

Programování pro Hardware

Rychlokurs Verilogu a FPGA

Stručný obsah

- Úvod, využití, motivace
- Základy programování
 - Typické workflow
 - Střípky z Verilog HDL
 - Simulace
- Problémy a omezení
- „Praktické“ aplikace

Úvod – FPGA, CPLD

- Programovatelné hradlové pole
 - Logické funkce, paměť, propojení, ...
- Field-Programmable Gate Array (FPGA)
 - Logické jednotky (Logic Elements, Logic Blocks)
 - Propojení mezi jednotkami
- Complex programmable logic device (CPLD)
 - Odlišná implementace (Macro Cell)
 - Jednodušší, levnější

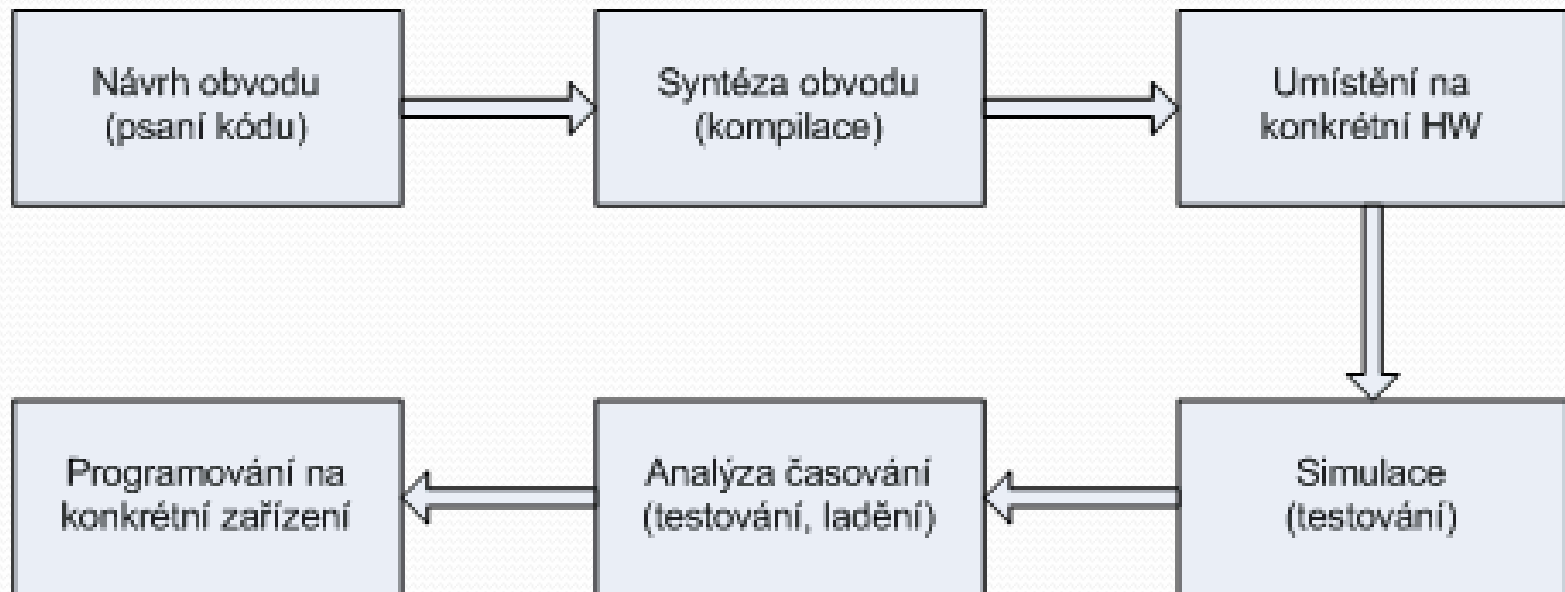
Úvod - Motivace

- Stále jde o obvody
 - Rychlé, paralelní, zpracování
- Rekonfigurovatelnost
 - Různé úlohy = Jedno zařízení, různé konfigurace
 - Snadná aktualizace, rychlejší vývoj
- Standardní nástroje
 - Vývojové prostředky
 - Konfigurační nástroje

Úvod – Typické použití

- Oblasti použití
 - Zpracování signálů (obrazu, zvuku, ...)
 - Řízení
 - Kryptografie
- Výhody
 - Umožňuje paralelní zpracování → vhodné pro dobře paralelizovatelné úlohy
 - Velmi vhodné pro proudová data

Programování



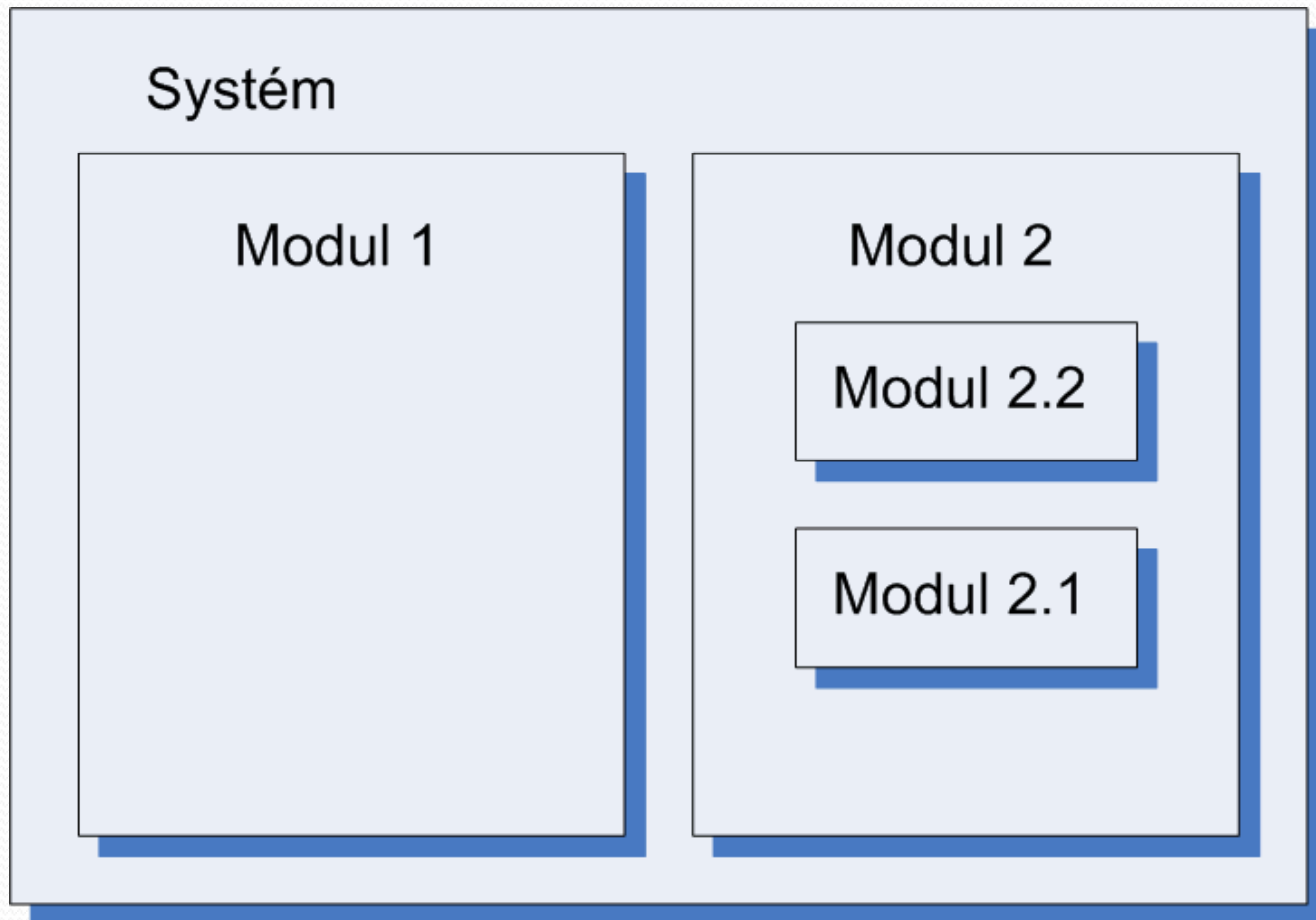
Programování – Návrh obvodu

- Speciální jazyky (Hardware Definition Language)
 - ISP (1972) – první známý HDL
 - VHDL
 - UDLI
 - **Verilog**
- Popis nezávislý na konkrétním HW
- Speciální jazykové konstrukce
- Rozšíření pro testování a ladění
 - Design i testy ve stejném jazyce

Programování - Verilog

- Historie
 - Vytvořen 1983/1984 (Phil Moorby, Prabhu Goel)
 - Verilog-95 – IEEE standard
 - Verilog 2001 – IEEE standard, rozšíření Verilog-95 o nové konstrukce
 - Verilog 2005 – drobné opravy a vylepšení
- Podpora v nástrojích
 - Lepší se neptat...

Programování - Moduly



Programování - Moduly

- Modul = základní stavební jednotka
 - Kód + Data + Zapojení
- Části modulu
 - Vstupy, výstupy
 - Vnitřní propojení
 - Registry („proměnné“)
 - Instance podmodulů
 - Kód
 - Inicializační kód
 - Obsluha událostí

Programování – příklad

- (Ne)praktický příklad:

```
module Counter(KEY, LED);  
input KEY;  
output[7:0] LED  
reg[7:0] CMD;  
  
initial  
    LED <= 0;  
  
always @ (negedge KEY)  
    Cnt = Cnt + 1;  
  
assign LED = ~Cnt;  
endmodule
```

Programování - příklad

- (Ne)praktický příklad v praxi

Programování – datové typy

- Register / Net
- Čísla, samá čísla
 - Unsigned integer (lze i signed), lze zadat velikost v bitech
 - Reálná čísla
- Pole
 - Vícerozměrná pole, Asociativní pole, Dynamická pole
- Struktury
- Podmoduly
- ...

Programování – bloky kódu

- Reakce na události

always @ (**posedge** K) ...

- Inicializace

initial ...

- Zápis

- **begin / end**
- Obvyklé konstrukce jako v C nebo Pascalu: **if**, **for**, **while**, **case**
 - Pozor, musí být syntetizovatelné

Programování – speciality

- Paralelní blok

- **fork**

- a = 1;

- b = 2;

- join**

- Časování

- #5 a = 1

- „Setrvalé“ přiřazení

- **assign** a = b + c;

- Vyčkávání

- @ (posedge K) ; wait(X)

Programování – co se nevešlo

- Podprogramy
 - Task, Function
- Práce se „soubory“
 - \$readmemh, \$readmemb
- Preprocesor
- Ladící výpisy

Simulace a testování

- Programátorský přístup
 - Simulátor + program pro řízení testu dodávající vstupy a testující výstupy
 - Jak testovaný obvod, tak testující program ve Verilogu
 - Není snadné na přípravu (záludné chyby nemusejí být jen v testovaném modulu)
- „Simulační“ přístup
 - Samostatná aplikace pro simulaci obvodu
 - Vstupem i výstupem jsou vektory hodnot na vstupech/výstupech testovaného modulu

Simulace a testování

- Praktická ukázka

Praktické problémy

- Podpora prvků jazyka ve vývojových prostředích
 - Nebo alespoň v Quartus II
 - Asociativní pole, reálná čísla, paralelní bloky, ...
- Podpora prvků jazyka při syntéze
 - Cykly, soubory, reálná čísla, dynamická pole, ...
 - Některé nepodporované části jazyka lze nahradit speciálními moduly
- Dostupnost nástrojů („nic není zadarmo“)
- Dokumentace

Praktické problémy II

- Podivné a nečekané chování a jeho ladění
 - Příručka není skutečný hardware; simulace také ne
- Low-level jazyk s low-level výrazovými prostředky
 - Pouze základní datové typy a operátory
 - Rozšiřující knihovny funkcí (Altera MegaFunctions)
 - Rozhraní, lepší aritmetika (reálná čísla), Flash loader
- Kapacita hardware
 - Max II – cca 2100 logických jednotek
 - Obvod pro sčítání ve FP – 700+ logických jednotek
- Komunikace s okolním světem

Řešení některých problémů

- Reálná čísla
 - Zařízení s větší kapacitou (až 200 000 log. jednotek)
 - Přejít do fixed-point aritmetiky
- Složitější matematické funkce
 - Součástí cizích knihoven, tvorba vlastních funkcí

Řešení některých problémů

- Komunikace s okolním světem
 - Hardwarové porty + vlastní konektor do PC
 - Nutné implementovat vybraný komunikační protokol
 - JTAG
 - Komunikační protokol, určeno pro vnější testování
 - Složitější programování z pohledu PC aplikace
 - Přenosová rychlost(?)
 - Iniciativa na straně PC – nelze zahájit komunikaci z FPGA
 - Možné použití
 - Programování, nahrávání/stahování parametrů obvodu (např. váhy neuronové sítě)

Komunikace s okolním světem

- Praktická ukázka

Další praktické ukázky

- Vyhledávací automat (Aho-Corasickové)

...

```
case (State)
```

```
    0:      State = 2;
```

```
    1:      State = 3;
```

```
    2:      State = 2;
```

```
    3:      State = 5;
```

```
    4:      State = 6;
```

```
    5:      State = 2;
```

```
    6:      begin
```

```
                State = 2;
```

```
                FOUND = 0;
```

```
            end
```

```
endcase
```

...

Další praktické ukázky

- Vyhledávací automat
 - Ošklivý a nepřehledný kód
 - Není nutné vytvářet ručně, tvorbu pro zadanou množinu slov lze automatizovat (program generující program)

Další praktické ukázky

- „Průtoková“ Bayesovská klasifikace
 - Vyhledávací automat dává výskyty slo v → naivní Bayesovská klasifikace
 - Průběžné odhady score pro porovnání $P(H|W_1, W_2, \dots)$
 - Výpočet v logaritmech pravděpodobnosti
 - Celočíselná aritmetika

Další možnosti

- Perceptronová síť, skoková přechodová funkce
 - Realizovatelné
 - Celočíselná reprezentace
 - Neuron jako 1 modul
- Sigmoidová přenosová funkce
 - Obtížně realizovatelné (floating-point výpočty)
 - Fixed-point aritmetika (je dostatečně přesná?)
 - Učení

Odkazy, doporučené čtení

- Výrobci (zařízení i nástroje)
 - Altera – <http://www.altera.com>
 - Xilinx – <http://www.xilinx.com>
- Open-source implementace
 - Icarus Verilog – <http://www.icarus.com/eda/verilog/>
- Užitečné weby, další informace
 - <http://www.fpga4fun.com>
 - <http://www.verilog.com>
 - <http://www.verilog.net>

Závěr & Diskuse

- Zde je prostor pro vaše dotazy