VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV MIKROELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF MICROELECTRONICS

MATICOVÝ LED DISPLAY 8X64

LED MATRIX 8X64

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

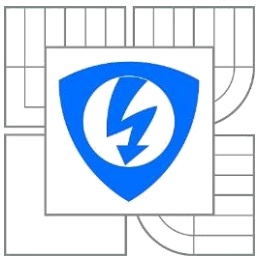
AUTOR PRÁCE — MICHAL MAREK

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. PAVEL ŠTEFFAN, Ph.D.

SUPERVISOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií Ústav mikroelektroniky

Bakalářská práce

bakalářský studijní obor
Mikroelektronika a technologie

Student: Michal Marek ***ID:*** 155196
Ročník: 3 ***Akademický rok:*** 2014/2015

NÁZEV TÉMATU:

Maticový LED display 8x64

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte a zrealizujte maticový LED display o velikosti 8x64 bodů. Pro řízení matice využijte vhodný mikrokontrolér. Při návrhu se zaměřte na možnost zadávání zobrazovaného textu pomocí PC přes rozhraní USB a LAN. Důležitým aspektem návrhu je výsledná cena zobrazovače.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce

Termín zadání: 10.2.2015 ***Termín odevzdání:*** 4.6.2015

Vedoucí práce: doc. Ing. Pavel Šteffan, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Jiří Háze, Ph.D.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Tato semestrální práce se zabývá popisem LED diodových matic a způsobem jejich řízení. V projektu je popsána funkce řídících obvodu pro LED matice, funkce řadiče a způsoby programování mikrokontrolérů. To zahrnuje programování mikrokontrolérů přes USB a změny ve zdrojových kódech přes síť ethernet za pomoci modulu pro připojení k ethernetu, použití standartních sběrnic pro sériové přenosy a způsoby řízení LED diod pomocí příslušných knihoven. Cílem je nastínit řešení pro způsob řízení LED pomocí softwarové aplikace Windows a navrhnout komplexní sestavu obsahující všechny příslušné bloky, která zajistí jednoduchost ovládání pro nové uživatele.

Klíčová slova

Matice, dioda, LED dioda, LED matice, displej, maticový displej, řídící obvody LED, řízení LED diod, Ethernet modul, řadič, formulářová aplikace Windows, ethernet

Abstract

The bachelor thesis describes what is a LED matrix and how to control them. The function of the control circuits for the LED matrixes, the function of the current controller and microcontroller programming method are delineates in the project. These include microcontroller programming via universal serial bus, changing source code parameters via local ethernet network with the ethernet connect module, using standard peripheral buses for serial transmissions and the ways of controlling LEDs using relevant libraries. Objective is to introduce a solution for LED controlling using the Windows software application and propose complex assembly includes all the necessary blocks, which provides simple operating for new users.

Key words

Matrix, diode, LED diode, LED matrix, display, matrix display, LED drivers, control LEDs, Ethernet shield, current controller, Windows form application, ethernet

Bibliografická citace

MAREK, M. *Maticový LED display 8x64.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2015. 38 s. Vedoucí bakalářské práce, doc. Ing. Pavel Šteffan, Ph.D.

Prohlášení

Prohlašuji, že svoji bakalářskou práci na téma **„Maticový LED displej 8x64“** jsem vypracoval samostatně pod vedením vedoucího semestrálního projektu 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 práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 4. června 2015 .....

podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce **doc. Ing. Pavlu Šteffanovi, Ph.D** za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování práce.

V Brně dne 4. června 2015 .....

podpis autora

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

Digitální obvody nacházejí v současné době širokou škálu aplikací. Hlavní výhodou digitálních elektronických obvodů je lepší, resp. přesnější kontrola nad jejich logickými stavy, které jsou vybírány konečné posloupnosti hodnot. Využití je možné pro výpočty matematických operací nebo provádění jiných přesně požadovaných funkcí. Splňují tedy požadavky pro řízení a kontrolu všemožných aplikací v oblastech elektrotechniky a výrobních procesů.

Práce v první části shrnuje základní teorií integrovaných řadičů (budičů) pro displeje. Obvykle nacházejí aplikace v řízení sedmi-segmetových displejů nebo maticových LED displejů. Budiče jako samostatný blok nevykonávají žádné procesy bez vstupního podnětu. Z tohoto důvodu je v teoretické části popsáno jako budiče použít a také naznačeno jak tyto obvody řídit pomocí hardwarových modulů s mikroprocesory. Výhoda použití budičů také spočívá v principu řízení pomocí sériového toku dat. Tento fakt vede také k redukci počtu vodičů mezi řídícím obvodem a budičem a mimo jiné také k požadavku použití oscilátoru pro hodinový signál s vyšší frekvencí.

Druhá část práce popisuje výsledky praktické realizace displeje pomocí budících obvodů a programovatelného mikroprocesoru, který je řízen na úrovni lokální sítě v reálném čase. Taková realizace vyžaduje zajištění kompatibility mezi tokem dat ze sítě a mikroprocesorem. Na základě libovolného internetového nebo smyšleného protokolu a vhodného převodu dat sítě a sběrnici kompatibilní pro daný mikroprocesor je možné kompatibilitu zajistit.

Předností použití digitálních zobrazovacích zařízení (displejů) ve srovnání s obrázkem či textem tištěným na daném podkladu je vzdálená kontrola nad zařízením (komunikace pomocí logických stavů) nebo např. velká univerzálnost (rychlá možnost přepisu). Za pomoci řídících obvodů je lze využívat pro potřebné účely, umožňuje-li to zařízení. S ohledem na cenu, tedy i spotřebu je nutno pohlížet na fakt, že LED diody se vyznačují vysokou účinností přeměny elektrické energie na světlo. [5]

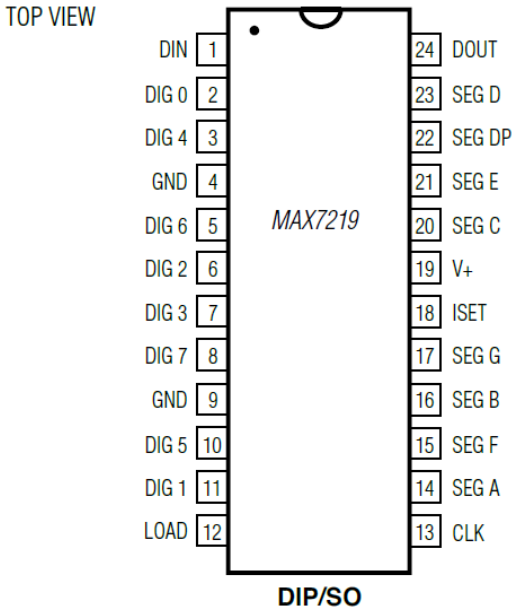
1 TEORETICKÁ ČÁST

1.1 MAX 7219

MAX 7219 je elektronický integrovaný obvod od výrobce značky MAXIM. Integrovaný obvod je určený pro elektrické řízení LED-diodových displejů (LED-Driver). Jeho použití po ovládání LED displejů není nezbytné, ale je optimální jej využívat, protože velmi zjednodušuje způsoby komunikace mezi mikroprocesorem a výstupním zařízením, tedy LED displejem. Jeho vstupní piny jsou kompatibilní pro sériová rozhraní, která jsou zároveň výstupy vybraného mikroprocesoru. Při návrhu a realizaci obvodů na deskách plošných spojů se umísťuje řídící obvod do blízkosti LED displeje. Každý LED displej vyžaduje svůj driver. Z driveru jsou vedeny paralelní výstupy přímo do jednotlivých segmentů displeje. Velmi zjednodušenou formou lze tvrdit, že obvod se chová podobně jako demultiplexor, který převádí sériově jdoucí data do paralelních větví. Pod pojmem data si lze představovat posloupnost dvou logických stavů (log. 1 a log. 0), které představují dvě různé úrovně signálu a jsou nositelem informace. Jeden MAX7219 je schopný řídit 64 individuálních segmentů. Má 2 x 8 výstupních pinů pro vytvoření imaginárního pole 8 x 8. [4]

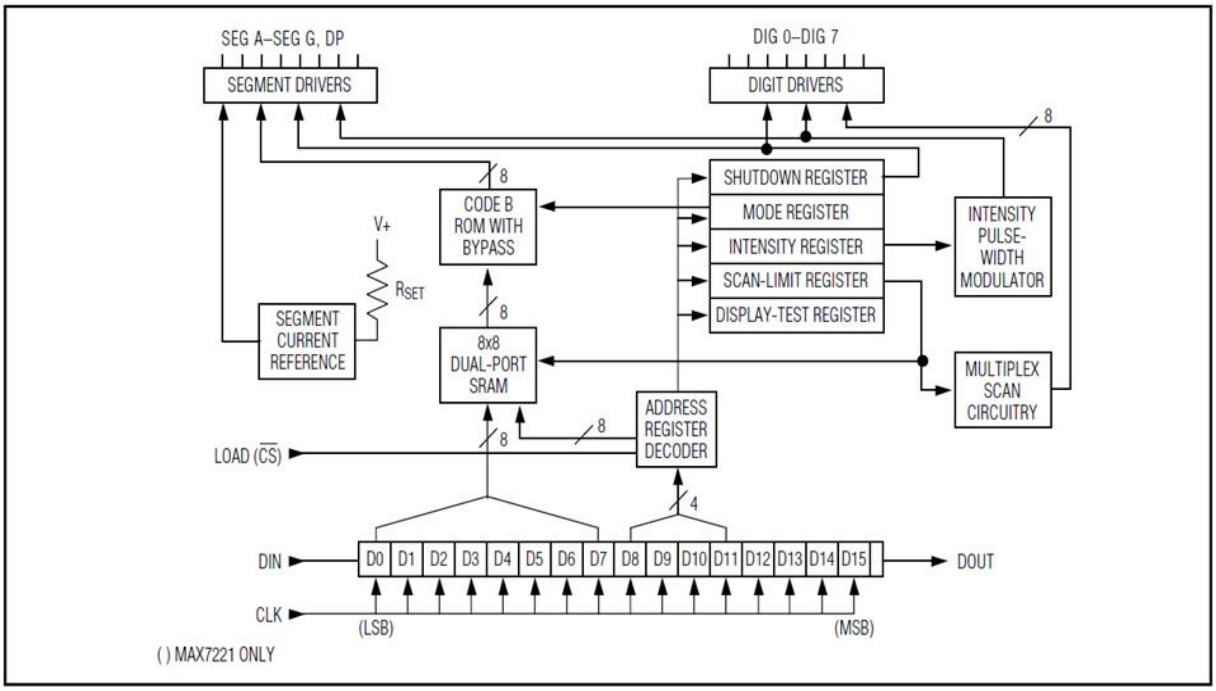
MAX7219 zahrnuje registr vstupních dat, dekodér adres pro registry, dvouportovou paměť S-RAM, blok nastavení referenčních proudů pro diodové segmenty, dekodér BCD (hexadecimálního formy) kódu pro převod na binární kód, pro řízení intenzity svícení LED řídící registr, pulze šířkový modulátor (PWM) a řadiče pro jednotlivé segmenty.

Sériový vstup MAX7219 je kompatibilní pro sériová rozhraní SPI, QSPI a MAX 7221 také pro Microwire. Jako výstupní blok, resp. LED-diodový displej se k tomuto řídícímu obvodu připojují sedmi-segmentové LED-displeje, LED-diodové matice, 64 samostatných LED-diod nebo jiné LED-zobrazovače (Např. zobrazovač sloupcových grafů).



Obr. 1 – Zobrazení pouzdra integrovaného obvodu MAX 7219 a číslování vývodů [1]

BLOKOVÉ SCHÉMA A POPIS



Obr. 2 – Blokové schéma obvodu MAX 7219 [1]

Na vstupní části obvodu se nachází 16-ti bitový posuvný registr řízený náběžnou hranou hodinového signálu. Vstupní svorka LOAD zajišťuje povolení k nahrání dat do paměti zařízení. Stejnosměrné napájecí napětí obvodu je 5 V. K napájecí svorce se připojuje rezistor k omezení proudu, který zároveň nastavuje referenční hodnotu proudu pro blok nastavení proudu do diodových segmentů. Přímo ze vstupního posuvného registru se nahrávají data do paměti. Registry kontroly limitů se nacházejí za výstupem dekodéru vstupních dat pro registry, který je připojen na posuvný registr vstupních dat. Ve výstupní části jsou řadiče pro LED-diodové segmenty, které nastavují požadované proudy na anody LED-diod.

VSTUPNÍ DATA

Do obvodu MAX7219 jsou po kompatibilních sériových sběrnicích posílána data v 16bitových paketech. Vstupním pin pro data je DIN, pro hodinový signál vstup CLK a pro signál k nahrání dat pin LOAD. Načítání dat do posuvného registru probíhá s příchodem náběžné hrany hodinového signálu bez ohledu na logickou úroveň na pinu LOAD. S 16. náběžnou hranou hodinového signálu přichází poté příkaz pro nahrání dat, tedy na pin LOAD je přivedena úroveň logické 1. Je nutné zajistit příchod logické 1 na LOAD dříve než přijde následující sestupná hrana hodinového signálu, jinak budou data ztracena, resp. přepisována bity z následujícího paketu, a tudíž budou nesprávná. S ohledem na velikost registru bude proces nahrání paketu trvat 16 náběžných hran. Po 16. náběžné hraně hodinového signálu se data začínají přenášet na výstupní pin DOUT s každou následující sestupnou hranou hodinového signálu.

Tab. 1 – struktura 16-ti bitového paketu vstupních dat obvodu MAX 7219 [1]

D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12-D15
Data ve formě samostatných bitů pro řadič segmentů								4bitová adresa pro registry (i pro digity)				x

Data D0 - D7 se využívají především pro řízení segmentů přímo, pro nastavení módu ve, kterém bude obvod pracovat (operativní mód nebo mód vypnutí = „shutdown“), pro řízení dekodéru BCD kódu nebo pro řízení intenzity LED segmentů. Data D8-D11 jsou užita pro řízení řadičů digitů (digit 0 – digit 7), avšak informace je zapsána ve čtyřbitové binární kombinace.

ŘÍZENÍ INTENZITY

V obvodu MAX 7219 je možné řídit intenzitu LED-diod dvěma způsoby. Jednak analogově externím rezistorem, resp. potenciometrem, který se připojuje mezi piny V_+ a I_{SET}. Maximální (špičková) hodnota proudu z proudových zdrojů řadiče segmentů je omezována na 1/100 proudu, který protéká svorkou I_{SET}. Požadovanou minimální hodnotou elektrického odporu externího rezistoru je R = 9,53 kΩ, který nastavuje proud do segmentů I_{SEG} = 40 mA. Druhý způsob je digitální řízení intenzity. V tomto případě se řídí intenzita programově, resp. pomocí vstupních dat. Řízení zajišťuje registr pro řízení intenzity a modulátor pulzně šířkové modulace (PWM modulátor). Registr pracuje s bity D0-D3 ze vstupních dat. Modulátor pracuje tak, že z maximální hodnoty proudu segmetu nastavené rezistorem R_{SET} vytváří 16 úrovní modulace, reps. 16 úrovní střídy modulovaného signálu v poměru s = 31/32 až 1/32 (což je minimální intenzita), viz tabulka č. 2. intenzita).

Tab. 2 – Tabulka vstupních dat pro nastavení parametrů PWM pro řízení intenzity [1]

Poměr šířky pulsu/perioda	Binární kód				Hexadecimální kód
	D3	D2	D1	D0	D3-D0
1/32	0	0	0	0	0xX0
3/32	0	0	0	1	0xX1
5/32	0	0	1	0	0xX2
7/32	0	0	1	1	0xX3
9/32	0	1	0	0	0xX4
11/32	0	1	0	1	0xX5
13/32	0	1	1	0	0xX6
15/32	0	1	1	1	0xX7
17/32	1	0	0	0	0xX8
19/32	1	0	0	1	0xX9
21/32	1	0	1	0	0xXA
23/32	1	0	1	1	0xXB
25/32	1	1	0	0	0xXC
27/32	1	1	0	1	0xXD
29/32	1	1	1	0	0xXE
31/32	1	1	1	1	0xXF

MÓD SHUTDOWN

Instrukce pro shutdown pochází z vlastních knihoven MAX7219 určených pro řízení LED diodových zařážení. Po příchodu instrukce pro shutdown, resp. vypnutí daného zařízení, se zastaví interní oscilátory pro kontrolu limitů, všechny větvě z řadičů LED segmentů jsou přitaženy k zemi (pulled down) a všechny piny řadičů samotných jsou přitaženy k napájecímu napětí V+. Z toho vyplývá přerušení mezi řadičem a zařízením LED. Potom jsou všechny LED diody zhasnuty. Vzhledem k tomu, že doba trvání přechodu do módu shutdown nebo naopak náběh obvodu je velmi rychlý, lze tuto funkci využívat k úsporám energie. Pro minimální proudové odběry se vstupy MAX 7219 připojují k neměnným větvím, tj. napájecí větev V+ nebo zemnící větev GND. Mezní rychlost zotavení z režimu shutdown je t_{STO} = 250 µs.

DEKODÉR BCD KÓDU

Po uvedení BCD dekodéru do funkce dochází k převodu vstupních dat z hexadecimálního kódu do binární kombinace. Je-li BCD dekodér vypnut, potom vstupní data nastavují individuálně jednotlivé segmenty. Zápis hexadecimálního kódu je kratší. Příchozí data D0-D7 mohou nést informaci o tom, které část datové části příchozího paketu bude dekódována z hexadecimálního tvaru kódu do binárního.

Tab. 3 - Vstupní data pro nastavení režimu dekódování [1]

MÓD DEKÓDOVÁNÍ	DATA REGISTRU								HEXA KÓD
	D7	D6	D5	D4	D3	D2	D1	D0	
Žádné dekódování	0	0	0	0	0	0	0	0	0x00
Dekódování DIG 0	0	0	0	0	0	0	0	1	0x01
Dekódování DIG 3 - DIG 0	0	0	0	0	1	1	1	1	0x0F
Dekódování DIG 7 - DIG 0	1	1	1	1	1	1	1	1	0xFF

ŘADIČE PRO ŘÁDKY A SLOUPCE

Řadič je koncový řídící blok diodových segmentů. Je přímo spojený s anodami jednotlivých LED diod. Druhý přítomný řadič je obvykle spojen se společnými katodami segmentů (nepřímo přes tranzistory). Úkolem řadiče je nastavit do aktivních segmentů požadovaný proud a mít kontrolu nad jeho velikosti. Chová se tedy jako proudový zdroj. Dalším jeho úkolem je přepínání zobrazených řádků. Jelikož se obrazec na displejích obvykle vykresluje postupně je nutno posílat data do řádků (sloupců) postupně tedy po řádcích nebo po sloupcích. Frekvenci přepínání mezi řádky (sloupci) zajišťuje interní oscilátor o pevně dané frekvenci F_{OSC} = 800 Hz. Výsledná frekvence chodu postupného vykreslování se vypočítá podle počtu aktivních segmentů. Výpočet pracovní frekvence dle vzorce:

$$F = \frac{8 * F_{OSC}}{N} \quad [Hz] \qquad (2.1)$$

kde N …. Počet aktivních segmentů v řádku nebo sloupci

Současně s předešlými operacemi druhý řadič připojený na společné katody diod zajišťuje s příchodem informace na anody příslušné propojení anod se zemnící svorkou (pro uzavření elektrického obvodu).

1.2 KNIHOVNY

V infomačních technologiích je knihovna soubor, které obsahuje zdrojový kód (resp. program v daném programovacím jazyce), které vykonává určité funkce. V knihovnách jsou obvykle naprogramovány funkce pro konkrétní aplikaci (zařízení). Funkce, které knihovna vykonává, fungují na základě jejich volání v novém zdrojovém kódu, který využívá tyto knihovny. Její funkce pracují následně: Na základě volání funkce poskytne knihovna po provedení operace svá výstupní data nebo na základě vstupních dat při volání funkce provede operaci, ze které vychází výstupní data. Názvy volaných funkcí se musí shodovat s názvem funkce programované v knihovně a volají se prostřednictvím proměnné daného typu (specifická pro danou knihovnu).

1.2.1 KNIHOVNY PRO ŘÍZENÍ LED MATIC

Pro řízení LED diodových matic existují například knihovny pod názvem LED Control. Za použití knihovny lze ovládat LED maticové displeje, sedmi-segmentové LED displeje nebo jiné libovolné zobrazovací zařízení LED s maximálním počtem bodů 64. Knihovna načtená v jednom zdrojovém kódu dokáže obsluhovat současně 8 takových zařízení.

Soubory knihovny se umisťují do kořenové složky knihoven vývojového softwaru, ve kterém je psán zdrojový kód. V rámci zdrojového kódu, v němž se využívá funkcí (operací) knihovny, je nezbytné nadefinovat propojení knihovny s tímto zdrojovým kódem, a to pomocí následující syntaxe v záhlaví kódu:

#include <LedControl.h>

Knihovna se vyznačuje vlastní datovým typem *LedControl*. S použitím tohoto datové typu se zavádí 4 argumenty (vstupní data pro knihovnu).

LedControl lc = LedControl (*a, b, c, d*)

Knihovna je připravena pro sériové komunikace zařízení, ve kterém je zkompilován zdrojový kód a zařízení LED displeje. Proto postačuje použití 3 pinů pro sériový datový tok, např. pro sériovou sběrnici typu SPI.

Argumenty datového typu *LedControl a, b, c* představují výstupní pinu mikroprocesoru pro sériovou sběrnici:

a – Master output slave input MOSI (data input DIN)

b – Clock (Hodinový signál)

c – LOAD (načtení dat)

Argument *d* datového typu *LedControl* představuje adresu zařízení, do kterého jsou posílána data. Může nabýt hodnoty 0 – 7, pro maximální počet obsluhovaných zařízení 8. O konfiguraci vlastností pinů, tj. počáteční úroveň a vstup/výstup se automaticky stará knihovna. [14]

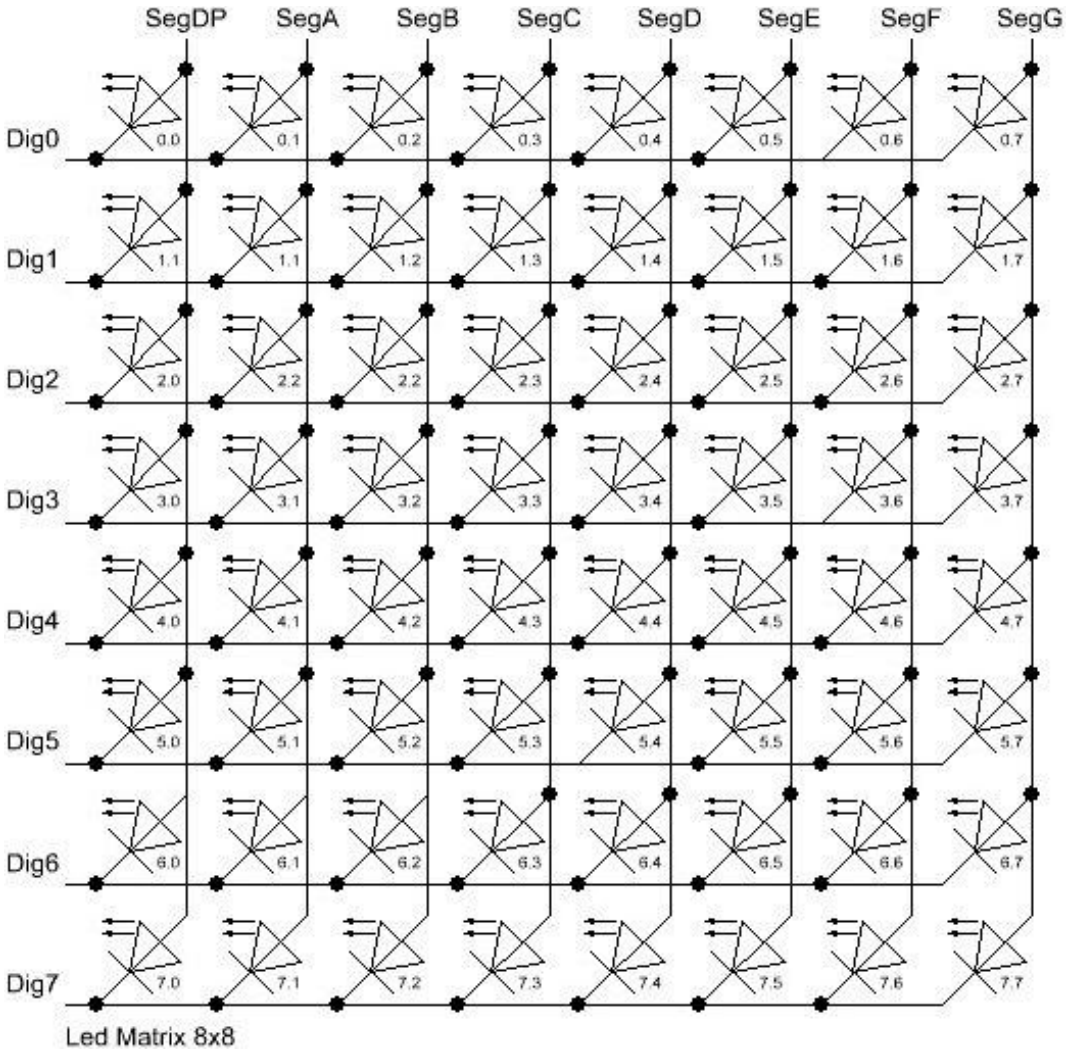
1.3 LED DIODOVÉ MATICE

Led matice je zařízení s uspořádáním LED diod do řádků a sloupců. Matice je nutno řídit procesorem. V samotné komponentě se jedná pouze o správné uspořádání a zapojení. Led diody jsou v umístěny ve dvou osách. Tvoří tudíž dvojrozměrné pole o velikosti dle počtu použitých diod. Na trhu se nejčastěji vyskytují modifikace velikosti 5x8 pixelů (diod) a 8x8 pixelů. Celé pole diod je zakomponováno do speciálního plastového pouzdra včetně vnitřních propojení a přívodů ke vstupním hřebenovým kontaktům, které jsou normovaně uspořádány ve dvou řadách z dolní strany zařízení.

Matice diod slouží například pro zobrazování obrázků, textu a jejich posuvy nebo jiné efekty. Proto je nutno je řídit procesorem pro kontrolu připojovaných pinů. Jednak kvůli kontrole (řízení) jednotlivých diod a kvůli regulaci proudu.

Na obrázku č. x lze pozorovat, že anody LED diod jsou ve sloupcích propojeny jako jeden uzel a katody jsou společné v řádcích. Z toho plyne nemožnost zobrazení celého požadovaného obrazce v jeden okamžik z důvodu rozsvícení více diod, než je požadováno. Řešením problém je rozsvěcování diod po řádcích (digitech) nebo sloupcích Segmentech s určitou frekvencí. Postupné rozsvěcování zajišťují řadiče LED displejů. Během postupného rozsvěcování je vždy pod proudem pouze jeden řádek. Celá operace probíhá s takovou frekvencí, aby lidské oko nezaznamenalo blikání LED diod. Obvykle se tyto LED diodové displeje skládají do kaskád pro rozšíření velikosti pole. Poté je jeho velikost například 8x32 pixelů nebo 16x16 pixelů. Takové varianty lze sestavit ze čtyř led diodových matic. [6]

Obr. 3 – Pohled na součástku LED diodové matice velikosti 8x8



Obr. 4 – Vnitřní schéma zapojení LED diod v matici [2]

1.4 SÍŤOVÁ KOMUNIKACE S MIKROPROCESOREM

1.4.1 ETHERNET MODUL

Ethernet modul je elektrický obvod sestavený z integrovaných obvodů a dalších komponent osazených na desce plošných spojů. To zahrnuje kontakty pro ethernetové kabely, piny pro připojení dalšího modulu apod. Zařízení je uzpůsobené pro komunikaci s MCU. Usazení hřebenových pinů je z konstrukčního hlediska nastaveno po snadné připojení do základních verzí platformy Arduino, např. model UNO. Druhou stranou komunikace je síť ethernet, resp. připojení k řídícímu prostředí, které ovládá přes lokální síť MCU ATMEL MEGA 328P. Modul je kompatibilní pro připojení ethernetového kabelu typu RJ45. Obvod vyžaduje napájecí napětí 5V (Obvod může být napájen samostatně nebo z modulu MCU).

POPIS ZAŘÍZENÍ

Hlavou Ethernet modulu je integrovaný obvod (ethernet chip) Wiznet W5100. Zajišťuje síťovou komunikaci, která může probíhat pomocí podporovaných komunikačních protokolů TCP nebo UDP. Protokol TCP je rozšířenější a spolehlivější díky své funkci „handshaking“, kde mezi klientem a serverem probíhá ověřovací komunikace, zda může být spojení navázáno a zda navázáno je. To zajistí vyšší spolehlivost obousměrného přenosu paketů mezi koncovými stanice (tedy serverem a klientem). Zařízení podporuje 4 současná socketová připojení. K zápisu programových dat ze sítě do externího procesoru jsou využity knihovny ethernetu.

WIZNET W5100

Ethernetový čip zajišťuje kontrolovaný převod dat z lokálních sítí (ethernet) do známých sériových nebo paralelních sběrnic (USB, SPI, GPIO). S nastavením registrů a operační paměti poskytuje W5100 datové připojení k síti. Pro správnou činnost výrobce doporučuje nadefinování v rámci inicializace:

Operační registry:

- Mode register
- Interrupt mask register
- Retry time-value register
- Retry count register

Registry nastavení sítě:

- Gateway Address Register
- Source Hardware Address Register
- Source Hardware Address Register
- Source IP Address Register

W5100 podporuje 4 socketová připojení současně. Socket je mechanismus zprostředkování vzdálené komunikace nebo lokální komunikace charakteru klient/server. Strana serveru zahajuje komunikaci a vytváří se nový socket s novou adresou. Jedná se o charakteristiku způsobu komunikace mezi sebou dvěma uzly.

Nezbytnou operací během inicializace je přiřazení paměti příchozím socketům ze sériové sběrnice standartu USART (R_X).

Způsoby datové komunikace:

Zařízení se vyznačuje několika způsoby síťové komunikace. Datovou komunikaci s připojenou sítí lze realizovat na základě síťových komunikačních protokolů typu TCP, UDP, IP-raw a MAC-raw.

Protokol TCP se vyznačuje pokročilým způsobem komunikace. Zřízení spojení mezi serverem a klientem probíhá s předstihem, čímž je bezpečně navázáno bez okolního rušení a následuje tok dat, který probíhá pomocí IP adres a čísel portu systému. TCP protokol se vyznačuje dobrou kvalitou poskytovaných služeb. Datová komunikace je duplexní (obousměrná). Pro ukončení navázaného spojení vysílá jedna ze stran požadavek na ukončení. Zařízení může pracovat v klient módu, kde přestavuje klienta nebo v server módu, kde se chová jako server.

Protokol UDP je spojově neorientovaný a pro přenos datagramů méně spolehlivý transportní protokol. Bez předchozího navázání spojení přenáší data, která nekontrolovaně putují topologií sítě. Přenosovou jednotkou je datagram, který se vyznačuje podobnou strukturou jako paket. Skládá se z 8bajtové hlavičky a datového prostoru. Hlavička nese informaci o IP adrese (4 bajty), použitém portu (2 bajty) a velikosti dat (2 bajty). Není zaručen příjem vyslaných dat ve správné posloupnosti a data mohou být překryta jinými a vyslaný paket zahozen. Oproti TCP se jedná o přenos s požadavkem na přenosovou rychlost. Z důvodů vysoké rychlosti nastává riziko překrývání paketů na přijímací straně, které nestíhá obsloužit všechna přijatá data. V případě použití Wiznet W5100 jsou tato rizika minimalizována, a lze tvrdit, že přenos proběhne ve většině případu spolehlivě.

Způsob příjmu dat při použití IP – raw nebo MAC – raw je totožný s protokolem UDP. Jedná se o zvláštní přístup k socketu (přímé odesílání a příjem síťových paketů).

Pro komunikaci WIZNET W5100 s mikrokontrolérem (mikroprocesorem) nabízí možnost 3 způsobů připojení, a to na rozhraní přímé nebo nepřímé sběrnice (Direct / Indirect bus I/F) a sběrnice SPI. Sběrnice SPI je kompatibilní s mnoha mikrokontroléry. [10]

Obvod ethernetového čipu je řízen sadou instrukcí, která je posílána ze strany MCU, který se na této oboustranné komunikaci označuje jako SPI master. Komunikace probíhá po zmíněné sběrnici pomocí 4 signálů:

- Slave select (/SS)
- Serial clock (SCLK – hodinový signál)
- MOSI (Master out slave in)
- MISO (Master in slave out)

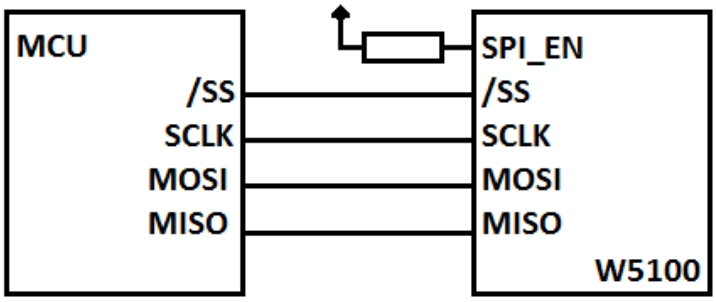
Směr datového toku signálu MOSI je z MCU do W5100 a datový tok MISO je ve směru opačném. Pin SPI_EN slouží k aktivaci komunikace po této sběrnici.

Sběrnice typu SPI pracuje ve 4 různých módech (pod čísly: 0, 1, 2, 3). Jednotlivé režimy se liší pouze fází a polaritou hodinového signálu a jak hodinový signál řídí tok dat (použití náběžné/sestupné hrany). [9]

Tab. 4 - Tabulka módů SPI sběrnice [9]

Název režimu	Polarita signálu CLOCK	Fáze signálu CLOCK
SPI_MODE0	0	0
SPI_MODE1	0	1
SPI_MODE2	1	0
SPI_MODE3	1	1

W5100 podporuje běžné režimy, tj. režimu 0 a 3. Rozdílem je pouze polarita hodinového signálu, ve stavu, kdy sběrnice není aktivní. Data jsou k příjemci načítána s náběžnou hranou a z odesílatele vysílána se sestupnou hranou.



Obr. 5 – Způsob propojení obvodu W5100 s mikrokontrolérem [10]

KNIHOVNY PRO ARUDINO ETHERNET MODUL

Knihovny pro Ethernet obsahují program v jazyce C, ve kterém je naprogramováno několik funkcí umožňující připojení k modulu k síti. Modul poté může sloužit v režimech server nebo klient. V případě serveru přijímá příchozí požadavky na spojení a v případě klienta se pokouší o připojení k serveru.

Dle použití se knihovny dělí do několika tříd. Třída ethernet zajišťuje inicializaci a navázání spojení s lokální sítí. Třída IP adress zajišťuje připojení pomocí IP adres (lokálně nebo vzdáleně). Třída serveru zajišťuje vytvoření serveru, který může vysílat a přijímat data od

připojených klientů. Třída client zajistí chování obvodu jako klient připojený k serveru, který vysílá a přijímá data od serveru a poslední třída UDP class umožňuje komunikaci (obousměrnou) s protějškem na základě UDP datagramů. [12]

1.5 PROGRAMOVACÍ JAZYK C

V současné době jeden z nepoužívanějších standartů (jazyků) pro psaní systémového softwaru a aplikací (kompilátory, operační systémy, knihovny). Jeho použití je tedy možné ve většině systémového programování, je velmi snadno čitelný (ve srovnání s jinými programy) a navíc i velmi univerzální, tedy přenositelný do jiných architektur beze změn nebo s minimálními úpravami.

V roce 1983 se American National Standards Institute (ANSI) dohodla na sestavení komise X3J11, aby byla vytvořena standardní specifikaci C. Po dlouhém a pracném procesu byl standard dokončen v roce 1989 a schválen jako ANSI X3.159-1989 „Programming Language C“. Tato verze jazyka je často stále označována jako ANSI C. V roce 1990 byl standard ANSI C (s drobnými změnami) adoptován institucí International Organization for Standardization (ISO) jako „ISO 9899|ISO/IEC 9899:1990“. Jednalo se v podstatě o vytvoření jakéhosi standartu, resp. vytvořit nadmnožinu (soubor) zahrnující mnoho vlastností jazyka doposud používané pouze lokálně.

Samotný programovací standart ANSI C je v současnu kompatibilní s většinou aplikací. Téměř každý překladač je s tímto standartem schopný pracovat. Z důvodu častého používání různých knihoven a tedy i příkazů ve zdrojovém kódu programu se někdy jazyk stávat přeložitelným pouze pro určené platformy z obecného pohledu. Z pohledu konkrétní aplikace se jedná opět o standart (například knihovny pro řízení LCD displeje se ve většině variant velmi neliší).

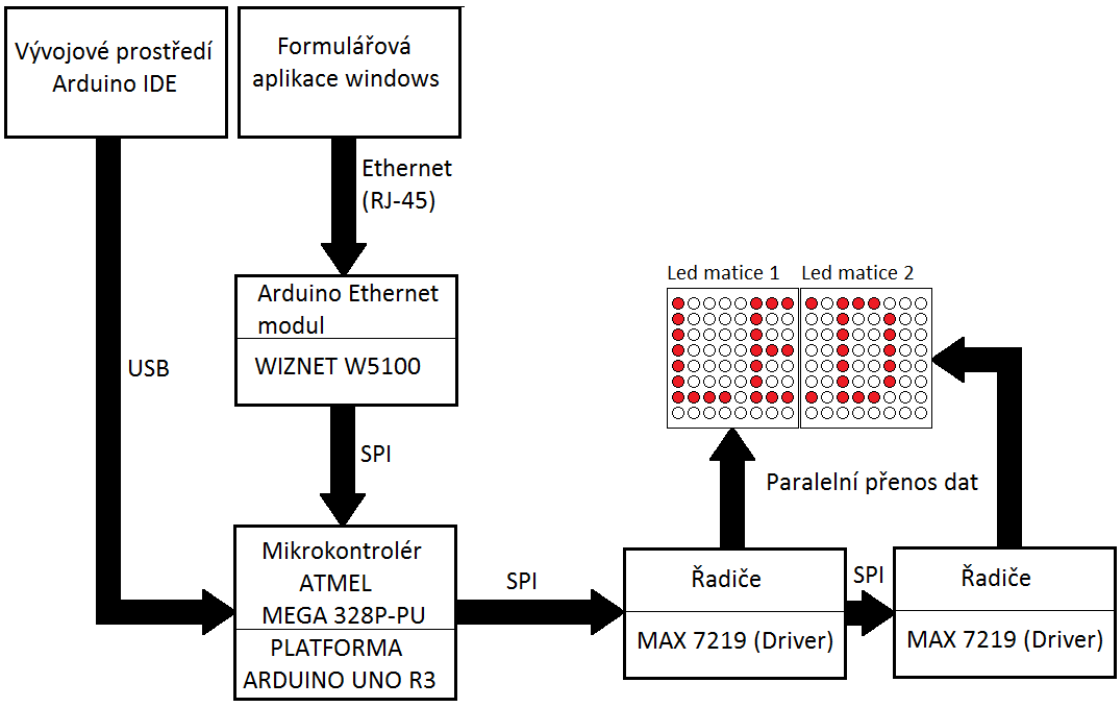
V současné dobře existuje taktéž několik rozšíření jazyka C např. jazyk C++, které úzce spojen z jazykem C. C++ je od jazyka C odvozeno. Pro dané aplikací se C++ chová striktněji a se standartním C není úplně kompatibilní. [13]

Použitím jazyka C lze provádět matematické operace, řízení procesorů nebo programování softwarových aplikací. Základem pro použití jazyka je vybrat vhodné vývojové softwarové prostřední, které podporuje překlad tohoto jazyka. Základem je nadefinování použitých knihoven pro jazyk C a příslušné operandy, se kterými se v daném programu pracuje. Následuje zápis těla programu, kde se definuje vykonávání daných funkcí pro danou aplikaci. Při správnosti se poté zdrojový kód překládá do spustitelné podoby pro procesor (kontrolér). Jedná se o převod zdrojového kódu do bitové podoby. V této podobě se program již velmi těžce upravuje. Proces kompilace zajišťuje nejčastěji vývojový software. Vývojový software obvykle umožňuje také použití funkce debug, která ve fází, kdy je program laděn kontroluje chyby ve zdrojovém kódu, které potencionálně zamezují chodu programu.

2 PRAKTICKÁ ČÁST

2.1 OBECNÝ POPIS MOŽNÉHO ŘEŠENÍ

Tato kapitola bude pojednávat navrhovaném řešení jak ovládat soustavu LED diodových matic. Použití mikrokontroléru je pro vysílání požadavků na řízení LED nezbytné. Požadavkem je, aby měl zvolený mikrokontrolér integrovanou dostatečnou statickou paměť, resp. aby byl vymezen dostatečný prostor pro uložení příkazů pro řízení LED. Pro takovéto aplikace se jeví jako vhodné mikrokontroléry z řady ATMEL MEGA nebo příp. z řady PIC od MICROCHIP. Pro svoji aplikaci jsem se rozhodl pro použití mikrokontroléru ATMEL MEGA328P. Důvodem použití je jeho integrace na platformě vývojovém platformě a snadný softwarový přístup (oproti řadě PIC jednodušší a přehlednější zdrojový kód). Pro programování platforem existuje vyvojové prostředí IDE nebo např. AVR studio. Po několika jednoduchých krocích lze zřídit rychlé propojení softwaru (vývojového prostředí) IDE s hardwarem (vývojová platforma) pomocí standartní univerzální sériové sběrnice USB. Pomocí vývojového prostředí arudino IDE předpokládám naprogramování mikrokontroléru pro řízení LED diod. Úkolem Formulářové aplikace je zasílat data do sítě, a to pomocí transportního protokololu TCP/IP, pro který je kompatibilní ethernet modul. Ten v návrhu slouží k převodu dat z ethenetu na sběrnici SPI, pomocí níž komunikuje mikrokontrolér ATMEL MEGA328P. Pro kompatibilitu s mikrokontrolérem používám řídící obvody s řadiči pro LED matice MAX7219, které jsou taktéž kompatibilní pro sériové komunikace SPI. Celá architektura je vyobrazena na blokovém schématu. [3]



Obr. 6 – Blokové schéma navrhované metody řízení LED diodových matic

2.1.1 PROGRAMOVÁNÍ MIKROKONTROLÉRU

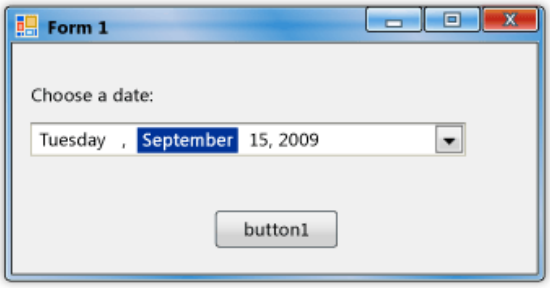
Data pro zobrazení písma a znaků na matici musí vyházet z nějakého zdroje. Tímto zdroje je mikrokontrolér. Je-li naprogramovaný, resp. je-li do něj implementován zdrojový kód, může zařízení pracovat nezávisle na propojení s počítačem. Počítačový software pouze zřizuje změnu určitých dat, tedy přepis textu, zobrazení jiného obrazce apod. Lze si pod tímto představit i úplnou změnu programu. Bez použití připojení k softwaru PC pomocí USB by proběhly pouze počáteční nastavení a potom by byly cyklicky prováděny nadefinované operace. Například zobrazení stále stejného textu. Tato operace vyžaduje však připojení napájecího napětí do platformy, resp. k mikrokontroléru.

Po připojení platformy k systému PC pomocí USB se do PC automaticky nahrají ovladače a s pomocí vývojového softwaru vzniká propojení s mikrokontrolérem. Touto cestou budou řešeny rozsáhlejší změny ve zdrojovém kódu implementovaném v mikrokontroléru.

2.1.2 FORMULÁŘOVÁ APLIKACE WINDOWS

Pro snadné uživatelské ovládání LED matice navrhuji pro běžné uživatele naprogramování jednoduché aplikace v jazyce C. Právě jazyk C volím z důvodu jeho velké univerzálnosti a přehlednosti programu. Především je snadno čitelný i amatérskými programátory. Použití funkční formulářové aplikace umožní uživateli nejjednodušší možný způsob přístupu k LED matici. Úpravy přímo ve zdrojovém kódu vyžadují více programátorských znalostí a pro uživatele se tato metoda jeví velmi neprakticky.

Formulářové aplikace pro Windows lez programovat několika způsoby. Na základě vlastních znalostí volím softwarové prostřední Microsoft Visual studio, a to především na základě dřívějších zkušeností se softwarem. V programu je možné stanovit velikosti zobrazených oken a textových polí nebo tlačítek. Těchto pár základních objektů mi postačuje k vytvoření jednoduché formulářové aplikace. Mimo možnosti nastavování vzhledu aplikace je možné jednotlivá textová pole, tlačítka apod. přímo jednoduše programovat, tedy nadefinovat, jaká operace bude vykonána např. při stisku tlačítka. Dle vlastního plánu navrhuji použití textového pole pro zápis textu, který chce uživatel zobrazit na diodové matici. V jeho blízkosti umístit tlačítko pro, při jehož stisku se vypsaný text převede do číselného formátu, který přísluší každému znaku dle standardizované tabulky ASCII. Ta obsahuje možnost zobrazení celkem 128 různých znaků. Dále by aplikace měla umožňovat takový způsob ovládání, aby byl text na matici zobrazen staticky nebo v celé délce posouván.



Obr. 7 – Příklad vzhledu formulářové aplikace [11]

2.1.3 ŘÍZENÍ MIKROKONTROLÉRU

Ve práci je navrženo řízení mikrokontroléru pomocí lokální sítě ethernet. Připojení k této síti zajistí blok ethernet modulu, který spojuje datovou sběrnici MCU s ethernetem. Za pomocí integrovaného čipu WIZNET W5100 se mohou přijímat data z lokální sítě. K tomu je vhodné použití často využívaného síťového transportního protokolu TCP/IP, který komunikuje na základě síťových IP adres. To zajistí snad propojení s osobním počítačem. Pro případné budoucí použití je řízení MCU vhodné i v případě, že zařízení matice bude situováno na špatně přístupném místě. Tímto pouhé připojení k síti (v budovách se nachází obvykle rozvody sítě) vyřeší problém propojování s počítačem na tomto místě a řízení dat se provede vzdáleně (po síti). Ethernetový převodník WIZNET W5100 propojí lokální síť s MCU ATMEL MEGA328P pomocí sběrnice SPI, pro kterou jsou obě zařízení kompatibilní. Při zasílání dat do mikrokontroléru nejsou plánovány změny architektury programu implementovaném v jeho paměti, ale pouze malé změny v definovaných proměnných příslušných datových typů pro zavolání dané funkce. V podstatě se tedy jedná jen o změny parametrů uvnitř zdrojového kódu.

2.1.4 ŘÍDÍCÍ OBVOD MAX 7219 A LED DIODOVÉ MATICE

Obvod MAX 7219 je budič přímo určený pro řízení LED diodových matic 8x8 (případně až osmičíselných sedmisegmentových displejů). Obvod používám také z důvodu integrace řadičů pro samotné LED diody přímo v MAX 7219. Nutnost řešit řízení řadičů v celé architektuře tedy odpadá. Max 7219 je kompatibilní pro sériové sběrnice. Tento fakt zjednodušuje konstrukce takovýchto a podobných zařízení. Pro komunikace s mikrokontrolérem postačují pouze 3 datové vodiče a vodič napájecí a zemnící. Mimo tato fakta, bloky s obvodem MAX 7219 je možné řadit do kaskády, což umožňuje vytvořit velikost LED diodového maticového displeje až o velikosti 8x64 pixelů. Integrovaný obvod má vstup i výstup a pro data se tedy jeví i jako průchozí. Jak vychází z možností knihoven, lze ovládat maximálně 8 zařízení, tedy 8 obvodů MAX7219. Každý řídící obvod ovládá 1 LED matici.

Na trhu je MAX7219 dostupný společně s maticí 8x8, pro kterou je MAX7219 vhodný jako řídící obvod i s řadiči. K tomu je přibalena i deska plošných spojů s vrtanými otvory. Tímto se nabízí možnost montáže obvodů do řady tak, aby strany LED matic k sobě přiléhaly a tvořili tak souvislý displej. Při tomto úkonu je nutno dodržet stejnou orientaci všech LED

diodových matic (matice nesmí být umístěna ve smyslu číslování řádků v opačné posloupnosti). V této problematice uvažuji s návrhem vlastní desky plošných spojů s vodivým motivem, který zjednoduší montáž obvodů do řady a současně zajistí snadné a přehledné elektrické připojení. Odpovídající velikost desky by měla zajistit montáž 8mi LED matic s MAX7219 do řady a také možnost montáže platformy s mikrokontrolérem a modul pro připojení k ethernetu.

2.2 POPIS VÝSLEDNÉHO ŘEŠENÍ

V rámci praktického sestrojení navrženého systému použiji navrhovaný způsob řešení. Hlavou systému je mikrokontrolér ATMEL MEGA 328P integrovaný na modulu Arduino.

Pro vývojové prostředí, které ovládá LED displej je použit standart C#, snadný pro programování jednoduchých aplikací Windows Forms. Z několika hledisek usnadňuje tvorbu formulářové aplikace (viz kap. 7.5). Propojení formulářové aplikace zajistí komunikace po lokální síti mezi PC a mikrokontrolérem zajištěná pevným ethernetovým kabelovým propojením a převodníkem sériových dat ethernetu na sběrnici typu SPI, kompatibilní pro ATMEL MEGA 328P. Knihovny pro řízení LED diod zajistí ovládání řídícího obvodu MAX 7219, resp. vysílání požadovaných rámců dle příkazů hlavního programu na datový vstup MAX 7219 (SPI sběrnice).

Kompletní ucelení je vyřešeno navrženou a vyrobenou deskou plošných spojů, která plní funkci především nosného substrátu všech modulů a částečně také propojovací pole mezi moduly (na DPS je jednostranný motiv).

2.2.1 HLAVNÍ PROGRAM

Hlavní část programu je psána ve vývojovém prostředí IDE, které obsahuje také debugger pro kontrolu chyb programu a kompilátor. Kompilátor přeloží zdrojový kód do kódu kompatibilního pro cílové zařízení. V tomto případě je cílovým zařízením MCU ATMEL MEGA 328P integrovaný na modulu. Programátor pro převod zdrojového kódu do strojového kódu zahrnuje vývojové prostředí.

PROPOJENÍ MIKROKONTROLÉRU A PC

Vývojové prostředí po překladu zasílá strojový na zvolený port sériového rozhraní USB, které propojuje osobní počítač s vývojovým model s mikrokontrolérem. Pro komunikaci prostředí s daným portem PC je třeba nastavit v programu číslo portu COM (communication port), na abychom zajistili tok dat do mikrokontroléru. Jelikož mikrokontrolér není kompatibilní pro USB data, na modulu s mikrokontrolérem se nachází také převodník USB dat na typ komunikace UART (typ převodníku CH340G). Pro UART komunikaci je kompatibilní obvod MCU ATMEL MEGA, resp. se tímto způsobem komunikace nahrávají data do paměti mikrokontroléru. Data nesou informaci o funkci, kterou má procesor vykonávat. V podstatě se skládají s příkazů, které do kontroléru vysíláme z PC. Komunikace UART používá dvě

jednosměrné datové linky Rx a Tx (tedy reciever a transmitter). Každá z nich zajišťuje komunikaci v opačném směru. Proud dat má charakter 8-mi bitových rámců, jejichž hranice tvoří start bit (úroveň log. 0) a stop bit (úroveň log. 1). USB připojení zajišťuje také funkci napájecí pro mikroprocesor. USB rozhraní přenáší napájecí napětí standardizované pro většinu digitálních systému 5V a maximální proud 500 mA. Však na modulu se nacházejí také stabilizační prvky napájecího napětí na úroveň 5 V i 3,3 V. [15]

BLOK PRO ŘÍZENÍ LED MATIC

Hlavní program lze rozlišit na 2 části. Jedna z těchto částí zajišťuje různé způsoby vypisování textu na LED diodový displej a druhá část síťovou komunikaci. Tento blok funguje na principu zápisu vstupního textu do dynamicky alokované paměti, tedy do pole dynamické velikosti. Její velikost je určena délkou vstupního textu. Zadávání vstupního textu zajišťuje ethernet (resp. samotný přenos). Proměnná *buff* dočasně ukládá data z ethernetu. Velikost řetězce *buff* je zjištěna pomocí příkazu:

len = strlen(buff);

A je uloženo do proměnné len. Tímto je určena délka textu. Jelikož se pod každým znakem datového typu *char* skrývá dekadicky číslo znaku z ASCII tabulky, je proveden převod do binární podoby znaků je proveden na základě tohoto faktu. Pro převod je nutné z určitého paměťového prostoru čerpat uložená data nesoucí informaci o bitové kombinaci. K tomuto je využíván přímo hlavní program (tedy paměť MCU). V programu se pracuje s konstantním polem data:

const byte data[] = { … bitové kombinace, …};

Zde jsou uloženy všechny bitové kombinace pro zobrazení základních znaků z ASCII tabulky. Data jsou primárně použitelná na displeji s výškou 8 pixelů (diod) v jednom sloupci. O opačném případě by pravděpodobně docházelo k neúplnému vykreslení. V hlavním programu je toto pole odřádkované takovým způsobem, aby pro uživatele působilo jako tabulka znaků (1 řádek -> 7 hodnot). Každá bitová kombinace nebo každé číslo jsou odděleny čákou (dle zápisu v jazyce C). Z tohoto vychází, že v tabulce se vyskytuje jistá pravidelnost. Tato pravidelnost má nevýhody nadbytečně zapsaných bitových kombinací v paměti, aby pravidelnost byla zachována, avšak negativním faktorem tohoto řešení je zbytečné obsazení v paměti MCU.

Strukturu jednoho řádku lze vyšetřit ze tří hledisek:

a) z hlediska čtení určitých datových typů:

   data[] = { int, int, byte, byte, byte, byte, byte, … }

b) z hlediska skutečných hodnot

   *data[] = {*4, 0, B00000100, B00101010, B00101010, B00011110, B00000000,…}

c) z hlediska použití

*data[] = {*délka znaku, výška znaku, čtyři 8-mi bitové kombinace určující obsazení rozsvěcených LED ve svislé ose, nevyužitá bitová kombinace, … }

Nevyužitá bitová kombinace je zde k doplnění konstantní velikosti řádku. Výhody stejných řádků poté využívá funkce *transfer*, která převádí ASCII znaky z řetězce *buff* do bitového pole potřebné velikosti (resp. násobí se detekovaná velikost řetězce).

Funkce *transfer* zajistí vyprázdnění pole od předchozích bitových kombinací tím způsobem, že jej naplní logickými nulami (celá potřebná velikost je zaplněna). Z důvodu pozůstalých neurčitých stavů z předchozích činností je vhodné tuto operaci provádět. V opačném případě by mohlo docházet k efektu sčítání nových a původní bitových kombinací (hardwarově) a vznikly by nežádoucí kombinace bitů v poli, z čehož plyne chybné vyobrazení. Dále funkce zajistí detekci ASCII znaku a na základě pravidelnosti „tabulky“ data přepočte polohu ASCII znaku s bitovým ekvivalentem. Dochází k detekci délky znaku (počet 8-mi bitových kombinací) a binární podoba je následně uložena do pole s názvem *pole*. Pole bytů poté představuje požadovaný text k výpisu na displej v bitových kombinacích po sloupcích včetně mezer mezi jednotlivými znaky. Zápis na displej nezahrnuje diakritiku (avšak nabízí se možnost doplnění).

```
void transfer()
    {
        int i = 0;
            for (i = 0; i < 7 * len; i++)
            {
                pole[i] = B00000000;
            }
            d = 0;
            for (b = 0; b < len; b++, d++)
            {
                for (c = 0; c < (data[(buff[b] - 32) * 7]); c++, d++)
                {
                    pole[d] = data[((buff[b] - 32) * 7) + 2 + c];
                }
            }
    }
```

V okamžiku doběhu funkce transfer je poli uložena kompletní bitová kombinace požadavku na vykreslení. Nyní s ní lze libovolně pracovat. Pro zobrazování pole jsou použity 2 funkce s názvy *static_write* a *shift_write*. Funkce *static_write* zajišťuje statický výpis textu na displej. Funkce prochází obsah pole a vypisuje dle příkazu na vykreslení řádku/sloupce na displej patřičné kombinace. Funkce *shift_write* pracuje na velmi podobné principu s tím rozdílem, že polohy vykreslování daných kombinací na displej se v čase mění, respektive posouvají se směrem vlevo (při správné orientaci displeje).

BLOK PRO KONFIGURACI ETHERNETOVÉ KOMUNIKACE

V tomto oddílu je popsána jednu z hlavních programových funkcí, a to komunikace zajišťující zasílání dat pomocí lokální síťové komunikace. Základem pro sestrojení ethernetové komunikace je funkční PC se síťovým adaptérem pro ethernet a portem pro RJ45 a modul (obvod) schopným přijímat tok dat ethernetu na straně hardwaru.

Z pohledu softwaru je nutno nastavit IP adresu modulu, aby bylo možno k němu v síti přistupovat (resp. stane se viditelným členem sítě).

Dalším krokem je určení MAC adresy zařízení. Ethernetová MAC adresa je složena ze 48 bitů a obvykle se zapisuje hexadecimálně jako šest dvojic hexadecimálních čísel. MAC adresa je důležitý identifikátor na linkové vrstvě OSI/OSI modelu. Zajišťuje doručování dat v rámci sítě. Na nových modulech se obvykle MAC adresa přikládá. V případě dříve zakoupených produktech MAC adresu zvolit.

Zadání adres IP a MAC se provede následně:

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

byte ip[] = { 192, 168, 1, 111 };

Poté je nutno nakonfigurovat typ ethernetové komunikace a na základě příkazů knihovny Ethernet. Pro tuto aplikaci je použita komunikace typu klient – server, kdy server přestavuje hardwarový modul. PC takto vysílá požadavky do serveru, který vykoná danou operaci.

Způsob zápisu:

Ethernet.begin(mac, ip); *// Na základě IP a MAC adresy je zařízení na síti k dispozici*

server.begin(); *// Nastaví ethernet modul (ethernet čip) jako server a začne naslouchat klienty*

Po předchozí deklaraci je již možno v reálném čase sledovat data na ethernetovém spojení. Následující příkazy nastaví reálné sledování ethernetové linky v čase. Opět zde figuruje vlastní datový typ knihovny ethernet EthernetClient, která popisuje klienta v síti. Do proměnné tohoto datové typu se v práci s ethernetem ukládá stav, zda je server přístupný v daném čase. Deklarace je následná:

EthernetClient client = server.available();

Je-li splněno, dostává se program do podmínky, ve které je zkoumáno, zda je připojen klient k serveru. V okamžiku připojení klienta je nadefinována lokální proměnná *char c*. Je-li klient přístupný komunikaci, začne se do proměnné c zapisovat znaky detekované na ethernetu.

Popsané akce reprezentuje následující program:

```
if (client)
{
        while (client.connected())
        {
                char c;
                if (client.available())
        {
                        c = client.read();
        }
```

Řídící aplikace ve Windows pracuje na základě navrženého protokolu. To souvisí také s nastavením portu na hodnotu větší než 1023 a zároveň menší než 2^{16}. Číslo portu určuje, která data patří ke kterému procesu. Do hodnoty portu 1023 se jedná o tzv. „*well – known*“ porty, které pracují se známými protokoly např. SMTP nebo HTTP (či HTTPS) apod.

Identifikaci příchozích informací a jejich ukončení v protokolu zajišťuje znak tabulátoru (TAB), který programově reprezentuje zápis */t*.

O příjem dat se nyní starají již popsané konfigurace ethernetu. Nyní je důležité nastavení, jakým způsobem bude programu s daty pracovat. Deklarovaný řetězec *Buff* se používá jako schránka pro přijímaná data. Na straně řídící aplikace je popsaný protokol programově vytvořen. Na straně hardwaru, resp. MCU jej detekuji.

Detekci, po stránce programové, řeším použitím stavového automatu. Pro stavový automat je opět nadefinován vlastní datový typ výčtového charakteru (tedy je přesně určeno, jakých hodnot může datový typ nabývat). Uživatel si pro nový výčtový typ sám definuje množinu přípustných hodnot uvedením všech jejich prvků. [16]

Deklarace datového typu je následná:

```
typedef enum
{
        INIT,
        TEXT,
        TYPE,
        DYNAMIC,
        STATIC,
        ERR
}TState;
```

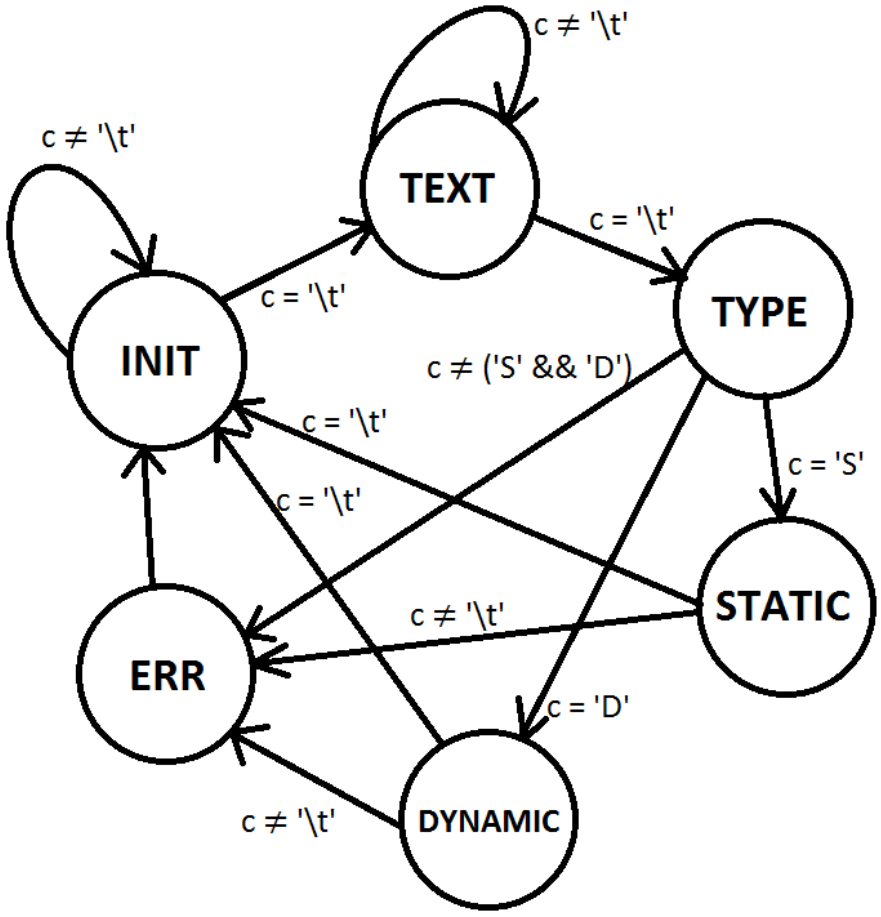
Zápis nám říká, že datový typ Tstate zvoleného názvu proměnné, může nabývat pouze hodnot z vypsaného výčtu. O totožné operace (s různými proměnnými a hodnotami) se jedná, zapíšeme-li:

int a = 1; *// datový typ **int**, proměnná **a**, hodnota **1***

nebo

*TState state = INIT; // datový typ **TState**, proměnná **state**, hodnota **INIT**.*

Stavový automat ve stavu INIT čeká na „start znak“, který detekuje začátek textu. Nepřichází-li znak */t*, stavový automat zůstává ve stavu INIT. Po detekci znaku se přesouvá do stavu TEXT. Ve stavu TEXT se do řetězce buff zapisují příchozí znaky z proměnné *c* (příchozí znaky od klienta). V tomto stavu program setrvá, dokud nedetekuje znak tabulátoru. Detekuje-li se snad tabulátoru, následně je očekávána informace o způsobu výpisu textu (staticky nebo dynamicky) ve stavu TYPE. Pro zadání způsobu je očekáván po znaku tabulátoru znak *S (staticky)* nebo znak *D (dynamicky)*. Nepřijde-li jeden z uvedených znaků, program přechází do stavu ERR (chyba). Je zavedena proměnná *sod* datového typu boolean, která nepřímo určuje jaký způsob výpisu. Stavy DYNAMIC, STATIC a ERR, kterými může stavový automat ještě projít než se dostane zpět do stavu INIT slouží v programu spíše k ověření právnosti funkce (ke zjištěná, zda se automat vyskytuje ve stavu, který požaduje uživatel). Ověření je možné způsobem výpisu textu na sériovou linku.



Obr. 8 – blokové schéma stavového automatu pro detekci dat z ethernetu

2.2.2 POUŽITÉ KNIHOVNY HLAVNÍHO PROGRAMU

K automatickému nastavení pinů, jejich charakteru (vstup, výstup), způsobů čtení dat z těchto PINŮ nebo zasílání dat a případné závislosti dohromady tvořící funkce slouží programové knihovny. Tyto knihovny lze dohledat na internetu nebo si jej uživatel může vytvořit sám. V tomto případě jsou použity dostupné knihovny. Základní požadované funkce všechny knihovny obsahují a je možné, ale ne nezbytné dělat jejich úpravy nebo případně psát vlastní knihovnu. To je důvodem jejich použití při tvorbě projektu.

LED CONTROL

Knihovna LED control se stará o řízení LED diodových driverů MAX 7219. Na tento obvod se příkazy vysílají jako sériová data. Postupně přicházející příkazy zadávají do driveru způsob vykreslení na LED matici, resp. polohu řádku nebo sloupce a příslušnou bitovou kombinaci. Knihovna dokáže současně ovládat 8 zařízení MAX 7219 (každý přísluší k jedné LED matici 8x8). Knihovna LED control zahrnuje především programovou konfiguraci módů, registrů a způsob příjmu dat po sběrnici SPI. Mimo jiné zajistí také konfiguraci pinů MCU a nadefinuje chování sběrnice (na příslušných I/O pinech MCU). Pro inicializaci sběrnice je nutno nadefinovat čísla pinů pro sběrnici SPI. Tato podmínka se plní na základně vlastního datového typu knihovny LED Control, který nakonfiguruje také počet použitých zařízená (modulů s Maticí 8x8). Pro řízení LED diod pomocí MAX7219 využívám následující příkazy z knihovny LedControl:

Nastavení sběrnice SPI pro MAX 7219

LedControl lc = LedControl(int (data), int (clk), int (cs), int (dev));

data … číslo pinu pro přenos SPI DATA
clk … číslo pinu pro přenos SPI CLOCK (hodinový signál)
cs … číslo pinu pro přenos SPI LOAD
dev … počet řízených driverů MAX 7219

Nastavení intenzity

lc.setIntensity(int (addr), int (Intensity));

addr … číslo zařízení, se kterým pracujeme

Intensity … hodnota intenzity svícení LED v 16 úrovních (viz řízení intenzity pomocí MAX7219)

Vymazání současných dat pro displej

lc.clearDisplay(int (addr));

addr … číslo zařízení, se kterým pracujeme

Ovládání sloupce

lc.setColumn(int (addr), int (col), byte (value));

addr … číslo zařízení, se kterým pracujeme (hodnota 0 - 7)

col … číslo sloupce (hodnota 0 - 7)

value … rozsvícení diod pomocí bitové kombinace (např. B01110101)

Ovládání řádku

lc.setRow(int (addr), int (row), byte (value));

addr … číslo zařízení, se kterým pracujeme

row … číslo řádku (hodnota 0 - 7)

value … rozsvícení diod pomocí bitové kombinace

ETHERNET

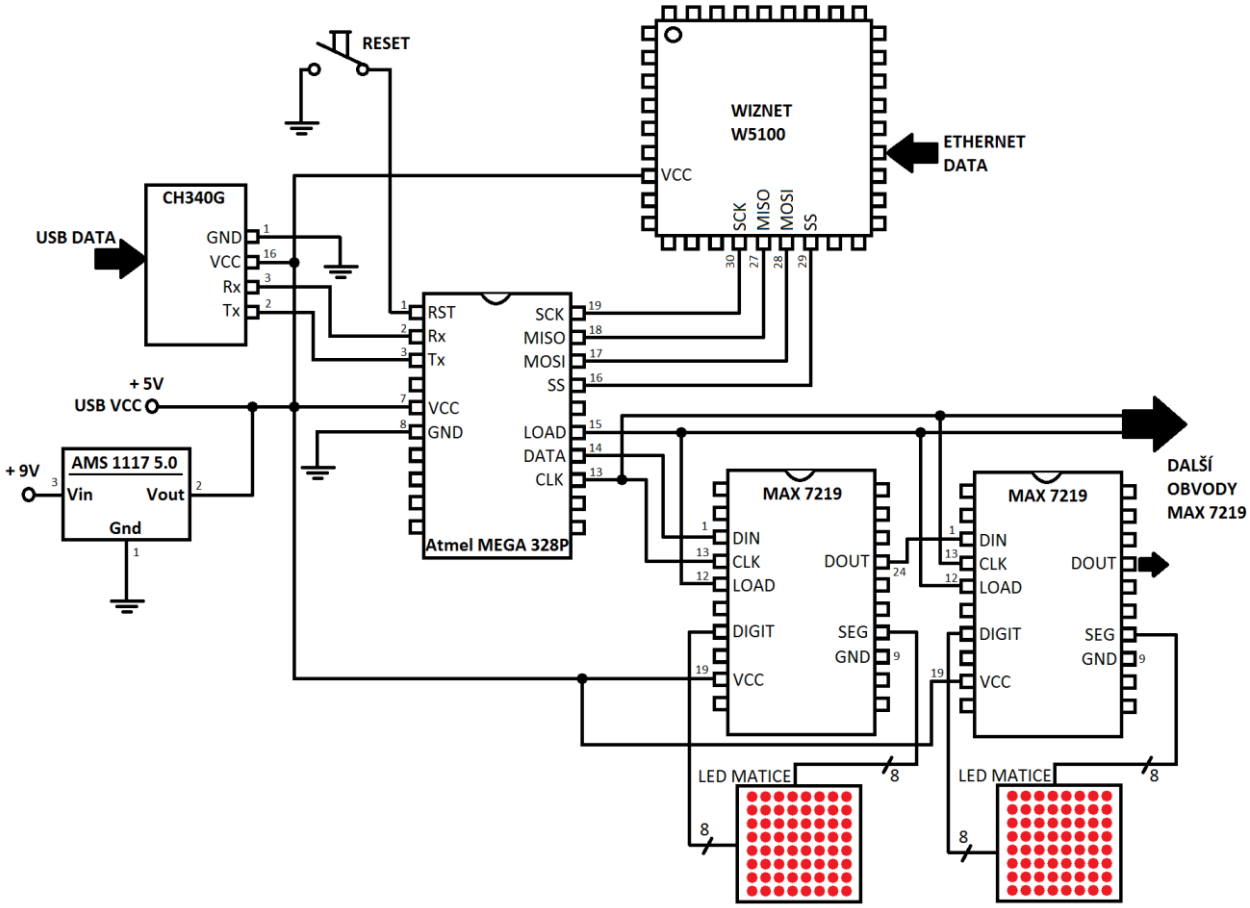
Primární komunikace s mikrokontrolérem probíhá po USB. Avšak doba komunikace probíhá řádově v sekundách s převodem na standart UART a nahrání do MCU. Mimo jiné je tato cesta nevýhodná z hlediska přepisování nahraného strojového kódu v MCU. Tímto by se případné řízení LED displeje jevilo jako velmi nepraktické. Proto je výhodné použití ethernet shieldu s rychle pracujícím převodníkem WIZNET W5100, který vykonává převod ethernet data/SPI v reálném čase. Převod probíhá i v případě, že na sériové lince ethernetu nejsou žádná užitečná data. Na výstupu převodníku jsou tedy data poté k dispozici a je na uživateli, zda s nimi bude pracovat (tedy jaké podmínky nebo identifikátory určují přítomnost užitečných dat na sériové lince).

SPI

Použití knihovny není pro funkci nezbytné, avšak pouze výhodné při tvorbě programu. Na ethernet dokáže samostatně obsloužit ethernetovou komunikaci, včetně komunikace mezi převodníkem na SPI a MCU. Zasílání dat na sériovou linku nám umožňuje kontrolu při inicializaci ethernetu. Například je možné kontrolovat nadefinovanou IP adresu (případně i MAC adresu) zařízení. Při změně stavu na ethernetu nebo například v případě, že na ethernet přichází různé bloky dat lze pomocí sledování sériové linky pomocí sériového monitoru v prostředí IDE správnost příchozích dat. Tato data mohou představovat jak text (typ CHAR), tak číslo (datový typ INT aj.). Při dalším ladění programu je praktické použití této funkce pro kontrolu výstupních veličin funkcí postavených na základě cyklů (cyklicky je tedy zapisována daná hodnota do daného objektu programu).

2.2.3 SCHÉMA ZAPOJENÍ

Kapitola popisuje blokový popis zapojení celého systému s číly pinů jednotlivých bloků. Jednotlivé bloky obvykle představuje intergrovaný obvod nebo modul s danou součástkou. Bloků MAX 7219 s příslušnou LED diodovou maticí je v obvodu zapojeno 8. Zapojení nevyznačených bloků se neliší od zapojení mezi první na druhým blokem naznačeným na obrázku schématu zapojení.



Obr. 9 – Schéma zapojení celého systému

2.2.4 ŘÍDÍCÍ APLIKACE

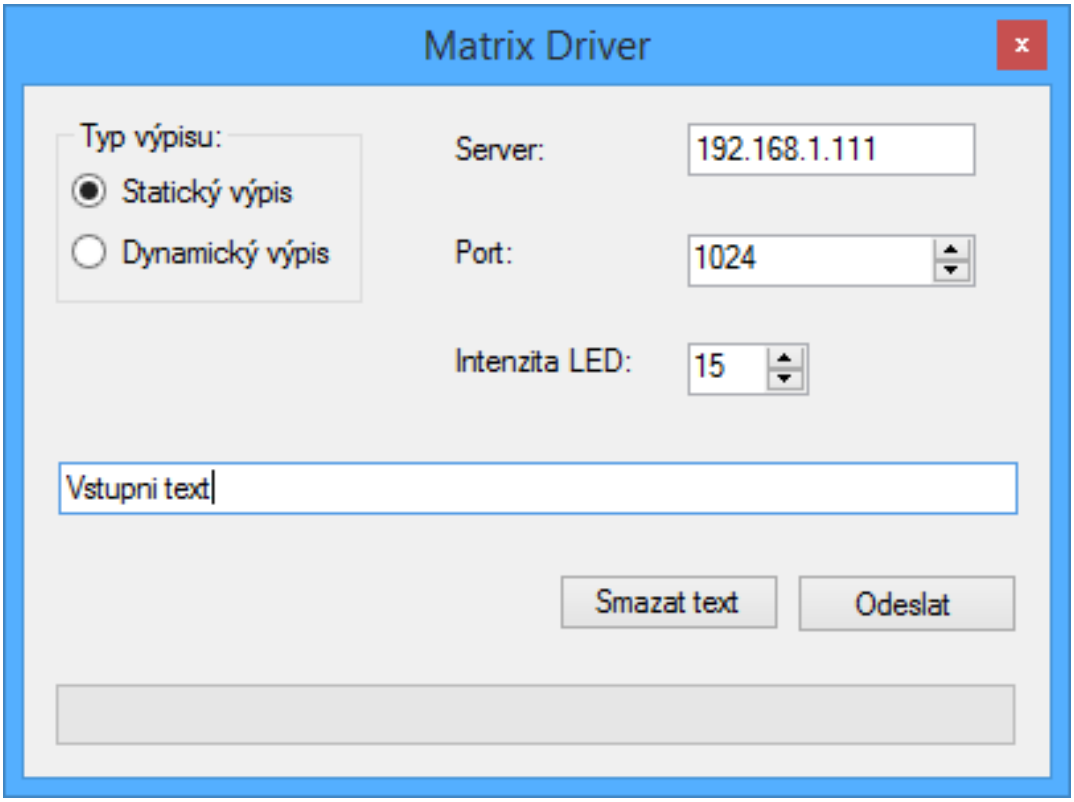
Pro vytvoření formulářové aplikace používám program Microsoft Visual Studio 2013. Pro vytvoření formulářové aplikace, v podstatě dialogového okna windows s možností zápisu vstupního textu, je použit objektově orientovaný jazyk C#. Tento jazyk programově velmi ulehčuje programování tlačítek, textových polí a podobných prvků formulářové aplikace. Použití klasického jazyka C by v tomto úkolu bylo velmi neefektivní. V takovém případě bychom dospěli k velmi složité programové části a málo funkcí výsledné aplikace. Tento nepoměr se v současné době vymyká trendům.

Objektové programování je založeno na použití programových objektů, které jsou již vyvinuty. K objektům se přistupuje na základě jejich funkce, vstupních a výstupních veličin

(parametrů). Pro objektové programování není podstatná vnitřní struktura již naprogramovaných struktur. Jedná se o poněkud moderní náhled na to, jakým způsobem programovat. Na kvalitně vytvořených a znovupoužitelných základech se staví objektově orientovaný program, strukturovaný odlišně od dřívějších metod psaní programu. [17]

Pro programování řídící aplikace ve windows v programu MS Visual studio využívám nepoužívanější soubor funkcí a příkazů verzi NET. Framework 3.5. Visual studio je výhodné použít z důvodu detekcí práce mimo hranice pole, špatné přetypování proměnné, nabízí se při zápisu zdrojového kódu použitelné funkce a proměnné. Souhrn těchto a mnoha dalších faktorů z pohledu programátora značně zjednoduší vývoj nové aplikace.

V hlavní programové části řídící aplikace je naprogramováno chování tlačítek, textového pole, získání stavu horního přepínače (static, dynamic) apod. Desing hlavního okna je viditelný na obrázku číslo x. Textové pole slouží k zadání vstupního textu pro výpis na displej. Tlačítko smazat text vymaže zadaný text z textového pole (nikoliv z displeje). Tlačítko odeslat vydává příkaz k použití naprogramovaného vlastního protokolu a dle tohoto zasílá data na síťovou kartu PC a adaptér se stará o vysílání na ethernet. Řídící aplikace zahrnuje také nastavení. Lze na ní nastavit port, na které bude probíhat komunikace. Další nastavitelným argumentem je IP adresa severu, ke kterému se klient připojuje. Nastavení portu musí být shodné s portem konfigurovaným v inicializaci ethernetu v MCU. IP adresa nesmí být shodná s IP adresou serveru, jinak nedojde ke spojení. Každý prvek v síti musí mít vlastní jedinečnou IP adresu.



Obr. 10 – Grafický vzhled řídící formulářové aplikace ve Windows

PROTOKOL PŘENOSU DAT PŘES ETHERNET

Strana klienta sestavuje vysílání dat dle protokolu. Strana MCU (hardwarová) přijímá data, resp. detekuje data dle protokolu. Strana klienta sestavuje protokol ve třech funkcích. První funkce vytváří TCP socket pro síťovou kartu. Je nutno také připojit se ke streamu (toku dat) na síť. Toto již řeší druhá funkce společně s vytvořením protokolu pro přenos dle následujícího příkazu:

Byte[] field = System.Text.Encoding.ASCII.GetBytes("\t" + text + "\t" + type.ToString() + "\t");

Do pole *field* se zapíší všechny bitové kombinace, což zahrnuje kombinací pro tabulátory a kombinaci přetypovaného vstupního textu. Po zpracování síťovou kartou bity proudí sériově do cílového hardwaru.

Poslední funkce ukončuje TCP spojení a odpojuje program od streamu. V sekci main programu řídící aplikace je nutno protokol pro správnou funkci volat.

2.2.5 NÁVRH DPS

Návrh desky plošných spojů pro maticové displeje vzešel z myšlenky ucelit všechny moduly. Obvod není navržen jako komplet. Je složen z modulů a mechanické ucelení sestavy řeší deska plošných spojů jako substrát.

Moduly s LED maticemi a řadičem mají vývody ze dvou stran po pěti pinech pro připojení tzv. *„jumper wire“*. Vývody zahrnují na obou stranách Napájení i zem. Avšak napájecí a zemnící svorka je modulem průchozí. Datová sběrnice musí procházet obvody postupně, jelikož je postupně zpracovávají řadiče. Pro napájecí svorky ovšem dostačuje připojení z jedné strany modulu. To je důvodem montáže pinů s pěti vývody na straně desky, na níž je vedena síť napájení. Na výstupní části sběrnice SPI jsou použity pouze 3-pinové vývody (druhá strana modulů). Toto propojení dle návrhu schéma zapojení propojuje výstupní port modulu se vstupním následujícího. S horní strany (TOP) desky plošných spojů jsou montovány všechny moduly. Na desce je ze strany TOP také potisk (obrysy modulů, názvy hřebenových pinů). Ze spodní strany (strana BOTTOM) se nachází vodivý motiv. Ze spodní strany je mimo vodivé cesty rozlévaná měď sítě GND. Konektory jsou pájené taktéž ze strany BOTTOM. Potisk z horní strany je bílé barvy. Povrchová úpravy strany s motivem je používaný standart OSP (organic surface protection). Tloušťka materiálu (FR4) je pro velké rozměry desky (375 x 100 mm) nestandartní, a to 3,2 mm.

Pro návrh desky plošných spojů jsem využil mezi návrháři známý program Eagle (verze 7.2). Program je uživatelsky snadno ovladatelný a také defaultně zahrnuje kvantum knihoven elektronických součástek a dalších elektronických komponent pro osazování a také jejich popis. Model součástky se skládá z popisu pinů a jejich názvů a grafický popis o rozměrech součástky. Pro návrh potisku na straně TOP používám knihovnu s vlastní součástkou, která nemá definovány žádné piny, pouze vlastní obrys. Tímto způsobem lze obejít mnoho rozměrových výpočtů o poloze děr pro mechanické uchycení modulů. Vrtané otvory pro uchycení jsou průměru 3 mm (to je standartní velikost děr pro uchycení DPS).

3 ZÁVĚR

V první části práce jsou shrnuty nejpodstatnější informace o řídícím obvodu LED diodových matic MAX7219, co jsou to LED diodové matice a jak fungují a základní informace o programovacím jazyce C. Okrajově je taktéž popsáno, jak zajistit propojení mikrokontroléru se síti, a které prvku k tomuto účelu slouží. Bylo zjištěno, že realizace je možná pomocí obvodu ethernetového převodníku z dat ethernetu na danou sériovou sběrnici. Tyto požadavky splňuje například ethernetový čip WIZNET W5100. Stručně je vysvětlena také duplexní komunikace pomocí sběrnice SPI (MOSI/MISO) mezi obvodem WIZNET W5100 a MCU.

Pro řízení celého systému (hlava systému) je použit modul s mikrokontrolérem ATMEL MEGA 328P. Tento mikrokontrolér je programovatelný pomocí sériové sběrnice USB. Jeho programovatelným vstupem je standart UART, proto je použit celý modul s MCU, na kterém se nachází také převodník dat USB na UART.

Vzhledem ke kompatibilitě MCU pro sběrnici SPI, byl pro řízení matic vybrán řídící obvod s proudovými budiči pro jednotlivé LED diodové segmenty matice MAX7219, který je na svém vstupu kompatibilní pro komunikaci po sběrnici SPI. V tomto bloku systému funguje SPI jako jednosměrná komunikace z MCU do MAX 7219. Tento obvod umožňuje řízení 8x8 segmentů, proto je v systému použit 8krát, vždy s příslušnou diodovou maticí. Tímto je splněn požadavek na velikost displeje (počtu pixelů) 8x64.

Pro ucelení systému a mechanické upevnění LED matic do správné polohy tak, aby jejich pravý a levý okraj na sebe vždy navazoval je navržena a vyrobena deska plošných spojů, která plní funkci nosného substrátu a propojovací struktury mezi jednotlivými moduly s LED maticemi.

Jelikož komunikace mikrokontroléru přes USB je relativně pomalá, je požadováno, aby řízení LED diodového displeje a zasílání dat do MCU z počítače probíhalo v reálném čase. K docílení je použit ethernetový modul s WIZNET W5100, který zajistí komunikaci systému po lokální síti. K přenosu dat po ethernetu je navržen komunikační protokol pracující na bázi detekce tzv. bílých znaků ASCII tabulky, které zajišťují oddělení dat. Protokol v MCU definuje, jak s daty pracovat. Pro vytvoření řídící byl zvolen objektově orientovaný programovací jazyk C#, z důvodů jeho užívání k tvorbě formulářových aplikací. Zmíněný komunikační protokol ethernetu vytváří tato řídící aplikace.

Celkovým blokovým sestavením je vytvořen systém pro řízení LED diodových matic v reálném čase po lokální síti.

POUŽITÉ ZDROJE

[1] MAXIM INTEGRATED. *MAX 7219/7221: Datasheet* [online]. USA: Maxim Integrated, 2003 [cit. 2014-12-16]. Dostupné z: http://datasheets.maximintegrated.com/en/ds/MAX7219-MAX7221.pdf

[2] ARDUINO. *Learning: Playground* [online]. Italia: Creative Commons, 2014 [cit. 2014-12-16]. Dostupné z: http://playground.arduino.cc/uploads/Main/LedMatrix.jpg

[3] IJACEK.007. Arduino: Arduino MAX7219 8x8 Matrix LED Displej. In: IJACEK.007. [online]. http://www.nozo.cz/, 2014 [cit. 2014-12-16]. Dostupné z: http://blog.ijacek007.cz/Arduino/arduino-max7219-8x8-matrix-led-displej/

[4] Wikipedie, *Demultiplexor* [online], Wikipedie: Otevřená encyklopedie, c2013, Datum poslední revize 23. 10. 2013, 11:00 UTC, [citováno 16. 12. 2014]. Dostupné z: http://cs.wikipedia.org/w/index.php?title=Demultiplexor&oldid=10881263/

[5] MICRODESIGNUM. *Polovodičové lasery a LED-ky: Úvod* [online]. ČR: Web Designum, 2007 [cit. 2014-12-16]. Dostupné z: http://www.microdesignum.cz/clanky/Polovodicove-lasery-a-LED-ky.html

[6] PANDATRON. *LED matice 8 řádků a 8 sloupců* [online]. Elektrotechnický magazín. ČR: Pandatron.cz, 2000, 2013 [cit. 2014-12-16]. ISSN 1803-6007. Dostupné z: http://pandatron.cz/?3862&led_matice_8_radku_a_8_sloupcu/

[7] TUUPOLA, Mika. LEANPUB AFFILIATE PROGRAM. *How does LED matrix work?* [online]. Designmod, 2014 [cit. 2014-12-16]. Dostupné z: http://www.appelsiini.net/2011/how-does-led-matrix-work

[8] GM ELEKTRONICS. *LED DISPLAY MATICOVÝ 32X32MM RG HD-M10EG88MD* [online]. Eshop. ČR: GM electronic, spol. s.r.o., 2014 [cit. 2014-12-16]. Dostupné z: http://www.gme.cz/led-display-maticovy-32x32mm-rg-hd-m10eg88md-p512-178/

[9] ARDUINO. *SPI library: A Brief Introduction to the Serial Peripheral Interface (SPI)* [online]. Italia: Creative Commons, 2014 [cit. 2014-12-16]. Dostupné z: http://arduino.cc/en/Reference/SPI

[10] WIZNET CO. *W5100: Datasheet* [online]. USA: WIZnet Co., 2008 [cit. 2014-12-16]. Dostupné z: https://www.sparkfun.com/datasheets/DevTools/Arduino/W5100_Datasheet_v1_1_6.pdf

[11] MICROSOFT, a.s. *Vytvoření Formulářová aplikace Windows pomocí technologie .NET Framework (C++): Vytvoření nového projektu modelu Windows Forms* [online]. Microsoft developer network. USA: Microsoft, a.s., 2014 [cit. 2014-12-16]. Dostupné z: http://msdn.microsoft.com/cs-cz/library/ms235634(v=vs.100).aspx

[12] ARDUINO. *Ethernet library* [online]. Popis knihoven. Italia: Creative Commons, 2014 [cit. 2014-12-16]. Dostupné z: http://arduino.cc/en/Reference/Ethernet

[13] Wikipedie, *C (programovací jazyk)* [online], Wikipedie: Otevřená encyklopedie, c2014, Datum poslední revize 27. 10. 2014, 10:54 UTC, [citováno 16. 12. 2014]. Dostupné z: http://cs.wikipedia.org/w/index.php?title=C_(programovac%C3%AD_jazyk)&oldid=11953327/

[14] ARDUINO. *LedControl: Creating a LedControl* [online]. Italia: Creative Commons, 2014 [cit. 2014-12-17]. Dostupné z: http://playground.arduino.cc/Main/LedControl#Setup/

[15] USART. UART, RS232, USB, SPI, I2C, TTL, etc. what are all of these and how do they relate to each other? *Electronics Stack exchange* [online]. Site desing [cit. 2015-06-02]. Dostupné z: http://electronics.stackexchange.com/questions/37814/usart-uart-rs232-usb-spi-i2c-ttl-etc-what-are-all-of-these-and-how-do-th/

[16] *C/C++: Učíme se C (26. díl) - Datové typy enum a union* [online]. [cit. 2015-06-02]. Dostupné z: http://www.builder.cz/rubriky/c/c--/ucime-se-c-26-dil-datove-typy-enum-a-union-155821cz/

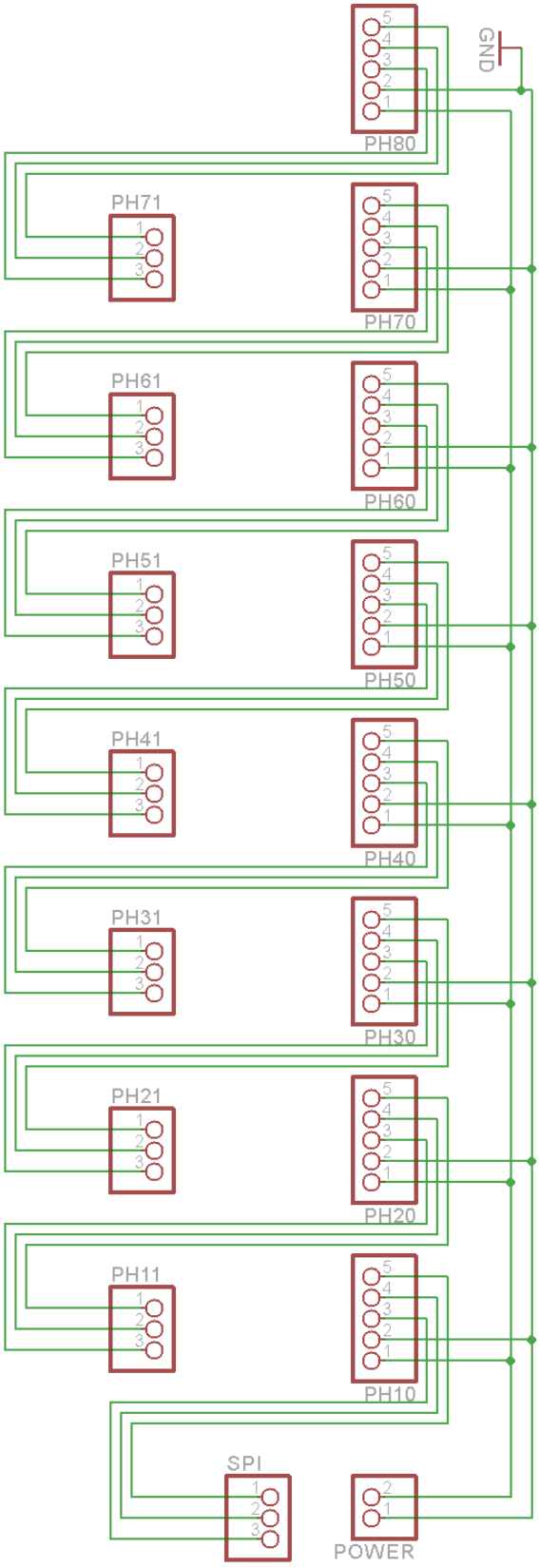
[17] ČÁPKA, David. *Úvod do objektově orientovaného programování v C#* [online]. ČR: itnetwork.cz, 2012 [cit. 2015-06-02]. Dostupné z: http://www.itnetwork.cz/c-sharp-tutorial-uvod-do-objektove-orientovaneho-programovani/

SEZNAM SYMBOLŮ A ZKRATEK

Zkratka	Vysvětlivka	Definice
MCU	Microcontroler unit	obvod/blok mikrokontroléru
SPI	Serial peripheral buse	Rozhraní sériové sběrnice
LED	Light emitting diode	Dioda s PN přechodem emitujícím světlo
MS	Microsoft	Společnost poskytující služby v oblasti IT
PC	Personal computer	Osobní počítač
USB	Universal serial buse	Universální sériová sběrnice
MAC	Media access control	Identifikátor síťového rozhraní
IP	Internet protocol	adresa, která identifikuje síťový prvek
UART	Universal asynchronous Rx/Tx	Sériová sběrnice s asynchronním přenosem
TCP	Transmission control protocol	Síťový protokol s garancí spolehlivosti přenosu
UDP	User datagram protocol	Síťový protokol bez garancí spolehlivosti přenosu

PŘÍLOHY

NÁVRH DPS – SCHEMATIC



NÁVRH DPS – BOARD