

## Kazalo:

Kazalo: .....	1
-- **** STUDENT: 64000225 .....	4
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	4
-- **** STUDENT: 64190088 .....	8
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	8
-- **** STUDENT: 64200100 .....	11
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	11
-- **** STUDENT: 64200112 .....	14
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	14
-- **** STUDENT: 64200163 .....	18
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	18
-- **** STUDENT: 64200238 .....	21
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	21
-- **** STUDENT: 64200288 .....	24
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	24
-- **** STUDENT: 64200296 .....	27
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	27
-- **** STUDENT: 64200385 .....	30
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	30
-- **** STUDENT: 64210132 .....	34
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Napačen inicializacijski polinom za LUT3 (pravi je 96) Pretvorbe pri $O \leq I(I'_{left})$ ni iz std_logic_vector( 0 downto 0 ), ampak je direktno v std_logic. ....	34
-- **** STUDENT: 64210290 .....	38
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	38
-- **** STUDENT: 64210382 .....	41
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	41
-- **** STUDENT: 64210384 .....	44
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Ni pripomb.....	44
-- **** STUDENT: 64210386 .....	47

-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Napačen inicializacijski polinom za LUT3 (pravi je 96). Napačen inicializacijski polinom za LUT4 (pravi je 6996) Za LUT3 je konstanta inicializacije 8 bitna - torej x"96" in ne 32 bitna, kot v vašem primeru - dejansko pa xst vrže ven preostale vrednosti nad 7. bitom. Podobna logika velja za LUT4 in LUT5.....	47
Napake sintetizatorja:.....	47
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 105: Syntax error near "signal".....	47
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 105: std_logic is not a component .....	47
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 106: Syntax error near "signal".....	47
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 106: std_logic_vector is not a component .....	47
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 106: Expected an architecture identifier in index .....	47
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 107: Syntax error near "signal".....	47
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 107: std_logic_vector is not a component .....	47
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 107: Expected an architecture identifier in index .....	47
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 108: Syntax error near "signal".....	47
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 108: std_logic_vector is not a component .....	47
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 108: Expected an architecture identifier in index .....	47
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 109: Syntax error near "signal".....	47
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 109: std_logic_vector is not a component .....	47
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 109: Expected an architecture identifier in index .....	47
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 110: Syntax error near "signal".....	47
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 110: std_logic_vector is not a component .....	47
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 110: Expected an architecture identifier in index .....	47
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 111: Syntax error near "signal".....	47
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 111: std_logic_vector is not a component .....	47
Sorry, too many errors.....	47
-- **** STUDENT: 64210445 .....	51
-- KOMENTARJI K OCENI NALOGE -- Matej Možek: Napačen inicializacijski polinom za LUT3 (pravi je 96) Napačen inicializacijski polinom za LUT4 (pravi je 6996) Za LUT3 je konstanta inicializacije 8 bitna - torej x"96" in ne 32 bitna, kot v vašem primeru - dejansko pa xst vrže ven preostale vrednosti nad 7. bitom. Podobna logika velja za LUT4 in LUT5.....	51
-- **** STUDENT: 64210455 .....	55

-- KOMENTARJI K OCENI NALOGE	-- Matej Možek: Ni pripomb.....	55
-- **** STUDENT: 64210457 .....		59
-- KOMENTARJI K OCENI NALOGE	-- Matej Možek: Napačen inicializacijski polinom za LUT3 (pravi je 96) .....	59
-- **** STUDENT: 64240429 .....		62
-- KOMENTARJI K OCENI NALOGE	-- Matej Možek: Ni pripomb.....	62
-- **** STUDENT: 64240430 .....		67
-- KOMENTARJI K OCENI NALOGE	-- Matej Možek: Ni pripomb.....	67
-- **** STUDENT: 64210113 .....		71
-- KOMENTARJI K OCENI NALOGE	-- Matej Možek: Ni pripomb.....	71
-- **** PREDLOGA VAJE .....		75
-- KOMENTARJI K OCENI NALOGE	-- Matej Možek: Ni pripomb.....	75

```

-- *****
-- **** STUDENT: 64000225
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

    signal in_lut6: std_logic_vector( 5 downto 0 );

begin

    Stage_xor_1:
    if I'length = 1 generate
        O <= I( 0 );    -- xor enega bita je kar ta bit ( x xor 0 ) = x
    end generate;

    Stage_xor_2:
    if I'length = 2 generate
        begin
            O <= I( I'right ) xor I( I'left ); -- xor dvobitnega vektorja std_logic_vector: desni bit xor levi bit
        end generate Stage_xor_2;
    end if;
end architecture tree_of_xor_lut6;

```

```

Stage_xor_3:
if I'length = 3 generate
begin
XOR3_LUT : LUT3
generic map ( INIT => X"96" )
port map (
0 => 0,          -- LUT general output
I0 => I( 0 ), -- LUT input
I1 => I( 1 ), -- LUT input
I2 => I( 2 ) );  -- LUT input
end generate Stage_xor_3;

Stage_xor_4:
if I'length = 4 generate
begin
XOR4_LUT : LUT4
generic map ( INIT => X"6996" )
port map (
0 => 0,          -- LUT general output
I0 => I( 0 ), -- LUT input
I1 => I( 1 ), -- LUT input
I2 => I( 2 ), -- LUT input
I3 => I( 3 ) );  -- LUT input
end generate Stage_xor_4;

Stage_xor_5:
if I'length = 5 generate
begin
XOR5_LUT : LUT5
generic map ( INIT => X"9669_6996" )
port map (
0 => 0,          -- LUT general output
I0 => I( 0 ), -- LUT input
I1 => I( 1 ), -- LUT input
I2 => I( 2 ), -- LUT input
I3 => I( 3 ), -- LUT input
I4 => I( 4 ) );  -- LUT input
end generate Stage_xor_5;

```

```

Stage_xor_6:
if I'length = 6 generate
begin
XOR6_LUT : LUT6
generic map( INIT => X"6996_9669_9669_6996" )
port map(
0 => 0,          -- LUT general output
I0 => I( 0 ), -- LUT input
I1 => I( 1 ), -- LUT input
I2 => I( 2 ), -- LUT input
I3 => I( 3 ), -- LUT input
I4 => I( 4 ), -- LUT input
I5 => I( 5 ) ); -- LUT input
end generate Stage_xor_6;

Stages: if I'length > 6 generate

signal Sixth1_in, Sixth2_in, Sixth3_in, Sixth4_in, Sixth5_in, Sixth6_in: std_logic;
signal Sixth_1 : std_logic_vector( I'length/6 - 1 downto 0 );
signal Sixth_2 : std_logic_vector( I'length/3 - 1 downto I'length/6 );
signal Sixth_3 : std_logic_vector( I'length/2 - 1 downto I'length/3 );
signal Sixth_4 : std_logic_vector( 2*I'length/3 - 1 downto I'length/2 );
signal Sixth_5 : std_logic_vector( 5*I'length/6 - 1 downto 2*I'length/3 );
signal Sixth_6 : std_logic_vector( I'length - 1 downto 5*I'length/6 );

begin
-- razdelimo vhodni vektor na šestine ( 1 - spodnja, 6 - zgornja )
Sixth_1    <= I( I'length/6 - 1 downto 0 );
Sixth_2    <= I( I'length/3 - 1 downto I'length/6 );
Sixth_3    <= I( I'length/2 - 1 downto I'length/3 );
Sixth_4    <= I( 2*I'length/3 - 1 downto I'length/2 );
Sixth_5    <= I( 5*I'length/6 - 1 downto 2*I'length/3 );
Sixth_6    <= I( I'length - 1 downto 5*I'length/6 );

-- povežemo šestine
Sixth_1_Tree: XorTreeStage generic map ( Sixth_1'length ) port map ( Sixth_1, Sixth1_in );
Sixth_2_Tree: XorTreeStage generic map ( Sixth_2'length ) port map ( Sixth_2, Sixth2_in );
Sixth_3_Tree: XorTreeStage generic map ( Sixth_3'length ) port map ( Sixth_3, Sixth3_in );
Sixth_4_Tree: XorTreeStage generic map ( Sixth_4'length ) port map ( Sixth_4, Sixth4_in );
Sixth_5_Tree: XorTreeStage generic map ( Sixth_5'length ) port map ( Sixth_5, Sixth5_in );

```

```

Sixth_6_Tree: XorTreeStage generic map ( Sixth_6'length ) port map ( Sixth_6, Sixth6_in );

XOR6_LUT : LUT6
generic map( INIT => X"6996_9669_9669_6996" )
port map(
0 => 0,          -- LUT general output
I0 => Sixth1_in,  -- LUT input
I1 => Sixth2_in,  -- LUT input
I2 => Sixth3_in,  -- LUT input
I3 => Sixth4_in,  -- LUT input
I4 => Sixth5_in,  -- LUT input
I5 => Sixth6_in ); -- LUT input

    end generate Stages;
end tree_of_xor_lut6;

```

```

-- *****
-- **** STUDENT: 64190088
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

    architecture tree_of_xor_lut6 of XorTreeStage is

component XorTreeStage
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end component;

begin
    -- KODO VPISUJETE SEM

    Stage_xor_1:
    if I'length = 1 generate
    0      <= I( 0 );
    end generate;

    Stage_xor_2:
    if I'length = 2 generate
    0      <= I( I'right ) xor I( I'left );
    end generate Stage_xor_2;

    Stage_xor_3:

```



```

if I'length = 3 generate
XOR3_LUT: LUT3
generic map ( INIT => X"96" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ) );
end generate Stage_xor_3;

Stage_xor_4:
if I'length = 4 generate
XOR4_LUT: LUT4
generic map ( INIT => X"6996" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ) );
end generate Stage_xor_4;

Stage_xor_5:
if I'length = 5 generate
XOR5_LUT: LUT5
generic map ( INIT => X"9669_6996" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ) );
end generate Stage_xor_5;

Stage_xor_6:
if I'length = 6 generate
XOR6_LUT: LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => I( 5 ) );
end generate Stage_xor_6;

Stages: if I'length > 6 generate

signal Sixth1_in : std_logic_vector( I'length*1/6 - 1 downto 0 );
signal Sixth2_in : std_logic_vector( I'length*2/6 - 1 downto I'length*1/6 );
signal Sixth3_in : std_logic_vector( I'length*3/6 - 1 downto I'length*2/6 );
signal Sixth4_in : std_logic_vector( I'length*4/6 - 1 downto I'length*3/6 );
signal Sixth5_in : std_logic_vector( I'length*5/6 - 1 downto I'length*4/6 );
signal Sixth6_in : std_logic_vector( I'length*6/6 - 1 downto I'length*5/6 );

```

```
signal Sixth1_xor, Sixth2_xor, Sixth3_xor, Sixth4_xor, Sixth5_xor, Sixth6_xor: std_logic;
```

```
begin
```

```
Sixth1_in    <= I( I'length*1/6 - 1 downto 0 );
```

```
Sixth2_in    <= I( I'length*2/6 - 1 downto I'length*1/6 );
```

```
Sixth3_in    <= I( I'length*3/6 - 1 downto I'length*2/6 );
```

```
Sixth4_in    <= I( I'length*4/6 - 1 downto I'length*3/6 );
```

```
Sixth5_in    <= I( I'length*5/6 - 1 downto I'length*4/6 );
```

```
Sixth6_in    <= I( I'length*6/6 - 1 downto I'length*5/6 );
```

```
Sixth1_Tree: XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, Sixth1_xor );
```

```
Sixth2_Tree: XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, Sixth2_xor );
```

```
Sixth3_Tree: XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, Sixth3_xor );
```

```
Sixth4_Tree: XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, Sixth4_xor );
```

```
Sixth5_Tree: XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, Sixth5_xor );
```

```
Sixth6_Tree: XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, Sixth6_xor );
```

```
XOR6_LUT : LUT6
```

```
generic map( INIT =>X"6996_9669_9669_6996" )
```

```
port map(
```

```
0 => 0,      I0 => Sixth1_Xor,  I1 => Sixth2_Xor,  I2 => Sixth3_Xor,  I3 => Sixth4_Xor,  I4 => Sixth5_Xor,
```

```
I5 => Sixth6_Xor );
```

```
end generate Stages;
```

```
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64200100
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin
    -- KODO VPISUJETE SEM
    Stage_xor_1:
    if I'length = 1 generate
        O <= I( 0 );
    end generate Stage_xor_1;

    Stage_xor_2:
    if I'length = 2 generate
    begin
        O <= I( 1 ) xor I( 0 );
    end generate Stage_xor_2;

    Stage_xor_3:

```

```

if I' length= 3 generate
begin
XOR_LUT3:LUT3
generic map( INIT => X"96" )
port map( 0=>0, I0=>I( 0 ),I1=>I( 1 ),I2=>I( 2 ) );
end generate Stage_xor_3;

Stage_xor_4:
if I' length= 4 generate
begin
XOR_LUT4:LUT4
generic map( INIT => X"6996" )
port map( 0=>0, I0=>I( 0 ),I1=>I( 1 ),I2=>I( 2 ),I3=>I( 3 ) );
end generate Stage_xor_4;

Stage_xor_5:
if I' length= 5 generate
begin
XOR_LUT5:LUT5
generic map( INIT => X"9669_6996" )
port map( 0=>0, I0=>I( 0 ),I1=>I( 1 ),I2=>I( 2 ),I3=>I( 3 ),I4=>I( 4 ) );
end generate Stage_xor_5;

Stage_xor_6:
if I' length= 6 generate
begin
XOR_LUT6:LUT6
generic map( INIT => X"6996_9669_9669_6996" )
port map( 0=>0, I0=>I( 0 ),I1=>I( 1 ),I2=>I( 2 ),I3=>I( 3 ),I4=>I( 4 ),I5=>I( 5 ) );
end generate Stage_xor_6;

Stages: if I'length > 6 generate
    signal xor_1, xor_2,xor_3,xor_4,xor_5,xor_6: std_logic;
    signal Sixth1_in: std_logic_vector( I'length-1 downto 5*I'length/6 );
    signal Sixth2_in: std_logic_vector( 5*I'length/6-1 downto 4*I'length/6 );
    signal Sixth3_in: std_logic_vector( 4*I'length/6-1 downto 3*I'length/6 );
    signal Sixth4_in: std_logic_vector( 3*I'length/6-1 downto 2*I'length/6 );
    signal Sixth5_in: std_logic_vector( 2*I'length/6-1 downto I'length/6 );
    signal Sixth6_in: std_logic_vector( I'length/6-1 downto 0 );

```

```
begin
```

```
Sixth1_in    <= I( I'length-1 downto 5*I'length/6 );  
Sixth2_in    <= I( 5*I'length/6-1 downto 4*I'length/6 );  
Sixth3_in    <= I( 4*I'length/6-1 downto 3*I'length/6 );  
Sixth4_in    <= I( 3*I'length/6-1 downto 2*I'length/6 );  
Sixth5_in    <= I( 2*I'length/6-1 downto I'length/6 );  
Sixth6_in    <= I( I'length/6-1 downto 0 );
```

```
N1_Tree: XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, xor_1 );  
N2_Tree: XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, xor_2 );  
N3_Tree: XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, xor_3 );  
N4_Tree: XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, xor_4 );  
N5_Tree: XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, xor_5 );  
N6_Tree: XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, xor_6 );
```

```
XOR_LUT6:LUT6
```

```
generic map( INIT => X"6996_9669_9669_6996" )  
port map( 0=>0, I0=>xor_1,I1=>xor_2,I2=>xor_3,I3=>xor_4,I4=>xor_5,I5=>xor_6 );  
end generate Stages;
```

```
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64200112
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic (N : natural);
    port( I: in std_logic_vector(N - 1 downto 0); -- vhodni vektor redukcije ima N vhodov
          O: out std_logic); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic (N : natural);
        port( I: in std_logic_vector(N - 1 downto 0); -- vhodni vektor redukcije ima N vhodov
              O: out std_logic); -- izhodni bit redukcije
    end component;

begin

    Stage_xor_1:
        if I'length = 1 generate
            O <= I(0);
        end generate Stage_xor_1;

    Stage_xor_2:
        if I'length = 2 generate
            begin
                O <= I(I'right) xor I(I'left);
            end generate Stage_xor_2;

```

```

Stage_xor_3:
  if I'length = 3 generate
    begin
      XOR_LUT3: LUT3
        generic map(
          INIT => X"96")
        port map(
          0 => 0,
          I0 => I(0),
          I1 => I(1),
          I2 => I(2));
    end generate Stage_xor_3;

Stage_xor_4:
  if I'length = 4 generate
    begin
      XOR_LUT4: LUT4
        generic map(
          INIT => X"6996")
        port map(
          0 => 0,
          I0 => I(0),
          I1 => I(1),
          I2 => I(2),
          I3 => I(3));
    end generate Stage_xor_4;

Stage_xor_5:
  if I'length = 5 generate
    begin
      XOR_LUT5: LUT5
        generic map(
          INIT => X"9669_6996")
        port map(
          0 => 0,
          I0 => I(0),
          I1 => I(1),
          I2 => I(2),
          I3 => I(3),

```

```

        I4 => I(4));
    end generate Stage_xor_5;

Stage_xor_6:
    if I'length = 6 generate
        begin
            XOR_LUT6: LUT6
                generic map(
                    INIT => X"6996_9669_9669_6996")
                port map(
                    0 => 0,
                    I0 => I(0),
                    I1 => I(1),
                    I2 => I(2),
                    I3 => I(3),
                    I4 => I(4),
                    I5 => I(5));
        end generate Stage_xor_6;

Stages:
    if I'length > 6 generate
        signal in_lut : std_logic_vector(5 downto 0) := "000000";

        signal Sixth1_in : std_logic_vector( ( ( I'length)/6) - 1 downto 0);
        signal Sixth2_in : std_logic_vector( ( (2*I'length)/6) - 1 downto ( ( I'length)/6) );
        signal Sixth3_in : std_logic_vector( ( (3*I'length)/6) - 1 downto ( (2*I'length)/6) );
        signal Sixth4_in : std_logic_vector( ( (4*I'length)/6) - 1 downto ( (3*I'length)/6) );
        signal Sixth5_in : std_logic_vector( ( (5*I'length)/6) - 1 downto ( (4*I'length)/6) );
        signal Sixth6_in : std_logic_vector( I'length - 1 downto ( (5*I'length)/6) );

        begin
            Sixth1_in <= I( ( ( I'length)/6) - 1 downto 0);
            Sixth2_in <= I( ( (2*I'length)/6) - 1 downto ( ( I'length)/6) );
            Sixth3_in <= I( ( (3*I'length)/6) - 1 downto ( (2*I'length)/6) );
            Sixth4_in <= I( ( (4*I'length)/6) - 1 downto ( (3*I'length)/6) );
            Sixth5_in <= I( ( (5*I'length)/6) - 1 downto ( (4*I'length)/6) );
            Sixth6_in <= I( I'length - 1 downto ( (5*I'length)/6) );

            Tree1: XorTreeStage generic map (Sixth1_in'length) port map (Sixth1_in, in_lut(0));
            Tree2: XorTreeStage generic map (Sixth2_in'length) port map (Sixth2_in, in_lut(1));
        end generate
    end if
end Stages;

```



```
Tree3: XorTreeStage generic map (Sixth3_in'length) port map (Sixth3_in, in_lut(2));
Tree4: XorTreeStage generic map (Sixth4_in'length) port map (Sixth4_in, in_lut(3));
Tree5: XorTreeStage generic map (Sixth5_in'length) port map (Sixth5_in, in_lut(4));
Tree6: XorTreeStage generic map (Sixth6_in'length) port map (Sixth6_in, in_lut(5));
```

```
XOR_LUT6_stg : LUT6
  generic map(
    INIT => X"6996_9669_9669_6996")
  port map(
    0 => 0,
    I0 => in_lut(0),
    I1 => in_lut(1),
    I2 => in_lut(2),
    I3 => in_lut(3),
    I4 => in_lut(4),
    I5 => in_lut(5));
```

```
end generate Stages;
```

```
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64200163
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );
          O: out std_logic );
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );
              O: out std_logic );
    end component;

begin
    Stage_xor_1: if I'length = 1 generate
    begin
        O <= I( I'left );
    end generate Stage_xor_1;

    Stage_xor_2: if I'length = 2 generate
    begin
        O <= I( I'right ) xor I( I'left );
    end generate Stage_xor_2;

    Stage_xor_3: if I'length = 3 generate
    begin
        XOR3_LUT3: LUT3

```

```

generic map ( INIT => X"96" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ) );
end generate Stage_xor_3;

Stage_xor_4: if I'length = 4 generate
begin
XOR4_LUT : LUT4
generic map( INIT => X"6996" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ) );
end generate Stage_xor_4;

Stage_xor_5: if I'length = 5 generate
begin
XOR5_LUT : LUT5
generic map( INIT => X"9669_6996" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ) );
end generate Stage_xor_5;

Stage_xor_6: if I'length = 6 generate
begin
XOR6_LUT : LUT6
generic map( INIT => X"6996_9669_9669_6996" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => I( 5 ) );
end generate Stage_xor_6;

Stages: if I'length > 6 generate
signal Xor_1_6, Xor_2_6, Xor_3_6, Xor_4_6, Xor_5_6, Xor_6_6: std_logic;
signal Sixth1_in : std_logic_vector( I'length - 1 downto I'length*5/6 );
signal Sixth2_in : std_logic_vector( I'length*5/6 - 1 downto I'length*4/6 );
signal Sixth3_in : std_logic_vector( I'length*4/6 - 1 downto I'length*3/6 );
signal Sixth4_in : std_logic_vector( I'length*3/6 - 1 downto I'length*2/6 );
signal Sixth5_in : std_logic_vector( I'length*2/6 - 1 downto I'length*1/6 );
signal Sixth6_in : std_logic_vector( I'length*1/6 - 1 downto 0 );

begin

```

```

Sixth1_in    <= I( I'length - 1 downto I'length*5/6 );
Sixth2_in    <= I( I'length*5/6 - 1 downto I'length*4/6 );
Sixth3_in    <= I( I'length*4/6 - 1 downto I'length*3/6 );
Sixth4_in    <= I( I'length*3/6 - 1 downto I'length*2/6 );
Sixth5_in    <= I( I'length*2/6 - 1 downto I'length*1/6 );
Sixth6_in    <= I( I'length*1/6 - 1 downto 0 );

Tree_1_6: XorTreeStage generic map ( N => Sixth1_in'length ) port map ( I => Sixth1_in, O => Xor_1_6 );
Tree_2_6: XorTreeStage generic map ( N => Sixth2_in'length ) port map ( I => Sixth2_in, O => Xor_2_6 );
Tree_3_6: XorTreeStage generic map ( N => Sixth3_in'length ) port map ( I => Sixth3_in, O => Xor_3_6 );
Tree_4_6: XorTreeStage generic map ( N => Sixth4_in'length ) port map ( I => Sixth4_in, O => Xor_4_6 );
Tree_5_6: XorTreeStage generic map ( N => Sixth5_in'length ) port map ( I => Sixth5_in, O => Xor_5_6 );
Tree_6_6: XorTreeStage generic map ( N => Sixth6_in'length ) port map ( I => Sixth6_in, O => Xor_6_6 );

XOR6_LUT : LUT6
generic map( INIT => X"6996_9669_9669_6996" )
port map(
    O => 0,      I0 => Xor_1_6,      I1 => Xor_2_6,      I2 => Xor_3_6,      I3 => Xor_4_6,      I4 => Xor_5_6,
    I5 => Xor_6_6 );
end generate Stages;
end tree_of_xor_lut6;

```

```

-- *****
-- **** STUDENT: 64200238
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin
    -- KODO VPISUJETE SEM

    Stage_xor_1:
    if I' length = 1 generate
        O <= I( 0 );    -- xor enega bita je kar ta bit ( x xor 0 ) = x
    end generate;

    Stage_xor_2:
    if I' length = 2 generate
        begin
            O <= I( 1 ) xor I( 0 );    -- xor dvobitnega vektorja std_logic_vector: desni bit xor levi bit
        end generate Stage_xor_2;
    end if;
end architecture tree_of_xor_lut6;

```

```

Stage_xor_3:
if I' length=3 generate
begin
XOR_LUT3: LUT3
generic map( INIT => X"96" )
port map(
O=>O, I0=>I( 0 ), I1=>I( 1 ), I2=>I( 2 )
);
end generate Stage_xor_3;

```

```

Stage_xor_4:
if I' length=4 generate
begin
XOR_LUT4: LUT4
generic map( INIT=> X"6996" )
port map(
O=>O, I0=>I( 0 ), I1=>I( 1 ), I2=>I( 2 ), I3=>I( 3 )
);
end generate Stage_xor_4;

```

```

Stage_xor_5:
if I' length=5 generate
begin
XOR_LUT5: LUT5
generic map( INIT=> X"9669_6996" )
port map(
O=>O, I0=>I( 0 ), I1=>I( 1 ), I2=>I( 2 ), I3=>I( 3 ), I4=>I( 4 )
);
end generate Stage_xor_5;

```

```

Stage_xor_6:
if I' length=6 generate
XOR_LUT6: LUT6
generic map( INIT=> X"6996_9669_9669_6996" )
port map(
O=>O, I0=>I( 0 ), I1=>I( 1 ), I2=>I( 2 ), I3=>I( 3 ), I4=>I( 4 ), I5=>I( 5 )
);
end generate Stage_xor_6;

```

```

Stages: if I'length > 6 generate

```

```

signal ladj1,ladja2,ladja3,ladja4,ladja5,ladja6 : std_logic;
signal Sixth1_in : std_logic_vector( I'length-1 downto 5*I'length/6 );
signal Sixth2_in : std_logic_vector( 5*I'length/6-1 downto 4*I'length/6 );
signal Sixth3_in : std_logic_vector( 4*I'length/6-1 downto 3*I'length/6 );
signal Sixth4_in : std_logic_vector( 3*I'length/6-1 downto 2*I'length/6 );
signal Sixth5_in : std_logic_vector( 2*I'length/6-1 downto I'length/6 );
signal Sixth6_in : std_logic_vector( I'length/6-1 downto 0 );

```

```
begin
```

```

Sixth1_in    <= I( I'length-1 downto 5*I'length/6 );
Sixth2_in    <= I( 5*I'length/6-1 downto 4*I'length/6 );
Sixth3_in    <= I( 4*I'length/6-1 downto 3*I'length/6 );
Sixth4_in    <= I( 3*I'length/6-1 downto 2*I'length/6 );
Sixth5_in    <= I( 2*I'length/6-1 downto I'length/6 );
Sixth6_in    <= I( I'length/6-1 downto 0 );

```

```

Stage1: XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, ladj1 );
Stage2: XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, ladj2 );
Stage3: XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, ladj3 );
Stage4: XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, ladj4 );
Stage5: XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, ladj5 );
Stage6: XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, ladj6 );

```

```
XOR6_LUT6: LUT6
```

```
generic map ( INIT => X"6996_9669_9669_6996" )
```

```
port map (
```

```
0 => 0,      I0 => ladj1,      I1 => ladj2,      I2 => ladj3,      I3 => ladj4,      I4 => ladj5,
```

```
I5 => ladj6
```

```
);
```

```
end generate Stages;
```

```
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64200288
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin
    -- KODO VPISUJETE SEM

    Stage_xor_1:
    if I' length = 1 generate
        O <= I( 0 );    -- xor enega bita je kar ta bit ( x xor 0 ) = x
    end generate;

    Stage_xor_2:
    if I' length = 2 generate
        begin
            O <= I( 1 ) xor I( 0 );    -- xor dvobitnega vektorja std_logic_vector: desni bit xor levi bit
        end generate Stage_xor_2;
    end if;
end architecture tree_of_xor_lut6;

```



```

Stage_xor_3:
if I' length=3 generate
begin
XOR_LUT3: LUT3
generic map( INIT => X"96" )
port map(
O=>O, I0=>I( 0 ), I1=>I( 1 ), I2=>I( 2 )
);
end generate Stage_xor_3;

```

```

Stage_xor_4:
if I' length=4 generate
begin
XOR_LUT4: LUT4
generic map( INIT=> X"6996" )
port map(
O=>O, I0=>I( 0 ), I1=>I( 1 ), I2=>I( 2 ), I3=>I( 3 )
);
end generate Stage_xor_4;

```

```

Stage_xor_5:
if I' length=5 generate
begin
XOR_LUT5: LUT5
generic map( INIT=> X"9669_6996" )
port map(
O=>O, I0=>I( 0 ), I1=>I( 1 ), I2=>I( 2 ), I3=>I( 3 ), I4=>I( 4 )
);
end generate Stage_xor_5;

```

```

Stage_xor_6:
if I' length=6 generate
XOR_LUT6: LUT6
generic map( INIT=> X"6996_9669_9669_6996" )
port map(
O=>O, I0=>I( 0 ), I1=>I( 1 ), I2=>I( 2 ), I3=>I( 3 ), I4=>I( 4 ), I5=>I( 5 )
);
end generate Stage_xor_6;

```

```

Stages: if I'length > 6 generate

```

```

signal testVar1,testVar2,testVar3,testVar4,testVar5,testVar6 : std_logic;
signal Sixth1_in : std_logic_vector( I'length-1 downto 5*I'length/6 );
signal Sixth2_in : std_logic_vector( 5*I'length/6-1 downto 4*I'length/6 );
signal Sixth3_in : std_logic_vector( 4*I'length/6-1 downto 3*I'length/6 );
signal Sixth4_in : std_logic_vector( 3*I'length/6-1 downto 2*I'length/6 );
signal Sixth5_in : std_logic_vector( 2*I'length/6-1 downto I'length/6 );
signal Sixth6_in : std_logic_vector( I'length/6-1 downto 0 );

```

```
begin
```

```

Sixth1_in    <= I( I'length-1 downto 5*I'length/6 );
Sixth2_in    <= I( 5*I'length/6-1 downto 4*I'length/6 );
Sixth3_in    <= I( 4*I'length/6-1 downto 3*I'length/6 );
Sixth4_in    <= I( 3*I'length/6-1 downto 2*I'length/6 );
Sixth5_in    <= I( 2*I'length/6-1 downto I'length/6 );
Sixth6_in    <= I( I'length/6-1 downto 0 );

```

```

Stage1: XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, testVar1 );
Stage2: XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, testVar2 );
Stage3: XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, testVar3 );
Stage4: XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, testVar4 );
Stage5: XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, testVar5 );
Stage6: XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, testVar6 );

```

```
XOR6_LUT6: LUT6
```

```
generic map ( INIT => X"6996_9669_9669_6996" )
```

```
port map (
```

```

0 => 0,      I0 => testVar1,      I1 => testVar2,      I2 => testVar3,      I3 => testVar4,      I4 => testVar5,
I5 => testVar6
);

```

```
end generate Stages;
```

```
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64200296
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin
    Stage_xor_1:
        if I'length = 1 generate
            0    <= I( 0 );    -- xor enega bita je kar ta bit ( x xor 0 ) = x
        end generate;

    Stage_xor_2:
        if I'length = 2 generate
            begin
                0    <= I( I'right ) xor I( I'left ); -- xor dvobitnega vektorja std_logic_vector: desni bit xor levi bit
            end generate Stage_xor_2;

    Stage_xor_3 :
        if I'length = 3 generate

```

```

begin
XOR3_LUT : LUT3
generic map ( INIT => X"96" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 )
);
end generate Stage_xor_3;

```

```

Stage_xor_4 :
if I'length = 4 generate
begin
XOR4_LUT : LUT4
generic map ( INIT => X"6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 )
);
end generate Stage_xor_4;

```

```

Stage_xor_5 :
if I'length = 5 generate
begin
XOR5_LUT : LUT5
generic map ( INIT => X"9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 )
);
end generate Stage_xor_5;

```

```

Stage_xor_6 :
if I'length = 6 generate
begin
XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => I( 5 )
);
end generate Stage_xor_6;

```

Stages:

```

if I'length > 6 generate
signal Sixth1, Sixth2, Sixth3, Sixth4, Sixth5, Sixth6 : std_logic;
signal Sixth1_in : std_logic_vector ( I'length/6 -1 downto 0 );
signal Sixth2_in : std_logic_vector ( I'length*2/6 -1 downto I'length/6 );
signal Sixth3_in : std_logic_vector ( I'length*3/6 -1 downto I'length*2/6 );
signal Sixth4_in : std_logic_vector ( I'length*4/6 -1 downto I'length*3/6 );
signal Sixth5_in : std_logic_vector ( I'length*5/6 -1 downto I'length*4/6 );
signal Sixth6_in : std_logic_vector ( I'length - 1 downto I'length*5/6 );
begin

-- division into sixths
Sixth1_in  <= I( I'length/6 - 1 downto 0 );
Sixth2_in  <= I( I'length*2/6 - 1 downto I'length/6 );
Sixth3_in  <= I( I'length*3/6 - 1 downto I'length*2/6 );
Sixth4_in  <= I( I'length*4/6 - 1 downto I'length*3/6 );
Sixth5_in  <= I( I'length*5/6 - 1 downto I'length*4/6 );
Sixth6_in  <= I( I'length - 1 downto I'length*5/6 );

Sixth1_tree: XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, Sixth1 );
Sixth2_tree: XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, Sixth2 );
Sixth3_tree: XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, Sixth3 );
Sixth4_tree: XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, Sixth4 );
Sixth5_tree: XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, Sixth5 );
Sixth6_tree: XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, Sixth6 );

XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => Sixth1,      I1 => Sixth2,      I2 => Sixth3,      I3 => Sixth4,      I4 => Sixth5,
I5 => Sixth6
);

end generate Stages;

end tree_of_xor_lut6;

```

```

-- *****
-- **** STUDENT: 64200385
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin
    -- KODO VPISUJETE SEM

    Stage_xor_1:
    if I'length = 1 generate
        O <= I( 0 );    -- ( x xor 0 ) = x
    end generate;

    Stage_xor_2:
    if I'length = 2 generate
        begin
            O <= I( I'right ) xor I( I'left );

        end generate Stage_xor_2;

```

```

Stage_xor_3:
  if I'length = 3 generate
    begin
      XOR3_LUT : LUT3
        generic map(
          INIT => X"96" )
        port map(
          0 => 0,
          I0 => I( 0 ),
          I1 => I( 1 ),
          I2 => I( 2 ) );

      end generate Stage_xor_3;

```

```

Stage_xor_4:
  if I'length = 4 generate
    begin
      XOR4_LUT : LUT4
        generic map(
          INIT => X"6996" )
        port map(
          0 => 0,
          I0 => I( 0 ),
          I1 => I( 1 ),
          I2 => I( 2 ),
          I3 => I( 3 ) );

      end generate Stage_xor_4;

```

```

Stage_xor_5:
  if I'length = 5 generate
    begin
      XOR5_LUT : LUT5
        generic map(
          INIT => X"9669_6996" )
        port map(
          0 => 0,
          I0 => I( 0 ),
          I1 => I( 1 ),

```

```

I2 => I( 2 ),
I3 => I( 3 ),
I4 => I( 4 ) );

end generate Stage_xor_5;
Stage_xor_6:
if I'length = 6 generate
    begin
    XOR6_LUT : LUT6
    generic map(
    INIT => X"6996_9669_9669_6996" )
    port map(
    0 => 0,
    I0 => I( 0 ),
    I1 => I( 1 ),
    I2 => I( 2 ),
    I3 => I( 3 ),
    I4 => I( 4 ),
    I5 => I( 5 ) );

    end generate Stage_xor_6;

Stages: if I'length > 6 generate
    signal Sixth1_0, Sixth2_0, Sixth3_0, Sixth4_0, Sixth5_0, Sixth6_0 : std_logic;
    signal Sixth1_In : std_logic_vector( I'length/6 - 1 downto 0 );
    signal Sixth2_In : std_logic_vector( 2*I'length/6 - 1 downto I'length/6 );
    signal Sixth3_In : std_logic_vector( 3*I'length/6 - 1 downto 2*I'length/6 );
    signal Sixth4_In : std_logic_vector( 4*I'length/6 - 1 downto 3*I'length/6 );
    signal Sixth5_In : std_logic_vector( 5*I'length/6 - 1 downto 4*I'length/6 );
    signal Sixth6_In : std_logic_vector( 6*I'length/6 - 1 downto 5*I'length/6 );
    begin

        Sixth1_In    <= I( I'length/6 - 1 downto 0 );
        Sixth2_In    <= I( 2*I'length/6 - 1 downto I'length/6 );
        Sixth3_In    <= I( 3*I'length/6 - 1 downto 2*I'length/6 );
        Sixth4_In    <= I( 4*I'length/6 - 1 downto 3*I'length/6 );
        Sixth5_In    <= I( 5*I'length/6 - 1 downto 4*I'length/6 );
        Sixth6_In    <= I( 6*I'length/6 - 1 downto 5*I'length/6 );

        XTree1: XorTreeStage generic map ( Sixth1_In'length ) port map ( Sixth1_In, Sixth1_0 );

```



```

XTree2: XorTreeStage generic map ( Sixth2_In'length ) port map ( Sixth2_In, Sixth2_0 );
XTree3: XorTreeStage generic map ( Sixth3_In'length ) port map ( Sixth3_In, Sixth3_0 );
XTree4: XorTreeStage generic map ( Sixth4_In'length ) port map ( Sixth4_In, Sixth4_0 );
XTree5: XorTreeStage generic map ( Sixth5_In'length ) port map ( Sixth5_In, Sixth5_0 );
XTree6: XorTreeStage generic map ( Sixth6_In'length ) port map ( Sixth6_In, Sixth6_0 );

XOR6_LUT : LUT6
generic map(
  INIT => X"6996_9669_9669_6996" )
port map(
  0 => 0,
  I0 => Sixth1_0,
  I1 => Sixth2_0,
  I2 => Sixth3_0,
  I3 => Sixth4_0,
  I4 => Sixth5_0,
  I5 => Sixth6_0 );

    end generate Stages;

end tree_of_xor_lut6;

```

```

-- *****
-- **** STUDENT: 64210132
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek:
Napačen inicializacijski polinom za LUT3 (pravi je 96)
Pretvorbe pri  $0 \leq I(I'left)$  ni iz std_logic_vector( 0 downto 0 ), ampak je direktno v std_logic.
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin

    Stage_xor_1:
    if I'length = 1 generate
    begin
        0    <= I( I'left );    -- xor enega bita std_logic_vector je kar ta bit ( x xor 0 ) = x
        -- ta stavek obenem opravlja pretvorbo std_logic_vector( 0 downto 0 ) v tip izhoda ( 0 ), ki je std_logic
    end generate Stage_xor_1;

    Stage_xor_2:
    if I'length = 2 generate
    begin

```

```
0      <= I( I'right ) xor I( I'left ); -- xor dvobitnega vektorja std_logic_vector: desni bit xor levi bit
end generate Stage_xor_2;
```

```
Stage_xor_3:
if I'length = 3 generate
begin
```

```
LUT3_inst : LUT3
generic map (
INIT => X"69" )
port map (
0 => 0,          -- LUT general output
I0 => I( 0 ), -- LUT input
I1 => I( 1 ), -- LUT input
I2 => I( 2 ) -- LUT input
);
```

```
end generate Stage_xor_3;
```

```
Stage_xor_4:
if I'length = 4 generate
begin
```

```
LUT4_inst : LUT4
generic map (
INIT => X"6996" )
port map (
0 => 0,          -- LUT general output
I0 => I( 0 ), -- LUT input
I1 => I( 1 ), -- LUT input
I2 => I( 2 ), -- LUT input
I3 => I( 3 ) -- LUT input
);
```

```
end generate Stage_xor_4;
```

```
Stage_xor_5:
if I'length = 5 generate
begin
```

```

LUT5_inst : LUT5
generic map (
  INIT => X"6996_9669" )
port map (
  0 => 0,          -- LUT general output
  I0 => I( 0 ), -- LUT input
  I1 => I( 1 ), -- LUT input
  I2 => I( 2 ), -- LUT input
  I3 => I( 3 ), -- LUT input
  I4 => I( 4 ) -- LUT input
);

end generate Stage_xor_5;

Stage_xor_6:
if I'length = 6 generate
begin

  LUT6_inst : LUT6
  generic map (
    INIT => X"6996_9669_9669_6996" )
  port map (
    0 => 0,          -- LUT general output
    I0 => I( 0 ), -- LUT input
    I1 => I( 1 ), -- LUT input
    I2 => I( 2 ), -- LUT input
    I3 => I( 3 ), -- LUT input
    I4 => I( 4 ), -- LUT input
    I5 => I( 5 ) -- LUT input
  );

end generate Stage_xor_6;

Stages: if I'length > 6 generate
-- Signali ki bodo pripeljali rezultate iz vsake šestine vhodnega vektorja na trenutni LUT6
signal Sixth1_Xor, Sixth2_Xor, Sixth3_Xor, Sixth4_Xor, Sixth5_Xor, Sixth6_Xor: std_logic;
-- Signali za razdelitev vhodnega vektorja na šestine
signal Sixth1_in : std_logic_vector( I'length/6 - 1 downto 0 );
signal Sixth2_in : std_logic_vector( I'length*2/6 - 1 downto I'length/6 );
signal Sixth3_in : std_logic_vector( I'length*3/6 - 1 downto I'length*2/6 );

```

```

signal Sixth4_in : std_logic_vector( I'length*4/6 - 1 downto I'length*3/6 );
signal Sixth5_in : std_logic_vector( I'length*5/6 - 1 downto I'length*4/6 );
signal Sixth6_in : std_logic_vector( I'length - 1 downto I'length*5/6 );

```

```
begin
```

```
-- razdelimo vhodni vektor na šestine
```

```

Sixth1_in  <= I( I'length/6 - 1 downto 0 );
Sixth2_in  <= I( I'length*2/6 - 1 downto I'length/6 );
Sixth3_in  <= I( I'length*3/6 - 1 downto I'length*2/6 );
Sixth4_in  <= I( I'length*4/6 - 1 downto I'length*3/6 );
Sixth5_in  <= I( I'length*5/6 - 1 downto I'length*4/6 );
Sixth6_in  <= I( I'length - 1 downto I'length*5/6 );

```

```
-- povežemo šestine z rekurzivnim povezovalnim stavkom
```

```

Sixth1_Tree: XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, Sixth1_Xor );
Sixth2_Tree: XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, Sixth2_Xor );
Sixth3_Tree: XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, Sixth3_Xor );
Sixth4_Tree: XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, Sixth4_Xor );
Sixth5_Tree: XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, Sixth5_Xor );
Sixth6_Tree: XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, Sixth6_Xor );

```

```
-- Pripeljemo izhode iz vsake šestine na vhode LUT6 in izhod LUT6 na izhod
```

```

LUT6_inst : LUT6
generic map (
  INIT => X"6996_9669_9669_6996" )
port map (
  0 => 0,      -- LUT general output
  I0 => Sixth1_Xor,  -- LUT input
  I1 => Sixth2_Xor,  -- LUT input
  I2 => Sixth3_Xor,  -- LUT input
  I3 => Sixth4_Xor,  -- LUT input
  I4 => Sixth5_Xor,  -- LUT input
  I5 => Sixth6_Xor   -- LUT input
);
end generate Stages;

```

```
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64210290
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin

    Stage_xor_1:
        if I'length = 1 generate
        begin
            O <= I( I'left );
        end generate Stage_xor_1;

    Stage_xor_2:
        if I'length = 2 generate
        begin
            O <= I( I'right ) xor I( I'left );
        end generate Stage_xor_2;

```

```

Stage_xor_3:
  if I'length = 3 generate
  begin
    XOR3_LUT : LUT3
    generic map(
      INIT => X"96" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ) );
    end generate Stage_xor_3;

Stage_xor_4:
  if I'length = 4 generate
  begin
    XOR4_LUT : LUT4
    generic map(
      INIT => X"6996" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ) );
    end generate Stage_xor_4;

Stage_xor_5:
  if I'length = 5 generate
  begin
    XOR5_LUT : LUT5
    generic map(
      INIT => X"9669_6996" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ) );
    end generate Stage_xor_5;

Stage_xor_6:
  if I'length = 6 generate
  begin
    XOR6_LUT : LUT6
    generic map(
      INIT => X"6996_9669_9669_6996" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
      I5 => I( 5 ) );
    end generate Stage_xor_6;

```

Stages:

```
if I'length > 6 generate
  signal piece0_xor : std_logic;
  signal piece1_xor : std_logic;
  signal piece2_xor : std_logic;
  signal piece3_xor : std_logic;
  signal piece4_xor : std_logic;
  signal piece5_xor : std_logic;
  signal piece5_in : std_logic_vector ( I'length-1 downto I'length*5/6 );
  signal piece4_in : std_logic_vector ( I'length*5/6-1 downto I'length*4/6 );
  signal piece3_in : std_logic_vector ( I'length*4/6-1 downto I'length*3/6 );
  signal piece2_in : std_logic_vector ( I'length*3/6-1 downto I'length*2/6 );
  signal piece1_in : std_logic_vector ( I'length*2/6-1 downto I'length/6 );
  signal piece0_in : std_logic_vector ( I'length/6-1 downto 0 );
begin
  piece5_in    <= I( I'length-1 downto I'length*5/6 );
  piece4_in    <= I( I'length*5/6-1 downto I'length*4/6 );
  piece3_in    <= I( I'length*4/6-1 downto I'length*3/6 );
  piece2_in    <= I( I'length*3/6-1 downto I'length*2/6 );
  piece1_in    <= I( I'length*2/6-1 downto I'length/6 );
  piece0_in    <= I( I'length/6-1 downto 0 );

  piece0_tree : XorTreeStage generic map( piece0_in'length ) port map( piece0_in, piece0_xor );
  piece1_tree : XorTreeStage generic map( piece1_in'length ) port map( piece1_in, piece1_xor );
  piece2_tree : XorTreeStage generic map( piece2_in'length ) port map( piece2_in, piece2_xor );
  piece3_tree : XorTreeStage generic map( piece3_in'length ) port map( piece3_in, piece3_xor );
  piece4_tree : XorTreeStage generic map( piece4_in'length ) port map( piece4_in, piece4_xor );
  piece5_tree : XorTreeStage generic map( piece5_in'length ) port map( piece5_in, piece5_xor );

  XOR6_LUT : LUT6
  generic map(
    INIT => X"6996_9669_9669_6996" )
  port map(
    0 => 0,      I0 => piece0_xor,  I1 => piece1_xor,  I2 => piece2_xor,  I3 => piece3_xor,  I4 => piece4_xor,
    I5 => piece5_xor );
end generate Stages;

end tree_of_xor_lut6;
```



```

-- *****
-- **** STUDENT: 64210382
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin
    Stage_Xor_1: if I'length = 1 generate
        O <= I( 0 );
    end generate Stage_Xor_1;

    Stage_Xor_2: if I'length = 2 generate
        O <= I( 0 ) xor I( 1 );
    end generate Stage_Xor_2;

    Stage_Xor_3: if I'length = 3 generate
        XOR3_LUT3: LUT3
        generic map ( INIT => X"96" )
        port map(
            O => O,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 )

```

```

);
end generate Stage_Xor_3;

Stage_Xor_4: if I'length = 4 generate
XOR4_LUT4: LUT4
generic map ( INIT => X"6996" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 )
);
end generate Stage_Xor_4;

Stage_Xor_5: if I'length = 5 generate
XOR5_LUT5: LUT5
generic map ( INIT => X"9669_6996" )
port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 )
);
end generate Stage_Xor_5;

Stage_Xor_6: if I'length = 6 generate
XOR6_LUT6: LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => I( 5 )
);
end generate Stage_Xor_6;

Stages: if I'length > 6 generate
signal Xor_1, Xor_2, Xor_3, Xor_4, Xor_5, Xor_6: std_logic;
signal Sixth1_in : std_logic_vector( I'length - 1 downto I'length * 5 / 6 );
signal Sixth2_in : std_logic_vector( I'length * 5 / 6 - 1 downto I'length * 4 / 6 );
signal Sixth3_in : std_logic_vector( I'length * 4 / 6 - 1 downto I'length * 3 / 6 );
signal Sixth4_in : std_logic_vector( I'length * 3 / 6 - 1 downto I'length * 2 / 6 );
signal Sixth5_in : std_logic_vector( I'length * 2 / 6 - 1 downto I'length / 6 );
signal Sixth6_in : std_logic_vector( I'length / 6 - 1 downto 0 );
begin
Sixth1_in    <= I( I'length - 1 downto I'length * 5 / 6 );
Sixth2_in    <= I( I'length * 5 / 6 - 1 downto I'length * 4 / 6 );
Sixth3_in    <= I( I'length * 4 / 6 - 1 downto I'length * 3 / 6 );

```

```

Sixth4_in    <= I( I'length * 3 / 6 - 1 downto I'length * 2 / 6 );
Sixth5_in    <= I( I'length * 2 / 6 - 1 downto I'length / 6 );
Sixth6_in    <= I( I'length / 6 - 1 downto 0 );

Tree1: XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, Xor_1 );
Tree2: XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, Xor_2 );
Tree3: XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, Xor_3 );
Tree4: XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, Xor_4 );
Tree5: XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, Xor_5 );
Tree6: XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, Xor_6 );

XOR6_LUT6: LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
    0 => 0,      I0 => Xor_1, I1 => Xor_2, I2 => Xor_3, I3 => Xor_4, I4 => Xor_5, I5 => Xor_6
);
end generate Stages;
end tree_of_xor_lut6;

```

```

-- *****
-- **** STUDENT: 64210384
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural := 64 ); -- Matej Možek: popravil iz 4096 na 64
    port( I: in std_logic_vector( N - 1 downto 0 ); -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

component XorTreeStage
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 ); -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end component;

begin

    stage_xor_1:
    if I' length = 1 generate
    begin
        o <= I( I'left );
    end generate stage_xor_1;

    stage_xor_2:
    if I' length = 2 generate
    begin
        o <= I( I'right ) xor I( I'left );
    end generate stage_xor_2;

```

```

stage_xor_3:
if I' length = 3 generate
begin
xor3_lut3: lut3
generic map ( init => X"96" )
port map ( o => o, I0 => I( 0 ), I1 => I( 1 ), I2 => I( 2 ) );
end generate stage_xor_3;

stage_xor_4:
if I' length = 4 generate
begin
xor4_lut4: lut4
generic map ( init => X"6996" )
port map ( o => o, I0 => I( 0 ), I1 => I( 1 ), I2 => I( 2 ), I3 => I( 3 ) );
end generate stage_xor_4;

stage_xor_5:
if I' length = 5 generate
begin
xor5_lut5: lut5
generic map ( init => X"9669_6996" )
port map ( o => o, I0 => I( 0 ), I1 => I( 1 ), I2 => I( 2 ), I3 => I( 3 ), I4 =>
I( 4 ) );
end generate stage_xor_5;

stage_xor_6:
if I' length = 6 generate
begin
xor6_lut6: lut6
generic map ( init => X"6996_9669_9669_6996" )
port map ( o => o, I0 => I( 0 ), I1 => I( 1 ), I2 => I( 2 ), I3 => I( 3 ), I4 =>
I( 4 ), I5 => I( 5 ) );
end generate stage_xor_6;

stages: if I'length > 6 generate
signal Xor_1, Xor_2, Xor_3, Xor_4, Xor_5, Xor_6: std_logic;
signal Sixth1_in: std_logic_vector ( I' length / 6 - 1 downto 0 );
signal Sixth2_in: std_logic_vector ( 2 * I' length / 6 - 1 downto I' length / 6 );
signal Sixth3_in: std_logic_vector ( I' length/2 - 1 downto 2 * I' length / 6 );
signal Sixth4_in: std_logic_vector ( 2 * I' length / 3 - 1 downto I' length / 2 );

```

[illegible]

```
-- *****
```

```
-- **** STUDENT: 64210386
```

```
-- *****
```

```
-- KOMENTARJI K OCENI NALOGE
```

```
-- Matej Možek:
```

Napačen inicializacijski polinom za LUT3 (pravi je 96).

Napačen inicializacijski polinom za LUT4 (pravi je 6996)

Za LUT3 je konstanta inicializacije 8 bitna - torej x"96" in ne 32 bitna, kot v vašem primeru - dejansko pa xst vrže ven preostale vrednosti nad 7. bitom. Podobna logika velja za LUT4 in LUT5.

Napake sintetizatorja:

```
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 105: Syntax error near "signal".
```

```
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 105: std_logic is not a component
```

```
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 106: Syntax error near "signal".
```

```
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 106: std_logic_vector is not a component
```

```
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 106: Expected an architecture identifier in index
```

```
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 107: Syntax error near "signal".
```

```
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 107: std_logic_vector is not a component
```

```
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 107: Expected an architecture identifier in index
```

```
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 108: Syntax error near "signal".
```

```
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 108: std_logic_vector is not a component
```

```
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 108: Expected an architecture identifier in index
```

```
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 109: Syntax error near "signal".
```

```
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 109: std_logic_vector is not a component
```

```
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 109: Expected an architecture identifier in index
```

```
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 110: Syntax error near "signal".
```

```
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 110: std_logic_vector is not a component
```

```
ERROR:HDLCompiler:402 - "64210386\unary_op_xor_tree_lut6.vhd" Line 110: Expected an architecture identifier in index
```

```
ERROR:HDLCompiler:806 - "64210386\unary_op_xor_tree_lut6.vhd" Line 111: Syntax error near "signal".
```

```
ERROR:HDLCompiler:40 - "64210386\unary_op_xor_tree_lut6.vhd" Line 111: std_logic_vector is not a component
```

Sorry, too many errors..

```
-- *****
```

```
library IEEE;
```

```
use IEEE.STD_LOGIC_1164.ALL;
```

```
entity XorTreeStage is
```

```
    generic ( N : natural := 64 ); -- Matej Možek: popravil iz 4096 na 64
```

```
    port( I: in std_logic_vector( N - 1 downto 0 ); -- vhodni vektor redukcije ima N vhodov
```

```
          O: out std_logic ); -- izhodni bit redukcije
```

```
end XorTreeStage;
```

```

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

component XorTreeStage
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 ); -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end component;

begin
    -- KODO VPISUJETE SEM

    Stage_xor_1:
    if I'length = 1 generate
        O <= I( 0 );
    end generate;

    Stage_xor_2:
    if I'length = 2 generate
    begin
        XOR6_LUT : LUT6
        generic map ( INIT => X"6996_9669_9669_6996" )
        port map (
            O => O,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => '0',      I3 => '0',      I4 => '0',      I5 => '0' );
        end generate Stage_xor_2;

    Stage_xor_3:
    if I'length = 3 generate
    begin
        XOR6_LUT : LUT6
        generic map ( INIT => X"6996_9669_9669_6996" )
        port map (
            O => O,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => '0',      I4 => '0',      I5 => '0' );
        end generate Stage_xor_3;

    Stage_xor_4:
    if I'length = 4 generate
    begin

```



```

XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => '0',      I5 =>
'0' );
end generate Stage_xor_4;

Stage_xor_5:
if I'length = 5 generate
begin
XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => '0' );
end generate Stage_xor_5;

Stage_xor_6:
if I'length = 6 generate
begin
XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => I( 5 ) );
end generate Stage_xor_6;

Stages:
signal Sixth01_in, Sixth02_in, Sixth03_in, Sixth04_in, Sixth05_in, Sixth06_in : std_logic;
signal Sixth01 : std_logic_vector( I'length/6 - 1 downto 0 );
signal Sixth02 : std_logic_vector( I'length/3 - 1 downto I'length/6 );
signal Sixth03 : std_logic_vector( I'length/2 - 1 downto I'length/3 );
signal Sixth04 : std_logic_vector( I'length - I'length/3 - 1 downto I'length/2 );
signal Sixth05 : std_logic_vector( I'length - I'length/6 - 1 downto I'length - I'length/3 );
signal Sixth06 : std_logic_vector( I'length - 1 downto I'length - I'length/6 );
begin

Sixth01      <= I( I'length/6 - 1 downto 0 );
Sixth02      <= I( I'length/3 - 1 downto I'length/6 );
Sixth03      <= I( I'length/2 - 1 downto I'length/3 );

```

```

Sixth04    <= I( I'length - I'length/3 - 1 downto I'length/2 );
Sixth05    <= I( I'length - I'length/6 - 1 downto I'length - I'length/3 );
Sixth06    <= I( I'length - 1 downto I'length - I'length/6 );

```

```

Six01: XorTreeStage generic map ( Sixth01'length ) port map ( Sixth01, Sixth01_in );
Six02: XorTreeStage generic map ( Sixth02'length ) port map ( Sixth02, Sixth02_in );
Six03: XorTreeStage generic map ( Sixth03'length ) port map ( Sixth03, Sixth03_in );
Six04: XorTreeStage generic map ( Sixth04'length ) port map ( Sixth04, Sixth04_in );
Six05: XorTreeStage generic map ( Sixth05'length ) port map ( Sixth05, Sixth05_in );
Six06: XorTreeStage generic map ( Sixth06'length ) port map ( Sixth06, Sixth06_in );

```

```

XOR6_LUT : LUT6

```

```

generic map ( INIT => X"6996_9669_9669_6996" )

```

```

port map (

```

```

0 => 0,      I0 => Sixth01_in,  I1 => Sixth02_in,  I2 => Sixth03_in,  I3 => Sixth04_in,  I4 => Sixth05_in,
I5 => Sixth06_in );

```

```

end generate Stages;

```

```

end tree_of_xor_lut6;

```

```

-- *****
-- **** STUDENT: 64210445
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek:
Napačen inicializacijski polinom za LUT3 (pravi je 96)
Napačen inicializacijski polinom za LUT4 (pravi je 6996)
Za LUT3 je konstanta inicializacije 8 bitna - torej x"96" in ne 32 bitna, kot v vašem primeru - dejansko pa xst vrže
ven preostale vrednosti nad 7. bitom. Podobna logika velja za LUT4 in LUT5.
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural := 64 ); -- Matej Možek: popravil iz 4096 na 64
    port( I: in std_logic_vector( N - 1 downto 0 ); -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 ); -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

    signal in_lut6: std_logic_vector( 5 downto 0 );

begin

    Stage_xor_1:
    if I'length = 1 generate
        O <= I( 0 ); -- xor enega bita je kar ta bit ( x xor 0 ) = x
    end generate;

    Stage_xor_2:

```

```

if I'length = 2 generate
begin
XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => '0',      I3 => '0',      I4 => '0',      I5 => '0' );
end generate Stage_xor_2;

Stage_xor_3:
if I'length = 3 generate
begin
XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => '0',      I4 => '0',      I5 => '0' );
end generate Stage_xor_3;

Stage_xor_4:
if I'length = 4 generate
begin
XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => '0',      I5 =>
'0' );
end generate Stage_xor_4;

Stage_xor_5:
if I'length = 5 generate
begin
XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => '0' );
end generate Stage_xor_5;

Stage_xor_6:
if I'length = 6 generate
begin

```

```

XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => I( 5 ) );
end generate Stage_xor_6;

Stages:
if I'length > 6 generate
signal Sixth1_in, Sixth2_in, Sixth3_in, Sixth4_in, Sixth5_in, Sixth6_in: std_logic;
signal Sixth1 : std_logic_vector( I'length/6 - 1 downto 0 );
signal Sixth2 : std_logic_vector( I'length/3 - 1 downto I'length/6 );
signal Sixth3 : std_logic_vector( I'length/2 - 1 downto I'length/3 );
signal Sixth4 : std_logic_vector( I'length - I'length/3 - 1 downto I'length/2 );
signal Sixth5 : std_logic_vector( I'length - I'length/6 - 1 downto I'length - I'length/3 );
signal Sixth6 : std_logic_vector( I'length - 1 downto I'length - I'length/6 );
begin
-- razdelimo vhodni vektor na polovici ( l - spodnja, u - zgornja )
Sixth1 <= I( I'length/6 - 1 downto 0 );
Sixth2 <= I( I'length/3 - 1 downto I'length/6 );
Sixth3 <= I( I'length/2 - 1 downto I'length/3 );
Sixth4 <= I( I'length - I'length/3 - 1 downto I'length/2 );
Sixth5 <= I( I'length - I'length/6 - 1 downto I'length - I'length/3 );
Sixth6 <= I( I'length - 1 downto I'length - I'length/6 );

-- povežemo polovici
Six1: XorTreeStage generic map ( Sixth1'length ) port map ( Sixth1, Sixth1_in );
Six2: XorTreeStage generic map ( Sixth2'length ) port map ( Sixth2, Sixth2_in );
Six3: XorTreeStage generic map ( Sixth3'length ) port map ( Sixth3, Sixth3_in );
Six4: XorTreeStage generic map ( Sixth4'length ) port map ( Sixth4, Sixth4_in );
Six5: XorTreeStage generic map ( Sixth5'length ) port map ( Sixth5, Sixth5_in );
Six6: XorTreeStage generic map ( Sixth6'length ) port map ( Sixth6, Sixth6_in );
-- Hi_Tree: XorTreeStage generic map ( Hi_Half'length ) port map ( Hi_Half, Hi_Xor );

XOR6_LUT : LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => Sixth1_in,      I1 => Sixth2_in,      I2 => Sixth3_in,      I3 => Sixth4_in,      I4 => Sixth5_in,
I5 => Sixth6_in );

```

```
        -- 0  <= Lo_Xor xor Hi_Xor;      -- polovici združimo z 2-vhodnimi xor vrati  
    end generate Stages;  
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64210455
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
  generic ( N : natural );
  port(
    I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
    O: out std_logic    -- izhodni bit redukcije
  );
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

  component XorTreeStage
    generic ( N : natural );
    port(
      I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
      O: out std_logic    -- izhodni bit redukcije
    );
  end component;

begin

  Single_Input: if I'length = 1 generate
    O    <= I( 0 );
  end generate Single_Input;

  Dual_Input: if I'length = 2 generate
    O    <= I( 0 ) xor I( 1 );
  end generate Dual_Input;

```

```
Triple_Input: if I'length = 3 generate
```

```
LUT3_XOR: LUT3
```

```
generic map ( INIT => X"96" )
```

```
port map (
```

```
0 => 0,
```

```
I0 => I( 0 ),
```

```
I1 => I( 1 ),
```

```
I2 => I( 2 )
```

```
);
```

```
end generate Triple_Input;
```

```
Quad_Input: if I'length = 4 generate
```

```
LUT4_XOR: LUT4
```

```
generic map ( INIT => X"6996" )
```

```
port map (
```

```
0 => 0,
```

```
I0 => I( 0 ),
```

```
I1 => I( 1 ),
```

```
I2 => I( 2 ),
```

```
I3 => I( 3 )
```

```
);
```

```
end generate Quad_Input;
```

```
Quintuple_Input: if I'length = 5 generate
```

```
LUT5_XOR: LUT5
```

```
generic map ( INIT => X"9669_6996" )
```

```
port map (
```

```
0 => 0,
```

```
I0 => I( 0 ),
```

```
I1 => I( 1 ),
```

```
I2 => I( 2 ),
```

```
I3 => I( 3 ),
```

```
I4 => I( 4 )
```

```
);
```

```
end generate Quintuple_Input;
```

```
Sextuple_Input: if I'length = 6 generate
```

```
LUT6_XOR: LUT6
```

```
generic map ( INIT => X"6996_9669_9669_6996" )
```

```
port map (
```



```

0 => 0,
I0 => I( 0 ),
I1 => I( 1 ),
I2 => I( 2 ),
I3 => I( 3 ),
I4 => I( 4 ),
I5 => I( 5 )
);
end generate Sextuple_Input;

MultiStage: if I'length > 6 generate
signal      Xor_1, Xor_2, Xor_3, Xor_4, Xor_5, Xor_6: std_logic;
signal      Sixth1_in : std_logic_vector( I'length - 1 downto I'length * 5 / 6 );
signal      Sixth2_in : std_logic_vector( I'length * 5 / 6 - 1 downto I'length * 4 / 6 );
signal      Sixth3_in : std_logic_vector( I'length * 4 / 6 - 1 downto I'length * 3 / 6 );
signal      Sixth4_in : std_logic_vector( I'length * 3 / 6 - 1 downto I'length * 2 / 6 );
signal      Sixth5_in : std_logic_vector( I'length * 2 / 6 - 1 downto I'length / 6 );
signal      Sixth6_in : std_logic_vector( I'length / 6 - 1 downto 0 );
begin
Sixth1_in   <= I( I'length - 1 downto I'length * 5 / 6 );
Sixth2_in   <= I( I'length * 5 / 6 - 1 downto I'length * 4 / 6 );
Sixth3_in   <= I( I'length * 4 / 6 - 1 downto I'length * 3 / 6 );
Sixth4_in   <= I( I'length * 3 / 6 - 1 downto I'length * 2 / 6 );
Sixth5_in   <= I( I'length * 2 / 6 - 1 downto I'length / 6 );
Sixth6_in   <= I( I'length / 6 - 1 downto 0 );

XOR_Stage1: XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, Xor_1 );
XOR_Stage2: XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, Xor_2 );
XOR_Stage3: XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, Xor_3 );
XOR_Stage4: XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, Xor_4 );
XOR_Stage5: XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, Xor_5 );
XOR_Stage6: XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, Xor_6 );

Final_XOR_LUT6: LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,
I0 => Xor_1,
I1 => Xor_2,
I2 => Xor_3,

```

```
I3 => Xor_4,  
I4 => Xor_5,  
I5 => Xor_6  
);  
end generate MultiStage;  
  
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64210457
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek:
Napačen inicializacijski polinom za LUT3 (pravi je 96)
-- *****

library IEEE;
library unisim;
use IEEE.STD_LOGIC_1164.ALL;
use unisim.vcomponents.all;

entity XorTreeStage is
  generic ( N : natural );
  port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
        O: out std_logic );                        -- izhodni bit redukcije
end XorTreeStage;

architecture tree_of_xor_lut6 of XorTreeStage is

  signal      in_lut6 : std_logic_vector( 5 downto 0 );
  signal      in_lut5 : std_logic_vector( 4 downto 0 );
  signal      in_lut4 : std_logic_vector( 3 downto 0 );
  signal      in_lut3 : std_logic_vector( 2 downto 0 );

          signal      O1, O2, O3, O4, O5, O6 : std_logic;

begin

  -- Stage for 1-bit input
  Stage_xor_1: if I'length = 1 generate
  begin
    O <= I( I'left );    -- XOR of a single bit is the bit itself
  end generate Stage_xor_1;

  -- Stage for 2-bit input
  Stage_xor_2: if I'length = 2 generate
  begin
    O <= I( 0 ) xor I( 1 );    -- XOR of two bits
  end generate Stage_xor_2;

```

```

    -- Stage for 3-bit input
Stage_xor_3: if I'length = 3 generate
begin
in_lut3      <= I( 2 downto 0 );
XOR3_LUT : LUT3
generic map( INIT => X"69" )
port map( 0 => 0, I0 => in_lut3( 0 ), I1 => in_lut3( 1 ), I2 => in_lut3( 2 ) );
end generate Stage_xor_3;

    -- Stage for 4-bit input
Stage_xor_4: if I'length = 4 generate
begin
in_lut4      <= I( 3 downto 0 );
XOR4_LUT : LUT4
generic map( INIT => X"6996" )
port map( 0 => 0, I0 => in_lut4( 0 ), I1 => in_lut4( 1 ), I2 => in_lut4( 2 ), I3 => in_lut4( 3 ) );
end generate Stage_xor_4;

    -- Stage for 5-bit input
Stage_xor_5: if I'length = 5 generate
begin
in_lut5      <= I( 4 downto 0 );
XOR5_LUT : LUT5
generic map( INIT => X"6996_9669" )
port map( 0 => 0, I0 => in_lut5( 0 ), I1 => in_lut5( 1 ), I2 => in_lut5( 2 ), I3 => in_lut5( 3 ), I4 => in_lut5( 4 )
);
end generate Stage_xor_5;

    -- Stage for 6-bit input
Stage_xor_6: if I'length = 6 generate
begin
in_lut6      <= I( 5 downto 0 );
XOR6_LUT : LUT6
generic map( INIT => X"6996_9669_9669_6996" )
port map( 0 => 0, I0 => in_lut6( 0 ), I1 => in_lut6( 1 ), I2 => in_lut6( 2 ), I3 => in_lut6( 3 ), I4 => in_lut6( 4 ),
I5 => in_lut6( 5 ) );
end generate Stage_xor_6;

    -- Recursive stage for input length greater than 6

```

```
Stages: if I'length > 6 generate
```

```
    signal Sixth1_in : std_logic_vector( I'length - 1 downto I'length * 5 / 6 );
    signal Sixth2_in : std_logic_vector( I'length * 5 / 6 - 1 downto I'length * 4 / 6 );
    signal Sixth3_in : std_logic_vector( I'length * 4 / 6 - 1 downto I'length * 3 / 6 );
    signal Sixth4_in : std_logic_vector( I'length * 3 / 6 - 1 downto I'length * 2 / 6 );
    signal Sixth5_in : std_logic_vector( I'length * 2 / 6 - 1 downto I'length / 6 );
    signal Sixth6_in : std_logic_vector( I'length / 6 - 1 downto 0 );
```

```
begin
```

```
    -- Divide the input vector into six parts
```

```
    Sixth1_in  <= I( I'length - 1 downto 5*I'length/6 );
    Sixth2_in  <= I( 5*I'length/6-1 downto 4*I'length/6 );
    Sixth3_in  <= I( 4*I'length/6-1 downto 3*I'length/6 );
    Sixth4_in  <= I( 3*I'length/6-1 downto 2*I'length/6 );
    Sixth5_in  <= I( 2*I'length/6-1 downto I'length/6 );
    Sixth6_in  <= I( I'length/6-1 downto 0 );
```

```
    -- Instantiate XorTreeStage components for each part
```

```
    XorTreeStage1: entity work.XorTreeStage generic map( N => Sixth1_in'length ) port map( I => Sixth1_in, O => O1 );
    XorTreeStage2: entity work.XorTreeStage generic map( N => Sixth2_in'length ) port map( I => Sixth2_in, O => O2 );
    XorTreeStage3: entity work.XorTreeStage generic map( N => Sixth3_in'length ) port map( I => Sixth3_in, O => O3 );
    XorTreeStage4: entity work.XorTreeStage generic map( N => Sixth4_in'length ) port map( I => Sixth4_in, O => O4 );
    XorTreeStage5: entity work.XorTreeStage generic map( N => Sixth5_in'length ) port map( I => Sixth5_in, O => O5 );
    XorTreeStage6: entity work.XorTreeStage generic map( N => Sixth6_in'length ) port map( I => Sixth6_in, O => O6 );
```

```
    -- Combine the outputs of the stages with LUT 6
```

```
    XOR6_LUT : LUT6
    generic map( INIT => X"6996_9669_9669_6996" )
    port map( 0 => O, I0 => O1, I1 => O2, I2 => O3, I3 => O4, I4 => O5, I5 => O6 );
```

```
end generate Stages;
```

```
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64240429
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin
    -- KODO VPISUJETE SEM
    Stage_xor_1:
    if I'length = 1 generate
    begin
        O <= I( I'left );    -- xor enega bita std_logic_vector je kar ta bit ( x xor 0 ) = x
        -- ta stavek obenem opravlja pretvorbo std_logic_vector( 0 downto 0 ) v tip izhoda ( 0 ), ki je std_logic
    end generate Stage_xor_1;

    Stage_xor_2:
    if I'length = 2 generate
    begin
        O <= I( I'right ) xor I( I'left ); -- xor dvobitnega vektorja std_logic_vector: desni bit xor levi bit
    end generate Stage_xor_2;

```

```

Stage_xor_3:
if I'length = 3 generate
begin

XOR3_LUT3: LUT3
generic map ( INIT => X"96" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 )
);

end generate Stage_xor_3;

Stage_xor_4:
if I'length = 4 generate
begin

XOR4_LUT4: LUT4
generic map ( INIT => X"6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 )
);

end generate Stage_xor_4;

Stage_xor_5:
if I'length = 5 generate
begin

XOR6_LUT5: LUT5
generic map ( INIT => X"9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 )
);

end generate Stage_xor_5;

Stage_xor_6:
if I'length = 6 generate
begin

```

```

XOR6_LUT6: LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => I( 5 )
);

end generate Stage_xor_6;

Stages: if I'length > 6 generate
-- signal    Lo_Xor, Hi_Xor: std_logic;
-- signal    Lo_Half : std_logic_vector( I'length/2 - 1 downto 0 );
-- signal    Hi_Half : std_logic_vector( I'length - 1 downto I'length/2 );

-- input vector is split into 6 equal parts
signal Sixth1_in : std_logic_vector( I'length - 1 downto I'length * 5 / 6 );
signal Sixth2_in : std_logic_vector( I'length * 5 / 6 - 1 downto I'length * 4 / 6 );
signal Sixth3_in : std_logic_vector( I'length * 4 / 6 - 1 downto I'length * 3 / 6 );
signal Sixth4_in : std_logic_vector( I'length * 3 / 6 - 1 downto I'length * 2 / 6 );
signal Sixth5_in : std_logic_vector( I'length * 2 / 6 - 1 downto I'length / 6 );
signal Sixth6_in : std_logic_vector( I'length / 6 - 1 downto 0 );

-- outputs of each of those six parts:
signal Sixth1_out, Sixth2_out, Sixth3_out, Sixth4_out, Sixth5_out, Sixth6_out: std_logic;

begin
-- razdelimo vhodni vektor na polovici ( Lo spodnja, Hi zgornja )
-- Lo_Half  <= I( I'length/2 - 1 downto 0 );

-- pri lihih vrednostih doline vhodnega vektorja ( I ) dodatni element pripiemo zg. polovici
-- Hi_Half  <= I( I'length - 1 downto I'length/2 );

-- poveemo polovici z rekurzivnim povezovalnim stavkom
-- Lo_Tree: XorTreeStage generic map ( Lo_Half'length ) port map ( Lo_Half, Lo_Xor );
-- Hi_Tree: XorTreeStage generic map ( Hi_Half'length ) port map ( Hi_Half, Hi_Xor );
-- O  <= Lo_Xor xor Hi_Xor; polovici zdruimo z 2vhodnimi xor vrati

-- assign the separated input signal's parts to the inputs of the next stages
Sixth1_in  <= I( I'length - 1 downto I'length * 5 / 6 );
Sixth2_in  <= I( I'length * 5 / 6 - 1 downto I'length * 4 / 6 );

```



```

Sixth3_in    <= I( I'length * 4 / 6 - 1 downto I'length * 3 / 6 );
Sixth4_in    <= I( I'length * 3 / 6 - 1 downto I'length * 2 / 6 );
Sixth5_in    <= I( I'length * 2 / 6 - 1 downto I'length / 6 );
Sixth6_in    <= I( I'length / 6 - 1 downto 0 );

-- connect those divided input signals and outputs to a next XorTreeStage
Branch1: XorTreeStage
generic map (
  Sixth1_in'length
)
port map (
  0 => Sixth1_out,    I => Sixth1_in
);
Branch2: XorTreeStage
generic map (
  Sixth2_in'length
)
port map (
  0 => Sixth2_out,    I => Sixth2_in
);
Branch3: XorTreeStage
generic map (
  Sixth3_in'length
)
port map (
  0 => Sixth3_out,    I => Sixth3_in
);
Branch4: XorTreeStage
generic map (
  Sixth4_in'length
)
port map (
  0 => Sixth4_out,    I => Sixth4_in
);
Branch5: XorTreeStage
generic map (
  Sixth5_in'length
)
port map (
  0 => Sixth5_out,    I => Sixth5_in
);

```

```

    );
Branch6: XorTreeStage
generic map (
  Sixth6_in'length
)
port map (
  0 => Sixth6_out,    I => Sixth6_in
);

-- outputs of THIS ( the one on this floor of the recursion ) are then fed into the
-- final loop of the entire XOR element. If there were more levels, they are created
-- inside and their outputs are connected to the final LUT 6 of their floor of the
-- recursion -- took me long enough...

XOR6_LUT6: LUT6
generic map ( INIT => X"6996_9669_9669_6996" )
port map (
  0 => 0,      I0 => Sixth1_out,  I1 => Sixth2_out,  I2 => Sixth3_out,  I3 => Sixth4_out,  I4 => Sixth5_out,
  I5 => Sixth6_out
);

end generate Stages;

end tree_of_xor_lut6;

```

```

-- *****
-- **** STUDENT: 64240430
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin

    Stage_xor_1:
        if I'length = 1 generate
        begin
            O <= I( I'left );
        end generate Stage_xor_1;

    Stage_xor_2:
        if I'length = 2 generate
        begin
            O <= I( I'left ) xor I( I'right );
        end generate Stage_xor_2;

```

```

Stage_xor_3:
  if I'length = 3 generate
  begin
    XOR3_LUT : LUT3
    -- 2^3 = 8 ( 0 ... 7 )
    -- INIT = 96
    generic map ( INIT => X"96" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ) );
    end generate Stage_xor_3;

Stage_xor_4:
  if I'length = 4 generate
  begin
    XOR4_LUT : LUT4
    -- 2^4 = 16 ( 0 ... 15 )
    -- INIT = 6996
    generic map ( INIT => X"6996" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ) );
    end generate Stage_xor_4;

Stage_xor_5:
  if I'length = 5 generate
  begin
    XOR5_LUT : LUT5
    -- 2^5 = 32 ( 0 ... 31 )
    -- INIT = 9669 6996
    generic map ( INIT => X"9669_6996" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ) );
    end generate Stage_xor_5;

Stage_xor_6:
  if I'length = 6 generate
  begin
    XOR6_LUT : LUT6
    -- 2^6 = 64 ( 0 ... 63 )
    -- INIT = 6996 9669 9669 6996
    generic map ( INIT => X"6996_9669_9669_6996" )

```

```

port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => I( 5 ) );
end generate Stage_xor_6;

```

Stages:

```

if I'length > 6 generate
signal Sixth1_in : std_logic_vector( I'length/6 - 1 downto 0 );
signal Sixth2_in : std_logic_vector( I'length/3 - 1 downto I'length/6 );
signal Sixth3_in : std_logic_vector( I'length/2 - 1 downto I'length/3 );
signal Sixth4_in : std_logic_vector( 4*I'length/6 - 1 downto I'length/2 );
signal Sixth5_in : std_logic_vector( 5*I'length/6 - 1 downto 4*I'length/6 );
signal Sixth6_in : std_logic_vector( I'length - 1 downto 5*I'length/6 );
signal      Sixth1_out, Sixth2_out, Sixth3_out, Sixth4_out, Sixth5_out, Sixth6_out : std_logic;

begin

Sixth1_in    <= I( I'length/6 - 1 downto 0 );
Sixth2_in    <= I( I'length/3 - 1 downto I'length/6 );
Sixth3_in    <= I( I'length/2 - 1 downto I'length/3 );
Sixth4_in    <= I( 4*I'length/6 - 1 downto I'length/2 );
Sixth5_in    <= I( 5*I'length/6 - 1 downto 4*I'length/6 );
Sixth6_in    <= I( I'length - 1 downto 5*I'length/6 );

xor_tree1 : XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, Sixth1_out );
xor_tree2 : XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, Sixth2_out );
xor_tree3 : XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, Sixth3_out );
xor_tree4 : XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, Sixth4_out );
xor_tree5 : XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, Sixth5_out );
xor_tree6 : XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, Sixth6_out );

XOR6_LUT : LUT6
-- 2^6 = 64 ( 0 ... 63 )
-- INIT = 6996 9669 9669 6996
generic map ( INIT => X"6996_9669_9669_6996" )
port map(
0 => 0,      I0 => Sixth1_out,  I1 => Sixth2_out,  I2 => Sixth3_out,  I3 => Sixth4_out,  I4 => Sixth5_out,
I5 => Sixth6_out );
end generate Stages;

```

```
end tree_of_xor_lut6;
```

```

-- *****
-- **** STUDENT: 64210113
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic (N : natural);
    port( I: in std_logic_vector(N - 1 downto 0); -- vhodni vektor redukcije ima N vhodov
          O: out std_logic); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic (N : natural);
        port( I: in std_logic_vector(N - 1 downto 0); -- vhodni vektor redukcije ima N vhodov
              O: out std_logic); -- izhodni bit redukcije
    end component;

    signal lut6_in : std_logic_vector(5 downto 0);
    signal lut5_in : std_logic_vector(5 downto 0);
    signal lut4_in : std_logic_vector(3 downto 0);
    signal lut3_in : std_logic_vector(2 downto 0);

begin

    Stage_Xor_1: if I'length = 1 generate
        O <= I(0);
    end generate Stage_Xor_1;

```

```
Stage_Xor_2: if I'length = 2 generate
    0 <= I(0) xor I(1);
end generate Stage_Xor_2;
```

```
Stage_Xor_3: if I'length = 3 generate
    lut3_in <= I(2 downto 0);
    LUT3_XOR: LUT3
        generic map(INIT => X"96")
        port map(0 => 0,
            I0 => lut3_in(0),
            I1 => lut3_in(1),
            I2 => lut3_in(2));
end generate Stage_Xor_3;
```

```
Stage_Xor_4: if I'length = 4 generate
    lut4_in <= I(3 downto 0);
    LUT4_XOR: LUT4
        generic map(INIT => X"6996")
        port map(0 => 0,
            I0 => lut4_in(0),
            I1 => lut4_in(1),
            I2 => lut4_in(2),
            I3 => lut4_in(3));
end generate Stage_Xor_4;
```

```
Stage_Xor_5: if I'length = 5 generate
    lut5_in <= I(4 downto 0);
    LUT5_XOR: LUT5
        generic map(INIT => X"9669_6996")
        port map(0 => 0,
            I0 => lut5_in(0),
            I1 => lut5_in(1),
            I2 => lut5_in(2),
            I3 => lut5_in(3),
            I4 => lut5_in(4));
end generate Stage_Xor_5;
```



```

Stage_Xor_6: if I'length = 6 generate
    lut6_in <= I(5 downto 0);
    LUT6_XOR: LUT6
        generic map(INIT => X"6996_9669_9669_6996")
        port map(0 =>0,
            I0 =>lut6_in(0),
            I1 =>lut6_in(1),
            I2 =>lut6_in(2),
            I3 =>lut6_in(3),
            I4 =>lut6_in(4),
            I5 =>lut6_in(5));
end generate Stage_Xor_6;

```

```

Stages: if I'length > 6 generate
    signal xor1, xor2, xor3, xor4, xor5, xor6: std_logic;
    signal Sixth1_in : std_logic_vector(I'length - 1 downto I'length*5/6);
    signal Sixth2_in : std_logic_vector((I'length*5/6) - 1 downto I'length*4/6);
    signal Sixth3_in : std_logic_vector((I'length*4/6) - 1 downto I'length*3/6);
    signal Sixth4_in : std_logic_vector((I'length*3/6) - 1 downto I'length*2/6);
    signal Sixth5_in : std_logic_vector((I'length*2/6) - 1 downto I'length/6);
    signal Sixth6_in : std_logic_vector((I'length/6) - 1 downto 0);

    begin

        Sixth1_in <=I((I'length - 1) downto I'length * 5 / 6);
        Sixth2_in <=I((I'length*5/6) - 1 downto I'length * 4 / 6);
        Sixth3_in <=I((I'length*4/6) - 1 downto I'length * 3 / 6);
        Sixth4_in <=I((I'length*3/6) - 1 downto I'length * 2 / 6);
        Sixth5_in <=I((I'length*2/6) - 1 downto I'length / 6);
        Sixth6_in <=I((I'length/6) - 1 downto 0);

        TreeStage_Xor_1: XorTreeStage generic map (Sixth1_in'length) port map (I =>Sixth1_in,0 =>xor1);
        TreeStage_Xor_2: XorTreeStage generic map (Sixth2_in'length) port map (I =>Sixth2_in,0 =>xor2);
        TreeStage_Xor_3: XorTreeStage generic map (Sixth3_in'length) port map (I =>Sixth3_in,0 =>xor3);
        TreeStage_Xor_4: XorTreeStage generic map (Sixth4_in'length) port map (I =>Sixth4_in,0 =>xor4);
        TreeStage_Xor_5: XorTreeStage generic map (Sixth5_in'length) port map (I =>Sixth5_in,0 =>xor5);
        TreeStage_Xor_6: XorTreeStage generic map (Sixth6_in'length) port map (I =>Sixth6_in,0 =>xor6);
    end generate Stages;

```

```
XOR6_LUT6:LUT6
  generic map (INIT => X"6996_9669_9669_6996")
  port map (0 => 0,
            I0 => xor1,
            I1 => xor2,
            I2 => xor3,
            I3 => xor4,
            I4 => xor5,
            I5 => xor6);

    end generate Stages;
end tree_of_xor_lut6;
```

```

-- *****
-- **** PREDLOGA VAJE
-- *****
-- KOMENTARJI K OCENI NALOGE
-- Matej Možek: Ni pripomb
-- *****
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity XorTreeStage is
    generic ( N : natural );
    port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
          O: out std_logic ); -- izhodni bit redukcije
end XorTreeStage;

library unisim;
use unisim.vcomponents.all;

architecture tree_of_xor_lut6 of XorTreeStage is

    component XorTreeStage
        generic ( N : natural );
        port( I: in std_logic_vector( N - 1 downto 0 );    -- vhodni vektor redukcije ima N vhodov
              O: out std_logic ); -- izhodni bit redukcije
    end component;

begin

    Stage_xor_1:
        if I'length = 1 generate
            0      <= I( 0 );    -- xor enega bita je kar ta bit ( x xor 0 ) = x
        end generate;

    Stage_xor_2:
        if I'length = 2 generate
            begin
                XOR2_LUT : LUT2
                generic map(
                    INIT => X"6" )
                port map(

```

```

    0 => 0,      I0 => I( 0 ),      I1 => I( 1 )
  );
end generate;

```

```

Stage_xor_3:
  if I'length = 3 generate
  begin
    XOR3_LUT : LUT3
    generic map(
      INIT => X"96" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ) );
  end generate;

```

```

Stage_xor_4:
  if I'length = 4 generate
  begin
    XOR4_LUT : LUT4
    generic map(
      INIT => X"6996" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ) );
  end generate;

```

```

Stage_xor_5:
  if I'length = 5 generate
  begin
    XOR5_LUT : LUT5
    generic map(
      INIT => X"9669_6996" )
    port map(
      0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ) );
  end generate;

```

```

Stage_LUT6:
  if I'length = 6 generate
  begin
    XOR6_LUT : LUT6
    generic map(
      INIT => X"6996_9669_9669_6996" )
  end generate;

```

```

port map(
0 => 0,      I0 => I( 0 ),      I1 => I( 1 ),      I2 => I( 2 ),      I3 => I( 3 ),      I4 => I( 4 ),
I5 => I( 5 ) );
end generate Stage_LUT6;

```

Stages:

```

if I'length > 6 generate
signal Sixth1_in : std_logic_vector( I'length/6 - 1 downto 0 );
signal Sixth2_in : std_logic_vector( 2*I'length/6 - 1 downto I'length/6 );
signal Sixth3_in : std_logic_vector( 3*I'length/6 - 1 downto 2*I'length/6 );
signal Sixth4_in : std_logic_vector( 4*I'length/6 - 1 downto 3*I'length/6 );
signal Sixth5_in : std_logic_vector( 5*I'length/6 - 1 downto 4*I'length/6 );
signal Sixth6_in : std_logic_vector( I'length - 1 downto 5*I'length/6 );
signal Sixth1_out, Sixth2_out, Sixth3_out, Sixth4_out, Sixth5_out, Sixth6_out : std_logic;
begin
-- razdelimo vhodni vektor na šestine
Sixth1_in  <= I( I'length/6 - 1 downto 0 );
Sixth2_in  <= I( 2*I'length/6 - 1 downto I'length/6 );
Sixth3_in  <= I( 3*I'length/6 - 1 downto 2*I'length/6 );
Sixth4_in  <= I( 4*I'length/6 - 1 downto 3*I'length/6 );
Sixth5_in  <= I( 5*I'length/6 - 1 downto 4*I'length/6 );
Sixth6_in  <= I( I'length - 1 downto 5*I'length/6 );

-- povežemo šestine
Sixth1_Tree: XorTreeStage generic map ( Sixth1_in'length ) port map ( Sixth1_in, Sixth1_out );
Sixth2_Tree: XorTreeStage generic map ( Sixth2_in'length ) port map ( Sixth2_in, Sixth2_out );
Sixth3_Tree: XorTreeStage generic map ( Sixth3_in'length ) port map ( Sixth3_in, Sixth3_out );
Sixth4_Tree: XorTreeStage generic map ( Sixth4_in'length ) port map ( Sixth4_in, Sixth4_out );
Sixth5_Tree: XorTreeStage generic map ( Sixth5_in'length ) port map ( Sixth5_in, Sixth5_out );
Sixth6_Tree: XorTreeStage generic map ( Sixth6_in'length ) port map ( Sixth6_in, Sixth6_out );

XOR6_LUT : LUT6
generic map(
INIT => X"6996966996696996" )
port map(
0 => 0,      I0 => Sixth1_out,  I1 => Sixth2_out,  I2 => Sixth3_out,  I3 => Sixth4_out,  I4 => Sixth5_out,
I5 => Sixth6_out );

end generate Stages;
end tree_of_xor_lut6;

```