VV6404 & STV0673

USB & Parallel Port Digital Camera Chipset

DESCRIPTION

These USB/Parallel port camera chipsets from STMicroelectronics are at the heart of a variety of products which have proven to be highly successful in a demanding marketplace. Supported by comprehensive reference designs, technical backup and fully-featured software drivers, STMicroelectronics offers camera manufacturers the opportunity to benefit from rapid time to market with a product of proven quality.

Together with the support circuitry detailed in the reference design, the VV6404 sensor and STV0673 (CPIA1.5) form the basis of a CIF-resolution USB or Parallel Port camera. The sensor and co-processor perform the key functions of image sensing, digital video processing, video compression and USB bus interfacing.

The VV6404 is a ColorMOS^{TM} digital CMOS sensor which delivers outstanding picture quality. This CIF-resolution sensor has been created specifically to meet the standards required for personal video communications.

The STV0673 (CPIA1.5) co-processor receives image data from the sensor, which it processes, formats and passes to the USB port. It incorporates a digital video processor engine, which performs automatic exposure, automatic gain control and automatic white balance, together with colour matrixing, gamma correction and aperture correction. This data is passed to a proprietary video compression which delivers impressive frame rates with minimum impact on image quality. The USB interface supports USB isochronous data transfer mode, ensuring access to guaranteed bandwidth at all time, irrespective of how many peripherals are then added.

The Parallel port interface makes use of the ECP extended capabilities printer ports, and will use DMA transfer when available to maximise bandwidth. It will automatically select the best available transfer mode.

The chipset is backed by a fully-featured driver which provides a host of user-definable settings for optimum camera setup. The user interface supports a degree of customisation.

STMicroelectronics offers a range of support services to guarantee product quality, including test specifications and test software.

FEATURES

- Real-time video - up to 30fps CIF
- USB or ECP Parallel Port interface
- ColorMOS^{TM} sensor
- USB Power management compliant
- Proprietary hardware compression
- ECP DMA support for Parallel Port
- Automatic black level calibration
- Full VfW and TWAIN driver support

APPLICATIONS

- PC Camera
- Biometric identification
- Toys and games
- Image capture systems

SPECIFICATIONS

Pixel resolution	352 x 288 (CIF)
Array size	1/3" lens format
Min. illumination	15 lux
Exposure control	Automatic (to 146000:1)
Gain control	Automatic (to +20dB)
Signal/Noise ratio	42d
Supply voltage	5.0v DC +/– 5%
USB Compatibility	USB Specification V1.0 Meets full power management requirements
Parallel Port Compatibility	ECP DMA, ECP, or standar SPP parallel ports Auto-selecting
Supply current	<100mA
Operating temperature (ambient)	0^oC - 40^oC (for extended temp. info please contact STMicroelectronics)
Package type	Sensor: 48LCC STV0673: 100QFP

Table of Contents

1. Introduction

1.1 General description

The Colour Processor Interface ASIC 1.5 (CPiA 1.5) is a new revision of the CPiA ASIC that utilises a smaller physical package, while maintaining most of the features of the original CPiA device.

It is a digital video processor requiring only a single 4Mbit DRAM and minimum of passive support components to provide a complete PC peripheral video capture system.

CPiA 1.5 accepts raw digital video data from an STMicroelectronics CIF format CMOS sensor and is capable of transferring the resulting YUV video data to a host PC over either USB or Parallel Port interfaces at rates up to 30 frames per second.

The CPiA 1.5 architecture consists of three conceptually separate functional blocks: the Video Processor (VP), the Video Compressor (VC) and the Control Processor (CP).

- The VP controls the VV6404 sensors and processes the raw pixel data into CIF or QCIF YUV images
- This YUV data is compressed and transferred to the Host by the VC.
- The CP is responsible for coordinating system operation, responding to host requests and commands as well as performing automatic camera exposure control and colour balance.

2. Functional description

2.1 VV6404 sensor

The VV6404 sensor is a CIF format CMOS image sensor capable of outputing digital pixel data at frame rates, of upto 30 frames per second. It has a colour filter applied over the sensor array.

The 356 x 292 pixel sensor has an on-chip 8-bit analogue to digital converter and is designed to interface directly to the CPiA 1.5

An 8-bit pixel value is transmitted across the 4 wire tri-stateable databus as series pair of 4-bit nibbles, most significant nibble first. Along with the pixel data, codes representing the start and end frames and the start and end of lines are embedded within the video data stream to allow the video processor to synchronise with video data the camera module is generating.

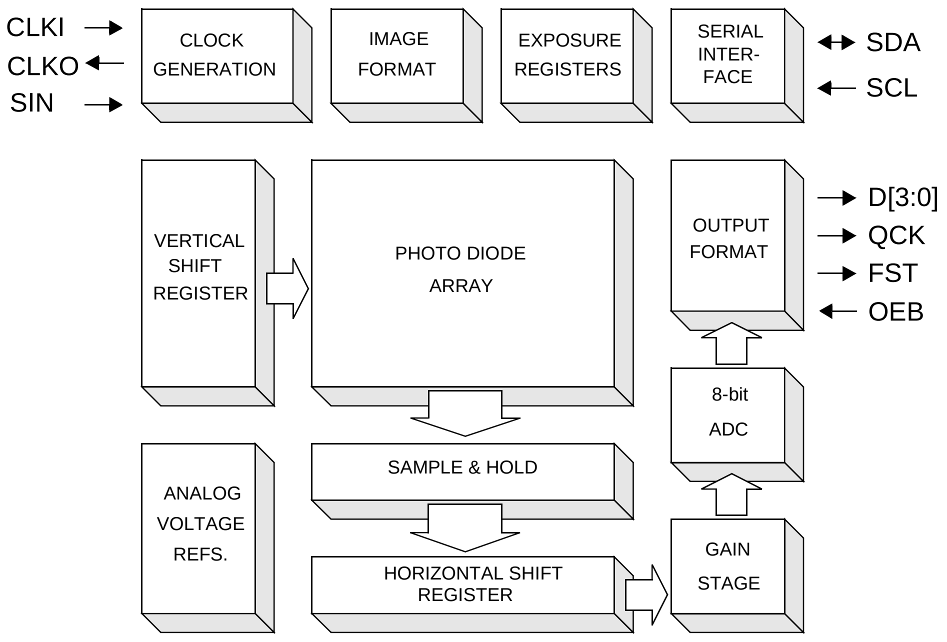


Figure 1 : Block Diagram of VV6404 Image Sensor

2.1.1 Image Format

The output image format is CIF (352 x 288 pixel array). To provide the colour co-processor with the extra information it needs for interpolation at the edges of the VV6404 pixel array, an additional border 2 pixels deep on all 4 sides of the array is enabled under serial interface control. The resulting image size of 356 x 292 pixels is the default power up state for this sensor. .

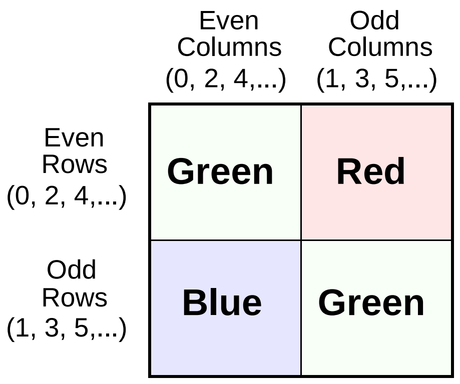


Figure 2 : Bayer Colourisation Pattern.

2.1.2 Digital Video Interface Format

The video interface consists of a unidirectional, tri-stateable 4-wire databus. The nibble transmission is synchronised to the rising edge of the system clock.

Read-out Order	Progressive Scan (Non-interlaced)
Form of encoding	Uniformly quantised, PCM, 8 bits per sample
Correspondence between video signal levels and quantisation levels:	Internally valid pixel data is clipped to ensure that 00_H and FF_H values do not occur when pixel data is being output on the data bus. This gives 254 possible values for each pixel (1 - 254). The video black level corresponds to code 16.

Table 1 : Video encoding parameters

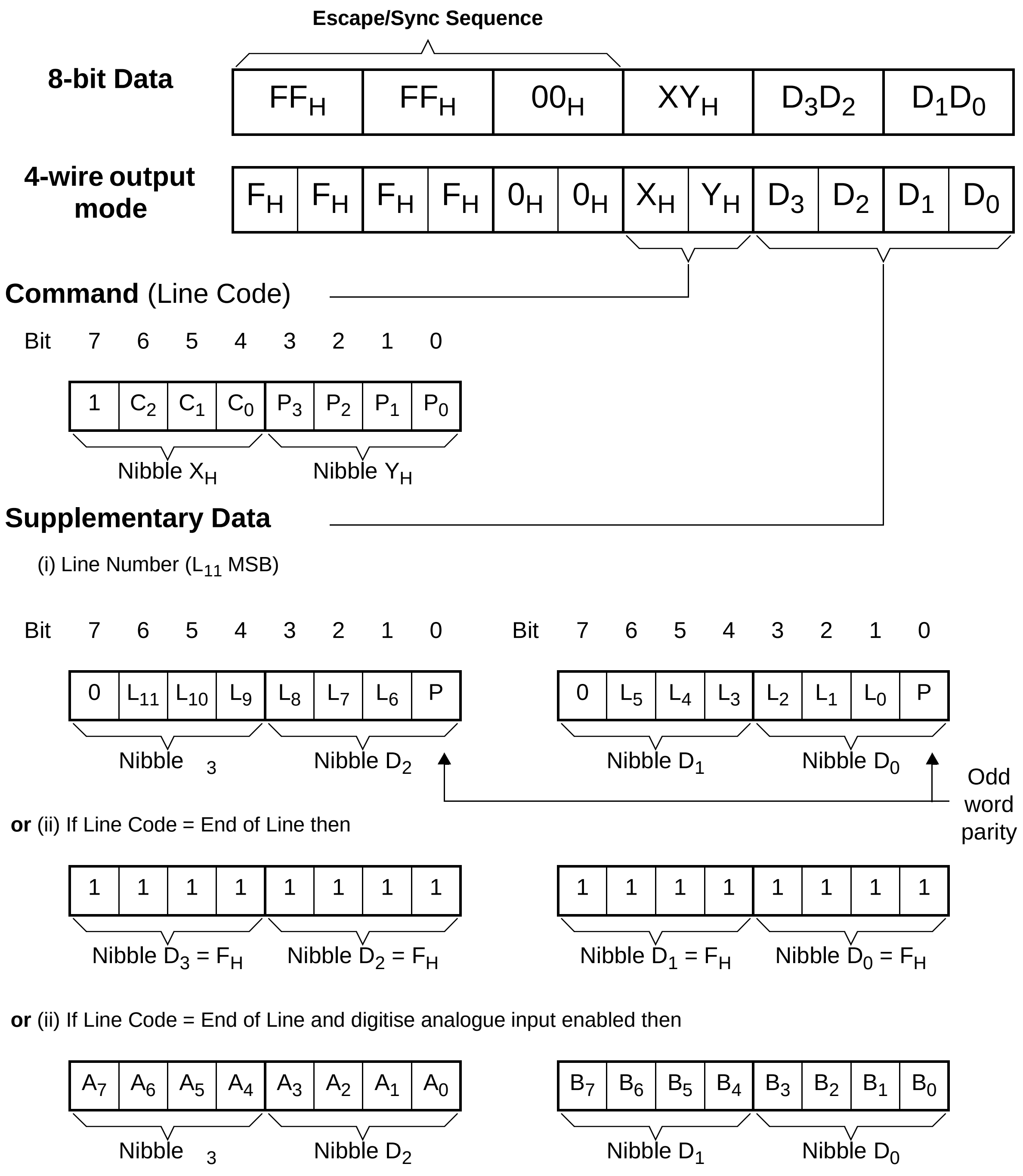
Digital video data is 8 bits per sample, transmitted as serial pairs of parallel 4-bit nibbles (most significant nibble first) on 4 wires.

Multiplexed with the sampled pixel data is control information including both video timing references and sensor status/configuration data. Video timing reference information takes the form of field start characters, line start characters, end of line characters and a line counter.

Where hexadecimal values are used, they are indicated by a subscript H, such as FF_H; other values are decimal.

Line Code	Nibble X_H (1 C_2 C_1 C_0)	Nibble Y_H (P_3 P_2 P_1 P_0)
End of Line	1000_2 (8_H)	0000_2 (0_H)
Blank Line (BL)	1001_2 (9_H)	1101_2 (D_H)
Black line (BK)	1010_2 (A_H)	1011_2 (B_H)
Visible Line (VL)	1011_2 (B_H)	0110_2 (6_H)
Start of Even Field (SOEF)	1100_2 (C_H)	0111_2 (7_H)
End of Even Field (EOEF)	1101_2 (D_H)	1010_2 (A_H)
Start of Odd Field (SOOF)	1110_2 (E_H)	1100_2 (C_H)
End of Odd Field (EOOF)	1111_2 (F_H)	0001_2 (1_H)

Table 2 : Embedded Line Codes



Escape/Sync Sequence
8-bit Data
FF_H FF_H 00_H XY_H D_3D_2 D_1D_0
4-wire output mode
F_H F_H F_H F_H 0_H 0_H X_H Y_H D_3 D_2 D_1 D_0
Command (Line Code)
Bit 7 6 5 4 3 2 1 0
1 C_2 C_1 C_0 P_3 P_2 P_1 P_0
Nibble X_H
Nibble Y_H
Supplementary Data
(i) Line Number (L_{11} MSB)
Bit 7 6 5 4 3 2 1 0
0 L_{11} L_{10} L_9 L_8 L_7 L_6 P
Nibble 3
Nibble D_2
Bit 7 6 5 4 3 2 1 0
0 L_5 L_4 L_3 L_2 L_1 L_0 P
Nibble D_1
Nibble D_0
Odd word parity
or (ii) If Line Code = End of Line then
1 1 1 1 1 1 1 1
Nibble $D_3 = F_H$
Nibble $D_2 = F_H$
1 1 1 1 1 1 1 1
Nibble $D_1 = F_H$
Nibble $D_0 = F_H$
or (ii) If Line Code = End of Line and digitise analogue input enabled then
A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0
Nibble 3
Nibble D_2
B_7 B_6 B_5 B_4 B_3 B_2 B_1 B_0
Nibble D_1
Nibble D_0

Figure 3 : Embedded Control Sequence

2.1 Video Processor Module

The CPiA 1.5 video processor module provides CIF-format 4:2:2-sampled digital video to the VC module at frame rates up to 30 frames per second and also interfaces directly to the 6404 image sensors. Using a 9-wire cable and minimal external components, the interface incorporates:

1. A 4-wire data bus SDATA[3:0] for receiving both video data and embedded timing references.
2. A 2-wire serial interface SSDA,SSCL for controlling the camera.
3. The clock for camera module SCLK.
4. 3V3 and 0V power lines. CPiA 1.5 is not required to provide power directly to the camera; camera power is derived from the system power supply.

The simplified block diagram shown below highlights the key functional blocks within CPiA 1.5's VP module.

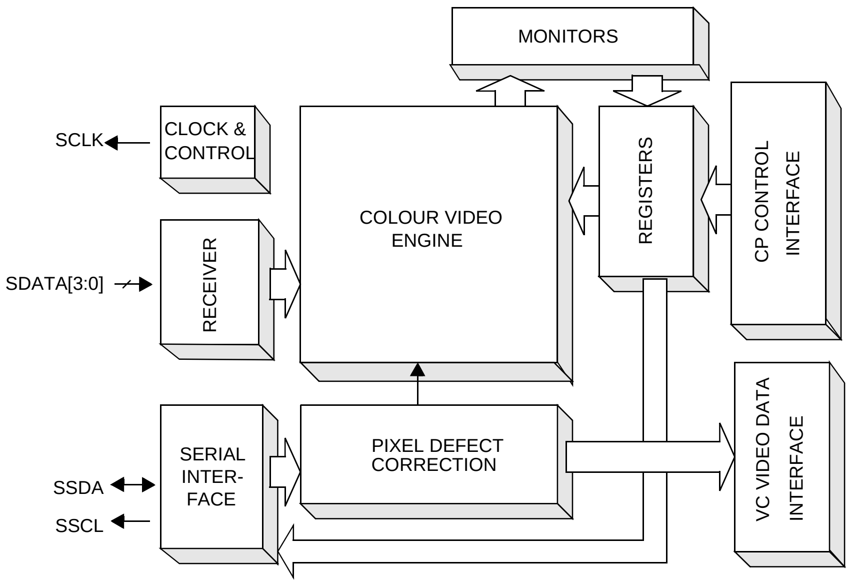


Figure 4 : Block Diagram of CPiA 1.5 Video Processor Module

CPiA 1.5 provides a master clock SCLK to the camera module. Each 8-bit pixel value generated by the camera is transmitted across the 4 wire databus SDATA[3:0] as a pair of sequential 4-bit nibbles, most significant nibble first. Codes representing the start and end frames and the start and end of lines are embedded within the video pixel data stream to allow the receiver to synchronise with the video data which the camera module is generating.

The video processing engine performs these basic functions on incoming data: full colour restoration at each pixel site from Bayer-patterned input data, matrixing/gain on each colour channel for colour purity, aperture correction for image clarity, gamma correction, and colour space conversion (including hue and saturation control) from RGB to YCbCr.

Image statistic monitors gather data required by the CP module for end-of-frame housekeeping tasks such as exposure control and colour balance.

The 2-wire camera serial interface provides complete control over how the sensor is setup and run.

Camera exposure and gain values are programmed via this interface.
The following table and diagram illustrate the camera power up sequence.

PU0	System Power Up
PU1	Sensor Internal-on Reset Triggers, the sensor enters low power mode and SDATA[3:0] is set to F_H.
PU2	CPiA 1.5 internally releases video processing modules from reset. NOTE - this event is under the control of the Control Processor, and does not occur until the host has requested video from the camera.
PU3	CPiA 1.5 enables the sensor clock, SCLK.
PU4-PU5	At least 16 SCLK clock periods after SCLK has been enabled CPiA 1.5 sends a “Soft-Reset” command to the sensor via the serial interface. This ensures that if a sensor is present then it is in low-power mode.
PU6	On detecting 32 consecutive F_H values on SDATA [3:0], CPiA 1.5 detects the camera
PU7-PU8	If present, CPiA 1.5 uploads the sensor defect map from camera head E^2PROM.
PU13-PU4	At least 16 SCLK clock periods after SCLK has been enabled, CPiA 1.5 sends an “Exit Low-Power Mode” command to the sensor via the serial interface. This initiates the sensors 4 frame start sequence.
PU15-PU16	One frame of alternating 9_H & 6_H data on SDATA[3:0] for the CPiA 1.5 to determine the best sampling phase for the nibble data SDATA[3:0].
PU17-PU18	4 Frames after the “Exit Low-Power Mode” serial comms, the sensor starts to output valid video data.

Table 3 : Power on states

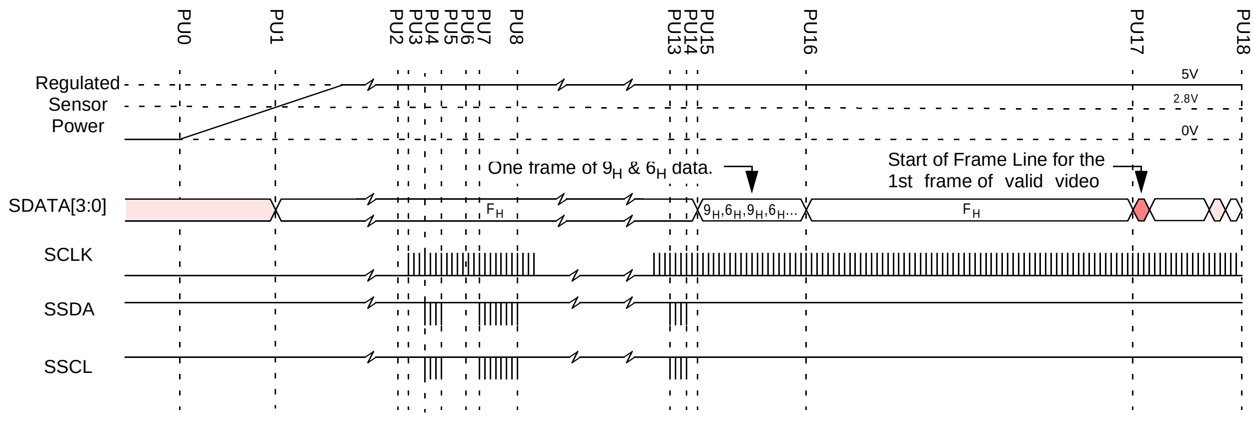


Figure 5 : Camera Head Interface Behaviour up to and including first valid video data

2.2 Video Compressor Module

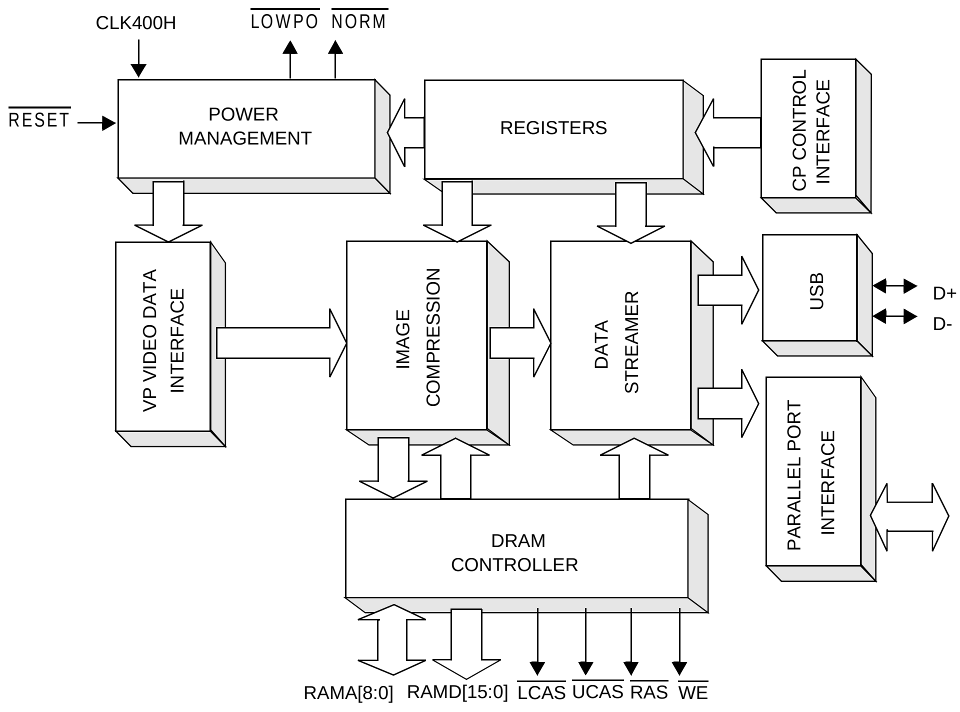


Figure 6 : Block Diagram of CPiA 1.5 Video Compressor Module

2.2.1 Power Management

The power management block is primarily responsible for low-level control of system power, clocks and reset sequences to ensure full compliance with the USB specification and power saving modes of operation. This module also includes a watchdog reset ensuring, for example, the system always returns to a safe state if power-failure or any other event has caused CPiA 1.5 to encounter an unknown and fatal error.

One of the main differences between the original CPiA and the CPiA 1.5, is that the CPiA 1.5 has been characterised to operate from a 3V3 supply to simplify system design. In addition the die-shrink has meant that the CPiA 1.5 reference design always draws less than 100mA even in high-power mode, so some of the CPiA power management circuit are no longer necessary.

When CPiA 1.5 is designed into a product providing full USB power consumption compliance, the system must provide CPiA 1.5 with VDD=3V3, a switched VDD=3V3A The separate VDDU (for internal USB differential pads) can be connected to 3V3.

If a 3V3 DRAM is used then the entire interface circuit can be powered via an inexpensive 3V3 linear regulator directly from the 5V USB Power inputs. There is no longer any requirement for the DC-DC switcher used in the previous CPiA design.

The 6404 sensor is supplied with VDDA directly from the USB 5V via a FET, to ensure that the sensor remains within DC specs. It is turned off when the CPiA 1.5 goes into suspend. Since the CPiA 1.5 still enumerates as a high power USB device by default, the input USB 5V can be guaranteed as a minimum of 4.6v.

NOTE: It is possible to configure the CPiA 1.5 to enumerate as a low power USB device, by tieing HP_DATA[7] to 0V. If this is done, it is necessary to include a 5V DC-DC switcher for the sensor VDDA to guarantee that it remains within spec.

VDD and VDDU must always be permanently connected to 3V3 system power supply. The switched 3V3A power line is typically supplied using a power FET external to CPiA 1.5, as is the switched VDDA for the sensor VDD. The power management module provides two outputs for driving the FET gates and hence is capable of enabling or disabling power to system-level components; $\overline{\text{LOPOW}}$ and $\overline{\text{NORM}}$.

NOTE: If CPiA 1.5 is used in a target application where only the parallel port interface is required and slightly higher module power consumption is permitted, a single, constant 5V supply can be safely fed to VDD, VDDA and VDDU.

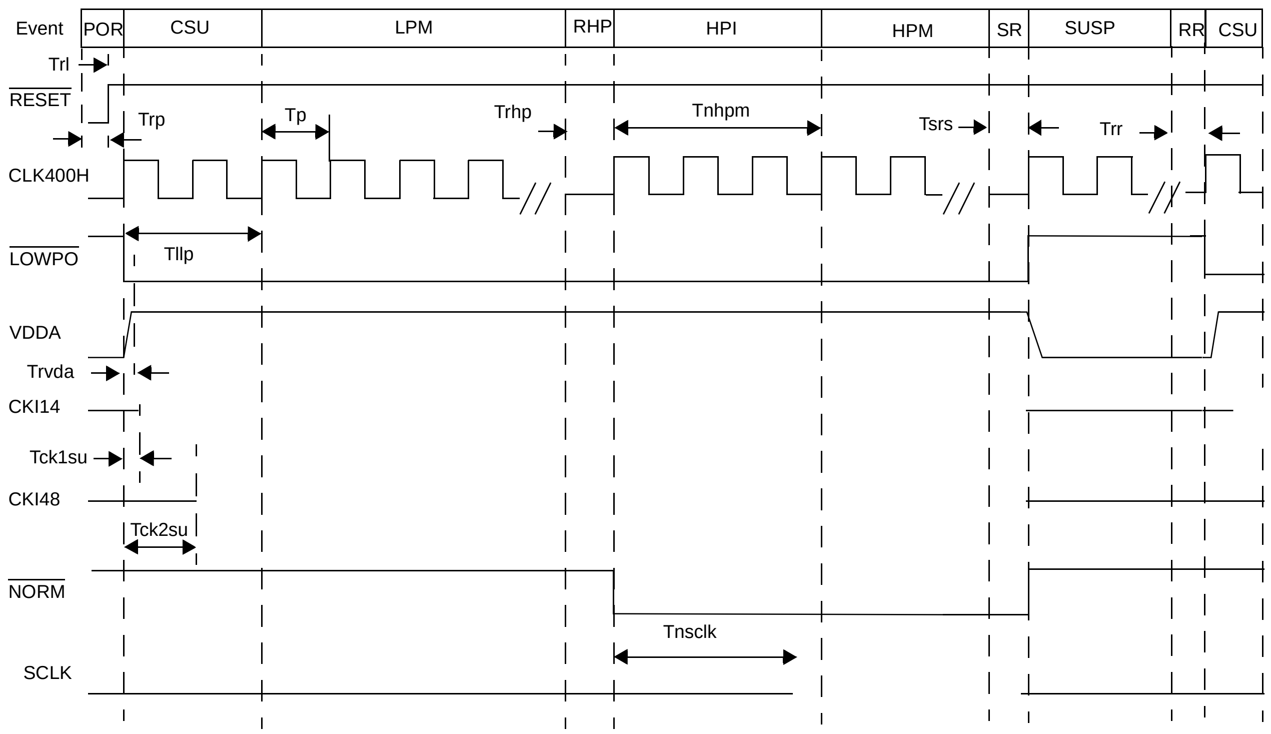
$\overline{\text{LOPOW}}$ is asserted when CPiA 1.5 is ready for VDDA to be powered and once power has been applied puts CPiA 1.5 into a state where commands received from the host PC can be processed. As well as 3V3A and VDDA becoming active, this also includes starting up of high-speed 14.318MHz and 48MHz clocks using external crystals, resetting and initialising all internal state machines and logic within VP, VC and CP modules. $\overline{\text{LOPOW}}$ is de-asserted at any time the host PC requests the module is either put into a USB SUSPEND condition or any other host-application-dependent mode when ultra low power consumption is required.

$\overline{\text{NORM}}$ is asserted when CPiA 1.5 is ready to put the module into the highest power mode of operation which occurs when the camera becomes active and high speed image transfers via USB or parallel port interfaces commences. $\overline{\text{NORM}}$ is used internally to CPiA 1.5 only but is provided as an optional output for driving customer-specific logic. For example, $\overline{\text{NORM}}$ can be used to illuminate an LED indicating the camera is active. $\overline{\text{NORM}}$ is also deasserted at any time the host requests USB SUSPEND mode.

$\overline{\text{LOPOW}}$	$\overline{\text{NORM}}$	Mode	Description	VDDA	Approx Module Current
1	1	Suspend	CPiA 1.5, camera module and DRAM components are in lowest power mode. VDDA is withdrawn as external FET is off, all fast clocks are disabled and CPiA 1.5 logic is in held reset	0V	<330uA
0	1	Low Power	CPiA 1.5 is in low power mode. Fast clocks enabled and CPiA 1.5 can process commands from host PC. Camera, video processing and DRAM controller modules are held in reset	5V	<100mA
0	0	High Power	All CPiA 1.5 modules, camera and DRAM components are active and video data is being transferred to host PC.	5V	<100mA
If VDD and VDDA is not controlled using LOPOW (VDD=VDDA=VDDU) in parallel port modes of operation, approx module power consumption will be <100mA at all times. In such cases, $\overline{\text{LOPOW}}$ and $\overline{\text{NORM}}$ can be considered redundant.					

Table 4 : Effect of $\overline{\text{LOPOW}}$ on CPiA 1.5-based module power consuption

CPiA 1.5's power management block requires two externally derived inputs: $\overline{\text{RESET}}$ provides a global reset to all CPiA 1.5 logic and CLK400H provides a low frequency clock to CPiA 1.5's internal power control state machines and should be constrained in accordance with the timing diagrams presented below. The Events are also described in the table below. After event RR, the process returns to CSU.



Timing Diagram Highlighting Power Management Events

Event	Description
POR	POWER ON RESET PROVIDED TO CPiA 1.5 BY EXTERNAL HARDWARE
CSU	HIGH FREQUENCY CLOCK START UP
LPM	LOW POWER MODE OF CPiA 1.5 OPERATION
RHP	HOST REQUEST FOR CPiA 1.5 TO ENTER HIGH POWER STATE
HPI	INITIALISE HIGH POWER MODE OF OPERATION
HPM	HIGH POWER MODE OF CPiA 1.5 OPERATION
SR	USB REQUEST POWER SAVING SUSPEND MODE OF OPERATION
SUSP	CPiA 1.5 HELD IN USB SUSPEND MODE
RR	USB REQUEST RESUME TO RECOVER LOW POWER MODE OF OPERATION

Table 5 : Timing Diagram Highlighting Power Management Events

Parameter	Description	min	max	Units
Trp	CPiA 1.5 global $\overline{\text{RESET}}$ pulse width	0.1	-	ms
Tp	CPiA 1.5 CLK400H period	2.25	2.75	ms
	CPiA 1.5 CLK400 duty	20:80	80:20	-
Trl	Time for low power mode after $\overline{\text{RESET}}$ deasserted	0	Tp	ms
Tck1su	14.318MHz Clock start-up time	-	Tp	ms
Tck2su	48MHz Clock start-up time	-	(2*Tp)-1	ms
Trvda	VDDA Power Supply rise time from $\overline{\text{LOWPO}}$ asserted	-	Tp	ms
Tllp	Time from $\overline{\text{LOWPO}}$ asserted to low power mode	-	2*Tp	ms
Tnsclk	Time from $\overline{\text{NORM}}$ to camera interface active	-	TBA	ms
Tnhpm	Time from $\overline{\text{NORM}}$ to high power mode	-	TBA	ms
Trhp	Time to acknowledge host request for high power mode	-	TBA	ms
Tsrs	Time for CPiA 1.5 to acknowledge and act upon USB suspend mode request	-	TBA	ms
Trrc	Time for CPiA 1.5 to acknowledge and act upon USB resume low power mode request	-	TBA	ms

Table 6 : Power management Parameters

2.3 Video Compression

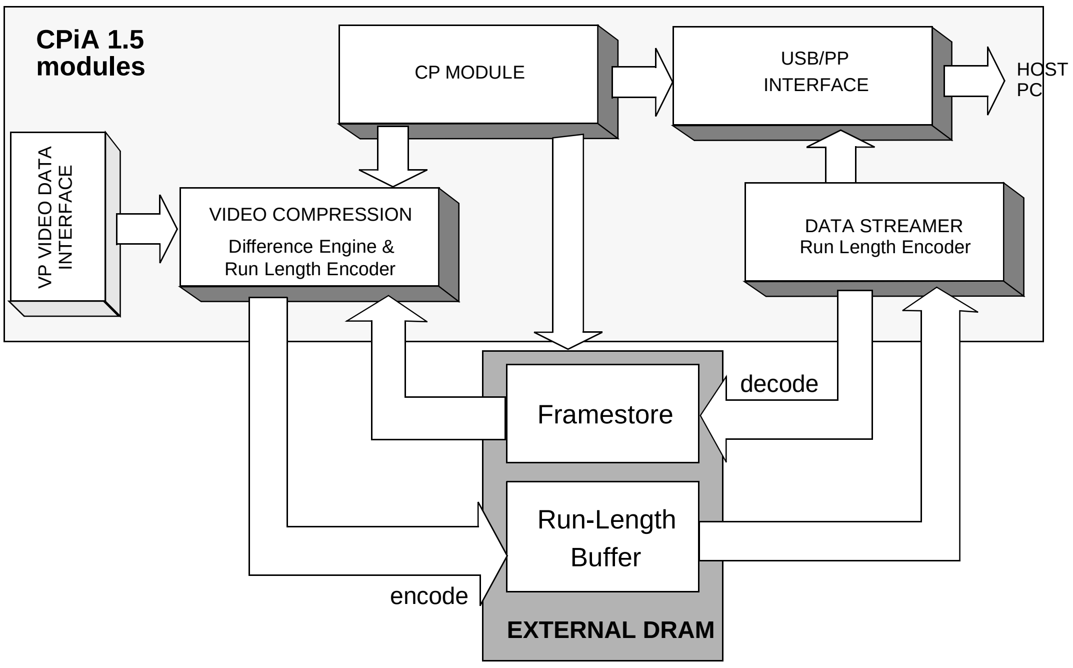


Figure 7 : Video compression components and VC Module interfacing to External DRAM

In the discussion to follow the CPiA 1.5 device is assumed to be interfaced to the the host PC via the USB, however the discussion equally well applies to the parallel port interface option that CPiA 1.5 also provides.

The VP module provides the video data stream to VC. The pixel-based compression algorithm operates by considering the differences between consecutive frames and encoding those differences in a Run-Length Buffer (RLB) held in external DRAM. The result is a run-length encoded stream of the changes from frame-to-frame. The stream consists of run-lengths of regions where no change has occurred, coded YCbYCr (YUV422) values of pixel-pairs which have changed significantly, additional codes to mark the start/end/length of frame rows, a frame header detailing parameters associated with the capture settings of the encoded data and finally codes to mark the end of the encoded frame.

The VC module implements the compression algorithm conceptually by way of two processes. The first process is the Differencing Engine/Run-Length Encoder (DE/RLE):

- Each pixel-pair of the current video frame is compared in turn with the corresponding pixel-pair stored in the DRAM FrameStore (FS). If this comparison indicates that there exists a significant difference in value for the given pixel-pair then that pixel-pair’s value is stored at the next locationin the RLB .The stored value is further coded to indicate its status as a pixel-pair value rather than a run-length. Conversely, if the comparison indicates that the difference is not significant, a run-length counter is incremented and stored at the current position in the RLB. The level of *significant* difference is dynamically calculated by the CP module from frame-to-frame and can be influenced by the host PC. When the DE/RLB finishes processing a frame the CP module is alerted indicating subsequent processes can commence.

The second process is the Data Streamer/Run-Length Decoder (DS/RLD):

- In parallel with the DE/RLE but delayed with respect to the current position in the RLB, the Data Streamer (DS) streams data from the RLB to USB interface module for transfer to the host. In addition, the DS also decodes the RLB stream back into the FS thus updating it. The software driver also decodes the received RLB stream on the host PC. When the DS/RLB reaches the end of the data stream the CP module is again alerted and the process repeats.

It is important to note that the two processes can run concurrently, that is the DE/RLE can be encoding the current frame into the RLB while at the same time the DE/RLE can be streaming out the previously encoded frame to the USB.

2.4 DRAM Interface

In order to perform video capture and compression, CPiA 1.5 uses a DRAM-based frame store. The integral DRAM controller is designed to support standard 256K x 16 EDO devices, with access times less than or equal to 60ns. Timing diagrams showing bus read and write cycles are highlighted below. The compression algorithm uses two main data structures within the DRAM:

- The Framestore holds a copy of the previous image uploaded to the host. The DE uses this information to determine significant differences between the FS image and the next image read from the VP.

- The Run Length Buffer stores the current frame in a run-length encoded form, ready for upload by the DS/USB to the host.

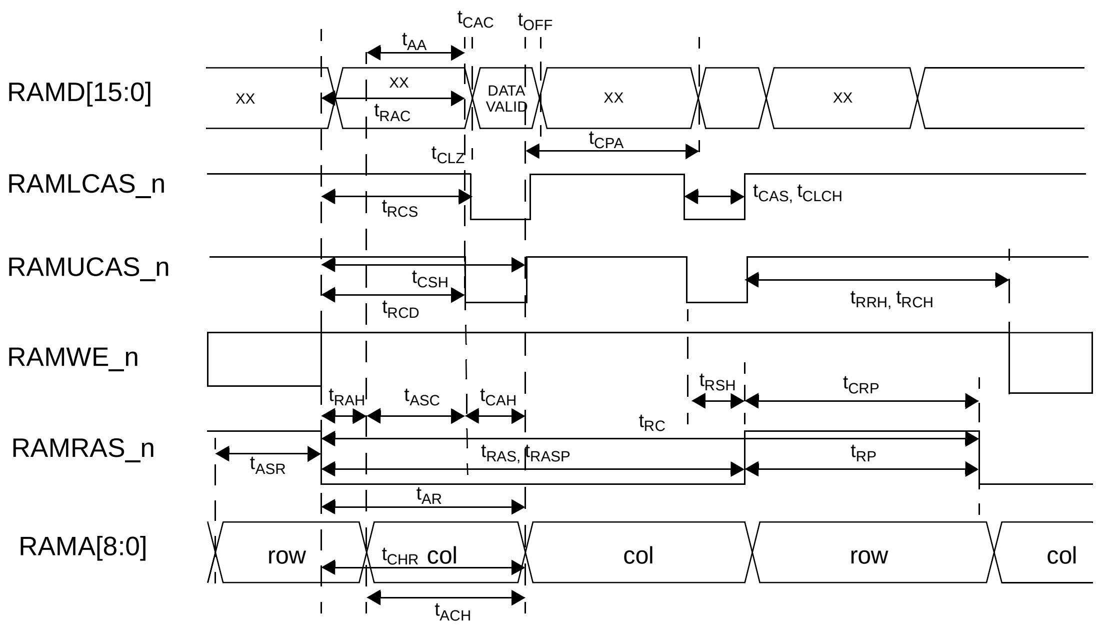


Figure 8 : DRAM Read Cycle Timings

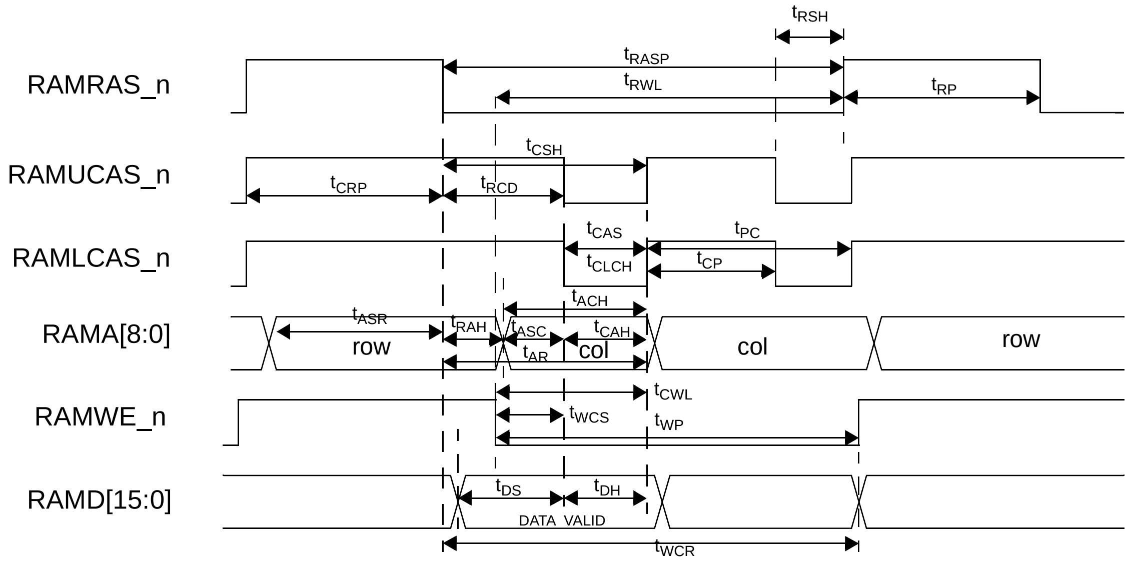


Figure 9 : DRAM Write Cycle Timing

Parameter	DRAM requirement (ns) Note 1	CPiA 1.5 DRAM Timing (ns) Note 2		Parameter	DRAM requirement (ns) Note 1	CPiA 1.5 DRAM Timing (ns) Note 2	
		min	max			min	max
t_{AA}	>30	43	58	t_{CAS}	>10	28	40
t_{ACH}	>15	52	58	t_{CAH}	>10	28	40
t_{AR}	>45	68	72	t_{CHR}	>10	28	40
t_{ASC}	>0	17	21	t_{CLCH}	>10	28	40
t_{ASR}	>0	68	72	t_{CLZ}	0	0	0
t_{CAC}	>17	19	26	t_{CP}	>8	28	40
t_{CRP}	>5	68	72	t_{CPA}	>35	62	65
t_{CSR}	>5	28	40	t_{CSH}	>45	68	72
t_{DH}	>10	28	40	t_{CWL}	>10	52	57
t_{DS}	>0	28	40	t_{OFF}	$0<t_{OFF}<15$	5.5	6
t_{RAC}	>60	63	67	t_{PC}	>25	68	72
t_{RAS}	>60	68	72	t_{RAH}	>10	11	21
t_{RRH}	>0	68	72	t_{RASP}	>60	68	72
t_{RCD}	$14<t_{RCD}<45$	28	40	t_{RC}	>105	136	144
t_{RCS}	>0	28	40	t_{RCH}	>0	68	72
t_{RP}	>40	68	72	t_{REF}	<8ms	-	16us
t_{RWL}	>15	52	57	t_{RPC}	>5	28	40
t_{WCH}	>10	28	40	t_{T}	<2	0.5	2
t_{WCS}	>0	17	20	t_{WCR}	>45	68	72
t_{WP}	>5	52	57	t_{RSH}	>15	28	40

Table 7 : CPiA 1.5 DRAM Interface Timing Parameters

Note 1 - DRAM timing parameters extracted from DRAM manufacturer 's worst-case spec data tables. Due to some variation in manufa cturer's figures it is recommended required values presented in this table are closely checked against the data tables from specific target DRAM manufacturers.

Note 2 - Min and Max timing derived from worst and best case design simulation with respect to process parameter derating with supply voltage and Tj. All timing parameters listed have been verified and validated through device characterisation with Ta@25C, VDD=VDDA=5V.

Data Structure	Size (bytes)	Size (Kbytes)
Framestore (FS)	202752	198
Run-Length Buffer (RLB)	205124	~201
Total (FS + RLB)	407876	~399

Table 8 : Framestore and Run Length Buffer DRAM partitioning

2.5 Control Processor

2.5.1 Control Processor Interfaces

As shown below, the Control Processor uses five main interfaces to control the operation of the CPiA 1.5 device:

- VP Control Registers
- VP Frame Start Interrupt
- VC Control Registers i.e. PP, USB interfaces, Compression Control and DRAM Interface
- USB_BANDWIDTH, ID_SELECT and DEBUG input pins
- UART serial comms (debug output only)

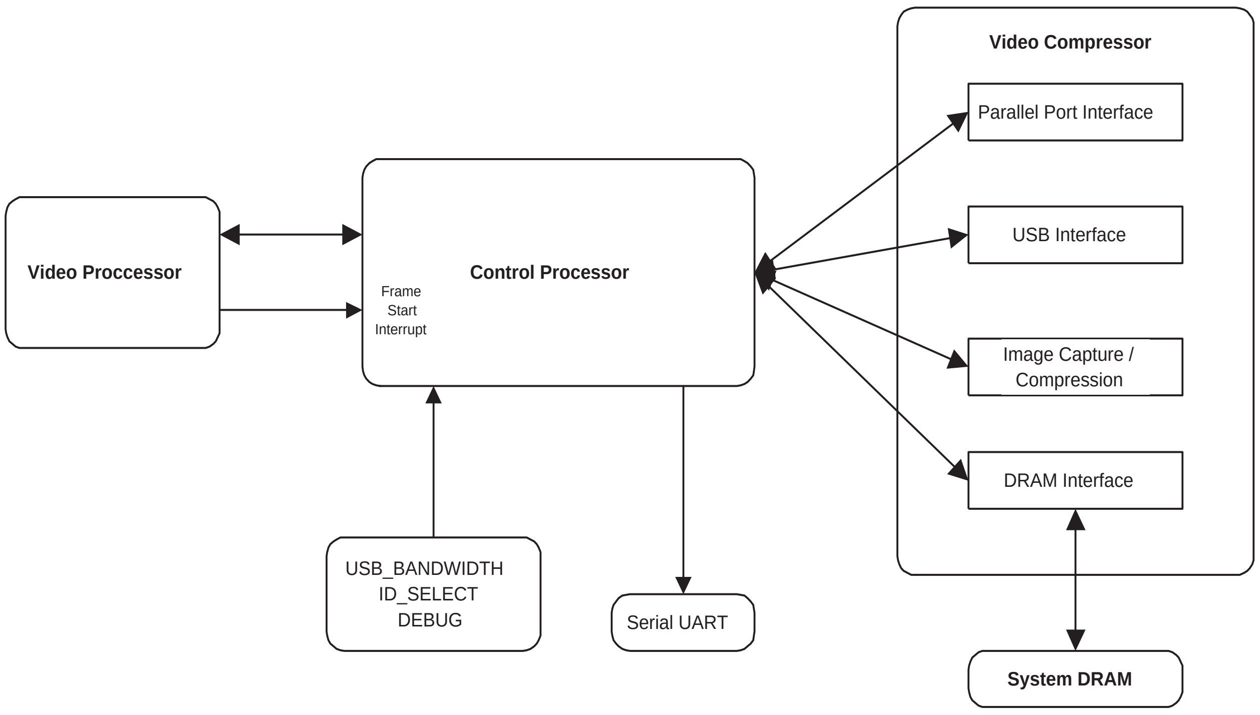


Figure 10 : Control Processor Interfaces

The VP Control Registers are used to control the colour processing of the raw sensor data. Colour channel gains, colour saturation and contrast values are all applied by the V P module.Also, communications to the sensor, for control of exposure time and gain, are done via the VP module.

The VP Frame Start interrupt is used to trigger Exposure Control and Auto White Balance algorithms that run on the Control Processor.

The VC Registers are used to control the host interfaces (both Parallel Port and USB) and the image capture / compression / upload processes. In addition access to the DRAM, used to store the frameheader, and during self-test mode, is via the VC regisiters. Control of transitions between the system power states of the camera is preformed via VC registers that access the Power Managment module.

The USB_BANDWIDTH, ID_SELECT and DEBUG inputs go directly to port pins on the embedded Control Processor core and modify several aspects of the camera behaviour. In normal operation these three inputs should be left unconnected (internally pulled up).

The USB_BANDWIDTH input can be used to select an alternate set of isochronous bandwidths for the USB interface. See the USB Interface section of this document for more details.

The ID_SELECT input is used to select a Device ID (Parallel Port interface) or Product ID (USB interface) that is used for combined USB and Parallel Port cameras. If held LO it causes a different 1284 Device ID to be returned on the parallel port interface, and a different USB Product ID on the USB Interface.

The DEBUG input if pulled LO will cause the Control Processor to send diagnostic output to the serial UART. This output consists mainly of a trace of all command and status data that has been received/transmitted over the host parallel port or USB interfaces.

The serial UART Tx signal is used to output diagnostic information. The Rx input is not used to receive serial data, but the Control Processor does check it's state after power up, or reset. If it is LO at this time the Control Processor enters a continuous Self-test Mode which it will remain in until the next system reset.

2.5.2 Control Processor Tasks

The Control Processor is responsible for a variety of functions necessary for the operation of the CPiA 1.5 device. These functions are:

- Initialisation of VP and VC modules
- Host communications via Parallel Port IEEE 1284 protocol
- Supervision of host USB interface
- Host command execution
- Image Capture / Compression Control
- Camera Auto Exposure
- Camera Auto White Balance
- Production system selftest diagnostics
- Diagnostic debug output

2.5.3 Selftest Diagnostics

If the Serial UART input Rx is pulled LO when the system is released from reset the Control Processor will enter Self-test Diagnostics Mode. It will continuously run a set of system level tests and output the result on the Serial UART Tx output. The serial format used is 19200 baud, 8-bit data, no parity, 1 stop bit, no flow control.

Below is typical output from the Selftest Mode.

```
====== CPiA 1.5Ver 1-30 ======

Continuous Selftest ...

1 - VP_EN            --------
2 - VC_INT           --------
3 - Host port        --------
4 - Pass-thru port   Fail
5 - USB core         --------

6 - VC register      --------
7 - DRAM access      --------
8 - VP register      --------
9 - Frame grab       --------
10 - VP data         --------
```

It should be noted that several of these self-tests were designed to test system level connectivity between the VP, VC and CP modules during the device development phase, and are now effectively obseleted with the full intergrated CPiA 1.5 device.

These self-tests do not attempt to test in anyway the connectivity, or correct operation, of the interface to the sensor.

VP_EN Test

Tests an internal CPiA 1.5 control signal - should never fail.

VC_INT Test

Tests an internal CPiA 1.5 control signal - should never fail.

Host Port

Tests for short circuits between signals on the Host Parallel port. Will fail if the camera is plugged in a host.

Pass-thru Port

Tests for short circuits and open circuits on the Pass-Through Parallel port.Requires a special loop back connector to do the test, and so will fail if this is not present.

USB Core

Tests that the USB core has been succesfully configured. Does not check actual communications with host. If the 48MHz Xtal is faulty, or starts up too slowly, this is test will fail. Because of this parallel port only designs will always show this test as failing.

VC Register Test

Tests internal CPiA 1.5 control signals - should never fail.

DRAM Access Test

Checks for correct CPiA 1.5 access to DRAM. It does this by writing a test pattern to DRAM, and then verifying it.

VP Register Test

Tests internal CPiA 1.5 control signals - should never fail.

Frame Grab Test

Tests internal CPiA 1.5 control signals - should never fail.

VP Data Test

Puts VP module into Colour Bar Mode, and captures a test image to DRAM and then verifies the correct data has been captured .If DRAM Access test fails this test will also fail.

2.5.4 Debug output

If the DEBUG input is pulled LO, the Control Processor will output diagnostic information on the Serial UART TX line. This output consists primarily of a trace of all the command and status data transactions that occur over the host parallel port or USB interfaces. It was primarily designed to be of use for developers working on the host driver software.

The serial format used is 19200 baud, 8-bit data, no parity, 1 stop bit, no flow control.

The following shows typical output following a system reset and then subsequent commands being received from the host by the camera.

```
 ====== CPiA 1.5 Ver 1-30 ======

Process loop ...
Detected PP i/f

Command = 01   CmdData = 00 00 00 00   DataIn = 01 14 01 00 A2 88 88 85

Command = 03   CmdData = 00 00 00 00   DataIn = 02 00 00 00 00 00 10 00

Command = 04   CmdData = 00 00 00 00

Command = A6   CmdData = 02 00 00 00
```

NOTE - When the debug output is enabled it significantly slows the handling of commands by the camera, and thus the framerate acheived on the host will be much reduced. In USB mode this may cause a failure of enumeration and so DEBUG should only be used once the enumeration sequence has occured.

3. External Interfaces

3.1 USB Interface

The USB interface is fully compliant with USB Specification Version 1.0. The USB interface is designed such that the camera attaches to the PC via a 4-wire USB cable, either directly at the PC motherboard USB connector (root hub) or via a USB hub (in turn connected to the PC motherboard root hub). The USB hub may be either a stand-alone USB hub device or a compound USB hub device incorporated, for example, in a USB monitor or printer.

USB transfers both signal and power over the 4-wire cable such that the camera is powered from the same cable/ connector over which it communicates with the PC. The 4-wire cable carries VBus (nominally +5V at the source) and GND wires to deliver power to devices and differential data lines, D+ and D-. The clock is transmitted encoded along with the differential data. The clock encoding scheme is NRZI with bit stuffing to ensure adequate transitions. A SYNC field precedes each packet to allow the receiver(s) to synchronise their bit recovery clocks. CPiA 1.5 provides fully compliant USB differential pads on-chip so there is no need to use an external, discrete bus tranceiver chip such as Philips' PDIUSBP11. In accordance with the USB Specification Version 1.0, the pads have been characterised for correct operation up to the maximum peripheral cable length of 5m.

3.1.1 CPiA 1.5-Based USB Camera

The CPiA 1.5-based USB camera is a high-power, bus-powered device. When it is first attached to a USB hub port it operates in its low-power mode consuming no more than 100mA. Once the camera has been successfully enumerated it may operate in its high-power mode. With the CPiA 1.5 even in high power mode the camera still consumes no more than 100mA, however in order to guarantee the USB power voltage level it is necessary to remain a high power device. Since the camera is a high-power, bus-powered device it will only enumerate successfully if connected to a self-powered USB hub. (A bus-powered USB hub is only capable of supplying 100mA at each of its downstream ports).

NOTE: If functionality with a bus-powered hub is important it is possible to configure the CPiA 1.5 to enumerate as a low power device, but some changes are needed to the Power management circuit if this is required.

The CPiA 1.5-based USB camera responds to suspend signalling issued by the USB (more than 3ms of bus IDLE time). In its suspend mode of operation the camera draws no more than 500uA. The camera does not support remote wakeup.

The CPiA 1.5 USB camera is a single configuration, single interface USB peripheral device. Its single Interface (Interface 0) supports Control Endpoint 0 and Isochronous IN Endpoint 1.

NOTE: The original CPiA ASIC used Interface 1, but this was found to cause issues with non-PC USB host implementations, so it was changed to Interface 0. This change is transparent to the rest of the functionality of the CPiA 1.5

All devices must support Control Endpoint 0 as this is the endpoint with which the PC communicates over the default pipe at device enumeration time. CPiA 1.5 utilises Control Endpoint 0 to respond to standard USB commands (such as those used at device enumeration time) and to Vendor specific CPiA 1.5 commands which are used to control the operation of the camera and to request status information from the camera. Note that the camera does not support a USB Class Specification and so only Vendor commands are used to control the operation of the camera.

Isochronous IN Endpoint 1 is used to transfer video data from the camera to the PC. Using an isochronous endpoint for this task guarantees bandwidth and latency for the transfer of video data. The bandwidth requested for video data transfer can be varied by means of the four defined Alternate Settings of Interface 0.

3.1.2 USB Control Transfers

Control transfers are used to pass control information to the CPiA 1.5-based USB camera from the host and to request state information back from the camera. For example, control transfers are used at device enumeration time when the host PC requests information about the USB device which has been dynamically attached to the bus to enable it to load the appropriate driver(s). CPiA 1.5 uses Control Endpoint 0 for all control transfers. Therefore Endpoint 0 handles both standard USB commands and Vendor specific CPiA 1.5 commands.

A control transfer consists of three phases:

- a SETUP phase
- a DATA phase (should one exist)
- a STATUS phase.

The SETUP phase contains details of the command being sent to the camera including, the specific command request, whether the command is a WRITE (host to camera) or READ (camera to host) command and the number of bytes expected in the DATA phase. (For further information on the SETUP phase please refer to USB Specification Version 1.0, Section 9.3).

The DATA phase is used to transfer a number of bytes of information associated with the command to the camera if this is a WRITE command or from the camera if this is a READ command. The DATA phase can consist of a number of discrete DATA transactions on the bus. With CPiA 1.5-based USB cameras, data is transfered in 8-byte packets with the remaining data bytes transfered in the last data transaction being less than 8 bytes if the total number of bytes to be transfered is not a multiple of 8. This is because the maximum packet size of Control Endpoint 0 in CPiA 1.5 is 8 bytes.

The STATUS phase is used to indicate whether or not all was well with the SETUP and DATA phases and consequently whether or not the control transfer has completed successfully. If at any time the camera receives an unrecognised command (for example due to data corruption on the USB) or it cannot process a valid command for some reason it returns a STALL handshake on the bus to indicate to the host that something is wrong.

3.1.3 USB Device Enumeration

The USB hub detects that a high speed USB device has been dynamically attached to one of its downstream ports by the presence of a 1.5KOhm pull-up resistor on the D+ differential data line on the upstream port of the device, as is the case with a CPiA 1.5-based USB camera. Before the port to which the camera has been connected can be enabled the hub ensures that the device is reset. It does this by issuing SE0 reset signalling on the USB differential data lines for at least 10ms and then enables the port. This causes the camera to ‘listen’ to USB traffic at bus address 0 until such time as it is assigned its own unique USB bus address. The camera is ready to respond to host access within 10ms of this reset signalling being de-asserted.

Note that the hub waits a specified delay time after device attach before issuing the reset signalling to ensure the device’s internal power supply has stabilised. During this delay time the bus is held in IDLE. On power-up CPiA 1.5 goes into its low power mode (LOPOW goes low) before it determines that the USB has been IDLE for more than 3ms and so cycles into its suspend mode of operation (LOPOW goes high). It is not until the host issues the reset signalling that CPiA 1.5 comes out of suspend and into its low power mode of operation again (LOPOW goes low).

Now that the camera is listening at bus address 0 the host begins to communicate with the camera. The initial communications comprise the device enumeration traffic sequence. The host first issues a *GetDescriptor* (DEVICE) command (SETUP Packet 80 06 00 01 00 00 40 00) to Control Endpoint 0 (which all USB devices must support) at bus address 0. The host only receives the first 8 data bytes of the DATA phase of this control transfer (that is, the first DATA transaction within the DATA phase) before issuing the STATUS phase.

The host then issues a *SetAddress* command (SETUP Packet 00 05 yz wx 00 00 00 00 where wxyz is the USB device address) to assign a unique USB bus address to the camera. Following the STATUS phase of a successful *SetAddress* command the camera then ‘listens’ and responds to USB traffic at this unique address only.

The host continues by issuing a *GetDescriptor* (DEVICE) command (SETUP Packet 80 06 00 01 00 00 12 00). In response to the *GetDescriptor* (DEVICE) command the camera returns its DEVICE descriptor which comprises 18 bytes of information. The DEVICE descriptor describes general information about the USB device.

The table below shows the DEVICE descriptor information returned by a CPiA 1.5-based USB camera in response to the *GetDescriptor* (DEVICE) command. This information can be checked at device enumeration time using a USB bus analyser such as the CATC Inspector when debugging a CPiA 1.5-based USB camera design.

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x12	Size of this descriptor in bytes.
bDescriptorType	1	0x01	DEVICE DescriptorType.
bcdUSB	2	0x0100	USB Specification Release Number identifying the release of the USB Specification that the device and its descriptors are compliant with. (Release 1.00)
bDeviceClass	1	0x00	Class Code (assigned by USB). This field is set to 0x00 meaning that the class information is specified in the Interface descriptor
bDeviceSubClass	1	0x00	Subclass Code (assigned by USB).
bDeviceProtocol	1	0x00	Protocol Code (assigned by USB). This field is set to 0x00, meaning that the device does not use class specific protocols on a device basis, however may use class specific protocols on an interface basis.
bMaxPacketSize0	1	0x08	Maximum packet size for Endpoint 0.
idVendor	2	0x0553	Vendor ID (assigned by USB).
idProduct	2	0x0002 (1)	Product ID (assigned by the manufacturer).
bcdDevice	2	0x0100	Device release number in binary-coded decimal.
iManufacturer	1	0x00	Index of string descriptor describing manufacturer.
iProduct	1	0x00	Index of string descriptor describing product.
iSerialNumber	1	0x00	Index of string descriptor describing the device's serial number.
bNumConfigurations	1	0x01	Number of possible configurations

Notes:

(1) The CPiA 1.5 device returns a Product ID as part of its DEVICE descriptor. This Product ID is used along with the Vendor ID to load the approriate device drivers for the CPiA 1.5-based USB camera. The device can return multiple Product ID numbers depending on the state of the ID_SELECT pin and the HP_DATA[4:0] pins. The ID_SELECT pin has an internal pull-up resistor which, when not connected, provides a Product ID of 0x0002 for a USB-only CPiA 1.5-based camera module. This is the default Product ID that is used with all versions of the standard STMicroelectronics USB camera driver, regardless of customer version.

It is recommended that the standard Product ID is used where possible, unless the CPiA 1.5 is being designed into a product that is different from a standard USB camera, when an alternate product ID can be assigned by STMicroelectronics.

Please contact STMicroelectronics if you require to make use of this feature.

Finally the host issues a *GetDescriptor* (CONFIGURATION) command (SETUP Packet 80 06 xx 02 00 00 FF 00). CPiA 1.5 has only one Configuration (index xx = 00). If xx is anything other than 00 CPiA 1.5 will STALL the default pipe. In response to the *GetDescriptor* (CONFIGURATION) command the camera returns its Configuration, Interface and Endpoint descriptors which comprise 73 bytes of information.

The tables below show the descriptor information returned by a CPiA 1.5-based USB camera in response to the *GetDescriptor* (CONFIGURATION) command. This information can be checked at device enumeration time using a USB bus analyser such as the CATC Inspector when debugging a CPiA 1.5-based USB camera design.

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x09	Size of this descriptor in bytes.
bDescriptorType	1	0x02	CONFIGURATION Descriptor Type.
wTotalLength	2	0x0049	Total length of data returned for this configuration. Includes the combined length of all descriptors (configuration, interface, endpoint, and class or vendor specific) returned for this configuration.
bNumInterfaces	1	0x01	Number of interfaces supported by this configuration.
bConfigurationValue	1	0x01	Value to use as an argument to *SetConfiguration* to select this configuration.
iConfiguration	1	0x00	Index of string descriptor describing this configuration.
bmAttributes	1	0x80	Configuration characteristics: Bus Powered and no Remote Wakeup.
MaxPower	1	0xC8	Maximum power consumption of USB device from the bus in this specific configuration when the device is fully operational. Expressed in 2mA units (0xC8 = 400mA).

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x09	Size of this descriptor in bytes.
bDescriptorType	1	0x04	INTERFACE Descriptor Type.
bInterfaceNumber	1	0x01	Number of interface.
bAlternateSetting	1	0x00	Value used to select alternate setting for the interface identified in the prior field.
bNumEndpoints	1	0x01	Number of endpoints used by this interface (excluding Endpoint 0).
bInterfaceClass	1	0xFF	Class code (assigned by USB). A value of 0xFF means that the interface class is vendor specific.
bInterfaceSubClass	1	0x00	Subclass code (assigned by USB).

Descriptor Field	Size (bytes)	Value	Description
bInterfaceProtocol	1	0xFF	Protocol code (assigned by USB). A value of 0xFF means that the device uses a vendor specific protocol for this interface
iInterface	1	0x00 (1)	Index of string descriptor describing this interface.

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x07	Size of this descriptor in bytes.
bDescriptorType	1	0x05	ENDPOINT Descriptor Type.
bEndpointAddress	1	0x81	The address of the endpoint on the USB device described by this descriptor: (IN endpoint, number 1)
bmAttributes	1	0x01	Field describing the endpoint's attributes: Isochronous transfer type.
wMaxPacketSize	2	0x0000/ 0x0000 (2)	Maximum packet size this endpoint is capable of sending: 0 bytes or 0 bytes (1).
bInterval	1	0x01	Interval for polling endpoint for data transfers (in ms). For isochronous endpoints this field must be set to 1 (ie. every USB frame).

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x09	Size of this descriptor in bytes.
bDescriptorType	1	0x04	INTERFACE Descriptor Type.
bInterfaceNumber	1	0x01	Number of interface.
bAlternateSetting	1	0x01	Value used to select alternate setting for the interface identified in the prior field.
bNumEndpoints	1	0x01	Number of endpoints used by this interface (excluding Endpoint 0).
bInterfaceClass	1	0xFF	Class code (assigned by USB). A value of 0xFF means that the interface class is vendor specific.
bInterfaceSubClass	1	0x00	Subclass code (assigned by USB).

Descriptor Field	Size (bytes)	Value	Description
bInterfaceProtocol	1	0xFF	Protocol code (assigned by USB). A value of 0xFF means that the device uses a vendor specific protocol for this interface
iInterface	1	0x00	Index of string descriptor describing this interface.

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x07	Size of this descriptor in bytes.
bDescriptorType	1	0x05	ENDPOINT Descriptor Type.
bEndpointAddress	1	0x81	The address of the endpoint on the USB device described by this descriptor: (IN endpoint, number 1)
bmAttributes	1	0x01	Field describing the endpoint's attributes: Isochronous transfer type.
wMaxPacketSize	2	0x01C0/ 0x0140 (1)	Maximum packet size this endpoint is capable of sending: 448 bytes or 320 bytes (1).
bInterval	1	0x01	Interval for polling endpoint for data transfers (in ms). For isochronous endpoints this field must be set to 1 (ie. every USB frame).

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x09	Size of this descriptor in bytes.
bDescriptorType	1	0x04	INTERFACE Descriptor Type.
bInterfaceNumber	1	0x01	Number of interface.
bAlternateSetting	1	0x02	Value used to select alternate setting for the interface identified in the prior field.
bNumEndpoints	1	0x01	Number of endpoints used by this interface (excluding Endpoint 0).
bInterfaceClass	1	0xFF	Class code (assigned by USB). A value of 0xFF means that the interface class is vendor specific.
bInterfaceSubClass	1	0x00	Subclass code (assigned by USB).
bInterfaceProtocol	1	0xFF	Protocol code (assigned by USB). A value of 0xFF means that the device uses a vendor specific protocol for this interface

Descriptor Field	Size (bytes)	Value	Description
iInterface	1	0x00 [1]	Index of string descriptor describing this interface.

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x07	Size of this descriptor in bytes.
bDescriptorType	1	0x05	ENDPOINT Descriptor Type.
bEndpointAddress	1	0x81	The address of the endpoint on the USB device described by this descriptor: (IN endpoint, number 1)
bmAttributes	1	0x01	Field describing the endpoint's attributes: Isochronous transfer type.
wMaxPacketSize	2	0x02C0/ 0x0268 [1]	Maximum packet size this endpoint is capable of sending: 704 bytes or 616 bytes [1].
bInterval	1	0x01	Interval for polling endpoint for data transfers (in ms). For isochronous endpoints this field must be set to 1 (ie. every USB frame).

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x09	Size of this descriptor in bytes.
bDescriptorType	1	0x04	INTERFACE Descriptor Type.
bInterfaceNumber	1	0x01	Number of interface.
bAlternateSetting	1	0x03	Value used to select alternate setting for the interface identified in the prior field.
bNumEndpoints	1	0x01	Number of endpoints used by this interface (excluding Endpoint 0).
bInterfaceClass	1	0xFF	Class code (assigned by USB). A value of 0xFF means that the interface class is vendor specific.
bInterfaceSubClass	1	0x00	Subclass code (assigned by USB).
bInterfaceProtocol	1	0xFF	Protocol code (assigned by USB). A value of 0xFF means that the device uses a vendor specific protocol for this interface
iInterface	1	0x00	Index of string descriptor describing this interface.

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x07	Size of this descriptor in bytes.
bDescriptorType	1	0x05	ENDPOINT Descriptor Type.
bEndpointAddress	1	0x81	The address of the endpoint on the USB device described by this descriptor: (IN endpoint, number 1)
bmAttributes	1	0x01	Field describing the endpoint's attributes: Isochronous transfer type.
wMaxPacketSize	2	0x03C0/ 0x0380 (1)	Maximum packet size this endpoint is capable of sending: 960 bytes or 896 bytes (2).
bInterval	1	0x01	Interval for polling endpoint for data transfers (in ms). For isochronous endpoints this field must be set to 1 (ie. every USB frame).

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x0E	Size of this descriptor in bytes.
bDescriptorType	1	0x03	STRING Descriptor Type.
wLANGID[0]	12	0x0904 0x0c04 0x0704 0x1004 0x0A0C 0x1604	The unicode IDs for the languages supported by the device American English, Std French, Std German, Std Italian, Trad Spanish, Portugese (Brazilian)

Descriptor Field	Size (bytes)	Value	Description
bLength	1	0x16	Size of this descriptor in bytes.
bDescriptorType	1	0x03	STRING Descriptor Type.
bString	20	"USB Camera"	This string is returned for all string descriptors > 0

Notes:

(1) The first number shows maximum packet size values returned when $\overline{\text{USB\_BANDWIDTH}}$ is left unconnected (internally pulled up), as is the case in normal operation. If USB_BANDWIDTH is pulled LO externally the second number is returned as the maximum packet size for the isochronous endpoint. For further information see section **2.2.4.4**.

On successful completion of these control transfers the host has gathered all relevant information about the camera and is able to load the appropriate drivers for the camera. Once this has been done the camera is said to be enumerated.

3.1.4 USB Isochronous Transfers

As stated in section **2.2.4.1**, Isochronous IN Endpoint 1 is used to transfer video data from the camera to the host. Interface 1 has four Alternate Settings defined allowing different USB bandwidths to be requested for this transfer of video data. Furthermore, it is possible to use the USB_BANDWIDTH input to select a second set of Alternate Setting isochronous bandwidths. In normal operation USB_BANDWIDTH is left unconnected (internally pulled up). If it is pulled LO externally a second set of Alternate Setting isochronous bandwidths are used. The table below shows the Alternate Setting isochronous bandwidths of Endpoint 1 for both USB_BANDWIDTH HI and LO.

Alternate Setting	**Isochronous bandwidth per ms USB_BANDWIDTH=1 (default)**	**Isochronous bandwidth per ms USB_BANDWIDTH=0**
0	0 bytes	0 bytes
1	448 bytes	320 bytes
2	704 bytes	616 bytes
3	960 bytes	896 bytes

Table 9 : USB Bandwidth settings

Note that the second set of Alternate Setting bandwidths available by pulling $\overline{\text{USB\_BANDWIDTH}}$ LO allow a slightly lower set of bandwidth settings to be used.

The isochronous bandwidth is reported as the number of bytes per USB frame. The USB is framed into 1ms time slots.

Alternate Setting 0 is always 0 bytes/ms. This ensures that we can select an Alternate Setting for the camera where no bandwidth is reserved. Alternate Setting 0 is the default Alternate Setting selected when the CPiA 1.5 USB driver is loaded, such that when the camera is enumerated but not in operation (no application is using the camera hardware to transfer video data) then no USB bandwidth is requested. This is essential for a USB-friendly peripheral such that other devices may use the full potential of the bus at this time.

When an application wishes to use the camera the CPiA 1.5 USB driver will request isochronous bandwidth for the transfer of video data. It will start by requesting the largest bandwidth supported by CPiA 1.5, that is Alternate Setting 3, to ensure the best possible framerate by default. If the USB bus has many isochronous devices attached and Alternate Setting 3 cannot be supported the CPiA 1.5 USB driver will then request Alternate Setting 2 and if necessary Alternate Setting 1 to guarantee bandwidth for the transfer of video data. If even the bandwidth requirement of Alternate Setting 1 cannot be supported by the bus at that time the user will have to stop using another isochronous device to free-up isochronous bus time to allow the USB camera to operate.

Additionally, the camera user can select the USB bandwidth they wish to use by means of the Multimedia Control Panel Settings Dialogue for the camera. Here they can select High (Alternate Setting 3), Medium (Alternate Setting 2) or Low (Alternate Setting 1) USB bandwidth. Thus if the CPiA 1.5 camera is currently using High bandwidth and they wish to connect and simultaneously use another isochronous USB device they can manually force the CPiA 1.5 camera to use less bandwidth by selecting Medium or Low bandwidth settings. Obviously this may result in lower framerate video (depending on the current compression setting).

Once an Alternate Setting has been selected and is supported by the bus then the CPiA 1.5 camera has guaranteed bandwidth and latency for the transfer of video data. This means that in every 1ms USB time slot an IN token will be issued to the CPiA 1.5 Endpoint 1 for the transfer of up to *MaxPacketSize* bytes of video data, where *MaxPacketSize* is the reserved bandwidth per ms. The CPiA 1.5 camera will transfer as much data as it currently has available up to *MaxPacketSize*. When no compression is selected then *MaxPacketSize* bytes of video data will be transfered in every USB frame. If compression is selected then less than *MaxPacketSize* bytes of video data may be transfered depending on the level of compression requested. When the device is sending a compressed image stream to the host the amount of data to be sent for each image frame can vary, depending on the compression ratio achieved. To enable the host to detect the end of valid image data the device inserts four *0xFF* values into the image stream to mark the end of image frame. As a sequence of four 0xFF values will never occur within a valid frame of image data the host can use this to simply detect the end of the image data, without having to decompress the image. The host driver checks the intergrity of the image data stream before it is decoded. If any corruption is detected the frame is discarded and another uploaded.

3.2 Parallel Port Interface

The parallel port interface is designed to attach directly to the parallel printer port of a PC. As such it contains 8 bi-directional data lines, 4 control inputs and 5 status outputs for connection to the host parallel port.

Control Signals	Status Signals	Data Signals
H_STROBE	H_FAULT	HP_DATA[7:0]
H_AUTOFD	H_ACK	
H_INIT	H_BUSY	
H_SELECTIN	H_PERROR	
	H_SELECT	

Table 10 : Parallel Port signals

This following sections give a guide to the low level protocols used to transfer command and image data over the parallel port interface. They should be read in conjunction with the document "IEEE Std 1284 - Standard Signaling method for a Bidirectional Parallel Peripheral Interface for Personal Computers". It is intended to give users enough information to design and debug boards utilising the CPiA 1.5 device.

3.2.1 Parallel Port Protocol

There are three classes of transfer that take place over the parallel port.

- IEEE Std 1284 Device ID Transfers
- Command / Status Transfers
- Image Stream Transfers

1284 Device ID transfers are used by the Microsoft Windows 95/98 operating systems to detect the presence of the CPiA 1.5-based camera at boot time and load the approriate drivers.

Command and Status transfers are used to pass control information to the CPiA 1.5-based camera from the host as well as returning state information back to the host.

Image Stream Transfers are used to transfer image data from CPiA 1.5 to the host. This data can be in a compressed or uncompressed format depending on how the CPiA 1.5 has been set up via the associated host PC software driver.

3.2.2 1284 Device ID Transfers

The CPiA 1.5 device can return a Device ID string, as defined in IEEE Std 1284, using Nibble Mode Reverse Channel Transfer. This Device ID is requested by the Microsoft Windows 95/98 operating systems during their PnP boot processes. It is used to load the approriate device drivers for the hardware attached to the port.

The device can return two possible Device ID strings dependent on the state of the $\overline{\text{ID\_SELECT}}$ pin. This pin has an internal pull-up resistor which, when not connected, provides a Device ID for a parallel port only CPiA 1.5-based camera module. When pulled low externally, the Device ID will be that which is used for a combined parallel port and USB module.

The complete Device ID's returned in these two situations are shown in the table below.

$\overline{\text{ID\_SELECT}}$	Returned Device ID
1(default)	MFG:VLSI Vision Ltd;MDL:PPC2 Camera;CMD:CPiA 1.5_1-20;CLS:MEDIA;DES:Parallel Port Camera;
0	MFG:VLSI Vision Ltd;MDL:DUAL-B Camera;CMD:CPiA 1.5_1-20;CLS:MEDIA;DES:USB / Parallel Port Camera;

Table 11 : Parallel Port Camera Device I.D.s

3.2.3 Command / Status Transfers

All command and status transfers are preceeded by the negotiation and setup phases as defined in the Std 1284 for the ECP and NIbble modes.

Forward data transfers always use the ECP transfer protocol.

Reverse data transfers use either ECP, or Nibble mode, transfer protocols, depending or whether the host has a bi-directional port or not.

Channel addressing and RLE as defined in the ECP spec are not used by CPiA 1.5.

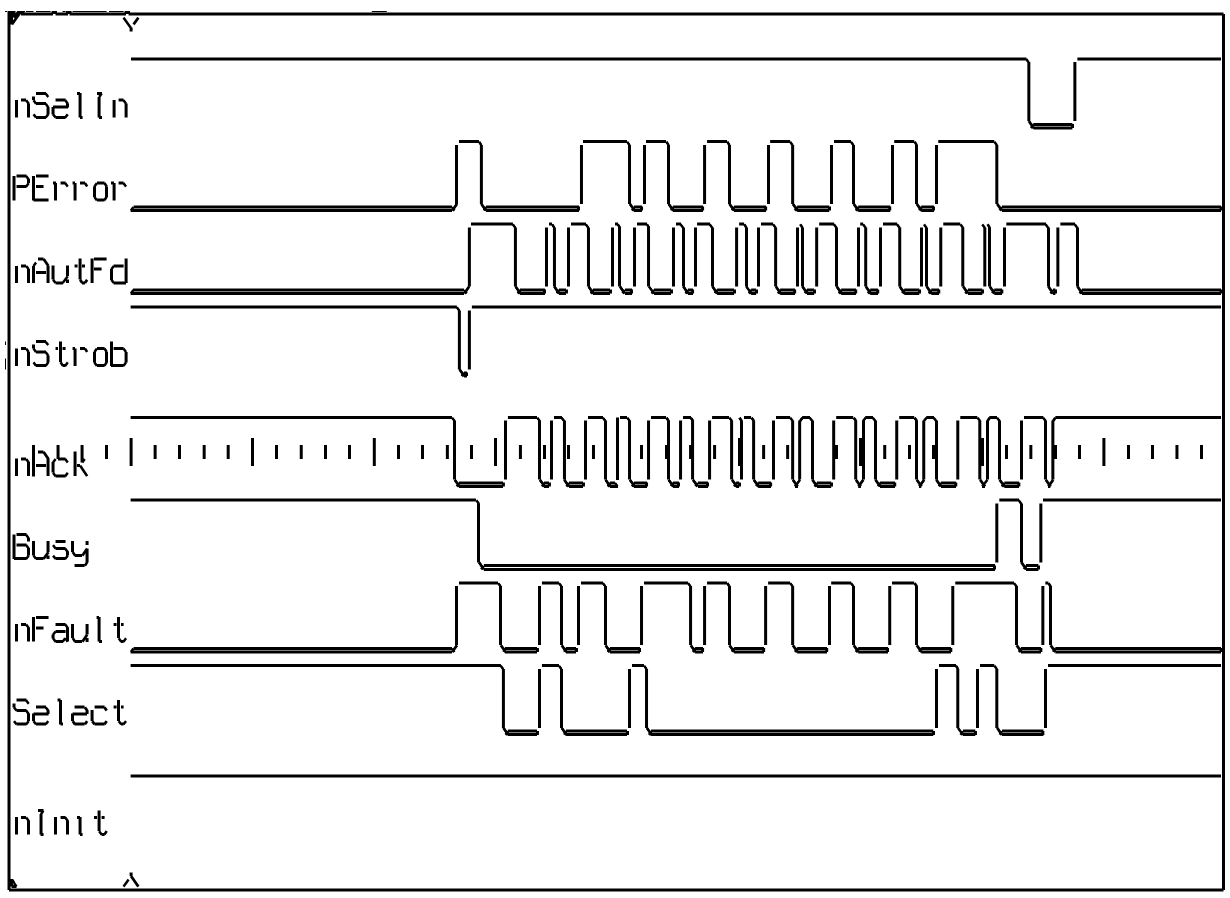
Data is transfered in 8 byte packets, each packet being preceeded by the 1284 negotiation and setup sequences, and terminated with a 1284 termination sequence.

Commands are passed to CPiA 1.5 in a single 8 byte packet, termed a *command transfer.* This can optionally be followed by the transfer of a *data transfer.* Bytes within the command packet are used to specify whether a data packet is to follow, and its direction.

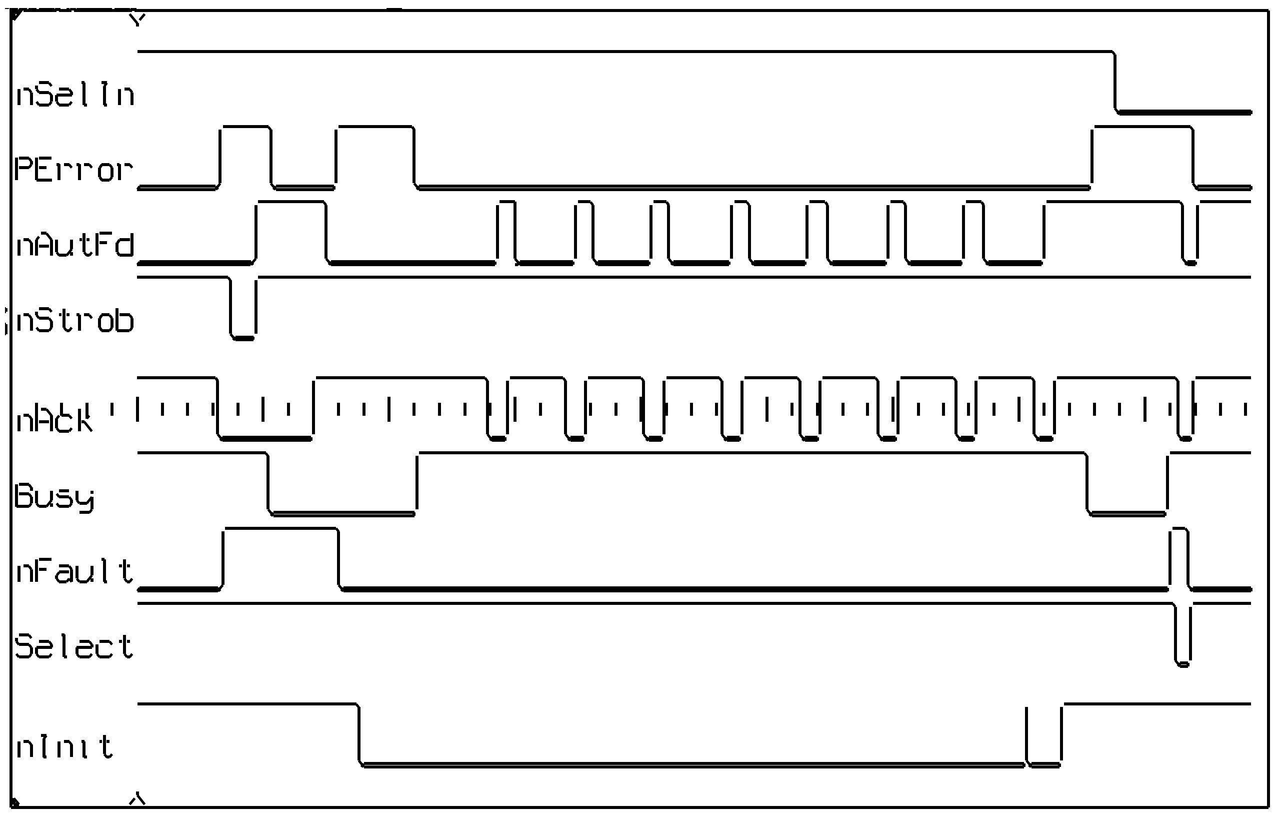
Most commands are not accompanied by data packets.

The following diagram shows the activity that occurs on the parallel port control and status lines during a command transfer to CPiA 1.5, via a Forward ECP Mode transfer. NOTE - The actual data is carried on the 8 data lines, which are not shown on the diagram.

The diagram below shows the activity that occurs on parallel port control and status lines during the return of status information to the host via a Nibble Mode transfer. In this transfer mode the actual data is carried on the PError, nAutoFd, nAck, Select lines.


nSelIn
PError
nAutFd
nStrob
nAck
Busy
nFault
Select
nInit

The diagram below shows the activity that occurs on parallel port control and status lines during the return of status information to the host via a Reverse ECP Mode transfer. NOTE - The actual data is carried on the 8 data lines, which are not shown on the diagram.



3.2.4 Image Stream Transfers

Transfers of image stream data are distinguished from those of the command and status data by the use of a slightly different negotiation phase preceeding the transfer. Image stream transfers operates only in the reverse direction i.e. CPiA 1.5 to host.

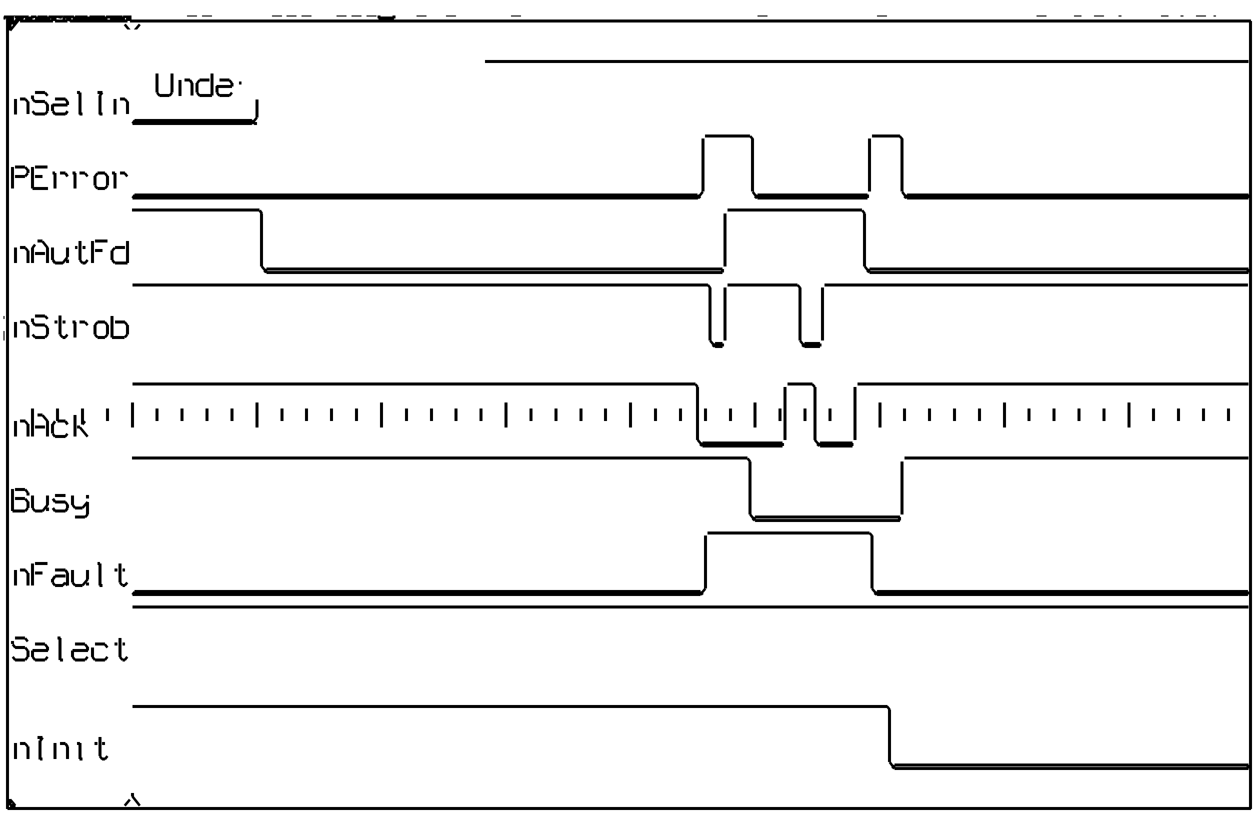
All image stream transfers are preceeded by a 1284 negotiation sequence which sets the "extensiblity link request" bit within the first Extensibility Request Value. The subsequent Extensiblity Request Value then determines which of two possible transfer modes will be used to transfered the data. Image Stream transfers are terminated with the standard 1284 termination sequence.

The protocol used to transfer the data depends on the capabilites of the host. If it has an ECP parallel port then the standard ECP byte level protocol is used. However, if it has a bi-directional, or 4 bit port, then a modified form of the 1284 Nibble mode transfer is used. This modified form, refered to hereafter as a *Nibble Stream mode* , is used in preference to the standard nibble mode to improve the maximum transfer rate. Nibble Stream mode differs from standard Nibble mode by the fact that both the rising and falling edges of the nAutoFd signal are used to clock data.

The diagram below shows the negotiation sequence that proceeds a Nibble Stream Mode transfer, and the start of of the actual image data transfer.

NOTE - Requests by the host for Nibble Stream or ECP Stream mode transfers can be easily disguished from standard Nibble Mode and ECP negotiations by the double nStrobe low pulses that occur during the negotiation phase.

The diagram below shows the negotiation sequence that proceeds a ECP Stream Mode transfer, no actual bytes are shown being transfered.



Notes on Image Stream Transfers

CPiA 1.5 produces a low going pulse, of approximately 200 µs, on the nAck line when image stream data is available to upload. The host driver can use this pulse to trigger a parallel port interrupt service routine to start the the image transfer process.

When the device is sending compressed image stream to the host the amount of data to be sent for each frame can vary, depending on the compression ratio achieved. So that the host can detect the end of valid data the device inserts a stream of *0xFF* values into the image stream after the host reads past the end the valid image data. As a sequence of four 0xFF values will never occur within a valid frame of image data the host can use this to simply detect the end of the image data, without having to decompress the image.

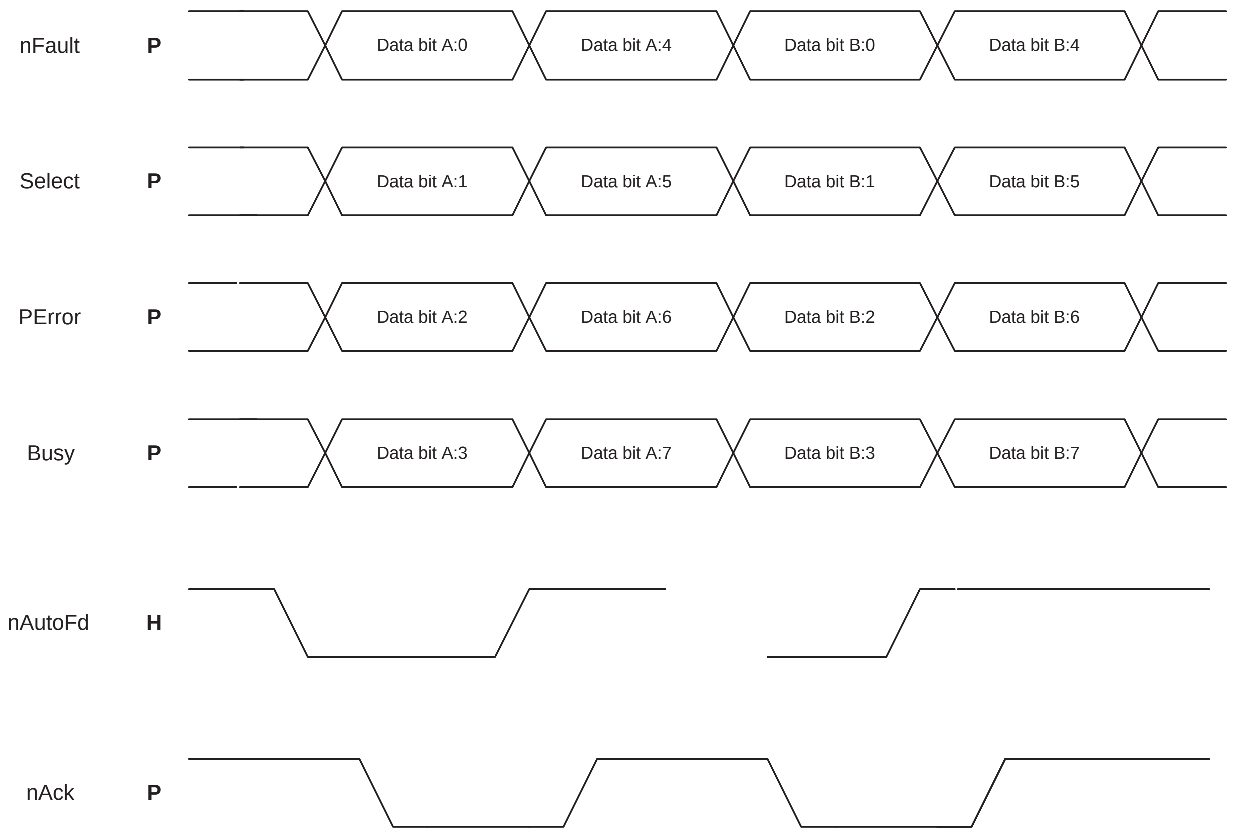
The host driver checks the intergrity of the image data stream before it is decoded, if any corruption is detected the frame is discarded and another uploaded.

If, while uploading uncompressed images, the transfer of data is monitored via an osciliscope, or logic analyser, it will be seen there are regular short pauses in the transfer, in the order of 200µs to 2ms each. These are caused by the host driver suspending the transfer while it copies data out of it's temporary buffer into the target frame buffer.

Details of Nibble Stream Mode Protocol

The figure below shows the details of the Nibble Stream Protocol. It shows two succesive bytes, A and B, being transfered.

The host requests a byte bysetting nAutoFd to LO. CPiA 1.5 will then present the lower nibble of the byte on the 4 status lines. It then sets nAck LO to signify that the data is ready. nAutoFd going HI causes the upper nibble to be placed on the status lines and nAck is then taken HI to signify the data is ready for the host to read.



3.2.5 Pass-Through Mode

The CPiA 1.5 device no longer includes any support for Pass-through mode. The pins required for this mode are no longer bonded out, and cannot be accessed in this device.

3.2.6 IEEE 1284 Mode Handshake Values

The following table shows the timing values of the parallel port interface in terms of the signal transitions defined in the IEEE Std 1284 figures 3 through 23.

Time	Minimum	Maximum	Description
T_H	0	0.5 s(note 1)	Host response time
T	0		Infinite response time
T_L	70 ns	10 ms (note 2)	Peripheral response time
T_R	70 ns		Peripheral response time (ECP Mode only)
T_S	(note 3)		Host recovery time (ECP Mode only)
T_P	0.5 µs		Minimum setup or pulse width
T_D	0		Minimum data setup time (ECP Mode only)

Table 12 : IEEE 1284 Mode Handshake Values

Notes:

1 - A delay of more than this period during a Command/Status transfer can cause the watchdog reset circuitry on CPiA 1.5 to time out and generate a system reset.

2 - Normally 10ms, however, certain commands being issued to CPiA 1.5 can caused this to increase to 40ms

3 - As an ECP Forward Channel stall condition will never be generated by CPiA 1.5, the Host Transfer Recovery handshake is not supported.

3.3 Snapshot control

The CPiA 1.5 supports the connection of a momentary action button for use as a snapshot button for still image capture. This button input is encoded into the frame header and can be interrogated by an application via the Private Interface to the STMicroelectronics camera driver. (For more information about the driver Private Interface please contact STMicroelectronics)

The following additional circuit is required to support the snapshot control. It provides hardware debounce for the switch.

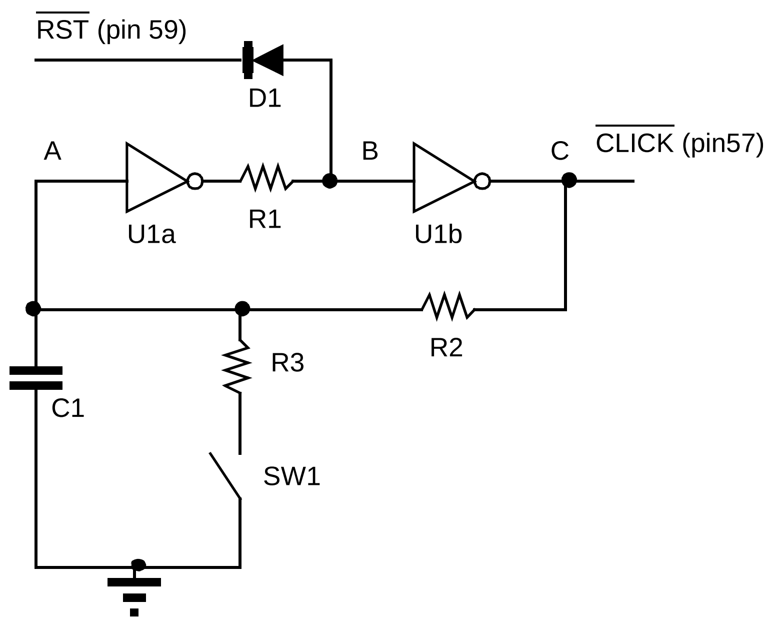

$\overline{\text{RST}}$ (pin 59)
D1
A
B
C
$\overline{\text{CLICK}}$ (pin57)
U1a
R1
U1b
R2
R3
C1
SW1

Figure 11 : Snapshot circuit schematic

Part	Description
R1, R2	Resistor 10K
R3	Resistor 100R
U1	74LS04 Hex Invertor (or equivalent)
D1	1N1418 Signal Diode (or equivalent)
C1	Capacitor 10nf ceramic
SW1	Switch - normally open momentary action

Table 13 : Snapshot circuit parts list

3.3.1 Circuit Description

The circuit is designed to work with active low signals and is designed to provide positive feedback to a high or low voltage held on a capacitor. The feedback paths are high impedance and can be over ridden by lower impedance signals.

In use point B is at the opposite voltage to points A and C, so if point A is high then B is low and C is high, C being high keeps A high.

The circuit will boot with A low, due to the capacitor. Before the circuit can be used it must be reset. This is acheived by pulling the $\overline{\text{RST}}$ signal low. When $\overline{\text{RST}}$ is low it overcomes the drive of the gate though the 100K resistor and point B goes low. Therefore point C goes high. The capactor starts to charge though the second 100K resistor and after 100mS (the time constant of 100nF and 10KOhms) point A goes high. $\overline{\text{RST}}$ can now be released as point B is now held low by the feedback though the gate.

When the user wishes to take a picture, they will press a normally open switch closing the contacts. This will discharge the capacitor though the 100R resistor. Point A will go low,point B will go high and point C ($\overline{\text{CLICK}}$) will go low, indicating that the switch has been pressed. The circuit state is kept if the switch is released by the feedback in the circuit.

When required the circuit can be reset, as above, by pulling $\overline{\text{RST}}$ low.

3.3.2 Operation

The snapshot button feature of the CPIA device, and Driver support for it, operate as follows:

When the push button is pressed the external circuit latches indicating that the button has been pressed. This is connected to the $\overline{\text{CLICK}}$ pin of CPIA 1.5.

After each frame is uploaded the driver checks the state of the button latch input. If it detects the button has been pressed it records this fact, and then clears the push button latch via the $\overline{\text{RST}}$ output on CPIA 1.5.

At any time the application may ask the driver, via the private interface, if any <button pressed> events have been detected, since the last time that the application asked the driver. Requesting this information from the driver causes the driver internal <button pressed> record to be reset.

The application is only informed of the fact that a <button pressed> event has been detected. It can not determine how may, or duration of the presses, actually occured since it last asked the driver.

NOTES

- Because of the external latch, very short push button events can be detected reliably.
- Because the driver effectively "latches" it's detection of push button events the application can poll the driver at any rate it wants without missing a push button event.
- Before the push button can be used the application must request that the driver start checking the push button state each frame. There is a performance hit for doing the button checking operation, so it is not enabled by default.
- Only "push button pressed" events are recorded by the driver. There is no support for "button down", "button up" style of events.
- There is currently no callback mechanism for the driver to directly inform the application of a button press event.

3.4 General Purpose Input and Output signals

The CPiA 1.5 device has spare Input and Output pins that can be used for general purpose I/O and controlled from the driver via the Private Interface. This allow additional features, not previously envisaged, to be added to the camera and controlled from application software on the PC.

(For more information on the Driver Private Interface please contact STMicroelectronics.)

In Parallel Port mode only one pin is available - Spare1 (Input or Output)

In USB mode the following parallel port pins can be used in addition:

H_STROBE (Input)
H_AUTOFD (Input) I
H_FAULT (Output)
H_INIT (Input)I
H_ACK (Output)
H_BUSY (Output)
H_PERROR (Output)
H_SELECT (Output)

NOTE: If these signals are being used as I/O in USB mode it is necessary to pull the H_SELECTIN pin to 0V, in order to prevent the ASIC entering Parallel Port mode on startup.

3.5 Sensor Serial interface

A serial interface connects the CPiA 1.5 to the VV6404 and defect map EEPROM. The serial interface allows the CPiA 1.5 to initialize the VV6404 after power-up and pass exposure settings at regular intervals. Sensor settings will be overwritten during normal operation. This means that the user should not need to access the sensor via the local serial interface. Sensor settings are already optimised for best performance.

The communication from host to slave takes the form of 8-bit data, with a maximum serial clock video processor frequency of 100 kHz. Since the serial clock is generated by the bus-master, it determines the data transfer rate. Data transfer protocol on the bus is shown in Figure 12.

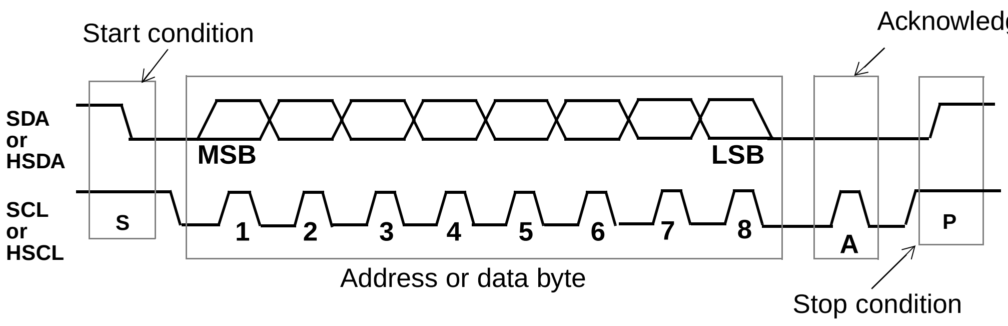


Figure 12 : serial interface Data Transfer Protocol

3.5.1 Data Format

The host serial interface is described, but the same principle applies to local serial interface. Information is packed in 8-bit packets (bytes) always followed by an acknowledge bit. The SDA line must be stable during the high period of *HSCL*. The exceptions to this are *start* (S) or *stop* (P) conditions when *HSDA* falls or rises respectively, while *HSCL* is high.

A message contains at least two bytes preceded by a *start* condition and followed by a *stop*.

The first byte contains the device address byte which includes the data direction (R/W), bit. The device address of VP3 is selectable via the SAD pin in the range 88_H-$8B_H$ (See Table 14). The LSB of the address byte indicates the direction of the message. If the LSB is set high then the master will read data from the slave and if the LSB is low then the master will write data to the slave. After the *R/W* bit is sampled, the data direction cannot be changed, until the next address byte with a new *R/W* bit is received.

The byte following the address byte contains the index of the register to be accessed.

The serial interface can address up to 128 registers in VV6404. When accessing VV6404, the MSB of the second byte is used to set/disable the automatic increment feature.

3.5.2 Serial address possibilities

Table 14 shows the valid serial interface addresses used with CPiA 1.5.

Serial Address		Comment
1010_XXX_R	$A0_H$ - AF_H	Serial EEPROM - 8 possible addresses
0010_000_R/W	20_H - 21_H	VV6404 address (fixed)

Table 14 : Allocation of Serial Interface Addresses

The byte following the sensor address byte contains the address of the first data byte (also referred to as the *index*).

3.5.3 Sensor serial interface Timing

Note: All values referred to the minimum input level (high) and maximum input level (low) logic levels as specified below.

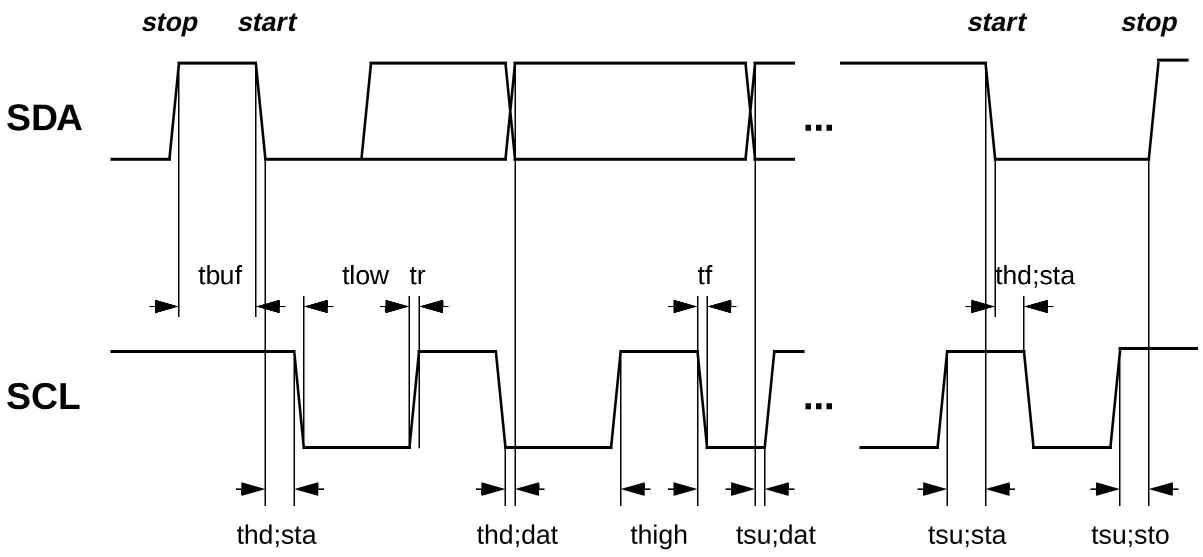


Figure 13 : Serial Interface Timing Characteristics

Parameter	Symbol	Min.	Max.	Unit
SCL clock frequency	fscl	0	100	kHz
Bus free time between a ***stop*** and a ***start***	tbuf	2	-	us
Hold time for a repeated ***start***	thd;sta	80	-	nS
LOW period of SCL	tlow	320	-	nS
HIGH period of SCL	thigh	160	-	nS
Set-up time for a repeated ***start***	tsu;sta	80	-	nS
Data hold time	thd;dat	0	-	us
Data Set-up time	tsu;dat	0	-	ns
Rise time of SCL, SDA (from V_{IL} = 0.2VDD - V_{IH} = 0.8VDD)	tr	-	300 (Note 1)	ns
Fall time of SCL, SDA (from V_H = 2.4v - V_L = 0.5v)	tf	-	300 (Note 1)	ns
Set-up time for a ***stop***	tsu;sto	80	-	nS
Capacitive load of each bus line (SCL, SDA)	Cb	-	200	pF

Table 15 : Serial Interface Timing Characteristics

Notes on Table 15:

(1) With 200pF capacitive load. It is recommended that pull-up resistors of 2.2-4.7k are fitted to both SDA and SCL lines.

4. Detailed specifications

4.1 Chipset VV6404+CPiA 1.5

Start-up mode	80mA
Active mode	90mA
Standby mode	400µA

Table 16 : VV6404+CPiA1.5 chipset current consumption

4.2 VV6404 Sensor

Image Format	352 x 288 pixels (CIF)
Pixel Size	12.0 x 11.0µm
Image array size	4.2mm x 3.2mm
Array Format	CIF
Exposure control	25000:1
Sensor signal / Noise ratio	42dB
Supply Voltage	5.0v DC +/-5%
Package type	48LCC
Operating Temp. range *	0^oC - 40^oC
Serial interface frequency range	0-100kHz

Table 17 : VV6404 Specifications

Note (*) For extended temperature information, contact STMicroelectronics

Parameter	Min.	Max.	Notes
V_{IL}	-0.5v	0.3 x V_{DD}	Guaranteed input low voltage
V_{IH}	0.7 x V_{DD}	V_{DD} + 0.5v	Guaranteed input high voltage
V_{OL}		0.4V	At max I_{OL} for pad type
V_{OH}	2.4v	V_{DD} - 0.5v	IOH = 100µA At max I_{OH} for pad type
T_J Junction Temp	0 deg C	100 deg C	
Internal Pullup resistor	35kΩ	150kΩ	
Internal Pulldown resistor	35kΩ	150kΩ	

Table 18 : VV6404 DC Characteristics

4.2.1 Device Power, NC and DC pins

Pin	Signal	Description
2,11,17,22,26,33,41,50,55,60,70,74,77, 87,	VSS	Digital padring & logic ground.
31,40,47,61	VDDA	Digital padring & logic power for VP & CP
6,19,30,66,86,92	VDD	Digital padring & logic power for VC
62	VSSU	Ground for the USB
65	VDDU	3.3V supply for the USB
78,79,80	NC	Not connected pins

Table 19 : Device Power Pins

4.3 Absolute Max Ratings

Description	Range	Unit
Ambient Temperature	0 to 40	^{o}C
StorageTemperature	-50 to 150	^{o}C
Voltage on USB D+/D-, Vcc3V to Vss	TBA	V
Voltage on any other I/O to Vss	TBA	V

Table 20 : Absolute Maximum Ratings

4.4 DC Characteristics

Parameter	Description	Min	Max	Units
VDD	Primary CPiA 1.5 Power Supply	3.3	5.5	V
VDDA	Secondary CPiA 1.5 Power Supply	3.3	5.5	V
VDDU	3.3V Power Supply for on-chip USB tranceiver	3.3	3.3	V
Isuspend	CPiA 1.5 suspend mode current		500	uA
Ilowpo	CPiA 1.5 low power modecurrent		100	mA
Iactive	CPiA 1.5 active, high power mode current		100	mA

Table 21 : DC Characteristics

Vilu, Vihu	USB differential pad D+/D- Vil, Vih	See Section 1.2	V
Volu, Vohu	USB differential pad D+/D-Vol, Voh		V
Iilu, Iihu	USB differential pad D+/D- Iil, Ili		mA
Vilp, Vihp	Parallel Port pads Vil, Vih		V
Volp, Vohp	Parallel Port pads Vol, Voh		V
Iilp, Iihp	Parallel Port pads Iil, Ili		mA
Vild, Vihd	Dram Interface pads Vil, Vih		V
Vold, Vohd	Dram Interface pads Vol, Voh		V
Iild, Iihd	Dram Interface pads Iil, Ili		mA
Vilm, Vihm	Power Management pads Vil, Vih		V
Volm, Vohm	Power Management pads Vol, Voh		V
Iilm, Iihm	Power Management pads Iil, Ili		mA

Table 21 : DC Characteristics

5. Pin descriptions, Pinouts, and Package Information

5.1 CPiA 1.5 Device Pinout

CPiA 1.5 has been packaged in an industry-standard 100 pin PQFP. The pinout has been carefully developed to minimise the physical size of the support printed circuit board by facilitating placement of and electrical routing to peripheral support components such as DRAM and parallel port connectors.

Reference should also be made to drawings located at the end of this data sheet for physical dimensions and other, more detailed mechanical information.

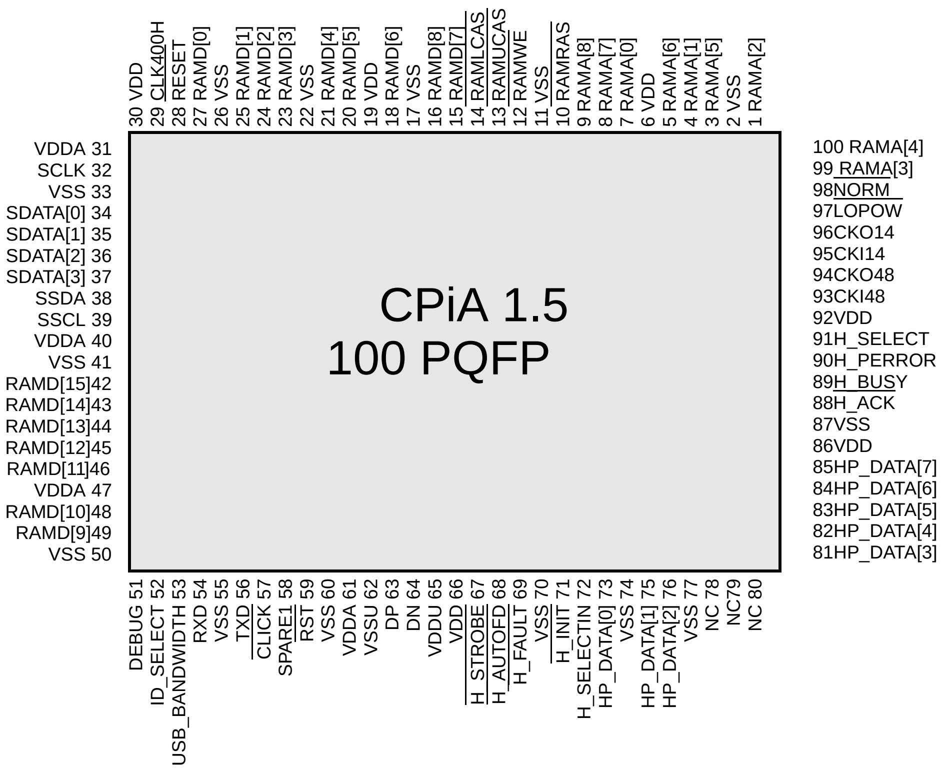


Figure 14 : CPiA1.5 Pinout Diagram

5.2 CPiA 1.5 Device I/O

Pin	Signal	Type	Description	drive
			Video Processor	
32	SCLK	O (4)	VVL6404 Sensor Clock (7.159MHz)	2 mA or 4mA
34	SDATA[0]	I (3)	VVL6404 Sensor Data Bus	-
35	SDATA[1]	I (3)	VVL6404 Sensor Data Bus	-
36	SDATA[2]	I (3)	VVL6404 Sensor Data Bus	-
37	SDATA[3]	I (3)	VVL6404 Sensor Data Bus	-
38	SSDA	I/O (2,3,6)	VVL6404 Sensor Serial Interface Data	2 mA
39	SSCL	I/O (2,3,6)	VVL6404 Sensor Serial Interface Clock	2 mA
			Control Processor	
51	$\overline{\text{DEBUG}}$	I	Control Processor Debug Output Select	-
52	$\overline{\text{ID\_SELECT}}$	I	USB Camera Device ID Select	-
53	$\overline{\text{USB\_BANDWIDTH}}$	I	Alternate USB Bandwith Settings	-
54	RXD	I	Control Processor Serial Communications	-
56	TXD	O(4,6)	Control Processor Serial Communications	2 mA
57	$\overline{\text{CLICK}}$	I	Snapshot button input (formerly spare0)	-
58	SPARE1	I	Control Processor spare (P3.4)	2 mA
59	$\overline{\text{RST}}$	I	Snapshot latch reset (formerly spare2)	2 mA
			USB	
63	DP	I/O (1,5,7)	USB Differential Output (+)	-
64	DN	I/O (1,5,7)	USB Differential Output (-)	-
			Parallel Port Interface	
67	$\overline{\text{H\_STROBE}}$	I (1,4,9)	Parallel port HOST STROBE control	-
68	$\overline{\text{H\_AUTOFD}}$	I (1,4,9)	Parallel port HOST AUTOFEED control	-
69	$\overline{\text{H\_FAULT}}$	O (5,9)	Parallel port HOST FAULT status	16 mA
71	$\overline{\text{H\_INIT}}$	I (1,4,9)	Parallel port HOST INIT control	-
72	$\overline{\text{H\_SELECTIN}}$	I (1,4,9)	Parallel port HOST SELECTIN control	-
73	HP_DATA[0]	I/O (1,4,5,9)	Parallel port Host data bus	16 mA
75	HP_DATA[1]	I/O (1,4,5,9)	Parallel port Host data bus	16 mA
76	HP_DATA[2]	I/O (1,4,5,9)	Parallel port Host data bus	16 mA
81	HP_DATA[3]	I/O (1,4,5,9)	Parallel port Host data bus	16 mA
82	HP_DATA[4]	I/O (1,4,5,9)	Parallel port Host data bus	16 mA
83	HP_DATA[5]	I/O (1,4,5,9)	Parallel port Host data bus	16 mA
84	HP_DATA[6]	I/O (1,4,5,9)	Parallel port Host data bus	16 mA
85	HP_DATA[7]	I/O (1,4,5,9)	Parallel port Host data bus	16 mA
88	$\overline{\text{H\_ACK}}$	O (5,9)	Parallel port HOST ACK status	16 mA
89	H_BUSY	O (5,9)	Parallel port HOST BUSY status	16 mA

Table 22 : CPiA 1.5 pin list

Pin	Signal	Type	Description	drive
90	H_PERROR	O (5,9)	Parallel port HOST PERROR status	16 mA
91	H_SELECT	O (5,9)	Parallel port HOST SELECT status	16 mA
			VC Master Clocks	
93	CKI48	I(8)	48.0MHz Oscillator Amplifier Input Pad	-
94	CKO48	O(7)	48.0MHz Oscillator Amplifier Output Pad	
95	CKI14	I	14.318MHz Oscillator Amplifier Input Pad	-
96	CKO14	O	14.318MHz Oscillator Amplifier Output Pad	
			Power Management Interface	
97	$\overline{\text{LOPOW}}$	O(7)	External device VCC control	2 mA
98	$\overline{\text{NORM}}$	O(7)	External device VCC control	2 mA
29	CLK400H	I (4)	Power management clock	-
28	$\overline{\text{RESET}}$	I (4)	Master Reset	-
			System Memory (DRAM) Interface	
99	RAMA[3]	O (10)	DRAM multiplexed address	2 mA
100	RAMA[4]	O (10)	DRAM multiplexed address	2 mA
1	RAMA[2]	O (10)	DRAM multiplexed address	2 mA
3	RAMA[5]	O (10)	DRAM multiplexed address	2 mA
4	RAMA[1]	O (10)	DRAM multiplexed address	2 mA
5	RAMA[6]	O (10)	DRAM multiplexed address	2 mA
7	RAMA[0]	O (10)	DRAM multiplexed address	2 mA
8	RAMA[7]	O (10)	DRAM multiplexed address	2 mA
9	RAMA[8]	O (10)	DRAM multiplexed address	2 mA
10	RAMRAS_n	O (10)	DRAM row address strobe	8 mA
12	RAMWE_n	O (10)	DRAM data write enable	2 mA
13	RAMUCAS_n	O (10)	DRAM upper column address strobe	8 mA
14	RAMLCAS_n	O (10)	DRAM lower column address strobe	8 mA
15	RAMD[7]	I/O (10)	DRAM data	2 mA
16	RAMD[8]	I/O (10)	DRAM data	2 mA
18	RAMD[6]	I/O (10)	DRAM data	2 mA
49	RAMD[9]	I/O (10)	DRAM data	2 mA
20	RAMD[5]	I/O (10)	DRAM data	2 mA
48	RAMD[10]	I/O (10)	DRAM data	2 mA
21	RAMD[4]	I/O (10)	DRAM data	2 mA
46	RAMD[11]	I/O (10)	DRAM data	2 mA
23	RAMD[3]	I/O (10)	DRAM data	2 mA
45	RAMD[12]	I/O (10)	DRAM data	2 mA
24	RAMD[2]	I/O (10)	DRAM data	2 mA

Table 22 : CPiA 1.5 pin list

Pin	Signal	Type	Description	drive
44	RAMD[13]	I/O (10)	DRAM data	2 mA
25	RAMD[1]	I/O (10)	DRAM data	2 mA
43	RAMD[14]	I/O (10)	DRAM data	2 mA
27	RAMD[0]	I/O (10)	DRAM data	2 mA
42	RAMD[15]	I/O (10)	DRAM data	2 mA

Table 22 : CPiA 1.5 pin list

If the target application requires that only one interface mode is required (eg. USB interface only), please refer to the above table and notes 7, 8 and 9 to establish how unused pins should be electrically connected on the PCB. Failure to ensure such pins are correctly connected may result in erroneous system behaviour.

Notes:

1. Internal pullups.
2. These pins shall go tri-state when $\overline{\text{LOPOW}}$ is high.
3. TTL level Schmitt input.
   $V_{IL(max)}$= 0.8V, $V_{IH\ (MIN)}$= 2.2V, schmitt trigger hysteresis=250mV (typ).
   Input leakage -10µA to +10µA, V_{IN} : 0 to DVDD
4. Output
   $V_{OL(max)}$ =0.4V @ I_{OL} =16mA
   $V_{OH(min)}$ =2.4V @ I_{OH} = -12mA
5. These signals to comply with USB Specification Version 1.0 section 7.
6. Open drain output.
   $V_{OL(max)}$ =0.4V @ I_{OL} =3mA
7. This pin does not require connection if CPiA 1.5 is used for parallel port interfacing products only.
8. This pin should be tied low if CPiA 1.5 is used for parallel port interfacing products only.
9. This pin does not require connection if CPiA 1.5 is used for USB interfacing products only.
10. These pins shall go tri-state when $\overline{\text{NORM}}$ is high.

Where I/O pad type is not explicitly defined in the above notes assume CMOS.

5.3 Mechanical Information

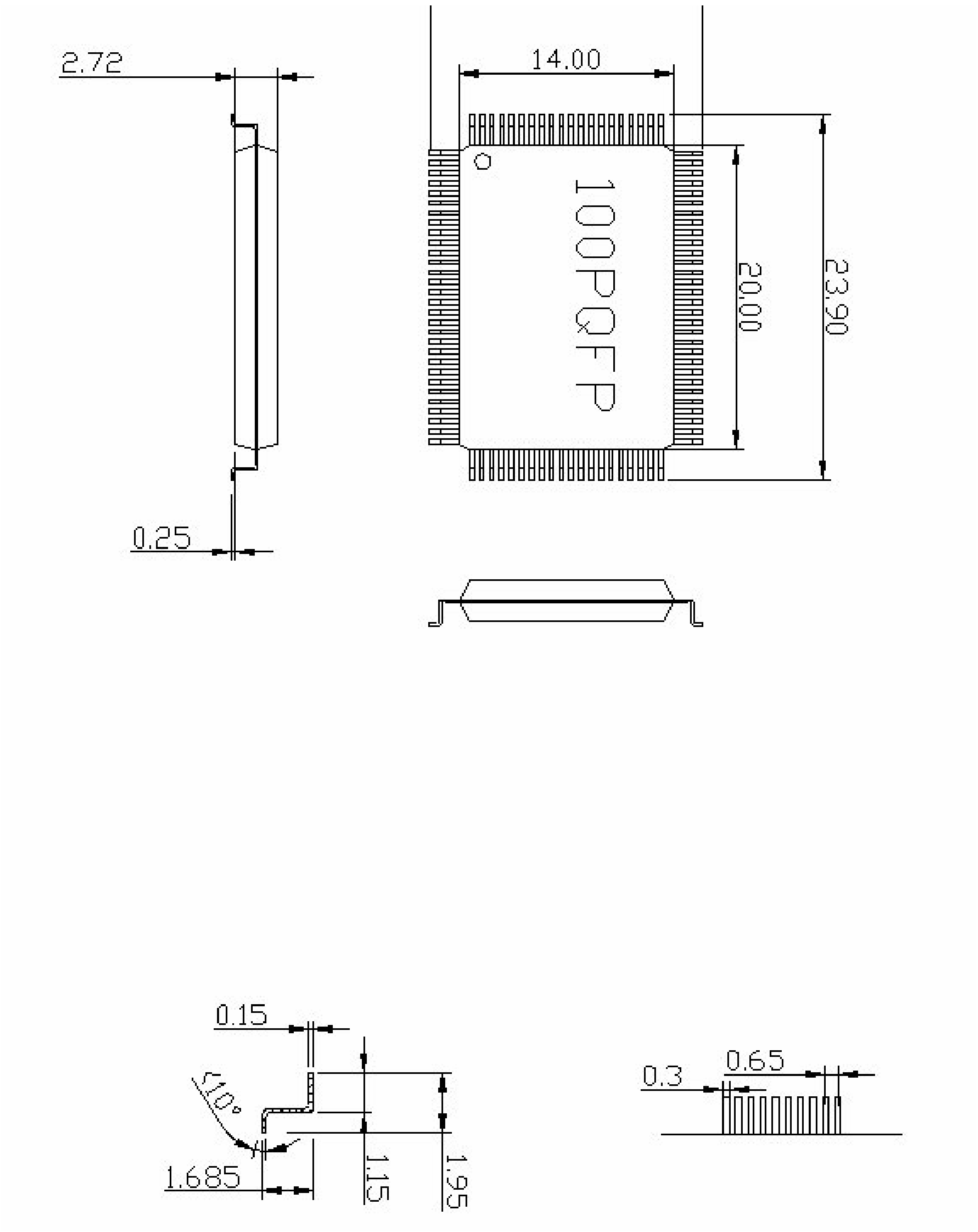


Figure 15 : CPiA1.5 Package Dimensions

5.4 VV6404 Pin listing

Pin	Name	Type	Description
POWER SUPPLIES			
1	AVCC	PWR	Core analogue power and reference supplies.
48	AGND	GND	Core analogue ground and reference supplies.
43	DVDD	PWR	Core digital power.
44	DVSS/Dsub	GND	Core digital ground.
6	AVDD	PWR	Output stage power.
5	AVSS	GND	Output stage ground.
7	VVDD	PWR	Analogue output buffer power.
9	VVSS	GND	Analogue output buffer ground.
14	ADCVDD	PWR	ADC power.
10	ADCVSS	GND	ADC ground.
19	VDD1	PWR	Digital padring & logic power.
30	VDD2	PWR	Digital padring & logic power.
36	VDD3	PWR	Digital padring & logic power.
20	VSS1	GND	Digital padring & logic ground.
29	VSS2	GND	Digital padring & logic ground.
37	VSS3	GND	Digital padring & logic ground.
ANALOGUE SIGNALS			
45	VBLOOM	OA	Anti-blooming pixel reset voltage
46	VBLTW	OA	Bitline test white level reference
47	VBG	OA	Internally generated bandgap reference voltage 1.22V
2	VCM/ VREF2V5	OA	Common-mode input for OSA and Internally generated 2.5 V reference voltage.
3	VRT	IA	Pixel reset voltage
4	VCDSH	IA	Reference Voltage
11	ADCbot	IA	Bottom voltage reference for ADC
13	ADCtop	IA	Top voltage reference for ADC
13	TopRef	OA	Internally generally top voltage reference for ADC
DIGITAL CONTROL SIGNALS			
12	HOLDPIX	ID↓	Test Pin
15	SIN	ID↓	Frame timing reset (soft reset).
16	RESETB	ID↑	System Reset. Active Low.

Table 23 : VV6404 Pinout

Pin	Name	Type	Description
SERIAL INTERFACE			
42	SCL	ID↑	Serial bus clock (input only).
41	SDA	BI↑	Serial bus data (bidirectional, open drain).
DIGITAL VIDEO INTERFACE			
39 38 35 34	D[3] D[2] D[1] D[0]]	ODT	Tristateable 4-wire output data bus. D[3] is the most significant bit.
33	QCK	ODT	Tristateable data qualification clock.
40	FST	ODT	Tristateable Frame start signal.
17	OEB	ID↓	Digital output (tristate) enable.
SYSTEM CLOCKS			
31	CLKI	ID	Oscillator input.
32	CLKO	OD	Oscillator output.

Table 23 : VV6404 Pinout

Key			
A	Analogue Input	D	Digital Input
OA	Analogue Output	ID↑	Digital input with internal pull-up
BI	Bidirectional	ID↓	Digital input with internal pull-down
BI↑	Bidirectional with internal pull-up	OD	Digital Output
BI↓	Bidirectional with internal pull-down	ODT	Tristateable Digital Output

Table 24 :

5.5 VV6404 Pinout Diagram

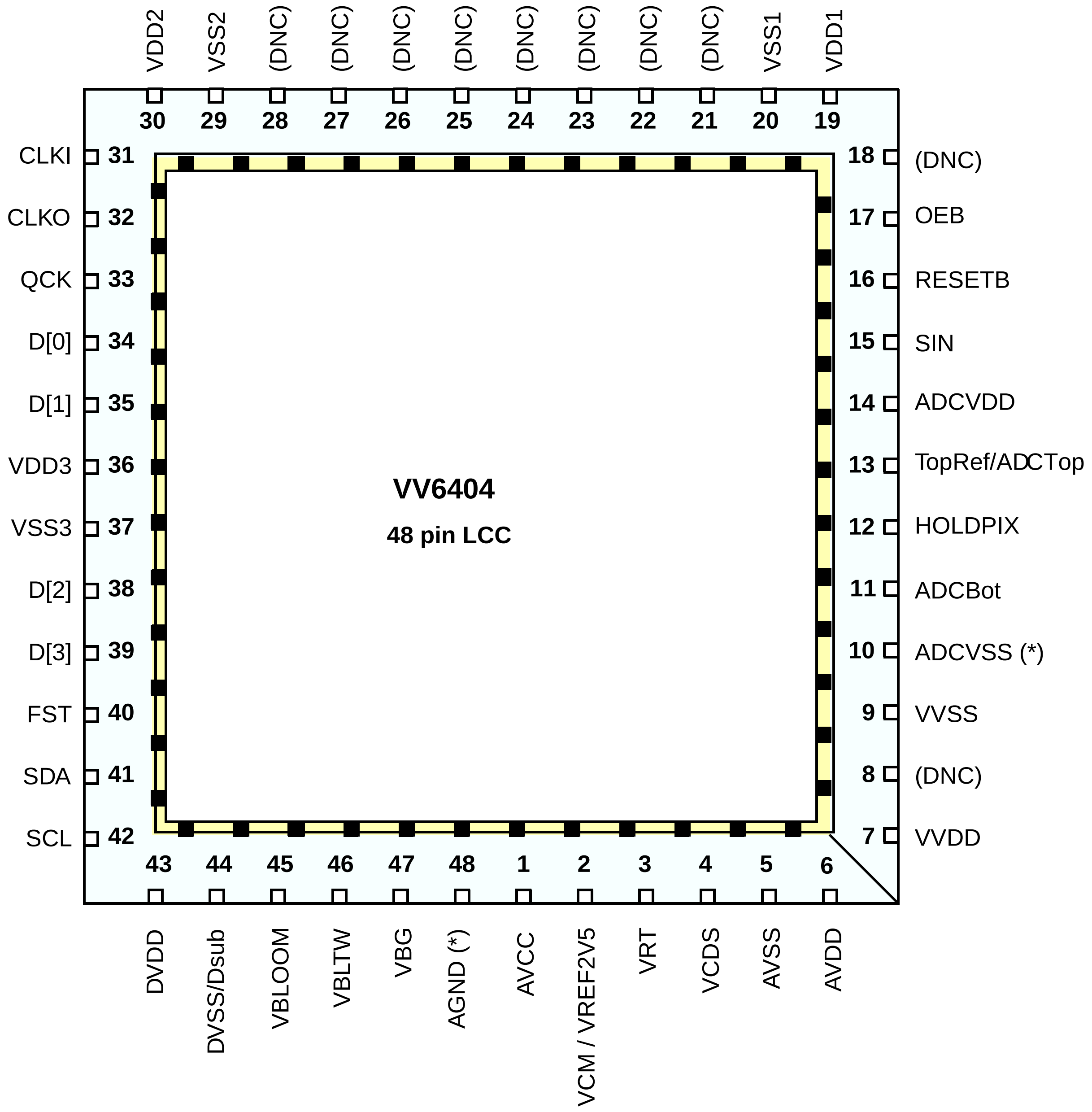


Figure 16 : VV6404 Pinout

Notes:

1. (DNC) - Do not connect these pins
2. (*) - Paddle connections
3. Die size = (7.4530 + 0.220mm) x (6.2142 + 0.270 mm)

5.6 VV6404 Package Dimensions

5.6.1 VV6404 in 48LCC

Notes.

1. Die is optically centred.
2. Refractive index of glass is ~1.52.
3. Distance to optical surface of Die.
4. Pixel area of sensor.

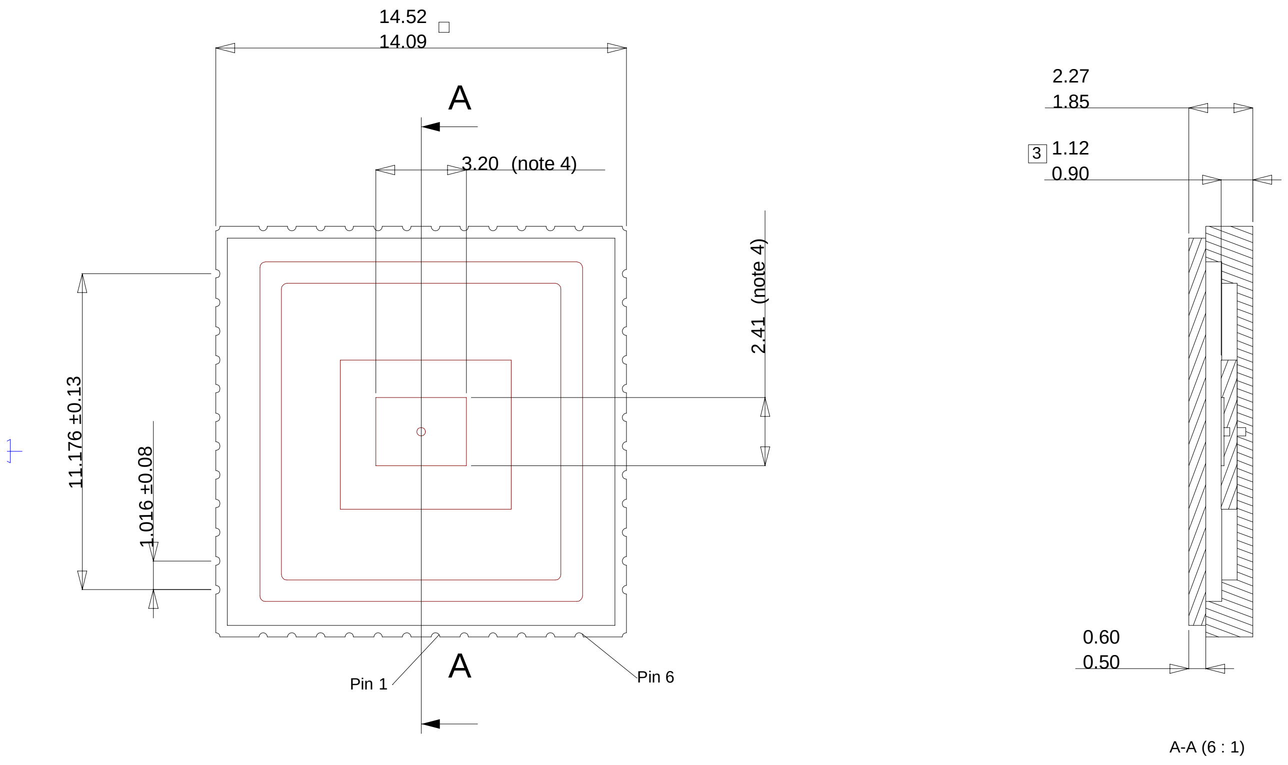


Figure 17 : VV6404 in 48LC

6. Ordering details

For more information on the correct sensor choice please contact STMicroelectronics.

Part number	Description
VV6404C001	CIF image sensor, 48LCC Package, defect-free
VV6404C001-B2	CIF image sensor, 48LCC Package, <37 defects (for use when defect correction is implemented)
STV0673-001	CPiA 1.5 companion co-processor

Table 25 : Ordering details

6.1 STMicroelectronics VV6404 + CPiA1.5 USB Reference Design

The STMicroelectronics VV6404 + CPiA1.5 Reference Design is a complete design package for implementing a production USB camera using this chipset. Complete schematics, gerber layout files, BOM and design recommendations are included in the reference design package. Please contact STMicroelectronics for details.

.

www.vvl.co.uk

www.st.com

asiapacific_sales@vvl.co.uk
central_europe_sales@vvl.co.uk
france_sales@vvl.co.uk
japan_sales@vvl.co.uk
nordic_sales@vvl.co.uk
southern_europe_sales@vvl.co.uk
uk_eire_sales@vvl.co.uk
usa_sales@vvl.co.uk

VLSI VISION LIMITED

A company of the STMicroelectronics Group