Střední průmyslová škola a Vyšší odborná škola, Liberec, příspěvková organizace

Herní engine pro PlayStation 1

Maturitní práce

Autor **Ráček Jan (2020)**

Obor **Informační technologie**

Vedoucí práce **Ing. Marek Pospíchal**

Školní rok **2023/2024**

Počet stran **27**

Počet slov **2728**



Anotace

Tato práce si klade za cíl zlepšit život programátorů PlayStationu 1 tím, že jim poskytne prostředky pro jednoduchý vstup do programátorské komunity PSX. Konečným výstupem této práce by měl být engine, který abstrahuje co nejvíce hardwarových specifik, aniž by vytvářel příliš velkou hardwarovou zátěž.

Summary

This work is set out to improve the life of PlayStation 1 programmers by giving them means of simple entry into the PSX programming community. The final output of this thesis should be an engine which abstracts as much of the hardware specifics as possible without creating too much hardware overhead.

Čestné prohlášení

Prohlašuji, že jsem předkládanou maturitní práci vypracoval sám a uvedl jsem veškerou použitou literaturu a bibliografické citace.

V Liberci dne 08.03.2024

Ráček Jan (2020)

Obsah

[Úvod 1](#_Toc160727498)

[1 Historie 2](#_Toc160727499)

[1.1 Zformování SCEI a vydání konzole 2](#_Toc160727500)

[1.2 Net Yaroze 3](#_Toc160727501)

[1.3 DRM 3](#_Toc160727502)

[1.3.1 Region lockout 3](#_Toc160727503)

[1.3.2 Implementace DRM 4](#_Toc160727504)

[2 Hardware 6](#_Toc160727505)

[2.1 Procesor 6](#_Toc160727506)

[2.2 Paměť 6](#_Toc160727507)

[2.3 GPU 6](#_Toc160727508)

[2.4 SPU 6](#_Toc160727509)

[2.5 GTE 7](#_Toc160727510)

[2.6 I/O 7](#_Toc160727511)

[3 Vývoj enginu 8](#_Toc160727512)

[3.1 SIO 8](#_Toc160727513)

[3.2 Rendering 9](#_Toc160727514)

[3.2.1 DMA chain 9](#_Toc160727515)

[3.2.2 Příkazy grafické karty 10](#_Toc160727516)

[3.3 Audio 13](#_Toc160727517)

[3.3.1 Přehrávání ADPCM sample 13](#_Toc160727518)

[3.3.2 CCDA playback 15](#_Toc160727519)

[3.4 Ovladače 16](#_Toc160727520)

[Závěr 18](#_Toc160727521)

[Seznam zkratek a odborných výrazů 19](#_Toc160727522)

[Seznam obrázků 20](#_Toc160727523)

[Použité zdroje 21](#_Toc160727524)

[A. Seznam přiložených souborů I](#_Toc160727525)

Úvod

V dnešní době je velmi složité vzít do ruky PlayStation 1 a bez cizí pomoci prozkoumat jeho hardware a naprogramovat pro něj vlastní software. Hlavními důvody jsou Hardwarová specifika. Nedostatečná dokumentace hardwaru a nedostatek návodů.

Cílem práce je vytvořit herní engine, který dovolí jakémukoliv programátorovi, který má základní chápání programovacího jazyka C++ a jeho objektů, naprogramovat pro tuto konzoli jednoduchou hru.

# Historie

PlayStation 1 poprvé vyšel v Japonsku 3. prosince 1994 (1) a stal se jednou z prvních konzolí, které úspěšně použily CD-ROM k distribuci jejich her. Jeho samotný vznik je zajímavý příběh, který začíná u společnosti Nintendo, pro kterou mělo Sony vytvořit CD-ROM verzi jejich SNES. Nintendu však nepovažovalo dohodu se Sony za přijatelnou, jelikož předávala autorské vlastnictví veškerého materiálu distribuovaného na CD-ROM Sony. Proto se Nintendo rozhodlo přejít na poslední chvíli ke konkurenčnímu Philipsu. (2)



Obrázek Původní PlayStation ve spolupráci s Nintendem

Sony nyní mělo velké sumy peněz investované do vývoje herní konzole s CD-ROM a jelikož nechtělo přenechat trh Nintendu, tak se i přes intenzivní odpor ve vedení firmy rozhodlo zahájit vývoj na své vlastní herní konzoli.

## Zformování SCEI a vydání konzole

Pouhý rok po definitivním ukončením spolupráce s Nintendem bylo zformováno Sony Computer Entertainment Inc. Bylo vytvořeno ve spolupráci se Sony Music, díky které mohlo SCEI vytvářet CD-ROM ve velkém. O rok později byl PlayStation oznámen a vývojáři se k němu hrnuli hlavně díky levné ceně produkce médií (CD-ROM) 3D vlastnostem konzole a nižší ceně konzole jako takové oproti konkurenci.

## Net Yaroze

V roce 1997 Sony vydalo Net Yaroze. Jednalo se o speciální verzi konzole, která byla mířená na koncové uživatele, kteří si chtěli naprogramovat vlastní hru. Byla naceněna mnohem výše, než základní verze a obsahovala navíc speciální sériový kabel, který umožňoval přímé nahrání programů do paměti konzole bez použití CD-ROM.

Konzole jako taková se hardwarově nelišila od základní verze jinak než černou barvou plastů. Měla však odemčený software, který neobsahoval žádné DRM ani Region-lockout (vlastnosti, které dále v práci bude potřeba obejít)

Dnes je již tato verze vysoce naceněnou raritou a k její sehnání člověk potřebuje trpělivost a velký peněžní obnos. Naštěstí ji ale nepotřebujeme.



Obrázek Net Yaroze s obsahem balení (3)

## DRM

Již v počátcích konzole Sony bojovalo s hackery, kteří se snažili na konzolích spouštět vypálené hry. K pochopení toho, jak se jim povedlo nakonec konzoli prolomit je ale potřeba znát, jak DRM v PlayStationu funguje.

### Region lockout

V konzolích minulých generací bývalo standardem zavádět uzamknutí konzolí na určité regiony. Hry a konzole pak musely být pořízeny ve stejném regionu, jinak je konzole odmítaly spustit. Důvody byly hardwarové, ale i ekonomické. NTSC vyžaduje jinou obnovovací frekvenci snímků, barevné kódování a reprezentaci barev, než PAL a hardware konzolí musel být pro toto pozměněn. Ekonomicky se Sony vyplatilo rozdělit si trh na 3 velké části, ve kterých mohlo různě naceňovat produkty a upravovat nabídku izolovaně od ostatních regionů. Regiony byly následující:

* NTSC-J – Japonsko
* NTSC-U/C – USA/Kanada
* PAL – Zbytek světa

### Implementace DRM

Sony chytře zabudovalo své DRM přímo do zdroje dat. CD-ROM měl na sobě speciální vlnitou stopu, kterou standartní CD mechaniky neuměly přečíst ani vypálit. Vlnění těchto stop mělo specifickou frekvenci, která určovala region disku.



Obrázek Vizualizace vlnité stopy určující region hry (4)

CD mechanika následně přímo komunikovala s BIOSem konzole, který odmítl hru spustit, pokud se region disku neshodoval s regionem konzole. Jelikož nebylo a stále není možné takový disk vypálit v domácím prostředí, tak museli hackeři objevit způsob, jak toto zabezpečení obejít.

#### Metoda prohazování disků

Sony se nepodařilo systém kontroly disků implementovat dostatečně dobře na to, aby se nenašla alespoň jedna triviální metoda jeho obejití. Jedná se o Metodu prohazování disků. Pokud uživatel bystře poslouchal zvuky, které mechanika vydává při startu konzole, tak mohl zjistit, kdy došlo k přečtení vlnité stopy a následně rychle prohodit originální disk za svůj vypálený. To dovolilo uživateli načíst jakoukoliv vypálenou hru a stačilo mu k tomu vlastnit pouze jednu libovolnou originální hru správného regionu. Tato metoda ale má své problémy, jelikož dochází k poškození mechaniky a disků při opětovném praktikování.

A person's hands on a cd player

Description automatically generated

Obrázek Uživatel praktikující metodu prohazování disků (4)

#### Modchipy

Mnohem chytřejší, a ne o moc mladší metodou jsou takzvané „modchipy“ jedná se o malé integrované obvody, které byly napájeny na základní desku konzole a odesílaly BIOSu správné údaje o regionu disku nezávisle na tom, co odeslala CD mechanika. Konzole tedy nastartovala jakýkoliv CD-R/CD-RW disk, který si uživatel vypálil bez ohledu na Region lockout.

Právě modchip je metoda, která bude v této práci využita pro spouštění kódu na reálném hardwaru.

# Hardware

Hardware PlayStationu je poněkud specifický tím, že se liší od počítačů tehdejší doby. Ať už se jedná o architekturu procesoru, nebo o způsob, kterým procesor komunikuje se svými periferiemi, tak tato konzole vystupuje ze zajetých kolejí, kde to jen jde.

## Procesor

Jako procesor máme k dispozici vylepšenou verzi 32bitového procesoru R3000A architektury MIPS s frekvencí 33.8688 MHz, 4KB instrukční cache a 1KB datové cache, která není automaticky obsluhovaná a programátor do ní musí manuálně zapisovat, chce-li ji použít. Procesor je značně omezený faktem, že nemá jednotku pro výpočty s plovoucí řádovou čárkou. Veškeré floating-point operace tedy musí být vyřešeny softwarově a celý návrh konzole pracuje s celými čísly.

## Paměť

Co se paměti týče, tak máme k dispozici 2 MB EDO DRAM paměti. Toto je velmi málo, a proto musíme načítat z CD-ROM pouze to, co opravdu potřebujeme. Přítomno je také 512 KB ROM, kde je uložený BIOS této konzole.

## GPU

Srdcem celé konzole je její grafická karta. Jedná se 32bitovou GPU vyvinutou společností Toshiba. Jako jedna z prvních konzolí zvládá nejen 2D, ale také 3D grafiku. K dispozici má 2MB video paměti, která je strukturovaná jako 2D pole 16bitových pixelů o velikosti 1024 x 256 pixelů. Do tohoto se musí vejít jak oba framebuffery, tak veškeré textury, které chceme při renderingu používat. Dalším problémem je fakt, že se jedná o 2D pole, kvůli čemuž musíme rozložení textur a framebufferů do paměti zpracovávat manuálně a musíme provádět tzv. „texture stacking“, abychom maximalizovali využití dostupného prostoru.

## SPU

Zvuk PlayStationu je zajištěn za pomoci zvukové jednotky, která umí přehrávat ADPCM data ze své vlastní 512 KB RAM na 24 kanálech. Obsahuje také reverb jednotku, která umí daný efekt dodat digitálně za běhu. Lze tak simulovat vchod do velké, prázdné místnosti apod.

## GTE

Jakékoliv 3D vykreslování se neobejde bez lineární algebry, která je pro běžné procesory velmi náročná, jelikož nejsou schopné dosáhnout té úrovně paralelizace, která je pro tyto výpočetní úkony třeba. Kvůli tomu má PlayStation zabudovaný „Geometry translation engine“ jedná se o koprocesor, který umí provádět maticové operace s daty na vstupních registrech a zapsat výsledek na výstupní registry s mnohem vyšší rychlostí, než kdybychom tyto výpočty řešili softwarově na procesoru.

## I/O

Hlavní vstupní periférií PlayStationu je jeho CD-ROM mechanika. Z té načítá veškerá data hry, kterou si chceme zahrát. Zvládá čtení rychlostí až 2x.

Snad druhou nejdůležitější periférií jsou ovladače. Ty se nachází na sériové lince SIO0 spolu s paměťovými kartami a mají vyvedené konektory ven vpředu konzole.

Druhou sériovou linkou, kterou konzole má je SIO1, která plní mnoho účelů. Při vývoji hry do ní programátor může posílat debug data, která pak může z počítače číst, nebo do ní zapisovat a tím nahrávat programy přímo do paměti konzole. Dále slouží k funkci multiplayerových her. Sony totiž prodávalo kabel, kterým bylo možno propojit dvě konzole a tím umožnit jejich komunikaci pro účely Multiplayeru.

# Vývoj enginu

Samotný vývoj je velmi složitý. Už z důvodu toho, že nemáme k dispozici žádnou standartní knihovnu, ani operační systém, který by se nám staral o paměť na haldě. Jediné, co máme k dispozici jsou registry konzole, do kterých můžeme přímo zapisovat.

## SIO

Prvním krokem pro práci na jakémkoliv embeded hardwaru je mít způsob, jak debugovat a vědět, co se na daném procesoru děje. Proto máme na PlayStationu k dispozici sériovou linku SIO1, která má vývod vzadu na konzoli. Pro tyto účely máme k dispozici registry



Tento kód nám definuje funkce *sio\_init()* a *printCharacter()*. Užívá při tom registry

* **SIO\_CTRL** (ovládací registr sériové komunikace, ve kterém nastavujeme vlastnosti registru, jako RTS)
* **SIO\_MODE** (nastavení módu sériové komunikace)
* **SIO\_BAUD** (nastavení baudrate, který je zde závislý na frekvenci procesoru)
* **SIO\_STAT** (jedná se o bitové pole, které nám nahlašuje stav sériového portu. Např. zda jsou ve frontě ještě nějaká data k odeslání)

Nejvyšší praktická rychlost, kterou lze využít je baudrate 115200. Po inicializaci sériové komunikaci si následně můžeme definovat nějakou takovouto funkci:



Pro pohodlnost jsem však do enginu přidal implementaci *printf()*, abychom mohli posílat formátované řetězce.

Tato sériová komunikace lze následně také použít k debuggingu. Existuje jistý software pro PlayStation jménem UNIROM, díky kterému můžeme připojit GDB k reálnému hardwaru a debutovat tak přímo z vývojového prostředí.

## Rendering

Na PlayStationu existují dva způsoby, jak komunikovat s grafickou kartou. Prvním je přímo zapisovat příkazy do jejího příkazového registru. To je však časově náročné, a hlavně to blokuje procesor.



### DMA chain

Alternativním způsobem odesílání příkazů nejen grafické kartě jsou tzv DMA (Direct memory access) chainy. Můžeme v paměti nadefinovat buffer, do kterého budeme zapisovat jednotlivé příkazy. Ukazatel na celý buffer pak můžeme předat grafické kartě, která z něho začne vykonávat příkazy a nebude blokovat procesor, který může začít připravovat další snímek.



Následně lze odeslat příkaz grafické kartě, aby začala vykonávat příkazy z tohoto bufferu.



### Příkazy grafické karty

Grafická karta má mnoho různých příkazů a vlastností. Následuje shrnutí těch nejdůležitějších.

* **gp0\_texpage** – určuje, která část VRAM obsahuje texturu, kterou má grafická karta využít.
* **gp0\_xy** – určuje offset v obrazovkovém prostoru pro vykreslení
* **gp0\_uv** – určuje UV souřadnice pro mapování textur. Hodnota je relativní v rámci texturové stránky
* **gp0\_triangle** – vykreslítříbodový polygon
* **gp0\_quad** – vykreslí čtyřbodový polygon
* **gp0\_rectangle** – vykreslí 2D obdélník

Jak tedy postavíme příkaz pro vykreslení jednoduchého čtverce? Musíme odeslat několik wordů (v případě PlayStationu 32bitové číslo), které představují bitové pole s nastavením vykreslení.

|  |  |  |
| --- | --- | --- |
| **Číslo bitu** | **Hodnota** | **Význam** |
| 31-29 | 011 | Příkaz k vykreslení čtverce |
| 28-27 | sss | Velikost čtverce |
| 26 | 1/0 | Texturovaný? |
| 25 | 1/0 | Průhledný? |
| 24 | 1/0 | Shaded? |
| 23-0 | rgb | První barva |

Velikost čtverce v tomto případě představuje režim, ve kterém grafická karta tento čtverec vykresluje. Může nabývat následujících hodnot:

|  |  |
| --- | --- |
| **Bity** | **Význam** |
| 00 | Vlastní velikost |
| 01 | Jeden pixel (1x1) |
| 10 | 8x8 pixelů |
| 11 | 16x16 pixelů |

Tím jsme pokryli první word, k vykreslení čtverce potřebujeme ale minimálně další dva. Celý příkaz k vykreslení pak vypadá nějak takto:

|  |  |  |
| --- | --- | --- |
| **Název wordu** | **Hodnota** | **Vysvětlení** |
| Barva | ccBBGGRR | Příkaz + barva (ignoruje se, pokud je čtverec texturovaný) |
| Vertex1 | YYYXXX | Indikuje levý horní roh, kam se čtverec vykresluje. |
| UV | ClutVVUU | Pouze, pokud je čtverec texturovaný. |
| Šířka+Výška | YsizXsiz | Pouze pokud jsou bity 28-27 příkazu nastaveny na 00, tak je třeba definovat vlastní velikost |

Kdybychom měli zapsat celou posloupnost příkazů pro vykreslení čtverce 128x128 modré barvy na pozici X:10, Y,20 ve VRAM v binární reprezentaci, vypadala by takto:

// První word

11 -> Definujeme, že chceme čtverec

00 -> Chceme si velikost nastavit sami

0 -> Náš čtverec není texturovaný

0 -> Náš čtverec není průhledný

0 -> Náš čtverec není shaded

11111111 -> Hodnota modré (255)

00000000 -> Hodnota zelené (0)

00000000 -> Hodnota červené (0)

// Druhý word

0000000000010100 -> Pozice na ose Y (20)

0000000000001010 -> Pozice na ose X (10)

// Třetí word

0000000010000000 -> Velikost Y (128)

0000000010000000 -> Velikost X (128)

Když celý tento příkaz odešleme grafické kartě, tak její výstup bude vypadat takto:

A blue square on a gray background

Description automatically generated

Obrázek Finální vykreslení našeho příkazu

## Audio

Jak bylo již zmíněno, PlayStation má SPU (Sound processing unit). Tato jednotka umí přehrávat stereo zvuky následujících typů:

* ADPCM sample
* CDXA stream
* CDDA playback
* ADSR

V této práci budou využity pouze dva z výše uvedených druhů zvuku. Čímž jsou ADPCM sample a CDDA playback.

### Přehrávání ADPCM sample

ADPCM sample jsou digitální waveform data, která SPU umí přečíst a přehrát. Po správné inicializaci SPU můžeme začít nahrávat data do její SPURAM, která funguje k ukládání ADPCM dat.



Po nahrání našeho samplu do SPURAM jsme připraveni ho přehrát. K přehrání musíme najít volný kanál mezi 24mi kanály, které má SPU k dispozici.



Po nalezení takového kanálu, pak můžeme náš ADPCM sample přehrát:



Za předpokladu, že náš sample má na konci nastavený loop-flag, tak ho SPU bude opakovat, dokud ho manuálně nepřerušíme.

### CCDA playback

CD mechanika PlayStationu je ve spolupráci s SPU schopná přehrávat běžné audio stopy z CD, jako normální CD přehrávač. Většinou se tohoto dosahuje tak, že se první stopa disku vytvoří jako datová, kam vývojáři hry zapíšou binární data jejich hry (kód, textury apod.) Veškeré další stopy jsou pak volné k zápisu běžných audio stop. Když tedy konzole načte vše, co potřebuje do RAM, tak potom mechanika nic nedělá, což nám umožňuje přímo přehrávat audio-stopu z CD mechaniky, aniž bychom tyto audio data museli nahrávat do paměti PlayStationu. Tento přístup má dvě velké výhody.

1. Je naprosto neblokující. Procesor toho přehrávání zahájí a pak se o něj již nemusí starat
2. Pokud finální CD vložíme do běžného CD přehrávače, tak máme jeden disk, na kterém je jak hra samotná, tak její OST, protože moderní CD přehrávače jsou schopny automaticky přeskočit datové stopy a přehrávat jen audio stopy.

K inicializaci CDDA potřebujeme následující:



Následně můžeme přehrát jakoukoliv stopu následujícím příkazem:



Tato interface funkce pošle příkaz na kontrolní registr CD mechaniky, kterým zahájí přehrávání. Je důležité si povšimnout makro funkce *itob()*. Tato funkce konvertuje číslo na číslo v kódu BCD. CD mechanika očekává, že ji číslo stopy dodáme v BCD.

## Ovladače

Jak bylo již zmíněno PlayStation má 2 sériové rozhraní. *SIO0* a *SIO1*. Funkce *SIO1* byla popsána v kapitole *3.1*. V této kapitole si popíšeme funkci rozhraní *SIO0*. Toto rozhraní je sdílené mezi všemi konektory, které nalezneme na přední straně konzole. To zahrnuje oba konektory pro ovladače a oba konektory pro paměťové karty. Implementace komunikace s paměťovou kartou bude popsána v další kapitole. Nyní se soustředíme na ovladače.

Nejprve musíme sériové rozhraní inicializovat.



Nyní potřebujeme funkci, která bude přijímat informace od ovladačů, která se bude volat každý snímek. Zde je potřeba zaimplementovat ukládání předchozího stavu, abychom mohli mít v API funkce jako GetButtonDown, která vývojáři vrátí true pouze jeden snímek, kdy bylo tlačítko stlačeno, a ne po celou dobu jeho stlačení.



Následně můžeme vytvořit API, které programátorovi zpřístupní stav ovladačů. To ale bude popsáno v pozdější kapitole.

## CD-ROM

Načítání dat z CD-ROM je poněkud složité, protože nemáme k dispozici žádný driver a musíme si vystačit s dokumentací a ovládacími registry CD-ROM. Kvůli délce ovládacího kódu si ho sem nemůžeme vypsat celý. Místo toho si popíšeme, jak s CD-ROM komunikujeme.

Jelikož diskové operace jsou velmi zdlouhavé, tak si nemůžeme dovolit blokovat procesor, dokud se nedokončí. Proto má CD-ROM vlastní kanál přerušení, jelikož čtení dat z disku může trvat klidně i desítky vteřin.

### Přečtení souboru z CD-ROM

Pokud chceme přečíst soubor z disku, musíme provést následující:

1. Musíme přečíst ISO9660 pathtable
   * Každý disk podle standardu ISO9660 obsahuje hned několik verzí této tabulky. Řekne nám, jaké jsou v disku složky a jaké jsou pozice souborů, abychom mohli najít sektory, ve kterých se nacházejí naše soubory.
2. Pošleme CD-ROM SEEK příkaz, který přemístí hlavu na stopu, na které se nachází začátek našeho souboru.
3. Použijeme příkaz CD-ROM READ, kterému poskytneme počet sektorů, které chceme přečíst, a ukazatel do paměti, kam se mají přečtená data zapsat. Jakmile se čtení dokončí, dostaneme přerušení a data jsou připravena k užití.

### Fudgebundle

Jedním nástrojem, který se při vývoji tohoto enginu velmi osvědčil je typ souborů Fudgebundle, který vyvinul můj dobrý italský kamarád spicyjpeg. Jedná se o specifikaci binárního souboru, který ukládá herní data do jednoho velkého souboru, který má v hlavičce hashovou tabulku, díky které můžeme hledat assety v rámci tohoto souboru za pomoci jejich názvů. Jednou velkou výhodou Fudgebundlu jsou jeho nástroje, které k němu spicyjpeg napsal, které automaticky zarovnávají textury do dvojrozměrné paměti namísto toho, aby si to musel programátor dělat sám. Výsledný soubor je jeden velký archiv, který načítáme najednou z CD-ROM do paměti, což značně urychluje načítání, protože nemusíme posílat individuální SEEK příkazy pro každý asset.

Závěr

Tak jsem se dostal až na konec.

Seznam zkratek a odborných výrazů

HTML

HyperText Markup Language – značkovací jazyk používaný pro tvorbu webových stránek.

Seznam obrázků

[Obrázek 1 Původní PlayStation ve spolupráci s Nintendem 2](#_Toc160648296)

[Obrázek 2 Net Yaroze s obsahem balení (3) 3](#_Toc160648297)

[Obrázek 3 Vizualizace vlnité stopy určující region hry (4) 4](#_Toc160648298)

[Obrázek 4 Uživatel praktikující metodu prohazování disků (4) 5](#_Toc160648299)

[Obrázek 5 Finální vykreslení našeho příkazu 12](#_Toc160648300)

Použité zdroje

1. **Sony Computer Entertainment.** Business Data. *Sony Computer Entertainment.* [Online] Sony Computer Entertainment, 2004. https://web.archive.org/web/20040422074823/http://www.scei.co.jp/corporate/data/bizdatajpn\_e.html.

2. **EDGE.** The Making Of: PlayStation. *EDGE.* [Online] 24. Duben 2009. https://web.archive.org/web/20140206193956/http://www.edge-online.com/features/making-playstation/.

3. **Sebo, Roberto.** Sony Net Yaroze with software development kit. *Flickr.* [Online] https://www.flickr.com/photos/scedevnet/5585974002/.

4. **ModernVintageGamer.** How the Sony PlayStation PS1 Security was defeated | MVG. *YouTube.* [Online] 9. Březen 2020. https://www.youtube.com/watch?v=7HOBQ7HifLE.

5. **Lameguy64.** PSn00bSDK. *Github.com.* [Online] 2023. https://github.com/Lameguy64/PSn00bSDK.

1. Seznam přiložených souborů

Na přiloženém datovém nosiči se nacházejí následující soubory a složky:

* **MP2010-Novák-Jan-L4-Tepelné\_čerpadlo.docx** – editovatelná verze dokumentace maturitní práce
* **MP2010-Novák-Jan-L4-Tepelné\_čerpadlo.pdf** – tisknutelná verze dokumentace maturitní práce
* **Výkresy** – kompletní výkresová dokumentace
* **Aplikace** – zdrojové kódy