Lab 3.0
Camera conceptual design

Problem statement

In this lab we will focus on the design of a custom master interface which is targeted towards a complex use case, where large amounts of data need to be moved from a peripheral, the TRDB-D5M camera, to memory.

High level description

The Camera driver IP module samples the pixels from the camera and store them in the SDRAM memory. It is configured by NIOS II processor. Once a frame transfer is completed the IP will set a flag in the LCD driver slave register.

The data is sent in 32 bits packets containing two RGB pixels.

The IP is split in three components:

- The Avalon Slave component, which contains the configuration registers as well as debug flags/registers.
- The Avalon Master component, which transfers the packets through a burst transfer of 4 packets to the SDRAM memory and sets the frame transfer complete flag in the LCD driver.
- The Data Acquisition component, which samples the camera data and processes it to creates the packets that are stored in a FIFO.

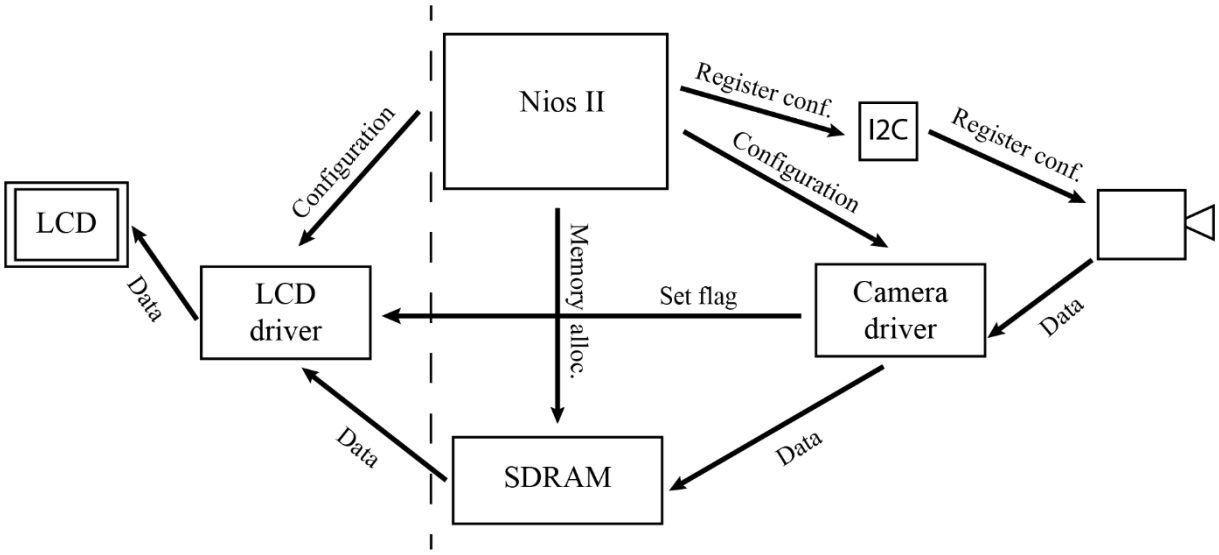

FIGURE 1 TOP LEVEL GLOBAL ARCHITECTURE VIEW

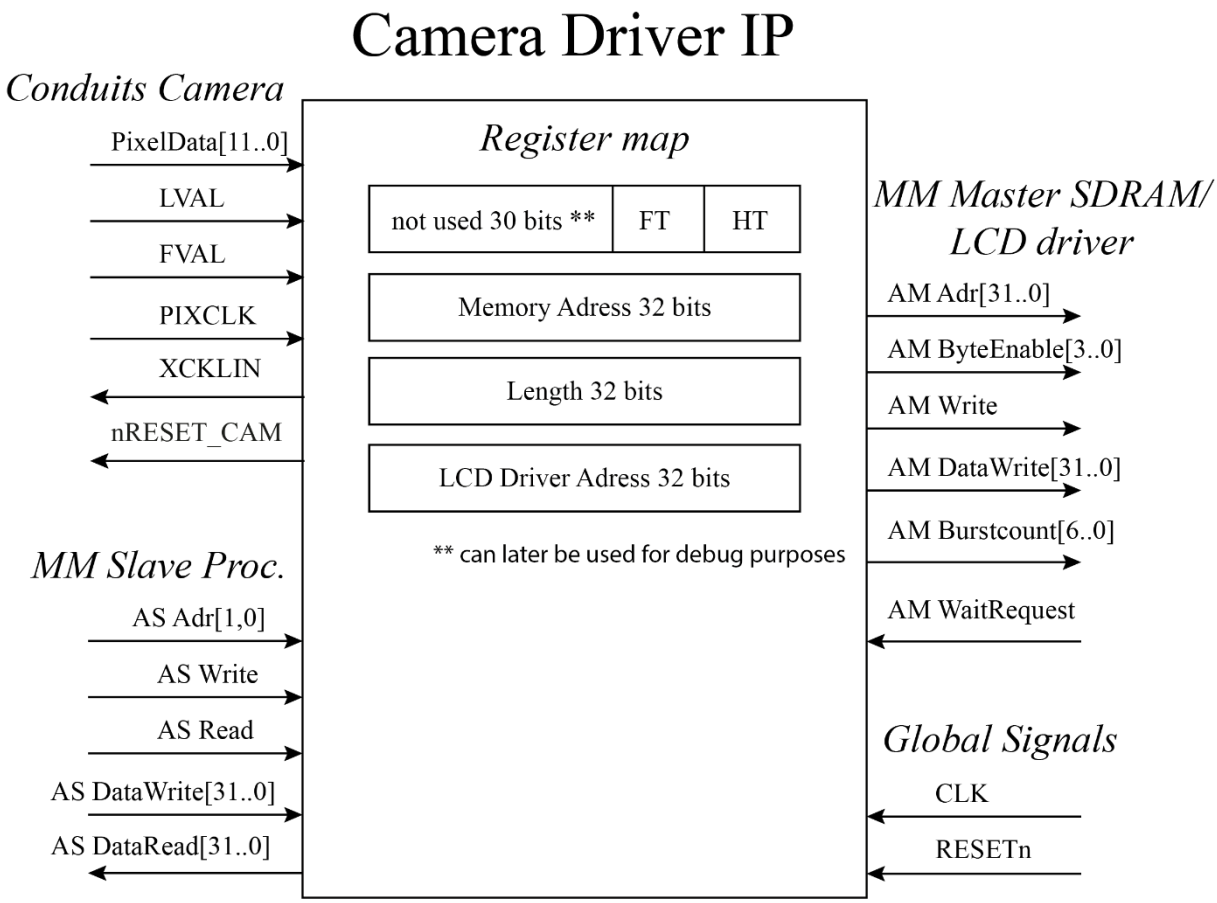

FIGURE 2 IP BLOCK DIAGRAM & REGISTER MAP

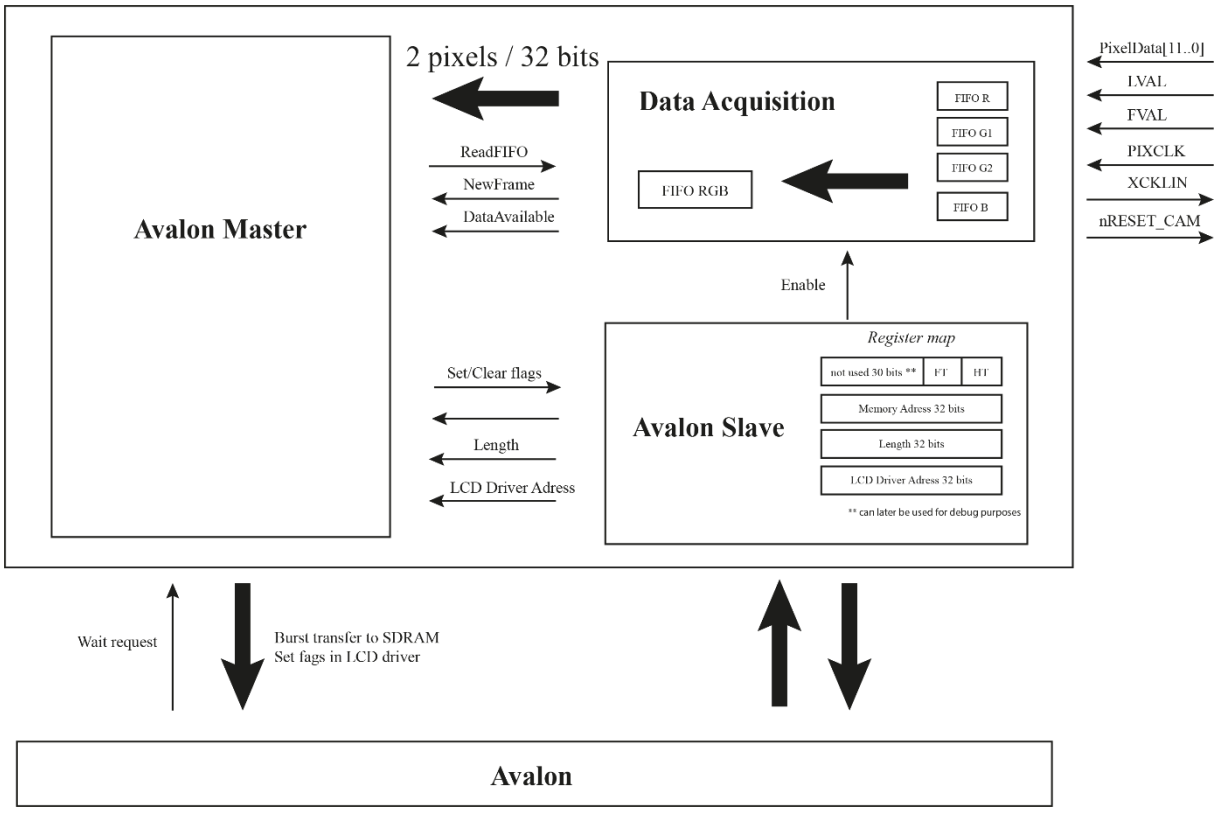

FIGURE 3 COMPONENTS DIAGRAM

Low level description

Pin diagram

The connections between the IP and the development board's pins will correspond to the following diagram.

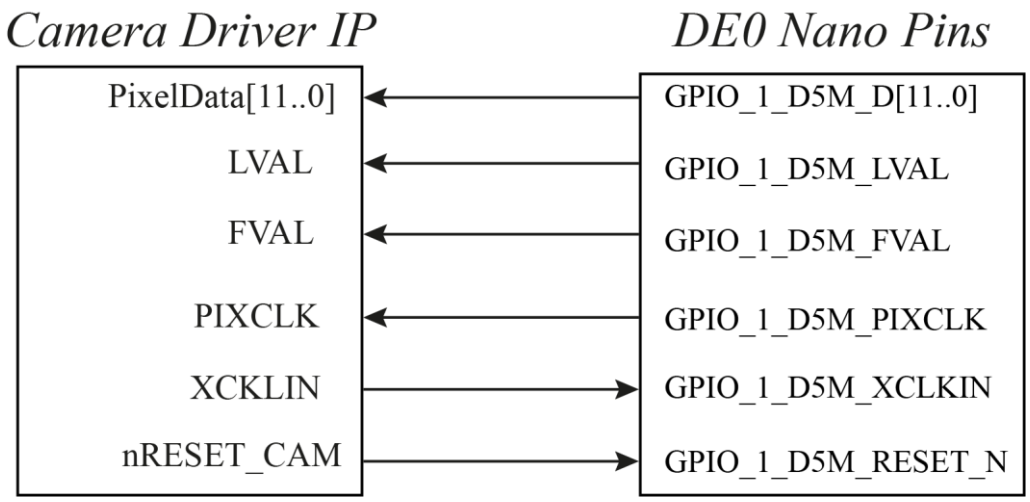

FIGURE 4 PIN DIAGRAM

Camera pixels reading and formatting

The output resolution of the camera is 640x480 **colored** Bayer pixels, meaning 640x480x4 single pixels that can be red, green 1, green 2 or blue. Since the LCD peripheral has a maximum resolution of 320x240 RGB pixels, we will skip 1 over 2 pixels of a kind on each row and skip 2 over 4 rows to reach the desired resolution (the highlighted pixels in the figure below are discarded), which means that we discard ¾ of the pixels sent by the camera. A new pixel is read on the falling edge of PIXCLK, only when both the camera signals FVAL and LVAL are active.

Concerning the color formatting, the first row output of the camera alternates between G1 and R pixels (see figure below), meaning that in order to create one RGB pixel we also need to read the next row containing the B and G2 values.

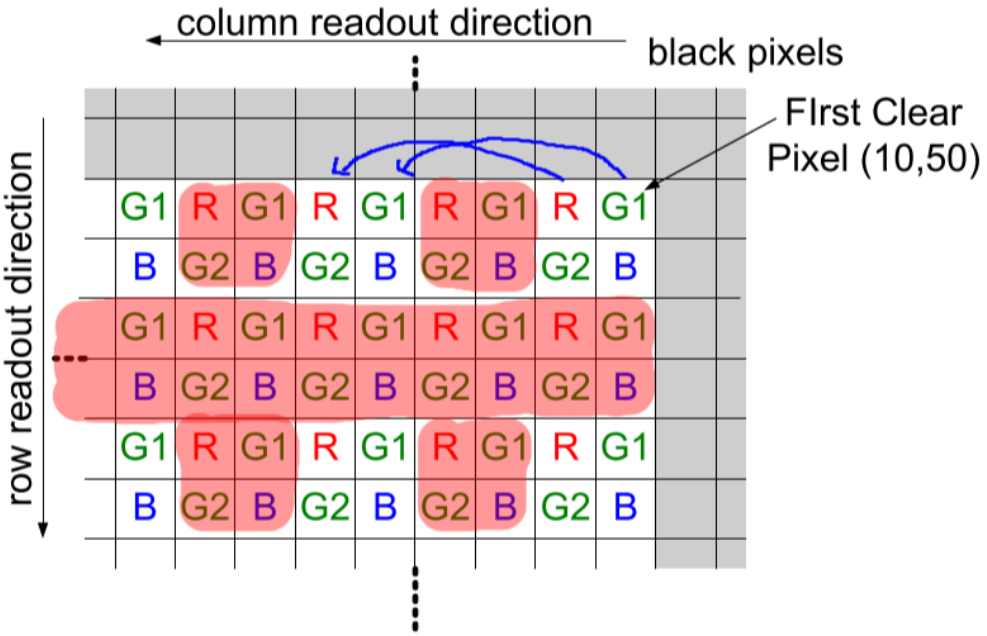

FIGURE 5 CAMERA PIXEL ORDER

That is why we will put every kind of pixel in a dedicated FIFO buffer (=4 FIFOs), so that when reading the 4 FIFOs together we will obtain the 4 matching Bayer pixels. The color resolution of the LCD is 5 bits for R and B, and 6 bits for G, so these 4 FIFOs have a word size of 5 bits, i.e. we only keep the most significant bits on lines D[11:7]. We then take the sum of the G1 and G2 pixels to have a G pixel on 6 bits. This new G pixel is concatenated along with the unmodified R and B pixels into a 16 bits RGB pixel. Then two of these 16 bits RGB pixels are themselves concatenated and put into another FIFO buffer, this one with a word size of 32 bits. This buffer is then read by the Avalon master in the incoming order, which means that the pixels are stored row by row in the DDR3 memory.

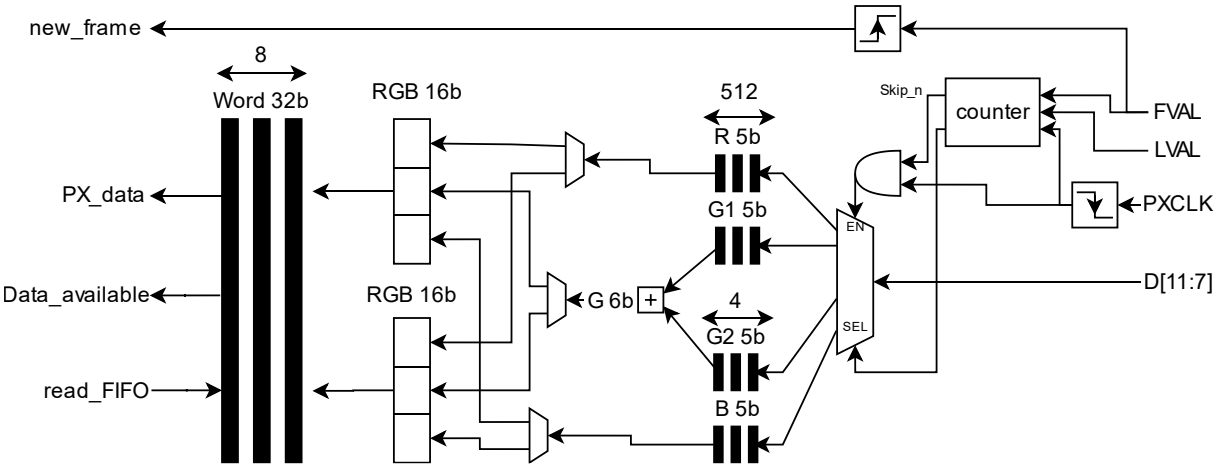

FIGURE 6 PIXEL AND FORMAT CONVERSION

The R and G1 FIFOs must have a length of at least one row, i.e. 320, so we take the next power of two (512). The length of the G2 and B FIFOs is chosen a bit arbitrarily as 4, to have some margin compared to 2 that would be enough, because each time a new pixel fills this FIFO the old pixel can be read and combined with the 3 others. The output FIFO length choice is explained in the Avalon Master component paragraph.

Memory organization

As explained in the last paragraph, the following memory organization is obtained.

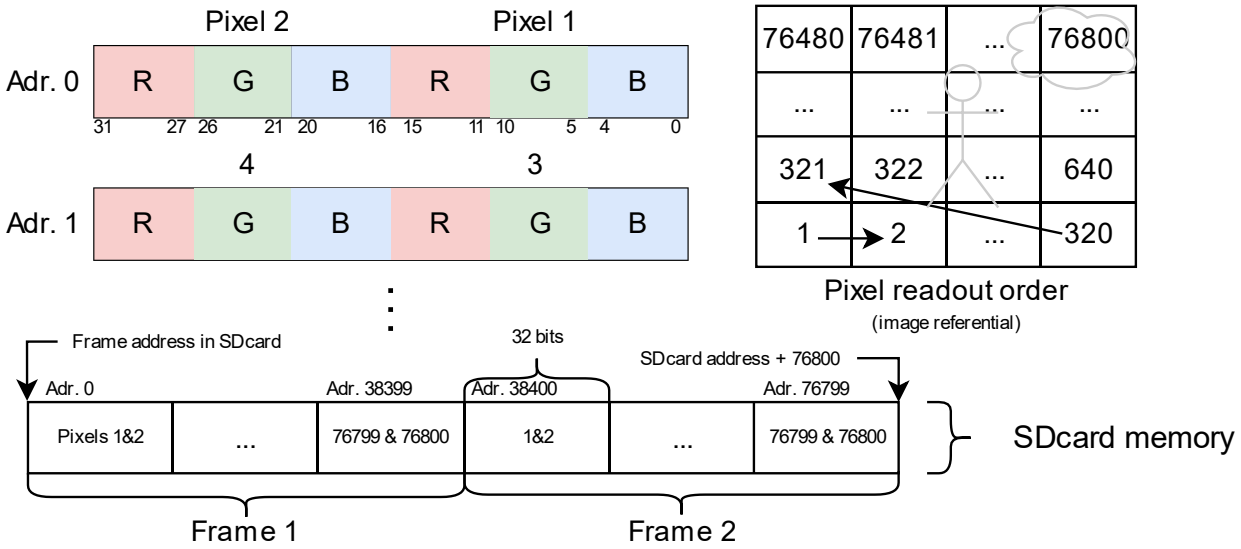

FIGURE 7 FRAME SAVING FORMAT IN MEMORY

Configuration of the camera

According to the TRDB-D5M camera datasheet, the lowest possible output resolution is 640 x 480. In order to select this output mode, we must configure the following registers, as shown in the table below. All the other registers can keep their default value, meaning (among others) that the capture mode is continuous Electronic Rolling Shutter.

Tabl 2.1 Register List and Default Values

1 = read-only，always 1； 0 = read-only，always 0； d = programmable； ? = read-only，dynamic

Register # Dec (Hex)	Register Description	Data Format (Binary)	Default Value Dec(Hex)	
R0:0(R0x000)	Chip Version	???? ???? ???? ????	6145 (0x1801)	
R1:0(R0x001)	Row Start	0000 0ddd dddd dddd	54 (0x0036)	
R2:0(R0x002)	Column Star	0000 dddd dddd dddd	16 (0x0010)	
R3:0(R0x003)	Row Size	0000 0ddd dddd dddd	1943 (0x0797)	1919
R4:0(R0x004)	Column Size	0000 dddd dddd dddd	2591 (0x0A1F)	2559
R5:0(R0x005)	Horizontal Blank	0000 dddd dddd dddd	0 (0x0000)	906
R6:0(R0x006)	Vertical Blank	0000 0ddd dddd dddd	25 (0x0019)	max(9, SW - 1919)
R7:0(R0x007)	Output Control	0d0d dddd dddd dddd	8066 (0x1F82)	
R8:0(R0x008)	Shutter Width Upper	0000 0000 0000 dddd	0 (0x0000)	
R9:0(R0x009)	Shutter Width Lower	dddd dddd dddd dddd	1943 (0x0797)	

SW

FIGURE 8 CONFIGURATION OF THE CAMERA REGISTERS

The shutter width controls how long the pixels integrate the incident light and thus the exposure, to have a higher or lower luminosity. This parameter will be tuned in software.

Timings and bus usage

In our configuration with EXTCLK=50MHz, the camera produces 40.3 frames per second (77.4 fps at 96MHz according to the datasheet). In order to send one processed frame of 320*240 = 76800 16bits pixels, we need to write 38400 32bits words in the DDR3 memory, which will be done in 9600 burst transfers, taking each 5 clock cycles if the bus is free. The transfer in memory could theoretically write 50MHz/ (9600*5) = 48 000fps, and a frame could be written in 1/48000 = 20.8 us of Avalon bus time. This means a bus usage of 20.8 *40.3fps = 840us/s for the data transfer from the camera to the DDR3 memory. The exact same bus time applies for the transfer from the memory to the LCD module since it reads the same amount of data, leading to a **usage of 0.17% of the bus** for frame transfer. This is low enough that we don't need to consider long waiting times for bus access.

Also, the minimal vertical blank of the camera is 9 rows, so 1920*9/50MHz = 346 us. Compared to the 20.8us to send one whole frame on the bus, we can be sure that all the buffers have been emptied before a new frame is sent by the camera.

Finally, during the reception of a frame from the camera, we keep only 1 pixel out of 4 received due to the dimension adaptation, then we keep 3 out of 4 dues to the Bayer to RGB conversion. Finally, the R, G and B pixels are combined into one 16bits RGB pixel, and two of them are combined into one 32bits word, which gives a ratio of 3/4/4/3/2 = 1/32. This ratio means that during a frame reception, one 32bits word is created every 32 clock cycles.

Avalon Master component

As two frames are stored in the SDRAM, the start address for a new frame will be either the first SDRAM address or the first SDRAM address plus an offset of LENGTH/2. The Avalon Master component will switch between the two alternatively. At the initialization, it starts with the first SDRAM address. N.B. The Avalon Master keeps track of the pixel count within a frame in order to send them to the right address and therefore would not require the NewFrame signal from the Data Acquisition component. But due to possible data losses, which would corrupt the pixel count, the NewFrame signal sets the start address for a new frame. The current frame number is kept in memory and the NewFrame signal combined with the current frame number sets the start address.

As the Data Acquisition starts to acquire a new frame from the camera (recall: rising edge of FVAL), the NewFrame signal is set. The Avalon Master sets the SDRAM address accordingly. At this point the FIFOs are all empty (see the Timings and bus usage paragraph).

The RGB FIFO from the data acquisition component will set the DataAvailable signal when it contains 4 packets (i.e. 8 pixels). The 4 packets will be read by the Avalon Master component and, through a burst write, sent to the SDRAM. The start address is incremented by 4 after each transfer.

The whole transfer (from rising edge of Data Available to last packet sent) lasts 6 clock cycles in the best case (no unexpected wait request occurs) = 2/3 packets per cycle (see timing diagram). The Avalon transfer lasts 5 cycles; therefore, the FIFO can be emptied at a rate of 4/5 packets per cycle (case where the Data Available signal is constantly high). The RGB FIFO is filled at a rate of 1/32 packets per cycle. The RGB FIFO won't therefore be overloaded except if the Avalon Bus is itself overloaded (but this won't be caused by the camera or LCD). If a wait request occurs, the read FIFO signal is cleared as long as wait request is set.

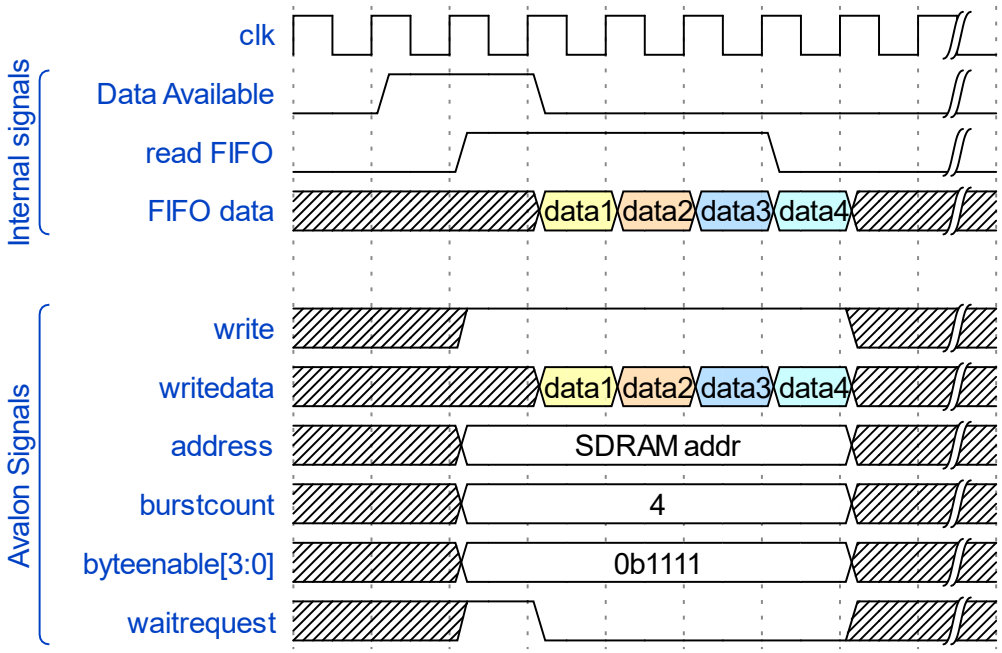

FIGURE 9 BURST TRANSFER TIMING DIAGRAM

The Avalon Master sets a flag in an LCD Driver slave register when a frame has been stored in the SDRAM. A different flag is used for the two frames. The address of the register is saved in the Avalon Slave component.

Avalon Slave component

The Avalon Slave component serves as interface between the IP and the processor. It contains 4 registers of 32 bits. First one is used for debug purposes and notifies the states of the IP. At the time of writing this report, two flags will indicate when a frame transfer is done. Other debug features could be added to this register when programming the IP. Second register contains the address of the SDRAM slave IP, third one the length of the memory allocated in the SDRAM (should correspond to two frames, so 76800) and fourth one the LCD driver slave register address where to set the frame transfer complete flags. All registers are accessible by the Avalon Master component. The Slave component sets an enable signal toward the Data Acquisition module when the length register is not zero.

Resulting finite state machine sequencing the subcomponents

The FSM support might not be (in our own opinion) the best to describe our module, but it was asked, so here it is. Apart from the Idle state, the “reading camera pixel” state is always active in parallel of the other states/actions.

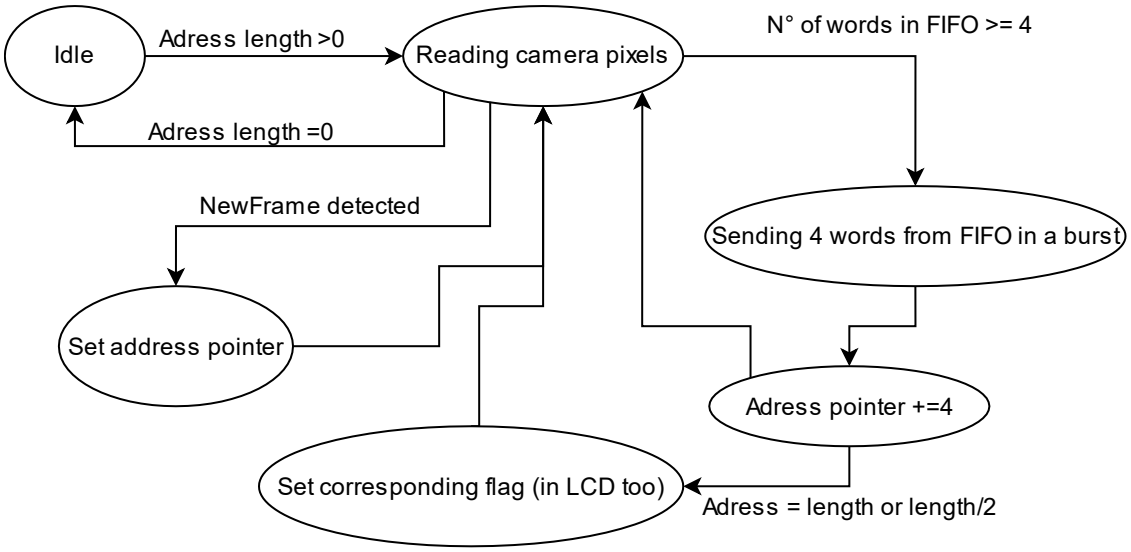

FIGURE 10 FSM BETWEEN THE SUBCOMPONENTS