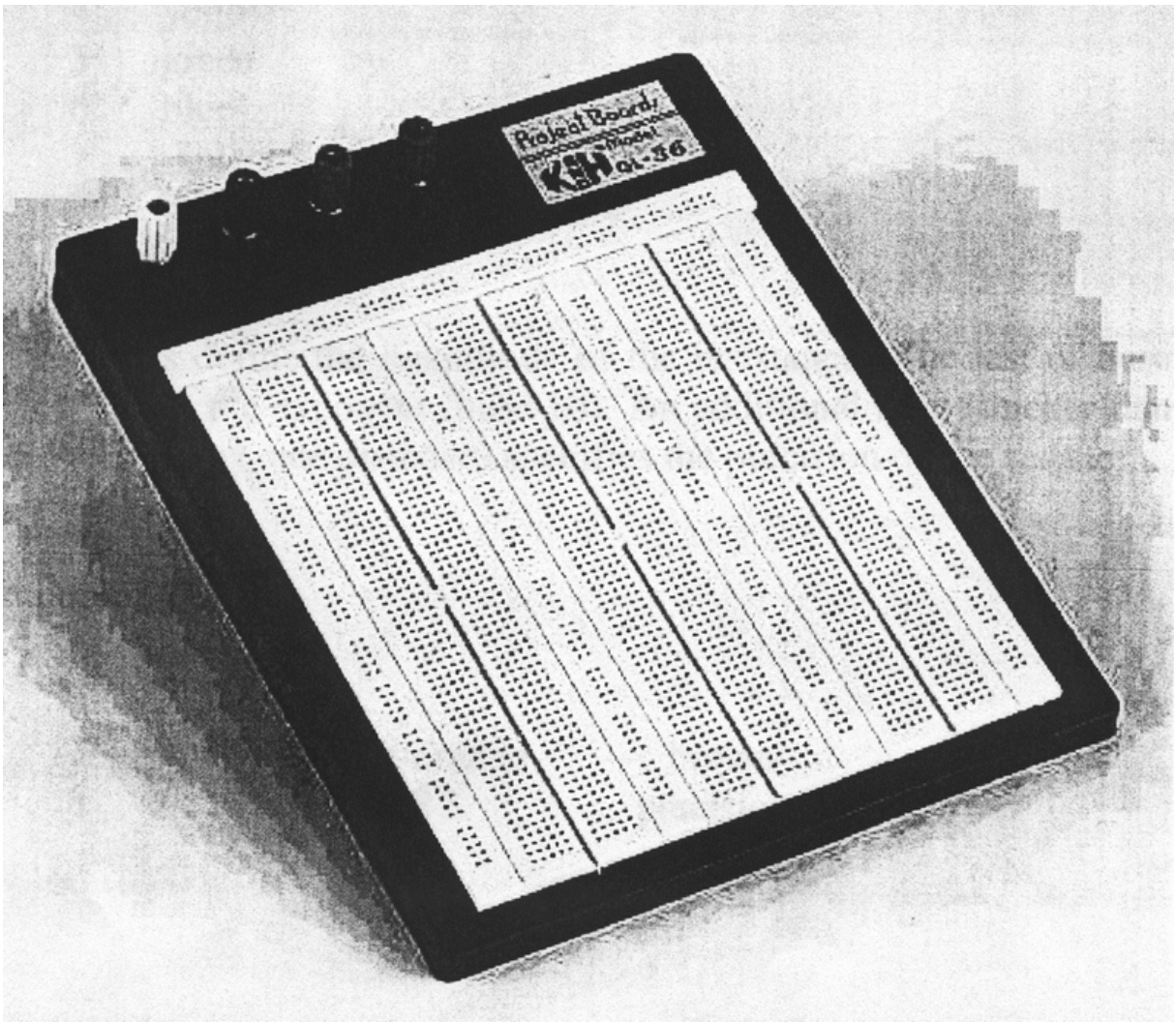
DIGITÁLNÍ ELEKTRONIKA V TEORII A EXPERIMENTECH

Přeloženo ze studijních materiálů
Erich Steinbauer, Dipl.-Ing. Dr., Ass.-Prof.

OBSAH

EXPERIMENTÁLNÍ ZAŘÍZENÍ

Pro pokusy používáme elektronickou desku.K propojení jsou použity neohebné vodiče.Tento systém umožňuje snadno a rychle zapojovat elektronické obvody a to umožňuje velmi dobrý elektrický kontakt .



KÓDOVÁNÍ REZISTORŮ

Barevné kroužky:

-1	zlatá
0	černá
1	hnědá
2	červená
3	oranžová
4	žlutá
5	zelená
6	modrá
7	fialová
8	šedá
9	bílá

3 nebo 4 kroužky určují velikost odporu.Poslední z těchto kroužků definuje exponent (základ 10).Doplňkový kroužek (silnější nebo v jiné vzdálenosti) specifikuje přesnost rezistoru.

Příklad(470kΩ):

žlutá-	fialová-	černá-	oranžová	--hnědá
4	7	0	E3	přesnost

DIGITÁLNÍ SIGNÁLY

- Analogový signál může pracovat s jakýmkoliv napětí v jejich určeném rozmezí.
  Příklad:Výstupní napětí zesilovače.

- Digitální signál může pracovat pouze s konečným počtem logických stavů (většinou se 2).
  V našem případě budeme brát v úvahu pouze 2 stavy:

  - F/W (falsch/wahr)
  - F/T (false/true)
  - 0/1

- Digitální obvody pracují s elektrickým napětím.Dva logické stavy musí být blízké dvěma nepřekrývajícím se napěťovým hodnotám.Tyto hodnoty jsou označovány jako L (low,low level) a H (high,high level).

- Zvolení těchto hodnot je libovolné a závisí na výrobní technologii elektronického obvodu.

- Pozitivní logika:
  Napěťová hodnota L je označena jako logický stav 0(F),
  napěťová hodnota H je označena jako logický stav 1(T).

- Negativní logika:
  Napěťová hodnota L je označena jako logický stav 1(T),
  napěťová hodnota H je označena jako logický stav 0(N).

VAROVÁNÍ:Definice logických operátorů jako AND (logický součin) nebo OR (logický součet) je vždy určena logickým stavem a NIKDY hodnotou napětí.

ZÁKLADNÍ LOGICKÉ FUNKCE

Pro základní logické funkce jsou anglické názvy užívány v cizích jazycích dost často.

AND (logický součin)

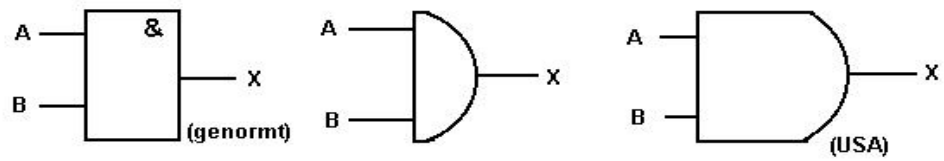
Syntaxe :

X=A $_{\Lambda}$ B (normovaný)
X=A • B
X=A & B

Pravdivostní tabulka :

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

Symboly :



OR (logický součet)

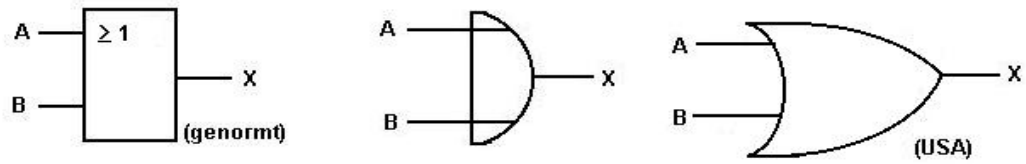
Syntaxe :

X= A $\vee$ B (normovaný)
X= A + B

Pravdivostní tabulka :

A	B	X
0	0	0
0	1	0
1	0	1
1	1	1

Symboly :



NOT (negace)

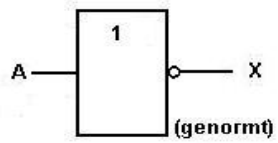
Syntaxe :

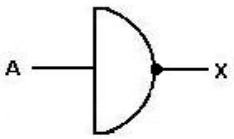
X= $\overline{A}$ (normovaný)
X = !A

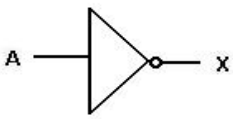
Pravdivostní tabulka :

A	X
0	1
1	0

Symboly:







Kompletní poučka:

Všechny logické funkce , které jsou definovány pravdivostní tabulkou mohou být vyjádřeny výrazem, který obsahuje pouze AND , OR , a NOT operátory. Všechny logické funkce mohou proto být dedukovány z těchto základních operátorů.

Znak: AND , OR a NOT jsou dostatečné , ale jsou ostatní menší jednotky funkcí (např. AND , NOT nebo dokonce jednotlivé funkce NAND = NOT AND).

ODVOZENÉ LOGICKÉ FUNKCE

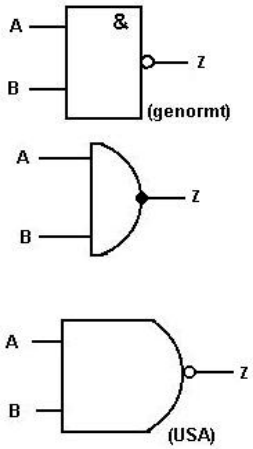
NAND

Syntaxe : X = $\overline{A \wedge B}$ (normovaný)

Pravdivostní tabulka :

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

Symboly :



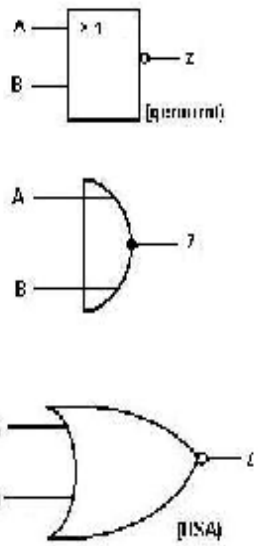
NOR

Syntaxe : X = $\overline{A \vee B}$ (normovaný)

Symboly :

Pravdivostní tabulka :

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0



XOR (antivalence, ne-ekvivalence, exkluzivní OR)

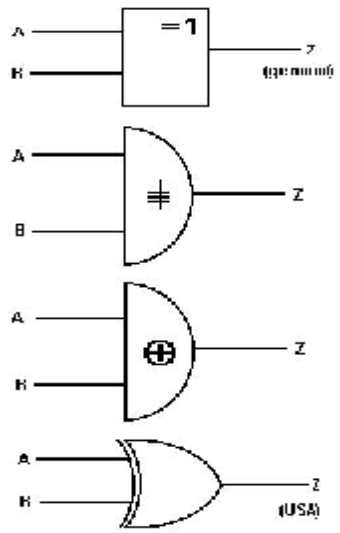
Je pravdivý , když jeho vstupy mají různé hodnoty .

Syntaxe : X = A + B

Pravdivostní tabulka :

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Symboly :



XNOR (ekvivalence , exkluzivní NOR)

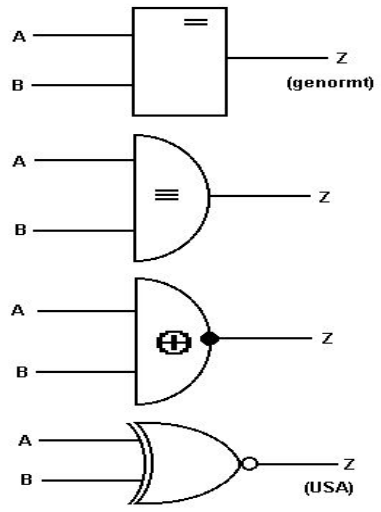
XOR operátor je pravdivý , když jeho vstupy mají stejné stavy .

Syntaxe : X = $\overline{A + B}$

Pravdivostní tabulka :

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

Symboly :



BOOLOVA ALGEBRA

Důkaz těchto pravidel může být proveden sestavením odpovídající tabulky pravděpodobnosti.

Komutativní zákon:

$A \wedge B = B \wedge A$ $\qquad$ $A \vee B = B \vee A$

Asociativní zákon:

$A \wedge (B \wedge C) = (A \wedge B) \wedge C$ $\qquad$ $A \vee (B \vee C) = (A \vee B) \vee C$

Distributivní zákon:

$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$ $\qquad$ $A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$

Absorpční zákon:

$A \wedge (A \vee B) = A$ $\qquad$ $A \vee (A \wedge B) = A$

Tautologie:

$A \wedge A = A$ $\qquad$ $A \vee A = A$

Zákon negace:

$A \wedge \overline{A} = 0$ $\qquad$ $A \wedge \overline{A} = 1$

Dvojitá negace:

$\overline{\overline{A}} = A$

De Morganův teorém:

$\overline{A \wedge B} = \overline{A} \vee \overline{B}$ $\qquad$ $\overline{A \vee B} = \overline{A} \wedge \overline{B}$

Operace s 0 a 1 :

$A \wedge 1 = A$ $\qquad$ $A \vee 0 = A$

$A \wedge 0 = 0$ $\qquad$ $A \vee 1 = 1$

ZÍSKÁVÁNÍ LOGICKÝCH ROVNIC

V digitální elektronice a logice jsou funkce většinou definovány pravdivostní tabulkou. Jak můžeme najít ekvivalentní logické rovnice tak, aby se funkce dala sestavit použitím jednoduchých komerčně dostupných nástrojů?

Disjunktivní normálová formulace

Například, logická funkce Y(A,B,C) může být definována následující pravdivostní tabulkou:

A	B	C	Y (output)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

K nalezení logických rovnic pro Y použijeme následující algoritmus:

- Najdi řádky pro které je Y=1.
- Z těchto řádků zformulujeme konjunkci (AND) proměnných x_i. Jestliže má proměnná má hodnotu 1, proměnná x_i se musí použít v konjunkci. Jestliže má hodnotu 0 proměnná x_i (z negace x_i) se použije.
- Dostaneme tolik konjunkcí kolik je počet 1 na výstupu.
- Konečně požadovaná logická funkce je disjunkce (OR) všech konjunkčních podmínek.

V našem případě:

$$Y = (\overline{A} \wedge B \wedge \overline{C}) \vee (A \wedge \overline{B} \wedge \overline{C}) \vee (A \wedge B \wedge \overline{C})$$

Proč tento algoritmus funguje?

- Každé z konjunkčních podmínek je 1 právě tehdy, mají-li proměnné hodnoty definované v odpovídajícím řádku.. Pro ostatní kombinace proměnných je podmínka rovna 0.

  Pro specifické vstupní hodnoty konjunkčních řádků, výstup logické funkce je potom 1 (protože disjunkce je 1, je-li alespoň 1 vstup roven 1).

- Jestliže výstup má být 0 odpovídající konjunkční podmínka se neobjeví v logické rovnici. Potom všechny existující konjunkční podmínky jsou 0 a výstupní proměnná bude také 0.

Poznámka:

- Tento algoritmus je také matematický důkaz teorému úplnosti, který říká, že každá kombinační logická funkce může být vyjádřena kombinací AND, OR a NOT.

- Takto je také možno definovat konjunkční normálovou formulaci logických funkcí (jednoduše aplikováním De Morganova teorému na konjunkční normálovou formulaci). Protože konjunkční normálová formulace je zřídka používána, nebude dále vysvětlována.

Zjednodušení logické funkce:

Zjednodušování funkce vzniklé z disjunktivní normálové formulace jsou obvykle zbytečně složité. Proto by měly být zjednodušeny aplikováním zákonů Boolovy algebry.

V našem případě:

$$Y = (\overline{A} \wedge B \wedge \overline{C}) \vee (A \wedge \overline{B} \wedge \overline{C}) \vee (A \wedge B \wedge \overline{C})$$

$$Y = \overline{C} \wedge \left[(\overline{A} \wedge B) \vee (A \wedge \overline{B}) \vee (A \wedge B) \right]$$

$$Y = \overline{C} \wedge \left[(\overline{A} \wedge B) \vee (A \wedge (B \vee \overline{B})) \right]$$

$$Y = \overline{C} \wedge \left[(\overline{A} \wedge B) \vee (A \wedge 1) \right]$$

$$Y = \overline{C} \wedge \left[(\overline{A} \wedge B) \vee A) \right]$$

$$Y = \overline{C} \wedge \left[(\overline{A} \wedge A) \wedge (B \vee A) \right]$$

$$Y = \overline{C} \wedge \left[1 \wedge (B \vee A) \right]$$

$$Y = \overline{C} \wedge (B \vee A)$$

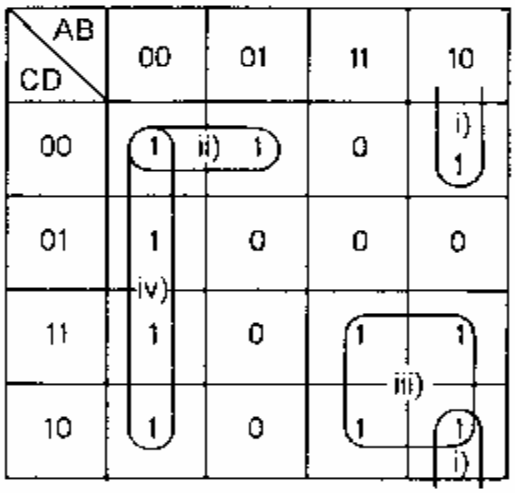
Výše uvedený algoritmus funguje dobře, ale je trochu zdlouhavý. Jak se dá zjednodušit přímo ? Tato otázka bude zodpovězena v příští kapitole.

Karnaughova mapa

Principielně algoritmus Karnaughovy mapy je založen na disjunktivní normálové formulaci. Jde pouze o chytrou metodu zapisování pravdivostní tabulky takovým způsobem, že možné zjednodušování logických rovnic lze jednoduše vyčíst z logického modelu.

Příklad:

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1



Algoritmus:

- Vstupní proměnné jsou zapsány do tabulky (půlka proměnných tvoří řádky a druhá sloupce). Sousední možné kombinace se liší pouze v proměnné (např. 00, 01, 11, 10).

- Potom se disjunktivní normálová formulace zapíše. V našem případě (pro oblast II) dostaneme následující výraz:

$$(\overline{A} \wedge \overline{B} \wedge \overline{C} \wedge \overline{D}) \vee (\overline{A} \wedge B \wedge \overline{C} \wedge \overline{D})$$

- Náš model je definován tak, že skupiny proměnných se liší pouze v jedné proměnné. Proto je možné použít distributivní zákon:

$$(\overline{A} \wedge \overline{C} \wedge \overline{D}) \wedge (B \vee \overline{B}) = \overline{A} \wedge \overline{C} \wedge \overline{D}$$

V tomto příkladě můžeme vidět obecné pravidlo Karnaughovy mapy:

- Jestliže v obdélníku nebo čtverci s 1, 4, 8, 16 … poli je výstupní proměnná vždy 1, konjunkci celé skupiny najdeme tak, že sepíšeme všechny vstupní proměnné, které mají stejnou hodnotu ve všech polích skupiny.

- Je třeba sestavit tolik skupin, aby každé pole, kde výstupní proměnná je 1 bylo zahrnuto alespoň v jedné skupině. V tomto kroku je možné sestavovat skupiny „přes okraj „ (např. spojování proměnných z prvního a posledního řádku).

V našem případě následující výsledek vznikne požitím skupin I.) – IV.) :

$$Y = (A \wedge \overline{B} \wedge \overline{D}) \vee (\overline{A} \wedge \overline{C} \wedge \overline{D}) \vee (A \wedge C) \vee (\overline{A} \wedge \overline{B})$$

Výsledek je už hodně zjednodušen zvláště v porovnání s úplnou disjunktivní normálovou formulací.

LOGICKÉ SKUPINY

- Základní logické funkce, odvozené logické funkce a složitější funkce jsou komerčně dostupné jako integrované obvody (IO).
  Spojením jednotlivých IO v elektronickém obvodu docílíme požadované funkce.

- Elektricky kompatibilní skupiny integrovaných obvodů jsou nazývány „logické rodiny“. Logické skupiny se liší v různých aspektech:

  - Pracovní napětí (ve většině případech 5 V, ale také je možné jiné napětí)
  - Rychlost změny
  - Příkon
  - Hustota integrace

- Příklady logických skupin jsou:

  - RTL (rezistorově-tranzistorová logika): již nepoužívaná
  - DTL (diodově-tranzistorová logika): již nepoužívaná
  - MOS (kov-kysličník-polovodič) logické skupiny:
    Výhody: nízký příkon, vysoká hustota integrace.
    Mnoho podskupin: PMOS, NMOS, CMOS,…
  - TTL (tranzistorově-tranzistorová logika): v současnosti nejvíce používaná.
    Výhody: levná, rychlá, snadno dostupná, standardizovaná
    Mnoho podskupin:
    - Standardní TTL
    - Nízko příkonová TTL (L)
    - Schottkyho TTL (S), rychlá, ale vyšší příkon
    - Nízko příkonová Schottkyho (LS), široké použití
    - Rozšířená nízko příkonová Schottkyho (ALS), široké použití

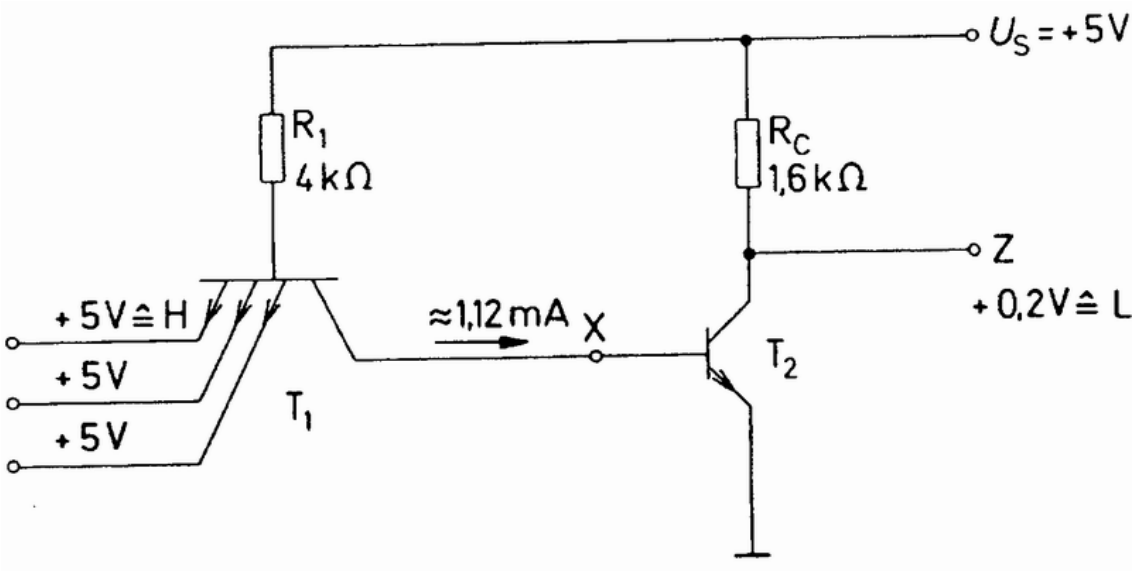
SKUPINA TTL

ELEKTRICKÉ VLASTNOSTI

Charakteristické vlastnosti:

- Pracovní napětí: +5 V
- Logické jednotky (členy) jsou realizovány pouze pomocí bipolárních tranzistorů.
- Diody jsou použity pouze pro posun napětí o 0.6 V, nebo jako ochrana před záporným napětím.
- Rezistory jsou použity pro dělení napětí a jako proudové omezovače. Jsou integrovány v obvodu.
- Vstupy do TTL obvodu jsou realizovány použitím více emitorových bipolárních tranzistorů.

Vstupní obvod (TTL)



- Jestliže jsou všechny vstupy připojeny na úroveň napětí (5 V), žádný proud nemůže protékat přes emitor tranzistoru T_1, ačkoli teče přes přechod kolektor-báze. Tento pracovní režim se u tranzistoru normálně nepoužívá. Nazývá se obrácený pracovní režim vstupního tranzistoru. Opačný proud teče do báze tranzistoru T_2 a otevírá ho. Výstupní napětí je poté dáno saturačním napětím T_2, které je přibližně 0,2 V.

- Jestliže je alespoň jeden ze vstupů připojen na potenciál země (0 V), tranzistorem T_1 protéká bázový proud. T_1 je proto plně vodivý a jeho kolektorové napětí je zmenšeno o saturační napětí 0.2 V. Proto je tedy tranzistor T_2 rozepnut a výstupní napětí bude přibližně 5 V.

- Výstup je v logické nule (L), jestliže všechny vstupy jsou v logické jedničce (H). Logická funkce v našem příkladu je hradlo NAND se 3 vstupy.

Poznámky:

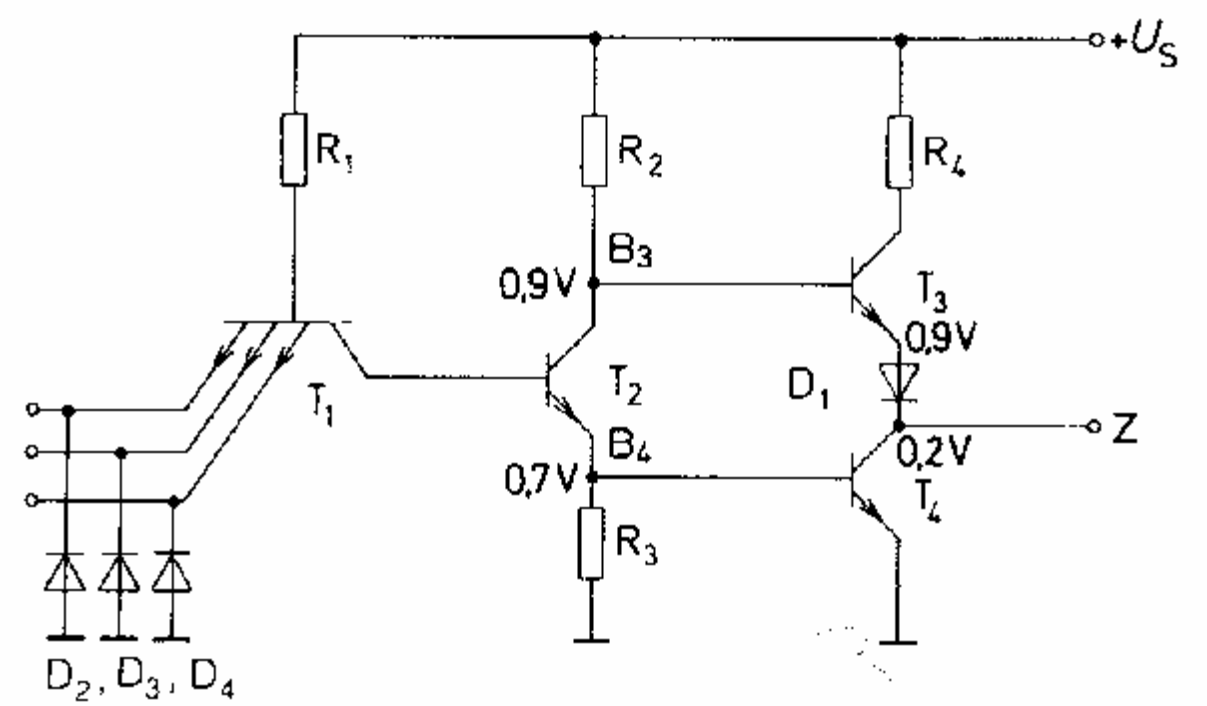
- Otevřený vstup TTL obvodu se chová tak, jako kdyby byl připojen na logickou jedničku (H). Není vhodné nechávat vstupy obvodu nezapojené, protože se tím může zvýšit citlivost obvodu na elektronický šum.

- Více-emitorový tranzistor může přepínat velmi rychle z normálního pracovního režimu do obráceného stavu. To je dáno tím, že není nutné odstranit elektronový náboj z oblasti báze, jako by to mělo normálně být, když je tranzistor rozepnut.

Výstupní Obvody (TTL)

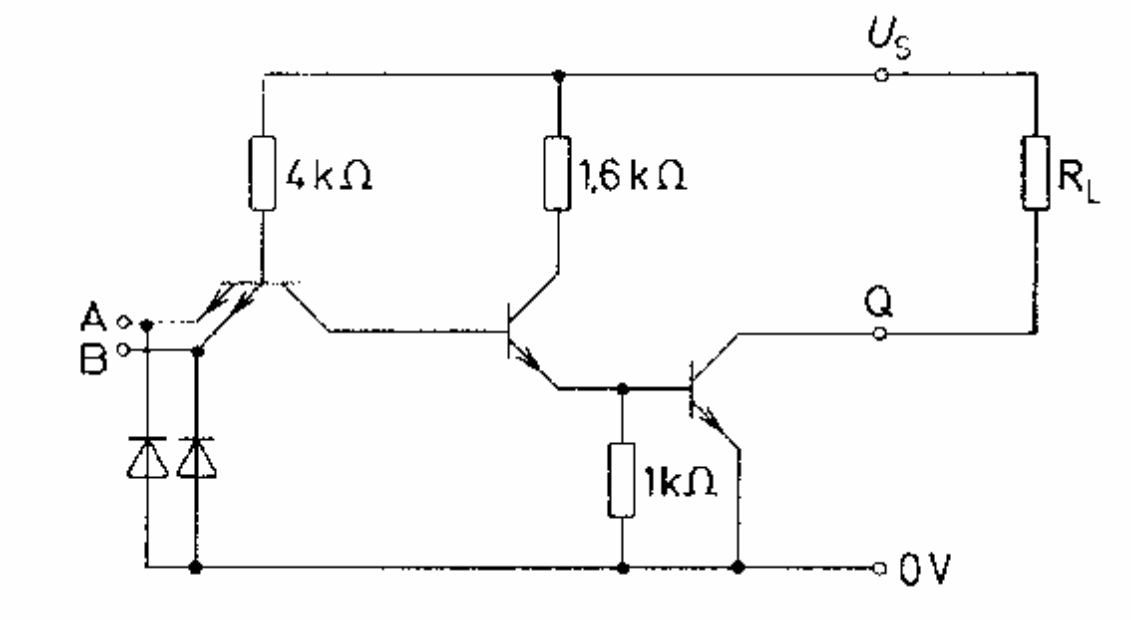
Výstupní obvod použitý v předcházejícím příkladu se příliš dobře nehodí v případě, jestliže výstup je připojen na vstupy dalšího obvodu. Jestliže jeho výstup je v úrovni H, výstupní proud teče přes kolektorový rezistor R_c a to způsobí snížení napětí, které nemůže být v mnoha případech tolerováno. Proto TTL obvody normálně používají následující výstupní obvod:

Push-pull output stage (totem pole output):



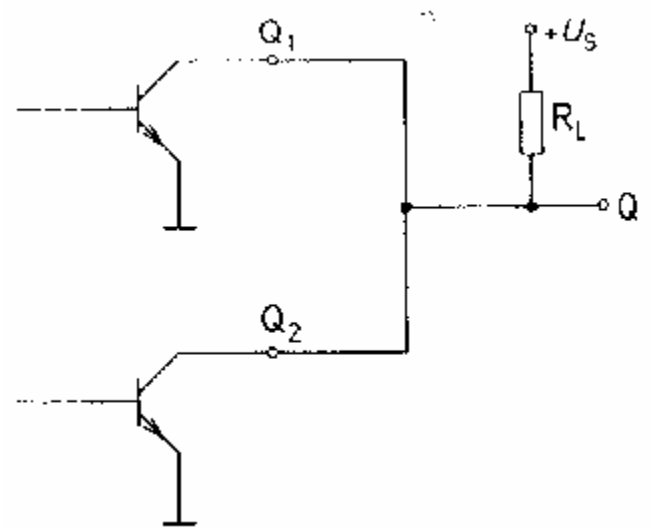
Jestliže výstupní úroveň je L, tranzistor T_3 je rozepnutý a T_4 je sepnutý. Proud z výstupu Z teče přes T_4 na zem. Jestliže výstupní úroveň je H, tranzistor T_3 je sepnutý a T_4 je rozepnutý. Výstupní proud potom teče přes rezistor R_4, přes T_3 a přes diodu na výstup Z. Od R_4 je nyní mnohem nižší než v předcházejícím příkladu „push-pull output stage“ může poskytnout vyšší výstupní proudy.

Výstup s otevřeným kolektorem



Tento druh výstupního obvodu používá externí kolektorový rezistor R_L. To je užitečné v následujících případech:

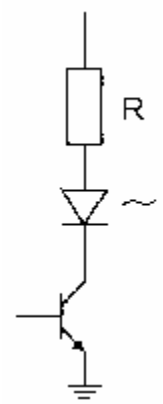
1. Spojení několika výstupů s jedním kolektorovým rezistorem.



Připojený výstup je H pouze jestliže oba tranzistory jsou rozepnuté, např. když oba obvody jsou ve stavu H. Jestliže používáme pozitivní logiku, toto zapojení představuje hradlo AND. Tento typ hradla AND s otevřeným kolektorovým výstupem je nazýván **vodičový AND**.

2. Spínání vnějších zařízení (relé, světlo emitující diody,…) přímo IO – např. výstup bez přídavných zesilovačů proudu (tranzistory).

   Příklad: Spínání světlo emitující diody (LED)



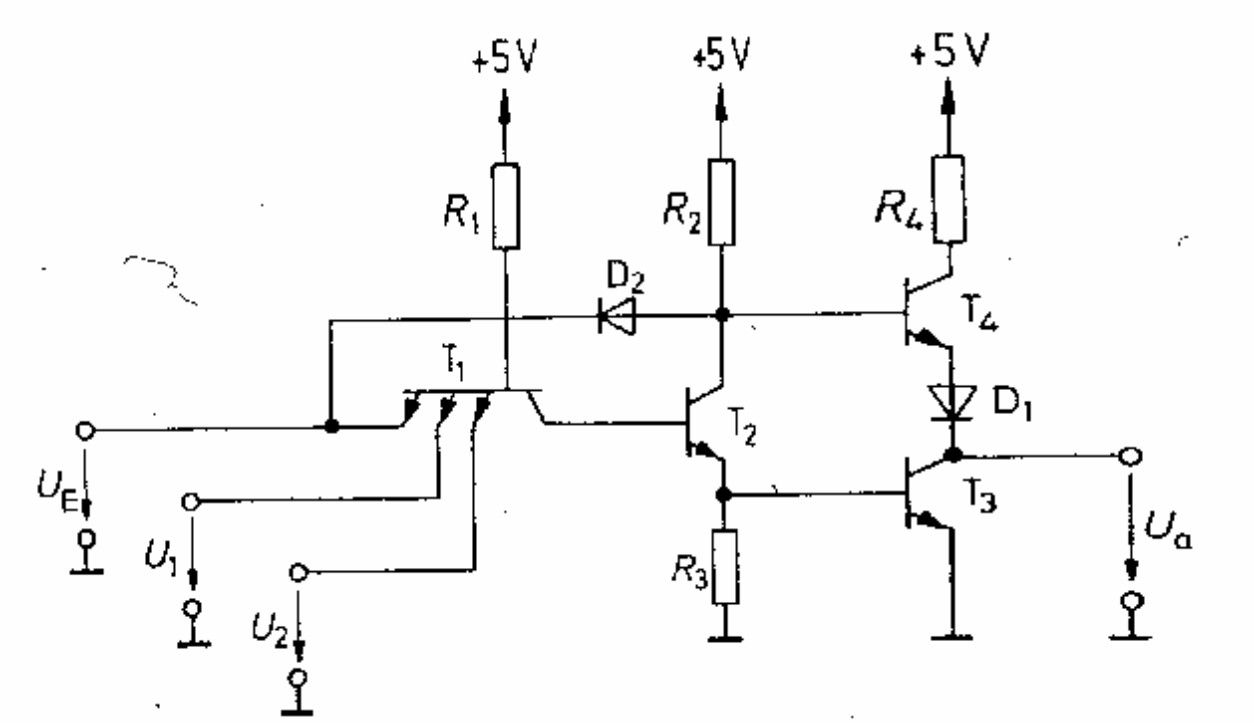
   Získání hodnoty rezistoru R:

   Typická světlo emitující dioda pracuje dobře při proudu kolem 5 mA (viz údaje o běžných LED diodách). Z diodové charakteristiky (závislost mezi napětím a proudem) můžeme vidět, že minimální napětí kdy LED dioda pracuje je v tomto případě 2 V. Navíc napětí na řídící tranzistoru s otevřeným kolektorovým výstupem je kolem 0.2 V (saturační napětí).
   Pro úroveň napětí 5 V je na rezistoru potom 5 V – 2 V – 0.2 V = 2.8 V.
   Z Ohmova zákona je hodnota R určena takto: R = 2.8 V / 5 mA = 560 Ω.
   Tato hodnota je obsažena ve standardní řadě hodnot rezistorů (řada R12).

Charakteristiky světlo emitujících diod používané v praktických cvičeních (naměřené):

Červená		Žlutá		Zelená	
U [V]	I [mA]	U [V]	I [mA]	U [V]	I [mA]
1.75	0.02	1.65	0.08	1.75	0.12
1.80	0.37	1.70	0.25	1.80	0.37
1.85	1.28	1.75	0.73	1.85	0.91
1.90	2.94	1.80	1.78	1.90	1.83
1.95	5.06	1.85	3.71	1.95	3.03
2.0	7.45	1.90	6.86	2.0	4.66
2.05	10.0	1.95	10.15	2.05	6.40

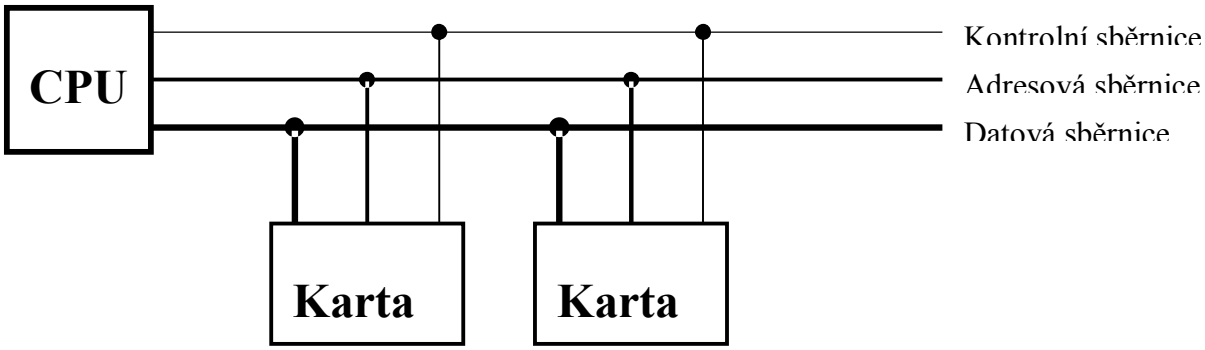
Třístavový výstup



Obr.9.35 TTL-Tristate-NAND-Gatter. Pro U_e=L je výstup indiferentní.

Třístavové spínací obvody mají normální dvojčinné (souměrné) výstupy, ale jsou doplněny jedním dodatečným vstupem. Jestliže umožní stav H (enabled), práce obvodu je stejná jako obvykle. Jestliže umožní stav L (disabled), zavřou se oba tranzistory (T_3 a T_4) do výchozího stavu. Na výstupu je potom mnohokrát větší odpor (>MΩ).

Třístavový spínací obvod se bude často vyskytovat ve výpočetní technice. V PC jsou všechny zástrčné karty paralelně spojeny s CPU.



Datová, dresová a kontrolní sběrnice se skládají mnoha elektrických zapojení (pro PC: 32 adresových zapojení, 32 datových zapojení). Zástrčné karty nesmí nikdy současně přepnout data na datovou sběrnici. Karta je spojena s datovou sběrnicí přes třístavově řazené vedení. Pouze tato karta dube vyzvána k přesunu dat, z CPU přes adresovou a kontrolní sběrnici, bude aktivovat třístavový převod a určí tím logický stav jeho vedení k datové sběrnici. Všechny ostatní karty zůstanou ve vysokém odporovém stavu.

Tento postup se jmenuje **multiplikace** dat. Všechny zástrčné karty, jsou přes selektivní datové vedení připojeny k CPU.

Základní údaje o skupině TTL

Příklad: NAND 74LS00 (Schottkyho nízko výkonová podskupina)

Operační podmínky:

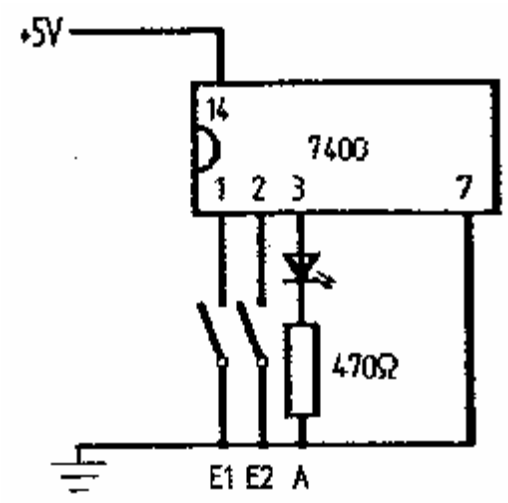
	Min.	Nom.	Max.	Jednotka
V_{cc} Napájecí napětí	4.5	5	5.5	V
I_{OH} Vysoká hladina výstupního proudu			-400	μA
I_{OL} Nízká hladina výstupního proudu			8	mA
Pracovní teplota	0		70	°C

Definice:

I_{OL} Nízká hladina výstupního proudu: to je maximální proud který můžeme naměřit na výstupu ve stavu L.

I_{OH} Vysoká hladina výstupního proudu: to je maximální proud který se vyskytuje při stavu H na výstupu.

Opatrně: 400μA není mnoho!
Příklad, jak se nedělá (z HPT, elektronický projekt):



Povolení: Součást použití otevřeného kolektoru, nebo stanovení proudového zesilovače tranzistorem.

Elektrické vlastnosti:

	Min.	Nom.	Max.	Jednotka
V_{IH} Vysoká hladina vstupního napětí				**V**
V_{IL} Nízká hladina vstupního napětí			0.8	V
V_{IK} Napětí vstupní svorce			-1.5	V
V_{OH} Vysoká hladina výstupního napětí	2.7	3.4		V
V_{OL} Nízká hladina výstupního napětí		0.25	0.4	V
I_{IH} Vysoká hladina vstupního proudu			20	μA
I_{IL} Nízká hladina vstupního proudu			-0.4	mA
I_{CC} Napájecí proud			4.4	mA

Definice:

V_{IH} Vysoká hladina vstupního proudu: Elektrické napětí bude nejnižší , pokud bude poznáno logické H.

V_{IL} Nízká hladina vstupního napětí: Elektrické napětí bude nejvyšší , pokud bude poznáno logické L.

V_{IK} Napětí vstupní svorce: Vstupní napětí na ochranné diodě.

V_{OH} Vysoká hladina výstupního napětí: Nejmenší dodávané výstupní napětí při stavu H.

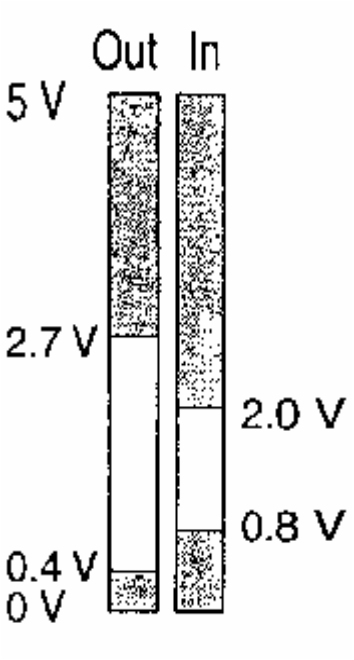
V_{OL} Nízká hladina výstupního napětí: Nejvýše dodávané výstupní napětí při stavu L.

I_{IH} Vysoká hladina vstupního proudu:Nejvyšší odebíraný vstupní proud při stavu H.

I_{IL} Nízká hladina vstupního proudu: Nejvyšší odebíraný vstupní proud při stavu L.

DŮSLEDKY:

Z napětí pro stav H a L (V_{OH}, V_{OL}, V_{IH}, V_{IL}) se dostal následující diagram.



- Každé uvolněné výstupní napětí je také platným vstupem.
- Ochranný odstup ve stavu L – je 0.4V
- Ochranný odstup ve stavu H – je přibližně 0.7V

Z výstupního a vstupního proudu (I_{OH}, I_{OL}) a vstupního proudu (I_{IH}, I_{IL}) je vidět:

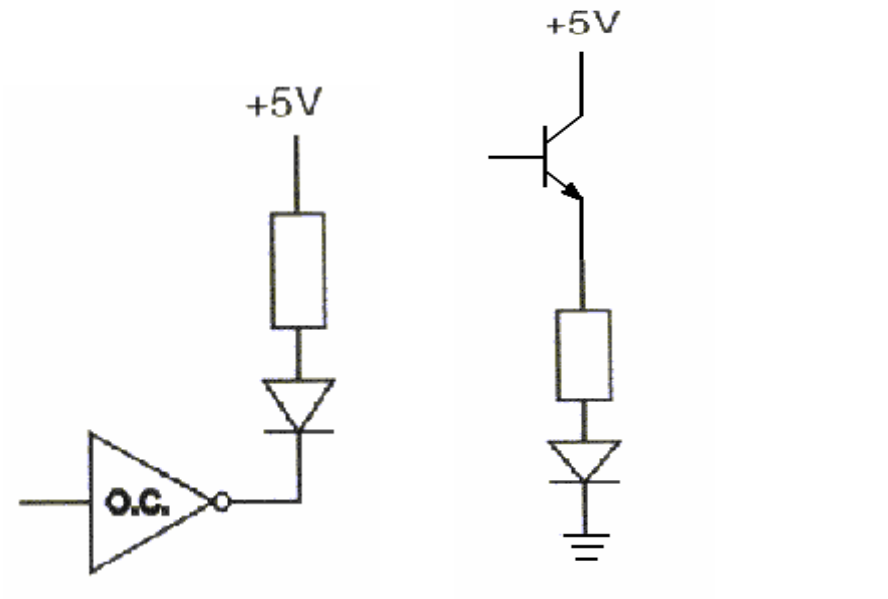
- Každý výstup může být spojen maximálně s 20 vstupy (LS-Serie)
- Poznámka: Při standardní sérii je pouze přípustné 10 vstupů.

Vypínací charakteristika:

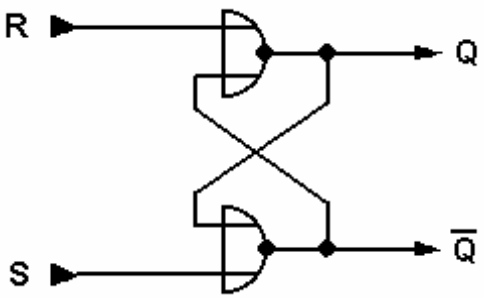
	Typ.	Max.
t_{PLH} L do H propagace přepnutí	9 ns	15 ns
t_{PHL} H do L propagace přepnutí	10 ns	15 ns

Cvičení:

1. Použijte hradlo NAND(74LS00). Potvrďte jeho pravdivostní tabulku a zobrazte stavy vstupu a výstupu pomocí svítivých diod (LED) k tomuto účelu použijte
   - (a) hradlo s otevřeným kolektorem (74LS05) nebo
   - (b) NPN tranzistor (emitorový sledovač)



2. Sestavte RS-klopný obvod pomocí hradel NOR (74LS02). Zobrazte stavy vstupu a výstupu pomocí LED diod. Zamyslete se nad funkcí tohoto elektronického obvodu



S	R	
0	0	předchozí stav
0	1	0
1	0	1
1	1	nedefinováno

Impuls na vstupu S může nastavit výstup Q klopného obvodu.
Impuls na vstupu R ho může vynulovat.RS klopný obvod je proto, jednoduchá paměťová buňka 1 bitové informace.

3. Vytvořte ekvivalentní hradlo (XNOR) za použití hradel NAND.
Najděte logickou funkci pravdivostní tabulky, výraz zjednodušte a realizujte za použití pouze hradel NAND.Invertor vytvořte propojením obou vstupů hradla NAND.

Pravdivostní tabulka:

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

4. Sčítačka se skládá z částí, které sčítají přesně 1 datový bit a vytváří sumační bitu S a bit přetečení C (přenos).Pravdivostní tabulka se nazývá „poloviční sčítačka“ a je definována takto:

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

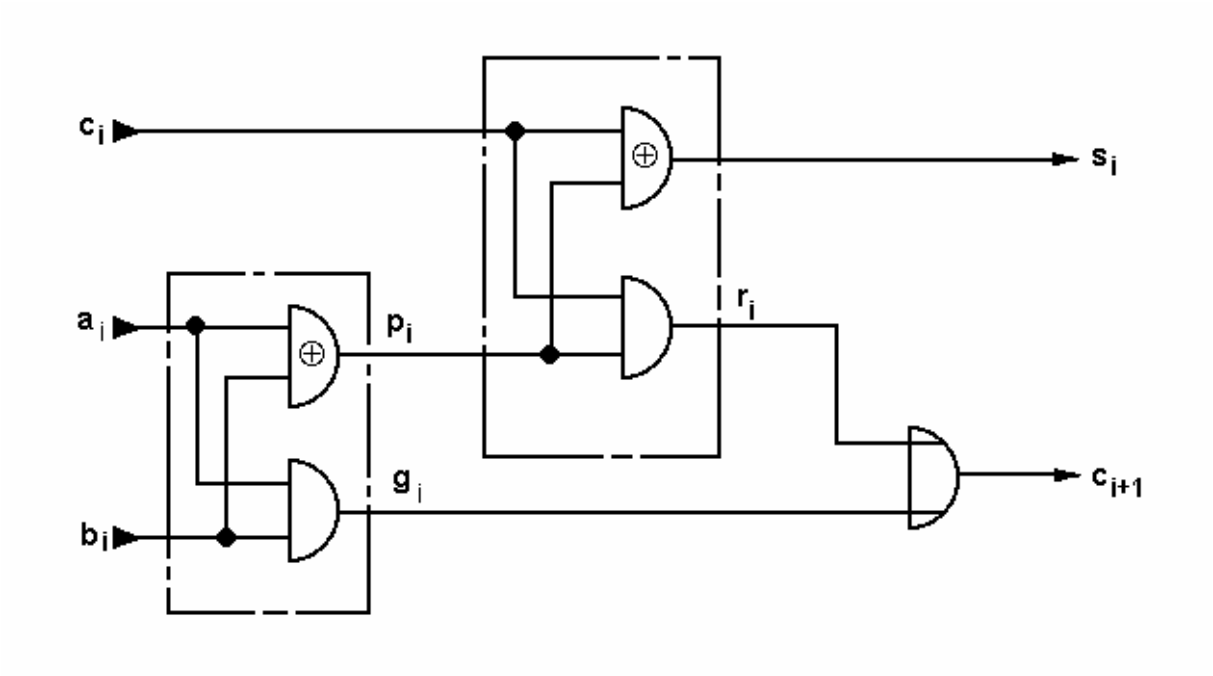
Najděte logickou funkci pro S a C a realizujte za použití minima hradel. Potvrďte pravdivostní tabulku.

Kromě předcházející „poloviční sčítačky“ bere „úplná sčítačka“ v úvahu předchozí (méně významný) bit (vst. přenos C_i).Stejně jako poloviční sčítačka Vytváří sumační bit S a bit přetečení (výst. přenos C_{i+1})

C_i	A	B	S	C_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Najděte logický výraz pro S a Ci+1 a sestavte úplnou sčítačku za použití minima hradel. Potvrďte pravdivostní tabulku.

Poznámka: Následující obvod je široce používán:



TTL TECHNIKA

Přehled

V současné době TTL skupina obsahuje více než 200 rozdílných elektronických Obvodů.Tyto obvody lze rozdělit podle jejich funkce:

- Hradla (AND,OR,NOT....)
- Data rozdělující obvody (multiplexory,dekodéry,buffry)
- Klopné obvody
- Paměti z klopných obvodů (registry,latche....)
- Aritmetické obvody (komparátory,sčítačky,posuvné registry.....)
- Čítače (binární,BCD....)
- Speciální účel (7-segmentový dekodér...)

Kromě toho je k dispozici ještě mnoho TTL kompatibilních IO

- Paměti
- Mikroprocesory
- Programovatelná logická zařízení(EPLD,CPLD,FPGA,....)

Další skupiny logických obvodů mohou být kompatibilní s TTL a to použitím vhodného konvertoru:

- CMOS – TTL konvertor
- ECL – TTL konvertor
- .....

Hradla

IO s hradly mají obvykle více než jedno hradlo(normálně 4-6)
Logická funkce:

- NAND
- AND
- NOR
- OR
- XNOR
- XOR
- NOT

Kromě toho jsou k dispozici další speciální vlastnosti:

- Více než dva vstupy
- Výstup s otevřeným kolektorem
- Vyšší proudové zatížení

Obvody distribuující data

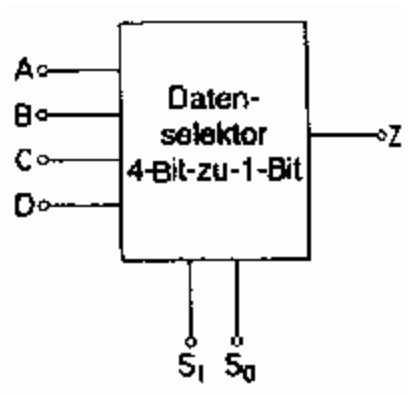
Účel:

- Propojení obvodů uvnitř desky tiskárny
- Přenos dat pomocí drátů a kabelů

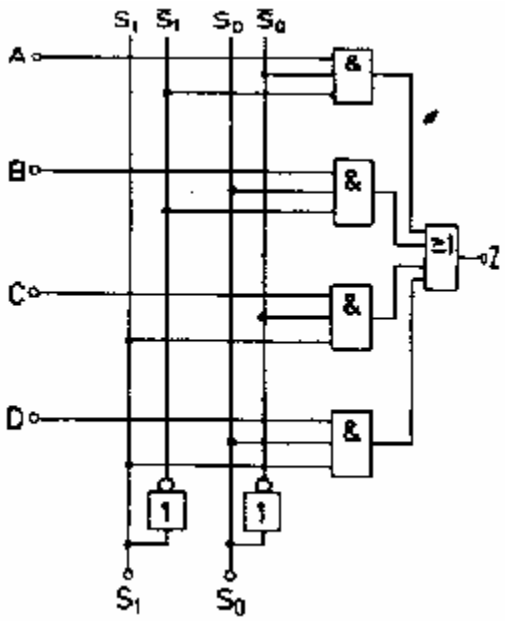
Multiplexer

Účel:
1 bitový multiplexor (vybírač dat) spojí přesně jeden z datových vstupů (obvykle 2,4,8...) s výstupem.Výběr se provádí pomocí výběrových vstupů (S0,S1).
Jednoduchý IO obsahuje více než jeden multiplexor, pak tato mnohonásobné datové bity mohou být přepínány současně.

Příklad: multiplexor 4 bity do 1 bitu



S1	S0	Z
0	0	A
0	1	B
1	0	C
1	1	D

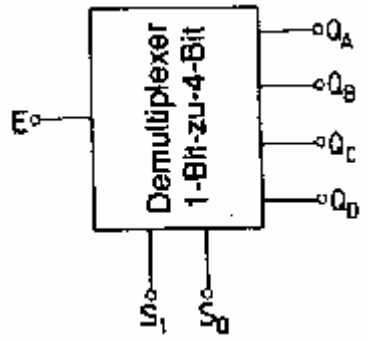


Poznámka:
Výběrové vstupy S1, S0 mohou být interpretovány jako binární reprezentace čísla. V desítkové soustavě jsou to 0,1,2,3. Toto číslo se nazývá adresa vybraného datového bitu.

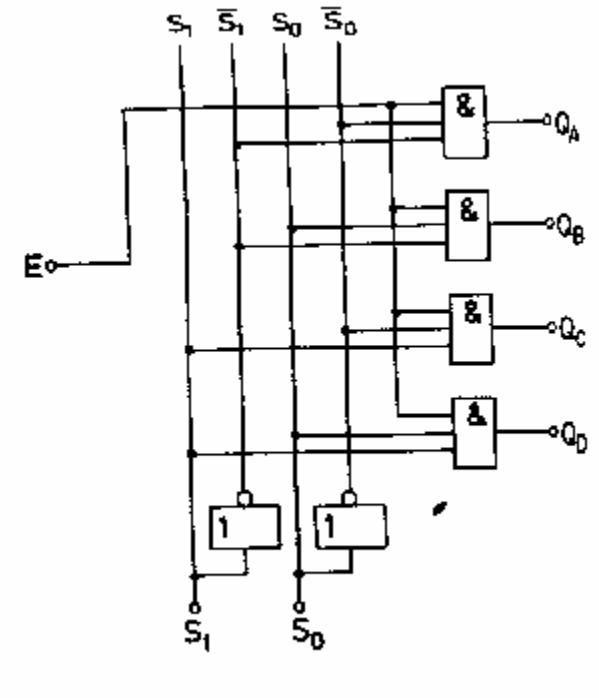
Demultiplexer

1 bitový demultiplexor spojuje vstup s několika výstupy. Výstup je vybrán pomocí určeného výběrového vstupu (S0,S1). V mnoha případech obsahuje jednoduchý IO více 1 bitových demultiplexerů.

Příklad: demultiplexer 1bit do 4 bitů

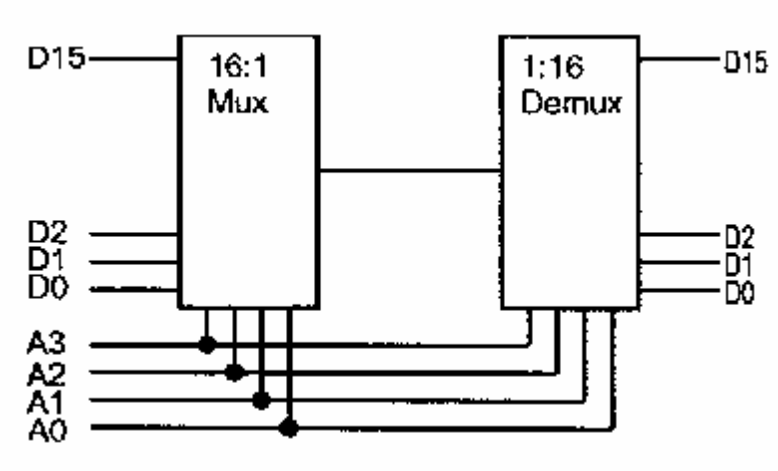


S1	S0	E =
0	0	Q_A
0	1	Q_B
1	0	Q_C
1	1	Q_D



Aplikace:

Přeneste 10 datových linek pomocí jednoho kabelu.

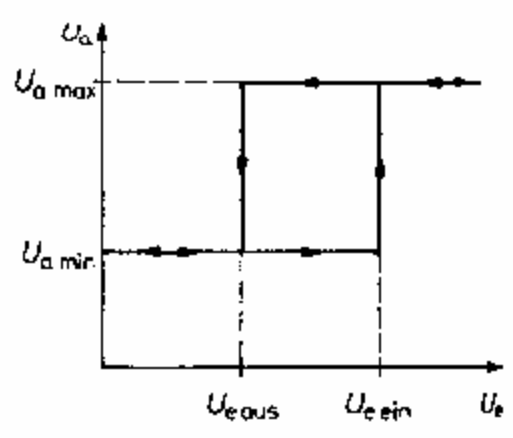


Když multiplexujeme nepoužijeme kabel, který potřebuje 16 drátů pro data a jeden drát pro potenciál země(17 drátů).
Když je použit 16:1 multiplexor (vysílač) a 1:16 demultiplexor(přijímač) je potřeba pouze jeden datový kabel, čtyři adresové dráty a jeden zemnící drát(6 drátů dohromady). Avšak data budou přenášeny sekvenčně a proto čas přenosu vzroste.

Oddělovač sběrnice

Cíl:

- Vyšší výstupní proudové zatížení než standardní TTL výstupy
  standart: 8 mA (L), 0,4 mA (H)
  typický oddělovač: (LS241): 24 mA (L), 15 mA (H)
- Kontrolovaný vzrůst a pokles násobeného signálu
- Třístavový výstup: pro sběrnicové systémy, multiplexory
- Výstupy Schmitova KO minimalizují přenos chyb (poruchový signál)

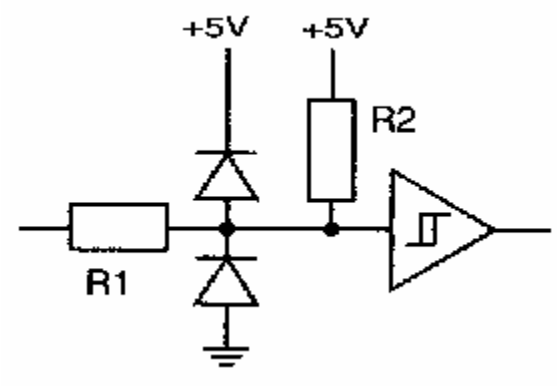


Metoda: Výstupní úroveň vzrůstá do H než vstupní napětí překročí prahové napětí U e ein. Když napětí klesne následně pod tuto úroveň po velmi krátký čas, mohl by být bez Schmitova obvodu na výstupu vytvořen poplachový signál (porucha).

- V Schmitově KO musí vstupní napětí poklesnout pod menší prahové napětí U e předtím než se výstup může vrátit do L. Proto se dosahuje lepší šumové imunity.
- Oddělovač je dostupný jako invertují i neinvertující obvod nebo s možností přenášení dat v obou směrech pro obousměrné sběrnicové systémy
- Mnoho bufrů poskytuje diodovou ochranu na vstupu, při připojení záporného napětí nezpůsobí zničení obvodu..
- Mnoho oddělovačů IO obsahuje 6 či 8 oddělovačů

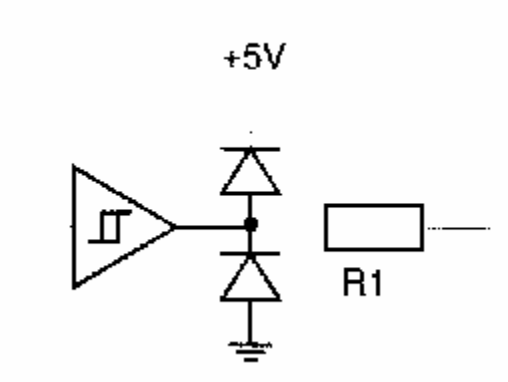
Cvičení:

1.vstup z konektoru



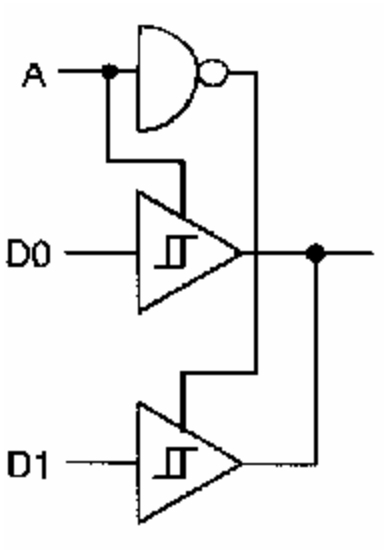
- Rezistor R1 omezuje vstupní proud jednou z diod je-li vstupní napětí > 5V nebo < 0V
  Rada: R1=100 Ω. Pokles napětí na odporu R1 nemůže být větší než 0,2 V
- Rezistor R2 udržuje vstupní napětí na přesně definované úrovni H
  Rada: R2=10 Ω. Pokles napětí na odporu R2 nemůže být větší než 0,2 V
- Není zaručen přechod na úroveň L protože I_{IL} je o mnoho větší než I_{IH}.Proto se požadují pull-down rezistory(raději menší), které by měli způsobit větší vstupní proud když je vstupní napětí na úrovni H
- Zvýšená šumová imunita způsobená vstupní charakteristikou Schmitova KO
- Diody se požadují pouze nejsou-li v oddělovači
- Když je oddělovač třístavový všechny stavy jsou možné

2.výstup na konektor



- Diody ochrání výstup když je napětí zdroje chybně připojeno
  Rezistor R1(50-100Ω) omezuje proud.
- Oddělovač poskytuje větší výstupní proud než TTL výstupy. Avšak možný napěťový pokles musíme brát v úvahu.
- Rezistor R1 je použit na vyrovnání výstupní impedance obvodu s impedancí připojeného koaxiálního kabelu. Když není vyrovnání správné může nastat odraz signálu.

3. Multiplexor pomocí třístavového oddělovače



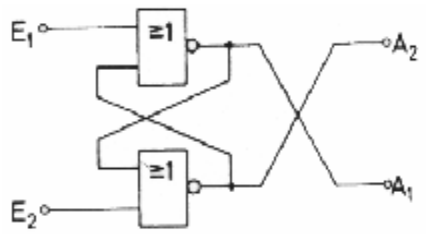
Metoda:
Pouze jeden z dvou třístavových oddělovačů je provozován. Přepínací signál je adresový vstup A.

Klopné obvody

Klopné obvody jsou bistabilní klopné obvody.Mohou uchovávat logický stav.

Známe už jeden z těchto obvodů:

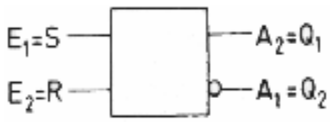
RS Klopný obvod z hradel NOR:



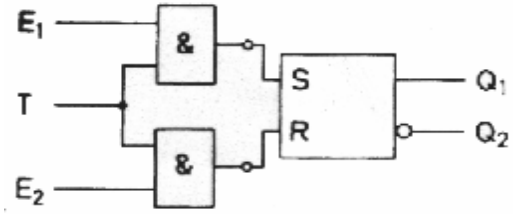
E_1 (S)	E_2 (R)	A_1 (Q_2)	A_2 (Q_1)	
0	0	A1	A2	Pamatuje
0	1	1	0	Reset
1	0	0	1	Nastavení
1	1	X	X	Zakázaný

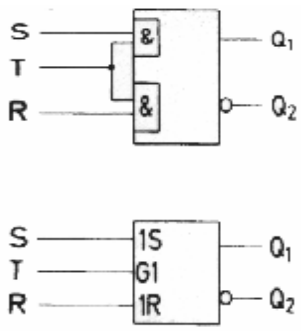
Tento klopný obvod je bez časového impulsu, data na vstupu E1 a E2 způsobí okamžitou změnu na výstupu.

Značka:



RS- klopný obvod s časovým impulsem
Kdykoliv logický stav muže být změněn pouze když je další signál T aktivní(H). Toho lze docílit když je impuls spojen s oběma vstupy hradel AND:





U tohoto typu klopného obvodu nelze změnit výstupní stavy když časování T je L, protože také R a S jsou L v tomto případě.

Tento obvod se nazývá RS klopný obvod citlivý na hodinový impuls a je členem skupiny synchronních klopných obvodů.

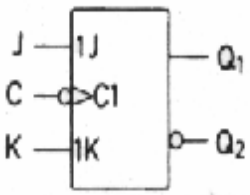
Hranou spustitelné klopné obvody

Tyto klopné obvody se použijí když musí být logický stav signálu zachován po přesně určený čas.Tyto obvody si pomatují vstupní stavy během změny hodinového impulsu z L → H (nebo z H→L). Proto se nazývají jednoduché. Kromě toho jsou také klopné obvody které uchovávají vstupní stav vnitřně na kladnou hranu(L → H) časového impulsu a převede tento vnitřní stav na výstupy s následující sestupnou hranou (H→L).Tento obvod se nazývá dvojitý klopný obvod, impulsový klopný obvod, master-slave klopný obvod.

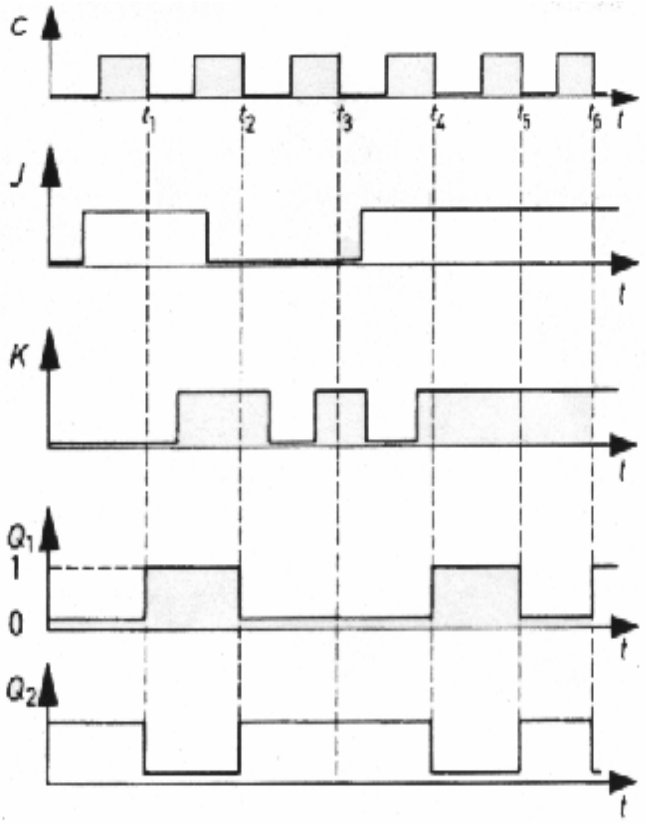
- Klopný obvod hranou spustitelný má široké uplatnění v základních pamětech digitální techniky
- Jelikož jsou logické výstupní stavy přesně definovány během přechodu časové hrany,jsou důležité některé podmínky uvedeny v následujících bodech:
  - Čas změny časového impulsu (L → H nebo H→L) musí být dost malý(několik ns)
  - Vstupní signály musí být stabilní několik ns před hodinovým impulsem(Set up) a musí být stabilní několik ns po(Hold)
- Když tyto podmínky nejsou splněny výstupní stav není definován a je nastavena náhodná úroveň.

JK klopný obvod

Na tomto příkladu si pamatuje vstupní úroveň během záporné hrany(výstupní úroveň,H→L) Jsou také obvody s kladným časováním(vedoucí hrana,L→H)

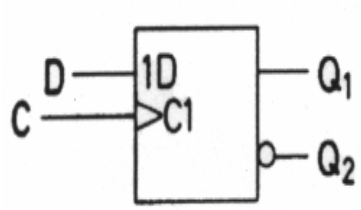


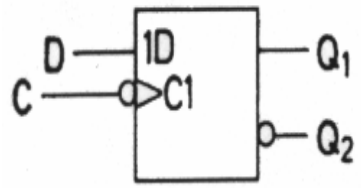
	t_n		t_{n+1}	
Fall	K	J	Q_1	Q_2
1	0	0	Q_{1n}	Q_{2n}
2	0	1	1	0
3	1	0	0	1
4	1	1	$\overline{Q}_{1n}$	$\overline{Q}_{2n}$



D klopný obvod

D klopný obvod pamatuje vstupy během časové hrany.Může být sestaven z JK-klopného obvodu nastavením $K=\bar{J}$





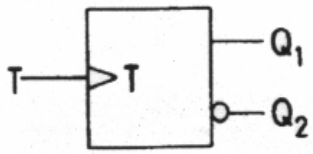
	t_n	t_{n+1}
Fall	D	Q_1
1	0	0
2	1	1

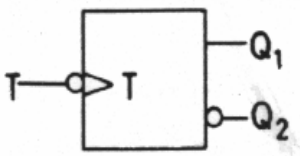
Užití:

- Datové paměti: statické paměťové obvody skládající se z mnoha D klopných obvodů.
- Čítače,posuvné registry a mnoho dalších obvodů se skládá z D klopného obvodu nebo z jiného klopného obvodu jenž je od D odvozen

T klopný obvod

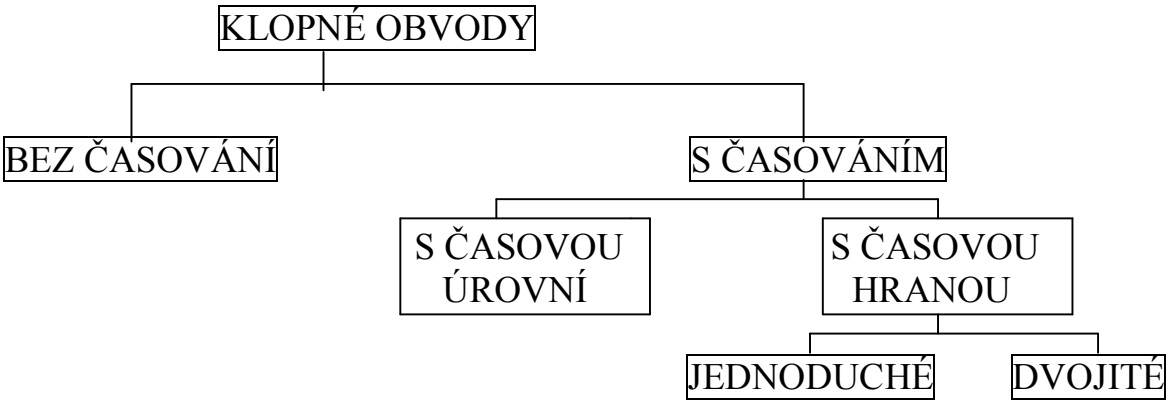
Tento klopný obvod mění výstup s každou hranou časování.Může být sestaven z D-klopného obvodu když je vstup D spojen s výstupem Q_2





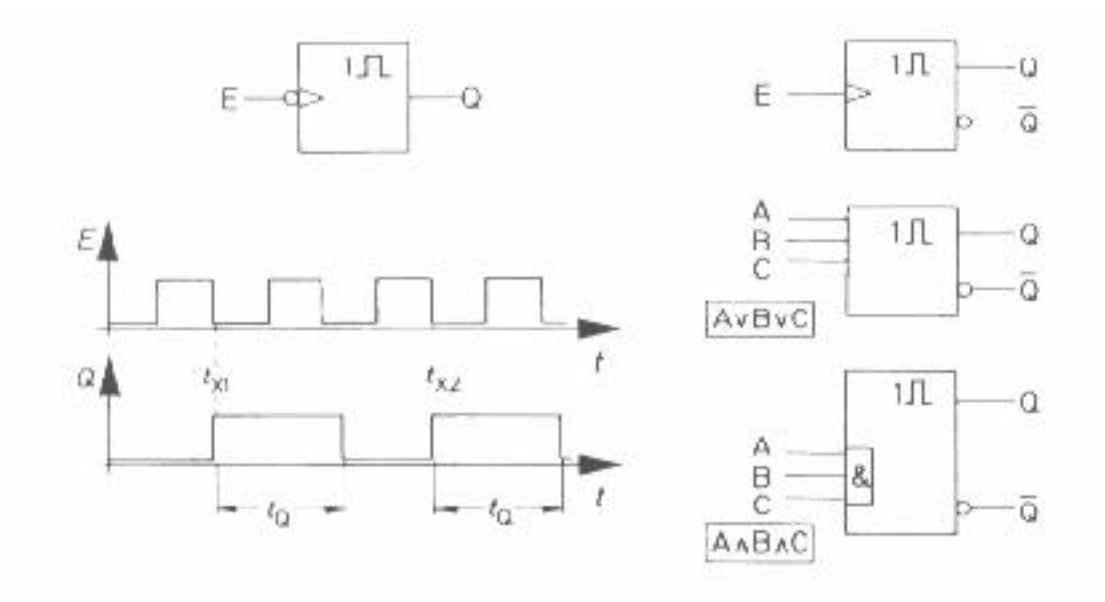
	t_n	t_{n+1}
Fall	Q_1	Q_1
1	0	1
2	1	0

Typy klopných obvodů (Přehled)

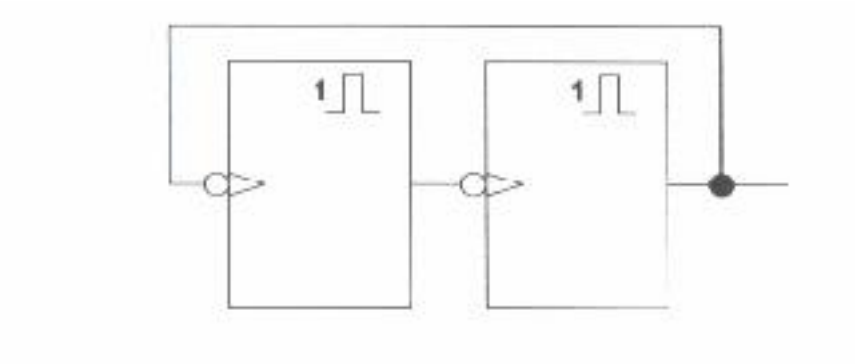


Monoflopy

Monoflopy (monostabilní klopné obvody) nepatři do skupiny dvoustabilních klopných obvodů, které jsme měli doposud.
Výstup monoflopu je normálně v logické úrovni L. Když je monoflop stopnut výstup jde na H se zamčenou hranou, setrvává tam nějaký čas T a padá potom zpátky na L. Čas T muže se přizpůsobit , když používá vnější rezistor a vnější kondenzátor.
Dodatečný hodinový impuls během času T (kdy vystup je H) je bud ignorován (non-retriggerable monnoflops) nebo čas T je prodloužený (retriggerable monoflops).



Použití:
Vývojový stupeň (generovaní) hodinového impulsu s regulovanou frekvenci.



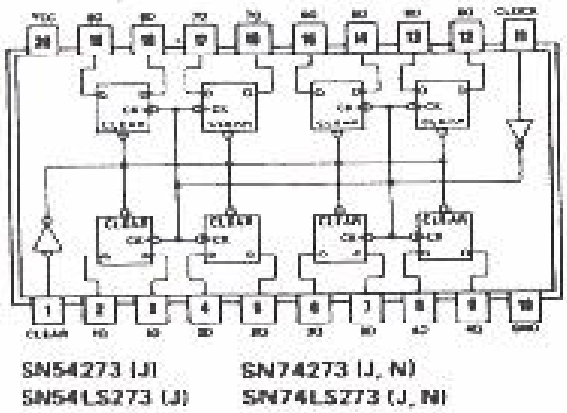
Paměťové obvody

Paměťové obvody skupiny TTS jsou zhotovené z klopných obvodu. Mají paměťovou kapacitu jenom 8 bitů. Větší paměťové obvody se popisují v následující kapitole.

Registry

Seznamy se skládají z klopných obvodů (obvykle 4,8). Navíc na vstupu dat a výstupu mají běžný výstupní hodinový impuls a dodatečně vstup pro čištění klopných obvodů.

Ku příkladu: LS273



Aritmetické obvody.

Lineární čísla (krátké objasnění):

- V digitální elektronice binární reprezentace čisel se většinou používá výlučně.
- Kladná čisla jsou kódovaná přimo do systému binárních čisel
  Příklad (velikost je 8 datových bitů).
  25(desítková soustava) = $1x2^4 + 1x2^3 + 0x2^2 + 0x2^1 + 1x2^0$ = 00011001
  Čísla od 0 do 255 (desítkové soustavy) mohou být znázorněny s použitím 8 bitů.
- Kladná čisla mohou také být znázorněny kódováním každého desítkového čisla jako 4 bitové binární číslo. Tento typ kódování se nazývá BCD (binárně kódovaná čísla v desítkové soustave).
  Priklad: 516: 1001 0001 1100

  Toto kódové schéma je dobře provedeno, aby ukázalo čisla na obecném 7 číslicovém displeji.

- Záporná čísla mohou být kódována s použitím doplňku druhé označovací soustavy. Toto kodovací schéma používá další pravidla:
  - Pro kladná čisla doplněk druhého znaku je stejný s binarním kódováním.
  - Pro záporná čísla se za prvé získá binární reprezentace absolutní hodnoty. Potom se doplní počet (například 0 je nahrazena 1, 1 je nahrazena 0, a naopak) a konečně je přidána 1.

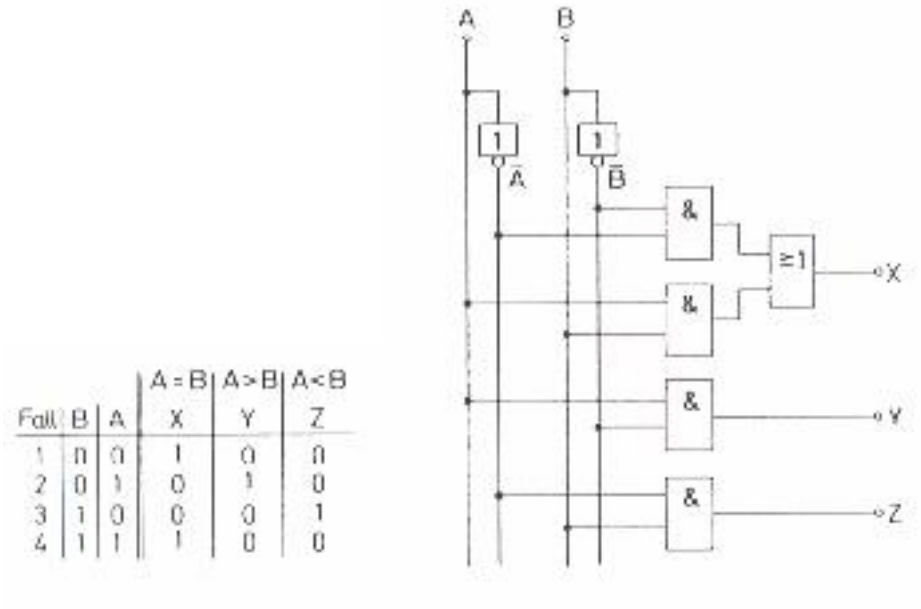
    Příklad: -13
    Binární znazornění +13 je: 00001101
    Doplněk : 11110010
    Plus : 00000001
    Výsledek : 11110011

- Čisla od –128 (1000000) do 127 (01111111) mohou být znázorněny pomocí použití druhého doplňkového kódování s velikostí 8 bitů.
- Velice významný bit druhého doplňkového počtu ukazuje jeho znaménko (1-negativní, 0-pozitivní)
- Důvodem a výhodou druhého doplňkového kódovaní je že odčítáni dvou čisel muže být zenšeno k sčítání.
  Příklad : 17-13=17+(-13)
  17 : 00010001
  -13 : 11110011
  suma : (1)00000100
  Suma se rovná +4, jestliže nadbyteční bit je vynechán.

Komparátor

Porovnávací jsou elektronické obvody, které mohou srovnávat velikost dvou binárně kódovaných čísel A,B. Porovnávací mají 3 výstupy
A=B, A>B a A<B

Příklad: 1 bitový srovnávací obvod.



			A = B	A > B	A < B
Fall	B	A	X	Y	Z
1	0	0	1	0	0
2	0	1	0	1	0
3	1	0	0	0	1
4	1	1	1	0	0

Spolu se skupinou TTL jsou také dostupně porovnávací pro více než jeden bit (příklad pro 4 bitová binární čísla), které mohou být kaskádově uspořádaná pro dokonce větši velikost (příklad LS85). Výstupy A>B a A<B mohou se použít pouze pro binárně kódovaná čísla, NE pro dvojitě doplňkové kódování.

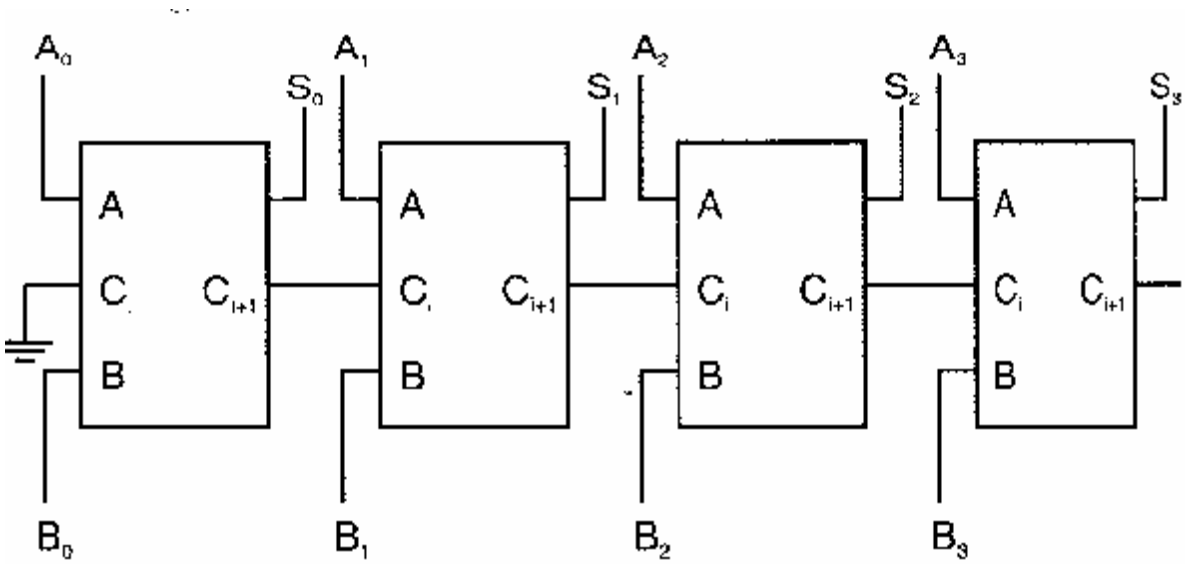
Sčítačka
Sčítačka se postaví dohromady z 1-bitových celosčítaček.

C_i	A	B	S	C_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Přitom C_i znamená přenos od předešlého místa , C_{i+1} přenos na další místo .
Kaskáda se nastane přesně jako u normálního počítání.

0110 + 0011

Pro tento výpočet potřebuje kaskádu 4 1-bitových celosčítaček . Nižší hodnotu sčítá A_0 , vyšší hodnotu A_3.

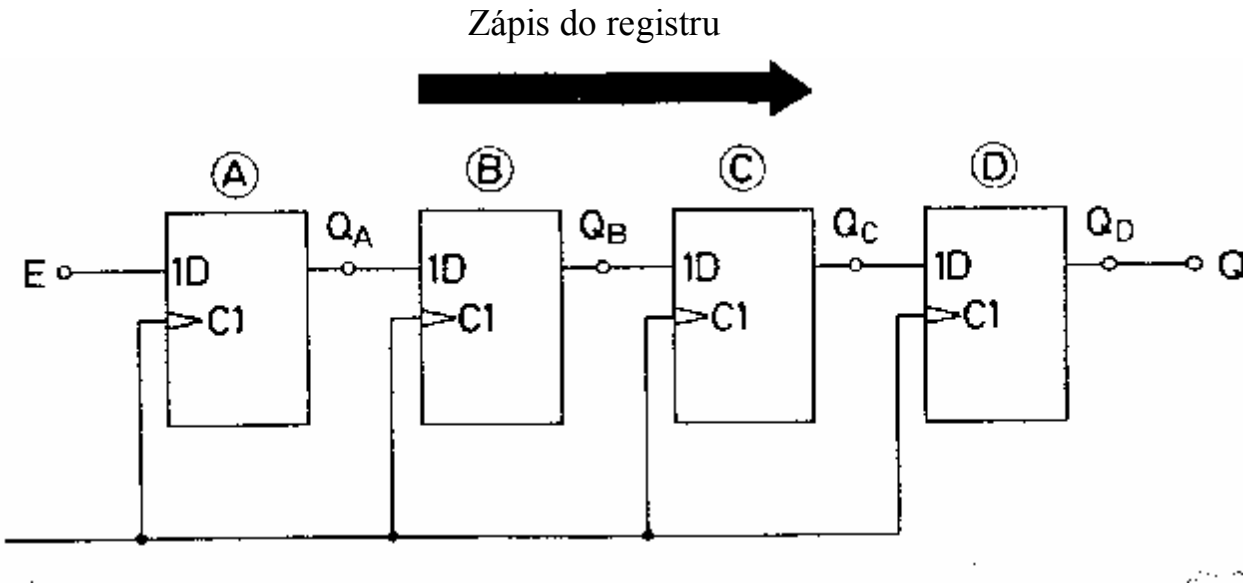


A0 : součet 0 + 1, $C_i = 0$, výsledek : 1, $C_{i+1} = 0$ (žádný přenos)
A1 : součet 1 + 1, $C_i = 0$, výsledek : 0, $C_{i+1} = 1$ (přenos)
A2 : součet 1 + 0, $C_i = 1$, výsledek : 0, $C_{i+1} = 1$ (přenos)
A3 : součet 0 + 0, $C_i = 1$, výsledek : 1, $C_{i+1} = 0$ (žádný přenos)
Výsledek tedy zní : 1001 žádný přenos (C_{i+1} od A3)

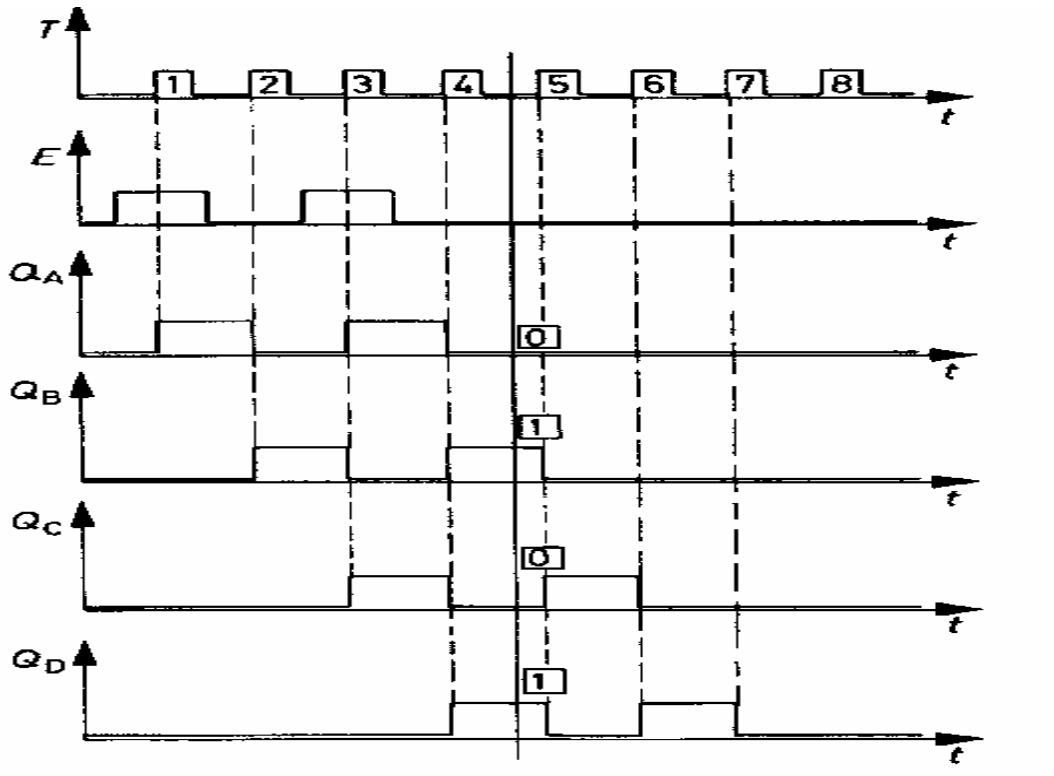
Zápis do registru

Zápis do registru záleží na několika klopných obvodech.

Princip jednoduchého zapisovacího registru (4 Bity , sériový vstup E, sériový výstup Q) :



Na jeden takt bude obsah zapisovacího registru posunut o jedno místo doprava.



Komplexní zapisovací registr má doplňkové schopnosti :

- Nastavení počátečního stavu (paralelní přístup)
- Výstup od všech klopných obvodů (paralelní výstup)
- Posunutí do leva nebo doprava

Použití:

- Paralelní přístup, dále posuv : Paralelně - sériová přeměna (např. u posílání přes sériové rozhraní COM1 , COM2 na PC).

- Sériový vnitřní posuv , dále paralelní výstup (např. přijímání přes sériové rozhraní COM1, COM2 u PC).

- Paralelní přístup bin. čítače, dále posunutí o 1 místo do leva (např. 00110101 ->01101010) Tato operace znamená pro jedno bin. číslo násobení 2. Dohromady se sčítání můžeme tvořit součinnými obvody:

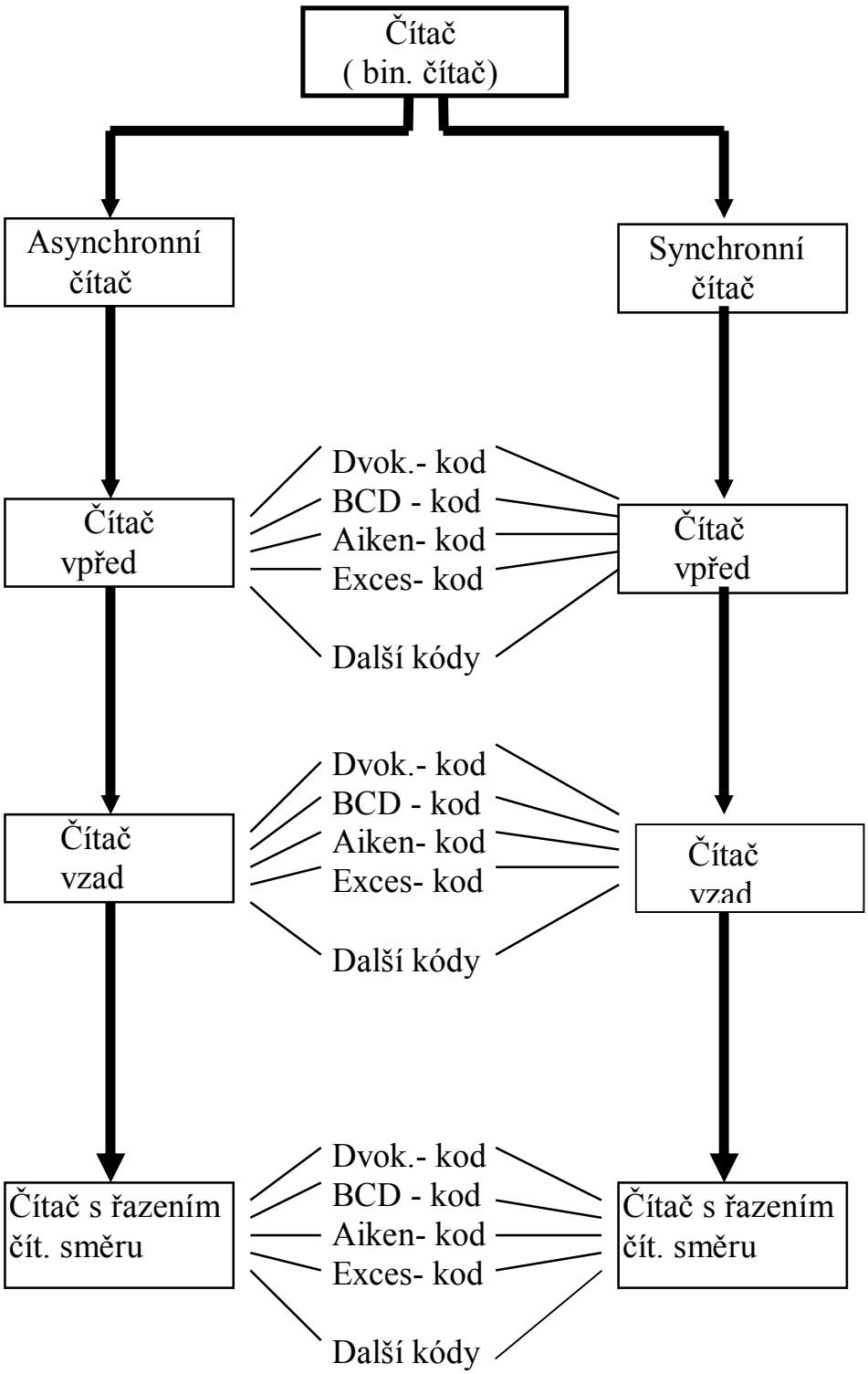
```
0011 x 0010
-----------

0000
 0000
  0011
   0000
-----------

0000110
```

- Paralelní přístup bin. čítače , dále posun o 1 místo doprava (např. 00110101-> 00011010) tato operace znamená celočíselné dělení přes 2.

Čítače

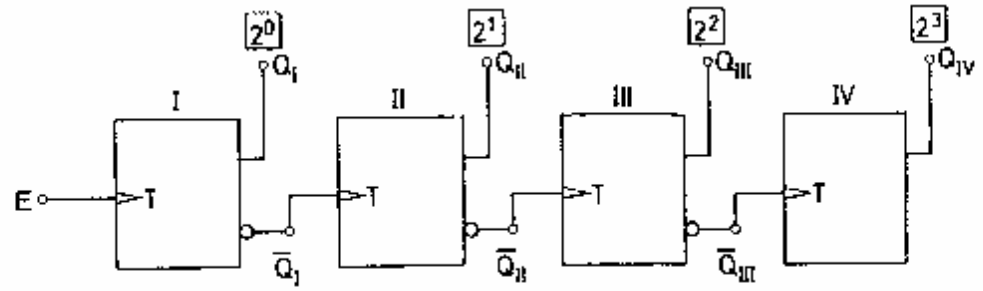
Čítače jsou obvody, které počítají taktové impulsy . Jejich výstupy dávají 1 bin. číslo, které je stejné jako počet taktovacích impulsů.

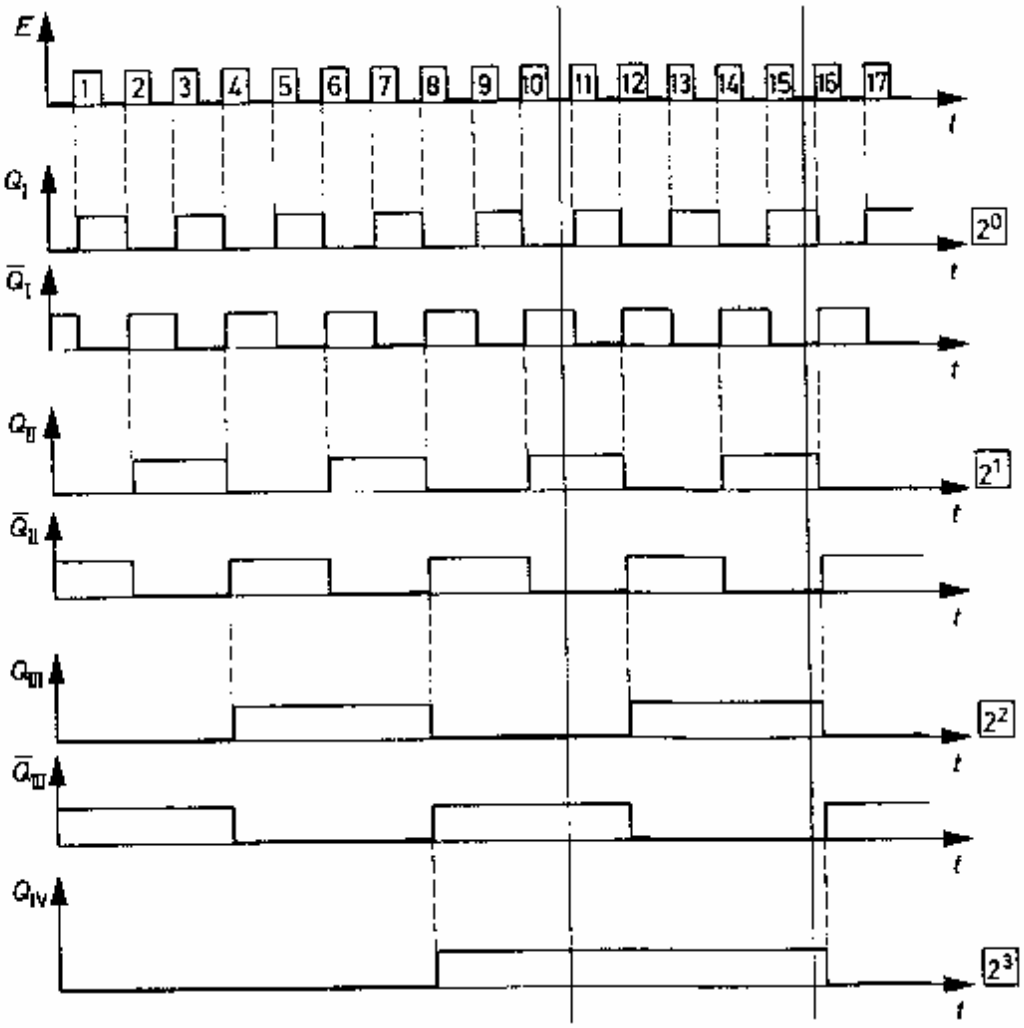
Čítač bude rozlišovat směr čítání a použité kódy.



Asynchronní čítač

Asynchronní čítač je tvořen klopnými obvody T . K připomenutí: klopné obvody T změní svůj stav při každém taktu.





po 10 impulsech | po 15 impulsech

1010 | 1111

Z toho vyplívá mnoho závěrů:

- Čítání dopředu nebo zpět
- Paralelní přístup pro vložení počáteční hodnoty
- Čítání v rozdílném kódování

Použití:

- Čítání impulsů
- Dělení frekvence: Bude-li čítač taktován s konstantní frekvencí, tak na prvním výstupu bude poloviční frekvence, na dalším 1/4 atd.
  Jestli že byla dosažena 0, čítač začíná opět čítat , dělení se může jakkoli přizpůsobit.

Poznámka:

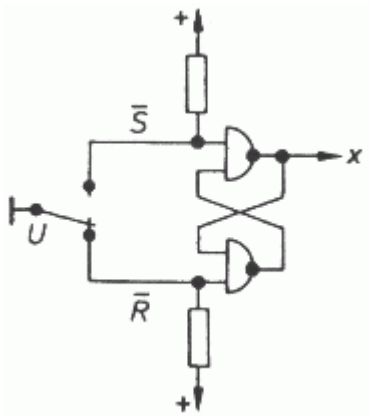
- Asynchronní čítač má jednu závažnou chybu: Když bude klopný obvod stále od předchozího taktován, trvá to při velkém čísle dosti dlouho, až je konečně přijmout nejvyšší Bit. Proto je možné čítat frekvenci jen omezeně.
  Synchronní čítač taktuje všechny klopné obvody současně a používá dodatečnou logiku , aby přenos předcházel (přenosové vyhledávání). Proto se může dosáhnout hlavně vyšší čítací frekvence.

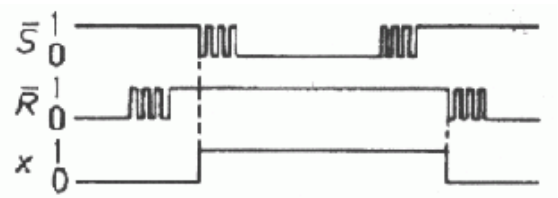
Cvičení:

4 bitový binární čítač s LED displayem

Použijte binární čítač (LS93) a vygenerujte hodinový impuls pomocí tlačítka. Když jsou mechanická tlačítka zmáčknuta nebo puštěna,jejich kontakty se mohou sepnout vícekrát za sebou.Toto chování se nazývá “zákmit“ mechanického kontaktu.Proto se musí na vygenerování jednoho dobře rozeznatelného hodinového impulsu použít “nekmitající“ obvod představený dole na obrázku.Binární výstup tohoto čítače může být zobrazen pomocí LED diod.

Nekmitající obvod (NAND LS00):

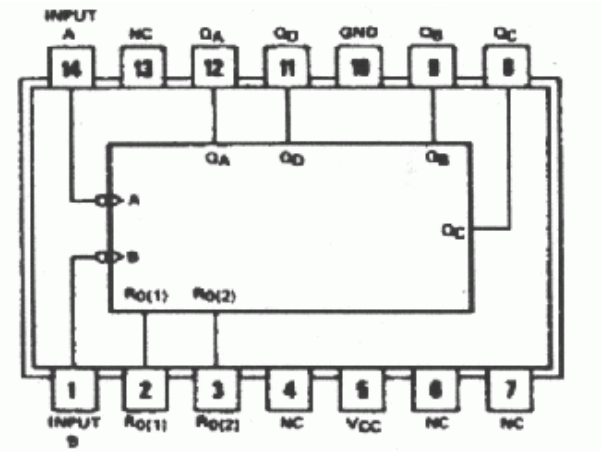




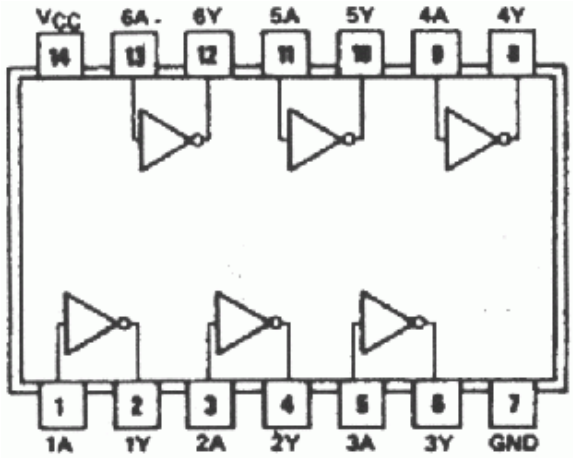
Čítač (LS93):

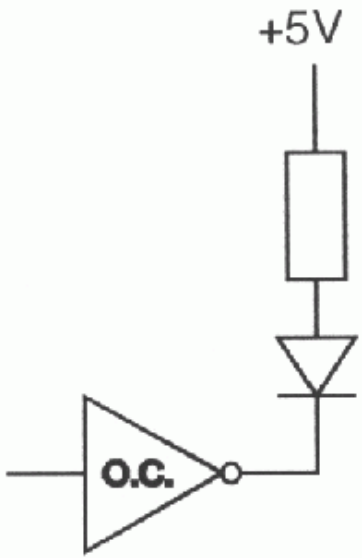
Připojení:

- Řídící napájení (5) na +5 V
- Zem (10) na 0 V
- Q_A (12) na B (1)
- Reset (2,3) na zem
- Vstup A (14) na výstup nekmitajícího obvodu.



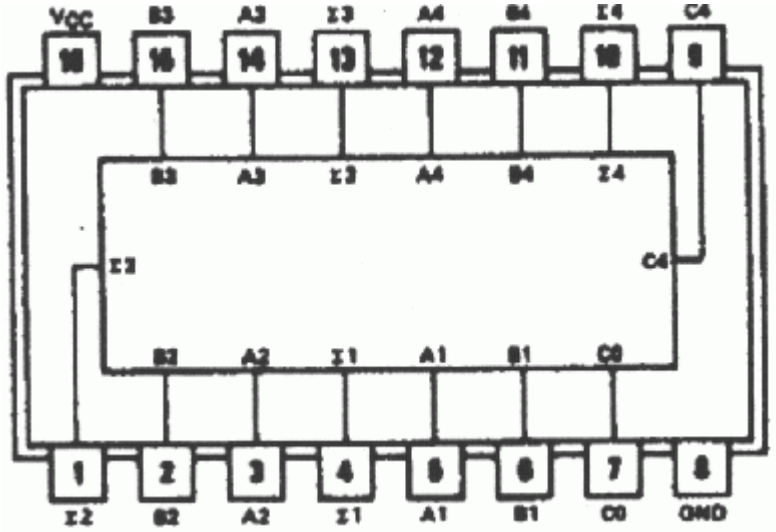
LED indikátory (LS05 invertory s otevřeným kolektorem)





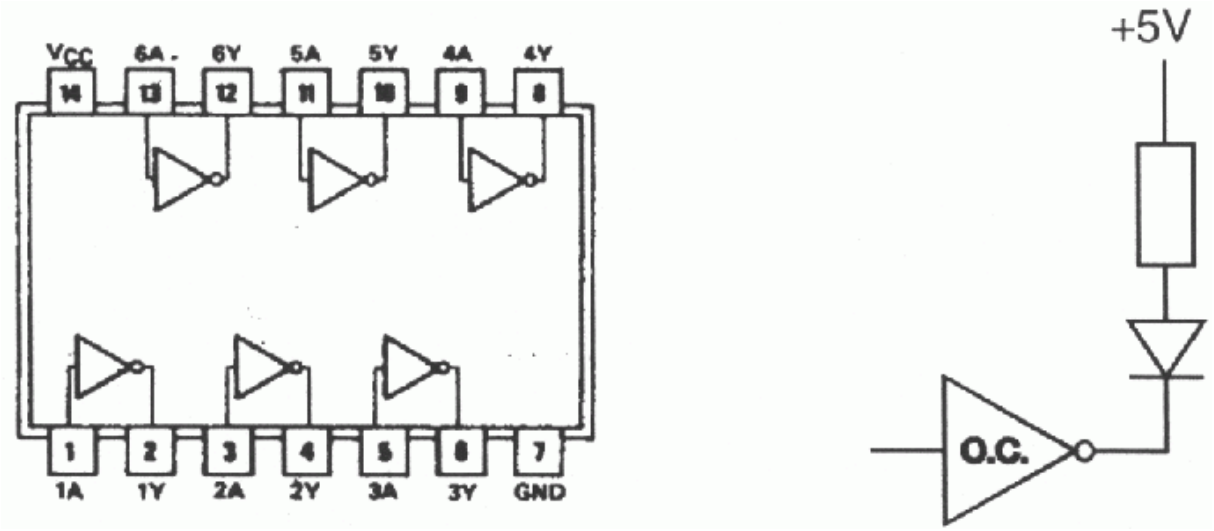
Sčítač s LED displayem

Použijte 4 bitový sčítač LS283.



SN54283 (J,W) **SN74283 (J,N)**
SN54LS283 (J,W) **SN74LS283 (J,N)**
SN54S283 (J) **SN74S283(J,N)**

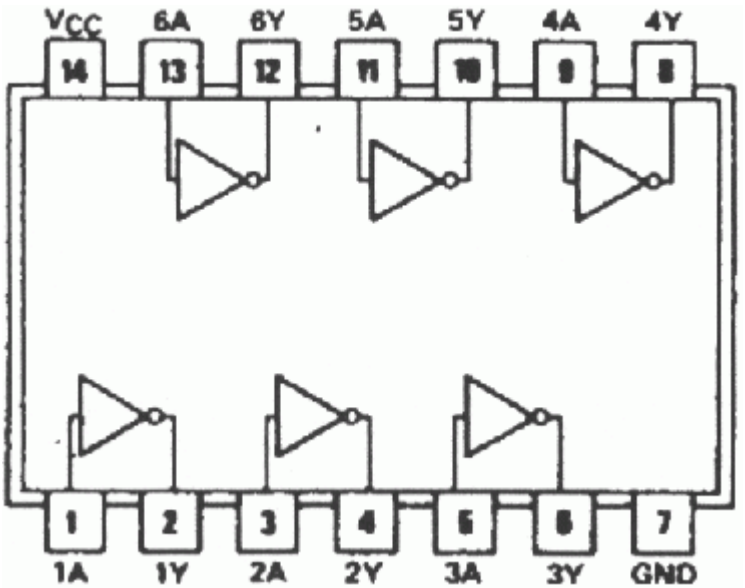
Zobrazte vstupní a výstupní bity pomocí LED diod (LS05 invertory s otevřeným kolektorem)



Zkuste odlišné kódovací schémata:

- Binární
- Dvojkové doplnění (přičti a odečti)

Vylepšete obvod pro odečítání kladných čísel.Vygenerujte tedy dvojkový doplněk vstupu B.Dosáhnete toho pomocí invertorů LS04.Požadované přičtení 1 může být vytvořeno bez dalšího hardware,jenom zapojením nosného vstupu C0.(připojením na řídící napájení)



SN5404 (J) **SN7404 (J,N)**
SN54H04 (J) **SN74H04 (J,N)**
SN54L04 (J) **SN74L04 (J,N)**
SN54LS04 (J,W) **SN74LS04 (J,N)**
SN54S04 (J,W) **SN74S04 (J,N)**

Semafor

Postavte provozní světelný okruh který zapne 3 LEDky ve správném pořadí zelená – žlutá –červená – žlutá - …

Krok 1:
Použijte čítač IC LS93 (binární čítač). Spojte následující:
Řídící napětí (5) na +5V
Zem (10) na 0V
Výstup Q_A (12) na vstup B (1)
Reset vstupy (2,3) na zem
Vstup A (14) musí být připojen na funkční generátor který generuje hodinový signál.(TTL kompatibilní, t.j mezi 0V a 5V) na frekvenci okolo 1 Hz. Zkontrolujte signál s osciloskopem předtím než jej zapojíte na vstup A.

Jestliže máte vše zapojeno správně,výstupy Q_A ,Q_B, Q_C, a Q_D ukazují binárně čítané vlny.Zkontrolujte to pomocí osciloskopu.

Krok 2:
Vygenerujte fázový signál provozního světla z výstupů Q_A a Q_B.Tyto výstupy čítají následující binární sekvence:00,01,10,11,00,…. (Q_A,Q_B)
LED diody by měli svítit:
zelená:čítač je na 00
žlutá: čítač je na 01 a 11
červená: čítač je na 10.

Následující rovnice definují LED signály:

zelená = $\overline{Q_A} * \overline{Q_B}$ (AND)

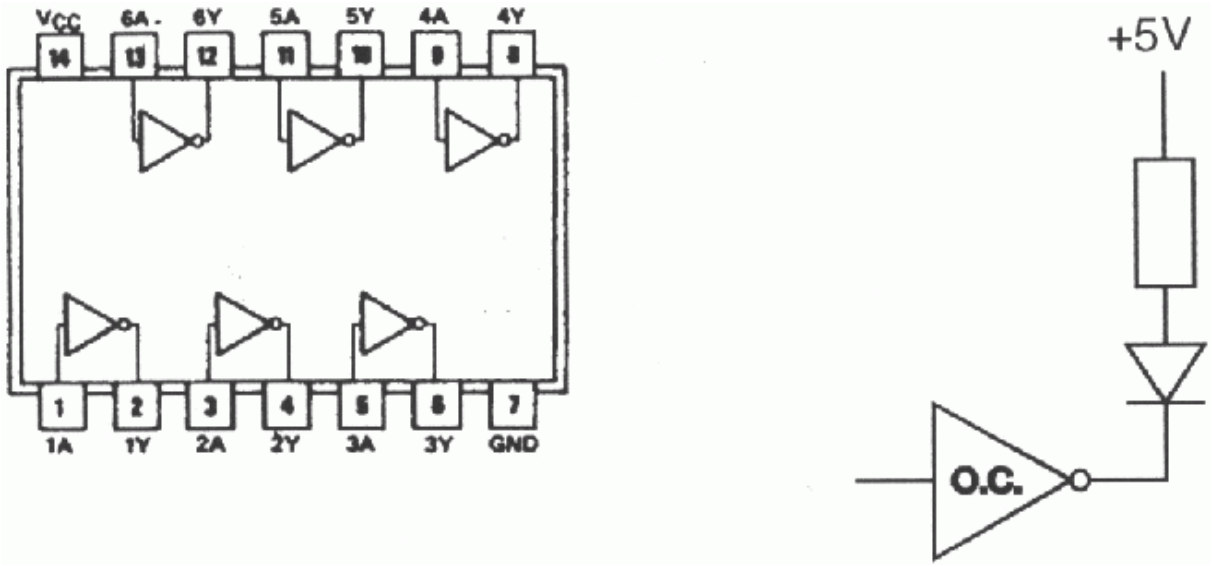
žlutá = Q_A

červená = $Q_B * \overline{Q_A}$ (AND)

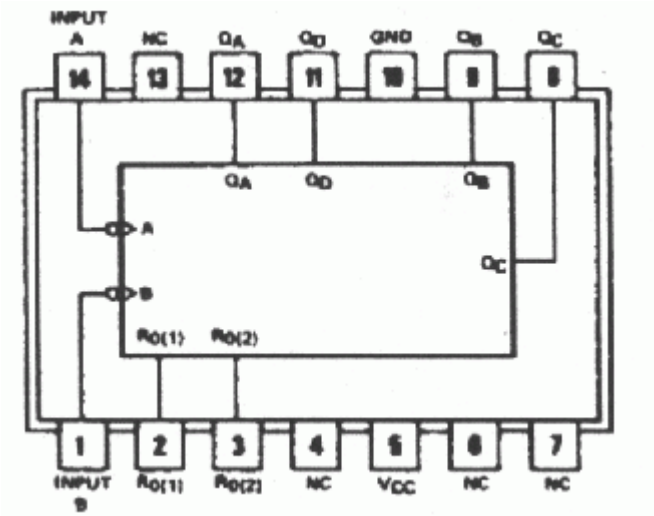
Použijte bránové funkce na implementaci těchto rovnic s minimem vyžadovaných integrovaných obvodů.

Krok 3:

Zobrazte tyto signály použitím červené,žluté a zelené LED diody, a naopak.
LS05 invertory s otevřeným kolektorem:



Čítač LS93:



PAMĚTI

Přehled

Typy paměťových zařízení:

- Random acess memory (RAM)
  - Možnost zápisu dat
  - Možnost čtení dat

- Read only memory (ROM)
  - Možnost čtení dat
  - Zařízení se programuje:
    - Při výrobě výrobcem (ROM)
    - Použitím speciálního hardwaru (PROM)
  - Některé typy ROM můžou být vymazány a znovu naprogramovány:
    - Ultrafialovým světlem (EPROM)
    - Elektrickým proudem (EEPROM)

Typické vlastnosti paměťových zařízení jsou:

- Velikost (více jak 64 Mbitů se běžně používá)
- Přístupová doba (záleží na typu zařízení, od 5ns do 100ns)
- Šíře jedné záznamové buňky (od 1 bitu do 32 bitů)

Adresování:

- Záznamové buňky jsou počítány od 0 do počtu buněk mínus 1.
- Tyto čísla jsou nazývána "**adresy**" odpovídajících buněk.

Random Access Memory (RAM)

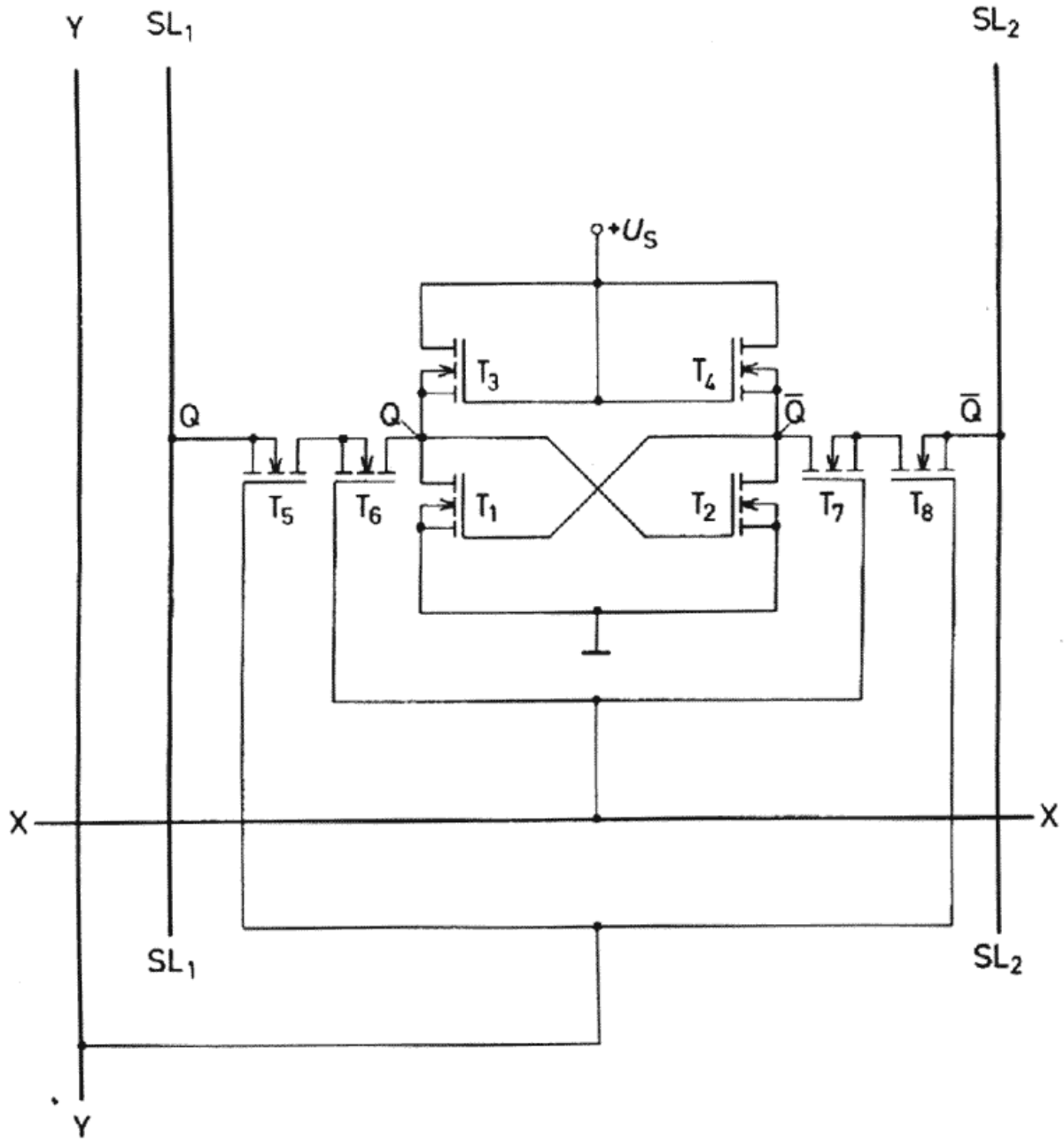
• RAM umožňuje přístup na každou svojí buňku.
(pro čtení nebo pro zápis)
• Postupnost adresování buněk je libovolná.
•Některé typy RAM jsou optimalizovány na velmi rychlý zápis nebo čtení postupně řazených buněk.Tyto paměti jsou používány jako video RAM v osobním počítači.

Existují 2 základní typy RAM:

• Statické RAM
• Dynamické RAM

Statické RAM

Záznamová buňka:

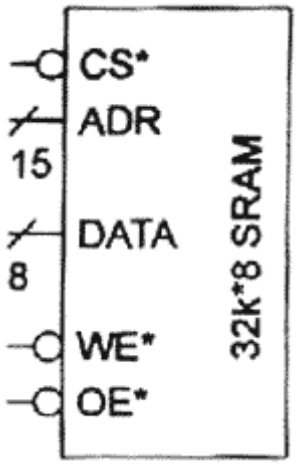


- Záznamová buňka vyžaduje 8 tranzistorů
- T1,T2 jsou klopné obvody, T3,T4 pracují jako rezistory.

- **Aktivace**: Záznamová buňka je aktivována použitím logických H úrovní na oba koordináty X a Y. To zapne tranzistory od T5 do T8
  a uchová logické úrovně Q a Q jsou napojeny na čtecí SL1 a SL2.
- **Čtení:** Okamžitě potom co záznamová buňka byla aktivována ,mohou být data přečtena z SL1 a SL2.
- **Zápis:** Jestliže nová datová hodnota je stejná jako současná žádná operace se neprovede.Data mohou být změněny jestliže je nová logická úroveň použita na SL1 a SL2 když je buňka aktivována…

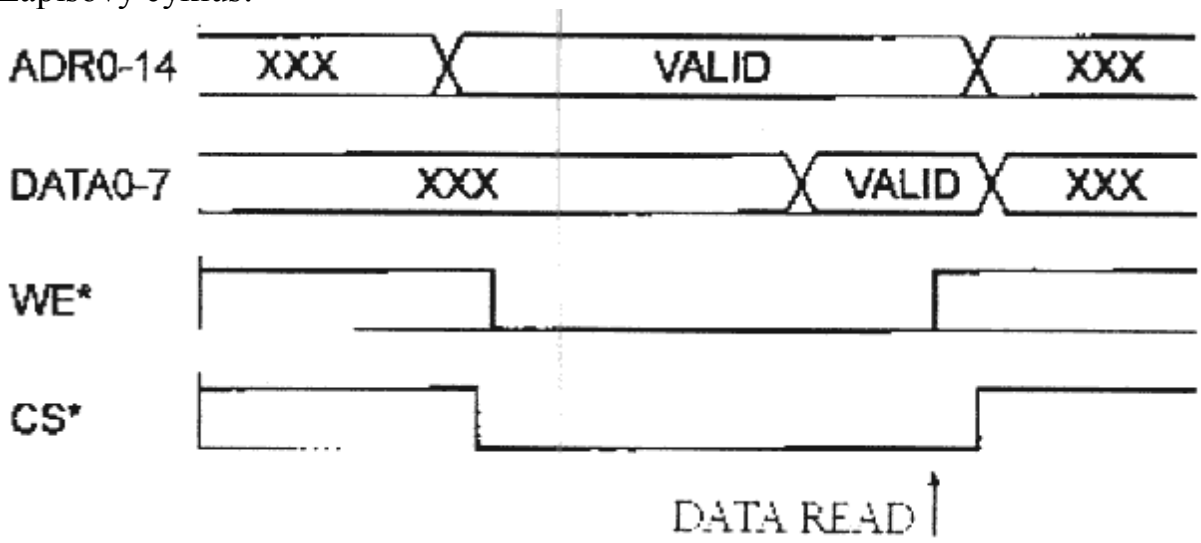
Statické paměti mohou být velmi snadno použity v elektronickém obvodu.

Příklad: 32K * 8 statická RAM

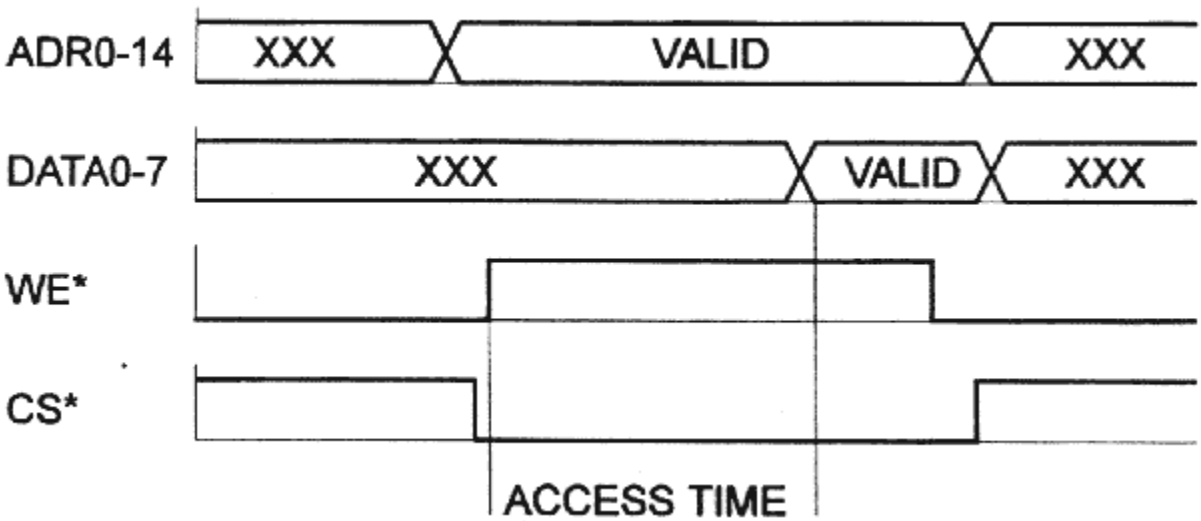


Signál	Jméno	Popis
CS*	chip select	CS* musí být aktivní (L) při zápisu a čtení.
ADR	address lines	Adresy pro 32 K (32768) záznamových buňek
DATA	data lines	data adresovaných záznamových buněk
WE*	write enable	definuje zápis (L) nebo čtení (H)
OE*	output enable	povolí vstup pro trojstavové buffery datalinek

Zápisový cyklus:



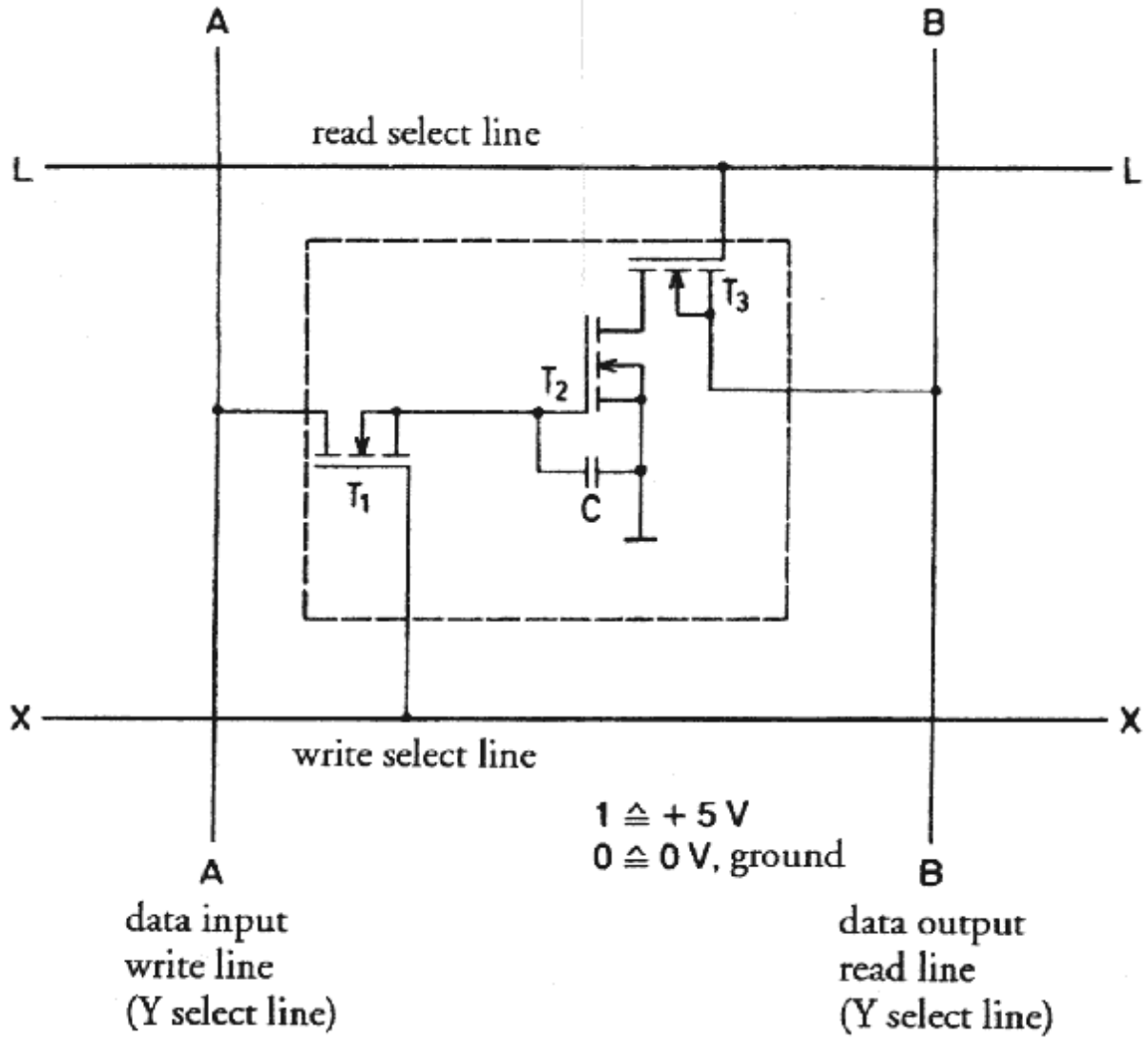
Čtecí cyklus:



(Přístupový čas)

Dynamické RAM

Záznamová buňka



Dynamická záznamová buňka vyžaduje pouze 3 tranzistory (ve srovnání s 8 u statické RAM).
Informace jsou uloženy jako elektrický náboj kapacitoru C tranzistoru T2.Jestliže je C nabit buňka si pamatuje 1,když vybit tak 0.

- **Zápis:** Nejprve je buňka aktivována použitím H levelu na X.To zapne tranzistor T1. Jestliže je teď linka A nastavena na H level,kapacitor C bude nabit, a jestliže je A nastavena na L level, kapacitor bude vybit.
  Konečně linka X se vrátí na L level.Tohle vypne tranzistor T1 a nabití kapacitoru C je zamčeno.

- **Čtení**:Nejprve je H level aplikován na linku B.Buňka je H levelem aktivována na lince L (linka vybírající čtení). Tohle zapne tranzistor T3.

  Jestliže buňka obsahuje 1,kapacitor C obsahuje elektrický náboj a záznamový tranzistor T2 je zapnut. Potom může proud protéct skrz B přes T3 a T2 je uzemněn. Tento proud je detekován na B a uložen jako 1.

  Jestliže informace v buňce je 0, kapacitor C neobsahuje elektrický náboj a tranzistor T2 je vypnut. Proto jím nemůže protékat proud když buňka obsahuje 0.

  Poznámka: Čtení nemění uloženou informaci

- **Refresh:** Kapacita C je velmi malá (0,1-1pF). Malé úniky proudu skrz bránu T2 vybijí kapacitor za několik ns. Proto informace v buňce musí být přepisována v periodických intervalech.

Výhody: Vyšší kapacita než u statických RAM

Nevýhody:

- Během periodických obnovovacích cyklů RAM nemůže být použita pro normální čtení nebo zápis. To zapříčiňuje malé prodlevy.

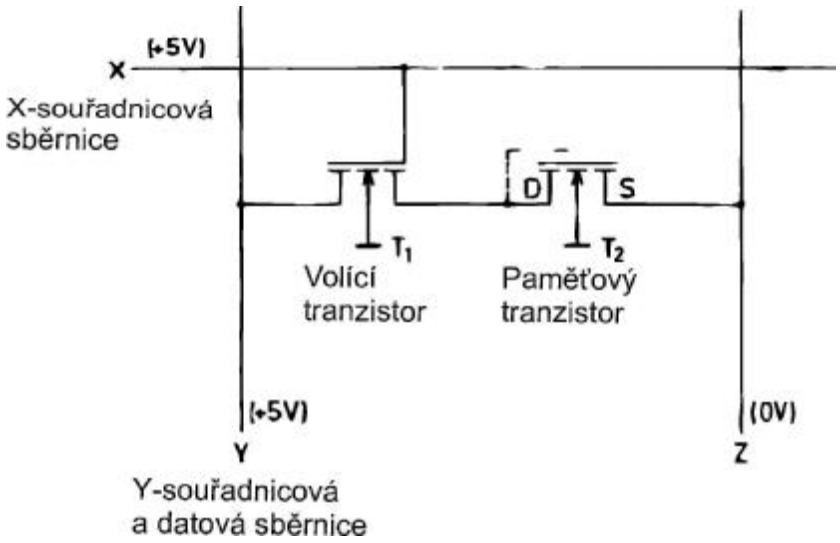
- Refresh vyžaduje přídavné elektronické obvody.

- Čtecí a zápisový cyklus vyžaduje velmi přesné časování. To způsobuje komplikovanost elektroniky.Pro tuto činnost se využívají ovládací čipy.

Read Only Memory (ROM)

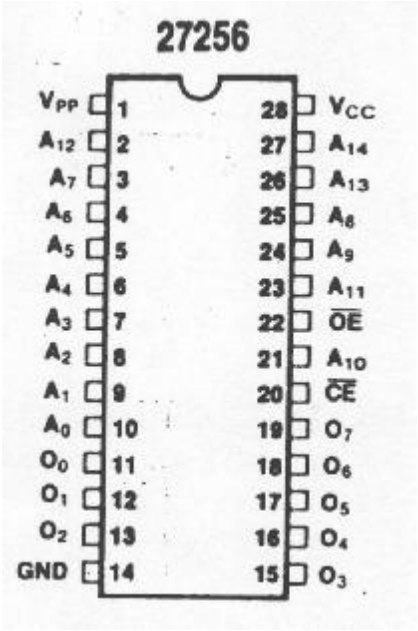
Příklad: EPROM

Paměťová buňka:



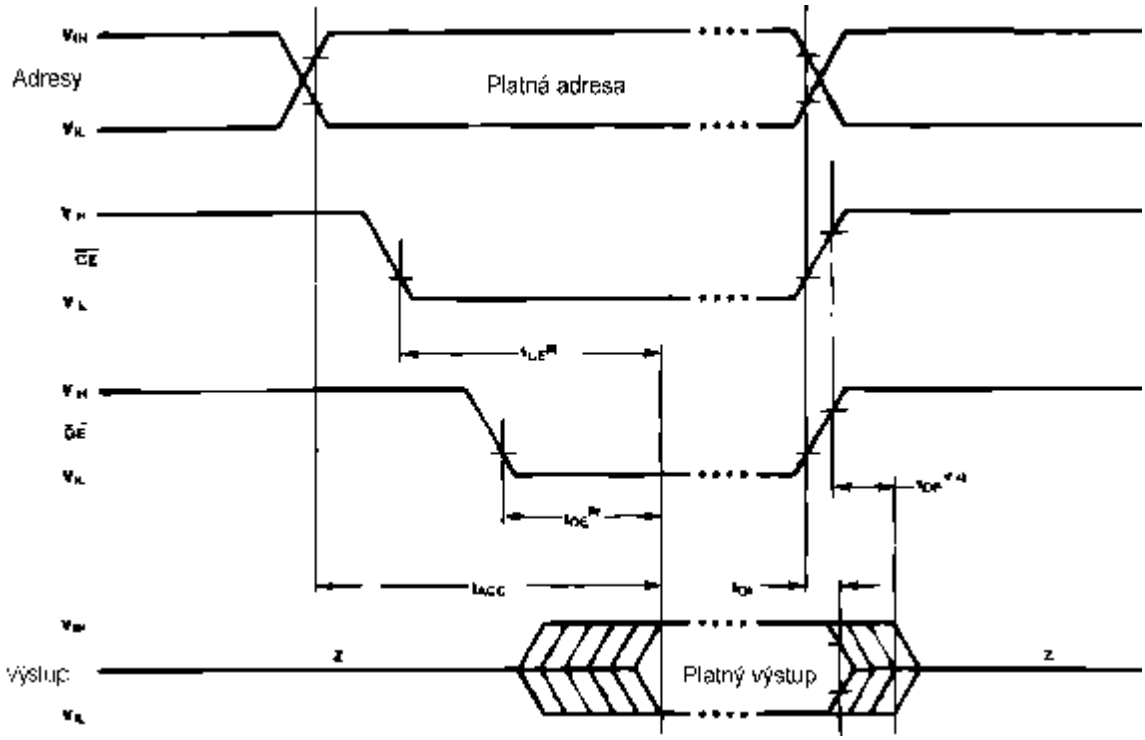
- Oba tranzistory jsou unipolární s indukovaným odděleným n-kanálem.
- Paměťový tranzistor T_2 obsahuje gate, který je vytvořen z vysoce izolačního materiálu tkzv.(floating gate = plovoucí gate). Ve vypnutém stavu není na gate náboj, T_2 je uzavřen.
- **Čtení:** Přiloží-li se na X a Y H-úroveň, stane se T_2 řídícím prvkem. Když je T_2 řídící, proud může téci pouze z Y do Z. Tento proud představuje danou informaci.
  Když obvodem neprotéká proud, na Y je hodnota napětí rovna 5V (tzn.poloha 1). Když proud protéká, napětí klesá na 0V (tzn. Poloha 0) .
  Vypnutý EPROM obsahuje všechny byty v poloze 1.
- **Programování:** probíhá pomocí nabíjení plovoucího gate přiložením vysokého napětí mezi D a substrát. V okolí gate vzniká vlivem tohoto napětí vysoké elektrické pole, elektrony putují z gate do D, místně (kvantově mechanický tunelový efekt).

Příklad: 32K * EPROM (INTEL 27256)



Signál	Název	Popis
CE*	Chip enable	CE* musí být aktivní (L), aby modul fungoval.
A_0-A_{14}	adresy	Adresová sběrnice pro 32 K (32768) paměťové buňky.
O_0-O_7	data	Data paměťové buňky
OE*	output enable	Zapojený výstup pro třícestný gate na datové sběrnici.
V_{pp},V_{cc}		Napájecí napětí 5V
GND		Uzemnění 0V

Čtecí cyklus:



CS* AND/OR OE* může zůstat stále aktivní. Při změně adresy jsou data platná po přístupové době okolo 100 ns.

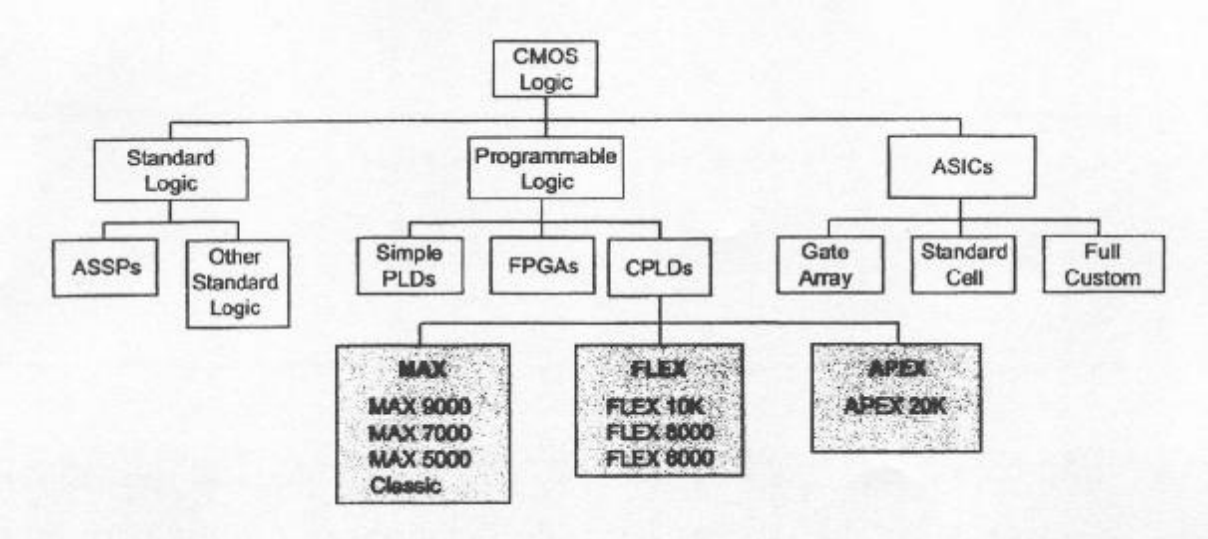
PROGRAMOVATELNÁ LOGICKÁ ZAŘÍZENÍ

Přehled

Moderní digitální obvody jsou dnes většinou složeny z TTL - modulů.
Jejich vlastnosti jsou:

- Nízko integrované, odtud pochází vysoký výběr modulů.
- Relativně vysoká energetická náročnost porovnatelná s více integrovanými obvody.

TTL-moduly vznikají v podstatě jako ,,Glue-Logic“ spojením více integrovanými IC (integrated circuit). Tyto obvody jsou většinou vyráběny CMOS technologií.



ASSP	Application specific standart product: Vysoko integrované logické zapojení s těsně definovanou funkcí například: (16-Bit čítač v multiplikačním zapojení).
Standart logic	Logické moduly v CMOS technologii, analogické s TTL skupinou.
Programable logic	**Vysoce integrované logické moduly, jejichž zapojení jsou definovány pomocí jejich využití a které jsou programovány pro využití v integrovaných obvodech.**
ASIC	Application specific integrated circuit: Binární spínací obvody, jejichž funkce je definovaná jejich využitím a jsou navrženy a vyrobeny jedním výrobcem. Nevýhoda: vysoká cena, nutnost výroby velkých sérií (10.000 – 100.000 kusů).

Komplexní programovatelné logické zařízení CPLD

Použití CPLD:

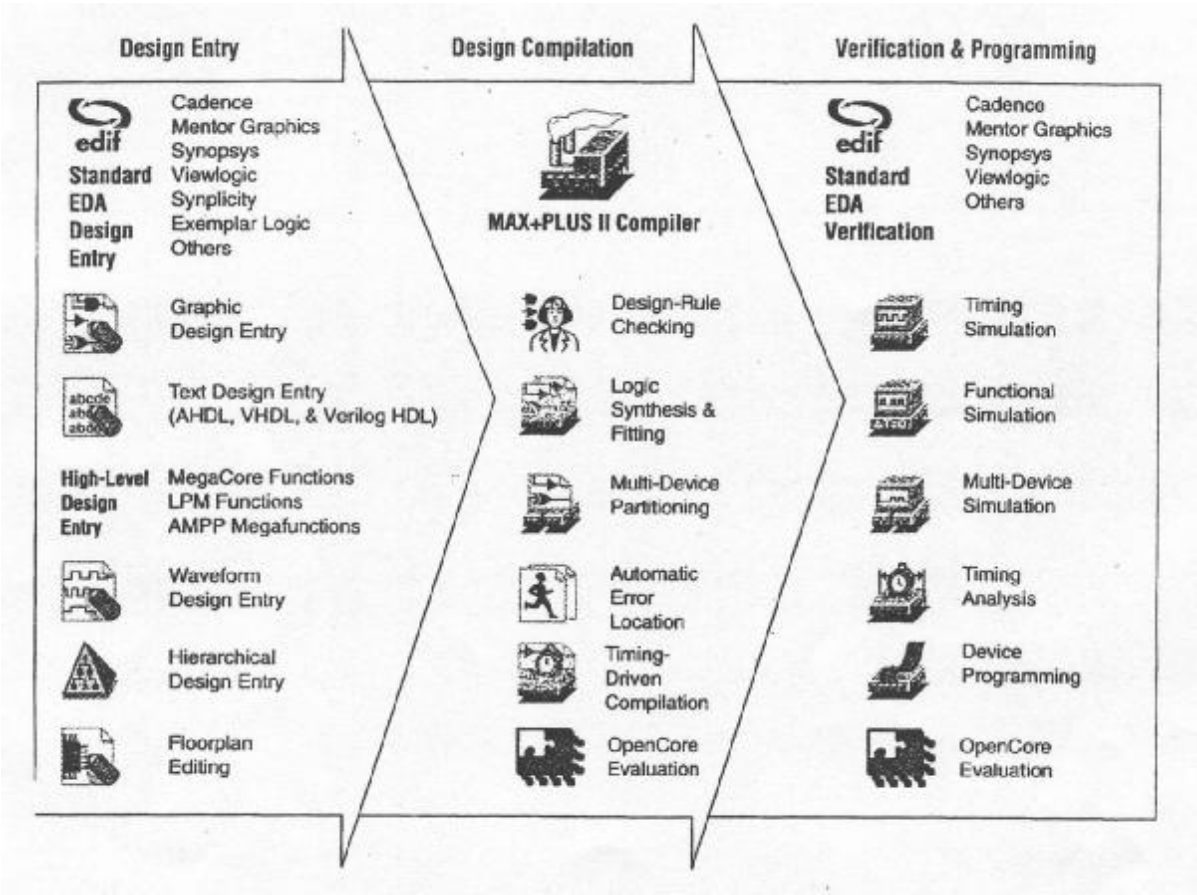
- CPLD jsou elektronické moduly, které jsou k dostání v obchodech, (stojí od několika ATS do přibližně 5000 ATS).

- Vlastností obvodů docílíme sestavením jejich logických zapojení. K dispozici máme několik možností:
- Nákres obvodu (schematické zapojení).
- Zakreslení vstupního a výstupního signálu (signálová forma).
- Definice zapojení v hardwarově orientovaném softwaru (např. AHDL, VHDL).

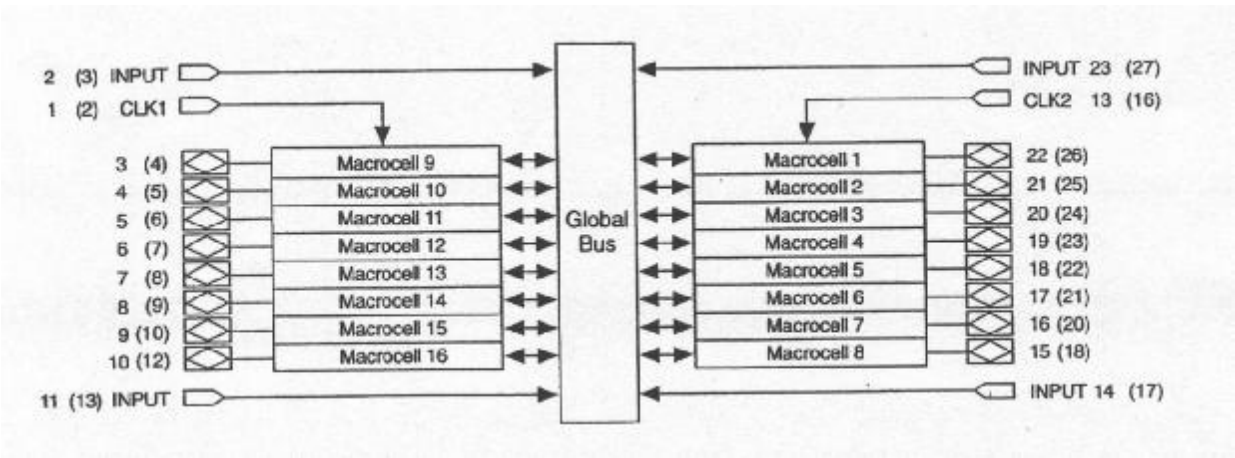
- Převzaté zapojení je kompilováno, simulováno, testováno, atd. počítačovým programem.
  Kompilátor poskytuje programová data pro moduly.

- Programová data jsou zapsána čítačem do modulu (jako u EPROM).
  Takto se stává integrovaný obvod funkčním.

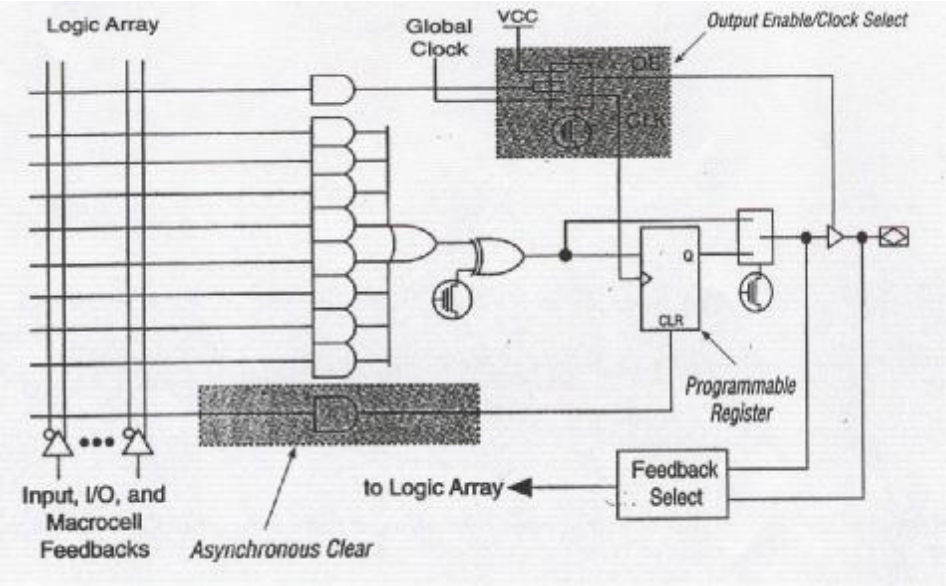
Příklad CPLD - vývoj systému (ALTERA, MAX+PLUS II)



Příklad jednoduchého CPLD (ALTERA,EP610)



Klasické schéma makrobuňky:



Global clock - centrální hodiny
Input, I/O, and macrocell feedbacks – vstup a výstup makrobuňky
Logik array – logické pole
Programmable register – programovatelný registr
Asynchronous clear – asynchronní mazání
Output enable/clock select – zapojený výstup/nastavení hodin

Podívejte se na tento CPLD:

- 16 makrobuněk, spojených přes globální sběrnici.
- Globální sběrnice je složena ze vstupních signálů a výstupních signálů makrobuňky.
- Jedna buňka je složena z logického pole, jehož spojení s globální sběrnicí jsou programovatelná.Tímto je vytvořena logická funkce z 8 vstupů (disjunktivní forma). Navíc každá buňka obsahuje klopný obvod, sériový třícestný zdroj, programovatelný spínač, kterým můžeme přemostit klopný obvod. S touto architekturou je možné již vytvořit komplexnější logické funkce, stavové stroje.
- Velké CPLD obvody obsahují více komplexnějších makrobuněk, tak jako komplikovaný programovatelný sběrnicový systémy (interconnect array).

Rozbor vědeckých plánů:
Standard CPLD, family FLEX 10K (Altera)

Tabulka 1. FLEX 10K součásti plánu

Součásti	EPF10K10 EPF10K10A	EPF10K20	**EPF10K30 EPF10K30A EPF10K30B**	**EPF10K40**	**EPF10K50 EPF10K50V EPF10K50B**
Typické brány (logická + RAM) *Pozn. / 1/*	10,000	20,000	30,000	40,000	50,000
Příslušné brány	7,000 až 31,000	15,000 až 63,000	22,000 až 69,000	29,000 až 93,000	36,000 až 116,000
Logické elementy (LEs)	576	1,152	1,728	2,304	2,880
Logické seřazovací bloky (LABs)	72	144	216	288	360
Zasazené sežazovací bloky (EABs)	3	6	6	8	10
Totální RAM bity	6,144	12,288	12,288	16,384	20,480
Maximální uživatelský počet I/O pinů	134	189	246	189	310

Tabulka 2. FLEX 10K součásti plánu

Součásti	EPF10K70	EPF10K100 **EPF10K100A EPF10K100B**	**EPF10K130V EPF10K130B**	**EPF10K180B**	**EPF10K250A EPF10K250B**
Typické brány (logická + RAM) Pozn. / 1/	70,000	100,000	130,000	180,000	250,000
Příslušné brány	46,000 až 118,000	62,000 až 158,000	82,000 až 211,000	119,000 až 248,000	149,000 až 310,000
Logické elementy (LEs)	3,744	4,992	6,656	9,728	12,160
Logické seřazovací bloky (LABs)	468	624	832	1,216	1,520
Zasazené seřazovací bloky (EABs)	9	12	16	16	20
Totální RAM bity	18,432	24,576	32,768	32,768	40,960
Maximální uživatelský počet I/O pinů	358	406	470	470	470

Poznámka k tabulce:
/ 1/ Pro design , který vyžaduje JTAG mezní scanovací – testování, vestavěné do JTAG obvodově, přispívá k více jak 31,250 doplňkovým branám.

Tabulka 5. FLEX 10K provedení							
Aplikace	**Použité zdroje**		**Provedení**				**Jednota**
	LEs	EAs	1-rychlostí stupeň *Pozn. /1/*	2-rychlostí stupeň	3-rychlostí stupeň	4-rychlostí stupeň	
16-bitový čítač *Pozn. /2/*	16	0		166	125	95	MHz
16-bitový akumulátor *Pozn. /2/*	16	0		166	125	95	Mhz
16-až 1 multiplexer *Pozn. /3/*	10	0		5.8	6.0	7.0	ns
256 x 8 RAM čtecí rychlost *Pozn. /4/*	0	1		118	103	84	Mhz
256 x 8 RAM zápisová rychlost *Pozn. /4/*	0	1		86	77	63	MHz

Poznámky k tabulce:

/1/ Konzultujte změny aplikací pro informace o 1-rychlostním stupni provedení.
/2/ Rychlostní stupeň této aplikace je omezen pro nízkou specifikaci.
/3/ Tato aplikace užívá kombinované vstupy a výstupy.
/4/ Tato aplikace užívá registrované vstupy a výstupy.

High – Performance CPLD, family APEX20K (Altera)

Tabulka 1. APEX 20K Součásti plánu									
Součásti	**EP20K60E**	**EP20K100E EP20K100**	**EP20K160E**	**EP20K200E EP20K200**	**EP20K300E**	**EP20K400E EP20K400**	**EP20K600E**	**EP20K1000E**	**EP20K1500E**
Maximum systém. bran	162,000	263,000	404,000	526,000	728,000	1,052,00	1,537,000	1,771,520	2,391,552
Typické brány	60,000	100,000	160,000	200,000	300,000	400,000	600,000	1,000,000	1,500,000
LEs	2,560	4,160	6,400	8,320	11,520	16,640	24,320	38,400	51,840
ESBs	16	26	40	52	72	104	152	160	216
Maximum RAM bitů	32,768	53,248	81,920	106,496	147,456	212,992	311,296	327,680	442,368
Maximum makrocelů	256	416	640	832	1,152	1,664	2,432	2,560	3,456
Maximální uživatelský počet pinů	204	252	316	382	408	502	624	708	808

Některé aplikace v našem institutu:

- Měřící systém pro radioaktivní radiaci (multikanálové analyzéry, multiparametricky shodné systémy)
- Mikrokontroler pro scanovací mikroskop (rastrový-tunelový mikroskop, výkonový mikroskop,…)
- Paprskový-odkláněcí systém pro urychlovače částic

STAVOVÉ STROJE

Přehled

Rozbor stavových strojů je základním konceptem moderní digitální elektroniky. Příklady postavení strojů v technice jsou:

- Průmyslově kontrolující využití: dopravní světla, robotika,…
- Domácí využití: mycí zařízení,…
- Mikropočítače: Dokonce i Pentium procesor je brán jako stavový stroj!

Průzkum komplexu digitálního obvodu z jiné strany:

Obvykle digitální obvod obsahuje:
- Vstupy (logické signály)
- Výstupy (logické signály)

Výstupní signály jsou jedinečnou funkcí:
- Vstupních signálů
- **a** historií předchozích vstupů (tzn. vstupních signálů, které byly předtím používány)

Příklad: čítač

Vstupy jsou:
- CLOCK „hodiny“ (počítač běžným způsobem sčítá na kladné části L -> H hodinového vstupu)
- CLEAR „čistí“ (jednoduše „čistí“ resp. nuluje hodnoty, až když se použije H-úroveň)

Výstupy jsou:
- Běžné počítané číslo jako kódované bity

Určité čítací výstupy jsou závislé na určitých vstupních signálech (CLEAR), ale v dodatku z historie z předchozích vstupních signálů (když byl čítač „vyčištěn“ resp. vynulován a kolik hodinových pulzů od té doby proběhlo).

V našem případě pozice-stav stroje je definována danými čítacími výstupy. Pro 4-bitový čítač existuje 16 možných stavů.
Jestliže je daný stav známý, potom další-následující stav může být jedinečně definován jako funkce vstupů.

Jestliže počet vstupů a výstupů je konečný, pak každé digitální obvod může být definován jako **konečný stavový stroj.** Daný stav je definován konečným seskupením digitálních signálů (jenž můžeme nazvat jako **kolísavý stav**)

Typy stavových strojů:

Principiálně záleží na přechodech mezi různými typy stavů, podle tohoto rozlišuje stavové stroje na 2 základní typy:

- **Asynchronní stavové stroje:**
  Stavové přechody jsou spouštěny dobře nadefinované, ale na druhou stranu svévolně logickými funkcemi ze vstupu.

- **Synchronní stavové stroje:**
  Stavové stroje zapadnou synchronně do jednoduchého vstupního signálu, jenž se nazývá „hodinový“ signál. (jen ve vzrůstajícím okraji L - > H hodinové části.)

Asynchronní stavové stroje jsou vzácně použity i v digitální elektronice, protože mohou snadno hlídat nespolehlivé reakce způsobené rozbitím či pozastavením časových……..!!

Proto budeme dále přiklánět jen k synchronním stavovým strojům.

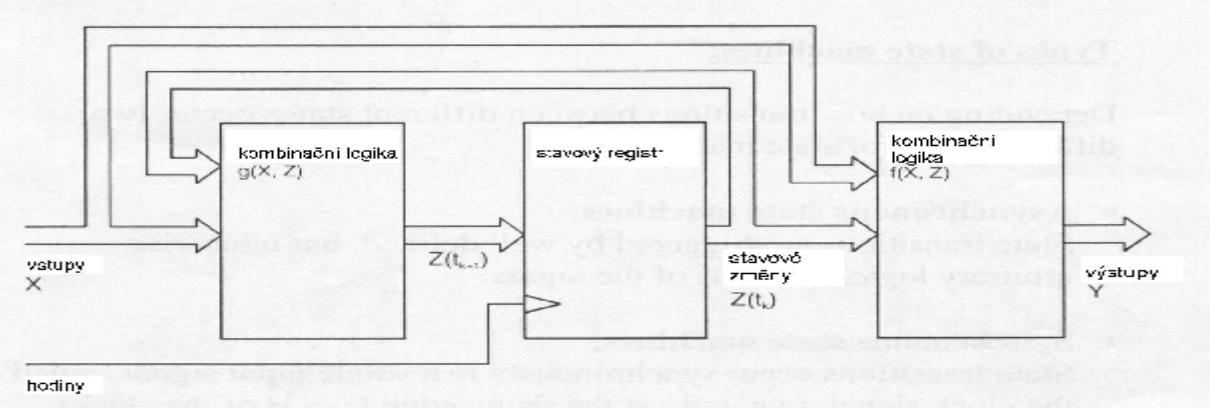
Synchronní stavové stroje

- Stavové přechody jsou analogicky synchronní se samostatným hodinovým vstupem. (jen na vzrůstajícím okraji hodin)

Typy synchronních stavových strojů:

Nejdůležitějším synchronním strojem tohoto typu je typ:

„MEALY“ (machine)



- Výstupy jsou závislé na konkrétním-daném stavu a na konkrétních vstupech:
$$Y(t_k) = f(X, Z(t_k))$$

- Další stav záleží na předchozím stavu vstupů:
$$Z(t_{k+1}) = g(X, Z(t_k))$$

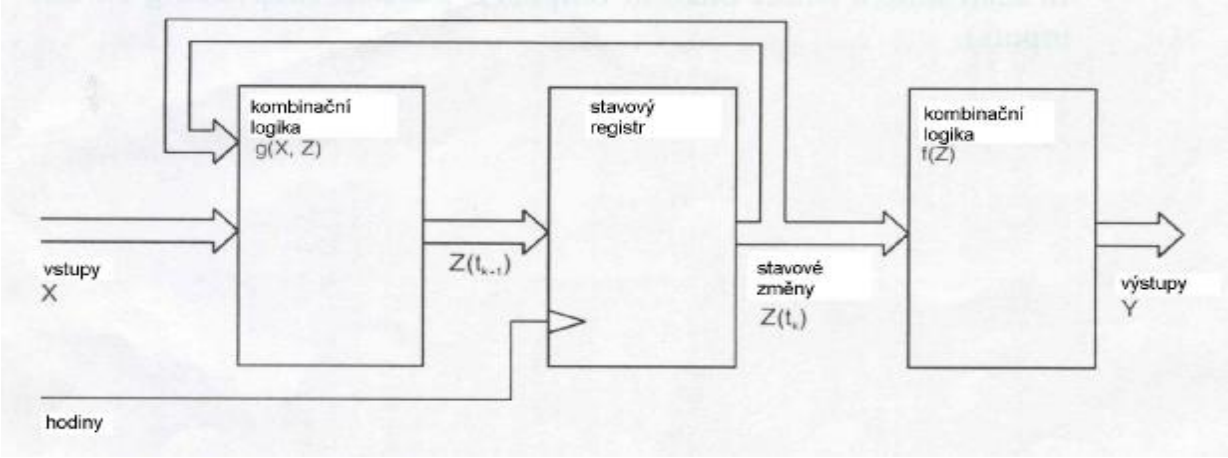
- V hodinovém pulsu bude další stav nahromaděn ve stavovém registru (tzn. „D FLIP-FLOPS”)

Proto stroje typu „MEALY“ , mají následující vlastnosti:

- Mění se **synchronně** do hodinového signálu
- **Výstupy** se mohou změnit pokaždé, jak funkce vstupu. (**asynchronně**)
- V každé pozici je třída výstupů možná , záleží na vstupech

Asynchronní vlastnosti výstupů tvoří charakter typů „MEALY“ velmi komplikovaně, protože zrušením či pozastavením může dojít ke zdržení signálu a to může dost nespolehlivě ovlivnit chování v obvodu skrz okamžité pulsy nebi nestabilní hladiny.
Proto se v nejvíce případech modifikovaných typů stavových typů strojů používá:

„MOORE“ (machine)



Tento systém je speciálním typem MEALY stroje:

- Výstupy závisí na daném-konkrétním stavu. Proto jsou v MOORE zařízení **pozice a všechny výstupy měněny synchronně do hodin.**

Definice stavových strojů

Následující metody mohou být použity k definování stavových strojů:

- „State diagram“ /stavový diagram/
- „State table“ /stavový stůl/
- „Flow diagram“
- Definice v hardwarové orientovaná programovým jazykem /AHDL, VHDL – hardwarové vysvětlení-popis jazyku/

Příklad:

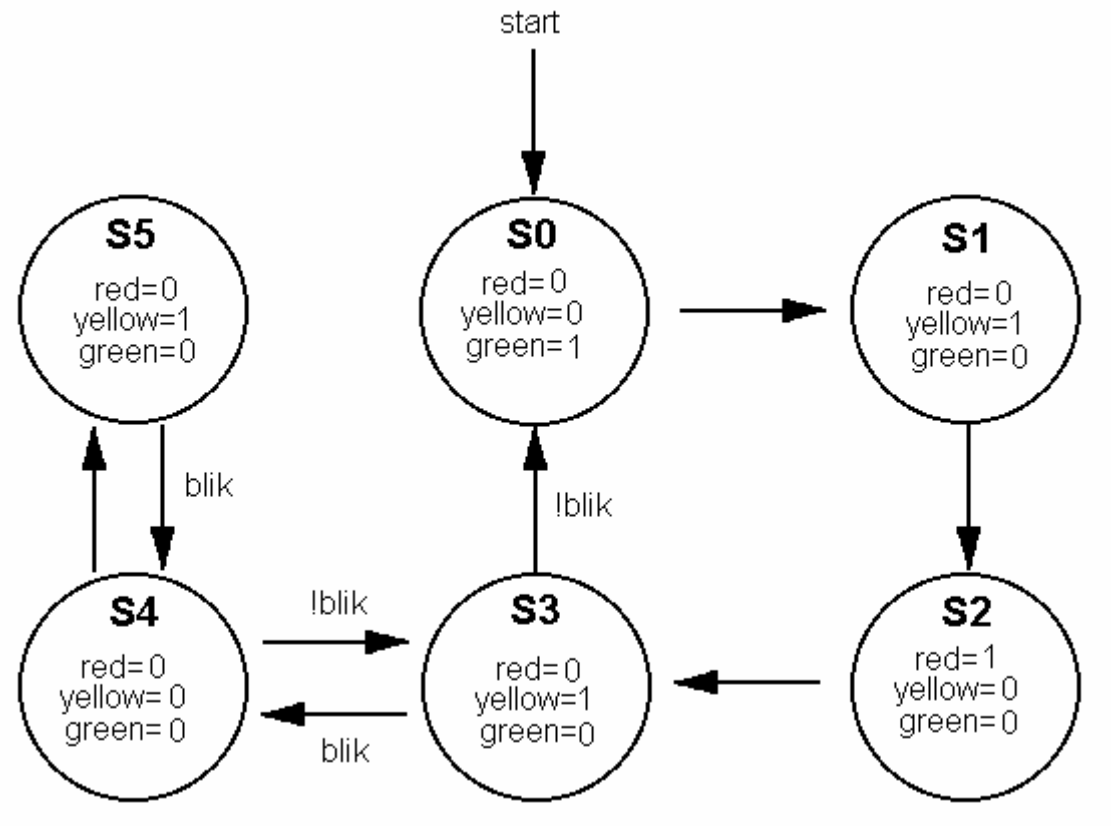
Popište semafor , který rozsvětsuje světla v následujícím pořadí zelená – žlutá – červená – žlutá … .S využitím dodatečného vstupního signálu (nazvaný blink) můžeme sepnout semafor tak , aby blikal žlutým světlem.Přechod k žlutému blikání není možný v kterémkoliv stavu pouze , když je stroj v normální žluté fázi(mezi červenou a zelenou fází).

Takovíto semafor nazýváme Moore – stroj s následujícími stavy:

S0:zelená
S1:žlutá (přechod ze zelené k červené)
S2:červená
S3:žlutá (přechod z červené k zelené)
S4:všechna světla zhasnuta
S5:žlutá (vždy bliká)

Výstupem jsou logické signály červená , žlutá , a zelená , které závisí pouze na aktuálním stavu stroje.

Stavový diagram:



- Pro Moor – výstupy stroje (jako v tomto příkladu) závisí pouze na aktuálním stavu.Proto jsou výstupy zapsány do stavových kruhů.
- Pro Moor – výstupy závisí na aktuálním stavu stroje a na vstupech.Proto jsou logické rovnice pro výstupy zapsány vně stavových kruhů.

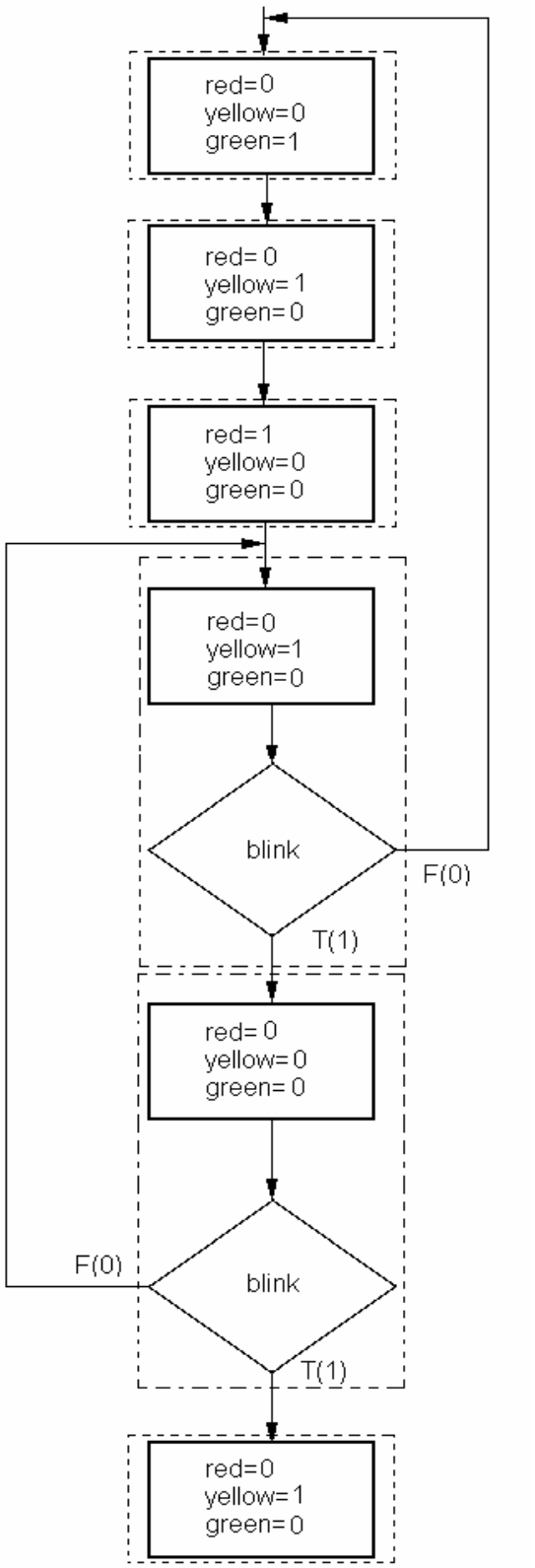
- Jestliže používáme hodiny s konstantní frekvencí , každá fáze semaforu má stejnou délku trvání.Tento nežádoucí jev můžeme snadno odstranit sekvenčním sčítáním identických stavů (např. S0_1,S0_2,…místo toho z S0 poskytneme delší zelenou fázi).

Tabulka stavů

Současný stav	Podmínka pro přechod, následující stav	Výstupy červená,žlutá,zelená
S0	S1	0,0,1
S1	S2	0,1,0
S2	S3	1,0,0
S3	bliká:S4 !bliká:S0	0,1,0
S4	bliká:S5 !bliká:S3	0,0,0
S5	S4	0,1,0

- Pro Mealy – stroj s nekonstantními výstupy,ale každý stav je funkcí vstupu.

Vývojový diagram
Algoritmický diagram stavového stroje



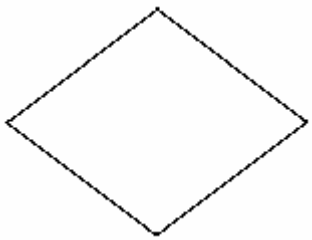
Symboly:

Stavy:

Nepodmíněný výstup:

Podmíněný výstup(pouze pro stroj Mealy):

Větev:



Důležité rozdíly programovacího jazyka:
Datové propojení stavového stroje v rámci stavu jsou paralelní (používá se kombinatorická logika).

Příkazy programovacího jazyky jsou prováděné sekvenčně.

Hardwarově orientovaný jazyk
(AHDL – Altera Hardware Description Language)

```
Title "Super Traffic Light"

SUBDESIGN traffic light
{
        sysclk          : INPUT; % system clock %
        reset  : INPUT; % system reset at power on %
        blink : INPUT; % blink request input %
        green : OUPUT;   % green light on output %
        yellow          : OUPUT; % yellow light on output %
        red             : OUPUT; % red light on output %
}

VARIABLE
        % declare state machine for traffic lights %
        light : MACHINE
                OF BITS ( g[2..0] )
                WITH STATES (       s0 = B"000",
                                    s1 = B"001",
                                    s2 = B"010",
                                    s3 = B"011",
                                    s4 = B"100",
                                    s5 = B"101" );
BEGIN
        % connect clock and rest inputs %
        light.clk = sysclk;
        light.reset = reset;

        % define state transition %
        CASE ( light ) IS
                WHEN s0 =>
                        green = 1;
                        yellow = 0;
                        red = 0;
                        light = s1;
                WHEN s1 =>
                        green = 0;
                        yellow = 1;
                        red = 0;
                        light = s2;
```

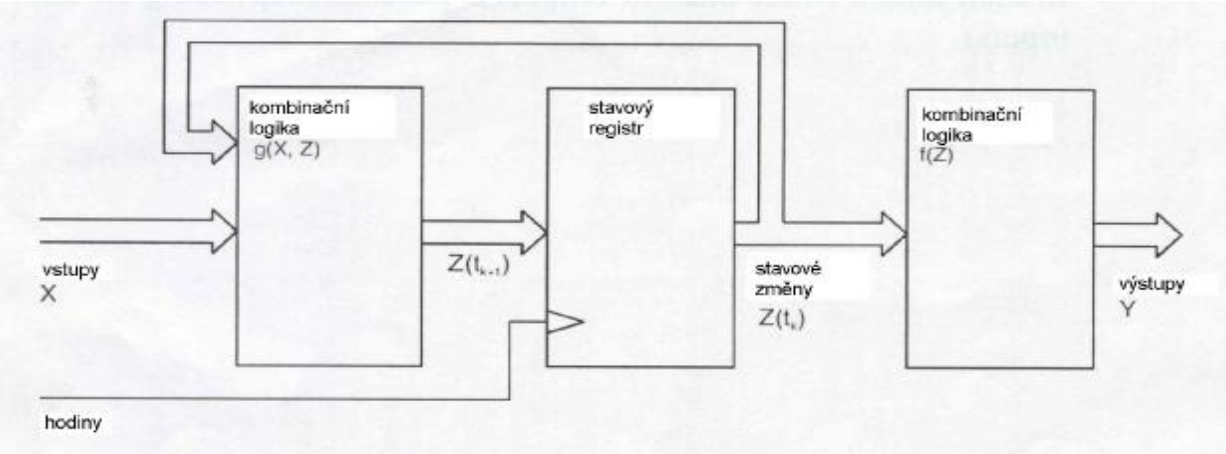
```
            WHEN s2 =>
                green = 0;
                yellow = 0;
                red = 1;
                light = s3;
            WHEN s3 =>
                green = 0;
                yellow = 1;
                red = 0;
                IF (blink) THEN
                    light = s4;
                ELSE
                    light = s0;
                END IF;
            WHEN s4:
                green = 0;
                yellow = 0;
                red = 0;
                IF (blink) THEN
                    light = s5;
                ELSE
                    light = s3;
                END IF;
            WHEN s5:
                green = 0;
                yellow = 1;
                red = 0;
                light = s4;
        END IF;
END;
```

Výhody AHDL:

- standardizovaný jazyk
- Existují kompilátory a simulační software.
- Návrh může být programován přímo do programovatelného logického zařízení(PLD). Když je programování hotovo máme IC s požadovanými vlastnostmi.

Vytváření elektronických obvodů:

Stroj Moor- (pouze pro opakování)

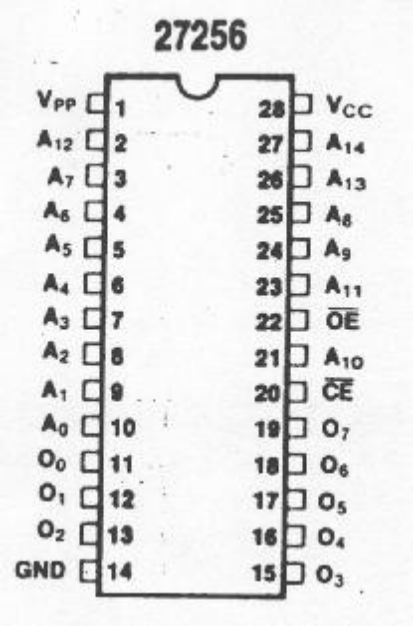


Vytváření elektronických obvodů:

- Kombinatorické funkce f a g můžeme vytvořit pomocí logických hradel.
  Výhody: přímé odesílání návrhu, jednoduchost.
  Nevýhody: i pro jednoduchý stavový stroj potřebujeme poměrně vysoké číslo IC.
- Kombinatorické funkce f a g můžeme vytvořit pomocí programovatelných paměťových zařízení (PROM nebo EPROM).
  Výhody: celá logika je obsažena v jednoduchém IC.
  Nevýhody: EPROM musí být programována EPROM programátorem.

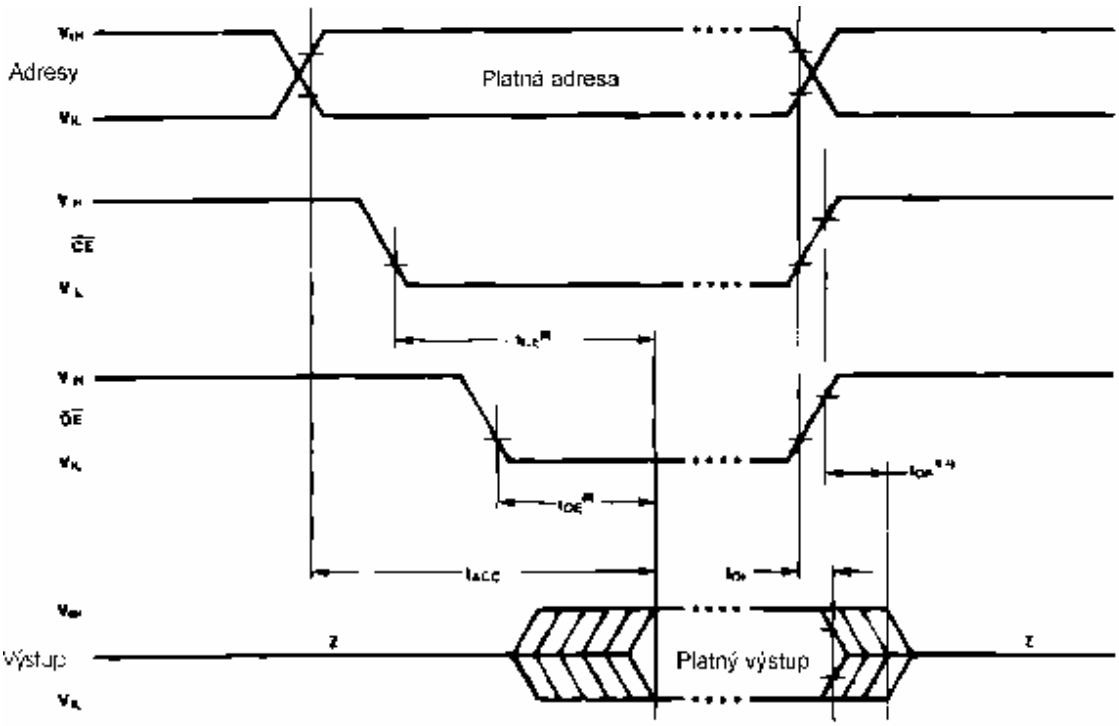
Jak můžeme vytvořit logickou funkci pomocí EPROM ?

- EPROM je paměťové zařízení, které můžeme naprogramovat pomocí vhodného programovacího hardwaru.
- Paměťové buňky EPROM mají často 8 bitovou kapacitu.
- Paměťové buňky EPROM jsou číslovány od 0 do maximálního možného čísla. Tyto čísla slouží pro vyvolání adres paměťových buněk..
- Když je binární adresa použita k určení vstupů EPROM ,binární data se objeví na datovém výstupu.



Signál	Jméno	Popis
CE*	chip přístupný	CE* musí být L k aktivování zařízení
A_0-A_{14}	adresové dráty	adresovací dráty pro 32 K (32768) paměťové buňky
O_0-O_7	datové dráty	data
OE*	výstup přístupný	přístupný drát pro třístavový nárazník na datových drátech
V_{PP},V_{CC}		stanovené napětí (5 V)
GND		uzemnění (0 V)

Signály:



CS* and/or OE* může zůstat nepřetržitě aktivní.Jestliže jsou v tomto případě adresovací signály změněny, data na výstupu jsou platná po přístupové době okolo 100 ns.

Příklad:

Realizace funkcí AND a NAND pomocí EPROM:

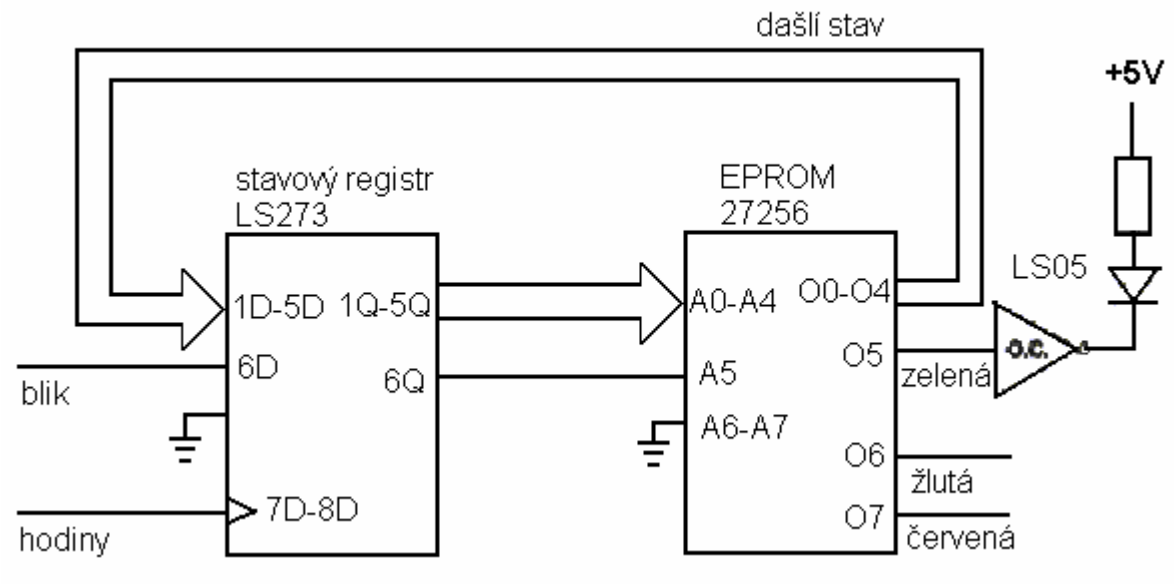
Obě funkce mají 2 vstupy (A,B) a jednoduchý výstup (ANDOUT,NANDOUT). Pro vstup jsou použity 2 adresové dráty (např. A0=A,A1=B), pro výstup používáme dva datové bity (např. D0=ANDOUT, D1=NANDOUT)
Pro všechny možné vstupní kombinace jsou odpovídající výstupní bity programovány do EPROM jako běžná data. V našem případě pravdivostní tabulku požadované EPROM zapíšeme jako:

A1, A0	D1, D0
0 0	1 0
0 1	1 0
1 0	1 0
1 1	0 1

Všechna nepoužitá datová slova můžeme naprogramovat jako 0.

- Tuto metodu můžeme použít v libovolně složitých kombinatorických funkcí logických proměnných k programování do jednoduchého IC.
- Omezení:
  číslo vstupu <= číslo adresovacího drátu EPROM
  číslo výstupu <= číslo datového bitu EPROM

Příklad:
Semafor:



Dodatečné propojení:

- V_{CC}, V_{PP} do 5V (stanovené napětí)
- GND do 0V (zem)
- LS273: CLR* do 5V (nepřístupná funkce clear)
- EPROM 27256: CS* , OE* do 0V (vybírá nepřetržitě čip, spínače na třístavovém výstupu).
- Hodiny funkčního generátoru (frekvence: 1Hz, jistí TTL – kompatibilní napětí!).

Programovací tabulka EPROM (část 1):

EPROM je navržena pro hodiny s frekvencí 1s.Jako popis stavů je opakovaně ukládána delší zelená a červená fáze.

blink = 0:

Adresa	765 43210(data)	Popis
0	010 00001	žlutá
1	010 00010	žlutá
2	100 00011	červená
3	100 00100	červená
4	100 00101	červená
5	100 00110	červená
6	100 00111	červená
7	100 01000	červená
8	100 01001	červená
9	100 01010	červená
10	100 01011	červená
11	100 01100	červená
12	110 01101	červená-žlutá
13	110 01110	červená-žlutá
14	001 01111	zelená
15	001 10000	zelená
16	001 10001	zelená
17	001 10010	zelená
18	001 10011	zelená
19	001 10100	zelená
20	000 10101	zelená
21	001 10110	zelená
22	000 10111	vypnuto
23	001 00000	zelená,skok na 0
24	000 00000	nevyužito
25	000 00000	nevyužito
26	000 00000	nevyužito
27	000 00000	nevyužito
28	000 11101	vypnuto(bliká)
29	000 00000	vypnuto(bliká),skok na normální operaci(0)
30	010 11111	žlutá(bliká)
31	010 11100	žlutá(bliká),skok na 28

Programovací tabulka EPROM (část 2):

blink = 1 (adresovaný bit 6):

Adresa	765 43210(data)	Popis
32	010 00001	žlutá
33	010 11100	žlutá,skok na bliknutí(28+32)
34	100 00011	červená
35	100 00100	červená
36	100 00101	červená
37	100 00110	červená
38	100 00111	červená
39	100 01000	červená
40	100 01001	červená
41	100 01010	červená
42	100 01011	červená
43	100 01100	červená
44	110 01101	červená-žlutá
45	110 01110	červená-žlutá
46	001 01111	zelená
47	001 10000	zelená
48	001 10001	zelená
49	001 10010	zelená
50	001 10011	zelená
51	001 10100	zelená
52	000 10101	vypnuto
53	001 10110	zelená
54	000 10111	vypnuto
55	001 00000	zelená,skok na 0
56	000 00000	nevyužito
57	000 00000	nevyužito
58	000 00000	nevyužito
59	000 00000	nevyužito
60	000 11101	vypnuto(bliká)
61	000 11110	vypnuto(bliká)
62	010 11111	žlutá(bliká)
63	010 11100	žlutá(bliká)

Algoritmus:

Tato programovací tabulka EPROM není úplně stejná jako zjednodušený stavový diagram.Provedením následujících modifikací dosáhneme realističtějšího semaforu:

- Doba trvání světelných fází je modifikována (zelená a červená trvají déle než žlutá).
- Na konci zelené fáze zelené světlo zableskne , jako je to obvyklé v Rakousku.
- Když semafor přepíná z červené na zelenou,červené a žluté světlo svítí současně.
- Stavový stroj startuje při žlutém světle.

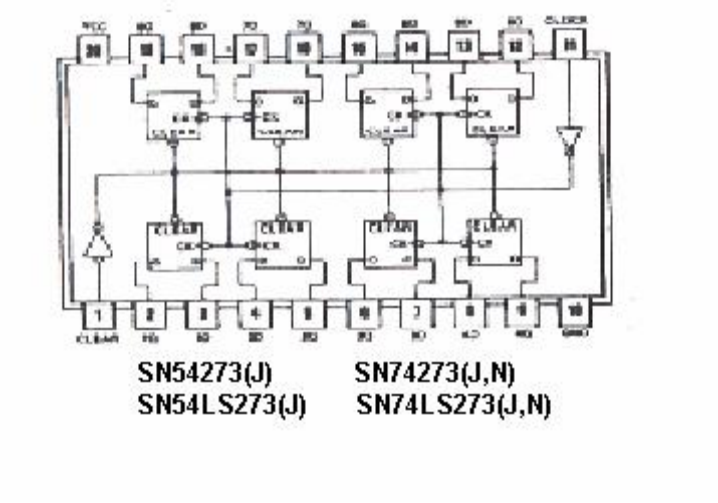
EPROM obsahuje adresy následujících stavů v datových bitech 0-4.Datové bity 5-7 jsou spínací signály pro barvy (zelená , žlutá , červená).

Datové bity 0-4 jsou spojeny se stavovým registrem.Po následujícím taktu jsou poslány na adresovací výstupy EPROM A0-A4 ke generování barvy signálu a adresy následujícího stavu.Na přidaném blink vstupu je propojení na adresovací bit A5.

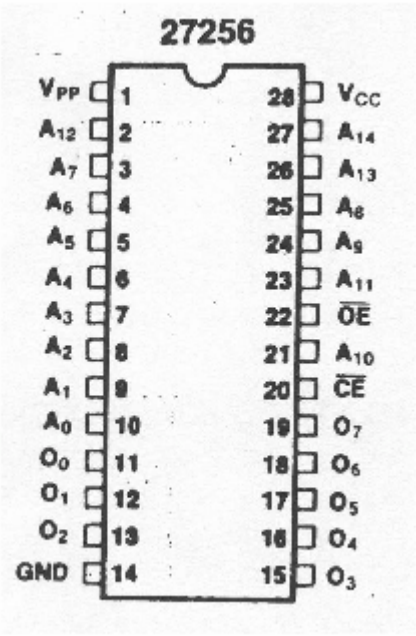
Pří normálním módo (blink = 0 a tudíž A5 = 0) je EPROM programována na cyklické opakování od stavu 0 do stavu 23. Jestliže je blink změněn na 1 adresovací bit je nastaven na 1. To přepne rozmezí adres EPROM na vyšší rozsah adres 32-63. Data z horní tabulky jsou převážně identické z daty ze spodní tabulky 0-31. Jakákoliv data na adrese 33 (konec žluté fáze) jsou prováděna jako skok na blikání na adrese 60. V tomto módu EPROM cyklicky opakuje od 60 do 63. Jestliže je blink vstup resetován na 0 , EPROM se přepne zpátky na spodní tabulku.Stav 29, skok na 0 je proveden a tak blikání je zrušeno.

Požadované části:

LS273:



EPROM 27256:



LS05: