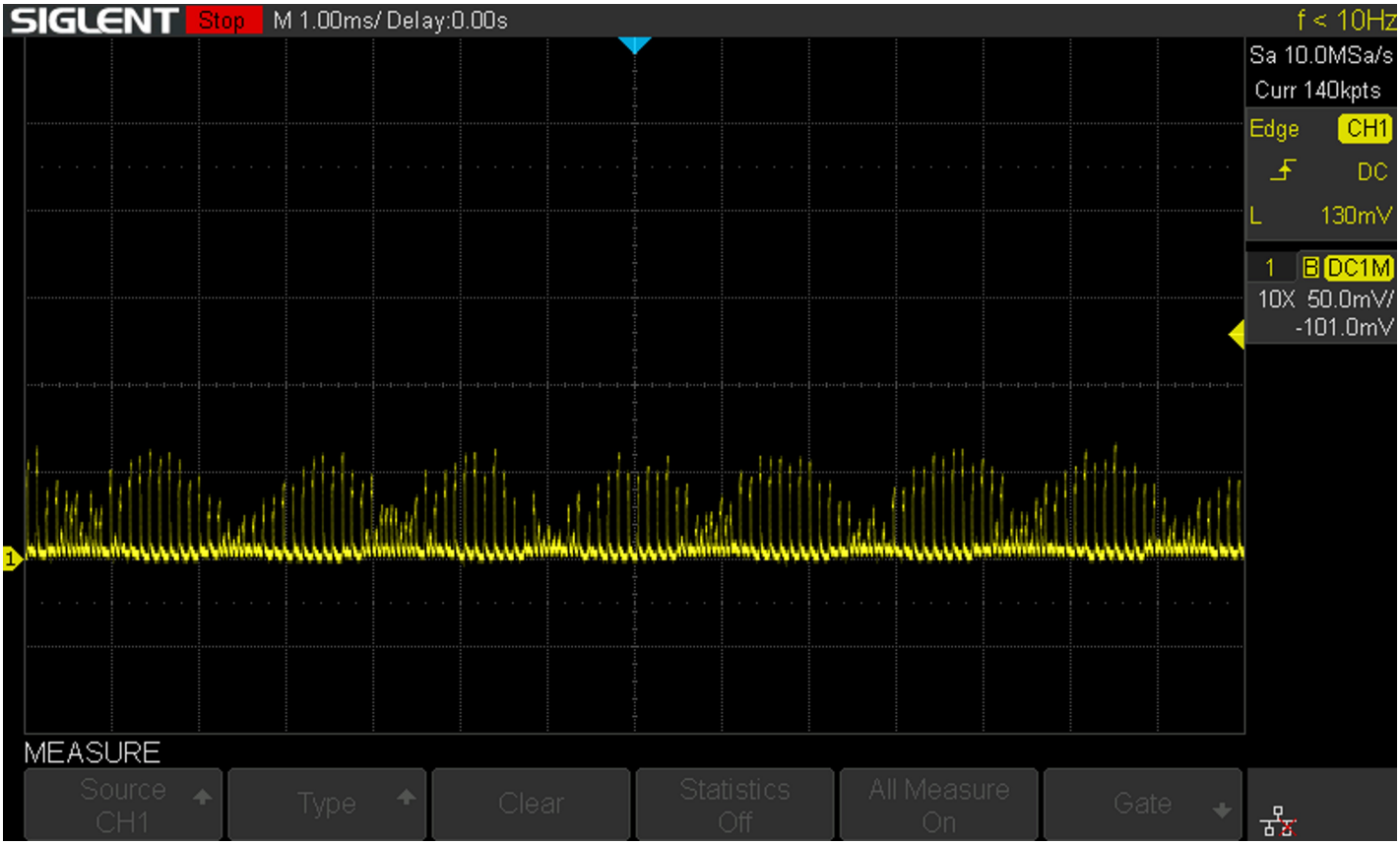
Testing V01

dinsdag 7 december 2021 09:28

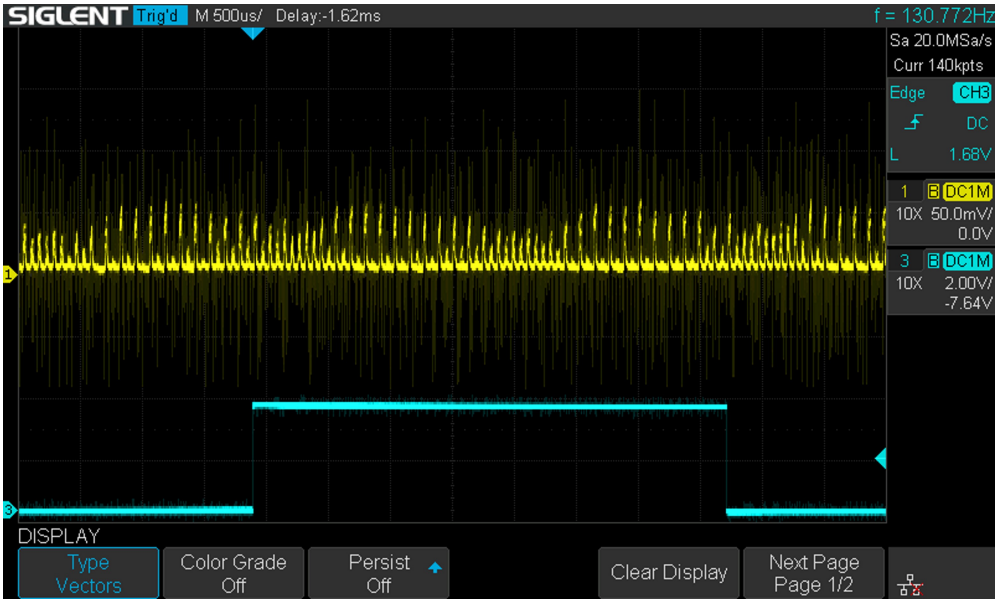
7-12-2021 09:28 Testing signal generation
Signal generation seems to work except the period signal. PA15 is by default a JTAG pin this causes problems and needs to be remapped propperly to work.

7-12-2021 09:30 Test signal reception and amplification.
#Touching the aluminium reflector pcb shows on the received signal, for now I switch back to FR4.
*Succes the expected signal is present, at the output but it is very small.



Posible explanations:
-Bias signal incorrect. (change phase of this signal)
-amplification to small analog signal to small, must be increased.
-PCB construction, to much ground capacitance with the receiver

7-12-2021 12:39 Test signal received when biased signal is shifted.
I fixed the timebase and period debug signals so they are outputted propperly and I can trigger the scope on the full signal period.



No shift

```
/* Calculate TX signals LUT location */
static uint32_t tx1Loc = 0;
uint32_t tx2Loc = calcBuffLoc(tx1Loc, TX_BUFF_SIZE / 8 * 1, TX_BUFF_SIZE);
uint32_t tx3Loc = calcBuffLoc(tx1Loc, TX_BUFF_SIZE / 8 * 2, TX_BUFF_SIZE);
uint32_t tx4Loc = calcBuffLoc(tx1Loc, TX_BUFF_SIZE / 8 * 3, TX_BUFF_SIZE);
uint32_t tx5Loc = calcBuffLoc(tx1Loc, TX_BUFF_SIZE / 8 * 4, TX_BUFF_SIZE);
uint32_t tx6Loc = calcBuffLoc(tx1Loc, TX_BUFF_SIZE / 8 * 5, TX_BUFF_SIZE);
uint32_t tx7Loc = calcBuffLoc(tx1Loc, TX_BUFF_SIZE / 8 * 6, TX_BUFF_SIZE);
uint32_t tx8Loc = calcBuffLoc(tx1Loc, TX_BUFF_SIZE / 8 * 7, TX_BUFF_SIZE);

// uint32_t biasLoc = tx1Loc;
uint32_t biasLoc = calcBuffLoc(tx1Loc, 0, TX_BUFF_SIZE);
```

Shifting the bias signal does not yield better results.
I shifted from 0 to 8 (bias period). No shift had the best result, so the bias is correct.

7-12-2021 13:26 Fix jitter first write full port register in one go.

- [x] *I believe pins that are mapped using Alternative function like SWD or UART are not affected when writing the OCTL register. Correct see images below, when an IO is configured in AF-mode the OCTL register is not connected to the output control

7.5.4. Port output control register (GPIOx_OCTL, x=A..G)

Address offset: 0x0C
Reset value: 0x0000 0000

This register has to be accessed by word(32-bit).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

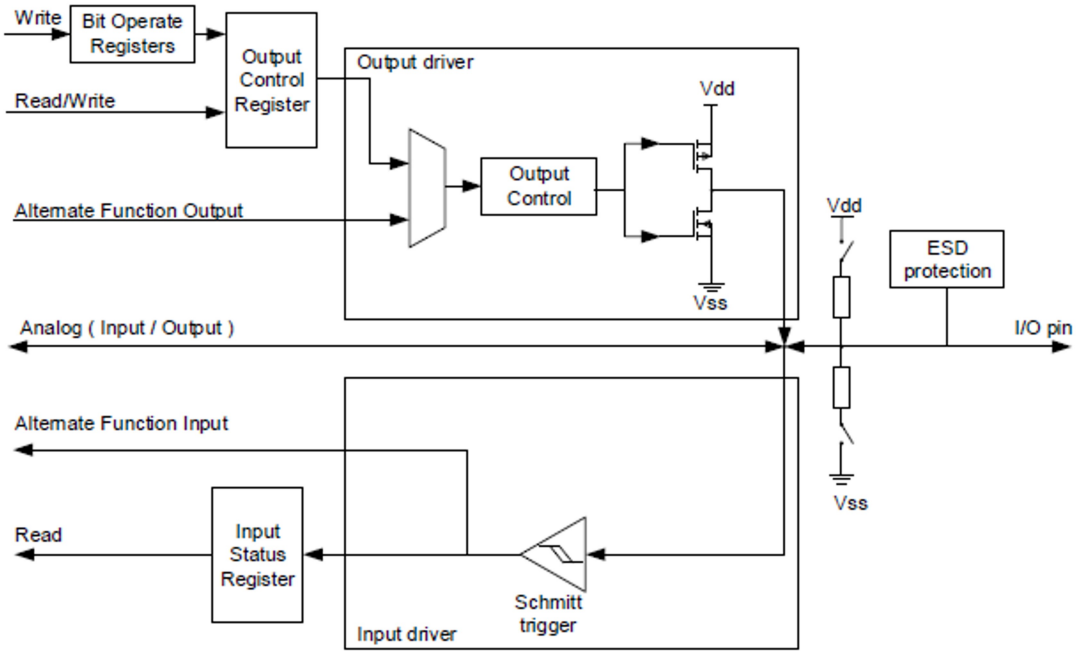
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OCTL15	OCTL14	OCTL13	OCTL12	OCTL11	OCTL10	OCTL9	OCTL8	OCTL7	OCTL6	OCTL5	OCTL4	OCTL3	OCTL2	OCTL1	OCTL0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits	Fields	Descriptions
31:16	Reserved	Must be kept at reset value
15:0	OCTLy	Port output control(y=0..15) These bits are set and cleared by software 0: Pin output low 1: Pin output high

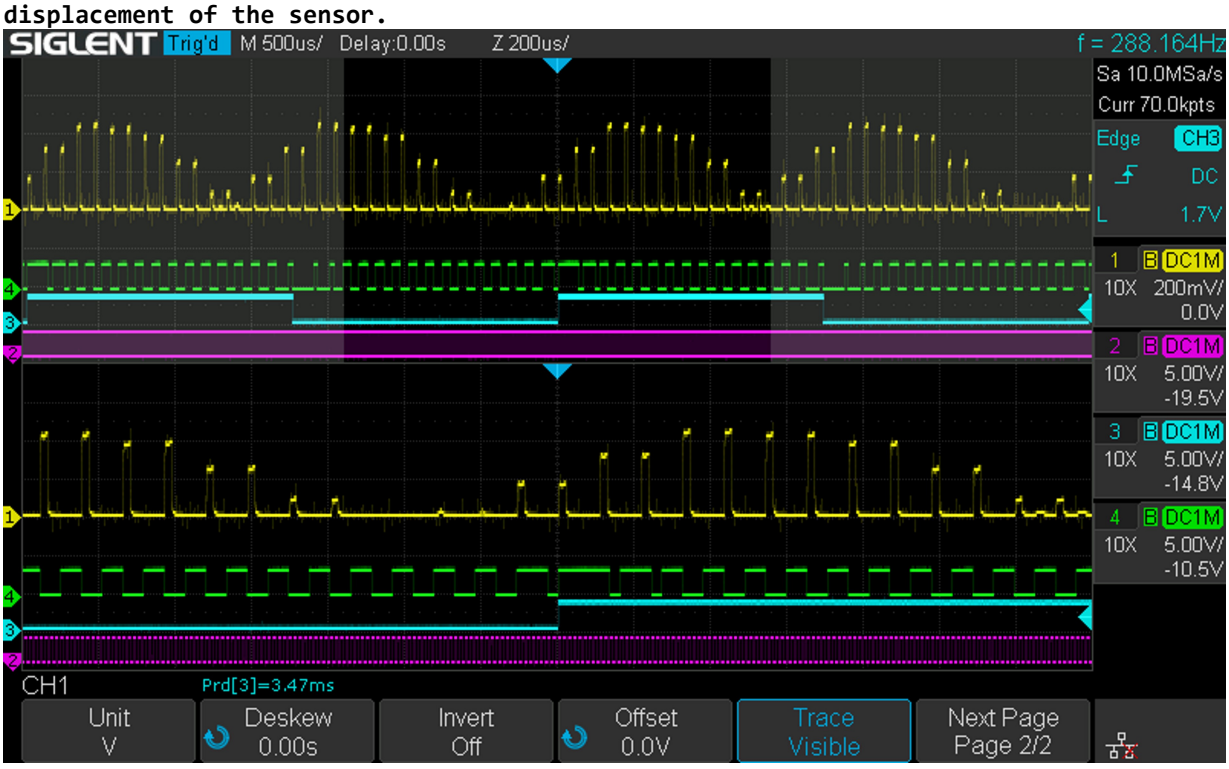
Table 7-1. GPIO configuration table

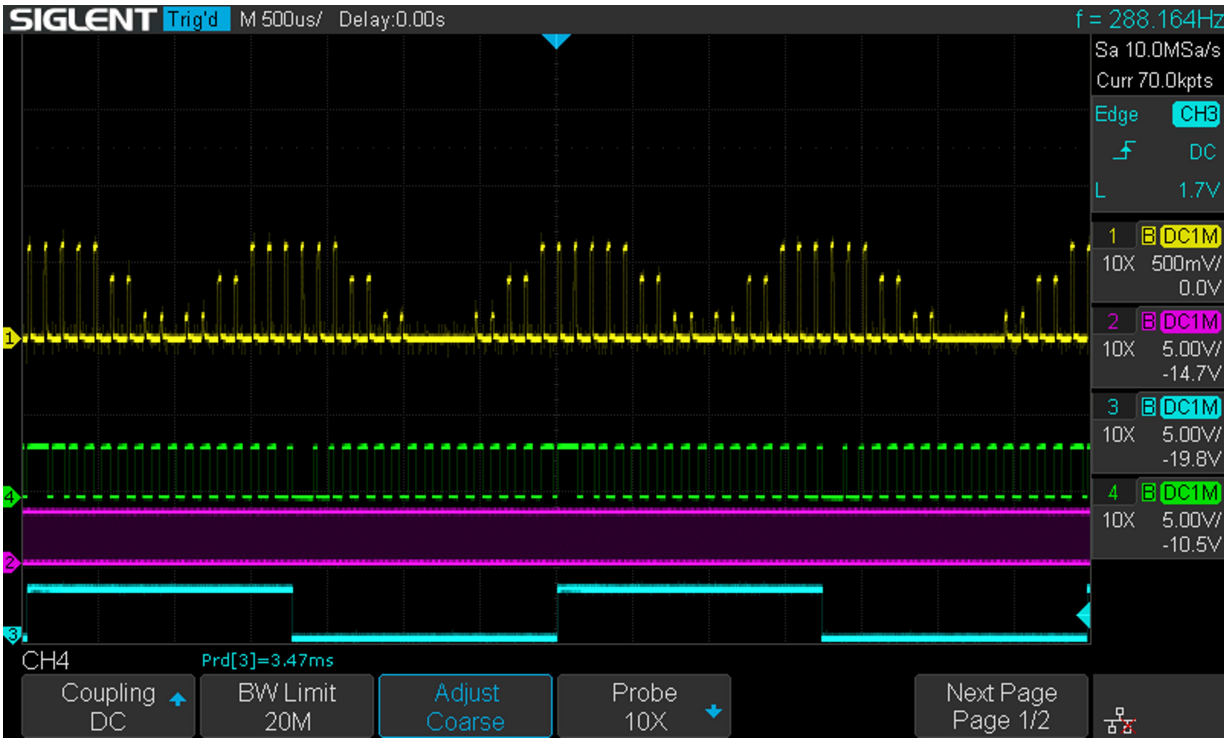
Configuration mode		CTL[1:0]	MD[1:0]	OCTL
Input	Analog	00	0	don't care
	Input floating	01		don't care
	Input pull-down	10		0
	Input pull-up	10		1
General purpose Output (GPIO)	Push-pull	00	00: Reserved 01: Speed up to 10MHz 10: Speed up to 2MHz 11: Speed up to 50MHz	0 or 1
	Open-drain	01		0 or 1
Alternate Function Output (AFIO)	Push-pull	10		don't care
	Open-drain	11		don't care

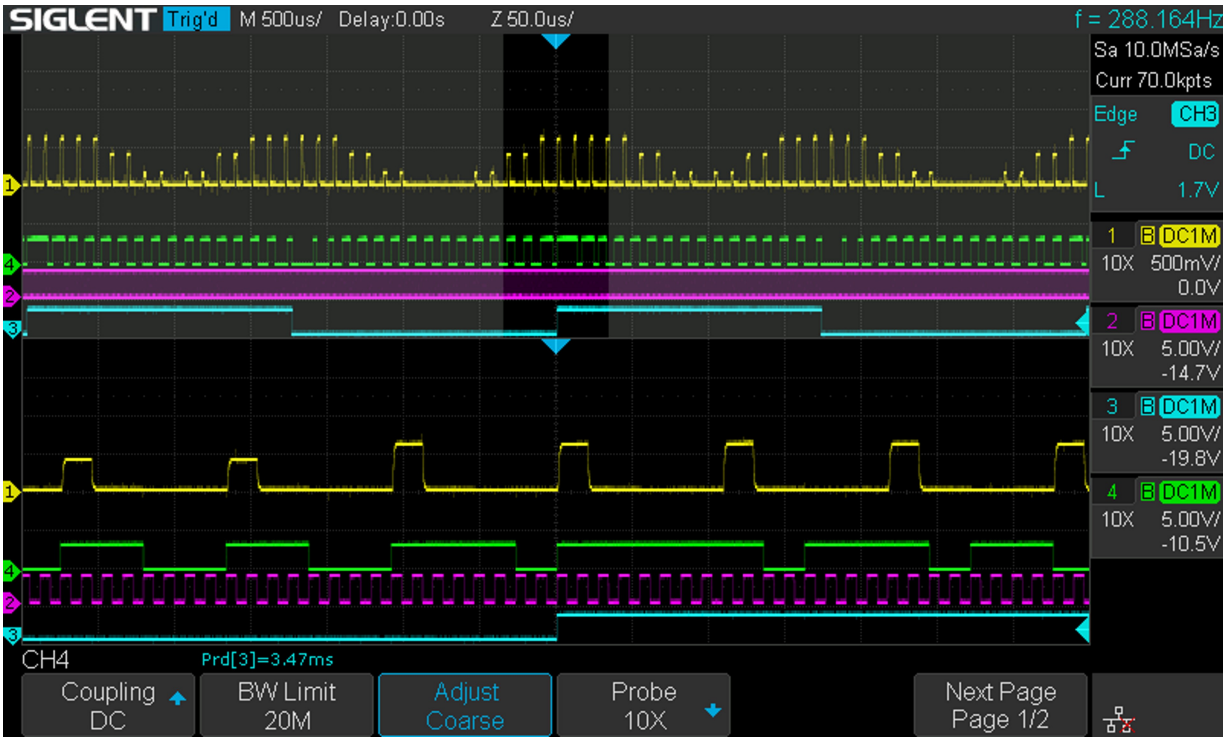
Figure 7-1. Basic structure of a standard I/O port bit

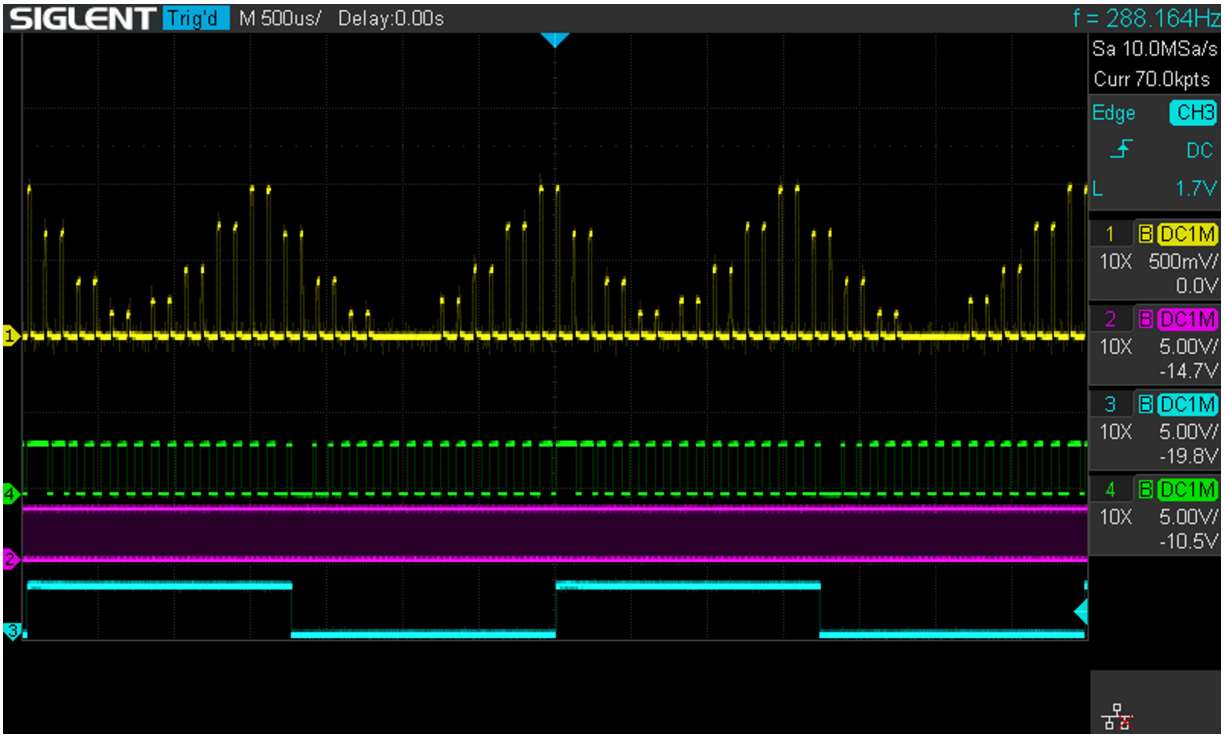


★ **7-12-2021 15:10 Test if phase of sensor signal changes with the displacement of the sensor.**




SIGLENT
Trig'd
M 500us/ Delay:0.00s
f = 288.164Hz
Sa 10.0MSa/s
Curr 70.0kpts
Edge CH3
DC
L 1.7V
1 B DC1M
10X 500mV/
0.0V
2 B DC1M
10X 5.00V/
-14.7V
3 B DC1M
10X 5.00V/
-19.8V
4 B DC1M
10X 5.00V/
-10.5V
CH4
Prd[3]=3.47ms
Coupling
DC
BW Limit
20M
Adjust
Coarse
Probe
10X
Next Page
Page 1/2

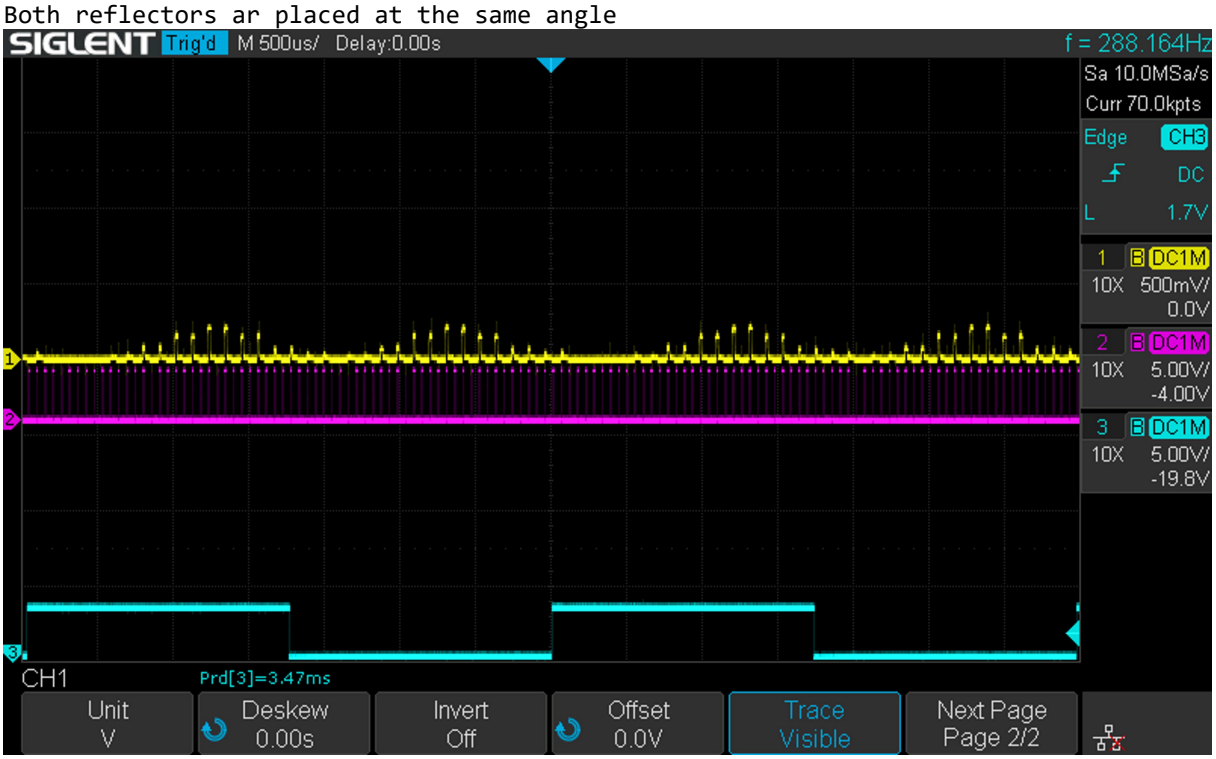

SIGLENT
Trig'd
M 500us/ Delay:0.00s
Z 50.0us/
f = 288.164Hz
Sa 10.0MSa/s
Curr 70.0kpts
Edge CH3
DC
L 1.7V
1 B DC1M
10X 500mV/
0.0V
2 B DC1M
10X 5.00V/
-14.7V
3 B DC1M
10X 5.00V/
-19.8V
4 B DC1M
10X 5.00V/
-10.5V
CH4
Prd[3]=3.47ms
Coupling
DC
BW Limit
20M
Adjust
Coarse
Probe
10X
Next Page
Page 1/2



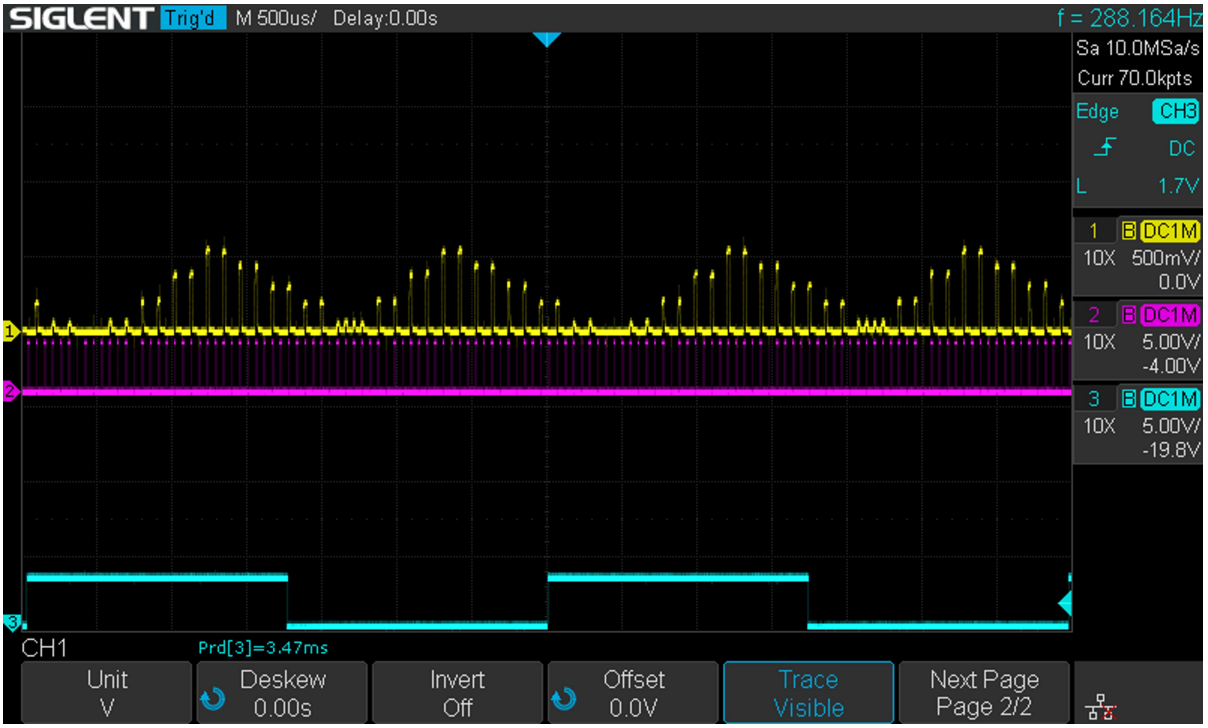
Yellow	RX sensor after amplification
Green	TX1 (SPWM signal)
Blue	(SPWM signal period = 512T)
Pink	Timebase (T)

It works!

7-12-2021 17:11 Comparison of Aluminium PCB & FR4
Both reflectors ar placed at the same angle



The aluminium pcb



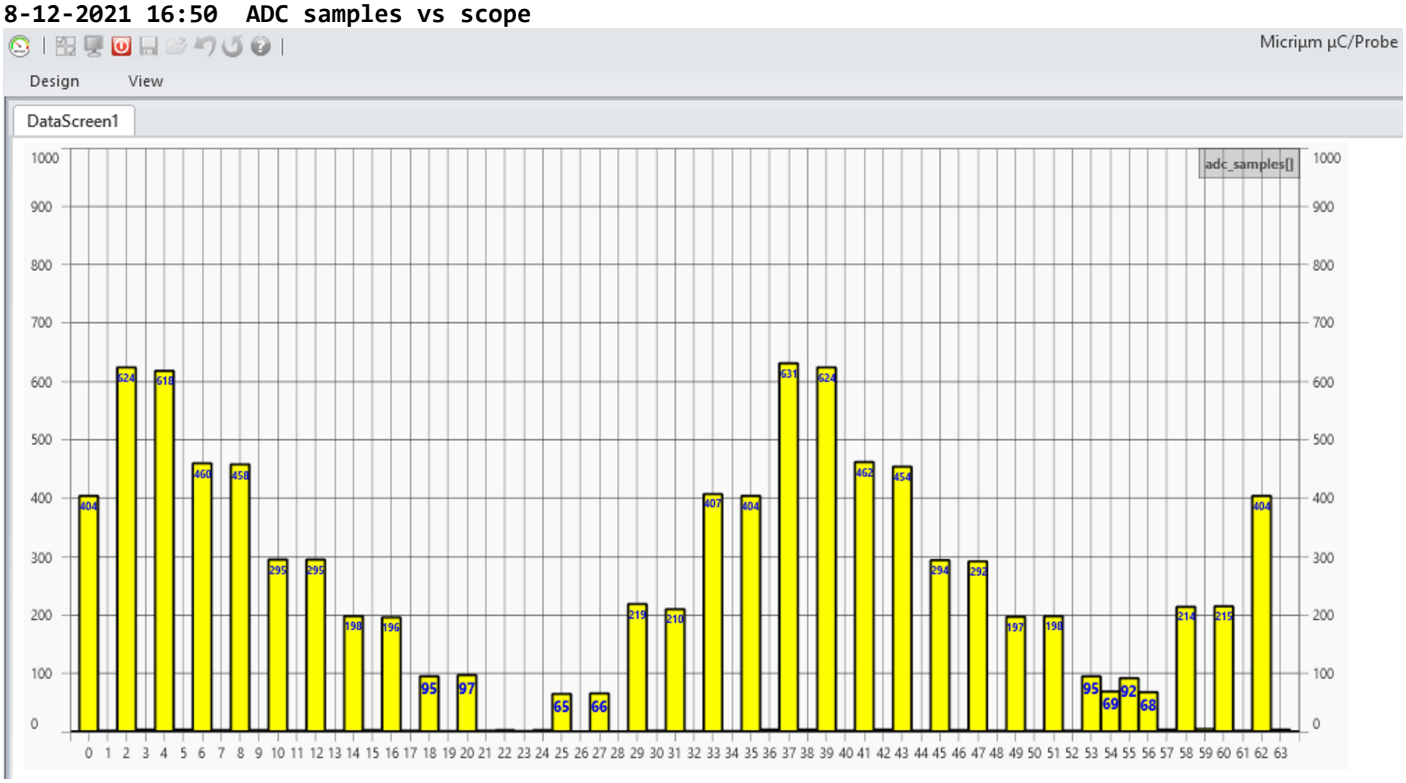
FR4 pcb

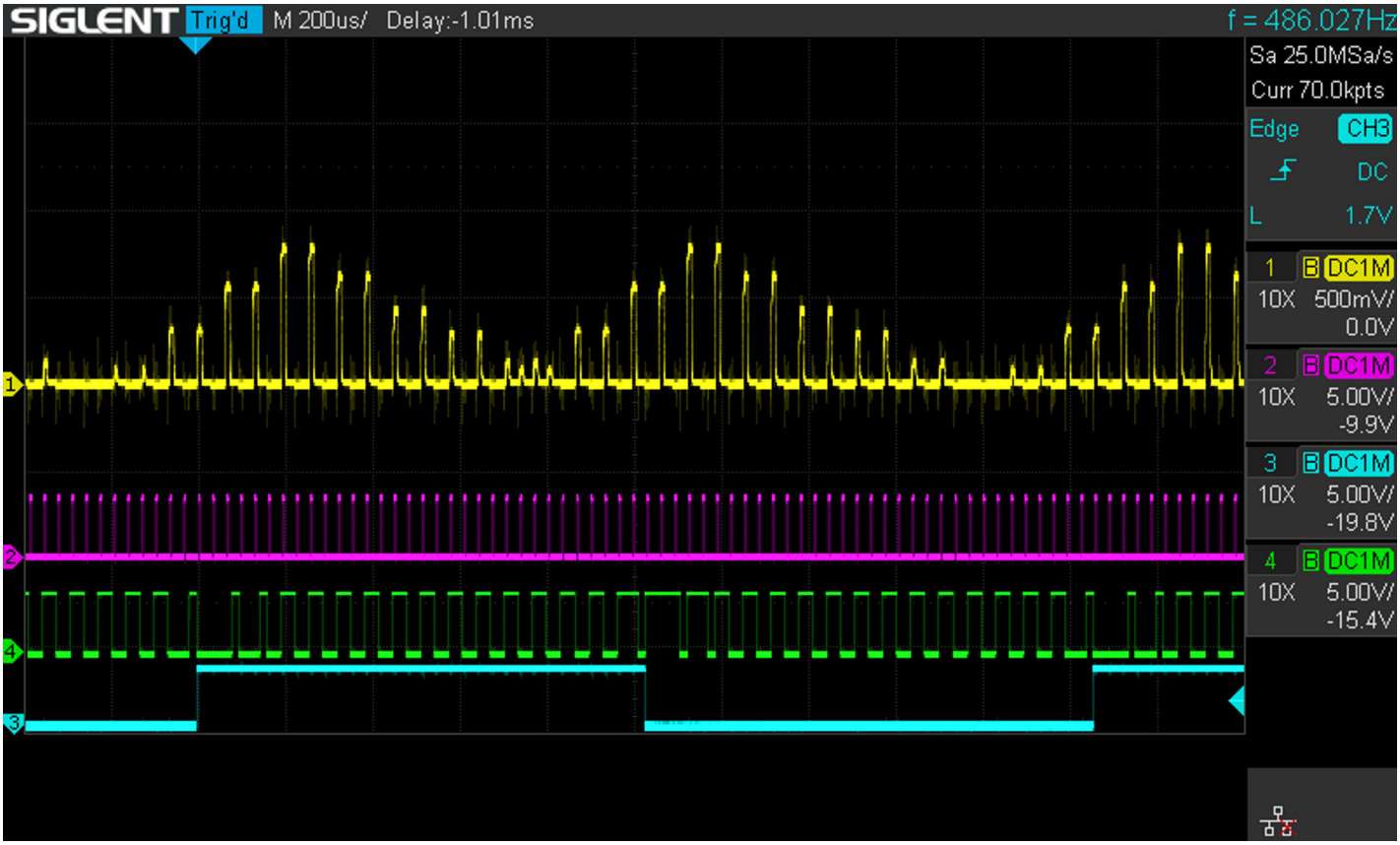
There is a clear difference between the PCB's in the amplitude of the received signal.
This could be explained by the distance between the copper layer on top and the conductive aluminum layer below. A bigger part of the RX signal is coupled to ground using the aluminum PCB.

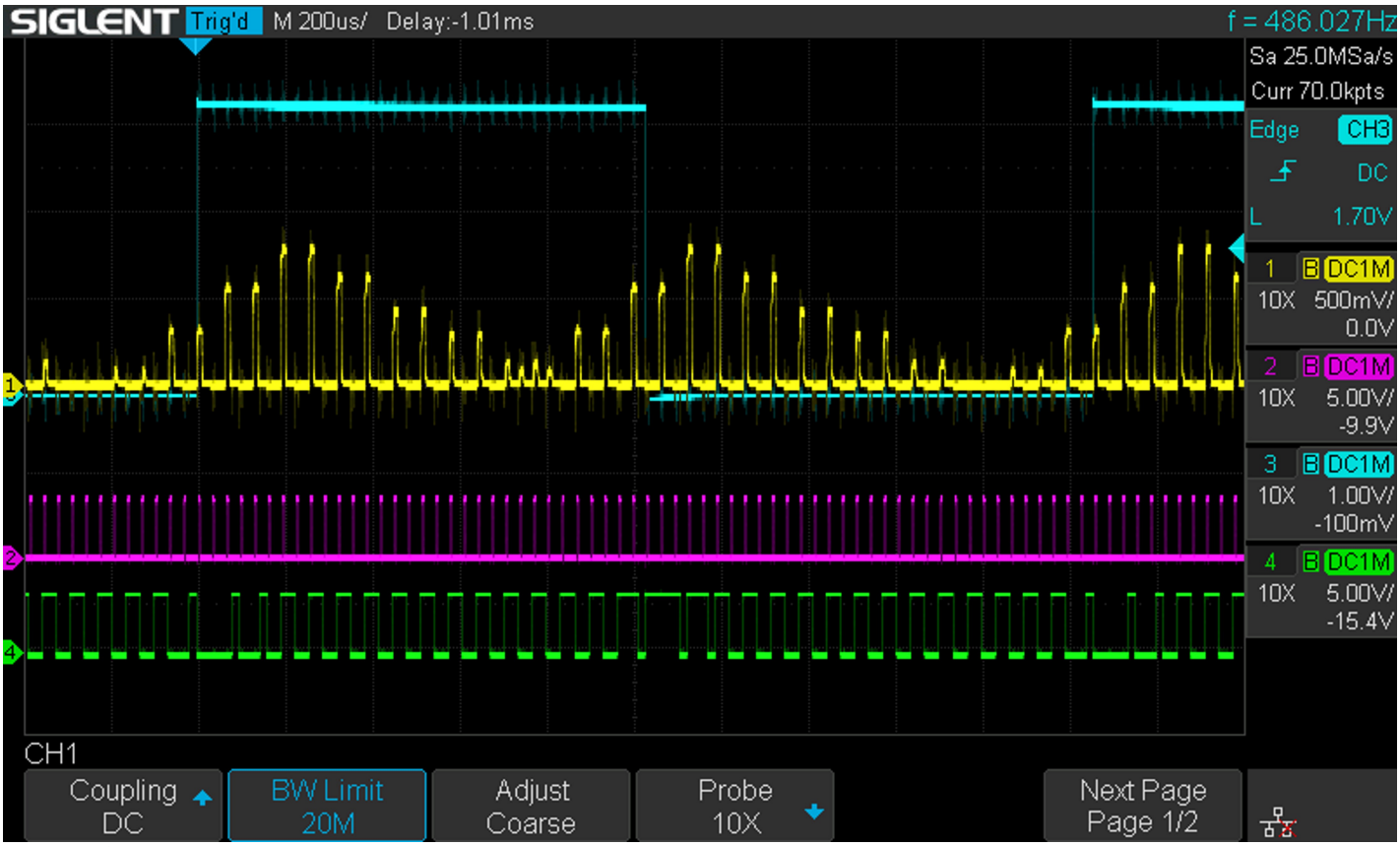
8-12-2021 00:35 Does signal output freq influence amplitude?

Quick test timer PCS from 365 to 200 does not have a measureable amplitude change
Changed to 2*108 yields ~2mS period for the signal, like the caliper uses.

8-12-2021 16:50 ADC samples vs scope



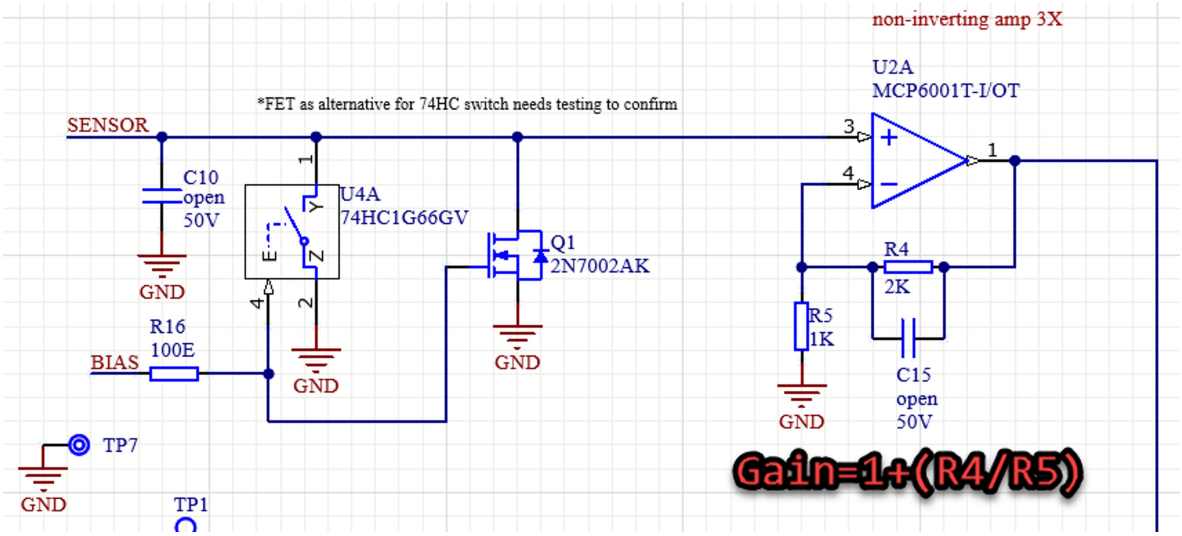


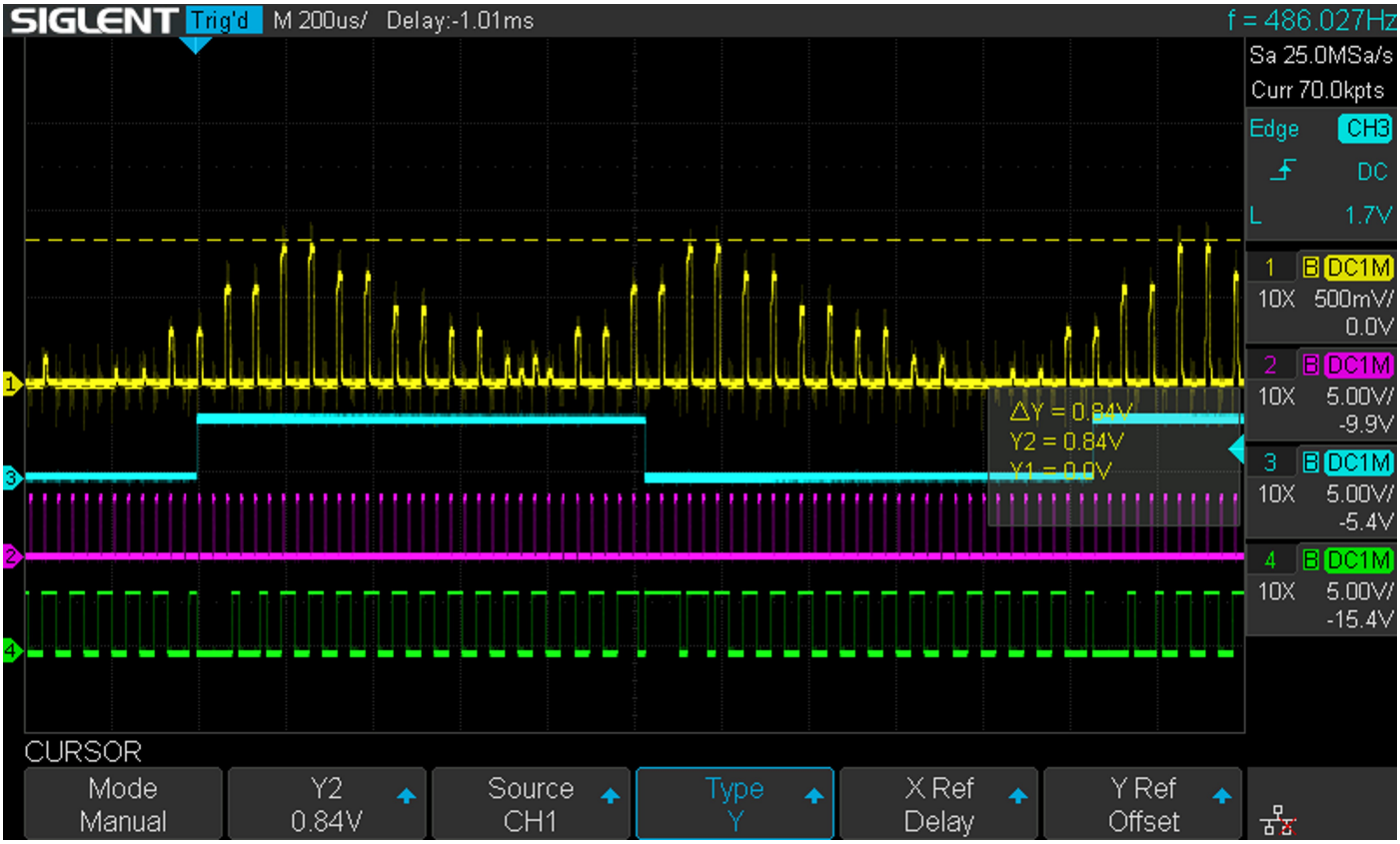


Signal is sampled correctly.

8-12-2021 17:01 Increasing the gain

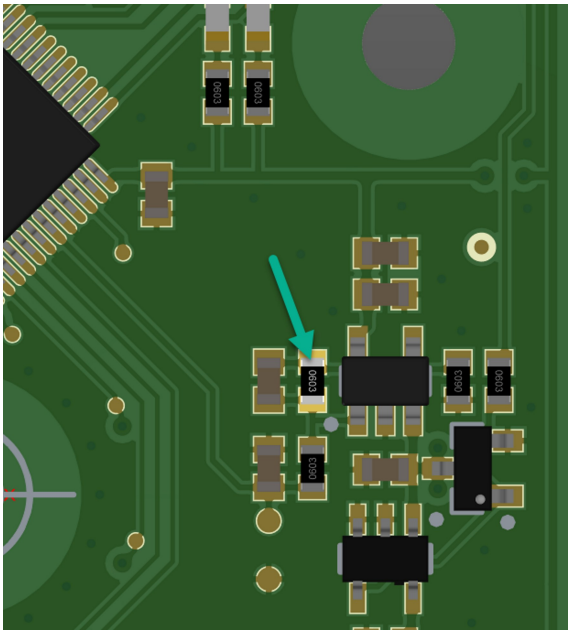
The full range of the ADC is not used, thus the sensor voltage can be increased. The V01 has a gain of 3 for the analog frontend. This results in a maximum voltage on the adc pin of 0.85V



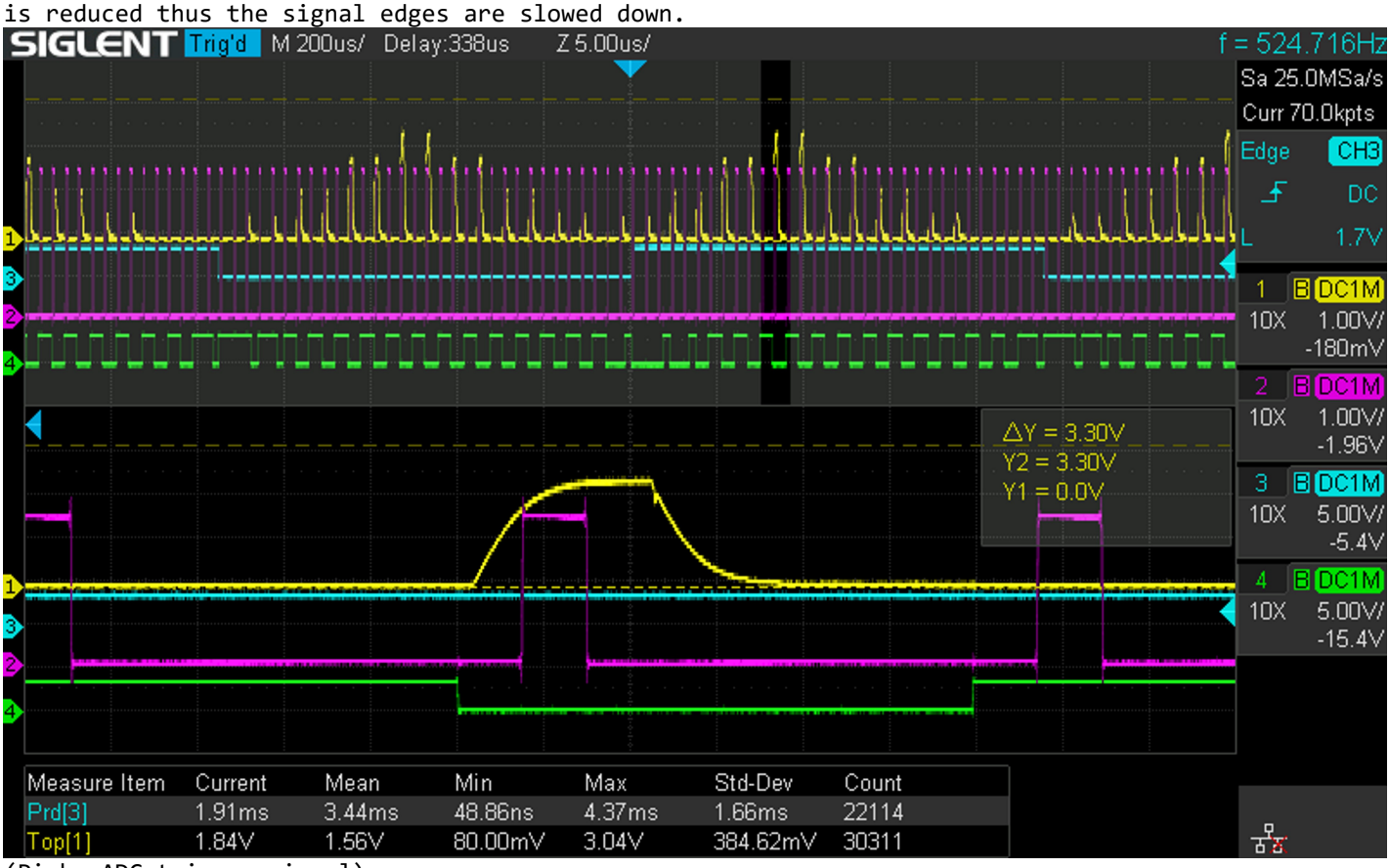


0.85/3=0.2833
3.3/0.2833=11.6484 maximum gain value for full range.

To get a gain of 11 R4 can be replaced with a 10K resistor.
This results in a signal gain of:
★ 1+10/1=11



The amplitude of the received signal now better fits within the ADC range.
However there is a new problem now, because of the reduced amplifier
bandwidth (gain bandwidth product) the maximum freq that can be amplified
is reduced thus the signal edges are slowed down.

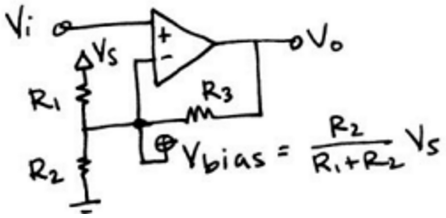


(Pink= ADC trigger signal)

```
timer_initpara.prescaler = 200;
```

8-12-2021 17:18 What if the opamp is biased above ground?

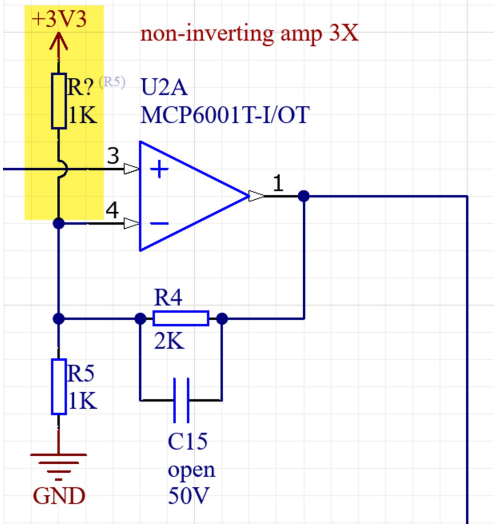
Split Resistor Biasing: This method is the simplest to construct, but has the greatest amount of noise, as capacitors can not be used. Basically, any resistor which used to go to ground, is now split into two resistors; one which goes to ground, and one which goes to the power supply rail. The ratio of R1 and R2 determine the bias voltage, and as long as R1//R2 is equal to the old resistor value, the circuit will operate the same way. This method has the drawback, that if multiple bias voltages are required, it will be very difficult to get the same voltages at each point. So, if there is DC coupling, you can very quickly begin to amplify this error in bias voltages.



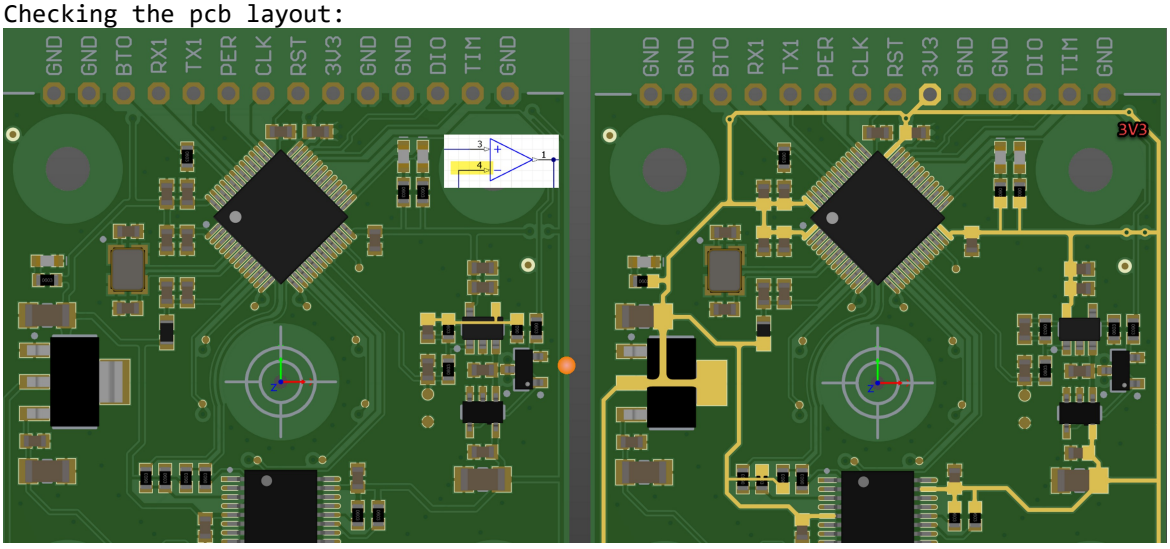
HOW TO BIAS AN O...

Link

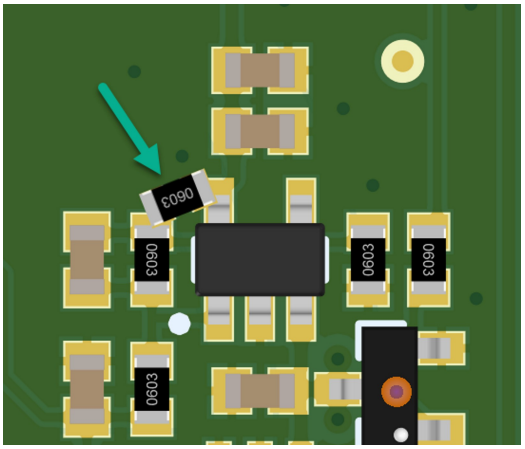
Adding a 1K resistor to the circuit, for this test:



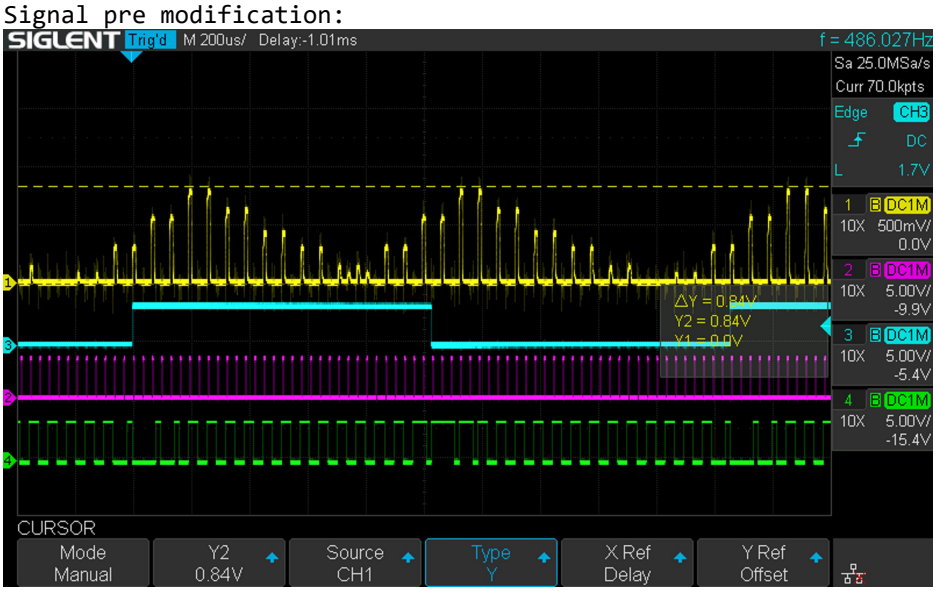
Checking the pcb layout:



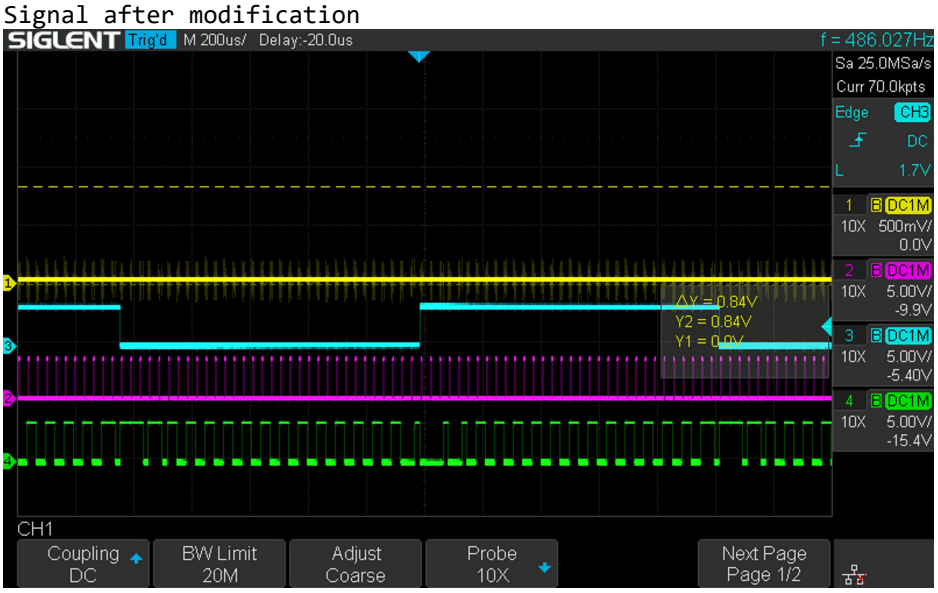
Modification:



Signal pre modification:



Signal after modification



Result: FAIL!, signal disappears after this mod. The bpard is restored.

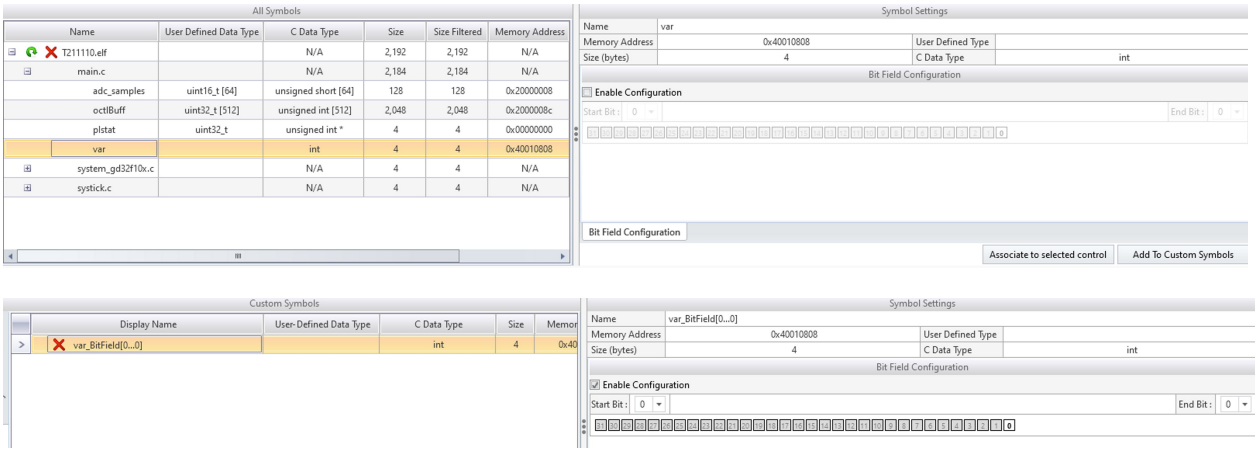
9-12-2021 09:52 Debugging using "Micrium uC-Probe"
It would be great if it were possible to plot variables like the sample buffer holding the sampled signal inside the MCU on the PC for debugging.

Printing this buffer over the serial port and plotting them is a posible solution but the big disadvantage is this method is that addition code has to be introduced on the target stealing CPU cycles needed for serial printing the buffer variables.

As it turns out there is an other way, your debugger can access to the device memory while the target is running using the debug access port without stealing cpu cycles from the target. (arm documentation). This is amazing and I didn't know about it until I was downloading all stm32 available development tools and found stm-studio-stm32. (STM Studio is replaced by stm32cubemonitor)
I stuck gold it was like magic. Why did I not know about this before, why did nobody tell me this existed? And are their other better tools available? As it turns out there are and I discovered "Micrium uC-Probe" that was a payed tool in the past but now is available for free. (sadly the official website is no longer available after silabs acquired Micrium, however I found a download link on a chinese forum and got the latest installation file)

"
µC/Probe is a Windows application designed to read and write the memory of any
embedded target processor during run-time. Memory locations are mapped to a set of
virtual controls and indicators placed on a dashboard.
"
Ucprobe should support importing SVD files to show the peripherals, however I'm not able to make this work.
As workaround a map a variable to a specific memory adress iside the project. (see keil doc for reference)

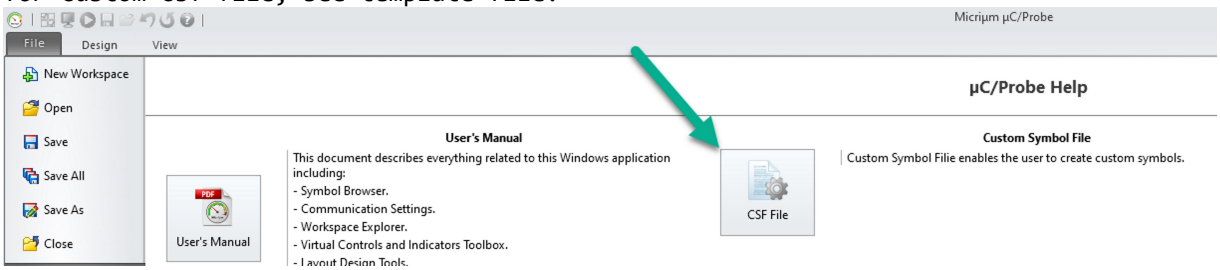
```
int var __attribute__((section(".ARM.__at_0x40010808"))); // GPIO input status reg
```



It works but is not great, it breaks debugging in keil.

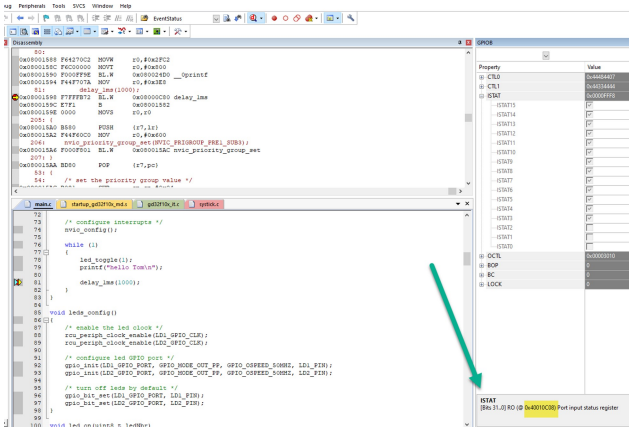
Default location Gigadevice Device Firmware Pack:	C:\Users\Admin\AppData\Local\Arm\Packs\GigaDevice\GD32F10x_DFP\2.0.2\SVD

~~Alternative method without modifing target code: look in manual ucprobe for custom csf file, see template file:~~



~~Loading the CSF file seem to work.~~

~~Test creating a CSF file for the board leds.~~
~~Memory address can be found using keil debugger:~~

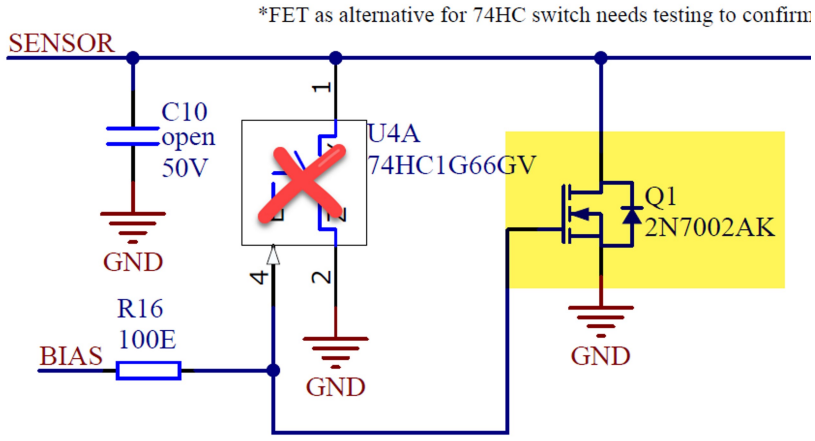


~~PB12~~	~~LED2~~
~~PB13~~	~~LED1~~

~~Mem add: 0x40010C08~~

I give up, it does not seem possible to plot a custom memmory address with a CSF file

13-12-2021 22:32 Test FET as alternative for analog switch



I assembled a board without the analog switch and with the FET.
There is no signal present. So this does not work. However this board was not tested before so it could be something else but I'm almost certain the FET does not work.