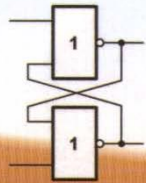
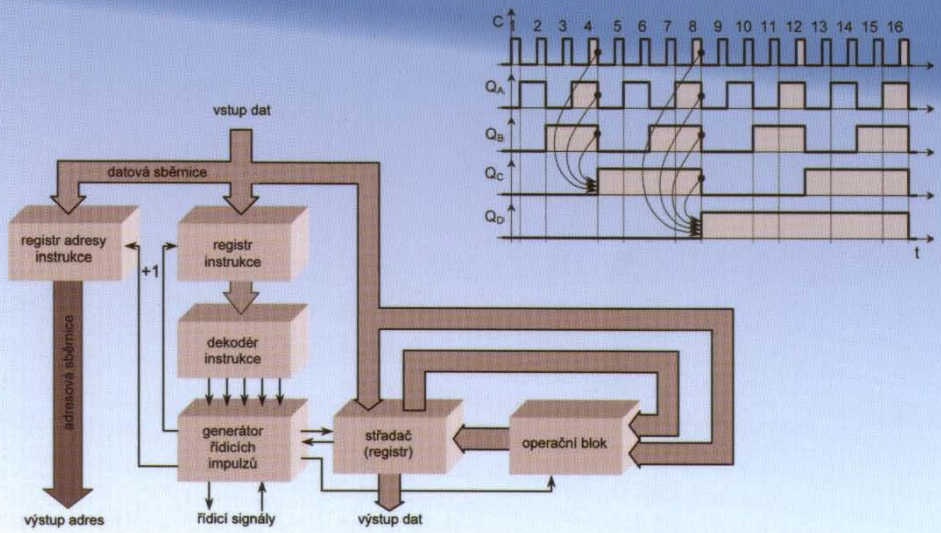


Jan Kesi

ELEKTRONIKA III

číslicová technika



RESET SET

R	S	Q	\bar{Q}
1	0	0	1
0	1	1	0
0	0	1	0
0	0	0	1



Učebnice

základní studijní materiál pro střední školy

Jan KESL

ELEKTRONIKA III

Číslicová technika

UČEBNICE

základní studijní materiál pro střední školy

2. vydání

Praha 2005



O knize

Učebnice ELEKTRONIKA III je věnována v zásadě logickým obvodům od nejjednodušších základních prvků až po složité obvody, ze kterých se skládá počítač.

Vychází z definice informace, jejím přenosem pomocí signálu, číslicovým zpracováním signálu s využitím algebry číslicové logiky a její realizací logickými prvky a obvody. Ve stručnosti knížka vysvětluje zásady minimalizace logických obvodů a dále se věnuje popisu jednotlivých kombinačních a sekvenčních obvodů.

Závěrečná část se zabývá přístupnou formou základními částmi počítače právě z hlediska využití v předchozí části probíraných logických obvodů.

Přílohou učebnice jsou příklady správných odpovědí na kontrolní otázky u každé kapitoly.

Výzva

Autor přislíbil, že další vydání bude aktualizovat a doplňovat podle ohlasů těch, kteří osobně elektroniku a elektrotechniku učí. To proto, aby si učitelé nemuseli pro svou potřebu sestavovat vlastní příručky, mnohdy i z desítky jiných zdrojů, což je pak zbytečně časově zatěžuje.

Připomínky posílejte na adresu naší redakce, která je uvedena na konci knihy.

Jan Kesl

ELEKTRONIKA III

Bez předchozího písemného svolení nakladatelství nesmí být kterákoli část kopírována nebo rozmnožována jakoukoli formou (tisk, fotokopie, mikrofilm nebo jiný postup), zadána do informačního systému nebo přenášena v jiné formě či jinými prostředky.

Autor a nakladatelství nepřijímají záruku za správnost tištěných materiálů. Předkládané informace jsou zveřejněny bez ohledu na případné patenty třetích osob. Nároky na odškodnění na základě změn, chyb nebo vynechání jsou zásadně vyloučeny.

Všechny registrované nebo jiné obchodní známky použité v této knize jsou majetkem jejich vlastníků. Uvedením nejsou zpochybněna z toho vyplývající vlastnická práva.

Veškerá práva vyhrazena.

© Ing. Jan Kesl, Praha 2005

Nakladatelství BEN – technická literatura, Věšínova 5, Praha 10

Jan Kesl: ELEKTRONIKA III

BEN – technická literatura, Praha 2005

2. vydání

ISBN 80-7300-182-9

OBSAH

1 ZPRACOVÁNÍ INFORMACE	5
1.1 Informace a signál	5
1.2 Číslicové signály	5
1.3 Číslicové zpracování analogových signálů	6
1.4 Zobrazení informací	7
1.5 Číselné soustavy a kódy	8
1.6 Dvuhodnotové zobrazení čísel a znaků	12
2 LOGICKÉ FUNKCE A OBVODY	15
2.1 Základní pojmy, význam logických obvodů	15
2.2 Booleova algebra	16
2.3 Základní logické funkce a logické členy	17
2.4 Pravidla Booleovy algebry	23
2.5 Minimalizace logických funkcí	26
2.6 Realizace logických obvodů	32
3 KOMBINAČNÍ LOGICKÉ OBVODY	36
3.1 Binární sčítačka	39
3.2 Generátor parity	42
3.3 Kodéry	43
3.4 Dekodéry	45
3.5 Multiplexery	50
3.6 Demultiplexery	52
3.7 Binární komparátor	52
4 SEKVENČNÍ LOGICKÉ OBVODY	55
4.1 Princip, klopné obvody RS	55
4.2 Klopný obvod typu D	63
4.3 Klopný obvod typu JK	63
4.4 Klopný obvod typu T	66
4.5 Klopný obvod JK dvoufázový	66
4.6 Paměťové registry	67
4.7 Čítače impulzů	71

5	MIKROPOČÍTAČOVÁ TECHNIKA.....	76
5.1	Struktura mikropočítače	76
5.2	Processor	78
5.3	Operační paměť	83
5.4	Druhy a vlastnosti pamětí	86
5.5	Realizace pamětí RAM	87
5.6	Realizace pamětí ROM	91
	ODPOVĚDI NA KONTROLNÍ OTÁZKY	95
	LITERATURA	99

1 ZPRACOVÁNÍ INFORMACE

1.1 Informace a signál

To, co lidé sdělují jiným lidem, nazýváme zcela obecně *informace*.

INFORMACE

Informace je obecný abstraktní pojem, který označuje obsah neboli význam tohoto sdělení – dokumentu nebo slovního spojení. Je to to, co snižuje neurčitost a co naopak zvyšuje pravděpodobnost, že dokážeme přesně předvídat nebo přesně poznat výsledky určité události. Informace jsou údaje, čísla, znaky, povely, instrukce, příkazy, zprávy apod. Informace je jedním ze základních pojmů kybernetiky, vědy o řízení v organismech. Má nehmotný charakter, ale je vždy spojena s nějakým fyzickým pochodem, který ji nese.

SIGNÁL

Nositelem informace je *signál*. Signál je určitá velikost napětí nebo proudu, světelný paprsek, určitá velikost tlaku plynu nebo tekutiny, otvor v papíře, určitá polarizace magnetického dipólu apod. Je to časově proměnná fyzikální veličina, již jsou zobrazeny údaje. Signál umožňuje získání informace, její přenos na dálku, uchování v paměti a zpracování na stroji pro zpracování informací, zejména v počítači, a to i po zániku zdroje informace.

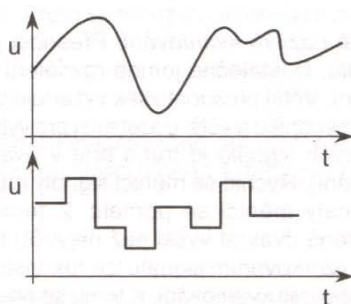
Dnes existují dvě vědy, jejichž předmětem zkoumání jsou informace. Jednou je teorie informace a druhou informatika.

1.2 Číslicové signály

SIGNÁL ANALOGOVÝ

SIGNÁL DIGITÁLNÍ

Na rozdíl od *signálu analogového*, který představuje jev probíhající spojitě, je *signál digitální (číslíkový)* nespojitý, tzn. že se mění skokem pouze v předem stanovených



Obr. 1.1 Signál spojitý a nespojitý

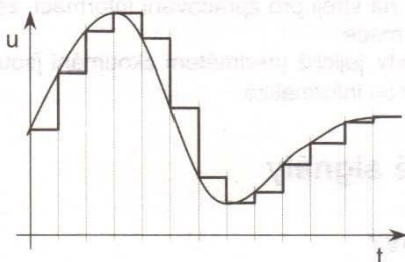
okamžicích. Každou hodnotu lze proto vyjádřit určitým číslem. Číslicový signál je posloupnost takovýchto čísel vzdálených od sebe o pevný časový interval. Názorné vysvětlení rozdílu mezi signálem analogovým a číslicovým je patrné na grafickém záznamu hodnot obou signálů (obr. 1.1).

1.3 Číslicové zpracování analogových signálů

V současné době se ve stále větší míře používají číslicové metody i při měření, přenosu, záznamu a zpracování analogových signálů. K tomu je nutné, aby napěťové signály, které jsou analogové, byly analogově číslicovým převodníkem převedeny na číslicové hodnoty, tj. aby byly digitalizovány.

A-Č PŘEVODNÍKY

Analogově-číslíkové převodníky jsou obvody, které v konstantních časových intervalech generují čísla, která jsou úměrná analogovému signálu. Rozsah amplitud analogového signálu se rozdělí na konečný počet úseků a každému úseku se přiřadí jedno číslo. V okamžiku vzorkování se zjišťuje, kterému úseku odpovídá vstupní napětí. Číslo, které je tomuto úseku přiřazeno, je pak použito jako digitalizovaná hodnota. Posloupnost číslicových hodnot (čísel) získaných v pravidelných časových odstupech je číslicový signál (obr. 1.2).



Obr. 1.2 Digitalizace analogového signálu

KVANTOVÁNÍ

Popsaný postup se také nazývá kvantování. Přesnost převodu závisí na počtu úseků, tj. na šířce intervalů. Dostatečně jemné rozdělení rozsahu hodnot umožní libovolně přesné kvantování. Větší přesnost však vyžaduje také nákladnější provedení analogově číslicového převodníku a větší počet míst pro vyjádření číslicových hodnot.

Při digitalizaci analogových signálů je třeba brát v úvahu i volbu vzorkovacích intervalů (periody vzorkování). Rychle se měnící signály musí být vzorkovány v kratších intervalech než signály měnící se pomalu. Z teorie vyplývá, že kmitočet vzorkování musí být nejméně dvakrát vyšší než nejvyšší kmitočet obsažený v signálu. Z hodnot získaných vzorkováním signálu lze rekonstruovat průběh původního signálu s přihlédnutím k přesnosti kvantování. K tomu se používá číslicově-analogový převodník.

Uvedený postup využívá např. číslicová gramofonová deska (CD). Analogové zvukové signály se digitalizují, v číslicovém tvaru se zpracují a zapíší. Při přehrávání se číslicové údaje opět převádějí na analogový signál, jímž se po zesílení budí reproduktory.

Problémy vznikají, mají-li analogové signály nezvykle velkou dynamiku (změnu amplitudy) nebo mění-li se velmi rychle, takže obsahují velmi vysoké kmitočty. Realizace analogově-číslcového převodníku, který má požadované vlastnosti (požadovanou vstupní dynamiku a přesnost), je velmi nákladná.

Bez ohledu na problémy, o nichž jsme se zmínili, jsou analogové postupy při měření, přenosu, záznamu a zpracování signálů stále více zatlačovány do pozadí v důsledku vývoje polovodičových součástek a s ním souvisejícího poklesu ceny číslicových obvodů.

Výhodou číslicového zobrazení signálů je větší bezpečnost dat a menší citlivost na rušení. Analogové signály podléhají různým (většinou neodstranitelným) rušením. Například náhodné poruchy amplitudy v důsledku šumu nelze rozpoznat nebo korigovat. To vyplývá z toho, že při spojitě probíhajících signálech jsou všechny hodnoty amplitudy možné, a tedy přípustné. Číslicové signály se převádějí do binární (dvojkové) číselné soustavy, to znamená, že číslicový signál získaný z analogového se zobrazí jako posloupnost pouze dvou možných stavů (hodnot), nejčastěji napětíových úrovní. Jednoduchými prostředky lze rozpoznat a korigovat i relativně velké odchylky od těchto napětíových úrovní. Lineární nebo nelineární zkreslení se při číslicovém (dvouhodnotovém) zobrazení nevyskytuje. Dalšího zabezpečení proti rušení se dosahuje použitím bezpečnostních kódů. Tím, že počítače jsou programovatelné, lze řešit mnoho úloh z oblasti řízení technologických procesů mnohem pružněji.

1.4 Zobrazení informací

Máme-li dvě možnosti a dozvíme se, že jedna z nich platí, získáme nejmenší možné množství informace. Pokud bychom totiž měli jen jednu možnost, nemusíme se rozhodovat. Toto nejmenší množství informace, volbu ze dvou možností, nazýváme bitem.

BIT

Bit je zkratka anglických slov *binary digit*, čili dvojkové číslo. Bit je takovou informací, kterou již nelze dělit. Proto se množství informace měří v bitech. Tato jednotka se značí písmenem *b*.

ULOŽENÍ INFORMACE

V souvislosti s použitím dvojkové soustavy pro uložení dat v paměti počítače je bit také označením pro dvojkovou číslici 0 nebo 1, tedy pro *uložení* nejmenšího množství *informace*.

DATA

Jakékoli vyjádření skutečnosti v podobě číslicových, abecedních nebo jiných grafických znaků, jimž lze přisoudit určitý význam a které lze přenášet a uchovávat

v paměti a i jinak je zpracovávat, se nazývá *data* nebo údaje. V počítačích se pojmem *data* označují jakékoli údaje, které jsou zpracovány programem.

Budeme se zabývat především zobrazením informace v číslicové technice používané v počítačích. Jedním z charakteristických rysů číslicové techniky je tvoření a používání kódů.

KÓDOVÁNÍ

Při *kódování* se převádí text nebo jiný soubor znaků do nového souboru. Obvykle se vytvoří dvě množiny slov, čísel nebo jiných znaků a podle dohodnutého systému pravidel pro jednoznačné přiřazení se prvky z první množiny vyjádří odpovídajícími prvky z druhé množiny.

KÓD

Kód sám o sobě je předpis, jak k sobě jednoznačně přiřadit prvky dvou množin, dvou seznamů. Prvky jedné množiny mohou být například úkony, které je počítač schopen vykonávat a prvky druhé množiny mohou být například čísla. Na číslo můžeme také nazírat jako na skupinu (na slovo), vytvořenou z příslušných znaků nebo z číslic.

Znak je jeden prvek z množiny smluvených prvků (může to být písmeno, číslice, značka, symbol), který je určen pro vyjádření, přenos nebo uchování informací, a kterému je přiřazen určitý význam. Seznam všech přípustných znaků, tedy množina všech znaků, se nazývá *abeceda*.

1.5 Číselné soustavy a kódy

Pro nás je nejběžnější *desítková (dekadická) soustava*, vycházející z deseti prstů, ale může být i jiná.

DVOJKOVÁ ČÍSELNÁ SOUSTAVA

Číslicové přístroje a zařízení jsou složeny z mnoha spínačů, které rozlišují jen dva stavy: zapnuto a vypnuto. Proto je výhodná k jejich popisu *dvojková (binární) číselná soustava*.

Dvojková soustava má za základ číslo 2 a používá dvou znaků 1 a 0. Tento jeden znak nazýváme dvojkové číslo (bit).

SROVNÁNÍ DESÍTKOVÉ A DVOJKOVÉ SOUSTAVY

Příklad desítkového čísla:

$$\begin{aligned} 7438 &= 7000 + 400 + 30 + 8 \\ &= 7 \cdot 10^3 + 4 \cdot 10^2 + 3 \cdot 10^1 + 8 \cdot 10^0 \end{aligned}$$

kde 7, 4, 3, 8 jsou čísla a 10^3 , 10^2 , 10^1 , 10^0 jsou tzv. *poziční váhy*.

Příklad dvojkového čísla:

$$1101 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$$

kde 1, 0 jsou čísla a 2^3 , 2^2 , 2^1 , 2^0 jsou poziční váhy, které můžeme sčítat.

$$1101 = 8 + 4 + 0 + 1 = 13$$

Dvojkové číslo 1101 představuje v desítkové soustavě číslo 13.

Dle uvedeného příkladu můžeme provádět převody mezi oběma soustavami.

PŘEVOD BINÁRNÍHO ČÍSLA NA DEKADICKÉ

Jednoduše sečteme váhy pozic, kde jsou čísla 1.

Například:

$$\begin{aligned} 1101 &= 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 \\ &= 8 + 4 + 0 + 1 = 13 \end{aligned}$$

Pro jednoznačnost zapisujeme takto:

$$1101_{(2)} = 13_{(10)}$$

PŘEVOD DEKADICKÉHO ČÍSLA NA BINÁRNÍ

a) metoda rozkladu

Dekadické číslo rozložíme na součty mocnin 2^n , tj.

$$\begin{aligned} 2^6 + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 &= \\ 64 + 32 + 16 + 8 + 4 + 2 + 1 &. \end{aligned}$$

Potom dekadické číslo složíme z těchto mocnin a dvojkové číslo sestavíme tak, že píšeme 1 tam, kde je mocnina zastoupena a 0 tam, kde mocnina chybí.

Například: $13 = 8 + 5 = 8 + 4 + 0 + 1$

píšeme: 1 1 0 1

Pro jednoznačnost zapisujeme takto:

$$13_{(10)} = 1101_{(2)}$$

b) metoda opakovaného dělení

Příklad převodu desítkového čísla 13 na dvojkové:

$$13 : 2 = 6 \text{ (napíšeme dolů), zbytek je } 1 \Rightarrow$$

$$6 : 2 = 3 \text{ (napíšeme dolů), zbytek je } 0 \uparrow$$

$$3 : 2 = 1 \text{ (napíšeme dolů), zbytek je } 1 \uparrow$$

$$1 : 2 = 0 \text{ zbytek je } 1 \uparrow$$

Zbytky napíšeme zdola nahoru ve směru šipek, tj. **1101**.

SEČÍTÁNÍ DVOJKOVÝCH ČÍSEL

Dvojková čísla lze samozřejmě sečítat, a to podobně jako desítková:

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 10 \text{ (součet je 0 a 1 je převod do vyššího řádu).}$$

Např:

$$\begin{array}{r} 1\ 1\ 1\ 1\ 0 = 16 + 8 + 4 + 2 + 0 = 30 \\ 0\ 1\ 1\ 0\ 1 = 8 + 4 + 0 + 1 = 13 \\ \hline 1\ 0\ 1\ 0\ 1\ 1 = 32 + 0 + 8 + 0 + 2 + 1 = 43 \end{array}$$

BCD KÓD

Tabulka převodu všech číslic desítkové soustavy (tj. 0 až 9) do dvojkové soustavy se nazývá *kód BCD* (Binary Coded Decimal) – viz *tabulku 1.1*.

Tabulka. 1.1 BCD kód

desítkové číslo	kód BCD			
	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Je to čtyřbitový kód, který se používá pro přímé binární kódování čísel v desítkové soustavě do soustavy dvojkové. Každá desítková číslice je v kódu BCD samostatně vyjádřena jako číslo ve dvojkové soustavě. Při kódování jsou přiřazeny jednotlivým bitům zprava doleva postupně váhy $2^0 = 1$, $2^1 = 2$, $2^2 = 4$, $2^3 = 8$. Desítková číslice je pak vyjádřena jako součet těch vah, na jejichž místě jsou v dvojkové soustavě jedničky. Protože je pro znaky 8 a 9 třeba čtyřmístný dvojkový kód, vyjadřuje se pro jednotnost každý znak desítkové soustavy v BCD kódu čtyřmístným kódem.

Pro zobrazení čísla v počítači se používá nejčastěji dvojková soustava, zřídka kód BCD. Ten se používá pro zobrazení desítkového čísla tak, že se každá číslice převede samostatně dle kódu BCD na čtyřmístné binární číslo. Tato jednotlivá čísla se proto musí od sebe oddělit mezerou.

Např. číslo 7438 lze dle *tabulky 1.1* vyjádřit jako

$$7438_{(10)} = 0111 \mid 0100 \mid 0011 \mid 1000_{(BCD)}$$

Nesmíme ovšem zaměňovat toto vyjádření s přímým převodem dekadického čísla do dvojkové soustavy. Stejně číslo z uvedeného příkladu má v poziční dvojkové soustavě hodnotu:

$$7438_{(10)} = 1110100001110_{(2)}$$

SOUSTAVA HEXADECIMÁLNÍ (ŠESTNÁCTKOVÁ)


Od dvojkové soustavy je odvozena soustava *osmičková* (oktalová) a soustava *šestnáctková* (hexadecimální). Zde si ukážeme pouze princip a použití často používané soustavy šestnáctkové.

Tabulka 1.2 Šestnáctkový kód

desítkové číslo	dvojkové číslo	šestnáctkové číslo
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Základem této soustavy je 2^4 , což je 16. Hexadecimální kód je čtyřbitový kód. Tento kód, na rozdíl od kódu BCD, využívá k vyjádření šestnácti různých znaků všech šestnácti kombinací čtyř bitů. Obdobně jako u BCD kódu se i na tento kód můžeme dívat jako na zestručnění zápisu čísla ve dvojkovém kódu. V tomto případě se nahradí postupně skupiny čtyř bitů zprava vždy jednou hexadecimální číslicí (tabulka 1.2).

Příklady:

$$124_{(10)} = \boxed{0111} \boxed{1100}_{(2)} = 7C_{(16)}$$


$$7438_{(10)} = 1110100001110_{(2)} = 1D0E_{(16)}$$

Mají-li být data převedená do dvojkové soustavy prakticky využita, nestačí je pouze uspořádat a zpracovávat. Data se musí také uchovávat, a to jednak natrvalo,

a jednak jako mezivýsledek k dalšímu zpracování. K uchování dat slouží různé typy *paměti*. Ukládání dat v paměti je třeba vhodně zorganizovat, aby se data snadno ukládala a snadno vyhledávala. Proto je celá paměť rozdělena na menší samostatné části, které jsou z hlediska ukládání údajů vzájemně nezávislé.

BUŇKA PAMĚTI

Základní část takto rozdělené paměti se nazývá *buňka paměti*. Do ní lze uložit informaci o velikosti 1 bit.

KAPACITA PAMĚTI

Pořadové číslo každé buňky se nazývá jeho *adresou* a celkový počet těchto buněk *kapacitou paměti*.

BYTE

Byte (čti bajt, symbol B), neboli *slabika*, obsahuje 8 bitů a je základní jednotkou pro členění a adresování operační paměti. Do jednoho bytu lze uložit číslo z intervalu od 0 do 255, neboť 2^8 je 256. Pokud se k uložení určitého znaku nevyužije všech osm bitů, tak se volné bity vyplní nulami.

Jednotky bit i byte jsou příliš malé pro určení množství zpracovaných dat. Proto byly zavedeny jejich násobky, které ale mají poněkud jiný význam než v soustavě SI:

1 B =	8 b
1 Kb =	1 024 b = 2^{10} b
1 KB =	1 024 B = 2^{10} B
1 MB =	1 048 576 B = 2^{20} B
1 GB =	1 073 741 824 B = 2^{30} B

Někdy se jako informační celek nezpracovává byte (8 bitů), ale *slovo* neboli *word*. Používá se slovo o délce dva byty (16 bitů), nebo čtyři byty (32 bitů).

1.6 Dvuhodnotové zobrazení čísel a znaků

Celé kladné číslo je v paměti uloženo zpravidla jako uspořádaná 16tice bitů, která představuje hodnotu čísla ve dvojkové soustavě. Jednotlivé bity jsou označeny exponentem 0 až 15, na který je umocněn základ – číslo 2.

ZOBRAZENÍ ČÍSEL

Například číslo:

$$123_{(10)} = 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$$

neboli

$$123_{(10)} = 11110011 \text{ je v paměti uloženo takto:}$$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	1	1

Chybějící dvojková čísla jsou obsazena nulami.

ZOBRAZENÍ ZNAKŮ

Výchozím předpokladem pro zobrazení znakových údajů je stanovení nějaké abecedy, neboli množiny přípustných znaků pro daný počítač.

Tato abeceda zpravidla obsahuje 26 velkých a malých písmen latinské abecedy, číslice 0 až 9, znak mezera a řadu dalších speciálních znaků (závorky, tečka, čárka, otazník atd.). Každý znak zakódujeme do určité uspořádané posloupnosti bitů pevné délky.

KÓD ZNAKŮ

Toto přiřazení, které se provádí většinou formou tabulky, se nazývá *kód znaků*. Mezi nejpoužívanější kódy patří sedmibitový kód ASCII (American Standard Code of Information Interchange) a kód ISO-7.

Tyto kódy jsou téměř totožné a můžeme jimi zakóduvat až 128 znaků. 96 znaků připadá na zobrazitelné znaky (malá a velká písmena, číslice, závorky atd.) a 32 znaků je řídicích.

Pro vyjádření českých znaků, kde jsou navíc háčky a čárky, se musí používat rozšířený ASCII kód. Tento kód používá pro kódování 8 bitů a je schopen vyjádřit 256 znaků.

V *tabulce 1.3* je ukázka principu kódování několika znaků v kódu ASCII. Např. písmeno A představuje v binárním kódu 100 0001 (viz poloha šipek v tabulce), nebo v hexadecimálním kódu 41.

Tabulka 1.3 Sedmibitový kód ASCII – princip

(16)	(16)	0	1	2	3	4	5	6	7
	(2)	000	001	010	011	100	101	110	111
0	0000	NUL				↑	P		
1	0001	←			1	A	Q	a	
2	0010				2	B		b	
3	0011				3	C		c	
4	0100					D		d	
5	0101					E			
.	.								
E	1110								
F	1111								

Kontrolní otázky – test T12

1. Co představuje pojem informace?
2. Co je to signál?
3. Jaký je rozdíl mezi spojitým a nespojitým signálem?
4. Čemu se říká kvantování?
5. Co je to bit?
6. Vysvětlete pojem data!
7. K čemu slouží kódování?
8. Co je to kód?
9. Jaký je princip dvojkové číselné soustavy?
10. Převedte dvojkové číslo 1101 do desítkové soustavy!
11. Převedte číslo 27 do dvojkové soustavy!
12. Sečtěte ve dvojkové soustavě čísla 0110 a 1011!
13. Jaký převod používá kód BCD?
14. K čemu se využívá hexadecimální (šestnáctková) soustava?
15. Převedte číslo 01001011 do šestnáctkové soustavy!
16. Jak lze uchovat zpracovávaná data?
17. Co je to 1 byte?
18. Kolik čísel lze uložit do paměti 1 B?
19. Kolikrát je větší 1 MB než 1 kB?
20. K čemu slouží kód ASCII?

2 LOGICKÉ FUNKCE A OBVODY

2.1 Základní pojmy, význam logických obvodů

Logický obvod je takový systém, jehož vstupní i výstupní veličiny nabývají jen dvou hodnot. Pracují tedy s dvouhodnotovým signálem *logická nula (0)* a *logická jednička (1)*.

Těmto hodnotám přiřazujeme informaci:

1 = zapnuto,	0 = vypnuto,	nebo
1 = proud prochází,	0 = proud neprochází,	nebo
1 = vyšší napětí,	0 = nižší napětí.	

DVOJKOVÁ ČÍSELNÁ SOUSTAVA

Vzhledem k této skutečnosti je pro popis takovýchto obvodů velmi vhodná *dvojková číselná soustava*.

Dvojková číselná soustava má pouze dva číselné symboly 0 a 1, které používáme k vyjádření hodnoty nespojitého signálu, resp. stavu logického obvodu.

ČÍSLICOVÉ OBVODY

Logické obvody můžeme proto nazývat *číslíkovými obvody*.

PRAVDIVOSTNÍ TABULKA

K určení jednoho výstupního stavu může být rozhodující logický stav dvou nebo více vstupních proměnných. Vzájemný vztah vstupů a výstupů nejlépe popisuje *pravdivostní tabulka*. Vstupy označujeme symboly A, B, výstup symbolem Y.

BOOLEOVA ALGEBRA

Návrh a sledování činnosti logických obvodů umožňuje tzv. *Booleova logická algebra*.

KOMBINAČNÍ LOGICKÉ OBVODY

Logické obvody dělíme na *kombinační* a *sekvenční*.

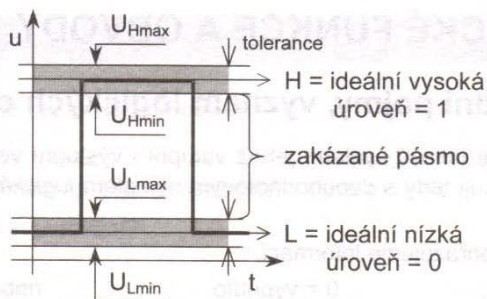
Výstup *kombinačních* obvodů je určen jen kombinací vstupních veličin.

SEKVENČNÍ LOGICKÉ OBVODY

Výstup *sekvenčních* obvodů je určen nejen kombinací vstupních veličin, ale i hodnotami předcházejícího stavu logického obvodu. Z toho plyne, že *sekvenční* logický obvod má paměť.

Zobrazení stavu logických proměnných se provádí nejčastěji napětíovým signálem – viz *obr. 2.1*. Úroveň 1 značíme H (high = vysoký) a úroveň 0 L (low = nízký).

Význam číslíkové techniky mimořádně vzrostl. Uplatňuje se především v oblasti číslíkových počítačů a v oblasti automatizovaných systémů řízení, ale proniká i do analogové techniky (digitální zpracování spojitého signálu zvukového i obrazového).



Obr. 2.1 Úrovnňová pásma logického signálu

2.2 Booleova algebra

V polovině 19. století vypracoval irský matematik G. Boole matematickou logiku jako aplikaci matematiky v oblasti logiky. Nazývá se algebra logiky nebo Booleova algebra. Je to souhrn pravidel a zákonů, které umožňují pracovat s logickými výroky jako s logickými proměnnými a funkcemi, a to formou algebraických operací.

LOGICKÁ FUNKCE

K vyjádření libovolné *logické funkce* se v Booleově algebře používají jen tři základní funkce:

logický součet, logický součin a logická negace.

LOGICKÁ OPERACE

Těmito základními funkcemi můžeme vyjádřit libovolnou *logickou operaci*.

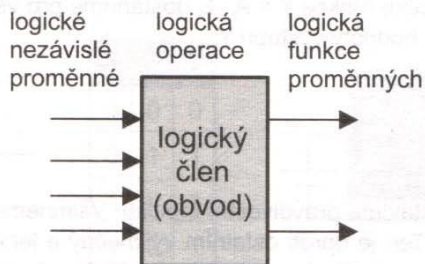
Logické proměnné a logické funkce i konstanty nabývají v Booleově algebře jen dvou hodnot, takže vlastně tato algebra počítá ve dvojkové soustavě.

Booleova algebra našla své praktické uplatnění až osmdesát let po svém vzniku, a to jako vhodný matematický aparát pro řešení logických obvodů. Využívá úsporný algebraický zápis logických vztahů. Pomocí zjednodušování a různých úprav logických funkcí umožňuje Booleova algebra hospodárný (minimalizační) návrh zařízení.

LOGICKÝ OBVOD

Booleovou algebrou můžeme vyjádřit chování *logického obvodu* (obr. 2.2). Naopak to znamená, že logický obvod slouží k realizaci matematické funkce, tj. ke zhmotnění logické operace. Logická operace je zpracování logických vstupních proměnných na určitou logickou funkci. Všeobecně je to předpis, který každé kombinaci (popř. časové posloupnosti) hodnot jedné nebo více nezávislých logických proměnných jednoznačně přiřazuje hodnoty jedné nebo více závislých logických proměnných.

Vstupní a výstupní proměnná logických obvodů jsou napětí (signály) obdélníkového průběhu dle obr. 2.1.



Obr. 2.2 Logický obvod

Pravdivostní tabulka jednoznačně určuje hodnotu logické funkce pro všechny možné kombinace hodnot vstupních nezávisle proměnných. Ve dvouhodnotové Booleově algebře se počet kombinací pro n proměnných rovná 2^n . Pro dva vstupy logického obvodu je proto počet možných kombinací a tudíž i různých výstupů $2^2 = 4$.

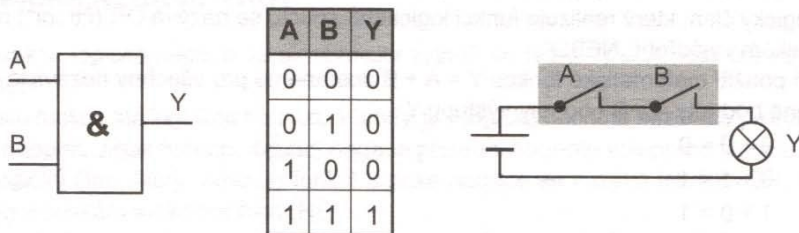
V dalším výkladu si probereme zmíněné tři základní funkce Booleovy algebry: logický součin, logický součet a logickou negaci. Pro úplný popis chování funkce uvedeme vždy grafickou značku, pravdivostní tabulku, realizaci obvodu a průběhy signálů.

2.3 Základní logické funkce a logické členy

LOGICKÝ SOUČIN – AND

Logický součin je logická funkce ve tvaru $Y = A \cdot B$, kde A a B jsou hodnoty vstupních veličin a Y je hodnota výstupu.

Logický člen, který realizuje funkci logického součinu se nazývá AND (angl. výraz čti „end“) nebo v českém vyjádření „A“. Jeho grafická značka, pravdivostní tabulka a realizace jsou na obr. 2.3. Vnitřní zapojení obvodu AND lze znázornit jako sériové zapojení dvou zapínacích kontaktů.



Obr. 2.3 AND grafická značka, pravdivostní tabulka a realizace

Při použití matematické funkce $Y = A \cdot B$ dostaneme pro všechny nezávisle proměnné hodnoty A a B hodnoty výstupu Y:

$$0 \cdot 0 = 0$$

$$0 \cdot 1 = 0$$

$$1 \cdot 0 = 0$$

$$1 \cdot 1 = 1$$

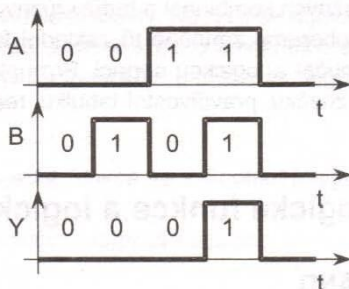
Ze všech údajů sestavíme pravdivostní tabulku. Všimněme si posledního řádku pravdivostní tabulky. Ten je oproti ostatním výjimečný a jeho slovním vyjádřením můžeme jednou větou vyjádřit chování tohoto logického obvodu AND:

Výstup je 1 tehdy a jen tehdy, jsou-li oba vstupy 1.

Celou činnost obvodu můžeme vyjádřit takto:

Žárovka svítí jen při sepnutí obou kontaktů A a B.

Chování logického členu je možné také znázornit graficky jako časovou závislost výstupu logického členu na kombinaci impulzních vstupů (obr. 2.4).



Obr 2.4 AND – grafická závislost výstupu na kombinaci vstupů

LOGICKÝ SOUČET – OR

Logický součet je logická funkce ve tvaru, $Y = A + B$ kde A a B jsou hodnoty vstupních veličin a Y je hodnota výstupu. Na obr. 2.5 je mimo grafické značky též pravdivostní tabulka a realizace OR. Vnitřní zapojení obvodu OR si můžeme představit jako paralelní zapojení dvou zapínacích kontaktů.

Logický člen, který realizuje funkci logického součtu se nazývá OR (čti „or“) nebo v českém vyjádření „NEBO“.

Při použití matematické funkce $Y = A + B$ dostaneme pro všechny nezávisle proměnné hodnoty A a B hodnoty výstupu Y:

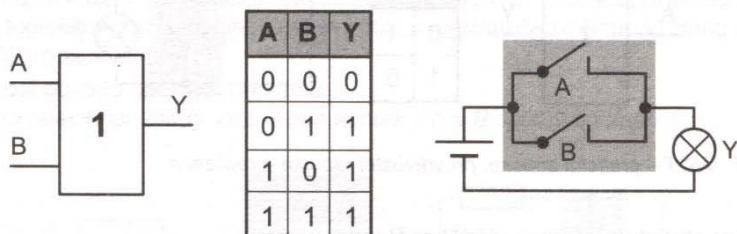
$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 1$$

Uvedené údaje opět tvoří pravdivostní tabulku.



Obr. 2.5 OR – grafická značka, pravdivostní tabulka a realizace

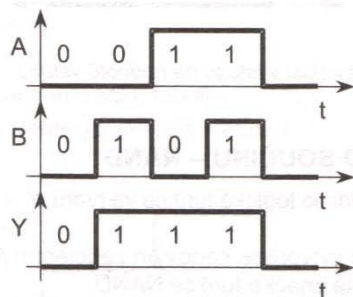
Opět můžeme vyjádřit jednou větou chování logického obvodu AND:

Výstup je 0 tehdy a jen tehdy, jsou-li oba vstupy 0.

Činnost obvodu můžeme vyjádřit také takto:

Žárovka svítí, je-li zapnut alespoň jeden z kontaktů A nebo B.

Grafické znázornění závislosti vstupních a výstupních veličin je na obr. 2.6.



Obr. 2.6 OR – grafická závislost výstupu na kombinaci vstupů

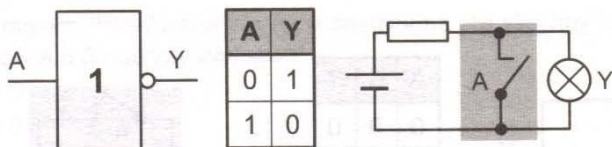
LOGICKÁ NEGACE – NOT

Funkce *logické negace* se algebraicky vyjádří ve tvaru: $Y = \bar{A}$, což čteme jako „A non“.

Tato funkce má logickou hodnotu 1 tehdy, je-li logická hodnota nezávisle proměnné 0 a naopak. Jinak řečeno, logická negace převrací hodnotu vstupního signálu.

Logický člen, který realizuje funkci logické negace se nazývá INVERTOR, NOT nebo v českém vyjádření člen „NE“.

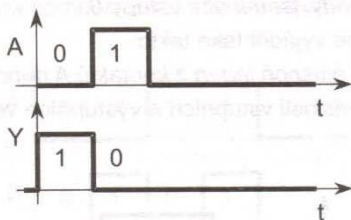
Na obr. 2.7 je grafická značka, realizace a pravdivostní tabulka, obr. 2.8 znázorňuje odpovídající průběhy napětí.



Obr. 2.7 NOT – grafická značka, pravdivostní tabulka a realizace

Činnost obvodu můžeme vyjádřit opět jednou větou:
Žárovka svítí jen tehdy, není-li kontakt A zapnut a naopak.

Nebo obecně:
Výstup je 0, je-li vstup 1 a naopak.

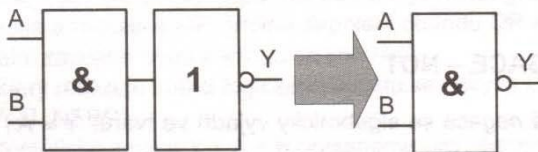


Obr. 2.8 NOT – grafická závislost výstupu na hodnotě vstupu

NEGACE LOGICKÉHO SOUČINU – NAND

Negace logického součinu je logická funkce ve tvaru $Y = \overline{A \cdot B}$. Tato funkce vznikne negací funkce AND.

Příslušný logický obvod vytvoříme sériovým zapojením AND a NOT dle obr. 2.9, kde je nakreslena i grafická značka funkce NAND.



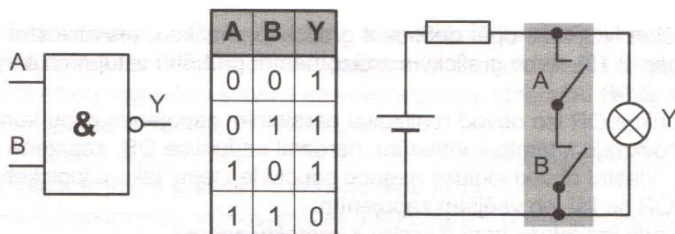
Obr. 2.9 NAND jako negace funkce AND a jeho grafická značka

Logický člen NAND lze opět definovat pravdivostní tabulkou, jeho činnost znázornit realizací (obr. 2.10), nebo grafickým znázorněním průběhů vstupních a výstupních veličin (obr. 2.11).

Obvod lze realizovat sériovým zapojením dvou kontaktů (viz AND), ale výstup (žárovka) je k těmto kontaktům, na rozdíl od členu AND, zapojen paralelně. Vlastní obvod logické negace součinu je stejný jako u logického součinu, od členu AND se liší vnějším zapojením.

Činnost obvodu opět jednou větou:

Žárovka nesvítí jen tehdy, jsou-li oba kontakty A a B současně zapnuty.



Obr. 2.10 NAND – grafická značka, pravdivostní tabulka a realizace

Pro každý řádek pravdivostní tabulky lze opět spočítat výstup Y:

$$0 \cdot 0 = 0 \Rightarrow \text{invert} = 1$$

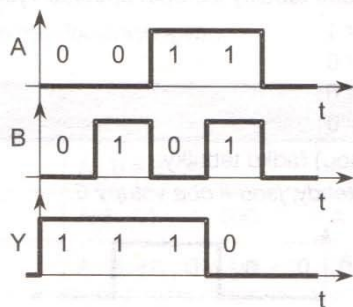
$$0 \cdot 1 = 0 \Rightarrow \text{invert} = 1$$

$$1 \cdot 0 = 0 \Rightarrow \text{invert} = 1$$

$$1 \cdot 1 = 1 \Rightarrow \text{invert} = 0$$

Vyjádříme čtvrtou (výjimečnou) řádku tabulky:

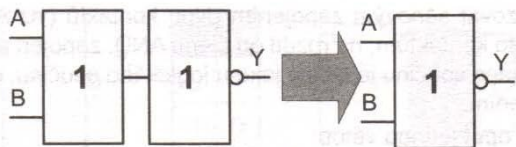
Výstup je 0 tehdy a jen tehdy jsou-li oba vstupy 1.



Obr. 2.11 NAND – grafická závislost výstupu na kombinaci vstupů

NEGACE LOGICKÉHO SOUČTU – NOR

Negace logického součtu je logická funkce ve tvaru $Y = \overline{A + B}$. Tato funkce vznikne negací OR. Příslušný logický obvod vytvoříme sériovým zapojením OR a NOT dle obr. 2.12, kde je nakreslena i grafická značka funkce NOR.

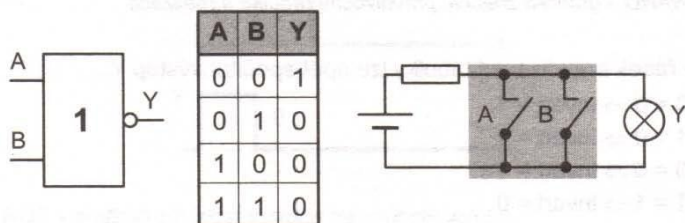


Obr. 2.12 NOR jako negace funkce OR a jeho grafická značka

Logický člen NOR lze opět definovat grafickou značkou, pravdivostní tabulkou, realizací (obr. 2.13), nebo grafickým znázorněním průběhů vstupních a výstupních veličin (obr. 2.14).

Jako u funkce OR lze obvod realizovat paralelním zapojením dvou kontaktů, ale výstup (žárovka) je k těmto kontaktům, narozdíl od funkce OR, zapojena paralelně (obr. 2.13). Vlastní obvod logické negace součtu je stejný jako u logického součtu, od funkce OR se liší jen vnějším zapojením.

Žárovka svítí jen tehdy, není-li žádný z kontaktů zapnut.



Obr. 2.13 NOR – grafická značka, pravdivostní tabulka a realizace

Pro každý řádek pravdivostní tabulky lze opět spočítat výstup Y:

$$0 + 0 = 0 \Rightarrow \text{invert} = 1$$

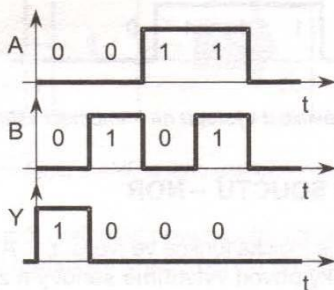
$$0 + 1 = 1 \Rightarrow \text{invert} = 0$$

$$1 + 0 = 1 \Rightarrow \text{invert} = 0$$

$$1 + 1 = 1 \Rightarrow \text{invert} = 0$$

Vyjádříme první (výjimečnou) řádku tabulky:

Výstup je 1 tehdy a jen tehdy, jsou-li oba vstupy 0.



Obr. 2.14 NOR – grafická závislost výstupu na kombinaci vstupů

Rozšířením tří základních operací Booleovy algebry o negaci logického součinu a o negaci logického součtu jsme získali takto pět základních členů, které tyto operace zajišťují: AND, OR, NOT, NAND a NOR. Z těchto členů, jako základních kamenů, lze sestavit libovolný logický obvod až ke složitosti počítače.

2.4 Pravidla Booleovy algebry

Při používání Booleovy algebry je nutno si uvědomit, že platí pouze pro logickou algebru, i když řada vztahů je shodná či podobná pravidlům číselné algebry. Číslice 1 a 0 jsou symboly logického stavu a ne velikosti čísla. Operační znaky + a \cdot vyjadřují logickou operaci a ne algebraickou operaci.

Proto v Booleově algebře platí $1 + 1 = 1$, kdežto v číselné algebře platí při sčítání $1 + 1 = 10$ (součet je 0 a přenos do vyššího řádu je 1).

V Booleově algebře jsou definovány *základní vztahy – postuláty* (tabulka 2.1) a od nich odvozená *základní pravidla* (tabulka 2.2).

Tabulka 2.1 Základní vztahy Booleovy algebry

$0 + 0 = 0$	$1 \cdot 1 = 1$	$\bar{1} = 0$
$0 + 1 = 1$	$0 \cdot 1 = 0$	$\bar{0} = 1$
$1 + 0 = 1$	$1 \cdot 0 = 0$	
$1 + 1 = 1$	$0 \cdot 0 = 0$	

Tabulka 2.2 Základní pravidla Booleovy algebry

Pravidlo	Algebraické vyjádření	
komutativní	$A \cdot B = B \cdot A$	$A + B = B + A$
asociativní	$A(B \cdot C) = (A \cdot B)C$	$A + (B + C) = (A + B) + C$
distributivní	$A \cdot B + A \cdot C = A(B + C)$	$(A + B)(A + C) = A + B \cdot C$
vyločení třetího	$A \cdot \bar{A} = 0$	$A + \bar{A} = 1$
agresivnosti hodnot 0 a 1	$A \cdot 0 = 0$	$A + 1 = 1$
neutrálnosti hodnot 0 a 1	$A \cdot 1 = A$	$A + 0 = A$
absorpce	$A \cdot A = A$	$A + A = A$
	$A(A + B) = A$	$A + A \cdot B = A$

Pravidlo	Algebraické vyjádření	
absorpce negace	$A + \bar{A}B = A + B$	$A(\bar{A} + B) = AB$
	$\bar{A} + AB = \bar{A} + B$	$\bar{A}(A + B) = \bar{A}B$
dvojitá negace	$\bar{\bar{A}} = A$	
De Morganovy zákony	$\overline{A \cdot B} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A} \cdot \bar{B}$

Nejdůležitější De Morganův zákon lze vyjádřit takto:

Negací funkce získáme nahrazením každé proměnné její negací a vzájemnou záměnou znaků součtu a součinu.

Např. výraz s negací můžeme upravit takto:

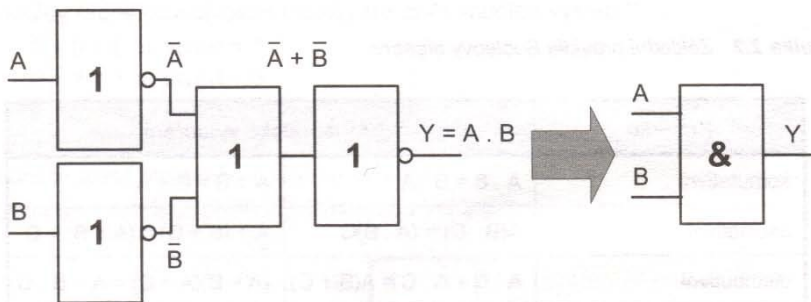
$$\overline{AB + C} = \bar{A}\bar{B} \cdot \bar{C} = \overline{(\bar{A} + \bar{B})} \cdot \bar{C}$$

De Morganovým zákonem lze dokázat vzájemnou nahraditelnost logických členů AND a OR s použitím invertoru.

Obvod na obr. 2.15 po výpočtu

$$Y = \overline{\bar{A} + \bar{B}} = \bar{\bar{A}} \cdot \bar{\bar{B}} = A \cdot B$$

můžeme nahradit pouze jedním členem AND.

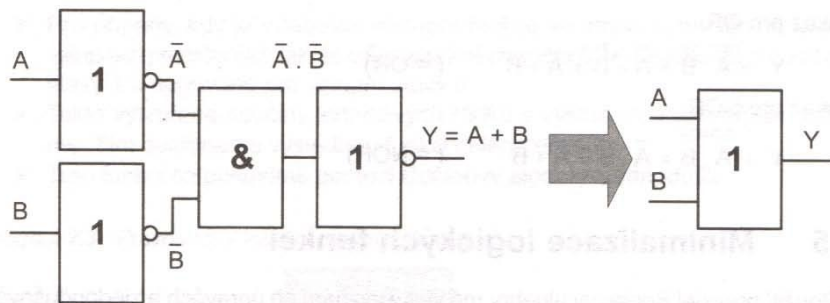


Obr. 2.15 Vytvoření AND z OR

Podobně obvod na obr. 2.16 po algebraickém výpočtu

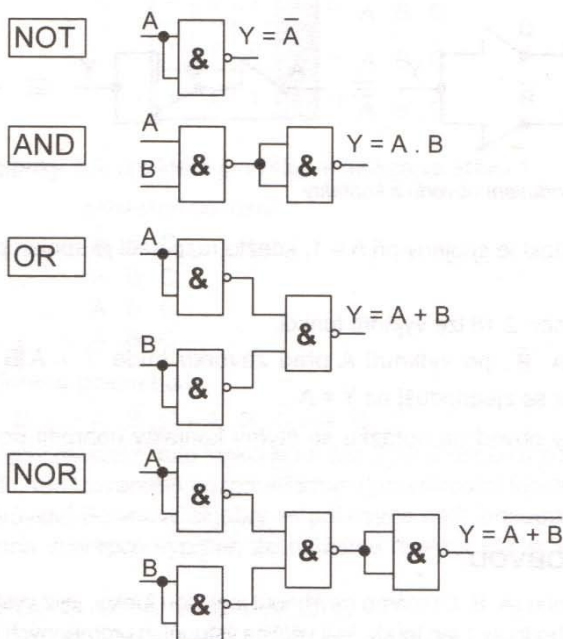
$$Y = \overline{\bar{A} \cdot \bar{B}} = \bar{\bar{A}} + \bar{\bar{B}} = A + B$$

nahradíme členem OR.



Obr. 2.16 Vytvoření OR z AND

Negovaný logický součin NAND tvoří úplný systém logických funkcí. Touto operací je možné realizovat libovolnou kombinační logickou funkci a logickými členy (hradly) NAND lze realizovat nejen AND, OR, NOT a NOR, ale i libovolný kombinační logický obvod (viz obr. 2.17).



Obr. 2.17 Logické obvody sestavené z logických členů NAND

Důkaz pro AND:

$$Y = \overline{\overline{A \cdot B}} = A \cdot B \quad (= \text{AND})$$

Důkaz pro OR:

$$Y = \overline{\overline{A} \cdot \overline{B}} = \overline{\overline{A} + \overline{B}} = A + B \quad (= \text{OR})$$

Důkaz pro NOR:

$$Y = \overline{\overline{\overline{A} \cdot \overline{B}}} = \overline{\overline{A} \cdot \overline{B}} = A + B \quad (= \text{NOR})$$

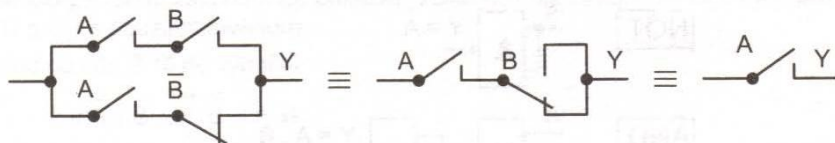
2.5 Minimalizace logických funkcí

Použití pravidel Booleovy algebry má velký význam při úpravách a zjednodušování logických výrazů a tím i schémat zapojení. Ukážeme si postup minimalizace na několika příkladech.

Příklad 1

KONTAKTNÍ OBVOD

Provedeme zjednodušení obvodu na obr. 2.18.



Obr. 2.18 Zjednodušení obvodu s kontakty

Zapínací kontakt je spojený při $A = 1$, kdežto rozpinací je spojen při $B = 0$, neboli pro $\overline{B} = 1$.

Zapojení na obr. 2.18 lze vyjádřit funkcí:

$Y = A \cdot B + A \cdot \overline{B}$, po vytknutí A před závorkou bude $Y = A \cdot (B + \overline{B})$, a jelikož $B + \overline{B} = 1$, výraz se zjednoduší na $Y = A$.

Můžeme tedy obvod na obrázku se čtyřmi kontakty nahradit pouze kontaktem jedním.

Příklad 2

MAJORITNÍ OBVOD

Pro tři proměnné (A, B, C) máme navrhnout logickou funkci, jejíž výstupní proměnná nabývá logické hodnoty 1 jen tehdy, je-li většina vstupních proměnných ve stavu logické hodnoty 1 (tzv. majoritní obvod).

Postup vytvoření logické funkce obvodu

- Do tabulky zapíšeme všechny kombinace vstupních proměnných ($2^3 = 8$) a v těch řádcích, kde jsou dvě nebo tři vstupní proměnné ve stavu 1, přiřadíme výstupním proměnným logickou hodnotu 1 (viz tabulku 2.3).

- Pro případy, kdy je v tabulce výstupní funkce ve stavu 1, napíšeme součin vstupních proměnných tak, že píšeme přímé proměnné (ne negované) pro vstupní stavy 1 a negované pro vstupní stavy 0.
- Takto vytvořené součiny jednotlivých řádků s výstupem 1 jednoduše sečteme. Tím dostaneme výslednou funkci celého obvodu.
- Tuto funkci se pokusíme pomocí Booleovy algebry zjednodušit.

Tabulka 2.3 Pravdivostní tabulka majoritní funkce

A	B	C	Y	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\Rightarrow \bar{A} \cdot B \cdot C$
1	0	0	0	
1	0	1	1	$\Rightarrow A \cdot \bar{B} \cdot C$
1	1	0	1	$\Rightarrow A \cdot B \cdot \bar{C}$
1	1	1	1	$\Rightarrow A \cdot B \cdot C$

Kombinace z tabulky 2.3, pro které je výstupní funkce ve stavu 1:

A B C	příslušné součiny
0 1 1	$\bar{A} \cdot B \cdot C$
1 0 1	$A \cdot \bar{B} \cdot C$
1 1 0	$A \cdot B \cdot \bar{C}$
1 1 1	$A \cdot B \cdot C$

Součtový tvar funkce potom bude:

$$Y = \bar{A} \cdot B \cdot C + A \cdot \bar{B} \cdot C + A \cdot B \cdot \bar{C} + A \cdot B \cdot C$$

Schéma zapojení realizující tuto funkci je na obr. 2.19. Z obrázku je patrná poměrná složitost obvodu, realizovaného bezprostředně z pravdivostní tabulky.

S použitím pravidel Booleovy algebry se pokusíme najít jednodušší ekvivalentní výraz. Ze zákona absorpce vyplývá, že můžeme dvakrát přičíst součin $A \cdot B \cdot C$, protože platí:

$$A \cdot B \cdot C = A \cdot B \cdot C + A \cdot B \cdot C + A \cdot B \cdot C$$

Pak dostaneme:

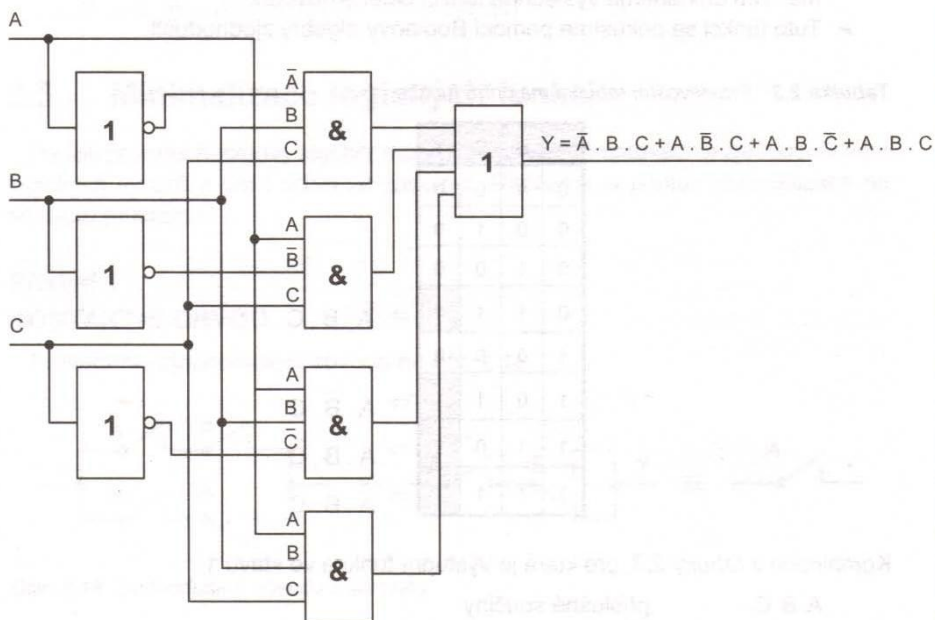
$$Y = (\bar{A} \cdot B \cdot C + A \cdot B \cdot C) + (A \cdot \bar{B} \cdot C + A \cdot B \cdot C) + (A \cdot B \cdot \bar{C} + A \cdot B \cdot C)$$

$$Y = BC(\bar{A} + A) + AC(\bar{B} + B) + AB(\bar{C} + C)$$

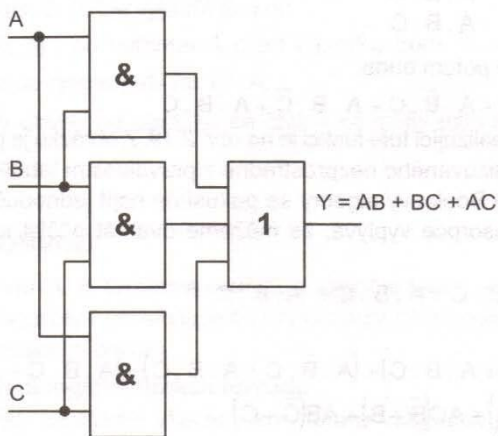
$$Y = BC \cdot 1 + AC \cdot 1 + AB \cdot 1$$

$$Y = AB + BC + AC$$

Schéma obvodu, který výslednou funkci realizuje, je na obr. 2.20. Ve srovnání s obr. 2.19 je podstatně jednodušší, i když realizuje stejnou funkci danou zadáním příkladu.



Obr 2.19 Realizace logické funkce $Y = \bar{A} \cdot B \cdot C + A \cdot \bar{B} \cdot C + A \cdot B \cdot \bar{C} + A \cdot B \cdot C$



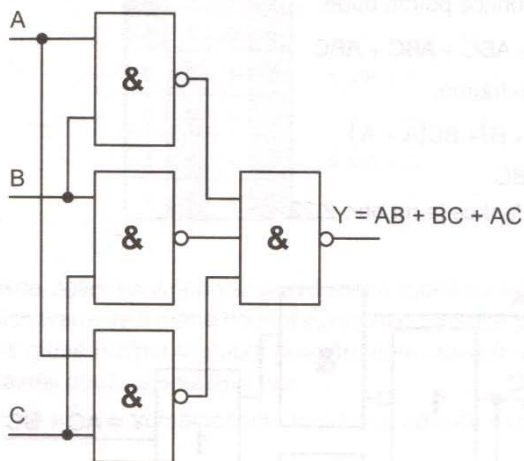
Obr 2.20 Realizace logické funkce $Y = AB + BC + AC$

Funkci $Y = AB + BC + AC$ můžeme dále upravit, např. dvojitou negací a de Morganovým zákonem:

$$Y = \overline{\overline{AB + BC + AC}}$$

$$Y = \overline{\overline{AB} \cdot \overline{BC} \cdot \overline{AC}}$$

Tuto novou funkci můžeme již realizovat jen ze samotných členů NAND (obr. 2.21).



Obr. 2.21 Jiný způsob realizace logické funkce $Y = AB + BC + AC$

Příklad 3 MULTIPLEXER

Návrh logického přepínače (multiplexeru).

Tabulka 2.4 Pravdivostní tabulka přepínače

A	B	C	Y	
0	0	0	0	
0	1	0	0	
1	0	0	1	\overline{ABC}
1	1	0	1	$AB\overline{C}$
0	0	1	0	
0	1	1	1	$\overline{A}BC$
1	0	1	0	
1	1	1	1	ABC

Máme navrhnout takový obvod, který pomocí proměnné C přepíná na jediný výstup buď vstupní proměnnou A (je-li nastaveno $C = 0$), nebo vstupní proměnnou B (pro $C = 1$).

Vyplníme tabulku této logické funkce (tabulka 2.4). V těch řádcích, kde je logická vstupní proměnná $C = 0$, musí být výstup Y totožný se vstupní proměnnou A a je-li $C = 1$, musí být výstup Y totožný s hodnotou B.

Součtový tvar funkce potom bude:

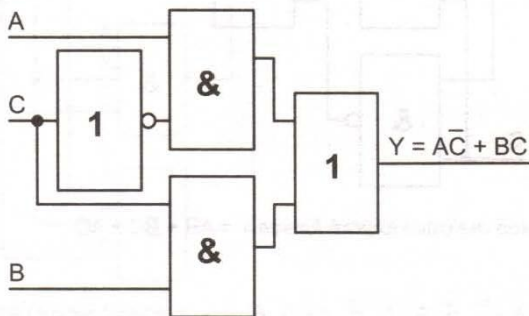
$$Y = A\bar{C} + A\bar{C} + \bar{A}BC + ABC$$

Po úpravách obdržíme:

$$Y = A\bar{C}(\bar{B} + B) + BC(\bar{A} + A)$$

$$Y = A\bar{C} + BC$$

Realizace této funkce je na obr. 2.22.



Obr. 2.22 Schéma přepínače

Poznámka: Další jiný možný způsob vyjádření logické funkce z pravdivostní tabulky je vytvoření základního součinného tvaru.

Pro případy, kdy je v tabulce výstupní funkce ve stavu 0, uděláme součin součtů vstupních proměnných tak, že píšeme přímé proměnné (ne negované) pro vstupní stavy 0 a negované pro vstupní stavy 1.

Při použití stejného příkladu č. 3 (MULTIPLEXER) vytvoříme součty pro ty řádky v tabulce 2.5 pro které je výstup 0 takto:

0 0 0	$A + B + C$
0 1 0	$A + \bar{B} + C$
0 0 1	$A + B + \bar{C}$
1 0 1	$\bar{A} + B + \bar{C}$

Součinný tvar funkce pak je

$$Y = (A + B + C) \cdot (A + \bar{B} + C) \cdot (A + B + \bar{C}) \cdot (\bar{A} + B + \bar{C})$$

což by bylo možné dále upravovat.

Máme navrhnout takový obvod, který pomocí proměnné C přepíná na jediný výstup buď vstupní proměnnou A (je-li nastaveno $C = 0$), nebo vstupní proměnnou B (pro $C = 1$).

Vyplníme tabulku této logické funkce (tabulka 2.4). V těch řádcích, kde je logická vstupní proměnná $C = 0$, musí být výstup Y totožný se vstupní proměnnou A a je-li $C = 1$, musí být výstup Y totožný s hodnotou B.

Součtový tvar funkce potom bude:

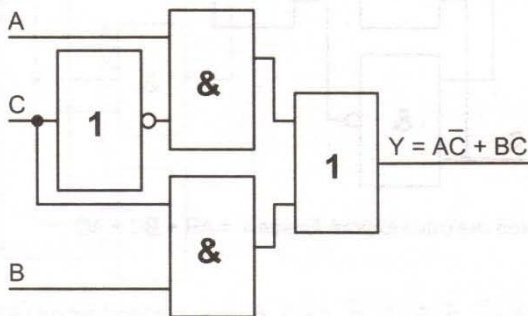
$$Y = A\bar{C} + A\bar{C} + \bar{A}BC + ABC$$

Po úpravách obdržíme:

$$Y = A\bar{C}(\bar{B} + B) + BC(\bar{A} + A)$$

$$Y = A\bar{C} + BC$$

Realizace této funkce je na obr. 2.22.



Obr. 2.22 Schéma přepínače

Poznámka: Další jiný možný způsob vyjádření logické funkce z pravdivostní tabulky je vytvoření základního součinného tvaru.

Pro případy, kdy je v tabulce výstupní funkce ve stavu 0, uděláme součin součtů vstupních proměnných tak, že píšeme přímé proměnné (ne negované) pro vstupní stavy 0 a negované pro vstupní stavy 1.

Při použití stejného příkladu č. 3 (MULTIPLEXER) vytvoříme součty pro ty řádky v tabulce 2.5 pro které je výstup 0 takto:

0 0 0	$A + B + C$
0 1 0	$A + \bar{B} + C$
0 0 1	$A + B + \bar{C}$
1 0 1	$\bar{A} + B + \bar{C}$

Součinný tvar funkce pak je

$$Y = (A + B + C) \cdot (A + \bar{B} + C) \cdot (A + B + \bar{C}) \cdot (\bar{A} + B + \bar{C})$$

což by bylo možné dále upravovat.

2.6 Realizace logických obvodů

Podle fyzikálního principu rozdělujeme logické obvody na *elektrické*, *magnetické*, *pneumatické*, *optické* atd. Elektrické logické obvody mohou být *reléové*, *diodové*, *elektronkové*, *tyristorové*, *tranzistorové*, *laserové* atd.

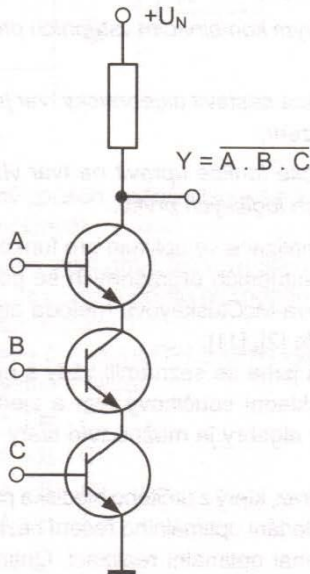
LOGICKÉ ČLENY S BIPOLÁRNÍMI TRANZISTORY

Ukážeme si pouze některé příklady tranzistorových logických obvodů. Tranzistory používané v logických obvodech pracují jako nelineární spínací prvky se dvěma pracovními body. Jeden bod je v závěrné oblasti, tranzistorem prochází minimální (zbytkový) proud, na výstupu tranzistoru je velké napětí (blízké napětí zdroje), tranzistor je zavřený. Druhý pracovní bod leží v oblasti nasycení (saturace), tranzistor je otevřený a prochází jím maximální proud.

Obvody s bipolárními tranzistory mohou mít různé vnitřní vazby:

- *Přímo vázané obvody (DCTL – Direct Coupled Transistor Logic).*

Tyto obvody mají na každém vstupu tranzistor. Tranzistory jsou navzájem propojeny přímo, bez vazebního členu. Sériovým zapojením tranzistorů vznikne člen NAND (obr. 2.23). Bude-li na všech vstupech A, B, C logická hodnota 1, tranzistory se otevřou a na výstupu bude logická hodnota 0, neboť tranzistory uzemní napájecí napětí.

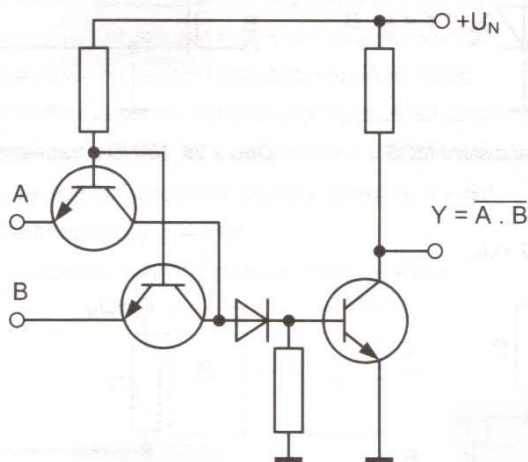


Obr. 2.23 Tranzistorový logický obvod DCTL

- Odporově vázané obvody (RTL – Resistor Transistor Logic).
- Obvody s odporově kapacitní vazbou (RCTL – Resistor Capacitor Transistor Logic).
- Obvody s vazbou dioda-tranzistor (DTL – Diode Transistor Logic).
- Emitorově vázané logické obvody (ECL – Emitter Coupled Logic).
- Obvody s vazbou tranzistor-tranzistor (TTL – Transistor Transistor Logic).

V současné době jsou tyto integrované obvody nejvíce používané, neboť mají velmi dobré dynamické vlastnosti (velkou spínací rychlost), široký sortiment typů a nízkou cenu. Podrobné informace o obvodech TTL uvádí literatura [9], [10].

Příklad zapojení tohoto obvodu je na obr. 2.24. Zapojení ostatních vyjmenovaných obvodů – viz literaturu [4].



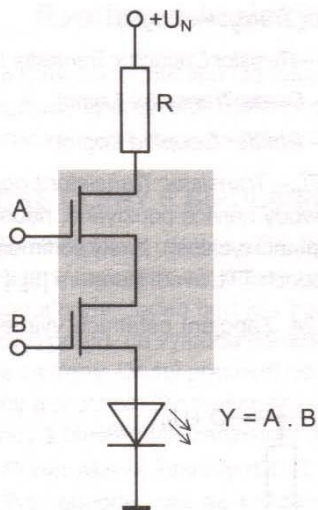
Obr. 2.24 Tranzistorový logický obvod TTL

LOGICKÉ ČLENY S TRANZISTORY MOS

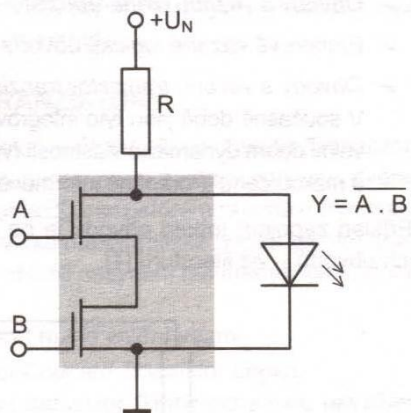
Vlastnosti těchto obvodů jsou v některých aplikacích často velmi výhodné v porovnání s vlastnostmi obvodů s bipolárními tranzistory. Je to především vysoký stupeň integrace daný jednoduchou technologií výroby, malá spotřeba energie, velký vstupní odpor a odolnost proti rušení. Nevýhodou těchto obvodů je malá spínací rychlost (zpoždění 70 až 200 ns). Příklady zapojení logických obvodů s tranzistory MOS jsou na obr. 2.25 až obr. 2.28. Přehled obvodů řady CMOS – viz literaturu [7], [8].

NAND se liší od funkce AND jen vnějším zapojením. Je to nejčastěji používaný logický obvod. Může mít i více vstupů než dva. V integrovaném obvodu je vždy umístěno několik, případně velké množství, logických obvodů.

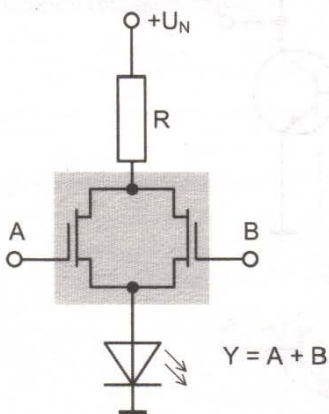
NOR se liší od funkce OR, rovněž jako NAND a AND, jen vnějším zapojením.



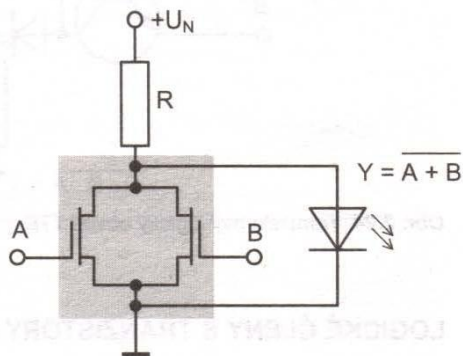
Obr. 2.25 AND z tranzistorů MOS



Obr. 2.26 NAND z tranzistorů MOS

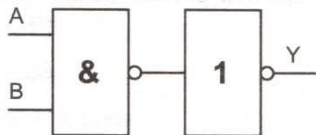


Obr. 2.27 OR z tranzistorů MOS

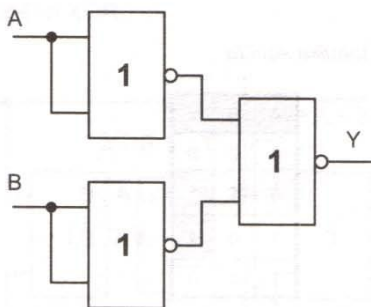


Obr. 2.28 NOR z tranzistorů MOS

1. Vysvětlete pojem logický obvod!
2. Co vyjadřuje pravdivostní tabulka číslicového obvodu?
3. Definujte kombinační logický obvod!
4. Definujte sekvenční logický obvod!
5. Čím se realizuje u logického obvodu vstup (přítomnost) logické 0 nebo 1?
6. Vyjmenujte tři základní logické funkce!
7. Jakými způsoby lze vyjádřit chování logického obvodu?
8. Vyjádřete jednou větou vlastnost logického obvodu AND!
9. Vyjádřete jednou větou vlastnost logického obvodu OR!
10. Napište pravdivostní tabulku logického obvodu NAND!
11. Napište pravdivostní tabulku logického obvodu NOR!
12. Jak se liší vnitřní zapojení logického obvodu AND oproti NAND?
13. Uveďte výsledek pro součet dvou jedniček v Booleově algebře ($1 + 1 = \dots$)!
14. Zjednodušte pomocí Booleovy algebry výraz $A(\bar{A} + B)$!
15. Napište De Morganovy zákony!
16. Odvoďte výslednou funkci Y následujícího obvodu!



17. Zjednodušte následující obvod!



18. Co znamená minimalizace logických funkcí?
19. Uveďte zjednodušeně postup při návrhu kombinačního logického obvodu!
20. Které prvky se používají pro realizaci logických obvodů?

3 KOMBINAČNÍ LOGICKÉ OBVODY

U těchto obvodů je výstup určen jen výhradně kombinací vstupních veličin.

Hodnoty výstupních veličin nezávisí na předcházejícím stavu logického obvodu, což znamená, že kombinační logický obvod neobsahuje paměťové prvky. Základní kombinační obvody jsou tyto:

- XOR neboli Exklusiv OR.
- Sčítačka.
- Generátor parity.
- Kodéry a dekodéry.
- Multiplexery a demultiplexery.
- Binární komparátor.

XOR NEBOLI EXLUSIV OR

Obvod XOR je jedním z nejpoužívanějších kombinačních logických obvodů, takže jej někteří autoři řadí mezi základní logické obvody (tj. AND, NAND, OR, NOR, NOT a XOR). O jeho důležitosti svědčí i to, že je nazýván různými jmény např. *Exklusiv OR*, *EX-OR*, *XOR*, *nerovnost*, *nonekvivalence*, *sčítačka modulo 2* atd. V dalším výkladu budeme používat nejjednodušší název XOR.

Vytvoříme obvod tak, aby splňoval pravidla sčítání ve dvojkové soustavě, tj.:

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 0$$

Dle těchto pravidel sestrojíme pravdivostní tabulku (tabulka 3.1).

Tabulka 3.1 Pravdivostní tabulka součtu

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

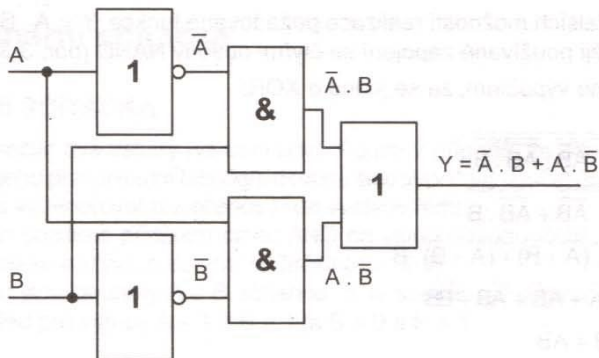
$\Rightarrow \bar{A} \cdot B$

$\Rightarrow A \cdot \bar{B}$

Z druhého a třetího řádku této tabulky vyplývá základní součtový tvar funkce:

$$Y = \bar{A} \cdot B + A \cdot \bar{B}$$

Realizace této funkce je na obr. 3.1 a obvod se nazývá XOR.



Obr. 3.1 Realizace funkce $Y = \bar{A} \cdot B + A \cdot \bar{B}$ (XOR).

Matematickou úpravou funkce $Y = \bar{A} \cdot B + A \cdot \bar{B}$ můžeme vytvořit různé způsoby její realizace.

Dokážeme například, že obvod na obr. 3.2 má stejnou funkci jako původní obvod dle obr. 3.1.

Vypočteme výstupní funkci tohoto obvodu:

$$Y = \overline{A + \overline{A + B} + B + \overline{A + B}}$$

$$Y = \bar{A} \cdot (\overline{A + B}) + \bar{B} \cdot (\overline{A + B})$$

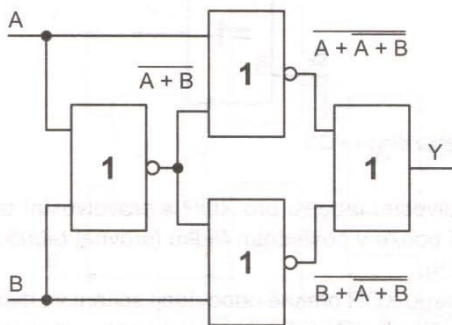
$$Y = \bar{A}(A + B) + \bar{B}(A + B)$$

$$Y = \bar{A}A + \bar{A}B + \bar{B}A + \bar{B}B$$

protože $\bar{A}A = 0$ a $\bar{B}B = 0$, bude

$$Y = \bar{A}B + \bar{B}A$$

Vidíme, že se jedná opět o XOR.

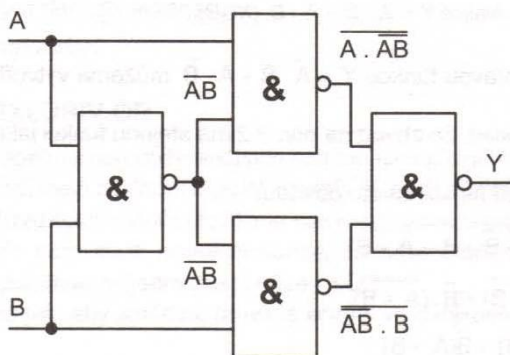


Obr. 3.2 Jiná realizace funkce XOR

Z několika dalších možností realizace požadované funkce $Y = \bar{A} \cdot B + A \cdot \bar{B}$ uvedeme nejčastěji používané zapojení se čtyřmi obvody NAND (obr. 3.3).

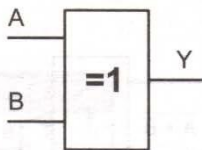
Opět dokážeme výpočtem, že se jedná o XOR:

$$\begin{aligned}
 Y &= \overline{\overline{A \cdot \bar{A} B} \cdot \overline{\bar{A} B}} \\
 Y &= \overline{\overline{A \cdot \bar{A} B} + \overline{\bar{A} B}} \\
 Y &= A \cdot (\bar{A} + B) + (\bar{A} + \bar{B}) \cdot B \\
 Y &= A\bar{A} + AB + \bar{A}B + \bar{B}B \\
 Y &= \bar{A}B + A\bar{B}
 \end{aligned}$$



obr. 3.3 XOR ze čtyřmi obvody NAND, výstup je rovněž $Y = \bar{A} \cdot B + A \cdot \bar{B}$

Obecně se pro zjednodušení kreslení složitějších obvodů používá jednoduchá grafická značka XOR dle obr. 3.4.



Obr. 3.4 Grafická značka členu XOR

Porovnáme-li pravdivostní tabulku pro XOR s pravdivostní tabulkou funkce OR, vidíme, že se obě liší pouze v posledním řádku (srovnej tabulku 3.1 s pravdivostní tabulkou OR v obr. 2.5).

Výstupní hodnoty členu XOR přesně odpovídají sčítání ve dvojkové soustavě a to jej předurčuje pro použití ve veškerých binárních matematických operacích.

XOR je základem tzv. poloviční a úplné sčítačky.

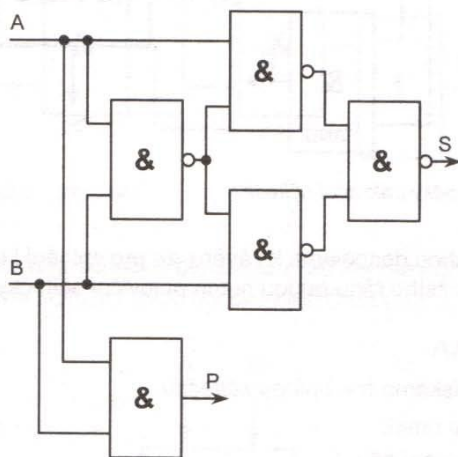
3.1 Binární sčítačka

POLOVIČNÍ SČÍTAČKA

XOR umí sečíst dva vstupy (vstupní bity). Pouze v případě, že oba vstupy jsou 1 (čtvrtá řádka jeho pravdivostní tabulky), dovede sice vypočítat součet (ten je $1 + 1 = 0$), ale nedovede vygenerovat tzv. přenos P do vyššího řádu.

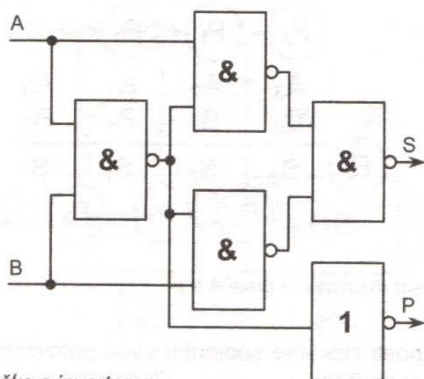
Této funkci docílíme přidáním členu AND na vstup obvodu XOR. Obdržíme tak zapojení, které se nazývá *poloviční sčítačka* (obr. 3.5).

Na obrázku jsou hodnoty A a B sčítance, S je součet a P je přenos do vyššího řádu. Například pro vstupy $A = 1$ a $B = 1$ je $S = 0$ a $P = 1$.



Obr. 3.5 Poloviční sčítačka – obvodové schéma

Stejnou funkci vzniku přenosu získáme připojením invertoru na výstup prvního členu NAND v obvodu XOR (obr. 3.6).

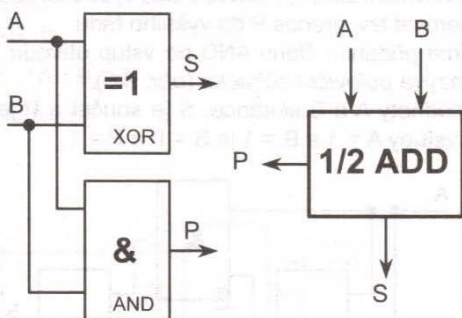


Obr. 3.6 Poloviční sčítačka s invertorem

Blokové schéma poloviční sčítačky a její grafická značka je na obr. 3.7.

Poloviční sčítačka umožňuje sečíst dva vstupy (dva vstupní bity) a přenést ze sčítání případný bit do vyššího řádu (= přenos).

Neumí ale přijmout bit (přenos) z nižšího řádu. To znamená, že může být pouze na začátku, tj. u součtu nejnižšího řádu.



Obr. 3.7 Blokové značky poloviční sčítačky

Jednoduchou úvahou dospějeme k závěru, že pro splnění i tohoto požadavku na přijetí přenosu z nižšího řádu budou nutné poloviční sčítačky dvě.

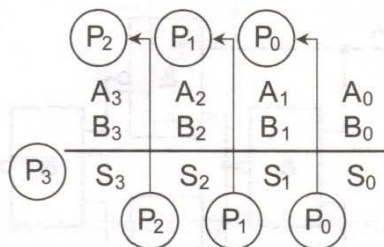
ÚPLNÁ SČÍTAČKA

Jejich spojením získáme tzv. úplnou sčítačku.

Úplná sčítačka musí umět:

- sečíst dva vstupní bity,
- přičíst k tomuto součtu binární bit přenosu z nižšího řádu,
- zjistit další bit přenosu do vyššího řádu a poslat jej dál.

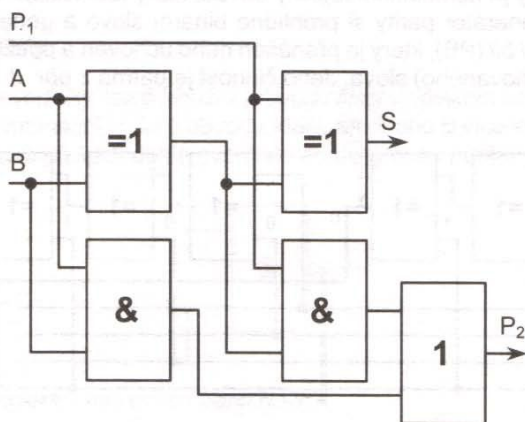
Systém přičítání přenosu do vyššího řádu je stejný jako u desítkové soustavy (obr. 3.8).



Obr. 3.8 Princip sčítání dvojkových čísel A a B

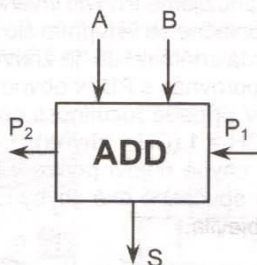
Požadované vlastnosti získáme spojením dvou polovičních sčítaček a přidáním logického členu OR (obr. 3.9).

Úplnou sčítačku můžeme zařadit do libovolného řádu n-bitového součtu. U prvního (respektive nultého) řádu lze použít jen poloviční sčítačku.

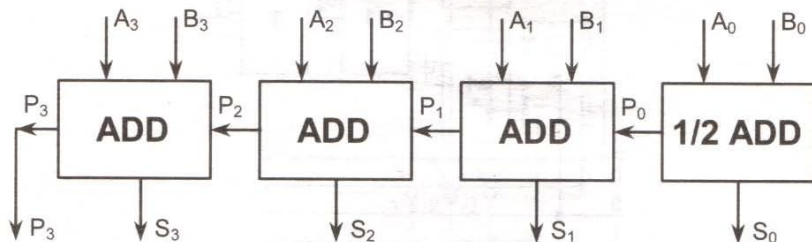


Obr. 3.9 Úplná sčítačka – zapojení

Pokud označíme úplnou sčítačku blokovou značkou dle obr. 3.10, můžeme nakreslit blokové schéma sčítačky dvou čtyřbitových čísel (obr. 3.11).



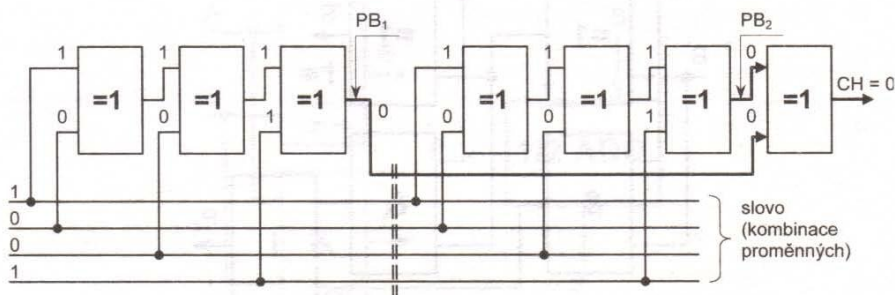
Obr. 3.10 Úplná sčítačka – bloková značka



Obr. 3.11 Sčítačka pro dvě čtyřbitová čísla

3.2 Generátor parity

Generátor parity je kombinační logický obvod, který identifikuje vznik chyby v binárním slově. Generátor parity si prohlédne binární slovo a generuje jednoduchý výstup, tzv. *paritní bit* (PB), který je přenášen nebo uchován a použije se ke kontrole přeneseného (uchovaného) slova. Jeho činnost je patrná z obr. 3.12.



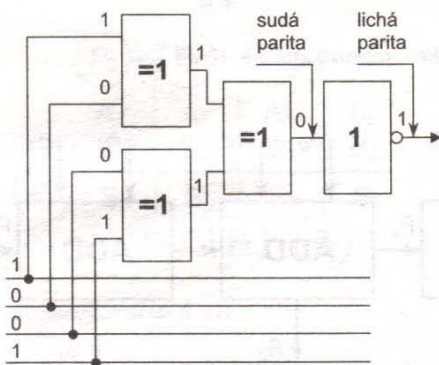
Obr. 3.12 Generátor parity

Činnost generátoru parity

- Je-li v binárním slově sudý počet logických hodnot 1, bude $PB = 0$.
- Je-li v binárním slově lichý počet logických hodnot 1, bude $PB = 1$.

Jedná se o tzv. *sudou paritu*, použijeme-li navíc invertor, mluvíme o tzv. *liché paritě*. Paritní bit PB_1 je přenášen společně se vstupním slovem. Je-li binární slovo čteno z paměti nebo obdrženo na vzdáleném místě, je znovu vyzkoušeno na generátoru parity a nový paritní bit PB_2 je porovnán s PB_1 v obvodu XOR. Oba paritní bity musí být stejné, neboli $PB_1 = PB_2$. V případě rozdílnosti obou bitů vznikla někde chyba v jednom bitu, a výstup bude $CH = 1$ ($CH =$ chyba).

Metoda předpokládá, že se chyba objeví pouze v jedné, a to libovolné, bitové pozici slova. Pokud by vznikly současně dvě chyby najednou, byly by oba paritní bity stejné, a chyba by se neobjevila.



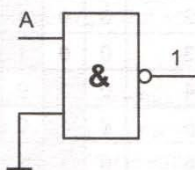
Obr. 3.13 Generátor parity – paralelní zapojení

Generátor parity je možné zapojit se stejnými logickými členy (XOR) i paralelně (obr. 3.13).

3.3 Kodéry

Kodéry jsou převodníky, které slouží k převodu čísel v nějakém kódu do jiného kódu.

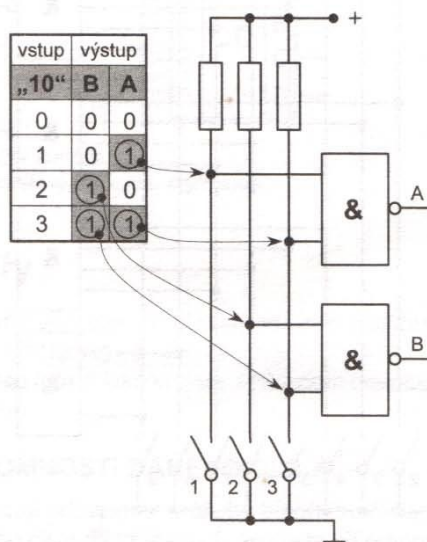
Kodéry jsou kombinační logické obvody, které z jednoho či více vstupů odpovídající desítkové soustavě převádí desítkové číslo do dvojkové soustavy.



Obr. 3.14 Vznik logické 1 na výstupu členu NAND

Zapojení kodéru sestrojíme pomocí kódovací tabulky. K tomu použijeme členy NAND, kde víme, že přivedením logické nuly na alespoň jeden vstup (což realizujeme uzemněním tohoto vstupu – viz obr. 3.14) dostaneme na jeho výstupu logickou 1. Jinak řečeno, pokud je v tabulce na obr. 3.15 logická 1 (A nebo B), musí být příslušný NAND na jednom vstupu, nebo v případě desítkového čísla 3 na obou vstupech, uzemněn. Na obr. 3.15 je tato skutečnost znázorněna šipkami.

Spínače na vstupu kodéru 1, 2 a 3 představují desítkovou soustavu a výstupy A a B z obou členů NAND dvojkovou soustavu.

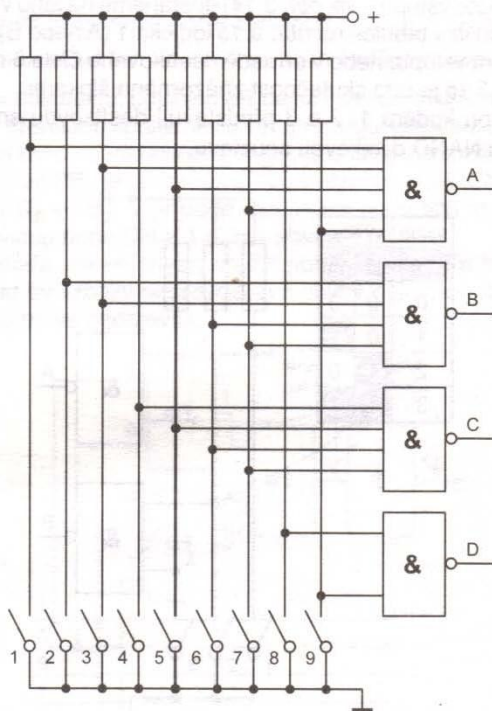


Obr. 3.15 Kodér čísel 0 až 3

KODÉR ČÍSEL 0 AŽ 9 NA ČTYŘBITOVÉ BINÁRNÍ ČÍSLO

Tabulka 3.2 BCD kód 0 až 9

desítkové číslo	kód BCD			
	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1



Obr. 3.16 Kodér 0 až 9 do kódu BCD

Pro sestavení obvodu, který kóduje všech deset čísel desítkové soustavy do dvojkové, použijeme stejný princip jako v předchozím kodéru 0 až 3. Z tabulky převodu desítkového čísla na dvojkové (*tabulka 3.2 – BCD kód*) plyne, že na NAND s výstupem A musí být připojeny spínače desítkových čísel 1, 3, 5, 7 a 9, na NAND s výstupem B pak spínače 2, 3, 6 a 7, na NAND s výstupem C spínače 4, 5, 6 a 7 a na NAND s výstupem D již jen spínače 8 a 9.

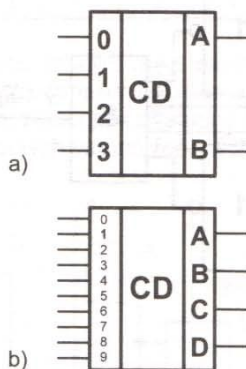
Ze stejného principu plyne i potřebný počet vstupů jednotlivých členů NAND. NAND-A musí mít pět vstupů, neboť ve sloupci A *tabulky 3.2* je pět jedniček, pro NAND-D stačí dva vstupy. Takto snadno sestrojíme celý kodér, i když se jedná o poměrně složité zapojení. Celkové zapojení kodéru je na *obr. 3.16*.

Činnost kodéru

Není-li desítkové číslo, které představuje příslušný spínač, spojeno se zemí, je na vstupu příslušného členu NAND logická 1 a na jeho výstupu 0.

Je-li číslo spínačem uzemněno, vznikne na vstupu příslušného členu NAND logická 0, a tudíž na jeho výstupu vznikne logická 1.

Oba probrané kodéry (0 až 3 a 0 až 9) můžeme zobrazit blokovou značkou (*obr. 3.17*).



Obr. 3.17 Blokové značky kodérů
a) kodér dvoubitový; b) kodér čtyřbitový

3.4 Dekodéry

Dekodéry jsou kombinační logické obvody, které zjišťují buď přítomnost určitého binárního čísla, nebo zjišťují stav binárního čísla.

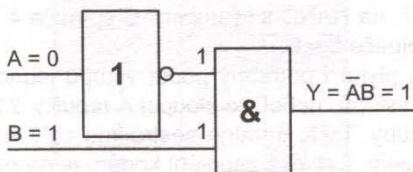
Obvody mají opačnou funkci jako kodéry. Příkladem dekodéru je převodník binárního čísla na desítkové.

A) ZJIŠTĚNÍ PŘÍTOMNOSTI DANÉHO ČÍSLA

Dekodéry, které zjišťují přítomnost určitého binárního čísla, musí převést každou číslici daného binárního čísla na logickou 1, aby byla přítomnost čísla identifikována. Tento požadavek lze zajistit jednoduše použitím invertorů a logického členu AND.

Příklad č. 1: signalizace čísla 01

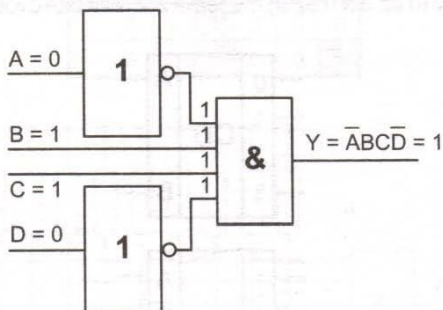
Obvod na obr. 3.18 signalizuje na výstupu přítomnost čísla 01 na vstupu dekodéru. Ostatní dvoubitová čísla mají výstup 0.



Obr. 3.18 Dekodér čísla 01

Příklad č. 2: dekodér čísla 0110

Logická 1 na výstupu dekodéru dle obr. 3.19 signalizuje přítomnost čísla 0110 na vstupu.

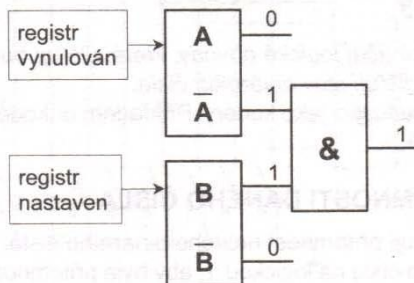


Obr. 3.19 Dekodér čísla 0110

Příklad č. 3: nastavení registrů

Obvod dle obr. 3.20 zjišťuje vynulování (nastavení výstupu na logickou 0) registru A, se současným nastavením (výstup = 1) registru B.

Poznámka: Pojem registru bude vysvětlen v kapitole č. 4 – Sekvenční logické obvody.



Obr. 3.20 Stav dvou registrů

B) ZJIŠTĚNÍ STAVU BINÁRNÍHO ČÍSLA

Jedná se o zjištění a identifikování všech možných stavů vstupních kombinací. Neboli dekodér převádí binární čísla na dekadické.

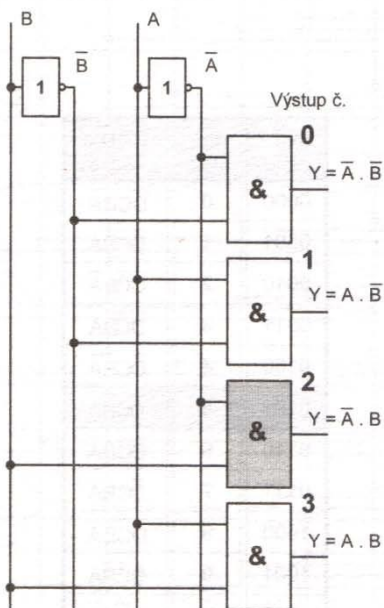
DEKODÉR DVOUBITOVÉHO ČÍSLA NA DESÍTKOVÉ – „JEDEN ZE ČTYŘ“

Dvoubitové číslo má $2^2 = 4$ kombinací, které můžeme očíslovat dle čtyř čísel (0 až 3) desítkové soustavy. Proto se dekodér nazývá též *jeden ze čtyř*.

Tabulka 3.3 Převod binárního čísla na desítkové

B	A	des. č.	Y
0	0	0	$\bar{B} \cdot \bar{A}$
0	1	1	$\bar{B} \cdot A$
1	0	2	$B \cdot \bar{A}$
1	1	3	$B \cdot A$

Opět použijeme převodovou tabulku mezi desítkovou a dvojkovou soustavou (tabulka 3.3). Pro každý řádek tabulky napíšeme funkci. Tuto funkci pro každý řádek, neboli desítkové číslo, zajistí pak příslušný člen AND výběrem (zapojením) vstupů z přímých a invertovaných vstupů (obr. 3.21).



Obr. 3.21 Dekodér jeden ze čtyř

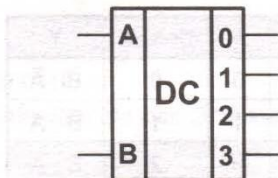
Činnost dekodéru

Podmínkou pro výstup logické 1 z příslušného členu AND je stav logické 1 na všech jeho vstupech.

Každý člen AND je zapojen dle funkce příslušného řádku převodní tabulky (*tabulka 3.3*) právě tak, aby pro tento a jen pro tento řádek byly všechny jeho vstupy rovny logické 1.

Například řádek pro desítkové číslo 2 (v *tabulce 3.3* označen šedě) má funkci logického součinu $A \cdot B$, a tudíž jen pro binární číslo $10_{(2)}$ ($A = 0, B = 1$) bude výstup členu AND č. 2 logická 1. Na výstupech všech ostatních členů AND bude logická 0.

Bloková značka dekodéru jeden ze čtyř je na *obr. 3.22*.



Obr. 3.22 Bloková značka dekodéru jeden ze čtyř

DEKODÉR ČTYŘBITOVÉHO ČÍSLA NA DESÍTKOVÉ – „JEDEN Z DESÍTI“

Tento dekodér BCD kódu se používá pro nejběžnější aplikaci převodu binárních čísel na desítková.

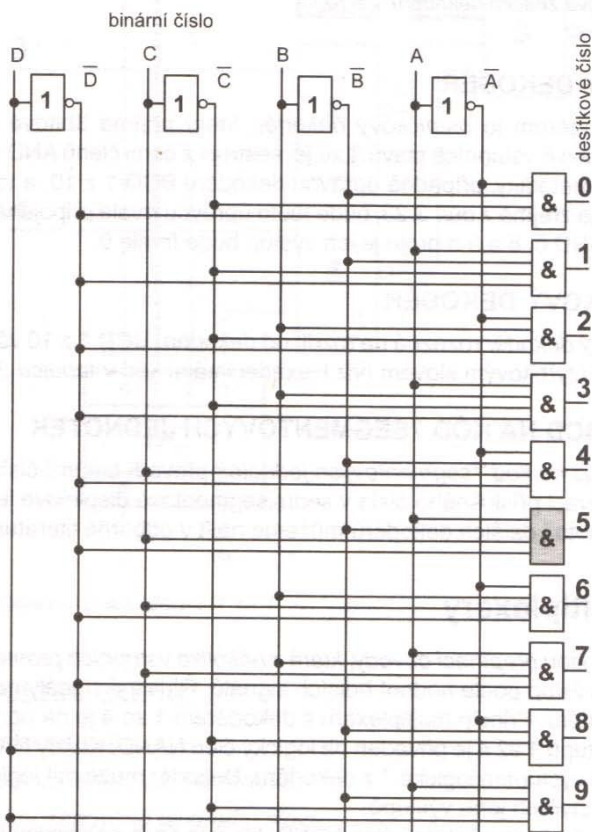
Tabulka 3.4 BCD kód

bin. číslo	des. číslo	funkce řádku
0000	0	$\overline{D}CBA$
0001	1	$\overline{D}CBA$
0010	2	$\overline{D}CBA$
0011	3	$\overline{D}CBA$
0100	4	$\overline{D}CBA$
0101	5	$\overline{D}CBA$
0110	6	$\overline{D}CBA$
0111	7	$\overline{D}CBA$
1000	8	$\overline{D}CBA$
1001	9	$\overline{D}CBA$
1010	10	$\overline{D}CBA$

Tabulka 3.4 BCD kód (pokračování)

bin. číslo	des. číslo	funkce řádku
1011	11	\overline{DCBA}
1100	12	$DC\overline{BA}$
1101	13	$DC\overline{B}\overline{A}$
1110	14	$DC\overline{B}A$
1111	15	$DCBA$
0000	16	\overline{DCBA}

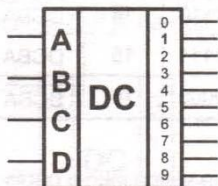
Činnost dekodéru na obr. 3.23 je stejná jako u popisovaného dekodéru jeden ze čtyř.



Obr. 3.23 Dekodér jeden z desíti

Jako příklad je šedě vyznačen v *tabulce 3.4* řádek pro desítkové číslo 5, který má funkci logického součinu $Y = \overline{D}CBA$, a tudíž jen pro binární číslo 0101₍₂₎ ($A = 1$, $B = 0$, $C = 1$, $D = 0$) budou všechny vstupy členu AND č. 5 v logické 1, a proto i jeho výstup bude logická 1 (na *obr. 3.23* je tento člen AND vyznačen šedě).

Od desítkových pozic 10 až 15 jsou řádkové funkce vždy jiné než u pozic 0 až 9. To znamená, že pro jakékoliv pozice 10 až 15 nebude splněna podmínka všech „jedničkových“ vstupů u členů AND 0 až 9 a proto nastaví na svých výstupech logickou 0.



Obr. 3.24 Bloková značka dekodéru 1 z 10

OSMIČKOVÝ DEKODÉR

Dalším dekodérem je *osmičkový dekodér*, který přijímá 3bitové slovo (CBA) a dekóduje všech 8 vstupních stavů. Lze jej sestavit z osmi členů AND a tří invertorů dle předchozí metodiky, případně úpravou dekodéru BCD 1 z 10, a to uzemněním vstupu D. Jak je zřejmé z *obr. 3.23*, bude touto úpravou trvale připojená logická 0 na vstupy členů AND č. 8 a 9 a proto jejich výstup bude trvale 0.

ŠESTNÁCTKOVÝ DEKODÉR

Šestnáctkový dekodér rozezná na rozdíl od dekodéru BCD 1 z 10 všech 16 stavů znázorněných čtyřbitovým slovem (viz Hexadecimální kód v *tabulce 3.4*).

DEKODÉR BCD NA KÓD 7SEGMENTOVÝCH JEDNOTEK

Dekodér BCD na kód 7segmentových jednotek převádí binární číslo na stav potřebný pro aktivaci příslušného čísla v sedmsegmentové displejové jednotce.

Zapojení tohoto i dalších dekodérů můžeme najít v odborné literatuře [2], [11].

3.5 Multiplexery

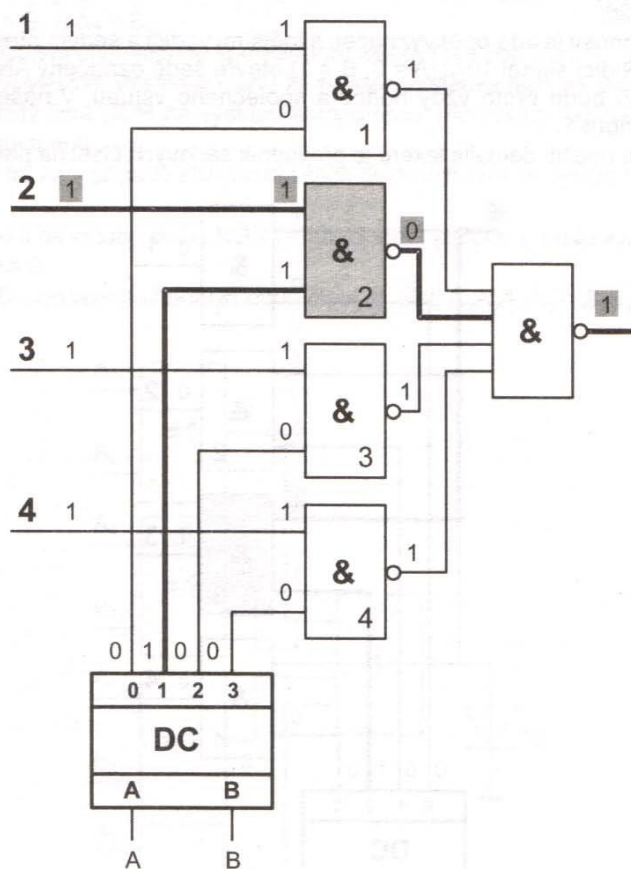
Multiplexery jsou přepínací obvody, které z několika vstupních proměnných přepínají na jeden výstup podle hodnot řídicích signálů. Řídicími signály může být např. výstup z dekodéru. Princip multiplexeru s dekodérem 1 ze 4 je na obr. 3.25.

Každý ze vstupů 1 až 4 je přiveden na logický člen NAND. Každý NAND může být otevřen pouze výstupem logické 1 z dekodéru. Dekodér může mít logickou 1 pouze na jednom ze čtyřech jeho výstupů.

Tento výstup 1 otevře příslušný člen NAND, který je tímto průchodný pro svůj druhý vstup (tj. vstup 1, nebo 2, nebo 3, nebo 4). Všechny ostatní členy NAND mají na výstupu

trvale logickou 1, neboť je z dekodéru přivedena logická 0 na jejich vstup. Tím jsou všechny vstupy multiplexeru, mimo vstup vedený do otevřeného členu NAND, blokovány.

Příslušný signál je otevřeným průchodem multiplexerem 2× invertován. V obr. 3.25 je silnější čarou vyznačen vstup otevřený řídicím binárním číslem BA = 01. Průchozí NAND je označen šedě.



Obr. 3.25 Multiplexer s dekodérem 1 ze 4

POUŽITÍ MULTIPLEXERŮ

- přepínání signálů,
- převod paralelních dat na sériová,
- generátor sériového binárního slova,
- vytváření Booleovských funkcí.

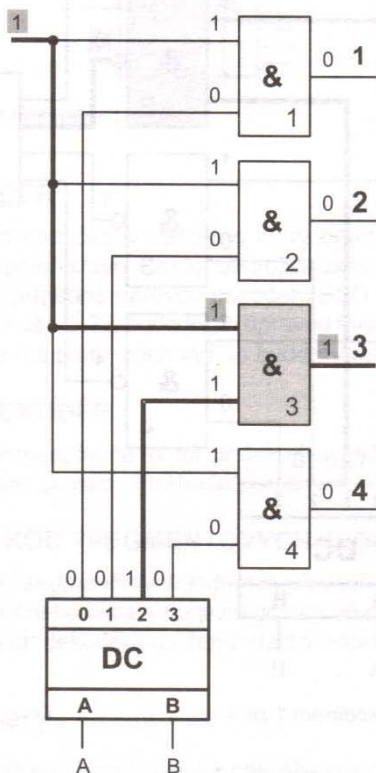
3.6 Demultiplexery

Demultiplexer je přepínací obvod, který jeden vstup přepíná na několik výstupů dle řídicího signálu (z dekodéru).

Je to v zásadě obrácený multiplexer. Můžeme jej též považovat za určitý druh dekodéru s jedním vstupem a několika výstupy. Demultiplexer s dekodérem 1 ze 4 je na obr. 3.26.

Příklad činnosti je zde opět vyznačen silnějšími vodiči a šedým otevřeným členem AND. Řídicí signál $10_{(2)}$ ($A = 0, B = 1$) otevře šedě označený AND č. 3 a na jeho výstupu bude proto vždy hodnota společného vstupu, v našem příkladu logická hodnota 1.

Nejčastější použití demultiplexerů je převodník sériových čísel na paralelní.



Obr. 3.26 Demultiplexer s dekodérem 1 ze 4

3.7 Binární komparátor

Binární komparátor je kombinační logický obvod, který porovnává dvě paralelní binární slova a generuje hodnotu 1, jestliže jsou obě slova rovna.

Je-li první binární slovo $A_3A_2A_1A_0$ a druhé binární slovo $B_3B_2B_1B_0$, musí platit pro jejich rovnost ($A_3A_2A_1A_0 = B_3B_2B_1B_0$) shodnost čísel příslušných řádů, tj. $A_0 = B_0$, $A_1 = B_1$, $A_2 = B_2$ a $A_3 = B_3$.

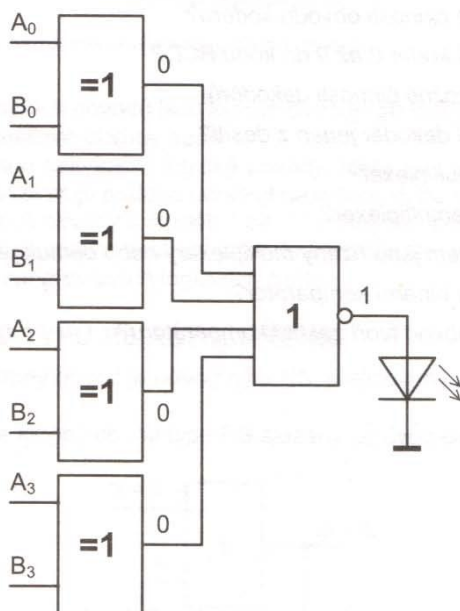
Příklad zapojení čtyřbitového komparátoru znázorňuje obr. 3.27.

Komparátor využívá vlastnosti členu XOR, který generuje nulu při shodnosti svých vstupů. Použitý člen NOR na výstupu komparátoru má výstup logické 1, má-li na všech vstupech nuly.

Znamená to, že v případě shodnosti všech binárních řádů je výstup komparátoru logická 1.

Naopak, je-li na vstupu členu NOR jedna (či více) logická 1, bude výstup komparátoru logická 0.

Dioda LED signalizuje shodnost obou čísel, tj. platí $A_3A_2A_1A_0 = B_3B_2B_1B_0$.



Obr. 3.27 Komparátor dvou čtyřbitových čísel

Více o kombinačních logických obvodech a jejich aplikacích nalezne čtenář v publikaci [11].

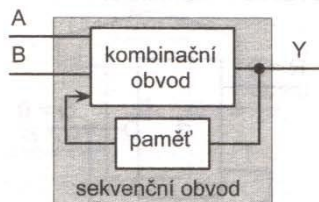
1. Vyjádřete slovně vlastnosti obvodu XOR!
2. Napište matematickou funkci logického obvodu XOR!
3. Z jakých členů se skládá poloviční sčítačka?
4. Co umí poloviční sčítačka?
5. Co neumí poloviční sčítačka?
6. Z jakých základních členů se skládá úplná sčítačka?
7. V jakém členu sčítačky může vzniknout bit přenosu do vyššího řádu?
8. Vysvětlete činnost generátoru parity!
9. K čemu se používá generátor parity?
10. Co jsou to kodéry?
11. Co vykonává kodér čísel 0 až 3?
12. Jaký je princip činnosti obvodu kodéru?
13. K čemu slouží kodér 0 až 9 do kódu BCD?
14. Uvedte dvě možné činnosti dekodérů!
15. K čemu slouží dekodér jeden z desíti?
16. Jak pracuje multiplexer?
17. Jak pracuje demultiplexer?
18. Kterým obvodem jsou řízeny multiplexery nebo demultiplexery?
19. K čemu slouží binární komparátor?
20. Jaký logický obvod tvoří základ komparátoru?

4 SEKVENČNÍ LOGICKÉ OBVODY

4.1 Princip, klopné obvody RS

Sekvenční logické obvody mají tu vlastnost, že jejich výstupní proměnné jsou určeny nejen kombinací hodnot vstupních proměnných v daném okamžiku, ale i minulými hodnotami některých proměnných.

Z toho vyplývá, že si sekvenční obvod musí pamatovat hodnoty z předcházejícího stavu, což znamená, že musí obsahovat paměťové členy. Sekvenční obvod se skládá z kombinační části a paměťové části (obr. 4.1).



Obr. 4.1 Blokové schéma sekvenčního logického obvodu

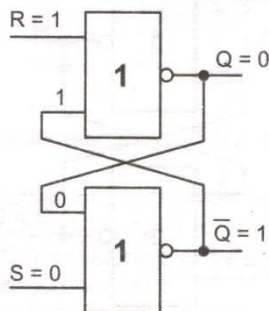
Základem sekvenčních obvodů jsou *klopné obvody*, ze kterých se konstruují dále čítače, registry, paměťové obvody apod.

Klopné obvody jsou sekvenční logické obvody, které se používají v číslicových počítačích, kde se vyskytuje potřeba uchovat na určitou dobu signál s logickou hodnotou 0 nebo 1 (neboli obvody s pamětí 1 bit). V principu odpovídají tranzistorovým bistabilním klopným obvodům, probíraným v druhém dílu této učebnice. Výhodnější je však realizace z integrovaných logických členů.

KLOPNÝ OBVOD TYPU RS ASYNCHRONNÍ

Nejjednodušší klopný obvod je *obvod typu RS*, realizovaný ze dvou členů NAND nebo NOR.

Proberme nejprve klopný obvod typu RS složený ze dvou členů NOR (obr. 4.2).

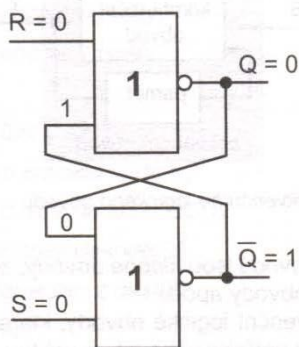


Obr. 4.2 Klopný obvod typu RS – stav „RESET“

Uvažujeme nejprve stav vstupů $R = 1$ a $S = 0$. Z pravdivostní tabulky členu NOR víme, že má na výstupu logickou 1 jen v případě obou vstupů v logické 0. Tato podmínka není splněna u horního členu NOR, neboť vstup je $R = 1$, a proto musí být jeho výstup $Q = 0$. Tato logická 0 se dostane zpětnou vazbou na vstup dolního členu NOR a ten vytvoří na svém výstupu $\bar{Q} = 1$. Zpětná vazba na horní člen NOR jeho výstup nezmění. Tím je jednoznačně dán výstup klopného obvodu $Q = 0$ a $\bar{Q} = 1$.

Tomuto stavu říkáme RESET = vynulování, protože hlavní výstup Q je v logické 0.

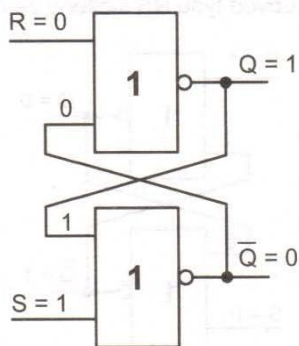
Změníme nyní vstup horního členu NOR z $R = 1$ na $R = 0$ (obr. 4.3). Vlivem zpětné vazby z dolního členu zůstane jeho druhý vstup 1, a jeho výstup bude proto $Q = 0$. Výstup dolního členu bude ze vstupů 0 a 0 $\bar{Q} = 1$. Vidíme, že se oba výstupy po vynulování vstupů ze stavu RESET nezměnily.



Obr. 4.3 Stav RESET po vynulování vstupů

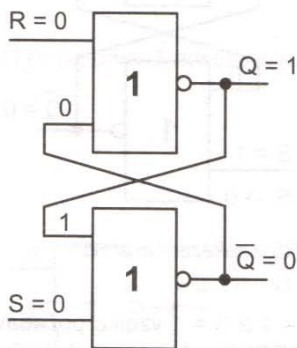
Při opačné situaci oproti stavu RESET, tzn. pro $R = 0$ a $S = 1$, bude i výstup opačný tj. $Q = 1$ a $\bar{Q} = 0$, jak je patrné z obr. 4.4.

Tomuto stavu říkáme SET = nastavení, protože obvod má hlavní výstup $Q = 1$.



Obr. 4.4 Klopný obvod typu RS – stav „SET“

Při změně $S = 1$ na $S = 0$ se opět výstupy $Q = 1$ a $\bar{Q} = 0$ nezmění (obr. 4.5). Nyní jsou tyto výstupy pro $S = 0$ a $R = 0$ právě opačné oproti vynulování obvodu ze stavu RESET (srovnej obr. 4.5 s obr. 4.3).



Obr. 4.5 Stav SET po vynulování vstupů

Přechod z obou stavů RESET a SET do stavu vynulování vstupů vyjadřuje tabulka 4.1. V tabulce je podchyceno, že výstupy obvodu po vynulování vstupů ($R = S = 0$) jsou možné oba: $Q = 0$ a $\bar{Q} = 1$ nebo $Q = 1$ a $\bar{Q} = 0$.

Tabulka 4.1 Dva stavy klopného obvodu typu RS po vynulování vstupů

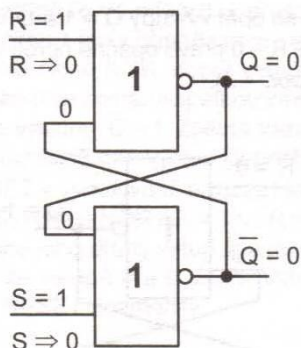
	R	S	Q	\bar{Q}
RESET	1	0	0	1
SET	0	1	1	0
	0	0	1	0
			0	1

Poněkud komplikovaná situace nastane, je-li $R = 1$ a zároveň $S = 1$ (tzn. SET a RESET najednou?!).

V tom případě jsou dle pravdivostní tabulky NOR oba výstupy Q i \bar{Q} v logické 0. To by tak nevadilo, avšak při změně $R = S = 1$ na $R = S = 0$ nastane labilní situace ukazovaná na obr. 4.6, kdy všechny vstupy jsou teď v logické 0 a oba členy NOR se snaží vytvořit na výstupu logickou hodnotu 1.

Ale kde? To není jednoznačně dáno a obvod přejde do některého stavu náhodně.

Tento stav ($R = S = 1$) je proto nepřipustný a označujeme jej jako zakázaný stav.



Obr. 4.6 Klopný obvod typu RS v zakázaném stavu

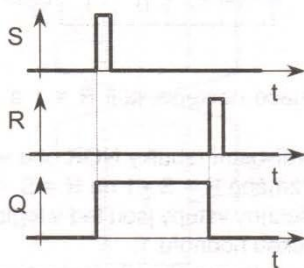
Při současném vstupu $R = 1$ a $S = 1$ vzniká požadavek na současné nastavení i vynulování, neboli SET a RESET najednou, což je protichůdné.

Shrneme výsledky našich úvah do celkové pravdivostní tabulky klopného obvodu typu RS – viz *tabulku 4.2*.

Tabulka 4.2 Celková pravdivostní tabulka klopného obvodu

R	S	Q	\bar{Q}	
0	0	0/1	1/0	
0	1	1	0	SET
1	0	0	1	RESET
1	1	X	X	zakázaný stav

Tabulku lze vyjádřit i graficky v časovém diagramu dle *obr. 4.7*. Z diagramu vidíme, že impuls S ($S = 1$) obvod *nastaví*, neboli vytvoří $Q = 1$ a impuls R ($R = 1$) obvod *vynuluje* na $Q = 0$.



Obr. 4.7 Činnost klopného obvodu RS – časový diagram

Funkci klopného obvodu lze vyjádřit i matematicky. V případě našeho klopného obvodu z členů NOR (obr. 4.8) platí pro výstup horního členu

$$Q = R + \bar{Q} \quad (1)$$

Pro výstup dolního členu platí

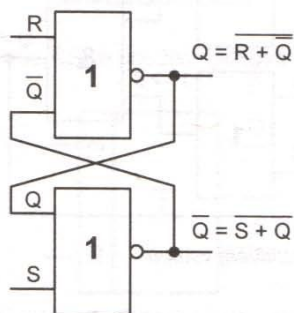
$$\bar{Q} = S + Q \quad (2)$$

Po dosazení výrazu (2) do (1) a po úpravě dostaneme

$$Q = \overline{R + S + Q}$$

$$Q = \bar{R} \cdot \overline{(S + Q)}$$

$$Q = \bar{R} \cdot (S + Q)$$



Obr. 4.8 Funkce klopného obvodu RS

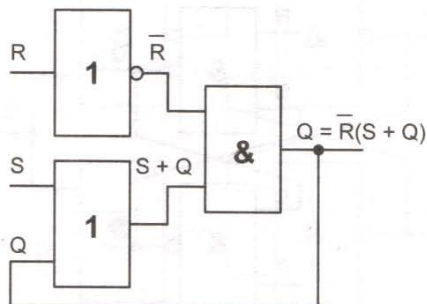
Snadno se přesvědčíme, že odvozená funkce platí pro všechny řádky pravdivostní tabulky:

$$R = 0, S = 0 \Rightarrow Q = 1(0 + Q) = Q$$

$$R = 0, S = 1 \Rightarrow Q = 1(1 + Q) = 1$$

$$R = 1, S = 0 \Rightarrow Q = 0(0 + Q) = 0$$

$$R = 1, S = 1 \Rightarrow Q = 0(1 + Q) = 0$$



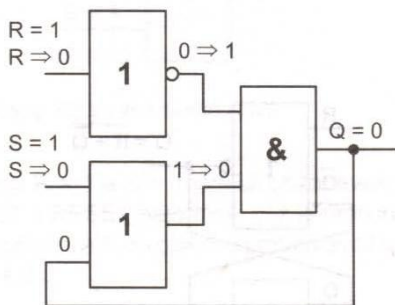
Obr. 4.9 Klopný obvod typu RS jinak

V poslední řádce se ale jedná se o zakázaný stav, neboť po přechodu do první řádky může Q být logická 0 i 1.

Z vypočteného výrazu $Q = \bar{R}(S + Q)$ plyne i jiná možnost realizace klopného obvodu (obr. 4.9).

Zde si znázorníme situaci zakázaného stavu, který u obvodu s dvěma členy NOR představoval vstup $R = S = 1$ (obr. 4.10).

Nula z výstupu klopného obvodu je přiváděna na jeden vstup členu OR. Při změně z $S = R = 1$ na $S = R = 0$ vznikne na výstupu OR nula, která zajistí zachování nuly i na výstupu celého klopného obvodu. Z této úvahy je zřejmé, že tento obvod nemá zakázaný stav.



Obr. 4.10 Klopný obvod po vynulování vstupů

Klopný obvod lze též vytvořit z členů NAND. Zapojení tohoto obvodu dle obr. 4.11 můžeme opět matematicky vyjádřit.

Pro výstup horního členu platí

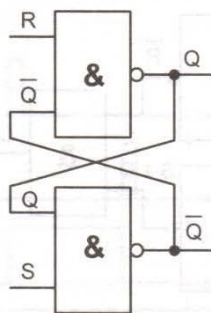
$$Q = \overline{R \cdot \bar{Q}} \quad (1)$$

Pro výstup dolního členu platí

$$\bar{Q} = \overline{S \cdot Q} \quad (2)$$

Po dosazení výrazu (2) do (1) dostaneme

$$Q = \overline{R \cdot \overline{S \cdot Q}} = \overline{R} + (S \cdot Q) = \overline{R} + S \cdot Q$$



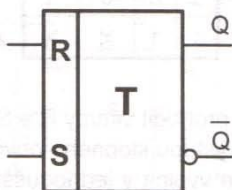
Obr. 4.11 Klopný obvod typu RS z hradel NAND

Výsledný výraz je odlišný od funkce klopného obvodu se členy NOR, avšak činnost obou obvodů je stejná. Odlišnost je pouze u zakázaného stavu, který nastane nyní při hodnotách vstupů $R = S = 0$ (viz tabulku 4.3).

Tabulka 4.3 Stavy klopného obvodu typu RS

R	S	Q	\bar{Q}	
0	0	X	X	zakázaný stav
0	1	1	0	SET
1	0	0	1	RESET
1	1	0/1	1/0	

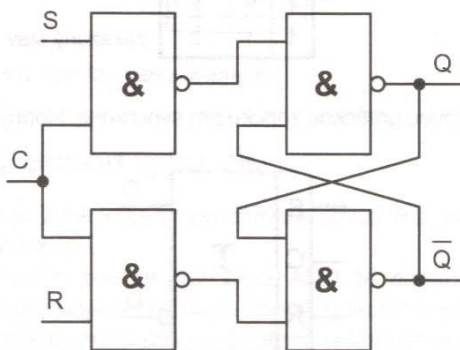
Pro kreslení klopného obvodu RS se používá bloková grafická značka dle obr. 4.12.



Obr. 4.12 Grafická značka klopného obvodu typu RS

KLOPNÝ OBVOD TYPU RS SYNCHRONNÍ

Doplňme-li před popisovaný klopný obvod RS další dva členy NAND (obr. 4.13), dostaneme obvod se stejnými vlastnostmi, ale jeho výstup bude možné měnit jen během synchronizačního hodinového impulsu C ($C = \text{clock}$, čti „klok“, z angl. – znamená hodiny).



Obr. 4.13 Synchronní klopný obvod typu RS

Chybí-li hodinový impulz ($C = 0$), je obvod pomocí prvních členů NAND *zablokovan*, neboť jejich výstup je trvale roven 1 a jakákoliv změna R nebo S nemůže změnit výstup Q.

Obvod se otevře teprve s příchodem hodinového impulzu ($C = 1$) a jedině v době jeho trvání je možné pomocí vstupů R a S výstup Q měnit.

Vstupní členy NAND zde působí jako inventory a proto bude zakázaný stav opačný, tj. pro hodnoty $S = R = 1$ (*tabulka 4.4*), jako u klopného obvodu asynchronního se členy NOR (*tabulka 4.2*).

Tabulka 4.4 Pravidlostní tabulka synchronního klopného obvodu typu RS

R	S	Q	\bar{Q}
0	0	0/1	1/0
0	1	1	0
1	0	0	1
1	1	X	X

zakázaný stav

Ze stejného důvodu je nutné prohodit vstupy R a S, aby platilo pro stavy RESET a SET odpovídající nastavení výstupu klopného obvodu.

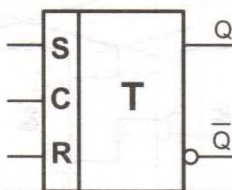
Pravidlostní tabulku můžeme vyplnit v jednodušším tvaru (*tabulka 4.5*), kdy ve třetím sloupci zapisujeme hodnotu výstupu po odeznění hodinového impulzu (Q_{n+1}).

Tabulka 4.5 Jednodušší úprava pravidlostní tabulky obvodu RS

R	S	Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	X

zakázaný stav

Opět lze použít blokovou grafickou značku pro synchronní klopný obvod dle *obr. 4.14*.

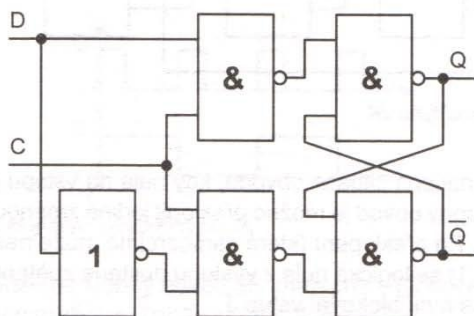


Obr. 4.14 Grafická značka klopného obvodu typu RS synchronního

4.2 Klopný obvod typu D

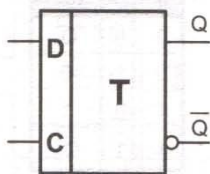
Obvod odstraňuje zakázaný stav propojením obou vstupů klopného obvodu typu RS invertorem. Obvod je řízen (synchronizován) hodinovým vstupem (obr. 4.15).

Klopný obvod překlápí jen během hodinového impulsu, jinak je zavřený. Invertor zajišťuje, že vstupy do původního klopného obvodu RS mohou být pouze rozdílné (je-li $R = 1$, pak $S = 0$ a naopak), neboli obvod pracuje pouze dle druhého a třetího řádku pravdivostní tabulky klopného obvodu RS. Touto úpravou má obvod pouze jeden vstup. Výstup klopného obvodu typu D kopíruje vstup a uchovává poslední hodnotu výstupu až do další změny vstupu. Obvod má paměť o velikosti jednoho bitu a využívá se proto v paměťových registrech.



Obr. 4.15 Klopný obvod typu D

Grafická bloková značka klopného obvodu typu D je na obr. 4.16.



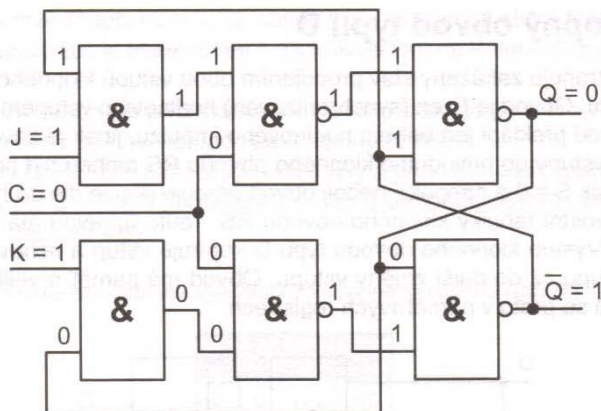
Obr. 4.16 Klopný obvod typu D – bloková značka

4.3 Klopný obvod typu JK

Klopný obvod JK je zdokonalený synchronní obvod RS, který nemá zakázaný stav vstupních proměnných.

To je docíleno přidáním součinných členů AND do vstupů S a R a zavedením zpětné vazby z výstupu klopného obvodu na vstup těchto členů AND (obr. 4.17).

Zabránění vzniku zakázaného stavu je docíleno zajištěním jednoho nulového zpětnovazebního vstupu (je-li $Q = 0$) pro dolní součinný člen AND nebo druhého nulového zpětnovazebního vstupu (je-li $Q = 0$) pro horní součinný člen AND.



Obr. 4.17 Klopný obvod typu JK

V obr. 4.17 je vyznačena situace obvodu, kdy nula na vstupu dolního členu AND blokuje vstup K a klopný obvod je možno překlopit jedině změnou vstupu J z nuly na logickou hodnotu 1. Po překlopení (které samozřejmě může nastat pouze při hodinovém impulzu $C = 1$) se logická nula z výstupu dostane zpětnou vazbou na vrchní člen AND a ten bude nyní blokovat vstup J.

Jednotlivé stavy klopného obvodu JK znázorňuje pravdivostní tabulka 4.6.

Tabulka 4.6 Stavy klopného obvodu JK

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	\overline{Q}_n

Sloupec Q_{n+1} znázorňuje stav výstupu obvodu po průchodu hodinového impulzu.

Q_n znamená, že se výstup po průchodu impulzů nemění, \overline{Q}_n znamená překlápění výstupu s náběžnou hranou hodinového impulzu C při trvalé podmínce vstupů $J = K = 1$.

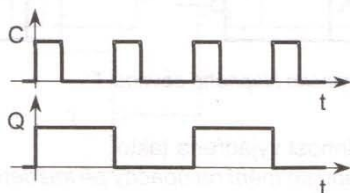
Všimněme si podrobněji této čtvrté řádky tabulky (tabulka 4.6), která u obvodu RS znamenala zakázaný stav.

Hodnoty proměnných na obr. 4.17 odpovídají momentální situaci po vynulování (resetování) obvodu a před příchodem hodinového impulzu (tj. ještě pro $C = 0$). Vzhledem k tomu, že dolní vstupní člen NAND (člen dole uprostřed) je blokován logickou 0 z výstupu dolního členu AND, zůstane jeho výstup i po příchodu hodinového impulzu ($C = 1$) na hodnotě logická 1.

Příchod hodinového impulsu však překlopí horní vstupní NAND na hodnotu logické 0. Tato změna zapříčiní překlopení výstupu klopného obvodu na $Q = 1$ a následnou změnu výstupu spodního členu NAND na $\bar{Q} = 0$. Zpětnou vazbou se nula přenesete teď na horní člen AND a ten zablokuje vliv vstupu J. Po skončení hodinového impulsu ($C = 0$) se zablokují oba vstupní členy NAND a výstup Q se proto nezmění. Teprve další hodinový impuls umožní nyní již dolnímu členu NAND, aby změnil výstup Q na hodnotu 1, a tudíž Q opět na hodnotu 0.

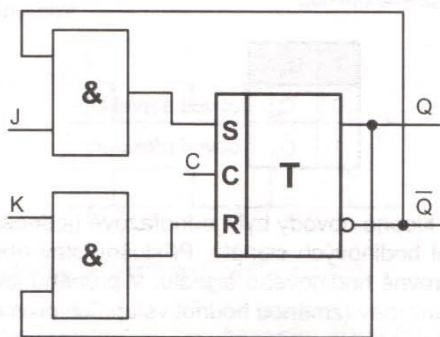
Díky této úpravě nemá obvod zakázaný stav a je možno jej provozovat při trvalé hodnotě logické 1 na obou vstupech J a K.

Grafické vyjádření výše popsané činnosti obvodu při trvalé hodnotě vstupů $J = K = 1$ a s periodickým hodinovým signálem znázorňuje obr. 4.18.

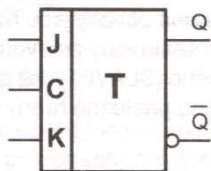


Obr. 4.18 Přepínání klopného obvodu typu JK

Zjednodušeně můžeme klopný obvod JK znázornit v blokovém tvaru (obr. 4.19), nebo přímo blokovou grafickou značkou (obr. 4.20).



Obr. 4.19 Klopný obvod typu JK v blokovém tvaru

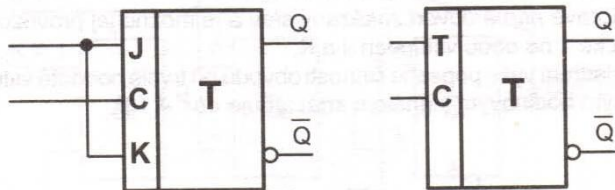


Obr. 4.20 Klopný obvod typu JK – bloková značka

4.4 Klopný obvod typu T

Klopný obvod tohoto typu získáme propojením obou vstupů J a K klopného obvodu typu JK. Vznikne jediný vstup, označený T (obr. 4.21).

Při tomto zapojení (je-li trvale $T = 1$) pracuje obvod tak, jak bylo v předchozím výkladu vysvětleno a znázorněno pro hodnoty $J = K = 1$ (obr. 4.18).



Obr. 4.21 Vznik a bloková značka klopného obvodu T

Slovně může být tato činnost vyjádřena takto:

Je-li $T = 1$, stav na výstupu se mění na opačný při každém příchodu synchronizačního (hodinového) impulsu C.

Je-li $T = 0$, obvod setrvává v původním stavu.

Této vlastnosti se využívá v čítačích (viz dále).

Obě možnosti zachycuje tabulka 4.7.

Tabulka 4.7 Stavy klopného obvodu T

T	Q_{n+1}	
0	Q_n	obvod nepřeklápí
1	\bar{Q}_n	obvod překlápí

Dosud popisované klopné obvody byly jednofázové (jednostupňové). Tyto obvody jsou řízeny úrovní hodinových signálů. Příslušný stav obvodu je zabezpečen v době jednotkové úrovně hodinového signálu. V průběhu jeho trvání může však obvod několikrát změnit stav (změnou hodnot vstupních proměnných).

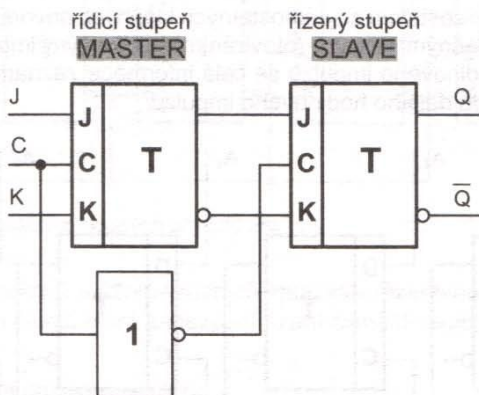
4.5 Klopný obvod JK dvoufázový

Dvoufázové (dvoustupňové) klopné obvody jsou řízeny hranou hodinových signálů (dynamické řízení). Obvody jsou sestaveny ze dvou klopných jednostupňových obvodů: řídicího (MASTER) a řízeného (SLAVE) – viz obr. 4.22. Změna stavu klopného obvodu může nastat jen v průběhu příslušné hrany hodinového signálu.

Činnost obvodu

V první fázi činnosti obvodu se informace přesune ze vstupu do řídicího stupně, zatímco druhý stupeň zachovává nezměněný stav (je blokován invertorem na jeho

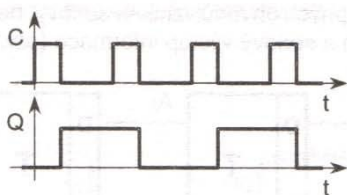
vstupu). Ve druhé fázi se informace přesune z řídicího stupně do řízeného s možností způsobit změnu stavu výstupní proměnné Q. Druhá fáze probíhá v době, kdy je řízený stupeň izolovaný od vstupní části nepřítomností hodinového signálu C v řídicím stupni.



Obr. 4.22 Klopný obvod JK dvoufázový

Informace přechází z MASTER na SLAVE při skončení hodinového impulsu (v jeho týlové hraně – obr. 4.23).

Je-li $J = K = 1$, tak každá týlová hrana hodinového impulsu obvod překlápí. Je-li $J = K = 0$ obvod nepřeklápí.



Obr. 4.23 Překlápění obvodu JK týlem hodinových impulsů

Obvod touto úpravou odstraňuje tzv. *hazardní stavy* (náhodně vzniklé), neboť překlopení obvodu je možné pouze během krátké doby trvání týlové hrany hodinového impulsu. Případné vnější poruchy během hodinového impulsu výstup klopného obvodu nemohou ovlivnit.

4.6 Paměťové registry

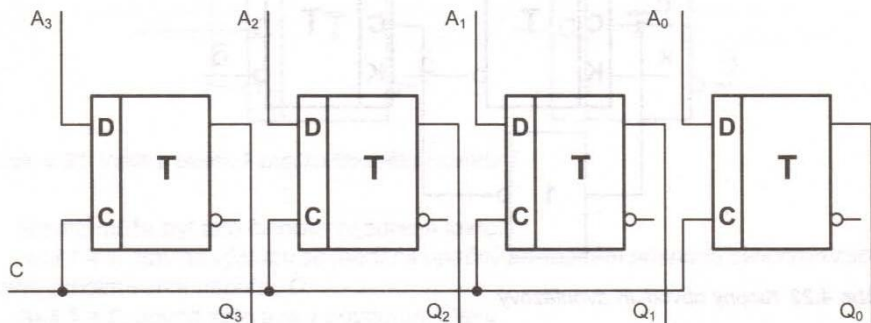
Registry jsou sekvenční logické obvody, které umožňují vložení a uchování informace.

Registr sestojíme vhodným propojením několika klopných obvodů. Jejich počet určuje délku registru a současně počet bitů dvojkové informace, která má být zaznamenána registrem.

Způsob propojení klopných obvodů určuje funkci, kterou registr provádí se skupinou bitů.

PARALELNÍ REGISTR

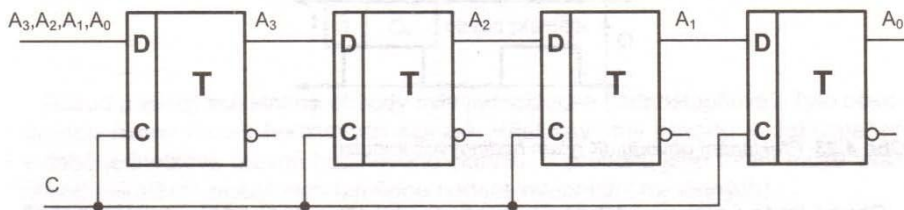
Paralelní registr je sestaven ze samostatných klopných obvodů typu D, které jsou vázány pouze společným řízením (otevíráním) hodinovým impulzem (obr. 4.24). Během jednoho hodinového impulzu se celá informace zaznamená do celého registru až do příchodu dalšího hodinového impulzu.



Obr. 4.24 Paralelní registr

SÉRIOVÝ REGISTR

Sériovým zapojením klopných obvodů vznikne *sériový*, neboli *posuvný registr*, který umožňuje sériový záznam a sériový výstup informace (obr. 4.25).

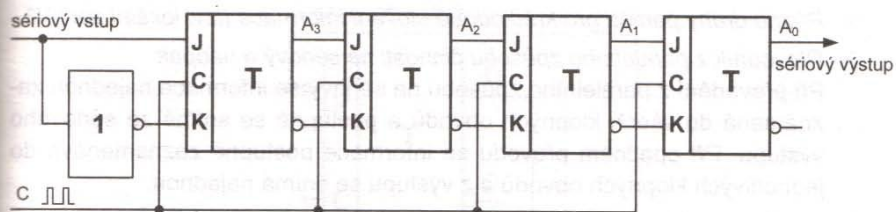


Obr. 4.25 Sériový – posuvný registr

Informace přivedená na vstup prvního klopného obvodu se příchodem hodinového impulzu přenesou na jeho výstup, který je spojen se vstupem dalšího klopného obvodu. U čtyřbitového registru se celá informace (tj. čtyřbitové číslo) zaznamená čtyřmi hodinovými impulzy, a to postupným posouváním celého obsahu registru o jeden stupeň vpravo. Přerušíme-li po tomto záznamu hodinové impulzy, zůstane informace zachována v registru. Informaci můžeme nyní paralelně neboli najednou vybrat. Při pokračování hodinových impulzů můžeme informaci sériově, neboli postupně, snímat z výstupu posledního členu.

POSUVNÝ REGISTR Z KLOPNÝCH OBVODŮ TYPU JK

Stejným způsobem pracuje i registr sestavený z klopných obvodů typu JK (obr. 4.26).



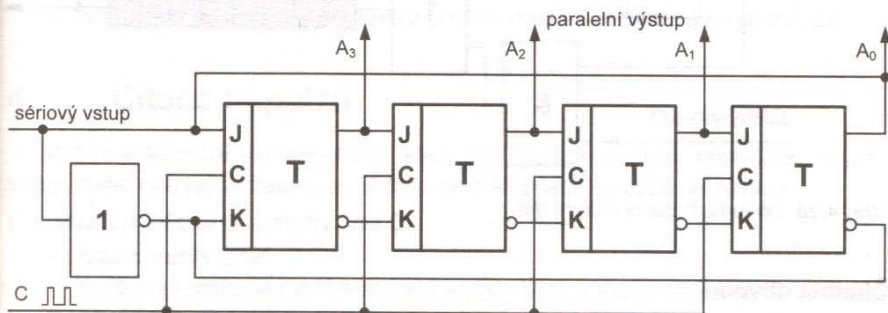
Obr. 4.26 Posuvný registr z klopných obvodů JK

Jednotlivé typy registrů můžeme různým způsobem kombinovat pomocí přidavných kombinačních členů, které zabezpečí řízení činnosti registrů.

Jsou to:

- *Jednosměrné posuvné registry.*
Posun informace je pouze doleva nebo pouze doprava.
- *Obousměrné neboli vratné posuvné registry.*
Směr posunu informace je možné volit.
- *Posuvné registry se sériovým vstupem a paralelním výstupem.*
Je možná i opačná varianta.
- *Kruhové registry.*
Výstup posledního klopného obvodu je spojen se vstupem prvního klopného obvodu – obr. 4.27.

KRUHOVÝ REGISTR



Obr. 4.27 Kruhový registr

V kruhovém registru, velmi často používaném, data rotují. Výstup z libovolného klopného obvodu na obr. 4.27 má kmitočet 1/4 hodinového kmitočtu, protože obsa-

huje čtyři členy JK. Obecně platí, že kruhový registr dělí vstupní kmitočet počtem použitých klopných obvodů, a lze jej proto použít jako dělič kmitočtu.

Použití registrů:

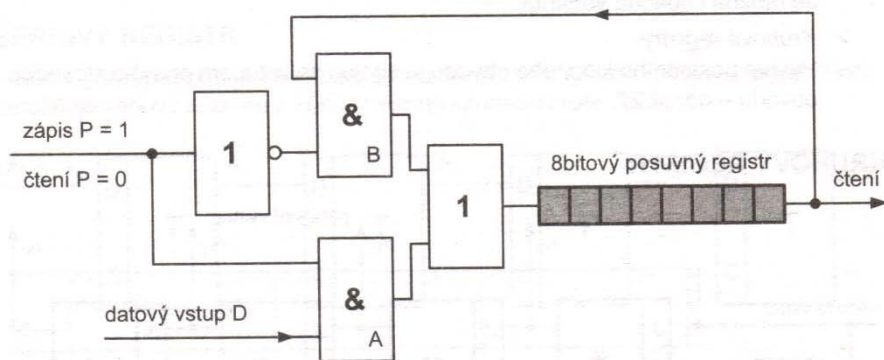
- *Různé druhy pamětí pro krátkodobé uložení informace (tzv. lokální paměť).*
- *Převodník z paralelního způsobu činnosti na sériový a naopak.*
Při převádění z paralelního způsobu na sériový se informace najednou zaznamená do všech klopných obvodů a postupně se snímá ze sériového výstupu. Při opačném převodu se informace postupně zaznamenává do jednotlivých klopných obvodů a z výstupu se snímá najednou.
- *Zpoždovací členy.*
Výstup posledního klopného obvodu je proti vstupu prvního klopného obvodu zpožděn o dobu, která je dána součinem periody hodinových impulzů a počtem klopných obvodů v registru.

PŘÍKLADY NA APLIKACI REGISTRŮ

příklad 1

PAMĚŤ POSUVNÉHO REGISTRU

Obvod na obr. 4.28 zajišťuje režim zápisu do registru nebo čtení dat z registru.



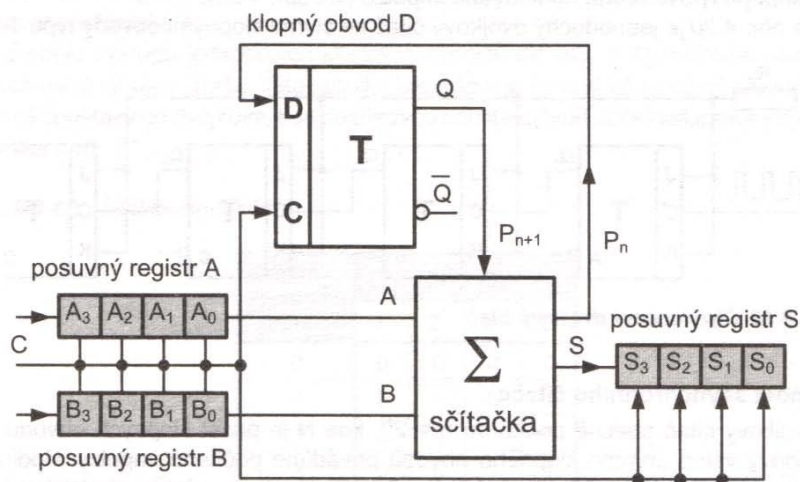
Obr. 4.28 Obvod pro zápis a čtení dat

Činnost obvodu

Zápis dat: $P = 1 \Rightarrow \text{AND}_B$ je zavřen, AND_A je otevřen a je možno posílat data D do registru.

Čtení dat: $P = 0 \Rightarrow \text{AND}_A$ je zavřen, AND_B je otevřen a je možno z registru data vysouvat. Data zároveň cyklují zpět na vstup do registru. Vstup dat je blokován členem AND_B .

příklad 2 SÉRIOVÁ SČÍTAČKA



Obr. 4.29 Sériová sčítačka

Činnost obvodu

1. Čísla jsou zapsána do registrů A a B.
2. Prvý impuls C přivede A_0 a B_0 na sčítačku, ta je sečte, součet pošle do registru S a přenos uloží do klopného obvodu D, který pracuje jako paměť.
3. Další impuls C sečte A_1 a B_1 a přičte případný přenos P_0 a uloží nový přenos P_1 do D.
4. Současně se součet přesouvá z prvního místa v registru S na druhé atd.

4.7 Čítače impulsů

Čítač je sekvenční logický obvod, který čítá (počítá) impulzy přivedené na jeho vstup, nebo dělí jejich frekvenci. Skládá se z klopných obvodů JK nebo T.

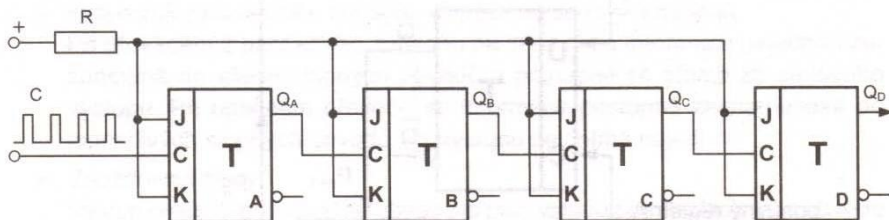
Dle principu existují dva druhy čítačů:

- *Asynchronní čítač.*
Výstup každého klopného obvodu je přiveden na vstup následujícího. Překlápění klopného obvodu se uskutečňuje postupně s každým hodinovým impulzem, což při více klopných obvodech přináší nevýhodné zpoždění.
- *Synchronní čítač.*
Čítač překlápí všechny klopné obvody současně a je řízen hodinovými synchronizačními impulzy.

ASYNCHRONNÍ ČÍTAČ

Asynchronní čítač využívá vlastností dvoustupňového klopného obvodu JK, který překlápí při tylové hraně hodinového impulsu (viz obr. 4.23).

Na obr. 4.30 je jednoduchý dvojkový čítač se čtyřmi klopnými obvody typu JK.



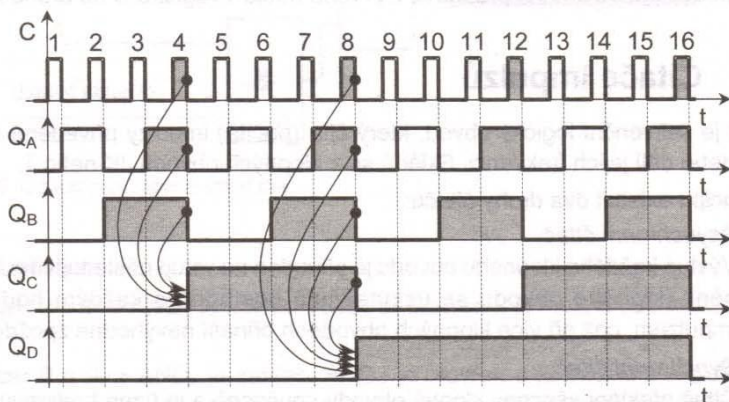
Obr. 4.30 Čtyřstupňový dvojkový čítač

Činnost asynchronního čítače

Dvojkový čítač obecně počítá do $n = 2^N$, kde N je počet klopných obvodů. Na hodinový vstup prvního klopného obvodu přivádíme počítaný impuls. Hodinové vstupy ostatních klopných obvodů jsou připojeny vždy na výstup předcházejícího klopného obvodu.

Klopný obvod T_A mění svůj stav při každém hodinovém impulsu, tzn. že kmitočet vstupních impulsů dělí dvěma. Klopný obvod T_B mění svůj stav po každém druhém vstupním hodinovém impulsu a vstupní kmitočet dělí čtyřmi atd. Obecně platí, že na výstupu N -tého klopného obvodu získáme kmitočet $f/2^N$. Při šestnáctém hodinovém impulsu nastane takový stav, že v průběhu trvání tohoto impulsu jsou všechny klopné obvody nastaveny na úroveň logické hodnoty 1 ($Q_A = Q_B = Q_C = Q_D = 1$).

Ve dvojkovém kódu to představuje desítkové číslo 15 ($1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0$). Tímto šestnáctým impulsu způsobí postupné překlopení všech klopných obvodů do výchozího stavu ($Q_A = Q_B = Q_C = Q_D = 0$).



Obr. 4.31 Časový diagram dvojkového čítače

Činnost čítače názorně vyjadřuje grafický záznam průběhů výstupů všech klopných obvodů (obr. 4.31). Na diagramu je vyznačena šedě podmínka pro překlopení následujícího klopného obvodu – výstupy všech předchozích klopných obvodů musí být ve stavu logické 1.

Dle stavu výstupů jednotlivých klopných obvodů dle obr. 4.31 můžeme sestavit pravdivostní tabulku čítače. Tato tabulka představuje binární kód, neboli převod desítkové soustavy do dvojkové, kde desítkovou soustavu tvoří počet vstupních impulzů (tabulka 4.8).

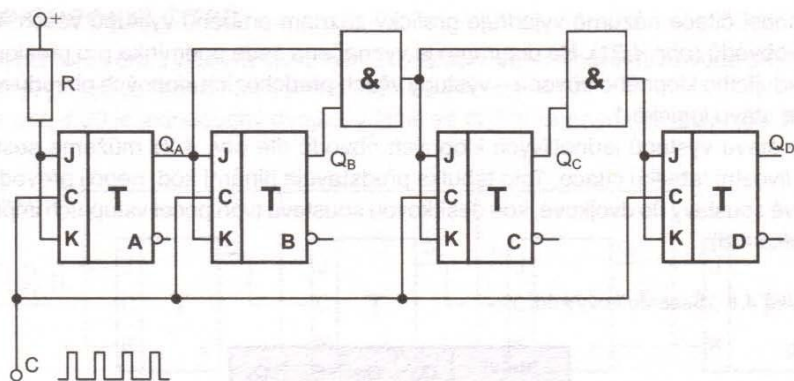
Tabulka 4.8 Šestnáctkový kód

počet vstupních impulzů	Q _D	Q _C	Q _B	Q _A
	2 ³	2 ²	2 ¹	2 ⁰
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0

Čítače, u kterých je každý klopný obvod spouštěn předcházejícím klopným obvodem, se nazývají asynchronní čítače. Zvětšující se počet klopných obvodů způsobuje zvětšování zpoždění, neboť zpoždění jednotlivých stupňů se sčítají. To omezuje rychlost sčítání asynchronního čítače.

SYNCHRONNÍ ČÍTAČ

V synchronních čítačích se příslušné klopné obvody spouštějí současně, protože vstupy hodinových impulzů jsou propojeny paralelně (obr. 4.32).



Obr. 4.32 Synchronní čítač

Činnost synchronního čítače

Čítač je sestaven z klopných obvodů JK, o kterých z předchozího výkladu víme, že při vstupních signálech $J = K = 1$ mění svůj stav vždy s příchodem hrany hodinového impulsu a při vstupních signálech $J = K = 0$ setrvávají ve svém původním stavu.

Vycházíme-li z těchto vlastností klopného obvodu, můžeme konstatovat, že první klopný obvod T_A mění stav výstupu Q_A s příchodem (případně s odchodem) každého hodinového impulsu, zatímco druhý klopný obvod T_B se překlápí při každém druhém hodinovém impulsu (výstup Q_B).

Podmínkou překlopení druhého klopného obvodu T_B je existence logické 1 na jeho vstupech J, K i C. Následující klopné obvody mohou překlopit opět jen s podmínkou jednotkových výstupů u všech předcházejících klopných obvodů. Tuto podmínku zajišťují přidané členy AND. Tím je zaručeno překlápění dle stejného pořádku jako u asynchronního čítače – viz obr. 4.31, ovšem s tím, že překlápění nastává u všech klopných obvodů najednou.

Činnost obvodu způsobuje, stejně jako u asynchronního čítače, že se počet vstupních impulsů (nebo jejich kmitočty) dělí dvěma, čtyřmi, osmi a šestnácti. Po šestnácti impulszech se uvede výstup čítače do původního (počátečního) stavu.

Snižovaný počet impulsů je možné znázornit na displeji a tak přesně měřit i poměrně vysoké kmitočty.

Synchronní čítače jsou složitější, ale jejich kmitočty čítání, limitovaný zpožděním jen jednoho stupně, může být proto vyšší než u asynchronních čítačů.

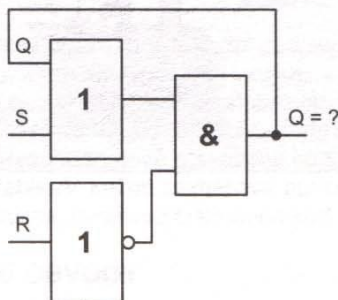
Kromě dvojkových čítačů, které počítají vstupní impulsy v dvojkovém kódu, existují čítače s různě zkráceným čítacím cyklem. Nejčastěji se setkáváme s dekadickými čítači, které jsou zapojeny tak, že za dvojkovým stavem $1001_{(2)} = 9_{(10)}$ následuje stav $0000_{(2)} = 0_{(10)}$ a současně je generován impuls pro přenos do vyššího dekadického řádu. Zkrácení čítacího cyklu je dosaženo pomocnými kombinačními obvody. Dekadické čítače mohou být zapojeny jako synchronní nebo jako asynchronní. Podobně se např. pro číslicové hodiny používají čítače s čítacím cyklem do 6 a do 12. Existují i zapojení s libovolně nastavitelným čítacím cyklem. Bližší údaje a zapojení těchto čítačů – viz [11].

Kontrolní otázky – test T15

1. Jaká je základní vlastnost sekvenčních logických obvodů?
2. Který obvod je základem sekvenčních logických obvodů?
3. Vyjmenujte minimálně čtyři druhy klopných obvodů!
4. Jaké vstupní hodnoty nastaví výstup Q klopného obvodu RS na logickou 1?
5. Co znamená zakázaný stav u obvodu RS?
6. Vyplňte pravdivostní tabulku klopného obvodu RS s dvěma členy NAND pro výstup Q (X je zakázaný stav)!

R	S	Q
0	0	?
0	1	?
1	0	?
1	1	?

7. Napište matematické vyjádření funkce klopného obvodu s dvěma členy NOR!
8. Odvoďte vstupní funkci pro následující obvod!



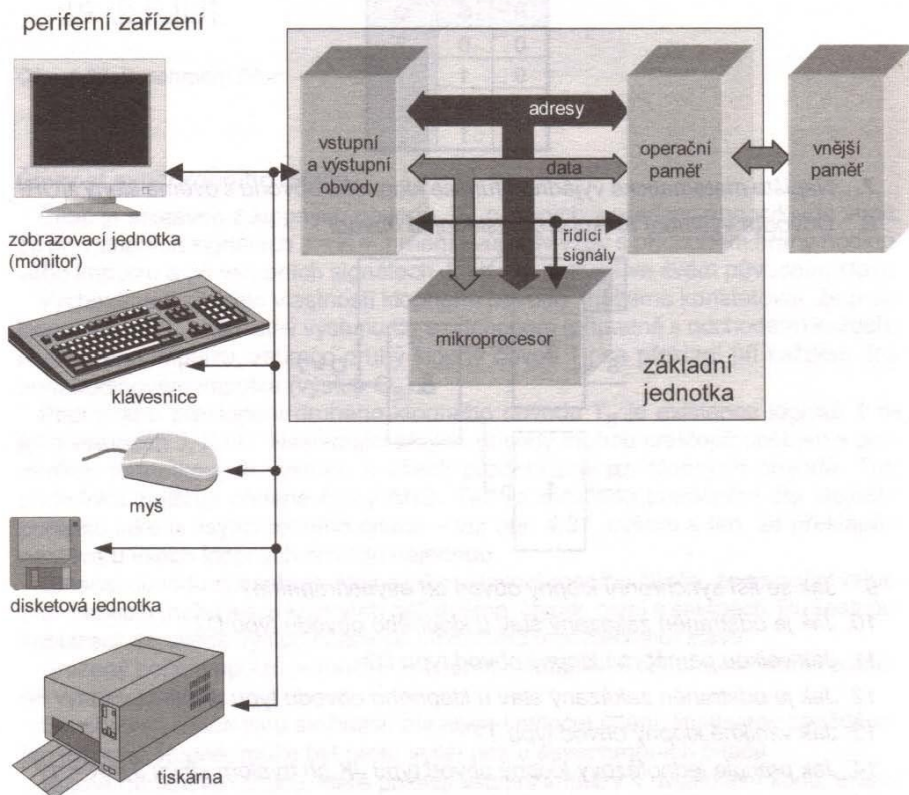
9. Jak se liší synchronní klopný obvod od asynchronního?
10. Jak je odstraněn zakázaný stav u klopného obvodu typu D?
11. Jak velkou paměť má klopný obvod typu D?
12. Jak je odstraněn zakázaný stav u klopného obvodu typu JK?
13. Jak vznikne klopný obvod typu T?
14. Jak pracuje jednofázový klopný obvod typu JK při trvalém stavu $J = K = 1$?
15. Jak pracuje dvoufázový klopný obvod typu JK při trvalém stavu $J = K = 1$?
16. Jaký je princip paměťového registru?
17. Jak pracuje kruhový registr?
18. Jaký je princip čítače?
19. Jak se liší synchronní čítač od čítače asynchronního?
20. Jaký je princip synchronního čítače?

5 MIKROPOČÍTAČOVÁ TECHNIKA

5.1 Struktura mikropočítače

HARDWARE, SOFTWARE

Mikropočítač se skládá z funkčních bloků, které se nazývají mikroprocesor, operační (hlavní paměť) a vstupní a výstupní obvody (obr. 5.1). Tyto funkční bloky spolu s dalšími logickými obvody a s vodiči potřebnými pro jejich spolupráci se nazývají *technické vybavení* nebo *technické prostředky* (angl. *hardware*) [6], [12].



Obr. 5.1 Struktura počítače

Aby bylo možné mikropočítač ovládat a aby mohl komunikovat s okolním světem, vybavuje se periferními zařízeními. Při použití mikropočítače je kromě technických prostředků nutné mít k dispozici i programy a data. Program obsahuje předpis pro zpracování dat, takže určuje a řídí činnost mikropočítače. Program, a tím i funkci

mikropočítače, lze měnit bez zásahu do technického vybavení. Programy a data se zahrnují pod souhrnný název *programové vybavení* (angl. *software*).

INSTRUKCE

Mikropočítač řeší určitou úlohu tím, že v časovém sledu provádí jednotlivé kroky programu (jednotlivé *instrukce*). Každá instrukce popisuje jednu relativně jednoduchou operaci, např. sečtení dvou dvojkových čísel. Komplikované algoritmy se zapisují jako posloupnost mnoha takovýchto jednoduchých základních operací.

MIKROPROCESOR

Mikroprocesor jako jádro mikropočítače generuje všechny řídicí signály potřebné pro současnou činnost a spolupráci všech použitých funkčních bloků. V závislosti na programu řídí výměnu dat s ostatními funkčními bloky, interpretuje instrukce programu a provádí je.

CPU

V anglosaské literatuře se mikroprocesor označuje zkratkou CPU (*Central Processor Unit*).

OPERAČNÍ PAMĚŤ

Program a data, která mají být zpracována, se zapisují do *operační paměti*. Tam se zapisují i mezivýsledky, které při výpočtech vznikají v mikroprocesoru.

Paměť je organizována po slovech nebo po slabikách. V osmibitovém mikropočítači obsahuje jedna paměťová buňka jednu slabiku, tedy 8 bitů. Instrukce programu, neboli řídicí příkazy, jsou dvouhodnotově zobrazená kódová slova, která se skládají z jedné nebo několika slabik. V každé paměťové buňce může tedy být zapsána jedna instrukce, část instrukce, dvojkové číslo nebo kód alfanumerického znaku.

VSTUPNÍ A VÝSTUPNÍ OBVODY

Vstupní a výstupní obvody spojují mikroprocesor s periferními zařízeními. Tyto obvody se skládají z registrů, které mají kapacitu jedno slovo. Prostřednictvím těchto registrů se provádí přenos znaků.

PERIFERNÍ ZAŘÍZENÍ

Jako *periferní zařízení* označujeme přístroje, které slouží ke komunikaci mikropočítače s okolním světem. Jsou to např. zobrazovací jednotky a tiskárny pro komunikaci s člověkem, vnější paměti (např. disková paměť) pro záznam programů a dat, která nejsou bezprostředně potřebná, ale i analogově číslicové převodníky, které spojují mikropočítač s řízeným procesem.

SBĚRNICE

Mikroprocesor je spojen s operační pamětí a se vstupními a výstupními obvody prostřednictvím různých vodičů. Jsou to: *datová sběrnice*, *adresová sběrnice* a *řídicí vodiče*. *Sběrnice* je skupina několika vodičů se stejnou funkcí. Obecně je sběrnici

nazývána soustava vodičů, jimiž proudí data, adresy nebo řídicí signály mezi komponentami PC. Veškeré informace přenášené sběrnicí se skládají z nul a jedniček – tedy z kombinací dvou stavů. Pokud má datová sběrnice 32 vodičů (každý vodič přenese jeden bit, jedná se tedy o 32bitovou sběrnici), může přenést 2^{32} kombinací. Čím více vodičů sběrnice má, tím lépe. Adresová sběrnice, většinou tvořená 16 vodiči, je nutná k tomu, aby bylo možné při výměně dat (čísel, znaků, instrukcí) mezi mikroprocesorem a operační pamětí adresovat žádanou buňku. Každé paměťové buňce je prostřednictvím adresového dekodéru přiřazena adresa ve tvaru 16místného dvojkového čísla. Mikroprocesor vyše prostřednictvím adresové sběrnice adresu, kterou žádá. Adresový dekodér potom připojí odpovídající paměťovou buňku prostřednictvím datové sběrnice na registr v mikroprocesoru. Pomocí řídicích signálů lze pak určit, zda má být přesunut obsah paměťové buňky do registru mikroprocesoru nebo naopak. Tento úkon se nazývá *čtení* (z paměti do mikroprocesoru) nebo *zápis* (z mikroprocesoru do paměti). Výměna dat mezi vstupními a výstupními obvody a mikroprocesorem probíhá stejně. V další části se podíváme do jednotlivých bloků základní jednotky počítače poněkud podrobněji.

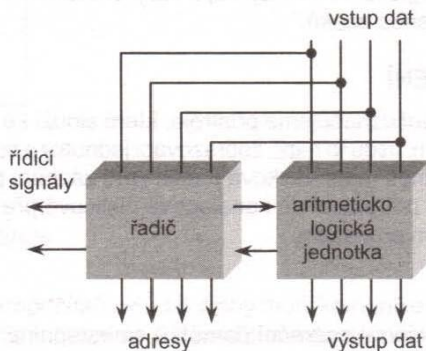
5.2 Processor

„Mozkem“ každého mikropočítače je procesor. Přitom není důležité, zda je použit k řízení pračky, průmyslové technologie nebo v kancelářském počítači, který obsahuje kartotéku zákazníků a vede účty. Procesor je nejkomplicovanější jednotka v mikropočítači. Funkuje jako rozhraní mezi programovým a technickým vybavením. Jeho prostřednictvím se z programů stávají v počítači reálné činnosti. Jako aktivní účastník řídí a sleduje veškerou činnost systému.

Podle názvoslovné normy ČSN 36 9001 je procesor definován jako „funkční jednotka počítače, která interpretuje a vykonává funkce“. Jinak řečeno, procesor získává data a instrukce z paměti, ukládá obojí tam, kde je ostatní jednotky budou mít po ruce, rozpoznává smysl instrukcí a pak instrukce provádí a výsledky zapisuje zpět do paměti.

ŘADIČ A ARITMETICKÁ JEDNOTKA

Procesor se skládá z *řadiče* a *aritmeticko-logické jednotky* (obr. 5.2).



Obr. 5.2 Propojení řadiče a aritmetické jednotky

Činnost procesoru je řízena programem. Program, který byl uložen do operační paměti, dává procesoru pokyny k provádění jednotlivých úkonů. Procesor zahájí svou práci tím, že přečte první instrukci programu a provede ji. Následující instrukce pak čte a provádí tak dlouho, až je zpracován celý program.

Jako příklad můžeme popsat podrobně jednotlivé kroky takto:

- a) Řadič přečte první instrukci: „Zapiš první sčítanec do aritmetické jednotky!“.
- b) Řadič přečte první sčítanec a zapíše jej do aritmetické jednotky.
- a) Řadič přečte následující instrukci: „Přičti druhý sčítanec ke sčítanci uloženému v aritmetické jednotce!“.
- b) Řadič provede tento příkaz.
- a) Řadič přečte třetí instrukci: „Zapiš součet do operační paměti!“.
- b) Řadič provede tento příkaz.

Tento příklad by bylo možné provádět dál, dokud by nebyl konečný výsledek vypočten a zapsán do operační paměti. Je zřejmé, že řadič je tou částí procesoru, která je aktivní a která zahajuje všechny činnosti. Aritmetická jednotka od něj pouze dostává příkazy a provádí operace.

Procesor má tedy více úloh. Musí ve správném pořadí číst z paměti instrukce, dekódovat je a potom je prostřednictvím aritmetické jednotky provádět.

Z uvedeného příkladu je patrné, že řadič pracuje dle programu vždy dvoustupňově:

- přečte instrukci,
- provede instrukci.

Pro procesor jsou instrukce vyjádřeny čísly. Každé instrukci odpovídá určité číslo, které procesor dokáže interpretovat jako instrukci. Tuto instrukci pak provede.

Díky pokroku technologie výroby polovodičových součástek je možné celý procesor realizovat jako jeden obvod LSI, který se nazývá *mikroprocesor*.

Mikroprocesor má určitý počet vstupních a výstupních vodičů, které umožňují výměnu dat mezi ním a pamětí nebo periferními zařízeními (*obr. 5.1*).

K *výstupním vodičům* patří adresové vodiče, jejichž prostřednictvím je možné volat určitou paměťovou buňku nebo vstupní a výstupní obvod, dále datové vodiče, které přenášejí informace do operační paměti nebo do vstupních a výstupních obvodů, a řídicí vodiče, které např. signalizují, zda se z určité paměťové buňky má číst nebo zda se má do ní zapisovat.

Ke *vstupním vodičům* patří kromě speciálních řídicích vodičů také datové vodiče, které přenášejí informaci z operační paměti nebo vstupních a výstupních obvodů do procesoru. K procesoru jsou tedy datové vodiče připojeny jako vstupní i jako výstupní.

Prostřednictvím svých vstupů i výstupů si procesor vyměňuje data s operační pamětí (popř. i se vstupními i výstupními obvody).

ŘADIČ

Řadič neboli *řídicí jednotka* (angl. *control unit* = CU) koordinuje všechny funkce počítače. Má tyto tři hlavní úkoly:

- Řídí pořadí v němž jsou prováděny instrukce programu.
- Dekóduje instrukce, případně je modifikuje.

- Vysílá do ostatních částí počítače všechny řídicí signály potřebné pro provádění instrukcí.

Jakým způsobem řadič tyto úkoly řeší, vidíme na obr. 5.3.

Činnost řadiče

Při zapnutí počítače se registr adresy vynuluje (program začíná na nule), tím se adresuje paměť počítače a na výstupu paměti se objeví číslo, které odpovídá kódu první instrukce.

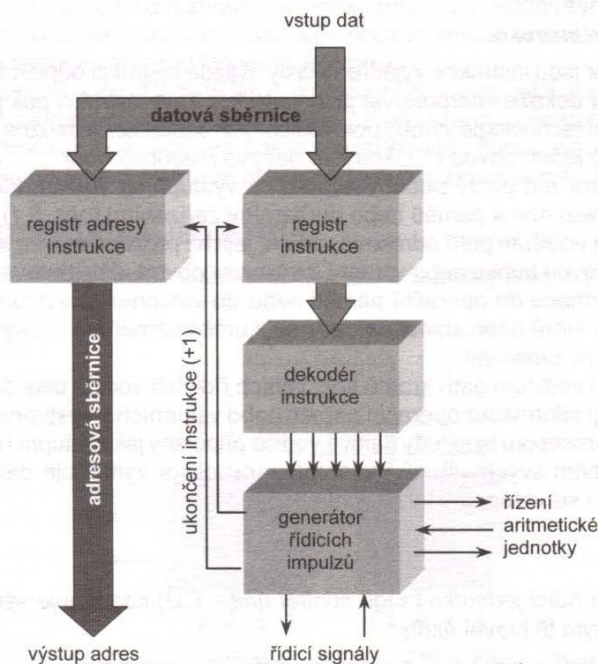
Tento kód se přivede do registru instrukce, kde se zapíše, aby měl připojený dekodér instrukce dost času k jeho dekodování.

Z dekodéru je dána informace do generátoru řídicích impulzů, která instrukce se má provést.

Generátor řídicích impulzů vysílá impulzy do všech ostatních částí – především do aritmeticko-logické jednotky.

Je-li instrukce provedena, vydá generátor povel registru adresy instrukce (+1) a na adresovém vstupu se objeví adresa následující instrukce a cyklus může probíhat znova.

Z obr. 5.3 je patrné, že datové spoje vedou k registru instrukce, ale také k registru adresy instrukce. Tím je umožněno, aby se speciálními instrukcemi měnil obsah tohoto registru. To je nutné např. tehdy, chceme-li v programu pokračovat v jiném úseku paměti. Adresa nového úseku paměti se potom vloží jako nový obsah do registru adresy instrukce, který tímto obsahem automaticky adresuje následující část programu a z ní čte příští instrukci.



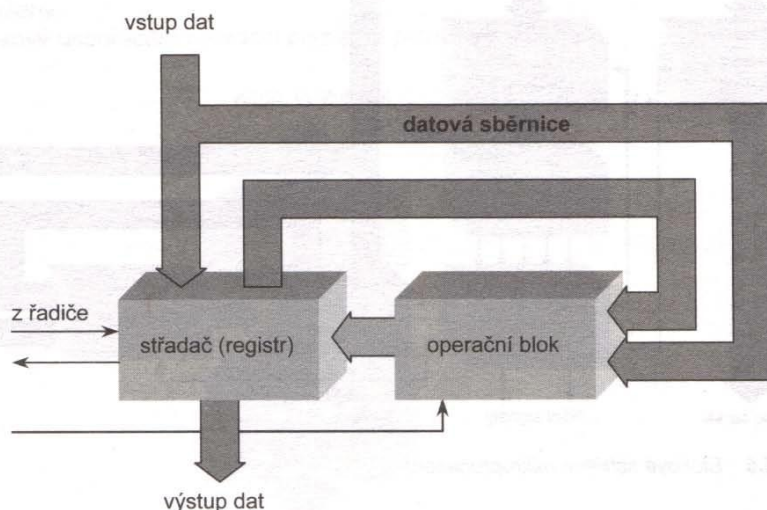
Obr. 5.3 Blokové schéma řadiče

ARITMETICKO-LOGICKÁ JEDNOTKA

Aritmeticko-logická jednotka (angl. *Arithmetic-Logic Unit* – ALU) je ta část počítače, která provádí výpočet, tj. aritmetické, logické a složitější operace, a to buď programem nebo pamětí převedením na sčítání a odečítání.

OPERAČNÍ BLOK

Jádrem aritmetické jednotky je *operační blok* (obr. 5.4). V něm se zpracovávají operandy (data), které jsou přivedeny na jeho dva vstupy, a na výstupu operačního bloku se výsledek operace předává k dalšímu použití.



Obr. 5.4. Blokové schéma aritmetické jednotky

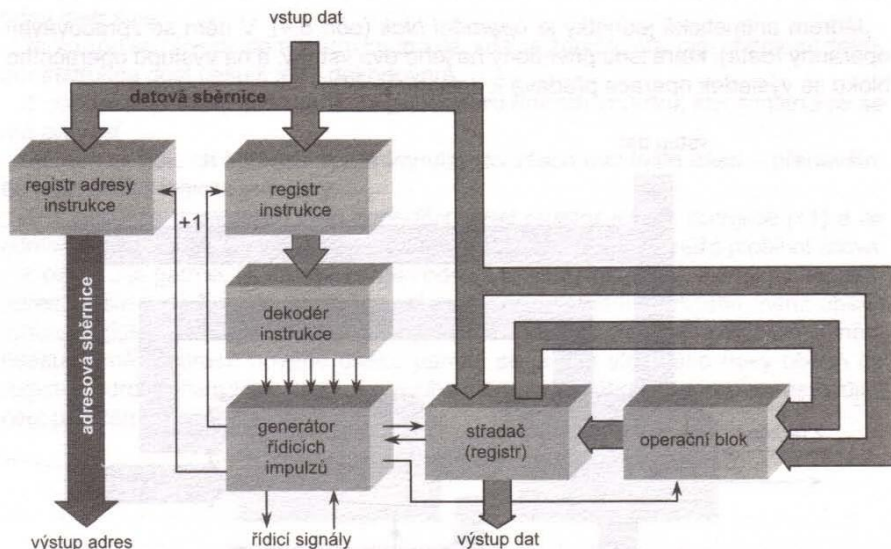
Aby bylo možné vypočítat součet dvou čísel, je třeba sdělit mikroprocesoru vlastní operační znak „sčítej“ a tři adresy, a to adresy obou sčítanců a adresu, na níž má být uložen výsledek. Instrukce, která by měla obsahovat všechny informace pro takový tříadresový procesor, by byla příslušně dlouhá. Téměř všechny běžně používané mikroprocesory však pracují na principu jednoadresového počítače. Kromě operačního znaku se navíc předává jen adresa druhého operandu. Avšak kde se vezme první operand a kam má být uložen výsledek?

STŘADAČ

K tomu aritmetická jednotka používá *střadač*, což je registr, v němž lze uchovat data. První operand se buď přečte samostatnou instrukcí z operační paměti a uloží se do střadače, nebo se tam nachází už jako výsledek předchozí operace. Při provádění vlastní operace se čte první operand ze střadače a druhý z operační paměti a výsledek se opět ukládá do střadače. Další instrukce může přenést výsledek ze

střadače do operační paměti. Jaká operace má být v aritmetické jednotce provedena, určuje prostřednictvím dekodéru instrukcí generátor řídicích impulzů řadiče (viz obr. 5.3).

Vzájemná spolupráce obou jednotek tj. řadiče a aritmetické jednotky je patrná z obr. 5.5, kde je znázorněna v zjednodušeném pojetí bloková struktura celého mikroprocesoru.



Obr. 5.5 Blokové schéma mikroprocesoru

Mikroprocesory se obvykle klasifikují podle délky zpracovávaných slov. Tato délka bývá 8, 16, nebo 32 bitů. Osmibitový mikroprocesor může zpracovávat slova s délkou 8 bitů. Má osmibitový střadač, a na každé adrese v operační paměti může být uloženo jedno slovo o délce 8 bitů.

Adresová, řídicí a datová sběrnice spojují mikroprocesor s ostatními bloky počítače. Avšak i uvnitř mikroprocesoru se data přenášejí po systému sběrnic. Například všechny registry včetně střadače jsou propojeny sběrnicí. Tím je umožněna výměna dat mezi jednotlivými registry.

Je rovněž možné přenášet po jedné sběrnici data ve dvou různých směrech. K tomu je třeba prostřednictvím řídicích vodičů určit, který blok je vysílačem a který přijímačem. Sběrnice, které mohou přenášet data v obou směrech, se nazývají *obousměrné*.

Mikroprocesor obsahuje řadu dalších obvodů, které jeho činnost urychlují nebo kontrolují. Např. *registr příznaků* hlídá mezní stavy, přípustnost aritmetických operací a umožňuje skoky programu (může měnit obsah adresy instrukce). *Cache paměti* umožňují načtení více dat ze sběrnice najednou, které pak čekají, až je procesor načte.

Aby byly mikroprocesory při časově náročných úkolech odlehčeny, existuje celá řada speciálních procesorů (koprocetorů), pracujících paralelně s hlavním procesorem, který jim přiděluje práci.

Jedná se např. o *aritmický procesor*, speciální vstupní a výstupní procesory, řadič pevného disku, procesor operační paměti atd.

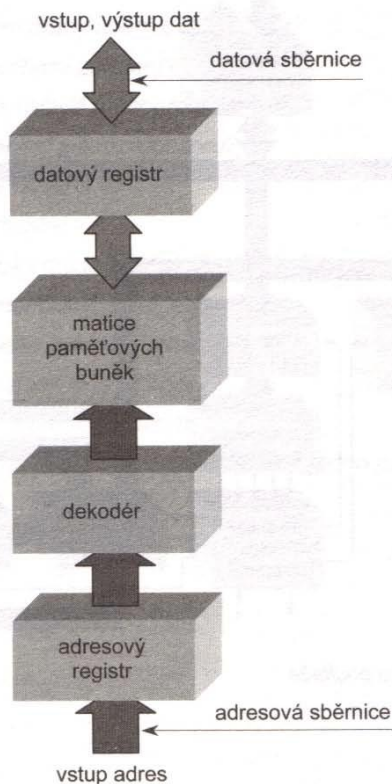
Moderní řešení mikroprocesorů (např. PENTIUM) umožňují zpracovávat několik instrukcí najednou.

Zakreslení a popisy těchto obvodů by byly na úkor jednoduchosti této publikace, proto odkazujeme na rozsáhlou počítačovou literaturu, např. [2], [6].

5.3 Operační paměť

Úlohou operační paměti je zapamatovat si na určitou dobu údaje, které jsou do ní vloženy.

Blokové uspořádání operační paměti je patrné z obr. 5.6.



Obr. 5.6 Blokové schéma operační paměti

Funkce jednotlivých bloků je následující:

Datový registr slouží k uchování na určitou dobu informace, která se má přečíst z paměťové buňky, nebo do ní napsat.

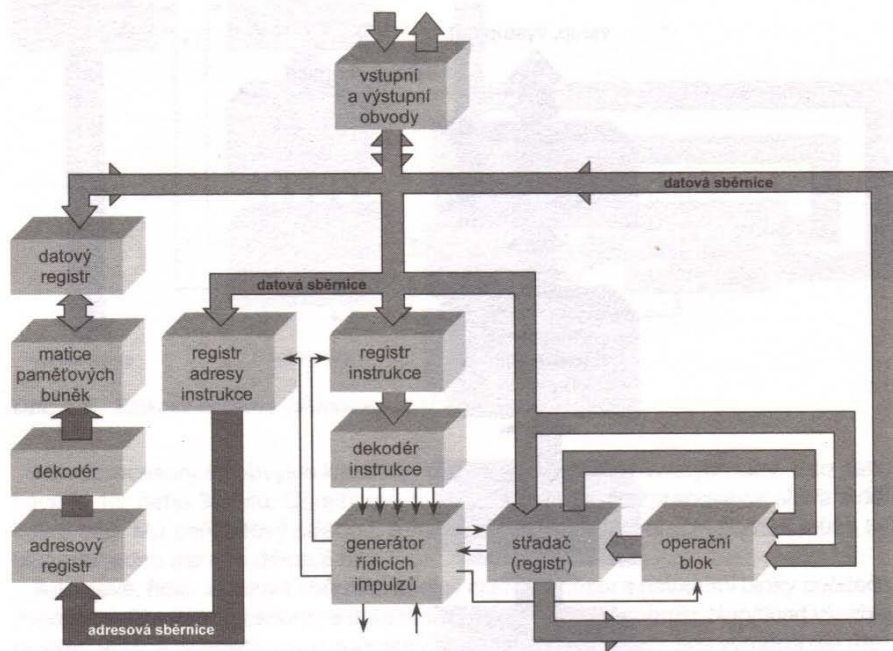
Matice paměťových buněk je tvořena z obvodů, které se vyznačují pamětí. Každá buňka má kapacitu jednoho bitu, 8 buněk, neboli 8 bitů tvoří jedno *paměťové místo* o kapacitě jednoho byte.

Dekodér připojuje (aktivuje) dle adresy odpovídající paměťovou buňku k příslušným proudovým zdrojům pro vybuzení záznamového nebo čtecího vodiče.

Adresový registr přijímá adresu paměťového místa z řadiče, tzn. adresu, do které se má zapisovat nebo z ní číst. Registr uchová adresu na potřebnou dobu.

Do tohoto místa jsme probrali, poněkud podrobněji, avšak rovněž blokově, vlastnosti a činnost dvou hlavních částí počítače – mikroprocesoru a paměti.

Blok vstupních a výstupních obvodů slouží pouze ke komunikaci počítače s periferními jednotkami. Spojením všech probíraných obvodů získáme blokové zapojení celého počítače (obr. 5.7).



Obr. 5.7 Blokové schéma počítače

Kontrolní otázky – test T16

1. Jaké funkční bloky má základní sestava počítače?
2. Co je to hardware?
3. Co je to software?
4. Co je to instrukce programu?
5. Jakou úlohu má v počítači mikroprocesor?
6. Jakou úlohu má operační paměť?
7. Co zajišťují vstupní a výstupní obvody?
8. Co to je a k čemu slouží sběrnice?
9. Jaké jsou základní druhy sběrnic dle funkce?
10. K čemu slouží periferní zařízení?
11. Z jakých bloků se skládá mikroprocesor?
12. Čím je řízena činnost mikroprocesoru?
13. Vyjmenujte bloky řadiče!
14. Popište činnost řadiče!
15. Jaké základní bloky má aritmetická jednotka?
16. Co zajišťuje aritmeticko-logická jednotka?
17. Proč je v aritmetické jednotce střadač?
18. Co znamená, že počítač je 32bitový?
19. Vyjmenujte bloky operační paměti!
20. Jakou kapacitu má jedna paměťová buňka a jedno paměťové místo?

5.4 Druhy a vlastnosti pamětí

V číslicových počítačích se používají paměti, které mají kvantitativně a kvalitativně různé vlastnosti.

KAPACITA PAMĚTI

Jedním z nejdůležitějších výkonnostních parametrů je *kapacita*, tedy počet paměťových míst.

VYBAVOVACÍ DOBA

Dalším parametrem je *vybavovací doba*, která udává časový interval, který uplyne od okamžiku, kdy je vyžádána informace, zapsaná na určitém paměťovém místě, do okamžiku, kdy je tato informace přečtena a předána do jednotky, která informaci žádala.

Schopnost uchování informace vyjadřuje, zda je paměť schopna celou informaci zachovat trvale, tj. po dlouhou dobu (měsíce, roky), nebo zda se data po vypnutí napájecího napětí nenávratně ztratí.

Rozlišujeme tři základní typy pamětí:

- *lokální paměť*,
- *operační paměť*,
- *velkokapacitní paměť*.

LOKÁLNÍ PAMĚŤ

Termínem *lokální paměť* je označena taková paměť, která je integrována v procesoru.

REGISTR

V mikropočítačové technice se pro ni používá název *registr* nebo *střadač* (angl. *accumulator*).

Takovéto paměti se používají např. k zápisu mezivýsledků při aritmetických operacích. Aby procesor mohl tyto operace provádět co nejrychleji, je vybavovací doba těchto pamětí velmi malá. Jejich kapacita je rovněž malá, obecně dosahuje hodnoty několika slov (po 8 nebo 16 bitech). Princip a zapojení paměťových registrů jsme poznali v kapitole Sekvenční logické obvody.

OPERAČNÍ PAMĚŤ

Operační paměť v počítači obsahuje jednak program, který má být proveden, a jednak velké množství dat, která se v průběhu provádění programu mají zpracovat. Obsah operační paměti není trvalý (paměť není energeticky nezávislá). To znamená, že operační paměť musí být naplněna daty dříve, než z ní program může číst. Operační paměti se dříve vyráběly z feromagnetických jader (feritové paměti). Dnes se používají polovodičové paměti z integrovaných obvodů s libovolným výběrem adresy, označované zkratkou **RAM** (z angl. Random Access Memory).

RAM

Operační paměť RAM je tedy paměť s přímým přístupem k datům, což znamená, že libovolné slovo může být vybráno pomocí jedné adresy. Do této paměti lze na libovolné paměťové místo zaznamenat několik bitů (slov) a podle potřeby zaznamenaná data z paměti číst.

ROM

Pro určité typy úloh počítače byl vyvinut speciální typ paměti, který umožňuje pouze čtení. Označuje se zkratkou **ROM** (z angl. Read Only Memory). Je to polovodičová paměť, která podobně jako paměť RAM umožňuje libovolné adresování, avšak navíc je permanentní a energeticky nezávislá, takže její obsah nelze měnit a uchovává informaci i po vypnutí napájecího napětí. Paměť ROM slouží pro uložení základních instrukcí, které řídí činnost počítače po jeho startu. Paměťová oblast mikropočítače pak většinou obsahuje určitý úsek sestavený pouze z paměti ROM, kde je uložen řídicí program, tzv. monitor, který je nutný pro správnou činnost a obsluhu celého počítače. Tento program se nesmí ztratit, a nesmí být tedy ani přepsán během práce počítače. Proto se takovýto program uloží do permanentní paměti. Do paměti ROM se programy zapisují u výrobce polovodičových součástek.

VELKOKAPACITNÍ PAMĚŤ

Velkokapacitní (vnější) paměť slouží pro dlouhodobé uchování velkých objemů dat. Jejich obsah zůstane zachován i po odpojení napájecího napětí (je to tedy *energeticky nezávislá paměť*). Vybavovací doba zde není tak důležitá, mnohem důležitější je kapacita této paměti.

PRUŽNÝ A PEVNÝ DISK

V mikropočítačové technice se nejčastěji používá malé provedení disku, nazývané *disketa* nebo *pružný disk* (angl. floppy-disk, zkratka FD) a *pevný disk* (angl. hard-disk, zkratka HD).

Ve velkokapacitní paměti je informace zpravidla zapsána sekvenčně, tedy v sérii za sebou. V takové paměti se najednou čte nebo píše celý blok několika údajů, které spolu souvisejí. Nelze tedy vybrat jedno konkrétní slovo, které je v paměti uloženo.

5.5 Realizace paměti RAM

POLOVODIČOVÉ PAMĚTI

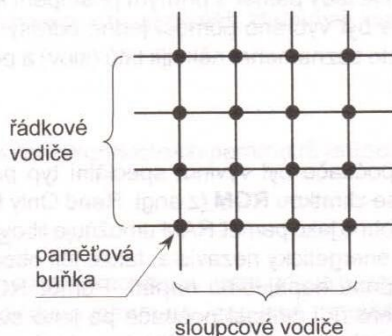
Vzhledem k zvládnutí technologie výroby integrovaných obvodů velké integrace (LSI) se *polovodičové paměti* staly nejrozšířenějším typem paměti.

Podle technologie výroby rozdělujeme polovodičové paměti na *bipolární* (s obvody TTL, ECL atd.) a *unipolární* (s tranzistory PMOS, NMOS, CMOS). Podle funkce rozlišujeme paměti statické a dynamické.

Paměťovým prvkem pro jeden bit je v paměti RAM bistabilní klopný obvod, složený ze dvou tranzistorů.

Jako všechny polovodičové paměti jsou i statické paměti RAM uspořádány na čipu do *matice*. Na obr. 5.8 vidíme její principiální uspořádání.

Matice je tvořena sítí *paměťových buněk*, které jsou realizovány bistabilními klopnými obvody.



Obr. 5.8 Matice paměťových buněk

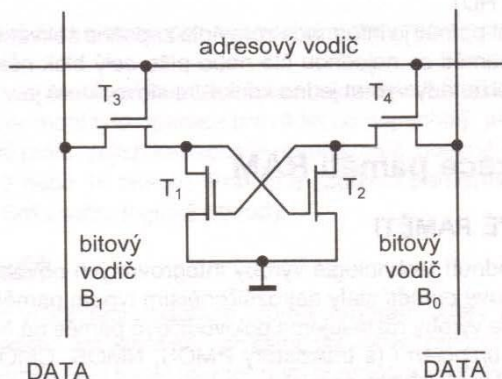
STATICKÉ PAMĚTI RAM (SRAM)

Vnitřní zapojení statické paměťové buňky je zjednodušeně znázorněno na *obr. 5.9*. Pro jednoduchost není na obrázku zakresleno napájení klopného obvodu.

Klopný obvod tvoří tranzistory T_1 a T_2 . Bitové vodiče B_1 a B_0 jsou připojeny na výstupy klopného obvodu pomocí tranzistorů T_3 a T_4 .

V klidovém stavu je *adresový vodič* přibližně na nulovém potenciálu, což způsobí zavření tranzistorů T_3 a T_4 , a tím i izolování bitových vodičů od klopného obvodu.

Aktivace buňky se provede přivedením napětí na adresový vodič. To způsobí otevření tranzistorů T_3 a T_4 , které připojí klopný obvod k bitovým vodičům. Je-li otevřen tranzistor T_1 , prochází v režimu čtení proud do bitového vodiče B_1 , a to je vyhodnoceno jako přečtení logické hodnoty 1. Vodič B_0 snímá inverzní hodnotu oproti vodiči B_1 .



Obr. 5.9 Zapojení paměťové buňky statické paměti

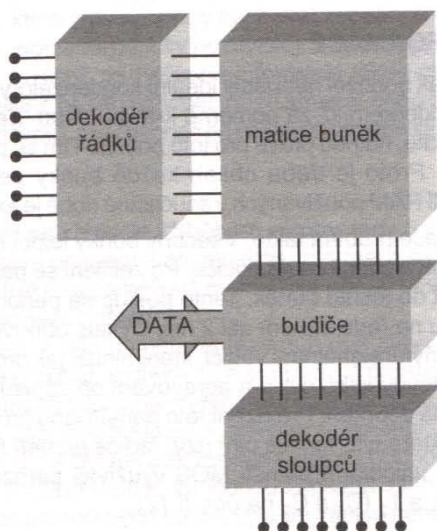
V režimu zapisování se po aktivaci adresového vodiče přivede bitovým vodičem B_1 přes tranzistor T_3 na řídicí elektrodu tranzistoru T_2 napětí, které jej otevře.

Pokles napětí na kolektoru tranzistoru T_2 se přeneše na řídicí elektrodu tranzistoru T_1 a ten se zavře. Po skončení záznamu se napětí na adresovém vodiči vrátí na úroveň klidového stavu.

Z obr. 5.9 je zřejmé, že pro jednu paměťovou buňku (jeden bit) je třeba použít nejméně čtyři tranzistory (dva ve funkci spínačů).

Adresování (výběr paměťových míst) prostřednictvím adresových vodičů je schématicky znázorněno na obr. 5.10. Součet počtu řádkových a sloupcových vodičů udává délku adresového slova. Data je třeba při zápisu i při čtení zesílit v budičích, aby činnost paměti probíhala bez chyby.

Technika statických pamětí RAM je již velmi dokonalá. Tyto paměti pracují velmi spolehlivě v širokém rozsahu teplot a za nepříznivých pracovních podmínek (např. při kolísání napájecího napětí).



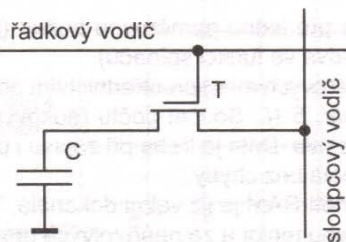
Obr. 5.10 Výběr paměťových míst

DYNAMICKÉ PAMĚTI RAM (DRAM)

Dynamické paměti RAM (DRAM) jsou v současné době nejpoužívanějším typem polovodičových pamětí. To je způsobeno velmi výhodným poměrem mezi jejich cenou a jejich kapacitou (cenou za bit) a dále malou spotřebou elektrické energie. Mají však i některé nevýhodné vlastnosti, především složitější řízení v porovnání se statickými pamětmi RAM (jak uvidíme dále).

Stejně jako u statických pamětí RAM jsou paměťové buňky na čipu uspořádány do matice. Strukturu jedné buňky znázorňuje obr. 5.11. Každá buňka je tvořena kondenzátorem a elektrickým přepínačem (na obrázku jej představuje tranzistor T), který ovládá nabíjení. Kondenzátor je schopen uchovat elektrický náboj. Není-li kondenzátor nabit, rovná se napětí na kondenzátoru přibližně nule. Nabíjením a vybíjením

kondenzátoru lze tedy zaznamenat dvouhodnotovou informaci (např. nabitý kondenzátor znamená logickou hodnotu 1, vybitý kondenzátor znamená logickou hodnotu 0).



Obr. 5.11 Princip paměťové buňky dynamické paměti

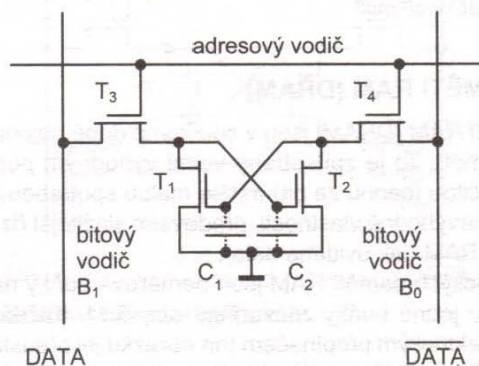
OBNOVOVÁNÍ INFORMACE

V reálném světě však bohužel neexistují ideální kondenzátory. Reálný kondenzátor se vybije svodovým odporem již za poměrně krátkou dobu. Zejména tehdy, jestliže má velmi malou kapacitu, nutnou právě pro toto použití. Tím se původně zaznamenaná informace ztrácí. Proto je třeba obsah každé buňky periodicky obnovovat. U dynamických pamětí RAM používaných v současné době je to obvykle každé 2 ms.

Obnovování informace probíhá takto: Všechny buňky ležící na jednom řádkovém vodiči se paralelně čtou a připojí se na budiče. Po zesílení se pak přečtená informace paralelně zapíše zpět do těchto buněk. Tento postup se periodicky opakuje tak, že všechny řádky přijdou na řadu během asi 2 ms. Cyklus obnovení informace se řídí zvenčí dvěma pomocnými signálními vodiči, které slouží jak pro zápis, tak pro čtení.

Časový průběh obnovovacího cyklu a adresování při zápisu a čtení je komplikovaný a musí být přesně dodržen. Pro řízení této paměti jsou proto vyvinuty speciální integrované obvody, které tyto funkce plní (tzv. *řadiče pamětí RAM*).

Dynamické obvody unipolární paměti MOS využívají parazitní kapacity řídicích elektrod tranzistorů T_1 a T_2 (C_1 a C_2 na obr. 5.12).



Obr. 5.12 Paměťová buňka DRAM

Pozornost je třeba věnovat také nevýhodné vlastnosti dynamických pamětí – větší citlivosti na rušení v porovnání se statickými pamětmi RAM. Nejvíce citlivé jsou na rušení částicemi α . Je to záření, které je schopno vniknout do jednotlivých paměťových buněk a vyvolat v nich dodatečný náboj. Tímto způsobem se může např. nabít nenabitý paměťový kondenzátor a tím se ztratí původně zaznamenaná informace. Částice α jsou vyzařovány hlavně z keramického materiálu pouzder integrovaných obvodů.

5.6 Realizace pamětí ROM

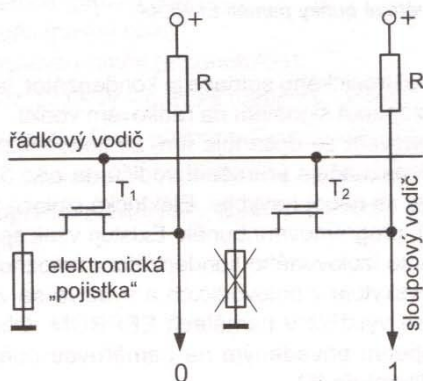
PERMANENTNÍ PAMĚTI

Všechny polovodičové paměti, jejichž typové označení končí písmeny ROM, jsou *permanentní paměti*, které jsou schopny zachovat zapsanou informaci i po vypnutí napájecího napětí (tj. jsou *energeticky nezávislé*). Můžeme je rozdělit do dvou hlavních skupin: paměti, v nichž se zapsaná informace nikdy nemůže změnit (ROM, PROM), a paměti, jejichž obsah uživatel může smazat nebo znovu naprogramovat pomocí speciálních přístrojů, a to dokonce po vestavění do obvodu (EPROM, EEPROM).

ROM

Obsah paměti ROM se zapisuje již během výroby, tedy u výrobce. Tento způsob programování se nazývá *programování maskou*. Informace se vkládá pomocí masky, která určuje vnitřní strukturu paměťové matice. Na obr. 5.13 vidíme strukturu buňky paměťové matice takového paměti.

Číslicovou informaci v této paměti reprezentují dva stavy „pojistky“ – stav, kdy pojistka je přepálená (logická hodnota 1), a stav, kdy je pojistka neporušená (logická hodnota 0). Zápis jednoho bitu do takového buňky lze pochopit velmi snadno. Jestliže chceme zapsat např. logickou hodnotu 1, přivedeme do buňky tak silný proud, že se pojistka přepálí. Pokud má být naopak zapsána logická hodnota 0, necháme pojistku nedotčenou. Z toho snadno vidíme, že jednou zapsaná informace nemůže být dále měněna, protože část paměťových buněk byla při zápisu nenávratně zničena.



Obr. 5.13 Struktura paměťové buňky paměti ROM

Uživatel, který chce zapsat do paměti ROM informaci, která je specifická pro jednu konkrétní aplikaci, musí předat výrobci požadavek na obsah této paměti. To se obvykle provádí formou programovací tabulky, v níž jsou detailně zapsána všechna data. Tento postup je velice pracný především pro uživatele a navíc je velmi nákladný. Z toho vyplynulo přání navrhnout takové obvody, které by uživatel mohl programovat sám. Tak vznikly paměti PROM (Programmable ROM), tj. *programovatelné permanentní paměti*.

PROM

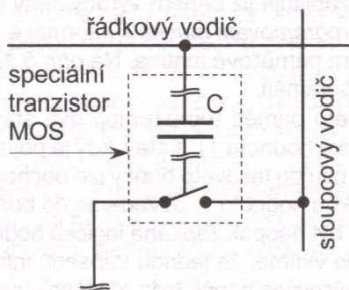
Paměti PROM jsou integrované obvody, které uživatel může sám naprogramovat pomocí speciálních programovacích přístrojů – *programátorů*.

Programování se zpravidla provádí až po ukončení vývoje programu, tj. tehdy, kdy už definitivně známe data, která mají být zapsána.

MAZATELNÉ PAMĚTI

Jestliže naopak potřebujeme programy stále a nebo občas měnit, je účelnější použít programovatelné *mazatelné paměti*.

Popis jejich funkce je v porovnání s pamětí RAM komplikovanější a proto jej značně zjednodušíme (viz obr. 5.14).



Obr. 5.14 Struktura paměťové buňky paměti EPROM

Na řídicím přívodu elektronického spínače je kondenzátor, jehož náboj rozhoduje o tom, zda lze spínač zapnout signálem na řádkovém vodiči.

Dlouhodobého pamatování se dosahuje tím, že kondenzátor je od svého okolí elektricky izolován, jak naznačuje přerušení vodičů na obr. 5.14, takže ani po vypnutí napájecího napětí se náboj nevybije. Elektrická izolace tohoto kondenzátoru ale komplikuje zápis, tj. programování buněk. Existují však speciální fyzikální jevy, které nabíjení takového izolovaného kondenzátoru umožňují. Je to tzv. *lavinový jev*, který se obecně vyskytuje v polovodičích a využívá se v pamětech EPROM, a *tunelový jev*, který se využívá v pamětech EEPROM. Oba jevy se vyvolávají poměrně vysokým napětím přivedeným na paměťovou buňku; jejich vysvětlení přenecháme odborné literatuře [6].

Postupně bylo vyvinuto více typů těchto pamětí.

EPROM

EPROM (Erasable PROM) patří mezi paměti, do nichž je možné opakovaně zapisovat. Paměťová informace se uchovává pomocí elektrického náboje. Ten je kvalitně izolovaný a tak udrží svoji hodnotu i po odpojení napájecího napětí. Také EPROM se programuje speciálním programátorem. Paměť je možno vymazat pomocí ultrafialového záření po dobu působení asi půl hodiny a po vymazání do ní opět zapsat nový program. EPROM lze poznat dle okénka na pouzdře, kterým vstupuje do paměti mazací ultrafialové záření. Z bezpečnostních důvodů se toto okénko po naprogramování zalepuje.

EEPROM

EEPROM (Electrically EPROM) jsou opět mazatelné paměti. Na rozdíl od EPROM se mažou elektrickými impulzy, doba mazání se pohybuje v milisekundách. Počet zápisů v EEPROM i EPROM je ohraničený, protože polovodičový materiál se časem unavuje; doba pamatování uložené informace je omezena asi na 10 až 20 let. V ROM i PROM není doba pamatování ohraničena. Pro EEPROM se používají i označení EAROM nebo E²ROM.

FLASH-PROM

Flash-PROM (někdy i Flash EPROM) je posledním typem ROM. Je rychlejší než předešlé typy, v principu se s ní dá pracovat jako s RAM, ale po odpojení napětí se nevymaže. Snese asi 1000 cyklů programování-výmaz. Její hlavní předností je možnost přeprogramování přímo v počítači.

VELKOKAPACITNÍ PAMĚTI

Velkokapacitní paměti patří do periferie číslicového počítače (viz obr. 5.1). Jelikož se nejedná o přímou souvislost s logickými obvody, omezíme se pouze na přehled jejich principů:

- *Paměti s pohyblivou magnetickou vrstvou,*
 - a) disketové paměti (pružný disk),
 - b) diskové paměti (pevný disk),
 - c) kazetové páskové paměti (magnetofon).
- *CD-ROM, CD-R, CD-RW – optická paměťová média.*
- *DVD (Digital Versatile Disc, původně Digital Video Disc).*
- *Blu-ray Disc – velkokapacitní optický disk na bázi modrého laseru.*
- *Další, např. magnetooptické disky, Bernoulliho disky, mechanika Jaz, ZIP atd. (viz literaturu [6]).*

1. Jaké tři druhy pamětí dle velikosti rozlišujeme?
2. Jaké vlastnosti má paměť RAM?
3. Jaké vlastnosti má paměť ROM?
4. K čemu slouží paměť ROM?
5. K čemu slouží vnější velkokapacitní paměť?
6. Kdy je paměť energeticky nezávislá?
7. Jaký obvod tvoří základ elektronické statické paměti RAM?
8. Co je to paměťová matice?
9. K čemu slouží adresový vodič v paměti?
10. Jak se provádí aktivace buňky?
11. Jakým způsobem je čten obsah buňky?
12. Jak se provádí zápis data do paměťové buňky?
13. Jaký je princip dynamické paměti RAM (DRAM)?
14. Jak se provádí obnovování informace u paměti typu DRAM?
15. Jak se vytvoří obsah paměti ROM?
16. Co je to paměť PROM?
17. Co znamená pojem mazatelná paměť ROM?
18. Co je to paměť EPROM?
19. Co je to paměť EEPROM?
20. Vymenujte některé systémy velkokapacitní paměti!

Kapitola 1 (Zpracování informace) – test T12

1. Obsah sdělení v dokumentu nebo ve slovním vyjádření.
2. Časově proměnná fyzikální veličina.
3. Spojitý signál se mění plynule, nespojitý skokem.
4. Metoda přiřazení čísla každému malému úseku analogového signálu.
5. Nejmenší množství informace určující volbu ze dvou možností.
6. Vyjádření skutečnosti v podobě číslíkových, abecedních nebo grafických znaků.
7. K převodu textu nebo souboru znaků do nového souboru.
8. Předpis jak k sobě jednoznačně přiřadit prvky dvou množin.
9. Dvojková soustava má za základ číslo 2 a používá dvou číslic 1 a 0.
10. 13.
11. 11011.
12. 10001.
13. Převod čísel 0 až 9 desítkové soustavy do dvojkové soustavy.
14. K zestručnění zápisu čísla ve dvojkovém kódu.
15. 4B.
16. Záznamem (uložením) do permanentní paměti.
17. Slabika, která obsahuje 8 bitů.
18. $2^8 = 256$.
19. $2^{10} = 1024$ krát.
20. K převodu znakových údajů na binární čísla.

Kapitola 2 (Logické funkce a obvody) – test T13

1. Obvod, jehož vstupní a výstupní veličiny nabývají jen dvou hodnot.
2. Vzájemný vztah vstupů a výstupů logického obvodu.
3. Výstup je určen jen kombinací vstupních veličin.
4. Výstup je určen nejen kombinací vstupních veličin, ale i hodnotami předcházejícího stavu logického obvodu.
5. Napětíovou úrovní (H – vysoká = 1, L – nízká = 0).
6. Logický součet, součin a negace.
7. Slovním výrokem, matematickou funkcí, pravdivostní tabulkou, realizačním obvodem.
8. Výstup je 1 tehdy a jen tehdy, jsou-li oba vstupy 1.
9. a) Výstup je 1, je-li alespoň jeden vstup 1.
b) Výstup je 0 tehdy a jen tehdy, jsou-li oba vstupy 0.
10. $Y = 1, 1, 1, 0$.
11. $Y = 1, 0, 0, 0$.
12. Vnitřní zapojení je stejné, odlišnost je ve vnějším zapojení.

13. $1 + 1 = 1$
14. $A \cdot B$
15. $A \cdot B = \overline{A + B}$, $A + B = \overline{A \cdot B}$
16. $Y = A \cdot B$ (AND).
17. Vznikne AND.
18. Zjednodušování obvodů použitím pravidel Booleovy algebry.
19. Pravdivostní tabulka, algebraická funkce, zjednodušení, návrh prvků pro realizaci.
20. Nejčastěji tranzistory.

Kapitola 3 (Kombinační logické obvody) – test T14

1. Výstup je 1, jsou-li vstupy rozdílné.
2. $Y = \overline{A} \cdot B + A \cdot \overline{B}$
3. XOR a AND.
4. Sečíst dva jednobitové vstupy a generovat případný přenos.
5. Přijmout bit z nižšího řádu.
6. 2× XOR, 2× AND, 1× OR.
7. V logickém členu AND.
8. Vygenerování paritních bitů na vstupu a výstupu a jejich porovnání.
9. K indikaci vzniku možné chyby.
10. Obvody, které slouží k převodu desítkového čísla na dvojkové.
11. Převádí čísla 0, 1, 2 nebo 3 na dvoubitové binární číslo.
12. Uzemněním libovolného vstupu členu NAND se na jeho výstupu vybudí logická 1.
13. Převádí desítkové číslo 0 až 9 na čtyřbitové binární číslo.
14. Zjištění přítomnosti binárního čísla nebo zjištění stavu binárního čísla.
15. Převádí čtyřbitové číslo na dekadické číslo 1 až 9.
16. Přepíná z několika vstupů vždy jen do jednoho výstupu.
17. Přepíná jeden vstup mezi několika výstupy.
18. Dekodérem.
19. K porovnání dvou binárních čísel.
20. XOR.

Kapitola 4 (Sekvenční logické obvody) – test T15

1. Výstup je určen kombinací vstupů a předchozím stavem výstupu.
2. Klopný obvod bistabilní.
3. RS, D, JK, T.
4. $R = 0$, $S = 1$.
5. Taková kombinace vstupních veličin, kdy po jejich změně na opačný stav překlopí výstup obvodu do náhodného stavu.
6. $Q = X$, 1, 0, 1/0.
7. $Q = R(S + Q)$

8. $Q = \overline{R}(S + Q)$
9. Synchronní obvod je otevírán přítomností hodinového impulsu C.
10. Zařazením invertoru mezi vstupy R a S.
11. 1 bit.
12. Zajištěním jednoho nulového zpětnovazebního vstupu pro $Q = 0$.
13. Z obvodu JK spojením jeho vstupů.
14. Obvod překlápí vždy s příchodem hodinového impulsu.
15. Obvod překlápí s tylem hodinového impulsu.
16. Sériové nebo paralelní spojení klopných obvodů typu D nebo JK.
17. Data v registru rotují.
18. Použití klopných obvodů JK nebo T, které překlápí s příchodem (nebo s odchodem) hodinového impulsu.
19. U synchronního čítače se příslušné klopné obvody spouštějí současně, v asynchronním postupně.
20. Překlopení každého následujícího členu čítače je podmíněno logickou 1 na všech výstupech předchozích členů.

Kapitola 5 (Mikro počítačová technika – Struktura mikro počítače, Procesor, Operační paměť) – test T16

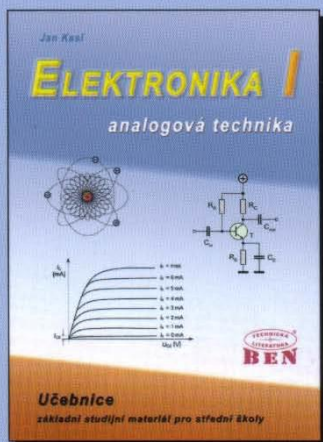
1. Vstupní a výstupní obvody, operační paměť, mikroprocesor.
2. Fyzické komponenty počítačového systému.
3. Pojem pro veškeré programové vybavení a data.
4. Jednotlivý krok programu popisující jednoduchou operaci.
5. Generuje z dat všechny řídicí signály, kterými pak řídí činnost ostatních bloků.
6. Zapamatovat si na určitou dobu údaje do ní vložené.
7. Propojení mikroprocesoru s periferními přístroji.
8. Skupina několika vodičů se stejnou funkcí propojující bloky počítače.
9. Adresová a datová.
10. Ke komunikaci PC s okolním světem.
11. Z řadiče a aritmetické jednotky.
12. Programem uloženým do operační paměti.
13. Registr instrukce, dekodér instrukce, generátor řídicích impulsů, registr adresy instrukce.
14. Zapiše 1. instrukci do registru, tu dekoduje, generátor řídicích impulsů vyšle příkaz a posune adresu o jedno místo.
15. Sřadač (registr) a operační blok.
16. Provádí aritmetické a logické operace.
17. Je to registr k ukládání vstupních dat a mezivýsledků.
18. Procesor, sběrnice atd. zpracovávají najednou slovo o délce 32 bitů.
19. Adresový registr, dekodér, matice paměťových buněk, datový registr.
20. Buňka jeden bit a paměťové místo jeden byte.

1. Lokální, operační a velkokapacitní.
2. Paměť s možností čtení i zápisu.
3. Paměť umožňující pouze čtení dat.
4. K uložení základních instrukcí pro řízení činnosti počítače po startu.
5. K dlouhodobému uchování velkých objemů dat.
6. Zůstane-li její obsah zachován i po odpojení napájecího napětí.
7. Bistabilní klopný obvod složený ze dvou tranzistorů.
8. Způsob uspořádání sítě paměťových buněk.
9. K aktivaci (připojení) příslušných buněk.
10. Přivedením napětí na adresový vodič.
11. Vyhodnocením proudu, který prochází otevřeným tranzistorem do bitového vodiče.
12. Přivedením napětí na řídicí elektrodu zavřeného tranzistoru.
13. Kondenzátor u kterého se obnovuje náboj.
14. Údaje buněk se periodicky přečtou, zesílí a znovu zapíší.
15. Pomocí programovací masky se přepálí (nebo zachovají) příslušné prvky (pojistky) v jednotlivých buňkách.
16. Paměť ROM, kterou si uživatel naprogramuje sám, nelze přepisovat.
17. Paměť ROM, kterou je možné změnit (vymazat a znovu zapsat).
18. Mazatelná permanentní paměť, kterou lze smazat ultrafialovým světlem.
19. Permanentní paměť mazatelná elektrickými impulzy.
20. Diskové paměti, CD-ROM, DVD disky.

LITERATURA

- [1] Hunger A. – Kohl A. **Mikropočítače pro každého**
Praha, SNTL 1990
- [2] Kolenička J. – Boltík J. **Technika počítačů**
Praha, SNTL 1988
- [3] Zůna P. **Informatika a výpočetní technika**
Praha, GRADA 1993
- [4] Bílek J. – Bayer J. **Základy automatizace**
Praha, SNTL 1990
- [5] Kralovič P. **Průmyslová elektronika**
Praha, SNTL 1998
- [6] Horák J. **Učebnice hardware**
Brno, Computer Press 1998
- [7] Jedlička P. **Přehled obvodů CMOS 4000 – 1. díl**
Praha, BEN – technická literatura 1995
- [8] Jedlička P. **Přehled obvodů CMOS 4000 – 2. díl**
Praha, BEN – technická literatura 2000
- [9] Jedlička P. **Přehled obvodů řady TTL 7400 – 1. díl**
Praha, BEN – technická literatura 1997
- [10] Jedlička P. **Přehled obvodů řady TTL 7400 – 2. díl**
Praha, BEN – technická literatura 1998
- [11] Matoušek D. **Číslicová technika**
Praha, BEN – technická literatura 2002
- [12] Vrátil Z. **Postavte si PC**
Praha, BEN – technická literatura 2002
- [13] Valášek P. – Loskot R. **Polovodičové paměti – 2. vydání**
Praha, BEN – technická literatura 2000

VAŠE DALŠÍ LITERATURA



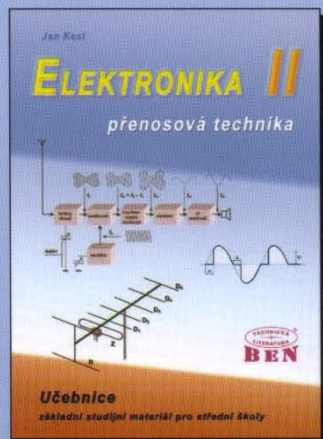
Elektronika 1 – analogová technika

Učebnice seznamuje studenty s vlastnostmi elektronických součástek: rezistoru, kondenzátoru, cívky, diody, tranzistoru, tyristoru a integrovaného obvodu, a dále s jejich aplikací v jednoduchých obvodech, zejména v usměrňovačích, zesilovačích a oscilátorech.

Učebnice umožňuje snadněji pochopit základní pojmy elektroniky, neboť je kladen důraz na jednoduchost a přehlednost. Příklady výpočtů jsou uváděny jen výjimečně, a to v kontrolních otázkách u konce každé kapitoly. Správné odpovědi jsou uvedeny na konci učebnice.

Z obsahu: Lineární prvky elektronických obvodů, Vakuové nelineární prvky (elektrony), Polovodičové nelineární prvky, Elektronické obvody...

Autor Ing. Jan Kestl, rozsah 120 stran B5, objednáací číslo 121117, MC 149 Kč.



Elektronika 2 – přenosová technika

Bezprostředně navazuje na tento první díl. Obsah lze rozdělit do třech tematických celků.

V první části se žák seznámí se základy impulzní techniky a s využitím polovodičových součástek v silnoproudé elektrotechnice.

Druhá část se zabývá principy přenosu informace (elektroakustika, modulace, demodulace, vznik a šíření elektromagnetických vln).

Náplní třetí části je pojednání o rozhlasovém a televizním přenosu.

Autor Ing. Jan Kestl, rozsah 128 stran B5, objednáací číslo 121118, MC 149 Kč.

TECHNICKÁ
LITERATURA
BEN
<http://www.ben.cz>

Doporučená cena
149 Kč

Objednáací číslo
121119

ISBN 80-7300-182-9



9 788073 001827