ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta elektrotechnická
Katedra elektroniky a informačních technologií

BAKALÁŘSKÁ PRÁCE

Řadiče pro malé OLED zobrazovače

Autor práce: **Michal Dykas**
Vedoucí práce: **Ing. Petr Weissar, Ph.D.**

2022

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta elektrotechnická

Akademický rok: 2021/2022

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Michal DYKAS**
Osobní číslo:	**E19B0004P**
Studijní program:	**B2644 Aplikovaná elektrotechnika**
Studijní obor:	**Aplikovaná elektrotechnika**
Téma práce:	**Řadiče pro malé OLED zobrazovače**
Zadávající katedra:	**Katedra výkonové elektroniky a strojů**

Zásady pro vypracování

1. Prozkoumejte řadiče používané pro OLED zobrazovací panely, uvažujte barevné i černobílé varianty.
2. Porovnejte podporované rozlišení, barevnou hloubku, způsoby komunikace, napájecí napětí a spotřebu, příp. další parametry.
3. Vytvořte vzorové příklady použití v jazyce C/C++ pro mikroprocesory ARM CortexM (příp. ESP32).

Rozsah bakalářské práce: **30 – 40**
Rozsah grafických prací: **dle doporučení vedoucího**
Forma zpracování bakalářské práce: **elektronická**

Seznam doporučené literatury:

Yiu, Joseph. The Definitive Guide to ARM® Cortex®-M3 and Cortex®-M4 Processors, Elsevier Science & Technology, 2013.
Online dokumentace výrobců.

Vedoucí bakalářské práce: **Ing. Petr Weissar, Ph.D.**
Katedra elektroniky a informačních technologií

Datum zadání bakalářské práce: **8. října 2021**
Termín odevzdání bakalářské práce: **26. května 2022**

Západočeská univerzita v Plzni
L.S.
Fakulta elektrotechnická

Prof. Ing. Zdeněk Peroutka, Ph.D.
děkan

Prof. Ing. Václav Kůs, CSc.
vedoucí katedry

V Plzni dne 8. října 2021

Abstrakt

Předkládaná bakalářská práce se zaměřuje na shromáždění informací ohledně řadičů pro malé OLED zobrazovače a následně jejich praktické řízení. Obsahuje základní informace o OLED zobrazovačích, přiblížení několika používaných řadičů a všeobecný přehled. Dále zkoumá způsoby komunikace a principy řízení a programování. Nakonec jsou tyto principy prakticky realizovány.

Klíčová slova

OLED zobrazovač, řadič, komunikační rozhraní

Abstract

The presented bachelor thesis is focused on information gathering and practical control of controllers for small OLED displays. It contains basic information about OLED displays, introduction of some used controllers and general overview. Further it research communication options and principles of controlling and programming. These principles are implemented in the end.

Key Words

OLED panel, controller, communication interface

Poděkování

Tímto bych velmi rád poděkoval vedoucímu práce panu Ing. Petru Weissarovi Ph.D. za poskytnutí potřebných prostředků a materiálů, cenné rady, vstřícnost a ochotu během zpracovávání této práce.

Obsah

Seznam symbolů a zkratek

Značka	Popisek	Jednotka
U_{DD}	Napájení polovodičové logiky	[V]
U_{CC}	Napájení panelu	[V]
U_{DDIO}	Napájení rozhraní řídící jednotky	[V]

OLED	Organic Light Emitting Diode
LED	Light-Emitting Diode
LCD	Liquid Crystal Display
PMOLED	Passive Matrix OLED
AMOLED	Active Matrix OLED
DSTN	Double Super Twisted Nematic
STN	Super Twisted Nematic
TFT	Thin Film Transistor
CMOS	Comlpementary Metal Oxide Semiconductor
RGB	Red Green Blue
RAM	Random Access Memory
SRAM	Static RAM
GDDRAM	Graphic Display Data RAM
ROM	Read Only Memory
EEPROM	Electrically Erasable Programmable ROM
GAC	Graphic Accelerating Command
4DGL	4 Dimensional Graphics Language
DMA	Direct Access Memory
SPI	Serial Peripheral Interface
I^2C	Inter Integrated Circuit
UART	Universal Asynchronous Receiver-Transmitter
SCK	Serial Clock
MISO	Master In, Slave Out
MOSI	Mastar Out, Slave In
SSEL	Slave Select
SSPBUF	Serial Input Buffer
SSPSR	Serial Input Shift Register
SDA	Synchronous Data
SCL	Synchronous Clock
GND	Ground
D/C	Data/Command
R/W	Read/Write
RES	Reset
CS	Chip Select
ACK	Acknowledge
FPU	Floating Point Unit
DSP	Digital Signal Processing
MPU	Memory Protection Unit
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
PWM	Pulse Width Modulation
SDIO	Secure Digital Input Output
GPIO	General Purpose Input Output

BR	Baud Rate
MSTR	Master Selection
SSI	Internal Slave Select
SSM	Software Slave Management
CPOL	Clock Polarity
CPHA	Clock Phase
DR	Data Register
CR	Control Register
TRISE	Time Rise
CCR	Clock Control Register
SR	Status Registr

Úvod

Malé OLED zobrazovače se v dnešní době používají ve spoustě případech, kde je potřeba zobrazovat jakékoli grafické i textové informace. Díky jejich parametrům získávají čím dál více na oblibě v porovnání s konkurenčními LCD zobrazovači. Můžeme se s nimi setkat například u MP3 a MP4 přehrávačů, kalkulaček, hodinek, ale i u světelných efektů a přístrojových desek automobilů.

Práce má za cíl přiblížit čtenáři technologii OLED, prozkoumat používané řadiče pro malé OLED zobrazovací panely v různých variantách, porovnat jejich podporované rozlišení, barevné hloubky, způsoby komunikace, napájecí napětí a spotřebu. Dále shrnout podstatné informace sloužící k pochopení principů komunikačních rozhraní mezi řadičem a mikrokontrolerem a k pochopení způsobů zobrazování požadovaných vzorů.

V neposlední řadě má za cíl vytvořit praktickou realizaci prozkoumaných a shrnutých informací ve vzorových příkladech použitím programovacího jazyka C/C++ pro mikroprocesory s jádrem ARM CortexM pro řadiče SSD1306 se zabudovanými panely o různém rozlišení a využívajícími různé druhy komunikačního rozhraní.

1 Řadiče pro malé OLED zobrazovače

1.1 OLED zobrazovače

OLED zobrazovače, využívají technologie LED diod, kdy emise světelných částic je produkována pomocí molekul organických látek. Celý zobrazovač je poté vyroben zapojením těchto diod, kdy každá z nich představuje jeden pixel, do určitého počtu řádků a sloupců. Počet těchto pixelů definuje konkrétní rozlišení. Každý jednotlivý pixel je ovládán samostatně a kdykoliv je na něj přiveden elektrický proud, začne vyzařovat své vlastní světlo, na rozdíl od LCD technologie, kdy vyzařování světla obstarává podsvícení. Díky tomuto nají OLED zobrazovače vysokou kvalitu obrazu, jasné barvy, a především velmi vysoký kontrast [1].

Obr. 1 Malý OLED zobrazovač 128x64 značky LaskaKit. Převzato z [2].

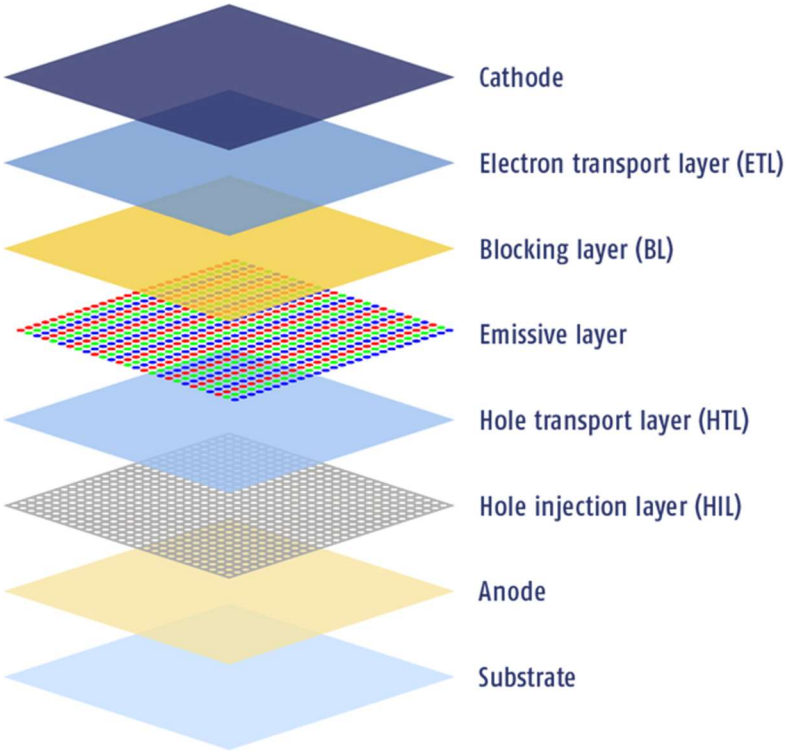
1.2 Využití OLED zobrazovačů

Oproti LCD technologii mají OLED zobrazovače díky absenci podsvícení několik výhod. Například pokud je potřeba vytvořit černou barvu, jednoduše stačí daný pixel zhasnout, čímž se vytvoří tzv. „dokonalá černá“. Obecně mají OLED nižší energetickou náročnost při zobrazení tmavších barev. Dále pak téměř dokonalé pozorovací úhly, nízká hmotnost a pružnost. Výhodou OLED zobrazovačů je pak také fakt, že mohou být naneseny prakticky na jakémkoliv podkladu.

OLED zobrazovače se dnes využívají prakticky kdekoliv, především však v mobilních zařízeních. V poslední době se jejich využití promítlo i do oblasti hodinek a televizí. Každoročně se vyrobí přes 500 milionů OLED zobrazovačů od různých výrobců a díky této konkurenci se postupně zlepšuje i samotná technologie, která nabízí lepší obraz, menší rozměry s vyšším rozlišením a další využití [1].

1.3 Struktura OLED

Základní struktura OLED je poměrně jednoduchá. Mezi anodu a katodu je umístěn organický emitor. Toto řešení by ovšem bylo poněkud neefektivní, a proto se do OLED za účelem zvýšení efektivity a dlouhé životnosti přidávají další mezivrstvy, jako vrstva přemisťující elektrony a blokovací vrstva. Celá tato struktura je následně vložena mezi obě elektrody a následně připevněna na podklad a připojena na řídící elektroniku [1].



Obr. 2 Základní struktura OLED. Převzato z [1].

1.4 Rozdělení OLED zobrazovačů

1.4.1 PMOLED

PMOLED zobrazovač neboli OLED s pasivní matricí využívá velmi jednoduchého principu ovládání, ve kterém je každý řádek ovládán postupně jeden po druhém. Neobsahuje žádný úložný kondenzátor, a tím pádem je každý pixel v každém řádku více méně většinu času vypnutý, což později způsobí úbytky napětí. Kvůli nutnosti tyto úbytky kompenzovat, je potřeba využít vyšší napětí, aby pixely mohly zářit, tak jak je potřeba. Například pokud je k dispozici deset řádků, musí řádek, který je zapnutý, svítit desetkrát jasněji než řádky ostatní. Takže zatímco je PMOLED poměrně snadné vyrobit, díky potřebnému vysokému napětí nejsou tolik efektivní, jejich rozlišení a velikost jsou poměrně omezené, protože čím více je k dispozici řádků, tím vyšší je potřebné napětí a materiály použité při výrobě trpí nižší životností. Z těchto důvodů jsou PMOLED obvykle malé a používají k zobrazení znaků nebo malých ikon v MP3 přehrávačích, pomocných displejích mobilních telefonů apod [3].

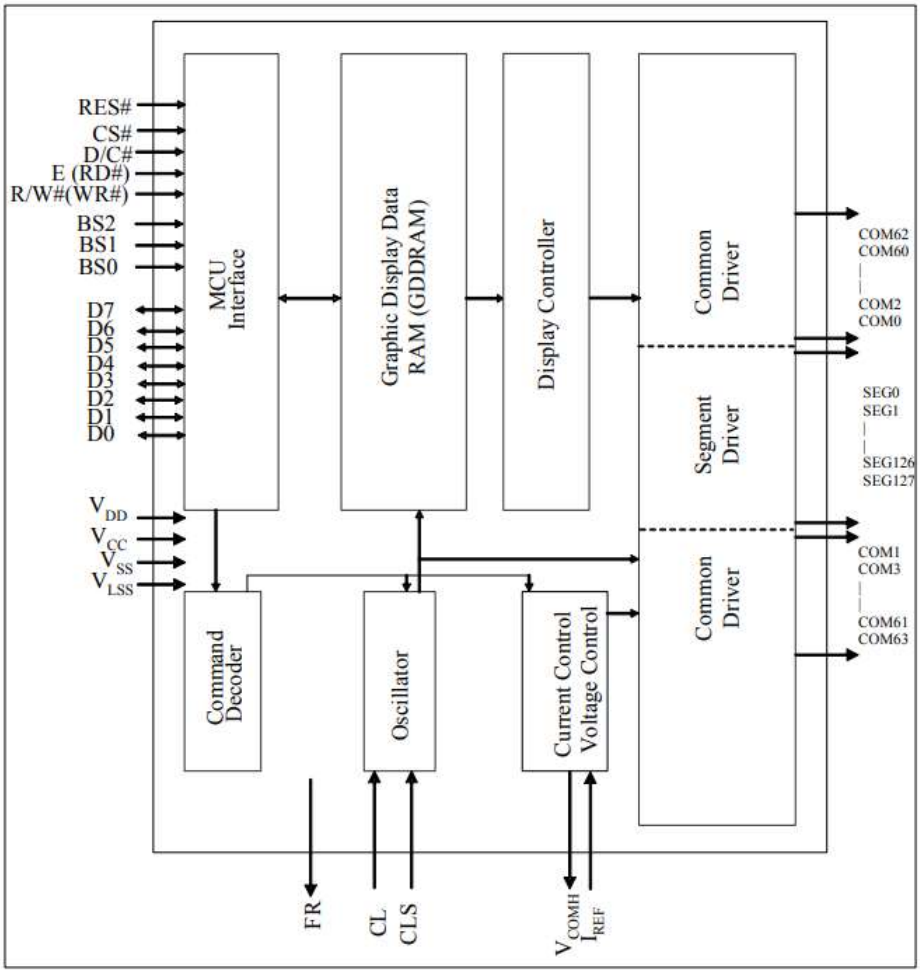
1.4.2 AMOLED

AMOLED zobrazovač neboli OLED s aktivní matricí je řízen TFT tranzistory, které obsahují i úložný kondenzátor sloužící k udržení stavu řádkových pixelů. Díky tomuto nedochází k úbytkům napětí jako v případě PMOLED, zamezuje se problikávání jednotlivých pixelů panelu, kde se rychle v cyklu mění jejich svit, a celkově je proto možné vyrobit zobrazovač větších rozměrů. Mezi další výhody patří vyšší zobrazovací frekvence a ostřejší barvy. Naopak nevýhodou je kvůli složitější struktuře vyšší cena a celkově náročnější výroba. Díky svým výhodám se využívají pro aplikace náročné na grafické zobrazování, jako například video a grafika u mobilních telefonů, televizí a monitorů [4].

1.5 Řadiče pro malé OLED zobrazovače

Řadič pro OLED zobrazovač je řídící jednotka, která se stará o řízení samotného OLED zobrazovače, umožňuje použití tenčích a bezrámečkových zobrazovačů, které jsou tím pádem flexibilnější a poskytují širší škálu barev a věrnější zobrazovací vlastnosti.

Ovládání zobrazovače za pomocí řadiče probíhá vysíláním řídících signálů a následně požadovaných dat na zobrazovací panel ve formě elektrických signálů, které reprezentují obrazové prvky, jako jsou například písmena a obrázky. Samotný řadič je umístěn v OLED zobrazovači.



Obr. 3 Blokový diagram řadiče SSD1306. Převzato z [5].

1.5.1 Řadič SSD1306

Řadič SSD1306 je jednočipový CMOS OLED řadič pro řízení grafických zobrazovacích panelů tvořených bodovými matricemi z OLED. SSD1306 obsahuje ovládání jasu, paměť RAM a oscilátor, čímž snižuje počet potřebných externích komponent a snižuje spotřebu elektrické energie. Data a příkazy jsou odesílány z řídící jednotky prostřednictvím volitelného paralelního, I^2C nebo periferního sériového rozhraní. Je vhodný pro spoustu kompaktních přenosných zařízení, jako například pomocný displej mobilního telefonu, MP3 přehrávače, kalkulačky apod. Parametry řadiče:

- Rozlišení: 128 x 64
- Napájení:
  - U_{DD} = 1,65V do 3,3V pro polovodičovou logiku
  - U_{CC} = 7V do 15V pro řízení panelu
- Displej:
  - Výstupní napětí řízení OLED, max. 15V
  - Proud pro jednotlivý segment, max 100uA
  - Běžný pokles proudu, max 15mA
  - 256krokové ovládání jasu
- Zabudovaný 128 x 64 bit SRAM buffer pro OLED
- Volitelné komunikační rozhraní:
  - 8bitové 6800/8080 paralelní rozhraní
  - ¾ sériové periferní rozhraní
  - I^2C rozhraní
- Funkce posouvání obrazu v horizontálním i vertikálním směru
- Synchronizační signál pro zápis na RAM
- Programovatelná snímková frekvence
- Přemapování řádků a sloupců
- Zabudovaný oscilátor
- Široký rozsah provozních teplot: -40°C do 85°C [6]

1.5.2 Řadič SSD1331

Řadič SSD1331 je jednočipový CMOS OLED řadič podporující barevnou 96RGB x 64 bodovou matrici. Je navržený pro běžný katodový OLED panel. Obsahuje paměť GDDRAM. Ke komunikaci využívá 8, 9 a 16bitové 6800/8080 paralelní rozhraní a sériové periferní rozhraní. Aby byla umožněna komunikace mezi řídící jednotkou o nižším napětí, má zvlášť oddělené napájení pro vstupní a výstupní rozhraní. Je vhodný pro mobilní telefony, MP3 a MP4 přehrávače. Parametry řadiče:

- Rozlišení: 96RGB x 64
- Zabudovaný 96 x 64 x 16 bit GDDRAM buffer podporující 65k barevnou hloubku
- Napájení:
  - U_{DD} = 2,45V do 3,5V pro polovodičovou logiku
  - U_{CC} = 8V do 18V pro řízení panelu
  - U_{DDIO} = 1,6V do U_{DD} pro rozhraní řídící jednotky
- Proud pro jednotlivý segment: 200uA
- Běžný pokles proudu: 60mA
- 256krokové řízení kontrastu pro každou barevnou komponentu a 16krokové řízení proudu
- Volitelné komunikační rozhraní:
  - 8, 9, 16bitové 6800 paralelní rozhraní
  - 8, 9, 16bitové 8080 paralelní rozhraní
  - Sériové periferní rozhraní
- Funkce změny barvy
- Sada GAC příkazů pro souvislý horizontální, vertikální a diagonální posun obrazu
- Programovatelná snímková frekvence
- Široký rozsah provozních teplot: -40°C do 85°C [7]

1.5.3 Řadič MLX81130

Řadič MLX81130 je plně integrovaný automobilový OLED řadič s vysokorychlostní robustní komunikačním rozhraním MeLiBu, které umožňuje vývoj výkonných a nákladově optimalizovaných světelných systémů. Obsahuje všechny potřebné komponenty k řízení 25 OLED s regulovatelným a programovatelným proudem až do 30mA. V případě nutnosti vyššího proudu je možné výstupy přemostit. Je vysoce kompatibilní s jinými MeLiBu integrovanými obvody, díky čemuž lze využít OLED ve stejné aplikaci s LED. Vhodným využitím tohoto řadiče jsou animované nebo statické světelné efekty interiéru a exteriéru aut, zadní sdružené svítilny a malé OLED zobrazovače v interiéru i exteriéru aut. Parametry řadiče:

- 25 OLED programovatelných do 30mA
- Napájení: 5,5V do 18V
- Komunikační rozhraní:
  - DMA
  - UART
- Paměť:
  - 32KB Flash
  - 2KB RAM
  - 512B EEPROM
  - Systém ROM
- 8bitové řízení intenzity světla
- Ukládání kalibračních dat OLED přímo v řadiči
- Programovatelné zkalibrované zdroje proudu pro OLED
- Přímé řízení OLED bez přídavných vnějších komponent
- Diagnostika OLED
- Aktualizace OLED v reálném čase bez zpoždění
- Provozní teplota: -40°C do 125°C [8]

1.5.4 Řadič GOLDELOX

GOLDELOX je zabudovaný grafický řadič s grafickými, textovými obrázkovými a animačními funkcemi navržený ke komunikaci s mnoha populárními OLED panely. Potřebná data a řídící signály jsou přenášena z čipu přes rozhraní přímo do zobrazovače. Patří do rodiny řadičů programovatelných vysoce pokročilým jazykem 4DGL, který je velmi jednoduchý na naučení a zároveň schopný řešit mnoho grafických aplikací. Všechny knihovny využívané k ovládání zobrazovačů implementují a sdílejí své funkční rozhraní, proto lze jedno řešení využít i u různých typů zobrazovačů, což nabízí výhodu úspory času a financí pro návrháře. Parametry řadiče:

- Rozlišení: 128 x 128
- Napájení:
  - U_{CC} = 3,3V, I_{ref} = 12mA
- Komunikační rozhraní:
  - Sériové 8bitové 8080
  - SPI
- Paměť:
  - 10KB Flash
  - 510B RAM
- 2 x GPIO port podporující:
  - Digitální I/O
  - A/D převodník s rozlišením 8/10 bitů
  - Joystick
  - Kabel Dallas 1
- 1 x 32bitový a 4 x 16bitový časovač
- Podpora barevných obrazů
- Grafické nástroje [9]

Tabulka 1: Nejpoužívanější řadiče [10]

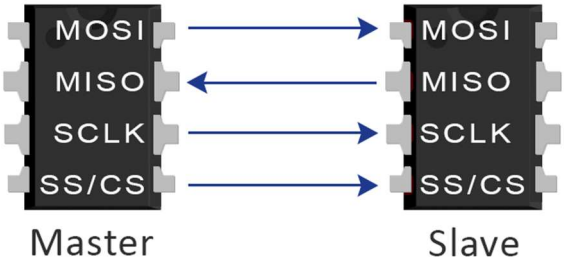
Typ	Rozlišení	Barva	Barevná hloubka	Komunikace	Napájení [V]	Spotřeba [mW]
SSD1306	128 x 64	Mono	2	SPI, I^2C	1,65 – 3,3	49,5
SSD1307	128 x 39	Mono	2	SPI, I^2C	1,65 – 3,3	132,0
SSD1309	128 x 64	Mono	2	SPI, I^2C	1,65 – 3,3	132,0
SSD1310	160 x 72	Mono	2	SPI, I^2C	1,65 – 3,5	268,8
SSD1311	100 x 32	Mono	2	SPI, I^2C	2,4 – 3,6	162,0
SSD1313	128/256 řádků ikon	Mono	2	SPI, I^2C	1,65 – 3,3	132,0
SSD1322	480 x 120	Mono	16	SPI	2,4 – 3,5	280,0
SSD1325	128 x 80	Mono	16	SPI	2,4 – 3,5	140,0
SSD1326	256 x 32	Mono	16	SPI, I^2C	2,4 – 3,5	87,5
SSD1327	128 x 128	Mono	16	SPI, I^2C	1,65 – 3,5	140,0
SSD1331	96 x 64	Barevný	65k	SPI	2,4 – 3,5	210,0
SSD1351	128 x 128	Barevný	262k	SPI	2,4 – 3,5	245,0
SSD1353	160 x 132	Barevný	262k	SPI	2,4 – 3,5	210,0
SSD1355	128 x 160	Barevný	262k	SPI	2,4 – 3,5	280,0
SSD7317	128 x 96	Mono	2	I^2C	1,65 – 3,5	268,8
MLX81130	25 OLED	Mono	2	DMA, UART	5,5 – 18	540,0
GOLDELOX	128 x 128	Barevný	-	SPI	3,3	39,6

1.6 Způsoby komunikace

Komunikační rozhraní se používá ke komunikaci neboli k přenosu dat a příkazů mezi řadičem a programovatelným mikrokontrolerem nebo řídící elektronikou bez použití mikrokontroleru. Mezi nejpoužívanější komunikační rozhraní v oblasti zobrazovačů patří SPI a I^2C.

1.6.1 SPI komunikace

Sběrnice SPI představuje jednu z forem sériových externích sběrnic sloužících pro vzájemné propojení dvou či více komunikačních uzlů, přičemž jeden uzel obvykle pracuje v režimu „master“, zatímco ostatní uzly pracují v režimu „slave“. Uzel, který vykonává řídící funkci, obsahuje generátor hodinového signálu, který je dále rozveden do všech ostatních uzlů, čímž je zprostředkován zcela synchronní a obousměrný přenos dat. Tento signál je rozváděn vodičem označován symbolem SCK. Kromě vodiče s hodinovým signálem jsou uzly dále propojeny dvojicí vodičů označovaných symbolem MISO a MOSI, pomocí nichž je umožněno obousměrně přenášet data. Posledním využívaným signálem je signál SSEL, díky němuž lze zvolit, který uzel bude sloužit jako master, a který jako slave. Všechny tyto čtyři signály vyžadují pro svou funkci pouze jednosměrné porty, což přispívá k jednoduchému a levnému využití této komunikace.



Obr. 4 SPI komunikace. Převzato z [11].

K řízení SPI jsou v nejjednodušším případě zapotřebí dva registry, datový záchytný registr SSPBUF a posuvný registr SSPSR. Do registru SSPSR je zapsán bajt, který byl správně přijat, ale doposud nebyl zpracován. Dále slouží k vysílání i příjmu jednoho bitu z celé osmice. Každý posun obsahu tohoto registru znamená, že se vysunutý bit v případě MOSI pošle na pin SDO a v případě MISO naopak logická hodnota přečtená z pinu SDI je zapsána do nejnižšího bitu posuvného registru. Uzel pracující jako master generuje hodinové impulsy, které jsou posílány po vodiči SCLK. Pomocí těchto impulzů je prováděna synchronizace vysílání i příjmu dat, to znamená časy, ve kterých dochází ke změně v SSPSR registru. Hodinový signál se rozděluje na konfigurační bity CKP a CKE

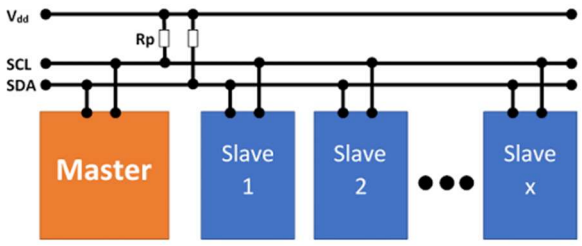
Konfiguračním bitem CKP je určena klidová úroveň, kdy se neprovádí vysílání dat a bitem CKE je určena vzestupná, či sestupná hrana hodinového signálu, kdy jsou data platná [12].

1.6.2 I²C komunikace

Jako v předchozím případě, i komunikace I²C využívá formu typu master-slave, ale v tomto případě není možné vybrat, který uzel bude master, a který slave, protože každému uzlu může být přiřazena jednoznačná adresa. Kromě elektrických charakteristik je ve specifikaci přesně stanoven i komunikační protokol. Sběrnice je tvořena dvojicí signálových vodičů. První signálový vodič SDA slouží pro oboustranný přenos dat a druhý signálový vodič SCL zařizuje synchronizaci pomocí hodinových impulzů. V praxi je nutné navíc přidat k této dvojici vodičů i společnou zem GND a vodiče SDA a SCL připojit přes pull-up rezistory na napájecí napětí, aby uzly v době své nečinnosti měli zvýšenou hodnotu napětí a setrvávaly v klidovém stavu.

Veškeré řízení sběrnice obstarává zařízení typu master, kterým může být pouze jedno zařízení, aby na sběrnici nedocházelo ke kolizím. Samotné zahájení komunikace vždy zahajuje master, který sníží napětí na vodiči SDA na logickou nulu, zatímco SCL je po určitou dobu na hodnotě logické jedničky. Tento stav nazývaný jako „start bit“ slouží k rozpoznání uzlů připojených na sběrnici. Ihned po vyslání start bitu začne master vysílat adresu uzlu, se kterým má nastavenou komunikaci a bit, který určuje směr přenosu. Každý bit musí před vysláním náběžné hrany hodinového impulzu ustálený stav.

Každému zařízení připojenému na sběrnici je přiřazena unikátní adresa, aby bylo jednoznačné, se kterým zařízením master komunikuje. U sběrnice typu I²C se využívá sedmibitová adresa přenášená v jednom bajtu, kdy poslední bit R/W určuje, zda se jedná o zápis, nebo čtení dat. Ihned po start bitu je možné zahájit přenos adresy. Master pošle po sběrnici všech sedm bitů adresy a bit R/W. Po přenosu všech osmi bitů provede slave porovnání přijaté adresy se svojí vlastní adresou. Pokud dojde ke shodě, v devátém cyklu hodinových impulzů se pošle potvrzovací příznak do zařízení master [13].

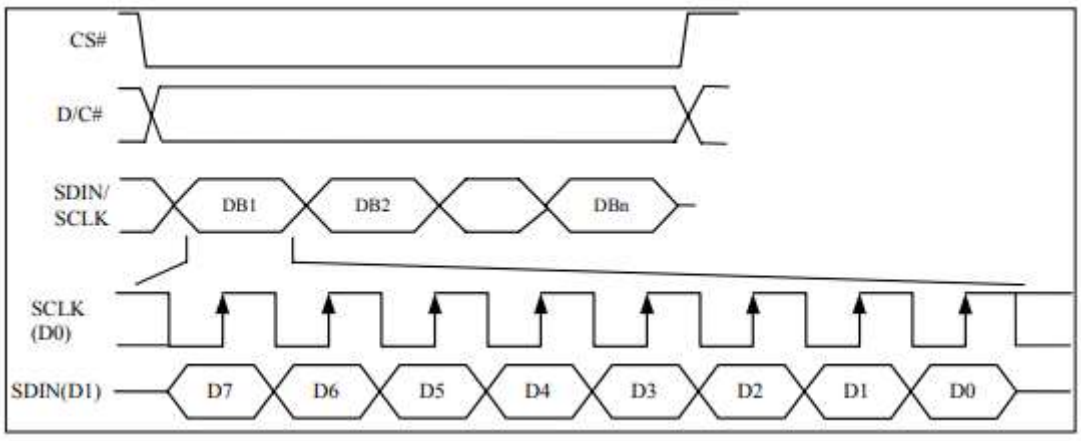


Obr. 5 I²C komunikace. Převzato z [14].

1.7 Inicializace SSD1306 a přenos dat a příkazů

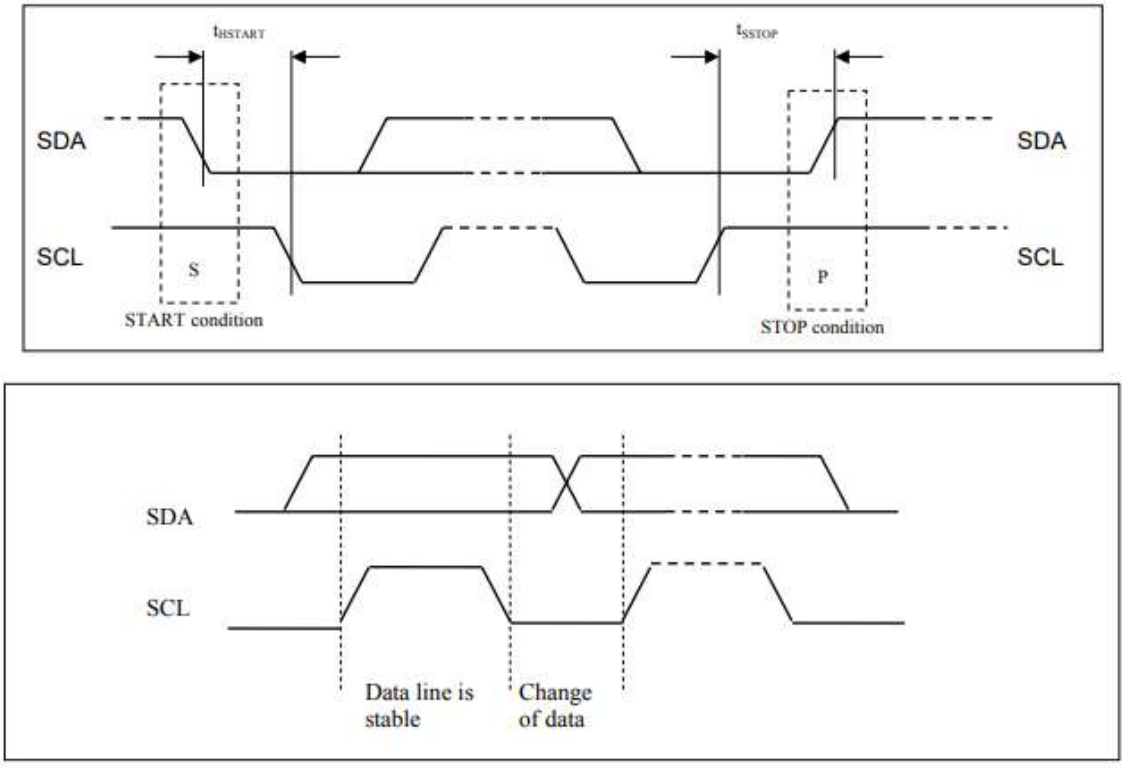
SSD1306 má interní příkazové registry, které se používají ke konfiguraci činnosti s mikroprocesorem. Po resetu řadiče by měli být registry nastaveny na příslušné hodnoty vzhledem k správné funkci aplikace. Registry jsou přístupné mikrokontroleru pomocí D/C# pinu u paralelního a SPI rozhraní, jehož hodnota určuje, jestli jsou posílána data, nebo příkazy nebo pomocí I^2C rozhraní s využitím R/W# pinu. Použití a hodnoty některých registrů závisí na konkrétní aplikaci. Pro úspěšnou inicializaci SSD1306 řadiče je zapotřebí následující sekvence úkonů. Po ustálení napájecího napětí je zapotřebí nejprve řadič resetovat. To se provede přivedením logické nízké úrovně na RES# pin a následně logické vysoké úrovně. Mezi těmito úkony je třeba vyvolat zpoždění minimálně tři mikrosekundy z důvodu ustálení napěťových úrovní. Ve chvíli, kdy je řadič resetován stačí poslat příkaz na zapnutí a následně už je možno nastavit zobrazovač podle osobních preferencí.

Přenos dat a příkazů se taktéž liší podle použitého komunikačního rozhraní. Při použití SPI rozhraní je nejprve potřeba nastavit D/C# pin. Nízká logická úroveň tohoto pinu reprezentuje posílání příkazu a vysoká logická úroveň naopak posílání dat. Následně je třeba nastavit pin CS# na nízkou logickou úroveň, aby bylo možno s řadičem komunikovat. Po nastavení komunikačních bitů je třeba nastavit časování přenosu. Datový pin je připojen na 8bitový posuvný registr, který s každou náběžnou hranou zapíše příchozí hodnotu a posune se o jeden bit a každou osmou náběžnou hranou zapíše data do grafické paměti k zobrazení. Nakonec je třeba nastavit přepnout CS# pin opět na vysokou logickou úroveň z důvodu prevence proti možné chybě.



Obr. 6 Procedura zápisu pomocí SPI. Převzato z [6].

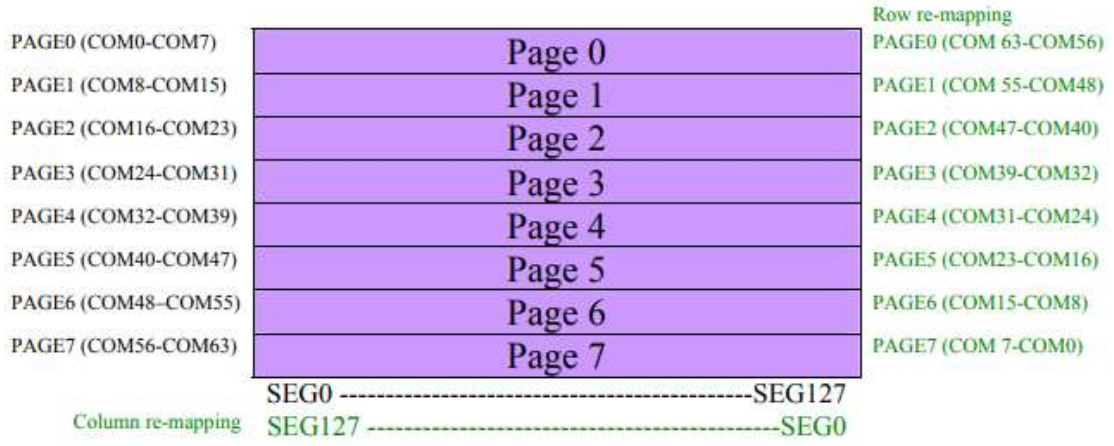
V případě I^2C rozhraní je přenos dat a příkazů poměrně složitější, neboť celý přenos realizují pouze piny SDA a SCL. Celý přenos začíná počáteční podmínkou, která je dána sestupnou hranou pinu SDA, zatímco pin SCL setrvává na vysoké logické úrovni. Následně je po rozhraní poslána adresa řadiče, kdy poslední bit určuje, jestli se jedná o zápis, či čtení. Po přenosu adresy je poslán řídící byte, obsahující primárně pouze bit Co, který určuje, zda následující data budou datová a D/C#. Dalších šest bitů je nastaveno na nízkou logickou úroveň. Následuje datový bajt, který reprezentuje data nebo příkaz podle nastavení bajtu předchozího. Pokud se jedná o data, budou zapsána na grafickou paměť. Po každém přeneseném bajtu je také přenesen bit ACK, který slouží pro kontrolu přenosu. Pokud již není potřeba přenášet data ani příkazy, celá komunikace se přeruší ukončující podmínkou, kdy pin SCL setrvává na vysoké logické úrovni při vzestupné hraně pinu SDA [6].



Obr. 7 Procedura zápisu pomocí I^2C. Převzato z [6].

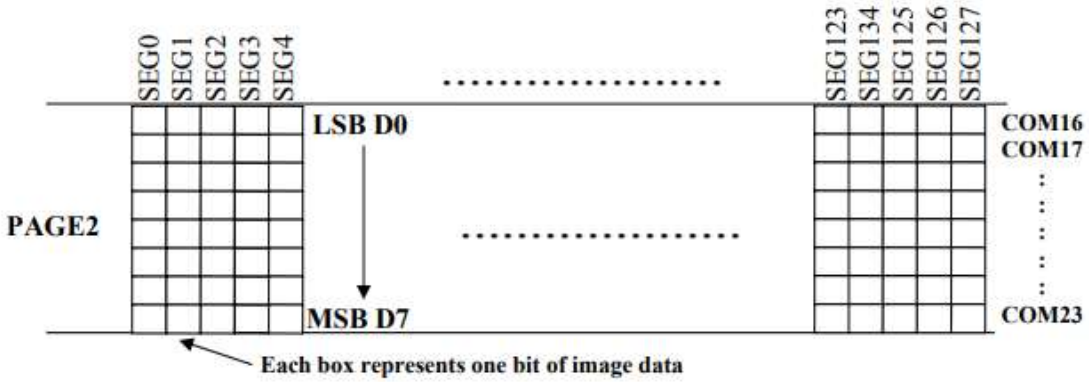
1.8 Zobrazení přenesených dat

K zobrazení dat slouží grafická GDDRAM paměť se softwarově určeným bitovým vzorem, který je určen k zobrazení. Velikost této paměti je určena samotným rozlišením konkrétního zobrazovače a je rozdělena na určitý počet stránek. Například u zobrazovače SSD1306 je dána jako 128 x 64 bitů a je rozdělena na osm stránek, a to PAGE0 až PAGE7.



Obr. 8 Grafické znázornění GDDRAM paměti. Převzato z [6].

Kdykoliv je datový bajt zapsán na GDDRAM, všechny řádky obrazových dat jsou zobrazeny na aktuální sloupec určený ukazatelem adresy na aktuální stránce. Nejnižší datový bit je zapsán do horního řádku, zatímco nejvyšší datový bit je zapsán do spodního řádku.



Obr. 9 Grafické znázornění zápisu bajtu. Převzato z [6].

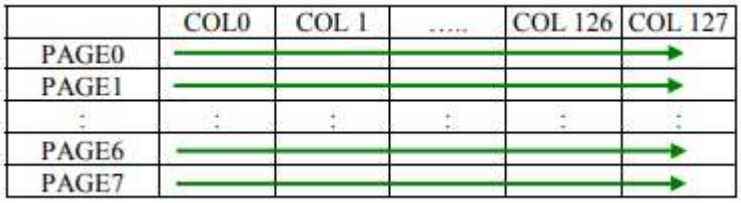
Z praktického hlediska je automatické nastavení ukazatelů poměrně neefektivní, proto je uživateli umožněno softwarově nastavit přemapování, kdy lze nastavit, který řádek a segment bude považován za počáteční a naopak. K tomuto lze využít vnitřní registr ukládající počátek [6].

Tabulka 2: Základní instrukce [6]

Příkaz	Adresa hexadecimálně	Popis
Display ON/OFF	AEh/AFh	Zapnutí/vypnutí OLED panelu
Set Lower Column	00h – 0Fh	Nastavení nejnižšího bitu 8bitového sloupce
Set Higher Column	10h – 1Fh	Nastavení nejvyššího bitu 8bitového sloupce
Set Page Start	B0h – B7h	Nastavení počáteční stránky
Set Memory Addressing Mode	20h	Nastavení adresního módu

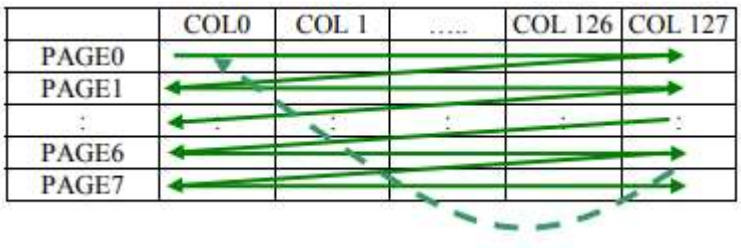
1.9 Adresní módy

Rozdělují se do tří odlišných módů. Stránkového, horizontálního a vertikálního. U stránkového se ukazatel sloupce automaticky inkrementuje o jedna. Ve chvíli, kdy je ukazatel roven poslednímu sloupci se automaticky vyresetuje a je znovu nastaven na první sloupec, zatímco ukazatel stránky zůstává stejný. Pokud chce uživatel použít jinou stránku, musí ji pomocí příkazu Set Page Start sám nastavit.



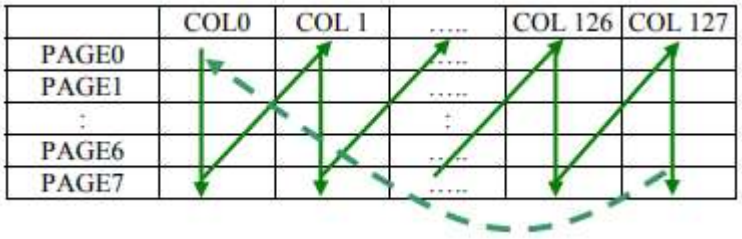
Obr. 10 Stránkový adresní mód. Převzato z [6].

Horizontální adresní mód funguje velmi podobně jako stránkový s tím rozdílem, že pokud ukazatel sloupce dosáhne své maximální adresy, dochází spolu s jeho resetem i k inkrementaci ukazatele stránky o jedna. Pokud je adresa obou ukazatelů maximální, oba se vyresetují.



Obr. 11 Horizontální adresní mód. Převzato z [6].

Poslední vertikální adresní mód funguje obráceně než předchozí dva. Po zápisu obrazových dat se místo ukazatele sloupce inkrementuje ukazatel stránky. Ve chvíli, kdy ukazatel stránky dosáhne konečné adresy, resetuje se a ukazatel sloupce se inkrementuje o jedna. Dosáhnou-li oba ukazatele své maximální adresy, taktéž se vyresetují [6].



Obr. 12 Vertikální adresní mód. Převzato z [6].

1.10 Mikrokontroler STM32F411RE

Mikrokontroler STM32F411RE je založen na vysoce výkonném jádře ARM Cortex pracujícím do frekvence až 100 Mhz. Je vybaven jednotkou FPU, která podporuje všechny instrukce a typy dat pro zpracování pomocí ARM jádra. Taktéž implementuje sadu DSP instrukcí a obsahuje ochranu paměti MPU pro zvýšení bezpečnosti aplikací. Pracuje s vysokorychlostní vestavěnou pamětí 512 kB Flash a 128 kB SRAM a s mnoho periferiemi připojenými ke dvěma APB sběrnicím, dvěma sběrnicím AHB a 32bitovou multi-AHM sběrnicové matici. Mezi periferie patří 12bitový A/D převodník, generátor hodinových pulzů, šest 16bitových časovačů, dva 32bitové časovače a PWM časovač. Pro přenos dat lze využít komunikační rozhraní SPI, I^2C, SDIO a USART. Provozní teplota se pohybuje v intervalu od -40 °C do 125 °C při napájecím napětí od 1,7 V do 3,6 V. Díky těmto vlastnostem a sadě úsporných režimů je tento mikroprocesor vhodný pro aplikace s řadiči pro malé OLED zobrazovače [5].

2 Praktická realizace

K praktické realizaci je využit řadič SSD1306 se zabudovaným panelem o rozlišení 128 x 32 za použití komunikačního rozhraní SPI a SSD1306 se zabudovaným panelem o rozlišení 128 x 64 za použití komunikačního rozhraní I^2C. Řadiče jsou řízeny mikroprocesorem STM32F411RE programovaným v programovacím jazyce C.

2.1 Softwarová inicializace OLED s SPI

```
void OLEDInit_SW(void)
{
   uint32_t tm = 0;
   OLED_RESET_LOW;
   for(tm = 0; tm < 50000; tm++)
     asm("nop");

   OLED_RESET_HIGH;
   for(tm = 0; tm < 50000; tm++)
     asm("nop");

   OLED_DC_LOW;
   OLED_CS_LOW;
   OLED_WriteCmd(SSD1305_DISPLAYON);
   OLED_WriteCmd(SSD1305_MEMORYMODE);
   OLED_CS_HIGH;
}
```

Softwarová inicializace je realizována funkcí OLEDInit_SW typu void. V první řadě je třeba zajistit zpoždění mezi jednotlivými resetovacími logickými úrovněmi. To je zařízeno pomocí deklarované proměnné tm typu 32bitového intu. Následně je na resetovací pin řadiče připojeného na pin D6 GPIOB portu mikrokontroleru přivedena nízká logická úroveň a vyvolané zpoždění, které by mělo být minimálně tři mikrosekundy. Po proběhlém zpoždění je na resetovací pin řadiče přivedena naopak vysoká logická úroveň a opět vyvolané zpoždění. Aby bylo možné s řadičem dále komunikovat a posílat příkazy, je třeba nastavit pin DC, který je připojen na pin D7 GPIOA portu, na nízkou logickou úroveň reprezentující příkaz a pin CS připojený na pin D10 GPIOB portu pro zvolení komunikace s řadičem. Prvním krokem pro zobrazování dat je zapnutí celého zobrazovače. Nejedná se ovšem o rozsvícení všech OLED na panelu, nýbrž přivedení napájecího napětí na jednotlivé segmenty. Po zapnutí panelu přichází na řadu volba paměťového adresního módu. V tomto

případě byl volen základní stránkový adresní mód. Nakonec je přerušena komunikace přivedením vysoké logické úrovně na pin CS.

2.2 Hardwarová inicializace OLED s SPI

```
void OLEDInit_HW(void)
{
   if (!(RCC->APB2ENR & RCC_APB2ENR_SPI1EN))
   {
     RCC->APB2ENR |= RCC_APB2ENR_SPI1EN;
     RCC->APB2RSTR |= RCC_APB2RSTR_SPI1RST;
     RCC->APB2RSTR &= ~RCC_APB2RSTR_SPI1RST;
   }
   SPI1->CR1 = SPI_CR1_BR_1 | SPI_CR1_BR_0;
   SPI1->CR1 |= SPI_CR1_MSTR;
   SPI1->CR1 |= SPI_CR1_SSI | SPI_CR1_SSM;
   SPI1->CR1 |= SPI_CR1_CPHA | SPI_CR1_CPOL;
   SPI1->CR2 = 0;
   SPI1->CR1 |= SPI_CR1_SPE;
   Nucleo_SetPinGPIO(OLED_PIN_SCK, ioPortAlternatePP);
   Nucleo_SetPinAFGPIO(OLED_PIN_SCK, 5);
   Nucleo_SetPinGPIO(OLED_PIN_MOSI, ioPortAlternatePP);
   Nucleo_SetPinAFGPIO(OLED_PIN_MOSI, 5);
   Nucleo_SetPinGPIO(OLED_PIN_CS, ioPortOutputPP);
   GPIOWrite(OLED_PIN_CS, 1);
   Nucleo_SetPinGPIO(OLED_PIN_RESET, ioPortOutputPP);
   GPIOWrite(OLED_PIN_RESET, 1);
   Nucleo_SetPinGPIO(OLED_PIN_DC, ioPortOutputPP);
   GPIOWrite(OLED_PIN_DC, 1);
}
```

Hardwarová inicializace je interpretována pomocí funkce OLEDInit_HW datového typu void. Nejprve je proveden reset a aktivace datové sběrnice, aby bylo možno dále přesně definovat nastavení rozhraní a komunikovat s řadičem. Tento krok je realizován v podmínce testující, zda již sběrnice není aktivovaná. Pokud tomu tak není, je nastaven povolovací bit v APB2ENR registru na SPI1, které je zapojeno na datovou sběrnici APB2. Následuje resetování pomocí APB2RSTR registru nejprve nastavením vysoké logické úrovně a poté nízké. Po resetu je realizována konfigurace samotného SPI rozhraní pomocí řídícího registru CR1 obsahujícím dělič vstupních hodin BR nastaveným na hodnotu 011, což odpovídá frekvenci 100Mhz APB2 sběrnice podělené 16. Následuje zvolení mikrokontroleru jako master zařízení přivedením vysoké logické úrovně do MSTR registru, nastavení práce s řízením slave režimu pomocí SSI a SSM registrů a volba vhodné polarity a fáze hodin

pomocí CPOL a CPHA registrů. Řídící registr CR2 sloužící k povolení přerušení a práce s DMA není využíván. Po nastavení všech potřebných registrů konfigurujících práci s SPI je nakonec nastaven registr SPE pro umožnění komunikace.

Následuje přiřazení řídích a datových bitů řadiče k výstupním pinům mikrokontroleru. Na alternativních výstupech jsou připojeny a nastaveny piny komunikující s řadičem přes SPI rozhraní, a to SCK na GPIOA D13 a MOSI na GPIOB D11. Ostatní komunikační piny CS, RESET a DC jsou připojeny na běžné výstupy mikrokontroleru nevyužívající SPI.

2.3 Přenos dat a příkazů pomocí SPI

```
void OLED_Write8(uint8_t b)
{
   SPI1->DR = b;
   SPI_WAIT(SPI1);
}
void OLED_WriteCmd(uint8_t cmd)
{
   OLED_DC_LOW;
   OLED_CS_LOW;
   OLED_Write8(cmd);
   OLED_CS_HIGH;
}
void OLED_WriteData(uint8_t data)
{
      OLED_DC_HIGH;
      OLED_CS_LOW;
      OLED_Write8(data);
      OLED_CS_HIGH;
}
```

Obě funkce zajišťující přenos využívají společnou funkci OLED_Write8, která do datového registru DR rozhraní SPI1 zapíše bajt předaný v argumentu a následně čeká, než se celý bajt přenese.

Funkce OLED_WriteCmd zajišťuje přenos příkazů, kdy při zavolání obdrží v argumentu 8bitové číslo reprezentující příkaz definovaný v manuálu. Ještě před samotným zápisem jsou z důvodu přenosu příkazu a povolení komunikace nastaveny piny DC a CS do nízké logické úrovně pomocí nadefinovaného makra. Následně je přivolána funkce OLED_Write8 s převzatým argumentem. Po přenosu je pin CS opět nastaven na vysokou logickou úroveň.

Funkce přenášející data OLED_WriteData funguje velmi podobně jako funkce předchozí, s tím rozdílem, že pin DC je nastaven na vysokou logickou úroveň z důvodu přenosu dat.

2.4 Hardwarová inicializace OLED s I²C

2.4.1 Konfigurace I²C s mikrokontrolerem

```
bool InitI2C1(i2cSpeed spd)
{
    if ((spd != i2cSpeed100k) && (spd != i2cSpeed400k))
        return false;

    if (!(RCC->APB1ENR & RCC_APB1ENR_I2C1EN))
    {
        RCC->APB1ENR |= RCC_APB1ENR_I2C1EN;
        RCC->APB1RSTR |= RCC_APB1RSTR_I2C1RST;
        RCC->APB1RSTR &= ~RCC_APB1RSTR_I2C1RST;
    }
    Nucleo_SetPinGPIO(GPIOB, 8, ioPortAlternatePP);
    Nucleo_SetPinAFGPIO(GPIOB, 8, 4);
    Nucleo_SetPinGPIO(GPIOB, 9, ioPortAlternateOC);
    Nucleo_SetPinAFGPIO(GPIOB, 9, 4);
    I2C_Reset();
    I2C1->CR1 = I2C_CR1_PE;
    I2C1->CR2 = 0;
```

Funkce InitI2C1 inicializuje veškerou komunikaci s řadičem pomocí I²C rozhraní. Zprvu je po přivolání ověřena požadovaná rychlost předaná v argumentu. Pokud není rychlost 100kb nebo 400kb za sekundu, program automaticky vyhodí chybu. Pokud byla jedna z těchto rychlostí zadána, přikračuje se k přiřazování jednotlivých komponent rozhraní k mikrokontroleru. Nejprve je proveden reset, stejně jako v případě použití rozhraní SPI, akorát I²C je připojeno na datovou sběrnici APB1 a resetovací registr se odlišuje pojmenováním jako I2C1. Dále jsou přiřazeny řídící piny SCL, který je fyzicky připojen na pin D15 GPIOB reprezentovaný ve vnitřní struktuře jako PB8 a SDA připojen na pin D14 GPIOB reprezentovaný jako PB9 pomocí funkcí pro přiřazení Nucleo_SetPinGPIO a Nucleo_SetPinAFGPIO. Po přiřazení následuje volání funkce pro reset periferních signálů a zápis do řídícího registru CR1 pro povolení periferní komunikace a CR2 pro vyčištění všech konfiguračních bitů.

2.4.2 Nastavení hodinového signálu

```
RCC->CFGR = (RCC->CFGR & ~RCC_CFGR_PPRE1) |
RCC_CFGR_PPRE1_DIV1;
int apbClk = SystemCoreClock;
int apbClkMhz = apbClk / 1000000;
I2C1->CR2 = apbClkMhz;
I2C1->CR1 = 0;
I2C1->TRISE = apbClkMhz + 1;
I2C1->CCR = I2C_SPEED(apbClk, spd, 0);
I2C1->CR1 |= I2C_CR1_ACK;
I2C1->CR1 |= I2C_CR1_PE;
```

Součástí inicializační funkce InitI2C1 je i nastavení hodinového signálu. Na počátku je nastaveno dělení frekvence hodinového signálu jednou, takže bude počítáno s původní frekvencí danou zabudovaným oscilátorem. Tato hodnota se zapíše do pomocné proměnné pro budoucí výpočty. Následuje přepočet této hodnoty na jednotky MHz a zápis do řídícího registru CR2. Registr CR1 je vynulován, čímž se vytvoří podmínka pro práci s registrem TRISE, který slouží k zajištění stability frekvence a musí být naprogramován minimálně na hodnotu SCL zvýšenou o jedna. Následuje nastavení registru CCR pro řízení hodinového signálu, jež obstarává výpočetní funkce s danými předanými argumenty. Nakonec je do řídícího registru CR1 zapsán potvrzovací bit a je povolena práce s periferií.

2.5 Reset a čtení stavu I²C

```
static void I2C_Reset(void)
{
  uint16_t tout;
  I2C1->CR1 |= I2C_CR1_SWRST;
  for (tout = 100; tout; tout--)
    __nop();
  I2C1->CR1 = 0;
}
```

Reset I²C rozhraní je zařízen funkcí I2C_Reset, kdy na začátku proběhne deklarace pomocné proměnné. Poté je nahozen resetovací příznak do kontrolního registru CR1 a vyvolané krátké zpoždění využívající pomocnou proměnou. Nakonec je resetovací příznak opět shozen.

```
static __inline uint16_t I2C_sr(void)
{
  uint16_t sr;
  sr  = I2C1->SR1;
  sr |= I2C1->SR2 << 16;
  return (sr);
}
```

Z důvodu nutnosti čtení stavu pro počáteční a koncovou podmínku a přenos adresy a dat je založena funkce I2C_sr. Po deklaraci proměnné na ukládání všech stavů je do ní zapsán stav celého stavového registru SR1 obsahující stavy všech požadovaných atributů krom koncové podmínky a bitově přičten stav pro koncovou podmínku nacházející se ve stavovém registru SR2 posunutý na pozici, která není využita. Po přivolání a provedení příkazů vrací hodnotu proměnné.

2.6 Přenos dat a příkazů pomocí I²C

```
bool I2C1_WriteByte(uint8_t devAdr, uint8_t cr, uint8_t val)
{
  I2C_Start();
  I2C_Addr(devAdr);
  I2C_Write(cr);
  I2C_Write(val);
  I2C_Stop();
  return true;
}
void OLED_WriteCmd(uint8_t cmd)
{
     I2C1_WriteByte(SSD1306_SLAVEADR, SSD1306_COMMANDI, cmd);
}

void OLED_WriteData(uint8_t data)
{
     I2C1_WriteByte(SSD1306_SLAVEADR, SSD1306_COMMANDI, data);
}
```

Přenos vykonává funkce I2C1_WriteByte. Jako předané argumenty slouží adresa slave zařízení, v tomto případě řadiče, dále řídící byte, který podle své hodnoty rozhoduje, zda jsou posílány data, nebo příkazy, a nakonec samotná hodnota, která má být přenesena. Celá funkce vykonává pouze volání ostatních funkcí zařizujících sekvenci přenosu. První je vygenerována počáteční podmínka, poté jsou postupně poslány adresa slave zařízení, řídící byte a data. Po odeslání všech dat je vygenerována koncová podmínka. Nakonec funkce vrací potvrzovací hodnotu, kterou lze využít pro ověření, zda byl přenos úspěšný.

2.7 Počáteční a koncová podmínka a přenos adresy a dat

```
static const uint16_t _timeoutI2C = MAX_TIMEOUT;
static bool I2C_Start(void)
{
  uint16_t w = _timeoutI2C;
  I2C1->CR1 |= I2C_CR1_START;
  while (!(I2C_sr() & I2C_SR1_SB))
  {
    if (w)
      w--;
    else
      break;
  }
  return w;
}

static bool I2C_Write(uint8_t val)
{
  uint16_t w = _timeoutI2C;
  I2C1->DR = val;
  while (!(I2C_sr() & I2C_SR1_BTF))
  {
    if (w)
      w--;
    else
      break;
  }
  return w;
}
```

Obě tyto funkce fungují prakticky stejným způsobem. Po zavolání je deklarována proměnná sloužící k odpočtu maximální doby přenosu. Následně je podle typu funkce do řídícího registru CR1 zapsán bit pro počáteční, nebo koncovou podmínku. Ve smyčce je pomocí funkce pro čtení testováno generování počáteční podmínky, nebo vynulování bitu určujícího, zda byla všechna data přenesena. Zároveň se v podmínce odpočítává maximální doba přenosu. Pokud se data nestihly přenést za nejvyšší dobu, cyklus je ukončen. Nakonec funkce vrací hodnotu tohoto času pro případnou kontrolu správnosti přenosu.

Stejným způsobem funguje i přenos adresy a dat. V těchto případech ovšem není využíván řídící registr CR1, ale datový registr DR, do kterého se zapisují data předaná v argumentu funkcí. Zbytek zůstává stejný jako pro funkce počáteční a koncové podmínky.

2.8 Nastavení kurzoru a rozsvícení/zhasnutí zobrazovače

```
void setCursor(uint8_t x, uint8_t y)
{
	OLED_WriteCmd(SSD1305_SETLOWCOLUMN + x);
	OLED_WriteCmd(SSD1305_SETHIGHCOLUMN + (x >> 4));
	OLED_WriteCmd(SSD1305_SETPAGESTART + y);
}
void ON_OFF(uint8_t onoff)
{
	for (uint8_t page = 0; page < 4; page++)
	{
		setCursor(0,page);
		for (uint8_t col = 0; col < SSD1305_OLEDWIDTH; col++)
		{
			OLED_WriteData(onoff);
		}
	}
	setCursor(0,0);
}
```

Před zobrazením obrazových dat je vždy potřeba určit kurzor definující souřadnice. Tuto problematiku řeší funkce setCursor. Při volání jsou jako dva argumenty předány číslo sloupce a stránky, kdy obě čísla musí být vždy o jedna menší, protože první sloupec i stránka jsou označeny číslem 0. Po zavolání funkce je nejprve nastavena souřadnice horního segmentu přičtením prvního zadaného argumentu k adrese určující nejnižší nibbl prvního sloupce. Následně je nastavena i souřadnice spodního segmentu přičtením zadaného argumentu k adrese určující nejvyšší nibbl prvního sloupce a bitový posunutím o čtyři pozice doprava, neboť nejvyšší nibbl je automaticky nastaven doprostřed. V posledním kroku je nastavena stránka určená k využití stejným způsobem, jako nastavení souřadnice horního segmentu, ale s použitím druhého argumentu.

Jelikož pro tento řadič neexistuje žádná funkce, která by byla schopna vypnout, nebo zapnout všechny segmenty, její úlohu přebírá funkce ON_OFF. Princip spočívá ve dvou vnořených cyklech. První slouží jako ukazatel stránky a druhý jako ukazatel sloupce. Po přivolání funkce je předán i parametr, který určuje, zda se mají všechny segmenty rozsvítit, nebo zhasnout. Při každém opakování prvního cyklu, jehož počet opakování je nastaven na počet stránek je nastaven kurzor na aktuální stránku určenou cyklem a první sloupec. Kurzor sloupce není potřeba nastavovat, jelikož pomocí adresního módu se ukazatel automaticky

inkrementuje o jedna. Dále vždy proběhne cyklus, který na každý sloupec aktuální stránky zapíše hodnotu předanou argumentem. Nakonec se vyresetuje kurzor, aby nenastal problém s dalším zobrazováním dat.

2.9 Zobrazovací funkce

```
void showData(bool pole[SSD1305_OLEDHEIGHT][SSD1305_OLEDWIDTH],int
pagesize, int height, int width)
{
	uint8_t toPut = 0;
	for (int h = 0;h<height/pagesize;h++)
	{
		setCursor(0,h);
		for (int g = 0;g<width;g++)
		{
			for (int i = 0;i<pagesize;i++)
			{
				if (pole[i+(h*pagesize)][g])
				{
					toPut |= 1 << i;
				}
			}
			OLED_WriteData(toPut);
			toPut = 0;
		}
	}
}
```

Jako funkce pro zpracování dat a zobrazení jich slouží funkce showData. Při volání této funkce jsou do argumentů předány údaje o zobrazovači jako počet segmentů na jednu stránku, jeho výška a šířka a taky dvourozměrné pole typu bool obsahující matici reprezentující jednotlivé segmenty. Před začátkem vlastního řešení je založena 8bitová proměnná pro ukládání dat získaných z pole. Následuje trojitý vnořený cyklus. První cyklus reprezentuje počet stránek, druhý řádek počet sloupců a poslední počet řádků na jedné stránce. V každém opakování prvního cyklu je nastaven kurzor vždy na první sloupec a stránku podle aktuálního cyklu. Celá metodika je implementována v posledním cyklu podmínkou, zápisem do předem deklarované proměnné a následně zobrazením zapsaných dat na zobrazovač a vynulování pomocné proměnné. V podmínce je testováno předané pole. Na první souřadnici pole je ve výpočtu k aktuálnímu řádku zahrnuta i hodnota aktuální stránky, protože cyklus dosahuje maximálně osmé pozice v poli, a tím pádem by nebylo možné přistoupit k datům za touto hranicí. Ve druhé souřadnici pole je pouze argument

ukazující na aktuální sloupec. Pokud je hodnota na takto určených souřadnicích 1, do pomocné proměnné je na konkrétní výši bitu zapsána 1 pomocí bitového posunu a bitového součtu. V opačném případě je zapsána 0.

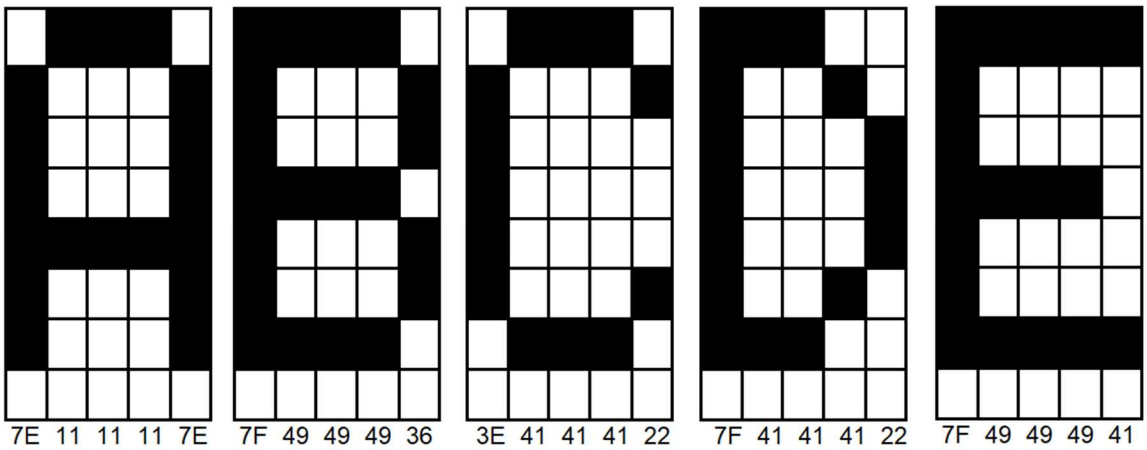
2.10 Využití zobrazovací funkce

```
for (int i = 0; i < 5; i++)
{
      for (int j = 0; j < 8; j++)
      {
            mask = mask << j;
            if ((letter[letterCount][i] & mask) == mask)
            {
                  toShow[j+(cursorPage*SSD1305_PAGESIZE)][counterCol
                  ] = 1;
            }
            else
            {
                  toShow[j+(cursorPage*SSD1305_PAGESIZE)][counterCol
                  ] = 0;
            }
            mask = 1;
      }
      counterCol++;
}
for (int i = 0; i < 8; i++)
{
      toShow[i+(cursorPage*SSD1305_PAGESIZE)][counterCol] = 0;
}
counterCol++;
if ((counterCol + 6) > SSD1305_OLEDWIDTH)
{
      counterCol = 0;
      cursorPage++;
}
if (cursorPage == SSD1305_OLEDHEIGHT/ SSD1305_PAGESIZE)
{
      cursorPage = 0;
}
letterCount++;
if (letterCount == 26)
{
      letterCount = 0;
}
showData(toShow,SSD1305_PAGESIZE,SSD1305_OLEDHEIGHT,SSD1305_OLEDWI
DTH);
```

Realizace praktického využití zobrazovací funkce je provedena v této části kódu, kdy se na zobrazovači postupně zobrazují sestupně písmena abecedy definované pomocí 8bitového dvourozměrného pole.

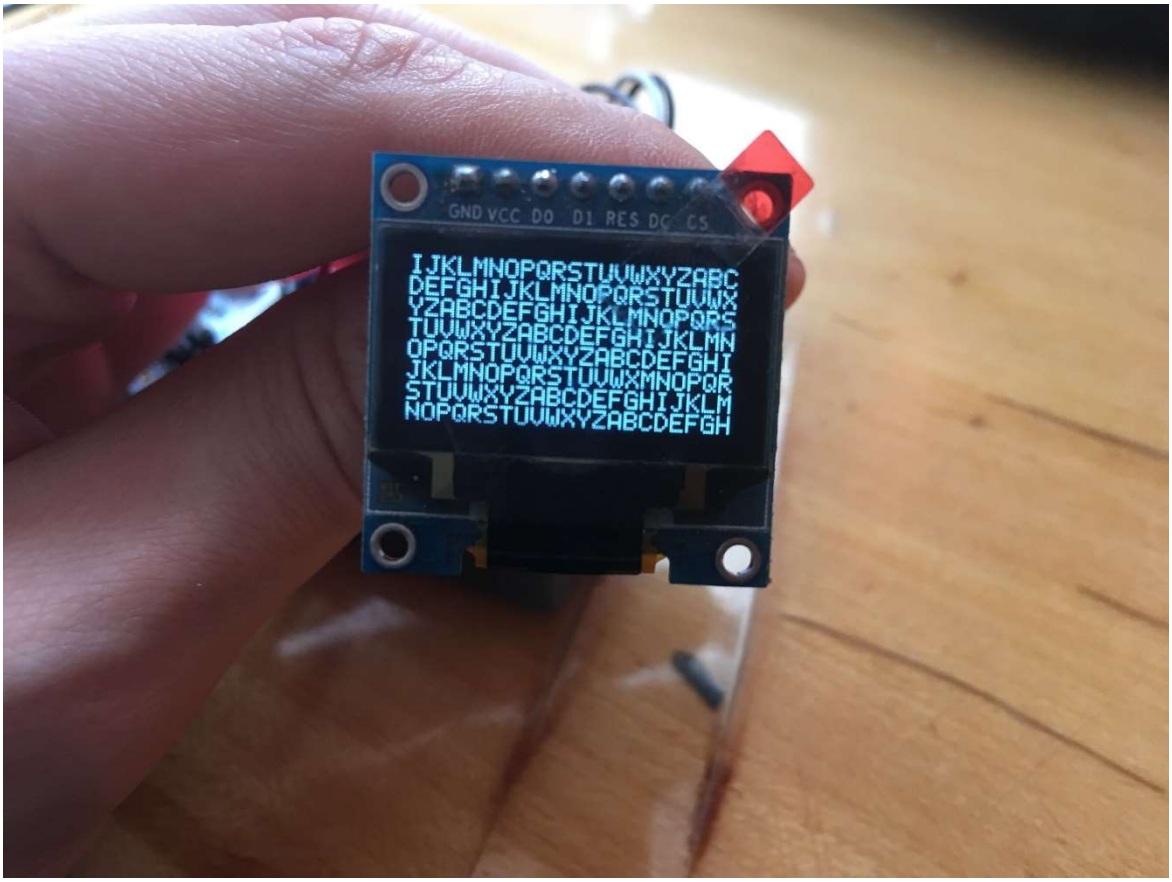
```
uint8_t letter[26][5] = {
      {0x7e, 0x11, 0x11, 0x11, 0x7e}, -> A
      {0x7f, 0x49, 0x49, 0x49, 0x36}, -> B
      {0x3e, 0x41, 0x41, 0x41, 0x22}, -> C
      {0x7f, 0x41, 0x41, 0x22, 0x1c}, -> D
      {0x7f, 0x49, 0x49, 0x49, 0x41}, -> E
      {0x7f, 0x09, 0x09, 0x09, 0x01}, -> F
      {0x3e, 0x41, 0x49, 0x49, 0x7a}, -> G
      {0x7f, 0x08, 0x08, 0x08, 0x7f}, -> H
      {0x00, 0x41, 0x7f, 0x41, 0x00}, -> I
      {0x20, 0x40, 0x41, 0x3f, 0x01}, -> J
      {0x7f, 0x08, 0x14, 0x22, 0x41}, -> K
      {0x7f, 0x40, 0x40, 0x40, 0x40}, -> L
      {0x7f, 0x02, 0x0c, 0x02, 0x7f}, -> M
      {0x7f, 0x04, 0x08, 0x10, 0x7f}, -> N
      {0x3e, 0x41, 0x41, 0x41, 0x3e}, -> O
      {0x7f, 0x09, 0x09, 0x09, 0x06}, -> P
      {0x3e, 0x41, 0x51, 0x21, 0x5e}, -> Q
      {0x7f, 0x09, 0x19, 0x29, 0x46}, -> R
      {0x46, 0x49, 0x49, 0x49, 0x31}, -> S
      {0x01, 0x01, 0x7f, 0x01, 0x01}, -> T
      {0x3f, 0x40, 0x40, 0x40, 0x3f}, -> U
      {0x1f, 0x20, 0x40, 0x20, 0x1f}, -> V
      {0x3f, 0x40, 0x38, 0x40, 0x3f}, -> W
      {0x63, 0x14, 0x08, 0x14, 0x63}, -> X
      {0x07, 0x08, 0x70, 0x08, 0x07}, -> Y
      {0x61, 0x51, 0x49, 0x45, 0x43}};-> Z
```

Každý řádek obsahuje pět hexadecimálních čísel reprezentujících zobrazení daného písmena na zobrazovači, kdy každé číslo představuje jeden sloupec.

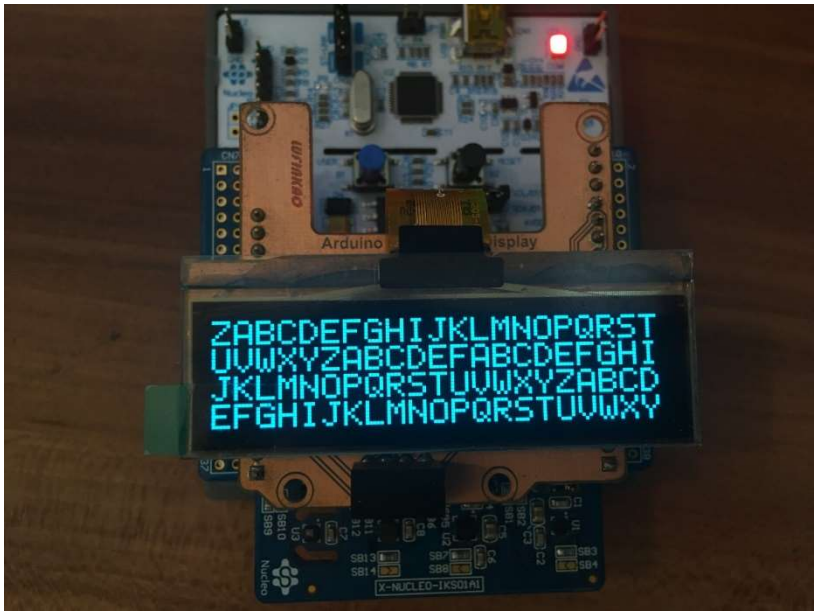


Obr. 13 Reprezentace zobrazovaných písmen

Vnořeným dvojitým cyklem je realizován přepis dat z tabulky písmen na pole reprezentující obrazová data. Ve druhém cyklu je definována maska pro zjišťování hodnot aktuálního písmene určeného k zapsání. Postupně se prochází všechny bity každého sloupce, a pokud je na dané souřadnici zapsána logická 1, přepíše se také do pole obrazových dat na konkrétní pozici. V opačném případě se zapíše logická 0. Dále je resetována maska pro použití na další bit. Po přepisu jednoho řádku se inkrementuje ukazatel sloupce a cyklus se opakuje pro další sloupec. Když se zapíše celé písmeno, je za něj zapsána i mezera, aby písmena navzájem nesplývala. Po této sekvenci je vždy proveden test na konec stránky pomocí podmínky z důvodu přesahu písmena přes stránku. Pokud je ukazatel zvýšený o šest vyšší než šířka zobrazovače, inkrementuje se ukazatel stránky a resetuje ukazatel sloupce. Dále je provedena kontrola na přetečení ukazatele stránky. Pokud je ukazatel vyšší než nejvyšší číslo stránky, je resetován na začátek. Nakonec je testováno aktuální písmeno, a pokud bylo zapsáno písmeno Z, nastaví se ukazatel znovu na písmeno A. Po provedení testů je zavolána zobrazovací funkce, která zobrazí zapsaná data.



Obr. 14 Zobrazovací funkce na panelu o rozlišení 128 x 64 S SPI



Obr. 15 Zobrazovací funkce na panelu o rozlišení 128 x 32 s SPI

Rozdíl v řízení mezi SPI a I²C komunikací:

SPI:

```
void OLED_WriteCmd(uint8_t cmd)
{
   OLED_DC_LOW;
   OLED_CS_LOW;
   OLED_Write8(cmd);
   OLED_CS_HIGH;
}
void OLED_WriteData(uint8_t data)
{
   OLED_DC_HIGH;
   OLED_CS_LOW;
   OLED_Write8(data);
   OLED_CS_HIGH;
}
```

I²C:

```
void OLED_WriteCmd(uint8_t cmd)
{
      I2C1_WriteByte(SSD1306_SLAVEADR, SSD1306_COMMANDI, cmd);
}

void OLED_WriteData(uint8_t data)
{
      I2C1_WriteByte(SSD1306_SLAVEADR, SSD1306_COMMANDI, data);
}
```

Po naprogramování obou případů použitého rozhraní je zajímavé si povšimnout, že v konečném ovládání za použití kteréhokoliv rozhraní pro stejný typ řadiče není skoro žádný rozdíl. Veškeré zobrazování a inicializace na OLED panelu využívá funkce OLED_WriteCmd a OLED_WriteData, kdy samotné ovládání zůstává stejné pro oba případy komunikačního rozhraní, a tedy jediný rozdíl nastává při inicializaci těchto funkcí.

Zhodnocení a závěr

V rozsahu této práce byla všeobecně představena technologie OLED zobrazovačů a vytvořen přehled konkrétních řadičů pro malé OLED zobrazovače, jejich porovnání a parametry. Dále byl konkrétně popsán jejich způsob řízení, zobrazování, komunikace a programování a vysvětlen princip a inicializace používaných komunikačních rozhraní včetně ilustračních časových diagramů potřebných pro jednodušší pochopení celé problematiky.

V praktické části byly implementovány předchozí informace do praktických ukázek, kde jsou popsané a vysvětlené inicializační, přenosové, zobrazovací a užitečné funkce sloužící k úspěšnému naprogramování řadiče pomocí mikrokontroleru a zobrazení požadovaných dat.

K praktické realizaci byl využit řadič SSD1305 s připojeným OLED panelem o rozlišení 128 x 32 s SPI komunikačním rozhraním a řadič SSD1306 s připojeným OLED panelem o rozlišení 128 x 64 s I^2C a SPI rozhraním. Programování bylo realizováno v programovacím jazyce C v prostředí Atollic TrueSTUDIO za použití mikrokontroleru STM32F411RE s jádrem ARM Cortex.

V prvním případě bylo naprogramování stoprocentně úspěšné a zobrazování probíhalo přesně podle původních úmyslů a totéž platí i v případě použití OLED zobrazovače s rozlišením 128 x 64 a SPI rozhraním. Bohužel v druhém případě se nepodařilo pomocí I^2C komunikace data zobrazit. Tato skutečnost nastala s nejvyšší pravděpodobností z důvodu chybného porozumění dokumentace.

Literatura a informační zdroje

[1] Ossila [online] [cit. 2022-17-04]. Dostupné z: https://www.ossila.com/pages/what-is-an-oled

[2] laskakit [online] [cit. 2022-17-04]. Dostupné z: https://www.laskakit.cz/oled-displej-modry-128x64-1-3--spi

[3] OLED-info [online] [cit. 2022-17-04]. Dostupné z: https://www.oled-info.com/pmoled-vs-amoled-whats-difference

[4] Huramobit [online] [cit. 2022-17-04]. Dostupné z: https://www.huramobil.cz/co-je-amoled-displej/blog-1071/

[5] ST [online] [cit. 2022-17-04]. Dostupné z: https://www.st.com/en/microcontrollers-microprocessors/stm32f411re.html

[6] Adafruit [online] [cit. 2022-17-04]. Dostupné z: https://cdn-shop.adafruit.com/datasheets/SSD1306.pdf

[7] Adafruit [online] [cit. 2022-17-04]. Dostupné z: https://cdn-shop.adafruit.com/datasheets/SSD1331_1.2.pdf

[8] Melexis [online] [cit. 2022-17-04]. Dostupné z: https://www.melexis.com/en/product/MLX81130

[9] GOLDELOX [online] [cit. 2022-17-04]. Dostupné z: https://4dsystems.com.au/mwdownloads/download/link/id/141/

[10] DATASHEETS [online] [cit. 2022-17-04]. Dostupné z: https://www.datasheets.com/en/part-details ssd1313-solomon-systech-57308596

[11] everythingRF [online] [cit. 2022-17-04]. Dostupné z: https://www.everythingrf.com/community/what-is-spi

[12] TIŠNOVSKÝ, Pavel, 2008. Externí sériové sběrnice SPI a I^2C. ROOT.CZ [online] [cit. 2022-17-04]. Dostupné z: https://www.root.cz/clanky/externi-seriove-sbernice-spi-a-i2c/

[13] ROOT.CZ [online] [cit.2022-17-04]. Dostupné z: https://www.root.cz/clanky/komunikace-po-seriove-sbernici-isup2supc/

[14] evision SYSTEMS [online] [cit. 2022-17-04] Dostupné z: ROOT.CZ [online] [cit. 2022-17-04]. Dostupné z: https://www.root.cz/clanky/komunikace-po-seriove-sbernici-isup2supc/

Přílohy

Příloha A

Přiložený ZIP soubor obsahuje složky s veškerými soubory kódů vytvořených v rámci této práce.

OLEDI2C:

- main.c
- nucleo_i2c.c
- nucleo_i2c.h
- OLED.c
- OLED.h
- stm_core.c
- stm_core.h

OLEDSPI:

- main.c
- OLED.c
- OLED.h
- SPI.c
- SPI.h
- stm_core.c
- stm_core.h