Milý řešiteli,

vítáme Tě u 3. série úloh 6. ročníku korespondenčního semináře MoRoUS. Jako v minulých letech i nyní Ti přinášíme úlohy ze světa profesora Morouse a jeho robotických společníků.

Řešení úloh Ti umožní nejen nahlédnout do zajímavých zákoutí umělé inteligence a robotiky, ale také zúčastnit se jarního soustředění s inspirativními lidmi, obzor rozšiřujícími přednáškami a nezapomenutelnými zážitky ;) A na nejlepší řešitele čekají na konci soutěže tematické ceny!

3. série 2019/2020

Termín odeslání 3. série: **22. 3. 2020**

Kam posílat řešení?

Až budeš mít řešení hotové, pošli nám je včetně všech nákresů, prográmků a prostě všeho co by nám usnadnilo opravování Tvé úlohy. Stačí, když pošleš řešení jen některých úloh nebo jejich částí, i k těm Ti pošleme komentáře a přičteme za ně body ;)

Řešení posílej e-mailem na adresu `seminar@morous.fel.cvut.cz`

Úloha č. 1: Dělba práce (20 bodů)

Po prázdninách jsou roboti nasazeni do práce a profesor Morous jim potřebuje předat časový rozvrh. Časovým rozvrhem myslíme počet hodin, které má každý z robotů odpracovat, tedy jedná se o vektor čísel o délce rovnající se počtu robotů. Pro každou továrnu, kde roboti pracují, má Morous poznamenaná dvě přirozená čísla M a N. M udává, kolik je v dané továrně třeba odpracovat celkem hodin, N udává kolik je v dané továrně k dispozici robotů.

Úkol 1.1

Každý robot v továrně musí pracovat minimálně 1 a maximálně 9 hodin. Počet pracovních hodin musí být celé číslo. Aby bylo možné přiřadit různě opotřebené roboty, různě dlouhým směnám, nesmí být v jednom časovém rozvrhu více robotů, kteří by pracovali stejný počet hodin. Pro $M = 30$, $N = 6$ vypište všechny použitelné časové rozvrhy. Permutace rozvrhu považujeme za stejný rozvrh, např. $[1, 2, 9]$, $[1, 9, 2]$, $[2, 1, 9]$ atd. je pro nás pouze jeden rozvrh.

Pozn.: Všechny použitelné časové rozvrhy pro $M = 12$, $N = 3$:

$$[1, 2, 9][1, 3, 8][1, 4, 7][1, 5, 6][2, 3, 7][2, 4, 6][3, 4, 5]$$

$[1, 2, 9]$ *je rozvrh kde jedna směna trvá 1 hodinu, jedna 2 hodiny a jedna 9 hodin. Je splněno $1 + 2 + 9 = M = 12$, všech $N = 3$ robotů je zaměstnaných,* ***žádný počet hodin v rozvrhu není více než jednou****. Jedná se o validní rozvrh. Všimněte si, že např. pro $M = 10$, $N = 1$, použitelný rozvrh neexistuje.*

Úkol 1.2

Za necelou hodinu se na profesor Morouse budou valit tisíce požadavků od robotů, aby jim pro konkrétní M a N vypsal všechny použitelné rozvrhy. Popište obecný postup (algoritmus) jak pro dané M a N vypsat všechny použitelné rozvrhy (nebo vypsat že použitelný rozvrh pro dané M, N neexistuje). Jak má profesor postupovat aby byl na dotazy robotů schopný vypisovat rozvrhy co nejrychleji?

Pozn.: Řešení úlohy by měl být postup, stejně důležité jako postup samotný je uvést ***proč váš postup funguje*** *a* ***jak jste k němu dospěli****.*

Úkol 1.3

Mějme zadání jako v úkolu 1, pouze s rozdílem, že může v jednom rozvrhu být libovolný počet robotů se stejnou dobou práce.

Příklad použitelných časových rozvrhů pro $M = 12$, $N = 3$:

$$[1,2,9][1,3,8][1,4,7][1,5,6][2,2,8][2,3,7][2,4,6][2,5,5][3,3,6][3,4,5][4,4,4]$$

tedy v našem ukázkovém příkladu existuje 11 použitelných rozvrhů.

Tvým úkolem je zjistit, kolik jedinečných použitelných rozvrhů existuje pro $M = 25$, $N = 4$?

Úkol 1.4

Mějme zadání jako v úkolu 3, napište obecný postup, kterým v co „nejrychlejším čase" zjistíme počet použitelných časových rozvrhů pro dané M a N.

Pozn.: „Nejrychlejší čas" – jde nám o čas v případech pro hodně velká M a N, viz ASYMPTOTICKÁ SLOŽITOST, dva různé postupy z nichž jeden bude 2× rychlejší budou hodnoceny bodově velmi podobně. Naopak postup který pro malá M, N bude bleskově rychlý a pro velká M, N (řádově stovky) by zabral tisíce let, bude hodnocený hůře.

*Pozn.: Řešení úlohy by měl být postup. Stejně důležité jako postup samotný je uvést, **proč váš postup funguje** a **jak jste k němu dospěli**.*

Úkol 1.5

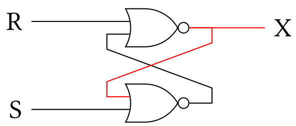
Mějme podobné zadání jako v úkolu 1, pouze s rozdílem, že počet pracovních hodin může být číslo racionální (tedy robot může opracovat např. 1,25 hodiny). Stále platí, že robot je omezen minimem 1 a maximem 9 hodin a že v časovém rozvrhu není více robotů se stejnou dobou práce. Kolik existuje různých použitelných časových rozvrhů v závislosti na M a N. Dokažte.

Úloha č. 2: Z kroniky doktora Daystrőma (20 bodů)

V minulém díle

Doktoru Daystrőmovi se omylem podařilo vyrobit tzv. ***klopný obvod****, přesněji obvod bistabilní (existují dva různé stavy, ve kterých se obvod může ustálit). Nejčastější použití těchto obvodů je v paměťových prvcích.*

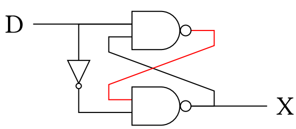
Jeden z nejzákladnějších obvodů je typ RS. Jeden z vývodů – R – funguje jako funkce reset, tj. vynuluje paměť nezávisle na předchozí hodnotě. Druhý – S – zase nastavuje (set) jedničku. Pokud ani jeden ze vstupů není aktivní, obvod si pamatuje předchozí stav. Celou funkci lze zapsat i zjednodušenou (úspornější) pravdivostní tabulkou, kde jako X_n *označíme aktuální stav a* X_{n+1} *stav nový. Jak můžeme vidět, v této variantě má vstup R větší prioritu než S.*



R	S	X	X
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

R	S	X_{n+1}
0	0	X_n
0	1	1
1	0	0
1	1	0

Protože kontrolovat obvod pomocí dvou různých vstupů může být nepraktické, lze drobnou modifikací získat klopný obvod typu D, který na svém výstupu přesně kopíruje vstup.

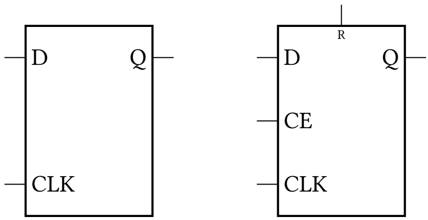


D	X	X
0	0	0
0	1	0
1	0	1
1	1	1

D	X_{n+1}
0	0
1	1

Na první pohled se toto může zdát jako neužitečná součástka, protože seberychlejší změna na vstupu se objeví i na výstupu a celý obvod se tedy vlastně chová tak, jako by tam vůbec nebyl. To ale Daystrőm napravil tím, že začal používat předřadné „synchronizační" hradlo. V reálných obvodech jsou všechny důležité části synchronizovány společným pulsem (nejčastěji tzv. hodinovým – v celém obvodu je jedna součástka, která v pravidelných intervalech tiká, posílá impuls). Klopné obvody tak například mění svůj stav pouze během krátkého synchronizačního pulsu a jinak se nenechají ovlivnit tím, co se děje na vstupu. A ani to ještě nemusí být finální stav, i synchronizační puls může být záhodno občas vypnout, takže jsou některé obvody vybaveny krom hodinového vstupu (CLK – clock) i vstupem pro povolení hodinového pulsu (CE – clock enable). Krom toho je někdy potřeba asynchronně (nezávisle na hodinovém pulsu) obvod „vynulovat“ (R – reset). V této podobě je obvod používán jako základ mnoha složitějších obvodů, které potřebují paměťové buňky.

Od této chvíle si tedy můžeme dovolit považovat paměťovou buňku jako jednu součástku a ve schématech ji používat jako základní stavební blok. Podle označení vstupů pak můžeme odhadnout, o jak složitou variantu se jedná a které z pokročilých funkcí potřebujeme.



Kombinací paměťového obvodu a kombinačního obvodu pak může získat mnohem „chytřejší“ obvody, kde stav obvodu už nezávisí jen na aktuální kombinaci vstupů, ale také na aktuálním „vnitřním“ stavu (ten je reprezentován právě paměťovými obvody). Takovému obvodu se pak říká **sekvenční obvod**.

Jak by tedy vypadal Daystrőmův čítač z minule? Potřebujeme si pamatovat dva bity, takže využijeme dva klopné obvody typu D. K nim už jen potřebujeme kombinační část, která bude umět zvětšovat a zmenšovat dva bity podle vstupních instrukcí. Nejlépe uděláme, když si nejprve zapíšeme tabulku, jak se má obvod chovat (hodnota x vyjadřuje libovolnou hodnotu, použijeme takovou, co nám nejvíce zjednoduší návrh).

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

A	B	X	Y	X	Y
1	0	0	0	0	1
1	0	0	1	1	0
1	0	1	0	1	1
1	0	1	1	0	0
1	1	0	0	x	x
1	1	0	1	x	x
1	1	1	0	x	x
1	1	1	1	x	x

Dalším krokem bude vymyslet funkce pro proměnné X a Y. Jedna z metod je vyrobení tzv. Karnaughovy mapy (po jedné pro každou proměnnou). Ve chvíli, kdy budeme mít matematický popis logických funkcí pro X a Y, je již sestavení kombinačního obvodu hračka.

AB \ XY	00	01	11	10
00	0	0	1	1
01	1	0	1	0
11	x	x	x	x
10	0	1	0	1

AB \ XY	00	01	11	10
00	0	1	1	0
01	1	0	0	1
11	x	x	x	x
10	1	0	0	1

$$X = B(X \oplus \overline{Y}) + A(X \oplus Y) + \overline{A}\overline{B}X$$

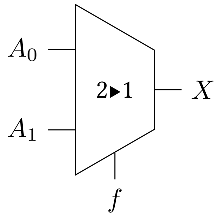
$$Y = A\overline{Y} + \overline{A}(B \oplus Y)$$

Pokračování příběhu

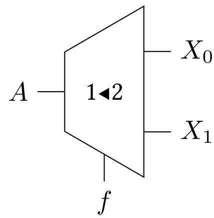
V Daystrőmově institutu bylo dnes až neobvykle živo, většina zaměstnanců slavila výročí založení. Doktor Daystrőm však mezi nimi nebyl, byť to byl on, kdo institut před více než třiceti lety založil. Doktor neměl oslavy rád a mnohem raději byl zalezlý ve své laboratoři a pracoval. Ani dnes tomu nebylo jinak. Spolu se svou novou asistentkou finalizoval návrh nového počítače řady Nästan Fyra Dator. K jeho dokončení mu však ještě chybělo několik kriticky důležitých součástek.

Nejprve Daystrőm věnoval úsilí k vyrobení součástky, která mu pomocí řídícího signálu f umožní vybrat ze dvou různých vstupů (A_0 a A_1) jeden, který se dostane na výstup (X) – tj. výběr 1 ze 2, součástku označíme jako 2▸1. Hned poté se vrhl na součástku opačnou, která umožňovala jeden vstupní signál (A) poslat do právě jednoho ze dvou výstupních vodičů (X_0 nebo X_1) – označíme ji jako 1◂2. O výběru opět rozhoduje signální vstup f.

A_0	A_1	f	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1



A	f	X_0	X_1
0	0	0	0
0	1	0	0
1	0	1	0
1	1	0	1



Úkol 2.1

Navrhni Daystrőmovy součástky ve verzi 4▸1 a 1◂4 pomocí logických hradel.

Pozn.: Pro rozhodování se mezi 4 vývody budeš potřebovat dva signální vodiče (f_0 a f_1).

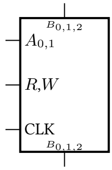
Další v pořadí byl paměťový obvod. Daystrőm vzal 12 klopných obvodů a uspořádal je do mřížky, tři obvody ve čtyřech řádcích. Jeho počítač si teď bude moci zapamatovat celá 4 slova. Je ale potřeba všechny obvody pospojovat do jednoho velkého paměťového čipu. Doktor začal přemýšlet, jak toho nejlépe docílit.

Protože slova budou mít tři bity (v řádku jsou tři obvody – paměťové buňky), k celému čipu povedou tři vodiče, které budou plnit funkci hlavní sběrnice (B_0, B_1, B_2). Z této sběrnice si paměťový čip bude brát informaci o tom, co se má do paměti uložit (ke každé buňce v řádku tedy bude propojka s jedním z vodičů ze sběrnice) a naopak, když bude potřeba z paměti číst, paměťové buňky budou svůj výstup promítat na sběrnici. Jistě bude potřeba sběrnici od paměti nějak vhodně oddělit, aby se informace ze sběrnice nebo naopak na sběrnici nedostala, když nemá (řídící signál pro čtení R a pro zápis W).

Protože v čipu jsou celkem čtyři různé řádky, musíme nějakým způsobem rozlišit, do kterého řádku chceme zapisovat (případně ze kterého chceme číst). Jak už jistě tušíš, na výběr ze 4 možností budeme potřebovat dva adresační bity (A_0, A_1).

Úkol 2.2

Nakresli vnitřní schéma paměťového čipu.



Pozn.: Můžeš používat libovolné součástky, které již v tuto chvíli Daystrőm zná a používá. Jistě se Ti krom obyčejných logických hradel budou hodit klopné obvody D (dokonce si vystačíš s nejobyčejnější tří-vývodovou verzí) a součástky 1◂4.

Už jen chybělo „srdce“ celého stroje. Daystrőm nejprve vzal svoji sčítačku a provedl drobnou modifikaci. Ve volné chvíli dodal ještě pár dalších obvodů a vše zapojil do sebe.

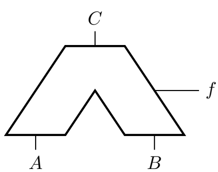
Vezmeme sčítačku z minulé úlohy a uděláme úpravu, která umožní sčítat dvě tříbitová čísla A a B. Dále potřebujeme obvod, který umí od odčítat (realizuje funkci $B - A$). Poté chceme mít obvod, který provádí výběr vyššího čísla (tj. operace $max(A, B)$). Jako poslední se bude hodit operace B (tedy jen a pouze zkopírování druhého vstupu na výstup).

Tyto čtyři kombinační obvody dáme vedle sebe, do každého z nich povedou tři vodiče pro operand A (A_0, A_1, A_2) a tři vodiče pro operand B (B_0, B_1, B_2). Obdobně, z každého budeme mít tři vodiče pro výsledek operace C (C_0, C_1, C_2).

Výběr operace bude řídit signál f (f_0, f_1), který zvolí, který z výstupů C se má opravdu použít.

Úkol 2.3

Nakresli vnitřní schéma aritmeticko-logické jednotky.



Pozn.: Tentokrát se Ti jistě budou hodit logická hradla na výrobu kombinačních obvodů a součástky 4▸1, případně 2▸1.

Úkol 2.4

Pokus se aritmeticko-logickou jednotku zjednodušit (zmenšit počet použitých součástek) tím, že jednotlivé operace nebudou v jednotce jako samostatné kombinační obvody (tj. například sjednocením některých operací do jedné).