Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

MT9F002

1/2.3-inch 14 Mp CMOS Digital Image Sensor

Table 1. KEY PERFORMANCE PARAMETERS

Parameter		Value
Optical format		1/2.3−inch (4:3)
Active pixels and imager size		• 4608 H x 3288 V: (entire array): 6.451 mm (H) x 4.603mm (V), 7.925mm diagonal • 4384 H x 3288 V (4:3, still mode): 6.138 mm (H) x 4.603 mm (V), 7.672 mm diagonal • 4608 H x 2592 V (16:9, video mode): 6.451 mm (H) x 3.629 mm (V), 7.402 mm diagonal
Pixel size		1.4 μm x 1.4 μm
Chief ray angle		0°, 11.4°, and 25°
Color filter array		RGB Bayer pattern
Shutter type		Electronic rolling shutter (ERS) with global reset release (GRR)
Input clock frequency		2–64 MHz
Maximum data rate	Parallel	96 Mp/s at 96 MHz PIXCLK
	HiSPi (4−lane)	700 Mbps/lane
Frame rate	14M resolution (4384H x 3288V)	Programmable up to 13.7 fps for HiSPi I/F, 6.3 fps for parallel I/F
	Preview VGA mode	• 30 fps with binning • 60 fps with skip2bin2
	1080p mode:	• 60 fps using HiSPi interface 2304 H x 1296 V (1080p +20%EIS) • 30 fps using parallel interface 2256 H x 1268 V (1080p +17%EIS)
ADC resolution		12−bit, on−chip
Responsivity		0.724 V/lux−sec (550 nm)
Dynamic range		65.3 dB
SNR_{MAX}		35.5 dB
Supply voltage	I/O Digital	1.7–1.9 V (1.8 V nominal) or 2.4–3.1 V (2.8 V nominal)
	Digital	1.7–1.9 V (1.8 V nominal)
	Analog	2.7–3.1 V (2.8 V nominal)
	HiSPi PHY HiSPi I/O (SLVS) HiSPi I/O (HiVCM)	1.7–1.9 V (1.8 V nominal) 0.3 – 0.9 V (0.4 or 0.8 V nominal) 1.7–1.9 V (1.8 V nominal)
Power Consumption	Full resolution 13.65 fps (HiSPi serial I/F, 12−bit)	724 mW
	1080p60 (HiSPi serial I/F, 10−bit)	XYbin2: 596 mW
	1080p30 (HiSPi serial I/F, 10−bit)	XYbin2: 443 mW
Package		48−pin iLCC (10 mm x 10 mm) and bare die
Operating temperature		−30°C to +70°C (at junction)

ON Semiconductor®

www.onsemi.com

ORDERING INFORMATION

See detailed ordering and shipping information on page 2 of this data sheet.

Features

- 1.4 μm Pixel with ON Semiconductor A−Pix™ Technology
- Simple Two−wire Serial Interface
- Auto Black Level Calibration
- Full HD Support at 60 fps for Maximum Video Performance
- 20 percent Extra Image Array Area in Full HD to Enable Electronic Image Stabilization (EIS)
- Support for External Mechanical Shutter
- Support for External LED or Xenon Flash
- High Frame Rate Preview Mode with Arbitrary Down−size Scaling from Maximum Resolution
- Programmable Controls: Gain, Horizontal and Vertical Blanking, Frame Size/Rate, Exposure, Left−right and Top−bottom Image Reversal, Window Size, and Panning
- Data Interfaces: Parallel or Four−lane Serial Highspeed Pixel Interface (HiSPi™) Differential Signaling (SLVS)
- On−chip Phase−locked Loop (PLL) Oscillator
- Bayer Pattern Downsize Scaler

***Parallel interface does not work for MT9F002 package parts**

Applications

- Digital Video Cameras
- Digital Still Cameras

General Description

The ON Semiconductor MT9F002 is a 1/2.3−inch CMOS active−pixel digital imaging sensor with an active pixel array of 4608 H × 3288 V (4640 H × 3320 V including border pixels). It can support 14−megapixel (4384 H × 3288 V) digital still images and a 1080p plus additional 20 percent pixels for electronic image

stabilization (4608 H × 2592 V) in digital video mode. The MT9F002 sensor is programmable through a simple two−wire serial interface, and has low power consumption.

ORDERING INFORMATION

Table 2. AVAILABLE PART NUMBERS

Part Number	Product Description	Orderable Product Attribute Description
MT9F002I12STCV−DP	RGB, 0° CRA, HiSPi, iLCC Package	Drypack, Protective Film
MT9F002I12−N4000−DP1	RGB, 12° CRA, HiSPi, iLCC Package	Drypack, Protective Film
MT9F002I12STCVH−GEVB	0° CRA, HiSPi, Head Board	
MT9F002I12−N4000H−GEVB	12° CRA, HiSPi, Head Board	
MT9F002D00C2EB−N3003−200	14 MP 1/2.3" CIS Die Sales	200 μm Thickness

GENERAL DESCRIPTION

The MT9F002 digital image sensor features ON Semiconductor breakthrough low−noise CMOS imaging technology that achieves near−CCD image quality (based on signal−to−noise ratio and low−light sensitivity) while maintaining the inherent size, cost, and integration advantages of CMOS.

When operated in its default 4:3 still−mode, the sensor generates a full resolution (4384x3288) image at 13 frames per second (fps) using the HiSPi serial interface. An on−chip analog−to−digital converter (ADC) generates a 12−bit value for each pixel.

FUNCTIONAL OVERVIEW

The MT9F002 is a progressive−scan sensor that generates a stream of pixel data at a constant frame rate. It uses an on−chip, phase−locked loop (PLL) to generate all internal clocks from a single master input clock running between 2 and 64 MHz. The maximum output pixel rate is 220 Mp/s for serial HiSPi I/F and 96 Mp/s for parallel I/F, corresponding to a pixel clock rate of 220 MHz and 96 MHz, respectively. A block diagram of the sensor is shown in Figure 1.

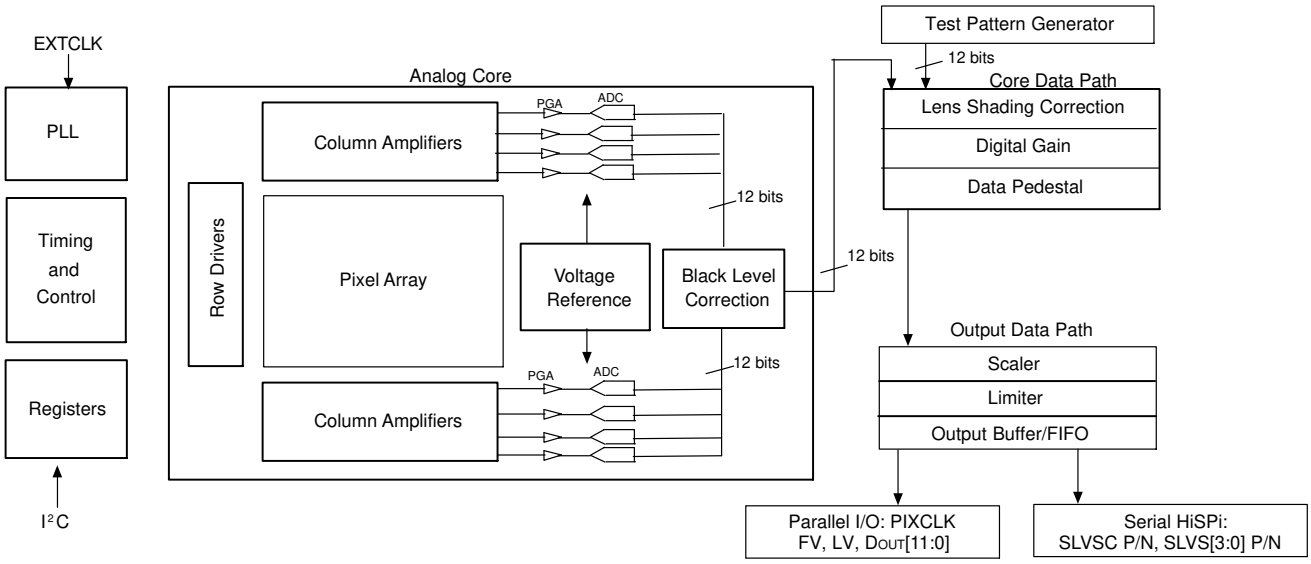


Figure 1. MT9F002 Block Diagram

The core of the sensor is a 14 Mp active−pixel array. The timing and control circuitry sequences through the rows of the array, resetting and then reading each row in turn. In the time interval between resetting a row and reading that row, the pixels in the row integrate incident light. The exposure is controlled by varying the time interval between reset and readout. Once a row has been read, the data from the columns is sequenced through an analog signal chain (providing offset correction and gain), and then through an ADC. The output from the ADC is a 12−bit value for each pixel in the array. The ADC output passes through a digital processing signal chain (which provides further data path corrections and applies digital gain).

The pixel array contains optically active and light−shielded (“dark”) pixels. The dark pixels are used to provide data for on−chip offset−correction algorithms (“black level” control).

The image black level is calibrated to compensate for analog offset and ensure that the ADC range is utilized well. It also reduces row noise in the image. The black level in the output image involves Fine Digital Correction and addition of Data Pedestal (42 LSB for 10−bit ADC, 168 LSB for 12−bit ADC)

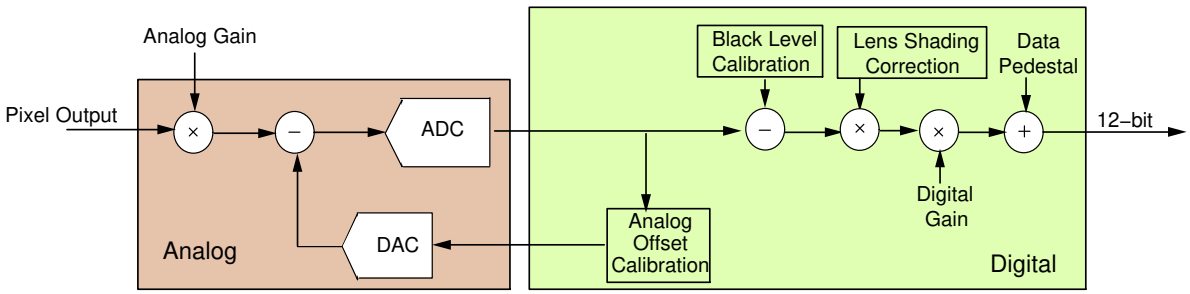


Figure 2. Data Flow Diagram

The sensor contains a set of control and status registers that can be used to control many aspects of the sensor behavior including the frame size, exposure, and gain setting. These registers can be accessed through a two−wire serial interface.

The output from the sensor is a Bayer pattern; alternate rows are a sequence of either green and red pixels or blue and green pixels. The offset and gain stages of the analog signal chain provide per−color control of the pixel data.

The control registers, timing and control, and digital processing functions shown in Figure 1 on page 2 are partitioned into three logical parts:

- A sensor core that provides array control and data path corrections. The output of the sensor core is a 12−bit parallel pixel data stream qualified by an output data clock (PIXCLK), together with LINE_VALID (LV) and FRAME_VALID (FV) signals or a 4−lane serial high−speed pixel interface (HiSPi).
- A digital shading correction block to compensate for color/brightness shading introduced by the lens or chief ray angle (CRA) curve mismatch.
- Additional functionality is provided. This includes a horizontal and vertical image scaler, a limiter, an output FIFO, and a serializer.

The output FIFO is present to prevent data bursts by keeping the data rate continuous. Programmable slew rates are also available to reduce the effect of electromagnetic interference from the output interface.

A flash output signal is provided to allow an external xenon or LED light source to synchronize with the sensor exposure time. Additional I/O signals support the provision of an external mechanical shutter.

Pixel Array

The sensor core uses a Bayer color pattern, as shown in Figure 3. The even−numbered rows contain green and red pixels; odd−numbered rows contain blue and green pixels. Even−numbered columns contain green and blue pixels; odd−numbered columns contain red and green pixels.

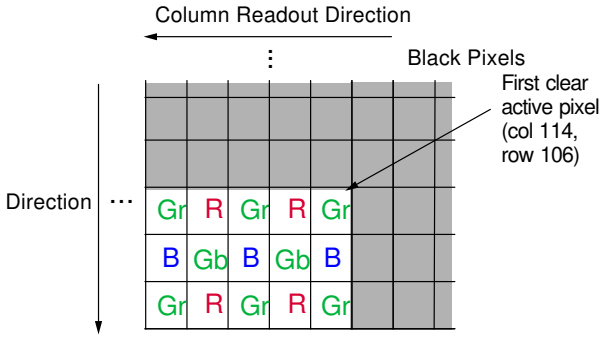


Figure 3. Pixel Color Pattern Detail (Top Right Corner)

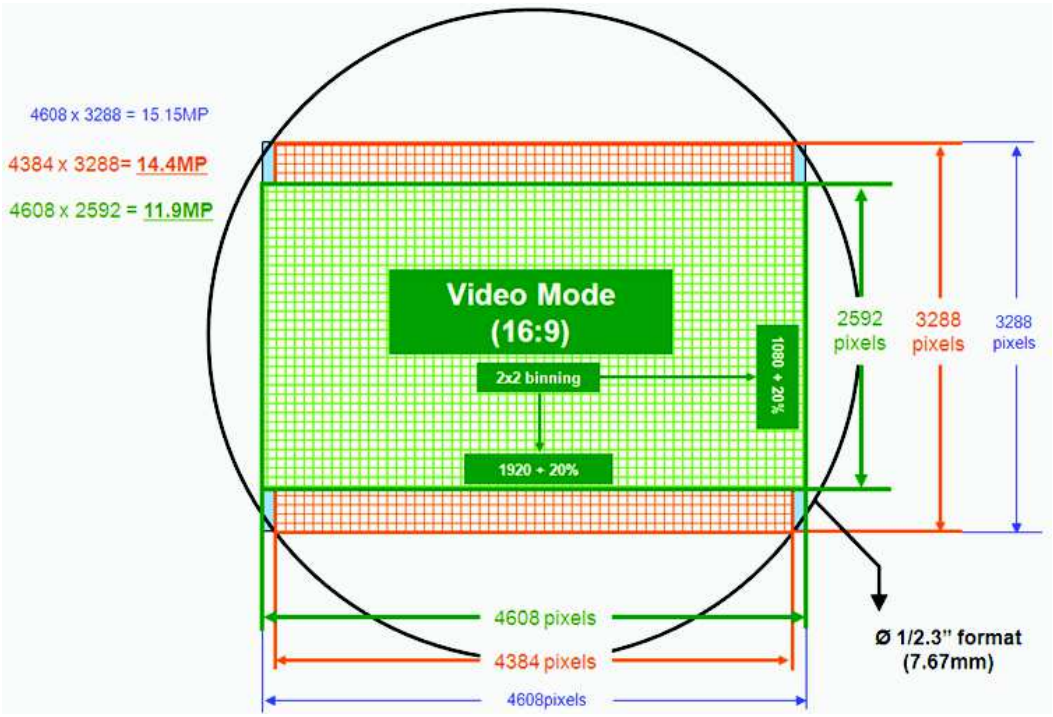


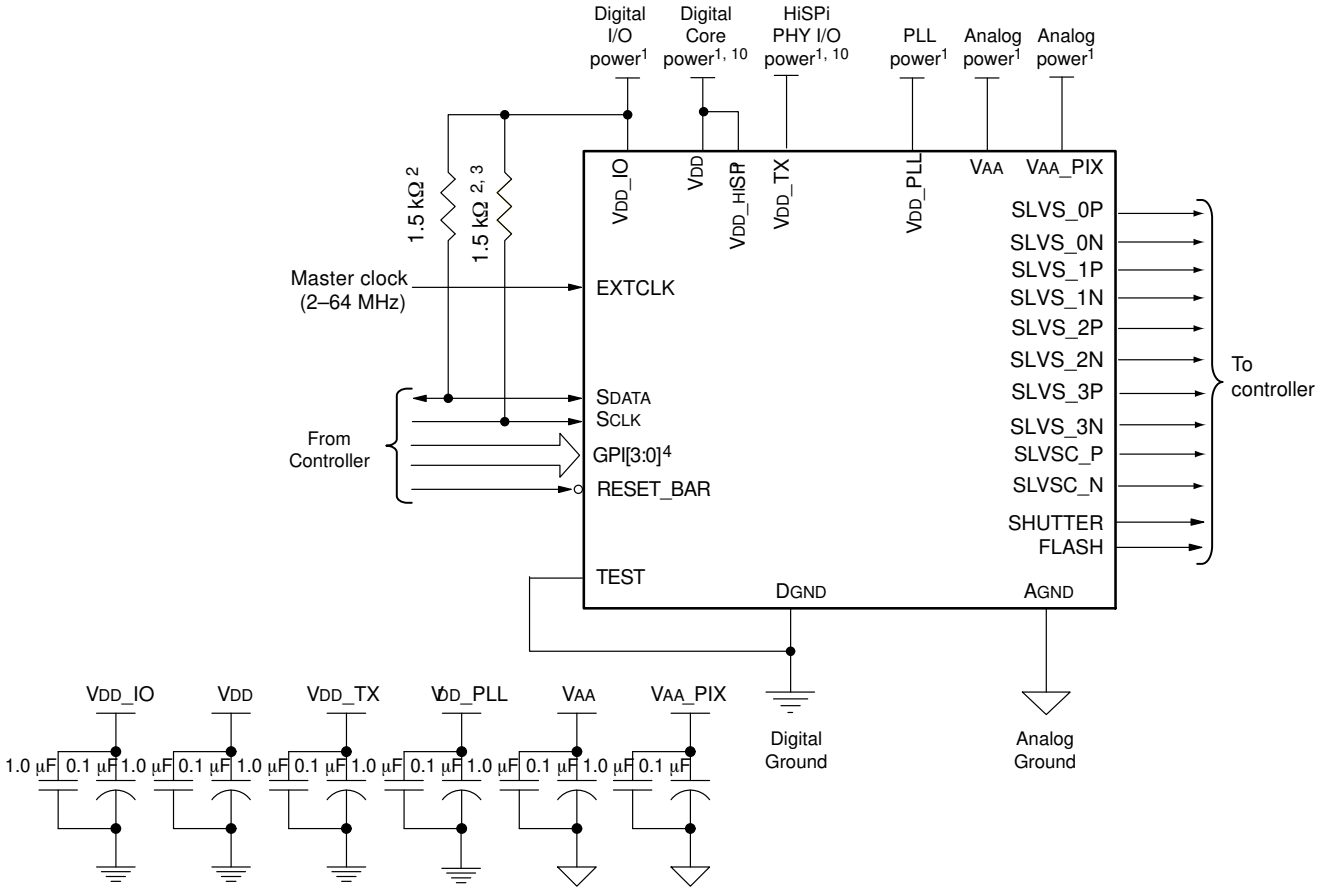
Figure 4. High−Resolution Still Image Capture + HD Video

OPERATING MODES

By default, the MT9F002 powers up with the serial pixel data interface enabled. The sensor can operate in serial HiSPi or parallel mode.

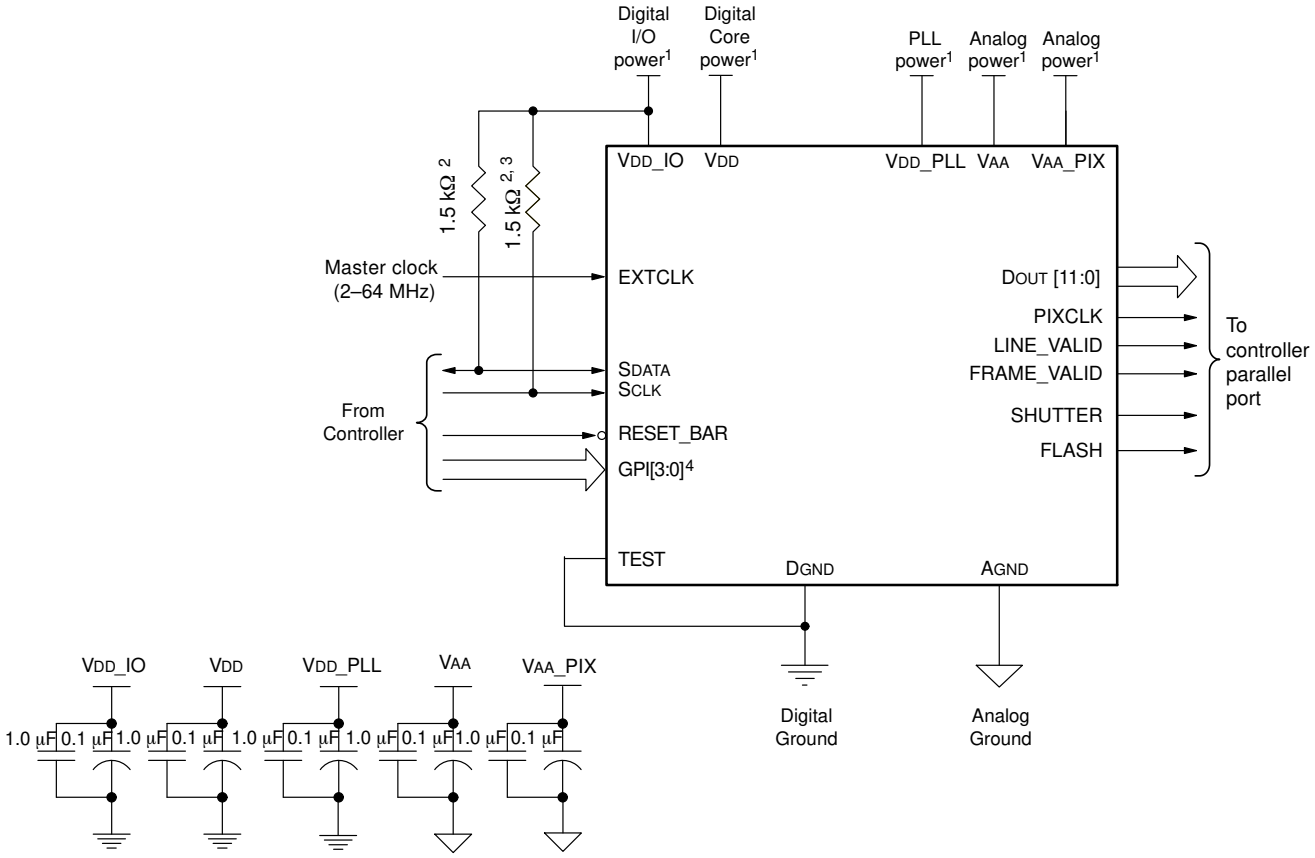
For low−noise operation, the MT9F002 requires separate power supplies for analog and digital power. Incoming digital and analog ground conductors should be placed in such a way that coupling between the two are minimized. Both power supply rails should also be routed in such a way that noise coupling between the two supplies and ground is minimized.

CAUTION: ON Semiconductor does not recommend the use of inductance filters on the power supplies or output signals.



Notes:
1. All power supplies should be adequately decoupled. ON Semiconductor recommends having 1.0 µF and 0.1 µF decoupling capacitors for every power supply.
2. ON Semiconductor recommends a resistor value of 1.5 kΩ, but a greater value may be used for slower two−wire speed.
3. This pull−up resistor is not required if the controller drives a valid logic level on S_{CLK} at all times.
4. The GPI pins can be statically pulled HIGH or LOW and can be programmed to perform special functions (TRIGGER/VD, OE_BAR, S_{ADDR}, STANDBY) to be dynamically controlled. GPI pads can be left floating, when not used.
5. V_{PP}, which is not shown in Figure 5, is left unconnected during normal operation.
6. The parallel interface output pads can be left unconnected when the serial output interface is used.
7. ON Semiconductor recommends that 0.1 µF and 10 µF decoupling capacitors for each power supply are mounted as close as possible to the pad. Actual values and results may vary depending on layout and design considerations. Check the MT9F002 evaluation headboard schematics for circuit recommendations.
8. TEST signals must be tied to D_{GND} for normal sensor operation.
9. ON Semiconductor recommends that analog power planes are placed in a manner such that coupling with the digital power planes is minimized.
10. For serial HiSPi HiVCM mode, set register bit R0x306E[9] = 1 and $V_{DD}\_TX = V_{DD}\_IO = 1.8$ V.

Figure 5. Typical Configuration: Serial Four−Lane HiSPi Interface



Notes:
1. All power supplies should be adequately decoupled. ON Semiconductor recommends having 1.0 μF and 0.1 μF decoupling capacitors for every power supply.
2. ON Semiconductor recommends a resistor value of 1.5 kΩ, but a greater value may be used for slower two−wire speed.
3. This pull−up resistor is not required if the controller drives a valid logic level on S_{CLK} at all times.
4. The GPI pins can be statically pulled HIGH or LOW and can be programmed to perform special functions (TRIGGER/VD, OE_BAR, S_{ADDR}, STANDBY) to be dynamically controlled. GPI pads can be left floating, when not used.
5. V_{PP}, which is not shown in Figure 6, is left unconnected during normal operation.
6. The serial interface output pads can be left unconnected when the parallel output interface is used.
7. ON Semiconductor recommends that 0.1 μF and 10 μF decoupling capacitors for each power supply are mounted as close as possible to the pad. Actual values and results may vary depending on layout and design considerations. Check the MT9F002 evaluation headboard schematics for circuit recommendations.
8. TEST signals must be tied to D_{GND} for normal sensor operation.
9. ON Semiconductor recommends that analog power planes are placed in a manner such that coupling with the digital power planes is minimized.

Figure 6. Typical Configuration: Parallel Pixel Data Interface (Die Only)

SIGNAL DESCRIPTIONS

Table 3 provides signal descriptions for MT9F002 die. For pad location and aperture information, refer to the MT9F002 die data sheet.

Table 3. SIGNAL DESCRIPTIONS

Signal	Type	Description
EXTCLK	Input	Master clock input, 2–64 MHz.
RESET_BAR	Input	Asynchronous active LOW reset. When asserted, data output stops and all internal registers are restored to their factory default settings.
S_{CLK}	Input	Serial clock for access to control and status registers.
GPI[3:0]	Input	General purpose inputs. After reset, these pads are powered−down by default; this means that it is not necessary to bond to these pads. Any of these pads can be programmed (through register R0x3026) to provide hardware control of the standby, output enable, S_{ADDR} select, shutter trigger or slave mode trigger (VD) function. Can be left floating if not used.
TEST	Input	Enable manufacturing test modes. Tie to D_{GND} for normal sensor operation.
S_{DATA}	I/O	Serial data from READs and WRITEs to control and status registers.
V_{PP}	Supply	Disconnect pad for normal operation. Power supply used to program one−time programmable (OTP) memory. Manufacturing use only.
V_{DD}_HiSPi	Supply	HiSPi PHY power supply. Digital power supply for the HiSPi serial data interface. This should be tied to V_{DD}
V_{DD}_TX	Supply	Digital power supply for the HiSPi I/O. For HiSPi SLVS mode, set register bit R0x306E[9] = 0 (default), and VDD_TX to 0.4 V. For HiSPi HiVCM mode, set register bit R0x306E[9] = 1, and VDD_TX = VDD_IO.
V_{AA}	Supply	Analog power supply.
V_{AA}_PIX	Supply	Analog power supply for the pixel array.
A_{GND}	Supply	Analog ground.
V_{DD}	Supply	Digital power supply.
V_{DD}_IO	Supply	I/O power supply.
D_{GND}	Supply	Common ground for digital and I/O.
V_{DD}_PLL	Supply	PLL power supply.
SLVS_0P	Output	Lane 1 differential HiSPi (SLVS) serial data (positive). Qualified by the SLVS serial clock.
SLVS_0N	Output	Lane 1 differential HiSPi (SLVS) serial data (negative). Qualified by the SLVS serial clock.
SLVS_1P	Output	Lane 2 differential HiSPi (SLVS) serial data (positive). Qualified by the SLVS serial clock.
SLVS_1N	Output	Lane 2 differential HiSPi (SLVS) serial data (negative). Qualified by the SLVS serial clock.
SLVS_2P	Output	Lane 3 differential HiSPi (SLVS) serial data (positive). Qualified by the SLVS serial clock.
SLVS_2N	Output	Lane 3 differential HiSPi (SLVS) serial data (negative). Qualified by the SLVS serial clock.
SLVS_3P	Output	Lane 4 differential HiSPi (SLVS) serial data (positive). Qualified by the SLVS serial clock.
SLVS_3N	Output	Lane 4 differential HiSPi (SLVS) serial data (negative). Qualified by the SLVS serial clock.
SLVS_CP	Output	Differential HiSPi (SLVS) serial clock (positive). Qualified by the SLVS serial clock.
SLVS_CN	Output	Differential HiSPi (SLVS) serial clock (negative). Qualified by the SLVS serial clock.
LINE_VALID	Output	LINE_VALID (LV) output. Qualified by PIXCLK.
FRAME_VALID	Output	FRAME_VALID (FV) output. Qualified by PIXCLK.
D_{OUT}[11:0]	Output	Parallel pixel data output. Qualified by PIXCLK.
PIXCLK	Output	Pixel clock. Used to qualify the LV, FV, and D_{OUT}[11:0] outputs.
FLASH	Output	Flash output. Synchronization pulse for external light source. Can be left floating if not used.
SHUTTER	Output	Control for external mechanical shutter. Can be left floating if not used.

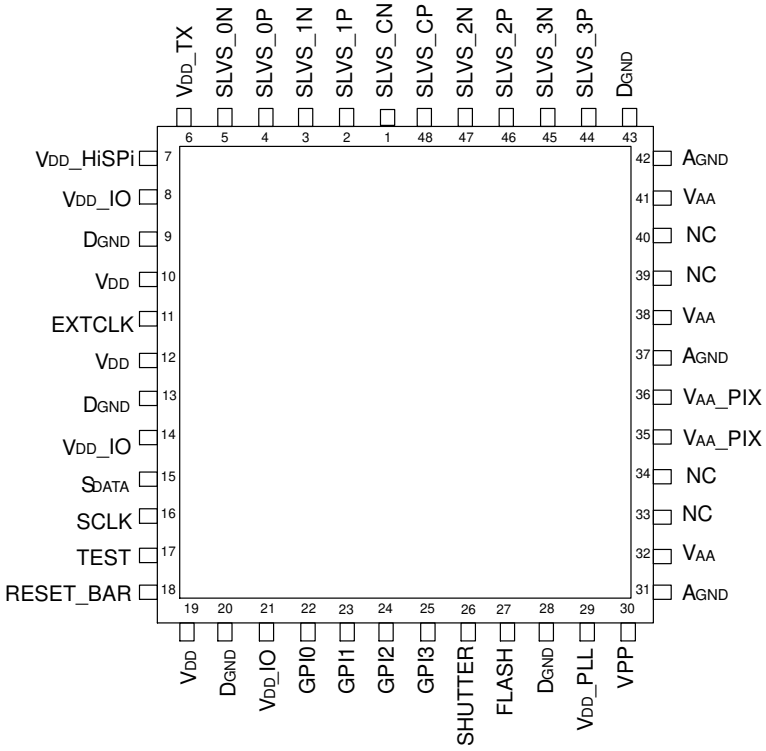


Figure 7. 48−Pin ILCC HiSPi Package Pinout Diagram

OUTPUT DATA FORMAT

Pixel Data Interface

The MT9F002 reads data out of the pixel array in a progressive scan over a High Speed serial data interface, or parallel data interface. RAW8, RAW10, and RAW12 image data formats are supported.

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	RAW12
D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	X	X	RAW10
D7	D6	D5	D4	D3	D2	D1	D0	X	X	X	X	RAW8

Figure 8. Data Formats

High Speed Serial Pixel Data Interface

The High Speed Serial Pixel (HiSPi) interface uses four data and one clock low voltage differential signaling (SLVS) outputs.

- SLVS_CP
- SLVS_CN
- SLVS_0P
- SLVS_0N
- SLVS_1P
- SLVS_1N
- SLVS_2P
- SLVS_2N
- SLVS_3P
- SLVS_3N

The HiSPi interface supports the following protocols: Streaming−S and Packetized−SP. The streaming protocol conforms to a standard video application where each line of active or intra−frame blanking provided by the sensor is transmitted at the same length. The packetized protocol will transmit only the active data ignoring line−to−line and frame−to−frame blanking data.

HiSPi Streaming Mode Protocol Layer

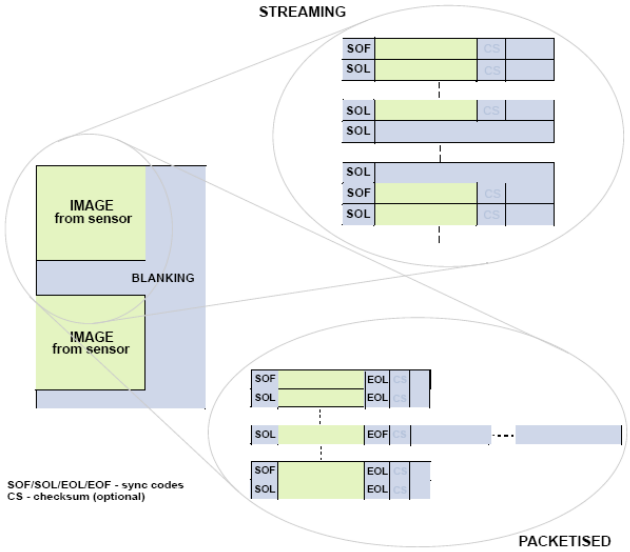
The protocol layer is positioned between the output data path of the sensor and the physical layer. The main functions of the protocol layer are generating sync codes, formatting pixel data, inserting horizontal/vertical blanking codes, and distributing pixel data over defined data lanes.

The HiSPi interface can only be configured when the sensor is in standby. This includes configuring the interface to transmit across 1, 2, or all 4 data lanes.

Protocol Fundamentals

Referring to Figure 9, it can be seen that a SYNC code is inserted in the serial data stream prior to each line of image data. The streaming protocol will insert a SYNC code to transmit each active data line and vertical blanking lines.

The packetized protocol will transmit a SYNC code to note the start and end of each row. The packetized protocol uses sync a “Start of Frame” (SOF) sync code at the start of a frame and a “Start of Line” (SOL) sync code at the start of a line within the frame. The protocol will also transmit an “End of Frame” (EOF) at the end of a frame and an “End of Line” (EOL) sync code at the end of a row within the frame.



Note: See the High−Speed Serial Pixel (HiSPi) Protocol Specification V1.00.00 for HiSPi details.

Figure 9. Streaming vs. Packetized Transmission

HiSPi Physical Layer

The HiSPi physical layer is partitioned into blocks of four data lanes and an associated clock lane. Any reference to the PHY in the remainder of this document is referring to this minimum building block.

The HiSPi PHY uses a low voltage serial differential output. The HiSPi PHY drivers use a simple current steering driver scheme with two outputs that are complementary to each other (V_{OA} and V_{OB}). It is intended that these drivers be attached to short−length 100 Ω differential interconnect to a receiver with a 100 Ω termination. CL represents the total parasitic excess capacitance loading of the receiver and the interconnect.

There are two standards:

- Scalable Low Voltage Serial (SLVS) which has low amplitude and common−mode voltage (VCM) but scalable using an external supply.
- High VCM scalable serial interface (HiVCM), which has larger scalable amplitude and a high common−mode voltage.

Comparison of SLVS and HiVCM

Here is a comparison of the differences between SLVS and HiVCM.

Table 4. SLVS AND HiVCM COMPARISON

Parameter	HiVCM	SLVS
Typical Differential Amplitude[1]	280 mV	200 mV
Typical Common Mode[1]	0.9 V	200 mV
Typical Power Consumption[2]	45 mW	4 mW
Transmission Distance	Longer distance	Short distance
LVDS FPGA Receiver Compatible	Yes	No

1. These are nominal values.
2. Power from load driving stage, digital/serializer logic (VDD_HiSPi) not included.

The HiSPi interface building block is a unidirectional differential serial interface with four data and one double data rate (DDR) clock lanes. The four Data lanes are 90

degrees out of phase with the Clock lanes. One clock for every four serial data lanes is provided for phase alignment across multiple lanes. Figure 10 shows the configuration between the HiSPi transmitter and the receiver.

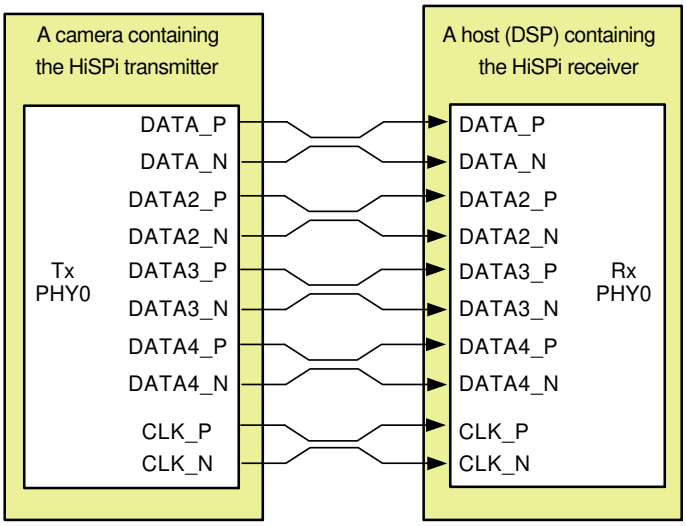


Figure 10. HiSPi Transmitter and Receiver Interface Block Diagram

The PHY will serialize a 10–, 12–, 14– or 16–bit data word and transmit each bit of data centered on a rising edge of the clock, the second on the falling edge of clock. Figure 11 shows bit transmission. In this example, the word is transmitted in order of MSB to LSB. The receiver latches data at the rising and falling edge of the clock.

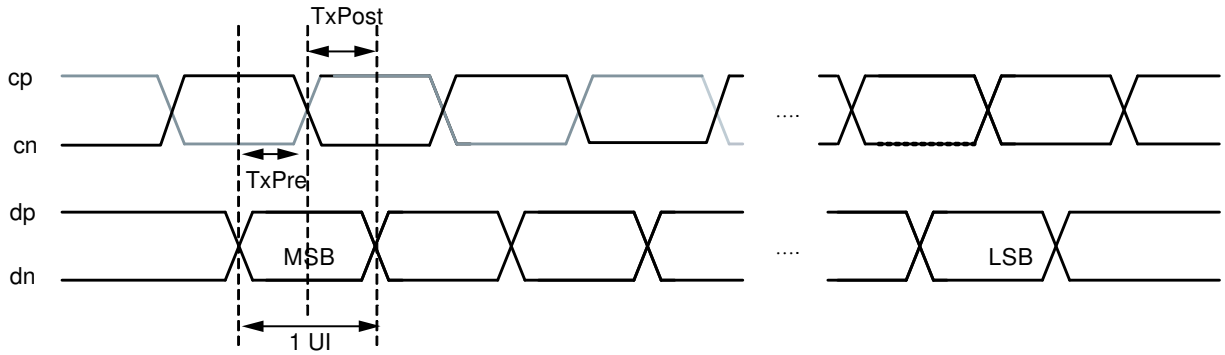


Figure 11. Timing Diagram

DLL Timing Adjustment

The specification includes a DLL to compensate for differences in group delay for each data lane. The DLL is connected to the clock lane and each data lane, which acts as a control master for the output delay buffers. Once the DLL has gained phase lock, each lane can be delayed in 1/8 unit interval (UI) steps. This additional delay allows the user to increase the setup or hold time at the receiver circuits and can be used to compensate for skew introduced in PCB design.

If the DLL timing adjustment is not required, the data and clock lane delay settings should be set to a default code of 0x000 to reduce jitter, skew, and power dissipation.

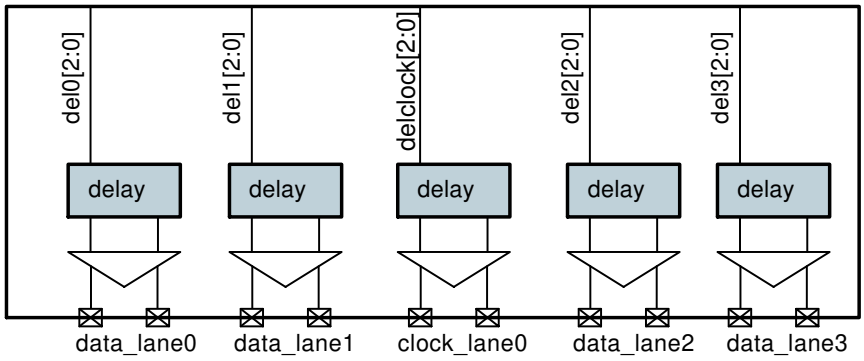


Figure 12. Block Diagram of DLL Timing Adjustment

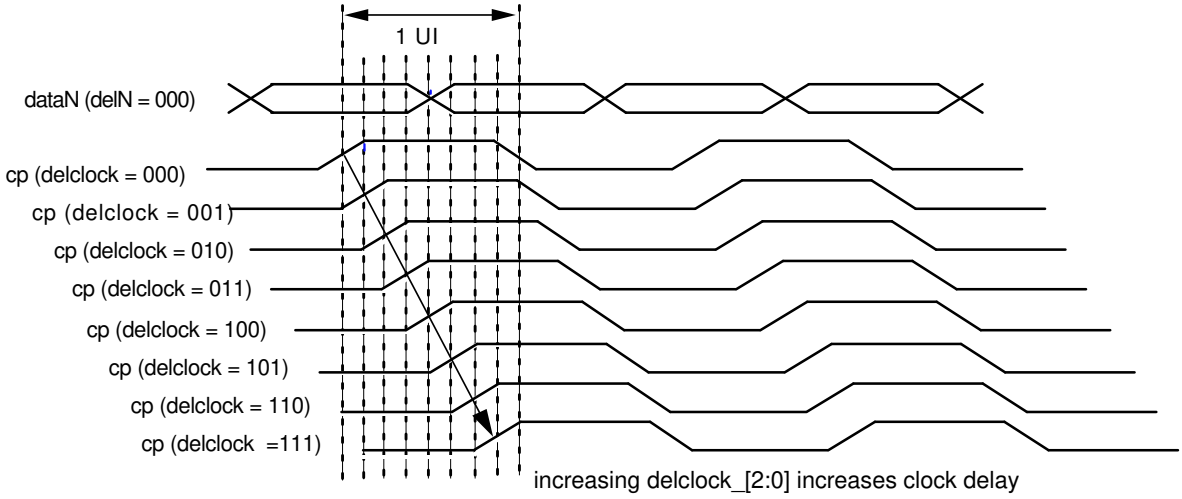
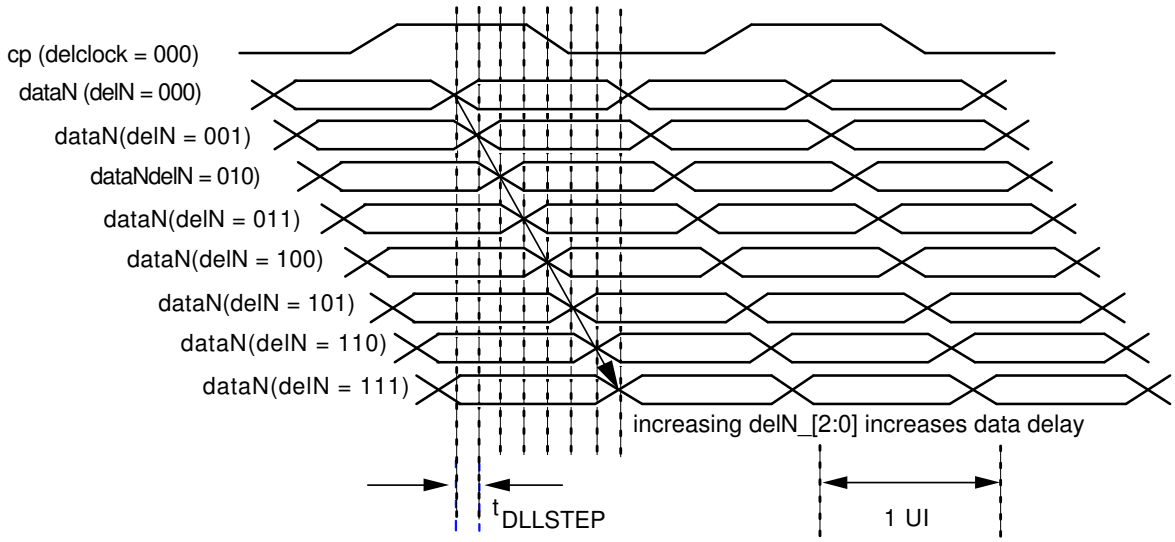


Figure 13. Delaying the clock_lane with Respect to data_lane



Note: See the High−Speed Serial Pixel (HiSPi) Physical Layer Specification V2.00.00 for details.

Figure 14. Delaying data_lane with Respect to the clock_lane

Parallel Pixel Data interface

MT9F002 image data is read out in a progressive scan. Valid image data is surrounded by horizontal blanking and vertical blanking, as shown in Figure 15. The amount of horizontal blanking and vertical blanking is programmable; LV is HIGH during the shaded region of the figure. FV timing is described in the “Output Data Timing (Parallel Pixel Data Interface)”.

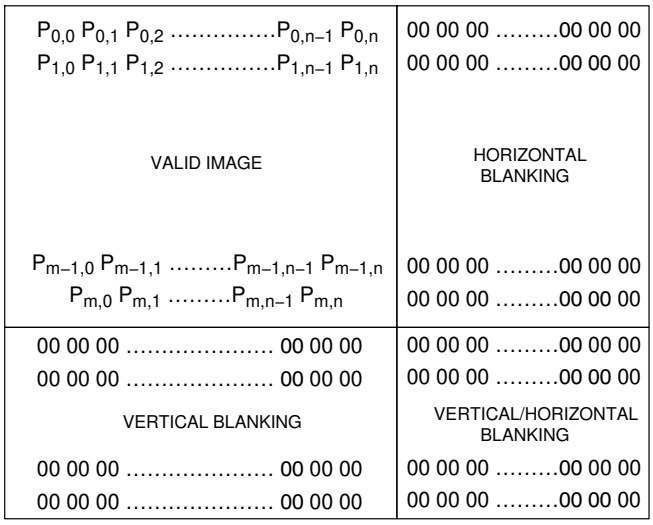


Figure 15. Spatial Illustration of Image Readout

Output Data Timing (Parallel Pixel Data Interface)

MT9F002 output data is synchronized with the PIXCLK output. When LV is HIGH, one pixel value is output on the 12−bit DOUT output every PIXCLK period. The pixel clock frequency can be determined based on the sensor’s master input clock and internal PLL configuration. The rising edges on the PIXCLK signal occurs one−half of a pixel clock period after transitions on LV, FV, and DOUT (see Figure 16). This allows PIXCLK to be used as a clock to sample the data. PIXCLK is continuously enabled, even during the blanking period. The MT9F002 can be programmed to delay the PIXCLK edge relative to the DOUT transitions. This can be achieved by programming the corresponding bits in the row_speed register.

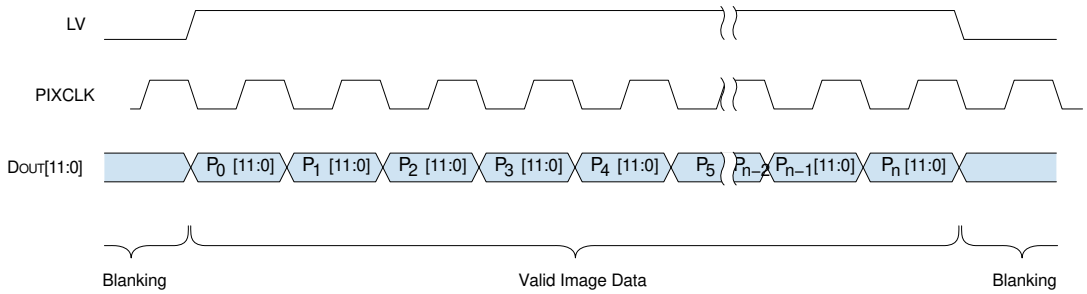


Figure 16. Pixel Data Timing Example

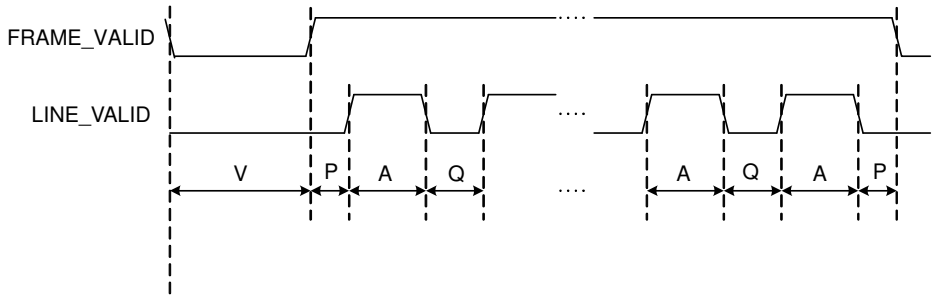


Figure 17. Frame Timing and FV/LV Signals

The sensor timing is shown in terms of pixel clock cycles (see Figure 16). The default settings for the on−chip PLL generate a pixel array clock (vt_pix_clk) of 110 MHz and an output clock (op_pix_clk) of 55 MHz given a 24 MHz input clock to the MT9F002. Equations for calculating the frame rate are given in “Frame Rate Control” on page 48.

Table 5. COMMON SENSOR READOUT MODES

Key Readout Modes	Output Resolution	Aspect Ratio	DFOV: 7.67 mm (%)	Subsampling Mode	Frame Rate	ADC Effective Bit−Depth	Data Rate (Mbps/Lane)
14M Capture	4384 H x 3288 V	(4:3)	100	n/a	13.7	12	660
1080p +20% EIS (3 Mp) Video	2304 H x 1296 V	(16:9)	96	x: Bin2 y: Bin2	60	10	550
	2304 H x 1296 V	(16:9)	96	x: Bin2 y: Bin2	30	10	275
720p +20% EIS (1.3 Mp) Video	1536 H x 864 V	(16:9)	64	x: Bin2 y: Bin2	60	10	550
	1536 H x 864 V	(16:9)	64	x: Bin2 y: Bin2	30	10	275
VGA Video (High Quality)	1096 H x 822 V	(4:3)	100	x: Skip2Bin2 y: Bin4	60	10	550
EVF1 – Preview (Low Power)	1096 H x 822 V	(4:3)	100	x: Skip2Bin2 y: Bin4	30	10	275
EVF2 – Preview (Low Power)	1152 H x 648 V	(16:9)	96	x: Skip2Bin2 y: Bin4	30	10	275

TWO−WIRE SERIAL REGISTER INTERFACE

The two−wire serial interface bus enables read/write access to control and status registers within the MT9F002.

The interface protocol uses a master/slave model in which a master controls one or more slave devices. The sensor acts as a slave device. The master generates a clock (S_{CLK}) that is an input to the sensor and is used to synchronize transfers. Data is transferred between the master and the slave on a bidirectional signal (SDATA). SDATA is pulled up to V_{DD_IO} off−chip by a 1.5 kΩ resistor. Either the slave or master device can drive SDATA LOW—the interface protocol determines which device is allowed to drive SDATA at any given time.

The protocols described in the two−wire serial interface specification allow the slave device to drive S_{CLK} LOW; the MT9F002 uses S_{CLK} as an input only and therefore never drives it LOW.

Protocol

Data transfers on the two−wire serial interface bus are performed by a sequence of low−level protocol elements:

1. a (repeated) start condition
2. a slave address/data direction byte
3. an (a no−) acknowledge bit
4. a message byte
5. a stop condition

The bus is idle when both S_{CLK} and SDATA are HIGH. Control of the bus is initiated with a start condition, and the bus is released with a stop condition. Only the master can generate the start and stop conditions.

Start Condition

A start condition is defined as a HIGH−to−LOW transition on SDATA while S_{CLK} is HIGH. At the end of a transfer, the master can generate a start condition without previously generating a stop condition; this is known as a “repeated start” or “restart” condition.

Stop Condition

A stop condition is defined as a LOW−to−HIGH transition on SDATA while S_{CLK} is HIGH.

Data Transfer

Data is transferred serially, 8 bits at a time, with the MSB transmitted first. Each byte of data is followed by an acknowledge bit or a no−acknowledge bit. This data transfer mechanism is used for both the slave address/data direction byte and for message bytes.

One data bit is transferred during each SCLK clock period. SDATA can change when SCLK is LOW and must be stable while SCLK is HIGH.

Slave Address/Data Direction Byte

Bits [7:1] of this byte represent the device slave address and bit [0] indicates the data transfer direction. A “0” in bit [0] indicates a WRITE, and a “1” indicates a READ. The default slave addresses used by the MT9F002 sensor are 0x20 (write address) and 0x21 (read address). Alternative slave addresses of 0x30 (write address) and 0x31 (read address) can be selected by enabling and asserting the SADDR signal through the GPI pin.

Alternate slave addresses can also be programmed through the i2c_ids register (R0x31FC−31FD). Note that this register needs to be unlocked through reset_register_lock_reg (R0x301A[3]) before is can be written to..

Message Byte

Message bytes are used for sending register addresses and register write data to the slave device and for retrieving register read data.

Acknowledge Bit

Each 8−bit data transfer is followed by an acknowledge bit or a no−acknowledge bit in the SCLK clock period following the data transfer. The transmitter (which is the master when writing, or the slave when reading) releases SDATA. The receiver indicates an acknowledge bit by driving SDATA LOW. As for data transfers, SDATA can change when SCLK is LOW and must be stable while SCLK is HIGH.

No−Acknowledge Bit

The no−acknowledge bit is generated when the receiver does not drive SDATA LOW during the SCLK clock period following a data transfer. A no−acknowledge bit is used to terminate a read sequence.

Typical Sequence

A typical READ or WRITE sequence begins by the master generating a start condition on the bus. After the start condition, the master sends the 8−bit slave address/data direction byte. The last bit indicates whether the request is for a read or a write, where a "0" indicates a write and a "1" indicates a read. If the address matches the address of the slave device, the slave device acknowledges receipt of the address by generating an acknowledge bit on the bus.

If the request was a WRITE, the master then transfers the 16−bit register address to which the WRITE should take place. This transfer takes place as two 8−bit sequences and the slave sends an acknowledge bit after each sequence to indicate that the byte has been received. The master then transfers the data as an 8−bit sequence; the slave sends an acknowledge bit at the end of the sequence. The master stops writing by generating a (re)start or stop condition.

If the request was a READ, the master sends the 8−bit write slave address/data direction byte and 16−bit register address, the same way as with a WRITE request. The master then generates a (re)start condition and the 8−bit read slave address/data direction byte, and clocks out the register data, eight bits at a time. The master generates an acknowledge bit after each 8−bit transfer. The slave's internal register address is automatically incremented after every 8 bits are transferred. The data transfer is stopped when the master sends a no−acknowledge bit.

Single READ From Random Location

This sequence (Figure 18) starts with a dummy WRITE to the 16−bit address that is to be used for the READ. The master terminates the WRITE by generating a restart condition. The master then sends the 8−bit read slave address/data direction byte and clocks out one byte of register data. The master terminates the READ by generating a no−acknowledge bit followed by a stop condition. Figure 18 shows how the internal register address maintained by the MT9F002 is loaded and incremented as the sequence proceeds.

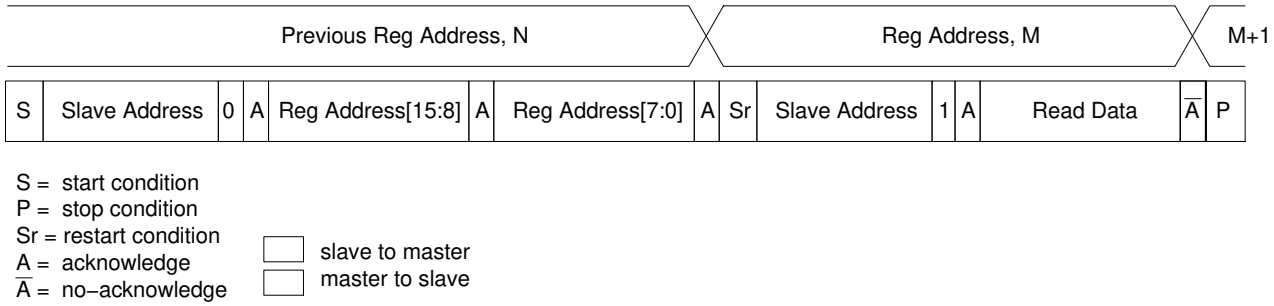


Figure 18. Single READ from Random Location

Single READ From Current Location

This sequence (Figure 19) performs a read using the current value of the MT9F002 internal register address. The master terminates the READ by generating a no−acknowledge bit followed by a stop condition. The figure shows two independent READ sequences.

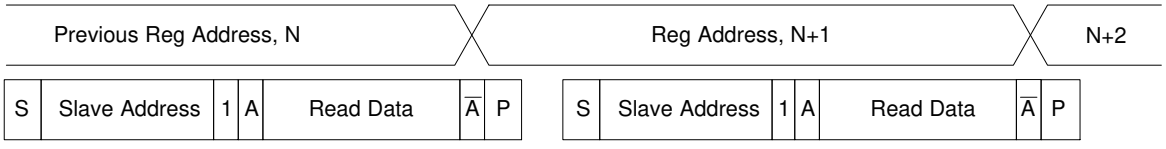


Figure 19. Single READ from Current Location

Sequential READ, Start From Random Location

This sequence (Figure 20) starts in the same way as the single READ from random location (Figure 18). Instead of generating a no−acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte READs until “L” bytes have been read.

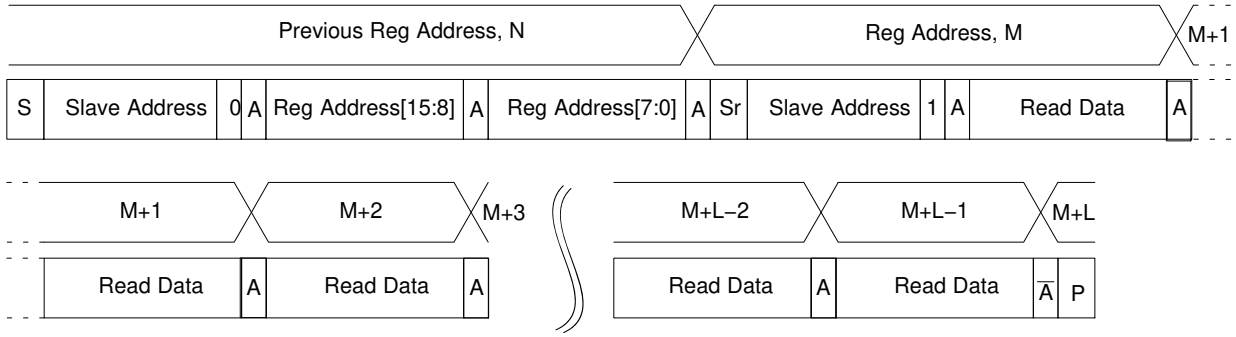


Figure 20. Sequential READ, Start from Random Location

Sequential READ, Start From Current Location

This sequence (Figure 21) starts in the same way as the single READ from current location (Figure 19). Instead of generating a no−acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte READs until “L” bytes have been read.

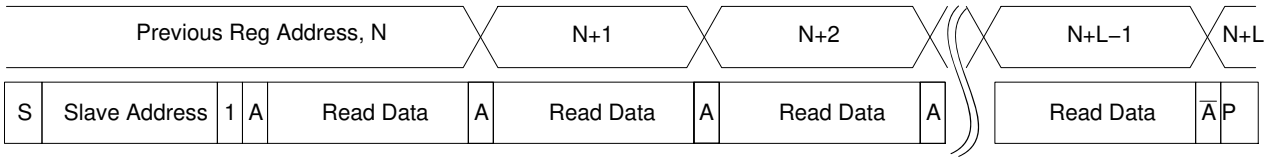


Figure 21. Sequential READ, Start from Current Location

Single WRITE to Random Location

This sequence (Figure 22) begins with the master generating a start condition. The slave address/data direction byte signals a WRITE and is followed by the HIGH then LOW bytes of the register address that is to be written. The master follows this with the byte of write data. The WRITE is terminated by the master generating a stop condition.

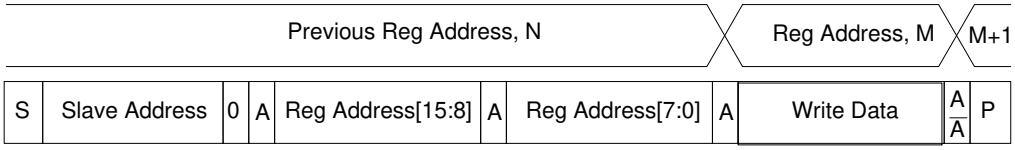


Figure 22. Single WRITE to Random Location

Sequential WRITE, Start at Random Location

This sequence (Figure 23) starts in the same way as the single WRITE to random location (Figure 22). Instead of generating a no−acknowledge bit after the first byte of data has been transferred, the master generates an acknowledge bit and continues to perform byte WRITEs until “L” bytes have been written. The WRITE is terminated by the master generating a stop condition.

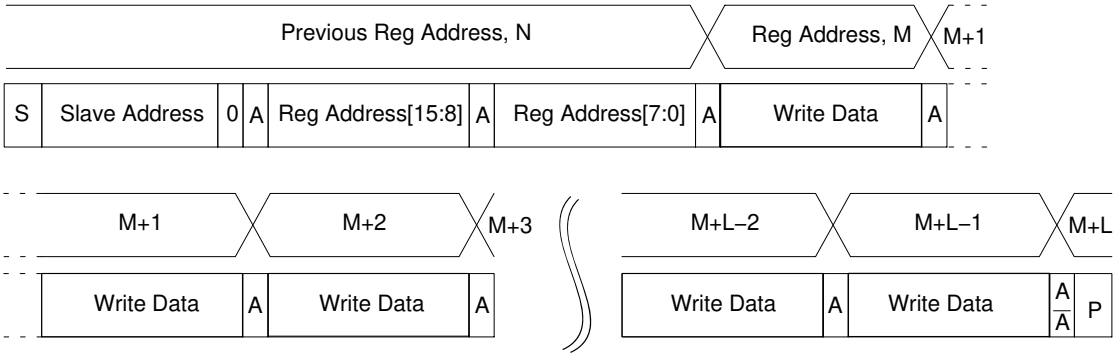


Figure 23. Single WRITE to Random Location

PROGRAMMING RESTRICTIONS

The following sections list programming rules that must be adhered to for correct operation of the MT9F002. Refer to the MT9F002 Register Reference document for register programming details.

Table 6. DEFINITIONS FOR PROGRAMMING RULES

Name	Definition
xskip	xskip = 1 if x_odd_inc = 1; xskip = 2 if x_odd_inc = 3; xskip = 4 if x_odd_inc = 7
yskip	yskip = 1 if y_odd_inc = 1; yskip = 2 if y_odd_inc = 3; yskip = 4 if y_odd_inc = 7; yskip = 8 if y_odd_inc = 15; yskip = 16 if y_odd_inc = 31; yskip = 32 if y_odd_inc = 63

X Address Restrictions

The minimum column address available for the sensor is 24. The maximum value is 4647.

Effect of Scaler on Legal Range of Output Sizes

When the scaler is enabled, it is necessary to adjust the values of x_output_size and y_output_size to match the image size generated by the scaler. The MT9F002 will operate incorrectly if the x_output_size and y_output_size are significantly larger than the output image. To understand the reason for this, consider the situation where the sensor is operating at full resolution and the scaler is enabled with a scaling factor of 32 (half the number of pixels in each direction). This situation is shown in Figure 24.

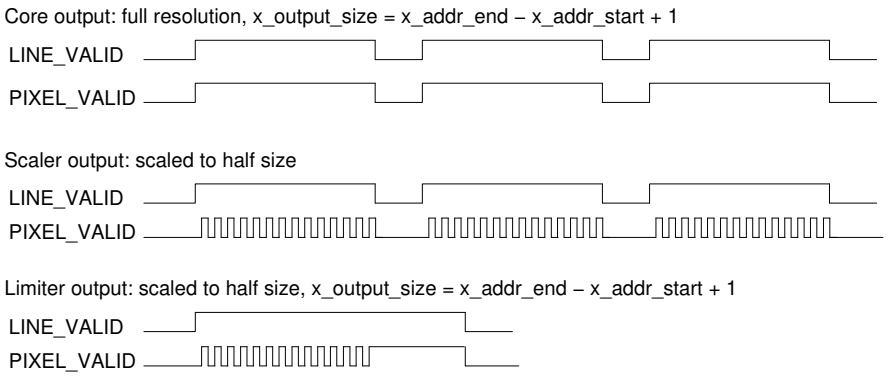


Figure 24. Effect of Limiter on the Data Path

In Figure 24, three different stages in the data path (see “Timing Specifications”) are shown. The first stage is the output of the sensor core. The core is running at full resolution and x_output_size is set to match the active array size. The LV signal is asserted once per row and remains asserted for *N* pixel times. The PIXEL_VALID signal toggles with the same timing as LV, indicating that all pixels in the row are valid.

The second stage is the output of the scaler, when the scaler is set to reduce the image size by one−half in each dimension. The effect of the scaler is to combine groups of pixels. Therefore, the row time remains the same, but only half the pixels out of the scaler are valid. This is signaled by transitions in PIXEL_VALID. Overall, PIXEL_VALID is asserted for (*N*/2) pixel times per row.

The third stage is the output of the limiter when the x_output_size is still set to match the active array size. Because the scaler has reduced the amount of valid pixel data without reducing the row time, the limiter attempts to pad the row with (*N*/2) additional pixels. If this has the effect of extending LV across the whole of the horizontal blanking time, the MT9F002 will cease to generate output frames.

A correct configuration is shown in Figure 25, in addition to showing the x_output_size reduced to match the output size of the scaler. In this configuration, the output of the limiter does not extend LV.

Figure 25 also shows the effect of the output FIFO, which forms the final stage in the data path. The output FIFO merges the intermittent pixel data back into a contiguous stream. Although not shown in this example, the output FIFO is also capable of operating with an output clock that is at a different frequency from its input clock.

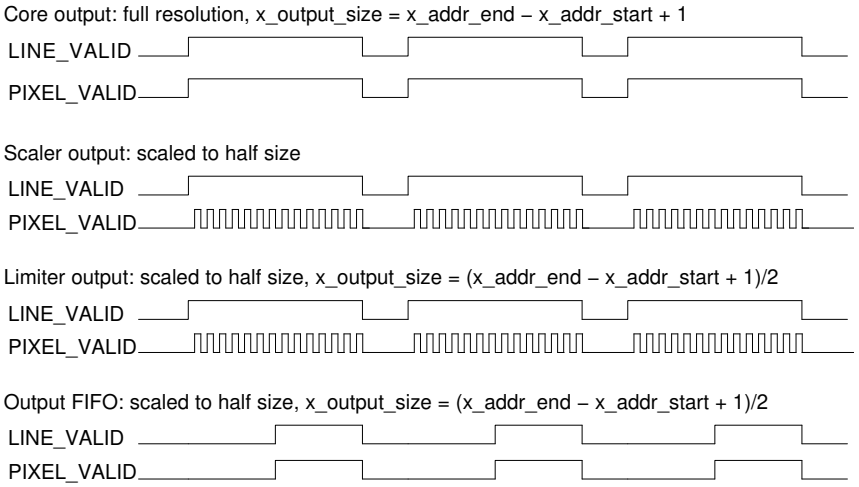


Figure 25. Timing of Data Path

Output Data Timing

The output FIFO acts as a boundary between two clock domains. Data is written to the FIFO in the VT (video timing) clock domain. Data is read out of the FIFO in the OP (output) clock domain.

When the scaler is disabled, the data rate in the VT clock domain is constant and uniform during the active period of each pixel array row readout. When the scaler is enabled, the data rate in the VT clock domain becomes intermittent, corresponding to the data reduction performed by the scaler.

A key constraint when configuring the clock for the output FIFO is that the frame rate out of the FIFO must exactly match the frame rate into the FIFO. When the scaler is disabled, this constraint can be met by imposing the rule that the row time on the serial data stream must be greater than or equal to the row time at the pixel array. The row time on the serial data stream is calculated from the x_output_size and the data_format (8, 10, or 12 bits per pixel), and must include the time taken in the serial data stream for start of frame/row, end of row/frame and checksum symbols.

CAUTION: If this constraint is not met, the FIFO will either underrun or overrun. FIFO underrun or overrun is a fatal error condition that is signaled through the data path_status register (R0x306A).

Changing Registers While Streaming

The following registers should only be reprogrammed while the sensor is in software standby:

- vt_pix_clk_div
- vt_sys_clk_div
- pre_pll_clk_div
- pll_multiplier
- op_pix_clk_div
- op_sys_clk_div

Programming Restrictions When Using Global Reset

Interactions between the registers that control the global reset imposes some programming restrictions on the way in which they are used; these are discussed in section “Global Reset”.

CONTROL OF THE SIGNAL INTERFACE

This section describes the operation of the signal interface in all functional modes.

Serial Register Interface

The serial register interface uses these signals:

- S_{CLK}
- SDATA
- SADDR (through the GPI pin)

S_{CLK} is an input−only signal and must always be driven to a valid logic level for correct operation; if the driving device can place this signal in High−Z, an external pull−up resistor should be connected on this signal.

SDATA is a bidirectional signal. An external pull−up resistor should be connected on this signal.

SADDR is a signal that can be optionally enabled and controlled by a GPI pin to select an alternate slave address. These slave addresses can also be programmed through R0x31FC.

This interface is described in detail in "Two−Wire Serial Register Interface".

Parallel Pixel Data Interface

The parallel pixel data interface uses these output−only signals:

- FV
- LV
- PIXCLK
- DOUT[11:0]

The parallel pixel data interface is disabled by default at power up and after reset. It can be enabled by programming R0x301A. Table 8 shows the recommended settings.

When the parallel pixel data interface is in use, the serial data output signals can be left unconnected. Set reset_register[12] to disable the serializer while in parallel output mode.

Output Enable Control

When the parallel pixel data interface is enabled, its signals can be switched asynchronously between the driven and High−Z under pin or register control, as shown in Table 7. Selection of a pin to use for the OE_N function is described in "General Purpose Inputs".

Table 7. OUTPUT ENABLE CONTROL

OE_N Pin	Drive Signals R0x301A−B[6]	Description
Disabled	0	Interface High−Z
Disabled	1	Interface driven
1	0	Interface High−Z
X	1	Interface driven
0	X	Interface driven

Configuration of the Pixel Data Interface

Fields in R0x301A are used to configure the operation of the pixel data interface. The supported combinations are shown in Table 8.

Table 8. CONFIGURATION OF THE PIXEL DATA INTERFACE

Serializer Disable R0x301 A−B[12]	Parallel Enable R0x301A−B[7]	Standby End−of−Frame R0x301A−B[4]	Description
0	0	1	Power up default. Serial pixel data interface and its clocks are enabled. Transitions to soft standby are synchronized to the end of frames on the serial pixel data interface.
1	1	0	Parallel pixel data interface, sensor core data output. Serial pixel data interface and its clocks disabled to save power. Transitions to soft standby are synchronized to the end of the current row readout on the parallel pixel data interface.
1	1	1	Parallel pixel data interface, sensor core data output. Serial pixel data interface and its clocks disabled to save power. Transitions to soft standby are synchronized to the end of frames in the parallel pixel data interface.

System States

The system states of the MT9F002 are represented as a state diagram in Figure 26 and described in subsequent sections. The effect of RESET_BAR on the system state and the configuration of the PLL in the different states are shown in Table 9.

The sensor's operation is broken down into three separate states: hardware standby, software standby, and streaming. The transition between these states might take a certain amount of clock cycles as outlined in Table 9.

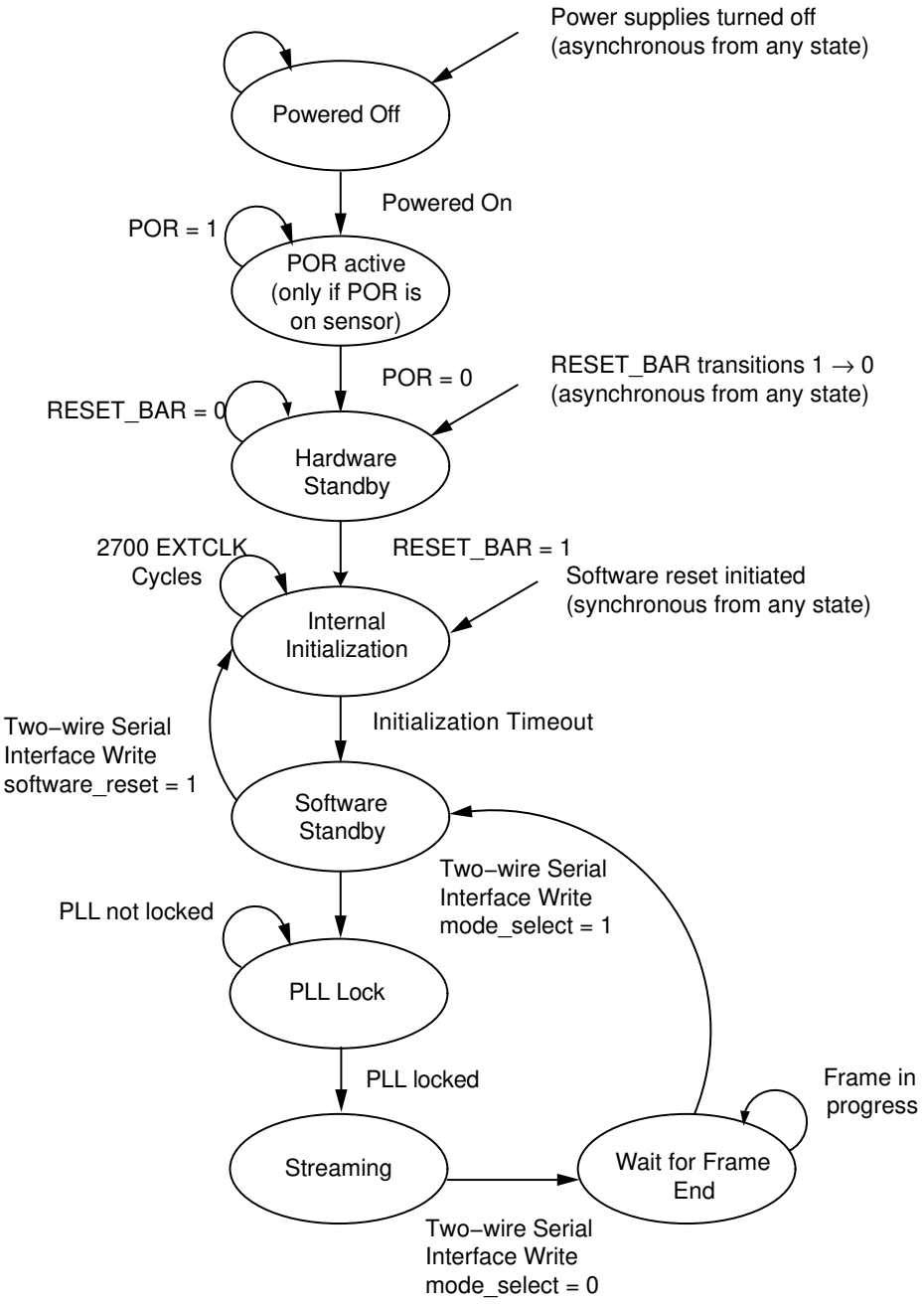


Figure 26. MT9F002 System States

Table 9. RESET_BAR AND PLL IN SYSTEM STATES

State	EXTCLKs	PLL
Powered off	x	VCO powered down
POR active	x	
Hardware standby	0	
Internal initialization	1	
Software standby		
PLL Lock		VCO powering up and locking, PLL output bypassed
Streaming		VCO running, PLL output bypassed
Wait for frame end		

NOTE: VCO = voltage−controlled oscillator.

Power−On Reset Sequence

When power is applied to the MT9F002, it enters a low−power hardware standby state. Exit from this state is controlled by the later of two events:

1. The negation of the RESET_BAR input.
2. A timeout of the internal power−on reset circuit.

It is possible to hold RESET_BAR permanently de−asserted and rely upon the internal power−on reset circuit.

When RESET_BAR is asserted it asynchronously resets the sensor, truncating any frame that is in progress.

When the sensor leaves the hardware standby state it performs an internal initialization sequence that takes 2700 EXTCLK cycles. After this, it enters a low−power software standby state. While the initialization sequence is in progress, the MT9F002 will not respond to READ transactions on its two−wire serial interface. Therefore, a method to determine when the initialization sequence has completed is to poll a sensor register; for example, R0x0000. While the initialization sequence is in progress, the sensor will not respond to its device address and READs from the sensor will result in a NACK on the two−wire serial interface bus. When the sequence has completed, READs will return the operational value for the register (0x2800 if R0x0000 is read).

When the sensor leaves software standby mode and enables the VCO, an internal delay will keep the PLL disconnected for up to 1 ms so that the PLL can lock. The VCO lock time is 1 ms (minimum).

Soft Reset Sequence

The MT9F002 can be reset under software control by writing “1” to software_reset (R0x0103). A software reset asynchronously resets the sensor, truncating any frame that is in progress. The sensor starts the internal initialization sequence, while the PLL and analog blocks are turned off. At this point, the behavior is exactly the same as for the power−on reset sequence.

Signal State During Reset

Table 10 shows the state of the signal interface during hardware standby (RESET_BAR asserted) and the default state during software standby. After exit from hardware standby and before any registers within the sensor have been changed from their default power−up values.

Table 10. SIGNAL STATE DURING RESET

Pad Name	Pad Type	Hardware Standby	Software Standby
EXTCLK	Input	Enabled. Must be driven to a valid logic level.	
RESET_BAR (XSHUTDOWN)			
GPI[3:0]		Powered down. Can be left disconnected/floating.	
TEST		Enabled. Must be driven to a logic 0.	
SCLK		Enabled. Must be pulled up or driven to a valid logic level.	
S_{DATA}	I/O	Enabled as an input. Must be pulled up or driven to a valid logic level.	
LINE_VALID	Output	High−Z. Can be left disconnected or floating.	
FRAME_VALID			
D_{OUT}[11:0]			
PIXCLK			
SLVS_0P			
SLVS_0N			
SLVS_1P			
SLVS_1N			
SLVS_2P			
SLVS_2N			
SLVS_3P			
SLVS_3N			
SLVS_CP			
SLVS_CN			
FLASH		High−Z.	Logic 0.
SHUTTER			

General Purpose Inputs

The MT9F002 provides four general purpose inputs. After reset, the input pads associated with these signals are powered down by default, allowing the pads to be left disconnected/floating.

The general purpose inputs are enabled by setting reset_register[8] (R0x301A). Once enabled, all four inputs must be driven to valid logic levels by external signals. The state of the general purpose inputs can be read through gpi_status[3:0] (R0x3026).

In addition, each of the following functions can be associated with none, one, or more of the general purpose inputs so that the function can be directly controlled by a hardware input:

- Output enable (see “Output Enable Control”)
- Trigger/VD (slave mode) – see the sections below
- Standby functions
- S_{ADDR} selection (see “Serial Register Interface”)

The gpi_status register is used to associate a function with a general purpose input.

Streaming/Standby Control

The MT9F002 can be switched between its soft standby and streaming states under pin or register control, as shown in Table 11. Selection of a pin to use for the STANDBY function is described in “General Purpose Inputs”. The state diagram for transitions between soft standby and streaming states is shown in Figure 26.

Table 11. STREAMING/STANDBY

STANDBY	Streaming R0x301A–B[2]	Description
Disabled	0	Soft standby
Disabled	1	Streaming
X	0	Soft standby
0	1	Streaming
1	X	Soft standby

Trigger Control

When the global reset feature is in use, the trigger for the sequence can be initiated either under pin or register control, as shown in Table 12. Selection of a pin to use for the TRIGGER function is described in “General Purpose Inputs”. In slave mode, the GPI pin also serves as VD signal input.

Table 12. TRIGGER CONTROL

Trigger	Global Trigger R0x3160–1[0]	Description
Disabled	0	Idle
Disabled	1	Trigger
0	0	Idle
X	1	Trigger
1	X	Trigger

Clocking

The sensor contains a phase−locked loop (PLL) for timing generation and control. The PLL contains a prescaler to divide the input clock applied on EXTCLK, a VCO to multiply the prescaler output, and a set of dividers to generate the output clocks. The PLL structure is shown in Figure 27.

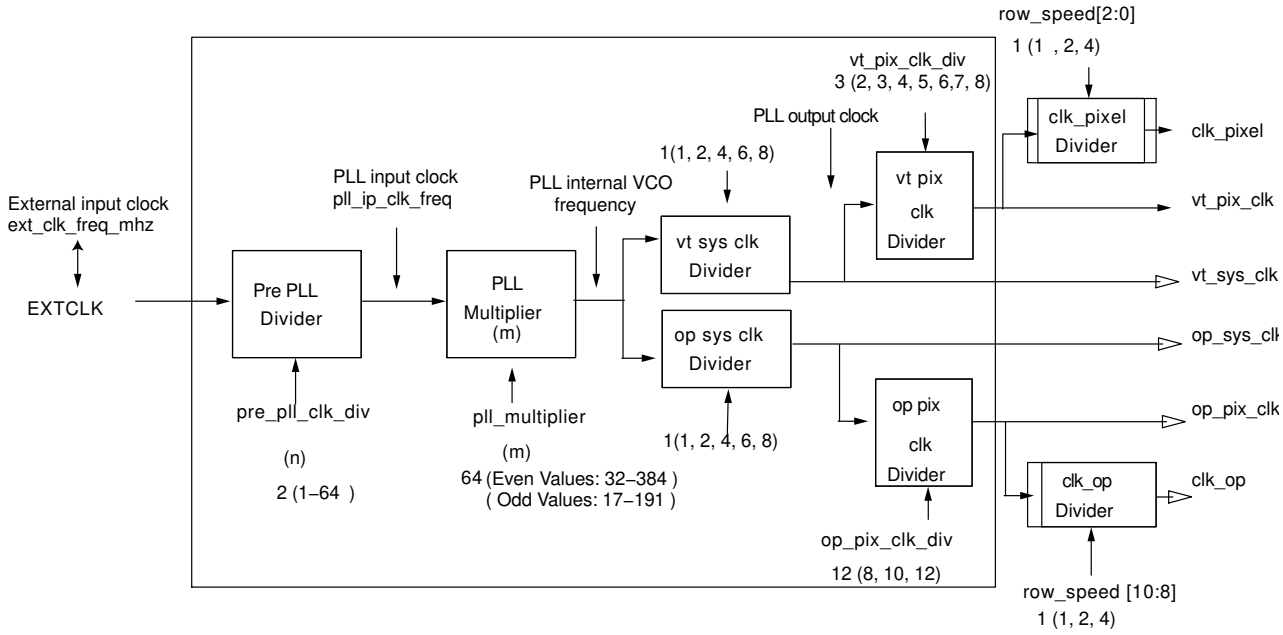


Figure 27. Clocking Configuration

Table 13. PLL PARAMETER RANGE

Parameter	Symbol	Min	Max	Units
External Input Frequency	fin	2	64	MHz
PLL Input (PFD) Frequency		2	24	MHz
VCO Clock Frequency	fvco	384	768	MHz

$$f_{PFD} = f_{in}/(n + 1), 2\ MHz \le f_{PFD} \le 24\ MHz \quad \text{(eq. 1)}$$

$$f_{VCO} = f_{in*m}/(n + 1), 384\ MHz \le f_{VCO} \le 768\ MHz \quad \text{(eq. 2)}$$

Figure 27 shows the different clocks and (in courier font) the names of the registers that contain or are used to control their values. Figure 27 also shows the default setting for each divider/multiplier control register and the range of legal values for each divider/multiplier control register. Default setup gives a physical 110 MHz internal clock for an input clock of 24 MHz. The maximum is 120 MHz.

From the diagram, the clock frequencies can be calculated as follows (eq.3):

NOTE: Virtual pixel clock is used as the basis for frame timing equations.

$$vt\_pix\_clk = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier \times (1 + shift\_vt\_pix\_clk\_div)}{pre\_pll\_clk\_div \times vt\_sys\_clk\_div \times vt\_pix\_clk\_div} = \frac{24\ MHz \times 165 \times 2}{6 \times 1 \times 6} = 220\ MHz \quad \text{(eq. 3)}$$

Internal pixel clock used to readout the pixel array:

$$clk\_pixel = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier \times (1 + shift\_vt\_pix\_clk\_div)}{pre\_pll\_clk\_div \times vt\_sys\_clk\_div \times vt\_pix\_clk\_div \times 2 \times row\_speed[2:0]} = \frac{24\ MHz \times 165 \times 2}{6 \times 1 \times 6 \times 2 \times 1} = 110\ MHz \quad \text{(eq. 4)}$$

External pixel clock used to output the data:

$$clk\_op = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier}{pre\_pll\_clk\_div \times op\_sys\_clk\_div \times op\_pix\_clk\_div \times row\_speed[10:8]} = \frac{24\ MHz \times 165}{6 \times 1 \times 12 \times 1} = 55\ MHz \quad \text{(eq. 5)}$$

Serial output clock:

$$op\_sys\_clk\_freq\_mhz = \frac{ext\_clk\_freq\_mhz \times pll\_multiplier}{pre\_pll\_clk\_div \times op\_sys\_clk\_div} = \frac{24\ MHz \times 165}{6 \times 1} = 660\ MHz \quad \text{(eq. 6)}$$

The parameter limit register space contains registers that declare the minimum and maximum allowable values for:

- The frequency allowable on each clock
- The divisors that are used to control each clock.

The following factors determine what are valid values, or combinations of valid values, for the divider/multiplier control registers:

- The minimum/maximum frequency limits for the associated clock must be met:
  - pll_ip_clk_freq must be in the range 2−24 MHz. Lower frequencies are preferred.
  - PLL internal VCO frequency must be in the range 384−768 MHz.
- The minimum/maximum value for the divider/multiplier must be met:
  Range for pre_pll_clk_div: 1−64.
- clk_op must never run faster than clk_pixel to ensure that the output data stream is contiguous.
- When the serial interface is used the clk_op divider cannot be used; row_speed[10:8] must equal 1.
- The value of op_sys_clk_div must match the bit−depth of the image when using serial interface. R0x0112−3 controls whether the pixel data interface will generate 12, 10, or 8 bits per pixel. When the pixel data interface is generating 8 bits per−pixel, op_pix_clk_div must be programmed with the value 8. When the pixel data interface is generating 10 bits per pixel, op_pix_clk_div must be programmed with the value 10. And when the pixel data interface is generating 12 bits per pixel, op_pix_clk_div must be programmed with the value 12. This is not required when using the parallel interface.
- Although the PLL VCO input frequency range is advertised as 2−24 MHz, superior performance (better PLL stability) is obtained by keeping the VCO input frequency as high as possible.

The usage of the output clocks is shown below:

- clk_pixel is used by the sensor core to control the timing of the pixel array. The sensor core produces two 10−bit pixels each clk_pixel period. The line length (line_length_pck) and fine integration time (fine_integration_time) are controlled in increments of half of the clk_pixel period.
- clk_op is used to load parallel pixel data from the output FIFO. The output FIFO generates one pixel each clk_op period. This clock also equals the output PIXCLK.
- Master clock frequency corresponds to vt_pix_clk/2.
- Serial clock (op_sys_clk) used for the serial output interface.

Programming the PLL Divisors

The PLL divisors must be programmed while the MT9F002 is in the software standby state. After programming the divisors, wait for the VCO lock time before enabling the PLL. The PLL is enabled by entering the streaming state.

An external timer will need to delay the entrance of the streaming mode by 1 millisecond so that the PLL can lock.

The effect of programming the PLL divisors while the MT9F002 is in the streaming state is undefined.

Clock Control

The MT9F002 uses an aggressive clock−gating methodology to reduce power consumption. The clocked logic is divided into a number of separate domains, each of which is only clocked when required.

When the MT9F002 enters a low−power state, almost all of the internal clocks are stopped. The only exception is that a small amount of logic is clocked so that the two−wire serial interface continues to respond to READ and WRITE requests.

FEATURES

Scaler

The MT9F002 supports scaling capability. Scaling is a "zoom out" operation to reduce the size of the output image while covering the same extent as the original image. That is, low resolution images can be generated with full field−of−view. Each scaled output pixel is calculated by taking a weighted average of a group input pixels which is composed of neighboring pixels. The input and output of the scaler is in Bayer format.

When compared to skipping, scaling is advantageous because it uses all pixel values to calculate the output image which helps avoid aliasing. Also, it is also more convenient than binning because the scale factor varies smoothly and the user is not limited to certain ratios of size reduction.

The MT9F002 sensor is capable of horizontal scaling and full (horizontal and vertical) scaling.

The scaling factor is programmable in 1/16 steps and is determined by.

$$\text{ScaleFactor} = \frac{\text{scale\_n}}{\text{scale\_m}} = \frac{16}{\text{scale\_m}} \quad \text{(eq. 7)}$$

scale_n is fixed at 16.

scale_m is adjustable with R0x0404

Legal values for m are 16 through 128. The user has the ability to scale from 1:1 (m = 16) to 1:8 (m = 128).

Scaler Example

When horizontal and vertical scaling is enabled for a 1:2 scale factor, an image is reduced by half in both the horizontal and vertical directions. This results in an output image that is one−fourth of the original image size. This can be achieved with the following register settings:

R0x0400 = 0x0002 // horizontal and vertical scaling mode

R0x0402 = 0x0020 // scale factor m = 32

Shading Correction

Lenses tend to produce images whose brightness is significantly attenuated near the edges. There are also other factors causing color plane nonuniformity in images captured by image sensors. The cumulative result of all these factors is known as image shading. The MT9F002 has an embedded shading correction module that can be programmed to counter the shading effects on each individual Red, GreenB, GreenR, and Blue color signal.

The Correction Function

Color−dependent solutions are calibrated using the sensor, lens system and an image of an evenly illuminated, featureless gray calibration field. From the resulting image, register values for the color correction function (coefficients) can be derived.

The correction functions can then be applied to each pixel value to equalize the response across the image as follows:

$$\text{Pcorrected(row, col)} = \text{Psensor(row, col)} * \text{f(row, col)} \quad \text{(eq. 8)}$$

where P are the pixel values and f is the color dependent correction functions for each color channel.

Each function includes a set of color−dependent coefficients defined by registers R0x3600–3726. The function's origin is the center point of the function used in the calculation of the coefficients. Using an origin near the central point of symmetry of the sensor response provides the best results. The center point of the function is determined by ORIGIN_C (R0x3782) and ORIGIN_R (R0x3784) and can be used to counter an offset in the system lens from the center of the sensor array.

SENSOR READOUT CONFIGURATION

Image Acquisition Modes

The MT9F002 supports two image acquisition modes:

1. Electronic rolling shutter (ERS) mode
   This is the normal mode of operation. When the MT9F002 is streaming; it generates frames at a fixed rate, and each frame is integrated (exposed) using the ERS. When the ERS is in use, timing and control logic within the sensor sequences through the rows of the array, resetting and then reading each row in turn. In the time interval between resetting a row and subsequently reading that row, the pixels in the row integrate incident light. The integration (exposure) time is controlled by varying the time between row reset and row readout. For each row in a frame, the time between row reset and row readout is fixed, leading to a uniform integration time across the frame. When the integration time is changed (by using the two−wire serial interface to change register settings), the timing and control logic controls the transition from old to new integration time in such a way that the stream of output frames from the MT9F002 switches cleanly from the old integration time to the new while only generating frames with uniform integration. See “Changes to Integration Time” in the MT9F002 Register Reference.
2. Global reset mode
   This mode can be used to acquire a single image at the current resolution. In this mode, the end point of the pixel integration time is controlled by an external electromechanical shutter, and the MT9F002 provides control signals to interface to that shutter. The operation of this mode is described in detail in “Global Reset”.

The benefit of using an external electromechanical shutter is that it eliminates the visual artifacts associated with ERS operation. Visual artifacts arise in ERS operation, particularly at low frame rates, because an ERS image effectively integrates each row of the pixel array at a different point in time.

Window Control

The sequencing of the pixel array is controlled by the x_addr_start, y_addr_start, x_addr_end, and y_addr_end registers. For both parallel and serial HiSPi interfaces, the output image size is controlled by the x_output_size and y_output_size registers.

Pixel Border

The default settings of the sensor provide a 4608H x3288V image. A border of up to 8 pixels (4 in binning) on each edge can be enabled by reprogramming the x_addr_start, y_addr_start, x_addr_end, y_addr_end, x_output_size, and y_output_size registers accordingly. This provides a total active pixel array of 4640H x 3320V including border pixels.

Readout Modes

Horizontal Mirror

When the horizontal_mirror bit is set in the image_orientation register, the order of pixel readout within a row is reversed, so that readout starts from x_addr_end and ends at x_addr_start. Figure 28 shows a sequence of 6 pixels being read out with horizontal_mirror = 0 and horizontal_mirror = 1. Changing horizontal_mirror causes the Bayer order of the output image to change; the new Bayer order is reflected in the value of the pixel_order register.

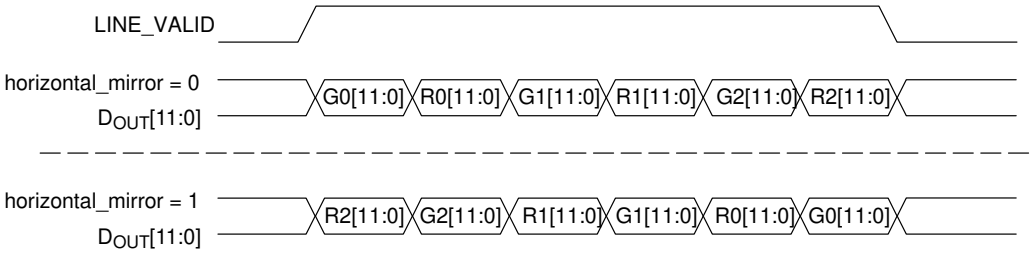


Figure 28. Effect of Horizontal Mirror on Readout Order

To enable image horizontal mirror mode, set register bit R0x3040[14]=1.

- 0 = Normal readout
- 1 = Readout is mirrored horizontally so that the column specified by x_addr_end_ is read out of the sensor first.

Vertical Flip

When the vertical_flip bit is set in the image_orientation register, the order in which pixel rows are read out is reversed, so that row readout starts from y_addr_end and ends at y_addr_start. Figure 29 shows a sequence of 6 rows being read out with vertical_flip = 0 and vertical_flip = 1. Changing vertical_flip causes the Bayer order of the output image to change; the new Bayer order is reflected in the value of the pixel_order register.