VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

AUTOMATIZOVANÉ PROGRAMOVÁNÍ VÍCE MIKROKONTROLÉRŮ AVR PŘES SPI SBĚRNICI

AUTOMATED FIRMWARE UPLOAD OF MANY AVR MICROCONTROLLERS OVER SPI BUS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

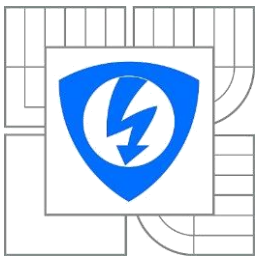
AUTOR PRÁCE — JIŘÍ BOŠTÍK

AUTHOR

VEDOUCÍ PRÁCE — Ing. PAVEL HANÁK, Ph.D.

SUPERVISOR

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student: Jiří Boštík ***ID:*** 145972
Ročník: 3 ***Akademický rok:*** 2013/2014

NÁZEV TÉMATU:

Automatizované programování více mikrokontrolérů AVR přes SPI sběrnici

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte obvod pro automatizované programování více mikrokontrolérů Atmel AVR přes SPI sběrnici za pomoci standardně dodávaných programátorů (AVR Dragon, JTAGICE MkII atd.). Pro jeho řízení využijte signálů, které jsou generovány přímo programátorem. Obvod musí správně fungovat i při řetězení operací, typicky chiperase - programování Flash - verifikace Flash - nastavení fuses - nastavení lock bitů. Po dokončení programovacího cyklu musí výstupy obvodu přejít do stavu vysoké impedance. Přitom se předpokládá, že všechny mikrokontroléry jsou napájeny současně a nelze je vypínat. Pro ovládání programátoru využijte dávkový soubor pro atprogram.exe, který je standardní součástí vývojového prostředí Atmel Studio 6. Výsledné řešení musí být snadno škálovatelné na větší i menší počty mikrokontrolérů; v práci uveďte schémata vámi navrženého obvodu pro programování 4, 12 a 40 kusů mikrokontrolérů. Navržené řešení pro 12 mikrokontrolérů otestujte například na kontaktním nepájivém poli.

DOPORUČENÁ LITERATURA:

[1] Bohumil Brtník, Číslicové systémy. BEN, 2011. ISBN 978-80-7300-4.
[2] Aplikační list Atmel AVR910: In-System Programming.
[3] Katalogový list Atmel ATtiny13A, kapitola 17.6. "Serial Programming".

Termín zadání: 10.2.2014 ***Termín odevzdání:*** 4.6.2014

Vedoucí práce: Ing. Pavel Hanák, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

Abstrakt

Cílem této práce je sestavit obvod pro automatizované programování více mikrokontrolérů Atmel AVR přes SPI sběrnici a navrhnout a realizovat demonstrační přípravek, který bude schopen tento úkol alespoň částečně automatizovat, tedy ideálně bez lidského zásahu jedním programátorem nahrát firmware do více mikrokontrolérů.

V teoretické části budou popsány součástky, které se v práci využívají, bude popsána jejich funkce a využitelnost. Pro přepínání mezi mikrokontroléry a tudíž i vyřešení dané problematiky, máme na výběr ze dvou možností. První možnost je sledování signálu „reset“, což je jednodušší varianta, nebo dekódování Atmel SPI instrukcí. Vzhledem k jednoduchosti a tedy i praktičnosti bude v práci používáno sledování signálu „reset“. Pro lepší pochopení bude v práci popsáno, jak celá problematika přepínání bude fungovat. Budou popsány jednotlivé kroky a pro lepší orientaci bude součástí i blokové schéma, které znázorní nejdůležitější části dané práce.

Praktická část se zaměřuje především na praktické vyzkoušení navrhnutého řešení. Abychom mohli sledovat signál „reset“ nejdříve musíme sladit asynchronní čítač se signálem „reset“ aby to správně reagovalo na sestupnou hranu. A dále přivedeme z asynchronního čítače BCD kód, pomocí kterého dekodér přepíná na jednotlivé mikrokontroléry pomocí spínacích tranzistorů, které se podle toho postupně programují. V práci bude používán programátor AVR Dragon, pro který bude využíváno dávkového souboru pro atprogram.exe, který je standardní součástí vývojového prostředí Atmel Studio 6. Součástí práce bude také alespoň částečné otestování navrženého problému na kontaktním nepájivém poli.

Tato práce by mohla být přínosná pro lidi, kteří častěji programují stejné mikrokontroléry se stejnými programy.

Klíčové pojmy:
mikrokontrolér Atmel AVR, SPI sběrnice, reset, programátor, BCD kód

Abstract

The aim of this work is to construct a circuit for automated programming more microcontrollers Atmel AVR through the SPI bus and to devise and execute demonstration plant that will be able to automate this task, at least partially, therefore, ideally without human intervention, one programmer to load the firmware into more microcontrollers.

The theoretical part describes the components that are used in the work, the description of their functionality and usability. For switching between microcontrollers we have a choice of two options. The first option is monitoring of the signal "reset", which is a simpler variant of SPI, or decoding the Atmel instructions. Thanks to the simplicity and practicality of the work will be used to monitoring of the signal of the "reset". For a better understanding in the work described how the whole problem will be work. They will describe each of the steps and for better orientation will be the block diagram too, which represents the most important part of the work.

The practical part focuses on the practical testing of the proposed solution. In order to monitor the signal of the "reset" first we have to coordinate asynchronous counter with signal "reset" it to properly reply to a downward edge. Next we bring from the asynchronous counter BCD code with which the decoder switches to each of the microcontrollers using the switching transistors and then they are gradually programmed. Work will be used the programmer AVR Dragon, for which it will be used by the batch file for atprogram.exe, which is a standard part of Atmel Studio 6. In the work will be at least a partial test of the proposed problem to contact layer the field.

Key words:

microcontrollers Atmel AVR, SPI bus, reset, programming, BCD code

BOŠTÍK, J. Automatizované programování více mikrokontrolérů AVR přes SPI sběrnici. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2014. 44 s. Vedoucí bakalářské práce Ing. Pavel Hanák, Ph.D..

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma automatizované programování více mikrokontrolérů Atmel AVR přes SPI sběrnici jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne 4.6.2014 ……

podpis autora

Poděkování

Děkuji vedoucímu Ing. Pavlu Hanákovi, Ph.D. za velmi užitečnou metodickou pomoc a cenné rady při zpracování bakalářské práce.

V Brně dne 4.6.2014 ……

podpis autora

Výzkum popsaný v této bakalářské práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

V Brně dne 4.6.2014 ……

podpis autora

Obsah

Seznam obrázků

Úvod

Cílem práce je sestavit obvod pro automatizované programování více mikrokontrolérů Atmel AVR přes SPI sběrnici za pomoci standardně dodávaných programátorů.

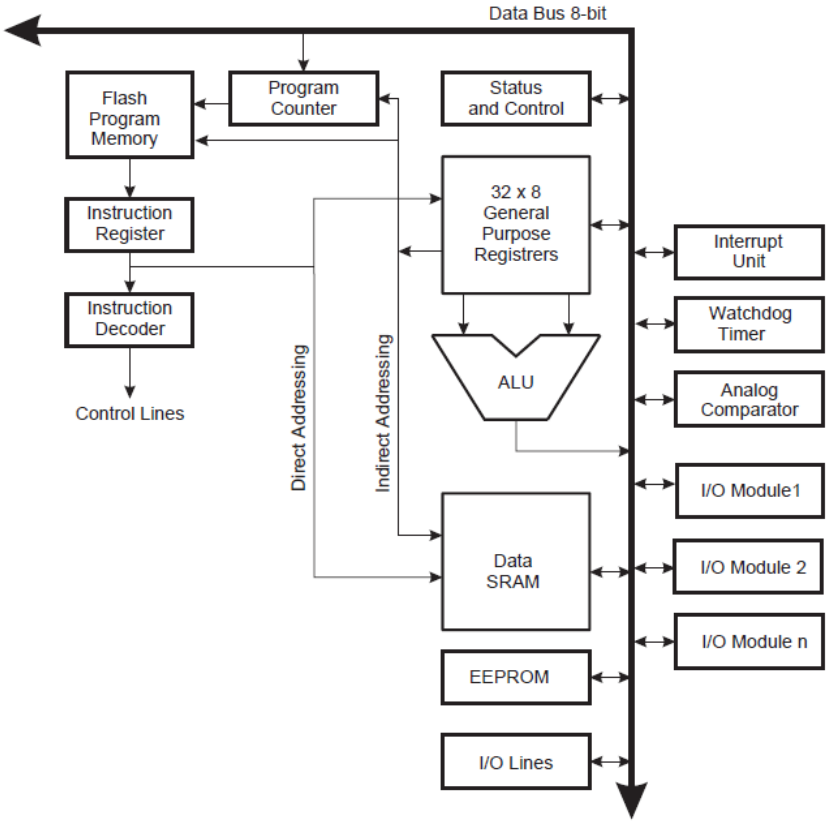
Semestrální práce se skládá ze dvou částí. V první části budou popsány součástky, jak fungují a jak jsou v práci využité. Poté bude vybrána jedna ze dvou možností, pro přepínání mezi mikrokontroléry. Na výběr máme sledování signálu „reset“ nebo dekódování Atmel SPI instrukcí. Dále bude popsáno celkové fungování problematiky přepínání. Budou popsány jednotlivé kroky a funkce součástek, které se využívají. Součástí bude i blokové schéma, které znázorní nejdůležitější části práce.

V druhé části se prakticky vyzkouší, zda navrhnuté řešení funguje a případně se navrhne jiné řešení, které by pomohlo vyřešit automatizované programování více mikrokontrolérů. Pro ovládání programátoru bude využit dávkový soubor pro atprogram.exe, který je standardní součástí vývojového prostředí Atmel Studio 6. Součástí práce bude také alespoň částečné otestování navrženého problému na kontaktním nepájivém poli.

1 Teoretický popis použitých obvodů, zařízení a software

1.1 Mikrokontroléry

Mikrokontroléry se dnes používají v nejrůznějších avšak zcela běžných věcech. Používají se pro řízení jednotlivých součástí přístrojů nebo třeba v elektronických zařízeních. Zcela určitě je lze najít například v autech, pro řízení bezpečnostních prvků, v mobilních telefonech, tiskárnách. Prostě je najdeme všude, kde se nachází nějaká elektronika. Nejvýznamnějšími faktory takto velkého využití mikrokontrolérů je zcela jistě jejich nízká cena a také spotřeba (ta se pohybuje v klidovém režimu v řádu jednotek mikroampér, za chodu pak může dosahovat minimální spotřeby přibližně 500 mikroampér). Další výhodou jsou malé rozměry těchto procesorů a poměrně velké množství periferií (převodníky A/D a D/A, sériové i paralelní porty i další). Mikrokontroléry jsou vyráběny v mnoha variantách a díky tomu může návrhář vybrat přesně obvod, který právě potřebuje a nemusí platit to, co by zůstalo nevyužito. Mikrokontroléry patřící do rodiny AVR, které využívají architekturu RISC (redukovaná instrukční sada) jsou povedeným výstupem architektury mikrokontrolérů přizpůsobených jazyku C. Architektura RISC je typická zpracováním instrukcí v průběhu jednoho hodinového cyklu (Mikrokontroléry PIC, 2012).



Obr. 1 - AVR architektura

1.1.1 ATtiny13A

Jedná se o mikrokontrolér z produkce firmy Atmel, který je založen na Harwardské architektuře s procesorem RISC. Je jedním z nejjednodušších mikrokontrolérů, které firma vyrábí.

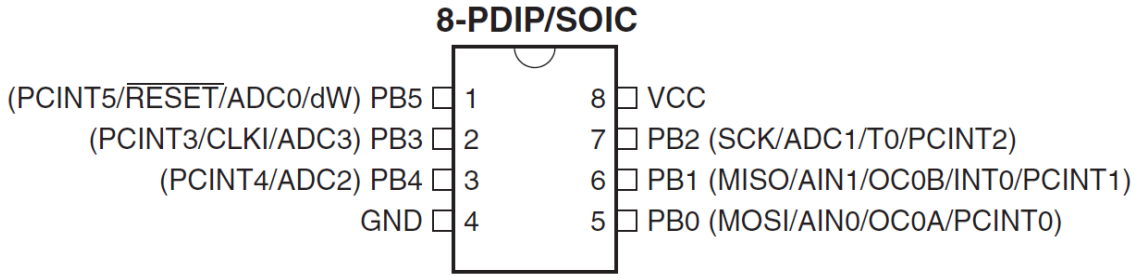
Je vybaven programovatelnou pamětí flash, datovou pamětí EEPROM a pamětí RAM pro dočasné ukládání dat. Má osmibitovou architekturu s instrukční sadou o velikosti 120 instrukcí. Dále je vybaven analogovým komparátorem (porovnávání s interní referencí 1,1 V). Obsahuje osmibitový čítač/časovač a čtyř kanálový desetibitový převodník A/D (+-2 LSB). Frekvence interního osciloskopu je 9,6 MHz resp. 4,8 MHz. Lze snadno programovat zápisem do paměti flash například pomocí sběrnice SPI a programátorů které firma Atmel za tímto účelem vyrábí (Duch, Miroslav, 2009), (Atmel, 2010). Obrázek je ze zdroje www.atmel.com

Piny:

VCC:	Napájecí napětí
GND:	Připojení na zem
Port B (PB5:PB0):	Obousměrné I/O porty s vnitřními odpory
RESET:	Slouží k restartu obvodu (stanovená délka pulsu)

Parametry:

Napájení:	2,7 V – 5,5 V
Typ pouzdra:	DIL 8
Taktovací kmitočet:	20 MHz
I/O piny:	6
Paměť RAM:	64 B
Paměť EEPROM:	64 B
Paměť flash:	1 kB

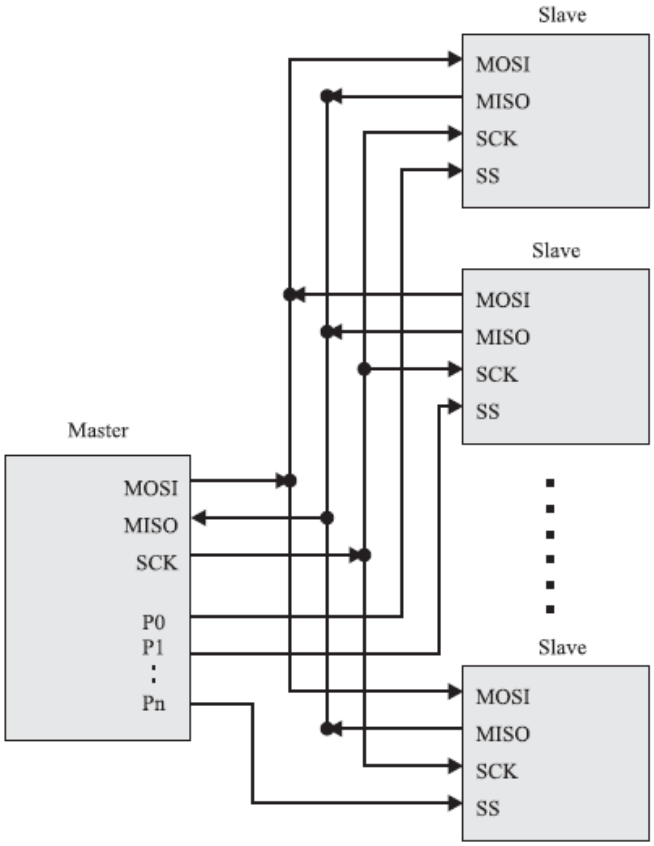


Obr. 2 - ATtiny13A

1.2 SPI sběrnice

Sériové periferní rozhraní SPI (Serial Peripheral Interface) se využívá pro komunikaci řídicích mikroprocesorů s integrovanými obvody (paměti, displeje, převodníky atd.). U některých mikrokontrolérů (AVR) je využívána, pro naprogramování paměti flash.

Komunikace je založena na společné sběrnici. Zařízení, které mezi sebou komunikují, jsou Master (řídicí) a slave (podřízený), přičemž master je vždy pouze jeden, ale zařízeních typu slave může být více (komunikace ovšem probíhá pouze s jedním, ne s několika najednou). Master řídí komunikaci pomocí hodinového signálu a slave podle tohoto signálu vysílá data. Velikost napětí v jednotlivých log. úrovních signálu v rozhraní SPI jsou dané použitou technologií. Rozsah frekvence hodinového signálu jsou jednotky až desítky MHz (Dudáček, K., 2002).



Obr. 3 - SPI sběrnice

Na obrázku 3 vidíme propojení jedno zařízení typu master s několika obvody typu slave. MOSI (Master Out, Slave In) je datový výstup obvodu Master připojený ke všem vstupů MOSI obvodů slave. Přesně naopak, už podle názvu, je MISO (Master In, Slave Out) datový vstup obvodu Master propojený s výstupy MISO všech obvodů slave. Hodinový signál SCK je výstupem master a u obvodů slave vstupem. U každého obvodu slave vidíme vstup SS (Slave Select) který slouží pro výběr

právě jednoho z obvodů slave. Jestliže je SS v úrovni log. 0, dané rozhraní SPI tohoto obvodu je neaktivní. Vstupy SS jsou propojeny jednotlivými vodiči s obvodem master. Když je mastrem mikrokontrolér, obvykle se SS připojují k portům, pro jednodušší výběr, se kterým obvodem bude navázána komunikace.

1.2 Atmel Studio 6.1

Atmel Studio 6.1 je vývojová platforma pro vývoj a ladění aplikací založených na mikrokontrolérech Atmel (MCU). Prostředí je vytvořeno pro snadné psaní a ladění aplikací napsaných v C / C + + nebo assembleru. Podporuje všechny 8 a 32 bitové AVR MCU, nové SoC, SAM3, SAM4 a SAM D20 MCU. Umožňují snadné propojení s Atmel ladícími a vývojovými deskami (Atmel, 2014).

1.2.1 ATprogram.exe

Atprogram.exe je program spustitelný v příkazovém řádku, pomocí něhož lze programovat mikrokontroléry nahráním, našeho již vytvořeného programu v textové podobě s příponou hex (elf), do vybraného mikrokontroléru. Vytvořením jednoduchého dávkového souboru můžeme programovat automaticky. Tato varianta programování zvládne všechny základní požadavky práce s mikrokontroléry (nahrát program, mazání paměti, verifikace atd), (Atmel,2014).

Příklad příkazu pro mazání paměti:

```
<cesta k atprogram.exe> -t <tool> -i <interface> -d
<device> chiperase
```

1.2.2 Dávkový soubor

Dávkové soubory, také známe jako dávkové programy popřípadě skripty, usnadňují rutinní a často se opakující práci. Jsou to neformátované textové soubory, obsahující jeden nebo více příkazů. Za názvem má příponu BAT nebo CMD.

Funguje tak, že napíšeme název souboru do příkazového řádku, následně se začnou provádět všechny příkazy v pořadí, jak jdou v souboru po sobě. Je možné zadávat všechny známé příkazy, například pomocí for, goto a if, můžeme vytvořit podmíněné zpracování příkazů, uvedených v dávkovém souboru. Dávkové soubory jsou užitečné například při spouštění několika po sobě jdoucích příkazu, které se opakují (Windows Server, 2014).

1.3 AVR Dragon

AVR Dragon je programátor vyráběný firmou Atmel. Používá se jako ladící a vývojový prostředek vytvořený pro práci s mikrokontroléry řady ATtiny a ATmega. Oproti ostatním programátorům je jednodušší, ale také neporovnatelně levnější a tím pádem všem dostupný. Podporuje všechny programovací módy rodiny mikrokontrolérů AVR a podporuje emulaci programu se zařízeními o velikosti flash paměti 32 kB a méně. Jsou na výběr dvě možnosti napájení, první je přímo z USB rozhraní (přes programátor) nebo externí zdroj napětí. AVR Dragon je kompatibilní s vývojovým prostředím Atmel Studio 6.1 popřípadě i se staršími nebo novějšími verzemi (mcu.cz, 2009), (AVR Dragon).
Obrázek je použit ze stránek www.atmel.com

Několik způsobů ladění a programování mikrokontroléru:

ISP ((In system Programming):	3 vodičové programovací rozhraní
JTAG (Joint Test Actoin Group):	4 vodičové programovací a ladící rozhraní (ladění pouze pro Flash)
PP (Paralel Programming):	paralelní programování nabízející vysokou rychlost
HVSP (High Voltage Serial Prog.):	seriová verze paralelního programování
dW (debugWire):	jednovodičové ladící rozhraní

Podporované mikrokontorléry:
ATmega16, ATmega168, ATmega169, ATmega32, ATmega3250P, ATmega325P, ATmega3290P, ATmega329P, ATmega48, ATmega88
ATtiny13, ATtiny2313, ATtiny25, ATtiny45 a ATtiny85.

Obr. 4 - AVR Dragon

1.4 Logické členy

Základním prvkem logických obvodů je logický člen (hradlo), ten má za úkol vyčíslovat logickou funkci. Nejčastěji se potkáme s hradly, která mají jeden nebo více vstupů a pouze jediný výstup. Hodnota na výstupu log. členu je funkcí vstupních hodnot.

$$y = f(x_1, x_2, \dots, x_n)$$

Někdy rozlišujeme v terminologii logický člen a hradlo. Hradlo je označení pro fyzickou součástku (např. NAND a jiné integrované obvody). Logický člen má pak význam jako prvek, který realizuje logickou funkci. Mezi základní logické členy (i hradla) patří AND, OR a NOT, pomocí kterých lze sestavit libovolný logický obvod. Při spojení těchto základních typů vznikne NAND a NOR, kterými je možno implementovat všechny číslicové systémy (Mikrokontroléry PIC, 2012).

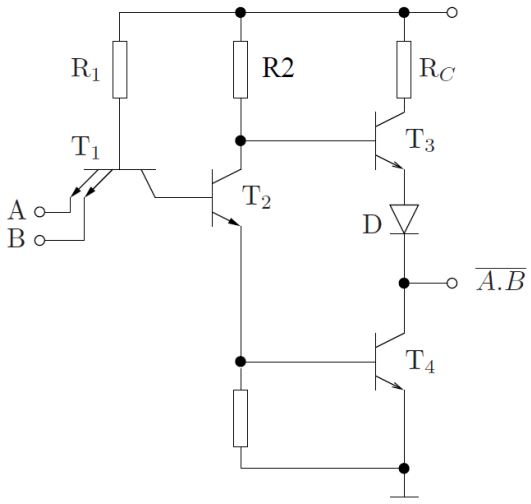
1.4.1 NAND

Hradlo NAND vytváří logickou funkci negovaného logického součinu zvanou Shefferova funkce. Hradlo AND je skoro stejné jako hradlo NAND, jediný rozdíl je že NAND má ještě navíc negovaný výstup. Tyto hradla mohou mít dva a více vstupů ale pouze jediný výstup. Z pravdivostní tabulky vidíme, když je alespoň na jednom vstupu log. 0, tak na výstupu bude vždy log . 1. Pomocí hradla NAND je možné realizovat libovolnou logickou funkci. V praxi to znamená že, nepotřebujeme mnoho různých integrovaných obvodů s různými hradly. NAND je jedním z nejvíce využívaných hradel, protože se dá snadno realizovat na čipu v závislosti na použité technologii (Mikrokontroléry PIC, 2012).

Log. funkce (dva vstupy): $Y = \overline{A \cdot B}$

Tabulka 1- Pravdivostní tabulka

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0



Obr. 5 - Schéma NAND

1.5 Klopné obvody

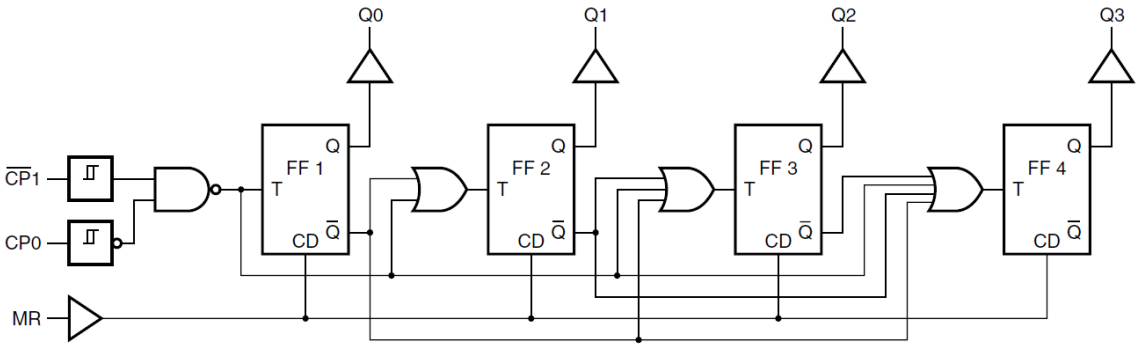
Klopné obvody (označované KO) jsou elektrické obvody, které na svých výstupech mohou mít pouze dva stavy, ke změně mezi nimi dochází skokově. Obvody se skládají především z hradel a jiných aktivních prvků. Je možné je použít jako impulzní generátory, paměťové prvky, časovače atd. (Brtník, Bohumil, 2011).

Rozdělují se do několika základních skupin:

Astabilní (AKO):	Vyznačují se tím, že nemají žádný stálý stav, kmitají z jednoho stavu do druhého
Monostabilní (MKO):	Mají jeden stabilní stav, tento stav se může změnit s příchodem spouštěcího impulzu a tím se překlopí do stavu opačného.
Bistabilní (BKO):	Má oba stavy stálé, slouží jako paměti.
Schmittův:	Obvod je používán k úpravě signálu (tvaru).

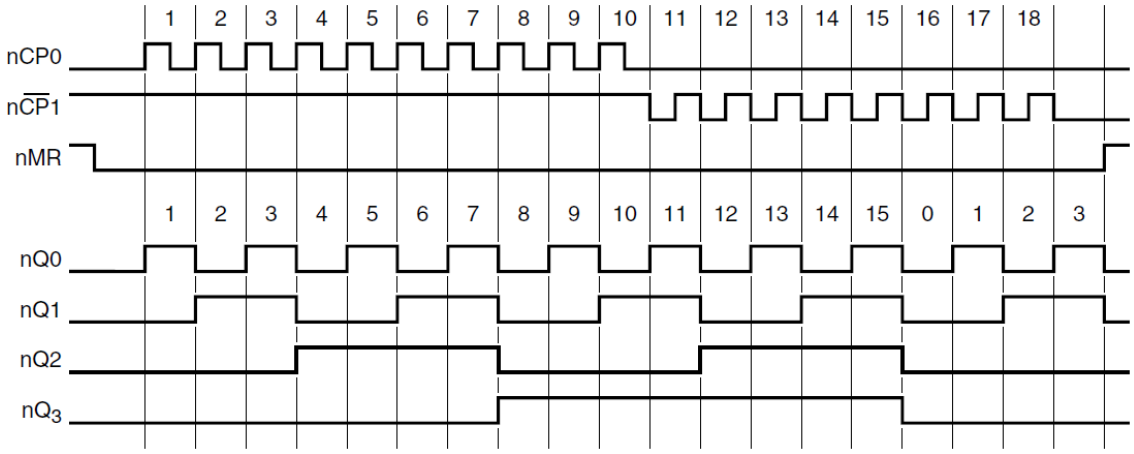
1.5.1 Asynchronní čítač

Asynchronní čítač je sestaven ze čtyř klopných obvodů. Čítač je čtyř bitový, binární (počítá ve dvojkové soustavě) a čítá nahoru. Vstupy CP1 a CP0 jsou hodinové vstupy, Q jsou výstupy obvodů a vstup MR je určen pro restartování celého obvodu. Výstupy jsou označeny Q0, Q1, Q2 a Q3, přičemž nejvyšší váhu má Q3 a nejnižší Q0. Výstupem tohoto asynchronního čítače je BCD kód na již zmiňovaných čtyřech výstupech (BCD kód i s převodem na decimální soustavu je v tabulce 2). Jednotlivé obvody jsou mezi sebou propojeny tak, že výstup Q je přiveden na vstup hradla a potom na vstup T následujícího obvodu. Obvod má vstup CP0 reagující na náběžnou hranu a vstup CP1 reagující na sestupnou hranu pulsu. Je zřejmé, že řídícím signálem je vstup CP0 (respektive CP1) prvního klopného obvodu a podle něho se mění výstupy ostatních klopných obvodů. (Čítače, 2012), (Philips, 2009). Zapojení obvodu vidíme na obrázku 6.



Obr. 6 - Asynchronní čítač

Na obrázku 7 vidíme, jaké jsou logické úrovně na výstupech jednotlivých klopných obvodů v závislosti na vstupním signálu. Vstupní signál je označen C (horní). Vidíme, že klopný obvod reaguje na sestupnou hranu. Jakmile zaregistruje na vstupu C sestupnou hranu, tak se výstupní logická hodnota neguje. Doba držení logické úrovně jednoho výstupu je vždy poloviční oproti výstupu následujícímu.



Obr. 7 - Závislost výstupního signálu na vstupním

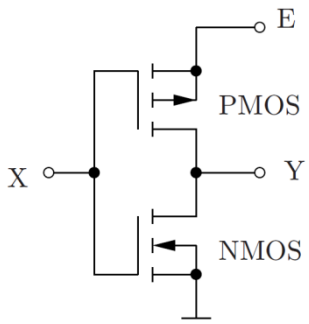
Počáteční předpoklad je, že všechny klopné obvody jsou vynulovány. Očíslujeme-li vstupní impulsy od 1 do 16, dostaneme všechny kombinace log. úrovní na jednotlivých výstupech. Každý výstup představuje jeden bit čísla ve dvojkové soustavě. Přidáním dalšího čítače lze zvýšit počet výstupů a získat libovolný n-bitový čítač, jaký potřebujeme. Jakmile čítač dosáhne maximální hodnoty (v tomto případě 16), pokračuje opět stejně od začátku s log. úrovní 0 na všech výstupech.

Tabulka 2 - Závislost výstupních úrovní na počtu impulsů

CP1	Q0	Q1	Q2	Q3
1	0	0	0	0
2	1	0	0	0
3	0	1	0	0
4	1	1	0	0
5	0	0	1	0
6	1	0	1	0
7	0	1	1	0
8	1	1	1	0
9	0	0	0	1
10	1	0	0	1
11	0	1	0	1
12	1	1	0	1
13	0	0	1	1
14	1	0	1	1
15	0	1	1	1
16	1	1	1	1

1.6 Technologie CMOS

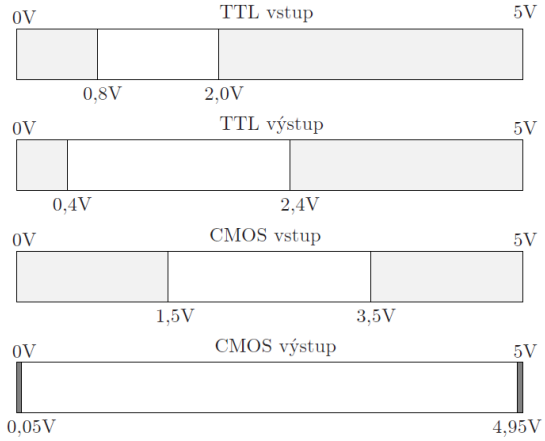
Technologie CMOS (Complementary Metal-Oxide-Semiconductor) je v současné době hojně využívaná v integrovaných obvodech. Především je využita na výrobu mikroprocesorů, pamětí typu SRAM a také v neposlední řadě na obrazové senzory. Technologie vychází z použití tranzistoru NMOS i PMOS. Tyto tranzistory se střídají ve funkci spínacího tranzistoru a zatěžovacího rezistoru MOS (Olivka, Petr, 2010).



Obr. 8 – Zapojení invertoru

Invertor funguje velice jednoduchým způsobem. Jestliže je na vstupu X logická 1, tak je tranzistor NMOS zapnutý a PMOS vypnutý. Na výstupu Y je logická 0, protože je výstup přiveden na zem. Když je na vstupu X log. 0, zapnutý je PMOS a vypnutý NMOS. Na výstupu je log. 1, protože vystup Y je připojen na napájecí napětí.

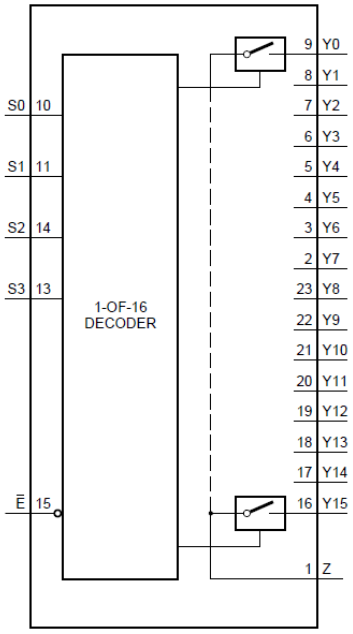
Z tohoto příkladu vidíme, že tranzistory fungují jako spínače a jestliže nezatěžujeme výstup takového obvodu, tak se spotřeba v klidovém stavu blíží nule. Napájení obvodů CMOS je v rozmezí od 3 do 16 V. Na obrázku 9 vidíme porovnání technologie CMOS a TTL z pohledu napěťových úrovní (na vstupech i výstupech). Díky velmi malému příkon, dobré šumové odolnosti, možnosti propojení s TTL a širokému rozsahu napájecího napětí se technologie CMOS stala nejžádanější technologií na trhu.



Obr. 9 - Srovnání technologií z pohledu napěťových úrovní

1.7 Multiplexer/demultiplexer

Jedná se o 16-ti kanálový vysokorychlostní multiplexer/demultiplexer, všechny piny fungují obousměrně (vstupní/výstupní). Vstup je propojen s výstupem přes tranzistor CMOS. Tento obvod lze použít nejen pro analogový, ale také pro digitální signál s vysokou frekvencí v řádu MHz. Rozsah napětí je 0 V až 10 V a v tomto rozmezí napájení se lišší i velikost odporu v sepnutém stavu. Při nejpoužívanějším napětí 5 V je velikost odporu 80 Ω a se zvyšováním napětí se snižuje (9 V – odpor 60 Ω). Maximální proud, který může obvodem procházet, je 50 mA. Obvod se vyrábí v několika provedeních: DIP24, SO24, SSOP24 a TSSOP24 (typ pouzdra). Na obrázku 10 vidíme funkční model (nxp.com, 2014), (Philips, 2009).



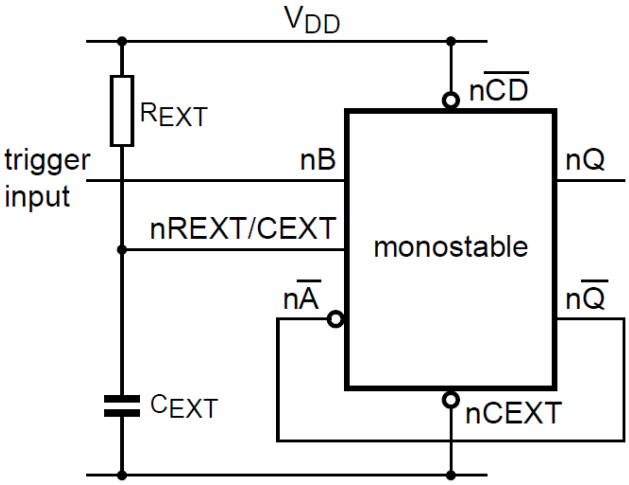
Obr. 10 - Funkční model

Na tomto modelu vidíme piny označené Y0 až Y15, ty slouží jako vstupy/výstupy. Další označení S0 až S3 pro adresní vstupy a Z je označen jeden pin, který slouží jako vstup/výstup. Poněkud odlišnou funkci má pin označený jako Ē, pokud je na pin přivedena log. 1 přejdou všechny vstupy/výstupy do stavu vysoké impedance (spojení jsou přerušena). Dekodér pomocí adresních vstupů vybírá jeden ze 16-ti pinů, který se (uvnitř obvodu) propojí s pinem Z. V okamžiku kdy se propojí, může začít procházet signál, přičemž jeden z těchto pinů bude vstupní a druhý výstupní (rozhodující je, jestli chceme obvod používat jako multiplexer nebo demultiplexer). V tabulce 2 jsou znázorněny logické úrovně jednotlivých adresních vstupů S0, S1, S2 a S3 a jim odpovídající pin Y_n spojený s pinem Z.

Tabulka je stejná jako v případě asynchronního čítače, pouze místo Q0 si představíme S0 a hodnota 1 ve sloupci C je naše Y0.

1.8 Monostabilní multivibrátor

Monostabilní multivibrátor je sekvenční obvody generující signál s nastavitelnou šířkou pulsu mající jeden stabilní stav. Puls je vygenerován na výstupu, jestliže je na vstup přivedena náběžná nebo sestupná hrana (podle zapojení obvodu). Na obrázku 11 vidíme zapojení reagující na náběžnou hranu. Šířka pulsu je dána RC článkem, který tvoří odpor R_{EXT} a kondenzátor C_{EXT}. RC článek je připojen na pin REXT/CEXT (resistor connection / external capacitor) a uzemněn společně s pinem CEXT (external capacitor connection). Vstup $\overline{CD}$ (clear direct input) je připojen na napájecí napětí, vstup je na pinu B a výstup je jako obvykle pin Q (Philips, 2009).



Obr. 11 - Monostabilní multivibrátor

2 Návrh možného řešení

Hlavním problémem této práce, kterým se budeme především zabývat, je najít řešení pro přepínání mezi jednotlivými mikrokontroléry. Při komunikaci mikrokontroléru a programátoru AVR Dragon je možnost sledovat signál reset či kód, který je přenášen mezi programátorem a mikrokontrolérem a pomocí nich realizovat přepínání mezi mikrokontroléry.

V této práci použijeme pro řešení problému sledování signálu reset, který je pro daný problém jednodušší variantou. Signál reset nabývá dvou logických úrovní, které jsou: logická 0 (0 V) a logická 1 (5 V - napájecí napětí). Při stavu kdy do mikrokontoréru nenahráváme program, nemažeme paměť a neprováníme další úkony, je signál reset v úrovni 5 V. Jestliže však začneme nahrávat program nebo provádět jiné operace, signál na dobu provádění těchto operací přeskočí do úrovně 0 V (logická 0). Díky tomuto jevu vznikne při začátku komunikace sestupná hrana (přechod z log. 1 do log 0) a při skončení hrana náběžná (přechod z log. 0 do log. 1) (obrázek 13). Jelikož potřebujeme program nejprve nahrát do mikrokontroléru a až poté přepnout na další mikrokontrolér, nabízí se možnost využít náběžné hrany jako vhodného okamžiku k přepnutí na další mikrokontrolér. Na této myšlence je založen hardware pro přepínání, ale je také zapotřebí, aby se v počítači opakoval příkaz, pomocí něhož se nahraje program (napsaný například v Atmel Studio 6.1) přes programátor do mikrokontroléru. Řešením je program dodávaný spolu s Atmel Studio 6.1, který když spustíme napsáním správného příkazu do příkazového řádku, tak nahraje program, vymaže paměť nebo provede jiné operace na mikrokontroléru, podle toho, co do příkazového řádku napíšeme. Aby nedocházelo k tomu, že pokaždé se musí příkaz znovu zadat, použijeme takzvaný dávkový soubor a do něho napíšeme příkaz, který potřebujeme a ještě využijeme některý z příkazů pro opakování (díky tomu dosáhneme například 12x zopakovaný příkaz pro nahrání programu do paměti). Tímto řešením, pomocí dávkového souboru, můžeme naprogramovat libovolný počet mikrokontrolérů, spuštěním pouze jediného souboru.

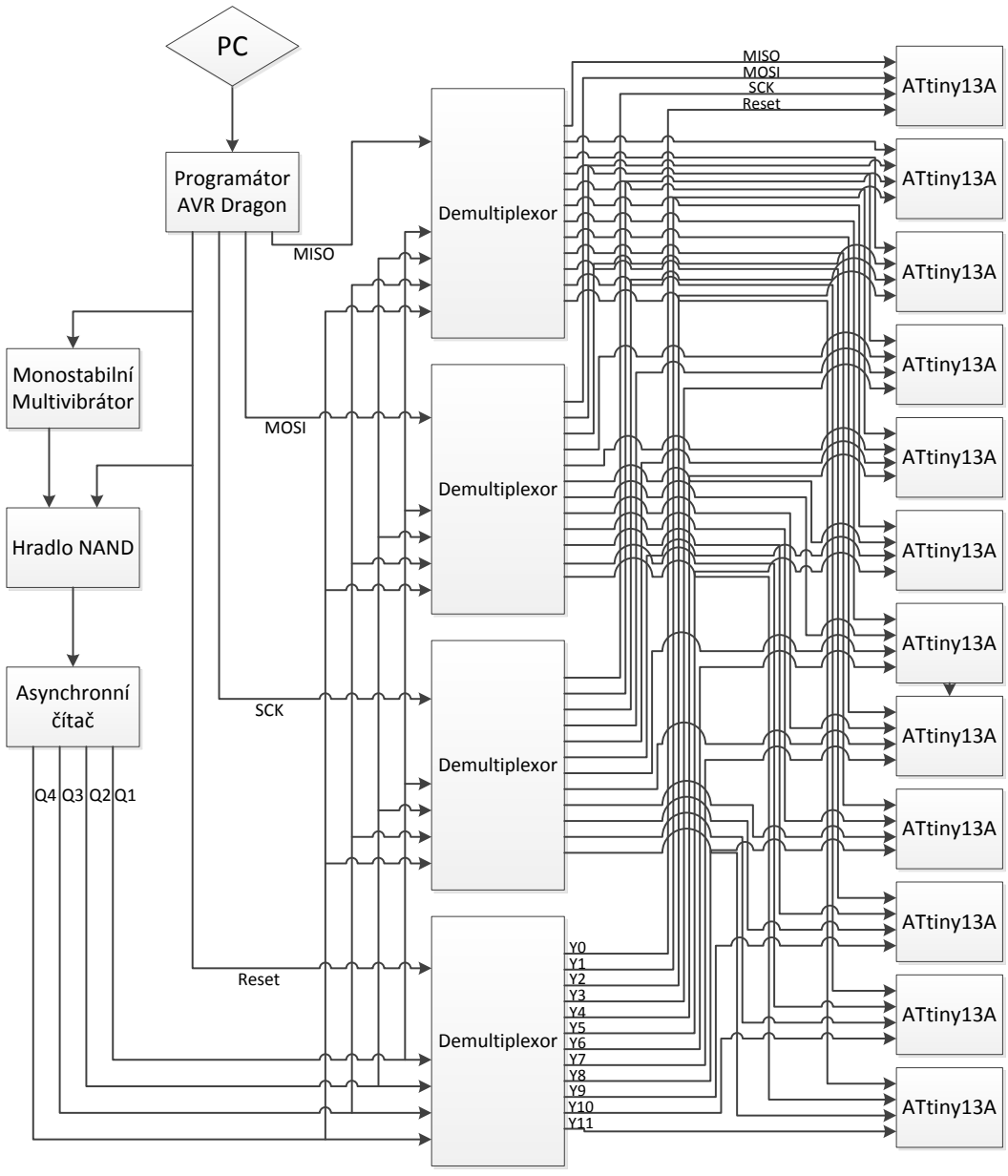
Takto vymyšlené a popsané řešení se zdá být snadné, ale je zde samozřejmě jeden poměrně velký problém, kterým je řetězení prováděných operací. Tím je myšleno, že se neprovádí pouze jedena operace (například pouze nahrávání programu do paměti), ale těchto operací je více. Typicky se jedná o řadu operací: chiperase (mazání paměti) - programování Flash - verifikace Flash - nastavení fuses - nastavení lock bitů. Všechny tyto operace se provedou nejprve na prvním mikrokontroléru a až po dokončení poslední operace se přepneme obvod na další mikrokontrolér. Problém je ten, že mezi prováděním jednotlivých operací nezůstává signál reset po celou dobu v logické 0, ale přeskakuje mezi nimi do logické 1 (to je vidět na obrázku 14). Jestliže by jsme při každé náběžné hraně přepínali na další mikrokontrolér operace by neproběhli všechny, ale pouze první z nich a hned po jejím dokončení by se při náběžné hraně přepínalo na další mikrokontrolér. Tento problém je možné vyřešit v dávkovém souboru nastavením delšího času pauzy (času setrvání v úrovni logické 1) mezi provedením jednoho

cyklu operací a cyklu následujícího. Tím by jsme měli větší čas trvání logické 1 mezi cykly operací, než čas trvání logické úrovně 1 mezi jednotlivými řetězícími se příkazy. Potom by bylo možné, pomocí těchto různě dlouhých časů zjistit, zda se má nebo nemá přepínat na další mikrokontrolér.

Porovnání těchto různě dlouhých časových úseků, je řešeno pomocí číslicových obvodů v praktické části práce. Takto vymyšleným obvodem docílíme přepínání mezi libovolným počtem mikrokontrolérů.

3 Praktické řešení

V teoretickém řešení byla popsána metoda, jak vyřešit automatické přepínání mezi mikrokontroléry. V praktické části je popsán celý návrh i s konkrétními součástkami, které jsou v obvodu využity. Je zde také popsáno několik problémů objevených, až během zapojení jednotlivých částí a následném proměřování obvodu. Tyto problémy budou ovšem zmíněny až na konci kapitoly.



Obr. 12 - Blokové schéma

Na obrázku 12 je blokové schéma celého obvodu. Jsou na něm vyznačeny a pojmenovány všechny použité integrované obvody. Dále vidíme všechny cesty, jak jsou obvody mezi sebou propojeny i s šipkami vyznačující směr. Obvody jsou seřazeny postupně tak, jak jimi prochází signál.

Jako první je znázorněn PC (počítač) z něhož řídíme počet opakování zadaných operací (počet opakování je stejný jako počet mikrokontrolérů), jaké operace budou provedeny na mikrokontrolérech i jiné věci. Následuje programátor, který je takovým prostředníkem mezi počítačem a mikrokontroléry, pomocí něho provádíme všechny operace na mikrokontrolérech. Další v řadě po sobě jdoucích zařízení jsou různé integrované obvody. Tyto integrované obvody budou popsány podrobněji, jak je vidíme v bokovém schématu postupně za sebou, směrem s procházejícím signálem.

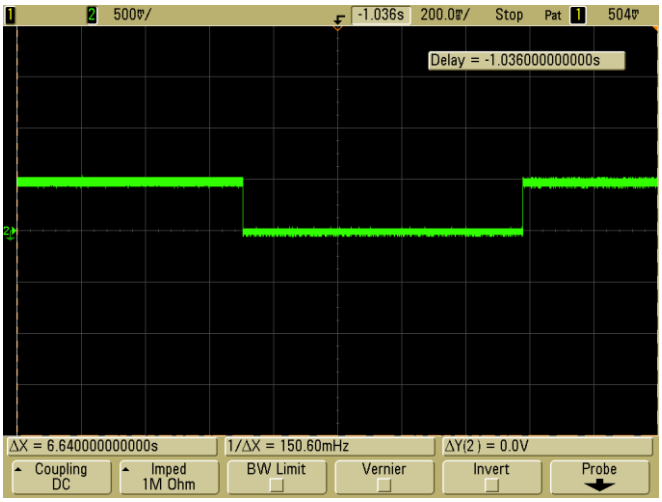
Praktické odzkoušení obvodu, bylo prováděno na nepájivém kontaktním poli, pomocí něhož se dá obvod během krátké doby sestavit a podle potřeby upravovat. Hlavní pomůckou pro proměřování a nalezení řešení problémů, byl osciloskop. Napájení celého obvodu bylo řešeno laboratorním zdrojem nastaveným na hodnotu 5 V. Jelikož není vytvořen žádný plošný spoj, práce je odzkoušena pouze na nepájivém kontaktním poli, nebyl důvod navrhovat vlastní napájení ze sítě 230 V.

3.1 Detekce náběžné hrany

V této části budeme částečně řešit zmíněný problém o řetězení operací a detekci náběžné hrany.

Jako první krok musíme nějakým způsobem detekovat náběžnou hranu signálu reset, která nastane při ukončení operace (např. ukončení nahrávání programu) a podle této hrany se bude řídit celé přepínání. K detekování je zapotřebí vybrat vhodný obvod a tím je monostabilní multivibrátor, známí pod označením HEF4528B. Funkce je popsána v teoretické části, takže není nutné ji vysvětlovat. Pro nás je důležité, že při přivedení náběžné hrany na vstup monostabilního multivibrátoru se vygeneruje puls na jeho výstupu a díky němu víme, že došlo k ukončení operace (např. ukončení nahrávání programu).

Na obrázku 13 je vidět průběh signálu reset. Nejprve je klidový stav v úrovni 5 V a potom je úroveň 0 V značící provádění nějaké operace. Po ukončení operace se vrátí zpět do úrovně 5 V.



Obr. 13 - Signál reset

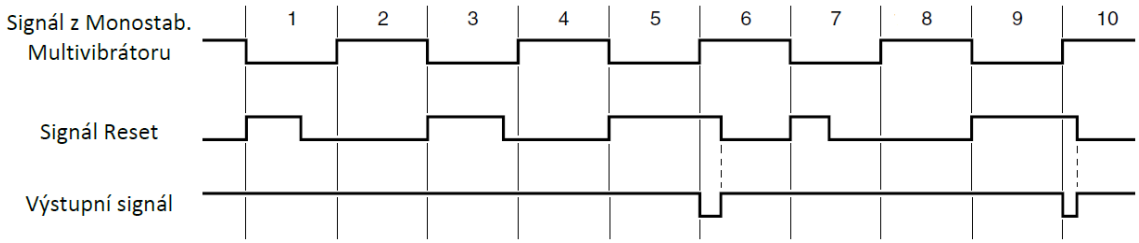
Šířku pulsu (čas trvání pulsu), kterou si nastavujeme kondenzátorem C_{EXT} a odporem R_{EXT}, musíme zvolit takovou, aby byla větší, než je šířka pauzy (logické 1) mezi řetězícími se operacemi a zároveň menší než je šířka pulsu mezi jednotlivými cykly příkazů (obrázek 14). Jestliže si změříme pauzu mezi řetězícími se operacemi (tu větší pauzu mezi cykly nastavujeme v dávkovém souboru podle potřeby), zjistíme, že doba jejího trvání je 300 ms (někdy může být i větší). Doba určená rovnicí uvedenou níže (šířka generovaného pulsu na výstupu), musí být větší, abychom měli rezervu. V našem případě je doba T 600 ms.

$$T = 1{,}1 \cdot R_{EXT} \cdot C_{EXT}$$

3.2 Porovnání pulsů

V dalším kroku budeme porovnávat pulsy. Jedním je signál reset přímo z programátoru a druhým je výstupní signál z monostabilního multivibrátoru. Tento druhý signál reaguje na náběžnou hranu signálu reset, jak jsme si říkali v předešlé části, tudíž pulsy obou signálů začínají zároveň (náběžná hrana signálu reset a sestupná hrana výstupního signálu monostabilního multivibrátoru nejsou vůči sobě posunuty v čase), ale jejich sestupná respektive náběžná hrana (signály mají vůči sobě opačný průběh) přichází po odlišně dlouhých dobách (obrázek 14). Díky tomu, že pulsy obou signálů začínají zároveň, můžeme porovnat jejich dobu setrvání v úrovni logické 1 respektive v úrovni logické 0 pomocí hradla NAND. Hradlo je dvouvstupové, takže na každý vstup přivedeme jeden signál. Když se podíváme na pravdivostní tabulku (tabulka 1), zjistíme, že při třech kombinacích logických úrovní na vstupech, je na výstupu hradla pokaždé logická 1, ale pouze u jedné (čtvrté a zároveň poslední) kombinace je logická 0. Tato kombinace má na obou vstupech logickou 1 a na výstupu je logická 0. Na obrázku 14 vidíme oba vstupující signály do hradla NAND a jeden výstupní. Je vidět, že signály se střídají

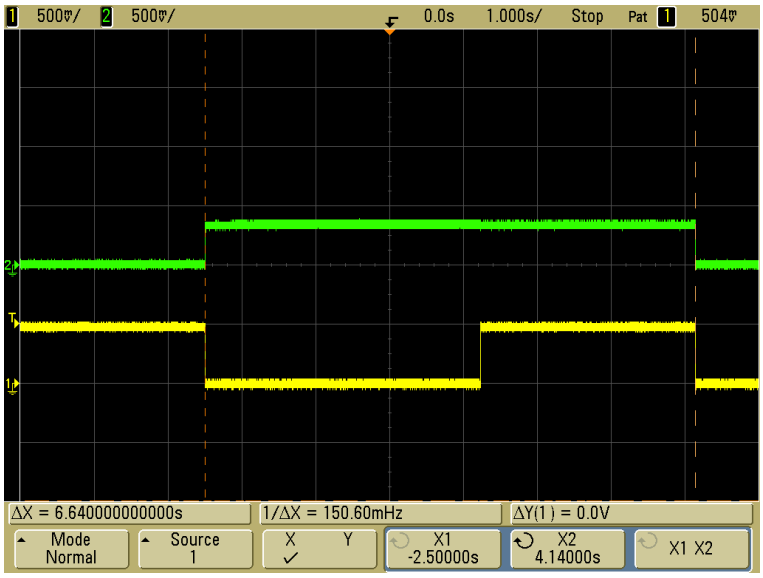
v úrovních takovým způsobem, že se na vstupech objeví kombinace logických jedniček pouze v případě, když trvání puls v úrovni logické 1 signálu reset je delší, než úroveň logické 0 signálu vystupujícího z monostabilního multivibrátoru. Výstupní signál má puls trvající stejnou dobu, jako je doba trvání logických 1 na obou vstupech zároveň. Tento puls označuje okamžik, kdy obvod přepne na další mikrokontrolér.



Obr. 14 - Vstupní signály do hradla NAND

3.3 Asynchronní čítač pulsů

Předposledním prvkem obvodu pro přepínání je asynchronní čítač, jehož funkce je popsána v teoretickém úvodu. Je sestaven ze čtyř klopných obvodů, které jsou zakomponovány v jednom integrovaném obvodu. Tento integrovaný obvod je označen HEF4520B. Práce asynchronního čítače spočívá v tom že, každá sestupná hrana pulsu, přivedená na vstup se projeví navýšením stavu výstupu o jeden (tabulka 2). Je důležité, aby vstupní puls začínal sestupnou hranou a při jeho skončení hranu náběžnou. Když se podíváme na porovnávání pulsů hradlem NAND obrázku 14, vidíme, že první hrana (sestupná) značí okamžik, kdy pauza mezi cykly překročila hranici času, značící, že se má přepnout na další mikrokontrolér. Druhá hrana (náběžná) označuje začátek další operace (např. nahrávání programu). Z toho je patrné, že kdyby byla sestupná hrana pulsu druhá (puls by byl opačně, tj. začínal by hranou náběžnou od 0 V do 5V), obvod by přepínal zároveň se začátkem nové operace. Jakmile by tento moment nastal, způsobil by chybu operace a následně její ukončení. Správný průběh vstupního pulsu je na obrázku 15.



Obr. 15 - Vstupní signál do asynchronního – žlutý, první výstup Q0 - zelený

3.4 Přepínání mezi mikrokontroléry

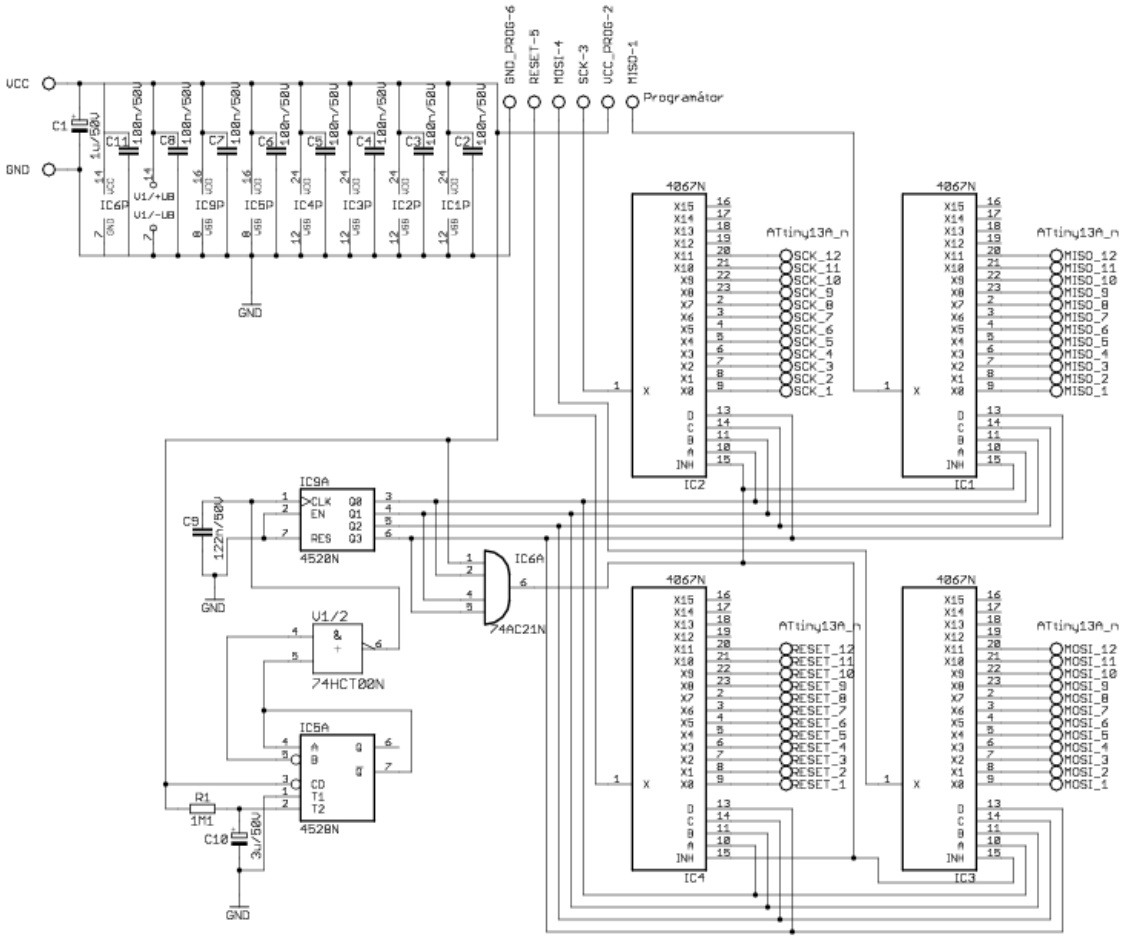
Úplně posledním prvkem obvodu je integrovaný obvod s funkcí multiplexor nebo demultiplexor. Pro naši potřebu budeme tuto součástku využívat jako demultiplexor. Opět je obvod popsán, včetně jeho funkce, v teoretické části. Tento integrovaný obvod má označení 74HC4067.

V obvodu je využit pro přepínání mezi mikrokontroléry. Je potřeba obsloužit 12 mikrokontrolérů ATtiny13A a ke každému jsou přivedeny čtyři dráty. Každý drát je určen pro jeden ze signálů MISO, MOSI, SCK nebo Reset, pomocí nichž jsou prováděny operace a příkazy na mikrokontrolérech. Je tedy jasné, že nebude stačit pouze jeden demultiplexor, ale budou potřeba čtyři. Na obrázku 12 vidíme, že každý z demultiplexorů přepíná jeden ze čtyř zmíněných signálů postupně na každý mikrokontrolér. Všechny čtyři multiplexory mají jeden vstup označený Z a na něj je přiveden právě jeden ze čtyř signálů. Pomocí dekodéru vybíráme jeden výstup, na který tento signál přivedeme. Tento výstup je pak přiveden na odpovídající vstup mikrokontroléru (jestliže je na daném demultiplexoru přepínán signál SCK, bude připojen na pin SCK u mikrokontroléru, stejné je to i u ostatních signálů).

Dekodér má 4 vstupy propojené s výstupy asynchronního čítače a podle těchto vstupních hodnot je určeno, jaký výstup Y0 až Y15 bude propojen se vstupem Z. Výběr výstupu podle hodnot na vstupech dekodéru, je v tabulce 2 (hodnoty CP1 odpovídají výstupům Y0 až Y15, a hodnoty Q0 až Q3 vstupů dekodéru S0 až S3). Z tabulky vidíme, že jsou výstupy vybírány vzestupně popořadě. Jak již bylo zmíněno, na každý ze čtyř demultiplexorů je přiveden jeden ze čtyř signálu a tím pádem je potřeba, aby na všech demultiplexorech byl vybrán

ve stejnou chvíli stejný výstup (například na všech byl vybrán výstup Y0). Z toho plyne, že všechny čtyři signály jsou přivedeny ve stejnou chvíli na odpovídající piny jednoho mikrokontroléru. Toto zajistíme propojením odpovídajících vstupů dekodérů mezi sebou (vstup S0 prvního dekodéru spojíme se vstupem S0 druhého atd.) a tím vyřešíme stejný výběr výstupů ve stejnou chvíli. Máme tedy všechny potřebné signály pro různé operace (mazání, nahrání programu atd.) přivedeny na jeden mikrokontrolér a po příchodu pulsu na asynchronní čítač se všechny signály zároveň přepnou na další mikrokontrolér.

3.5 Schéma



Obr. 16 - Schéma obvodu

Na obrázku je schéma zapojení celého obvodu včetně napájení a všech součástek potřebných ke zhotovení obvodu. V levém horním rohu vidíme konektor pro napájení obvodu, na kterém je připojen stabilizační kondenzátor a hned za ním je rozvedeno napájení k jednotlivým integrovaný obvodů. Napájení a uzemnění je spojeno s programátorem pro vyrovnání hodnot. Každý integrovaný obvod má na

napájecím pinu připojen blokovací kondenzátory s vyznačenými hodnotami. Ze čtyř demultiplexorů vedou přípoje ke konektorům na jednotlivé mikrokontroléry. Konektory jsou popsány podle toho, jaký signál jimi vede a číslem znamenajícím, na jaký mikrokontrolér bude připojen.

Vývod z demultiplexoru INH slouží pro přechod výstupů do stavu vysoké impedance. Jestliže je tento pin připojen na zem, demultiplexor je v normálním funkčním režimu. Když na tento pin přivedeme úroveň 5 V, výstupy přejdou do stavu vysoké impedance a tím se zamezí veškerému přenosu dat mezi vstupem a výstupem.

3.6 Stav vysoké impedance

Pin INH všech čtyř demultiplexorů je připojen na výstup čtyř vstupového hradlo AND. Hradlo má funkci součinu všech vstupů, což znamená, jestliže je na všech vstupech logická 1, na výstupu je taky logická 1, když je alespoň na jednom vstupu logická 0 a výstupu je logická 0. Vstupy hradla AND jsou připojeny na čtyři výstupy asynchronního čítače. Jakmile dojde čítač na poslední hodnotu, kdy má na všech výstupech logickou 1 na vstupech hradla budou pouze logické 1 a na výstupu také. Tím pádem se na pinu INH objeví logická 1 a výstupy přejdou do vysoké impedance. Ovšem je podmínkou, že počet opakování v dávkovém souboru prováděných cyklů operací musí být přesně 16.

Jestliže chceme přesný počet opakování cyklů jako je počet mikrokontrolérů, zjistíme z tabulky ****, že při dvanácti opakováních skončí čítač ve stavu, kdy na výstupu Q0 má log. 1, Q1 log. 1, Q2 log. 0 a Q3 log. 1. V tomto případě připojíme na tři vstupy hradla AND výstupy asynchronního čítače Q0, Q1 a Q3 a na čtvrtý vstup hradla AND připojíme napájecí napětí. Tím dosáhneme přechod výstupů demultiplexoru do vysoké impedance, když na těchto třech výstupech asynchronního čítače budou logické 1.

3.7 Problémy při zapojování

Problémy vzniklé při zapojování byli hlavně u asynchronního čítače. Ve schématu je vidět kondenzátor na vstupu čítače. Ten používáme z důvodu odfiltrování zákmitů, které přepínají čítač dříve, než je žádoucí. Proti tomuto problému byli, přidány před napájecí vývody všech integrovaných obvodů blokovací kondenzátory. Je důležité, aby byl kondenzátor, zapojeny nejlépe těsně k napájecímu pinu a druhý konec těsně k zemnícímu. Tím zajistíme co nejmenší okruh.

3.8 Příkazy pro spuštění programování

Jak již bylo v úvodu zmíněno, spouštění programování formou příkazů se realizuje pomocí dávkového souboru. Ten při spuštění vykoná příkazy v příkazovém řádku, přesně, jak jsou napsány v dávkovém souboru. Je možné řetězit více příkazů (operací), prováděných na jednom mikrokontroléru, za sebou.

Jestliže chceme nahrát do každého mikrokontroléru jiný program, je zřejmé že nebudeme moci použít opakovací cyklus, který se několikrát za sebou provede. Je nutné řetězec operací, který chceme provést na prvním mikrokontroléru, napsat na jeden řádek a další řetězec na nový řádek pro druhý mikrokontrolér atd. Sice je to nepraktické, ale jiným způsobem to nelze udělat.

Jinak je tomu, jestliže chceme nahrávat do všech mikrokontrolérů stejný program. V tomto případě je velice užitečné použít opakovací cyklus i s počtem opakování, aby se po posledním naprogramovaném mikrokontroléru dávkový soubor sám vypnul a nepokračoval dál.

Ukázky opakovacích cyklů a příkazů

```
set /a start=0
:zacatek
if %start%==12 goto stop

"C:\Program Files (x86)\Atmel\Atmel Studio
6.1\atbackend\atprogram.exe" -t avrdragon -i isp -d
attiny13a chiperase program -f c:\test1.elf

set /a start=%start%+1
ping localhost -n 5 >nul
goto zacatek
:stop
```

Tento zápis je celý obsah jednoho dávkového souboru. Na prvním řádku vidíme příkaz, zapisující hodnotu 0 do konstanty „start“. Díky tomu začíná počítání opakování od hodnoty 0 a né od nějaké náhodné hodnoty. Další příkaz je pouze návěstí, na které skočí program po provedení předposledního příkazu „goto zacatek“ a tím se provede zopakování příkazů, nacházejících se mezi těmito dvěma. Na třetím řádku se provádí podmínka. Jestliže se hodnota konstanty „start“ rovná dvanácti, provede se přeskok na návěstí stop a tím se ukončí celý cyklus. Když se hodnota konstanty „start“ nerovná hodnotě 12 (tak je v našem případě menší než 12) žádný přeskok na návěstí „stop“ se nebude provádět a pokračuje se na další řádek. Na dalších třech řádcích je napsán jeden příkaz provádějící samotné programování mikrokontroléru (všechny příkazy okolo slouží, pouze k zajištění určitého počtu opakování). Tento příkaz provádí pouze mazání mikrokontroléru a následné nahrání programu. Na dalším řádku vidíme počítání opakování. Po každém naprogramování se na tomto řádku hodnota konstanty „start“ navýší

o plus jedna. Z toho plyne, že jakmile se hodnota konstanty „start“ zvýší na hodnotu 12, na třetím řádku se provede přeskok na návěstí „stop“ a cyklus končí. Třetí řádek od konce je tam pouze z důvodu pauzy mezi jednotlivými cykly opakováni. Tuto pauzu je možné zvětšovat nebo zmenšovat, když za „-n“ napíšeme vyšší nebo nižší číslo (teď je tam číslo 5).

Ukázka příkazu programování

```
"C:\Program Files (x86)\Atmel\Atmel Studio
6.1\atbackend\atprogram.exe" -t avrdragon -i isp -d
attiny13a chiperase program -c --verify -f c:\test11.elf
write --verify -fs --values FFFFFFFE write --verify -lb --
values 10
```

Toto je řetězení několika operací prováděných na jediném mikrokontroléru. Popíšeme je, jak jdu za sebou:

```
"C:\Program Files (x86)\Atmel\Atmel Studio
6.1\atbackend\atprogram.exe"
```

Tato část je cesta k programu „atprogram.exe“, bývá tam, kde je nainstalováno Atmel Studio.

```
-t avrdragon -i isp -d attiny13a
```

První je název použitého programátoru, následuje název sběrnice a název programovaného mikrokontroléru.

```
chiperase program -c --verify -f c:\test11.elf
```

Nejprve se provede mazání programu, potom verifikace a dále cesta k uloženému programu, který se nahraje do mikrokontroléru.

```
write --verify -fs --values FFFFFFFE write --verify -lb --
values 10
```

Dalším krokem je opět verifikace, následuje nastavení fuses, opět se provede verifikace a nakonec nastavení lock bit.

4 Návrh obvodu pro různé počty mikrokontrolérů

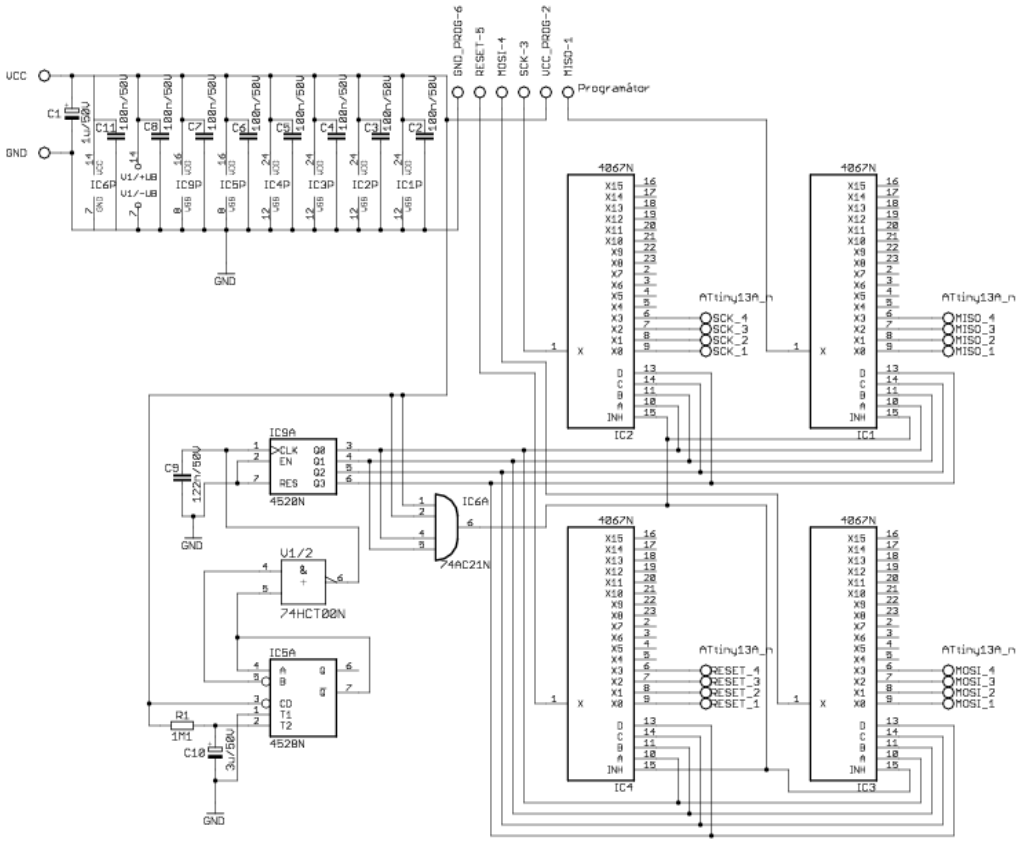
Posledním bodem této práce bude návrh přepínacího obvodu pro čtyři a pro čtyřicet mikrokontrolérů. Návrh pro čtyři mikrokontroléry je odzkoušen na kontaktním nepájivém poli. Druhý návrh pro čtyřicet mikrokontrolérů je odzkoušen pouze částečně z důvodu potřeby velkého množství integrovaných obvodů, zejména demultiplexorů.

4.1 Návrh pro čtyři mikrokontroléry

Obvod je stejný jako pro dvanáct mikrokontrolérů, protože demultiplexory jsou poměrně levná záležitost a tudíž nebyl důvod je měnit za jiné integrované obvody z důvodu velkého rozdílu cen. Schéma obvodu je na obrázku 17. Odlišnosti jsou jenom v rozdílném počtu konektorů jdoucích k mikrokontrolérům a se zapojením hradla AND.

Místo dvanácti konektorů na každém demultiplexoru jsou na každém čtyři konektory. Tomu to počtu se přizpůsobí i počet opakování cyklů v dávkovém souboru.

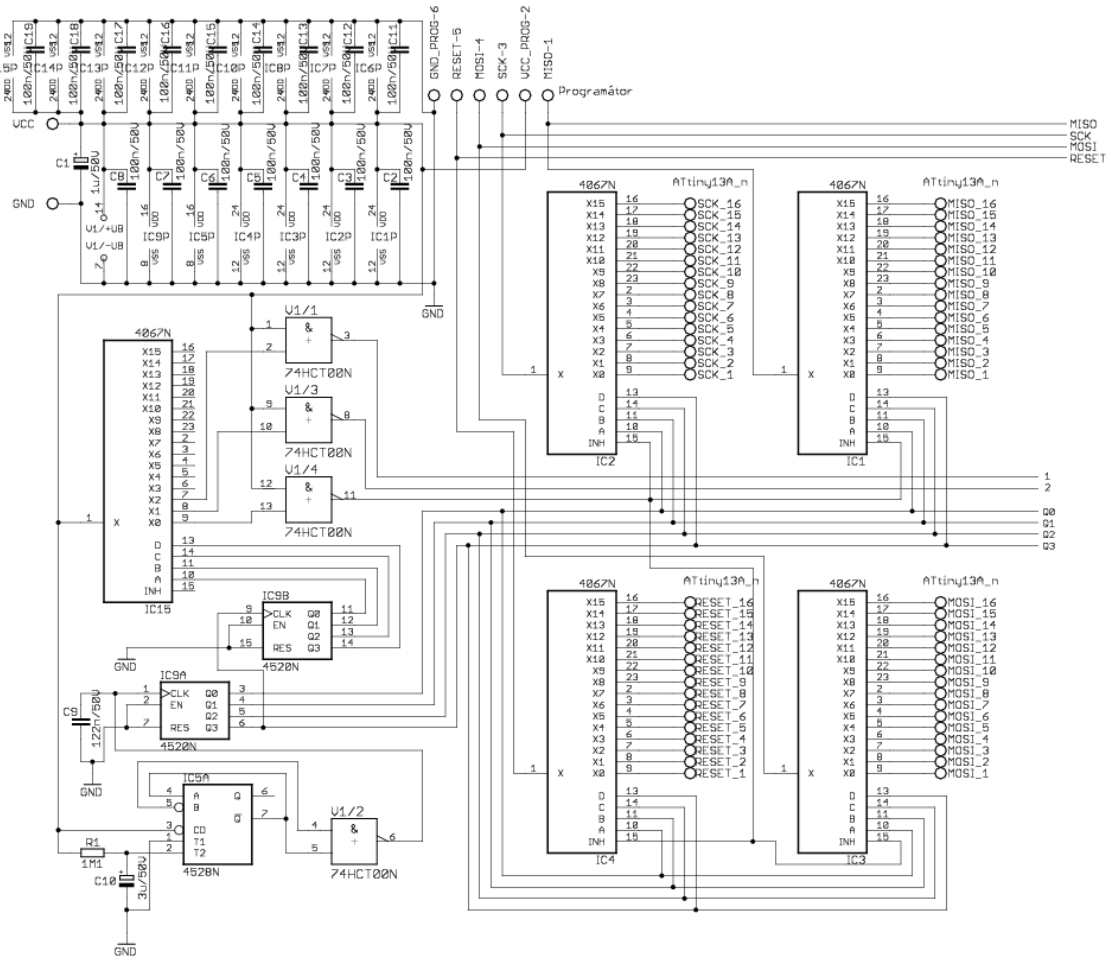
Hradlo AND musí mít dva vstupy připojené na výstupy asynchronního čítače Q0 a Q1, zbylé dva pak na napájecí napětí. Tímto zapojením zajistíme, že výstupy přejdou do stavu vysoké impedance.



Obr. 17 - Schéma návrhu pro čtyři mikrokontroléry

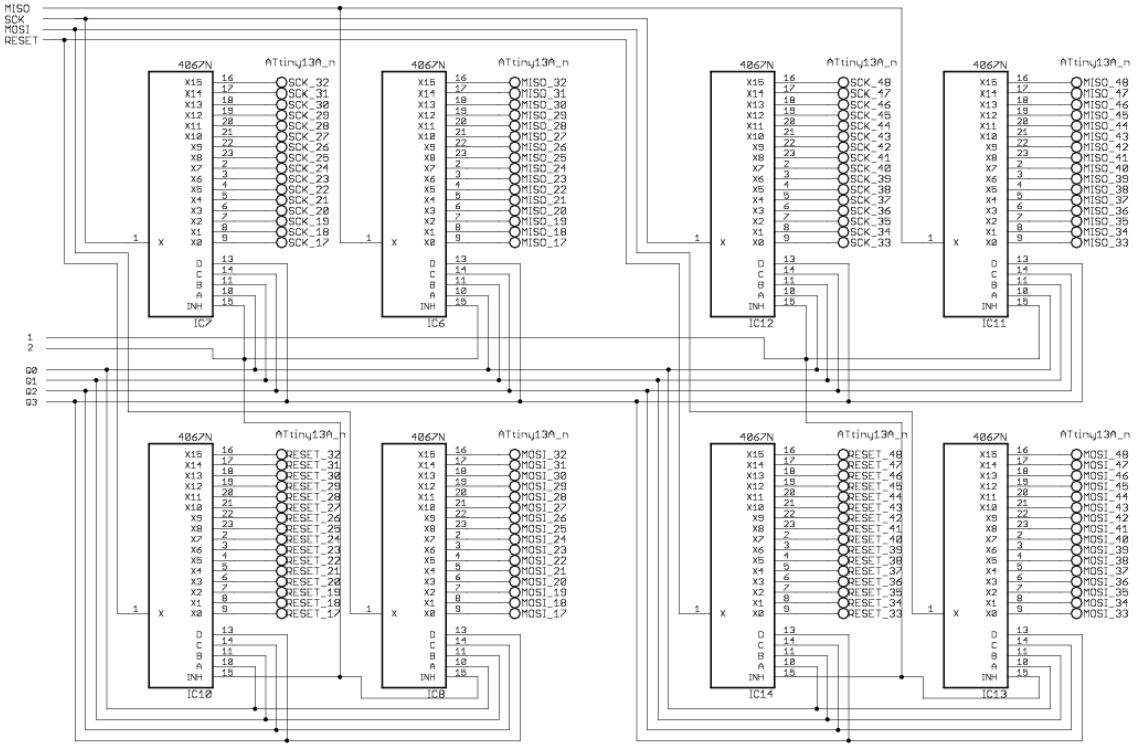
4.2 Návrh pro čtyřicet mikrokontrolérů

Obvod je složitější oproti obvodu s dvanácti mikrokontroléry, schéma je na obrázku 18 a 19. Obvod je použit stejný, akorát je zvětšen o další integrované obvody. Počet demultiplexorů se zvýšil o osm. Adresní vstupy S0 až S3 (na schématu X0 až X3)jsou propojeny opět všechny jako u verze s dvanácti mikrokontroléry. Jelikož je zapotřebí přepojovat mezi čtyřiceti mikrokontroléry, musíme pro každý signál (MISO, MOSI, SCK a Reset) mít propojeny tři demultiplexory. To znamená, potřebujeme tři demultiplexory na MISO, další tři na SCK atd. Jsou mezi sebou propojeny vstupy Z (ve schématu X). Tím pádem je na každém ze trojice demultiplexorů stejný signál. V tuto chvílí by ovšem přepínali demultiplexory ve trojicích zároveň a tím pádem by byli připojeny najednou tři mikrokontroléry. Je tedy potřeba, aby ve trojici pracoval nejprve jeden demultiplexor a až na tomto demultiplexoru bude ukončeno programování šestnáctého mikrokontroléru, tak aby se tento demultiplexor vypnul a začal pracovat druhý z trojice a stejným způsobem nakonec i třetí demultiplexor. Výběr jednoho z každé trojice demultiplexorů provádí ještě jeden samostatně zapojený demultiplexor. Na vstupy S0 až S3 tohoto samostatného demultiplexoru jsou přivedeny výstupy z nového (dalšího čítače). Tento nový čítač je stejný jako předešlí a je s ním propojen tak, že vstup CP1 nového čítače je spojen s výstupem Q3 čítače starého (toho co byl použit již ve verzi pro dvanáct mikrokontrolérů). To znamená, že jakmile dojde první čítač na hodnotu šestnáct (šestnáctý mikrokontrolér) nový čítač se zvýší o jeden. Díky tomu se přepne samostatný demultiplexor na další výstup. Vstup tohoto demultiplexoru je připojen na napájecí napětí. Tři výstupy tohoto demultiplexoru jsou spojeny se třemi hradly NAND a ty mají jeden vstup připojen na napájecí napětí a jeden právě s výstupy nového demultiplexoru. Jestliže je na výstupu demultiplexoru úroveň 5 V hradlo NAND bude mít na obou vstupech úroveň 5 V a na výstupu úroveň 0 V. Výstupy všech tří hradel NAND jsou připojeny k pinům INH. Jedno hradlo je připojeno k demultiplexorům přepínacím mezi mikrokontroléry 1 až 16, další hradlo k demultiplexorům přepínacím mezi mikrokontroléry 17 až 32 atd. Jelikož na výstupech tří hradel je pouze na jednom z nich úroveň 0 V (to je dáno samostatným demultiplexorem) je vybrána vždy pouze jedna skupina přepínacích demultiplexorů (buď skupina 1 až 16, 17 až 32 nebo 33 až 40 popřípadě více mikrokontrolérů až 48).



Obr. 18 – První část schéma obvodu pro čtyřicet mikrokontrolérů

Jelikož je schéma velké, je rozděleno na dvě schémata. V první části schéma vidíme na obrázku v levém horním rohu napájení všech integrovaných obvodů, obvod na přepínání mezi jednotlivými skupinami demultiplexorů (tato část se nachází pod napájecí částí) a také první skupinu demultiplexorů přepínajících mezi mikrokontroléry 1 až 16. Toto první schéma je připojeno ke druhému, přes pravou část, kde jsou označeny vývody (úplně vpravo).



Obr. 19 - Druhá část schéma obvodu pro čtyřicet mikrokontrolérů

Zde na obrázku vidíme druhé schéma připojující se ke schématu prvnímu vývody nalevo. Jsou zde pouze další dvě skupiny demultiplexorů (po čtyřech, první čtyři zleva druhé čtyři zprava). Konektory pro mikrokontroléry jsou až do čísla čtyřicet osm (je možné navíc připojit dalších osm mikrokontrolérů).

5 Závěr

Cílem této práce bylo sestavit obvod pro automatizované programování více mikrokontrolérů Atmel AVR přes SPI sběrnici a navrhnout a realizovat demonstrační přípravek. Pro vyřešení daného problému byl sledován signál „reset“, pomocí kterého jsme přepínali mezi jednotlivými mikrokontroléry. Během zkoušení se narazilo na problém, kdy signál reset na začátku programování skočil do log. 0, ale ještě před ukončením programování se objevilo několik pulzů do log. 1. To způsobovalo předčasné přepnutí na další mikrokontrolér. Tento problém byl vyřešen přidáním kondenzátorů a tím bylo dosaženo fungování obvodu pro automatizované programování více mikrokontrolérů.

6 Literatura

Atmel, *Aplikační list Atmel AVR910: In Systém Programming*. 2010 [cit. 2014-05-30]. Dostupné z: http://www.atmel.in/Images/doc0943.pdf

Atmel, *Katalogový list Atmel ATtiny 13A*, kapitola 17.6. „Seriál Programming". 2010 [cit. 2014-05-30]. Dostupné z: http://www.atmel.com/images/doc2535.pdf

Atmel. *Atmel studio* [online]. 2014 [cit. 2014-05-30]. Dostupné z: http://www.atmel.com/tools/atmelstudio.aspx

Atmel. *Technical Support* [online]. 2012 [cit. 2014-05-30]. Dostupné z: http://support.atmel.com/bin/customer.exe?=&action=viewKbEntry&id=686

AVR Dragon: návod k použití. [cit. 2014-05-30]. Dostupné z: http://www.gme.cz/img/cache/doc/752/534/prog-at-dragon-cznavod-1.pdf

BRTNÍK, Bohumil. *Číslicové systémy*. BEN, 2011. ISBN 978-80-7300-4.

Čítače. *Asynchronní čítače* [online]. 2012 [cit. 2014-05-30]. Dostupné z: http://izm.euweb.cz/cislicova_technika_3.pdf

DUDÁČEK, K. *Sériová rozhraní SPI, Microwire, I2 C a CAN*. 2002, 19 s. Dostupné z: http://home.zcu.cz/~dudacek/NMS/Seriova_rozhrani.pdf

DUCH, Miroslav. *Mikroprocesory AVR Tiny*. Střední průmyslová škola Trutnov, Školní 101, 2009, 47 s. Dostupné z: http://www.spstrutnov.cz/o-skole/projekty/moderni-vyuka-mikroprocesorove-techniky/mikroprocesory-avr-tiny-skripta.pdf

mcu.cz. *AVR Dragon* [online]. 2009 [cit. 2014-05-30]. Dostupné z: http://mcu.cz/news.php?extend.1588

Mikrokontroléry PIC. *Booleova algebra – logické funkce* [online]. 2012 [cit. 2014-05-30]. Dostupné z: http://mikrokontrolery-pic.cz/zaciname/cislicova-technika/booleova-algebra-logicke-funkce/

Mikrokontroléry PIC. *Realizace základních logických funkcí hradly NAND a NOR* [online]. 2012 [cit. 2014-05-30]. Dostupné z: http://mikrokontrolery-pic.cz/zaciname/cislicova-technika/kombinacni-logicke-obvody/realizace-zakladnich-logickych-funkci-hradly-nand-a-nor/

Mikrokontroléry PIC. *Co je to mikrokontrolér?* [online]. 2012 [cit. 2014-05-30]. Dostupné z: http://mikrokontrolery-pic.cz/zaciname/co-je-to-mikrokontroler/

npx.com *74HC(T)4067* [online]. 2014 [cit. 2014-05-30]. Dostupné z:http://www.nxp.com/products/logic/analog_switches/series/74HC_T_4067.html#overview

OLIVKA, Petr. *Technologie výroby číslicových obvodů*. Ostrava, 2010. Dostupné z: http://poli.cs.vsb.cz/edu/arp/down/technologie.pdf

PHILIPS. *HEF4528B: Product data sheet*. 2009. Dostupné z: http://www.gme.cz/

PHILIPS. *HEF4520B: Product data sheet*. 2009. Dostupné z: http://www.gme.cz/

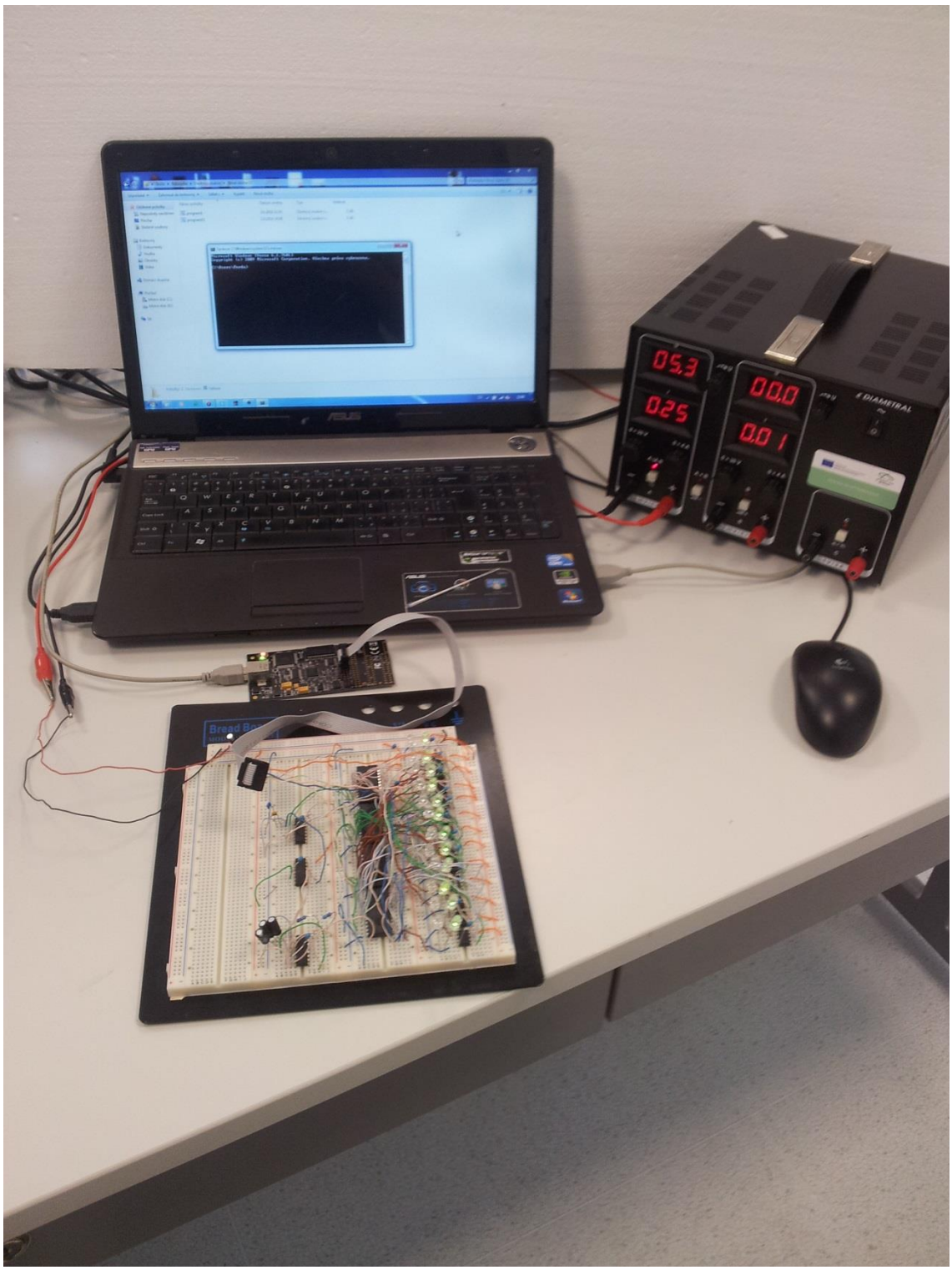
PHILIPS. 74HC4067;74HCT4067: *Product data sheet*. 2009. Dostupné z: http://www.gme.cz/

Windows Server. *Použití dávkových souborů* [online]. 2014 [cit. 2014-05-30]. Dostupné z: http://technet.microsoft.com/cs-cz/library/cc758944(v=ws.10).aspx

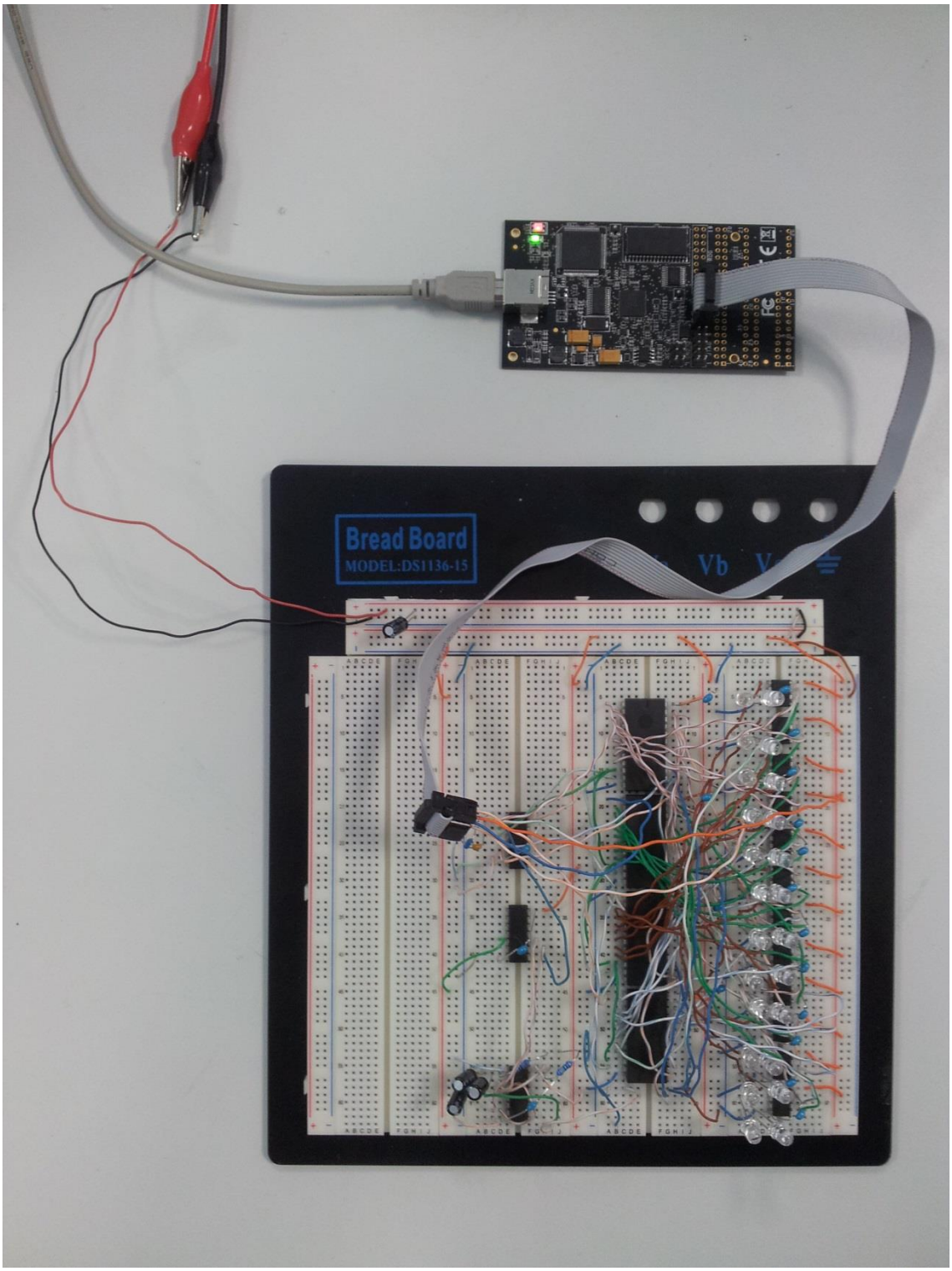
Bohumil Brtník, Číslicové systémy. BEN, 2011. ISBN 978-80-7300-4.

7 Seznam příloh

8 Přílohy



Obrázek 1



Obrázek 2