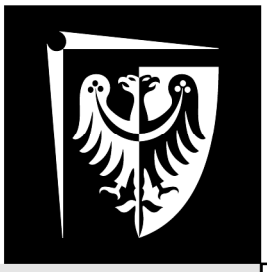


# Politechnika Wrocławska

## Podstawy Techniki Cyfrowej i Mikroprocesorowej

Laboratorium

Makiety dydaktyczne

Wydział Elektroniki Mikrosystemów i Fotoniki

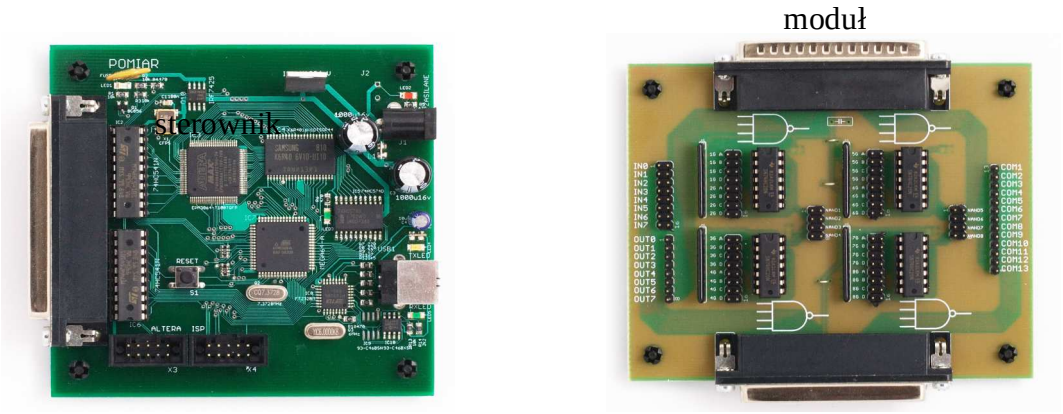
# 1. Opis elementów makiety

## 1.1. Wstęp

W trakcie zajęć laboratoryjnych z przedmiotu Podstawy Techniki Cyfrowej i Mikroprocesorowej studenci wykonują szereg zadań. Jednym z nich jest zestawianie, uruchamianie i testowanie prostych układów cyfrowych kombinacyjnych i sekwencyjnych przy użyciu makiet dydaktycznych. Makiety te zostały specjalnie opracowane na potrzeby kursu prowadzonego na Wydziale Elektroniki Mikrosystemów i Fotoniki.

Makieta składa się ze sterownika i dołączonych do niego modułów. Sterownik połączony jest z komputerem osobistym i obsługiwany przez dedykowany program. Sterownik i moduły wykonane są w postaci płytek drukowanych. Ich połączenia ze sobą dokonuje się bezpośrednio złączami D-SUB. Złącza te zapewniają ciągłość linii zasilania oraz magistral. Magistrale są wspólne dla wszystkich modułów użytych do budowy makiety. Zestawienie płytek zależy od tematu ćwiczenia.

Sercem makiety jest sterownik, do niego dołącza się szereg modułów z elementami składowymi budowanych układów cyfrowych. Wygląd sterownika i przykładowego modułu przedstawiono na rysunku 1:

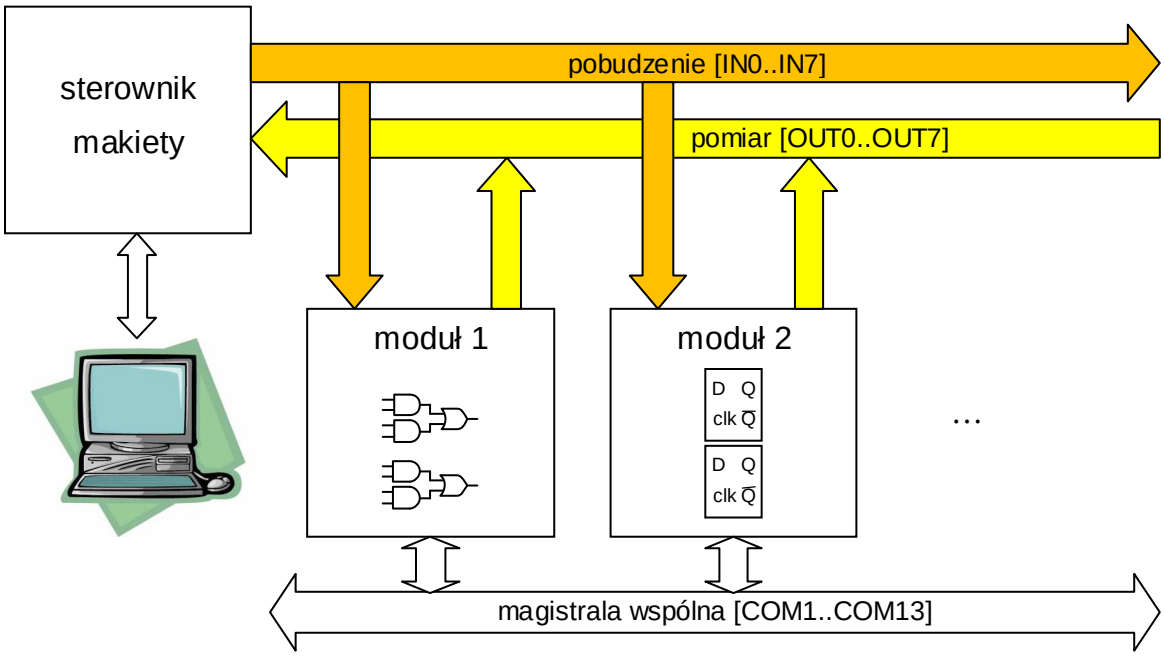



Rysunek 1. Płytka drukowana sterownika i przykładowego modułu.

Sterownik odpowiedzialny jest za zasilanie, generowanie sygnałów cyfrowych podawanych na wejścia badanego układu oraz pełni rolę prostego analizatora stanów logicznych, czyli zapewnia pomiar i akwizycję zmian stanów logicznych w czasie. Ogólnie, użytkowanie makiety sprowadza się do wykonania następujących czynności:

- zbudowanie układu cyfrowego będącego przedmiotem ćwiczenia zgodnie z uprzednio wykonanym projektem
- ustalenie przy pomocy oprogramowania pobudzenia (czyli odpowiedniej sekwencji słów binarnych, które z zadaną szybkością sterownik będzie podawał na wejścia układu cyfrowego) oraz parametrów analizatora stanów logicznych
- uruchomienie makiety, co powoduje podanie na nią zasilania, wygenerowanie sygnału pobudzającego, zmierzenie i zapamiętanie w wewnętrznej pamięci sterownika stanów wyjść układu cyfrowego
- przesłanie danych do komputera i zaprezentowanie ich na ekranie

Czynności dotyczące obsługi makiety są zautomatyzowane i ich wykonanie polega na wybieraniu odpowiednich poleceń w programie do obsługi makiety. Schemat blokowy stanowiska dydaktycznego zestawionego w ten sposób przedstawiono na rysunku 2.




Rysunek 2. Schemat blokowy makiety

Sterownik wysyła generowane przez siebie sygnały pobudzające na 8-bitową magistralę IN0..IN7. Linie tej magistrali są wyprowadzone na każdym z modułów na podwójne złącze szpilkowe oznaczone odpowiednimi symbolami. Złącza te można wykorzystać do połączenia linii magistrali z wejściami układów cyfrowych na modułach.

Analizator stanów logicznych w sterowniku bada stany logiczne na liniach magistrali OUT0..OUT7. Linie tej magistrali również są wyprowadzone na każdym z modułów na pojedynczych złączach szpilkowych. Aby było możliwe analizowanie zmian stanów logicznych w węźle układu cyfrowego należy połączyć go z jedną z linii OUTn. **Uwaga!** Pamiętać należy o tym, że nie wolno ze sobą łączyć wyjść układów cyfrowych, zatem **do jednej linii magistrali OUTn można podłączyć tylko jeden węzeł układu**.

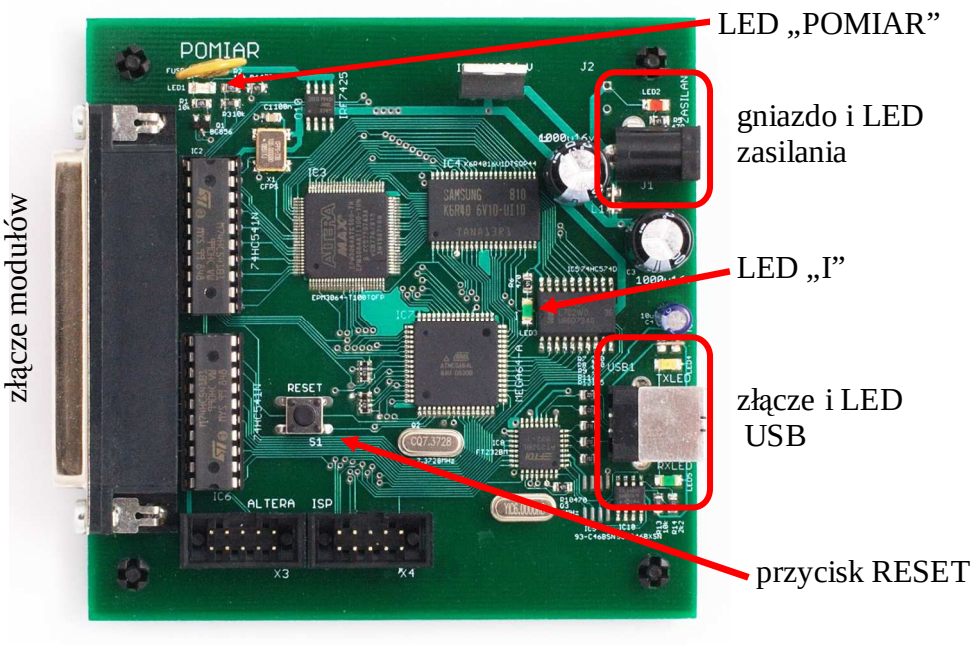
Dla wygody wykonywania połączeń pomiędzy modułami do dyspozycji jest również magistrala COM1..COM13, która nie jest dołączona do sterownika. Może być ona wykorzystywana do dokonywania połączeń pomiędzy modułami. Linie tej magistrali wyprowadzone są również w postaci złącz szpilkowych, oznaczonych COMn. Również pamiętamy o zasadzie, że jeśli do jednej linii magistrali COMn dołączone zostanie wyjście któregoś z układów na jednym z modułów, to nie można do tej linii dołączać innego wyjścia.

## 1.2. Sterownik

Serce makiety – sterownik – współpracuje z komputerem osobistym, na którym uruchomiony jest dedykowany program do obsługi makiety. Do sterownika należy dołączyć zasilanie (zasilacz wtyczkowy 5V 3A) oraz połączyć go przewodem USB z komputerem osobistym.

Sterownik jest systemem mikroprocesorowym opartym o mikrokontroler ATMEGA64L. Sterownik wyposażony jest w 512 kB pamięci SRAM, układ CPLD zarządzający akwizycją sygnałów logicznych, pracujący z zegarem 100 MHz oraz układ FT232 do obsługi interfejsu USB. Budowa sterownika opisana jest dokładnie

w załącznikach. Na sterowniku znajduje się również liniowy stabilizator napięcia dostarczający napięcia równego 3,3 V służącego do zasilania samego sterownika oraz dołączonych modułów. Widok sterownika przedstawiono na rysunku 3.



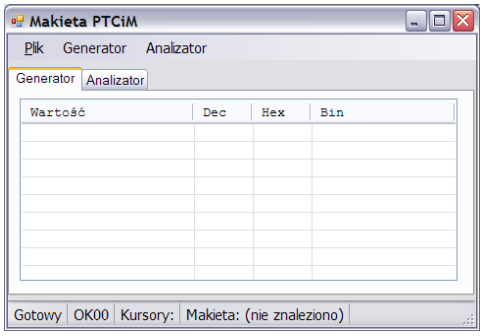

Rysunek 3. Widok płytki sterownika.

Sterownik wyposażono w pięć diod LED sygnalizujących stan, w jakim sterownik się znajduje:

- ZASILANIE – czerwona dioda sygnalizująca podłączenie zasilacza wtyczkowego
- TXLED, RXLED – diody żółta i zielona, sygnalizujące wysyłanie i odbieranie danych przez złącze USB; jeżeli świecą się obie diody światłem stałym oznacza to dołączenie USB do komputera ale brak napięcia zasilającego z zasilacza wtyczkowego
- POMIAR – niebieska dioda sygnalizująca podanie zasilania do modułów, które są dołączone do sterownika; napięcie to jest podawane wyłącznie na czas testowania prawidłowości działania układu cyfrowego
- I – dioda sygnalizująca stan, w jakim znajduje się sterownik:
    - światło ciągłe – sterownik jest gotowy na przyjęcie polecenia z komputera
    - brak świecenia – sterownik realizuje bieżące polecenie; jeśli ten stan utrzymuje się przez dłuższy czas oznacza to zawieszenie się sterownika, należy nacisnąć przycisk RESET
    - światło przerywane – błąd sterownika, zadziałanie układu zabezpieczającego przed przetężeniem bądź inna sytuacja awaryjna; należy wtedy wyłączyć zasilanie sterownika

## 1.3. Oprogramowanie

Ze sterownikiem ściśle współpracuje program komputerowy **Makieta PTCiM**. Umożliwia on programowanie sekwencji słów w generatorze, uruchamianie makiety i zapewnia wygodny, czytelny sposób na przedstawienie wyników działania makiety w postaci wykresu czasowego. Główne okno programu zaraz po jego uruchomieniu przedstawiono na rysunku 4.




Rysunek 4. Główne okno programu Makieta PTCiM.

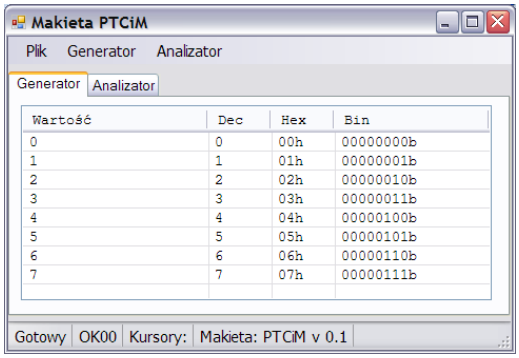
W oknie wyświetlane są pasek menu, dwie zakładki Generator i Analizator, w których widoczne będą program dla generatora i wyniki pracy analizatora stanów logicznych, oraz pasek stanu podzielony na cztery części: stan programu, symbol stanu makiety (istotne w przypadku nieprawidłowej pracy sterownika), położenie kursorów na ekranie analizatora stanów logicznych i wersja sterownika makiety (pojawia się dopiero po nawiązaniu połączenia z makietą)


Rysunek 5. Znajdowanie makiety.

Pierwszą czynnością po uruchomieniu programu, zasileniu sterownika i dołączeniu go do komputera przewodem USB jest nawiązanie komunikacji z makietą *(Plik→Znajdź makietę)*. Najprościej nacisnąć przycisk Szukaj (rysunek 5), program będzie próbował odnaleźć makietę na kolejno sprawdzanych portach COM. Po zakończeniu poszukiwania pojawi się komunikat informujący o powodzeniu lub niepowodzeniu wyszukania.

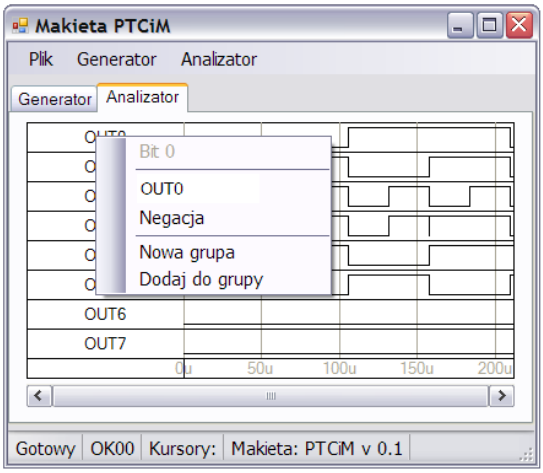
Po połączeniu się ze sterownikiem można ustalić w nim sekwencję słów w generatorze stanów logicznych. Dokonuje się tego poprzez podanie kolejno wartości 8-bitowych słów, które w trakcie pracy makiety będą podawane na magistralę IN0..IN7. Wartości tych słów można podawać w postaci liczb dziesiętnych (np. 17), szesnastkowych (5Eh) lub binarnych (1101001b). W przypadku wartości szesnastkowych lub binarnych konieczne jest dopisanie liter odpowiednio ‘h’ lub ‘b’ za podawaną liczbą.

Sekwencję słów generatora można ustalać bezpośrednio w programie korzystając z menu podręcznego na liście słów generatora lub przygotować w osobnym edytorze (np. w standardowym Notatniku) plik tekstowy, którego kolejne linie będą odpowiadały kolejnym słowom generatora, po czym taki plik można wczytać do programu *(Generator→Otwórz plik generatora)*. W oknie programu sekwencja słów jest wyświetlana we wszystkich trzech formach. Przygotowaną sekwencję słów generatora można zapisać do pliku korzystając z poleceń menu *Generator*. Zaprogramowanie generatora następuje po wybraniu polecenia *(Generator→Zaprogramuj)*.




Rysunek 6. Okno programu z przykładową sekwencją słów generatora stanów logicznych.

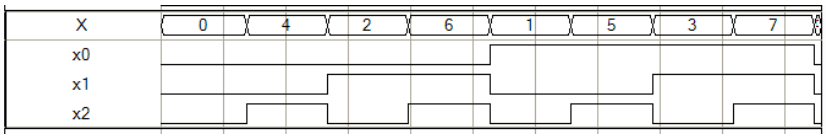
Po przygotowaniu sekwencji słów generatora można uruchomić makietę *(Analizator→Uruchom)*. Polecenie to powoduje uruchomienie makiety z rejestracją stanów logicznych na magistrali OUT0..OUT7, pobranie z makiety wyników rejestracji i wyświetlenie ich na ekranie w zakładce *Analizator* w postaci przebiegów czasowych.




Rysunek 7. Okno analizatora stanów logicznych z menu podręcznym.

Standardowo wyświetlonych jest 8 bitów, które zarejestrował analizator oznaczonych OUTn. Dla zwiększenia czytelności wykresów czasowych można zmodyfikować opisy bitów aby odpowiadały one znaczeniu prezentowanych sygnałów. Opis można modyfikować zmieniając go w menu podręcznym ze standardowego OUTn na żądany ciąg znaków. Dodatkowo, bit może być wyświetlany w postaci zanegowanej gdy zostanie zaznaczona opcja *Negacja* w menu podręcznym, o negacji informuje nakreślenie nad opisem bitu.

W przypadku, gdy grupa bitów ma jakieś znaczenie, np. są one kolejnymi bitami wielobitowego słowa kodowanego w kodzie NKB, można dla wygody połączyć je w grupę. Dla grupy bitów w oknie analizatora stanów logicznych podawana jest nie tylko wartość poszczególnych bitów ale również wartość słowa, które tworzą zgrupowane bity.




Rysunek 8. Grupa bitów w oknie analizatora stanów logicznych.

Aby wyświetlić zgrupowane bity należy utworzyć nową grupę *(menu podręczne→Nowa grupa)* po czym dodać do grupy bity w odpowiedniej kolejności *(menu podręczne bitu, który chcemy dodać→Dodaj do grupy)*. Bity są dodawane do ostatnio

utworzonej grupy

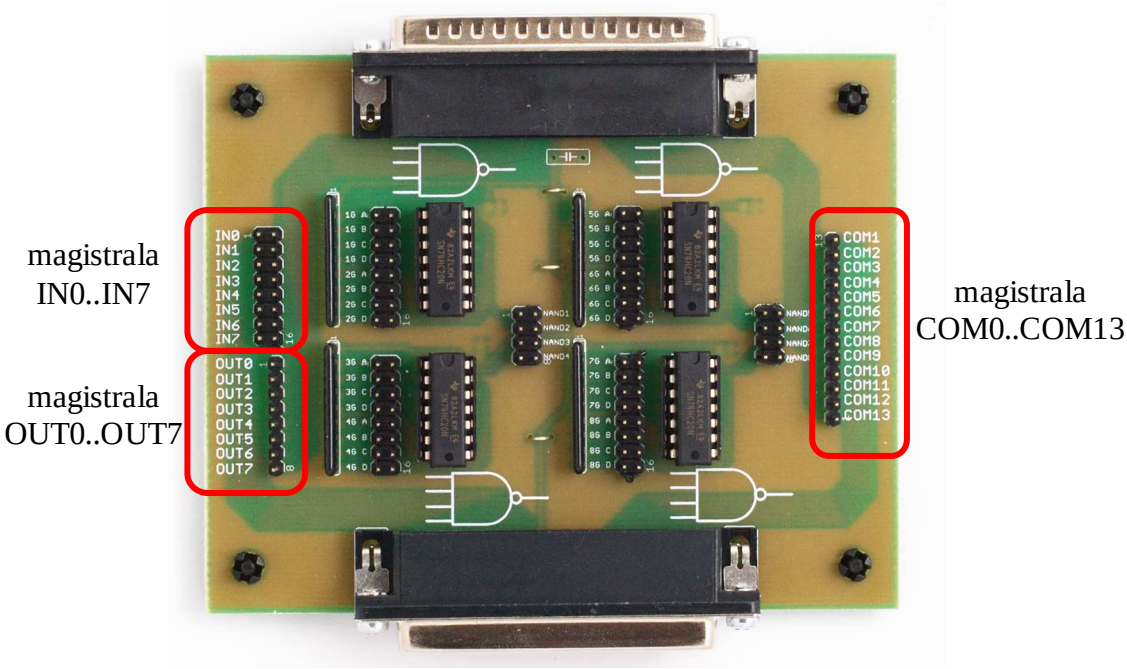
Podobnie jak w przypadku pojedynczych bitów, grupom można nadawać nazwy, dodatkowo można grupę wyświetlać w postaci rozwiniętej (jak na rysunku 8) lub zwiniętej (nie są wtedy wyświetlane stany bitów tworzących grupę) oraz można wybierać format (dziesiętny, szesnastkowy, binarny), w którym wyświetlane są wartości słów tworzonych przez zgrupowane bity. Czynności te można wykonać wybierając odpowiednie polecenie z menu podręcznego grupy. Przygotowany opis bitów można zapisać do pliku lub wczytać uprzednio stosowany opis bitów (polecenia w menu *Generator*)

W oknie analizatora stanów logicznych można zmieniać powiększenie przy użyciu kółka myszki. Pozwala to na dokładną obserwację opóźnień pomiędzy zmianami stanów logicznych na poszczególnych wejściach analizatora.

Dodatkowo do dyspozycji są kursory ustawiane poprzez kliknięcie prawym lub lewym przyciskiem na obszarze, w którym wyświetlone są przebiegi czasowe. Wartość odstępu czasowego pomiędzy pozycjami kursorów wyświetlona jest w pasku stanu aplikacji.

## 1.4. Moduły

Każdy z modułów jest wyposażony w dwa złącza D-SUB umożliwiające łączenie modułów jeden z drugim oraz identycznie rozmieszczone złącza szpilkowe magistral. Ich rozmieszczenie zostało pokazane na rysunku

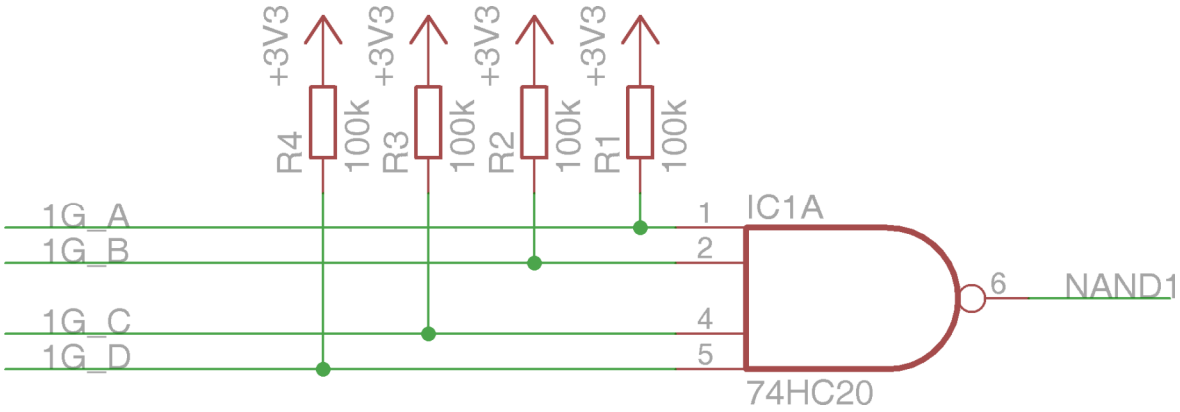



Rysunek 9. Widok modułu NAND i rozmieszczenie gniazd szpilkowych magistral.

Do konstrukcji modułów zastosowano układy rodziny 74HC – czyli układy wykonane w technologii CMOS, o układzie wyprowadzeń zgodnym z rodziną 74. Napięcie zasilania układów makiety wynosi 3,3V.

### 1.4.1. Moduł NAND

Na module NAND umieszczono cztery układy scalone 74HC20, w każdym z nich znajdują się dwie, czterowejściowe bramki NAND. Schemat połączeń dla pojedynczej bramki przedstawiono na poniższym rysunku:




Rysunek 10. Schemat połączeń na module NAND.

Na złączach szpilkowych zostały wyprowadzone wejścia i wyjścia bramek. Wejścia oznaczone są nG x , gdzie ‘n’ to numer bramki, ‘x’ to symbol wejścia (A, B, C, D). Wyjścia bramek oznaczono NANDn.

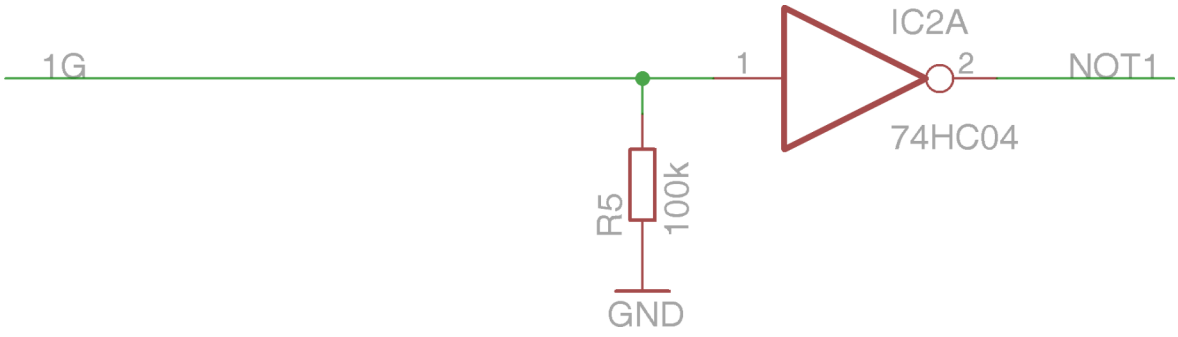
Niewykorzystanych wejść układów cyfrowych wykonanych w technologii CMOS nie powinno się pozostawiać „wiszących”, to znaczy nie podłączonych do niczego. Aby nie dopuścić do tego, wejścia bramek połączono rezystorami do zasilania. Wartości rezystorów są tak dobrane aby w przypadku, gdy do wejścia bramki dołączony zostanie sygnał cyfrowy, rezystor nie miał wpływu na stan logiczny, natomiast na niewykorzystanych wejściach zostanie wymuszony wysoki stan logiczny. W przypadku bramek NAND oznacza to, że na przykład w bramce, w które wykorzystane będą tylko wejścia A i C (czyli na B i D będzie zawsze ‘1’) stan wyjściowy Y będzie równy:

$$Y = \overline{A \cdot B \cdot C \cdot D} = \overline{A \cdot 1 \cdot C \cdot 1} = \overline{A \cdot C} \qquad (1)$$

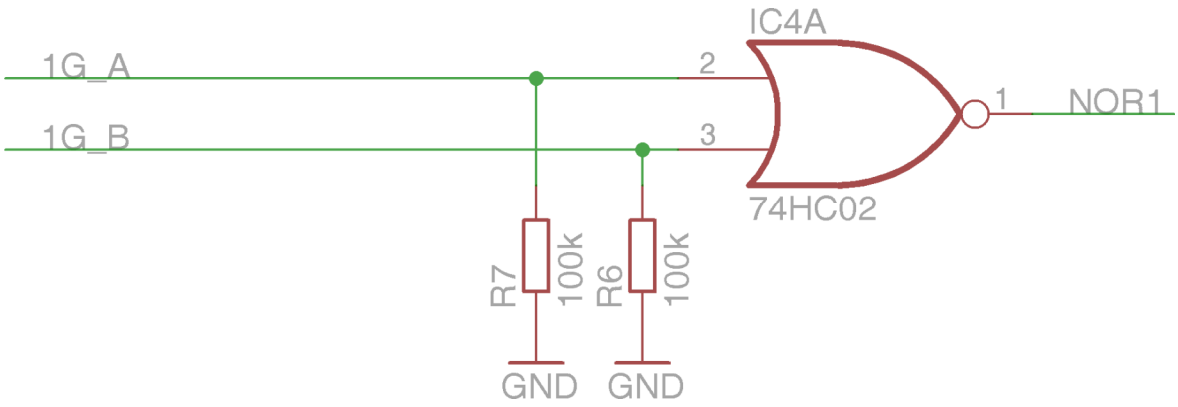
Jak widać, niewykorzystane wejścia nie będą wpływały na stan logiczny wyjścia. Bramki można więc używać jako dwu, trzy lub czterowejściowe bramki NAND lub, w przypadku wykorzystania tylko jednego wejścia, jako inwerter. Dodatkowo, jeżeli nie wykorzysta się żadnego z wejść, wtedy na wyjściu bramki otrzymamy stały, niski poziom logiczny.

### 1.4.2. Moduł bramek logicznych

Na module bramek logicznych znajdują się po jednym układzie 74HC04 (sześć negatorów) oraz po dwa układy 74HC02 (cztery dwuwejściowe bramki NOR). Schematy połączeń przedstawiono na rysunkach11 i12:




Rysunek 11: Schemat połączeń bramki NOT na module bramek logicznych.

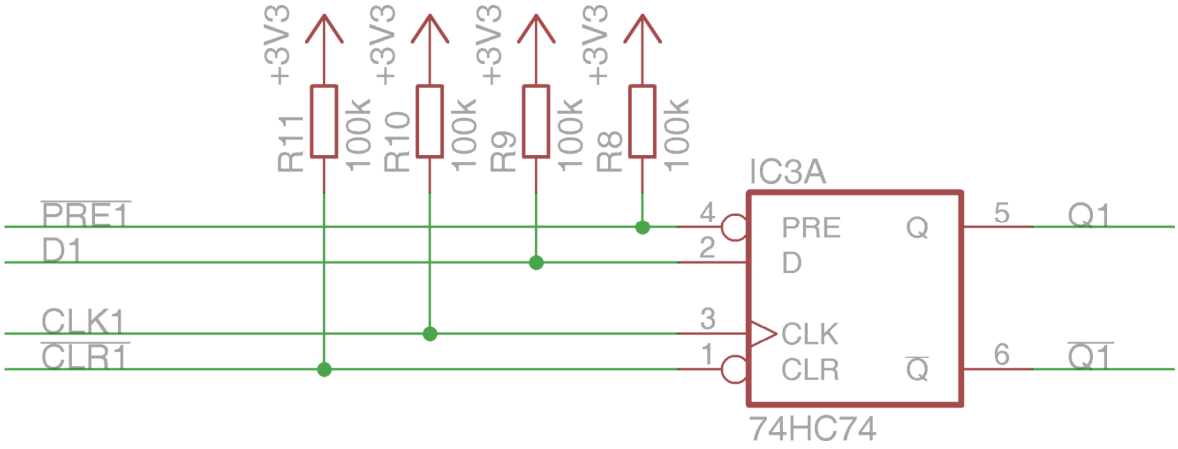



Rysunek 12: Schemat połączeń bramki NOR na module bramek logicznych.

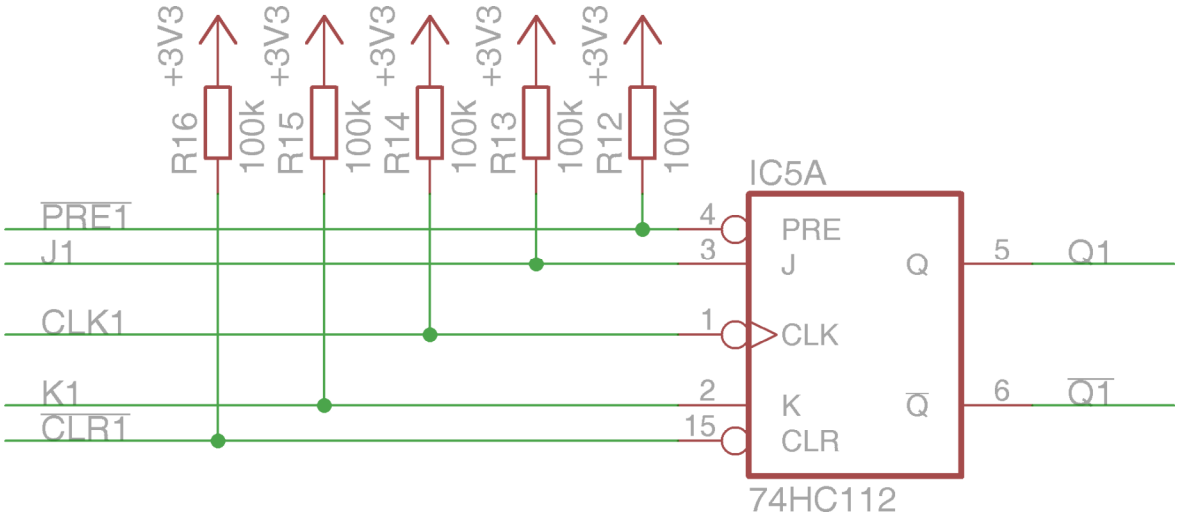
Oznaczenia wejść są podobne jak na module NAND. Obie bramki są wyposażone w rezystory wymuszające niski stan logiczny na niewykorzystanych wejściach. Jeżeli nie wykorzysta się żadnego z wejść wtedy na wyjściu bramki otrzyma się stały, wysoki poziom logiczny.

### 1.4.3. Moduł przerzutników synchronicznych

Na module przerzutników synchronicznych zamontowane są po dwa układy 74HC74 (podwójny przerzutnik typu D, taktowany dodatnim zboczem sygnału zegarowego) oraz dwa układy 74HC112 (podwójny przerzutnik JK taktowany ujemnym zboczem sygnału zegarowego). Oba rodzaje przerzutników posiadają asynchroniczne wejścia do ich wstępnego ustawiania i kasowania, pracujące w logice ujemnej (stanem aktywnym jest stan niski). Schemat połączeń na modułach przerzutników synchronicznych przedstawiono na rysunkach 13 i 14.




Rysunek 13. Schemat połączeń przerzutnika D na module przerzutników.




Rysunek 14. Schemat połączeń przerzutnika JK na module przerzutników.

Na każdym z modułów przerzutników znajdują się po 4 przerzutniki D i JK. Mają one podobnie oznaczone wejścia i wyjścia. Jeśli n to numer przerzutnika danego typu, to wejście zegarowe oznaczono CLKn, wejścia asynchronicznego ustawiania $\overline{\text{PREn}}$, wejście asynchronicznego zerowania $\overline{\text{CLRn}}$, wyjścia natomiast Qn i $\overline{\text{Qn}}$. Wejścia danych oznaczono odpowiednio Dn dla przerzutników D oraz Jn i Kn dla przerzutników JK.

Moduły wyposażono w rezystory wymuszające wysoki stan logiczny na wszystkich niewykorzystanych wejściach. W przypadku wejścia zegarowego i danych stanowi to wyłącznie zabezpieczenie przed niepożądanym zachowaniem układu cyfrowego, natomiast dla wejść asynchronicznych zapewnione jest wymuszenie na nich stanu nieaktywnego (wysokiego) gdy nie są one dołączone do źródła sygnału cyfrowego.

# 2. Przykład

Aby zilustrować sposób posługiwania się makietą rozważmy czynności potrzebne do zaprojektowania, skonstruowania i sprawdzenia prawidłowości działania prostego cyfrowego układu kombinacyjnego, opisanego funkcją boolowską $Y = \sum_{3}(0,1,5)$. Jest to układ kombinacyjny z 3-bitowym słowem wejściowym i jednobitowym słowem wyjściowym.

## 2.1. Projekt

Na początku procesu projektowania rozpisuję tabelę prawdy i minimalizuję funkcję boolowską opisującą sposób działania układu. Dla wygody przeprowadzenia weryfikacji prawidłowości działania układu cyfrowego, po przeprowadzeniu procesu minimalizacji, do tabeli prawdy dopisuję również wartości, jakie przyjmą implikanty dla każdego słowa wejściowego.

Tabela prawdy:

| X | Y |
|---|---|
| 000 (0) | 1 |
| 001 (1) | 1 |
| 010 (2) | 0 |
| 011 (3) | 0 |
| 100 (4) | 0 |
| 101 (5) | 1 |
| 110 (6) | 0 |
| 111 (7) | 0 |

Minimalizacja:

| $x_1x_0$ / $x_2$ | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 |

$$Y = \overline{x_2}\,\overline{x_1} + \overline{x_1}x_0$$

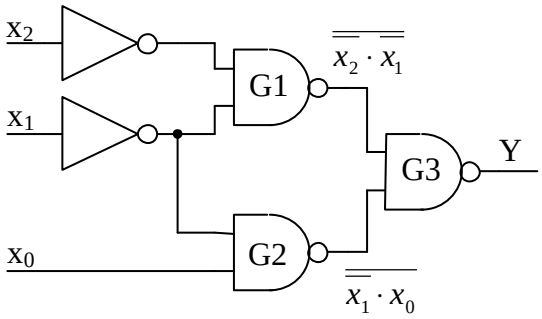
Tabela prawdy z implikantami:

| X | Y | $\overline{x_2}\,\overline{x_1}$ | $\overline{x_1}x_0$ |
|---|---|---|---|
| 000 (0) | 1 | 1 | 0 |
| 001 (1) | 1 | 1 | 1 |
| 010 (2) | 0 | 0 | 0 |
| 011 (3) | 0 | 0 | 0 |
| 100 (4) | 0 | 0 | 0 |
| 101 (5) | 1 | 0 | 1 |
| 110 (6) | 0 | 0 | 0 |
| 111 (7) | 0 | 0 | 0 |

Aby móc skonstruować układ kombinacyjny składający się z bramek NAND, działający zgodnie z otrzymaną minimalną postacią funkcji, należy wyrażenie to przekształcić zgodnie z regułą DeMorgana:

$$Y = \overline{x_2} \cdot \overline{x_1} + \overline{x_1} \cdot x_0 = \overline{\overline{\overline{x_2} \cdot \overline{x_1}} \cdot \overline{\overline{x_1} \cdot x_0}} = \overline{\overline{x_2} \cdot \overline{x_1} \cdot \overline{x_1} \cdot x_0}$$

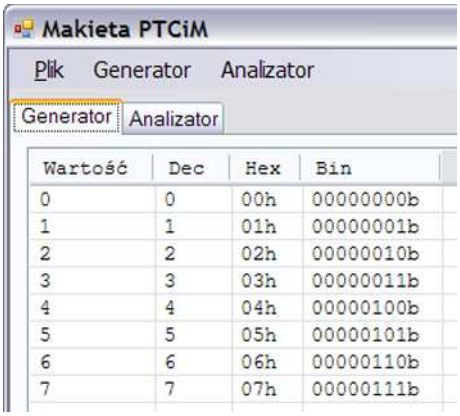
, przez co uzyskuję wyrażenie w postaci zanegowanego iloczynu dwu innych zanegowanych iloczynów, co da się zrealizować następującym układem bramek:




Rysunek 15. Schemat układu.

## 2.2. Przygotowanie makiety

Po zaprojektowaniu układu należy przygotować makietę do jego testowania. Na początku należy zdefiniować sekwencję słów, które sterownik poda na wejścia układu kombinacyjnego. Sprawa jest tu prosta – w tabeli prawdy zapisane mam słowa wejściowe w postaci rosnących wartości liczb kodowanych w kodzie NKB. Przygotowuję zatem generator w sposób, jak na rysunku 16.




Rysunek 16. Ustalenie sekwencji słów wejściowych układu kombinacyjnego.

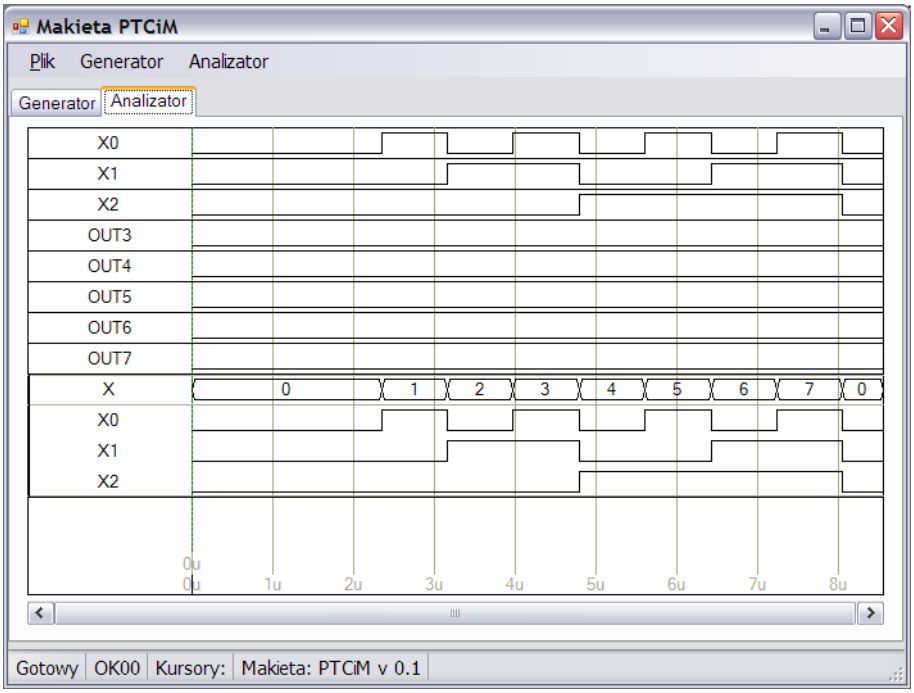
Tak przygotowaną sekwencję warto zapisać *(Generator→Zapisz plik generatora)* i trzeba przesłać ją do makiety *(Generator→Zaprogramuj)*. Po wykonaniu tych czynności można przystąpić do budowania i testowania układu.

## 2.3. Sprawdzenie prawidłowości działania makiety

Do weryfikowania prawidłowości działania układu służy analizator stanów logicznych. Jest on uruchamiany wraz z początkiem działania generatora ale działa w sposób zupełnie niezależny od niego.

Do weryfikowania prawidłowości działania układu trzeba mieć możliwość jednoczesnego obserwowania słów wejściowego, wyjściowego oraz stanów logicznych z wybranych węzłów badanego układu. Zacznijmy od obserwacji stanów logicznych generowanych przez generator na magistrali IN0..IN7.

Aby obserwować słowo wejściowe łączę bezpośrednio wybrane linie magistrali IN liniami magistrali OUT. Interesują mnie wyłącznie trzy najmłodsze bity magistrali IN zatem łączę przewodami IN0 z OUT0, IN1 z OUT1 oraz IN2 z OUT2 (patrz rysunek 21, kolor zielony). Po uruchomieniu makiety (Analizator→Uruchom) powinno dać się zaobserwować następujący przebieg:




Rysunek 17. Sprawdzenie prawidłowości działania generatora i analizatora stanów logicznych.

Dla zwiększenia czytelności nadałem odpowiednie etykiety poszczególnym bitom słowa wejściowego oraz dodałem grupę X. Widać, że makieta działa prawidłowo.

## 2.4. Wykonanie połączeń na makiecie i testowanie układu

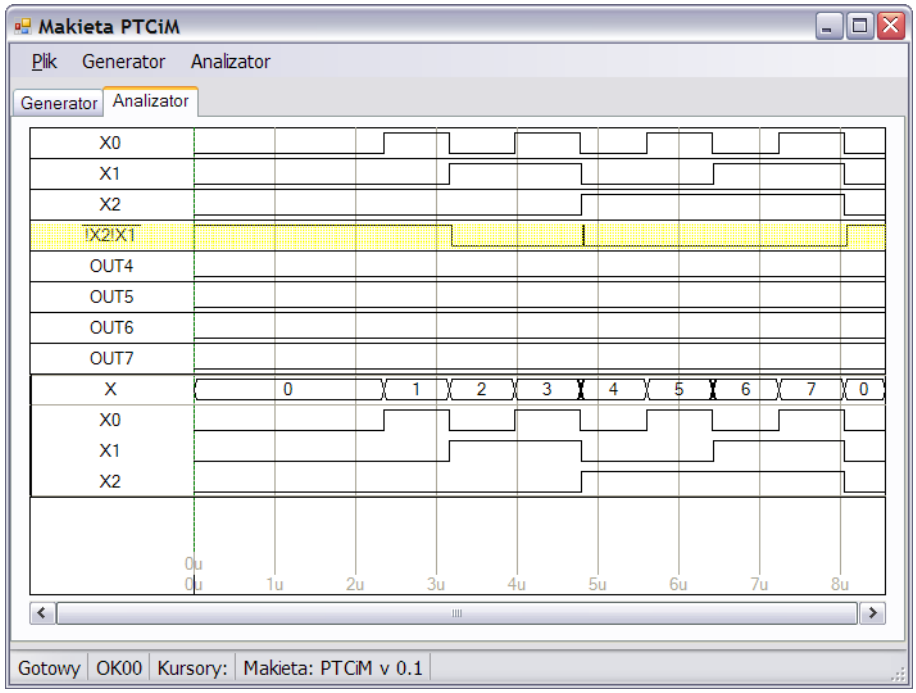
Po sprawdzeniu makiety można rozpocząć budowanie układu. Z projektu wiem, ze wystarczą do tego bramki NOT i NAND. Z analizy wyrażenia algebraicznego wynika, że na wejścia bramek NAND trzeba będzie podawać bit $x_0$ słowa wejściowego oraz zanegowane wartości bitów $x_1$ i $x_2$. Należy zatem na module bramek logicznych połączyć IN1 oraz IN2, na których dostępne są wartości $x_1$ i $x_2$, z wybranymi wejściami bramek NOT, na przykład 1G i 2G. Aby móc teraz przekazać do pozostałych części makiety

wartości $\overline{x_1}$ i $\overline{x_2}$ korzystam z magistrali COM, dołączając NOT1 do COM1, NOT2 do COM2 (patrz rysunek 21, kolor czerwony). Wtedy na wszystkich modułach magistrali wartości dwu zanegowanych bitów słowa wejściowego będą dostępne na złączach COM1 i COM2.

**Konstruowanie właściwego układu kombinacyjnego warto wykonywać systematycznie, dokładając do niego kolejne implikanty i za każdym razem sprawdzając prawidłowość działania nowostworzonej części układu**.

Łączenie elementów na makiecie rozpoczynam od wykonania fragmentu układu, który będzie ustalał wartość pierwszego implikantu $\overline{x_2}\,\overline{x_1}$. Jak wynika z rysunku 15, zanegowana wartość tego implikantu wyznaczana jest w przez bramkę G1, na którą podane mają być $\overline{x_1}$ i $\overline{x_2}$. W układzie można to uzyskać połączenie linii COM1 i COM2 z wejściami jednej z bramek NAND (patrz rysunek 21, kolor żółty).

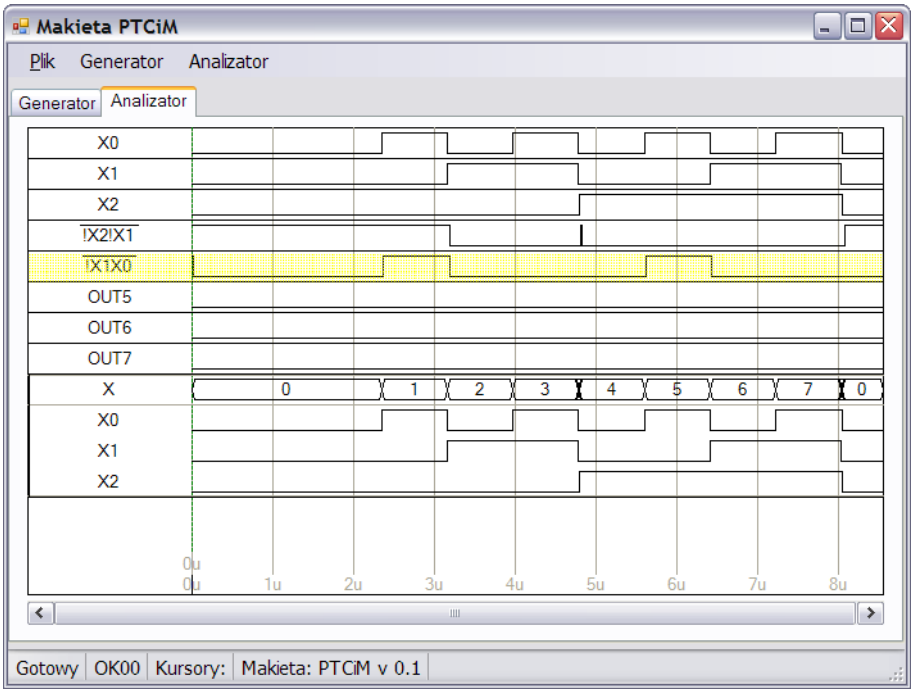
Dla sprawdzenia prawidłowości działania tej części układu dołączam wyjście bramki G1 do OUT3, która jest pierwszą wolną linią magistrali analizatora stanów logicznych i uruchamiam makietę. Po nadaniu w programie właściwej etykiety linii OUT3 i oznaczeniu jej jako zanegowana uzyskuję następujący wykres:




Rysunek 18. Makieta z prawidłowo działającym pierwszym implikantem.

Uzyskuję wysoki stan logiczny dla słów wejściowych 0 i 1, co odpowiada tabeli prawdy dla tego implikantu. Widoczny jest przejściowy stan logiczny niezgodny z funkcją boolowską przy przejściu z X=011 do X=100.

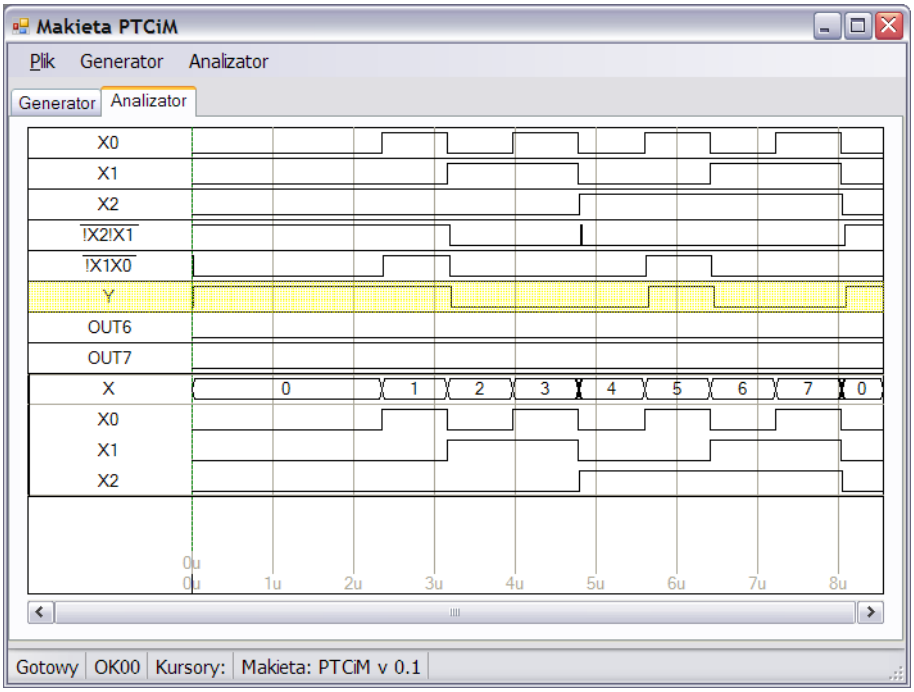
W analogiczny sposób łączę układ obliczający zanegowaną wartość drugiego implikantu (bramka G2), łączę wyjście bramki z OUT4 i uruchamiam makietę (patrz rysunek 21, kolor różowy). Po nazwaniu w odpowiedni sposób linii OUT4 powinienem otrzymać przebieg, jak rysunku:



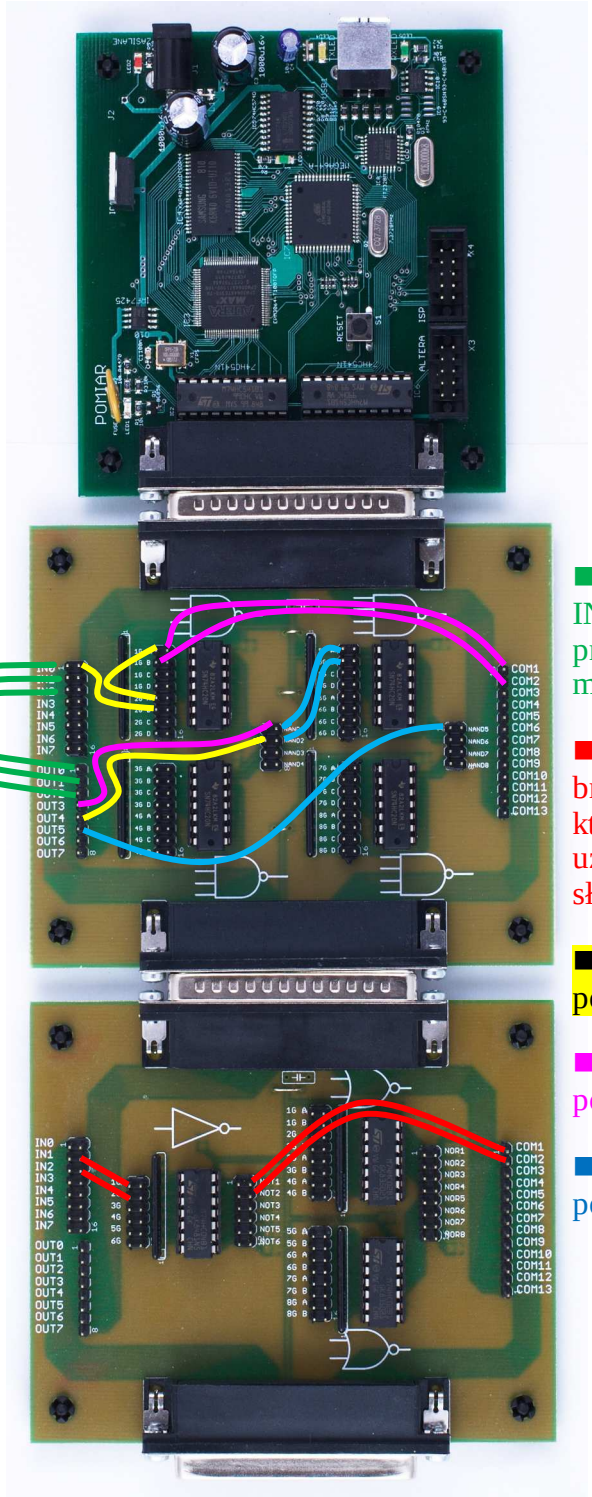

Rysunek 19. Makieta z prawidłowo działającymi dwoma implikantami.

Uzyskałem wysoki stan logiczny dla słów wejściowych 1 i 5, co zgadza się z tabelą prawdy tego implikantu.

Dopiero teraz dodaję do układu trzecią bramkę G3, która będzie ustalała stan bitu wyjściowego z układu, łączę jej wyjście z linią OUT5 (patrz rysunek 21, kolor niebieski), uruchamiam makietę, nadaję odpowiednią etykietę tej linii w programie. Powinienem otrzymać przebieg czasowy jak na rysunku:




Rysunek 20. Prawidłowo działająca makieta.




■ połączenia IN0-OUT0, IN1-OUT1, IN2-OUT2 do przetestowania działania makiety

■ połączenia na module bramek logicznych, dzięki którym na COM1 i COM2 uzyskałem zanegowane bity słowa wejściowego $x_1$ i $x_2$

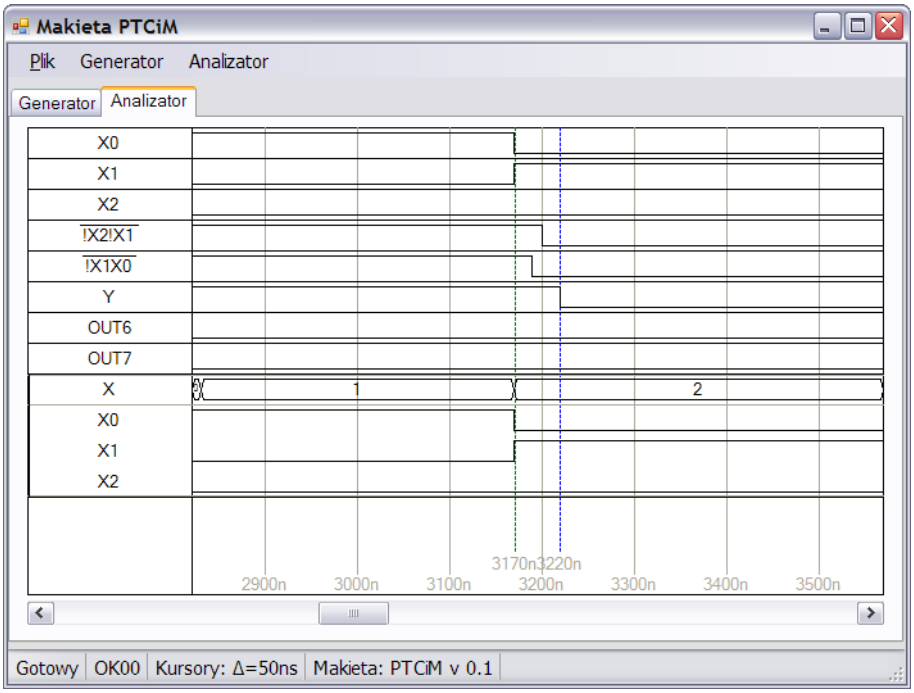
■ pierwszy implikant i jego połączenie z OUT3

■ drugi implikant i jego połączenie z OUT4

■ bramka końcowa i jej połączenie z OUT5

Rysunek 21. Kolejne fazy wykonywania połączeń na makiecie.

Analizator stanów logicznych pracuje z częstotliwością próbkowania 100MHz. Jest to częstotliwość wystarczająca dla zaobserwowania skończonych czasów propagacji w układach rodziny 74HC zasilanych napięciem 3,3V. Po powiększeniu fragmentu przebiegu, przy którym słowo wejściowe zmienia się z X=001 na X=010 można zaobserwować przesunięcia w czasie zmian stanów logicznych w poszczególnych elementach układu a korzystając z kursorów, można te czasy zmierzyć z rozdzielczością 10 ns.




Rysunek 22. Obserwacja czasów propagacji sygnału w układzie cyfrowym.

Całkowity czas propagacji sygnału, mierzony od chwili zmiany stanów logicznych na wejściu układu, wyniósł 50 ns.

# 3. Dodatki

## 3.1. Szczegóły budowy sterownika makiety

W przygotowaniu.