Vysoká škola báňská – Technická univerzita Ostrava

MIKROPOČÍTAČOVÉ ŘÍDICÍ SYSTÉMY I

učební text

Petr Palacký

Ostrava 2007

Recenze: Ing. Martin Kuchař, Ph.D.

Název: Mikropočítačové řídicí systémy I
Autor: Petr Palacký
Vydání: první, 2007
Počet stran: 123
Vydavatel a tisk: Ediční středisko VŠB – TUO

Studijní materiály pro studijní obor Elektronika fakulty Elektrotechniky a informatiky
Jazyková korektura: nebyla provedena.

Určeno pro projekt:
Operační program Rozvoj lidských zdrojů
Název: E-learningové prvky pro podporu výuky odborných a technických předmětů
Číslo: CZ.O4.01.3/3.2.15.2/0326
Realizace: VŠB – Technická univerzita Ostrava
Projekt je spolufinancován z prostředků ESF a státního rozpočtu ČR

ISBN 978-80-248-1494-0

POKYNY KE STUDIU

Mikropočítačové řídicí systémy I

Pro předmět Mikropočítačové řídicí systémy I 5. semestru oboru Elektronika jste obdrželi studijní balík obsahující

- integrované skriptum pro distanční studium obsahující i pokyny ke studiu
- CD-ROM s doplňkovými animacemi vybraných částí kapitol
- harmonogram průběhu semestru a rozvrh prezenční části
- rozdělení studentů do skupin k jednotlivým tutorům a kontakty na tutory
- kontakt na studijní oddělení

Prerekvizity

Pro studium tohoto předmětu se předpokládá absolvování předmětu Elektronika

Cílem předmětu

je seznámení se základními pojmy Mikroprocesor, mikropočítač, řídicí systém apod. Po prostudování modulu by měl student být schopen provést analýzu a syntézu mikropočítačových řídicích systémů.

Pro koho je předmět určen

Modul je zařazen do bakalářského studia oborů Elektrické stroje přístroje a pohony a Aplikovaná a komerční elektronika studijního programu Elektrotechnika Elektrotechnika, sdělovací a výpočetní technika , ale může jej studovat i zájemce z kteréhokoliv jiného oboru, pokud splňuje požadované prerekvizity.

Skriptum se dělí na části, kapitoly, které odpovídají logickému dělení studované látky, ale nejsou stejně obsáhlé. Předpokládaná doba ke studiu kapitoly se může výrazně lišit, proto jsou velké kapitoly děleny dále na číslované podkapitoly a těm odpovídá níže popsaná struktura.

Při studiu každé kapitoly doporučujeme následující postup:

Čas ke studiu: xx hodin

Na úvod kapitoly je uveden **čas** potřebný k prostudování látky. Čas je orientační a může vám sloužit jako hrubé vodítko pro rozvržení studia celého předmětu či kapitoly. Někomu se čas může zdát příliš dlouhý, někomu naopak. Jsou studenti, kteří se s touto problematikou ještě nikdy nesetkali a naopak takoví, kteří již v tomto oboru mají bohaté zkušenosti.

Cíl: Po prostudování tohoto odstavce budete umět

- popsat ...
- definovat ...
- vyřešit ...

Ihned potom jsou uvedeny cíle, kterých máte dosáhnout po prostudování této kapitoly – konkrétní dovednosti, znalosti.

Výklad

Následuje vlastní výklad studované látky, zavedení nových pojmů, jejich vysvětlení, vše doprovázeno obrázky, tabulkami, řešenými příklady, odkazy na animace.

Shrnutí pojmů 1

Na závěr kapitoly jsou zopakovány hlavní pojmy, které si v ní máte osvojit. Pokud některému z nich ještě nerozumíte, vraťte se k nim ještě jednou.

Otázky 1

Pro ověření, že jste dobře a úplně látku kapitoly zvládli, máte k dispozici několik teoretických otázek.

Úlohy k řešení

Protože většina teoretických pojmů tohoto předmětu má bezprostřední význam a využití v databázové praxi, jsou Vám nakonec předkládány i praktické úlohy k řešení. V nich je hlavní význam předmětu a schopnost aplikovat čerstvě nabyté znalosti při řešení reálných situací hlavním cílem předmětu.

Klíč k řešení

Výsledky zadaných příkladů i teoretických otázek výše jsou uvedeny v závěru učebnice v Klíči k řešení. Používejte je až po vlastním vyřešení úloh, jen tak si samokontrolou ověříte, že jste obsah kapitoly skutečně úplně zvládli.

Úspěšné a příjemné studium s touto učebnicí Vám přeje autor výukového materiálu

Petr Palacký

Obsah

1 ZÁKLADNÍ POJMY ČÍSLICOVÉ TECHNIKY 1
Výklad 1
1.1 Logické funkce a jejich zápis 1
1.2 Minimalizace prostřednictvím Karnaughových map 3
Shrnutí pojmů 1 5
Otázky 1 5
2 REALIZACE LOGICKÝCH ČLENŮ A JEJICH VLASTNOSTI 6
Výklad 6
2.1 Šumová imunita 9
2.2 Zásady práce s obvody TTL 10
2.2.1 Ošetření nevyužitých vstupů 10
2.2.2 Připojování vstupů nevyužitých logických obvodů. 10
2.2.3 Úpravy vstupních logických signálů 10
2.2.4 Tvarovací obvody 11
2.2.5 Výstupy logických členů 12
2.2.6 Rozvod logických úrovní 13
2.2.7 Rozvod společného vodiče 14
2.2.8 Buzení dlouhého vedení 15
Shrnutí pojmů 2 17
Otázky 2 17
Úlohy k řešení 2 18
3 POJEM MIKROPOČÍTAČ A MIKROPROCESOR 23
Výklad 23
3.1 Základní struktura mikropočítače 23
3.2 Struktura mikroprocesoru 24
3.2.1 Řadič 25
3.2.2 Soubor registrů 25
3.2.3 Aritmeticko-logická část 26
3.3 Základní činnost mikroprocesoru 27
3.3.1 Časování mikroprocesoru 27
3.3.2 Instrukční soubor 27
3.3.3 Způsoby adresování 29
3.4 Přerušení 32
Shrnutí pojmů 3 33
Otázky 3 33
Úlohy k řešení 3 33
4 SBĚRNICE 36
Výklad 36
4.1 Adresová sběrnice 36
4.2 Řídicí sběrnice 37
4.3 Datová sběrnice 37
Shrnutí pojmů 4 41
Otázky 4 41
Úlohy k řešení 4 41
5 ADRESNÍ DEKODÉRY 43
Výklad 43
5.1 Dekodér pro úplné dekódování adresy 44
5.2 Adresový dekodér s neúplným dekódováním adres 45
5.3 Adresový dekodér s lineárním přiřazením adresy 46
Shrnutí pojmů 5 49

Otázky 5 49
Úlohy k řešení 5 49
6 VZÁJEMNÝ STYK MIKROPOČÍTAČE S PERIFERIEMI 52
Výklad 52
6.1 Řízení komunikace 52
6.2 Programové řízení komunikace 53
6.3 Řízení komunikace pomocí přerušení 54
Shrnutí pojmů 6 56
Otázky 6 56
Úlohy k řešení 6 56
7 DRUHY PŘENOSU DAT 58
Výklad 58
7.1 Obvody vstupu a výstupu 60
7.2 Chyby přenosu a jejich redukce 61
7.3 Sériová rozhraní 62
7.3.1 Rozhraní RS232 62
7.3.2 Rozhraní RS422 63
7.3.3 Rozhraní RS485 64
7.3.4 Proudová smyčka 64
7.3.5 Rozhraní USB 65
7.3.6 Rozhraní SPI 67
7.3.7 Rozhraní I^2C 68
Shrnutí pojmů 7 71
Otázky 7 71
Úlohy k řešení 7 71
8 POLOVODIČOVÉ PAMĚTI 76
Výklad 76
8.1 Rozdělení pamětí podle přístupu 76
8.2 Rozdělení podle možnosti zápisu/čtení 77
8.3 Rozdělení pamětí podle principu elementární paměťové buňky 79
Shrnutí pojmů 8 80
Otázky 8 80
Úlohy k řešení 8 80
9 STYK MIKROPOČÍTAČE S ANALOGOVÝM PROSTŘEDÍM 82
Výklad 82
9.1 Analogový výstup mikropočítače 83
9.1.1 Číslicově analogový (D/A) převod 84
9.2 Analogový vstup mikropočítače 87
9.2.1 Analogově číslicový (A/D) převod. 89
9.3 Kódy pro A/D a D/A převody 92
9.3.1 Vyjádření čísel v číslicových systémech 92
9.3.2 Vyjádření znaménka 94
Shrnutí pojmů 9 95
Otázky 9 95
Úlohy k řešení 96
10 MIKROPROCESORY, MIKROPOČÍTAČE A MIKROPOČÍTAČOVÉ SYSTÉMY 100
Výklad 100
10.1 Rozdělení procesorů 100
10.1.1 První způsob dělení: 101
10.1.2 Druhý způsob dělení 101
10.1.3 Dělení podle architektury 103

Shrnutí pojmů 10 106
Otázky 10 106
Úlohy k řešení 10 106
Klíč k řešení 107

1. ZÁKLADNÍ POJMY ČÍSLICOVÉ TECHNIKY

Čas ke studiu: 1,5 hodin

Cíl Po prostudování tohoto odstavce budete umět

- definovat logickou funkci
- princip minimalizace logických funkcí

Výklad

Signály, které se vyskytují v číslicové technice, mohou nabývat pouze dvou možných hodnot, které označujeme jako logická jednička - log. 1, a logická nula - log. 0. Uvedené signály budou popisovány pomocí dvouhodnotových veličin.

Dvouhodnotové veličiny se v technice digitálních integrovaných obvodů zobrazují nejčastěji těmito způsoby:

- zobrazením pomocí úrovně fyzikální veličiny (napětí, proudu) - úroveň H (vyšší hodnota), L (nižší hodnota);
- zobrazením pomocí změny takové veličiny.

Digitální systémy se dělí na dvě velké skupiny:

- systémy kombinační, u nichž hodnoty výstupních veličin závisejí jen na okamžitém stavu vstupních veličin,
- systémy sekvenční, kde hodnoty výstupních veličin závisejí i na předchozím stavu systému, tyto systémy tedy obsahují paměťový prvek.

1.1 Logické funkce a jejich zápis

Základním pojmem při úvahách o kombinačních systémech představuje pojem kombinační logická funkce. *Kombinační logická funkce* je pravidlo přiřazující každé kombinaci hodnot **0** a **1** přiřazených vstupním proměnným z definičního oboru funkce jedinou hodnotu výstupní proměnné. Pro daný počet vstupních proměnných je těchto funkcí konečný počet. Kombinační logické funkce mohou být úplně nebo neúplně určené.

Úplně určená kombinační logická funkce je taková funkce, jejíž definiční obor zahrnuje všechny kombinace vstupních proměnných.

U *neúplně určené kombinační logické funkce* její definiční obor nezahrnuje některé tyto kombinace. Kombinací se zde rozumí kombinace hodnot **0** a **1** přiřazených jednotlivým vstupním proměnným. Úplně určeným funkcím se někdy říká úplné funkce, funkcím neúplně určeným pak neúplné funkce.

V technice číslicových obvodů se využívá tzv. Booleovy algebry, kterou můžeme chápat jako algebru na množině dvou prvků B = { 0; 1 } se třemi operacemi: logický součin, logický součet a negace. Booleovské proměnné (logické proměnné), vyjádřené např. x,y, mohou nabývat pouze hodnoty některého prvku z množiny B = { 0; 1 }. **Booleovská funkce** (logická funkce) n proměnných je definována jako jednoznačná funkce booleovských proměnných $x_1, x_2, \ldots x_n$ na množině B = { 0; 1 }.

Logická funkce může být zadána pravdivostní tabulkou, přičemž každou booleovskou funkcí je možné vyjádřit v kanonickém tvaru jako součet součinů (mintermů) – úplná disjunktivní normální forma (ÚDNF):

$$f(x_1,x_2...x_n)=\overline{x_1}\,\overline{x_2}...\overline{x_{n-1}}\,\overline{x_n}+\overline{x_1}\,\overline{x_2}...\overline{x_{n-1}}x_n+x_1x_2...x_{n-1}x_n$$

nebo v kanonickém tvaru jako součin součtů (maxtermů) – úplná konjunktivní normální forma (ÚKNF):

$$f(x_1,x_2...x_n)=(x_1+x_2+...+x_{n-1}+x_n)(x_1+x_2+...+x_{n-1}+\overline{x_n})(\overline{x_1}+\overline{x_2}+...+\overline{x_{n-1}}+\overline{x_n})$$

V praxi mohou být logické funkce zadány dále stavovými indexy S. Stavový index je dekadické vyjádření dvojkového čísla pro příslušnou vstupní proměnnou.

Např. kanonické vyjádření logické funkce pomocí ÚDNF, resp. ÚKNF:

$$f(x_1,x_2,x_3)=\overline{x_1}\,\overline{x_2}\,\overline{x_3}+\overline{x_1}\,\overline{x_2}x_3+x_1\overline{x_2}\,\overline{x_3}+x_1\overline{x_2}x_3+x_1x_2x_3$$

$$f(x_1,x_2,x_3)=(x_1+\overline{x_2}+x_3)(\overline{x_1}+\overline{x_2}+x_3)(x_1+\overline{x_2}+\overline{x_3})$$

lze zapsat pravdivostní tabulkou (tab. 1.1):

S	**X_3**	**X_2**	**X_1**	**$f(x_1,x_2,x_3)$**
0	0	0	0	1
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

Tab. 1.1 Pravdivostní tabulka úplně zadané logické funkce

a pomocí stavových indexů pro ÚDNF:

$$f(x_1,x_2,x_3)=\Sigma(0,1,4,5,7)$$

Píší se indexy S, pro které nabývá logická funkce hodnot 1, resp. pro ÚKNF

$$f(x_1,x_2,x_3)=\Pi(2,3,6),$$

které nabývá logická funkce hodnot 0.

V případě, že funkční hodnota logické funkce pro určitou kombinaci vstupních logických proměnných není jednoznačně definována, tzn. může nabývat obou hodnot z množiny [0,1], hovoříme o tzv. neúplně zadané logické funkci (viz. výše a tab. 1.1). Zápis neúplně zadané logické funkce pomocí

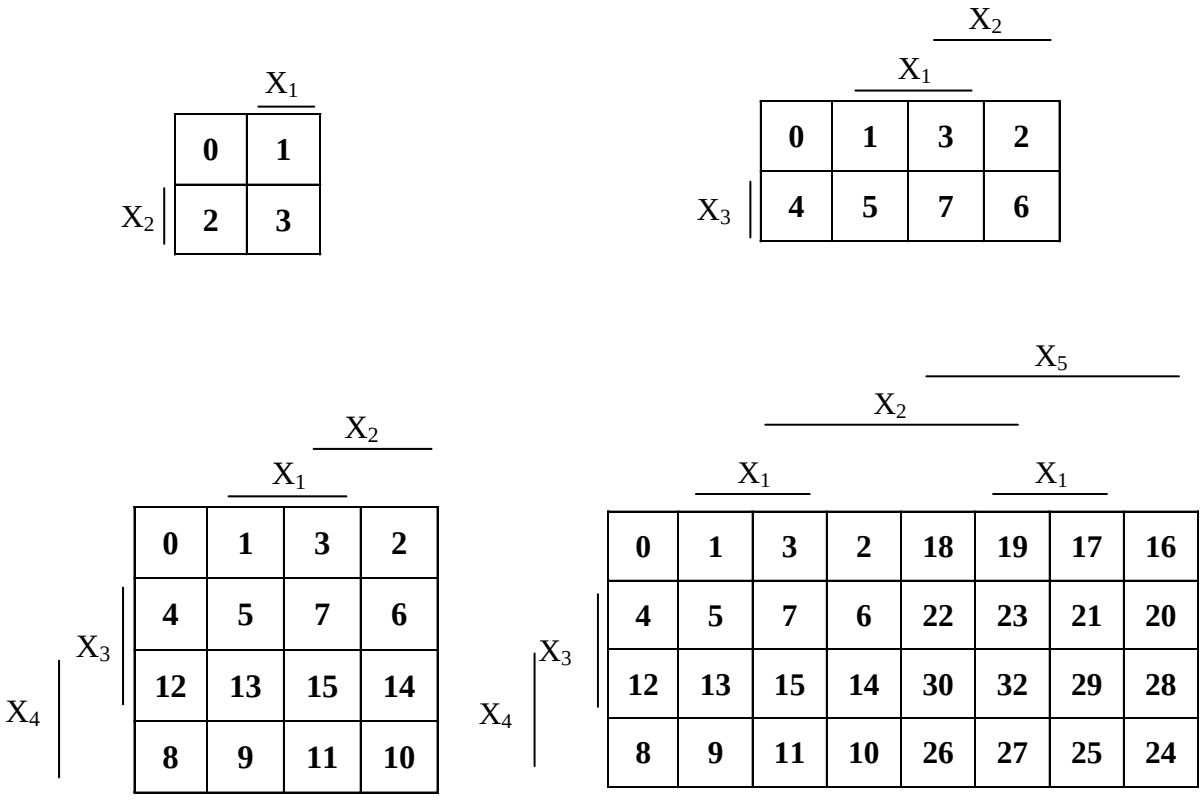
stavových indexů provedeme tak, že stavové indexy odpovídající neurčeným stavům uvedeme v závorce.

Logický obvod nemusí být popsán jedinou logickou funkcí. Při technické realizaci logické funkce však žádáme co nejmenší počet logických členů, a proto se snažíme nalézt vyjádření logické funkce o nejmenším počtu logických součtů, součinů, negací a logických proměnných. Takovému postupu říkáme *minimalizace logické funkce.* Výsledkem je minimální disjunktivní normální forma (MDNF), resp. minimální konjunktivní normální forma (MKNF). Obecný postup, jímž se hledají minimální normální formy, nazýváme minimalizací logické funkce. Řešení vhodné pro minimalizaci logických funkcí s menším počtem nezávislých logických proměnných můžeme provádět pomocí algebraických úprav s využitím vět Booleovy algebry nebo pomocí tzv. Karnaughových map.

1.2 Minimalizace prostřednictvím Karnaughových map

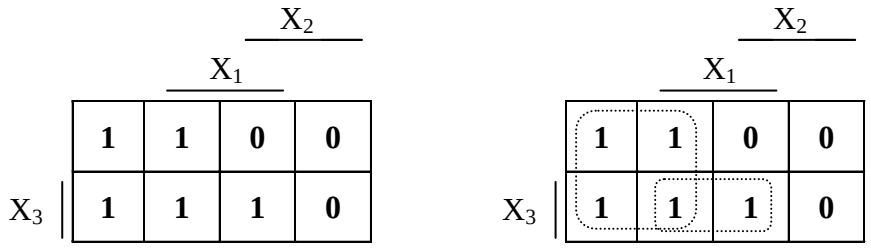
Karnaughovy mapy vycházejí z Vennova diagramu, známého z teorie množin, vhodně upraveného do sítě polí. Karnaughova mapa je uspořádána do čtverce nebo obdélníku, přičemž má tolik polí, kolik je kombinací vstupních nezávisle proměnných. Každé pole je možné popsat jeho stavovým indexem. Sousední pole se liší pouze v jedné proměnné. Příklady Karnaughových map pro 2 až 5 proměnných s označením polí stavovými indexy ukazuje obr. 1.1. Vstupní proměnné jsou vyznačeny pruhem nad příslušným řádkem či sloupcem mapy, přičemž pruh označuje část mapy, v níž uvedená proměnná nabývá hodnoty 1. V ostatní části mapy nabývá tato proměnná hodnoty 0.

Funkční hodnoty logické funkce se zapíší do mapy tak, že pro každou kombinaci vstupních proměnných se do odpovídajícího pole v mapě zapíše 1 (je-li funkční hodnota rovna 1), 0 (je-li funkční hodnota neurčena). Sousední pole lze spojovat do větších útvarů tzv. smyček. Zjednodušování logické funkce v případě hledání minimální disjunktivní normální formy MDNF se potom provádí tak, že je nutné pokrýt všechny jedničky soustavou co nejmenšího počtu co největších smyček, přičemž smyčky mohou (ale nemusí) zahrnovat případné neurčené stavy. Mezi sousední pole se počítají taky pole v rozích mapy. V případě hledání minimální konjunktivní normální formy MKNF pokrýváme smyčkami všechny nuly, případně neurčené stavy.



Obr. 1.1 Karnaughovy mapy pro 2 až 5 proměnných

Jako příklad si uvedeme zjednodušení předcházející logické funkce zadané tabulkou 1.1. Zápis logické funkce $f(x_1,x_2,x_3)=\Sigma(0,1,4,5,7)$ do Karnaughovy mapy bude tedy vypadat následovně:



Z mapy lze vidět, že k pokrytí všech jedniček je zapotřebí dvou smyček. Ve smyčce ze čtyř polí zůstává stále stejná proměnná $\overline{x_2}$ v obou sloupcích smyčky, kdežto proměnná x_1 se mění, což je totéž, jako bychom při algebraické úpravě provedli spojení $(x_1+\overline{x_1})=1$. Totéž platí v řádcích smyčky pro proměnnou x_3, neboť se také mění - $(x_3+\overline{x_3})=1$. Výsledek ze smyčky je tedy $\overline{x_2}$. Obdobně lze provést rozbor pro malou smyčku, pro kterou platí výsledek $x_1\cdot x_3$. Minimální

disjunktivní normální forma má potom tolik implikantů (případně degenerovaných implikantů), kolika smyček bylo zapotřebí k pokrytí všech jedniček v mapě.

Výsledná funkce je tedy $f(x_1, x_2, x_3) = x_2 + x_1 \cdot x_3$

X_3 \ X_1 \ X_2				
	1	**1**	**0**	**0**
	1	**1**	**1**	**0**

V případě určení minimální konjunktivni normální formy MKNF pro uvedený příklad jsou k pokrytí všech nul nutné dvě smyčky. V první smyčce se mění proměnná x_3. Smyčka pak bude popsána součtem negací proměnných, které se ve smyčce nemění. Pro druhou smyčku můžeme obdobně napsat. Výsledná funkce je tedy $f(x_1, x_2, x_3) = (x_1 + \overline{x_2})\ (\overline{x_2} + x_3)$

Shrnutí pojmů 1

Klíčová slova:
Logická funkce, minimalizace logické funkce, Karnaughova mapa

Otázky 1

1. Vysvětlete pojem logická funkce
2. Popište možné zápisy logické funkce
3. Jakými způsoby provádíme minimalizaci logické funkce
4. Co je to Karnaughova mapa
5. Vysvětlete princip minimalizace logických funkcí s pomocí Karnaughových map

2. REALIZACE LOGICKÝCH ČLENŮ A JEJICH VLASTNOSTI

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- definovat zásady práce s obvody TTL
- popsat princip logických obvodů
- vyřešit propojování logických obvodů

Výklad

Při návrhu logických obvodů většinou stačí, známe-li jejich logické funkce. Jestliže však chceme navržený logický obvod začlenit do obvodové struktury složitějšího systému, je pak již nutné znát i jeho elektrické vlastnosti a parametry.

S rozvojem automatického řízení výrobních procesů a tím i nutným rozvojem mikropočítačové techniky byl vyvozován neustálý tlak na zvětšování počtu logických funkcí na jednotku objemu při stoupající spolehlivosti. Technologie výroby křemíkových polovodičů se tak zdokonalila, že na jednom substrátu bylo možné vytvořit velké množství polovodičových prvků. Postupně byla vyvinuta řada logických stavebnic, které realizovaly základní funkce. Největší rozšíření a uplatnění dosáhla tranzistor-tranzistorová logika TTL (tranzistor na vstupu, tranzistor na výstupu) a její modifikace. Jediným vážným konkurentem bipolárních obvodů TTL jsou unipolární obvody CMOS, které mají ve statickém provozu zanedbatelnou spotřebu. S rostoucím kmitočtem se spotřeba zvyšuje a při kmitočtech kolem 1Mhz je spotřeba srovnatelná s obvody LSTTL.

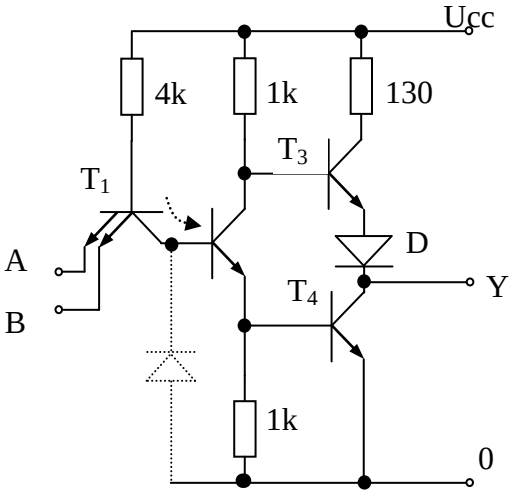
Jednou z nejrozšířenějších řad logických obvodů je notoricky známá řada 74… Podle výrobce se pak liší označením před tímto číslem. Osvojování této řady šlo tak daleko, že se jednotlivé řady různých výrobců mezi sebou neliší ani parametry ani polohou vývodů, ale dokonce ani topologií substrátu. Výhodou je pak naprostá zaměnitelnost a rovnocennost obvodů od jednotlivých výrobců.

Celá řada se vyznačuje až na nepatrné výjimky jednotným obvodovým řešením vycházejícím z hradla NAND. Zapojení hradla je zachyceno na obr. 2.1 a je dobré pro další návrhy zapojení znát jeho obvodovou činnost. Znalost funkce je pak velmi užitečná i v případech diagnostiky poruch v logických systémech. Logický návrh se musí podřizovat vlastnostem integrovaných obvodů. Opačná cesta vede k okamžitému či pozdějšímu neúspěchu.

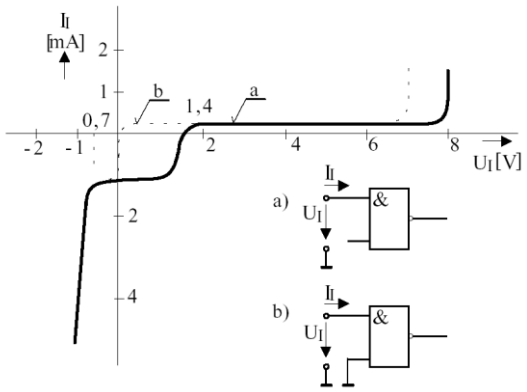
Bude-li alespoň na jednom ze vstupů A, B úroveň L, pak emitorovým přechodem tranzistoru T_1 bude protékat proud v naznačeném směru ze vstupu do vnějšího obvodu. Tranzistor T_1 bude tedy sepnut, přičemž na bázi tranzistoru T_2 bude malé napětí (U_{CEST1}) a tranzistor T2 bude uzavřen. Tranzistor T3 je zapojen jako emitorový sledovač, takže na vstupu získáme napětí menší o úbytek na přechodu báze-emitor T_3 a propustně polované diodě D_1; tedy úroveň H. Zvyšováním vstupního napětí se od hodnoty asi 0,7 V začne otevírat tranzistor T_2, napětí na jeho kolektoru začne klesat. Při hodnotě vstupního napětí cca 1,4 V se začíná otevírat i tranzistor T_4, napětí na jeho kolektoru začíná rychle klesat. Rezistor R_3 je přemostěn malým odporem přechodu báze-emitor tranzistoru T_4, přičemž zesílení stupně s tranzistorem T_2 prudce stoupá a urychluje se pokles napětí na bázi tranzistoru T_3. Tranzistor T_3 se uzavírá. Dalším zvyšováním vstupního napětí se uzavírá přechod emitor-báze tranzistoru T_1. Bází tranzistoru T_1 protéká proud, jehož velikost je cca (U_{CC} - U_{BC1} - U_{BC2} - U_{BE4})/R_1 a uzavírá se přes

propustně polovaný přechod báze-kolektor T_1 a emitorové přechody T2 a T4. Na výstupu je napětí UCES sepnutého tranzistoru T_4 a do vstupu členu teče malý proud. Obr. 2.4. ukazuje převodní charakteristiku členu NAND TTL.

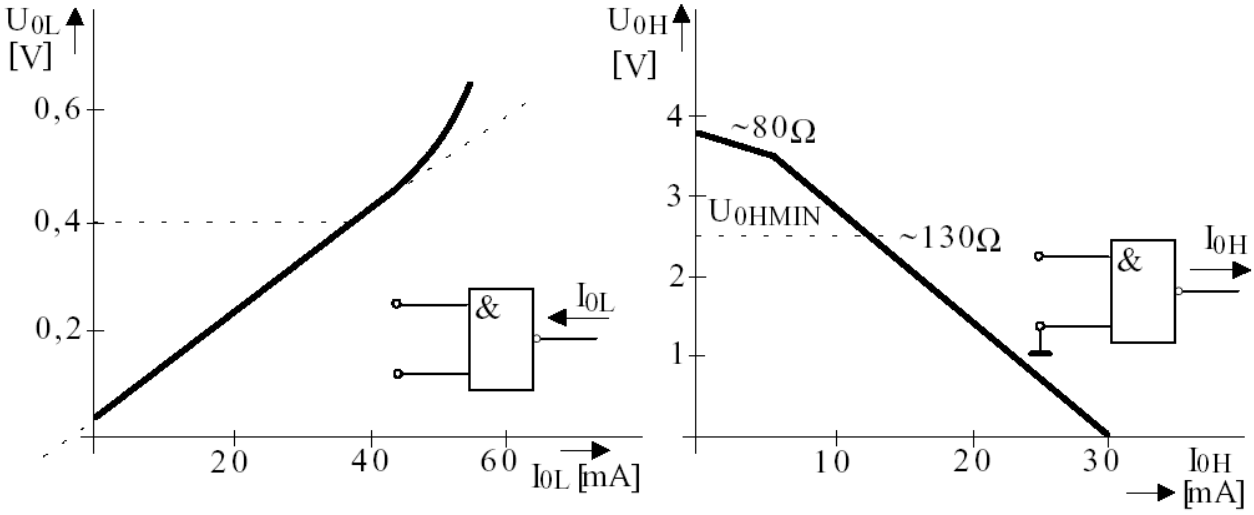
Přivedeme-li na vstup záporné napětí, pak při napětí UI- -0,7 V se otevírá parazitní dioda kolektor T_1 - podložka (čárkovaně). Jestliže proud touto diodou nebude omezen na hodnotu maximálně 15 mA, dojde ke zničení obvodu. Přivedením vstupního napětí většího než 8 V dochází k průrazu přechodu báze-emitor vstupního tranzistoru a v případě, že proud tranzistorem nebude omezen (cca 1 mA), dojde k jeho poškození. Obr. 2.2 ukazuje vstupní charakteristiku členu NAND.



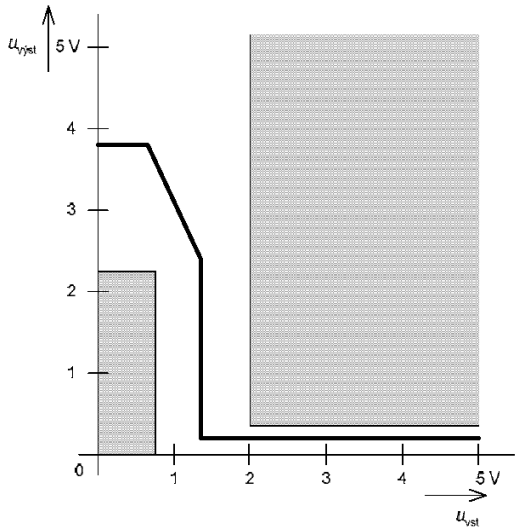
Obr. 2.1 Zapojení hradla NAND



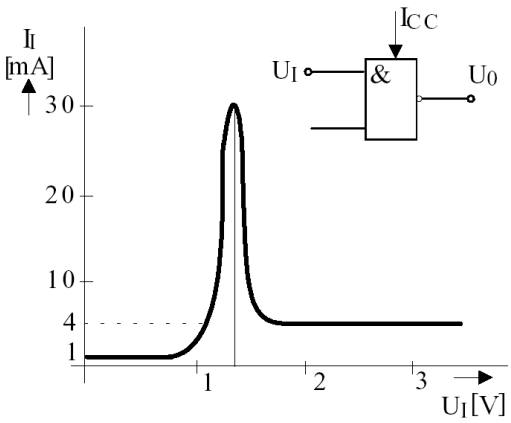
Obr. 2.2 Vstupní charakteristika hradla



Obr. 2.3 Výstupní charakteristiky



Obr. 2.4 Přechodová charakteristika hradla



Obr. 2.5 Odběrová charakteristika hradla

Při přechodu členu TTL z úrovně L do úrovně H jsou tranzistory T_3, T_4 na okamžik oba otevřeny, přičemž hodnota je omezena na hodnotu asi 30 mA odporem R4. Odběrová charakteristika členu NAND TTL (obr. 2.5) zahrnuje uvedený jev.

Bude-li změna na vstupu členu NAND TTL dostatečně strmá, pak odběrová špička bude trvat několik ns a příkon členu nebude znatelně ovlivněn. Ovšem vlivem rychlých změn proudu, zejména mění-li se stav výstupu mnoha členů současně, mohou vzniknout na nevhodně navrženém napájecím vedení napěťové špičky (desetiny až jednotky voltů), které mohou znemožnit správnou činnost obvodů, případně je i zničit. Vzhledem k velice strmým změnám proudu (spektrum kmitočtu stovky a více MHz) je nutné uvažovat napájecí vedení za vedení s charakteristikou impedancí. Je mylné uvažovat při složitých aplikacích jen ohmický odpor napájecího vedení, který bývá většinou zanedbatelný. Určitého zlepšení je možné dosáhnout zapojením blokovacích kondenzátorů ve vhodném místě napájecího vedení, čímž se sníží jeho charakteristická impedance.

Při návrhu je tedy nutné mít na zřeteli několik okolností:

- Při logické nule na vstupech teče proud z hradla ven.
- Výstupní odpor při logické jedničce je odlišný než výstupní odpor při logické nule a oba jsou dány parametry vnitřních součástek.
- Jestliže se dostane na výstup záporné napětí, sepne se tranzistor T3

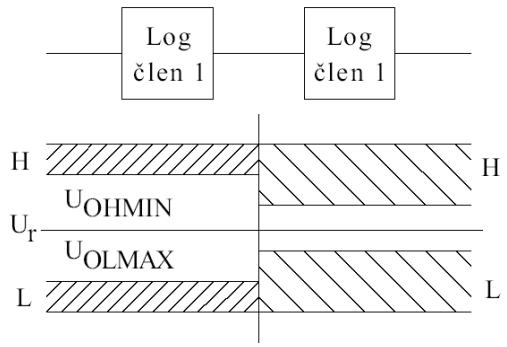
Standardní řada obvodů TTL, která není nijak zvlášť označována, byla a je modifikována, aby bylo vyhověno různým požadavkům. Tak vznikla rychlá řada H, (74H04) a řada s nízkým příkonem „L“ 74L04. Změny bylo dosaženo v podstatě jen změnou vnitřních odporů. U řady „L“ zvýšením jejich hodnot, což ale vedlo ke snížení rychlosti, a u řady „H“ snížením jejich hodnot. Dalšího zlepšení bylo dosaženo použitím Schottkyho diod ve struktuře některých tranzistorů, které zabraňují jejich saturaci (řada „S“ – rychlá, „LS“ – rychlá s nízkým příkonem). Dalším vylepšením vznikla řada „AS“ – rychlá, „ALS„ rychlá s nízkým příkonem. Poslední zmiňované řady využívají jednoemitorového vstupního tranzistoru. Poměrně novou řadou je řada „F“ s novým obvodovým řešením a menší teplotní a napěťovou závislosti. Je rychlejší než řada „S“ při 3 až 4 x menším příkonu.

Typ logiky	Napájecí napětí [V]	Logický zisk [-]	Zpoždění hradla [ns]	Max. frekvence [MHz]	Příkon hradla [nW]
CMOS 4000	3 až 15	50	40 až 20	8 až 16	10
CMOS 74C	3 až 15	50	50 až 30	3 až 8	10 až 30
CMOS 74SC	3 až 7	50	36	30	10
CMOS 74HC	2 až 6	10	6	60	10
CMOS 74HCT	5	10	6	60	10^3
CMOS 74HCU	2 až 6	10	6	60	10^3
TTL 74	5±5%	10	10	35	10^7
TTL 74L	5±5%	10	33	3	10^6
TTL 74S	5±5%	10	3	125	$1{,}9*10^7$
TTL 74LS	5±5%	20	10	45	$2*10^6$
TTL 74AS	5±5%	20	1,5	200	$2{,}2*10^7$
TTL 74ALS	5±5%	20	4	50	10^6

Tab. 2.1 Shrnutí některých parametrů obvodů

2.1 Šumová imunita

Z charakteristik se rovněž může odvodit tzv. šumová odolnost. Je to úroveň napětí, která ještě může vniknout do spojů mezi integrovanými obvody, aniž by došlo k reakci na tento parazitní signál. Podle katalogových údajů jsou u hradel TTL zaručovány následující hodnoty napětí logických úrovní:



Obr.2.6 Podmínky napojení dvou hradel TTL

	LOG 0	**LOG 1**
VSTUP	**0÷0,8**	**2÷U_{CC}**
VÝSTUP	**0÷0,4**	**2,4÷U_{CC}**

Tab.2. 2 Hodnoty logických úrovní hradla TTL

Napojíme-li na sebe dvě hradla vzniká tedy napěťová rezerva. Z této rezervy pak určujeme šumovou imunitu obvodu a v daném případě je její zaručená velikost 0,4 V. Tato hodnota je sice malá, avšak vzhledem k tomu, že impedance hradel je rovněž malá, nemohou se kapacitní přeslechy uplatnit. Rovněž induktivní přeslechy jsou vzhledem k úrovní přenášených proudů zanedbatelné.

2.2 Zásady práce s obvody TTL

Při návrhu elektronických přístrojů s logickými obvody je nutno dodržet požadované napěťové úrovně vstupních signálů a při větších vstupních proudech některých logických obvodů (až I_L = 2 mA u STTL) respektovat také omezení velikosti vnitřního odporu zdroje signálu. Kromě statických parametrů je rovněž potřeba respektovat minimální přípustnou strmost hran vstupního logického signálu (např. v katalogu se uvádí limit 1 V/µs pro běžné obvody TTL a až 1 V/s pro obvody s hysterezní převodní charakteristikou).

2.2.1 Ošetření nevyužitých vstupů

U vícevstupových logických obvodů se musí věnovat péče vstupům, které nejsou funkčně využity. Nepoužité vstupy zásadně nelze ponechávat nepřipojené, neboť za těchto podmínek není přesně definována jejich vstupní logická úroveň. U standardních obvodů TTL se sice nepřipojený vstup nejčastěji chová jako by byl nastaven na úroveň H, ale má v tomto případě velmi nízkou odolnost proti rušení. U požadavků na rychlost odezvy těchto obvodů se navíc může projevit nežádoucím způsobem i zpoždění způsobené nabíjecím procesem, vázaným na parazitní kapacitu nepřipojeného vstupu (emitoru víceemitorového vstupního tranzistoru T1 obvodu TTL). Časové zpoždění reakce výstupu činí u obvodů TTL přibližně 1 ns na každý nepřipojený vstup. Proto platí zásada neponechávat nevyužité vstupy logických obvodů TTL nepřipojeny. V podstatě připojíme nevyužitý vstup na zdroj napětí definované úrovně L nebo H tak, aby nebyla narušena logická funkce ošetřovaného obvodu.

U vícevstupového obvodu NAND nebo AND musíme tedy nevyužité vstupy budit úrovní H. Můžeme to udělat např. tak, že připojíme nevyužité vstupy přes rezistor s odporem R < 15 kΩ (v rušeném prostředí raději R < 5 kΩ). Nevyužitý vstup lze připojit přímo na rozvod napájecího napětí U_{CC} = 5 V, pokud je zaručeno, že toto napětí v žádném případě nepřekročí mezní hodnotu 5,5 V (tedy ani při zapínání a vypínání). Můžeme je však připojit i k výstupu nepoužitého invertoru, jehož vstup jsme připojili na společný vodič. V nejjednodušším případě můžeme spojit nevyužité vstupy se vstupy použitými. U obvodů NOR nebo OR je to jinak: aby nebyla narušena jejich logická funkce, je nutno jejich nepoužité vstupy připojit na úroveň L, obvykle tak, že je připojíme na společný vodič. Můžeme ovšem stejně jako v předchozím případě připojit jednoduše nevyužité vstupy paralelně k použitým.

2.2.2 Připojování vstupů nevyužitých logických obvodů.

Při návrhu elektronického přístroje se může stát, že na desce s plošnými spoji zůstane nevyužit jeden nebo dokonce více logických členů. V tomto případě je vhodné připojit vstupy těchto nevyužitých obvodů na takovou úroveň, aby spotřeba těchto obvodů byla minimální. Například obvod NAND standardní TTL má proudovou spotřebu asi 1 mA při výstupní úrovni H a spotřebu asi 3 mA při výstupní úrovni L. Proto je vhodné vstupy nevyužitých obvodů NAND připojit na zem, čímž šetříme 2 mA na každý logický člen.

2.2.3 Úpravy vstupních logických signálů

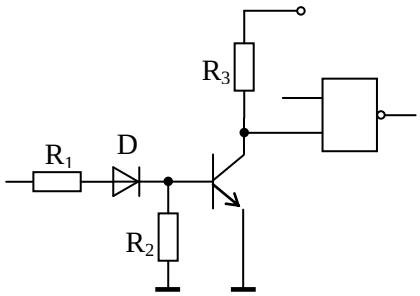
Často se vstup logického obvodu ovládá mechanickým kontaktem – nejčastěji tlačítkem. Při sepnutí tlačítka většinou dochází k zakmitávání kontaktů => vícenásobná změna logické úrovně. Zakmitávání je nutné odstranit korekčním obvodem. Nejčastěji používanými obvody jsou R-S klopný obvod a monostabilní kopný obvod.

Pokud jsou napěťové úrovně vyšší než předepsané, je nutné použít převodníků logických úrovní. Možná zapojení těchto převodníků jsou uvedeny na obr. 2.7.

Na obr. 2.8 je zapojení univerzálního převodníku pro napětí až ±100V. Použije-li se v tomto zapojení místo diody Zenerova dioda, je možné zpracovávat vstupní signály se stejnosměrnou složkou.

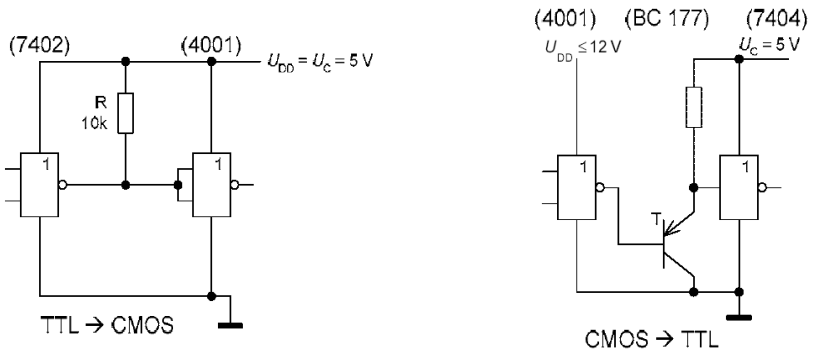
Tento převodník nalezne uplatnění všude tam, kde ovládací napětí 5V je nedostatečné (zaoxidované kontakty potřebují k protlačení proudu mnohem vyšší, např. 24V). Jinou možností je pak použití relátek v pouzdru DIL. Tyto relátka navíc poskytují i galvanické oddělení.

Obr. 2.7 Snížení vstupního napětí ve stavu H



Obr. 2.8 Univerzální převodník pro vstupní signály v rozmezí ±100V

Další přizpůsobení je nutné provést při kombinaci logických obvodů různých technologií. Unipolární integrované obvody CMOS jsou schopny pracovat v širokém rozmezí napájecích napětí U_{DD} = 3 až 18 V, popř. u moderních řad U_{DD} = 2 až 6 V. Lze je tedy provozovat i při napájecím napětí U_{DD} = 5 V a dalo by se očekávat, že budou schopny v tomto případě přímé spolupráce s obvody TTL. Avšak odlišné elektrické vlastnosti a provozní vlastnosti tuto spolupráci ztěžují.

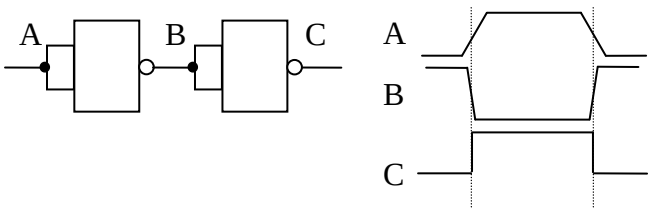


Obr. 2.9 Vzájemné propojení obvodu TTL a CMOS

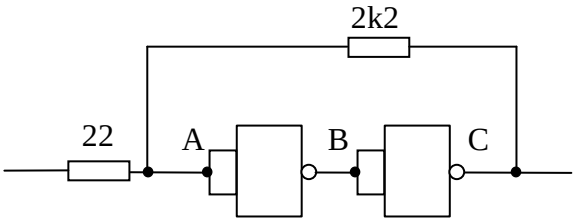
2.2.4 Tvarovací obvody

Signály, které získáme z převodníků napětí, obvykle nemají příliš strmé hrany, a proto je nutné obnovit strmost hran. Tvarovače musíme bezpodmínečně použít, budí-li signály vstupy klopných obvodů (některé vstupy externího přerušení u mikroprocesorů), a všude tam, kde hrany logického

signálu jsou delší než 400ns. Nejjednodušším obvodem je zapojení kaskády hradel, obr. 2.10. Zapojení se může vylepšit úpravou na obr. 2.11 Do obvodu je zavedena kladná zpětná vazba, která urychlí přechod hradla přes rozhodovací úroveň. Toto zapojení lze použít pro signály s hranou asi do 1 μs.

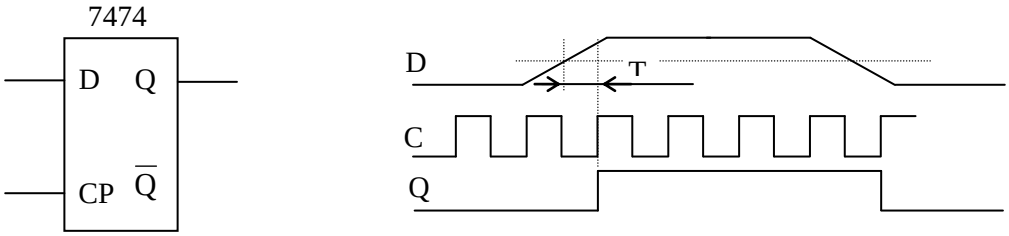


Obr. 2.10 Tvarování signálu hradly



Obr. 2.11 Tvarování signálu hradly s kladnou zpětnou vazbou

Ke tvarování signálů s delšími hranami používáme Schmitové klopné obvody (např 74132 nebo analogové). Další možnost tvarování signálu pomocí D klopného obvodu je na obr. 2.12. Nevýhodou je však nutnost buzení hodinovými pulsy.



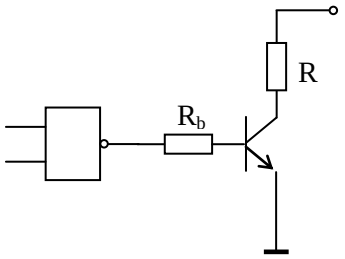
Obr. 2.12 Tvarování signálu D – klopným obvodem

2.2.5 Výstupy logických členů

Každé zařízení musí nějakým způsobem zajišťovat styk s řízeným objektem, informovat okolí o svých vnitřních stavech a v mnoha případech také využít zpracovaných logických signálů k řízení akčního zařízení. K tomu je zapotřebí obvykle vyšší energetická úroveň signálů, než kterou jsou schopny dodat samotné integrované obvody eventuelně vstup/výstupní port mikropočítače.

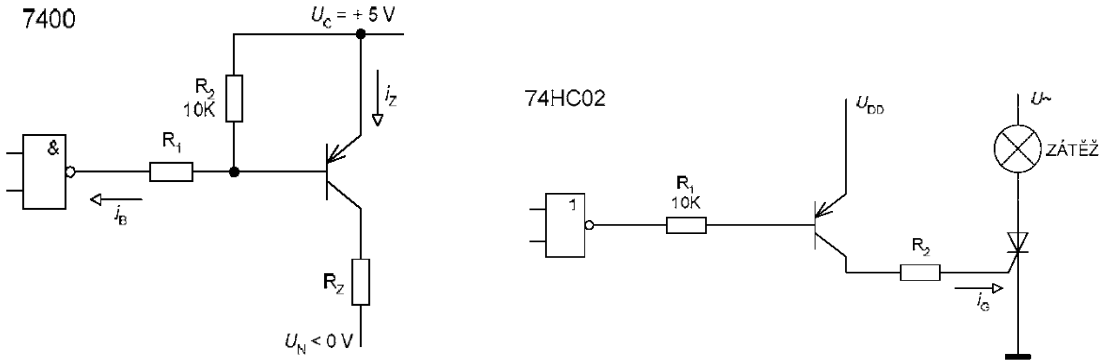
Nejčastěji používaným výstupním prvkem je relé nebo výkonový tranzistor. Pro sepnutí relé se používá tranzistorový zesilovač napojený na integrovaný obvod. Je třeba si uvědomit, že proud do báze je omezen pouze vnitřním odporem hradla. Podle toho je třeba volit i typ tranzistoru. Napojíme-li

však výstup z hradla přímo do báze tranzistoru, nelze pak již použít tento výstup k buzení vstupu dalšího hradla. Jedno z možných zapojení výstupních členů je na obr. 2.13.



Obr. 2.13 Zapojení výstupního členu

Při buzení výkonové zátěže z výstupu logického obvodu při aktivní úrovni L použijeme poněkud odlišná zapojení. V nejjednodušších případech můžeme připojit zátěž mezi sběrnici napájecího napětí a výstup logického obvodu. Jako příklad je uvedeno na obr. 2.14 připojení budicí cívky relé k výstupu výkonového logického členu NAND. Největší proud tekoucí zátěží nesmí být větší než mezní hodnota výstupního proudu hradla. Paralelně k indukční zátěži zapojujeme ještě ochrannou diodu D, která ochrání výstup logického obvodu proti nebezpečným napěťovým špičkám, vznikajícím při odpojení zátěže.



Obr. 2.14 výkonové zátěže při aktivní úrovni L

2.2.6 Rozvod logických úrovní

Pro správnou činnost zařízení sestaveného z číslicových integrovaných obvodů je důležité dodržovat pravidla určená pro vzájemné propojování těchto prvků mezi sebou. Jde hlavně o problémy rozvodu napájecích napětí a rozvodu logických signálů na vzdálenost větší než 25cm. Strmosti hran u obvodů TTL řady 74… jsou v okolí 10 ns. Z harmonické analýzy plyne, že to jsou sinusové kmitočty řádově desítky MHz; jsou to tedy problémy spadající do oblastí přenosu vf signálu.

Pro správnou funkci integrovaných obvodů je třeba především zajistit napájení. Zde se uplatňují dva požadavky:

1. Požadavek malého činného odporu vedení a zdroje
2. Požadavek malé impedance vedení a zdroje pro vf signál

Požadavek 1. je nutný pro proudové špičky, které se u TTL obvodů vyskytují při přechodu ze stavu „L“ do stavu „H“ a opačně. Tento stav se zhoršuje, jsou-li vstupy obvodů buzeny signálem, který nemá strmé hrany. Pracuje-li několik logických členů synchronně, jsou i tyto špičky synchronní. I toto je důvod, proč se těsně k napájecím vývodům pouzdra obvodu připojuje paralelně kondenzátor.

Požadavek 2. je co do velikosti určen především indukčnostmi přívodů. Proto se snažíme, aby jednotlivé napájecí části tvořily malé smyčky. Smyčky se pak ještě zmenšují zapojením kapacit, které působí jako vf zkrat.

Vzájemné propojování hradel mezi sebou se uskutečňuje více způsoby. Základní propojení se provádí plošným spojem. Mimo desku se mohou použít vodiče, souosé kabely, plošné spoje atd. U vedení musíme vždy počítat s tím, že na každém vedení mohou nastat tyto stavy:

1. signál se průchodem zpožďuje
2. na vedení dochází k odrazům
3. vedení se nabíjí a vybíjí
4. vedení se mohou navzájem ovlivňovat

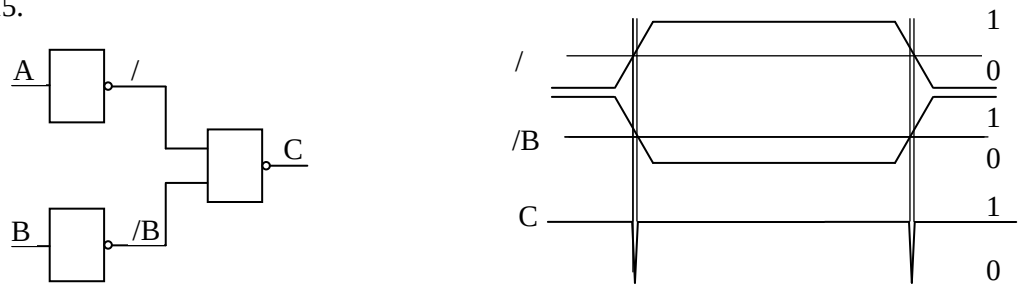
Stav 1 vychází z rovnice pro šíření signálu po vedení. Můžeme uvažovat, že zpoždění na 1 m délky vedení je asi 5 ns. Srovnáme-li tento čas s dobou zpoždění hradla vidíme, že oba časy jsou srovnatelné.

Stav 2 vyplývá z teorie vf vedení. Odrazům na vedení lze zabránit jen správným přizpůsobením dlouhého vedení. To se obvykle provádí tak, že se toto vedení zakončuje činným odporem o velikosti rovné charakteristické impedanci vedení Z_0. Není-li tomu tak, dochází k odrazům. Z charakteristik TTL obvodů vyplývá, že pro každý stav vstupu a výstupu je jiná impedance hradla. Proto je nutné u dlouhých vedení budit vždy jen jeden vstup jednoho hradla a ostatní vstupy mají být s tímto vstupem spojeny a nemají se využívat jako vstupy další logické proměnné. Dlouhé vedení se vůbec nesmí napojovat na vstupy klopných obvodů, pokud to není výslovně povoleno výrobcem.

Stav 3 způsobuje kapacita vedení, jež se musí nabíjet a vybíjet. Takto vzniklé proudy mohou na impedancích přívodů napájení způsobit napěťové špičky a ty pak mohou následně porušit šumovou imunitu obvodu. Zde je patrný další důvod připojování kondenzátorů k napájecím vývodům hradla.

Stav 4 má základní vlastnosti velmi jednoduché, ale v praxi nám činí největší potíže. Jedná se o přeslechy na vedení. Pro delší napojení je tento problém možno vyřešit použitím souosých kabelů nebo twist vedení. Jednotlivými vodiči, jsou-li umístěny proti zemní desce, je možné přenášet signály na vzdálenost asi 0,5m. Jednotlivé vodiče se nesnažíme svazovat do svazku, neboť klesá vzájemná impedance jednotlivých vodičů. Velmi dobré výsledky poskytuje metoda ovíjených spojů.

Při napojení jednotlivých obvodů TTL se můžeme setkat ještě s dalším neduhem a to jsou hazardní signály. Tyto vznikají vlivem časového rozptylu průchodu signálu. Ve skutečných logických sítích se může vyskytovat celá kombinace základních hazardních stavů. Jeden z případů je zachycen na obr. 2.15.



Obr. 2.15 Hazardní stav

2.2.7 Rozvod společného vodiče

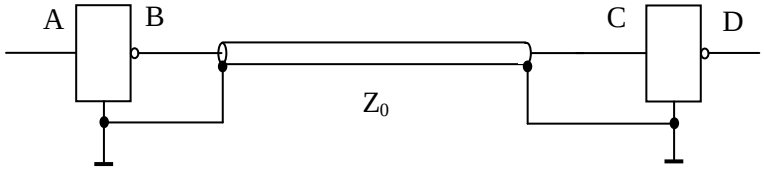
V mnoha případech je tento aspekt podceňován, ale špatně navržený rozvod může být zdrojem mnoha nečekaných závad při uvádění zařízení do chodu. V zásadě by měl být společný vodič realizován tak

(ať už drátem nebo měděnou fólií na desce s plošnými spoji), aby měl co nejmenší odpor a zanedbatelnou indukčnost. Těmto požadavkům vyhoví vodiče s velkým průřezem. Při návrhu plošných spojů se proto vždy snažíme navrhovat rozvody zemí co nejširší. Pokud se v zařízení kromě logických obvodů vyskytují také další typy obvodů, např. lineární integrované obvody, výkonové členy s relé apod., je nutné každý typ těchto obvodů propojovat samostatným společným (zemním) vodičem. Tyto skupinové vodiče zemí pak propojíme v jediném místě zařízení, nejčastěji u zemnicí svorky nejkritičtěji provozovaného operačního zesilovače (který zpracovává nejmenší úrovně signálu) nebo u napájecích zdrojů. Při dimenzování společného vodiče musíme uvažovat i maximální proudy, které jím mohou protékat. Nejsou vzácností desky plošných spojů s odběry až několik ampérů. Ze stejných důvodů jsou kladeny přísné nároky také na minimální přechodové odpory použitých konektorů a na spoje zemí mezi jednotlivými konektory v přístrojové skříni.

2.2.8 Buzení dlouhého vedení

Jak již bylo řečeno výše, podle doby šíření signálu po vedení a doby jeho náběžných resp. sestupných hran rozlišujeme vedení na krátká a dlouhá. U rychlých TTL obvodů je nutné vedení signálu na vzdálenost větší než 25 cm považovat za dlouhé vedení. V koaxiálním kabelu je rychlost šíření asi 5 ns na metr a v kroucené dvojlince (twisted pair, twist) asi 6 ns na metr. Každé vedení má svou charakteristickou impedanci Z_0, která je dána mechanickým uspořádáním a použitým dielektrikem. Koaxiální kabel má charakteristickou impedanci Z_0 = 50, 75, 90 Ω, twist kolem 120 Ω, spoj šíře 2 mm po obou stranách plošného spoje tloušťky 1 mm 50Ω, vodič o průměru 1 mm a 1 mm nad vodivou deskou 50Ω. Dlouhé vedení by mělo být impedančně přizpůsobené, tj.zakončené impedanci Z_0. Pokud není, tak na konci vedení dochází k odrazům, které se superponují na přenášený signál a zkreslují jeho průběh (vznikají i záporné napěťové špičky).

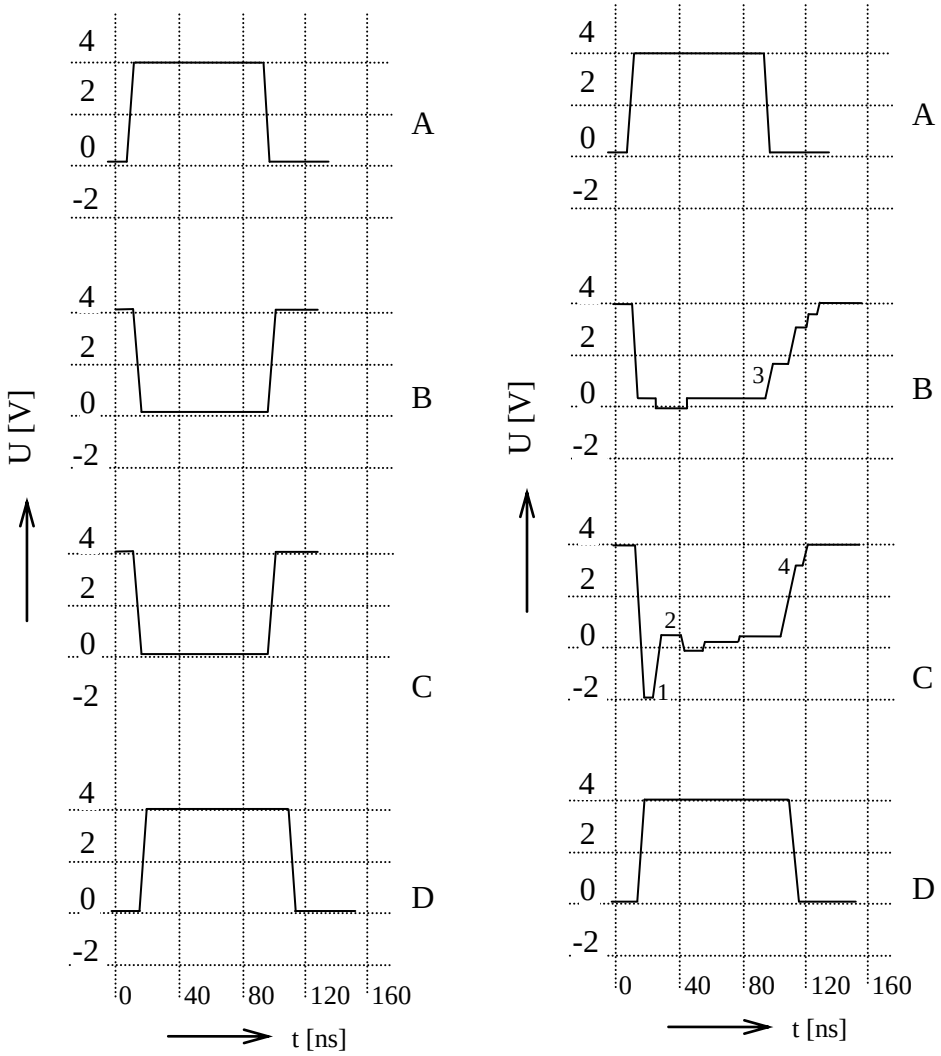
Z odrazů na vedení vyplývá, že z výstupu jednoho hradla je možné budit pouze jedno dlouhé vedení.Při zapojení několika vedení jsou impedance Z_0 zapojeny paralelně. Při různých délkách vedení k tomu ještě přistupuje další vlastnost, že odezvy z těchto vedení jsou různé, čímž vznikají složité přechodové jevy. se všemi důsledky na funkci logiky. Na vstup jednoho hradla má být zapojeno pouze jedno dlouhé vedení. Ostatní vstupy mají být s tímto vstupem propojeny a nemají se používat jako vstup další logické úrovně.



Obr. 2.16 Přenos pomocí dlouhého vedení

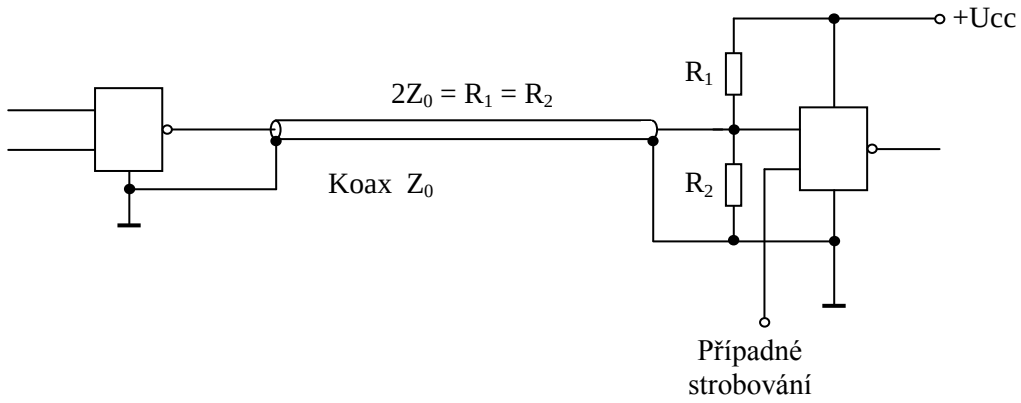
Důsledky nepřizpůsobeného vedení zachycuje obr. 2.17. Při vedení do délky 25 cm vidíme, že průběhy nejsou deformovány. Uplatní se pouze zpoždění vlivem délky vedení. Pro delší vedení (v našem případě 2 m) si můžeme všimnout, že zůstává nezměněn budicí signál v bodě A a D. V bodech B a C se vlivem odrazů signál značně deformuje. Strmé zůstávají sestupné hrany, neboť výstupní impedance hradla je v tomto režimu velmi malá; pouze v bodě C se objeví záporná špička 1, neboť zde je vedení (z hlediska praxe) ve stavu, jako by bylo naprázdno, a napětí se tedy odráží v opačné fázi. Amplituda je však omezena na hodnotu asi -1,5 V, což způsobí substrátová dioda. Tato dioda vznikne automaticky topologickým uspořádáním vlastního integrovaného obvodu; je však na průraz citlivá, její proražení způsobí výpadek činnosti hradla. Odraz od výstupu hradla se projeví jako špička 2. Amplituda této špičky je asi 0,4 V, a tedy na mezi šumové imunity. Její velikost je podmíněna amplitudou špičky 1 a poměrem výstupní impedance hradla a impedance Z_0 vedení. Maxima dosahuje pro impedanci Z_0 = 75 Ω. Dalším charakteristickým bodem je bod 3. Tento schod má amplitudu danou poměrem výstupní impedance hradla k impedanci Z_0 vedení. Délka schodu je pak dána dvojnásobkem

zpoždění na jedné délce vedení. Amplitudu schodu můžeme zvětšit, budeme-li vedení budit výkonovým hradlem MH7440. Další schod nastává v místě 4. Tento schod, protože je závislý na velikosti schodu 3, musí být nad úrovní 2,5 V a nemá tedy žádný vliv na průběh v bodě D.

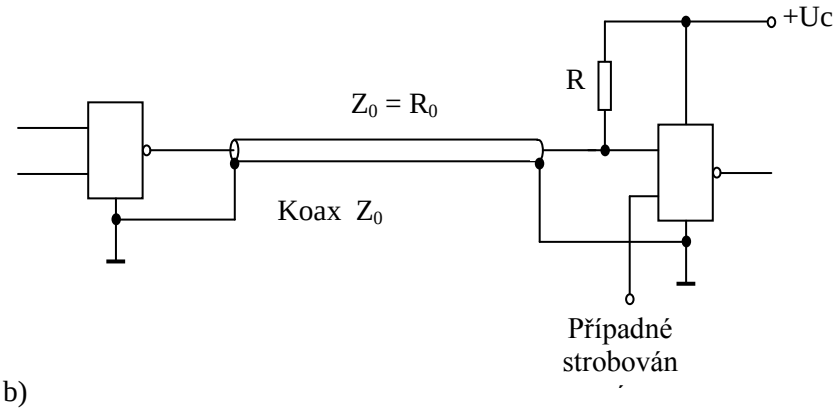


Obr. 2.17 Poměry na vedení do 20cm (vlevo) a 2m (vpravo)

Vedení zakončené vstupem obvodu TTL je vzhledem k jeho velké vstupní impedanci impedančně nepřizpůsobené. Přizpůsobovací obvody jsou na obr. 2.18 Nejvhodnější pro přenos na větší vzdálenosti je koaxiální kabel (velká odolnost proti rušení –přeslechům), do tří metrů lze použít twist, vodič těsně nad vodivou deskou do 50cm, drátové a plošné spoje do 30cm.



a)



b)

Obr. 2.18 Přizpůsobení dlouhého vedení

Shrnutí pojmů 2

Klíčová slova:
Hazardní stav, Hradlo TTL, Šumová imunita

Otázky 2

1. Vysvětlete princip hradla NAND
2. Uveďte napájecí napětí logických obvodů
3. Co je to šumová imunita
4. Uveďte některé zásady práce s obvody TTL
5. Popište princip rozvodu logických úrovní na velké vzdálenosti

6. Co je to hazardní stav
7. Navrhněte výstupní obvod pro napájení žárovky obvodem TTL
8. Jaké jsou zásady buzení dlouhého vedení obvody TTL

Úlohy k řešení 2

1. Napájecí napětí obvodů TTL je

a) 5V ± 5%
b) ± 12V
c) 3 ÷ 15V
d) 3,3V

2. Je možné zaměnit obvody řady 74LS… s obvody řady 74HCT…

a) ano, prakticky ve všech případech
b) ne, jedná se o zcela odlišné technologie
c) ano, po úpravě napěťových úrovní
d) ano a navíc je možno tyto obvody napájet napětím v rozsahu 3 až 7V

3. Je možné připojit na zem výstup hradla TTL?

- ano, ale jen jeden člen v obvodu DIL
- ne, došlo by ke zničení hradla
- jen při stavu log. 0 na výstupu
- ano

4. Při připojení záporného napětí na vstup hradla TTL

- se hradlo chová stejně jako při připojení logické 0
- dojde nastavení výstupu do logické 1
- dojde ke zničení vstupního tranzistoru
- výstup přejde do stavu vysoké impedance

5. Logické úrovně CMOS jsou definovány

a) log0: 30%Ucc, log1: 70%Ucc
b) log0: 0,4V, log1: 2,8V
c) log0: 0,8V, log1: 2,4V
d) log0: 40%Ucc, log1: 60%Ucc

6. Přivedením napětí vyššího než 8 V na vstup hradla TTL

- dochází k nastavení výstupu do logické 1
- dojde k průrazu vstupního tranzistoru
- chová se obvod stejně jako při logické 1 na vstupu
- dochází k nastavení výstupu do úrovně logické 0

7. Napájecí napětí logických obvodů CMOS 4000 je:

e) 5V ± 5%
f) ± 12V
g) 3 ÷ 15V
h) 3,3V

8. Pomalé přechody z úrovně logické 1 do logické 0 na vstupu hradla TTL

- má za následek zničení obvodu
- způsobí zvýšení příkonu hradla
- nemá vliv na příkon hradla
- způsobí nesprávnou činnost obvodu

9. Tvarovací obvody se používají především tam, kde hrany logických úrovní dosahují délky větší než:

a) 1 ns
b) 1 ms
c) 400 ns
d) 400 us

10. Za dlouhé vedení z hlediska rychlosti hradel při jejich propojování

- se považuje vedení signálů delší než 100 cm
- se považuje vedení kratší než 1 mm
- se považuje vedení delší než 20 cm
- se nepovažuje žádné propojení v rámci desky plošných spojů

11. Při napojení hradla na dlouhé vedení

- by neměly být na jeho vstup připojeny žádné další vstupy
- můžeme připojit další vstupy hradel TTL
- může být na výstup připojeno více dlouhých vedení různých délek
- je třeba provést impedanční přizpůsobení vedení

12. Napěťové úrovně obvodů TTL pro stavy log.1 a log.0:

e) jsou odvozeny od napájecího napětí
f) jsou pevně stanoveny pro vstup i výstup hradla
g) při 5V napájení jsou stejné jako u CMOS 4000
h) závisí na použité řadě (LS, AS...)

13. Napěťové úrovně obvodů CMOS pro stavy log.1 a log.0:

a) jsou odvozeny od napájecího napětí
b) jsou pevně stanoveny pro vstup i výstup hradla
c) při 5V napájení jsou stejné jako u TTL
d) závisí na použité řadě (4000, HC, HCT...)

14. Napěťová úroveň obvodů TTL pro stav log.0 je:

a) pro vstup 0÷1V, pro výstup 0÷0.4V
b) pro vstup 0÷0.8V, pro výstup 0÷0.4V
c) je závislá na napájecím napětí
d) 0÷0.4V pro vstup i výstup hradla

15. Napěťová úroveň obvodů CMOS 4000 pro stav log.1 je:

a) pro vstup vetší než 3V, pro výstup větší než 4.2V
b) je závislé na napájecím napětí
c) je větší než 0.7 U_{cc}
d) pro vstup vetší než 2,1V, pro výstup větší než 3,6V

16. Příkon hradla logických obvodů TTL a CMOS ve statickém režimu:

a) je shodný
b) je větší u řady TTL
c) je větší u řady CMOS
d) je u CMOS stejný jako v dynamickém režimu

17. Pojem šumová imunita:

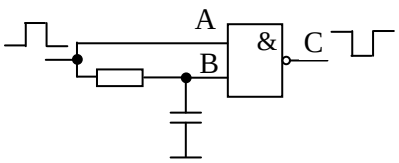
a) udává počet logických vstupů hradla které můžeme budit z jednoho výstupu
b) je odolnost obvodů vůči rušivým signálům
c) je odvozena z logických úrovní hradel
d) platí jen pro technologii CMOS

18. Nepoužité vstupy hradla NAND je třeba připojit:

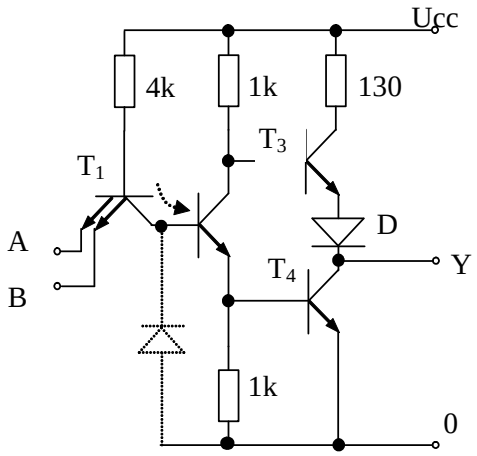
a) paralelně k použitým vstupům
b) na napětí logické úrovně L
c) na napětí logické úrovně H
d) není třeba ošetřovat

19. Toto zapojení slouží ke:

a) zpoždění náběžné hrany výstupního signálu
b) zpoždění sestupné hrany výstupního signálu
c) úpravě výstupní úrovně signálu
d) ošetření nevyužitých vstupů obvodu



20. Jakému typu hradla odpovídá toto zapojení:



a) AND
b) NAND
c) OR
d) NOR

21. Nezapojené vstupy u TTL obvodů způsobují časové zpoždění reakce výstupu přibližně:

a) 1 ms na jeden nezapojený vstup
b) 1 us na jeden nezapojený vstup
c) 1 ns na jeden nezapojený vstup
d) 1 ps na jeden nezapojený vstup

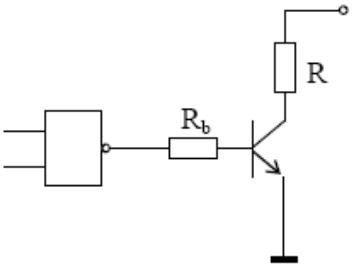
22. Dva nevyužité vstupy 4-vstupého obvodu NAND nebo AND ošetříme tím, že připojíme:

a) oba na úroveň L
b) první na úroveň H druhý na úroveň L
c) oba na úroveň H
d) první na úroveň L druhý na úroveň H

23. Aby měl nevyužitý 4-vstupý člen NAND typu TTL nejmenší proudovou spotřebu zapojíme:

a) všechny čtyři vstupy na úroveň H
b) první dva vstupy na úroveň L a druhé dva na úroveň H
c) první dva vstupy na úroveň H a druhé dva na úroveň L
d) všechny čtyři vstupy na úroveň L

24. Zapojení na obrázku představuje:



a) univerzální převodník pro vstupní signály
b) stav vysoké impedance u datové sběrnice
c) příklad obvodu tvarování signálu hradly
d) příklad zapojení výstupního členu

25. Při návrhu zapojení s logickými obvody je vhodné:

a) umístit co nejblíže obvodu blokovací kondenzátory
b) udržovat minimální vzdálenost mezi dvěma logickými obvody
c) zajistit definovaný stav po zapnutí napájecího napětí
d) používat pasivní součástky s minimální indukčností

26. Schmittův klopný obvod:

a) má hysterezi na vstupech
b) potřebuje ke své funkci hodinový signál
c) je převodník úrovní signálu z TTL logiky na CMOS
d) je vhodný k tvarování signálů

27. Logické obvody s výstupy typu „otevřený kolektor“:

a) jsou tvořeny na výstupu komplementární dvojící tranzistorů
b) jsou určeny k řízení sběrnic
c) potřebují pro log.1 „pull up“ rezistory
d) jsou vhodné pro převod úrovní z TTL logiky na CMOS

28. Při logické úrovní L na vstupu hradla TTL:

- teče proud dovnitř hradla
- teče proud ven z hradla
- teče proud ven z hradla jen u obvodu AS a ALS
- teče proud dovnitř jen u obvodu AS a ALS

29. Výstupní odpor hradla TTL je:

- stejný při logické úrovni H i L
- odlišný při logické úrovni H od L
- závislý na výstupním proudu
- závislý na napájecím napětí

3. POJEM MIKROPOČÍTAČ A MIKROPROCESOR

Čas ke studiu: 3 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- vysvětlit pojmy mikroprocesor a mikropočítač
- popsat princip a činnost mikroprocesoru
- popsat způsoby adresování
- vysvětlit princip a použití přerušení

Výklad

3.1 Základní struktura mikropočítače

Obvody mikropočítače můžeme rozložit do pěti částí, jak je zobrazeno na obr. 3.1. Data jsou mikropočítačem zpracovávána po slovech. V daném časovém okamžiku (takt procesoru) mikropočítač pracuje s jedním slovem. Typická délka slova je 8, 16, 32 nebo 64 bitů. Nejrozšířenější jsou mikropočítače s délkou slova 8 a 16 bitů. Osmibitové slovo nazýváme bajt.

❑ Generátor taktů

Generuje hodinový (taktovací, synchronizační) signál, který synchronizuje činnost samotného procesoru a také jeho spolupráci s ostatními částmi mikropočítače. U současných typů mikroprocesoru bývá již tento generátor jejich součástí.

❑ Mikroprocesor

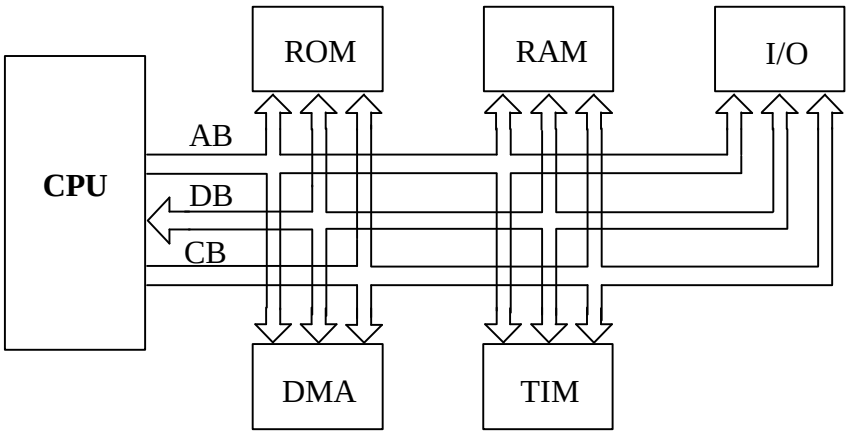
Je základním prvkem mikropočítače. Řídí jeho celou činnost. Zajišťuje provádění instrukcí uložených v paměti, řídí toky dat ze vstupních částí mikropočítače tyto data zpracovává a následně řídí tok dat směrem k výstupním portům.

❑ Paměť ROM

Obsahuje ve většině případů instrukce, které zajišťují realizaci daného algoritmu řízení pro přizpůsobení mikropočítače určité aplikaci. Dále paměť může obsahovat konstanty a neměnné tabulky používané v programu. Z této paměti lze pouze číst, programuje se při výrobě. Určitými variantami paměti ROM jsou paměti PROM, EPROM, o kterých bude zmínka v dalším textu.

❑ Paměť RWM

Označována někdy také RAM zajišťuje dočasné uložení dat zpracovávaných mikroprocesorem. Data do paměti může mikroprocesor uložit a opět zpětně vyzvednout. Do této paměti lze tedy i zapisovat.



Obr. 3.1 Blokové schéma mikropočítače

❑ Vstupní a výstupní porty

Umožňují spojení mikropočítače s okolním prostředím (klávesnice, display, výkonové akční členy atd.). Tyto však někdy nemusí být součástí každého mikropočítače. Pak je spojení s vnějším prostředím provedeno jiným způsobem, což bude vysvětleno dále.

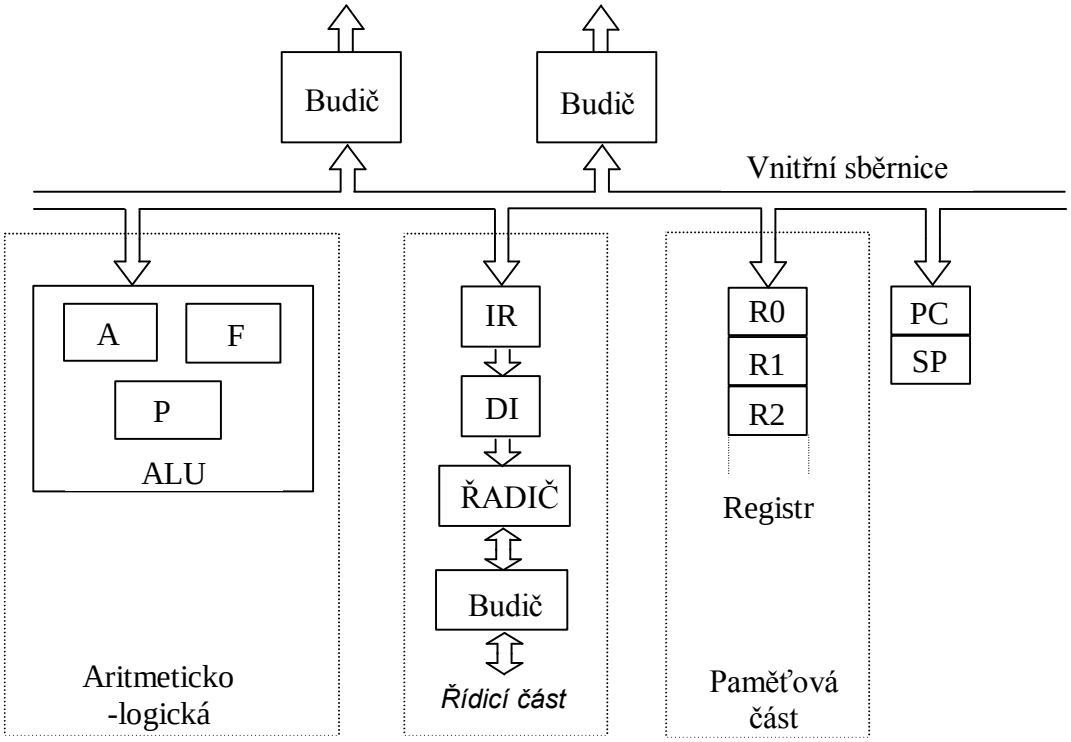
Těchto pět částí tvoří nutný základ mikropočítače. Mikropočítač však může být tvořen i dalšími částmi, jež zefektivňují jeho práci. O těchto bude rovněž hovořeno později.

3.2 Struktura mikroprocesoru

Termín mikroprocesor označuje ve většině případů integrovaný obvod vyrobený technologií vysoké hustoty integrace. U starších typů mikroprocesoru nebyly vždy součástí mikroprocesoru všechny jeho nezbytné části jako jsou např. generátor hodinového signálu, obvody pro řízení sběrnice a jiné. Výrobci k těmto mikroprocesorům dodávali specializované obvody, které doplňovaly mikroprocesor o tyto části. Tyto obvody pak tvořily spolu s mikroprocesorem tzv. skupinu mikroprocesoru. Současné typy vyráběných mikroprocesorů jsou již těmito potřebnými částmi vybaveny přímo na čipu.

Zjednodušené blokové schéma blíže nespecifikovaného mikroprocesoru je na obr.3.2. Toto schéma je sestaveno jen pro naše účely a obsahuje všechny základní části. Konkrétní mikroprocesory se mohou od této sestavy odlišovat.

Každý mikroprocesor obsahuje **řadič**, který řídí chod celého mikroprocesoru a je tvořen registrem instrukcí, obvodem pro dekódování instrukce a řídicím obvodem, což souhrně nazýváme **řídicí částí.** Další částí mikroprocesoru je **aritmeticko-logická jednotka,** soubor registrů – tzv. **paměťová část** a sběrnice, které všechny tyto části propojují.



Obr. 3.2 Příklad vnitřního blokového schématu mikroprocesoru

3.2.1 Řadič

Řídí a koordinuje činnost všech ostatních částí mikroprocesoru. Podílí se i na vytváření vnějších signálů řídicí sběrnice celého mikropočítače. Jeho hlavní činností je načítat instrukci po instrukci z paměťových míst a po dekódování ji zpracovat. Adresa, odkud má danou instrukci načíst, se při každém instrukčním cyklu nachází v programovém čítači (Program Counter-PC nebo Istruction Pointer-IP) jehož obsah se po provedené instrukci nejčastěji modifikuje prostou inkrementací. To je v případě, že jsou instrukce v paměti uloženy postupně za sebou. Někdy je obsah programového čítače modifikován naplněním jinou hodnotou, než která by vznikla inkrementací. Je to v případech, kdy je třeba provést jinou instrukci, která přímo nenásleduje za právě provedenou. Takový případ nastává po instrukcích skoku (větvení), volání podprogramu, přerušení atd.

Adresu pak musí řadič zpřístupnit paměťovému místu, kde je uložen program - tedy instrukce - nebo data. V některých případech instrukce zabírá více než jedno paměťové místo. Pak jsou její jednotlivé části uloženy v paměti postupně za sebou. Současně jsou řadičem generovány v potřebné časové posloupnosti i signály potřebné k řízení paměti. Ta po té vystaví potřebnou instrukci nebo její část na datovou sběrnici (viz níže), odkud si ji řadič vyzvedne a uloží do registru instrukcí (IR). V tomto registru zůstává instrukce po celou dobu jejího výkonu.

3.2.2 Soubor registrů

Tvoří tzv. paměťovou část mikroprocesoru. Jsou to v podstatě samostatné izolované paměťové buňky, které jsou vytvořeny přímo ve struktuře mikroprocesoru a jsou adresovatelné přímým zapsáním dat. Tento způsob adresace zajišťuje velkou rychlost přístupu dat bez použití adresových a řídicích sběrnic. Tuto paměťovou část mikroprocesor využívá k dočasnému uchování informací - dat v průběhu vykonávání operací nebo je využívána programem, tedy vlastně uživatelem. Různé typy mikroprocesorů se liší také počtem registrů. Někdy bývá uspořádání takové, že existuje jakási základní sada registrů, tzv. banka a další registry jsou tvořeny stejnými názvy přepnutím další banky. Jinými slovy je v každém okamžiku (strojovém cyklu) přístupná pouze jedna sada registrů. Při požadavku přístupu do jiné banky je nutno danou banku nastavit jako aktuální. Toto je jeden z mnoha možných způsobů uspořádání paměťové části mikroprocesoru nikoliv mikropočítače.

3.2.3 Aritmeticko-logická část

Je další důležitou částí mikroprocesorů. Tvoří ji aritmeticko-logická jednotka a podílí se na vykonávání všech aritmetických a logických operací mikroprocesoru. U jednoduchých procesorů umožňuje sčítat a odčítat, u výkonnějších typů pak může provádět násobení, dělení a posuvy slov ve dvojnásobné přesnosti. Oproti kalkulačkám mají tedy mnohem menší matematickou vybavenost a složitější matematické operace je třeba řešit prostřednictvím numerických algoritmů v programu. Mikropočítače, které jsou předurčeny pro vykonávání složitějších nebo přesnějších matematických operací v krátkých časech, je vhodné doplnit dalšími technickými prostředky (např. koprocesory). Některé složitější mikroprocesory mohou již tyto technické prostředky obsahovat ve své struktuře. Představiteli těchto mikroprocesorů jsou např. signálové procesory. Obsahují např. na čipu implementovanou hardwarovou násobičku, která umožňuje vynásobit dvě čísla v jednom strojovém cyklu.

Některé aritmeticko-logické jednotky ke své činnosti vyžadují tzv. střadač (akumulátor), t.j. registr, ve kterém je ve většině případů uložen jeden z operandů vstupujících do matematické operace. další operand je po té k dispozici buď přímo z operačního kódu (opcode) instrukce anebo se může nacházet v nějakém registru či paměti mikropočítače. Je-li přítomen střadač, pak u takových mikroprocesorů se vždy účastní matematických operací. Výsledek je po té uložen zpět do střadače. Tento způsob užívání střadače může některým uživatelům poněkud komplikovat práci nezbytnými přesuny při přísunu dat, která se stávají operandy jednotlivých operací a při úklidu výsledků, který je nutný, abychom si je nepřepsali.

Některé typy mikroprocesorů však mohou provádět matematické operace bez účasti střadače (u takových pak střadač vůbec neexistuje) a to tak, že pro tyto operace využívá buď soubor registrů na čipu nebo přímo paměťové pozice. Takové mikroprocesory se vyznačují většími možnostmi z hlediska uživatele pro vývoj programu. Menší nevýhodou při využívání paměti pro matematické operace je doba přístupu do paměti. Ta může mít za následek zmenšení výpočetní rychlosti. Pak je v tomto případě vhodné využívat pamětí (např. umístěných přímo na čipu, pokud jsou ve struktuře mikroprocesoru obsaženy), které nezatěžují mikroprocesor svou přístupovou dobou.

Součástí aritmeticko-logické části je tzv. příznakový registr (Flags). Jednotlivé jeho bity nám dokumentují prováděné matematické operace, informují nás o významných vlastnostech výsledků. Na základě těchto bitů můžeme pak provádět modifikaci daného algoritmu např. prostřednictvím podmínek v programu.

Toto je základní struktura obecného, blíže nespecifikovaného mikroprocesoru. Na tento popis základních částí a obr.3.2 se budeme ještě dále v textu odvolávat.

Nyní si ještě popíšeme základní parametry, podle kterých je možné mikroprocesory dále dělit.

Prvním parametrem je šířka vnitřní sběrnice, jinými slovy kolikabitový je . V některých případech platí, že šířka vnitřní sběrnice mikroprocesoru (jeho vnitřní sběrnice dat) je shodná se šířkou datové sběrnice mikropočítačové struktury s tímto procesorem. V jiných je typické, že mikroprocesor obsahuje bloky, které jsou kvůli zvýšení výkonnosti uvnitř vícebitové. Šířka vnější sběrnice je v takovém případě redukovaná. Obvykle se tedy šířka vnitřní sběrnice mikroprocesorů posuzuje podle šířky slova registrů eventuelně střadače.

Druhým parametrem je rychlost mikroprocesoru posuzována podle frekvence krystalu (oscilátoru), lépe však podle výkonu mikroprocesoru daným počtem vykonaných instrukcí za sekundu. Tento údaj se udává v tzv. MIPSech, t.j. miliónech instrukcí za sekundu. Vzhledem k tomu, že instrukce trvají různý počet strojových cyklů, je tento údaj určitým průměrem. U mikroprocesorů dnešní doby trvá provedení většiny instrukcí pouze jeden strojový takt.

Dalšími parametry může být např. velikost adresovatelného prostoru, údaje o technologii výroby, které mají vliv na napájecí napětí a proud, apod.

3.3 Základní činnost mikroprocesoru

3.3.1 Časování mikroprocesoru

U mikroprocesorů je časování odvozeno z hodinových signálů, které mikroprocesor synchronizují. Ve většině případů je to signál vzniklý dělením frekvence krystalového oscilátoru. Základní časovou jednotkou je tedy perioda tohoto signálu zvaná takt.

Každá instrukce se potom vykonává určitý počet taktů (dob), celý tento čas nazýváme instrukční cyklus. Délka instrukčního cyklu se může lišit podle složitosti instrukce. Některé instrukce, ve většině případech instrukce s podmínkou, mají proměnnou délku instrukčního cyklu. Podle délky instrukčních cyklů jednotlivých instrukcí jsme však schopni určit délku trvání jednotlivého programu. Toto např. vyžadujeme u výpočtu programového zpoždění. S růstem výkonnosti mikroprocesorů se však výrobci snaží o efektivnější činnost mikroprocesoru, což vede k tomu, že některé bloky pracují vlastně paralelně a jednotlivé instrukční cykly se překrývají - tzv.pipelining. Pak je výpočet doby programu velmi problematický, ne-li nemožný.

Mezi instrukčním cyklem a jednotlivými takty ještě obvykle rozeznáváme tzv. strojové cykly. Jsou to dílčí operace, ze kterých se instrukční cyklus sestává a obvykle souvisejí s činnostmi, které proběhnou mezi mikroprocesorem a jeho okolím. Prvním strojovým cyklem každého instrukčního cyklu je ve většině případech načtení operačního kódu instrukce do instrukčního registru IR. Dalším strojovým cyklem instrukce dekóduje na daný operační kód, v dalších strojových cyklech nastává čtení operand neboli dat a vykonání instrukce.

Strojový cyklus se pak skládá z jednotlivých taktů a je jednoznačně stanoveno, jakou činnost mikroprocesor v každém taktu vykonává, kdy generuje které řídicí signály, kdy naopak řídicí signály testuje. V manuálech jednotlivých mikroprocesorů jsou časové posloupnosti uvedeny v časových diagramech zápisu a čtení z a do paměti, periférií apod.

3.3.2 Instrukční soubor

Již bylo řečeno, že řadič načítá do registru instrukcí jednotlivé instrukce z nějakého paměťového místa. Tyto jsou však v paměti uloženy v tzv. binární formě, tedy ve formě nul a jedniček. Je zřejmé, že tato forma zápisu není pro přípravu programu vhodná. I kdybychom převedli tuto formu na hexadecimální a tedy daleko přehlednější, stále není nejvhodnější. Snahou je vyjádřit instrukce i jejich adresy nějakou vhodnou, přehlednou a pro uživatele přijatelnou formou. Jedním z takových vhodných způsobů zápisu programu je tzv. Jazyk symbolických adres (JSA). Každý typ mikroprocesoru má jazyk symbolických adres definovaný výrobcem a soupis instrukcí v tomto zápisu bývá součástí každého manuálu. Nic však nebrání tomu, abychom použili pro stejný typ mikroprocesoru jazyk symbolických adres svůj nebo vypracovaný jinými uživateli. Výrobci taktéž dodávají i části programového vybavení a pomocné prostředky pro vývoj programového vybavení a potom tedy použití stejného JSA usnadňuje a často i podmiňuje užívání těchto prostředků. V jazyce symbolických adres je každá instrukce tvořena krátkým snadno zapamatovatelným jménem (mnemonickou zkratkou), které bývá tvořeno zkrácením popisu funkce instrukce.

Instrukce se skládá jak už bylo řečeno z jednoho nebo několika slov. Obsahuje vždy nejprve operační kód instrukce (Opcode). Některým instrukcím to stačí, jiné obsahují bezprostředně v sobě data nebo adresu, které, pokud se nevejdou do jednoho slova, jsou uloženy v dalších slovech. Šířka slova v této kapitole je v podstatě shodná s šířkou datové sběrnice mikroprocesoru. Někdy bývá zvykem nazývat slovem 16 bitů. Instrukce můžeme rozdělit podle jejich činností do několika skupin.

- **Instrukce přesunů**

Tato skupina instrukcí je vlastně základní. Je nutno mít možnost přesouvat data mezi jednotlivými částmi mikroprocesoru i mikropočítače. Tyto instrukce přesouvaná data neovlivňují. K přesunům dochází mezi jednotlivými registry mikroprocesoru, mezi paměťovými buňkami i mezi sebou navzájem. K těmto instrukcím rovněž řadíme i ty, které daný registr nebo paměťovou buňku naplňují

novými daty. Dále do této skupiny také řadíme instrukce pro práci se zásobníkem (podrobněji o zásobníku viz. kap.3.4). Mikroprocesor obvykle dovoluje přesuny tolikabitových dat, kolikabitový sám je. Nabízí i několik instrukcí ve dvojnásobné délce. Tyto pak vlastní přesun provádějí nadvakrát. Dalším typem instrukcí přesunu jsou pak instrukce přesunu bloku dat. U většiny mikroprocesorů tyto instrukce neovlivňují příznaky. Nelze tedy data přesunou a ihned testovat, zdali jsou např. nulové. Je nutno provést instrukci, která tyto příznaky nastaví. Není to však pravidlem a je tedy třeba se podrobně seznámit s daným typem mikroprocesoru a jeho instrukčním souborem.

❑ Instrukce aritmetické

Mezi tyto patří instrukce prováděné v aritmeticko-logické jednotce. Mikroprocesory jsou vybaveny pouze jednoduššími operacemi (sčítání, odčítání, násobení a dělení) v délce slova odpovídající mikroprocesoru a několika instrukcemi v dvojnásobné přesnosti. Zpracování složitějších operací, jak už bylo řečeno, je prováděno numerickými algoritmy nebo je přenecháno různým matematickým koprocesorům. Podle těchto instrukcí také snadno zjistíme, zda se u mikroprocesoru podílí na matematických operacích střadač anebo zda mikroprocesor dovoluje umístění operandů a výsledku i v jiných registrech či paměťových buňkách. Tyto instrukce zpravidla ovlivňují příznakový registr, který vlastně obsahuje další informace o výsledku.

❑ Instrukce logické

Tyto instrukce řeší logické operace jako je např. logický součet, součin apod. s daty. Většinou se tyto instrukce vztahují na celé slovo. Taktéž tyto instrukce mohou ovlivňovat některé příznaky.

❑ Instrukce posuvu a rotace

Provádějí posun jednotlivých bitů binárního čísla (slova) o jeden doleva nebo doprava. U posuvů je charakteristické, že bit, který opustí nejvyšší nebo nejnižší pozici slova se ztrácí a na druhé straně je nahrazen nejčastěji nulou nebo jedničkou podle druhu posuvu. U těchto instrukcí se velmi často s výhodou využívá skutečnost, že posuv doprava s doplněním nejvyšší pozice nulou představuje dělení dvěma a posuv doleva s doplněním nejnižší pozice nulou představuje naopak násobení dvěma.

U rotace probíhá taktéž posun s tím, že bit který opouští nejvyšší pozici je zařazen na pozici nejnižší a naopak. Rotace je prováděna s příznakovým bitem CY nebo bez něj podle použité instrukce.

❑ Instrukce skoku (větvení)

Tato skupina instrukcí ovlivňuje programový čítač, což ve své podstatě způsobí, že program nepokračuje následující instrukcí ale instrukcí, která se nachází na adrese, jež je uvedena jako parametr instrukce skoku. Rozeznáváme několik druhů instrukcí skoku.

❑ Nepodmíněný skok

Nastavuje vždy programový čítač na hodnotu danou parametrem instrukce. Parametr je v zápisu programu uveden ve většině případech jako návěští, které udává adresu, kde má daný program pokračovat. Toto návěští se pak v zápisu programu umísťuje před danou část programu.

❑ Podmíněný skok

Vykoná skok pouze je-li splněna určitá podmínka opět uvedena v parametrech instrukcí. Není-li splněna podmínka, program pokračuje následující instrukcí.

❑ Volání podprogramu

Funkce této instrukce je podobná nepodmíněnému skoku s tím rozdílem, že před modifikací programového čítače novou hodnotou se původní hodnota uloží do tzv. zásobníkové paměti (viz. kap.8). Tím se pak umožní návrat na instrukci následující za instrukcí volání poddprogramu po návratu z podprogramu.

❑ Návrat z podprogramu

Tato instrukce modifikuje programový čítač hodnotou, která byla poslední uloženou v zásobníkové paměti. Program dále pokračuje v místě hlavního programu, ve kterém nastalo volání podprogramu viz. předchozí instrukce.

Poslední dvě instrukce mohou být podmíněné - jsou tedy vykonány pouze při splnění podmínky v parametru instrukce.

❑ Instrukce řídicí

Zde můžeme zařadit instrukce, které v mikroprocesoru něco nastavují. Jsou to např. povolení a zákaz přerušení, výběr bank registrů, pamětí, nastavení nebo nulování bitů některých registrů atd.

3.3.3 Způsoby adresování

Jak již bylo řečeno, většina instrukcí vykonává činnosti s určitými daty. Tyto data jsou ve své podstatě operandy instrukce. Adresování pak řeší, jakým způsobem se mikroprocesor k těmto datům dostává. Je důležitou informací při seznamování se s činností mikroprocesoru a přibližně nastiňuje filozofii jeho práce.

Např. u instrukcí přesunu musí mikroprocesor adresovat jak zdroj dat této operace, tak i cíl, takže adresuje dvakrát. Toto může provádět různými metodami. Celou řadu těchto metod můžeme spolu kombinovat, což někdy vytváří poměrně komplikované, ale velmi efektivní vytváření adresy.

Některé instrukce adresují implicitně, což znamená, že adresa nebo adresy jsou již ukryty v operačním kódu instrukce. Nejčastěji je toto adresování typické pro instrukce pracující se střadačem. I když se může uživateli při psaní programu v jazyce symbolických adres zdát, že je tento operand vyjádřen explicitně, jedná se o implicitní vyjádření adresy.

Pozn.: Pro ilustraci uvádíme záměrně více instrukcí od různých mikroprocesorů, které vykonávají stejnou činnost.

❑ Bezprostřední adresování

U tohoto adresování není uvedena adresa operandu v pravém slova smyslu, nýbrž po operačním znaku následuje bezprostředně sám operand. Je vlastně součástí instrukce nebo-li operačního kódu (Opcode). Jako příklad můžeme použít instrukce, které naplní registr B hodnotou 0A3h. Zápisy pak mohou vypadat:

MVI B,0A3h

MOV B,#0A3h

LD B,0A3h

LACC #1024

podle typu procesoru.

Bezprostředním operandem je zde číslo 0A3h. Operand je zapsán v hexadecimálním kódu, o čemž informuje písmeno h za číslem. Zápis nuly před číslem je nutný pro většinu překladačů, začíná-li číslo

písmenem. Cíl dat, tedy registr B, je adresován implicitně, defacto je dán instrukcí, resp. operačním kódem. Toto adresování je též někdy nazýváno *Adresa nultého řádu*.

❑ Přímé adresování

Adresa se u tohoto zápisu nachází ihned za operačním kódem instrukce. Jako příklad uveďme instrukci přesunu obsahu paměťové buňky z adresy 2000h do střadače:

LDA 2000h,

MOV A, 2000h ... u I8086 x

LACC 2000h ... u DSP TMS 320C50

Operační paměť	
Adresa	obsah
1FFEh	345
1FFFh	12
2000h	58
2001h	10
2002h	0

Instrukce LDA 2000h	Obsah střadače (registru A)
před provedením instrukce	10
po provedení instrukce	58

❑ Registrované adresování

Instrukce se většinou skládá ze dvou částí. Z operačního znaku, který určuje co má procesor vykonávat a z adresové části, která obsahuje adresy nebo operandy nebo oboje. Některé instrukce však sestávají pouze z operačního znaku a přesto explicitně určují adresy operandů. Operandy se nacházejí v registru. Instrukce pak obsahuje název tohoto registru. Uvedený způsob adresování lze tedy také označit jako přímé registrové adresování.

MOV R0,R1

MOV A,B

❑ Nepřímé adresování

Při tomto způsobu adresování se v parametru instrukce nachází odkaz na registr nebo adresu, kde teprve je uložena adresa operandu. Nejčastěji se tento způsob vyskytuje v podobě nepřímého registrového adresování. I když se tento způsob zdá na první pohled poněkud komplikovaný, jedná se o dosti často využívaný způsob adresování. Někdy se také můžeme setkat s názvem adresa druhého řádu.

ADD A, @R0

ADD *

ADD M

Operační paměť	
Adresa	obsah
33h	345
32h	12
31h	58
30h	10
2Fh	0

	Obsah registru R0
	30h
Instrukce ADD A, @R0	Obsah střadače (registru A)
před provedením instrukce	1
po provedení instrukce	11

Jak již bylo řečeno, jedná se o často využívaný způsob adresování. Např. pouhou inkrementací registru R0 a opakováním instrukce ADD pak můžeme k akumulátoru přičíst obsah dalšího paměťového místa.

❑ Relativní adresování

Adresová část instrukce, tedy její parametr neurčuje v tomto případě přímo adresu paměti, ale představuje tzv. posunutí (relativní adresu). Úplnou absolutní adresu obdržíme teprve přičtením relativní adresy ke vztažné (bázové) adrese, která je uložena v bázovém registru. Bází může být rovněž i programový čítač a díky tomu můžeme provádět i relativní skoky. V tomto případě se jedná o tzv. autorelativní adresování.

Např. instrukce JMP +5 k obsahu programového čítače přičte 5. Toto provede skok v programu o pět adresových míst vpřed. Tato vlastnost relativního adresování umožňuje přesunout program do jiných oblastí pamětí aniž bychom museli měnit adresy u skokových a jiných instrukcí s relativním adresováním. Program psaný s takovými instrukcemi je tedy bez úprav přemístitelný.

❑ Stránkové adresování

Jedná se v podstatě o sloučení adresování přímého a relativního. Abychom mohli zkrátit relativní adresu obsaženou v instrukci, rozdělíme paměť na menší celky. Bázová adresa určuje počátek stránky a relativní adresa pak určuje pozici na stránce. Např. u šestnáctibitového adresování můžeme paměť rozdělit na osmibitové adresované stránky. Stránkový registr pak obsahuje horních 8 bitů a spodních 8 je obsaženo v parametru instrukce - přímá adresa. Běžné instrukce dovolují pohyb pouze po právě vybrané stránce, přechod na jinou stránku je proveden tzv. dlouhým skokem, kdy se mění nejen ukazatel v rámci stránky, ale i obsah stránkového registru. Tento způsob adresování zkracuje délku instrukcí.

❑ Segmentové adresování

Využívá obdobných principů výše uvedených adresování. Může vypadat tak, že všechny instrukce mikroprocesoru jsou schopny adresovat pouze část adresového prostoru, tzv. segment. Segmentových registru může být několik, jeden je určen pro program, druhý pro data, další pro zásobník a dle potřeby i další. Počátek segmentu nám pak určuje obsah odpovídajícího segmentového registru příslušného paměťového prostoru.

Relativní způsoby adresování se s výhodou používají pro práci s tabulkami nebo pro hromadné přenosy dat.

❑ Bitové adresování

Některé mikroprocesory mají podporu pro vykonávání jednobitových logických operací rozšířenu také o schopnost adresovat část paměti po jednotlivých bitech. V takovém případě hovoříme o bitovém adresování. Jde v podstatě o jistý druh přímého nebo registrového adresování s tím, že se jedná o možnost podrobnějšího adresování uvnitř paměťové buňky.

❑ Adresování dat v zásobníku

Ve strukturách mikroprocesoru se ve většině případů vyskytuje tzv. zásobníková paměť. Tato má pro činnost mikroprocesoru velký význam, protože se do ní ukládají tak důležité informace, jako jsou návratové adresy z podprogramů, stavové informace (stavové slovo - příznakový registr), střadač, bity určující výběr bank apod. O této paměti a její adresování budeme podrobněji hovořit v následující kapitole.

3.4 Přerušení

Systém přerušení zajišťuje rychlou reakci mikroprocesoru a mikropočítače na nějaký podnět buď z okolí mikropočítače, např. sepnutí havarijního spínače nebo přímo z technického vybavení mikropočítače či mikroprocesoru, např. přetečení časovače, chyba dělení nulou atd. Téměř naprostá většina mikroprocesorů je vybavena možnosti tzv. přerušení. Tato možnost podporována hardwarovým uspořádáním mikroprocesoru je pak v mikropočítačovém systému rozšířená o celý systém přerušení. Tento systém pak obsahuje ve své struktuře několik vstupů žádosti o přerušení. Na tyto vstupy mohou být připojeny nejrůznější zdroje přerušení, které mohou aktivovat žádost o přerušení v libovolných okamžicích. První činnost, kterou s těmito žádostmi systém provede, je výběr těch žádostí, které mají v daný okamžik význam. Ostatní mohou být zakázány. V mikropočítačové terminologii se tomuto výběru říká **maskování** zdrojů přerušení. Dále pak je třeba vyřešit otázku priority, to znamená v případě více žádostí o přerušení rozhodnout, která bude obsloužena jako první. Na základě toho pak vzniká žádost o přerušení právě probíhajícího programu mikroprocesoru. V případě přijetí žádosti, což ještě také může záviset na aktuálním stavu činnosti mikroprocesoru, informuje tímto signálem na řídicí sběrnici a následuje fáze identifikace zdroje přerušení, během které je stanovená adresa, na které se začne vykonávat obslužný podprogram přerušení. Tato adresa je potom načtena do programového čítače. Zde je tedy činnost původního programu přerušena a chystá se na předání řízení jinam. Je však ještě nutno provést velmi důležitou úschovu několika informací, jež nám později umožní po vykonání obslužného programu přerušení vrátit se k provádění původního programu. Ještě před provedením první instrukce obslužného podprogramu je nutno uložit obsah programového čítače, který určuje návratovou adresu. To však většinou zajišťují technické prostředky daného mikroprocesoru. Kromě obsahu programového čítače obvykle také uschováváme obsahy dalších registrů zejména těch, které využívá jak přerušovaný, tak přerušující program. Toto již však většinou provádíme instrukcemi v obslužném programu přerušení. Po vykonání vlastního algoritmu obslužného programu je pak nutné všechny uchované informace vrátit do původních prostorů (registrů). Obsah programového čítače se posléze obnovuje automaticky vykonáním instrukce návratu z přerušení.

Více žádosti o přerušení může být obsluhováno postupně tak, že žádost později aktivovaná je obsloužena až po skončení obslužného programu předchozího přerušení. Takovému zpracování přerušení říkáme **jednoúrovňová přerušení.**

Druhou možností je přerušit i obslužný podprogram přerušení. Takový případ může nastat v případě dalšího výskytu žádosti o přerušení s vyšší prioritou. Nastává pak přerušení v přerušení neboli **víceúrovňové přerušení.**

Při více zdrojích přerušení se pak celá situace může zkomplikovat neboť při častých žádostech o přerušení může dojít k zablokování takového systému anebo v lepších případech k neobsloužení přerušení s nižší prioritou. Návrh systému a software pak vyžaduje nutnost možnosti dynamického přiřazování priorit jednotlivým zdrojům.

Zbývá ještě objasnit, kam se ukládají ony informace při obsluhování jednotlivých přerušení. Pro úschovu těchto informací je téměř každý mikropočítačový systém vybaven tzv. zásobníkem. Ten je součástí buď samotného mikroprocesoru nebo paměti mikropočítače. U některých typů mikroprocesorů se dokonce vyskytují tzv. stínové registry některých důležitých registrů, do nichž se pak obsahy registrů při obsluhách přerušení ukládají automaticky.

Zásobníková paměť (stack nebo sklípek) je v podstatě paměť typu LIFO neboli paměť se sériovým přístupem. Při používání této paměti se již nestaráme o to, kam se daná data ukládají, pouze se musí zabezpečit správný sled výběru. Adresa je většinou určována obsahem registru SP (Stack Pointer), jehož počáteční obsah je také možno u některých mikropočítačů nastavit softwarově. Jeho další modifikaci již pak zabezpečují většinou instrukce ukládání do zásobníku (instrukce PUSH) a výběr ze zásobníku (instrukce POP). Vzhledem k tomu, že u většiny mikropočítačů je tento zásobník součástí běžné operační paměti RAM, je nutné hlídat nebo se na základě struktury programu ujistit, aby obsah registru SP nedosáhl dna paměti anebo aby naopak nepřetekl do té části paměti RAM, která je určena např. pro program. Celá řada mikroprocesorů opuštění tohoto prostoru nehlídá.

Shrnutí pojmů 3

Klíčová slova:
Mikroprocesor, mikropočítač, řadič, registr, přerušení, adresování

Otázky 3

1. Vysvětlete pojem mikroprocesor
2. Vysvětlete rozdíl mezi mikropočítačem a mikroprocesorem
3. Co tvoří mikropočítač
4. Vysvětlete pojmy řadič, registr, aritmeticko-logická jednotka
5. Popište základní činnost mikroprocesoru
6. Co je to instrukční soubor
7. Objasněte některé způsoby adresování
8. Vysvětlete princip přerušení

Úlohy k řešení 3

30. Základní část mikropočítače – mikroprocesor obsahuje:

a A/D převodník
b aritmeticko-logickou jednotku (ALU)
c čítač instrukcí (PC)
d univerzální čítače a časovače

31. Mikropočítač je

a programovatelná logická jednotka
b programovatelná logická jednotka využívající alespoň jeden podpůrný obvod
c programovatelná jednotka sestavena z mikroprocesoru a dalších nutných podpůrných obvodů
d programovatelná logická jednotka obsahující klávesnici a terminál pro styk s uživatelem

32. Programový čítač

a je možné modifikovat instrukcí tomu určenou
b je modifikován pouze inkrementací samotným mikroprocesorem
c je možné modifikovat programově
d je modifikován pouze mikroprocesorem při instrukci volání podprogramu

33. Adresová sběrnice mikropočítačového systému

a je obousměrná
b je jednosměrná
c slouží k přenosu signálů CS
d vyžaduje třístavovou logiku

34. Šířka vnitřní datové sběrnice mikroprocesoru

a má vliv na celkovou rychlost procesoru
b nemá žádný vliv na rychlost procesoru
c určuje délku strojového cyklu
d určuje přesnost mikroprocesoru

35. Aritmeticko logická část mikroprocesoru

a zajišťuje aritmetické a logické operace
b zajišťuje a koordinuje činnost všech částí mikroprocesoru
c tvoří paměťovou část mikroprocesoru
d podílí se na výpočtu adres

36. Šířka vnitřní sběrnice mikroprocesoru

a je stejná jako šířka vnější sběrnice mikroprocesoru
b nemusí být stejná jako šířka vnější sběrnice mikroprocesoru
c je menší než šířka vnější sběrnice mikroprocesoru
d je větší než šířka vnitřní adresové sběrnice

37. Časování mikroprocesoru

a je odvozeno z hodinových signálů oscilátoru
b má vliv na přesnost matematických operací
c zajišťuje synchronizaci mikroprocesoru
d se zajišťuje programově

38. Paměťová část mikroprocesoru je tvořena

a řadičem
b budiči sběrnice
c souborem registrů
d aritmeticko-logickou částí

39. Zásobník

a je tvořen pamětí typu LIFO a využívá se např. pro uchování návratové adresy při přerušení nebo pro uložení kontextu registrů při volání podprogramu
b je tvořen pamětí typu FIFO a využívá se např. pro uchování návratové adresy při přerušení nebo pro uložení kontextu registrů při volání podprogramu
c slouží k načítání instrukcí z programové paměti
d používá se při matematických operacích

40. Do zásobníku

a můžeme ukládat libovolná data
b procesor neumožňuje programově ukládat data
c procesor ukládá návratovou adresu při přerušení
d procesor ukládá mezivýsledky matematických operací

4. SBĚRNICE

Čas ke studiu: 1,5 hodin

Cíl Po prostudování tohoto odstavce budete umět

- vysvětlit princip činnosti společné sběrnice
- vysvětlit stav vysoké impedance
- popsat přenos informace po společné sběrnici
- druhy sběrnic v mikropočítačových systémech

Výklad

Jak již bylo zmíněno v úvodu předchozí kapitoly, jsou nedílnou součástí každého mikropočítače, ale i samotného mikroprocesoru, sběrnice. Jejich úkolem je přenášet informace mezi mikroprocesorem a ostatními částmi mikropočítače (paměti, port apod.) eventuelně mezi částmi uvnitř mikroprocesoru ale i mezi mikropočítačem a okolím. Z těchto možností využití sběrnic pak také vyplývá jedno z kritérií dělení sběrnic. Než přejdeme k samotnému dělení je nejdříve nutno si udělat představu o principu přenosu informace po sběrnici.

V běžných logických sítích je obvyklé, že výstupní signál jednoho obvodu je přiveden na jeden nebo několik vstupů dalších obvodů. Tedy jinak řečeno, k vodiči, jimž se přenáší informace, je připojen pouze jeden její zdroj (budič) a několik snímačů. V technice mikropočítačů je využit jiný princip. Všechny bloky mikropočítače jsou paralelně propojeny souborem vodičů – společnou sběrnicí (Bus). Tyto bloky však mají schopnost informaci jak snímat, tak ji na sběrnici předávat. Ke sběrnici je tedy připojeno několik zdrojů informace a aby přenos měl smysl a aby nedošlo ke kolizi, je třeba určit:

- Který blok bude informaci na sběrnici dodávat a který snímat.
- Kdy je informace na sběrnici platná

Jedním z účastníků každého přenosu na sběrnici, jak bude vysvětleno v dalších kapitolách, bývá až na výjimku samotný mikroprocesor. Ten v těchto případech rovněž určuje směr přenosu, druhého účastníka přenosu, kdy má být informace na sběrnici platná a další nutné údaje o přenosu. Z výše uvedeného pak vyplývá první z kritérií dělení sběrnic. V technice mikropočítačů tedy rozeznáváme:

- Datové sběrnice
- Adresové sběrnice
- Řídicí sběrnice

4.1 Adresová sběrnice

Pokud chce mikroprocesor číst data z nějakého bloku nebo je zapisovat, musí nějakým způsobem sdělit místo čtení i zápisu. Toto místo je identifikováno tzv. adresou, která se přenáší po adresové sběrnici. Zdrojem této informace jak již bylo řečeno je mikroprocesor.

Počet bitů adresové sběrnice (počet vodičů) odpovídá počtu bitů adresy, kterou je schopen mikroprocesor vytvořit a určuje maximální využitelný adresovatelný prostor. Např. osmibitová adresová sběrnice adresuje maximálně $2^8 = 256$ adres, šestnáctibitová adresuje 65536, tj. 64k adres.

Těchto prostorů může být ovšem více, což pak závisí na architektuře samotných mikroprocesorů. Univerzální mikroprocesory mají obvykle dva adresové prostory – pro adresování paměti a pro adresování vstupů a výstupů. Každý blok, který s mikroprocesorem komunikuje, musí být umístěn v některém z těchto dvou prostorů. Tyto dva prostory nejsou rovnocenné, rozlišení adresovatelného prostoru zajišťuje řídicí sběrnice.

4.2 Řídicí sběrnice

Je spíše souhrnem individuálních signálů aktivních v různých okamžicích s různým významem. Jednotlivé signály mají též různé zdroje. Některé jsou generovány mikroprocesorem, některé mohou být ovlivňovány jinými bloky. K jednotlivým blokům jsou pak přivedeny jen ty signály, které se jich týkají. Některé z typických signálů řídicí sběrnice jsou uvedeny níže.

RESET
Signál, kterým je vybaven každý mikroprocesor a který ho uvádí do základního (výchozího) stavu. Aktivuje ho buď uživatel nebo je generován přídavným obvodem např. při zapnutí napájení. Tímto signálem mohou být uváděny do základního stavu i jiné bloky mikropočítače.

MR (RD) Memory Read (Read) – signál čtení z paměti (čtení z bloků)
Signály zabezpečující časování přenosu informací z pamětí či jiných blíže nespecifikovaných bloků do mikroprocesoru nebo mikropočítače.

MW (WR, WE) Memory Write (Write, Write Enable) – zápis do paměti (zápis do bloků)
Signály zabezpečující časování přenosu informací do pamětí či jiných blíže nespecifikovaných bloků z mikroprocesoru nebo mikropočítače.

IOW/IOR – Input-Output Write/Read – zápis/čtení do/z výstupu
Některé architektury mikroprocesorů či mikropočítačů podporují oddělené instrukce pro zápis do periferních bloků. Pak jejich řídicí sběrnice obsahují signály IOW/IOR pro časování přenosu do/z těchto bloků.

READY – připravenost obvodu
Někdy je nutno k mikroprocesoru připojit obvody, které s hlediska svého časování nevyhovují a nestíhají požadovanou činnost. V takovém případě musí být možnost pozastavit činnost mikroprocesoru například tak, že před svou další činností vloží tzv. čekací smyčky (Wait stavy) a např. při čtení z paměti si na platná data počká. Musí být však informován o žádosti pozastavení právě signálem READY.

Řídicí sběrnice může dále obsahovat alespoň jeden hodinový signál pro potřeby jednotlivých bloků. Uvedené řídicí signály se u různých mikroprocesorů liší.

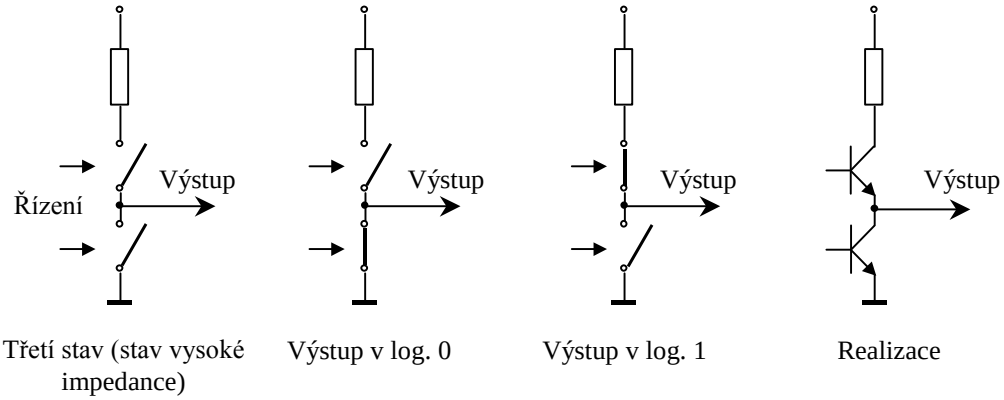
4.3 Datová sběrnice

Po této sběrnici probíhá přenos všech dat v mikropočítači. Data se přenášejí vždy mezi dvěma bloky mikropočítače – například z paměti do mikroprocesoru a podobně. Jak již bylo řečeno účastní se mikroprocesor až na výjimku jako přijímač a vysílač všech přenosů v mikropočítači. Záměrně jsme popis této sběrnice ponechali až na konec tohoto kritéria dělení sběrnic, neboť je nutné si nyní vysvětlit princip obousměrného a několika blokového přenosu informací po této sběrnici.

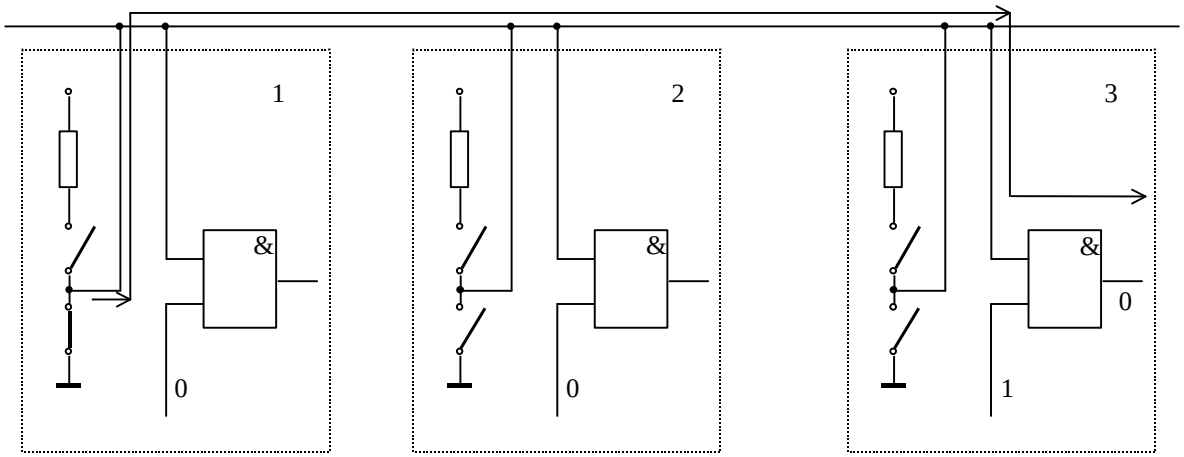
Z výše uvedeného vyplývá, že při této úvaze je nutné, aby v jakémkoliv okamžiku byl aktivní pouze jeden vysílač, jinak řečeno budič sběrnice. Při nedodržení této podmínky by na datové sběrnici došlo k neurčitosti signálu, v horším případě pak ke zničení jednoho nebo obou vysílacích obvodů. Proto je tedy nutné vybavit bloky připojované na datovou sběrnici obvody, jež umožní odpojení tohoto bloku od datové sběrnice, je-li to s hlediska přenosu informace (tj. když se zrovna neúčastní přenosu) nutné. Takové obvody nazýváme třístavovými budiči sběrnice. Jejich princip a realizace je na obr. 4.1. Příklad jednoho vodiče obousměrné datové sběrnice s třístavovými budiči je na obr. 4.2.

V okamžiku zachyceném na tomto obrázku probíhá přenos logické nuly z bloku 1 do bloku 3, ostatní budiče i snímače jsou odpojeny. Obrázek 4.3. zachycuje dvě obvyklá označení třístavových budičů.

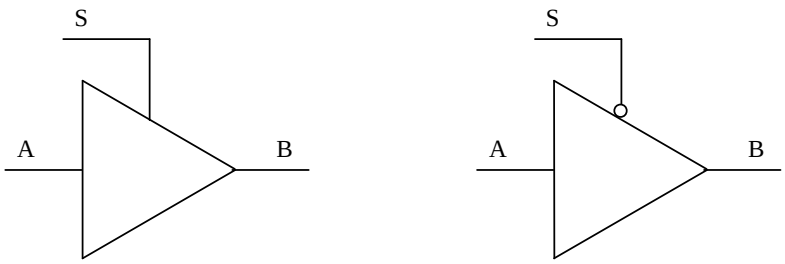
Ovládání třetího stavu nějakého bloku vybaveného třístavovým budičem je provedeno signálem CS (CE) Chip Select (Chip Enable).



Obr. 4.1 Obousměrná datová sběrnice



Obr. 4.2 Příklad vedení v jednom směru



Obr. 4.3 Značení třístavových budičů

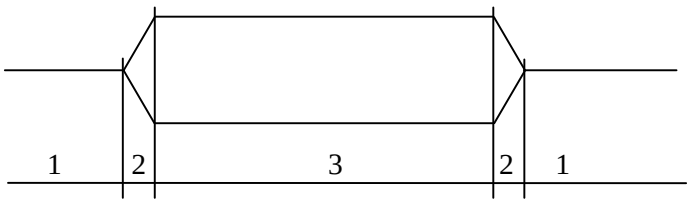
Důležitým parametrem datové sběrnice je její počet bitů (její tzv. šířka). Tento parametr má vliv na rychlost komunikace, neboť udává, kolik bitů se přenáší najednou. Nemá však smysl, aby tento počet byl větší, než kolikabitový je mikroprocesor.

V souvislosti s šířkou adresové a datové sběrnice se nabízí, že celkový počet bitů paměťového prostoru (paměti) získáme vynásobením obou šířek. U některých mikroprocesorů to však není pravda. Paměťové buňky zůstávají osmibitové i u šesnáctibitových mikroprocesorů (mikropočítačů). Je tak umožněn přístup i k menším jednotkám. Tak je možno šetřit paměť pro „kratší“ data, ale významnějším důvodem je kompatibilita vyšších typů mikroprocesorů s nižšími při změně počtů bitů.

Maximální délka sběrnic je omezena bezproblémovým přenosem signálů při dané frekvenci přenosu a nepříznivých vlivech jako je vzájemná kapacita vodičů, parazitní indukčnosti atd. Bývá nejvýše desítky centimetrů.

V některých případech je vhodné šetřit počet vodičů a proto se provádí tzv. multiplexování sběrnic. Signály dvou sběrnic (nejčastěji datové a adresové) jsou vedeny ve společných vodičích tak, že jsou aktivní v různou dobu. Jaký význam mají právě přenášená data na multiplexované sběrnici se pozná podle signálů řídicí sběrnice. U některých mikroprocesorů bývá počet vodičů (bitů) datové sběrnice nižší než adresové, čímž se mohou multiplexovat jen dolní bity adresové sběrnice s datovými.

Další důležitou úlohu při přenosu informace po sběrnicích má tzv. časování. Z těchto důvodů každý výrobce mikroprocesorových komponent uvádí v manuálech časovací diagramy. Tyto diagramy jsou také velmi důležité pro návrhy a realizace mikroprocesorových řídicích systémů. Používané značení v těchto diagramech pak vysvětluje obr. 4.4.



Obr. 4.4 Časovací diagram

Úseky 1 informují o stavu vysoké impedance, tedy že všechny budiče jsou odpojené od sběrnice. V úsecích 2 se informace na vodičích mění. V úseku 3 jsou pak data na sběrnici platná, informace je ustálená.

Třístavové budiče sběrnice mohou být tedy jednosměrné nebo obousměrné. Tyto budiče mají tedy ještě kromě schopnosti oddělovat od sběrnice také důležitou úlohu spočívající ve výkonovém posílení. Obvykle je těmito budiči již vybavována naprostá většina pamětí, obvody vstupu a výstupu, řadičů přerušení, čítačů apod. Tam, kde implementovaný budič nestačí pro připojený počet bloků, posiluje se sběrnice vložením dalších budičů. Je ovšem nutné mít na paměti, že každé vložení jakéhokoliv prvku do cesty signálu způsobuje jeho zpoždění dané zpožděním hradel v signálové cestě a možný výskyt problémů při časování sběrnic.

Další kritérium dělení sběrnic pak souvisí s architekturou mikropočítačových řídicích systémů. Rozeznáváme pak:

- Vnitřní sběrnice mikroprocesoru
- Vnitřní sběrnice mikropočítače
- Vnější sběrnice mikropočítače

Mikroprocesor můžeme v prvém přiblížení chápat jako černou skříňku se známým vnějším chováním vzhledem k signálům a instrukcím programu a jeho **vnitřní sběrnice mikroprocesoru** nás téměř nebude zajímat. Její architektura je dána převážně architekturou mikroprocesoru.

Vnitřní sběrnice mikropočítače nám pak propojuje mikroprocesor s ostatními prvky mikropočítače a dovoluje nám tedy vytvářet různá uspořádání mikropočítače, optimální cenově a funkčně pro danou aplikaci.

Vnější sběrnice mikropočítače nebo také sběrnice styku s okolím či sběrnice multiprocesorového systému má řadu zvláštností, vyplývající z charakteru okolí. Ve složitějších systémech, sestavených z několika mikropočítačů, je okolím jednoho mikropočítače jiný mikropočítač. Sběrnice jsou pro takový účel poněkud rozšířeny a upraveny, neboť je třeba navíc rozhodovat o prioritě podřízených mikropočítačů. Priorita se může v průběhu algoritmu dynamicky měnit podle okamžitých potřeb systému. Jindy se pomocí vnější sběrnice mikropočítače připojují k danému mikropočítači další přídavná zařízení. V malých systémech jsou tato zařízení připojována ke společně sdílené vnitřní sběrnici mikropočítače, ve velkých systémech jsou přídavná zařízení připojována k vnější sběrnici prostřednictvím řadiče této sběrnice. (Sběrnice MULTIBUS, ISA atp.)

Pozn.:
S rozvojem počítače se standardů vystřídalo několik, některé byly tak úspěšné, že ještě dnes se přes všechny své neřesti používají, na jiné se velmi rychle a ochotně zapomnělo.

ISA
Ve zkratce z „industry standard architecture“ představuje nejstarší ze standardů, datovaných z dob počítačového středověku – procesoru 286. Frekvence, na kterém sběrnice pracuje dělá o něco více než 8 MHz. Data mohou v šířce šestnácti bitů. Typická přenosová rychlost dělá asi 5 MB/s. ISA je zatím nejslavnější ze sběrnic, neboť i dnes, po více než desítce let se stále slot ISA na základních deskách nachází. Důvodů je více, ale mezi ty hlavní asi patří skutečnost ze karet pracujících na principech ISA je sále moc, a ISA je na výrobu nejlevnější ze sběrnic.

EISA (extended ISA) je prvním rozšířením ISA. Rozeznat od ISY lze tak, že uvnitř jejího konektoru jsou dvě řady kontaktů, nikoli jediný jako u ISY. Důvod je jednoduchý: do EISA můžeme umístit kartu ISA. Přenosová rychlost se zvýšila na 33 Mb/s a šířka dat na 32 bitů. Frekvence musela zůstat kvůli kompatibilitě s ISOU stejná. VESA neboli „video equipment standards association“ byla zkonstruována za éry procesoru 486. Jedná se spíše o další rozšíření ISY, protože VESA je umístěna půl centimetru za ISOU a karta pracující na principu ISA tedy může využívat služeb sběrnice VESA. K tomu musí být karta o něco delší. Frekvenci VESY lze nastavit v rozsazích 25 – 50 MHz.

PCI
Byl a dosud je převratnou novinkou v oblasti systémové sběrnice. Frekvence práce PCI sběrnice se pohybuje v intervalu 25 – 33 MHz. Šířka přenášených dat se vyšplhala na 64 bitů, a typická přenosová rychlost měří 132 Mb/s. Zdaleka největší novinkou je konfigurace typu „Plug & Play“, kdy uživatel nemusí zadávat (jako např. u ISY) adresu karty, volné IRQ nebo DMA. Pro instalaci karty PCI ji stačí zastrčit do (vypnutého!) počítače a systém by měl zkonfigurovat zbytek automaticky.

AGP
Pochází stejně jako PCI z dílny Intelu. Jedná se o speciální „klon“ PCI, jíž mohou proudit pouze data pro grafický adaptér. AGP (accelerated graphic port) je přímo propojen s pamětí, takže data nemusí proudit přes systémovou sběrnici. Tento způsob může přenos dat zvýšit více než dvojnásobně. Typická přenosová rychlost AGP je 264 Mb/s. Vylepšením AGP (postupně 2x a 4x – stejně jako u cdromu) se přenosová rychlost stále zvyšuje.

PCMCIA
„Personal computer memory card association“ - Byla navržena hlavně pro notebooky. Jedná se o velmi malý externí konektor, do kterého se zasunují externí rozšiřovací karty. Její princip je podobný s ISA, až na tom, že vlastní rozšíření se mohou instalovat i za běhu počítače, takže sběrnice potřebuje i speciální software.

USB
Neboli Universal Serial Bus někdy z legrace překládaná jako univerzální sériová budoucnost, je poměrně novým konektorem, který přinesl spoustu změn. Například se přes jediný konektor může připojit 127 různých zařízení, ta je možno připojovat a odpojovat za chodu počítače, a použít bez restartu systému. Napájení pro zařízení jsou integrovány přímo v USB kabelu, což velmi „zelegantní“ vzhled PC.

Závěrem
Oblast systémové sběrnice byla delší dobu přehlížena, a tak se zdá, že si to hardwarové vývojové týmy v poslední době vynahrazují. Např. relativně nové USB už utrpělo upgrade na USB 2.0 jehož rychlost je nyní srovnatelná s rozhraním FireWire. Nedovoluji si odhadnout kam až by mohl vývoj v tomto směru jít, ale přihlédneme-li k tomu, že ještě na kartách ISA bylo zapotřebí přepínat switche, a dnes stačí za běhu sytému připojit třeba skener, snad se dočkáme třeba doby, kdy nebude nic výjimečného že nebudeme muset pro konkrétní zařízení instalovat ovladače, ale systém si je jednoduše přímo stáhne z instalovaného zařízení.

Shrnutí pojmů 4

Klíčová slova:
Sběrnice, Stav vysoké impedance – třetí stav

Otázky 4

1. Vysvětlete pojem společná sběrnice
2. Vysvětlete princip přenosu dat po sběrnici.
3. Čím je charakterizován stav vysoké impedance
4. Vyjmenujte některé signály řídicí sběrnice

Úlohy k řešení 4

41. Třístavové budiče sběrnice

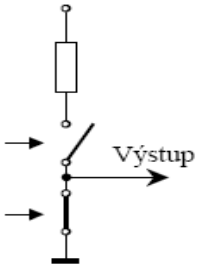
a umožňují multiplexování signálů
b jsou zpravidla obousměrné
c umožňují napojení více obvodů k jedné datové sběrnici
d slouží ke galvanickému oddělení obvodů

42. Obvody typu LATCH

a se využívají pro zachycení dat např. u multiplexované sběrnice
b vytvářejí signály pro sériovou komunikace
c tvoří základ převodníků paralelní komunikace na sériovou
d jsou nutné pro připojení rozhraní RS 232

43. Zapojení na obrázku u datové sběrnice představuje :

a stav vysoké impedance
b výstup v logické 1
c výstup v logické 0
d neurčitou logickou úroveň



44. Vnitřní datová sběrnice procesoru:

a využívá sériového přenosu
b využívá paralelního přenosu
c je obousměrná
d neumožňuje multiplexování

45. Osmibitová adresová sběrnice umožňuje adresovat:

a 8 adres
b 256 adres
c 65536 adres
d neomezený počet adres

46. Třístavová sběrnice:

a třetí stav má neurčitou logickou úroveň
b neexistuje
c třetí stav odpovídá vysoké impedanci
d třetí stav přepíná sběrnici mezi vstupní a výstupní

47. Pomocí kterého obvodu lze ošetřit multiplexovanou sběrnici:

a budičem sběrnice
b obvodem s otevřeným kolektorem
c obvodem LATCH
d Schmittovým klopným obvodem

48. Posilovač sběrnice 74LS245 se používá k:

a oddělení datové sběrnice od adresové
b jednoduché konstrukci oscilátoru
c oddělení sběrnic a zvýšení zatížitelnosti
d zachycení adresy na multiplexované datové sběrnici

49. Pojem multiplexovaná sběrnice v mikropočítačové technice znamená

a využití společných vývodů procesoru pro datovou a adresovou sběrnici
b využití společných vývodů procesoru pro řídicí a adresovou sběrnici
c využití společných vývodů procesoru pro datovou a řídicí sběrnici
d využití společných vývodů procesoru pro řídicí, datovou a adresovou sběrnici

5. ADRESNÍ DEKODÉRY

Čas ke studiu: 2 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- vysvětlit pojem adresní dekodér
- popsat princip a činnost úplného a neúplného dekodéru adres
- vysvětlit pojem mapování periferií
- vysvětlit princip univerzálního dekodéru

Výklad

Abychom mohli vysvětlit princip adresních dekodérů nutných k návrhu většiny řídicích mikropočítačových systémů, bude nutno několika větami objasnit chování a činnost mikroprocesoru při přenosech dat na sběrnicích, tj. přenosech mezi registry a pamětmi a při I/O operacích.
Přenosy informací na sběrnicích se mohou uskutečnit programovým řízením a nebo řízením technickými prostředky.

Řízení přenosu technickými prostředky se využívá pro přenosy velkých bloků dat, např. pro přenos dat z přídavných zařízení do paměti RAM. Na tomto přenosu informace se podílí mikroprocesor jen částečně nebo je úplně vyřazen. Takový přenos je řízen řadičem přímého přístupu do paměti (DMA). Problematice DMA je věnována samostatná kapitola.

Při přenosu dat s programovým řízením se na tomto přenosu podílí výhradně mikroprocesor tím, že postupně provádí instrukce účelně seřazené do segmentu řídicího programu. Výhodou je levná realizace a snadná změna algoritmu přenosu změnou příslušného segmentu řídicího programu. Nevýhodou je pak u pomalejších mikroprocesorů celkově pomalý přenos a tudíž jej lze použít jen pro pomalá přídavná vnější zařízení.

Přenos dat s programovým řízením je obecně přenosem mezi dvěma registry. U mikroprocesoru probíhá velmi často mezi některými vnitřními registry, případně mezi vnitřním registrem a buňkou vnější paměti (tj. v podstatě také registrem). Z pohledu instrukčního souboru můžeme přenosy mezi vnitřními a vnějšími registry rozdělit na přenosy paměťové a I/O. To znamená, že umožňuje-li architektura mikroprocesoru oba tyto přenosy, mívají pak tyto dva adresové prostory – pro adresování paměti a pro adresování vstupů a výstupů. Výběr prostoru je pak určen příslušnými instrukcemi pro daný prostor. U mikropočítačů vybavených mikroprocesory s oběma těmito adresovými prostory pak záleží na hardwareovém návrhu, zda periferní zařízení umístíme do prostoru vstupů a výstupů (I/O) nebo do prostoru paměťového. Pak se ovšem o tento prostor dělí s fyzickými pamětmi. Tato nevýhoda je však zanedbatelná, neboť většina periferií resp. jejich záchytné registry zabírají jen minimální adresový prostor.

Umístění periferních zařízení do některého z těchto prostorů se nazývá mapování. Výhoda mapování periferií do I/O prostoru spočívá v rychlejším přístupu do tohoto prostoru, zpravidla instrukcemi IN a OUT. Výhoda mapování do paměťového prostoru pak ve větším množstvím použitelných instrukcí pro přenos dat (k dispozici jsou všechny instrukce používané pro práci s pamětmi). U mikropočítačů, které jsou osazeny mikroprocesory pouze s paměťovým adresovým prostorem, je nutno periferní zařízení namapovat vždy do paměťového prostoru. U takového konkrétního mikropočítače nás pak zajímá obsazení paměťového prostoru, jež je možno znázornit tzv. mapou paměti, do které se

zakresluje obsazení dílčích úseků paměťového prostoru jednotlivými pamětmi, které úseky jsou neobsazeny, kde jsou adresovány periferní zařízení atd.

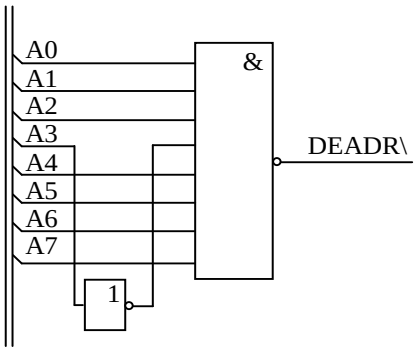
Nyní se dostáváme k samotným adresovým dekodérům. Vzhledem k tomu, že paměťový prostor mikroprocesoru bývá obsazen více než jednou fyzickou pamětí eventuelně periferními zařízeními, je nutné při přenosech dat s mikroprocesorem rozhodnout, které zařízení je ke komunikaci určeno. Tento úkol plní právě adresový dekodér. Jeho výstupy jsou v podstatě signály Chip Select (CS) pro jednotlivé obvody. Signál CS, jak již víme z předchozí kapitoly, připojuje daný obvod k datové sběrnici tak, že jeho sběrnici přepne ze stavu vysoké impedance do aktivního stavu.

Adresový dekodér může být stavěn jako dekodér pro:

- úplné dekódování adresy
- neúplné dekódování adresy
- lineární přiřazení adresy
- univerzální přiřazení dekódovaných adres

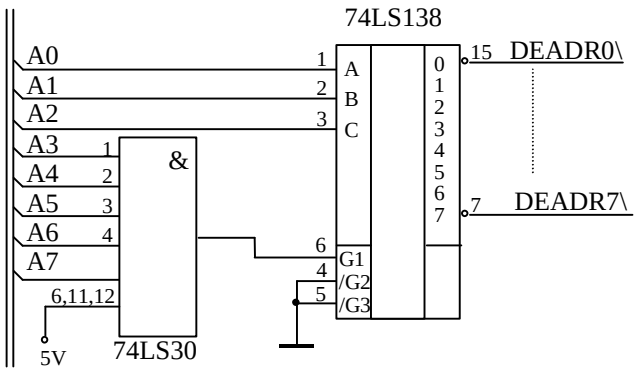
5.1 Dekodér pro úplné dekódování adresy

Při úplném dekódování je jisté unikátní adrese ADRi přiřazen pouze jeden signál DEADRi a obráceně: jistému signálu DEADRi odpovídá pouze jedna adresa ADRi.
Pro jednoduchost zapojení budeme u následujících schémat předpokládat šířku adresové sběrnice 8 bitů.



Obr. 5.1 Úplné dekódování adres

Na obr. 5.1 je zapojení adresového dekodéru s úplným dekódováním adresy ADR1 = 11101111 =0Efh. Dekodér je jednoduchý, ale dokážeme ihned posoudit, jak rychle by narůstal počet IO pro větší počet adres. V takovém případě dáváme přednost některému z vyráběných dekodérů typu „1 z N“, např. 74LS138. Na obr. 5.2 je příklad zapojení dekodéru s úplným dekódováním adresy s obvodem 74138.

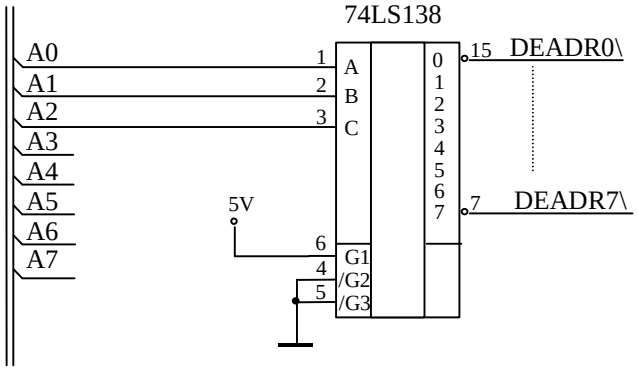


Obr. 5.2 Dekodér adresy

Pro tento dekodér platí následující přiřazení:
DEADR0 = 0F8h,
DEADR1 = 0F9h,
.
.
.
DEADR7 = 0Ffh,
tedy opět jedinečné adresy.

5.2 Adresový dekodér s neúplným dekódováním adres

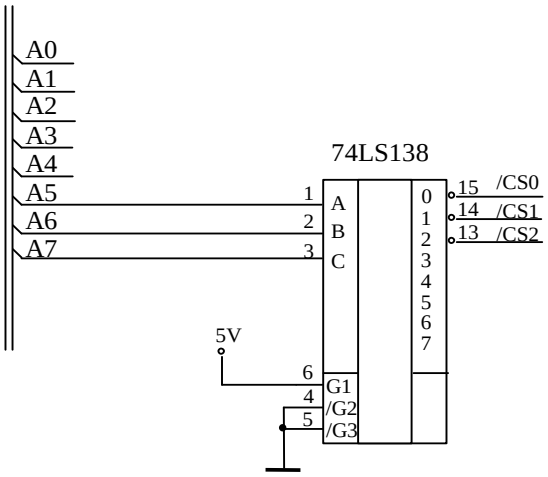
Při neúplném dekódování nejsou uvažovány některé řády adresy s tím, že jistému signálu DEADRi přísluší adresový prostor ADRk , kde k = 1, 2, 3.... Takto postavený adresový dekodér je levnější, ale vyžaduje pozorné přiřazování adres k jednotlivým adresovaným obvodům. Vizuálně dvě zcela odlišné adresy v programu mohou ve skutečnosti adresovat tentýž obvod. Na obr. 5.3 je zapojení adresového dekodéru s neúplně dekódovanou adresou. Výstupní signál DEADR0 je generován pro každou adresu mající AB0 až AB2 = 0. Například adresy 00h, 08h, 10h atd. jsou rovnocenné adresy a patří do téhož adresového prostoru. Je tedy lhostejné, kterou z nich použijeme v programu, výsledkem bude výběr stejného obvodu.



Obr. 5.3 Neúplné dekódování adresy

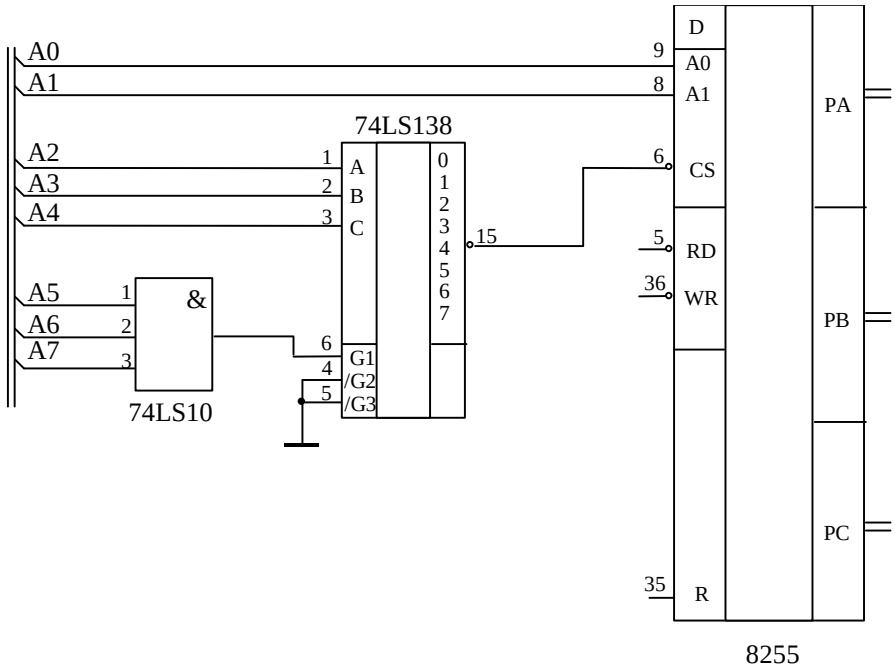
Další možnost zapojení dekodéru s neúplným adresováním je na obr. 5.4. Výběrové signály určují následující oblasti adresového prostoru:

$\overline{CS0}$ = 00h ÷ 1Fh
$\overline{CS1}$ = 20h ÷ 2Fh
$\overline{CS2}$ = 30h ÷ 3Fh



Obr. 5.4 Neúplné adresování – další řešení

Tyto dekodéry se velmi často využívají pro přidělování paměťových prostorů pro jednotlivé typy pamětí (EPROM. RAM, FLASH atd.) eventuélně přídavných periférií v mikropočítačovém systému. Některé složitější podpůrné vstupně-výstupní obvody, zejména programově nastavovatelné, mají několik vnitřních registrů a vestavěný adresový dekodér s úplným dekódováním adresy. V takovém případě můžeme úplného dekódování adres dosáhnout i s vnějším neúplným dekodérem tím způsobem, že neúplně dekódujeme adresový prostor s velikostí úměrnou prostoru vnitřního dekodéru. Na obr. 5.5 je uveden jeden takový příklad úplného dekódování adres pro podpůrný integrovaný obvod pro sériový vstup-výstup, s nímž se seznámíme v další kapitole. Vnější dekodér s integrovaným obvodem 74LS138 je neúplný, neboť neuvažuje adresní bity AB0 a AB1.



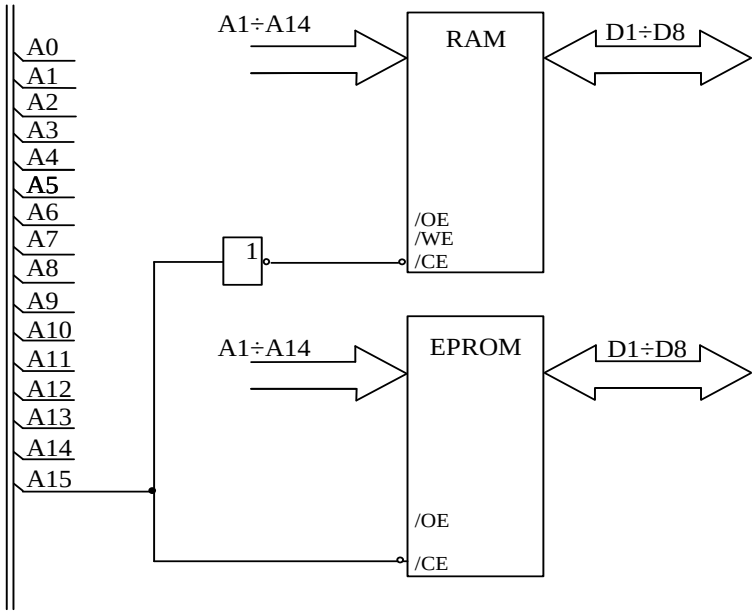
Obr. 5.5 Závislé dekódování adresy

5.3 Adresový dekodér s lineárním přiřazením adresy

U těchto dekodérů jsou jednotlivé řády adresy pokládány přímo za výstupní výběrové signály dekodéru. Platí:

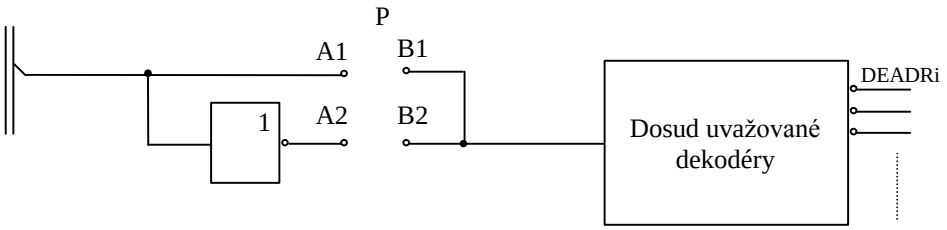
DEADRi = ADRi

a žádný adresový dekodér není třeba realizovat. je to tedy nejlevnější adresový dekodér. Jeho nevýhodou je možnost připojit pouze m přídavných zařízení u adresy s m řády. Jeden z velmi často využívaných dekodérů s lineárním přiřazením adresy je na obr. 5.6. U něj je využit 15 bit adresy k rozdělení 64 kB adresového prostoru na dvě poloviny. Nejedná se o lineární přiřazení pro oba tyto prostory, ale jen pro prostor 0h až 7FFFh. Dekódování druhé poloviny pak vyžaduje použití invertoru 74LS04. Paměti mohou být např. typu 27C256 a 62256.



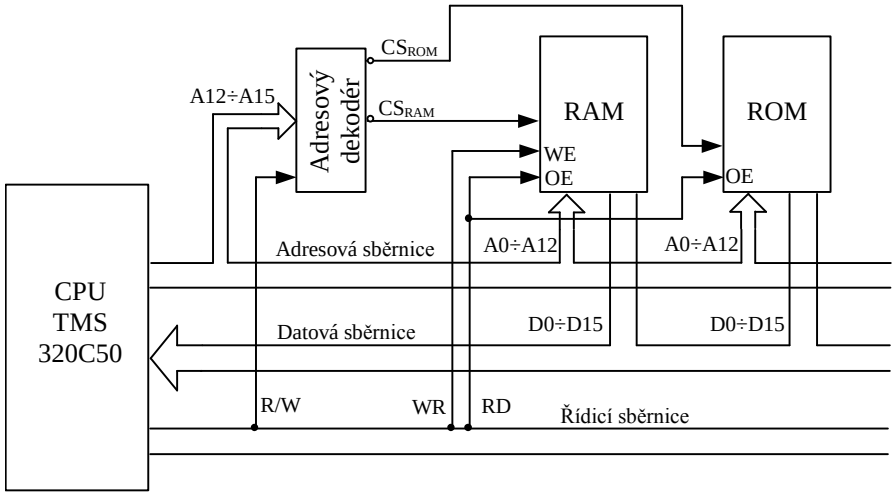
Obr. 5.6 Příklad využití lineárního adresového dekodéru (viz text)

V praxi používané adresové dekodéry jsou ve většině případů kombinacemi zde uvedených typů dekodérů. Jinak se sestaví dekodér pro předem známou aplikaci s konečnou více nerozšiřovanou sestavou mikropočítače a jinak pro předem neznámé aplikace s požadavkem modulové rozšiřitelnosti mikropočítače – v takovém případě jde o maximální univerzálnost. Univerzální řešení je zpravidla složitější a dražší. Jeden z takových univerzálních adresových dekodérů je na obr. 5.7. V každé adresní lince Ai je vložen invertor a vhodně konstruovaným přepínačem P (např. DIP) můžeme libovolně zvolit skutečnou vstupní adresu. Vzhledem k tomu., že toto nastavování většinou probíhá jen před zasunutím modulu do rozšiřovaného mikropočítače, je možné přepínač adres nahradit pájenými propojkami.



Obr. 5.7 Princip univerzálního dekodéru adresy

Na obr 5.8 je pak vysvětlen princip využití adresového dekodéru pro adresaci pamětí v jednoduchém mikropočítačovém systému s časováním jednotlivých signálů při čtení i zápisu z/do paměti.

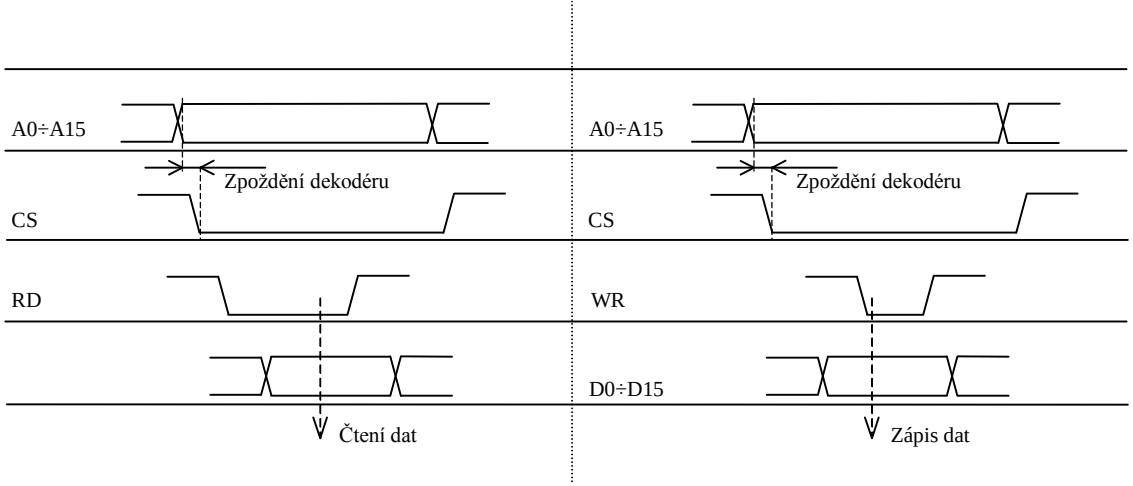


Obr. 5.8 Princip využití dekodéru

Jako adresový dekodér může být použit jeden s výše uvedených adresových dekodérů. V tomto systému je společná datová i programová paměťová část, jedná se tedy o strukturu Von Neumann. Paměti se zde dělí o celkový paměťový prostor nabízený mikroprocesorem. Další signály /CS by mohly být použity k adresování přídavných periferií, jež by také využívaly společný paměťový prostor.

Na dalších obrázcích (5.9, 5.10) je znázorněno časování jednotlivých signálů při čtení resp. zápisu z/do paměti. Při čtení např. z paměti ROM je nejprve generována adresa jednotlivých dat. Z této adresy dekóduje adresový dekodér s určitým zpožděním signál /CS. Toto zpoždění pak závisí na zpoždění jednotlivých komponentů, ze kterých je daný dekodér sestaven. Dále následuje signál RD generovaný mikroprocesorem. Po určité době závisející na přístupové době paměti ROM se na datové sběrnici objeví data.

U zápisu do paměti je časování obdobné jen s vyjímkou signálu WR, který mikroprocesor aktivuje až po vystavení dat na datovou sběrnici. Je nutné si uvědomit, že sled signálů generovaných mikroprocesorem je dán výrobní technologií daného mikroprocesoru a tudíž je potřeba osazovat adresové dekodéry součástkami s rychlostmi, odpovídající časovým diagramům jednotlivých mikroprocesorů. Tato skutečnost se někdy stává zdrojem obtíží při návrhu mikropočítačových systémů. Některé mikroprocesory jsou vybaveny možností prodlužovat časový sled těchto signálů generováním tzv. wait stavů. Využití této možnosti při návrhu pak ale způsobí zpomalení celkové činnost daného systému.



Obr. 5.9 Čtení z paměti

Obr. 5.10 Zápis do paměti

Shrnutí pojmů 5

Klíčová slova:
Adresový dekodér, mapování periferií a paměti, „wait“ stavy

Otázky 5

1. Vysvětlete pojem adresový dekodér
2. Popište možnosti připojení periferních zařízení k mikroprocesoru.
3. Vyjmenujte druhy adresových dekodérů
4. K čemu je využíván signál „Chip Select“

Úlohy k řešení 5

50. Signál chip select (CS):

a slouží k aktivaci komunikace obvodů na sběrnici
b slouží k přechodu do režimu spánku
c většinou bývá aktivní v log.0
d je vyhodnocován adresovým dekodérem

51. Neúplné adresování u adresových dekodérů znamená

a přiřazení několika adresám pouze jeden signál
b přiřazení jedné adrese pouze jeden signál
c přiřazení jedné adrese několik signálů
d přiřazení jednotlivým adresovým bitům jednotlivé signály

52. Adresování s lineárním přiřazením adres znamená

a přiřazení několika adresám pouze jeden signál
b přiřazení jedné adrese pouze jeden signál
c přiřazení jedné adrese několik signálů
d přiřazení jednotlivým adresovým bitům jednotlivé signály

53. Úplné adresování u adresových dekodérů znamená:

a přiřazení několika adresám pouze jeden signál
b přiřazení jedné adrese pouze jeden signál
c přiřazení jedné adrese několik signálů
d přiřazení jednotlivým adresovým bitům jednotlivé signály

54. Pomocí kterého obvodu lze ošetřit multiplexovanou sběrnici:

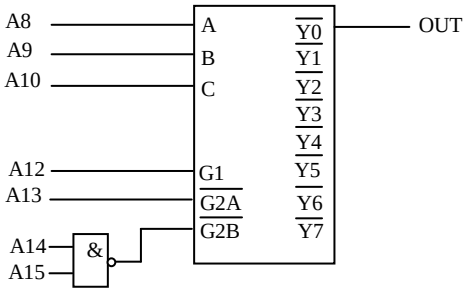
a budičem sběrnice
b obvodem s otevřeným kolektorem
c obvodem LATCH
d Schmittovým klopným obvodem

55. Posilovač sběrnice 74LS245 se používá k:

a oddělení datové sběrnice od adresové
b jednoduché konstrukci oscilátoru
c oddělení sběrnic a zvýšení zatížitelnosti
d zachycení adresy na multiplexované datové sběrnici

56. Jakou paměťovou oblast vybírá adresový dekodér uvedený na obrázku

a. D000÷D0FF + D800÷D8FF
b. D8FF÷DFFF
c. 0200÷D8FF
d. D000÷D8FF

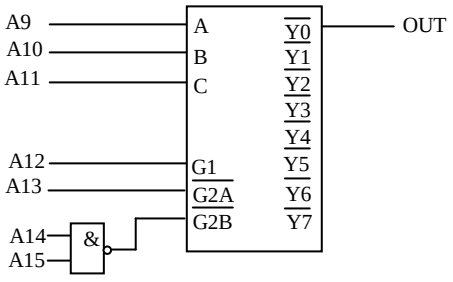


57. Který způsob styku periferie s mikroprocesorem je výhodnější z hlediska využitelného počtu instrukcí pro práci s periferiemi

a klasický, s využitím I/O prostoru
b mapovaný do paměťového prostoru
c s použitím I/O bran
d všechny výše uvedené jsou stejně výhodné

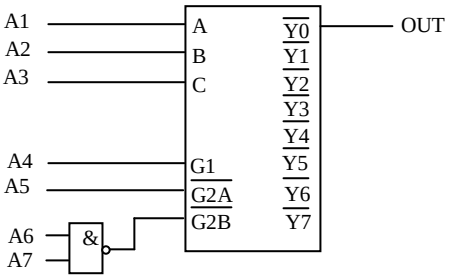
58. Jakou paměťovou oblast vybírá adresový dekodér uvedený na obrázku

a. 0200÷02FF
b. D000÷D1FF
c. DE00÷DFFF
d. 2200÷22FF



59. Jakou paměťovou oblast vybírá adresový dekodér uvedený na obrázku

a. D000÷D0FF
b. D0÷D1
c. 00÷01
d. D7÷DF



6. VZÁJEMNÝ STYK MIKROPOČÍTAČE S PERIFERIEMI

Čas ke studiu: 1,5 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- vysvětlit princip řízení komunikace
- popsat princip a činnost řadiče přerušení
- vysvětlit pojem periodické testování

Výklad

Má-li mít mikropočítač v určité aplikaci praktický význam, musí k němu být připojena periferní zařízení, umožňující styk mikropočítače s okolním prostředím. Vzhledem k mikropočítači je součástí okolního prostředí jednak člověk, jednak vlastní fyzikální okolí, se kterým má mikropočítač spolupracovat. Mezi nejběžnější periferie, umožňující styk počítače s člověkem můžeme zařadit například alfanumerickou klávesnici, obrazovkový displej, tiskárnu, ale i specializovanou tlačítkovou klávesnici určitého přístroje řízeného mikropočítačem, či jeho displej.

Další, největší skupinu, tvoří periferní zařízení umožňující připojení mikropočítače na řízený proces nebo na zkoumané prostředí. Zde patří všechny druhy A/D a D/A převodníků, zesilovače, čítače, tvarovače a mnoho dalších zařízení sloužící k rozmanité transformaci veličin.

Tato zařízení jsou nejčastěji umístěna vně mikropočítače, některé ovšem mohou být tvořeny i vlastními obvody mikropočítače. Jejich připojení a komunikaci řeší uživatel mikropočítače pro každý konkrétní případ. Tyto periférie se obvykle připojují k mikropočítači prostřednictvím tzv. stykových obvodů. Stykový obvod může být rovněž součástí daného periferního zařízení a musí zejména zajišťovat:

- dekódování instrukcí pro styk s okolím a řízení odpovídajících dílčích činností v potřebných časových souvislostech
- dekódování adresy jisté části okolí, se kterou právě probíhá styk mikropočítače
- dočasné uchování informace ve vyrovnávací paměti
- konverzi informace pro účely styku

6.1 Řízení komunikace

Existují dva případy zahájení komunikace mikropočítače nebo mikroprocesoru s periferiemi:

Zahájení komunikace z iniciativy programu
Toto je případ, kdy příkaz k zahájení komunikace je v podstatě součástí instrukcí probíhajícího programu, např. když algoritmus programu potřebuje informace z okolí, eventuelně chce vyslat nějaký výsledek do okolí. Zde ovšem musí být zajištěno, že daná periferie je schopna přijmout informaci v daný okamžik anebo musí informaci v okamžik určený mikropočítačem vystavit. Jako příklad těchto periferií si můžeme představit stavové spínače nebo sdělovače (signalizační zařízení).

Zahájení komunikace z iniciativy periférie
Nastává v případě, že jistá periferie chce poslat svou informaci do mikropočítače nebo naopak pro svou činnost potřebuje informaci z mikropočítače. Zde nastává však problém v navázání komunikace,

neboť se mikropočítač může nacházet v činnosti, kdy ihned nemůže s periferií komunikovat. Mikropočítač se však nějakým způsobem musí dozvědět, že je vyžadována komunikace a která periferie je iniciátorem výzvy.

Pokud tedy komunikace je aktivovaná nikoliv mikropočítačem nýbrž periferií, lze postupovat v zásadě čtyřmi způsoby:

Obvodové řešení daného periferního zařízení je voleno tak, že určitou operaci může uskutečnit, aniž by o tom musel vlastní mikropočítač vědět. Toto řešení ovšem jde přímo proti filozofii „minimum obvodového řešení – maximum programového zabezpečení“, kterou se v některých případech, když je to možné, snažíme v technice mikropočítačů řídit. Nicméně, pokud operace vyžaduje rychlejší odezvu než můžeme zajistit některým ze tří následujících způsobů řešení, stejně nám nezbývá často nic jiného, než použít přídavného obvodového řešení pomocí obvodů nízké a střední integrace.

Zařízení které chce inicializovat nějakou operaci, použije příznakový bit (Flag) nebo skupinu příznakových bitů. Mikropočítač pak může programově hodnotu příznakového bitu číst a testovat a určitou operaci inicializovat pouze, když je jeho hodnota „1“. Pomocí příznakových bitů oznamuje vlastně periferní zařízení svůj stav (připraven, zaneprázdněn, porucha atd.), skupina příznakových bitů vytváří stavové slovo (Status Word). Mikropočítač tedy nejprve přečte stavové slovo dodané periferií a na základě analýzy tohoto slova rozhodne o další činnosti. Tomuto řízení budeme říkat ***programové řízení*** nebo někdy také ***metoda periodického testování***.

Zařízení, které chce inicializovat nějakou operaci, pomocí speciálního signálu procesoru přeruší právě probíhající program. Procesor přeruší to, co právě dělal, vykoná požadovanou akci komunikace, potom se vrátí zpět, kde byl přerušen a pokračuje v původní činnosti. K obsluze komunikace více zařízení tímto způsobem musí mít mikroprocesor nebo mikropočítač k dispozici obsáhlejší systém přerušení.

Pokud operace, kterou chce periferní zařízení inicializovat, spočívá v přenosu bloku dat z periferního zařízení do paměti mikropočítače nebo naopak, může se tato operace uskutečnit pomocí tzv. přímého přístupu do paměti (DMA).

V rámci těchto textů se budeme podrobněji zabírat programovým řízením komunikace a řízením pomocí přerušení.

6.2 Programové řízení komunikace

Při použití tohoto principu je komunikace mezi mikroprocesorem nebo mikropočítačem a periferiemi řízena výhradně programovými prostředky, tedy s využitím instrukcí pro vstup a výstup ve spojení s instrukcemi, které umožňují testování logických proměnných a instrukcemi skoků. Použití principu programového řízení je velmi jednoduché u těch mikropočítačů, které mají přímo vnější vstup příznakového (stavového) bitu a instrukce podmíněných skoků, umožňující větvení programu podle hodnoty příznakového bitu. Například jednočipový mikropočítač 8048 má dva vstupní signály T0 a T1 přímo testovatelné instrukcemi podmíněných skoků. Stejně tak signálový procesor TMS 320C50 disponuje vstupem BIO se stejným významem. Takových mikroprocesorů a mikropočítačů však není mnoho a tak nezbývá nic jiného než stavový bit přivést na jeden z pinů vstup-výstupní brány nebo stavovým bitům či stavovému slovu věnujeme jeden z I/O portů. Stavovou informaci pak mikroprocesor čte běžnými instrukcemi a na jejím základě pak zahajuje a řídí komunikaci s danou periferií.

Nyní vzniká otázka, jakým způsobem zorganizovat celkové programové vybavení obsluhy komunikace tak, aby hodnoty stavových bitů byly včas zjištěny.

Typickým případem informace nevyžadující příliš častou pozornost mikroprocesoru jsou data z klávesnice. Platnost znaku při stisknutí klávesy můžeme indikovat nastavením stavového bitu. Tento stavový bit se s ohledem na rychlost činnosti mikroprocesoru (1MIPS) mění velmi pomalu, protože víme, že člověk může zadat z klávesnice jen několik znaků za vteřinu. Bude-li mikroprocesor sledovat tento stavový bit jednou za 0,1 až 0,5 s, neměl by minout žádný znak. Často lze program určitého

zařízení sestavit tak, že opakovaně vykonává určitou posloupnost instrukcí. Tato posloupnost je vzhledem k rychlosti dnešních mikroprocesorů a mikropočítačů natolik krátká, že by nemělo činit problémy snímat stavový bit mnohem častěji např. co 10ms. Pokud však doba trvání je delší, dá se toto testování zařadit vícekrát (např. voláním podprogramu) do posloupností instrukcí algoritmu programu.

Jsou ovšem případy, kdy mikroprocesor spustí určitou fázi komunikace s periferním zařízením a ví, že po určité krátké době toto zařízení nastaví stavový bit a bude vyžadovat rychlou odezvu na tento stav. Tento případ se pak u programového řízení řeší tzv. čekací smyčkou, tzn. že počítač testuje daný stavový bit, dokud nebude nastaven. Toto však může v některých případech způsobit kolizi programu, když z jakéhokoliv důvodu není stavový bit nastaven. Pak program v takovéto čekací smyčce uvázne. Samozřejmě se i toto dá řešit programovým způsobem, např. zařazením počitadla do této čekací smyčky.

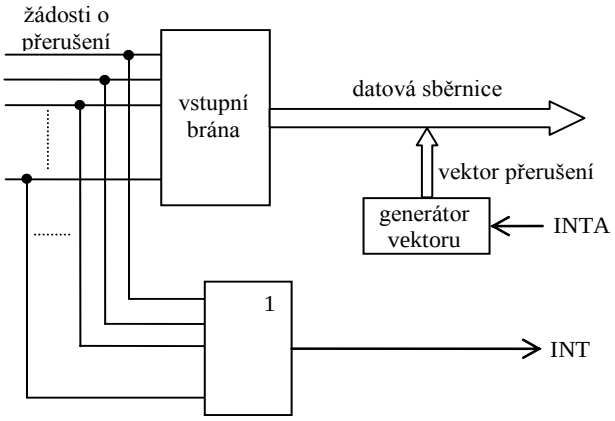
6.3 Řízení komunikace pomocí přerušení

Systém přerušení mikroprocesoru významně usnadňuje programové řízení spolupráce s periferními zařízeními především ve fázi zjišťování žádosti o obsluhu. Jakmile periferní zařízení (pro jednoduchost zatím předpokládáme pouze jedno) požaduje od mikropočítače obsluhu (dílčí přenos dat, různá hlášení atd.), aktivuje přerušovací signál vstupující do mikroprocesoru. Procesor po zjištění žádosti o obsluhu přeruší právě probíhající program a přejde do obslužného programu. Po jeho provedení se procesor vrátí do původního místa v programu, kde byl přerušen a pokračuje v dalším provádění hlavního programu.

U tohoto řízení však nastává poněkud problém při obsluze více zařízení. Jak již bylo výše uvedeno, je bez dalších obvodových úprav potřeba rozsáhlejšího přerušovacího systému. Některé mikroprocesory však mají méně přerušovacích vstupů než kolik bude k němu připojeno periferních zařízení. Žádáme-li tedy obsluhu několika periferních zařízení, můžeme zvětšovat počet přerušovacích vstupů do procesoru, nebo použít jednoho společného přerušovacího vstupu a při vybavení žádosti o přerušení vhodným způsobem zjistit, které zařízení o obsluhu žádalo. Tato informace je nejčastěji mikroprocesoru poskytnuta ve formě tzv. vektoru přerušení, tj. v podstatě adresa příslušného obslužného podprogramu.

❑ Některé možnosti identifikace zařízení:

1. Požadavky na obsluhu od zařízení jsou logicky sečteny a procesor přečte stavové slovo, které bude obsahovat identifikační znak zařízení, které žádá o obsluhu. Jinými slovy, při požadavku na obsluhu si procesor v programu obsluhy přerušení otestuje např. jednotlivé bity brány, ke které jsou připojena žádosti o přerušení všech periferních zařízení a podle programového nastavení určí skok do příslušného podprogramu obsluhy daného zařízení. Přijetí požadavků o obsluhu od dvou či více zařízení najednou, je třeba programově ošetřit. Tento způsob je schematicky naznačen na obr. 6.1.



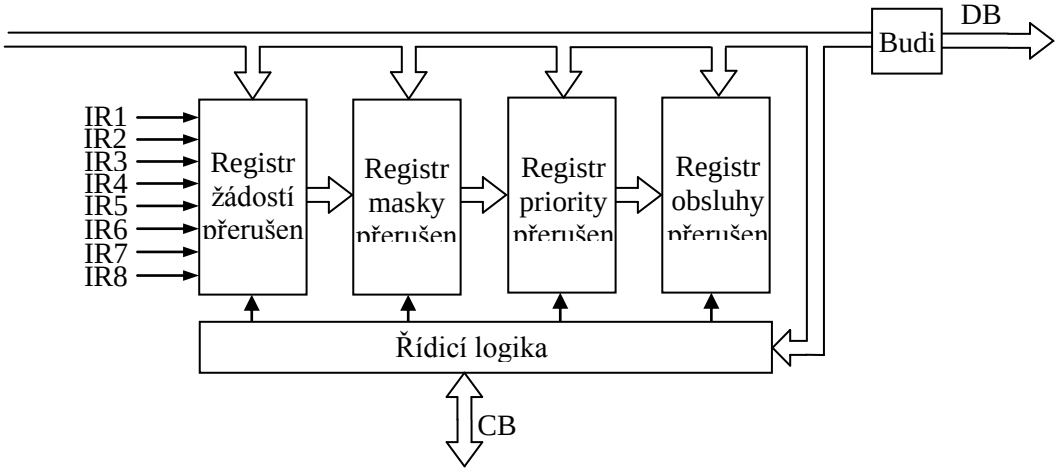
Obr. 6.1 Identifikace zdroje přerušení programovými prostředky

2. Požadavky na obsluhu zařízení jsou opět logicky sečtena a signál potvrzení žádosti od mikroprocesoru je sériově veden přes všechna periferní zařízení. Priorita je tedy dána zařazením jednotlivých zařízení vůči tomuto signálu. Pokud nastane žádost o přerušení, procesor vyšle potvrzující signál nejprve prvnímu zařízení (tedy zařízení s nejvyšší prioritou) a to, pokud nežádalo o přerušení, propustí tento signál dál. Jakmile signál dojde k zařízení jež požadavek o přerušení vydalo, je nutné zabránit dalšímu průchodu tohoto signálu k dalším zařízením a dále je nutno procesoru předat tzv. vektor přerušení, tj. adresu, na které se nachází obslužný podprogram. Tento způsob již vyžaduje přídavné řešení technickými prostředky.

3. Použití tzv. řadiče přerušení, který jednak po přijetí žádosti o přerušení vysílá identifikační znak daného zařízení, jednak přiřazuje priority a jejich dynamickou změnu jednotlivým zařízením při více žádostech najednou a také zajišťuje maskování zdrojů.

   Podrobnější blokové schéma takového řadiče je na obr.6.2. Žádosti o přerušení jsou nejprve konfrontovány s registrem masky, který vlastně rozhoduje o jejich maskování. Dále jsou žádosti zapamatovány v registru žádosti o přerušení. Blok priorit rozhoduje o prioritě jednotlivých žádostí. Základní činností je tedy přijmout požadavky na přerušení, rozřadit je podle jejich priority, vybrat jeden požadavek se současně nejvyšší prioritou a poslat procesoru kompletní informace o adrese, na které je program obsluhy přerušení. Obvod je nutné před činností naprogramovat. Programováním se nastavují priority jednotlivých zdrojů, masky, a v neposlední řadě i adresy obsluhy přerušení příslušným zdrojům. U některých řadičů lze naprogramovat dynamicky proměnnou prioritu žádostí. Sekvence ošetření žádosti přerušení vypadá zjednodušeně následovně:

   - Jedna nebo více žádostí nastaví náběžnou hranou odpovídající bity v registru žádostí přerušení
   - Řadič vyhodnotí žádosti o přerušení a posílá signál žádost o přerušení do procesoru
   - Procesor po akceptaci posílá signál o přijetí žádosti
   - Řadič generuje na sběrnici instrukci CALL. Tato instrukce je akceptována mikroprocesorem
   - Vykonání instrukce způsobí generování dalšího signálu do řadiče
   - Řadič posílá na datovou sběrnici vektor přerušení odpovídající žádosti.
   - Mikroprocesor vykonává instrukci CALL s daným vektorem přerušení



Obr. 6.2 Zjednodušené blokové schéma řadiče přerušení

Shrnutí pojmů 6

Klíčová slova:
Metoda periodického testování

Otázky 6

1. Vysvětlete pojem periodické testování
2. Popište princip komunikace pomocí řadiče přerušení.

Úlohy k řešení 6

60. Řadič přerušení

a. vytváří instrukce skoku
b. slouží pro vytváření signálů CS
c. slouží ke generování vektorů přerušení, jejich priority obsluhy, maskování atd.
d. je součástí každého mikroprocesoru

61. Mapování paměti je

a určení oblasti paměti pro zásobníkovou paměť
b určení velikosti paměti ROM a RAM
c rozdělení paměťového prostoru na paměťovou a nepaměťovou část
d určení počáteční startovací adresy

62. Maskování přerušení:

a je určování priority jednotlivým žádostem o přerušení
b umožňuje ignorovat žádosti o přerušení
c slučuje jednotlivá přerušení do skupin
d zakazuje všechny vnější žádosti o přerušení

7. DRUHY PŘENOSU DAT

Čas ke studiu: 3 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- vysvětlit rozdíl mezi paralelním a sériovým přenosem
- vysvětlit rozdíl mezi asynchronním a synchronním přenosem
- popsat chyby přenosu a jejich eliminaci

Výklad

Mezi charakteristikami mikropočítače a ovládaného okolí je značný nesoulad. Úkolem stykových obvodů je vyloučení tohoto nesouladu, který se projeví odlišným časováním přenosu, rychlostí přenosu, zobrazením a kódováním informace atd.

K mikropočítači budeme připojovat jak moderní tak i starší a méně moderní zařízení. Při posuzování komunikace mikropočítače se zařízením se významně uplatní:

- Způsob časového přidělení sběrnic
  - Trvale připojená jednouúčelová sběrnice
  - Dočasně přidělená víceúčelová sběrnice
- Způsob časování přenosu
  - Asynchronní
  - Synchronní
- Šířka datové (přenosové sběrnice)
  - Paralelní přenos
  - Sériový přenos

Přenos každé informace musí být definován v čase synchronizačními signály s různým stupněm volnosti vzájemné funkční vazby.

Synchronní přenos je charakterizován vazbou na pevný časový úsek (takt) během celého přenosu, s výhodou se využívá hodinový signál mikroprocesoru. Komunikující části systému a okolí jsou stejně rychlé. Pokud není rychlost komunikace stejná, je možné využít principů z kapitoly 6.

Asynchronní přenos je z časového hlediska opakem synchronního, rozborem časovacích signálů obecně nezjistíme žádný pravidelný časový interval. Časování je plně podřízeno funkčním charakteristikám pomalejšího zařízení.

Šířka sběrnice bezprostředně souvisí s mnoha požadovanými charakteristikami přenosu, zejména se spolehlivostí, přípustnými náklady a výkonem. Sběrnici vytváří soubor linek, který by měl být optimální. Realizace každé linky vyžaduje příkon, prostor a náklady přímo úměrné délce linky. Při přenosu na větší vzdálenosti proto dáváme přednost sériovému přenosu s malým počtem linek. Základní způsoby zmenšení počtu linek vždy souvisí se způsobem kódování a zobrazení přenášené informace.

Paralelní přenos má největší přenosový výkon, jeho realizace je nejdražší a existuje nebezpečí vzájemného rušení jednotlivých linek. Považuje se proto za vhodný pro přenos na krátké vzdálenosti

při vysokém přenosovém výkonu. Sběrnice uvnitř mikropočítače jsou převážně řešeny pro paralelní přenos.

- *Jednoslovní synchronní paralelní přenos*
  Šířka datové sběrnice stykových obvodů je zpravidla totožná s šířkou vnitřní datové sběrnice mikropočítače. Přenos je časován signály, které jsou totožné s vnitřními synchronizačními hodinovými signály mikropočítače nebo jsou z nich časově odvozeny. Přenos je úplně řízen programem a je tedy použitelný pro pomalá zařízení, která jsou v libovolném okamžiku schopna komunikovat v synchronismu s mikropočítačem. Taková situace je možná v případě ovládaného okolí, jež má dynamický charakter a pracuje synchronně s mikropočítačem anebo má okolí statický charakter a je téměř lhostejné, ve kterém okamžiku komunikace proběhne. Synchronizace může být také provedena obvodovým řešením zastavení mikroprocesoru signálem READY nebo programovými čekacími stavy (WAIT STATE).

- *Jednoslovní asynchronní paralelní přenos*
  Jedná se o typ přenosu, který je vhodný pro malé vzdálenosti a pomalá zařízení, jež nejsou schopna komunikace v libovolném okamžiku. Přenos je tedy zahájen až po splnění základní podmínky přenosu: Zařízení je schopno komunikace. Zjišťováním této podmínky se tento přenos liší od synchronního, proto bývá někdy nazýván podmíněným jednoslovním paralelním přenosem.
  Přenos je řízen programem a jeho algoritmus provádí nejprve zjišťování splnění podmínek pro zahájení přenosu a provedení vlastního přenosu. V některých případech se tento typ komunikace nazývá „Handshaking“.

Sériový přenos je v podstatě zvláštní typ paralelního přenosu se šířkou datové sběrnice 1 bit. Vyznačuje se levnější realizací a menším nebezpečím vzájemného rušení. Považuje se proto za vhodný pro přenos na větší vzdálenosti, případně v prostředích se silnějším rušením. U malého počtu linek je únosné dobré obvodové řešení linky s účinným potlačením rušení.

Přenášenou informaci je nutno na straně vysílače zakódovat do časové posloupnosti bitů – nulových a jedničkových logických impulsů, a na straně přijímače opět převést do paralelní formy. Základními částmi sériového vstupně výstupního kanálu bude tedy obvod, jenž převádí paralelní informaci na sériovou a opačně.

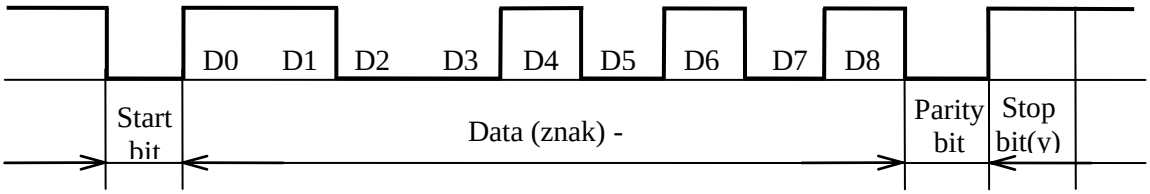
Nositelem informace bývá stejně jako u paralelního přenosu amplitudově (dvouhodnotově) modulované elektrické napětí, pro sériový přenos poněkud vyšší napěťové úrovně viz násl. kapitola.

Rychlost sériového přenosu se udává v počtech bitů za sekundu tzv**. Baudech (Bd)**. Rychlost dále také závisí na tom, kolika bity je daná informace reprezentována.

Z časového hlediska může být sériový přenos buď synchronní nebo asynchronní. V obou případech musíme do vysílané informace vkládat dodatečnou synchronizační informaci. Přijímač musí nejen číst bity se stejnou frekvencí jakou jsou vysílány, ale musí rovněž každý bit číst přibližně ve středu intervalu pro něj vymezeného – hrany signálu mohou být totiž deformovány.

- *Synchronní přenos*
  Používá se pro rychlejší přenos a umožňuje vyšší vytížení přenosové linky. Jednotlivé znaky zprávy nejsou odděleny synchronizační obálkou a následují bezprostředně za sebou. Synchronizace je zajištěna pomocí synchronizačních znaků, které jsou vysílány v dohodnutém tvaru spolu s informacemi. Synchronizace je společná pro vysílač i přijímač. Přenášený znak může být doplněn paritním bitem.

- *Asynchronní přenos*
  Uskutečňuje synchronizaci přijímače a vysílače na začátku každého přenášeného znaku, reprezentovaného podle volby pěti až osmi bity. Jednotlivé znaky nemusí tedy na sebe bezprostředně navazovat, mohou být mezi nimi libovolné mezery. Příklad přenosu je na obr. 7.1.



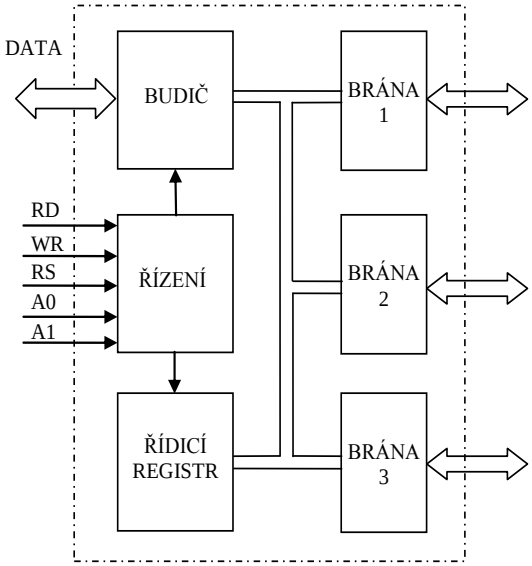
Obr. 7.1 Asynchronní přenos

V mezerách mezi znaky je na signálovém vodiči úroveň logická „1“. Přenos znaku je započat start-bitem úrovně logické „0“. Sestupná hrana tohoto bitu slouží pro synchronizaci vysílače s přijímačem. Pak následuje přenos informace v dohodnutém formátu – tzv. protokol přenosu. Ve většině případech se přenáší jako první nejnižší bit. Za posledním bitem znaku může následovat paritní bit a posléze dohodnutý počet stop-bitů úrovně logická „1“. Počet stop-bitů tak vymezuje minimální mezeru mezi vysílanými znaky informace. Volitelné parametry přenosu jako jsou počet přenášených bitů znaku, parita lichá, sudá či žádná, rychlost přenosu atd. se nastavuje před zahájením přenosu shodně u vysílače i přijímače.

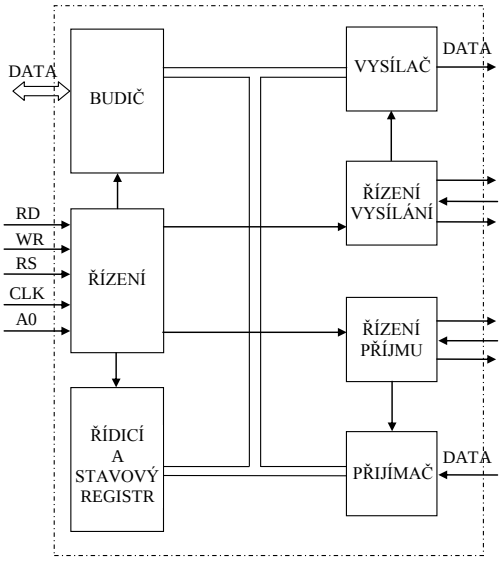
7.1 Obvody vstupu a výstupu

Mnoho výrobců polovodičových součástek dodává mimo vlastních mikroprocesorů a pamětí taktéž ještě celou řadu specializovaných obvodů rovněž vysokého stupně integrace, sloužící k snadné realizaci vstupních a výstupních portů. Tyto obvody jsou velmi často programovatelné, tzn. že procesor před vlastním použitím těchto obvodů jako V/V kanálů vysílá do řídicích registrů těchto obvodů tzv. řídicí a povelová slova, která nastaví tyto obvody do určitého režimu. Mezi nejčastěji používané obvody V/V patří programovatelný paralelní V/V obvod, programovatelný obvod pro asynchronní nebo synchronní sériový přenos USART a programovatelný časovač PIT.

Mezi specializované pak patří řadič DMA, řídicí obvod klávesnice, řadiče disků řídicí obvody sběrnice a další.



Obr. 7.2 Paralelní V/V obvod



Obr. 7.3 Sériový V/V obvod

V dnešní době jsou však tyto obvody ve většině případech implementovány přímo na čip samotného mikroprocesoru.

7.2 Chyby přenosu a jejich redukce

Na každý přenosový kanál působí poruchy, které mohou zkreslit přenášenou informaci takovou měrou, že se přijatá informace liší od vyslané. Při přenosu může tedy docházet k chybám. Poruchy můžeme kategorizovat:

- podle jejich charakteru na
  - fluktuační
  - harmonické
  - impulsní
- podle jejich zdroje na
  - přírodního původu
  - průmyslového původu
  - vznikající v zařízení
- podle časového výskytu
  - dočasně působící
  - trvale působící

Podrobnějším rozborem bychom však došli k zjištění, že vznik poruchy je náhodný proces, který nelze předvídat a kterému nelze zabránit. Řadou opatření však můžeme účinně omezit vliv poruch na přenos informace.

Možná a prakticky realizovatelná opatření budou patřit do dvou oblastí:

- správné řešení stykových obvodů a přenosového vedení
- volba vhodného kódového zabezpečení

Přenosové vedení je při vyšších kmitočtech a krátkých hranách impulsního signálu nutno chápat jako elektrické vedení, u kterého může docházet k odrazům, přeslechům a k rušení cizími elektromagnetickými poli. Proto je nutno věnovat pozornost tomu, aby

- vedení byla pokud možno fyzicky krátká, homogenní a na obou koncích impedančně správně zakončena
- souběžné linky měly pokud možno malou kapacitní vazbu
- funkční bloky mikropočítače a stykových obvodů byly správně zemněny s důsledným rozlišením vysokofrekvenční (signálové) země a přístrojové země (kostry).

Informace se přenáší kódovaně. Kódem rozumíme přiřazení dvou množin. Vhodně zvolené kódování může zabezpečit přenášenou informaci proti chybám.

Zabezpečené kódy dělíme na kódy:
- zjišťující chybu (detekční)
- umožňující chybu opravit (samoopravné)

Kód se schopností zjistit k-chyb musí mít $m \geq k+1$.
Kód umožňující opravit k-chyb musí mít $m \geq 2k+1$.

Nejznámějším případem detekčního kódu je kód zabezpečený paritou. Vznikne doplněním jistého kódu o jeden paritní bit, do kterého umístíme kontrolní číslici. Hodnota paritního bitu se vytváří v závislosti na počtu jedniček v doplňovaném kódovém slově. U liché parity musí být počet všech jedniček (včetně paritního bitu) lichý, u sudé parity sudý.

Paritní kódy mají $m = 2$ a jsou tedy schopny detekovat jednu chybu eventuelně lichý počet chyb. Nesouhlas parity upozorní na chybu, ale nelze již z něj nijak vyvodit umístění chyby ve slově. Zabezpečení paritou je velmi rozšířené u mikroprocesorů a ve většině případech je u nich přizpůsoben jejich instrukční soubor.

Velmi zajímavé jsou cyklické kódy, které se uplatní při přenosu po dávkách. Užívají se u magnetických diskových pamětí a v některých komunikačních protokolech. Zabezpečení dávky cyklickým kódem vychází z myšlenky nezabezpečovat jednotlivá slova dávky, ale vypočítávat na základě jejich hodnoty doplňkové kontrolní slovo pro celou dávku. K výpočtu kontrolního slova (CRC) dochází v průběhu vysílání slov dávky, závěrem je vysíláno CRC. Na přijímací straně se vypočte nové CRC ze zabezpečené dávky a z výsledku se usuzuje na bezchybnost přenosu.

7.3 Sériová rozhraní

Sériová komunikační rozhraní se v mikropočítačové technice používají ke dvěma základním účelům:

1. Ke komunikaci mezi jednotlivými mikropočítačovými moduly. Typická délka vedení je zde v řádu jednotek až stovek metrů, takže fyzická vrstva v různé míře řeší i problémy odolnosti proti rušení atd. Kromě známých a rozšířených rozhraní (RS232, RS485) se používají i specializovaná rozhraní resp. sběrnice (CAN, TTP). Při rostoucích nárocích na přenosovou kapacitu se pro toto propojení často používá i Ethernet.
2. Ke komunikaci mezi integrovanými obvody, případně ke komunikaci mezi mikropočítačovými moduly na krátkou vzdálenost. Typická délka vedení zde nepřesahuje jednotky metrů. Často používaná rozhraní pro komunikaci mezi jednotlivými IO jsou rozhraní Microwire, SPI a I2C.

Důvodem používání sériové komunikace mezi jednotlivými obvody je především zmenšení počtu vývodů jejich pouzder. Při použití sériových pamětí se zredukuje množství adresních, datových a řídicích vývodů obvyklé (nikoliv sériové) paměti na tři až čtyři vývody. To umožňuje zmenšit rozměry pouzdra i spojové desky, protože odpadá prostorově náročné propojování velkým počtem vodičů.

Další výhodou může v některých případech být možnost připojení obvodů se sériovým rozhraním i k mikrokontrolérům bez vyvedené vnitřní sběrnice, v krajním případě i bez příslušného řadiče sériového rozhraní. Funkce řadiče je potom realizována programově s využitím několika vývodů vhodného portu mikrokontroléru. Nelze tak většinou dosáhnout plné rychlosti daného rozhraní, ale tato skutečnost nemusí být v některých případech na závadu.

Jak již bylo výše uvedeno, využívá se z důvodu větší odolnosti proti rušení přenosu informace fyzikální úprava nositele informace a určitého způsobu realizace přenosových (stykových) obvodů. Pro sériové přenosy pak vznikly různé standardy, které odpovídají určitému normalizovanému předpisu.

7.3.1 Rozhraní RS232

RS 232 používá dvě napěťové úrovně. Logickou 1 a 0. Log. 1 je někdy označována jako **marking state** nebo také klidový stav, Log. 0 se přezdívá **space state.**

Log. 1 je indikována zápornou úrovní, zatímco logická 0 je přenášena kladnou úrovní výstupních vodičů. Povolené napěťové úrovně jsou uvedeny v tabulce.

Úroveň	Vysílač	Přijímač
Log. L	+5 V to +15 V	+3 V to +25 V
Log. H	-5 V to -15 V	-3 V to -25 V
Nedefinovaný	-3 V to +3 V	

Nejběžněji se pro generování napětí používá napěťový zdvojovač z 5 V a invertor. Logické úrovně jsou potom přenášeny napětím +10 V pro log. 0 a –10 V pro log. 1.

Standard RS 232 uvádí jako maximální možnou délku vodičů 15 metrů, nebo délku vodiče o kapacitě 2500 pF. To znamená, že při použití kvalitních vodičů lze dodržet standard a při zachování jmenovité kapacity prodloužit vzdálenost až na cca 50 metrů.

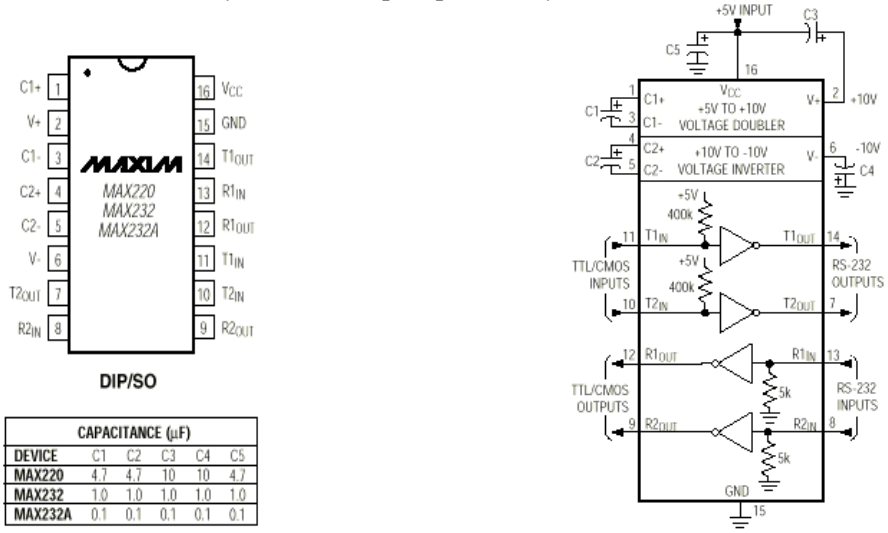
Kabel lze také prodlužovat při snížení přenosové rychlosti, protože potom bude přenos odolnější vůči velké kapacitě vedení. Uvedené parametry počítají s přenosovou rychlostí 19200 Bd.

Texas Instruments uvádí jako výsledek pokusných měření následující délky vodičů v závislosti na přenosové rychlosti. Vzhledem k „laboratorním“ podmínkách tohoto měření je třeba brát tyto údaje pouze jako orientační. V praxi je třeba počítat s rušením atd.

Používají-li se v zařízení TTL nebo CMOS obvody, bude nutno jejich výstupní úrovně upravit na požadované napěťové úrovně RS232. K tomuto účelu složí převodníky RS232 např. od firmy MAXIM MAX232.

Jedná se o převodník TTL na RS232. Obsahuje dvě dvojice oddělovačů konvertujících napěťové úrovně. Napětí pro RS 232 se získává pomocí nábojové pumpy, a výstupní napětí proto značně závisí na kvalitě použitých kondenzátorů, která u elektrolytických kondenzátorů časem značně klesá. Napětí je možno získat na pinech 2 a 6 a použít pro další obvody.

Vzhledem k úspěšnosti MAX232 začalo mnoho firem vyrábět obvody pinově kompatibilní v nižší cenové hladině. U jednoho z těchto výrobců (tuším AD232), které u nás svého času prodávalo GM, je potřeba opačně polarizovat jeden z elektrolytů, tak uvádí firemní katalogový list. Vzhledem k předpokládané kompatibilitě to však mnoho vývojářů neověřuje, a potom vznikají časem velmi komplikované závady. Doporučujeme proto používat buď originální obvody MAXIM, nebo dobře prostudovat “substituční“ obvody vzhledem k předpokládaným odlišnostem.



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

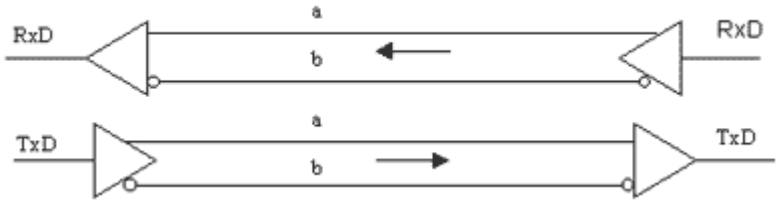
Obr. 7.4 Vnitřní zapojení a pouzdro převodníku TTL/RS232

7.3.2 Rozhraní RS422

Linka RS422 používá jeden pár vodičů pro signál RxD a druhý pro signál TxD. Z toho vyplývá, že použijeme-li linku RS422 k prodloužení přenosové vzdálenosti místo “třídrátové” RS232 (RxD, TxD, GND), nic se nemusí na způsobu komunikace měnit a není tedy třeba ani zásah do software.

Každý ze signálů linky je přenášen po dvojici vodičů, nejlépe v provedení twistový pár. Vodiče označované **a** a **b** jsou vysílačem buzeny v protifázi a přijímač vyhodnocuje jejich napěťový rozdíl. Tímto principem se odstraní součtové (aditivní) rušení.

Linky mohou být vedeny až na vzdálenost 1600m (vodiče s kapacitou do 65pF/m) a lze je větvit.



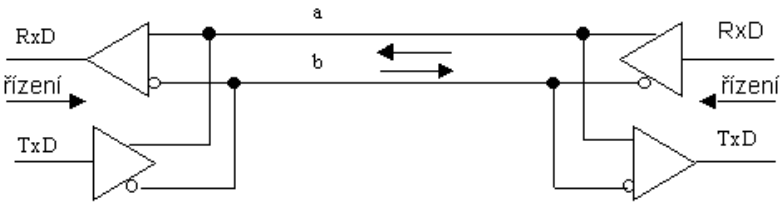
Obr. 7.5 Schéma zapojení linky RS422

7.3.3 Rozhraní RS485

Narozdíl od linky RS-422 používá rozhraní RS-485 jen jeden pár vodičů pro oba směry toku dat. Je tedy třeba směr komunikace přepínat a to může být problém zvláště v případech, kdy s touto možností software nepočítá.

Přepínání směru komunikace jistě bude vyřešeno u zařízení, které obsahuje už standardně linku RS-485. Pokud však používáme zařízení s vyvedenou linkou RS-232 (například počítač PC) a následným převodníkem RS-232/RS-485, je třeba přepínání směru zajistit. Nejvhodnější způsob je použít pro přepnutí některý volný řídící signál linky RS-232 (například DTR nebo RTS), jeho ovládání však musí umožnit použitý program (pozor při psaní takového programu pro PC – díky různým bufferům na nových motherboardech to není úplně jednoduché).

Jestliže není signál pro přepnutí k dispozici, je jedinou možností použít převodník linky RS232 na RS485 s automatickým přepínáním. I to má však úskalí. Takový převodník je stále přepnut na příjem z linky RS485 a při zjištění dat vysílaných ze strany linky RS232 se přepne na vysílání. V režimu vysílání však převodník zůstane ještě po nějakou dobu (protože nemůže přesně identifikovat konec dat). Jestliže během této doby začne na linku vysílat někdo jiný, dojde ke kolizi a data nejsou přijata. Problémy s přepínáním lze však po prozkoumání konkrétní situace téměř vždy vyřešit. Poslední možností je použít linku RS422, která přepínání nepotřebuje.
U rozvětvených linek může být počet zařízení na lince maximálně 16, avšak existují přijímače s menší zátěží, takže jich může být až 128. Linka by měla být provedena jako linie s krátkými odbočkami, ne jako strom nebo hvězda.



Obr. 7.6 Schéma zapojení linky RS 485

7.3.4 Proudová smyčka

Pro komunikaci na větší vzdálenosti se občas používá také proudová smyčka 0/20 mA (pozor, nezaměňovat se smyčkou 0(4) až 20 mA pro přenos analogových veličin).

Proudová smyčka je vysoce odolná proti rušení, neboť smyčkou buď teče, nebo neteče proud 20 mA. Převod je relativně jednoduchý a dosah se řádově zvýší na stovky metrů, zatímco komunikační rychlost zůstane zachována. Z ekonomických důvodů se na proudovou smyčku převádějí pouze signály RxD a TxD z plného rozhraní RS232. Pokud potřebujete přenášet ještě nějaké signály (RTS, CTS / DTR, DSR), použijte 2kanálovou verzi převodníku.

Délka vedení je omezena kapacitou a ohmickým odporem vedení. V praxi je rozhodujícím faktorem jen odpor. U běžně užívaných proudových smyček bývá povolený odpor vedení do 200Ω. Proudové smyčky se od sebe liší provedením vysílačů a přijímačů. Mohou být aktivní nebo pasivní, přímé nebo galvanicky oddělené.

Galvanické oddělení může být i tím hlavním důvodem, který vede k použití proudové smyčky. Vedení proudové smyčky se realizuje velmi jednoduše, v minimálním případě stačí čtyři vodiče, které ani nemusí být zkrouceny (ve většině případů stejně budou, použijeme-li sériově vyráběný kabel).

Propojuje se vzájemně vždy jeden vysílač s jedním přijímačem, a to v kombinaci aktivní vysílač – pasivní přijímač nebo pasivní vysílač – aktivní přijímač.

7.3.5 *Rozhraní USB*

Rozhraní USB se během posledních let stalo zcela běžnou součástí spotřební elektroniky připojitelnou k počítači a svými vlastnostmi zcela zastínilo klasický sériový port RS-232. Tento standard vznikl v roce 1995 a byl vyvinutý ve spolupráci společností Compaq, Intel, IBM, Microsoft, NEC a dalších. Podpora USB se poprvé objevila u operačního systému Windows 95 OEM Service Release 2.1 (označovaná jako 4.000.111) a stala se běžnou součástí všech dalších operačních systémů. Podpora formátu USB 2.0 je dostupná v aktualizujících servisních balíčcích pro systémy Windows 2000 a XP.

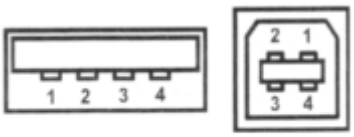
Základní parametry rozhraní USB:

- Komunikační rychlost od 1,5 Mbit/s do 480Mbit/s
- Komunikační vzdálenost do 5m
- Možnost připojení více zařízení
- Rozhraní obsahuje 5V napájení
- Lze připojit až 127 zařízení pomocí jednoho typu konektoru.
- USB zajišťuje správné přidělení prostředků (IRQ, DMA, ...).

❑ Elektrické parametry USB

USB rozhraní používá dva typy konektorů. Plochý konektor "typ A" je dnes obsažen na základní desce každého PC, vyrobeného v posledních několika letech a to v počtu minimálně dvou. Nejnovější základní desky mají kořenový HUB rozšířen o další integrovaný HUB a celkem tak poskytují až osm portů USB.

Druhý konektor "typ B" je určen pro periferní zařízení, čímž je zároveň definován standard propojovacího kabelu. Oby typy konektorů jsou znázorněny na obr. 7.7 a popis vodičů je uveden v tab. 7.1



Obr. 7.7 Konektory USB

Číslo	Jméno	Barva	Popis
1	V bus	Červená	Napájecí napětí
2	D-	Bílá	Data (-)
3	D+	Zelená	Data (+)
4	GND	černá	Zem

Tab. 7.1 Popis pinů USB konektoru

sériový port	115 kb/s
USB 1.1 Low Speed	1,5 Mb/s
USB 1.1 Full Speed	12 Mb/s
paralelní port ECP/EPP	24 Mb/s
FireWire IEEE 1394	400 Mb/s
USB 2.0 High Speed	480 Mb/s

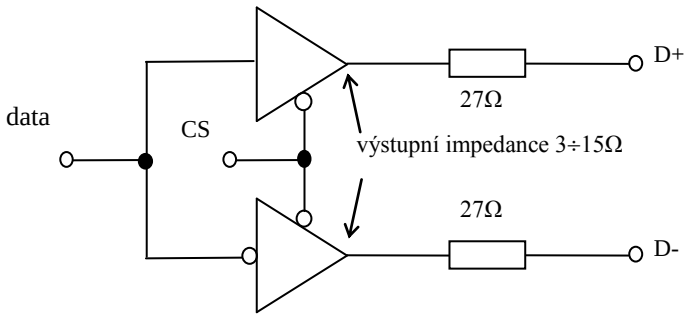
Tab. 7.2 Porovnání rozhraní

❑ Komunikace

Všechny tři verze USB, dané přenosovými rychlostmi (Low Speed, Full Speed, High Speed), mohou být použity a provozovány současně pro připojení různých typů periferií k jednomu počítači. Uvedené verze se od sebe liší jak provedením kabelu, tak elektrickými parametry rozhraní připojeného zařízení. Přenos je diferenční s napěťovými úrovněmi kompatibilními s TTL:

- log. 1 se přenáší tak, že vodič D+ je na napěťové úrovni 2,8V a současně vodič D- je na úrovni 0,3V,
- log. 0 je řešena opačně, čili D- je na napěťové úrovni 2,8V a D+ je na úrovni 0,3V.

Pro využití maximální přenosové rychlosti v režimu High Speed (až 480Mb/s) může být kabel dlouhý nejvíce 5 metrů, přičemž musí být stíněný a kroucený (twisted). V nejpomalejším režimu Low Speed (do 1,5Mb/s) je možné použít i nestíněný a nekroucený kabel, ale u tohoto kabelu je maximální délka pouze 3 metry. Nejlepších výsledků je vždy dosaženo při použití stíněného krouceného kabelu. Přenosová impedance kabelu je v obou případech 60Ω. Pro obě zmíněné varianty je použito shodné zapojení vysílačů, uvedené na obr.7.8. Minimální diferenční vstupní napětí přijímače je 200mV a to při souhlasném rušivém napětí až 2,6V aniž by se omezila funkčnost systému.



Obr. 7.8 Zapojení vysílačů

Rozhraní USB obsahuje vlastní rozvod napájecího napětí o hodnotě 5V a 0,25V. Zařízení mohou být napájena přímo z USB sběrnice, pokud jejich proudový odběr nepřekročí 100mA. Požadavek na větší odběr (až 500mA) musí být ohlášen při enumeraci (rozpoznání) zařízení. Vyšší odběr než je 100mA, může mít nejvýše jedno zařízení na celé USB sběrnici a při nedostatku rezerv může systém toto zařízení odmítnout. Pokud mají USB zařízení vlastní zdroj (např. počítač má vlastní systém pro distribuci napájení nezávislý na USB), je řízen USB sběrnicí (zapínání, vypínání, "stand-by" mód atd.). Každý segment USB umožňuje omezený přenos výkonu pro napájení USB zařízení, přičemž zařízení může být současně napájeno z vlastního zdroje.

7.3.6 Rozhraní SPI

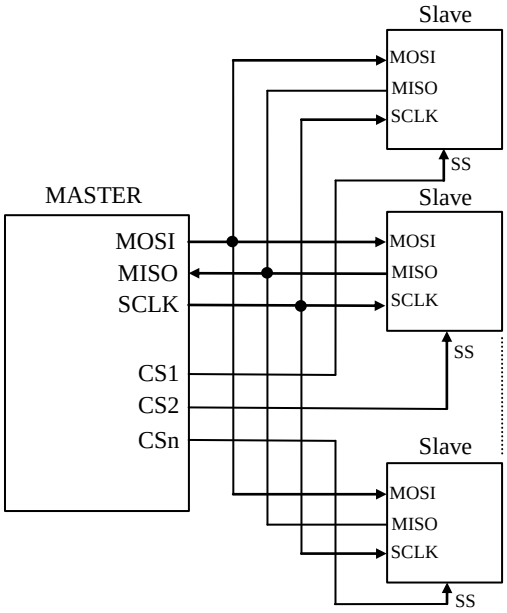
Rozhraní SPI je určeno především pro připojení vnějších pamětí, A/D převodníků a dalších obvodů k mikrokontroléru, případně pro vzájemnou komunikaci mezi mikrokontroléry. U některých mikrokontrolérů je SPI využíváno i pro programování jejich vnitřní paměti Flash.

Základní koncepce systému využívajícího sběrnici SPI je následující (viz Obr. 7.9):

V systému mohou být zapojeny dva nebo více obvodů. Jeden z obvodů, obvykle procesor, je typu Master, ostatní jsou typu Slave. Jednotlivé obvody jsou propojeny čtyřmi vodiči:

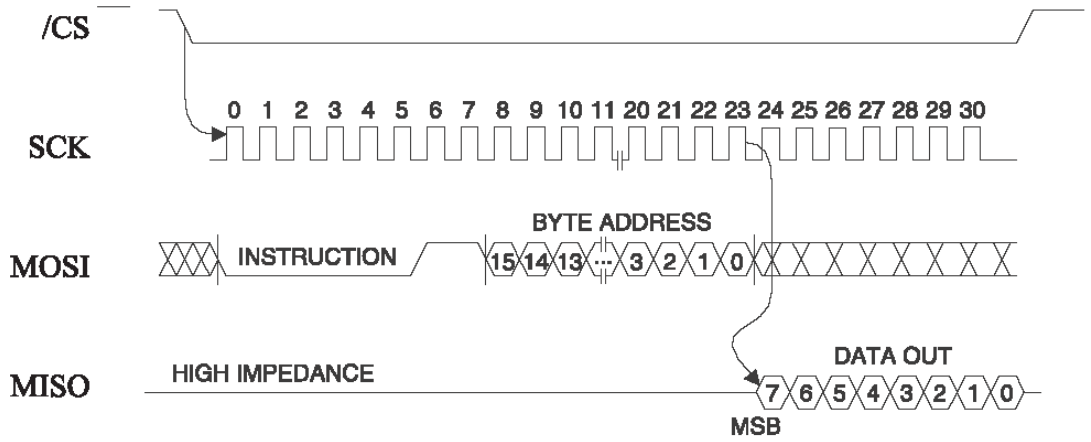
- Datový výstup MOSI (Master Out, Slave In) obvodu Master je připojen na vstupy MOSI všech obvodů Slave
- Datový vstup MISO (Master In, Slave Out) obvodu Master je propojen s výstupy MISO všech obvodů Slave.
- Výstup hodinového signálu SCK je připojen na vstupy SCK všech obvodů Slave.
- Každý obvod Slave má vstup SS (Slave Select) pro výběr obvodu. Je-li SS v neaktivní úrovni, je rozhraní SPI daného obvodu neaktivní a jeho výstup MISO je ve vysokoimpedančním stavu. Vstupy SS jednotlivých obvodů jsou samostatnými vodiči propojeny s obvodem Master. Je-li obvodem Master mikrokontrolér, bývají tyto vodiče připojeny k některému z jeho portů. Tak lze snadno vybírat obvod, se kterým má být v daném okamžiku vedena komunikace.

Přenosy na sběrnici SPI probíhají vždy mezi obvodem Master a některým z obvodů Slave. Oba obvody obsahují posuvné registry, které jsou v okamžiku komunikace propojeny tak, jak je schematicky naznačeno na obr. 7.9.

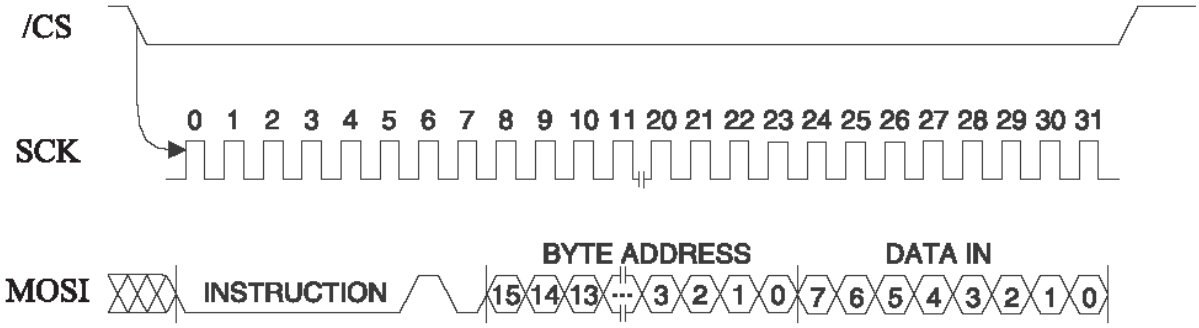


Obr. 7.9 koncepce systému se sběrnicí SPI

Obvod Master generuje hodinový signál, který řídí posouvání obou posuvných registrů. Klidová úroveň signálu SCK a vztah mezi datovým a hodinovým signálem je dán parametry CPOL a CPHA. Pokud je rozhraní SPI realizováno specializovaným řadičem, je obvykle možné tyto parametry v řadiči nastavit. Je-li rozhraní SPI realizováno programově, musí být okamžiky změny úrovně datových a hodinových signálů zvoleny tak, aby přijímající obvod vzorkoval ustálená data.



Obr. 7.10 Čtení z paměti (SPI přenos)



Obr. 7.11 Zápis do paměti (SPI přenos)

7.3.7 Rozhraní I²C

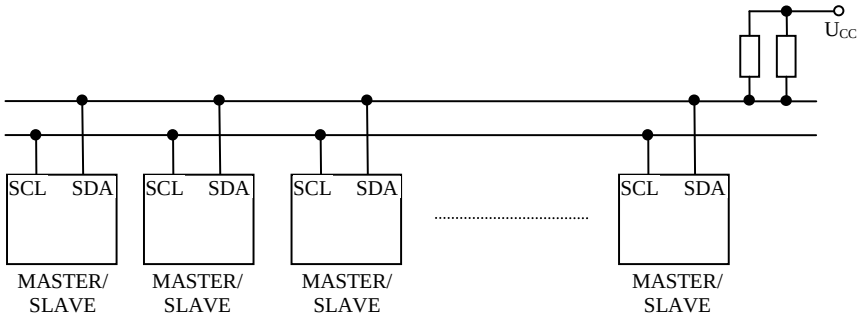
Zatím co RS232 je stále jeden z nejpoužívanějších prostředků komunikace mezi PC a jednočipem, pro komunikaci mezi čipy samotnými se spíše používá I²C. Jedná se obousměrnou komunikaci pomocí dvou drátů, kterou pro svá zařízení navrhl Philips. Nechal si ji patentovat (patent číslo 9398 393 40011), a tak z důvodu patentových poplatků někteří výrobci nehovoří o I²C (Inter IC), nýbrž např. o TWI (Two Wire Serial Interface - dvoudrátové sériové rozhraní).

Asi základní rozdíl od RS232 je zavedení adresace už do samotného protokolu a proměnná rychlost komunikace. U RS-232 obě zařízení mohla mluvit současně — to není u I²C možné, a tak vždy pouze jedno zařízení *„mluví“* na datové lince (SDA) a *„rychlost mluvy“* je určena pulzy na druhé, časové lince (SCL). I²C pak řeší problémy pokud některý z účastníků nestíhá, případně pokud by chtělo *„mluvit“* příliš mnoho zařízení současně.

Stejně jako RS-232 má i I²C několik možných variant, z nichž se detailněji budeme věnovat pouze jediné. Jednotlivé varianty se liší způsobem adresace zařízení (7 vs. 11 bitová adresa), rychlostí komunikace (maximální frekvence 100kHz vs. 400kHz; existují i 1 MHz) a zda se jedná o komunikaci s jedním či více zařízeními typu *master*.

Koncepce rozhraní

Jednotlivé stanice rozhraní I²C jsou propojeny jedním datovým vodičem (SDA) a jedním hodinovým vodičem (SCL).

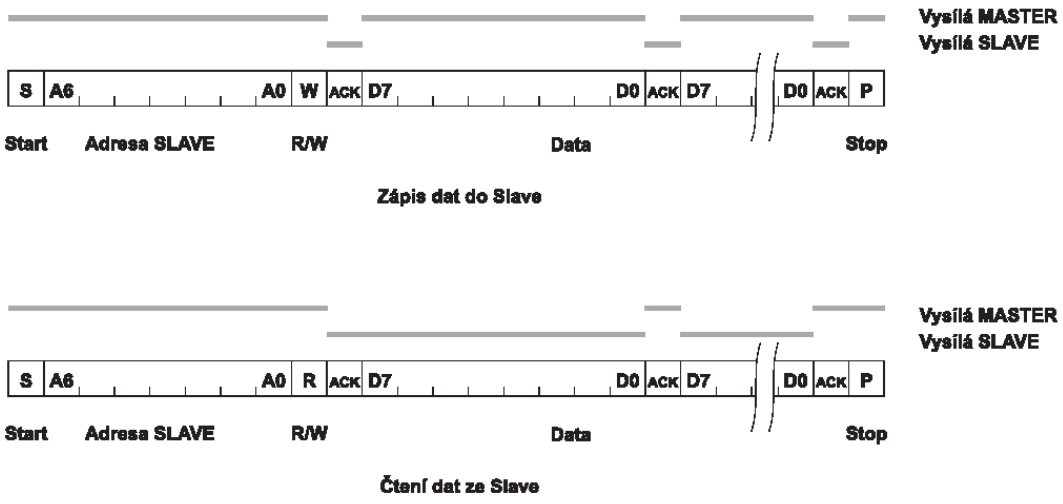


Obr. 7.12 Celková koncepce sběrnice I²C

Z elektrického hlediska jsou oba vodiče typu otevřený kolektor. Jejich maximální délka je dána jejich nejvyšší přípustnou kapacitou 400 pF. Oba vodiče jsou připojeny přes pull-up odpory k napájecímu napětí 5V.

Komunikace

Formát rámce se sedmibitovou adresou při přenosu na I²C je na Obr. 7.13. Každému přenosu předchází vyslání podmínky START. Potom je vysílána 7bitová adresa příjemce a jeden bit R/W, který indikuje požadovanou operaci (čtení/zápis). Další bit ACK je vysílán s úrovní L a je určen k potvrzení přijímací stanicí. Dále jsou přenášena data ve směru určeném předchozím bitem R/W. Každý byte je následován jedním bitem ACK. Po ukončení přenosu je vyslána podmínka STOP.



Obr. 7.13 Protokol komunikace I²C

Podmínka START je vygenerována změnou úrovně z H na L na datové lince SDA, STOP podmínka pak při změně z L na H při úrovni H na lince SCL. Potvrzení ACK je generováno úrovní L na lince SDA při L na SCL.

V klidovém stavu jsou na obou vodičích SDA i SCL úrovně H. Platí pravidlo, že data na lince SDA se mohou měnit jen když je na lince SCL úroveň L. Výjimkou jsou výše uvedené případy.

Poté co byla započata komunikace podmínkou start, příslušný vysílač vyšle na datovou linku SDA postupně osm datových bitů, které se posouvají hodinovými impulsy na lince SCL, vysílanými

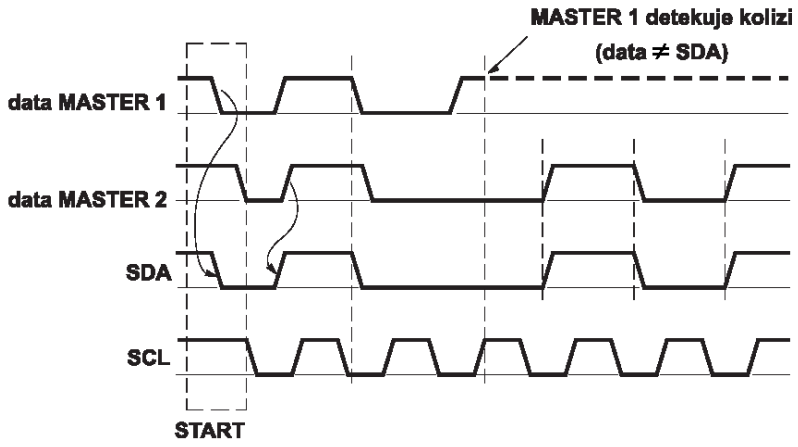
prvkem master. **Přenos začíná bitem s nejvyšší váhou**. Poslední (devátý) bit slouží pro potvrzení(acknowledge) příjmu od příslušného přijímače pomocí nízké úrovně. Toto potvrzení zároveň znamená, že obvod je schopen přijímat další byte. Pokud je zde logická jednička, obvod nebyl správně adresován, došlo k chybě, nebo prostě sděluje, že již nebude přijímat.

Pokud *master* čte data z obvodu *slave*, přenos je stejný až na to, že data generuje SLAVE, podle toho, jak se mění hodiny generované *masterem*. Pokud již MASTER nepotřebuje číst data (chce ukončit přenos), nemůže toho dosáhnout jednoduchou STOP podmínkou, ale při čtení posledního bytu ponechá potvrzovací bit ACK v logické jedničce (čímž nedojde k potvrzení) a obvod SLAVE uvolní datovou linku. Vlastní ukončení přenosu se poté dosáhne podmínkou stop.

Arbitrace

U rozhraní I^{2}C existuje i režim MULTI-MASTER. Tento režim je zvláštní v tom, že na jedné sběrnici může být více obvodů master, kteří si toto postavení vzájemně předávají. To znamená, že v jednom okamžiku může být na sběrnici jen jeden master a ostatní jsou slave. Poté se třeba tento master stane slavem a předá se řízení jinému obvodu, který bude nyní master.

Pro arbitraci se na I^{2}C používá metoda s detekcí kolize. Každá ze stanic může zahájit vysílání, je-li předtím sběrnice v klidovém stavu. Během vysílání musí neustále porovnávat vysílané bity se skutečným stavem SDA. Je-li zjištěn rozdíl mezi očekávaným a skutečným stavem linky SDA, je to indikace kolize mezi několika stanicemi. Vzhledem k charakteru sběrnice (otevřené kolektory) může k této situaci dojít, pokud určitá stanice vysílá úroveň H, zatímco jiná stanice vysílá úroveň L. Stanice, která na lince zjistí úroveň L zatímco sama vysílá H musí vysílání okamžitě ukončit (viz Obr. 7.14).



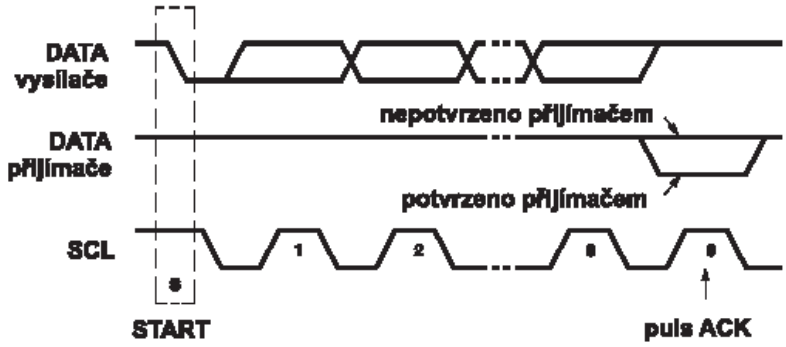
Obr. 7.14 Arbitrace na I^{2}C

K arbitraci většinou dochází během vyslání několika prvních bitů, kdy je vysílána adresa přijímací stanice. Pokud by se např. dvě stanice současně pokusily o zápis do stejného obvodu, nastane kolize až při přenosu vlastních zapisovaných dat. V krajním případě, kdy několik stanic současně zapisuje stejná data na stejnou adresu, nemusí být kolize vůbec detekována.

Adresování Každá stanice (v praxi např. každý integrovaný obvod) připojená na I^{2}C má přidělenou 7bitovou adresu. Po zachycení podmínky START porovnávají všechny obvody svou adresu s adresou, která je vysílána na sběrnici. Zjistí-li některý z obvodů shodu, je vysílání určeno právě jemu a musí přijetí adresy potvrdit bitem ACK. Potom přijímá resp. vysílá další data.

Popisované adresování se týká adresy jednotlivých *stanic* na I^{2}C. Je-li stanicí např. paměťový obvod, jedná se o adresu celého obvodu. Adresa jednotlivých *slov* v paměti a požadovaná operace se do paměti přenáší v datové části rámce I^{2}C.

Potvrzování Každý vysílaný byte (včetně adresy) je následován vysláním jednoho bitu ACK. Vysílající stanice jej vysílá v úrovni H. Přijímající stanice potvrzuje přijetí tím, že v době vysílání ACK připojí SDA na úroveň L (viz Obr. 7.15). Pokud vysílající stanice nedostane potvrzení příjmu, ukončí vysílání podmínkou STOP.



Obr.7.15 potvrzování příjmu sběrnice I^2C

Shrnutí pojmů 7

Klíčová slova:
Paralelní a sériový přenos, Handshaking, Baud

Otázky 7

1. Vysvětlete pojem sériový přenos
2. Popište rozdíl mezi asynchronním a synchronním přenosem
3. V čem se vyjadřuje rychlost sériového přenosu
4. Jaké chyby se vyskytují u přenosů dat a jakým způsobem je možné je eliminovat
5. K čem slouží sériová rozhraní.
6. Jakým způsobem je řešena arbitrace u I2C přenosu

Úlohy k řešení 7

63. Asynchronní sériový přenos standartu UART probíhá přenosem:

a datových bitů
b start bitu, datových bitu, parity a stop bitu
c datových bitů a hodinového signálu
d záhlaví, paketu a zápatí

64. Rozhraní SPI slouží k:

a paralelní komunikaci mezi jednotlivými mikropočítačovými moduly na dlouhé vzdálenosti
b sériové komunikaci mezi jednotlivými mikropočítačovými moduly na dlouhé vzdálenosti
c paralelní komunikaci mezi integrovanými obvody
d sériové komunikaci mezi integrovanými obvody

65. Signál MOSI v systémech využívajících sběrnici SPI u obvodů typu Master slouží k:

a k příjmu dat z obvodů typu Slave
b k odesílání dat k obvodům typu Slave
c k výběru obvodů typu Slave (kterýmmu bude posílat data)
d k příjmu dat z obvodů typu Slave i k odesílaní dat k obvodům typu Slave

66. Obvody v systémech využívajících sběrnici SPI jsou propojeny :

a 1 datovým vodičem, 1 hodinovým vodičem a vodičem pro výběr integrovaných obvodů
b datovými vodiči, 1 hodinovým vodičem a vodičem pro výběr integrovaných obvodů
c datovými vodiči, 1 hodinovým vodičem a vodičem pro výběr integrovaných obvodů
d datovými vodiči a 1 hodinovým vodičem

67. Sběrnice SPI je typu:

a multimaster, multislave
b multimaster
c singlemaster, singleslave
d singlemaster

68. V systémech se sběrnicí I^2C jsou jednotlivé stanice připojeny:

a 1 datovým vodičem (SDA) a 1 hodinovým vodičem (SCL)
b 2 datovými vodiči (SDA1 a SDA2) a 1 hodinovým vodičem (SCL)
c 2 datovými vodiči (SDA1 a SDA2)
d 1 datovým vodičem (SDA), 1 vodičem pro výběr stanice(SS) a 1 hodinovým vodičem (SCL)

69. V systémech se sběrnicí I^2C jsou v klidovém stavu (volná sběrnice) vodiče:

a datové v úrovni H, hodinové v úrovni L
b datové v úrovni L, hodinové v úrovni H
c datové i hodinové v úrovni L
d datové i hodinové v úrovni H

70. Každá stanice připojená na sběrnici I^2C má přidělenou vlastní adresu o délce:

a 1 bitu
b 3 bitů
c 7 nebo 11 bitů
d 16 bitů

71. Maximální přípustná frekvence hodinového signálu SCL u rozhraní I^2C je:

a 100 Hz nebo 400 kHz
b 10 kHz nebo 40 kHz
c 100 kHz nebo 4MHz
d 100 kHz nebo 400 kHz

72. Rozhraní I^2C má ve srovnání s rozhraním RS-232:

a proměnnou rychlost komunikace, současné mluvení (přenos z) 2 zařízení
b stálou rychlost komunikace, současné mluvení (přenos ze) 2 zařízení
c stálou rychlost komunikace, současně může mluvit jedno zařízení
d proměnnou rychlost komunikace, současně může mluvit 1 zařízení, adresace v protokolu

73. Přenos pomocí rozhraní RS232 je charakterizován:

a úrovní -3 až -25V odpovídající log.1
b úrovní -3 až -25V odpovídající log.0
c symetrickým vedením
d nesymetrickým vedením

74. Přenos pomocí rozhraní RS 232 je charakterizován

a úrovní –3 až -25V odpovídající logické „1“, na vzdálenost do 1 km, s nesymetrickým přenosem
b úrovní +3 až +25V odpovídající logické „0“, na vzdálenost do 15 m, se symetrickým přenosem
c úrovní +3 až +25V odpovídající logické „1“, na vzdálenost do 15 m, s nesymetrickým přenosem
d úrovní +3 až +25V odpovídající logické „0“, na vzdálenost do 15 m, s nesymetrickým přenosem

75. Přenos pomocí rozhraní RS485 je charakterizován:

a symetrickým vedením
b nesymetrickým vedením
c možnosti přenosu na větší vzdálenosti než RS232
d větší odolnosti proti rušení než RS232

76. Obvod USART je představitelem:

a architektury aritmeticko-logické jednotky
b obvodu pro paralelní synchronní přenos dat
c univerzální asynchronní a synchronní přijímač a vysílač
d univerzální synchronní přijímač a vysílač

77. Obvod typu USART provádí

a převod analogového signálu na digitální
b převod napěťových úrovní TTL a +/-13V
c převod paralelního slova na sériové a naopak
d galvanické oddělení komunikace

78. Obvod MAX232:

a má napěťový výstup s úrovněmi cca ± 9V
b má proudový výstup s úrovněmi cca ± 10mA
c pracuje na principu nábojové pumpy
d galvanicky izoluje vstup a výstup sériového rozhraní

79. 1 baud odpovídá přenosu:

a 1 kbitu za sekundu
b 1 kBytu za sekundu
c bitů za sekundu
d 1 bitu za sekundu

80. Rozhraní USB se vyznačuje:

a možností připojení pouze 1 zařízení, komunikační vzdáleností do 5 m a rychlostí do 1 Mbit/s
b možností připojení více zařízení, komunikační vzdáleností do 15 m a rychlostí do 480 Mbit/s
c možností připojení více zařízení, komunikační vzdáleností do 0,5 m a rychlostí do 1 Mbit/s
d možností připojení více zařízení, komunikační vzdáleností do 5 m a rychlostí do 480 Mbit/s

81. Rozhraní USB obsahuje:

a napájení +5 V
b napájení +15 V
c napájení +12 V
d neobsahuje napájení

82. Kód CRC

a je kód využívající se pro kódování informace z inkrementálního čidla
b slouží k účinné detekci vzniklých chyb včetně chybových shluků
c je samoopravný kód používající se při diskových operacích
d kód využívající kontrolní součet

83. Poruchy působící na přenos informace lze podle charakteru kategorizovat na

a neharmonické, harmonické náhodné
b harmonické, fluktuační, náhodné, souvislé
c impulsní, harmonické, fluktuační
d dočasně působící, průmyslové

84. Poruchy působící na přenos informace lze podle jejich zdroje kategorizovat na

a přírodního původu, dočasně působící a trvale působící
b přírodního původu, průmyslového původu a vznikající v zařízení
c fluktuační, harmonické a impulsní
d impulsní, dočasně působící a trvale působící

85. Poruchy působící na přenos informace lze podle časového výskytu kategorizovat na

a dočasně působící a impulsní
b impulsní a harmonické
c dočasně působící a trvale působící
d přírodního původu a průmyslového původu

86. Paritní bit

a určuje, zda je počet jeniček v přenášeném slově lichý nebo sudý
b využívá se při kontrole matematických operací
c slouží k detekci přenosu
d vyjadřuje kontrolní součet

8. POLOVODIČOVÉ PAMĚTI

Čas ke studiu: 1,5 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- vysvětlit kritéria rozdělení polovodičových pamětí
- popsat základní vlastnosti polovodičových pamětí
- princip programování pamětí typu EPROM, EEROM, FLASH

Výklad

Abstraktní pojetí paměti jako schopnosti se uplatňuje i v technice, ale v rámci mikropočítačů přikládáme slovu paměť obvykle význam konkrétní, neboť označuje neživý hmotný objekt, který má tuto schopnost. Jinak řečeno slovo paměť se užívá jako označení paměťového zařízení libovolného typu, případně i paměťové součástky.

Nejhrubší dělení pamětí je na vnitřní a vnější paměti mikropočítače. Vnitřní paměti jsou zpravidla polovodičové a vnější obsahují obvykle pohyblivé části. V této kapitole se budeme věnovat právě polovodičovým pamětem.

8.1 Rozdělení pamětí podle přístupu

Přístupem (access) k paměti rozumíme zápis dat do určitého místa v paměti nebo čtení dat z tohoto místa, s cílem uložit data z procesoru nebo periferního zařízení do paměti, nebo je naopak přenést z paměti opačným směrem.

❑ Paměti s libovolným přístupem

Tyto paměti jsou označovány zkratkou RAM (Random Access Memory). U pamětí RAM nazýváme přístup libovolným, resp. náhodným proto, že jednotlivá místa takové paměti se od sebe liší jen adresou, kterou lze volit při každém přístupu libovolně a nezávisle na adresách použitých při předchozích nebo následujících přístupech. Paměťová místa jsou si rovnocenná co do trvání přístupu i způsobu řízení

❑ Paměti se sériovým přístupem

Sériový neboli sekvenční přístup vede na zkratku SAM (Serial Access Memory). Sériový přístup znamená, že adresy nelze generovat libovolně, ale sekvenčně v souladu s posloupností uložených dat v paměťových místech. V oblasti polovodičových pamětí je klasickým představitelem posuvný registr. Patří sem i některé paměti se speciálním přístupem, viz. následující kapitola. Tento druh pamětí nelze však zaměňovat s pamětmi RAM, u nichž je zápis a čtení prováděno po jednom vodiči. Adresa a data jsou zde přenášeny bit po bitu.

❑ Paměti se speciálními způsoby přístupu

a) Dvoubránové a vícebránové paměti

b) Asociativní paměti, resp. paměti CAM – paměti adresované obsahem

c) Paměti typu zásobník, nazývané též sklípková paměť, Stack nebo LIFO (Last In – First Out)

d) Paměti typu fronta, označované jako FIFO (First In – First Out)

e) Paměti s kombinovaným řízením, které umožňují libovolný přístup a v zásadě patří mezi RAM, ale jejich obvody umožňují přepnutí na automatickou inkrementaci/dekrementaci adresy. Pak dovolují po dodání jediné adresy realizovat rychlý sekvenční přístup k celé posloupnosti sousedních paměťových míst (přenos bloku dat).

8.2 Rozdělení podle možnosti zápisu/čtení

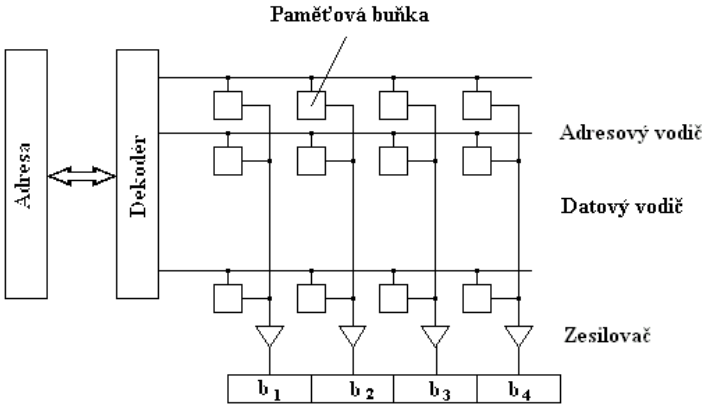
Prakticky každá paměť ve vztahů k datům plní tři funkce:

- zápis
- pamatování
- čtení

Jsou však paměti, které za běžných provozních podmínek jsou schopny jen posledních dvou funkcí. Zápis je proveden jen jednou.

❑ Paměti pro záznam a čtení RWM (Read Write Memory)

U těchto pamětí je možno zapisovat i číst za běžného provozu, kdy záznam i čtení trvají přibližně stejnou dobu. Polovodičové paměti RWM bývají energeticky závislé, tj. po vypnutí se uchovávaná informace ztratí. Pro trvalé uchování informace je nutno tyto paměti napěťově zálohovat. Dnes již zastaralý typ feritových pamětí byl typem RWM, ale nebyl energeticky závislý.

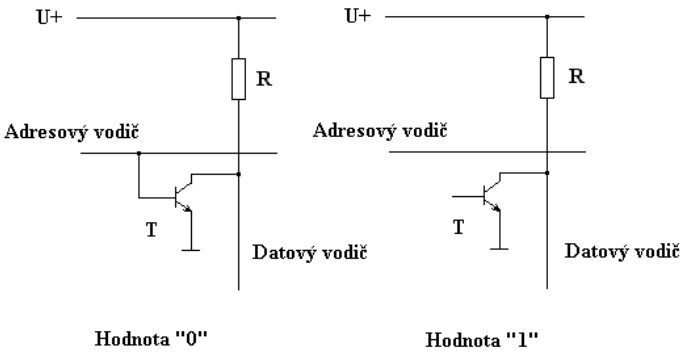


Obr. 8.1 Realizace paměti RWM

❑ Pevné, resp. permanentní paměti ROM

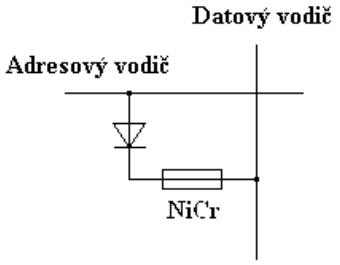
U těchto pamětí je možno za provozu jen číst. Data však z nich nezmizí ani při ztrátě napájení a tato energetická nezávislost je jednou a možná i jedinou z hlavních předností těchto pamětí. Tyto paměti pak ještě dělíme na:

a) ROM – informace se do této paměti zaznamenává při výrobě. Tyto druhy paměti nalézáme především u jednočipových mikropočítačů používaných v aplikačních zařízeních.



Obr. 8.2 Realizace paměťové buňky ROM

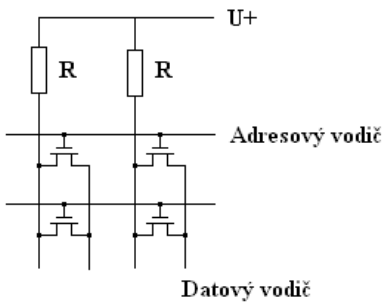
b) PROM, čili programovatelná ROM – informaci do ní může zaznamenat uživatel.



Obr. 8.3 Realizace paměťové buňky PROM

c) EPROM, mazatelné a znovu programovatelné ROM – informaci do ní programuje uživatel a navíc se tato může vymazat pomocí ultrafialového záření přes okénko na pouzdře. Pak je možno do této paměti opět zaznamenat další informaci.

d) EEPROM, elektricky mazatelná a programovatelná ROM – informace může být v této paměti přepisována při zachování elektrické nezávislosti při uchovávání informace. Při současném stupni vývoje však může dělat zápis jen ve speciálním režimu se zvýšeným napětím signálů a především výrazně pomaleji než probíhá čtení. Vývoj však spěje k tzv. bleskovým EEPROM (Flash Memory), které tuto zřejmou nevýhodu potlačují. Data lze přepisovat sice mnohem pomaleji než číst, ale podstatně rychleji než u dosavadních EEPROM a to ve standardním režimu bez zvyšování napětí signálů. Rovněž počet možných cyklů přepisování je mnohem větší.



Obr. 8.4 Realizace paměťové buňky EEPROM

8.3 Rozdělení pamětí podle principu elementární paměťové buňky

❑ Paměti SRAM (Static Random Access Memory)

Paměti SRAM uchovávají informaci v sobě uloženou po celou dobu, kdy jsou připojeny ke zdroji elektrického napájení. Paměťová buňka SRAM je realizována jako bistabilní klopný obvod, tj. obvod, který se může nacházet vždy v jednom ze dvou stavů, které určují, zda v paměti je uložena 1 nebo 0.

U SRAM pamětí se používá dvou datových vodičů. Vodič Data je určený k zápisu do paměti. Vodič označený jako \Data se používá ke čtení. Hodnota na tomto vodiči je vždy opačná než hodnota uložená v paměti. Takže na konci je nutno ji ještě negovat. Při zápisu se na adresový vodič umístí hodnota logická 1. Tranzistory T_1 a T_2 se otevřou. Na vodič Data se přivede zapisovaná hodnota (např. 1). Tranzistor T_1 je otevřen, takže jednička na vodiči Data otevře tranzistor T_4 a tímto dojde k uzavření tranzistoru T_3. Tento stav obvodu představuje uložení hodnoty 0 do paměti. Zcela analogicky tato buňka pracuje i při zápisu hodnoty 1. Rozdíl je pouze v tom, že tranzistor T_4 zůstane uzavřen a to způsobí otevření tranzistoru T_3.

a) b)

Obr. 8.5 Realizace jedné buňky SRAM v technologii a) MOS; b) TTL

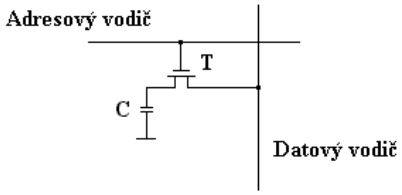
Při čtení je opět na adresový vodič přivedena hodnota logická 1, což opět způsobí otevření tranzistorů T_1 a T_2. Jestliže byla v paměti zapsána hodnota 1, je tranzistor T_4 otevřen (tj. na jeho výstupu je hodnota 0). Tuto hodnotu obdržíme na vodiči \DATA. Opět zcela analogicky v případě uložené hodnoty 0, kdy tranzistor T_4 je uzavřen (tj. na jeho výstupu je hodnota 1).

Poznámka: Tranzistory T_5 a T_6 plní pouze funkcí rezistorů.

Paměti SRAM je možné uskutečnit i v technologii TTL. Buňka takovéto paměti pracuje na podobném principu jako buňka v technologii MOS

❑ Paměti DRAM (Dynamic Random Access Memory)

V paměti DRAM je informace uložena pomocí elektrického náboje na kondenzátoru. Tento náboj má však tendenci se vybíjet i v době, kdy je paměť připojena ke zdroji elektrického napájení. Aby nedošlo k tomuto vybití a tím i ke ztrátě uložené informace, je nutné periodicky provádět tzv. **refersh**, tj. oživování paměťové buňky. Tuto funkci plní některý z obvodů čipové sady.



Obr. 8.6 Realizace jedné buňky paměti DRAM v technologii TTL

Při zápisu se na adresový vodič přivede hodnota logická 1. Tím se tranzistor T otevře. Na datovém vodiči je umístěna zapisovaná hodnota (např. 1). Tato hodnota projde přes otevřený tranzistor a nabije kondenzátor. V případě zápisu nuly dojde pouze k případnému vybití kondenzátoru (pokud byla dříve v paměti uložena hodnota 1).

Při čtení je na adresový vodič přivedena hodnota logická 1, která způsobí otevření tranzistoru T. Jestliže byl kondenzátor nabitý, zapsaná hodnota přejde na datový vodič. Tímto čtením však dojde k vybití kondenzátoru a zničení uložené informace. Jedná se tedy o buňku, která je destruktivní při čtení a přečtenou hodnotu je nutné opět do paměti zapsat.

Buňka paměti DRAM je velmi jednoduchá a dovoluje vysokou integraci a nízké výrobní náklady. Díky těmto vlastnostem je používána k výrobě operačních pamětí. Její nevýhodou je však vyšší přístupová doba (60 - 70 ns) způsobená nutností provádět refresh a časem potřebným k nabití a vybití kondenzátoru.

Shrnutí pojmů 8

Klíčová slova:
Polovodičová paměť

Otázky 8

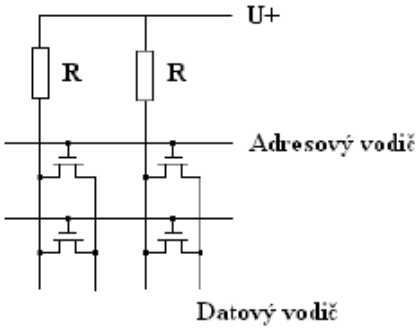
1. Vysvětlete princip polovodičové paměti
2. Vysvětlete rozdíl mezi pamětmi RWM a ROM
3. Jaký je principielní rozdíl mezi statickými a dynamickými pamětmi

Úlohy k řešení 8

87. Použitím sériových pamětí proti paralelním se počet datových, adresních a řídích vývodů :

a sníží
b zvýší
c nezmění
d podle typu sériového přenosu

88. Na obrázku je znázorněna realizace paměťové buňky typu:



a ROM
b EEPROM
c PROM
d RWM

89. Paměť EPROM

a se může pouze číst
b můžeme elektricky číst i mazat
c se ztrátou napětí ztrácí svá data
d umožňuje mazání dat UV zářením

90. U paměti EPROM

a probíhá zápis mnohem pomaleji než čtení
b lze provést zápis jen jednou
c nelze provést zápis
d je zápis mnohem rychlejší než čtení

91. Paměť ROM:

a slouží jako paměť zásobníku
b může obsahovat konstanty a tabulky
c slouží pro uložení proměnných
d uchovává data i bez napájecího napětí

92. Obvod 27C256 je

a paměť EPROM o kapacitě 256 kB
b paměť RAM o kapacitě 256 kb
c paměť RAM o kapacitě 256 kB
d paměť EPROM o kapacitě 32 kB

93. Paměť s označením 2764 je:

a paměť PROM s kapacitou 8 kByte
b paměť EPROM s kapacitou 64 kbitů
c paměť statická RAM s kapacitou 64kByte
d paměť EROM s kapacitou 64 kByte
e paměť ROM s kapacitou 8 kByte

94. Paměť s označením 61256 s šířkou datového slova 8 bitů má

a 16 adresových vývodů
b 15 adresových vývodů
c 8 adresových vývodů
d 256 adresových vývodů

95. Paměť s označením 2764 s šířkou datového slova 8 bitů má

a 13 adresových vývodů
b 12 adresových vývodů
c 8 adresových vývodů
d 64 adresových vývodů

9. STYK MIKROPOČÍTAČE S ANALOGOVÝM PROSTŘEDÍM

Čas ke studiu: 3 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- způsoby napojení mikropočítače na analogové prostředí
- principy D/A převodů
- principy A/D převodů
- způsoby zobrazení čísel v mikropočítači

Výklad

Mnoho zdrojů informace v okolí mikropočítače má spojitý, analogový výstup. Abychom číslicovému rozhraní mikropočítače umožnili zpracování těchto informací, musíme realizovat především převod analogových signálů na číslicové. Podobně pro řízení analogových akčních členů, které má mikropočítač ovládat, musíme jeho rozhraní vybavit převodníky číslicových signálů na analogové.

Pro napojení mikropočítače na analogové prostředí, však ještě musíme tyto převodníky doplnit dalšími obvody, jež budou muset zajišťovat některé z následujících činností:

- zesílení energie signálu
- převod signálu z jiného fyzikálního nositele
- linearizaci čidla
- potlačení souhlasného signálu a šumu
- izolaci různorodých zemí

- úpravu kmitočtového spektra signálu – filtraci
- volbu měřicího kanálu – multiplex
- normování úrovně signálu
- vzorkování signálu

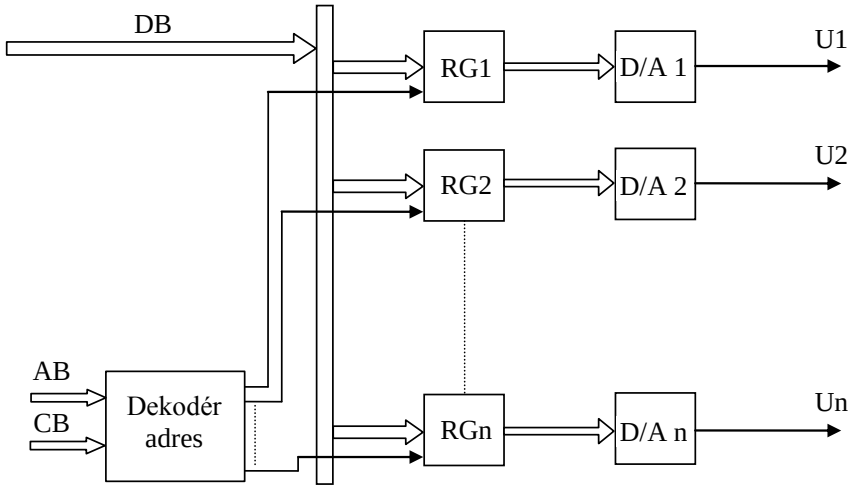
Některé z operací pak nelze řešit jinak než použitím specializovaných obvodů. Jiné úlohy mohou být řešeny diskrétními prvky anebo do jisté míry jej lze řešit i softwarově. Volba hranice mezi hardwarovým a softwarovým řešením obsluhy styku mikropočítače s analogovým okolím závisí především na zvážení časové náročnosti operace realizované jedním z uvedených způsobů a na stupni využití mikropočítače jinými úlohami. Vysoký stupeň integrace však dovoluje řešit tyto obvody přímo v rámci čipu mikropočítače.

9.1 Analogový výstup mikropočítače

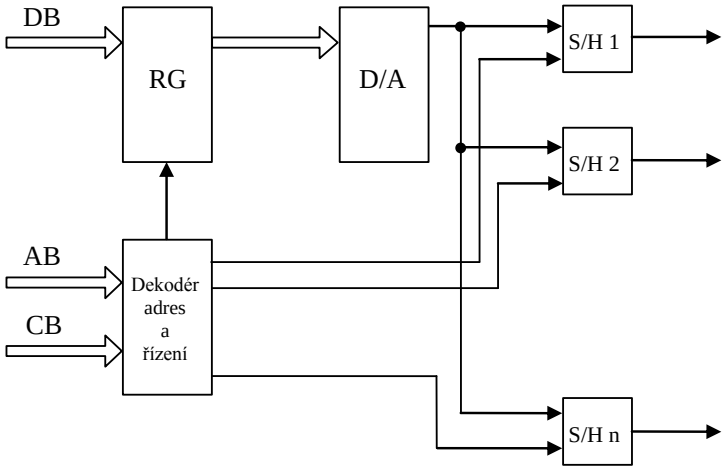
Řešení výstupních analogových kanálů je možné řešit dvěma způsoby:

- paralelním řazením více D/A převodníků
- centrálním D/A převodníkem s demultiplexovanými výstupy ve formě analogových pamětí

Možnosti jsou blokově znázorněny na obr. 9.1 resp. obr. 9.2. Výstupní signál bývá zpravidla normován do úrovní ±10V eventuelně 0÷20mA. Další úprava – např. výkonové zesílení – je pak věcí místního akčního členu, který je výstupem ovládán.



Obr. 9.1 Analogové výstupy s oddělenými D/A převodníky



Obr. 9.2 Analogové výstupy s využitím vzorkovacích zesilovačů

Jako analogový výstup mikropočítače je samozřejmě použitelná jakákoliv jiná jednotka s číslicovým vstupem a analogovým výstupem. Příkladem mohou být programovatelné zdroje napětí, proudu, kmitočtu atd. Jejich spojení a komunikaci s mikropočítačem pak řešíme jako připojení číslicových výstupů.

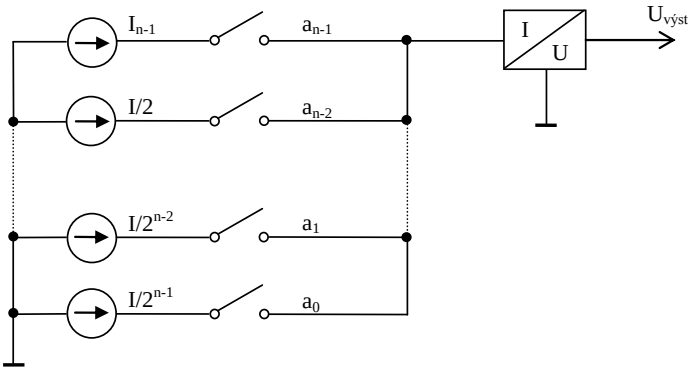
9.1.1 Číslicově analogový (D/A) převod

Je řada možností, jak převést číslicovou veličinu na analogovou a naopak. V jednoduchých aplikacích, kdy není mikropočítač využit řízeným procesem, může se více či méně využít pro dílčí funkce převodníku (viz. převodník tvořený filtrací digitálního výstupu na němž je signál s proměnnou střídou, které je pak úměrná velikost analogového signálu). V častějších případech, zejména kdy je mikropočítač plně vytížen, jsou převody zajišťovány vnějšími obvody – převodníky – a vhodnými vazebními obvody.

Při realizaci D/A převodu musíme přiřadit n-bitovému číslu 2^n hodnot analogové veličiny. Využijeme toho, že celé číslo si můžeme rozdělit na jednotlivé bity, které mají svou váhu a číslo je tak vyjádřeno sumou:

$$x = \sum_{i=0}^{n-1} a_i \cdot 2^i$$

Nahradíme-li číselnou váhu jednotlivých bitů odpovídajícími kvanty, např. el. proudu, stačí proudové zdroje doplnit spínači ovládanými příslušnými bity převáděného čísla, sečtením vytvořit výsledný proud a převodník je v principu hotov. Doplníme ho ještě obvykle převodníkem proud-napětí. Princip takového převodníku je uveden na obr. 9.3.

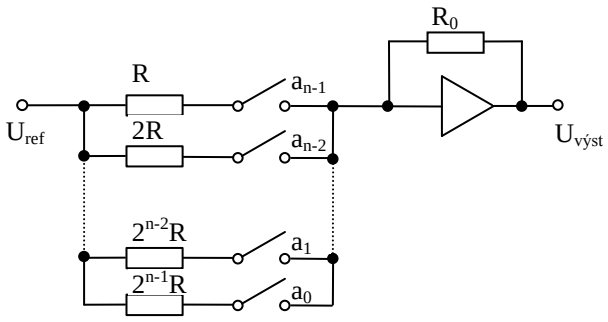


Obr. 9.3 Princip číslicově-analogového převodníku

Převod probíhá podle vztahu:

$$I_{výst.} = \sum_{i=0}^{n-1} a_i \frac{I}{2^{n-i-1}}$$

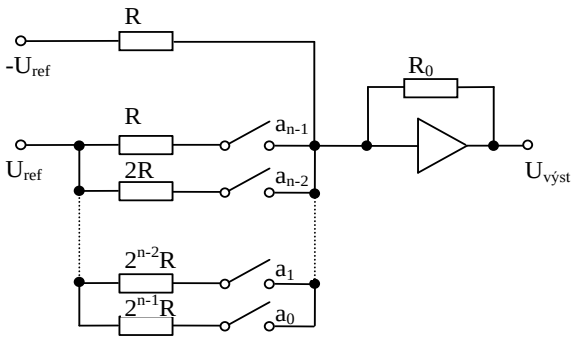
Praktická realizace převodníku je trochu jiná. Zdroje proudu jsou nahrazeny jedním referenčním napětím a váhovými odpory spínanými jednotlivými bity na vstup operačního zesilovače. Převodník s takovouto odporovou sítí je na obr. 9.4.



Obr. 9.4 D/A převodník s odporovou váhovou sítí

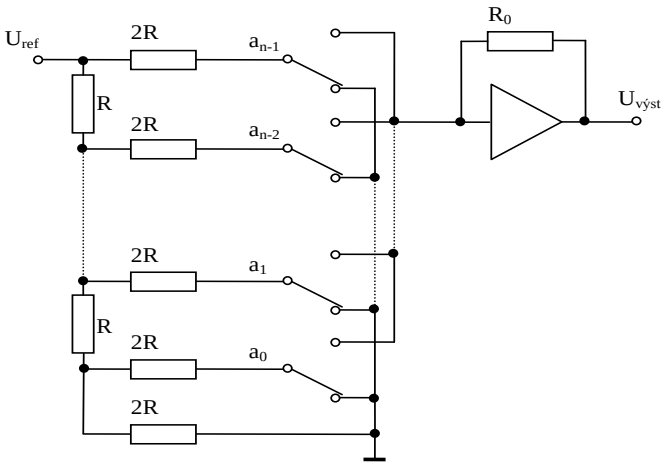
Výstupní napětí převodníku pak odpovídá hodnotě převáděného čísla.

$$U_{výst} = -U_{ref} \cdot \frac{R_0}{2^{n-1} \cdot R} \cdot \sum_{i=0}^{n-1} a_i \cdot 2^i$$



Obr. 9.5 Oboupolaritní D/A převodník

Při praktické realizaci převodníku jako jediného integrovaného obvodu ještě vadí velký rozdíl hodnot váhových odporů od R do 2^{n-1}R. Proto se s výhodou využívá kombinované odporové sítě R-2R, jejíž uspořádání je na obr. 9.6.



Obr. 9.6 D/A převodník s odporovou sítí R-2R

Pro výstupní proud a napětí platí:

$$I_{výst} = \sum_{i=0}^{n-1} a_i \frac{U_{ref}}{2R \cdot 2^{n-i-1}} = \frac{U_{ref}}{2^n \cdot R} \cdot \sum_{i=0}^{n-1} a_i \cdot 2^i$$

$$U_{výst} = -U_{ref} \cdot \frac{R_0}{2^n \cdot R} \cdot \sum_{i=0}^{n-1} a_i \cdot 2^i$$

Pro praktické využití převodníků je nezbytné porozumět jejich parametrům a uvést je do souvislosti s požadavky řešené aplikace.

❑ Rozlišovací schopnost

Je jedním z nejdůležitějších parametrů. Vyjadřujeme ní počet diskrétních stupňů výstupního analogového napětí. Rozlišovací schopnost je v přímé souvislosti s počtem bitů, ze kterých sestává převáděné slovo. Např. osmibitový převodník má 2^8 = 256 stupňů, což odpovídá rozlišovací schopnosti 100/256 = 0,4%.

❑ Přesnost

Je dalším důležitým parametrem D/A převodníku. V praxi se skutečná převodní charakteristika může lišit od ideální v důsledku různých vlivů. Celková přesnost převodníku podstatně závisí na stabilitě referenčního napětí. Statická nestabilita referenčního napětí ovlivňuje přesnost, nemá však vliv na linearitu a rozlišovací schopnost.

❑ Rychlost převodu

Je určena počtem vstupních slov převodníku, která mohou být tímto převodníkem převedena na analogové napětí nebo proud za jednotku času. Někdy se též uvádí doba převodu, což je reciproká hodnota rychlosti převodu – je to doba mezi přivedením vstupního slova na převodník a okamžikem dosažení ustáleného napětí nebo proudu.

❑ Rozsah

Je definován jako maximální napěťový rozkmit převodníku buď jedné nebo obou polarit.

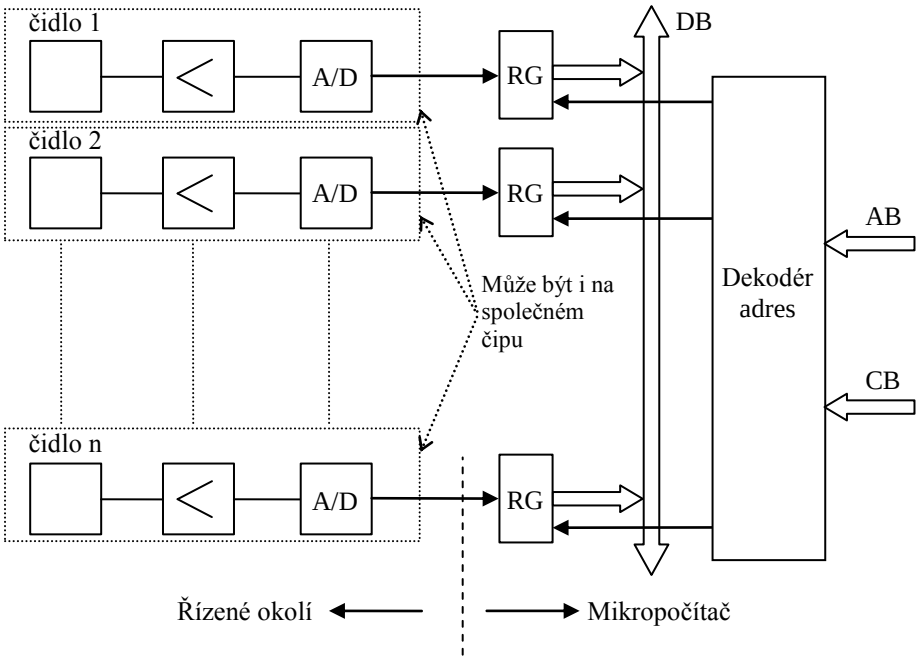
Výstupní napětí D/A převodníku je schodovité, jeho hodnoty mohou nabývat jen diskrétních hodnot. Změna jednoho diskrétního stupně odpovídá nejnižšímu bitu vstupního slova. Chyba způsobená diskrétními úrovněmi výstupního napětí může dosáhnout maximálně ±1/2hodnoty odpovídající nejnižšímu bitu vstupního slova.

Připojení D/A převodníku k datové sběrnici mikropočítače je realizováno prostřednictvím registru, který vlastně funguje jako výstupní brána a zároveň jako vyrovnávací registr pamatující si poslední hodnotu. Starší převodníky tento registr neměly a bylo tedy nutné jej tímto registrem doplnit. Dnes jsou běžné monolitické převodníky zahrnující jak tento registr tak i přesný zdroj referenčního napětí v jednom pouzdře. Některé jednočipové mikropočítače obsahují již ve své struktuře D/A převodník. Není to však tak běžné jako přítomnost A/D převodníku na čipu.

9.2 Analogový vstup mikropočítače

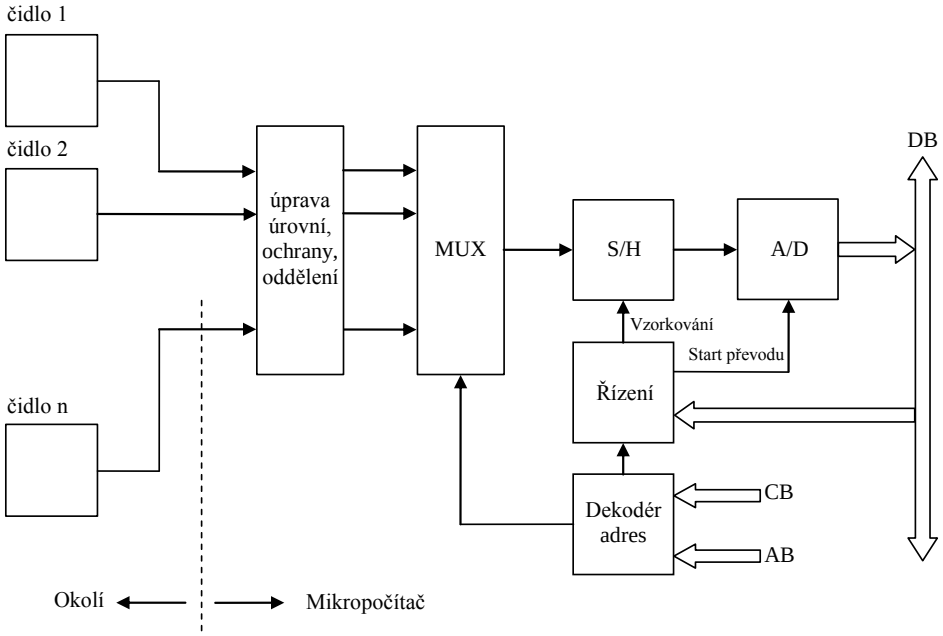
Řídicí mikropočítačový systém bývá mnohdy řešen jako decentralizovaný systém, který s výhodou využívá možnosti číslicového přenosu dat mezi odlehlými zdroji a příjemci informace. Malé rozměry a spolehlivost mikropočítačů umožnily přiblížit zpracování analogových vstupních veličin místu jejich vzniku a zvýšit tak jejich přesnost a spolehlivost měření.

Zdroji analogové informace – zpravidla proudu nebo napětí – jsou čidla fyzikálních veličin umístěná v řízeném okolí. Výstupní signály z čidel je třeba zesílit a normovat do konvenčních rozsahů el. veličin. V místě čidla pak může být realizován i A/D převod. Toto řešení je vhodné pro rozsáhlé systémy. Místní převodník bývá vybaven výstupním registrem, jehož obsah je neustále obnovován. Přenos výstupních dat z převodníku do mikropočítače je většinou sériový.



Obr. 9.7 Decentralizované analogové vstupy mikropočítačového systému

V prostorově soustředěných soustavách, vybavených čidly stejného druhu, je vhodné tato čidla připojit na analogový vstup mikropočítače pomocí vhodného multiplexoru. A/D převodník pak bývá jen jeden a postupně převádí jednotlivé měřicí kanály. Toto řešení se pak nazývá centralizovaný systém. Velmi často je tento systém patrný u jednočipových mikropočítačů s integrovaným A/D převodníkem na čipu. Bývá jen jeden s možností několika měřených kanálů. Někdy bývá nutno u tohoto řešení provést zachycení všech veličin v jeden okamžik. To pak vyžaduje vzorkovací obvod pro každý měřený obvod zvlášť.



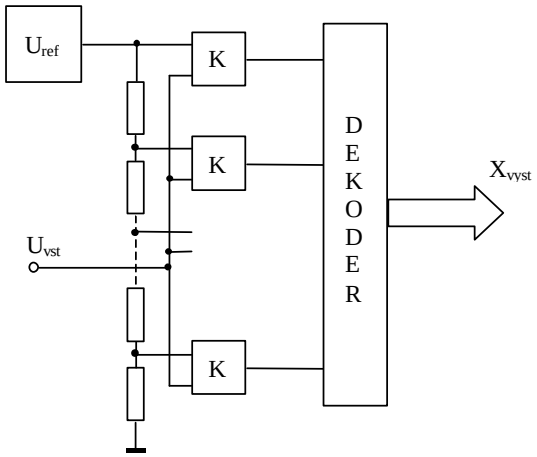
Obr. 9.8 Centralizovaná vstupní jednotka s A/D převodníky

9.2.1 *Analogově číslicový (A/D) převod.*

Analogově číslicový převodník je obvod, který generuje výstupní číslo odvozené ze vstupního analogového signálu vzhledem k analogové referenční veličině. Výstupní číslo může být udáno v paralelním nebo sériovém tvaru. Používané kódy jsou stejné jako u D/A převodníků (viz násl. podkap.). Vstupním analogovým signálem je zpravidla napětí. Analogová reference bývá většinou pevná, méně často proměnná (poměrový A/D převodník). Existuje celá řada principů převodu.

❑ Paralelní A/D převodník

Jeho princip je velmi jednoduchý. Vytvoříme tolik referenčních napětí, kolik hodnot chceme rozeznávat.



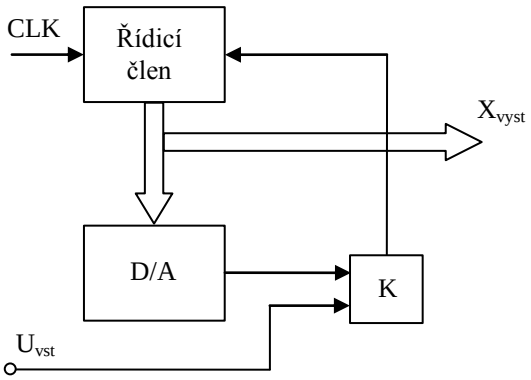
Obr. 9.9 Paralelní A/D převodník

Pak určíme, mezi kterými je dané převáděné napětí. Realizace je naznačena na obr. 9.9. Referenční napětí rozdělíme odporovým děličem složeným ze shodných odporů na kvanta odpovídají hodnotě jednoho bitu. Odporů bude 2^n.

Převáděné napětí je přiváděno na vstup každého komparátoru, na jejichž druhý vstup je zaveden příslušný díl referenčního napětí. Komparátor vyhodnocuje, které napětí na jeho vstupu je větší. Nejvyšší překlopený komparátor určuje výsledek převodu. Výstupy z komparátoru jsou přivedeny na kodér, jenž vytváří z dané kombinace vstupu n-bitové číslo. Převodníky se vyznačují velmi vysokou rychlostí, nevýhodou je pak velký počet komparátorů (2^n-1). V současné době se vyskytují tyto převodníky v monolitické formě se všemi komparátory na čipu. Jejich rychlost se pohybuje v jednotkách nanosekund.

❑ Kompenzační A/D převodík

Druhou skupinu převodníků tvoří převodníky kompenzační. Jeho princip je vyznačen na obr. 9.10. Součástí tohoto převodníku je D/A převodník, jehož výstup je porovnáván v komparátoru s převáděným vstupním signálem. Vstup tohoto D/A převodníku je připojen k řídicímu členu, jehož úkolem je vytvářet takové číslo, které po převedení na analogovou hodnotu D/A převodníkem je stejné s hodnotou vstupního signálu. Existuje několik strategií, jak se řídicí člen k takovému číslu přibližuje.



Obr. 9.10 Princip kompenzačního A/D převodníku

Metoda vzestupného čítání – inkrementační

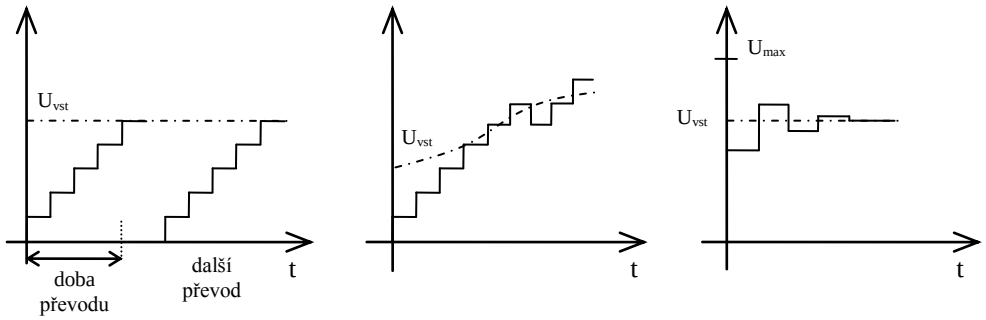
Řídicím členem je vzestupný čítač, který při každém převodu začíná čítat od nuly. S narůstáním jeho obsahu roste i napětí na výstupu D/A převodníku a jakmile je dosaženo stejného napětí jako je měřené (převáděné), dojde k překlopení komparátoru, což je signál zpětně pro řídicí člen k ukončení čítání. Nyní je možné číst na výstupu čítače výsledek převodu. Doba převodu je úměrná velikosti převáděného napětí a je k-násobkem periody hodinového pulsu čítače, kde k je počet kroků nutných k uskutečnění převodu. Variantou této metody je metoda dekrementační.

Metoda sledovací

Předchozí metodu lze poměrně jednoduše modifikovat tak, že použijeme obousměrný čítač, který snižuje či zvyšuje svou hodnotu podle stavu komparátoru. Přes poměrně malou změnu má převodník podstatně jiné vlastnosti. Řídicí člen se na počátku nenuluje, ale vychází z počáteční hodnoty dané výsledkem předchozího kroku převodu. Převodník tak sleduje trvale převáděný signál a dává téměř okamžité výsledky. Výjimkou je start převodníku a stavy, kdy se signál mění rychleji než je schopen převodník sledovat. Tuto vlastnost pak s výhodou využijeme tak, že navrhneme převodník podle nejvyšší strmosti signálu, která se může vyskytnout. Převodník pak nebude reagovat (nebo jen minimálně) na krátké rušivé signály.

Metoda postupné aproximace

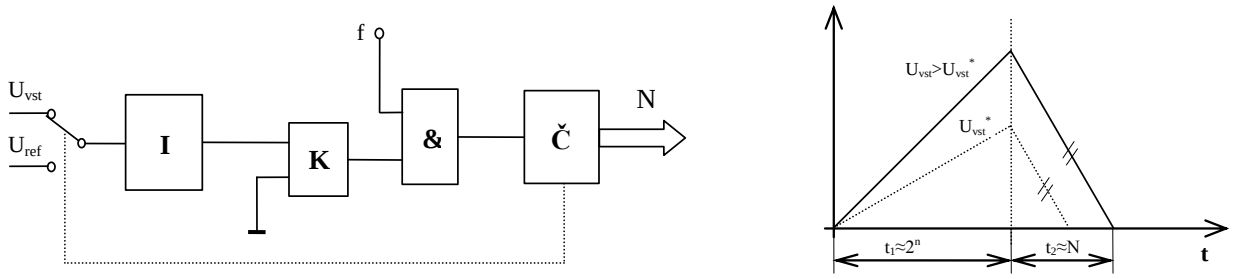
Také tato metoda odstraňuje neurčitou dobu převodu u inkrementační metody. Vychází vlastně z iterační metody půlení intervalu. Řídicí blok, který se nazývá registr postupných aproximací, nastaví nejprve nejvyšší bit na „1“ a ostatní na „0“ Tak vlastně vytvoří na výstupu D/A převodníku polovinu maximálního napětí. Komparátor rozhodne, ve které polovině intervalu se nachází převáděný signál a podle toho se nejvyšší bit nuluje nebo nechává nastaven a současně se nastavuje druhý nejvyšší bit. Algoritmus se opakuje. Takto je v n krocích realizován n-bitový převodník. Pro běžné převodníky se doba převodu pohybuje v jednotkách mikrosekund někdy i stovkách nanosekund. Tento typ převodníků je nejčastěji používán právě ve spojení s mikropočítačovým řídicím systémem. Jeho rychlost i přesnost vyhovuje většině řídicích aplikací.



Obr. 9.11 Způsoby řízení kompenzačních A/D převodníků

❑ Integrační převodník

Tyto převodníky můžeme charakterizovat poměrně dlouhou dobou převodu řádu 1 až 10 ms, značnou dosažitelnou přesností a obvodovou jednoduchostí. Setkáváme se s nimi především v měřicí technice. Převodník pracuje zpravidla ve dvou nebo třech krocích. Základem převodníku je integrátor I, který integruje měřený signál $U_{vst.}$ po konstantní dobu danou naplněním n-bitového čítače Č hodinovými pulsy frekvence f. Velikost výstupního napětí integrátoru je úměrná střední hodnotě měřeného (převáděného) napětí. Po naplnění čítače se přepne vstup integrátoru k referenčnímu napětí opačné polarity a čítač odměřuje čas, za který dosáhne výstup integrátoru nulové hodnoty. Pak se čítání zastaví a obsah čítače je v podstatě výsledné převedené číslo N. Tato hodnota je úměrná střední hodnotě vstupního převáděného napětí. Princip převodníku ilustruje obr. 9.12.



Obr. 9.12 Princip integračního převodníku

Jako pomocný třetí krok může předcházet vynulování zbytkových napětí integrátoru, komparátoru a vstupního zesilovače.

Typickými představiteli těchto převodníků jsou monolitické obvody ICL7106, ICL7107, ze starších pak velmi populární C520D. Tyto pracují se třemi až čtyřmi měřeními za sekundu a umožňují oboupolaritní vstupní signál a jejich výstup přímo podporuje napojení zobrazovacích LCD nebo LED jednotek.

A/D převodníky charakterizují nejdůležitější dva parametry.

❑ Rychlost převodu

Z hlediska dynamických chyb je vhodné oddělit převodníky integračního typu od ostatních. U integračních převodníků lze s výhodou využít jejich filtračních vlastností. Potlačují totiž periodické průběhy s násobkem periody rovným době integrace. Protože se nám zpravidla jedná o filtraci rušivých napětí elektrovodné sítě, které se superponují na měřený signál, budeme volit dobu integrace např. 20ms. Doba převodu je pak dána dobou integrace a dobou trvání ostatních taktů převodu (integrace reference, nulování offsetu apod.)Rychlost převodu je obrácená hodnota doby převodu.

V případě požadavku na převod okamžitých hodnot analogové vstupní veličiny použijeme neintegrační typ převodníku, doplněný vstupním vzorkováním a paměťovým obvodem.

❑ Rozlišovací schopnost

Je určena počtem úrovní, na něž jsme rozdělili rozsah vstupní analogové veličiny; udává se zpravidla přibližně výrazem $1/2^n$, kde n je počet bitů výstupního slova

❑ Kvantizační chyba

Je v přímé souvislosti s počtem úrovní kvantování. Dosahuje hodnoty ±1/2 LSB výstupního slova převodníku.

❑ Linearita

Rozumíme jí maximální odchylku od přímky, spojující dva koncové body převodní charakteristiky vstup – výstup. Vyjadřuje se ve zlomcích LSB. Je neovlivnitelným parametrem A/D převodníku.

Chyby vznikající napěťovým posunem nebo změnou měřítka převodu je možné kompenzovat nastavovacími prvky převodníku (nastavením nuly a zisku).

9.3 Kódy pro A/D a D/A převody

Pro kódování analogových hodnot v číslicových systémech se využívá několik možností. Pro názornost je nutné objasnit zobrazení čísel v mikropočítačových systémech

9.3.1 Vyjádření čísel v číslicových systémech

Pro zobrazení hodnot signálů číslicového systému je možné použít libovolné číselné soustavy. S ohledem na rozvoj a realizaci číslicových obvodů se dvěma stabilními stavy, má praktický význam se zabývat pouze soustavou dvojkovou. Dvojková binární soustava je číselná soustava jejíž základem je hodnota B = 2. Jako v každé jiné soustavě je číslo tvořeno posloupností číslic (bitů), které nabývají pouze dvou hodnot 0 nebo 1. Nyní se seznámíme s vyjádřením kladných i záporných čísel s pevnou řádovou čárkou.

❑ Nezáporná celá čísla

Tato interpretace je nejpřirozenější a na první pohled jediná rozumná. Popišme si nyní, jak jsou tato čísla v počítači přiřazena dvojkovým číslům.

Každému bitu přiřadíme určitou váhu vyjádřenou jako mocnina dvou. Nejméně významný bit a_0 má váhu 2^0 další bit a_1 má váhu 2^1 atd. Číselnou hodnotu n-bitového čísla pak vyjádříme vztahem:

$$x = \sum_{i=0}^{n-1} a_i \cdot 2^i$$

Stanovme pak další informace:

- nejmenší číslo 0
- největší číslo 2^n-1
- rozdíl dvou sousedních čísel 1
- počet rozlišitelných čísel 2^n

Tato čísla bývají označována jako INTEGER bez znaménka

❑ Nezáporná reálná čísla

Rozšířením oboru zpracovávaných čísel na čísla reálná je nesporně užitečné. Dosáhneme ho tak, že nejnižšímu bitu nepřiřadíme váhu 2^0, ale 2^{-m}, kde m je celé kladné číslo. Tím jsme vlastně dovnitř dvojkového čísla vložili řádovou čárku tak, že vpravo od ní zůstává m bitů, jejichž váha je tvořena zápornými mocninami 2 tj. zlomkovými čísly ½, ¼ atd. Vztah pro výpočet hodnoty, kterou pak číslo v mikropočítači představuje bude

$$x = \sum_{i=0}^{n-1} a_i \cdot 2^{i-m}$$

A opět:

- nejmenší číslo 0
- největší číslo $2^{-m}.(2^n-1)$
- rozdíl dvou sousedních čísel 2^{-m}
- počet rozlišitelných hodnot 2^n

Poznámka:

Pokud si zkusíte takto pro nějaké *m* převést do dekadického tvaru nějaké binární číslo, zjistíte, že může mít v desítkové interpretaci hodně desetinných míst. Neznamená to ovšem, že by se zvýšila přesnost, ale je to způsobeno tím, že rozdíl dvou sousedních čísel má tolik desetinných míst, kolik činí hodnota *m*. Celkový počet rozlišitelných hodnot se pochopitelně nemění, takže do binárního tvaru lze přesně převést pouze omezený počet čísel reálných (2^n).

Vzniká tak celá řada možností, jak interpretovat čísla v mikropočítači. Prakticky se využívají pouze dvě hodnoty *m*.

Pro m = 0 získáváme již známá čísla INTEGER bez znaménka.

Pro m = n dostaneme tzv. čísla FRACTION, pro která platí:

$$x = \sum_{i=0}^{n-1} a_i \cdot 2^{i-n}$$

nejmenší číslo 0

největší číslo $1-2^{-n}$

rozdíl dvou sousedních čísel 2^{-n}

počet rozlišitelných hodnot 2^n

Tato čísla jsou tedy menší než jedna a používají se tam, kde např. z časových důvodů nestihne mikroprocesor zpracovat čísla v pohyblivé řádové čárce tj. například v některých řídicích aplikacích zvláště s jednoduššími procesory. Práce s čísly FRACTION vyhovuje pro výpočty připravené v poměrných hodnotách. Dále je poměrně užitečné, že tato čísla jsou odolná proti přetečení při násobení (oba činitelé jsou menší než jedna).

9.3.2 *Vyjádření znaménka*

Doposud jsme hovořili pouze o vyjádření kladných čísel s pevnou řádovou čárkou. Potřebujeme-li zobrazovat čísla se znaménkem je zřejmé, že počet bitů rezervovaných na rozsah čísel bude nutno zvětšit o jeden bit, který ponese informaci o znaménku. Čtyři nejpoužívanější vyjádření čísel v pevné řádové čárce, která si nyní popíšeme, jsou graficky zobrazena na obr. 9.13.

❑ Vyjádření ± absolutní hodnota

Při tomto vyjádření představuje nejvyšší bit znaménko (obvykle 0...+, 1...-) a zbývajících b-1 bitů určuje absolutní hodnotu zobrazovaného čísla. Vyjadřujeme-li celá čísla a máme k dispozici b bitů, pak můžeme zobrazovat kladná i záporná čísla v intervalu <0, 2^{b-1}-1>. Ve vyjádření existují dvě nuly, z nichž použití záporné nuly 1000...000 se většinou zakazuje.

Tento kód se často využívá u A/D převodníků, které často pracují v okolí nulového napětí, neboť se snadno uskuteční přechod z malého kladného napětí na malé záporné a opačně. V tomto kódu se nemění větší počet údajových bitů při přechodu nulovou úrovní.

❑ Vyjádření záporných čísel jednotkovým doplňkem

U tohoto vyjádření jsou kladná čísla vyjádřena stejně jako ve tvaru ± absolutní hodnota. Záporná čísla jsou v jednotkovém doplňku, který označujeme ${}^1x$ a vypočteme z následujícího vztahu:

$$ {}^1x = 2^b - 1 - x = 2^b - 1 - \sum_{i=0}^{b-2} a_i \cdot 2^i $$

Hodnota 2^b-1 představuje číslo, které má na všech n bitech samé jedničky. Odečteme-li od této hodnoty b-1 bitové číslo *X*, získáme číslo mající oproti číslu *X* všechny bity negované. Vyjadřujeme-li celá čísla a máme k dispozici b bitů, můžeme zobrazovat čísla kladná i záporná v intervalu <0, 2^{b-1}-1>. Nejvyšší bit značí znaménko (0...+, 1...-). Ve vyjádření opět existuje kladná (000...0) i záporná (111...1) nula.

Jednotkový doplněk se používá (ale méně než dvojkový) v číslicových zařízeních, které uskutečňují aritmetické operace. Pomocí jednotkového doplňku se uskutečňuje odečítání sčítáním tak, že se vytvoří jednotkový doplněk menšitele, který se přičte k menšenci a k výsledku se přičte jednička v nejnižším dvojkovém řádě. Toto nazýváme kruhovým přenosem

❑ Vyjádření záporných čísel dvojkovým doplňkem

Záporná čísla ve dvojkovém doplňku, který označujeme ${}^2X$, vypočteme z následujícího vztahu

$$ {}^2X = 2^b - X = 1 + {}^1X = 2^b - \sum_{i=0}^{b-2} a_i \cdot 2^i $$

Jak vyplývá z předešlého textu, získáme dvojkový doplněk tak, že všechny bity čísla X znegujeme (jednotkový doplněk) a pak k němu přičteme jedničku na nejnižší pozici. Vyjadřujeme-li celá čísla a máme k dispozici b bitů, můžeme zobrazovat kladná čísla v intervalu <0,2^{b-1}-1> a záporná čísla v intervalu <-1,-2^{b-1}>. Nejvyšší bit opět značí znaménko (0…+, 1…-). Ve vyjádření již existuje jen jedna nula (000…0). Budeme-li vytvářet dvojkový doplněk čísla se zlomkovou částí, potom jedničku

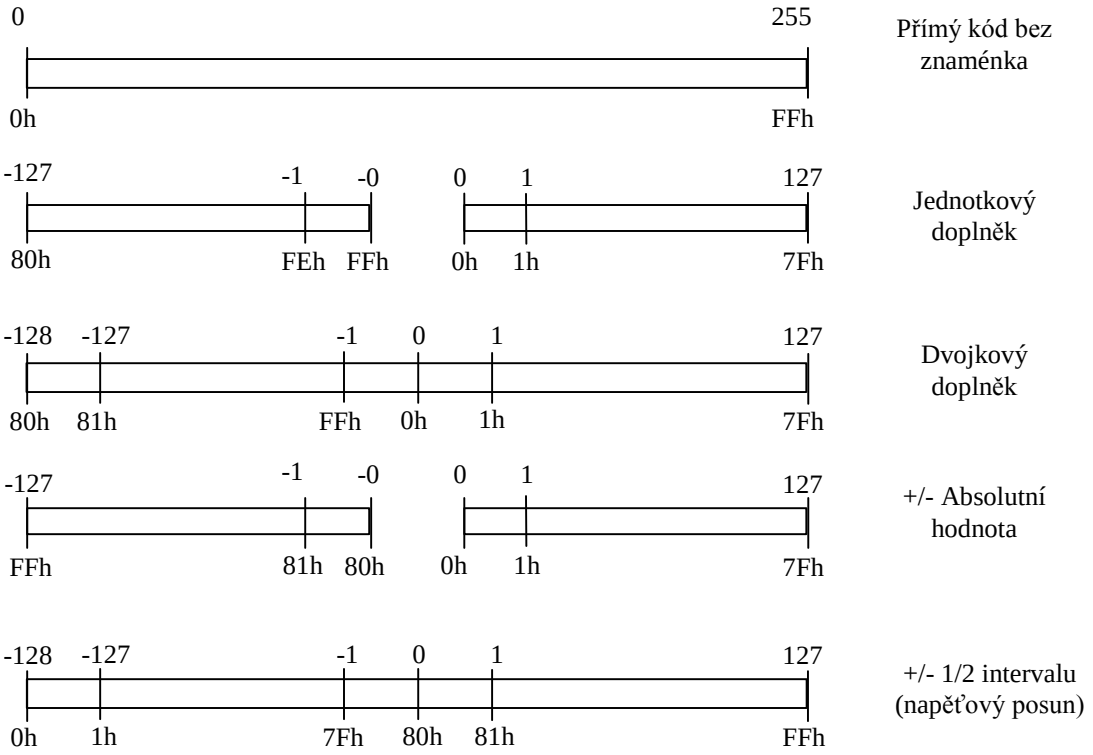
přičítáme k bitu s nejmenší váhou 2^{-n}. Jedná se o nejčastěji používaný způsob zobrazení čísel v technických realizacích.

❑ Vyjádření 1/2 intervalu

Číslo v tomto vyjádření dostaneme ze vztahu

$$^{1/2}X = 2^{b-1} + X = 2^{b-1} + \sum_{i=0}^{b-2} a_i \cdot 2^i$$

Bit s nejvyšší váhou opět značí znaménko s tím ale, že je 0…- a 1…+. Stejně jako ve dvojkovém vyjádření má toto vyjádření jen jednu nulu. U tohoto kódu je velmi jednoduchý převod na dvojkový doplněk. Nevýhodou je změna ve všech dvojkových řádech při přechodu z nuly na nejmenší záporné číslo. U A/D převodníků bývá změna analogového napětí v okolí nuly poměrně dosti častá.



Obr. 9.13 Grafické znázornění rozsahu kladných a záporných čísel v základních formátech pro 8 bitů.

Shrnutí pojmů 9

Klíčová slova:
A/D převod, D/A převod

Otázky 9

1. Popište způsoby připojení mikropočítače k analogovému prostředí
2. Popište principy D/A převodu
3. Popište principy A/D převodů
4. Jakým způsobem je možné vyjádřit znaménko v mikropočítačovém systému

Úlohy k řešení

96. Rychlost převodu D/A převodníků ovlivňuje

a hodnota vstupního napětí
b rychlost přeběhu použitého operačního zesilovače a velikost změny vstupního napětí
c velikost taktovacího kmitočtu
d šířka převáděného slova

97. Nestabilita referenčního napětí u D/A převodníku ovlivňuje:

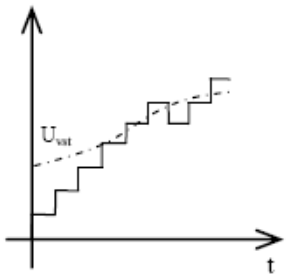
a přesnost, rozlišovací schopnost
b přesnost, rozlišovací schopnost, linearitu
c přesnost
d rozlišovací schopnost

98. Nejrychlejší typ A/D převodníku je:

a kompenzační
b integrační
c paralelní
d integrační s dvojí integrací

99. Na obrázku je naznačen princip kompenzačního A/D převodníku s:

a integračním způsobem řízení
b aproximačním způsobem řízení
c sledovacím způsobem řízení
d jiným způsobem řízení



100. Konstantní dobu převodu mají převodníky:

a integrační
b paralelní
c kompenzační aproximační
d kompenzační sledovací

101. Rychlost převodu u A/D převodníků s postupnou aproximací závisí

a frekvenci použité u aproximačního registru a šířce výstupního slova
b šířce výstupního slova a velikosti vstupního převáděného napětí
c pouze na velikosti vstupního převáděného napětí
d na velikosti referenčního napětí

102. DA převodník používají ke své funkci AD převodníky

a paralelní
b kompenzační
c integrační
d integrační s dvojí integrací

103. Odporová síť R/2R se používá:

a v integračních AD převodnících
b v paralelních AD převodnících
c ke kompenzaci AD převodníků
d v konstrukci DA převodníku

104. Rozlišovací schopnost osmibitového AD převodníku se vstupním rozsahem 10V je přibližně:

a 0,4V
b 40mV
c 1,25V
d 155mV

105. Sample/Hold obvod se používá:

a k linearizaci AD převodníku
b ke zvýšení rozlišovací schopnosti
c ke snížení kvantizační chyby
d jako vstupní zachytávací jednotka některých typů AD převodníků

106. Nejlepší filtrační účinek pro superponovaná rušivá napětí mají:

a Samle/ Hold obvody
b kompenzační AD převodníky
c integrační AD převodníky
d AD převodníky s diferenciálními vstupy

107. Převodník s postupnou aproximací vyžaduje ke své funkci:

a D/A převodník
b čítač
c obvod LATCH
d integrátor
e sčítačku

108. Referenčního napětí využívají ke své činnosti:

a DA převodníky
b integrační AD převodníky
c paralelní AD převodníky
d inkrementační AD převodník

109. Komparátor vyžaduje ke své funkci

a integrační A/D převodník
b kompenzační A/D převodník
c paralelní A/D převodník
d všechny výše uvedené

110. Integrační A/D převodník je oproti kompenzačnímu A/D převodníku

a rychlejší
b přesnější
c mají přibližně stejnou rychlost, ale integrační je přesnější
d mají přibližně stejnou přesnost, ale integrační je rychlejší

111. Osmibitový paralelní A/D převodník obsahuje mimo jiné

a 4 komparátory
b 8 komparátorů
c 16 komparátorů
d 256 komparátorů

112. Kompenzační A/D převodník pracující na principu metody vzestupného čítání je oproti kompenzačnímu A/D převodníku pracující na principu metody sledovací

a rychlejší
b pomalejší
c zhruba stejně rychlý
d rychlejší s výjimkou startu

113. Obvody ICL7106, ICL7107 obsahují

a A/D převodník s dvojí integrací
b paralelní A/D
c D/A převodník
d aproximační registr

114. Dvojí vyjádření nuly se nevyskytuje u vyjádření znaménka:

a S kódováním ± absolutní hodnota
b S kódováním dvojkovým doplňkem
c S kódováním jednotkovým doplňkem a ± 1/2 intervalu (napěťový posun)
d S kódováním dvojkovým doplňkem a ± 1/2 intervalu (napěťový posun)

115. Dvojí vyjádření nuly využívá:

a jednotkový doplněk
b dvojkový doplněk
c napěťový posun
d ±absolutní hodnota

10. MIKROPROCESORY, MIKROPOČÍTAČE A MIKROPOČÍTAČOVÉ SYSTÉMY

Čas ke studiu: 1,5 hodiny

Cíl Po prostudování tohoto odstavce budete umět

- způsoby rozdělení mikroprocesorů
- vysvětlit rozdíl mezi architekturou Von Neumann a Harvardskou
- vysvětlit rozdíl mezi architekturou CISC a RISC

Výklad

Historie mikroprocesorů začíná na začátku sedmdesátých let, kdy se vývojový pracovník firmy Intel rozhodl, že jednu se zadaných úloh nebude řešit několika specializovanými obvody, jak se předpokládalo, ale že vytvoří obvod programovatelný tak, aby mohl s jeho pomocí řešit úloh několik. Tak byl položen základ k prvnímu čtyřbitovému mikroprocesoru I 4004. Čtyřbitové mikroprocesory vyráběla celá řada firem zejména pro použití v kalkulačkách a hrách a další jednoduchých aplikacích. Nejvýznamnějším výrobcem v této oblasti se stala firma Texas Instruments.
Avšak prvním skutečně masově užívaným mikroprocesorem se stal mikroprocesor I 8080. Ten představoval světově užívaný standard v oblasti osmibitových mikropočítačů. V době svého vzniku byl zhruba desetkrát výkonnější než jiné typy. Zajímavé je, že ve stejném roce (1974) byl uveden na trh nejen konkurenční osmibitový mikroprocesor firmy Motorola – 6800, ale i první šestnáctibitový mikroprocesor na jednom čipu. Z mikroprocesoru I 8080 vychází i mikroprocesor Z 80 firmy Zilog, vytvořený v roce 1974 několika pracovníky, kteří firmu Intel opustili. O rok později je představen zlepšený I 8085. Jmenované mikroprocesory prakticky obsadily trh a v některých případech jsou využívány dodnes.
Další vývojový krok představují šestnáctibitové mikroprocesory, které se staly základem komerčně úspěšných osobních počítačů. Také zde byla úspěšná firma Intel s typem I 8086. Ta dále pokračuje s dalšími typy ´286, ´386, ´486, Pentium atd. Společně s ní zaujímá nejvýznamnější místo na trhu firma Motorola, jejíž mikroprocesory jsou například základem osobních počítačů Apple – MacIntosh.
Nutno připomenout, že rozvoj mikroprocesorů musel být doprovázen rozvojem i dalších obvodů, zejména pamětí a vývojových prostředků.
Již s vývojem šestnáctibitových mikroprocesorů bylo jasné, že tyto přesáhnou své původní určení jednoduchého automatizačního prostředku. Přitom bylo výrobcům jasné, že opustit tuto oblast by bylo neuvážené, protože představuje ohromný trh. Pokladny v obchodech, počítadla benzínových pump, domácí elektronika, číslicově řízené obráběcí stroje a další, to jsou jen namátkově vybrané příklady uplatnění. Přitom užití výkonných mikroprocesorů by zde nebylo účelné. Proto vznikly monolitické mikropočítače tzv. jednočipové mikropočítače, jež mají většinu podpůrných obvodů a bloků mikropočítače přímo v jedné součástce, na jednom čipu. Jsou nabízeny opět celou řadou firem. Když nahlédneme k firmě Intel jsou to typy 8048, 8051, 8096 atd.

10.1 Rozdělení procesorů

Procesory se mohou dělit podle různých hledisek do různých skupin. Tyto skupiny se samozřejmě dají dále rozdělit do různých podskupin podle dalších hledisek, jako je například architektura

mikroprocesoru, koncepce apod. Některé skupiny se vzájemně překrývají jako jsou RISC a CISC architektury.

10.1.1 První způsob dělení:

a) CPU (Central Processor Unit)

Mikroprocesory, jak je známe asi všichni, jsou srdcem našich osobních počítačů, ať PC nebo MACů a také jakýchkoliv jiných stolních počítačů. Jejich hlavní vlastnosti jsou:

- otevřenost procesoru, který neobsahuje žádná periferní zařízení přímo v sobě (na jednom čipu) a všechny periferie musí být připojeny externě. Např. paměť, vstupně-výstupní zařízení (sériové porty, paralelní porty, USB, zvuková karta.
- vysoká cena; oproti ostatním typům mikroprocesorů se cena řídí především trhem
- vysoký výkon; v podstatě jsou CPU svým výkonem daleko před jednočipovými mikropočítači, dnes velmi často i před DSP.
- vysoká spotřeba a ztrátový výkon, tyto mikroprocesory se velmi zahřívají a ve většině případech potřebují velmi účinné chlazení. Má to souvislost především s výpočetním výkonem mikroprocesoru a jeho vysokým odběrem (až jednotky ampér).
- velké rozměry pouzdra

b) MCU (Micro Controller Unit)

Spíš nesou označení jako jednočipové mikropočítače, i když si každý výrobce své „brouky“ nazývá podle svého a tak občas nastává trochu zmatek. Tento typ se používá téměř kdekoliv; počínaje osobním počítačem (klávesnice), řídicí jednotka v automobilech, v některých měřicích přístrojích apod. Hlavní výhody:

- nízká cena; nejlevnější mikrořadiče se dají dnes pořídit i do 50 kč, ty nejdražší cca kolem 1 000 Kč.
- nízká spotřeba v klidovém režimu se může pohybovat v jednotkách μA, za chodu kolem stovek μA.
- malé rozměry mikropočítače
- menší možnost rozšíření

c) DSP

Určitým kompromisem mezi klasickým CPU a mikropočítačem je signálový procesor. Jak jeho název napovídá, je určen především ke zpracování signálů. Stručně řečeno slouží signálový procesor k tomu, aby data, která do něj vstoupí, zpracoval a co nejrychleji je předal na výstup. Jeho výhody:

- Vysoká rychlost zpracování číslicových dat.
- Velmi rychlé matematické operace jak v plovoucí, tak v pevné desetinné čárce. Současná hodnota, kterou dosahují kvalitní DSP, se pohybuje kolem 60 MFLOPS (60 miliónů operací v plovoucí desetinné čárce za sekundu).
- Schopnost zpracovávat velké objemy dat.
- Signálový procesor je již speciální variantou, kde ostatní hlediska jako spotřeba, cena atd. závisí na aplikacích, ve které je procesor použit. Pomocí DSP se dosahuje ve vašich SoundBlasterech a podobných zvukových kartách efektů se zvukem. Stejně tak může DSP zajišťovat kompresi zvuku, obrazu apod. Dále se používají u HDD u čtecích hlav jako D/A převodník. Signálové procesory jsou velmi náročné na přístupovou rychlost pamětí, které požívají. Je to právě proto, že se pracuje s velkým objemem dat, na něž se navíc aplikují funkce jako je rychlá Fourierova transformace (např. spektrální analyzátor)

10.1.2 Druhý způsob dělení

Mezi návrháři počítačů byl dlouho všeobecně rozšířen názor, že čím bohatší instrukční sada procesoru, tím výkonnější počítač. Výrobci procesorů se pak doslova předháněli v tom, kolik složitých instrukcí přidají do svých procesorů. Doufali, že tyto složité instrukce budou programátory bezezbytku využívány a že se postupně nějaký jazyk vysoké úrovně stane přímo strojovým kódem. Předpoklad výrobců mohl snad být oprávněný v době, kdy se programy psaly pouze v assembleru a byly zároveň šity na míru konkrétním procesorům.

Později však rozsáhlé testy velkého vzorku typických programů prokázaly, že ačkoli návrhář často vynaloží značné úsilí při řešení problému, jak implementovat universálně přijatelnou instrukci, pouze několik málo tvůrců překladačů ji dokáže využít. V praxi je totiž nakonec využívána jen jistá podmnožina instrukčního souboru a adresních režimů procesoru. Nejvíce je využívána skupina jednoduchých instrukcí, zatímco ostatní relativně složitější instrukce se používají jen málo. Procesor, obsahující veliké množství složitých instrukcí a způsobů adresování, je více komplikovaný a v důsledku toho i nevýkonný. Provedení i těch nejzákladnějších a nejjednodušších instrukcí trvá u takového procesoru déle, než by mohlo trvat jednoduššímu procesoru, který nemá tak obsažnou sadu instrukcí. Vedle toho mají složitější procesory komplikovanější vývoj. Uživatel také draze platí za nevyužité technické prostředky.

Na základě výše uvedených rozporů je možno rozdělit teorie struktury procesorů do dvou základních směrů. První z nich, původní teorie, je známa pod názvem CISC (Complex Instruction Set Computer) neboli počítač s kompletním souborem instrukcí. Snahou výrobců těchto procesorů je tedy vybavovat procesory co nejširší sadou instrukcí. Pravým protipólem CISC je skupina procesorů známá jako RISC (Reduced Instruction Set Computer) neboli počítač s redukovaným souborem instrukcí. Jejich výrobci se naopak snaží maximálně omezit instrukční sadu procesorů. Klíčovou je myšlenka o přenesení některých výpočetních funkcí procesoru z technických prostředků na programové. Jednodušší technické prostředky tak umožní vykonávat dané funkce realizované programovými prostředky rychleji než složité technické prostředky. Instrukční sada těchto procesorů je tedy redukována pouze na jednoduché a rychle prováděné instrukce, na ty, které bývají nejvíce využívány. Zároveň jsou tyto instrukce implementovány tak, aby mohly být prováděny, co nejrychleji.

Koncepce RISC se však netýká jen pouhého redukování instrukčního souboru. Je to spíše celá soustava názorů a představ o tom, jak by měl být počítač realizován a jak by měl fungovat.

❑ Definice RISC

Architektura RISC realizuje na první pohled paradoxní požadavek: větší výpočetní výkon za nižší cenu a s jednodušším instrukčním souborem. Procesory RISC fungují na třech základních principech:

- Regularita a jednoduchost souboru instrukcí, které dovolují opakované použití jednoduchých bloků obvodů pro provádění většiny instrukcí.
- V každém strojovém cyklu pokud možno končí provedení jedné instrukce. Žádná instrukce nemůže počítat adresy svých vlastních operandů a realizuje pouze operace nad registry Load a Store.
- Instrukce mají pevnou délku a neměnný formát

Charakteristické znaky:

- minimální instrukční soubor (80 - 150 instrukcí)
- jednoduché způsoby adresování
- jeden nebo málo formátů instrukcí
- řízení jednoduchou pevnou logikou
- jednocyklové strojové operace (většina instrukcí se provádí během jednoho taktu)
- styk s pamětí výhradně prostřednictvím operací Load a Store (strojové instrukce pracující přednostně s lokálními registry procesoru)
- datové operace pouze nad registry
- rychlé paměti na uložení programů a operandů (paměť cache běžící na stejné frekvenci jako vlastní procesor, velký počet programově přístupných registrů pro operandy)
- zřetězená realizace instrukcí
- optimalizující kompilátor - společný vývoj architektury procesoru a optimalizujícího kompilátoru.

❑ Definice CISC

Systémy s rozsáhlým komplexním souborem instrukcí jsou založeny na architektuře CISC (Complex Instruction Set Computer). Jsou to např. procesory VAX firmy Digital Equipment, série

mikroprocesorů 80X86 Intel a 680X0 firmy Motorola. Pro tyto procesory je typická implementace architektury pomocí mikroprogramování. Výsledkem je velký počet specializovaných typů instrukcí, z časového pohledu mohou trvat až 300 strojových cyklů. Mikroprogramování poskytuje možnost nabízet spektrum strojů se stejnou architekturou, ale přesto rozdílnou hardwarovou realizací. S rostoucím objemem sady instrukcí však byla pro překladače překládající programy do strojového kódu využita celá škála speciálních instrukcí. Z toho vyplývá, že komplexní instrukce jsou používány jen zřídka. Při zpracování programu z vyššího programovacího jazyka se velká část času spotřebuje na čtení informace z pracovní paměti a naopak.

Charakteristické znaky:

- velké množství instrukcí (100-250)
- některé instrukce vykonávají speciální funkce
- velké množství adresovacích módů
- instrukce o proměnné délce
- instrukce pracují s operandy v paměti
- mnoho a složité způsoby adresování

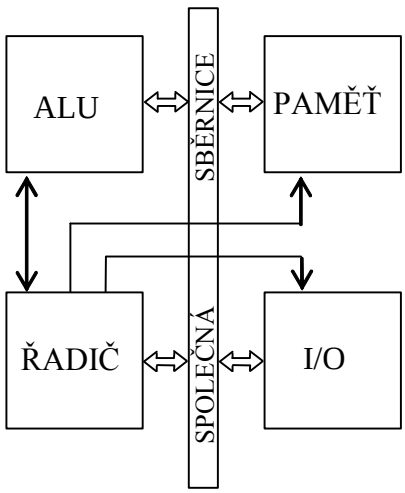
10.1.3 Dělení podle architektury

Pojem architektura mikropočítače je chápán jako obor studující strukturu, organizaci, realizaci a funkci počítačů. Struktura popisuje propojení jednotlivých funkčních bloků, organizace řeší dynamické interakce těchto bloků a řízení styku mezi nimi, realizace se zabývá návrhem vnitřních obvodů daného funkčního bloku a funkce uvádí chování počítače jako celku navenek.

Základní architektonické koncepce byly postupně převzaty i do návrhu mikroprocesorových obvodů. Potřeba různorodosti zpracování vyvolala nejprve modifikace a pak i nové myšlenky v uspořádání jednotlivých složek architektury technického zařízení. V mikroprocesorové technice rozeznáváme dva základní druhy architektur:

❑ Architektura Von Neumann

Stala se základem většiny mikroprocesorů, další architektury se pak lišily v jednotlivých částech. Základními body jsou:



Obr. 10.1 Architektura von Neuman

- Mikroprocesor se skládá ze 4 funkčních bloků: paměti, řadiče, aritmeticko-logické jednotky (ALU) a vstupních a výstupních jednotek.

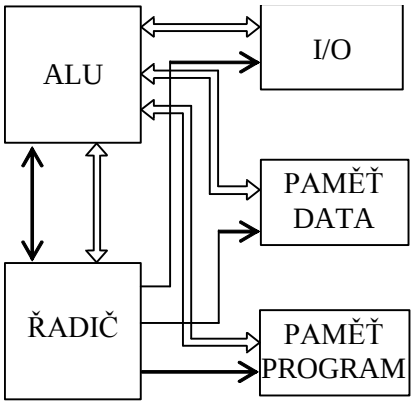
- Struktura zařízení je nezávislá na typu řešení úlohy, činnost mikroprocesoru je řízena obsahem paměti
- Instrukce a operandy jsou uloženy v téže paměti.
- Paměť je rozdělena do buněk stejné velikosti, jejichž pořadová čísla se používají jako adresy
- Program je tvořen posloupností elementárních příkazů (instrukcí), v nichž zpravidla není obsažena hodnota operandu (uvádí se pouze jeho adresa), takže program se při změně dat nemění. Instrukce se provádějí jednotlivě v pořadí, v němž jsou zapsány do paměti.
- Změna pořadí provádění instrukcí se vyvolá instrukcí podmíněného nebo nepodmíněného skoku.
- Pro reprezentaci instrukcí a čísel (operandů, výsledků, adres) se používá dvojkové signály a dvojková číselná soustava.

Výhody:
- plynule se dá měnit velikost prostoru pro data a program v paměti

Nevýhody:
- všechny operace procházejí přes společnou sběrnici, což zpomaluje proces
- možnost přepisu programu daty a naopak

❑ Harvardská architektura

Vychází z architektury von Neuman. Jejím hlavním znakem je oddělení paměti dat od paměti programu.



Obr. 10.2 Harvardská architektura

Výhody:
- daty nelze přepsat program
- každá paměť je připojena k ALU přes vlastní sběrnici
- umožňuje paralelní přenos dat a instrukcí ⇒ větší rychlost

Nevýhody:
- při návrhu nutno znát předem, kolik bude potřeba prostoru pro data a program (mapování paměti)

Mikroprocesory by se ještě dále mohly z hlediska architektury dělit na dvě skupiny podle možnosti připojení periferních obvodů. V kapitole 4 byly popsány dva způsoby styku – klasické – tj. mapování do I/O prostoru nebo neklasické – mapování do paměťového prostoru. První způsob lze využít pouze u mikroprocesorů, které jsou vybaveny I/O adresovým prostorem a odpovídajícími instrukcemi pro adresování tohoto prostoru (většinou OUT a IN).

Poznámka:

RISC nebo CISC?

Nejprve – spíše pro pořádek – si povězme, co vůbec znamenají zkratky RISC a CISC. RISC = Reduced Instruction Set Computing = výpočet s redukovaným počtem instrukcí, kdežto CISC = Complex Instruction Set Computing = výpočet s úplným počtem instrukcí. Větev vývoje procesorů, se začala dělit někdy po dvanácti letech od uvedení prvních čipů. Zatímco procesory CISC pracují s velkým množstvím instrukcí potřebných pro zpracování nejrůznějších úloh, procesory RISC jsou vybaveny pouze základní sadou nejčastěji používaných instrukcí. V případě, že je třeba, aby RISCový procesor provedl složitější operaci, musí ji sestavit z instrukcí, kterými je vybaven – tzv. instrukce základní sady. Má – li však být taková instrukce složena, je třeba dát k tomu nějaký podnět. Ten musí přijít "zvenku" tedy od operačního systému nebo od aplikace napsané přímo pro daný procesor. Procesory RISC jsou navrženy tak, že se tyto elementární instrukce zpracovávají velmi rychle. Nárůst výkonu, kterého je dosaženo zrychlením nejpoužívanějších instrukcí, pak značně převyšuje časové ztráty vzniklé skládáním složitějších instrukcí. Z hlediska výhodnosti a nevýhodnosti jednotlivých architektur nelze opomenout ani to, že procesory RISC potřebují při zachování stejného výkonu mnohem méně tranzistorů integrovaných na jednom křemíku. Konkrétně procesoru CISC s 3,1milionem tranzistorů odpovídá RISC s 1,2milionem tranzistorů. Z této skutečnosti plyne, že čipy CISC jsou větší, nákladnější na výrobu a náročnější na spotřebu elektrické energie (proto se třeba v malých palmtopech používají výhradně procesory s architekturou RISC), která se mění v teplo a tím vznikají další problémy. K rozšíření CISC procesorů – v šedesátých a sedmdesátých letech - vedla zejména jejich nenáročnost na kapacitu paměti (paměťové moduly byly v té době drahé a měly omezenou kapacitu, já osobně si pamatuji dobu, kdy 1MB paměti RAM stál 1000,-). Konstrukce mnoha procesorů vycházela v té době z požadavku minimalizace paměťových nároků programů. Jediným východiskem vedoucím k omezení těchto nároků bylo maximální zjednodušení aplikací (softwaru) za cenu zvýšení složitosti procesoru. Významné změny kapacity, dostupnosti a ceny pamětí v osmdesátých a zejména v devadesátých letech však přinutily vývojáře procesorů k přehodnoceni jejich koncepcí. Architektura RISC má tedy sice větší požadavky na paměť a vyžaduje složitější kompilátory, ale to už v současné době není rozhodující. Myslím, že prvotní otázka "RISC nebo CISC?" by měla znít spíše "Kdy definitivně RISC?" Už dnes se v Pentiích nepoužívá čistá architektura CISC, ale využívá se jakéhosi "vývojového" mezistupně k RISC. Typickými zástupci RISCových procesorů jsou klasické serverové procesory – Alpha na 575MHz se 4MB paměti cache, dále UltraSPARC- II na 400MHz také se 4MB cache a SGI MIPS R10000 nebo R12000 na 300MHz s 8MB cache. Tyto procesory se využívají nejčastěji v paralelním zapojení pro dosažení maximálního výkonu. Takto lze vedle sebe zapojit např. 128, ale i více procesorů. Úplně jinou kapitolou procesorů s architekturou RISC jsou palmtopy, herní konzole (mezi nejznámější patří SONY Playstation), satelitní přijímače apod. Zde trh s procesory plně ovládla firma MIPS (dceřinná společnost SGI). Důvody pro použití architektury RISC jsou myslím jasné z předchozích řádků: menší požadavky na elektrickou energii, menší rozměry, nižší tvorba tepla, nenáročná výroba (z důvodu nižšího počtu základních instrukcí) a z toho plynoucí nižší cena.

Shrnutí pojmů 10

Klíčová slova:
Architektura Von Neumann, Harvardská architektura

Otázky 10

1. Popište způsoby dělení mikroprocesorů
2. Popište rozdíly, výhody a nevýhody architektury Von Neumann a Harvardské.
3. Vysvětlete princip architektury CISC a RISC

Úlohy k řešení 10

116. Architektura Von Neumann je charakteristická:

a oddělením paměťového prostoru pro program a data
b společnou datovou a adresovou sběrnici
c nepřítomností řadiče instrukcí
d paralelním přenosem dat a instrukcí

117. Harvardská architektura je charakteristická těmito vlastnostmi :

a společná datová a programová paměť, každá paměť je připojená k ALU přes společnou sběrnici, neumožňuje paralelní přenos dat a instrukcí
b společná datová a programová paměť, každá paměť je připojená k ALU přes vlastní sběrnici, neumožňuje paralelní přenos dat a instrukcí
c oddělená datová a programová paměť, každá paměť je připojená k ALU přes vlastní sběrnici, neumožňuje paralelní přenos dat a instrukcí
d oddělená datová a programová paměť, každá paměť je připojená k ALU přes vlastní sběrnici, umožňuje paralelní přenos dat a instrukcí

118. Vyberte správná tvrzení:

a 8-mi bitový mikropočítač může mít 32-bitovou adresovou sběrnici
b všechny procesory stejné architektury mají stejné instrukce
c Harvardská architektura využivá zřetězené zpracování instrukcí (pipeline)
d procesory dnešních PC využívají Von Neumanovu architekturu

Klíč k řešení

1. Napájecí napětí obvodů TTL je

   a) 5V +- 5%

2. Je možné zaměnit obvody řady 74LS… s obvody řady 74HCT…

   a) ano, prakticky ve všech případech

3. Je možné připojit na zem výstup hradla TTL?

   a) ano, ale jen jeden člen v obvodu DIL

4. Při připojení záporného napětí na vstup hradla TTL

   c) dojde ke zničení vstupního tranzistoru

5. Logické úrovně CMOS jsou definovány

   a) log0: 30%Ucc, log1: 70%Ucc

6. Přivedením napětí vyššího než 8 V na vstup hradla TTL

   b) dojde k průrazu vstupního tranzistoru

7. Napájecí napětí logických obvodů CMOS 4000 je:

   c) 3 - 15V

8. Pomalé přechody z úrovně logické 1 do logické 0 na vstupu hradla TTL

   b) způsobí zvýšení příkonu hradla

9. Tvarovací obvody se používají především tam, kde hrany logických úrovní dosahují délky větší než:

   b) 1 ms

10. Za dlouhé vedení z hlediska rychlosti hradel při jejich propojování

    c) se považuje vedení delší než 20 cm

11. Při napojení hradla na dlouhé vedení

    a) by neměly být na jeho vstup připojeny žádné další vstupy

    c) je třeba provést impedanční přizpůsobení vedení

12. Napěťové úrovně obvodů TTL pro stavy log.1 a log.0:

    b) jsou pevně stanoveny pro vstup i výstup hradla

13. Napěťové úrovně obvodů CMOS pro stavy log.1 a log.0:

a) jsou odvozeny od napájecího napětí

14. Napěťová úroveň obvodů TTL pro stav log.0 je:

b) pro vstup 0÷0.8V, pro výstup 0÷0.4V

15. Napěťová úroveň obvodů CMOS 4000 pro stav log.1 je:

b) je závislé na napájecím napětí

16. Příkon hradla logických obvodů TTL a CMOS ve statickém režimu:

b) je větší u řady TTL

17. Pojem šumová imunita:

b) je odolnost obvodů vůči rušivým signálům

c) je odvozena z logických úrovní hradel

18. Nepoužité vstupy hradla NAND je třeba připojit:

a) paralelně k použitým vstupům

c) na napětí logické úrovně H

19. Toto zapojení slouží ke:

a) zpoždění náběžné hrany výstupního signálu

20. Jakému typu hradla odpovídá toto zapojení:

b) NAND

21. Nezapojené vstupy u TTL obvodů způsobují časové zpoždění reakce výstupu přibližně:

b) 1 ns na jeden nezapojený vstup

22. Dva nevyužité vstupy 4-vstupého obvodu NAND nebo AND ošetříme tím, že připojíme:

c) oba na úroveň H

23. Aby měl nevyužitý 4-vstupý člen NAND typu TTL nejmenší proudovou spotřebu zapojíme:

a) všechny čtyři vstupy na úroveň H

24. Zapojení na obrázku představuje:

d) příklad zapojení výstupního členu

25. Při návrhu zapojení s logickými obvody je vhodné:

a) umístit co nejblíže obvodu blokovací kondenzátory

26. Schmittův klopný obvod:

d) je vhodný k tvarování signálů

27. Logické obvody s výstupy typu „otevřený kolektor“:

c) potřebují pro log.1 „pull up“ rezistory

28. Při logické úrovní L na vstupu hradla TTL:

a) teče proud ven z hradla

29. Výstupní odpor hradla TTL je:

b) odlišný při logické úrovni H od L

30. Základní část mikropočítače – mikroprocesor obsahuje:

b aritmeticko-logickou jednotku (ALU)

c čítač instrukcí (PC)

31. Mikropočítač je

c programovatelná jednotka sestavena z mikroprocesoru a dalších nutných podpůrných obvodů

32. Programový čítač

b je modifikován pouze inkrementací samotným mikroprocesorem

d je modifikován pouze mikroprocesorem při instrukci volání podprogramu

33. Adresová sběrnice mikropočítačového systému

b je jednosměrná

34. Šířka vnitřní datové sběrnice mikroprocesoru

a má vliv na celkovou rychlost procesoru

35. Aritmeticko logická část mikroprocesoru

a zajišťuje aritmetické a logické operace

36. Šířka vnitřní sběrnice mikroprocesoru

b nemusí být stejná jako šířka vnější sběrnice mikroprocesoru

37. Časování mikroprocesoru

a je odvozeno z hodinových signálů oscilátoru

c zajišťuje synchronizaci mikroprocesoru

38. Paměťová část mikroprocesoru je tvořena

c souborem registrů

39. Zásobník

a je tvořen pamětí typu LIFO a využívá se např. pro uchování návratové adresy při přerušení nebo pro uložení kontextu registrů při volání podprogramu

40. Do zásobníku

a můžeme ukládat libovolná data

c procesor ukládá návratovou adresu při přerušení

41. Třístavové budiče sběrnice

c umožňují napojení více obvodů k jedné datové sběrnici

42. Obvody typu LATCH

a se využívají pro zachycení dat např. u multiplexované sběrnice

43. Zapojení na obrázku u datové sběrnice představuje :

c výstup v logické 0

44. Vnitřní datová sběrnice procesoru:

b využívá paralelního přenosu

c je obousměrná

45. Osmibitová adresová sběrnice umožňuje adresovat:

b 256 adres

46. Třístavová sběrnice:

c třetí stav odpovídá vysoké impedanci

47. Pomocí kterého obvodu lze ošetřit multiplexovanou sběrnici:

c obvodem LATCH

48. Posilovač sběrnice 74LS245 se používá k:

c oddělení sběrnic a zvýšení zatížitelnosti

49. Pojem multiplexovaná sběrnice v mikropočítačové technice znamená

a využití společných vývodů procesoru pro datovou a adresovou sběrnici

50. Signál chip select (CS):

b slouží k přechodu do režimu spánku

c většinou bývá aktivní v log.0

51. Neúplné adresování u adresových dekodérů znamená

a přiřazení několika adresám pouze jeden signál

52. Adresování s lineárním přiřazením adres znamená

d přiřazení jednotlivým adresovým bitům jednotlivé signály

53. Úplné adresování u adresových dekodérů znamená:

b přiřazení jedné adrese pouze jeden signál

54. Pomocí kterého obvodu lze ošetřit multiplexovanou sběrnici:

c obvodem LATCH

55. Posilovač sběrnice 74LS245 se používá k:

c oddělení sběrnic a zvýšení zatížitelnosti

56. Jakou paměťovou oblast vybírá adresový dekodér uvedení na obrázku

a. D000-D0FF + D800-D8FF

57. Který způsob styku periferie s mikroprocesorem je výhodnější z hlediska využitelného počtu instrukcí pro práci s periferiemi

b mapovaný do paměťového prostoru

58. Jakou paměťovou oblast vybírá adresový dekodér uvedení na obrázku

b. D000-D1FF

59. Jakou paměťovou oblast vybírá adresový dekodér uvedení na obrázku

b. D0-D1

60. Řadič přerušení

c. slouží ke generování vektorů přerušení, jejich priority obsluhy, maskování atd.

61. Mapování paměti je

c rozdělení paměťového prostoru na paměťovou a nepaměťovou část

62. Maskování přerušení:

b umožňuje ignorovat žádosti o přerušení

63. Asynchronní sériový přenos standartu UART probíhá přenosem:

b start bitu, datových bitu, parity a stop bitu

64. Rozhraní SPI slouží k:

d sériové komunikaci mezi integrovanými obvody

65. Signál MOSI v systémech využívajících sběrnici SPI u obvodů typu Master slouží k:

b k odesílání dat k obvodům typu Slave

66. Obvody v systémech využívajících sběrnicí SPI jsou propojeny :

c datovými vodiči, 1 hodinovým vodičem a vodičem pro výběr integrovaných obvodů

67. Sběrnice SPI je typu:

d singlemaster

68. V systémech se sběrnicí I2C jsou jednotlivé stanice připojeny:

a 1 datovým vodičem (SDA) a 1 hodinovým vodičem (SCL)

69. V systémech se sběrnicí I2C jsou v klidovém stavu (volná sběrnice) vodiče:

d datové i hodinové v úrovni H

70. Každá stanice připojená na sběrnici I2C má přidělenou vlastní adresu o délce:

c 7 nebo 10 bitů

71. Maximální přípustná frekvence hodinového signálu SCL u rozhraní I2C je:

d 100 kHz nebo 400 kHz

72. Rozhraní I2C má ve srovnání s rozhraním RS-232:

d proměnnou rychlost komunikace, současně může mluvit 1 zařízení, adresace v protokolu

73. Přenos pomocí rozhraní RS232 je charakterizován:

a úrovní -3 až -25V odpovídající log.1

74. Přenos pomocí rozhraní RS 232 je charakterizováno

d úrovní +3 až +25V odpovídající logické „0“, na vzdálenost do 15 m, s nesymetrickým přenosem

75. Přenos pomocí rozhraní RS485 je charakterizován:

a symetrickým vedením

c možnosti přenosu na větší vzdálenosti než RS232

d větší odolnosti proti rušení než RS232

76. Obvod USART je představitelem:

c univerzální asynchronní a synchronní přijímač a vysílač

77. Obvod typu USART provádí

c převod paralelního slova na sériové a naopak

78. Obvod MAX232:

c pracuje na principu nábojové pumpy

79. 1 baud odpovídá přenosu:

d 1 bitu za sekundu

80. Rozhraní USB se vyznačuje:

d možností připojení více zařízení, komunikační vzdáleností do 5 m a rychlostí do 480 Mbit/s

81. Rozhraní USB obsahuje:

a napájení +5 V

82. Kód CRC

d kód využívající kontrolní součet

83. Poruchy působící na přenos informace lze podle charakteru kategorizovat na

c impulsní, harmonické, fluktuační

84. Poruchy působící na přenos informace lze podle jejich zdroje kategorizovat na

b přírodního původu, průmyslového původu a vznikající v zařízení

85. Poruchy působící na přenos informace lze podle časového výskytu kategorizovat na

c dočasně působící a trvale působící

86. Paritní bit

a určuje, zda je počet jeniček v přenášeném slově lichý nebo sudý

c slouží k detekci přenosu

87. Použitím sériových pamětí proti paralelním se počet datových, adresních a řídích vývodů :

a sníží

88. Na obrázku je znázorněna realizace paměťové buňky typu:

d RWM

89. Paměť EPROM

d umožňuje mazání dat UV zářením

90. U paměti EPROM

a probíhá zápis mnohem pomaleji než čtení

91. Paměť ROM:

d uchovává data i bez napájecího napětí

92. Obvod 27C256 je

d paměť EPROM o kapacitě 32 kB

93. Paměť s označením 2764 je:

b paměť EPROM s kapacitou 64 kbitů

94. Paměť s označením 61256 s šířkou datového slova 8 bitů má

b 15 adresových vývodů

95. Paměť s označením 2764 s šířkou datového slova 8 bitů má

a 13 adresových vývodů

96. Rychlost převodu D/A převodníků ovlivňuje

b rychlost přeběhu použitého operačního zesilovače a velikost změny vstupního napětí

97. Nestabilita referenčního napětí u D/A převodníku ovlivňuje:

b přesnost, rozlišovací schopnost, linearitu

98. Nejrychlejší typ A/D převodníku je:

c paralelní

99. Na obrázku je naznačen princip kompenzačního A/D převodníků s:

c sledovacím způsobem řízení

100. Konstantní dobu převodu mají převodníky:

b paralelní

c kompenzační aproximační

101. Rychlost převodu u A/D převodníků s postupnou aproximací závisí

a frekvenci použité u aproximačního registru a šířce výstupního slova

102. DA převodník používají ke své funkci AD převodníky

b kompenzační

103. Odporová síť R/2R se používá:

d v konstrukci DA převodníku

104. Rozlišovací schopnost osmibitového AD převodníku se vstupním rozsahem 10V je přibližně:

b 40mV

105. Sample/Hold obvod se používá:

d jako vstupní zachytávací jednotka některých typů AD převodníků

106. Nejlepší filtrační účinek pro superponovaná rušivá napětí mají:

c integrační AD převodníky

107. Převodník s postupnou aproximací vyžaduje ke své funkci:

a D/A převodník

108. Referenčního napětí využívají ke své činnosti:

b integrační AD převodníky

109. Komparátor vyžaduje ke své funkci

d všechny výše uvedené

110. Integrační A/D převodník je oproti kompenzačnímu A/D převodníku

b přesnější

111. Osmibitový paralelní A/D převodník obsahuje mimo jiné

d 256 komparátorů

112. Kompenzační A/D převodník pracující na principu metody vzestupného čítání je oproti kompenzačnímu A/D převodníku pracující na principu metody sledovací

b pomalejší

113. Obvody ICL7106, ICL7107 obsahují

a A/D převodník s dvojí integrací

114. Dvojí vyjádření nuly se nevyskytuje u vyjádření znaménka:

b S kódováním dvojkovým doplňkem

115. Dvojí vyjádření nuly využívá:

a jednotkový doplněk

d +-absolutní hodnota

116. Architektura Von Neumann je charakteristická:

b společnou datovou a adresovou sběrnici

117. Harvardská architektura je charakteristická těmito vlastnostmi :

d oddělená datová a programová paměť, každá paměť je připojená k ALU přes vlastní sběrnici, umožňuje paralelní přenos dat a instrukcí

118. Vyberte správná tvrzení:

a 8-mi bitový mikropočítač může mít 32-bitovou adresovou sběrnici

c Harvardská architektura využivá zřetězené zpracování instrukcí (pipeline)

d procesory dnešních PC využívají Von Neumanovu architekturu