ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA ELEKTROTECHNICKÁ

KATEDRA ŘÍDICÍ TECHNIKY

BAKALÁŘSKÁ PRÁCE

Hodiny s hlasovým výstupem v FPGA

Jan Libosvár
Vypracoval

Ing. Roman Bartosinski
Vedoucí bakalářské práce

Praha 2008

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne ……. …….
podpis

Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé bakalářské práce, Ing. Romanu Bartosinskému, za odbornou pomoc, cenné rady a připomínky, kterými přispěl k vypracování této bakalářské práce.

ANOTACE

Téma bakalářské práce pojednává o funkčním principu programovatelných hradlových polí FPGA a významu jazyka VHDL pro hardwarový popis (návrh). Dále je popsán mikrokontrolér PicoBlaze, vývojová deska Celoxica RC10 s FPGA obvodem Spartan3 3S1500L od firmy Xilinx a princip pulzní šířkové modulace. Prakticky je práce zaměřena na implementaci modulu audio výstupu a jeho předvedení na aplikaci hodin s hlasovým výstupem.

ANNOTATION

Subject of the bachelor thesis deals with a functional principle of programmable field arrays FPGA and meaning of VHDL langure in hardware description (design). Next is described microcontroller PicoBlaze , development kit Celoxica RC10 with Xilinx Spartan3 3S1500L FPGA and principle of pulse-width modulation. Practicall task is focused on implementation of audio module and its demonstration in application of clocks with voice output.

Obsah

1. ÚVOD

Účelem této bakalářské práce je seznámení se s návrhem a implementací pro programovatelná hradlová pole FPGA, hardwarovým popisem (návrhem) v jazyce VHDL a mikrokontrolérem PicoBlaze. Dalším cílem bylo seznámit se s vývojovou deskou RC10 s FPGA obvodem Spartan3 3S1500L od firmy Xilinx. Tato vývojová deska je, kromě jiného, opatřena přímo výstupem audio signálu.

Dále se práce zabývá implementací modulu audio výstupu, který je použit v aplikaci hodin reálného času. Hodiny každou hodinu oznámí čas. Pro implementaci audio modulu se používá pulzní šířková modulace, která představuje poměrně jednoduchý způsob, jak převést data uložená v paměti na analogový signál. Nahrané zvukové vzorky jsou uloženy v blokové paměti BRAM v FPGA. Nastavování času a vzorkovací frekvence pulzního šířkového modulátoru se provádí pomocí sériové linky UART.

2. TEORETICKÝ POPIS

2.1 Jazyk VHDL

Jak už zkratka VHDL – VHSIC (Very High Speed Integrated Circuits) Hardware Description Language - napovídá, jedná se o programovací jazyk na vyšší úrovni, který je určen k popisu hardware. Na základě tohoto popisu se syntetizérem vygeneruje popis na nižší úrovni – na úrovni jednotlivých logických hradel (makrobuňek, logických funkcí). Kvalita vygenerovaného popisu je dána právě syntetizérem. Dalšími HDL jazyky jsou například ABEL (popis na nižší úrovni), Verilog (užívaný spíše v USA), atd.

Jazyk vznikl na začátku 80. let minulého století jako program Ministerstva obrany Spojených států amerických pro vojenské účely ke specifikaci číslicových systémů. Účelem bylo vytvoření jazyka, který je přenositelný mezi jednotlivými výrobci a návrhářskými týmy. Kromě přenositelnosti mezi návrhovými systémy má být jazyk určen pro dokumentaci, modelování a simulaci[1] rozsáhlých systémů. Nyní je jazyk použitelný i pro syntézu[2]. VHDL je otevřený standard, takže k jeho používání není potřeba licence, jako je tomu např. u jazyka ABEL. Jazyk je vhodný pro návrh obvodů ultravysoké integrace.

Daná architektura v jazyce VHDL může být popsána různými styly zápisu. Hlavními styly jsou behaviorální, strukturální, případně styl popisující tok dat.

Behaviorální zápis (anglicky behavioral) popisuje přímo chování. Architektura je složena z procesů, které mají algoritmický charakter (příkazy, podmínky, cykly, zpracovávání proměnných atd.).

Styl popisující tok dat (anglicky dataflow) je speciální typ behaviorálního zápisu, kdy se kromě sekvenčních příkazů užívá i souběžných (paralelních) příkazů.

Strukturální (anglicky structural) popis spočívá ve vkládání komponent do více úrovní hierarchie a jejich propojení. Každá komponenta může být popsána zase jedním z těchto stylů. Komponenta na nejnižší úrovni je vždy popsána behaviorálně, resp. pomocí toku dat.

[1] Simulace logických obvodů – ověření funkce, zjištění časových parametrů před fyzickou implementací
[2] Syntéza VHDL – převedení zdrojového kódu do obecného schématu, které je v dalších krocích převedeno do logické struktury cílového obvodu

Styly zápisu nejsou striktně definovány, proto je v hardwarovém popisu v praktické části této práce použito kombinace obou hlavních stylů, tj. behaviorální a strukturální.

Větší podrobnosti ohledně syntaxe a struktury VHDL naleznete např. v [1].

2.2 FPGA

2.2.1 Zařazení pojmu FPGA

Akronym FPGA (Field Programmable Gate Array) je označení pro programovatelná hradlová pole. Architektura FPGA patří do skupiny obvodů programovatelných logických obvodů PLD (Programmable Logic Device). Kromě FPGA patří do této skupiny obvody SPLD [3] (Simple Programmable Logic Device – jednoduché programovatelné logické obvody)a CPLD [4] (Complex Programmable Logic Device – komplexní programovatelné logické obvody). Mezi nejznámější jména firem, zabývající se výrobou PLD, patří firmy Xilinx, Altera, Lattice Semiconductor, Atmel a jiné. V dnešní době dosahují obvody FPGA několik desítek miliónů ekvivalentních hradel[5]. Obvody FPGA se řadí mezi obvody s velmi vysokým stupněm integrace (VLSI – Very Large Scale Integration). Některé obvody FPGA je možno překonfigurovat přímo za běhu.

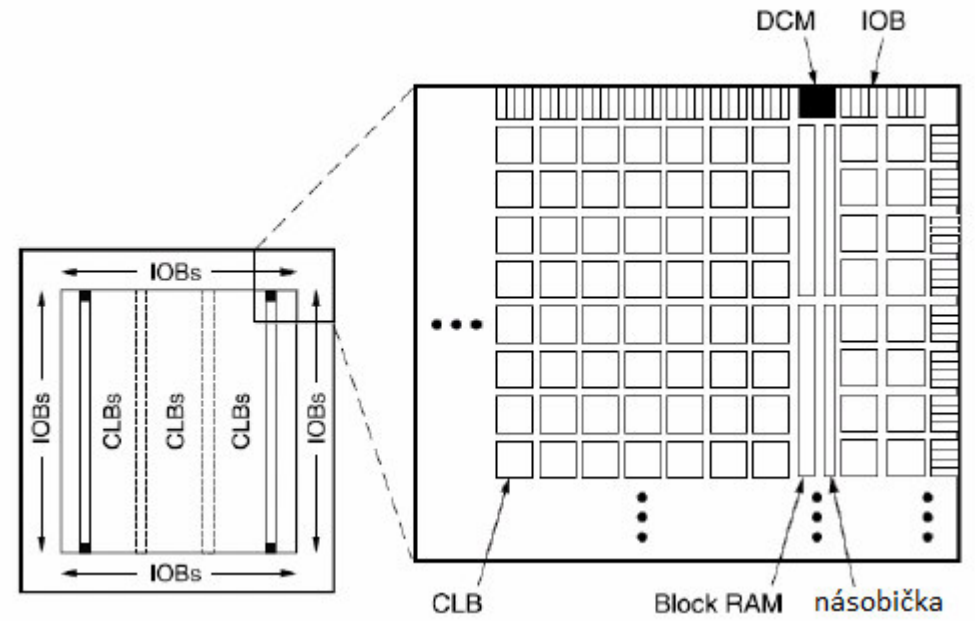
2.2.2 Struktura FPGA

V následujícím textu bude vysvětleno, jak obvody FPGA principiálně fungují. Popis bude vycházet ze stejné struktury a terminologie, kterou užívá firma Xilinx.

Typickou strukturu obvodu FPGA představuje matice konfigurovatelných logických bloků CLB (Configurable Logic Block). Kolem tohoto pole logických bloků jsou umístěny vstupně-výstupní bloky IOB (input/output block). Úkolem těchto bloků je propojení vnějších signálů (na jednotlivých pinech) a signálů v poli logických bloků CLB. Vstupně-výstupní bloky obvykle obsahují klopné obvody, neobsahují však žádnou kombinační logiku. Blokové schéma obvodu FPGA je uvedeno na obr. 2.1.

[3, 4] Bližší informace o obvodech SPLD a CPLD najdete v [1]

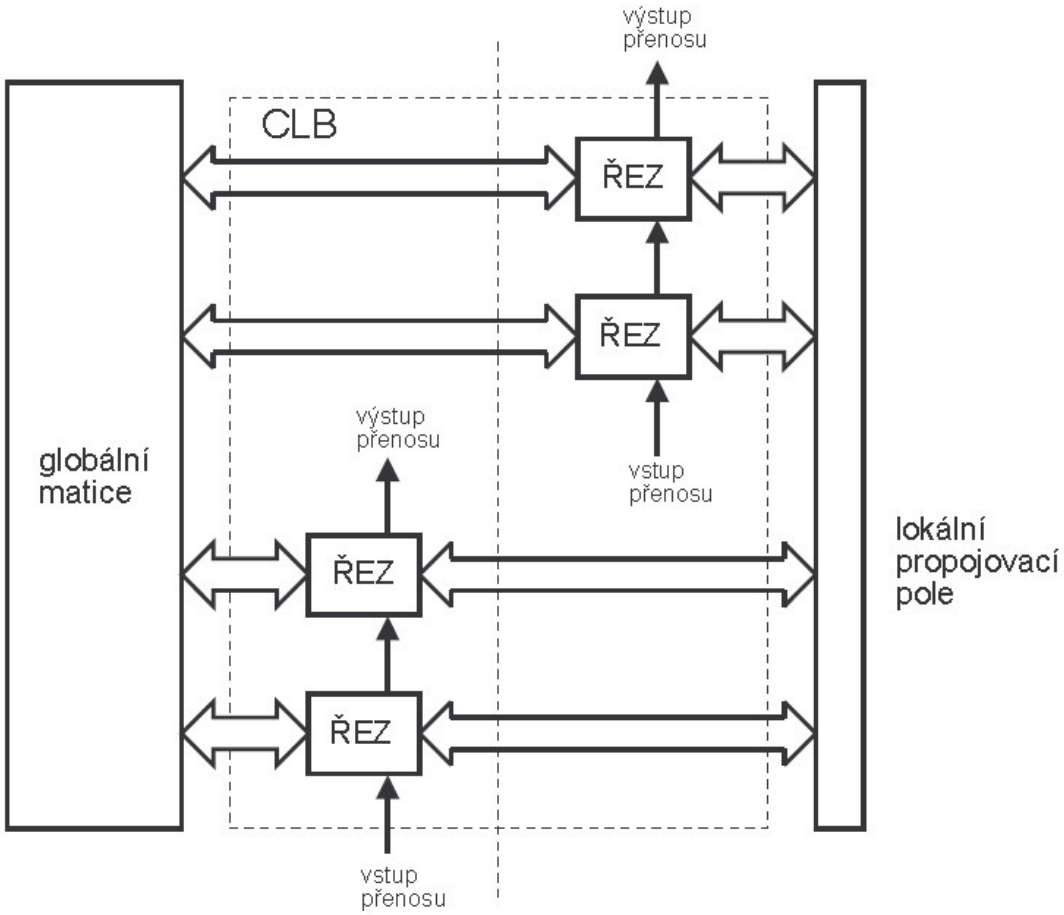
[5] Ekvivalentní počet hradel – měřítko, podle kterého se stanovuje velikost obvodu; udává počet, kolika dvouvstupovými hradly NAND a NOR je možno daný programovatelný obvod nahradit



Obr. 2.1 Blokové schéma obvodu FPGA

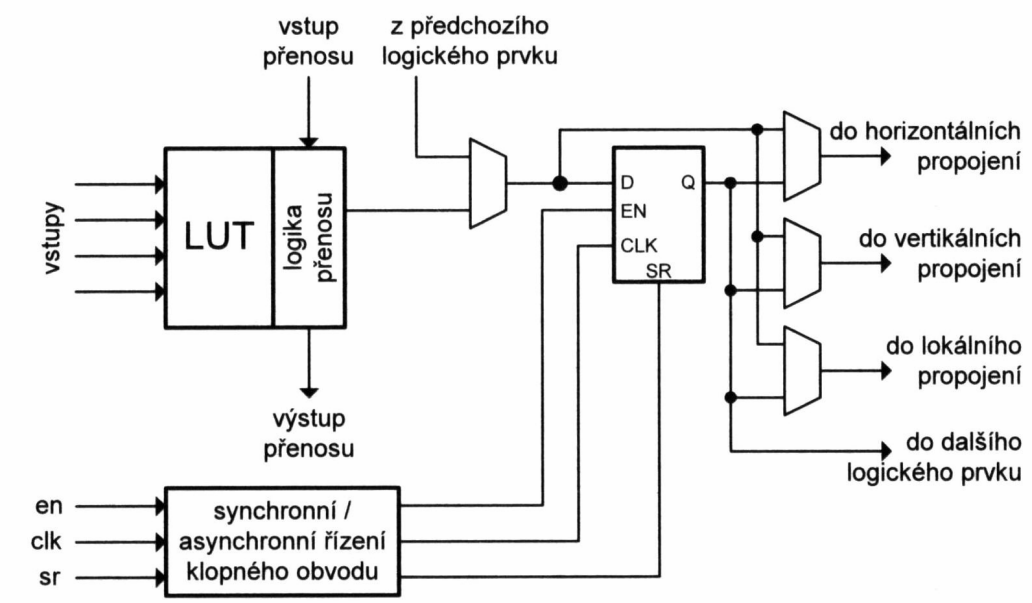
Dále jsou v poli logických bloků vestavěny blokové paměti BlockRAM, hardwarové násobičky, obvody pro řízení hodinového signálu DCM (Digital Clock Manager) a spousta dalších obvodů. Moderní FPGA obsahují dokonce celé procesory (novější obvody Virtex od firmy Xilinx). Pro vzájemné propojení všech bloků slouží globální propojovací matice. V případě sousedních logických bloků lze jejich signály propojit bez použití této propojovací matice. Tím dosáhneme menšího zpoždění přenosu. Tento parametr je důležitý například pro sčítačky či násobičky.

Konfigurovatelný logický blok (obr. 2.2) je složen z několika logických prvků neboli buněk (Logic Cell) a lokálního propojovacího pole, pomocí kterého se spojují jednotlivé logické buňky, případně logické buňky v nejbližších logických blocích. Obvody Spartan3 od firmy Xilinx, s jehož typem jsem pracoval v konkrétním hardwarovém návrhu, obsahují v každém logickém bloku 8 logických buněk rozdělených po dvou do 4 shodných řezů, tzv. SLICES. Následující obrázek ukazuje strukturu konfigurovatelného logického bloku, který obsahuje 4 řezy, 2 nezávislé přenosové řetězce (sloužící např. k realizaci rychlých sčítaček) a připojení ke globální propojovací matici a rychlé připojení k sousedním členům pomocí lokálního propojovacího pole.



Obr 2.2 Schéma logického bloku

Jeden logický prvek (polovina řezu) je složen z funkčního generátoru v podobě 4-vstupé vyhledávací tabulky (LUT – Lookup Table), rychlé logiky přenosu (realizace sčítaček, čítačů apod.), paměťového prvku v podobě klopného obvodu a bloku sloužícího k synchronnímu či asynchronnímu řízení tohoto klopného obvodu. Schematicky je toto znázorněno na obr. 2.3. Vyhledávací tabulka představuje jednoduchou konfigurovatelnou paměť ROM a slouží k vytváření jednoduchých funkcí. Dále je logická buňka opatřena multiplexory, pomocí kterých se propojují logické buňky do větších a složitějších celků dovolujících vytvářet složitější kombinační funkce. Pro co nejefektivnější syntézu existují různé algoritmy implementace používané v návrhových nástrojích jako je Xilinx ISE (Integrated Software Environment).

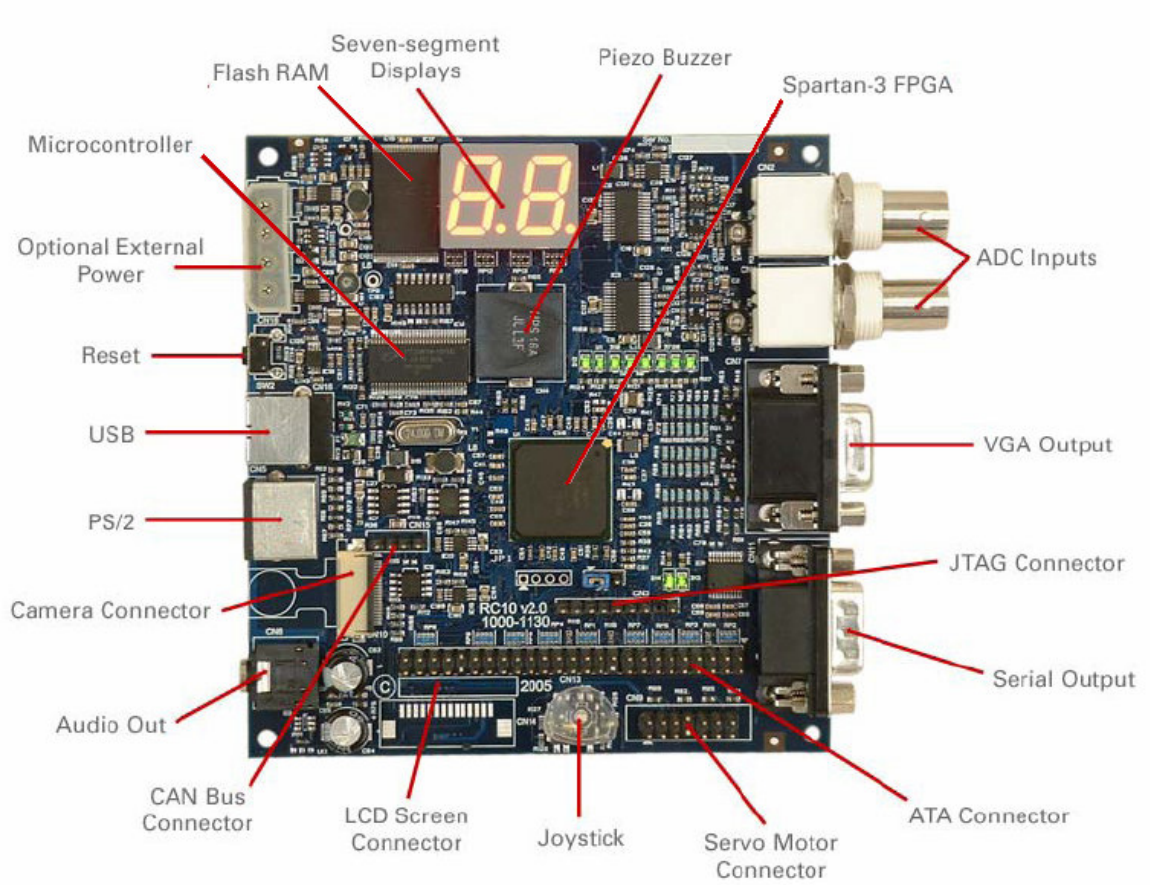


Obr. 2.3 Blokové schéma logické buňky

3. POUŽITÉ PROSTŘEDKY

3.1 Vývojová deska Celoxica RC10

Vývojová deska RC10 od firmy Celoxica je nejjednodušší zařízení ze série RC desek, které najde pro svou jednoduchost uplatnění hlavně při výuce a seznamování se s návrhem aplikací založených na FPGA. Deska je však díky různým periferiím a vstupně/výstupním konektorům (viz obr. 3.1) univerzální a dá se použít k velkému množství sofistikovaných aplikací.



Obr. 3.1 Celoxica RC10

Deska je osazena FPGA Spartan 3 XC3S15000L v pouzdře FG320 (rozměr 23 x 23 mm). Na piny FPGA jsou přímo připojena tato zařízení:

- USB mikrokontrolér Cypress CY7C68013-56pvc FX2 – slouží k řízení spojení mezi USB portem a připojeným PC, provádí konfiguraci samotného FPGA a ovládá přístup do paměti FLASH
- video výstup s VGA konektorem
- RS232 sériový port
- audio výstupy – stereo pwm výstup s RC sítí
  – piezo měnič
- PS/2 konektor pro připojení klávesnice nebo myši
- 50 pinový konektor
- rozhraní pro připojení ke sběrnici CAN
- konektor pro ovládání až 4 servomotorů
- 2 vstupy analogově digitálního převodníku
- 8 zelených LED diod
- 2 sedmisegmentové displeje
- 5-ti kontaktový mikrojoystick

Dále jsou na FPGA připojeny:

- hodinový signál o frekvenci 48 MHz a další 2 vstupy pro externí zdroj hodinového signálu
- 2 LED diody pro indikaci napájení právě provedené rekonfigurace FPGA
- paměť flash RAM o velikosti 16 MB – slouží k ukládání konfigurací pro FPGA, je přístupná pouze přes USB mikrokontrolér
- tlačítko reset – provádí nahrání konfigurace FPGA z prvního paměťového místa flash paměti
- JTAG konektor

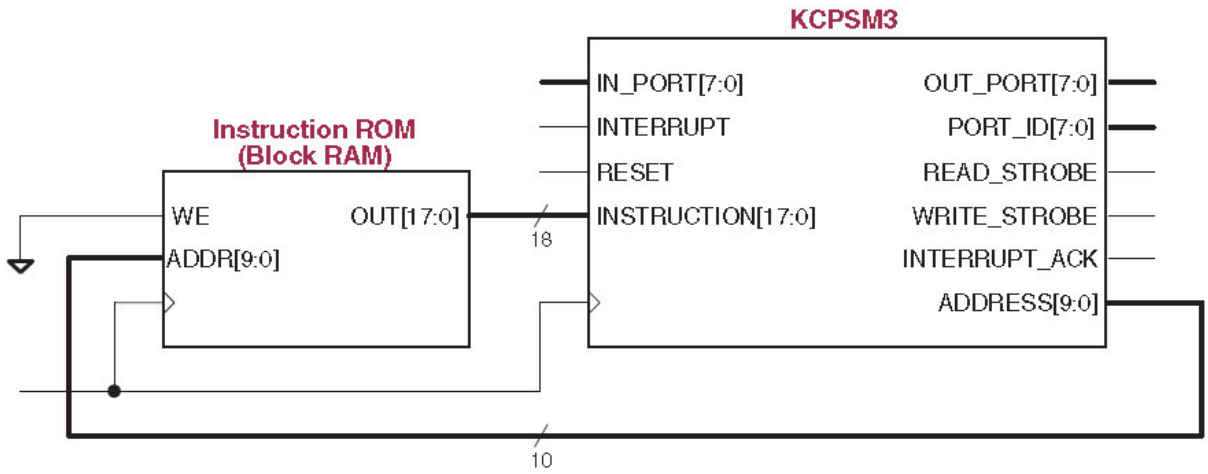
3.2 Mikrokontrolér PicoBlaze

Mikrokontrolér PicoBlaze ve skutečnosti není žádná hardwarová součástka, jak by si mnozí mohli myslet, ale soft-core 8-bitový mikrokontrolér společnosti Xilinx realizovaný v hradlovém poli FPGA. Tento mikrokontrolér je speciálně navržen a optimalizován pro FPGA série Spartan a Virtex.

Mikrokontrolér PicoBlaze na čipu FPGA obsazuje pouhých 192 logických buněk, může však dosahovat vysokého výkonu v podobě až 100 MIPS (milión instrukcí za sekundu) podle cílového FPGA. PicoBlaze je k dispozici zdarma ve formě syntetizovatelného VHDL kódu.

PicoBlaze je 8-bitový RISC mikrokontrolér, který zpracovává 18-ti bitové instrukce uložené v jedné paměti BlockRAM. PicoBlaze dokáže adresovat paměť o velikosti 1024 programových instrukcí. Každá instrukce se vykoná ve dvou hodinových taktech. Pro převod zdrojového kódu mikrokontroléru se používá KCPSM3 assembler, který podle šablony vytvoří paměť BlockRAM s daným programem.

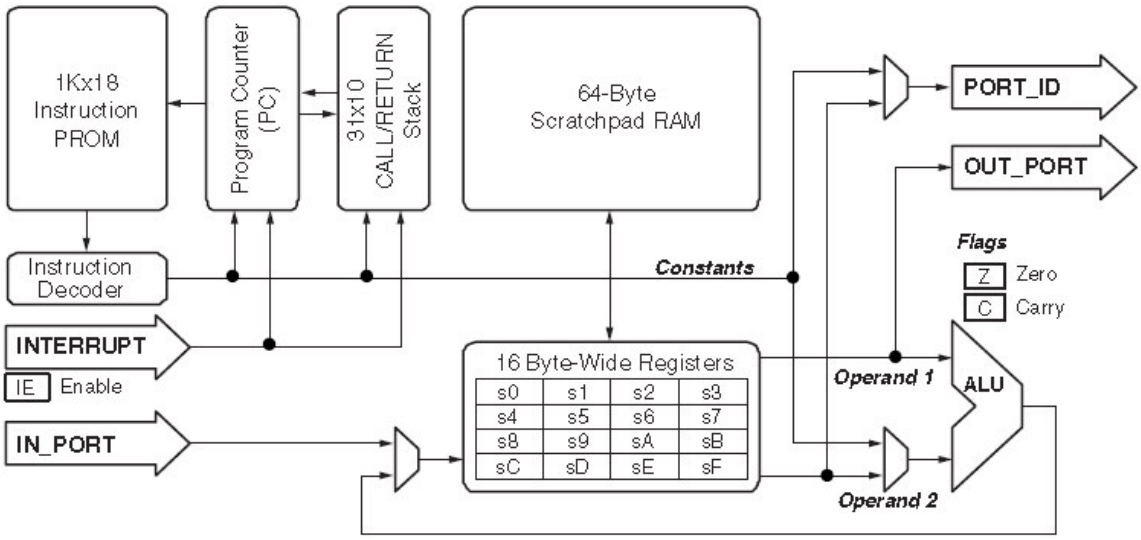
Připojení paměti instrukcí k mikrokontroléru a jeho vstupy a výstupy jsou na obr. 3.2.



Obr. 3.2 Připojení paměti k mikrokontroléru PicoBlaze

Mikrokontrolér dále nabízí (viz. obr. 3.3)

- 16 obecně použitelných registrů s0 - sF o velikosti 1 byte – pro lepší přehlednost je registry možno pojmenovat a používat pod symbolickými názvy
- aritmeticko logickou jednotku ALU, která provádí základní aritmetické operace (sčítání, odčítání), bitové logické operace, porovnávání, posun a rotaci registrů. Výsledek některých operací ovlivňuje příznaky ZERO a CARRY
- 64 bytovou paměť, kterou lze adresovat přímým nebo nepřímým adresováním
- 256 vstupních a 256 výstupních portů
- zásobník pro uložení 31 adres



Obr. 3.3 Blokové schéma mikrokontroléru PicoBlaze

3.2.1 Vstupní a výstupní porty

PicoBlaze disponuje 256 vstupními a 256 výstupními porty. Adresa portu je při vstupní nebo výstupní operaci uvedena na výstupním portu PORT_ID. Správné připojení určitého zařízení na vstup mikrokontroléru (respektive připojení výstupu mikrokontroléru na dané zařízení) provádí externí logika FPGA (v podobě multiplexerů resp. demultiplexerů). Při vstupní (výstupní) operaci se ve druhém taktu nastaví signál READ_STROBE (WRITE_STROBE), který určuje připravenost k dané operaci.

3.2.2 Čítač instrukcí

Čítač instrukcí (PC - Program Counter) ukazuje na instrukci, která se má vykonat v dalším cyklu. Jde o 10-ti bitový registr, který se automaticky inkrementuje. Instrukce skoku v programu, skok do podprogramu, přerušení a návrat z přerušení naplní čítač danou adresou. Reset mikrokontroléru čítač instrukcí vynuluje. Pokud dojde k přetečení čítače, začíná čítat opět od 0.

3.2.3 Zásobník

Zásobník slouží k uložení až 31 adres při operaci volání podprogramu a operaci přerušení. Pokud je zásobník celý naplněn, dochází k cyklickému posunu adres a nejstarší adresy se ztrácejí. Se zásobníkem také souvisí ukládání příznaků ZERO a CARRY. Tyto příznaky se při volání podprogramů neukládají a při návratu z podprogramů se s tím musí počítat. Při obsluze přerušení se ZERO i CARRY ukládají.

3.2.4 Přerušení

Jak je vidět na obrázku 3.2, PicoBlaze disponuje jedním vstupním signálem pro přerušení. Přerušení se obslouží, pokud je nastaven příznak INTERRUPT_ENABLE (IE). Při přerušení se na vrchol zásobníku uloží aktuální hodnota v čítači instrukcí a vzápětí se do něj nastaví adresa 3FF hex. To je adresa vektoru přerušení (poslední instrukce v paměti instrukcí). Dále se uloží příznaky CARRY a ZERO. Po obsluze přerušení se obnoví původní příznaky a jako adresa následující instrukce v PC se obnoví adresa z vrcholu zásobníku. Přerušení se obslouží v nejhorším případě za 5 strojových cyklů. PicoBlaze na příjem přerušení odpoví nastavením výstupního signálu INTERRUPT_ACK.

3.2.5 Reset

Vstup mikrokontroléru RESET restartuje PicoBlaze do původního stavu. Signál RESET je také vygenerován automaticky po konfiguraci FPGA. Vynulují se příznaky ZERO, CARRY, INTERRUPT_ENABLE, čítač instrukcí a ukazatel zásobníku. Registry s0 – sF, 64 bytová RAM paměť a samozřejmě paměť instrukcí zůstanou zachovány.

4. PRAKTICKÉ ŘEŠENÍ

4.1 Zvukové vzorky

Na vývojové desce RC10 není žádná paměť přímo přístupná obvodu FPGA (pouze paměť typu FLASH, která je přístupná jedině pomocí USB mikrokontroléru). Vzorky jsem tedy uložil přímo do struktury FPGA. Protože jsem pro uložení vzorků omezen velikostí použitého FPGA, snažil jsem se, aby tyto vzorky zabíraly co nejmenší místo. Frekvence lidského hlasu se přibližně pohybuje do 3,5 kHz, proto je nutné lidský hlas zaznamenávat a vzorkovat s frekvencí alespoň dvakrát větší (Shannon-Kotělnikův teorém). Pokud bychom tuto podmínku nesplnili, došlo by k překrytí frekvenčních spekter (aliasingu) a při následné reprodukci bychom získali zkreslenou informaci, tedy nepříliš čistý zvukový signál. Proto jsou použité vzorky zvuku nahrány a uloženy se vzorkovací frekvencí 8000 Hz. Jednotlivé vzorky jsou kvantovány v 8 bitech (256 kvantizačních úrovní) a to pouze jeden kanál (mono). I když jsem použil takto nízké rozlišení, zvukový výstup je pro tento účel dostatečně slyšitelný. Jedna sekunda nahraného zvuku bude tedy zabírat 8000 bytů paměti.

4.2 PWM modulace

4.2.1 Princip PWM

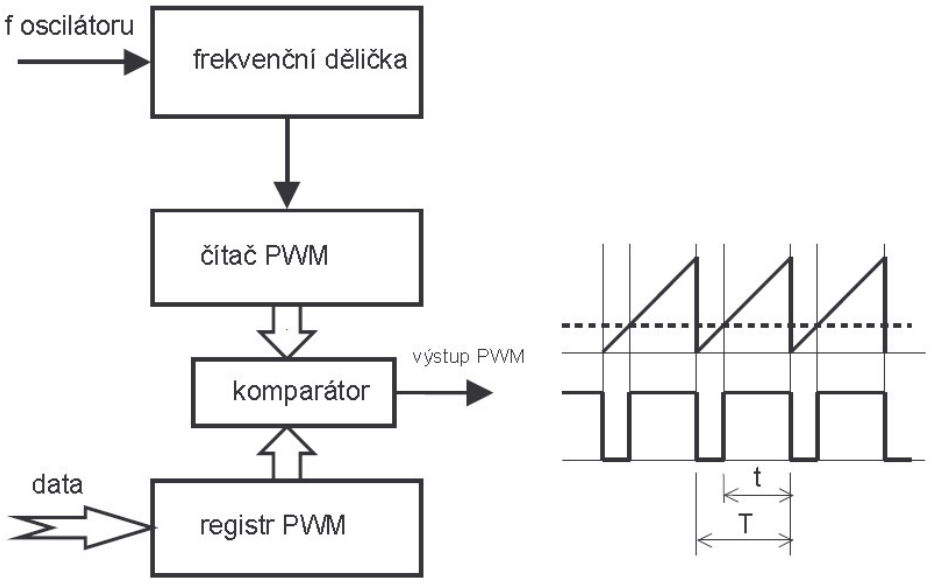
Pro reprodukci uloženého zvuku je vhodné použít pulzní šířkové modulace (PWM[6]). Pomocí PWM modulace se jednoduše převedou číslicová data na analogový signál. Signál kódovaný pulzní šířkovou modulací má konstantní periodu T a mění se šířka pulzu, kdy je signál v logické 1. Poměr doby, kdy je signál v log. 1, a periody se nazývá střída a je přímo úměrný střední hodnotě napětí, jak dokazuje následující vztah:

$$U_{stř} = \frac{1}{T}\int u(t)dt = \frac{1}{T}\int_0^t Ud\tau = U\frac{t}{T}$$

[6] - Pulse-width modulation

4.2.2 Generování PWM

Číslicové generování PWM signálu je vysvětleno pomocí obr. 4.1. Do frekvenční děličky je přiveden hodinový signál oscilátoru. Frekvenční dělička slouží k nastavování doby periody PWM signálu. Blok čítač PWM je 8 bitový čítač, který se zvyšuje s každou náběžnou hranou signálu z frekvenční děličky. Hodnota PWM čítače se komparátorem porovnává s hodnotou uloženou v PWM registru. Když je hodnota čítače větší než porovnávaná referenční hodnota, PWM výstup se překlopí z 1 do 0 (případně z 0 do 1).



Obr. 4.1 Princip pulzní šířkové modulace

4.3 Modul audio výstupu

Pro implementaci modulu audio výstupu se použije zmíněného pwm modulátoru s nastavitelnou periodou vzorkování, jehož princip byl vysvětlen v předchozí kapitole. Pokud se nepoužije frekvenční dělička, perioda modulátoru bude dána pwm čítačem, konkrétně hodnotou, do které čítá. Při použití 8 bitového čítače se čítač vynuluje po 256 hodinových taktech. Frekvence pwm modulátoru je tedy $f_{osc}/256$. Vývojová deska Celoxica RC10 používá hodiny o taktu 48 MHz. Ty jsou přivedeny na pwm modulátor, takže frekvence pwm modulátoru je $f_{pwm}=48 \cdot 10^6 / 256=187\ 500$ Hz. To je pro případ, kdy je dělička rovna 1, tedy maximální možná frekvence pwm modulátoru. Frekvenční dělička dokáže dělit frekvence pouze celočíselně, proto jsem vybral 8 hodnot, pro které jsou všechny výsledky dělení celočíselné bezezbytku. Hodnoty a příslušné frekvence jsou uvedeny v tab. 4.1.

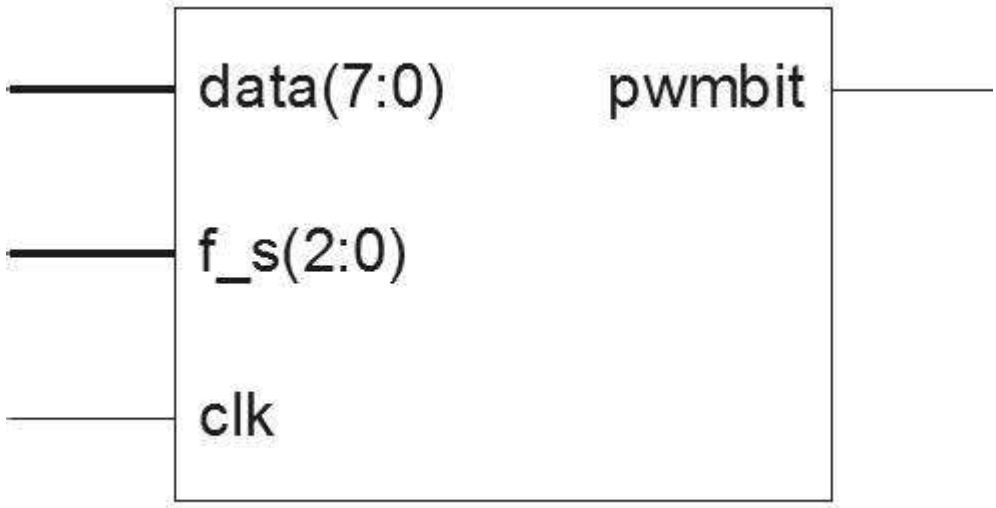
Tab. 4.1 Hodnoty frekvenční děličky příslušné frekvenci modulátoru

f_{PWM} (Hz)	**Hodnota frekvenční děličky**
7 500	25
12 500	15
18 750	10
31 250	6
46 875	4
62 500	3
93 750	2
187 500	1

Frekvenční dělička funguje jako čítač, takže například pro získání modulační frekvence 62 500 Hz bude hodnota čítače (děličky), do které se bude čítat, rovna 3. Čítač se po dosažení této hodnoty resetuje a čítá od 0. Zároveň se inkrementuje hodnota pwm čítače. Pwm čítač se inkrementuje s frekvencí $48 \cdot 10^6 / 3 = 16$ MHz, ale má 256 stavů, takže výsledná frekvence modulátoru bude $16 \cdot 10^6/256=62\ 500$ Hz.

Tabulka, ze které se vybírá hodnota frekvenční děličky, je v hardware implementována pamětí ROM. Ta se adresuje 3 bitovým signálem přicházejícím z vnějšku tohoto modulu.

Hodnota pwm čítače se komparátorem porovnává s 8 bitovým registrem, do kterého jsou posílány zvukové vzorky stejnou rychlostí, jakou byly navzorkovány. V případě použití modulu audio výstupu pro reprodukci hlasu je to tedy 8000 S/s. Z tohoto důvodu musí pracovat pwm modulátor na frekvenci větší než 16 kHz, aby byl dodržen Shannon-Kotělnikův teorém. Podle výsledku komparace se překlápí výstupní signál v podobě pwm modulace.

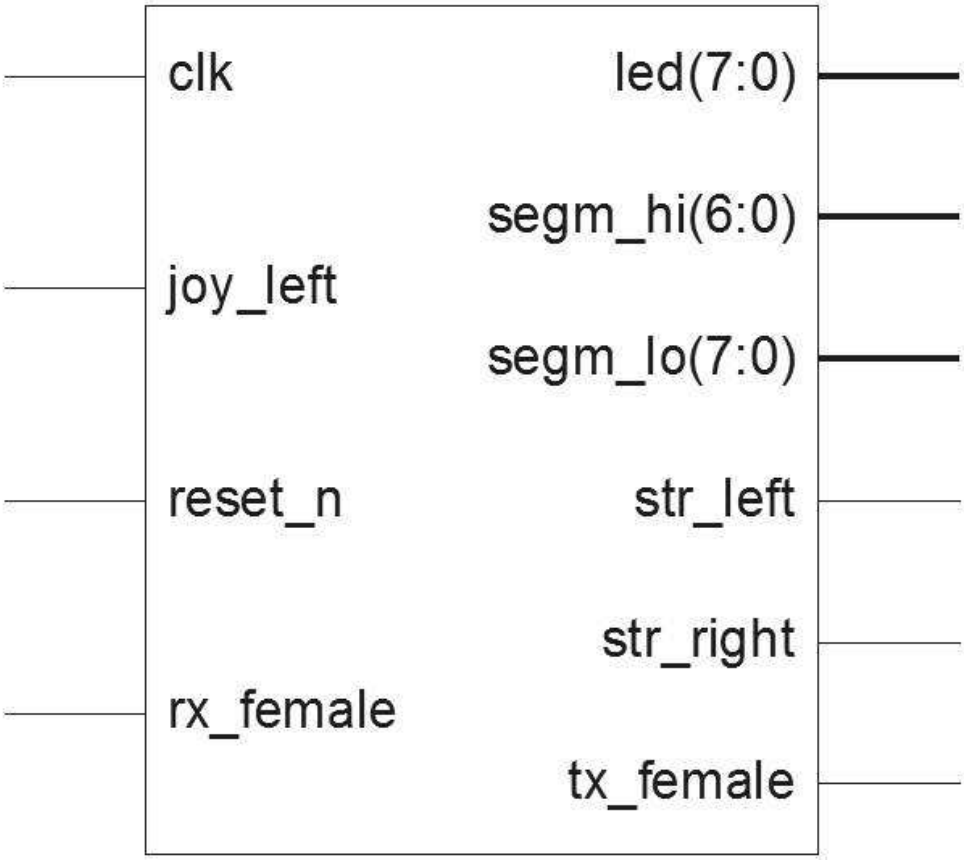


Obr. 4.2 Komponenta modulu audio výstupu

Na obr. 4.2 je znázorněna komponenta modulu audio výstupu použitá ve vyšší hierarchické úrovni v aplikaci hodin s audio výstupem.

4.4 Demonstrace funkčnosti modulu audio výstupu

Pro demonstraci funkčnosti vytvořeného modulu byla zvolena aplikace hodin, které budou oznamovat čas v podobě hlasového výstupu. Mikrokontrolér PicoBlaze je zde použit pro počítání času, jeho zobrazování na dvoumístném 7-segmentovém displeji a adresování paměti se zvukovými vzorky, které se posílají do komponenty starající se o převod hlasu. Program pro PicoBlaze je popsán v kapitole 4.4.4**.** Pomocí mikrokontroléru se dále nastavuje vzorkovací frekvence audio modulu. RTL schéma celé entity znázorňující použité vstupy a výstupy je na obr. 4.3. Tyto vstupy a výstupy jsou již přímo vyvedeny na piny FPGA a odkud vedou k daným periferiím.



Obr. 4.3 Vstupy a výstupy použitého hardwarového návrhu

4.4.1 Popis ovládání hodin

Po připojení napájení (USB kabelu) se rozsvítí LED dioda oznamující, že deska RC10 je pod napětím. Na dvoumístném 7-segmentovém displeji se zobrazí „00“. První 3 v řadě umístěné LED (počítáno zleva) jsou použity pro indikaci použité vzorkovací frekvence pwm modulátoru. Svítí 1. a 3. LED zleva (bity 7 a 5). Ty značí, že vzorkovací frekvence pwm modulátoru je nastavena na 62 500 Hz. Zbývající LED nesvítí. Hodiny mají nastavený čas „00:00:00“. Na 7-segmentovém displeji se zobrazují střídavě hodiny a minuty: 3 sekundy hodiny, dalších 7 sekund minuty. Zobrazení hodin je indikováno svítící desetinnou tečkou na 7-segmentovém displeji pro zobrazení jednotek. V případě, že jsou zobrazeny minuty, lze podržením levého tlačítka joysticku zobrazit hodiny. Každých 10 sekund se postupně směrem zprava doleva rozsvěcuje zbývajících 5 LED. Tím získáme přibližnou představu o vteřinách.

Pro nastavení správného času je potřeba k vývojové desce připojit sériové rozhraní UART. Sériová linka komunikuje rychlostí 115 200 Bd, 8 data-bit, 1 stop-bit bez parity. Pomocí sériové linky se také nastavuje frekvence pwm modulátoru. Použitelné příkazy včetně rozsahu parametrů uvádí tab. 4.2.

Při stisku tlačítka joysticku dolů se provede reset vzorkovací frekvence a vynulování vteřin. Každou hodinu se v připojeném reproduktoru ohlásí čas (hodiny v rozmezí 1 hodina – 12 hodin).

Tab. 4.2 Prikazy pro ovládání hodin

Příkaz	**Význam, přípustné hodnoty parametrů**
cas *hh:mm*	nastaví čas, *hh* = <0-23>, *mm* = <0-59>
hod *hh*	nastaví pouze hodiny, *hh* = <0-23>
min *mm*	nastaví pouze minuty, *mm* = <0-59>
sample *n*	vybere vzorkovací frekvenci, *n* = <0-7>

Tab. 4.3 Hodnoty frekvencí pwm modulátoru

n	**f_{PWM} (Hz)**
0	7 500
1	12 500
2	18 750
3	31 250
4	46 875
5	62 500
6	93 750
7	187 500

4.4.2 Popis funkce hodin

Obr. 4.4 znázorňuje zjednodušené blokové schéma celkového zapojení aplikace. Mikrokontrolér PicoBlaze adresuje pomocí signálu ADDRESS paměť programu. Instrukce jsou přiváděny na vstupní port INSTRUCTION.

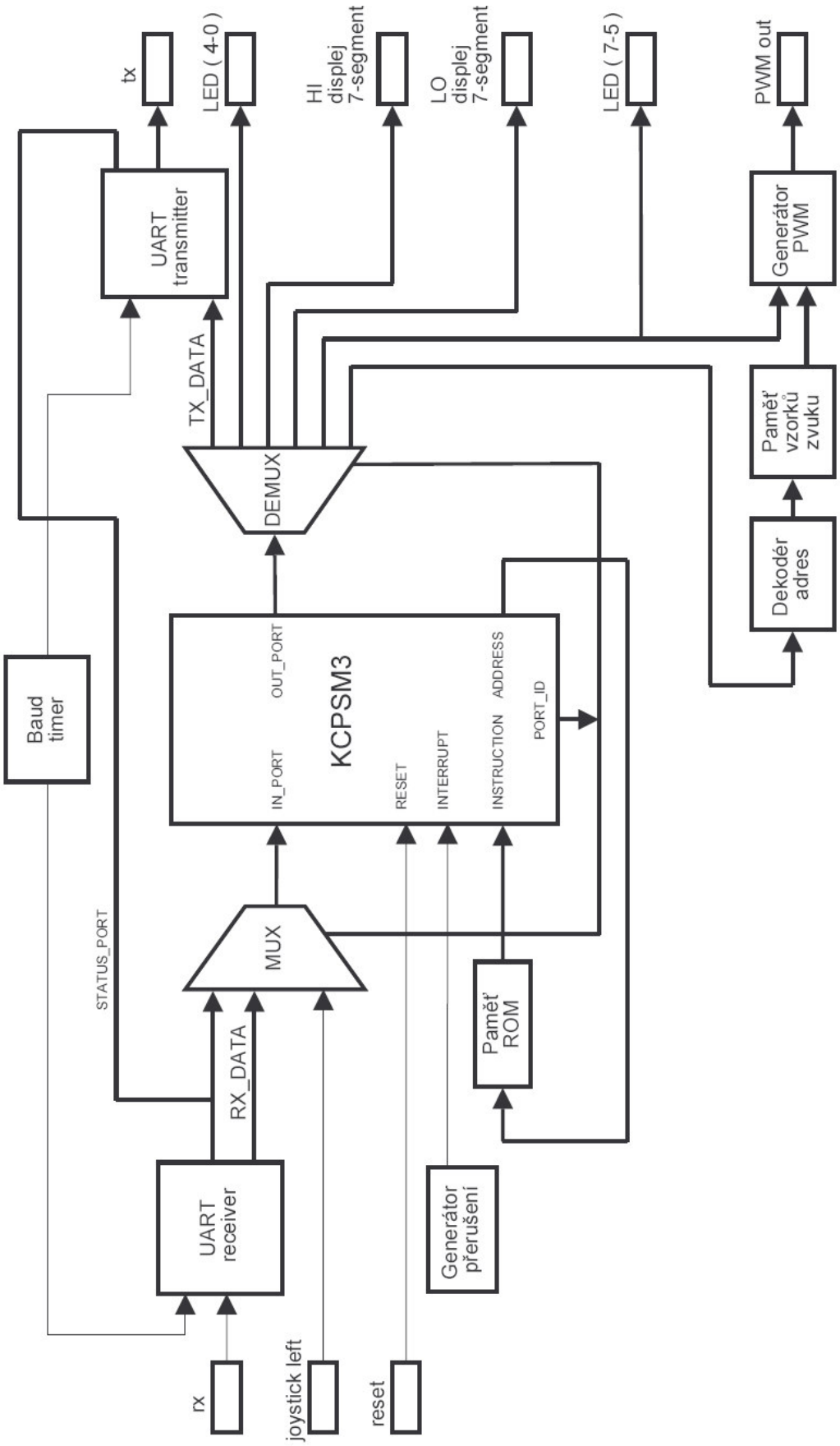
Pomocí čítače pracujícího jako frekvenční dělička se generuje každou sekundu krátký impulz, který je připojen na vstup INTERRUPT mikrokontroléru. Přerušení se využívá k počítání sekund.

Přes multiplexery jsou na vstup IN_PORT připojeny :

- levé tlačítko mikrojoystick
- stavové signály vysílače a přijímače sériového rozhraní
- buffer přijímače sériového rozhraní

K výstupním portu OUT_PORT se přes demultiplexer připojují následující zařízení:

- LED
- 7-segmentový displej – jednotky
- 7-segmentový displej – desítky
- modul audio výstupu pro nastavení vzorkovací frekvence
- dekodér adres pro paměť vzorků



Obr. 4.4 Zjednodušené blokové schéma aplikace

O aktivním vstupu multiplexeru (resp. výstupu demultiplexeru) rozhoduje signál PORT_ID. Nastavování signálů write_to_uart a read_from_uart se také řídí pomocí portu PORT_ID.

Pro sériovou komunikaci UART bylo použito doporučené zapojení Kena Chapmana ze společnosti Xilinx. To využívá na straně příjmu (resp. odesílaní) 16 bytový buffer a samotný přijímač (resp. vysílač). Podrobnosti v [6]. Vysílač i přijímač potřebují pro správný chod hodiny o frekvenci 16krát vyšší, než je přenosová rychlost. Tuto funkci zajišťuje čítač v podobě frekvenční děličky.

Na výstupní porty PicoBlaze je připojen dekodér adres. Ten podle desítek a jednotek hodin přítomných na portu OUT_PORT určí začátek a konec zvukového vzorku v paměti, na kterou ukazuje. Adresa se od počáteční hodnoty do koncové hodnoty inkrementuje s rychlostí, kterou byl zvukový záznam vzorkován. Adresa se tedy mění s frekvencí 8 000 Hz. Každé ohlášení času se skládá z časového údaje a příslušného tvaru podstatného jména „hodin“. Proto se cyklus čítání adres zopakuje ještě jednou se změnou počátečních a koncových adres vzorků.

4.4.3 Paměť vzorků

V posledním odstavci kapitoly 4.4.2 bylo vysvětleno, jakým způsobem je paměť vzorků adresována. Kapitola 4.4.3 pojednává o tom, jak takovou paměť vytvořit a jak ji naplnit daty.

Paměť byla vytvořena v programu CORE Generator, který je součástí návrhového nástroje ISE. CORE Generátor nabízí spoustu předpřipravených funkcí jako jsou matematické funkce, sběrnice rozhraní PCI, FIR filtry, paměti, apod. Pro vytvoření paměti byla použita funkce Block Memory Generator 2.4, volba paměti typu Single Port ROM. Jde o paměť pouze pro čtení, a tak nabízí naplnění paměti podle inicializačního souboru. Soubor musí být ve správném tvaru podle dané šablony [9]. Soubor se vyplní podle šablony hodnotami vyseparovanými z uloženého zvukového souboru formátu WAV[7] (nekomprimovaný v PCM[8]).

[7] WAV – Waveform audio format
[8] PCM – Pulse Code Modulation – Pulzně kódová modulace

Vygenerovaná paměť má velikost 61 440 x 8 bitů. Při vzorkovací frekvenci 8000 vzorků za vteřinu lze do paměti uložit zvukový záznam o délce 7.68 s. Tato doba je pro účel demonstrační aplikace dostačující.

Výstup paměti je v zapojeném obvodu napojen přímo na vstup modulu audio výstupu, který byl popsán v kapitole 4.3. Jak již bylo řečeno, paměť je třeba adresovat s frekvencí stejnou, jakou byl zvukový signál navzorkován.

4.4.4 Paměť programu

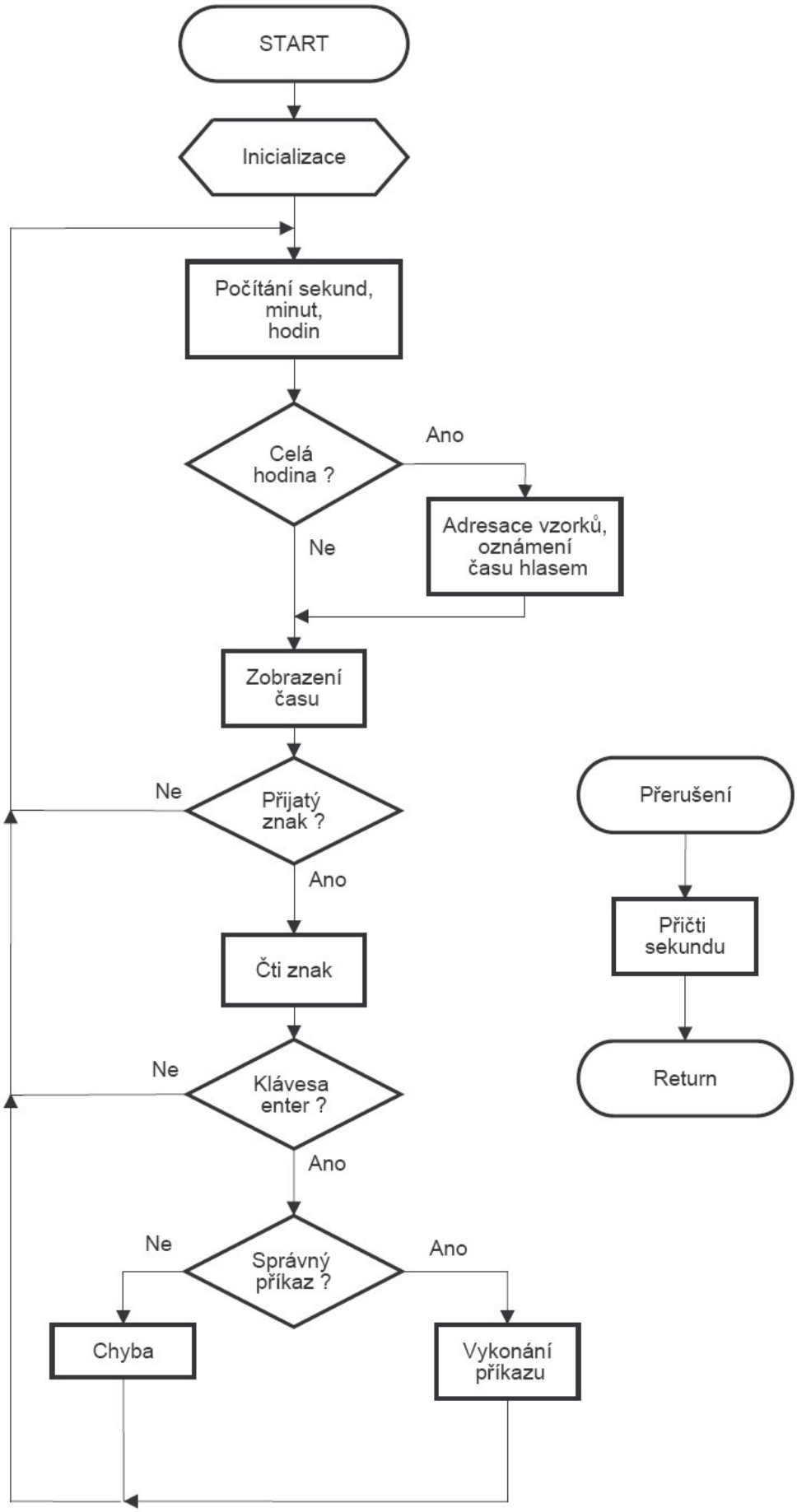
V paměti programu jsou uloženy instrukce, které určují chování a funkci mikrokontroléru PicoBlaze. Na obr. 4.5 je zjednodušený algoritmus programu.

Po resetu mikrokontroléru dojde k inicializaci - do 64 bytové paměti se uloží konstanty pro rozsvěcované segmenty daného čísla na 7-segmentovém displeji. Dále se po sériové lince UART přenese úvodní text.

Na začátku hlavní smyčky programu se provádí počítání sekund, minut a hodin. Pro vteřiny, minuty a hodiny jsou vyhrazeny registry zvlášť pro jednotky a desítky (tedy celkem 6 registrů). Je povoleno přerušení, které se opakovaně vyvolává každou vteřinu. Obsluha přerušení pouze inkrementuje hodnotu v registru pro jednotky vteřin. Po návratu z obsluhy přerušení se jen kontroluje, zda se nemá zvýšit hodnota registru vyššího řádu (desítky sekund →jednotky minut →….→desítky hodin). V případě, že došlo ke změně celé hodiny, přes výstupní porty se do dekodéru adres odešlou jednotky i desítky hodin. Další registr je vyhrazen pro zobrazování desítek sekund na LED. Každých 10 sekund se v tomto registru provede operace bitový posun o jednu pozici vlevo s vyplňováním jedniček zprava. Následně se hodnota registru odešle na adresu, kde jsou umístěny LED.

Pro zobrazení času na 7-segmentovém displeji se používá nepřímá adresace 64 bytové paměti registrem čísla, které chceme zobrazit. Na prvních 10 bytech této paměti jsou uloženy přímo hodnoty pro rozsvícení daného čísla, číslo odpovídá pozici (např. na adrese 0x03 jsou konstanty pro zobrazení čísla 3).

Dále se kontroluje, zda nejsou v bufferu přijímače sériové linky nějaká data. Jestliže je buffer prázdný, celý cyklus se opakuje. V případě, že jsou v bufferu připravena data, přečte se poslaný znak a zpátky se po sériové lince odešle (echo). Přečtené znaky se ukládají na



Obr. 4.5 Algoritmus programu mikrokontroléru PicoBlaze

posledních 16 pozic 64 bytové paměti. V případě příjmu znaku Backspace se provede výmaz znaku na obrazovce a při příjmu dalšího znaku se provede přepis znaku v paměti. Po obdržení znaku CR (carriage return) se testuje, zda znaky přijaté a uložené v paměti odpovídají známým příkazům. Je li příkaz neznámý či se špatnými argumenty, sériová linka vypíše chybu. V případě známých příkazů dojde k převodu argumentů z ASCII do dekadického tvaru. Podle typu příkazu se převedené číslo buď nahraje do příslušného registru (nastavování času) nebo pošle přímo na výstup mikrokontroléru (nastavení frekvence pwm modulátoru). Celý běh se dále opakuje ve smyčce.

5. ZÁVĚR

Cílem mé bakalářské práce bylo seznámit se s jazykem VHDL a technologií hradlových polí FPGA. K tomuto účelu mi byla zapůjčena vývojová deska Celoxica RC10, která je ideálním prostředkem pro experimentování v oblasti návrhu hardwarových aplikací.

Práce byla mojí první zkušeností s navrhováním programovatelných hradlových polí. Nejdříve jsem musel prostudovat základy programování v jazyce VHDL. Seznámil jsem se s vývojovým prostředím Xilinx ISE, ve kterém jsem se postupně snažil pochopit filozofii a způsob návrhu jednotlivých komponent.

Stěžejní část mé práce spočívala ve vytvoření hardwarové aplikace hodin s hlasovým výstupem pro obvod FPGA Spartan3 XC3S1500L. Pro správný návrh aplikace jsem musel nastudovat princip generování zvuku pomocí pulzní šířkové modulace PWM.

Při zpracování úkolů jsem dosáhl stanovených požadavků podle zadání bakalářské práce. V případě nároků na uložení většího množství zvukových dat by bylo možné například využít paměť FLASH, která však není přímo přístupná použitému FPGA.

Práce pro mne byla přínosem a pomohla mi proniknout do problematiky hardwarových návrhů.

LITERATURA A POUŽITÉ INTERNETOVÉ ZDROJE

[1] PINKER J., POUPA M., Číslicové systémy a jazyk VHDL, BEN – technická literatura, Praha 2006, 1. vydání.

[2] Open source hardware: Budoucnost programovatelných hradlových polí. ScienceWorld.cz [online] 05.11.2004 Dostupné na WWW: http://www.scienceworld.cz/sw.nsf/ID/614DFEDC5A6E25BDC1256F190048F776

[3] Nebojte se FPGA, HW server. Dostupné na WWW: http://hw.cz/Teorie-a-praxe/Dokumentace/ART365-Nebojte-se-FPGA.html

[4] HAZDRA P., Jazyk VHDL. Katedra mikroelektroniky FEL ČVUT. Dostupné na WWW: http://www.micro.feld.cvut.cz/home/hazdra/x/prednasky/Prednaska25.pdf

[5] Uživatelská příručka mikrokontroléru PicoBlaze . Dostupné na WWW: http://www.xilinx.com/bvdocs/userguides/ug129.pdf

[6] CHAPMAN K., UART Transmitter and Receiver Macros. Evergreen.edu. Dostupné na WWW: http://grace.evergreen.edu/dtoi/arch06w/asm/KCPSM3/Docs/UART_Manual.pdf

[7] KOLOUCH. M., Programovatelné logické obvody a hradlová pole. Dostupné na WWW: http://www.urel.feec.vutbr.cz/~kolouch/pld/

[8] Pulse-width modulation, Wikipedia.org. Dostupné na WWW: http://en.wikipedia.org/wiki/Pulse-width_modulation

[9] Field-programmable gate array, Wikipedia.org. Dostupné na WWW: http://en.wikipedia.org/wiki/Fpga

[9] COE file syntax, Xilinx.com. Dostupné na WWW: http://toolbox.xilinx.com/docsan/xilinx9/help/iseguide/mergedProjects/coregen/html/cgn_coe_file_syntax.htm

SEZNAM PŘÍLOH

Příloha 1 Statistické údaje

Device Utilization Summary

Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Flip Flops	352	26,624	1%
Number of 4 input LUTs	729	26,624	2%
Logic Distribution			
Number of occupied Slices	487	13,312	3%
Number of Slices containing only related logic	487	487	100%
Number of Slices containing unrelated logic	0	487	0%
Total Number of 4 input LUTs	905	26,624	3%
Number used as logic	729		
Number used as a route-thru	72		
Number used for Dual Port RAMs	16		
Number used for 32x1 RAMs	52		
Number used as Shift registers	36		
Number of bonded IOBs	30	221	13%
IOB Flip Flops	22		
Number of Block RAMs	31	32	96%
Number of MULT18X18s	1	32	3%
Number of GCLKs	1	8	12%
Total equivalent gate count for design	2,054,447		
Additional JTAG gate count for IOBs	1,44		

Příloha 2 **Obsah přiloženého CD disku**

Složka **/text/** - text bakalářské práce

- Libosvar_Jan_bp.pdf

Složka **/code/** - zdrojové kódy programu

- clock.psm (program pro KCPSM3)
- fpgaclock.vhd (hlavní entita)
- clock.vhd (Paměť ROM)
- pamet_vzorku.vhd (Paměť vzorků)
- pamet_vzorku.ngc
- genpwm.vhd (modul audio výstupu)
- kcpsm3.vhd (KCPSM3)
- kcuart_tx.vhd (vysílač sér. linky)
- uart_tx.vhd
- kcuart_rx.vhd (přijímač sér. linky)
- uart_rx.vhd
- bbfifo_16_8.vhd (buffer)
- fpgaclock.ucf (user constraint file)
- fpgaclock.bit (bitstream)