, 4:2:2 digital component format
- Provides motion video or mixture with OSD to the on-chip NTSC/PAL encoder
- Each hardware window has the following attributes:
  - window position: any even pixel horizontal position on screen; windows with decimated video have to start from an even numbered video line also
  - window size: from 2 to 720 pixel wide (even values only) and from 1 to 576 lines
  - window base address
  - data format: bitmap, YCrCb 4:4:4, YCrCb 4:2:2, and empty
  - bitmap resolution: 1, 2, 4, and 8 bits per pixel
  - full or half resolution for bitmap and YCrCb 4:4:4 windows
  - bitmap color palette base address
  - blend enable flag
  - 4 or 16 levels of blending
  - transparency enable flag for YCrCb 4:4:4 and YCrCb 4:2:2
  - output channel control
- Provides graphics acceleration capability with bitBLT hardware

#### 8.1.1 Description

The OSD module handles OSD data from different OSD windows and blends it with the video. It accepts video from the Video Decoder, reads OSD data from SDRAM, and produces three sets of video output: two go to the on-chip PAL/NTSC Encoder and another set to the digital output that goes off chip. Contents of these three sets of video output can be individually chosen. The OSD module defaults to standby mode, in which it simply sends video from the Video Decoder to all outputs. After being configured and activated by the ARM CPU, the OSD module reads OSD data and mixes it with the video output. The CPU is responsible for turning on and off OSD operations. The BitBlt hardware which is attached to the OSD module provides acceleration for memory block moves and graphics operations. Figure 12 shows the block diagram of the OSD module. The various functions of the OSD are described in the following subsections.



**Figure 12. OSD Module Block Diagram**

### 8.1.2 OSD data storage

The OSD data has variable size. In the bitmap mode, each pixel can be 1, 2, 4, or 8 bits wide. In the graphics YCrCb 4:4:4 or CCIR 601 YCrCb 4:2:2 modes, it takes 8-bit per components, and the components are arranged according to 4:4:4 (Cb/Y/Cr/Cb/Y/Cr ...) or 4:2:2 (Cb/Y/Cr/Y ...) format. In the case where RGB graphics data needs to be used as OSD, the application should perform software conversion to Y/Cr/Cb before storing it. The OSD data is always packed into 32-bit words and left justified. Starting from the upper left corner of the OSD window, all data is packed into adjacent 32-bit words. The dedicated bitBLT hardware expedites the packing and unpacking of OSD data. This allows the ARM to access individual pixels using an internal shifter within the OSD module.

Table 17 shows the fraction of a full size OSD window (720 x 576) supported by the available SDRAM OSD space. This assumes that the decoded B picture is in normal display mode (non letterbox).

**Table 17. SDRAM OSD Window Size**

Decoder Options			Fraction of a full-size OSD window supported				
internal display sync	Video Input Buffer split		B storage	OSD window resolution (bits/pel)			
	SDRAM	DRAM		24	8	4	2
NO	356,376	0	Full-res	0.06	0.18	0.36	0.72
			Half-res	0.21	0.63	1.26	2.51
YES	356,376	0	Full-res	0.12	0.36	0.72	1.44
			Half-res	0.27	0.81	1.62	3.24
YES	225,304	131,072	Full-res	0.27	0.81	1.62	3.24
			Half-res	0.38	1.13	2.25	4.50
YES	159,768	196,608	Full-res	0.28	0.83	1.67	3.34
			Half-res	0.43	1.28	2.57	5.14

## 8.2 Setting Up an OSD Window

An OSD window is defined by its attributes. Besides storing OSD data for a window into SDRAM, the application program also needs to update window attributes and other setup in the OSD module as described in the following subsections.

### 8.2.1 CAM Memory

The CAM memory contains X and Y locations of the upper left and lower right corners of each window. The application program needs to set up the CAM and enable selected OSD windows. The

priority of each window is determined by its location in the CAM. That is, the lower address window always has higher priority. In order to swap the priority of windows, the ARM has to exchange the locations within the CAM.

### **8.2.2 Window Attribute Memory**

The OSD module keeps a local copy of window attributes. These attributes allow the OSD module to calculate the address for the OSD data, extract pixels of the proper size, control the blending factor, and select the output channel.

### **8.2.3 Color Look Up Table**

Before using bitmap OSD, the application program has to initialize the 256 entry color look up table (CLUT). The CLUT is mainly used to convert bitmap data into Y/Cr/Cb components. Since bitmap pixels can have either 1, 2, 4, or 8 bits, the whole CLUT can also be programmed to contain segments of smaller size tables, such as sixteen separate 16-entry CLUTs.

### **8.2.4 Blending and Transparency**

Bitmap or graphic 4:4:4 or 4:2:2 displays may be blended with an MPEG video display when they overlap the video display or with the background color when they do not overlap an MPEG window. If window blending is enabled, the amount of blending is determined by a blend factor. Table 18 indicates the supported blending levels.

Color blending on the pixel level is also supported. This feature is available for the bitmap displays only. If color blending is enabled, the amount of blending of each pixel is determined by the LSB of the chrominance components of the pixel itself: in other words, the blend factor of a pixel corresponds to the LSB of the Cb and Cr values that are stored in the CLUT entry corresponding to that pixel. As shown in Table 18, window blending supports 16 different levels, whereas color blending can support either 4 or 16 levels, according to the selected blending mode.

**Table 18. Blending Levels**

Blending Type	Window Blend Factor	Cb[0]	Cr[0]	Cb[1]	Cr[1]	OSD window contribution	MPEG Video contribution
NoBlend	don't care	don't care	don't care	don't care	don't care	1	0
ColorB4	don't care	0	0	don't care	don't care	1/4	3/4
ColorB4	don't care	0	1	don't care	don't care	1/2	1/2
ColorB4	don't care	1	0	don't care	don't care	3/4	1/4
ColorB4	don't care	1	1	don't care	don't care	1 (Opaque)	0
ColorB16	don't care	0	0	0	0	1/16	15/16
ColorB16	don't care	0	0	0	1	1/8	7/8
ColorB16	don't care	0	0	1	0	3/16	13/16
ColorB16	don't care	0	0	1	1	1/4	3/4
ColorB16	don't care	0	1	0	0	5/16	11/16
ColorB16	don't care	...	...	...	...	...	...
ColorB16	don't care	1	1	1	1	1 (Opaque)	0
WindowB16	0	don't care	don't care	don't care	don't care	1/16	15/16
WindowB16	$0 \leq N \leq 15$	don't care	don't care	don't care	don't care	$(N+1)/16$	$(15-N)/16$
WindowB16	15	don't care	don't care	don't care	don't care	1 (Opaque)	0

The hardware also supports a transparency mode with bitmap, graphic 4:4:4 or 4:2:2 displays. If transparency is enabled, then any pixel on the graphic or bitmap display that has a value of 0 allows video to be displayed. Essentially, 0 valued pixels are considered the transparent color, i.e. the MPEG video underneath or the background color will show through the bitmap. Table 19 shows the connection between transparency and blending on the same window.

**Table 19. Blending and transparency**

Transparency Enable	Blending Type	Window Blend Factor	OSD window contribution	MPEG Video contribution
0	NoBlend	don't care	1	0
0	ColorB4	don't care	depending on $Cb_0$ and $Cr_0$	depending on $Cb_0$ and $Cr_0$
0	ColorB16	don't care	depending on $Cb_0, Cr_0, Cb_1, Cr_1$	depending on $Cb_0, Cr_0, Cb_1, Cr_1$
0	WindowB16	$0 \rightarrow 15$	depending on blend factor	depending on blend factor
1	NoBlend	don't care	0 if pixel value = 0 1 if pixel value $\neq$ 0	1 if pixel value = 0 0 if pixel value $\neq$ 0
1	ColorB4	don't care	0 if pixel value = 0 else	1 if pixel value = 0 else

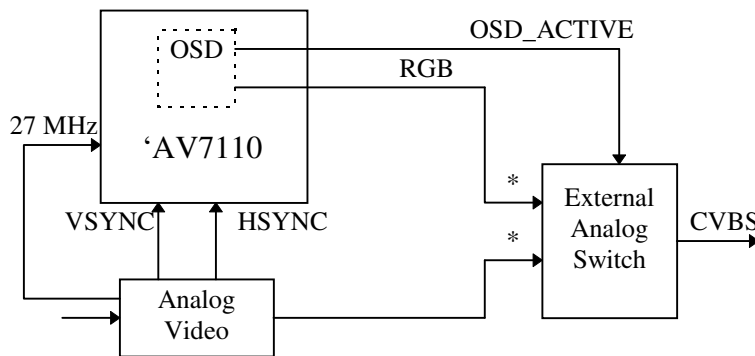
Transparency Enable	Blending Type	Window Blend Factor	OSD window contribution	MPEG Video contribution
			depending on Cb <sub>0</sub> and Cr <sub>0</sub>	depending on Cb <sub>0</sub> and Cr <sub>0</sub>
1	ColorB16	don't care	0 if pixel value = 0 else depending on Cb <sub>0</sub> ,Cr <sub>0</sub> ,Cb <sub>1</sub> ,Cr <sub>1</sub>	1 if pixel value = 0 else depending on Cb <sub>0</sub> ,Cr <sub>0</sub> ,Cb <sub>1</sub> ,Cr <sub>1</sub>
1	WindowB16	0→15	0 if pixel value = 0 else depending on blend factor	1 if pixel value = 0 else depending on blend factor

### 8.2.5 Hardware Cursor

A shape programmable cursor of up to a 32x24 pixels (if two colors; 64x24 pixels if single color) rectangular area is provided using hardware window 0. With window 0, the cursor always appears on top of other OSD Windows. The size, color (up to two), shape, and the blinking frequency of the cursor can be specified via APIs. Furthermore, APIs are also provided to enable vertical (*i.e.*, over the X-axis) mirroring and/or 2x horizontal stretching of the cursor, effectively creating a cursor of up to 64x48 pixels (if two colors; 128x48 pixels if single color). When hardware window 0 is designated as the cursor, only seven windows are available for the application. If a hardware cursor is not used, then the application can use window 0 as a regular hardware window.

### 8.2.6 Output Channels

When an OSD window is enabled, it has an attribute, Disp\_Ch\_Cntl[2:0], that defines the output channel on which the window is displayed. The signal OSD\_ACTIVE is asserted for the duration when an OSD window is being output on the RGB outputs. With an appropriate external delay element, this signal can be used to control an external analog switch to mix the OSD graphics generated by the 'AV7110 with external analog video source. The side effect of enabling OSD\_ACTIVE is that the CVBS will no longer be valid. Figure 13 shows the connections for synchronization and mixing of OSD with external analog video data.



\* The horizontal blanking delay of RGB (see Section 17.1 for timing) and Analog Video may be different. In that case the user needs to provide external delay to synchronize these two signals.

**Figure 13. Using the OSD\_ACTIVE Signal for External Analog Video**

The following table shows how to control the output channels:

**Table 20. OSD Module Output Channel Control**

Disp_Ch_Cntl [2:0]	Channel 2 (4:4:4) to Analog RGB matrix	Channel 1 (4:2:2) to Digital Video Output	Channel 0 (4:2:2) to PAL/NTSC Encoder
000	MPEG Video	MPEG Video	MPEG Video
001	MPEG Video	MPEG Video	Mixed OSD Window
010	MPEG Video	Mixed OSD Window	MPEG Video
011	MPEG Video	Mixed OSD Window	Mixed OSD Window
100	Mixed OSD Window	MPEG Video	MPEG Video
101	Mixed OSD Window	MPEG Video	Mixed OSD Window
110	Mixed OSD Window	Mixed OSD Window	MPEG Video
111	Mixed OSD Window	Mixed OSD Window	Mixed OSD Window

When windows are overlapping, it is not possible to view both complete windows on separate output channels. For example, Figure 14 shows the display modes which are allowed between any two channels (Channel 0 and Channel 1 in this example).

		Video Only	Full OSD picture	Non-overlapped OSD	Bottom of Overlapped OSD <sup>**</sup>
Ch0 NTSC/PAL Encoder Outputs					
Ch1 Digital video Output					
Video Only		YES	YES	YES	YES
Full OSD picture		YES	YES	YES	NO
Non-overlapped OSD		YES	YES	YES	NO
Top of Overlapped OSD		YES	YES	YES	NO

\*\* W2 has to be disabled in this case..

**Figure 14. OSD Output Channels Matrix**

**8.3 The BitBLT hardware**

The bitBLT hardware provides a faster way to move a block of memory from one space to the other. It reads data from a source location, performs shift/mask/merge/expand operations on the data, and finally writes it to a destination location. This hardware enables the following graphics functions:

- Set/Get Pixel
- Horizontal/Vertical Line Drawing
- Block Fill

- Font BitBLTing
- Bitmap/graphic BitBLTing
- Transparency

### 8.3.1 Sources and Destinations

The allowable source and destination memories for bitBLT are defined in Table 21.

**Table 21. Source and Destination Memories for BitBLT**

Source Memory	Destination Memory	
	SDRAM	Ext_Bus Memory
SDRAM	YES	YES
Ext_Bus Memory	YES	YES

### 8.3.2 Source and Destination Window Formats

The types of source and destination OSD windows supported by the bitBLT are the given in the following table (HR stands for half horizontal resolution).

**Table 22. Allowable BitBLT Window Formats**

Source OSD Window	Destination OSD Window				
	YCrCb 4:4:4	YCrCb 4:4:4_HR	YCrCb 4:2:2	Bitmap	Bitmap_HR
YCrCb 4:4:4	YES	YES	NO	NO	NO
YCrCb 4:4:4_HR	YES	YES	NO	NO	NO
YCrCb 4:2:2	NO	NO	YES	NO	NO
Bitmap	YES	YES	NO	YES	YES
Bitmap_HR	YES	YES	NO	YES	YES

Since the bitmap allows resolutions of 1, 2, 4, or 8 bits per pixel, the bitBLT will drop the most significant bits or pad with 0s when swapping between windows of different resolution. For half horizontal resolution OSD, the horizontal pixel dimension must be even numbers. For YCrCb 4:2:2 data, the drawing operation is always on 32-bit words, two adjacent pixels that align with the word boundary.

### 8.3.3 Transparency

In a block move operation, the block of data may also be made transparent to allow text or graphic overlay. The pixels of the source data are combined with the pixels of the destination data. When transparency is turned on and the value of the source pixel is non-zero, the pixel is written to the destination. When the value of the pixel is zero, the destination pixel remains unchanged. Transparency is only allowed from bitmap to bitmap, and from bitmap to YCrCb 4:4:4.

## 9. Video Output Interfaces

### 9.1 Analog Video Output - NTSC/PAL Encoder module

- Supports NTSC and PAL B, D, G/H, and I display formats
- Outputs Y/C or RGB, and Composite video with 9-bit DACs
- Generates 100/75 format color bar for PAL mode testing
- Complies to the RS170A standard
- Composite and RGB signals comply with ITU-R BT. 470-3 and ITU-R BT. 471-1
- Supports MacroVision Anti-taping function on composite video
- Provides sync signals with option to accept external sync signals

The composite video output can be either PAL or NTSC format. The default output format at powerup is PAL. Changing between NTSC and PAL mode can be done via an API which selects the output mode of the NTSC/PAL encoder. Note that the Video decoder microcode ROM is specific to NTSC or PAL and required for proper operation.

In addition to composite video, the 'AV7110 also provides either an analog S-video (Y - Luminance, C - Chrominance) signal or an analog RGB output with 720 pixel resolution. These signals share pins on the 'AV7110. All outputs conform to the RS170A standard. Table 23 shows the various combinations of signals available at the output pins. Selection of output is done via application accessible API.

**Table 23. Multiplexed Video Output Definitions**

Pin Name	Video Signal Output onto Pin			
	CASE 1	CASE 2	CASE 3	CASE 4
Y/COMP_OUT	Composite	Composite	Composite	Luminance (Y)
R/C_OUT	Red	Chrominance (C)	---	Chrominance (C)
G/Y_OUT	Green	Luminance (Y)	---	---
B_OUT	Blue	---	---	---

The video synchronization signal pins VSYNC and HSYNC on the 'AV7110 are digital outputs that default to tri-state mode at power up. Internally generated synchronization signals are used by the NTSC/PAL encoder. The user can select the source of video sync signals via an API. If internal source is selected then the VSYNC and HSYNC pins are configured as output pins. In RGB output mode, the horizontal blanking region is programmable via API and has a range with respect to the Composite output of  $\pm 81$  clock cycles ( $\pm 1 \mu\text{sec}$ ) from the default value.

MacroVision™ version 7.01 is enabled via an API; the default state is off. A version of the 'AV7110 where the MacroVision anti-taping circuitry is permanently disabled is also available.

### 9.2 Teletext and Wide Screen Signaling (WSS) Insertion (PAL mode only)

#### 9.2.1 Teletext insertion into PAL CVBS

The 'AV7110 supports the insertion of a Teletext signal onto PAL CVBS. Teletext data is transmitted in PES packets. The ARM is responsible for processing the PES packets and saving them in memory until they are needed by the OSD module.

- Teletext is inserted into TV lines 7-22 and 320-335 as specified in the 'line offset' field of the PES data field
- Teletext is only inserted into the analog CVBS signal.



- The clock period of the Teletext data should be  $144\text{nsec} \pm 5\%$
- Teletext transmission in DVB is covered by document ETS 300 472
  - The Teletext bit rate is  $6.9375\text{MHz} \pm 100\text{ppm}$ .
  - Teletext data starts at  $10.2 \mu\text{Sec}$  after the leading edge of the horizontal synchronization pulse
  - Teletext insertion conforms to the EBU Teletext spec ITU-R rec. 653 system B
- Each Teletext line contains 45 bytes of data
  - 16 bits of clock run in, generated in the 'AV7110
  - 43 bytes of data from the PES payload
- Teletext is only inserted in PAL mode

### 9.2.2 WSS insertion into CVBS

- WSS data is inserted into TV line 23 only
- WSS transmission is specified in ESTI standard 300 294, 'Television systems 625-line television wide screen signaling (WSS)
- WSS is only inserted in PAL mode.
- WSS is only inserted into analog CVBS
- WSS data clock frequency is  $5\text{MHz} \pm 5\%$

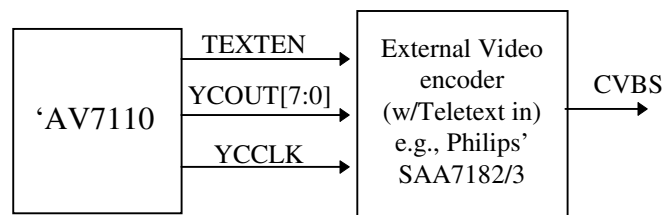
### 9.2.3 Insertion mode

An API is provided for the user to select if the Teletext signal is to be inserted internally or externally. WSS can only be inserted internally. Teletext data resolution in the composite signal for internal insertion and on the TEXTEN pin for external insertion is 27MHz.

**Internal insertion:** The Teletext and WSS data are inserted by the on-chip NTSC/PAL encoder into the analog CVBS signal.

**External insertion:** The Teletext signal is externally inserted via an external video encoder that has Teletext insertion capability. To allow this mode of operation, the Teletext signal is always output on the TEXTEN pin. The video signal on the digital out (YCOUT[7:0]) and the Teletext signal on TEXTEN are compliant with Philips™ SA7182/3 digital video encoder's input requirements.

Note that the external insertion option is not available if the 32-bit EBI mode is used due to pin multiplexing. Figure 15 shows external insertion of the Teletext signal.



**Figure 15. External Insertion of Teletext signal**

## 9.3 Closed Caption, Extended Data Services, and Video Aspect Ratio Identification Signal Insertion (NTSC mode only)

Closed Caption (CC) and Extended Data Services (EDS) are transmitted as picture layer user data. The video decoder extracts the CC and EDS information from the video bit-stream and sends it to the NTSC/PAL encoder module.

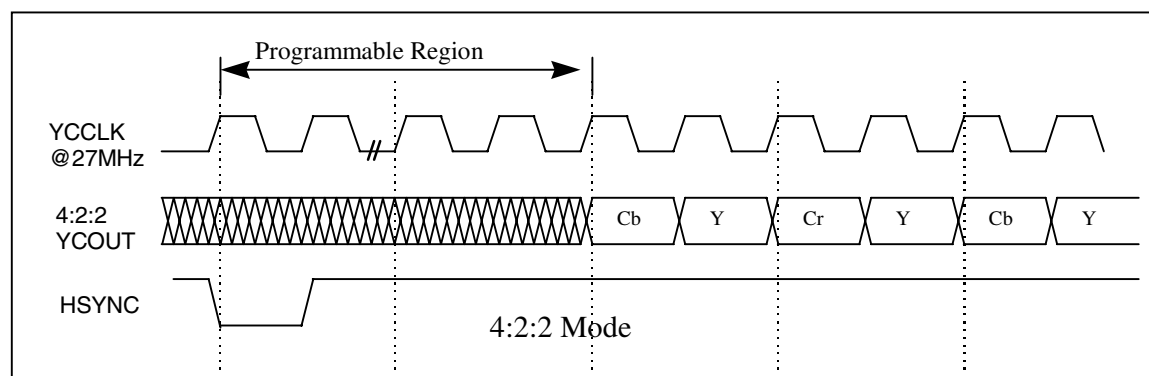
The video decoder also extracts the aspect ratio from the bit-stream and sends it to the ARM which prepares data according to the Video Aspect Ratio Identification Signal (VARIS) standard, EIAJ CPX - 1204. The ARM then sends the code to the NTSC/PAL encoder to insert CC data into the analog or digital video output.

The CC is inserted as ASCII code into the 21<sup>st</sup> video line; VARIS codes into the 20th video line for NTSC. An API is available to enable or disable the insertion.

## 9.4 Digital Video Output

NOTE: This interface is not supported when 32-bit EBI is used due to multiplexed signals.

A programmable blanking region is available which has a programmable range of +32/-31 YCCLK cycles from the default value (Default = 248 cycles for PAL) and is configurable via API. Figure 16 shows this region for 4:2:2 digital video output. Note that this programmability does not affect the Analog video output signals of the 'AV7110. For that the user must use the Analog video output programmable features described in section 9.1.



**Figure 16. Digital Video Output Timing**

### 9.4.1 PAL Mode Digital Video Output

The digital output is in 4:2:2 component format. The content of the video could be either pure video or the blended combination of video and OSD.

The pin assignments for the digital video output signals are:

YCOUT(8)	8-bit Cb/Y/Cr/Y data output
YCCLK(1)	27 MHz clock output

### 9.4.2 NTSC Mode Digital Video Output

The digital output includes video in either 4:4:4 or 4:2:2 component format, plus the aspect ratio VARIS code at the beginning of each video frame. The video output format is programmable by the user but defaults to 4:2:2. The content of the video could be either pure video or the blended combination of video and OSD.

The pin assignments for the digital video output signals are as follows:

YCOUT(8)	8-bit Cb/Y/Cr/Y and VARIS multiplexed data output
YCCLK(1)	27 MHz clock output
YCCTRL(2)	2-bit control signals to distinguish between Y/Cb/Cr components and VARIS code

The interpretation of YCCTRL is defined in the following table.

**Table 24. Digital Output Control**

SIGNALS	YCCTRL[1]	YCCTRL[0]
Component Y	0	0
Component Cb	0	1
Component Cr	1	0
VARIS code	1	1

The aspect ratio VARIS code includes 14 bits of data plus a 6-bit CRC, to make a total of 20 bits. In NTSC the 14-bit data is specified as shown in Table 25.

**Table 25. VARIS Code Specification**

	Bit Number	Contents
Word0 A	1	Communication aspect ratio: 1 = full mode (16:9), 0 = 4:3
	2	Picture display system: 1 = letter box, 0 = normal
	3	Not used
Word0 B	4 5 6	Identifying information for the picture and other signals (sound signals) that are related to the picture transmitted simultaneously
Word1	4-bit range	Identification code associated to Word0
Word2	4-bit range	Identification code associated to Word0 and other information

The 6-bit CRC is calculated, with the preset value to be 63, based on the equation

$$G(X) = X^6 + X + 1.$$

The 20-bit code is further packaged into 3 bytes according to the following format.

**Table 26. Three Byte VARIS Code**

	b7	b6	b5	b4	b3	b2	b1	b0
1st Byte	---	---	Word0 B			Word0 A		
2nd Byte	Word2				Word1			
3rd Byte	VID_ EN	---	CRC					

The three byte VARIS code is constructed by the ARM as part of the initialization process. The code is transmitted during the non-active video line starting from the 1st byte. The application can set the VID\_EN bit that signals the NTSC/PAL encoder to enable (1) or disable (0) the VARIS code. The value of the Aspect Ratio bit depends on the setup of 'AV7110 video output and the source bit stream as shown in the Table 27. The application can change the video output format using an software API call.

**Table 27. Coding of Aspect Ratio in the VARIS Code**

	Source 4:3	Source 16:9
Video Out 4:3	4:3	4:3
Video Out 16:9	4:3	16:9

The timing of the VARIS output based on a 27 MHz clock is shown in the timing diagrams in Section 17.1. The timing of both 4:4:4 and 4:2:2 digital video output formats is also described there.

## 10. Audio Decoder

### 10.1 Features

- Decodes MPEG audio layers I and II
- Supports all MPEG-1 and MPEG-2 data rates and sampling frequencies
- Provides automatic audio synchronization
- Supports 16- and 18-bit PCM data
- Outputs in both PCM and SPDIF formats
- Provides error concealment (by muting) for synchronization or bit errors
- Provides frame-by-frame status information
- Supports half-frequency modes
- Supports playback of 16-bit PCM data (PCM bypass), audio elementary stream, and audio PES
- Provides read/write accesses to audio input buffers' read/write pointers

The audio module receives MPEG compressed audio data from the TC, decodes it, and outputs audio samples in PCM format. APIs are provided for the ARM CPU to initialize/control the audio decoder via a control register and to read status information from the decoder's status register.

Audio frame data and PTS information are stored in the SDRAM in packet form. The audio module decodes the packets to extract the PTS and audio data. The audio decoder uses this PTS value to perform playback synchronization (see Section 11 for more details). The application software determines and enables the PID of the MPEG-1 compliant bit-stream in a MPEG-2 audio program which contains both the MPEG-1 compliant bit-stream and the MPEG-2 extension bit-stream.

The ARM can control the operation of the audio module via a 32-bit control register. The ARM may reset or mute the audio decoder, select the output precision and over-sampling ratio, and choose the output format for dual channel mode. The ARM can read status information from the audio module. One (32-bit) register provides the MPEG header information and sync, CRC, and PCM status.

The audio module has two 32-bit registers: a read/write control register and a read-only status register. The registers are defined in Table 28.

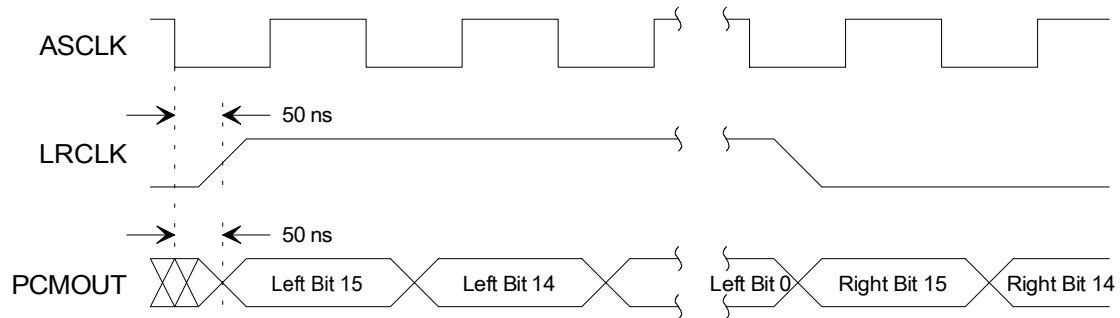
**Table 28. Audio Module Registers**

Register #	Location	Description
0 (Control Register - R/W)	31:22	Reserved (set to 0)
	21	Disable Sync 0 = Enable sync 1 = Disable sync
	20	Byte Swap 0 = No swap (bypass PCM data is in Little Endian) 1 = Swap bytes (bypass PCM data is in Big Endian)
	19	Elementary Stream Playback 0 = Off (normal operation) 1 = On, incoming data goes directly to audio decoder, bypassing the AFE
	18	PCM Bypass 0 = Disable bypass (normal operation) 1 = Enable bypass of AFE and audio decoder
	17	Copyright Bit Auto-Update Enable: 0 = Allows software to write to bit 16 1 = Bit 16 is updated from the MPEG header
	16	See bit 17 above. (this bit is output through SPDIF)
0 (cont.)	15:9	category (programmed by user, output through SPDIF)
	8	PCM Clock source select (PCMSRC) defaults to 0 for Internal PLL
	7:4	Output format select (also called PCMSEL[3:0])

Register #	Location	Description
(Control Register - R/W)		See Table 29
	3:2	Dual Channel Mode Output Mode Select 00 = Ch 0 on left, Ch 1 on right 01 = Ch 0 on both left and right 10 = Ch 1 on both left and right 11 = Reserved
	1	Mute 0 = Normal operation 1 = Mute audio output
	0	Reset 0 = Normal operation 1 = Reset audio module
1  (Status Register - R only)	31	Stereo Mode 0 = All other 1 = Dual mode
	30:29	MPEG-1 Sampling Frequency [kHz] <u>MPEG ID (bit 23) = 1      MPEG ID (bit 23) = 0</u> 00 = 44.1                      = 22.05 01 = 48.0                      = 24.0 10 = 32.0                      = 16.0 11 = Reserved                = Reserved
	28:27	De-emphasis Mode 00 = None 01 = 50/15 microseconds 10 = Reserved 11 = CCITT J.17
	26	Synchronization Mode 0 = Normal operation 1 = Sync recovery mode
	25	CRC Error 0 = No CRC error or CRC not enabled in bit-stream 1 = CRC error found
	24	PCM Underflow 0 = Normal operation 1 = PCM output underflowed
	23:4	MPEG header without sync word bit(s) 23 = ID bit                      (equals SAMPFREQ[2]) 22:21 = Layer 20 = Protection bit 19:16 = Bitrate index 15:14 = Sampling frequency (equal SAMPFREQ[1:0]) 13 = Padding bit 12 = Private bit 11:10 = Mode 9: 8 = Mode extension 7 = Copyright 6 = Original/home 5: 4 = Emphasis
3:0	Reserved	

## 10.2 PCM Audio Output

The PCM audio output from the 'AV7110 is a serial PCM data line, with associated Bit Clock (ASCLK) and left/right clock (LRCLK). PCM data is output serially on PCMOOUT using the serial clock ASCLK, as shown in Figure 17. The data output of PCMOOUT alternates between the two channels, as designated by LRCLK. The data is output most significant bit first. In the case of 18-bit output, the PCM word size is 24 bits. The first six bits are zero, followed by the 18-bit PCM value.



**Figure 17. PCM Output Timing (16-bit PCM format)**

Since the audio sampling frequency may change from one audio frame to another, whenever the frequency changes, the audio decoder soft mutes its output for one audio frame and notifies the ARM processor of the change in sampling frequency via an FIQ. Table 29 shows the SAMPFREQ and PCMSEL values that are not reserved. Any not defined there should be set to 0. The output PCM format and the over-sampling ratio between PCMCLK and ASCLK are specified by the four-bit register PCMSEL[3:0] which can be set via an API. The relationship between ASCLK and LRCLK is given as follows:

$$\begin{aligned} \text{16-bit PCM format:} & \quad \text{ASCLK} = \text{LRCLK} \times 32 \\ \text{18-bit PCM format:} & \quad \text{ASCLK} = \text{LRCLK} \times 48 \end{aligned}$$

The PCMCLK equals ASCLK if there is no over-sampling; otherwise PCMCLK equals 8x of ASCLK.

**Table 29. PCMCLK frequencies**

SAMPFREQ[2:0] (output to external PLL)	Sampling Frequency (LRCLK)	PCMSSEL[1] (Format select)	ASCLK (bit clock)	PCMCLK	
				PCMSSEL[0] = 0 (no over-sampling)	PCMSSEL[0] = 1 (8X over-sampled)
110	32 KHz	= 0 (16-bit PCM)	1.0240MHz	1.0240MHz	8.1920MHz
		= 1 (18-bit PCM)	1.5360MHz	1.5360MHz	12.2880MHz
100	44.1KHz	= 0 (16-bit PCM)	1.4112MHz	1.4112MHz	11.2896MHz
		= 1 (18-bit PCM)	2.1168MHz	2.1168MHz	16.9344MHz
101	48KHz *	= 0 (16-bit PCM)	1.5360MHz	1.5360MHz	12.2880MHz
		= 1 (18-bit PCM)	2.3040MHz *	2.3040MHz *	18.4320MHz
010	16 KHz	= 0 (16-bit PCM)	0.5120MHz	0.5120MHz	4.0960MHz
		= 1 (18-bit PCM)	0.7680MHz	0.7680MHz	6.1440MHz
000	22.05KHz	= 0 (16-bit PCM)	0.7056MHz	0.7056MHz	5.6448MHz
		= 1 (18-bit PCM)	1.0548MHz	1.0548MHz	8.4672MHz
001	24KHz	= 0 (16-bit PCM)	0.7680Hz	0.7680MHz	6.1440MHz
		= 1 (18-bit PCM)	1.1520MHz	1.1520MHz	9.2160MHz

\* This is the Power on Default settings

### 10.2.1 Using an External Audio PLL

Either the internal PLL or an external PLL may be used based on the setting of the PCMSRC bit of the Control Register (bit 8). If an External Audio PLL is used, the PCMCLK must be input to the device from the external clock source. Depending on how the external PLL is connected to the 'AV7110, the control signals can either be sent via the extension bus interface (EBI - Section 12) or the I<sup>2</sup>C interface (Section 14.8). According to these control signals the external audio PLL should adjust the clock PCMCLK to that shown in Table 29. In the event the audio sampling frequency changes from one audio frame to the next, an IRQ can be generated and the application software has the opportunity to update the settings of the external PLL according to the setting of bits 23, 15, and 14 (SAMPFREQ[2:0]) in the audio status register.

### 10.3 PCM Bypass

The audio decoder has a PCM-bypass feature that allows 16-bit PCM data to be loaded into the audio buffer and pass through the decoder to the PCMDATA output. Half-Sampling is not supported in PCM Bypass mode. If PCM data is at 96KHz sampling frequency, it is necessary for the application software (in conjunction with API's) to first down sample the data to 48KHz or lower. On output, if the 18-bit output data format is selected (see Table 29), the 16-bit bypass PCM data is padded with one zeroes on the LSB end, and six zeroes on the MSB end.

API's are provided to set up the audio decoder in the bypass mode and to obtain the start and end addresses of the audio buffer in the SDRAM. User software can use these API routines to load PCM data directly into the audio buffer. The input PCM data transfer can be carried out via the Extension Bus Interface (Section 12) or via the 1394 interface (through the SRAM, then DMA; see Section 13.1). The PCM data must be first reordered by the user into: 2 bytes left, 2 bytes right, 2 bytes left, etc., prior

to the transfer. For each two bytes (left or right), the most significant byte is loaded first (Little Endian data format). The front end of the audio decoder is also capable of performing byte swapping if the PCM data is in the Big Endian format. This feature can be activated together with the PCM-bypass mode through an API. If an incomplete block is loaded at the end, the entire block is not output. PCM data starts to output after four blocks (512 bytes) have been loaded. To change back to compress-data input, the audio decoder must be reset via an API.

#### **10.4 Elementary Stream Playback**

In normal operation, the audio decoder front end (Audio Front end, AFE) expects audio PES packets from the TPP and performs synchronization and time stamp extraction. It is also possible to turn the front end processing off through an API so that audio elementary stream can be fed directly to the decoder, bypassing the front end processing. However, in such a case, user software will be completely responsible for audio/video synchronization.

#### **10.5 SPDIF Audio Output**

The SPDIF output conforms to the consumer format of the AES3 standard for serial transmission of digital audio data. When using an external PLL (PCMSRC=1), SPDIF is supported only if 8X over-sampled PCMCLK is supplied (*i.e.*, PCMSEL[0]=1). The validation bit is disabled prior to muting the output during channel change or a Sample Frequency change. During Sample frequency change, the SPDIF output is invalid. The External SPDIF decoder should recognize this and mute the output. It's PLL is expected to recover automatically when the new frequency is achieved.



## 11. Synchronization

The 'AV7110 uses the System Time Clock (STC) and Presentation Time Stamp (PTS) to determine the playback time of the decoded audio and video data. The relationship between STC and PTS is determined at the time of the generation of the respective elementary streams (ES). Upon reception, the de-multiplexing software in the 'AV7110 extracts the video PTS and inserts it in an array of PTS and byte-count information maintained for the video decoder. For the audio, the PTS information is extracted by the audio decoder front end from the PES. The Video decoder contains an STC counter that is clocked by the system clock (27 MHz.) and normalized to a 90 KHz reference. The audio decoder does not contain a counter. Instead, it gets updated from the system time reference counter every 1.4 msec.

### 11.1 System Time Clock

The system reference counter is a 42 bit hardware/software hybrid counter as shown in Table 30.

**Table 30. System time clock model**

Bit 41 - 16	Bit 15 - 9	Bit 8 - 0
Software extension	Hardware counter (90 KHz.)	Hardware counter (27 MHz) rollover at count of 300

A copy of the full 42 bit counter is kept in a set of two software long variables (64 bits total length)

When the lower 9 bits of the hardware counter roll over, the 90 KHz hardware counter is incremented. When the hardware counter is incremented, an FIQ is generated to the firmware. The firmware will increment the software extension and write the value of the  $STC_{sys} + offset$  to the audio/video decoder if the audio/video decoder synchronization is enabled.

### 11.2 Startup Synchronization

At startup the hardware STC counter is free running and the increments to the software extension of the system STC take place every 1.4 ms. Both audio and video decoders are initialized to "free-run". This condition is identical for system startup as well as for re-synchronization after a channel change. When a PCR channel is activated by the application software through an API, the on-chip software monitors the incoming data in the PCR designated channel and reset the system common reference counter  $STC_{sys}$  to the first PCR that arrives in the designated stream (full 42 bits). At the same time, the video and audio STC are initialized to the same value, if they are enabled via the appropriate API calls.

### 11.3 Runtime Synchronization

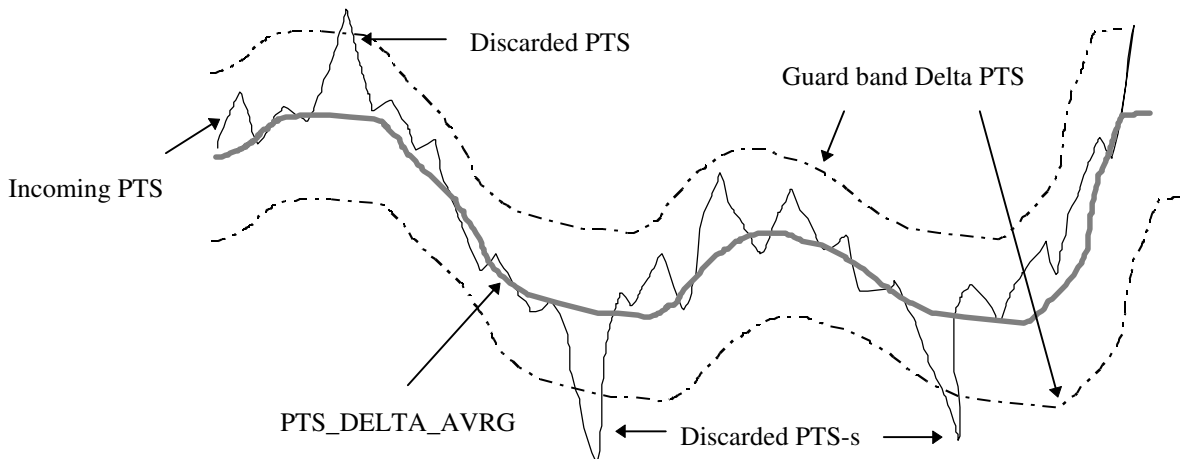
After the initial synchronization setup, the software will monitor the incoming video PTS values and compares them to the most recent STC. The difference between the two should not exceed a user programmable  $MAX\_PTS\_STC\_DELTA$ . If it does, the PTS is discarded. Otherwise the delta is stored in a circular 8 entry array. The average value of the array is calculated according to:

$$PTS\_DELTA\_AVRG = (\Sigma Array[0..7]) \gg 3$$

The value of  $MAX\_PTS\_STC\_DELTA$  is defined by the user via a call to the API function `sysSetSyncParms()`.

If the absolute value of the incoming PTS exceeds a pre-determined maximum delta from the  $PTS\_DELTA\_AVRG$ , the value is discarded (see Figure 18). This condition is calculated according to:

$$\text{unsigned} (PTS - PTS\_DELTA\_AVRG) \leq PTS\_GUARDBAND$$



**Figure 18. PTS tracking diagram**

### 11.4 Audio Synchronization

The audio synchronization of STC and PTS can be enabled/disabled through an API. The STC for the audio decoder time reference is loaded by the firmware every time the System reference counter ( $STC_{sys}$ ) “rolls over”. This roll over will occur every  $2^6$  ticks of the 90 KHz portion of the hardware system clock counter in the ‘AV7110. This is approximately every 1.4 msec. The firmware receives an FIQ from the counter and write  $STC_{sys} + aud\_offset$  to the STC register in the audio decoder. The value for  $aud\_offset$  is 0 by default after initialization and after audio channel change. The value for  $aud\_offset$  can be derived from either of two sources:

1. Setup by application software via `syncSetStcOffset()` API.
2. Xideo decoder signal in case of Audio-video sync (lip-sync).

### 11.5 Video Synchronization

The video synchronization of STC and PTS can be enabled/disabled via an API. The STC for the video decoder is clocked by the 27 MHz. system clock. The initial value of the video STC ( $STC_{vid}$ ) is written to the video decoder from the firmware. This happens at the time that the first PCR from the designated PCR channel arrives at the AF. In some cases the PTS can consistently be off by a certain amount, causing the video buffer to eventually over- or under-run. If this occurs, the firmware will detect the under/over-runs via FIQ-s and adjust the Video STC with an offset, such that the under/over-run ceases.

The algorithm to prevent the occurrence of under-flow caused by consistent  $PTS < STC$  is based on a monitoring of the delta ( $PTS - STC_{sys}$ ) every time the  $STC_{sys}$  counter rollover interrupt occurs in the following manner:

1. The current ( $PTS - STC$ ) is read from the video decoder, software registers the delta value.
2. If the  $STC_{sys} + GUARDBAND\_VALUE > PTS$  twice in a row, then adjust the STC in all the appropriate channels by  $STC = STC - ADJUSTMENT$ . In case lip-sync is enabled, the audio STC offset is updated and a new STC value is written out to the audio STC .

The values of `GUARDBAND_VALUE` and `ADJUSTMENT_VALUE` are defined by a call to the function `sysSetSyncParms()`.

In case the user wants to force a reset of the video STC reference the video STC is re-initialized with the sum of the `vid_offset + STCsys`. The `video_offset` variable can also be set by the application software using the `syncSetStcOffset()` API.

### **11.6 Audio-video Synchronization (*lip-sync*)**

If audio-video synchronization is required, the user application software should use the video decoder as the reference for both audio and video synchronization. The video decoder will synchronize itself as described in Section 11.5. The audio STC initialization and updates, however, are done differently from the description given in Section 11.4.

If the firmware calculates that an offset to the video STC is required as described in Section 11.5, it stores the video offset into the `aud_offset` variable too.

The  $(STC_{vid})$  should be used as reference for  $STC_{aud}$ . Thus, every time the  $STC_{sys}$  rolls over, the audio decoder update is calculated as follows:

$$STC_{aud} = STC_{sys} + aud\_offset$$

If this offset causes the audio decoder to generate audio buffer under- or over-run, then the firmware signals to the application software that lip-sync could not be achieved.

## 12. Extension Bus Interface (EBI)

The extension bus interface is a 32-bit or 16-bit bi-directional data bus with a 25-bit address. It also provides 3 external interrupts and a wait line. All the external memories or I/O devices are mapped to the 32-bit address space of the ARM. There are six internally generated Chip Selects (CSx) for devices such as EPROM memory, modem, front panel, front end control, parallel output port, and 1394 Link device. Each CS has its own defined memory space and a programmable wait register which has a default of maximum allowable values as defined in Table 31. The number of wait states depends on the content of the register, with a minimum of one wait state. The EXTWAIT signal can also be used to lengthen the access time if a slower device exists in that memory space. These are all programmable by the application software using APIs.

The active low output signal EXTTOE/EXTACTIVE is user selectable to be either asserted during EBI read cycles only (EXTTOE), the default selection, or asserted for both read and write cycles (EXTACTIVE). This signal equals the logical AND of all the chip selects as well as DRAM access on the EBI. The timing relationship of this signal with respect to other signals can be found in Section 17.1.

When the 32-bit EBI mode is selected and the ARM is operating in Thumb mode, instruction pre-fetch is supported. Each instruction fetch by the ARM to the external memory on the EBI will transfer two 16-bit instructions. One is sent immediately to the ARM and the other is stored in a local register to service the next instruction access.

During the Reset of the 'AV7110, all EBI signals are held in a tri-state condition until the Reset signal is released. Note that this includes the full 32-bit version of the EBI because the device does not configure the bus for 16 or 32 bit operation until it comes out of reset. External pull-up resistors are required on the control logic to prevent falsely enabling external devices during reset. Furthermore, roughly 10 k $\Omega$  pull-up resistors are recommended for the address and data lines as well to prevent problems associated with "floating" busses.

### 12.1 Address Range and Wait State of Chip Select

The Extension Bus supports the connection of 6 devices using the pre-defined chip selects. Additional devices may be used by externally decoding the address bus. Table 31 shows the name of the device, its chip select, address range, and programmable wait state range. Every device connected to the EBI is required to have tri-stated data outputs within 1 clock cycle (CLK40) following the removal of chip-select. DMA accesses to the EBI are interleaved with ARM based accesses. This one clock cycle (24 nanoseconds) constraint is to ensure a correct DRAM write when interleaved with DMA transfers. The tri-state timing requirement after other transactions (including DRAM) is 2 clock cycles or 48 nssec. with one exception. An ARM access to EPROM (CS1 region) during a DMA transfer to the EBI. This transaction requires the 24 ns restriction between DMA access and CS1 access because of the priority of code fetches to the ARM.

**Table 31. Extension Bus Chip Select Assignments**

Chip Select	Byte Address Range	Wait State	Device
CS1	2C00 0000 - 2DFF FFFF	1 - 5	EPROM (up to 32 MBytes)
N/A	2E00 0000 - 2EFF FFFF	N/A	DRAM (up to 16 Mbytes, on RAS1)
N/A	2F00 0000 - 2FFF FFFF	N/A	DRAM (up to 16Mbytes, on RAS3)
CS2	3000 0000 - 31FF FFFF	1 - 7	Peripheral Device
CS3	3200 0000 - 33FF FFFF	1 - 7	Peripheral Device
CS4	3400 0000 - 35FF FFFF	1 - 7	Peripheral Device
CS5	3600 0000 - 37FF FFFF	1 - 7	Peripheral Device
CS6	3800 0000 - 39FF FFFF	1 - 7	Peripheral Device

CS2, CS3, CS4, CS5 and CS6 all have the same characteristics. The ARM performs reads and writes to these devices through the Extension Bus. CS1 is intended for ARM application code, but writes will not be prevented. The user needs to provide the number of wait states for each CS during the initial set up. The default wait state is the maximum allowable wait state.

## 12.2 EBI Read and Write Cycles

The Extension Bus supports connection to external EPROM, SRAM, or ROM memory and DRAM with its 32-bit or 16-bit data and 25-bit address. It also supports DMA transfers to/from the Extension Bus. DMA transfers within the extension bus are not supported directly. They may be accomplished from user application software by using one DMA to the internal Data RAM, followed by a second DMA to transfer back to the EBI. Extension Bus single read and write cycle timings are shown in Figure 37 and Figure 38 of Section 17.1 respectively. The number of wait states can be calculated by the following formula:

$$\# \text{ of wait states} = \text{round\_up}[(8 + \text{device\_cycle\_time}) / 24.68]$$

For example, a device with 80 nsec read timing requires four wait states.

The user application software can program the read pattern on the Extension Bus as single or multiple. The single access mode allows one access during each CS cycle; multiple access allows more than one access during one CS cycle. The data is latched based on the programmed wait state for the respective chip select. The write access is always single. Both the read and write timing examples shown in Section 17.1 show the case of four wait states.

## 12.3 Interrupts

The 'AV7110 Extension Bus has three external interrupt lines. Each interrupt has a dedicated acknowledge pin (EXTACK[2:0]). One additional interrupt, BDIRQ (HSDI\_SIG7), is dedicated to the 1394 interface and it has no associated acknowledge signal. All the interrupts generate an IRQ to the ARM and application software is responsible for providing their service routine.

These interrupts are handled by a centralized interrupt controller. The interrupt mask and priority are managed by the firmware. The BDIRQ and three extension bus interrupts are connected to a total of four different IRQs. When the interrupt service routine on the ARM begins servicing one of the extension bus IRQs, it should first issue the corresponding acknowledge signal. At the completion of the IRQ, the ARM should reset the acknowledge signal.

## 12.4 The EXTWAIT signal

The EXTWAIT signal is an alternative way for the ARM to communicate with slower devices. It can be used together with the programmable wait state, but it has to become active before the programmable wait cycle expires. When the combined total wait states exceeds its maximum, the decoder is not guaranteed to function properly. The EXTWAIT signal is synchronized internally with the on-chip 81MHz clock. When a device needs to use the EXTWAIT signal, it should set the programmable wait state to at least 3. The duration of EXTWAIT signal should be at least 24.7 nanoseconds. Because the EXTWAIT signal has the potential to stall the whole decoding process, the ARM will cap its waiting to 500 nanoseconds. Afterwards, the ARM assumes the device that generated the EXTWAIT has failed and will ignore EXTWAIT from then on. Only a software or hardware reset can activate the EXTWAIT signal again. The timing diagram shown in Section 17.1 is an example of a read using the EXTWAIT signal. This example assumes that the wait state is set to 3 and the final effect of using EXTWAIT is equivalent to 5 wait states. If a device can not generate EXTWAIT within 30 nanoseconds, the user can set a larger wait state. The maximum response time of EXTWAIT signal can be calculated as follows:

$$\text{Max. response time of EXTWAIT (in nanoseconds)} = 12 + (\text{programmed wait state} - 3) \times 25$$

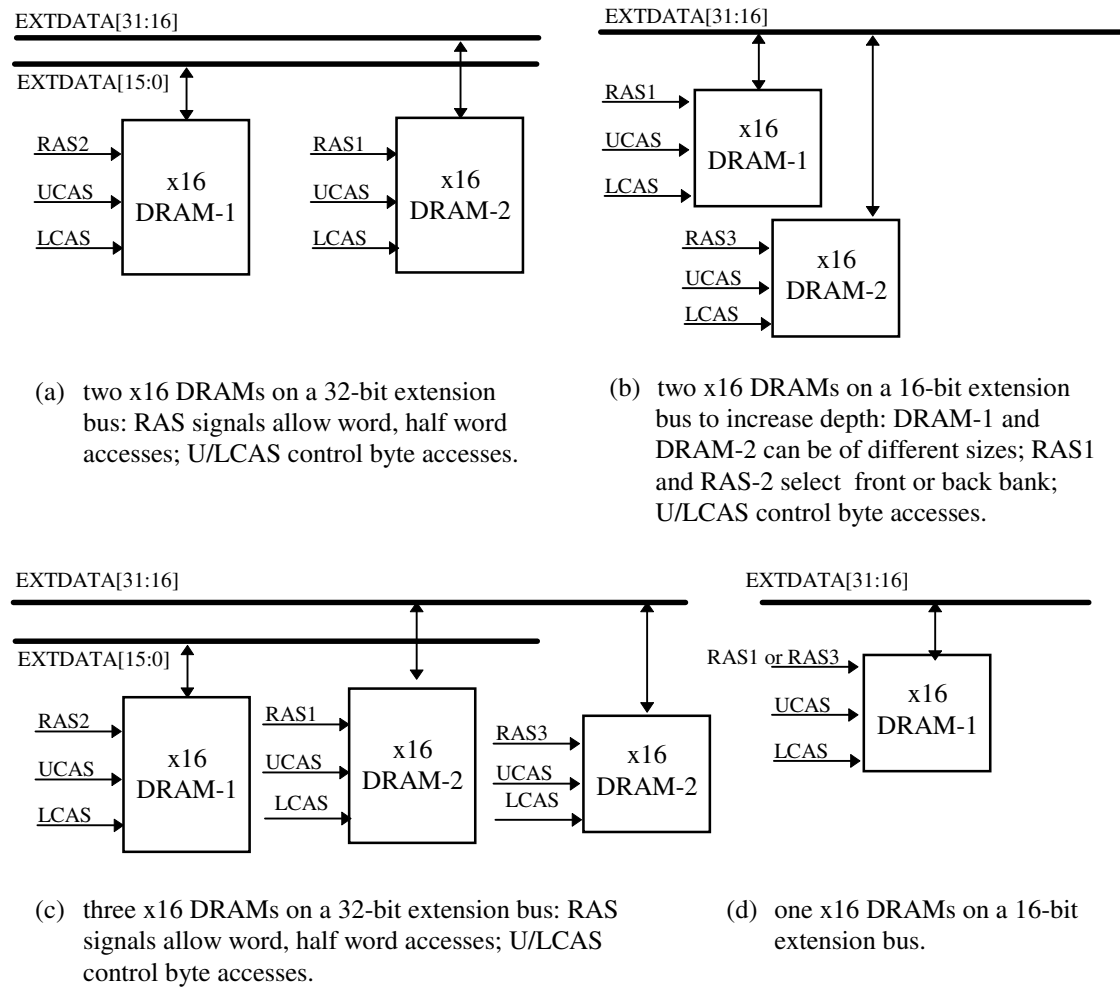
The maximum 500 nanoseconds wait time applies to each ARM access to the Extension Bus. If an 8 bit device is connected to the Extension Bus, the EBI automatically converts the single ARM word access to 4 consecutive byte accesses, and the maximum wait time applies to the total of the 4 byte accesses. In the consecutive read access where the CS remains active during the reading of four bytes, each byte access has to complete within 125 nanoseconds. For a device that requires longer access delay, say 250 nanoseconds, then the access from the ARM has to be limited only to half words. In the write access where the CS toggles for each byte write, then each write can take 500 nanoseconds. For reference, Section 17.1 also shows the same example but for a write cycle.

## 12.5 The Extension Bus DRAM

The Extension Bus supports access to 60ns and 70ns DRAM, as well as 60ns EDO DRAM; the default is 70ns DRAM. Figure 32 and Figure 33 of Section 17.1 respectively show the read and write timing of this DRAM interface. The DRAM must have a column address that is 8-bit, 9-bit, or 10-bit. The DRAM must have a data width of 8 or 16 bits. Byte access is allowed even when the DRAM has a 16-bit data width. The system default DRAM configuration is 9-bit column address and 16-bit data width. The user can reconfigure the extension bus to 32-bit with an API. The firmware will automatically verify the configuration during start up. Connection between the 'AV7110 and the DRAM is *glueless*, that is address bit A(i) on the EBI should be connected directly to address bit input A(i) on the DRAM.

The byte access is specified by the signal UCAS for EXTDATA[31:24], and LCAS for EXTDATA[23:16]. Another RAS signal (RAS3) is available to support an additional (possibly removable) DRAM device (such as a PCMCIA DRAM card) of 16-bit width. This DRAM device is to be mapped to a fixed (pre-determined) address partition (see Table 31). Configuration and existence of this additional DRAM device is determined by application software at power up of the set top box. Hot insertion or removal of this additional DRAM device is not allowed. The RAS2 is for increasing the width of DRAM configuration to 32-bit. Possible DRAM configurations are shown in Figure 19.

When the 'AV7110 is transferring data to /from the DRAM via DMA, it makes full use of the page mode read/write cycle and will read/write a byte/half-word/word every 50nsec (2 clock cycles). When 16-bit wide DRAM is accessed from the ARM, the DRAM controller will make use of page mode read cycle to transfer each 32 bit word. Each new 32-bit write/read is addressed independently to the DRAM.

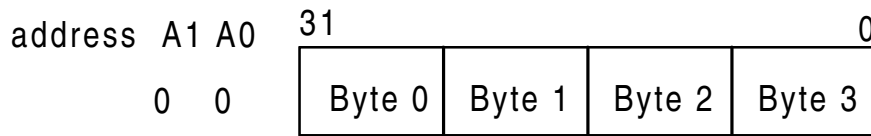


**Figure 19. Examples of DRAM connections to 16-bit and 32-bit extension busses**

**12.6 Byte Ordering on the Extension Bus**

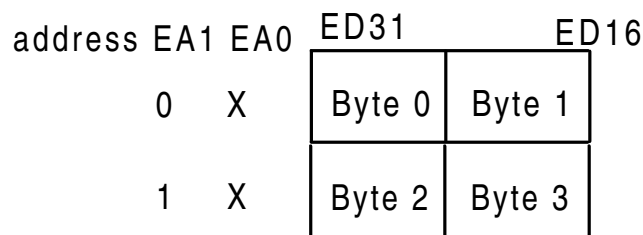
The Big Endian byte ordering is adopted inside the 'AV7110 and on the Extension Bus. That is, the most significant byte on the Extension Bus is from EXTDATA[24] to EXTDATA[31]. Figure 20 illustrates how the 4 bytes from the ARM bus are converted to 2 consecutive 16-bit half words on the Extension Bus or to 4 consecutive bytes if an 8-bit device were used.

**In the 'AV7100/ARM**

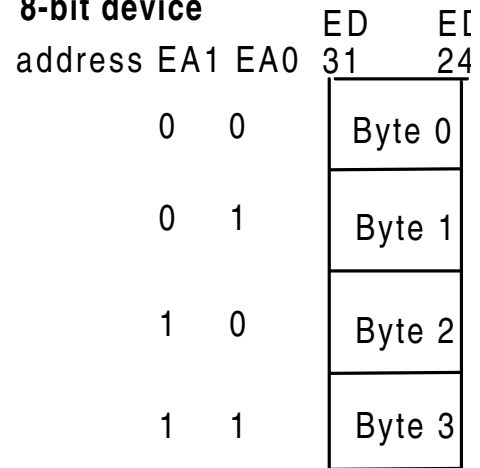


**In the Extension Bus**

**16-bit device**



**8-bit device**



**Figure 20. Byte ordering on the Extension Bus**

The application software initialization should specify the data size of the device connected to the Chip Selects. They can be programmed individually as x8 or x16 interface(or x32, if 32-bit EBI is used). The EBI does the routing between the internal 32 bit data and the 8 bit or 16 bit device on the Extension Bus. That is, the application program can write a word to an 8-bit device on the Extension Bus and the EBI will convert it to 4 consecutive byte writes. In read mode, the byte access is available to 8, 16 and 32 bit devices. In write mode, the byte access is available for DRAM and 8 bit device connecting to EXTDATA[31:24]. Similar constraints applies to 16 bit access.



### 13. High Speed Data Interface (HSDI)

The 'AV7110 provides a high speed data interface. This allows the user to select between an external connection to an IEEE 1284 parallel peripheral, an IEEE 1394 device, or an external DMA device. Logic for these three interfaces shares signal pins and hence only one can be active at any given time. Table 32 shows the pin definitions for the HSDI in it's various configurations. Switching from one interface to another will require a predefined sequence of API calls. Two signal pins (HSDI\_STATUS[1:0]) indicate which of the three interfaces is active. They can be used to control external logic for interface arbitration. The definition of these signals is shown in Table 32. The default configuration on power up is the IEEE 1394 device (HSDI\_STATUS[1:0] = 00).

**Table 32. High Speed Data Interface Signal Pin Assignment**

Pin Name	I/O	IEEE 1394 I/F	IEEE 1284 I/F	External DMA
HSDI_DATA[7:0]	I/O	BDI[7:0] (In/Out)	1284_D[7:0] (In/Out)	EDMA_DB[7:0] (In/Out)
HSDI_SIG1	O	BDIRW (Out)	$\overline{1284\_ERROR}$ (Out)	$\overline{EDMA\_RD}$ (Out)
HSDI_SIG2	O	$\overline{BDIEN}$ (Out)	$\overline{1284\_ACK}$ (Out)	EDMA_DACK (Out) *
HSDI_SIG3	I or O	$\overline{BDAVAIL}$ (In)	1284_STROBE (In)	EDMA_A[3] (Out)
HSDI_SIG4	I/O, I or O	BDIF[2] (In/Out)	$\overline{1284\_SLCTIN}$ (In)	EDMA_A[2] (Out)
HSDI_SIG5	I/O, I or O	BDIF[1] (In/Out)	$\overline{1284\_AF}$ (In)	EDMA_A[1] (Out)
HSDI_SIG6	I/O or O	BDIF[0] (In/Out)	1284_SLT (Out)	EDMA_A[0] (Out)
HSDI_SIG7	I	$\overline{BDIRQ}$ (In)	$\overline{1284\_INIT}$ (In)	EDMA_DREQ (In) *
HSDI_SIG8	O	N/A (Drives High)	1284_PEND (Out)	$\overline{EDMA\_CS}$ (Out)
HSDI_SIG9	O	N/A (Drives High)	1284_BUSY (Out)	$\overline{EDMA\_WR}$ (Out)
HSDI_STATUS[1:0]	O	00	1x (see Section 13.3)	01

\* The polarity of these signals is selectable by a user callable API and defaults to active Low

An internal DMA unit is dedicated for the HSDI port and is under user level API control. There are some restrictions to the source, destinations and data types of this DMA unit. These are detailed in Table 33. A DMA write from the TPP to the HSDI in any of it's configurations, only sends the payload. It is impossible to view and perform a DMA transfer to the HSDI from the same data source. The only exception to this is for 1394 M packets, where the entire transport packet is sent - including the header.

**Table 33. Types of DMA Transfers Allowed for Data Ports**

Interface Configuration	Source Data Types	Possible Destination for DMA Input	Possible Sources for DMA Output
IEEE 1394/TPP (Dedicated I/F <sup>1</sup> )	M Packets	TPP	TPP
IEEE 1394 I/F	I/A Packets	DRAM or SDRAM <sup>2</sup>	TPP, DRAM or SDRAM
IEEE 1284 I/F	Elementary Stream or Data	DRAM or SDRAM	TPP, DRAM or SDRAM
External DMA	Elementary Stream or Data	DRAM or SDRAM	TPP, DRAM or SDRAM

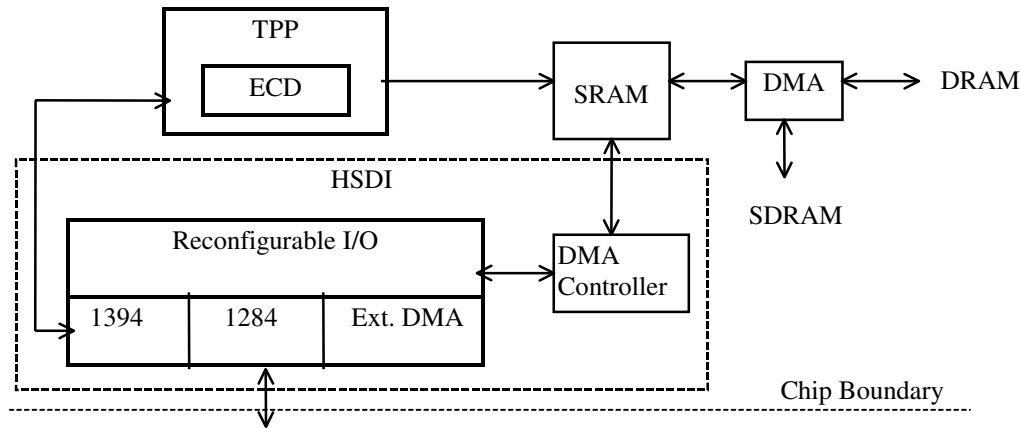
(1) This is a dedicated data path which is separate from the DMA channel.

(2) The source or destination between DRAM and SDRAM is a two-stage process using the on chip SRAM as an intermediate step.

The functional block diagram of HSDI is given in Figure 21. The HSDI supports the following concurrent data transfers.

- TPP/program to decoder; TPP/program to 1394 (using dedicated I/F - see note 1 of Table 33)
- TPP/program to decoder; 1394/DMA to/from memory
- TPP/program to decoder; memory/DMA to/from HSDI

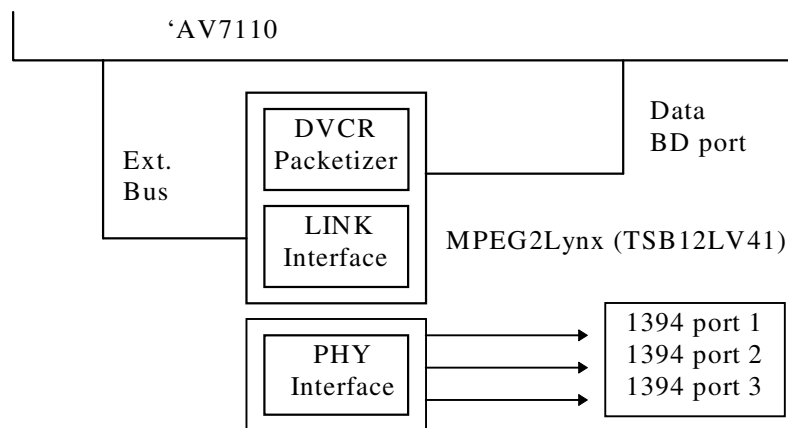
- TPP/program to decoder; TPP/DMA to HSDI (must be different PID's)
- TPP/program to decoder; TPP/program to 1394; memory/DMA to/from 1394
- TPP/program to 1394; 1394/program to TPP; memory/DMA to/from 1394
- 1394/program to TPP; memory/DMA to/from 1394
- 1394/program to TPP; TPP/DMA to 1394



**Figure 21. Functional Block Diagram of High Speed Data Interface**

### 13.1 IEEE 1394 Interface

On power-up the IEEE 1394 Interface is enabled on the HSDI. An API call can be used to configure the HSDI for another mode or to return to the 1394 Interface if necessary. To complete the 1394 implementation, the 'AV7110 requires an external Packetizer and Link Layer Controller, and a Physical Layer device. Figure 22 shows the connection between the 'AV7110 and these devices.



**Figure 22. 1394 Interface**

The command and control between the Packetizer or the Link layer interface device and the CPU is transmitted via the Extension Bus. The MPEG packet data from the bulk 1394 is transferred via dedicated pins on the 1394 interface from/to the TPP. The HSDI DMA channel is used for transferring Isochronous (I) or Asynchronous (A) packets to and from the Internal Data RAM. This data can then be processed by the application software as needed.

The 1394 interface on the 'AV7110 is designed to match that of Texas Instrument's MPEG2Lynx Integrated Circuit (IC - TSB12LV41) via the following 15 signals:

**Table 34. 1394 Interface Signals**

Signals	I/O	Description
BDICLK (CLK40)	Out	40.5 MHz clock for extension bus and peripherals.
BDI[7:0]	In/Out	8 bit parallel bi-directional data bus. This bus is only driven by the 'AV7110 or the MPEG2Lynx when BDIEN is active in write mode.
BDIRW	Out	When BDIEN is active, this signal indicates the direction of the transfer. When high (READ mode), the 'AV7110 reads data from the MPEG2Lynx and vice versa.
BDIEN	Out	This signal is active low. <ul style="list-style-type: none"> <li>in READ mode : Indicates that 'AV7110 wants to read a data. The 'AV7110 will drive this signal low for one clock period only when BDAVAIL is active.</li> <li>in WRITE mode : Indicates that data is ready for the MPEG2Lynx on BDI[7:0].</li> </ul>
BDAVAIL	In	This active low signal indicates to the 'AV7110 that a full packet is available on the bulky data interface. This signal should only be active when a full packet is available and should stay active until the transfer of the last byte in a packet is completed.
BDIF[2:0]	In/Out	Type of transmitted data. These signals are driven by the 'AV7110 when the 'AV7110 writes data to the MPEG2Lynx, and by the MPEG2Lynx when the 'AV7110 reads data from it. See Table 35 for meaning of these bits. For a full description see the MPEG2Lynx's specification. These signals are only driven when BDIEN is low.
BDIRQ	In	This is a general purpose interrupt line used for the 'AV7110/MPEG2Lynx interface. It must be set when a I/A packet is ready in the micro interface FIFO. For the micro interface FIFO, the meaning of the interrupt must be configurable as first or last "quadelet" of a I/A packet. The 'AV7110 must be able to access the interrupt register on the MPE2Lynx on the micro interface to know which interrupt has been sent. The 'AV7110 must also have access to the write pointer of the 200 byte FIFO.

**Table 35. 1394 Control Lines Description**

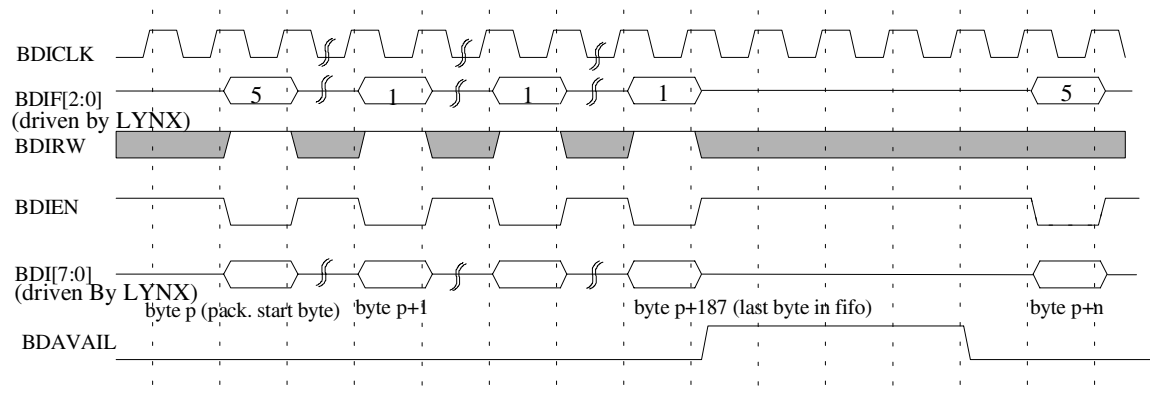
BDIF[2]	BDIF[1]	BDIF[0]	
0	0	0	Reserved
0	0	1	Byte of MPEG-2 transport packet (M-packet)
0	1	0	Byte of an unformatted Isonchronous packet (I-packet)
0	1	1	Byte from the A-packet except the last byte
1	0	0	Non Valid data
1	0	1	First byte of MPEG-2 transport stream (equivalent of BYTESTART)
1	1	0	Last byte of unformatted Isonchronous packet
1	1	1	The last byte of the A-packet

### 13.1.1 The 'AV7110 reads data from the MPEG2Lynx

When a full packet of data for the 'AV7110 to read is ready in its FIFO, the MPEG2Lynx sets the BDAVAIL signal low (active). When ready, the 'AV7110 activates the BDIEN (low) and BDIRW (high) for each byte it reads from BDI[7:0]. For each data byte read, the 'AV7110 also decodes the BDIF signals. If these signals indicate an MPEG-2 byte (1 or 5), the 'AV7110 generates the appropriate FEC-equivalent signals (PACCLK, Byte\_Start, and DCLK) to send the read data byte directly to the TPP for further processing. If BDIF has value other than 1 or 5, the 'AV7110 puts the data into a 4 byte shift register and transmits this word by DMA to the Internal Data RAM. Note that BDIF never equals 4 since the 'AV7110 always reads the data out of the MPEG2Lynx at a compatible rate. Furthermore, the 'AV7110 never drives BDIEN low more than once every two system clock periods. The BDAVAIL should only be set active by the MPEG2Lynx when a full packet is ready in the FIFO.

The MPEG2Lynx assumes the convention that MPEG-2 packets (M-packets) always have the highest priority in using the Bulky Data interface. That is, the MPEG2Lynx will preempt an on-going transmission of an I-packets before its completion to transmit an M-packet if it is ready for transmission. An M-packet transmission, on the other hand, is never preempted.

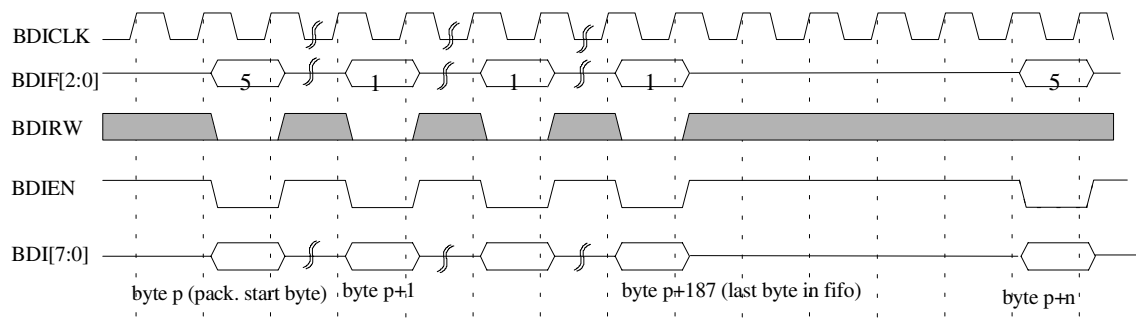
Figure 23 shows the functionality of the interface signals when the 'AV7110 reads data from the MPEG2Lynx. The maximum input rate of MPEG-2 data coming from the MPEG2Lynx and routed through the transport stream hardware demultiplexer block (TPP) is approximately 64.8 Mbits/second. The maximum input bit rate over one transport packet is 60 Mbits/second. Additional timing details can be found in Section 17.5.



**Figure 23. 1394 Interface Read Sequence**

### 13.1.2 The 'AV7110 writes data to the MPEG2Lynx

When the 'AV7110 writes data to the MPEG2Lynx (record mode), it drives BDIF, BDIEN, and BDIRW for one clock (system) period. BDIEN is driven low for only one byte and must be driven high between two byte transmissions. This is illustrated in Figure 24. Additional timing details can be found in Section 17.5.



**Figure 24. 1394 Interface Write Sequence**

### 13.1.3 The 'AV7110 Internal Data Path for 1394

In recording mode, the 'AV7110 sends either encrypted or clean packets to the 1394 interface. The packet is transferred as soon as it arrives. When recording encrypted data, the ECD module is bypassed. In the case of recording decrypted data, the TPP sends the encrypted part of the payload to the ECD module, and then forwards each byte to the 1394 interface. No CPU processing is done to the packet during recording. The TPP automatically modifies the header if the packet is decrypted.

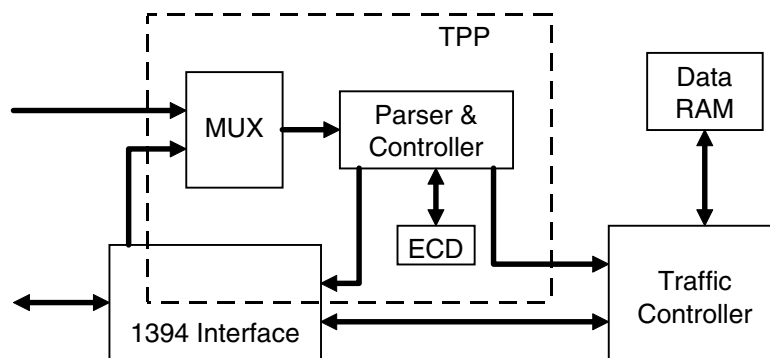
Note that the same bit stream cannot be sent to the 1394 interface twice (*e.g.*, once after decryption, and twice as part of a full transport stream with the by-pass capability). Both the MPEG2Lynx and the 'AV7110 support only one MPEG-2 transport stream channel on the Bulky Data interface.

During playback mode, MPEG-2 transport packets coming from the 1394 interface go directly to the TPP module.

Figure 25 shows the functional block diagram of the data flow between the TPP, ECD, and 1394 interface. Note that the major portion of the 1394 interface function is implemented in the HSDI module. Dedicated data lines are used for the 1394 interface that allows the 'AV7110 to work in the following different modes.

- Decode/decrypt/display one channel and record it.
- Decode/decrypt/display one channel and record it encrypted (pay-per-view).
- Decode/display one channel while recording all or part of 32 PIDs from a transponder, with each selected PIDs either encrypted or decrypted. In addition non selected PIDs can also be recorded in their original format (decryption is not possible for non selected PIDs).
- While the TPP is receiving MPEG-2 data from the 1394 port, the 'AV7110 can also record part of the bit stream it is receiving to the 1394 port
- Send the whole transport stream from the FEC to the 1394 port without any filtering or decryption and, at the same time, decode/decrypt/display a program from either the same transport stream or from another transport stream received through the 1394 port.

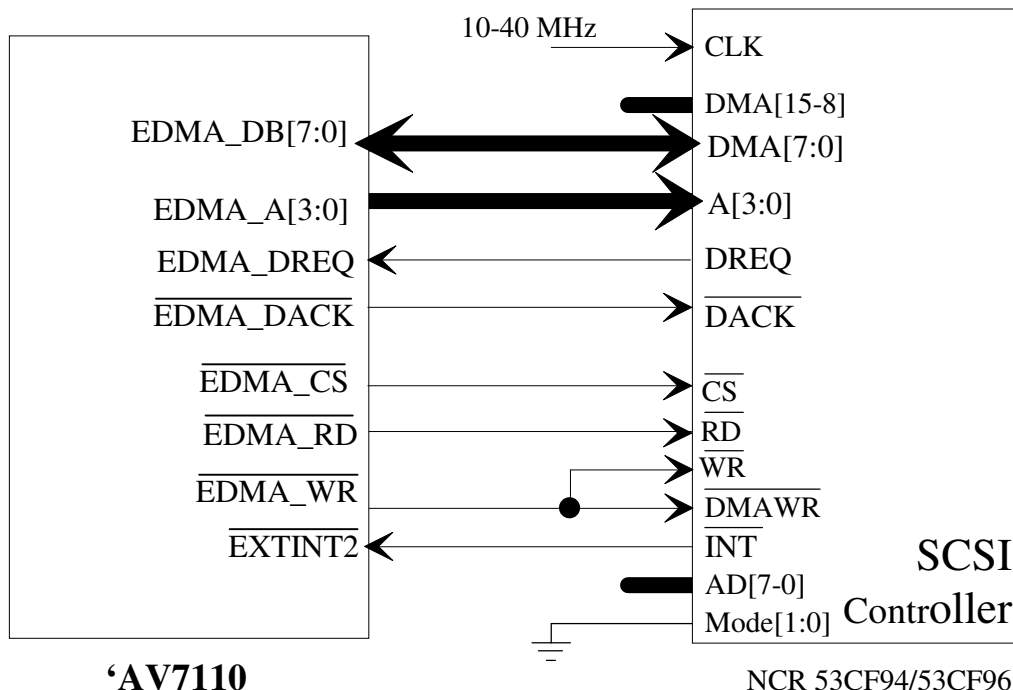
If the 1394 is used in the mode where the input and the output are multiplexed, the total bandwidth of the input data stream and the output data stream must be less than 160 Mbps. Note, however that these limits represent only the constraints imposed by the hardware. More stringent limits on the bit rates could be imposed by the 'AV7110 software processing requirements.



**Figure 25. 1394 Data Flow Block Diagram**

### 13.2 External DMA Interface

When configured for External DMA (EDMA) operation (HSDI\_STATUS[1:0] = 01), the HSDI is capable of being connected directly to an external data processing device. One example of this is an Enhanced SCSI Controller (ESC) device such as the NCR 53CF94-1/53CF96-1. Figure 26 illustrates the connections for this example configuration.



**Figure 26. Interfacing the 'AV7110 to a SCSI chip**

Because there is only an 8-bit bus on the HSDI, the ESC is limited to an 8-bit, Single Bus Architecture so it's mode selection inputs should be set to mode 0. The upper data bits can be left floating because they have internal pull-ups on the ESC. Likewise, the AD7-0 inputs of the ESC will not be used for this mode and can be left unconnected. Note that the ESC requires an external clock source from 10 MHz to 40 MHz. Because the CLK40 signal of the 'Av7110 is actually a 40.5 MHz clock it should not be used directly for this Controller without violating it's specification. Table 36 shows the signals needed for the EDMA interface.

**Table 36. External DMA Interface Signals**

Signals	I/O	Description
EDMA_DB[7:0]	In/Out	External Controller data bus.
EDMA_A[3:0]	Out	External Controller register select.
EDMA_WR	Out	Write signal. (Active Low)
EDMA_RD	Out	Read signal. (Active Low)
EDMA_DREQ	In	DMA request signal. *
EDMA_DACK	Out	DMA Data strobe select *
EDMA_CS	Out	Chip select for External Controller (Active Low)

\* The polarity of these signals is selectable by a user callable API and defaults to active Low

With the 'AV7110 configured for External DMA operation on the HSDI, data can be transferred at a maximum data rate of 5MBytes/sec. Interfacing to the HSDI DMA is done through API and so data transfer is under firmware control. Specific data formats such as the Enhanced SCSI Controller (ESC)

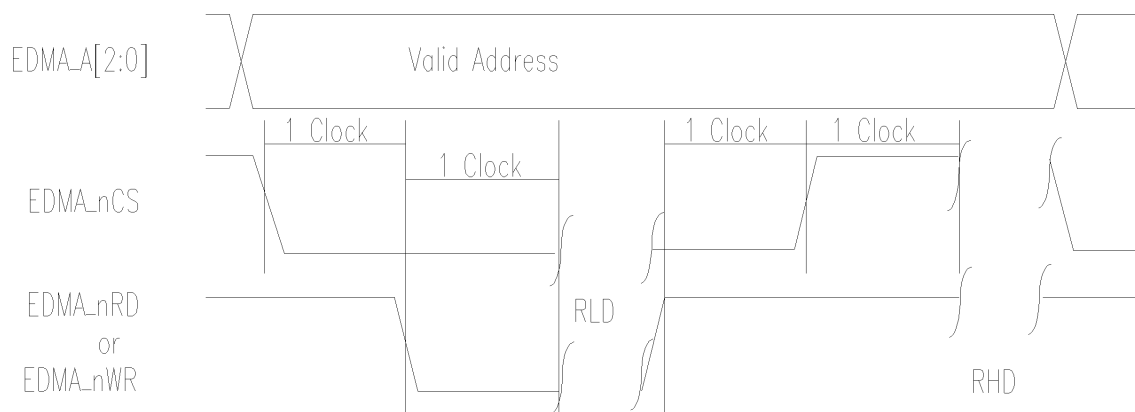
are handled by application software. Table 37 details the EDMA Register which allows the application software to configure the wait states and other factors of the EDMA Port. This register defaults to all 0's on power up.

**Table 37. EDMA Register Definition**

Location	Description
31:16	DMA Transfer Size [in Bytes] (total number of transfers)
15:10	DMA Burst Size [in Bytes] (transfers within a DACK frame) 0 = Bursts the entire DMA Transfer Size
9	DACK Polarity 0 = Active Low 1 = Active High
8	DREQ Polarity 0 = Active Low 1 = Active High
7:6	Register Access Low Delay ( <b>RLD</b> ) (see Note)
5:4	Register Access High Delay ( <b>RHD</b> ) (see Note)
3:2	DMA Transfer Low Delay ( <b>DLD</b> ) (see Note)
1:0	DMA Transfer High Delay ( <b>DHD</b> ) (see Note)

NOTE: This sets the number of internal (CLK40) clock cycles of delay to add.

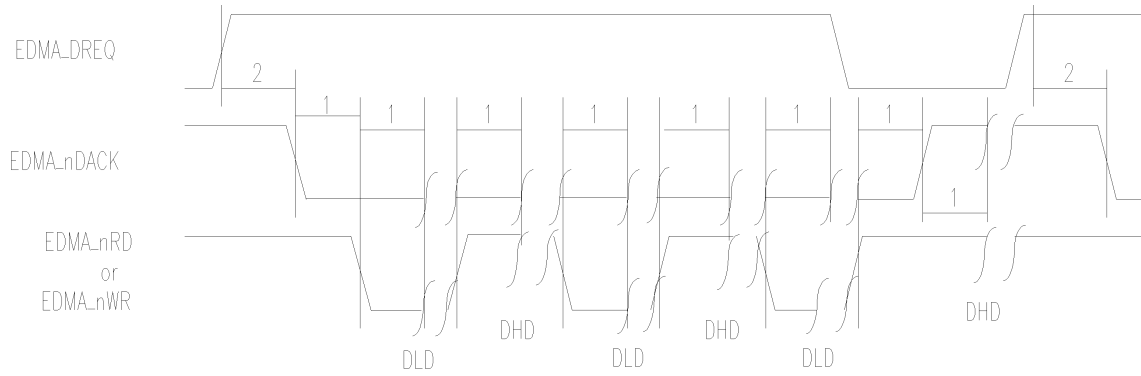
Command communication from the 'AV7110 to the ESC is done using the 4-bit EDMA\_A bus which is mapped into the ARM Addressing space of the 'AV7110 when the EDMA port is selected. This will allow the application software to perform such functions as writing directly to the registers of an external ESC. The definition of the function of these memory locations is dependent on the external device used and as such, entirely up to the user. An access to this memory space while the EDMA is selected will automatically generate a read or write cycle on the HSDI using the EDMA\_RD and EDMA\_WR signals and the EDMA\_CS signal. The functionality as well as the programmability for these accesses is shown in Figure 27.



NOTE: These Delay numbers are in units of internal clock (CLK40) cycles

**Figure 27. Programmable Delay for Register Accesses on the EDMA port**

The EDMA\_DREQ signal should be used by the External Controller to initiate an external DMA ('AV7110 DMA between the HSDI and the external device). EDMA\_DREQ is ignored until the DMA has been properly configured using the appropriate API calls. The source/destination addresses, the DMA size and the burst are under application software control via this API. Once asserted, the 'AV7110 will respond by transferring the data one byte at a time between the external device and the Internal Data RAM using the EDMA\_RD and EDMA\_WR signals and the EDMA\_DACK signal as shown in Figure 28.



NOTE: These Delay numbers are in units of internal clock (CLK40) cycles

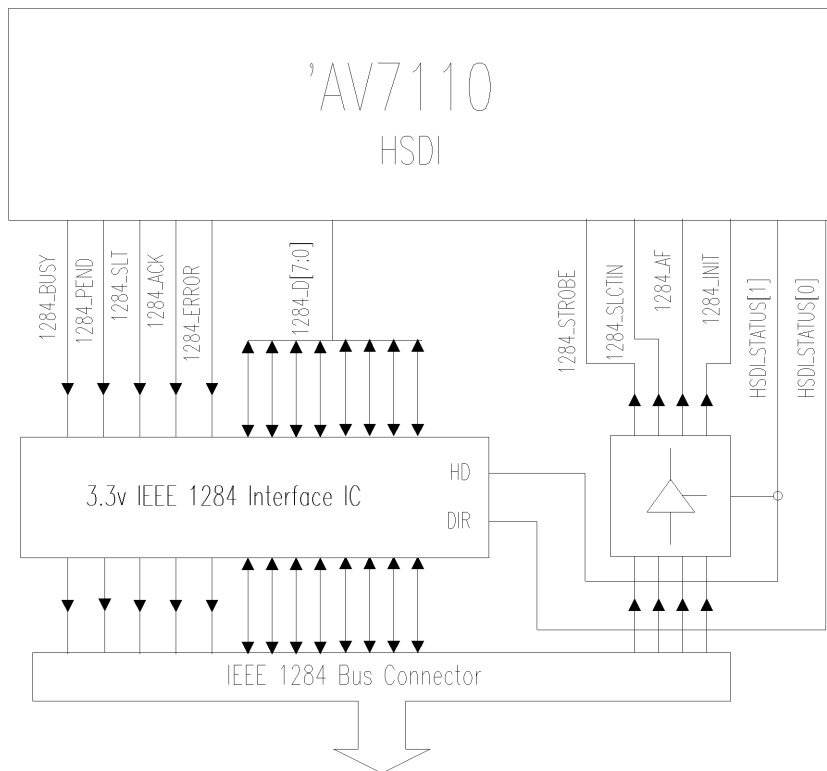
**Figure 28. Programmable Delay for DMA Accesses on the EDMA port**

### 13.3 IEEE 1284 Interface

The IEEE 1284 interface on the 'AV7110 has the following characteristics:

- Supports transfer of up to 10 Mbits/second
- Supports the compatibility, nibble, and byte mode
- Supports the ECP mode except for run length coding compression
- Does not support the EPP mode
- Peripheral mode only

The IEEE 1284 interface is used to connect the 'AV7110 to an external host. An example of the connections required is shown in Figure 29.



**Figure 29. Interfacing the 'AV7110 to an IEEE-1284 Bus**

Table 38 describes the functionality of the 19 signals of the IEEE 1284 interface. Note that the HSDI\_STATUS[1] signal is used to select the 1284 interface. Once set to this configuration, the



HSDI\_STATUS[0] signal determines the communication direction of the 1284 interface. A Low on this pin signals to the external device that the data direction is into the 1284 HSDI.

**Table 38. IEEE 1284 Interface Signals**

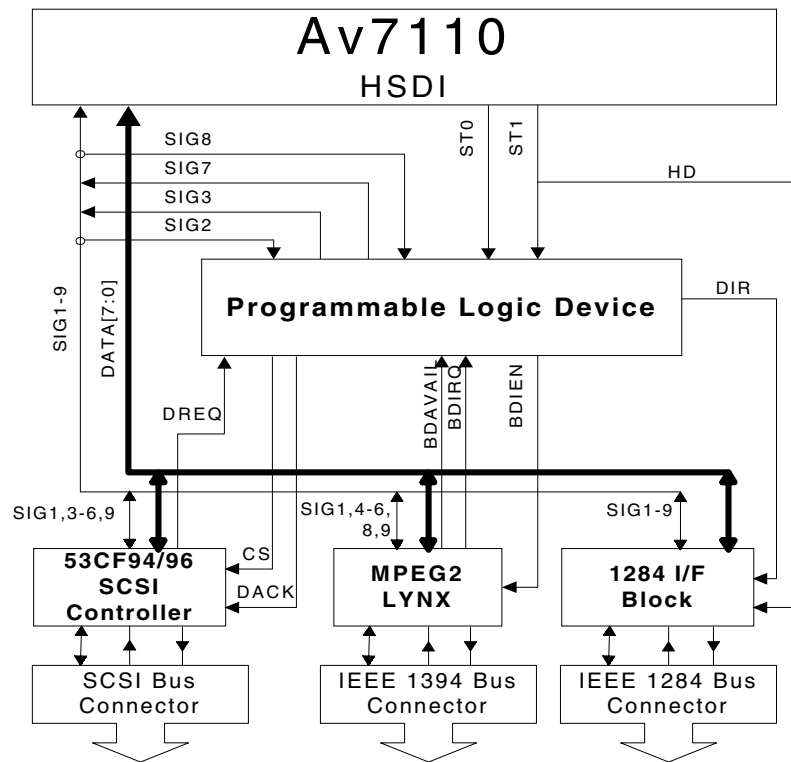
Signals	I/O	Description
1284_D [7:0]	In/Out	1284 data bus.
$\overline{1284\_ERROR}$	Out	1284 peripheral error (active low).
$\overline{1284\_ACK}$	Out	1284 data acknowledge (active low).
1284_STROBE	In	1284 data transition detect (active low).
$\overline{1284\_SLCTIN}$	In	1284 peripheral select (active low).
$\overline{1284\_AF}$	In	1284 peripheral auto-feed mode (active low).
1284_SLT	Out	1284 peripheral on-line.
$\overline{1284\_INIT}$	In	1284 peripheral reset (active low).
1284_PEND	Out	1284 paper end signal.
1284_BUSY	Out	1284 peripheral busy.
HSDI_STATUS[1]	Out	1284 Interface Enable
HSDI_STATUS[0]	Out	1284 Direction signal

The application software interfaces with the peripheral interface controller via API calls. To send data to the host, the application software will issue an API call to transfer the data from memory to the peripheral interface via a DMA. To receive data from the host, the application software will issue an API to transfer a predetermined number of bytes between the peripheral interface to internal Data Memory via the HSDI DMA unit. The application software must have a way to determine the number of bytes of data that is to be processed by the DMA as well as how to process it.

### **13.4 Combining the 1394, 1284 and External Controller Interfaces**

With additional external logic it is possible to combine the three HSDI interfaces within a single design. Note that performing the API call to changes interfaces involves resetting the HSDI, so transfers between interfaces is not possible (i.e. 1394 to 1284). Switching between HSDI modes should be detected by external logic and proper care should be taken to prevent contention between the external devices and the HSDI interface. Figure 30 shows the block diagram of an example of this configuration based on the devices used previously.

The signals CS, DACK, BDIEN, DIR and HD are essentially enables for the various interface blocks. Any signal not explicitly called out in the figure is connected in the same manner as the individual interface descriptions of the previous sections.



**Figure 30. Combined MPEG2Lynx, 1284 and SCSI Controller**

In this example, the logic is grouped into a single programmable logic device. For simplicity, some of the HSDI signal names have been shortened (i.e. ST1 & ST0 to represent HSDI\_STATUS[1:0], etc.). Note that the 1284 Interface Block is completely disabled (tri-stated) when it is not selected and so can be directly connected to the HSDI bus with no additional glue logic.

## 14. Communication Processor

### 14.1 Features

- Provides two programmable timers
- Provides three general purpose UARTs - two at up to 115.2Kbps and one at up to 9.6Kbps
- Accepts IR input signals
- Generates IR output signals
- Provides up to nine general purpose I/Os
- Manages I<sup>2</sup>C and JTAG interfaces

This module contains a collection of buffers, control registers, and control logic for various interfaces, such as UARTs, IR, I<sup>2</sup>C, and JTAG. All the buffers and registers are memory mapped and individually managed by the ARM CPU. Interrupts are used to communicate between these interface modules and the ARM CPU.

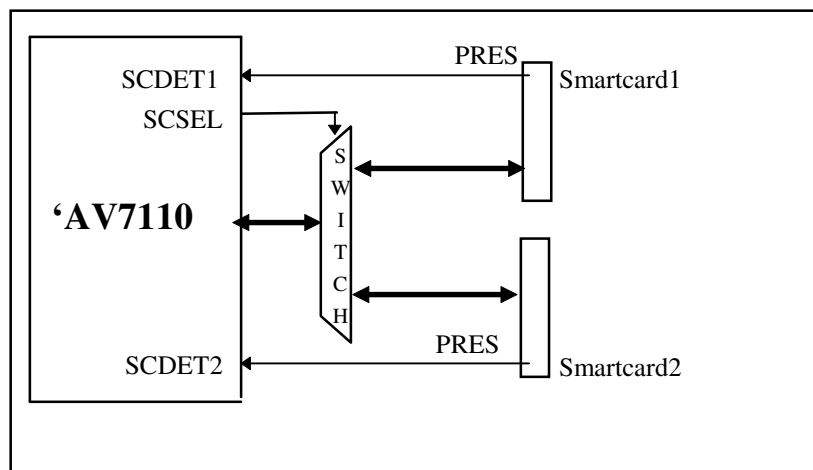
### 14.2 Smart Card Interface

- Both T=0 and T=1 protocols of the ISO 7816-3 standard are supported (except the bit-synchronous protocol)
- Up to two Smart Card devices directly supported
- Transmission/reception of parity bits are user programmable
- Shared 8-byte FIFO for transmitter/receivers
- User-programmable FIFO fullness (one quarter, a half, or three quarters full) interrupt
- User-programmable FIFO timeout (maskable interrupt)
- User select hardware flow control
- Integer and half-integer baud rate divisors
- Loop back mode for self-test

The 'AV7110 includes an interface to the Smart Card access control system. The interface consists of a high speed UART and logic to comply with both the ISO 7816-3 and the NDC standards. Applicable firmware drivers to control the interface are also included. Two Smart Cards can be supported via a smart card select pin which can be set via an API (see Figure 31). Only one Smart Card can be active at any time and switching from one smart card to another requires a full power down sequence of the active card followed by a full power up sequence on the second card.

**Table 39. Smart Card Pin description**

Pin name	Description
SCDET1, SCDET2	Card detect inputs. Polarity of these pins when a card is inserted is under API control
SCRESET	Output to Smart Card to reset the card
SCCLK	Output to Smart Card for clock
SCVppEN	Output to Smart Card to enable the programming voltage which needs to be generated externally. Polarity of this pin when a card is inserted is under API control.
SCVccEN	Output to Smart Card to enable the supply voltage which needs to be generated externally. Polarity of this pin when a card is inserted is under API control.
SCDATAIO	Serial data line, bi-directional
SCVccDetect	Smart Card Vcc detect input signal (for NDC mode), this pin is multiplexed with IO3.
SCSEL	Smart Card selection signal: Low = Smart Card 1 Selected, High = Smart Card 2 Selected.



**Figure 31. The ‘AV7110’s Interface to Two Smart Cards**

Table 40 and Table 41 give the usable Smart Card Interface clock frequencies for a given F value. The Nominal frequency for each ISO 7816-3 F value in the tables are calculated by multiplying F by 9600Hz. The Minimum values are 10% from Nominal and the Maximum values come from the ISO specification. All frequencies are in MHz. 2.025 MHz and 8.1 MHz are also supported but do not fit into the tables. All these clock frequencies are generated internally from the 81MHz clock.

**Table 40. Group 1 F Values**

F	Maximum	Nominal	Minimum	Usable SC Interface Clocks
372	5.0	3.57	3.22	3.375, 4.05, 4.5
558	6.0	5.36	4.83	6.0
744	8.0	7.14	6.43	6.75
1116	12.0	10.71	9.65	10.125
1488	16.0	14.28	12.86	13.5
1860	20.0	17.86	16.08	16.2, 18.0

**Table 41. Group 2 F Values**

F	Maximum	Nominal	Minimum	Usable SC Interface Clocks
512	5.0	4.92	4.43	4.5
768	7.5	7.37	6.64	6.75
1024	10.0	9.83	8.85	9.0
1536	15.0	14.75	13.28	13.5
2048	20.0	19.66	17.70	18.0

Although specific F/D pairs are not explicitly described here, every F/D pair is supported as long as the equation “(F/D)/Iclkpb” would result in an integer or half integer (i.e. 1.5, 2.5 ...). Note that the minimum value is 1, hence 0.5 is excluded. Iclkpb is defined as the number of internal clocks per bit (15.5 or 16). Both 15.5 and 16 internal clocks per bit are supported in the smart card interface hardware of the ‘AV7110. Selection of this is under software control. The Group 1 entries use 15.5 internal clocks per bit, where the internal clock is the smart card clock divided by the Baud Rate Divisor (BRD). Thus for F=372 and D=1 (default ISO 7816-3 values):

$$\begin{aligned} \text{BRD} &= 372/15.5 \\ &= 24 \end{aligned}$$

The Group 2 entries use 16 internal clocks per bit, where the internal clock is the smart card clock divided by the Baud Rate Divisor (BRD). Thus for F=512 and D=1:

$$\begin{aligned} \text{BRD} &= 512/16 \\ &= 32 \end{aligned}$$

Integer and half-integer values for BRD are supported by the smart card interface hardware. Therefore, if  $F=558$  and  $D=8$ :

$$\begin{aligned} \text{BRD} &= (F/D)/15.5 \\ &= 558/8 / 15.5 \\ &= 4.5 \end{aligned}$$

The 'AV7110 will output control signals to turn the card's VCC and VPP on and off as required but external switching is also required. The data I/O pins are 3.3V and require external level shifters to interface with 5V devices. These pins will require protection against over-voltage or ESD damage.

### 14.3 Timers

The 'AV7110 has two general purpose timers which are user programmable. Both timers contain 16 bit counters with 16 bit pre-scalers, allowing for timing intervals of 25 ns to 106 seconds. Each timer, timer 0 and timer 1, has an associated set of control and status registers. These registers are defined in Table 42.

**Table 42. Timer Control and Status Registers**

Register Name	Read/Write	Description	
tcr	R/W	Control Register	
		31 - 6	Reserved (set to 0)
		5	tint_mask - timer interrupt mask: 0 = enable interrupts 1 = mask interrupts
		4	reserved (set to 0)
		3	reserved (set to 0)
		2	soft - soft stop: 0 = reload counters on 0 1 = stop timer on 0
		1	tss - timer stop: 0 = start timer 1 = stop timer
		0	trb - timer reload 0 = do not reload 1 = reload the timer (read 0)
preld prd tddr	R/W	Pre-load Values	
		31 - 16	Value for timer to preload at timer start up or at timer rollover
		16 - 0	Value for pre-scaler to preload at timer start up or at pre-scaler rollover
tim32 tim psc	R	Actual Time Value	
		31 - 16	Actual timer value
		16 - 0	Actual pre-scaler value

The countdown timers are composed of two counters: the timer pre-scaler (psc), which is pre-loaded from tddr and counts down every sys\_clock; and the timer counter (tim), pre-loaded from prd. When psc = 0, it pre-loads itself and decrements tim by one. This divides the sys\_clock by the following values:

$$\begin{aligned} &(\text{tddr} + 1) * (\text{prd} + 1), \quad \text{if tddr and prd are not both 0,} \quad \text{or} \\ &2, \quad \text{if tddr and prd are both 0.} \end{aligned}$$

When tim = 0 and psc = 0, the timer will issue an interrupt if the corresponding tint\_mask is not set. Then both counters are pre-loaded if soft = 0. If soft is 1, the timer stops counting.

The timer control register (tcr) can override normal timer operations. The timer reload bit (trb), causes both counters to pre-load, while the timer stop bit (tss), causes both counters to stop.

## 14.4 UARTs

The 'AV7110 includes three general purpose UARTs that are memory mapped and fully accessible by user application programs. A set of APIs exists to assist in programming them. The output of the UARTs are digital and requires external level shifters for RS232 compliance. These UARTs support full duplex mode and are double buffered with sufficient FIFO space to minimize the interrupt frequency to the ARM even when they are operating at their maximum transmission speeds.

UART 1 and 2 support handshaking through RTS and CTS signals. They work with baud rates of 1200, 2400, 4800, 9600, 14,400, 19,200, 28,800, 57,600 and 115,200 bps. These two UARTs transmit/receive 1 start bit, 7 or 8 data bits, optional parity, and 1 or 2 stop bits.

UART 3 has no support for hardware handshaking, it works with baud rates of 1200, 2400, 4800, and 9600 bps. It is possible for the user application software to implement software handshaking with any of the UARTs.

The UARTs are fully accessible using the API and can generate interrupts when data is received or the transmit buffer is empty. The CPU also has access to a status register for each UART that contains flags for such errors as data over-run or framing errors.

## 14.5 IR input port

Hardware is provided to capture and deliver IR data bits to user software for command decoding. Sample IR command decoding driver software for several commonly used IR formats are provided with the 'AV7110.

API software is provided to configure the hardware to recognize IR format with different parameter settings:

- Programmable duration for High and low levels
- data frame length (up to 32 bits)
- maximum time between frame (a 13-bit counter, each count equals one period of CLK40)
- LSB/MSB first
- repeat pattern present
- stop pattern present
- compare enable (ignore repeated frames if this bit is set)

At start up, user application software sets up the parameters above and defines the encoding pattern of the preamble, stop, repeat, data zero, and data one by storing the duration of the high and low levels of each pattern in the IR-input-RAM. The IR hardware then looks for IR input (assumed demodulated) which matches the pattern:

```
<preamble><data> [<stop>]
<repeat> [<repeat>]
```

The "<repeat> [<input>]" input is recognized only if the repeat-pattern-present parameter is set and the input is preceded by a valid frame input. The 'AV7110's IR decoder supports the following two IR frame formats:

- [<preamble><data>]
- [<preamble><data><stop>]
- [<repeat>]

The <stop> pattern is used in some IR formats at the end of the data stream.

The [<repeat>] frame is used by some IR remote transmitters to indicate that a command key is being pressed and the command, which was sent right before the first [<repeat>], should be repeated. That is, if the received sequence of IR frames is:

```
[<preamble><data>] [<repeat>] [<repeat>] ... [<repeat>]
```

... then the key which causes the first [<preamble><data>] frame should be assumed by the receiver to be pressed for the duration of the [<repeat>] sequence.

The patterns of the <preamble>, <stop> and <repeat> components are each defined by a sequence of high (modulated) and low (not modulated) levels of programmable duration. Each duration is defined by a minimum value and a maximum value (in counts of 200 ns to 100 µs) to allow some error tolerances in the received pulse width.

The pattern of a <data> component is defined by the data bits being received. The number of data bits in the <data> component is fixed by frame-data-length (max length 32). Encoding patterns of data bit "one" and "zero" are each defined by a sequence of high and low levels of programmable duration. Each duration is defined by a minimum value and a maximum value (in counts of 200 ns - 100 µs) to allow some error tolerances in the received pulse width.

When a valid input frame is found, the IR input hardware stores the decoded data value in a register and sends an IRQ to the application software. With this API, software users can develop customized command decode driver software to decode virtually any IR format commonly in use. The only limit on how many change of levels in each of the components is that this data has to fit into the IR format RAM (size 64x12 bits).

## 14.6 IR output port

Hardware is provided on the 'AV7110 to generate drive signal on the output pin according to input data bit and format control signals provided by user software or to just retransmit the input signal received at the IR input port. External buffering is required to drive an IR LED.

In the first mode, demodulated input IR signals from IRIN are relayed directly to IROUT. The user has the option to have this signal modulated with a selected modulation frequency but NO transcoding is done. That is, the output IR signal has the same format as the input signal.

In the second mode, the application software generates an IR data word and writes it to a register, D Entry (or 2 registers A and D entries). Writing to the D Entry register by ARM activates the IR-encode module which then reads this data from A and D registers, encodes the data according to the output IR format stored in the IR format RAM and the content of the output-format-register, optionally modulates it with a selected modulation frequency, and sends the signal out on IROUT.

In order to communicate with IR receivers of different IR formats, the 'AV7110's IR output encoder supports a very flexible IR frame format. The format of the frame is specified through APIs and can be changed as frequently as per IR command transmission.

A frame format may consist of any combination of the following with up to 10 entries (including the E entry):

<preamble>	H entry
<data1>	A entry
<data2>	D entry
<misc1>	X entry
<misc2>	Y entry
<stop>	Z entry
<eof>	E (End of Format) entry

For example, a frame format can be defined as one of the following (though not limited to these examples):

```
[H D E]          (REPEAT-POSITION-START 1,
                  REPEAT-POSITION-END 2)
                  [H A H D H A E]          (REPEAT-POSITION-START
3,
                  REPEAT-POSITION-END 4)
                  [H D X Y E]          (REPEAT-POSITION-START
4,
                  REPEAT-POSITION-END 4)
[H H H A H D X Y Z E]
[H A X H D Y Z E]
```

Application software can also set up the registers REPEAT-POSITION-START and REPEAT-POSITION-END to point to the range of entries within a frame format to be repeated. The encoder will repeatedly send all the entries from that pointed to by REPEAT-POSITION-START to that pointed to by REPEAT-POSITION-END until the software writes to a REPEAT-STOP register (indicating key release). The encoder terminates the frame by sending the rest of the entries in the format register until the one before the Z entry. The encoder will then send the Z pattern if applicable.

The period for repetition (the time interval between the successive repeat portions) and the time interval between start of frame and the first repeat portion are also user programmable.

The pulse patterns for the different components (<preamble>, <data1/data2>-zero, <data1/data2>-one, <misc1>, <misc2> <stop>), if applicable, are defined in the same fashion as for those in the frame format for decoding, except that the level duration are defined by single counts instead of ranges. In operation, the application software can select the entries to be included in the IR format, on a frame by frame basis, through a control register (output-format) in the IR encoder.

The only limit on the number of change of levels in each of the entries is that the data has to fit into the IR format RAM (total size 128 words approx. and 11 bits). Maximum storage space for each entry type is given as follows:

- <preamble>: 9 words
- <data1/data2>-zero: 5 words
- <data1/data2>-one: 5 words
- <misc1>: 9 words
- <misc2>: 9 words
- <stop>: 9 words

## 14.7 General Purpose I/Os

The 'AV7110 has four dedicated (IO1, IO2, IO4 and IO5) and five multiplexed general purpose I/O pins (IO3 and IO6-IO9) which are user configurable. Each I/O port has its own 32-bit control/status register, IOCSRn, where n ranges from 1 to 9.

If an I/O is configured as an input and the delta interrupt mask is cleared, an IRQ is generated when an input changes state. If the delta interrupt mask is set, interrupts to the ARM are disabled. If no other device drives the I/O pin while it is configured as an input, it is held high by an internal pull-up resistor.

If an I/O is configured as an output (by setting the cio bit in the corresponding control/status register), the value contained in the io\_out bit of the control/status register is output. Interrupt generation is disabled when an I/O is configured as an output.

> iocsr io control/status register 0 (reset to 32'h00000020)

The definition of the control/status registers is given in Table 43 and Table 44.



**Table 43. I/O Control/Status Registers, IOCSRn**

Bit Number	Name	Description
31-6	Reserved	Set to 0 (read only)
5	gpio_en	Selects register for muxed gpios, gpio=0, other=1(default)
4	ddio	Delta detect bit: Configured as Output: (cio = 1) Always Reads as a '0' Configured as Input: (cio=0) Reads a '1' if bit 0 has changed since ddio was last cleared. Otherwise reads a '0' NOTE: Always write a '1' to this bit when enabling IRQ generation. This will clear the ddio bit.
3:2	gpio_irq1:0	See Table 44
1	cio	configure I/O: 0 = input 1 = output
0	io	IO bit: Writeable when IO is configured as an output (cio=1). Reads value on IO pin when IO is configured as input (cio=0).

**Table 44. GPIO\_IRQ Bit Definitions**

gpio_irq1	gpio_irq0	Function
0	0	Disable IRQ
0	1	An IRQ is generated on the Rising Edge
1	0	An IRQ is generated on the Falling Edge
1	1	An IRQ is generated on a State Change

## 14.8 I<sup>2</sup>C Interface

The 'AV7110 includes an Inter Integrated Circuit (I<sup>2</sup>C) serial bus interface that can act as either a single master (default) or slave. Only the 'standard mode' (100 kbit/s) I<sup>2</sup>C-bus system is implemented; 'fast mode' is not supported. The interface uses 7-bit addressing only. In slave mode, the address of the 'AV7110 is programmed by an API. In master mode, the 'AV7110 initiates and terminates transfers and generates clock signals.

To put the device in slave mode, the ARM must write to a control register in the block. The API must set the slave mode select and a 7-bit address for the 'AV7110. It must also send a software reset to the I<sup>2</sup>C to complete the transition to slave mode.

In slave mode, when the programmable address bits match the applied address, the 'AV7110 will respond accordingly. The 'AV7110 will also respond to general call commands issued to address 0 (the general call address) that change the programmable part of the slave address. These commands are 0x04 and 0x06. No other general call commands are acknowledged, and no action is taken.

## 15. Device Control Interfaces

### 15.1 Reset

The 'AV7110 requires a hardware reset on power up. Reset of the device is initiated by pulling the RESET pin low, while the clock is running, for at least 100 ns. The following actions will then occur:

- All signals on the EBI are held in a Tri-State condition until Reset is released
- input data on all ports is ignored
- external memory is sized
- data pointers are reset
- all modules are initialized and set to a default state
  - the TPP tables are initialized
  - the audio decoder is set for 16 bit output with 8x over-sampling
  - the OSD background color is set to black and video data is selected for both the analog and digital outputs
  - MacroVision is disabled
  - the I<sup>2</sup>C port is set to master mode

When the reset sequence is finished, the device begins to accept data. All data input prior to the end of the reset sequence is ignored.

### 15.2 JTAG Pins

Five JTAG pins, which are connected to the internal ARM module, are available for debug purposes.

## 16. Register Maps

### 16.1 NTSC/PAL Registers - Base Address 0x72000000

Register	Size	R/W	Offset	Bits	Value	Description	
ENC_MODE0_REG	8	R/W	0X0000	7	0	Video Mode register 0 BLNK - blanking enable normal mode (default)	
					1	enforced blanking (Only sync and color burst are encode in this mode)	
				6	0	reserved	
					5	0	reserved
				4	0	FLTMOD - Chroma low pass filter select 1.5MHz cutoff frequency (default)	
						1	3MHz cutoff frequency
				3:2	0,X	CBAR1,CBAR0 - color bars control normal mode (default)	
						1,0	75% color bars
				1:0	0,0	1,1	100%(NTSC), 95%(PAL) color bars
						0,0	reserved
ENC_MODE2_REG	6	R/W	0x0004	5	1	Video Mode register 2 Black and White mode (0=default, 1=BW)	
						3:2	4
				1:0	0,0		CbYCr latch timing change 11 'to resolve vdata timing between OSD and Encoder module (TBD) Reserved
ENC_CC_EN_REG	2	R/W	0x0008	1	0	closed caption enable CCEN1 - 2nd field caption encode enable CCEN0 - 1st field caption encode enable disable (default)	
						1	enable If CCEN0=0, caption data reg0/1 can not be written and have all zero values. If CCEN1=0, caption data reg2/3 can not be written and have all zero values.
ENC_CC_REG0	8	R/W	0x000C	7	6:0	caption data0 always zero (but odd parity is encoded for caption signal) 1st caption data in 1st field(ODD)	
ENC_CC_REG1	8	R/W	0x0010	7	6:0	caption data1 always zero (but odd parity is encoded for caption signal) 2nd caption data in 1st field(ODD)	
ENC_ESD_REG0	8	R/W	0x0014	7	6:0	caption data2 always zero (but odd parity is encoded for caption signal) 1st caption data in 2nd field(EVEN)	
ENC_ESD_REG1	8	R/W	0x0018	7	6:0	caption data3 always zero (but odd parity is encoded for caption signal) 2nd caption data in 2nd field(EVEN)	
ENC_MODE1_REG	7	R/W	0x001C	7		video mode register 1 VYNCRST - Vsync Reset	

Register	Size	R/W	Offset	Bits	Value	Description
				5	0	SYNCSEL - Sync source selection Internal Sync
				4:3	1	External Sync
					00	NTSC/PAL mode selection NTSC
					01	General Pal (BGI) (default)
					10	reserved
					11	reserved
				2	0	CBKILL - Color burst kill on/off CBKILL off (default)
					1	CBKILL on (no color burst signal)
				1	0	SYNCIOCTL - Vsync/Hsync port direction Vsync/Hsync are output
					1	Input mode
				0 -	0	SETUP - setup level select (valid when NTSC format) 0% setup
					1	7.5% setup
	4	R/W	0x0020			video mode3
				3		Video DAC output enable (composite)
				2		Video DAC output enable (R when RGB/YC_SEL=0 else C)
				1		Video DAC output enable (G when RGB/YC_SEL=0 else Y)
				0		Video DAC output enable (B when RGB/YC_SEL=0, else testdump)
	5	R/W	0x0024			RGB control register 1 Offset for rgb active display area start point recommended value is '01000'
	5	R/W	0x0028			RGB control register 2 Offset for rgb active display area end point recommended value is '11011'
	3	R/W	0x002C			RGB control register 3
				2	0	RGB/YC_SEL (DAC input mux select) R/G/B selection
					1	Y/C/testdump selection
				1:0		Phase control of 444 YCbCr input for RGB
ENC_STATUS_REG	4	R	0x011C			Status
				7:4		reserved (read zeroes)
				3	0	EVEN - even/odd field indicator odd
					1	even
				2	0	VBLNK - Vertical blanking period indicator active video line
					1	blanking line
				1		CST1 - caption data status 1
					0	Just after 2nd field caption enable OR Whenever vertical line counter goes to 21.
					1	After H/W reset or just after 2nd field(EVEN) caption disable or whenever caption data reg0 or reg1 are updated during 1st field caption enable.
				0		CST0 - caption data status 0
					0	Just after 1st field caption enable, OR

Register	Size	R/W	Offset	Bits	Value	Description
					1	Whenever vertical line counter goes to 21. After H/W reset or just after 1st field(ODD) caption disable or whenever caption data reg0 or reg1 are updated during 1st field caption enable.
						<p>Actually, CST1 and CST0 can be regarded as data flag with caption data register 0/1 and 2/3 respectively. Whenever the caption module is enabled, they are cleared. Then, they are set to 1 just after the caption data register is updated by ARM or VDEC module. On line 21 (caption transfer line), the CST1 and CST0 are cleared for EVEN and ODD field respectively.</p> <p>Video decoder module generates interrupt to ARM processor after transferring caption data into caption data register in encoder module. Then, ARM processor can read caption data from encoder and provide it to external encoder if needed.</p>
ENC_ATTR0_REG	8	R/W	0x0120	7:6 5:3 2:0		video attribute0 - not used - PAL: not used, NTSC: WORD0-B - PAL: not used, NTSC: WORD0-A
ENC_ATTR1_REG	8	R/W	0x0124	7:4 3:0		video attribute1 - PAL: GROUP2, NTSC: WORD2 - PAL: GROUP1, NTSC: WORD1
ENC_ATTR2_REG	8	R/W	0x0128	7 6		video attribute2 NTSC: {ATR_EN,-,CRC(6bit)} PAL: {ATR_EN,-,GROUP4(3bit),GROUP3(3bit)} ATR_EN - attribute insertion enable 0 = no insertion (default) 1 = insertion - not used
						Whenever the video attr data reg2 is written, the current value of video attr reg0 and reg1 is latched for active use. Though reg0 and reg1 can be updated several times, but the final value of reg0 and reg1 will not be used as actual ID value until the reg2 is written.
ENC_HSYNC_REG	6	R/W	0x0130	5:0		Active region offset with respect to hsync pin HSOF - hsyn offset (This value controls the adjustment timing as indicated in Section 9 Figure 18) This is to control Td value between 82 and 146 based on DCP requirements.

### 16.2 BitBLT Registers - Base Address 0x70000000

Register	Size	R/W	Offset	Bits	Value	Description
BLT_GO			0x30			
BLT_FREE_SEMA	1	R	0x34	0	0	module is currently in use

Register	Size	R/W	Offset	Bits	Value	Description
					1	module is free
BLT_RESET_REG	1	W	0x38	0		reset_semaphore - A soft reset feature is provided to interrupt a BitBLT task
BLT_HOLD_SEMA			0x3C			

### 16.3 CCP Registers - Base Address 0x6E000000

Register	Size	R/W	Offset	Bits	Value	Description
SCURB	16	R	0x0000			Uart receive buffer
				15:9		reserved (read zeroes)
				8	0 1	Parity Good Error
				7:0		received value, underflow returns most recent value
SCUTB	16	W	0x0000			Uart transmit buffer
				15:9		reserved (write data ignored)
				8	0 1	Last information field byte (T=1 only) Not last byte Last information field byte
				7:0		transmit data, overflow data ignored
SCCCR	16	R/W	0x0004			Smart card control (reset to 0x0033)
				19	0 1	clock mode 16x clock mode 15.5 clock mode
				18	0 1	Parity enable Disable parity generation and checking Parity enabled
				17	0 1	VCC detect SCVCCDETECT input pin not used SCVCCDETECT input pin used
				16	0 1	FIFO interrupt level half 1/4 transmit & 3/4 receive
				15:8		gtim - additional guardtime etus
				7:4		nnaq - number of nak retries allowed per character
				3:0		nper - number of parity error retries allowed per character
SCSR	16	R	0x0008			Smart card status register
				14	0 1	Error detection code status Good Bad
				13	0 1	Rx/TX status for turnarounds No loopback: HW/SW reset or last char Loopback: HW/SW reset or wrt to BRD rcvd. No loopback: HW/SW reset or wrt to BRD rcv Loopback: set when char is received.
				12	0 1	Icc Direct convention Inverse
				11:8		Transmit FIFO byte count
				7:4		Receive FIFO byte count

Register	Size	R/W	Offset	Bits	Value	
				3	0 1	Smart card 1 status Card is out Card is in
				2	0 1	Smart card 0 status Card is out Card is in
				1		Smart card Vcc
				0	0 1	tx_enz SCI is driving SMIO line SCI is not driving SMIO line
SCICR	16	R/W	0x000C			Control register (reset to 0x000204)
				21	0 1	SCVppEN,SCVccEN active level Active High Active Low
				20	0 1	Vcc enable disable enable
				19	0 1	Smart Card select Card 0 Card 1
				18	0 1	Waiting time Rollover No rollover Rollover
				17		FIFO time-out mask
				16	0 1	Loopback transmit to receive No loop back Loop back
				15	0 1	Transmitter/receiver FIFO mask Unmask Mask
				14	0 1	Error detection code LRC (Checksum) CRC (Only valid if pts = 1)
				13	0 1	Protocol type selection (pts) T=0 T=1
				12	0 1	Flow control disable enable
				11	0 1	Uart reset inactive active
				10	0 1	Initial character override No override Inhibits deactivation of contacts if TS is bad Allows incoming stream from smart card to be read
				9		Smart card detect level
				8		Initialization complete mask
				7		Data ready mask
				6		Work waiting time mask
				5		Transmitter holding register emth mask
				4		Vpp enable

Register	Size	R/W	Offset	Bits	Value	
					0	Disable
					1	Enable
				3:0		Clock select
					0x0	2.025 MHz clock
					0x4	3.375 MHz clock
					0x1	4.050 MHz clock
					0x5	4.500 MHz clock
					0x2	6.000 MHz clock
					0x6	6.750 MHz clock
					0x3	8.100 MHz clock
					0x8	9.000 MHz clock
					0x9	10.125 MHz clock
					0x7	13.500 MHz clock
					0xA	16.200 MHz clock
					0xB	18.000 MHz clock
SCBRD	16	R/W	0x0010	14		Baud rate divisor (reset to 0x18)
						Baud rate control
					0	No effect
					1	Subtract 1/2 from Brd [13:0]
				13:0		baudrate divisor
VDTIM	16	R/W	0x0014			Vcc detect time (reset to 0x704E)
WWTIM	16	R/W	0x0018			Work wait time (reset to 0x39F7)
SCRXEDC	16	R	0x001C			Receiver error detect code (reset to 0xFFFF)
SCBWT	16	R/W	0x0020			Block waiting time (Reset to 0x4B9A)
SCFTO	16	R/W	0x0024			FIFO time out (reset to 0x4B9A)
SCIS	16	R/W	0x0028			Smart Card status register (reset to 0x0000) (write one to clear)
				13		FIFO time-out status
					0	No Rx FIFO time-out
					1	Rx FIFO time-out occurred
				12		oe - (one to clear) overflow error
					0	No overflow error
					1	Overrun error
				11		Transmit FIFO half full
				10		Transmitter holding register interrupt
				9		Receive FIFO half full
				8		Data Ready
				7		Work waiting time error
				6		Smart card interface initialization completed
				5		Smart card 1 detect event
				4		Smart card 0 detect event
				3		Vcc detect event
				2		Answer to reset never occurred
				1		Nak error
				0		Parity error
SCRC	16	R	0x002C			Smart card register (Reset to 0x0255)
				15:8		Inf Bytes
				7:6		reserved (read zeros)
				5		Direct lsb
					0	If direct convention, transmit CRC msb first
					1	If direct convention, transmit CRC lsb first
						Not used in inverse convention
				4		rxcrc



Register	Size	R/W	Offset	Bits	Value	
					0 1	Inhibit CRC generation in receiver for last two bytes Do not inhibit
				3:1		Length byte position in frame
				0	0 1	Generate CRC with lsb first Generate CRC with msb first
SCTXEDC	16	R	0x0030	15:0		Transmit EDC register
	16	R	0x0100	15:8 7:0		i2crb : Used to send i2c data to arm reserved (read zeroes) received value, underflow returns most recent value
	16	W	0x0100	15:8 7:0		i2ctb : Used to send arm data to i2c reserved (write data ignored) transmit data, overflow data ignored
	16	R	0x0104	15:7 6:0		i2car : Used to send received address to arm reserved (read zeroes) received value, underflow returns most recent value
	16	W	0x0104	15:7 6:0		i2caw : Used by arm to send address to i2c reserved (write data ignored) transmit data, overflow data ignored
	16	R/C	0x0108	15 14 13		i2cs : Status register (reset to 0x2200) strt - detected that start occurred (1) stop - detected that a stop occurred rnw_slv - slave mode received r/w bit
				12 11 10 9 8 7 6 5 4 3 2 1 0		write to master (1), read (0) ack - no acknowledge received (1) cnt_scl - clock output tri state contrl cnt_sda - data output tri state control acka - address acknowledge received (0) ackd - data acknowledge received (0) int_req - one cycle interrupt to arm gencall - general call address received addr_cmplt - complete address received data_cmplt - complete byte received ireq (write one to clear) - status of interrupt clear(0), pending(1) busy_bit - tells when i2c state machine is busy reserved (read zero) reserved (read zero)
	16	R/W	0x010C	15 14 13 12 11 10 9:0		i2cc : Control register (reset to 0x8000) mstr - set to master(1) or slave(0) rnw - master mode=> read from slave(1) write to slave(0) rpt_strt - generate a repeated start init_strt - initiate a start time_out - a time out has occurred, reset when a start is detected not_ack - a not acknowledge sent after the last data byte reserved (read zero)
	16	W	0x0110			i2ccas : Used by arm to set i2c address

Register	Size	R/W	Offset	Bits	Value
				15:7 6:0	reserved (write data ignored) transmit data, overflow data ignored
	16	W	0x0114	15:0	i2cti : Timing interrupt value used by to set i2c timing interrupt interrupt data, overflow data ignored
	16	R/W	0x0300	31:6 5 0 1 4 3 2 1 0	tcr : Timer control register reserved (read zeros) tint_mask - mask tint 0 don't mask timer interrupt 1 mask timer interrupt free - emulation mode free bit softe - emulation mode soft bit softn - normal mode soft bit tss - timer stop trb - timer reload, read zero
	16	W	0x0304	31:16 15:0	prid prd - period register tddr - timer divide down
	16	R/W	0x0308	31:0	count: Actual timer value
	16	R/W	0x0400	31:6 5	tcr reserved (read zeros) timer interrupt mask
				0 1 4 3 2 1 0 1 0	0 don't mask timer interrupt 1 mask timer interrupt emulation mode free bit emulation mode soft bit normal mode soft bit timer stop 0 start 1 stop - detected that a stop occurred timer reload
	16	W	0x0404	31:0	Preload value
	16	R/W	0x0408	31:0	Actual timer value
URB	16	R	0x0500	15:8 7:0	uart receive buffer reserved (read zeros) received value, underflow returns most recent value
UTB	16	W	0x0500	15:8 7:0	uart transmit buffer reserved (write data ignored) transmit data, overflow data ignored
ASPCR	16	R/W	0x0504	15 14 13 12 11 10 9 0 1 8 0 1	async serial port control register free soft uart reset reserved (read zero) transmit interrupt mask received interrupt mask data bits 0 seven 1 eight 8 parity select for seven bits 0 if pen=1, odd parity if pen=0, low (space) 1 if pen=1, even parity

Register	Size	R/W	Offset	Bits	Value
				7	if pen=0, high (mark) parity enable
				0	no parity
				1	parity
				6	stop bits
				0	1 stop bit
				1	2 stop bits
				5	reserved (read zero)
				4	set break
				0	normal operation
				1	force a zero on tx commands
				3:2	fifo interrupt level
				00	
				01	1/4
				10	1/2
				11	3/4
				1	enable hardware handshaking
				0	reserved (read zero)
USTAT	16	R/C	0x0508	15	uart status register
				14	parity error occurred
				13	reserved (read zero)
				12	break interrupt occurred
				11	transmitter empty
				10	transmitter fifo empty
				9	framing error
				8	overflow error
				7:3	data ready
				2	receive fifo count
				1:0	interrupt: inserted for testing purposes
					reserved (read zeros)
BRD	16	R/W	0x050C	11:0	baud rate divisor register
	16	R	0x0600	15:8	uart receive buffer
				7:0	reserved (read zeros)
					received value, underflow returns most recent value
	16	W	0x0600	15:8	uart transmit buffer
				7:0	reserved (write data ignored)
					transmit data, overflow data ignored
	16	R/W	0x0604	15	async serial port control register
				14	free
				13	soft
				12	uart reset
				11	reserved (read zero)
				10	transmit interrupt mask
				9	received interrupt mask
				0	data bits
				0	seven
				1	eight
				8	parity select for seven bits
				0	if pen=1, odd parity
				1	if pen=0, low (space)
				0	if pen=1, even parity
				1	if pen=0, high (mark)
				7	parity enable
				0	no parity

Register	Size	R/W	Offset	Bits	Value	
				6	1	parity
					0	stop bits
					1	1 stop bit
				5		2 stop bits
				4		reserved (read zero)
					0	set break
					1	normal operation
				3:2		force a zero on tx commands
						fifo interrupt level
					01	1/4
					10	1/2
					11	3/4
				1		enable hardware handshaking
	16	R/C	0x0608	0		reserved (read zeros)
				15		uart status register
				14		parity error occurred
				13		reserved (read zero)
				12		break interrupt occurred
				11		transmitter empty
				10		transmitter holding register empty
				9		framing error
				8		overflow error
				7:3		data ready
				2		receive fifo count
				1:0		interrupt: inserted for testing purposes
						reserved (read zeros)
	16	R/W	0x060C	11:0		baud rate divisor register
UART2	16	R	0x0700			uart receive buffer
				15:8		reserved (read zeros)
				7:0		received value, underflow returns most recent value
	16	W	0x0700			uart transmit buffer
				15:8		reserved (write data ignored)
				7:0		transmit data, overflow data ignored
	16	R/W	0x0704			async serial port control register
				15		free
				14		soft
				13		uart reset
				12		reserved (read zero)
				11		transmit interrupt mask
				10		received interrupt mask
				9		data bits
					0	seven
					1	eight
				8		parity select for seven bits
					0	if pen=1, odd parity
					1	if pen=0, low (space)
					0	if pen=1, even parity
					1	if pen=0, high (mark)
				7		parity enable
					0	no parity
					1	parity
				6		stop bits
					0	1 stop bit
					1	2 stop bits
				5		reserved (read zero)

Register	Size	R/W	Offset	Bits	Value	
				4	0	set break
					1	normal operation
				3:0		force a zero on tx commands
						reserved (read zeros)
	16	R/C	0x0708	15		uart status register
				14		parity error occurred
				13		reserved (read zero)
				12		break interrupt occurred
						transmitter empty
				11		transmitter fifo empty
				10		framing error
				9		overflow error
				8		data ready
				7:6		receive fifo count
				5:0		reserved (read zeros)
	16	R/W	0x070C	11:0		baud rate divisor register
IR_DEC_PREAMBLE_BASE	17w	R/W	0x0800			preamble
IR_DEC_REPEAT_BASE	11w	R/W	0x0850			repeat
IR_DEC_STOP_BASE	7w	R/W	0x0880			stop
IR_DEC_DATA0_BASE	9w	R/W	0x08A0			data 0
IR_DEC_DATA1_BASE	9w	R/W	0x08D0			data 1
IR_ENC_PREAMBLE_BASE	9w	R/W	0x0900			preamble
IR_ENC_MISC1_BASE	9w	R/W	0x0924			misc 1
IR_ENC_MISC2_BASE	9w	R/W	0x0948			misc 2
IR_ENC_STOP_BASE	9w	R/W	0x0990			stop
IR_ENC_DATA0_BASE	5w	R/W	0x09C0			A/D entry 0
IR_ENC_DATA1_BASE	5w	R/W	0x09E4			A/D entry 1
IR_DEC_REG_BASE	9	R/W	0x0C00	9		IRIN inverted
				8-4		Frame length
				3		compare enable
				2		Stop pattern present
				1		Repeat pattern present
				0		LSB/MSB first
	15	R/W	0x0C04	14:0		Max time between frames
	12	R/W	0x0C08	11:0		Count enable period
IR_PROGRAM_ADDR	1	R/W	0x0C0C	0		Program
IR_INBUSY_ADDR	3	R	0x0C10	2		DIRQ_ST
				1		NEW_FRAME
				0		IRINBUSY
IR_DOUT_ADDR	32	R	0x0C14	31:0		DOUT
IR_ENC_REG_BASE	9	R/W	0x0E00	9		IROUT inverted
				8-4		A/D entries data length
				3		Relay/software commands
				2		Modulated/Pulse
				1		INTERVAL_CNT_MODE
				0		LSB/MSB first
	30	R/W	0x0E04	29-15		Interval 1
				14-0		Interval 0
	12	R/W	0x0E08	11-0		Count Enable Period

Register	Size	R/W	Offset	Bits	Value	
IR_FORMAT_ADDR	30	R/W	0x0E0C	29-0		Format register
IR_REPEAT_POS_ADDR	10	R/W	0x0E10	9-5 4-0		REPEAT-POSITION-END REPEAT-POSITION-BEGIN
IR_REPEAT_STOP_ADDR	1	R/W	0x0E14	0		REPEAT-STOP
IR_CARRIER_ADDR	22	R/W	0x0E18	21-11 10-0		Carrier HI width Carrier LO width
IR_A_ENTRY_ADDR	32	R/W	0x0E20	31-0		A entry
IR_D_ENTRY_ADDR	32	R/W	0x0E24	31-0		D entry
IR_OUTRDY_ADDR	2	R	0x0E28	1 0		EORQ_ST OROUTRDY
GPIO1	32	R/W	0x1000	31:6		reserved (read zeros)
				5	0 1	select register for muxed gpios gpio other
				4		ddio0 - delta detect io bit. Write a one to clear ddio0.
				3:2	00 01 10 11	gpio0_irq1: gpio1_irq0 Disable IRQ Rising edge IRQ Falling edge IRQ State change IRQ
				1	0 1	cio0 - configure IO input output
				0		io0 writeable if cio0 is 1, reads value on gpio pad
GPIO2	32	R/W	0x1100			Same as GPIO1
GPIO3	32	R/W	0x1200			Same as GPIO1
GPIO4	32	R/W	0x1300			Same as GPIO1
GPIO5	32	R/W	0x1400			Same as GPIO1
GPIO6	32	R/W	0x1500			Same as GPIO1
GPIO7	32	R/W	0x0600			Same as GPIO1
GPIO8	32	R/W	0x1700			Same as GPIO1
GPIO9	32	R/W	0x1800			Same as GPIO1

#### 16.4 Audio Decoder Registers - Base Address 0x6C000000

Register	Size	R/W	Offset	Bits	Value	Description
AUD_CTRL_REG	32	R/W	0x0020	31:22		reserved (read zeroes)
				21	0 1	Disable Sync Enable Sync Disable Sync
				20	0 1	Byte Swap no swap (bypass PCM data is in Little Endian) swap bytes (bypass PCM data is in Big Endian)
				19	0 1	Elementary Stream Playback off (normal operation) on, incoming data goes directly to audio decoder bypassing the AFE

Register	Size	R/W	Offset	Bits	Value	Description
				18		PCM bypass
					0	disable bypass (normal operation)
					1	enable bypass of AFE and audio decoder
				17	0	Copyright bit auto-update enable
					1	allows software to write to bit 16
						Bit 16 is updated from the MPEG header
				16		see bit 17 above (bit is output through SPDIF)
				15:9		Category (programmed by user, output through SPDIF)
				8		PCM Clock source select (PCMSRC) defaults to 0 for Internal PLL
				7:4		Output format select (also called PCMSEL[3:0]) See PCMCLK frequencies table in Section 10.
				3:2	00	Dual channel mode output mode select Ch 0 on left, Ch1 on right
					01	Ch 0 on both left and right
					10	Ch 1 on both left and right
					11	reserved
				1	0	Mute Normal operation
					1	Mute audio output
				0	0	Reset Normal operation
					1	Reset audio module
AUDIO_SCR	32	R/W	0x0024	31:0		Audio SCR register bits 31:0
	32	R/W	0x0028	31:0		Audio SCR register bit 32
AUD_PLL_REG	32		0x0040			Audio PLL control register
AUD_PLL_CNT	16		0x0080			Audio PLL counter
AUD_FIQ_STAT_REG	32	R/W	0x0400	0	0	FIQ Status register PLL Locked
					1	SF Change
AUD_STAT_REG	32	R	0x0800	31	0	Stereo Mode all other
					1	dual mode
				30:29	00	Sampling Frequency 44.1 KHz if MPEG ID equals 1
					01	22.05 KHz if MPEG ID equals 0
					10	48.0 KHz if MPEG ID equals 1
					11	24.0 KHz if MPEG ID equals 0
						32.0 KHz if MPEG ID equals 1
						16.0 KHz if MPEG ID equals 0
						reserved
				28:27	00	De-emphasis Mode None
					01	50/15 microseconds
					10	Reserved
					11	CCITT J.17
				26	0	Synchronization Mode Normal operation
					1	Sync recovery mode
				25		CRC Error
					0	No CRC error or CRC not enabled in stream
					1	CRC error found

Register	Size	R/W	Offset	Bits	Value	Description
				24	0 1	PCM Underflow Normal operation PCM output underflowed
				23 22:21 20 19:16 15:14 13 12 11:10 9:8 7 6 5:4		23:4 - MPEG header without sync word ID bit (equals SAMPFREQ[2]) Layer Protection bit Bitrate index Sampling frequency (equals SAMPFREQ[1:0]) Padding bit Private bit Mode Mode extension Copyright Original/home Emphasis
				3:0		Version number of the audio decoder

### 16.5 Video Decoder Registers - Base Address 0x6A000000

Register	Size	R/W	Offset	Bits	Value	Description
VID_ARM_IF	24	R/W	0x00	23:22      11 0	           00 01 10 11   host_int video_irq	Host control register local state (reset to 10) run reserved halt reset host_int video_irq
OVFLMODE		R	0x04		0 1	Overflow Mode All data is in the SDRAM Split buffer is active and video data may exist in DRAM
VIDEO_SCR	24	R/W	0x0C  0x10	23:0  0	   0 1	SCR timer_high SCR(31:8) SCR timer mode reg SCR frequency 27 MHz 90 KHz
VID_STAT_REG	8	R	0x14	10  9  8  7  6	       0 1  0 1  0 1  0 1  0 1  0	Vsync Reset no reset Vsync Reset Valid EDS Data (NTSC) EDS data not valid EDS data is valid Valid CC Data (NTSC) CC data not valid CC data is valid Interactive Command Done command in progress Command done User Data Ready not ready



Register	Size	R/W	Offset	Bits	Value	Description
				5	1	user data ready
					0	Aspect Ratio Change
					1	no change
				4	1	aspect ratio has changed
					0	True Size Change
					1	Full size picture
				3	1	True size picture
					0	Vid Watermark Underflow
					1	no error
				2	1	WM Underflow
					0	Vid Watermark Overflow
					1	no error
				1	1	WM Overflow
				0		Reserved
					0	Bitstream Error
					1	no error
					0	bitstream error
VID_CMD_REG	8	R	0x18	1		Interrupt status
					0	Wide Screen
					1	enable pan scan
				0	1	disable pan scan
					0	SCR Valid
					1	not valid
					0	valid
	32	R	0x34	23		Video decoder status
						blk_busy bit indicates VLD is busy in decoding blocks.
				22		error bit is set if a bitstream error is detected.
				21		reserved
vid_control			0x40			Control and status register
				23:22		Operating mode
					00	run
					01	single step
					10	halt
					11	reset
				21		unused
				20:16		
					00000	Normal operation
					00010	read horizontal ROM lower 32 bits
					00011	read horizontal ROM upper 16 bits
					00100	read vertical ROM
					01nnn	read linebuffer nnn
					11nnn	write linebuffer nnn
						Test data is written to and read from the vid_test register. The address is written to the vid_addr register
				15		reserved for microcode internal use
				14:13		unused
				12		blank output
					0	normal display
					1	black output
				11		sync_deci
					0	sync video output to send_norm

Register	Size	R/W	Offset	Bits	Value	Description
					1	sync video output to send_deci
				10	0 1	inverted fsync (field polarity) even field (bottom) odd field (top)
				9	0 1	fsync (field polarity) odd field (top) even field (bottom)
				8:7	00 01 10 11	Line offset for even field same as odd field (normal) +1 (even field starts one line later) -2 (even field starts two lines earlier) -1 (even field starts one line earlier)
				6	0 1	active area current line is in inactive area current line is in active area
				5:2	n	data request delay (n*4 cycles)
				1	0 1	provide next luma line address provide two next luma line addresses
				0	0 1	don't provide chroma line address provide next chroma line address
vid_mode			0x44	23	0 1	VideoOut operating modes test mode no test mode test mode (12/24 instead of 480/576)
				22:15	0 96 128 144 153 170 192 250 ...	Increment value for horizontal filter. This value controls the upsampling ratio. 1/1 (no upsampling) 8/3 upsampling 2/1 upsampling 16/9 upsampling 5/3 upsampling 3/2 upsampling 4/3 upsampling 720/704 upsampling ... increment value = round_down(256 * source_pixels/output_pixels)
				14:12	000 001 010 011 100 101 110 111	vertical ration 1/1 (normal) 3/4 (letterbox) 3/5 (letterbox) 4/5 (letterbox) unused 1/2 (decimation) 1/4 (decimation) unused
				11:6		horizontal pan-scan offset (n/16th pel resolution)
				5	0 1	SIF mode no SIF mode SIF mode on

Register	Size	R/W	Offset	Bits	Value	Description
				4	0 1	PAL/NTSC NTSC mode (480 lines) PAL mode (576 lines)
				3	0 1	frame type chroma interpolation field type chroma interpolation
				2	0 1	monochrome color black and white
				1:0	00 01 10 11	Number of source pixels to transfer from SDRAM into the line memories (May be more or less than the actual number of pixels in source image) 720 544 480 360
vid_lptr1	22	R/W	0x48			Byte start address of next luma line, word aligned
vid_cptr	22	R/W	0x4C			Byte start address of next chroma line, word aligned
vid_lptr2	22	R/W	0x50			Byte start address of one line after next luma line, word aligned
vid_cc			0x54	16 15:0	0 1	Closed captioning register Closed caption mode normal extended closed caption data
vid_hptime	10	R/W	0x58	9:0	720 360 180	Number of active output pixels per line Normal mode Half decimation Quarter decimation
vid_vfifo		R/W W R	0x5C	31:0 23:16		FIFO data input FIFO input register for data from SDRAM Version number of the FIFO stage
		R R		15:8 7:0		Version number of the vertical stage Version number of the horizontal stage
vid_addr		R/W R W	0x60	21:0 7:0		Byte start address for data transfer from SDRAM Test address for memory test mode.

### 16.6 OSD Registers - Base Address 0x68000000

Register	Size	R/W	Offset	Bits	Value	Description
OSDCUR_REG	12	R/W	0x0000			Cursor Configuration register
				11	0 1	Mirror repeat mode bit off on
				10	0 1	Line repeat mode bit off on
				9		Pixel repeat mode (half-resolution)

Register	Size	R/W	Offset	Bits	Value	Description
					0 1	off on
				8	0 1	Cursor dual color mode bit single dual
				7	0 1	Display cursor channel : cvbs off on
				6	0 1	Display cursor channel : digital video off on
				5	0 1	Display cursor channel : encoder off on
				4	0 1	Cursor enable bit Off On
				3:0	N	Cursor blinking rate control N fields on - N fields off
OSDREG_BGCOLOR	24	R/W	0x0004	23:0		Background color register Cb,Y,Cr of background color default : black (0x801080)
OSDREG_WINENMASK	8	R/W	0x0008	7 6 5 4 3 2 1 0		Window enable mask register Window 7 enable Window 6 enable Window 5 enable Window 4 enable Window 3 enable Window 2 enable Window 1 enable Window 0 enable
OSDREG_MPEGVIDEOEN	2		0x000C	1 0	0 1	MPEG video input enable video "audio-enable" feature enable bit auto-enable off HW re-enables bit 0 after 3 fields and then clears this bit
				0	0 1	mpeg video enable Background MPEG Video Decoder output
OSDREG_DECX0Y0Y1	30	R	0x0018		Y1	X0Y0Y1- Decimation window coord
				29:20		x0 coord (actual written = x0 + 1)
				19:10		y0 coord (actual written = y0 + 1)
				9:0		y1 coord (actual written = y1 + 1)
					default	"0000000001","0000000000","0111011111"
			0x001C			Reserved
			0x0020			Reserved
	12		0x0024	11 10 9 8 7:4 3:0		BB request disable configuration control EBI access control SDRAM access vbi always enable hbi always enable y-based control x-based control
	3		0x0028			OSD test configuration mode

Register	Size	R/W	Offset	Bits	Value	Description
				1 0		iddq testing on : disable all memories memory read/write test on: disable OSD local memory write
OSDCUR_COLOR0	24		0x0030	23:0		Cb,Y,Cr of cursor color 0
OSDCUR_COLOR1	24		0x0034	23:0		Cb,Y,Cr of cursor color 1
TT_WSS_CFG	15		0x0038	14:0		Teletext/WSS configuration register
TT_LINE_BIT	32		0x003C	31:0		Teletext valid line flag for two fields

### 16.7 TC Registers - Base Address 0x66000000

Register	Size	R/W	Offset	Bits	Value	Description
IRQ_SET_REG	32	W	0x0010			Sets bits in the (unmapped) IRQ request register. Each '1' written sets the corresponding irqreq bit.
IRQ_RST_REG	32	W	0x0014			Clears bits in the (unmapped) IRQ request register. Each '1' written resets the corresponding irqreq bit.
IRQ_MASK_REG	32	R/W	0x0018			Each '1' causes the corresponding IRQ request to be masked.
IRQ_STAT_REG	32	R	0x001C			Combination of (unmapped) IRQ request and irqmask. Any '1' will cause a IRQ to be generated.
IRQ Bit definitions						
				29 27 25		hspint external interrupt 2 external interrupt 1
				23 21 19 17 15 13 11 9 7 5 3 1		external interrupt 0 CCP GPIO CCP UART2 CCP UART1 CCP UART0 CCP IR_DIRQ CCP I2C OSD vblank end OSD vblank begin CCP TIMER1 CCP SCI CCP TIMER0 All others reserved for software use.
EBI_WAIT TC_WAIT	32	R/W	0x0020	27-24 23-20 19-16 15-12 11-8 7-4 3-0		Programmable wait states for ebi devices CS6. resets to 0100. CS5. resets to 0111. CS4. resets to 0111. CS3. resets to 0111. CS2. resets to 0111. CS1. resets to 0101. Reserved
EBI_INTACK	32	R/W  W W W	0x0024	8 7 6		extintack[2:0] directly feed the extack pins. extintack[8:3] are used for bitwise control of extintack[2:0]: writing '1' clears extack[2] writing '1' clears extack[1] writing '1' clears extack[0]

Register	Size	R/W	Offset	Bits	Value	Description
		W		5		writing '1' sets extack[2]
		W		4		writing '1' sets extack[1]
		W		3		writing '1' sets extack[0]
		R		2-0		External interrupt status
TC_EXTMEMCFG	32	R/W	0x002C			extension bus configuration.
				22	0 1	extoen control pin. low during read (default) during chip select
				21		Reserved - set to zero
				20	0 1	cs6n access mode multi access single access
				19	0 1	cs5n access mode multi access single access
				18	0 1	cs4n access mode multi access single access
				17	0 1	cs3n access mode multi access single access
				16	0	cs2n access mode multi access
					1	single access
				14	0 1	cs1n access mode multi access single access
				13:12	00 01 11	Eb bus width : cs6n 8 bit bus 16 bit bus 32 bit (RAS2) bus
				11:10	00 01 11	Eb bus width : cs5n 8 bit bus 16 bit bus 32 bit (RAS2) bus
				9:8	00 01 11	Eb bus width : cs4n 8 bit bus 16 bit bus 32 bit (RAS2) bus
				7:6	00 01 11	Eb bus width : cs3n 8 bit bus 16 bit bus 32 bit (RAS2) bus
				5:4	00 01 11	Eb bus width : cs2n 8 bit bus 16 bit bus 32 bit (RAS2) bus
				3:2	00 01 11	Eb bus width : dram 8 bit bus 16 bit bus 32 bit (RAS2) bus
				1:0	00 01	Eb bus width : cs1n 8 bit bus 16 bit bus

Register	Size	R/W	Offset	Bits	Value	Description
					11	32 bit (RAS2) bus
DMAGENINT	4	R	0x0030	3:0		General DMA init There are four general DMAs and each bit 3:0, when set to '1', indicates that DMA is either pending or executing.
DMAGENSET	4	W	0x0034	3:0		Set General DMA init Mark a general DMA for processing by writing a '1' to the corresponding location. The location should have been determined and reserved by reading register #15. In overflow mode, every time this register is written, the general DMA count is incremented to determine when to place backflow DMA at high priority.
DMAGENXTFREE	3	R	0x0038	2 1:0		Next general DMA init A '1' indicates that the DMA setup location given in bits 1:0 is in use. When read returns the value 0-3 of the next available general DMA.
						The DMA pointer is incremented so that the given location is now reserved for the ARM. This DMA must be setup and the init pin set (register 14) or the general DMA will stall.
ARM_GP_BURST_SIZ	10	R/W	0x0048	9:0		General DMA burst size The largest general DMA in bytes that can be performed by the hardware before returning to the idle state. Note: A DMA can be larger, the hardware will break it down into DMAs of the burst size or less.
SD_BUF0_START	22	R/W	0x0054			Video circular buffer address in SDRAM
SD_BUF0_END	22	R/W	0x0058			
SD_BUF1_START	22	R/W	0x005C			Audio circular buffer address in SDRAM
SD_BUF1_END	22	R/W	0x0060			
VID_WRITE_COPY	22	R	0x0064			Copy of video write pointer Video's don't step over point
VID_WRITE_LIMIT	22	R	0x0068			
TC_AUD_RESET		W	0x006C			Flush Audio BS buffer A write to this register will cause the audio read/write pointers to be reset to the start of circular buffer #1. The audio is forced to perform a read on circular buffer #1.
TC_SW_RESET	1	W	0x00A0			Write '1' to reset chip
TC_IRQ_RAW_STAT	32	R	0x00B4			IRQ Status Each bit corresponds to one of the 32 FIQ interrupts listed in register #1. If a bit is high, it is active and may or may not have caused the interrupt to the ARM. More than one can be active at a time. The FIQ mask has no effect on this register. The fiqn signal will return high only after all unmasked values in this register are low.
TC_STAT	32	R	0x00BC			TC status The TC status register is related to the TC FIQ. Bits 6:0 can cause a FIQ.

Register	Size	R/W	Offset	Bits	Value	Description
				30	1	Entered overflow buffer mode. Packets are being set to DRAM.
				6	1	The next EOP packet has come in before the last was serviced.
				5	1	The split buffer has overflowed.
				4	1	HSP input buffer has overflowed.
				3	1	Audio has overflowed.
				2	1	Audio has underflowed.
				1	1	Video has overflowed, not valid in split buffer mode.
				0	1	TPP has been stalled.
TC_PTS_COUNT	24	R	0x00C0	23:0		Video PTS counter Counts every byte sent to the video buffer. The count is incorrect if the TPP DMA has stalled as a result of an overflow.
TC_PTS_RESET	1	W	0x00C4	0	1	Reset video PTS counter Note: There is not synchronization between video DMAs and the reset
	22	R/W	0x00C8	21:0		Audio read pointer Writes must be word address.
	22	R	0x00CC	21:0		Audio write pointer
	9	R/W	0x00D4	8:0		GPIO interrupt status On a read, this is the status of the GPIOs 9:1. Each bit has to be cleared by writing a '1' to its location.
	25	R/W	0x00D8	24:0		Overflow buffer start addr LSBs of the overflow buffer start address in DRAM
	25	R/W	0x00DC	24:0		Overflow buffer end addr LSBs of the overflow buffer end address in DRAM.
	8	R/W	0x00E0	7:0		Backflow burst size Backflow transfer size in bytes. A multiple of the burst size or slightly less is the most efficient size.
	25	R	0x00E4	24:0		Overflow buf read pointer 25 LSBs of the read pointer DRAM address for the overflow buffer.
	25	R	0x00E8	24:0		Overflow buf write pointer 25 LSBs of the write pointer DRAM address for the overflow buffer.
	24	R	0x00EC	23:0		Overflow buffer byte count The number of video data bytes in the split buffer.
	2	W	0x00F0	1 0	1 1	Overflow control Enable overflow Force overflow. this will force all packets through the overflow buffer.
	16	R/W	0x00F4	15:0		Backflow try The amount of space that must be available in the SDRAM before a backflow is performed. This value must be larger than the backflow burst size register #57. Ideally it should be a multiple of the



Register	Size	R/W	Offset	Bits	Value	Description
						burst size.
	16	R/W	0x00F8	15:0		Backflow Priority The number of general DMAs that can be set up before the backflow DMA is given highest priority. If backflow can not be performed when given the highest priority, the DMA count is reset.
EBI_DRAM_CFG	4	R/W	0x00FC			EBI DRAM configuration
				3:2	00 01	CAS 8-bit cas address 9-bit cas address
					10	10-bit cas address
				1	0 1	Dual DRAM depth-wise (RAS) width-wise
				0	0 1	Extended Tcp Mode Extended Tcp Mode Reduced Tcp Mode
	1	R/W	0x0100			EBI enable prefetch
				0	0 1	prefetch disabled prefetch enabled
	2	R	0x0104			HSP DMA init
				1	1	DMA for buffer 1 has been initialized and is pending
				0	1	DMA for buffer 0 has been initialized and is pending
	2	R	0x0108			Next free HSP buffer
				1	0 1	The DMA is available The DMA setup is busy
				0	0 1	Buffer 0 Buffer 1
	2 22 2 2 32 24 3	W W R/W R/W R R/W R/W	0x010C 0x0110 0x0114 0x0118 0x011C 0x0120 0x0124			HSP set DMA Ready HSP clear DMA ready HSP Acknowledge delay HSP select 1284/1394/EXTDMA Address Debug Bit Blit overflow mask Split buffer overflow
	32	R/W	0x01F0			HSP output buffer

### 16.8 TPP Registers - Base Address 0x64000000

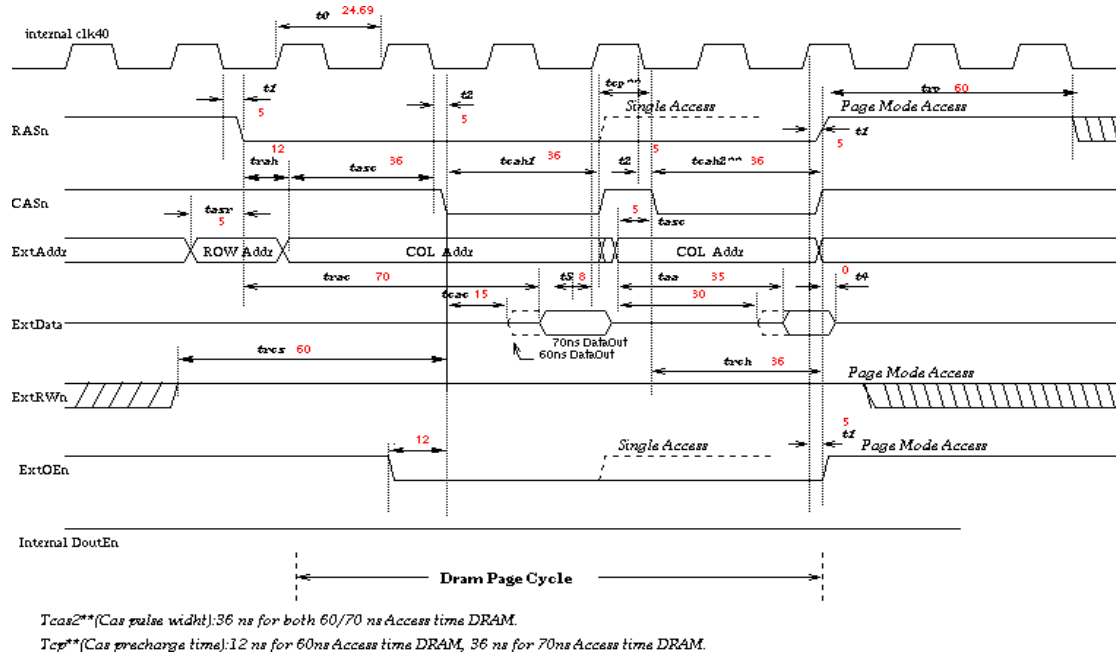
Register	Size	R/W	Offset	Bits	Value	Description
TPP_STATUS_REG	32	R/W	0x0020			Status Register
				31		TC Error/FIFO Error
				30:25		Unused
				24:23		Packet Type
				22		Auto buffer bit
				21		Stream error
				20		unused

Register	Size	R/W	Offset	Bits	Value	Description
				19 18 17 16 15:12 11:10 9 8		Duplicat Packet Continuity count error unused PID/SCID Valid Current Continuity count Adaptation Field Transport Priority unused
				7 6:5 4:0		Packet unit start Descrambling Control Logical Channel
TPP_CPU_CNTR	23	R/W	0x0024	22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0		Control Register Enable FEC on power up Enable Armbuf back 2 back PID's CAM test mode CAM test search CAM Bypass CpuMemLock input source select - fec or 1394 full bypass selective decode full bypass for 1394 Transfer AF Reserve Types Enable Stream Cipher Disable Vertical Blanking Update Sigma Delta Mode PCR Mode PACWAIT mode enable <unused> DERROR pin polarity BYTESTRT Valid Polarity PACCLK Valid Polarity DCLK direction DCLK polarity Enable Duplicate Packets Transport Error Enable Enable Match
TPP_SCR_COUNT	16	R	0x0028			This 16 bit hardware reference clock counter runs at 27MHz. It is user to trigger a SCR FIQ upon wraparound.
TPP_SCR_LATCH	16	R	0x002C			The SCR register latches the scr counter when an Auxiliary packet is encountered from the bitstream.
	1	W	0x0030			HW filter control
	2	R	0x0034			HW filter status
TPP_SIGMA_DAC	8	R/W	0x0038			This eight bit data register affects the input voltage to the VCxO. A midpoint value of 127 or 128 represents baseline.
TPP_SIGMA_LOW	16	R/W	0x003C			The sigma delta operating frequency is directly related to the noise. Essentially, the faster the frequency, the lower the noise. This frequency is divided from the 40MHz input clock. The divider can be set in

Register	Size	R/W	Offset	Bits	Value	Description
						this 16 bit register. A default divider value is loaded on reset. The divider is not an ordinary clock divider. Rather, it
						is a counter value which pulses a sigma delta cycle when the count value reaches the registered value. Then, the counter is reset to zero and starts over again.
TPP_SIGMA_HI	32	R/W	0x0040			HW filter high order match word
	32	R/W	0x0044			HW filter low order match word
	7	W	0x0048			HW filter attr
	32	R	0x004C			HW filter left results
	32	R	0x0050			HW filter right results
TPP_EOP_POS	5	R/W	0x005C			This 8 bit register moves the EOP FIQ location. If the EOP FIQ is wanted on the 100th input byte, set EOP compreg to 99.
	8	R/W	0x0060			cntrl1_1394
	8	R	0x0068			status_1394
	32	R/W	0x0070			crc_input
	32	R/W	0x0074			crc output
	8	R	0x0078			crc status
	2	W	0x007C			crc control
	32	R/W	0x0800			HW Filter table
	32	R/W	0x1000			HW Filter mask

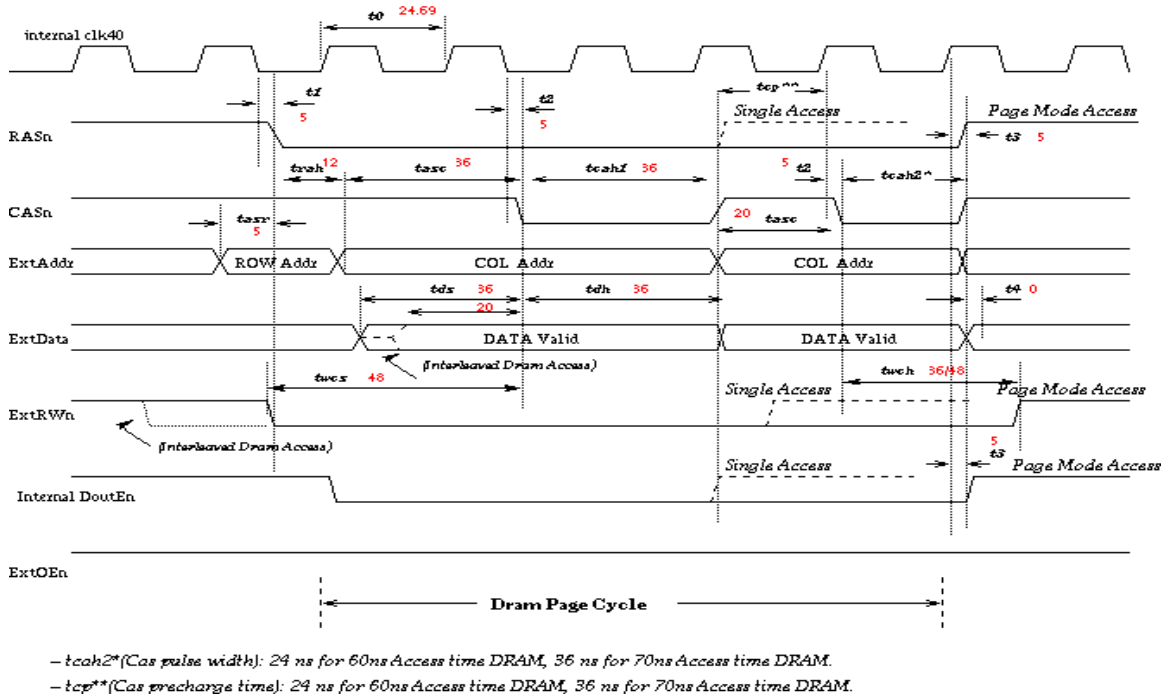
## 17. Timing Diagrams and DC Parameters

## 17.1 DRAM Interface Timing



**Figure 32. Read Access from DRAM**

(DRAM Single or Page Mode Write with new RAS – 60/70ns dram support)



**Figure 33. Write Access from DRAM**

(Start of Page Mode(DMA) Write with RAS and RAS2 – 60/70ns dram support)

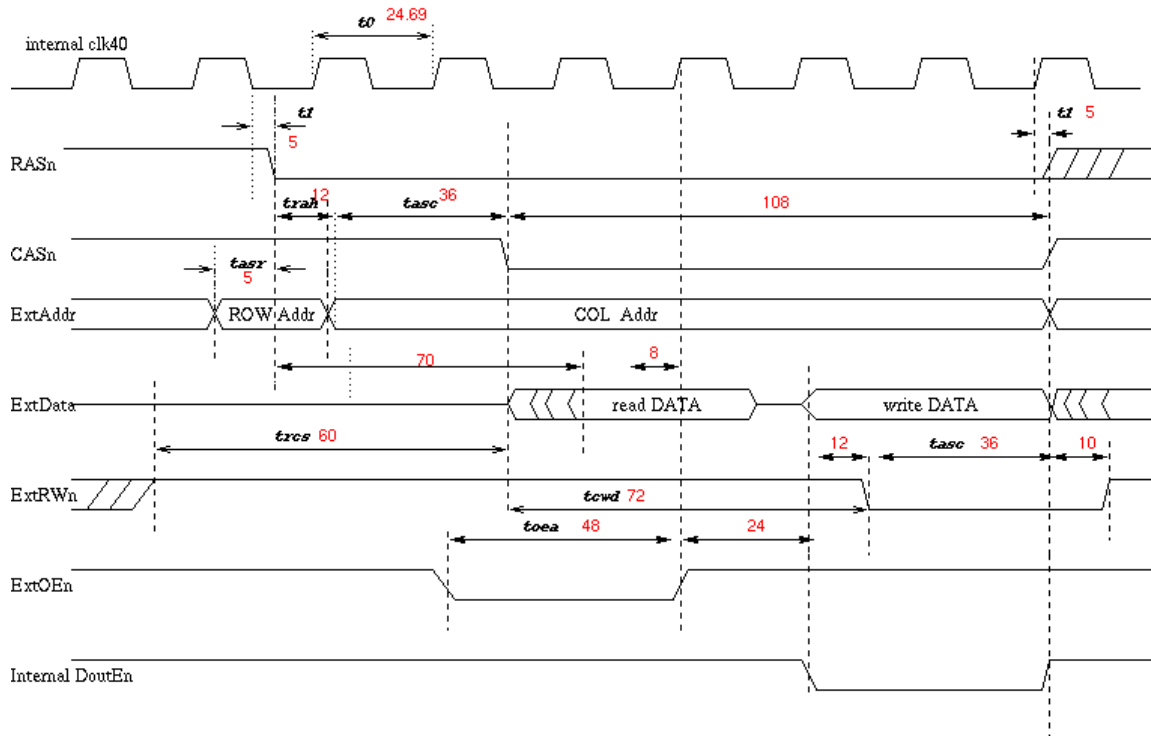


Figure 34. Read-Modify-Write Access for 32 Bit DRAM

(End of Page Mode(DMA) Write with RAS and RAS2 – 60/70ns dram support)

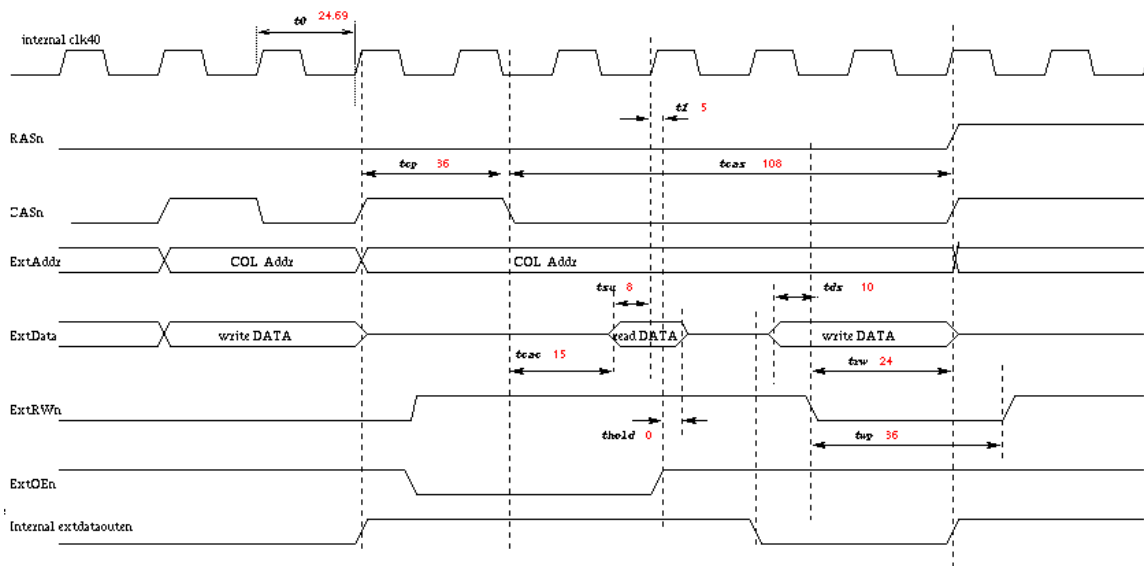


Figure 35. Read-Modify-Write Access for 32 Bit DRAM

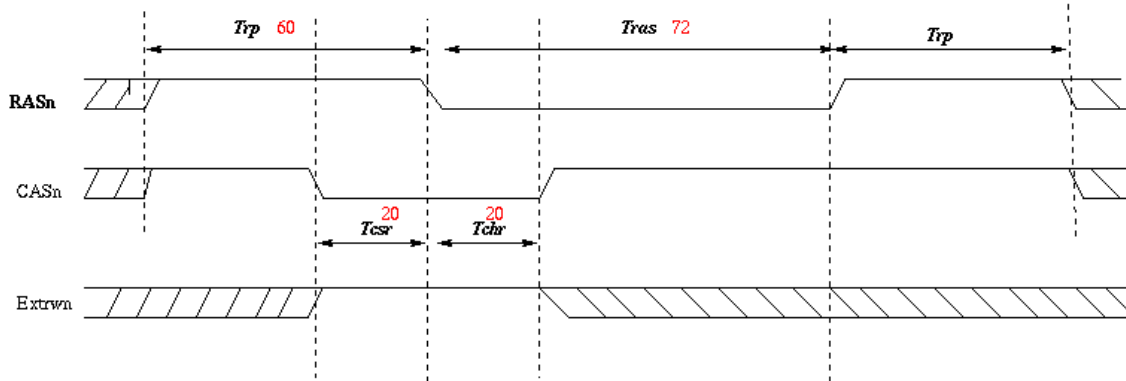
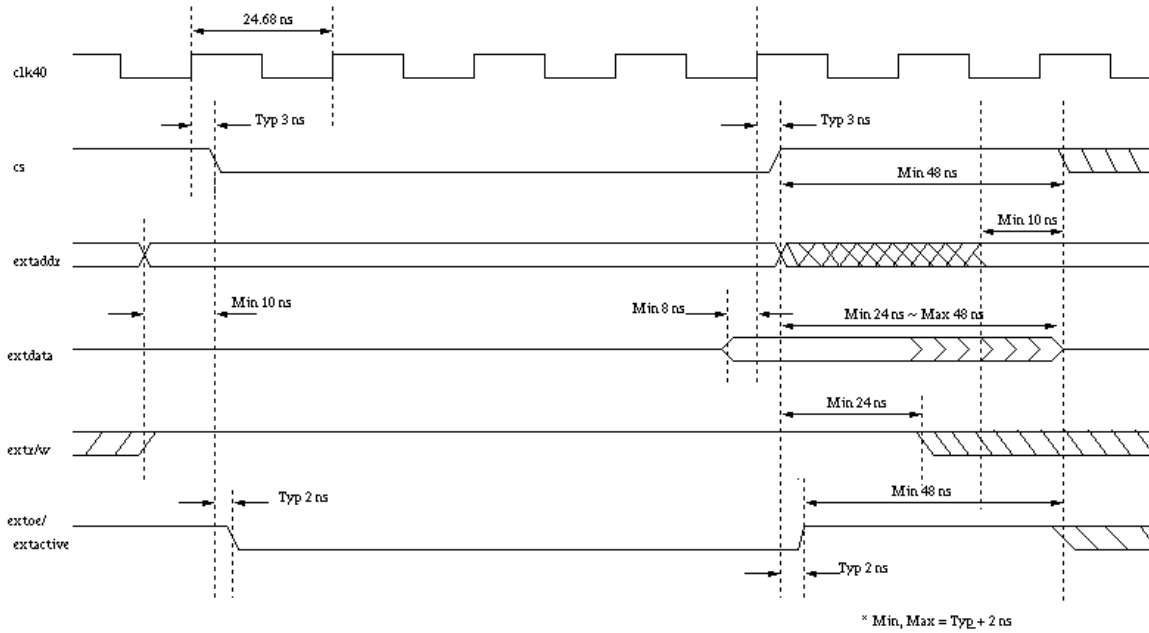


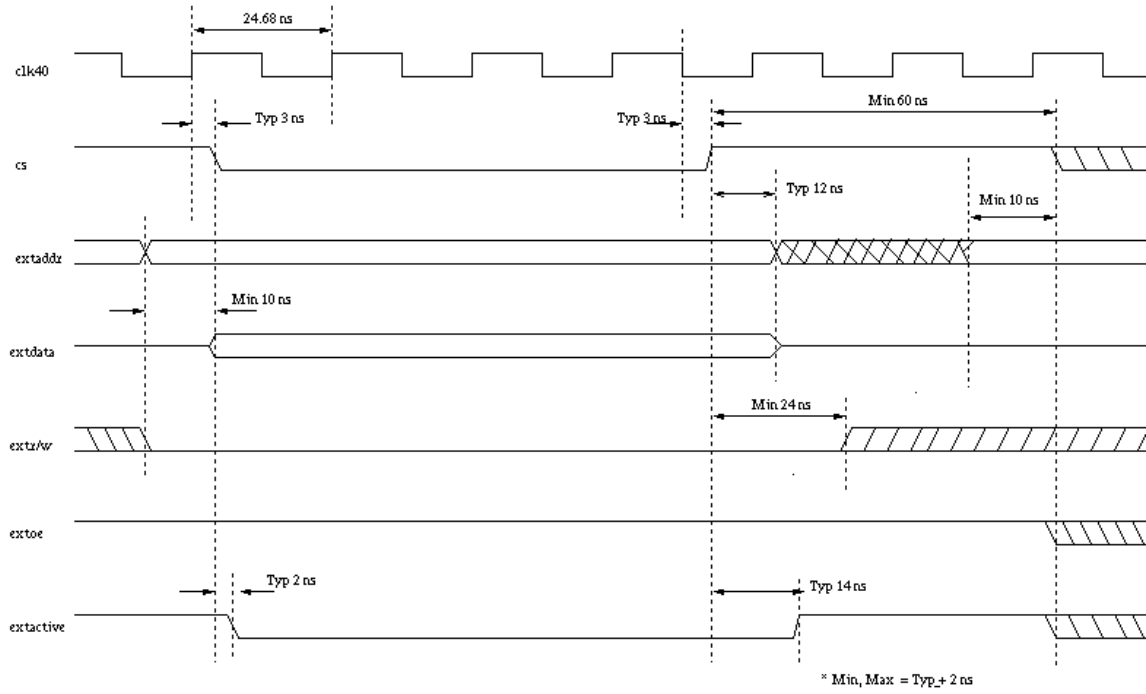
Figure 36.  $\overline{\text{CAS}}$  Before  $\overline{\text{RAS}}$  DRAM Refresh Timing

### 17.2 EBI Interface Timing

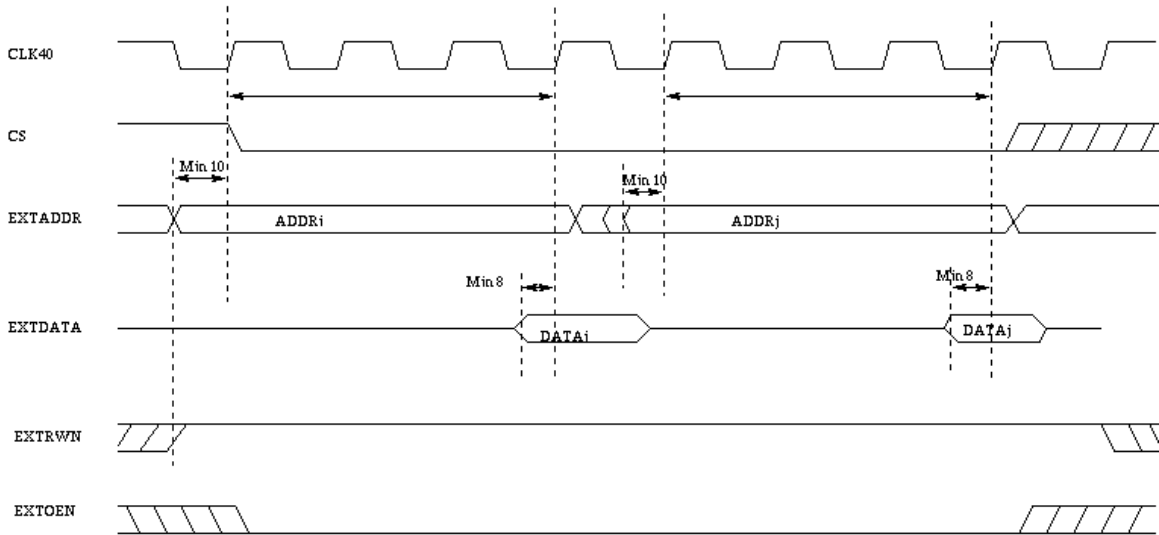


\* This delay could be a max of 24 or 48 nsec. See explanation on tristate requirements in Section 12.1

Figure 37. Extension Bus Single Access Read Timing (4 Wait States)



**Figure 38. Extension Bus Write Timing (4 Wait States)**



**Figure 39. Extension Bus Timing for Multi-Access Mode (3 Wait States)**



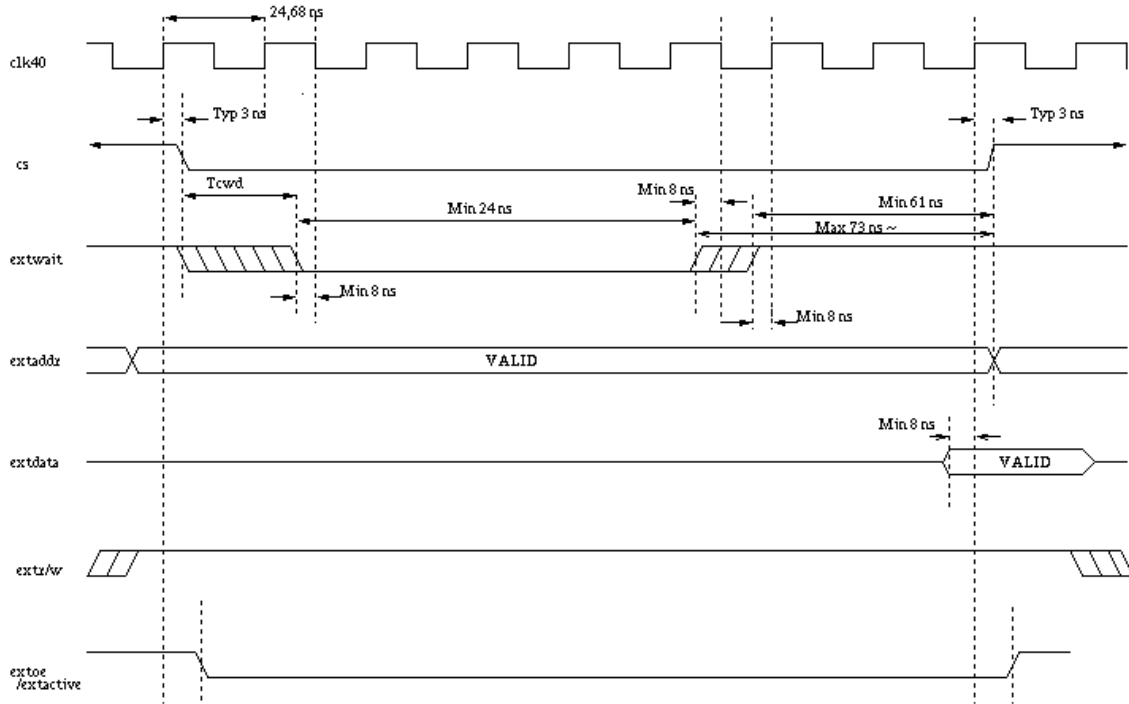
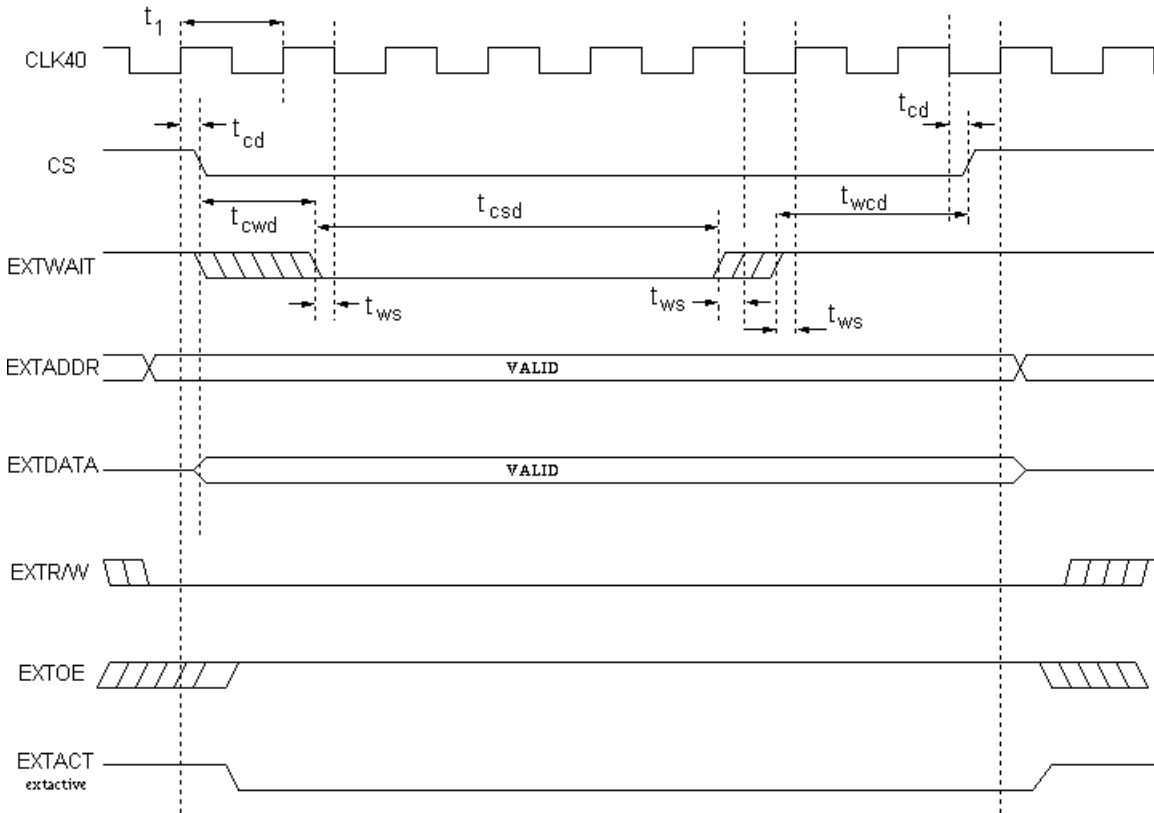


Figure 40. Read with EXTWAIT Active

PARM	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
CLK40	System Clock Frequency			40.5	MHz
$t_1$	Clock Period	24.7			ns
$t_{csd}$	Chip Select Delay		3		ns
$t_{c wd}$	Chip Select to EXTWAIT Delay * = (waits - 3) x $t_1$ , where programmed waits >3	*			ns
$t_{cs}$	Chip Select Active time	24			ns
$t_{ws}$	EXTWAIT Setup to Clock	8			ns
$t_{wcd}$	EXTWAIT to Chip Select Inactive delay	48		61	ns



**Figure 41. Write with EXTWAIT Active**

### 17.3 SDRAM Interface Timing

PARAMETER	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
$t_{ras}$	RAS to RAS Delay (Activation to Activation)		6		clks
$t_{rcd}$	RAS to CAS Delay (Activation to access)		3		clks
$t_{lat}$	Latency Delay		3		clks
$t_{cmax}$	Chip Select/Address Active Delay			9.0	ns
$t_{cmin}$	Chip Select/Address Inactive Delay	3.0			ns
$t_{ds}$	Data Setup time	3.0			ns
$t_{dh}$	Data Hold time	3.0			ns

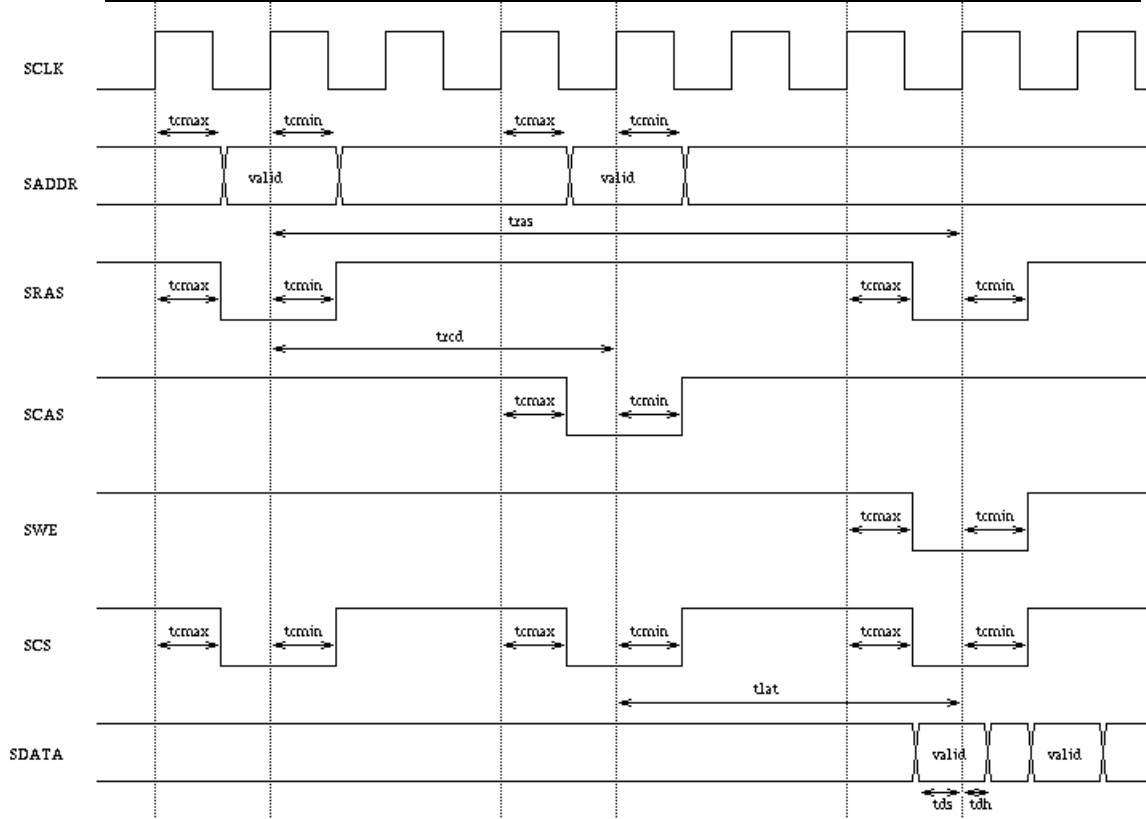


Figure 42. SDRAM Read Cycle

PARAMETER	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
$t_{ras}$	RAS to RAS Delay (Activation to Activation)		6		clks
$t_{rcd}$	RAS to CAS Delay (Activation to access)		3		clks
$t_{rwl}$	Data to RAS Delay (Last data to next Activation)		2		clks
$t_{cmax}$	Chip Select/Address Active Delay			9.0	ns
$t_{cmin}$	Chip Select/Address Inactive Delay	3.0			ns
$t_{dmax}$	Data Valid Delay			9.0	ns
$t_{dmin}$	Data Invalid Delay	3.0			ns

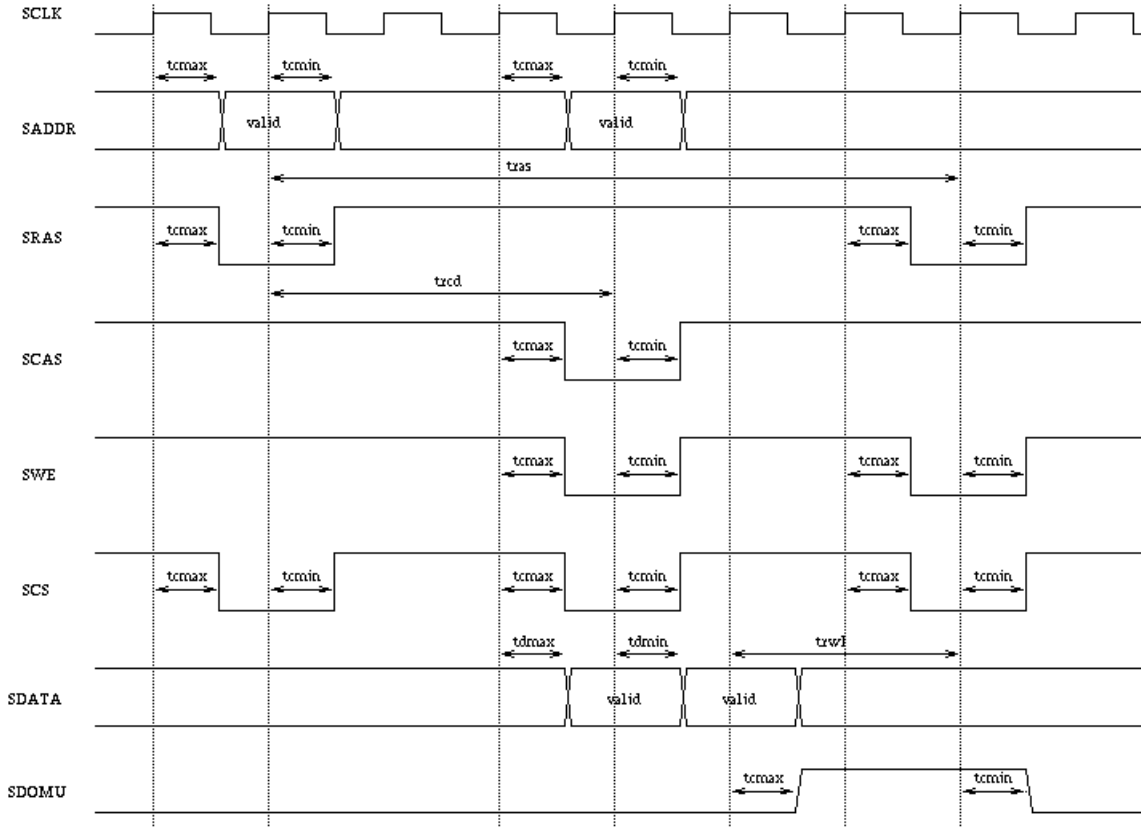


Figure 43. SDRAM Write Cycle

PARAMETER	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
$t_{rrd}$	RAS to RAS Delay (dual bank Activate)		2		clks
$t_{rcd}$	RAS to CAS Delay (for same Bank)		3		clks
$t_{lat}$	Latency Delay		3		clks
$t_{cmax}$	Chip Select/Address Active Delay			9.0	ns
$t_{cmin}$	Chip Select/Address Inactive Delay	3.0			ns
$t_{ds}$	Data Setup Time	3.0			ns
$t_{dh}$	Data Hold Time	3.0			ns

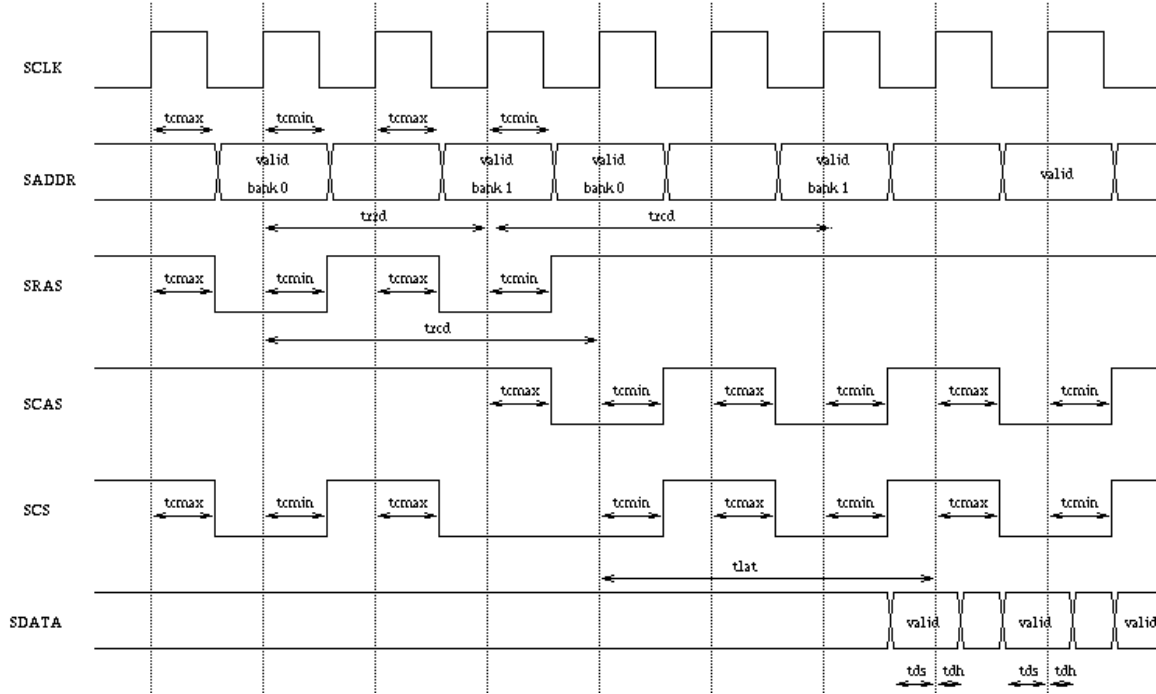


Figure 44. SDRAM Multi Read Cycle

PARAMETER	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
$t_{rrd}$	RAS to RAS Delay (dual bank Activate)		2		clks
$t_{rcd}$	RAS to CAS Delay (for same Bank)		3		clks
$t_{cmax}$	Chip Select/Address Active Delay			9.0	ns
$t_{cmin}$	Chip Select/Address Inactive Delay	3.0			ns
$t_{dmax}$	Data Valid Delay			9.0	ns
$t_{dmin}$	Data Invalid Delay	3.0			ns

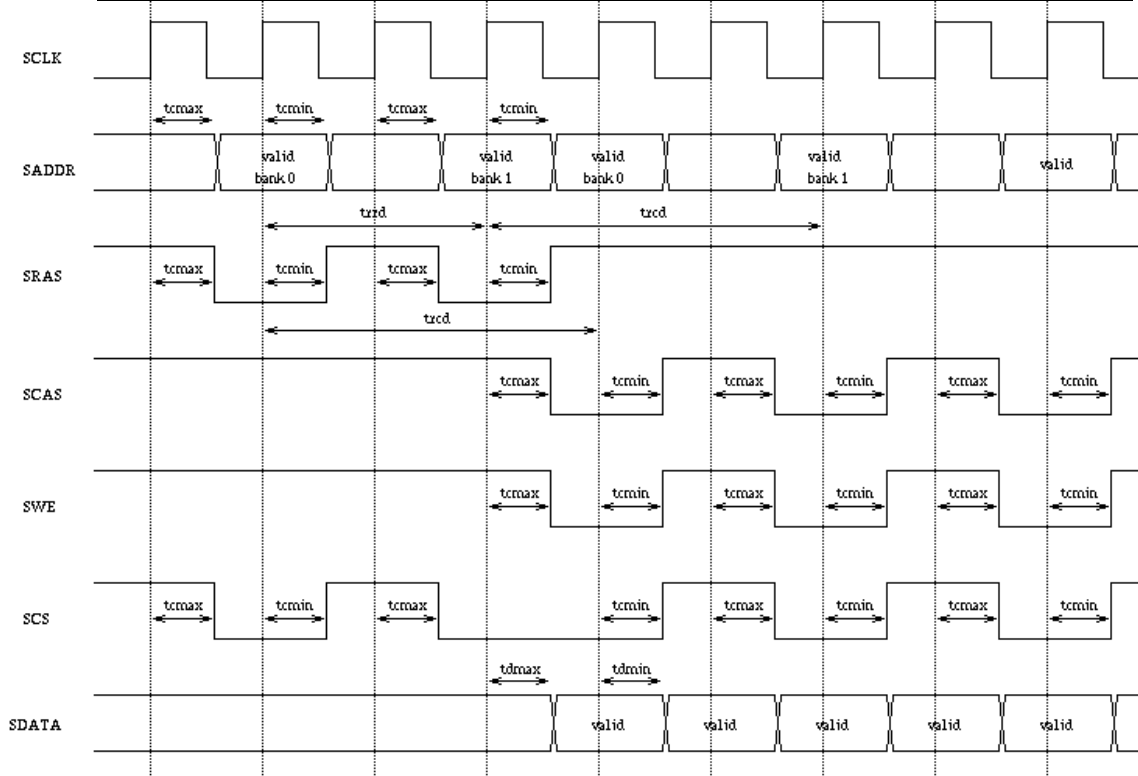
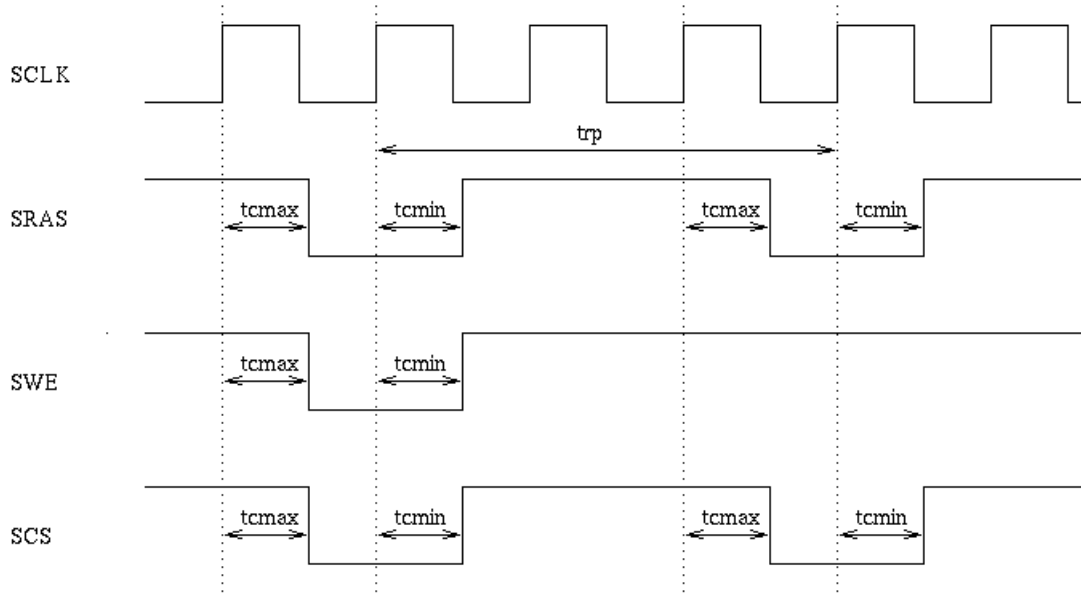


Figure 45. SDRAM Multi Write Cycle

PARAMETER	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
$t_{rp}$	RAS to RAS Delay (Deactivate to Activate)		3		clks
$t_{cmax}$	Chip Select/Address Active Delay			9.0	ns
$t_{cmin}$	Chip Select/Address Inactive Delay	3.0			ns



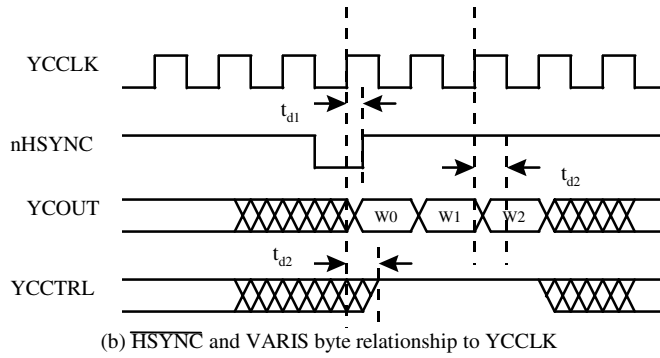
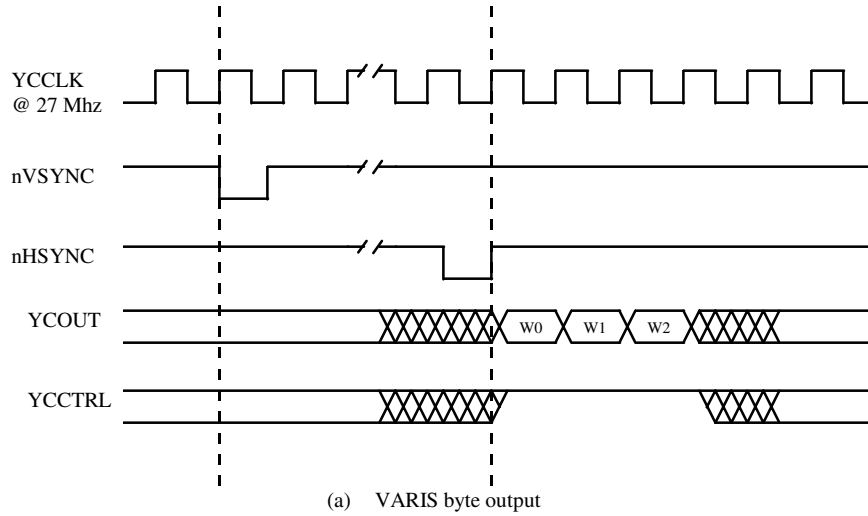
**Figure 46. Successive SDRAM Operations**





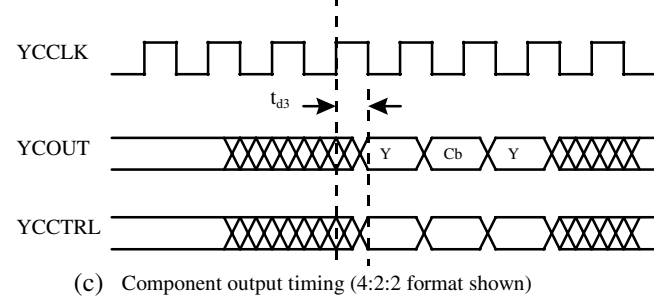
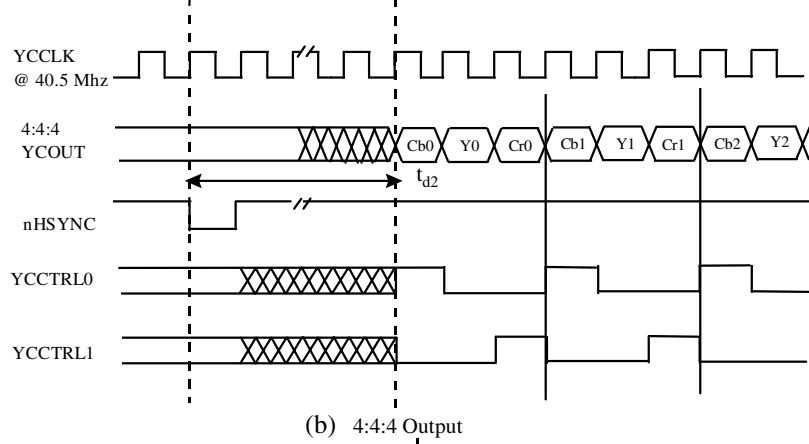
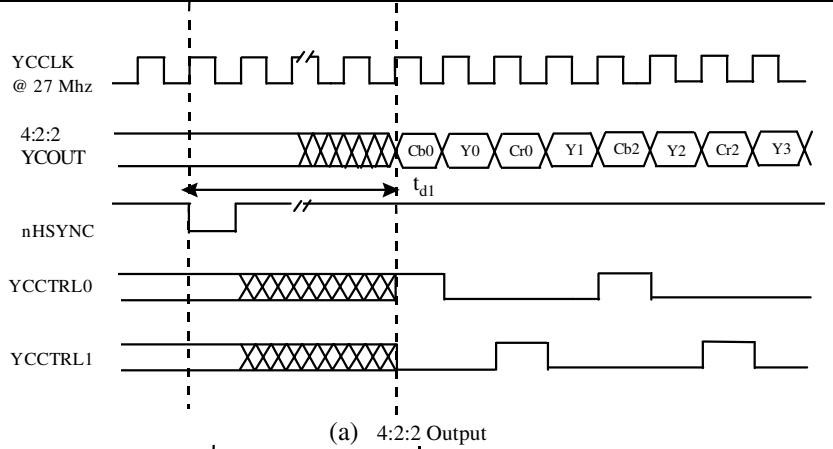
### 17.4 Video Output Timings

PARA	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
$t_{d1}$	YCCLK to HSYNC (internal sync mode)			5	ns
$t_{d2}$	YCLK to YCOUT[7:0], YCCTRL[1:0] valid	5		25	ns



**Figure 47. VARIS Code Output Timing**

PARA	PARAMETER DESCRIPTION		MIN	TYP	MAX	UNIT
t <sub>d1</sub>	Data Start Timing (4:2:2 Data Output)	NTSC	f <sub>clk</sub> = 27 MHz	232/f <sub>clk</sub>		s
		PAL	f <sub>clk</sub> = 27 MHz	248/f <sub>clk</sub>		
t <sub>d2</sub>	Data Start Timing (4:4:4 Data Output)	NTSC	f <sub>clk</sub> = 40.5 MHz	232/f <sub>clk</sub>		s
		PAL	f <sub>clk</sub> = 40.5 MHz	248/f <sub>clk</sub>		
t <sub>d3</sub>	YCLK to YCOUT[7:0], YCCTRL[1:0]		5		25	ns



**Figure 48. Digital Video Output Timing**

### 17.5 HSDI Timing Diagrams

PARAM	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
CLK40	System Clock Frequency			40.5	MHz
$t_1$	Clock Period	24.7			ns
$t_2$	Clock High Time	10			ns
$t_3$	Clock Low Time	10			ns
$t_4$	_SIG1, _SIG2 Valid Delay	2		12	ns
$t_5$	_SIG1, _SIG2 Hold Time	2		12	ns
$t_6$	_SIG[4:6], _DATA[7:0] Setup Time	10			ns
$t_7$	_SIG[4:6], _DATA[7:0] Hold Time	3			ns
$t_8$	HSDI_SIG3 Setup Time	10			ns
$t_9$	HSDI_SIG3 Hold Time	3			ns

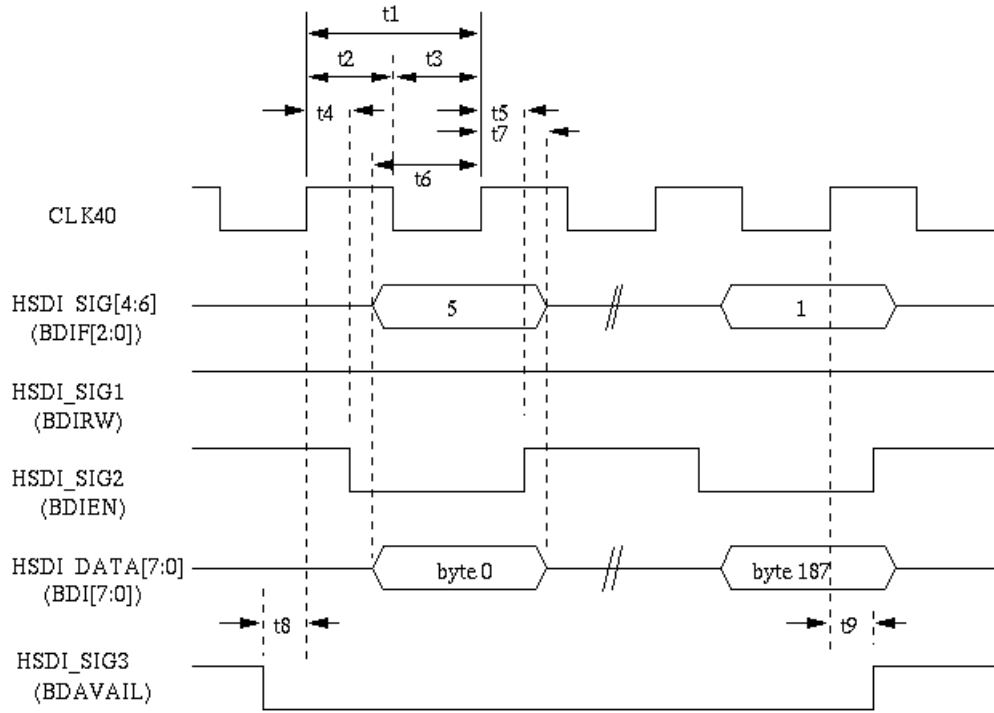


Figure 49. 1394 Read Cycle Timing

PARAM	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
CLK40	System Clock Frequency			40.5	MHz
$t_1$	Clock Period	24.7			ns
$t_2$	Clock High Time	10			ns
$t_3$	Clock Low Time	10			ns
$t_4$	$\_SIG1, \_SIG2, \_SIG[4:6], \_DATA[7:0]$ Valid Delay	2		12	ns
$t_5$	$\_SIG1, \_SIG2, \_SIG[4:6], \_DATA[7:0]$ Hold Time	2		12	ns
$t_6$	HSDI_SIG7 Setup Time	10			ns
$t_7$	HSDI_SIG7 Hold Time	3			ns
$t_8$	HSDI_SIG7 Pulse Width	35			ns

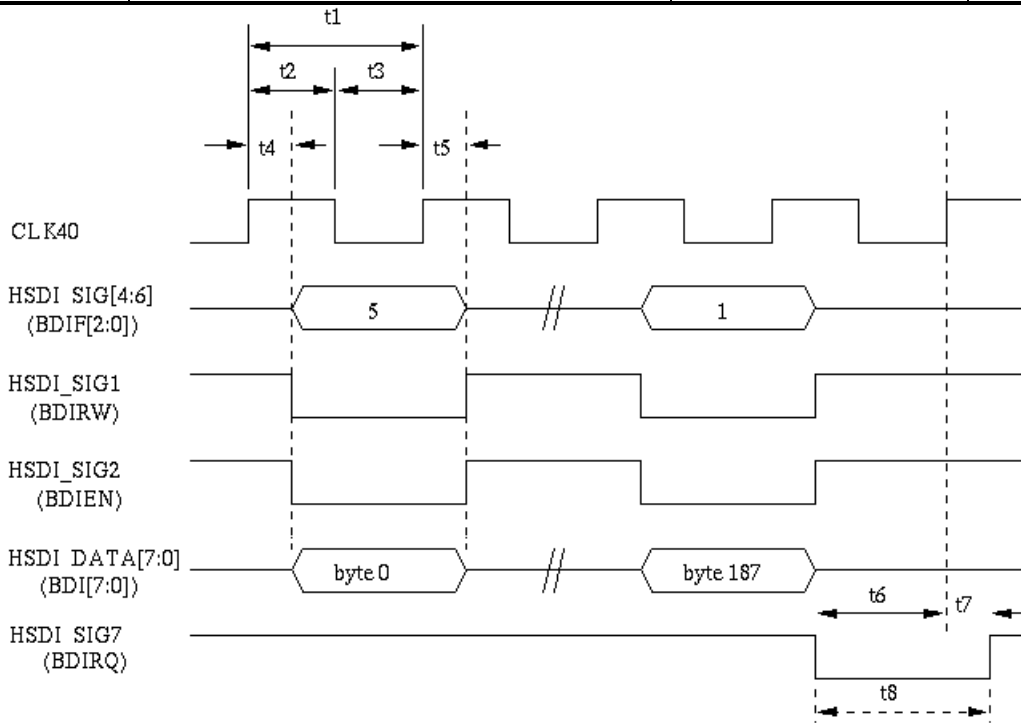
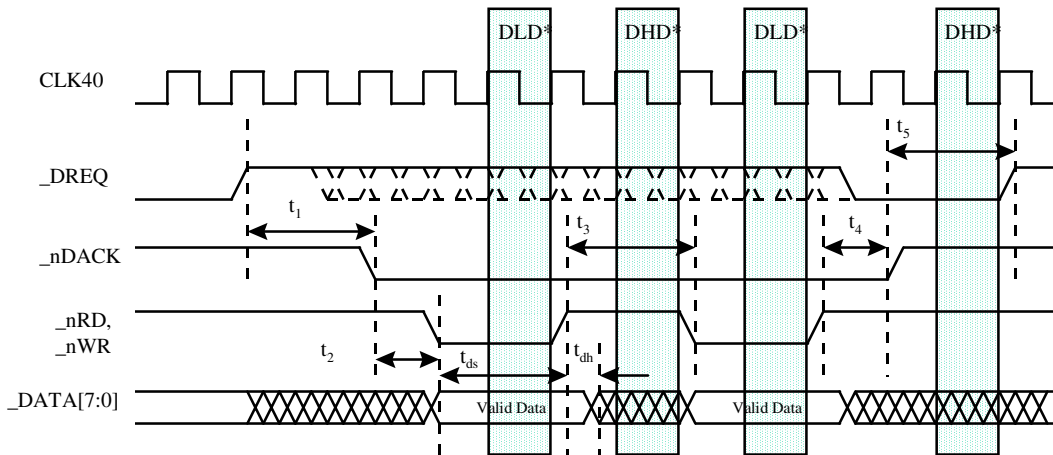


Figure 50. 1394 Write Cycle Timing

### 17.6 External DMA

PARAM	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
CLK40	System Clock Frequency		40.5		MHz
$t_{clk}$	Clock Period		24.7		ns
$t_1$	DREQ Active to DACK ActiveClock Period	24.7	$2 t_{clk} + 4$		nsns
$t_2$	DACK Active to RD or WR ActiveClock High Time	10	$t_{clk} + 4$		nsns
$t_3$	RD/WR Inactive to RD or WR ActiveClock Low Time	10	$(DHD)t_{clk} + 4 *$		nsns
$t_4$	RD or WR Inactive to DACK Inactive_SIG1, _SIG2, _SIG[4:6], _DATA[7:0] Valid Delay	2	$t_{clk} + 4$	12	nsns
$t_5$	DACK Inactive to DREQ Active_SIG1, _SIG2 _SIG[4:6], _DATA[7:0] Hold Time	2	$(DHD)t_{clk} + 4 *$	12	nsns
$t_{rds7}$	Read Data Setup time HSDI_SIG7 Hold Time	3	$(DLD)t_{clk} + 10 *$		nsns
$t_{rdh8}$	Read Data Hold time HSDI_SIG7 Pulse Width	35	$(DHD)t_{clk} + 4 *$		nsns
$t_{wds}$	Write Data Setup time		$(DLD)t_{clk} + 10 *$		ns
$t_{wdh}$	Write Data Hold time		$(DHD)t_{clk} + 4 *$		ns

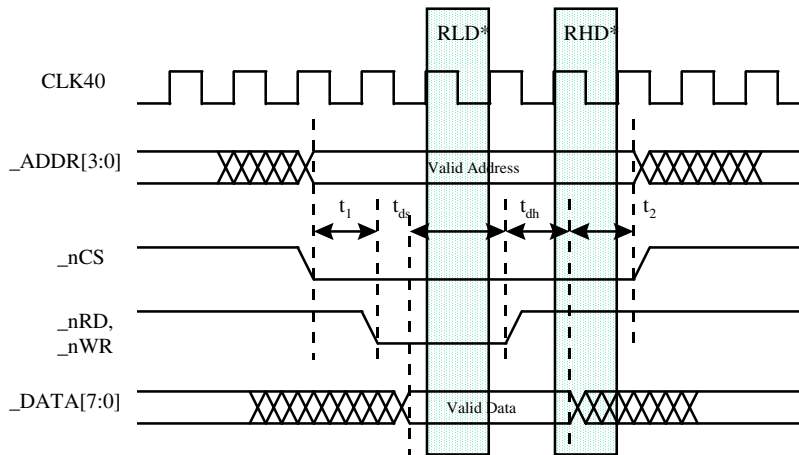
\* See Description of DHD/DLD for details about these programmable delay values.



**Figure 51. External DMA Cycle Timing**

PARM	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
CLK40	System Clock Frequency		40.5		MHz
$t_{clk}$	Clock Period		24.7		ns
$t_1$	CS Active to RD or WR ActiveClock Period	24.7	$t_{clk} + 4$		nsns
$t_2$	RD or WR Inactive to CS, ADDR InvalidClock High Time	10	$(RHD)t_{clk} + 4 *$		nsns
$t_{rds6}$	Read Data Setup time HSDI_SIG7 Setup Time	10	$(RLD)t_{clk} + 10 *$		nsns
$t_{rdh7}$	Read Data Hold time HSDI_SIG7 Hold Time	3	$(RHD)t_{clk} + 4 *$		nsns
$t_{wds}$	Write Data Setup time		$(RLD)t_{clk} + 10 *$		ns
$t_{wdh}$	Write Data Hold time		$(RHD)t_{clk} + 10 *$		ns

\* See Description of RHD/RLD for details about these programmable delay values.



**Figure 52. External DMA "Register Access" Timing**

### 17.7 Input Stream Timing

PARAMETER	PARAMETER DESCRIPTION	MIN	TYP	MAX	UNIT
	DCLK Clock Frequency			9.1	MHz
$t_1$	Clock Period	109.89			ns
$t_2$	Clock High Time	50			ns
$t_3$	Clock Low Time	50			ns
$t_4$	PACCLK, BYTE_START, DATAIN[7:0], DERROR Setup Time	10			ns
$t_5$	PACCLK, BYTE_START, DATAIN[7:0], DERROR Hold Time	2			ns

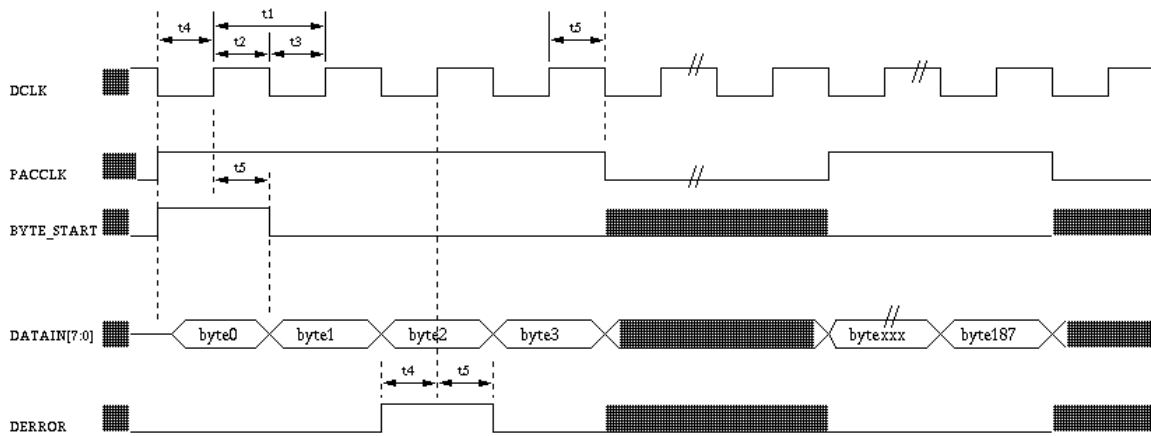


Figure 53. Input Interface Timings

## 17.8 DC Characteristics

Recommended commercial operating conditions

Parameter	Description	MIN	NOM	MAX	Units
VDD3_3V	Digital Supply Voltage 3.3V	3.1	3.3	3.5	V
VDD2_5V	Digital Supply Voltage 2.5V	2.4	2.5	2.7	V
VCC0A-VCC3A	Video DAC Supply Voltage	3.1	3.3	3.5	V
VCCLA1-VCCLA2	Audio PLL Supply Voltage	3.1	3.3	3.5	V
VI	Signal Input Voltage	0.0		VDD3_3V	V
VO	Signal Output Voltage	0.0		VDD3_3V	V
VIH	High Level DC input Voltage	2.0		VDD3_3V	V
VIL	Low Level DC input Voltage	0.0		0.8	V
TA	free-air temp range	0.0		70	°C
TJ	Junction temperature range	0.0		115	°C

Electrical Characteristics over recommended operating free-air temperature range (unless otherwise noted)

Param	Description	Test Conditions	MIN	NOM	MAX	Units
VOH	High-Level Output Voltage	IOH = -4mA			VDD3_3V-0.6	V
VOL	Low-Level Output Voltage	IOL = 4mA			0.5	V
IIL	Low-Level input current	VI = VIL			+/- 20	µA
IIH	High-Level input current	VI = VIH			+/- 20	µA
IOZ	Off-state output current	VI = VCC or 0 V			+/- 20	µA
ICC	Digital Supply current	VDD2_5V = 2.7V and CLK27= 27 MHZ			480	mA
ICC2	Digital Supply Current	VDD3_3V = 3.6V and CLK27 = 27 MHZ			110	mA
Ci	Input Capacitance				8	pf
Co	Output Capacitance				8	pf



### 17.9 SDRAM Memory Map for First Silicon firmware

#### 17.9.1 PAL mode Memory Map

Address	Content	Size *	Comments
CDFE FFFF	Expansion Memory Region (OSD/User Data)	2 MB	
CC20 0000			
CC1F FFFF	OSD or other use	230,400 B or 225 kB	
CC1C 7C00			
CC1C 7BFF	B Frame	230,400 B or 225 kB	
CC18 F800			
CC18 F7FF	Second Reference Frame	622,080 B or 607.5 kB	
CC0F 7A00			
CC0F 79FF	First Reference Frame	622,080 B or 607.5 kB	
CC05 FC00			
CC05 FBFF	Video Buffer	2,744,320 b or 335 kB	
CC00 C000			
CC00 BFFF	Audio Buffer	65,536 b or 8 kB	
CC00 A000			
CC00 9FFF	Video Microcode	36,864 B or 36 kB	
CC00 1000			
CC00 0FFF	Tables and FIFOs	3 kB	
CC00 0400			
CC00 03FF	Pointers	1 kB	
CC00 0000			

\* B = Bytes, b = Bits

\* See Table 3 through Table 5

**17.9.2 NTSC mode Memory Map**

<b>Address</b>	<b>Content</b>	<b>Size *</b>	<b>Comments</b>
CDFE FFFF	Expansion Memory Region (OSD/User Data)	2 MB	
CC20 0000			
CC1F FFFF	OSD or other use	241,408 B or 235.75 kB	
CC1C 5100	B Frame	426,240 B or 416.25 kB  (.82 Frames)	
CC1C 50FF			
CC15 D000	Second Reference Frame	518,400 B or 506.25 kB	
CC15 CFFF			
CC0D E700	First Reference Frame	518,400 B or 506.25 kB	
CC0D E6FF			
CC05 FE00	Video Buffer	2,748,416 b or 335.5 kB	
CC05 FDFE			
CC00 C000	Audio Buffer	65,536 b or 8 kB	
CC00 BFFF			
CC00 A000	Video Microcode	36,864 B or 36 kB	
CC00 9FFF			
CC00 1000	Tables and FIFOs	3 kB	* See Table 3 through Table 5
CC00 0FFF			
CC00 0400	Pointers	1 kB	
CC00 03FF			
CC00 0000			

\* B = Bytes, b = Bits

## IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to or to discontinue any semiconductor product or service identified in this publication without notice. TI advises its customer to obtain the latest version of the relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products to current specifications in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Unless mandated by government requirements, specific testing of all parameters of each device is not necessarily performed.

TI assumes no liability for TI applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, express or implied, is granted under any patent right, copyright, mask work right, or any other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be used.