Pracownia elektryczna i elektroniczna

UKŁADY CYFROWE

Instrukcja nr 12

Opracował: Szczepan Marusiński

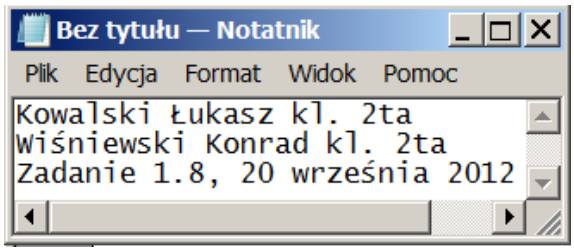
Wstęp

1. Zajęcia w Pracowni elektrycznej i elektronicznej z Układów cyfrowych realizowane są przy użyciu programu symulacyjnego **Multimedia Logic i Didital Works** w oparciu o dostarczone instrukcje.

2. Ocenianie pracy uczniów jest dokonywane przez zaliczenie pisemne lub ustne po przedstawieniu plików graficznych sporządzonych w następujący sposób.

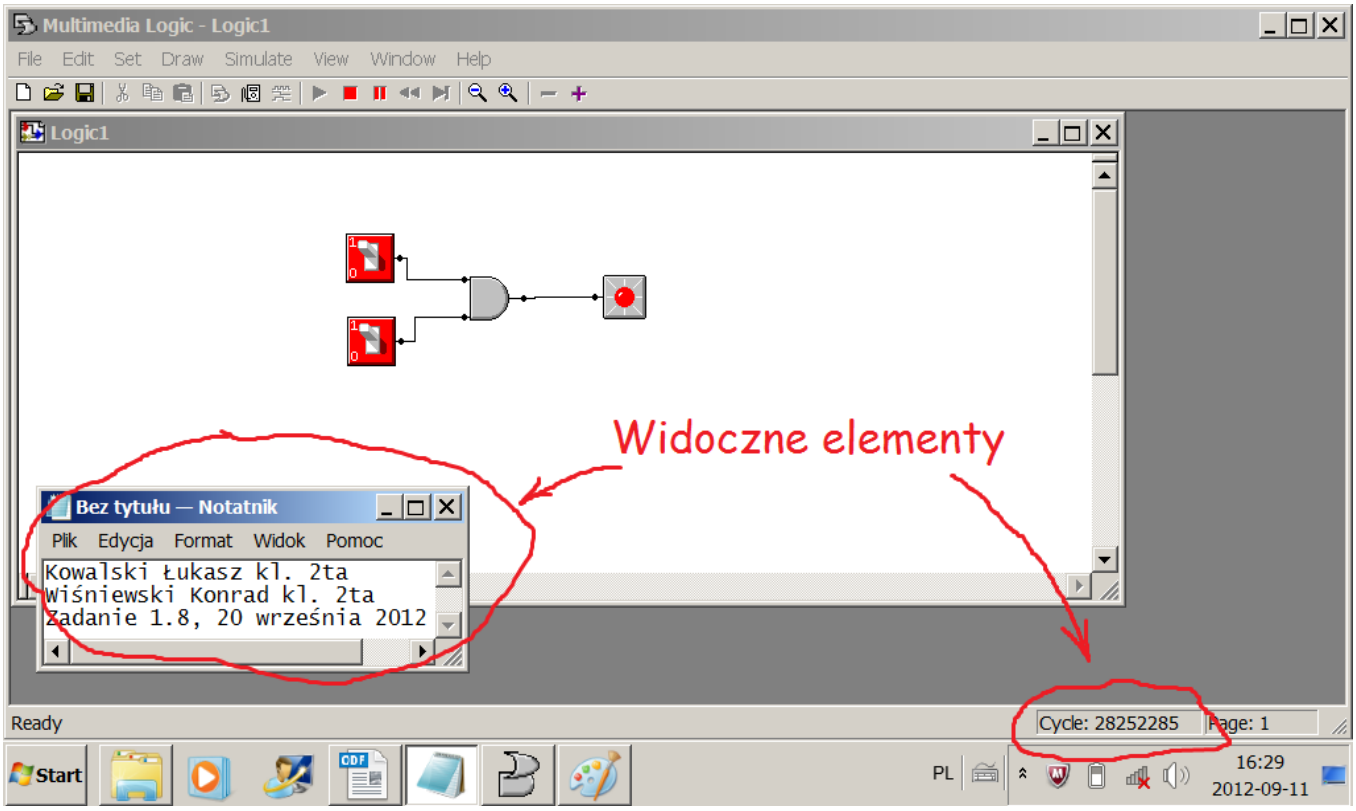
a) Wykonujemy poszczególne zadania i włączamy symulację ich działania.

b) Otwieramy nowy plik tekstowy, w którym umiszczamy w kolejnych linijkach następujące informacje: Nazwisko imię ucznia klasa, zadanie nr data



Plik ten powinien wyglądać jak na obrazku wyżej, czyli okienko powinno być małe lecz zawierać cały czytelny tekst.

c) W czasie symulacji układu w programie **Multimedia Logic** należy uaktywnić okienko pliku tekstowego i umieścić na tle okna w lewym dolnym rogu ekranu, jak na rysunku niżej.



d) Robimy *screen* obrazu naciskając klawisz **PrtSc** na klawiaturze komputera i wklejamy ten obraz do programu **Paint** lub **Open Office Writer.**

e) Zapisujemy ten obraz pod zawą **Zadanie nr**, np. *Zadanie 1.18*, w formacie *.jpg, w folderze na pulpicie komputera o nazwie **3ta Imię Nazwisko Pracownia elektryczna.**

f) Jeśli zdarzą się w klasie pliki z taką samą liczbą w okienku *Cycle:*, nie będą oceniane.

Wykaz najważniejszych praw logiki do wykorzystania na ćwiczeniach labratoryjnych

1) $a + b = b + a$
2) $a(b + c) = ab + ac$
3) $(a + b) + c = a + (b + c)$
4) $a + 0 = a$
5) $a + 1 = 1$
6) $a + \sim a = 1$
7) $\sim\sim a = a$
8) $ab = ba$
9) $a + bc = (a + b)(a + c)$
10) $(ab)c = a(bc)$
11) $a1 = a$
12) $a0 = 0$
13) $a\sim a = 0$
14) $a + ac = a$ $\quad$ $a(a + b) = a$
15) $a + \sim ab = a + b$ $\quad$ $a(\sim a + b) = ab$
16) $a + a = a$ $\quad$ $aa = a$
17) $\sim(a + b) = \sim a\sim b$ $\quad$ $\sim(ab) = \sim a + \sim b$ $\quad$ prawa de Morgana
18) $a \downarrow b = \sim a\sim b = \sim(a + b)$ $\quad$ funkcja Pirce'a (NOR)
19) $a \mid b = \sim a + \sim b = \sim(ab)$ $\quad$ funkcja Sheffera (NAND)
20) $a \oplus b = a\sim b + \sim ab$ $\quad$ suma modulo 2 (1 gdy argumenty są różne) (Ex-OR)
21) $a \otimes b = ab + \sim a\sim b = \sim (a \oplus b)$ $\quad$ równoważność (1 gdy argumenty są jednakowe) (Ex-Nor)

W powyższych wzorach znak ~ oznacza negację tylko najbliższego argumentu. Gdy znak negacji stoi przed nawiasem czytamy go tak, jak by negacja była umieszczona nad całym wyrażeniem w nawiasie, stąd:

$\sim(a + b)$ oznacza $\overline{a + b}$
$\sim ab$ oznacza $\bar{a}\,\underline{b}$
$\sim a\sim b$ oznacza $\overline{\bar{a}\,\bar{b}}$

Wiadomości wstępne

Pamięci służą do przechowywania informacji zakodowanej w postaci dwójkowej. Przy konstruowaniu pamięci są wykorzystywane różne zjawiska fizyczne. Podstawowe znaczenie ma zjawisko magnesowania materiałów ferromagnetycznych oraz niektóre zjawiska zachodzące w półprzewodnikach.

Pamięci półprzewodnikowe wytwarzane są jako układy scalone, głównie o dużym stopniu scalenia. Rozmaite typy pamięci różnią się sposobem zapisywania i odczytywania informacji, sposobem jej pamiętania, technologią wytwarzania itp.

Pamięci dzieli się na: pamięci zapisywane (typu zapis-odczyt), których zapisywanie jest równie łatwe jak odczytywanie, oraz pamięci stałe (ROM), których zapisywanie jest znacznie trudniejsze niż odczytywanie. Pamięci ROM w układach cyfrowych służą wyłącznie do przechowywania zapisanej wcześniej informacji.

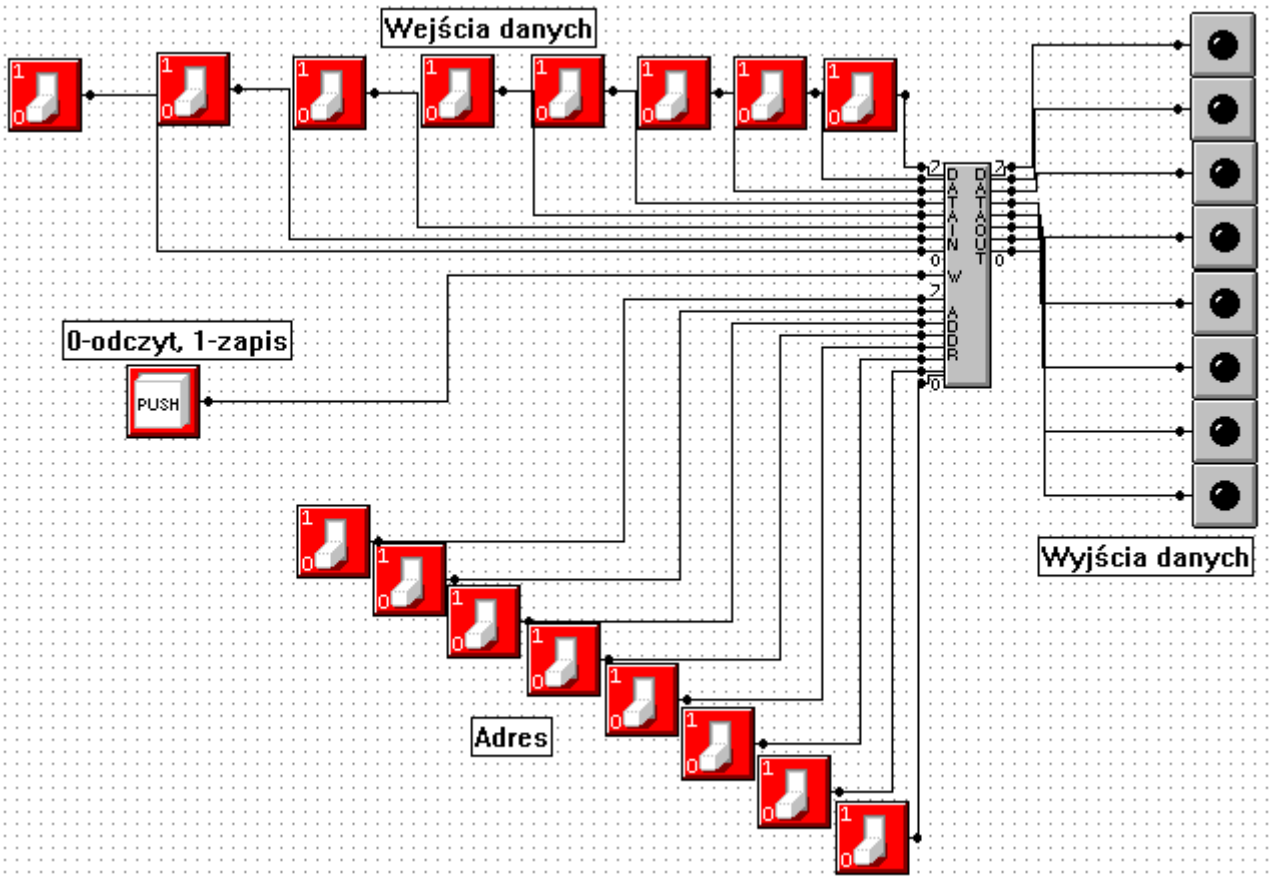
Pamięć składa się z komórek pamięci, w których są przechowywane poszczególne bity zapamiętanej informacji, oraz z układów pomocniczych. Komórki pamięci są rozróżniane za pomoca adresów.

Rolę komórek pamięci w pamięciach zapisywalnych pełnią bądź przerzutniki bistabilne, bądź układy przechowujące informację w postaci ładunku elektrycznego (w pojemnościach tranzystora unipolarnego). W przerzutnikach informacja może być przechowywana długo pod warunkiem dostarczania napięcia zasilającego, natomiast w pojemności tranzystora unipolarnego

zaledwie ułamek sekundy. Funkcję pamiętania wspomaga się więc częstym odczytywaniem i zapisywaniem informacji, tzw. odświeżaniem, np. z częstotliwością 500 Hz. Pamięci wymagające odświeżania nazywamy dynamicznymi, a te które tego nie wymagają statycznymi.

Zadanie 12.1

W programie Multimedia Logic zbuduj układ podobnie jak na rysunku poniżej. Przy użyciu wejść adresowych, wejść danych i wejścia zapis/odczyt wprowadź do pamięci słowa: S1 = 01010101; S2 = 11110000; S3 = 11001100; S4 = 00111100; S5 = Nr_komp; odpowiednio pod adresy: A1 = 00000001; A2 = 00000010; A3 = 00000011; A4 = 00000100; A5 = 00000101. Po zapisaniu wszystkich słów w pamięci odczytaj zawartości zapisanych komórek tworząc pliki graficzne. (Format pamięci ustaw jako Binary).

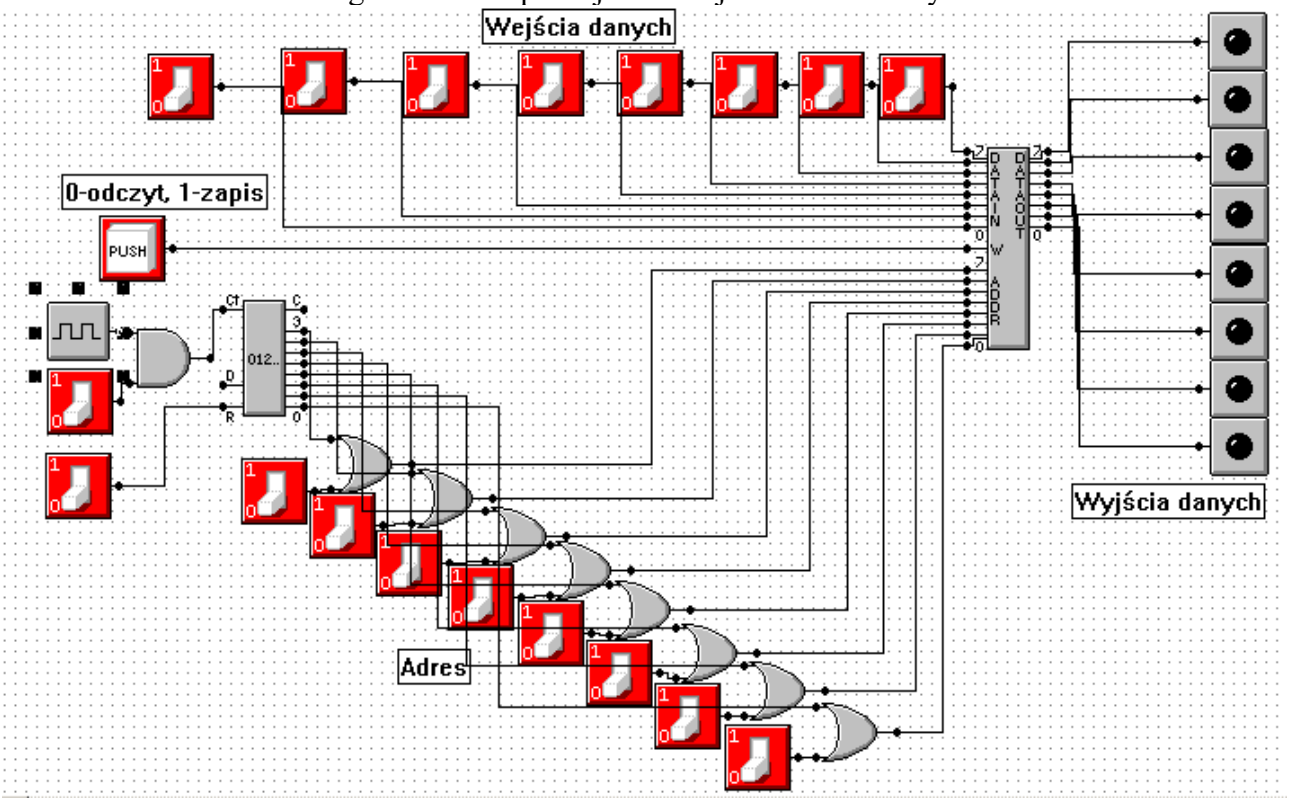


Zadanie 12.2

W programie Multimedia Logic zbuduj układ podobnie jak w Zadaniu 12.1 używając pamięci 16 bitowej (opcja formatu). Przy użyciu wejść adresowych, wejść danych i wejścia zapis/odczyt wprowadź do pamięci słowa: S1 = 00000001; S2 = 00000010; S3 = 00000100; S4 = 00001000; S5 = Nr_komp; odpowiednio pod adresy: A1 = 0000000000000001; A2 = 0000000000000010; A3 = 0000000000000011; A4 = 0000000000000100; A5 = 0000000000000101. Po zapisaniu wszystkich słów w pamięci odczytaj zawartości zapisanych komórek tworząc pliki graficzne. (Format pamięci ustaw jako Binary).

Zadanie 12.3

W programie Multimedia Logic zbuduj układ jak na rysunku poniżej. Opcję formatu pamięci ustaw jako Binary. Czasy załączenia i wyłączenia oscylatora ustaw na 1000000. Do pamięci wprowadź słowa o wartościach podaneych w poniższej tabeli. Następnie ręczne zadajniki adresu ustaw w pozycji 0, załącz licznik i zresetuj go. Poproś nauczyciela o sprawdzenie działania układu w celu zaliczenia tego zadania. Spróbuj ustawić jakieś inne efekty świetlne.

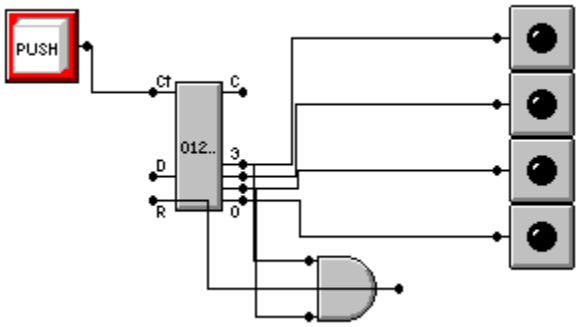


Lp	Adres binarny	Dane/słowa
0	00000000	11110000
1	00000001	00001111
2	00000010	11110000
3	00000011	00001111
4	00000100	11110000
5	00000101	00001111
6	00000110	10000001
7	00000111	01000010
8	00001000	00100100
9	00001001	00011000
10	00001010	00100100
11	00001011	01000010
12	00001100	10000001
13	00001101	10000000
14	00001110	01000000
15	00001111	00100000
16	00010000	00010000
17	00010001	00001000
18	00010010	00000100
19	00010011	00000010
20	00010100	00000001
21	00010101	10010010
22	00010110	01001001
23	00010111	00100100
24	00011000	10010010
25	00011001	01001001
26	00011010	00100100
27	00011011	10010010
28	00011100	01001001
29	00011101	00100100
30	00011110	10000000
31	00011111	11000000
32	00100000	01000000
33	00100001	11000000
34	00100010	00100000
35	00100011	01100000

36	00100100	10100000
37	00100101	00100000
38	00100110	10100000
39	00100111	01100000
40	00101000	00100000
41	00101001	00010000
42	00101010	00110000
43	00101011	01010000
44	00101100	10010000
45	00101101	00010000
46	00101110	10010000
47	00101111	01010000
48	00110000	00110000
49	00110001	00010000
50	00110010	00001000
51	00110011	00011000
52	00110100	00101000
53	00110101	01001000
54	00110110	10001000
55	00110111	00001000
56	00111000	10001000
57	00111001	01001000
58	00111010	00101000
59	00111011	00011000
60	00111100	00001000
61	00111101	00000100
62	00111110	00000010
63	00111111	00000001

Zadanie 12.4

W programie Multimedia Logic zbuduj licznik mod 10 jak na poniższym rysunku. Wyjścia tego licznika podłącz do wejść adresowych pamięci. Wyjścia pamięci podłącz do wyświetlacza siedmiosegmentowego. Zaprogramuj pamięć tak, by na wyświetlaczu pojawiały się kolejno wszystkie cyfry. Stwórz dziesięć plików graficznych dla każdej wyświetlonej cyfry.



Zadanie 12.5

Analogicznie jak w zadaniu 12.4 zbuduj układ wyświetlający po kolei liczby od 0 do 99. Stwórz pliki graficzne z wyświetlonymi następującymi liczbami: Nr_komputera; Nr_komputera x 5; Twój(wasz)_Nr_w_dzienniku.

Po wykonaniu wszystkich ćwiczeń upewnij się, czy masz komplet plików graficznych.