**První cvičení**

1. sestavte program, který bude na některém pinu procesoru blikat diodou s frekvencí asi 5Hz

až sem za 3-

1. „Udělejte z PICa drát“ , tedy naprogramujte AD převodník v procesoru. DA převodník na SPI, zvolte vhodný vzorkovací kmitočet. Předveďte, že obvod funguje. Dále předveďte aliasing.

až sem za 2

1. Příklad podle bodu 2 upravte tak, aby se choval jako derivační člen.

až sem za 1

**Druhé cvičení**

1. naprogramujte procesor tak, aby z DA převodníku vysílal sinusový signál o kmitočtu 250Hz. Vzorkovací kmitočet volte 8kHz. Sinusovku počítejte DSP algoritmem.
2. Příklad 1 upravte tak, aby na jednom výstupu převodníku vysílal 250Hz, na druhém 200Hz.
3. Sestavte program, který bude neustále vysílat číslici 2 pomocí tónové volby. Výsledné napětí porovnejte na osciloskopu se skutečným napětím z telefonu.
4. K programu podle bodu 3 připojte operační zesilovač v zapojení „emitorový sledovač“. Pomocí vysílané DTMF vytočte na ústředně ATEUS420 pobočku 22.

ad 1

DSP algoritmus známe, tak ho prostě implementujeme. Naprogramujeme nějaký čítač, který bude generovat 8kHz vzorkovací kmitočet, v přerušení uděláme vždy jeden krok a výsledek vyšleme na DA převodník. Náš algoritmus generuje výsledky v rozsahu < -1 ; 1> , takže budete muset hodnoty vynásobit vhodným číslem. Dále nezapomeňte, že čísla, která posíláte do DA převodníku, jsou pouze kladná, tedy že máme nulu na úrovni 2,048V . Při výpočtu použijte float , jako globální proměnné, jinak se všechny hodnoty zapomenou po retfie

ad 2

Necháme běžet dva algoritmy vedle sebe (tedy v přerušení budeme počítat dvě nezávislé sinusovky). Trochu problém bude s vysíláním dvou hodnot za sebou, protože konec vysílání jsme ošetřili v interruptu. Doporučuji uložit si druhé vysílaná číslo do dvou globální proměnné. První číslo odvysíláme zápisem do SPIBUF, nezapomeneme dát CS do 0. V přerušení pak vrátíme CS do 1, a pokud jsme vyslali jenom první číslo, chvíli počkáme, dáme CS do 0 a zápisem do SPIBUF odvysíláme druhé číslo. Pokud jsem už vyslali obě čísla, dáme CS do 1 a konec. Samozřejmě, potřebujeme vědět, zda vysíláme první číslo nebo druhé číslo. Pro to si uděláme nějakou globální proměnnou a pomocí čísla v ní rozhodneme, které číslo jsme vyslali.

Ad 3

Tady opět necháme běžet dva algoritmy vedle sebe, jejich výsledky sečteme. Musíme si ujasnit, jakých velikostí budou dosahovat naše čísla, a vhodně upravit jejich velikost. K získání signáůu telefonu použijeme ústřednu ATEUS420. TA má pro nás výhodu – jeden z ab drátů je uzeměn. TAkže si prostě připojíme telefon a dáme na něj druhý kanál osciloskopu.

Ad 4

Výstupní signál z převodníku **pravděpodobně** nedodá dostatečný proud. Proto za něj připojíme operační zesilovač v zapojení jako emitorový sledovač. Nezapomeneme oddělit pomocí elektrolytických kondenzátorů – **na ab drátech telefonu je v klidu 48V !** Ústřednu vyzvedneme pomocí telefonu.

**Třetí cvičení**

1. Sestavte program, který bude při stisknutí jednoho z desíti tlačítek vysílat tónovou volbou číslice 0 – 9 . Za převodník přidejte opět emitorový sledovač a na ústředně ATEUS420 vytočte libovolnou pobočku.
2. Sestavte program, který postupně obvolá všechny pobočky ústředny ATEUS420. Program sám vyzvedne linku pomocí tranzistoru.
3. Sestavte program, který postupně zavolá na mobilní telefony alespoň pěti vašich spolužáků. Pro připojení ke školní síti použijte ústřednu ATEUS420.

Ad 1

A tady je to také jednoduché – testujeme vstupní bity, při stisknutí bitu nastavíme správná čísla do algoritmů a algoritmus odstartujeme. Pokud zjistíme, že tlačítko už není stisknuté, algoritmus chcípneme a na výstupy vysíláme stabilně 2.048V (střed rozsahu)

Tady by se nám určitě hodilo pole s parametry DSP algoritmů pro jednotlivé číslice. Protože toto pole je neměnné, použijeme mechanismus PSV (čísla jsou uložena v paměti EEPROM) To se dělá podobně jako ve file <http://ozeas.sdb.cz/panska/mikroproc/33EV32/programovani/progr_a_vysv/zaklady/chvala.h>

čísla samozřejmě mohou být v jakékoli soustavě, pole může být int, float atd atdd., jak potřebujete

Za převodníkem máme samozřejmě operační zesilovač stejně jako v předchozím cvičení.

Ad 2

Pobočku ústředny budeme vyzvedávat pomocí tranzistoru s rezistorem v kolektoru (nebo drainu). Jeho bázi (nebo gate, podle vaší volby) pak budeme ovládat pomocí portu mikroprocesoru. Napětí a proudy při vyzvednutí 3A zná, 3D si to přečte v <http://ozeas.sdb.cz/panska/3A/DI/ukol_Goertzel/prehled_signalizace.doc>

kap. 4. Signalizace typu U , nebo si to prostě změříte na telefonu.

Postup bude tedy následující: vyzvednout, chvíli počkat, vytočit číslo pobočky, chvíli nechat zvonit, zavěsit. A tak dokolečka. Pobočky mají čísla 20 – 29 .

Ad 3

K připojení na školní síť použijeme ústřednu ATEUS420. Tím dosáhneme toho, že můžeme pracovat s nesymetrickými ab dráty (jeden drát uzemněn), a školní symetrická ústředna tím nebude trpět. Školní ústřednu připojíme na státní linky ústředny ATEUS. Musíme samozřejmě vytočit 0 – přístup na státní linky. Postup bude tedy následující: vyzvednout, chvíli počkat, vytočit 0 , chvíli počkat, vytočit číslo spolužáka, chvíli nechat zvonit, zavěsit. A tak dokolečka.

**Čtvrté cvičení – pro 3A**

**dallas DS18B20 nebo 18S20** , podle toho, co máte

1. Sestavte program, který „vydráždí“ obvod DALLAS k vygenerování PRESENCE pulzu. RESET i PRESENCE pulz sledujte na osciloskopu 4-
2. Program podle bodu 1 doplňte diodou, která bude indikovat přítomnost PRESENCE pulzu. 4
3. Sestavte program, který bude periodicky vysílat na obvod DALLAS příkaz READ ROM. NA osciloskopu pozorujte odpověď DALLASu 3
4. Program podle bodu 3 doplňte tak, aby vysílaný ROM code poslal do PC prostřednictvím UARTU a PICKIT2 – UART TOOL 2
5. Sestavte program, který bude měřit teplotu pomocí obvodu DALLAS. Výsledky bude posílat do PC pomocí UARTu. 1

ad 1

Bude nutno použít assembler, neboť časování je dosti náročné na rychlost a přesnost. O koexistenci assebbleru a C si přečtěte na <http://ozeas.sdb.cz/panska/mikroproc/33EV32/programovani/asm_progr_a_vysv/asm_a_C/>

během provádění podprogramu doporučuji vypnout přerušení a pak zase obnovit jeho původní stav, něco jako

push INTCON2

bclr INTCON2, #GIE

....................

................

pop INTCON2

ad 2

program prostě doplníme testováním bitu, na kterém je DALLAS v době, kdy se má objevit PRESENCE pulz. Podle jeho stavu pak rozsvítíme nebo zhasneme nějakou diodu. Můžeme také nechat vrátit hodnotu z assemblerovského podprogramu do Cčkového programu a zhasnutí a rozsvícení diody udělat tam.

ad 3

sestavíme podprogram, který bude na DALLAS vysílat osmibitové číslo. Opět nezapomeneme zakázat a povolit přerušení. Uděláme ho v asm, ale tak, aby byl zavolatelný v C. A můžeme ho volat v C

ad 4

dále si sestavíme podprogram, který bude číst 8 bitů z DALLASu. Inicilaizaci a ovládání UARTu si najdeme v <http://ozeas.sdb.cz/panska/mikroproc/33EV32/programovani/asm_progr_a_vysv/predvadeni/> - soubor pisnicka05.s Dále samozřejmě v <http://ozeas.sdb.cz/panska/mikroproc/33EV32/pdf/family/UART.pdf>

ad 5

A měření teploty je potom hračka

vyšleme SKIP ROM

dále CONVERT

musíme počkat, než se to změří

potom SKIP ROM

dále Read Scratchpad a čteme 16 bitů.

Tady samozřejmě můžeme přečíst celou pamě’t RAM, ale je nám to nanic, protože teplota je v prvních dvou buňkách. Nevadí, že čtení zastavíme někde uprostřed. Každá další akce musí začít RESET pulzem, a to znamená, že jakákoli předchozí nedokončená akce se tím ruší.

Můžeme to udělat i elegantněji obráceně:

přečteme teplotu

vyšleme CONVERT

počkáme

a tak stále dokolečka

První měření je pak samozřejmě nesmyslné, ale to nevadí.

**Páté cvičení – pro 3D**

1. Ke všem následujícím programům potřebujete „drát z PICa“. Protože jste si ho již měli udělat, nebudeme ho známkovat. Pokud není řečeno jinak, volte vzorkovací kmitočet 8kHz.
2. Sestavte program, který bude sloužit jako zpožďovací linka s maximálním možným zpožděním, tedy analogový signál ze vstupu dorazí na analogový výstup se zpožděním. Vzorkovací kmitočet volte 8kHz nebo 44 kHz.
3. Program podle bodu 2 upravte tak, že signál budete vysílat na oba kanály DA převodníku. První kanál bude mít maximální možné zpoždění, druhý poloviční.
4. Program podle bodu 2 doplňte třemi vstupními bity, kterými budeme řídit velikost zpoždění. Pokud je na vstupních bitech číslo 8 , je zpoždění maximální. Pokud je na vstupních bitech číslo 7, je zpoždění 7/8 maxima. Pokud je na vstupních bitech číslo 6, je zpoždění 6/8 maxima. Pokud je na vstupních bitech číslo N, je zpoždění N/8 maxima.
5. Sestavte program, který bude sloužit jako invertor – tedy signál na výstupu bude o 180 stupňů posunut oproti vstupnímu.
6. Sestavte program, který bude sloužit jako omezovač. Program kopíruje napětí ze vstupu na výstup. Pokud je vstupní napětí větší než 3,5V, je na výstupu 0V.
7. Sestavte program, který bude sloužit jako měnitelný atenuátor. Program kopíruje napětí ze vstupu na výstup. Procesor mý dále dva vstupní bity. Pokud je na vstupních bitech číslo 0, signál se nemění. Pokud je na výstupních bitech číslo 1, je výstupní signál o 6dB nižší. Pokud je na vstupních bitech číslo N , je výstupní signál o N\*6 dB nižší.
8. Naprogramujte procesor tak, aby se choval jako Shmittův klopný obvod s rozhodovacími úrovněmi úrovněmi 2V a 3V. Do obvodu přiveďte sinusový signál a na osciloskopu předveďte, že obvod funguje.
9. Sestavte program, který bude sloužit jako detektor špiček. Na vstup přivedeme libovolný analogový signál. Pokud je napětí vstupního signálu větší než 4V, rozsvítí se dioda na dobu 1 sec, a to i v případě, že špička je velmi krátká.

ad 2

Vzorky, které přečteme z AD převodníku, si uložíme do pole. Vysílat na SPI budeme vzorky z pole tak, abychom dosáhli maximálního možného zpoždění. Do datasheetu se podívejte na velikost paměti a podle toho si udělejte meze pole. Doporučuji program udělat tak, že nejdříve přečteme vzorek z pole a vyšleme na SPI, následně do stejného místa uložíme vzorek z AD převodníku. Pak inkrementujeme index pole. Samozřejmě, v prvním cyklu budeme vysílat nesmysly. To nevadí.

ad 3

vzorky pro druhý kanál čteme z vhodného místa v poli

ad 4

podle vstupního čísla měníme místo, ze kterého čteme výstupní vzorky z pole.

ad 9

naprogramujeme nějaký čítač, který bude v přerušení periodicky testovat data z AD převodníku. Pokud zjistíme větší napětí, zapneme jiný čítač, který vygeneruje požadovaný pulz pro rozsvícení diody. Přerušení od tohoto druhého čítače se zavolá jen jednou, pak se samo zakáže a čeká opět na povolení a puštění od algoritmu, který sleduje velikost vstupního napětí.