CMOS Camera Interface Specification

BACKGROUND

The CMOS camera’s used as part of the Telecommunications Project are supplied by OmniVision as an evaluation board. This board contains a 17.73MHz crystal which is used as the CMOS camera’s clock source. This clock source is internally divided by 2 to give the camera a system clock of 8.865MHz. This clock source forms the basis for the PCLK signal, this is the signal which is used to clock the camera’s data out on the Y[7..0] outputs.

The original intention for interfacing the Mitsubishi microcontroller (MCU) with the CMOS camera was to use PCLK as an interrupt source (P8.2 INT0) for the MCU. Using this original concept this would mean that the MCU would be interrupted every 112.8ns. This would mean that the MCU would spend all of its time servicing interrupts from the CMOS camera which is simply not practical.

OmniVision’s evaluation board allows for an external clock to be used to replace the onboard crystal. This solution was also investigated but it was also found to be impractical. The data from the camera Y[7..0] is the camera’s luminance data. This data must then be compared to a threshold to determine whether an LCD module’s pixel should be on or off. As each luminance byte is used to generate a single bit, 8 such luminance bytes are required to generate a single byte for transmission in the E1 network. The video channel on the E1 network is 64kbps. Therefore, the external clock required for the camera would have to have been 512kbps. This also would have been too fast for the MCU and would have only permitted the MCU to perform a maximum of 30 instructions between each interrupt.

SOLUTION

The implemented solution involves capturing a frame of data to local RAM and then making this data available to the MCU. The block diagram below illustrates the hardware that has been used to capture frame data.

Figure 1: CMOS Camera Frame Capture Hardware Block Diagram
There are three control signals that are required to capture and then retrieve the data from the CMOS camera. These are shown below in figure 2 along with the data signals.

The sequence of events that is required to set up and acquire data from the CMOS camera is as follows:

1. Use the Inter Integrated Circuit (I²C) bus to initialise and configure the CMOS Camera. Some of the CMOS camera documentation refers to the I²C bus as the Serial Camera Control Bus (SCCB). The I²C bus is a synchronous serial data transfer protocol that was initially developed by Philips for exchanging data between integrated circuits. This protocol has become a de-facto industry standard and is supported by the MCU. The I²C bus consists of 2 signals SCL (clock) and SDA (data) and it has a maximum speed of 100kbps. For more information on the I²C bus see the Mitsubishi web site, here you will find a data sheet on how to implement an I²C bus using an internal UART as well as some sample code.

2. Assert the Get Data signal. Get Data is an output signal from the MCU. This will initiate a frame capture sequence.

3. When the frame has been captured and is available the CMOS Camera interface hardware will assert the Data Available signal. Data Available is an input signal to the MCU. The time it takes for this signal to be asserted depends on the size of the CMOS camera’s viewing window. The viewing windows size can be set up using the appropriate I²C commands. As a guideline, the time it takes to sample a frame for:
   i) Window size (default - CIF) 352 x 288 (103,952 pixels) is 11.44ms,
   ii) Window size (QCIF) 176 x 144 (25,344 pixels) is 2.86ms, and
   iii) Window size (LCD size) 160 x 128 (20,480 pixels) is 2.32ms.
   NB. 1. these figures are calculated using 8.865MHz as the CMOS camera’s system clock (as supplied on the evaluation board), and
      2. these figures are the time it takes to capture a frame. The time it takes to assert the Data Available signal after the Get Data signal has been asserted may be twice this time period. This is because the Altera’s state machine waits until the next frame starts (using VSYNC signal) before writing the data to RAM. As such, the worst case is if the Get Data signal is asserted just after a new frame has commenced. In this case the state machine waits for the next frame to start. Therefore, it will take a maximum of twice the frame sampling time to assert the Data Available signal.

4. Use the Data Clk to clock in the data from the CMOS Camera interfaces RAM. Data Clk is an output signal from the MCU The positive going edge of Data Clk will get the next byte of data from RAM.
**POWER UP CONFIGURATION**

The datasheet for the OV6120 Black and White CMOS camera states that some pins have a dual purpose, ie. on power up a pin is sampled as an input and it then reverts to an output. An example of this is the Y7/CS0 signal. On power up this signal, CS0, is used to set the LSB of the i2c address, then it reverts to the Y7 output (MSB luminance byte).

Nowhere in the datasheet does it say for how long the pin is an input or an output. This is critical because if you have two interconnected devices acting as outputs then it is possible to damage either device. OmniVision’s technical support people provided the following explanation when asked about the input/output timing characteristics of the CMOS Camera:

> “The way they (dual function I/O pins) are meant to be used as programming pins during power up is to use a 10K resistor tied either high or low. When the pin becomes an output, it can easily drive the pin even though it has the resistor tied to it.”

Therefore, the CMOS Camera interface hardware has a series of 10k resistors and jumpers to allow some programming on power up. The most import parameter to be setup in this way is the i2c address. The i2c protocol allows for a master, in this case the MCU, to talk to many slave devices. To allow this each slave i2c device must have a unique address. Therefore, if the i2c address is not set then the MCU will be unable to modify any settings via i2c commands. The i2c address is set by the SD0, SD1 and SD2 parameters.

**EXTENSIONS**

The CMOS Camera interface hardware will be provided with the Altera EPLD programmed to perform all the functions previously mentioned in this document. The Altera logic may be modified to further alleviate some of the workload from the MCU. Using the 8-way DIP switch on the main pcb it is possible to do the data thresholding and compression in the Altera. This would involve modifying the state machine in the Altera so that instead of simply reading data from RAM it read the data, compared the data to the threshold and then placed the result in a shift register. When the shift register was full, ie. 8 bits had been generated, then the result could be transferred to the MCU. Please note, this is merely a suggested extension, the thresholding of data can be done equally as well (but slower) in the MCU.