



Double injection threshold scan

Thanks to Maurice Garcia-Sciveres, Timon Heim and Magne Lauritzen

Presented by Simon K. Huiberts

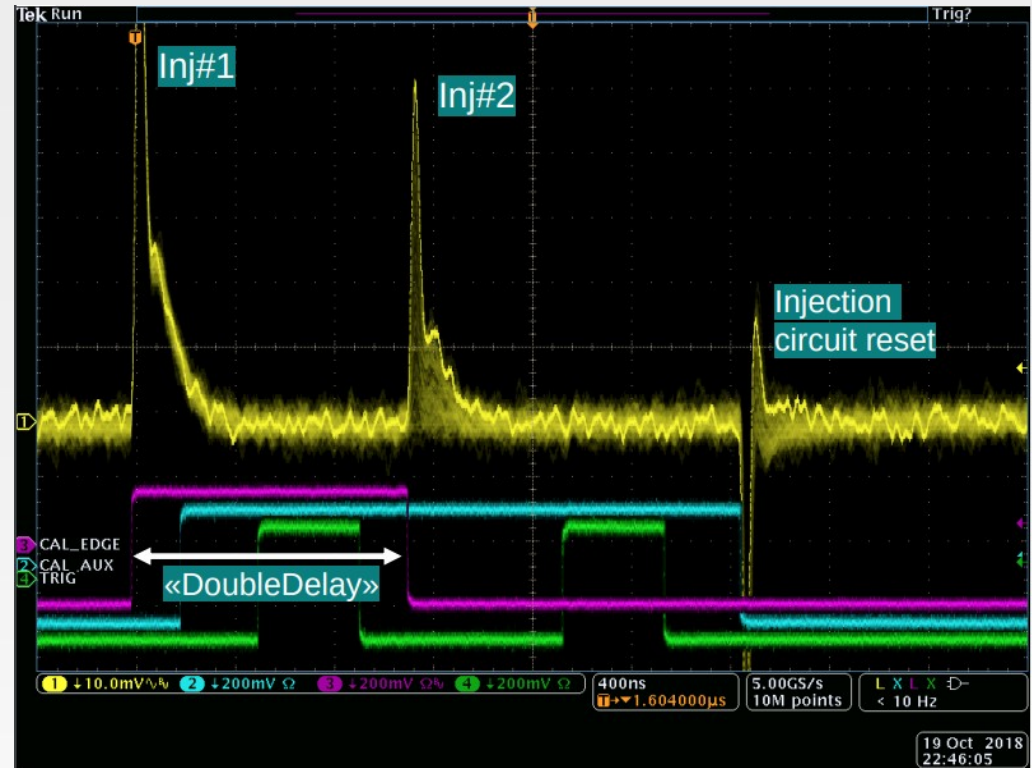


UNIVERSITY OF BERGEN



Introduction

- A **double injection scan** sends out **two consecutive charge injections** into a single pixel
- Purpose is to investigate the behaviour of the Front-end (FE) after charge injections and data readout
- **How it's done:**
 - Injecting a constant charge into the pixel (Inj#1)
 - Wait a set period (DoubleDelay)
 - Injecting a second charge of varying magnitude (Inj#2)
 - Send triggers to read out the data
 - Be able to vary charge of Inj#1, trigger windows and space between the injections

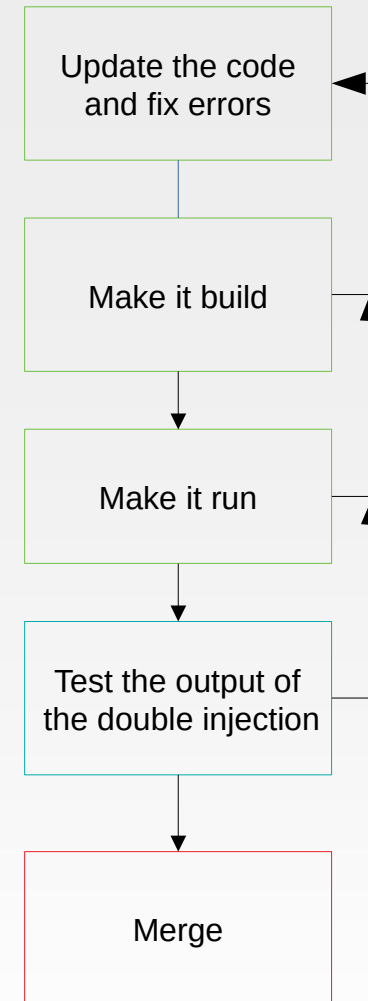


Double injection scan taken by an oscilloscope. Figure by Magne Lauritzen



Previously work

- The double injection scan had previously been implemented and tested in YARR by Magne Lauritzen locally.
- My task was to **update the code** and **test the output** of the double injection scan before **merging** it with the newest version of YARR
- Forked the latest version of YARR from the developer branch in Gitlab locally
- Gradually implemented the double injection code into YARR
- After some time was able to **build successfully**
- I was also able to make it **run without errors** after updating and fixing the code
- Needed to test the double injection methods to see if it gave the right output

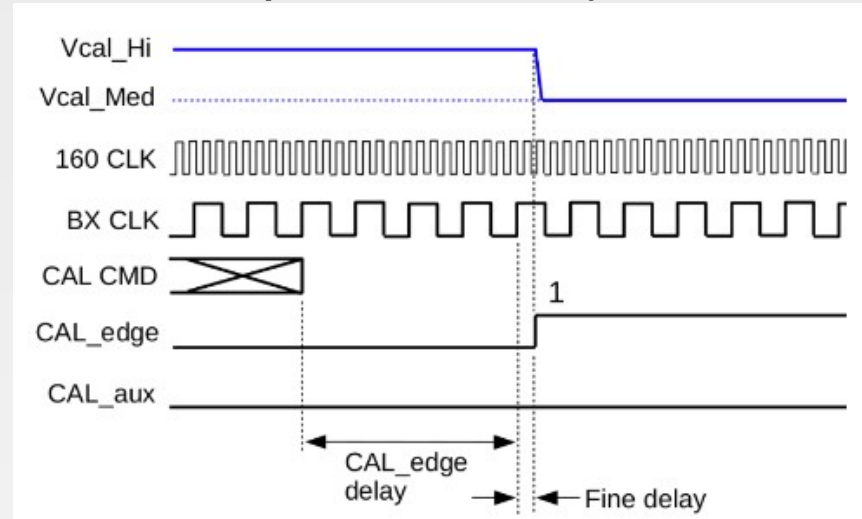




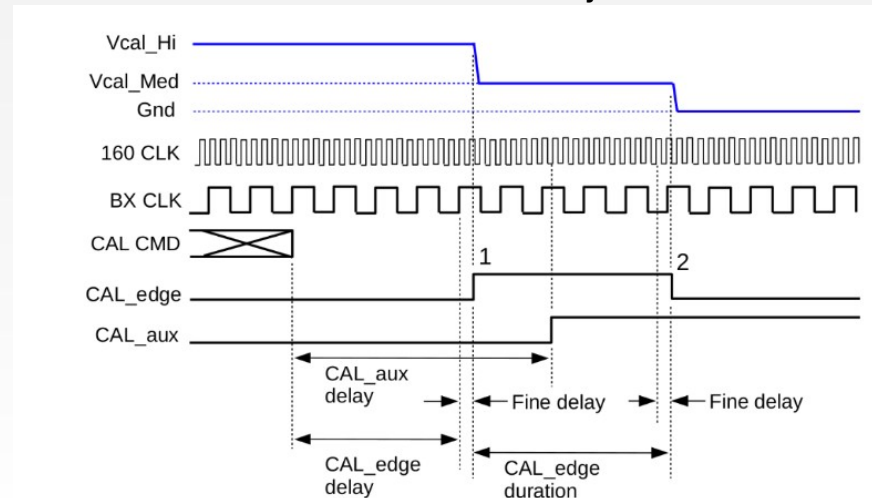
Cal command

- The **Cal command** controls the generation of two internal signals **CAL_edge** and **CAL_aux**
- Injecting charge into the pixel is done when these internal signals are changed
 - Top figure: The CAL command changes CAL_edge from **low to high**
- Inject charge from **Vcal_Hi** to **Vcal_Med**
- CAL_edge can either be set to a **single step mode** (top figure) or a **pulse mode** (bottom figure)
 - In **step mode** CAL_edge it will **stay up**
 - In **pulse mode** it will **stay up** only for a given time and **then go low** again
 - Inject **twice** with only one **CAL command**

Step mode with one injection



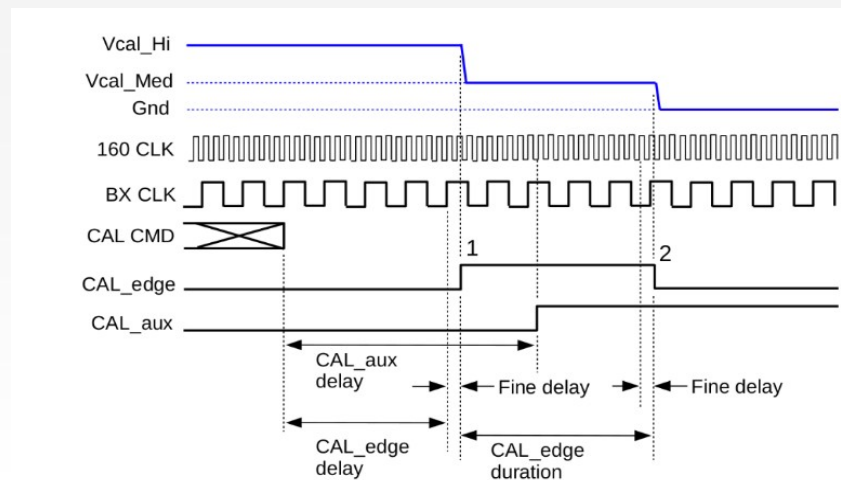
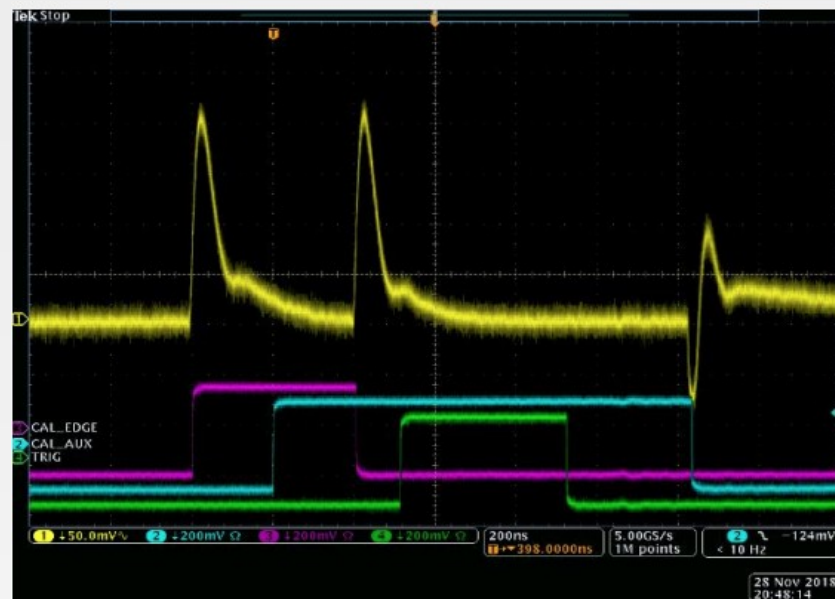
Pulse mode with two injections





Double injection with single CAL command

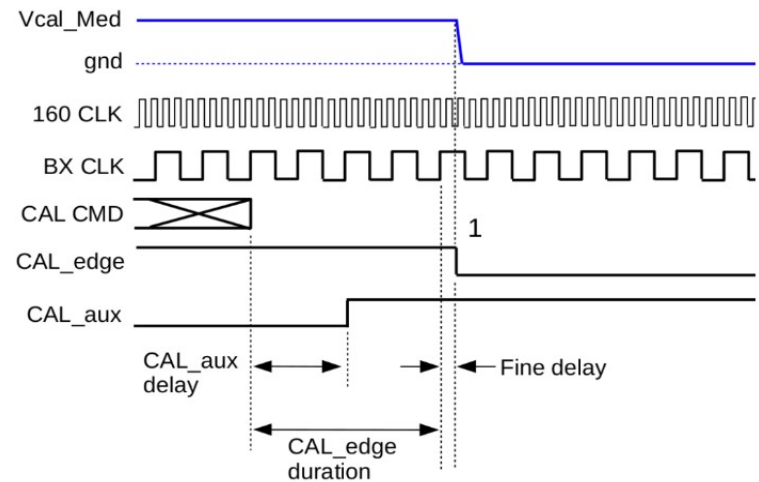
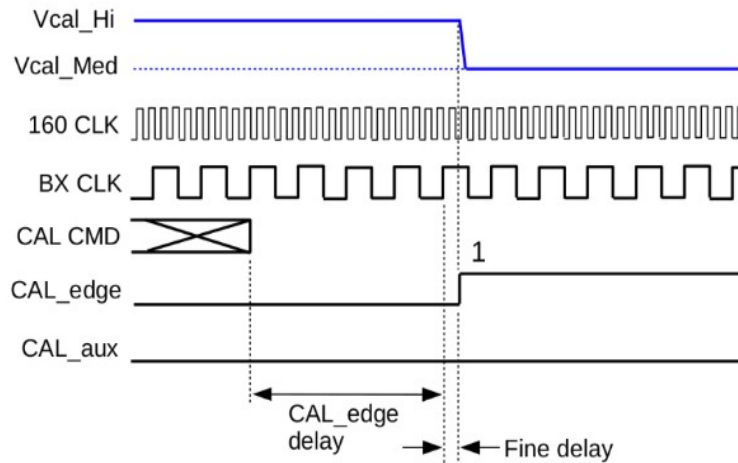
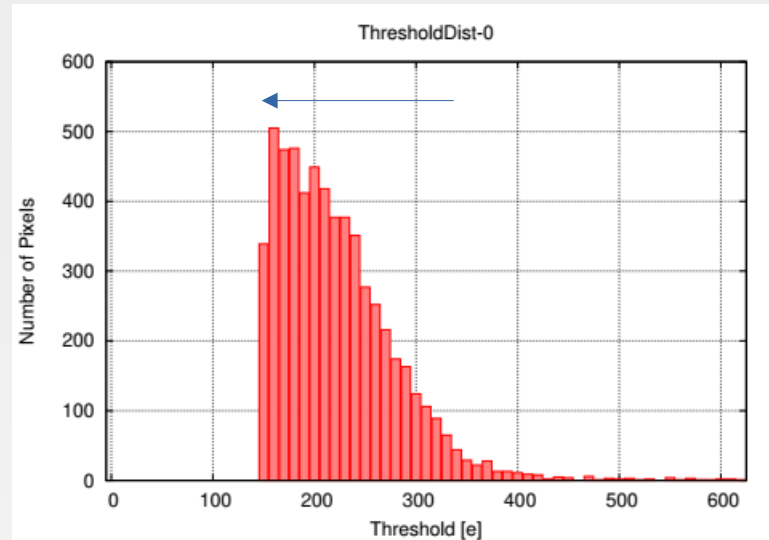
- **Doubledelay** is the **period of time** between the two charge injections
- If doubledelay value is small you want two rapid injections
- Use **pulse mode**
 - Inject from Vcal_Hi to Vcal_Med and Vcal_Med to Gnd in a **single CAL command**
- During testing, this particular method (doubledelay < 15) gave correct output
- Meaning that the double injection **worked** when using a **single CAL command**





Double injection with two CAL commands

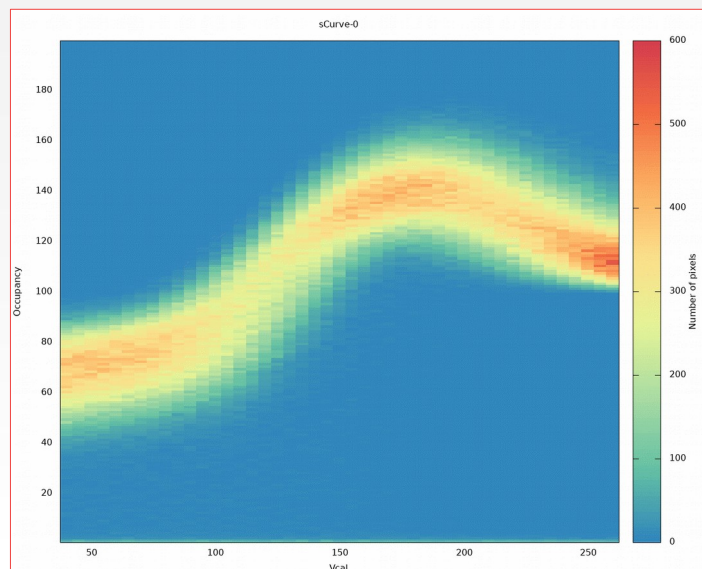
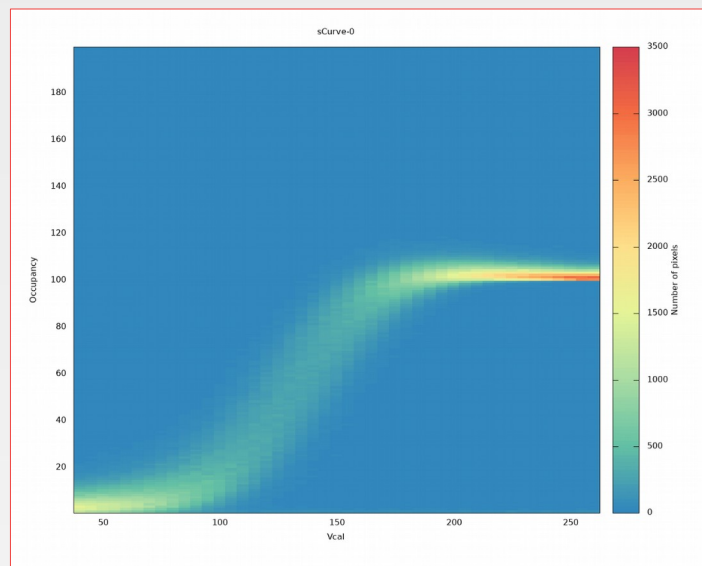
- If **doubledelay** is larger have to use **two Cal commands**
- Inject from Vcal_hi to Vcal_med first (bottom left figure) and then with a second Cal command from Vcal_med to Gnd (right bottom figure)
- During testing, this method was **not working** properly
 - Observed: Mean of the threshold distribution would move to the left when the charge of the first injection increases





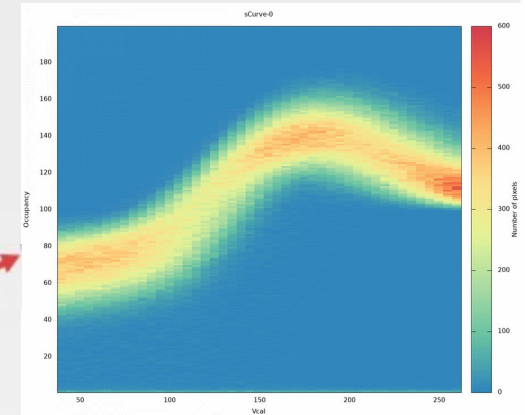
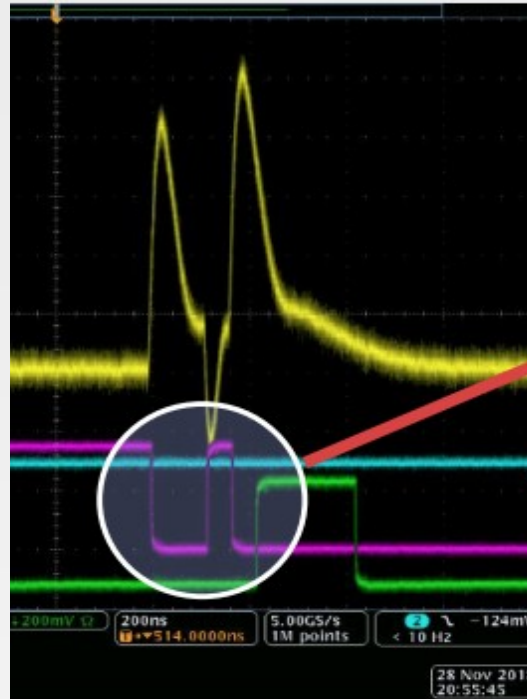
Observed S-Curves

- An **S-curve** is used to find the **charge threshold** of a pixel
- Inject charge multiple times at varying magnitudes
- Plot the number of times the pixel registered a hit (Occupancy) vs. charge injection magnitude (Vcal)
- If you inject 100 times at each charge step, S-curve should go from 0 to 100 hits
- During testing, observed strange S-Curves
 - Pointed in the direction of two charge pulses injected close to each other

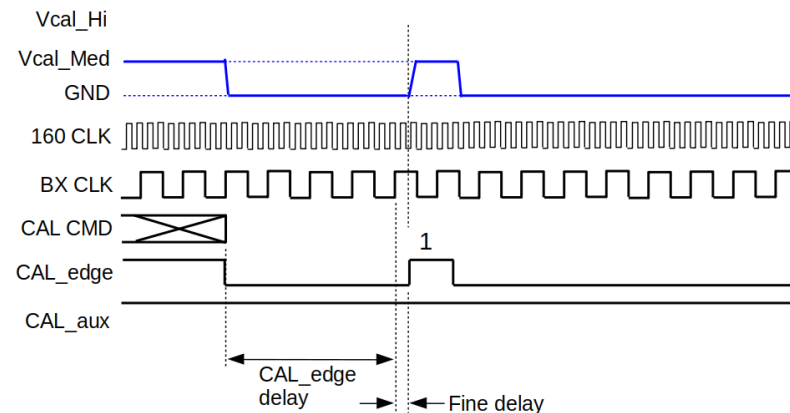


RD53A Cal command bug

- Thought that these strange S-Curves could be a convolution of two charge pulses
- Realized that this could be an effect of a reported issue of the Cal command:
 - **CAL_edge** is forced low **immediately** after the CAL command has been received
- Could result in two injections close together instead of a planned single injection
- RD53A bug explained what we observed during testing



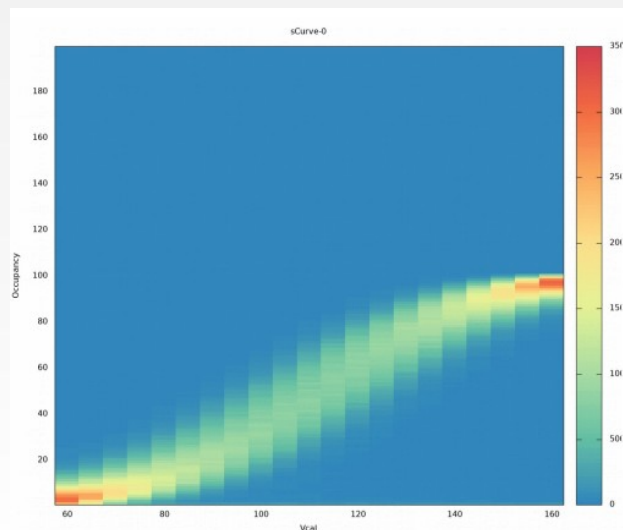
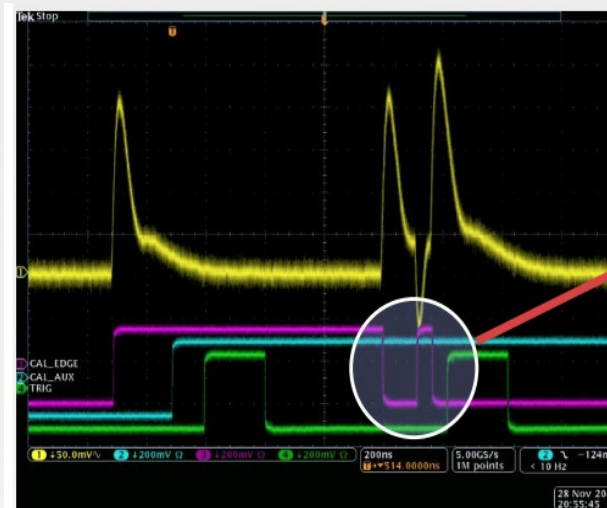
standard charge injection: Cal(m=1,del=16,dur=1,A=1,Adel=0)



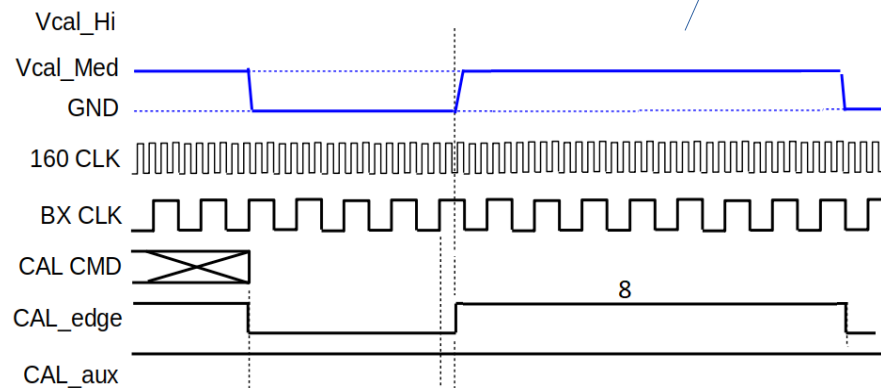


Working Cal command solution

- **Increased the width of the CAL_edge signal** in the second CAL command
- **Separate** the two charge injection pulses
- **Solution seems to work** so far for the parameters tested (triggers, doubledelay, magnitude of charge)
- Working to get this fix merged into YARR



Extend this width





Conclusion

- A double injection scan sends out two consecutive charge injections into a single pixel
- Task was to update and test a previously working double injection scan in YARR
- Able to make it build and to run
- Testing gave wrong output when using two CAL commands to create the double injections
 - Due to the CAL command bug in the RD53A chip
- Worked around this bug and was able to create a working Cal command method for the double injection
- **Ongoing work**
 - Characterize the working method in more detail
 - Test other solutions
 - Merge into YARR



Thank you for your attention!



UNIVERSITY OF BERGEN





Backup



UNIVERSITY OF BERGEN

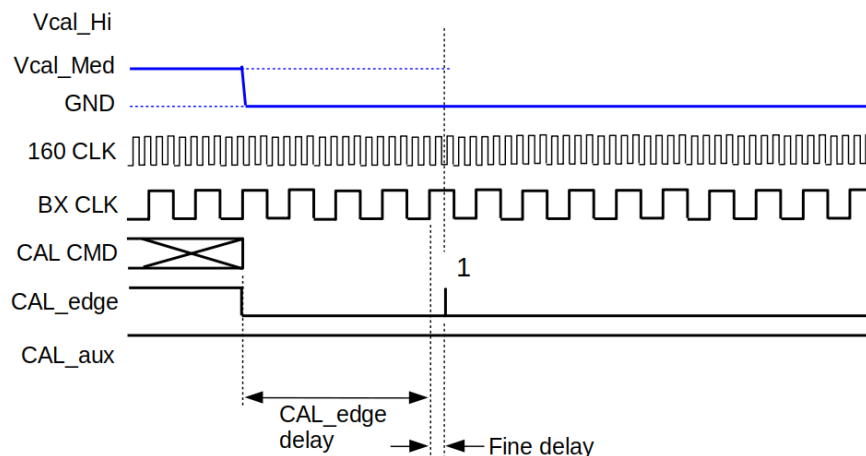




Another solution we want to test

- In the second CAL command set width of CAL_edge = 0
- Only inject one charge because of the bug forcing the CAL_edge low after the CAL CMD.
- Will be tested further when the oscilloscope is at place

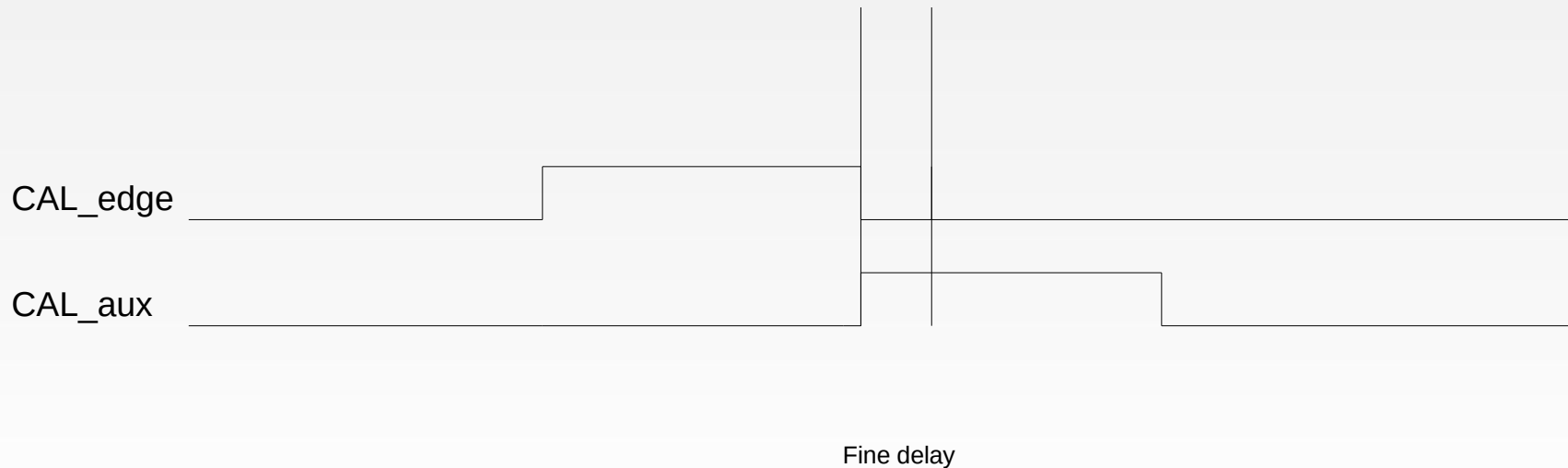
standard charge injection: Cal(m=1,del=16,dur=0,A=1,Adel=0)





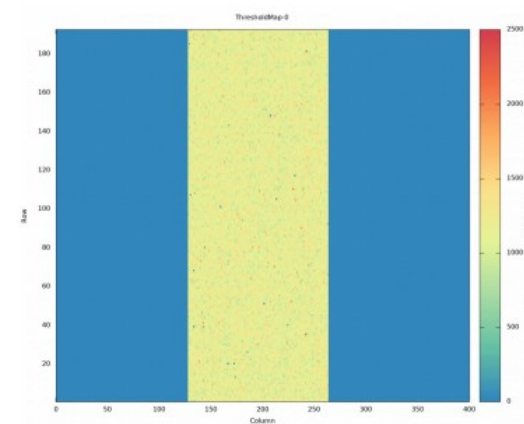
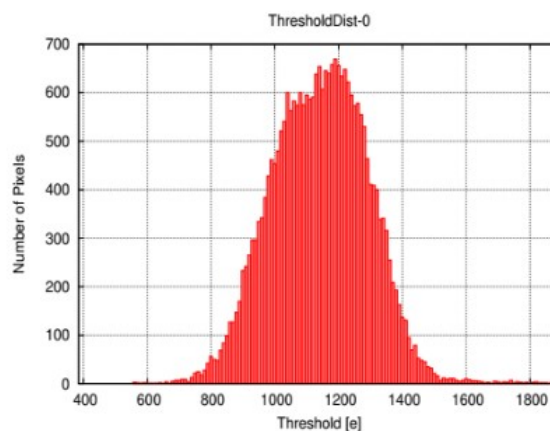
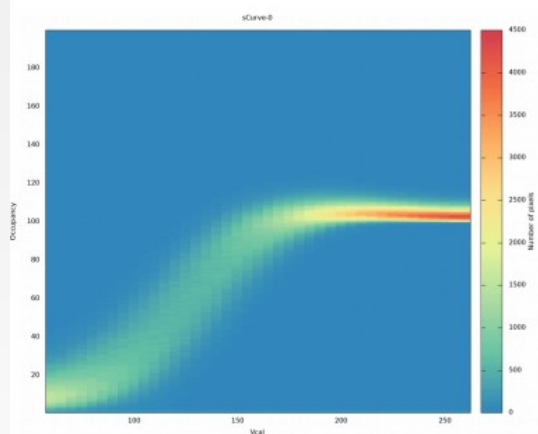
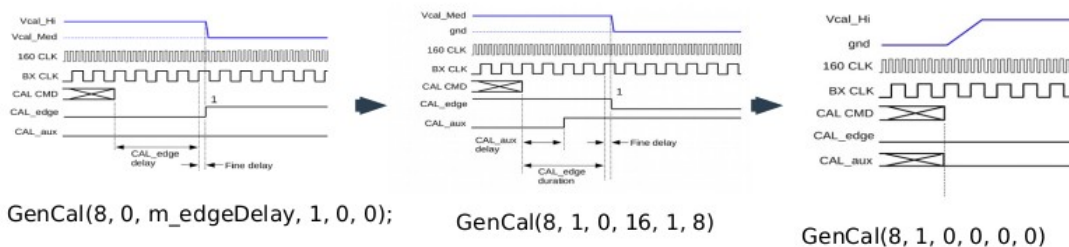
Originally in code

- First CMD: Cal(8, 0, m_edgeDelay, 1, 0, 0)
- Second CMD: Cal(8, 1, 0, 0, 1, 0)
- Reset CMD: Cal(8, 1, 0, 0, 0, 0)
- No completely sure what happens in the middle

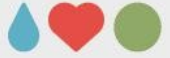




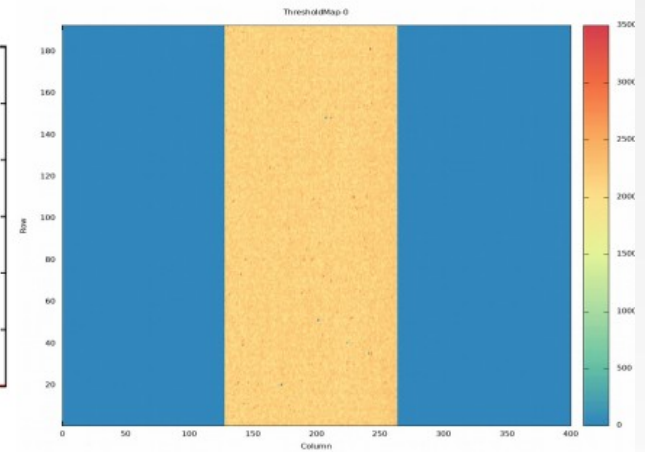
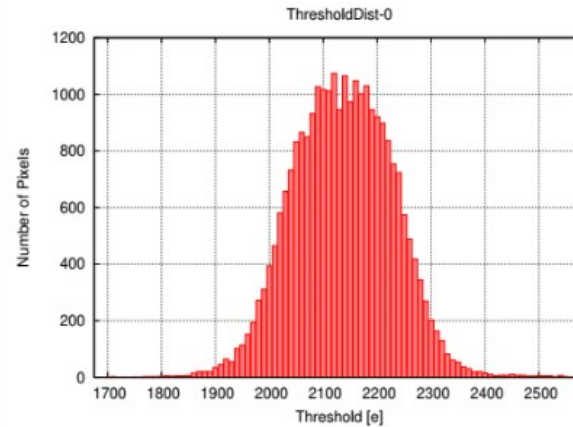
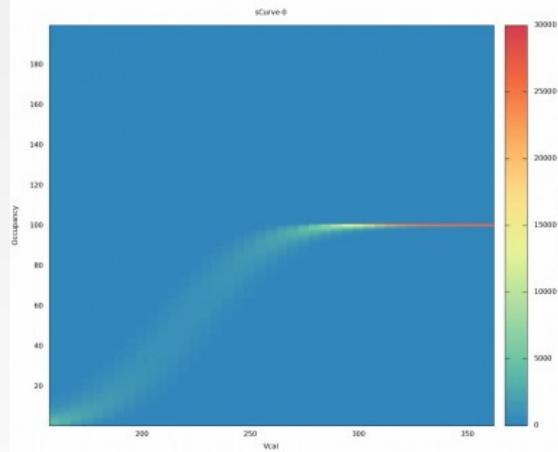
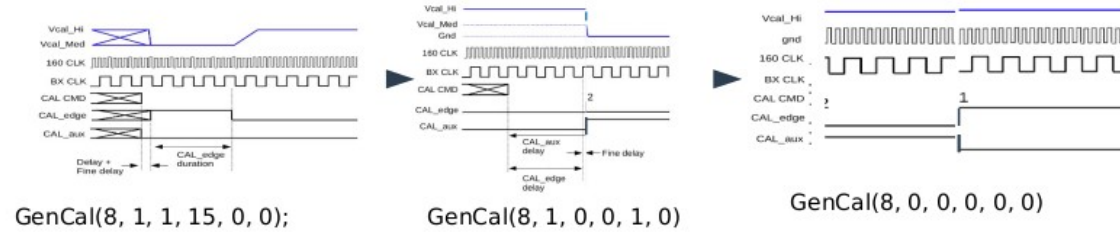
Method 1: Not working



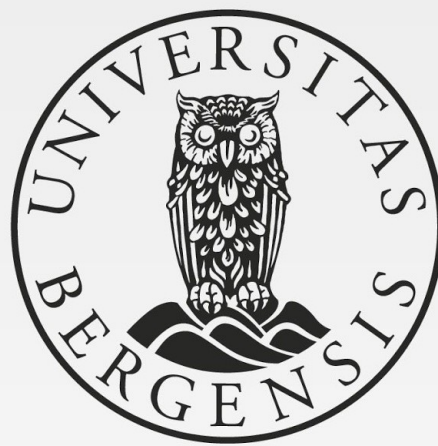
Injection 1 = 1000e, Ntrig1 = 0, Ntrig2 = 16



Method 2: Not working



Injection 1 = 1000e, Ntrig1 = 16, Ntrig2 = 16



UNIVERSITY OF BERGEN