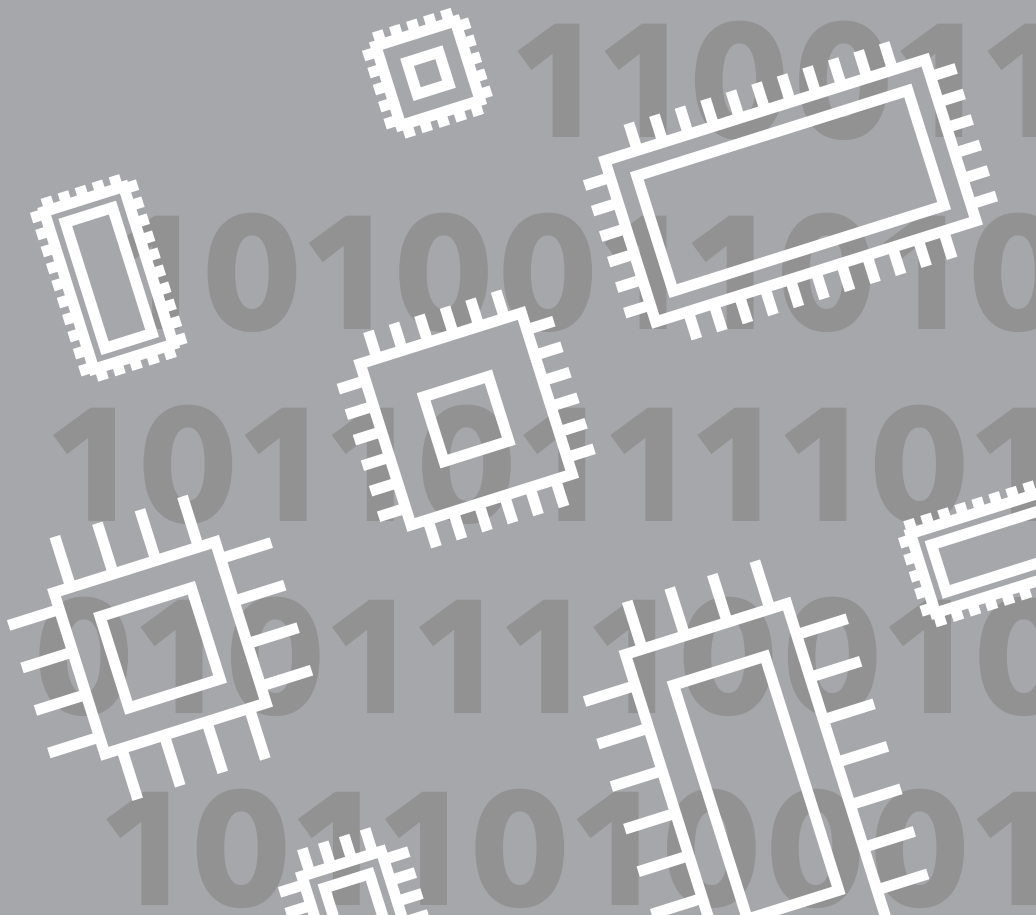
Martin Malý

Data, čipy, procesory

Vlastní integrované obvody na koleni



Edice CZ.NIC

DATA, ČIPY, PROCESORY
Vlastní integrované obvody na koleni

Martin Malý

Vydavatel:
CZ.NIC, z. s. p. o.
Milešovská 5, 130 00 Praha 3
Edice CZ.NIC
www.nic.cz

1. vydání, Praha 2020
Kniha vyšla jako 25. publikace v Edici CZ.NIC.
ISBN 978-80-88168-56-0

CZ.NIC | SPRÁVCE DOMÉNY CZ

ISBN 978-80-88168-56-0

— Martin Malý

Data, čipy, procesory
Vlastní integrované obvody na koleni

— Edice CZ.NIC

Poděkování

Poděkování

Děkuju všem lidem, kteří mi při psaní fandili. Všem čtenářům předchozích knih, co mi psali, že s nimi je začala elektronika zase bavit. Všem lidem, kteří mi psali, že se těší na pokračování. Všem těm, kteří četli rukopis a přispěli radami a názory. Všem těm, kteří mi dali ve chvílích pochybností podporou najevo, že to není úplně zbytečná práce, že to někoho zajímá a že nepíšu do zdi.

Speciální díky patří nadšencům, kteří podpořili vznik knihy na Patreonu i jinde: Josef Adamčík, Stanislav Jurný, Michal Kočer, Martin Ludik, Karel Nenička, Filip Novák, Dan Tománek, Radomír Vávra, Kamil Zmeškal – a určitě jsem na někoho zapomněl, za což se omlouvám.

Velké poděkování patří vydavateli a všem lidem z Edice CZ.NIC, kteří z rukopisu udělali knihu. Už potřetí!

A v neposlední řadě patří poděkování mé tolerantní partnerce Míše.

Díky, díky, díky!

— Poděkování

Předmluva vydavatele

Předmluva vydavatele

Vážení čtenáři,

dostává se vám do ruky třetí kniha od Martina Malého, tentokrát na téma programovatelná hradlová pole (FPGA, anglicky Field Programmable Gate Array). Na předchozí dvě navazuje jen velmi volně, ovšem hodí se mít pročtené „Hradla, volty, jednočipy". Předchozí knihy vám pomohou zasadit nové informace do správného kontextu, ale jelikož je téma dost odlišné, neztratíte se, ani pokud je neznáte.

Vždycky jsem obdivoval lidi, kteří hardware rozumí a ví, co se uvnitř děje a jak. Já sám jsem se nikdy nedostal o moc dál, než jsou základy ze středoškolské fyziky. Vývoj software mě pohltil daleko víc a vydal jsem se touto cestou. I proto je pro mne téma FPGA velmi lákavé. Je to možnost, jak vyžít své zkušenosti a schopnosti úplně jiným způsobem – ponořit se do návrhu složitého hardware, i když své hardware znalé kolegy běžně děsím tím, co si doma bastlím a jak.

Než knihu začnete číst, měl bych jedno varování pramenící z mé osobní zkušenosti s touto knihou. Není snadné ji dočíst. Nebudete ani ve třetině, když vás přepadne neodvratné nutkání pořídit si nějaký ten dev kit a příklady si zkoušet naživo. Přeci jen číst si o tom a opravdu to dělat, jsou dvě různé věci. Doporučuji však pokusit se nutkání odolat. Vím, není to snadné. Ale i když už začátek zní velmi lákavě, vyplatí se počkat si na pozdější kapitoly, kdy se třeba dozvíte, jak generovat grafický výstup, připojit SD kartu nebo kde najít již připravené moduly. Sčítání bitů zní zajímavě, ale teprv když zařízení dělá něco samo o sobě (bez debuggeru), tak to má to správné kouzlo. A během čtení se vám budou postupně odkrývat nové a nové možnosti, co s FPGA dělat, a nápady, co s FPGA podniknout, budou jen a jen přibývat. A určitě některé z nich ovlivní i výsledný výběr dev kitu.

Ohledně FPGA najdete na Internetu spoustu informací, ale není úplně snadné rozmyslet si, kde a čím začít. Tato kniha vás však velmi poutavou formou se světem FPGA snadno seznámí, navnadí vás a dá vám užitečné stavební kameny do začátku. Pak už zbývá si jen pořídit dev kit, vše si prakticky vyzkoušet a začít tvořit.

Přeji příjemné čtení a mnoho zajímavých pokusů s FPGA.

Michal Hrušecký, CZ.NIC

Předmluva

Předmluva

Psal se rok 2005 a já se probíral příspěvky v internetové konferenci elektroniků, elektrotechniků a vůbec elektrošotoušů a bastlířů. Většina příspěvků se točila okolo tehdy populárního jednočipu PIC16C84, popřípadě kolem toho, jaké vybavení je pro dílnu dostatečné, a najednou do diskuse vstoupil *opravdový amatér.* Člověk, pro kterého byla mikroelektronika kouzelný svět, který s nadšením objevoval a se kterým se seznamoval.

Na začátku příspěvku se omluvil za to, že se vůbec dovoluje na něco zeptat (ano, to patřívalo k bontonu... *omluvte mě, vy moudří, že mám hloupý začátečnický dotaz),* a pak položil skvělou otázku: Jestli by mu někdo nemohl poradit, jak by si mohl splnit svůj sen, totiž **navrhnout si vlastní mikroprocesor.**

A já se opět, jak mávnutím kouzelného proutku, ocitl v roce 1984 v okresní knihovně, v oddělení techniky, a držel jsem v ruce knihu „Polovodičové paměti a jejich použití". Hned v úvodu autor popisoval, jak se vytvářejí tranzistory MOS a jak pracují. Bylo to tak jasné a pochopitelné, že jsem věřil – opravdu jsem tomu věřil! – že je možné si takové tranzistory dělat doma na koleni. Však křemík je všude, ty donory a akceptory by se taky někde sehnat daly, to přeci musí jít! A věřil jsem tomu, že jednou, jednoho dne, si ve sklepě v Petriho misce stvořím vlastní polovodič! Inu, byl rok 1984, bylo mi 11 let a připadalo mi snazší si udělat vlastní integrované obvody, než doufat, že si je jednou koupím v Elektře na rohu.

Nota bene když jsem v ruce držel knihu, kde to všechno bylo popsané. Jak se z tranzistorů poskládají hradla, z hradel klopné obvody, multiplexory, matice paměťových buněk... prostě všechno. I vlastní mikroprocesor bych zvládnul, určitě!

Úplně jsem cítil atmosféru té knihovny a svou dětskou radost, když jsem si do sešitu obkresloval tranzistory, křemíkové struktury a obvody a navrhoval jsem si vlastní komponenty. Tužkou, na papíře... Moc se mi líbilo, že si někdo takový sen udržel i po dvaceti letech a statečně se zeptal: „chtěl bych si navrhnout vlastní mikroprocesor, jak na to?"

Nepřekvapivě dostal *neskutečnej kartáč* od osazenstva konference, plus mínus ve stylu „my tu řešíme reálné problémy a na takovéhle bláznivé fantazie tady nejsme zvědaví, kšá!" *Vlastní procesor? To nejde, na to nikdy mít nebudeš, to nikdy nezvládneš, to ti žádná fabrika nikdy nevyrobí, to nikdy nikdo nebude používat, tak s tím neotravuj.*

A po zhruba dvaceti odpovědích v tomto stylu přišel jeden z členů té konference a tazatele nezadupal, ale naopak ho povzbudil. Ať si z těch řečí kolem nic nedělá, ať to klidně zkusí, protože se u toho naučí o elektronice mnohem víc než všichni ostatní účastníci té diskuse dohromady, a ať se nebojí, že to nevyjde, protože i tak získá spoustu neocenitelných znalostí. No a nakonec dodal, že nejjednodušší bude podívat se na obvody CPLD nebo FPGA a naučit se nějaký HDL,

jazyk, kterým je možné ty obvody programovat.

Jak to s tazatelem dopadlo a jestli si vlastní procesor někdy navrhl, to netuším. Já si jen pamatuju, že jsem si kamsi v hlavě udělal poznámku: FPGA, HDL, *zajímavé.* A pak to na dlouhé roky vytěsnil, protože to bylo ve škatulce „drahé, nedostupné, není na hraní" - a já chci elektroniku hlavně na hraní.

Není to tak dlouho, tak sedm let, kdy se najednou objevily levné kity s FPGA, dostupné i pro nás, bastlíře, a začaly se objevovat první nesmělé pokusy a první konstrukce. A tehdy jsem si i já koupil svůj první kit, asi za tisícovku, a po několika týdnech experimentů jsem stvořil funkční repliku mikropočítače PMI-80.

Dnes jsou FPGA ještě dostupnější, a ti z vás, co už vymačkali maximum ze svých Arduin, BluePill a jiných Raspberry, se možná začínají poohlížet právě po těchto obvodech. Nedivím se – pověst, která je předchází, je zajímavá. *Obvod, který může být čímkoli, co zvládnete nadefinovat,* no není to sen?

Možná se trochu bojíte, možná máte v hlavě nějaký vnitřní majáček, který vám říká, že *to je složité, nezvládnete to, na nic to nebude...* Kašlete na majáček! Vážně! Pusťte se do toho.

Kniha, kterou právě držíte v ruce a kterou si za chvíli koupíte, je přesně to, co potřebujete, aby se z vás, z člověka, *co by to rád zkusil,* stal člověk, který si to zkusil – a možná ho to chytlo a bude pokračovat! Ukážeme si, co jsou vlastně FPGA, jaké možnosti coby amatér máte, a pak se naučíme jeden z univerzálních jazyků pro popis elektroniky, totiž VHLD. Zabrousíme i do Verilogu, naučíme se obvody navrhovat, testovat, simulovat, naučíme se, jak se ve VHDL zapisují základní konstrukční prvky, jak se skládají dohromady, jak se vytvářejí obvody pomocí popisu jejich chování, připravíme si sadu užitečných elementů pro vlastní pokusy, a pak si ukážeme nejen to, jak ve FPGA vytvoříte celý počítač, ale i to, jak si uděláte vlastní mikroprocesor.

Kniha není ani technická příručka, ani učebnice. Na to je příliš hovorová a příliš populární. Nenahradí vám vysokoškolská skripta a po jejím přečtení asi nebudete připraveni nastoupit do vývojové laboratoře a živit se jako konstruktér s FPGA obvody. To ani není její cíl. Její cíl je jiný: ukázat vám zajímavý svět obvodů FPGA a uživatelsky definované elektroniky, což je dnešní „hi-tech" oblast, zbavit vás ostychu a strachu, že *tomu nebudete rozumět,* a povzbudit ve vás chuť zkoušet a vymýšlet nové věci.

I kdyby to mělo být něco neužitečného a nepraktického.

Pojďme na to!

Důležité post scriptum: K této knize je dostupný web ***https://datacipy.cz/****, kde kromě tipů na další čtení a užitečných odkazů najdete i zdrojové kódy všech příkladů z knihy, včetně testů, a také kódy, které byly příliš dlouhé a do knihy se nevešly.*

Obsah

1 FPGA? Co, prosím?

1 FPGA? Co, prosím?

Trocha historie nikoho nezabije, na rozdíl od sestavování složitých kombinačních obvodů...

No dobře, přeháním, ani sestavování kombinačních obvodů není smrtící, ale představte si takový dekodér, jaký jsme používali v knize Porty, bajty, osmibity.

(Pokud jste tuto knihu nečetli, tak vám prozradím, že jsme většinou dekódovali 16 bitů adresové sběrnice tak, aby byl adresován jeden ze dvou paměťových obvodů, popřípadě jsme dekódovali tři bity adresy na 8 různých signálů pro výběr obvodů.)

Teď si jej představte o něco složitější. Představte si jemnější škálování adresního prostoru, třeba po 1 kB blocích, to máte šest adresních vstupů, a představte si složitější paměťovou mapu, kde třeba prvních 8 kB je RAM, pak 8 kB ROM, pak 16 kB zase RAM, 1 kB prostoru pro periferie, ten se patnáctkrát zrcadlí, posledních 16 kB je zase ROM, ale stránkovaná...

Samozřejmě že lze nakreslit pravdivostní tabulku (spíš tabuli), Karnaughovu mapu, sepsat logické výrazy a pokoušet se to převést do NANDů, NORů, třívstupových, čtyřvstupových, osmivstupových hradel a různých AND-OR-INVERTů. Nakonec skončíte s něčím, co vyžaduje zabírá pět integrovaných obvodů, některé ale použité třeba jen z poloviny. Funguje to, to ano, ale topí to a zabírá to spoustu místa.

Naštěstí moderní elektronika udělala od sedmdesátých let obrovský skok kupředu, a během uplynulé dekády se i ty nejmodernější obvody staly dostupné běžným smrtelníkům – tedy lidem, jako jsme my. Dnes máme možnost navrhnout si celé systémy, které vyžadovaly desítky či stovky integrovaných obvodů, pomocí jazyků pro formalizovaný popis logických obvodů, v počítači vše nasimulovat, a nakonec nahrát do „prázdného obvodu“, který se tak promění v cokoli, co chceme.

A právě o tom je celá kniha, kterou právě držíte v ruce!

1.1 Programovatelné obvody

PROM

V letech dávno minulých se podobné problémy, pokud jste na to měli příslušné vybavení, řešily pomocí pamětí PROM. Abychom si rozuměli: Opravdu platí, že PROM jsou programovatelné paměti ROM, a že by v nich měly být nějaké hodnoty konstant a tak, ale když to vezmete kolem a kolem, tak takový kombinační logický výraz můžeme zapsat do tabulky, spočítat jeho hodnoty pro všechny možné vstupní kombinace, a pak výsledky naprogramovat do paměti PROM. Jednou zapsaná data zůstanou zapsaná navždy, tak co by ne?

Oblíbená paměť PROM byla typu 74188 / 74288. Má organizaci 32x8, tedy 32 slov po 8 bitech. Jinými slovy má pět adresních vstupů a osm datových výstupů, takže s ní hravě pokryjete případy kombinačních obvodů, které mají do pěti vstupů a do osmi výstupů.

Druhá oblíbená paměť byla 74287 s organizací 256x4. Tedy osm vstupů, čtyři výstupy.

Když jste se podívali v osmdesátých letech do Amatérského radia na číslicové konstrukce, byla taková paměť PROM často i na místech, kde by si autor vystačil s dvěma pouzdry. Asi bylo leckdy jednodušší použít paměť PROM. Mám ale takové podezření, že to hodně záleželo na tom, zda dotyčný návrhář měl přístup k těmto obvodům a k programátoru, nebo naopak zda jeho šuplík oplýval spíš obvody TTL SSI a MSI...

A tak se obvody 188 a 287 objevovaly v rolích dekodérů a složitých kombinačních obvodů mezi procesorem a jeho periferiemi (vžil se název „glue logic“), až do doby, než někoho napadlo, že by to šlo jinak.

Zákaznické obvody

Někteří výrobci čipů nabízeli „prázdné logické obvody“, ovšem s tím, že jejich konfiguraci si zadal zákazník. Takový obvod nabízela třeba i Tesla, ale nejznámější v našich končinách bude výrobce Ferranti a jeho zákaznický obvod ULA, použitý v ZX Spectru. Ve skutečnosti šlo o předchůdce pozdějších obvodů CPLD, kde se konfigurace neukládala do paměti, ale byla z výroby natvrdo „vypálena“ do křemíku. Samozřejmě pro kusovou výrobu byly takové obvody nedostupné, ale pokud jste jich chtěli odebrat tisícové série, měli jste možnost.

PLA

Programmable Logic Array, ve zkratce PLA a česky programovatelné logické pole, je součástka, která přišla na trh za tím účelem, který jsme si právě popsali: vytvořit složitý kombinační obvod v jednom pouzdru. Na rozdíl od PROM, kde se stylem „brute force“ spočítaly hodnoty pro všechny možné kombinace a ty se zapsaly do paměti, u PLA byl návrh bližší tomu obvodovému.

PLA si můžeme představit jako sestavu „pole AND“, „pole OR“ a „pole INVERT“. Každý z těchto bloků je zapojený jako matice N sloupců (vstupy) a M řádků (logická hradla). Podle toho, které propojky naprogramujeme (podobně jako u PROM), takovou funkci na výstupu dostaneme.

Podobnou funkci měly obvody PAL (Programmable Array Logic). Pomocí propojek („fuses“) se při programování určí, které vstupní signály mají vést do jakého bloku AND_OR_INVERT. Obvody PAL se postupně vyvinuly, podobně jako paměti, nejprve do podoby mazatelné UV světlem (PALC) a posléze do elektricky přeprogramovatelných obvodů (PALCE).

Výrobci začali do obvodů přidávat i složitější celky. Například možnost mít u výstupů registr nebo signál na výstupu dále zpracovávat.

Obvody PAL se programovaly podobně jako PROM. Technicky vlastně o PROM / EPROM šlo. Aby nebylo nutné ručně počítat, které propojky se mají nastavit a které nechat, existovaly nástroje jako ABEL, CUPL nebo PALASM, které dokázaly zpracovat logické výrazy zapsané v nějaké formalizované podobě a z nich připravit výstup, vhodný k programování obvodů (nejčastěji ve formátu JEDEC).

Společnost Lattice představila v roce 1983 další vylepšení obvodů PAL s názvem GAL – Generic Array Logic. Tyto obvody byly vývodově kompatibilní s obvody PAL, ale šlo je jednodušeji přeprogramovat, některé z nich i v hotovém zařízení („in place" nebo „in circuit").

CPLD

Obvody PAL a GAL dokázaly nahradit několik obvodů SSI, MSI. Jejich nástupci, obvody CPLD (Complex Programmable Logic Devices), nahradily několik tisíc hradel. Novější generace až stovky tisíc hradel.

Obvody CPLD mají s předchozími generacemi programovatelných obvodů společný princip zaznamenávání konfigurace do interní paměti (EE)PROM, a mnohé mají napevno přiřazené určité vnitřní bloky ke konkrétním pinům.

Hlavní rozdíl je ale ten, že CPLD obsahují řádově víc vnitřních bloků a mají mnohem komplexnější možnosti vnitřního propojení těchto bloků.

Vnitřní bloky jsou rovněž mnohem bohatší než u PAL/GAL. Buňka (macrocell) se typicky skládá z klopného obvodu / registru (představme si ho jako klopný obvod D s nastavením a nulováním, jako je v obvodu 7474), konfigurovatelné logické sítě na jeho vstupech a konfigurovatelných multiplexorech na výstupech.

Například u oblíbených CPLD řady XC9500 od společnosti Xilinx udává poslední dvojice či trojice číslic označení počet těchto buněk. Typ s označením XC9572 jich má 72. V této rodině máte na výběr mezi 36, 72, 108, 144, 216 a 288 makrobuňkami. Největší zástupce této řady, XC95288, nabízí zároveň 6400 hradel k použití.

Řada XC9500 používá pro konfiguraci vnitřní paměť FLASH s udávanou výdrží 10.000 cyklů mazání / zápis. Xilinx už tyto obvody nevyrábí, přesto jsou k sehnání a pro amatérské konstrukce jsou stále vhodné, protože na rozdíl od mnoha pozdějších obvodů dokáží pracovat s pětivoltovou logikou.

Obvody CPLD už stěží kdokoli naprogramuje ručně. K vývoji se používají jazyky, řazené do rodiny jazyků HDL (Hardware Definition Language), typicky VHDL nebo Verilog. V těchto jazycích (později se důkladně seznámíme s VHDL) popisujete hardware pomocí výrazů, které definují buď propojení menších celků, nebo jejich chování. Vývojářské nástroje převedou takto zapsané výrazy do velkého souboru dat pro konkrétní obvod, a pomocí programátoru (většinou typu JTAG) se data zapíšou do obvodu CPLD.

FPGA

Dostali jsme se k těm nejvýkonnějším programovatelným obvodům. Dokáží nahradit desítky tisíc až desítky milionů logických hradel a nejnovější obvody tohoto typu jsou svou složitostí a strukturou srovnatelné se současnými mikroprocesory.

Obvody FPGA (Field-Programmable Gate Array) rozšiřují koncept CPLD a posouvají jej opět o řád dále. Kromě makrobuněk a kombinační logiky, která ve FPGA bývá řešena pomocí tabulek (LUT, Look-Up Tables), nabízejí tyto obvody další funkční celky, jako jsou PLL pro generování frekvencí, paměti, násobičky, AD a DA převodníky, ...

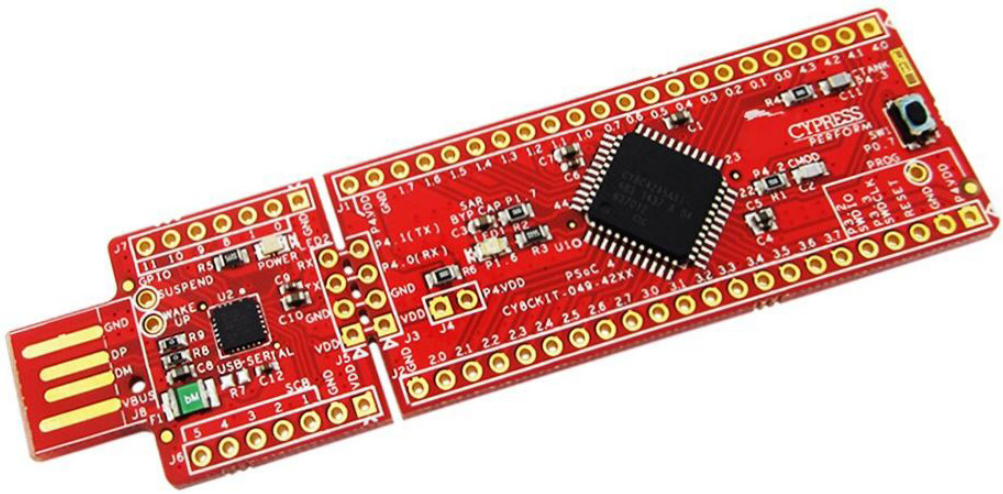
Na rozdíl od obvodů CPLD mívají obvody FPGA svou konfiguraci uloženou nikoli ve vnitřní paměti, ale v paměti vnější, nejčastěji v podobě sériové FLASH. Po startu systému se z této paměti načte konfigurace do FPGA.

SoC

Zajímavý koncept představují obvody SoC (System-on-a-Chip), které v sobě kombinují mikroprocesorové jádro, paměť, standardní periferie a programovatelnou logiku. Vývojáři tak mohou i některé specializované funkce syntetizovat přímo v obvodu, bez nutnosti vytvářet speciální zapojení.

Příkladem mohou být obvody PSoC od Cypress (nyní Infineon). Rozšiřují běžné mikrokontroléry s jádrem ARM o univerzální digitální bloky UDB. Ty můžete naprogramovat (či přesněji konfigurovat) sami, a to buď v editoru komponent, v grafickém editoru stavových strojů, nebo pomocí jazyka Verilog.

Vývojové kity jsou levné a dostupné, například kit CY8CKIT-049-42XX s cenou do deseti dolarů obsahuje čip CY8C4245AXI-483. Ten nabízí procesor ARM Cortex-M8 s maximální frekvencí 48 MHz, 32 kB FLASH, 4 kB RAM, A/D převodník, operační zesilovače, komparátory, PWM a 4 programovatelné logické bloky. U některých konstrukcí může právě programovatelná logika nahradit buď velkou část výkonu, nebo vlastní konstrukci podpůrných obvodů.



K čemu mi je FPGA?

Přesně! *To je vážná průmyslová věc, to není hračka pro bastliče!* Jako bych ty řeči slyšel. Možná je slýcháte taky a bojíte se světem FPGA vůbec zabývat, protože *na to přeci musíte být dírkovaná inženýrka,* abyste tomu rozuměli.

Mám dobrou zprávu: Nemusíte být *dírkovaní,* a přesto si můžete FPGA skvěle užít. Samozřejmě, být vámi, bych se nepouštěl do návrhu průmyslových obvodů, tam je potřeba přeci jen kromě zkušeností i nějaký teoretický základ, ale na takové to domácí hraní – a nebojme se to říct: *bastlení* – je FPGA docela fajn. Představte si to: V jednom takovém obvodu si můžete vytvořit osmibitový procesor, paměť, sériový UART, displej s výstupem na televizi nebo VGA a rozhraní pro klávesnici a SD kartu. Celý počítač. V jednom obvodu. Za pětikilo! No není to sen?

V obvodu FPGA můžete klidně vytvořit hned několik procesorů. Klidně i procesorovou matici. Transputer. Paralelní výpočty tak budou probíhat opravdu paralelně, podobně jako ve vaší grafické kartě. Můžete implementovat klidně neuronovou síť a trénovat ji, na co potřebujete. Díky tomu, že její jednotlivé neurony budou skutečně obvody a ne jen datová struktura v paměti počítače, budou pracovat opravdu paralelně – a opravdu rychle!

Nadšenci do kryptografie a kryptoměn zase mohou FPGA využít jako masivní paralelní počítadla hashů. Rychlost takových počítačů není omezena frekvencí procesoru, ale jen zpožděním při šíření signálu logickými prvky.

FPGA je zkrátka něco jako obří krabice Lega. Ale opravdu obří. Můžete si z ní poskládat téměř cokoli, ale musíte být schopni to poskládat ze základních součástek. A právě v této knize se to společně naučíme.

Ale než se do toho pustíme, tak mi dovolte kus nezbytné teorie, tentokrát formou otázek a odpovědí.

1.2 Jaké FPGA?

Pro amatérské použití se moc nehodí nejnovější a nejvýkonnější obvody. Jejich schopnosti jsou daleko před potřebami amatérské praxe a jejich cena vysoko nad možnostmi amatérské peněženky. Ale i v té spodní, dostupné části spektra nalezneme dostatek obvodů pro konstrukci velmi zajímavých zařízení. Cílem této kapitoly je představit si základní kity, které pořídíte za ceny do tisíce korun, což je rozumná částka, kterou domácí rozpočet unese. Ta nejjednodušší kombinace, viz dál, vyjde cca na 500 Kč.

Kdo vyrábí FPGA?

Největší dva výrobci jsou Xilinx a Altera (Alteru před nedávnem koupil Intel). Kromě nich vyrábí FPGA i další firmy, např. Lattice.

V čem se píše pro FPGA?

FPGA jsou *programovatelná logická pole,* je tedy třeba je naprogramovat. Nejznámější jazyky jsou VHDL a Verilog, ale používají se i jiné (SystemC např.)

Xilinx, nebo Altera / Intel?

Dva největší výrobci FPGA. Jejich řady jsou do určité míry srovnatelné, ale navzájem nekompatibilní. Od Xilinxu pravděpodobně využijete řadu Spartan, konkrétně obvody z řad Spartan 3 a Spartan 6. Od Altery, resp. Intelu, pak řadu Cyclone, konkrétně Cyclone II a Cyclone IV.

Typy, řady a generace FPGA

Značení řad FPGA je na první pohled trošku nepřehledné, ale logiku má. Vezměme si jako příklad výrobce Xilinx a jeho obvody FPGA. Xilinx nabízí FPGA s různými názvy a číselnými označeními. Obecně se dá říct, že číselné označení udává „generaci" – Spartan 3, Spartan 4, ... Spartan 7 jsou postupně výkonnější a výkonnější obvody, vytvořené s novými technologiemi (Spartan 6 například je vytvořen 45nm technologií, sedmá generace pak 28nm, generace UltraScale má technologii 20 nm, UltraScale+ používá 16 nm). V prvních generacích šlo o Spar-

tany, později přibyly další typy, jako Artix, Kintex a Virtex. Spartan jsou levné a nenáročné obvody, Virtex ty nejdražší, nejrychlejší a nejlépe vybavené. Každý obvod má několik různých mutací, podle počtu vývodů nebo logických jednotek uvnitř.

V sedmé generaci přišly obvody Spartan 7, Artix 7, Kintex 7 a Virtex 7, všechny vyrobené technologií 28 nm, a každý v několika verzích.

Spartan 7 se vyrábí jako XC7S6, XC7S15, XC7S25, XC7S50, XC7S75 a XC7S100. XC je obecné označení FPGA od Xilinxu, 7 označuje generaci, písmeno S naznačuje, že jde o Spartan, a poslední číslo udává počet logických buněk (v tisících). Čím větší obvod, tím víc vývodů (S6 jich má 100, S100 až 400), tím víc jednotek PLL, tím víc integrovaných paměťových bloků (20 kB až 480 kB) atd.

Artix stejné generace nabízí obvody XC7A12, A15, ... až XC7A200. Označení je analogické předchozímu, písmeno A udává typ Artix, poslední číslo pak opět hrubý počet logických buněk v tisících (12800 až 215360). Opět platí přímá úměra mezi počtem buněk, počtem vývodů, velikostí paměti (80 kB až 1,4 MB), DSP jednotek atd. Artix ale nabízí i moduly pro PCIe nebo rychlé transceivery.

Kintex 7 se skládá z obvodů XC7K70, K160, K325, ... XC7K480. Logika číslování je opět tatáž, platí i přímá úměra mezi velikostí a dalšími parametry. Dá se říct, že Kintex má vybavení jako Artix, jen má všeho o trochu víc a něco má vylepšené. Moduly DSP má vylepšené, modul PCIe je pro verzi 2 atd.

Virtex 7 si můžete dopřát v nejmenší variantě XC7V330 (326400 logických buněk, 3000 kB RAM) či v té největší XC7V1140 (1,1 milionu logických buněk, 7 MB RAM), některé modely mají i transceivery schopné přenosu rychlostí 28 Gbps, kromě PCIe verze 2 se objevují i PCIe verze 3...

V době psaní knihy je sedmá generace už překonána generacemi UltraScale a UltraScale+, které přinášejí opět větší množství rychlejších modulů. Řádově miliony logických buněk, megabajty RAM, stovky transceiverů, tisíce DSP...

U Intelu převzali řady, které vyráběla Altera, a pokračují v nich. Potkáváme se tedy s obvody z řad MAX, Cyclone, Arria, Stratix a Agilex.

Cyclone, uvedená na trh v roce 2002, má generace II, III, IV, V a 10 (ano, po Cyclone V přišla generace Cyclone 10). Kromě první a druhé generace jsou všechny stále doporučené pro používání.

Téměř stejné generace jsou u typu Stratix: III, IV, V a 10.

MAX je k dispozici jako generace II, V a 10. Generace II a V představují nikoli FPGA, ale CPLD – jednodušší předchůdce FPGA. Za FPGA výrobce označuje až řadu MAX 10.

„Desátá" generace u Intelu / Altery odpovídá zhruba „sedmé" generaci u Xilinxu. Najdete různé variace na MAX 10, Cyclone 10, Arria 10 a Stratix 10. MAX jsou jednočipová FPGA s integrovanou konfigurační pamětí a spolu s Cyclone tvoří „low end". Obvody Arria a Stratix jsou už z kategorie „System on a Chip" (SoC), protože obsahují výkonné procesory ARM Cortex. Stratix je spíše zaměřený na výpočetní výkon, zatímco Arria na komunikaci.

Co je vhodné pro začátečníka?

Platí, že vyšší řada nabízí větší obvody s více logickými celky, do kterých se vejde větší a složitější konstrukce. Volba výrobce ovlivní i další rozhodování. Podle výrobce použijete vývojové prostředí (ISE WebPack nebo Quartus II), a každý výrobce používá jiné obvody pro programování přes JTAG. Jazyky naštěstí můžete použít u obou stejné.

Neexistuje obecná rada, jestli Xilinx nebo Altera. Já dlouho upřednostňoval Xilinx, teď mi připadají kity s FPGA od Altery dostupnější a propracovanější.

Styl práce se zas tak moc neliší. Pokud s FPGA začínáte, zvolte si jednu z těchto možností, později to můžete změnit. A jestli nevíte jakou, vyberte Alteru / Intel – důvod je, že na eBay koupíte levné čínské programátory pro Alteru levněji než levné čínské programátory pro Xilinx. *Navíc se mi zdá, že Quartus od Altery překládá VHDL rychleji než Xilinx ISE, nemluvě o tom, že starší Quartus v edici zdarma najdete na webu s menšími obtížemi než starší ISE WebPack a nové IDE Xilinx Vivado podporuje jen řady 7 a UltraScale, takže třeba pro kit s Virtexem 4 nebo se Spartanem 3 musíte hledat staré ISE, registrovat se a žádat o „licenci zdarma".*

Stručně: **Pokud jste začátečník a nevíte, kterého výrobce zvolit, odpověď zní Altera / Intel.**

A co Lattice?

Ano, jsou i další výrobci, jako například Lattice se svými obvody Mach, iCE nebo ECP5 / ECP3 / ECP2 / XP. I tento výrobce nabízí vývojové prostředí zdarma, včetně emulátoru, a až na některé detaily se práce s těmito obvody neliší od práce s FPGA od Xilinxu či Altery (Intelu). Někdy ale může být těžší sehnat vhodné vývojové desky či programátory.

VHDL, nebo Verilog?

Další rozhodování se bude týkat použitého jazyka. VHDL i Verilog jsou použitelné jazyky pro všechny FPGA i CPLD, je tedy na vás, co si vyberete. Oba jazyky jsou si do určité míry podobné svými schopnostmi a přístupem. Pro začátek si ale vyberte jeden, a ten se naučte. **Pokud nevíte**

jaký, bude to VHDL.

VHDL je populárnější v Evropě, v USA spíš Verilog. VHDL je trošku víc podobný jazyku Pascal, Verilog zase připomíná C. Jinak je to oblíbené dilema, o kterém se lze mnoho hodin přít. (Pro klid duše: Ano, lze použít komponentu, napsanou ve Verilogu, ve vlastním projektu v VHDL, a je to snadné.)

Co budete potřebovat?

1. Kit

Doporučuju pro úplný začátek malý kit s obvodem z rodiny Cyclone II: EP2C5T. Malý, a přesto dostatečně výkonný, abyste v něm rozběhli např. osmibitový počítač s procesorem Z80, pamětí a BASICem.

V ČR má tento kit v nabídce například e-shop HW Kitchen: https://hwkitchen.cz/

Pro zkušenější nebo náročnější mohu doporučit velmi slušně vybavený kit s EP4CE6E22, kde najdete i SDRAM, FLASH, VGA nebo PS/2, a přitom ho lze stále koupit za velmi zajímavé ceny.

2. Programátor

Čínská kopie USB Blasteru funguje a je k sehnání doslova za pár korun

3. Vývojové prostředí (IDE)

Altera nabízí Quartus II. Stahujte verzi 13.0 SP 1, ta podporuje použitý FPGA Cyclone II (novější jej už nepodporují). Ve verzi „Web Edition" je zdarma. Nyní, po odkoupení Altery společností Intel, bylo vývojové prostředí přejmenováno na Quartus Prime, a ve verzi Lite je zdarma.

https://fpgasoftware.intel.com/13.1/?edition=web

4. Znalost VHDL

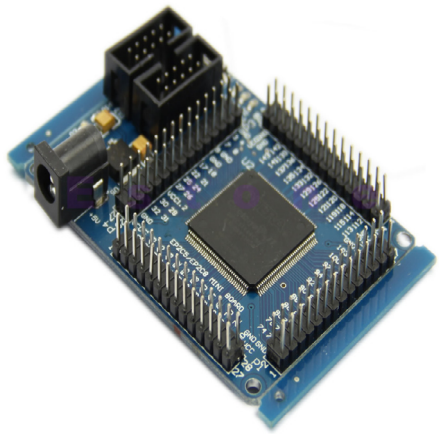
Najdete hned v další kapitole.

1.3 Jaký kit vybrat?

Já mohu doporučit velmi jednoduchý, levný, a přesto výkonný kit s obvodem Cyclone II od Altery, konkrétně EP2C5T144. Najdete ho na AliExpressu nebo na eBay. Klíčová slova jsou „EP2C5T144 FPGA Mini Development Board USB Blaster Programmer" – za sadu včetně programátoru USB Blaster, respektive jeho čínské kopie, dáte okolo čtyř stokorun (v době psaní knihy).

https://datacipy.cz/c2ebay/
https://datacipy.cz/c2ali/

Velká výhoda tohoto kitu je, že obsahuje vše nezbytné, ale nic navíc, takže máte k dispozici kompletní sadu vývodů. Nezapomeňte, že FPGA si většinou s pětivoltovou logikou moc nerozumí. Konkrétně tento obvod bude fungovat s logikou 3.3 V a nižší. 5 V na vstupu jej pravděpodobně zničí.

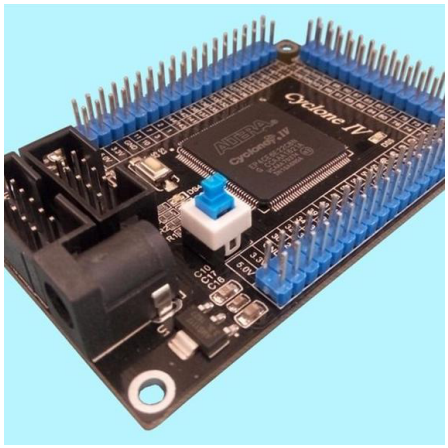


Kity s Cyclone IV

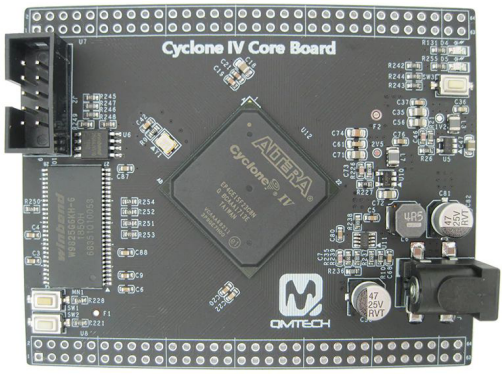
Výhoda kitů s čipy Cyclone IV je, že jsou jen o málo dražší, ale většinou podstatně vybavenější než ten nejmenší s Cyclone II. Často bývají integrovány různé periferie, jako LED nebo rozhraní pro VGA s konektorem, a někdy se objeví i integrovaná paměť SDRAM. Navíc podpora pro Cyclone IV je i v bezplatné verzi aktuální revize vývojového prostředí Quartus Prime Lite (v době psaní knihy šlo o verzi 19.1).

Můžete najít i jednoduché kity, podobné tomu předchozímu, za ceny okolo 700 Kč (v době psaní

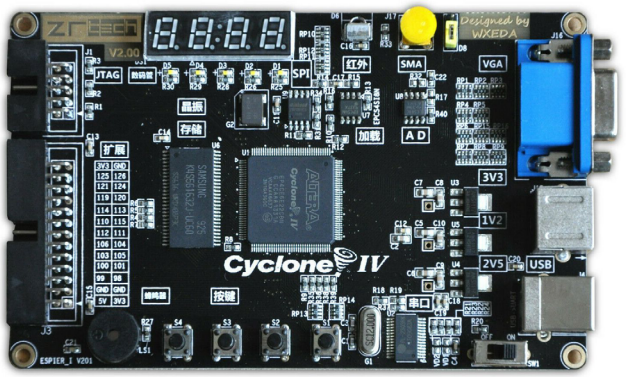
knihy). Hledejte klíčová slova „*FPGA Development Board EP4CE6E22C8N*“.



Za ceny do 1000 Kč lze koupit kit s větším čipem EP4CE15 a 32 MB SDRAM. Klíčová slova jsou, nepřekvapivě, „*EP4CE15 SDRAM*“

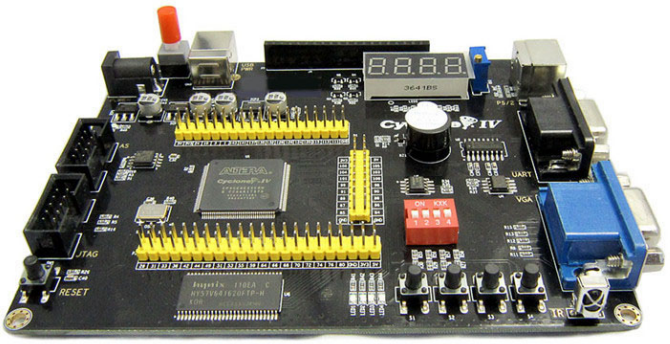


Lépe vybavené kity vás přijdou na částku lehce převyšující 1000 Kč, většinou tak do 1500 Kč. Není konkrétní výrobce ani převažující typ, spousta čínských výrobců si vyvíjí vlastní varianty, které se často liší od ostatních jen v detailech zapojení vývodů nebo rozmístění konektorů. Já například používám tento vývojový kit:



Většinou se jedná o různě upravené klony vývojových kitů Terasic DE (http://www.terasic.com.tw/), které se staly de facto standardem. I tyto kity můžete sehnat za ceny do 2000 Kč.

Já sám pro konstrukce, které vyžadují víc prostoru, používám „kit OMDAZZ" – nedělám si iluze o tom, že jde opět o nějaký *samo domo* klon čehosi, ale je dobře dostupný a dostatečně levný. Proto ho doporučuji i vám, pokud chcete zkusit nějakou náročnější konstrukci. Nevýhoda je, že není tak flexibilní jako kity „bez ničeho", výhoda naopak to, že obsahuje spoustu konektorů (sériový, PS/2, VGA), čidel (teplota, IR, tlačítka) i výstupů (LED, LED displej, konektor pro LCD displej, bzučák) ...



Podrobnější popis naleznete v příloze, když jej budete shánět, hledejte klíčová slova *omdazz* nebo *cyclone IV EP4CE6 board NIOSII FPGA*.

https://datacipy.cz/c4ebay
https://datacipy.cz/c4ali

E-shop HW Kitchen by měl mít tento kit také v nabídce, více na jejich stránkách: https://hwkitchen.cz

Pokud chcete větší a výkonnější čipy, můžete sáhnout např. po těchto deskách:

CYC1000 s Cyclone 10

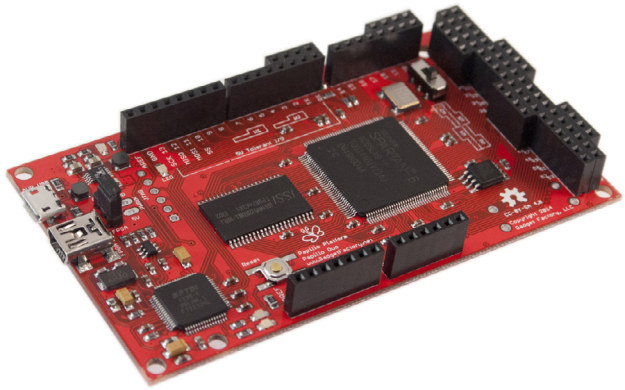
https://shop.trenz-electronic.de/en/TEI0003-02-CYC1000-with-Cyclone-10-FPGA-8-M-Byte-SDRAM

https://datacipy.cz/c10ebay

Kity s obvody Xilinx (Spartan atd.)

Známý kit, který používá nejmenší Spartan 3, se jmenuje Papilio One. Používá obvody XC3S250 nebo XC3S500, které se od sebe liší především množstvím použitelných logických bloků a dedikované SRAM (24 kB, resp. 40 kB). Ve verzi Pro najdete desku se Spartanem 6 XC6SLX9 (ovšem za vyšší cenu, okolo 80 USD). Zajímavá deska od stejného výrobce je Papilio Duo. Představte si Arduino, jak jej znáte, s AVR ATMega32, a spolu s ním na jedné desce Spartan 6 (XC6SLX9), 512 kB (nebo 1 MB) SRAM, USB rozhraní a konektory, kompatibilní s Arduinem. Cena je okolo 100 USD, podle velikosti použité paměti.

http://store.gadgetfactory.net/fpga/



Za ceny okolo 30 USD můžete sehnat i poměrně slušně vybavené kity s obvody z řady Xilinx Spartan 6, především pak XC6SLX9.

https://datacipy.cz/x6ebay
https://datacipy.cz/x6ali

I kity se Spartanem 7 lze pořídit okolo tisícikoruny. Opět zajímavá volba může být Spartan Edge Accelerator – shield pro Arduino se Spartanem 7S15, obvodem ESP32 a standardním rozhraním pro kameru. Dodává jej Seeeduino za 36 USD (v době psaní knihy):

https://www.seeedstudio.com/Spartan-Edge-Accelerator-Board-p-4261.html

Do sta dolarů seženete i kity s čipy Artix 7.

https://store.digilentinc.com/cmod-a7-breadboardable-artix-7-fpga-module/

https://www.seeedstudio.com/Perf-V-Based-on-Xilinx-Artix-7-FPGA-RISC-V-opensource--p-4058.html

Hotové konstrukce

Kromě popsaných kitů, které jsou určené primárně pro testování FPGA a vývoj s těmito čipy, existuje i kategorie zařízení takříkajíc „hackovatelných“. V těchto konstrukcích jsou použité přednastavené obvody FPGA, ale autoři více či méně úmyslně nechávají pokročilým uživatelům možnost nahrát vlastní konfiguraci a upravit si zařízení dle vlastní fantazie.

Autor jedné takové konstrukce, Gameduina, v dokumentaci přímo psal: „Funguje jako grafická karta k Arduinu, ale když vás to znudí, můžete jej přeprogramovat a používat jako FPGA kit s Arduinem.“

Takových zařízení je víc a některé z nich si ještě v knize představíme. Kromě zmíněného Gameduina nebo počítače V6Z80P (ani jedna konstrukce se už nevyrábí) to je například „reimplementace 8/16bitových počítačů“ MiST, Spectrum Next, nebo z poslední doby procesor MyMensch od společnosti Western Design Center.

Zmíněná společnost, založená spolutvůrcem procesoru 6502 Billem Menschem, dodneška tento procesor vyrábí, a nabízí kromě fyzické podoby i jeho HDL implementaci. MyMensch je poměrně levný a jednoduchý kit s čipem Altera / Intel 10M08S z řady MAX10. Z výroby je v ní nahrána konfigurace, která odpovídá procesorům 65C02, 65C816 nebo 65C165, spolu s obvody VIA a ACIA. Díky vyvedenému konektoru JTAG můžete i tento kit překonfigurovat dle své libosti.

První pokus s FPGA

Máme kit, programátor, nainstalované IDE, připojili jsme to k PC přes USB, a co dál?

Nejprve troška teorie.

FPGA je po zapnutí úplně tuhý kus křemíku, který k tomu, aby něco zajímavého dělal, potřebuje nejprve nakrmit konfiguračními daty. Ty bývají nejčastěji uložené mimo FPGA, v sériové paměti FLASH, ale můžete je při ladění nacpat do FPGA i přes programovací rozhraní JTAG.

Kit EP2C5T144 například nabízí dvě rozhraní, do kterých lze zapojit USB Blaster: JTAG a AS. Pomocí JTAG dostanete konfiguraci do FPGA při ladění. Přeložit, nahrát, testovat… Po vypnutí a zapnutí se ale načte nová konfigurace zase z FLASH. Pokud chcete uložit konfiguraci přímo do této paměti, použijete AS (Active Serial). Existují i způsoby, jak nahrát obsah do FLASH přes JTAG, tzv. „indirect programming“.

Pro první experiment využijeme JTAG.

Vlastní návrh proběhne ve vývojovém prostředí – IDE. Ovládání IDE není triviální, ale pokud máte nějaké zkušenosti s vývojovým prostředím typu Visual Studio, Eclipse apod., brzy se sžijete i s těmito.

Obě se od sebe liší, každé má jinak pojmenovaná menu, ale základ je stejný: ke každému pro-

jektu existuje Project. Project v jednom místě schraňuje všechny soubory, potřebné k naprogramování FPGA. Jde především o popis funkce v některém z jazyků (VHDL, Verilog a další). K těmto zdrojovým souborům si IDE vytvoří velké množství různých konfiguračních souborů (většinou je nemusíte editovat přímo, ale jsou na to v IDE nástroje). Jedním z takových nástrojů je nástroj, kterým můžete určit, který vývod FPGA má mít jakou funkci. (V IDE Quartus se tato funkce jmenuje Pin Planner.)

Překlad probíhá v několika krocích. Zase – názvosloví se liší, ale postup zhruba odpovídá následujícímu:

- První krok je analýza a syntéza. V něm se ze zdrojových kódů vytvoří návrh pro konkrétní FPGA. Překladač zkontroluje syntax, vyhodnotí logické výrazy, vazby mezi nimi, spočítá, kolik elementů bude potřeba, vytvoří seznam signálů, které bude potřeba připojit na piny FPGA…

- Ve druhém kroku se IDE snaží vhodně rozmístit komponenty do vnitřku FPGA. Zde se zohlední například i údaje z Pin Planneru. Po tomto kroku je jasno, zda se váš návrh do FPGA vejde, nebo zda je potřeba něco někde změnit.

- Třetí krok je vlastní překlad. Z výstupu druhého kroku se připraví konfigurační soubory, které se budou nahrávat do FPGA.

- Následují nejrůznější testy, hledání kritických míst, a závěrečný report, z něhož se např. dozvíte, nakolik jste využili možností svého FPGA.

Zde překlad končí. Další krok je vlastní programování do FPGA.

Kroky jsou přehledně vidět v IDE, takže víte, co už je splněno a co vás ještě čeká.

Nemusíte psát nutně všechno ve zdrojovém kódu – IDE obsahují i nástroje pro vizuální návrh, kde si obvod sestavíte, jako byste ho kreslili v Eagle nebo jiném EDA nástroji.

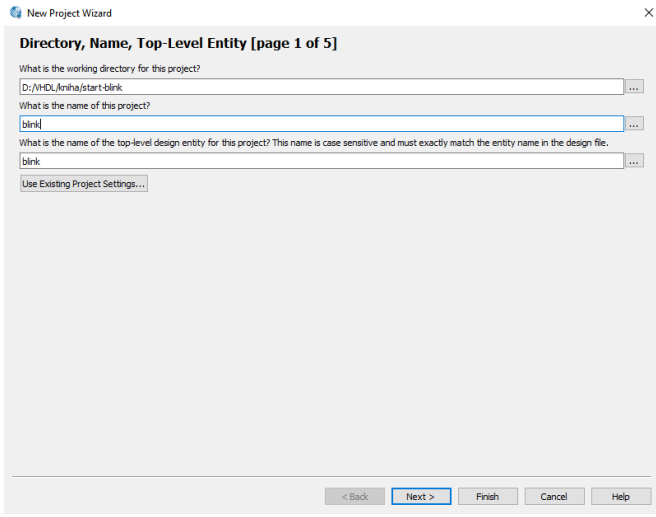
Hello world, model FPGA

Pokud máte všechno potřebné, tj. kit, programátor i IDE, můžete si zkusit „blikat LEDkou“, což je taková hardwarová obdoba Hello world.

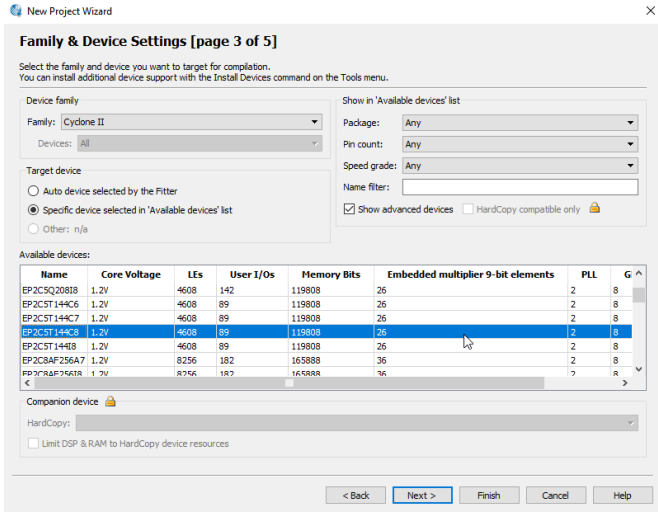
Budu popisovat blikání LEDkou pro kit EP2C5, programátor USB Blaster a IDE Quartus.

1. Vytvoření projektu

Začínáme vytvořením projektu. Jako vždy: File – New Project Wizard.



Projekt pojmenujeme „blink“ a pokračujeme (Next). Krok 2 jen přeskočíme (Next). Ve třetím kroku je potřeba vybrat použité FPGA. Zadejte rodinu „Cyclone 2“, čip je „EP2C5T144C8“.



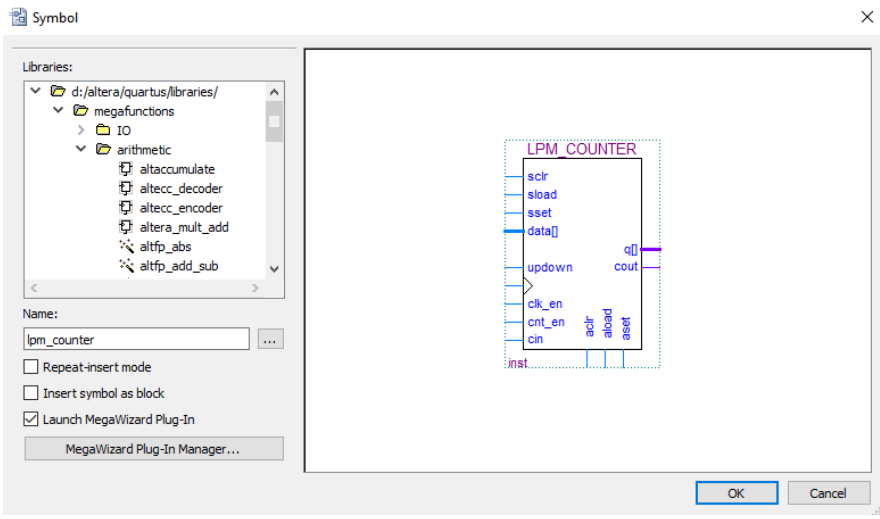
Next, next, finish…

2. Zapojení

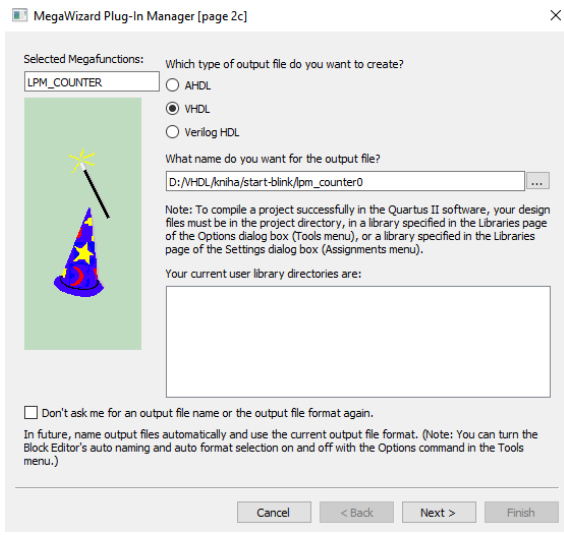
Blikání můžeme zařídit mnoha způsoby. Já vybral ten, kdy si na hodinový vstup (kam je připojený externí signál s kmitočtem 50 MHz) připojíme čítač, kterým podělíme frekvenci natolik, aby bylo blikání pozorovatelné pouhým okem. To znamená ideálně 24 bitů a víc. Kit má navíc 3 LED, takže nechám blikat všechny tři a zapojím je k čítači na výstupní bity 24, 25 a 26. Dělič nemusíme vytvářet z elementárních obvodů – Quartus obsahuje knihovnu (neskromně nazvanou Megalibrary), která obsahuje sadu nejrůznějších obvodů, od jednoduché logiky až po komplexní obvody typu řadiče SDRAM, rozhraní PCIe nebo síťové vrstvy PHY. Je mezi nimi i univerzální čítač.

Jak jsem slíbil výš, nebudeme obvod popisovat zdrojovými kódy, ale nakreslíme si ho. Vyberte tedy File – New – Block Diagram/Schematic File. Otevře se známý „tečkovaný papír“, kam můžete umístit komponenty a propojovat je.

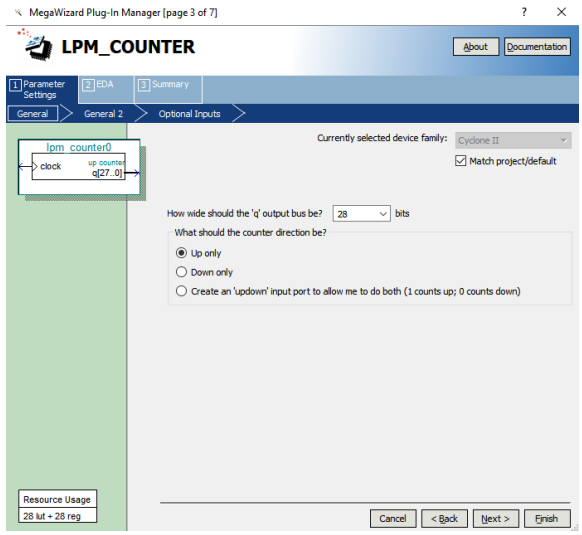
Nejprve tedy umístíme komponentu. Vyberte si z „Megafunctions“, složky „Arithmetic“ obvod, který se jmenuje „lpm_counter“.



Klikněte na OK, otevře se průvodce nastavením.

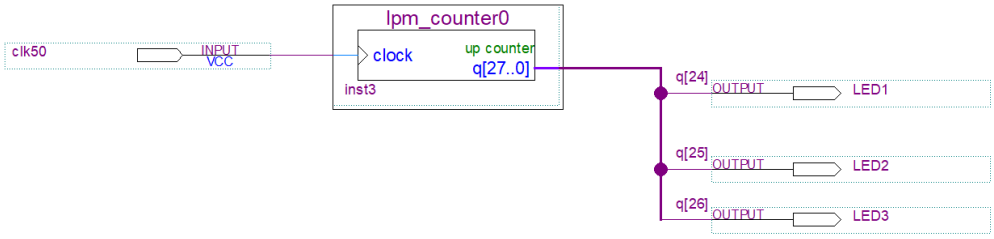


Jako jazyk zvolíme VHDL (teď je to jedno) a pojmenování necháme takové, jaké průvodce nabízí. Next. V dalším kroku zvolíme bitovou šířku. Já zvolil 28 bitů a čítání nahoru (mohu vytvořit i čítač dolů, popřípadě obousměrný). Další volby necháme tak, jak jsou nastavené, na konci klikneme na Finish.



Proč 28 bitů? Protože potřebuju, aby čítač zvládl napočítat takovou hodnotu, která bude blízká 50 milionům. 28bitové číslo může mít maximální hodnotu přes 268 milionů, což je dostatečný počet impulsů na to, aby byly vidět pouhým okem.

Teď nakreslíme zapojení. Na vstup připojíme vstupní pin, pojmenujme ho clk50. Na výstup připojíme sběrnici (Bus), která se bude jmenovat „q[27..0]“ – tedy má 28 linek. Využijeme z nich ale jen tři. Připravme si tři výstupní piny LED1-3 a připojme je ke sběrnici jako q[24], q[25] a q[26].

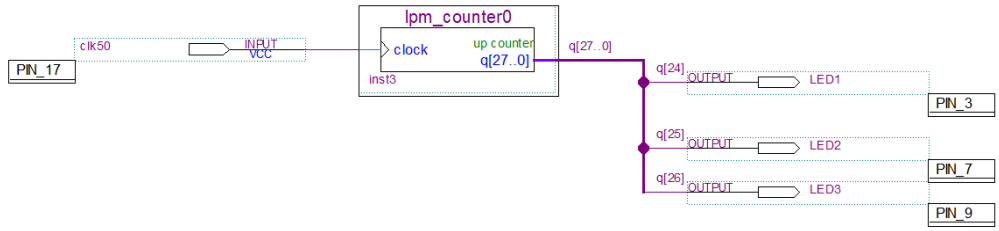


Soubor uložte jako „blink.bdf“, zkuste si zadat překlad (Processing – Start – Analysis & Synthesis, *Ctrl-K*), a pokud je vše OK, je načase připojit piny z nákresu k fyzickým. Otevřte si Pin Planner (Assignments – Pin Planner) a zadejte správné piny.

Jaké? Podle schématu kitu je hodinový vstup 50 MHz připojený na pin 17 a LED jsou připojené k pinům 3, 7 a 9. Stejně tedy přiřadíme i piny v planneru.

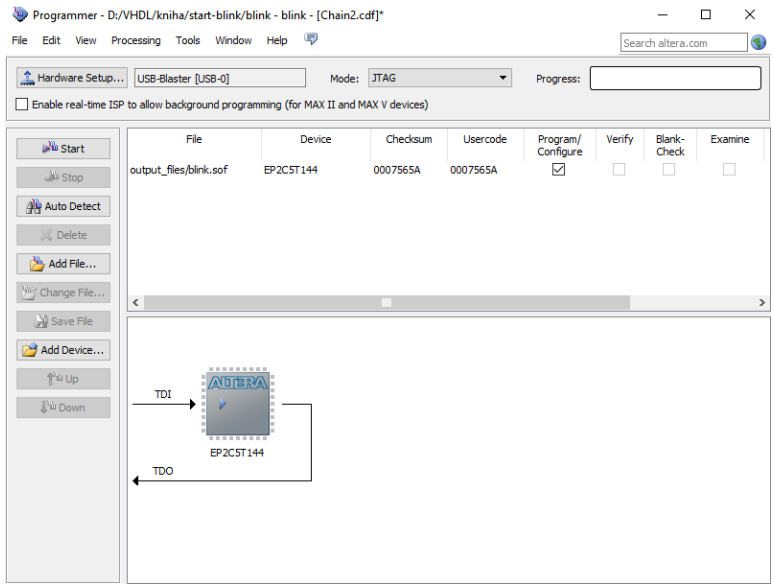
Node Name	Direction	Location	I/O Bank	VREF Group	Fitter Location	I/O Standard	Reserved	Current Strength	Differential Pair	Weak Pull-Up Resistor
clk50	Input	PIN_17	1	B1_N0	PIN_17	3.3-V LV...default)		24mA (default)		
LED1	Output	PIN_3	1	B1_N0	PIN_32	3.3-V LV...default)		24mA (default)		
LED2	Output	PIN_7	1	B1_N0	PIN_31	3.3-V LV...default)		24mA (default)		
LED3	Output	PIN_9	1	B1_N0	PIN_30	3.3-V LV...default)		24mA (default)		
<<new node>>										

Po zavření planneru vidíme, že se přiřazení promítlo do schématu:



Teď tedy můžete spustit celý překlad (Processing – Start Compilation, *Ctrl-L*). Měl by proběhnout bez chyb.

Pokud je vše v pořádku, je načase programovat. Připojte USB Blaster a počkejte, až se nainstaluje. Možná bude potřeba vyřešit ovladače… Blaster připojte do konektoru JTAG na kitu a kit zapojte k externímu zdroji s napětím 5 V. Měla by svítit POWER LED. Otevřte programátor (Tools – Programmer), vyberte jako nástroj „USB Blaster“, mód „JTAG“ (pokud byste programovali natrvalo, viz výše, zde zadáte AS). Pokud se vám neukáže vpravo soubor, přidejte ho ručně (Add File, najdete jej v adresáři output_files pod názvem blink.sof). A pak už stačí jen kliknout na Start.



Během několika sekund se LED na kitu rozblikají. Hurá!

Po chvíli zjistíte, že blikají nějak divně, že to počítání moc neodpovídá, jako by snad počítaly směrem dolů, a při pohledu na zapojení kitu vám to dojde: *LED nejsou připojené na zem, ale na Vcc.* Tedy inverzně. Jako první samostatné cvičení si můžete zkusit, jak do celého zapojení přidat tři invertory…

Pro kit OMDAZZ použijte stejný postup, jen s několika změnami:

- *Čip není Cyclone II, ale Cyclone IV, typ EP4CE6E22C8N*
- *Hodinový vstup CLK je na pinu 23*
- *LED jsou na pinech 85, 86 a 87*