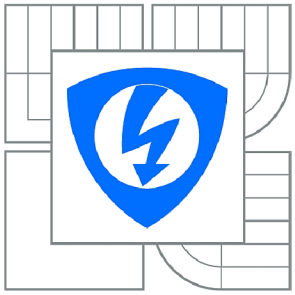
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

GENEROVÁNÍ A ZOBRAZENÍ QR KÓDŮ NA EMBEDDED GRAFICKÉM OLED DISPLEJI

EMBEDDED QR CODE GENERATOR WITH A GRAPHIC OLED DISPLAY

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

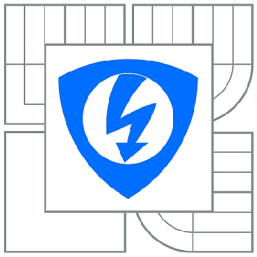
AUTOR PRÁCE — Bc. TOMÁŠ LAKOMÝ

AUTHOR

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

SUPERVISOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Tomáš Lakomý	***ID:***	139285
Ročník:	2	***Akademický rok:***	2014/2015

NÁZEV TÉMATU:

Generování a zobrazení QR kódů na embedded grafickém OLED displeji

POKYNY PRO VYPRACOVÁNÍ:

Vytvořte zařízení s běžným mikrokontrolérem (pokud možno Atmel AVR či AVR32), které dokáže z RS-232 sběrnice přijmout textový (ASCII) řetězec, vygenerovat z něj QR kód a ten následně zobrazit na grafickém OLED displeji. Displej bude typu Densitron DD-128128FC-6A s rozlišením 128x128 pixelů, proto se zabývejte pouze QR kódy do velikosti, které na něm lze ještě zobrazit. Nalezněte vhodné knihovny a stanovte výpočetní a paměťovou náročnost generování takových QR kódů. Následně vyberte mikrokontrolér, který těmto požadavkům s přiměřenou rezervou vyhoví, pokud možno bez potřeby externích pamětí. Realizujte funkční vzorek napájený z 24 V, včetně návrhu a osazení plošného spoje. Při návrhu se snažte, aby byl celý plošný spoj skrytý pod displejem.

DOPORUČENÁ LITERATURA:

[1] Richard Barnett, Embedded C Programming and the Atmel AVR, 2nd edition. Cengage Learning, 2006. ISBN 978-1418039592.
[2] Carolyn Eby, QR Code Tutorial. K dispozici online na http://www.thonky.com/qr-code-tutorial/.
[3] Katalogový list Densitron DD-128128FC-6A.

Termín zadání:	9.2.2015	***Termín odevzdání:***	26.5.2015

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

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

UPOZORNĚNÍ:

ABSTRAKT

Tato diplomová práce se zabývá kódováním informací a jejich generování do QR kódů. Pro tento účel je v práci popsán princip fungování QR kódů a také návrh zařízení. Toto zařízení je schopno zakódovat informaci přijatou přes sériovou linku a následně ji zobrazit v QR kódu na OLED displeji. Zařízení obsahuje mikroprocesor, proto je zde nastíněn postup při programování a testování zařízení.

KLÍČOVÁ SLOVA

AVR, generování, kódování, QR kód, DPS, OLED displej

ABSTRACT

This master's thesis deals with coding information and QR codes generating. There is described operation principle of QR codes for this purpose, and design of the equipment. This device is able to encode the information received through the serial port and display it in the QR code on the OLED display. Microprocessor is part of the device, so there is outlined the procedure for programming and testing equipment.

KEYWORDS

AVR, generating, coding, QR code, PCB, OLED display

LAKOMÝ, T. *Generování a zobrazení QR kódů na embedded grafickém OLED displeji*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2015. 56 s. Vedoucí diplomové práce: Ing. Pavel Hanák, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma Generování a zobrazení QR kódů na embedded grafickém OLED displeji jsem vypracoval samostatně pod vedením vedoucího diplomové 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 diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové 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 ....

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce Ing. Pavlu Hanákovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne ....

(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

QR kódy jsou dvojrozměrné čárové kódy, které jsou nejvíce rozšířeny v zemi svého vzniku, tedy v Japonsku, ale v poslední době už jsou často používané i u nás. Slouží k rychlému načtení zakódované zprávy čtecím zařízením.

Nalézají se na různých reklamních letácích, billboardech či výrobcích. Jejich přečtení a dekódování je pro uživatele s čtecím zařízením velmi jednoduché a rychlé. Zkratka „QR“ pochází od vzniku z anglického „quick response“, což v doslovném překladu znamená „rychlá odpověď“.

Základem jsou černé a bílé bloky, ze kterých jsou vytvořeny obrazce, které rozpoznává čtečka, která nejčastěji bývá jako aplikace v mobilním telefonu nebo tabletu. Aplikace pro čtení QR kódu jsou zdarma a začínají být standardem výbavy předinstalovaného SW mobilních zařízení.

Je to také způsob, jak informace jednoduchým způsobem přenést do mobilního zařízení, přičemž nezáleží na tom, co je v QR kódu zakódováno.

Nejčastěji se kódují odkazy na webové stránky, telefonní čísla, GPS souřadnice nebo kódy na obalech potravin.

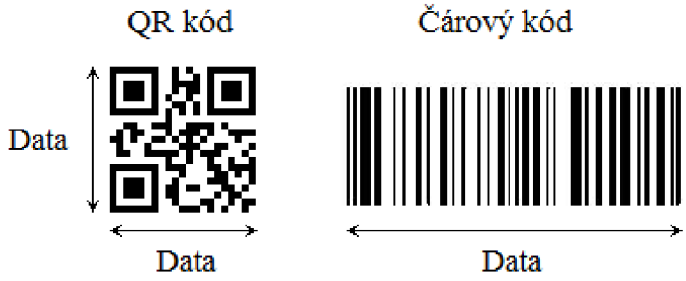
QR kódy se používají i v oblasti cestovního ruchu. Jsou jimi např. označena významná místa, turista si tak může jednoduše zobrazit bližší informace o památce, místu či významné osobnosti.

Novodobé využití QR kódu nabízí široké možnosti, jakou je například moderní QR platba.

Kromě samotných dat jsou však zakódovány ještě další informace, které nenesou zakódovaný text, ale prozrazují QR čtečce důležité informace, které potřebuje k dekódování. Je to nejen korekce chyb, ale také informace o verzi symbolu, použitém standardu a další dodatečné informace.

1 QR KÓDY

QR kód je nová verze klasického čárového kódu. Na rozdíl od něj ale obsahuje informace jak ve vertikálním, tak i v horizontálním směru (viz. Obrázek 1). Tím dokáže zakódovat několikanásobně více informací. QR kódy nejsou tvořeny svislými čárami, ale černobílými bloky (černé body představují logické jedničky, bílé logické nuly), které se skládají ve čtvercové obrazce. Oproti klasickému čárovému kódu nejsou QR kódy určeny pouze pro jednoznačné označení zboží, neobsahují tedy předem definované informace s danými pravidly. QR kódy slouží pro uložení a následně rychlé načtení jakékoliv informace. [1]



Obrázek 1: Rozdíl mezi QR kódem a čárovým kódem

1.1 Historie QR kódů

Už ve čtyřicátých letech se v USA objevily první pokusy s čárovými kódy, kdy studenti Harvardovy univerzity prováděli pokusy a pomocí čárových kódů sledovali skladové zásoby.

O podobu, jakou mají čárové kódy dnes, se zasloužili Američané Bernard Silver a Norman J. Woodland. Tečky a čárky Morseovy abecedy vertikálně protáhli a roku 1952 patentovali.

V sedmdesátých letech se čárové kódy začaly používat pro označování zboží. Výrobky však nebyly kódy označeny již z výroby, ale prodavači tyto kódy museli na výrobky lepit stejně tak, jako se zboží v obchodech dříve označovalo cenovkami. Používání čárových kódů ve větší míře došlo v USA až v 80. letech. Ke standardizaci dochází až v posledních třiceti letech v souvislosti se vznikem a rozvojem mezinárodní organizace GS1, která se stará o rozvoj příslušných standardů a hlavně registruje do systému čárových kódů, čímž je umožněno jejich mezinárodní jednoznačné využití. Tato organizace, která má pobočky ve sto zemích světa, poskytuje i poradenské a konzultační služby.

QR kód, jako nástupce čárového kódu byl vytvořen až v roce 1994 japonskou firmou Denso-Wave, dceřinou společností firmy Toyota, která je známá výrobou v automobilovém průmyslu. Standard QR kódu je definován v normě ISO/EIC 18004: 2006. K používání těchto kódů není potřeba žádná licence. [1]

1.2 Typy QR kódů

1.2.1 QR kód Model 1 a Model 2

Model 1 je originální QR kód. Poslední verzí tohoto kódu je verze 14 (73x73 bodů), který je schopen uložit až 1167 číslic. Model 2 je vylepšená verze Modelu 1 s největší verzí 40 (177x177 bodů) který je schopen uložit až 7089 číslic. Ve většině případů se QR kód používá v modelu 2. [8]

Obrázek 2: QR kód Model 1 a Model 2 [8]

1.2.2 Mikro QR kód

Hlavním rozdílem Mikro QR kódu je, že obsahuje jen jeden kotvící bod, na rozdíl od normálního QR kódu, který má kotvící body umístěny ve třech rozích symbolu. Navíc QR kód vyžaduje alespoň čtyři body široký prostor kolem symbolu, zatímco pro Mikro QR kód stačí jen prostor široký dva body. Tato konfigurace Mikro QR kódu umožňuje tisk v menších oblastech, kde by se normální QR kód nevlezl. Nejvyšší verze tohoto kódu je verze M4 (17x17 bodů), kdy je možné uložit až 35 číslic. [8]

Obrázek 3: Mikro QR kód [8]

1.2.3 iQR kód

Kód, který může být generovaný jako čtverec nebo jako obdélník. Může být jako převrácený černobílý inverzní kód nebo bodový vzor. Maximální verze, je teoreticky verze 61 (422x422 bodů), kdy je možné uložit okolo 40000 číslic. [8]

Obrázek 4: Čtvercový a obdélníkový iQR kód [8]

1.2.4 SQRC

SQRC je druh QR kódu, vybavený funkcí omezující čtení. Může být použitý pro ukládání soukromých informací, interních firemních informací a podobně. Tyto kódy lze číst pouze konkrétními typy skenerů. Data pro SQRC se skládají z veřejné části a soukromé části, je tedy možné uložit dvě úrovně kontrolních informací v jednom kódu. Vzhled se nijak neliší od běžného QR kódu. [8]

Obrázek 5: SQRC kód [8]

2 TVOŘENÍ QR KÓDŮ

2.1 Struktura

QR kód je čtverec, který se už na první pohled liší od obyčejného dvourozměrného kódu. Skládá se z několika důležitých částí.

2.1.1 Kotvící body

V obou horních a levém dolním rohu se nacházejí kotvící symboly (Position patterns), které slouží ke snadnější lokalizaci levého horního kotvícího symbolu a tak i celého QR kódu čtecím zařízením (viz. Obrázek 6).

Obrázek 6: Kotvící symboly

2.1.2 Separátor

Kotvící symboly jsou odděleny od zbytku kódu bílým pruhem - separátorem (viz. Obrázek 7). Šířka separátoru je jeden bod.

Obrázek 7: Separátor

2.1.3 Zaměřovací body

Všechny verze kódu mají vyhrazený 6. řádek a sloupec na tzv. zaměřovací body (Timing patterns), což jsou střídající se bílé a černé body (viz. Obrázek 8). Tyto body slouží k určování rozměrů kódu.

Obrázek 8: Zaměřovací body

2.1.4 Korekce chyb a číslo masky

Dále všechny QR kódy obsahují také informaci o korekci chyb a čísle masky. Zde je zakódována jedna ze čtyř druhů korekce chyb a jedna z osmi typů masek. V každém QR kódu jsou tyto informace zobrazeny dvakrát (viz. Obrázek 9).

Obrázek 9: Informace o korekci chyb a čísle masky

2.1.5 Zarovnávací symboly

U první verze QR kódů najdeme jen tyto speciální symboly, ale u verze 2 a výše se objevují speciální zarovnávací symboly (Aligment pattern), které slouží k rozdělování kódu na jednotlivé oblasti (viz. Obrázek 10). S vyšší verzí kódu roste i jejich počet.

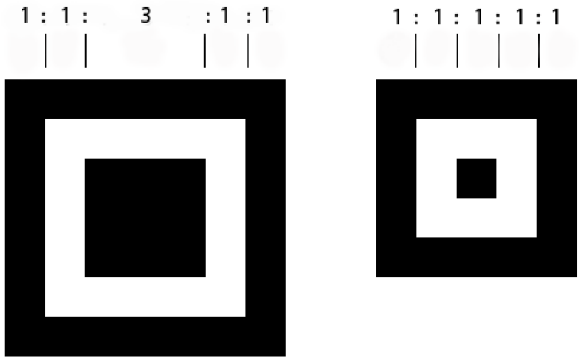
Obrázek 10: Zarovnávací symboly

2.1.6 Quiet zone

Kolem celého QR kódu musí být bílý okraj (Quiet zone). Tento okraj slouží k tomu, aby kód nesplýval s okolím a byl snadněji a správně přečten čtecím zařízením (viz. Obrázek 11). Pro většinu QR kódů je tento okraj minimálně 4 body široký, výjimkou je micro QR kód, u kterého může být okraj široký pouze 2 body.

Obrázek 11: Quiet zone

Jednotlivé rozměry kotvících a zaměřovacích symbolů lze vidět níže (viz. Obrázek 12). Zbývající prostor slouží pro data, informace o masce a korekci chyb.



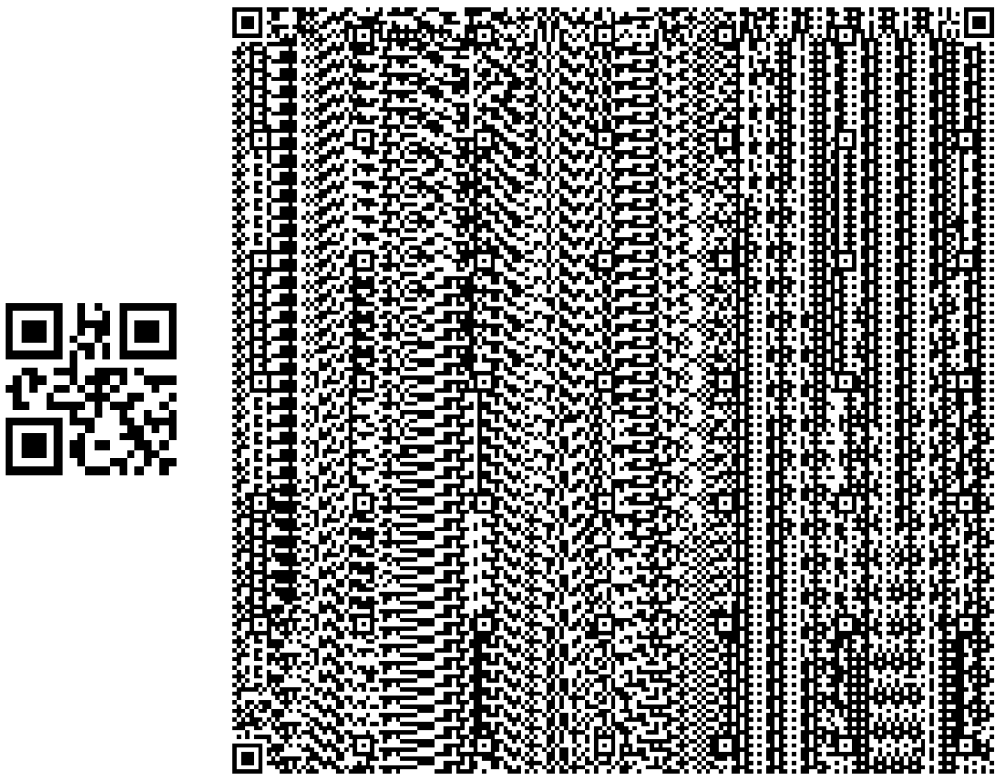
Obrázek 12: Rozměry kotvících a zarovnávacích symbolů

2.2 Velikost

QR kódy mohou mít různé velikosti. Nejmenší matice má velikost 21x21 bodů (verze 1), největší matice 177x177 bodů (verze 40), která dokáže zakódovat až 7 089 znaků (viz. Tabulka 1). Jednotlivé verze mají vždy o 4 body na stranu více, než verze předchozí. Se zvyšující verzí se zvětšuje i počet zarovnávacích symbolů. Níže můžeme vidět stejnou zakódovanou zprávu ve verzi 1 a verzi 40 (viz. Obrázek 13).

Tabulka 1: Počet znaků v QR kódech

Typ obsahu	Počet znaků	Indikátor
Číslice	7089	0001
Písmena a číslice	4296	0010
8bitová data	2953	0100
Kanji znaky	817	1000



Obrázek 13: QR kód verze 1 a verze 40

2.3 Kódování dat

Pro kódování je možné použít několik druhů dat (viz. Tabulka 1). Po výběru typu dat QR kodér vytvoří z daného řetězce znaků matici černých a bílých polí. Aby bylo možné kódy přeměnit v binární data, musí se matice převést také na binární. Černá pole se interpretují jako 1, bílá jako 0. Nejprve se mění písmena, čísla a interpunkční znaménka za pomoci standardní tabulky na čísla (viz. Tabulka 2). Například řetězec „AHOJ" se převede na čísla 10, 17, 24 a 19. Poté z těchto čísel kodér vytvoří páry, aby se šetřilo místo při kódování. První číslo se vynásobí 45 a k výsledku se přičte číslo následující. Pro „AH" to znamená 467. Písmenům „OJ" odpovídá číslo 1099. Pokud řetězec obsahuje lichý počet znaků, poslední znak se číslem 45 nenásobí. Čísla 467 a 1099 konvertuje kodér na 11 bitové binární číslo. Číslu 467 odpovídá 001 1101 0011, číslu 1099 odpovídá v binárním kódu číslo 100 0100 1011. Pokud tato čísla spojíme za sebe, dostaneme řetězec „AHOJ" v QR kódu. Po tomto přepočtu změní kodér bitový kód na bloky o velikosti 8 bitů. Z 22 bitového toku „AHOJ" vzniknou celkem 3 bloky, poslední má však jen 6 bitů a zbývající čísla se doplní nulami. [3]

Tabulka 2: Znaky pro alfanumerické kódování

Znak	Hodnota	Znak	Hodnota	Znak	Hodnota	Znak	Hodnota
0	0	C	12	O	24	SP	36
1	1	D	13	P	25	$	37
2	2	E	14	Q	26	%	38
3	3	F	15	R	27	*	39
4	4	G	16	S	28	+	40
5	5	H	17	T	29	-	41
6	6	I	18	U	30	.	42
7	7	J	19	V	31	/	43
8	8	K	20	W	32	:	44
9	9	L	21	X	33		
A	10	M	22	Y	34		
B	11	N	23	Z	35		

2.4 Korekce chyb

Z různých důvodů může dojít k mechanickému poškození kódu (ušpinění, utržení apod.), proto obsahuje matice i modul pro korekci chyb. Používají se čtyři třídy této korekce (viz. Tabulka 3). Nejnižší je třída L, která si poradí se 7 % poškozením (viz. Obrázek 15). Naopak nejlepší je třída H, která zvládne opravit až 30 % poškozeného kódu (viz. Obrázek 16). Pro každou velikost QR kódu existuje určitý poměr dat a korekce chyb, které se použijí při vytváření (viz. Tabulka 4).

Tabulka 3: Korekce chyb

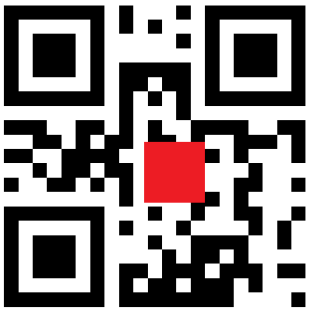
Korekce chyb	Obnovení kódu
L	7 %
M	15 %
Q	25 %
H	30 %

Tabulka 4: Počty bajtů pro korekci chyb a data

Typ korekce chyb	Indikátor úrovně	Bajty pro korekci chyb	Bajty pro data
L	01	7	19
M	00	10	16
Q	11	13	13
H	10	17	9

Čím nižší třída korekce je použita, tím více informací lze zakódovat a naopak. Jen při verzi matice 40 a korekci třídy L je možné zakódovat nejvíc znaků, a to 7089 numerických nebo 4296 alfanumerických. To z toho důvodu, že kromě samotných dat obsahují i speciální kontrolní výpočty, takzvané Reedovy-Solomonovy, kterými se dopočítávají poškozená data. Informace, o úrovni ochrany proti poškození, je uložena vpravo od spodního pozičního symbolu (viz. Obrázek 14). [5]

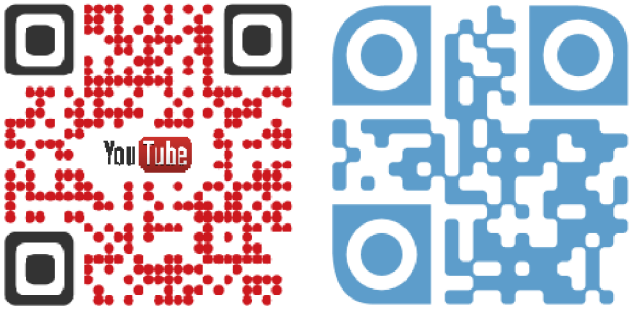
Obrázek 14: Třídy korekce chyb



Obrázek 15: Poškození kódu s korekcí chyb typu L

Obrázek 16: Poškození kódu s korekcí chyb typu H

Díky korekci chyb nemusí být QR kódy jen pravidelné. Černé a bílé čtverce, mohou mít různě zaoblené hran, mohou být různě barevné a je možné do nich vkládat obrázky či loga. Všechny tyto úpravy a geometrické nepřesnosti se oproti normálnímu kódu odfiltrují, jakoby to byly nečistoty nebo poškození a kód bude fungovat úplně stejně.



Obrázek 17: Graficky upravené QR kódy [4]

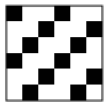
2.5 Masky

Některé vzory v matici QR kódu mohou být obtížné pro správné přečtení čtecím zařízením (velké jednobarevné plochy). Aby se tomu zabránilo, je definováno osm masek, z nichž každá změní QR kód podle určitého vzoru (viz. Tabulka 5). Při aplikaci masky se musí vybrat ta, která vytvoří nejlépe čitelný obrazec a vykazuje nejmenší počet nežádoucích vlastností a tzv. trestných bodů. Trestné body se udávají za tyto vlastnosti:

- Za 5 po sobě jdoucích čtverců maska dostane 3 trestné body. Každý čtverec navíc znamená další trestný bod. Toto pravidlo platí, jak ve vertikálním, tak i v horizontálním směru.
- Za čtverec stejné barvy o velikosti 2x2 bodů jsou připsány 3 body. Např. obrazec o velikosti 3x2 bodů je považován za 2 překrývající se čtverce 2x2, tedy 6 trestných bodů.
- Za symbol obsahující za sebou 4 bílé, černý, bílý, 3 černé, bílý a černý čtverec se přičte 40 trestných bodů. Tato kombinace je možná i naopak.
- Nakonec je důležitý poměr bílých a černých čtverců. Nejprve se zjistí počet všech a počet černých čtverců v matici. Dále se vpočítá procento černých čtverců ((počet černých – počet všech) * 100). Pokud je výsledek např. 43 %, tak určíme nejbližší nižší a nejbližší vyšší násobek 5 tohoto čísla. V tomto případě to bude 40 a 45. Z obou těchto čísla odečteme 50 (|40 - 50| = 10 a |45-50| = 5). Obě čísla vydělíme 5 (10/5 = 2 a 5/5 = 1). Menší z těchto dvou čísel vynásobíme 10. Výsledek bude 10 a to odpovídá 4 trestným bodům.

Aby čtečka QR kódů mohla obrazce převést zpět na bity, musí kodér přidat do matice dvě důležité informace. Typ korekce chyb a použitou masku. Pro tyto informace je v matici vyhrazeno speciální místo (viz. Obrázek 9). [6]

Tabulka 5: Vzory pro masky

Kód	Vzorec	Perioda	Vzor
000	$(x + y)\ mod\ 2$	$2x2$	
001	$y\ mod\ 2$	$1x2$	
010	$x\ mod\ 3$	$3x1$	
011	$(x + y)\ mod\ 3$	$3x3$	
100	$\left[\left(\frac{y}{2}\right) + \left(\frac{x}{3}\right)\right] mod\ 2$	$6x4$	
101	$(x \cdot y)\ mod\ 3 + (x \cdot y)\ mod\ 2$	$6x6$	
110	$[(x \cdot y)\ mod\ 3 + (x \cdot y)]\ mod\ 2$	$6x6$	
111	$[(x \cdot y)\ mod\ 3 + (x + y)]\ mod\ 2$	$6x6$	

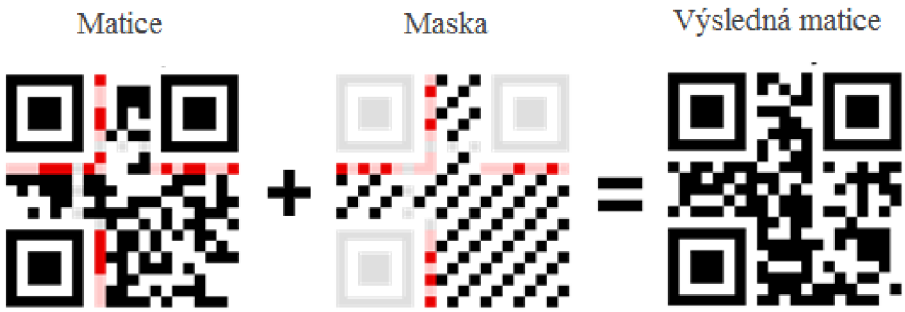
Pro výpočet masek se využívá funkce XOR. Jednotlivé bity matice se porovnávají s jednotlivými bity masky. Pokud jsou porovnávány dva stejné bity (0x0 nebo 1x1) výsledkem je 0, pokud jsou porovnávány odlišné bity (1x0 nebo 0x1) výsledkem je 1. Nově vzniklá matice se výrazně liší od matice původní.

```
Vstupní matice    101101101001011
Maska             101010000010010
Výstupní matice   000111101011001
```

Obrázek 18: Příklad použití funkce XOR při aplikaci masky [7]

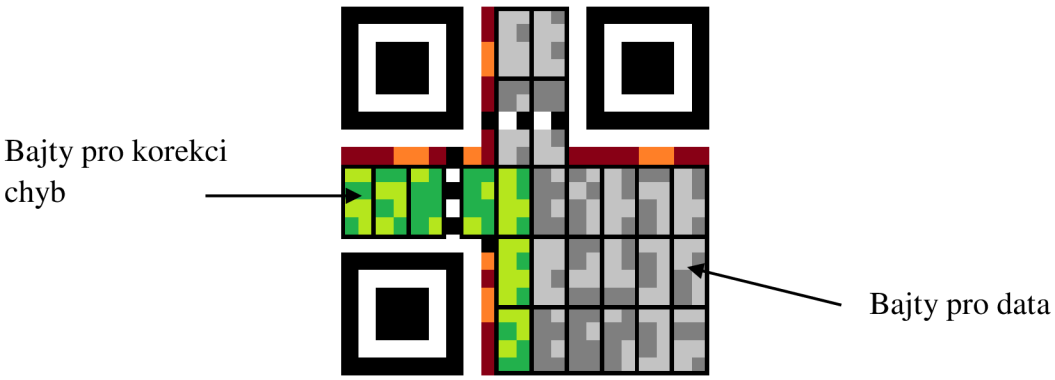
První 2 bity přenášení informaci o úrovni korekce chyb (viz. Tabulka 3). Další 3 bity určují použitý maskovací vzor (viz. Tabulka 5.). Proměnné x a y označují sloupec popřípadě řádek matice. Všechny vzory začínají vždy log. 1 v levém horním rohu.

Podle příkladu výstupní matice (viz. Obrázek 18) lze vidět, že matice obsahuje korekci chyb úrovně M a vzor masky se vzorcem $(x + y)\ mod\ 3$.



Obrázek 19: Použití masky [7]

Tímto víme, že matice bude obsahovat 16 bajtů pro data a 10 bajtů pro korekci chyb. Tyto bajty se nazývají Codewords (CW). Jeden CW obsahuje 8bitů informací, jak již bylo řečeno dříve, log. 1 jako černý a log. 0 jako bílý symbol. V matici se vyskytují pravidelně za sebou v obdélníkovém tvaru v poměru 2x4. Někdy je ale třeba tento tvar porušit, a to hlavně z důvodu výskytu zarovnávacích symbolů, zaměřovacích bodů nebo např. konce matice. Matice tedy bude obsahovat celkem 26 CW (viz Obrázek 20).



Obrázek 20: Codewords

3 GENEROVÁNÍ A ZOBRAZENÍ QR KÓDU

Úkolem této práce je vytvoření funkčního zařízení pro generování QR kódů ze vstupu sériového portu a následně jej zobrazovat na OLED displej s rozlišením 128x128 pixelů. Zpracování dat, následný výpočet a vytvoření QR kódu bude obstarávat mikroprocesor.

3.1 Výběr mikroprocesoru

Pro vyváření kódů, jak již bylo zmíněno, je využit mikroprocesor. Prvním úkolem v této práci tedy bylo určit celkovou paměťovou náročnost tvorby QR kódu a dle této náročnosti vybrat vhodný mikroprocesor s dostatečně velkou pamětí. Tvorba programu na generování QR kódu je značně náročná, a tedy byla využita jedna z volně dostupných knihoven. V tomto případě byla využita knihovna „libqrencode“.

Tuto knihovnu bylo nutno upravit tak, že pro každou alokaci paměti a následné uvolňování byly připravené proměnné, do kterých se přičítala nebo odečítala velikost právě přiřazené paměti. Tím se dosáhlo toho, že se pro určitý řetězec znaků zjistila odpovídající hodnota potřebné paměti.

Dále bylo možné například zmenšit paměťovou náročnost ještě některými úpravami. Jedna z nich např. při výpočtu testování nejlepší masky. Všech 8 masek se násobilo s původní maticí a nejlepší výsledek znamenal nejlepší matici, která byla následně použita. Tím, že pro tento případ bylo prioritou použití méně paměti, oproti rychlosti, bylo možné tyto masky zkoušet po jedné, uložit si jejich index a poté ji smazat. Tímto způsobem se vyzkoušelo všech 8 masek a podle nejlepšího indexu se použila nejlepší maska.

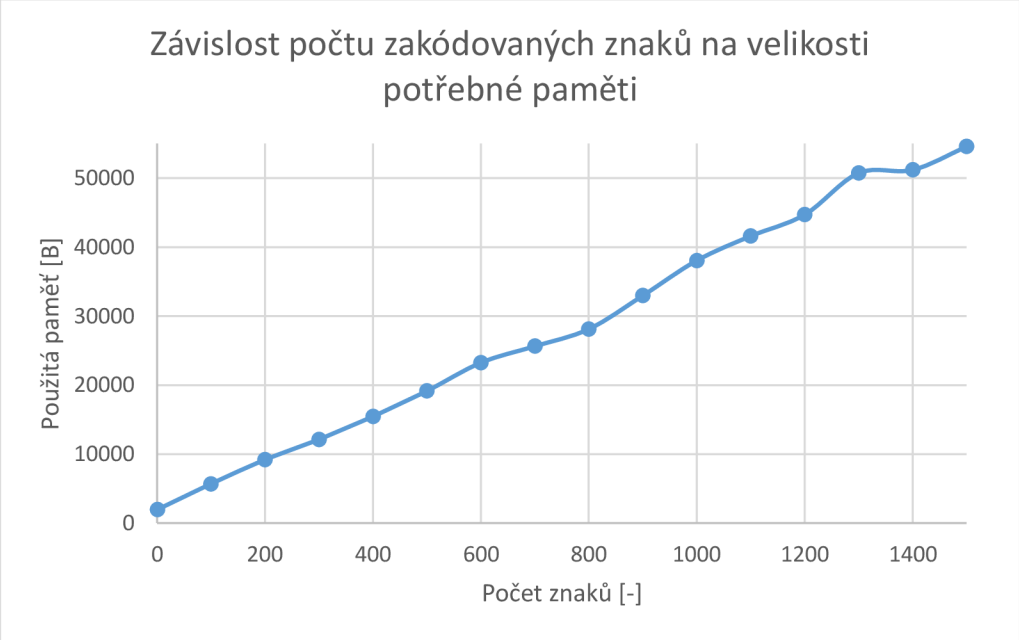
Byly testovány různě dlouhé řetězce znaků (viz. Tabulka 6). Nejmenší možný QR kód, který obsahuje jen jediný zakódovaný znak má 21x21 pixelů a potřebuje 1968 B paměti. Na první pohled se zdá závislost počtu zakódovaných slov a potřebné paměti téměř lineární (viz. Obrázek 21).

Pokud se ale provedlo měření podrobněji, a to pro 1 až 100 znaků (viz. Tabulka 7), šlo vidět, že závislost lineární není. V některých oblastech, se při zvýšení počtu znaků potřebná paměť navýšila jen nepatrně (viz. Obrázek 22).

Testováním bylo zjištěno, že pro zakódování 1500 znaků je potřeba 54575 B paměti, čímž se zároveň vytvořil QR kód o velikosti 125x125 pixelů. Tento počet pixelů tedy ještě vyhovuje, pro zadaný displej. Pro mikroprocesor bude tedy potřeba minimálně 54575 B paměti SRAM.

Tabulka 6: Potřebná paměť pro počet znaků (1 - 1500)

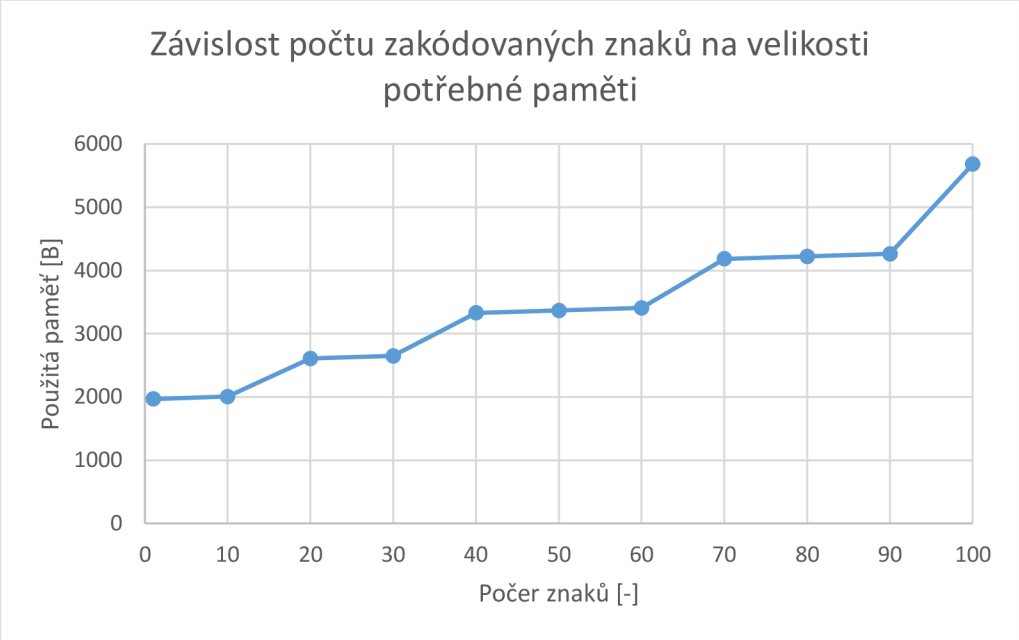
Počet znaků	Potřebná paměť [B]	Počet pixelů
1	1968	21x21
100	5682	37x37
200	9180	49x49
300	12124	57x57
400	15447	65x65
500	19145	73x73
600	23247	81x81
700	25630	85x85
800	28112	89x89
900	32970	97x97
1000	38058	105x105
1100	41608	109x109
1200	44705	113x113
1300	50755	121x121
1400	51188	121x121
1500	54575	125x125



Obrázek 21: Závislost počtu zakódovaných znaků na velikosti potřebné paměti (1 – 1500 znaků)

Tabulka 7: Potřebná paměť pro počet znaků (1 - 100)

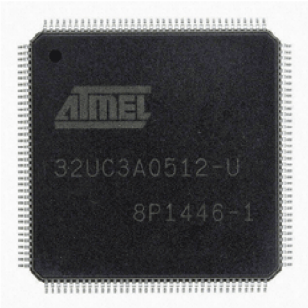
Počet znaků	Potřebná paměť [B]	Počet pixelů
1	1968	21x21
10	2007	21x21
20	2609	25x25
30	2648	25x25
40	3329	29x29
50	3368	29x29
60	3407	29x29
70	4184	33x33
80	4223	33x33
90	4262	33x33
100	5682	37x37



Obrázek 22: Závislost počtu zakódovaných znaků na velikosti potřebné paměti (1 – 100 znaků)

Následně byl vybrán mikroprocesor typu AT32UC3A0512 (viz. Obrázek 23). Tento mikroprocesor má velikost paměti SRAM 64 kB, což by mělo být pro tento případ dostačující. Zde jsou některé další parametry mikroprocesoru:

- Typ integrovaného obvodu: Mikrokontrolér AVR32
- Kapacita paměti Flash: 512 kB
- Kapacita paměti SRAM: 64 kB
- Počet pinů: 144
- Kmitočet taktování: 66 MHz
- Montáž: SMD
- Pracovní napětí: 1.8 - 3.3 V
- Počet kanálů PWM: 13



Obrázek 23: AT32UC3A0512

3.2 Vývojový kit

Pro ověření funkčnosti, bylo potřeba využít zařízení s mikroprocesorem se stejnými nebo velmi podobnými parametry. Jednou z možností bylo, vytvořit si zařízení zkušebně za pomocí kontaktního nepájivého pole nebo využít komerční vývojový kit. Pro tetování byl tedy vybrán vývojový kit EVK1105, který obsahuje právě potřebný mikroprocesor AT32UC3A0512 s dostatečnou pamětí. Tento kit obsahuje také displej ET024006DHU s rozlišením 240x320 pixelů (viz: Obrázek 24).

Obrázek 24: Atmel Evaluation Kit 1105

Tento kit byl programován za pomocí programátoru JTAG ICE MKII, který byl propojen s PC se softwarem AVR studio. Kód bylo třeba upravit, nastavit potřebné parametry displeje a naprogramovat funkci pro výstupní vykreslování.

Obrázek 25: Kit EVK1105 s programátorem JTAG ICE MKII

Zde je vidět část zdrojového kódu, při nastavování QR kódu. Proměnné QRCODE_TEXT se přiřazuje řetězec libovolných znaků, který má být zakódován. Proměnná OUT_FILE_PIXEL_PRESCALER zase určuje, kolikrát bude zobrazovaný pixel zvětšený.

Dále je zde možné nastavit verzi QR kódu od 1 do 40, od čeho se poté odvíjí jeho velikost.

Dalším parametrem je korekce chyb. Nyní je nastavena nejvyšší, tedy korekce typu H (QR_ECLEVEL_H), která dokáže obnovit až 30% poškozeného kódu.

Taky je potřeba nastavit typ kódovaných dat, jako jsou například numerické, alfanumerické, 8bitové nebo Kanji znaky.

Posledním parametrem je tzv. „case sensitive“. To znamená, zda se budou rozlišovat velké a malé písmena (1) nebo ne (0).

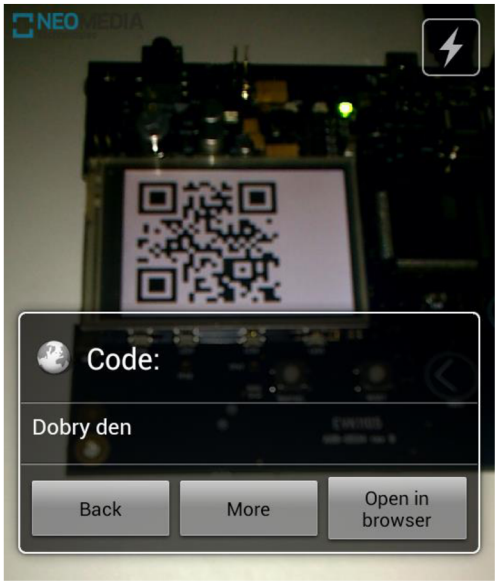
```
#define QRCODE_TEXT  "Dobry den"       // Kodovany text
#define OUT_FILE_PIXEL_PRESCALER  1

QRcode* pQRC;
char* szSourceSring = QRCODE_TEXT;

int draw_qr(unsigned char*p, int size);

void qr(void)
{
pQRC = QRcode_encodeString(szSourceSring, 1, QR_ECLEVEL_H,
QR_MODE_8, 1);

      if (pQRC)
      {
      draw_qr(pQRC->data, pQRC->width);
      }
}
```

Po úspěšném nastavení se povedlo vygenerovaný QR kód zobrazit na displeji (viz. Obrázek 26).

Obrázek 26: QR kód zobrazený na displeji

Nyní bylo ještě třeba ověřit, zda půjde QR kód rozpoznat čtečkou v mobilním telefonu a poté i jestli se povedlo zobrazit stejnou zprávu, která byla kódována. Zde ale nastal problém, protože čtečka nechtěla zobrazený QR kód načíst.

Jak se ale píše v kapitole 2.1.6, pro QR kód je nutné mít okolo něj minimálně 4 body široký bílý okraj, aby nesplýval s okolím. Po odsazení tohoto kódu od hran displeje se už povedlo čtečkou QR kód načíst, zobrazit zprávu a zjistit, že celý proces proběhl správně (viz. Obrázek 27).



Obrázek 27: Načtení QR kódu

4 NÁVRH VLASTNÍHO ZAŘÍZENÍ

4.1 Návrh obvodu

Jak již bylo řečeno, úkolem této práce je vytvoření funkčního zařízení pro generování QR kódů ze vstupu sériového portu a následně jej zobrazovat na OLED displeji s rozlišením 128x128 pixelů. Zařízení se tedy muselo skládat z několika důležitých částí, které bylo třeba nejprve navrhnout. K navrhování schématu i samotného vzhledu desky byl použit program Eagle od firmy CadSoft.

4.1.1 Napájení

Pro toto zařízení bylo zvoleno napájecí napětí 24V, ale pro zvolené součástky bylo potřeba 3 různé úrovně napětí. Proto bylo nutné toto napájecí napětí upravit podle potřeb pomocí lineárního napěťového stabilizátoru a step-down měniče.

4.1.2 Step-down měnič

Jelikož displej bylo potřeba napájet kromě 3,3 V i 13 V, bylo nezbytné navrhnout, jak se tyto úrovně napětí získají. Zatímco 3,3 V budou vytvářeny pomocí lineárního stabilizátoru (viz. 4.1.3), pro 13 V byl navrhnut step-down měnič, který převede toto napětí z napájecích 24 V. Jako měnič byl použit integrovaný obvod LMR14203XMK od firmy Texas Instruments. Firma Texas Instruments také nabízí online aplikaci, kde pro tento měnič a jeho zadané požadované parametry navrhne hodnoty jednotlivých dostupných součástek ve správném zapojení.

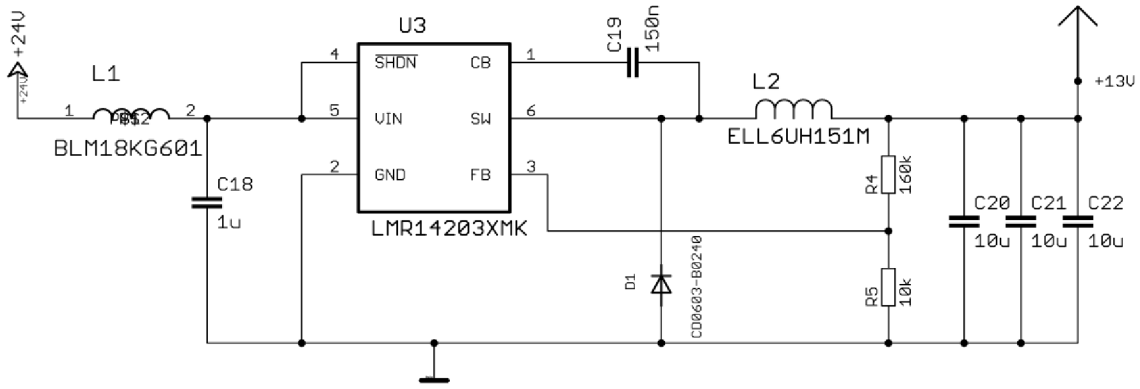
Na vstupu měniče jsou připojeny tlumivka a kondenzátor, které zamezují vysokofrekvenčnímu rušení. Stejně tak kondenzátor na výstupní straně, ten byl ovšem z prostorového omezení nahrazen paralelním spojením třech daleko menších kondenzátorů stejné hodnoty. Doporučené hodnoty výstupního kondenzátoru jsou 22 µF – 100 µF. Pro větší účinnost by měla být vzdálenost výstupního kondenzátoru od diody D1 a tlumivky L2 co nejkratší. Odporový dělič na výstupu reguluje výstupní napětí, a lze ho získat, zvolením hodnoty jednoho rezistoru a dopočítáním druhého podle vzorce:

$$R4 = R5 \cdot \left(\left(\frac{Vout}{0{,}765}\right) - 1\right) \qquad [\Omega]$$

Kondenzátor C19 bývá použit jen pro aplikace, kde je vstupní napětí větší, než dvojnásobek výstupního napětí, pro dostatečné otevírání spínačů. Jako dioda D1 byla zvolena Schottkyho dioda s proudem v propustném směru 200 mA a maximálním napětím v propustném směru 550 mV. V případech, kde je vstupní napětí mnohem větší, než výstupní napětí, je možné použít diodu s nižší průměrnou proudovou zatížitelností. Jako tlumivka L2 byla zvolena tlumivka ELL6UH51M s indukčností 150 µH. Pro snížení magnetického a elektrostatického šumu by měla být umístěna v blízkosti pinu SW. Seznam všech součástek i s jejími hodnotami je přiložen v příloze.

Tabulka 8: Parametry step-down měniče

Technické parametry	
Vstupní napětí	4,5 – 42 V
Výstupní napětí	0,765 – 34 V
Výstupní proud	300 mA
Počet výstupů	1
Počet pinů	6
Pouzdro	SOT-23



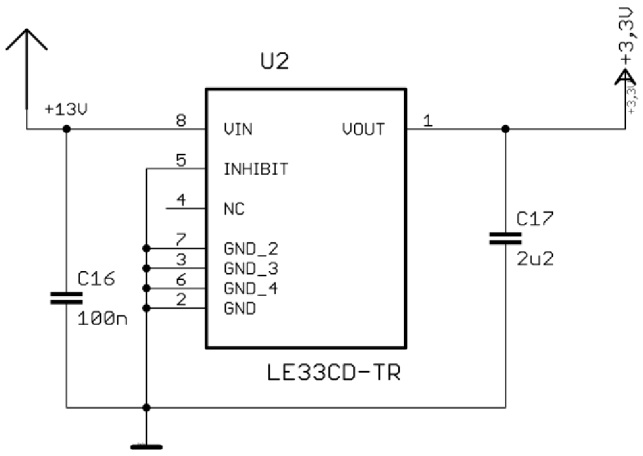
Obrázek 28: Zapojení step-down měniče

4.1.3 Lineární stabilizátor

Mikroprocesor, displej i RS-232 převodník jsou napájeni z 3,3 V, tudíž bylo potřeba napětí získat z 13 V za pomocí lineárního napěťového stabilizátoru snížit na tuto hodnotu. Byl vybrán lineární stabilizátor LE33CD-TR od firmy ST Microelectronics. Na vstupu a výstupu stabilizátoru jsou mezi pin a GND připojeny 2 filtrační kondenzátory, které zamezují vysokofrekvenčnímu rušení.

Tabulka 9: Parametry lineárního stabilizátoru

Technické parametry	
Vstupní napětí	5,3 – 18 V
Výstupní napětí	3,3 V
Výstupní proud	150 mA
Počet pinů	8
Pouzdro	SO-8



Obrázek 29: Zapojení lineárního stabilizátoru

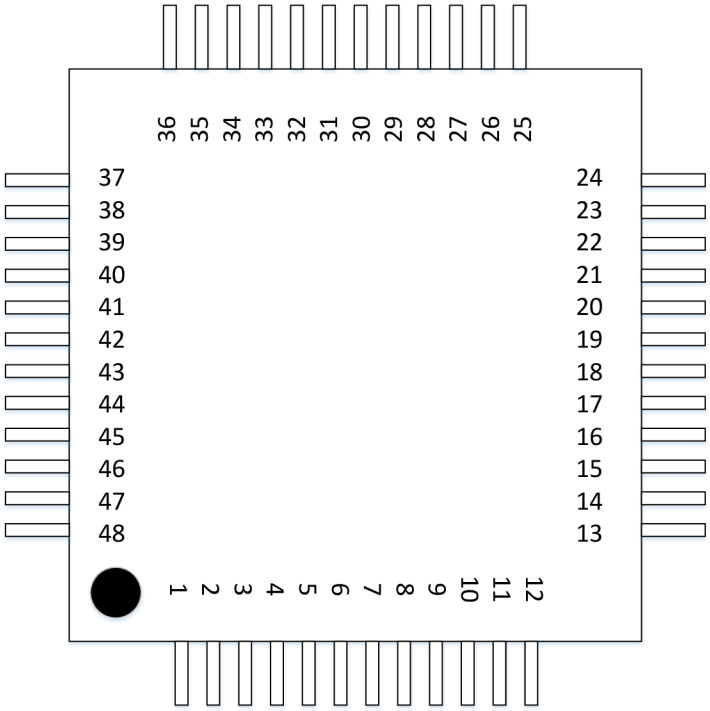
4.1.4 Mikroprocesor

Na vývojovém kitu byl použit mikroprocesor AT32UC3A0512, který měl pro testování dostačující parametry, avšak pro návrh vlastní desky bylo výhodnější použít mikroprocesor AT32UC3B1512, který má hodně podobné parametry, ale je výrazně menší. Tímto se na desce ušetřilo důležité místo. K napájecím vstupům jsou připojeny filtrační kondenzátory. Tento mikroprocesor je napájen 3,3 V, ale některé piny musely být napájeny pouze 1,8 V. Tuto hodnotu si mikroprocesor sám reguluje z 3,3 V a posílá na pin VDDOUT, který je použit pro napájení pinů VDDPLL a VDDCORE. Mikroprocesor je v pouzdře QFN a má 48 pinů (viz. Obrázek 30).

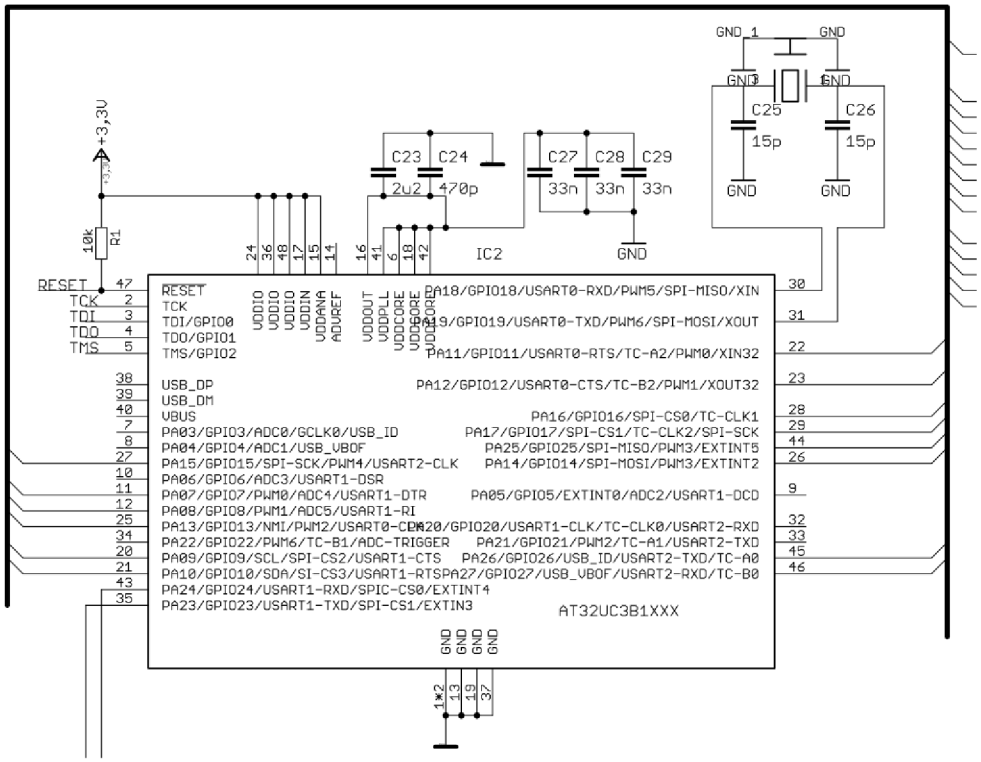
K tomuto mikroprocesoru byl připojen krystal, který pro něj generuje frekvenci 25 MHz. Bez připojení krystalu by mikroprocesor používal interní oscilátor s frekvencí pouze 115 KHz.

Tabulka 10: Porovnání parametrů otestovaného a použitého mikroprocesoru

Parametr	AT32UC3A0512	AT32UC3B1512
Flash paměť	512 kB	512 kB
Počet pinů	144	48
Max. frekvence	66 MHz	60 MHz
CPU	32-bit AVR	32-bit AVR
Počet I/O pinů	109	28
UART	4	2
SRAM	64 kB	96 kB



Obrázek 30: Označení pinů mikroprocesoru



Obrázek 31: Zapojení mikroprocesoru

4.1.5 Displej

O zobrazování zadaných QR kódů se stará OLED displej typu Densitron DD-128128FC-6A s rozlišením 128x128 pixelů. Pro zobrazování je u tohoto displeje využitelných pouze 125x125 pixelů. Pro správnou funkci displeje je potřeba nastavit správnou napájecí sekvencí. Touto sekvencí se rozumí přivedení napětí 3,3 V a po zpoždění 100 ms, přivést napětí 13 V. Po této inicializační sekvenci dojde k rozsvícení displeje a je možno jej využít pro zobrazování informací (viz. 4.3.1). Použití součástek v zapojení bylo realizováno podle doporučených hodnot v katalogovém listu displeje.

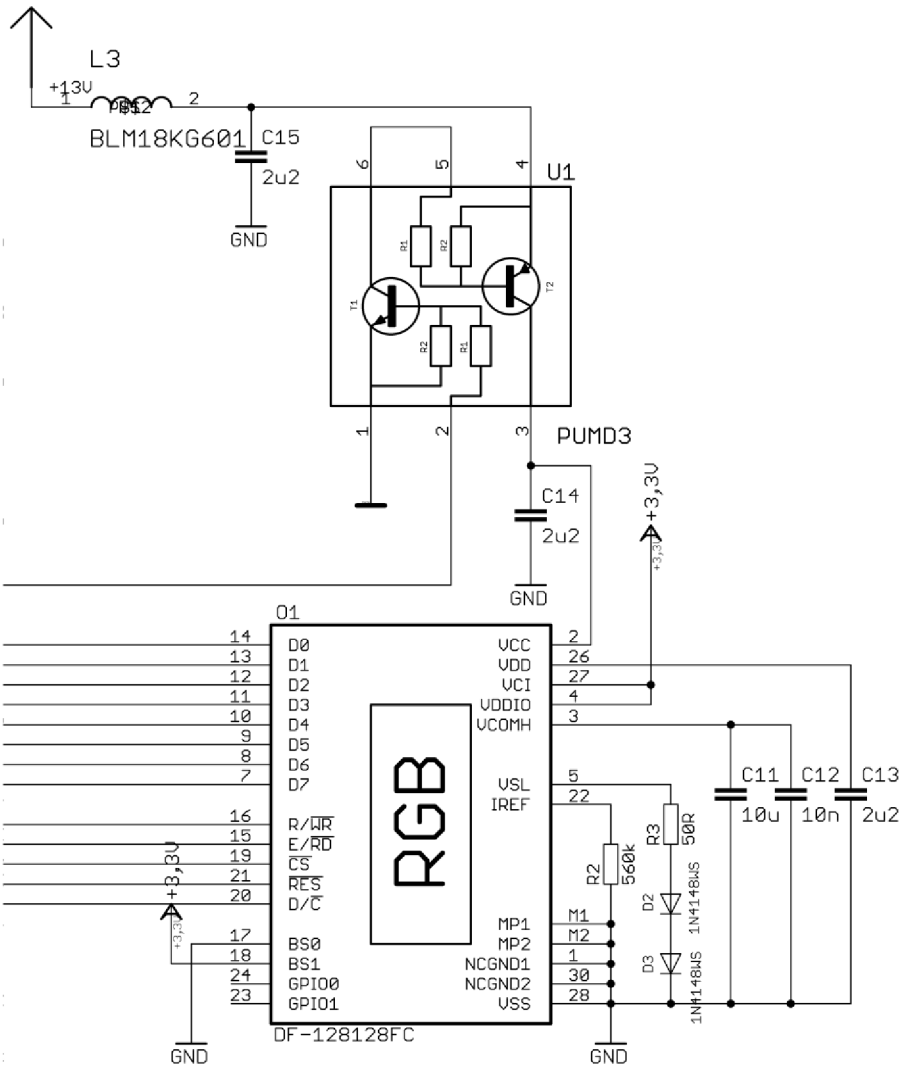
Vstup 13 V jdoucích do displeje jen v určitém okamžiku bylo potřeba softwarově ovládat pomocí tranzistorů. Pro tento případ byl použit dvojitý tranzistor PUMD3. Tato součástka obsahuje v jednom pouzdře PNP tranzistor, NPN tranzistor a 2 rezistory s hodnotou 10 kΩ. Tyto součástky se nazývají „resistor-equipped transistors“.

Tabulka 11: Parametry displeje

Technické parametry	
Typ displeje	OLED
Rozhraní	Paralelní, sériové
Rozlišení displeje	128x128 pixelů
Zobrazovací oblast	28,855x28,864 mm
Ovladač IC	SSD1351
Mód displeje	Pasivní matice (1,5“)
Napájecí napětí	3,3 V, 13 V

Tabulka 12: Parametry tranzistorů

Technické parametry	
Napětí kolektor - emitor	50 V
DC kolektorový proud	100 mA
Polarita tranzistorů	NPN, PNP
Počet pinů	6
Výkonové ztráty	200 mW
Pouzdro	SOT-363
Provozní teplota max.	150 °C



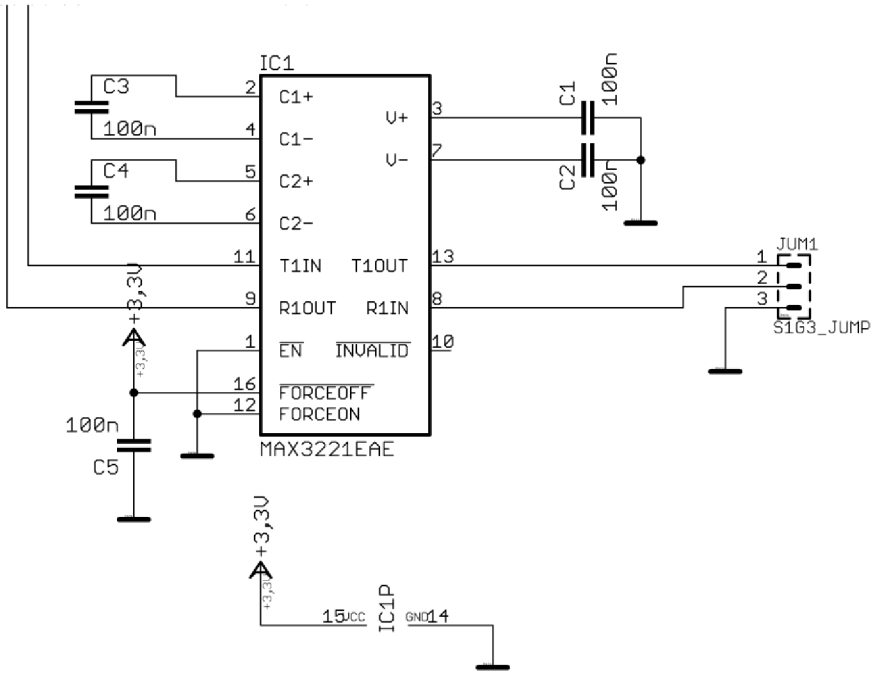
Obrázek 32: Zapojení displeje

4.1.6 RS-232 převodník

Jako převodník pro komunikaci mikroprocesoru s PC po sériové lince slouží RS-232 převodník MAX3221EEAE. Typická hodnoty použitých kondenzátorů v zapojení bývá 100 nF. Převodník je propojen s mikroprocesorem pomocí dvou pinů T1IN a R1OUT. Díky tomuto bude možné kódovat do QR kódu řetězec znaků zasílaný z PC přes sériovou linku. Pro propojení s PC bylo potřeba použít převodník z RS232 na USB.

Tabulka 13: Parametry RS-232 převodníku

Technické parametry	
Provedení	SMD
Typ obvodu	Driver/receiver
Typ sběrnice	RS232
Počet kanálů	1
Napájecí napětí	3; 5,5 V
Provozní teplota	-40 – 85 °C
Pouzdro	SSOP16



Obrázek 33: Zapojení RS-232 převodníku

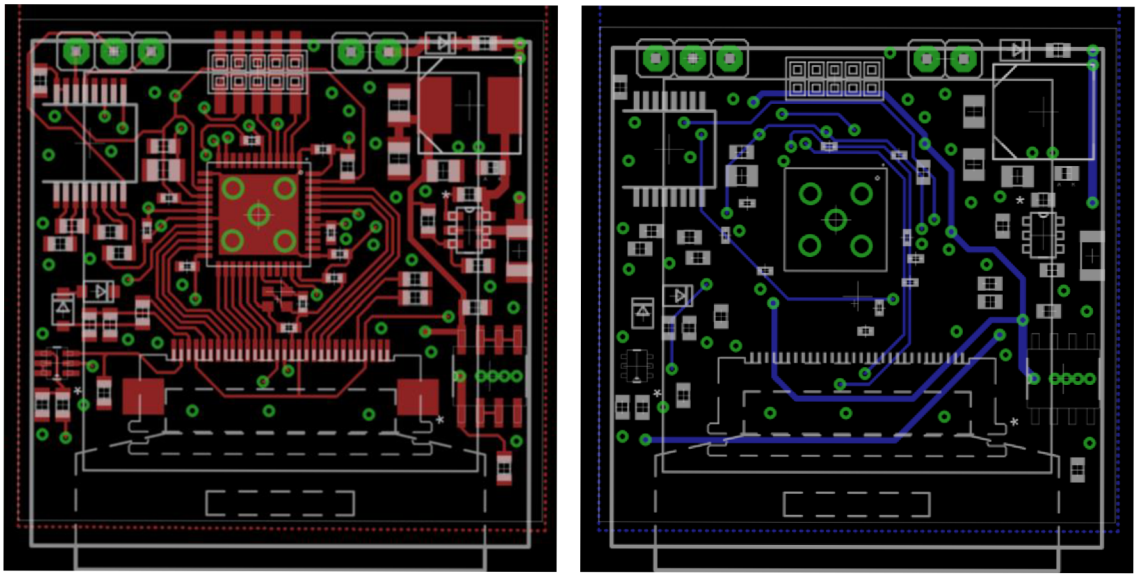
4.2 Návrh DPS

Propojením všech výše zmíněných částí bylo možné vytvořit funkční zařízení, nejprve však bylo potřeba navrhnout vzhled desky, tedy rozmístění a propojení jednotlivých součástek.

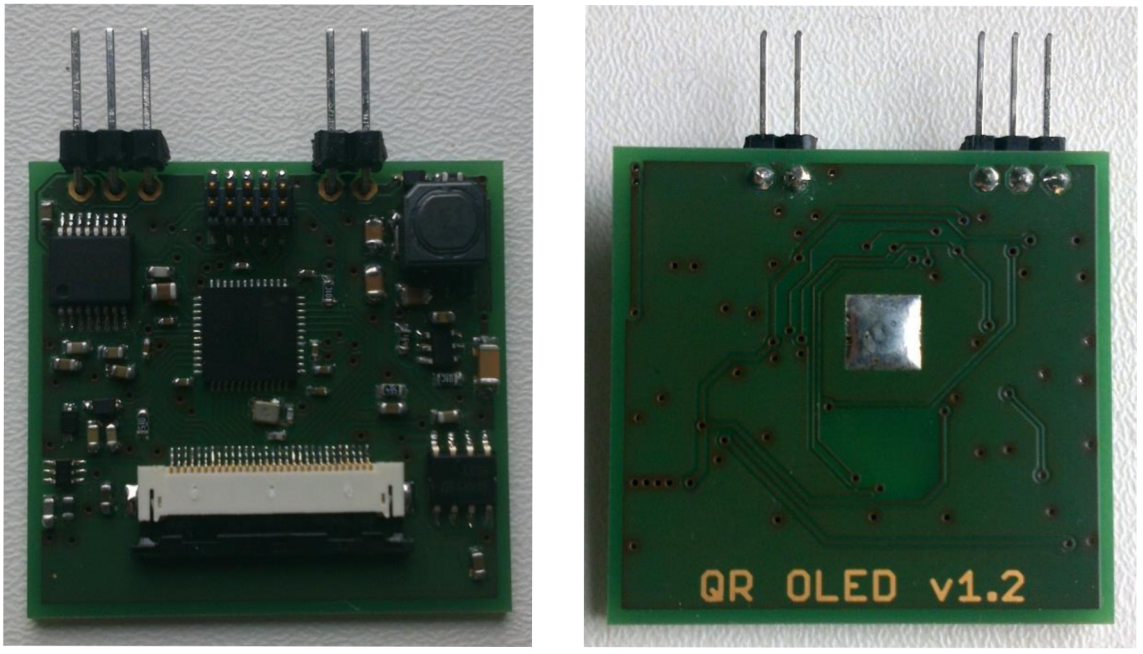
4.2.1 Rozložení součástek

Při vytváření plošných spojů bylo třeba dbát na správné rozložení a propojení jednotlivých součástek. Tím, že použitých součástek není mnoho, bylo vhodné se je pokusit rozložit tak, aby je bylo možno umístit na desku, která se přibližně rovnala rozměrům displeje, tedy 33,8 x 34 mm. Na takto malé desce se ovšem vyskytl problém s plošnými spoji, kdy bylo na malém prostoru potřeba umístit velké množství těchto spojů. Pro usnadnění propojení bylo potřeba tedy použít plošné spoje ve dvou vrstvách. Druhá vrstva byla použita zejména pro rozvádění napětí k součástkám po celé desce. Spoje mezi součástkami jsou široké 0,2 mm a tam, kde se spoj větví na více spojů nebo kde protékají větší proudy, byla použita šířka 0,4 mm nebo až 0,6 mm.

V horní části desky se nacházejí všechny potřebné konektory. Z levé strany to jsou 3 piny pro komunikaci po sériové lince (GND, Rx a Tx), dále 10-pinový konektor pro programátor JTAG a na pravé straně jsou to napájecí piny (GND a +24 V). Od napájecích pinů je napětí vedeno k step-down měniči a jeho potřebným součástkám. Celý tento blok se nachází v pravém horním rohu desky. Pod měničem v pravém dolním rohu desky se nachází lineární stabilizátor s oběma kondenzátory. Od tohoto stabilizátoru je rozváděno napětí 3,3 V po celé desce hlavně pomocí bottom vrstvy (viz. Obrázek 34). V samotném středu desky se nachází mikroprocesor, ke kterému vede nejvíce spojů ze všech součástek. Obzvlášť při spojení mezi mikroprocesorem a displejem bylo nutné, zvolit piny pro komunikaci tak, aby se zbytečně nekřížily. V nejlepším případě tak, aby byly všechny vedle sebe. Okolo mikroprocesoru jsou k napájecím vstupům připojeny filtrační kondenzátory. Dalším problémem bylo připojení krystalu, jelikož musel být v těsné blízkosti mikroprocesoru i s potřebnými dvěma kondenzátory. Navíc piny pro připojení tohoto krystalu byly v prostoru mezi piny pro připojení k displeji, což nebylo nejvhodnější. Tímto se musely spoje rozšířit a tak na desce zabíraly více prostoru. Pod mikroprocesorem byla deska provrtána, upravena a po zapájení použita jako chladící ploška pro lepší odvod tepla. Na spodní straně desky bylo vyhrazeno místo pro displej. Po připojení k desce ji displej z velké části překrýval. Na levé straně desky se nachází kondenzátory a diody nezbytné pro správnou funkci displeje. V tomto prostoru se taky nachází dvojtý tranzistor, který obsahuje NPN a PNP tranzistor a slouží pro ovládání přívodu napětí 13 V do displeje. Poslední část desky je vlevo nahoře, kde se nachází integrovaný obvod sloužící pro sériovou linku jako převodník z TTL na RS232. Vyrobená deska je zobrazena níže (viz. Obrázek 35).



Obrázek 34: Návrh vrchní (Top) a spodní (Bottom) vrstvy

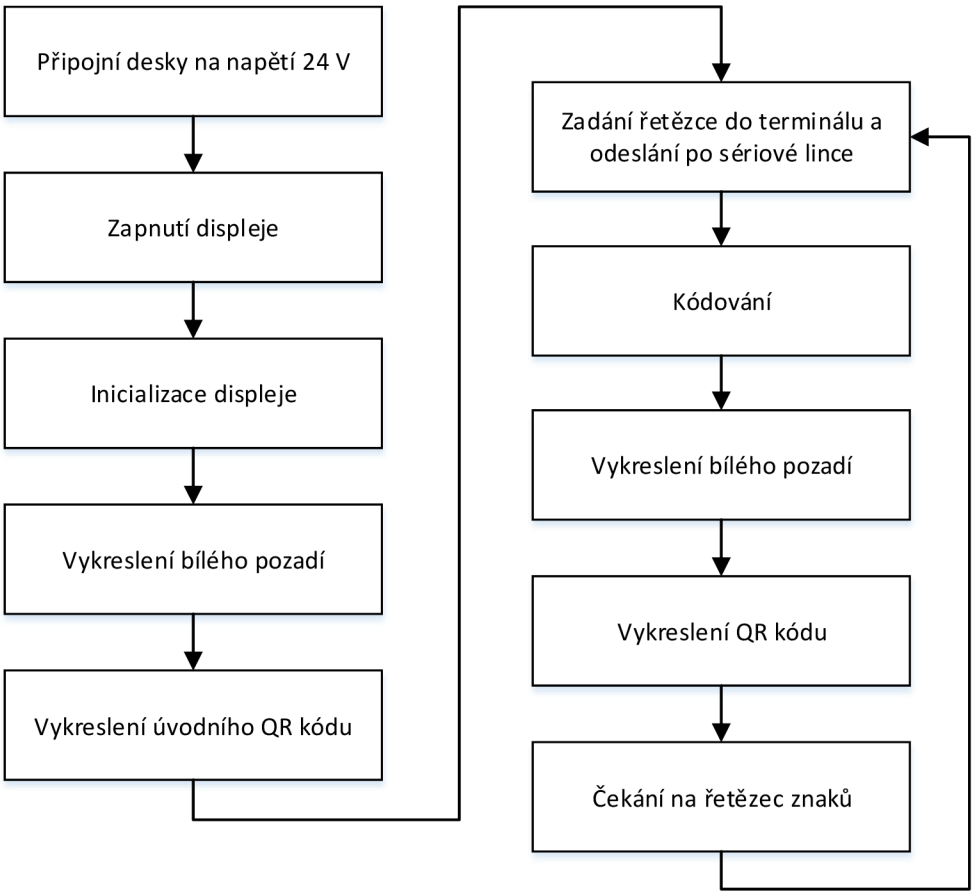


Obrázek 35: Vyrobená a součástkami osazená deska

4.3 Programování

Níže je uvedeno schéma, ve kterém jsou vidět jednotlivé kroky při postupu uvedení zařízení do provozu (viz. Obrázek 36). Ještě před samotným programováním bylo ale potřeba zařízení otestovat a proměřit, jestli jsou všechny součástky napájeny správnou úrovní napětí. Také bylo potřeba zkontrolovat, zda někde v obvodu nedochází ke zkratu nebo se některé součástky příliš nepřehřívají. Po úspěšné kontrole mohlo přijít na řadu samotné naprogramování.

Postup programování probíhal ovšem v mírně odlišném pořadí, dle priority pro funkčnost zařízení. Nejprve bylo nutné zjistit, jak displej funguje, zapnout jej a inicializovat. Poté bylo potřeba naprogramovat vykreslování na displej. Generování QR kódů bylo již předpřipravené z testovací části v semestrálním projektu, a tudíž ho bylo potřeba jen patřičně upravit. Nakonec se místo zadávání řetězce pro zakódování přímo ve zdrojovém kódu připravila komunikace přes sériovou linku, kde se řetězec zadává přes terminál v PC.



Obrázek 36: Blokové schéma zařízení

4.3.1 Zapnutí displeje

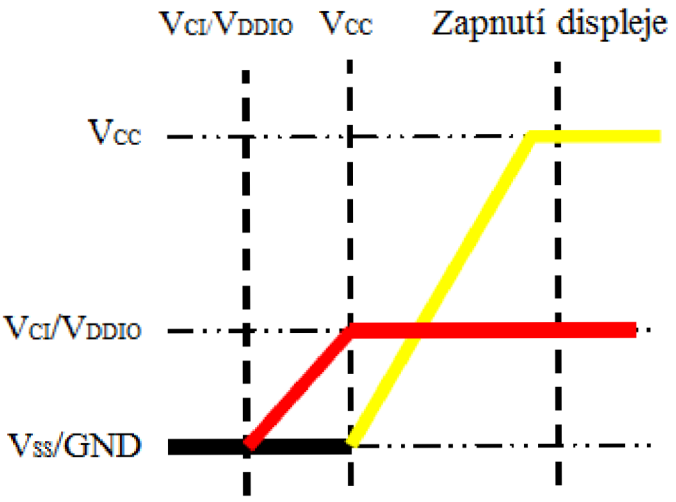
Displej komunikuje s mikroprocesorem pomocí 8 datových pinů D0 – D7, 5 pinů $R/\overline{WR}$, $E/\overline{RD}$, $\overline{CS}$, $\overline{RES}$, $D/\overline{C}$ a byl nastavený 8bitový paralelní komunikační protokol 8080.

Tabulka 14: Názvy pinů displeje

D0 – D7	Vstupní/Výstupní data
$R/\overline{WR}$	Zápis
$E/\overline{RD}$	Čtení
$\overline{CS}$	Výběr čipu
$\overline{RES}$	Reset
$D/\overline{C}$	Data/Příkaz

Při zapínání displeje je potřeba dodržet danou sekvenci následujících příkazů:

1) Přivedení VCI a VDDIO
2) Vypnutí displeje
3) Inicializace displeje
4) Vymazání displeje
5) Přivedení VCC
6) Zpoždění 100 ms
7) Zapnutí displeje



Obrázek 36: Graf zapínací sekvence

Napětí 13 V, které musí být přivedeno na displej až po napětí 3,3 V, je řízeno mikroprocesorem pomocí dvou tranzistorů. Níže je uveden příklad, jak je pomocí příkazů v programovacím prostředí nastavován pin PA09, který řídí přivádění napětí 13 V do displeje. Nejprve je potřeba tento pin povolit a nastavit jako výstupní pomocí příkazů `gpers` a `oders`. Vynulováním (`CLR_PIN(PIN_PWR)` a zasláním log, 1 (`SET_PIN(PIN_PWR)`) se povolí přívod napětí, po kterém přijde na řadu inicializace displeje.

```
#define PIN_PWR 9

void main_init (void)
{
      board_init();
      AVR32_GPIO.port[0].gpers = 1 << PIN_PWR;
      AVR32_GPIO.port[0].oders = 1 << PIN_PWR;
      CLR_PIN(PIN_PWR);
      delay1(100);
      lcdInit();
      SET_PIN(PIN_PWR);
}
```

4.3.2 Inicializace displeje

Inicializace je série příkazů a dat pro základní nastavení displeje. Zasláním těchto příkazů bude displej připraven na další komunikaci. Mezi základní parametry, které je třeba nastavit, patří například nastavení řádku, sloupce, výběr módu, nastavení funkcí, GPIO, kontrast, atd. Každému příkazu nebo datům je přiřazená určitá hexadecimální hodnota. Nejprve bylo potřeba povolit všechny porty na displeji, se kterými se bude dále pracovat a nastavit jim hodnotu log. 1.

```
void lcdPortInit(void)
{
        //set enable D0 - D7
      AVR32_GPIO.port[0].gpers |= 0xFF << 10;
        //set output D0 - D7
      AVR32_GPIO.port[0].oders |= 0xFF << 10;
        //set 1 D0 - D7
      AVR32_GPIO.port[0].ovrs |= 0xFF << 10;

        //set enable WR, RD
      AVR32_GPIO.port[0].gpers |= 0x03 << 7;
        //set output WR, RD
      AVR32_GPIO.port[0].oders |= 0x03 << 7;
        //set 1 WR, RD
      AVR32_GPIO.port[0].ovrs |= 0x03 << 7;
```

```
    //set enable CS, RES, D/C
    AVR32_GPIO.port[0].gpers |= 0x07 << 25;
    //set output CS, RES, D/C
    AVR32_GPIO.port[0].oders |= 0x07 << 25;
    //set 1 CS, RES, D/C
    AVR32_GPIO.port[0].ovrs |= 0x07 << 25;
}
```

Pro zjednodušení a přehlednost kódu inicializace byly ještě vytvořeny funkce pro výběr zápisu příkazu (writeCommand) nebo dat (writeData), které se musely rozlišovat.

V obou funkcích se nastaví pro zaslání příkazu nebo dat pin D/$\overline{C}$. Pro případ zaslání příkazu se použije příkaz `ovrc`, což znamená nastavení na log. 0. a pro zapsání dat se použije příkaz `ovrs`, což znamená nastavení na log. 1. Dále se nastavil aktuální volný port pro komunikaci a vše se potvrdilo nastavením pinu $\overline{CS}$ na log. 0.

```
#define SET_PIN(p) AVR32_GPIO.port[0].ovrs = (1 << p);
#define CLR_PIN(p) AVR32_GPIO.port[0].ovrc = (1 << p);

void writeCommand(uint8_t a)
{
    CLR_PIN(PIN_DC);
    setLcdPort(a);
    CLR_PIN(PIN_CS);
    WAIT_LCD;
    CLR_PIN(PIN_WR);
    WAIT_LCD;
    SET_PIN(PIN_WR);
    WAIT_LCD;
}

void writeData(uint8_t a)
{
    SET_PIN(PIN_DC);
    setLcdPort(a);
    CLR_PIN(PIN_CS);
    WAIT_LCD;
    CLR_PIN(PIN_WR);
    WAIT_LCD;
    SET_PIN(PIN_WR);
    WAIT_LCD;
}
```

Inicializace je nastavena do dvou částí. Nejprve se provede inicializace portů, vynuluje a zpět se nastaví na log. 1 na pin reset. Do této části patří ještě příkazy `COMMANDLOCK` a `DISPLAYOFF`. V druhé části se už posílají příkazy a data s určitými hexadecimálními hodnotami. Část samotné inicializace vypadá takto:

```
void lcdInit(void)
     {
          lcdPortInit();
          delay1(10);
          CLR_PIN(PIN_RST);
          delay1(10);
          SET_PIN(PIN_RST);
          delay1(10);
          writeCommand(SSD1351_CMD_COMMANDLOCK);      // 0xFD
          writeData(0x12);
          writeCommand(SSD1351_CMD_COMMANDLOCK);      // 0xFD
          writeData(0xB1);
          writeCommand(SSD1351_CMD_DISPLAYOFF);       // 0xAE
     }
     void lcdOn(void)
     {
          writeCommand(SSD1351_CMD_CLOCKDIV);         // 0xB3
          writeCommand(0xF1);
          writeCommand(SSD1351_CMD_SETCOLUMN);        // 0x15
          writeData(0x00);
          writeData(0x7F);
          writeCommand(SSD1351_CMD_SETROW);           // 0x75
          writeData(0x00);
          writeData(0x7F);
          writeCommand(SSD1351_CMD_STARTLINE);        // 0xA1
          writeData(0);
          writeCommand(SSD1351_CMD_DISPLAYOFFSET);    // 0xA2
          writeData(0x0);
          writeCommand(SSD1351_CMD_SETGPIO);          // 0xB5
          writeData(0x00);
          writeCommand(SSD1351_CMD_FUNCTIONSELECT);   // 0xAB
          writeData(0x01);
     }
```

4.3.3 Vykreslení pozadí

Inicializovaný displej byl nyní připraven k zobrazování kódů. Jelikož jeho pozadí po inicializaci bylo složeno z několika barev, střídajících se po každém pixelu, a tudíž by na něm QR kód moc dobře nevyniknul, bylo nutné nejprve vykreslit pozadí. Pozadí bylo možno vykreslit libovolnou barvou, ale pro dobrou viditelnost bylo pozadí vykresleno barvou bílou.

Parametry `SSD1351HEIGHT` a `SSD1351WIDTH` udávají šířku a výšku celého displeje, tudíž, je procházen celý displej pixel po pixelu a následně každý zbarven na požadovanou barvu.

Příklad kódu, jak je vykreslováno bílé pozadí na displej pixel po pixelu:

```
void clearScreen(void)
{
      int x;
      int y;
      int color = WHITE;
      goTo(0, 0);
      for(x=0;x<SSD1351HEIGHT*SSD1351WIDTH;x++)
      {
            writeData(color >> 8);
            writeData(color);
      }
      return;

      for(x=0;x<128;x++)
      {
            for(y=0;y<128;y++)
            {
                  drawPixel(x, y, WHITE);
            }
      }
}
```

4.3.4 Vykreslení QR kódu

Jak již bylo popsáno výše, zadaná informace se nejprve pomocí funkce zakóduje. Poté už bylo možno na bílé pozadí tuto informaci vykreslit v podobě QR kódu. Vykreslování probíhá pomocí funkce, která postupně prochází zakódovaný binární řetězec znaků, a podle něj vykresluje bílé nebo černé obrazce:

```
#define K
#define O 10

int draw_qr(unsigned char*pSourceData, int size)
{
     int x,y;
     int unWidth = size;
     int K = 118/size;

     for(y = 0; y < unWidth; y++)
     {
           for(x = 0; x < unWidth; x++)
           {
                 if (*pSourceData & 1)
                 {
                 rawFillRect( x*K+O, y*K+O, K, K, BLACK);
                 }
                 else
```

```
                    {
                    rawFillRect(x*K+O, y*K+O, K, K, WHITE);
                    }
                    pSourceData++;
               }
          }
          return(0);
     }
```

Kód se nejprve vykresloval po pixelech, tudíž byl zobrazený výsledek hodně malý a proto bylo potřeba upravit vykreslování tak, aby se místo pixelů vykreslovaly o něco větší čtverce. Tyto čtverce bylo poté možné K-krát zvětšit podle rozměrů vygenerovaného kódu a roztáhnout tak QR kód na celý displej. Pro to, aby byl vykreslený QR kód čitelný pro čtecí zařízení, bylo potřeba zachovat tzv. „quiet zone“ (viz. Obrázek 11). Tohoto jsme docílili odsazením QR kódu od okraje minimálně o 4 pixely.

4.3.5 Sériová linka

Komunikací přes sériovou linku se rozumí, že jsou data přes komunikační kanál přenášená postupně po jednotlivých bitech. Pro propojení PC se zařízením je použit převodník z USB na RS232. Zapojení na desce je sestaveno z 3 nejdůležitějších pinů pro komunikaci: GND, RxD a TxD. V našem případě se jedná o jednosměrnou komunikaci z PC k mikroprocesoru, proto stačí na výstupu z PC zapojit pin RxD (Vysílání) a na vstupu do desky pin TxD (Příjem). Pin GND musí být zapojen v obou případech.

Při programování musely být nejprve nastaveny systémové hodiny dané sériové linky. Také muselo být definováno, pomocí čeho se bude komunikovat. Pro nás to byl `USART1`, proto byl nastaven a také nastaveny jeho použité piny na mikroprocesoru.

```
sysclk_enable_peripheral_clock(usart);
volatile struct avr32_usart_t *usart = &AVR32_USART1;
static const gpio_map_t USART_GPIO_MAP =
{
{AVR32_USART1_RXD_0_0_PIN, AVR32_USART1_RXD_0_0_FUNCTION}
{AVR32_USART1_TXD_0_0_PIN, AVR32_USART1_TXD_0_0_FUNCTION}
};
```

Dalším krokem bylo nastavení parametrů, které jsou důležité pro komunikaci. Je nutné tyto parametry nastavit totožné jak v programu, tak i v použitém terminálu. Těmito parametry jsou `baudrate`, `charlength`, `paritytype`, `stopbits` a `channelmode`.

```
static const usart_options_t opt =
{
      .baudrate = 57600,
      .charlength = 8,
      .paritytype = USART_NO_PARITY,
      .stopbits = USART_1_STOPBIT,
      .channelmode = USART_NORMAL_CHMODE
};
```

Tím, že každý pin mikroprocesoru mohl zastávat více funkcí, bylo potřeba jej nastavit na funkci USART. Poté bylo potřeba inicializovat sériovou linku s danou frekvencí mikroprocesoru (25 MHz) a nastavením výše zmíněných parametrů.

```
gpio_enable_module(USART_GPIO_MAP, sizeof(USART_GPIO_MAP) /
sizeof(USART_GPIO_MAP[0]));

usart_init_rs232(usart, &opt, BOARD_OSC0_HZ);
```

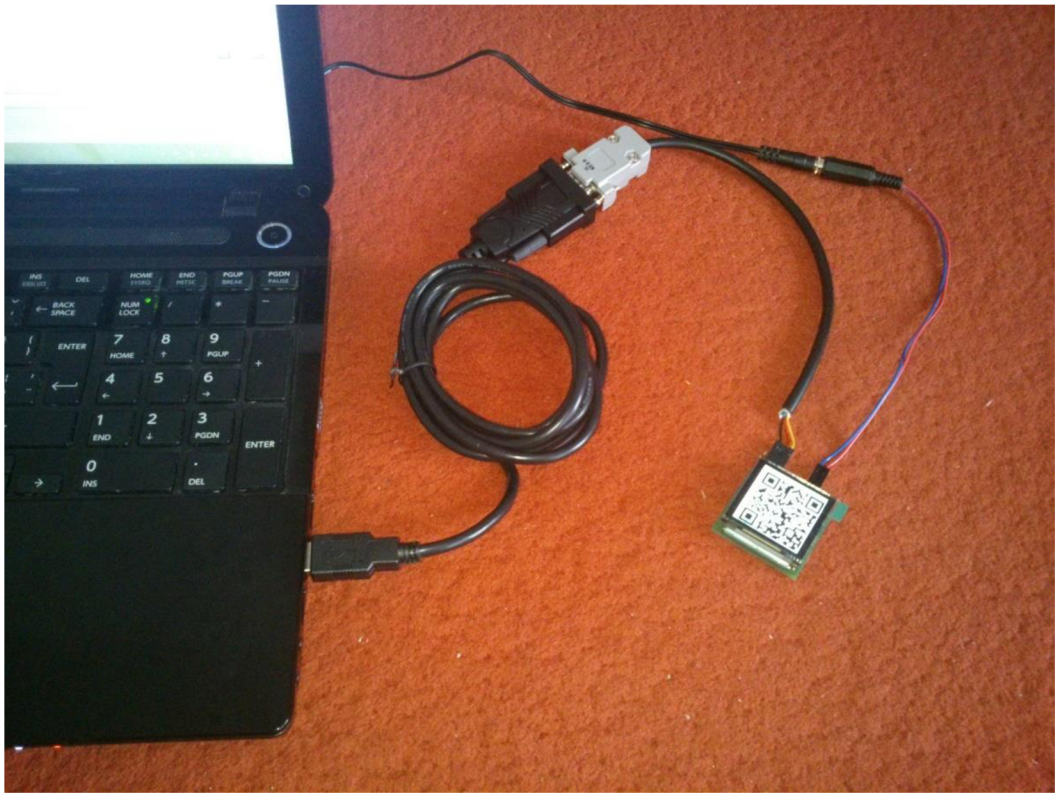
Nyní bylo možné komunikovat PC se zařízením pomocí sériové linky. Bylo potřeba nastavit, aby se každý příchozí znak z terminálu přečetl a uložil do pomocného pole. Je možno načíst právě tolik znaků, kolik je nastavený maximální počet `maxLen-2`. Jakmile došlo k odeslání řetězce klávesou „Enter“ (zapsání hexadecimální hodnoty 0x0d) nebo byl vyčerpán nastavený maximální počet znaků, bylo celé pole použito jako zdroj, pro zakódování do QR kódu a vykreslení na displej.

```
int usart_get_line(char* p, int maxLen)
{
      int c;
      int Len = 0;
      int rv;
      while(Len+2 < maxLen)
      {
            rv = usart_read_char(&AVR32_USART1, &c);
            if(rv == USART_RX_EMPTY)
            continue;
            if(rv == USART_RX_ERROR)
            {
                  continue;
            }
            if(c == 0x0D) break;
            p[Len] = c;
            Len++;
      }
      p[Len] = 0;
      return(Len);
}
```

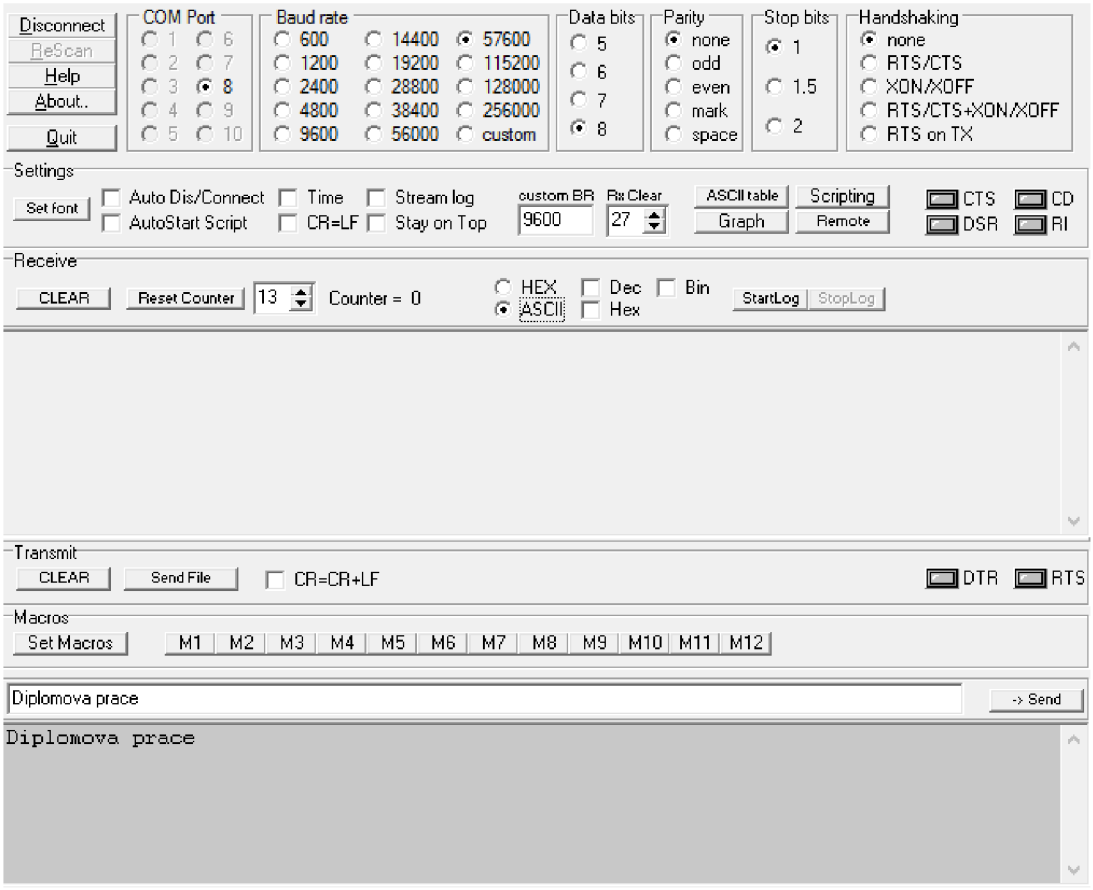
Zde lze vidět sekvenci výše zmíněných funkcí, pro celý průběh aplikace na zařízení:

```
strcpy(QRCODE_TEXT, "Tomas Lakomy");

main_1();
clearScreen();
qr();

for(;;)
{
      usart_get_line(QRCODE_TEXT, sizeof(QRCODE_TEXT));
      clearScreen();
      qr();
 }
```

Po připojení zařízení k napájení se zobrazí úvodní QR kód a zařízení čeká na zaslání informace. Propojením s PC a zasláním nějaká informace z terminálu, lze tuto informaci vidět v podobě QR kódu na displeji. Zařízení je nastaveno tak, že nadále čeká na další zadanou informaci, kterou by zobrazilo.



Obrázek 37: Propojení zřízení s PC



Obrázek 38: Zápis řetězce do terminálu

Obrázek 39: Zobrazení QR kódu na displeji

ZÁVĚR

V této diplomové práci jsem se nejprve seznámil s QR kódy, jejich strukturou, funkcí, použitím a s jednotlivými kroky při jejich tvoření. Pro jejich vytváření lze využít mnoho funkcí a knihoven. Nejprve bylo potřeba najít volně dostupnou knihovnu pro generování těchto kódů, zprovoznit jí a poté i patřičně upravit. Speciální funkcí pro počítání paměťové náročnosti, byl zjišťován potřebný paměťový prostor mikroprocesoru pro výpočetní funkce. Tato náročnost je závislá na generování a vykreslování různých druhů QR kódů. Podle této získané hodnoty bylo možné vybrat mikroprocesor s dostačující hodnotou paměti. Dále bylo důležité zjistit, zda simulačně odzkoušený program pro generování QR kódů je možno využít i na testovacím přípravku. Práci bylo možno zjednodušit využitím vývojového kitu místo kontaktního nepájivého pole. Byl tedy vybrán vývojový kit EVK1105 s mikroprocesorem AT32UC3A0512 a displejem ET024006DHU s rozlišením 240x320 pixelů. Nyní bylo potřeba patřičně nastavit displej a naprogramovat samotné vykreslování QR kódu.

V další části práce bylo nezbytné navrhnout vlastní zařízení pro zobrazování vygenerovaných QR kódů. Bylo navrhnuto a sestaveno schéma zapojení celého obvodu a vybrány jednotlivé součástky na výrobu. Na tomto zařízení byl použit displej DD-128128FC-6A s rozlišením 128x128 pixelů a místo mikroprocesoru AT32UC3A0512 byl použít pro toto zařízení vhodnější mikroprocesor AT32UC3B1512, který má daleko menší rozměry, což bylo pro tento případ vhodnější. Byl proveden návrh DPS a následně deska vyrobena a osazena součástkami. Programování mikroprocesoru probíhalo v několika částech. Nejprve povolení a příprava jednotlivých portů, následně inicializace displeje, vykreslování a v neposlední řadě byla nastavena komunikace PC se zařízením po sériové lince. Program pro generování QR kódů byl upraven z programu testovaného na vývojovém kitu. Výsledkem této práce je zařízení, které po zadání určité informace do terminálu v PC zobrazí tuto informaci na displej zakódovanou v podobě QR kódu.

LITERATURA

[1] *Ikaros Elektronický Časopis o Informačni Společnosti / Ústav Informačních Studií a Knihovnictví Praha* [online]. 2011 [cit. 2014-10-22]. ISSN 1212-5075. Dostupné z: http://ikaros.cz/qr-kody-jako-zvlastni-druh-dvourozmerneho-kodu

[2] CASSELMAN, Bill. How to Read QR Symbols Without Your Mobile Telephone. In: *Feature Column*[online]. 2013 [cit. 2014-10-28]. Dostupné z: http://www.ams.org/samplings/feature-column/fc-2013-02

[3] KLEGA, Vratislav. *Chip: počítačový magazín* [online]. 05/2008. Praha: Vogel Publishing, 2008, roč. 2008 [cit. 2014-11-07]. ISSN 1210-0684. Dostupné z: http://www.chip.cz/casopis-chip/earchiv/vydani/r-2008/chip-05-2008/qr-kody-sejmout-a-na-web-05-08/

[4] QR Code generator. *Unitag* [online]. 2011. vyd. 2011 [cit. 2014-11-07]. Dostupné z: https://www.unitag.io/qrcode

[5] ČÍŽEK, Jakub. *Zapomeňte na škaredé černobílé QR kódy* [online]. Praha: Computer Press a.s., 2013 [cit. 2014-11-07]. ISSN 1212-8554. Dostupné z: http://www.zive.cz/clanky/zapomente-na-skarede-cernobile-qr-kody/sc-3-a-168530/default.aspx

[6] EBY, Carolyn. *Thonky: QR Code Tutorial: Introduction* [online]. 2013 [cit. 2014-11-14]. Dostupné z: http://www.thonky.com/qr-code-tutorial/introduction/

[7] *Wikiversity: Reed–Solomon codes for coders* [online]. 2014 [cit. 2014-11-14]. Dostupné z: http://en.wikiversity.org/wiki/Reed%E2%80%93Solomon_codes_for_coders

[8] QR Codes. *Types of QR Codes* [online]. 2014 [cit. 2014-11-20]. Dostupné z: http://www.qrcode.com/en/codes/

[9] Webtech Desighner. *Texas Instruments* [online]. 2015 [cit. 2015-05-20]. Dostupné z: http://webench.ti.com/webench5/power/webench5.cgi?VinMin=20&VinMax=30&O1V=13&O1I=0.05&base_pn=LMR14203X&AppType=None&op_TA=50&lang_chosen=en_US&optfactor=3&action=simulate

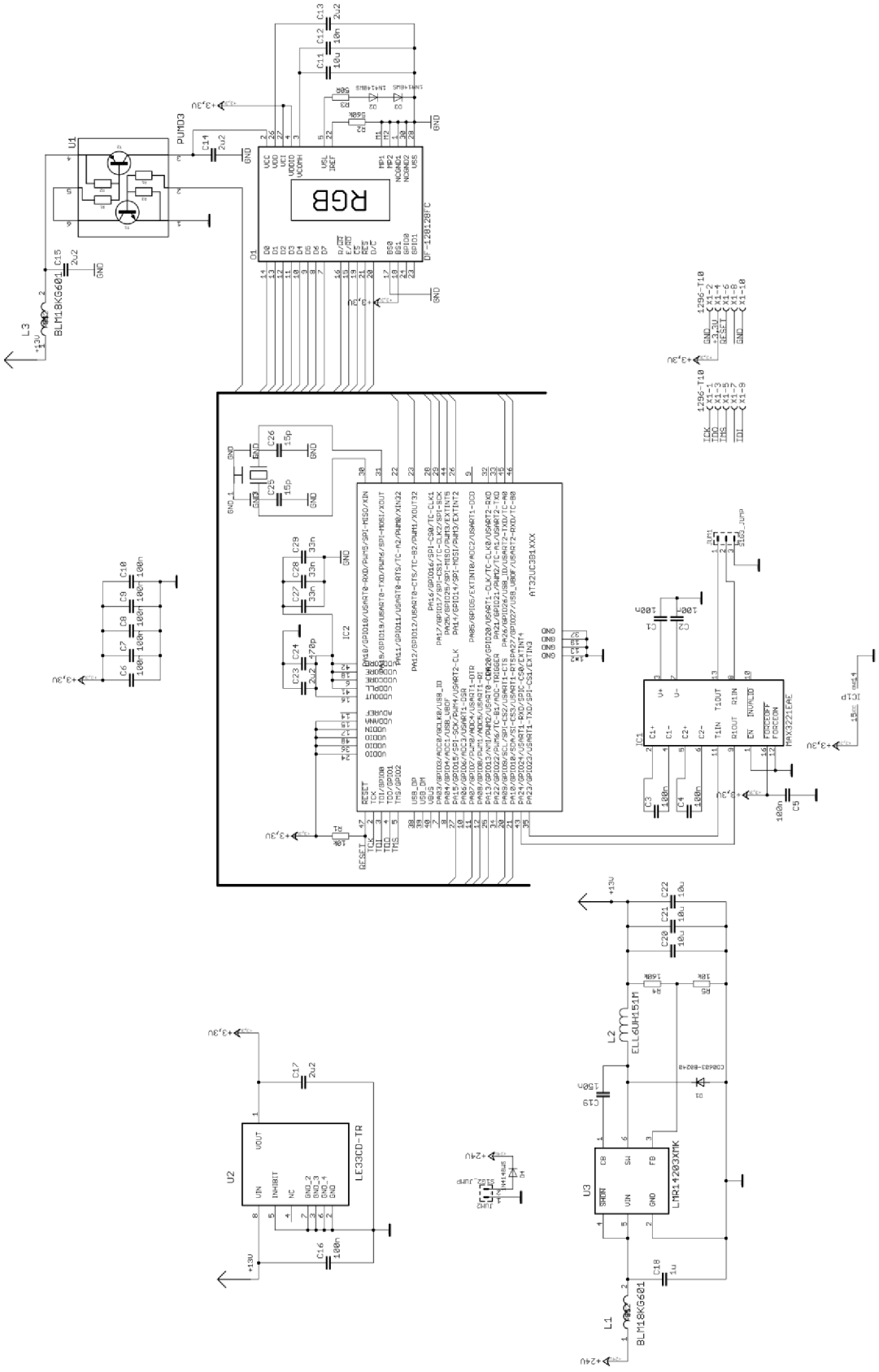
SEZNAM SOUČÁSTEK

Označení	Hodnota	Pouzdro	Popis
C1	100n	0603	Keramický kondenzátor
C2	100n	0603	Keramický kondenzátor
C3	100n	0603	Keramický kondenzátor
C4	100n	0603	Keramický kondenzátor
C5	100n	0603	Keramický kondenzátor
C6	100n	0402	Keramický kondenzátor
C7	100n	0402	Keramický kondenzátor
C8	100n	0402	Keramický kondenzátor
C9	100n	0402	Keramický kondenzátor
C10	100n	0402	Keramický kondenzátor
C11	10µ	0603	Keramický kondenzátor
C12	10n	0603	Keramický kondenzátor
C13	2,2µ	0603	Keramický kondenzátor
C14	2,2µ	0603	Keramický kondenzátor
C15	2,2µ	0603	Keramický kondenzátor
C16	100n	0603	Keramický kondenzátor
C17	2,2µ	0603	Keramický kondenzátor
C18	1µ	0603	Keramický kondenzátor
C19	150n	0603	Keramický kondenzátor
C20	10µ	0805	Keramický kondenzátor
C21	10µ	0805	Keramický kondenzátor
C22	10µ	0805	Keramický kondenzátor
C23	2,2µ	0805	Keramický kondenzátor
C24	470p	0603	Keramický kondenzátor
C25	15p	0402	Keramický kondenzátor
C26	15p	0402	Keramický kondenzátor
C27	33n	0402	Keramický kondenzátor
C28	33n	0402	Keramický kondenzátor
C29	33n	0402	Keramický kondenzátor
D1	CD0603-B0240	0603	Schottkyho dioda
D2	1N4148WS	SOD323	Dioda
D3	1N4148WS	SOD323	Dioda
D4	1N4148WS	SOD323	Dioda
IC1	MAX3221EAE	SSOP16	RS-232 převodník
IC2	AT32UC3B1512	QFN48	Mikroprocesor
JUM1	S1G3	S1G3	Konektor
JUM2	S1G2	S1G2	Konektor
JUM3	1296-T10	1296-10	Konektor
L1	BLM18KG601	0603	Tlumivka

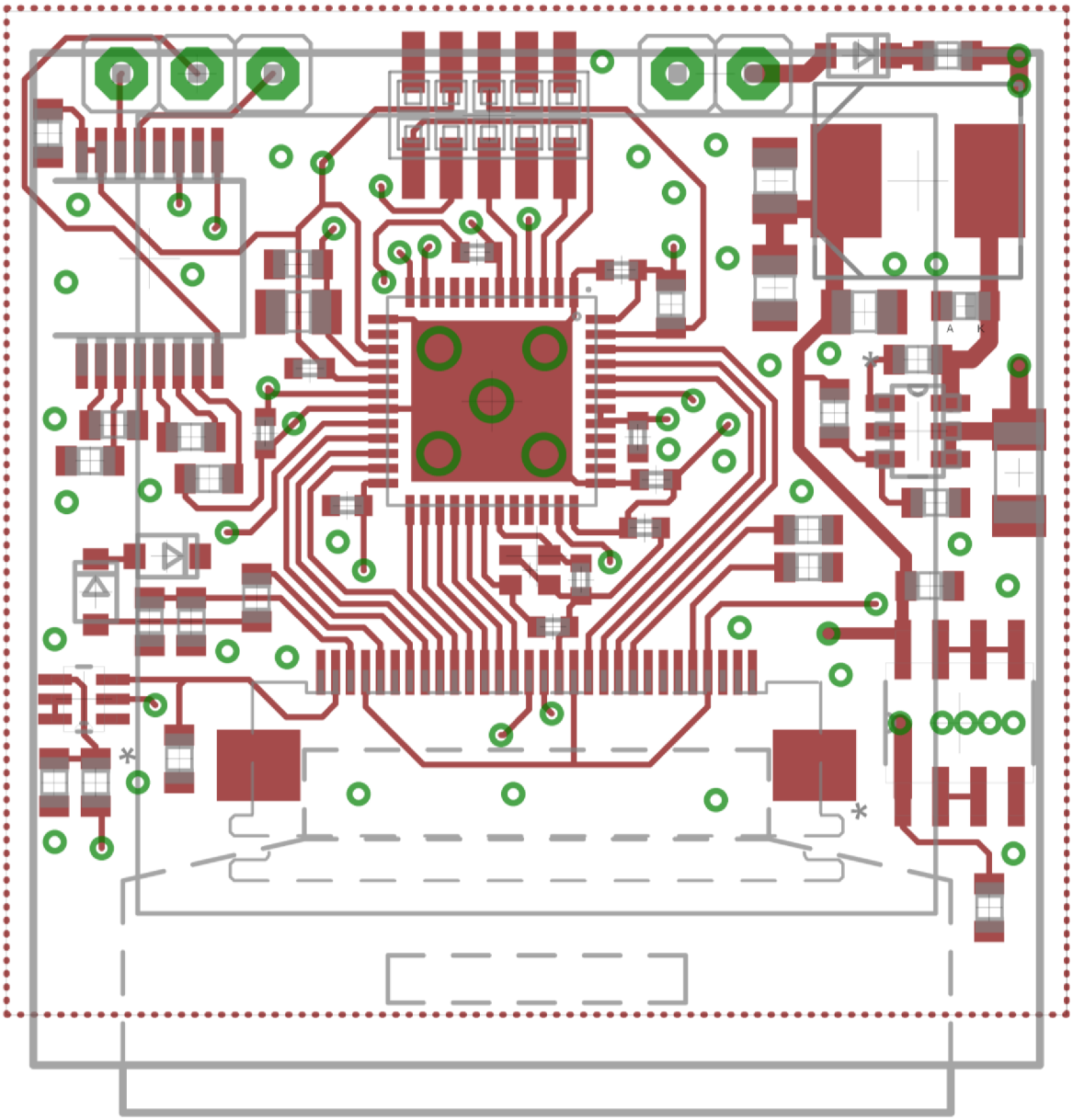
Označení	Hodnota	Pouzdro	Popis
L2	ELL6UH151M	ELL6XH	Tlumivka
L3	BLM18KG601	0603	Tlumivka
O1	DF-128128FC	DF-128128FC	Displej
R1	10k	0603	Rezistor
R2	560k	0603	Rezistor
R3	50R	0603	Rezistor
R4	160k	0603	Rezistor
R5	10k	0603	Rezistor
U1	PUMD3	SOT363	NPN a PNP tranzistor
U2	LE33CD-TR	SO8	Napěťový stabilizátor
U3	LMR14203XMK	SOT23	Step-Down měnič
X	25MHz	T3	Krystal

SEZNAM PŘÍLOH

Příloha A: Schéma zapojení



Příloha B: Návrh desky - Top vrstva



Příloha C: Návrh desky - Bottom vrstva