

LABORATORIUM - ELEKTRONIKA
Liczniki scalone – działanie, wykorzystanie, konfiguracja

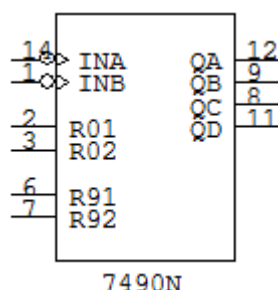
1. Cel ćwiczenia

Celem ćwiczenia jest pokazanie podstawowych rozwiązań liczników scalonych, zasady ich działania i realizowanych przez nie funkcji, a także ich konfiguracji do działania w określony sposób.

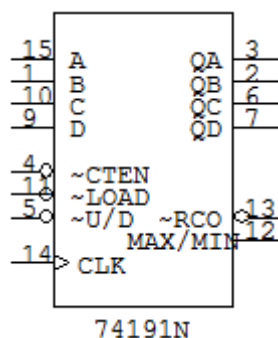
Dodatkowo w trakcie ćwiczenia poznaje się program do symulacji układów elektronicznych MultiSIM.

2. Wstęp teoretyczny

Licznik scalony, jak sama nazwa wskazuje, jest układem scalonym służącym do liczenia/zliczania. Liczba bitów wyjścia danego układu definiuje "do ilu" w danym układzie można liczyć. Najczęściej stosowane są liczniki 4-bitowe, co oznacza, że można na nich liczyć od 0000 do 1111 (binarnie), czyli od 0 do 15 (dziesiętnie). Spotyka się też liczniki scalone dekadowe, czyli zliczające od 0000 do 1001, więc od 0 do 9. Takim licznikiem jest na przykład układ scalony o oznaczeniu 7490:



Z kolei licznikiem scalonym liczącym w pełnym zakresie 4-bitowym jest przykładowo układ 74191:



Przeanalizujmy działanie tego układu. Widzimy licznik. Ma on 14 widocznych pinów. W MultiSIMie przyjęto, że piny „po lewej” to wejścia, piny „po prawej” to wyjścia, a zatem układ ten ma 8 wejść i 6 wyjść. Należy pamiętać, że gdybyśmy używali tego układu w rzeczywistości, to ma on jeszcze dwa niewidoczne tu piny, do których należałoby podłączyć zasilanie.

Jak używać takiego licznika?

Należy zrobić to, co się robi zawsze, gdy chcemy zidentyfikować lub użyć jakiegokolwiek układu scalony - szukamy jego noty katalogowej, np. na www.datasheetcatalog.com. Co można znaleźć w takiej notce?

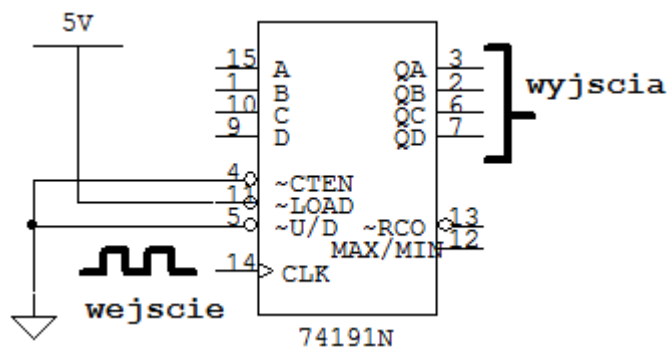
Na pewno informację, że pin CLK to wejście liczące tego układu. Tu właśnie podamy zerojedynkowy sygnał, który będzie zliczany. To podstawa. Podobnie jak fakt, że piny Q_D, Q_C, Q_B i Q_A to wyjścia licznika. To tu kolejno będą się pojawiały zliczane liczby (oczywiście w postaci dwójkowej). Warto przy tym pamiętać, że zasadą jest, że Q_A jest bitem najmniej znaczącym, tzn. odpowiada on 2⁰=1, natomiast Q_D jest bitem najbardziej znaczącym, tzn. odpowiada on 2³=8.

Aby użyć tego układu musimy jeszcze zadbać o piny ~CTEN, ~LOAD i ~U/D. Pozostałe możemy na razie zaniedbać.

Po kolei:

- Pin ~U/D w nocie katalogowej jest opisany jako „up/down input”. Sprawa jest banalna. Tym pinem decydujemy jak ma liczyć licznik tzn. w górę czy w dół. Falka przy opisie (oraz kółko przy wejściu na schemacie graficznym) oznacza negację, a zatem by liczyć w górę (up), na ten pin należy podłączyć zero, by w dół (down) – jedynkę;
- Pin ~LOAD jest opisany jako „asynchronous parallel load input”. Jest to zatem wejście decydujące o zerowaniu licznika, czy też mówiąc ogólniej, o narzuceniu licznikowi pewnego stanu początkowego (który, jak się przekonamy później, niekoniecznie musi oznaczać zero). Ma on falkę, co oznacza, że aby wyzerować licznik należy podać tu zero, natomiast żeby licznik „normalnie” zliczał, powinna być tu podana jedynka.
- Pin ~CTEN jest opisany jako „count enable input”. Jest to zatem wejście które „pozwała” liczyć. Też ma falkę, czyli żeby licznik mógł liczyć, należy tu podać zero.

Wiemy już zatem, że licznik skonfigurowany jak na rysunku poniżej będzie zliczał impulsy podawane na wejście CLK w górę, w zakresie od 0 do 15, a swój aktualny stan podawał na wyjścia Q_D, Q_C, Q_B i Q_A.



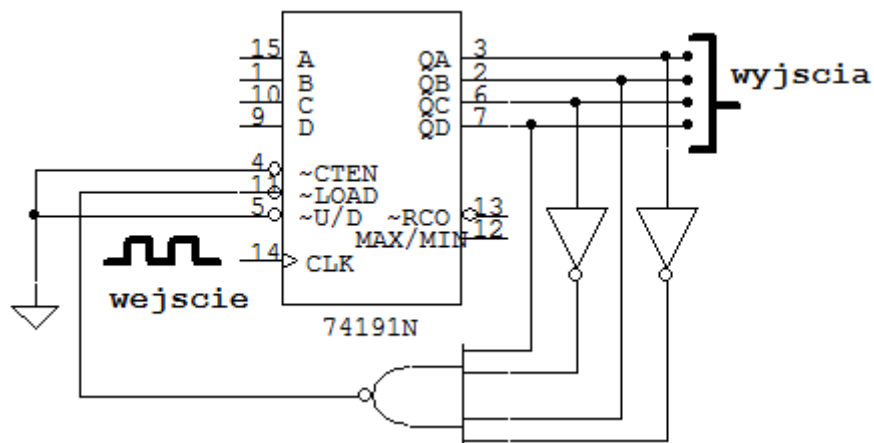
Skracanie cyklu licznika

A co jeżeli nie chcemy liczyć do 15, a np. do 9? Wtedy właśnie wykorzystuje się wejście ~LOAD. Jak już wiemy, podanie na ten pin logicznego zera wyzeruje układ. Ale skąd wziąć to zero? Powinno się ono pojawić automatycznie, w chwili gdy licznik doliczy do 10. Warto podkreślić, że do 10, nie do 9 (mimo, że to właśnie do 9 chcemy zliczać), ponieważ licznik nie powinien być zerowany w momencie, kiedy doliczy do 9, ale dopiero gdy „przekroczy” 9. Czyli pozwalamy mu zliczyć do 9, a dopiero gdy chce on przeskoczyć na 10, wykrywamy ten moment i zerujemy licznik.

Zatem wykrywamy dziesiętną 10, czyli binarnie 1010 (tj. $1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 1 \cdot 8 + 0 \cdot 4 + 1 \cdot 2 + 0 \cdot 1 = 8 + 2 = 10$). Jest to stan 1010 pojawiający się oczywiście się na pinach wyjściowych Q_D, Q_C, Q_B i Q_A (skoro, jak wspomniano wcześniej, Q_A jest bitem najmniej znaczącym, to wykrywamy stan Q_DQ_CQ_BQ_A=1010). Jak więc to zrobić?

Wykrywamy konkretny stan – 1010. Chcemy zerować układ dokładnie w momencie, gdy Q_D=1 i Q_C=0 i Q_B=1 i Q_A=0. Kluczem jest tu użyty spójnik - „i”. Skoro go tam wstawiłem, to chcę tam wykonać operację logiczną I czyli AND, co znaczyłoby, że do układu wstawić należy czterowejsiową bramkę AND, która to wykryje ten stan.

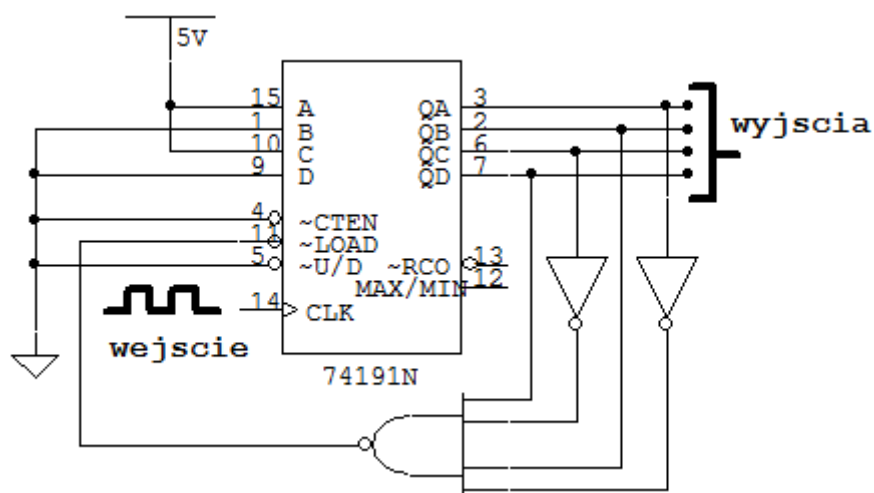
No, prawie, zapomnieliśmy o jednym. Bramka AND daje na swoim wyjściu jedynkę, gdy na jej wejściach są jedynki. A my mamy na pinie zerującym LOAD „falkę”, czyli zerowanie nie następuje w wyniku podania tam jedynki, lecz zera. To znaczy, że musimy jeszcze odwrócić (zanegować) wyjście tej bramki. Sposoby na to są dwa - albo negujemy wyjście bramki AND bramką NOT, albo po prostu zamiast bramki AND wstawiamy bramkę NAND (NAND = NOT AND). Jeżeli skorzystamy z drugiego z tych sposobów, to układ liczący od 0 do 9 zrealizowany na liczniku 74191, wygląda jak na rysunku poniżej.



Jak widać na tym właśnie rysunku, na wejścia bramki NAND podane są bezpośrednio wyjścia Q_D i Q_B , bo to na tych wyjściach, zgodnie z tym co opisano wyżej, wykrywamy jedynki, natomiast wyjścia Q_C i Q_A podane są przez bramki NOT, bo to na nich wykrywamy zera, więc trzeba je "zmienić" w jedynki. Dzięki temu bramkę NAND podane zostaną 4 jedynki i licznik wyzeruje się w żądanym momencie.

Dobrze, nauczyliśmy się zerować licznik w dowolnym momencie. Co jeszcze można z nim zrobić? Skoro wiemy jak skończyć liczenie w dowolnym momencie, to pojawia się kolejne pytanie - jak zacząć liczenie od określonego stanu? Jeśli chcemy liczyć nie od 0, ale np. od 5? Do zmiany stanu wejściowego licznika służą piny wejściowe D, C, B i A. Skoro chcę liczyć od 5, to na te właśnie piny muszą po prostu podać binarny odpowiednik 5 czyli DCBA=0101 (tj. $0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 0 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 = 4 + 1 = 5$).

Końcowy układ wygląda zatem tak:



Piny D i B są podłączone do masy, czyli do logicznego zera, natomiast piny C i A podłączone są do $V_{CC} = 5 [V]$, czyli do logicznej jedynki, a skoro wcześniej zadaliśmy o to, by układ zerował się po zliczeniu to 9, to w konsekwencji liczy on teraz **w zakresie od 5 do 9**.

Stosując opisaną wyżej metodykę można skonfigurować licznik do zliczania w dowolnym (oczywiście maksymalnie 4-bitowym) zakresie, zarówno w górę, jak i w dół (po zadaniu o odpowiednie podłączenie pinu $\sim U/D$).

W celu zbudowania układu zliczającego w szerszym zakresie liczniki można ze sobą łączyć – metodykę postępowania w takich przypadkach opisano w odpowiednim DODATKU - dostępnym na stronie przedmiotu.

3. Kilka słów o programie MultiSIM

Program MultiSIM (dawniej Electronics Workbench) jest jednym z najpopularniejszych narzędzi do projektowania i analizy obwodów elektronicznych. Oprogramowanie to pozwala w prosty i intuicyjny sposób symulować, analizować oraz projektować układy wszelkiego rodzaju układy elektroniczne - analogowe i cyfrowe.

Aby zbudować dany układ w programie MultiSIM należy „poukładać” odpowiednio elementy i układy w oknie programu i je połączyć. Połączenia powstają w sposób bardzo prosty – wystarczy kliknąć w dwa punkty które chcemy ze sobą połączyć.

Elementy i układy potrzebne w budowanym układzie najprościej jest wstawić klikając w oknie program prawym przyciskiem myszy i wybierając „Place Component”. Pojawi się wtedy okno w którym można wybrać potrzebne komponenty. W przypadku układów budowanych podczas tego ćwiczenia można je znaleźć w następujących grupach:

- zasilanie V_{CC} , i GND – **Sources** → **POWER_SOURCES**,
- wyświetlacz siedmiosegmentowy typu wspólna anoda – **Indicators** → **HEX_DISPLAY**,
- licznik 74191N – **TTL** → **74STD**,
- bramki logiczne potrzebne w punkcie 4.2.4 można znaleźć w grupie **TTL->74STD**:
 - 7400 – 4 x bramka NAND 2-wejściowa,
 - 7410 – 3 x bramka NAND 3-wejściowa,
 - 7420 – 2 x bramka NAND 4-wejściowa,
 - 7430 – 1 x bramka NAND 8-wejściowa,
 - 7404 – 6 x bramka NOT,
- urządzenia pomiarowe bardziej skomplikowane, jak np. potrzebne w tym ćwiczeniu generator (**Function Generator – XFG1**) i analizator stanów logicznych (**Logic Analyzer – XLA1**) są dostępne po prawej stronie okna programu.

Jeżeli dany komponent układu chcemy w jakiś sposób zmodyfikować, odwrócić itp., należy kliknąć nań prawym przyciskiem myszy – wszelkie potrzebne opcje są tam dostępne.

Jeżeli chcemy zmienić parametry działania jakiegoś układu trzeba kliknąć go dwukrotnie.

Gdy układ jest gotowy, wystarczy kliknąć widoczny u góry po prawej stronie okna programu „włącznik O/I” – zacznie się symulacja. Można też zamiast tego używać skrótu klawiaturowego – symulację startuje / zatrzymuje klawisz F5.

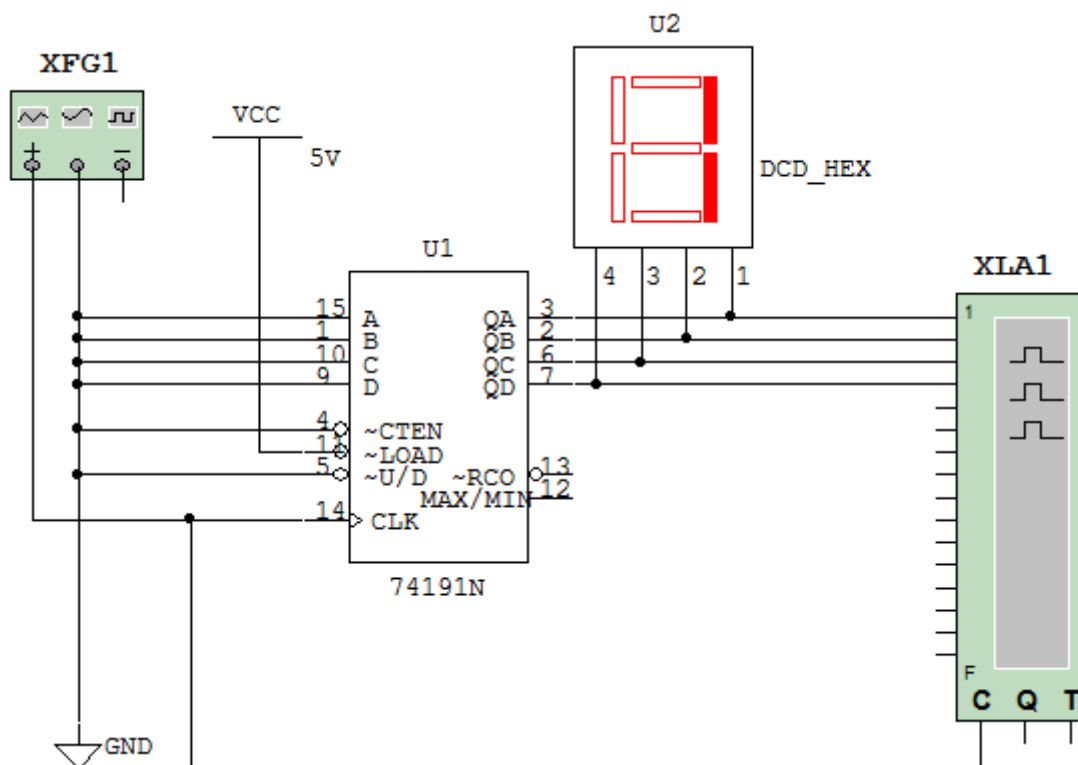
Dodatkowe uwagi:

- **należy pamiętać, że po każdej zmianie wartości (np. amplitudy lub częstotliwości sygnału wejściowego) należy ponownie uruchomić układ, żeby program w nowej symulacji uwzględnił wprowadzone zmiany,**
- **układu nie da się modyfikować gdy jest „włączony”, wszelkie zmiany w układzie da się wprowadzić dopiero gdy symulacja jest wyłączona,**
- **program numeruje wprowadzane komponenty w kolejności ich wstawiania, oczywiście numeracja nie ma żadnego wpływu na działanie układu :)**

4. Przebieg ćwiczenia

4.1. Zrealizować w programie MultiSIM układ z licznikiem '191 wg rysunku:

(na poprzedniej stronie instrukcji wskazano gdzie w programie tym można znaleźć wszystkie elementy/układy elektroniczne wykorzystywane w tym ćwiczeniu)



4.2. Ustawić następujące parametry układów (parametry zmienia się dwukrotnie klikając w dany układ):

- generator XFG1: sygnał TTL, czyli:
 - + kształt prostokątny,
 - + amplituda (*Amplitude*) = 2.5 [V],
 - + składowa stała (*Offset*) = 2.5 [V],
 - o częstotliwości (*Frequency*) = 0.5 [kHz],
- analizator stanów logicznych XLA1: źródło sygnału zegarowego (przycisk *Set* w sekcji *Clock*) – zewnętrzne (*External*).

Pozostałe parametry pozostawić bez zmian.

4.3. Otworzyć okno analizatora stanów logicznych, następnie uruchomić symulację.

- 4.3.1. W oknie analizatora (po dwukliku) pojawią się przebiegi czasowe z wejść i wyjść układu 74191, zliczanie powinno być też widoczne na wyświetlaczu siedmiosegmentowym. W przypadku gdyby zliczanie było zbyt szybkie lub zbyt wolne można zmniejszyć/zwiększyć częstotliwość pracy generatora XFG1.
- 4.3.2. Naszkicować w protokole te przebiegi w zakresie jednej pełnej pętli licznika (czyli od stanu 0000 do 0000).
- 4.3.3. Na przebiegach z wejścia (C) i wyjść (Q_A, Q_B, Q_C, Q_D) układu uzyskanych na laboratorium zaznaczyć okresy wyświetlania poszczególnych liczb.

4.4. Zaprojektować i zrealizować na tym liczniku układy zliczające w zadanych przez prowadzącego zakresach.
Sposób konfiguracji licznika scalonego do liczenia w zadanym zakresie opisano we wstępie do ćwiczenia.
Sprawdzić poprawność działania swoich układów.

4.5. Ewentualne wnioski i przemyślenia można zamieścić na odwrocie protokołu.

5. Literatura

- [1] Filipkowski A., „Układy elektroniczne analogowe i cyfrowe”, WN-T, Warszawa 1978
- [2] Głocki W., Grabowski L., „Pracownia podstaw techniki cyfrowej”, WSiP, Warszawa 1998
- [3] Górecki P., „Układy cyfrowe, pierwsze kroki”, Wydawnictwo BTC, Warszawa 2004
- [4] Kalisz J., „Cyfrowe układy scalone w technice systemowej”, WMON, Warszawa 1997
- [5] Pieńkos J., Turczyński J., „Układy scalone TTL w systemach cyfrowych”, WKiŁ, Warszawa 1986
- [6] Sasal W., „Układy scalone serii UCA64/UCY74, parametry i zastosowania”, WKiŁ, Warszawa 1985
- [7] www.datasheetcatalog.com

Opracowanie ćwiczenia: Seweryn Lipiński