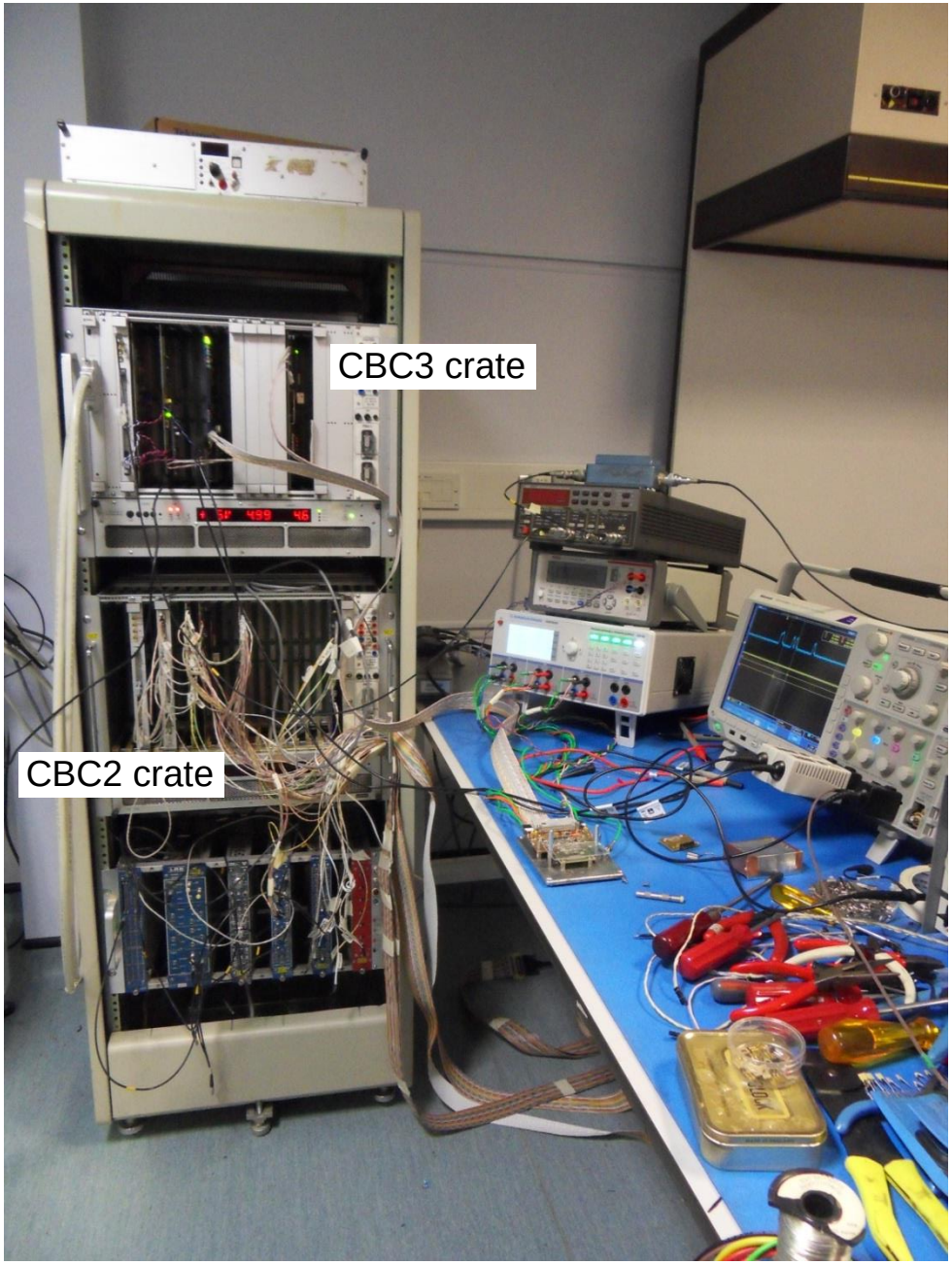
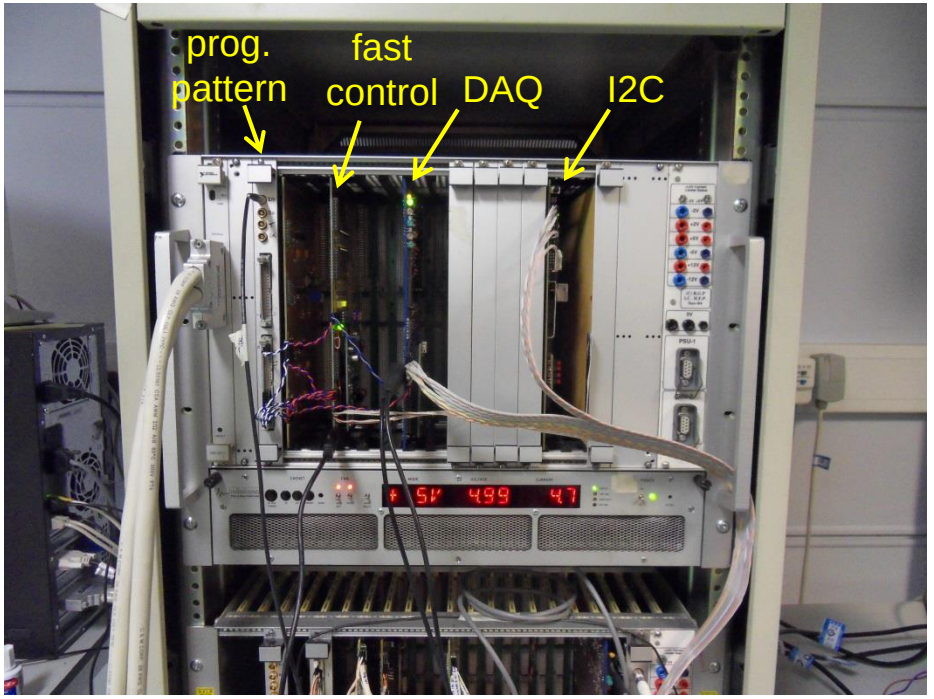
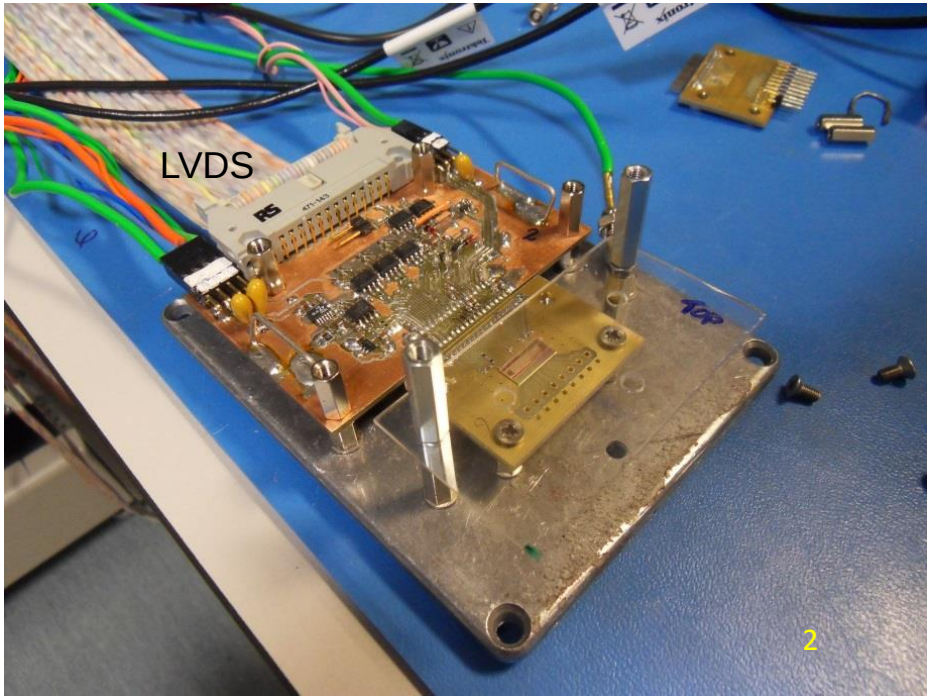
CBC3 first results

systems meeting, 16th December, 2016.

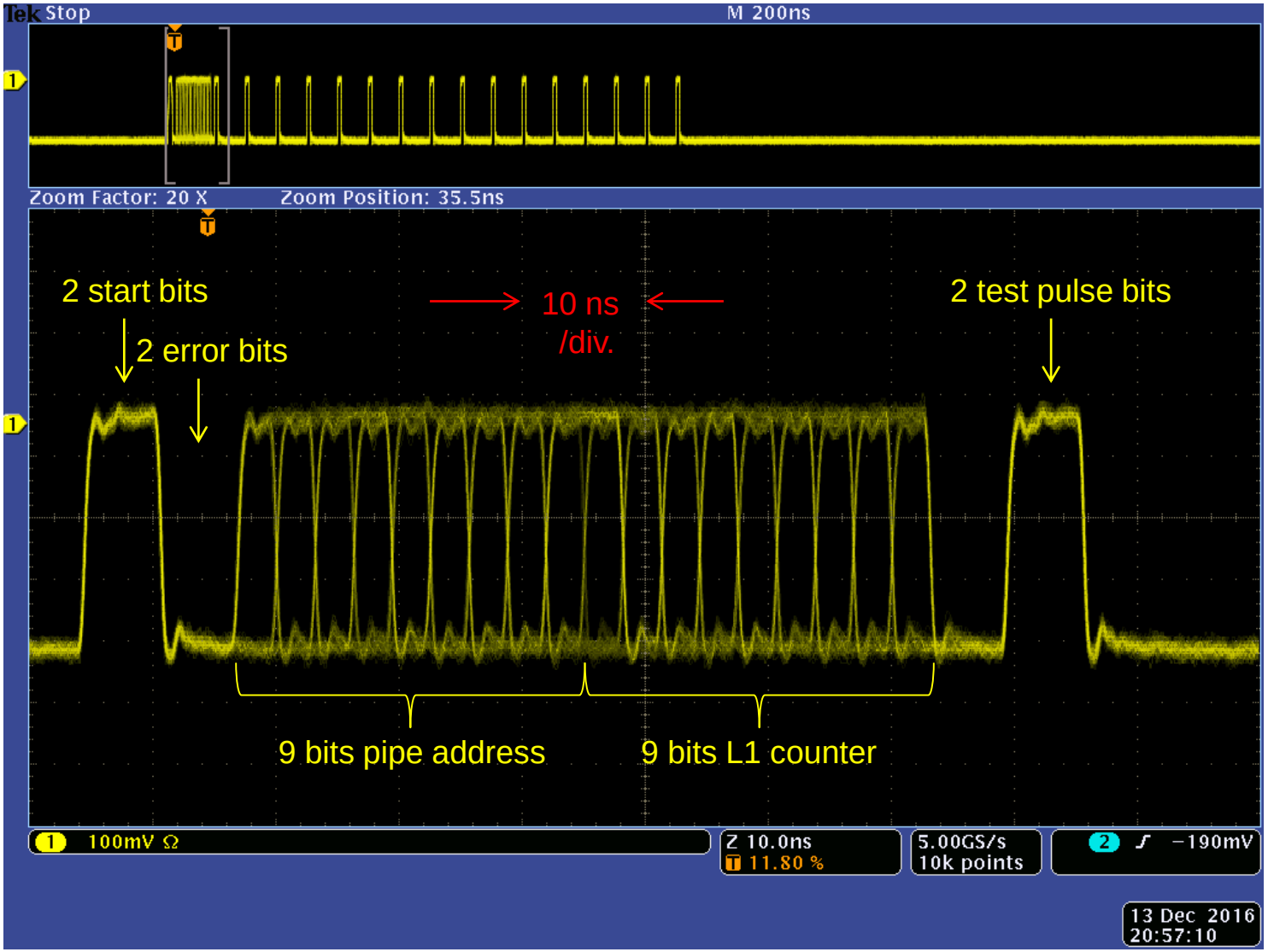
VME test setup







scope picture of L1 triggered data



differential probe close to chip output, scope on persistence

CBC3 I2C front panel

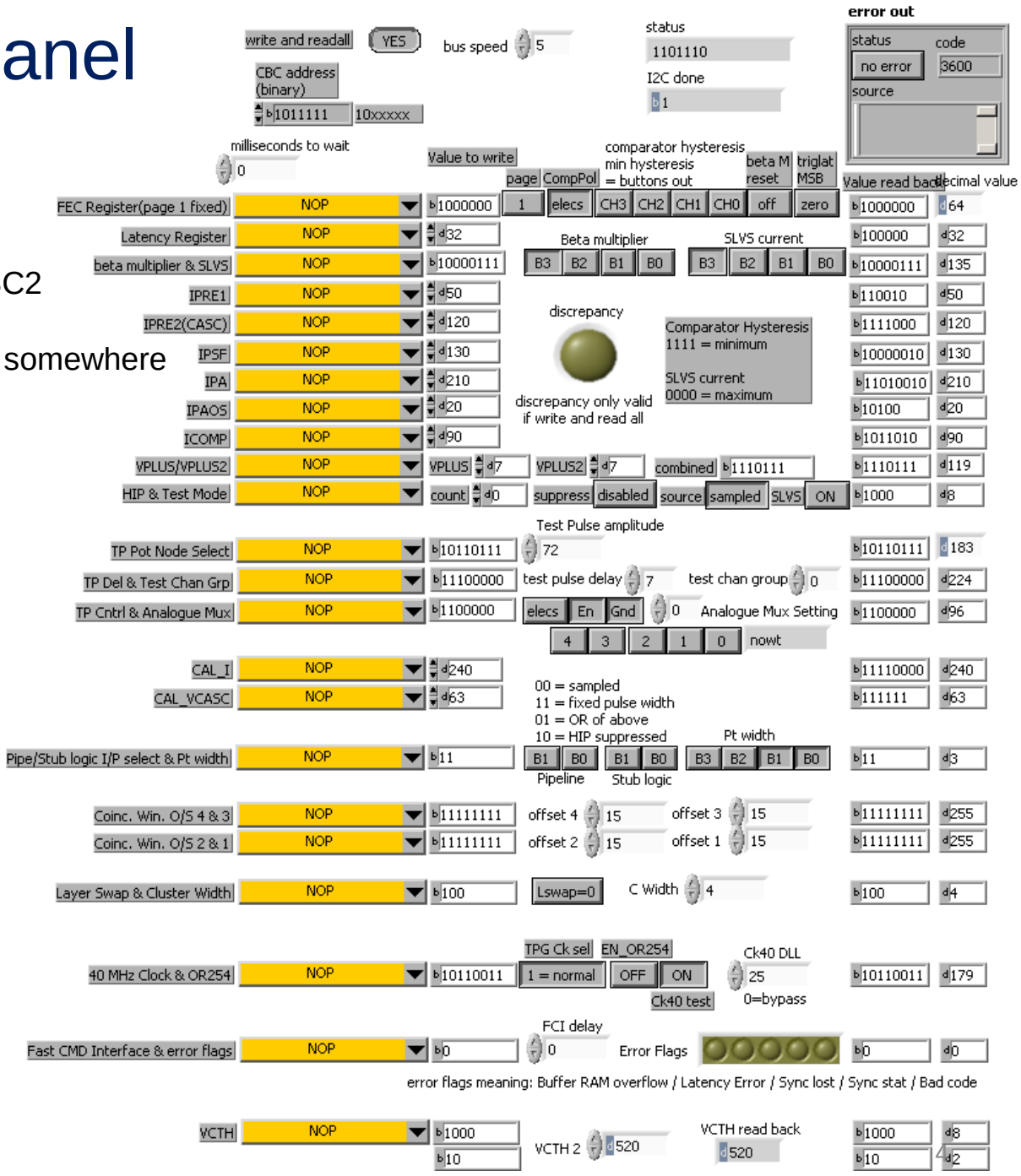
observations

“quite” complicated - more so than CBC2

relatively easy to get a bit set wrongly somewhere

GUI helps to spot incorrect setting

not just a list of addresses
& data values

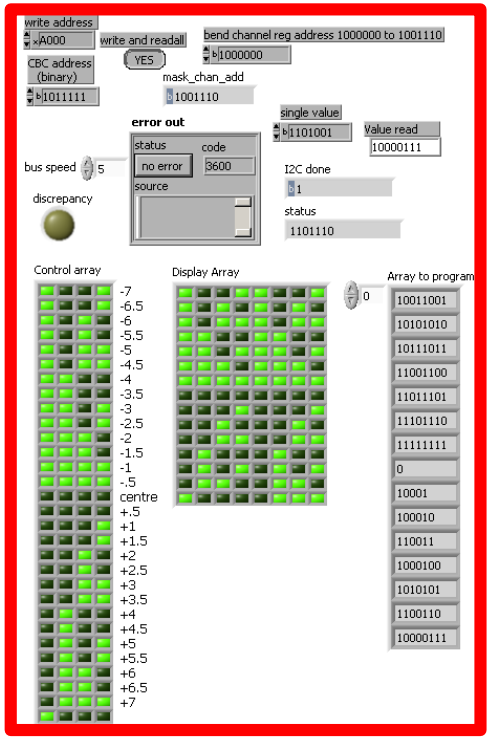


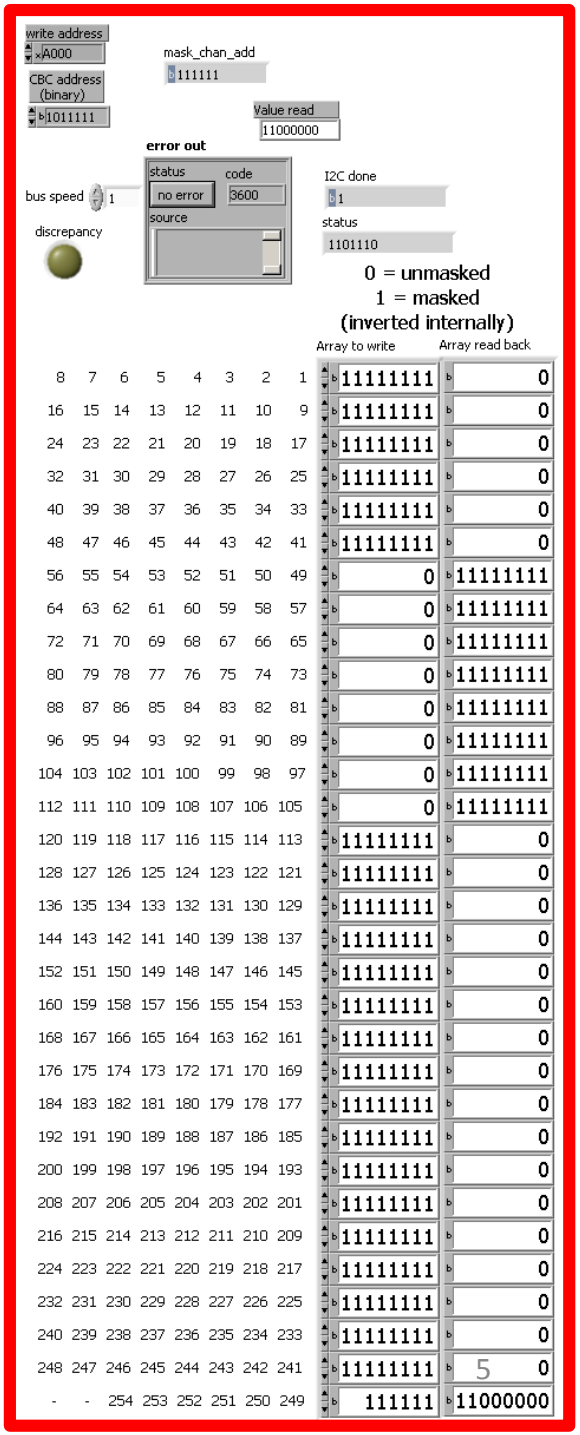
other control panels

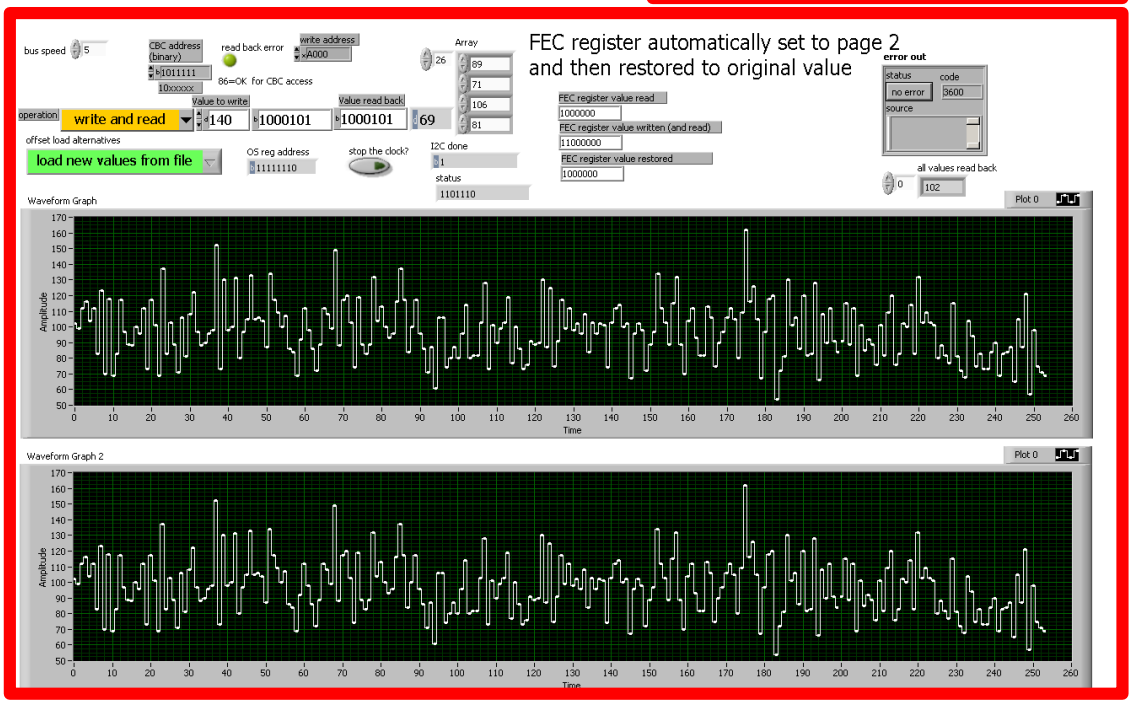
channel offsets

channel mask

bend LUTs







running the chip

many things to set up correctly

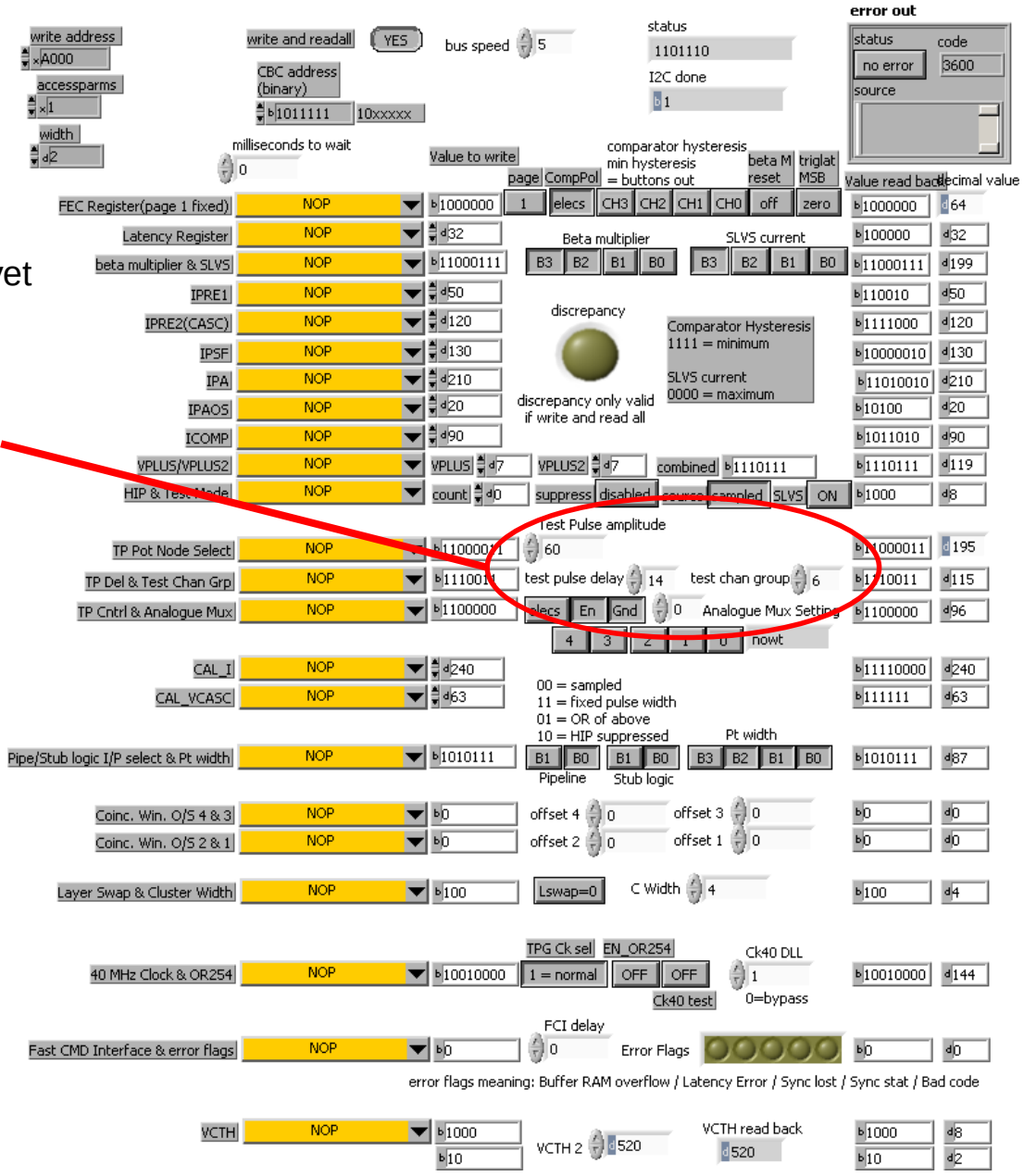
user manual will explain
draft exists - not ready for circulation yet

spec. document already goes some way

test pulse allows to exercise most features

same as CBC2 - arrangement of channels
allows to generate stubs

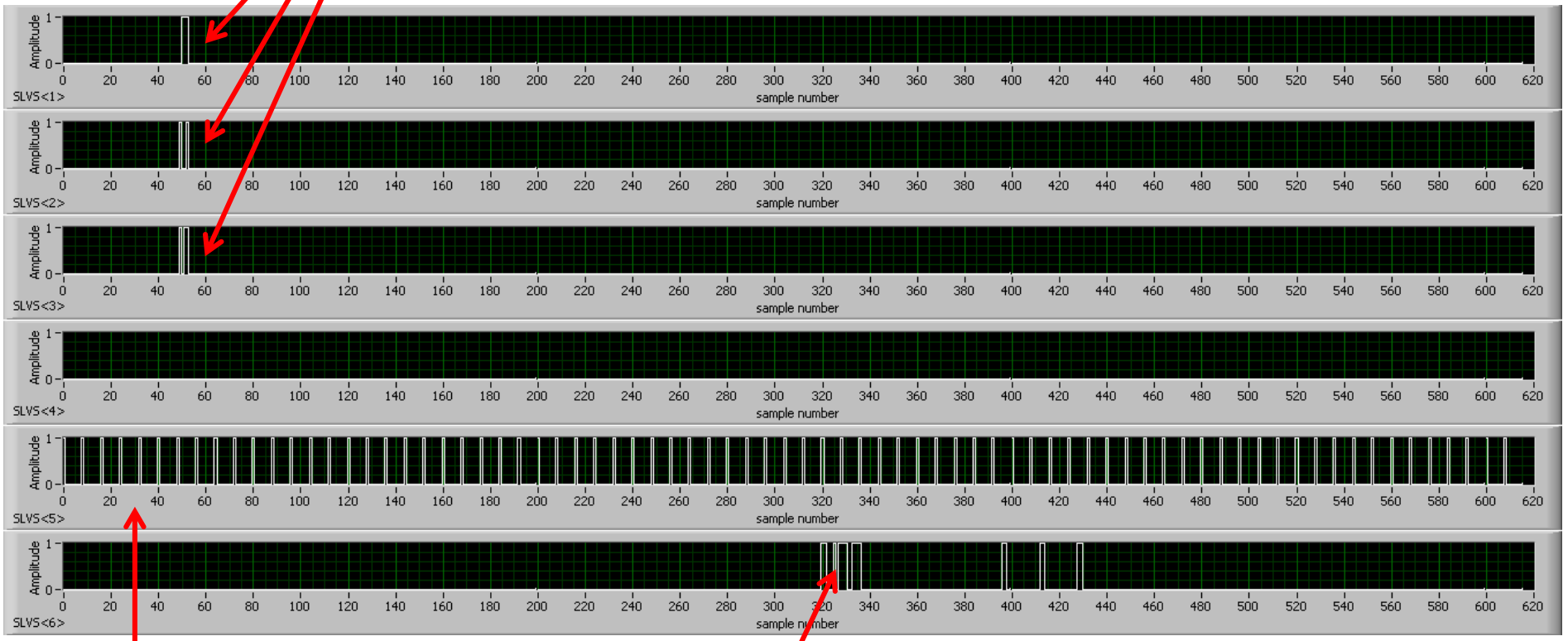
can limit number of stubs generated
using channel mask



running the chip example

example here shows activity on 6 output lines for 3 stubs generated

SLVS<1>, <2> & <3> shows stub address data



SLVS<5> shows sync pulse every 25 nsec

SLVS<6> shows digital header followed by 3 pairs of hits

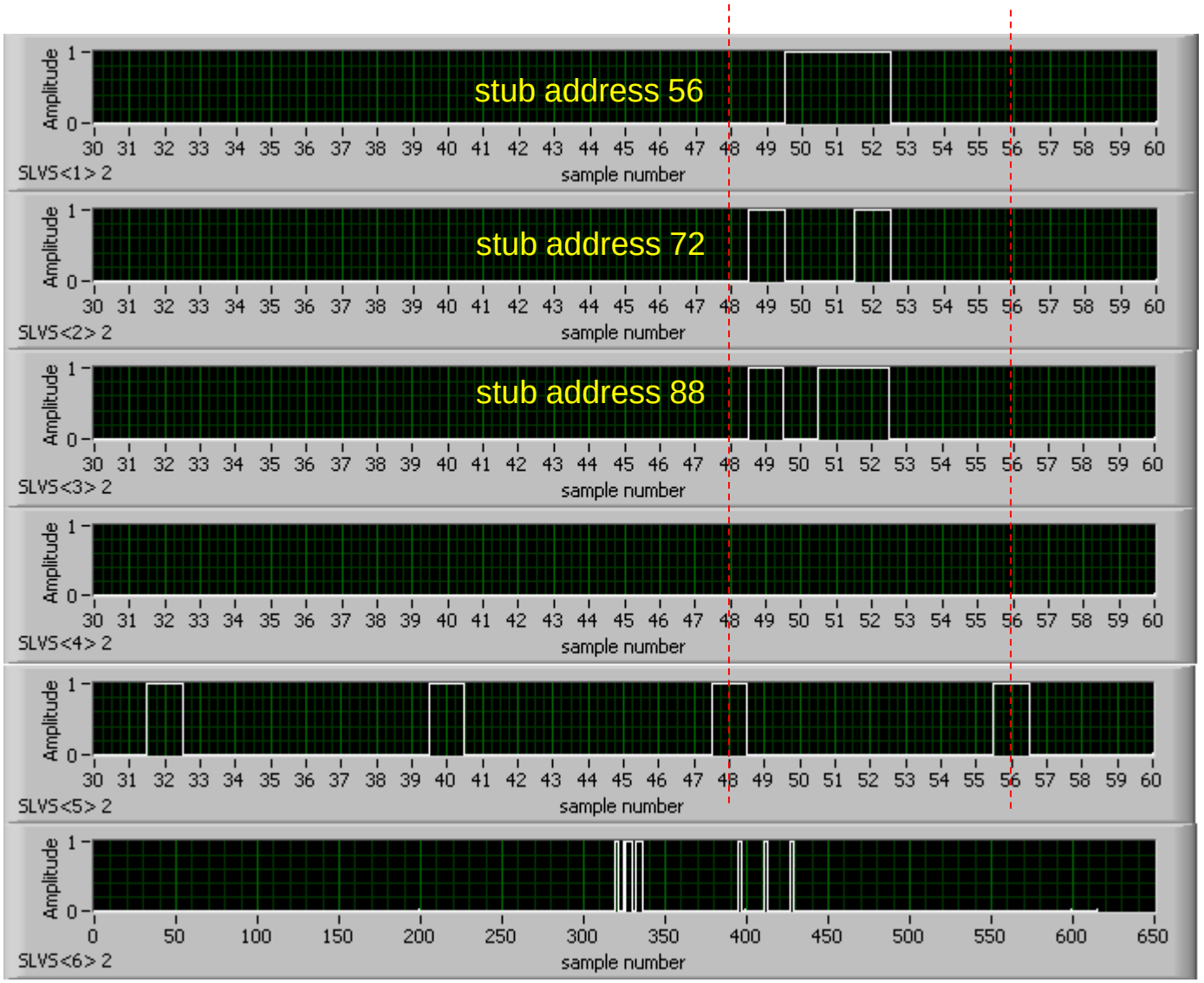
note: no bend information, because test pulse fires channels directly above each other (seed channel and channel in centre of window => bend = 0)

zooming in

test pulse timing set up so that hit confined to one timeslot

stub addresses 16 channels apart as expected from test pulse

only get even stub addresses as odd values require adjacent hits on the same layer (test pulse can't do that)



now unmask another pair of channels to generate a 4th stub

4 stubs

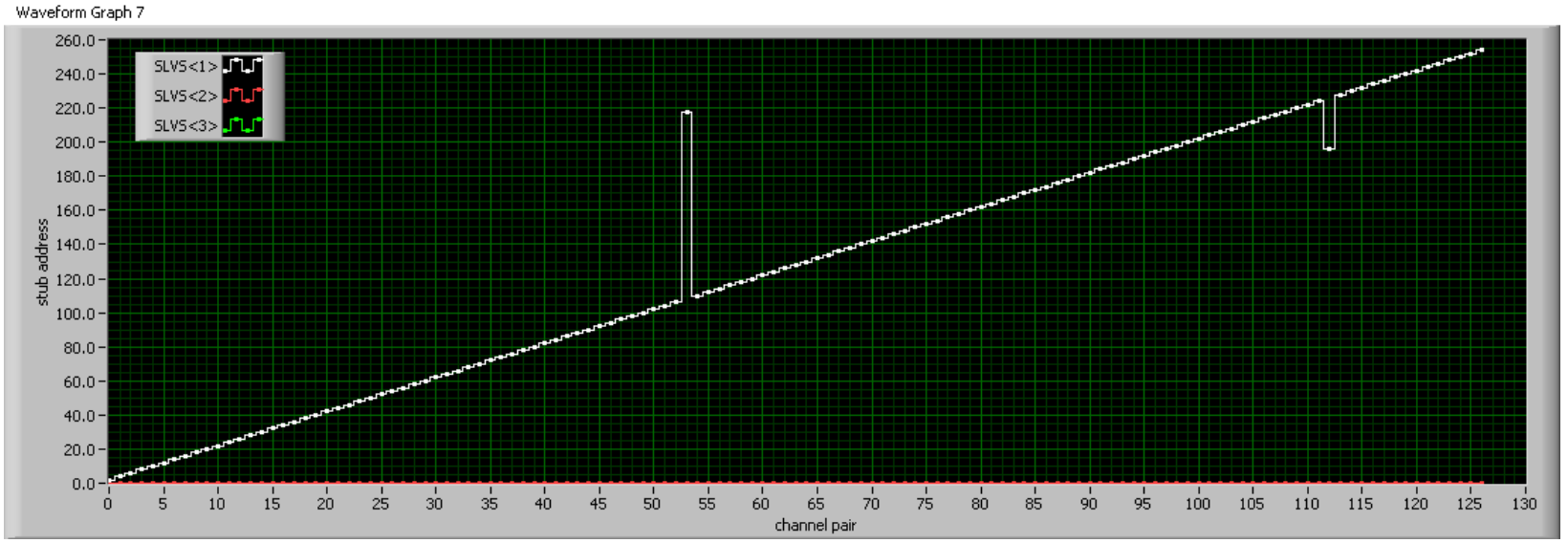
stub overflow bit
set in SLVS<5>

4 pairs of channels
in triggered data stream

everything appears to be working - can programmatically sweep test pulse through all channels and check to see all stub addresses present

sweep single stub through all channels

stub address generated by chip



channel pair fired by test pulse (127 channel pairs altogether)

stub address sweeps between 2 and 254

2 stub addresses not correct

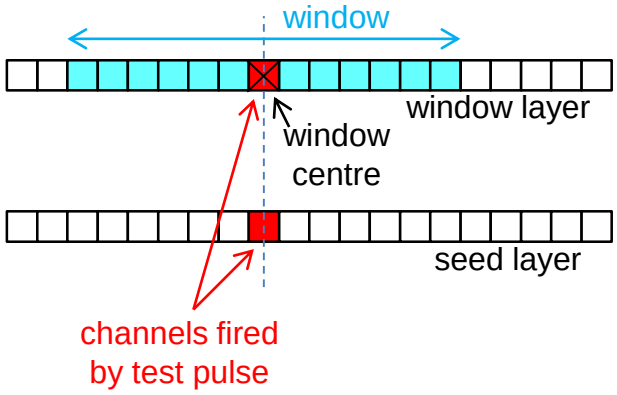
get 218 instead of 106
196 instead of 226

=> get address 218 and 196 twice

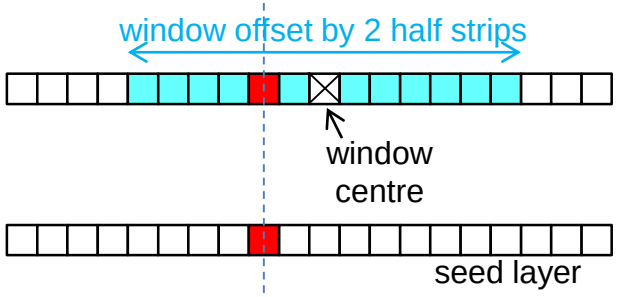
can also get bend info

test pulse fires seed layer channel and window layer channel immediately above

=> bend value 0 should be returned if no offset applied



but can change bend value by offsetting window

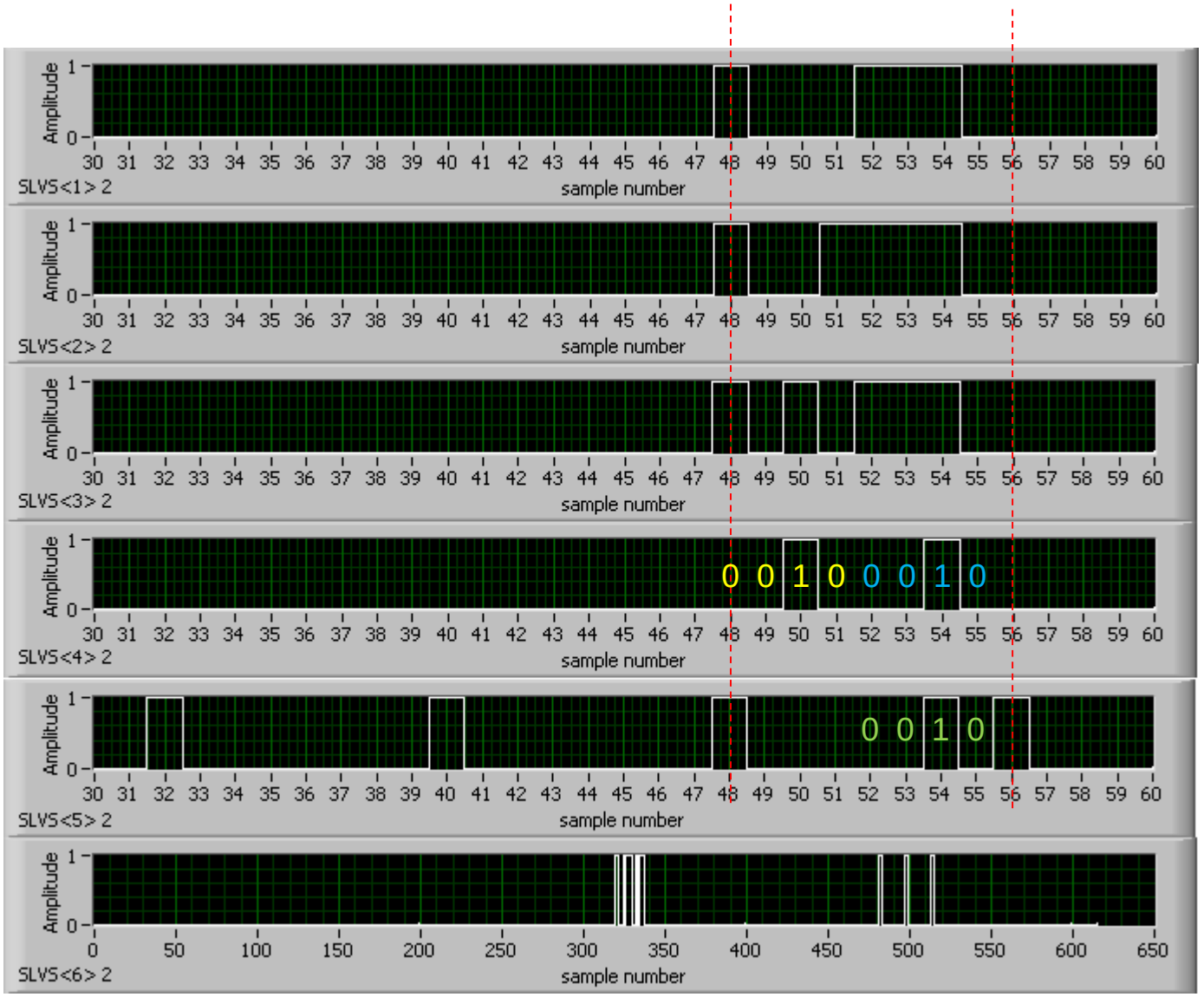


should see non-zero bend value

3 stubs + bend data

setting window offsets

bend data now appears in the expected locations



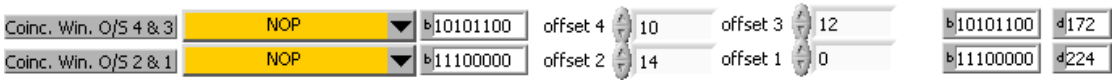
choose window offsets appropriately

for example to see bend values of

0, 1, 2, 3 (decimal)

program window offsets to

0, 14, 12, 10



desired offset [strips]	value to program [decimal]	bend value expected for test pulse hits [binary]	
-3	6	1101	
-2½	5	1101	
-2	4	1110	
-1½	3	1110	
-1	2	1111	
-½	1	1111	
0 (centre)	0	0000	(0)
+½	15	0000	
+1	14	0001	(1)
+1½	13	0001	
+2	12	0010	(2)
+2½	11	0010	
+3	10	0011	(3)

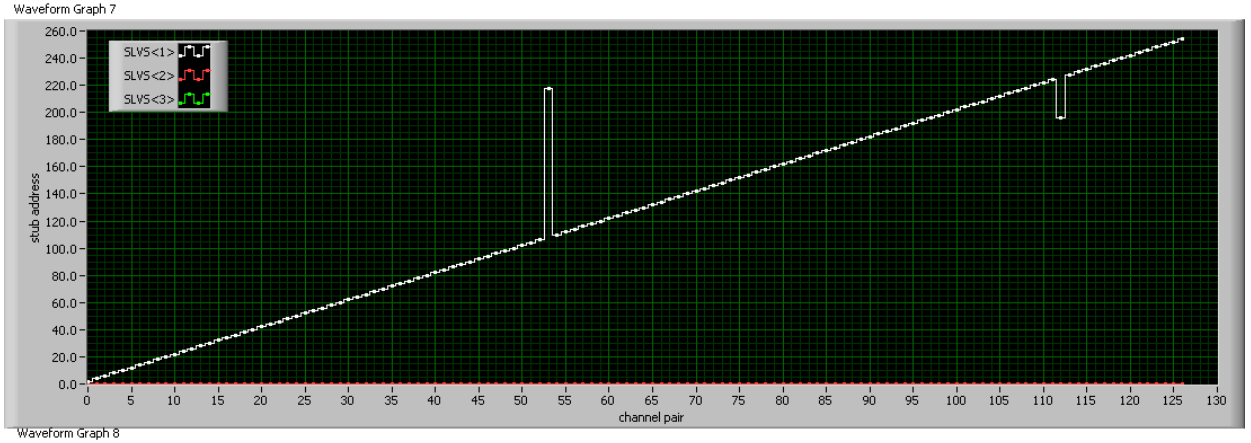
sweep stub again

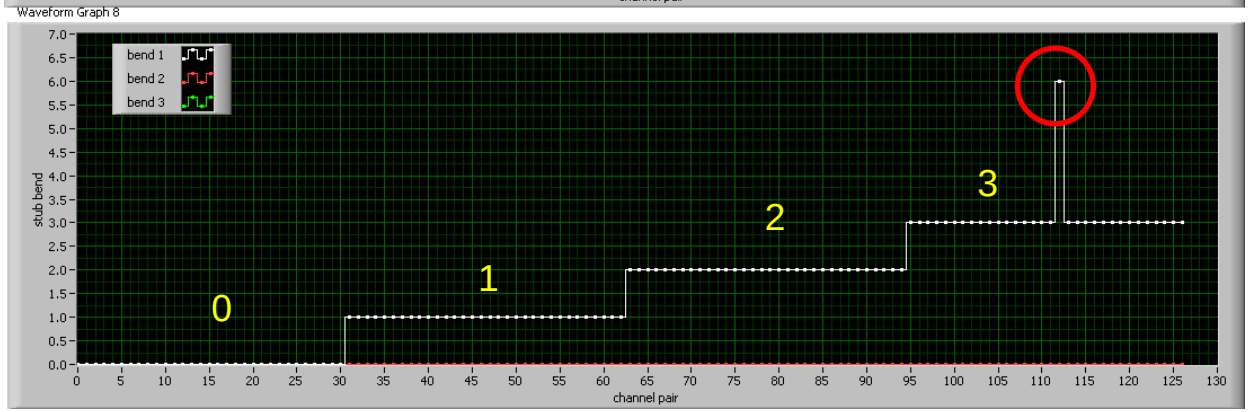
sweeping single stub through all channel pair locations

see 4 groups of channels with expected bend values

but one channel not returning expected bend value

same channel that also returns incorrect stub address





31 chans 32 chans 32 chans 32 chans

explanation of incorrect stub addresses & bend

- Verilog code for the Stub Gathering Logic has some typos that slipped through the checking process.

Extra 1 appended to 7-bit address, so truncation occurs

```
assign Stub_Info_52_Half_int = {7'b0110100,1'b1,Bend[259:255]};
wire [12:0] Stub_Info_53_int;
assign Stub_Info_53_int = {7'b0110101,1'b0,Bend[264:260]};
wire [12:0] Stub_Info_53_Half_int;
assign Stub_Info_53_Half_int = {7'b0110101,1'b1,Bend[264:260]};
wire [12:0] Stub_Info_54_int;
assign Stub_Info_54_int = {7'b01101101,1'b0,Bend[269:265]};
wire [12:0] Stub_Info_54_Half_int;
assign Stub_Info_54_Half_int = {7'b0110110,1'b1,Bend[269:265]};
wire [12:0] Stub_Info_55_int;
```

```
wire [12:0] Stub_Info_111_int;
assign Stub_Info_111_int = {7'b1101111,1'b0,Bend[554:550]};
wire [12:0] Stub_Info_111_Half_int;
assign Stub_Info_111_Half_int = {7'b1101111,1'b1,Bend[554:550]};
wire [12:0] Stub_Info_112_int;
assign Stub_Info_112_int = {7'b1110000,1'b0,Bend[559:555]};
wire [12:0] Stub_Info_112_Half_int;
assign Stub_Info_112_Half_int = {7'b1110000,1'b1,Bend[559:555]};
wire [12:0] Stub_Info_113_int;
assign Stub_Info_113_int = {7'b1110001,1'b0,Bend[564:559]};
wire [12:0] Stub_Info_113_Half_int;
assign Stub_Info_113_Half_int = {7'b1110001,1'b1,Bend[564:559]};
wire [12:0] Stub_Info_114_int;
assign Stub_Info_114_int = {7'b1110010,1'b0,Bend[569:565]};
wire [12:0] Stub_Info_114_Half_int;
```

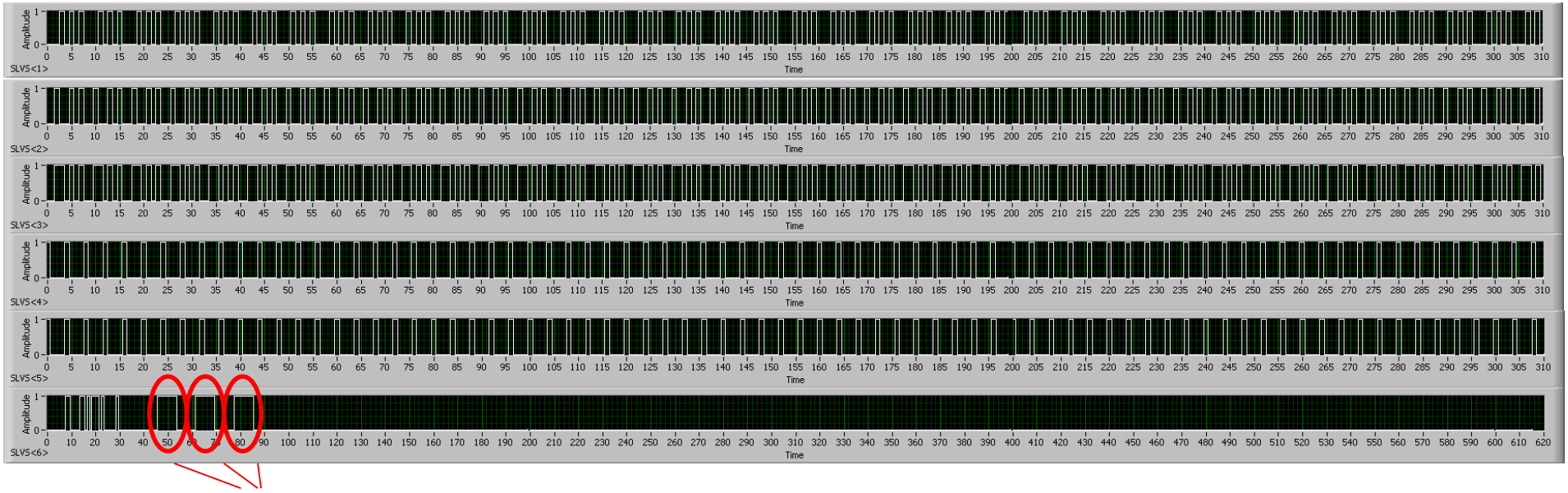
Should have been Bend [564:560]

- These channels can be masked on the existing chips and if necessary/desirable the issue can be corrected with a metal-mask-only change.
- We will review how this occurred and tighten-up our procedures to prevent future occurrences.

force continuous stub generation

set VCTH threshold so all channels constantly firing

selectively unmask channels to generate clusters



3 x 4 strip seed and window clusters

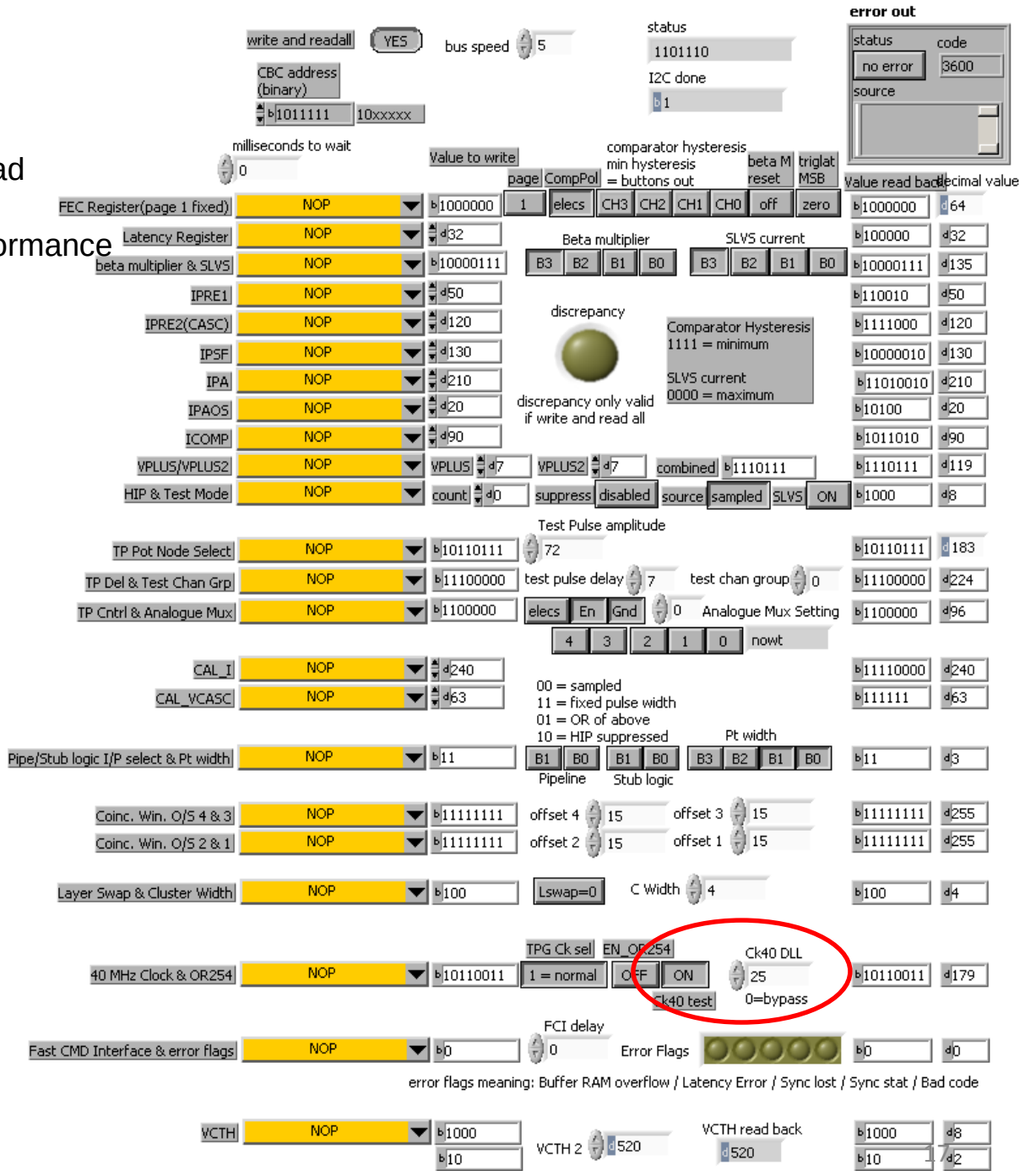
can use this technique to generate fixed pattern in SLVS output lines
also to verify cluster width discrimination logic, Pt window width logic, layer swap, ...
(principle proven - exhaustive check not yet implemented)

Ck40 test

can switch output of DLL to a test pad

test feature only - to verify DLL performance

for normal operation leave OFF

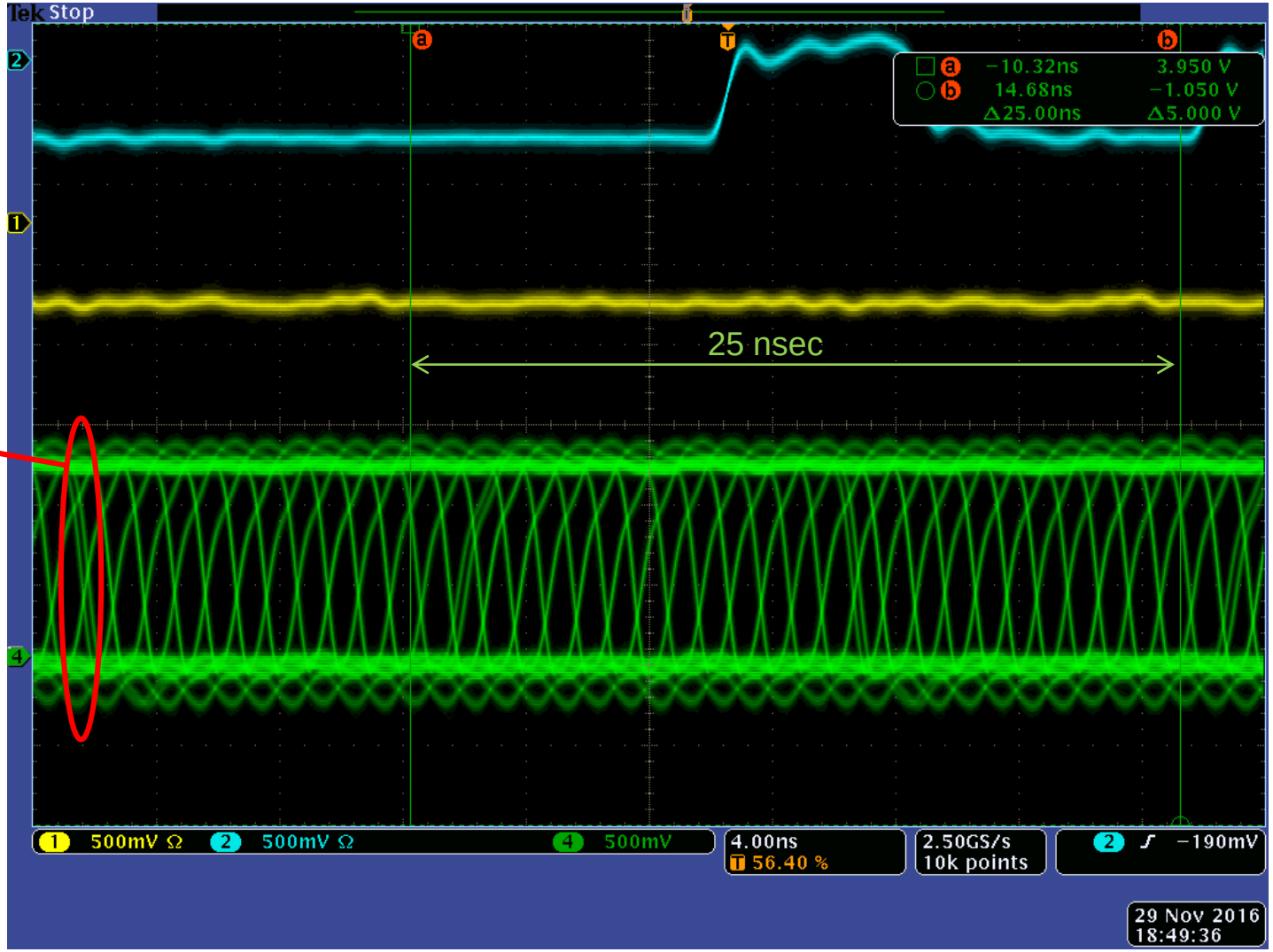


Ck40 test

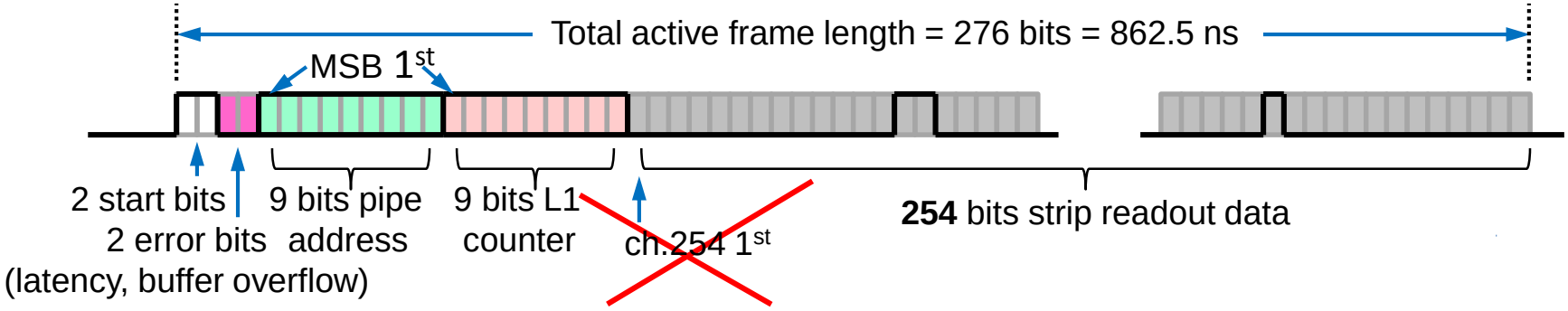
scope on infinite persistence

select Ck40DLL taps one at a time

small difference between 25 nsec delay and bypass due to different signal path lengths



deviation from specification



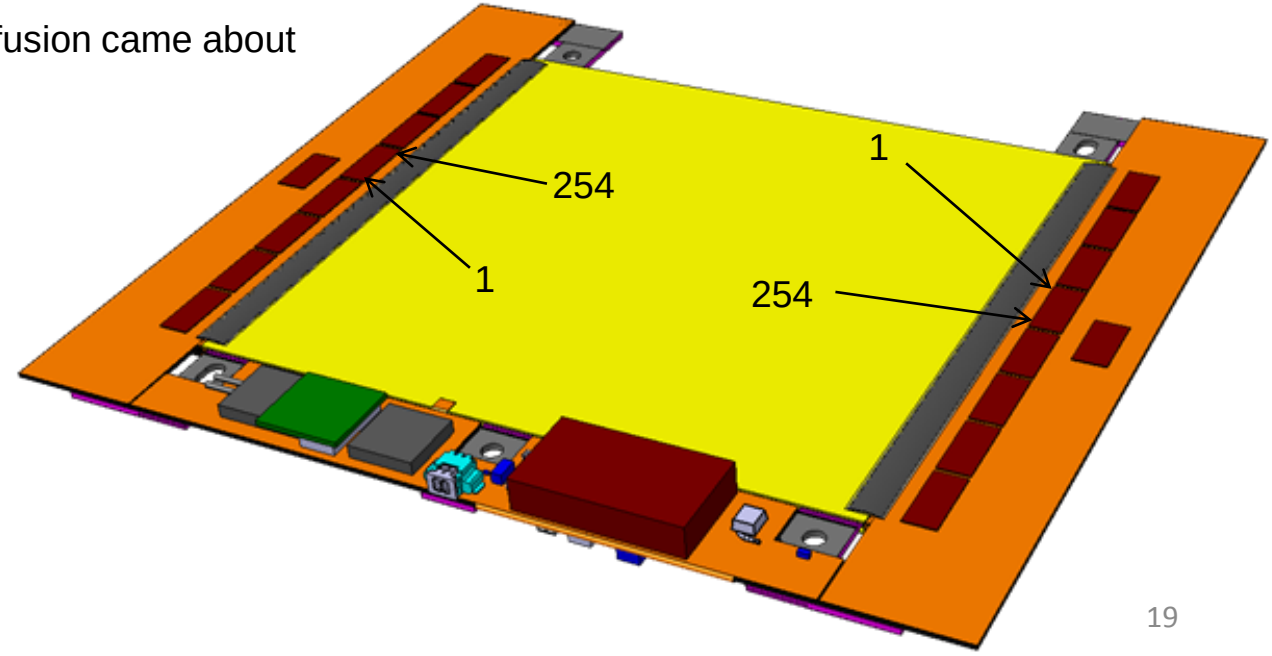
specification says channel 254 first

chip **actually** produces channel 1 first (same as CBC2)

not clear to me how this confusion came about

is it a big problem? (for CIC)

can we just modify spec.?



conclusions on digital functionality so far

chip is working well

a few bugs (that will have to be fixed) but nothing disastrous

plenty left to test …

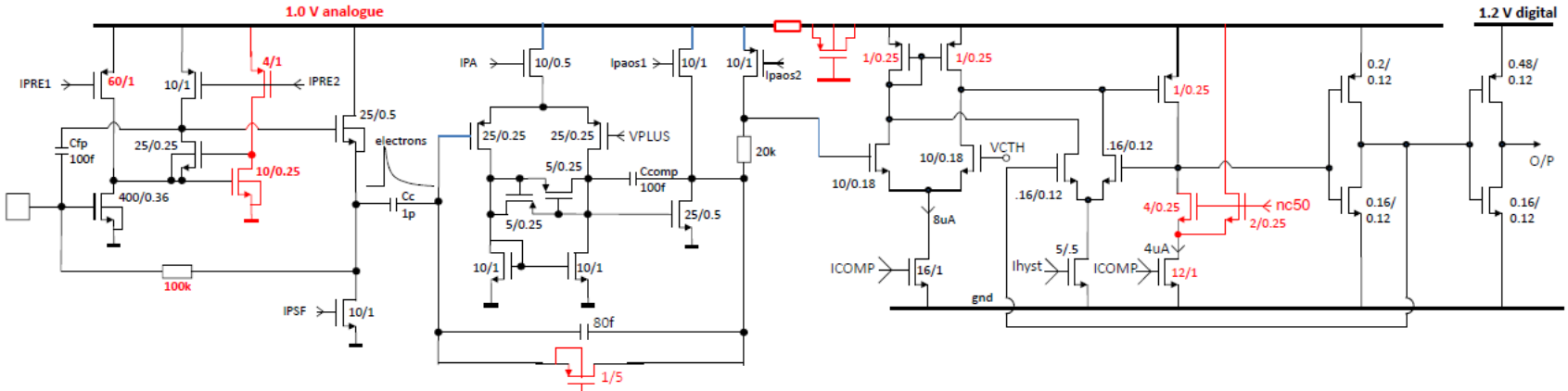
move now onto some analogue results

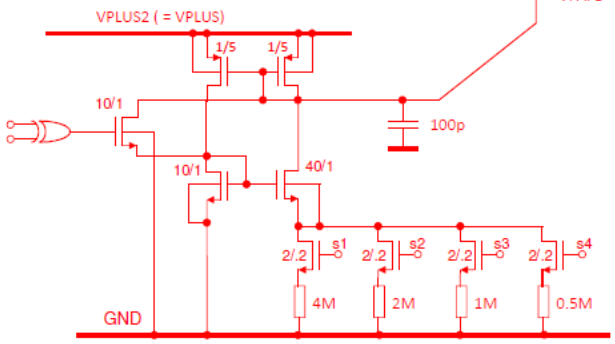
main analogue changes from CBC2

provision to run more current in input device if required

new preamp cascode bias scheme to eliminate "shadow effect"

pre & postamp polarity switch options removed





new postamp feedback bias scheme & current neutral comparator addresses CM effects observed when many channels fire

global comparator threshold voltage VCTH generated by 10-bit resistor ladder DAC for linearity & monotonicity

bandgap and VDDA

bandgap has 6 bits tuning register
to compensate for process variations

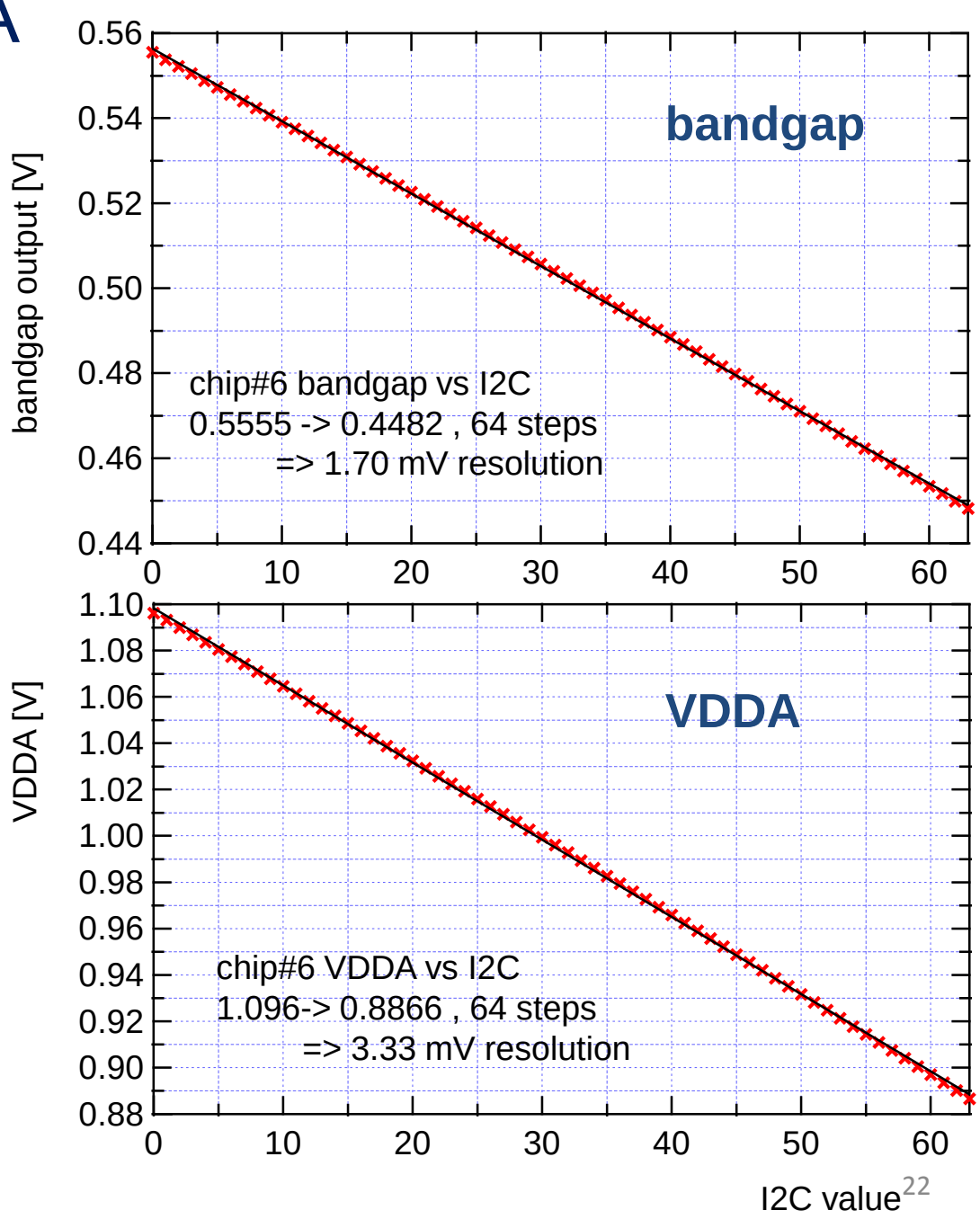
once value is chosen, can blow fuses
to store it as default

e-fuse operation not yet looked at

LDO provides VDDA

value = 2x bandgap

LDO clearly working, but no detailed
studies yet



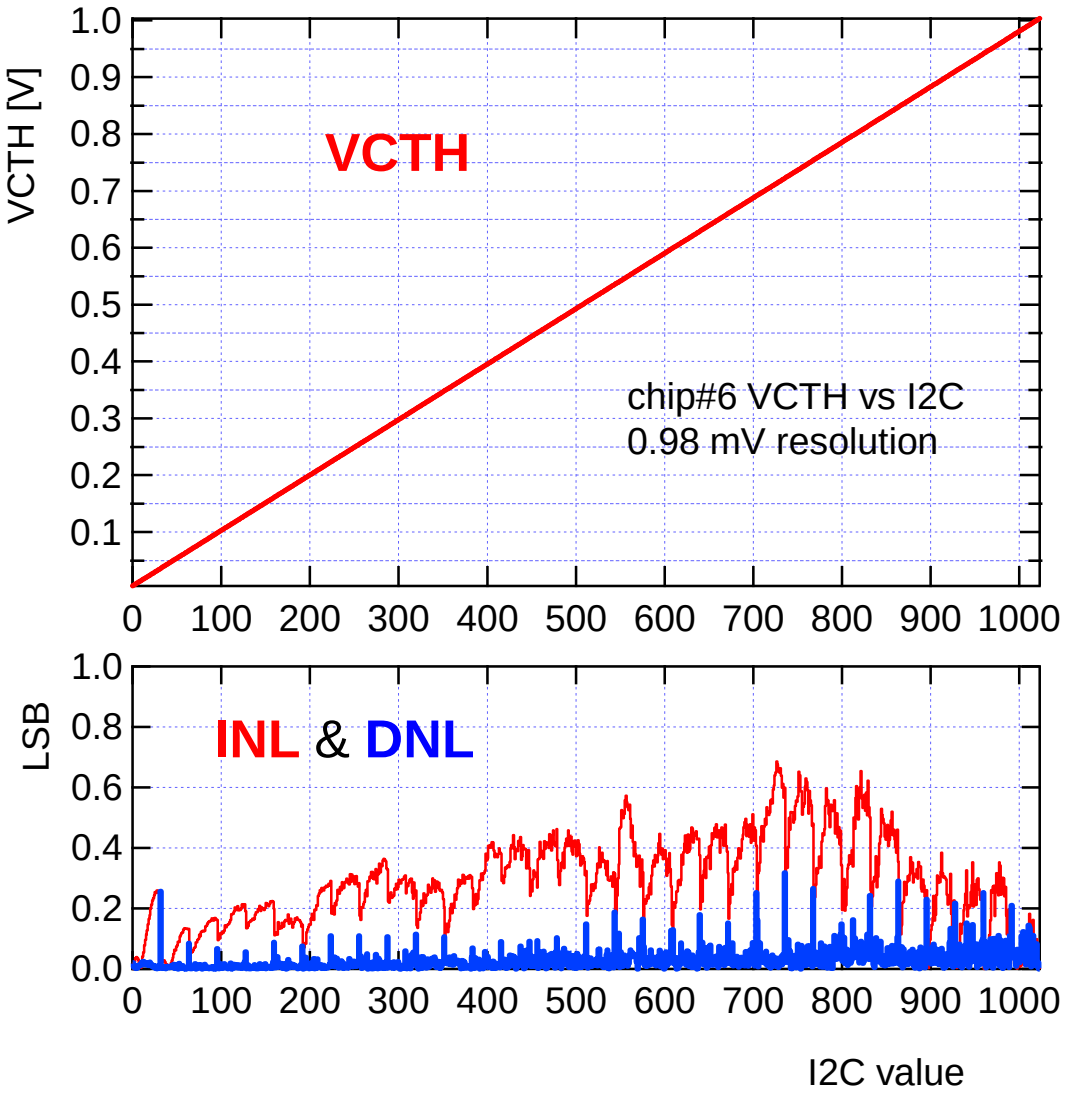
VCTH

VCTH now generated by 10-bit resistor ladder DAC

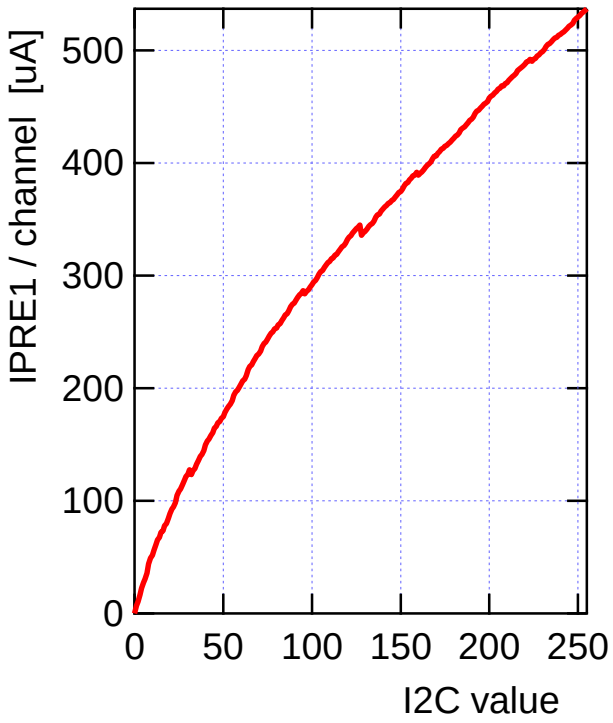
~ 1 mV resolution (~ 150 electrons)

$$INL = \left| \left(\frac{V_{meas} - V_{zero}}{V_{LSB\text{-}IDEAL}} \right) - I2C_{value} \right|$$

$$DNL = \left| \left(\frac{V_{m+1} - V_m}{V_{LSB\text{-}IDEAL}} \right) - 1 \right|$$



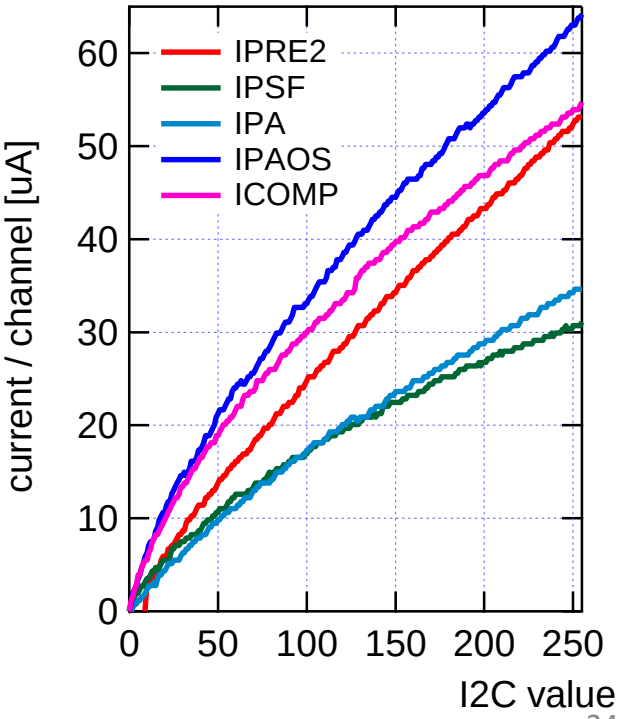
bias currents



IPRE1 (main source of current in input transistor)
now has ~3x fullscale range compared with CBC2

=> can cope with larger sensor capacitance

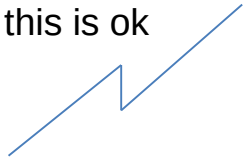
other current biases
also ok



offsets tuning change

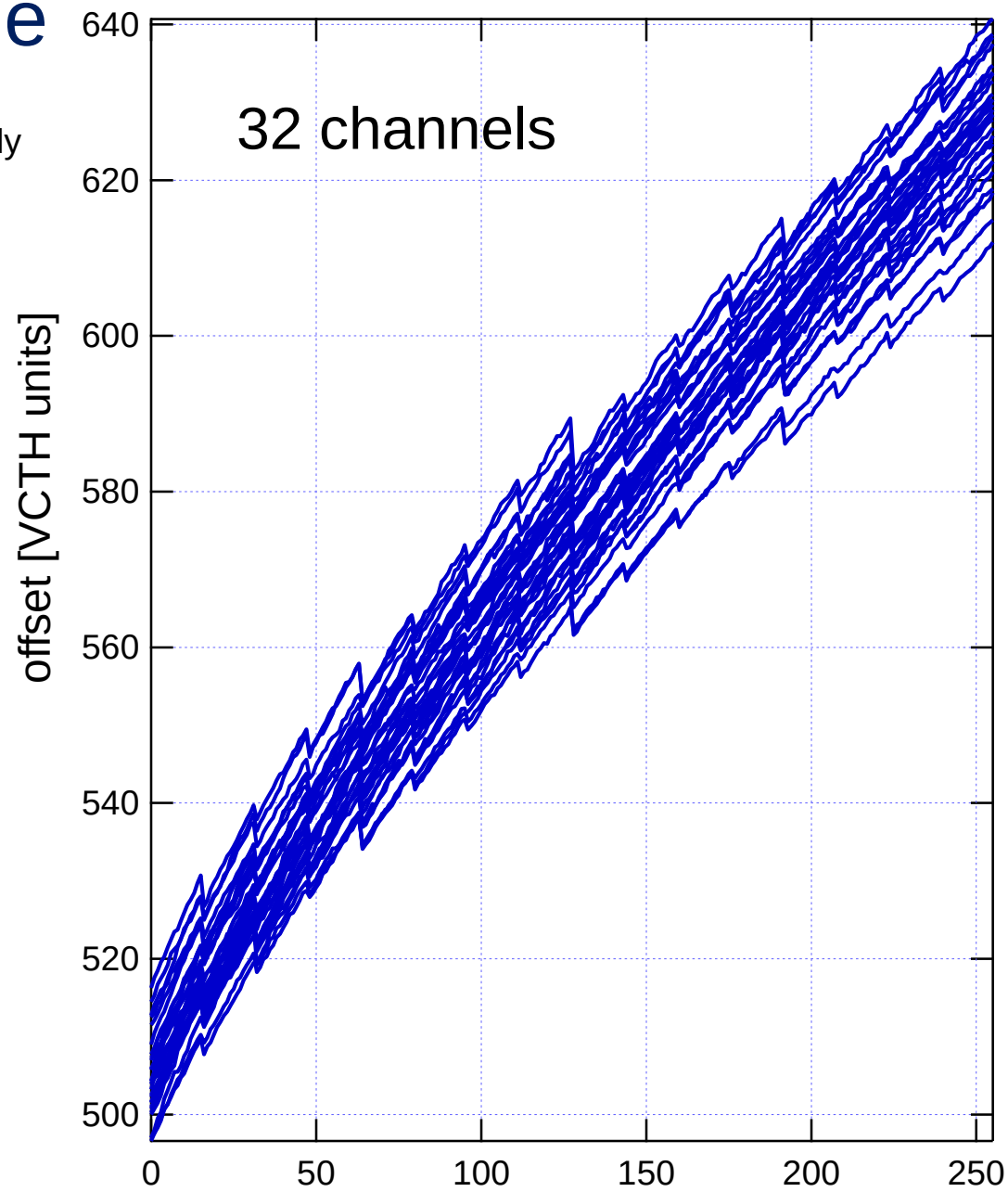
CBC2 offsets tuning mechanism deliberately made non-monotonic to avoid gaps where can't set offset accurately

this is ok



don't want this

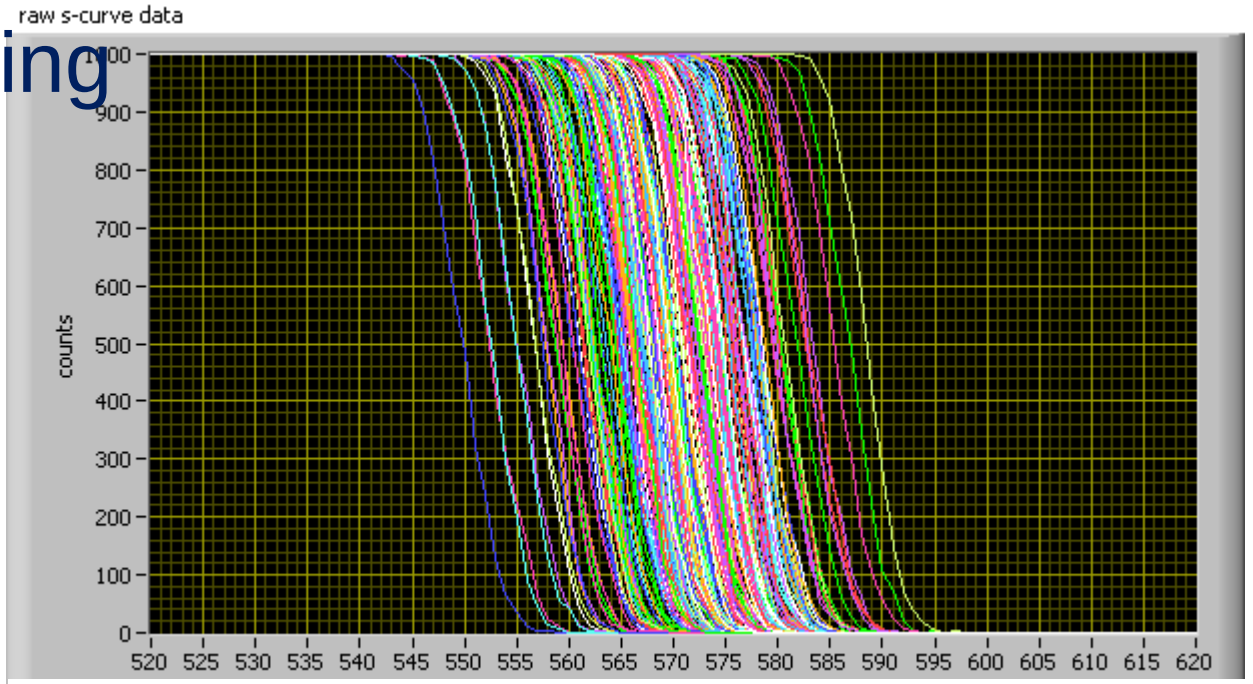
see * for more details



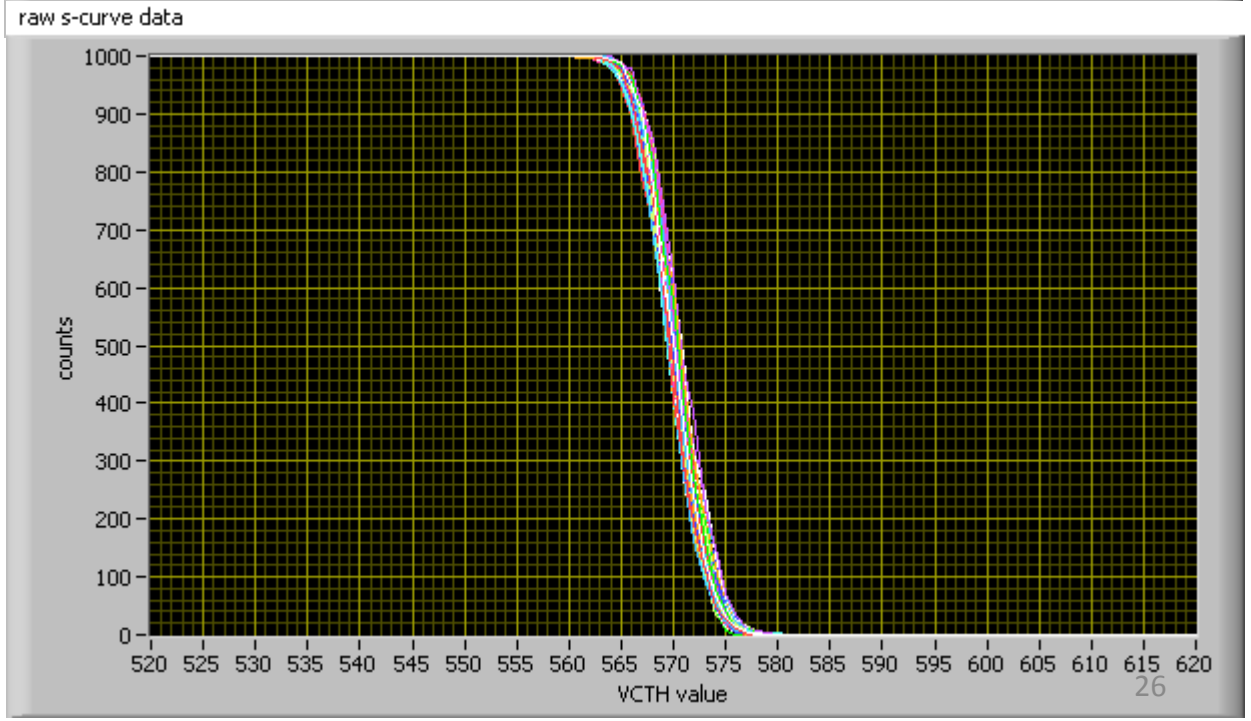
* http://www.hep.ph.ic.ac.uk/~dmray/systems_talks/2016/CBC3_systems_Jan2016.pdf

scurves and tuning

all channel offsets set to
same value (110 in this case)



after tuning
32 channels at a time



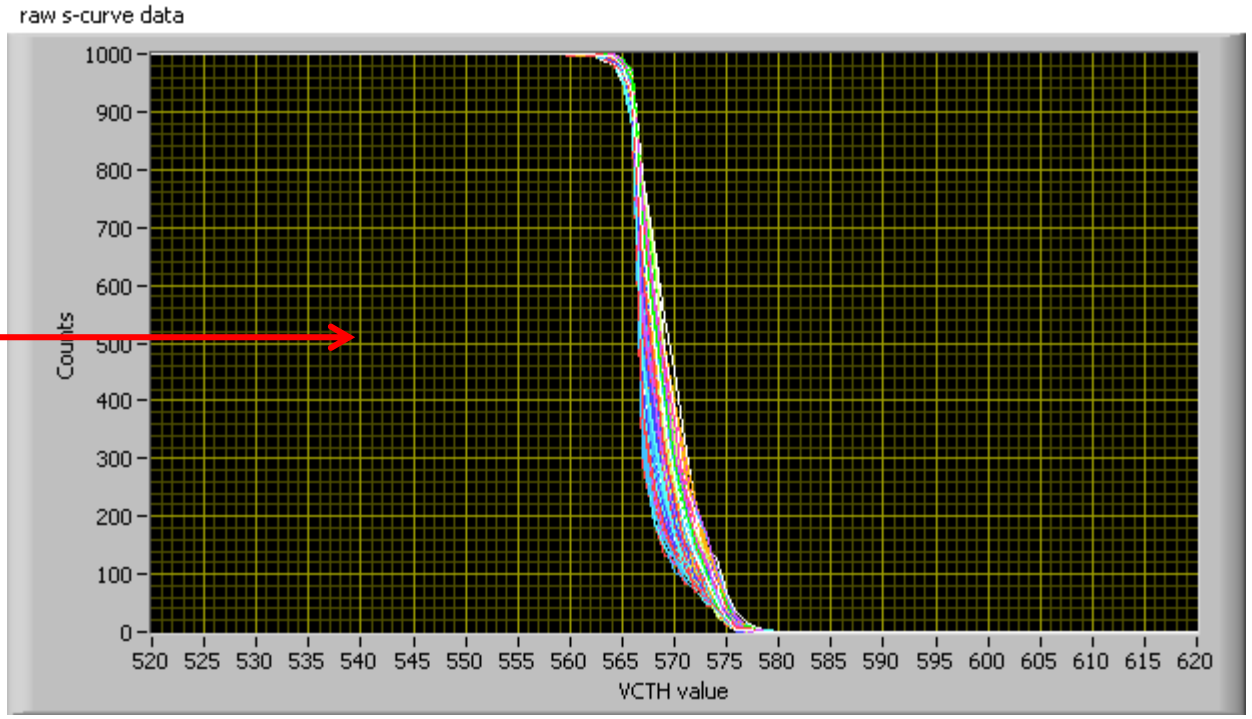
scurves and tuning

can s-curves be acquired for all channels simultaneously?

not without some distortion

~12 mA increase in VDDD current consumption during period of maximum channels firing activity

but should work better when chip is bump-bonded



=> promising for antenna tests on hybrids

hit detect cct tests

hit detect sensitive to short pulses that only exceed comp. thresh. for short period between clock edged

-> *Fixed Pulse Width*

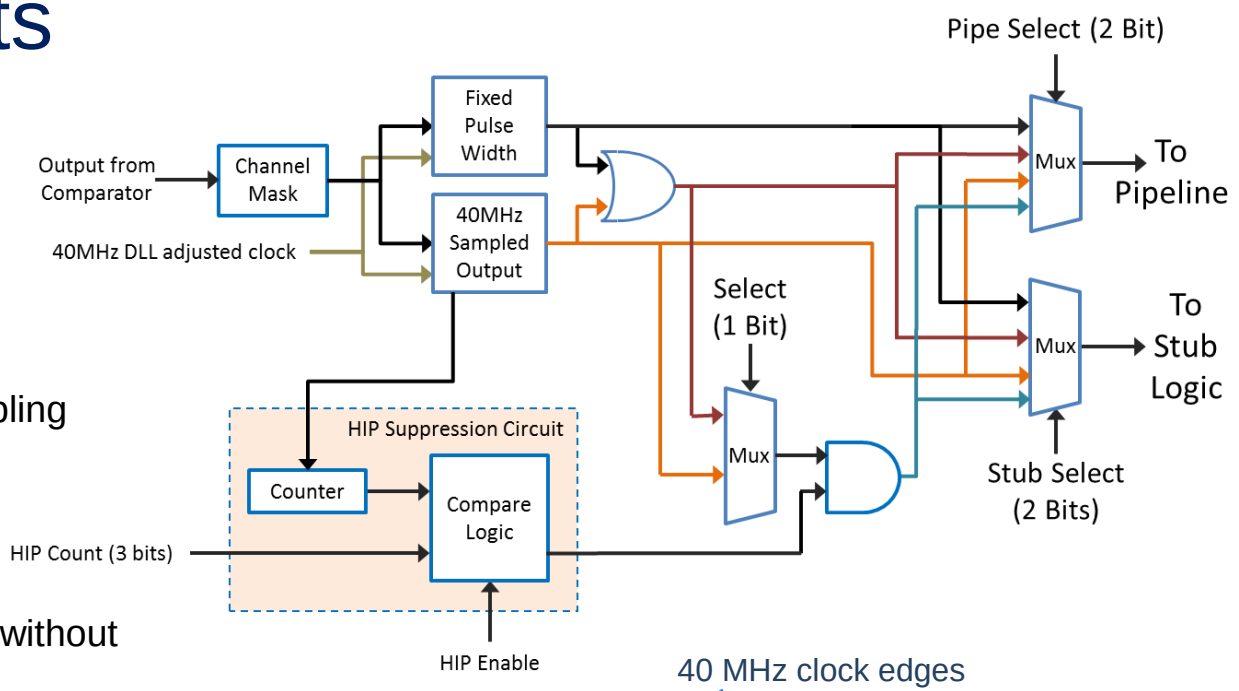
also performs simple 40 MHz sampling

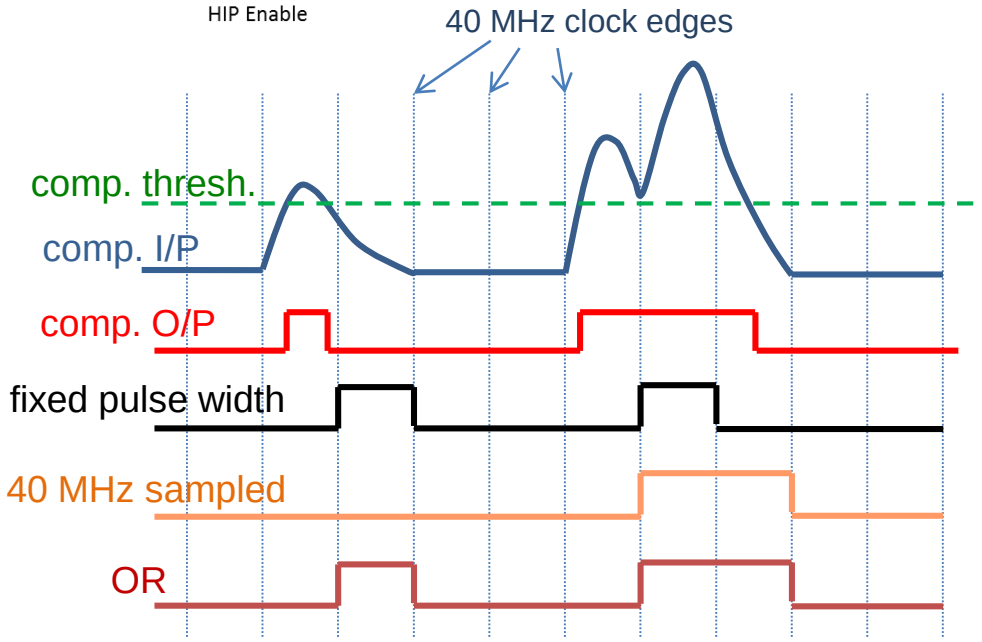
-> *40 MHz Sampled Output*

combining the two by simple OR gives efficiency for piled up pulses without inefficiency for smaller signals

2 separate multiplexers gives flexible choice of which signals can be fed to pipeline and stub logic

(for normal operation choose OR output to both **and** enable HIP suppression)

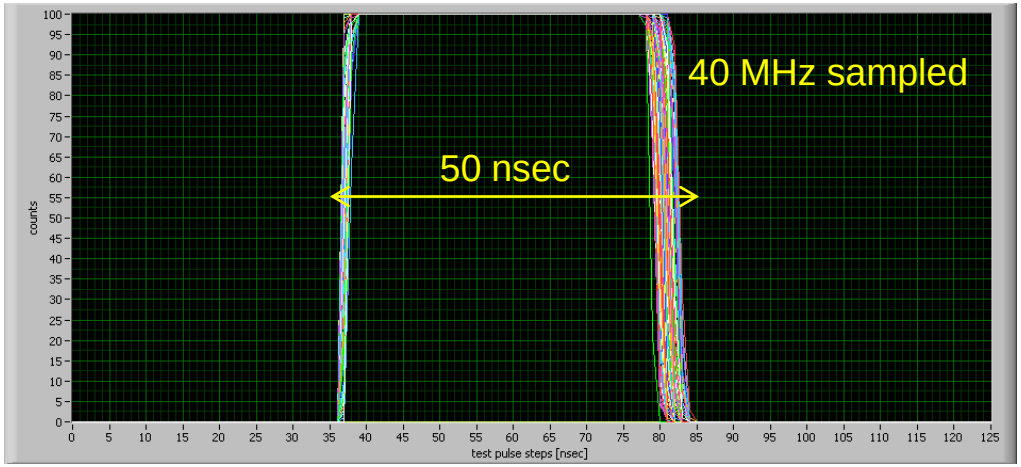


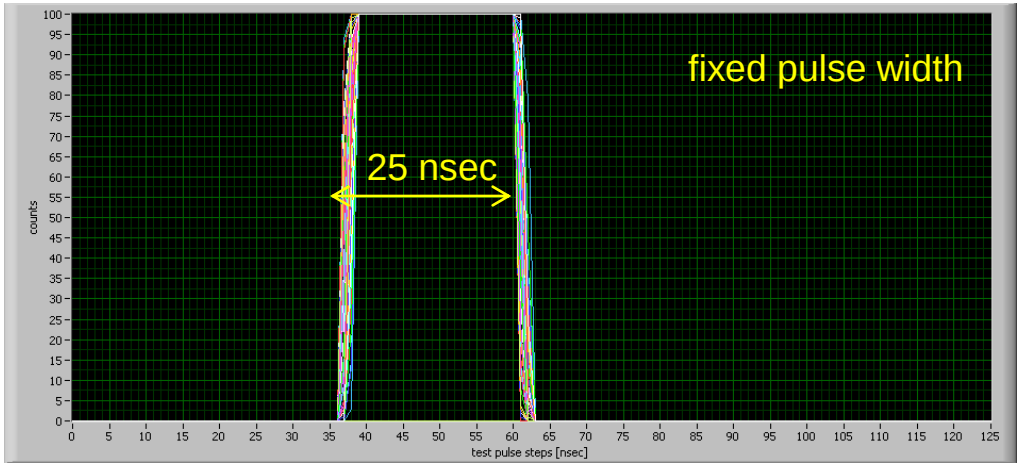


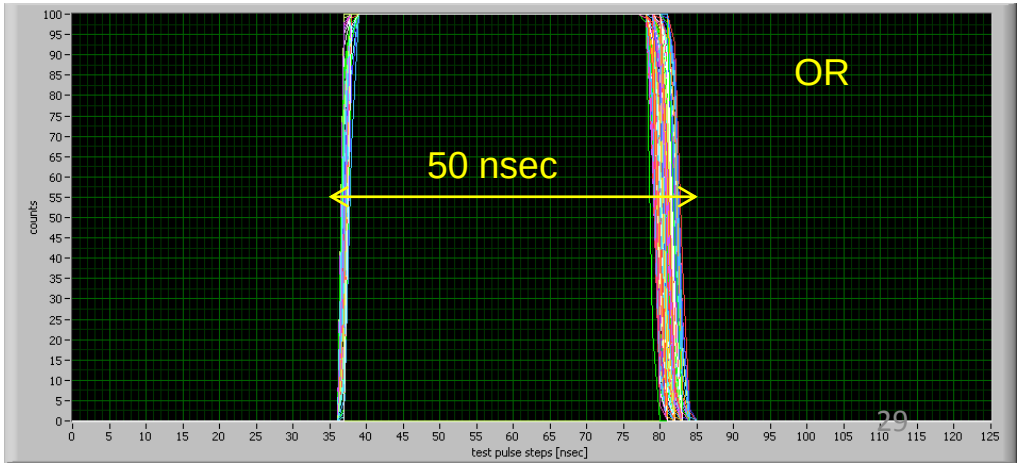
hit detect cct tests

verify operation using test pulse, sweeping test pulse trigger in 25 nsec steps, using test pulse DLL to give finer steps of 1 nsec.

pictures here for test pulse amplitude of 60 (decimal), ~ 5 fC







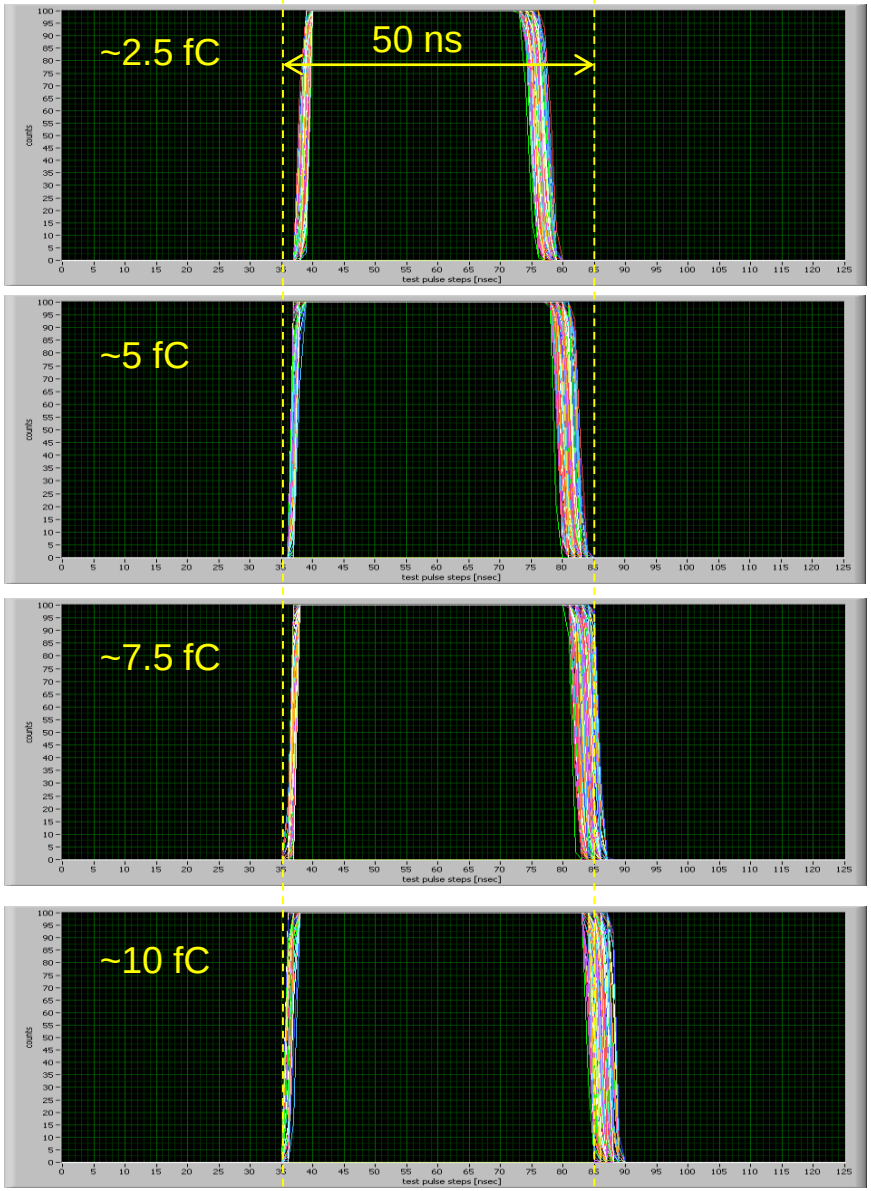
test pulse sweeps

use test pulse to look at signal duration

test pulse amplitudes approximate only

VCTH set to ~ 1.25 fC

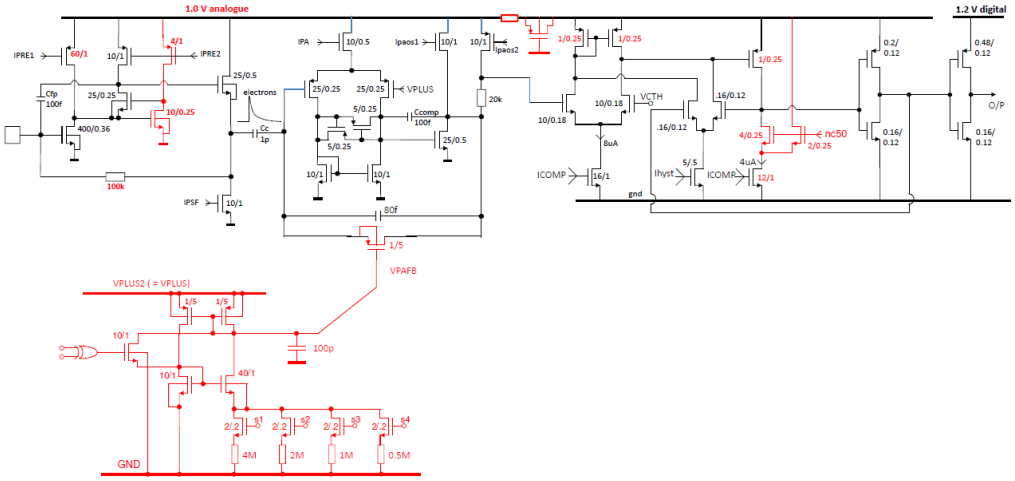
pulse width ~ ok for nominal postamp feedback settings

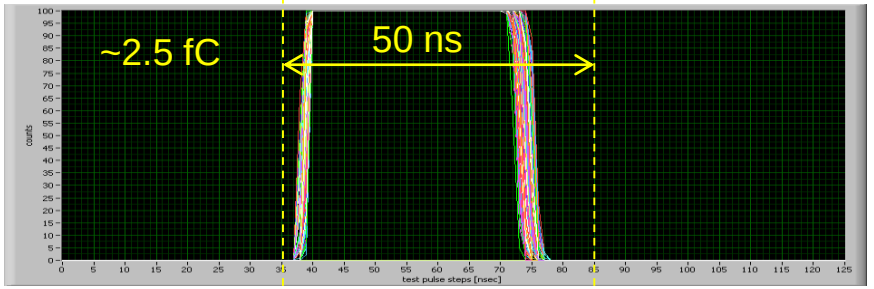


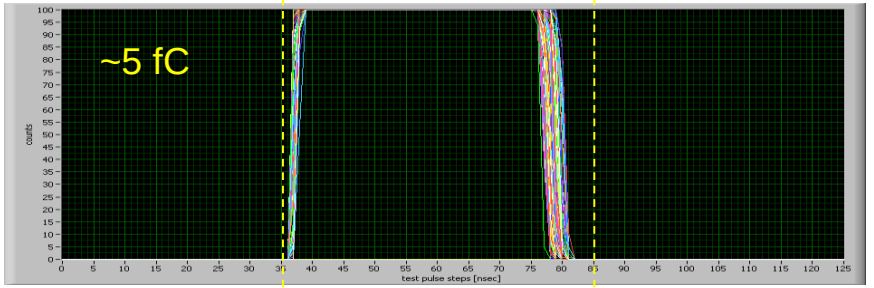
test pulse sweeps

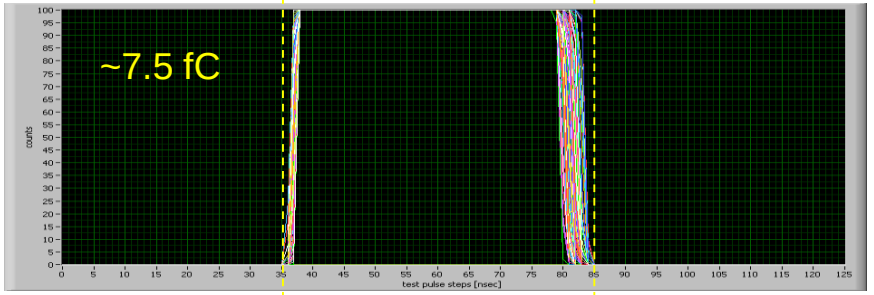
adjustment of pulse width possible using post-amplifier feedback value VPAFB

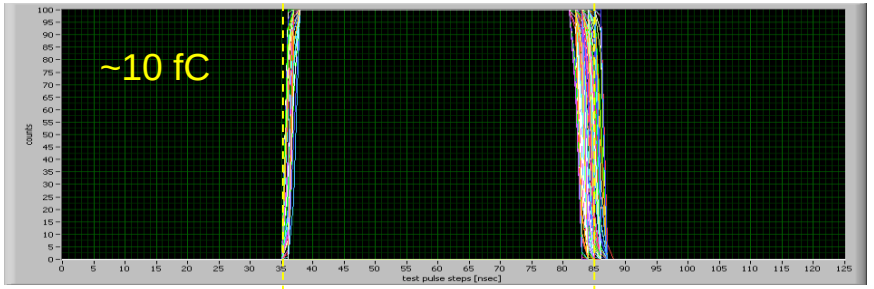
result for minimum resistance



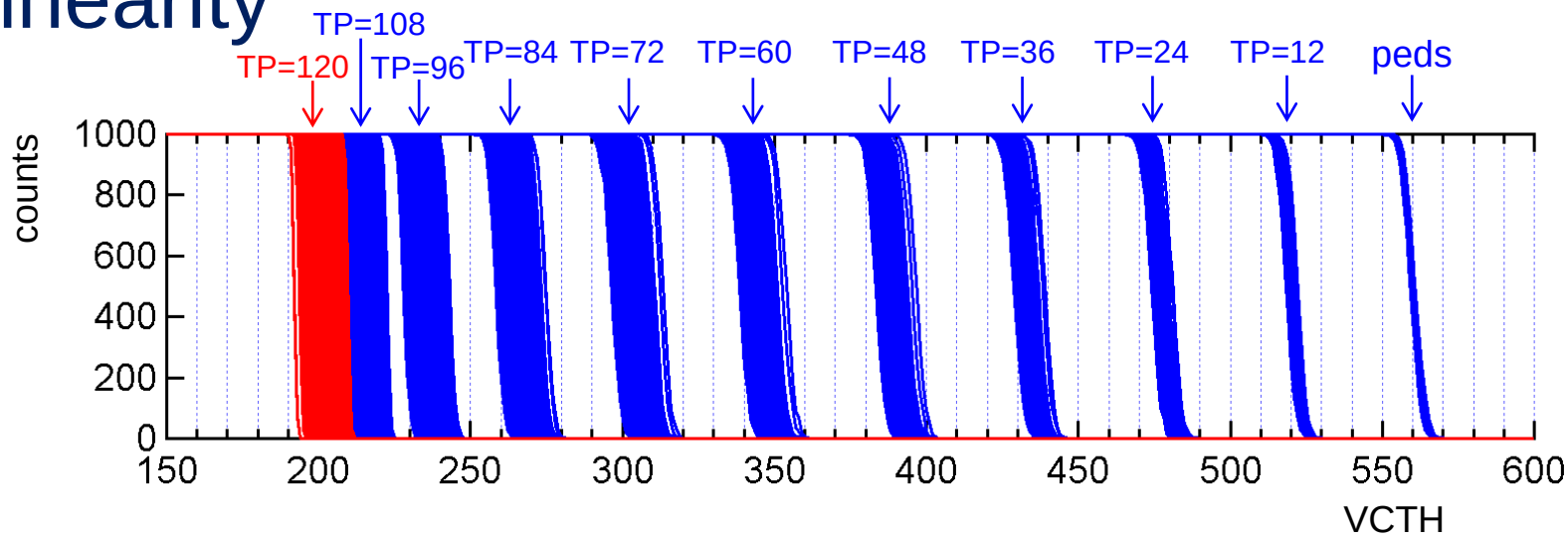








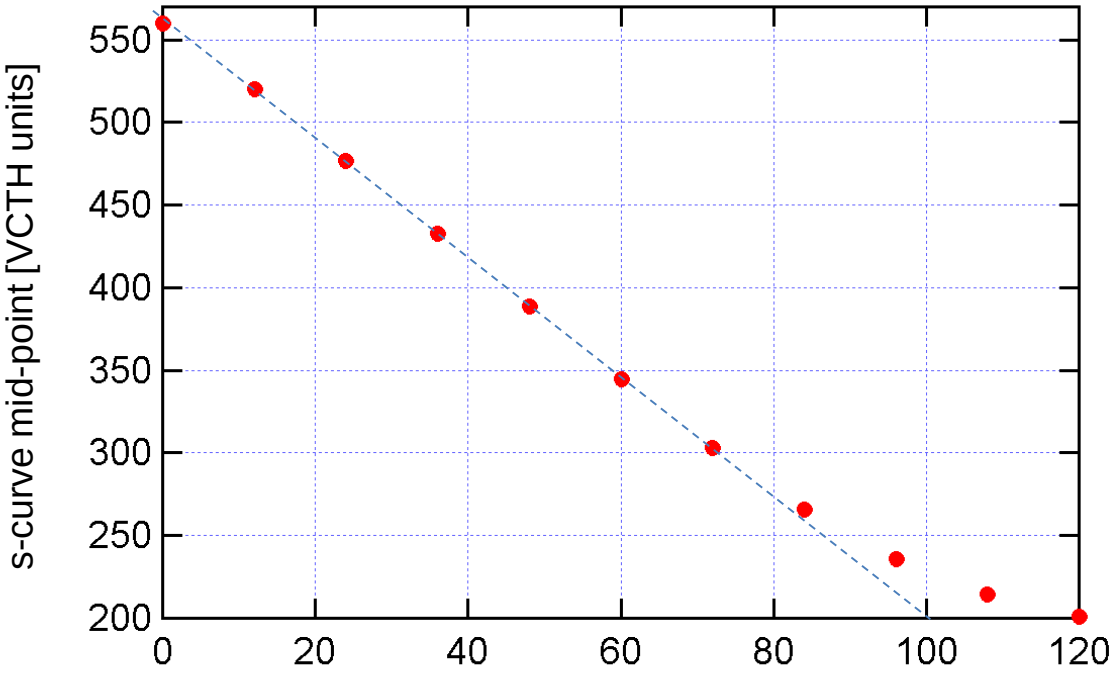
gain & linearity



using approximate calibration

(TP value of 12 = 1 fC)

get ~ 40 mV / fC



power

digital power higher than hoped

for VDDD = 1.2V measure 33.6 mA, equivalent to 160 uW / channel

(if switch SLVS drivers off get 20.7 mA)

=> overall chip power goes to ~ 510 uW/channel, unless sacrifice some analogue power

note: analogue power calculations based on 1.2V (not 1.25)

CBC3 spec. “The target power consumption for 5cm strips (~8pF) is 450 µW/Channel, assuming 350 µW/Channel analogue and 100 µW/Channel digital power consumption.”

yield

1st batch of 9 chips bonded

5 ok, 4 had problems

3 drew no power, showed no activity
1 drew high current

2nd batch of 9 chips

all ok (power, I2C ok, produce data)

summary

still early days, but CBC3 is working

- some digital issues to work around, but nothing to stop progress

analogue front end appears ok

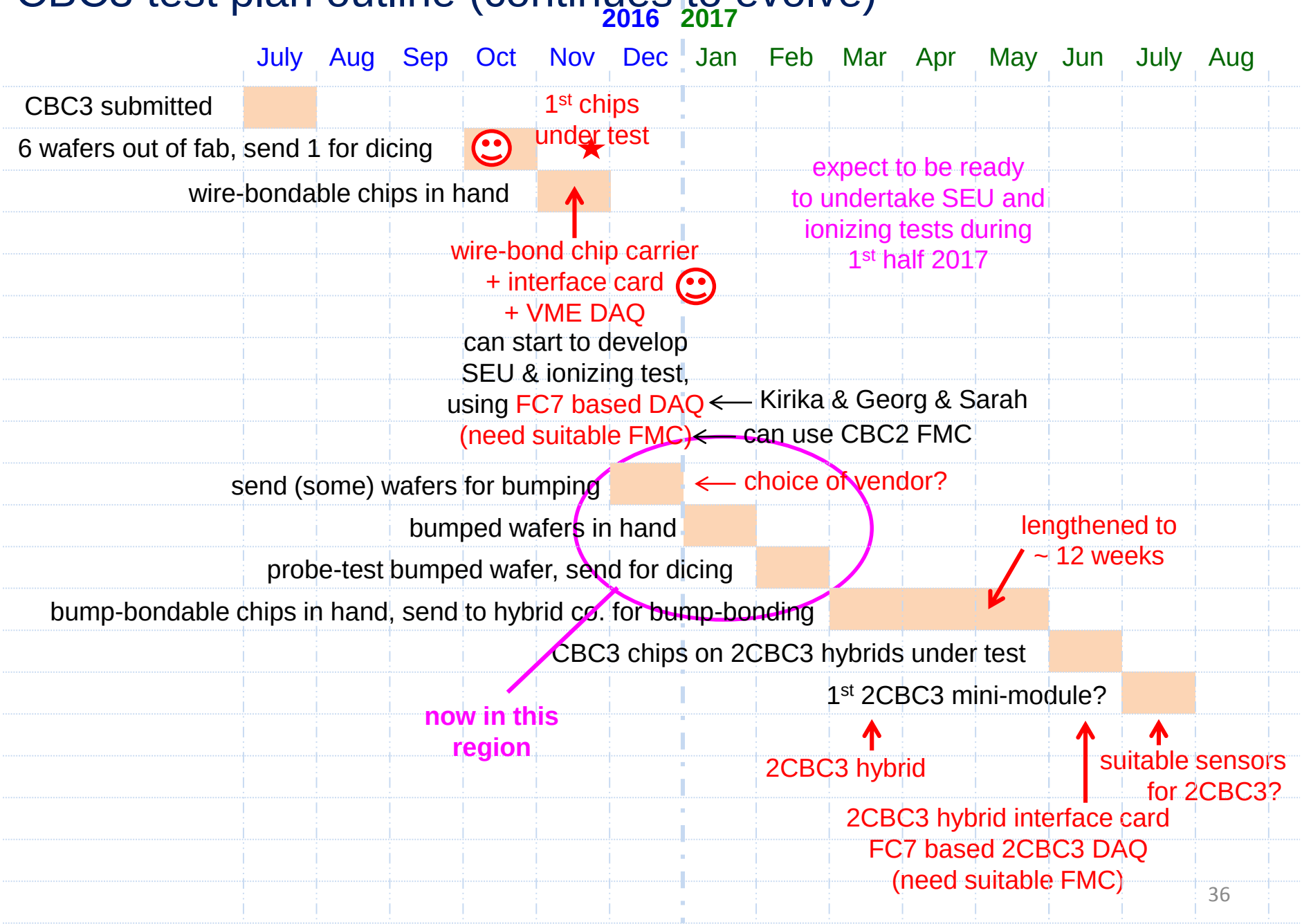
- more to investigate, but need to bump-bond and connect to sensors to get true performance

SEU & ionizing irradiation test procedures can now be developed based on wirebonded single chip setup

next priority to develop wafer probe test

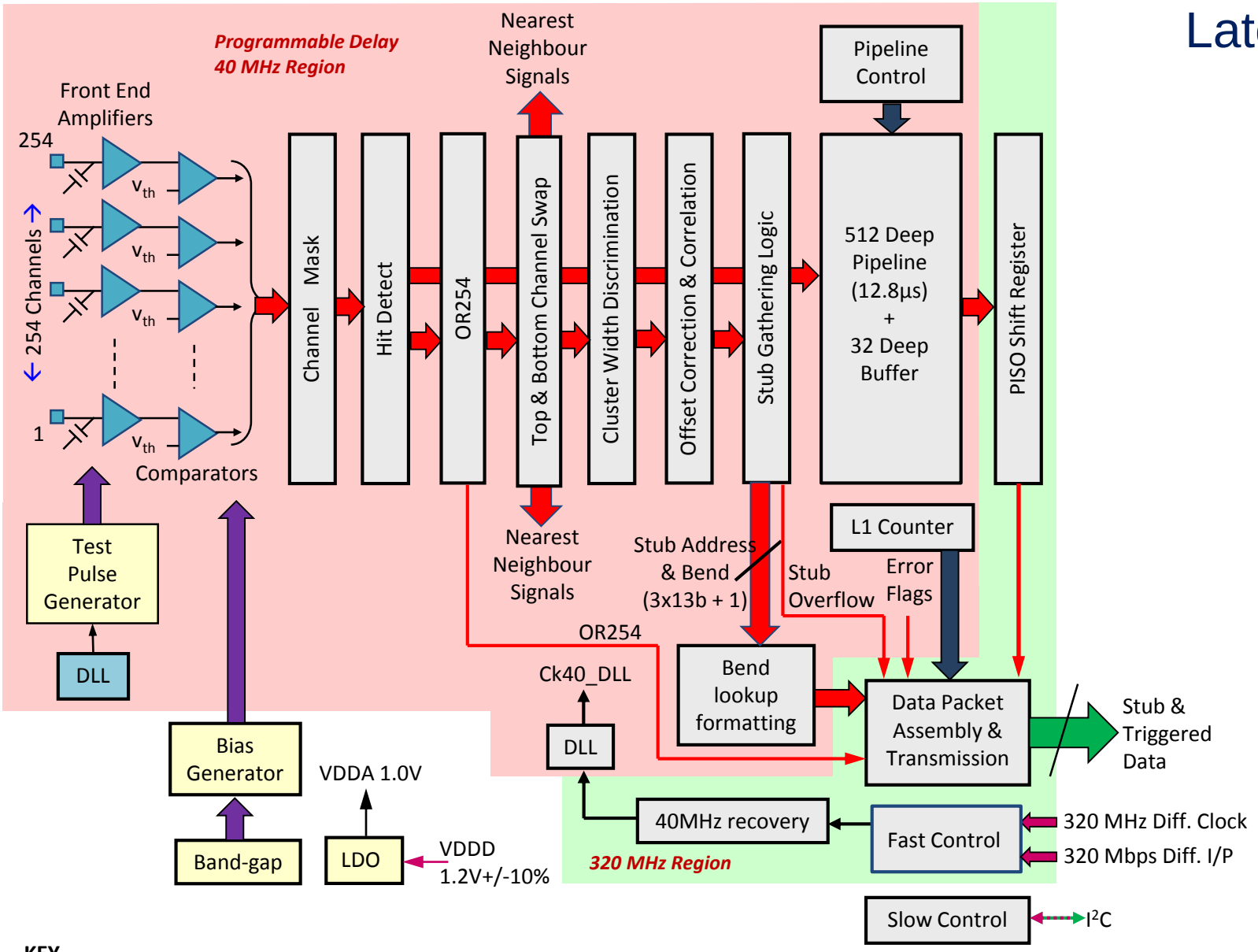
short term schedule on next page

CBC3 test plan outline (continues to evolve)



extra

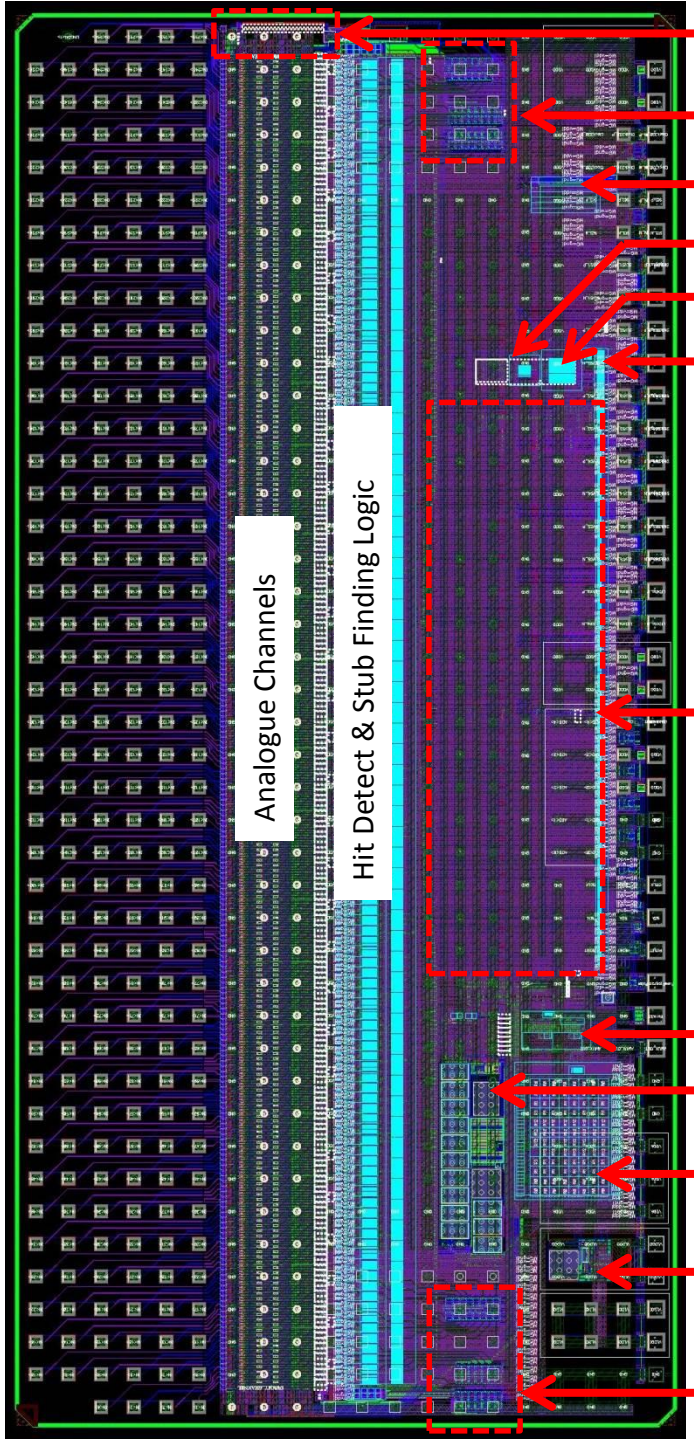
Latest Block Diagram



KEY

- Digital Data Path
- Analogue Signals
- Differential SLVS Output Bus
- Differential SLVS Input
- Bi-directional Slow Control

CBC3



test pulse circuit

I/O to neighbour

DLL

bend LUT

data assembly

L1 counter & FIFO

512 deep pipeline & O/P buffer

bandgap

I2C & biases

10b DAC for VCTH

LDO

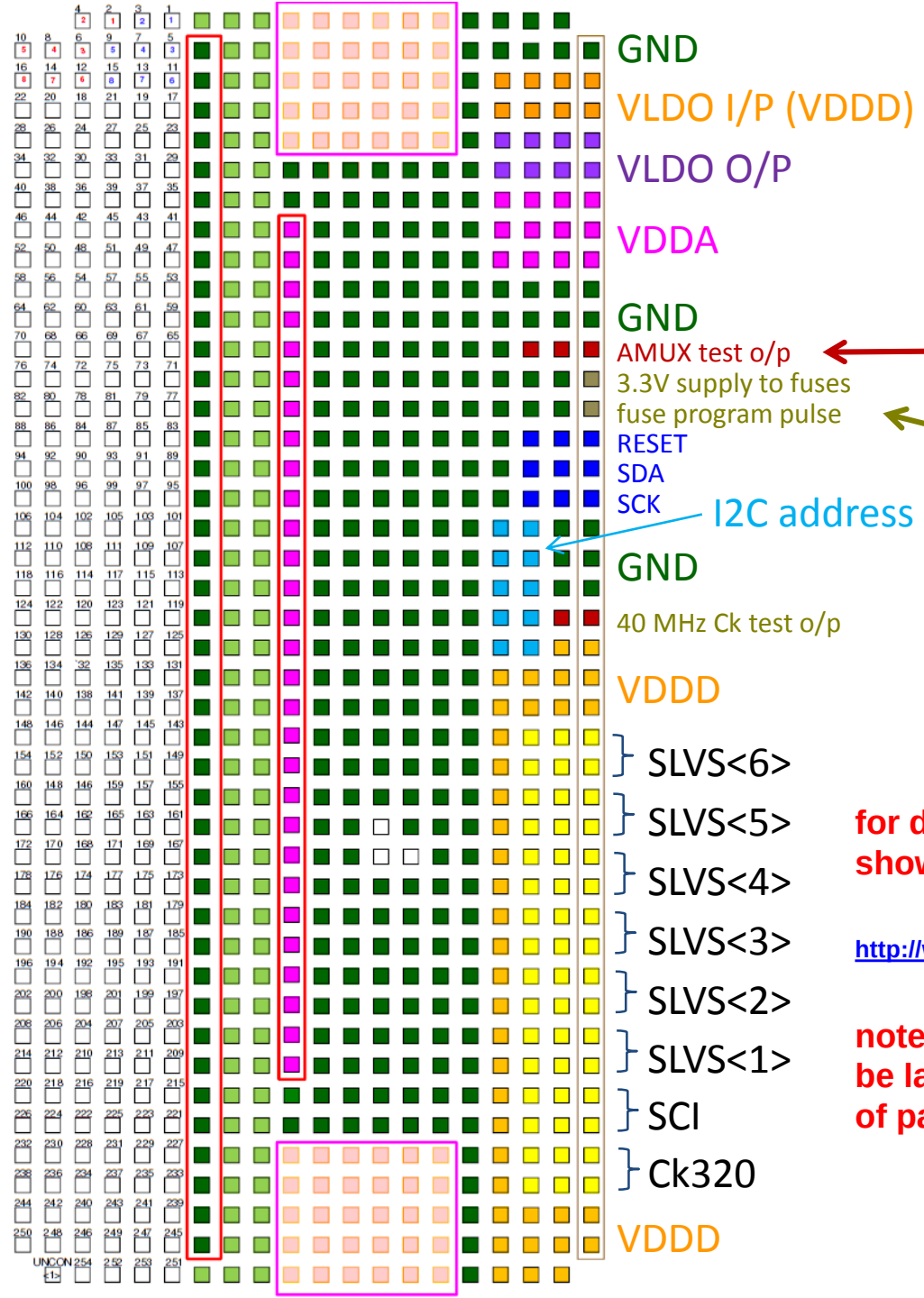
I/O to neighbour

final layout picture for reference

20 columns, 43 rows
(1 more column than CBC2)

5.25 mm x 11 mm

pad allocations



pads as viewed on hybrid surface
(as if looking through chip)

right-most column for wire-bond / wafer probe

like CBC2 gives access to internal
bias currents and voltages

chip ID can be set by e-fuses (19 bits)
will be programmed at wafer probe time
every chip will have unique ID

CERN PMOS bandgap reference
also trimmed by e-fuses (6 bits)

for detailed picture, prepared by Lawrence Jones, showing all pads labelled, download:

http://www.hep.ph.ic.ac.uk/~dmray/pictures/CBC3_PADS_Footprint1.png

note: downloadable picture shows pads as should be laid out on hybrid surface (flipped version of pads on chip)

CBC3 digital interfaces

output data: up to 3 stubs data transmitted to CIC/BX
6 SLVS diff pairs @ 320 Mbps

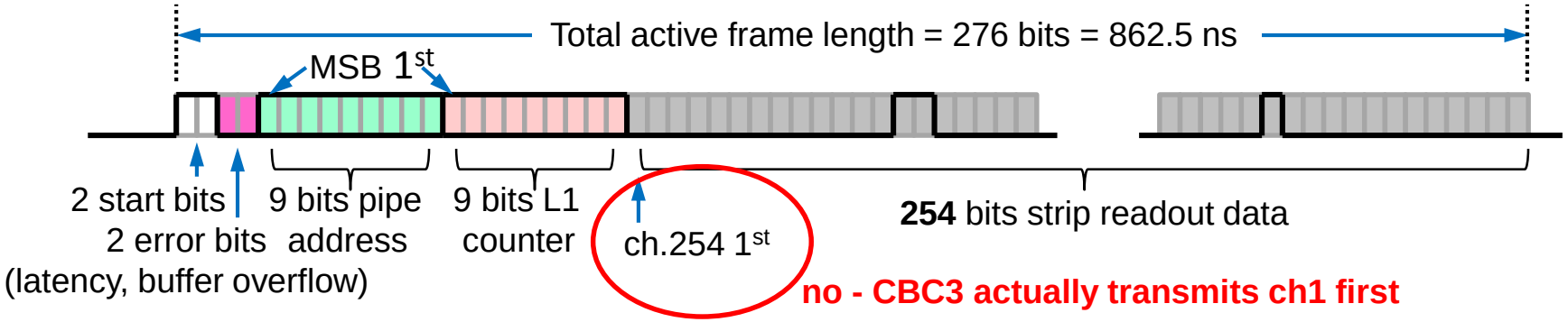
25 ns

S1<7>	S2<7>	S3<7>	B2<3>	Sync	R
S1<6>	S2<6>	S3<6>	B2<2>	Error	R
S1<5>	S2<5>	S3<5>	B2<1>	OR254	R
S1<4>	S2<4>	S3<4>	B2<0>	SoF	R
S1<3>	S2<3>	S3<3>	B1<3>	B3<3>	R
S1<2>	S2<2>	S3<2>	B1<2>	B3<2>	R
S1<1>	S2<1>	S3<1>	B1<1>	B3<1>	R
S1<0>	S2<0>	S3<0>	B1<0>	B3<0>	R

R = L1 triggered readout data
time flow **top** to **bottom** (e.g. S1<7> output first)

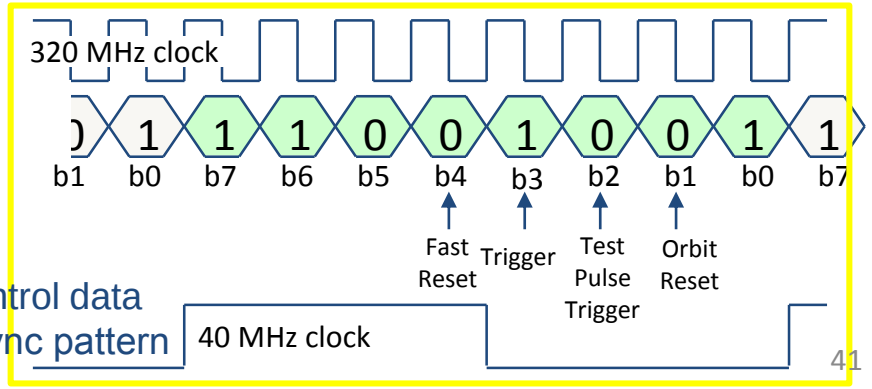
readout data

readout data frame length 950 nsec
=> up to 1 MHz L1 triggering capability



fast control

320 MHz clock
320 Mbps fast control line



40 MHz generated from fixed sync pattern in fast control data
normal command structure can't be confused with sync pattern