Tento článek je původním rukopisem textu publikovaného v časopise DPS Elektronika A-Z:
J. Šťastný. Simulace číslicových obvodů na hradlové úrovni: model návrhu, DPS Elektronika od A do Z, pp. 6 - 12, leden/únor 2016

Simulace číslicových obvodů na hradlové úrovni: model návrhu

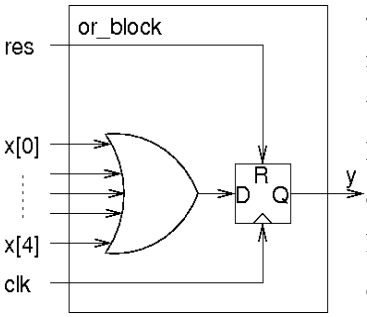
Jakub Šťastný

ASICentrum, s.r.o.

Katedra teorie obvodů FEL ČVUT Praha

1 Úvod

Při provádění simulací na hradlové úrovni je někdy třeba sestoupit i na nižší úrovně abstrakce a začít zkoumat popis návrhu vygenerovaný návrhovými nástroji. Proto zatímco v předchozím článku v čísle 3/2015 jsme se zabývali spíše teorií, nyní se zaměříme na praktické aspekty modelování obvodů na hradlové úrovni. Ukážeme si jednoduchý příklad obvodu navrženého na RTL úrovni a posléze rozebereme jeho simulační model po rozmístění a propojení s kompletní informací o zpoždění na všech prvcích obvodu (*post-implementation timing simulation*). Poznamenejme zde, že návrhové nástroje umožňují také simulace s jinými úrovněmi detailů, jejich použití je ale omezené (např. pro FPGA Xilinx viz [1], strana 17, stručný výčet základních možností byl i v článku [2]). Pro demonstraci je využita FPGA platforma Xilinx; výběr je dán osobní zkušeností autora článku. Poznatky prezentované v článku jsou ovšem aplikovatelné jak při návrhu číslicových systémů na FPGA platformě, tak při návrhu zákaznických integrovaných obvodů.



Obrázek 1: Schéma demonstračního návrhu na RTL úrovni. Obrázek or_rtl.eps.

Jako příklad použijeme jednoduchý obvod s pětivstupovým hradlem OR a registrem na jeho výstupu; schéma obvodu na RTL úrovni je na obrázku 1. Jednoduchost je zde vynucena omezeným místem dostupným na stránkách časopisu, i tak bude popis na hradlové úrovni rozsáhlý. Přesto lze na tomto jednoduchém obvodu demonstrovat vše důležité. RTL kód demonstračního obvodu lze nalézt na obrázku 2.

Návrh je po syntéze a rozmístění a propojení popsán pro simulaci ve formě schématu obvodu vyjádřeného ve zdrojovém kódu (*netlist*) spolu s tzv. SDF (*Standard Delay File*) souborem. Ten obsahuje informace o zpoždění na obvodových prvcích a spojích spolu s časovými parametry klopných obvodů. Oba soubory jsou potřebné pro vybudování kompletního simulačního modelu návrhu. Spustíme tedy v prostředí Xilinx Vivado jak syntézu, tak rozmístění a propojení našeho demonstračního obvodu do FPGA obvodu *xa7a15tpcg236-1I* a následně také časovou simulaci. Dojde při tom k vygenerování následujících souborů:

- soubor s netlistem v jazyce Verilog s názvem *or_block_time_impl.v* (v prostředí Xilinx Vivado ho lze v projektu nalézt v adresáři *jmeno_projektu.sim\sim_1\impl\timing*).
- SDF soubor s názvem *or_block_time_impl.sdf* (lze nalézt ve stejném adresáři jako netlist).

Obsah obou souborů nyní detailněji popíšeme.

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

ENTITY or_block IS
  PORT (
    clk : IN std_logic;
    res : IN std_logic;
    x   : IN  std_logic_vector (4 DOWNTO 0);
    y   : OUT std_logic
  );
END ENTITY or_block;

ARCHITECTURE rtl OF or_block IS
  SIGNAL y_d : std_logic;
BEGIN
  y_d <= x(0) OR x(1) OR x(2) OR x(3) OR x(4);
  reg : PROCESS (clk, res)
  BEGIN
    IF (res = '1') THEN
      y <= '0';
    ELSIF clk='1' AND clk'EVENT THEN
      y <= y_d;
    END IF;
  END PROCESS reg;
END ARCHITECTURE rtl;
```

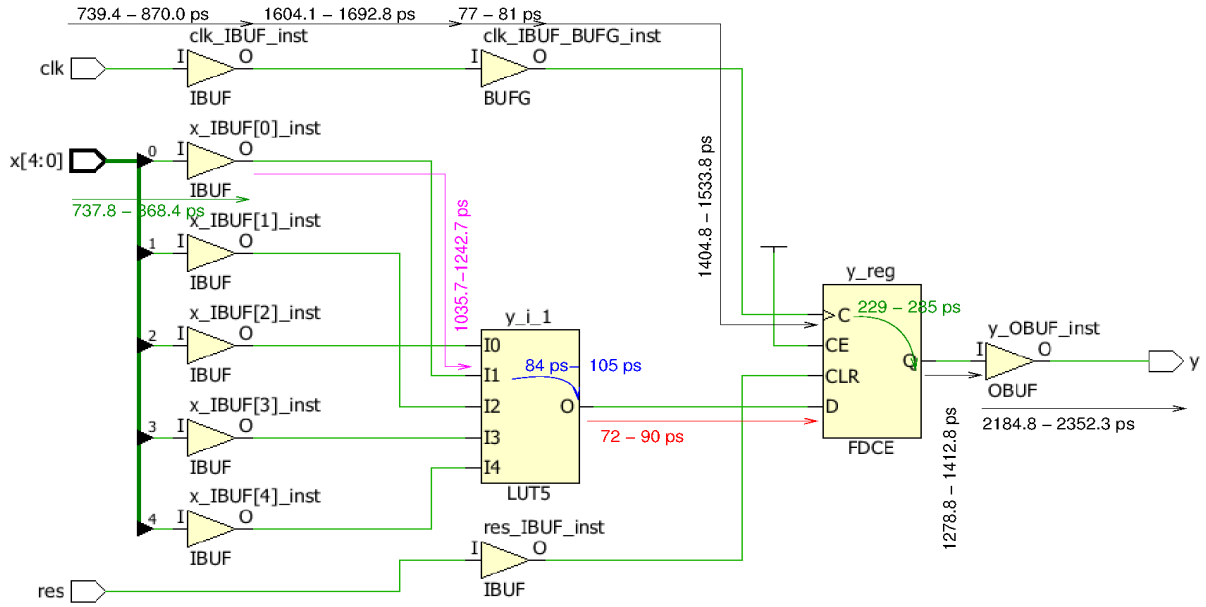
Obrázek 2: Návrh demonstračního obvodu na RTL úrovni.

2 Netlist

Grafické schéma demonstračního obvodu po syntéze, rozmístění a propojení lze nalézt v obrázku 3, netlist potom v obrázku 4. Srovnáte-li netlist se schématem, můžete vidět, že je jeho textovou reprezentací; netlist obsahuje přímo instance modelů technologických buněk dostupných v použitém FPGA obvodu ze kterých se skládá návrh. Modely buněk potom simulují jak zpoždění na buňkách, tak kontrolují dodržení časových parametrů klopných obvodů. Popišme nyní nejzajímavější části netlistu:

- součástí netlistu je volání *$sdf_annotate* zvýrazněné v kódu modrou barvou. Pomocí něj jsou nastaveny časové parametry modelů jednotlivých buněk z SDF souboru. Tomuto procesu se říká zpětná anotace (*back annotation*) a provádí se před startem vlastní simulace (viz např. [3], strana 412). Poznamenejme, že během anotace může simulátor vypsat řadu varování, ta by měla být jedno po druhém zkontrolována návrhovým týmem a vysvětlena či návrh patřičně upraven. Doporučujeme také při prvním startu simulace na hradlové úrovni zkontrolovat, že opravdu k anotaci došlo – například tak, že v simulaci v časových průbězích v okně *Wave* namátkou zkontrolujeme, že náhodně vybrané logické prvky mají očekávaná zpoždění.
- zelenou barvou je zvýrazněna instance tabulky (LUT – *Look Up Table*) *y_i_1* implementující vlastní kombinační funkci *OR*. Všimněte si binárního řetězce v parametru *INIT – 0xFFFFFFFE*. Tak je definován obsah tabulky adresované jejími vstupy *I0-I4*, t.j. vstupy obvodu v pořadí *x(2)*, *x(0)*, *x(1)*, *x(3)*, *x(4)* – tabulka je tedy o velikosti 32 bitů. Její výstup je v logické nule jen tehdy, jsou-li všechny vstupy tabulky v logické nule (pak je adresovaný bit 0, který je v obsahu tabulky jediný v log. 0) – to je implementace logické funkce OR.
- fialovou barvou je označená direktiva *timescale*, která definuje interpretaci časových údajů v souboru a časovou jednotku simulátoru, více viz [4].
- červeně je zvýrazněn kód, který inicializuje celý návrh po startu simulace a simuluje tak globální reset FPGA

obvodu. Globální reset trvá 100 ns během kterých model návrhu v FPGA obvodu nebude reagovat na vnější podněty (více viz např. [1], strana 13).



Obrázek 3: Schéma implementace RTL kódu z obrázku 2 a příklad časování v návrhu. Barevně označené časové parametry jsou stejně obarveny i ve výpisu na obrázku 7. Soubor or_timing.eps

3 SDF Soubor

Výpis SDF souboru pro demonstrační obvod je na obrázku 7; důležité části jsou opět vyznačeny barevně. SDF soubor lze zhruba rozdělit do dvou částí, ty jsou ve výpisu odděleny zvýrazněnými komentáři:

1. **hlavička souboru** obsahuje informace, které popisují soubor jako celek. V příkladu můžeme vidět řadu záznamů, jejichž význam je na první pohled patrný; zastavme se tedy jen u dvou zajímavějších: *TIMSECALE* a *DIVIDER*. Pole *TIMESCALE* je ekvivalent direktivy *timescale* zmíněné výše a definuje jednotku použitou pro interpretaci všech záznamů v SDF souboru. V našem případě je v poli *TIMESCALE* uvedena 1 ps, tedy všechny hodnoty v SDF souboru je třeba násobit jednou pikosekundou, aby byl získán skutečný čas, který reprezentují. Pole *DIVIDER* definuje znak použitý pro oddělení úrovní hierarchie v cestách k identifikujících jednotlivé prvky v netlistu. V hlavičce SDF souboru může být někdy uveden i technologický roh pro který je SDF generováno – pole *PROCESS*, *VOLTAGE* a *TEMPERATURE*, v našem příkladě ale tato použita nejsou. Více detailů k záznamům v hlavičce SDF souboru lze nalézt v dokumentu [5]. Poznamenejme ještě, že komentář se píše buď za dvě lomítka (//), nebo mezi /* a */.
2. **popis jednotlivých buněk**. Po hlavičce následují v souboru jednotlivé záznamy uvozené klíčovým slovem *CELL*. Ty popisují zpoždění a časové parametry jednotlivých buněk v návrhu. V dalším textu detailněji popíšeme dva takové typické záznamy. Zvláštním případem je v popisu poslední záznam, který definuje zpoždění na spojích (zde) na nejvyšší úrovni hierarchie pomocí konstrukce *INTERCONNECT*.

V dalším textu uvidíte, že časové parametry logických obvodů jsou často uváděny jako trojice hodnot, dovolíme si proto malou vsuvku. Časové parametry číslicového obvodu se často uvádí ve třech konfiguracích – tzv. rozích (*corners*):

- **konfigurace pro nejpomalejší obvodové prvky** (tzv. *slow corner*, nebo *max corner* – maximum ve smyslu maximálního zpoždění). Simulace s nejpomalejšími prvky jsou užitečné pro kontrolu dodržení předstihu (*setup check*) na sekvenčních prvcích v obvodu. Napájecí napětí je zde nejmenší, jaké dokáže v aplikaci dodat napájecí zdroj, teplota nejvyšší možná (ovšem ne nutně vždy) a výrobní proces nejpomalejší možný.
- **konfigurace pro nejrychlejší obvodové prvky** (tzv. *fast corner*, nebo *min corner*). Simulace s nejrychlejšími prvky je praktická pro ověření dodržení přesahu (*hold check*) na všech prvcích v obvodu. Napájecí napětí je zde obvykle typicky největší možné, teplota nejnižší možná a výrobní proces nejrychlejší možný.
- **Konfigurace pro obvodové prvky za typických podmínek** (tzv. *typical corner*). Typické obvodové podmínky se nepoužívají tak často jako min/max rohy. Za typických podmínek má napájecí napětí nominální hodnotu, teplota se často používá pokojová a výrobní proces v tomto případě vede na hradla s typickým zpožděním.

Podívejme se nyní blíže na dva příklady popisu zpoždění.

3.1 Kombinační buňka

Zabývat se budeme pětivstupovou tabulkou *y_i_1*, v SDF souboru označenou komentářem. Popis začíná záznamem *CELL*, následující záznam *CELLTYPE* přímo definuje typ instancované buňky (jméno entity, který je instancována). V našem případě se jedná o *LUT5*, která je definovaná v technologické knihovně pro FPGA obvod. Nejzajímavější části popisu jsou následující:

- Záznam *DELAY* definuje samotná zpoždění v buňce, ta jsou určena řadou záznamů *IOPATH*. Například *IOPATH I1 O* definuje zpoždění v cestě od vstupu *I1* na výstup *O* buňky. Po definici následují dva záznamy o zpoždění, oba shodné: *(84.0:105.0:105.0) (84.0:105.0:105.0)*. V tomto případě je první záznam pro případ, kdy výstupní port buňky přechází z log. 0 do log. 1 a druhý pro obrácený případ (více viz [5], kapitola 5.4.1). Jednotkou v hlavičce souboru je pikosekunda, trojice čísel oddělených dvojtečkou specifikuje po řadě minimální, typické a maximální zpoždění. Zpoždění hrany signálu procházející buňkou po cestě $I1 \rightarrow O$ je tedy v rozmezí 84 ps – 105 ps podle operačních podmínek (teplota, napájecí napětí) a procesu výroby, také viz obrázek 3, kde je zpoždění pro názornost vyznačeno.
- Druhým typem záznamu je záznam *PATHPULSE*. Ten specifikuje, jak se buňka bude chovat ke krátkým pulzům; zde je definováno, že buňkou neprojdou pulzy kratší, než 50 ps (více viz [5], kapitola 5.4.14).

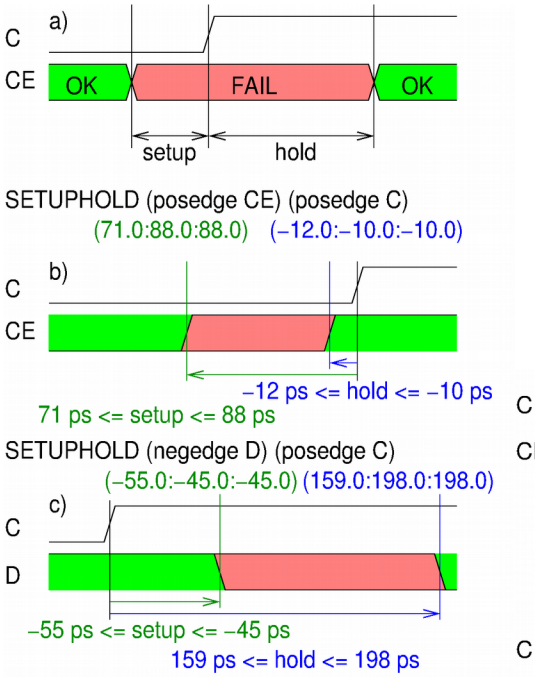
```
module or_block (clk,res,x,y);
  input clk, res, x;
  output y;

  wire clk, wire clk_IBUF;
  wire clk_IBUF_BUFG;
  wire res, res_IBUF;
  wire [4:0]x, [4:0]x_IBUF;
  wire y, y_OBUF, y_d;

initial begin
 $sdf_annotate("or_block_time_impl.sdf",,,,"tool_control");
end
  BUFG clk_IBUF_BUFG_inst
       (.I(clk_IBUF),
        .O(clk_IBUF_BUFG));
  IBUF clk_IBUF_inst
       (.I(clk),
        .O(clk_IBUF));
  IBUF res_IBUF_inst
       (.I(res),
        .O(res_IBUF));
  IBUF \x_IBUF[0]_inst
       (.I(x[0]),
        .O(x_IBUF[0]));
... vynechané podobné instance bufferů ...
  OBUF y_OBUF_inst
       (.I(y_OBUF),
        .O(y));
  LUT5 #(
    .INIT(32'hFFFFFFFE))
    y_i_1
       (.I0(x_IBUF[2]),
        .I1(x_IBUF[0]),
        .I2(x_IBUF[1]),
        .I3(x_IBUF[3]),
        .I4(x_IBUF[4]),
        .O(y_d));
  FDCE #(
    .INIT(1'b0))
    y_reg
       (.C(clk_IBUF_BUFG),
        .CE(1'b1),
        .CLR(res_IBUF),
        .D(y_d),
        .Q(y_OBUF));
endmodule
`ifndef GLBL
`define GLBL
`timescale  1 ps / 1 ps

module glbl ();

    parameter ROC_WIDTH = 100000;
    parameter TOC_WIDTH = 0;

... vynechané deklarace, které nejsou podstatné pro tento článek ...

    initial begin
        GSR_int = 1'b1;
        PRLD_int = 1'b1;
        #(ROC_WIDTH)
        GSR_int = 1'b0;
        PRLD_int = 1'b0;
    end

    initial begin
        GTS_int = 1'b1;
        #(TOC_WIDTH)
        GTS_int = 1'b0;
    end

endmodule
`endif
```

Obrázek 4: Netlist demonstračního obvodu; netlist byl krácen kvůli úspoře místa, vynechané části jsou komentovány.

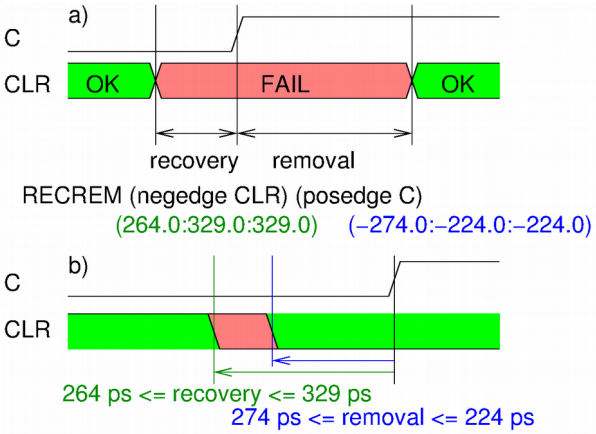
3.2 Registr

Anotace registru je v SDF souboru také zvýrazněná komentářem:

- V záznamu *IOPATH* vidíme oproti předchozí buňce novinku: zápis *IOPATH (posedge CLR) Q* znamená, že definujeme zpoždění od náběžné hrany signálu *CLR* (asynchronní reset) na výstup buňky *Q*. To dává smysl, protože asynchronní reset se na výstupu projeví jen při aktivaci – zde s náběžnou hranou na vstupu *CLR*.
- Záznam *TIMINGCHECK* definuje časové parametry klopného obvodu. Ty jsou pak použity během simulace na hradlové úrovni – model registru kontroluje jejich dodržení. Následuje specifikace mezních podmínek časování pro jednotlivé případy.
- Záznam *SETUPHOLD* definuje současně předstih i přesah na datových vstupech klopného obvodu vůči vzorkovací hraně hodin, viz obrázek 5. Jedním příkladem je záznam *(SETUPHOLD (posedge CE) (posedge C) (71.0:88.0:88.0) (-12.0:-10.0:-10.0))*. Tím je stanoven předstih a přesah, který musí dodržet signál *CE* (konkrétně jeho náběžná hrana) vůči náběžné hraně hodinového signálu *C*. Situace je rozebrána v časovém diagramu na obrázku 5b. První trojice čísel *71.0:88.0:88.0* definuje předstih klopného obvodu opět pro všechny tři provozní rohy; kladný předstih je před hranou hodin. Druhá trojice čísel *-12.0:-10.0:-10.0* definuje přesah po hraně hodin; ten je zde záporný – proto interval přesahu končí už *před* vzorkovací hranou hodin. Podobně je na obrázku 5c popsán případ pro *SETUPHOLD (negedge D) (posedge C)* – definice kontroly předstihu a přesahu od sestupné hrany na datovém signálu *D* vůči vzorkovací náběžné hraně hodin *C*. V tomto případě je negativní čas pro předstih a pozitivní pro přesah, situace je tedy obrácená. Pro lepší pochopení poznamenejme, že obecná situace v obrázku 5a je znázorněna pro případ, kdy je jak předstih, tak přesah klopného obvodu pozitivní.
- Záznam *RECREM* definuje parametry zotavení po resetu klopného obvodu, viz obrázek 6. V SDF souboru lze nalézt konkrétní zápis *RECREM (negedge CLR) (posedge C)* – ten definuje hodnotu zotavení po resetu, kterou musí dodržet uvolnění asynchronního resetu sestupnou hranou signálu *CLR* vůči náběžné hraně hodinového signálu *C*. O praktických dopadech zotavení po resetu (*reset recovery* a *reset removal timing parameters*) se lze více dočíst například v [6].
- Záznam *PERIOD (posedge C)* definuje minimální periodu hodinového signálu měřenou od jeho náběžné hrany (*posedge C*). Tři hodnoty definují opět limit pro všechny technologické rohy, zde tedy mohou mít hodiny minimální periodu 870 ps pro *min corner* a 1000 ps pro *max corner* a *typical corner*.
- Podobně, konstrukce *WIDTH (posedge CLR)* definuje minimální šířku pulzu na signálu *CLR* měřenou od náběžné hrany na *CLR*, kterou je třeba dodržet, aby byl registr bezpečně vynulován. Bude-li na vstupu *CLR* užší pulz, registr může být metastabilní. Interpretace časových parametrů je stejná jako u záznamu *PERIOD*.



Obrázek 5: Interpretace záznamu SETUPHOLD. Zeleně jsou vyznačeny časové intervaly, kdy se může měnit datový signál vstupující do registru, červeně intervaly kdy změna datového signálu způsobí metastabilitu klopného obvodu. Soubor setup_hold.eps.



Obrázek 6: Interpretace záznamu RECREM. Význam barev je stejný jako na obrázku 5. Soubor recrem.eps.

3.3 Zpoždění na spojích

Poslední druh konstrukce popisuje zpoždění na spojích v netlistu. Ten je v SDF souboru specifikován pomocí konstrukce *INTERCONNECT*. Jednoduchý příklad je znázorněný červeně – zápis *INTERCONNECT y_i_1/O y_reg/D* popisuje zpoždění spoje mezi výstupem *O* buňky *y_i_1* a vstupem *D* registru *y_reg*. Zpoždění je popsáno opět dvěma trojicemi hodnot (72.0:90.0:90.0) (72.0:90.0:90.0); v tomto případě je první trojice zpoždění přechodu 0 → 1 na spoji a druhá trojice popisuje zpoždění přechodu 1 → 0 na spoji; více viz [5], kapitola 5.4.1. Poznamenejme zde, že zpoždění na spojích mezi buňkami je běžně modelováno jako zpoždění vložené na vstupu cílové buňky, v našem případě tedy jako „virtuální“ zpožďovací člen vložený do vstupu *D* v registru *y_reg*.

4 Výpočet časových parametrů obvodu

V obrázku 3 jsou vyznačena zpoždění některých prvků a spojů mezi nimi; v SDF souboru jsou příslušné řádky označeny stejnou barvou pro snazší orientaci. Jako malé cvičení můžeme nyní spočítat několik časových parametrů celého návrhu:

- jaký je celkový čas za který se dostane náběžná hrana na vstupu *x(0)* na vstup *D* klopného obvodu *y_reg*? Pohledem do anotovaného schématu na obrázku 3 můžeme jednoduše spočítat následující:
  nejmenší zpoždění: *737,8 ps + 1035,7 ps + 84 ps + 72 ps = 1929,5 ps*
  největší zpoždění: *868,4 ps + 1242,7 ps + 105 ps + 90 ps = 2306,1 ps*
- jaký je celkový čas za který se po přivedení náběžné hrany na vstup *clk* aktualizuje výstup *y* obvodu? Opět, pohledem do schématu s anotacemi můžeme napsat tyto výpočty:
  nejmenší zpoždění: *739,4 ps + 1604,1 ps + 77 ps + 1404,8 ps + 1278,8 ps + 229 ps + 2184,8 ps = 7517,9 ps*

největší zpoždění: *870 ps + 1692,8 ps + 81 ps + 1533,8 ps + 1412,8 ps + 285 ps +2352,3 ps = 8227,7 ps*

Vidíme, že pokud chceme, aby se výstup stihl aktualizovat z pohledu vnějšího systému ještě ve stejné periodě hodin, můžeme přivést hodiny o maximální frekvenci zhruba 121.5 MHz.

Na základě dalších informací uvedených v SDF souboru a těchto výpočtů si nyní čtenář může jako domácí cvičení spočítat jaký předstih a přesah proti hraně hodin na vstupu *clk* musí dodržet vstup *x(0)* aby na klopném obvodě *y_reg* nemohlo dojít k porušení předstihu a přesahu. Kalkulace zpoždění na spojích v obvodu v praxi ovšem neprovádíme ručně, tyto příklady ale mohou rámcově ozřejmit způsob, jakým pracuje statická časová analýza.

Čtenář si dále nemůže nevšimnout výrazného nepoměru mezi zpožděním na buňkách (například zpoždění napříč buňkou *LUT5 y_i_1* je 84 – 105 ps) a zpožděním na spojích (například z výstupu *Q* registru y_reg na vstup *I* bufferu *y_OBUF_inst* je 1278.8 ps – 1412.8 ps). Dominance zpoždění na spojích je pro moderní technologie výroby číslicových obvodů charakteristická.

5 Závěr

I přes omezený rozsah příspěvku a zjednodušení celé problematiky článek čtenáři přiblížil základní konstrukce používané pro modelování číslicových obvodů na hradlové úrovni. Jako jednoduché cvičení doporučujeme čtenáři otevřít si některý ze svých projektů a prohlédnout si SDF soubor i netlist. Další informace lze získat studiem odkazované literatury na konci článku, případně pěkného návodu v dokumentu [1].

6 Použitá literatura

[1] XILINX. Vivado Design Suite User Guide Logic Simulation, UG900, verze 2014.2 (4. července 2014)

[2] ŠŤASTNÝ, Jakub. Simulace číslicových obvodů: úvod. *DPS Elektronika od A do Z*, leden/únor 2015, s. 23-27

[3] BERGERON, Janick. Writing Testbenches: Functional Verification of HDL Models. Springer, 2nd edition, 2003.

[4] ŠŤASTNÝ, Jakub. Simulace číslicových obvodů: triky i úskalí simulace. *DPS Elektronika od A do Z*, březen/duben 2015, s 20-23

[5] IEEE 1497-2001 *IEEE Standard for Standard Delay Format (SDF) for the Electronic Design Process,* verze 14.12.2001 [vid. 15. března 2015] Dostupné z http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=972829

[6] CUMMINGS, Clifford. Asynchronous & Synchronous ResetDesign Techniques – Part Deux. In: SNUG 2003 Boston. [vid. 29. října 2015] Dostupné z http://www.sunburst-design.com/papers/CummingsSNUG2003Boston_Resets.pdf

```
//Hlavička souboru
(DELAYFILE
(SDFVERSION "3.0" )
(DESIGN "or_block")
(DATE "Mon Sep 28 17:45:11 2015")
(VENDOR "XILINX")
(PROGRAM "Vivado")
(VERSION "2015.2")
(DIVIDER /)
(TIMESCALE 1ps)
//Časové parametry buněk
(CELL
  (CELLTYPE "IBUF")
  (INSTANCE clk_IBUF_inst)
  (DELAY
    (PATHPULSE (50.0))
    (ABSOLUTE
      (IOPATH I O (739.4:870.0:870.0) (739.4:870.0:870.0))
    )
  )
)
(CELL
  (CELLTYPE "BUFG")
  (INSTANCE clk_IBUF_BUFG_inst)
  (DELAY
    (PATHPULSE (50.0))
    (ABSOLUTE
      (IOPATH I O (77.0:81.0:81.0) (77.0:81.0:81.0))
    )
  )
)
(CELL
  (CELLTYPE "IBUF")
  (INSTANCE x_IBUF\[0\]_inst)
  (DELAY
    (PATHPULSE (50.0))
    (ABSOLUTE
      (IOPATH I O (737.8:868.4:868.4) (737.8:868.4:868.4))
    )
  )
)

... vynechané anotace instancí - téměř stejné jako IBUF výše ...
//Popis zpoždění kombinační buňky
(CELL
  (CELLTYPE "LUT5")
  (INSTANCE y_i_1)
  (DELAY
    (PATHPULSE (50.0))
    (ABSOLUTE
      (IOPATH I4 O (84.0:105.0:105.0) (84.0:105.0:105.0))
      (IOPATH I3 O (84.0:105.0:105.0) (84.0:105.0:105.0))
      (IOPATH I2 O (84.0:105.0:105.0) (84.0:105.0:105.0))
      (IOPATH I1 O (84.0:105.0:105.0) (84.0:105.0:105.0))
      (IOPATH I0 O (84.0:105.0:105.0) (84.0:105.0:105.0))
    )
  )
)
//Popis zpoždění registru
(CELL
  (CELLTYPE "FDCE")
  (INSTANCE y_reg)
  (DELAY
    (ABSOLUTE
      (IOPATH (posedge CLR) Q (525.0:654.0:654.0))
      (IOPATH C Q (229.0:285.0:285.0) (229.0:285.0:285.0))
    )
  )
    (TIMINGCHECK
      (SETUPHOLD (posedge CE) (posedge C) (71.0:88.0:88.0) (-12.0:-10.0:-10.0))
      (SETUPHOLD (negedge CE) (posedge C) (71.0:88.0:88.0) (-12.0:-10.0:-10.0))
      (RECREM (negedge CLR) (posedge C) (264.0:329.0:329.0) (-274.0:-224.0:-224.0))
      (SETUPHOLD (posedge D) (posedge C) (-55.0:-45.0:-45.0) (159.0:198.0:198.0))
      (SETUPHOLD (negedge D) (posedge C) (-55.0:-45.0:-45.0) (159.0:198.0:198.0))
      (PERIOD (posedge C) (870.0:1000.0:1000.0))
      (WIDTH (posedge C) (435.0:500.0:500.0))
      (WIDTH (posedge C) (435.0:500.0:500.0))
      (WIDTH (posedge CLR) (435.0:500.0:500.0))
    )
)
(CELL
  (CELLTYPE "OBUF")
  (INSTANCE y_OBUF_inst)
  (DELAY
    (PATHPULSE (50.0))
    (ABSOLUTE
      (IOPATH I O (2184.8:2352.3:2352.3) (2184.8:2352.3:2352.3))
    )
  )
)
//Anotace zpoždění na spojích
(CELL
    (CELLTYPE "or_block")
    (INSTANCE )
    (DELAY
      (ABSOLUTE
      (INTERCONNECT clk_IBUF_BUFG_inst/O y_reg/C (1404.8:1533.8:1533.8) (1404.8:1533.8:1533.8))

      (INTERCONNECT clk_IBUF_inst/O clk_IBUF_BUFG_inst/I (1604.1:1692.8:1692.8)
                                                         (1604.1:1692.8:1692.8))
      (INTERCONNECT res_IBUF_inst/O y_reg/CLR (785.9:929.9:929.9) (785.9:929.9:929.9))
      (INTERCONNECT x_IBUF\[0\]_inst/O y_i_1/I1 (1035.7:1242.7:1242.7) (1035.7:1242.7:1242.7))
... vynechané řádky pro další bity x_IBUF, stejné jako výše ...
      (INTERCONNECT y_i_1/O y_reg/D (72.0:90.0:90.0) (72.0:90.0:90.0))
      (INTERCONNECT y_reg/Q y_OBUF_inst/I (1278.8:1412.8:1412.8) (1278.8:1412.8:1412.8))
      )
    )
)
)
```

Obrázek 7: Model časování příkladu v souboru SDF, soubor byl krácen kvůli úspoře místa, vynechané řádky jsou komentovány.