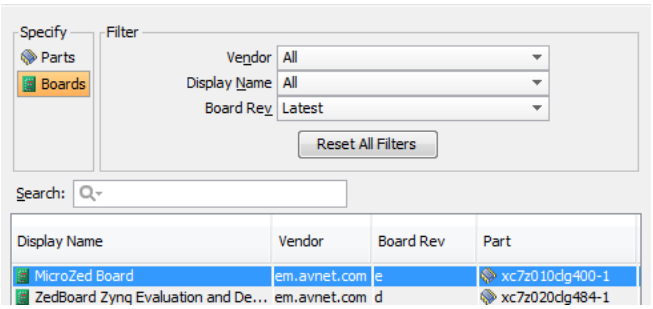
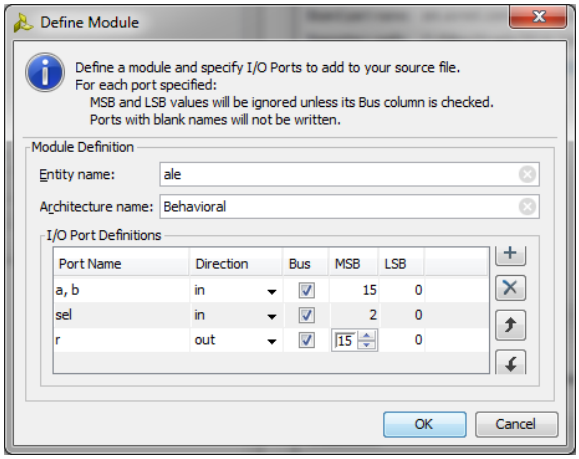
Načrtovanje vezja v programu Vivado

Za načrtovanje digitalnih vezij bomo uporabljali razvojno orodje Xilinx Vivado 2014.2 WebPACK, ki je brezplačno dostopno na http://www.xilinx.com/support/download.html

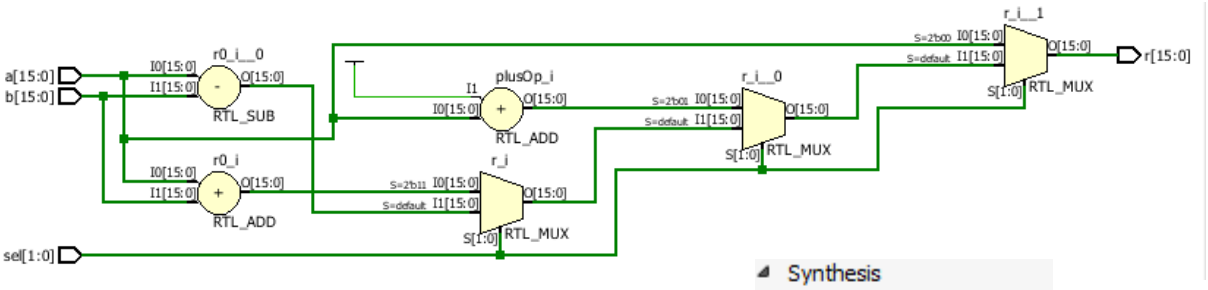
Najprej naredimo nov projekt **Create New Project**, določimo ime (npr. vaja1) in lokacijo, ki naj bo na D:\des\moja mapa. V naslednjem oknu izberemo **RTL Project**, nato pa v naslednjem oknu kliknemo na **Create File** ter določimo ime datoteke (npr. ALE) in vrsto: VHDL. V zadnjem oknu izberemo FPGA vezje ali ploščo. Izbrali bomo razvojno ploščo *MicroZed*, ki ima enako vezje kot Red Pitaya.



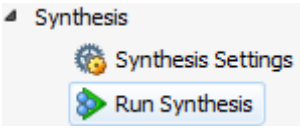


Pred zaključkom se pojavi še okno v katerem določimo vhodne in izhodne signale nove VHDL datoteke.

Program je na podlagi vnešenih podatkov naredil datoteko z ogrodjem opisa vezja v jeziku VDHL. Datoteko dopolnimo s stavki, ki opisujejo delovanje vezja. Ko shranimo spremembe (ctrl+s) se avtomatsko preverijo osnovna sintaktična pravila in v primeru napake dobimo sporočila v zavihku **Messages**. Program ob shranjevanju naredi elaboracijo opisa vezja in če ni napak lahko RTL strukturo vezja pogledamo z izbiro iz menija **Flow, Open Elaborated Design**:



Sintezo digitalnega vezja izvedemo s klikom na Run Synthesis:



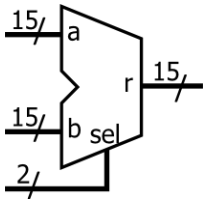
Po sintezi izberemo v oknu **View Reports** in si ogledamo poročila: **Vivado Synthesis Report** in **Utilization Report**. V prvem poročilu vidimo število in vrsto RTL gradnikov in število celic v vezju, v drugem poročilu pa zasedenost FPGA matrike (npr. število uporabljenih vpoglednih tabel LUT).

Simulacijo poženemo z ikono **Run Simulation** in izbiro **Run Behavioral Simulation**. V simulatorju nastavimo vrednosti vhodnim signalom (**Force Constant** ali **Clock**) in opazujemo izhode.

1. vaja: Kombinacijska vezja

Kombinacijska vezja so digitalna vezja, pri katerih je izhodna vrednost odvisna le od trenutnega vhoda. V jeziku VHDL opišemo operacije nad signali z aritmetičnimi in logičnimi operatorji (**+, -, *, and, or, not, nand, nor, xor**), dekodiranje pa s pogojnimi in izbirnimi stavki, pri katerih je izhod definiran ob vseh pogojih.

1.1 Aritmetična enota



Naredi 16-bitno aritmetično enoto z dvobitnim izbirnim signalom **sel**, ki določa vrsto operacije. Kadar je **sel** enak "00" naj bo na izhodu vrednost vhoda a, pri "01" naj bo na izhodu (a+1), pri "10" naj bo na izhodu vsota (a+b), pri "11" pa razlika (a-b).

Uporabi numerično knjižnico in deklariraj 16-bitne vektorje kot signale podatkovnega tipa **unsigned**. Opiši vezje v jeziku VHDL, preveri delovanje na simulatorju in naredi sintezo vezja.

Iz katerih gradnikov je vezje sestavljeno ? _____

Zapiši velikost sintetiziranega vezja: število RTL celic_____ , zasedenost LUT _____

1.2 Optimizacija vezja

Z upoštevanjem lastnosti računskih operacij lahko naredimo manjše vezje. Razliko lahko izračunamo kot a + (**not** b) + 1. Deklarirajmo notranji signal bi in izračunajmo izhod po postopku:

$$bi = \begin{cases} b, & sel = 10, \\ \textbf{\textit{not}}\ b, & sel = 11 \\ 0, & sicer \end{cases}$$

Izhod naj bo enak a + bi, kadar je sel enak "00" ali "10", sicer pa naj bo enak a + bi + 1.

Opiši vezje v jeziku VHDL in naredi sintezo vezja.

Iz katerih gradnikov je vezje sestavljeno ? _____

Zapiši velikost sintetiziranega vezja: število RTL celic _____ , zasedenost LUT _____