Michał Sobczak

Simple High Performance Computing

Ridero

2021

ISBN 978-83-8273-126-2
Książka powstała w inteligentnym systemie wydawniczym Ridero

Spis treści

> WSTĘP

„To invent, you need a good imagination and a pile of junk.” — *Thomas Alva Edison*

O Książce

Jest to pierwsza część serii *Simple High Performance Computing* o tytule ***„Podstawy elektroniki i budowa komputera w symulatorze”***. Swoim zakresem obejmuje zagadnienia związane z elektroniką, a konkretnie jej podstawami, począwszy od pierwotnych zasad fizyki przekładających się na komponenty elektroniczne, z których można budować złożone układy, a finalnie najprostszą obliczeniową maszynę cyfrową. Największy nacisk kładę na praktyczną część zagadnienia, aczkolwiek teoria również jest przedstawiana.

O Serii

Seria *Simple High Performance Computing* dotyczy przetwarzania wysokiej wydajności. Ma w zamyśle składać się z przynajmniej sześciu części. Uszeregowanie kolejnych części względnie postępuje wraz z poziomem złożoności zagadnienia, aczkolwiek na pomysł bieżącej części wpadłem dopiero po jakimś czasie. Zauważyłem bowiem, że temat wydajnego przetwarzania będzie bez tego mocno niekompletny. Było to dla mnie również odświeżenie wiedzy jak i całkiem ciekawa przygoda, w szczególności budując cyfrową maszynę obliczeniową.

Przykłady i konwencje

Przykłady do niniejszej książki znajdują się w repozytorium pod adresem

https://github.com/michalasobczak/simple_hpc

w folderze **SeriesPartOne**.

Przy poszczególnych zrzutach ekranu w nawiasach okrągłych znajduje się informacja, którego przykładu rysunek dotyczy, np. **(p28)**. Liczby w nawiasach kwadratowych oznaczają pozycję bibliografii, np. **[007]**. Bibliografia znajduje się na końcu książki. Rysunki, ilustracje i diagramy oznaczone są w treści jako **Rys**. Tabele oznaczone są jako **Tab**. Wzory oznaczone są jako **Wz**.

> PODSTAWY ELEKTRONIKI

Od teorii aż po budowę układów filtrujących. Ten rozdział to początek przygody. Kolejne rozdziały będą opisywały tematykę na coraz wyższym poziomie. Jednak to dzięki podstawom wiedza będzie uporządkowana.

Bardzo interesuje mnie **cały szereg procesów** z jakimi mamy do czynienia w komputerach, począwszy od podania **zasilania** a skończywszy na wyniku działania konkretnego **programu** na ekranie. Aby jednak dotrzeć do tego miejsca, gdzie uzyskujemy jakiś wynik, czeka całkiem długa droga. Na tej drodze na samym początku pojawiają się podstawowe **aspekty elektroniki** takie jak napięcie, prąd, opór, dioda, tranzystor, bramka czy układ scalony. Dla mnie również będzie to forma **przypomnienia** materiału ze studiów. Pozwoli to nieco lepiej zrozumieć istotę rzeczy jaką jest **przetwarzanie informacji**. Ten rozdział dotyczy elektroniki w układzie analogowym. Układy cyfrowe będą przedstawione w kolejnych rozdziałach.

*Uwaga: w kolejnych rozdziałach omawiać będę budowę **komputera cyfrowego**. Istnieją również **komputery analogowe**, lecz ich zastosowanie jest dużo bardziej ograniczone i służy rozwiązywaniu szczególnych problemów. Stosowaną miarą i nośnikiem w takich komputerach są **fizyczne własności**. [007]*

Napięcie

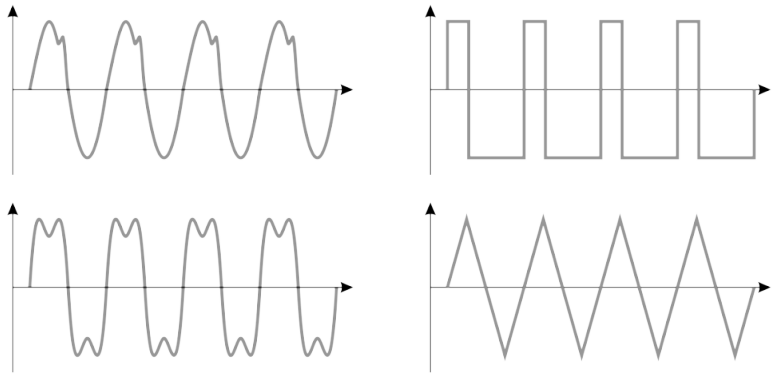
Miara sił ładunków elektrycznych. Różnica potencjału elektrostatycznego.

Elektronika bazuje na **fizyce**. Podstawowymi tutaj pojęciami są **napięcie**, **prąd** i **opór**. Od tego powinniśmy zacząć. Napięcie jest zatem **miarą siły ładunków elektrycznych**. Napięcie między dwoma punktami obwodu elektrycznego lub pola elektrycznego jest **różnicą potencjału elektrostatycznego** pomiędzy tymi punkami. Napięcie mierzymy w **woltach**. Do pomiaru napięcia używamy **woltomierza**. Napięcie zawsze mierzymy **między dwoma punktami**, ponieważ jest to różnica potencjałów. Napięcie może być **stałe**, **zmienne**, **przemienne**. To, dlaczego korzystamy z danej formy napięcia zależne jest od przeznaczenia, strat, sprawności i zapewne też bezpieczeństwa. Różnica pomiędzy zmiennością a przemiennością to kwestia **kształtu przebiegu** i jego rozkładu na osi. Napięcie i prąd to pojęcia nierozerwalnie ze sobą związane. Jeśli napięcia w obwodzie nie ma, to nie płynie w nim prąd.

Przemienność

Okresowa forma sygnału z przebiegami po obu stronach osi

[204] Prąd przedstawiony jako **kształt**, sygnał może mieć różne postaci, w zależności w jaki sposób jest on inicjowany. Jeśli nadamy mu daną formę okresową, w której jego wartość **przemiennie będzie przybierała ujemną i dodatnią** postać, wówczas taki prąd nazwiemy przemiennym.

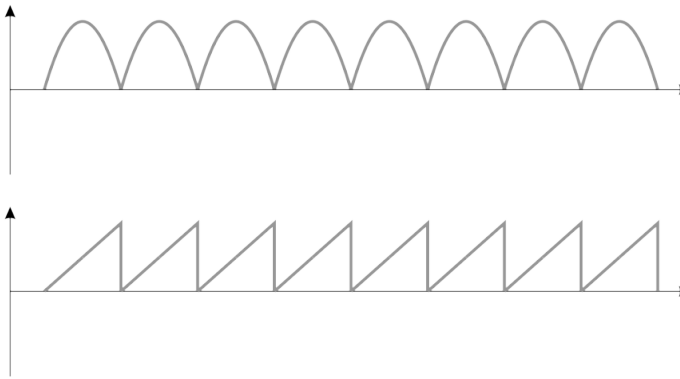


Rys. Przebiegi przemienne [204]

Zmienność

Okresowa forma sygnału po jednej stronie osi

[204] Jeśli jednak będzie ta forma **okresową, ale nie będzie przemienna**, wówczas będziemy mogli powiedzieć jedynie o prądzie okresowym, ale zmiennym. Sygnał taki jest powtarzalny, ale jego przebieg w zakresie amplitudy nie przekracza osi na wykresie. Różnica pomiędzy sygnałem stałym, zmiennym i przemiennym jest istotna z punktu widzenia projektowania układów. Jeśli dana postać sygnału czy to zmienna czy przemienna powinna mieć określoną formę, to zadaniem różnego rodzaju komponentów i układów będzie im tą docelową formę dostarczyć w możliwie prosty sposób.



Rys. Przebiegi zmienne [204]

Częstotliwość i napięcie szczytowe

Amplitudy sygnału a faktyczne napięcie

[204] W gniazdku w domu mamy prąd przemienny o określonych parametrach, np. napięciu 230V oraz częstotliwości 50Hz. Oznacza to, że w 1 sekundzie ma miejsce **50 zmian (przemian) z wartości szczytowej**, po czym spadając, przecinana jest oś w punkcie zero (a dokładniej przez **poziom równowagi**), by osiągnąć przeciwstawną wartość. Zmiany te tworzą **sinusoidę prądu przemiennego**. W sieci może wystąpić napięcie o określonym **odchyleniu**, pomiędzy 5 a 10% w zależności od charakterystyki dostarczanego napięcia jak i samej **instalacji** będącej jego nośnikiem. Zatem przyjmując wartości maksymalne jest to około 250V **napięcia skutecznego**. Dla przebiegu idealnie sinusoidalnego wartością szczytową sygnałów przemiennych będzie

$$U_{max} = U_{sk} * \sqrt{2}.$$

Dla przebiegu trójkątnego byłby to odpowiednio pierwia-

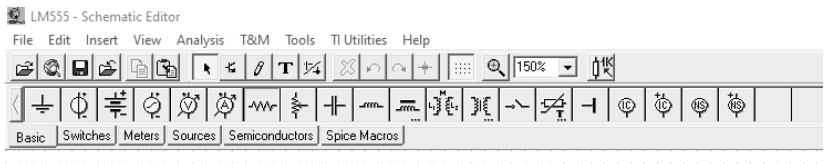
stek z 2. Dla przebiegu kwadratowego będzie to 1. Dla przeciętnego napięcia w gniazdku **napięcie szczytowe** może wynosić około 350 — 360V.

Uwaga: Amplituda to inaczej maksymalne ***odchylenie sygnału od poziomu równowagi*** *Amplitudę napięcia oznaczamy jako Um, a prądu jako Im. Kończąc ten temat warto wspomnieć, że* ***kształt przebiegu prądu*** *jest sinusoidalny ponieważ rdzeń generatora pracuje w ruchu kołowym, który to jest nierozerwalnie związany właśnie z kształtem sinusoidy.*

Symulacja obwodów

Budowa i badanie obwodów w programie TINA TI

Przekładając teorię na praktykę możemy posłużyć się pakietem **TINA** od DesignSoft w wersji dla Texas Instruments. Jest to narzędzie do budowy **wirtualnych układów analogowych jak i cyfrowych**. Podstawowa wersja udostępnia minimalny niezbędny zestaw elementów, które można układać w formie obwodu. Najważniejszą jednak funkcją (przynajmniej dla mnie) jest **możliwość przeprowadzania symulacji**.



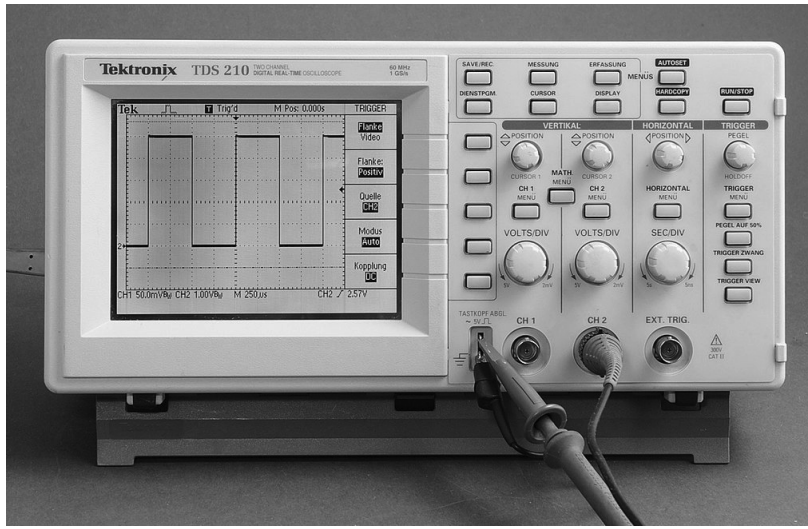
Rys. Dostępne elementy obwodu w pakiecie TINA

Wszystkie prezentowane przykłady symulacji są przygotowane właśnie w programie TINA TI V9 (Schematic Editor), Special Complementary Basic Edition dostępnej nieodpłatnie. Źródła schematów znajduje się w **repozytorium do niniejszej publikacji, do samodzielnego studiowania**

github.com/michalasobczak/simple_hpc

w folderze

SeriesPartOne



Rys. Oscyloskop [W]

[101] Do celów badania układu przyjmuje się pojęcia **liniowości** i **nieliniowości**. Układ z przynajmniej jednym elementem nieliniowym będzie układem nieliniowym. **Nieliniowość polega na braku możliwości opisania charakterystyki prądowo-napięciowej danego elementu w postaci równania prostej**. Opisując układ mieszany, składający się zarówno z elementów liniowych jak i nieliniowych punkt przecięcia ich charakterystyk będzie punktem pracy.

> *Uwaga: Jeśli nie mamy potrzeby testować obwodu możemy zastosować inne pakiety dla elektroniki takie jak Frizting, LibrePCB czy EAGLE.*

Prawo Ohma

Teoria i praktyka

[204] Jest to **jednostka rezystancji w układzie SI**. 1 om to opór elektryczny istniejący pomiędzy dwoma punktami przewodu, gdy niezmienna różnica potencjałów jednego wolta działająca między tymi dwoma punktami wywołuje w tym przewodzie prąd o natężeniu jednego ampera.

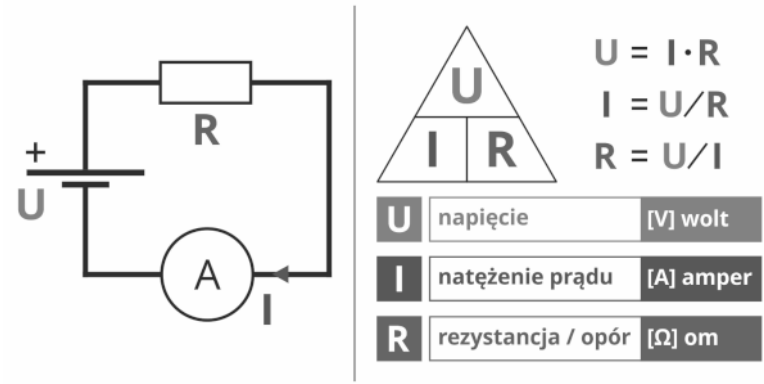
$$1\,\Omega = \frac{1\,\mathrm{V}}{1\,\mathrm{A}} = \frac{1\,\mathrm{kg}\cdot 1\,\mathrm{m}^2}{1\,\mathrm{s}^3\cdot 1\,\mathrm{A}^2}.$$

Wz. Jednostka rezystancji

Większość, jeśli nie wszystkie, elementy posiadają **wewnętrzną rezystancję**, np. baterie. Nie jest to własność w prosty sposób mierzalna. Obwody mające na celu pokazać pewne zasady przyjmują, że przykładowo źródło zasilania jest ogniwem idealnym a wszelkie wewnętrzne opory są kompensowane przez dodatkowe komponenty takiego układu z przyjęciem pewnego **marginesu błędu**.

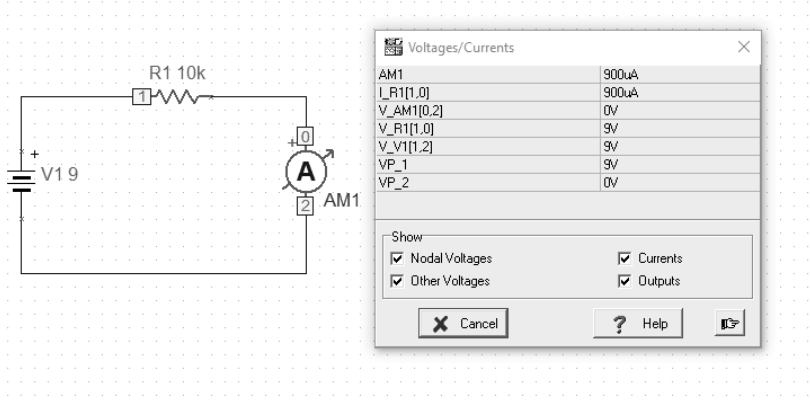
Podanie w układzie jednoelementowym rezystora o zbyt małym oporze powoduje *de facto* **zwarcie styków źródła zasilania**, np. baterii co zapewne doprowadzi do jej degradacji. Im większy **przekrój przewodu**, tym mniejsza jest jego rezystancja i większa powierzchnia zdolna oddawać ciepło — stąd przewód taki wytrzyma większe natężenie prądu.

[202] Natężenie prądu płynącego przez przewodnik jest wprost proporcjonalne do napięcia przyłożonego między jego końcami. **Jeśli zachowamy stały opór, wówczas zwiększanie napięcia powoduje zwiększenie natężenia prądu.** Przy stałym napięciu, zwiększenie oporu powoduje zmniejszenie natężenia prądu.



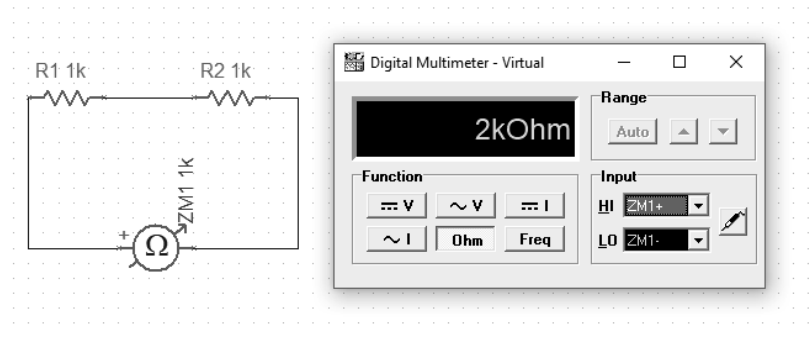
Rys. Prawo Ohma [204]

[202] Przy założeniu, że bateria jako źródło energii elektrycznej ma napięcie 9V a rezystor wskazuje opór 10k Ω, to na amperomierzu, który wpina się w obwód szeregowo, otrzymamy wynik około 900mA. Wynik ten powinien obejmować również **wewnętrzną rezystancję amperomierza** oraz faktyczne napięcie baterii. Należy również brać pod uwagę **rezystancję samego obwodu**. Wynik w postaci 0,9A będzie zatem jedynie **przybliżeniem**.



Rys. Przykładowy obwód z baterią, rezystorem i amperomierzem (p01)

Kolejny przykład pokaże w jaki sposób mierzymy rezystancję oporników łączonych **szeregowo**. Jest to po prostu ich **suma**.



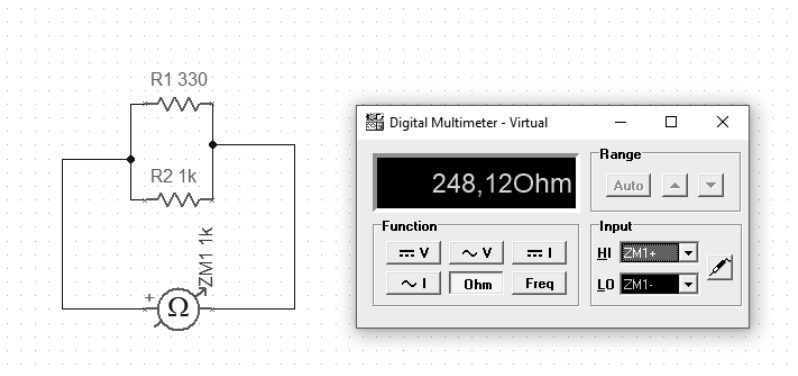
Rys. Suma wartości rezystancji szeregowej (p02)

Dla porównania możemy podłączyć rezystory **równolegle**.

Tutaj obliczenie wartości wyjściowej rezystancji jest nieco inne i opisane jest **wzorem**:

$$R = (R1 * R2)/(R1 + R2)$$

Wz. Rezystancja równolegle



Rys. Połączenie rezystorów równolegle (p03)

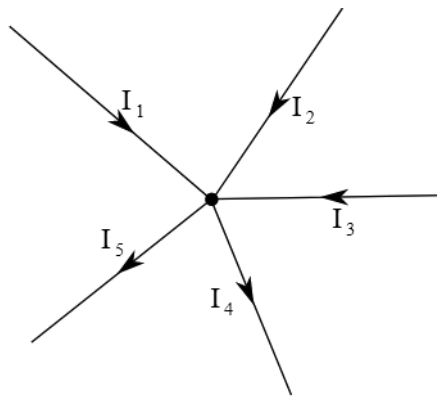
Przy takim połączeniu, efektywna rezystancja **będzie mniejsza niż najmniejszy komponent** w takiej równoległej konfiguracji.

*„**[Georg Simon Ohm]** Sformułował (1827) prawo opisujące związek pomiędzy natężeniem prądu elektrycznego a napięciem elektrycznym, znane współcześnie jako prawo Ohma. Badał nagrzewanie się przewodników przy przepływie prądu elektrycznego. Znalazł zależność oporu od formy geometrycznej przewodnika" [W]*

Prawo Kirchhoffa

Teoria i praktyka

[W] Jedną z podstawowych zasad przepływu prądu jest **pierwsze prawo Kirchhoffa**, które mówi, że dla węzła obwodu elektrycznego **suma algebraiczna natężeń prądów wpływających i wypływających jest równa zeru**. Suma natężeń prądów wpływających do węzła jest równa sumie natężeń prądów wypływających z tego węzła. Prawo to wynika z zasady **zachowania ładunku**.

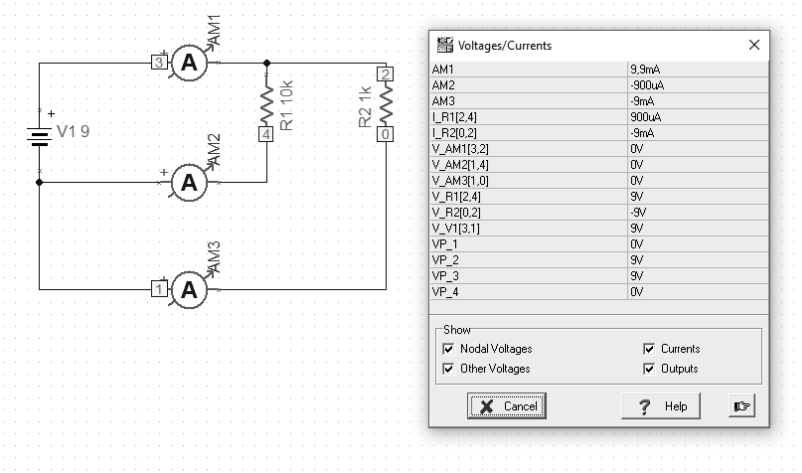


Rys. Pierwsze prawo Kirchhoffa

Drugie prawo Kirchhoffa mówi, że **suma spadków napięcia w obwodzie zamkniętym jest równa zeru**, przy założeniu, że spadek napięcia jest jego ujemnym przyrostem.

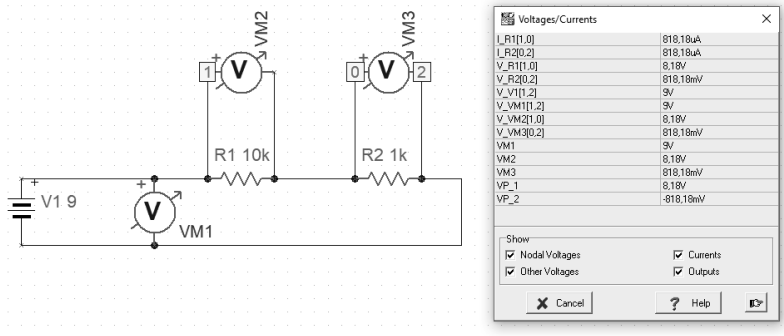
[202] Dla celów zobrazowania prawa Kirchhoffa przygotowałem poniższy prosty przykład. Zaczynamy od przykładu dotyczącego **pierwszego prawa**. Wpięte są 2 rezystory, odpowiednio 10K i 1K. Kolejno wpinamy 3 amperomierze. **Suma**

prądów z obwodów kolejnych oporników daje wartość prądu dla całego obwodu. Możemy ten prąd mierzyć w punkcie, gdzie jest 1 wejście i 2 wyjścia.



Rys. Pierwsze prawo Kirchhoffa (p04)

Kolejny przykład obrazuje **drugie prawo Kirchhoffa**. Przykład pokazuje różnicę w sposobie podłączania elementów obwodu. Podłączając równolegle uzyskujemy te same wartości napięcia. Podłączając szeregowo, **suma spadków napięć na poszczególnych elementach będzie równa napięciu zasilającemu**.



Rys. Drugie prawo Kirchhoffa (p05)

*„**Gustav Robert Kirchhoff** (ur. 12 marca 1824 w Królewcu, zm. 17 października 1887 w Berlinie) — niemiecki fizyk, twórca prawa promieniowania cieplnego dotyczącego zależności między zdolnością emisyjną i absorpcyjną, oraz praw dotyczących obwodów elektrycznych (pierwsze i drugie prawo Kirchhoffa)” [W]*

Dzielnik napięcia

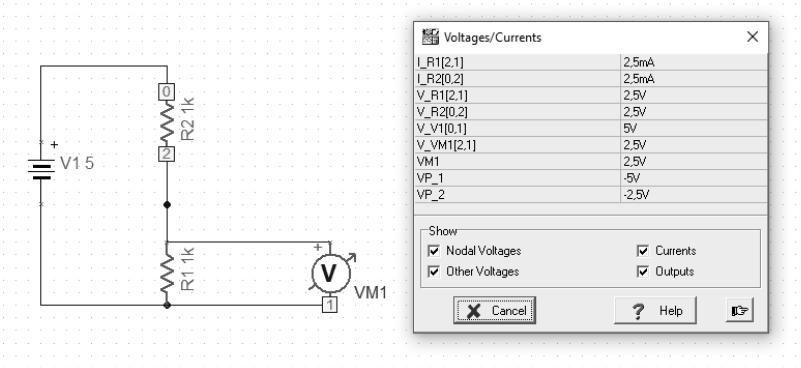
Pomniejszanie wartości napięcia

[202] Następnym przykładem jaki chce praktycznie zaprezentować jest popularny dzielnik napięcia. Umożliwia on **pomniejszenie napięcia wyjściowego** do pożądanej wartości dzięki własnościom opisanym nieco wcześniej.

$$U_{wyj} = U_{zas} \cdot \frac{R_1}{R_1 + R_2}$$

Wz. Napięcie wyjściowe w dzielniku

Mając na uwadze powyższy wzór, możemy przyjąć, że chcemy pomniejszyć wartość napięcia wejściowego 5V do oczekiwanego 2,5V na wyjściu. Stosujemy zatem **dwa rezystory o oporze 1K podłączone szeregowo**.

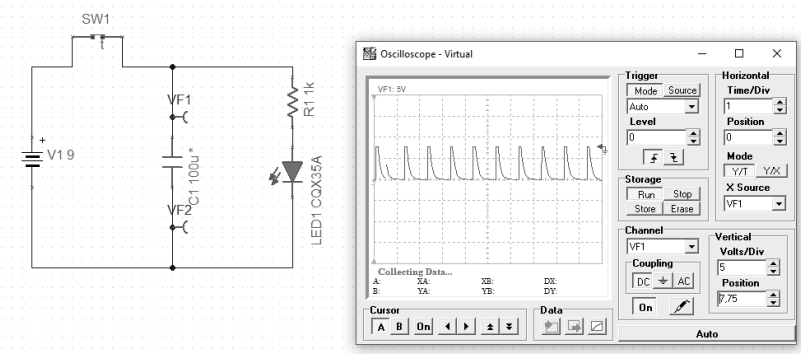


Rys. Dzielnik napięcia (p06)

Kondensator

Filtrowanie sygnału

[202] Kondensatory dzielimy na **spolaryzowane** i **niespolaryzowane** w zależności czy kierunek włączenia ich do obwodu ma znaczenie. Generalnie rzecz ujmując, **kondensatory filtrują przebieg źródła zasilania**. Przyjmuje się, że dłuższe wyprowadzenie z kondensatora to biegun dodatni, co czasami na odzwierciedlenie w oznaczeniu na schemacie. Kondensatory **elektrolityczne** składają się z okładki metalowej i elektrolitowej, przełożone papierem. Kondensatory bezbiegunowe (niespolaryzowane) mogą być zbudowane z elementów ceramicznych lub folii. Występuję również kondensatory tantalowe, które zarówno oferują duże pojemności i mają relatywnie małe straty. **Rozmiar fizyczny kondensatora jest uzależniony nie tylko od jego pojemności ale również maksymalnego napięcia pracy.**



Rys. Kondensator w zastosowaniu z przełącznikiem czasowym i diodą LED (p07)

W powyższym przykładzie do obwodu wstawiamy **przełącznik czasowy**, który w cyklu 1 sekundowym załącza się na 100ms powodując **ładowanie kondensatora**. Po określonym parametrami elementów czasie dioda LED zaczyna emitować światło. Po rozłączeniu ładowania przez przełącznik następuje rozładowywanie kondensatora. Czasem aktywacji diody sterujemy poprzez **określenie pojemności wykorzystanego kondensatora**.

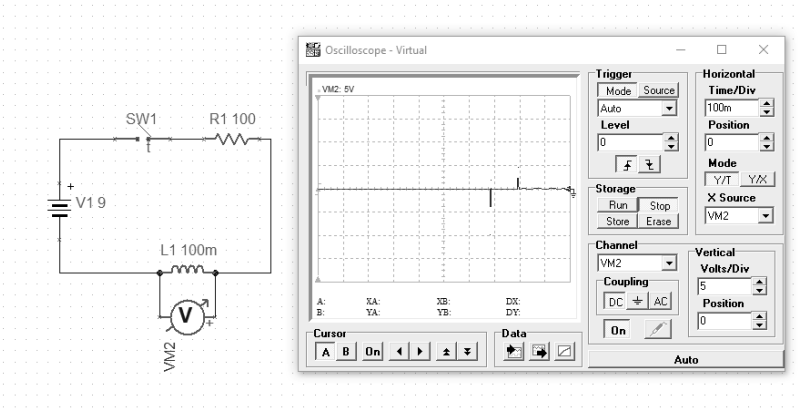
Główne jednak zadanie dla kondensatorów to **filtrowanie zasilania**. Zabieg ten stosuje się aby **ochronić wrażliwe komponenty** np. mikrokontroler w układzie cyfrowym. Najskuteczniejsze jest stosowanie różnych typów dla filtrowania określonych zakłóceń. Kondensatory wpina się w obwód **równolegle**.

> *„Kondensator to urządzenie do czasowego przechowywania ładunku elektrycznego. To, co uważa się za pierwszy kondensator, nazwano słoikiem Leyden, który został wynaleziony przez Pietera van Musschenbroeka w 1746 roku na Uniwersytecie Leyden (lub Leiden) w Holandii. Był to szklany słoik owinięty wewnątrz i na zewnątrz cienką metalową folią. Folia zewnętrzna była połączona z ziemią, a folia wewnętrzna była połączona ze źródłem elektryczności, takim jak generator elektrostatyczny. Chociaż w tamtym czasie nie rozumiano, jak to działa, eksperymentatorzy odkryli, że słoik lejdejski wydawał się przechowywać ładunek elektryczny nawet po odłączeniu go od generatora.” [206]*

Cewka

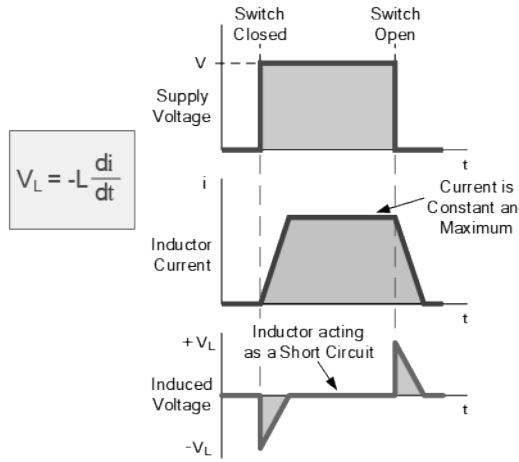
Energia w polu magnetycznym

[209] Tak jak kondensator działa na zasadzie przechowywania energii w postaci pola elektrycznego, tak **induktor przechowuje energię w postaci pola magnetycznego**. Energia przechowywana w cewce jest obliczana w kategoriach prądu. Kondensator działa jako izolator obwodu, podczas gdy **cewka działa jako jego przewodnik**. Cewki charakteryzują się **indukcyjnością** wyrażaną w **henrach** [H]. Cewki wpina się w obwód **szeregowo** wraz z zasilanym urządzeniem. Cewki charakteryzują się prądem maksymalnym, jako, że ich uzwojenia mają pewną rezystancję, przepływ prądu powoduje więc odkładanie się na nich napięcia, co w iloczynie daje wydzielaną przez nich moc. Regulowanie parametrów pracy ma na celu **uniknięcie przegrzaniu cewki**.



Rys. Różnica pomiędzy ciągłym zasilaniem a zasileniem okresowym (p08)

Powyższy przykład (z cyklicznym włącznikiem) ilustruje własność cewki w postaci występowania **wstecznej siły elektromotorycznej**. Jeśli będziemy nieustannie podawać napięcie, wówczas cewka będzie zachowywała się jak kawałek przewodu. Jeśli jednak będą występowały **zmiany** w natężeniu prądu przepływającego przez cewkę, wówczas na pomiarze możemy obserwować **skoki sygnału**.



Rys. Napięcie i prąd w cewce [209]

„Stosowanie transformatorów elektrycznych i cewek indukcyjnych w sferze elektrycznej jest tak powszechną praktyką, że trudno wyobrazić sobie świat bez tych urządzeń. Kiedy właściwość indukcyjności została odkryta w latach 30. XIX wieku przez Josepha Henry'ego i Michaela Faradaya (oddzielnie i na różnych kontynentach), technologia została zrewolucjonizowana. Indukcyjność po raz pierwszy odkrył Faraday w prosty, ale dziwny sposób: owinął papierowy cylinder drutem, przymocował końce drutu do galwanometru (urządzenia służącego do pomiaru prądu elektrycznego) i przesuwał magnes do i z cylindra. Galwanometr zareagował na to, ujawniając wytwarzanie niewielkiego prądu." [201]

Półprzewodniki

Podstawa dla budowy diod i tranzystorów

Półprzewodnikiem nazywamy materiały, których **przewodnictwo elektryczne mieści się pomiędzy przewodnikami a izolatorami**. Ich rezystancja i przewodnictwo jest uzależnione od temperatury i domieszek. [W] Najczęściej komercyjnie stosowanymi półprzewodnikami są te oparte na **krzemie**, w przeszłości również na **germanie**. Poza tymi dwoma pierwiastkami (w formie krystalicznej) stosowane jest całe spektrum innych **substancji z grup 13 — 15**, które mogą formować związki dwu, trzy lub cztero elementowe.

Złącze p-n

Podstawowy element systemu półprzewodnikowego

[003] Aby uzyskać zwiększoną przewodność półprzewodników stosuje się **dodatki atomów innych pierwiastków**. Przykładowo *arsen* lub *antymon* powodują pojawienie się swobodnych elektronów, będących źródłem ładunków ujemnych. Półprzewodniki te należą do typu **n** (*negative*). Z kolei *ind* i *bar* **powodują niedomiar elektronów**, gdyż ich atomy zabierają z otoczenia elektrony atomom krzemu, powodując **powstawanie ładunków dodatnich**. Są to półprzewodniki typu **p** (*positive*). Półprzewodniki typu **n** i typu **p** złączone ze sobą na drodze specjalnego zabiegu technologicznego, tworzą zasadniczą część każdego elementu półprzewodnikowego, zwaną warstwą **p-n**. Na granicy tych półprzewodników powstaje złącze **p-n**, przewodzące prąd od **p** do **n**.

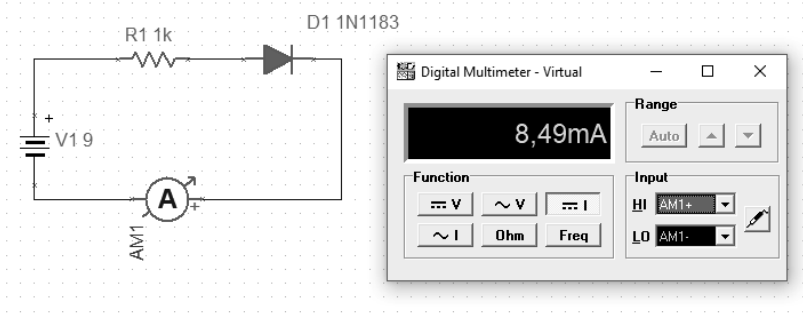
Dioda półprzewodnikowa

Komponent budowy prostowników i elementów sygnalizacyjnych

[003] Komponent **dwuelementowy** (*di* od dwa), przeznaczony na przykład do budowy **prostowników**, **zabezpieczeń** lub w przypadku diod LED również jako element **sygnalizujący**, oświetleniowy. Jest to złącze **p-n** o oporze w **kierunku przewodzenia** ok. 10 000 razy mniejszym niż w **kierunku zaporowym**.

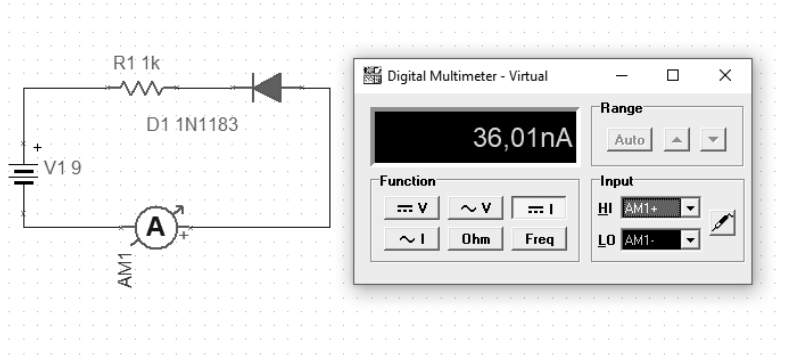
[101] Za pomocą diod można **odtworzyć amplitudę sygnału**, **przesunąć sygnał** o składową stałą lub **zwielokrotnić napięcie**.

[202] **Dioda krzemowa** podłączana powinna być zgodnie ze strzałką symbolu, jeśli chcemy aby **przewodziła prąd**. Ustawienie jej odwrotnie to **stan zaporowy**, w którym do określonego napięcia nie powinno być przewodzenia. Jeśli przekroczymy parametry projektowanej pracy takie diody, wówczas zostanie uszkodzona i może przewodzić **prąd w obu kierunkach**.



Rys. Dioda LED w kierunku przewodzenia (p09)

Jeśli diodę podłączymy w **kierunku zaporowym** mierzony prąd powinien być **znikomy**. W poniższym przykładzie prąd wynosi około 36 nA.



Rys. Dioda LED podłączone zaporowo (p10)

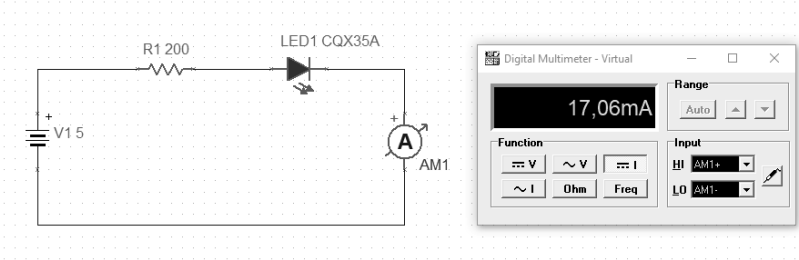
Spadek napięcia na diodach (tutaj LED) zależy od stosowanej substancji, które określa danych kolor luminescencji. Spadki te występują przeważnie od **1.7 do 3.6 V**. Diody posiadają określone parametry pracy i należy podając napięcie do układu zadbać o **właściwą wartość prądu, który ograniczamy szeregowym rezystorem**. Kolejność połączenia w połączeniu szeregowym w tym wypadku nie ma znaczenia, zatem rezystor może być i przed i po diodzie.

$$R = \frac{U_{zas} - U_{diody}}{I_{diody}}$$

(najczęściej 0,01–0,02 A)

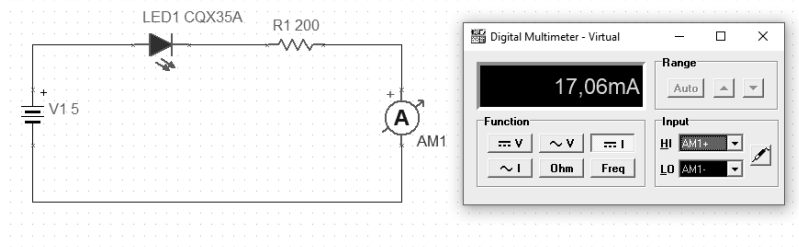
Wz. Wymagana rezystancja opornika do zasilenia diody

Aby zobrazować, że **kolejność połączenia rezystora i diody nie ma znaczenia**, poniżej znajdują się dwa zbliżone przykłady to ilustrujące.



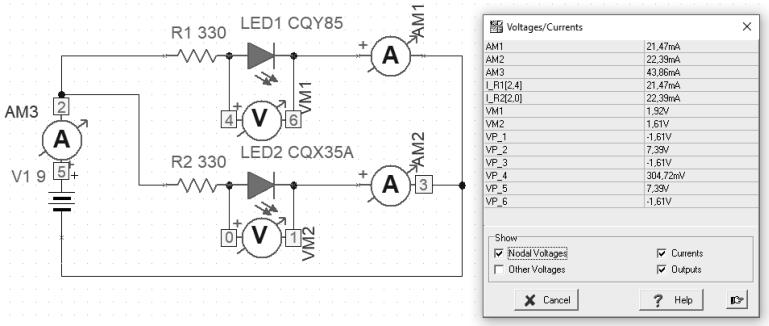
Rys. Rezystor „przed" diodą (p11)

W obu przykładach uzyskujemy taki sam rezultat.



Rys. Rezystor „po" diodzie (p12)

Prąd możemy tłumaczyć jako **analogia ciśnienia wody** w układzie zamkniętym, zatem kolejność nie będzie miała szczególnego znaczenia przy podłączaniu szeregowym elementów.



Rys. Równoległe podłączenie szeregów dioda + rezystor (p13)

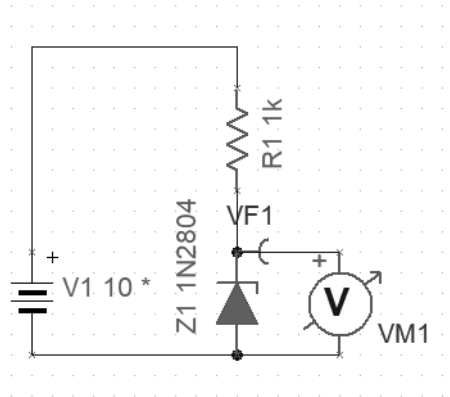
Diody możemy podłączyć w szeregi jak powyżej. Istotnymi parametrami pracy są **napięcie przewodzenia, standardowy i maksymalny prąd oraz parametry wsteczne**. Znając te parametry możemy dobrać odpowiednio oporniki i ułożyć całość w obwód jak powyżej.

„Niemiecki fizyk Ferdinand Braun, 24-letni absolwent Uniwersytetu w Berlinie, studiował charakterystykę elektrolitów i kryształów przewodzących prąd na Uniwersytecie w Würzburgu. W 1874 r., badając kryształy galeny (siarczek ołowiu) końcówką cienkiego metalowego drutu, zauważył, że prąd płynie swobodnie tylko w jednym kierunku. Braun odkrył efekt prostowania elektrycznego, który zachodzi w miejscu styku metali z niektórymi materiałami krystalicznymi. Braun zademonstrował to urządzenie półprzewodnikowe w Lipsku w 1876 roku, ale nie znalazło ono użytecznego zastosowania aż do pojawienia się radia na początku XX wieku. " [207]

Dioda Zenera

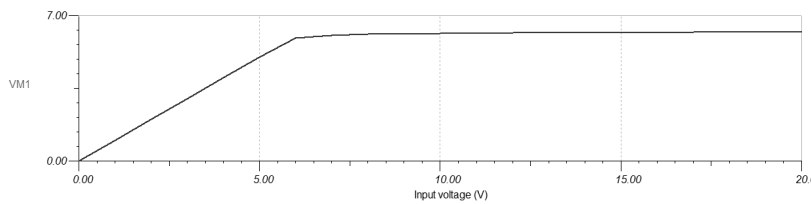
Źródło napięcia odniesienia

[101] Diody Zenera wykorzystywane są najczęściej jako **źródło napięcia odniesienia o niemal stałej wartości**. Jeśli napięcie wejściowe jest mniejsze od **napięcia Zenera** dioda nie przewodzi i nie stabilizuje napięcia.



Rys. Diodę Zenera podłączamy zaporowo (p14)

[209] Dioda Zenera zawsze jest używana jako stabilizator w warunkach polaryzacji zaporowej. Można zaprojektować **układ regulatora napięcia stałego z diodą Zenera**, aby zapewnić **stałą** wartość napięcia na obciążeniu, niezależnie od **zmian napięcia wejściowego** lub **prądu obciążenia**.



Rys. Pomiar napięcia przebicia

Jak widać na powyższym wykresie rzeczywiście dla modelu **1N2804** diody Zenera wartość napięcia przebicia wynosi 6.8V, zgodnie z notą katalogową.

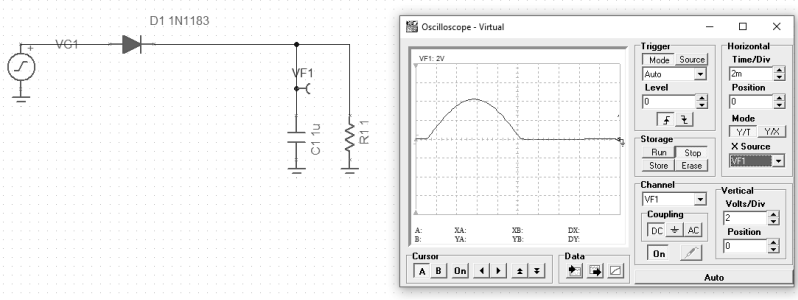
Filtrowanie sygnału

Regulowanie kształtu i przebiegu sygnałów

Filtr jednopołówkowy

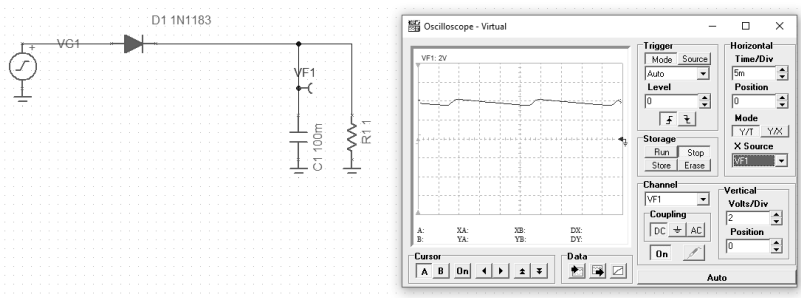
Regulowanie jednej części osi przebiegu sygnału

[101] Ciekawym przykładem, bo pokazującym aspekt przemienności oraz zastosowania podstawowych komponentów pasywnych, jest poniższy obwód. Jest to *de facto* **prostownik jednopołówkowy**.



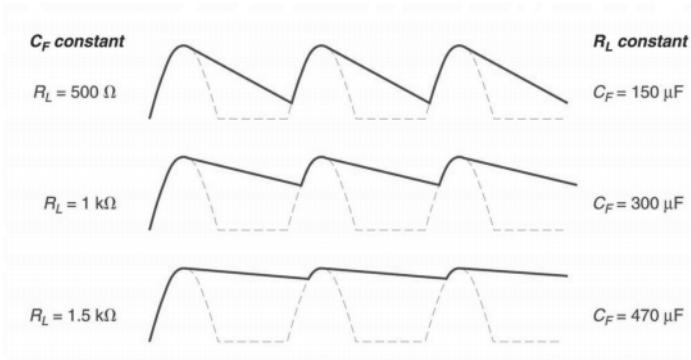
Rys. Filtrowanie jednej połówki przebiegu sygnału (p15)

Mamy tutaj możliwość **„*sterowania*" przebiegiem**, który pozostał dla nas dostępny, dzięki zastosowano **diody**. Zmieniając **parametry kondensatora i rezystora** wpływamy na pozostały sygnał. Możemy tym układem **wykrywać amplitudę sygnału**, gdzie fala źródłowa ma wielokrotnie większą częstotliwość niż przenoszony docelowy sygnał (np. radio z sygnałem rzędu MHz a sygnał przenoszony to < 20 kHz).



Rys. Wpływ zwiększenia pojemności kondensatora na przebieg sygnału

Stosowanie kondensatora to **wygładzanie przebiegu napięcia**. Występuje tutaj właściwość w postaci **czasu ładowania i rozładowania**. Poprawę (do pewnych granic) filtracji uzyskujemy dzięki **zwiększeniu pojemności kondensatora**.



Rys. Parametryzacja filtrowania sygnału

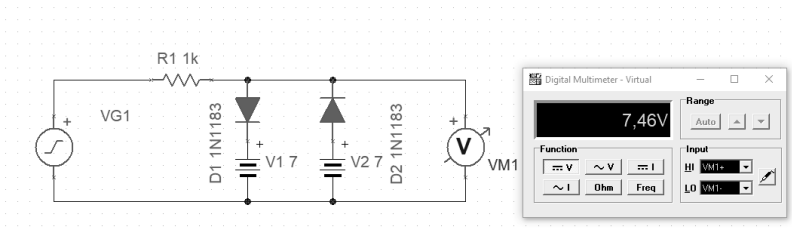
Włączenie do układu rezystora ogranicza tak zwany udar prądowy występujący przy załączeniu zasilania, gdzie w początkowej fazie prąd jest ograniczony jedynie znikomymi

rezystancjami diody. Obecność kondensatora wpływa na wartość **prądu wstecznego diody**. Filtrować sygnał można również za pomocą **filtra indukcyjnego, czyli cewki**. Najczęściej stosuje się **kombinacje** wszystkich tych elementów, tak aby ograniczyć różne kategorie sygnałów niepożądanych dla układu.

Równoległe diody

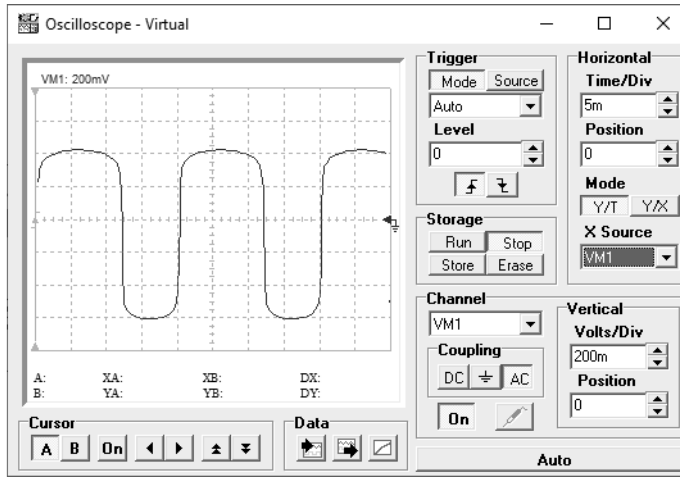
Filtrowanie obu pół-fal napięcia

[101] Można wykorzystać równoległe połączenie diod ustawionych **przeciwstawnie** do uzyskania **efektu ograniczania obu pół-fal napięcia**.



Rys. Ograniczanie obu pół-fal napięcia (p16)

Zadanym **sygnałem wejściowym była sinusoida** o amplitudzie 10V i częstotliwości 50Hz. Przez zastosowanie równoległych diod, można **ograniczać sygnał** jak również **kształtować go w pożądany sposób**.

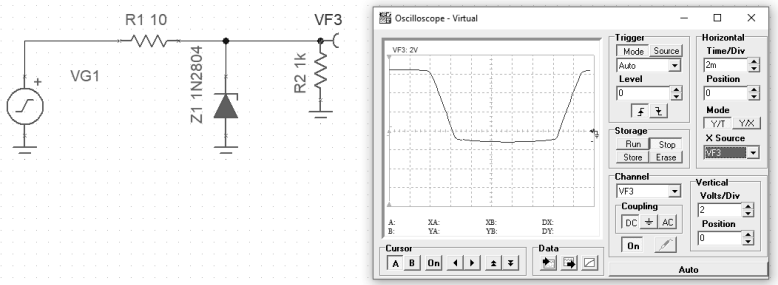


Rys. Wynikowy sygnał ograniczony równoległymi diodami

Filtrowanie diodą Zenera

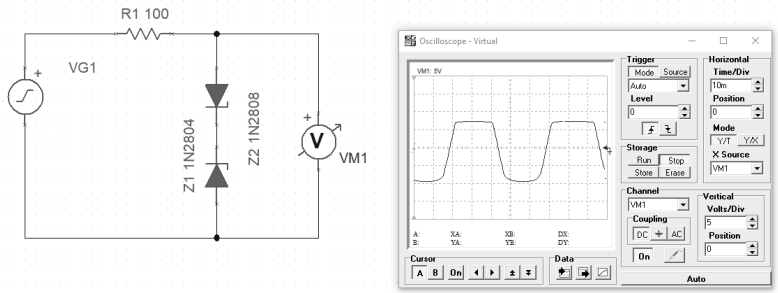
Selektywne kształtowanie sygnału

[101] Pojedyncza dioda tego rodzaju **ograniczy sygnał do 0.7V w jednym kierunku a w przeciwnym do napięcia Zenera**. Diody Zenera można łączyć szeregowo, jedna z nich będzie spolaryzowana w kierunku przewodzenia ograniczając napięcie symetrycznie do wartości Vz+0.7V.



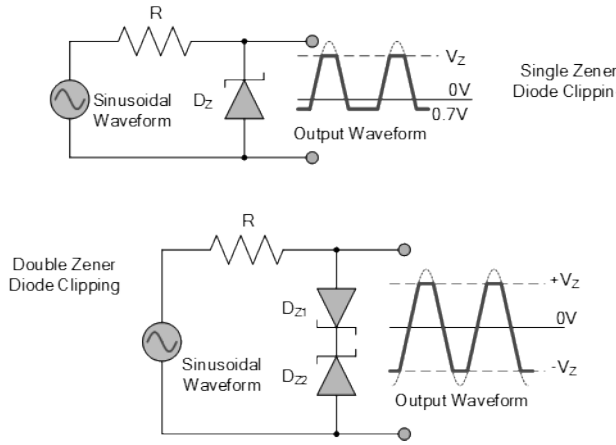
Rys. Filtrowanie diodą Zenera (p17)

Przy łączeniu diod Zenera przeciwstawnie uzyskujemy wynik jak poniżej.



Rys. Diody Zenera zestawione szeregowo (p18)

Różnica w stosowaniu jednej lub dwóch diod Zenera to **wybitność sygnału**.

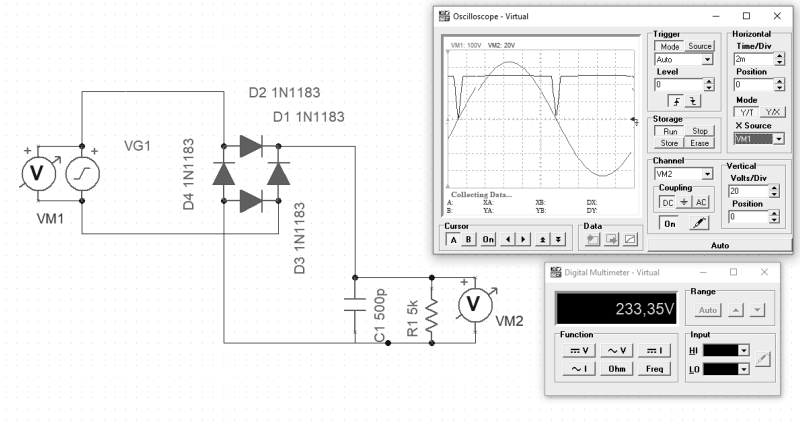


Rys. Różnice w stosowaniu jednej lub dwóch diod Zenera

Mostek Graetz'a

Prostownik mostkowy

[101] Jest to **prostownik mostkowy**. Nie wymaga użycia transformatora. **Naprzemiennie pracują pary diod**. Wytwarza napięcie **dwupulsowe**. Rozwiązanie **polskiego wynalazcy Karola Pollaka**, który jest również wynalazcą kondensatora elektrolitycznego.



Rys. Mostek Graetz'a (p19)

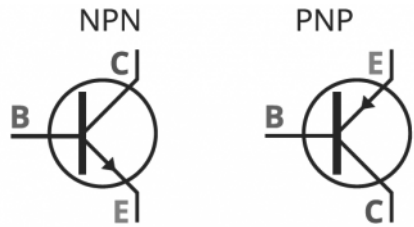
Aby zobrazować jego **pulsacyjną naturę** dobrałem odpowiednie wartości użytego kondensatora i rezystora. Dzięki temu na wynikowym sygnale jest to bardzo wyraźnie widoczne.

> TRANZYSTOR

Sterowalny element mogący pełnić rolę przełącznika lub wzmacniacza. Podstawa budowy wszelkich układów logicznych. Bez tranzystora nie byłoby cyfrowej rewolucji i komputerów jakie znamy dzisiaj.

[101] Tranzystor jest elementem **sterowanym — sygnałem małej mocy** (prąd bazy/napięcie bramki) **sterujemy** przepływem sygnału o **większej mocy** (prąd kolektora/drenu).

[202] Tranzystor to element **półprzewodnikowy**, który steruje przepływem prądu lub wzmacnia go. Tranzystory **bipolarne** składają się z trzech **warstw** półprzewodnika. Wyprowadzenia tranzystora to emiter (E), baza (B) oraz kolektor (C). Wyróżniamy dwa typy tranzystorów bipolarnych, tj. **NPN** i **PNP**. Tranzystor typu NPN przewodzi, jeśli do bazy podamy wysoki potencjał względem emitera. Tranzystor typu PNP przewodzi, jeśli do bazy podamy niski potencjał względem emitera.



Rys. Typy tranzystorów bipolarnych NPN i PNP [202]

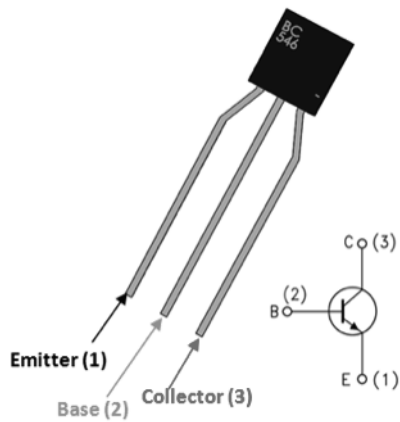
[003] Wymienione typy różnią się tylko **przeciwnymi kierunkami prądów** dopływających do elektrod. Jeśli w tranzystorze **p-n-p** między emiter i bazę lub kolektor i emiter włączy się źródło prądu stałego o biegunie minus, na kolektorze popłynie prąd zerowy tranzystora, który zależy od wartości przyłożonego napięcia. Po przekroczeniu dopuszczalnej wartości napięcia prąd gwałtownie wzrasta, powodując zniszczenie tranzystora.

„Pierwszy tranzystor został pomyślnie zademonstrowany 23 grudnia 1947 roku w Bell Laboratories w Murray Hill w stanie New Jersey. Bell Labs jest oddziałem badawczym American Telephone and Telegraph (AT&T). Trzema osobami, którym przypisuje się wynalezienie tranzystora, byli William Shockley, John Bardeen i Walter Brattain. Wprowadzenie tranzystora jest często uważane za jeden z najważniejszych wynalazków w historii. " [208]

Przykładowy tranzystor NPN

Bipolarny tranzystor, sterowany prądowo

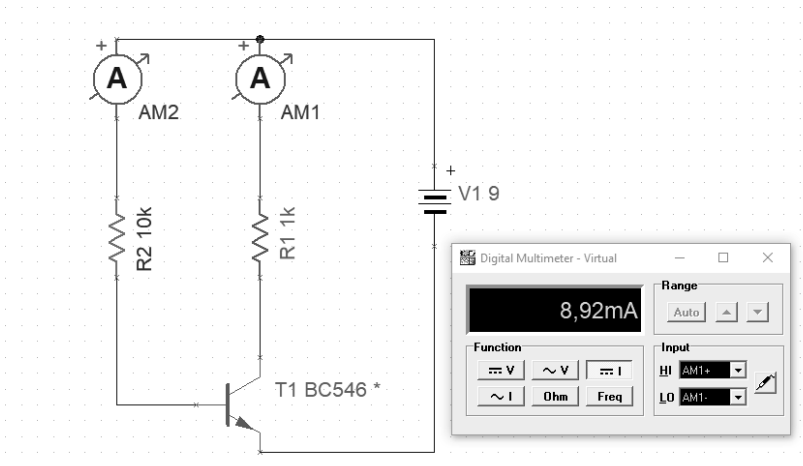
[004] Tranzystor to połączenie słów **transfer** i **rezystor, jest to trioda półprzewodnikowa, trójelektrodowy przyrząd półprzewodnikowy o własnościach wzmacniających**. Poniżej prezentuję pierwszy przykład zastosowania tranzystora typu NPN, konkretnie modelu BC546. Maksymalny ciągły prąd kolektora (Ic) wynosi 100mA.



Rys. Tranzystor BC546 typu NPN

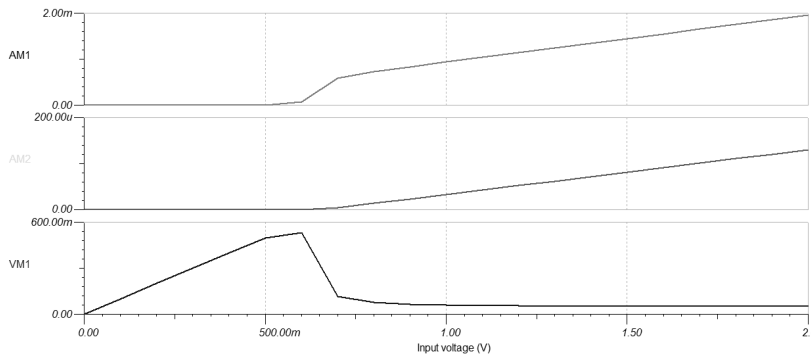
Tranzystora można używać m.in. jako **przełącznik**. Podając niewielki prąd na bazę, uzyskujemy wzmocnienie na złączu pomiędzy kolektorem a emiterem. Prąd bazy jak i kolektora ograniczamy za pomocą odpowiednio dobranych rezystorów. Użycie limitowania o niewłaściwych parametrach zniszczy tranzystor. **Relacja pomiędzy prądem bazy a kolektorem**

to współczynnik beta, który dla tego konkretnego modelu wynosi od 110 do 800 jednostek (hfe). Maksymalne napięcie pomiędzy kolektorem a emiterem (Vceo) to 65V.



Rys. Prądy bazy i kolektora (p20)

W powyższym układzie przy prądzie bazy wynoszącym poniżej 1mA prąd, którym możemy sterować wynosi prawie 9mA. Jeśli usuniemy rezystor R1, wówczas nieograniczany prąd wyniesie ponad 100 mA. Jeśli jednak chcemy przykładowo sterować takim tranzystorem, musimy brać pod uwagę **na jakich prądach maksymalnych mogą pracować sterowane komponenty**, np. dioda LED.



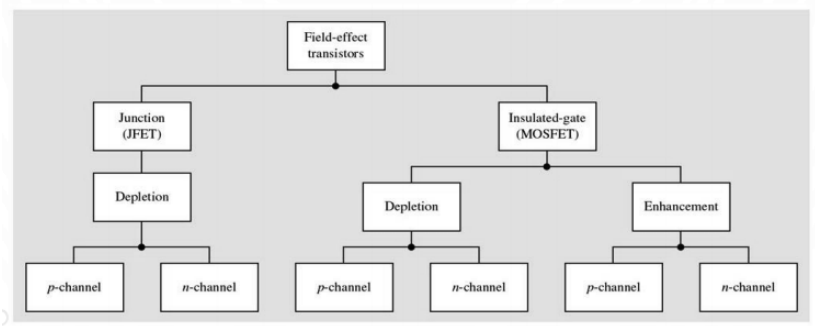
Rys. Analiza wejścia w przedziale 0—2V

Badamy ten układ napięciowo (zakładka *Analysis*), na wejściu w przedziale 0—2V. Daje to nam 3 komponenty wyjściowe, dwa amperomierze i jeden woltomierz (pomiędzy stykami kolektora i emitera). Podanie napięcia na poziomie 700mA powoduje pojawienie się prądu w gałęzi bazy. Wraz z tym prądem pojawia się **efekt wzmocnienia, zwany współczynnikiem beta**.

Tranzystory FET

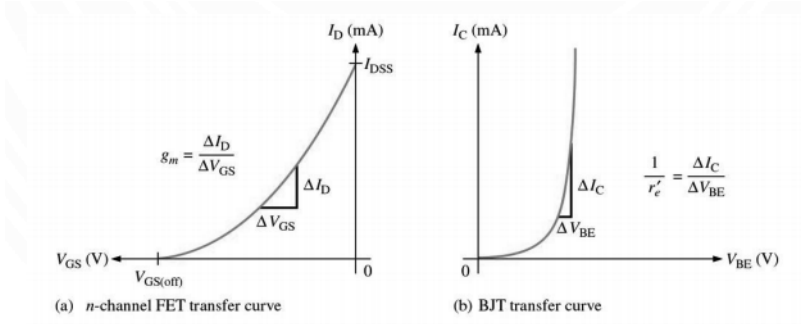
Typ tranzystora, opierający się na efekcie pola, sterowany napięciowo.

[101] Ten typ tranzystorów opiera się na działaniu **efektu pola. Są to tranzystory polowe. Wykorzystują one oddziaływanie pola elektrycznego na rezystancję półprzewodnika**. Dwa rodzaje tranzystorów FET to JFET oraz MOSFET.



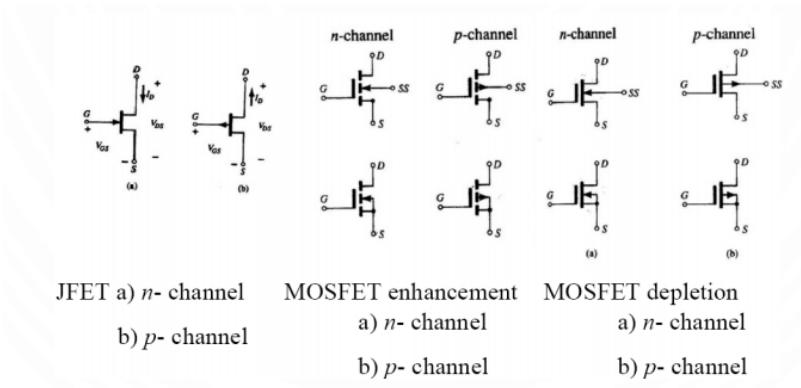
Rys. Kategorie tranzystorów polowych [101]

Sterowanie prądem w tranzystorach JFET odbywa się za pomocą **przyłożonego napięcia**. JFET jest przyrządem **normalnie załączonym**, bez podania napięcia na bramkę. Pomiędzy złączami **występują pojemności**.



Rys. Różnica w krzywych transferu (BJT vs FET)

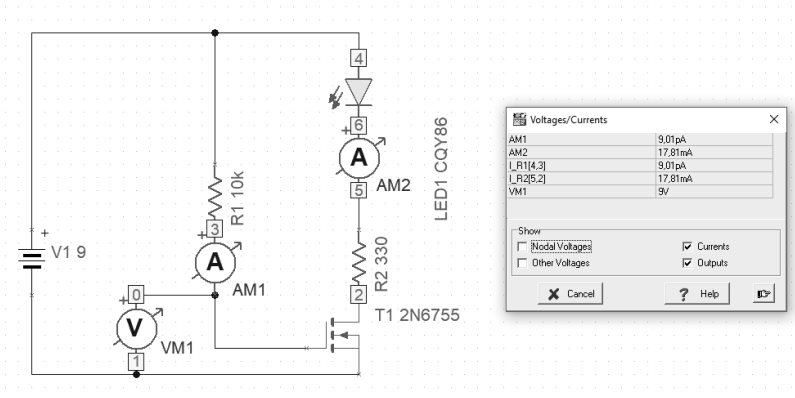
Ogólnie tranzystory typu FET są **mniejsze i szybsze** niż BJT. Są jednak bardziej **wrażliwe** na wyładowania elektrostatyczne.



Rys. Symbole tranzystorów FET

Dzięki temu, że tranzystory te sterowane są **napięciem a nie prądem**, kontrolowanie przepływu dużych prądów jest **ułatwione** i bardziej **efektywne**. Zatem jeśli pracujemy z du-

żymi prądami, w pierwszej kolejności możemy spojrzeć w kierunku tych właśnie tranzystorów.



Rys. Tranzystor MOS n-kanałowy, model 2N6755 (p21)

Jak widać na powyższym przykładzie **prąd na bramce wynosi zaledwie 9pA**, w momencie, gdy sterujemy przepływem prądu około 18mA w gałęzi z diodą LED.

Sygnały i układ cyfrowe

Układy operujące na wartościach dyskretnych

[101] Elektronika cyfrowa posługuje się sygnałami przyjmującymi **jeden z dwóch dyskretnych stanów** (niski i wysoki). **Stan niski** to przeważnie 0V w odniesieniu do analogowej wartości, natomiast **stan wysoki** przeważnie w przedziale od -5 do +5V. Można ze sobą łączyć układy wykonane w tej samej technologii, np. TTL do TTL lub CMOS do CMOS, ale przy łączeniu układów z różnych technologii należy **porównywać poziomy napięć**. Większość komponentów będzie miała wskazane poziomy wysoki i niski **wyrażone w napięciach** zarówno dla wejścia jak i wyjścia w stosunku do określonej temperatury.

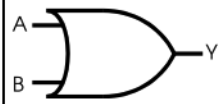
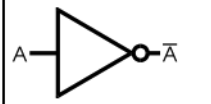
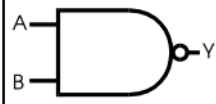
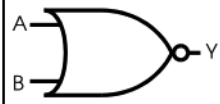
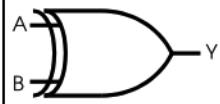
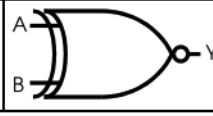
Układy cyfrowe dzielimy na **kombinacyjne, sekwencyjne**. Te sekwencyjne dzielimy dalej na **synchroniczne** oraz **asynchroniczne**. Synchroniczność oznacza **zależność stanu wyjść od zewnętrznego** zegara. Sama sekwencyjność oznacza, że stan wyjścia jest zależny zarówno od wejść w danym momencie, ale również od **wejść w poprzednich przejściach**. Są również układy kombinacyjne, których wyjścia zależą wyłącznie od wejścia w danych chwili.

Bramki logiczne

Scalone układy realizujące funkcje logiczne

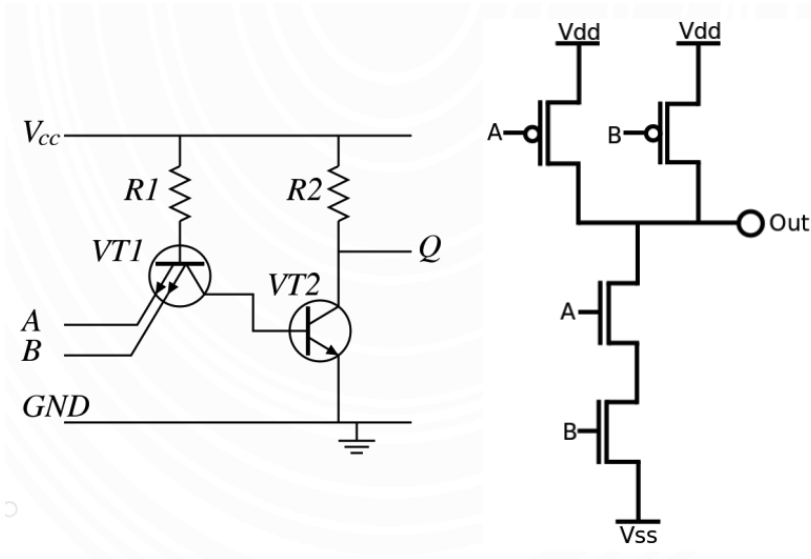
[W] Podstawą logiki cyfrowej, poza zastosowaniem **tranzystora** jest operowanie na **logice Boole'a**. Ta postać algebry została opracowana przez angielskiego matematyka George'a Boole'a w 1847 roku, w publikacji pod tytułem „*Matematyczna analiza logiki*".

Tablica 3.1 Podstawowe bramki logiczne

FUNKCJA LOGICZNA	SYMBOL LOGICZNY	WYRAŻENIE ALGEBRAICZNE	TRUTH TABLE		
			Inputs		Output
			A	B	Y
AND		$A \cdot B = Y$	0 0 1 1	0 1 0 1	0 0 0 1
OR		$A + B = Y$	0 0 1 1	0 1 0 1	0 1 1 1
NOT *(Inverter)*		$A = \overline{A}$	0 1		1 0
NAND		$\overline{A \cdot B} = Y$	0 0 1 1	0 1 0 1	1 1 1 0
NOR		$\overline{A + B} = Y$	0 0 1 1	0 1 0 1	1 0 0 0
XOR		$A \oplus B = Y$	0 0 1 1	0 1 0 1	0 1 1 0
XNOR		$\overline{A \oplus B} = Y$	0 0 1 1	0 1 0 1	1 0 0 1

Rys. Podstawowe bramki logiczne [102]

Logika Boole'a jest to pewien typ **struktury algebraicznej** stosowanej w matematyce, informatyce oraz elektronice cyfrowej, którą tutaj właśnie staram się w podstawowy sposób opisać.

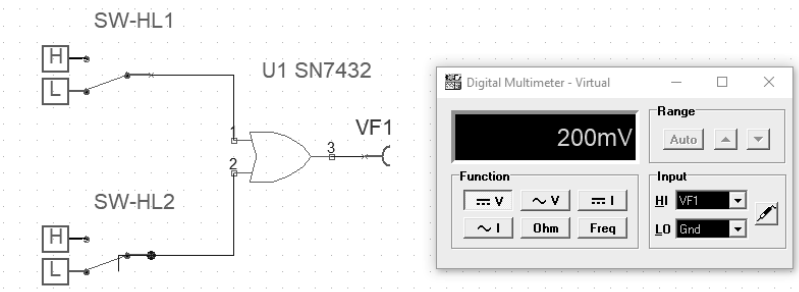


Rys. Implementacja bramki NAND w TTL i CMOS

Suma logiczna OR

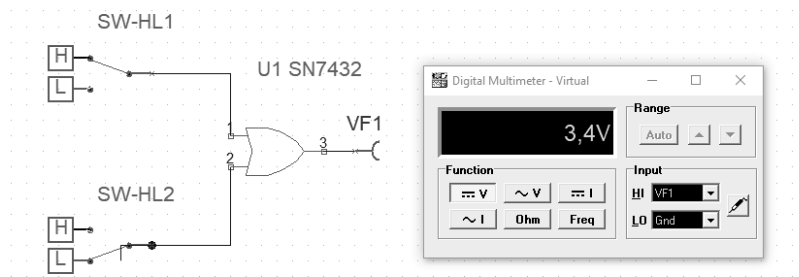
Układ logiczny

Jak już wcześniej wspomniałem, wejście i wyjścia mają przez producenta **określone poziomy napięciowe**, które powinny być odpowiednio **interpretowane** (tj. uwzględnione podczas projektowania) przez pożądany obwód. Prezentuje to poniższy przykład **sumujący**:



Rys. Poziom niski to 200mV (p22)

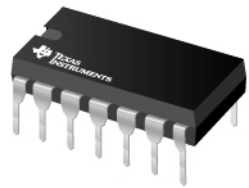
W powyższym przykładzie mamy wykorzystaną **gotową bramkę sumy logicznej** (OR), która da wynik 1 jeśli którykolwiek z sygnałów wejściowych będzie równy 1.



Rys. Poziom wysoki to 3,4V

Dla użytego komponentu stanem wysokim będzie napięcie o wartości 3,4V. Tym komponentem jest układ **SN7432,**

który posiada 4 bramki sumy logicznej możliwe do wykorzystania.



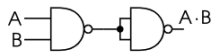
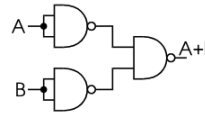
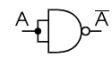
Rys. Układ SN7432

Bramka NAND

Układ logiczny

[102] Bramki AND, OR, NOT tworzą **funkcjonalnie pełny zestaw elementów**. Oznacza to, że można z nich zbudować **dowolnie złożony układ logiczny**. Za pomocą wyłącznie bramek NAND albo wyłącznie bramek NOR można także zrealizować dowolnie złożoną funkcję, w tym również podstawowe funkcje, np. AND, OR lub NOT. Najszerszą ofertę producentów układów cyfrowych stanowią bramki **NAND**, gdyż są to **bramki uniwersalne**.

Tablica 4.2 Funkcje logiczne **AND**, **OR**, **NOT** w realizacji układowej z bramek **NAND**

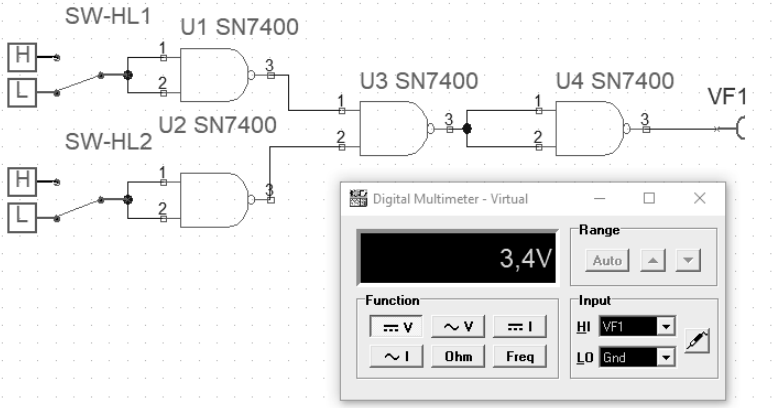
AND	**OR**	**NOT** *(inverter)*
		
$\overline{\overline{A \cdot B}} = A \cdot B$	$\overline{\overline{A} \cdot \overline{B}} = \overline{\overline{A + B}} = A + B$	$\overline{A \cdot A} = \overline{A}$

Rys. Funkcje podstawowe w realizacji z bramek NAND [102]

Przekształcenie

Realizacji funkcji bramki NOR za pomocą bramek NAND

Za pomocą bramki, a konkretnie wielu bramek **NAND** możemy zrealizować układ funkcjonujący jako bramka **NOR**. Jak już wcześniej zostało nadmienione, bramki łączące funkcje NOT oraz OR lub AND mają bardzo szerokie zastosowanie w implementowaniu pozostałych bramek czy nawet całych układów.

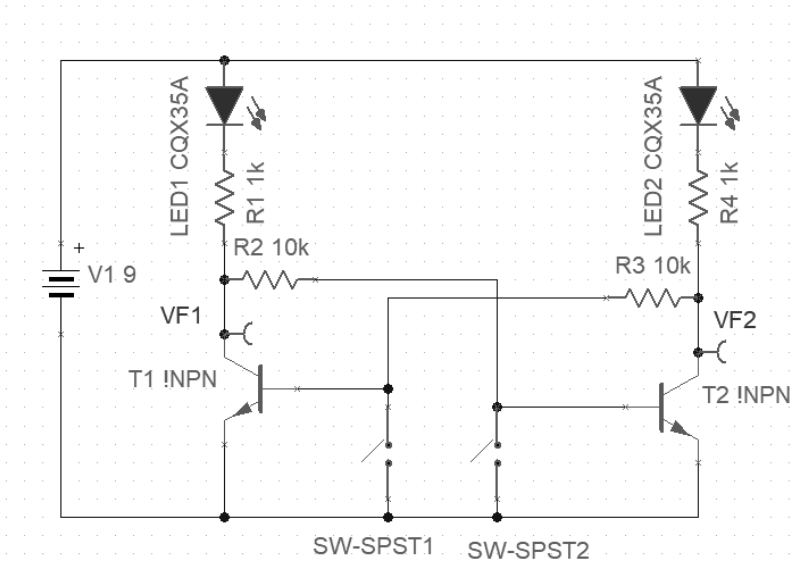


Rys. Bramka NOR zaimplementowana za pomocą bramek NAND (p23)

Przerzutnik bistabilny

Układ typu flip-flop, zapamiętujący stan

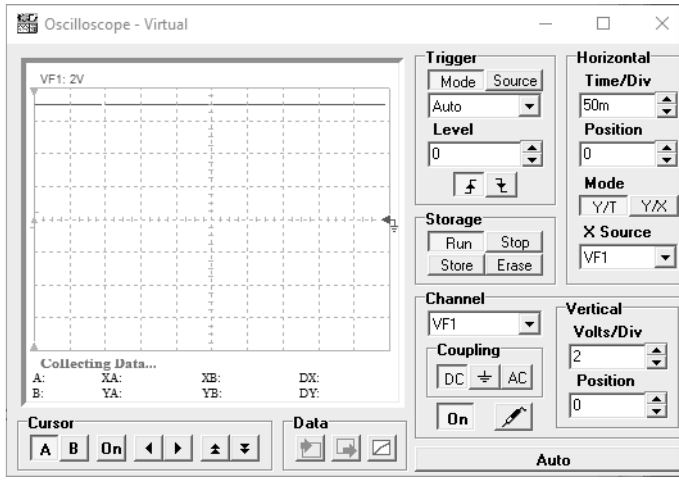
[202] Zapewne pierwszym przykładem jaki można wskazać dla układu cechującego się jakimkolwiek **zapamiętywaniem stanu** działania będzie **przerzutnik bistabilny**. Charakteryzuje się on zastosowaniem dwóch tranzystorów bipolarnych (BJT) typu NPM oraz dwóch diod LED. Mamy także 4 rezystory oraz 2 przełączniki.



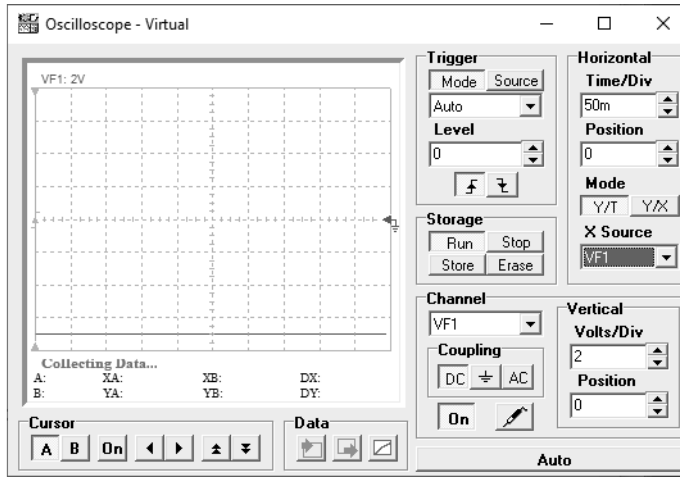
Rys. Przerzutnik bistabilny (p24)

Zasadę działania tego układu możemy obserwować za pomocą oscyloskopu lub składając samodzielnie ten układ w rzeczywistości. Przełącznik nr 1 (po lewej) zamykamy i otwieramy. W tym momencie załączamy do świecenia diodę po prawej

stronie. **„*Przerzucamy*" sygnał na drugą stronę**. Zamykając i otwierając przełącznik po prawej stronie „*przerzucamy*" z kolei sygnał z prawej z powrotem na lewą stronę.



Rys. Domknięcie i otwarcie lewego przełącznika



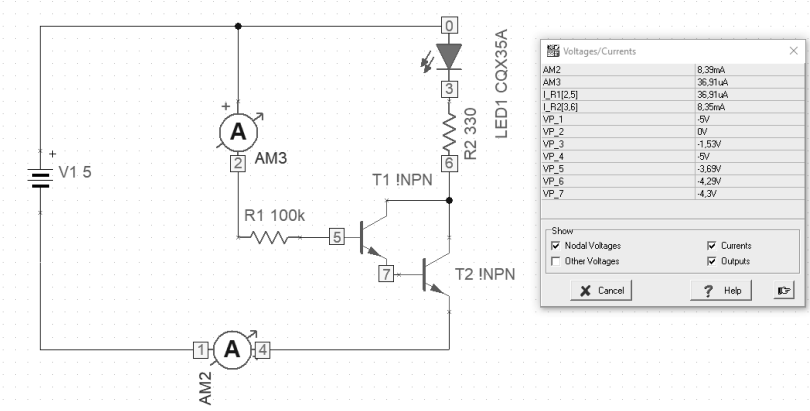
Rys. Domknięcie i otwarcie prawego przełącznika

Jak widać na powyższych ekranach przełączanie powoduje zmianę stanu z wysokiego na niski przy kolejnych przełączeniach. Jeśli po załączeniu symulacji zamkniemy lewy przełącznik (zaświeci prawa dioda) to po zamknięciu prawego przełącznika gaśnie prawa dioda. Po takim zabiegu otwieranie po którejś ze stron przełącznika powoduje **zapalanie się diody po danej stronie**.

Układ Darlingtona

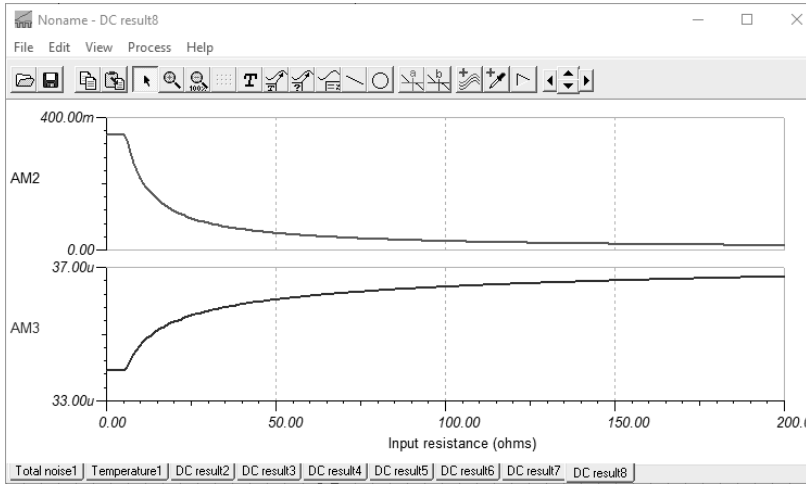
Tranzystorowy układ w efektem wzmocnienia prądowego

Aby zapewnić w układzie wystarczający prąd, niezbędny do zasilenia podłączonych urządzeń, np. silnika lub żarówki możemy rozważyć zastosowanie układu Darlingtona. Jest to **para tranzystorów bipolarnych**, w tym przypadku NPN. Pomiędzy oboma tranzystorami występuje **wzmocnienie prądowe**, które będzie dużo większe niż zastosowanie pojedynczego tranzystora. Wystąpi jednak większy **spadek napięcia**.



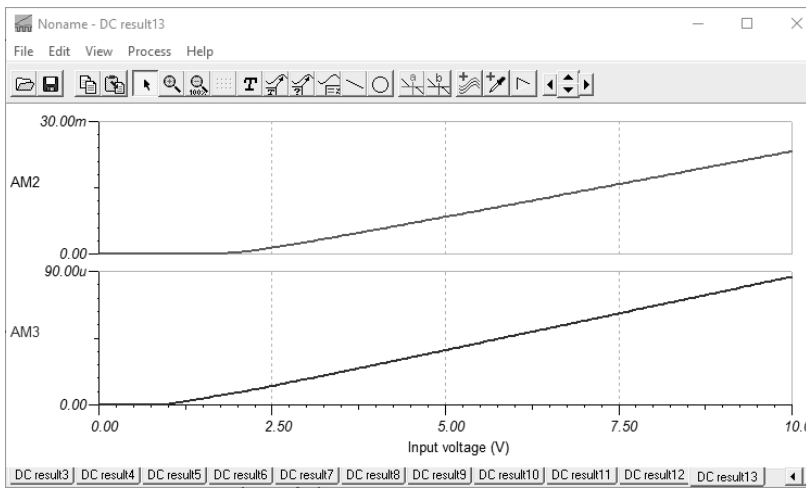
Rys. Układ Darlingtona (p25)

Parametryzacja rezystora R2 pomiędzy **0 a 200** (w zakładce *Analysis*) daję poniższą **odpowiedź układu**.



Rys. Parametryzacja rezystora R2

Parametryzacja napięcia w przedziale od **0 do 10** (w zakładce *Analysis*) daje poniższą odpowiedź układu.



Rys. Parametryzacja napięcia wejściowego

Uwaga: Przy założeniu, że prąd wymagany do diody LED to od około kilku do kilkudziesięciu mA, to budowanie układu z parą tranzystorów może wydawać się nadmierne, ale na potrzeby prezentacji powinno być wystarczające.

> UKŁADY SCALONE

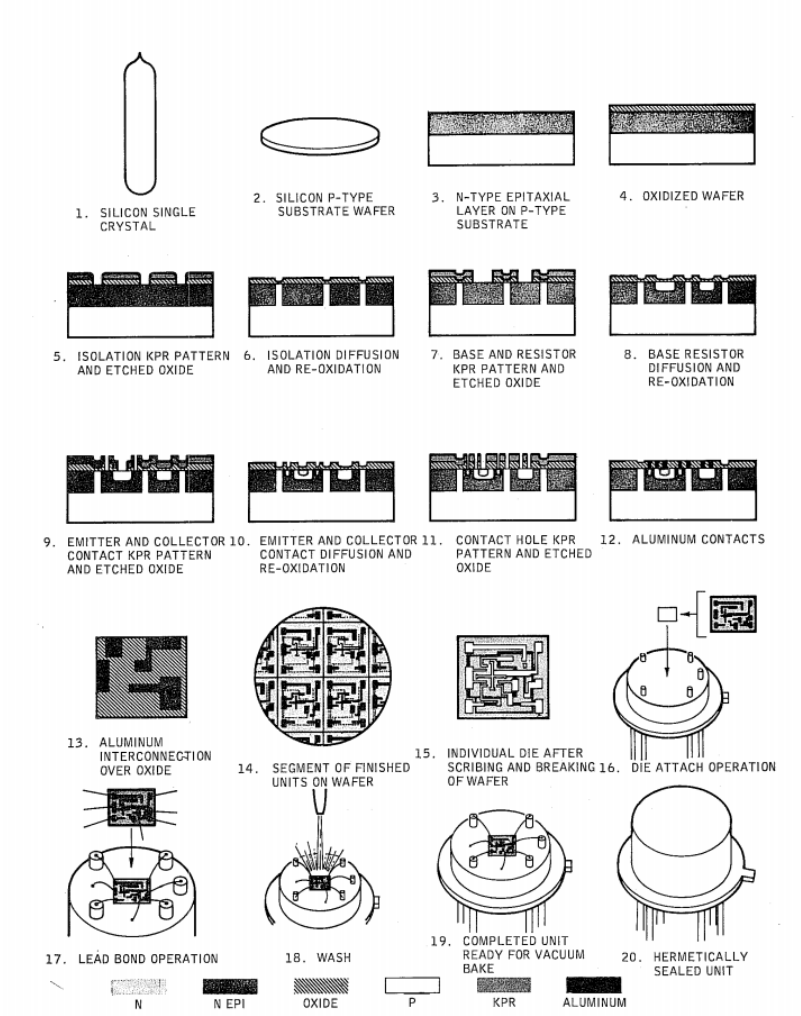
Zminiaturyzowane, zintegrowane układy oferujące złożone operacje. Dzięki miniaturyzacji zwiększa się szybkość reakcji i działania, a w dłuższej perspektywie zmniejsza się koszty produkcji i obsługi.

Po krótkim odświeżeniu wiedzy dotyczącej podstaw elektroniki i tranzystorów teraz czas na **układy złożone**. Zamiast **samemu** budować poszczególne układy, mamy możliwość korzystać z **gotowych komponentów**. Przykładem takiego układu scalonego może już być nawet para tranzystorów Darlingtona jeśli będzie w **zwartym układzie**.

Organizacja i produkcja

Proces produkcji układów scalonych jest skomplikowany a mnogość dostępnych opcji duża

[006] Proces wytwarzania układów scalonych opartych na półprzewodnictwie składa się z wielu etapów. Opiera się na **krystalicznych formach, deponowaniu warstw, naświetlaniu, płukaniu** itd. Poniżej przykładowy proces stosowany w latach 1960-tych w firmie Control Data Corporation.



Rys. Proces wytwarzania układu scalonego [006]

[006] Zaletami stosowania układów scalonych są niskie koszty, zmniejszony rozmiar i waga, mniejsze pobór prądu, zwiększona wydajność, zwiększona niezawodność.

DTL (DIODE-TRANSISTOR LOGIC)

Figure 5-1 shows the logic diagram and schematic of a popular DTL logic inverter.

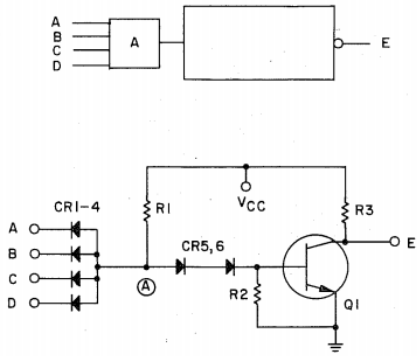


Figure 5-1. DTL Logic

Rys. Przykład logiki DTL [006]

[W] Układy mogą być budowane przykładowo w konwencji **RTL** (logika rezystor-tranzystor), **TTL** (logika tranzystor-tranzystor), **CMOS** (metal-tlenek-półprzewodnik), **DTL** (logika dioda-tranzystor) itd.

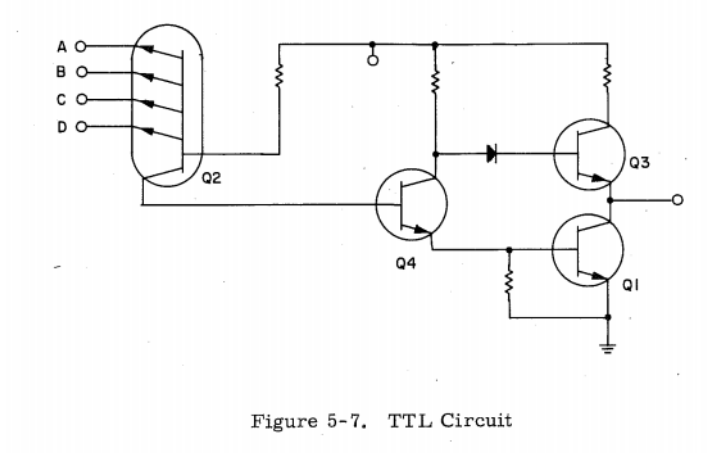


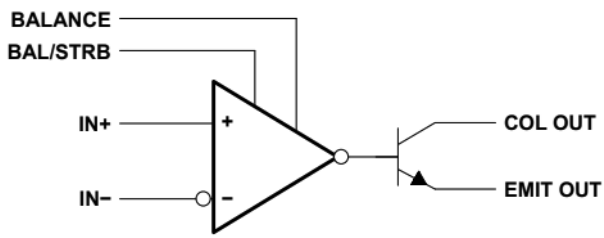
Figure 5-7. TTL Circuit

Rys. Przykład logiki TTL [006]

Komparator LP311

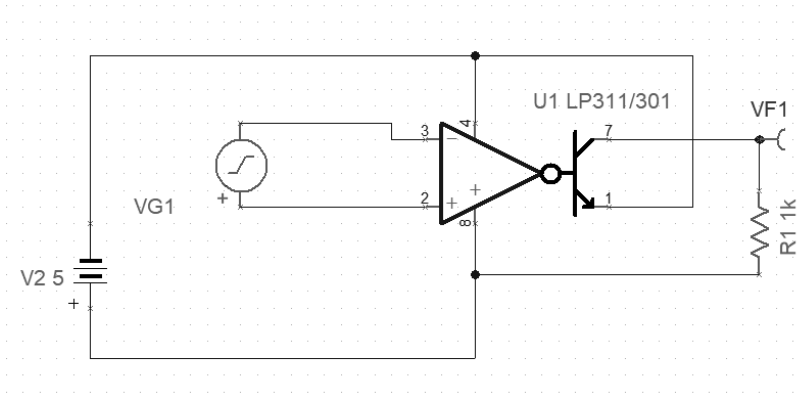
Układ porównujący napięcia, dzięki któremu możemy analizować przemienny sygnał

[203] Układ ten to **komparator napięcia różnicowego**. Występuje pod kilkoma innymi oznaczeniami, w wielu dostępnych wariantach obudowy.

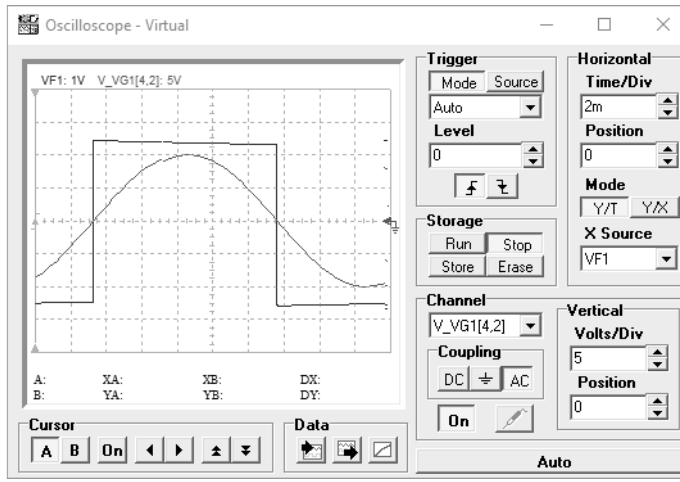


Rys. LP311

Jeśli napięcie na wejściu nieodwracającym jest większe niż na wejściu odwracającym, wynikiem działania układu będzie wyjście w stanie wysokim. W przeciwnym razie wyjście będzie w stanie niskim. Dzięki takiej własności, możemy zbudować układ z przemiennym sygnałem wejściowym i **wykrywać przejścia przez stan równowagi** na wartości zerowego napięcia.



Rys. Wykrywanie przejścia przez stan równowagi (p26)

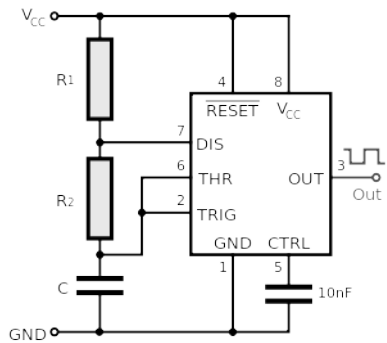


Rys. Sygnał wejściowy i wyjściowy

LM555

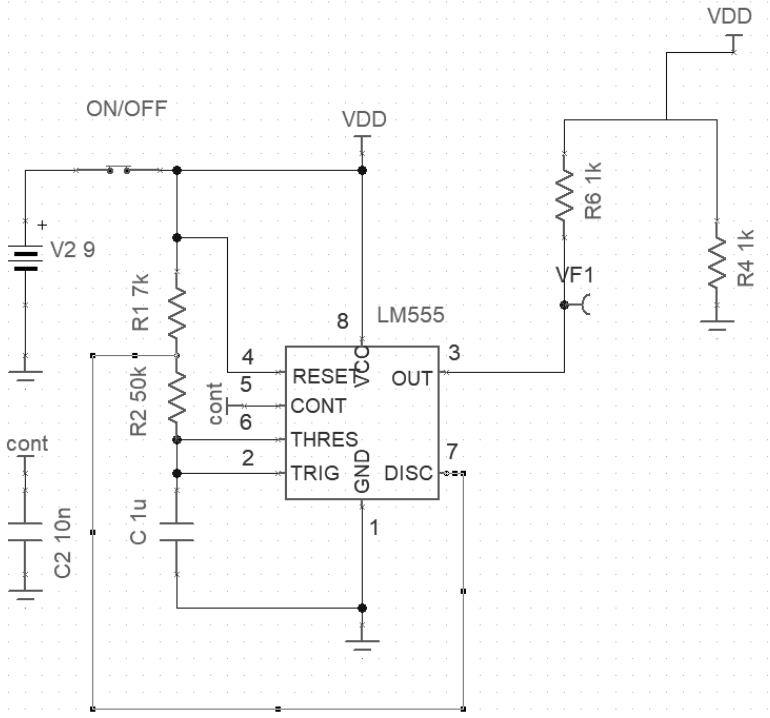
Zegar taktujący pracę maszyny cyfrowej.

[W] Układ scalony umożliwiający realizację **układu czasowego lub multiwibratora**. Zaprojektowany w 1970 w firmie Signetics. Układ wewnątrz składa się z 23 tranzystorów, 2 diod i 16 rezystorów. Ma trzy tryby działania: **monostabilny** (generator pojedynczego impulsu), **astabilny** (generator impulsów prostokątnych) lub **bistabilny** (przerzutnik dwustanowy typu *flip-flop*).

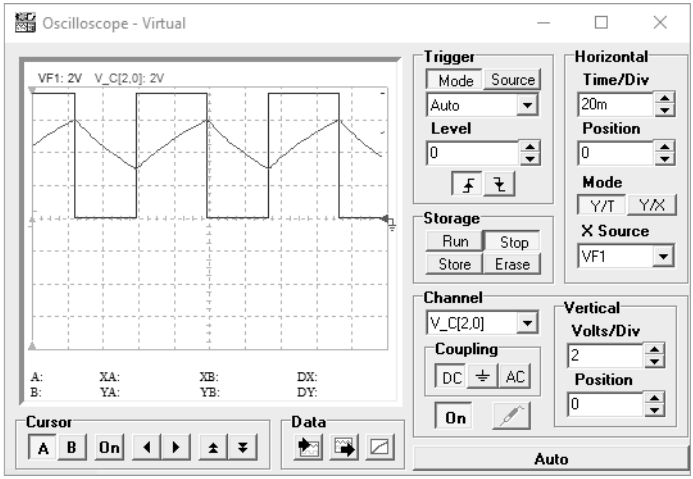


Rys. Przykładowy obwód z LM555

Sam układ LM555 składa się wewnątrz z dzielnika napięcia, komparatorów napięć, przerzutnika RS oraz buforu wyjściowego i tranzystora rozładowującego **zewnętrzny kondensator**.



Rys. Układ LM555 w praktyce (p27)



Rys. Porównanie sygnału wyjściowego z napięciem kondensatora

Jak widać na powyższym ekranie z oscyloskopu **wzrost napięcia na kondensatorze powoduje powstawanie sygnału wysokiego na wyjściu**. Opadanie napięcia zaś powoduje podawanie stanu niskiego na wyjściu. Za **regulowanie przebiegu** sygnału wyjściowego odpowiadają dwa rezystory, R2 i R1 oraz kondensator C1.

[W] W trybie **astabilnym**, czas trwania stanu **wysokiego** na wyjściu wyraża się następującym wzorem:

$$t_1 = \ln(2) \cdot (R1 + R2) \cdot C,$$

Wz. Czas trwania stanu wysokiego

a czas trwania stanu **niskiego**:

$$t_2 = \ln(2) \cdot R2 \cdot C.$$

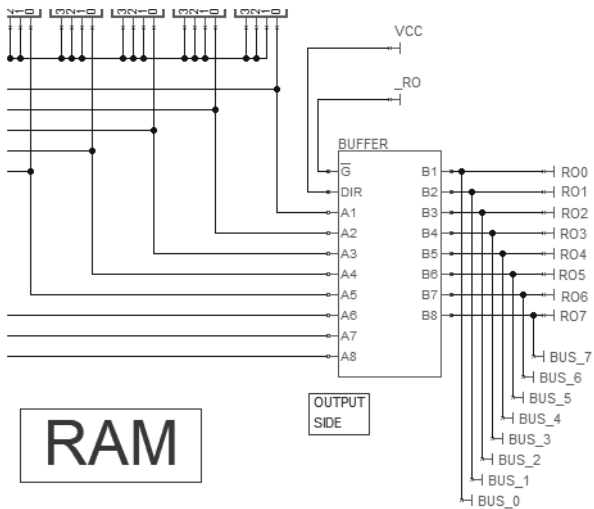
Wz. Czas trwania stanu niskiego

Bardzo ciekawy **efekt** uzyskamy, jeśli znacząco zwiększymy pojemność kondensatora C2 do przykładowo 100mF.

8-bitowy SN74LS245 transceiver (bufor)

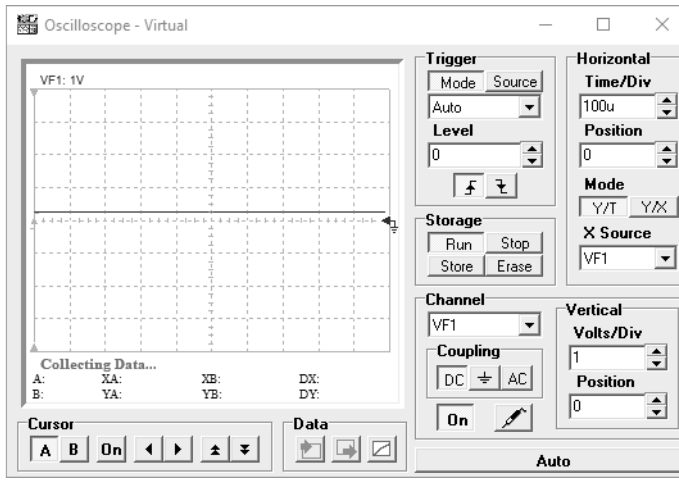
Bufor przechwytujący wyjścia z innych komponentów, umożliwiający na diagnostykę zawartości rejestrów.

Jednym z podstawowych elementów, które mogą być wykorzystane do budowy komputera jest **nadajniko-odbiornik**, zwany też zamiennie **buforem**. Układ taki może być użyty do sterowania cyfrowym wejściem i wyjściem. Jest to układ dwukierunkowy, zatem teoretycznie mamy możliwość określania kierunku przepływu sygnału. W przykładzie układ ten będzie pełnił jednak rolę *de facto* bufora, który umożliwia **przechwycenie wyjścia z innych komponentów**. Dzięki takiemu zabiegowi jesteśmy w stanie lepiej **diagnozować wewnętrzny status komputera**.



Rys. SN74LS245 transceiver

Z punktu widzenia symulacji, warto nadmienić, że stosując program TINA należy dla złożonych układów **wskazywać zasilanie i masę w ich parametrach konfiguracyjnych** poprzez podanie nazw określonych **zworek** (*jumpers*). Z tego też powodu, układy te są prezentowane w okrojonej postaci względem not katalogowych — brakuje bowiem złączy za to odpowiedzialnych. Dla konkretnie tego elementu brakujące złącza to numery 10 i 20, odpowiednio masa i zasilanie.



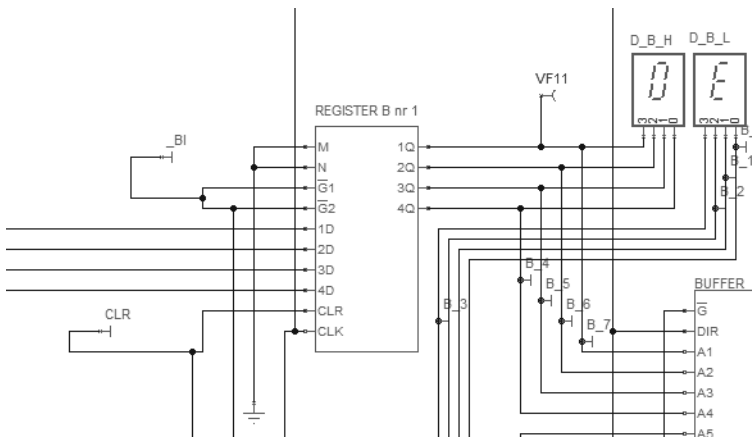
Rys. Stan niski układu

Stan niski dla układu znajduje się w okolicach 300 — 400 mV. Stan wysoki zaś wynosi około 3.3V. Na proponowanym przykładzie wejściem sterujemy za pomocą przełącznika stanu wysokiego i niskiego. **Wejście to złącza A, wyjście to złącza B**. Ostatnią wartą zwrócenia uwagi kwestią jest zanegowanie złącza numer 19, tj. G. Złącze to oznacza **aktywację wyjścia** (*Output Enable*). Podając masę aktywujemy układ, zatem wejście przenoszone jest na wyjście.

4-bitowy rejestr SN74LS173A

Układ zapamiętujący 4 bity danych z możliwością sterowania wejściem i wyjściem

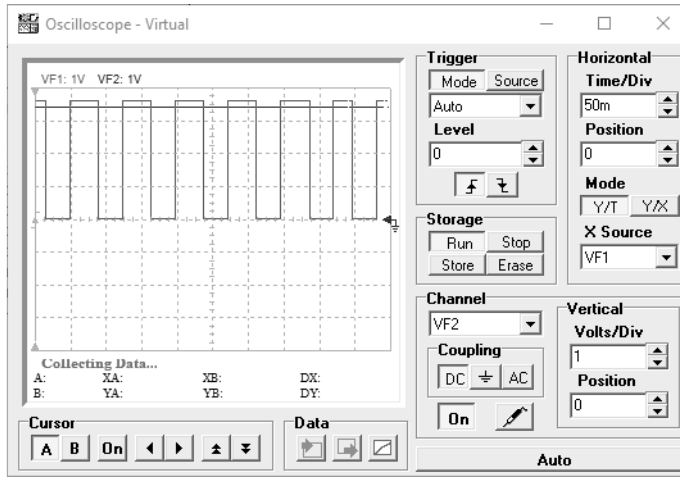
Kolejny komponent to **rejestr pełniący rolę pamięci na dane**. Jego funkcjonowanie polega na przechwytywaniu wejścia i zatrzymywanie go do momentu określenia, że ma ono zostać uwolnione na wyjście. Odczytywanie stanu rejestru nie powoduje kasowania jego zawartości.



Rys. Rejestr SN74LS173A

Funkcjonowanie tego rejestru zależy od **częstotliwości zegara**, który podłączamy na złączu CLK. Do tego złącza podłączony jest zegar LM555 opisany wcześniej. Na wysokim stanie zegara przy złączach nr 9 i 10 (G1, G2) w stanie niskim, dane są ładowane do **przerzutników *flip-flop***. Złącza te są zanegowane. Złącza nr 1 i 2 (M, N) sterują wyjściem. W tym przypadku są podłączone do masy na stałe, czyli **zawsze z układu**

mamy aktywne wyjście na złączach 3 do 6 (1Q, 2Q, 3Q, 4Q). To co wchodzi na wejścia 1D, 2D, 3D, oraz 4D zawsze jest dostępne na wyjściu.



Rys. Sygnał zegara i stan wyjścia

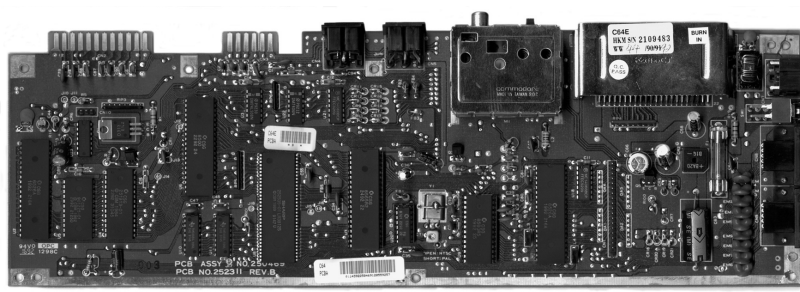
Maksymalne częstotliwości zegara sterującego pracą układu wynoszą od 35 — 50 MHz w zależności od wersji układu. Szybkość reakcji rejestru ma wpływ na szybkość pracy komputera jako całości.

> DYSKRETNY KOMPUTER, PROGRAMOWALNY PROCESOR ARYTMETYCZNY

„Sprzęt i oprogramowanie są równoważne w takim sensie, że jedno można zastępować drugim. Każda operacja realizowana przez oprogramowanie może być wbudowana bezpośrednio w sprzęt, a dowolna instrukcja realizowana przez sprzęt, może być również symulowana programowo.” [005]

Samodzielnie zbuduj procesor arytmetyczny z rejestrami ogólnymi, rejestrami specjalnymi, szyną danych, pamięcią operacyjną oraz dekoderem mikrorozkazów. Programuj na warstwie pamięci operacyjnej jak i pamięci mikroprogramów zmieniając charakterystykę maszyny. Definiuj swój własny kod maszynowy.

[005] Układy elektroniczne wraz z pamięcią i urządzeniami wejścia/wyjścia tworzą **sprzęt maszyny cyfrowej**. Sprzęt składa się z fizycznych obiektów — układów scalonych, pakietów obwodów drukowanych, przewodów, zasilaczy, pamięci, czytników kart, drukarek wierszowych i końcówek — a nie z idei abstrakcyjnych ani instrukcji. Odpowiedzi na pytania jak te obiekty są skonstruowane i jak one działają należy do **elektroniki**.



Rys. Płyta główna komputera Commodore 64 [W]

W poprzednim rozdziale przedstawiłem kilka przykładów **gotowych układów**. Były to komparator i zegar. Nie bez przyczyny jednym z przykładów był układ LM555, gdyż będzie on

potrzebny do budowy komputera. W tym rozdziale, chciałbym przedstawić sposób w jaki przedstawione wcześniej zasady i komponenty mogą zostać wykorzystane do **budowy pełnoprawnego komputera**. Za pełnoprawny komputer rozumiem maszynę, która potrafi **zapisać i odczytać daną w pamięci a także potrafi zmienić punkt sterowania** w zależności od wejścia. Oznacza to innymi słowy, że komputer powinien móc operować na pamięci i wykonywać rozkazy skoku warunkowego.

Uwaga: Chętnych do budowy zgodnie z poniższym opisem uprzedzam, że w materiałach do książki znajdują się źródła i schematy, zatem nie ma konieczności układania wszystkiego samodzielnie.

[005] Aby móc przystąpić do próby utworzenia **komputera, maszyny cyfrowej**, która będzie mogła być programowalna, w tym sensie, że będzie w stanie **przechować, przemieścić** dane, wykonać proste **obliczenia** a także posiadać **instrukcje sterujące** wykonaniem programu — opiszę tutaj kilka podstawowych aspektów budowy typowej maszyny tego rodzaju.

Proces i jego program

Proces jest wykonującym się programem o określonym stanie.

[005] Podstawowym pojęciem potrzebnym do zrozumienia organizacji maszyn cyfrowych jest pojęcie **procesu**. Jest on rozumiany w zasadzie jako wykonujący się **program**. Jest to aktywna jednostka, która może powodować **zdarzenia**. Proces jest przeciwieństwem programu, który jest jednostką pasywną. Program staję się procesem po jego aktywowaniu poprzez pobranie go przez mechanizm wykonawczy danego komputera. W każdej chwili proces jest w pewnym **stanie**. Stan procesu wskazuje miejsce obliczenia, w którym znajduje się proces. Stan zawiera wszystkie informacje potrzebne do **kontynuacji zatrzymanego procesu**. Stan procesu zawiera m.in. program, wskaźnik instrukcji, wartości zmiennych i stałych oraz stany urządzeń. Jeśli program napisany jest tak aby zależny był od czasu, wtedy możliwości ku jego zatrzymaniu i wznowieniu pracy są zdecydowanie ograniczone i mogą powodować nieprawidłowości w uzyskiwanych wynikach. Może się zdarzyć, że program będzie sam zmieniał swoją treść.

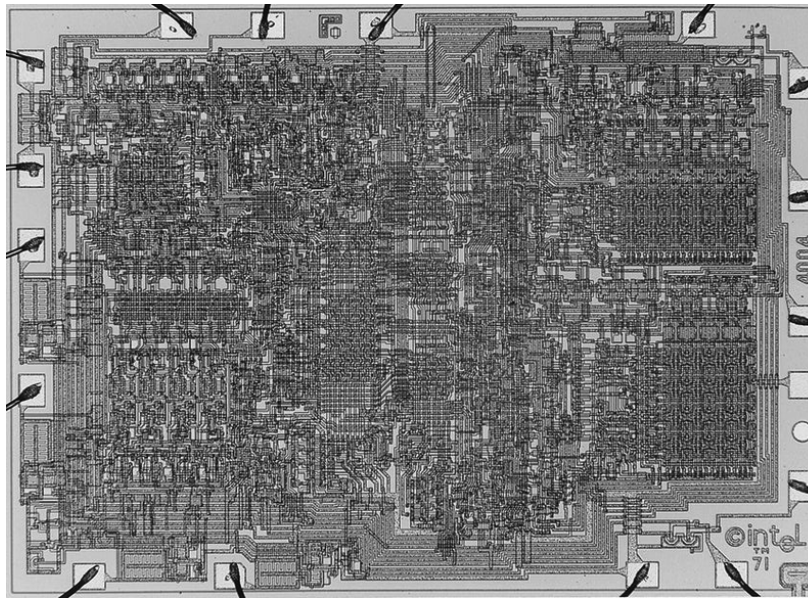
Poziomy realizacji

Układy elektroniczne, które fizycznie lub logicznie realizują poszczególne operacje zlecane komputerowi poprzez podawanie rozkazów na każdym kolejnym poziomie realizacji.

Procesor

Dyskretny lub scalony układ realizujący programowalną obsługę arytmometru, rejestrów oraz pamięci, umożliwiający wykonywanie obliczeń i sterowań.

[005] Ilość funkcji komputera, które obsługiwane są na poziomie sprzętowym a nie na poziomie mikroprogramowania (kolejny poziom) zależy od tego jaką przyjmuje się dla danego komputera **architekturę**. Przykładowo można podać, że operację mnożenia można realizować za pomocą dodawania. To czy mnożenia będzie realizowane jako dedykowany układ elektroniczny czy też za pomocą mikroprogramu, zależy od wybranej architektury rozwiązania. Istnieją też sposoby aby warstwa sprzętowa mogła wykonywać **więcej niż jedną instrukcję w danym momencie**. Wyróżniamy maszyny wieloprocesorowe, maszyny typu SIMD (*Single Instruction Multiple Data*), maszyny hybrydowe z wieloma jednostkami funkcjonalnymi oraz maszyny potokowe.



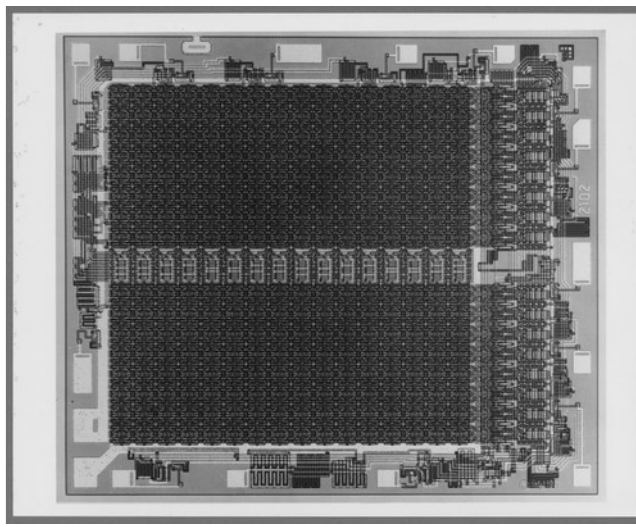
Rys. Przekrój procesora Intel 4004 [212]

W relacji procesora do pamięci istotną kwestią jest fakt, że **procesor sam z siebie nie jest w stanie rozróżnić danej od rozkazu**, jeśli nie zastosujemy dodatkowo jakiegoś metabitu, który by na tą własność wskazywał. Jeśli wykonamy skok do komórki pamięci z obszarem danych, wówczas zachowanie takiego programu będzie nie tyle nieokreślone co określone tym, co jest zawartością komórki, w której w danym momencie znajdują się dane a nie rozkazy.

Pamięć

Roboczy obszar przechowywania danych i rozkazów, z których może korzystać procesor centralny. Może być również częścią samego procesora.

[005] Jest to część maszyny cyfrowej, w której przechowywane są programy i dane. Podstawową jednostką pamięci jest **cyfra dwójkowa** zwana bitem. Bit może mieć wartości 0 lub 1. Jest to najprostsza możliwa jednostka. Maszyny cyfrowe mają arytmetykę dwójkową, gdyż jest ona **efektywna**.



Rys. Pamięć Intel 2102 SRAM [211]

Stosowanie **stanu niskiego i wysokiego** jest bezpieczniejsze niż przykładowo układ pracujący z napięciem 10V, gdzie każdy kolejny stan byłby krotnością 1V. Z pewnością taka maszyną optymalnej gospodarowałaby układem elektronicznym, ale mogłaby podlegać zakłóceniom, które potencjalnie mogłyby destabilizować jej pracę. Różne maszyny mają

różną **długość słowa w komórce pamięci**.

Magazyn

Trwały, niezależny od zasilania nośnik danych.

Pamięci o krótkim czasie dostępu są kosztowne względem całości systemu komputerowego, dlatego dla **dużych trwałych zbiorów danych** stosuje się pamięci wolniejsze, tańsze. **Im krótszy czas dostępu tym większy koszt** produkcji. Chcąc przechowywać dane w sposób trwały, mamy do wyboru pamięć **taśmową, dyskową, bębnową, optyczną lub półprzewodnikową**.

Poziom 1 — mikroprogramowanie

Mikroprogram znajdujący się w specjalnej pamięci, pracujący na osobnym liczniku, działający na niskim stanie zegara, steruje otwieraniem i zamykaniem poszczególnych ścieżek do komponentów elektronicznych. Alternatywą dla takiego sterownia są dedykowane układy logiczne mające w swojej konstrukcji zaszyte algorytmy analogiczne do mikroprogramu.

[005] Większość rozkazów komputera powoduje **przemieszczanie** danych z jednej części maszyny cyfrowej do innej, a niektóre wykonują proste **sprawdzenia**. Ten poziom nazywany jest poziomem mikroprogramowania, a realizujące go interpretatory nazywane są **mikroprogramami**. Jest to tak zwany poziom nr 1. Wyróżniamy dwa główne sposoby realizacji mikroprogramowania, **poziomy** i pionowy. Ten pierwszy sposób oznacza, że każdy krok mikroprogramu **steruje wieloma elementami układu**. Ten drugi, czyli pionowy oznacza, że kolejne mikrorozkazy mają bardzo wąskie znacze-

nie. Wybór konkretnego sposobu zależy od wielu czynników, przykładowo od oczekiwanej częstotliwości pracy. Zbiór rozkazów i organizacja poziomu mikroprogramowania to w istocie zbiór rozkazów i organizacja sprzętu, czyli procesora centralnego. Natomiast zbiór rozkazów i organizację konwencjonalnego poziomu maszynowego wyznacza mikroprogram, a nie sprzęt.

Poziom 2 — język maszynowy

Pierwszy dostępny dla programisty język najniższego poziomu, w którym można rozwiązywać problemy. Rozkazy maszynowe mapowane są na mikrorozkazy.

Na tym etapie definiujemy **instrukcje, które będą tłumaczone przez fizyczną maszynę na określone sterowania bramek** i pamięci przez mikrokod poziomu pierwszego. Za pomocą języka maszynowego uzyskujemy wystarczający poziom abstrakcji aby móc relatywnie wygodnie programować komputer. [005] Maszyna z dużą liczbą rozkazów oznacza, że każdy rozkaz jest bardziej wyspecjalizowany. Kompilatory języków wyższego poziomu, na ogół działają lepiej na maszynach z małym, dobrze dobranych zbiorem rozkazów niż na maszynach mający duży zbiór rozkazów.

Uwaga: jest to ostatni poziom, który będzie realizowany w tym rozdziale

Poziom 3 — system operacyjny

Sposób organizacji dostępu do czasu procesora i bezpiecznego przydzielania obszarów pamięci dla programów użytkownika.

Programy użytkownika obecne są w **przestrzeni pamięci** ale nie są tam same. Obok nich mogą pracować programy innych użytkowników, a także sam system operacyjny, który z reguły będzie **zarządzał czasem dostępu do procesora, pamięci i urządzeń peryferyjnych**. Możliwości jakie dostarczają systemy operacyjne uzależnione są od możliwości procesorów, jak przykładowo wsparcie dla realizacji pamięci **wirtualnej**, pamięci **chronionej** itd. Programiści operujący na niskich poziomach abstrakcji maszyny cyfrowej to **programiści systemowi**. Część funkcji poziomu trzeciego może być realizowana zarówno bezpośrednio na języku maszynowym czy nawet mieć korelację do mikroprogramowania.

Poziom 4 — język asemblera

Czytelna, symboliczna reprezentacja kodu maszynowego z możliwościami organizacji kodu poprzez bloki, komentarze czy nawet makra.

Ten poziom, jest w rzeczywistości **symboliczną wersją jednego z niższych języków**. Umożliwia on pisanie programów dla poziomów 1, 2 lub 3 w postaci relatywnie wygodnej, dużo bardziej wygodnej chociażby od programowania samym kodem maszynowym. Program, który wykonuje **tłumaczenie** na tym poziomie nazywany jest asemblerem. Jest to w zasadzie pierwszy ściśle **niezależny** na tej ścieżce poziom, dzięki któremu kod zaczyna mieć **cechy przenoszalnego** pomiędzy różnymi typami (zbliżonymi do siebie) maszyn.

Poziom 5 — język problemowy

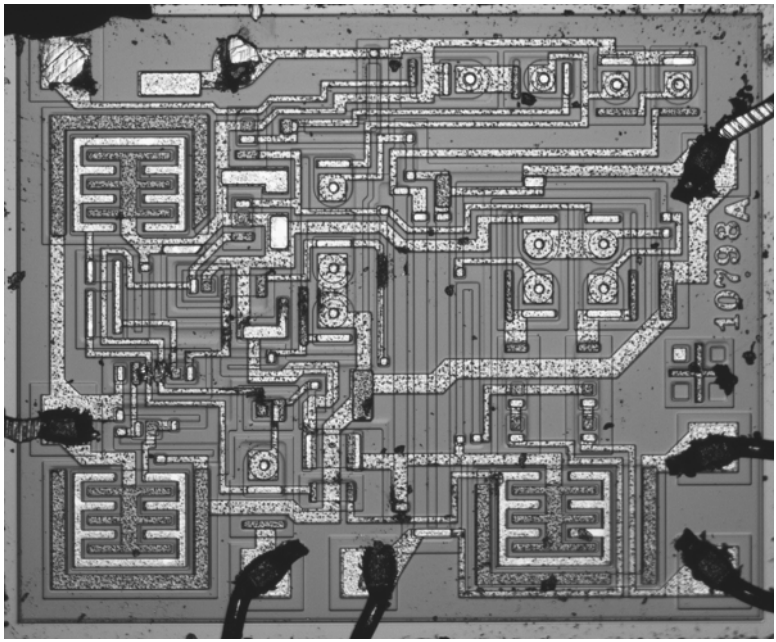
Naturalna reprezentacja algorytmów i struktur danych, kompilowalna pośrednio przez asembler lub wprost do języka maszynowego.

Mając różne problemy do rozwiązania najwygodniej jest stosować języki wysokiego poziomu, **w pełni niezależne** nie tylko od typu czy rodzaju maszyny ale nawet stosowanej architektury i trybu pracy. Poza językami kompilowalnymi wprost do kodu maszynowego lub pośrednio przez asemblery mamy dostępne również **maszyny wirtualne i interpretery**.

Szyna danych i ogólny rejestr A

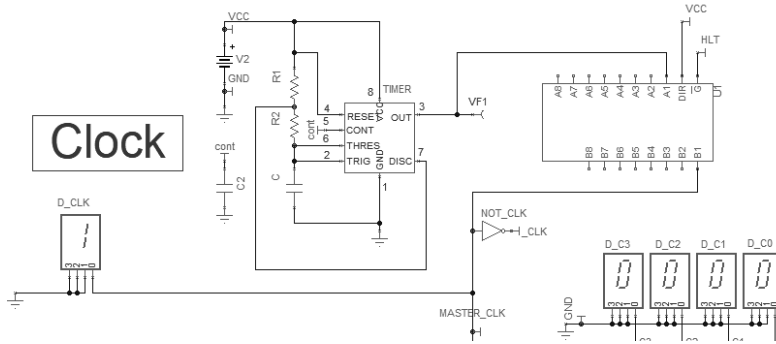
8-bitowa magistrala łącząca większość komponentów komputera w całość

W tej części opiszę jak złożyć **szynę danych używając dotychczas przedstawionych komponentów**, czyli LS245 oraz LS173, czyli odpowiednio 8-bitowego odbiorniko-nadajnika i rejestru 4-bitowego. Układ ten zaczyna być już całkiem skomplikowany, zatem jego prezentacja w formacie książkowym jest nieco utrudniona. Jego budowa będzie rozłożona na kilka fragmentów. Częścią obwodu jest zegar LM555, którego sygnał przekazany jest to dwóch rejestrów 4-bitowych.



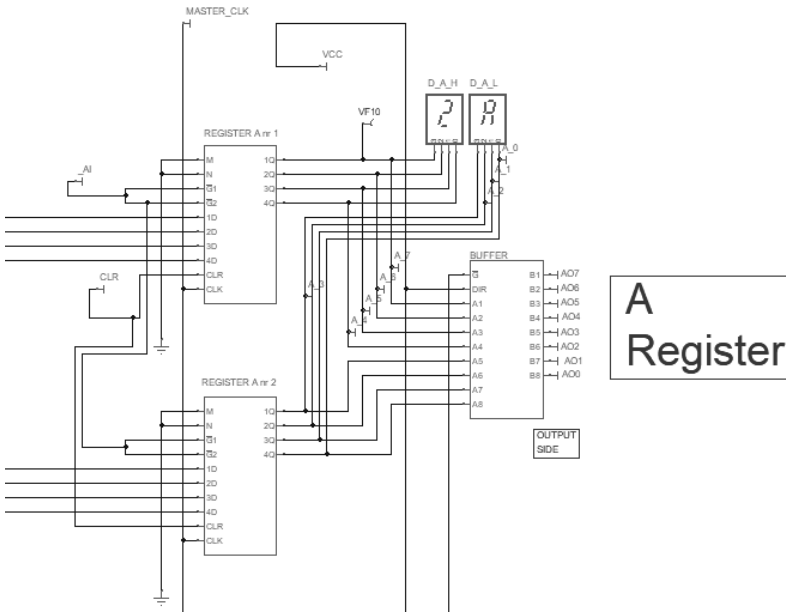
Rys. Przekrój układu 555 [210]

Rejestry te są załączone aby **zawsze przekazywały sygnał wyjściowy** a sterujemy jedynie tym, czy przyjmują one sygnał wejściowy. **Oba 4 bitowe sygnały tworzą sygnał 8-bitowy** przekazywany do układu typu *transceiver*. Komponent ten ma **sterowane wyjście**, tj. określamy, czy wejście przepisywane jest na wyjście.



Rys. Układ zegara (p28)

Podsystem zegara to ten sam układ, który był prezentowany wcześniej, z kilkoma jednak drobnymi różnicami. Wprowadzone zostały 3 zworki, GND, VCC i MASTER_CLK oraz _CLK. Po kolei odpowiedzialne za **masę, zasilanie i sygnały samego zegara**. Taka notacja w programie TINA umożliwia łatwiejsze punkty odniesień i można je traktować jako bezpośrednio podłączone przewody o danej nazwie.



Rys. Układy rejestru

Układ rejestru to ten sam, który był wcześniej prezentowany. Z lewej mamy sygnały danych na liniach D, po prawej wyjścia na liniach Q. Sygnały sterujące to _AI (ładowanie), CLR (czyszczenie). Jeśli jakieś dane zostaną załadowane do rejestru będziemy **zawsze mogli je zobaczyć na wyświetlaczu** znakowym, gdyż rejestr został ustawiony na złączach M i N w stan określający, że wyjście jest zawsze aktywne. Linie sygnału wyjściowe poza wyświetlaczem, **kierowane są dalej do układu typu *transceiver***, który pełni rolę bufora wstrzymującego lub aktywującego wyjście z rejestru na szynę danych. Należy zwrócić uwagę, że całość może pracować jako wejście lub jako wyjście, zamiennie.

Rejestry ogólne A i B

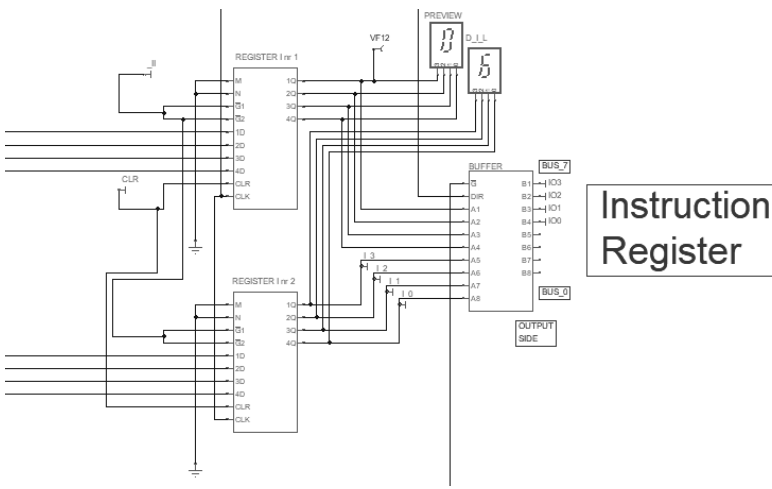
Więcej rejestrów to więcej możliwości realizacji funkcji

W projektowanym komputerze stosowane będą **dwa 8-bitowe rejestry ogólnego przeznaczenia**. Rejestr A znajduje się bezpośrednio pod zegarem. Rejestr B znajduje się poniżej rejestru A. Rejestry A i B mają bardzo zbliżoną konstrukcję. Oba rejestry podłączone są do tej samej jednej szyny danych, która przebiega od góry do dołu na obwodzie po lewej jego stronie.

Rejestr instrukcji

Rejestr specjalnego zastosowania, służący do sterowania pracą komputera

W komputerze poza rejestrami ogólnego przeznaczenia, potrzebny przeważnie jest też rejestr instrukcji. Jego **budowa jest niemalże identyczna** jak pozostałych rejestrów. Wartość instrukcji, czyli **dolne wartości** sygnału będą pobierane bezpośrednio ze zworek przez mechanizm **mikroprogramowania**. Wartości **adresów**, czyli **górne wartości** sygnału wchodzą do rejestru i mogą z niego dalej wyjść do pozostałych komponentów komputera.



Rys. Rejestr instrukcji

Celem uproszczenia ścieżek (okablowania) pomiędzy poszczególnymi odległymi komponentami stosowane są **zworki**. Dzięki czemu na wyjściu z buforów określamy w programie TI-

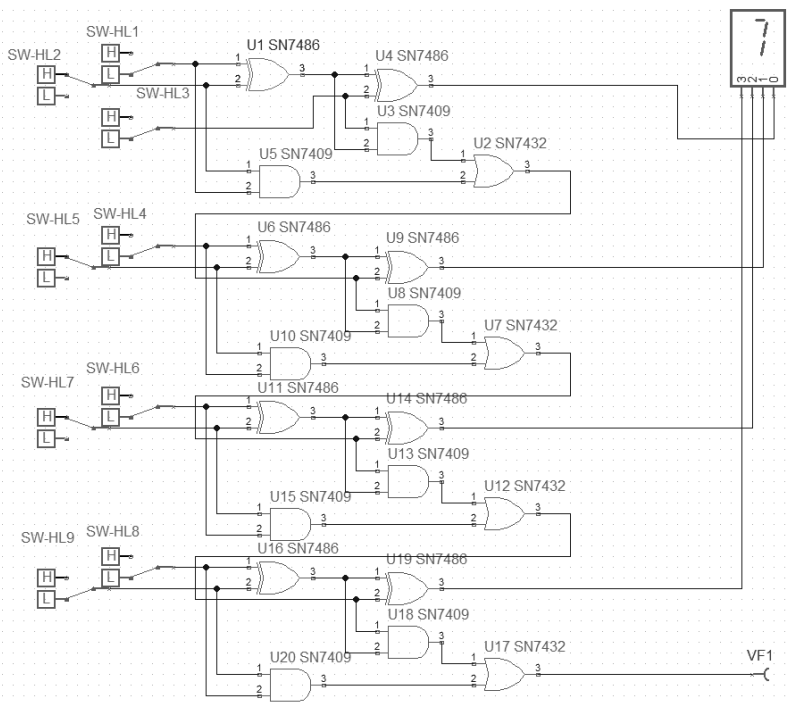
NA tylko i wyłącznie zworkę **docelową**, poprzez podanie **identycznej nazwy** jak tych zworek co zostały podłączone bezpośrednio na liniach szyny danych. Każdy rejestr otrzymał swoje zworki wyjściowe oraz analogiczne zworki na liniach szyny danych.

Całkiem sensowną optymalizacją może być zastosowanie **makr**, które pozwalają na grupowanie komponentów, dzięki czemu na diagramie występuje jedynie ograniczona liczba elementów. Wersja TINA-TI 9.2.30 nie udostępnia jednak tych funkcji i należałoby użyć programu TINA Design Suite.

Sumator

Układ sumujący zrealizowany na bramkach

Kolejnym elementem w jaki należy wyposażyć komputer jest **jednostka arytmetyczna**. Jednym z jej składników jest bitowy **układ sumujący**. Taki układ można złożyć za pomocą bramki NAND lub NOR, ale czytelniej na pierwszy raz jest zastosować bramki **XOR, AND i OR**.



Rys. Układ sumujący

Wejścia 3 do 0 wyświetlacza odpowiadają kolejno za bity

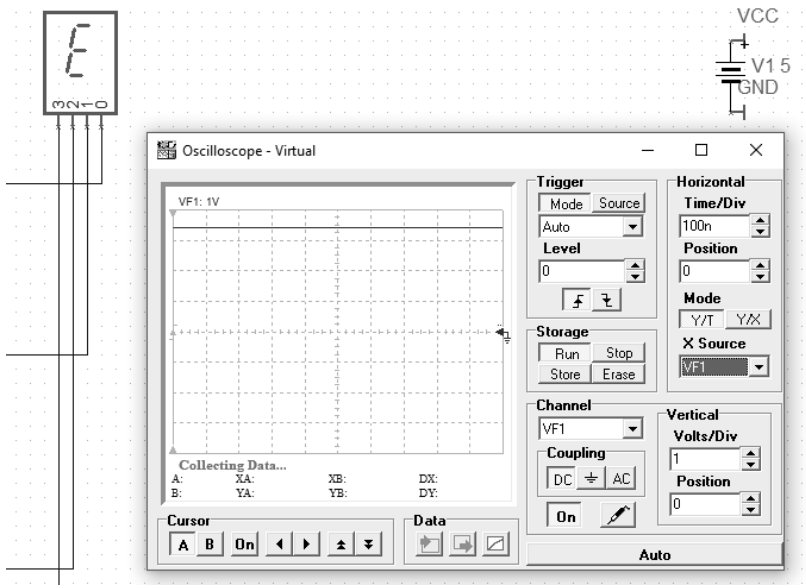
od prawej do lewej strony. Na dole mamy złącze napięciowe (*voltage pin*) na potrzeby **bitu przeniesienia**. Tą wartość prezentujemy na oscyloskopie.

0111+0011=1010

Liczba (binarna) 1010 to A (hex) i taki też wynik prezentuje się na wyświetlaczu na układzie testowym (dziesiętnie 7+3). Żeby jednak przetestować bit przeniesienia należy użyć nieco innych wartości wejściowych (dziesiętnie 15+15):

1111+1111=1 1110

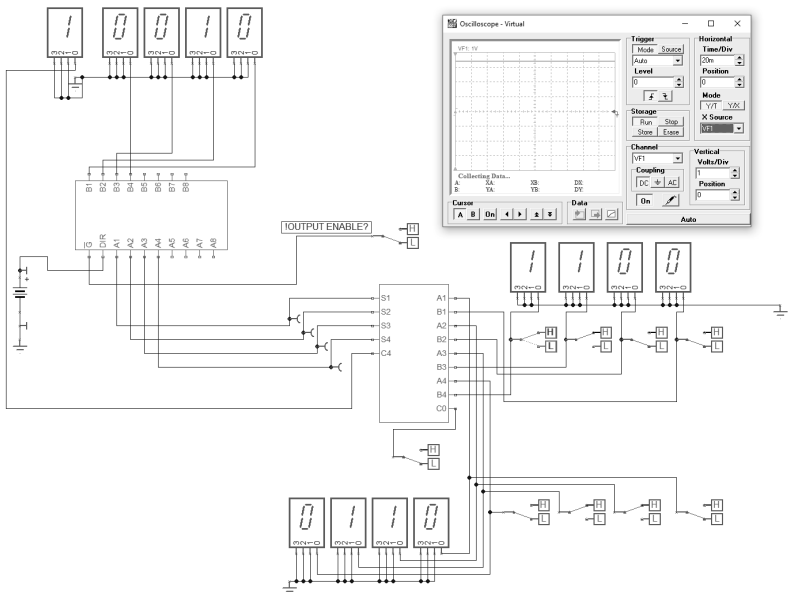
Na wyświetlaczu powinna zaprezentować się wartość E (dziesiętnie 14) oraz **poziom wysoki na oscyloskopie podłączonym do złącza bitu przeniesienia**.



Rys. Testowanie bitu przeniesienia

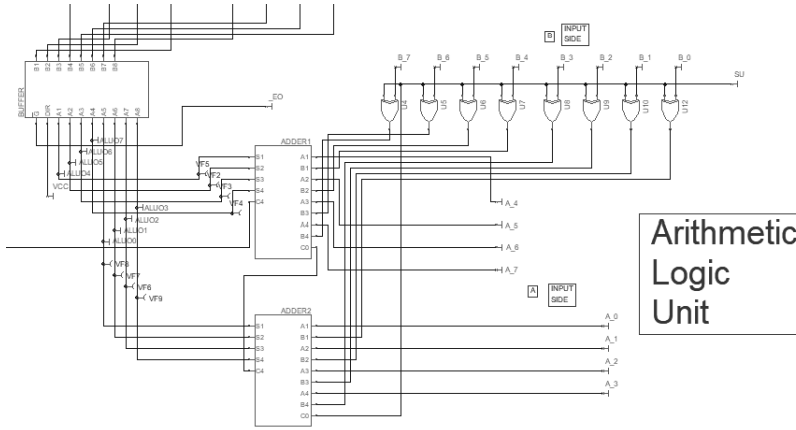
ALU

Zamiast realizować samodzielnie (tj. dyskretnie) wszystkie operacje arytmetyczne mamy możliwość zastosować **gotowe układy** ku temu przeznaczone. Jednym z nich jest LS283, który udostępnia **sumowanie 4-bitowe z bitem przeniesienia**. Dla ułatwienia diagnostyki do obwodu podłączyłem szereg wyświetlaczy LCD na każdą z liczba wejściowych. Wejście B jest u dołu, wejście A jest po prawej. Wyjście natomiast jest u góry po lewej, zawiera ono ponadto bit przeniesienia (skrajny lewy).



Rys. 4-bitowy sumator na układzie SN74LS283

Jako jednak, że mamy potrzebę wykonywać **operacje 8-bitowe**, potrzebujemy **dwóch sumatorów 4-bitowych**. Wyjście komponowane jest za pomocą bufora LS245.

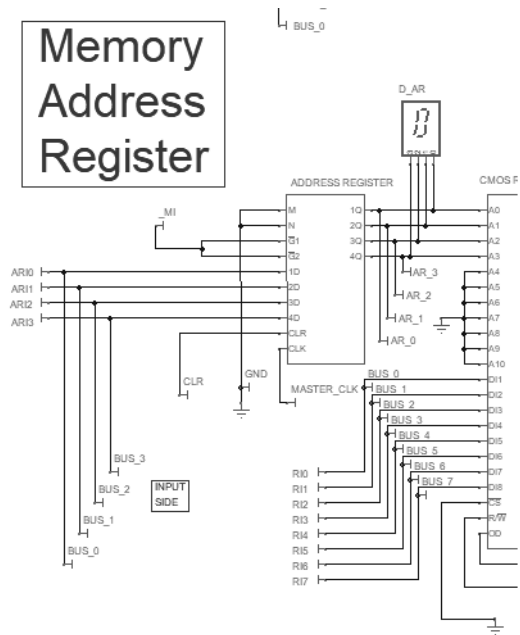


Rys. 8-bitowy sumator z funkcją odejmowania (na bramkach XOR)

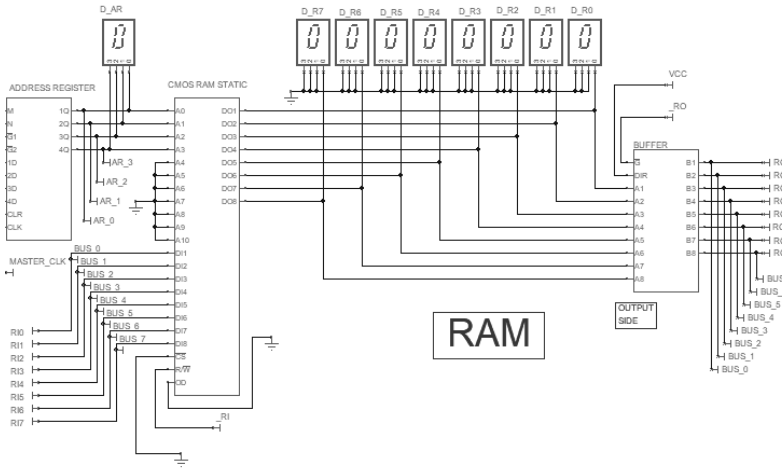
Poza funkcją sumowania, układ posiada teraz także **funkcję odejmowania**. Realizowane jest to na **bramkach XOR** na wejściu B. Układ poza sumą A i B ma możliwość odejmowania A-B.

Pamięć operacyjna

Aby móc pracować wygodnie na większych zbiorach danych dostępnych w szybkim trybie, komputer wyposażamy w układ pamięci operacyjnej. Jest to w tym konkretnym przypadku **pamięć półprzewodnikowa typu CMOS** w postaci **statycznej**. W komputerach osobistych stosujemy przeważnie pamięci dynamiczne z aktywnym odświeżaniem komponentów podtrzymujących stan poszczególnych komórek. Takie rozwiązanie jest efektywne kosztowo.



Rys. Rejestr adresowy



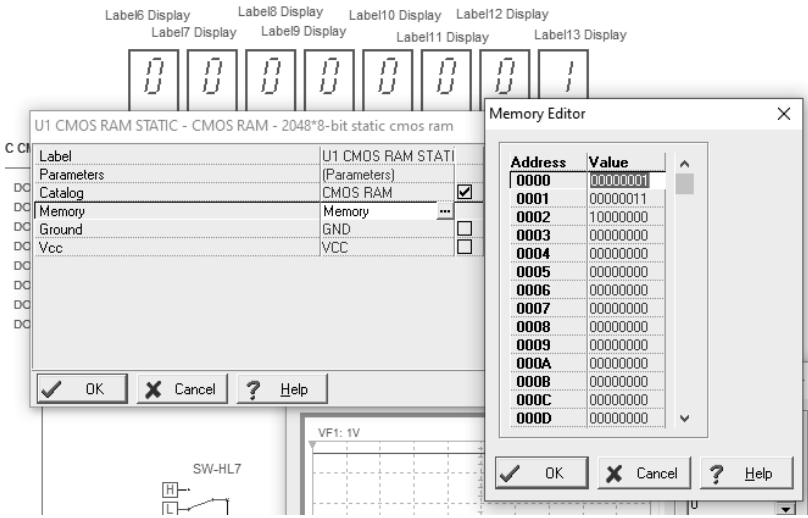
Rys. Pamięć CMOS

Zastosowana na powyższym diagramie pamięć to **układ CMOS RAM 2048 * 8 bit**. Oznacza to że mamy 2048 słów 8 bitowych dostępnych do **zapisu i odczytu**. Pamięci CMOS w programie TINA występują w 3 odmianach (różniących się rozmiarem i długością słowa), które można przyrównać do układu pokroju uPD5101L. Dokumentacja programu TINA nie wskazuje na żaden konkretny model pamięci. Układ ten można stosować jako pamięć operacyjna, pamięć rejestru czy nawet pamięć używana tylko do odczytu, jeśli mamy też taką potrzebę.

Poza układem pamięci na obwodzie występuje również **zegar i rejestr**. Zegar w wiadomy sposób reguluje częstotliwość pracy poszczególnych komponentów, w tym przypadku rejestru adresowego, który ma szerokość 4 bitów. Taka **szerokość adresacji** umożliwia wskazanie jedynie **16 komórek pamięci**. Na potrzeby tego przykładowego komputera powinno to w zupełności wystarczyć. Jeśli jednak komputer miałby posiadać większe możliwości symulacyjne, wówczas szerokość adresu

pamięci musiałaby znacząco wzrosnąć. Alternatywnie rozkazy ładowania adresu mogłyby również składać się z kilku następujących po sobie wywołaniach.

Układem pamięci sterujemy za pomocą 11 linii adresowych (korzystamy z 4, A0-A3) oraz złączami CS!, R/W i OD. Złącze CS! to **aktywacja układu** (zanegowana). Złącze R/W (również zanegowane) ustawione w stan wysoki oznacza **odczyt pamięci**. Jeśli podamy stan niski, wówczas będziemy **pisać do pamięci**. Złącze OD to **aktywacja linii wyjścia** układu. Moduł rejestru adresu sterowany jest za pomocą 2 przełączników, są to para G1 i G2 (oba zanegowane) oraz CLR. Para służy do **załączenia odczytu** a CLR **czyści zawartość** rejestru jaka jest w nim zapisana.

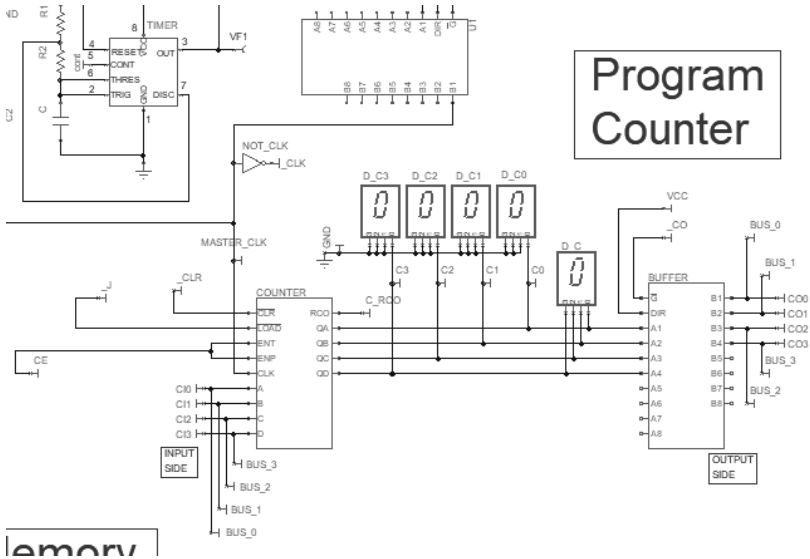


Rys. Ustawianie wartości domyślnych pamięci

Bardzo ciekawą i przydatną funkcją układu pamięci jest **możliwość ustawiania bieżących wartość** z jakimi układ startuje po załączeniu symulacji. W przypadku programu TINA-TI jest to **włączenie oscyloskopu**.

Licznik programu

Aby móc sterować wykonaniem programu, w komputerze konieczne jest posiadanie licznika, który będzie umożliwiał **wskazywanie następnej instrukcji do wykonania** lub przechodzeniem do instrukcji w **operacji skoku** bezwarunkowego lub warunkowego.



Rys. Licznik programu na układzie LS161

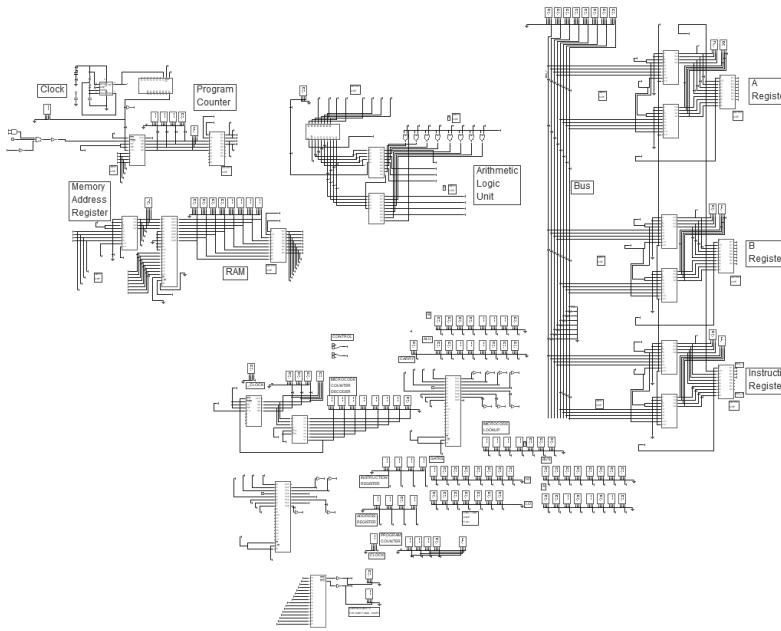
Układ licznika składa się z zegara, który steruje częstotliwością pracy oraz układu **SN74LS161**, który jest **4 bitowym licznikiem z bitem przeniesienia**. Możemy układ uruchomić tylko z samym zegarem i wtedy będzie liczył binarnie **od 0000 do 1111** lub **wskazać** określony punkt startowy, czyli *de facto*

wykonać skok w programie.

Uwaga: Jeśli chodzi o możliwości symulacyjnie w programie TINA-TI, to zauważyłem, że tak jak licznik nie ma problemów z pracą na większych częstotliwościach, tak układ LS161 potrafi działać niestabilnie. Nie jest jednak celem zapewnienie, że budowany komputer będzie działał szybko w symulatorze. Celem jest tylko przejście przez najbardziej charakterystyczne komponenty dyskretne, z których kiedyś mógł składać się komputer a obecnie w większości przypadków są zawarte na jednym układzie scalonym o wysokiej wydajności. Projektowany tutaj komputer można w zasadzie nazwać programowalnym kalkulatorem, coś jak ***Texas Instruments TI-74S****.*

Montaż

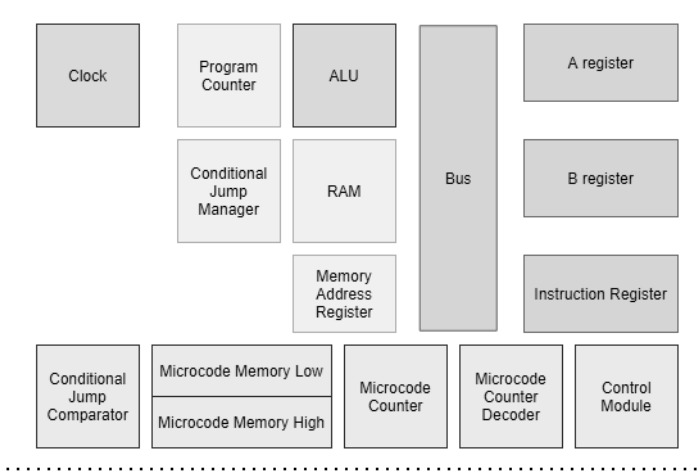
Na tym etapie po montażu zegara, licznika programu, pamięci i rejestru pamięci, jednostki arytmetycznej, rejestru instrukcji i adresów oraz rejestrów A i B, przychodzi czas na finalne podłączenie całości i sprawdzenie czy układ jest ciągły i mierzalny w symulatorze. Wystarczy za każdym razem **uruchamiać oscyloskop** i weryfikować pracę w **wybranych punktach obwodu**.



Rys. Zbiorczy widok całego układu

Aby nie wprowadzić zbędnej plątaniny ścieżek wystarczy

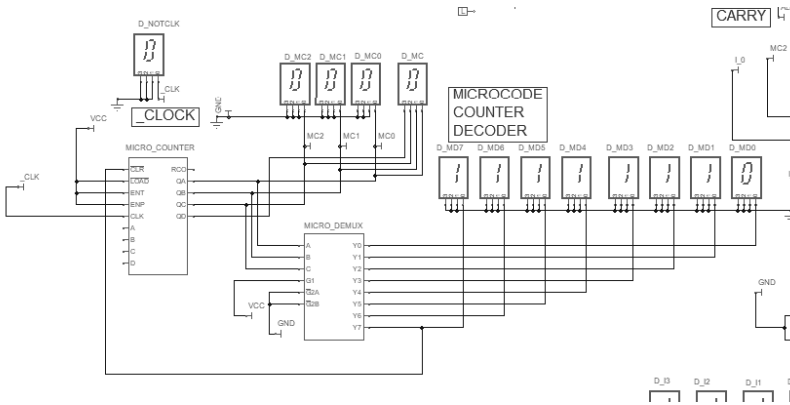
tam gdzie to możliwe, **stosować zworki**, które łączą dwa nawet odległe punkty bez ryzyka pomyłki w okablowaniu. Bardzo ważna jest również **numeracja i kolejność** bitów względem poszczególnych ścieżek.



Rys. Schemat całego układu

Mikroprogram

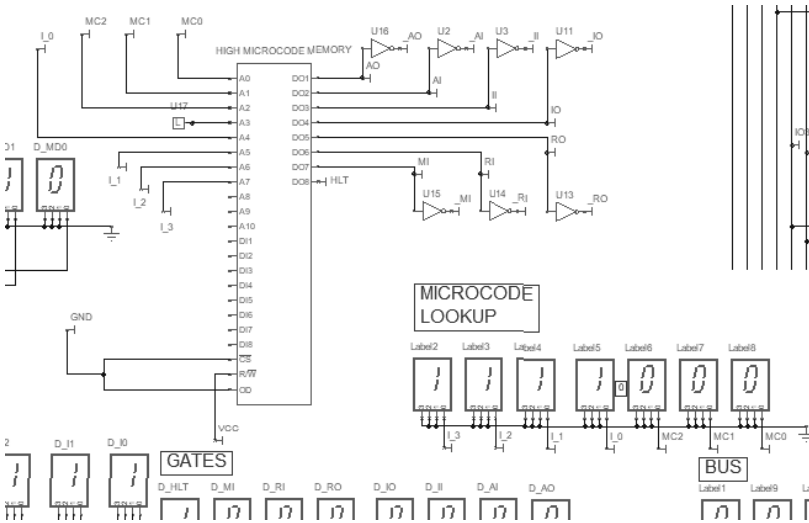
Po montażu **podstawowych komponentów** czas na rezygnację z ręcznego sterowania komputerem (jeśli ktoś przygotował równolegle poszczególne komponenty osobno) i wykonanie niezbędnego **automatu sterującego**. Jest nim **pamięć** i układ **dekodujący mikroprogramy**. Są to **sekwencje bitów sterujących poszczególnymi bramkami na kolejnych cyklach zegara**.



Rys. Mikrodekoder

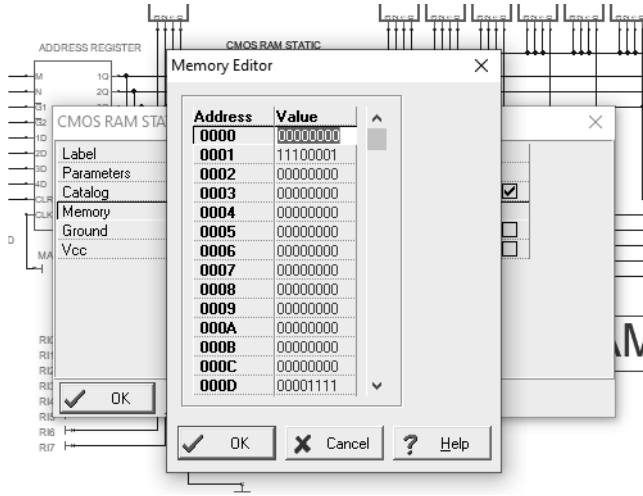
Mikrodekoder przedstawiony powyżej **liczy binarnie** (za pomocą układu licznika) a jego liczenie jest resetowane za pomocą **dekodera i podłączonej linii Y7 z CLR!**. Dzięki temu licznik podrozkazów mieści się na 3 bitach, tj. może przyjmować 8 wartości. Każdy pełen rozkaz, przykładowo operacja załadowania wartość z pamięci do rejestru A (LDA 14), **składa się z wielu podrozkazów** takich jak załadowanie instrukcji z pamięci do rejestru instrukcji czy wstawienie warto-

ści z pamięci operacyjnej do rejestru.



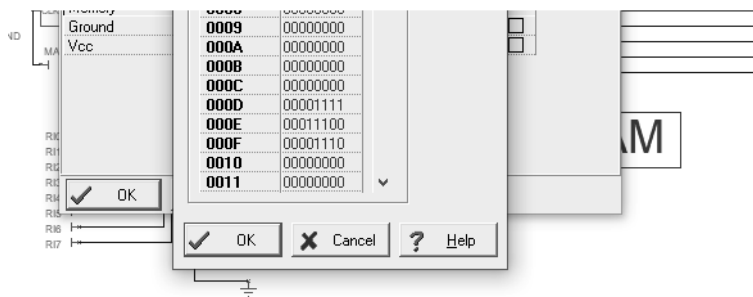
Rys. Pamięć wysoka mikroprogramu

Pamięć mikroprogramu to dwa układy CMOS RAM typu statycznego pokrywające kolejno wysoką część i dolną część wartości wynikowych. Dwa układy są potrzebne gdyż symulator oferuje pamięci 8 bitowe. W komputerze, **sterowanie składa się z 15 bitów sterujących**. Adresem tych podrozkazów jest **4 bitowa wartość kodu instrukcji** oraz **3 bitowy numer kolejny podrozkazu**. Przyjmując, że instrukcje LDA, ADD i HLT mają odpowiednio wartości 0001, 0010 oraz 1111 ich postaci podrozkazów są następujące (w pamięci operacyjnej program przechowujemy w postaci 4-bitowego OPERANDU i 4-bitowej INSTRUKCJI):



Rys. Pamięć operacyjna z programem (RAM)

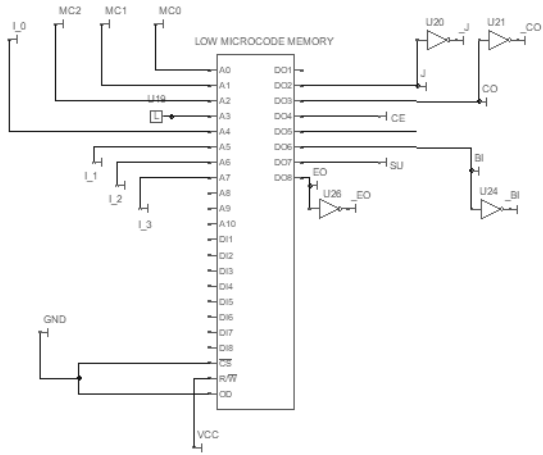
Zerowa komórka pamięci przeznaczona jest na **inicjalizację komputera**. W komórce pierwszej (0001) znajduje się program 1110 0001, co oznacza załadowanie (LDA) wartości 1110 (14 dziesiętnie) do rejestru A.



Rys. Dane i rozkaz zatrzymania komputera (RAM)

W komórkach 000E i 000F znajdują się **dane**, które będą

wykorzystywane przez programy. Komórka 000D zawiera rozkaz 1111, czyli HLT oznaczające **zatrzymanie komputera**.



Rys. Pamięć niska mikroprogramu

Procedura inicjalizacji komputera to *de facto* **pobranie kolejnej instrukcji do wykonywania**. Komputer startuje od adresu 0000 i **szuka odpowiedniego mikrorozkazu** dla tego kroku tj. 0000 i 000 a dalej 0000 001, 0000 010 itd. Wszystkie 15 wysterowań to **połączenie obu pamięci mikroprogramów** (niska i wysoka). Adresowane są one w ten sam sposób, czyli najpierw podajemy kod instrukcji a potem kolejny numer mikrorozkazu. Za pomocą licznika mikrorozkazów oraz rejestru instrukcji uzyskujemy taką parę wysterowań. **Układ bitów mikroprogramu** jest następujący dla **pamięci wysokiej**:

NR	HLT	MI	RI	RO	IO	II	AI	AO
	Zatrzymanie pracy	Rejestr adresu IN	RAM IN	RAM OUT	Instrukcje OUT	Instrukcje IN	Rejestr A IN	Rejestr A OUT
000	0	1	0	0	0	0	0	0
001	0	0	0	1	0	1	0	0

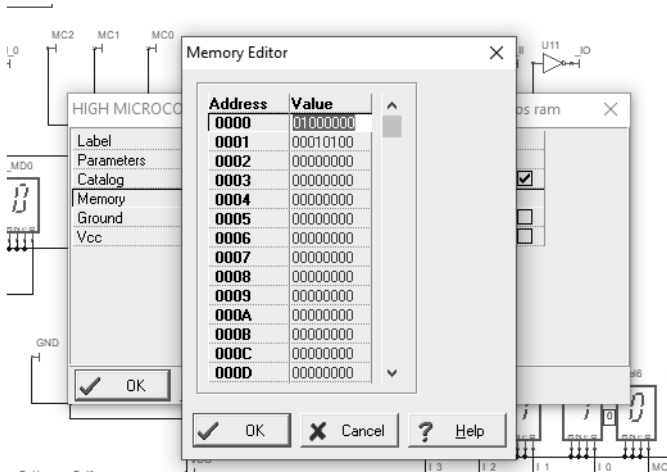
Tab. Pamięć wysoka

Dla **pamięci niskiej** układ mikroprogramu jest z kolei następujący:

NR	EO	SU	BI	OI	CE	CO	J	/
	ALU OUT	Odejmowanie	Rejestr B IN	Pusty	Counter Enable	Counter Out	Jump	/
000	0	0	0	0	0	1	0	0
001	0	0	0	0	1	0	0	0

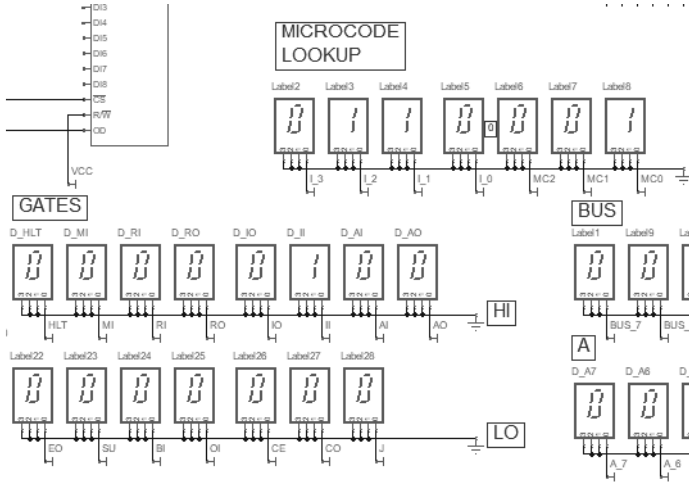
Tab. Pamięć niska

Jako całość, wygląda to następująco (pamięć wysoka):



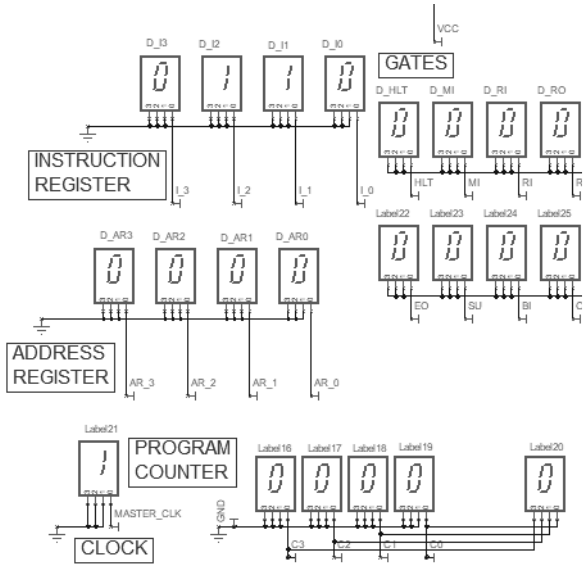
Rys. Sekwencja inicjalizująca komputera (mikrorozkazy)

Aby **wykonać** rozkaz LDA konieczne jest wykonanie **9 zmian bramek sterujących**. Dla operacji ADD jest to 11 zmian bramek. Zatrzymanie pracy komputera to jedna bramka.



Rys. Panel kontrolny

W wyżej opisany sposób można zrealizować póki co **większość podstawowych operacji przeniesienia danych z i do pamięci oraz wykonania odejmowania czy dodawania**. Jeśli mamy możliwość wykonać takie operacje, wówczas można pokusić się o stwierdzenie, że można zapewne dużo więcej, np. mnożenie czy dzielenie, nawet bez dodatkowej elektroniki.



Rys. Podgląd zawartości rejestrów i liczników

Na tym etapie wykonanie sekwencji początkowej oraz instrukcji LDA 14 daje efekt w postaci wartości z komórki nr 14 przeniesionej do rejestru A, co widać na powyższej ilustracji.

Instrukcja skoku

Dla instrukcji wykonującej **bezwarunkowy skok** (rozkaz JMP=0110) do innego obszaru pamięci, aby tam przenieść wykonywanie kodu programu, należy ustawić następujące parametry w pamięci **mikrokodu**:

INSTRUKCJA	ADRES (HEX)	PAMIĘĆ WYSOKA	PAMIĘĆ NISKA
JMP	0060	01001000	00000010
JMP	0061	00000100	00000000

Tab. Instrukcja skoku bezwarunkowego

Zestaw instrukcji

Dla **pozostałych instrukcji** (LDA=0001, ADD=0010, HLT=1111) postaci mikroprogramu są następujące:

INSTRUKCJA	ADRES (HEX)	PAMIĘĆ WYSOKA	PAMIĘĆ NISKA
LDA	0010	01000000	00000100
LDA	0011	00010100	00001000
LDA	0012	01001000	00000000
LDA	0013	00010010	00000000

INSTRUKCJA	ADRES (HEX)	PAMIĘĆ WYSOKA	PAMIĘĆ NISKA
ADD	0020	01000000	00000100
ADD	0021	00010100	00001000
ADD	0022	01001000	00000000
ADD	0023	00010000	00100000
ADD	0024	00000010	10000000

INSTRUKCJA	ADRES (HEX)	PAMIĘĆ WYSOKA	PAMIĘĆ NISKA
HLT	00F0	10000000	00000000

Tab. Mikrosterowanie pozostałych operacji

Przykładowy program

Poniższy program zawiera wszystkie dotychczasowe instrukcje. Stosowanie instrukcji JMP powoduje, że instrukcja HLT z komórki 000D nigdy nie zostanie wykonana.

> *Uwaga: Jest szczególna możliwość, że tak się zdarzy, ale tylko wtedy gdy **zdestabilizujemy pracę komputera**, np. poprzez podanie **zbyt wysokiej częstotliwości** pracy. Poszczególne komponenty mogą wtedy nie realizować oczekiwanych zadań w **oczekiwanym czasie**, stąd też wysterowanie bramek może być inne niż właściwe dla podanych w programie instrukcji.*

ADRES (BIN)	PROGRAM	OPIS
0000	0000 0000	NOP (Inicjalizacja)
0001	1110 0001	LDA 14
0002	1111 0010	ADD 15
0003	*0000 0110*	*JMP 0000*
000D	0000 1111	HLT
000E	0001 1100	28
000F	0000 1110	14

Tab. Przykładowy program

Ograniczenia

Wyżej przedstawiona maszyna to komputer o **8 bitowej szynie danych** oraz **4 bitowym adresowaniu**. W związku z czym, możliwe do stosowania jest jedynie **16 bajtów pamięci operacyjnej**. Układ, który dostępny jest w symulatorze to pamięć 11 bitowa z 8 bitowym słowem. Oznacza to, że można przechować w niej 16 384 bity, tj. 2048 słów 8 bitowych. Aby dostosować uproszoną formę komputera wyżej opisanego należy **przebudować rejestr instrukcji, licznik programu oraz rejestr pamięci**. Na samym końcu należałoby dostosować odpowiednio mikroprogram. Stosując 8 bitów i tak maksymalna ilość pamięci byłaby niezbyt wystarczająca do wygodnej pracy przy programowaniu, tj. dostępnych byłoby zaledwie 256 adresów.

Uwaga: Przykładowo komputer taki jak Commodore 64 używa 16 bitowego adresowania, dzięki czemu można teoretycznie uzyskać dostęp aż do 65 536 bajtów. Faktycznie jest to mniej, gdyż w przestrzeni adresowej był również KERNAL, BASIC i mapa znakowa.

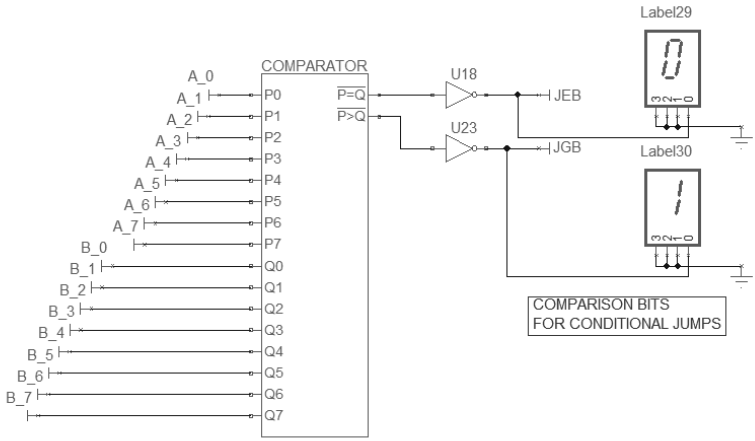
Skok warunkowy

Ostatnim elementem proponowanego tutaj komputera jest **instrukcja skoku warunkowego**. Jest ona niezbędna, żeby móc rzeczywiście pisać programy, rozwiązujące codzienne (i niecodzienne także) problemy. Instrukcja ma kod operacji 0011. Adresowanie zatem zaczyna się od 0011 0000, czyli 30 w notacji szesnastkowej.

INSTRUKCJA	ADRES (HEX)	PAMIĘĆ WYSOKA	PAMIĘĆ NISKA
JE	0030	01001000	00010000
JE	0031	00000100	00000000

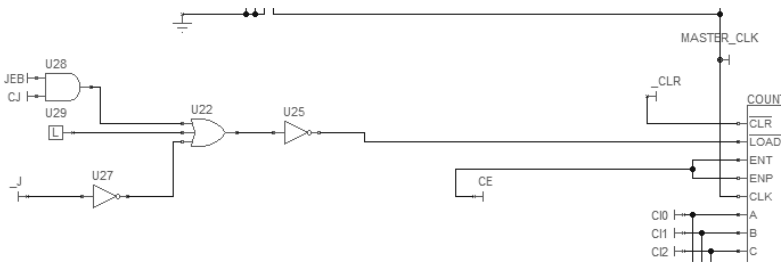
Tab. Instrukcja skoku warunkowego

Układ elektroniczny jaki umożliwia realizację tej instrukcji to **komparator 74LS682**, który na **wejściu przyjmuje dwie liczby 8 bitowe i zwraca stan wysoki jeśli są równe**, na drugim złączu raportując natomiast stan wysoki jeśli P>Q. W naszym przypadku P to rejestr A, a Q to rejestr B.



Rys. Komparator 74LS682

Druga zmiana to wejścia **_J w liczniku programu**. Składa się ono teraz z **kilku bramek**, które stwierdzają, że skok wykonamy jeśli jest on bezwarunkowy lub jeśli ustawiona jest **flaga skoku warunkowego (JC) i warunek jest spełniony (JEB)**.



Rys. Logika skoku warunkowego

Na potrzebę mikroprogramu, **złącze CJ to zmapowany 5 bit pamięci niskiej** (wcześniej pusty). Wykonując program, zmodyfikowany podaniem instrukcji skoku warunkowego zamiast bezwarunkowego, **skok ten nastąpi tylko jeśli rejestr są sobie równe, A=B, tj. P=Q**.

> PODSUMOWANIE

Poprzednie moje książki dotyczyły wyłącznie oprogramowania. Ta książka zmienia ten trend. Dotychczas poszukiwałem rozwiązań na problemy związane z budową programów, ich wdrażaniem lub testowaniem. Rozpoczynając serię mam nadzieje, że w niedługich odstępach uda się wydawać kolejne jej części. Tak aby wrócić względnie szybko do tematu oprogramowania, do którego jest mi jednak najbliżej. Tutaj postanowiłem spróbować swoich sił w tematyce budowy sprzętu elektronicznego. Jest to bardzo ciekawy obszar, który dalej mnie zadziwia. W szczególności technologia wytwarzania ekstremalnie małych komponentów sprawia wrażenie wręcz magicznej. Odnośnie kolejnych części, to będzie to również podróż po historii komputerów i przetwarzania informacji.

Bibliografia

Podstawowa

Jest to główna literatura, na której bazowana jest cała część teoretyczna z przykładami. Stanowi przygotowanie do rozdziału dotyczącego budowy minimalnej cyfrowej maszyny obliczeniowej.

Książki

[001] „Układy scalone w pytaniach i odpowiedziach", praca zbiorowa, WNT, 1987

[002] „Podstawy i praktyka programowania mikroprocesorów", Grabowski, Koślacz, WNT, 1980

[003] „Mały poradnik mechanika", praca zbiorowa, WNT, 1976

[004] „Encyklopedia techniki — podstawy techniki", praca zbiorowa, WNT, 1974

[005] „Organizacja maszyn cyfrowych w ujęciu strukturalnym", A. Tanenbaum, WNT, 1980

[006] „An Introduction to Integrated Circuits", praca zbiorowa, Control Data Corporation, 1967

[007] „Handbook of Analog Computation", praca zbiorowa, Electronic Associates at Princeton, 1967

Wykłady

[101] „Elektronika", S. Niespodziany, Politechnika Warszawska

[102] „Układy cyfrowe", praca zbiorowa, Politechnika Łódzka

Internet

[201] „History of Transformers and Inductors", Michelle Farnum, cettechnology.com

[202] „Kurs elektroniki", D. Szymański, forbot.pl

[203] „LM311 Differential Comparator IC", praca zbiorowa, microcontrollerslab.com

[204] „Prąd", A. Szulc, teoriaelektryki.pl

[205] „Build an 8-bit computer from scratch", B. Eater, eater.net

[206] „Capacitors”, praca zbiorowa, Engineering and Technology History Wiki, ethw.org

[207] „Who invented the diode?”, D. Laws, Computer History Museum, computerhistory.org

[208] „The History of the Transistor”, T. Watkins, San José State University, sjsu.edu

[209] „Electronics Tutorials”, praca zbiorowa, electronic-tutorials.ws

[210] „555 timer teardown: inside the world’s most popular IC”, Ken Shirriff, righto.com

[211] „Intel® 2102 SRAM Memory die”, Intel Museum Archive, calisphere.org

[212] „4004 die shot”, Intel Corporation, wikichip.org

[W] „Wikipedia”, praca zbiorowa, wikipedia.org

Inne

[301] „Catalog 1977”, praca zbiorowa, NEC Microcomputers Inc.

Dodatkowa

Jest to uzupełniająca lektura względem podstawowej. Nie jest bezpośrednio cytowana, ale była studiowana równolegle. Należy ją traktować jako wprowadzenie lub dopełnienie do treści.

[401] „Programowanie w języku assembler”, S. Kruk, PLJ, 1992

[402] „Konstrukcja kompilatorów”, Waite, Goos, WNT, 1984

[402] „Simplicity is Complex”, H. Kopetz, Springer, 2019

[403] „Systemy mikrokomputerowe”, P. Misiurewicz, WSP, 1982

[404] „Projektowanie urządzeń cyfrowych”, F. Wagner, WNT, 1978

[405] „Arytmetyka maszyn cyfrowych”, I. Flores, WNT, 1963

[406] „Konstruowanie urządzeń elektronicznych”, praca zbiorowa, WNT, 1975

[407] „Modułowe systemy mikrokomputerowe”, praca zbiorowa, WNT, 1984

[408] „Układy scalone analogowe i cyfrowe”, Millman, Halkias, WNT, 1976

[409] „Architektura komputerów”, J. Biernat, Politechnika Wrocławska, 2002

[410] „Półprzewodnikowe układy logiczne”, Misiurewicz, Grzybek, WNT, 1975

[411] „Pamięci maszyn cyfrowych”, Nowak, Sawicki, WNT, 1977

[412] „Mikroprocesory poradnik”, S. A. Money, WKŁ, 1990

[413] „Fizyczne podstawy mikroelektroniki”, G. I. Jepifanow, WNT, 1976