100MSps oscilloscope with RP2040

En este artículo mostraré el diseño de un osciloscopio basado en el microcontrolador RP2040 de Raspberry. Este osciloscopio utiliza un ADC08100 que es un conversor de 8bits capaz de funcionar a una freq máxima de 100MHz. La caracteristica principal de este diseño es poder utilizar el propio microcontrolador para controlar el ADC, gracias a la potencia de un periferico especial del microcontrolador llamado PIO.

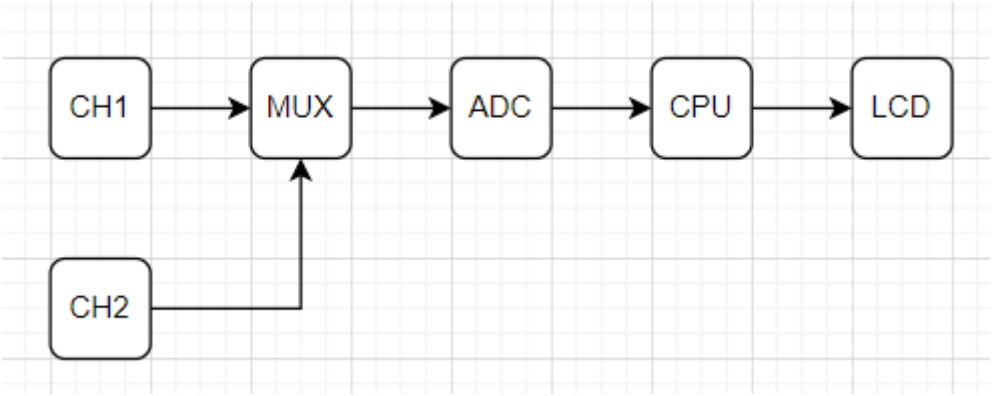
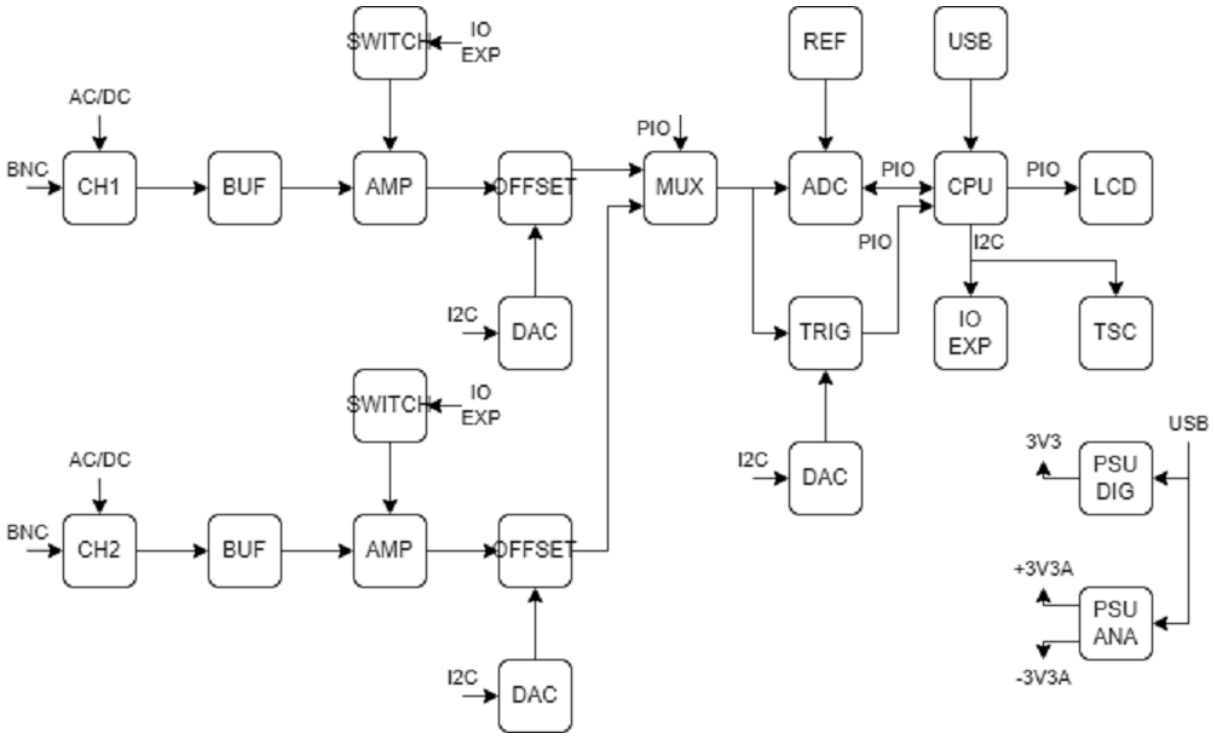
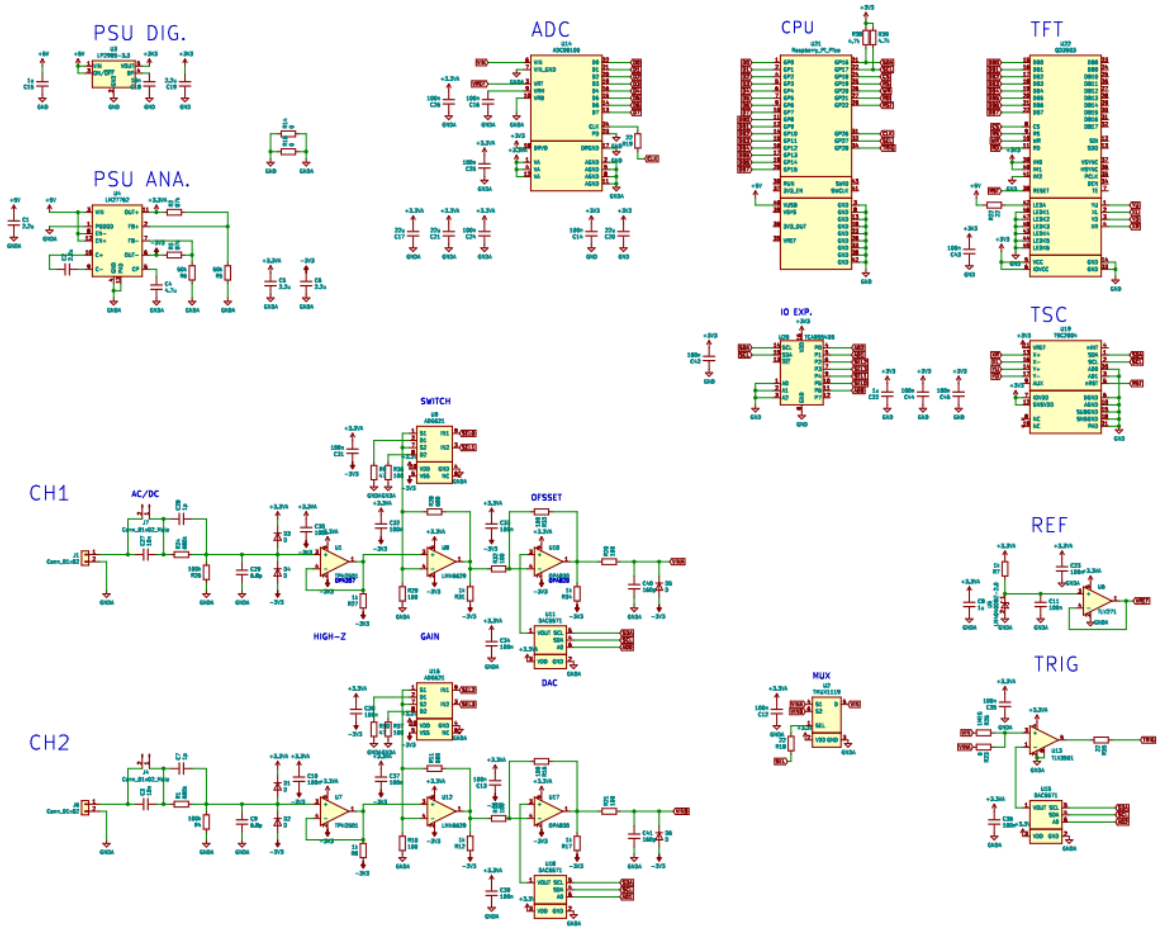
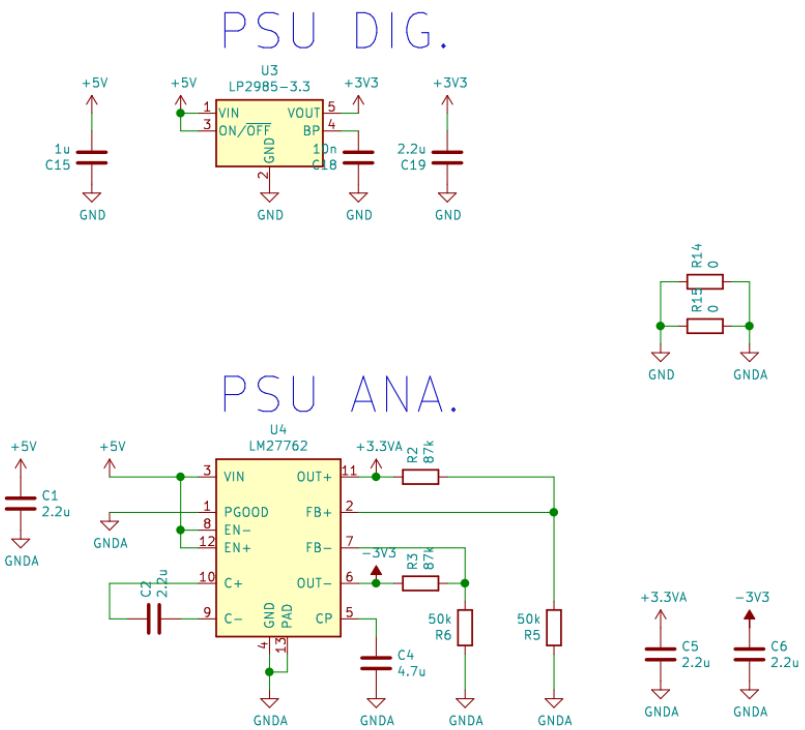
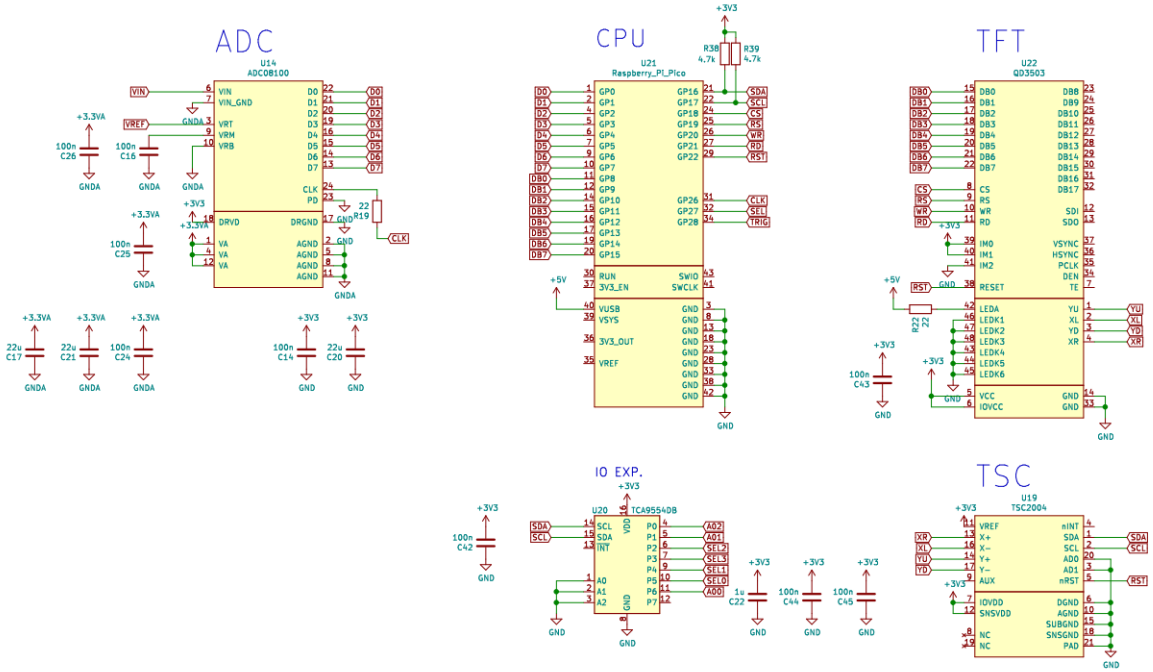


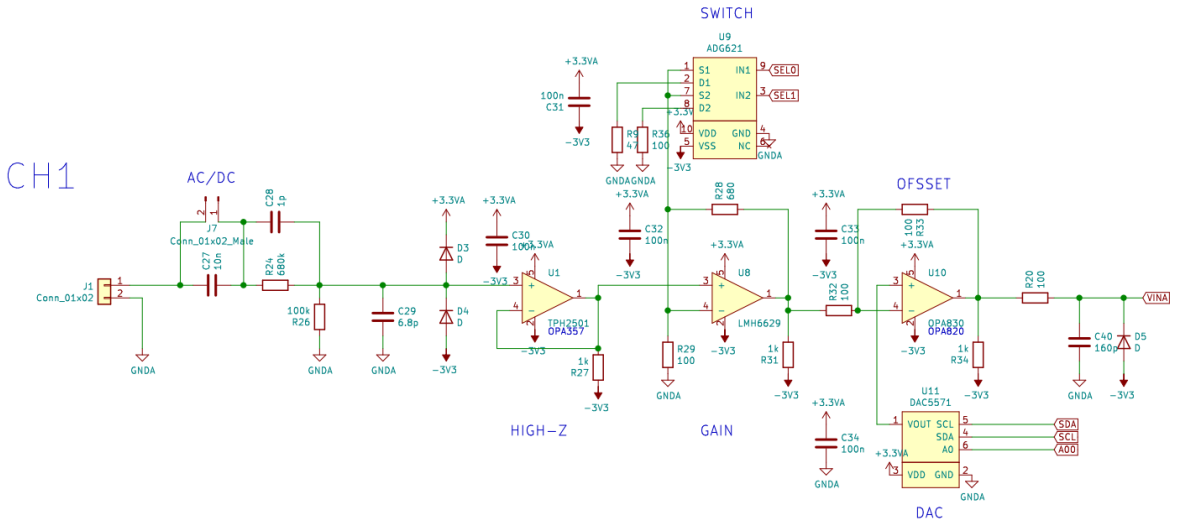
Diagrama general

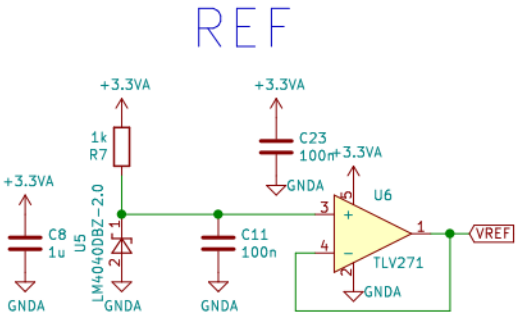

SWITCH
IO EXP
REF
USB
AC/DC
PIO
BNC
CH1
BUF
AMP
OFFSET
MUX
ADC
PIO
CPU
PIO
LCD
PIO
I2C
I2C
DAC
TRIG
IO EXP
TSC
SWITCH
IO EXP
AC/DC
I2C
DAC
USB
3V3
PSU DIG
BNC
CH2
BUF
AMP
OFFSET
+3V3A
PSU ANA
-3V3A
I2C
DAC

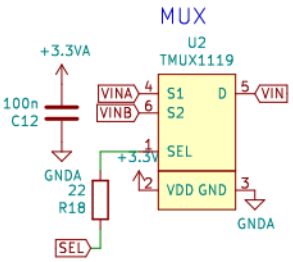

PSU DIG.
ADC
CPU
TFT
PSU ANA.
IO EXP.
TSC
SWITCH
CH1
AC/DC
OFFSET
REF
HIGH-Z
GAIN
DAC
MUX
TRIG
CH2

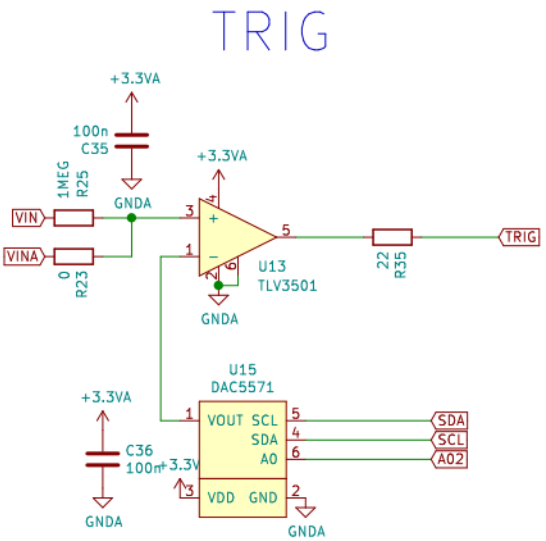

PSU DIG.
U3
LP2985-3.3
+5V
+5V
+3V3
+3V3
VIN
ON/OFF
GND
VOUT
BP
1u
C15
10n
C18
2.2u
C19
GND
GND
GND
GND
R14
0
R15
0
GND
GNDA
PSU ANA.
U4
LM27762
+5V
+5V
+3.3VA
VIN
PGOOD
EN−
EN+
C+
C−
GND
PAD
OUT+
FB+
FB−
OUT−
CP
C1
2.2u
C2
2.2u
R2
87k
R3
87k
50k
R6
50k
R5
C4
4.7u
−3V3
GNDA
GNDA
GNDA
GNDA
GNDA
+3.3VA
−3V3
C5
2.2u
C6
2.2u
GNDA
GNDA

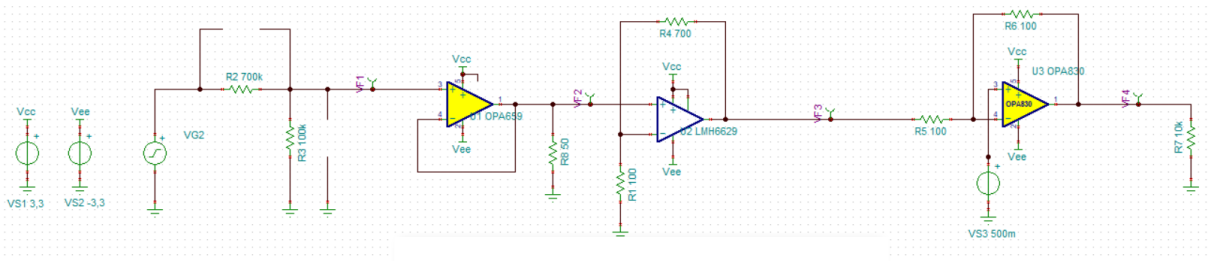

ADC
U14
ADC08100
VIN
VIN_GND
VREF
VRT
VRM
VRB
GNDA
DRVD
DRGND
VA
AGND
CLK
PD
D0
D1
D2
D3
D4
D5
D6
D7
+3.3VA
+3V3
100n
C26
C16
C25
22u
C17
C21
C24
C14
C20
R19
22
CLK
CPU
U21
Raspberry_Pi_Pico
GP0
GP1
GP2
GP3
GP4
GP5
GP6
GP7
GP8
GP9
GP10
GP11
GP12
GP13
GP14
GP15
GP16
GP17
GP18
GP19
GP20
GP21
GP22
GP26
GP27
GP28
RUN
3V3_EN
VUSB
VSYS
3V3_OUT
VREF
SWIO
SWCLK
SDA
SCL
CS
RS
WR
RD
RST
CLK
SEL
TRIG
R38
4.7k
R39
4.7k
+5V
GND
TFT
U22
Q03503
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
DB8
DB9
DB10
DB11
DB12
DB13
DB14
DB15
DB16
DB17
CS
RS
WR
RD
SDI
SDO
IM0
IM1
IM2
RESET
VSYNC
HSYNC
PCLK
DEN
TE
LEDA
LEDK1
LEDK2
LEDK3
LEDK4
LEDK5
LEDK6
YU
XL
YD
XR
VCC
IOVCC
R22
22
100n
C43
IO EXP.
U20
TCA9554DB
SCL
SDA
INT
A0
A1
A2
P0
P1
P2
P3
P4
P5
P6
P7
SEL0
SEL1
SEL2
SEL3
A00
A01
A02
1u
C22
100n
C44
100n
C45
100n
C42
TSC
U19
TSC2004
VREF
X+
X-
Y+
Y-
AUX
IOVDD
SNSVDD
NC
nINT
SDA
SCL
ADO
AD1
nRST
DGND
AGND
SUBGND
SNSGND
PAD
XR
XL
YU
YD
RST

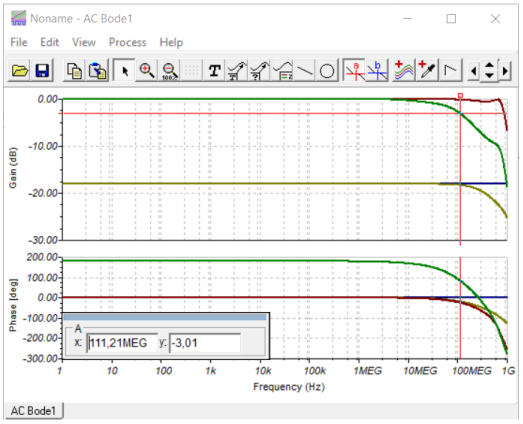

CH1
AC/DC
SWITCH
OFSSET
HIGH-Z
GAIN
DAC
U9
ADG621
U1
TPH2501
OPA357
U8
LMH6629
U10
OPA830
OPA820
U11
DAC5571
J1
Conn_01x02
J7
Conn_01x02_Male
C27 10n
C28 1p
R24 680k
100k R26
C29 6.8p
D3 D
D4 D
C30 100n
C31 100n
C32 100n
C33 100n
C34 100n
R5 47
R36 100
R28 680
R29 100
1k R27
1k R31
R32 100
100 R33
1k R34
R20 100
C40 160p
D5 D
VINA
SDA
SCL
A00
SEL0
SEL1
+3.3VA
-3V3
GNDA

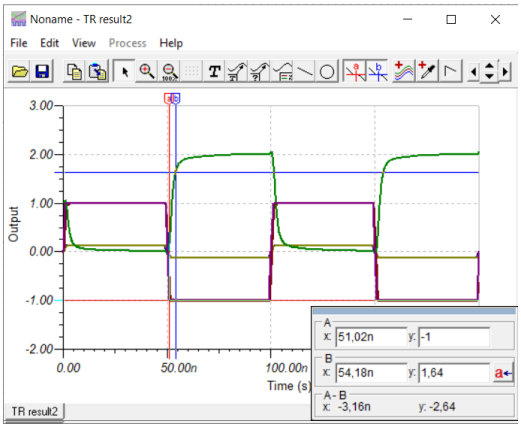

REF
U6
TLV271
U5
LM4040DBZ-2.0
1k R7
C8 1u
C23 100n
C11 100n
VREF
+3.3VA
GNDA

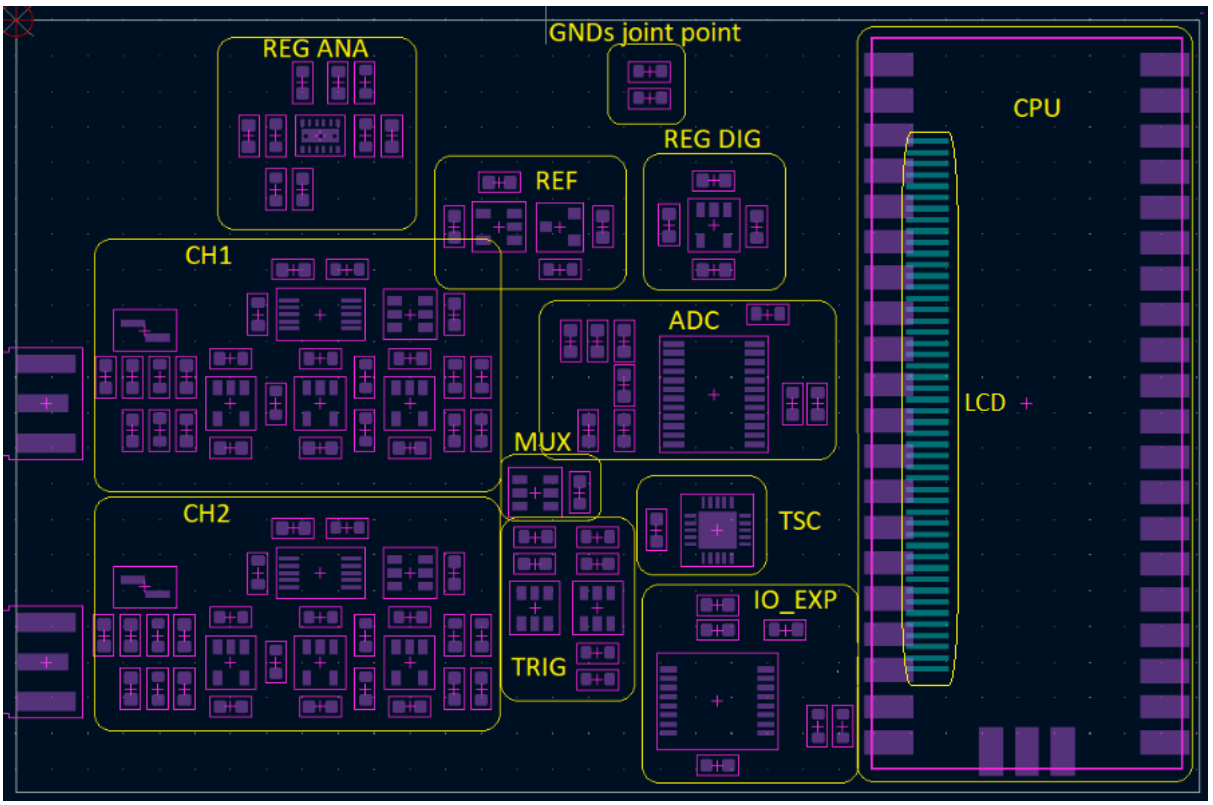

MUX
U2
TMUX1119
100n C12
22 R18
VINA
VINB
VIN
SEL
S1
S2
D
VDD GND
+3.3VA
GNDA

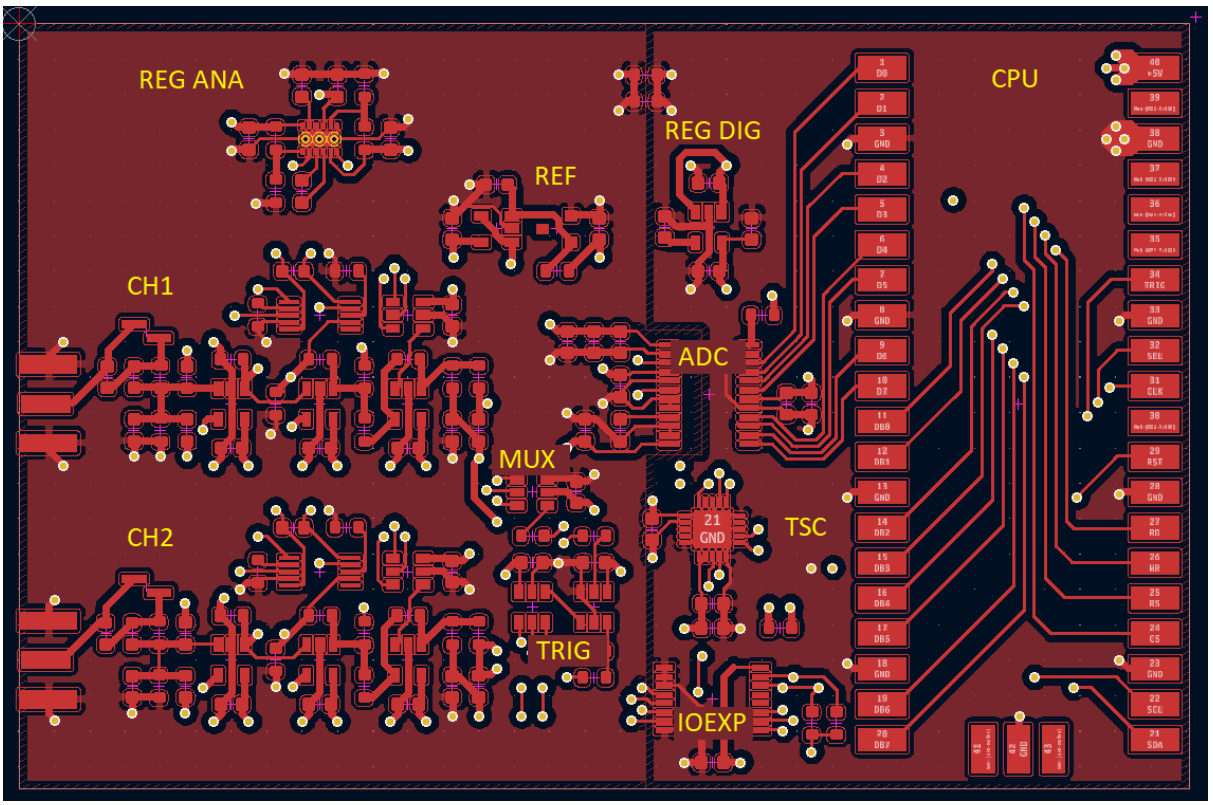

TRIG
U13
TLV3501
U15
DAC5571
100n C35
1MEG R25
0 R23
22 R35
100n C36
VIN
VINA
TRIG
SDA
SCL
A02
VOUT SCL
SDA
A0
VDD GND
+3.3VA
GNDA

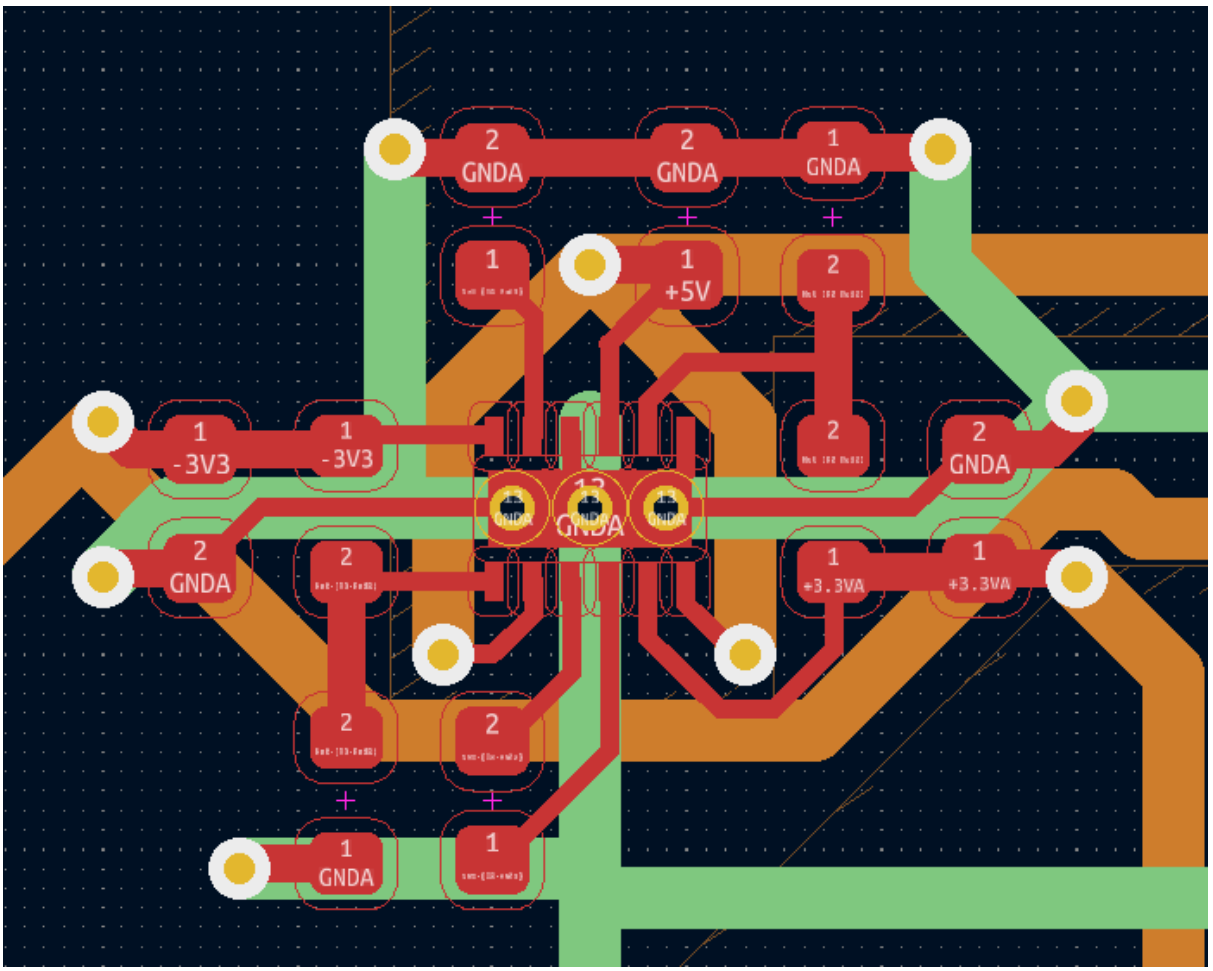

Vcc
Vee
VS1 3,3
VS2 -3,3
VG2
R2 700k
R3 100k
VF1
U1 OPA659
R8 50
VF2
R4 700
U2 LMH6629
R1 100
VF3
R5 100
R6 100
U3 OPA830
VS3 500m
VF4
R7 10k

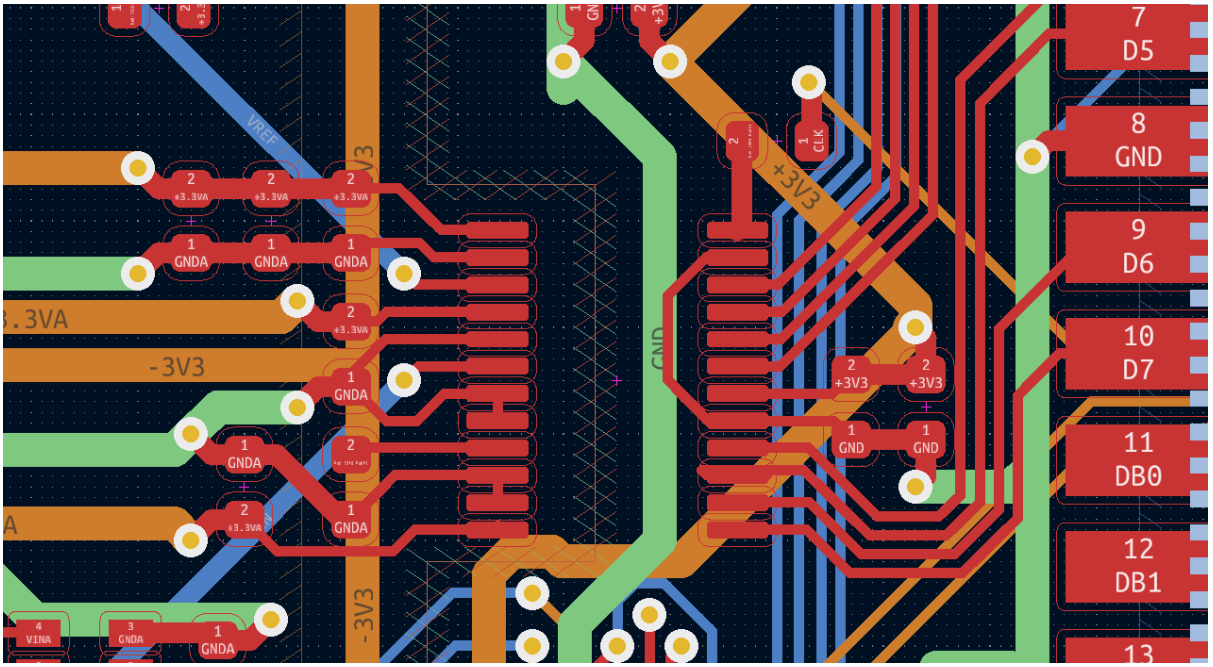

Noname - AC Bode1
File Edit View Process Help
Gain (dB)
Phase (deg)
Frequency (Hz)
A
x: 111,21MEG y: -3,01
AC Bode1

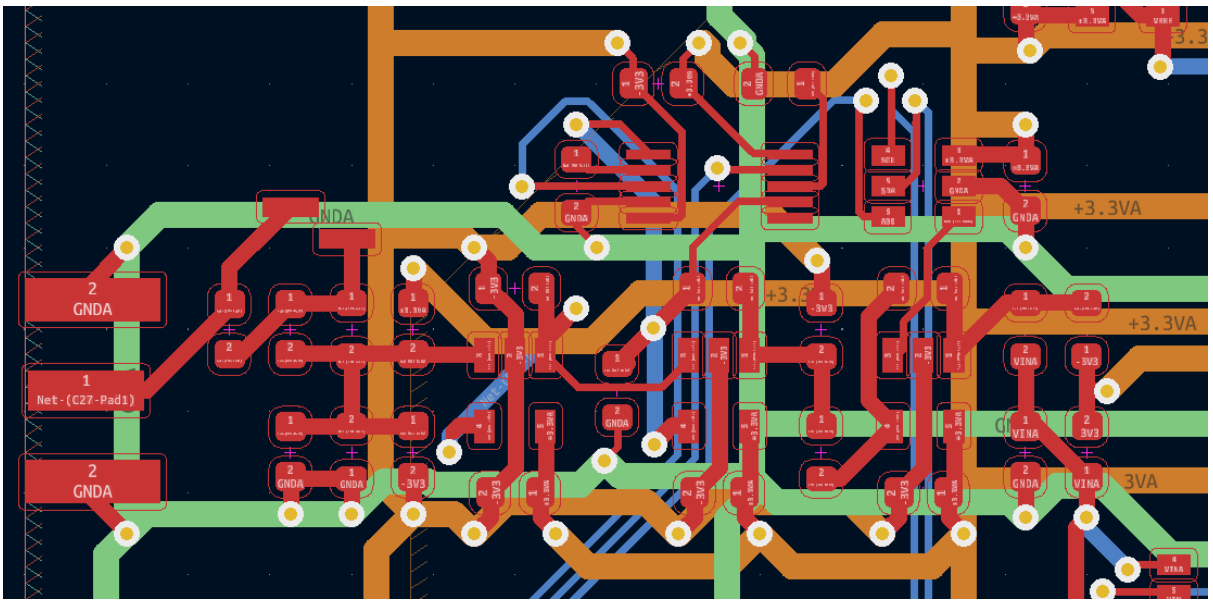

Noname - TR result2
File Edit View Process Help
Output
Time (s)
A
x: 51,02n y: -1
B
x: 54,18n y: 1,64
A - B
x: -3,16n y: -2,64
TR result2

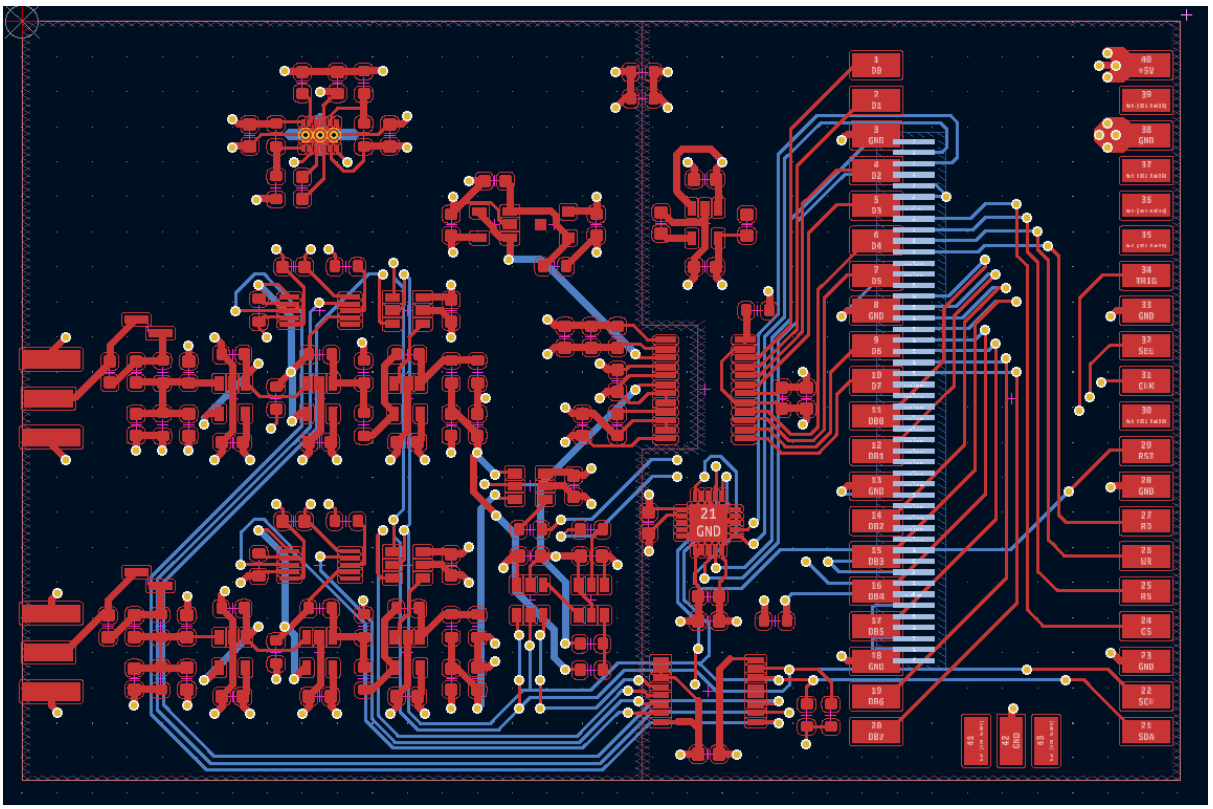

GNDs joint point
REG ANA
REG DIG
CPU
REF
CH1
ADC
LCD
MUX
CH2
TSC
IO_EXP
TRIG

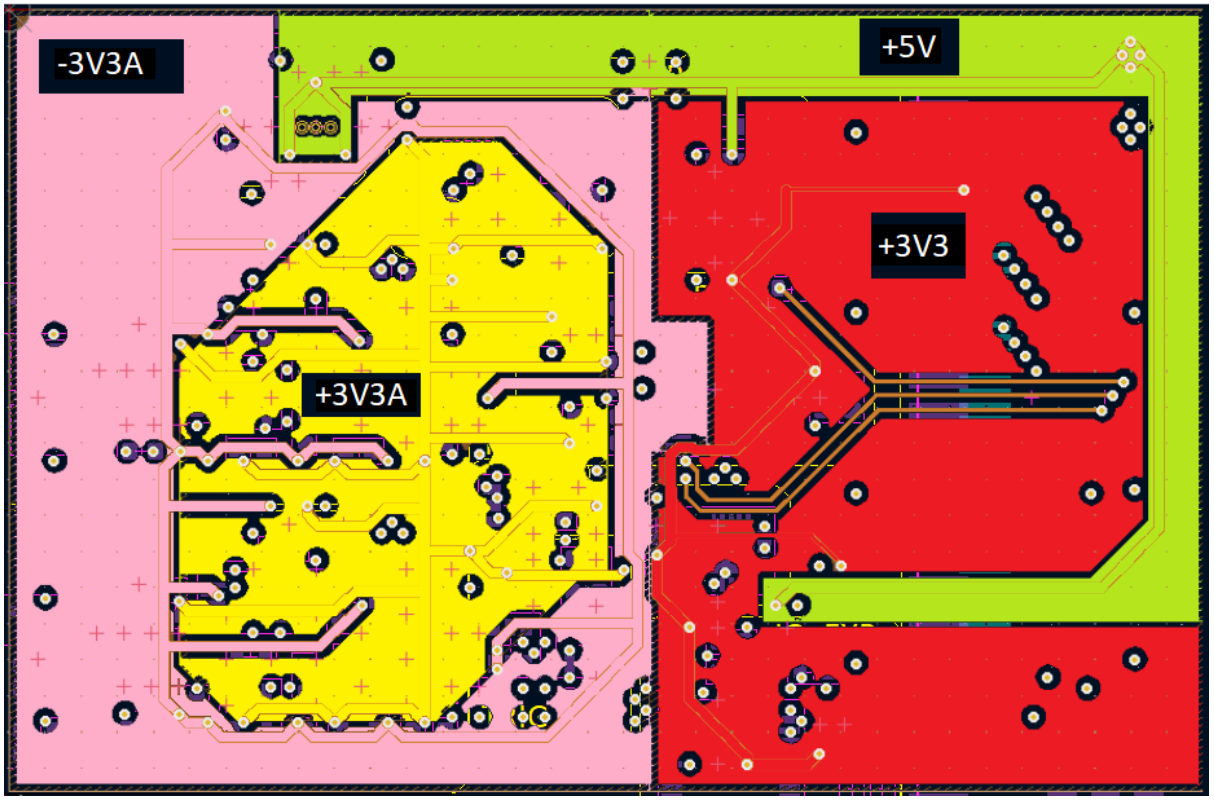

REG ANA
CPU
REG DIG
REF
CH1
ADC
MUX
CH2
TSC
TRIG
IOEXP

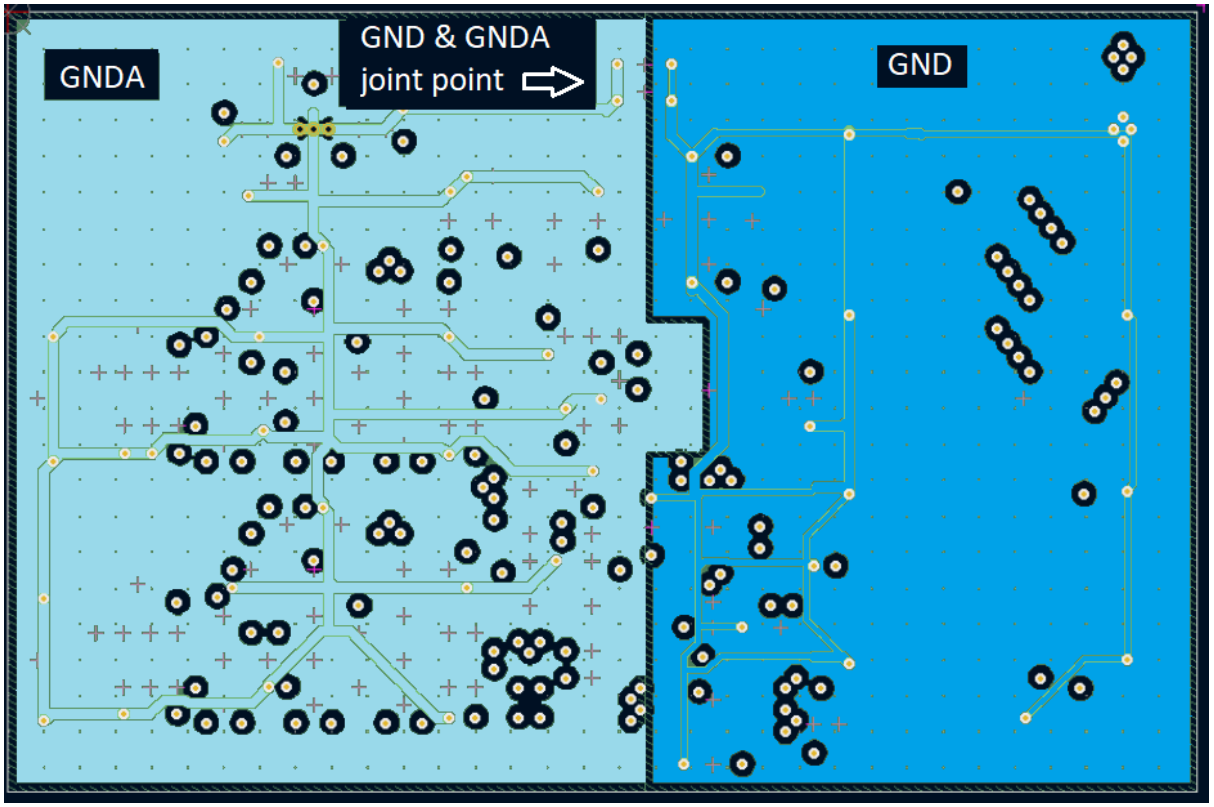

2
GNDA
2
GNDA
1
GNDA
1
1
+5V
2
1
-3V3
1
-3V3
2
2
GNDA
2
GNDA
13
GNDA
1
+3.3VA
1
+3.3VA
2
2
2
2
1
GNDA
1

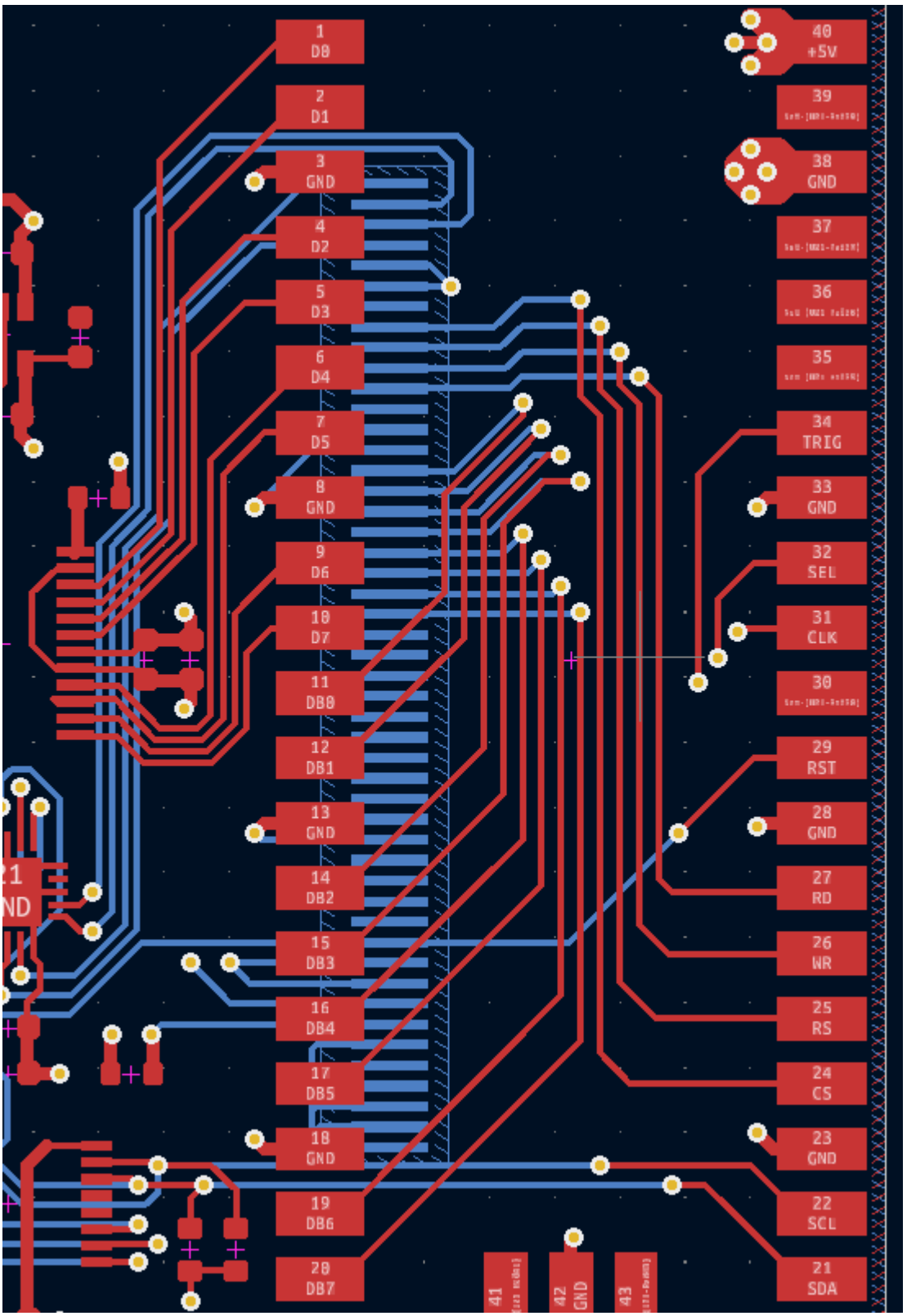

VREF
-3V3
GNDA
+3V3
CLK
GND
7
D5
8
GND
9
D6
10
D7
11
DB0
12
DB1
13
VINA
GNDA

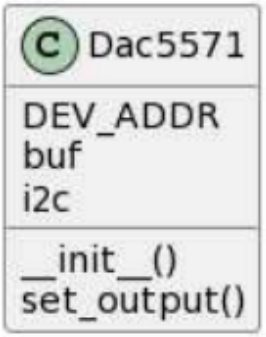

2
GNDA
1
Net-(C27-Pad1)
2
GNDA
+3.3VA
+3.3VA
3VA

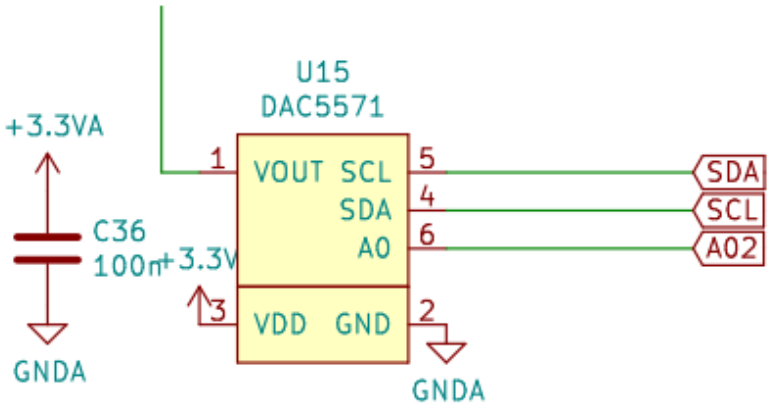

21
GND


-3V3A
+5V
+3V3
+3V3A

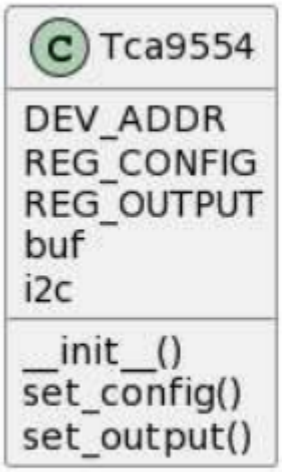

GNDA
GND & GNDA
joint point
GND

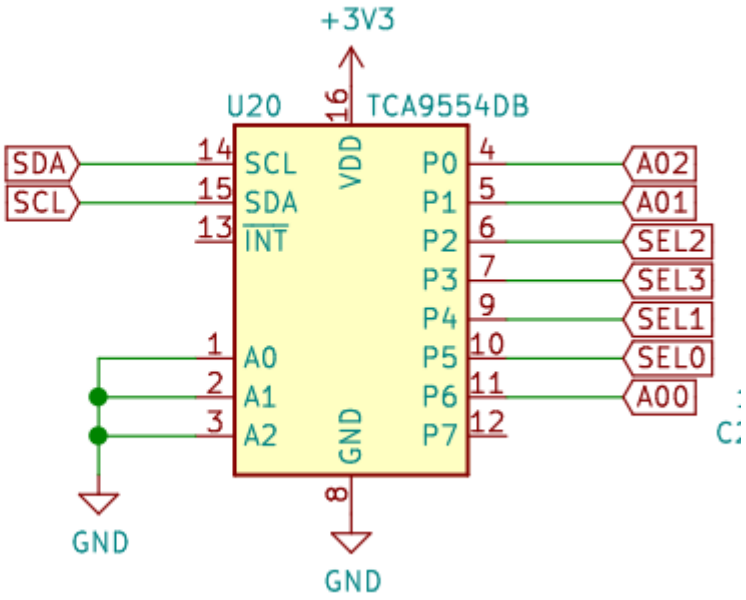

1
D0
2
D1
3
GND
4
D2
5
D3
6
D4
7
D5
8
GND
9
D6
10
D7
11
DB0
12
DB1
13
GND
14
DB2
15
DB3
16
DB4
17
DB5
18
GND
19
DB6
20
DB7
40
+5V
39
38
GND
37
36
35
34
TRIG
33
GND
32
SEL
31
CLK
30
29
RST
28
GND
27
RD
26
WR
25
RS
24
CS
23
GND
22
SCL
21
SDA
41
42
GND
43

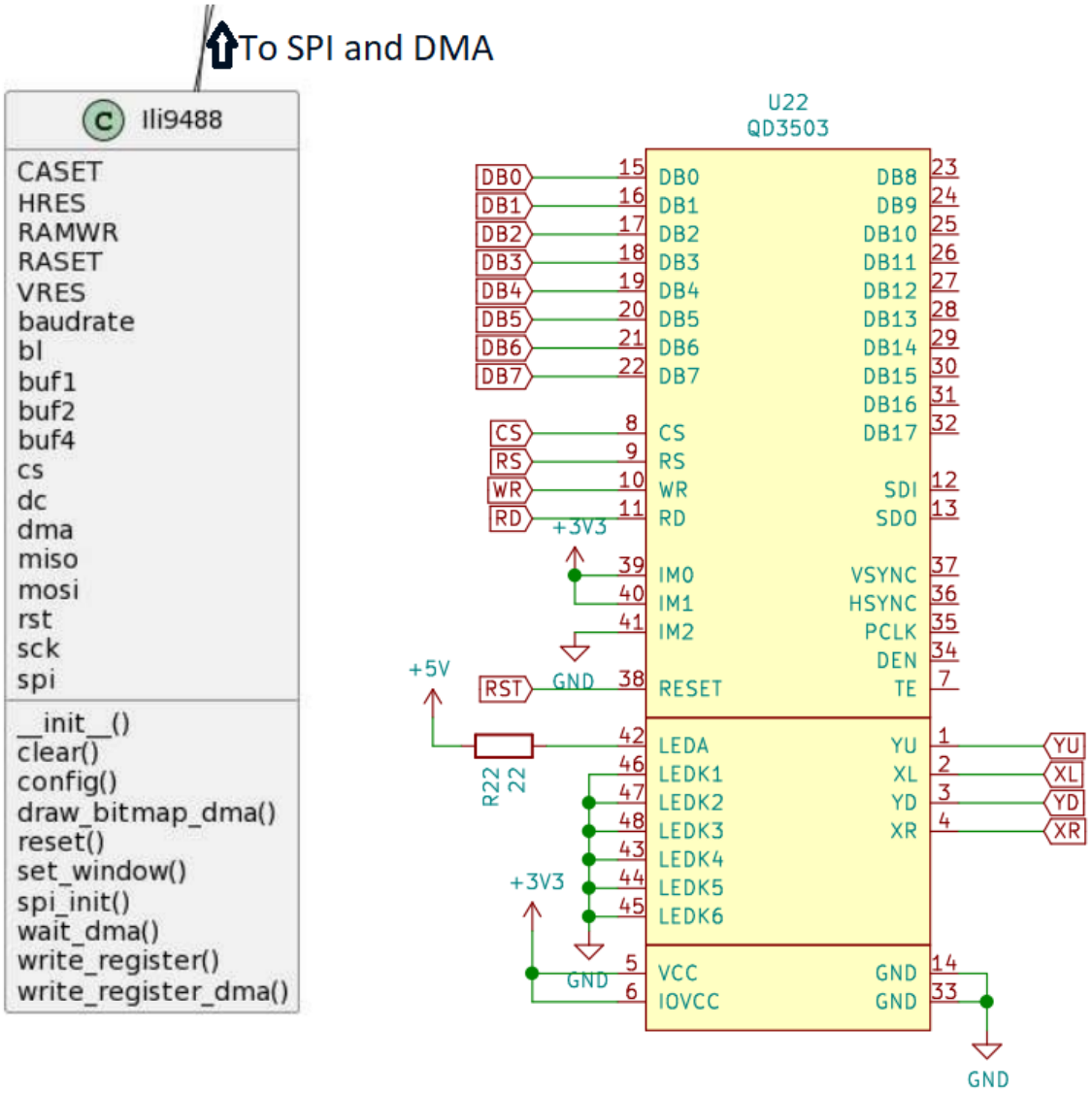

Dac5571
DEV_ADDR
buf
i2c
__init__()
set_output()


U15
DAC5571
+3.3VA
C36
100n
GNDA
+3.3V
VOUT
SCL
SDA
A0
VDD
GND
1
2
3
4
5
6
SDA
SCL
A02
GNDA

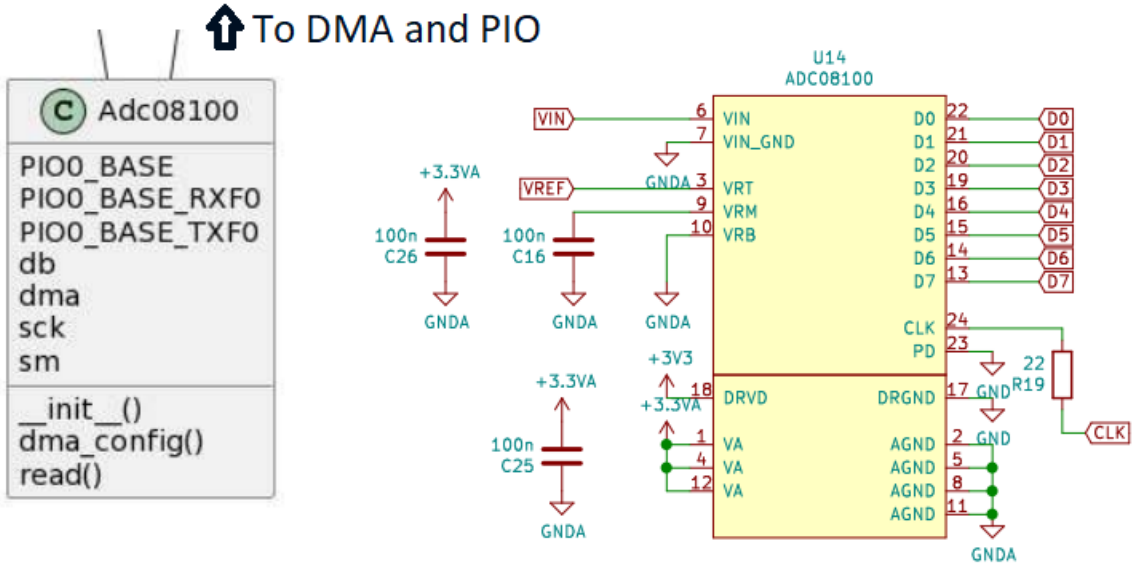
IO EXP.


Tca9554
DEV_ADDR
REG_CONFIG
REG_OUTPUT
buf
i2c
__init__()
set_config()
set_output()


+3V3
U20
TCA9554DB
SDA
SCL
SCL
SDA
INT
A0
A1
A2
VDD
GND
P0
P1
P2
P3
P4
P5
P6
P7
A02
A01
SEL2
SEL3
SEL1
SEL0
A00
GND
GND

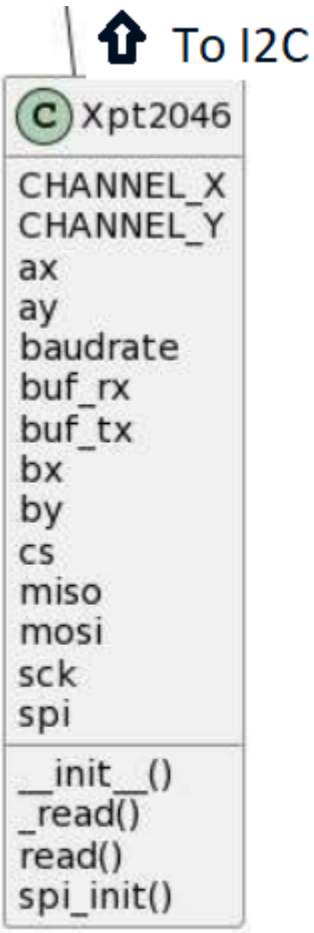

To SPI and DMA
Ili9488
CASET
HRES
RAMWR
RASET
VRES
baudrate
bl
buf1
buf2
buf4
cs
dc
dma
miso
mosi
rst
sck
spi
__init__()
clear()
config()
draw_bitmap_dma()
reset()
set_window()
spi_init()
wait_dma()
write_register()
write_register_dma()
U22
QD3503
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
CS
RS
WR
RD
+3V3
IM0
IM1
IM2
GND
+5V
RST
RESET
R22
22
LEDA
LEDK1
LEDK2
LEDK3
LEDK4
LEDK5
LEDK6
+3V3
VCC
IOVCC
GND
DB8
DB9
DB10
DB11
DB12
DB13
DB14
DB15
DB16
DB17
SDI
SDO
VSYNC
HSYNC
PCLK
DEN
TE
YU
XL
YD
XR
GND

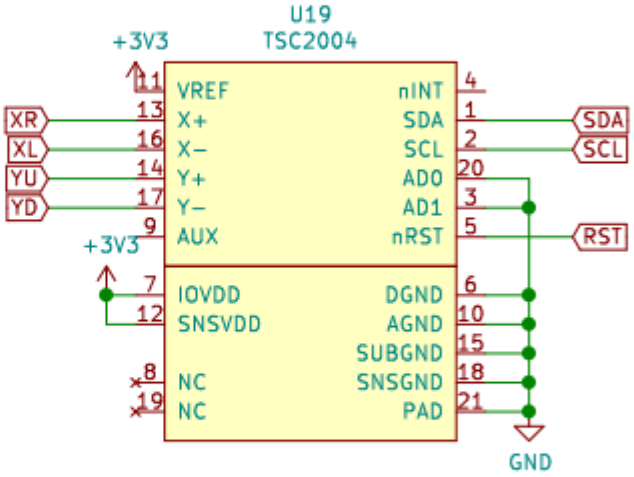
```
class Ili9488:
    HRES = 480
    VRES = 320

    CASET = 0x2A
    RASET = 0x2B
    RAMWR = 0x2C
    def  __init__(self, baudrate, cs, sck, mosi, miso, dc, rst, bl):
    
    def reset(self):
    
    def spi_init(self):
    
    def write_register(self, reg, buf):

    def write_register_dma(self, reg, buf, is_blocking=True):    

    def wait_dma(self):
    
    def config(self):

    def set_window(self, x, y, w, h):
        
    def draw_bitmap_dma(self, x, y, w, h, buf, is_blocking=True):
        self.set_window(x, y, w, h)
        self.write_register_dma(Ili9488.RAMWR, buf, is_blocking)
            
    def clear(self, color):
```



```
@rp2.asm_pio(
    sideset_init=(rp2.PIO.OUT_HIGH, rp2.PIO.OUT_HIGH),
    in_shiftdir=rp2.PIO.SHIFT_LEFT,
)
def build_sm_adc08100():
    in_(pins, 8).side(0b0)
    push(block).side(0b1)

@rp2.asm_pio(
)
def build_sm_adc08100_trigger():

class Adc08100:
    PIO0_BASE      = 0x50200000
    PIO0_BASE_TXF0 = PIO0_BASE + 0x10
    PIO0_BASE_RXF0 = PIO0_BASE + 0x20

    def __init__(self, sps, sck, db, use_trigger=False):
    
    def dma_config(self, buf, count, ring_size_pow2=0):
    
    def read(self, buf, dma_config=True):
        if(dma_config):
            self.dma_config(buf, len(buf))
        
        self.dma.enable()
        self.sm.active(True)
        while(self.dma.is_busy()):
            pass
        self.sm.active(False)
        self.dma.disable()
```

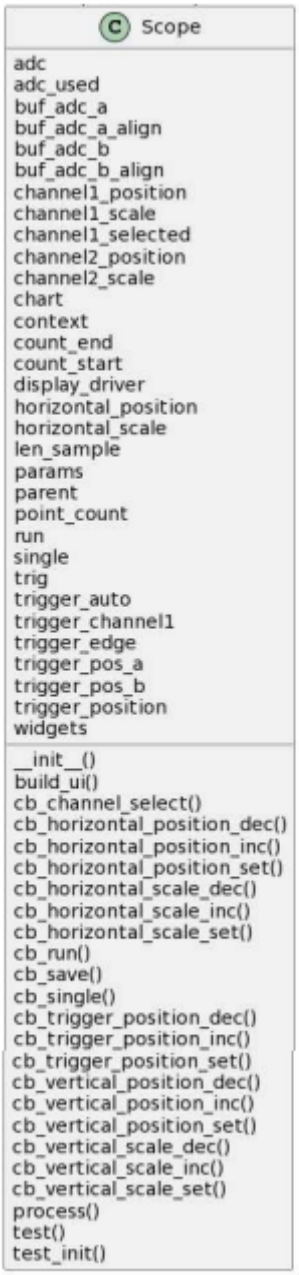
```
@rp2.asm_pio()
def build_sm_trigger_rising():
    label("prog_trigger")
    pull(block)
    mov(y, osr)
    pull(block)
    mov(x, osr)
    irq(4)

    label("loop_y_dec")
    nop()
    jmp(y_dec, "loop_y_dec")

    wait(0, pin, 0)
    wait(1, pin, 0)

    label("loop_x_dec")
    nop()
    jmp(x_dec, "loop_x_dec")

    push(block)
    nop()[4]
    irq(block, 1)

    jmp("prog_trigger")

@rp2.asm_pio()
def build_sm_trigger_falling():

class Trigger:
    def __init__(self, sps, trig, rising):

    def read(self, pre, post, addr_stop):
        self.dma.config(
        )

        self.sm.put(pre)
        self.sm.put(post)

        self.dma.enable()
        self.sm.active(True)

        while(self.dma.is_busy()):
            pass

        self.sm.active(False)
        self.dma.disable()
```


To I2C
Xpt2046
CHANNEL_X
CHANNEL_Y
ax
ay
baudrate
buf_rx
buf_tx
bx
by
cs
miso
mosi
sck
spi
__init__()
_read()
read()
spi_init()

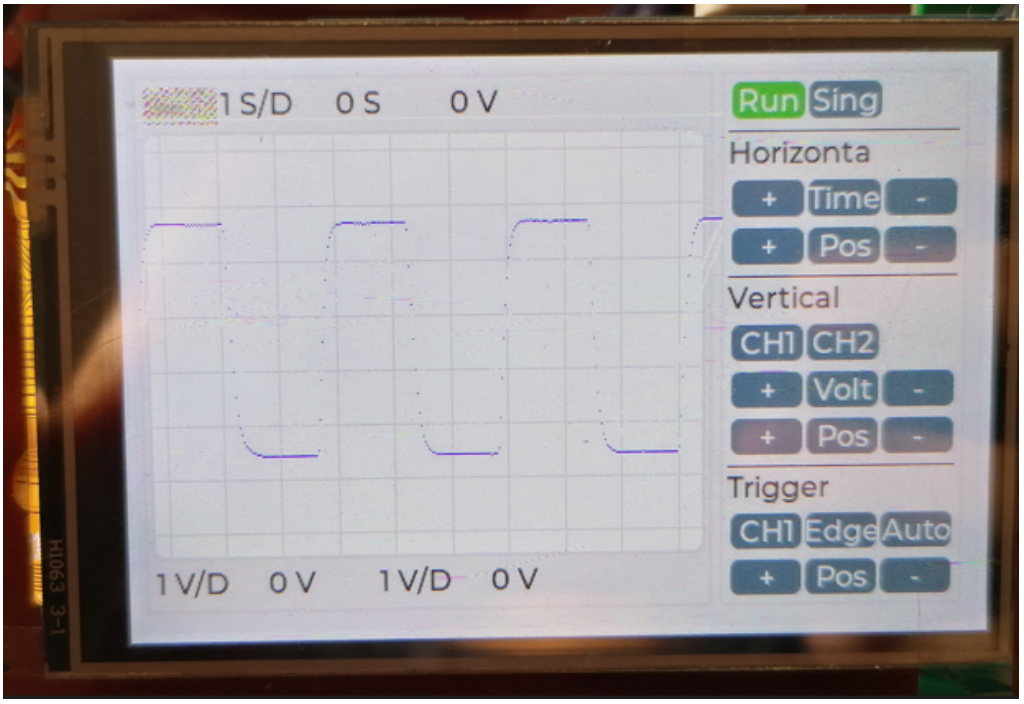

U19
TSC2004
+3V3
VREF
X+
X−
Y+
Y−
AUX
IOVDD
SNSVDD
NC
NC
nINT
SDA
SCL
ADO
AD1
nRST
DGND
AGND
SUBGND
SNSGND
PAD
XR
XL
YU
YD
SDA
SCL
RST
GND

```
class Xpt2046:
    
    CHANNEL_X = 0x90
    CHANNEL_Y = 0xD0
    
    def  init  (self, baudrate, cs, sck, mosi, miso, ax=1, bx=0, ay=1, by=0):
    
    def spi init(self):
    
    def  read(self):
    
    def read(self):
        xacc = 0
        yacc = 0
        for i in range(4):
            x, y = self._read()
            if(x and y):
                xacc += x
                yacc += y
            else:
                return 0,0
        
        x = xacc/4
        y = yacc/4
        
        x = self.ax*x + self.bx
        y = self.ay*y + self.by
        
        return int(x),int(y)
```

```
class DMA:
    DMA_BASE    = 0x50000000
    DMA_EN      = 0x01 << 0
    HIGH_PRIO   = 0x01 << 1
    INCR_READ   = 0x01 << 4
    INCR_WRITE  = 0x01 << 5
    DREQ_PIO0_RX0 = 0x04 << 15
    DREQ_PIO0_RX1 = 0x05 << 15
    DREQ_SPI1_TX  = 0x12 << 15
    DREQ_PERMANENT= 0x3F << 15
    IRQ_QUIET   = 0x01 << 21
    BUSY        = 0x01 << 24
    CHAIN_TO_POS = 0x0B
    RING_SEL_POS = 0x0A
    RING_SIZE_POS= 0x06

    def __init__(self, channel_number):

    def config(self, src_addr, dst_addr, count, src_inc, dst_inc, trig_dreq, ring_sel=False, ring_size_pow2=0):
        machine.mem32[self.CHx_CTRL_TRIG]   = 0
        machine.mem32[self.CHx_READ_ADDR]   = src_addr
        machine.mem32[self.CHx_WRITE_ADDR]  = dst_addr
        machine.mem32[self.CHx_TRANS_COUNT] = count

        trig_val = 0
        if(src_inc):
            trig_val |= DMA.INCR_READ
        if(dst_inc):
            trig_val |= DMA.INCR_WRITE

        trig_val |= self.channel_number << DMA.CHAIN_TO_POS
        trig_val |= ring_sel << DMA.RING_SEL_POS
        trig_val |= ring_size_pow2 << DMA.RING_SIZE_POS
        trig_val |= trig_dreq

        machine.mem32[self.CHx_CTRL_TRIG]   = trig_val

    def enable(self):

    def disable(self):

    def is_busy(self):
```


Scope
adc
adc_used
buf_adc_a
buf_adc_a_align
buf_adc_b
buf_adc_b_align
channel1_position
channel1_scale
channel1_selected
channel2_position
channel2_scale
chart
context
count_end
count_start
display_driver
horizontal_position
horizontal_scale
len_sample
params
parent
point_count
run
single
trig
trigger_auto
trigger_channel1
trigger_edge
trigger_pos_a
trigger_pos_b
trigger_position
widgets
__init__()
build_ui()
cb_channel_select()
cb_horizontal_position_dec()
cb_horizontal_position_inc()
cb_horizontal_position_set()
cb_horizontal_scale_dec()
cb_horizontal_scale_inc()
cb_horizontal_scale_set()
cb_run()
cb_save()
cb_single()
cb_trigger_position_dec()
cb_trigger_position_inc()
cb_trigger_position_set()
cb_vertical_position_dec()
cb_vertical_position_inc()
cb_vertical_position_set()
cb_vertical_scale_dec()
cb_vertical_scale_inc()
cb_vertical_scale_set()
process()
test()
test_init()

```
def build_ui( self, parent ):
    width = 160
    point_count = 480-width

    self.context.append( parent )
    set_context( self.context )
    set_widgets( self.widgets )

    with Cont():
        with Cont( 0, 0, 480-width, 320-15 ):
            with Column():
                lblw= 74
                with Row():
                    add_label( "#Status", w=lblw )
                    add_label( "#lHS", w=lblw ).set_text( "1 S/D" )
                    add_label( "#lHP", w=lblw ).set_text( "0 S" )
                    add_label( "#lTP", w=lblw ).set_text( "0 V" )

                self.chart = lv.chart( self.context[-1] )
                self.chart.set_size( 480-160-12, 300-5-50-10 )
                #self.chart.set_style_bg_color( lv.palette_main( 0 ), 0 )
                self.chart.set_div_line_count( 8, 10 )
                with Row():
                    add_label( "#lVS1", w=lblw ).set_text( "1 V/D" )
                    add_label( "#lVP1", w=lblw ).set_text( "0 V" )
                    add_label( "#lVS2", w=lblw ).set_text( "1 V/D" )
                    add_label( "#lVP2", w=lblw ).set_text( "0 V" )
        with Cont( 480-width, 0, width-15, 320-15 ):
            with Column():
                with Row():
                    add_button( "Run#M" ).add_event_cb( self.cb_run, lv.EVENT.VALUE_CHANGED, None )
                    self.widgets["Run#M"].add_flag( lv.obj.FLAG.CHECKABLE )
                    self.widgets["Run#M"].get_child(0).set_text( lv.SYMBOL.PLAY )
                    add_button( "Sing#M" ).add_event_cb( self.cb_single, lv.EVENT.CLICKED, None )
                    self.widgets["Sing#M"].get_child(0).set_text( lv.SYMBOL.NEXT )
                    add_button( "#Save" ).add_event_cb( self.cb_save, lv.EVENT.CLICKED, None )
                    self.widgets["#Save"].get_child(0).set_text( lv.SYMBOL.SAVE )
                add_line( 0, 2, 120+2*4, 1 )

                add_label( "Horizontal", w=100 )
                with Row():
                    add_button( "+#HT" ).add_event_cb( self.cb_horizontal_scale_inc, lv.EVENT.CLICKED, None )
                    self.widgets["+#HT"].get_child(0).set_text( lv.SYMBOL.PLUS )
                    add_button( "Time" ).add_event_cb( self.cb_horizontal_scale_set, lv.EVENT.CLICKED, None )
                    add_button( "-#HT" ).add_event_cb( self.cb_horizontal_scale_dec, lv.EVENT.CLICKED, None )
                    self.widgets["-#HT"].get_child(0).set_text( lv.SYMBOL.MINUS )
                with Row():
                    add_button( "+#HP" ).add_event_cb( self.cb_horizontal_position_inc, lv.EVENT.CLICKED, None )
                    self.widgets["+#HP"].get_child(0).set_text( lv.SYMBOL.PLUS )
                    add_button( "Pos#H" ).add_event_cb( self.cb_horizontal_position_set, lv.EVENT.CLICKED, None )
                    add_button( "-#HP" ).add_event_cb( self.cb_horizontal_position_dec, lv.EVENT.CLICKED, None )
                    self.widgets["-#HP"].get_child(0).set_text( lv.SYMBOL.MINUS )
                    self.widgets["+#HP"].add_event_cb( self.cb_horizontal_position_inc, lv.EVENT.LONG_PRESSED_REPEAT, None )
                    self.widgets["-#HP"].add_event_cb( self.cb_horizontal_position_dec, lv.EVENT.LONG_PRESSED_REPEAT, None )
```



```
import lvgl as lv

class Display_Driver:
    def __init__(self,width,height,lcd,tsc,fb_rows=64):
        
    def init_gui(self):
        lv.init()
        
        self.fb1 = bytearray(self.width*self.fb_rows)
        self.fb2 = bytearray(self.width*self.fb_rows)
        print("len(fb1)",len(self.fb1))
        print("len(fb2)",len(self.fb2))
        
        print("disp_draw_buf_t()")
        self.disp_draw_buf = lv.disp_draw_buf_t()
        self.disp_draw_buf.init(self.fb1,self.fb2,len(self.fb1)//lv.color_t.SIZE)

        print("disp_drv_t()")
        self.disp_drv = lv.disp_drv_t()
        self.disp_drv.init()
        self.disp_drv.draw_buf = self.disp_draw_buf
        self.disp_drv.flush_cb = self.disp_drv_flush_cb
        self.disp_drv.hor_res = self.width
        self.disp_drv.ver_res = self.height
        self.disp_drv.register()

        print("indev_drv_t()")
        self.indev_drv = lv.indev_drv_t()
        self.indev_drv.init()
        self.indev_drv.type = lv.INDEV_TYPE.POINTER
        self.indev_drv.read_cb = self.indev_drv_read_cb
        self.indev_drv.register()

    def disp_drv_flush_cb(self,disp_drv,area,color):
        #print("disp_drv_flush_cb",area.x1,area.x2,area.y1,area.y2)
        
        if(self.dma_running == True):
            self.lcd.wait_dma()
            self.dma_running = False
        
        if(self.is_fb1):
            fb = memoryview(self.fb1)
        else:
            fb = memoryview(self.fb2)
        self.is_fb1 = not self.is_fb1
        
        x = area.x1
        y = area.y1
        w = area.x2 - area.x1 + 1
        h = area.y2 - area.y1 + 1
        self.lcd.draw_bitmap_dma(x, y, w, h, fb[0:w*h*lv.color_t.SIZE], is_blocking=False)
        self.dma_running = True
        
        disp_drv.flush_ready()
        
    def indev_drv_read_cb(self,indev_drv,data):
```

```
import lvgl as lv

context = []
widgets = {}

def set_context( ctx ):

def get_context():

def set_widgets( wgts ):

def get_widgets():

style = lv.style_t()
style.init()
style.set_pad_all( 2 )
style.set_pad_gap( 2 )
style.set_radius( 2 )
style.set_border_width( 0 )
style.set_bg_color( lv.palette_lighten( lv.PALETTE.GREY, 3 ) )

class Cont:
    def __init__( self, x=5, y=5, w=480-10, h=320-10 ):
        self.obj = lv.obj( context[-1] )
        self.obj.add_style( style, 0 )
        #self.obj.set_flex_flow( lv.FLEX_FLOW.COLUMN )
        self.obj.set_pos( x, y )
        self.obj.set_size( w, h )
        #if( len( context ) == 3 ):
        self.obj.set_style_border_width( 0, 0 )

    def __enter__( self ):
        context.append( self.obj )
        return self.obj

    def __exit__( self, a, b, c ):
        context.pop()

class Column:

class Row:

def add_button( name, w=40, h=20, radius=5, checkable=False ):

def add_spinbox( name, w=100, h=40 ):

def add_label( name, w=40, h=20 ):

def add_line( x=0, y=0, w=40, h=1, width=1 ):
```

```
def capture( sps, buf, rising, pre, post ):
    db  = machine.Pin( 0 ) # out
    sck = machine.Pin( 21 ) # side 0
    mux = machine.Pin( 20 ) # side 0
    trig = machine.Pin( 7, machine.Pin.IN, machine.Pin.PULL_DOWN ) # trigger
    #trig = machine.Pin( 19, machine.Pin.IN, machine.Pin.PULL_DOWN ) # trigger

    buf_addr_aligned = (uctypes.addressof( buf )+0xFF)&0xFFFFFF00
    buf_offset_aligned = buf_addr_aligned - uctypes.addressof( buf )

    t0 = time.ticks_us()
    adc = Adc08100( sps, mux, db, use_trigger=True )
    t1 = time.ticks_us()

    t0 = time.ticks_us()
    adc.dma.config(
        Adc08100.PIO0_BASE_RXF0,
        buf_addr_aligned,
        0xFFFFFFFF,
        src_inc=False,
        dst_inc=True,
        trig_dreq=DMA.DREQ_PIO0_RX0,
        ring_sel=True,
        ring_size_pow2=8
    )
    t1 = time.ticks_us()

    t0 = time.ticks_us()
    adc.dma.enable()
    adc.sm.active( True )
    t1 = time.ticks_us()

    t0 = time.ticks_us()
    trigger = Trigger( sps, trig, rising )
    t1 = time.ticks_us()

    DMA1_TRIG = 0x50000000 + 1 * 0x40 + 0x0C

    t0 = time.ticks_us()
    trans_count0 = machine.mem32[ adc.dma.CHx_TRANS_COUNT ]
    trigger.read( pre, post, DMA1_TRIG )
    trans_count1 = machine.mem32[ adc.dma.CHx_TRANS_COUNT ]
    t1 = time.ticks_us()

    t0 = time.ticks_us()
    adc.dma.disable()
    adc.sm.active( False )
    t1 = time.ticks_us()

    trans_count_diff = trans_count0 - trans_count1 # - 1

    
    trans_count_diff = trans_count_diff&0xFF

    return buf_offset_aligned, trans_count_diff
```