ARM VIDC20

Data Sheet

Document Number: ARM DDI 0030E
Issued: Feb 1995

Proprietary Notice

Change Log

Issue	Date	By	Change
E	Feb 1995	BJH	Updates to signal descriptions and pins.

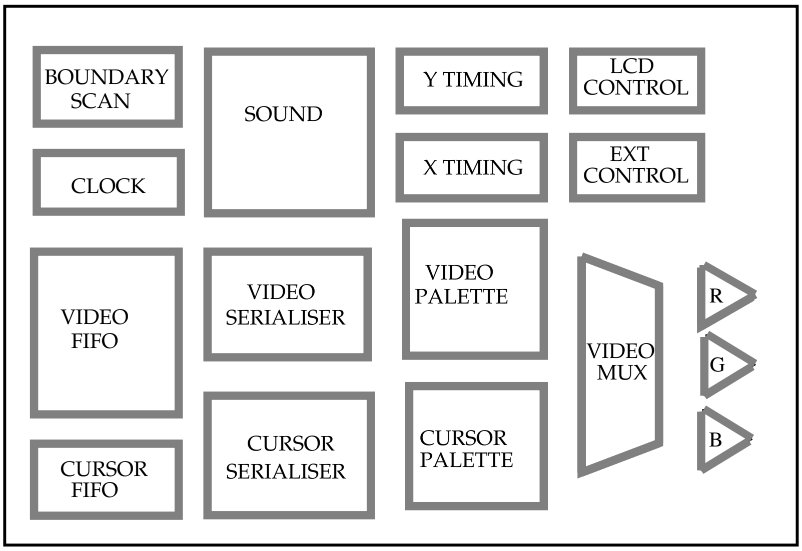
Preface

The VIDC20 is a highly flexible video controller capable of meeting the needs of a wide range of video requirements. The device is highly programmable and can be used in many different system configurations. The device will directly drive colour monitors and also has a LCD driver for direct interfacing of LCD screens for portable computer products. Hi-resolution monochrome monitors may also be driven at pixel rates of up to 400 MHz with minimal external hardware.

The device also supports two stereo sound systems, one featuring up to eight channels each with its own stereo position, and the other, a serial sound port suitable for connection to an external CD DAC.

Applications

- *Desktop computers*
- *Portable computers*
- *PC video cards*
- *Graphics engines*
- *Multimedia*
- *Portable consumer products*



Feature Summary

- *VGA, Super VGA, and XGA resolution*
- *8 bits, giving 16M colours*
- *Direct driving of LCD and CRT screens*
- *1, 2, 4, 8, 16, 32 bits per pixel*
- *1.0 micron CMOS process*
- *Up to 100MHz pixel rate*
- *Low power onsumption*
- *IEEE 1149.1 boundary scan*
- *144 PQFP package*

Table of Contents

1.0 Introduction

The VIDC20 offers the video system designer a flexible, high performance video controller for power and cost sensitive applications. The VIDC20 can be incorporated within a broad family of end products each with varying degrees of video requirements (e.g Portable LCD system through to higher performance Super VGA desktop products).

The VIDC20 flexible bus interface provides hardware support for interfacing to mixed memory systems (e.g. VRAM and DRAM) in conjunction with an external memory controller. The device also incorporates two stereo sound systems - an 8-bit (logarithmic) system, featuring up to eight channels each with its own stereo position, and an serial sound output port suitable for connection to an external CD DAC.

VIDC20 has a 64 bit data bus allowing a high data bandwidth (160 MByte/s) from an external memory system. VIDC20 takes data from these memory banks under DMA control. VIDC20 will however work with a 32 bit data bus, at a correspondingly lower bandwidth. Hence an entry level system is possible containing only one bank of DRAM. For higher resolution displays, VIDC20 can take its video data from VRAM, and on-chip hardware support is provided for this.

1.1 Typical System Configurations

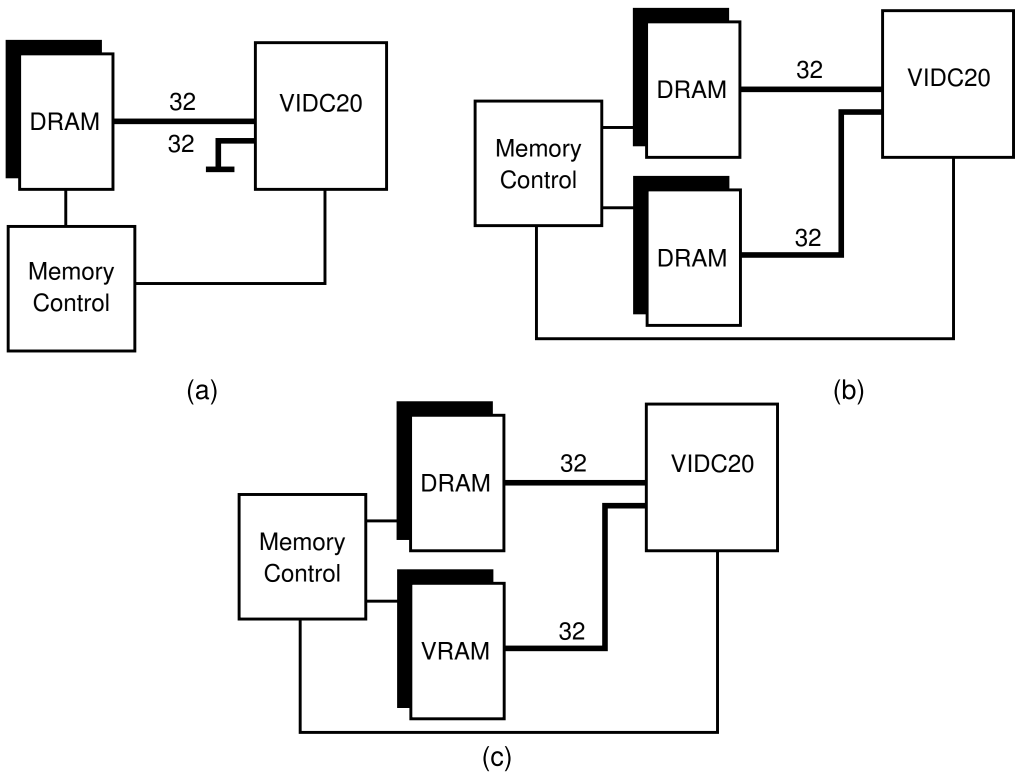


Figure 1: Typical Memory System Configurations

1.2 Major Features

1.2.1 Flexible Video System

VIDC20 contains 296 write-only registers which offer a high degree of flexibility to the system programmer. 256 of these are used as the 28-bit video palette entries. These are programmed via an auto-incrementing address pointer. The remaining registers are specific control registers and allow the user to program the display parameters.

1.2.2 Hardware Cursor

VIDC20 has a hardware cursor for all its display modes - HiRes, Interlace and LCD display modes. By offering cursor support on chip the designer benefits in terms of speed and lower software overhead, resulting in an improved *look and feel*. The cursor is 32 pixels wide and any number of pixels high and can be displayed in 4 colours including transparent from its own 28 bit wide palette. In this way a cursor of any shape and size can be defined within the 32 pixel wide limit.

1.2.3 Palette

VIDC20 has a 28-bit wide 256-entry palette where each entry uses 8-bits for Red, 8 for Green and 8 for Blue, and 4 bits for external data. These external bits may be used outside the chip for a variety of purposes such as supremacy, fading, Hi-Res and LCD driving.

Look Up Tables (LUT) allow for logical to physical translation and gamma correction.The Red Green and Blue LUTs each drive their respective DACs, and the Ext LUT is normally configured to drive the 4 bit output port.

There are three 8-bit linear monotonic DACs (Red, Green and Blue) which give a total of 16 M possible colors. The DACs are designed to operate upto 100 MHz and drive doubly-terminated 75Ω lines directly.

1.2.4 Pixel Clock

VIDC20 is capable of generating a display at any pixel rate up to 100MHz. The pixel clock may be selected from one of 3 sources, and then the selected frequency of this clock may be further divided down by a factor of between 1 and 8.

VIDC20 contains an on-chip phase comparator which, when used in conjunction with an external Voltage Controlled Oscillator (VCO), form a Phase Locked Loop. This configuration allows a single reference clock to generate all the required frequencies for any display mode thus obviating the need for multiple external crystals.

1.2.5 Display Modes

Irrespective of the memory configuration used, VIDC20 is capable of many different display formats. In addition to the normal linear CRT display, VIDC20 can generate a true interlaced display, or can generate a display suitable for either very high resolution displays, single or dual-panel LCDs.

For CRT displays, VIDC20 is capable of operating in a variety of pixel modes - 1,2,4,8,16,32 bits/pixel. VIDC20 can also directly drive LCD displays in 1,2 or 4 bits per pixel via an internal 16-level grey scaler. The grey scaler algorithm adopted is patented.

1.2.6 Power Management

The device is designed for power sensitive applications and incorporates design features to minimise power consumption. Typical power consumption for the device is 0.3 Watts. A *power down mode* allows power savings to be made when the device is not in use - e.g. in conjunction with a battery powered LCD system. Additional power sensitive features include the powering down of functions of the device currently not in use, such as the video DACs, sound DACs and the LCD grey scaler. In addition the palette design has been segmented such that only one eighth of the palette is enabled and clocked at any one time.

1.2.7 On-Chip Sound System

The VIDC20 supports two systems - an 8-bit (logarithmic) system using an internal dedicated DAC, featuring up to eight channels each with its own stereo position, and a 32-bit serial sound output suitable for driving external CD DACs.

In the 8-bit mode the device can work with 1,2,4 or 8 stereo channels, using time division multiplexing to synthesise left and right outputs. The sample rate is programmable through the Sound Frequency Register.

Enhanced 32-bit stereo sound is offered by the serial sound output which consists of a three pin serial interface. Each 32-bits sample consists of 16 bits for the left channel and 16 bits for the right channel.

1.3 Block Diagram

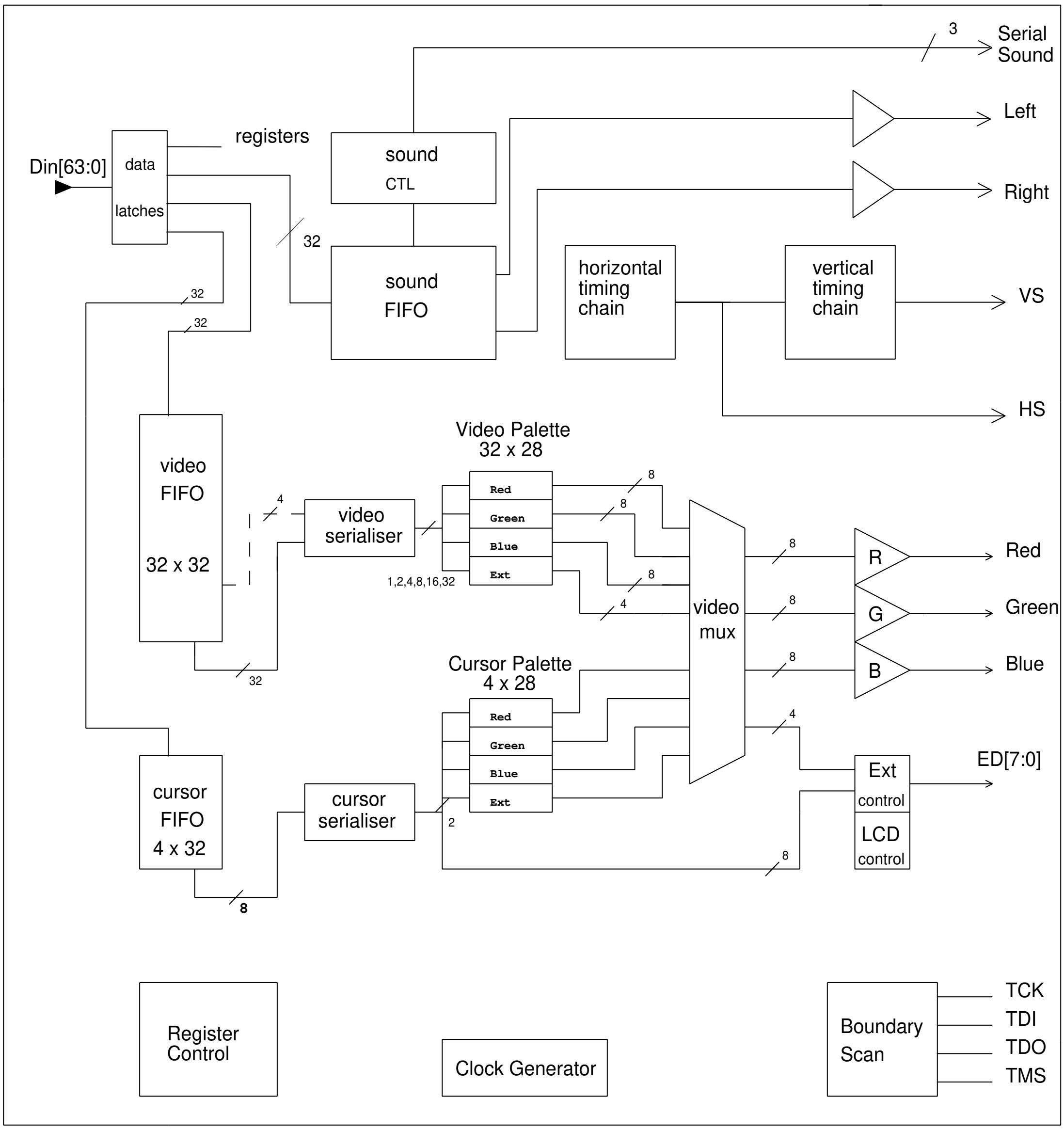


Figure 2: VIDC20 Block Diagram

2.0 Signal Description

Name	Pin	Type	Description
BOUT	39	OA	Blue Analog Output. The video signal analog outputs are designed to drive doubly-terminated 75 lines directly into a CRT.
BUSCLK	123	I	Memory Bus Clock. When configured for a synchronous bus interface, this clock drives both VIDC20 and the memory system. It is used to latch data during sound and video DMAs. BUSCLK must be tied low in async mode.
DIN[63:0]	121-117, 115-103, 101-100, 97-96, 94-88, 86-80, 76-67, 65-60, 58-53, 51-46	I	Data Bus In. All data to the chip is supplied on this bus. Data for register programming is always supplied on the lower half of the bus. When in async 32 bit mode, all data is supplied on the lower half of the bus, and the upper 32 bits should be tied low. When using VRAM, video data only is supplied on the upper half of the bus, and all other data on the lower half. When in split bank mode, interleaved DRAM provides 64 bit data during DMAs. See *Chapter 3.0 System Configurations* for details on architectures.
ECLK	22	O16	External Clock. When enabled, this clock validates the data on **ED[7:0]**. In normal video mode, it runs at the pixel rate, but when LCD data is being produced, it runs at a quarter of the pixel rate.
ED[7:0]	12-14, 16-20	O8	External Data. This is the digital output port of the chip. From this, the digital equivalent of the analog output may be produced in any colour, or data from the external palette may be produced. This may be used for a variety of purposes such as fading, supremacy, or serialisation for driving high resolution monitors. Also, data for driving LCD panels is output from this port. Data produced is validated by **ECLK**.
EREG[1:0]	5,6	OS8	External Register Data. The data from these pins is bits 1 and 0 of the external register. They may be used to control external devices.
ESEL[1:0]	7-9	I	External Data Select. These two bits determine the external port output. This may be either digital red, green or blue data, external data, or LCD data.
FLYBK	1	OS8	Fly Back. This gives information about the vertical display. It goes high at the start of the first raster not in display, and goes low again on the first raster in the active display. Frame buffer updates should be made during the fly back period for a flicker free display.
GOUT	40	OA	Green Analog Output. The video signal analog outputs are designed to drive doubly-terminated 75 lines directly into a CRT.
HCLK	125	I	High speed Clock.See *Chapter 5.0 Pixel Clock* for details of the clocks.

Table 1: Signal Description

Name	Pin	Type	Description
HSYNC	31	OS	Horizontal Synchronisation. There are two synchronisation outputs on VIDC20, **HSYNC** and **VSYNC**. Dependant on the state of bits 17 and 16 in the external register, either horizontal or a composite (NOR) sync may be output on this pin, in either polarity. The width of the **HSYNC** pulse is definable in units of 2 pixels.
LS	27	OA	Left Analog Sound. Analog left hand stereo channel sound output.
nPROG	140	I	NOT Program. When this signal is low, data from **DIN[32:0]** is written to one of the registers. The register address is on the upper bits of the bus, and the data is on the lower bits of the bus.
nQCLK	132	OS8	NOT VRAM Clock. When in VRAM mode, this clock is output when data is required for the video FIFO. Externally, **nQCLK** must be inverted to make **QCLK** and fed back into VIDC20. It is the inverted version, **QCLK** that actually clocks the VRAM and thus validates the data. This scheme accounts for skew from clock loading.
nRESET	143	I	NOT Reset. This is a level sensitive input signal which must be asserted during power up. The chip is forced into power down mode (bit 14 of the Control Register is set high), and thus a clock must be left running.
nSNDAK	138	I	NOT Sound Acknowledge. This signal goes low to indicate that a sound DMA is taking place. Like its video counterpart, it has two timings dependant on the bus interface. Note that a sound DMA request is9 a request for 4 words of data.
nSNDRQ	135	OS8	NOT Sound Request. This signal goes low to initiate a sound DMA cycle. It is driven high again after **nSNDAK** falls.
nVIDAK	139	I	NOT Video Acknowledge. This signal goes low to indicate that video or cursor DMA is taking place. This signal has two timings, dependent on whether the bus interface is synchronous or asynchronous. In the synchronous case, **nVIDAK** is held low throughout the DMA, and data is clocked into the FIFO by **BUSCLK**. In the asynchronous case, data is strobed into the FIFO on the falling edge of **nVIDAK**. DMA is a request for 4 words of data so **nVIDAK** must be driven low 4 times in the asynchronous case.
nVIDRQ	136	OS8	NOT Video Request. This signal goes low to initiate video DMA. If **VnC** is high, then the data transferred will be video data, otherwise it will be cursor. **nVIDRQ** is driven high again after **nVIDAK** falls.
PCOMP	130	O4	Phase Comparator Output.See *Chapter 5.0 Pixel Clock.*
QCLK	133	I	VRAM Clock. See **nQCLK**.
QSF	142	I	Transfer Status Flag. This is an input from VRAM. When this signal changes state, it indicates that a VRAM transfer cycle can take place. This causes VIDC20 to start a video DMA which must be externally interpreted as a transfer cycle. See *Chapter 3.0 System Configurations* and *Applications Note 15* for details of connecting VRAM to VIDC20.

Table 1: Signal Description

Name	Pin	Type	Description
RCLK	124	I	Reference Clock. See *Chapter 5.0 Pixel Clock* for details of the clocks into VIDC20. This clock may also drive the sound system.
ROUT	41	OA	Red Analog Output. The video signal analog outputs are designed to drive doubly-terminated 75 lines directly into a CRT.
RS	28	OA	Right Analog Sound. Analog right hand stereo channel sound output.
SCLK	144	I	Sound Clock. This signal can be used to clock the sound system, when a clock asynchronous to the video system is required. See *Chapter 13.0 Sound.*
SDCLK	3	OS8	Serial Data Clock. When the sound system is in serial sound mode, this clock is output and validates serial data on its rising edge.
SDO/MUTE	4	OS8	Serial Data Out / Mute. This pin has two functions depending on whether the sound mode is either analog, VIDC10 compatible, or digital serial sound. In digital mode, serial sound data is output from this pin. In analog mode, this signal goes high between samples to allow for DAC settling.
SINK	141	I	External Sink. This signal is used to synchronise VIDC20 with another video system. When high, is set to zero, and in an interlaced display system, the odd field is selected. The horizontal timer system is unaffected by this signal.
SIREF	25	IA	Sound Reference Current. A reference current must be fed into this pin in order to calibrate the sound DAC outputs. For most applications, a resistor to **VDD** is sufficient, although a constant current source is recommended.
TCK	33	IP	Test Clock. This is the boundary scan clock. There is an internal pullup resistor to **VDD**, so this should normally be left unconnected.
TDI	35	IP	Test Data In. Data for boundary scan testing is input to the chip on this pin. There is an internal pullup resistor to **VDD**, which should be left unconnected in normal mode.
TDO	36	OS8	Test Data Out. Output data from boundary scan testing.
TMS	34	IP	Test mode select. An input to the boundary scan logic, which should be left unconnected in normal mode since it has an internal pullup resistor to **VDD**.
VCLKI	126	I	Phase Comparator Clock In. See *Chapter 5.0 Pixel Clock.*
VCLKO	127	OS8	Phase Comparator Clock Out.See *Chapter 5.0 Pixel Clock.*
VDD	8,10,21,23, 44,45,59,78, 87,98,102, 116,129,131	P	Positive (+5V) Supply.
VDD_Analog	38	P	Positive (+5V) supply for analog video system.
VDD_Sound	26	P	Positive (+5V) supply for analog sound system.

Table 1: Signal Description

Name	Pin	Type	Description
VIREF	37	IA	Video Reference Current. The video DACs need a reference current in order to calibrate them. A constant current source is recommended, although a resistor up to **VDD** is sufficient for many applications.
VnC	134	OS8	Video NOT Cursor. This signal is used by the external memory controller to differentiate between video and cursor DMA requests.
VSS	11,15,24,30, 43,52,66,77, 79,95,99, 122,128,137	P	Supply Ground.
VSS_Analog	42	P	Supply ground for analog video system.
VSS_Sound	29	P	Supply ground for analog sound system.
VSYNC	32	OS16	Vertical Synchronisation. Dependant on the state of bits 19 and 18 in the external register, either vertical or a composite (XNOR) sync may be output on this pin, in either polarity. The width of the **VSYNC** pulse may be defined in units of a raster.
WS/LnR	2	OS8	Word Select / Left NOT Right. Again, this pin has two functions depending on whether the sound mode is either analog or digital. In digital mode, this signal denotes whether the output serial data is for the left hand stereo channel (ws=0) or the right hand channel. In analog mode this signal gives the same stereo direction information, but the polarity is reversed.

Table 1: Signal Description

2.1 Key to Signal Types

I	Input (TTL threshold)	**IP**	Input (TTL threshold) with pull up resistor
IA	Input Analog	**O4**	Output (4mA Drive)
O8	Output (8mA Drive)	**O16**	Output (16mA Drive)
OS8	Output (8mA slew-limited drive)	**OS16**	Output (16mA slew-limited drive)
OA	Output Analog	**P**	Power supply

3.0 System Configurations

VIDC20 has a 64 bit data bus and there are four basic modes of bus operation.These are described below, with ARM memory controller examples. VIDC20 can be used with any memory controller which supports quad-word DMA -e.g. ARM's MEMC10.

3.1 Asynchronous 32-bit mode

This is the simplest mode, and in this configuration VIDC20 behaves almost exactly like VIDC10. Only the lower 32 bits of the data bus are used, and the upper 32 bits, though ignored, must be tied low.

VIDC20 makes requests for data to MEMC10 via the **nVIDRQ** line, and MEMC10 supplies the address to the single bank of DRAM of the next four words to be used. The four words of data come from the DRAM into the lower 32 data bits of VIDC20 and from there directly into the video FIFO within VIDC20.

Similarly, VIDC20 requests data for the hardware cursor sprite when it is required (during the horizontal sync time), and this is supplied as four words out of a different area of the DRAM. This data passes into the same lower 32 data bits of VIDC20 and from there into the cursor FIFO within VIDC20. Finally, the registers within VIDC20 must be programmed. This is achieved by a processor write of the data into VIDC20 register addresses. Again the data is passed to VIDC20 on the lower 32 data bits. MEMC10 has an asynchronous DMA interface (i.e. the synchronisation is carried out within MEMC10). To configure VIDC20 for this mode, the **BUSCLK** input must be tied permanently LOW, the SnA bit in the Data Control Register set LOW, and BUS[1:0] in the Data Control Register programmed to value 01.

The frame buffer consists of a linear image in the DRAM. The start and end addresses of the buffer are programmable in MEMC10. The cursor buffer also consists of a (much smaller) linear image in the DRAM. The start address of this buffer is also programmable in MEMC10.

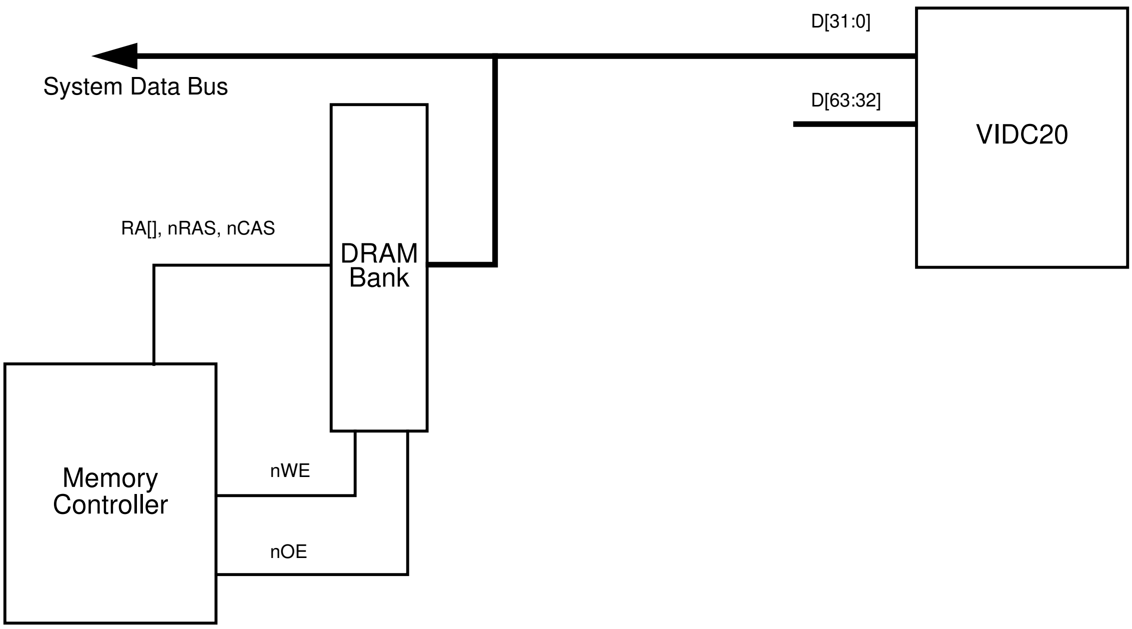


Figure 3: VIDC20 connected in Asynchronous 32 bit mode

3.2 Synchronous 32-bit mode

This mode can be identical to the above mode, with the memory controller and VIDC20 both programmed to have an asynchronous interface like MEMC10 (SnA = 0), or the memory controller and VIDC20 can be configured to have a synchronous interface (SnA = 1). The latter is recommended, as it is more efficient and permits interlace to function correctly. Except for programming of the SnA bit, this mode is identical to the MEMC10 mode described above and so BUS[1:0] is programmed to value 01 again. This mode must be configured if dual-panel LCDs are to be used.

3.3 64-bit mode

This is the standard configuration, and is an extension of the above mode. Again VIDC20 makes requests for data to the memory controller via the **nVIDRQ** line, but now the memory controller supplies the address to two banks of DRAM of the next eight words to be used. The four double-words of data come from both DRAM banks into the 64 data bits of VIDC20 and from there into the video FIFO within VIDC20 as 8 words. The two banks of DRAM are separated by a set of bidirectional buffers, and a simple PAL is required to control the DRAM enable lines and the buffers.

In this mode both the memory controller and VIDC20 must be programmed to have a synchronous interface (SnA = 1). VIDC20 is configured by setting BUS[1:0] to value 11 in the Data Control Register.

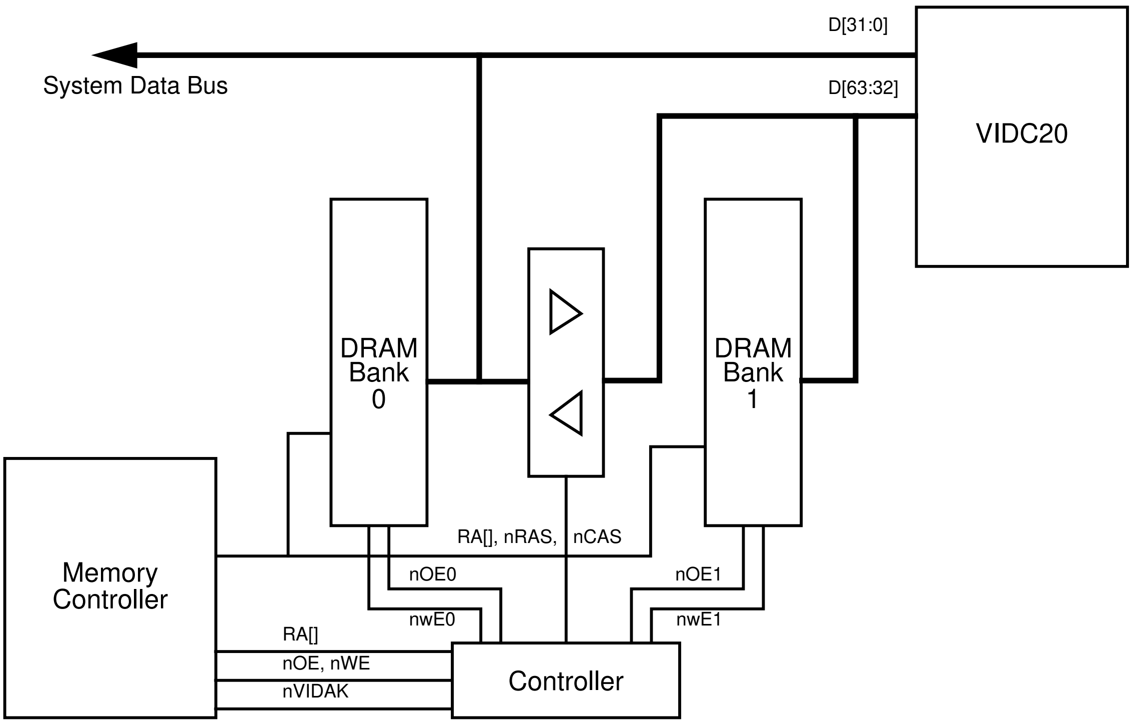


Figure 4: VIDC20 connected in 64 bit mode

3.4 Split Bank mode

This configuration, is similar to the above, except that in this mode VIDC20 always receives video data on the upper 32 data bits only, and it receives cursor and programming data on the lower 32 bits only. Hence in this configuration the cursor and programming data come from one bank of DRAM, and the video data comes from a separate bank of RAM, which may be DRAM or VRAM. This mode is primarily intended for use with VRAM, and more details are given in the next section. If DRAM is used, then a buffer would be needed to separate the buses, and a simple PAL to control them. For split bank mode, VIDC20 is configured by setting BUS[1:0] to value 10, and SnA = 1.

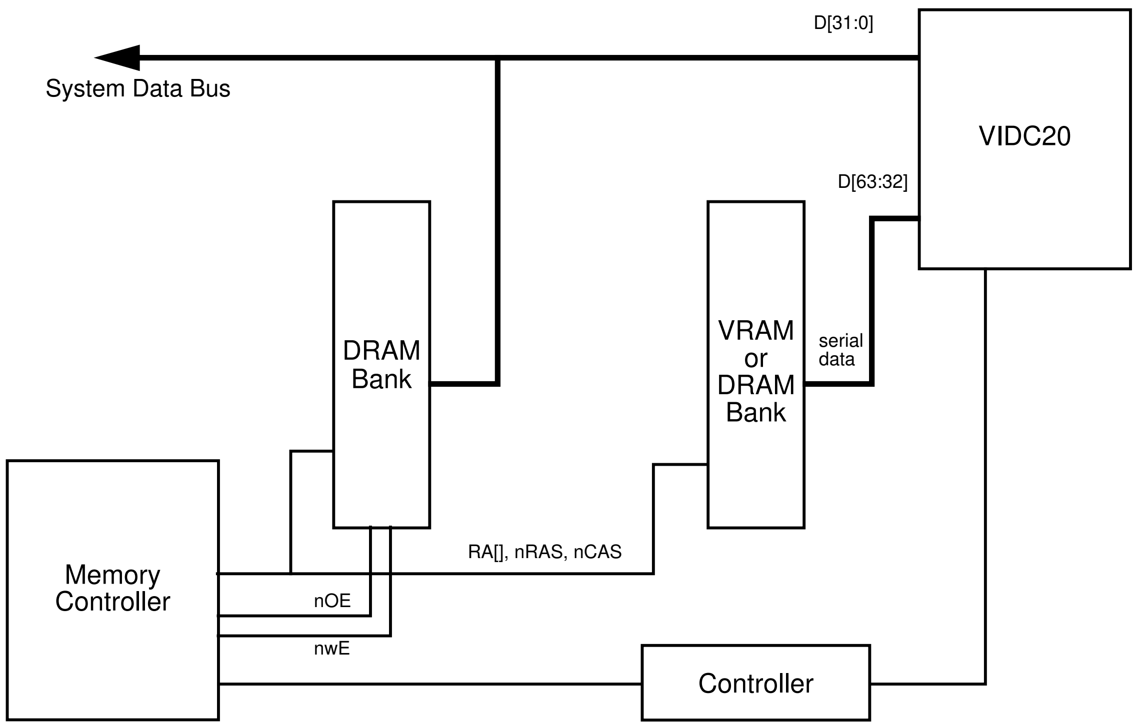


Figure 5: VIDC20 connected in Split Bank mode

3.5 Using VRAM with VIDC20

Where very high bandwidth displays are required, VIDC20 should be programmed into split bank mode, and split serial port VRAM used for the frame store. The advantage of this configuration is that although there is a very high transfer rate from memory to VIDC20, the system bus is not used. Split serial port VRAM must be used to ensure that video data transfers are not interrupted whilst transfer cycles take place.

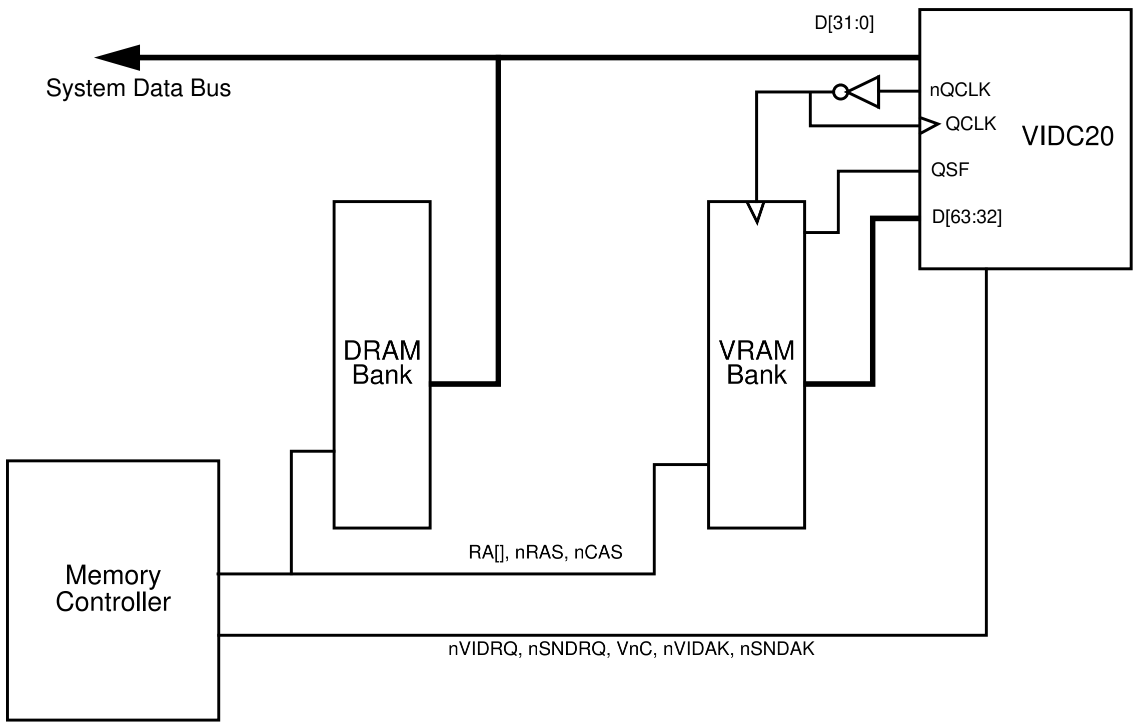


Figure 6: Using VRAM with VIDC20

A brief description of the interface is given here. For more details, please refer to *Applications Note A015* "VIDC20 *VRAM Interface*", available from Advanced RISC Machines.

The video data transfer works as follows. When the video FIFO within VIDC20 is ready to accept more data, the **nQCLK** output is enabled, and this clocks the VRAM SAM register four times. The resulting four words of data are then read into the video FIFO. When all of one half of the SAM register has been read, the VRAM's QSF output changes state. This edge is used by VIDC20 to generate nVIDRQ. This goes straight to the memory controller as normal, but generates a VRAM transfer cycle.

VIDC20's **nQCLK** output is derived from the pixel clock. Bits [7:0] of the Control Register are used to select the pixel clock frequency, and bits [17:16] of the Data Control Register select what division of this frequency is output as **nQCLK**. The frequency of **nQCLK** should be chosen such that the data rate into the video FIFO is greater than or equal to the data rate out of the FIFO.

For example, a system running with a pixel rate of 80MHz at 8 bits per pixel (bps) consumes data at a rate of 80MByte/s. Therefore, the minimum frequency that the SAM register should be clocked at is 20MHz, (Data Control Register [17:16] = 11). Under these conditions, there would be a continuous transfer of data from VRAM to VIDC20. If a higher **nQCLK** frequency were chosen, then the transfer would become more 'bursty', like a normal DMA. It is important to note here that when video DMAs are happening continuously, sound DMAs and programming can still occur. This is because activity on the upper and lower buses is totally independent.

In general, the algorithm for the minimum **nQCLK** frequency is;

nQCLK = Pixel_frequency x bpp / 32

and the data control register is then programmed accordingly.

Due to the high load on **nQCLK**, it must be externally buffered and inverted before driving the VRAMs. The output from the buffer must be fed back into VIDC20 on the **QCLK** input. This ensures that, at the high frequencies involved, clock skew does not effect the validity of data sampled by the chip.

3.5.1 Dual Banks of VRAM

This is an extension of the above mode, where two interleaved banks of VRAM are used. This allows even higher bandwidth displays (e.g. 80MHz pixel rate at 16bps) to be used. The **nQCLK** output from VIDC20 should then go to an external divide-by-two circuit. The resulting positive and negative clock signals are then used to clock the SAM registers of the two banks of VRAM. For example, a system with a pixel rate of 80MHz and 16bps requires data at 160MByte/s. Therefore, if **nQCLK** is programmed to half the pixel rate, 40MHz, and externally this is halved to produce positive and negative clocks, then each bank of VRAM produces data at 20MWord/s, or 80MByte/s, giving the required total of 160MByte/s.

3.6 Display Options

Irrespective of the above configuration used, VIDC20 is capable of many different display formats. In addition to the normal linear CRT display, VIDC20 can generate a true interlaced display, or can generate a display suitable for either single or dual-panel LCDs.

3.6.1 Interlaced Display

When the memory controller is used, interlaced displays may be generated where the image created in the video buffer is the same irrespective of whether interlace is turned on or not. When interlace is turned on, the address generator displays every other raster in the video memory in each field. Two fields make up a complete frame.

3.6.2 LCDs

This is described in more detail later, but the dual-panel LCD option allows VIDC20 to output 2 data streams simultaneously, with one stream for each panel of a dual-panel LCD. The image created in the video buffer is the same irrespective of whether this mode is selected or not. This mode can be programmed to generate 1,2, or 4 bits/pixel grey-level displays. The memory controller is required, and the 32 bit data bus mode must be selected when using the dual-panel (duplex) mode. This restriction is imposed by the memory controller.

4.0 Programming Model

4.1 VIDC20 Registers

Apart from the video and cursor FIFOs, VIDC20 contains 296 write-only registers. These split into 2 categories; the 256 28-bit video palette entries, and all the rest. The video palette entries are written via an auto-incrementing address pointer. All the other registers (including the 28-bit cursor palette) are written directly with the address encoded in the top 4 or 8 bits of the data word.

In order to define the display format correctly, eleven registers need to be programmed as shown in the diagram below:

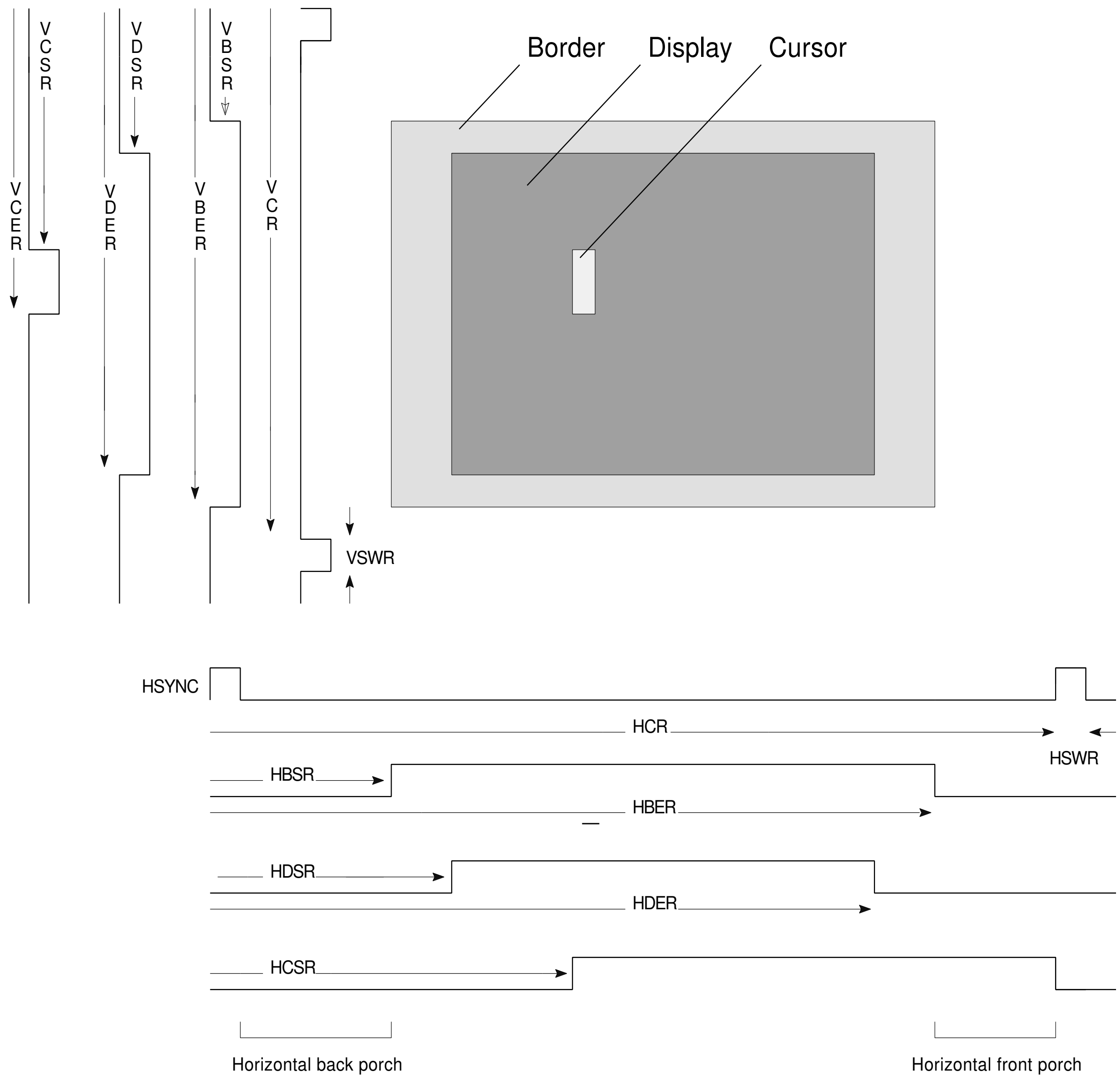


Figure 7: VIDC20 display format definitions

The register allocation is shown in *Table 2: VIDC20 register allocation*. An x denotes the actual data field, and any unused bit should be programmed with a logic zero. Do not access any register at any location other than that shown as the actual register map is multiple-mapped.

The External Register, Control Register, Sound Control Register and Data Control Register all contain bits that are not initialised at power up, and so must be programmed before VIDC20 will operate correctly.

Address(hex)	Register
0xxxxxxx	Video Palette
100000xx	Video Palette Address Register
20000000	RESERVED
300000xx	LCD Offset register 0
310000xx	LCD offset register 1
4xxxxxxx	Border Colour Register
5xxxxxxx	Cursor Palette logical colour 1
6xxxxxxx	Cursor Palette logical colour 2
7xxxxxxx	Cursor Palette logical colour 3
8000xxxx	Horizontal Cycle Register
8100xxxx	Horizontal Sync Width Register
8200xxxx	Horizontal Border Start Register
8300xxxx	Horizontal Display Start Register
8400xxxx	Horizontal Display End Register
8500xxxx	Horizontal Border End Register
8600xxxx	Horizontal Cursor Start Register
8700xxxx	Horizontal Interlace Register
8800xxxx	Test Register (write Hcount)
8C00xxxx	Test Register (write H-all)
9000xxxx	Vertical Cycle Register
9100xxxx	Vertical Sync Width Register
9200xxxx	Vertical Border Start Register
9300xxxx	Vertical Display Start Register
9400xxxx	Vertical Display End Register

Table 2: VIDC20 register allocation

Address(hex)	Register
9500xxxx	Vertical Border End Register
9600xxxx	Vertical Cursor Start Register
9700xxxx	Vertical Cursor End Register
9800xxxx	Test Register (write Vcount)
9A00xxxx	Test Register (increment Vcount)
9C00xxxx	Test Register (write V-all)
A000000x}	Stereo Image Registers
A700000x}	
B00000x	Sound Frequency Generator
B10000x	Sound Control Register
C00xxxxx	External Register
D000xxxx	Frequency Synthesis Register
E00xxxxx	Control Register
F000xxxx	Data Control Register

Table 2: VIDC20 register allocation

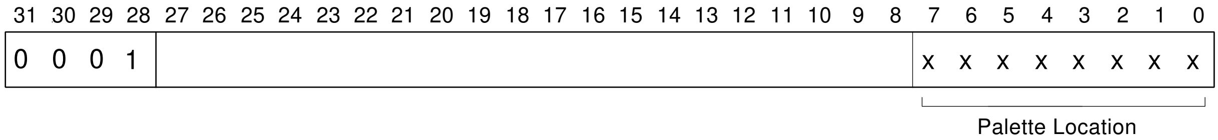
4.1.1 Video Palette: Address 0H

All entries of the video palette are written at address 0. In order to write any or all of the palette locations, the address pointer must first be written, as described below. The palette is programmed with a 28 bit word representing the physical data field.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	E	E	E	E	B	B	B	B	B	B	B	B	G	G	G	G	G	G	G	G	R	R	R	R	R	R	R	R
				Ext Physical Colour				Blue Physical Colour								Green Physical Colour								Red Physical Colour							

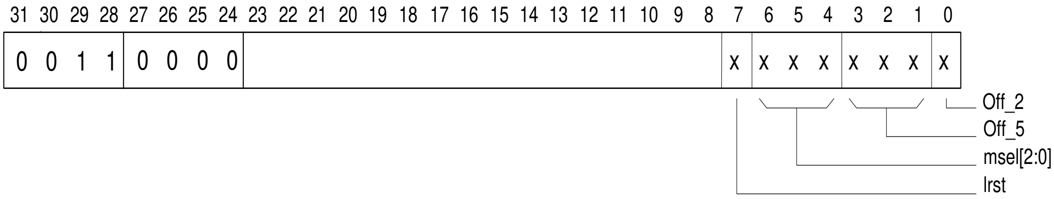
4.1.2 Video Palette Address Pointer: Address 1H

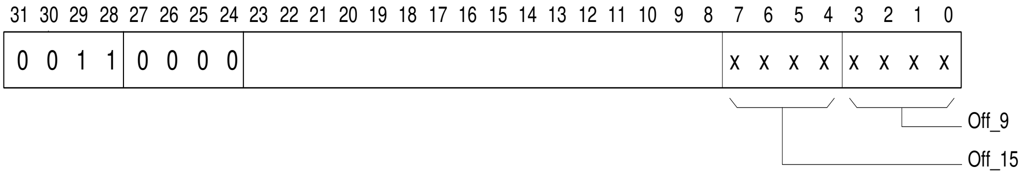
The address pointer is programmed at address 1, and it may be programmed to any value from 0 to 255. The first write to the palette will then occur at this location, and the address pointer will post-increment so that the next palette write will occur to the following location. The counter will wrap around from 255 to 0. Once the address pointer has been written, any number of palette locations can be programmed, and the pointer can be reprogrammed at any time if only part of the whole palette is to be updated.



4.1.3 LCD Offset Registers: Addresses 30H and 31H

These two, eight bit registers define the offsets required for driving a dual panel LCD screen. Register 0 defines the offsets for the five and two frame duty cycle grey scales, as well as reset and test mode bits. Register 1 defines the offsets for the nine and fifteen frame duty cycle grey scales.





The registers values are dependant upon the size of the LCD screen to be driven, and are calculated in the following way:

Off_15 = (3xL + 8) mod 15
Off_9 = (7xL + 4) mod 9
Off_5 = (1xL + 3) mod 5
Off_2 = 0

Where L is the number of lines in the upper panel of the dual panel LCD screen.

Bits 7-4 of register 0 are only used in test mode, and must all be set to zero in normal operation.

msel[2:0] are test bits and should be programmed low.

4.1.4 Border Colour Register: Address 4H

This register defines the physical border colour, and is programmed with a 28 bit word. Note that this register is programmed directly, independent of the value of the video palette address pointer.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	E	E	E	E	B	B	B	B	B	B	B	B	G	G	G	G	G	G	G	G	R	R	R	R	R	R	R	R
				Ext Physical Colour				Blue Physical Colour								Green Physical Colour								Red Physical Colour							

4.1.5 Cursor Palette: Addresses 5H-7H

These three registers are programmed with the physical colour of the three logical cursor colours. Note that cursor logical colour 00 is defined as being transparent (i.e. no cursor display), and this location is used for the Border Colour Register above.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	x	x	E	E	E	E	B	B	B	B	B	B	B	B	G	G	G	G	G	G	G	G	R	R	R	R	R	R	R	R
		Logical Colour		Ext Physical Colour				Blue Physical Colour								Green Physical Colour								Red Physical Colour							

4.1.6 Horizontal Cycle Register (HCR): Address 80H

This register defines the period, in pixels, of the horizontal scan. i.e. display time + retrace time.

This is a 14 bit register of which the bottom 2 bits must be programmed to 0. If N pixels are required in the horizontal scan period, then value (N-8) should be programmed into the HCR. (N must be a multiple of 4).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	0											x	x	x	x	x	x	x	x	x	x	x	x	0	0
																		HCR value													

4.1.7 Horizontal Sync Width Register (HSWR): Address 81H

This register defines the period, in pixels, of the **HSYNC** pulse.

This is a 14 bit register of which the bottom bit must be programmed to 0. If N pixels are required in the **HSYNC** pulse, then value (N-8) should be programmed into the HSWR. (N must be a multiple of 2).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	1											x	x	x	x	x	x	x	x	x	x	x	x	x	0
																		HSWR value													

4.1.8 Horizontal Border Start Register (HBSR): Address 82H

This register defines the time, in pixels, from the start of the **HSYNC** pulse to the start of the border display.

This is a 14 bit register of which the bottom bit must be programmed to 0. If N pixels are required in this time, then value (N-12) should be programmed into the HBSR. (N must be a multiple of 2).

Note that this register must always be programmed, even when a border is not required. If a border is not required, then the value in the HBSR must be such as to start the border in the same place as the display start. i.e. $N_{HBSR} = N_{HDSR}$.

31 30 29 28	27 26 25 24	23 22 21 20 19 18 17 16 15 14	13 12 11 10 9 8 7 6 5 4 3 2 1 0
1 0 0 0	0 0 1 0		x x x x x x x x x x x x x 0
			HBSR value

4.1.9 Horizontal Display Start Register (HDSR): Address 83H

This register defines the time, in pixels, from the start of the **HSYNC** pulse to the start of the video display.

This is a 14 bit register of which the bottom bit must be programmed to 0. If N pixels are required in this time, then value (N-18) should be programmed into the HBSR. (N must be a multiple of 2).

31 30 29 28	27 26 25 24	23 22 21 20 19 18 17 16 15 14	13 12 11 10 9 8 7 6 5 4 3 2 1 0
1 0 0 0	0 0 1 1		x x x x x x x x x x x x x 0
			HDSR value

4.1.10 Horizontal Display End Register (HDER): Address 84H

This register defines the time, in pixels, from the start of the **HSYNC** pulse to the end of the video display. (i.e. the first pixel which is not display).

This is a 14 bit register of which the bottom bit must be programmed to 0. If N pixels are required in this time, then value (N-18) should be programmed into the HBER. (N must be a multiple of 2).

31 30 29 28	27 26 25 24	23 22 21 20 19 18 17 16 15 14	13 12 11 10 9 8 7 6 5 4 3 2 1 0
1 0 0 0	0 1 0 0		x x x x x x x x x x x x x 0
			HDER value

4.1.11 Horizontal Border End Register (HBER): Address 85H

This register defines the time, in pixels, from the start of the **HSYNC** pulse to the end of the border display. (i.e. the first pixel which is not border).

This is a 14 bit register of which the bottom bit must be programmed to 0. If N pixels are required in this time, then value (N-12) should be programmed into the HBER. (N must be a multiple of 2).

Again, if no border is required, this register must still be programmed such that $N_{HBER} = N_{HDER}$.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	0	1											x	x	x	x	x	x	x	x	x	x	x	x	x	0

HBER value

4.1.12 Horizontal Cursor Start Register (HCSR): Address 86H

This register defines the time, in pixels, from the start of the **HSYNC** pulse to the start of the cursor display.

This is a 14 bit register of which all bits may be programmed. If N pixels are required in this time, then value (N-17) should be programmed into the HCSR. The cursor can thus be programmed to start on any pixel. In HiRes mode, the cursor can still only be programmed to start on a normal pixel boundary. However, because the resolution of the cursor can be defined to a micro-pixel, by using different cursor images it is possible to position the cursor to any micro-pixel.

Note that only the cursor start position needs to be defined, as the cursor is automatically disabled after 32 pixels in normal mode, or 16 pixels in HiRes mode. If a cursor smaller than this is required, then the remaining bits in the cursor pattern should be programmed to logical colour 00 (transparent).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	1	0											x	x	x	x	x	x	x	x	x	x	x	x	x	x

HCSR value

4.1.13 Horizontal Interlace Register (HIR): Address 87H

This register must be programmed if an interlaced sync. display is required. Otherwise it may be ignored. It defines the position of the **VSYNC** pulses in the odd field, and as such defines the offset of the odd and even fields. It should normally be programmed to be half way along a raster, so if value L is written into the HCR, then value L/2 should be written into the HIR.

This is a 14 bit register of which the bottom bit must be programmed to 0.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	0	1	1	1											x	x	x	x	x	x	x	x	x	x	x	x	x	0

HIR value

4.1.14 Horizontal Test Registers: Addresses 88H & 8CH

Two registers are provided for testing the chip in production. Neither of these registers are intended to be used during normal operation of the device. Writing a 14 bit value to address 88H writes that value into the horizontal counter immediately.

Writing a 14 bit value to address 8CH writes that value into all the horizontal registers simultaneously.

4.1.15 Vertical Cycle Register (VCR): Address 90H

This 13 bit register defines the period, in units of a raster, of the vertical scan. i.e. display time + flyback time.

If N rasters are required in a complete frame, then value (N-2) should be programmed into the VCR.

If an interlaced display is selected, (N-3)/2 must be programmed into the VCR. [N must be odd]. Here N is still the number of rasters in a complete frame, *not* a field.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	0	0	0	0											x	x	x	x	x	x	x	x	x	x	x	x	x	x

VCR value

4.1.16 Vertical Sync Width Register (VSWR): Address 91H

This 13 bit register defines the width, in units of a raster, of the **VSYNC** pulse.

If N rasters are required in the **VSYNC** pulse, then value (N-1) should be programmed into the VSWR. The minimum value allowed for N is 1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	0	0	0	1												x	x	x	x	x	x	x	x	x	x	x	x	x

VSWR value

4.1.17 Vertical Border Start Register (VBSR): Address 92H

This 13 bit register defines the time, in units of a raster, from the start of the **VSYNC** pulse to the start of the border display.

If N rasters are required in this time, then value (N-1) should be programmed into the VBSR.

If no border is required, this register must still be programmed, in this case to the same value as the VDSR.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	0	0	1	0												x	x	x	x	x	x	x	x	x	x	x	x	x

VBSR value

4.1.18 Vertical Display Start Register (VDSR): Address 93H

This 13 bit register defines the time, in units of a raster, from the start of the **VSYNC** pulse to the start of the video display.

If N rasters are required in this time, then value (N-1) should be programmed into the VDSR.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	0	0	1	1												x	x	x	x	x	x	x	x	x	x	x	x	x

VDSR value

4.1.19 Vertical Display End Register (VDER): Address 94H

This 13 bit register defines the time, in units of a raster, from the start of the **VSYNC** pulse to the end of the video display. (i.e. the first raster on which the display is *not* present).

If N rasters are required in this time, then value (N-1) should be programmed into the VDER.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	0	1	0	0												x	x	x	x	x	x	x	x	x	x	x	x	x

VDER value

4.1.20 Vertical Border End Register (VBER): Address 95H

This 13 bit register defines the time, in units of a raster, from the start of the **VSYNC** pulse to the end of the border display. (i.e. the first raster on which the border is not present).

If N rasters are required in this time, then value (N-1) should be programmed into the VBER.

If no border is required, then this register must be programmed to the same value as the VDER.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	0	1	0	1												x	x	x	x	x	x	x	x	x	x	x	x	x

VBER value

4.1.21 Vertical Cursor Start Register (VCSR): Address 96H

This is a 15 bit register. The lower 13 bits define the time, in units of a raster, from the start of the **VSYNC** pulse to the start of the cursor display. If N rasters are required in this time, then value (N-1) should be programmed into the VCSR. The upper 2 bits are used to control the display of the cursor in duplex LCD mode. They should be programmed to zero in all other modes.

When the upper 2 bits are programmed to be 11 (split screen) the meaning of VCSR and VCER are altered as follows. The cursor is displayed in the lower half-screen only from the value of VDSR to the value of VCSR, and again in the upper half screen only from the value of VCER to the value of VDER. This allows a cursor to be positioned across the boundary of the upper and lower half screens of an LCD.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	0	1	1	0										x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Bits 12–0: VCSR value

Bits 14–13:

00 normal operation
01 upper half-screen only
10 lower half-screen only
11 split screen

4.1.22 Vertical Cursor End Register (VCER): Address 97H

This 13 bit register defines the time, in units of a raster, from the start of the **VSYNC** pulse to the end of the cursor display. (i.e. the first raster on which the cursor is not present).

If N rasters are required in this time, then value (N-1) should be programmed into the VCER.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	0	1	1	1												x	x	x	x	x	x	x	x	x	x	x	x	x

Bits 12–0: VCER value

4.1.23 Vertical Test Registers: Addresses 98H, 9AH & 9CH

Three registers are provided for testing the chip in production. None of these registers are intended to be used during normal operation of the device.

Writing a 13 bit value to address 98H writes that value into the vertical counter immediately.

Writing any value to address 9AH increments the vertical counter by one raster immediately.

Writing a 13 bit value to address 9CH writes that value into all the vertical registers simultaneously.

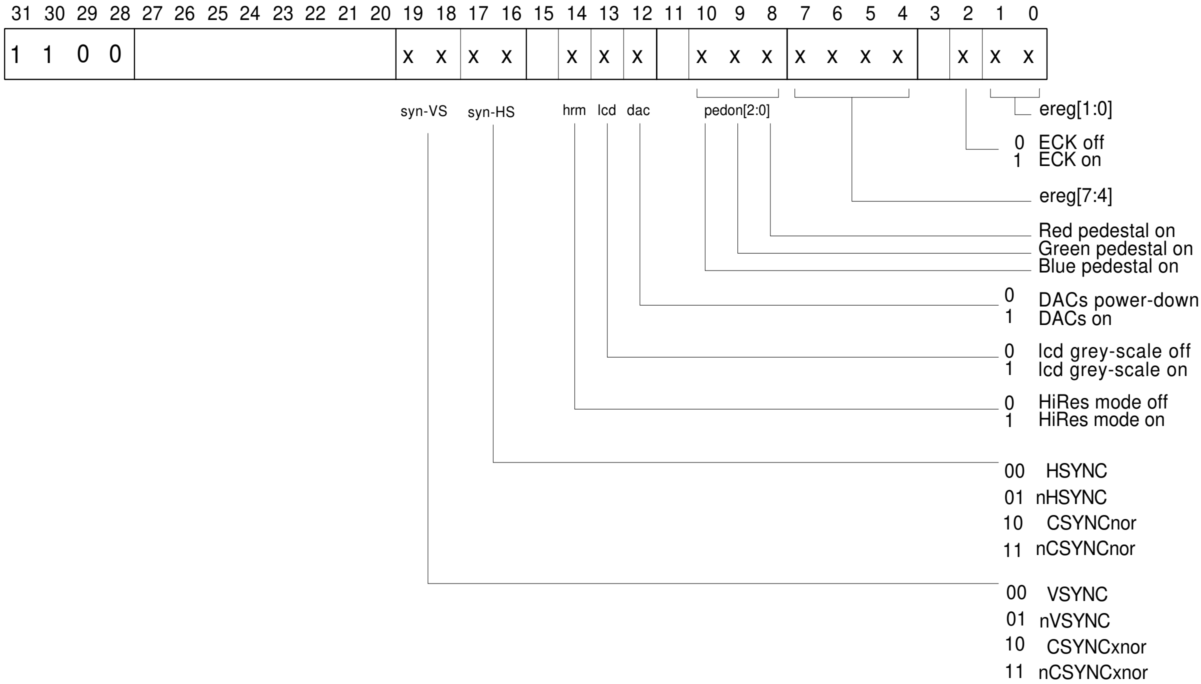
4.1.24 External Register (ereg): Address CH

This register contains the control bits for the external functions of VIDC20. In particular it controls the DACs, the configuration of the External Port **ED[7:0]**, and the configuration of the sync lines.

ereg[1:0] are exported from the chip as **EREG[1:0]**.
ereg[7:4] are exported from the chip on **ED[7:4]** if **ESEL[1:0]=3**. Refer to *Chapter 11.0 External Support*.

The use of pedon[2:0] and dac is defined in *Chapter 12.0 Analog Outputs*.

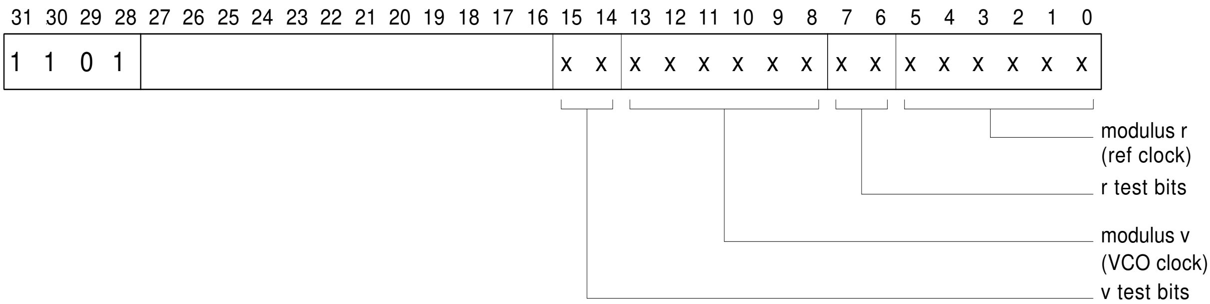
The uses of lcd and hrm are defined in *Chapter 10.0 Liquid Crystal Displays* and *Chapter 9.0 Hi-Res Support* respectively.



VIDC20 can export a variety of sync configurations on the pins **HSYNC** and **VSYNC**, as specified by the bits syn-HS and syn-VS respectively. For further explanation see *Chapter 11.0 External Support*.

4.1.25 Frequency Synthesizer Register (fsynreg): Address DH

VIDC20 is able to drive a VCO to provide a suitable input frequency for the pixel clock derived from a reference clock. This is achieved by dividing the reference clock by modulus r, and the VCO clock by modulus v, and comparing the resulting frequencies. Refer to *Chapter 5.0 Pixel Clock* for a more detailed explanation. The two moduli, r and v are each 6 bit values, and are programmed in this register.



Associated with each counter are 2 test bits which should normally be programmed to 0.

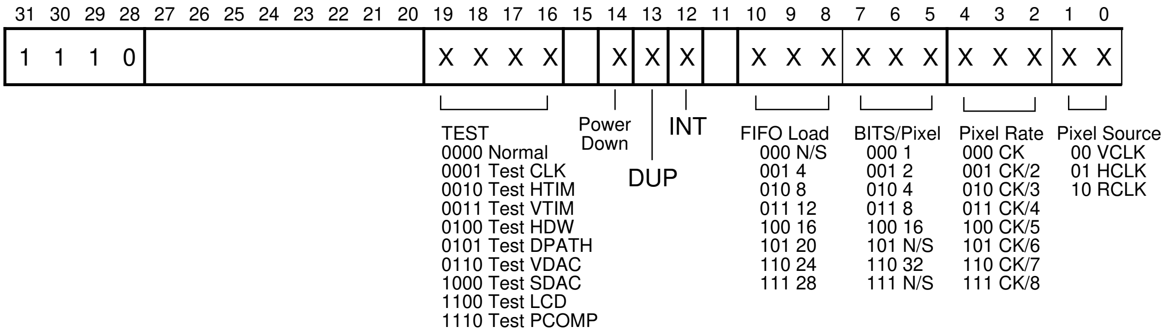
Setting bit[6] forces the phase comparator HIGH, which drives **PCOMP** HIGH.
Setting bit[7] clears the r-modulus counter.
Setting bit[14] forces the phase comparator LOW, which drives **PCOMP** LOW.
Setting bit[15] clears the v-modulus counter.

To reduce power consumption, this register should be programmed with large values when the frequency synthesiser is not in use. In particular, bits [6] and [14] should not be set at the same time.

To get a modulus of r, then value (r-1) should be programmed into the fsynreg. Likewise for the v-modulus.

4.1.26 Control Register (conreg): Address EH

The main control register determines the basic operation of the chip. In particular the pixel clock source, the pixel rate, the number of bits/pixel, the control of the video FIFO, and the data format are programmed. In addition there is a 4 bit test register which must be programmed to zero for normal operation.



The pixel clock (pixclk) is selected from one of 3 sources, corresponding to the respective input pins, and the selected clock is then fed through a prescaler as defined by the 3 bits conreg[4:2]. The output of this prescaler is the actual pixel clock. See *Chapter 5.0 Pixel Clock* for more detail.

The Video FIFO can be programmed to have any number of quad words loaded into it at the start of display. The value chosen should take into account the bandwidth of the display as well as the latency of the DMA subsystem. Refer to *Chapter 3.0 System Configurations* before programming these values.

If int is set, then VIDC20 will generate a display with interlace syncs. Otherwise, non-interlace syncs will be generated. If a true interlace display is required, then the memory controller must also be set up to generate an interlace display, as defined in the memory controller datasheet, and the HDW register in VIDC20 set up accordingly.

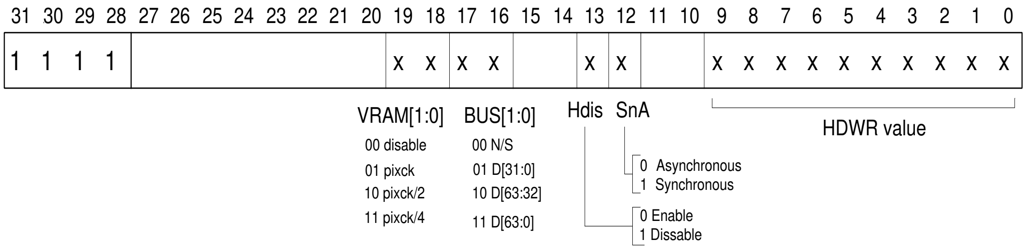
Setting the dup bit configures the display for dual-panel LCDs. This is described further in*Chapter 10.0 Liquid Crystal Displays*, but note that the memory controller must also be configured for this type of display.

Note that after a reset the Control Register should be the first register programmed. The Power Down bit (14) must immediately be programmed LOW. The test registers bits (16 to 19) also should be programmed LOW, as any other setting will inhibit normal operation.

4.1.27 Data Control Register (DCTL): Address FH

This register controls the way in which the external 64 bit bus is controlled. The control bit SnA defines the type of bus transfer to be expected for the DMAs. Generally, if MEMC10 is driving VIDC20, then SnA should be programmed LOW, and the **BUSCLK** input tied low. If the memory controller is fitted, SnA should be programmed HIGH, and **BUSCLK** should be connected to the DRAM memory clock.

VIDC20 can work in several different bus configurations, as described in *Chapter 3.0 System Configurations*. Data for programming the device, and data for the cursor and sound FIFOs is always presented on the lower 32 data bits (**D[31:0]**). Data for the video FIFO can be presented on the lower 32 bits only or the upper 32 bits only, or as 64 bits at a time. These modes are defined by the programming of BUS[1:0]. Note that if SnA is programmed to be 0, then BUS[1:0] must be programmed to be 01.



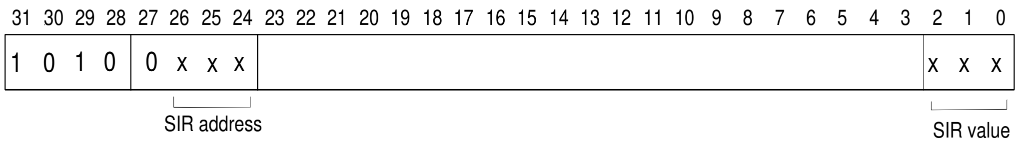
When using VIDC20 with VRAM, bits [19:18] must be programmed. These define the division of the pixel clock that is used to clock the VRAM (refer to *Chapter 3.0 System Configurations* for details). When the system memory does not contain Video RAM, these bits must be programmed to zero to save on power.

The horizontal display width is also defined in this register, and should be programmed to be the number of words of data in a displayed raster. It must be programmed in most configurations of the device, as it inhibits a DMA request near the end of a raster, when there are enough words in the video FIFO for that raster. The request is uninhibited after the **HSYNC** at the start of the next raster. When driving a dual panel LCD screen, this register must be programmed with twice the number of words in a displayed raster. This is a new feature to VIDC20, and for full compatibility with VIDC10, it may be disabled. This is achieved by programming bit 13 high.

Programming the horizontal display width has two functions: in normal mode, VIDC20 may be programmed so that the number of bits in a raster is a multiple of 32 (not 128 as is the case with VIDC10); in interlace mode the memory controller can safely update the video pointer during **HSYNC** time. In interlace mode, the number of bits in a raster must be a multiple of the number of words first loaded into the FIFO at the start of display. Note that if SnA is programmed to be 0, then the HDWR value is ignored (program bit 13 to 1), and the request is not inhibited near the end of the raster.

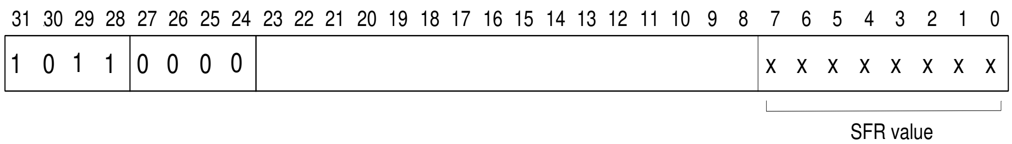
4.1.28 Stereo Image Register 0-7: Addresses A0H-A7H

These are 8, 3 bit registers which define the stereo position for the eight possible channels, as defined in *Chapter 13.0 Sound*.



4.1.29 Sound Frequency Register: Address B0H

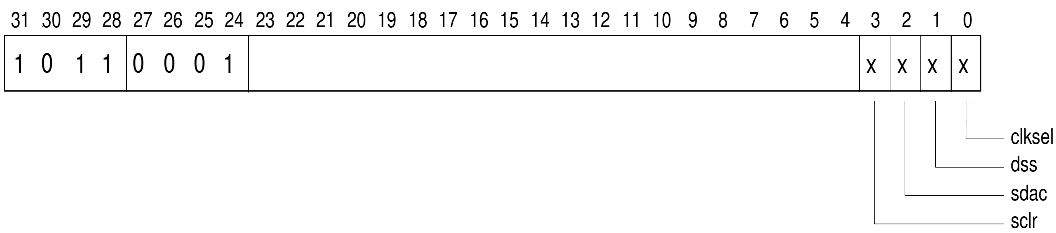
This 8 bit register specifies the byte sample rate of the sound data. It is defined in units of 1µS. See *Chapter 13.0 Sound* for more detail.



If a sample rate of N µs is required, then N-2 should be programmed into the SFR. N may take any value between 3 and 256.

4.1.30 Sound Control Register: Address B1H

This is a 4 bit register which defines various control bits for the sound system.



Bit 3: sclr

This bit should NEVER be programmed high.

Bit 2: sdac

When high, VIDC10 compatible sound is produced and the sound DACs are enabled. Two digital signals are also output from the chip in this mode. The first, **WS/LnR**, denotes whether the sound is for the left or right stereo channel (left = high). The other, **SD0/MUTE**, goes high between samples, when the sound DACs are being muted to allow for settling. These two signals are intended to ease the connection of external audio processing systems, but for basic operation can be ignored.

Bit 1: serial sound

This bit is used to select serial sound mode.

Bit 0: clksel

This bit is used to select which clock is used in the sound system. When high, the 24M reference clock is used (usually for VIDC10 sound only), when low the optional sound clock is used.

5.0 Pixel Clock

VIDC20 is capable of generating a display at any pixel rate up to 100MHz. The pixel clock may be selected from one of 3 sources, and then the frequency of this clock may be further divided down by a factor of between 1 and 8. These attributes are programmed by the lower 5 bits of the control register.

If a maximum of 3 master frequencies are sufficient, then the clock inputs can be used directly. However, it is often a requirement to have many different master clock frequencies. In order to obviate the need for many crystals on the PCB, VIDC20 is designed to drive a Voltage Controlled Oscillator (VCO) to provide the master frequency. The VCO and filter are external to VIDC20, but everything else is built into the chip. Operation is as follows.

A reference frequency is supplied on the pin **RCLK**. This reference signal will come from a crystal oscillator external to VIDC20, and it is recommended that the frequency, F_{ref}+, is 24MHz. The signal from the VCO is input into VIDC20 on the pin **VCLKIN**. **VCLKOUT** is simply the inverse of **VCLKIN**, and this may be used to bias the input signal about the threshold if the VCO output is not a full amplitude signal. The VCO output should be reasonably square if operation at 100MHz is to be achieved.

The reference clock is divided by a programmable number set by the r-modulus in the fsynreg. The VCO clock is divided by a programmable number set by the v-modulus in the fsynreg. Each of the moduli may be a 6 bit number. The output of each of these dividers is fed into a phase comparator, and the result is output from VIDC20 as **PCOMP**. This pin should then be filtered and used to control the VCO output frequency. In this way, the VCO can be set to have a frequency of $v/r * F_{ref}$.

The phase comparator is of the phase-frequency type. The output **PCOMP** is normally tristate, but when the VCO frequency needs to be decreased the output is LOW, and when the VCO frequency needs to be increased the output is HIGH. When the 2 frequencies are in lock, **PCOMP** will normally be tristate, but will be driven to the mid point for a very short time (a few ns) every r/F_{ref}+ period. The output impedance of this pin when it is driven is about 50Ω.

The choice of filter and VCO is left to the user. It is important to avoid any low-frequency modulation of the VCO frequency. It has been found that a suitable VCO is a 74AC04 inverter element with feedback, with the supply voltage controlled by the **PCOMP** output. See *Application Note 17:* "VIDC20 *Clock Sources*".

With this approach, an enormous number of frequencies are possible. The recommended F_{ref} of 24MHz can be used to yield the following common VCO frequencies. For some frequencies, there are many possible values of r and v. In this case it is sensible to choose a set of values which favours the filter response. (Remember large moduli yield a lower comparison frequency).

It may be best to limit the VCO range, and use the prescaler within VIDC20 to get a lower pixel rate than the VCO frequency. It is expected that the VCO range may have to be constrained so that it cannot provide the highest frequencies at which VIDC20 can operate. In this case, a single high-frequency clock can be fed into VIDC20 on the **HCLK** pin, and this can be selected for the pixel clock.

r-modulus	v-modulus	VCO frequency / MHz
6	2	8.0
4	2	12.0
3	2	16.0
2	2	24.0
41	43	25.171
50	59	28.320
3	4	32.0
2	3	36.0
31	58	44.903
12	35	70.0

Table 3: Synthesised VCO frequency settings

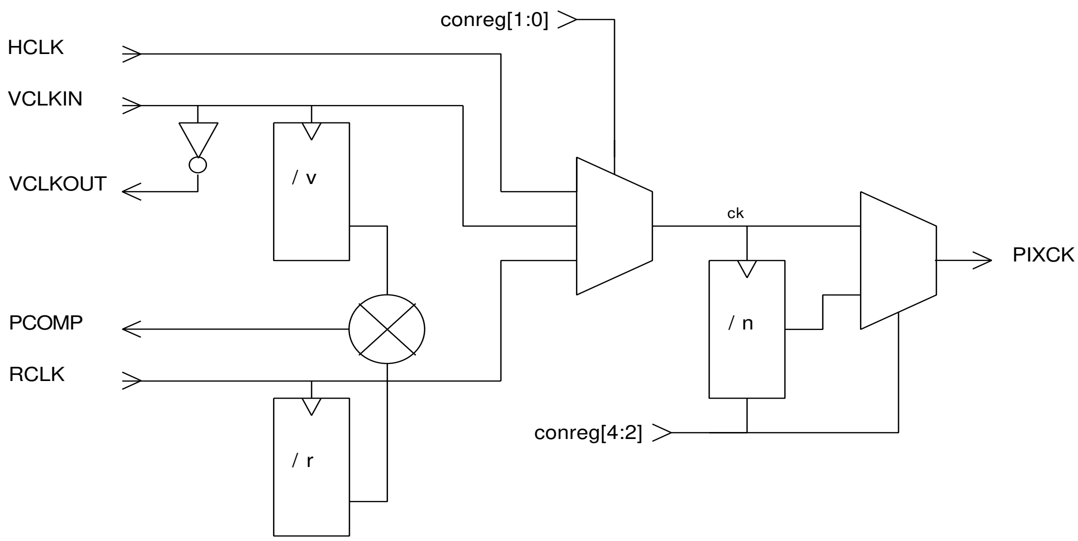


Figure 8: Subsystem for Pixel Clock Generation

6.0 Setting the FIFO preload value

The Video FIFO is a 32 entry, 32 bit wide FIFO. Words of video data are clocked into the top of the FIFO under control of **BUSCLK** and **nVIDAK**. In 64 bit mode, two words are loaded into the FIFO on each clock cycle. Words are clocked out of the bottom of the FIFO as the video system displays the data, which is controlled by the pixel clock.

The FIFO is flushed during vertical flyback time, so before the start of the frame the FIFO is empty. At the start of the frame a video request is made to the memory system by asserting **nVIDRQ**. When a predetermined number of words have been loaded into the FIFO the request is removed. As the data in the FIFO is displayed, further video requests are made to refill the FIFO to the desired level.

The Control Register includes a 3 bit field (bits 10:8) to set the preload value of the Video FIFO. In this way the FIFO can be programmed to load 4,8,12,16,20,24 or 28 words of data into the FIFO at the start of frame. Note that in 64 bit bus mode, the memory system will deliver data in blocks of 8 words, so programming the preload value to be an odd number will result in the FIFO filling to the next even value. After the start of frame, the FIFO will request more data when the number of words in it falls below the preloaded value.

The point at which the FIFO should request more data to be loaded is dependent upon system considerations: if the FIFO is reloaded too late, there is a danger that it will run out of data (underflow); if it is reloaded too early, then there is a danger that the data will not fit into the FIFO (overflow). In general, the higher the bandwidth of the screen, then the more words need to be preloaded into the FIFO. In a low bandwidth screen mode, it is not always desirable to have a large preload value, as the bus traffic will have long bursts of data transfer at the start of the frame.

The optimum value to be preloaded depends upon the screen mode in use (ie the rate at which data is read from the FIFO), and both the latency of the memory controller and the rate at which data is provided to VIDC20. It is generally prudent to program the minimum value possible to keep the bus traffic even.

Let:

n	be the value programmed into the control register.
v	(words/ s) be the rate at which video data is displayed
lmax	(s) be the maximum latency in the memory system. (This is the maximum time between VIDC20 requesting more video data and the memory system delivering the first word of that data).
lmin	(s) be the minimum latency in the memory system.
Pb	(s) be the period of **BUSCLK**

If the FIFO is almost empty then it takes 0.05us for a word of data to reach the bottom of the FIFO before it can be used.

The minimum value for n is deduced from the following condition to avoid the FIFO underflowing:

There are 4n words in the FIFO when the FIFO requests more data, and if not refilled, then the FIFO would be empty in 4n/v us. The request from the FIFO is synchronised to **BUSCLK** before **nVIDRQ** is asserted to the memory controller. This takes a maximum of 2Pb.

So n must be chosen such that $4n/v > (2Pb + lmax + 0.05)$.

The maximum value for n is deduced from the following condition to avoid the FIFO overflowing:

In 32 bit bus mode, n may take the maximum value of 7, and the FIFO can never overflow, as there will always be 4 words available in the top of the FIFO, even if the video request is serviced immediately.

In 64 bit bus mode, if n takes the maximum value of 6, then the FIFO will never overflow, as there will always be 8 words available in the top of the FIFO, even if the video request is serviced immediately.

However, in very high video bandwidth modes, the value of 7 may be used. Then the criteria is as follows. There are 4 words free in the FIFO when it requests more data. If the request is serviced as soon as possible, then there must be enough room in the FIFO for 8 words, when the last word enters the FIFO. The FIFO requires another 4/v us to free up 4 extra words (giving a total of 8 free words). The minimum time from the FIFO requesting data to getting the last word into the FIFO is $Pb + lmin + 3Pb$.

So, if $(Pb + lmin + 3Pb) < 4/v$, then n may be programmed to value 7.

7.0 The Palette

VIDC20 has a 28-bit wide 256-entry palette which is constructed out of 3 8-bit wide look-up-tables (LUTs), each with 256 entries, named Red, Green, and Blue, and one 4 bit wide LUT with 16 entries, named Ext. The Red, Green and Blue LUTs each drive their respective DACs, and the Ext LUT is normally configured to drive the **ED[3:0]** output port, except when Hires mode or LCD mode is selected. These bits may be used outside the chip for a variety of purposes such as supremacy, fading, Hi-Res and LCD driving. The **ED[7:4]** output port is normally driven from the Ext register, ereg[7:4], which may be written at any time, so these bits can be used as DC control bits.

The mapping of the logical colours through the LUTs is dependent on the mode in use, as follows:

- In 1,2,4 bits/pixel modes, the logical data is fed simultaneously to all 4 LUTs. This gives a fully flexible palette with any logical colour being mapped to any physical colour, and any **ED[3:0]** value. The palette will give 16 colours from a selection of 2^{24}.
- In 8 bits/pixel modes, the logical data is fed simultaneously to all 4 LUTs. This gives a fully flexible palette with any logical colour being mapped to any physical colour. Logical colours 0-15 access the Ext LUT, and logical colours 16-255 access location 0 of the Ext LUT. The Ext LUT again drives **ED[3:0]**. The palette will give 256 colours from a selection of 2^{24}.
- In the 16 bits/pixel mode, a patented technique has been developed. This approach is highly flexible and allows many different addressing modes e.g. 5-5-5, 5-6-5 etc. In this mode 2^{16} colours are available from a selection of 2^{24}. Please see the *Applications Note "16 bit Colour"*.
- In the 32 bits/pixel mode, 24 bits from the logical field will drive the 256 entries in each of the colour LUTs (8 bits to each LUT) and 4 bits will drive the Ext LUT. The upper 4 bits are discarded.The palette will give the full range of 2^{24} colours.

Note that where a logical field does not drive all the palette entries (such as in 4 bits/pixel mode) only the lower part of the palette is used. Unused sections need not be programmed.

When HiRes mode or LCD mode is selected, then the palette must be set up in a predetermined configuration. This is explained in *Chapter 9.0 Hi-Res Support* and *Chapter 10.0 Liquid Crystal Displays* respectively.

7.1 Palette Updating

A signal **FLYBK** is provided which is identical to that on VIDC10. i.e. **FLYBK** goes HIGH at the start of the first raster which is not display, and goes LOW at the start of the first raster which is display. The rising edge of this signal may cause an interrupt in the memory controller, and the palette should be updated at this time for flicker-free updating.

8.0 Cursor

VIDC20 has a hardware cursor 32 pixels wide and any number of pixels high. Its 2 bits/pixel allow 4 colours, which include “transparent” plus three other colours from a selection of 2^{24}. It is possible to display the cursor in the horizontal border, but not in the vertical border.

The cursor has a 3 entry palette which is 28 bits wide, allowing each cursor logical colour to be any physical colour. In addition, there is a 28 bit wide border colour register.

At the start of every frame, 16 bytes of cursor data are transferred to VIDC20 during the horizontal retrace period. This is enough data for two raster's worth of cursor. After they have been displayed, a request is made for another 16 bytes. Thus, in normal mode, requests are made on every other raster on which there is cursor, and enough data is transferred for two rasters each. In Hi-Res mode, a request is made every raster. Note that the cursor data is always transferred in bursts of four words, and is always presented to VIDC20 on the lower 32 data bits, irrespective of the memory configuration chosen.

Note that even when a cursor is not being displayed, VIDC20 will still make a request for cursor data at the start of every frame. The memory controller may choose to ignore this, or may satisfy the request with arbitrary data.

8.1 Cursor in HiRes Mode

In order to allow micro-pixel resolution of the cursor in Hi-Res mode when operating at 4 micro-pixels per normal pixel, it is necessary to define 2 bits per micro-pixel, or 8 bits per normal pixel. The 16 bytes of cursor data available for each raster can thus generate 64μ-pixels of cursor. In Hi-Res mode the cursor palette is not used (though the border may be programmed). Refer to *Chapter 9.0 Hi-Res Support*.

The cursor is always positioned to align with a normal pixel. In order to position the cursor to a μ-pixel horizontally, four different copies of the cursor are required: each copy defines the cursor offset by a single μ-pixel. It is possible to define transparency to a resolution of a μ-pixel, so by selecting the correct cursor image, the required position can be achieved.

8.2 Cursor in Interlace Mode

As stated earlier, VIDC20 and a suitable memory controller can generate true interlace displays, where the data in memory is contiguous irrespective of whether interlace is selected or not. The cursor however needs special treatment. A description of how the cursor is handled with an ARM memory controller follows.

VIDC20 can only display a cursor starting on an even field, and there must be the same number of rasters of cursor in each field. The cursor data consists of a linear buffer, and the ARM memory controller only reinitialises the cursor pointer during the retrace period of every even frame. Thus when the cursor is required to start on an even field the cursor buffer consists of the data for the even field, followed by the data for the odd field. When the cursor is required to start on an odd field the cursor buffer consists of a transparent raster followed by the data for the odd field, followed by the data for the even field, ending with a transparent raster. It is simplest therefore to have two cursor buffers as described. Depending on whether the cursor is required to start on an even or odd field, the cursor buffer pointer in the memory controller is programmed to point to one or other of the two cursor buffers described. The cursor should only be moved as a result of the **FLYBK** interrupt, and if it is moved at the end of an even field, then it is necessary to reprogram the cursor buffer pointers.

8.3 Cursor in LCD mode

VIDC20 is capable of displaying the hardware cursor in LCD mode. However, because of the split-screen nature of duplex LCDs, the cursor needs special attention. If the cursor is entirely in the upper or lower half-screen, then the cursor should be programmed as normal, but VCSR[14:13] should be programmed accordingly (10 = upper half-screen; 01 = lower half-screen). If the cursor "straddles" the split screen, then the cursor image in memory must start at the top of the lower half-screen, and end with the bottom of the upper half screen. Hence two contiguous images of the cursor image are required, and the start pointer moved accordingly. In practice, four images of the cursor are required, to ensure that a resolution of one raster is maintained across the boundary. As the cursor moves from one panel to the other, the pointer to the cursor image in memory must be moved. For more details on handling the cursor in LCD mode, please refer to the LCD section within the memory controller datasheet and the ARM Application Note on LCD screens.

In the case where the cursor straddles the split screen, the meaning of the VCSR and VCER registers are changed. The VCER register now defines the start of cursor in the upper half-screen, and the VCSR defines the end of the cursor in the lower half-screen. Thus the cursor is actually displayed in the lower half-screen from the start of display until VCSR, and then again in the upper half-screen from VCER until the end of display. This mode is selected by programming VCSR[14:13] = 11.

9.0 Hi-Res Support

VIDC20 is able to support colour screens with a resolution up to 1024 by 768 pixels. For higher resolutions, externally serialising the data is required to produce monochrome (or grey-level) pictures. In this scheme one 15ns-pixel could theoretically be serialised to make eight 2ns-pixels i.e. about 500MHz.

9.1 VIDC20 Support for Hi-Res Mode

When the hrm bit in the Ext register is set, and **ESELl[1:0]** are set to value 10, VIDC20 outputs 8 bits of data for every normal pixel on the **ED[7:0]** port. These bits can then be serialised to form a high frequency monochrome pixel stream; alternatively they can be serialised to 2 or 4 bits, which could then drive a high-speed monochrome DAC for grey level displays. With VIDC20 running at a fundamental clock frequency of about 100MHz, the external serial clock could be running at up to several hundred MHz. In order for the external circuit to be able to synchronise to the VIDC20 output data, VIDC20 also outputs a pixel clock synchronous to the data stream when the hrm bit is set.

In this mode, with **ESELl[1:0]** set to value 10, the video data is driven from the Blue LUT, which outputs data **BPD[7:0]**. Depending on how the external serialiser circuit is arranged, the LUT must be set up to give a one-one correlation between the logical address and the physical data value. So, for example, if 4 bits are externally serialised into a single bit stream, then 4 bits/pixel mode should be selected, and **ED[6,4,2,0]** should be used. The lower 16 words of the Blue LUT should be programmed to give all 16 combinations of **BPD[6,4,2,0]**. If 8 bits are externally serialised to give a single bit-stream, then 8 bits/ pixel mode should be selected, and all 256 values of the Blue LUT should be programmed as a one-one mapping.

Hardware cursor support is provided as follows. The cursor palette is not used, though the Blue border may be programmed. Eight bits of cursor data (**CD[7:0]**) are defined for each normal pixel. The 8 bits are divided into 4 pairs, with the lsb (least significant bit) of each pair defining whether the video data (BPD) or the msb (most significant bit) of the cursor pair is displayed. Each cursor bit-pair operates on 2 bits of the video data (BPD) according to the following tables:

CD[7]	CD[6]	ED[7]	ED[6]
0	0	BPD[7]	BPD[6]
0	1	0	0
1	0	BPD[7]	BPD[6]
1	1	1	1

Table 4: Deriving high speed 2 bit cursor data from the normal 8 bit output - CD[6&7]

CD[5]	CD[4]	ED[5]	ED[4]
0	0	BPD[5]	BPD[4]
0	1	0	0
1	0	BPD[5]	BPD[4]
1	1	1	1

Table 5: Deriving high speed 2 bit cursor data from the normal 8 bit output - CD[4&5]

CD[3]	CD[2]	ED[3]	ED[2]
0	0	BPD[3]	BPD[2]
0	1	0	0
1	0	BPD[3]	BPD[2]
1	1	1	1

Table 6: Deriving high speed 2 bit cursor data from the normal 8 bit output - CD[2&3]

CD[1]	CD[0]	ED[1]	ED[0]
0	0	BPD[1]	BPD[0]
0	1	0	0
1	0	BPD[1]	BPD[0]
1	1	1	1

Table 7: Deriving high speed 2 bit cursor data from the normal 8 bit output - CD[0&1]

So if the external circuit serialises **ED[6,4,2,0]** into a single bit stream, or **ED[7:0]** into a 2-bit data stream then the cursor can be positioned and defined to any micro-pixel: in each case the cursor can be transparent, black or white. If all 8 bits are serialised into a single very high frequency bit stream, then the cursor can only be positioned and defined to units of 2 micro-pixels.

10.0 Liquid Crystal Displays

VIDC20 is capable of driving single panel Liquid Crystal Displays at 1, 2, 4, 8, 16 or 32 bits per pixel, and dual panel LCDs at 1, 2 or 4 bits per pixel. Grey-scaling is provided at up to 16 shades. VIDC20 is also capable of driving single panel colour LCDs with no grey scaling in its normal (video) mode. Two control bits are provided for LCD operation:

lcd (bit 13 in the Ext register) configures the external data port **ED[7:0]** for LCD operation, and enables the grey-scaling logic (**ESELl[1:0]** must be set to 01);

dup (bit 13 in the control register) enables duplex mode, and should be set for dual-panel LCDs.

10.1 LCD grey-scaling

To obtain a grey-scaled output fromVIDC20, the lcd bit (bit 13 in the Ext register) must be set. This configures the External port for LCD operation. The DACS should be disabled to save power since VIDC20 can not drive both CRT and LCD displays simultaneously. In order to get this data out of the **ED[7:0]** port, **ESEL[1:0]** must be set to value 01.

VIDC20 provides a grey-scaling algorithm which modulates the data output. Grey-scaling is possible at 1, 2 or 4 bits/pixel. The data is output from the chip as one or two 4-bit quantities, depending on whether single or dual panel LCDs are used, at one quarter of the pixel rate. The lower 4 bits of the Green LUT control the upper panel (**ED[7:4]**), and the 4 bits of the Ext LUT control the lower panel (**ED[3:0]**). Thus, the palette can still be used to provide a mapping of logical to physical colour. The cursor palette is used similarly, though the programming of the cursor position needs special treatment - refer to the chapter on the cursor. If a single panel LCD is used, then **ED[7:4]** should be used, and the Green LUT programmed accordingly (**ED[3:0]** are held low in this mode). The grey-scaling logic lies between the output of the video multiplexer and the external port as can be seen in *Figure 2: VIDC20 Block Diagram* and works as follows.

There are effectively 16 physical grey levels available, and in 1,2, or 4 bits/pixel mode the palettes are programmed to give a mapping of the logical colour to physical shade. The resultant 4 bit pixel value out of the video multiplexer is modulated according to its value and the raster number and the point on the raster at which it is generated. The result is a single bit which on average is HIGH for a time equal to the actual 4 bit value. For a single panel screen, 4 of these bits are then collected together and output as a nibble at one quarter of the pixel rate on **ED[7:4]**. **ED[4]** represents the 4th pixel, and **ED[7]** represents the 1st pixel.

If duplex mode is selected, then the pixel stream for the upper half screen is obtained from the Green LUT and that for the lower half screen is obtained from the Ext LUT. Both these pixel streams are passed through the grey-scale logic simultaneously and output as two nibbles on **ED[7:4]** (upper half screen) and **ED[3:0]** (lower half screen).

10.2 Dual Panel LCDs (Duplex Mode)

This mode requires a suitable memory controller. Duplex mode is configured by setting the dup control bit as well as the lcd control bit. The screen parameters are set up according to the requirements of the LCD panel. (Note that, since the upper and lower panels are driven simultaneously, VIDC20 only produces data for half the total number of lines on the dual panel. Thus, the vertical registers must be programmed as if there were only one panel). VIDC20 requests data in units of two quad-words. The first quad word the memory controller delivers is for the upper half-screen, and the second quad-word is for the lower half-screen. VIDC20 then serialises the data into 2 simultaneous bit-streams as described above. 1,2 or 4 bits/pixel may be selected.

10.3 Single Panel Colour LCDs

If neither dup nor lcd control bits are set, then the **ED[7:0]** port may be used to gain access to all of the physical bits out of the video multiplexer. This would allow many other types of display to be driven. Please see the ARM Application Note “Using VIDC20 with LCDs”.

11.0 External Support

11.1 The External Port

VIDC20 has an 8 bit output port, **ED[7:0]**, a 2 bit output control port, **EREG[1:0]**, and a synchronous clock, **ECLK**, which have different functions in different modes. The port is controlled by 2 input signals, **ESEL[1:0]** which essentially select which of the bytes from the video multiplexer are chosen. **EREG[1:0]** are always driven, so it is possible to connect **EREG[1:0]** to **ESEL[1:0]**, and then VIDC20 completely controls this port.

When **ESEL[1:0]** = 0, the Red LUT is output on **ED[7:0]**.

When **ESEL[1:0]** = 1, if lcd = 0 the Green LUT is output on **ED[7:0]**. If lcd = 1, then the grey-scaled LCD signals are output. **ED[7:4]** carries the data for the upper half screen from the Green LUT, and **ED[3:0]** carries the data for the lower half screen from the Ext LUT. Note that if lcd = 1, then data is output at one-quarter of the VIDC20 pixel rate, as the data output actually represents 4 pixels for each half-screen.

When **ESEL[1:0]** = 2, if hrm = 0, the Blue LUT is output on **ED[7:0]**. If hrm = 1, the multiplexed Blue LUT and HiRes cursor data is output on **ED[7:0]**. Refer to the chapter on HiRes displays for more detail. If hrm = 1, then **ED[7:0]** is retimed, and delayed by one extra pixel.

When **ESEL[1:0]** = 3, if dac = 0, **ED[3:0]** are driven by the Ext LUT, and **ED[7:4]** are driven by the value of the Ext Register,**EREG[7:4]**, which is intended as a DC control port in this mode. If dac = 1, **ED[3:0]** are delayed by one pixel, so that they are exported from the chip in the same pixel as the analog data to which they correspond. In this configuration **ED[3:0]** bits may be used for supremacy, for overlaying pictures on a pixel-by-pixel basis. Because several bits are output, analog fading and mixing on a pixel basis is possible.

11.1.1 ECLK

ECLK is output along with the data **ED[7:0]**, so that the data can be externally latched and multiplexed. **ECLK** is controlled by lcd and **EREG[2]**. If **EREG[2]** = 0, then **ECLK** is output as logic 0. This should be configured whenever **ECLK** is not required, in order to save power. If **EREG[2]** = 1, then if lcd = 0, **ECLK** is the pixclk, output synchronously with the data stream. If lcd = 1, then **ECLK** is the LCD clock, which runs at a quarter of the pixel rate. The lcd clock is only enabled whilst horizontal display data is being output and is synchronous to the data stream.

11.2 Power Saving Considerations

The External Port can consume a lot of power, but steps may be taken to minimise power usage. In particular, it is very important not to load the signals heavily, especially **ECLK** which can clock at the pixel rate. When it is not in use, it should not be putting out the raw pixel data, but should be outputting static signals. This is done by selecting **ESEL[1:0]** = 3, and setting all entries of the Ext LUT to be all one value. **ECLK** should be turned off by setting **EREG[2]** = 0.

If an LCD is fitted, but not operated, it may be necessary to power down the input signals to it. This can be achieved by setting bit 13 low, which disables the grey scaler, and by disabling the external port as described above.

11.3 Vertical and Horizontal Synchronisation

Software control over the polarities of the synchronisation pulses is provided. Two types of Composite Sync may be output, each of either polarity. The logical OR of Hsync an Vsync may be output on the Horizontal Sync (**HSYNC**) pin, and the XOR of Hsync and Vsync may be output on the Vertical Sync (**VSYNC**) pin. Equalisation pulses in the composite synchronisation signal are supported for interlace mode. When LCD mode has been selected, the external **HSYNC** and **VSYNC** pulses are modified in accordance to the requirements of an LCD screen.

The **HSYNC** and **VSYNC** pins are programmed with the Ext Register, **ereg[19:16]**.

In addition to the above, VIDC20 outputs another signal named **VnC**, (Video /not Cursor) for memory controllers to use to discriminate between video and cursor requests. **VnC** is unaffected by the configuration of the **HSYNC** and **VSYNC** pins.

11.4 Genlocking

Genlocking support identical to that on VIDC10 is provided: a pin is provided to reset the vertical counter to the first raster (SINK), and the horizontal locking is achieved by comparing the phase of the **HSYNC** pulses and phase-locking the input clock. In addition, the chip outputs **FLYBK**.

FLYBK goes HIGH at the start of the first raster which is not active display (but may be in the border), and goes LOW at the start of the first raster which is active display. **FLYBK** thus gives information about the vertical timing.

12.0 Analog Outputs

VIDC20 outputs analog R, G, and B signals. It is designed to drive doubly-terminated 75Ω lines directly.

12.1 DAC Control

There are 4 control bits in the Ext Register associated with the DACs. These are dac and ped[2:0].

12.1.1 Power-Save Mode

When dac is HIGH, the DACs are all enabled and will generate a current proportional to the digital values from the video multiplexer. When dac is LOW, the reference current into all 3 DACs is turned off, so the DACs generate no output current, and hence consume much less power. This is useful when operating in LCD mode, or at any time when the screen should be blanked.

12.1.2 Pedestal Current

The DACs may be programmed to generate a pedestal offset of 20 lsb equivalent currents. These are controlled individually by pedon[2:0], though they will typically all be programmed on or off together, depending on the monitor characteristics. pedon[0] controls the red pedestal, pedon[1] the green pedestal, and pedon[2] the blue pedestal. If pedon[n] is HIGH, the pedestal current is switched on as the border starts, and is turned off as the border ends.

12.2 Video DAC Currents

The DACs are each 8 bit resolution, so they source 256 units of current according to the digital value from the video multiplexer. The current step is set by a common reference current, **VIREF**. The recommended reference current is 0.56mA which gives a DAC step of 69μA. Hence digital value 0 gives 0 current and digital value 0xFF gives an output current of (255 * 69) 17.6mA. If pedon is set, then during display time, digital value 0 will generate (20 * 69) 1.38mA, and digital value 0xFF will generate (275 * 69) 18.98mA. A 4.3k resistor connected between**VIREF** and **VDD** will provide the desired 0.56mA at 2.6V.

12.2.1 DAC Accuracy

At 100MHz the DACs are accurate to 8 bits absolute resolution. They will always be monotonic.

12.3 Monochrome Output

VIDC20 does not generate a separate composite monochrome signal. This can be generated by resistively mixing the R,G and B externally, if required.

12.4 ESD protection

Additional circuitry has been incorporated to give ESD protection of around 1KV, though it is recommended to fit catch diodes to the PCB.

13.0 Sound

VIDC20 has two sound circuits built into it. These are an 8-bit VIDC10 compatible system, and a 32-bit serial sound interface suitable for driving external CD DACs. Only one of these systems should be used at once, as they share common circuitry.

13.1 The Sound Core

At the core of the sound system is a 4 word FIFO and a byte wide latch. When empty, the FIFO fills completely by a DMA request. Data is then clocked out of the FIFO, one byte at a time through the latch.

13.2 VIDC10 Sound

This mode can work with 1, 2, 4 or 8 stereo channels. The eight bit sound data is fed through a DAC to produce the sound signal. The first quarter of each sample is muted to allow for DAC settling. During this time, the external, digital **SDO/MUTE** signal goes high. The stereo image is synthesised by time division multiplexing the sound signal between the left and right outputs. A digital output, **WS/LnR**, is provided which denotes when the output data is for the left (**WS/LnR** = 1) or right stereo channel. The stereo position for each channel is held in the stereo image registers, and can be in one of eight positions as described below.

SIR Value	Stereo Position
0	Undefined
1	100% Left
2	83% Left
3	67% Left
4	Centre
5	67% Right
6	83% Right
7	100% Right

Table 8: Stereo Position Values

In 8 channel mode, the channels are sampled sequentially starting with channel 0. In 4 channel mode, the fifth byte sampled is channel 0 again, and so stereo image register 4 must be programmed to the same value as that of register 0, and so on. In 2 channel mode, registers 0, 2, 4 and 6 correspond to channel 0, and registers 1, 3, 5 and 7 correspond to channel 1. In single channel mode, all eight registers must be programmed to the same value.

The sample rate is programmable via the Sound Frequency Register (SFR). The SFR is programmable in units of 1µS, and the minimum value is 3µS. In eight channel mode, each channel is sampled at a rate of one eighth the value of the SFR.

The DAC transfer characteristic consists of 16 linear segments (chords). Each chord has 16 steps, and the step size in one chord is twice that of the preceding chord. This gives an approximation to the “µ225 law”. The characteristic is shown in the figure below, for the positive and negative halves. Note that bit 0 of the sound data is used as the sign bit.

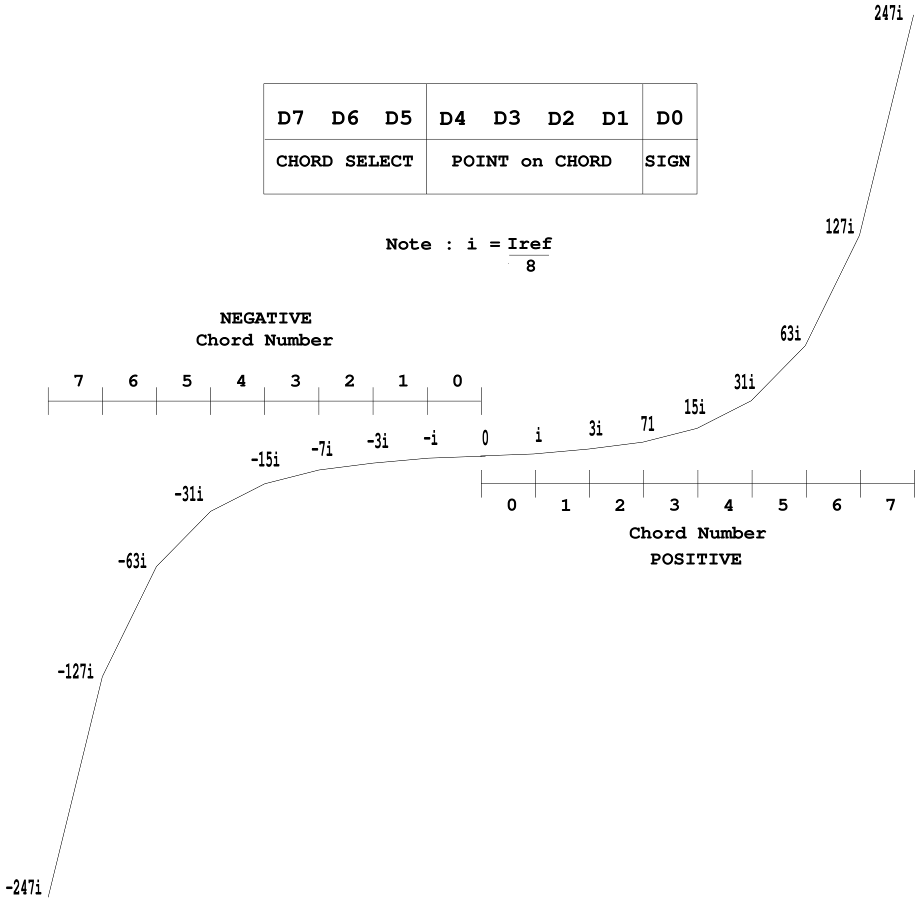


Figure 9: VIDC20 Sound DAC Characteristic

The outputs are current sinks. The magnitude of the output is a function of the sound reference current. The reference current is equal to the step size of the highest chord. It is recommended that the reference current is 32µA, which may be provided by a 88.5k resistor to**VDD**. The digital outputs, **SDO/MUTE** and **WS/LnR** are provided for advanced external audio systems but for basic operation they can be ignored.

When connected to VDD by a 1k pullup resistor the analog stereo outputs**LS** and **RS** will have a voltage range of 4.0 1V with a 2mA reference current.

13.3 The Serial Sound Interface

The serial sound interface is new to VIDC20, and offers far superior 32 bit stereo sound, at the expense of some additional external circuitry. The serial sound system consists of a three pin serial interface.

When in this mode, bytes from the sound FIFO are output in most-significant-first order. This is because the serial sound output must go msb first to be compatible with other serial sound devices. Each byte of data is loaded into a parallel-in, serial-out register, and clocked out under control of the bit clock.

Each 32 bits sample consists of 16 bits for the left hand channel, and 16 bits for the right hand channel. To distinguish between them, a 'word select' (**WS/LnR**) signal is produced. This signal changes when the lsb of the previous word is output. When **WS/LnR** is high, the right hand channel is being output. This timing relationship is shown below.

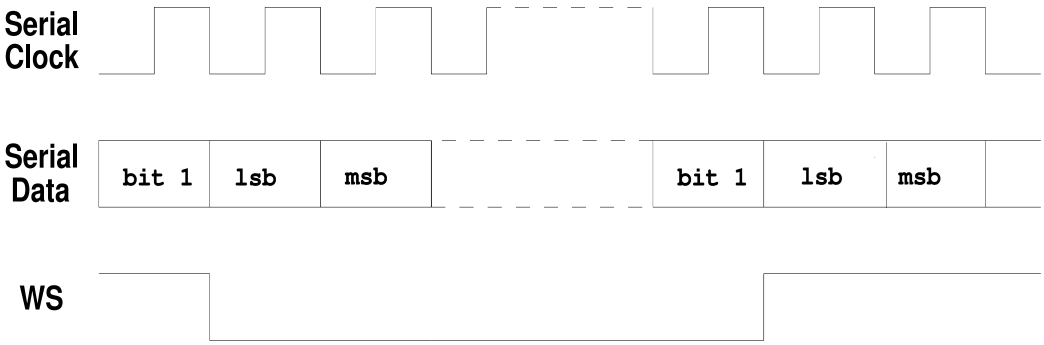


Figure 10: Serial Sound Output Timing

The serial sound output can be used with any DAC with a serial sound input. Many DACs require a 11.2896MHz input clock, and to reduce the number of on board crystals required, VIDC20 can cope with this frequency on the **SCLK** input. When using this, the following parameters need to be programmed in the registers.

serial sound (SCTL Register bit 2) = 1
clksel (SCTL Register bit 0) = 1
Sound Frequency Register = 2

VIDC20 is not limited to operating with this frequency alone, however the Sound Frequency Register must be set to produce the necessary bit rate accordingly.

13.4 Sound Outputs

When the sound system is in VIDC10 mode, analog sound data is output from the sound DACs. There are two outputs, one for the left hand channel, and one for the right. Also, two digital outputs are produced. On the **SDO/MUTE** pin, the mute signal is output. This goes high for a period between samples to allow for DAC settling. On the **WS/LnR** pin, the Left-not-Right signal is output. This goes high when the analog output is to the left hand channel, and goes low when it is for the right hand channel. These two digital outputs are intended to ease the connection of an external audio processing system, and can be ignored for normal use.

When in serial sound mode, these digital outputs are SDO (serial data out) and WS (word select). Also, the serial data clock, **SDCLK** is output. When no sound is required, (sctl[2:1]=0), these outputs are stable (**SDCLK**=0, **SDO/MUTE**=0, **WS/LnR**=1).

14.0 Boundary Scan Test Interface

The boundary-scan interface conforms to on the IEEE Std. 1149.1- 1990, Standard Test Access Port and Boundary-Scan Architecture (please refer to this standard for an explanation of the terms used in this section and for a description of the TAP controller states.)

14.1 Overview

The boundary-scan interface provides a means of testing the core of the device when it is fitted to a circuit board, and a means of driving and sampling all the external pins of the device irrespective of the core state. This latter function permits testing of both the device's electrical connections to the circuit board, and (in conjunction with other devices on the circuit board having a similar interface) testing the integrity of the circuit board connections between devices. The interface intercepts all external connections within the device, and each such "cell" is then connected together to form a serial register (the boundary scan register). The whole interface is controlled via 5 dedicated pins: **TDI**, **TMS**, **TCK** and **TDO**. *Figure 11: Test Access Port (TAP) Controller Sate Transitions* shows the state transitions that occur in the TAP controller.

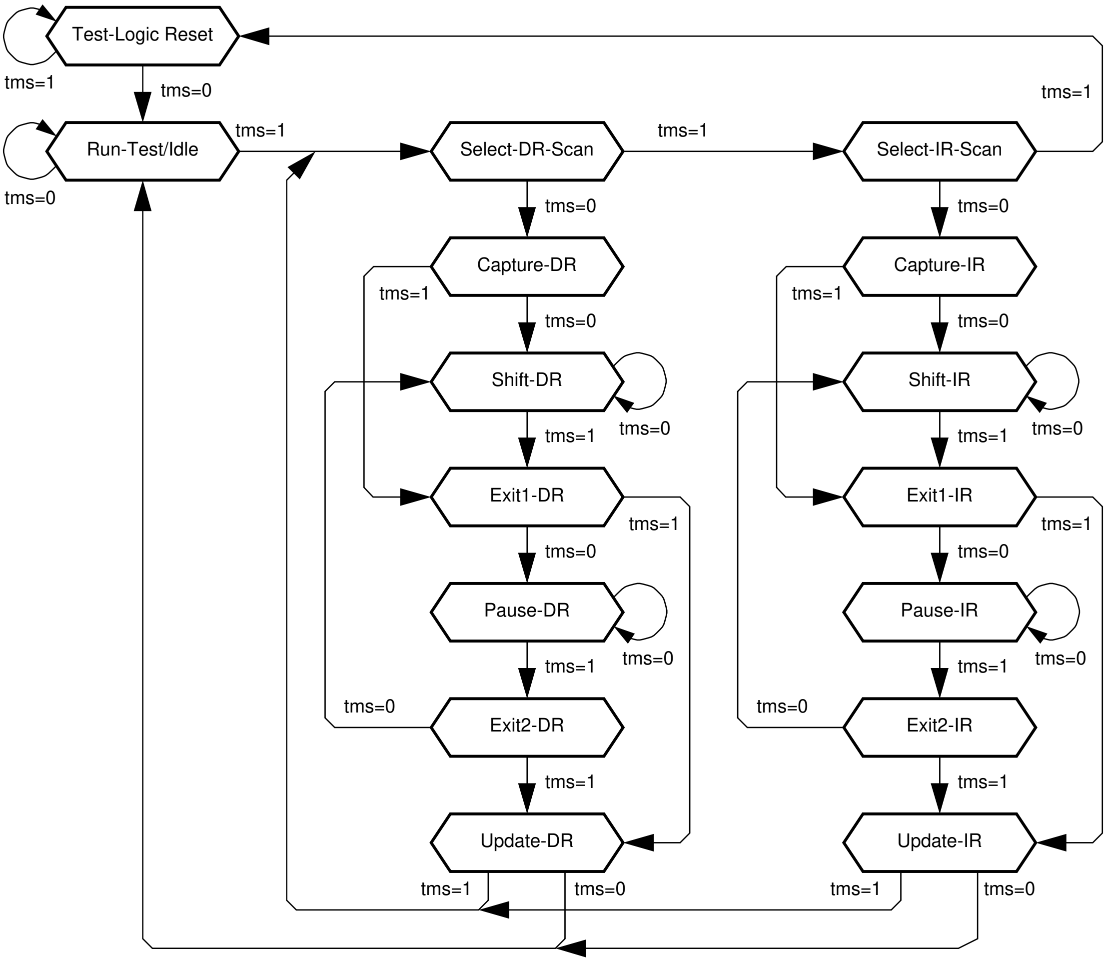


Figure 11: Test Access Port (TAP) Controller Sate Transitions

14.2 Power -On Reset

The boundary-scan interface includes a state machine controller (the TAP controller). Upon power-up, this is automatically forced into the reset state via internal power-on-reset logic. At any other time, the state machine may be reset by holding **TMS** HIGH and applying 5 cycles of **TCK**.

The action of reset is as follows:

System mode is selected (i.e. the boundary scan chain does NOT intercept any of the signals passing between the pads and the core).

IDcode mode is selected. If **TCK** is pulsed, the contents of the ID register will be clocked out of **TDO**.

14.3 Pullup Resistors

The IEEE 1149.1 standard effectively requires that **TDI** and **TMS** should have internal pullup resistors. In order to minimise static current draw, these resistors are NOT fitted to VIDC20. Accordingly, the 4 inputs to the test interface (the above 3 signals plus **TCK**) must all be driven to good logic levels to achieve normal circuit operation.

14.4 Instruction Register

The instruction register is 4 bits in length.

There is no parity bit. The fixed value loaded into the instruction register during the CAPTURE-IR controller state is: 0001.

14.5 Public Instructions

The following public instructions are supported:

Instruction	Binary Code
EXTEST	0000
SAMPLE/PRELOAD	0011
CLAMP	0101
HIGHZ	0111
CLAMPZ	1001
INTEST	1100
IDCODE	1110
BYPASS	1111

In the descriptions that follow, **TDI** and **TMS** are sampled on the rising edge of **TCK** and all output transitions on **TDO** occur as a result of the falling edge of **TCK**.

14.5.1 EXTEST (0000)

The BS (boundary-scan) register is placed in test mode by the EXTEST instruction.

The EXTEST instruction connects the BS register between **TDI** and **TDO**.

When the instruction register is loaded with the EXTEST instruction, all the boundary-scan cells are placed in their test mode of operation.

In the CAPTURE-DR state, inputs from the system pins and outputs from the boundary-scan output cells to the system pins are captured by the boundary-scan cells. In the SHIFT-DR state, the previously captured test data is shifted out of the BS register via the **TDO** pin, whilst new test data is shifted in via the **TDI** pin to the BS register parallel input latch. In the UPDATE-DR state, the new test data is transferred into the BS register parallel output latch. Note that this data is applied immediately to the system logic and system pins. The first EXTEST vector should be clocked into the boundary-scan register, using the SAMPLE/ PRELOAD instruction, prior to selecting INTEST to ensure that known data is applied to the system logic.

14.5.2 SAMPLE/PRELOAD (0011)

The BS (boundary-scan) register is placed in normal (system) mode by the SAMPLE/PRELOAD instruction.

The SAMPLE/PRELOAD instruction connects the BS register between **TDI** and **TDO**.

When the instruction register is loaded with the SAMPLE/PRELOAD instruction, all the boundary-scan cells are placed in their normal system mode of operation.

In the CAPTURE-DR state, a snapshot of the signals at the boundary-scan cells is taken on the rising edge of **TCK**. Normal system operation is unaffected. In the SHIFT-DR state, the sampled test data is shifted out of the BS register via the **TDO** pin, whilst new data is shifted in via the **TDI** pin to preload the BS register parallel input latch. In the UPDATE-DR state, the preloaded data is transferred into the BS register parallel output latch. Note that this data is not applied to the system logic or system pins while the SAMPLE/ PRELOAD instruction is active. This instruction should be used to preload the boundary-scan register with known data prior to selecting the INTEST or EXTEST instructions (see the table below for appropriate guard values to be used for each boundary-scan cell).

14.5.3 CLAMP (0101)

The CLAMP instruction connects a 1 bit shift register (the BYPASS register) between **TDI** and **TDO**.

When the CLAMP instruction is loaded into the instruction register, the state of all output signals is defined by the values previously loaded into the boundary-scan register. A guarding pattern (specified for this device at the end of this section) should be pre-loaded into the boundary-scan register using the SAMPLE/ PRELOAD instruction prior to selecting the CLAMP instruction.

In the CAPTURE-DR state, a logic 0 is captured by the bypass register. In the SHIFT-DR state, test data is shifted into the bypass register via **TDI** and out via **TDO** after a delay of one **TCK** cycle. Note that the first bit shifted out will be a zero. The bypass register is not affected in the UPDATE-DR state.

14.5.4 HIGHZ (0111)

The HIGHZ instruction connects a 1 bit shift register (the BYPASS register) between **TDI** and **TDO**.

When the HIGHZ instruction is loaded into the instruction register, all outputs are placed in an inactive drive state.

In the CAPTURE-DR state, a logic 0 is captured by the bypass register. In the SHIFT-DR state, test data is shifted into the bypass register via **TDI** and out via **TDO** after a delay of one **TCK** cycle. Note that the first bit shifted out will be a zero. The bypass register is not affected in the UPDATE-DR state.

14.5.5 CLAMPZ (1001)

The CLAMPZ instruction connects a 1 bit shift register (the BYPASS register) between **TDI** and **TDO**.

When the CLAMPZ instruction is loaded into the instruction register, all outputs are placed in an inactive drive state, but the data supplied to the disabled output drivers is derived from the boundary-scan cells. The purpose of this instruction is to ensure, during production testing, that each output driver can be disabled when its data input is either a 0 or a 1.

A guarding pattern (specified for this device at the end of this section) should be pre-loaded into the boundary-scan register using the SAMPLE/PRELOAD instruction prior to selecting the CLAMPZ instruction.

In the CAPTURE-DR state, a logic 0 is captured by the bypass register. In the SHIFT-DR state, test data is shifted into the bypass register via **TDI** and out via **TDO** after a delay of one **TCK** cycle. Note that the first bit shifted out will be a zero. The bypass register is not affected in the UPDATE-DR state.

14.5.6 INTEST (1100)

The BS (boundary-scan) register is placed in test mode by the INTEST instruction.

The INTEST instruction connects the BS register between **TDI** and **TDO**.

When the instruction register is loaded with the INTEST instruction, all the boundary-scan cells are placed in their test mode of operation.

In the CAPTURE-DR state, the complement of the data supplied to the core logic from input boundary-scan cells is captured, while the true value of the data that is output from the core logic to output boundary- scan cells is captured. Note that CAPTURE-DR captures the complemented value of the input cells for testability reasons.

In the SHIFT-DR state, the previously captured test data is shifted out of the BS register via the **TDO** pin, whilst new test data is shifted in via the **TDI** pin to the BS register parallel input latch. In the UPDATE-DR state, the new test data is transferred into the BS register parallel output latch. Note that this data is applied immediately to the system logic and system pins. The first INTEST vector should be clocked into the boundary-scan register, using the SAMPLE/PRELOAD instruction, prior to selecting INTEST to ensure that known data is applied to the system logic.

Single-step operation is possible using the INTEST instruction.

14.5.7 IDCODE (1110)

The IDCODE instruction connects the device identification register (or ID register) between **TDI** and **TDO**. The ID register is a 32-bit register that allows the manufacturer, part number and version of a component to be determined through the TAP.

When the instruction register is loaded with the IDCODE instruction, all the boundary-scan cells are placed in their normal (system) mode of operation.

In the CAPTURE-DR state, the device identification code (specified at the end of this section) is captured by the ID register. In the SHIFT-DR state, the previously captured device identification code is shifted out of the ID register via the **TDO** pin, whilst data is shifted in via the **TDI** pin into the ID register. In the UPDATE-DR state, the ID register is unaffected.

14.5.8 BYPASS (1111)

The BYPASS instruction connects a 1 bit shift register (the BYPASS register) between **TDI** and **TDO**.

When the BYPASS instruction is loaded into the instruction register, all the boundary-scan cells are placed in their normal (system) mode of operation. This instruction has no effect on the system pins.

In the CAPTURE-DR state, a logic 0 is captured by the bypass register. In the SHIFT-DR state, test data is shifted into the bypass register via **TDI** and out via **TDO** after a delay of one **TCK** cycle. Note that the first bit shifted out will be a zero. The bypass register is not affected in the UPDATE-DR state.

14.6 Test Data Registers

Figure 12: Boundary Scan Block Diagram illustrates the structure of the boundary scan logic.

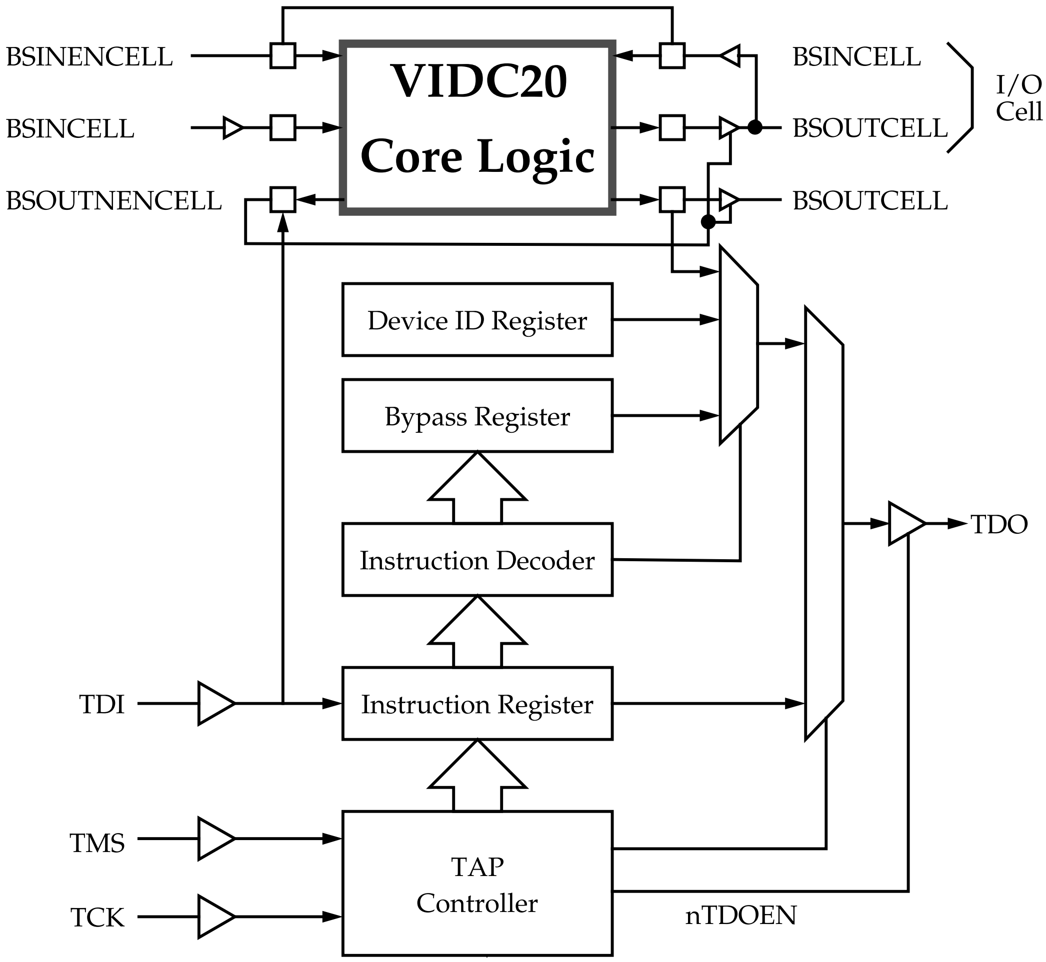


Figure 12: Boundary Scan Block Diagram

14.6.1 Bypass Register

Purpose: This is a single bit register which can be selected as the path between **TDI** and **TDO** to allow the device to be bypassed during boundary-scan testing.

Length: 1 bit

Operating Mode: When the BYPASS instruction is the current instruction in the instruction register, serial data is transferred from **TDI** to **TDO** in the SHIFT-DR state with a delay of one **TCK** cycle.

There is no parallel output from the bypass register.

A logic 0 is loaded from the parallel input of the bypass register in the CAPTURE-DR state.

14.6.2 VIDC20 Device Identification (ID) Code Register

Purpose: This register is used to read the 32-bit device identification code. No programmable supplementary identification code is provided.

Length: 32 bits

The format of the ID register is as follows:

31 — 28	27 — 12	11 — 1	0
Version	**Part Number**	**Manufacturer Identity**	**1**

Please contact your supplier for the correct Device Identification Code.

Operating Mode: When the IDCODE instruction is current, the ID register is selected as the serial path between **TDI** and **TDO**.

There is no parallel output from the ID register.

The 32-bit device identification code is loaded into the ID register from its parallel inputs during the CAPTURE-DR state.

14.6.3 VIDC20 Boundary Scan (BS) Register

Purpose: The BS register consists of a serially connected set of cells around the periphery of the device, at the interface between the core logic and the system input/output pads. This register can be used to isolate the core logic from the pins and then apply tests to the core logic, or conversely to isolate the pins from the core logic and then drive or monitor the system pins.

Operating modes: The BS register is selected as the register to be connected between **TDI** and **TDO** only during the SAMPLE/PRELOAD, EXTEST and INTEST instructions. Values in the BS register are used, but are not changed, during the CLAMP and CLAMPZ instructions.

In the normal (system) mode of operation, straight-through connections between the core logic and pins are maintained and normal system operation is unaffected.

In TEST mode (i.e. when either EXTEST or INTEST is the currently selected instruction), values can be applied to the core logic or output pins independently of the actual values on the input pins and core logic outputs respectively. On the VIDC20 all of the boundary scan cells include an update register and thus all of the pins can be controlled in the above manner. Additional boundary-scan cells are interposed in the scan chain in order to control the enabling of tristateable buses.

The correspondence between boundary-scan cells and system pins, system direction controls and system output enables is as shown in *Table 10: Boundary Scan Signals & Pins*. The cells are listed in the order in which they are connected in the boundary-scan register, starting with the cell closest to **TDI**. All boundary-scan register cells at input pins can apply tests to the on-chip core logic.

The values stored in the BS register after power-up are not defined. Similarly, the values previously clocked into the BS register are not guaranteed to be maintained across a Boundary Scan reset.

14.7 Boundary Scan Interface Signals

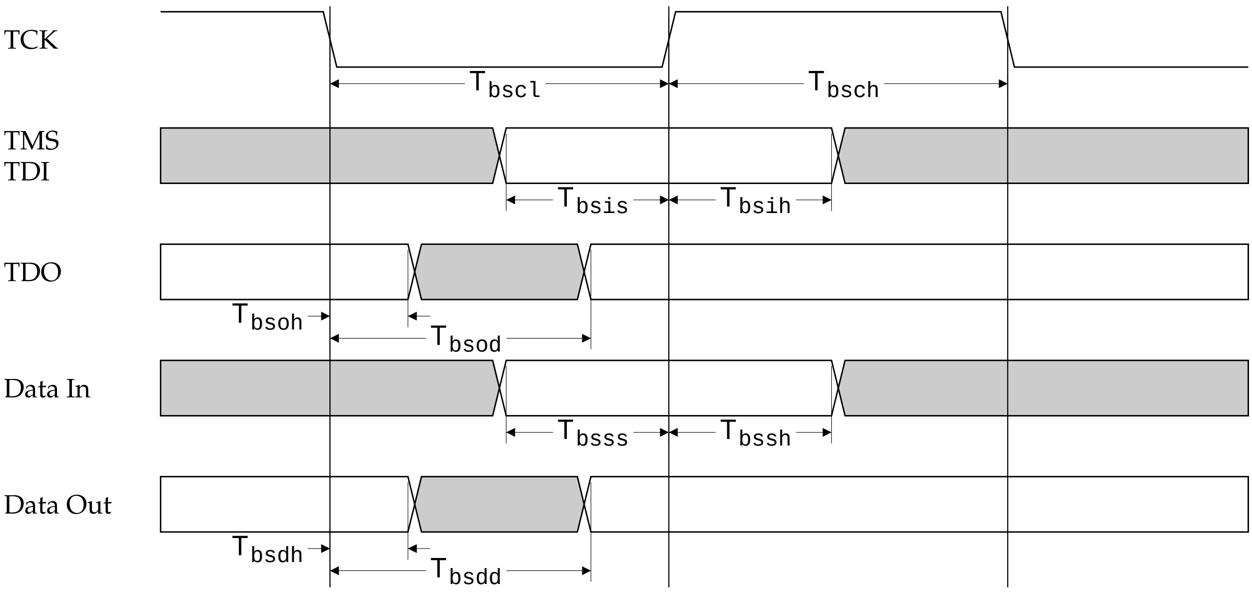


Figure 13: Boundary Scan General Timing

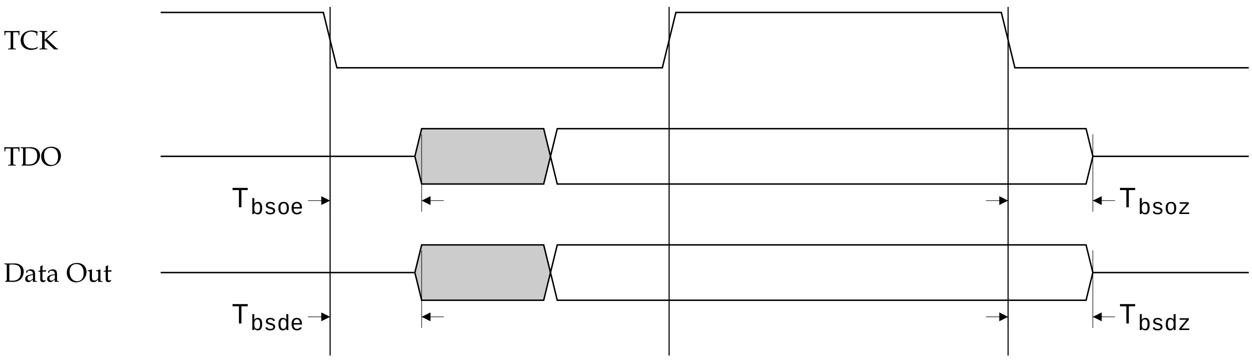


Figure 14: Boundary Scan Tri-state Timing

Figure 15:

Symbol	Parameter	Min	Typ	Max	Units	Notes
Tbscl	**TCK** low period	50			ns	
Tbsch	**TCK** high period	50			ns	

Table 9: VIDC20 Boundary Scan Interface Timing

Symbol	Parameter	Min	Typ	Max	Units	Notes
Tbsis	**TDI**,**TMS** setup to [TCr]	10			ns	
Tbsih	**TDI**,**TMS** hold from [TCr]	10			ns	
Tbsoh	**TDO** hold time	5			ns	1
Tbsod	TCf to **TDO** valid			40	ns	1
Tbsss	I/O signal setup to [TCr]	5			ns	4
Tbssh	I/O signal hold from [TCr]	20			ns	4
Tbsdh	data output hold time	5			ns	5
Tbsdd	TCf to data output valid			40	ns	
Tbsoe	**TDO** enable time	5			ns	1,2
Tbsoz	**TDO** disable time			40	ns	1,3
Tbsde	data output enable time	5			ns	5,6
Tbsdz	data output disable time			40	ns	5,7

Table 9: VIDC20 Boundary Scan Interface Timing

Notes:

1. Assumes a 25pF load on **TDO**. Output timing derates at 0.072ns/pF of extra load applied.
2. **TDO** enable time applies when the TAP controller enters the Shift-DR or Shift-IR states.
3. **TDO** disable time applies when the TAP controller leaves the Shift-DR or Shift-IR states.
4. For correct data latching, the I/O signals (from the core and the pads) must be setup and held with respect to the rising edge of **TCK** in the CAPTURE-DR state of the SAMPLE/PRELOAD, INTEST and EXTEST instructions.
5. Assumes that the data outputs are loaded with the AC test loads (see AC parameter specification).
6. Data output enable time applies when the boundary scan logic is used to enable the output drivers.
7. Data output disable time applies when the boundary scan is used to disable the output drivers.

No.	Signal	Cell Name	Pin	Type
from tdi				
1	DIN[0]	din[0]	46	IND
2	DIN[1]	din[1]	47	IND
3	DIN[2]	din[2]	48	IND
4	DIN[3]	din[3]	49	IND
5	DIN[4]	din[4]	50	IND
6	DIN[5]	din[5]	51	IND
7	DIN[6]	din[6]	53	IND
8	DIN[7]	din[7]	54	IND
9	DIN[8]	din[8]	55	IND
10	DIN[9]	din[9]	56	IND
11	DIN[10]	din[10]	57	IND
12	DIN[11]	din[11]	58	IND
13	DIN[12]	din[12]	60	IND
14	DIN[13]	din[13]	61	IND
15	DIN[14]	din[14]	62	IND
16	DIN[15]	din[15]	63	IND
17	DIN[16]	din[16]	64	IND
18	DIN[17]	din[17]	65	IND
19	DIN[18]	din[18]	67	IND
20	DIN[19]	din[19]	68	IND
21	DIN[20]	din[20]	69	IND
22	DIN[21]	din[21]	70	IND
23	DIN[22]	din[22]	71	IND
24	DIN[23]	din[23]	72	IND
25	DIN[24]	din[24]	73	IND
26	DIN[25]	din[25]	74	IND
27	DIN[26]	din[26]	75	IND
28	DIN[27]	din[27]	76	IND
29	DIN[28]	din[28]	80	IND
30	DIN[29]	din[29]	81	IND
31	DIN[30]	din[30]	82	IND
32	DIN[31]	din[31]	83	IND
33	DIN[32]	din[32]	84	IND
34	DIN[33]	din[33]	85	IND
35	DIN[34]	din[34]	86	IND
36	DIN[35]	din[35]	88	IND
37	DIN[36]	din[36]	89	IND
38	DIN[37]	din[37]	90	IND
39	DIN[38]	din[38]	91	IND
40	DIN[39]	din[39]	92	IND
41	DIN[40]	din[40]	93	IND
42	DIN[41]	din[41]	94	IND
43	DIN[42]	din[42]	96	IND
44	DIN[43]	din[43]	97	IND
45	DIN[44]	din[44]	100	IND
46	DIN[45]	din[45]	101	IND
47	DIN[46]	din[46]	103	IND
48	DIN[47]	din[47]	104	IND
49	DIN[48]	din[48]	105	IND
50	DIN[49]	din[49]	106	IND
51	DIN[50]	din[50]	107	IND
52	DIN[51]	din[51]	108	IND
53	DIN[52]	din[52]	109	IND
54	DIN[53]	din[53]	110	IND
55	DIN[54]	din[54]	111	IND
56	DIN[55]	din[55]	112	IND
57	DIN[56]	din[56]	113	IND
58	DIN[57]	din[57]	114	IND
59	DIN[58]	din[58]	115	IND
60	DIN[59]	din[59]	117	IND
61	DIN[60]	din[60]	118	IND
62	DIN[61]	din[61]	119	IND
63	DIN[62]	din[62]	120	IND
64	DIN[63]	din[63]	121	IND
65	BUSCLK	busclk	123	INC
66	RCLK	rclk	124	INC
67	HCLK	hclk	125	INC
68	VCLK	vclk	126	INC
69	nQCLK	Nqclk	132	OUT
70	QCLK	qclk	133	INC
71	VnC	vNc	134	OUT
72	nSNDRQ	Nsndrq	135	OUT
73	nVIDRQ	Nvidrq	136	OUT
74	nSNDAK	Nsndak	138	INC
75	nVIDAK	Nvidak	139	INC
76	nPROG	Nprog	140	INC
77	SINK	sink	141	IND
78	QSF	qsf	142	IND
79	nRST	Nrst	143	IND
80	SCLK	sclk	144	INC
81	FLYBK	flybk	1	OUT
82	WS	ws	2	OUT
83	SDCLK	sdclk	3	OUT
84	SDO	sdo	4	OUT
85	EREG1	ereg1	5	OUT
86	EREG0	ereg0	6	OUT
87	ESEL1	esel1	7	IND
88	ESEL0	esel0	9	IND
89	ED[0]	ed[0]	12	OUT
90	ED[1]	ed[1]	13	OUT
91	ED[2]	ed[2]	14	OUT
92	ED[3]	ed[3]	16	OUT
93	ED[4]	ed[4]	17	OUT
94	ED[5]	ed[5]	18	OUT
95	ED[6]	ed[6]	19	OUT
96	ED[7]	ed[7]	20	OUT
97	ECLK	eclk	22	OUT
98	SIREF	siref	23	OE0
99	HSYNC	hsync	31	OUT
100	VSYNC	vsync	32	OUT
to tdo				

Table 10: Boundary Scan Signals & Pins

Boundary Scan Table Key:

INC	Clock Input
IND	Data Input
OUT	Output pad
OE0	Active-Low ouput enable

15.0 Packaging

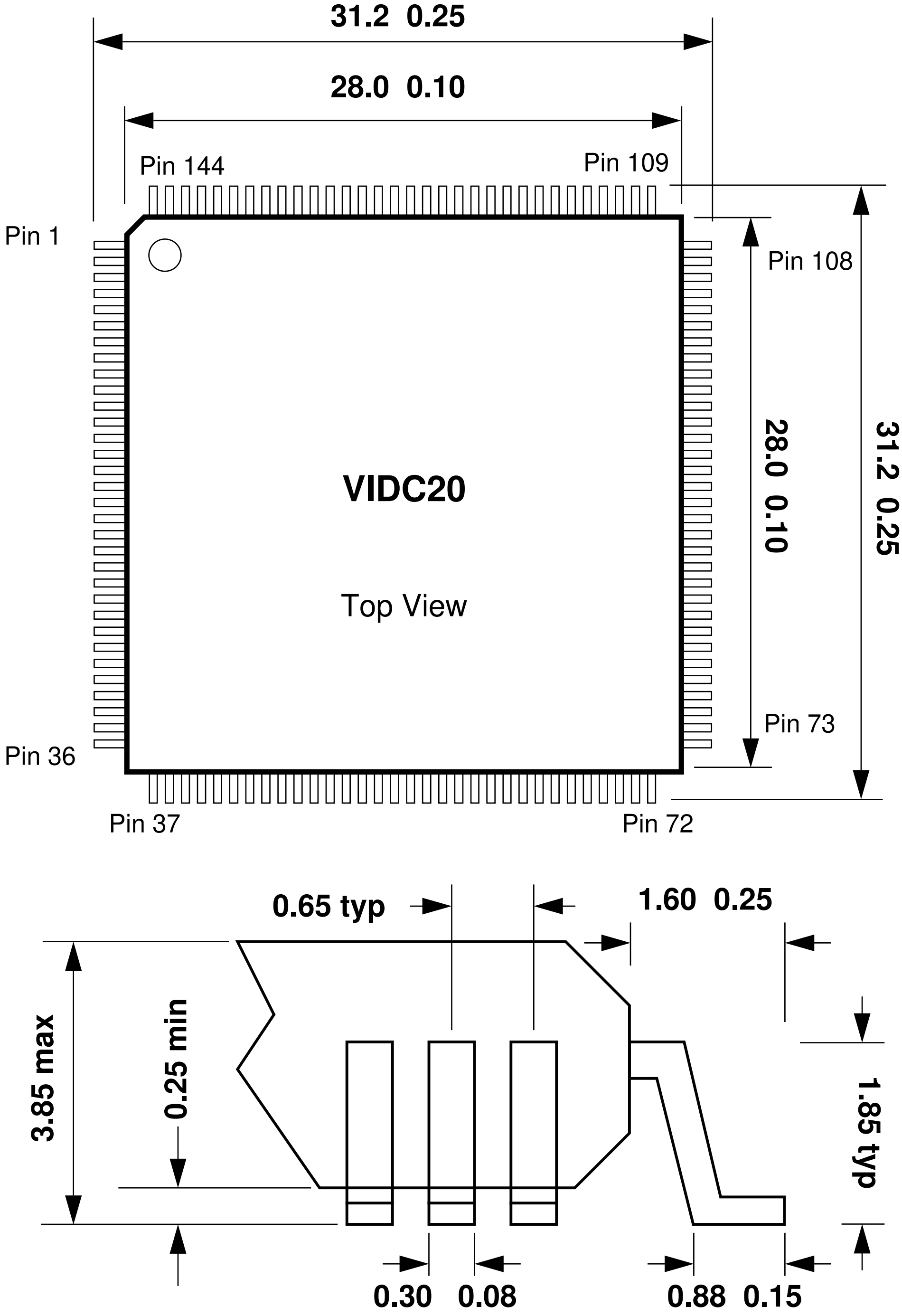


Figure 16: VIDC20 Packaging Layout and Dimensions

16.0 Pinout

Pin	Name	Pin	Name	Pin	Name	Pin	Name
1	FLYBK	33	VIREF	65	DIN[24]	97	DIN[52]
2	WS	34	VDD_ANALOG	66	DIN[25]	98	DIN[53]
3	SDCLK	35	BOUT	67	DIN[26]	99	DIN[54]
4	SDO	36	GOUT	68	DIN[27]	100	DIN [55]
5	EREG[1]	37	ROUT	69	VSS	101	DIN[56]
6	EREG[0]	38	VSS_ANALOG	70	VDD	102	DIN[57]
7	ESEL[1]	39	VSS	71	DIN[28]	103	DIN[58]
8	ESEL[0]	40	VDD	72	DIN[29]	104	DIN[59]
9	VDD	41	DIN[0]	73	DIN[30]	105	DIN[60]
10	VSS	42	DIN[1]	74	DIN[31]	106	DIN[61]
11	ED[7]	43	DIN[2]	75	DIN[32]	107	DIN[62]
12	ED[6]	44	DIN[3]	76	DIN[33]	108	DIN[63]
13	ED[5]	45	DIN[4]	77	DIN[34]	109	BUSCLK
14	ED[4]	46	DIN[5]	78	DIN[35]	110	RCLK
15	ED[3]	47	DIN[6]	79	DIN[36]	111	HCLK
16	ED[2]	48	DIN[7]	80	DIN[37]	112	VCLKI
17	ED[1]	49	DIN[8]	81	DIN[38]	113	VCLKO
18	ED[0]	50	DIN[9]	82	DIN[39]	114	VSS
19	VDD	51	DIN[10]	83	DIN[40]	115	PCOMP
20	ECLK	52	DIN[11]	84	DIN[41]	116	VDD
21	VSS	53	DIN[12]	85	DIN[42]	117	NQCLK
22	SIREF	54	DIN[13]	86	DIN[43]	118	QCLK
23	VDD_SOUND	55	DIN[14]	87	VDD	119	VNC
24	LS	56	DIN[15]	88	VSS	120	NSNDRQ
25	RS	57	DIN[16]	89	DIN[44]	121	NVIDRQ
26	VSS_SOUND	58	DIN[17]	90	DIN[45]	122	NSNDAK
27	HS	59	DIN[18]	91	DIN[46]	123	NVIDAK
28	VS	60	DIN[19]	92	DIN[47]	124	NPROG
29	TCK	61	DIN[20]	93	DIN[48]	125	SINK
30	TMS	62	DIN[21]	94	DIN[49]	126	QSF
31	TDI	63	DIN[22]	95	DIN[50]	127	NRST
32	TDO	64	DIN[23]	96	DIN[51]	128	SCLK

Table 11: Pinout