<table>
<tr><td colspan="4">Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie
LABORATORIUM Teoria Automatów</td></tr>
<tr><td colspan="2">Temat ćwiczenia</td><td colspan="2">Przejazd kolejowy</td></tr>
<tr><td colspan="2">Grupa ćwiczeniowa:</td><td colspan="2">Poniedziałek 8.00</td></tr>
<tr><td>L.p.</td><td>Imię i nazwisko</td><td>Ocena</td><td>Podpis</td></tr>
<tr><td>1.
2.
3.
4.</td><td>Krzysztof Wesołowski
Paweł Górka
Łukasz Bondyra
Jakub Tutro</td><td></td><td></td></tr>
<tr><td colspan="3">Data wykonania ćwiczenia:</td><td>17.11.2008</td></tr>
</table>

Wstęp:

Celem ćwiczenia było zaprojektowanie układu logicznego, który kierowałby semaforami i szlabanem na przejeździe kolejowym. Do układu wchodziły sygnały z trzech czujników, jeden znajdował się na samym przejeździe (czujnik b), zaś 2 przed nim, po każdej ze stron (czujniki a i c). Założeniem było, iż pociąg nie może się cofać, w jednej chwili na przejeździe może się znaleźć jeden pociąg, który może wjeżdżać na przejazd z dowolnej strony.

Przygotowanie teoretyczne

Po przyjęciu $d = a \vee c$ (jako że sygnał od każdego z czujników wjazdu pociągów ze względu na symetrię niesie taką samą informację różnią się jedynie strona od której nadjeżdża pociąg) otrzymaliśmy następującą tabelę pierwotną automatu:

Q \ db	Wejście automatu				Y – wyjście automatu
	00	01	11	10	
0	**0**	-	-	1	0
1	2	-	6	**1**	1
2	**2**	3	-	-	1
3	4	**3**	6	-	1
4	**4**	-	-	5	0
5	0	-	-	**5**	0
6	-	3	**6**	5	1

gdzie:
Q – obecny stan
db – wartości odczytane z czujników (opisane powyżej)
Y – wyjście – czy szlaban będzie podniesiony
Pogrubione liczby oznaczają stany stabilne.

Tabela ta przedstawia automat Moore'a – wyjścia są zależne wyłącznie od stanu automatu. W celu jego wykonania musielibyśmy użyć 3 komórek pamięci (do zakodowania 6 stanów). Aby zmniejszyć ilość elementów automat można zminimalizować – połączyć niektóre stany w jeden stan odpowiednio reagujący na sygnały wejściowe.

Przy minimalizacji możemy minimalizować automat zarówno do automatu Moore'a jak i Mealy'ego, jednak w tej sytuacji lepsze okazało się zastosowanie automatu Mealy'ego – udało się zmniejszyć ilość stanów do 4 co pozwoli zapamiętywać stan w dwóch jednostkach pamięci.
Po zminimalizowaniu automat posiadał 4 stany, które od razu postanowiliśmy zakodować za pomocą bitów:

Q -stan(y)	q1, q2 – bity pamięci
0	00
1,2	01
3,4,6	11
5	10

Po wpisaniu kodować (powyższych) i połączonych stanów tabela minimalnego automatu Moore'a wygląda tak:

q1, q2 \ db	00	01	11	10
00	**00,** 0	-, -	-, -	01, 1
01	**01** , 1	11, 1	11, 1	**01,** 1
11	**11,** 0	**11**, 1	**11,** 1	10, 0
10	00, 0	**-, -**	-, -	**10,** 0

W każdej komórce zapisujemy stan do którego ma przejść automat (zakodowany bitowo), oraz po przecinku wyjście które ma być ustawione dla stanu po lewej i wejścia powyżej.

Kolejnym krokiem jest rozpisanie tabel dla q1', q2' oraz Y'. Tabele te będą przedstawiać nowe wartości bitów stanu (q1 i q2) oraz wyjście dla danego stanu i wejścia. Tworzymy je przepisując odpowiednie części pół z powyższej tabelki.

Tabelka dla q1':

q1, q2 \ db	00	01	11	10
00	0	-	-	0
01	0	1	1	0
11	1	1	1	1
10	0	-	-	1

Pokryliśmy pola tabelki tak aby otrzymać jedynki i zera tam gdzie są wymagane. Poza tym wykorzystaliśmy myślniki – pola o wartości dowolnej (gdyż w rzeczywistości sytuacja w której by wystąpiły się nie może zdarzyć) do stworzenia najprostszych funkcji logicznych. Powyższa tabela, zwana tabelą Karnaugha pozwala w prosty sposób uzyskać najprostsze funkcje logiczne. Im większy jest prostokąt o będącej potęgą dwójki ilości pól z jedynką tym prostsza postać funkcji logicznej. W tym wypadku pozostaje nam zapisanie każdego pokrycia jako prostej funkcji logicznej (czyli koniunkcji odpowiednich wyrazów pochodzących z

pierwszego wiersza/kolumny) i zsumowanie ich za pomocą alternatywy.
Otrzymana funkcja to: $q_1' = b \vee q_1 q_2 \vee d q_1$

Tabelka dla q2’:

db / q1, q2	00	01	11	10
00	0	-	-	1
01	1	1	1	1
11	1	1	1	0
10	0	-	-	0

W powyższej tabelce (którą można znaleźć również w instrukcji), poza podanymi (brązowymi) pokryciami trzeba było uwzględnić też pokrycie czerwone. Istniał tu bowiem hazard statyczny jedynki występujący gdy w stanie kodowanym 01 wejście zmieniało się 10<->00 (należy pamiętać że w tabeli Karnaugha skrajne pola są swoimi sąsiadami). Jako że nasz automat jest automatem asynchronicznym i wszystkie zmiany występują jako prawie natychmiastowa reakcja na zmianę wejścia, bez likwidacji tego hazardu występowało zjawisko wyścigu, powodujące niepoprawne działanie automatu.
Po naniesieniu poprawek pokrycia zapisaliśmy funkcją: $q_2' = b \vee d\overline{q_1} \vee \overline{d} q_2 \vee \overline{q_1} q_2$.

Tabelka dla Y:

db / q1, q2	00	01	11	10
00	0	-	-	1
01	1	1	1	1
11	0	1	1	0
10	0	-	-	0

Tutaj bez niespodzianek, zapisujemy przejścia miedzy stanami za pomocą funkcji logicznej:
$Y = b \vee \overline{q_1} q_2 \vee d\overline{q_1}$

Realizacja ćwiczenia

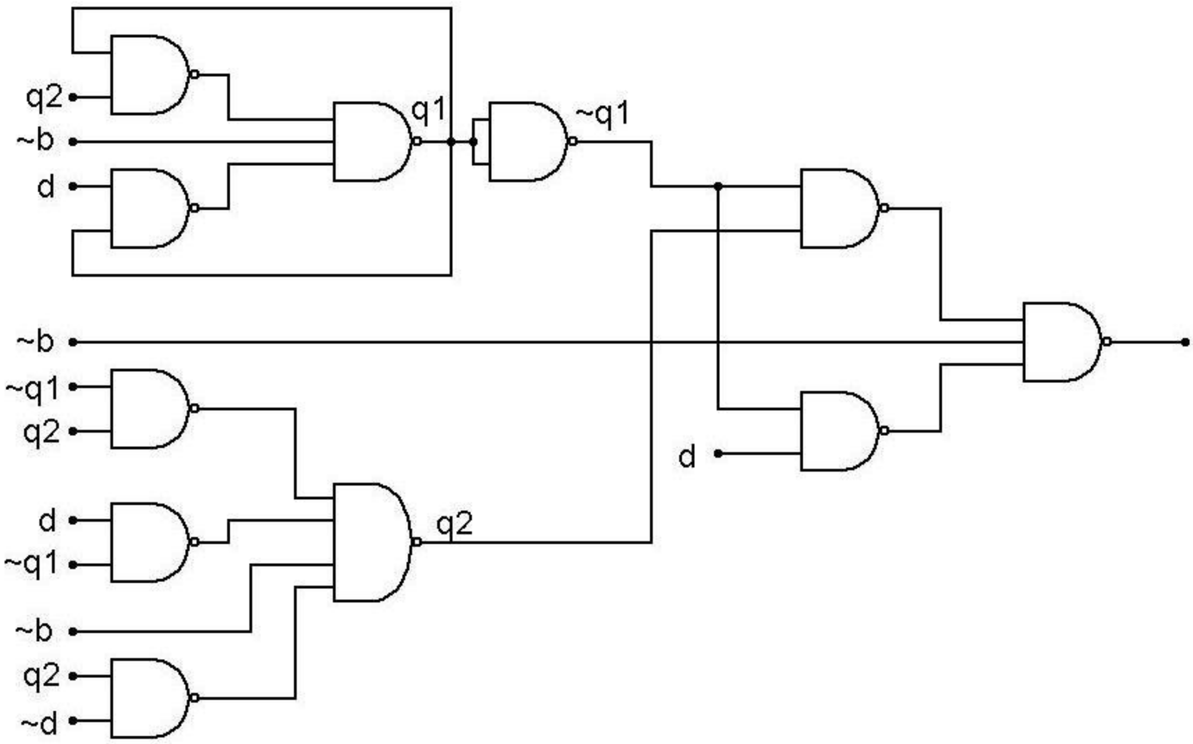
Układ w trakcie zajęć jest realizowany na bramkach NAND, które są najwygodniejsze do stosowania w pracowni – za ich pomocą można zapisać każdą funkcję logiczną (tworzą system funkcjonalnie pełny). W związku z tym rozpisujemy nasze funkcje korzystając z praw de’Morgana oraz zaprzeczeń (realizowanych za pomocą bramki NAND z spiętymi nóżkami) tak aby składały się tylko z zaprzeczonych koniunkcji:

$$q_1' = b \vee q_1 q_2 \vee d q_1 = \overline{\overline{b} \wedge \overline{q_1 q_2} \wedge \overline{d q_1}}$$

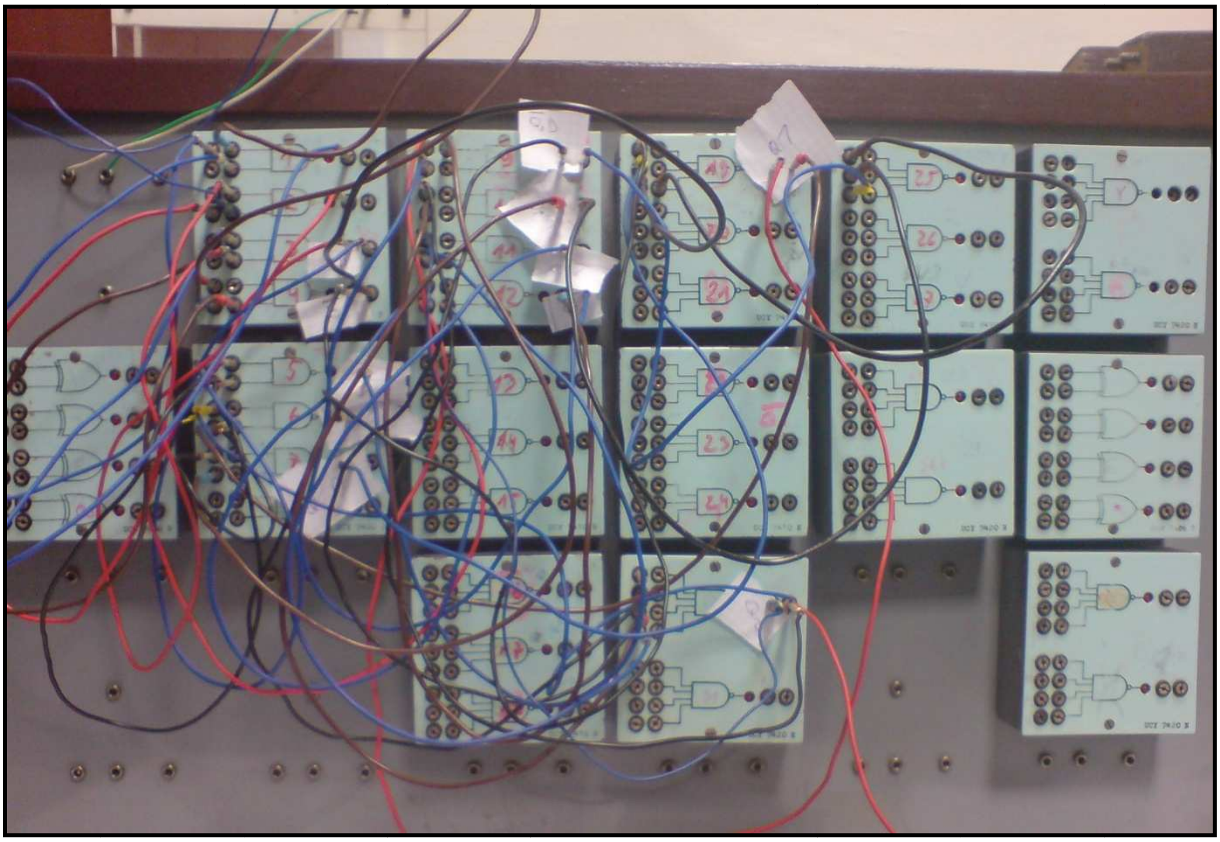
$$q_2' = b \vee d\overline{q_1} \vee \overline{d} q_2 \vee \overline{q_1} q_2 = \overline{\overline{b} \wedge \overline{q_2 \overline{q_1}} \wedge \overline{d \overline{q_1}} \wedge \overline{\overline{d} q_2}}$$

$$Y = b \vee \overline{q_1} q_2 \vee d\overline{q_1} = \overline{\overline{b} \wedge \overline{\overline{q_1} q_2} \wedge \overline{d \overline{q_1}}}$$

Wykorzystując powyższe funkcje stworzyliśmy schemat połączeń w naszym układzie. W celu zachowania przejrzystości sygnały zanegowane (które powstały w innych miejscach układu/uzyskaliśmy spinając nóżki bramki NAND) zapisaliśmy po prostu jako zanegowane wejścia.



W następnej kolejności połączyliśmy bramki za pomocą przewodów, pamiętając o wyłączeniu zasilania przed łączeniem bramek. Wyjścia z „krytycznych”, często używanych bramek opisaliśmy karteczkami, co pozwoliło zachować jako-taką czytelność takiej realizacji.



Wnioski z wykonanego ćwiczenia

Przy użyciu bramek logicznych typu NAND zamodelowaliśmy zadany układ. Automat pracował prawidłowo, jednak przy włączeniu go do zasilania wymagał zresetowania, ponieważ ustawiał się na stanie $q_1 q_2 = 11$. Można by taki „reset" zrealizować jako układ inicjujący stan w bramkach logicznych, jednak nie było to przedmiotem tego ćwiczenia.

W powyższym ćwiczeniu nauczyliśmy się projektować układ według opisu słownego. Na podstawie zadanego problemu stworzyliśmy automat wykonujący odpowiednie zadania oraz zapoznaliśmy się z pojęciem sprzężenia zwrotnego, wykorzystywanego we wszystkich automatach sekwencyjnych. Wyciągając wnioski z poprzednich ćwiczeń, zauważyliśmy również, iż powyższy układ można zaprojektować w sposób prostszy i bardziej niezawodny przy użyciu przerzutników synchronizowanych zegarem. W obecnym układzie ich rolę pełni układ bramek ze sprzężeniem zwrotnym.