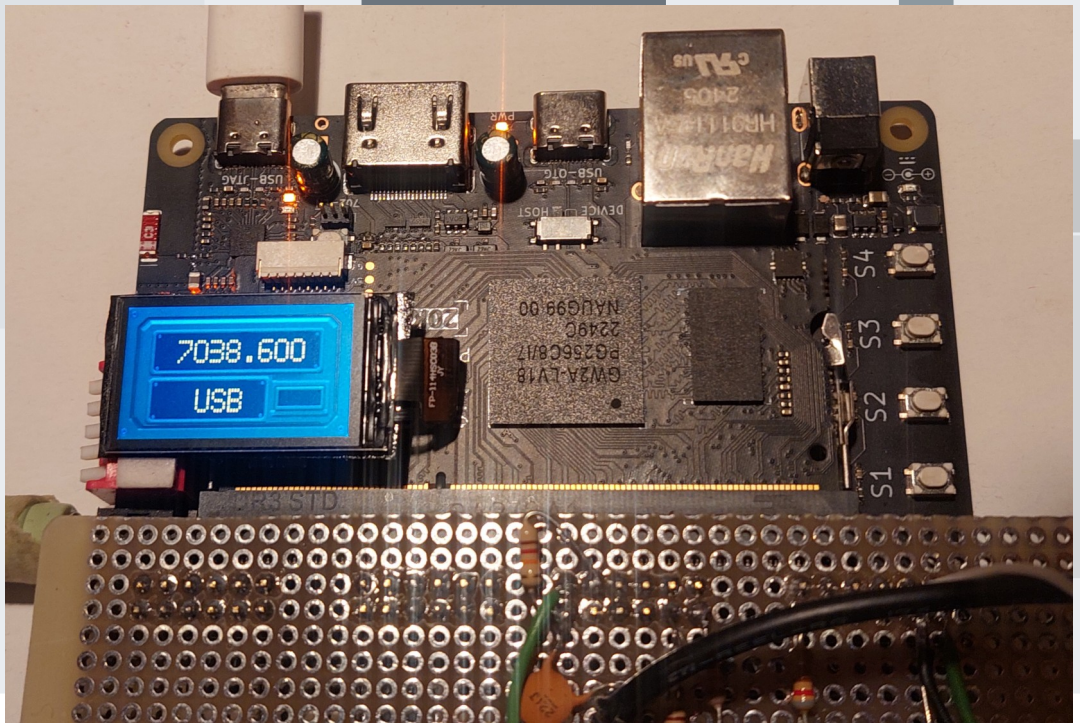
RISC-V®

baita ovest

LiteX by EnjoyDigital

Build your hardware, easily!



LVGL

SiPEED

GOWIN

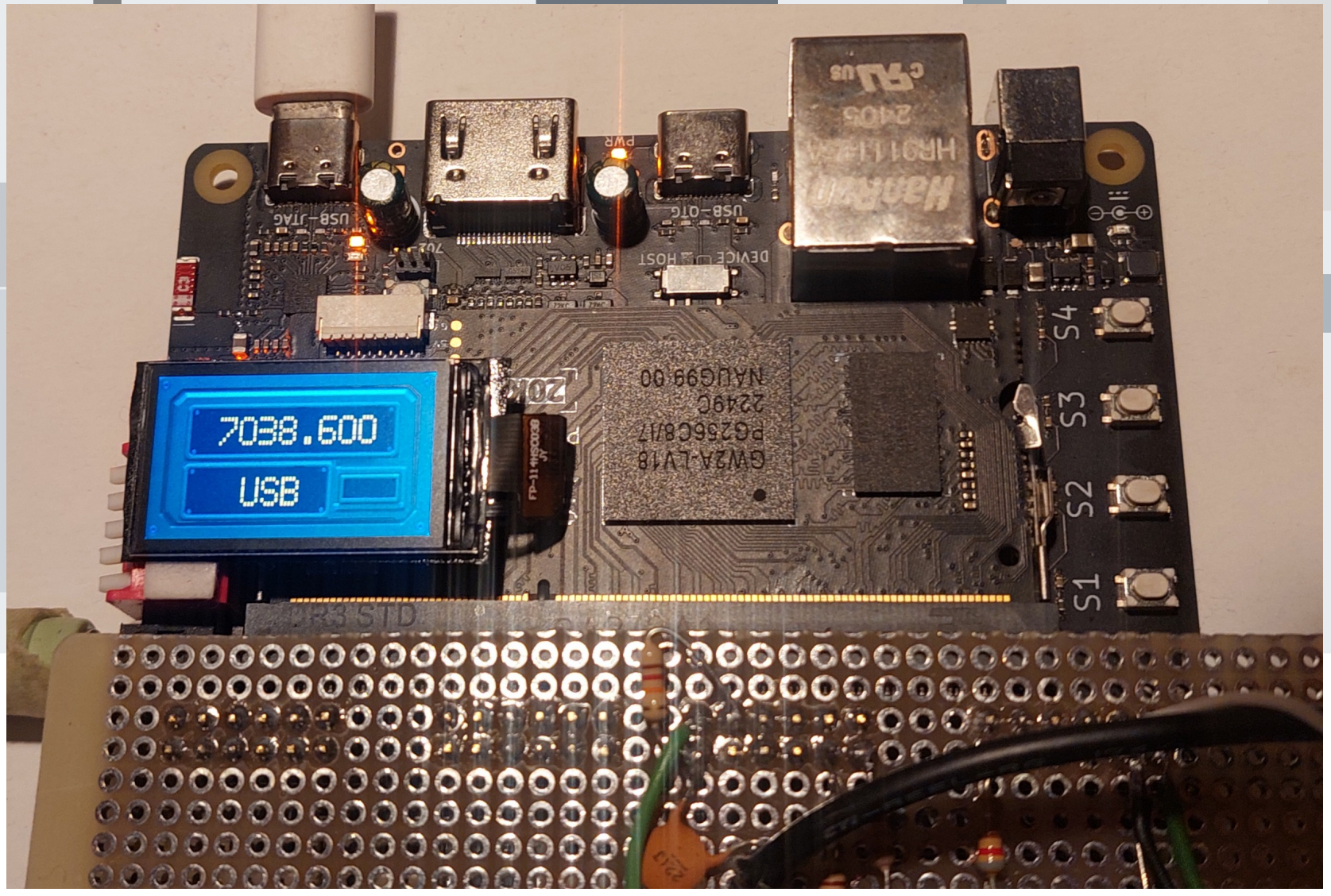
PROGRAMMING FOR THE FUTURE

SoCmel_1

SDR on a Litex SoC

Alberto I4NZX,

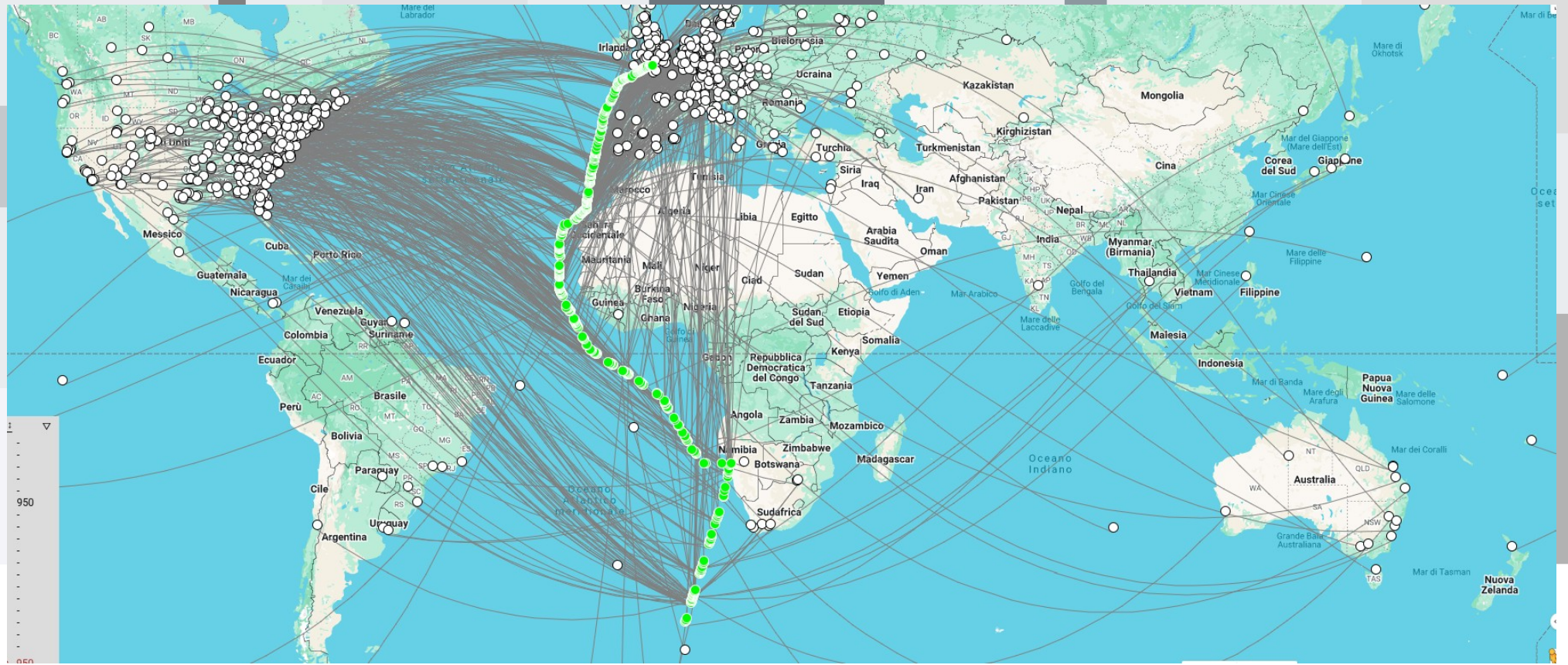
Baita Ovest, November 2025

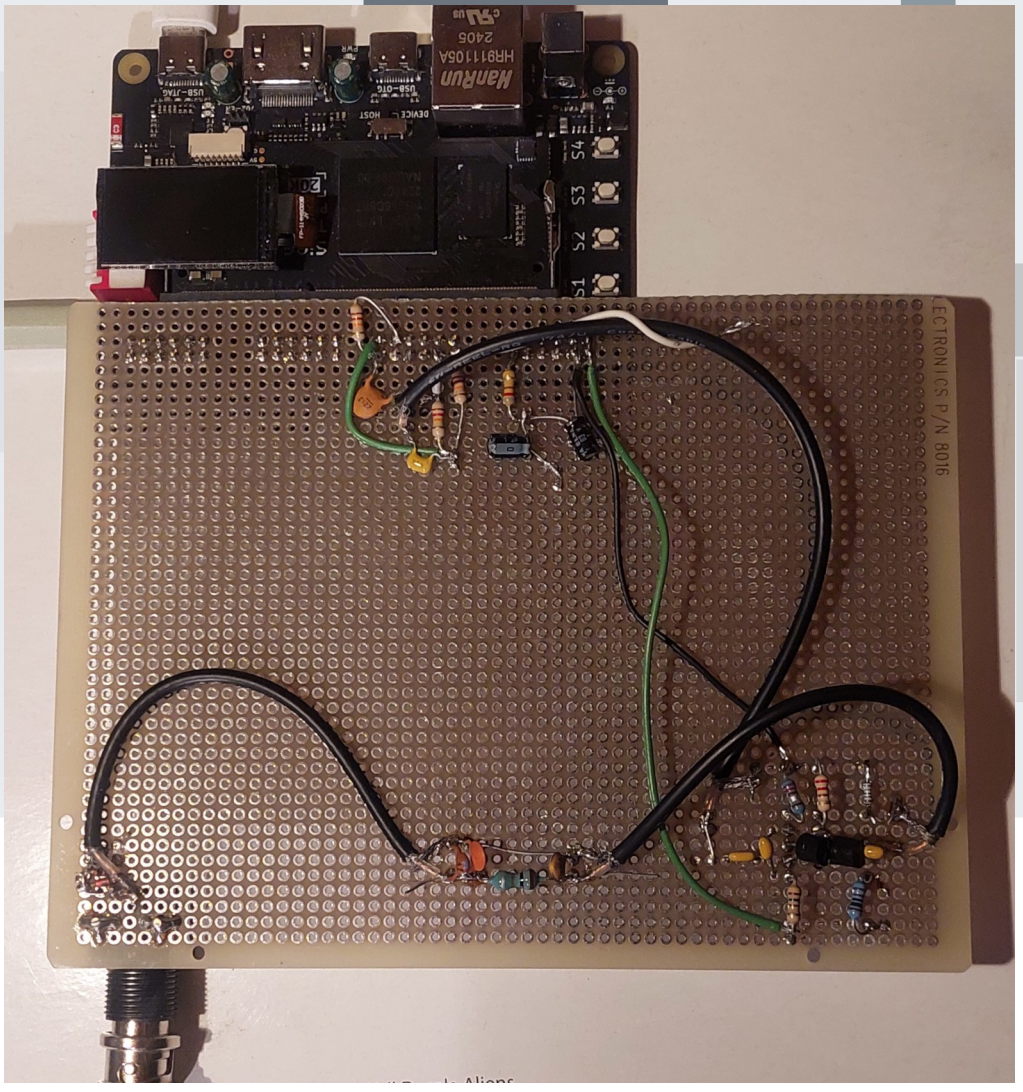

7038.600
USB
S1
S2
S3
S4
USB-JTAG
USB-OTG
DEVICE
HOST
PWR

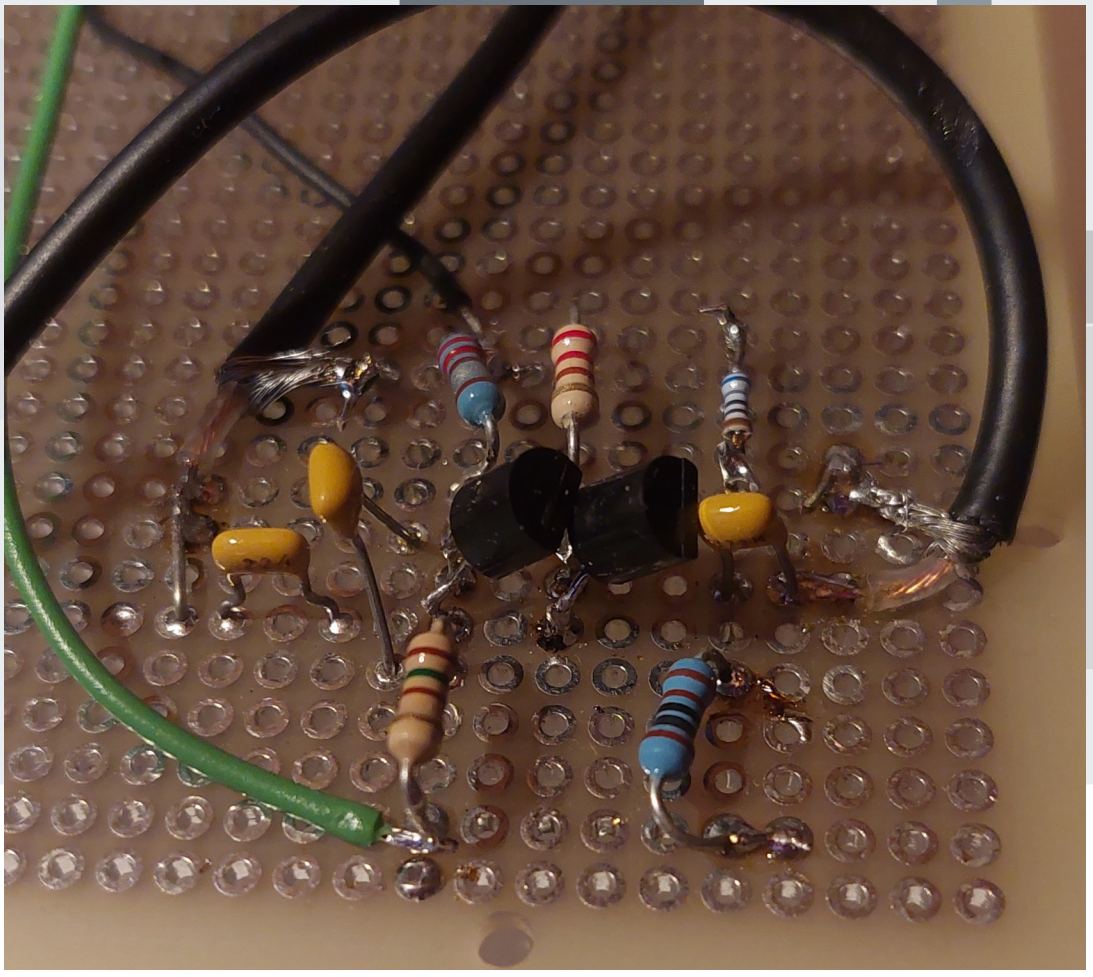

band count
LF -
MF -
160m -
80m -
60m -
40m 89
30m -
20m -
17m -
15m -
12m -
10m -
8m -
6m -
4m -
2m -
70cm -
23cm -
Total spots 89
Canada
Stati Uniti
Messico
Cuba
Porto Rico
Guatemala
Nicaragua
Venezuela
Guyana
Suriname
Colombia
Ecuador
Perù
Brasile
Bolivia
Paraguay
Cile
Uruguay
Groenlandia
Islanda
Svezia
Norvegia
Finlandia
Russia
Danimarca
Irlanda
Polonia
Bielorussia
Ucraina
Romania
Spagna
Portogallo
Grecia
Turchia
Kazakistan
Mongolia
Kirghizistan
Turkmenistan
Siria
Iraq
Iran
Afghanistan
Pakistan
Nepal
India
Cina
Corea del Sud
Giappone
Myanmar (Birmania)
Thailandia
Vietnam
Filippine
Malesia
Indonesia
Australia
Papua Nuova Guinea
Marocco
Algeria
Tunisia
Libia
Egitto
Sahara Occidentale
Mauritania
Mali
Niger
Ciad
Sudan
Nigeria
Ghana
Burkina Faso
Guinea
Etiopia
Somalia
Kenya
Gabon
Repubblica Democratica del Congo
Tanzania
Angola
Zambia
Mozambico
Namibia
Zimbabwe
Botswana
Madagascar
Sudafrica
Arabia Saudita
Oman
Yemen
Oceano Atlantico settentrionale
Oceano Pacifico settentrionale
Oceano Pacifico meridionale
Oceano Atlantico meridionale
Oceano Indiano
Mappa
Satellite

wspr.rocks


German Research Icebreaker Polarstern
POLARSTERN
DPØPOL /mm
German Amateur Radio Station


Irlanda
Bielorussia
Ucraina
Kazakistan
Mongolia
Romania
Kirghizistan
Turchia
Turkmenistan
Siria
Iraq
Iran
Afghanistan
Pakistan
Cina
Corea del Sud
Giappone
Nepal
India
Myanmar (Birmania)
Thailandia
Vietnam
Filippine
Malesia
Indonesia
Papua Nuova Guinea
Australia
Nuova Zelanda
Tunisia
Algeria
Libia
Egitto
Arabia Saudita
Oman
Yemen
Mauritania
Mali
Niger
Ciad
Sudan
Burkina Faso
Guinea
Nigeria
Ghana
Sudan del Sud
Etiopia
Somalia
Kenya
Repubblica Democratica del Congo
Tanzania
Angola
Zambia
Mozambico
Zimbabwe
Namibia
Botswana
Madagascar
Sudafrica
Messico
Cuba
Porto Rico
Guatemala
Nicaragua
Venezuela
Guyana
Suriname
Colombia
Ecuador
Perù
Brasile
Bolivia
Paraguay
Cile
Uruguay
Argentina
Oceano Atlantico Meridionale
Oceano Indiano
Mare del Labrador
Mare di Okhotsk
Mare del Giappone (Mare dell'Est)
Mar Cinese Orientale
Mar Cinese Meridionale
Mare delle Filippine
Golfo del Bengala
Mar Arabico
Golfo di Aden
Golfo del Siam
Mare delle Laccadive
Mar di Banda
Mare degli Arafura
Mare delle Salomone
Mar dei Coralli
Mar di Tasman
Grande Baia Australiana
950


HanRun
HR911105A
2405
USB-OTG
DEVICE
HOST
S1
S2
S3
S4



A Darlington amplifier is a 2-port device: RF input, and combined RF output and bias input. It is housed in a 4-lead package including 2 ground leads; connecting both of them to external ground will minimize common path impedance for best RF performance. Internal resistors in Figure 1 determine the DC operating point of the transistors and provide feedback to set RF gain, bandwidth, and input and output impedances to optimum values.

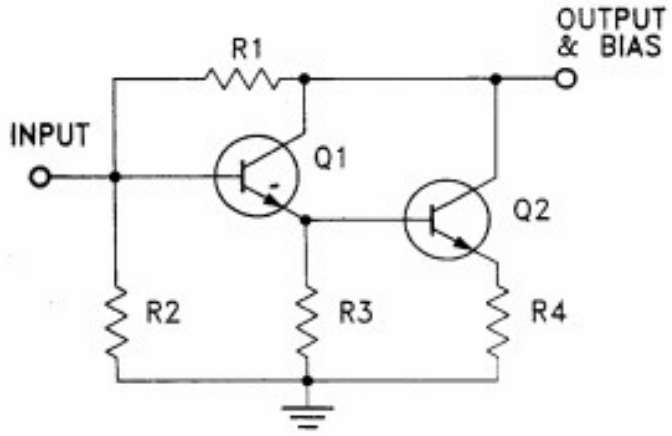
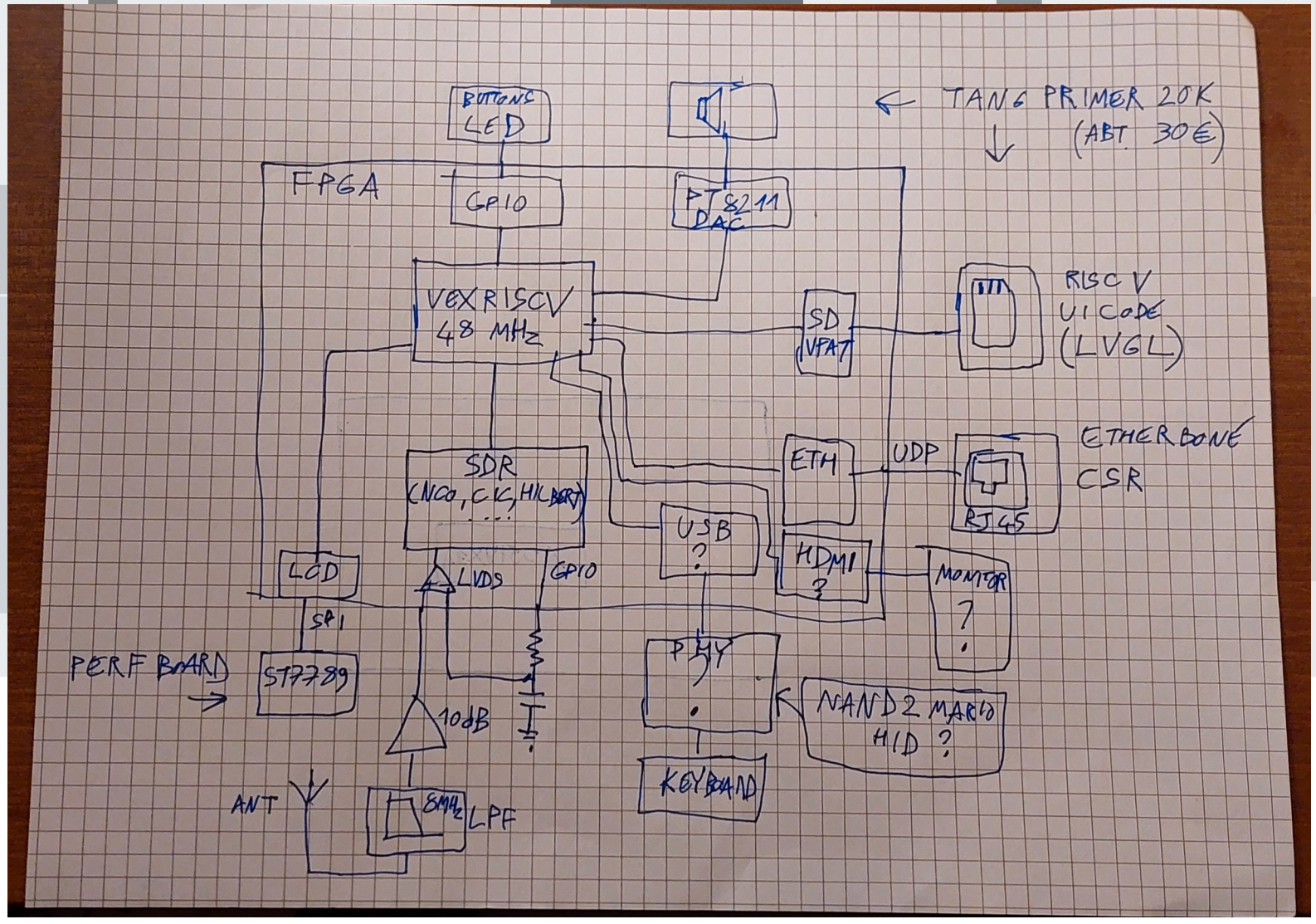
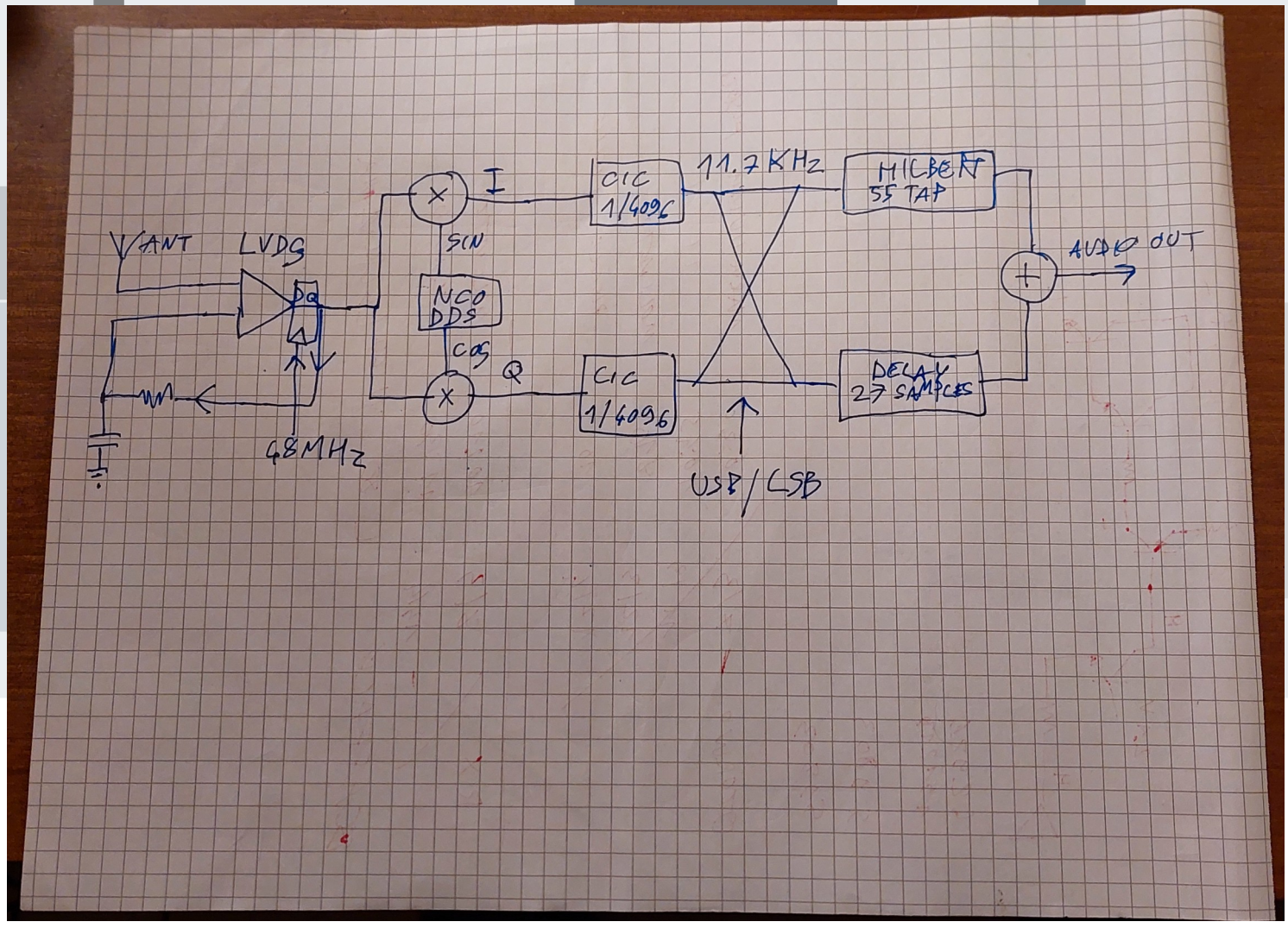
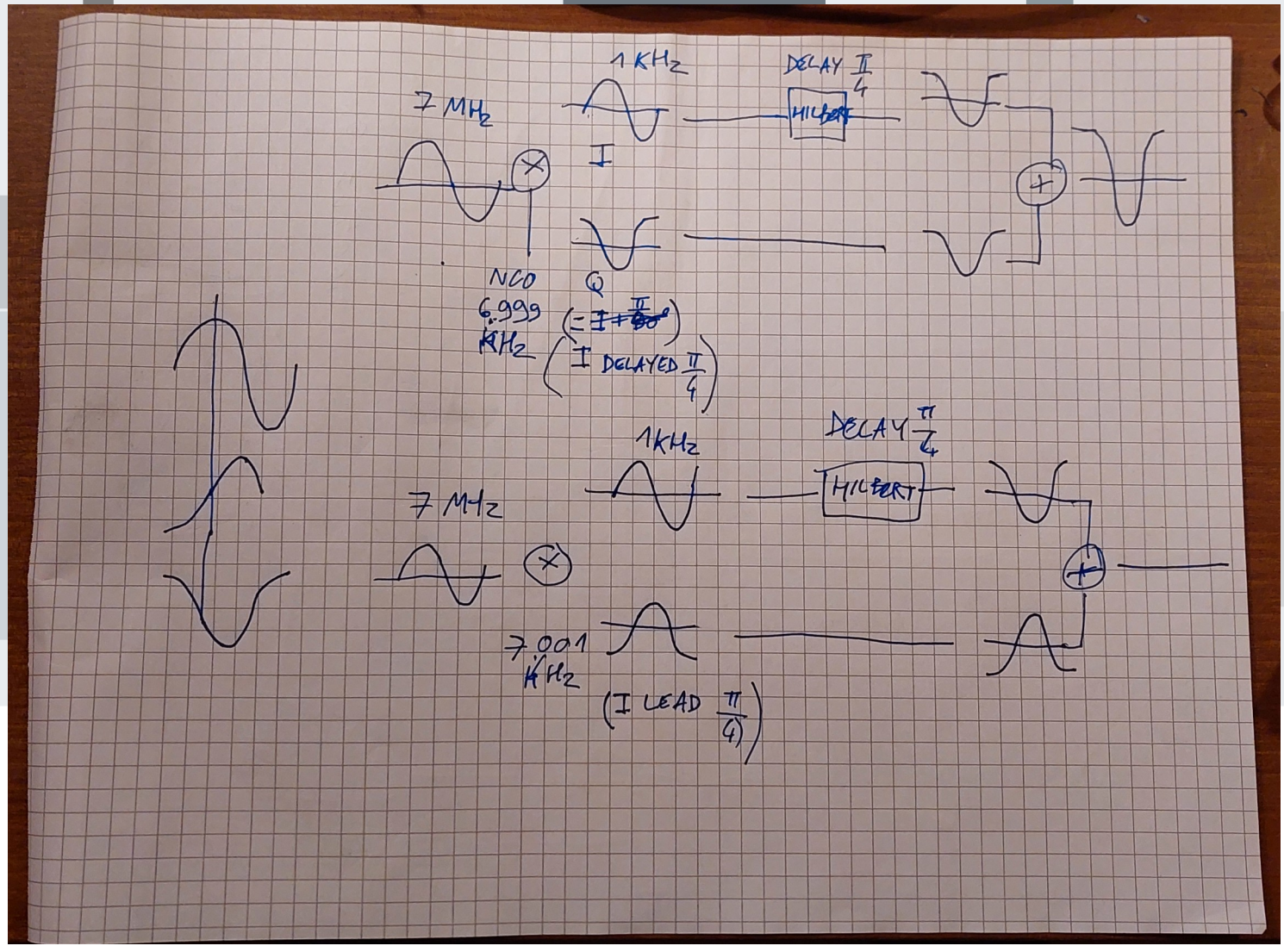
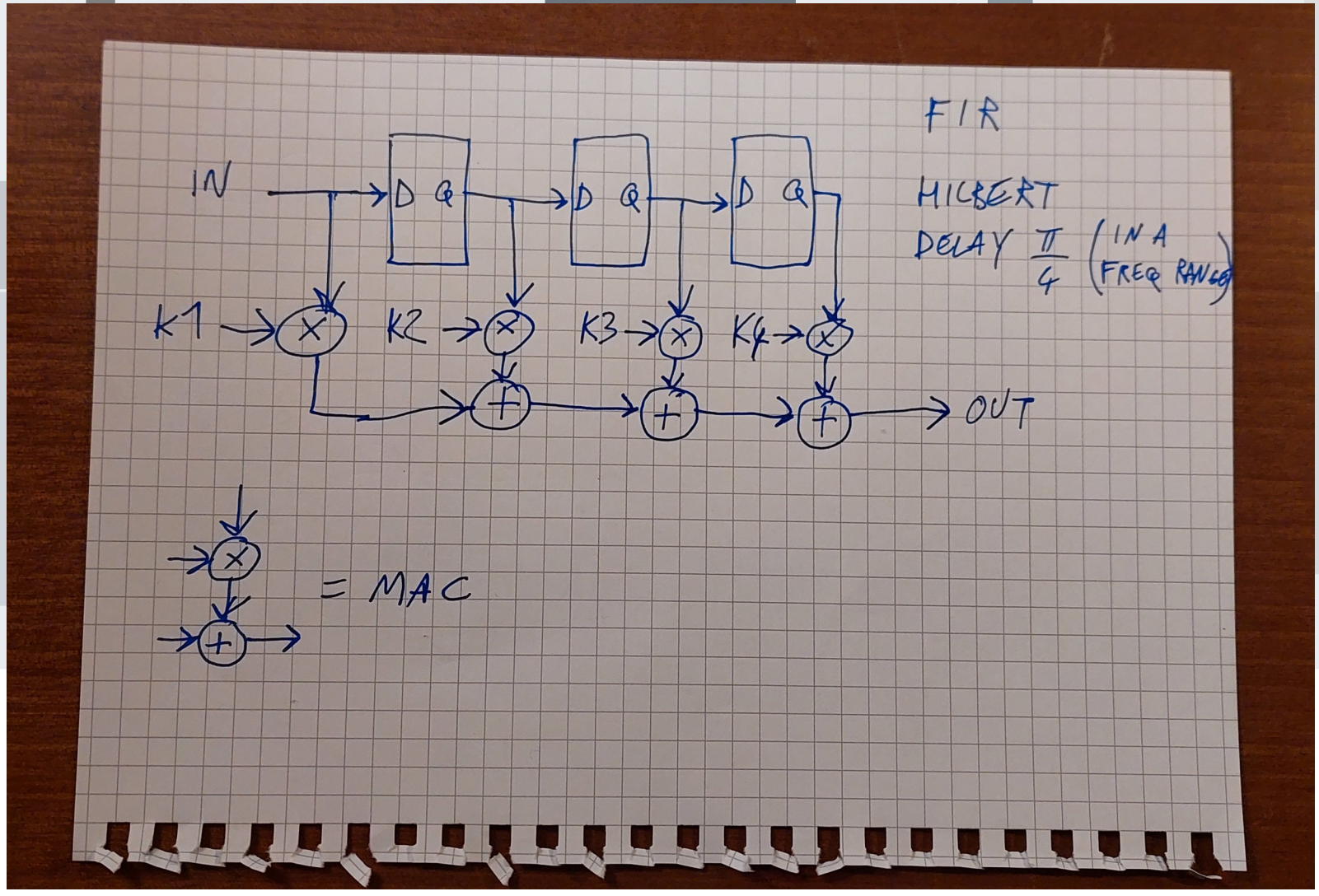


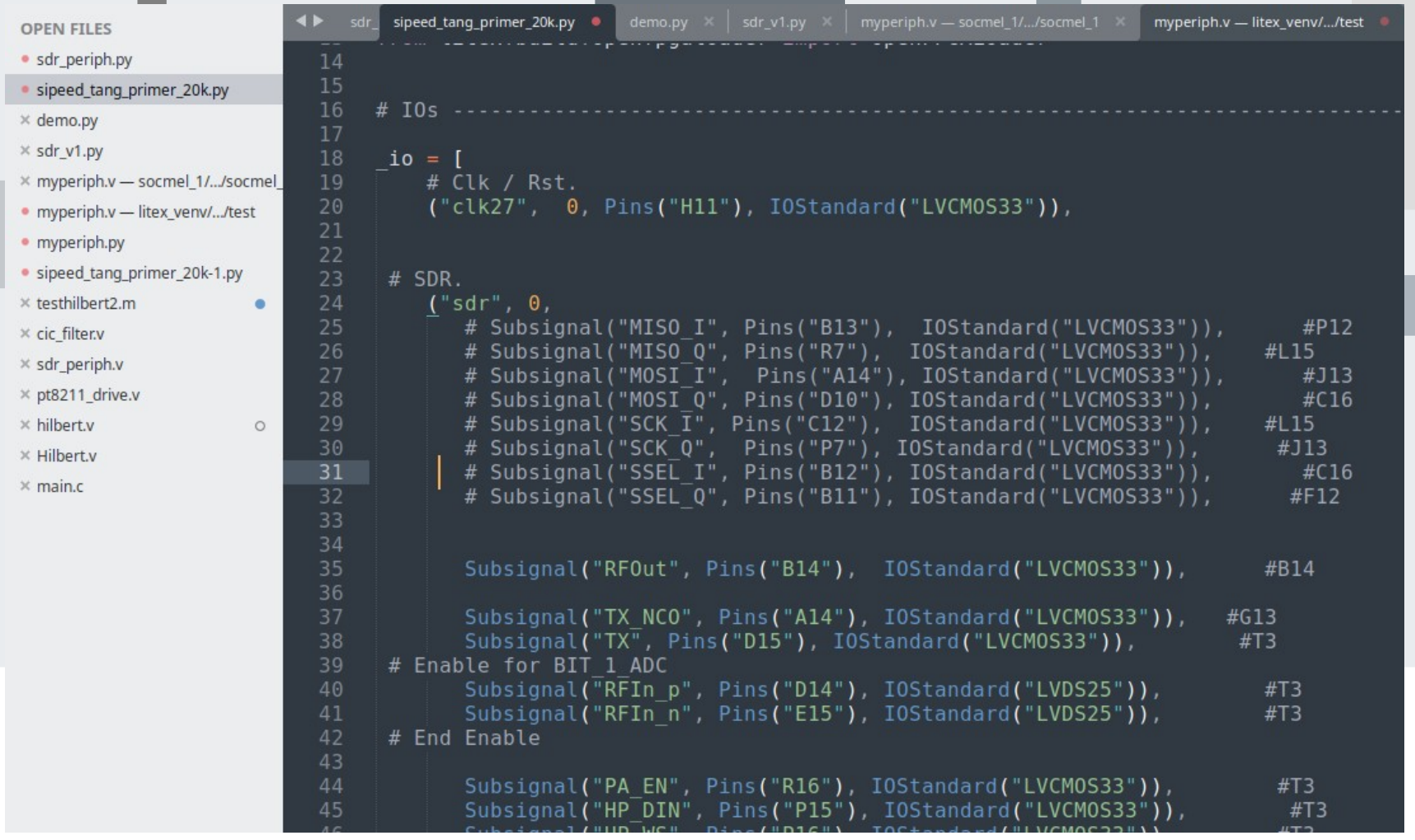
Figure 1 Schematic Diagram


BUTTONS LED
TANG PRIMER 20K
(ABT. 30€)
FPGA
GPIO
PT8211 DAC
VEXRISCV 48 MHz
SD VFAT
RISC V UICODE (LVGL)
SDR (NCO, CIC, HILBERT ...)
ETH
UDP
ETHERBONE CSR
RJ45
USB ?
HDMI ?
MONITOR ?
LCD
LVDS
GPIO
SPI
PERF BOARD
ST7789
10dB
PHY
NAND2 MARIO HID ?
KEYBOARD
ANT
8MHz
LPF


VANT
LVDS
SIN
COS
NCO DDS
I
Q
CIC 1/4096
CIC 1/4096
11.7 KHz
HILBERT 55 TAP
DELAY 27 SAMPLES
AUDIO OUT
48MHz
USB/LSB


7 MHz
1 KHz
DELAY π/4
HILBERT
I
NCO
6.999
KHz
Q
(-I + 90°)
(I DELAYED π/4)
1KHz
DELAY π/4
HILBERT
7 MHz
7.001
KHz
(I LEAD π/4)


FIR
IN
D Q
D Q
D Q
HILBERT
DELAY π/4 (IN A FREQ RANGE)
K1
K2
K3
K4
OUT
= MAC


OPEN FILES
sdr_periph.py
sipeed_tang_primer_20k.py
demo.py
sdr_v1.py
myperiph.v — socmel_1/.../socmel_
myperiph.v — litex_venv/.../test
myperiph.py
sipeed_tang_primer_20k-1.py
testhilbert2.m
cic_filter.v
sdr_periph.v
pt8211_drive.v
hilbert.v
Hilbert.v
main.c
sdr_
sipeed_tang_primer_20k.py
demo.py
sdr_v1.py
myperiph.v — socmel_1/.../socmel_1
myperiph.v — litex_venv/.../test
14
15
16 # IOs ----------
17
18 _io = [
19 # Clk / Rst.
20 ("clk27", 0, Pins("H11"), IOStandard("LVCMOS33")),
21
22
23 # SDR.
24 ("sdr", 0,
25 # Subsignal("MISO_I", Pins("B13"), IOStandard("LVCMOS33")), #P12
26 # Subsignal("MISO_Q", Pins("R7"), IOStandard("LVCMOS33")), #L15
27 # Subsignal("MOSI_I", Pins("A14"), IOStandard("LVCMOS33")), #J13
28 # Subsignal("MOSI_Q", Pins("D10"), IOStandard("LVCMOS33")), #C16
29 # Subsignal("SCK_I", Pins("C12"), IOStandard("LVCMOS33")), #L15
30 # Subsignal("SCK_Q", Pins("P7"), IOStandard("LVCMOS33")), #J13
31 # Subsignal("SSEL_I", Pins("B12"), IOStandard("LVCMOS33")), #C16
32 # Subsignal("SSEL_Q", Pins("B11"), IOStandard("LVCMOS33")), #F12
33
34
35 Subsignal("RFOut", Pins("B14"), IOStandard("LVCMOS33")), #B14
36
37 Subsignal("TX_NCO", Pins("A14"), IOStandard("LVCMOS33")), #G13
38 Subsignal("TX", Pins("D15"), IOStandard("LVCMOS33")), #T3
39 # Enable for BIT_1_ADC
40 Subsignal("RFIn_p", Pins("D14"), IOStandard("LVDS25")), #T3
41 Subsignal("RFIn_n", Pins("E15"), IOStandard("LVDS25")), #T3
42 # End Enable
43
44 Subsignal("PA_EN", Pins("R16"), IOStandard("LVCMOS33")), #T3
45 Subsignal("HP_DIN", Pins("P15"), IOStandard("LVCMOS33")), #T3

```
        platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/sdr_periph.v")
  #      platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/CIC.v")
        platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/Hilbert.v")
        platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/NCO.v")
        platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/pt8211_drive.v")
 # Non Abilitare      platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/PWM.v")
        platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/SinCos.v")
 # Non Abilitare     platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/TX_SPI.v")
 #       platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/gowin_mult.v")
 #       platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/gowin_mult1.v")
        platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/dds_ii.v")
        platform.add_source("/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/sdr_periph/verilog/cic_filter.v")
        self.submodules.sdr_periph = SDRPeriph(platform)
        self.add_csr("sdr_periph")


# Build --------------------------------------------------------------------------------------------

def main():
    from litex.build.parser import LiteXArgumentParser
    parser = LiteXArgumentParser(platform=sipeed_tang_primer_20k.Platform, description="LiteX SoC on Tang Primer 20K.")
    parser.add_target_argument("--dock",         default="standard",       help="Dock version (standard (default) or lite.")
    parser.add_target_argument("--flash",        action="store_true",      help="Flash Bitstream.")
    parser.add_target_argument("--sys-clk-freq", default=48e6, type=float, help="System clock frequency.")
    sdopts = parser.target_group.add_mutually_exclusive_group()
    sdopts.add_argument("--with-spi-sdcard",            action="store_true", help="Enable SPI-mode SDCard support.")
    sdopts.add_argument("--with-sdcard",                action="store_true", help="Enable SDCard support.")
    parser.add_target_argument("--with-spi-flash",      action="store_true", help="Enable SPI Flash (MMAPed).")
    parser.add_target_argument("--with-video-terminal", action="store_true", help="Enable Video Terminal (HDMI).")
    ethopts = parser.target_group.add_mutually_exclusive_group()
    ethopts.add_argument("--with-ethernet",        action="store_true",     help="Add Ethernet.")
    ethopts.add_argument("--with-etherbone",       action="store_true",     help="Add EtherBone.")
    parser.add_target_argument("--eth-ip",         default="192.168.1.50",  help="Etherbone IP address.")
    parser.add_target_argument("--remote-ip",      default="192.168.1.100", help="Remote IP address of TFTP server.")
    parser.add_target_argument("--eth-dynamic-ip", action="store_true",     help="Enable dynamic Ethernet IP addresses setting.")
```

```
(litex_venv) alberto@alberto-ThinkPad-X230-2:~/litex_venv/litex-boards/litex_boards/targets/socmel_1$ ./sipeed_tang_primer_20k_soc_socmel_1.py
--sys-clk-freq=48e6 --soc-csv=soc.csf --with-etherbone --cpu-type=vexriscv --cpu-variant=lite --l2-size=512 --with-spi-sdcard --eth-ip 192.168
11.50 --build --csr-csv build/sipeed_tang_primer_20k/csr.csv --doc
>>> XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXUsing custom platform with dock = standard
INFO:GW1NPLL:Creating GW1NPLL.
INFO:GW1NPLL:Registering Single Ended ClkIn of 27.00MHz.
INFO:GW1NPLL:Creating ClkOut0 sys2x_i of 96.00MHz (+-10000.00ppm).
INFO:SoC:
INFO:SoC:       __   _ __      _ __
INFO:SoC:      / /  (_) /____ | |/_/
INFO:SoC:     / /__/ / __/ -_)>  <
INFO:SoC:    /____/_/\__/\__/_/|_|
INFO:SoC:  Build your hardware, easily!
INFO:SoC:--------------------------------------------------------------------------------
INFO:SoC:Creating SoC... (2025-12-18 16:38:11)
INFO:SoC:--------------------------------------------------------------------------------
INFO:SoC:FPGA device : GW2A-LV18PG256C8/I7.
INFO:SoC:System clock: 48.000MHz.
INFO:SoCBusHandler:Creating Bus Handler...
INFO:SoCBusHandler:32-bit wishbone Bus, 4.0GiB Address Space.
INFO:SoCBusHandler:Adding reserved Bus Regions...
INFO:SoCBusHandler:Bus Handler created.
INFO:SoCCSRHandler:Creating CSR Handler...
INFO:SoCCSRHandler:32-bit CSR Bus, 32-bit Aligned, 16.0KiB Address Space, 2048B Paging, big Ordering (Up to 32 Locations).
INFO:SoCCSRHandler:Adding reserved CSRs...
INFO:SoCCSRHandler:CSR Handler created.
INFO:SoCIRQHandler:Creating IRQ Handler...
INFO:SoCIRQHandler:IRQ Handler (up to 32 Locations).
INFO:SoCIRQHandler:Adding reserved IRQs...
INFO:SoCIRQHandler:IRQ Handler created.
INFO:SoC:--------------------------------------------------------------------------------
INFO:SoC:Initial SoC:
INFO:SoC:--------------------------------------------------------------------------------
INFO:SoC:32-bit wishbone Bus, 4.0GiB Address Space.
INFO:SoC:32-bit CSR Bus, 32-bit Aligned, 16.0KiB Address Space, 2048B Paging, big Ordering (Up to 32 Locations).
INFO:SoC:IRQ Handler (up to 32 Locations).
INFO:SoC:--------------------------------------------------------------------------------
INFO:SoC:Controller ctrl added.
INFO:SoC:CPU vexriscv added.
INFO:SoC:CPU vexriscv adding IO Region 0 at 0x80000000 (Size: 0x80000000).
INFO:SoCBusHandler:io0 Region added at Origin: 0x80000000, Size: 0x80000000, Mode:  RW, Cached: False, Linker: False.
INFO:SoC:CPU vexriscv overriding sram mapping from 0x01000000 to 0x10000000.
INFO:SoC:CPU vexriscv setting reset address to 0x00000000.
INFO:SoC:CPU vexriscv adding Bus Master(s).
```

```
//--------------------------------------------------------------------------------
// Auto-generated by LiteX (26076b3e2) on 2025-12-18 16:38:12
//--------------------------------------------------------------------------------

//--------------------------------------------------------------------------------
// CSR Includes.
//--------------------------------------------------------------------------------

#include <generated/soc.h>
#ifndef __GENERATED_CSR_H
#define __GENERATED_CSR_H
#include <stdint.h>
#include <system.h>
#ifndef CSR_ACCESSORS_DEFINED
#include <hw/common.h>
#endif /* ! CSR_ACCESSORS_DEFINED */

#ifndef CSR_BASE
#define CSR_BASE 0xf0000000L
#endif /* ! CSR_BASE */

//--------------------------------------------------------------------------------
// CSR Registers/Fields Definition.
//--------------------------------------------------------------------------------

/* MYPERIPH Registers */
#define CSR_MYPERIPH_BASE (CSR_BASE + 0x0L)
#define CSR_MYPERIPH_CSR0_ADDR (CSR_BASE + 0x0L)
#define CSR_MYPERIPH_CSR0_SIZE 1
#define CSR_MYPERIPH_CSR1_ADDR (CSR_BASE + 0x4L)
#define CSR_MYPERIPH_CSR1_SIZE 1
#define CSR_MYPERIPH_CSR2_ADDR (CSR_BASE + 0x8L)
#define CSR_MYPERIPH_CSR2_SIZE 1
#define CSR_MYPERIPH_CSR3_ADDR (CSR_BASE + 0xcL)
#define CSR_MYPERIPH_CSR3_SIZE 1
#define CSR_MYPERIPH_CSR_R_ADDR (CSR_BASE + 0x10L)
#define CSR_MYPERIPH_CSR_R_SIZE 4

/* MYPERIPH Fields */

/* SDR_PERIPH Registers */
#define CSR_SDR_PERIPH_BASE (CSR_BASE + 0x800L)
#define CSR_SDR_PERIPH_CSR0_ADDR (CSR_BASE + 0x800L)
#define CSR_SDR_PERIPH_CSR0_SIZE 1
#define CSR_SDR_PERIPH_CSR1_ADDR (CSR_BASE + 0x804L)
#define CSR_SDR_PERIPH_CSR1_SIZE 1
#define CSR_SDR_PERIPH_CSR2_ADDR (CSR_BASE + 0x808L)
#define CSR_SDR_PERIPH_CSR2_SIZE 1
#define CSR_SDR_PERIPH_CSR3_ADDR (CSR_BASE + 0x80cL)
#define CSR_SDR_PERIPH_CSR3_SIZE 1
#define CSR_SDR_PERIPH_CSR_R_ADDR (CSR_BASE + 0x810L)
#define CSR_SDR_PERIPH_CSR_R_SIZE 4

```

```
275 #define CSR_UART_EV_ENABLE_RX_OFFSET 1
276 #define CSR_UART_EV_ENABLE_RX_SIZE 1
277
278 //--------------------------------------------------------------------------------
279 // CSR Registers Access Functions.
280 //--------------------------------------------------------------------------------
281
282 #ifndef LITEX_CSR_ACCESS_FUNCTIONS
283 #define LITEX_CSR_ACCESS_FUNCTIONS 1
284 #endif
285
286 #if LITEX_CSR_ACCESS_FUNCTIONS
287
288 /* MYPERIPH Access Functions */
289 static inline uint32_t myperiph_csr0_read(void) {
290         return csr_read_simple((CSR_BASE + 0x0L));
291 }
292 static inline void myperiph_csr0_write(uint32_t v) {
293         csr_write_simple(v, (CSR_BASE + 0x0L));
294 }
295 static inline uint32_t myperiph_csr1_read(void) {
296         return csr_read_simple((CSR_BASE + 0x4L));
297 }
298 static inline void myperiph_csr1_write(uint32_t v) {
299         csr_write_simple(v, (CSR_BASE + 0x4L));
300 }
301 static inline uint32_t myperiph_csr2_read(void) {
302         return csr_read_simple((CSR_BASE + 0x8L));
303 }
304 static inline void myperiph_csr2_write(uint32_t v) {
305         csr_write_simple(v, (CSR_BASE + 0x8L));
306 }
307 static inline uint32_t myperiph_csr3_read(void) {
308         return csr_read_simple((CSR_BASE + 0xcL));
309 }
310 static inline void myperiph_csr3_write(uint32_t v) {
311         csr_write_simple(v, (CSR_BASE + 0xcL));
312 }
```

```python
#!/usr/bin/env python3

import time


from litex import RemoteClient

wb = RemoteClient(csr_csv="/home/alberto/litex_venv/litex-boards/litex_boards/targets/test/build/sipeed_tang_primer_20k/csr.csv" )
wb.open()

# Dump all CSR registers of the SoC
#for name, reg in wb.regs.__dict__.items():
#    print("0x{:08x} : 0x{:08x} {}".format(reg.addr, reg.read(), name))

# For some reason registers need a dummy write at start
wb.write(0xF0000800, 0x0)
wb.write(0xF0000804, 0x1)
wb.write(0xF0000808, 0x2)
wb.write(0xF000080c, 0x3)

wb.write(0xF0000010, 0xa5a5a5a5)
wb.write(0xF0000014, 0xa5a5a5a5)
wb.write(0xF0000018, 0xa5a5a5a5)
wb.write(0xF000001c, 0x5a5a5a5a)

Frequency = 7072000
NCOstep = ((pow(2,64) * Frequency // 48000000))
NCOstepHi = NCOstep >> 32
NCOstepLo = NCOstep & 0x00000000ffffffff



print ("NCOstep ", hex(NCOstep))
print ("NCOstepHi ", hex(NCOstepHi))
print ("NCOstepLo ", hex(NCOstepLo))
wb.write(0xF0000000, NCOstepLo)
wb.write(0xF0000004, NCOstepHi)
wb.write(0xF0000008, 0xa5a5a5a5)
wb.write(0xF000000c, 0x5a5a5a5a)
```

```
    static uint32_t prev_NCOFrequency = 0;
    if (NCOFrequency != prev_NCOFrequency) {
        NCOIncrement = (pow(2,64) * NCOFrequency) / SAMPLE_FREQUENCY;
        NCOIncrementHi = NCOIncrement >> 32;
        NCOIncrementLo = NCOIncrement & 0x00000000ffffffff;

        sdr_periph_csr0_write(NCOIncrementLo);
        sdr_periph_csr1_write(NCOIncrementHi);

        prev_NCOFrequency = NCOFrequency;
    }
    if (DesiredSideband != prev_DesiredSideband) {
     sdr_periph_csr2_write(DesiredSideband + ((AudioVol & 0xf) << 1));
     prev_DesiredSideband = DesiredSideband;
}
```

```c
void ui_init(lv_display_t *disp) {

        /* set screen background to black */
    lv_obj_t *scr = lv_screen_active();
    lv_obj_set_style_bg_color(scr, lv_color_hex(0x000000), 0);
    lv_obj_set_style_bg_opa(scr, LV_OPA_100, 0);


    LV_IMG_DECLARE(army_panel);
    lv_obj_t * img1 = lv_image_create(lv_scr_act());
    lv_img_set_src(img1, &army_panel);
    lv_obj_align(img1, LV_ALIGN_CENTER, 0, 0);
    lv_obj_set_size(img1, 240, 135);

        // create label
    LV_FONT_DECLARE(lv_font_scoreboard_32 );
    label = lv_label_create(scr);
    lv_obj_set_align(label, LV_ALIGN_OUT_TOP_LEFT);
    lv_obj_set_height(label, LV_SIZE_CONTENT);
    lv_obj_set_width(label, LV_SIZE_CONTENT);
lv_obj_set_pos(label, 38 , 28);    // Or in one function
//lv_obj_set_style_text_font(label, &lv_font_montserrat_32, 0);

lv_obj_set_style_text_font(label, &lv_font_scoreboard_32 , 0);
lv_obj_set_style_text_color(label, lv_color_hex(0xffbf00), 0);
lv_label_set_recolor(label, true);                      //Enable re-coloring by commands in the text
lv_label_set_text(label, "Hello World!");

label2  = lv_label_create(lv_screen_active());
lv_label_set_long_mode(label2, LV_LABEL_LONG_MODE_SCROLL_CIRCULAR);     //Circular scroll
lv_obj_set_width(label2, 100);
lv_label_set_text(label2, "START VALUE ");
lv_obj_set_pos(label2, 42 , 78);
lv_obj_set_style_text_color(label2, lv_color_hex(0xffbf00), 0);
lv_obj_set_style_text_font(label2, &lv_font_scoreboard_32, 0);
lv_obj_set_style_anim_duration(label2, 4000, LV_PART_MAIN);

bar1 = lv_bar_create(lv_screen_active());
lv_obj_set_size(bar1, 38, 12);
lv_obj_set_align(label, LV_ALIGN_OUT_TOP_LEFT);
lv_obj_set_pos(bar1, 168 , 88);
lv_obj_set_style_bg_color(bar1, lv_color_hex(0xffbf00), LV_PART_INDICATOR);
lv_obj_set_style_bg_color(bar1, lv_color_hex(0x181713), LV_PART_MAIN);
lv_bar_set_value(bar1, bar1_value, LV_ANIM_OFF);
```

```
54
55  //This array should be provided by libgcc.a, but for some reason it is not found
56  const uint8_t __clz_tab[256] =
57  {
58    0,1,2,2,3,3,3,3,4,4,4,4,4,4,4,4,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,
59    6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,
60    7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,
61    7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,
62    8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,
63    8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,
64    8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,
65    8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8
66  };
67
```

ANALOG DEVICES

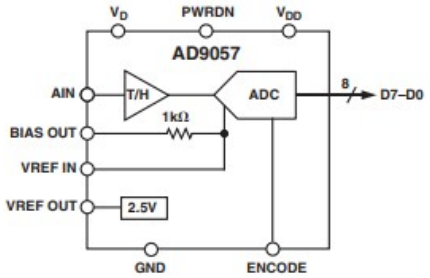
8-Bit 40 MSPS/60 MSPS/80 MSPS A/D Converter

AD9057

FEATURES
8-Bit, Low Power ADC: 200 mW Typical
120 MHz Analog Bandwidth
On-Chip 2.5 V Reference and Track-and-Hold
1 V p-p Analog Input Range
Single 5 V Supply Operation
5 V or 3 V Logic Interface
Power-Down Mode: <10 mW
3 Performance Grades (40 MSPS, 60 MSPS, 80 MSPS)

APPLICATIONS
Digital Communications (QAM Demodulators)
RGB and YC/Composite Video Processing
Digital Data Storage Read Channels
Medical Imaging
Digital Instrumentation

FUNCTIONAL BLOCK DIAGRAM



PRODUCT DESCRIPTION

The AD9057 is an 8-bit monolithic analog-to-digital converter optimized for low cost, low power, small size, and ease of use. With 40 MSPS, 60 MSPS, or 80 MSPS encode rate capability and full-power analog bandwidth of 120 MHz, the component is ideal for applications requiring excellent dynamic performance.

To minimize system cost and power dissipation, the AD9057 includes an internal 2.5 V reference and a track-and-hold (T/H) circuit. The user must provide only a 5 V power supply and an

power-down function may be exercised to bring total consumption to <10 mW. In power-down mode, the digital outputs are driven to a high impedance state.

Fabricated on an advanced BiCMOS process, the AD9057 is available in a space-saving 20-lead shrink small outline package (20-lead SSOP) and is specified over the industrial temperature range (–40°C to +85°C).

Customers desiring multichannel digitization may consider the AD9059, a dual 8-bit, 60 MSPS monolithic based on the

A better ADC...

https://github.com/alberto-grl

https://
www.youtube.com/
watch?v=JL7FPLmdea0