The timer for t_{conv} must be a different timer from the PWM clock timer. Values the PWM timer needs to produce are in the range: $\frac{1}{10}\mu s \rightarrow 66\mu s$.

Nvm, we don't need this timer.

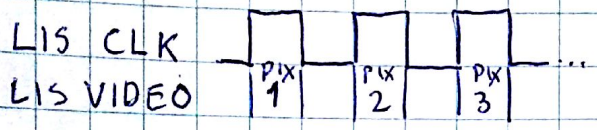
LIS clock is running

LIS RST is low. LIS RST ↓

pixels are reset, integration period (IntPd) begins.

... after N LIS clock periods, IntPd ends when LIS RST ↑

LIS SYNC (out) ↓↑ readout begins



If LIS CLK period = 20μs, (let that be 50/50 dc)

then $f_{LIS\_CLK} = \frac{1}{20\mu s} = \frac{1}{2}\frac{1}{10\mu s} = \frac{1}{2}\cdot\frac{1}{10}\frac{1}{\mu s} = 50\,kHz$

$f_{LIS\_CLK\_MIN} = 15\,kHz \rightarrow$ Maximum period = 66μs

LET the frame readout time be 10ms.

There are 770 optical pixels, 13 dark pixels and one dummy pixel.

770 + 13 + 1 = 784 pixels

10 ms ÷ 784 pixels ≈ 12.75 μs pixel clock period. (make this an integer)

Say, 15μs, so frame readout takes $15\frac{\mu s}{pix} \times 784\,pixels = 11.76\,ms$

or $20\frac{\mu s}{pix} \times 784\,pixels = 15.68\,ms$

Zooming in on the readout of a single pixel:

LIS CLK is low. LIS CLK ↓

pixel is ready for readout.

LIS CLK is clocked by ADC CLK

Pull ADC CONV low in ISR

ADC CONV is high. ADC CONV ↓ ON AN ADC CLK ↑

Can ADC CLK be 5MHz? It is physically possible, but it means there is no time to execute code in an ISR triggered by ADC CLK.

This is framed wrong. The ADC CONV is not synchronized to the clock.

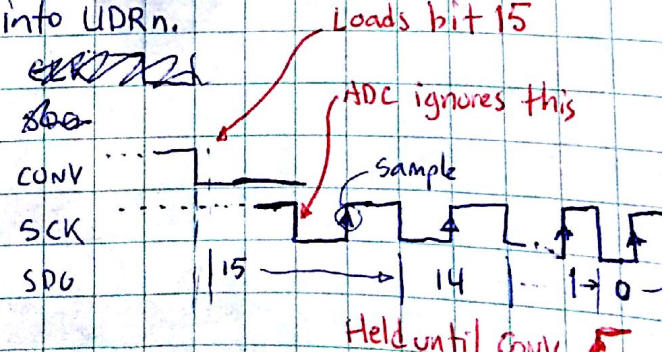
A timer is started when ADC CONV ↓ ... when the timer expires, ADC CONV ↑ AND the SPI transfer begins. ← t_{CONV} = 4.66μs

On the first reading, we do not care what is read out.

We start by ADC CONV ↓ (done in software)

Load a 16-bit value into UDRn.

ADC SCK is idle high.



Op-amp and isolation RC filter set the settling time from the falling edge of ADC CONV. Set this time to 16 ADC SCK.

$T_{SETTLE} \leq 16\,T_{SCK}$ This sets f_{SCK}.

After reading the USART buffer, pull ADC CONV high. You do not need to time t_{CONV} because it is guaranteed to finish before the next pixel is ready.