SAIMON 1

NÁVOD - ČÁST DRUHÁ

POČÍTAČOVÁ LOGIKA A DIGITÁLNÍ ČÍSLA

Kapitola 1 - Jak počítač „vidí“ čísla

Tak v této části návodu se budeme věnovat úvodu do konstrukce počítačů. Jak si pamatuje čísla, co je to vlastně číslo. Tím bude asi nejlepší začít. Vrátíme se na chvíli k jednomu z prvních obvodů z minulého dílu, ledka se spínačem.

Obrázek 1.1: Ledka se spínačem

Víme, že ledka svítí, když sepneme spínač. A když ho nesepneme, tak nesvítí. Já odteď budu říkat, že zmáčknuté tlačítko znamená jedničku. Nezmáčknuté znamená nulu. Stejně tak to, že ledka svítí, znamená 1. A když ledka nesvítí, znamená to 0. Tohle teď napíšu do tabulky

S3	LED1	
0	0	Tlačítko nezmáčknuté = 0, LED nesvítí = 0
1	1	Tlačítko zmáčknuté = 1, LED svítí = 1

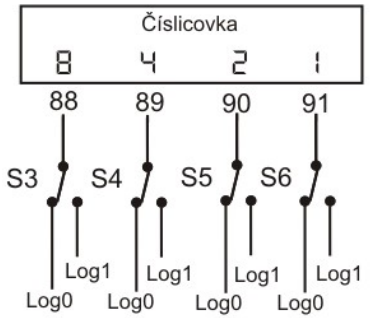
Obrázek 1.2: Tabulka hodnot obvodu se spínačem a ledkou

Když ledka svítí, tak skrz zmáčknuté tlačítko teče elektrický proud. Připomínám, že teče od plus až k vývodu ledky zvaný anoda, proteče ledkou a z vývodu katoda pak teče do mínus. Jinými slovy, když na anodu přivedeme plus, katoda je spojená s mínus, tak ledka svítí. A to znamená hodnotu 1.

V tomhle díle už pro nás plus bude nejen plus, ale také budeme říkat **Logická 1**, zkráceně **log 1**. A když existuje logická 1, tak určitě existuje i logická 0. Kdybychom na zdířku 93 přivedli mínus, tak LED1 svítit nebude, a tomu budeme říkat stav s hodnotou 0. Mínus pro nás už také bude víc, budeme říkat **Logická 0**, zkráceně **log 0**.

Teď to zopakuji s novým názvoslovím - když přivedeme hodnotu **log 1** na zdířku 93 u **LED1**, zdířka 92 bude vést na hodnotu **log 0**, tak bude ledka ve stavu 1.

Jak počítač „vidí“ čísla: Možná to někdo z vás ví, možná ne, ale počítače neumí pracovat s ničím jiným, než právě se stavy zapnuto a vypnuto, tedy že napětí je nebo není. Říkáme log0 (není napětí) a log1 (je napětí). A pomocí toho umí dát dohromady třeba čísla. Zapojte následující schéma.



Obrázek 1.3: Čísla a počítač

Úloha 1: Zapojení *: 123 - 126, 126 - 129, 129 - 132, 124 - 127, 127 - 130, 130 - 133, 124 - log0, 123 – log1, log1 - NAPÁJENÍ číslicovky, 122 - 88, 125 - 89, 128 - 90, 131 - 91*

Stiskem spínačů nastavujeme hodnotu log1 na vývody číslicovky. Když zmáčkneme například S4, které vede do zdířky 89, zobrazí se na číslicovce číslo 4. Digitální čtyřka je přímo pod zdířkou nakreslená. Zkuste S4 vypnout a zmáčknout jiné. Zobrazí se opět číslo pod zdířkou. Dále zkuste kombinace stisknutých tlačítek. Dokážete napsat číslo pro každou z kombinací uvedené v tabulce na obrázku 1.4? Která čísla v tabulce chybí a jaká by to byla kombinace nul a jedniček?

S3	S4	S5	S6	ČÍSLO
0	0	0	1	
0	0	1	1	
0	1	0	1	
0	1	1	0	
1	0	0	1	

Obrázek 1.4: Tabulka kombinací

Velmi rychle zjistíte, že kombinací, jak stisknout tlačítka, je víc, než je čísel. Čísel je 0,1,2,3,4,5,6,7,8,9. To je deset číslic (0 je první, 1 druhá, 9 desátá atd.). Kombinací celkem můžete najít 16. Aby se kombinace daly využít, tak se ty, pro které nám nezbyla číslice, zobrazují jako písmeno. A to podle tabulky na obrázku 1.5.

S3	S4	S5	S6	ČÍSLO
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7

S3	S4	S5	S6	ČÍSLO
1	0	0	0	8
1	0	0	1	9
1	0	1	0	A
1	0	1	1	b
1	1	0	0	C
1	1	0	1	d
1	1	1	0	E
1	1	1	1	F

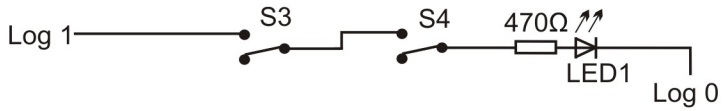
Obrázek 1.5: Tabulka zobrazení čísel a hexa-znaků

Místo čísla 10, na které bychom potřebovali už dva číselné znaky (1 a 0) , zobrazíme písmeno A, aby nám stačil jeden znak. Místo 11 zobrazujeme písmenko b, 12 = C, 13 = d, 14 = E a 15 = F. Způsobu zobrazení čísel pomocí nul a jedniček se říká **dvojková**, nebo **binární soustava** čísel. Tu máme vymyšlenou právě pro počítače, které znají jen vypnuto = 0, zapnuto = 1. Když přidáme písmena ABCDEF k číslicím, abychom využili všechny kombinace nul a jedniček, tak tomu říkáme **šestnáctková**, nebo **hexadecimální soustava** čísel. Někde se uvádí název hexadekadická. Ta soustava čísel, ve které jsme zvyklí počítat my lidi, se jmenuje **desítková**, neboli **decimální** (někde se uvádí dekadická). Důvod je ten, že máme na rukou deset prstů a bylo jednoduché každému přiřadit číslici. Kdybychom měli prstů šestnáct, tak by se nám s počítači dnes pracovalo lépe.

Binární soustava čísel bude pro nás nejdůležitější. Číslům 1010, 100, 1110, 111 atd. budeme říkat binární zápis čísla, a já možná budu v textu používat název binární číslo. Budu tím myslet takovýto zápis nul a jedniček. Těm jednotlivým jedničkám a nulám budeme říkat **bity**. Bit (z anglického BInary digiT) je pro nás právě tahle nejmenší jednotka, která může být 0 nebo 1. Číslo 1010 jsou 4 bity, 100 jsou 3 bity, 11 jsou dva, 10110 je 5 bitů a tak dále.

Kapitola 2 - Logické funkce

Víme tedy už, co to je logická 1 (to je plus, zapnuto, svítí) a logická 0 (to je mínus, vypnuto, nesvítí). Na kroužku jsem na začátku druhého pololetí položil otázku. Dokázal by někdo vymyslet obvod, kde budou dva spínače a ledka s ochranným rezistorem, která se rozsvítí jen v tom případě, že zmáčkneme oba dva najednou? Tenkrát na kroužku šli dva žáci k tabuli, schovali se za ni tak, aby na sebe neviděli (asi abych věděl, že to vymyslel každý sám a kdo lépe), a začali to řešit. A vyřešili. Schéma obvodu je na obrázku 2.1.



Obrázek 2.1 - Funkce AND se spínači

***Úloha 2: Zapojení** : 123 - log1, 122 - 126, 125 - 93, 92 - log0.*

Je jasně vidět, že musím zapnout **oba** spínače **zároveň**, aby obvodem mohl téci elektrický proud. "A zároveň" se anglicky řekne **AND**. Zkusíme zapsat různé kombinace zmáčknutí do tabulky. Kdy bude LED svítit?

S3	S4	LED1
0	0	0
0	1	0
1	0	0
1	1	1

Obrázek 2.2: Tabulka funkce AND a její schematická značka

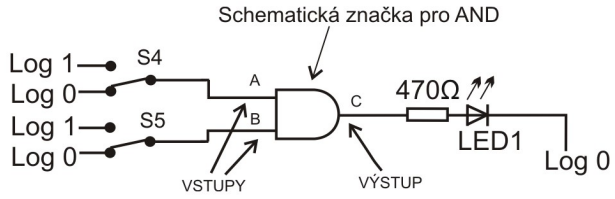
Z tabulky je vidět, že LED bude svítit jen v případě, že budou obě tlačítka ve stavu 1. To, co jsme právě vytvořili, se nazývá **logická funkce AND** neboli **logický součin**. Spínače S3 a S4 jsou pro nás **vstupy** logické funkce, LED je pro nás **výstup** logické funkce.

Teď se na chvíli vrátím k integrovaným obvodům, které byly popsané v kapitole 9 prvního dílu návodu. Tam jsem mimo jiné psal, že integrovaný obvod má mnoho funkcí, podle jeho typu, a že uvnitř něj je spousta součástek.

V této kapitole bych chtěl více popsat integrovaný obvod na stavebnici označený jako IO-2 7400 4xNAND. Znamená to, že uvnitř jsou 4 jakési obvody. Tyto obvody tvoří logickou funkci NAND, kterou popíšu dále. Nejdříve bych chtěl vysvětlit, co to jsou vstupy logické funkce a co jsou to výstupy logické funkce. **Vstup logické funkce** je takové místo v obvodu, kam přivádíme hodnotu **log1** nebo **log0**. Vstupů může být i **více**. Já je budu značit A a B. V jiné literatuře můžete najít značení například X1, X2. Podle hodnot vstupů se může měnit **výstup. Výstup logické funkce** je vždy jen **jeden**. Já ho budu značit například písmenem C. Opět v jiné literatuře můžete najít jiné značení, písmenkem Y.

Nejdříve vysvětlím, co vlastně znamená pojem logická funkce, když už jsme ji vytvořili**. Logická funkce** je **zatím** pro naše účely **obvod**, který má jeden nebo více **vstupů** a jeden **výstup**. Na vstupy přivádíme hodnoty log0 nebo log1, obvod je vyhodnotí, a nastaví výstup na odpovídající hodnotu. Tou může být log0 nebo log1. Nastavení výstupu vždy provádí podle příslušné tabulky. Například námi vytvořená funkce AND nastaví hodnotu výstupu odpovídající hodnotám vstupů podle tabulky na obrázku 2.2.

Na stejném obrázku je vedle tabulky i schematická značka logické funkce AND. Je tam napsáno, kde jsou vstupy a kde je výstup. Jak zapojit obvod, aby se hodnoty vstupů měnily stiskem spínačů? Odpověď je na obrázku 2.3. Stav výstupu logické funkce můžeme sledovat například ledkou (svítí – výstup má hodnotu log1, nesvítí – výstup má hodnotu log0)



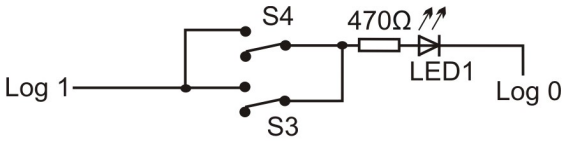
Obrázek 2.3: Zapojení vstupů a výstupů funkce AND

Pokud má nějaký obvod takovou funkci, že nastavuje hodnotu výstupu podle hodnot vstupů v tabulce na obrázku 2.2, nazýváme ho logická funkce AND a značíme ho schematickou značkou, která je na obrázku 2.2 nakreslená vpravo vedle tabulky. Jak se zapojují vstupy a výstup je vysvětleno na obrázku 2.3. Schematickou značku AND ale na stavebnici Saimon 1 nenajdete. Musíme ji sestavit z jiných logických funkcí. Jak provést vysvětlím později. Zatím nám stačí, že víme, jak pracovat se vstupy a výstupy a co to vlastně je.

Vstupů tedy může být více. Napadá otázka, umíme udělat takový obvod, aby měla funkce více než dva vstupy? To znamená větší počet tlačítek se stejnou funkcí? Je to velmi jednoduché. Prostě jich za sebe zapojíme více. V kapitole o rezistorech v prvním díle návodu je napsáno, že takovému způsobu zapojení se říká **sériové zapojení** spínačů (jsou za sebou). Na obrázku 2.4. vidíme logickou funkci AND se třemi vstupy.

Obrázek 2.4 - Funkce AND se třemi spínači

Zkuste teď sami vymyslet takový obvod, ve kterém budou dva spínače, ledka a její ochranný rezistor. Ledka bude svítit, když bude alespoň jeden ze dvou spínačů sepnutý, nebo budou sepnuté oba dva. Řešení je na obrázku 2.5.



Obrázek 2.5: Funkce OR se spínači

***Úloha 3: Zapojení** : 123 - log1, 126 - 123, 122 - 93, 125 - 93, 92 - log0*

Je vidět, že když stiskneme jakékoliv tlačítko, tak proud obvodem poteče. Můžeme zmáčknout oba spínače, **nebo** jen jeden z nich. "Nebo" se řekne anglicky OR. Tomuhle způsobu zapojení budeme říkat **logická funkce OR** neboli **logický součet.** Mám stejnou otázku, jako na předchozí stránce, totiž kdo vymyslí obvod pro logickou funkci OR s více, než dvěma vstupy? Zase je to velmi jednoduché. Tomuhle zapojení spínačů se říká **paralelní** (jsou vedle sebe). Řešení tentokrát nechám na čtenáři.

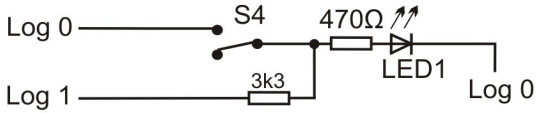
Tabulka funkce OR je na obrázku 2.6. Těmto tabulkám se říká **pravdivostní tabulky.** Log 1 pro nás také bude znamenat **pravda** (angl. **TRUE**) a log0 nebude lež, ale **nepravda** (angl. **FALSE**). Tabulky říkají, při jakých hodnotách vstupů jsou výstupy pravda nebo nepravda.

.

S3	S4	LED1
0	0	0
0	1	1
1	0	1
1	1	1

Obrázek 2.6: Pravdivostní tabulka funkce OR a její schematická značka

Pro nás bude důležitá ještě jedna funkce. Ta je na obrázku 2.7. Ze schématu asi není úplně jasné „co to dělá“.Zkuste obvod nejdříve zapojit.

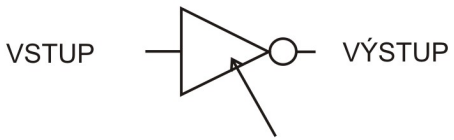


Obrázek 2.7: Funkce NOT se spínačem

Úloha 4: Zapojení : 9 - *125, 125 - 93, 92 - log0, 126 – log0, 10 - log1*

Když spínač nezmáčkneme, proud obvodem protéká od plus, přes rezistor 3k3, pak přes ochranný rezistor 470 a přes LED1 do mínus. Když spínač zmáčkneme, proud poteče od plus, přes odpor 3k3, a do mínus rovnou přes spínač. Přes ledku nepoteče nic. Ledka svítí (1), když spínač sepnutý není (0). A nesvítí (0), když spínač sepnutý je. Je to opačně. "Opačně" se do angličtiny překládá jako **NOT**. Pravdivostní tabulka funkce NOT je na obrázku 2.8.

S4	LED1
0	1
1	0



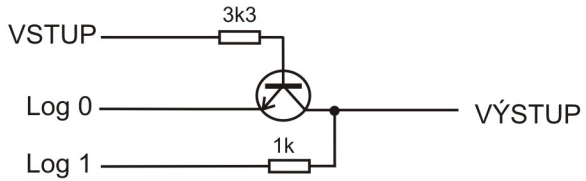
SCHÉMATICKÁ ZNAČKA
LOGICKÉ FUNKCE **NOT**

Obrázek 2.8: Pravdivostní tabulka funkce NOT a její schematická značka

Funkce, která „obrací“ log1 na log0, nebo obráceně log0 na log1, se nazývá **logická funkce NOT,** neboli **negace**. Obvod, který nám z logické 1 udělá logickou 0, bude mít funkci NOT. Tyhle všechny funkce počítač „umí“ a potřebuje, aby mohl fungovat.

Uvnitř počítače ale není nikdo, kdo by mačkal tlačítka. Nastavování hodnot vstupů se řeší součástkou, se kterou jsme se seznámili v předchozím díle návodu. Měla mimo jiné i funkci spínače. Vzpomínáte, jak jsme používali tranzistor jako spínač? No a v počítači jsou obvody zapojené pomocí tranzistorů, které tam nahrazují mechanické spínače.

Jak se nahradí mechanický spínač tranzistorem? Začnu s nejjednodušší funkcí. A to je NOT. Schéma zapojení je na obrázku 2.9.



Obrázek 2.9: Funkce NOT s tranzistorem

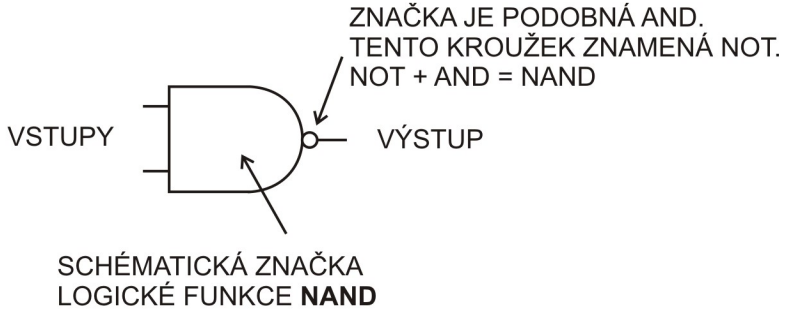
Úloha 5: Zapojení *:VSTUP je na zdířce 60A, 61 - log0, 58 - log1 , 59 je VÝSTUP*
Zapojení na starší verzi stavebnice Saimon: VSTUP je na zdířce 7, 8 - 60, 61 - log0, 58 - log1 , 59 je VÝSTUP

Když na vstup přivedeme log1, otevřeme tranzistor – do báze teče elektrický proud přes rezistor 3k3, tranzistor sepne a proud poteče přes rezistor 1k a přes tranzistor rovnou do minus. Otevřený tranzistor **spojuje emitor a kolektor**, na kolektoru a tím pádem na výstupu bude hodnota log0. Pokud přivedeme na vstup log0, tranzistor sepnutý nebude a na výstupu bude log1 - přes rezistor 1k.

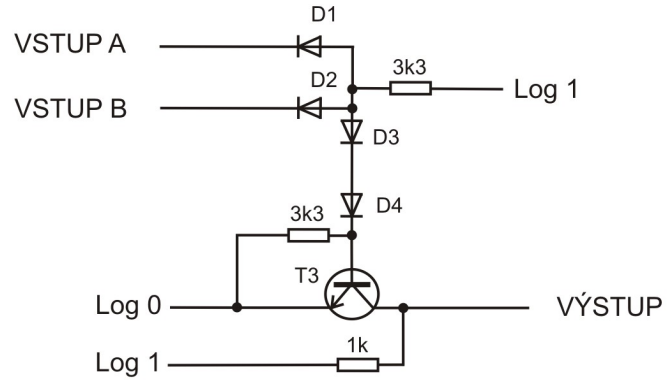
Funkce AND a OR není tak snadné zapojit pomocí tranzistorů. Ukážeme si jinou funkci, kterou je jednoduché zapojit. Bude to AND obráceně. Obráceně je NOT, čili něco jako NOT AND, zkráceně se to píše **NAND**.

Jak vypadá pravdivostní tabulka funkce AND, to víme z tabulky na obrázku 2.2. Stručně - když jsou zmáčknuté oba spínače, ledka svítí, jinak ne. NAND bude mít podobnou tabulku, jen výstup bude obráceně. Tedy ledka **nebude** svítit jen v případě, že jsou všechna tlačítka stisknutá.

S3	S4	LED1
0	0	1
0	1	1
1	0	1
1	1	0



Obrázek 2.10: Tabulka funkce NAND a její schematická značka.

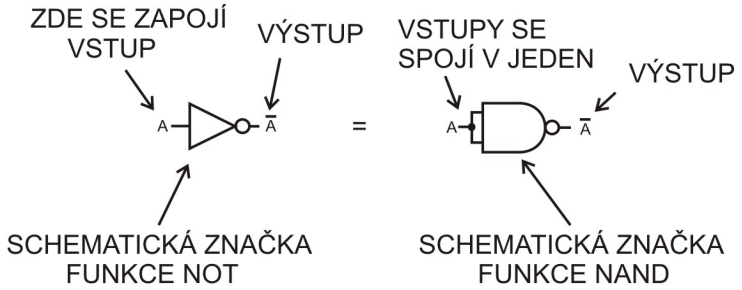


Obrázek 2.11: Schéma funkce NAND pomocí tranzistoru a diod

***Úloha 6: Zapojení** : 9 - log1, 10 – 101, 101 – 103, 103 – 105, 104 – 107, 58 – log1, 60 – 106, 61 – log0, 61 – 60A, 58 – VÝSTUP, 100 – VSTUP A, 102 – VSTUP B*

Obvod funkce NAND je velmi jednoduchý. Navíc má jednu vlastnost, kterou AND, OR ani NOT nemají. Pomocí dostatečného počtu funkcí NAND můžeme sestavit jakoukoliv logickou funkci. Tedy obvod, který bude mít stejnou funkci, jako právě zmíněné logické funkce AND, OR, NOT. Proto je na stavebnici integrovaný obvod IO-2 7400, který obsahuje čtyři logické funkce NAND a integrovaný obvod s jinými logickými funkcemi tam není (ale bude jich dost na modulu Saimon2). S pomocí funkcí NAND můžeme sestavit ostatní logické funkce a já vám ukážu jak.

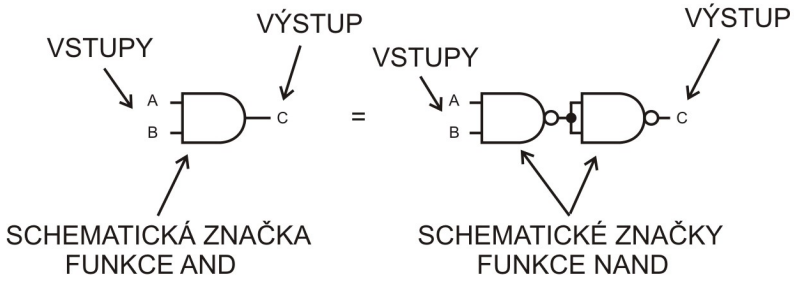
Začneme s nejjednodušší funkcí, a to je NOT. Pravdivostní tabulka je na obrázku 2.8. Sestavení funkce NOT pomocí jedné funkce NAND je jednoduché. NOT má pouze jeden vstup a jeden výstup. Spojíme vstupy NANDu a je to hotové. Schéma je na obrázku 2.12.



Obrázek 2.12: Zapojení funkce NOT pomocí funkce NAND

Obvod vlevo má stejnou funkci jako obvod vpravo. A to je negace, NOT.

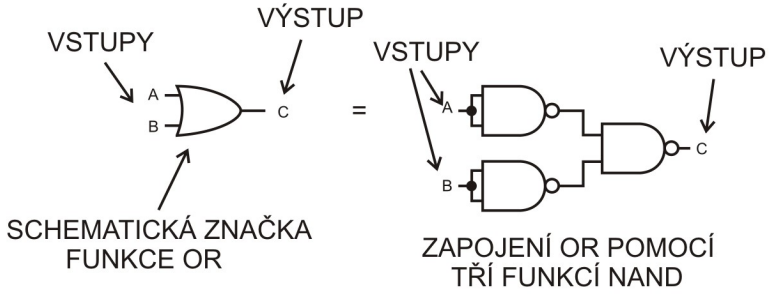
Kdybychom chtěli udělat z funkce NAND funkci AND, tak by musel být výstup funkce v pravdivostní tabulce na obrázku 2.10 „obráceně“, místo nuly jednička a místo jedniček nuly. To už vás možná napadne, stačí umístit za NAND ještě funkci NOT. Když víme, jak se NOT pomocí NAND vytvoří, je to jednoduché. Schéma obvodu je na obrázku 2.13.



Obrázek 2.13: Schéma sestavení funkce AND pomocí dvou funkcí NAND

Vstupy zde značím písmeny A a B, výstup písmenem C. Pokud budete hledat na internetu, můžete nalézt jiné značení. Vstupy se také značí X1, X2 a výstup písmenem Y.

Teď vytvoříme funkcí OR pomocí tří funkcí NAND. Když se pozorně zadíváte na pravdivostní tabulku na obrázku 2.10, tak si možná všimnete, že funkce NAND je něco jako OR pro logické nuly na vstupech. Stačí, aby alespoň jeden ze vstupů měl hodnotu log0, a výstup bude mít hodnotu log1. Toho se dá využít. Funkce OR má tu vlastnost, kdy stačí, aby alespoň jeden vstup měl hodnotu log1 a výstup bude mít hodnotu log1 také. Co jsem vlastně udělal ve schématu na obrázku 2.14 nechám na důvtipu čtenáře.



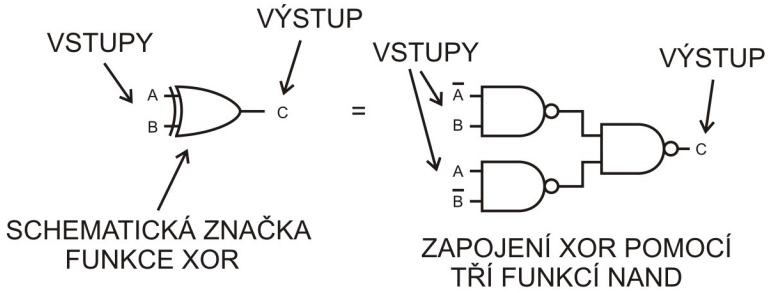
Obrázek 2.14: Schéma sestavení funkce OR pomocí tří funkcí NAND

Jak vidíte, funkce OR je náročnější na zapojení. Aby se vám logické funkce dobře sestavovaly, jsou výše uvedené schémata v jednodušší formě uvedené přímo na desce stavebnice, nad obvodem IO-2 7400, což je integrovaný obvod obsahující 4 funkce NAND.

Ještě pro úplnost uvedu logickou funkci **XOR**, která se nám bude také hodit. Je to vylučovací OR. Výstup nabývá hodnoty log1 pouze v případě, že hodnotu log1 má **pouze** jeden ze vstupů. Pravdivostní tabulka a schematická značka je na obrázku 2.15, její sestavení pomocí funkcí NAND je na obrázku 2.16.

A	B	VÝSTUP
0	0	0
0	1	1
1	0	1
1	1	0

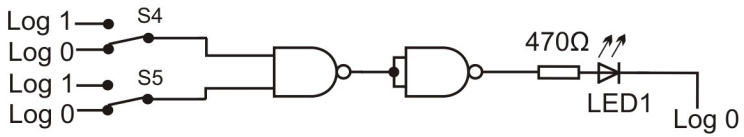
Obrázek 2.15: Pravdivostní tabulka funkce XOR a její schematická značka



Obrázek 2.16: Schéma sestavení funkce XOR pomocí tří funkcí NAND

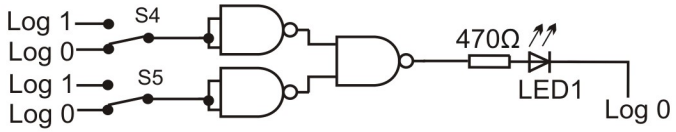
Otázka – dokážete zapojit funkci XOR jen pomocí spínačů? (Nápověda a zároveň řešení je zapojení úlohy 3 z prvního dílu návodu)

Ještě uvedu zapojení funkcí AND a OR na stavebnici pomocí IO-2 7400, spínačů pro změnu stavu vstupů a ledky pro sledování výstupu.



Obrázek 2.17: Funkce AND pomocí funkce NAND

***Úloha 7: Zapojení** : log1-123, log1-126, log0-124, log0-127, 92-log0, 73-68, 68-69, 71-122, 72-125, 70-93.*
napájení IO7400 - log1



Obrázek 2.18: Funkce OR pomocí funkce NAND

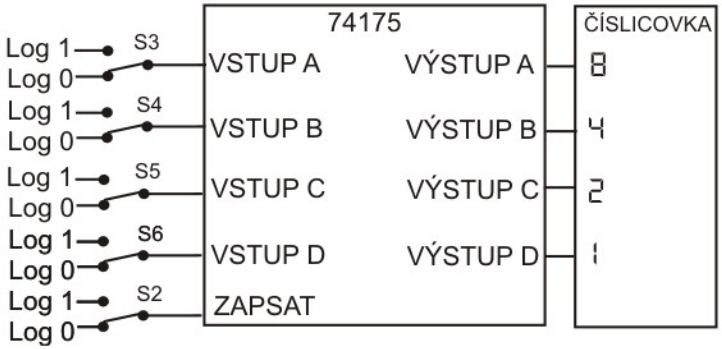
***Úloha 8: Zapojení**: 68 - 69, 71 - 72, log1 - 126, log0 - 127, 124 - log0, 123 - log1,70 - 62, 73 – 63, 72 - 125, 68 - 122, 64 - 93, 92-log0,*
napájení IO7400 - log1

Tohle všechno se bude hodit, až budeme stavět svůj první automatický systém, což je takový obvod, který sám něco dělá, reaguje na stisk tlačítek a podobně. A dělá to stále dokolečka. Prostě nějak "žije". To je účel stavebnice Saimon. Jak postavit takový obvod vysvětlím ve třetím díle návodu pro Saimon 1.

Kapitola 3 – Elektronická paměť

Abychom se dostali k podstatě a účelu stavebnice Saimon, je na čase prozkoumat poslední, nejdůležitější integrovaný obvod. Má označení IO-3 74175 4xD-KO. U tohohle integrovaného obvodu můžete vidět více vstupů a dokonce více výstupů. To je v pořádku. Jeden výstup mají obvody logických funkcí. **Paměťový obvod může mít výstupů více**.

Funkce obvodu 74175 je pamatovat si. Pokud vstupy - které jsou označeny písmeny A,B,C,D - obvodu nastavíme na určitou hodnotu (log0 nebo log1) a přivedeme na krátkou dobu hodnotu log1 na zdířku označenou ZAPSAT, tak obvod nastaví výstupy na stejnou hodnotu, jako byla hodnota na vstupu se stejným písmenkem, jako má výstup. **Ukážeme si to na příkladu**.



Obrázek 3.1: Obvod 74175 - ukázka funkce

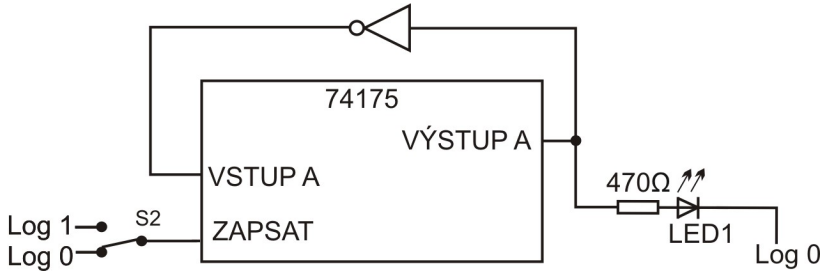
Úloha 9: Zapojení *: krátké drátky: 123 - log1, 126 - log1, 129 - log1, 132-log1 , 124 - log0 , 127 - log0, 130 - log0, 133-log0 , log1 - 120, log0 - 121, log1-napájení číslicovky, log1 - napájení IO74175,*
Dlouhé drátky: 122 - 77, 125 - 76, 128 - 83, 131 - 84, 119 - 87, 79 - 88, 74 - 89, 81 - 90, 86 - 91.

Pokud vše funguje správně, tak číslo, které nastavíte kombinací sepnutých spínačů S3 až S6, si obvod 74175 zapamatuje po stisku spínače S2 a zobrazí se na číslicovce. Můžete zapínat a vypínat spínače S3 - S6 jak chcete, ale zobrazené číslo se nezmění.

Obvod 74175 je malá paměť, říká se mu registr, neboli **klopný obvod.** To znamená, že nastaví, neboli naklopí své výstupy podle logické hodnoty, kterou jsme přivedli na odpovídající vstup. To zapamatování, neboli naklopení, proběhne při stisku spínače S2, tedy ve chvíli, kdy změníme log0 na log1 na vývodu označeném jako ZAPSAT. Zapamatováni se děje jen při té **změně z log0 na log1**. Tomu říkáme **Náběžná hrana,** nebo **Hodinový pulz.** I když na vývodu ZAPSAT zůstane log1 nebo log0 trvale, tak se výstupy nezmění.

Obvod má na stavebnici nakreslené vstupy A, B, C, D, a výstupy A, B, C, D. K čemu jsou, to už teď víme. Když na nějaký vstup, třeba VSTUP A, přivedeme log1, na zdířku ZAPSAT přivedeme náběžnou hranu, tak na stejně označeném výstupu, tedy VÝSTUP A, objeví také log1. Má ale ještě výstupy, které mají nad písmenkem čáru. Ta čára znamená, že ten výstup je obráceně oproti výstupu bez čáry. Když bude na zdířce VÝSTUP A hodnota log1, tak na výstupu $\overline{A}$ bude hodnota log0. Je to vlastně vývod výstupu A, za kterým je ještě zapojená funkce NOT. To je moc důležité pro konstrukci obvodů, které se naučíme ve třetím díle návodu.

Malý úvod, jaké obvody to budou, udělám už teď. Zkuste zapojit následující schéma.

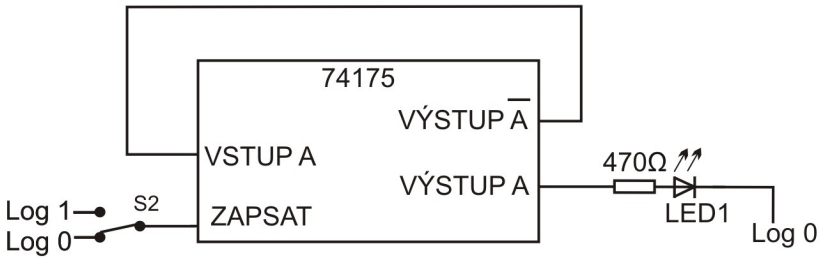


Obrázek 3.2: Obvod 74175 jako jednoduchý sekvenční obvod

***Úloha 10: Zapojení** : 71 - 72, 71 - 79, 73 - 77, log1 - napájení 7400, log1 - napájení 74175, log1 - 120, log0 - 121, 79 - 93, 119 - 87, 92 - log0*

Když zapojíme obvod z obrázku 3.2, tak bude mít velmi zajímavou funkci. Po stisku S2 si zapamatuje úroveň, která je na vstupu A. Tam ale bude opačná hodnota, než je na výstupu A, protože funkce NOT nám ji otočí. Když bude na výstupu A log0, tak si obvod zapamatuje log1, protože bude mít na vstupu A opačnou hodnotu, díky funkci NOT. Nastaví log1 na výstupu A. Funkce NOT nám ji ale zase otočí na log0 a přivede na vstup. A po dalším stisku si obvod zapamatuje log0, kterou nastaví na výstupu A. A tak dále. Mačkáním S2 by se měla na výstupu pravidelně měnit hodnota z log0 na log1 a obráceně.

IO 74175 má v sobě zabudovanou funkci NOT pro výstupy, jak již bylo řešeno. Jsou to ty výstupy s čarou, které mají vždy opačnou hodnotu, než výstup se stejným písmenkem. Obvod na obrázku 3.3 má stejnou funkci, jako obvod na obrázku 3.2. Dobře si je oba prohlédněte.

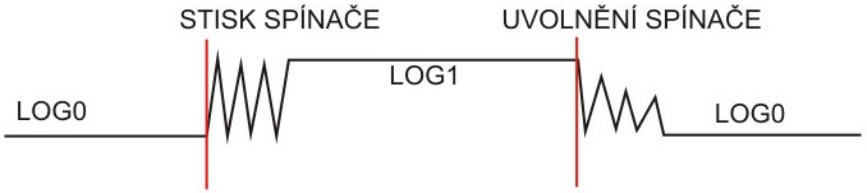


Obrázek 3.3: Obvod 74175 jako jednoduchý sekvenční obvod

Tak. Když to ale zkusíte zapojit, tak stiskem spínače S2 zjistíte, že se hodnota na výstupu mění nějak nepřesně. LED1 bliká dost nepravidelně, rozhodně se nedá mluvit o tom, že by se pravidelně střídala 0 a 1. Stisknutím spínače S2 se totiž nezmění úroveň z log0 na log1 okamžitě. Chvíli to jakoby „jiskří“, rychle se to mění z log0 na log 1 a obráceně a až pak se hodnota ustálí. To jiskření se děje tak rychle, že to lidské oko není schopné vidět, ale integrovaný obvod to „vidí“. Když spínač uvolníme, děje se to samé, zase chvíli trvá, než se z log1 stane log0. Obvod 74175 každou změnu bere jako povel zapamatovat si.

Spínač je pro „kvalitní“ změny hodnot na výstupu nepoužitelný. Potřebujeme nějaký kvalitní zdroj změny z log0 na log1, kde změna proběhne jen jednou.

Obrázek 3.4 ukazuje představu, jak to "jiskření" u spínače asi probíhá. Tomu jiskření říkáme zákmit nebo **přechodový jev**.

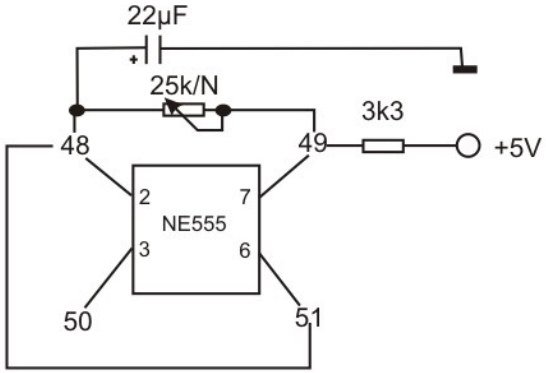


Obrázek 3.4: Zákmit při stisknutí spínače.

Kapitola 4 – Obvody pro získání hodinových pulzů

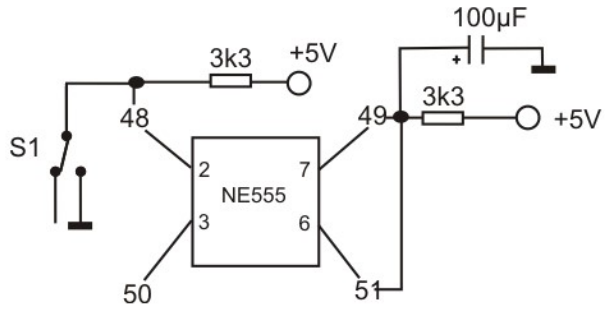
Nejlepší zdroj náběžné hrany, čili hodinového pulsu, je integrovaný obvod NE555. Jeho pro nás podstatné funkce známe z prvního dílu Saimon 1. Já je zde zopakuji.

Astabilní obvod – na výstupu (zdířka 50) se stále mění hodnota z log0 na log1 a zase zpět. Neustále tedy generuje hodinové pulsy (náběžné hrany). Rychlost jsme určovali hodnotou kondenzátoru v obvodu a hodnotou odporu potenciometru. Obvod je na obrázku 4.1.



Obrázek 4.1 - Astabilní zapojení obvodu NE555

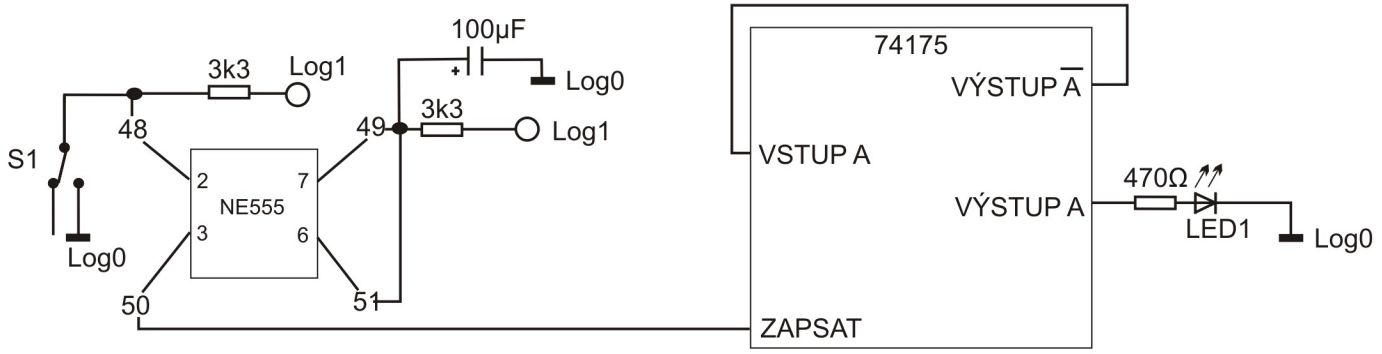
Monostabilní obvod – na výstupu je hodnota log0. Při stisku spínače se na určitou dobu hodnota výstupu změní na log1. Tím se vygeneruje hodinový puls. Po uplynutí času se změní zpět na log0. Schéma obvodu je na obrázku 4.2.



Obrázek 4.2 – Monostabilní zapojení obvodu NE555

Tyto dvě schémata pro získání kvalitního hodinového pulsu pro nás budou podstatná. Existují další zdroje hodinových pulsů, které uvenu na konci návodu jen pro doplnění, ale zase – je toho plný internet, stačí jen „googlit“.

Zapojíme teď sekvenční obvod s IO 74175, který má přiveden na zdířku ZAPSAT hodinový puls z obvodu NE555. Vybereme nejdříve monostabilní zapojení.



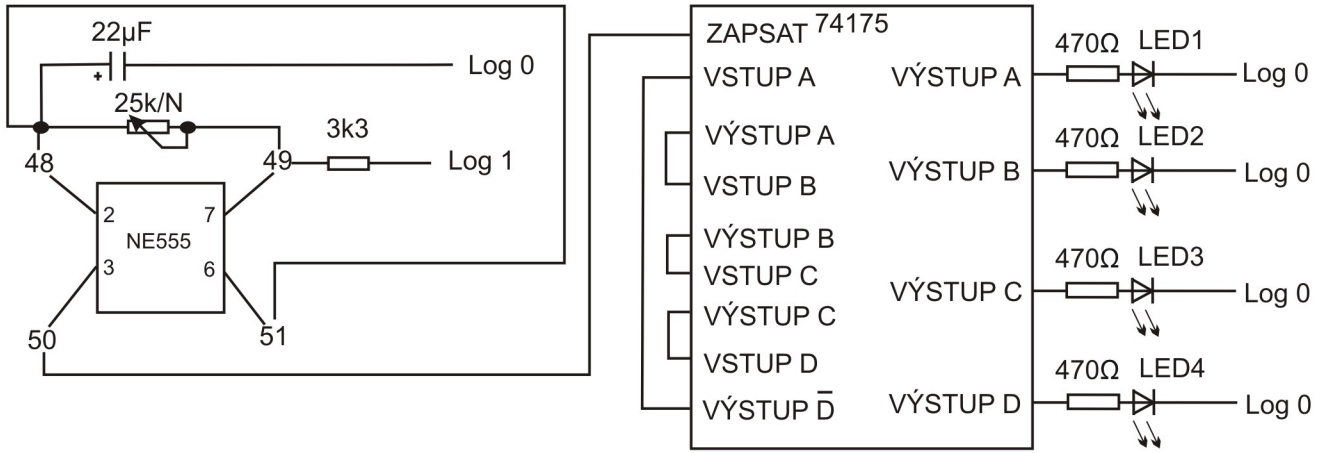
Obrázek 4.3 - Obvod 74175 jako jednoduchý sekvenční obvod s generátorem hodinových pulzů

***Úloha 11: Zapojení**: 10 - 51, 9 – log1, 8 - 48, 7 – log1, 49 - 51, 49 - 43, 48 - 116, 117 - log0, 77 - 78, 79 - 93, 92 -log0 , 42 - log0, 50 - 87*
Zapnout IO 555 a IO74175.

Obvod na obrázku 4.3 už bude fungovat perfektně. Stiskem spínače se bude hodnota výstupu krásně měnit. Pokud místo monostabilního zapojení NE555 jako zdroje hodinových pulzů zapojíte astabilní zapojení, bude to blikat. Ve třetím díle návodu si ukážeme, k čemu je to dobré a budeme stavět velmi jednoduchý počítač.

Pro pochopení účelu stavebnice Saimon je touto kapitolou řečeno vše podstatné a můžete přejít k třetímu dílu návodu, kde se sekvenční obvody a konečné automaty naučíme navrhovat a zapojovat. Další kapitoly jsou pro dobrovolníky, zvědavce, tatínky, co si hrají se stavebnicí a další nadšence. Jsou tam uvedeny další obvody pro získání hodinových pulsů, také tam vysvětluji, co to je klopný obvod a jak funguje, jak si zapojit na stavebnici jedno-bitovou paměť, ale to pro pochopení návrhu našich sekvenčních obvodů není podstatné. I když je toho „plný internet“, uvádím to v tomto návodu pro úplnost.

Obvod, který se mě osobně moc líbí, využívá vlastnosti obvodu v úloze 11. Zapojíme ale všechyn vstupy a výstupy. Výsledkem bude krásný světelný „had“. Funkci obvodu zkuste odhadnout.



Obrázek 4.4 – Světelný had

***Úloha 12: Zapojení**:*

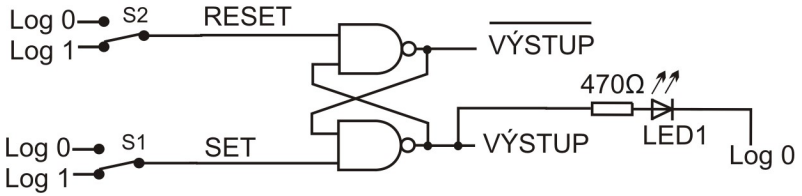
časovač: *49-10 , 9-log1, 48-51 , 23-48 , 24-49 , 48-39 , 38-log0 , 50-87*

integrovaný obvod: 77-85 , 79-76 , 74-83 , 81-84

LEDky: 79-93 , 74-95 , 81-97 , 86-99 , 92-94 , 94-96 , 96-98 , 98-log0

Kapitola 5 – Další tvarovací obvody

Vynikající, a přitom jednoduchý obvod pro kvalitní signál změny z log0 na log1 a obráceně, se jmenuje RS. Znamená to Reset - Set, česky něco jako Vynuluj - Nastav. Jeho schéma je na obrázku 5.1.

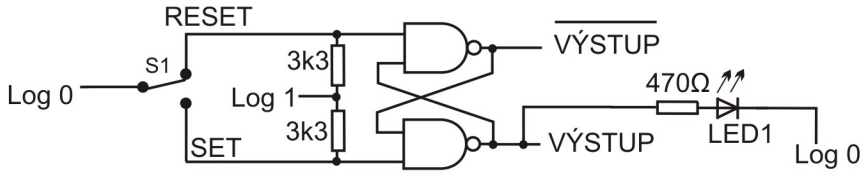


Obrázek 5.1: Obvod RS

***Úloha 13: Zapojení** : 116 - 71, 117 - log0, 120 - log0, 73 - 69, 72 - 70, 68 - 119, 73 - 93, 92 – log0, Zapnout IO-2 7400*

Spínačem S1 obvod natrvalo nastavíme na log1 na výstupu. Jakékoliv zakmitání nemá na výstup vliv, protože obvod se pomocí S1 dá jen zapnout, vypnout už ale ne. K vypnutí, neboli nastavení log0 na výstupu, slouží spínač S2. Který umí obvod zase jen vypnout, takže zákmity nám také nevadí. Výstup je označen dole, na tom spodním NANDu. Na tom horním je výstup přesně opačný, čili když je dole log1, tak nahoře bude log0 (zkuste si na něj přivést třeba LED2, bude svítit opačně k LED1). A ještě poznámka - kdo si všiml, tak to nastavení provádíme logickými nulami, čili po stisknutí například S1 se na vstup, který je označený SET přivede log0 a obvod nastaví na výstupu log1.

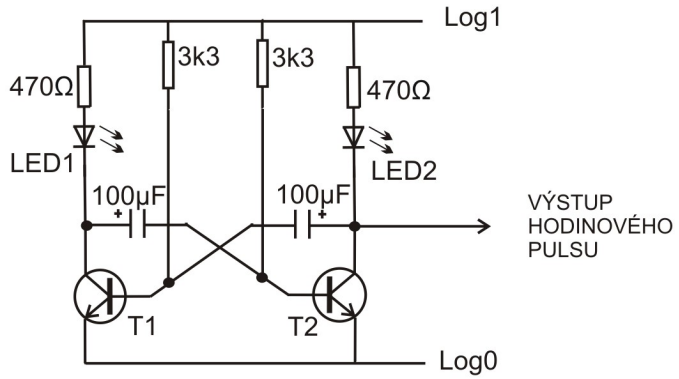
Zapojení z obrázku 5.1 není moc pohodlné, protože musíme mačkat dvě tlačítka. Lepší je ovládat to jen jedním spínačem. Takové schéma zapojení je na obrázku 5.2.



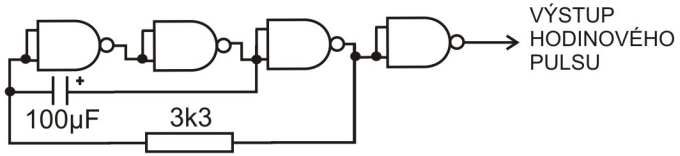
Obrázek 5.2: Obvod RS s jedním spínačem

***Úloha 14: Zapojení** : 116 - log0, 117 - 72, 111 - 72, 73 - 68, 70 - 71, 69 - 110, 69 - 118, 73 - 93, 92 – log0, Zapnout IO-2 7400*

Dál chci uvést, že jako zdroj hodinového signálu se dají použít i blikače a majáky z prvního dílu návodu. Uvedu zde jen schémata, kde bude označen výstup hodinového signálu.



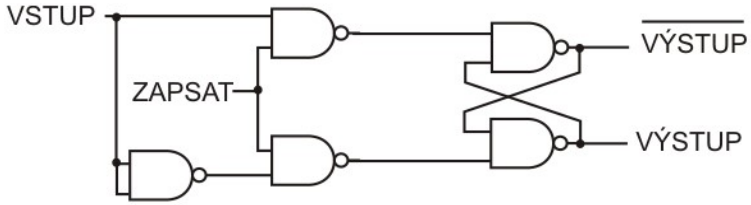
Obrázek 5.3: Tranzistorový blikač jako zdroj hodinového pulsu



Obrázek 5.4: Blikač s obvodem 7400 jako zdroj hodinového pulsu

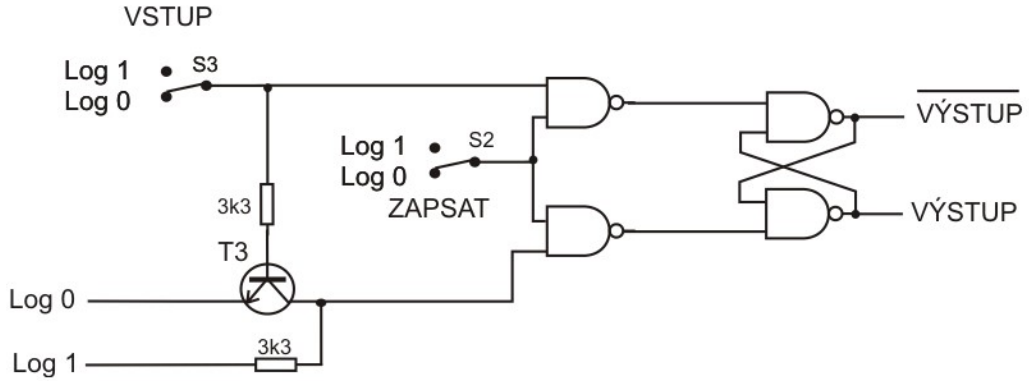
Kapitola 6 - Klopný obvod typu D

Zde chci ukázat, co je vlastně uvnitř IO 74175 a jak si zapojit jeden vstup-výstup (paměťovou buňku) pomocí funkcí NAND. Ve schématu je obvod SR, který už známe, ale je zapojený ještě s dalšími NANDy. Takovému obvodu se říká **D-klopný obvod**. Opět jeden obrázek za tisíc slov, nuže hleďte na obrázek 6.1.



Obrázek 6.1: D-Klopný obvod.

Kdo by si to chtěl zapojit, tak si musíme trochu pomoci, protože NANDy máme na stavebnici jen čtyři, ale zde jich je potřeba pět. Nejlepší bude NAND vlevo, který plní funkci NOT, nahradit obvodem s tranzistorem (obrázek 2.9). Můžete si to zkusit zapojit a vyzkoušet.



Obrázek 6.2: D-Klopný obvod s tranzistorem

Teď je i jasné, proč má IO74175 i opačné výstupy, protože v tom obvodu nám prostě vznikly a je jednoduché je vyvést ven.

Obsah

Slovník pojmů

(přidržením ctrl + klik myší na tučná slovíčka se v návodu přesunete na místo jeho prvního popisu)

Logická 0, Log.0:	Stav vypnuto, nesvítí, není tam napětí.
Logická 1, Log.1:	Stav zapnuto, svítí, je tam napětí.
Dvojková, **binární soustava** čísel:	Počítačový způsob zobrazení čísel pomocí nul a jedniček
Šesnáctková, **hexadecimální soustava** čísel:	Desítková soustava čísel rozšířená o znaky A,B,C,D,E,F.
Desítková, **decimální** soustava čísel:	Soustava vymyšlená pro člověka – má deset prstů, tak se dobře počítá od jedné do deseti.
Bit (angl. BInary digiT)	Nejmenší jednotka zápisu čísla, znamená zapnuto, vypnuto, tvoří ji log 1 nebo log 0
Logická funkce:	Pro nás obvod, který má jeden nebo **více** vstupů a **jeden** výstup. Nastavuje hodnotu výstupu podle hodnot vstupů podle příslušné pravdivostní tabulky
Vstup logické funkce	Místo v obvodu, kam přivádíme hodnotu log0 nebo log1, například pomocí spínačů.
Výstup logické funkce	Místo v obvodu, kde se nastavuje odpovídající logická hodnota podle hodnot vstupů.
Logická funkce **AND**:	Logický součin – výstup nabývá hodnoty log1 pouze pokud oba vstupy mají hodnotu log 1
Logická funkce **OR**:	Logický součet – výstup nabývá hodnoty log1 pokud jeden, nebo oba vstupy mají hodnotu log 1
Logická funkce **NOT**, **negace**:	Logický opak – výstup nabývá opačné hodnoty, než na kterou je nastaven vstup
Logická funkce **NAND**:	Je to logická funkce podobná AND. Výstup je ale opačně. Tak, jako bychom za AND zapojili ještě logickou funkci NOT

Logická funkce **XOR**:	Vylučovací (exkluzivní) OR. Výstup nabývá hodnoty log1 pokud **pouze** jeden ze vstupů nabyl hodnoty log1
pravdivostní tabulky:	Tabulky, v kterých jsou napsány hodnoty vstupů (log0 nebo log1) a odpovídající výstupy podle příslušné logické funkce, ke které je tabulka napsána.
TRUE:	Česky „pravda“, logická 1, stav napětí.
FALSE:	Česky „nepravda“, logická 0, stav bez napětí.
Hodinový pulz, **Náběžná hrana**:	Změna z hodnoty log0 na hodnotu log1. Musí proběhnout jen jednou.