Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

THIS SPEC IS OBSOLETE

Spec No: 001-14862

Spec Title: LUPA-300 SLAVE MODE OPERATION - AN5091

Sunset Owner: Evelyn Beard (EYB)

Replaced by: None

LUPA-300 Slave Mode Operation - AN5091

When working in slave mode the timing for windowing and different exposure times is not very straightforward. This application note tries to explain the different possible timing in more detail. The explanation is split up in 4 different scenarios:

No Windowing, Integration Time <= Readout Time

When no windowing is used the entire pixel array (640x480) is read out so the readout time is 4 ms (fixed). The following signals should also be fixed:

- Reset of pixel array: the pixel array is reset by making int_time1 go low for a fixed period of 160 clock cycles (2 µs at 80 MHz)
- Frame overhead time: During the FOT the image is being sampled on the internal storage nodes. This FOT should be 600 clock cycles (7.5 µs at 80 MHz) and is achieved by making all three int_time signals go low at the same time.

The exposure time will start at the rising edge after the 160 clock cycles (see *Figure 1*).

After the desired integration time one should make int_time signals (all three of them!) low for 600 clock cycles. This is what we call the FOT.

The readout will start (frame valid will go high) during the FOT (frame transfer) and last 4 ms (until the next FOT of 600 clock cycles).

The time between two FOTs is constant (4 ms). Only the time between the rising of FOT and the falling of internal reset (both controlled by int_time signals) should be adjustable. It is the time between these two events that specifies the exposure time. Frame and Line_valid signals can be used to synchronize the FPGA.

No Windowing, Integration Time > Readout Time

When no windowing is used the entire pixel array (640x480) is read out so the readout time is 4 ms (fixed). The following signals should also be fixed:

- Reset of pixel array: the pixel array is reset by making int_time1 go low for a fixed period of 160 clock cycles (2 us at 80 MHz)
- Frame overhead time: During the FOT the image is being sampled on the internal storage nodes. This FOT should be 600 clock cycles (7.5 µs at 80 MHz) and is achieved by making all three int_time signals go low at the same time.

Integration time will start at the rising edge after these 160 clock cycles (see *Figure 18* in the data sheet).

After the desired integration time one should make the int_time signals (all three of them!) low for 600 clock cycles (7.5 µs at 80 MHz). This is what we call the FOT.

The readout will start (frame valid will go high) during the FOT (frame transfer) and last 4 ms.

Right after the FOT you can make int_time_1 go low again (for 160 clock cycles) to reset the pixels again. This means a new exposure cycle will start.

Different from the scenario above, the time between two FOTs is not constant. In this case the time between two FOTs is longer than 4 ms. It is the time between the reset (immediately after the previous FOT) and the next FOT that specifies the exposure time in this scenario. In this scenario one should use only the line valid pulses (count them) to synchronize your FPGA. The rising edge of Frame_valid can be used to arm this counter but don't use the falling edge of Frame_valid.

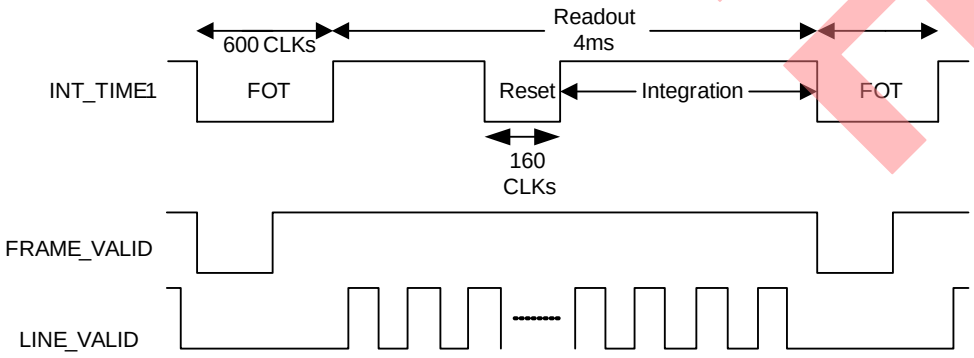

Figure 1. No Windowing, Integration Time <= Readout Time

OBSOLETE

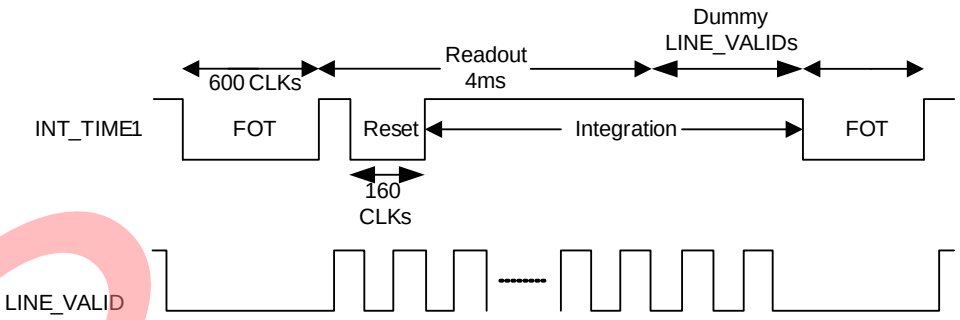

Figure 2. No Windowing, Integration Time > Readout Time

Windowing in Y, Integration Time <= Readout Time

Windowing in startX, startY and nb pix is straightforward (just upload the registers). Only for Ysize is the configuration more difficult.

In this case, the readout time is <4 ms. The following signals should be fixed:

- Reset of pixel array: the pixel array is reset by making int_time1 go low for a fixed period of 160 clock cycles (2 µs at 80 MHz)
- Frame overhead time: During the FOT the image is being sampled on the internal storage nodes. This FOT should be 600 clock cycles (7.5 µs at 80 MHz) and is achieved by making al three int_time signals go low at the same time.

In this scenario one should start by making all int_time signals go low at the same time for 600 clock cycles (7.5 µs at 80 MHz). The Y size is now determined by the moment you make them all go low again (for 600 clock cycles). This means that if you have 2 ms (4 ms/2) between 2 FOTs, your Ysize will be 240 lines (480/2).

Somewhere between these FOTs inttime_1 signal should go low for 160 clock cycles (2 µs at 80 MHz) to reset all pixels.

Integration time will start at the rising edge after these 160 clock cycles and end when all three int_time signals go low together (for 600 clock cycles).

The readout will start (frame valid will go high) during the FOT (frame transfer).

The time between two FOTs is constant for a certain window. Only the time between the rising of FOT and the falling of internal reset (both controlled by int_time signals) should be adjustable. It is the time between these two events that specifies the exposure time in this scenario. Only Line_valid signals can be used to synchronize the FPGA—use the rising edge of Frame_valid to arm your counter but don't use the falling edge. It is important to count the line_valids because too many line valids will be coming out.

Windowing in Y, Integration Time > Readout Time

Windowing in startX, startY and nb pix is straightforward (just upload the registers). Only for Ysize is the configuration more difficult.

In this case, the readout time is <4 ms. The following signals should be fixed:

- Reset of pixel array: the pixel array is reset by making int_time1 go low for a fixed period of 160 clock cycles (2 µs at 80 MHz)
- Frame overhead time: During the FOT the image is being sampled on the internal storage nodes. This FOT should be 600 clock cycles (7.5 µs at 80 MHz) and is achieved by making al three int_time signals go low at the same time.

In this scenario you should start by making all int_time signals go low at the same time for 600 clock cycles (7.5 µs at 80 MHz). Right after this FOT one can make int_time_1 go low (for 160 clock cycles) to reset the pixels. This means a new exposure cycle will start.

Integration time will start at the rising edge after these 160 clock cycles and end when all three int_time signals go low together (for 600 clock cycles).

The readout will start (frame valid will go high) during the FOT (frame transfer).

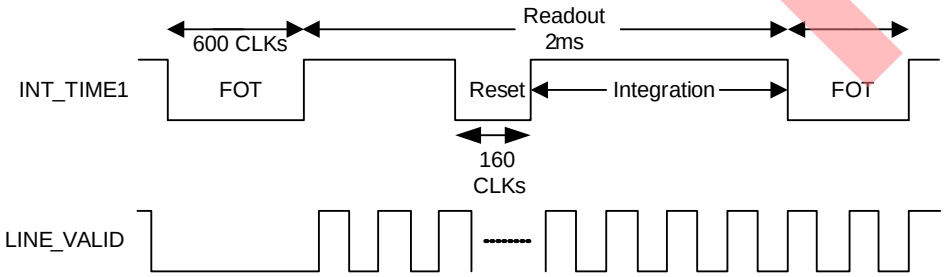

Figure 3. Windowing in Y, Integration Time <= Readout Time

OBSOLETE

The time between two FOTs is not constant. In this case the time between two is longer than the readout time. It is the time between the reset (immediately after the previous FOT) and the next FOT that specifies the exposure time in this scenario. In this scenario only the line valid pulses (count them) should be used to synchronize your FPGA and create the Ysize. You can use the rising edge of Frame_valid to arm your counter but don't use the falling edge of Frame_valid. It is important to count the line_valids because too many line valids will be coming out.

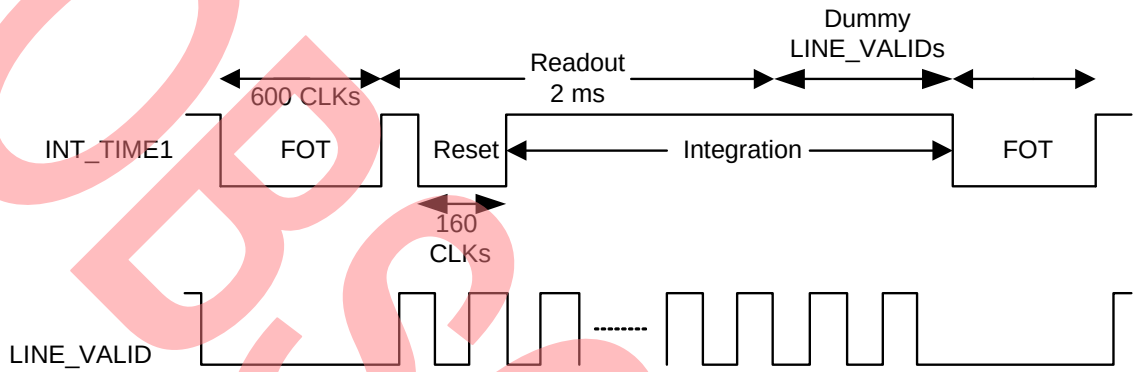

Figure 4. Windowing in Y, Integration Time > Readout Time

All product and company names mentioned in this document are the trademarks of their respective holders.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com

© Cypress Semiconductor Corporation, 2006-2013. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

This Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.

OBSOLETE