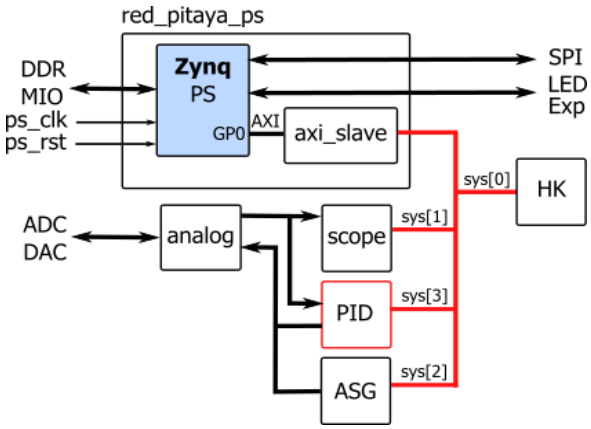
3. in 4. vaja: generator signalov

V projekt logičnega vezja na Red Pitayi bomo dodali komponento za skaliranje in generiranje signala.

3.1 Red Pitaya Classic https://lniv.fe.uni-lj.si/xilinx/redpitaya-classic.htm

Digitalni sistem na Red Pitayi z imenom *classic* vsebuje osciloskop (scope), signalni generator (ASG) in regulator PID. Glavna komponenta v jeziku System Verilog povezuje procesor (PS) z moduli po sistemskem vodilu (sys), kjer je naslovni prostor razdeljen na posamezne segmente.

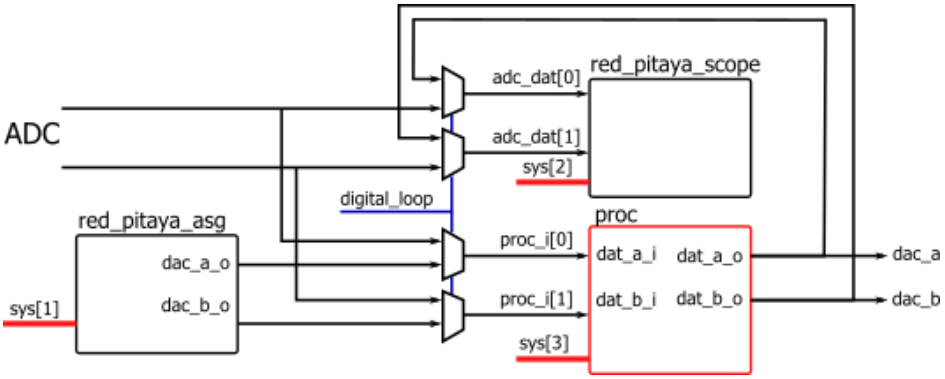
signal	naslov bloka	enota (komponenta)
CS[0]	0x40000000	Housekeeping (id)
CS[1]	0x40100000	Oscilloscope (scope)
CS[2]	0x40200000	Arbitrary sig. generator (asg)
CS[3]	0x40300000	PID controller (pid)
CS[4]	0x40400000	Analog mixed signals (ams)



Modul Housekeeping (HK) skrbi za osnovne parametre. Npr. nastavitev *Digital Loopback* je na naslovu: 0x4000000C

3.2 Skaliranje signalov

1. Uporabi arhiv **redpitaya-proc.zip** s projektom *classic* in komponento *proc* za obdelavo signalov.



   Odpri **Vivado 2019**, v konzoli nastavi projektno mapo in izvedi skripto za izdelavo projekta:

```
cd c:/proj/ime/redpitaya
source ./make_project.tcl
```

2. Dopolni komponento *proc*, ki množi vhodne vrednosti z 8-bitno amplitudo v formatu 4.4. Dodaj logiko za nasičenje izhoda ob prekoračitvi območja (-8192…8191). Za preizkus delovanja vključi v projekt testno strukturo in naredi simulacijo.
   - V projekt vključi testno strukturo kot simulacijski vir. V zavihku Sources, Simulation Sources pa nastavi testProc kot glavno komponento (Set as Top).
   - Na časovnem diagramu nastavi analogni prikaz signalov **dat_a_i** in **dat_a_o** (Radix: Signed, Waveform style: Analog).

3. Prevedi projekt in preizkusi delovanje na razvojni plošči STEMlab.
   - Prenesi konfig. datoteko (*.bit) in izvedi skripto za povezavo s spletno aplikacijo

```
./nastavi.sh red_pitaya_top.bit
```

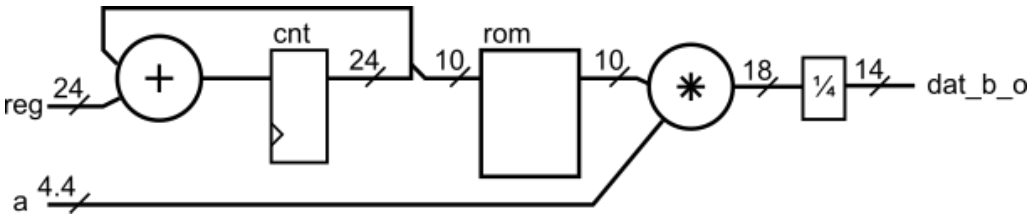
   - Odpri aplikacijo *Oscilloscope* in preizkusi z nastavljanjem registrov v konzoli terminala.

```
monitor 0x4000000c 1
monitor 0x40300054 20
```

4.1 Numerično krmiljen oscilator

Komponenti proc bomo dodali generator sinusnega signala z izhodom na drugem kanalu.

1. Dodaj v ops komponente 24-bitni register **reg**, ki ga pišemo in beremo na naslovu x"00060" in 24-bitni števec **cnt**, ki se ob fronti ure povečuje za vrednost **reg**.
2. Dodaj še komponento (rom.vhd) s sinusno tabelo in deklariraj ustrezne povezovalne signele. Naslov poveži s števcem (uporabi zgornjih 10 bitov), podatek pa pomnoži z 8-bitno amplitudo in poveži na izhod **dat_b_o**.



VHDL predznačeno množenje

```
mul_b <= signed(rom_data) *
         signed('0' & a);
```

Predznačeno množenje v Verilogu

```
wire        [ 10-1: 0] rom_adr ;
reg signed  [  4-1: 0] amp ;
wire signed [ 10-1: 0] rom_data ;

assign dat_o = rom_data * $signed({1'b0, amp});
```

3. Preveri delovanje oscilatorja s simulacijo. Uporabi testno strukturo, ki ji dodaj kodo za nastavitev registra na naslovu 60, npr.

```
-- nastavi reg
addr_i  <= x"00000060";
wdata_i <= x"00100000";
wen_i <= '1';
wait for T;
```

4.2 Preizkus na razvojni plošči

4. Naredi sintezo, implementacijo in prenesi konfiguracijsko datoteko na razvojno ploščo. Ponovno odpri aplikacijo osciloskop, da bo naložena nova datoteka.
5. V terminalu s programom monitor nastavi vrednosti registrov na naslovih:

```
monitor 0x4000000c 1
monitor 0x40300054 20
monitor 0x40300060 100
```

Razmisli

- V kakšnem obsegu lahko nastavljamo amplitudo in frekvenco?
- Napiši enačbo za pretvorbo željene frekvence v ustrezno nastavitev registra.