Full MPEG1 video and audio decoder (FMPEG) SAA7131

1. FEATURES

Features of the FMPEG:

- Fully ISO 11172-1, -2 and -3 compatible single chip MPEG1 video and audio decoder
- Decodes a video and/or audio stream with the system layer or without a system layer (single bitstream mode)
- Decodes data streams as described in the Video-CD 2.0 specification
- The video decoder processes video streams that meet the constrained parameter set as defined in the MPEG1 video standard.
- ISA bus host interface (16 bit IO device with software programmable IO ports and interrupt level)
- The video decoder handles bitrates up to 5 Mbit/sec
- Decodes SIF/CIF video formats (352 * 288, 320 * 240)
- Full screen standard VGA mode (640 * 480) capability using a line repeat mechanism
- Windowing capabilities
- TV-diaplay mode with 50 Hz or 60 Hz field frequency
- Supports different DMSD (Digital Multi Standard Decoder) and VMC (Vesa Media Channel) video output formats
- Provides a special still picture mode allowing decoding of a picture of 704 * 480 pixels
- YC_RC_B 422, RGB 888, RGB 565 or RGB 555 video output
- Supports different video clocks from 22 MHz to 32 MHz
- Genlock capabilities for video
- Real-time horizontal pixel interpolation with 1-2-1 filter or pixel repeat function
- Possibility for read back of reconstructed video data
- Supports audio sample rates of 32 KHz, 44.1 KHz and 48 KHz
- Supports IIS (Philips) and Sony serial audio output format with 16, 18 or 20 bits accuracy
- Allows play back of PCM audio data (16 bits/sample) via ISA bus
- Possibility of mixing/attenuating decoded MPEG audio (or PCM audio data from ISA bus) before external audio is added
- Supports external serial audio input in IIS (Philips) or Sony format with 8, 16, 18 or 20 bit sample accuracy and a sample frequency equal to the sample frequency of the decoded/PCM audio and mixing this external audio with decoded or PCM audio
- Possibility to read back decoded audio

2. GENERAL DESCRIPTION

2.1. System Envoirement

The SAA7131 (FMPEG) is a MPEG1 video and audio decoder for 80x86 personal computers with ISA bus, which requires only 4 Mbit of DRAM as a working memory and is designed to operate in a great variety of applications. The FMPEG can decode MPEG1 streams with system layer or streams without a system layer (single bitstream mode) as well as data streams as described in the Video-CD 1.1/2.0 specification. The reconstructed video and audio data can be output in selectable formats in order to support different applications.

Figure 1. shows the FMPEG used in a PC envoirement. The device receives the compressed data stream which is stored e.g. on a CD-ROM via the ISA bus interface. Coding information and the decoder status can be accessed by the host through a number of registers.

The reconstructed video data is output via the video bus using YC_RC_B or RGB formats. The Video output can be synchronized to VGA graphics (VGA-display mode) as shown in Figure 1. or to the DMSD bus (TV-display mode). The VGA-display mode provides full screen display of video data or output of video data in a window.

The reconstructed audio data is output either in IIS (Philips) and Sony serial output format with 16, 18 or 20 bits accuracy and sample frequency depending on the coded audio data.

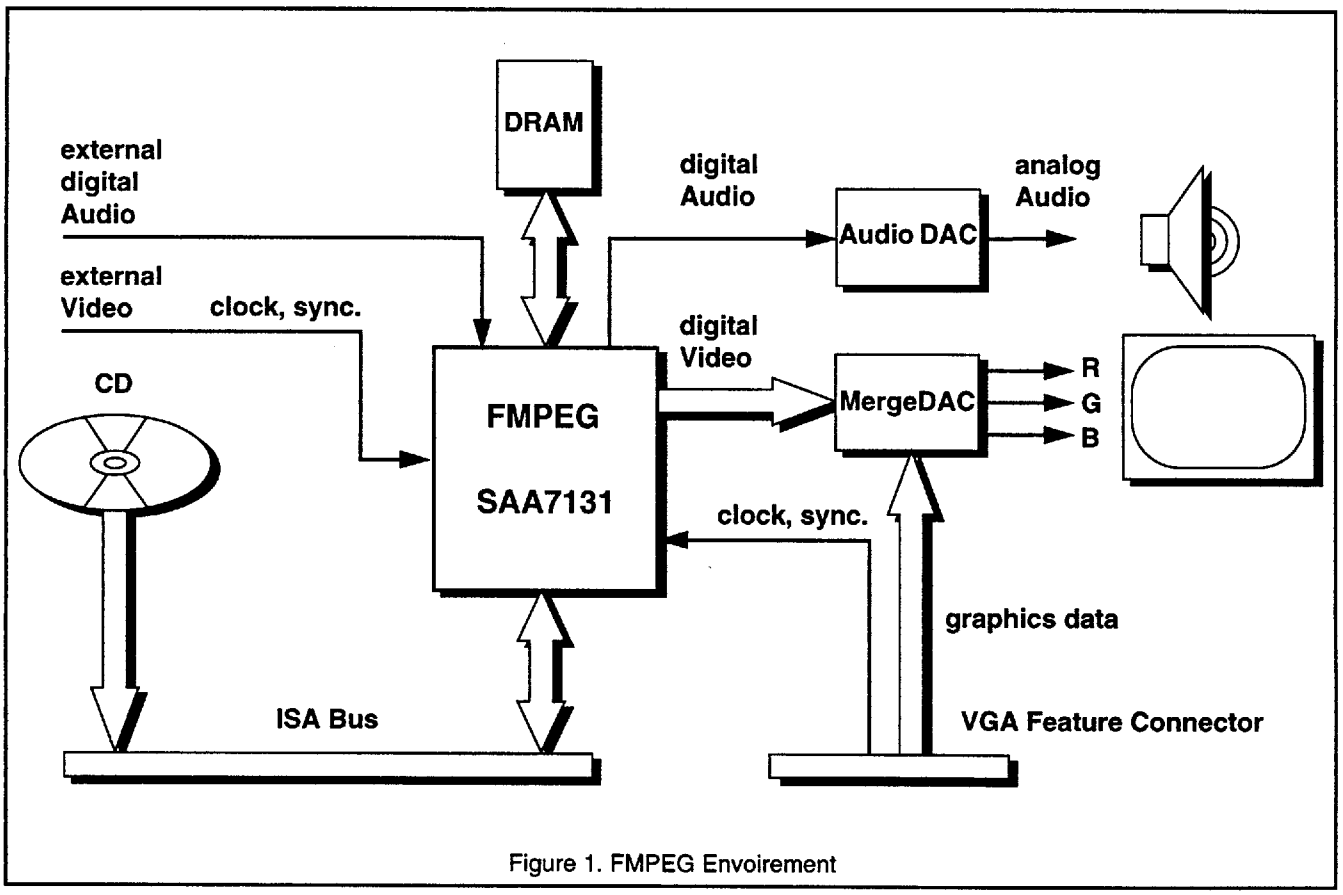


Figure 1. FMPEG Envoirement

Host Interface

The FMPEG IC communicates with the HOST system via an ISA bus interface. Therefore two 16 bit IO-ports and interrupts with four possible interrupt levels are supported. One IO-port is used to select the internal address, the other to read or write the cor-

responding data. Both the IO-port base addresses and the interrupt levels are software programmable. The ISA bus interface provides the following functions:

- Controlling the FMPEG by the host (access to registers and memories)
- Transfer the compressed data stream to the FMPEG (eg. from CD-ROM drive)
- Reading back reconstucted video and audio data
- Play back of PCM audio data

Decoding Video

The FMPEG video decoder can decode one of the 16 video streams which are allowed to be multiplexed in an ISO11172 stream. The stream number can be changed at any time. The video decoder will decode/display data of the new stream with the highest priority.

The FMPEG provides two main display modes.

1. TV-display mode:
   In TV-display mode the video output is synchronized with synchronization signals from the DMSD bus. This allows the FMPEG video output to be genlocked to a second video source.
2. VGA -display mode:
   In VGA -display mode the FMPEG obtains the synchronization signals from the standard VGA graphic. This allows to merge the VGA graphics with the reconstructed MPEG video.

The video output is selectable. RGB 888, RGB 565 or RGB 555 as well as YC_RC_B 422 video outputs are supported with different signal/pin mappings of the 24 video data lines in order to use DMSD or VMC (VESA Media Cannel) output formats.The clock signals Additionally the reconstructed video data can be read back via the ISA bus interface to be used for other purposes.

Upsampling is supported by the FMPEG in horizontal as well as in vertical direction. Horizontal upsampling can be done by pixel repeat or pixel interpolation. Vertical upsampling is done by line repeat. Additionally the FMPEG allows to read out the reconstructed video data in interlaced mode, which means reading even lines in even fields and odd lines in odd fields when used. This allows to generate different output formats. E.g.:

- CCIR 601 format:
  The FMPEG is able to convert the reconstructed video in MPEG SIF format to the CCIR 601 format.
- VGA Full screen format:
  In full screen mode the reconstructed MPEG video fully fills the VGA screen. The full screen format is 640 pixels by 480 lines.
- VGA Window format:
  In window mode the position and the size of a rectangular area on the screen is defined where the active video is shown. Inside this window a part of reconstructed picture or the whole reconstructed picture is displayed.

For interfacing with other video components a special output pin (PXQ) is provided that flags a predefined area within in the output. This signal allows e.g. in VGA window mode switching between video and graphics on the VGA screen.

Decoding Audio

The audio decoder is a full layer I and layer II decoder. It can decode all specified bitrates, modes and sample frequencies. Both the 50/15 microsec. and the CCITTJ.17 deemphasis are supported too.

The serial output can be either in Philips IIS or Sony format. The number of bits per sample is 32. The output accuracy of the audio samples is rounded to 16, 18 or 20 bits, whereby sample frequencies of 32 KHz, 44.1 KHz and 48 KHz are supported. The audio decoder is the timing master and generates the bit clock and word clock.

Mixing and attenuation of the left and right channel audio data is supported. Additionally the audio decoder allows combining the reconstructed audio stream with an external digital audio source (Figure 1.). The external audio serial input of the FMPEG can be either in Philips IIS or Sony format with possible sample accuracies are 8, 16, 18 or 20 bits. The sample frequency of the

external audio input is always identical to serial output.

The FMPEG provides the possibility of reading back (decoded) PCM data. In this case the audio decoding process is slaved to the read back process. Because of this non-real time effect, the audio is not audible and audio decoder will automatically output all zeros on the serial output.

The FMPEG is also able to play back PCM audio data. The PCM audio data can be mixed and attenuated as well as combined with external digital audio in the same way as if MPEG audio data is processed.

System timing

The FMPEG IC is operating with different clocks:

1. System clock:
   The system clock is generated by a 40 MHz crystal.
2. Video clock:
   If TV-display mode, where DMSD specific synchronization signals are used, is chosen the line locked clock (22 MHz - 32 MHz) which is obtained from DMSD bus has to be applied to the FMPEG.
   If VGA-display mode is selected the VGA pixel clock which is provided from the standard VGA graphics has to be used. In this mode VGA synchronization sigals are used as well.
3. Audio clock:
   The audio clock of 256*Fs (with Fs = 32 KHz, 44.1 KHz or 48 KHz sample frequency) is generated internally by the FMPEG or can be applied externally. Additionally an extra clock signal can be applied to the ASC pin of the FMPEG which can be used as a source for generating the needed audio clock.
4. MPEG system time clock:
   The MPEG system time clock (STC) of 90 KHz is generated internally by the FMPEG itself and output on the NCLU pin for external use.

2.2. System Functions

Figure 2. shows a blockdiagram of the FMPEG with it's main functional blocks and 4 Mbit external DRAM as the working memory.

Video

The video data input parser selects the required video stream of compressed data packets from the incoming ISO tream and stores the data in the 57 kbyte video FIFO located in the DRAM. The size can be adjusted in case of larger still pictures need to be stored. All system layer information is removed prior to storing the data into the video FIFO. The system clock reference (SCR), decoding time stamp etc. is passed to the system controller. It is also possible to let the input parser search for reserved video words, e.g. a search for sequence can be initiated. Then all leading data is discarded.

The datasorter retrives data from the video FIFO and uses variable and fixed length codes to sort the data into picture information present to the system controller and macro block data presented to the IDCT and the frame reconstructor.

The frame reconstructor uses only IDCT data in case of I-frames to be reconstructed or IDCT data and data of previously and / or next reconstructed frames depending on the reconstruction of P- or B-frames. The reconstructed video data is stored in one of the three video buffers. The assignment of the buffer numbers is done by the system controller.

The video generator retrives the reconstructed video data from one of the three video buffers and presents it in different ways according to the information programmed in the system controller. All windowing and other display related features are handled by this module. The video generator is either synchronized to YUV Bus synchronization signals to support the FMPEG being genlocked to an external video source or is synchronized to VGA synchronization signals in order to allow the video data being directly merged to the computer graphics.

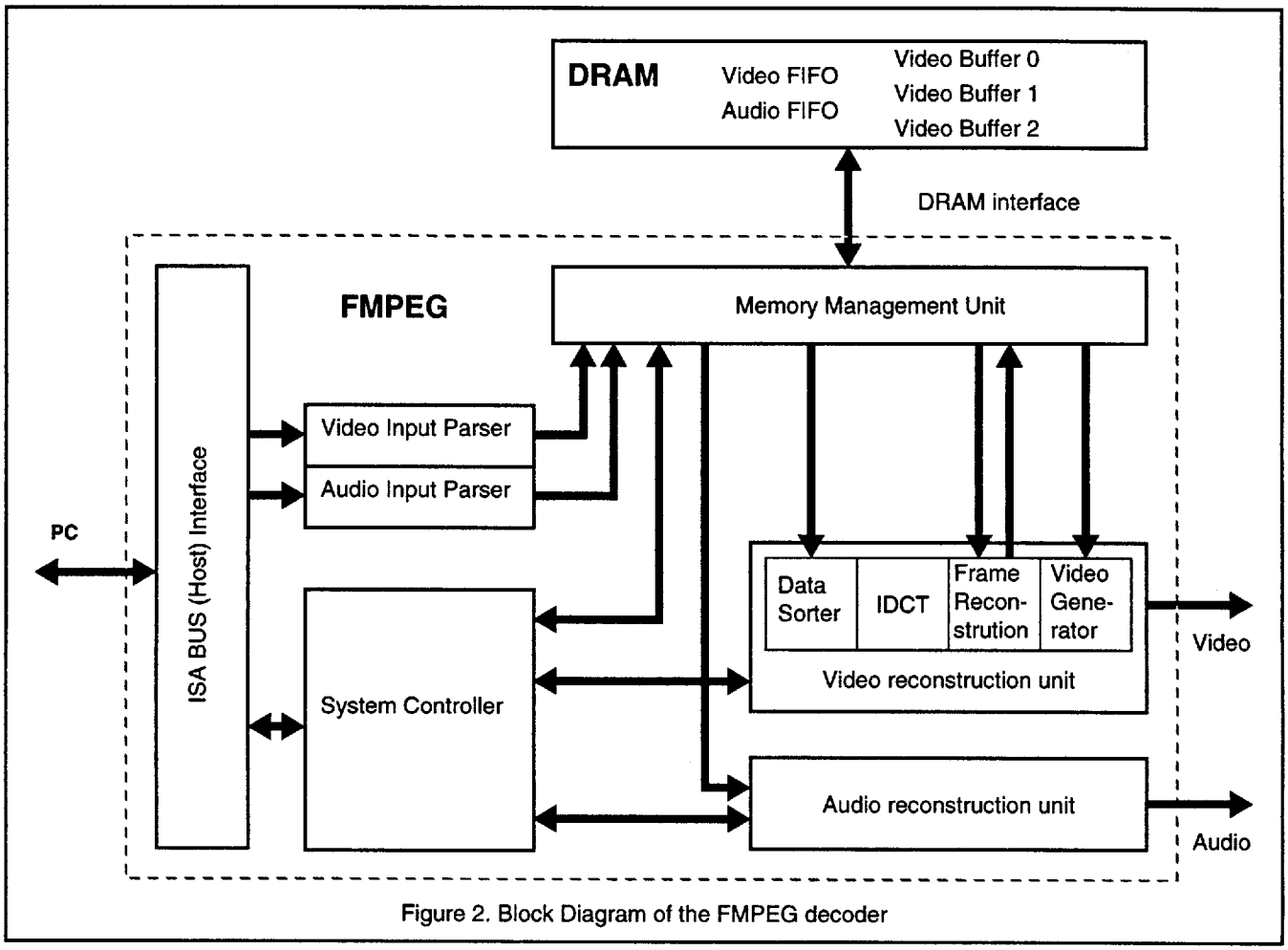


Figure 2. Block Diagram of the FMPEG decoder

Audio

The audio input parser selects the required audio stream (compressed data packets) from the incoming ISO stream and stores the data into the 8 kbyte audio FIFO which is located in the DRAM. The packcode and the system header is removed before the data is stored in the audio FIFO. The system clock reference (SCR) is passed to the system controller. The input parser performs a consistency check on pack and packet information (checks the packet length).

The audio reconstruction unit reconstructs the audio data and performs the required deemphasis (50/15 microsec. or the CCITTJ.17 deemphasis) on the audio data. The audio reconstruction unit furthermore performs attenuation of the reconstructed audio data and mixing with external audio and is generating the needed audio data output format (IIS or Sony).

Memory management

The Memory management unit (MMU) takes care of all arbitration to and from the DRAM, but also calculates all necessary addresses. Together with the CAS before RAS refresh scheme the MMU safeguards all accesses to the DRAM. FIFO underflow and overflow detection is done for both the Video and the audio FIFO.

System controller

The system controller is a microcode based RISC processor. The system controller:

- interprets all host commands
- signals the host on request all relevant decoding and status information and
- koordinates the FMPEG internally

At system layer the system controller compares the decoding time stamp versus the system clock reference counter and decides to start decoding the next audio frame. In case of the time stamp information indicates a positive time gap the decoding process will stop and a soft mute sequence (10 to 16 ms before a complete mute) will be started. Every 32 audio samples the system controller checks again if the system clock reference exeeds the decoding time stamp.

3. IO INTERFACES

3.1. ISA Bus Interface

The FMPEG communicates with the host via the ISA Bus Interface. Therefore the FMPEG supports 16 bit IO read and IO write accesses as well as interrupts. Only two IO ports are needed. Via the first IO port the internal addresses of the FMPEG are selected. The second IO port is used for data transfer.

Interrupts are indicated to the host via one of the interrupt lines IRQ [15, 12, 11, 10]. The interrupt level is programmable. An interrupt remains asserted util the interrupt status is read.

The Data Output ENable signal (DOEN) is provided to enable external data line buffers which are assumed to be used for buffering the data lines D [15 ... 00].

The pins CNR0 and CNR1 are containing the two LSB's of the chip number, which is needed to program the two IO port addresses and the interrupt level of the FMPEG. So these pins can be wired to a fixed level in a PCB design.

A reset has always to be given to the FMPEG after powering on for a minimum time of 1µsec.

For more detailed information about the ISA Bus Interface refer to chapter 6.1. on page 32. The pinning of this interface is described in chapter 5.1..

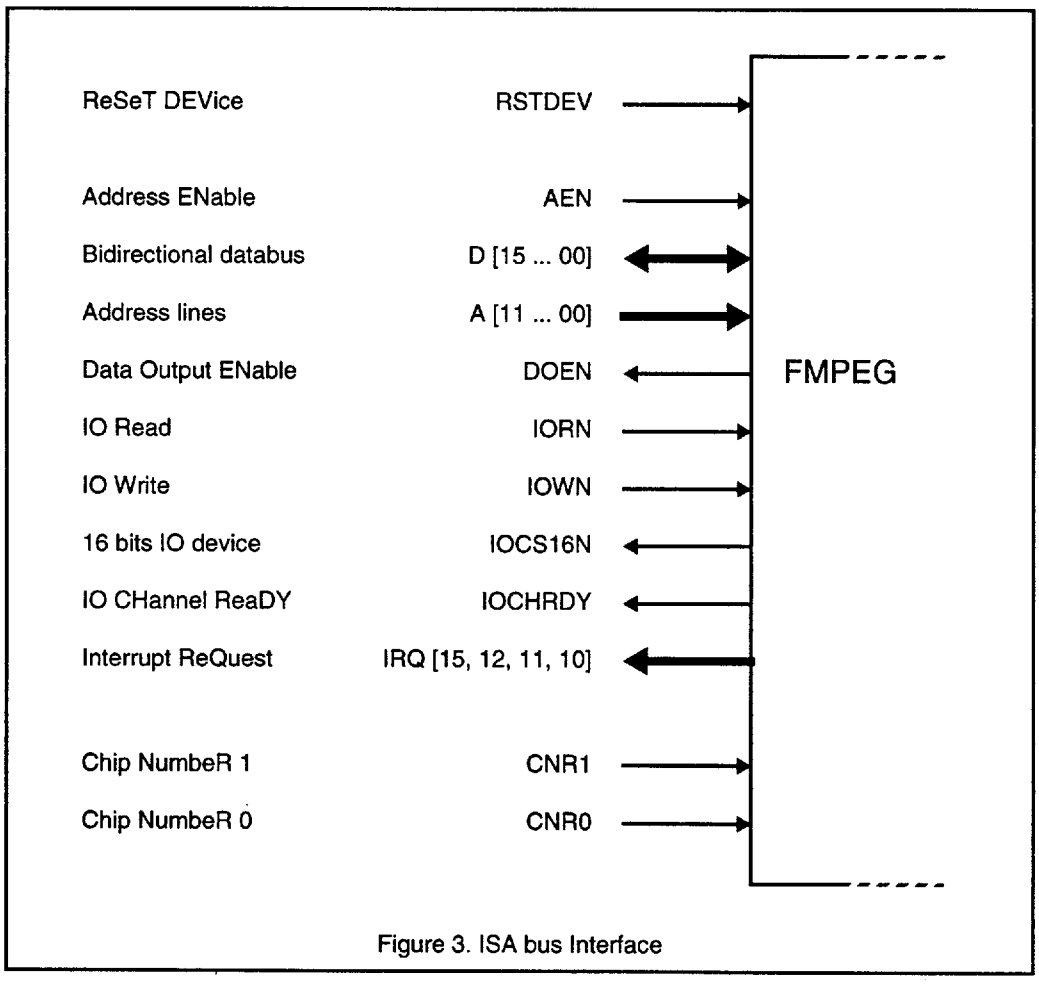


Figure 3. ISA bus Interface

3.2. DRAM Interface

Via the DRAM Interface the following data transfers between the FMPEG and the 4 Mbit external DRAM are performed:

- Writing the coded video and audio data which are output by the Video Input Parser and Audio Input Parser to the video FIFO and audio FIFO.
- Reading coded video and audio data from the video FIFO and audio FIFO for decoding.
- Writing reconstructed video data to the selected video buffer.
- Reading reconstructed video data from one or two of the video buffers depending on the reconstruction of P-pictures (unidirectional predicted) and B-pictures (bidirectional predicted).
- Reading the reconstructed video data from the video buffer for displaying.

The interface consists of 16 bidirectional data lines DR [15 ... 00], 9 address lines AR [8 ... 0] for multiplexed row/column addresses, two address select signals CASN and RASN and the write enable signal WEN. All DRAM accesses and necessary address calculations are handeled by the Memory Management Unit (MMU) of the FMPEG.

A detailed description of usage of the DRAM is given in chapter 6.2. on page 36. The pinning of this interface is described in chapter 5.1..

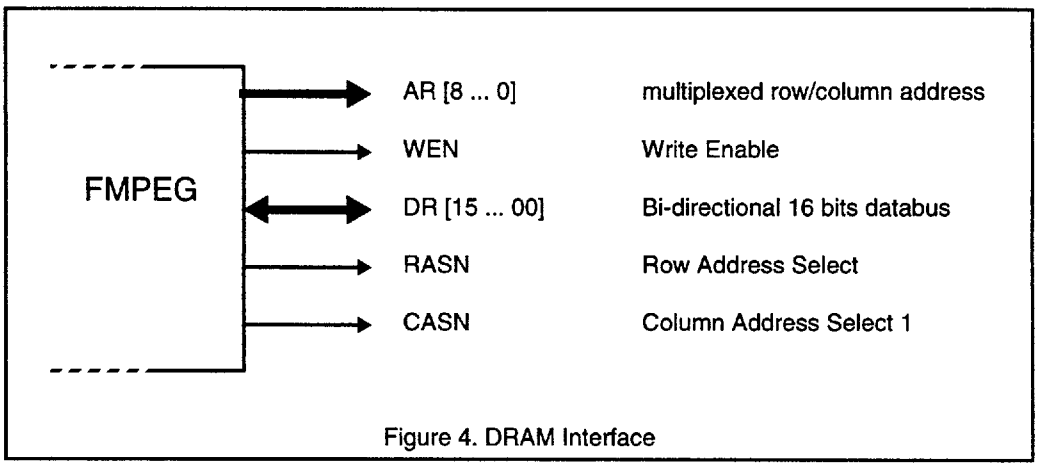


Figure 4. DRAM Interface

The FMPEG requires 4 Mbit of DRAM which is organized as 256 K * 16 Bit. DRAM's with 70 ns access time must be used.

Possible type numbers are:

- Hitachi HM 514260-7
- Samsung KM 416C256-7
- Micron MT 4C16256-7
- NEC UPD 4241600-7
- Toshiba TC 514260BZ-70
- Mitsumi M5M 44170AL-7
- Texas TMS 45160-70DZ

3.3. Video bus

Via the video bus the reconstructed video data is output in different video output formats using 8, 16 or all 24 video data lines VD [23 ... 00]. The video data is output parallel or in time multiplex in oder to support the output formats of digital multi standard decoder (DMSD; SAA7151B, SAA7191B, SAA7110 (OCF)) or different VESA Media Cannel (VMC) output formats. All possible RGB and luminace/chrominance output formats are listed in table 15, 17 and 16 on page 42 to page 43.

The video output can be synchronized either to the DMSD specific signals VS, HREF, CREF and LLC (TV-display mode) or to the VGA specific signals VSYNC, HSYNC and VIDEOCLK (VGA-display mode). Sychronization to the DMSD specific signals allows the FMPEG to be genlocked to a second video source, e.g. an external video signal which is decoded by an One Chip Front end (OCF, SAA7110). Synchronization to the VGA specific signals allows the reconstucted MPEG video directly to be merged with standard VGA computer gaphics. For the clock inputs LLC and VIDEOCLK clock frequencies of 22 MHz to 32 MHz are allowed to be used.

The Pixel Qualifier signal (PXQ) can flag a prefefined area in the video output. The location and the size of this area is programmable.

The video output is described in detail in chapter 6.3.1. on page 39, including the video output formats and windowing functions. The pinning of this interface is described in chapter 5.1..

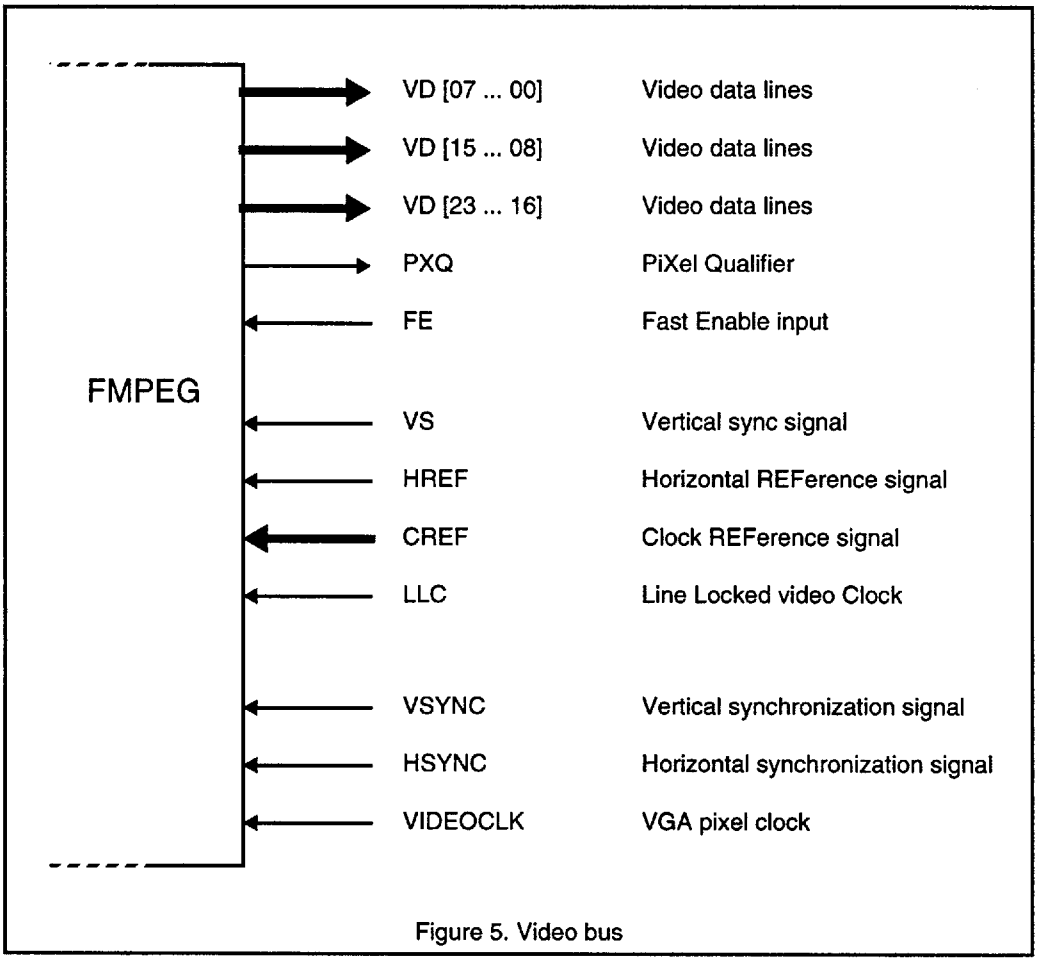


Figure 5. Video bus

3.4. Audio Interface

The Audio Interface of the FMPEG consits of a serial audio output, a serial audio input as well as three pins for providing the needed audio clock. The serial audio output uses the pins CLOUT, WSOUT ans DOUT and can be configured as an IIS bus or as an output LSB fixed format (Sony format). The serial oudio output is always the timing master.

To combine reconstructed MPEG audio or PCM audio data with external audio the external serial audio data has to be input via the DIN pin in IIS or LSB fixed format. The sample frequency of the external audio input must always be equal to the sample frequency of the decoded audio or the played back PCM audio respectively, but the serial audio data format (IIS or LSB fixed) can be different between audio input and audio output. The external audio input can be used either as timing master or as timing slave. When it is used as timing master the pins CLOUT and WSOUT of the audio output have to be chosen as the bit clock and word select. CLIN and WSIN have to be taken as the bit clock and word select if the external audio input is timing slave.

The audio clock of 256*Fs (Fs is audio sample frequency) has to be provided to the FMPEG is input via the AUCK pin. To this pin the internally generated audio clock which is output via the LLA pin can be connected or an external clock of the needed audio clock frequency can be applied. The internally generated audio clock is again derived from the line locked clock (LLC), from the VGA pixel clock (VIDEOCLK) or from an external clock source which is connected to the ASC pin.

Refer to chapter 6.4. on page 50 for more detailed information. The pinning of this interface is described in chapter 5.1..

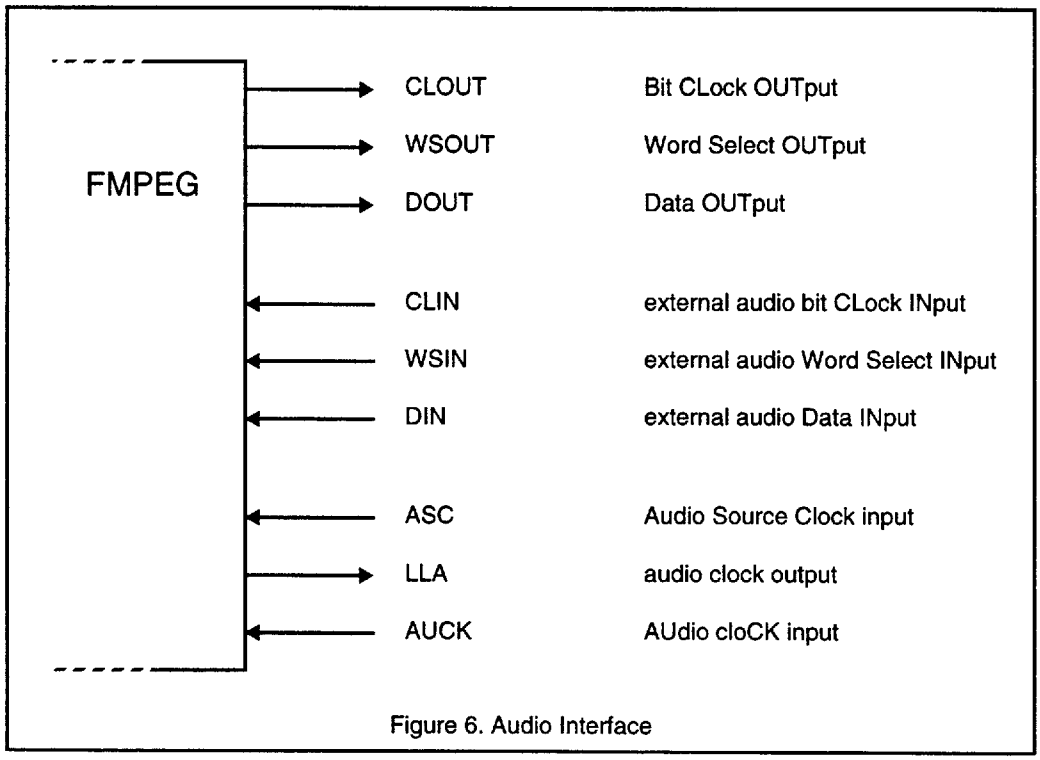


Figure 6. Audio Interface

4. OPERATING CONDITIONS

4.1. Electrical conditions

4.1.1. Absolut maximum ratings

SYMBOL	PARAMETER	MIN	MAX	UNIT
V_{DDD}	Supply voltage	-0.5	+7.0	V
V_I	Input voltage	-0.5	+7.0	V
V_O	Output voltage	-0.5	+7.0	V
V_{ESD}	Electrostatic handling for all pins	-	± 2000	V
P_{tot}	total power dissipation	-		W
T_{stg}	Storage temperature	-65	+150	°C
T_{amb}	Operating ambient temperature range	0	+70	°C

4.1.2. Electrical characteristics

SYMBOL	PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
Supply						
V_{DDD}	supply voltage range		4.5	5	5.5	V
I_{DDD}	total supply current		-	-		mA
Inputs						
V_{IL}	input voltage LOW		-0.5	-	0.8	V
V_{IH}	input voltage HIGH		2.0	-	V_{DDD}+0.5	V
I_{LI}	input leakage current		-	-		uA
C_I	input capacitance		-	-		pF
$C_{I.IO}$	input capacitance	I/O at high impedance	-	-		pF
Outputs						
V_{OL}	output voltage LOW		0	-		V
V_{OH}	output voltage HIGH			-	V_{DDD}	V

Table 1: Electrical characteristics

SYMBOL	PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
I_{OL}	output current LOW		-		-	mA
I_{OH}	output current HIGH		-		-	mA
I_{LO}	output leakage current		-	-		mA
C_{Ld}	output load capacitance		-		-	pF

Table 1: Electrical characteristics

4.2. Timing characteristics

4.2.1. ISA Bus Interface

SYMBOL	PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{CC}	crystal clock cycle time		-	25	-	ns
t_{CSD}	address valid to IOCS16 delay		-	-		ns
t_{CYD}	IOWN/IORN to IOCHRDY inactive delay		-	-		ns
t_{DED}	IOWN / IORN to DOEN delay			-		ns
t_{DESD}	DOEN delay during ISA Bus setup		-	-		ns

Table 2: ISA Bus timing

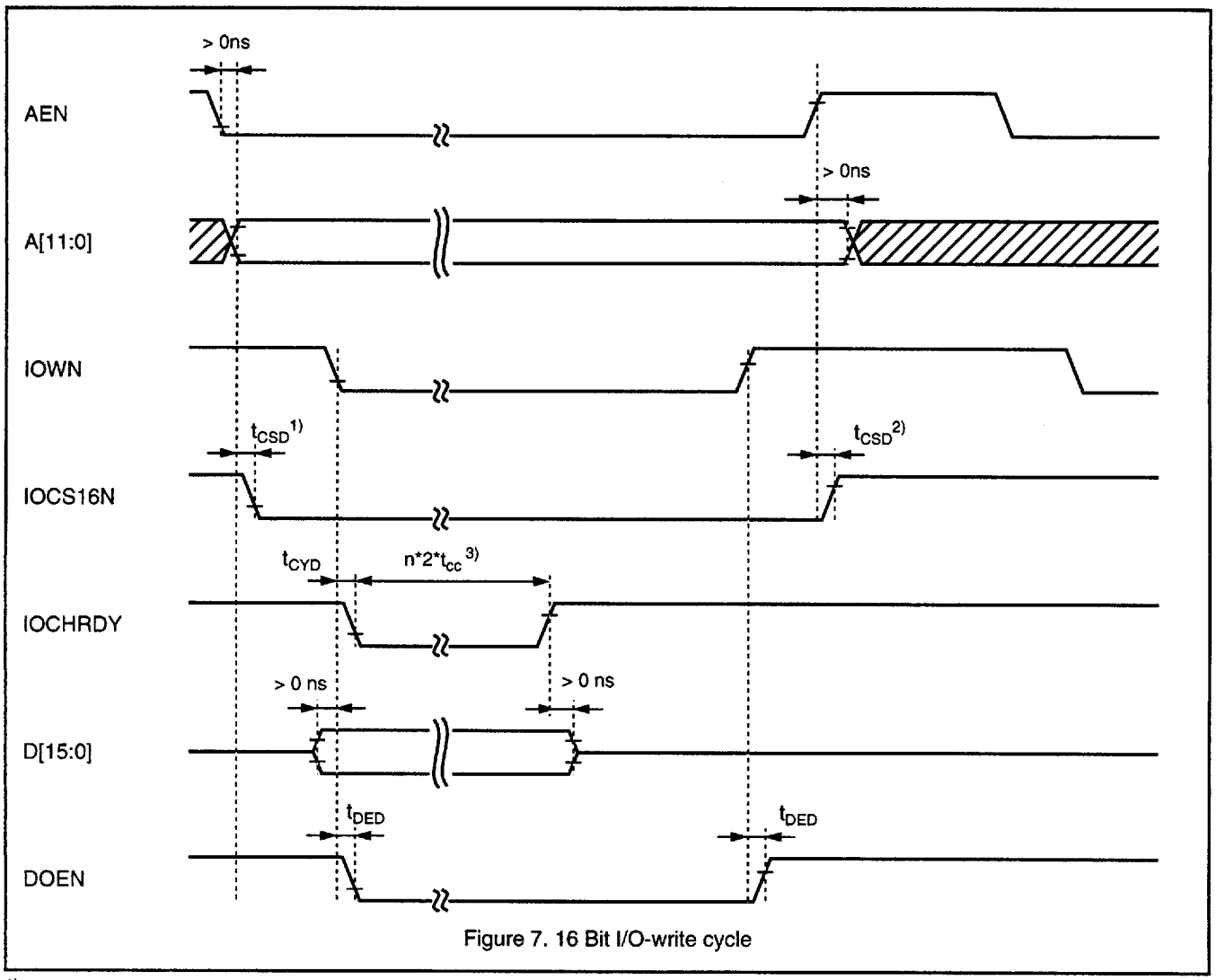


Figure 7. 16 Bit I/O-write cycle

1) The ISA Bus Interface setup must be finished and AEN must be low. If the address (A[11:0]) is valid before AEN is low the falling edge of the AEN signal becomes the reference for t_{CSD}.

2) The address (A[11:0]) must be valid. If the address (A[11:0]) is invalid before the rising edge of the AEN signal the address lines are becoming the reference for t_{CSH}.

3) n depends on the internal function which is accessed ($1 \leq n < m$)

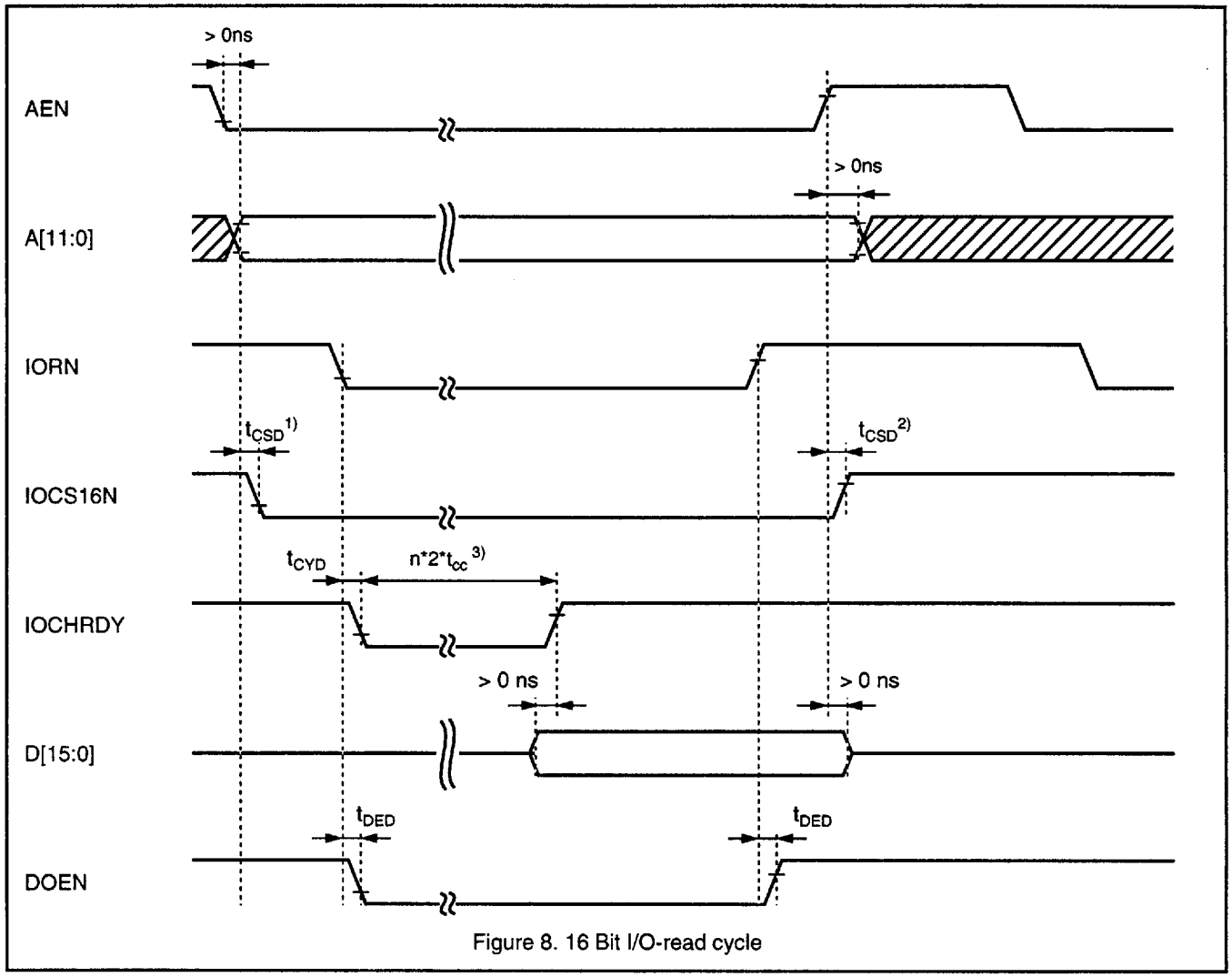


Figure 8. 16 Bit I/O-read cycle

1) The ISA Bus Interface setup must be finished and AEN must be low. If the address (A[11:0]) is valid before AEN is becoming active the falling edge of the AEN signal becomes the reference for t_{CSD}.

2) The address (A[11:0]) must be valid. If the address (A[11:0]) is invalid before the rising edge of the AEN signal the address lines are becoming the reference for t_{CSH}.

3) n depends on the internal function which is accessed ($1 \leq n < m$)

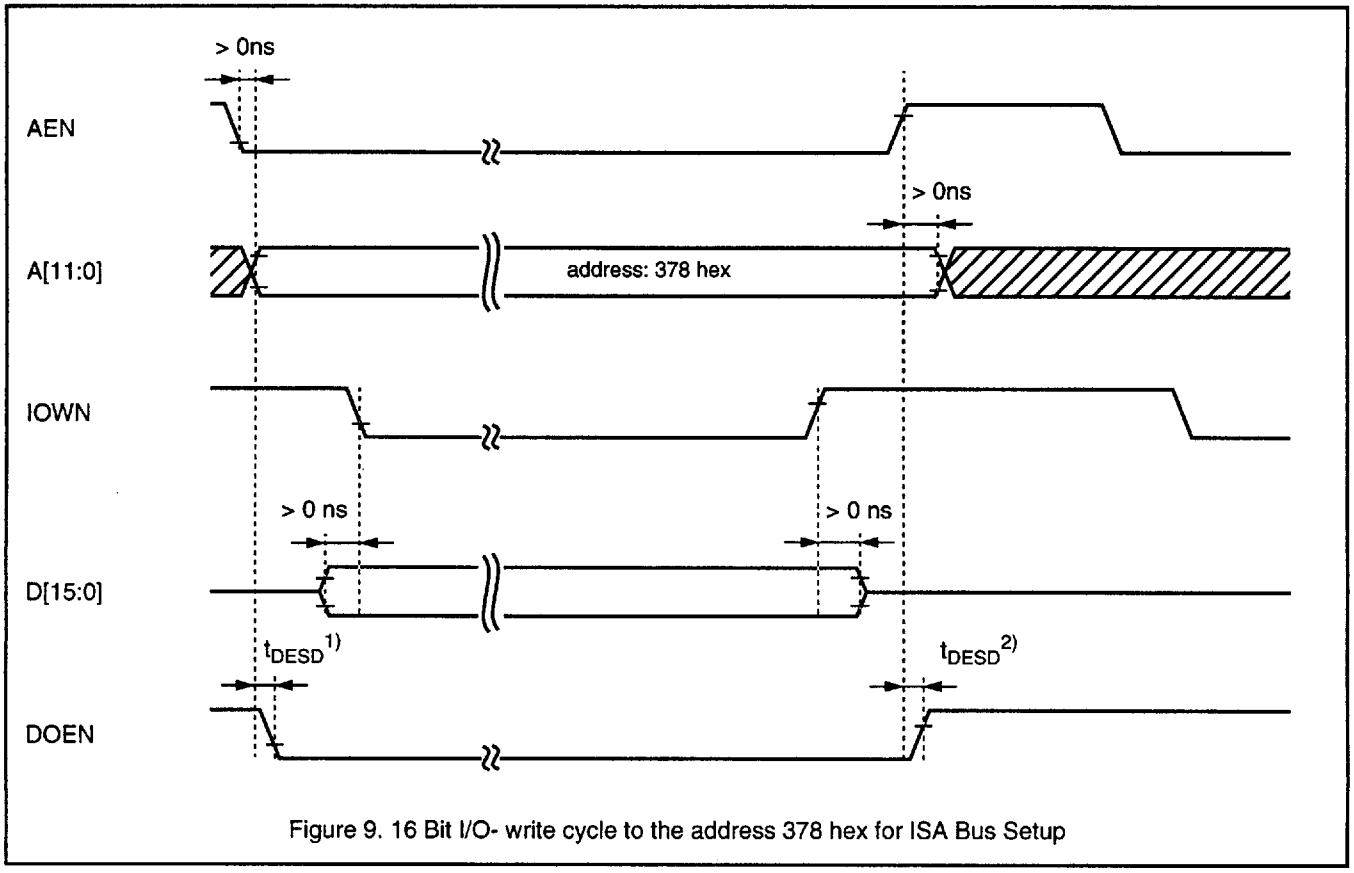


Figure 9. 16 Bit I/O- write cycle to the address 378 hex for ISA Bus Setup

1) During ISA Bus Interface setup: AEN must be active. If the address (A[11:0]) is valid before AEN is becoming active the falling edge of the AEN signal becomes the reference for t_{DESD}.

2) During ISA Bus Interface setup: The address (A[11:0]) must be valid. If the address (A[11:0]) is invalid before the rising edge of the AEN signal the address lines are becoming the reference for t_{DESH}.

4.2.2. DRAM Interface

SYMBOL	PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{CC}	crystal clock cycle time			25		ns
t_{RSH}	RASN hold time		-	1	-	ns
t_{WCH}	Write command hold time				-	ns
t_{ADSR}	Row address setup time			-	-	ns
t_{ADH}	Address / write Data hold time			-	-	ns
t_{DRS}	read Data setup time			-	-	ns
t_{DRH}	read Data hold time			-	-	ns
t_{CSR}	CASN setup time (refresh cycle)			-		ns

Table 3: DRAM timing

1 Typical value for t_{RSH} which can not be guaranteed.

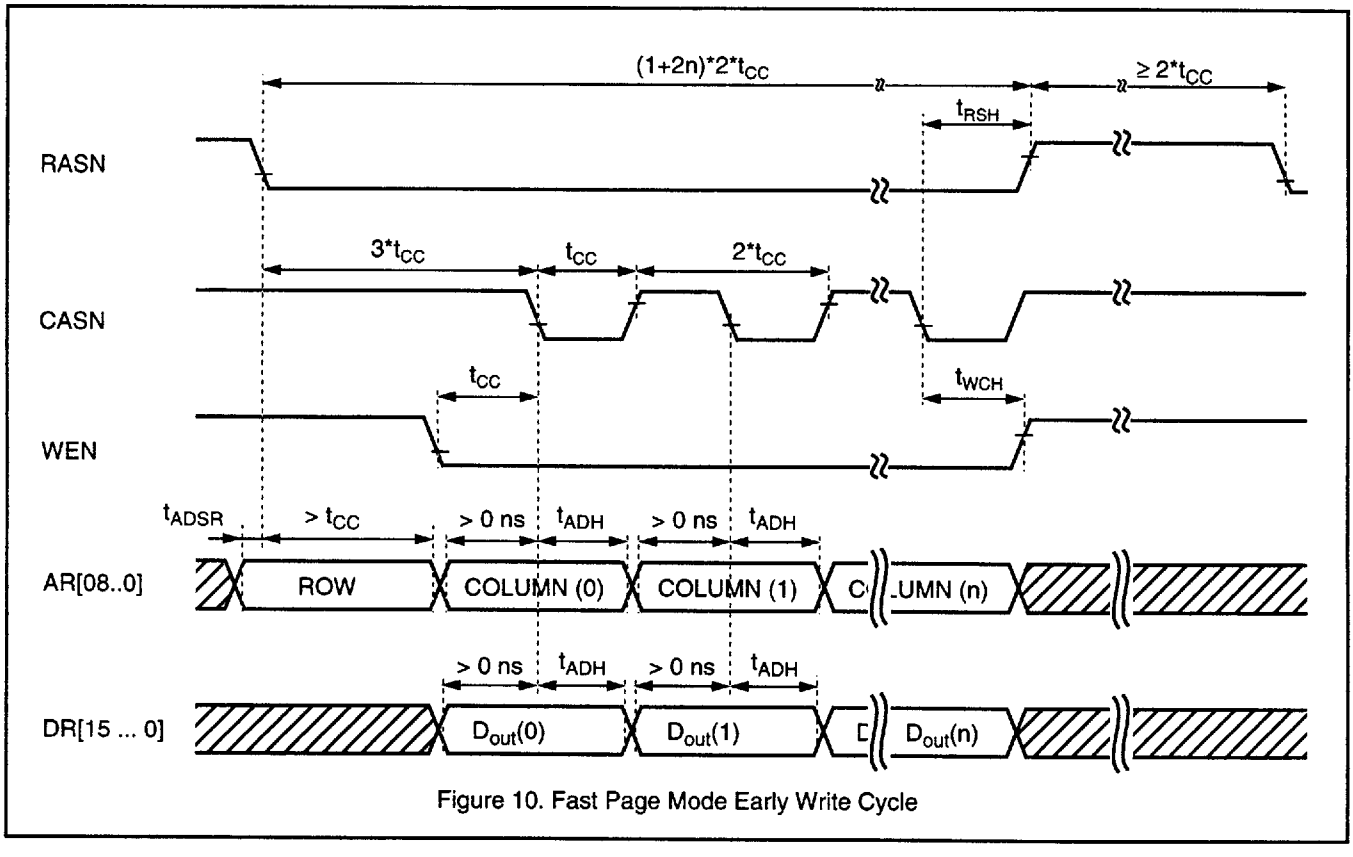


Figure 10. Fast Page Mode Early Write Cycle

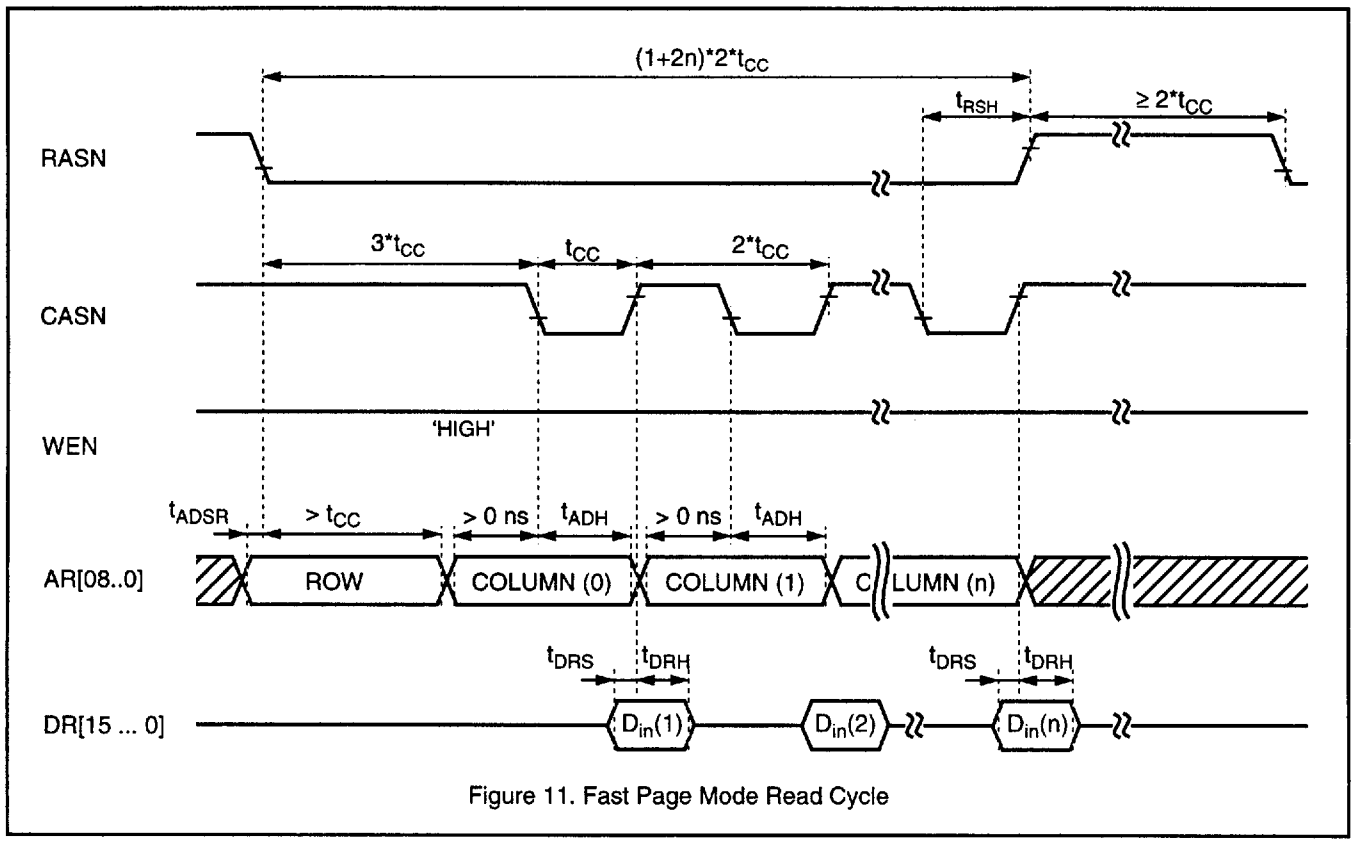


Figure 11. Fast Page Mode Read Cycle

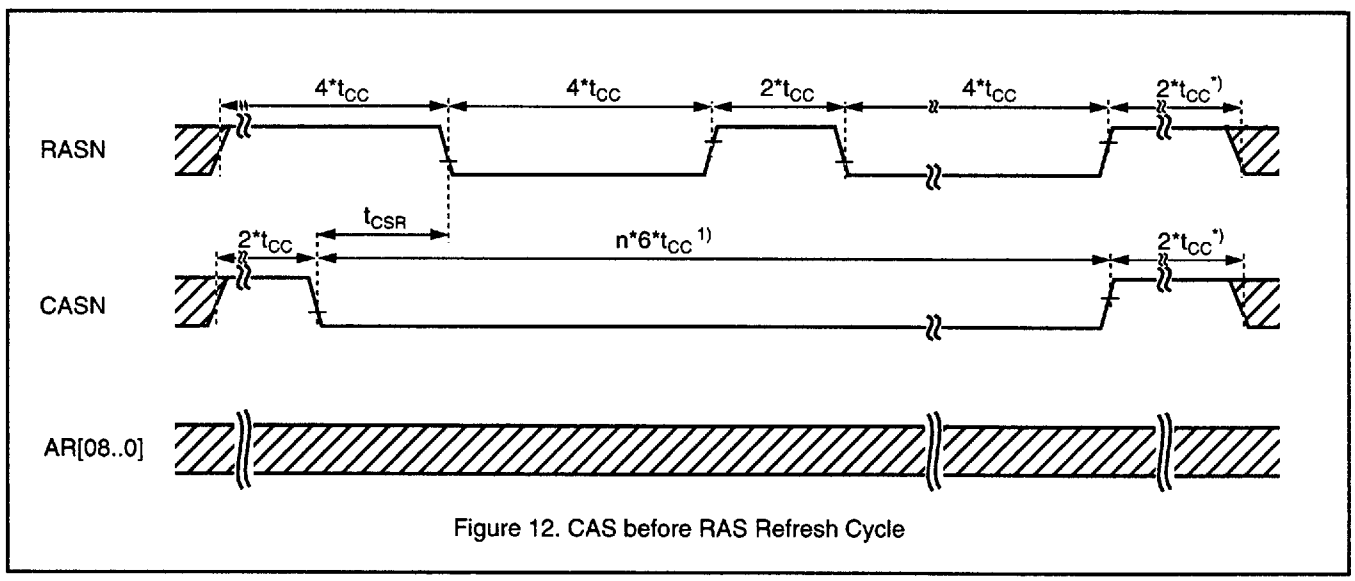


Figure 12. CAS before RAS Refresh Cycle

4.2.3. Video bus

SYMBOL	PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
Clock						
t_{LLC}	DMSD clock cycle time		31,3	-	45,4	ns
t_{VCLK}	VGA clock cycle time		31,3	-	45,4	ns
DMSD specific						
t_{CRS}	CREF setup time			-	-	ns
t_{CRH}	CREF hold time			-	-	ns
t_{HRS}	HREF setup time			-	-	ns
t_{HRH}	HREF hold time			-	-	ns
t_{FES}	FE setup time			-	-	ns
t_{FEH}	FE hold time			-	-	ns
t_{CPD}	CREF to Pixelqualifier delay		-	-		ns
VGA specific						
t_{HSS}	HSYNC setup time			-	-	ns
t_{HSH}	HSYNC hold time			-	-	ns
t_{VSS}	VSYNC setup time			-	-	ns
t_{VSH}	VSYNC hold time			-	-	ns
t_{PXD}	Pixelqualifier delay		-	-		ns
t_{PXH}	Pixelqualifier hold time			-	-	ns
Video data						
t_{VD}	Video data delay		-	-		ns
t_{VH}	Video data hold time			-	-	ns
t_{VTT}	Video data to tristate delay		-	-		ns
t_{VFT}	Video data from tristate setup time			-	-	ns

Table 4: Video bus timing

DMSD specific timing:

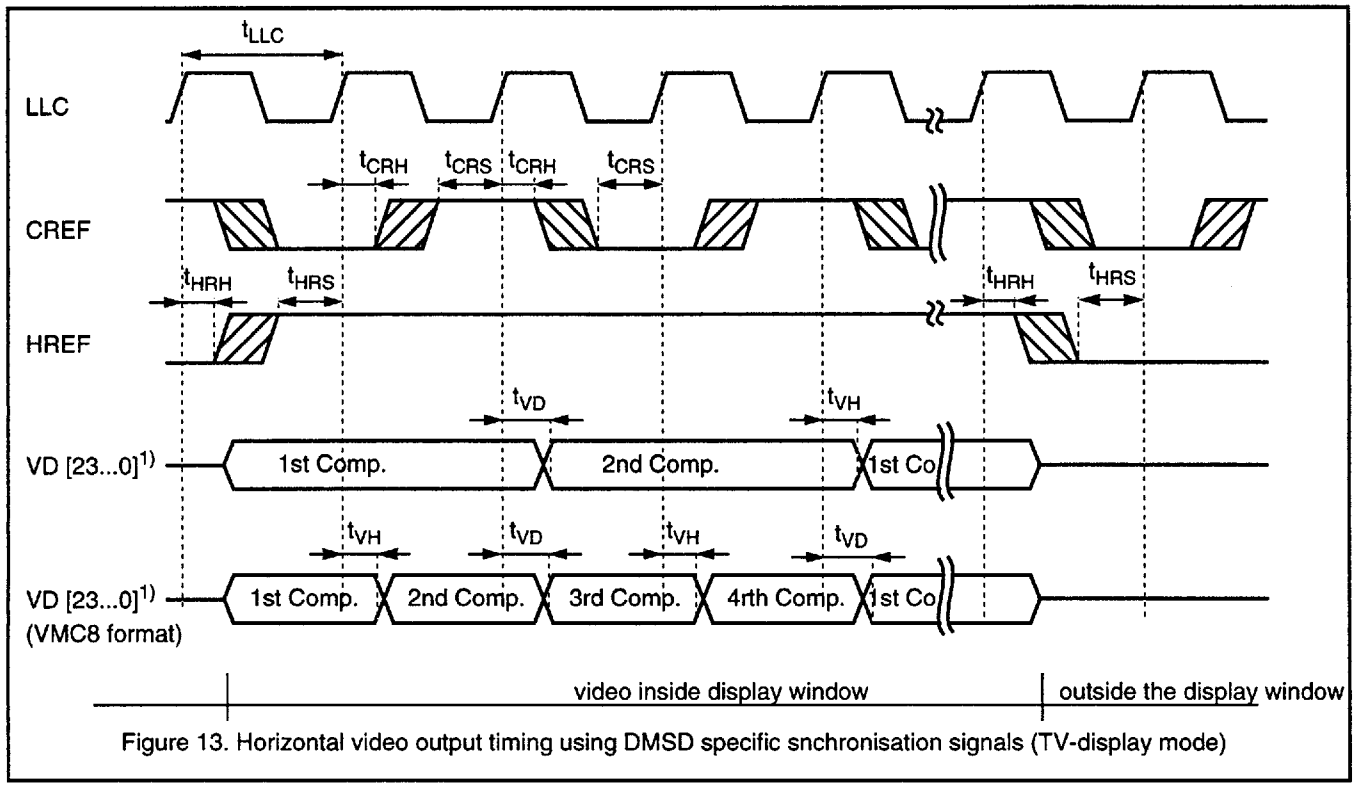


Figure 13. Horizontal video output timing using DMSD specific snchronisation signals (TV-display mode)

1) Outside the display window the video data lines VD[23 ...0} are either tristated or the video data output is blanked (Y = 16, U = 128, V = 128 or R=G=B = 16).

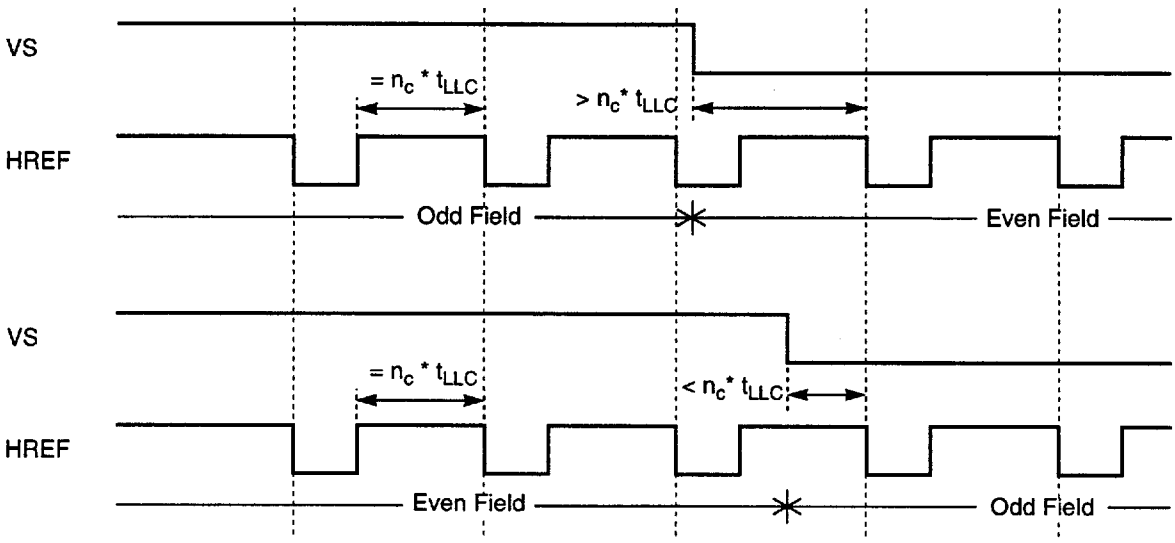


Figure 14. Vertical video data output timing, even and odd field detection

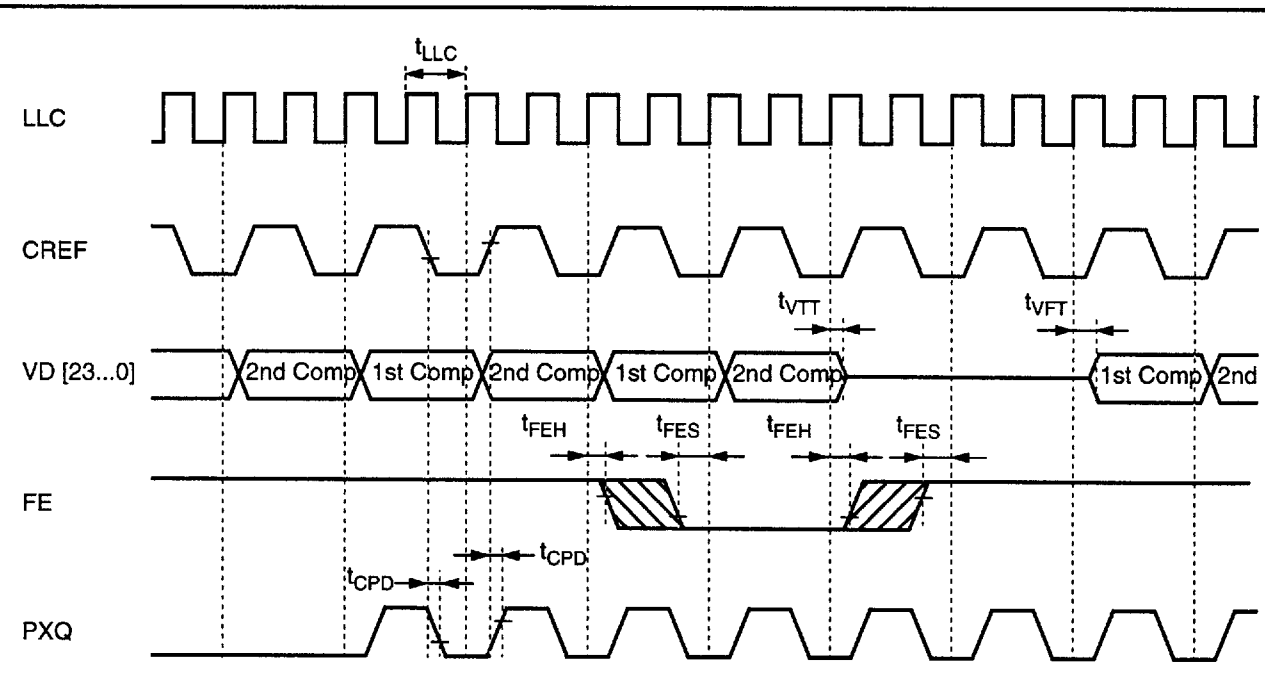


Figure 15. FE and PXQ timing using DMSD specific synchronisation signals (TV-display mode) (1b)

VGA specific timing

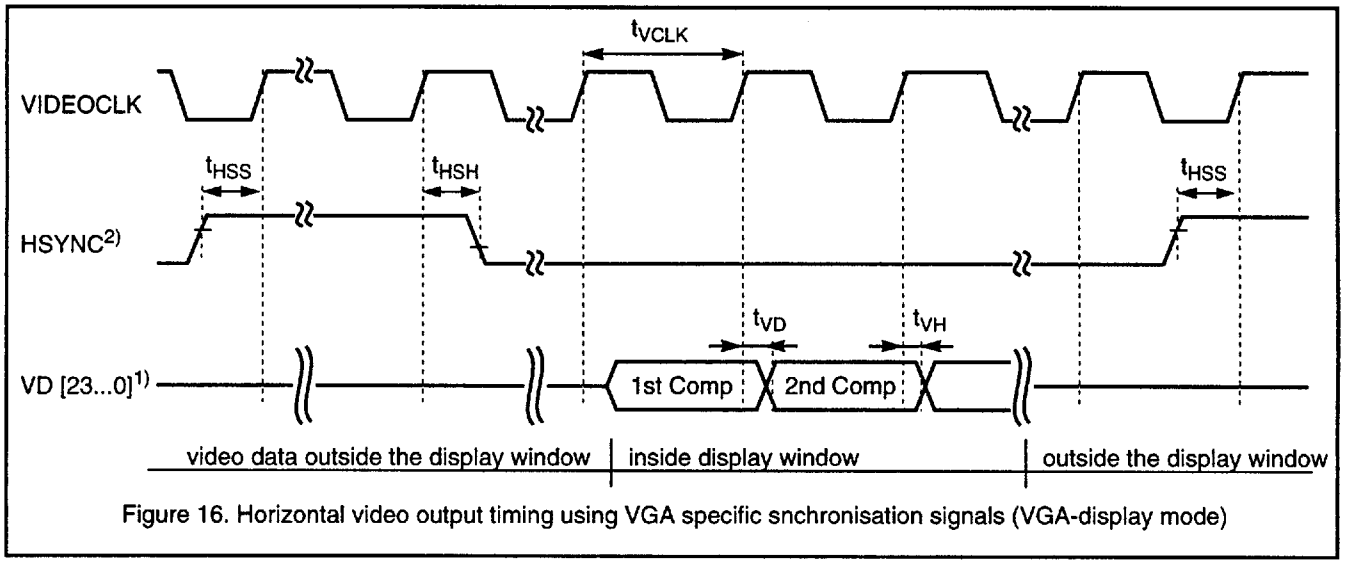


Figure 16. Horizontal video output timing using VGA specific snchronisation signals (VGA-display mode)

1) In case of using VMC8 video output formats four components are output successively.
Outside the display window the video data lines VD[23 ...0} are either tristated or the video data output is blanked (Y = 16, U = 128, V = 128 or R=G=B = 16)

2) HSYNC might be high active or low active.

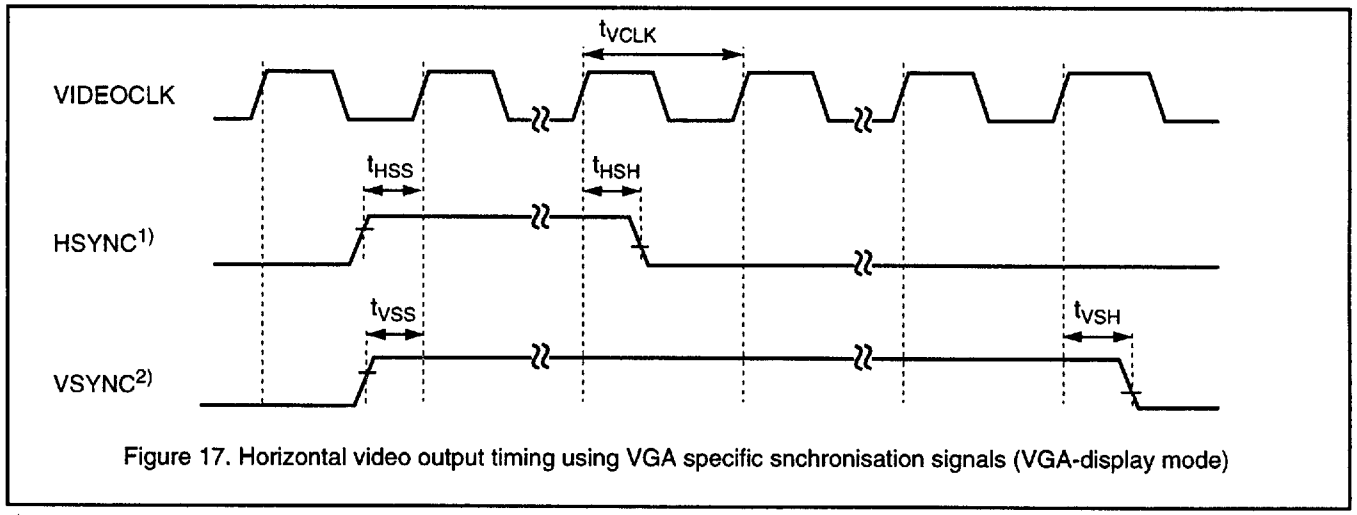


Figure 17. Horizontal video output timing using VGA specific snchronisation signals (VGA-display mode)

1) HSYNC might be high active or low active.

2) VSYNC might be high active or low active.

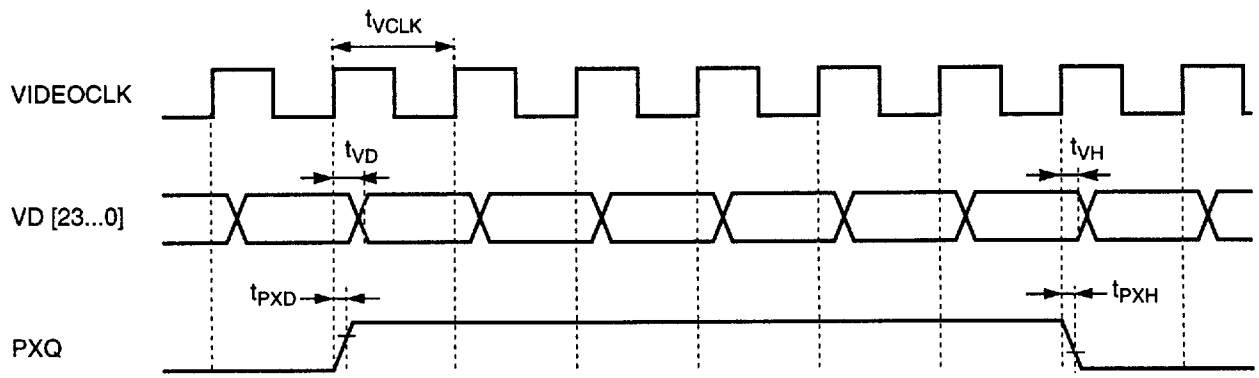


Figure 18. PXQ timing using VGA specific synchronisation signals (TV-display mode) (1b)

4.2.4. Audio Interface

SYMBOL	PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{BCK}	Bitclock		81	-	122	ns
t_{BCKH}	Bitclock high time			-		ns
t_{BCKL}	Bitclock low time			-		ns
t_{PWo}	Word Select out delay		-	-		ns
t_{HWo}	Word Select out hold time			-	-	ns
t_{PDo}	Audio data out delay		-	-		ns
t_{HDo}	Audio data out hold time			-	-	ns
t_{SWi}	Word Select in setup time			-	-	ns
t_{HWi}	Word Select in hold time			-	-	ns
t_{SDi}	Audio data in setup time			-	-	ns
t_{HDi}	Audio data in hold time			-	-	ns

Table 5: Audio Interface timing

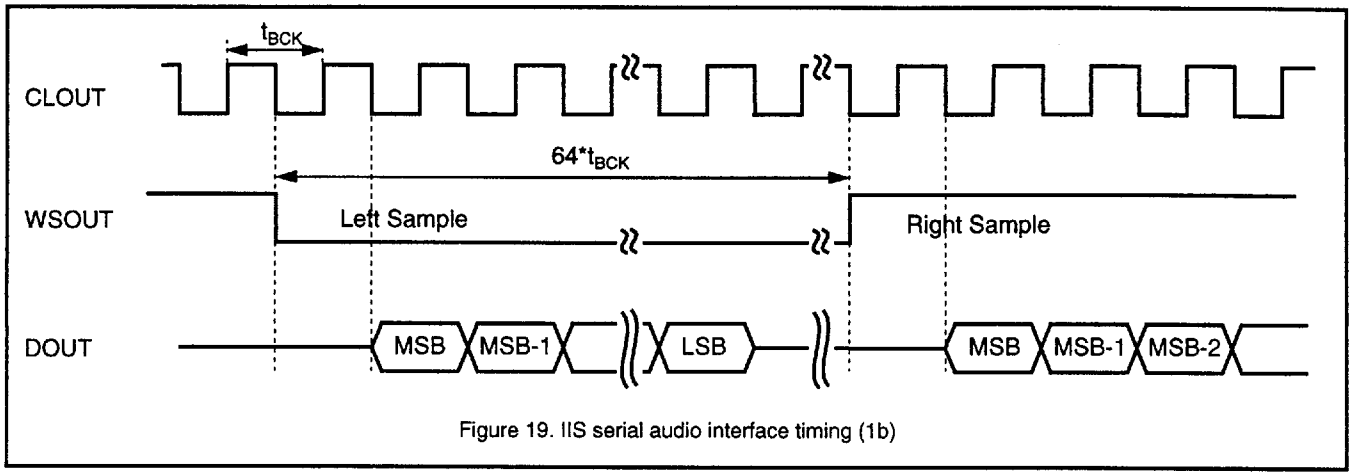


Figure 19. IIS serial audio interface timing (1b)

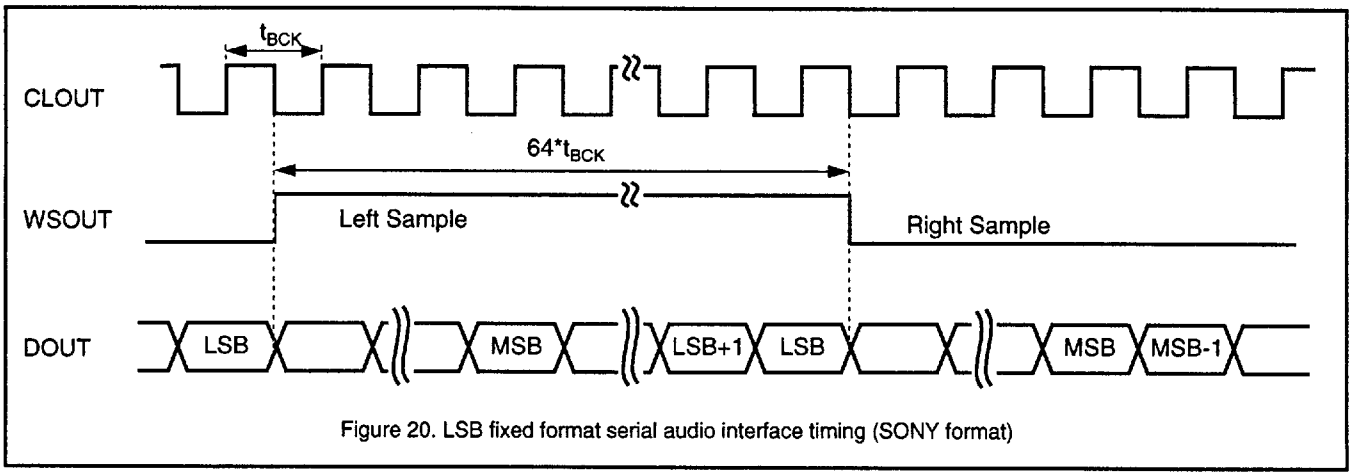


Figure 20. LSB fixed format serial audio interface timing (SONY format)

Audio output

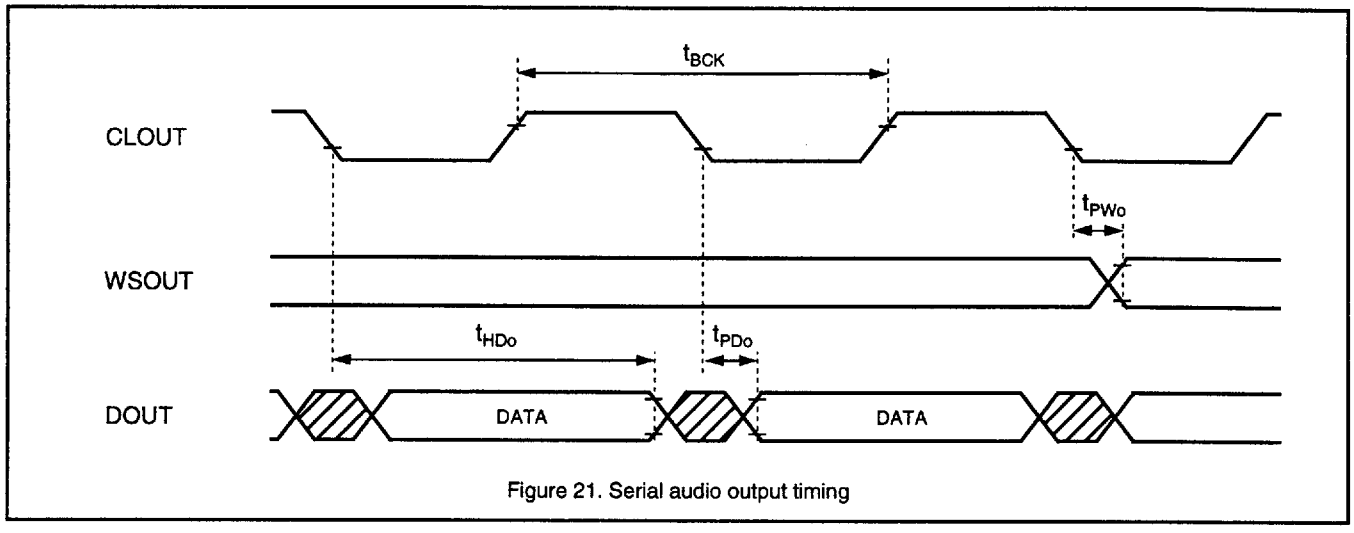


Figure 21. Serial audio output timing

Note: An audio clock with 50 % duty cycle is required at the audio clcok input pin (AUCK, pin 160) to obtain a bit clock with 50 % duty cycle.

Audio input

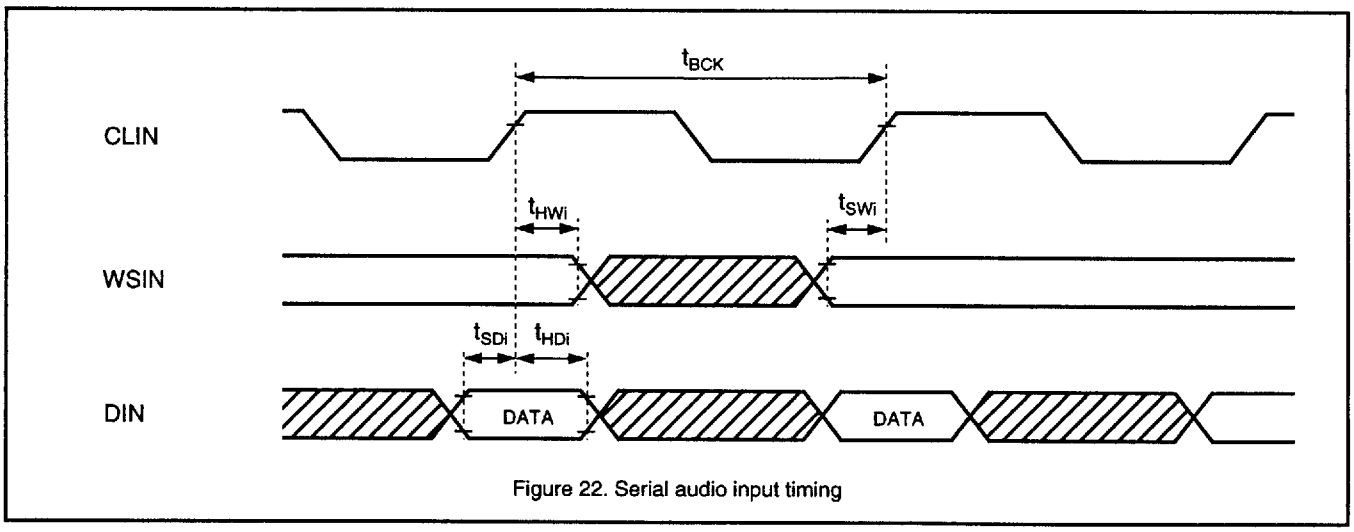


Figure 22. Serial audio input timing

Note: When the audio input interface is used in the slave mode the bitclock CLIN and the wordselect WSIN are obtained from the external audio source. To use the audio input interface as a timing master the the bitclock input CLIN and the wordselect input WSIN must be connected to the bitclock and wordselect (CLOCU and WSOUT) of the audio output interface.

5. PACKAGE

5.1. Pin description

Signal	Pin No.	Status	Signal description
ISA BUS Interface			
RSTDEV	81	I	ReSeT DEVice. Active HIGH reset line. Minimum HIGH time 1µsec.
D[15..00]	138 - 141 144 - 149 152 - 157	I, O	Bidirectional databus.
A[11..00]	103 - 95 92 - 90	I	Address lines For IO port addressing only A[09..00] are valid. A[11,10] are for future use to support "plug and play".
AEN	106	I	Address ENable Active HIGH input signal asserted during DMA. When active the FMPEG IC will not evaluate the address.
IORN	108	I	IO Read cycle Active LOW input requesting for data from the addressed IO device.
DOEN	107	O	Data Output ENable Active LOW output signal used, during an IO-read cycle, to enable the external data line buffers.
IOWN	109	I	IO Write cycle Active LOW input indicating that data is valid on the bus for the addressed IO device.
IOCS16N	110	O	16 bits IO device Active LOW output that indicates, during an IO cycle, that the addressed device is an 16 bits IO device.
IOCHRDY	111	O	IO CHannel ReaDY Active HIGH output signal. Driven LOW during an IO cycle when the bus cycle must be lenghtened.
IRQ[15,12,11,10]	114 - 117	O	Interrupt ReQuest Active HIGH output signal that indicates the occurence of certain events.
CNR0	119	I	Chip NumbeR inputs These two bits are part of the FMPEG identifation code that is used for software configuring the IO-port addresses and interrupt level. (see chapter 6.1.1. on page 32)
CNR1	118	I	

Table 6: FMPEG Pinning

Signal	Pin No.	Status	Signal description
The DRAM interface			
AR[08..00]	53, 54 57 - 62 65	O	9 bits multiplexed row/column address
WEN	67	O	Write Enable Active LOW write strobe.
DR[15..00]	41 - 46 49, 50 68 - 70 73 - 77	I, O	Bi-directional 16 bits databus
CASN	51	O	Column Address Select 1 Active LOW column address strobe for the data (DR[15..00]).
RASN	66	O	Row Address Select Active LOW row address strobe.
Video bus: General signals			
VD[23..00]	10, 9 6 - 1 20 - 17 14 - 11 30 - 25 22, 21	O	Video data output lines [23..00] For the mapping of the video signals to the output pins see table 15 on page 42, table 16 on page 43 and table 17 on page 43
PXQ	122	O	PiXel Qualifier An active HIGH signal that flags a predefined area on the screen. The size and the location of this predefined area is programmable.
FE	123	I	Fast Enable input Active HIGH signal. When inactive the video data outputs are made tri-state.
Video bus: DMSD specific signals			
VS	36	I	Vertical Sync Active HIGH. Active during the vertical retrace. When active the video output is blanked.
HREF	34	I	Horizontal REFerence signal Acive LOW. Active during the horizontal retrace. When active the video output is blanked.
CREF	35	I	Clock REFerence signal Indicates which rising edge of the LLC must be used to change the output pixels.
LLC	33	I	Line Locked video Clock Video timing clock of 2 times the pixel clock.

Table 6: FMPEG Pinning

Signal	Pin No.	Status	Signal description
Video bus: VGA specific signals			
HSYNC	125	I	Horizontal synchronization signal The HSYNC signal has a programmable polarity.
VSYNC	126	I	Vertical synchronization signal The VSYNC signal has a programmable polarity.
VIDEOCLK	129	I	VGA pixel clock
Audio interface: Audio output signals			
CLOUT	135	O	Bit CLock OUTput
WSOUT	136	O	Word Select OUTput
DOUT	137	O	audio Data OUTput
Audio interface: external Audio input signals			
CLIN	131	I	external audio bit CLock INput
WSIN	130	I	external audio Word Select INput
DIN	132	I	external audio Data INput
Clock pins			
SYSCLK0	88	I	System clock oscillator input pins A crystal of 40 MHz must be connected.
SYSCLK1	89	O	
AUCK	160	I	AUdio cloCK input Clock input for the 8.192 MHz, 11.2896 MHz or 12.288 MHz clock. In case of using the internally generated audio clock, which is generated by the audio DTO, this pin is connected to the LLA pin (pin 37).
NCLU	80	O	90 KHz MPEG system clock output
LLA	37	O	audio clock output Digital output of the audio DTO which contains the internally generated audio clock. This pin provides the audio clock, which can be applied to the AUCK pin (pin160).
ASC	38	I	Audio Source Clock input An external clock signal applied to this pin can be used as the input for the audio DTO, which generates the audio clock.

Table 6: FMPEG Pinning

Signal	Pin No.	Status	Signal description
Test pins			
TEST_MOD	85	I	TEST_MODe (must be connected to ground in normal mode)
TEST_SHT	86	I	TEST SHifT (must be connected to ground in normal mode)
TEST_EN	87	I	TEST ENable (must be connected to ground in normal mode)
TST0	82	I	Test mode pin 0 (must be connected to ground in normal mode)
TST1	83	I	Test mode pin 1 (must be connected to ground in normal mode)
TST2	39	I	Test pin (must be connected to ground in normal mode)
SCW	84	I	Test pin (must be connected to ground in normal mode)
Voltage supply			
VDDD	7, 15, 23 31, 47, 55 63, 71, 78 93, 104 112, 120 127, 133 142, 150 158	I	+ 5V supply voltage
VSSD	8, 16, 24 32, 48, 56 64, 72, 79 94, 105 113, 121 124, 128 134, 143 151, 159	I	GND
Reserved			
Reserved	40, 52	-	open

Table 6: FMPEG Pinning

5.2. Package layout

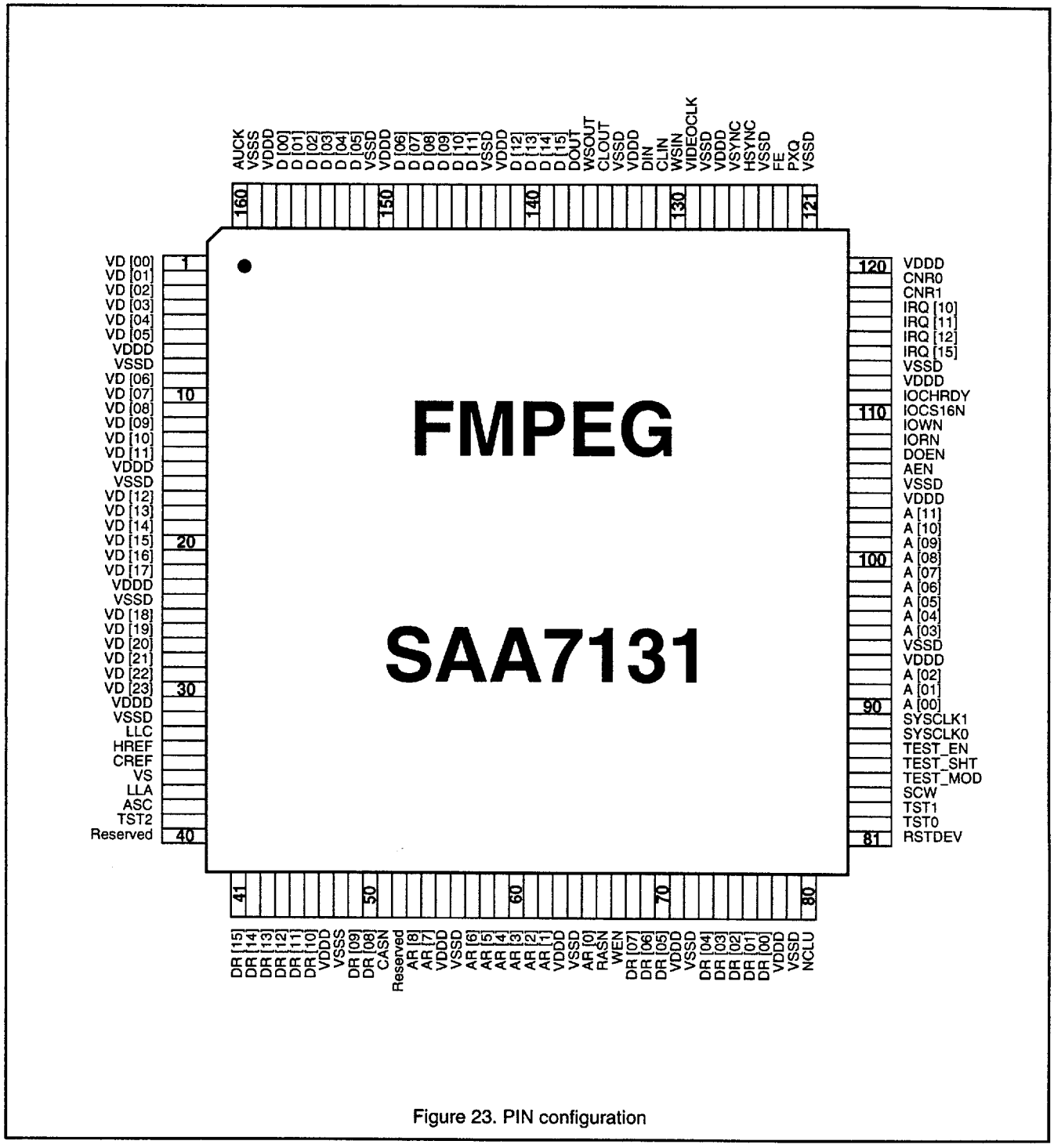


Figure 23. PIN configuration

6. SYSTEM BLOCK DESCRIPTION

This chapter describes hardware registers and functions of the FMPEG more in detail. All registers which are listed in this chapter are hardware registers and can be accessed directly via the ISA Bus. All other contents written or read on the other addresses of the internal address area Bxxx hex which is used for controlling the device are evaluated by the microcode which must be downloaded from the Host via the ISA Bus Interface. Since the access (IO addresses and data) of to these functions are dependend of the microcode all other functions are described in principle. Crossreferences to the sofware driver functions (chapter 7.1.1. on page 63 are given).

Registers, which are mentioned in the text, have following syntax: REGISTER_NAME [REGISTER_BIT_NAME, ...]. If no REGISTER_BIT_NAME is given the register with all its contents is addressed.

6.1. ISA Bus interface

Controlling the FMPEG, transfering the compressed data stream to the FMPEG, reading back reconstucted video and audio data and playing back PCM audio data is done via the ISA bus interface. Therefore IO-read, IO-write and interrupts are supported.

Only two IO ports are needed. With one IO access (write) to the first IO port (internal address port) the internal address to access to registers an internal RAM is selected. Via the second port (data port, read/write) the actual data transfer is done. In the following these ports two IO ports will be called internal address port and data port.

For interrupts generated by the FMPEG the following four interrupt levels are possible: IRQ_10, IRQ_11, IRQ_12 and IRQ_15. The interrupt level is selected during ISA Bus Interface setup.

6.1.1. ISA Bus Interface setup

The port addresses and interrupt level of the FMPEG are software programmable. Before the FMPEG can be accessed as an IO device the port addresses and the Interrupt level have to be determined. This is done by writing a predefined sequence to the centronics port: ADDRESS :HEX'378'.

The sequence consists of a 5 byte header:

HEX'73'	HEX'50'	HEX'41'	HEX'53'	HEX'45'	CNR	DSEL	DH	DL

The next byte sent contains the chip-number (CNR). The 6 MSB's of the chip-number are hardwired in the FMPEG IC. The two LSB's are copied from the CNR[1:0] input pins. The basic FMPEG IC number is : HEX'08'. The next byte indicates the item to be programmed (DSEL):

DSEL = 0 : IO-port 1. (Address port).
DSEL = 1 : IO-port 2. (Data port).
DSEL = 2 : Interrupt level.
DSEL = 3 : Complete setup

The fourth sequence with DSEL = 3 has to be sent to complete the setup of both IO-ports and the Interrupt level.

The last two bytes contain the data (DH + DL), most significant byte first. When the interrupt level is programmed (DSEL=3) The 4 LSB's of the least significant byte (DL) indicate the interrupt level (Refer to: "FmpegDevice::InitBusInterface" on page 66):

DL, Bit 0 : Interrupt 10.
DL, Bit 1 : Interrupt 11.
DL, Bit 2 : Interrupt 12.
DL, Bit 3 : Interrupt 15.

An example of a programming cycle is:
Port 1 : address: HEX'25A'
Port 2 : address: HEX'1F2'
Interrupt level : 12.

PORT ADDRESS : HEX'278' :

	HC1	HC2	HC3	HC4	HC5	CNR	DSEL	DH	DL
Sequence 1 :	73	50	41	53	45	08+USER[1:0]	00	02	5A
Sequence 2 :	73	50	41	53	45	08+USER[1:0]	01	01	F2
Sequence 3 :	73	50	41	53	45	08+USER[1:0]	02	00	04
Sequence 4 :	73	50	41	53	45	08+USER[1:0]	03		

For each sequence applies that all bytes must be send consecutively.

The address port is write only. The data port is bidirectional. It is assumed that external buffers are used for buffering the data lines. External tristateable buffers are required for connection of the IOCS16 and IOCHRDY signal lines to the bus (Application Notes, chapter 7.1., "FmpegDevice::InitBusInterface" on page 66).

Interrupts can be generated by the FMPEG IC at any time. The events on which interrupts are generated are evaluated internally by the microcode programm and then output via an IRQ line of the ISA bus interface. The IRQ signal remains asserted until the interrupt status is read.

6.1.2. Microcode download

The microcode which is needed for operating is downloaded to six internal address areas of the FMPEG after a reset. Thi task has to be performed after the ISA Bus Interface has been setup.

Host access: write address areas: 2xxx hex, 4xxx hex, 6xxx hex, 8xxx hex, Axxx hex and Cxxx hex

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Micro code data															

Bit	Function
Bit 15 ... 0	Micro code data This register must be regarded as a data port through which the micro code is downloaded. The download area is divided over 6 internal address areas.

Table 7: Micro code download (MCL)

The microcode should be downloaded by the FMPEG driver software (Application Notes, chapter 7.1., "FmpegDevice::LoadMicroCode" on page 68).

6.1.3. MPEG Data Input

A compressed MPEG data stream can be send to the FMPEG for video or audio decoding only or for video and audio decoding at the same time. This depends on the internal address area the MPEG data steam will be written to. Only video data is extracted from MPEG data strams which are written to Video data input (VDI) and only audio data is retieved from MPEG data streams which are send the audio data input (ADI). Both the video and audio data are retrived when MPEG data streams are written to the audio/video data input (AVI).

If the internal addresses of the audio, video or audio/video input is selected via the internal address port all subsequent write actions to the data port will be send to the VDI, ADI or AVI port respectively until a new internal address is selected.

The compressed audio- and video data should be transterred by the FMPEG driver software (Application Notes, chapter 7.1., "FmpegDevice::WriteMpegData" on page 84, strean data buffer functions chapter 7.1.1.3.).

Video data input (VDI)

Host access: write address area: 5xxx hex

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VDI	Video Data Input															

Bit	Function
Bit 15 ... 0	Video data input port Data written to this area will be send, after parsing, to the video FIFO, which is located in the external DRAM.

Table 8: Video data input (VDI)

Audio data input (ADI)

Host access: write address area: 7xxx hex

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADI	Audio Data Input															

Bit	Function
Bit 15 ... 0	Audio data input port Data written to this area will be send, after parsing, to the audio FIFO, which is located in the external DRAM.

Table 9: Audio data input (ADI)

Audio / Video data input (AVI)

Host access: write address area: 9xxx hex

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AVI	Audio / Video Data Input															

Bit	Function
Bit 15 ... 0	Audio / Video data input port Data written to this area will be send, after parsing in the video parser and the audio parser, to the video FIFO and audio FIFO respectively.

Table 10: Audio / Video data input (AVI)

6.1.4. Identification registers

Chip identification code (CIC)

Host access: read address: B089 hex (45193 dec)

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CIC	1	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1

Bit	Function
Bit 15 ... 0	Chip identification code: HEX'9411'

Table 11: Chip identification number register (CIC)

The identification code can be read by the FMPEG driver software (Application Notes, chapter 7.1., "FmpegDevice::ReadRegister" on page 67).

6.2. DRAM

6.2.1. DRAM organization

The DRAM which has to be applied to the FMPEG as working memory has the following functions:

- Storing the compressed video and audio data which have already being parsed in the video FIFO and audio FIFO respectively.
- Storing the reconstucted video frames in the video buffers.

When the FMPEG is decoding MPEG video in SIF resolution as well as MPEG audio the DRAM is organized like shown in Figure 24. The size of the video buffer 0, video buffer 1 and video buffer 2 is then fixed to 149 kbytes.

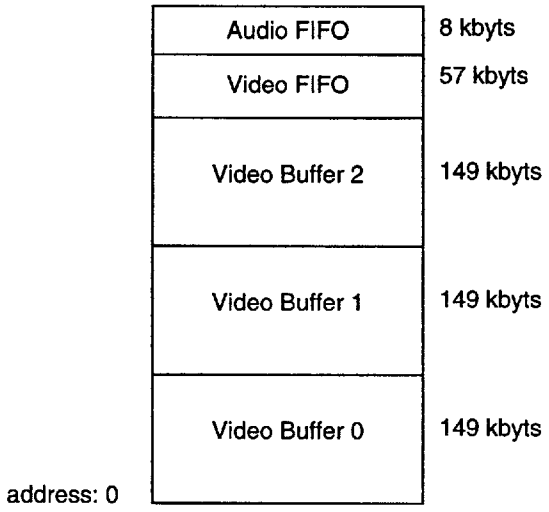


Figure 24. Diagram of DRAM organization

When the FMPEG is decoding MPEG video in SIF resolution the size of the video buffer 0, video buffer 1 and video buffer 2 is fixed to 149 kbytes and the memory management MMU determines which video buffer will be selected to store the next decoded picture.

In case of decoding still pictures with a higher resolution than MPEG SIF resolution the all three video buffers will be used as one buffer to store the reconstructed high resolution picture. This is done automaticly so that there are no additional settings necessary. On the other hand the user has to take care that enough free space is left in the DRAM for the reconstructed video data if high resolution still pictures are decoded. Otherwise the reconstructed video data will overwrite the contents of the FIFOs. If e.g. a still picture with the maximum resolution of 704 * 480 pixels is stored inside the external DRAM only 17kbytes will be left for the video and audio FIFOs. So the sizes of the video and audio FIFO must be canged to smaller sizes in this case.

To control the usage of the external DRAM, the FMPEG provides the following functions:

- Set video DRAM start page
  Dram page where the video FIFO starts.
- Set video block size as a multiple of 4 words
  Indicates the maximum number of words (in a multiple of 4 words) the HOST wants to write to the video FIFO. An interrupt will be generated when there is enough room in the FIFO to store such a block.

- Read video FIFO write pointer as a multiple of 4 words
  Indicates, where the next video data will be written in the video FIFO. Only the bits 17 to 2 of the 18 bits address are presented.
- Read video FIFO read pointer as a multiple of 4 words
  Indicates, where the next video data will be read from the video FIFO. Only the bits 17 to 2 of the 18 bits address are presented.
- Read video FIFO free size as a multiple of 4 words
  This value is updated by the system controller prior to each interrupt or when it is explicitly requested.
- Set audio DRAM start page
  Dram page where the audio FIFO starts. This value determines at the same time the upper bound of the video FIFO because the video FIFO is located directly below the audio FIFO.
- Set audio block size as a multiple of 4 words
  Indicates the maximum number of words (in a multiple of 4 words) the HOST wants to write to the audio FIFO. An interrupt will be generated when there is enough room in the FIFO to store such a block.
- Read audio FIFO write pointer as a multiple of 4 words
  Indicates, where the next audio data will be written in the audio FIFO. Only the bits 17 to 2 of the 18 bits address are presented.
- Read audio FIFO read pointer as a multiple of 4 words
  Indicates, where the next audio data will be read from the audio FIFO. Only the bits 17 to 2 of the 18 bits address are presented.
- Audio FIFO free size as a multiple of 4 words
  This register is updated by the system controller prior to each interrupt or when it is explicitly requested.

The video and audio FIFO usage should be handeled by the FMPEG device driver (Application Notes, chapter 7.1., "FmpegDevice::SetFifoSpaceThreshold" on page 86, "FmpegDevice::GetFifoSpace" on page 84).

6.2.2. DRAM access

Via two address areas random access to the DRAM is provided. This allows e.g. to read back the reconstructed video data.

Random access DRAM page address

Host access: write address area: 1xxx hex

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AVI	-	-	-	-	-	-	Random access DRAM page address									

Bit	Function
Bit 9 ... 0	Random access DRAM page address The 9 LSB's are taken as the DRAM page address.
Bit 15...10	Reserved

Table 12: Random access DRAM page address

Random access word address

Host access: write address area: 3xxx hex

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AVI	-	-	-	-	-	-	Random access word address									

Bit	Function
Bit 9 ... 0	Random access word address The 9 LSB's are taken as the word address. When the page or word address is selected with the address port all accesses to the data port are interpreted as "DRAM random access". The DRAM address is automatically incremented after each access.
Bit 15...10	Reserved

Table 13: Random access word address

6.3. Video

6.3.1. Video display control

TV- and VGA-display mode (register VCON [DMSD, VSP, HSP])

The video display process can be synchronized to the DMSD synchronization signals (TV-display mode) or to the VGA synchronization signals (VGA-display mode). Since the FMPEG does not generate syncronization signals for the video data output on its own DMSD or VGA synchronization signals have to be provided externally (Should be set by the driver software, Application Notes, chapter 7.1., "FmpegDevice::SetVideoSyncInput" on page 69).

Depending on the display mode the FMPEG uses the different input signal lines and provides some different functions. When the TV-display mode is chosen the FMPEG is able to detect even and odd fields (see figure 14 on page 22). Except for VMC8 video output formats video data is ouput every second cycle of the line locked clock LLC (qualified with CREF) in this mode. In VGA-display mode video data is output during each cycle of the clock VIDEOCLK when enabled. Additionally the polarity of the synchonization signals HSYNC and VSYNC is programmable (Should be set by the driver software, Application Notes, chapter 7.1., "FmpegDevice::SetHSyncPolarity" on page 70 and "FmpegDevice::SetVSyncPolarity" on page 69).

Typical clock frequencies for the line locked clock LLC are: 24.5454 MHz (60 Hz field frequency), 27 MHz (50 Hz or 60 Hz field frequency) or 29.5 MHz (50 Hz field frequency). A typical VIDEOCLK clock frequency is 25.176 MHz.

Video size (register VCON [HUS, HIP, LREP]

The FMPEG provides four output options to determine the horizontal and vertical size of the video output. These functions must be selected depending on using TV- or VGA-display mode, the video output size chosen by the user and the size of the reconstructed picture (motion video in SIF format or high resolution still picture)

Horizontal output options:

Horizontal upsampling (VCON [HUS]):
Using horizontal upsampling twice the horizontal size of a reconstructed picture is obtained at the video output. If horizontal upsampling is not enabled the reconstructed video is output with the original resolution in horizontal direction.

Horizontal interpolation (VCON [HIP]):
This function only useful if horizontal upsampling is enabled. Horizontal upsampling can be achieved either by pixel repeat (each pixel of the reconstructed picture is output twice) or by horizontal interpolation with a 1-2-1 filter.

Vertical output options:

Vertical line repeat (VCON [LREP]):
If vertical line repeat is not used, each line of the reconstructed video is output only once during a frame/field period. If vertical line repeat is enabled each line of a reconstructed picture is output two times successively.

These functions should be set by the driver software (Application Notes, chapter 7.1., "FmpegDevice::SetDisplayMode" on page 71 and "FmpegDevice::SetUpsamplingMode" on page 72).

Inactive Video mode (register VCON [IVM])

The Inactive Video mode can be blanked or tristated. This means that video data lines VD[23 ..00) are eiter tristated or that they are containg the blanking values (Y = 16, U = 128, V = 128 or R=G=B = 10 hex) during the time the area outside the display window (see chapter 6.3.3. on page 44) is output (Should be set by the driver software, Application Notes, chapter 7.1., "FmpegDevice::SetInactiveVideoMode" on page 71).

Video configuration register (VCON)

Host access: write only address: B080 hex (45184 dec)

This register must only be accessed when the display mode is disabled.

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VCON	SDI	0	ADLD	DLD	-	-	-	-	-	IVM	LREP	HIP	HUS	HSP	VSP	DMSD

Bit	Function
Bit 0	DMSD Mode (DMSD) 0 : VGA-display mode (VGA syncronization signals used, default) 1 : TV-display mode (DMSD syncronization signals used) VTS determines whether the synchronization signals and video clock from the DMSD pins (LLC, CREF, HREF, VS) or the VGA pins (VIDEOCLK, HSYNC, VSYNC) are used. This setting also determines which clock signal LLC or VIDEOCLK is used for audio clock generation, if VCON[SDI] is set to "0".
Bit 1	Vertical Sync Polarity (VSP) 0 : Active LOW VSYNC (default) 1 : Active HIGH VSYNC VSP deterrmines the vertical sync active level in case of the VTS bit set to VGA-timing. When DMSD-timing is selected, the VPS bit is a don't care.
Bit 2	Horizontal Sync Polarity (HSP) 0 : Active LOW HSYNC (default) 1 : Active HIGH HSYNC HSP determines the horizontal sync active level in case of the VTS bit set to VGA-timing. When DMSD-timing is selected, the VPS bit is a don't care.
Bit 3	Horizontal UpSampling (HUS) 0 : Upsampler enabled (default) 1 : Upsampler disabled HUS determines whether or not the reconstructed video format is upsampled by the factor two in horizontal directionprior to output.
Bit 4	Horizontal InterPolation (HIP) 0 : Horizontal interpolator enabled (default) Luminance pixels are horizontally interpolated with 1-2-1 filter. The position of the chrominance pixels is shifted one pixel to the left and the missing pixels are interpolated. 1 : Horizontal sample repeat When the HUS bit is set to "0" (Upsampler enabled), the upsampling can be done by sample repeat (HIP = 1) or sample interpolation (HIP = 0).
Bit 5	Vertical Line Repeat (LREP) 0 : Vertical line repeat OFF (default) 1 : Vertical line repeat ON When the VLR bit is set all lines to be displayed are shown twice in one frame period.
Bit 6	Inactive Video Mode (IVM) 0 : Blanking output. Y = 16, U = 128, V = 128 or R=G=B = 10 hex (default) 1 : Tri-state output Status of the video pins VD[23..00] outside the display window.
Bit 11 ... 7	Reserved

Table 14: Video configuration register (VCON)

Bit	Function
Bit 12	Download mode for system controller (DLD) Can be used as soft reset.
Bit 13	Audio decoder download (ADLD) Can be used as soft reset.
Bit 14	always set to "0"
Bit 15	Select audio DTO input (SDI) This bit determines if one of the video clocks (LLC or VIDEOCLK) or a clock which is applied to the ASC pin (pin 38) is used for internal generation of the audio clock (refer to chapter 6.5. on page 57). 0 : Video clock selected as source clock for the audio clock 1 : ASC input selected as source clock for the audio clock

Table 14: Video configuration register (VCON)

6.3.2. Video output formats

YC_RC_B 422 to RGB 888 conversion

The FMPEG is capable to convert YC_RC_B to RGB video output. This is done in two steps. In the first step the YC_RC_B 422 format is converted to the YC_RC_B 444 format. This is done by copying the chrominance pixels. The second step is the transformation to RGB, where the following equations are applied:

$R = Y + 1.371 (C_R - 0.5)$

$G = Y - 0.698 (C_R - 0.5) - 0.336 (C_B - 0.5)$

$B = Y + 1.732 (C_B - 0.5)$

The R,G and B values are limited to a value between 0 and 255. In the implementation both YC_RC_B and RGB will be represented by 8 bits per component.

Signal to pin mapping

The FMPEG IC supports various output formats using 24, 16 or 8 video data output lines of the video bus VD(23 ... 0). The video output format should be set by the FMPEG driver software.

The FMPEG is capable to output luminace/chrominace or RGB video data using different signal to pin mappings as follows:

Using 24 ouput pins: - DMSD RGB888

Using 16 output pins: - DMSD YUV ($U = C_B$, $V = C_R$)
- VMC16 YC_RC_B
- VMC16 RGB555
- VMC16 RGB565
- VMC16 RGB888

Using 8 output pins: - VMC8 YC_RC_B
- VMC8 RGB555
- VMC8 RGB565

For all VMC-type output formats normally the VGA-timing is selected. For the VMC8 formats, 16 bits video data is again time multiplexed (refer to table 16 on page 43 and table 17 on page 43). So it takes two clock cycles of the VIDEOCLK to output one pixel on the video data lines, which means that reading the data is done at half the realtime speed. At the end of each line how-

ever, the internal fifo will be cleared. This means that (since the active line is 640 pixels) the picture must be not wider than 320 pixels and left aligned with HSYNC.

For VMC16 RGB888 the 32 bit data is multiplexed over 16 output pins. So, here the same restrictions apply for picture width and position.

If a VMC8 video format is used together with DMSD specific synchronization signals the video data is output with every LLC clock cycle and not with every second like when VMC16 or DMSD formats are used.

The video output formats should be set by the FMPEG driver software (Application Notes, chapter 7.1., "FmpegDevice::SetVideoOutputFormat" on page 70).

Table 15, Table 16 and Table 17 are showing the signal to pin mapping on video bus VD(23 ... 0).

	DMSD RGB888	DMSD YUV ($U=C_B$, $V=C_R$)		VMC16 YC_RC_B	
Pin	Component	Component		Component	
		1st	2nd	1st	2nd
VD[23]	R7	-	-	C_R7	Y27
VD[22]	R6	-	-	C_R6	Y26
VD[21]	R5	-	-	C_R5	Y25
VD[20]	R4	-	-	C_R4	Y24
VD[19]	R3	-	-	C_R3	Y23
VD[18]	R2	-	-	C_R2	Y22
VD[17]	R1	-	-	C_R1	Y21
VD[16]	R0	-	-	C_R0	Y20
VD[15]	G7	Y17	Y27	-	-
VD[14]	G6	Y16	Y26	-	-
VD[13]	G5	Y15	Y25	-	-
VD[12]	G4	Y14	Y24	-	-
VD[11]	G3	Y13	Y23	-	-
VD[10]	G2	Y12	Y22	-	-
VD[09]	G1	Y11	Y21	-	-
VD[08]	G0	Y10	Y20	-	-
VD[07]	B7	U7	V7	C_B7	Y17
VD[06]	B6	U6	V6	C_B6	Y16
VD[05]	B5	U5	V5	C_B5	Y15
VD[04]	B4	U4	V4	C_B4	Y14
VD[03]	B3	U3	V3	C_B3	Y13
VD[02]	B2	U2	V2	C_B2	Y12
VD[01]	B1	U1	V1	C_B1	Y11
VD[00]	B0	U0	V0	C_B0	Y10

Table 15: Signal / pin mapping for DMSD RGB888, DMSD YUV and VMC16 YC_RC_B

Pin	VMC16 RGB555		VMC16 RGB565		VMC16 RGB888	
	Component		Component		Component	
	1st	2nd	1st	2nd	1st	2nd
VD[23]	G25	0	G24	R27	R7	0
VD[22]	G24	R27	G23	R26	R6	0
VD[21]	G23	R26	G22	R25	R5	0
VD[20]	B27	R25	B27	R24	R4	0
VD[19]	B26	R24	B26	R23	R3	0
VD[18]	B25	R23	B26	G27	R2	0
VD[17]	B24	G27	B24	G26	R1	0
VD[16]	B23	G26	B23	G25	R0	0
VD[15]	-	-	-	-	-	-
.....	...	...	...	...	...	...
VD[08]	-	-	-	-	-	-
VD[07]	G15	0	G14	R17	B7	G0
VD[06]	G14	R17	G13	R16	B6	G1
VD[05]	G13	R16	G12	R15	B5	G2
VD[04]	B17	R15	B17	R14	B4	G3
VD[03]	B16	R14	B16	R13	B3	G4
VD[02]	B15	R13	B15	G17	B2	G5
VD[01]	B14	G17	B14	G16	B1	G6
VD[00]	B13	G16	B13	G15	B0	G7

Table 16: Signal / pin mapping for VMC16 RGB555, RGB565 and RGB888

Pin	VMC8 YC_RC_B				VMC8 RGB555				VMC8 RGB565			
	Component				Component				Component			
	1st	2nd	3rd	4th	1st	2nd	3rd	4th	1st	2nd	3rd	4th
VD[23]	-	-	-	-	-	-	-	-	-	-	-	-
.....	...	...	...	...	...	...	...	...	...	...	...	...
VD[08]	-	-	-	-	-	-	-	-	-	-	-	-
VD[07]	C_B7	Y17	C_R7	Y27	G15	0	G25	0	G14	R17	G24	R27
VD[06]	C_B6	Y16	C_R6	Y26	G14	R17	G24	R27	G13	R16	G23	R26
VD[05]	C_B5	Y15	C_R5	Y25	G13	R16	G23	R26	G12	R15	G22	R25
VD[04]	C_B4	Y14	C_R4	Y24	B17	R15	B27	R25	B17	R14	B27	R24
VD[03]	C_B3	Y13	C_R3	Y23	B16	R14	B26	R24	B16	R13	B26	R23
VD[02]	C_B2	Y12	C_R2	Y22	B15	R13	B25	R23	B15	G17	B25	G27
VD[01]	C_B1	Y11	C_R1	Y21	B14	G17	B24	G27	B14	G16	B24	G26
VD[00]	C_B0	Y10	C_R0	Y20	B13	G16	B23	G26	B13	G15	B23	G25

Table 17: Signal / pin mapping for VMC8 YC_RC_B, RGB555 and RGB565

6.3.3. Windowing

The location where reconstucted video is output relative to the vertical and horizontal synchronisation signals and the visible part of reconstructed video are determined by window parameters, which define the display window, the viewport and the reconstructed picture window. Additionaly the Pixelqualifier parameters flag an area on the display where the reconstructed video data have to be shown. This flag is output on the PXQ (Pixel qualifier) pin of the FMPEG.

Counting the window parameters is always related to the number of reconstructed MPEG SIF pixels and lines, which can be output during the time of two successive vertical synchronization events (VSYNC for VGA timing and VS for DMSD timing), so for one frame period or one field period respectively.

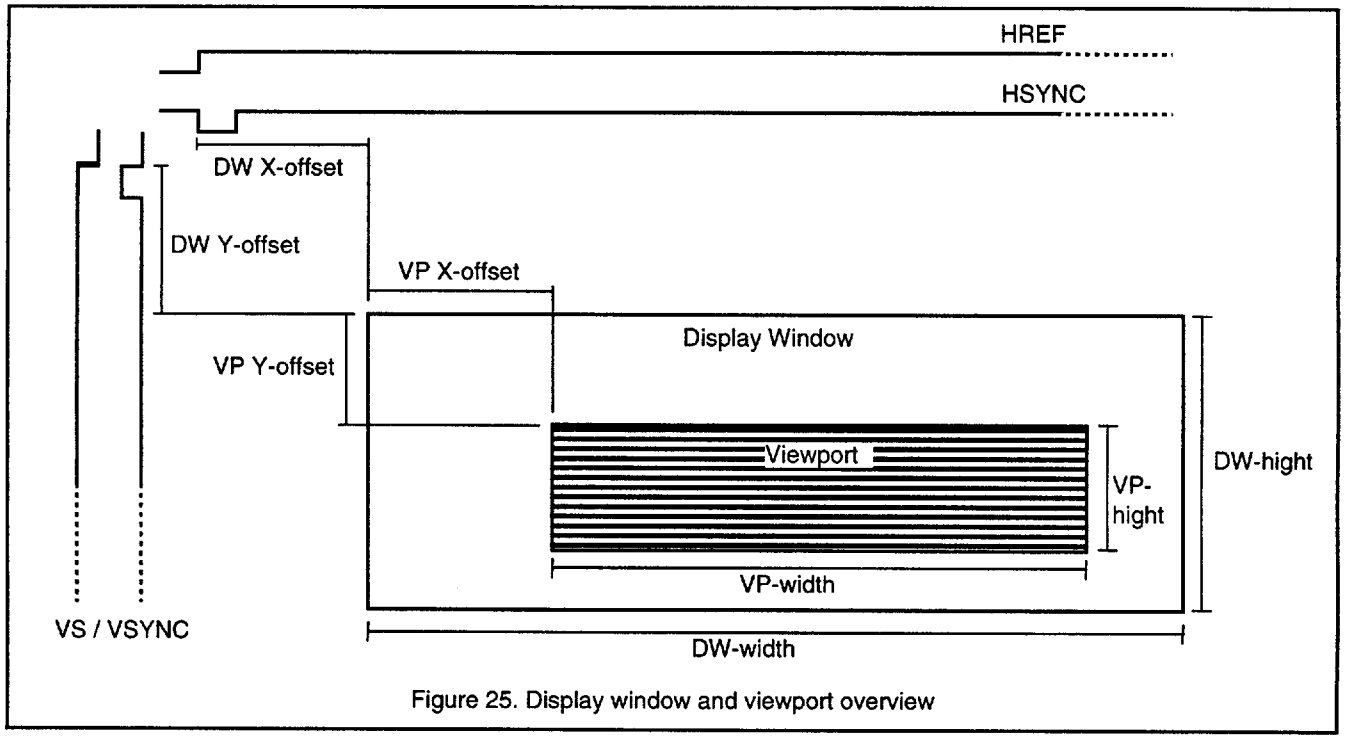


Figure 25. Display window and viewport overview

The socalled "display window" defines an oblong window where the active (non blanked) video is shown (see Figure 25.). The display window is defined by four parameters (DW X-offset, DW Y-offset, DW-width, DW-hight) The horizonal and vertical offsets are measured relative synchronization signals (Application Notes, chapter 7.1., "FmpegDevice::SetDisplayWindow" on page 73).

The video data output lines can be set to two different staus in that area of the output, which is not occupied by the display window (Application Notes, chapter 7.1., "FmpegDevice::SetInactiveVideoMode" on page 71):

- All data outside the display window is blanked. In RGB mode this means: R = G = B = 10 hex . In YC_RC_B mode Y = 10 hex and $C_R = C_B$ = 80 hex.
- All data outside the display window is tri-state.

Additionally the video output can be enabled or disabled completely. This function is synchronized to VSYNC or VS depending on the Synchronisation signals which are used (Application Notes, chapter 7.1., "FmpegDevice::SetVideoOutputEnable" on page 76).

The viewport defines the area where the video is shown inside the display window and is defined with four parameters too (VP X-offset, VP Y-offset, VP-width, VP-hight , should be set by the FMPEG driver software, Application Notes, chapter 7.1.,

"FmpegDevice::SetViewportWindow" on page 74). The colour of that area of the display window, which is not occupied by the viewport can be select (Application Notes, chapter 7.1., "FmpegDevice::SetBorderColor" on page 75). The output of the viewport again can be hidden or shown synchronized with VSYNC or VS respectively (Application Notes, chapter 7.1., "FmpegDevice::SetViewportEnable" on page 76).

A picture offset is defined, which determines that pixel of the reconstructed MPEG video which is shown in the top left corner of the view port. The two parameters of the picture offset are measured from the top left corner of the reconstructed video. That part of the reconstructed MPEG video which is than displayed inside the viewport is allowed to be smaller than the viewport itself but can be larger since the viewport limoits the visible area of the reconstructed video (Application Notes, chapter 7.1., "FmpegDevice::SetPictureOffset" on page 72).

The pixel qualifier again is defined by four parameters. It flags an rectangular area during video data output and is provided for external use via the PXQ pin (pin) of the FMPEG. It's horizonal and vertical offsets are measured relative synchronization signals and it's width and hight relative to the offsets (Application Notes, chapter 7.1., "FmpegDevice::SetQualifierWindow" on page 74). The timing of this signal varies depending on using DMSD or VGA specific synchronisation signals (Refer to figure 15 on page 22 and figure 18 on page 24).

The windowing functions should be handeled by the FMPEG driver software since their values have to be set depending on using upsampling functions and the use synchronisation DMSD or VGA specific synchronisation signals.

6.3.4. Upsampling and Still picture modes

The FMPEG provides horizontal upsampling and vertical linerepeat to double the horizontal and vertical size of displayed video. In horizontal direction. Additionally horizontal upsampling can be performed by pixelrepeat or by interpolation with a 1-2-1 filter.

If DMSD specific synchronisation signals are used an additional mode, the interlaced mode, can be selected. When the interlaced mode is selected even lines of the reconstructed video are output in even fields and odd lines are output during odd fields.. When the interlaced mode is not used all lines of the reconstruced video are read progressively.

The FMPEG decoder allows decoding (still) and storage of pictures with an area of up to 704 * 480 pixels. There are several possibilities for display depending on the size of reconstructed still picture and the chosen display mode (TV- or VGA-display mode).

Reconstructed still pictures with a size up to MPEG SIF resolution are handled like motion video. So horizontal and vertical upsampling can be used to enlarge the picture for displaying.

When reconstructed still pictures with a horizontal size and/or a vertical size greater than MPEG SIF resolution are decoded the video buffers 1, 2 and 3 are used as one buffer to store the amount of reconstructed video data. Decoding and displaying in parallel is not possible in this case. To be able to decode still pictures with more than 1192 macro blocks the video FIFO size and/or the audio FIFO size have to be decreased to be able to store the reconstructed video data. If a still picture with the maximum resolution has to be decoded only 17 kbytes will be left for the video FIFO and the audio FIFO.

Additionally different settings for displaying depending on TV- or VGA-display mode have to be used:

1. Displaying still pictures in VGA-display mode:
   To display still pictures which exeed the MPEG SIF resolution either in horizontal or in vertical direction vertical line repeat or horizontal upsampling can be used to adapt the aspect ratio if this is required. Still pictures with a resolution up to 640 *480 pixels can be displayed completely on a standart VGA graphics display. In this case horizontal upsampling and vertical line repeat aren't used to obtain a full screen output.
   If a still picture with the maximum size of 704 * 480 pixels is decoded it can't be displayed completely on the standart VGA graphics display. In this case a reconstructed picture offset has to be set to select a part of the still picture for displaying.

2. Displaying still pictures in TV-display mode:
   The displaying process in horizontal direction is in TV-display mode the same as VGA-display mode. So horizontal upsampling can be used to adapt the reconstructed picture to the vodeo output in horizontal direction.
   Because in TV-display mode an interlaced output is used each reconstructed picture is normally output in both fields. So a decoded picture in MPEG SIF resolution fully fills the display if at the same time horizontal upsampling is used. In this case vertical line repeat is not selected.
   To display a still picture with a higher resolution up to the maximum size of 704 * 480 pixels the horizontal upsampling must be disabled and the video output must be set to interlaced readout mode. This means that even lines of the reconstructed still picture are read during even fields and odd lines are read during odd fields.

The upsampling, linrepeat and interlace functions should be handeled by the FMPEG driver software.

6.3.5. Video play control

Compressed video stream selection:

An ISO11172 stream allows 16 video streams to be multiplexed. The FMPEG video decoder can decode one of the 16 . The video decoder will store only the selected video stream data in the video FIFO. The stream number can be changed at any time. The video decoder will decode/display data of the new stream with the highest priority (Application Notes, chapter 7.1., "FmpegDevice::SetVideoStreamID" on page 80).

Input filters:

Before the decoder can start decoding, first an I-type picture must be found. Other type pictures must be skipped. When required, the decoder can also skip data until a sequence header is found or until a group_of_pictures header is found filter. As soon as a sequence header or group_of_pictures header is found the filter must be disabled. The video decoder will always skip user data and extension data.

Video play back functions:

- Play forward:
  Decode and display the selected video stream with a selectable factor to normal speed. The decoder will, in case of a system layer, use the System Clock Reference and decoding time stamps to synchronize the play back process. In case of single bit-stream mode the VBV-delay must be used as decoding delay.
- Pause:
  Freeze the display. The decoding process will stop. The pause will be cancelled by a new "play forward" command or a "step" command.
- Still:
  The decode handles an incoming I-type picture as still picture.
- Skip:
  The content of one video buffer is displayed continously and decoding of I and P-frames is continoued to be able to reenter the play mode immediately, so that no search for an I-picture has to be performed.
- Stop:
  To stop the video decoding a reset of the video decoder is performed.

Further Video decoder commands:

- Setting Layer available
  - Single bitstream
  - System layer (default)
- Search for an I-type picture
  All data will be skipped at the input until an I-picture is found. a command completed interrupt will be send to the host when the next I-picture is found and the decoder will pause.
- Search for Group of pictures
  Search for group of pictures header. When this bit is set the decoder will skip all pictures until a GOP header is found. When a GOP header is found a command completed interrupt will be given if enabled.
- Search for Sequence
  Search for a sequence header. When this bit is set the decoder will skip all pictures until a sequence header is found. Then a command completed interrupt will be send to the host if enabled.
- Reset
  This command resets the video decoder. The ISA bus interface and the micro code do not need to be reinitialized if enabled.
- Disable decoding of B-type pictures
- Disable decoding of P-type pictures.
- Count I pictures only
- Disable decoding of I-type pictures
- Disable partial decoding
  Partial decoding of B-pictures can be needed, when the bottom part of a picture is displayed at the top of the screen, so just after a VS or VSYNC occured. Since decoding always starts synchronized with VSYNC or VS there would not be enough time left to decode a whole picture if the top border of the reconstructed picture would exceed the screen.
- Disable re-decoding
  Redecoding is used by the decoder to allow scrolling of partial decoded pictures, when e.g. the decoder is running in slow motion.
- FAST decoding
  The decoding process is done as fast as possible. In this mode the play back speed value is not taken into account.
- Set the number of pictures to decode

Video status information:

- Video system clock reference
  32 bits of the Video system clock reference can be read which is counted with 90/32 KHz MPEG system clock. The LSB of the 33 bit wide Video system clock reference is not presented.
- Video decoding time stamp
  32 bits of the last Video decoding time stamp which was available in the MPEG data stream. The LSB of the 33 bit wide Video time stamp is not presented.
- Pixel aspect ratio
- Picture rate
- Status of the currently displayed picture:
  - Picture height, picture width
  - Displayed picture is first of a sequence
  - Displayed picture is first of a group of pictures
  - Picture startcode
  - Displayed picture is the last of a sequence
  - Broken link bit is set in the GOP header

- Closed GOP bit is set in the GOP header
- Displayed picture is a P-type picture
- Displayed picture is a B-type picture
- Displayed picture is a D-type picture
- Displayed picture is an I-type picture
- Number of pictures decoded or skipped since last reset
- Number of pictures displayed since last reset
- Timcode in houres, minutes, seconds and pictures

The video play control functions and status information should be handeled by the FMPEG software driver. Several functions and status informations which are available are used to realize different application functions by combining them (Application Notes, chapter 7.1., "FmpegDevice::SetVideoStreamID" on page 80, "FmpegDevice::SetPlaybackSpeed" on page 80, "FmpegDevice::StartPlayback" on page 82, "FmpegDevice::StepPictures" on page 82, "FmpegDevice::SetFreezeMode" on page 81, "FmpegDevice::StopPlayback" on page 83 and "FmpegDevice::ResetDecoder" on page 83).

6.3.6. Video Interrupts

Interrupts can be generated by the FMPEG at any time. Each type of interrupt can be enabled or disabled.

If an interrupt occurs this is indicated to the host by the one of the interrupt request lines of the FMPEG (IRQ[15, 12, 11, 10]) which was selected during the ISA Bus interface setup.

Additionally the FMPEG provides a socalled 'video timer' which can be used to generate Interrupts on regulary intervals. An internal register is incremented using the 90 KHz clock until the user specified value is reached. Than an interrupt is generated (if enabled) and the internal register is set to zero.

The following socalled video interrupts can be generated by the FMPEG:

- Sequence code found interrupt
  Set at a VSYNC/VS when the displayed picture (when displayed the first time) is the first of a sequence.
- Group_of_pictures code found interrupt
  Set at a VSYNC/VS when the displayed picture (when displayed the first time) is the first of a group of pictures.
- Picture code found interrupt
  Set at a VSYNC/VS when a picture is displayed the first time.
- End_of_sequence code found interrupt
  Set at a VSYNC/VS when the displayed picture (when displayed the first time) is the last of a sequence.
- VSYNC/VS interrupt
  Set at the leading edge of a VSYNC/VS.
- Fifo Space available interrupt
  Set when for both the Audio and Video FIFO there is room to store data like secified as 'Video block size' and 'Audio block size' (refer to chapter 6.2.1. on page 36). This condition is only evaluated at the moment another (enabled) interrupt will be generated or at a VSYNC/VS.
- Window updated interrupt
  Set at VSYNC/VS when a requested window update is carried out.
- Command Completed interrupt
  Set when:
  - A Clear command is carried out.
  - A Serch for sequence command is carried out.

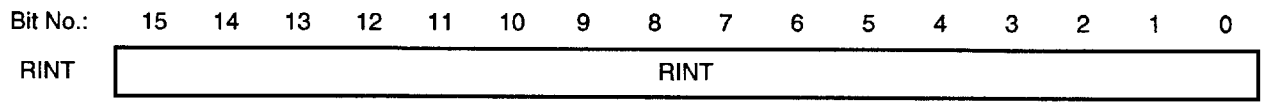
- A Serarch for GOP command is carried out.
- A Skip command is carried out.

- Sequence code found by decoder interrupt
  Set when the video decoder has detected this code. When this interrupt bit is set also the 'Group_of_pictures code found by decoder' and 'Picture code found by decoder' interrupt bits will be set.
- Group_of_pictures code found by decoder interrupt
  Set when the video decoder has detected this code. When this interrupt bit is set also the 'Picture code found by decoder' interrupt bit will be set.
- Picture code found by decoder interrupt
  Set when the video decoder has detected this code.
- End of sequence at decoder interrupt
  Set when the decoder has detected this code.
- Timer interrrupt
  Set at a VSYNC when the video timer is expired. The video timer can be used to generate regulary interrupts independent from a VSYNC or VS respectively.
- Decoder delayed error interrupt
  Set at a VSYNC when the decoder should have started decoding according to the time stamp but cannot start because it is not ready decoding the previous picture. (This can be caused by FIFO underflows).
- Decoder error interrupt
  Set when the video decoder encounters a violation of MPEG data stream format. It is not guaranteed that all types of errors are detected.
- Video data input error interrupt
  Set when the video system layer parses encounters a violation of the system layer format. It is not guaranteed that all types of error are detected.

When the status has been read the last action to reset the interrupt is an access to the reset interrupt register (RINT).

Reset interrupt register (RINT)

Host access: read/write address: B082 hex (45186 dec)



Bit	Function
Bit 15...0	Reset interrupt (RINT) If this register is accessed the interrupt pin is released. This register must be accessed to reset a Video interrupt as well as an Audio interrupt.

Table 18: Reset interrupt register (RINT)

The video interrupts should be handeled by the FMPEG driver software (Application Notes, chapter 7.1., "FmpegDevice::SetInterruptEnable" on page 85 and "FmpegDevice::GetInterruptStatus" on page 85).

6.4. Audio

Figure 26 shows a block diagram of the audio reconstruction unit. Audio data which shall be played back is fed from the audio FIFO into the audio reconstruction unit can be coded audio data or PCM data. The MPEG AUDIO DECODER is bypassed if PCM audio data is played back.

The audio decoder contains a audio post processing unit (PPU) where the left and right channel audio data can be mixed and attenuated. Additionally reconstructed audio or PCM audio can be combined with an external audio source (ADD & LIMIT block). Then the audio data is formatet to a serial data stream (IIS or LSB fixed format).

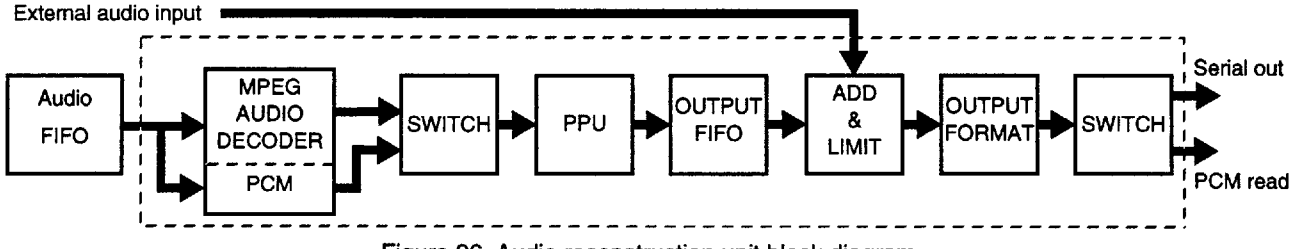


Figure 26. Audio reconstruction unit block diagram

6.4.1. Attenuation / Mixing of MPEG audio

The audio post processing unit (PPU figure 26, Block diagram in figure 27) attenuates and mixes the decoded MPEG audio data. Therefore four attenuator values have to be set. Two of these values determine the left to left and right to right output level of the left and right audio channel. The other two values are used to add left channel audio data to the right audio cannel output and vice versa. The audio processing unit attenuates the left and the right channel and/or the inter channel crosstalk in 1 dB steps from 0 dB to -62 dB or instantanously mutes the channels. As a result it is possible to obtain e.g. a mono audio signal from the FMPEG or to switch the left and the right audio output.

The transfer function of the Audio post processing unit (PPU) is:

LEFT_OUT = LIMIT { (LEFT_IN * LL) + (RIGH_IN * RL) }

RIGHT_OUT = LIMIT { (RIGHT_IN * RR) + (LEFT_IN * LR) }

LIMIT { P } = P if P ≤ 0 dB, 0 dB if P > 0 dB

LL (LEFT_IN to LEFT_OUT attenuation), LR (LEFT_IN to RIGHT_OUT attenuation), RR (RIGHT_IN to RIGHT_OUT attenuation) and RL (RIGHT_IN to LEFT_OUT attenuation) are the attenuator values in the range from 0 to 63. The relation between attenuator value and the actual attenuation in dB is shown in table 51.

The attenuator values should be set by the FMPEG driver software (Application Notes, chapter 7.1., "FmpegDevice::SetAudioVolume" on page 79).

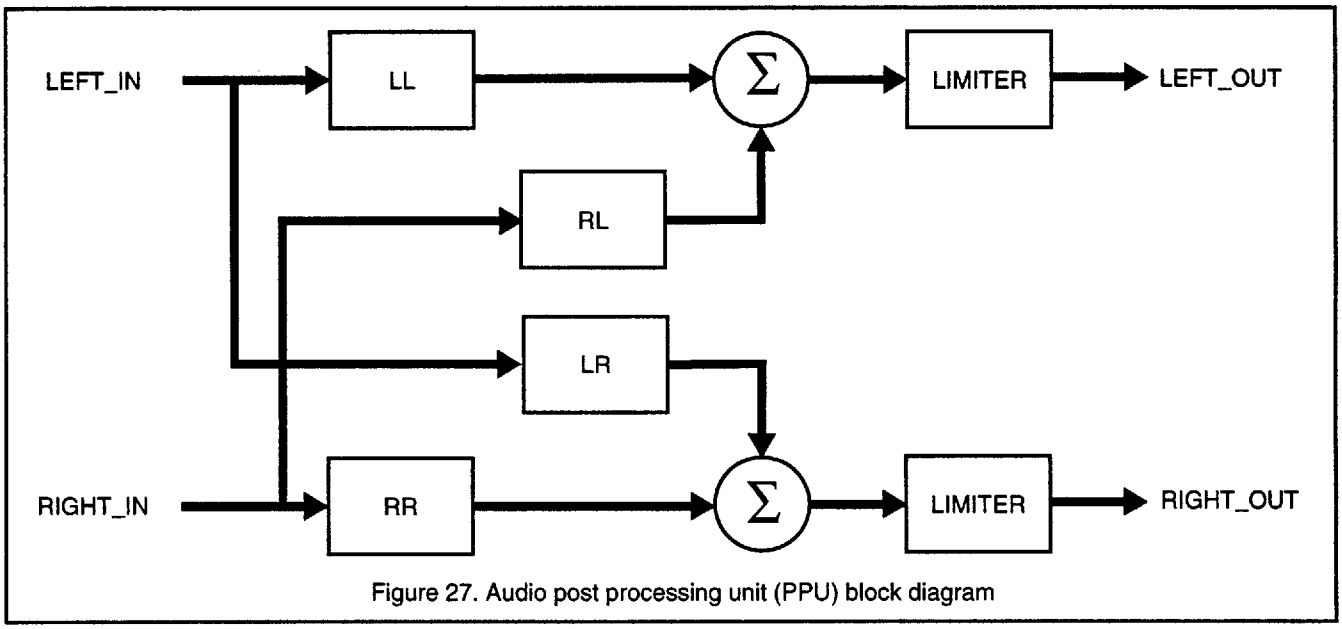


Figure 27. Audio post processing unit (PPU) block diagram

Attenuator Value LL, LR, RR or RL	Attenuation
0	0 dB
1	1 dB
2	2 dB
3	3 dB
4	4 dB
. . . .	. . . .
60	60 dB
61	61 dB
62	62 dB
63	∞ dB

Table 19: Attenuator value versus Attenuation

6.4.2. Audio output and input formats

Serial audio output (register AFR[AOF, ACC])

The serial output can be either in Philips IIS or Sony format. The number bits per sample is 32. The output accuracy of the audio samples is rounded to 16, 18 or 20 bits by the Output formatter. The FMPEG audio output is the timing master and generates the bit clock and word clock. The bit clock frequency depends on the sample frequency (table 20).

Sample frequency (Fs)	Bit clock
32 Khz.	2.0480 MHz.
44.1 KHz.	2.8224 MHz.
48 KHz.	3.0720 MHz.

Table 20: Sample frequency / bit clock

The audio clock which depends on the sample frequency of the coded MPEG audio can be generated by the FMPEG itself using the audio DTO (see chapter 6.5. on page 57) or can be taken from an external source (refer to chapter 6.5.1. on page 58). The frequency of this audio clock is always 265 * Fs where Fs is the sample frequency.

The audio output format should be set by the FMPEG driver software (Application Notes, chapter 7.1., "FmpegDevice::SetAudioOutputFormat" on page 78, "FmpegDevice::SetAudioOutputEnable" on page 78).

External audio input (register AFR[XAF, XAC, EXA])

The external audio serial input can be either in Philips IIS or Sony format and the external audio input can be timing master or timing slave. If the FMPEG is the timing master for the external audio source than the bit clock and the word clock are taken from the serial audio output (CLOUT, pin 135 and WSOUT, pin 136). However the serial audio data format of the audio input and the audio output can be still different in this case.

If the external audio source generates the bit clock and the word clock itself than these signals are input via WSIN and CLIN pins (pins 130 and 131 of the FMPEG). In this mode the FMPEG drops samples from the external audio signal or inserts zero samples if the bit clock of the external source and the internal source have a phase drift to each other.

The number of bits per sample of the external serial audio input is 32. The possible sample accuracies of the external audio input are: 8, 16, 18 or 20 bits. The sample frequency is always identical for serial out and serial in and table 20 also applies for the external audio input. However the serial audio data format (IIS or LSB fixed) can be different for external audio input and the audio output.

The external audio data is added to the attenuated reconstructed MPEG audio or PCM audio data and limited to the 0 dB level before it is output (see figure 26 on page 50).

The audio input format should be set by the FMPEG driver software (Application Notes, chapter 7.1., "FmpegDevice::SetAudioInputFormat" on page 77, "FmpegDevice::SetAudioInputEnable" on page 77).

PCM read back (register AFR[ARB] and area APO)

The FMPEG allows to read back (decoded) PCM data by using the AFR[ARB] function. The audio decoder can store 32 stereo audio samples in the socalled subband fifo. When this fifo is filled a fifo full interrupt will be sent to the HOST if it is enabled.

When the HOST has read a fifo contents via the audio PCM ouput area (APO) the audio processor will write a new samples into the subband fifo. The first sample read after an interrupt is always a left sample. The next samples are alternating right and left samples.

The interrupt can be masked. This allows polling of the interrupt status registers when interrupts are not wanted. The audio decoding process is slaved to the read back process. This means that decoding stops as soon as the subband fifo is filled and no data is read via the APO area. Because of the non-real time effect of PCM read back the audio is not audiblein this mode. So the audio decoder will automatically output all zero's on the serial output. It is the responsibility of the application to select the correct output accuracy of 16 bits.

Audio format register (AFR)

Host access: read/write address: B081 hex (45185 dec)

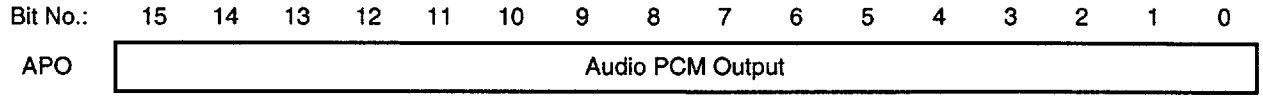
Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFR	"0"				ARB	-	-	EXA	XAC		XAF		ACC		AOF	

Bit	Function
Bit 1, 0	Audio output Format (AOF) 00 : Philips IIS (default) 01 : Sony 10 : Reserved 11 : Reserved
Bit 3, 2	Audio output accuracy (ACC) 00 : 16 bits (default) 01 : 18 bits 10 : 20 bits 11 : Reserved
Bit 5, 4	External audio format (XAF) 00 : Philips IIS (default) 01 : Sony 10 : Reserved 11 : Reserved
Bit 7, 6	External Audio accuracy (XAC) 00 : 16 bits (default) 01 : 18 bits 10 : 20 bits 11 : 8 bits
Bit 8	Enable external audio (EXA) 0 : External audio OFF (default) 1 : External audio ON
Bit 10 ... 9	Reserved
Bit 11	Audio read back (ARB) 0 : Audio read back disabled (default) 1 : Audio read back enabled
Bit 15...12	Set to "0"

Table 21: Audio format register (AFR)

Audio PCM output area (APO)

Host access: read address area: Dxxx hex



Bit	Function
Bit 15 ... 0	Audio PCM output When the audio decoder is in audio read back mode, PCM data can be read from the audio PCM output. When it is indicated that the subband fifo is filled by an interrupt or by a status register 32 audio samples can be read (32 left and 32 right samples.). The first sample read is always a left sample. The following samples, which are read, are alternately right and left samples.

Table 22: Audio PCM output area (APO)

The audio read back function should be handeled by the FMPEG device software driver.

6.4.3. Audio play control

Compressed audio stream selection:

The audio decoder will store only the selected audio stream data in the video FIFO. The stream number can be changed at any time (Application Notes chapter 7.1., "FmpegDevice::SetAudioStreamID" on page 81).

Audio play back functions:

- Audio Play command
  Start play back of audio data
- Audio Stop command
  Stop the audio decoding process. The audio FIFO will be cleared. The audio output will be made zero. All data send to the FIFO will be skipped
- Mute, Demute command
  Mute demute the audio output. Decoding continues but the audio output is muted if a mute command had been given.
- Pause command
  Pause the audio decoding. Decoding of the next audio frame will be inhibited. The pause mode can disabled by the Audio PLAY command or by the audio STOP command. This command is carried out by stopping the audio system clock reference.

Further audio decoder commands

- Selecting the available layer
  - Single bitstream or
  - System layer (default)
- Select playback mode
  - playback of MPEG audio data
  - playback of CDDA data

Video status information:

- Reading the audio system clock reference
  16 Bit 2's complement number representing Bit 21 to 6 of the system clock reference.
- Reading the audio decoding time stamp
  16 Bit 2's complement number representing Bit 21 to 6 of the audio decoding time stamp.
- Reading audio header information such as:
  - Protection bit
  - Represented audio layer
  - MPEG ID
  - Emphasis used
  - Original / home bit
  - Copyright bit
  - Stereo, Joint stereo, Dual Cannel or single channel mode
  - Mode extension for joint stereo
  - Private bit
  - Padding Bit
  - Used sampling frequrncy
  - Bit rate Index

The audio play control functions and status information should be handeled by the FMPEG software driver. Several functions and status informations which are available are used to realize different application functions by combining them (Application Notes, chapter 7.1., "FmpegDevice::SetAudioStreamID" on page 81, "FmpegDevice::StartPlayback" on page 82, "FmpegDevice::StopPlayback" on page 83 and "FmpegDevice::ResetDecoder" on page 83).

6.4.4. Audio interrrupts

Interrupts can be generated by the FMPEG at any time. Each type of interrupt can be enabled or disabled.

If an interrupt occurs this is indicated to the host by the one of the interrupt request lines of the FMPEG (IRQ[15, 12, 11, 10]) which was selected during the ISA Bus interface setup.

The following socalled audio interrupts can be generated by the FMPEG:

- Subband fifo full interrupt
  Set in the PCM read back mode, when the subband fifo contains 32 (left and right) PCM samples.
- Stream updated interrupt
  Set when the audio decoder starts to decode frames of a new stream number. The first selected stream number after reset is also regarded as a "new" stream number.
- Frame header updated interrrupt
  Set when the audio decoder has optained updated audio header information.
- Audio decoder error interrupt
  Set when the CRC check indicates an error or when the audio decoder detects a violation of the MPEG standard while parsing the data stream. It is not guaranteed that all types of violations are detected.
- Audio synchronisation error interrupt
  Set when the audio input parser lost sync with the system stream. This means that no reserved code is found after counting down the packet lenght.

- End of ISO interrupt
  Set when the audio parser detects an End of ISO start code in the data stream (before data is sent to the audio FIFO).

Audio interrupts (register RINT)

After the status has been read the last action to reset the interrupt is an access to the reset interrupt register (RINT, refer to page 49).

The audio interrupts should be handeled by the FMPEG driver software (Application Notes, chapter 7.1., "FmpegDevice::SetInterruptEnable" on page 85 and "FmpegDevice::GetInterruptStatus" on page 85).

6.5. System timing

The FMPEG IC requires several clocks. Figure 28 shows the different clocks (except for the system clock) which are used in the FMPEG and the generation of the audio clock and the MPEG system time clock. The FMPEG operates with the following clocks:

1. System clock.
   This clock is generated by the 40 MHz crystal.

2. Video clock (22 - 32 MHz) e.g.:

   - 27 MHz TV-display mode: (13.5 MHz pixel freq. and 50 Hz/60 Hz field freq.)
   - 29.5 MHz TV-display mode: (14.75 MHz pixel freq. and 50 Hz field freq.)
   - 24.5454 MHz TV-display mode: (12.2727 MHz pixel freq. and 60 Hz field frequency)
   - 25.176 MHz VGA-display mode: (25.176 MHz. pixel freq. and 60 Hz. field frequency)

   The FMPEG always needs a video clock. This video clock can be either the line locked clock input via the LLC pin in TV-display mode or the pixel clock of standert VGA graphics input via the VIDEOCLK pin. The video clock in combination with with the specific synchronazation signals is needed for video data output generation. Additionally it can be used as a source for the audio clock.

3. Audio clock:

   - 8.192 MHz for 32 Khz sample frequency (256 * Fs)
   - 11.2896 MHz for 44.1 Khz sample frequency (256 * Fs)
   - 12.288 MHz for 48 KHz sample frequency (256 * Fs)

   The appopriate audio clock is always needed, because the 90 KHz MPEG system time clock is derived from this clock. The audio clock has to be selected depending on the sample frequency of the audio data and has to be applied via the AUCK pin. This clock can be generated inside the FMPEG using the audio DTO which outputs the via the LLA pin (see below).

4. External audio source clock:
   If it is not wanted to derive the audio clock from a video clock (LLC or VIDEOCLK) the audio clock can be derived from this external clock which is input via the ASC pin of the FMPEG. The frequency of this clock should be greater than twice the needed audio clock frequency and maximum frequency is 32 MHz.

5. MPEG system time clock:
   The MPEG system time clock is derived from the audio clock which is input via the LLA pin and generated inside the FMPEG using the system time clock DTO (STC DTO). For external use this clock is provided via the NCLU pin.

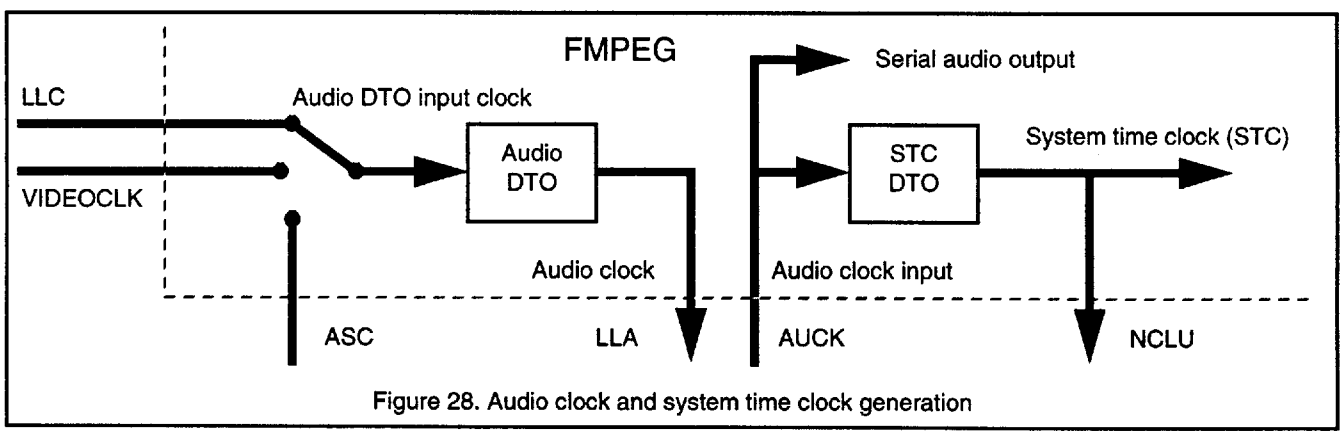


Figure 28. Audio clock and system time clock generation

6.5.1. Generating the audio clock

Audio DTO (registers VADT0,VADT1, VASEL)

The input clock for the Audio DTO can be one of the clocks LLC or the VIDEOCLK which is than selected depending on using DMSD or VGA specific synchronisation signals or a clock signal which is applied to the audio source clock pin (ASC, Pin 38) as shown in Figure 28. The source is selected by the setting the register VCON [DMSD] and VCON [SDI].

The ratio between the input clock of the Audio DTO and the audio clock is determined by two values C0 and C1 which have to be loaded into the two registers VADT0 for the value C0 and VADT1 for the value C1. These values have to be selected depending on the frequency of the input clock of the Audio DTO (provides via the pins LLC, VIDEOCLK or ASC) and the audio clock, which is needed. Values for C0 and C1 are shown in table 25.

Additionally a third register has to be set depending on the ratio between the frequency of the input clock of the Audio DTO and the audio clock frequency. In case of a frequency at the audio DTO input which is equal to or lower than two times the audio clock frequency the OSEL Bit of the VASEL register has to be set to GS (last column of table 25). This results in a shorter time where the audio clock is logic 'high' which is needed in this case.

The internally generated audio clock which is output via the LLA pin may not be jitter free. This frquency jitter depends on the frequency of the Audio DTO input clock and becomes smaller with using input clocks of higher frequencies

The Audio DTO settings should be set by a FMPEG driver software function (see Application Notes, see chapter 7.1. on page 63).

Video to Audio DTO C0, C1 (VADT0, VADT1)

Host access: write VADT0 address: B084 hex (45188 dec)
VADT1 address: B085 hex (45189 dec)

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VADTn	Video to Audio DTO (C0) / Video to Audio DTO (C1)															

Bit	Function
Bit 15 ... 0	Video to Audio DTO (C0, C1) VADT0 (C0) : Value after (power-on) reset: HEX'7000' VADT1 (C1) : Value after (power-on) reset: HEX'7600' The value C0 of the register VADT0 and the value C1 of the register VADT1 are deterrmining the ratio between the frequency of the input clock of the video to audio DTO and the frequency of the audio clock which is generated by the video to audio DTO. The values are listed in table 25 on page 59.

Table 23: Video to Audio DTO register (VADT0, VADT1)

Video to Audio DTO output select (VASEL)

Host access: write address: B086 hex (45190 dec)

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VASEL	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OUT_SEL

Bit	Function
Bit 0	Audio DTO output select (OUT_SEL) 0 : GC (default after (power-on) reset) 1 : CC The Audio DTO output select bit has to be set to GS if the ratio between the frequency of the input clock of the audio DTO and the audio clock frequency is equal or smaller than 2. This value is listed in table 25.
Bit 15 ... 1	Reserved

Table 24: Video to Audio DTO output select register (VASEL)

Audio DTO input clock (LLC, VIDEOCLK os ASC)	Audio clock (LLA)	VADT0 C0	VADT1 C1	OUT_SEL
MHz	MHz	(HEX)	(HEX)	
32.0	11.2896	0x3720	0x9AE0	CC/GC
32.0	8.192	0x4000	0x4600	CC/GC
32.0	12.288	0x6000	0x6600	CC/GC
27.0	11.2896	0x6200	0x77A0	CC/GC
27.0	8.192	0x4000	0x4600	CC/GC
27.0	12.288	0x4000	0xB360	CC/GC
29.5	11.2896	0x3750	0xA715	CC/GC
29.5	8.192	0x2000	0xACC4	CC/GC
29.5	12.288	0x6000	0x7988	CC/GC
24.54	11.2896	0x4360	0xB0E4	CC/GC
24.54	8.192	0x2C00	0xA82A	CC/GC
24.54	12.288	0x5800	0xA838	GC

Table 25: Values for the Video to Audio DTO registers (VADT0, VADT1, VASEL)

Audio DTO input clock (LLC, VIDEOCLK os ASC)	Audio clock (LLA)	VADT0 C0	VADT1 C1	OUT_SEL
MHz	MHz	(HEX)	(HEX)	
25.176	11.2896	0x7000	0x7600	CC/GC
25.176	8.192	0x4000	0x9420	CC/GC
25.176	12.288	0x4000	0xBCC0	CC/GC
22.0	11.2896	0x6E40	0x9768	GC
22.0	8.192	0x4000	0x9420	CC/GC
22.0	12.288	0x6000	0xB420	GC

Table 25: Values for the Video to Audio DTO registers (VADT0, VADT1, VASEL)

6.5.2. Generating the system time clock

STC DTO (registers A9DT0, A9DT1)

The appropriate audio clock connected to the audio clock input pin AUCK is also used for system time clock generation via the system time clock DTO (STC DTO, figure 28).

The ratio between the audio clock and the 90KHz MPEG system time clock STC is again determined by two values C0 and C1. These values which are listed in table 27 are set depending on the needed audio clock. The system time clock is output via the NCLU pin of the FMPEG for external use.

The STC DTO settings should be set by the FMPEG driver software (see Application Notes, see chapter 7.1. on page 63).

Audio to MPEG 90 KHz DTO C0, C1 (A9DT0, A9DT1)

Host access: write A9DT0 address: B087 hex (45191 dec)
A9DT1 address: B088 hex (45192 dec)

Bit No.:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A9DTn	-	-	-	Audio to MPEG 90 KHz DTO (C0) / Audio to MPEG 90 KHz DTO (C1)												

Bit	Function
Bit 12 ... 0	Audio to MPEG 90 KHz DTO C0, C1 A9DT0 (C0) : Value after (power-on) reset: HEX'002D' A9DT1 (C1) : Value after (power-on) reset: HEX'002D' The value C0 of the register A9DT0 and the value C1 of the register A9DT1 are deterrmining the ratio between the frequency of the input clock of the system time clock DTO and the frequency of the system time clock (STC) which is generated by the system time clock DTO. The values are listed in table 27.
Bit 15...13	Reserved

Table 26: Audio to MPEG 90 KHz DTO C0, C1 registers (A9DT0, A9DT1)

Audio clock (LLA)	**System time clock (NCLU)**	**A9DT0 C0**	**A9DT1 C1**
KHz	KHz	(HEX)	(HEX)
8192	90	002D	002D
11289.6	90	0019	01F9
12288	90	001E	001E

Table 27: Values for the system time clock DTO registers (A9DT0, A9DT1)

6.6. FMPEG address areas and registers

Addres sarea hex	Addres sarea	Page
0xxx	Reserved	-
1xxx	Random access DRAM page address	38
2xxx	Micro code download 1 port	33
3xxx	Random access word address	38
4xxx	Micro code download 2 port	33
5xxx	Video data input (VDI)	34
6xxx	Micro code download 3 port	33
7xxx	Audio data input (ADI)	34
8xxx	Micro code download 4 port	33
9xxx	Audio / Video data input (AVI)	35
Axxx	Micro code download 5 port	33
Bxxx	FMPEG Registers (see table 29)	62
Cxxx	Micro code download 6 port	33
Dxxx	Audio PCM output area (APO)	54
Exxx	Reserved	-
Fxxx	Reserved	-

Table 28: FMPEG address areas

Register	Name	Address		Page
		hex	dec	
... Reserved ...				
Video configuration register	VCON	B080	45184	40
Audio format register	AFR	B081	45185	53
Reset interrupt register	RINT	B082	45186	49
Video to Audio DTO C0	VADT0	B084	45188	58
Video to Audio DTO C1	VADT1	B085	45189	58
Video to Audio DTO output select	VASEL	B086	45190	59
Audio to MPEG 90 KHz DTO C0	A9DT0	B087	45191	60
Audio to MPEG 90 KHz DTO C1	A9DT1	B088	45192	60
Chip identification code	CIC	B089	45193	35
... Reserved ...				

Table 29: FMPEG registers (ordered by addresses)

7. Application Notes

7.1. Software

7.1.1. FMPEG Device Library (FMPEGDEV.DLL)

The FMPEG device library provides functions for high-level control of the FMPEG full MPEG1 video and audio decoder chip (SAA7131). The library functions can be divided into three categories:

- Register Access
- Device Functions
- Stream Data Buffer

The register access functions are used to initialize the ISA-bus interface of the FMPEG device and to access FMPEG registers directly.

The device functions provide an abstraction of the FMPEG device for easy access to the FMPEG functionality. These functions control the video decoder and the audio decoder simultaniously and add some functionality not provided by the device which simplify programming of the device.

The stream data buffer functions are used to provide the FMPEG device with MPEG stream data. These functions make it simple to interface the FMPEG device with a CD-ROM. There's no need for the client to implement an interrupt handler, a simple timer function which provides the buffer regularly with data is sufficient.

A list of the available functions per category is given below:

Register Access

- GetVersion
- SetDeviceID
- GetDeviceID
- InitBusInterface
- WriteRegister
- ReadRegister
- WriteDataBlock

Device Functions

- LoadMicroCode
- ResetDevice
- SetVideoSyncInput
- SetVSyncPolarity
- SetHSyncPolarity

- SetVideoOutputFormat
- SetInactiveVideoMode
- SetDisplayMode
- SetUpsamplingMode
- SetPictureOffset
- SetDisplayWindow
- SetViewportWindow
- SetQualifierWindow
- SetBorderColor
- SetVideoOutputEnable
- SetViewportEnable
- SetAudioInputFormat
- SetAudioInputEnable
- SetAudioOutputFormat
- SetAudioOutputEnable
- SetAudioVolume
- SetPlaybackSpeed
- SetFreezeMode
- SetVideoStreamID
- SetAudioStreamID
- StepPictures
- StartPlayback
- StopPlayback
- ResetDecoder
- GetFifoSpace
- WriteMpegData
- SetInterruptEnable
- GetInterruptStatus
- SetFifoSpaceThreshold

Stream Data Buffer

- InstallBuffer
- RemoveBuffer
- GetBufferSpace
- WriteBuffer
- ResetBuffer
- CommandCompleted

Following the functions listed above will be described in detail.

7.1.1.1. Register Access

FmpegDevice::GetVersion

Syntax

void GetVersion(LPWORD *lpwVersion*)

Get driver version number.

Parameters

LPWORD *lpwVersion*

Points to a variable that is filled with the driver version number. The high-byte contains the major version number, the low-byte the minor version number.

FmpegDevice::SetDeviceID

Syntax

void SetDeviceID(WORD *wID*)

Specify device ID to select specific FMPEG device.

Parameters

WORD *wID*

Specify the device ID of the FMPEG device. The ID corresponds to the signal value on the USER pins of the device to be selected.

Notes

Up to four FMPEG devices can be present in the system. The device ID can be used to select a specific FMPEG device. The ID corresponds to the signal values on the USER pins of the device to be selected.

FmpegDevice::GetDeviceID

Syntax

void GetDeviceID(LPWORD *lpwID*)

Get device ID used to select specific FMPEG device.

Parameters

LPWORD *lpwID*

Points to a variable that is filled with the current device ID.

FmpegDevice::InitBusInterface

Syntax

int Initialize(WORD *wAddrPort*, WORD *wDataPort*, BYTE *bIRQLevel*)

Allocate resources used by the driver and initialize the FMPEG device.

Parameters

WORD *wAddrPort*
Specifies a 16-bit address port to be used by the device.

WORD *wDataPort*
Specifies a 16-bit data port to be used by the device.

BYTE *bIRQLevel*
Specifies the IRQ level to be used by the device. Valid IRQ levels are 10, 11, 12, and 15.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function will initialize the FMPEG ISA-bus interface and download the micro-code for the video decoder controller and the audio decoder controller. It will also allocate a stream buffer and install an interrupt handler which transfers data from the stream buffer to the device.

This function must be called first and completed successfully before other functions can be called (except GetVersion).

NOTE. When the FMPEG ISA-bus interface has been configured the device does not respond to subsequent configurations. If the configuration parameters should change, the device has to be turned off before it responds to the new configuration.

FmpegDevice::WriteRegister

Syntax

void WriteRegister(WORD *wAddress*, WORD *wData*)

Write a value to specific FMPEG register.

Parameters

WORD *wAddress*
FMPEG register address.

WORD *wData*
New register value.

Comments

This function should be used only for debugging purposes.

FmpegDevice::ReadRegister

Syntax

WORD ReadRegister(WORD *wAddress*)

Read value from specific FMPEG register.

Parameters

WORD *wAddress*
FMPEG register address.

Return Value

Returns value in specified register.

Comments

All registers can be read since write-only registers are saved into shadow registers.

This function should be used only for debugging purposes.

FmpegDevice::WriteDataBlock

Syntax

void WriteDataBlock(WORD *wAddress*, LPWORD *lpData*, DWORD *cwData*)

Write a block of data to specific FMPEG register.

Parameters

WORD *wAddress*
FMPEG register address.

LPWORD *lpData*.
Points to memory block with word data.

DWORD *cwData*
Number of words to transfer from memory block.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function should be used only for debugging purposes.

7.1.1.2. Device Functions

FmpegDevice::LoadMicroCode

Syntax

int LoadMicroCode()

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function should have completed successfully before other functions can be called (except GetVersion, InitBusInterface, WriteRegister, ReadRegister, and WriteDataBlock).

FmpegDevice::ResetDevice

Syntax

int ResetDevice()

Reset the FMPEG device, i.e. revert to state after initialization.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function resets the decoder and initializes the FMPEG registers to default values.

FmpegDevice::SetHSyncPolarity

Syntax

int SetHSyncPolarity(WORD *wPolarity*)

Specify VGA HSYNC signal input polarity.

Parameters

WORD *wPolarity*

Specifies the signal polarity. The following values can be used:

0 = active low HSYNC (default)
1 = active high HSYNC

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::SetVideoOutputFormat

Syntax

int SetVideoOutputFormat(WORD *wFormat*)

Specify video output format (DMSD,VMC,YUV/RGB).

Parameters

WORD *wFormat*

Specifies the video output format. The following values can be used:

0 = DMSD/DAVE YUV (default)
1 = VMC16 YUV
4 = VMC8 YUV
8 = DMSD/DAVE RGB888
9 = VMC16 RGB888
10 = VMC16 RGB565
11 = VMC16 RGB555
14 = VMC8 RGB565
15 = VMC8 RGB555

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::SetVideoSyncInput

Syntax

int SetVideoSyncInput(WORD *wSyncInput*)

Specify input pins used for video synchronisation signals (DMSD/VGA).

Parameters

WORD *wSyncInput*
Specifies the synchronisation input. One of the following values can be used:

0 = VGA synchronisation inputs (default)
1 = DMSD synchronisation inputs

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

For VGA synchronisation inputs the polarity of the signal can be specified using SetVSyncPolarity and SetHSyncPolarity.

FmpegDevice::SetVSyncPolarity

Syntax

int SetVSyncPolarity(WORD *wPolarity*)

Specify VGA VSYNC signal input polarity.

Parameters

WORD *wPolarity*
Specifies the signal polarity. The following values can be used:

0 = active low VSYNC (default)
1 = active high VSYNC

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::SetInactiveVideoMode

Syntax

int SetInactiveVideoMode(WORD *wMode*)

Specify whether blanking or tree-state outputs should be used outside the display window and inside the display window when video output is disabled.

Parameters

WORD *wMode*

Specifies the mode used for blanking outside the display window. The following values can be used:

0 = use tri-state outputs (default)
1 = use blanking signal

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::SetDisplayMode

Syntax

int SetDisplayMode(WORD *wDisplayMode*)

Specify whether pictures should be displayed in normal size or in doublesize on VGA or TV screen.

Parameters

WORD *wDisplayMode*

Specifies the display mode of the video output signal. One of the following modes can be selected:

0 = VGA normal size (default)
1 = VGA double size (horizontal upsampling, vertical line repeat)
2 = TV half size (interlace on)
3 = TV normal size (horizontal upsampling)

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The SetUpsamplingMode function can be used to choose between horizontal pixel-repeat or horizontal interpolation.

FmpegDevice::SetUpsamplingMode

Syntax

int SetUpsamplingMode(WORD *wUpsamplingMode*)

Specify whether interpolation or pixel repeat should be used for horizontal upsampling.

Parameters

WORD *wUpsamplingMode*

Specifies the mode used for horizontal upsampling. One of the following modes can be selected:

0 = horizontal sample-repeat (default)
1 = horizontal interpolation

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::SetPictureOffset

Syntax

int SetPictureOffset(WORD *wLeft*, WORD *wTop*)

Specify the picture coordinate to be displayed in the top-left corner of the viewport window.

Parameters

WORD wLeft

Specifies the offset from the left side of the picture to the left side of the visible picture area in MPEG pixels.

WORD wTop

Specifies the offset from the top size of the picture to the top size of the visible picture area in MPEG pixels.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The dimensions of the picture area that is shown is equal to the dimensions of the viewport window. If the viewport window is smaller than the picture, the picture area that is visible can be scrolled using this function.

FmpegDevice::SetDisplayWindow

Syntax

int SetDisplayWindow(WORD *wLeft*, WORD *wTop*, WORD *wWidth*, WORD *wHeight*)

Specify the position and size of the display window relative to the leading edge of HSYNC and VSYNC or HREF and VS respectively.

Parameters

WORD *wLeft*
Specifies the offset from the left side of the screen to the left side of the display window in screen pixels.

WORD *wTop*
Specifies the offset from the top side of the screen to the top side of the display window in screen pixels.

WORD *wWidth*
Specifies the horizontal window size in screen pixels.

WORD *wHeight*
Specifies the vertical window size in screen pixels.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

In VGA normal size display mode, one MPEG pixel corresponds to one screen pixel. In VGA-double size display mode, one MPEG pixel corresponds to two screen pixels both horizontally and vertically.

FmpegDevice::SetViewportWindow

Syntax

int SetViewportWindow(WORD *wLeft*, WORD *wTop*, WORD *wWidth*, WORD *wHeight*)

Specify the position and size of the viewport window relative to the display window.

Parameters

WORD *wLeft*
Specifies the offset from the left side of the display window to the left side of the viewport window in screen pixels.

WORD *wTop*
Specifies the offset from the top side of the display window to the top side of the viewport window in screen pixels.

WORD *wWidth*
Specifies the horizontal window size in screen pixels.

WORD *wHeight*
Specifies the vertical window size in screen pixels.

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::SetQualifierWindow

Syntax

int SetQualifierWindow(WORD *wLeft*, WORD *wTop*, WORD *wWidth*, WORD *wHeight*)

Specify the position and size of the pixel qualifier window relative to the leading edge of HSYNC and VSYNC or HREF and VS respectively.

Parameters

WORD *wLeft*
Specifies the offset from the left side of the screen to the left side of the qualifier window in screen pixels.

WORD *wTop*
Specifies the offset from the top side of the screen to the top side of the qualifier window in screen pixels.

WORD *wWidth*
Specifies the horizontal window size in screen pixels.

WORD *wHeight*
Specifies the vertical window size in screen pixels.

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::SetBorderColor

Syntax

int SetBorderColor(BYTE *bRed*, BYTE *bGreen*, BYTE *bBlue*)

Specify the RGB color to be used in the display window not covered by the viewport window.

Parameters

BYTE *bRed*
Specifies border red color value.

BYTE *bGreen*
Specifies border green color value.

BYTE *bBlue*
Specifies border blue color value.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

In YUV color modes the specified RGB color components are automatically converted to YUV color components using the following formulas:

Y = 0.29900 * R + 0.58700 * G + 0.11400 * B

U = - 0.16874 * R - 0.33126 * G + 0.50000 * B + 128

V = 0.50000 * R - 0.41869 * G - 0.08131 * B + 128

Blanking is represented by RGB = 16,16,16 or YUV = 16,128,128.

FmpegDevice::SetVideoOutputEnable

Syntax

int SetVideoOutputEnable(BOOL *fEnable*)

Enable or disable video output.

Parameters

BOOL *fEnable*

The following values can be used:

0 = video output disabled (default)
1 = video output enabled

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The SetInactiveVideoMode function can be used to specify whether a blanking signal or tri-state outputs should be used.

FmpegDevice::SetViewportEnable

Syntax

int SetViewportEnable(BOOL *fEnable*)

Enable (show) or disable (hide) video output inside the viewport window.

Parameters

BOOL *fEnable*

The following values can be used:

0 = video output in viewport disabled (default)
1 = video output in viewport enabled

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

Video is only shown when both video output and viewport have been enabled. Video output can be enabled using the SetVideoOutputEnable function.

FmpegDevice::SetAudioInputFormat

Syntax

int SetAudioInputFormat(WORD *wFormat*, WORD *wAccuracy*)

Specifiy signal format for external audio input pins (CLIN, WSIN, DIN).

Parameters

WORD *wFormat*

Specifies audio signal format. The following values can be used:

0 = Philips I2S (default)
1 = Sony

WORD *wAccuracy*

Specifies audio signal accuracy. The following values can be used:

0 = 16 bits (default)
1 = 18 bits
2 = 20 bits

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The external audio input is mixed with the MPEG audio signal. To enable the external audio input the SetAudioInputEnable function should be used.

FmpegDevice::SetAudioInputEnable

Syntax

int SetAudioInputEnable(BOOL *fEnable*)

Mix external audio input signal with MPEG audio signal.

Parameters

BOOL *fEnable*

The following values can be used:

0 = disable external audio input (default)
1 = enable external audio input

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::SetAudioOutputFormat

Syntax

int SetAudioOutputFormat(WORD *wFormat*, WORD *wAccuracy*)

Specifiy signal format for audio output pins (CLOUT, WSOUT/SYNC, DOUT).

Parameters

WORD *wFormat*

Specifies audio signal format. The following values can be used:

0 = Philips I2S (default)
1 = Sony

WORD wAccuracy

Specifies audio signal accuracy. The following values can be used:

0 = 16 bits (default)
1 = 18 bits
2 = 20 bits

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::SetAudioOutputEnable

Syntax

int SetAudioOutputEnable(BOOL *fEnable*)

Enable/disable (mute) audio output signal.

Parameters

BOOL *fEnable*

The following values can be used:

0 = disable external audio input (default)
1 = enable external audio input

Return Value

Returns zero if successfull or non zero error value otherwise.

EmpegDevice::SetAudioVolume

Syntax

int SetAudioVolume(WORD *wLeftToLeft*, WORD *wRightToLeft*,
WORD *wLeftToRight*, WORD *wRightToRight*)

Set audio volume for different audio paths.

Parameters

WORD *wLeftToLeft*
Specifies left channel to left output volume. Volume level is (0 - wVolume) dB. Zero means minimum volume (-63 dB), 63 means maximum volume (0 dB).

WORD *wRightToLeft*
Specifies right channel to left output volume. Volume level is (0 - wVolume) dB. Zero means minimum volume (-63 dB), 63 means maximum volume (0 dB).

WORD *wLeftToRight*
Specifies left channel to right output volume. Volume level is (0 - wVolume) dB. Zero means minimum volume (-63 dB), 63 means maximum volume (0 dB).

WORD *wRightToRight*
Specifies right channel to right output volume. Volume level is (0 - wVolume) dB. Zero means minimum volume (-63 dB), 63 means maximum volume (0 dB).

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The following volume settings can be used for typical applications:

	wLeftToLeft	wRightToLeft	wLeftToRight	wRightToRight
mono program	63	63	63	63
dual language (default language)	63	0	63	0
dual language (second language)	0	63	0	63
stereo program	63	0	0	63

Table 30: Volume settings for different applications

Note: the values are different from those used inside the device.

FmpegDevice::SetPlaybackSpeed

Syntax

int SetPlaybackSpeed(WORD *wSpeed*)

Parameters

WORD *wSpeed*

Specifies a speed factor from 1 to 15. The playback speed is equal to (speed factor / 8) * nominal speed. Default speed factor is 8.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The speed factor only affects the video decoder. Simultanious decoding of video and audio is not possible with a speed factor other than 8.

FmpegDevice::SetVideoStreamID

Syntax

int SetVideoStreamID(WORD *wStreamID*)

Parameters

WORD *wStream*

Specifies the video stream ID. Values from 0 to 15 can be used.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function should only be used in the stopped state.

Most MPEG files contain only one video stream with ID 0. For VIDEO CD files a stream ID of 1 is normally used for normal resolution still-pictures and a stream ID of 2 for high-resolution still-pictures.

FmpegDevice::SetAudioStreamID

Syntax

int SetAudioStreamID(WORD *wStreamID*)

Parameters

WORD *wStreamID*
Specifies the audio stream ID. Values from 0 to 31 can be used.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function should only be used in the stopped state.

FmpegDevice::SetFreezeMode

Syntax

int SetFreezeMode(BOOL *fFreeze*)

Freeze or unfreeze the frame buffer should be frozen.

Parameters

BOOL *fFreeze*

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function can be used to freeze and or unfreeze the display during playback. The freeze mode is not affected by the Start-Playback or StopPlayback function.

FmpegDevice::StepPictures

Syntax

int StepPictures(WORD *wStepFlags*, WORD *wStepCount*)

Step a number of pictures and display the last picture.

Parameters

WORD *wStepFlags*
Option flags of the step function. (...)

WORD *wStepCount*
Specifies the number of pictures to step. A value of 1 is used to step to the next picture.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function skips wStepCount - 1 pictures as fast as possible. It then decodes and displays one picture. The picture which is displayed doesn't have to be an I-picture.

The application is responsible for filling the stream buffer with data while the device is stepping.

FmpegDevice::StartPlayback

Syntax

int StartPlayback()

Start playback of video and/or audio.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The application is responsible for filling the stream buffer with data while the device is playing.

FmpegDevice::StopPlayback

Syntax

int StopPlayback()

Stop playback of video and/or audio.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The device will enter the paused state and the buffers will remain filled.

FmpegDevice::ResetDecoder

Syntax

int ResetDecoder()

Clear the decoder fifo. The decoder will start decoding when an intra-picture is found in the new data.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function should be called when a jump is made in the MPEG data.

This function is called when the level 3 function ResetBuffer is called.

FmpegDevice::GetFifoSpace

Syntax

int GetFifoSpace(LPDWORD *lpdwFree*)

Get free space in decoder fifo.

Parameters

LPDWORD *lpdwFree*
Points to a DWORD variable that is filled with the fifo size in bytes. The size returned is always an even number of bytes.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The size returned is actually the free size in the video decoder fifo or the free size in the audio decoder fifo, whichever is less. Therefore a fifo overflow should not occur, regardless of the type of data (video or audio).

FmpegDevice::WriteMpegData

Syntax

int WriteMpegData(LPBYTE *lpData*, DWORD *cbData*)

Write MPEG data to the decoder fifo.

Parameters

LPBYTE *lpData*
Points to a memory block of MPEG data.

DWORD *cbData*
The number of bytes in the memory block. This should be an even number of bytes.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This functions scans the data for a MPEG system clock reference. When the StartPlayback function is used, the latest system clock reference, when available, will be used to initialize the system clock of the decoder.

FmpegDevice::SetInterruptEnable

Syntax

int SetInterruptEnable(DWORD *dwFlags*)

Parameters

DWORD *dwFlags*

Specifies the events for which a hardware interrupt should be generated. The following event flags can be used:

FIFO_SPACE_AVAILABLE
COMMAND_COMPLETE
SEQUENCE_HEADER_FOUND
GROUP_OF_PICTURES_FOUND
PICTURE_FOUND
SEQUENCE_END_CODE_FOUND
DECODER_DELAYED_ERROR
VIDEO_STREAM_ERROR
SYSTEM_STREAM_ERROR
AUDIO_STREAM_UPDATE
FRAME_HEADER_FOUND
AUDIO_STREAM_ERROR
AUDIO_SYNC_ERROR

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::GetInterruptStatus

Syntax

int GetInterruptStatus(LPDWORD lpdwFlags)

Get the events which have occurred that resulted in a hardware interrupt and acknowledge the interrupt.

Parameters

LPDWORD *lpdwFlags*

Points to a DWORD variable that will be filled with the interrupt event flags. See the SetInterruptEnable function for a list of the possible flags.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This functions acknowledges the interrupt and clears the device's interrupt status registers.

FmpegDevice::SetFifoSpaceThreshold

Syntax

int SetFifoSpaceThreshold(DWORD *dwThreshold*)

Set the threshold level for the FIFO_SPACE_AVAILABLE interrupt.

Parameters

DWORD *dwThreshold*
Specifies the free fifo space threshold in bytes at which a interrupt should be generated.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

The FMPEG device checks the free space in the video and audio fifo at each VSYNC/VS. An interrupt will be generated if the free space in both fifos is equal or greater than the specified threshold level and the FIFO_SPACE_AVAILABLE interrupt is enabled.

7.1.1.3. Stream Data Buffer

FmpegDevice::InstallBuffer

Syntax

int InstallBuffer(DWORD *dwSize*)

Install interrupt handler and data buffer to supply FMPEG device with MPEG data.

Parameters

DWORD *dwSize*
Specifies the size of the data buffer located In PC memory.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function installs an interrupt handler which handles the FIFO_SPACE_AVAILABLE interrupt and allocates a MPEG data buffer located in PC memory. When this interrupt occurs, data is transferred from the PC buffer to the device.

This function should have completed successfully before other buffer functions can be used (RemoveBuffer, GetBufferSpace, ResetBuffer, WriteBuffer).

FmpegDevice::RemoveBuffer

Syntax

int RemoveBuffer()

Remove host data buffer and interrupt handler.

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::GetBufferSpace

Syntax

int GetBufferSpace(LPDWORD *lpdwFree*)

Get free stream buffer space.

Parameters

LPDWORD *lpdwFree*
Points to a DWORD variable that will be filled with the free buffer space in bytes.

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::WriteBuffer

Syntax

int WriteBuffer(LPBYTE *lpData*, DWORD *cbData*)

Write data to stream buffer.

Parameters

LPBYTE *lpData*
Specifies a pointer to a memory block with data.

DWORD *dwData*
Specifies the number of bytes to write.

Return Value

Returns zero if successfull or non zero error value otherwise.

FmpegDevice::ResetBuffer

Syntax

int ResetBuffer()

Stop the video and audio decoder and clear the stream buffers.

Return Value

Returns zero if successfull or non zero error value otherwise.

Comments

This function should be used after a jump is made in the data source (e.a. file seek) and before new data is written to the stream buffer.

This functions calls ResetDecoder.

FmpegDevice::CommandCompleted

Syntax

BOOL CommandCompleted()

Return true if command completed interrupt has occurred.

Return Value

Returns TRUE if command completed interrupt has occurred and the COMMAND_COMPLETE interrupt status flag is set.

Comments

If an interrupt occurs with the COMMAND_COMPLETE status flag set, this flag is copied to another variable by the library. This variable is not reset by subsequent interrupts which do not have the COMMAND_COMPLETE status flag set. After a call to CommandCompleted() the status flag will be reset.

7.2. ISA Bus Interface

All FMPEG accesses from the host are done via the ISA Bus Interface. It is assumed that external buffers are used for buffering the bidirectional data lines. The special signal line (DOEN, pin 107) is provided to enable the output of the buffers. The direction is switched with the IORC signal which is applied to the IOR pin (pin 107) of the FMPEG.
The outputs IOCS16 (pin 110) and IOCHRDY (pin 111) are implemented as TTL outputs. External tristateable buffers are required for connection to the bus.

7.3. Audio Clock

The FMPEG is able to derive an auidio clock of 256*fs frequency internally from the line locked clock LLC (pin), from the VGA graphics clock connected to VIDEOCLK input (pin) or from another clock source via the ASC input (pin) by using the Audio DTO. Since the ratio between input clock of the Audio DTO and its output on the LLA pin (pin) is nomally not integer the Audio DTO must be able to build this noninteger ratio which leeds to a highfreqency clock jitter on the audio clock. This may cause a reduction of the performance of an audio DA converter which is connected to this clock. THe effect is also dending on DA converter which is used.

This clock jitter is in a range of one clock cycle of the source clock and so is decreasing with using source clocks of higher frequencies.

However it is suggested to reduce the high frequency clock jitter futhermore by using a PLL. In figure 29 of a PLL configuration using a HCT9046 is shown which is able reduce the jitter for all needed audio clock frequencies from 8.192 MHz to 12.288 MHz.

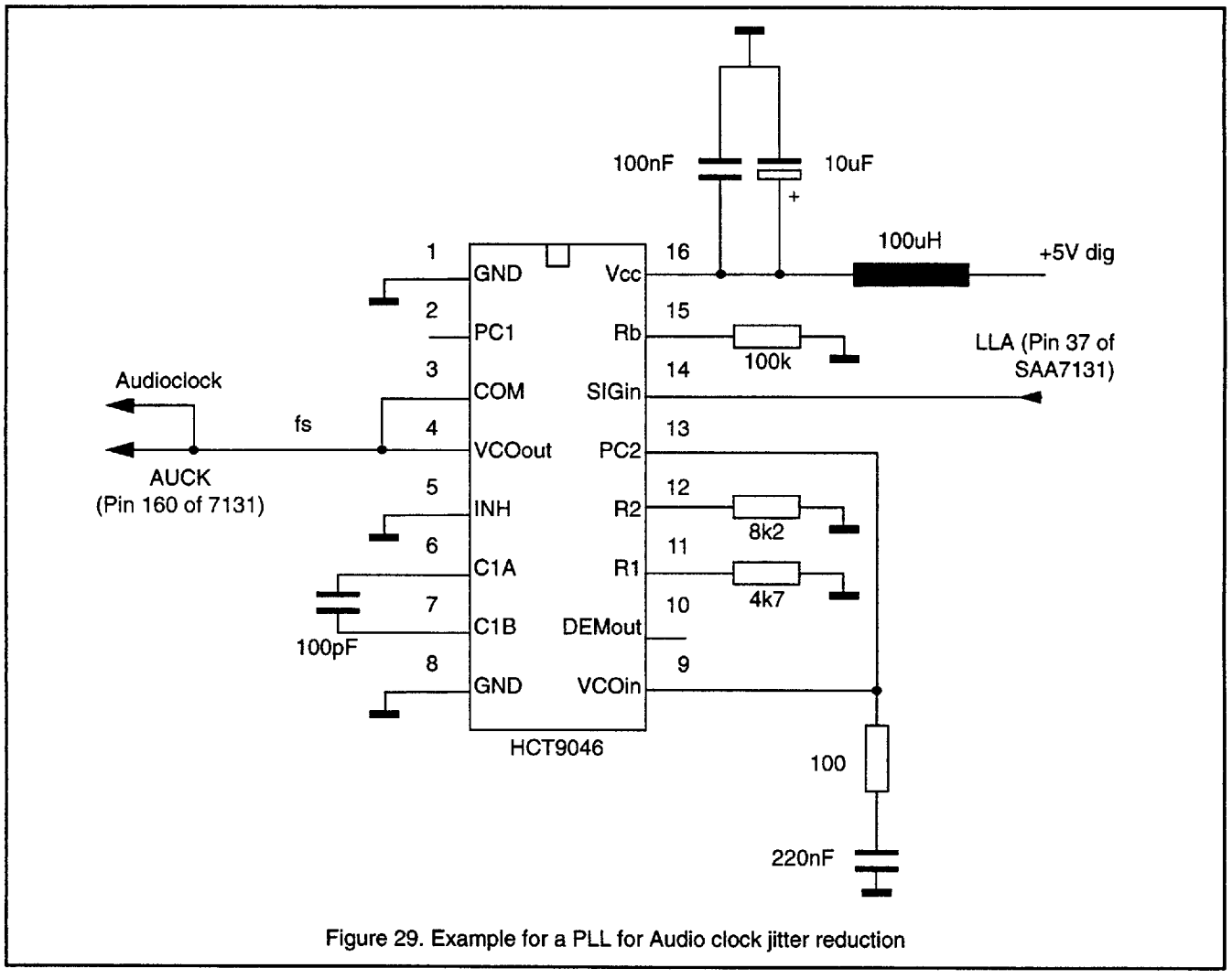


Figure 29. Example for a PLL for Audio clock jitter reduction

7.4. SAA7131A Pinning

The SAA7131A version will operate with a 3.3 Volt power supply on the core. The pinning of this version is based on the SAA7131 except for some power supply pins which need a 3.3 V power supply.

7.4.1. Pin description

Signal	Pin No.	Status	Signal description
ISA BUS Interface			
RSTDEV	81	I	ReSeT DEVice. Active HIGH reset line. Minimum HIGH time 1µsec.
D[15..00]	138 - 141 144 - 149 152 - 157	I, O	Bidirectional databus.
A[11..00]	103 - 95 92 - 90	I	Address lines For IO port addressing only A[09..00] are valid. A[11,10] are for future use to support "plug and play".
AEN	106	I	Address ENable Active HIGH input signal asserted during DMA. When active the FMPEG IC will not evaluate the address.
IORN	108	I	IO Read cycle Active LOW input requesting for data from the addressed IO device.
DOEN	107	O	Data Output ENable Active LOW output signal used, during an IO-read cycle, to enable the external data line buffers.
IOWN	109	I	IO Write cycle Active LOW input indicating that data is valid on the bus for the addressed IO device.
IOCS16N	110	O	16 bits IO device Active LOW output that indicates, during an IO cycle, that the addressed device is an 16 bits IO device.
IOCHRDY	111	O	IO CHannel ReaDY Active HIGH output signal. Driven LOW during an IO cycle when the bus cycle must be lenghtened.
IRQ[15,12,11,10]	114 - 117	O	Interrupt ReQuest Active HIGH output signal that indicates the occurence of certain events.
CNR0	119	I	Chip NumbeR inputs These two bits are part of the FMPEG identifation code that is used for software configuring the IO-port addresses and interrupt level. (see chapter 6.1.1. on page 32)
CNR1	118	I	

Table 31: FMPEGA Pinning

Signal	Pin No.	Status	Signal description
The DRAM interface			
AR[08..00]	53, 54 57 - 62 65	O	9 bits multiplexed row/column address
WEN	67	O	Write Enable Active LOW write strobe.
DR[15..00]	41 - 46 49, 50 68 - 70 73 - 77	I, O	Bi-directional 16 bits databus
CASN	51	O	Column Address Select 1 Active LOW column address strobe for the data (DR[15..00]).
RASN	66	O	Row Address Select Active LOW row address strobe.
Video bus: General signals			
VD[07..00]	10, 9 6 - 1	I, O	Video data output lines [07..00] Bidirectionl data lines. When the outputs are disabled data can be written under control of the WRITE_STROBE signal. For the mapping of the video signals to the output pins see table 15 on page 42, table 16 on page 43 and table 17 on page 43
VD[23..08]	20 - 17 14 - 11 30 - 25 22, 21	O	Video data output lines [23..08] For the mapping of the video signals to the output pins see table 15 on page 42, table 16 on page 43 and table 17 on page 43
PXQ	122	O	PiXel Qualifier An active HIGH signal that flags a predefined area on the screen. The size and the location of this predefined area is programmable.
FE	123	I	Fast Enable input Active HIGH signal. When inactive the video data outputs are made tri-state.
WRITE_STROBE	124	I	Active high signal indicating that a byte on VD[07..00] must be written to an internal register. When the internal register is not yet "emptied" by the HOST, the write action is ignored.
Video bus: DMSD specific signals			
VS	36	I	Vertical Sync Active HIGH. Active during the vertical retrace. When active the video output is blanked.
HREF	34	I	Horizontal REFerence signal Acive LOW. Active during the horizontal retrace. When active the video output is blanked.

Table 31: FMPEGA Pinning

Signal	Pin No.	Status	Signal description
CREF	35	I	Clock REFerence signal Indicates which rising edge of the LLC must be used to change the output pixels.
LLC	33	I	Line Locked video Clock Video timing clock of 2 times the pixel clock.
Video bus: VGA specific signals			
HSYNC	125	I	Horizontal synchronization signal The HSYNC signal has a programmable polarity.
VSYNC	126	I	Vertical synchronization signal The VSYNC signal has a programmable polarity.
VIDEOCLK	129	I	VGA pixel clock
Audio interface: Audio output signals			
CLOUT	135	O	Bit CLock OUTput
WSOUT	136	O	Word Select OUTput
DOUT	137	O	audio Data OUTput
Audio interface: external Audio input signals			
CLIN	131	I	external audio bit CLock INput
WSIN	130	I	external audio Word Select INput
DIN	132	I	external audio Data INput
Clock pins			
SYSCLK0	88	I	System clock oscillator input pins A crystal of 40 MHz must be connected.
SYSCLK1	89	O	
AUCK	160	I	AUdio cloCK input Clock input for the 8.192 MHz, 11.2896 MHz or 12.288 MHz clock. In case of using the internally generated audio clock, which is generated by the audio DTO, this pin is connected to the LLA pin (pin 37).
NCLU	80	O	90 KHz MPEG system clock output
LLA	37	O	audio clock output Digital output of the audio DTO which contains the internally generated audio clock. This pin provides the audio clock, which can be applied to the AUCK pin (pin160).
ASC	38	I	Audio Source Clock input An external clock signal applied to this pin can be used as the input for the audio DTO, which generates the audio clock.

Table 31: FMPEGA Pinning

Signal	Pin No.	Status	Signal description
Test pins			
TEST_MOD	85	I	TEST_MODe (must be connected to ground in normal mode)
TEST_SHT	86	I	TEST SHifT (must be connected to ground in normal mode)
TEST_EN	87	I	TEST ENable (must be connected to ground in normal mode)
TST0	82	I	Test mode pin 0 (must be connected to ground in normal mode)
TST1	83	I	Test mode pin 1 (must be connected to ground in normal mode)
TST2	39	I	Test pin (must be connected to ground in normal mode)
SCW	84	I	Test pin (must be connected to ground in normal mode)
Voltage supply			
VDDDPY	7, 15, 23 47, 63, 78 93, 112 127, 142 150	I	+ 5V pad supply voltage
VSSDPY	8, 16, 24 48, 64, 79 94, 113 128, 143 151	I	pad GND
VDDDCO	55, 71 120, 133 158	I	+ 3.3V core supply voltage
VSSDCO	56, 72, 121, 134 159	I	core GND
VDDDCL	31, 104	I	+ 3.3V clock buffer supply voltage
VSSDCL	32, 105	I	clock buffer GND
Reserved			
Reserved	40, 52	-	open

Table 31: FMPEGA Pinning

7.4.2. Package layout

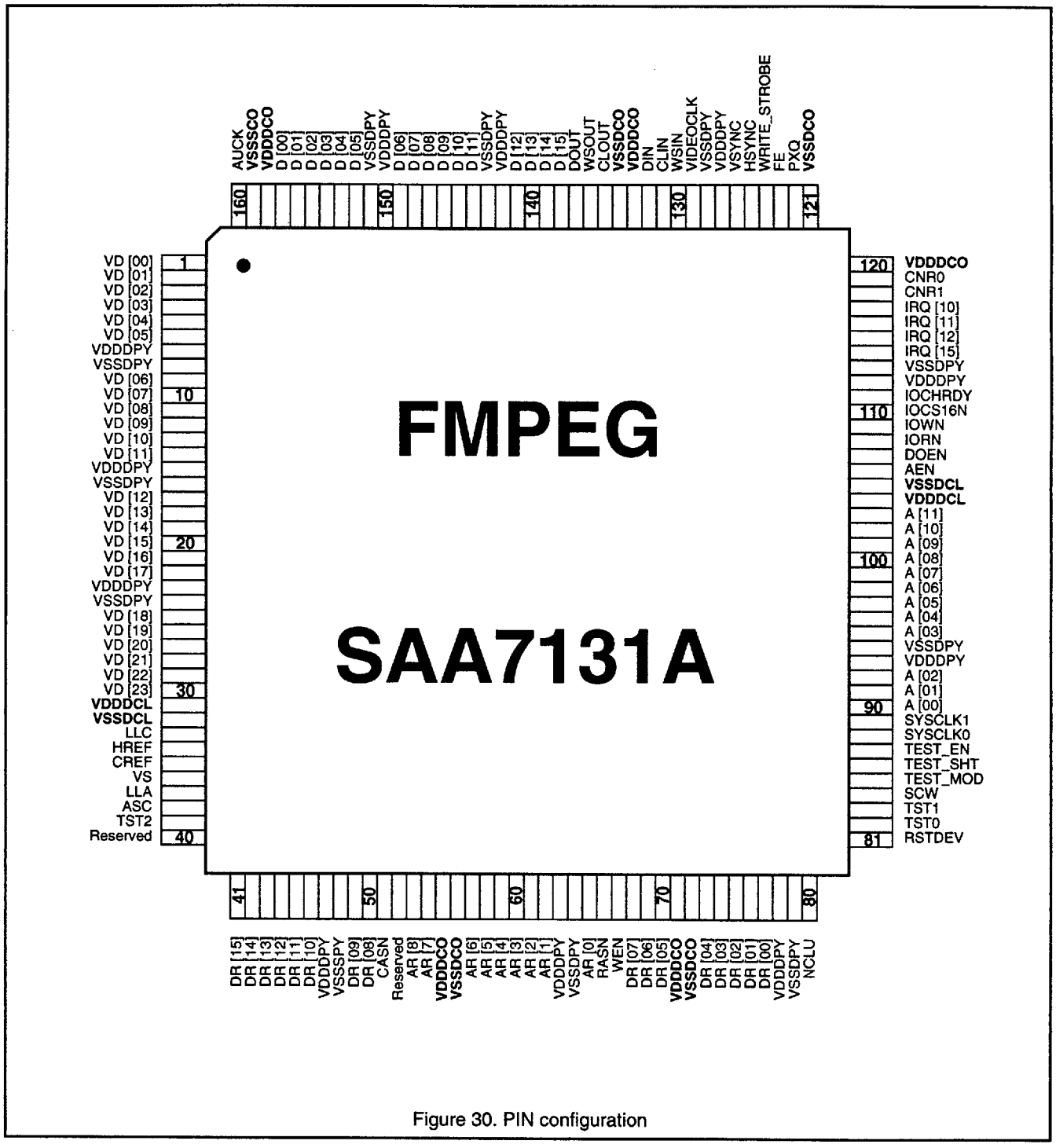


Figure 30. PIN configuration