3. Digitalno sito

Naredili bomo sintezo digitalnega sita s končnim odzivom (FIR). Sito izračuna izhod z množenjem trenutnega in zakasnjenih vhodov s koeficienti (konvolucija) po enačbi:

$$y[t] = \sum_{i=0}^{N-1} c[i] * x[t-i]$$

3.1 Osnovna izvedba

- Uporabili bomo funkcjo **fir** [1] in vključili v projekt opis sita: **fir.cpp**, **fir.h** in test: **fir_test.cpp**. Parametri funkcije so: vhod **x**, izhod **y** in matrika koeficientov **c**. Koda vsebuje dve zanki: *shift_loop* za pomikanje vhodnih podatkov in *mac_loop* za izračun izhoda. V **fir.h** so definirani podatkovni tipi in konstanta **N**, ki določa število koeficientov oz. red sita:

```
typedef int coef_t;
typedef int data_t;
typedef int acc_t;

#define N 11
```

- Naredi sintezo vezja za različno število koeficientov: **N**=3, 5, 9, 11 in ugotovi, kateri parameter sintetiziranega vezja se najbolj spreminja:

izvedba	latenca / interval	DSP48E	FF	LUT
N=3				
N=5				
N=9				
N=11				

- Dodaj direktivo za razvijanje zanke shift_loop (**UNROLL**), nato pa še za razdelitev zbirke (pomnilnika) shift_reg na posamezne registre **ARRAY_PARTITION**. Razvij tudi zanko mac_loop in opazuj učinek na sintezo vezja.

- Dodaj na funkcijo direktivo: **PIPELINE**. Kaj se zgodi če odstraniš ostale direktive?

izvedba	latenca / interval	DSP48E	FF	LUT
UNROLL shift				
ARRAY_PARTITION				
UNROLL mac				
PIPELINE				

3.2 Vmesnik

- Preglej seznam signalov vezja. Kakšen je vmesnik za koeficiente?
  Določi vrsto vmesnika BRAM, nato pa direktivo **RESOURCE**: ROM_1P.

- Deklariraj seznam koeficientov znotraj opisa sita. Kakšen je sedaj rezultat sinteze?

```
coef_t c[] = {53, 0, -91, 0, 313, 500, 313, 0, -91, 0, 53};
```

- Dodaj na spremenljivko s koeficienti direktivo: **INTERFACE**, saxi_lite.
  Kaj pa če dodamo: **ARRAY_PARTITION**?

izvedba	latenca / interval	DSP48E	FF	LUT
INTERFACE: BRAM				
RESOURCE: ROM_1P				
INTERFACE: s_axilite				
ARRAY_PARTITION				

3.3 Optimizacija algoritma

- Če vemo, da so koeficienti sita simetrični, lahko optimiziramo vezje. Spremeni računsko zanko in ugotovi kakšna je zasedenost vezja ?

Namig: za polovično sito potrebujemo N=6 koeficientov; 5 in sredinski koeficient. Velikost pomikalnega registra bo 2N-1, ustrezno je potrebno popraviti tudi indekse zank. Množenje s sredinskim koeficientom naredimo izven druge zanke:

```
acc = 0;
mac_loop: for (i = N-2; i >= 0; i--) {
      acc += (shift_reg[i]+shift_reg[2*N-2-i]) * c[i];
}
acc += shift_reg[N-1] * c[N-1];
```

- Naredi sintezo in zapiši koliko ciklov je potrebnih za celoten algoritem (zanka se ponovi 1024-krat).

izvedba	latenca / interval	DSP48E	FF	LUT

Literatura

[1] R. Kastner, J. Matai, S. Neuendorrfer, Parallel Programming for FPGAs,The HLS Book, 2018, http://kastner.ucsd.edu/hlsbook/