# Budowa sterownika do wyświetlacza 7-segmentowego wykorzystując multipleksery

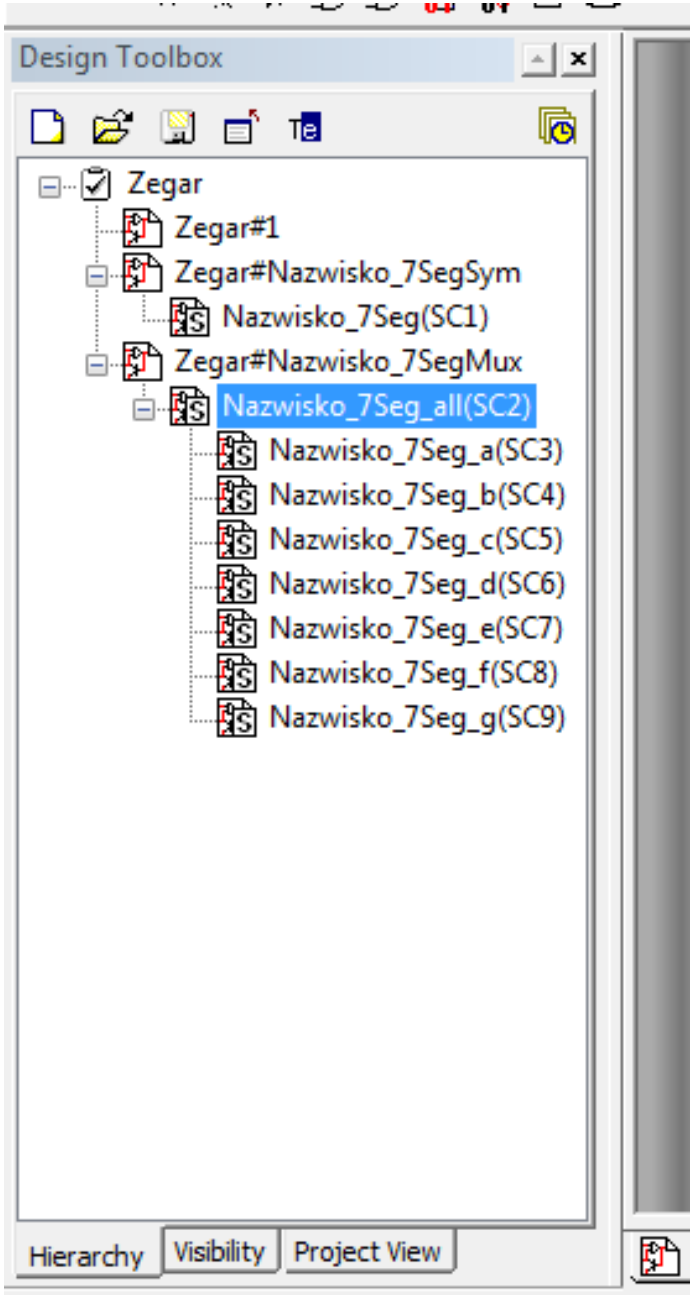
**1.** Należy otworzyć projekt z poprzedniego
ćwiczenia o nazwie „Zegar.ms13”

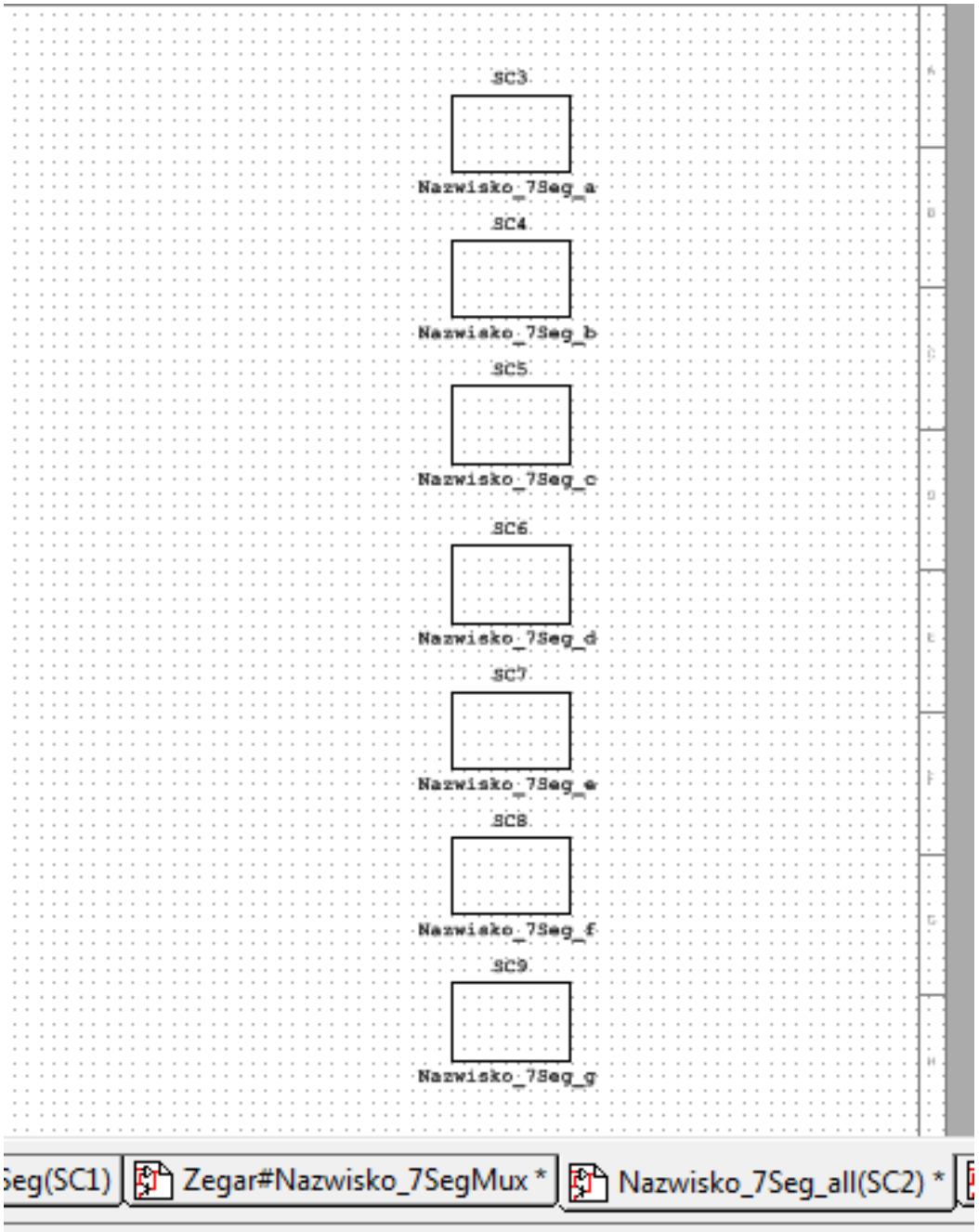
**2.** Stworzyć kolejną stronę „Multi-page”
zakończoną „…_7SegMux”

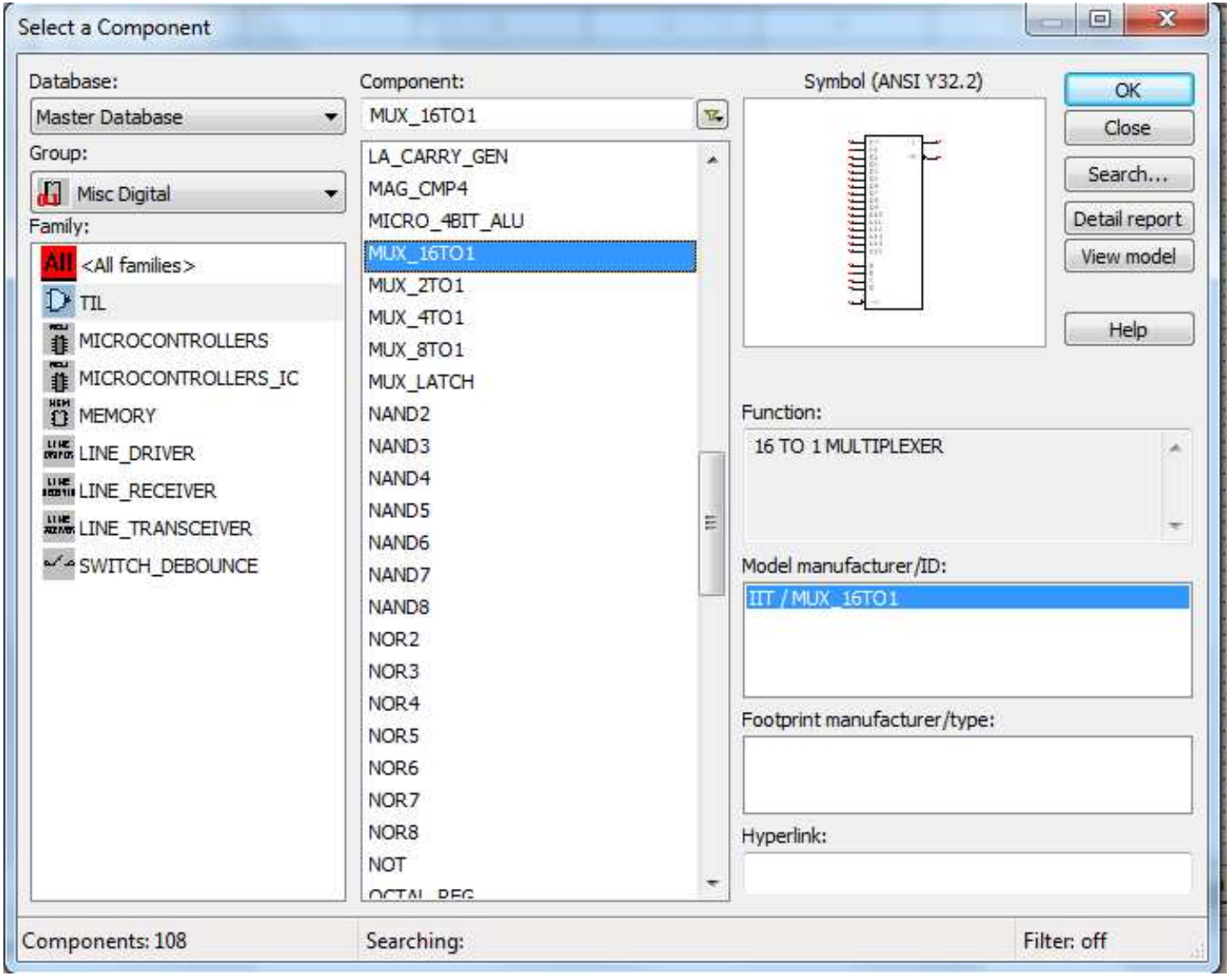
**3.** Stworzyć „New subcircuit”: zakończony „…_7Seg_all”,
który zawierać  będzie 7 multilplekserów
(po jednym na segment wyświetlacza)

**4.** Stworzyć 7 podobwodów zakończonych „…_7Seg_a”;
„…_7Seg_b”; „…_7Seg_c”; itd.

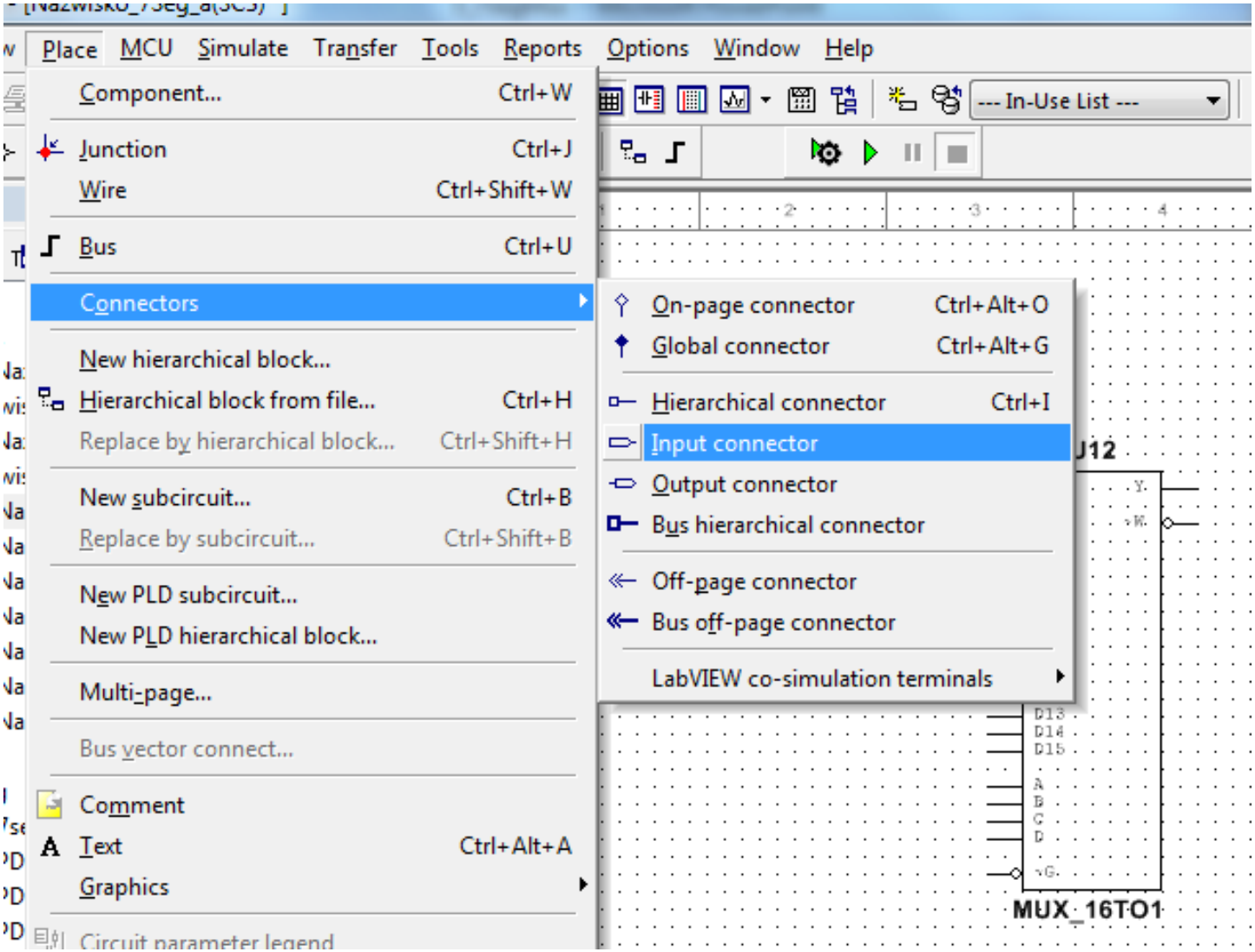
Uwaga! Każda nazwa strony zaczyna się od nazwiska
i każdy symbol dodany na schemacie ma zawierać nazwisko
(należy edytować symbol)


Design Toolbox
Zegar
Zegar#1
Zegar#Nazwisko_7SegSym
Nazwisko_7Seg(SC1)
Zegar#Nazwisko_7SegMux
Nazwisko_7Seg_all(SC2)
Nazwisko_7Seg_a(SC3)
Nazwisko_7Seg_b(SC4)
Nazwisko_7Seg_c(SC5)
Nazwisko_7Seg_d(SC6)
Nazwisko_7Seg_e(SC7)
Nazwisko_7Seg_f(SC8)
Nazwisko_7Seg_g(SC9)
Hierarchy
Visibility
Project View


SC3
Nazwisko_7Seg_a
SC4
Nazwisko_7Seg_b
SC5
Nazwisko_7Seg_c
SC6
Nazwisko_7Seg_d
SC7
Nazwisko_7Seg_e
SC8
Nazwisko_7Seg_f
SC9
Nazwisko_7Seg_g
eg(SC1)
Zegar#Nazwisko_7SegMux *
Nazwisko_7Seg_all(SC2) *



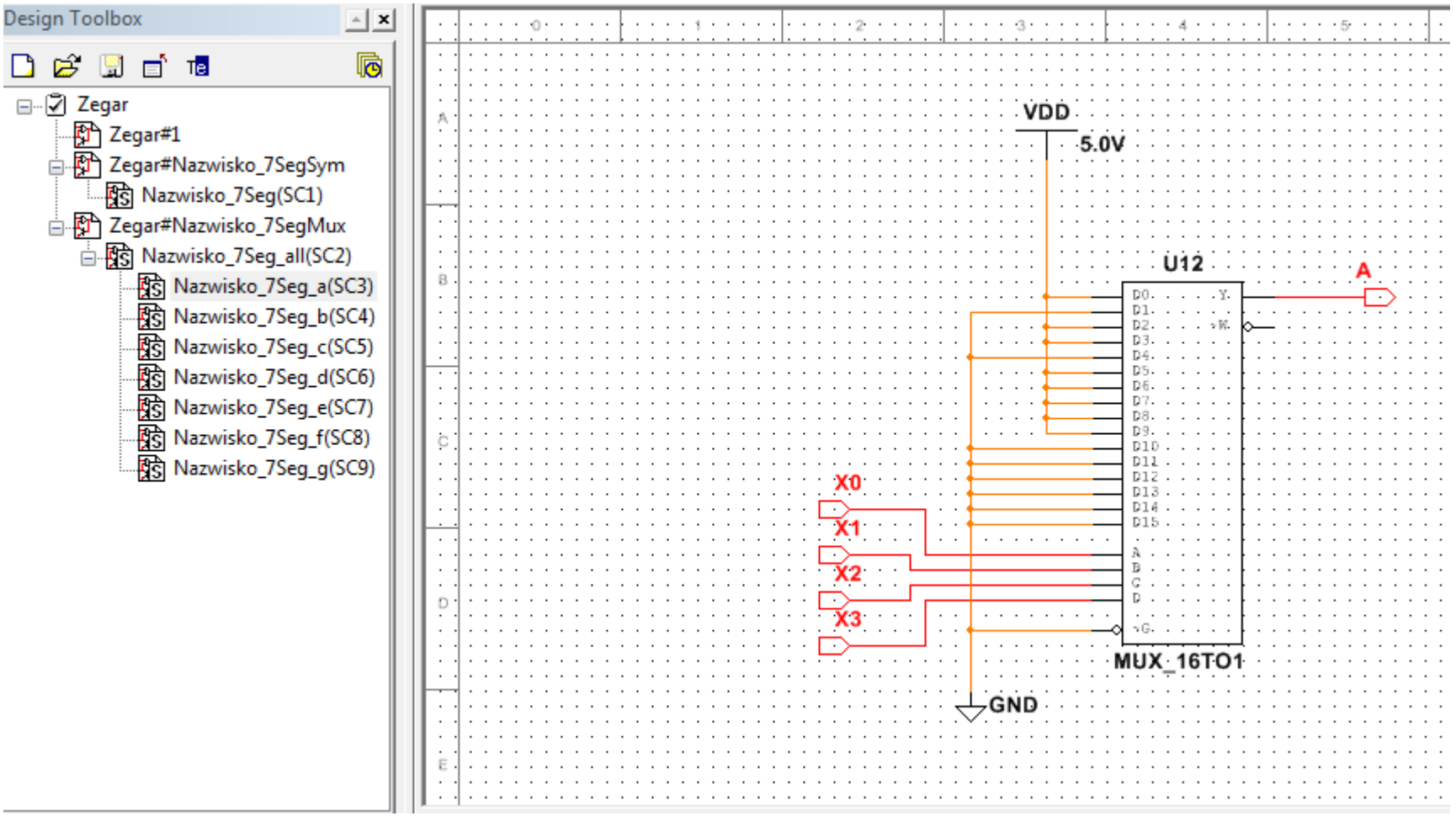

Elementy cyfrowe – multipleksery 16 na 1




Należy dodać do każdego z siedmiu segmentów (a-g) 4 wejścia (X0-X3 -> „Input connector”) oraz 1 wyjście (A-G -> „Output connector”)

| $x_3$ | $x_2$ | $x_1$ | $x_0$ | a | b | c | d | e | f | g |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | x | x | x | x | x | x | x |
| 1 | 0 | 1 | 1 | x | x | x | x | x | x | x |
| 1 | 1 | 0 | 0 | x | x | x | x | x | x | x |
| 1 | 1 | 0 | 0 | x | x | x | x | x | x | x |
| 1 | 1 | 1 | 0 | x | x | x | x | x | x | x |
| 1 | 1 | 1 | 1 | x | x | x | x | x | x | x |

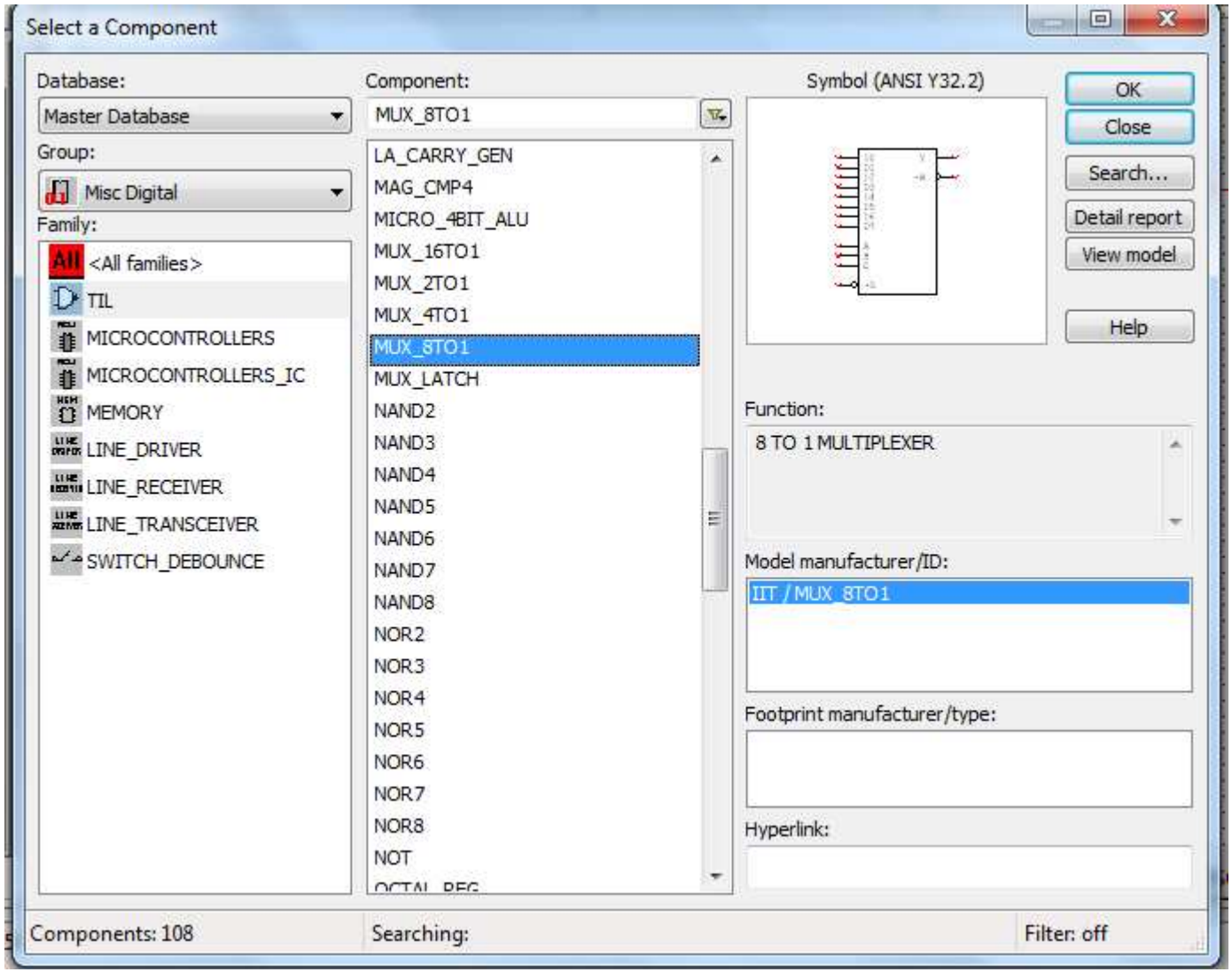
**Tablica prawdy**




Jak powinien prezentować się segment „a”

**5**. Analogicznie zbudować segmenty b-e. Symbol każdego segmentu musi zawierać nazwisko.

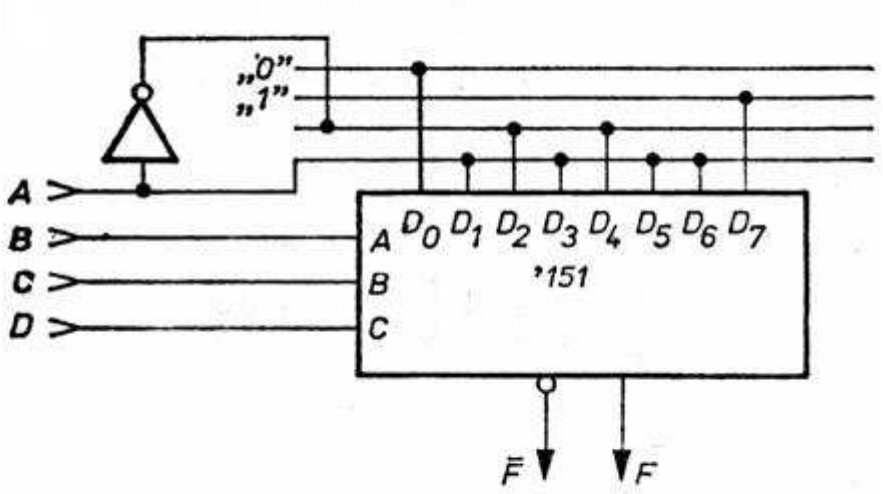
**6.** Segment f wykonać używając multipleksera 8 na 1 (3 wejścia sterujące):



Będziemy dodatkowo potrzebować bramkę inwertera NOT. Na wejścia adresowe podajemy X3 ÷ X1. Wejścia danych łączymy z poziomem wysokim, niskim, sygnałem X0 lub z zanegowanym sygnałem X0. Poniżej przedstawiono przykład realizacji funkcji logicznej na multiplekserze.

Poniższa tabela nie dotyczy segmentu F - przypadkowa zbieżność oznaczeń!

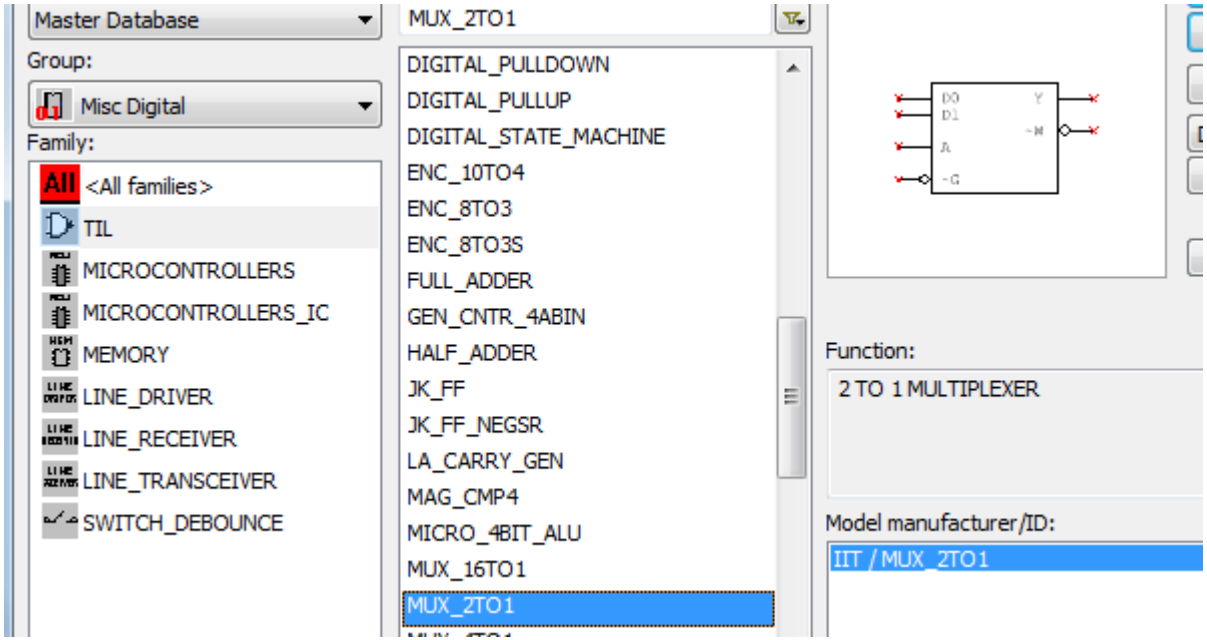
| Wejścia | | | | Wyjście | |
|---|---|---|---|---|---|
| Wejścia adresowe | | | | | |
| D | C | B | A | F | |
| 0 | 0 | 0 | 0 | 0 | $D_0 = 0$ |
| 0 | 0 | 0 | 1 | 0 | |
| 0 | 0 | 1 | 0 | 0 | $D_1 = A$ |
| 0 | 0 | 1 | 1 | 1 | |
| 0 | 1 | 0 | 0 | 1 | $D_2 = \bar{A}$ |
| 0 | 1 | 0 | 1 | 0 | |
| 0 | 1 | 1 | 0 | 0 | $D_3 = A$ |
| 0 | 1 | 1 | 1 | 1 | |
| 1 | 0 | 0 | 0 | 1 | $D_4 = \bar{A}$ |
| 1 | 0 | 0 | 1 | 0 | |
| 1 | 0 | 1 | 0 | 0 | $D_5 = A$ |
| 1 | 0 | 1 | 1 | 1 | |
| 1 | 1 | 0 | 0 | 0 | $D_6 = A$ |
| 1 | 1 | 0 | 1 | 1 | |
| 1 | 1 | 1 | 0 | 1 | $D_7 = 1$ |
| 1 | 1 | 1 | 1 | 1 | |

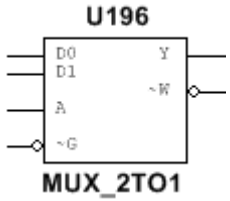


W przykładzie powyżej zrealizowano na multiplekserze funkcję F, która ma cztery wejścia: A, B, C, D. Trzy najstarsze wejścia funkcji (D, C, B) podano bezpośrednio na wejścia adresowe multipleksera. Następnie porównywano najmłodsze wejście funkcji A z wyjściem funkcji F. Porównywano po dwa, kolejne wiersze (odstęp między wierszami przedstawiony na rysunku).
Możliwe przypadki (dotyczące danej pary):
- niezależnie od stanu A, funkcja F daje na wyjściu 0 -> linię danych D (wejście danych multipleksera) łączymy z 0;
- niezależnie od stanu A, funkcja F daje na wyjściu 1 -> linię danych D łączymy z 1;
- funkcja F daje na wyjściu dokładnie taką wartość jaką ma sygnał wejściowy A -> linię danych D łączymy z sygnałem A;
- funkcja F daje na wyjściu wartość przeciwną do wartości przyjmowanej przez sygnał A -> linię danych D łączymy z zanegowanym sygnałem A.

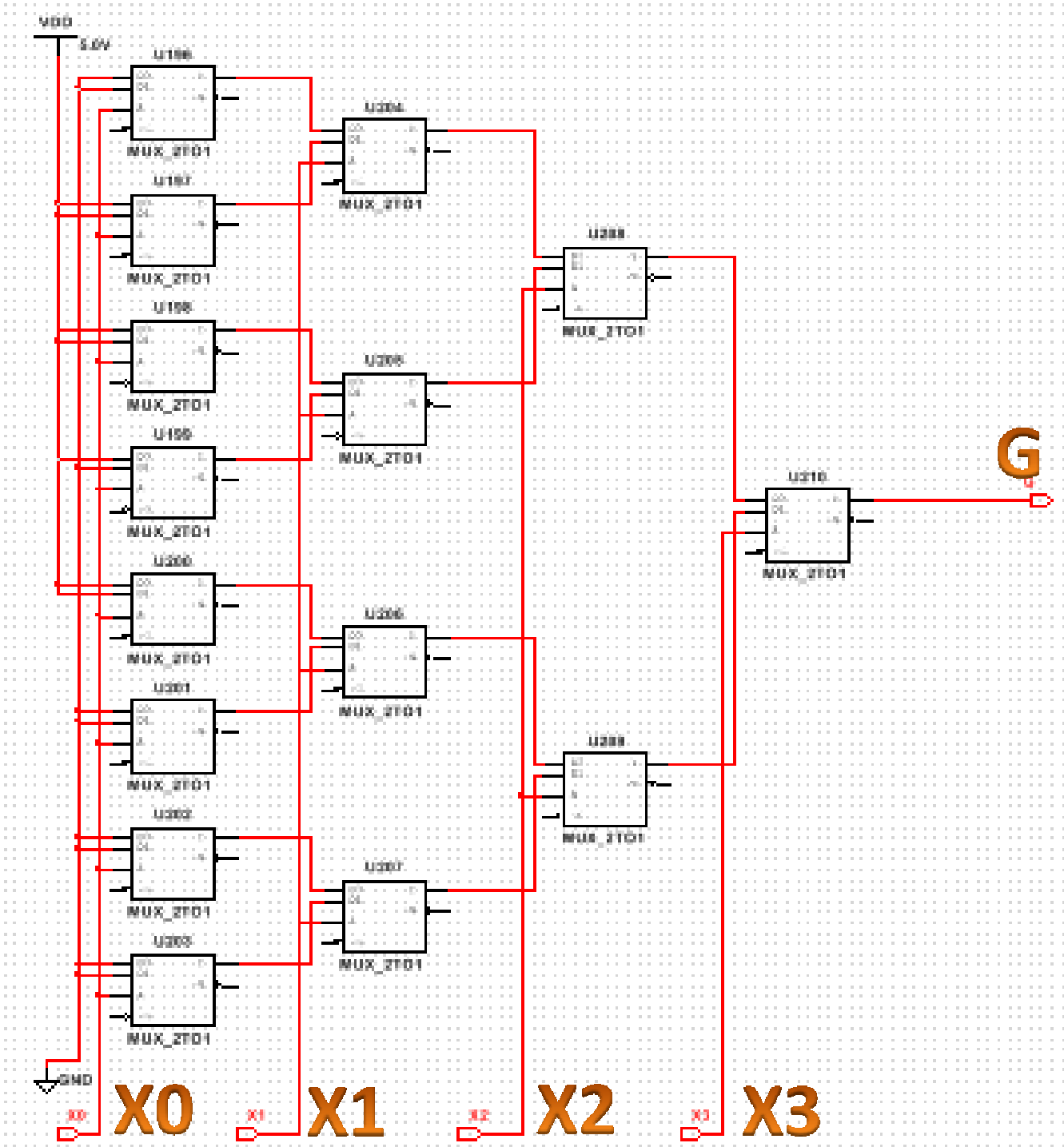
**Na podstawie przykładu zbudować układ dla segmentu „F". Oprócz multipleksera 8 na 1 należy wykorzystać inwerter „NOT" z tej samej biblioteki.**

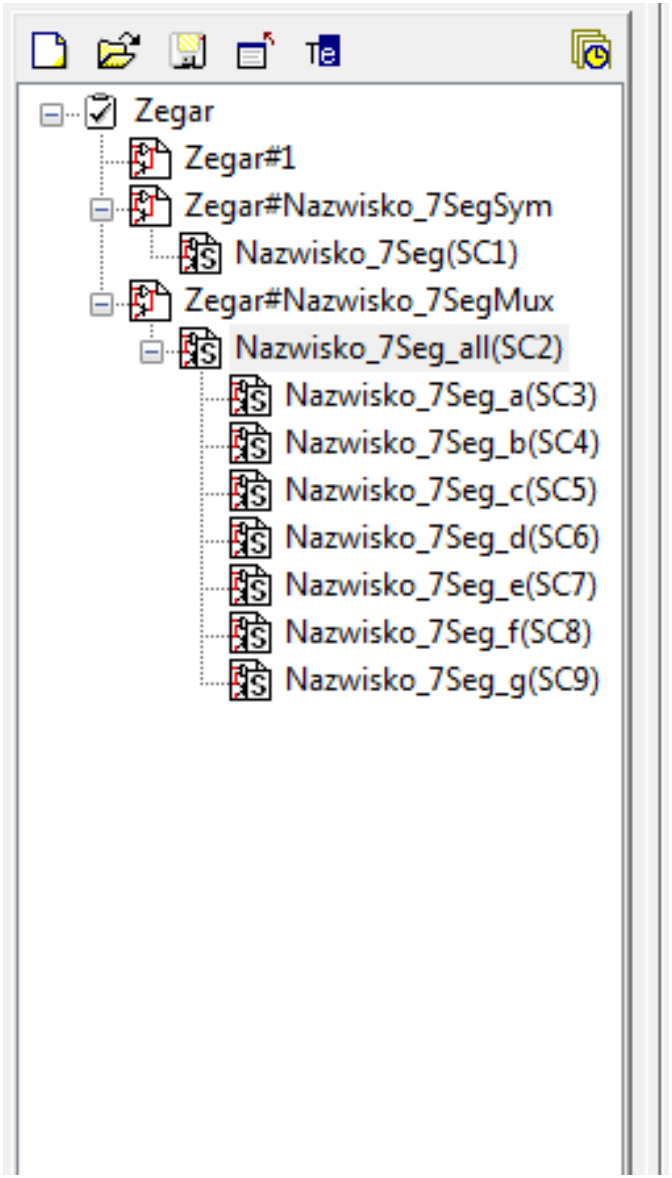
**7.** Segment „g” zbudować wykorzystując multipleksery 2 na 1.

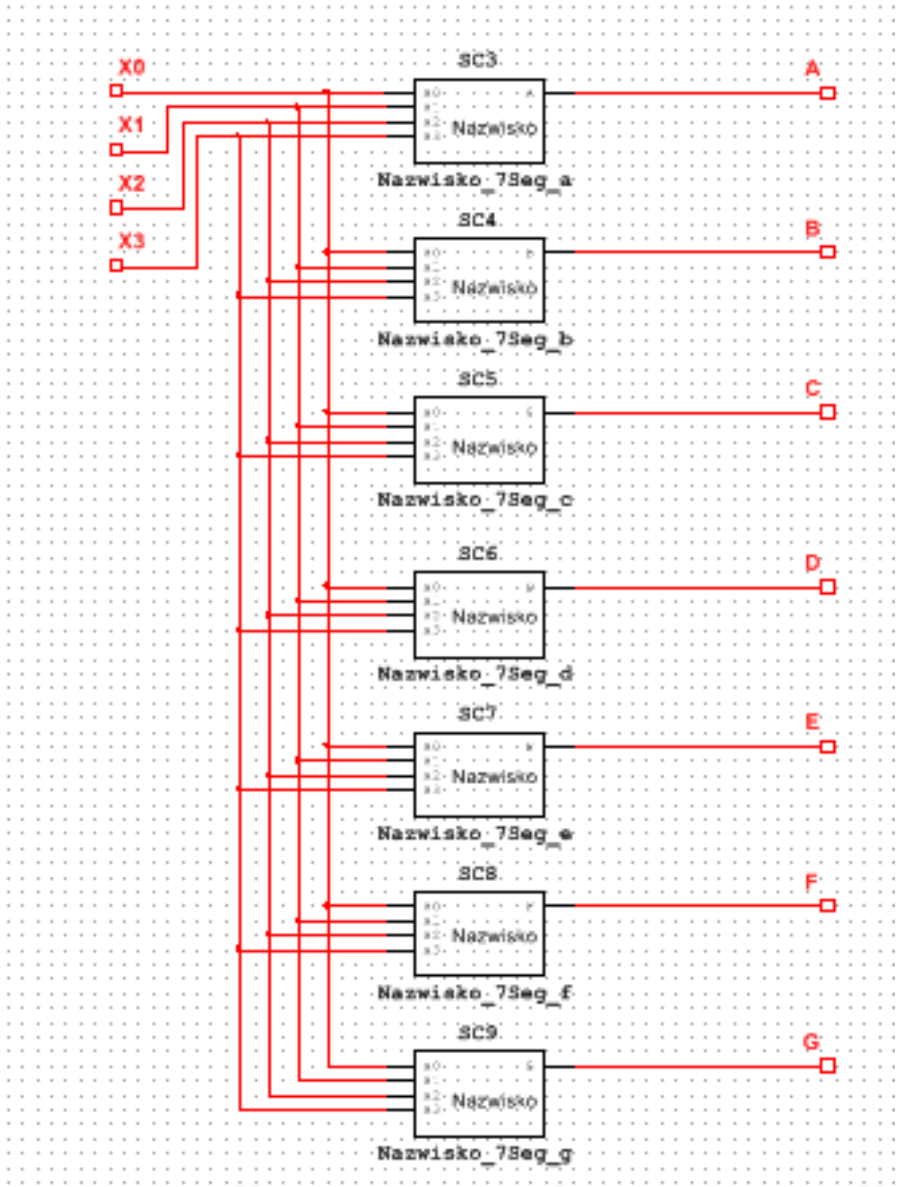


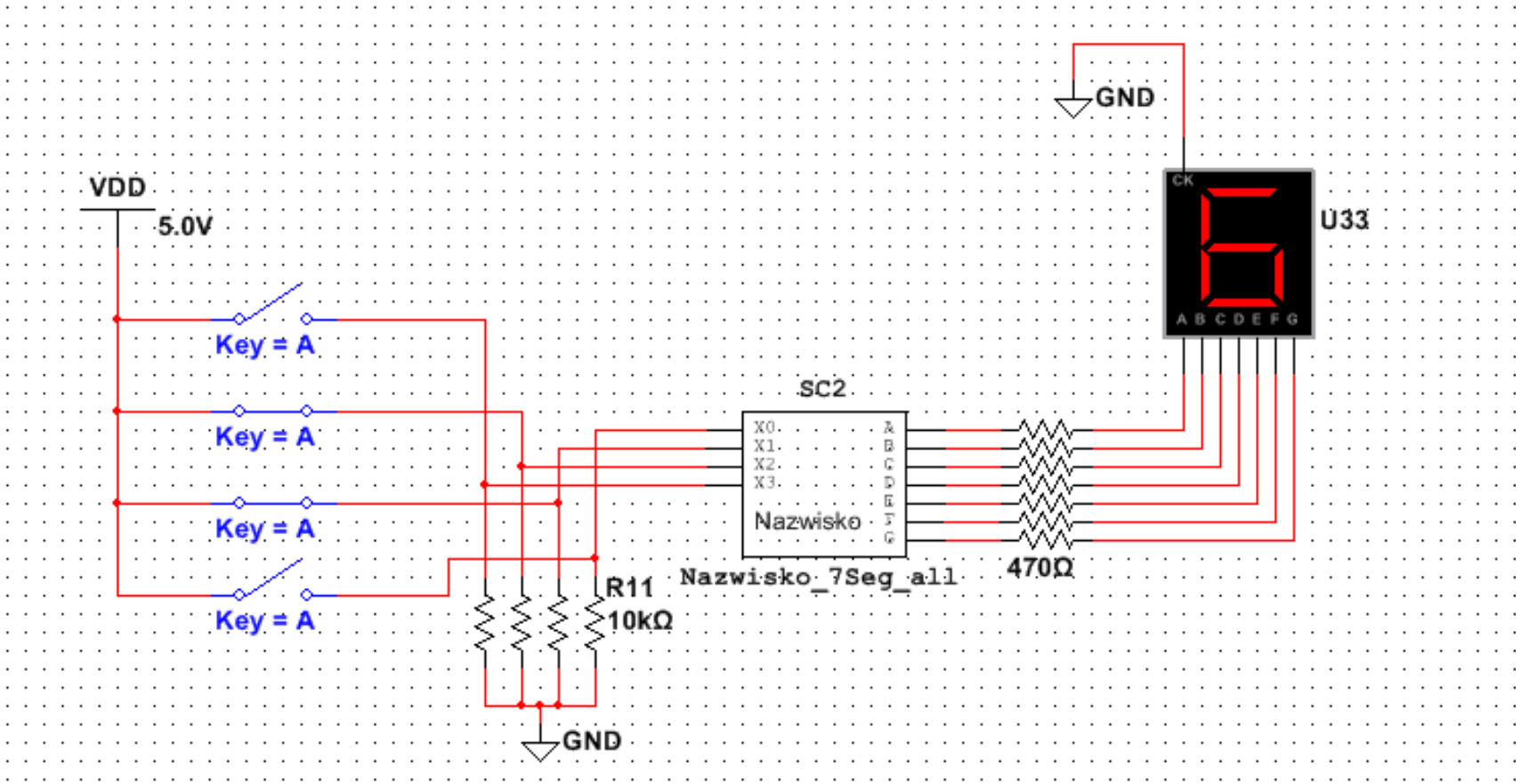

Aby zrealizować potrzebną funkcję logiczną (16 wejść i 1 wyjście) należy łączyć multipleksery w pary -> podłączając wyjścia dwóch multiplekserów do wejścia kolejnego.
Rysunek na kolejnym slajdzie.


VDD
5.0V
U199
MUX_2TO1
U197
MUX_2TO1
U198
MUX_2TO1
U199
MUX_2TO1
U200
MUX_2TO1
U201
MUX_2TO1
U202
MUX_2TO1
U203
MUX_2TO1
U204
MUX_2TO1
U205
MUX_2TO1
U206
MUX_2TO1
U207
MUX_2TO1
U208
MUX_2TO1
U209
MUX_2TO1
U210
MUX_2TO1
G
GND
X0
X1
X2
X3
X0
X1
X2
X3




**7.** Na schemacie „Nazwisko_7Seg_all” dodać wejścia i wyjścia „Hierarchical connector”.




**8.** Końcowy schemat „Nazwisko_7SegMux”.
Zweryfikować jego działanie dla wszystkich cyfr.