Układy mikroprocesorowe. Przykłady rozwiązań

Autor: Bartłomiej Zieliński
ISBN: 83-7197-702-6
Format: B5, stron: 130

Książka prezentuje podstawy konstrukcji urządzeń cyfrowych i mikroprocesorowych. Zawiera ona omówienia wybranych układów scalonych małej, średniej i dużej skali integracji oraz liczne przykłady ich zastosowań. Pewne zdziwienie czytelnika może wprawdzie budzić dobór mikroprocesorów do ilustracji niektórych zagadnień (są to mikroprocesory 8-bitowe Z-80, 8048 i 8051). Jeżeli jednak głębiej przyjrzeć się współczesnym układom mikroprocesorowym, okazuje się, że podstawowe techniki konstrukcyjne – a takie właśnie prezentuje książka – mimo upływu 20 lat, pozostały niezmienione.

W poszczególnych rozdziałach znajdziesz:

- podstawowe właściwości elektryczne układów cyfrowych oraz zasady ich stosowania;
- funkcje wybranych cyfrowych układów scalonych małej i średniej skali integracji oraz różnorodne przykłady ich zastosowań;
- wyprowadzenia wybranych mikroprocesorów 8-bitowych (Z-80, 8048, 8051) oraz zasady konstrukcji jednostki centralnej z ich wykorzystaniem;
- pamięci stałe (ROM, PROM, EPROM, EEPROM) oraz statyczne (SRAM) oraz zasady tworzenia bloków pamięci o zadanej organizacji;
- pamięci dynamiczne (DRAM) oraz zasady tworzenia bloków pamięci dynamicznej;
- programowalne układy równoległego wejścia-wyjścia (8255, Z-80 PIO) oraz ich zastosowanie do sterowania klawiaturą i wyświetlaczem;
- programowalne układy czasowo-licznikowe (8253, Z-80 CTC) oraz ich zastosowanie do pomiaru zależności czasowych;
- zasady tworzenia złożonych układów wejścia-wyjścia oraz dołączania ich do różnych typów mikroprocesorów.

Książka przeznaczona jest dla studentów kierunków takich, jak informatyka, elektronika czy automatyka. Może być również wykorzystana przez uczniów techników elektronicznych, a także przez wszystkich zainteresowanych podstawami konstrukcji sprzętu komputerowego.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Rozdział 6.

Układy równoległego wejścia-wyjścia

Jednostka centralna wyposażona w pamięci programu i danych nie tworzy jeszcze pełnego systemu mikroprocesorowego. Każdy system musi bowiem komunikować się z otoczeniem. Komunikacja ta może być prowadzona szeregowo lub równolegle. Transmisja szeregowa, w której informacja przesyłana jest szeregowo bit po bicie, stosowana jest najczęściej do łączenia kilku lub więcej układów w celu uzyskania przetwarzania rozproszonego. Natomiast transmisja równoległa, polegająca na przesyłaniu od razu całych bajtów, pozwala m.in. na wyposażenie systemu mikroprocesorowego w układ komunikacji z użytkownikiem. W najprostszym przypadku układ taki zawiera klawiaturę i zespół wyświetlaczy 7-segmentowych.

Programowalne układy wejścia-wyjścia

Układ równoległego wejścia-wyjścia można zbudować wykorzystując układy TTL, takie jak rejestry i bufory, także z wyjściami trójstanowymi. Obsługa takiego układu wymaga jednak programowego generowania i sprawdzania niezbędnych sygnałów sterujących. Oczywiście jest to rozwiązanie najprostsze sprzętowo, ale pochłaniające cenny czas mikroprocesora. Ponadto złożony moduł może zawierać nawet kilkanaście układów scalonych. Zamiast nich można wykorzystać specjalizowane, programowalne układy dużej skali integracji, specjalnie przeznaczone do stosowania w modułach równoległego wejścia-wyjścia. Układy takie pozwalają nie tylko na sprzętową realizację sygnałów sterujących, lecz potrafią także wykonywać szereg dodatkowych czynności, takich jak np. automatyczne wykrywanie zmiany stanu określonych sygnałów. Nie trzeba chyba dodawać, jak wpływa to na odciążenie mikroprocesora.

Na kolejnym etapie rozwoju układów scalonych znajdują się specjalizowane sterowniki określonych urządzeń, np. stacji dyskietek, dysków twardych, monitorów itp. Układy te wykraczają poza ramy niniejszej książki i nie będą tu omawiane.

Układ 8255

Układ 8255 firmy Intel jest programowalnym układem równoległego wejścia-wyjścia. Jest on wyposażony w dwie grupy wyprowadzeń, służących do:

- dołączenia układu do magistrali systemowej,
- sterowania urządzeniami zewnętrznymi.

Interfejs systemowy układu zawiera następujące sygnały:

- $D_0 – D_7$ — dwukierunkowa, 8-bitowa magistrala danych z wyjściami trójstanowymi;
- $A_0 – A_1$ — wejścia adresowe układu, służące do wyboru jego rejestrów wewnętrznych;
- $\overline{CS}$ — wejście uaktywnienia układu, aktywne w stanie niskim;
- $\overline{RD}$ — wejście żądania odczytu informacji z układu, aktywne w stanie niskim;
- $\overline{WR}$ — wejście żądania zapisu informacji do układu, aktywne w stanie niskim;
- RES — wejście zerowania układu, aktywne w stanie wysokim.

Interfejs urządzeń zewnętrznych zawiera trzy dwukierunkowe, 8-bitowe porty wejścia-wyjścia: A, B i C. Porty mogą pracować w kilku trybach, przy czym pewne tryby jednych portów wymuszają użycie pozostałych portów w określonych trybach.

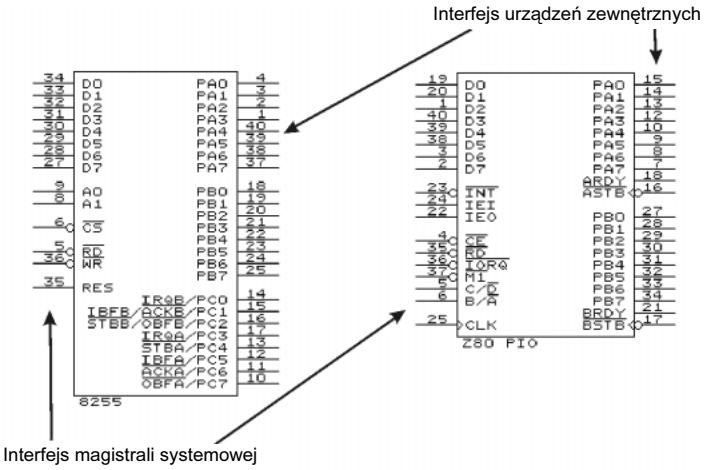
W trybie „0” — prostego wejścia-wyjścia (bez potwierdzenia) — może pracować dowolny port. Podczas programowania trybu pracy należy ustalić m.in. kierunek transmisji obowiązujący dla całego portu, przy czym port C może być podzielony na dwie połowy, dla których kierunek ten można ustalić indywidualnie. Układ zapamiętuje dane wyjściowe, natomiast dane wejściowe — nie. Odczyt informacji polega zatem na odczytaniu aktualnego stanu wyprowadzeń portu.

W trybie „1” — jednokierunkowym z potwierdzeniem — mogą pracować porty A i B. Sygnały sterujące dla portów w tym trybie zapewnia port C, tak więc niektórych jego bitów nie można wówczas swobodnie wykorzystać. Tryb i kierunek transmisji (wejście lub wyjście) można ustalić indywidualnie dla obu portów. Dane wejściowe i wyjściowe są zapamiętywane w rejestrach układu.

Port A jest jedynym, który może pracować w trybie „2” — dwukierunkowym z potwierdzeniem. Port B może wówczas pracować w dowolnym trybie, natomiast port C zapewnia sygnały sterujące, podobnie jak w trybie „1”. Dane wejściowe i wyjściowe są zapamiętywane w rejestrach układu.

Znaczenie sygnałów sterujących w trybach „1” i „2” jest następujące:

- IRQ — sygnał zgłoszenia przerwania dla mikroprocesorów (aktywny w stanie wysokim);
- $\overline{\text{STB}}$ — wejście wpisujące dane z urządzenia zewnętrznego do portu (stan aktywny — niski);
- IBF (ang. Input Buffer Full) — zajętość odbiornika; stan aktywny (wysoki) oznacza, że mikroprocesor nie odczytał jeszcze danych wpisanych przez urządzenie;
- $\overline{\text{OBF}}$ (ang. Output Buffer Full) — zajętość nadajnika; stan aktywny (niski) oznacza, że urządzenie może pobrać z portu informację wpisaną przez mikroprocesor;
- $\overline{\text{ACK}}$ — wejście potwierdzające odebranie informacji przez urządzenie (stan aktywny — niski).



Rysunek 6.1. *Wyprowadzenia układów 8255 i Z-80 PIO*

Układ Z-80 PIO

Układ Z-80 PIO firmy Zilog jest programowalnym układem równoległego wejścia-wyjścia. Podobnie jak 8255 jest on wyposażony w dwie grupy wyprowadzeń, służących do:

- dołączenia układu do magistrali systemowej,
- sterowania urządzeniami zewnętrznymi.

Interfejs systemowy układu zawiera następujące sygnały:

- $D_0 – D_7$ — dwukierunkowa, 8-bitowa magistrala danych z wyjściami trójstanowymi;
- $B / \overline{A}$ — wejście adresowe układu, służące do wyboru rejestrów wewnętrznych portu A lub B;

- $C/\overline{D}$ — wejście adresowe układu, służące do wyboru rejestru sterującego lub danych portu;
- $\overline{CE}$ — wejście uaktywnienia układu, aktywne w stanie niskim;
- $\overline{M1}$ — wejście sygnału $\overline{M1}$ z mikroprocesora;
- $\overline{IORQ}$ — wejście sygnału $\overline{IORQ}$ z mikroprocesora;
- $\overline{RD}$ — wejście żądania odczytu informacji z układu, aktywne w stanie niskim;
- IEI — wejście zezwolenia na generację przerwania;
- IEO — wyjście zezwolenia na generację przerwania;
- $\overline{INT}$ — wyjście zgłoszenia przerwania, aktywne stanem niskim;
- CLK — wejście zegarowe.

Interfejs urządzeń zewnętrznych zawiera dwa dwukierunkowe, 8-bitowe porty wejścia-wyjścia: A i B, z których każdy jest wyposażony w dwa sygnały sterujące (wyjście RDY i wejście $\overline{STB}$). Porty mogą pracować w kilku trybach. Znaczenie sygnałów sterujących jest zależne od trybu pracy portu.

Tryb „0" jest trybem wyjściowym z potwierdzeniem. Wpisanie danych do rejestru portu powoduje przejście sygnału RDY tego portu w stan aktywny (poziom wysoki). Sygnał ten pozostaje aktywny do chwili pojawienia się narastającego zbocza sygnału $\overline{STB}$, po czym jest zerowany. Jeżeli sygnał $\overline{STB}$ nie pojawił się, to RDY zostanie automatycznie wyzerowane na początku kolejnego zapisu danych. Dzięki temu podczas operacji zapisu RDY jest zawsze w stanie niskim (nieaktywnym).

Bezpośrednie połączenie $\overline{STB}$ i RDY powoduje wygenerowanie impulsu o czasie trwania równym okresowi zegara, podawanego na wejście CLK. Nie jest wówczas generowane przerwanie.

Tryb „1" jest trybem wejściowym z potwierdzeniem. Otoczenie zapisuje informację do rejestru portu danych za pomocą sygnału $\overline{STB}$. Zbocze narastające tego sygnału powoduje zgłoszenie przerwania oraz przejście wyjścia RDY w stan niski (brak gotowości, czyli bufor wejściowy zapełniony). Narastające zbocze sygnału $\overline{RD}$ powoduje przejście RDY w stan wysoki.

Tryb „2" jest trybem dwukierunkowym z potwierdzeniem, przy czym może w nim pracować tylko port A. Sygnały sterujące portu B zabiera się wówczas na potrzeby portu A, tak więc port B może pracować tylko w trybie „3" — bitowym. Sygnały sterujące portu A używane są w operacjach wyjściowych, a portu B – w wejściowych. Poza tym zasady stosowania obu par sygnałów są takie same jak w trybach „1" i „2".

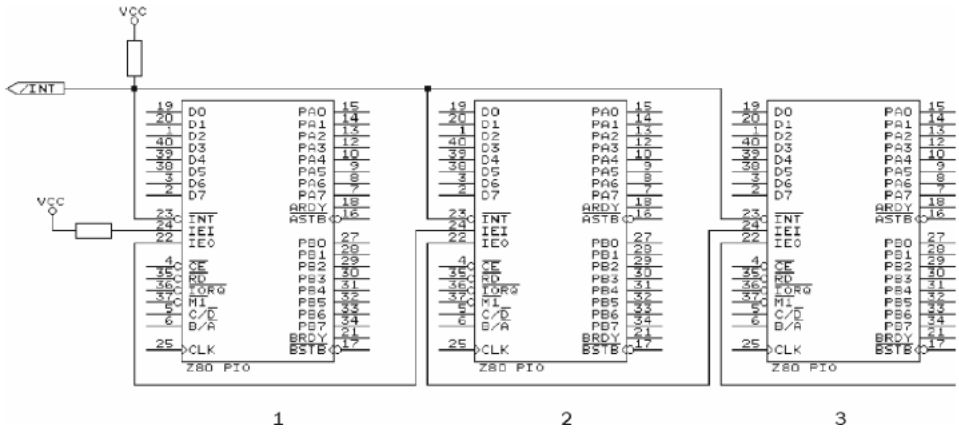
Tryb „3" — tzw. bitowy — nie wymaga używania sygnałów sterujących. W trybie tym można m.in. automatycznie sprawdzać zadane warunki logiczne. Spełnienie

takiego warunku (przejście wartości wyniku funkcji logicznej z „0” na „1”) może spowodować generację przerwania. Tryb ten można zatem stosować do sterowania obiektem, unikając konieczności ciągłego programowego sprawdzania stanu poszczególnych sygnałów.

Nieco większego komentarza wymaga układ przerwań, stosowany w układach mikroprocesorowych rodziny Z-80. W pewnym sensie można powiedzieć, że układy te realizują koncepcję „rozproszonego sterownika przerwań”. Nie ma tu bowiem potrzeby stosowania specjalnego układu pełniącego tę funkcję.

Określenie priorytetów przerwań pochodzących od poszczególnych układów odbywa się już na etapie projektowania systemu mikroprocesorowego. Każdy układ rodziny Z-80 jest wyposażony w wejście IEI oraz wyjście IEO. Służą one do kaskadowego połączenia układów w łańcuch priorytetów (ang. *daisy chain*[1]). Układ o najwyższym priorytecie ma wejście IEI stale ustawione na „1”, a jego wyjście IEO steruje wejściem IEI kolejnego układu. Wyjścia $\overline{\text{INT}}$ wszystkich układów są zwarte i dołączone do wejścia $\overline{\text{INT}}$ mikroprocesora Z-80.

Załóżmy, że układy nr 2 i 3 zgłaszają przerwanie. Mikroprocesor generuje cykl przyjęcia przerwania, który jest dekodowany przez wszystkie układy na podstawie stanu linii $\overline{\text{M1}}$ i $\overline{\text{IORQ}}$. Układ nr 1 może odpowiedzieć (jego wejście IEI ma stan wysoki), ale nie zgłaszał przerwania. Dlatego też jego wyjście IEO jest w stanie wysokim. Układ nr 2 może zatem odpowiedzieć i rzeczywiście to czyni, gdyż zgłaszał przerwanie. Jego wyjście IEO przyjmuje wówczas stan niski, zatem układ nr 3 i ewentualne kolejne układy nie mogą w tej chwili odpowiedzieć na przyjęcie przerwania. Układ nr 2 jest zatem jedynym, który odpowiedział mikroprocesorowi i wysłał mu wektor przerwania. Mikroprocesor przechodzi wobec tego do obsługi przerwania od układu nr 2.



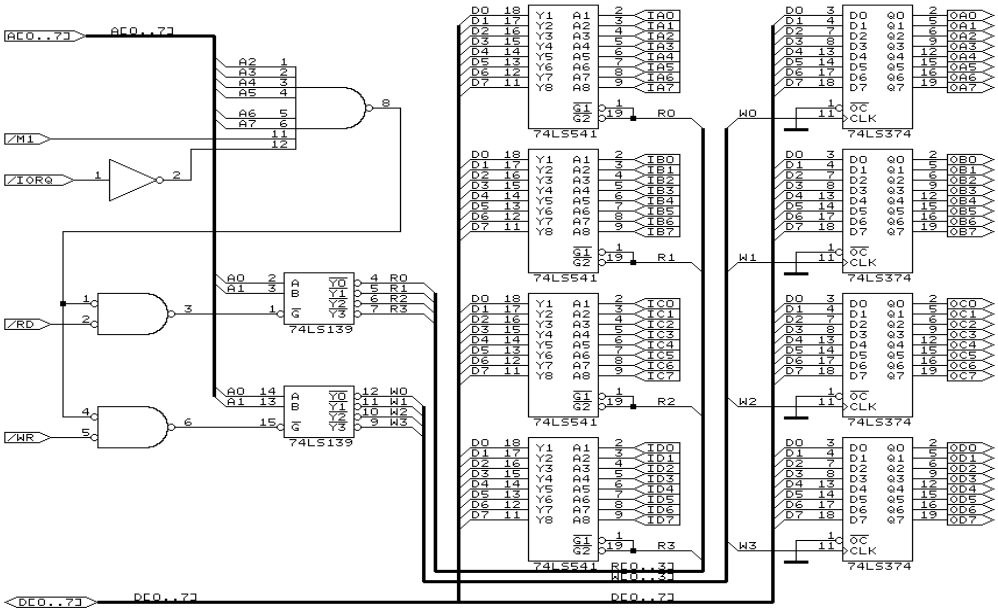
Rysunek 6.2. *Zasada działania przerwań w układach Z-80 PIO*

[1] W dosłownym tłumaczeniu — „łańcuch stokrotek”.

Przykłady

Przykład 1.

Zaprojektować prosty moduł wejścia-wyjścia, zawierający cztery rejestry wejściowe i cztery rejestry wyjściowe. Moduł powinien współpracować z mikroprocesorem Z-80 i zajmować adresy: a) FC – FF_h lub b) 3C – $3F_h$ w przestrzeni wejścia-wyjścia.



Rysunek 6.3. *Rozwiązanie przykładu 1a*

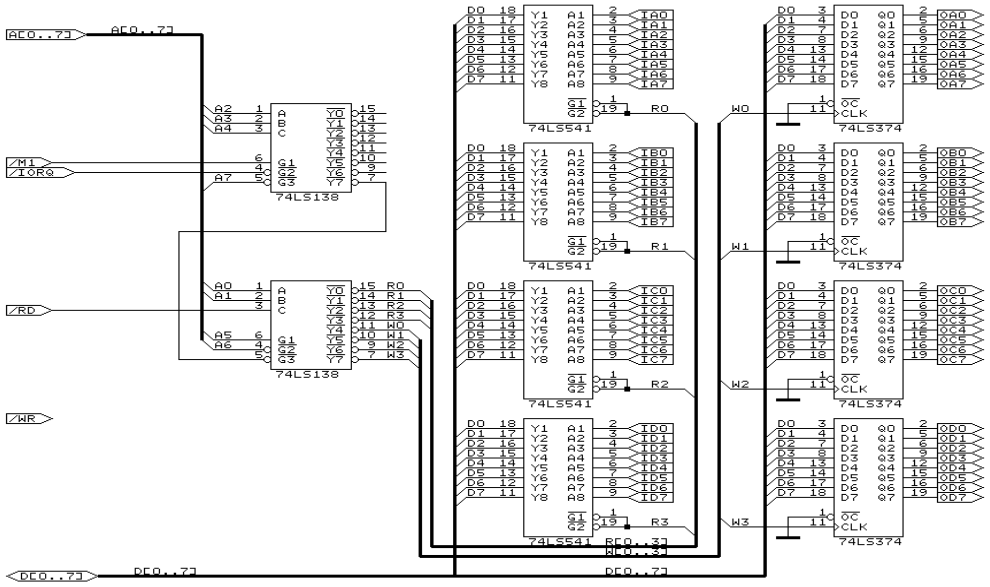
Skoro moduł ma zawierać w sumie osiem rejestrów (cztery wejściowe i cztery wyjściowe) i zajmować tylko cztery komórki w przestrzeni adresowej, to rejestry wyjściowe muszą współdzielić adresy z rejestrami wejściowymi. Od strony programowej wygląda to tak, jak gdyby pod jednym adresem był zarówno rejestr wejściowy, jak i wyjściowy. Natomiast w układzie cykle odczytu odwołują się fizycznie do innego układu, niż cykle zapisu.

Mikroprocesor Z-80 generuje 8-bitowe adresy w przestrzeni wejścia-wyjścia. Dwie najmłodsze linie adresowe służą do wyboru układu wewnątrz modułu, natomiast pozostałe — do sprawdzenia, czy adres na magistrali znajduje się w zakresie przeznaczonym dla modułu. Warunek ten należy sprawdzać, gdy sygnał $\overline{\text{IORQ}}$ jest w stanie niskim, co oznacza dostęp mikroprocesora do przestrzeni wejścia-wyjścia. Dodatkowo można sprawdzać stan sygnału $\overline{\text{M1}}$ — powinien być on w stanie wysokim. Upewniamy się w ten sposób, że mikroprocesor rzeczywiście zamierza przeprowadzić odczyt lub zapis układu wejścia-wyjścia, a nie zamierza realizować cyklu potwierdzenia przerwania, w którym oba te sygnały są aktywne.

Jeżeli adres jest rzeczywiście adresem modułu, to w zależności od realizowanego cyklu maszynowego uaktywnia się jeden z demultiplekserów 74139. Jeden z nich włącza układy wyjściowe, drugi natomiast — wejściowe. Wybór konkretnego układu odbywa się na podstawie stanu dwóch najmłodszych linii adresowych mikroprocesora.

Układy wyjściowe to oczywiście rejestry — zastosowano tu układy 74374, które zapamiętują informację w chwili wystąpienia zbocza narastającego sygnału wpisującego CLK. Umożliwia to sterowanie wejść sygnałami aktywnymi w stanie niskim, ponieważ mikroprocesor Z-80 utrzymuje dane na magistrali jeszcze przez pewien krótki czas po przejściu sygnału $\overline{RD}$ lub $\overline{WR}$ w stan wysoki. Czas ten wystarcza do zapamiętania informacji przez układ 74374.

Układy wyjściowe to bufory z wyjściami trójstanowymi. Dlaczego bufory? Otóż przedstawiony układ nie wykorzystuje jakichkolwiek sygnałów sterujących ze strony otoczenia systemu. Nie ma możliwości zapamiętania danych wejściowych w rejestrze, bo nie można określić momentu, w którym powinno to nastąpić. Dlatego też jedynym sposobem odczytania informacji z otoczenia jest wczytanie danych znajdujących się na wejściach modułu w chwili realizowania cyklu odczytu przez mikroprocesor. Zamiast buforów można też zastosować rejestry 74573, przy czym ich wejścia wpisujące (C) powinny być w stanie wysokim. W zasadzie układy te pełnią wówczas funkcję buforów.



Rysunek 6.4. *Rozwiązanie przykładu 1b*

Dekoder adresów w wersji b) można zrealizować wykorzystując dwa demultipleksery 74138. Pierwszy z nich sprawdza linie adresowe $A_2 - A_4$ oraz A_7, a także stan sygnałów sterujących. Układ jest aktywny, gdy mikroprocesor przeprowadza odczyt lub zapis przestrzeni adresowej wejścia-wyjścia przy A_7 = „0”. Wówczas, jeżeli wyjścia

$A_2 – A_4$ także są w stanie wysokim, wyjście Y7 przyjmuje stan niski. Powoduje to uaktywnienie drugiego układu 74138, o ile A_5 = „1” i A_6 = „0”. Wówczas wybrane wyjście tego układu przyjmuje stan niski. Wybór wyjścia odbywa się na podstawie stanu dwu najmłodszych linii adresowych oraz sygnału $\overline{RD}$. Gdy mikroprocesor realizuje odczyt, wybiera się jedno z wyjść $Y0 – Y3$, gdy zapis — $Y4 – Y7$.

Włączenie drugiego demultipleksera następuje, gdy określone wyjście pierwszego jest aktywne. Gdyby zatem wprowadzić możliwość wyboru wyjścia, można by było ustawiać moduł w jednej z ośmiu przestrzeni adresowych, zależnie od numeru wyjścia pierwszego demultipleksera, które steruje wejściem bramkującym drugiego układu.

Warto zadać sobie pytanie, czy pominięcie sygnału zapisu nie spowoduje niepożądanego uaktywniania się układów wyjściowych. Mikroprocesor Z-80 odwołuje się do przestrzeni adresowej wejścia-wyjścia tylko w trzech cyklach maszynowych: odczytu, zapisu i potwierdzenia przerwania. Ostatni z nich można wykluczyć, ponieważ pierwszy demultiplekser 74138 jest wówczas nieaktywny. Ponieważ realizowany jest zawsze albo odczyt, albo zapis, wystarczy sprawdzić tylko jeden sygnał — w tym przypadku $\overline{RD}$. Jeżeli bowiem cykl maszynowy nie jest cyklem odczytu, to na pewno jest to zapis.

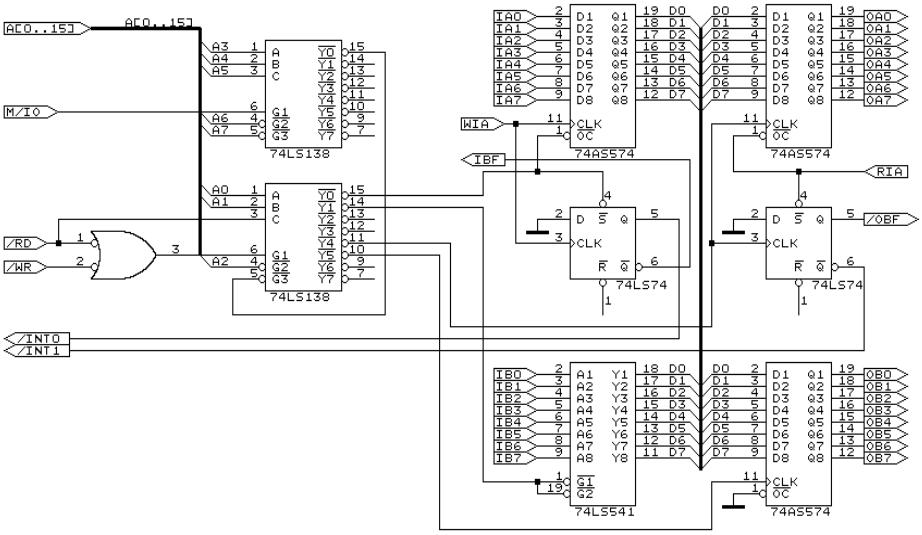
Pozostała część modułu pozostaje bez zmian.

Przykład 2.

Zaprojektować system wejścia-wyjścia, realizujący zapamiętywanie informacji wejściowej i wyjściowej. Układ powinien informować mikroprocesor o zapisaniu danych wejściowych, a otoczenie o ich odczytaniu oraz informować otoczenie o zapisaniu nowej informacji, a mikroprocesor — o jej odczytaniu. Ponadto moduł powinien zapewniać możliwość przesyłu danych bez potwierdzenia. Moduł powinien współpracować z mikroprocesorem 8051 i zajmować adresy $00 – 01_h$ w przestrzeni zewnętrznej pamięci danych. W zadaniu należy również umożliwić późniejsze dołączenie 64 KB zewnętrznej pamięci danych.

Aby móc później dołączyć 64 KB pamięci danych, należy wprowadzić do układu sygnał, umożliwiający określenie, czy cykl odczytu lub zapisu zewnętrznej pamięci danych ma się odnosić do pamięci, czy też do przedstawionego modułu. Sygnał ten jest na schemacie oznaczony jako M/IO; jego stan wysoki oznacza odwołanie do modułu, niski zaś — do pamięci. W ramach modułu nie określono sposobu jego wytwarzania. Przykładowy sposób rozwiązania to użycie wolnej linii któregoś z portów mikroprocesora 8051. Wówczas sygnał ten można łatwo zmieniać programowo.

Przyjęto, że podczas odwołań do rejestrów modułu mikroprocesor generuje adres 8-bitowy. Adres może być też 16-bitowy, jednak linie $A_8 – A_{15}$ nie są uwzględnione w dekoderze adresów. Dekoder uaktywnia się, gdy bity $A_2 – A_7$ są w stanie niskim, sygnał M/IO — w wysokim, a mikroprocesor realizuje cykl odczytu lub zapisu zewnętrznej pamięci danych. Wówczas, zależnie od stanu linii A_0 i A_1 oraz sygnału $\overline{RD}$ uaktywnia się odpowiednie wyjście drugiego demultipleksera 74138, umożliwiając zapis lub odczyt żądanego rejestru.



Rysunek 6.5. *Rozwiązanie przykładu 2.*

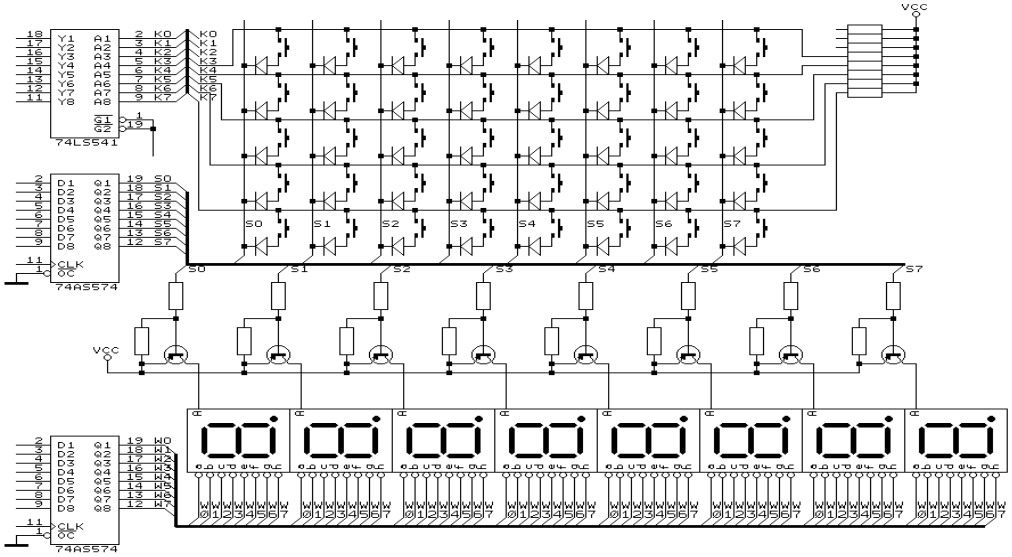
Pod adresem 01_h znajduje się zestaw układów wejścia-wyjścia, które umożliwiają transmisję bez potwierdzenia. Działają one tak samo, jak w przykładzie 1. — dane wyjściowe wpisywane są do rejestru 74574, a wejściowe wczytywane są bez zapamiętywania przez bufor 74541.

Pod adresem 00_h umieszczono zestaw rejestrów realizujących wejście-wyjście z potwierdzeniem. Przesyłane dane zawsze zapisuje się do odpowiedniego rejestru. Zapis pod adres 00_h powoduje wpisanie danej do układu wyjściowego 74574 oraz wpisanie „0” do przerzutnika 7474. Na skutek tego sygnał $\overline{\text{OBF}}$ poprzez stan niski informuje otoczenie, że w rejestrze wyjściowym są dane do pobrania. Aby je odczytać, otoczenie powinno ustawić sygnał $\overline{\text{RIA}}$ w stan niski. Wówczas na wyjściach $OA_0 – OA_7$ pojawia się zawartość rejestru wyjściowego, a do przerzutnika wpisuje się stan „1”. Powoduje to zgłoszenie przerwania $\overline{\text{INT1}}$ do mikroprocesora.

Otoczenie może także zapisać dane do układu wejściowego. W tym celu ustawia ono odpowiednio wejścia $IA_0 – IA_7$, których stan zapamiętuje się w rejestrze podczas narastającego zbocza sygnału WIA. Powoduje to wpisanie „0” do przerzutnika, który generuje do otoczenia sygnał IBF informujący o zapełnieniu rejestru wejściowego oraz zgłasza przerwanie $\overline{\text{INT0}}$ do mikroprocesora. Jeżeli w procedurze obsługi przerwania mikroprocesor odczyta zawartość rejestru, to do przerzutnika wpisze się „1”, co spowoduje przejście sygnału IBF w stan niski. Oznacza to, że otoczenie może zapisać kolejny bajt danych.

Przykład 3.

Zaprojektować moduł wyświetlania multipleksowanego i klawiatury matrycowej, zawierający 64 klawisze i 8 wyświetlaczy. Wykorzystać dwa rejestry wyjściowe i jeden wejściowy.



Rysunek 6.6. *Rozwiązanie przykładu 3.*

Jeden rejestr zawiera adres załączonego wyświetlacza, drugi — kod wyświetlanej cyfry. Kod ten podaje się bezpośrednio na wybrany wyświetlacz, zatem można włączyć dowolną kombinację jego segmentów. Zmiana kodu BCD lub dwójkowego na kod wyświetlacza odbywa się wyłącznie na drodze programowej. W rejestrze wyboru aktywnego wyświetlacza wszystkie bity — prócz jednego — mają stan „1”. Wysterowanie wspólnego wejścia segmentów wyświetlacza wymaga zastosowania wzmacniacza tranzystorowego, ponieważ prąd płynący przez to wejście jest zbyt duży dla wyjść układów cyfrowych. Jeżeli tranzystor jest dołączony do wyjścia będącego w stanie niskim, to napięcie baza-emiter wystarcza do wprowadzenia go w stan nasycenia, co powoduje załączenie wyświetlacza. Napięcie na kolektorze jest wówczas bliskie V_{CC}. Gdy natomiast wyjście rejestru jest w stanie wysokim, tranzystor znajduje się w stanie odcięcia (zatkania) i wyświetlacz jest wyłączony.

Stan niski na wybranym wyjściu rejestru można też użyć do wyboru sprawdzanej linii klawiatury. Jest to zatem klawiatura z „krążącym zerem”. Załóżmy, że wciśnięty klawisz znajduje się na przecięciu linii S_i i K_i, przy czym linia S_i jest wybrana — na odpowiadającym jej wyjściu rejestru jest stan niski. Stan ten przenosi się na linię K_i i może być odczytany przez bufor 74541. Znając numer linii S_i i K_i można jednoznacznie określić, który klawisz został wciśnięty.

Dołączenie diod do każdego klawisza gwarantuje poprawną pracę klawiatury. Gdyby diod nie było, mikroprocesor mógłby błędnie interpretować stan klawiatury. Załóżmy, że wciśnięto trzy klawisze, znajdujące się na przecięciu linii S_i i K_i, S_j i K_i oraz S_j i K_j. Pobudzenie linii S_i powoduje ustawienie linii K_i w stan niski. Stan ten następnie przenosi się przez drugi klawisz na linię S_j, a z niej — przez trzeci klawisz — na K_j. W rezultacie mikroprocesor odczytując stan klawiatury wczyta bajt, w którym dwa bity mają wartość „0". Poprawność lokalizacji wciśniętego klawisza zależy teraz od kolejności sprawdzania bitów słowa stanu. Jeżeli wcześniej sprawdzona będzie linia K_i, to mikroprocesor otrzyma parę S_i i K_i, a więc kod pierwszego klawisza. Gdyby natomiast najpierw sprawdził linię K_j, otrzyma parę S_i i K_j, oznaczającą klawisz, który nie był wciśnięty. Opisany efekt błędnego dekodowania klawiatury matrycowej można całkowicie usunąć przez włączenie diod w taki sposób, by stan aktywny linii przechodził tylko w kierunku $S \rightarrow K$. Kierunek włączenia diod zależy więc od tego, czy jest to klawiatura z „krążącym zerem", czy z „krążącą jedynką".

Przedstawiony układ jest dość prosty, wymaga jednak stosunkowo złożonej obsługi programowej. Do zadań mikroprocesora należą tu:

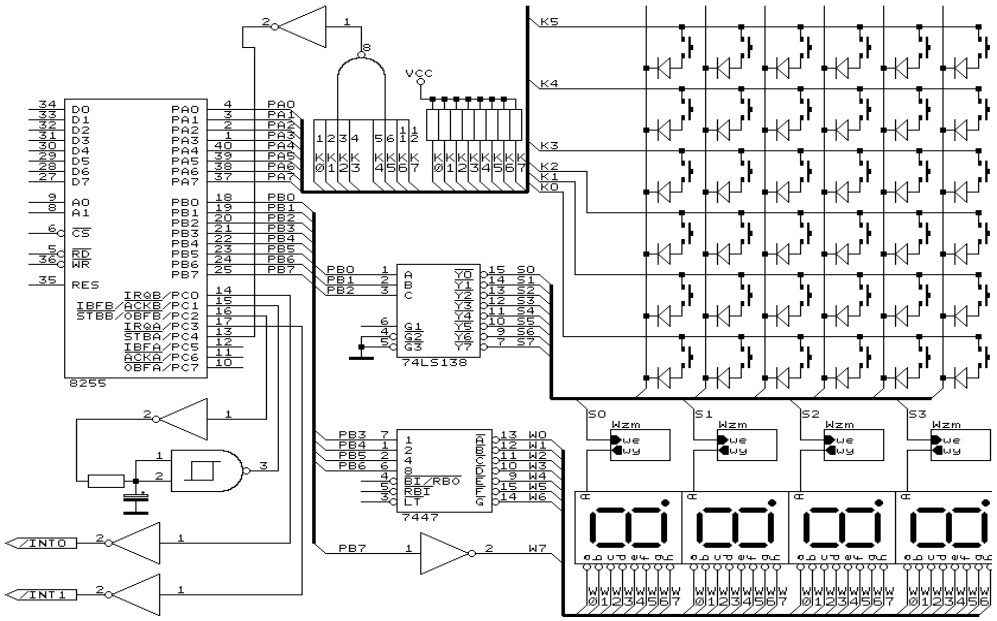
- zmiana kodu dwójkowego na kod „1 z *n*" przed zapisaniem do rejestru wyboru aktywnej linii;
- zmiana kodu BCD lub dwójkowego na kod wyświetlacza 7-segmentowego przed zapisaniem do rejestru wyświetlanego znaku;
- zmiana kodu „1 z *n*" na kod dwójkowy po odczytaniu stanu klawiatury;
- sprawdzanie, czy wciśnięto jakikolwiek klawisz;
- dotrzymywanie zadanej częstości przełączania wyświetlaczy i przeglądania klawiatury.

Wszystkie wymienione zadania można zrealizować sprzętowo, tj. za pomocą odpowiednich układów scalonych, także przy użyciu programowalnych układów równoległego wejścia-wyjścia.

Przykład 4.

Zaprojektować moduł wyświetlania multipleksowanego i klawiatury matrycowej, zawierający 64 klawisze i 8 wyświetlaczy. Alfabet wyświetlacza składa się wyłącznie z cyfr dziesiętnych w kodzie BCD. Wykorzystać układ 8255 pracujący w trybie z potwierdzeniami. Układ powinien współpracować z mikroprocesorem 8051 i zgłaszać dwa osobne przerwania:

- od klawiatury — gdy wykryto wciśnięcie klawisza,
- od wyświetlacza — gdy trzeba zmienić aktywny wyświetlacz.



Rysunek 6.7. *Rozwiązanie przykładu 4.*

Port PB układu 8255 pracuje jako wyjście. Linie PB_0 – PB_2 poprzez demultiplekser 74139 wybierają aktywny wyświetlacz i pobudzają klawiaturę. Wspólne wejścia wyświetlaczy steruje się za pośrednictwem wzmacniaczy tranzystorowych. Na wyjściach PB_3 – PB_6 podawany jest kod cyfry do wyświetlenia. Dodatkowo najstarszy bit portu steruje wyświetlaniem kropki dziesiętnej poprzez bramkę wzmacniającą.

Wpisaniu informacji do portu PB towarzyszy automatyczna aktywacja sygnału $\overline{\text{OBFB}}$. Powoduje to ładowanie kondensatora. Gdy kondensator się naładuje, bramka Schmitta wytwarza sygnał $\overline{\text{ACKB}}$, informujący o „zużyciu" danych portu PB i konieczności wpisania nowej informacji. Układ 8255 generuje sygnał przerwania od portu PB. Odpowiednio dobierając opóźnienie w układzie można zatem zmieniać częstość odświeżania wyświetlaczy.

Port PA układu pracuje jako wejście służące do wczytywania stanu klawiatury. Gdy układ wykryje wciśnięty klawisz, jedna z linii K_i przyjmuje stan niski, skutkiem czego wyjście 8-wejściowej bramki NAND przyjmuje stan wysoki. Na wejściu $\overline{\text{STBA}}$ pojawia się stan aktywny, a zawartość linii portu PA wpisuje się do rejestru wewnętrznego w układzie 8255. Po zapamiętaniu tej informacji układ automatycznie generuje sygnał przerwania od portu PA. Sygnał ten można wykorzystać w mikroprocesorze 8051 jako zgłoszenie przerwania od klawiatury.

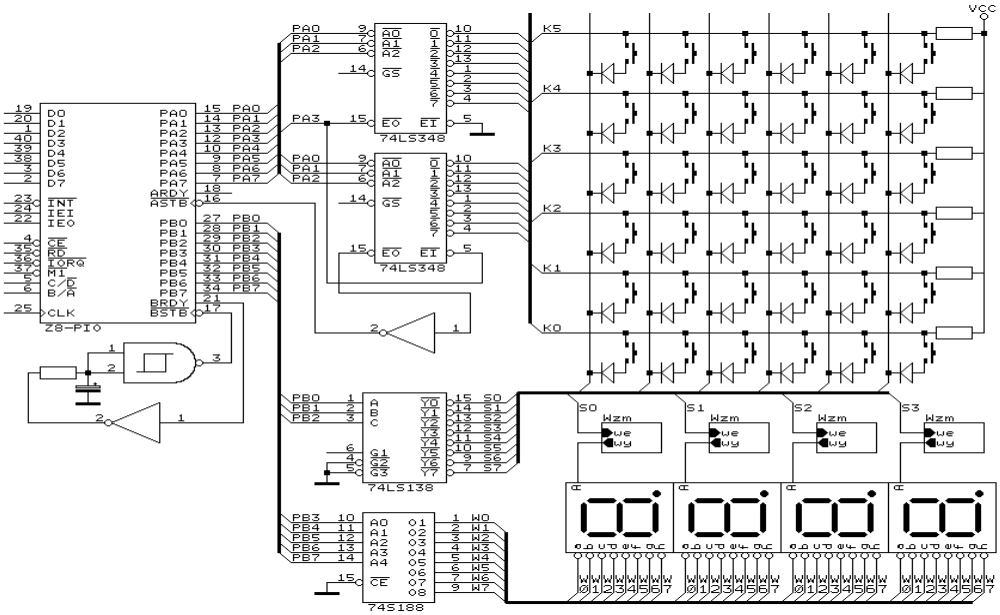
Dzięki sprzętowej realizacji wielu funkcji modułu mikroprocesor musi tylko zmieniać kod „1 z *n*" na kod dwójkowy po odczytaniu stanu klawiatury. Czynność ta jest wykonywana w procedurze obsługi przerwania klawiatury. Alternatywnie można

zakodować stan klawiatury używając kodera priorytetowego, np. układu 74148 lub 74348. Rozwiązanie takie jest konieczne, gdy liczba linii K_i przekracza liczbę wolnych linii wejściowych portu.

Przykład 5.

Zaprojektować moduł wyświetlania multipleksowanego i klawiatury matrycowej, zawierający 128 klawiszy i 8 wyświetlaczy. Alfabet wyświetlacza składa się z 32 znaków. Wykorzystać układ Z-80 PIO pracujący w trybie z potwierdzeniami. Układ powinien współpracować z mikroprocesorem Z-80 i zgłaszać dwa osobne przerwania:

- od klawiatury — gdy wykryto wciśnięcie klawisza,
- od wyświetlacza — gdy trzeba zmienić aktywny wyświetlacz.



Rysunek 6.8. *Rozwiązanie przykładu 5.*

Aby można było sterować ośmioma wyświetlaczami i pokazywać na nich do 32 różnych znaków, potrzeba 8 linii wyjściowych portu. Oczywiście przełączanie wyświetlaczy wymaga zewnętrznego dekodowania (układ 74138) i wykorzystuje 3 najmłodsze bity portu. Pozostałe bity podają kod znaku do wyświetlenia. Zmianą tego 5-bitowego kodu na kod wyświetlacza zajmuje się pamięć PROM (74S188) o organizacji 32×8. Jest to układ z wyjściami z otwartym kolektorem, a więc może sterować wyświetlaczami bezpośrednio, o ile prąd świecenia pojedynczego segmentu nie przekracza maksymalnego prądu wyjściowego układu w stanie niskim.

Układ opóźniający włączony między linie sterujące portu PB powoduje automatyczną generację przerwania od tego portu po upływie zadanego czasu.

Jeżeli nie można wykorzystać więcej niż 8 linii wyjściowych do przeglądania klawiatury, to układ musi mieć 16 linii wejściowych z klawiatury. Do dyspozycji pozostaje jednak tylko jeden port 8-bitowy. Dlatego też konieczne jest użycie dwóch koderów priorytetowych (74348) połączonych kaskadowo. Wyjścia adresowe ($\overline{A0} - \overline{A2}$) tych układów są trójstanowe, można więc połączyć je razem, ponieważ są one aktywne tylko wtedy, gdy układ jest włączony (wejście $\overline{EI}$ w stanie niskim), a wśród wejść informacyjnych co najmniej jedno jest aktywne.

Jeśli wszystkie wyjścia klawiatury (linie $K_0 - K_{15}$) są nieaktywne, to wyjścia adresowe pierwszego układu są w stanie wysokiej impedancji, a wyjście $\overline{EO}$ przyjmuje stan niski. Identyczna sytuacja panuje na wyjściach drugiego kodera, zatem wejście $\overline{STB}$ układu PIO pozostaje nieaktywne.

Jeśli aktywna jest co najmniej jedna linia klawiatury dołączona do pierwszego kodera, to wybiera on linię o najwyższym priorytecie, a jej numer pojawia się na wyjściach adresowych. Jednocześnie wyjście $\overline{EO}$ przyjmuje stan wysoki, co blokuje pracę drugiego układu. Stan tego sygnału uzupełnia też zakodowany numer linii, dzięki czemu można stwierdzić, z którego kodera pochodzi informacja. Wyjście $\overline{EO}$ drugiego kodera jest też w stanie wysokim, a więc na wejściu $\overline{STB}$ układu PIO jest stan aktywny. W związku z tym 4-bitowy kod linii klawiatury jest zapamiętywany w rejestrze wejściowym układu PIO.

Jeżeli linie przyłączone do pierwszego kodera są nieaktywne, ale aktywna jest co najmniej jedna linia wejściowa drugiego kodera, to na jego wyjściach pojawia się numer linii o najwyższym priorytecie. Sygnał $\overline{EO}$ jest w stanie wysokim, co powoduje wpisanie informacji do rejestru wejściowego układu PIO. Jest ona uzupełniona o stan wyjścia $\overline{EO}$ pierwszego kodera (w tym przypadku jest to stan niski).

Zapamiętanie informacji w porcie PA powoduje zgłoszenie przerwania od tego portu.

Układ PIO może zgłaszać przerwania od obu portów, o ile zostanie odpowiednio zaprogramowany. Ponieważ układ ma tylko jedno wyjście zgłoszenia przerwania, rozróżnienie ich odbywa się dopiero w trakcie cyklu potwierdzenia przyjęcia przerwania przez mikroprocesor na podstawie wektora przerwań, który PIO przekazuje mikroprocesorowi.