STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 10: Elektrotechnika, elektronika a telekomunikace

Procesor vytvořený z logických hradel

Ondřej Urbánek
Olomoucký kraj

Olomouc 2025

STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

Obor č. 10: Elektrotechnika, elektronika a telekomunikace

Procesor vytvořený z logických hradel

A processor made of logic gates

Autoři: Ondřej Urbánek
Škola: VOŠ a SPŠE Olomouc, Božetěchova 3, 779 00 Olomouc
Kraj: Olomoucký kraj
Konzultant: Ing. Zuzana Veselá

Město a vročení

Prohlášení

Prohlašuji, že jsem svou práci SOČ vypracoval/a samostatně a použil/a jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Olomouci dne 11.4.2025……
Ondřej Urbánek

Poděkování

Chtěl bych poděkovat Ing. Zuzaně veselé za poskytnuté informace ohledně SOČ a pomoc při tvorbě dokumentace. Vždy jsem věděl na koho se mohu obrátit.

Anotace

Ve své práci jsem se zabýval především vývojem vlastní architektury procesoru, následně i jeho návrhem, konstrukcí a oživením. Dalšími základními body práce byly: konstrukce celého systému pouze pomocí logických hradel, modulární provedení systému a vizualizace toku dat mezi jednotlivými moduly. Celý procesor je programovatelný pomocí panelu, který je součástí přední desky a obsahuje i externí vstupy a výstupy pro ovládání reálných prvků. Je schopen vykonávat všechny základní operace jako jiný běžný procesor, například logické i aritmetické výpočty, práce s daty, včetně ukládání a načítání z pamětí, podmíněné i nepodmíněné skoky v programu, generace pseudonáhodných čísel a další.

Klíčová slova

Procesor; logická hradla; programovatelný; sběrnice; výpočetní technika

Annotation

In my work, I focused primarily on the development of my own processor architecture, followed by its design, construction, and activation. Other fundamental points of the work were the construction of the entire system using solely logic gates, modular design of the system, and the visualization of the data flow between individual modules. The entire processor is programmable using a panel that is part of the front board and also includes external inputs and outputs for controlling real-world elements. It is capable of performing all basic operations like any other common processor, for example, logical and arithmetic calculations, data manipulation including storing and loading from memory, conditional and unconditional jumps in the program, pseudo-random number generation, and other.

Keywords

Processor; logic gates; programmable; system bus; computing

OBSAH

Úvod

Cílem projektu bude vytvoření procesoru vlastní architektury z jednoduchých integrovaných obvodů, obsahujících pouze logická hradla. Dalšími použitými součástkami budou například rezistory, kondenzátory, led diody, aj.

Základním bodem projektu bude návrh vlastní architektury s použitím co možná nejmenšího množství vnějších zdrojů. Dále se projekt bude zaměřovat na logický návrh jednotlivých desek plošných spojů, nákres jejich zapojení, sestavení a následné oživení a otestování celého systému. Při úspěšném provedení bude procesor schopen několika základních operací. Ty budou mimo jiné zahrnovat logické a aritmetické výpočty, přesun dat mezi pamětí a jednotlivými bloky a postupné vykonávání instrukcí s možností podmíněných i nepodmíněných skoků na specifický řádek programu. Celý systém bude pracovat v osmibitové hloubce.

Jednotlivé části systému budou rozděleny na samostatné moduly. Ty budou propojeny sběrnicemi a díky adresaci budou jejich pozice zaměnitelné. Díky této možnosti bude celý systém zcela modulární. Průhledná konstrukce zajistí přímou viditelnost všech částí systému. Projekt tak bude využitelný jako učební pomůcka.

Dokumentace projektu bude zpracována do několika kapitol podle bodů zadání. První z bodů bude rozdělen dle jednotlivých modulů.

Projekt bude založený na mých předešlých úspěšných pokusech o sestavení různých procesorů a dalších výpočetních zařízení v neobyčejném virtuálním prostředí - ve hře Minecraft. Může tedy poukazovat na fakt, že moderní způsoby edukace zahrnují i herní průmysl.

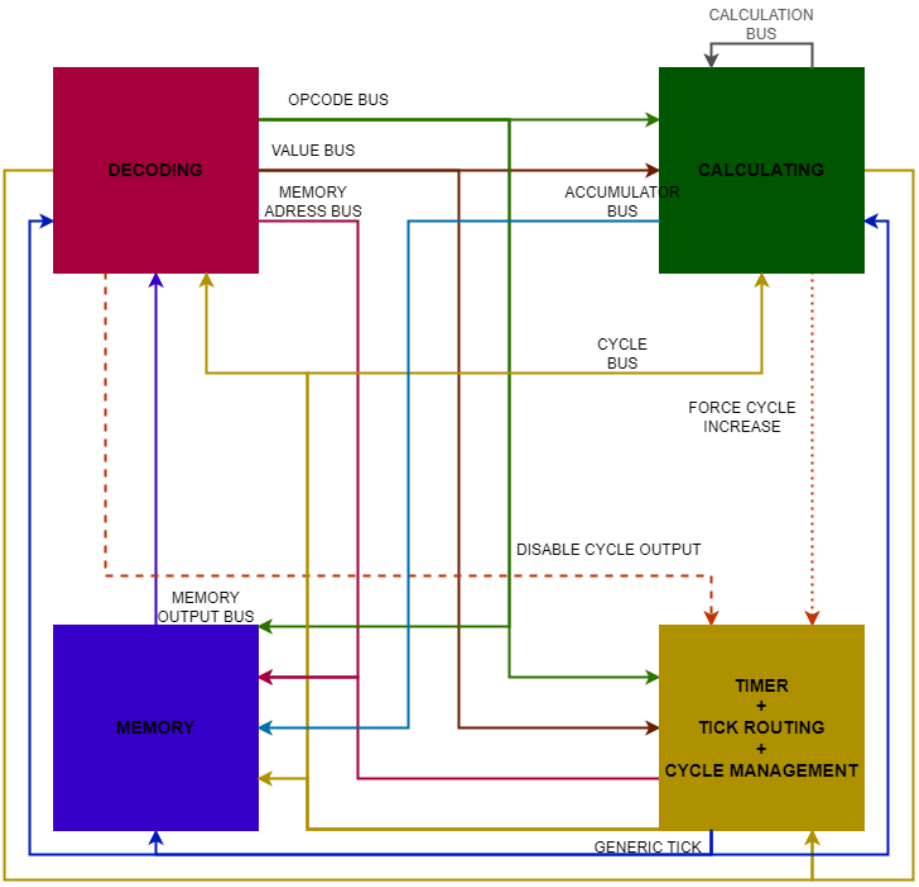
1. Architektura procesoru

Jednotlivé moduly jsou propojeny datovými sběrnicemi a dalšími řídícími a informačními spoji. Každý z modulů přijímá stejná data z hlavní fyzické sběrnice přes stejný typ konektoru. Data jsou následně odfiltrována, zpracována, popřípadě ukládána a výsledky převedeny zpět na hlavní fyzickou sběrnici. Každý modul je řízen pomocí unikátních adres. Tento systém umožňuje náhodné umístění jednotlivých modulů, jednoduchou rozšiřitelnost a přehlednost.

1.1 Konektorová deska

Slouží k propojení jednotlivých modulů pomocí konektorů na hlavní fyzické sběrnici o 62 bitové šířce. Na desce je také několik pull-up rezistorů, sloužících k vyzvednutí napětí na jednotlivých vodičích sběrnic v záporné logice. Díky architektuře, kterou tento systém využívá, je možné rozšíření o další moduly po připojení libovolného počtu konektorových desek.

Obrázek č.1: Blokové schéma - sběrnice.

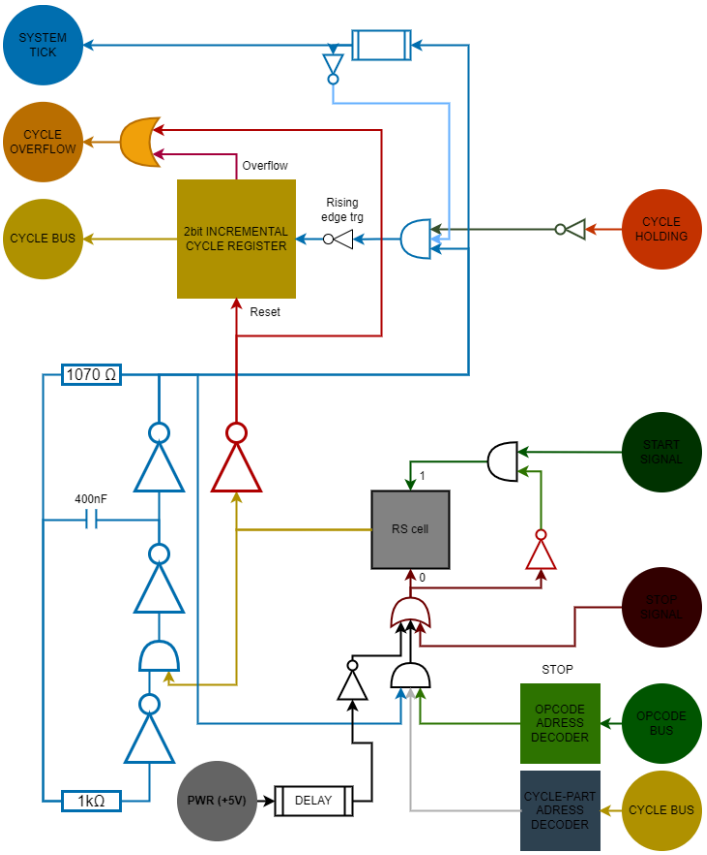


1.2 Modul cyklů

Modul cyklů obsahuje v prvé řadě časovač, složený ze tří vzájemně propojených negací, časovacích kondenzátorů a rezistorů. Jeho výstup je po 100us zpoždění vyveden do systému jako obecný tik hodin. Pro povolení překlápění hodin je nutné, aby byl klopný obvod určující chod systému aktivní. Ten lze aktivovat pouze manuálně a deaktivovat buď manuálně, nebo systémovou operací. Časovací obvod má i možnost změny frekvence mezi učební (1 Hz) a pracovní (1 kHz), volbou připojených kondenzátorů. Proti náhodnému spuštění po zapojení je obvod jištěn časovačem, napojeným přímo na napájecí napětí.

Druhou částí modulu cyklů je obvod, který drží ve své dvoubitové paměti aktuální cyklus procesoru. Jeho výstup je vyveden zpět na hlavní sběrnici, spolu s výstupem informujícím o jeho přetečení. Obvod je navýšen - inkrementován při každém tiku hodin, pokud není momentálně blokován jinou částí systému. Cykly jsou prováděny v následujícím pořadí: posun v programu, dekódování operačního kódu, dekódování hodnoty instrukce, provedení operace.

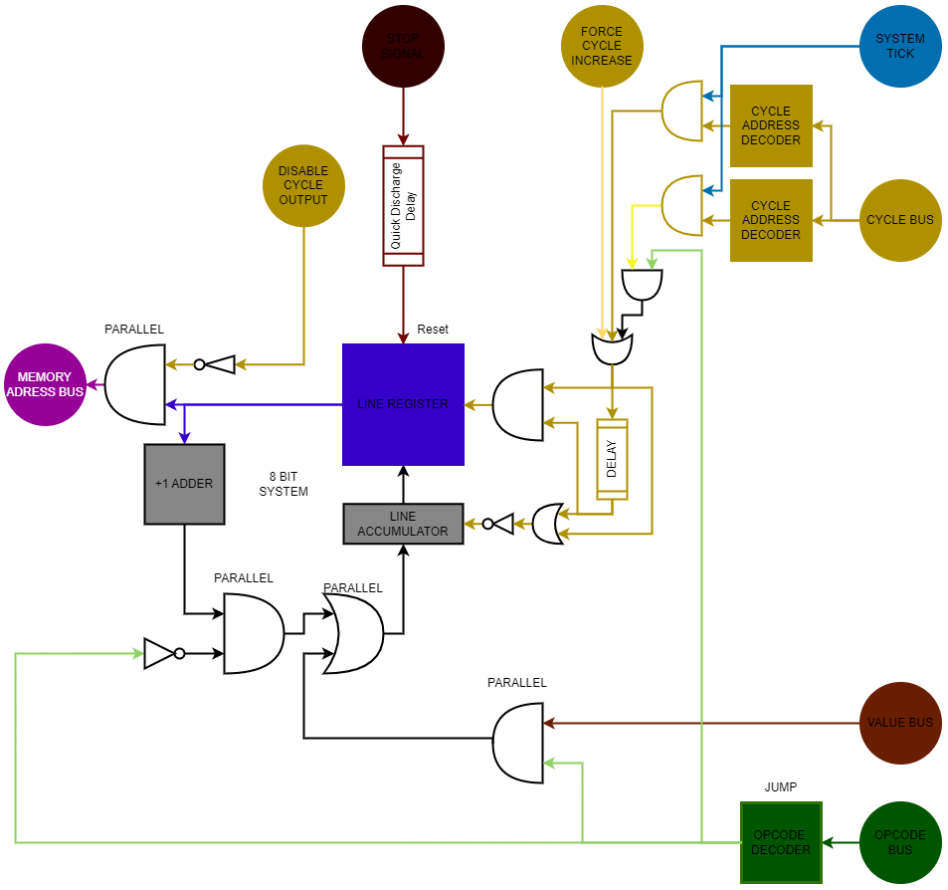
Obrázek č.2: Blokové schéma - modul cyklů.



1.3 Modul programu

Tento modul zajišťuje správnou posloupnost vykonávání programu a provádí v něm i skoky. Jeho hlavní částí je osmibitová paměť, držící aktuální pozici programu. Tato paměť je součástí obvodu, který při přepisu navýší hodnotu o jeden řádek. Jelikož se pracuje s hodnotou, která je v daném čase zároveň i přepisována, obvod využívá dvoufázový zápis. Ten v prvé řadě přepíše akumulátor před pamětí a následně paměť hodnotou z akumulátoru. Tento přepis se spustí vždy v prvním cyklu (posun v programu) nebo je vynucen jiným modulem, například modulem podmíněných skoků. Pokud systém provádí nepodmíněný skok v programu, při kterém je jasně definovaný řádek, na který se má posunout, původní přísun dat je zablokován a paměť je přepsána výstupní hodnotou z dekodéru. Paměť se dá i vyresetovat delším podržením vypínacího tlačítka celého systému. Fyzický výstup paměti držící aktuální řádek programu je přes propustná hradla připojen na hlavní sběrnici. Ostatní moduly mají možnost tento výstup zablokovat a uvolnit tak část sběrnice pro jiná využití.

Obrázek č.3: Blokové schéma - modul programu.



1.4 Modul paměti

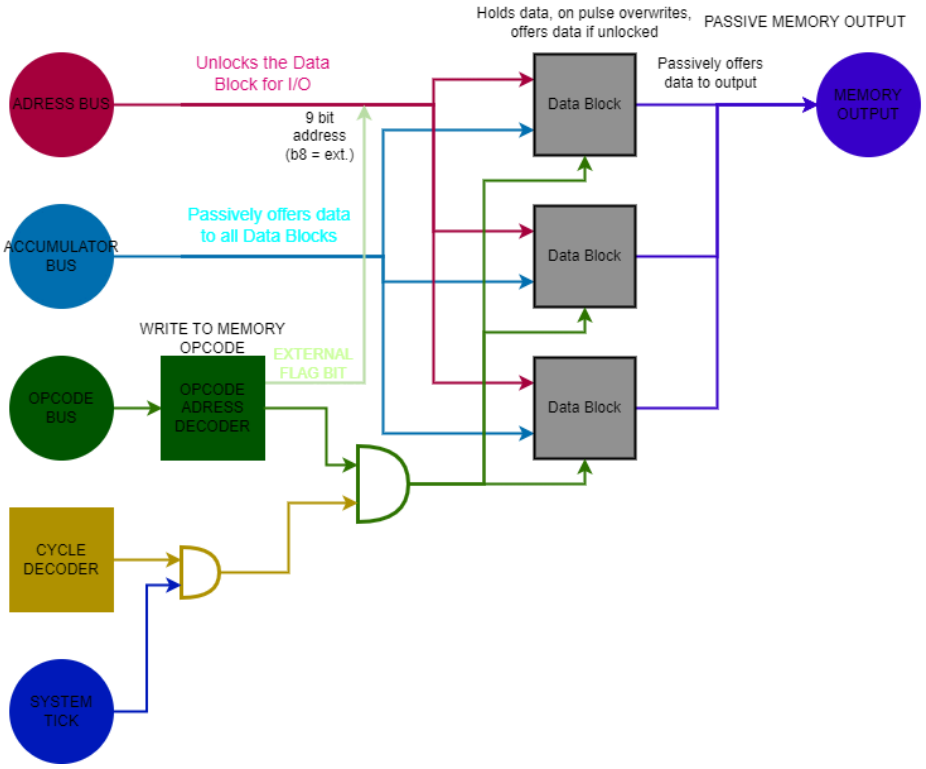
Modul paměti je určen k uchovávání programu a dalších dat systému. Tento modul vyžaduje možnost zápisu a čtení osmibitových informací ze specifikovaných bloků.

Pro povolení zápisu a čtení je adresa příslušně rozšifrována. V případě správné adresy se na daný blok umožní záznam dat a jeho výstup je propouštěn na hlavní fyzickou sběrnici. Kvůli úspoře prostoru a počtu použitých hradel je rozšifrování adres rozděleno do dvou úrovní. Prvních pět bitů určuje požadovaný paměťový modul. Tato část adresy není pevně stanovená a dá se nastavit pomocí propojek přímo na desce. Zbylé 3 bity určují požadovaný blok paměti v modulu. Tato část adresy je pevně stanovená a nedá se změnit. Každý paměťový modul obsahuje 8 paměťových bloků. Jeden blok drží osmibitovou informaci. Díky osmibitové adresaci můžeme zaměřit až 256 různých paměťových bloků na 32 modulech s různými adresami. Celková maximální velikost paměti je 2048 bitů.

Zápis na blok je proveden příslušnou operací, nebo vynuceným přepisem, který nevyžaduje žádné dešifrování operačního kódu, správný cyklus ani tik hodin. Tato forma zápisu se využívá především pro manuální programování paměti.

Jednotlivé buňky paměti, držící samostatné bity, jsou zkonstruovány z hradel OR, zapojených přes hradla AND, spuštěná v klidovém režimu, zpět do vstupu prvního hradla. Při zápisu propustíme informaci na druhý vstup hradla OR, ta se v obvodu zacyklí a zůstane tak uložena až do doby rozepnutí hradla AND, nebo do odpojení napájení. Správný přepis je zajištěn krátkodobým časovačem, který provádí zápis až po smazání dat.

Obrázek č.4: Blokové schéma - modul paměti.



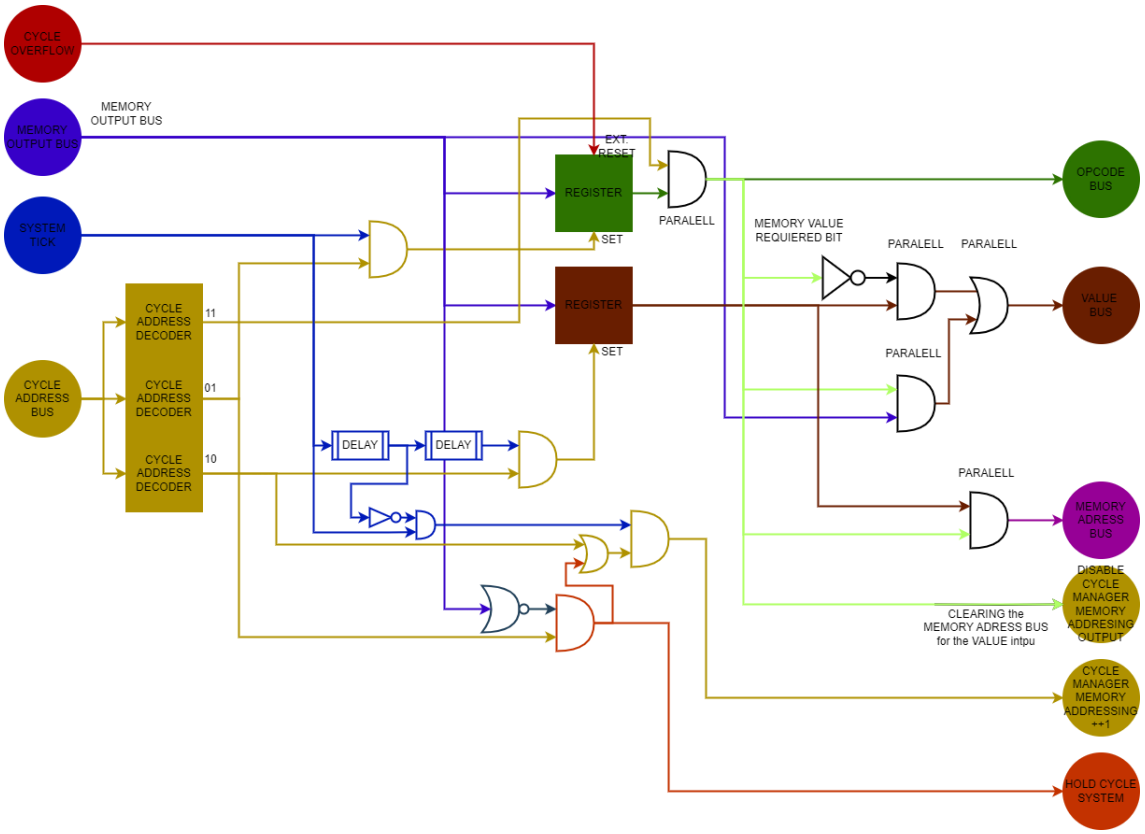
1.5 Modul dekódování

Tento modul rozděluje vstupní informace z paměti, představující jednotlivé řádky programu, na dvě použitelné části. Data jsou propuštěna na hlavní sběrnici až po úspěšném načtení a uložení všech dat do svých registrů.

Nejdříve se modul pokouší o načtení platného operačního kódu. Pokud jsou příchozí informace rovné nule, modul dekódování je vyhodnotí jako vynechanou operaci, vynutí posun programu na další řádek, zablokuje navýšení systémového cyklu a pokusí se operaci načíst znovu. Po úspěšném načtení a uložení operačního kódu do registru, se povolí navýšení systémového cyklu a obvod je připraven pro načtení hodnoty operace. Ta je uložena, i když je rovna nule. Jelikož operační kód upravuje funkci systému, mohl by zabránit správné funkci dekódování. Proto je výstup registru operačního kódu zablokován až do posledního systémového cyklu.

Hodnota operace může představovat i adresu paměťového bloku, na kterém se požadovaná hodnota nachází. Pro určení, jestli se jedná o adresu, nebo přímou hodnotu, je použitý poslední bit operačního kódu jako „informační vlajka". Pokud je tato vlajka přítomná, jedná se o adresu paměti a obvod přizpůsobí tok informací podle potřeby. Hodnota operace je propuštěna na sběrnici adresace paměti, jejímž původním vstupem je aktuální řádek programu. Ten je z přístupu na sběrnici vyblokován. Vstupní informace do modulu dekódování je přivedena na hodnotový výstup. Na hlavní sběrnici se tedy propustí operační kód a hodnota paměťového bloku s adresou hodnoty operace. Systém je tak schopen pracovat s hodnotami přímo naprogramovanými i s těmi uloženými na specifikovaných pozicích v paměti.

Obrázek č.5: Blokové schéma - modul dekódování.



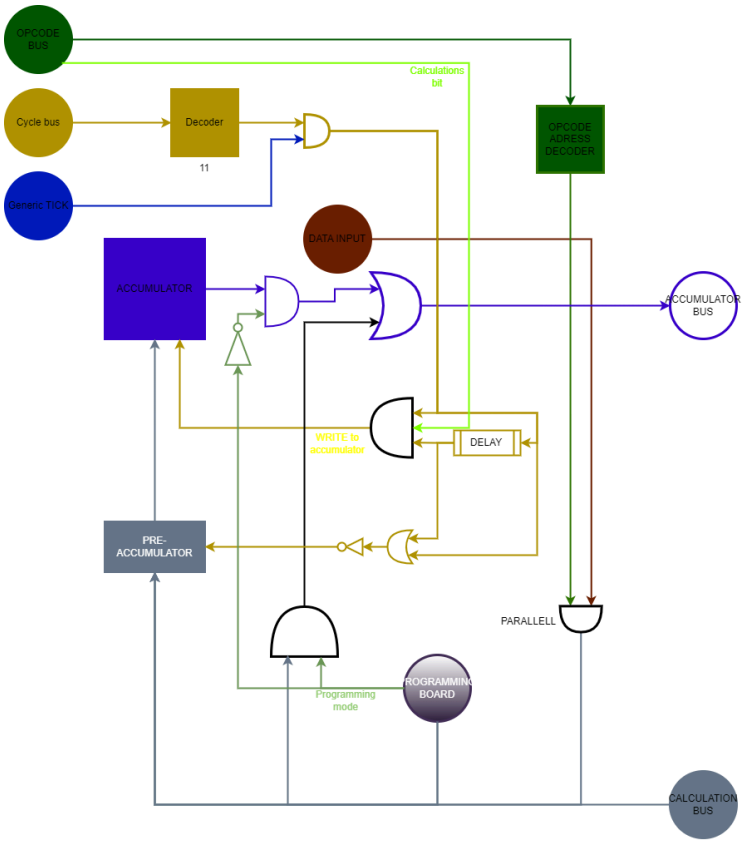
1.6 Modul akumulátoru

Základní modul pro všechny operace pracující s čísly. Jeho hlavní částí je akumulátor, který drží hodnotu pro výpočty, přesun dat aj. Akumulátor poskytuje svoji uloženou hodnotu celému systému pro různá využití. S touto hodnotou pracují téměř všechny výpočetní moduly, které následně poskytují výsledky zpět akumulátoru přes výpočetní sběrnici. Akumulátor také používá dvoufázový přepis dat, aby nedošlo ke změně výsledku výpočtu před dokončením zápisu. Zápis je proveden pouze v případě, že je aktivní informační vlajka, určující operaci s výsledkem. Tato vlajka je součástí operačního kódu.

Tento modul je schopen i přepisu akumulátoru hodnotou operace, pokud je provedena příslušná instrukce.

Pokud je systém v programovacím režimu, původní tok dat je přerušen a data přicházející z výpočetní sběrnice nejsou navedena do akumulátoru, ale přímo na výstup modulu. Tato funkce umožňuje přímé propouštění dat na vstup paměťových modulů, čehož se využívá při programování.

Obrázek č.6: Blokové schéma - modul akumulátoru.



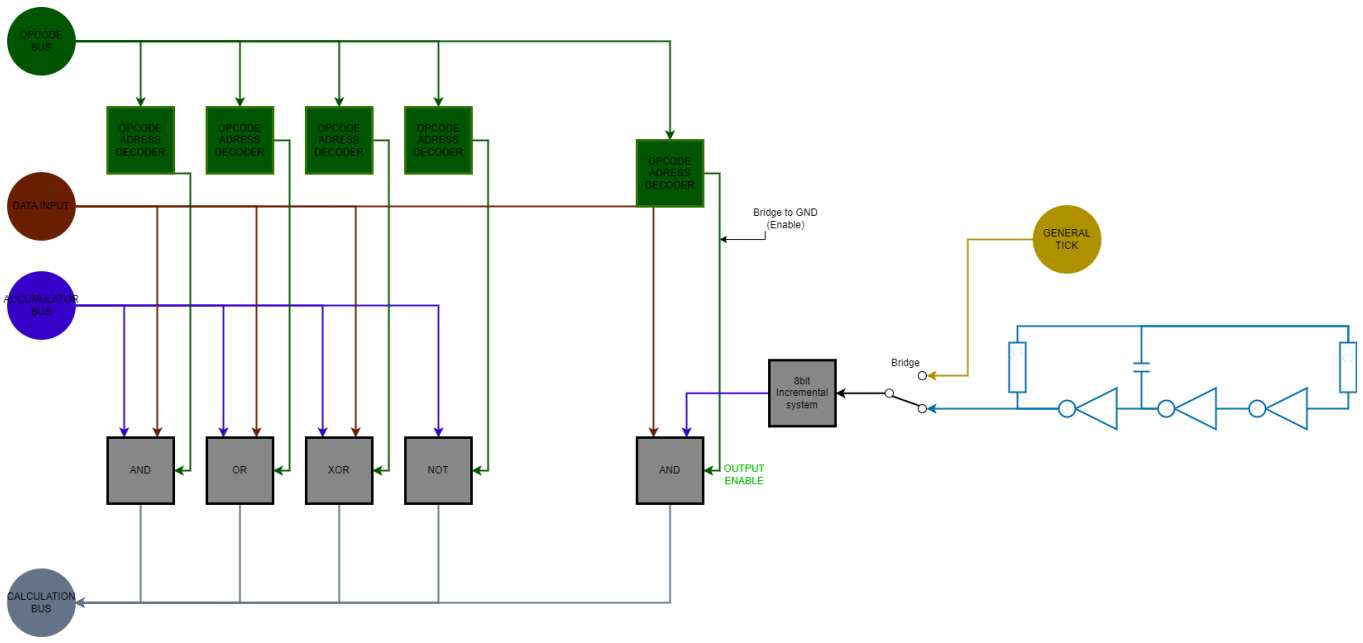
1.7 Modul logických operací a náhodné generace

Tyto dva moduly jsou zkombinovány kvůli úspoře množství desek plošných spojů.

Modul logických operací jednoduše kombinuje hodnotu uloženou v akumulátoru a hodnotu operace po dekódování. Ke každému logickému prvku je přiřazen dekodér operačního kódu. Jedná li se o požadovaný prvek, dešifrování propustí výsledek logického výpočtu na výpočetní sběrnici. Logické operace, vestavěné do modulu, jsou AND (logický součin), OR (logický součet), XOR (nonekvivalence) a NOT (logická negace), který jako jediný nevyžaduje druhou veličinu a je připojen pouze k akumulátoru.

Modul náhodné generace je jednoduchý inkrementální obvod, držící momentální hodnotu, která se při každém řídícím signálu zvýší. Jelikož je složité predikovat jaká hodnota na obvodu v daný okamžik bude, jedná se o takový pseudo-náhodný generátor čísel. Původ tohoto signálu je díky propojce volitelný. Základní možností je hodinový obvod, který je součástí modulu náhodné generace. Tato možnost spíná obvod rychleji a čísla jsou tak hůře predikovatelná. Druhou možností je připojit inkrementální obvod přímo na hodinový signál systému. Pokud je požadovanou operací generace náhodného čísla, propustí se logický součin pseudo-náhodného čísla a hodnoty operace. Tímto se při programování navolí bity, které chceme vygenerovat.

Obrázek č.7: Blokové schéma - modul logických a náhodných operací.



1.8 Modul aritmetických operací a binárních posuvů

Je jedním z nejkomplexnějších modulů celého systému. Jeho účelem je provádění aritmetických výpočtů (sčítání, odčítání, násobení, dělení) a binárních posuvů do obou stran o požadovaný počet číslic.

Hlavní částí modulu je postupný součtový obvod. Ten je vyžadován pro všechny prováděné matematické operace. Neustále sčítá dvě proměnné a v případě že je aktivován jinou částí modulu, propustí výsledek na výpočetní sběrnici. Poskytuje také informaci o přetečení nebo nulovém výsledku. Na začátku modulu je operační kód rozšifrován a podle požadované operace je zbytek obvodu příslušně nastaven.

Sčítání je provedeno přímo v součtovém obvodu. Číslo v akumulátoru je spolu s hodnotou operace přivedeno na součtový obvod, který je aktivován, a tak propouští výsledek součtu na výpočetní sběrnici.

Odčítání je v základu provedené stejně jako sčítání. Jediným rozdílem je, že odečítaná hodnota (hodnota operace) není přivedena na součtový člen přímo, ale přes negace a obvod navyšující hodnotu o 1. Výsledkem sečtení takto upraveného čísla je jeho odečtení.

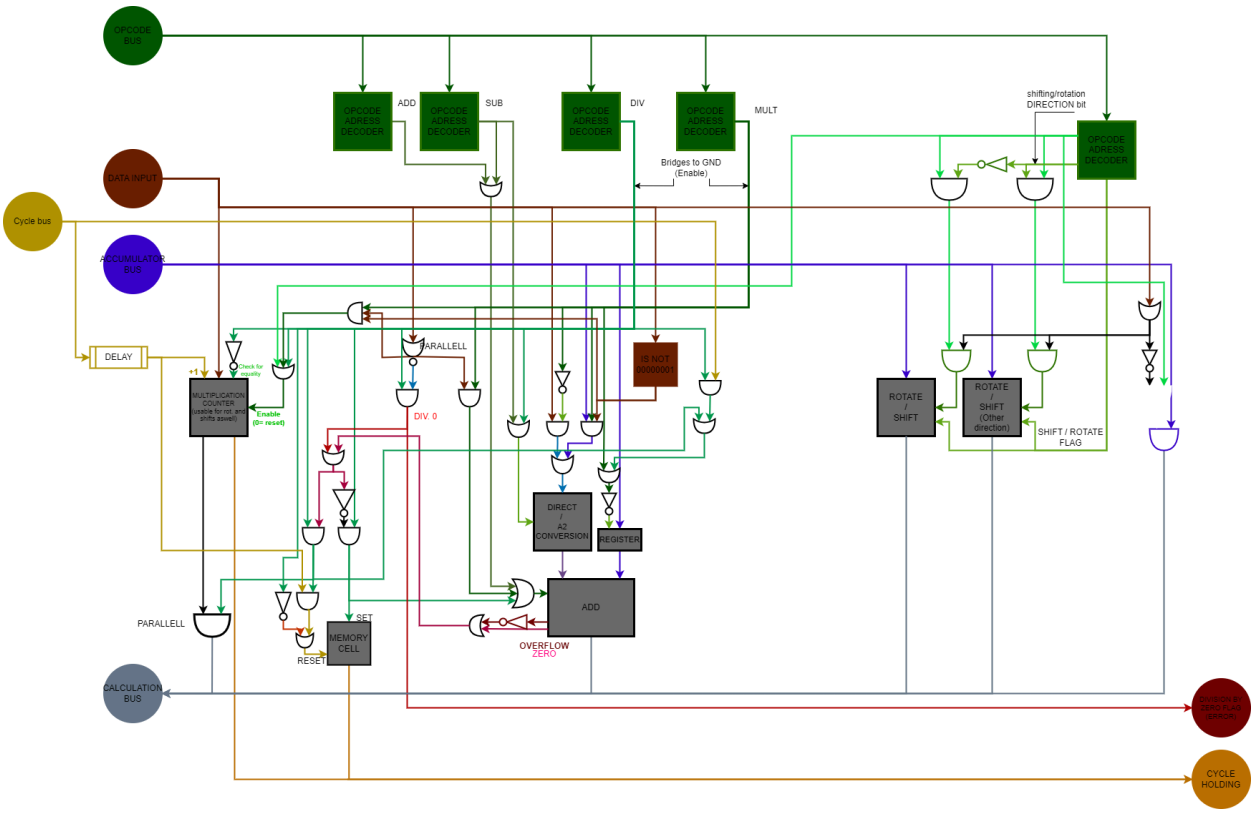
Násobení vyžaduje inkrementální obvod, který svoji hodnotu navýší při každém pulzu systémových hodin. Tyto pulzy zároveň zapisují výsledek sčítání v modulu akumulátoru. Na součtový obvod je přivedena hodnota akumulátoru do obou vstupů. Jeden ze vstupů prochází přes registr, který drží svoji hodnotu od počátku operace. Obvod tak opakovaně přičítá původní číslo k výsledku. Inkrementální obvod blokuje navyšování cyklu systému, dokud se jeho číslo a hodnota operace neshodují. Obvod tak přičte požadované číslo samo k sobě tolikrát, kolikrát je určeno programem.

Dělení také využívá inkrementální obvod, který ale nekontroluje rovnost čísel a navyšování cyklu systému je blokováno přímo operací dělení. Obvod je přepnut do režimu odečítání a opakovaně se z akumulátoru odečítá požadovaná hodnota. Při každém odečtení dojde k přetečení součtového obvodu. Jakmile k němu nedojde, nebo se na výsledku zobrazí nula, zapíše se do akumulátoru hodnota inkrementálního obvodu a znovu se povolí navyšování cyklu systému. Výsledkem je tak počet možných odečtení hodnoty operace od čísla v akumulátoru. Při dělení nulou je operace okamžitě vyhodnocena jako ukončená a modul pošle do systému chybové hlášení, zachytitelné a zpracovatelné v programovacím modulu.

Binární posuv je realizován pouhým přepojením pozic vstupních a výstupních vodičů. Obvod je schopen i binárních rotací pomocí hradel AND, která vedou poslední bit na pozici prvního a naopak. Různé směry operace jsou řešeny dvěma zrcadlově překlopenými obvody. Pro posuv nebo rotaci o daný počet míst je znovu využitý inkrementální obvod, který porovnává počet za sebou provedených posuvů s hodnotou operace.

Modul obsahuje ještě další ochranné prvky. Například, jedná-li se o opakovanou operaci, v tuto chvíli bez opakování, operand je propuštěn přímo jako výsledek. Systém také hlídá, jestli nemá dojít k nulovému výsledku. Dalším prvkem je paměťová buňka, která zajišťuje, že operace dělení neskončí dříve, než se zapíše konečný výsledek. Inkrementální obvod je neustále resetován, až do úspěšného rozšifrování požadované operace.

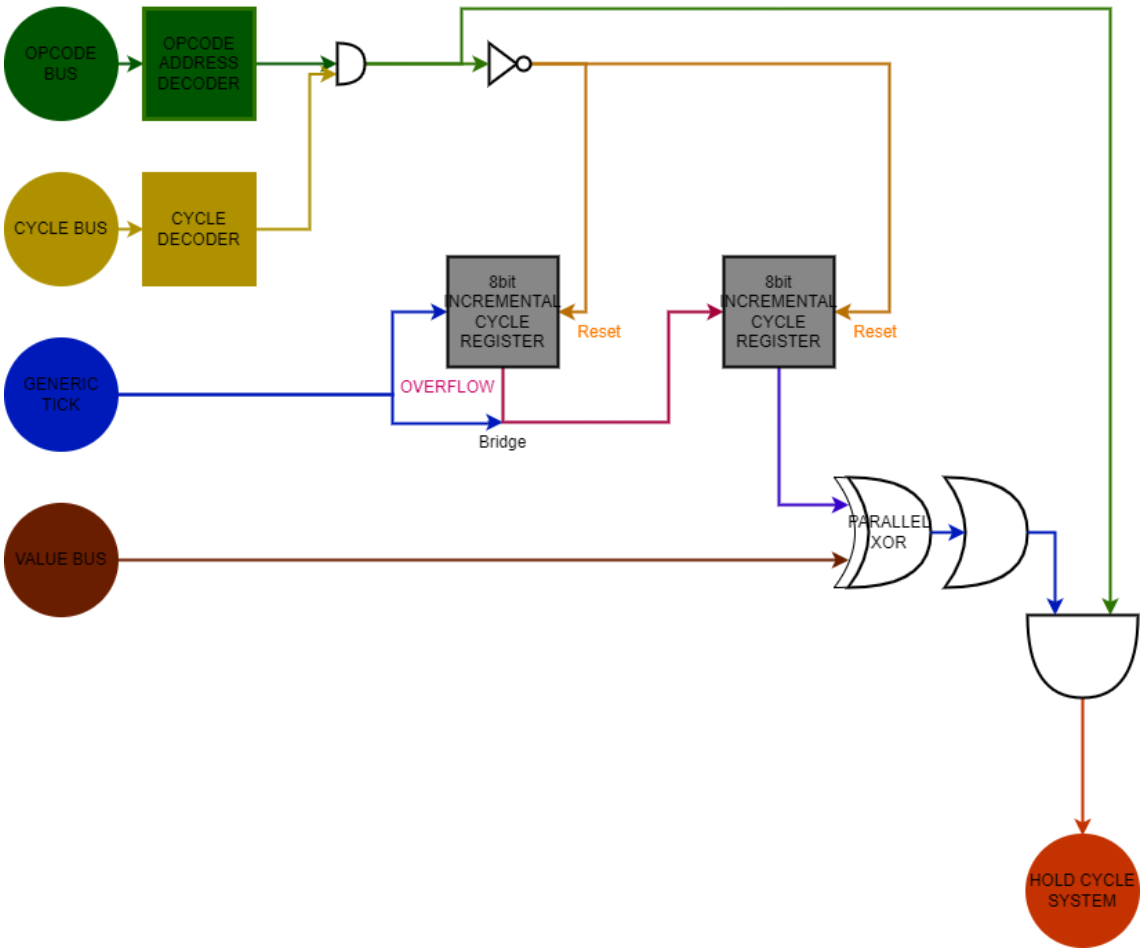
Obrázek č.8: Blokové schéma - modul aritmetických operací a binárních posuvů.



1.9 Modul pozastavení programu

Jednoduchý modul, využitelný jako dočasné pozastavení celého systému. Jeho základem je inkrementální obvod, jehož hodnota je navýšena při každém pulzu hodin od doby rozšifrování požadované operace. Ta také zablokuje navyšování cyklu systému. Inkrementální obvod je osmibitový. Udrží tak 256 promeškaných hodinových pulzů a jeho přetečení navýší hodnotu dalšího osmibitového inkrementálního obvodu. Ten je porovnáván s hodnotou operace a v případě rovnosti je modul cyklů odblokován a procesor pokračuje dále v programu. Vstup druhého obvodu se dá pomocí propojek přemostit přímo na hodinové pulzy, čímž se podstatně zkrátí doba čekání a navýší přesnost. Systém je tak schopen "prospat" 0 - 255 hodinových pulzů, nebo až 65 536, ale s rozlišením 256.

Obrázek č.9: Blokové schéma - modul pozastavení programu.



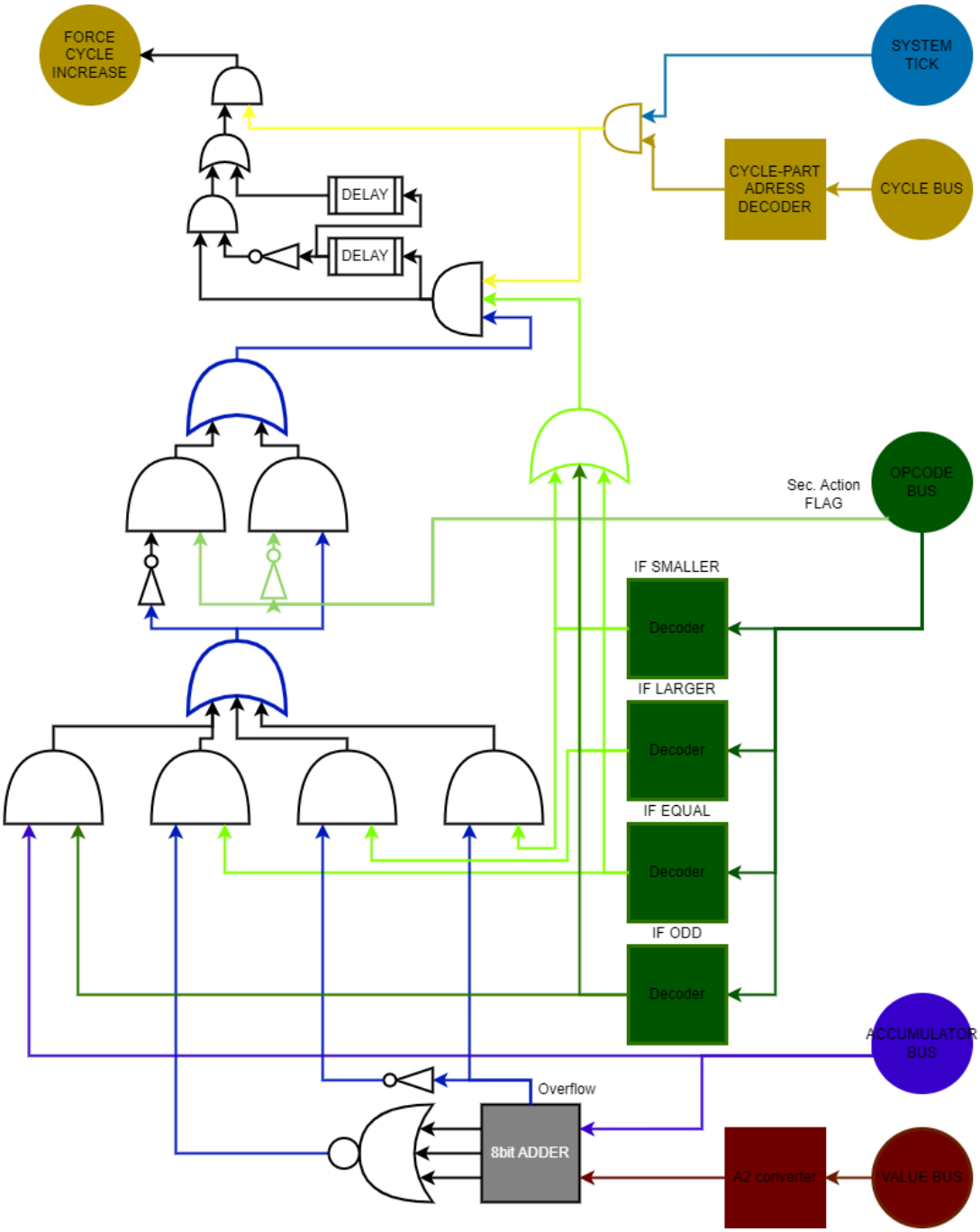
1.10 Modul podmíněných skoků

Podmíněné skoky jsou v programu využity jako způsob kontroly splnění různých podmínek. Je-li požadovaná podmínka splněna, program provede následující řádek kódu, jinak program následující řádek vynechá.

Podmínky se určují podle vzájemného stavu dvou čísel (např. A>B, B!=A, …). Ke kontrole jakékoli podmínky nám stačí znát rozdíl hodnot. Je-li rozdíl roven nule, hodnoty se shodují. Je-li výsledek záporný, druhá hodnota je větší a naopak. Základním prvkem modulu je tedy osmibitový sčítací obvod a převodník hodnoty operace na zápornou hodnotu. Kombinace těchto obvodů dosahuje odčítání.

Ke každé podmínce patří příslušný obvod dešifrování. Pokud jde o požadovanou podmínku a stav sčítacího obvodu nebo jeho výsledku potvrzuje splnění dané podmínky, program postupuje nerušeně dál. V opačném případě se spustí třífázový časovač, který dvakrát vyšle signál pro navýšení aktuálního řádku programu a následně je zablokován, aby nedošlo k dalším nechtěným skokům. Je-li aktivní příslušná informační vlajka, program se ptá na nesplnění podmínky a obvod invertuje informaci o jejím splnění.

Obrázek č.10: Blokové schéma - modul podmíněných skoků.



1.11 Modul programování, vstupů a výstupů

Modul, který není nezbytně potřebný pro správnou funkci procesoru. Jeho hlavním účelem je manuální ovládání a monitorování systému.

Jeho součástí je soustava několika 8 bitových LED displejů, zobrazujících aktuální hodnoty na všech sběrnicích, a také samostatné LED diody, zobrazující ostatní informace systému, zachytitelné z hlavní fyzické sběrnice.

Další částí je ovládání chodu systému, pomocí dvou tlačítek (START a STOP). Při delším podržení zastavujícího tlačítka dojde k vynulování registru, uchovávajícího aktuální řádek programu. Aktivace určité funkce je signalizována pomocí příslušných LED diod.

Manuální programování paměti, kterému napomáhá i zobrazovací část modulu, je provedeno pomocí dvou bloků po osmi spínačích. Jeden ovládá adresu požadovaného bloku pro zápis. Druhý určuje osmibitovou informaci, která bude na daný blok paměti zapsána. Před začátkem programování je vhodné systém zastavit a přepnout do programovacího režimu pomocí příslušného spínače. Až poté jsou hodnoty vysílány na sběrnici a dají se sledovat na zobrazovacích led diodách. Zápis je proveden stisknutím příslušného tlačítka. Při úspěšném vyslání zápisového signálu je rozsvícena příslušná signalizační LED dioda.

Modul obsahuje i registr, který zachytává chybové hlášení systému (např. dělení nulou). Po zachycení informuje uživatele pomocí LED diody. Registr se dá vyresetovat příslušným tlačítkem.

Poslední částí modulu jsou externí vstupy a výstupy. Ty fungují na stejném principu jako paměťové moduly systému. Pro jejich zaměření je však potřeba příslušné informační vlajky, která je součástí operačního kódu. Modul obsahuje dva jedenáctipinové porty, které vysílají osmibitová data, informaci o přítomnosti hodnoty, přijímají signál požadavku o externí resetaci registru a poskytují uzemnění. Modul také obsahuje dva osmibitové vstupy, které jsou připojeny na osm spínačů a osm tlačítek přímo na modulu. Jak vstupy, tak i výstupy modulu dešifrují svoji adresu pomocí nonekvivalence. Jejich adresy se tak dají nastavit pomocí příslušných spínačů.

Blokové schéma - modul

2. LOGICKÉ ZAPOJENÍ

Hlavním pilířem projektu je jeho sestavení pouze z logických hradel. Je proto zapotřebí navrhnout zapojení všech bloků do nejmenších podrobností.

Na logickém zapojení je zřetelně patrná jedna ze základních myšlenek projektu. Celý systém, velice komplexní, je sestaven z velkého množství jednoduchých součástek, stejně jako moderní procesory. Neustálé opakování různých kombinací logických hradel tvoří celek, schopný komplikovanějších funkcí. Postupným přidáváním bran dosáhneme možnosti provádět základní i složitějších operace a výsledně jsme schopni sestavit celý funkční procesor.

Logické zapojení, stejně jako celá architektura procesoru, se snaží využít opakovatelně použitelného zapojení různých obvodů pro zjednodušení návrhu a sekvenčních obvodů, oproti větším kombinačním, pro snížení počtu potřebných hradel. Dojde tak ale k prodloužení doby strávené na určitých operacích. Většina logického zapojení je již součástí návrhu architektury. Nepopsané bloky, jako například "inkrementální obvod", jsou také složeny z logických hradel. Tato zapojení jsou k nahlédnutí ve formě schémat v příloze přiložené k dokumentaci. (viz. přílohy č. 12 - 22)

3. Elektrické zapojení

Kvůli velkému množství hradel, nacházejících se na jednotlivých deskách, bylo nutné vymyslet efektivní, ale hlavně opakovatelný způsob zapojení. Vnitřní zapojení většiny modulů je proto provedeno stejným stylem.

Jedna strana desky plošných spojů je převážně využívána pro umístění integrovaných obvodů a vyvedení jejich kontaktů do prokovů (propojení obou stran desky plošných spojů) v různých vzdálenostech v čistě horizontální rovině. Druhá strana propojuje prokovy se sběrnicemi, nebo mezi sebou. Její cesty jsou převážně vertikální, tím je zijištěno, že si navzájem nepřekážejí.

Integrované obvody musí být logicky umístěny do sloupců, podle na sebe navazujících vývodů. Jelikož jsou sběrnice navrženy po osmi bitech a jeden integrovaný obvod obsahuje většinou 4 hradla, je schopen zpracovat přesně polovinu sběrnice. Obvody, zpracovávající informace sběrnice, jsou proto často rozděleny do dvou podobných sloupců.

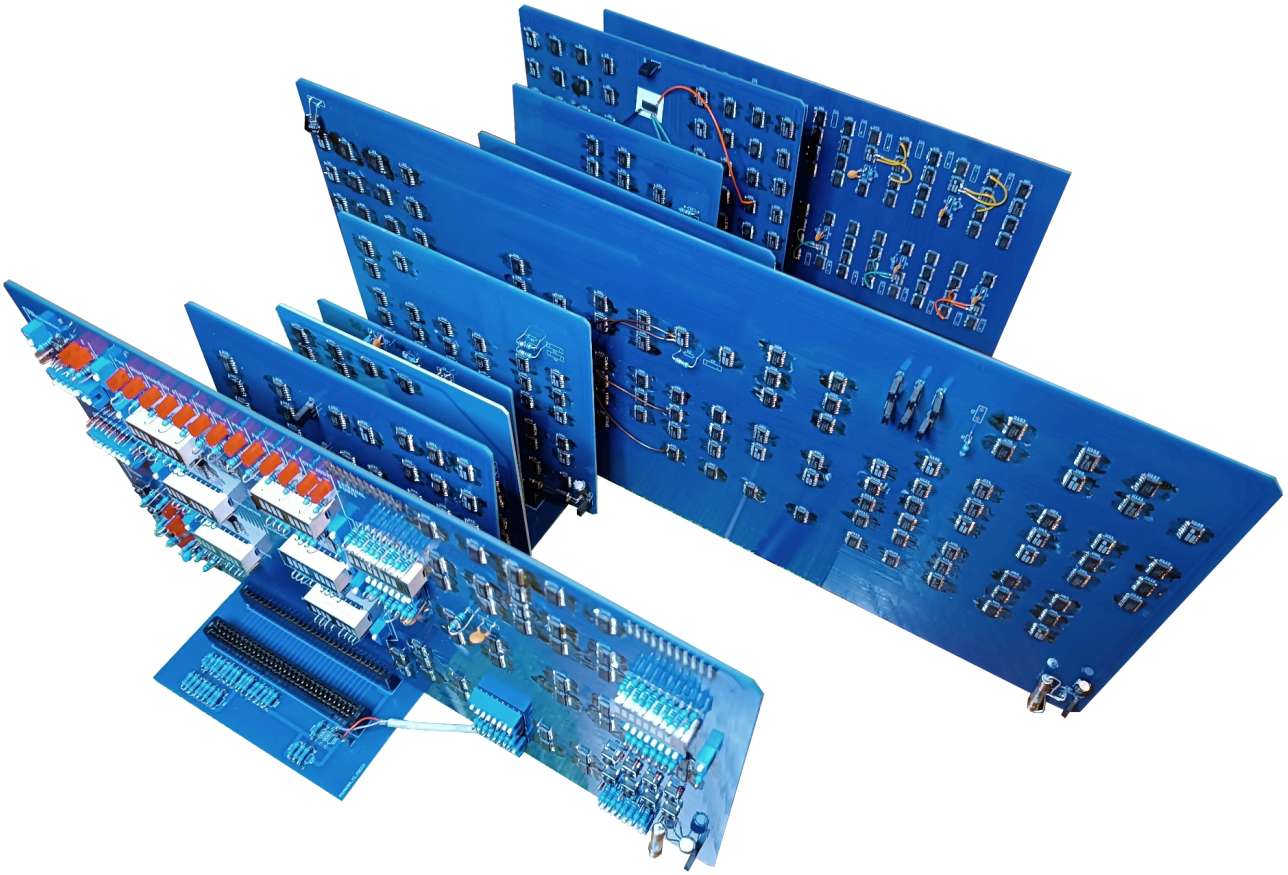
Paměťové moduly jsou rozděleny do bloků držících jednu osmibitovou informaci. Ta je uložena do dvou téměř identických sloupců integrovaných obvodů. Po jejich boku je sloupec hradel, řídící celý blok paměti. Tato struktura je prostorově efektivní a velice snadno replikovatelná, což je u pamětí důležitým aspektem.

Inkrementální obvody jsou složeny z několika identických bloků. Každý z nich zpracovává jeden bit výsledné informace. Takový blok se skládá z osmi NAND a několika dalších řídících hradel. Nejjednodušší způsob sestavení takového bloku je řádek čtyř hradel propojených v řádcích nad nimi. Finální verze návrhů jsou k nahlédnutí jako přílohy přiložené k dokumentaci. (viz. přílohy č. 23 - 33)

4. KONSTRUKCE

Desky plošných spojů byly zakoupeny z čínské firmy JLCPCB, s vyleptanými cestami a hotovými prokovy. Finální osazování bylo provedeno manuálně. Všechny použité součástky jsou rozepsány do tabulek. Součástí konstrukce bylo i průběžné testování různých částí systému, v případě potřeby manuálního přemostění různých spojů pomocí vodičů, nebo výměny typu hradla. Osazené desky plošných spojů jsou k nahlédnutí v sekci přílohy dokumentace. (viz. přílohy č. 1 - 11)

Obrázek č.11: Dokončená realizace projektu.



Tabulka č.1: Seznam použitých součástek.

Součástka	Poznámka	Množství	Cena
Integrovaný obvod AND	74HC08D (SOP-14)	283	877 Kč
Integrovaný obvod OR	74HC32D (SOP-14)	174	209 Kč
Integrovaný obvod NOT	74HC04D (SOP-14)	136	163 Kč
Integrovaný obvod NAND s kolektorovým výstupem	74HC03D (SOP-14)	96	576 Kč
Integrovaný obvod NAND	74HC00D (SOP-14)	68	313 Kč
Integrovaný obvod XOR	74HC86D (SOP-14)	28	193 Kč
Integrovaný obvod NOR 3vstupový	74HC27D (SOP-14)	32	260 Kč
Kolíková lišta	rovná	92	15 Kč
Kolíková lišta	v pravém úhlu	22	5 Kč
Jumper do kolíkové lišty	-	26	54 Kč
Kondenzátor časovací	keramický 1nF	21	36 Kč
Kondenzátor časovací	keramický 1uF	3	9,3 Kč
Kondenzátor stabilizační	polarizovaný 47uF	19	47 Kč
Odpor časovcí	100 kΩ	26	26 Kč
Odpor vytahovací	4,7 kΩ	69	70 Kč
Odpor pro led diody	470 Ω	87	90 Kč
Odpor pro hodiny	470 kΩ	1	1 Kč
Odpor pro hodiny	1 kΩ	2	3 Kč
Dioda usměrňovací	-	36	36 Kč
Spínač DIP x1	-	10	10 Kč
Spínač DIP x8	-	7	112 Kč
Tlačítko	-	12	14 Kč
LED dioda	-	19	13 Kč
LED dioda x10	-	8	80 Kč
Držák pojistky	-	28	59 Kč
Hliníkový profil	-	-	1330 Kč
ISA slot	-	16	512 Kč
Desky plošných spojů	-	14	8300 Kč

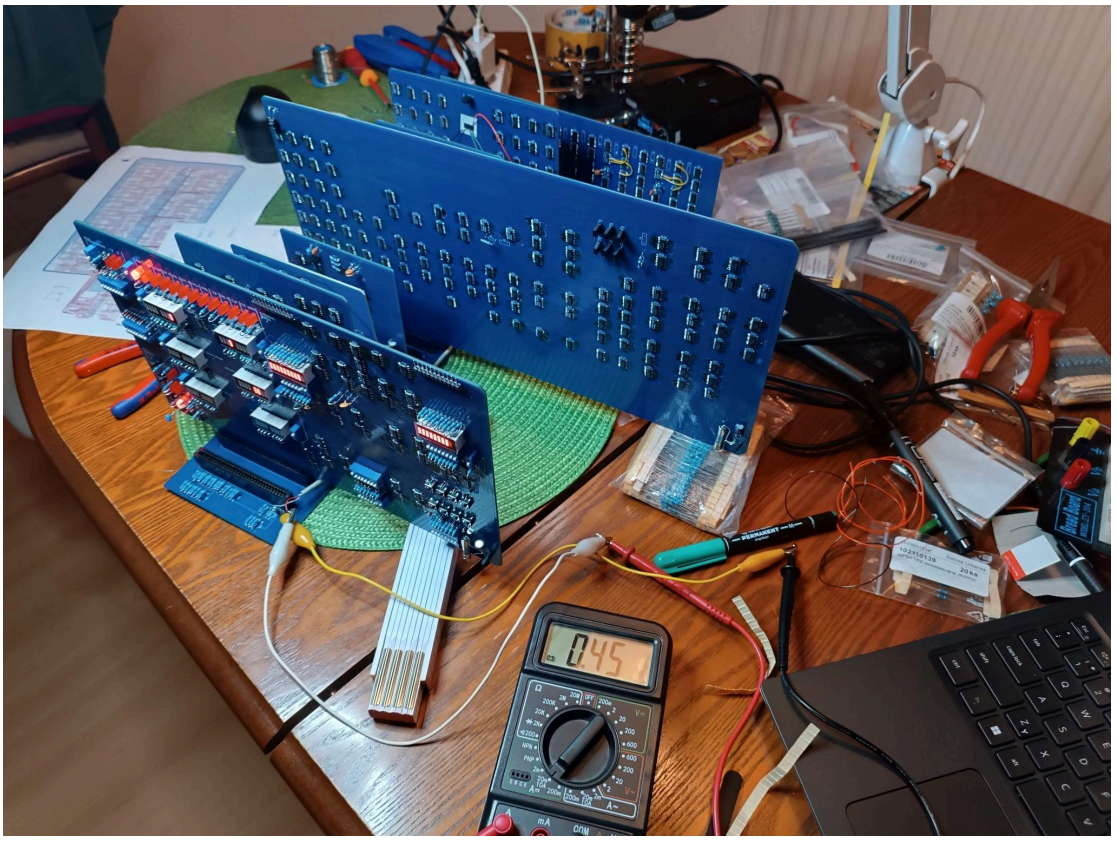
5. TESTOVÁNÍ

Obvod byl kvůli odhalení chyb, provedených ve fázi elektrického nákresu, testován průběžně již ve fázi konstrukce. Nejdříve byla úspěšně zkonstruována a otestována přední programovací deska plošných spojů, jejíž zobrazovací systém a možnost základního posílání dat do systému značně zjednodušili a urychlili testování zbylých modulů.

Při testování byl odhalen nespočet malých konstrukčních chyb, které byly následně opraveny. Některé z nepřesností v elektrickém nákresu vyžadovaly úpravy zapojení. Ty byly kvůli snížení nákladů a doby potřebné k vypracování projektu provedeny většinou vodičem, přemosťujícím chybně zapojené kontakty.

Testování se tak prokázalo nejen užitečným ale i potřebným. Ve finální verzi produktu jsou téměř všechny defekty opraveny a systém je plně funkční. Otestování každého možného využití systému by bylo příliš náročné a zdlouhavé. Zatím je tedy nutné se spoléhat na částečné otestování, které odhalilo většinu chyb. Je ale možné, že systém obsahuje další opravitelné konstrukční chyby.

Obrázek č.12: Oživení a testování procesoru.



6. FUNKCE

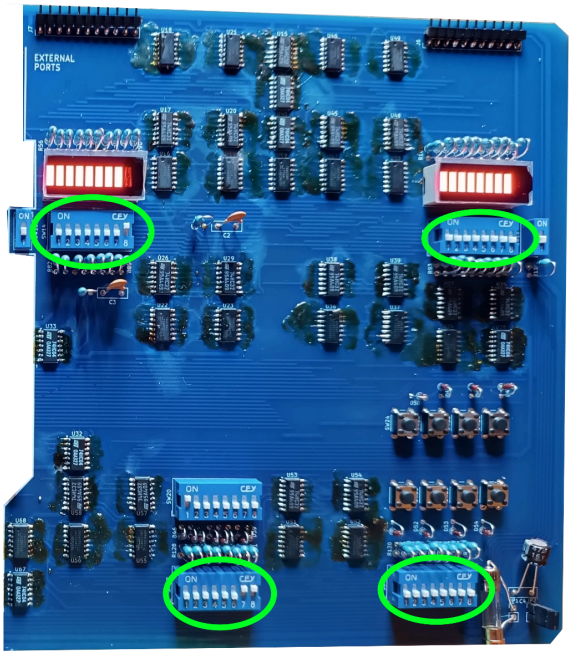
Jelikož je sestavovaným projektem procesor, je možné naprogramovat téměř jakýkoliv program pomocí jeho modulů. Systém je připraven i na prodlužování hlavní fyzické sběrnice, což by umožnilo připojení dalších modulů, rozšiřujících funkce procesoru. Základní moduly, které se v systému nachází, jsou postačující pro sestavení nespočtu využitelných programů. Procesor lze pomocí externích portů na programovacím modulu připojit k dalším zařízením, dále rozšiřujícím možnosti jeho využití.

Tato kapitola slouží jako popis vyzkoušení různých předem otestovaných programů, rozepsaných do několika podkapitol.

K nastavení procesoru do správného režimu se využívá několika příslušných prvků:

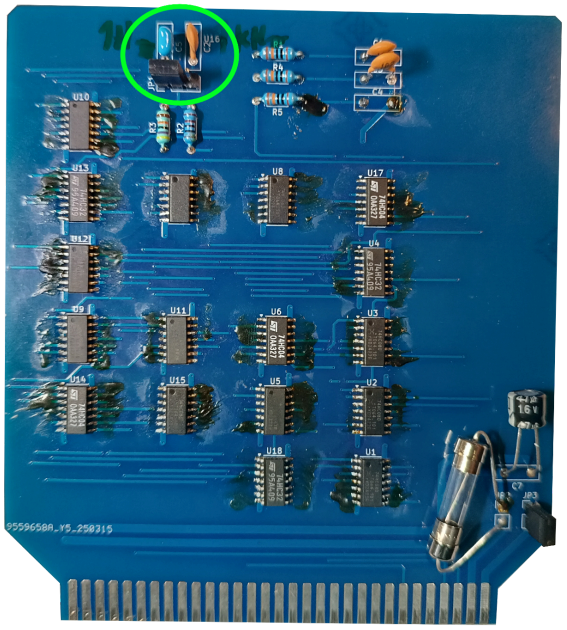
Externí porty (vstupní i výstupní) mají nastavitelnou adresu. K jejímu určení slouží fyzické přepínače pod příslušným prvkem.

Obrázek č.12: Označení adresace externích portů.



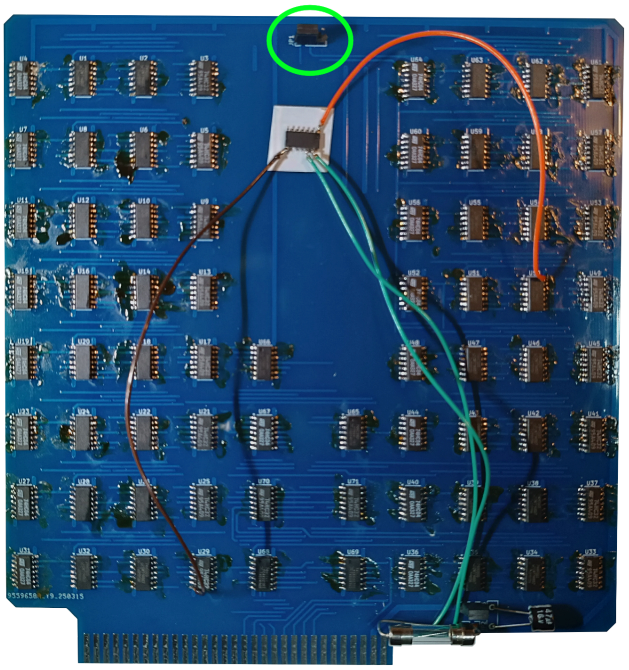
Pro nastavení frekvence procesoru se využívá propojky na modulu cyklů. Nastavitelné frekvence jsou 1 Hz (pro testování a učení) a 1 kHz (pro rychlý chod procesoru).

Obrázek č.13: Označení nastavení frekvence procesoru.



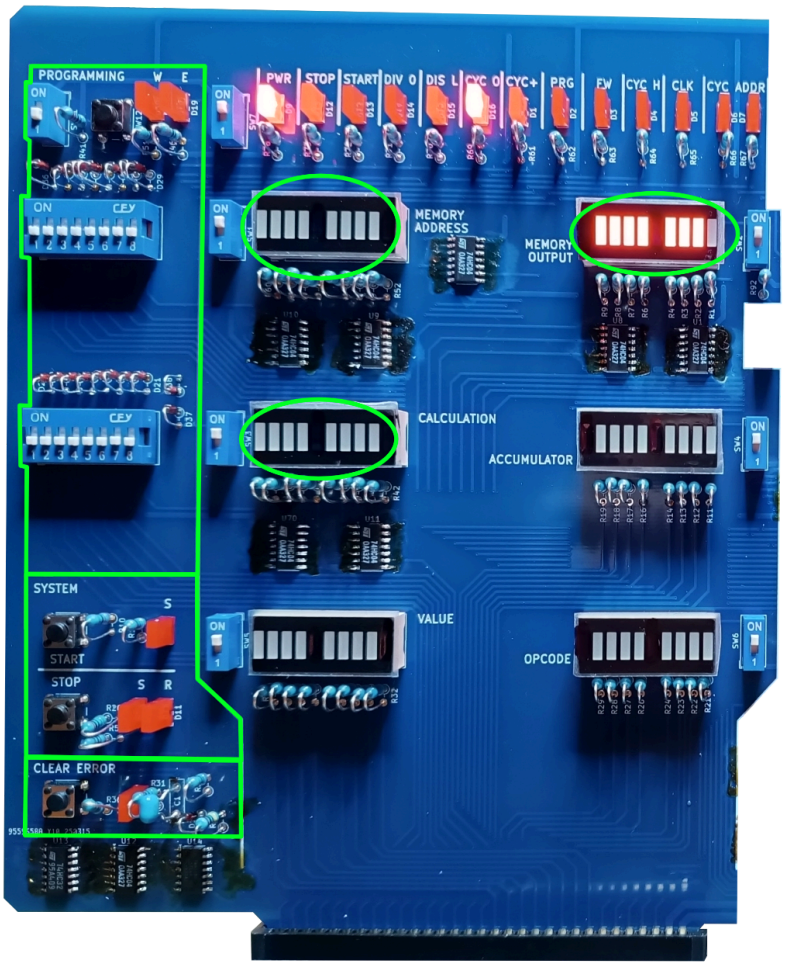
Pro přepnutí režimu časování pozastavovacího modulu slouží další propojka. Ta nastavuje chod mezi přesnějším blokem, schopným přeskočení až 255 hodinových pulzů a delším blokem, který umožňuje přeskočení až 65 280 pulzů, ale s rozlišením pouze 256. Režim časování se ve většině případů nastavuje podle příslušné frekvence procesoru.

Obrázek č.14: Označení nastavení režimu pozastavovacího modulu.



Na levé straně přední části procesoru se nacházejí všechny komponenty, potřebné k jeho základnímu ovládání.

Obrázek č.15: Označení částí pro ovládání procesoru.



Ve vrchní části, označené “PROGRAMMING”, se nachází spínač pro sepnutí programovacího režimu, jehož stav je znázorněn příslušnou led diodou označenou “E” pro Enable. Ten přenastaví systém tak, aby bylo možné s pamětmi pracovat manuálně. Vedle něj je tlačítko pro zápis nastavených hodnot do určeného paměťového bloku. Při pokusu o přepis se rozsvítí led dioda označená “W” pro Write. První (vyšší) z osminásobných spínačů nastavuje požadovanou adresu paměťového bloku, do kterého bude zápis proveden. Druhý (nižší) ze spínačů určuje hodnotu, která bude do paměťového bloku vepsána. Obě hodnoty jsou viditelné na displejích vedle nich. Hodnota uložená v zaměřené paměti je zobrazena na displeji označeném “MEMORY OUTPUT”. Pro programování je k dispozici tabulka operačních kódů níže.

Další část, označená “SYSTEM”, je využívána k ovládání chodu procesoru. Tlačítko “START” spustí hodinový obvod procesoru. Druhé tlačítko “STOP” jej zastaví a při

prodlouženém držení vynuluje pozici v programu, na kterém se procesor nachází. Každá z akcí je symbolizována příslušnou led diodou.

Poslední část “CLEAR ERROR” zachycuje chybové hlášení systému. Pokud nastane, led dioda se rozsvítí. Její zhasnutí můžeme vynutit jedině manuálně pomocí příslušného tlačítka.

Tabulka č.2: Seznam operačních kódů.

ACTION	OPCODE								VALUE								
	ACTION BLOCK ADRESS				S	F	E	M	b7	b6	b5	b4	b3	b2	b1	b0	
Pass / Wait	0	0	0	0	0	F	0	0	NUMBER OF CYCLES								Sec. = Wait
Jump	0	0	0	1	0	-	-	M	CYCLE NUMBER								
Load	0	0	0	0	1	0	E	M	NUMBER / MEMORY								Ext. = I/O Ports
Store	0	0	1	0	0	1	E	M	0 / MEMORY								Ext. = I/O Ports
Random	0	0	0	1	1	-	-	M	SPECIFIED BITS								
Shift / Rotate (LEFT)	0	0	1	0	1	F	-	M	-	-	-	-	COUNT				Sec. = rotate
Shift / Rotate (RIGHT)	0	0	1	1	1	F	-	M	-	-	-	-	COUNT				Sec. = rotate
Negation	0	1	0	0	1	-	-	-	-	-	-	-	-	-	-	-	
Logic multiplication	0	1	0	1	1	-	-	M	MEMORY / BIN. SEQ								
Logic sum	0	1	1	0	1	-	-	M	MEMORY / BIN. SEQ								
Logic exclusive sum	0	1	1	1	1	-	-	M	MEMORY / BIN. SEQ								
Addition	1	0	0	0	1	-	-	M	MEMORY / NUMBER								
Subtraction	1	0	0	1	1	-	-	M	MEMORY / NUMBER								
Multiplication	1	0	1	0	1	-	-	M	MEMORY / NUMBER								
Division	1	0	1	1	1	-	-	M	MEMORY / NUMBER								
If Greater	1	1	1	0	0	F	-	M	MEMORY / NUMBER								Sec. = not gr.
If Equal	1	1	1	1	0	F	-	M	MEMORY / NUMBER								Sec. = not eq.
If Lesser	1	1	0	1	0	F	-	M	MEMORY / NUMBER								Sec. = not le.
If ODD	1	1	0	0	0	1	-	M	MEMORY / NUMBER								
If EVEN	1	1	0	0	0	0	-	M	MEMORY / NUMBER								
Stop	1	1	1	1	1	1	1	1	-	-	-	-	-	-	-	-	

6.1 JEDNODUCHÁ ROVNICE

Jeden z nejjednodušších základních programů. Jedná se o pouhý výpočet Y pro směrnicovou rovnici, v tomto případě Y = 10 . X + 20.

Nastavení procesoru:

Procesor nastavíme na frekvenci 1 Hz (je možné přenastavit i na 1 kHz) a pozastavovací modul na přesnější odpočet (při nastavení na 1 kHz je vhodnější použít větší odpočet). Pro zobrazení výsledku můžeme připojit na externí výstup dekodér a sedmisegmentový displej. Adresu portu nastavíme na 0000 0000.

Provedení:

Procesor připojíme do napájení, ujistíme se, že je jeho chod zastaven a přepneme do programovacího režimu.

Postupně vepíšeme program do paměti podle tabulky níže. Obsah paměti můžeme poté znovu manuálně překontrolovat.

Procesor uvedeme do pracovního režimu a ujistíme se, že je aktuální pozice v programu vynulována.

Číslo dosazené do rovnice, je uloženo na 10. řádku programu. Ideálně ho nastavíme tak, aby nedošlo k přetečení, tedy menší než 24.

Spustíme program a sledujeme jeho průběh, dokud se chod sám nezastaví. Na displeji je zobrazen výsledek.

Program:

TABULKA Č.3: UKÁZKOVÝ PROGRAM 1 - JEDNODUCHÁ ROVNICE.

Slovně	**V binárním kódu**	**řádek**
Načti číslo na pozici (operace) 9 (hodnota)	0000 1001 (operace)	1
	0000 1001 (hodnota)	2
Vynásob číslem 10	1010 1000	3
	0000 1010	4
Přičti číslo 20	1000 1000	5
	0001 0100	6
Ulož na externí pozici 0	0010 0110	7
	0000 0000	8
Zastav	1111 1111	9
Číslo X	XXXX XXXX	10

6.2 Generátor náhodných čísel

Další velice jednoduchý program. Ukazuje možnost procesoru provádět programy v nekonečném cyklu.

Nastavení procesoru:

Procesor nastavíme na frekvenci 1 kHz a pozastavovací modul na větší odpočet. Výsledek stačí zobrazovat pomocí displeje na externím portu, jelikož nemá žádný reálný význam.

Provedení:

Procesor připojíme do napájení, ujistíme se, že je jeho chod zastaven a přepneme ho do programovacího režimu.

Postupně vepíšeme program do paměti podle tabulky níže. Obsah paměti můžeme poté znovu manuálně překontrolovat.

Procesor uvedeme do pracovního režimu a ujistíme se že je aktuální pozice v programu vynulována.

Spustíme program a sledujeme, jak se postupně generují náhodná čísla na displeji. Program bude neustále pokračovat v práci, dokud jej manuálně nezastavíme.

Program:

Tabulka č.4: Ukázkový program 2 - generátor náhodných čísel.

Slovně	**V binárním kódu**	**řádek**
Prázdný řádek pro skok	0000 0000	1
Vygeneruj náhodné číslo (operace) plné šířky (hodnota)	0001 1000 (operace)	2
	1111 1111 (hodnota)	3
Ulož na externí pozici 0	0010 0110	4
	0000 0000	5
Počkej 2 cykly	0000 0100	6
	0000 0010	7
Skoč na první řádek programu	0001 0000	8
	0000 0000	9

6.3 ARITMETICKÁ POSLOUPNOST

Třetí ukázkový program znázorňuje možnost postupného zpracovávání čísel a podmíněných skoků. Procesor bude postupně počítat každý člen aritmetické posloupnosti rovnice: $a_n = 5 + 5.n$

Nastavení procesoru:

Procesor nastavíme na frekvenci 1 kHz a pozastavovací modul na větší odpočet. Pro zobrazení výsledku můžeme připojit na externí výstup dekodér a sedmisegmentový displej. Adresu portu nastavíme na 0000 0000.

Provedení:

Procesor připojíme do napájení, ujistíme se, že je jeho chod zastaven a přepneme ho do programovacího režimu.

Postupně vepíšeme program do paměti podle tabulky níže. Obsah paměti můžeme poté znovu manuálně překontrolovat.

Procesor uvedeme do pracovního režimu a ujistíme se že je aktuální pozice v programu vynulována.

Spustíme program a sledujeme, jak se postupně zobrazuje každý další člen posloupnosti. Po čísle 50 se systém sám zastaví.

Program:

Tabulka č.5: Ukázkový program 3 - aritmetická posloupnost.

Slovně	**V binárním kódu**	**řádek**
Načti číslo (operace) 5 (hodnota)	0000 1000 (operace)	1
	0000 0101 (hodnota)	2
Prázdný řádek pro skok	0000 0000	3
Přičti číslo 5	1000 1000	4
	0000 0101	5
Ulož na externí pozici 0	0010 0110	6
	0000 0000	7
Počkej 4 cykly	0000 0100	8
	0000 0100	9
Je li číslo v akumulátoru menší než 50	1101 0000	10
	0011 0010	11
Skoč na třetí řádek programu	0001 0000	12
	0000 0010	13
Zastav	1111 1111	14

6.4 FIBONACCIHO POSLOUPNOST

Další z ukázkových programů je Fibonacciho posloupnost. Procesor postupně vypočítá a zobrazí prvních několik členů Fibonacciho posloupnosti. Tento program je ukázkou schopnosti procesoru pracovat s pamětí při výpočtech. Pro ukázku není program záměrně ošetřen proti přetečení.

Nastavení procesoru:

Procesor nastavíme na frekvenci 1 kHz a pozastavovací modul na větší odpočet. Pro zobrazení výsledku můžeme připojit na externí výstup dekodér a sedmisegmentový displej a nastavíme adresu portu na 0000 0000.

Provedení:

Procesor připojíme do napájení, ujistíme se, že je jeho chod zastaven a přepneme ho do programovacího režimu.

Postupně vepíšeme program do paměti podle tabulky níže. Obsah paměti můžeme poté znovu manuálně překontrolovat.

Procesor uvedeme do pracovního režimu a ujistíme se, že je aktuální pozice v programu vynulována.

Spustíme program a sledujeme zobrazování jednotlivých členů Fibonnaciho posloupnosti. Všímáme si přetečení osmibitového akumulátoru, které nastane po čísle 233.

Program:

Tabulka č.6: Ukázkový program 4 - Fibonacciho posloupnost.

Slovně	**V binárním kódu**	**řádek**
Načti číslo (operace) 0 (hodnota)	0000 1000 (operace)	1
	0000 0000 (hodnota)	2
Ulož na externí pozici 0	0010 0110	3
	0000 0000	4
Načti číslo 1	0000 1000	5
	0000 0001	6
Prázdný řádek pro skok	0000 0000	7
Ulož na externí pozici 0	0010 0110	8
	0000 0000	9
Počkej 4 cykly	0000 0100	10
	0000 0100	11
Ulož na pozici 21	0010 0100	12
	0001 0101	13
Přičti číslo na pozici 22	1000 1001	14
	0001 0110	15
Ulož na externí pozici 0	0010 0110	16
	0000 0000	17
Počkej 4 cykly	0000 0100	18
	0000 0100	19
Ulož na pozici 22	0010 0100	20
	0001 0110	21
Přičti číslo na pozici 21	1000 1001	22
	0001 0101	23
Skoč na čtvrtý řádek programu	0001 0000	24
	0000 0100	25
Použitelný řádek paměti	//// ////	26
Použitelný řádek paměti	//// ////	27

Alternativní program:

Tabulka č.7: Alternativní ukázkový program - Fibonacciho posloupnost.

Slovně	**V binárním kódu**	**řádek**
Načti číslo na pozici (operace) 21 (hodnota)	0000 1001 (operace)	1
	0001 0101 (hodnota)	2
Prázdný řádek pro skok	0000 0000	3
Ulož na externí pozici 0	0010 0110	4
	0000 0000	5
Počkej 4 cykly	0000 0100	6
	0000 0100	7
Ulož na pozici 21	0010 0100	8
	0001 0101	9
Přičti číslo na pozici 22	1000 1001	10
	0001 0110	11
Ulož na externí pozici 0	0010 0110	12
	0000 0000	13
Počkej 4 cykly	0000 0100	14
	0000 0100	15
Ulož na pozici 22	0010 0100	16
	0001 0110	17
Přičti číslo na pozici 21	1000 1001	18
	0001 0101	19
Skoč na čtvrtý řádek programu	0001 0000	20
	0000 0100	21
Číslo 0	0000 0000	22
Číslo 1	0000 0001	23

6.5 Collatzův problém

Poslední ukázkový program je Collatzův problém, známý také jako rovnice 3n + 1. Tento program začíná na specifikovaném čísle. Pokud je aktuální číslo sudé, vydělíme ho dvěmi, je-li číslo liché, vynásobíme ho třemi a přičteme jedničku. Rovnice se pak opakuje a postupně se vždy dopracujeme stejného výsledku.

Nastavení procesoru:

Procesor nastavíme na frekvenci 1 kHz a pozastavovací modul na větší odpočet. Pro zobrazení výsledku můžeme připojit na externí výstup dekodér a sedmisegmentový displej a nastavíme adresu portu na 0000 0000.

Provedení:

Procesor připojíme do napájení, ujistíme se, že je jeho chod zastaven a přepneme ho do programovacího režimu.

Postupně vepíšeme program do paměti podle tabulky níže. Obsah paměti můžeme poté znovu manuálně překontrolovat.

Procesor uvedeme do pracovního režimu a ujistíme se, že je aktuální pozice v programu vynulována.

Spustíme program a sledujeme postupně vypočítané hodnoty na displeji. Program se zastaví v jedničce.

Program:

Tabulka č.8: Ukázkový program 5 - Collatzův problém.

Slovně	**V binárním kódu**	**řádek**
Načti číslo (operace) 102 (hodnota)	0000 1000 (operace)	1
	0110 0110 (hodnota)	2
Prázdný řádek pro skok	0000 0000	3
Ulož na externí pozici 0	0010 0110	4
	0000 0000	5
Je-li číslo v akumulátoru menší než 2	1101 0000	6
	0000 0010	7
Zastav	1111 1111	8
Je-li číslo v akumulátoru sudé	1100 0000	9
	0000 0010	10
Skoč na devatenáctý řádek programu	0001 0000	11
	0001 0011	12
Vynásob třemi	1010 1000	13
	0000 0011	14
Přičti číslo 1	1000 1000	15
	0000 0001	16
Skoč na třetí řádek programu	0001 0000	17
	0000 0010	18
Prázdný řádek pro skok	0000 0000	19
Vyděl dvěma	1011 1000	20
	0000 0010	21
Skoč na třetí řádek programu	0001 0000	22
	0000 0010	23

Závěr

Při návrhu architektury procesoru jsem vycházel ze svých úspěšných pokusů o návrh plně funkčního počítače v mých 15 letech. Na nových úpravách návrhu jsem usilovně pracoval několik měsíců a jednoznačně se jednalo o nejsložitější část projektu. Úspěšně jsem ji dokončil a prakticky potvrdil její funkčnost.

Zhotovení logického návrhu bylo natolik obsáhlé a komplexní, že se musí počítat s určitou procentuální chybou. I tak mi tato část projektu zabrala snad nejkratší dobu. V celku bych řekl že se mi návrh podařil, až na pár nezávažných vyjímek.

Elektrické zapojení jsem dle mého názoru navrhl přesně a čistě, s důrazem na přehlednost a symetrii. Na návrhu každé desky plošných spojů jsem pracoval několik hodin až dnů. S výsledkem svých návrhů jsem nadmíru spokojen.

Konstrukce pro mě byla další nesmírně náročnou částí, která zabrala nespočet dnů usilovné práce. Musel jsem manuálně připájet přes deset tisíc kontaktů na téměř tisíci integrovaných obvodech a ostatních součástkách. Vzhledem k ručnímu pájení v domácích podmínkách, nebylo v mých silách zaručit stoprocentní úspěšnost každého spoje. S touto částí jsem také vysoce spokojen.

Oživení systému se mi podařilo s nečekanou úspěšností. Téměř každou překážku ve funkci jsem svižně v konstrukci vyhledal a následně opravil. Převážně se jednalo o nedokonalé propojení jednotlivých spojů.

Funkci procesoru jsem pečlivě prověřil. Je však téměř nemožné otestovat každý možný průběh. Systém je nadále testován jakýmkoli dalším použitím.

Na výsledek své práce jsem náležitě hrdý, jelikož se mi podařilo navrhnout a sestrojit celý funkční procesor dle vlastní architektury.

Po ohlédnutí zpět na průběh konstrukce bych volil nějaké jiné postupy. Například strojové osazení desek plošných spojů.

Díky modulárnímu systému nejsem omezen aktuální konstrukcí projektu a mohu jej dále rozšiřovat a upravovat novými moduly a vylepšovat tak schopnosti mého procesoru.

Seznam použité literatury

[1] Wikipedie Otevřená encyklopedie, Seznam integrovaných obvodů série 7400 [online].

Dostupné z:
https://en.wikipedia.org/wiki/List_of_7400-series_integrated_circuits

[2] Wikipedie Otevřená encyklopedie, Instrukční cyklus [online].

Dostupné z:
https://en.wikipedia.org/wiki/Instruction_cycle

[3] Wikipedie Otevřená encyklopedie, Kruhový oscilátor [online].

Dostupné z:
https://en.wikipedia.org/wiki/Ring_oscillator

[1] David MATOUŠEK. Číslicová technika - základy konstruktérské praxe. 1. vyd. Praha: BEN - technická literatura, 2001. ISBN 80-7300-025-3.

Seznam obrázků a tabulek

PŘÍLOHY

Příloha č. 1: Konektorová deska.

Příloha č. 2: Modul cyklů.

Příloha č. 3: Modul programu.

Příloha č. 4: Modul paměti.

Příloha č. 5: Modul dekódování.

Příloha č. 6: Modul akumulátoru.

Příloha č. 7: Modul logických operací a náhodné generace.

Příloha č. 8: Modul aritmetických operací a binárních posuvů.

Příloha č. 9: Modul pozastavení programu.

Příloha č. 10: Modul podmíněných skoků.

Příloha č. 11: Modul programování, vstupů a výstupů.

Příloha č. 12: Logické schéma - konektorová deska.

Příloha č. 13: Logické schéma - modul cyklů.

Příloha č. 14: Logické schéma - modul programu.

Příloha č. 15: Logické schéma - modul paměti.

Příloha č. 16: Logické schéma - modul dekódování.

Příloha č. 17: Logické schéma - modul akumulátoru.

Příloha č. 18: Logické schéma - modul logických operací a náhodné generace.

Příloha č. 19: Logické schéma - modul aritmetických operací a binárních posuvů.

Příloha č. 20: Logické schéma - modul pozastavení programu.

Příloha č. 21: Logické schéma - modul podmíněných skoků.

Příloha č. 22: Logické schéma - modul programování, vstupů a výstupů.

Příloha č. 23: Návrh DPS - konektorová deska.

Příloha č. 24: Návrh DPS - modul cyklů.

Příloha č. 25: Návrh DPS - modul programu.

Příloha č. 26: Návrh DPS - modul paměti.

Příloha č. 27: Návrh DPS - modul dekódování.

Příloha č. 28: Návrh DPS - modul akumulátoru.

Příloha č. 29: Návrh DPS - modul logických operací a náhodné generace.

Příloha č. 30: Návrh DPS - modul aritmetických operací a binárních posuvů.

Příloha č. 31: Návrh DPS - modul pozastavení programu.

Příloha č. 32: Návrh DPS - modul podmíněných skoků.

Příloha č. 33: Návrh DPS - modul programování, vstupů a výstupů.