

Status Digital Design

RD50 CMOS-Meeting



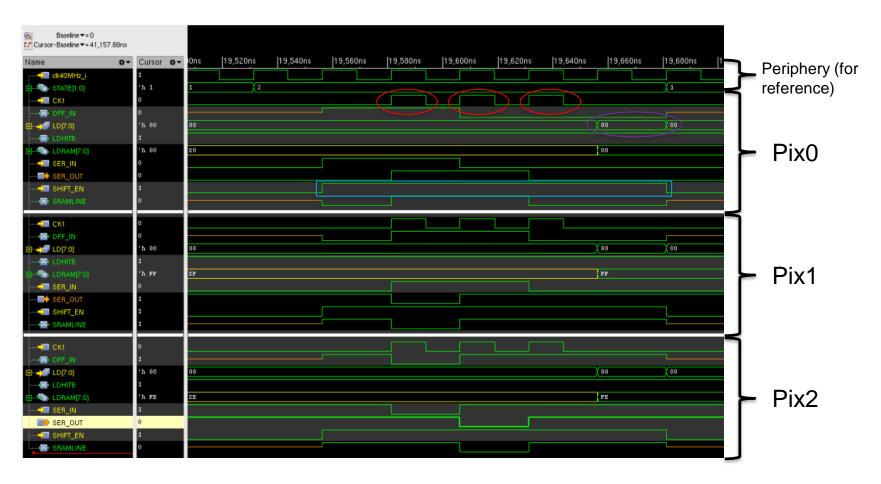


- Yellow = Inputs of the module
- Orange = Output of the module
- Green = Internal signals of the module
- Scripts available to produce these curves with 1 command
- Clock-gating cell from PDK used
 - *\$LF15A_HOME* variable must be set (done during installing the PDK)
 - PDK library needs to be compiled (script available)



Shifting-WRITE Pixel





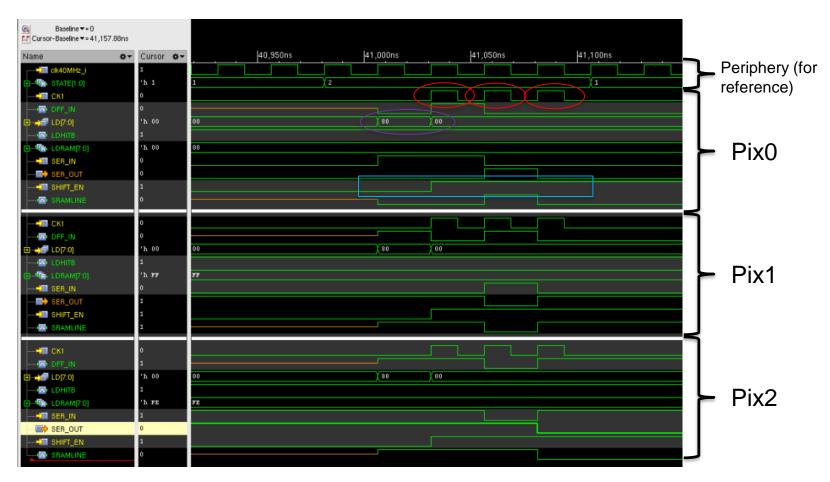
- 3 Clks to shift data in pixel
 - SHIFT_EN high
- LD Signal to write into SRAMs
 - SHIFT_EN high for writing to the SRAMs

Patrick Sieberer









- LD Signal to write into SRAMs
 - SHIFT_EN low for reading to the SRAMs
- 3 Clks to shift data in pixel
 - SHIFT_EN high

Patrick Sieberer



Shift-WRITE EOC



EOC0

Baseline ▼ = 0 L [*] Cursor-Baseline ▼ = 19,6	89ns	
Name	or Cursor o	- 800ns 900ns 1000ns 1100ns 1200ns 1300ns 1400ns 1500ns 1600ns 1700ns 1
	1	
	'h 1	1 (2)1
📫 CK1	0	
■ Te→ LD[7:0]	,F 00	00 (01)(00
- ISER_IN	1	
SER_OUT	0	
EI SHIFT_EN	0	
for the conf_bit[2:0]	'h 7	
⊕ 10 stream[2:0]	'Ъ 001	xxx xx0 x10 010
■ The shift_cyc_counter	'a 0	0 (1) 2) 3) 4) 5) 0
	0	
	0	
	0	
shift_rw	0	
⊕ ¶ serializer [5:0]	'b 000110	100100

- 6 clocks for 1 cycle:
 - 1 for recognizing STATE
 - 3 for shifting to/from the 3 pixels
 - 1 before shifting, to assert *LD* (Read only)
 - 1 after shifting, to assert *LD* (write only)



Recording Hits - Pixel



Pix0

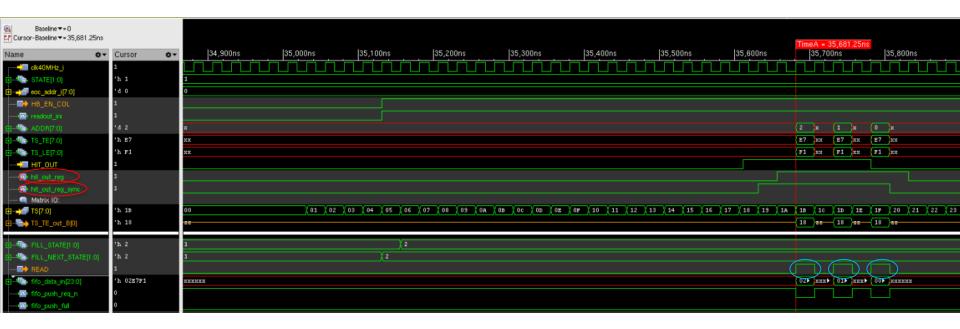
Baseline ▼= 0 Lf Cursor-Baseline ▼= 35,800ns												TimeA = 3	5.800ns	
Name 😽	Cursor 🗢	35,000ns	35,10	Ons	35,200ns	35,300ns	35,400r	is 35	,500ns	35,600ns	3	5,700ns		15,800ns
	0													
ADDR_IN[7:0]	'h 02	02												
📑 TE	1													
E \$ LE	1													
Slow	0													
- 刘 DIGILOGIC_IN	0													
DIGILOGIC_OUT	0													
	0													
🐼 HB	0													
> = HIT_IN	0													
🖬 нп_оит	0													
	0													
QSR1	0													
QSR2	0													
RDINTB	1													
	1													
READEN	0													
⊞) ∰ TS[7:0]	'h 1F	00 0	1 02 03 04	<u> Xos Xos Xo</u>	17 108 109 10.	A 0B 0C 0D	0E 0F 10	11 12 13	14 <mark> 15 16 </mark> :	17 18 19		10 10 1	-7	20 21 22
■ ■ ADDR_OUT_B[7:0]	'h FF	-22										SE FE		==
⊕- ₩+ TS_LE_B[7:0]		22									Or			+z
□ TS_TE_B[7:0]	'h 18	55					-			-	18		- 18	22
WRINTLEB	0													
WRINTTEB	0													

- Data at output bus only when **RDINTB is low**. (Only for 1 clk)
- Other values: Signals from different pixels.
- One clk pause, needed to reset the pixel after readout -> Cannot be done faster. (Limitation by the architecture)

Patrick Sieberer



Recording Hits - EOC



- 2 DFFs serve as synchronizer
 - *HIT_OUT* is <u>not</u> synchronous
- *READ* starts to toggle once a hit is registered in 2. DFF



Filling TX FIFO - EOC



🔍 🕂 Curs	Baseline ▼= 0 pr-Baseline ▼= 38,631.25ns													TimeA =	38,631.2	5ns			
Name	¢+	Cursor 😽 🗸	38,100ns	38,200ns		38,300ns		38,400ns	 38	3,500ns			38,600ns			38,	700ns		 l
>	clk40MHz_i	1																	l
•	READ_STATE[1:0]	'h 2	2																l
•	READ_NEXT_STATE[1:0]	'h 2	2																
	read_cyc	0																	l
😐 🜆	fifo_data_out[23:0]	'h 0027F1	02E7F1	01E7	1								0027F1				0	00000	l
	fifo_pop_empty	0																	ľ
	fifo_pop_req_n	1																	l
····-	eoc_rqt_data_i	0			<u> </u>														l
····-	ecc_rd_i	1																	
>	eoc_token_prev_i	0			1									l					
	eoc_flag	1																	
	ecc_token_next_o	1																	t
	select_eoc	1		(
• •	data_eoc_reg[31:0]	'h 0001E7F1	0000000	0002E7F1								0001E	7 F 1			(0	000E7F1		
	ecc_data_o[31:0]	'h 0001E7F1	0000000		00022751					(01	000000			0001E>	00000000	(0	00027F1		
	EOC FIFO				\smile														
8 🐨	mem[0:7]	[8 x 24 bits]	[8 x 24 bits]																
🗉 🖣	• mem(0)	'h 02E7F1	02E7F1																
😐 🖣	• mem(1)	'h 01E7F1	01E7F1																l
🗈 🖣	• mem[2]	'h 00E7F1	00E7F1																l
🕒 🖣	• mem[3]	'h 000000	000000																l
🕀 🖣	a. mem[4]	'h 000000	000000																
🗉 🖣	a. mem[5]	'h 000000	000000																
😐 🖣	ie mem[6]	'h 000000	000000																
e 4	• mem[7]	'h 000000	000000																l
																			-
											_	_	_	-	_	_	_		
						Debu	ia ma	ode											
							ag mi	ouc				N	orm	nal r	noc	le			

• EOCs take care of token handling

- eoc_rqt_data puts data into data_out_reg
 - Active, until there is at least 1 piece of data in 1 EOC
- data on output bus (eoc_data_out) only when token + eoc_read in previous cycle



Filling TX FIFO - CU



Baseline ▼ = 0 Encore Baseline ▼ = 38,631.25ns				laa				TimeA = 38,631,25ns
Name 🗢		38,100ns	38,200ns	38,300ns	38,400ns	38,500ns	38,600	ns 38,700ns
	'd 0	0						
🕀 🔝 IDLE[1:0]	'd 1	1						
READ_EOCS[1:0]	'd 2	2						
DEBUG(1:0)	'd 3	3						
STATE[1:0]	'd 2	1 3) 1	2		
NEXT_STATE[1:0]	'd 2	1 3			(1)(2			
dk40MHz_j	1							
·····	0							
Et eoc_rd_o	1							
📫 eoc_rqt_data_o	0							
	1							
	1							
	0							
	1							
	0							
	0							
🔍 TX FIFO								
🖻 👘 mem(0:3)	{'h 0102E7▶							
庄	'h 0102E7F1	0000000	0102E7	'F1				
💷 🚛 mem[1]	'h 0002E7F1	00000000				0002E7F1		
⊞ 40 mem(2)	'h 0101E7F1	00000000						0101E7F1
[]	'h 00000000	0000000						000127F1
			De	n ebug mode]	LNo	ormal r	node

- In debug mode, only 1 piece of data is sent to TX-FIFO
 - Keeps the data on the output bus, until next readout cycle starts
- I normal mode, 1 piece of data every clock for maximum speed



Data readout – IDLE & DEBUG

Baseline ▼= 0 M Cursor-Baseline ▼= 41,146.875r								TimeA = 41,146.875ns
	Cursor 💁	40,600ns	40,700ns	40,800ns	40,900ns	41,000ns	41,100ns	41,200ns
UNINIT[1:0]	'd. 0	0						
🖽 💷 IDLE(1:0)	'd 1	1						
SEND_DATA[1:0]	'd 2	2						
🗉 🔟 DEBUG(1:0)	'd 3	3						
E 40 STATE[1:0]	'd 3	1)(3		
INEXT_STATE[1:0]	'd 3	1				X 3		
			ה המכוע היא המספרים את הנה המוז היו המספרים את המכוע היו היו היו היא היו					דריות את האורים באיני בארג בירוא אורי היה איני באיני בארג האורים אורי היה האורים באיני באורים אורים אורים באינ
	1							
	0							
+)⊐ rst_n_i	1							
wit_counter	· d. 0	ianeonnaeonaennaeon		an a	aannannaaannaannaaannaaanaa		maanaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa	
word_counter	·d 0	2 0	^	0 1	<u>(2</u>	10 11	18	10
🗊 👘 data_to_ser(39:0)	'h ee_eeee>	_^	3c_Bc3cBc3c XFF_FFFFFFFF	EREFERENCE SC BC3CBC3		CD_0102E7F1	FF_FFFFFFF	EF_EEEEEE
	0							
debug_toggle	1							
	1							
	0							
	0							
	0							
	0							
EIIIII tx_fifo_rd_n_o	1							
⊞⇒∰ tx_fifo_rdata_i[31:0]	'h 0002E7F1	0102E7F1					000227F1	
	1							
🖳 🚽 rst_n	1							

- One word is 40 bits
 - SOF: 10 bytes with EE
 - EOF: 10 bytes with FF
 - IDLE: 3C and BC alternating (3 times).
 - Kommas from 8bit10encoding
- In DEBUG mode, only one frame is sent (= data word + SOF + EOF)



Data readout – DATA



Baseline ▼ = 0 LT Cursor-Baseline ▼ = 41,957ns														TimeA = 41,957ns
Name o-	Cursor o-	ns	41,300	ns	41,400ns		41,500ns	41,600n		41,700ns	41,8	00ns	41,900ns	42,000ns
⊕ 🖸 UNINIT[1:0]	'd 0	0												
😟 🔲 IDLE[1:0]	'd 1	1												
B-SEND_DATA[1:0]	'd 2	2												
DEBUG[1:0]	.a 3	3												
E The STATE[1:0]	'd 2	<u>3)</u>												
H NEXT_STATE[1:0]	'd 2	3 2	2											
	1													
ck40MHz_i	1													
	1													
fit_counter	'a e	0												
word_counter	'd 2	0		1	2)(0)1	2	_ <u>(</u> 3)(+	<u> </u>	0	1)(2)(0)
😐 👘 data_to_ser[39:0]	'h FF_FFFF⊧	EE_EEEEEEE		3CTBC3CBC3C		EE_EEEEEEE	CD_002E7F1	CD_0101E7F1	CD_0001E7F1	CD_0000E7F1	TE_FFFFFFFF	EE_EEEEEEE	30_80308030	FF_FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
📑 data_out_o	1	ſ	ากกกกกกกก		Մ			տույ						
debug_toggle	0										_			
	0	L												
	1													
	0													
	1										Y			
⊕ → tx_fifo_rdata_i[31:0]	'h 0002E7F1	0002E7F1						010127F1	000127F1	000027F1	000227F1			
	1													
🗘 rst_n	1													

- In normal readout mode, the length of the frame is not determined.
 - Frame ends when FIFO is empty OR
 - Maximum frame size (= one full FIFO) is reached
- If no data, IDLE frame is sent
- Flag, which control which word is sent next are checked 1 clk (of the 640MHz) earlier, to guarantee no additional clks in the transmission
- Encoding: Here, 'Encoding' means simply attaching 'CD' -> much easier for understanding







Baseline ▼= 0 F Cursor-Baseline ▼= 41,571.875ns						TimeA = 41,571.875ns				
Name 🔷 🗸	Cursor 🗢	41,300ns	41,400ns	41,500ns		41,600ns		41,700ns	41,80	lons
₽ 🔽 UNINIT[1:0]	'd 0	0								
	'd 1	1								
	'd 2	2								
	'a 3	3								
	'd 2	3 🕺 2								
NEXT_STATE[1:0]	'd 2	3 💥 2								
ck640MHz_i	1									
cik40MHz_j	v 1									
	_ 'a 0	a								
	'a 2	0 11	2 2		www.ww	2	//////////////////////////////////////	<u>annannannannannannan</u> X4	15	1 0
		EE_EEEEEEE C		EE_EEEEEEEE		·	9D_1D4E3A31			EE_EEEEEEE
	11 75_1D4EF	LE_ELEEEE								
	1			inneneenen nine e m					<u></u>	
data_out_o	1				านการ์				, <u>.</u> ,	Û
debug_toggle	1 0				nuruh				<u>,</u>	ŶŨIJŨ
	1 0 0				nurnî				<u>,</u>	
debug_toggle debug_tx_i no enable_serialzer	1 0 1 1								,	
debug_togle 	1 0 1 1								<u>, , , , , , , , , , , , , , , , , , , </u>	
debug_toggle debug_tx_j debug_tx_j debug_tx_i debug_tx_i debug_tx_i debug_tx_i debug_tx_i	1 0 1 1 1								, <u>,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,</u>	
debug_toggie debug_tx_j debug_tx_i debug_tx_i ende_send_data_frame end_data_frame tx_fifo_empty_j	1 0 1 1 1 0									1
	1 0 1 1 0 1 1							ſ		1
	1 0 1 1 0 1 1 0 1 1 1						(0001E7F1	й П.П.П.П.П.П.П.П.П.П.П.П.П.П.П.П.П.П.П.)) (0002E7¥1	
- Bi debug_togje - Bi debug_togj - Bi ende_sendeet - Bi send_data_i - Bi send_data_i - Bi to_fifo_enopt_i - Bi to_fifo_rdn_0 - Bi fordata_i[31:0] - Bi fordata_i[31:0]	1 0 1 1 1 0 1 1 1 1 1 1 1							ſ		
debug_btj debug_btj debug_btj debug_btj debug_btj debug_btj endebug_btj endebug_btj forme forme tr_ffo_empty_j debug_ffo_n fo_ffo_rdata_[B10] fo_n ret_n	1 1							ſ		
	1 1	200227F1						ĴŊŊŊŊ V X000027+1)))(000227¥1	

- Chipware IP from Cadence used
- Encoding takes 1 clk -> data is requested + encoded at end of every frame
 - Adds some unnecessary encoding
 - Still better the encoding at every clk edge



Encoding - Details





- New data requested at this edge (takes 1 edge)
- Encoded at this edge







- Write something similar to this presentation into Documentation
- Add readout mode with FREEZE signal
- Convert TS to gray counter
- Some minor changes on Code missing (MUXing SER_OUT,... -> issues listed in gitlab)
- I2C Wishbone currently checked by Jose
- Cross-check waveforms with Nissar
- Start with synthesis (~ 1-2 weeks from now)