POLITECHNIKA GDAŃSKA
WYDZIAŁ ELEKTRONIKI, TELEKOMUNIKACJI I INFORMATYKI

Imię i nazwisko studenta: Cezary Wieczorkowski
Nr albumu: 188584
Poziom kształcenia: studia pierwszego stopnia
Forma studiów: stacjonarne
Kierunek studiów: Elektronika i telekomunikacja
Profil: Komputerowe systemy elektroniczne

PRACA DYPLOMOWA INŻYNIERSKA

Tytuł pracy w języku polskim: Projekt i budowa laboratoryjnego stanowiska sterującego szyną danych

Tytuł pracy w języku angielskim: Design and construction of a laboratory data bus control station

Opiekun pracy: dr inż. Kamil Stawiarski

Streszczenie

Celem projektu było zaprojektowanie oraz uruchomienie zestawu laboratoryjnego umożliwiającego realizację ćwiczeń z sterowania szyną danych w laboratorium techniki cyfrowej. W ramach projektu dokonano analizy istniejącego stanowiska pod kątem możliwości rozbudowy oraz poprawy funkcjonalności. Następnie zaprojektowano oraz zbudowano prototyp nowego stanowiska uwzględniając wnioski z analizy poprzedniego.

Słowa kluczowe: Technika cyfrowa, stanowisko laboratoryjne, emulacja, szyna danych, ALU
Dziedzina nauki i techniki, zgodne z wymogami OECD: nauki inżynieryjno–techniczne: automatyka, elektronika, elektrotechnika i technologie kosmiczne

Abstract

The goal of this project was to design and construct a laboratory set that enables the performance of exercises on data bus control in a digital electronics laboratory. As part of the project, an analysis of the existing station was carried out in terms of the possibility of expansion and improvement of functionality. Then, a prototype of the new station was designed and built, taking into account the conclusions from the analysis of the previous one.

Keywords: Digital electronics, laboratory set, emulation, data bus, ALU

Spis treści

Wykaz ważniejszych oznaczeń i skrótów

ALU	(ang. arithmetic logic unit) jednostka arytmetyczno-logiczna
RAM	(ang. random access memory) pamięć o dostępie swobodnym
LED	(ang. light emitting diode) dioda elektroluminescencyjna
LCD	(ang. liquid crystal display) wyświetlacz ciekłokrystaliczny
GPIO	(ang. general-purpose input/output) uniwersalne wejście/wyjście
I/O	(ang. input/output) wejście/wyjście
I2C	(ang. inter integrated circuit) szeregowy synchroniczny interfejs komunikacyjny

1 Wstęp i cel pracy

Elektronika cyfrowa obecna jest w prawie każdym nowoczesnym urządzeniu. Jest to jedna z ważniejszych dziedzin elektroniki która umożliwia rozwój nowoczesnych technologii które leżą u podstaw nowoczesnego społeczeństwa. W procesie projektowania układów cyfrowych zawsze zachodzi potrzeba wymiany danych pomiędzy różnymi podzespołami układu. Jednym z sposobów realizacji procesu wymiany danych jest zastosowanie szyny danych. jest to zespół linii (przewodów) służących do przesyłania danych pomiędzy nadajnikami oraz odbiornikami do niej podłączonymi.
Celem pracy jest budowa stanowiska laboratoryjnego do realizacji ćwiczeń z sterowaniem szyną danych w laboratorium techniki cyfrowej. To ćwiczenie ma za zadanie zobrazować studentom jak można zrealizować komunikację między różnymi układami cyfrowymi przy pomocy wspólnej magistrali danych. Rozwiązania magistrali danych są powszechnie stosowane w nowoczesnych układach cyfrowych takich jak mikrokontrolery oraz mikroprocesory w celu wymiany danych pomiędzy różnymi podzespołami danego układu. Zrozumienie mechanizmów działania systemu opartego o magistralę danych pomoże studentom w zrozumienia zasad działania bardziej skomplikowanych układów cyfrowych.

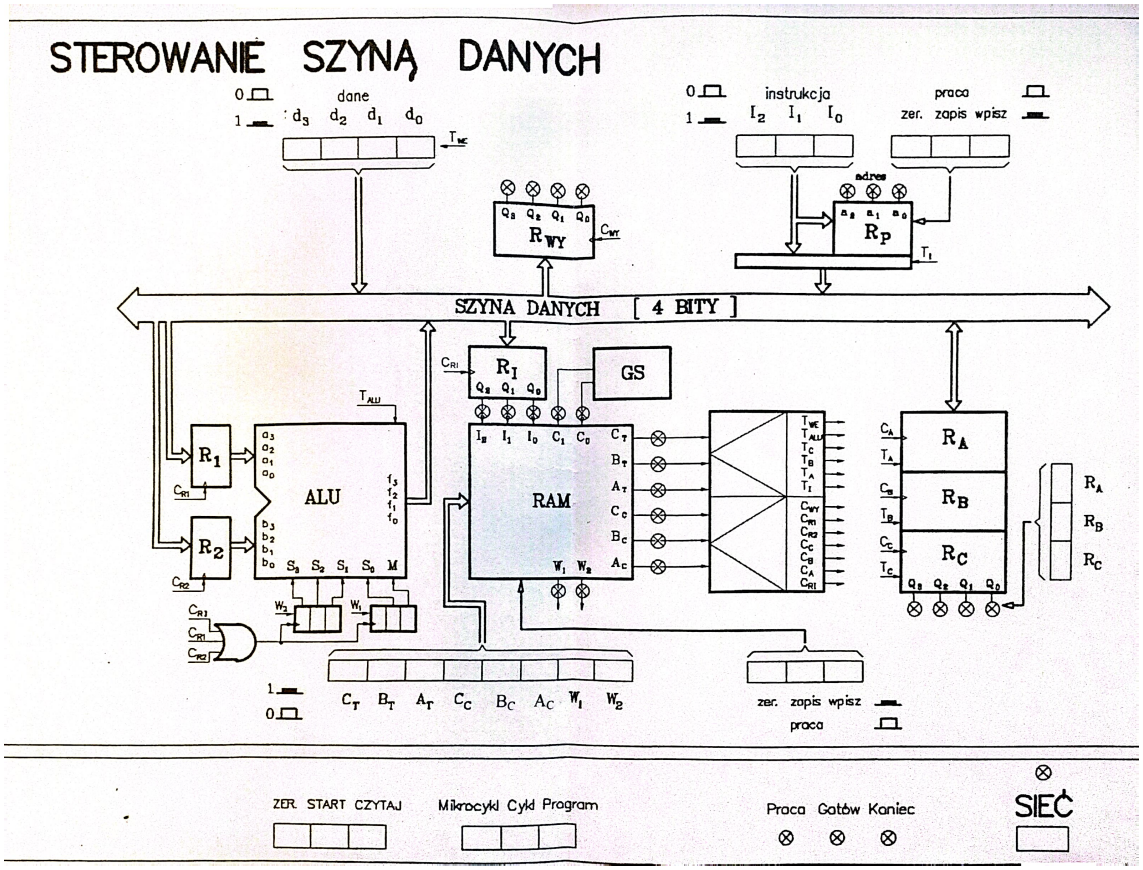
2 Analiza istniejącego stanowiska

Poniższy rozdział został napisany na podstawie załącznika nr 1.

2.1 Zasada działania stanowiska

2.1.1 Opis stanowiska

Zestaw laboratoryjny modeluje system komunikacji oparty na 4 bitowej szynie danych. W skład zestawu wchodzą urządzenia nadawcze i odbiorcze podłączone do szyny danych, logika sterująca procesem wymiany danych oraz elementy interfejsu użytkownika. Na rysunku 2.1 możemy zobaczyć płytę czołową zestawu która przedstawia jego schemat blokowy oraz interfejs użytkownika.
W przypadku tego stanowiska działanie wspomnianych urządzeń jest emulowane na dwóch mikrokontrolerach. W dalszej części rozdziału działanie zestawu zostanie opisane tak jakby było to urządzenie zbudowane z podzespołów przedstawionych na jego płycie czołowej.



Rysunek 2.1: Płyta czołowa zestawu

2.1.2 Elementy składowe stanowiska

Pamięć RAM

Pamięć ram w zestawie jest zorganizowana jako 32 słowa po 8 bitów.

Rysunek 2.2: Symbole przypisane poszczególnym bitom słowa w pamięci RAM

Zawartość słów w pamięci RAM kontroluję pracę zestawu:

- bity C_T, B_T, A_T - wybór nadajnika na szynie
- bity C_C, B_C, A_C - wybór odbiornika na szynie
- bity W_1, W_2 - wybór operacji ALU

Można więc powiedzieć że pamięć RAM zawiera instrukcje pracy zestawu. W celu wykonania kolejnych instrukcji należy zwiększyć o 1 obecny adres pamięci. Wykonanie jednej instrukcji z pamięci RAM nazywamy mikrocyklem.
Wejścia adresowe pamięci są połączone z rejestrem Rp (trzy najstarsze bity) oraz z generatorem Gs (dwa najmłodsze bity). Trzy najstarsze bity wyjścia danych pamięci połączone są z wejściami adresowymi dekodera 1 natomiast kolejne trzy z wejściami adresowymi dekodera 2. Dwa najmłodsze bity wyjścia danych pamięci połączone są wejściami informacyjnymi rejestrów przesuwnych instrukcji ALU.

Generator GS

Generator GS jest licznikiem modulo 4. Jego wyjścia połączone są z dwoma najmłodszymi wejściami adresowymi pamięci RAM. Podczas pracy generator wygeneruje kolejno wartości dwóch najstarszych bitów adresu od 00 do 11. W zależności od trybu pracy zestawu generowane są kolejne adresy pamięci do zapisu lub odczytu i wykonani jako instrukcje w mikrocyklu.

Dekoder sygnałów sterujących

Dekodery sygnałów sterujących to układy konwertujące liczby binarne na kod 1 z n. Dekoder 1 wytwarza sygnały sterujące nadajnikami które podawane są na wejścia sterujące przyłączaniem poszczególnych nadajników do szyny danych.

C_T	B_T	A_T	T_1	T_A	T_B	T_C	T_{ALU}	T_{WE}
0	0	0	1	0	0	0	0	0
0	0	1	0	1	0	0	0	0
0	1	0	0	0	1	0	0	0
0	1	1	0	0	0	1	0	0
1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	1

Rysunek 2.3: Tablica prawdy dla dekodera 1

Dekoder 2 działa analogicznie do dekodera 1 ale wytwarza sygnały sterujące odbiornikami. Sygnały generowane przez dekoder 1 podawane są na wejścia zapisu poszczególnych odbiorników.

C_C	B_C	A_C	C_{R1}	C_A	C_B	C_C	C_{R1}	C_{R2}	C_{WY}
0	0	0	1	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0
0	1	1	0	0	0	1	0	0	0
1	0	0	0	0	0	0	1	0	0
1	0	1	0	0	0	0	0	1	0
1	1	0	0	0	0	0	0	0	1

Rysunek 2.4: Tablica prawdy dla dekodera 2

Jednostka arytmetyczno logiczna ALU

Układ ten realizuje operacje arytmetyczne i logiczne na dwóch liczbach 4 bitowych zapisanych w rejestrach R_1 oraz R_2. Rodzaj wykonywanej operacji określa pięciobitowe słowo M, S_3, S_2, S_1, S_0 które jest zapisywane w dwóch 3 bitowych rejestrach przesuwnych. Na rysunku 2.1 widzimy sposób połączenia rejestrów przesuwnych w układzie. Na ich wejścia danych podane są sygnały W_1, W_2 z pamięci RAM. Natomiast na wejścia zegarowe podana jest suma logiczna sygnałów C_RI, C_R1, C_R2. Z tych zależności wynika, że podczas projektowania instrukcji z użyciem ALU w celu ustawienia odpowiedniego kodu operacji należy przewidzieć wygenerowanie sygnałów C_RI, C_R1, C_R2. Na rysunku 2.5 przedstawiono sposób kodowania operacji do wykonania przez ALU.

I_2	I_1	I_0	C_1	C_0	C_T	B_T	A_T	C_C	B_C	A_C	W_1	W_2
X	X	X	0	0							M	S_3
			0	1							S_2	S_1
			1	0							X	S_0
			1	1							X	X

Rysunek 2.5: Zasada kodowania operacji do wykonania przez ALU

Jednostka arytmetyczno logiczna ma możliwość zrealizowania 16 operacji arytmetycznych oraz logicznych:

Lp.	S_3	S_2	S_1	S_0	Operacje arytmetyczne M = 0	Operacje logiczne M = 1
1.	0	0	0	0	$Y = 0$	$Y = \overline{R1}$
2.	0	0	0	1	$Y = R1 \div R2$	$Y = \overline{R1 \wedge R2}$
3.	0	0	1	0	$Y = R1 \times R2 - 1$	$Y = \overline{R1} \vee R2$
4.	0	0	1	1	$Y = R1 + R2$	$Y = 1$
5.	0	1	0	0	$Y = R1^2 - R2^2$	$Y = R1 \oplus R2$
6.	0	1	0	1	$Y = R1 \times 2$	$Y = \overline{R1} \Leftrightarrow \overline{R2}$
7.	0	1	1	0	$Y = \frac{R1 + R2}{2}$	$Y = R1 \Rightarrow R2$
8.	0	1	1	1	$Y = R1 \times R2$	$Y = R1 \vee R2$
9.	1	0	0	0	$Y = R1 \,\%\, R2$	$Y = \overline{R1} \Rightarrow \overline{R2}$
10.	1	0	0	1	$Y = R1 \div 2$	$Y = \overline{\overline{R1} \wedge R2}$
11.	1	0	1	0	$Y = (R1 \times R2) - (R1+R2)$	$Y = R1 \wedge R2$
12.	1	0	1	1	$Y = R1 - R2$	$Y = \overline{R1 \oplus R2}$
13.	1	1	0	0	$Y = (R1 - R2)^2$	$Y = \overline{R2}$
14.	1	1	0	1	$Y = R1$	$Y = \overline{R1 \vee R2}$
15.	1	1	1	0	$Y = R1^2$	$Y = R1 \Leftrightarrow R2$
16.	1	1	1	1	$Y = R1 - 1$	$Y = \overline{R1 \oplus R2}$

Rysunek 2.6: Lista operacji realizowanych przez ALU

Rejestry pomocnicze

Rejestry są układami przechowującymi dane. W zestawie występują następujące rejestry:

- R_A, R_B, R_C - są to rejestry przeznaczone do wykorzystania przez użytkownika, mogą one zarówno odbierać jak i nadawać dane na szynę
- $R_1 R_2$ - przechowują argumenty operacji ALU, mogą tylko odbierać dane z szyny
- R_{wy} - służy do prezentacji wyników operacji wykonywanych przez zestaw laboratoryjny, może tylko odbierać dane z szyny
- R_I - przechowuje trzy najstarsze bity adresu pamięci, może tylko odbierać dane z szyny

- R_p - to zespół ośmiu rejestrów przeznaczonych do przechowywania adresów pamięci RAM pod którymi znajdują się instrukcje do wykonania przez zestaw. Rejestr ten jest wykorzystywany w trybie pracy Program

2.1.3 Interfejs użytkownika

Na rysunku 2.1 poza schematem blokowym zestawu widzimy przyciski oraz diody LED służące do sterowania zestawem i monitorowania jego pracy.

Kontrola pracy zestawu

Do sterowania pracą zestawu służą następujące przyciski:

- ZER. - przycisk zerowania generatora GS
- START - przycisk rozpoczynający pracę zestawu
- CZYTAJ - przycisk inicjujący odczyt z klawiatury dane i kontynuacje pracy zestawu
- Mikrocykl, Cykl, Program - przełączniki wyboru trybu pracy zestawu
- klawiatura dane - klawiatura do wprowadzania danych wejściowych na szynę danych

Zapis i odczyt danych z pamięci RAM

Do obsługi zapisu pamięci RAM służą przyciski:

- zer. - zerowanie adresu przy zapisie i odczycie pamięci RAM
- zapis/praca - wybór pomiędzy zapisem danych do pamięci RAM a pracą zestawu
- wpisz - zapis danych z klawiatury do RAM po jego wciśnięciu adres zwiększa się o 1
- klawiatura C_T - W_2 - przyciski do ustawiania wartości bitowej słowa wpisywanego do pamięci RAM

Zapis i odczyt danych z R_P

Zapis oraz odczyt danych z rejestru odbywa się podobnie jak w przypadku pamięci RAM z użyciem odpowiednich przycisków:

- klawiatura dane - klawiatura do wprowadzania danych wejściowych na szynę danych
- klawiatura instrukcja - klawiatura do wprowadzania wartości do rejestru R_P
- zer. - zerowanie adresu przy zapisie i odczycie pamięci R_P
- zapis/praca - wybór pomiędzy zapisem danych do pamięci R_P a pracą zestawu
- wpisz - zapis danych z klawiatury instrukcja do R_P po jego wciśnięciu adres zwiększa się o 1

Monitorowanie pracy zestawu

Na płycie czołowej zestawu znajdują się również diody LED obrazujące wartości

- Adresu pamięci RAM
- Wartości słowa w pamięci RAM
- Wartości słowa w rejestrze R_{WY}
- Wartości słowa w jednym z rejestrów R_A, R_B, R_C wybieranych przyciskami
- Adres rejestru R_P

2.1.4 Tryby pracy stanowiska

Stanowisko posiada trzy tryby pracy:

- Mikrocykl - zestaw po wciśnięciu przycisku START wykona tylko jedną instrukcję z pamięci RAM, zostanie wygenerowana tylko jedna wartość w generatorze GS. Po jej wykonaniu zestaw zatrzyma się i będzie oczekiwał na ponowne wciśnięcie przycisku START.
- Cykl - zestaw po wciśnięciu przycisku START wykona cztery kolejne instrukcje z pamięci RAM. Nastąpią cztery kolejne zmiany wyjść w generatorze GS. Po wykonaniu czterech instrukcji zestaw zatrzyma się i będzie oczekiwał na ponowne wciśnięcie przycisku START.
- Program - po wciśnięciu przycisku START zestaw poda na wyjście rejestru R_P kolejno osiem instrukcji które zostały do niego uprzednio zapisane. Układ wygeneruje kolejne 32 stany generatora GS, zmiana wartości w rejestrze R_P nastąpi co czwartą wartość generatora GS.

Niezależnie od trybu pracy zestawu automatycznie generowane są tylko wartości generatora GS. W celu wykonania kolejnych instrukcji należy manualnie załadować wartość trzech najstarszych bitów adresu pamięci kolejnej instrukcji do rejestru R_I. W tym celu w pierwszej instrukcji każdego bloku czterech instrukcji w pamięci RAM należy umieścić wartość 000000XX. Taka wartość instrukcji odpowiada ustawieniu rejestru R_I jako odbiornika a rejestru R_P jako nadajnika. W przypadku pracy w trybie Mikrocykl oraz Cykl wartość trzech najstarszych bitów adresu pamięci kolejnej instrukcji należy za każdym razem manualnie wprowadzić na klawiaturze instrukcji. Natomiast w trybie Program kolejne osiem wartości rejestru R_P zostanie automatycznie podanych na jego wyjście.

2.2 Krytyczna ocena stanowiska

2.2.1 Założenia funkcjonalne

Zestaw w obecnej formie pozwala na realizacje prostych programów obrazujących działanie szyny danych w praktyce. Urządzenia podłączone do szyny współpracują ze sobą w sposób przemyślany pozwalają na realizowanie prostych operacji na jednostce ALU, zapis danych do rejestrów, pobieranie oraz prezentowanie danych użytkownikowi. Tryby pracy zestawu pozwalają na łatwe zaobserwowanie procesów zachodzących w zestawie podczas wykonywania programu. Tryby cykl i mikrocykl ułatwiają debugowanie programów przechodząc przez każdy cykl lub mikrocykl programu.

2.2.2 Interfejs użytkownika

Interfejs użytkownika zestawu opiera się na przyciskach jako urządzeniach wejściowych oraz diodach LED jako urządzeniach wyjściowych. Wizualizacja stanu poszczególnych rejestrów oraz bitów za pomocą diod LED jest czytelna i intuicyjna. W połączeniu ze schematem blokowym na płycie pozwala to w łatwy sposób zrozumieć jakie działania zestaw wykonuje w danym momencie. Możliwość podglądu adresu pamięci RAM oraz słowa pod nim zapisanego ułatwia znajdywanie błędów w programach uruchamianych na zestawie.
Wprowadzanie danych do zestawu (wartości pamięci RAM, rejestru R_P oraz danych na szynę) odbywa się za pomocą klawiatur. Jest to kilka przycisków gdzie każdy przycisk odpowiada jednemu bitowi wprowadzanego słowa. Wciśnięty przycisk odpowiada wartości 1 a wyciśnięty wartości 0. Wprowadzanie wartości danych binarnych w ten sposób jest intuicyjne i szybkie.
Pozostałe przyciski zestawu służą do kontroli jego pracy oraz kontroli procesów zapisu i odczytu wartości z pamięci RAM oraz rejestru R_P. Jest to sposób sprawdzony i stosunkowo intuicyjny. Głównym ograniczeniem tego podejścia jest trudność w dokonaniu zmian w interfejsie gdyż jego elementy są stałymi elementami płyty czołowej zestawu. Kolejnym ograniczeniem tego podejścia jest ograniczona możliwość sposobów zapisu oraz odczytu pamięci RAM oraz rejestru R_P. Zapis oraz odczyt musi się odbywać w sposób sekwencyjny co znacząco wydłuża proces w przypadku gdy zachodzi potrzeba zmiany lub odczytu wartości znajdujących się w dalszych regionach pamięci lub rejestru.

2.2.3 Platforma sprzętowa

W obecnej formie zestaw jest oparty o dwa mikrokontrolery Atmega 8 oraz Atmega 32. Odpowiadają one za symulację pracy komponentów składowych zestawu oraz obsługę interfejsu użytkownika. Emulacja komponentów zestawu programowo niesie za sobą szereg zalet: możliwość łatwej modyfikacji działania zestawu, mniejszy pobór mocy niż w przypadku dedykowanych układów scalonych. Podejście emulacyjne jest po części wymuszone ograniczoną dostępnością w handlu dedykowanych układów scalonych takich jak jednostki ALU ze względu na ich znikomą popularność nowych projektach.
Dużym problemem w przypadku tego zestawu jest to, że kod źródłowy programów pracujących na mikrokontrolerach w zestawie nie jest dostępny. W przypadku potrzeby naprawy błędów w oprogramowaniu bądź chęci rozwinięcia funkcjonalności zestawu konieczne byłoby napisanie całego oprogramowania od nowa. Wiązałoby się to z dużym nakładem pracy. Architektura oparta o dwa osobne układy na których pracują różne programy również nie jest optymalna. Podział zadań na dwa mikrokontrolery komplikuje proces pisania oprogramowania oraz naprawy potencjalnych błędów.

3 Koncepcja nowego stanowiska

3.1 Założenia projektowe

Po przeanalizowaniu poprzedniego zestawu laboratoryjnego wypracowano następujące założenia dla projektu nowego zestawu:

- Zestaw musi zachować bloki funkcjonalne oraz zasadę działania poprzedniego zestawu.
- Zestaw ma zachować tryby pracy: cykl, mikrocykl oraz program.
- Zestaw musi emulować komponenty rzeczywistego układu szyny danych.
- Zestaw musi umożliwiać łatwą naprawę błędów w jego funkcjonowaniu oraz rozbudowę o nowe funkcjonalności.
- Zestaw musi zachować przejrzysty sposób wizualizacji stanu poszczególnych komponentów.
- Zestaw musi posiadać uniwersalny interfejs użytkownika pozwalający na łatwą obsługę zestawu oraz późniejsze jego modyfikacje.

3.2 Projekt sprzętowy

3.2.1 Wybór mikrokontrolera

Ze względu na wybrane podejście symulacji programowej działania układu szyny danych, mikrokontroler jest centralnym elementem budowanego układu. W celu zapewnienia możliwości dalszej rozbudowy oraz wprowadzania poprawek do układu musi być to nowoczesna i łatwo dostępna platforma. Istniejąca wersja zestawu jest oparta o mikrokontroler Atmega 8 oraz Atmega 32. Są to urządzenia pracujące z częstotliwością 16 MHz oraz posiadające odpowiednio 1 Kb oraz 2 Kb pamięci RAM [1] [2]. Większość mikrokontrolerów dostępnych obecnie na rynku oferuje znacząco większe możliwości.
Jako mikrokontroler do budowy zestawu wybrano układ RP2040 firmy Rassbery Pi [3] obecny na płytce rozwojowej Rassbery Pi Pico. Jest to układ oparty o architekturę ARM Cortex M0+, posiada 264 Kb pamięci RAM oraz pracuje z częstotliwością 133 MHz. Układ posiada duży zapas zasobów sprzętowych co pozwoli na szerokie możliwości rozbudowy zestawu w przyszłości. Dodatkowo dostępność układu na płytce rozwojowej uprości proces budowy sprzętowej części zestawu gdyż płytka ta zawiera wszystkie elementy niezbędne do pracy mikrokontrolera.

3.2.2 Elementy interfejsu użytkownika

Interfejs użytkownika zestawu powinien umożliwiać łatwą obsługę zestawu oraz umożliwiać łatwe wprowadzenie zmian w jego strukturze. Takie wymagania najlepiej spełni interfejs oparty o wyświetlacz wyświetlający menu z opcjami do wyboru oraz interfejs do nawigacji pomiędzy nimi. Jako wyświetlacz zdecydowano się zastosować wyświetlacz znakowy LCD 20x4. Jest to wyświetlacz oparty o popularny układ sterujący HD44780 [4]. Dzięki prostocie obsługi oraz długiej obecności na rynku tego typu wyświetlacze są wspierane przez bardzo szeroką gamę oprogramowania.
W celu nawigacji po menu zdecydowano się zastosować enkoder kwadraturowy. Jest to element generujący impulsy na podstawie ruchu obrotowego. Pozwala on na stwierdzenie kierunku oraz prędkości obrotu. Tego typu urządzenia stosowane są w układach sterowania pracą silników lub jako elementy wejściowe interfejsu użytkownika. Enkodery stosowane jako element nawigacyjny

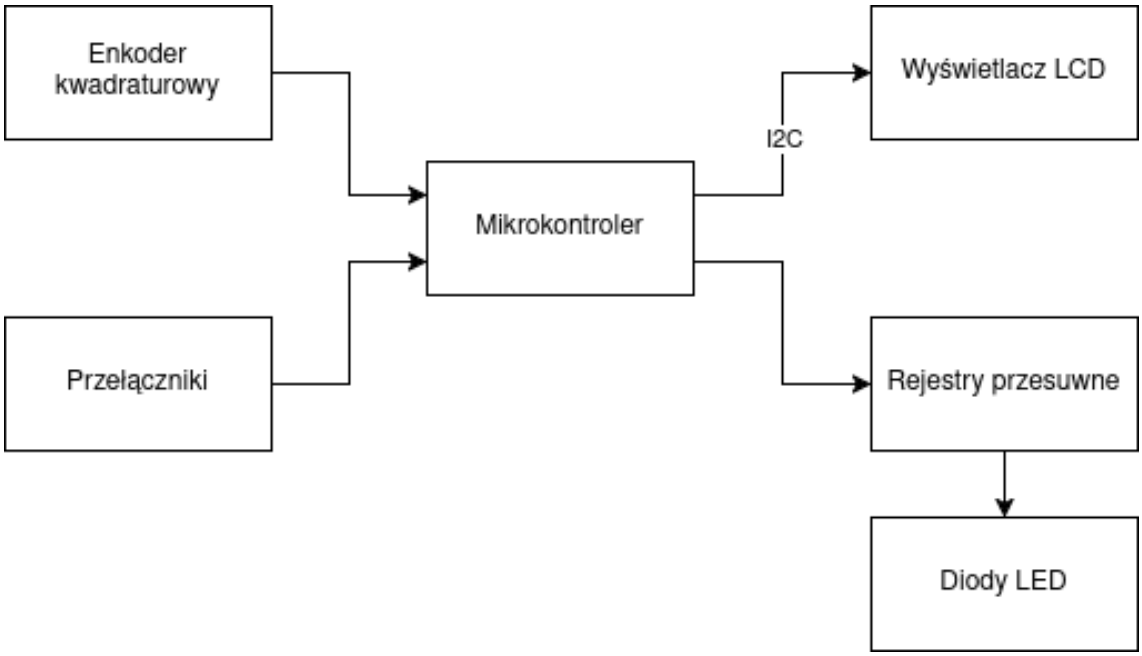
często posiadają wbudowany przycisk [5] dzięki czemu obracając enkoderem można wybierać opcje menu a naciskając go można potwierdzać wybór.
Dodatkowo w celu zapewnienia wizualizacji stanu elementów podczas pracy zestawu zdecydowano się ja w oryginale zastosować diody LED do wizualizacji stanu adresów oraz słowa pamięci RAM, stanu rejestru R_{WY} oraz wybranego rejestru (R_A, R_B, R_C).
Jako elementy do wprowadzania danych binarnych zdecydowano się zachować przełączniki jak w oryginalnym zestawie. Dla uproszczenia obsługi zestawu dane do pamięci RAM oraz rejestru R_P będą wprowadzane za pomocą jednego zestawu ośmiu przełączników natomiast dane użytkownika na szynę będą wprowadzana za pomocą osobnych czterech przełączników.

3.2.3 Dodatkowe peryferia

Sumując liczbę końcówek GPIO mikrokontrolera potrzebną do obsługi wybranych peryferiów otrzymujemy 43. Mikrokontroler RP2040 posiada 30 końcówek GPIO [3] co oznacza, że nie jest możliwe bezpośrednie podłączenie wszystkich peryferiów. W celu obsługi wszystkich peryferiów konieczne jest zastosowanie dodatkowych układów scalonych rozszerzających ilość dostępnych końcówek GPIO.
Jednym z prostszych tego typu układów jest rejestr przesuwny. Jest to układ który posiada wejście zegarowe oraz wejście danych danych. Po każdym zboczu zegara dane z wejścia danych są przesuwane do kolejnej komórki rejestru. Niektóre rejestry przesuwne posiadają wyjście danych z ostatniej komórki rejestru. Dzięki temu można połączyć kilka rejestrów kaskadowo. Jednym z takich układów jest 74HC595 [6]. Jest to 8 bitowy rejestr przesuwny z wyjściem równoległym. Rejestry te zastosowano do sterowania diodami LED. Użyto trzech układów ze względu na konieczność wysterowania 22 diod.
W celu redukcji ilości końcówek GPIO potrzebnych do obsługi wyświetlacza LCD zdecydowano się zastosować ekspander I/O PCF8574 [7]. Jest to układ który pozwala na sterowanie 8 bitowym portem wejścia/wyjścia za pomocą interfejsu I2C. Dzięki temu obsługa wyświetlacza zajmuje tylko dwie końcówki GPIO potrzebne do obsługi interfejsu I2C.

3.2.4 Schemat blokowy sprzętowej części zestawu

Na rysunku 3.1 przedstawiono schemat blokowy sprzętowej części zestawu.



Rysunek 3.1: Schemat blokowy sprzętowej części zestawu

3.3 Architektura oprogramowania

3.3.1 Struktura programu

3.3.2 Emulacja elementów układu szyny danych

3.3.3 Obsługa peryferiów sprzętowych

3.3.4 Obsługa interfejsu użytkownika

4 Implementacja stanowiska

4.1 Część sprzętowa

4.2 Część programowa

5 Testy stanowiska

Opis testów przeprowadzonych na stanowisku. Przykłady oraz analiza użytych programów testowych.

6 Podsumowanie

Podsumowanie rezultatów pracy, wnioski oraz możliwości dalszego rozwoju projektu.

Bibliografia

[1] *Atmega 8*. Rev.2486AA–AVR–02/2013. Atmel Corporation. Lut. 2013. URL: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-2486-8-bit-AVR-microcontroller-ATmega8_L_datasheet.pdf.

[2] *Atmega 32*. 2503Q–AVR–02/11. Atmel Corporation. Lut. 2011. URL: https://ww1.microchip.com/downloads/en/DeviceDoc/doc2503.pdf.

[3] *RP2040*. Ver. eec2b0c-clean. Raspberry Pi Ltd. Paź. 2024. URL: https://ww1.microchip.com/downloads/en/DeviceDoc/doc2503.pdf.

[4] *HD44780 LCD Starter Guide*. URL: https://www-leland.stanford.edu/class/ee281/handouts/lcd_tutorial.pdf (term. wiz. 08. 01. 2025).

[5] *Rotary Encoder*. URL: https://electropeak.com/learn/rotary-encoder-how-it-works-how-to-use-with-arduino/ (term. wiz. 08. 01. 2025).

[6] *SNx4HC595*. Texas Instruments. Paź. 2021. URL: https://www.ti.com/lit/ds/symlink/sn74hc595.pdf.

[7] *PCF8574*. Texas Instruments. Wrz. 2024. URL: https://www.ti.com/lit/ds/symlink/pcf8574.pdf.

Spis rysunków

Spis tabel

7 Załączniki

7.1 Załącznik nr 1: Instrukcja obsługi stanowiska

9. STEROWANIE SZYNĄ DANYCH

9.1. WSTĘP

Celem ćwiczenia jest zapoznanie studentów z zagadnieniami sterowania transmisją informacji poprzez szynę danych, pomiędzy kilkoma nadajnikami i odbiornikami. Z tymi zagadnieniami można spotkać się często przy konstrukcji cyfrowych układów sterowania i układów współpracujących z mikroprocesorami.

Opisany zestaw laboratoryjny jest symulatorem czterobitowej szyny danych. Układ został fizycznie zrealizowany w oparciu o dwa mikroprocesory i umożliwia realizację przesłań danych pomiędzy różnymi układami przyłączonymi do szyny danych oraz wyświetlanie informacji o stanie poszczególnych bloków.

9.2. OPIS ZESTAWU LABORATORYJNEGO

Na wstępnie wyjaśnijmy znaczenie terminu „*szyna danych*". Szyna danych (ang. *data bus*) to zespół linii (przewodów) służących do przesyłania danych między elementami połączonymi za jej pośrednictwem. Fizycznie szyna danych to przewody, do których przyłączone zostały wejścia i wyjścia układów współpracujących z szyną. Nadajnik ma wyjścia podłączone do szyny danych, natomiast odbiornik – wejścia. Układy będące nadajnikami informacji, albo mają wyjścia trójstanowe, albo są podłączone do szyny poprzez bramki z wyjściami trójstanowymi.

Sytuacja jest prosta, gdy do szyny podłączony jest tylko jeden nadajnik – wtedy trzeba tylko wygenerować sygnał zapisujący dane pojawiające na szynie do wybranego odbiornika. W przypadku, gdy do szyny danych podłączonych jest kilka źródeł informacji konieczny jest wtedy układ sterujący wyjściami trójstanowymi nadajników oraz sygnałami strobującymi odbiorników. Zagadnienie to zostanie omówione w p.9.2.3.

Czasami jednak, mówiąc „*szyna danych*" mamy na myśli cały zespół nadajników i odbiorników połączonych galwanicznie przewodami, które są fizycznie szyną - w tym przypadku czterobitową.

W tym zestawie przesłania po szynie danych i przyłączanie do niej różnych układów są symulowane programowo przy wykorzystaniu dwóch mikrokontrolerów z rodziny AVR (układów Atmega8 i Atmega32).

Działanie tego zestawu laboratoryjnego zostało opisane tak, jakby to było urządzenie zbudowane z podzespołów przedstawionych na jego głównej płycie czołowej i operacje przesłań byłyby dokonywane w konkretnym sprzęcie, przy wykonywaniu wyspecyfikowanych instrukcji i fizycznych dekoderów sterujących przesyłaniem informacji przez szynę danych.

Z punktu widzenia użytkownika nieistotna jest budowa całego fizycznego układu w oparciu o mikrokontrolery i to, że po odpowiednim zaprogramowaniu symulują one pracę sprzętowej szyny danych. Istotne jest to, że układ symulatora zachowuje się tak samo jak równoważny mu układ zbudowany z układów TTL średniej i małej skali integracji.

Poprawność działania szyny danych należy kontrolować obserwując zawartości rejestrów wyświetlane na LED-ach oraz stany diod sygnalizacyjnych.

Widoki płyt: czołowej i pulpitu zestawu laboratoryjnego przedstawiono na Rys.9.1., który zamieszczono w ***Części 2*** instrukcji ***„TECHNIKA CYFROWA. LABORATORIUM. CYKL ĆWICZEŃ GRUPOWYCH."***.

9.2.1. OZNACZENIA ELEMENTÓW NA PULPICIE ZESTAWU

Przełączniki

ZER. – przycisk zerowania generatora sterującego GS.

START – przycisk inicjujący rozpoczęcie wykonywania cyklu, mikrocyklu lub programu, w zależności od ustawienia przełączników wyboru trybu (rodzaju) pracy zestawu.

CZYTAJ – przycisk inicjujący odczyt danych z klawiatury **dane** i ponowny start działania układu sterującego, który zatrzymał się po napotkaniu instrukcji czytania danych.

Mikrocykl
Cykl
Program } – zespół przełączników zależnych - wybór trybu pracy zestawu laboratoryjnego.

Sieć – przełącznik do włączania/wyłączania zasilania zestawu laboratoryjnego.

Diody LED

Praca – LED sygnalizujący wykonywanie przez zestaw cyklu, mikrocyklu lub programu.

Gotów – LED sygnalizujący oczekiwanie na wpis danych wejściowych - dioda zgaśnie po odczycie stanu przycisków z klawiatury **dane**, po naciśnięciu przycisku **CZYTAJ**.

Koniec – LED sygnalizujący zakończenie wykonywania przez zestaw cyklu, mikrocyklu lub programu.

SIEĆ – LED sygnalizujący włączenie zasilania zestawu laboratoryjnego.

9.2.2. OPIS ELEMENTÓW NA GŁÓWNEJ PŁYCIE CZOŁOWEJ ZESTAWU

Na płycie głównej zestawu laboratoryjnego *STEROWANIE SZYNĄ DANYCH* przedstawiono jego schemat blokowy. Uwidoczniono na nim połączenia między układami współpracującymi z szyną oraz elementy układu sterującego pracą szyny danych. Ponadto na płycie głównej zestawu laboratoryjnego znajdują się przyciski i przełączniki służące do programowania i sterowania pracą zestawu, oraz diody LED, których przeznaczenie opisano poniżej.

Przełączniki

a) w dolnej części płyty głównej:

zer. – przycisk zerowania adresu przy zapisie oraz odczycie RAM-u.

zapis/praca – przełącznik wyboru trybu pracy RAM: *wciśnięty = zapis RAM-u, wyciśnięty = praca zestawu.*

wpisz – przycisk do zapisania, do RAM-u słowa ustawionego na klawiaturze C_T, B_T, A_T, C_C, B_C, A_C, W_1, W_2, po jego wciśnięciu adres automatycznie zwiększa się o 1.

C_T, B_T, A_T, C_C, B_C, A_C, W_1, W_2 – klawiatura do ustawiania słowa przy zapisie RAM-u: *przycisk wciśnięty = „1”, wyciśnięty = „0”.*

R_A, R_B, R_C – zespół przełączników zależnych do wyboru rejestru, którego zawartość ma być wyświetlana na diodach, umieszczonych na płycie czołowej zestawu laboratoryjnego, poniżej tych rejestrów.

b) w górnej części płyty głównej:

dane (d_3, d_2, d_1, d_0) – klawiatura do wprowadzania danych wejściowych: *przycisk wciśnięty = „1”, wyciśnięty = „0”.*

instrukcja (I_2, I_1, I_0) – klawiatura do wprowadzania kodu instrukcji: *przycisk wciśnięty = „1”, wyciśnięty = „0”.*

zer. – przycisk przy rejestrze R_P, do zerowania adresu przy zapisie, oraz odczycie rejestru programu.

praca/zapis – przełącznik przy rejestrze R_P, do wyboru trybu pracy rejestru programu: *wciśnięty = zapis R_P, wyciśnięty = praca zestawu;*

wpisz – przycisk przy rejestrze R_P, do zapisania słowa ustawionego na klawiaturze I_2, I_1, I_0 do R_P, po jego wciśnięciu adres automatycznie zwiększa się o 1.

Diody LED

I_2, I_1, I_0, C_1, C_0 – wyświetlanie aktualnego adresu pamięci, przy zapisie jak i odczycie RAM-u.

C_T, B_T, A_T, C_C, B_C, A_C, W_1 ,W_2 – wyświetlanie wartości słowa sterującego wpisanego do pamięci RAM, znajdującego się pod adresem wyświetlanym przez diody I_2, I_1, I_0, C_1, C_0, przy odczycie RAM-u.

adres (a_2, a_1, a_0) – wyświetlanie adresu aktualnie wykonywanej instrukcji z rejestru R_P.

diody rejestrów R_A, R_B, R_C (Q_3, Q_2, Q_1, Q_0) – wyświetlane zawartości poszczególnych rejestrów. Wybór rejestru dokonuje się poprzez wciśnięcie odpowiedniego przycisku: R_A, R_B lub R_C, które znajdują się z prawej strony rejestrów.

diody rejestru R_{WY} (Q_3, Q_2, Q_1, Q_0) – wyświetlanie zawartości rejestru R_{WY}.

9.2.3. BLOKI FUNKCJONALNE I ELEMENTY ZESTAWU

1) Układ sterujący

Układ sterujący pracą zestawu laboratoryjnego *STEROWANIE SZYNĄ DANYCH* zbudowany jest z następujących zespołów:

- pamięci RAM o organizacji 32 x 8 (32 słowa 8-bitowe),
- generatora sygnałów zegarowych GS,
- dwóch dekoderów adresowych (BIN → 1 z n).

◄ Pamięć RAM

Jest to najważniejsza część tego zestawu laboratoryjnego. W niej są zapisywany kody poszczególnych mikroinstrukcji, które mają być realizowane w zestawie po wczytaniu instrukcji zadanej do wykonania.

Wejścia adresowe pamięci RAM połączone są z wyjściami rejestru instrukcji R_I (bity I_2, I_1, I_0) i z wyjściami generatora GS (bity C_1, C_0); przy czym C_0 jest najmłodszym bitem adresu, a I_2 – najstarszym.

Wyjścia pamięci RAM oznaczone symbolami: C_T, B_T, A_T, są połączone z wejściami adresowymi dekodera **1**; wyjścia: C_C, B_C, A_C - z wejściami adresowymi dekodera **2**; natomiast wyjścia W_1 i W_2 – z wejściami informacyjnymi rejestrów pomocniczych R_1 i R_2.

Do wejść danych RAM-u przyłączone są sygnały z przełączników: C_T, B_T, A_T, C_C, B_C, A_C, W_1, W_2, na których ustawiane są wartości słów zapisywanych w pamięci.

Do wejść sterujących pracą pamięci przyłączone są sygnały sterujące zapisem i odczytem RAM-u – z przełączników i przycisków: **zer.**, **praca/zapis**, **wpisz** (znajdują się one w dolnej części płyty głównej).

Organizację pamięci RAM, z uwidocznieniem jej podziału na obszary przyporządkowane poszczególnym instrukcjom przedstawiono w tabeli 9.1.

I_2	I_1	I_0	C_1	C_0	C_T	B_T	A_T	C_C	B_C	A_C	W_1	W_2
0	0	0	0	0							M	S_3
			0	1							S_2	S_1
			1	0							X	S_0
			1	1							X	X
0	0	1	0	0							M	S_3
			0	1							S_2	S_1
			1	0							X	S_0
			1	1							X	X
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
1	1	1	0	0							M	S_3
			0	1							S_2	S_1
			1	0							X	S_0
			1	1							X	X

Tabela. 9.1. Organizacja zawartości pamięci RAM.

W każdym słowie pamięci RAM zapisujemy informacje opisujące działania w danym mikrocyklu w następującej kolejności:

- na pozycjach C_T, B_T, A_T - zakodowaną informację o numerze nadajnika wykorzystywanego w tym mikrocyklu (patrz Tabela.9.2.).
- na pozycjach C_C, B_C, A_C - zakodowaną informację o numerze odbiornika wykorzystywanego w tym mikrocyklu (patrz Tabela.9.3.).
- na pozycjach W_1 i W_2 - zakodowaną informację o funkcji realizowanej przez ALU przy wykonywaniu danej instrukcji (patrz Tabela.9.4. i Tabela.9.5.).

◄ Generator GS

Generator sygnałów zegarowych GS steruje wykonywaniem poszczególnych instrukcji. Ponadto – w czasie zapisu pamięci RAM - wyznacza kolejne adresy, pod którymi mają być zapisywane kody mikroinstrukcji.

Stany jego wyjść C_1, C_0 przyjmują kolejno wartości: 00, 01, 10, 11. W trybie **praca**, każde przejście między kolejnymi stanami C_1, C_0 oznacza, że został wykonany jeden mikrocykl. Do realizacji każdej instrukcji muszą zostać wykonane mikroinstrukcje w czterech mikrocyklach. W trybie **zapis** stan wyjść generatora wyznacza adresy, pod którymi zapisujemy kolejne słowa do pamięci RAM.

◄ Dekodery sygnałów sterujących

Dekoder **1**, na którego wejścia podawane są z RAM-u sygnały C_T, B_T, A_T, wytwarza sygnały sterujące T_i. Jego sygnały wyjściowe: T_I, T_A, T_B, T_C, T_{ALU}, T_{WE}, podawane są na wejścia sterujące przyłączaniem poszczególnych nadajników do szyny danych. Wyznaczają one moment otwarcia trójstanowych wyjść umożliwiając wystawienie na szynę danych informacji z wybranego nadajnika. Sposób generowania sygnałów sterujących T_i na wyjściach dekodera **1** przedstawiono w tabeli 9.2.

C_T	B_T	A_T	T_I	T_A	T_B	T_C	T_{ALU}	T_{WE}
0	0	0	1	0	0	0	0	0
0	0	1	0	1	0	0	0	0
0	1	0	0	0	1	0	0	0
0	1	1	0	0	0	1	0	0
1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	1

Tabela. 9.2. Tablica pracy dekodera **1** - wytwarzającego sygnały sterujące nadajnikami.

Dekoder **2**, na którego wejścia podawane są z RAM-u sygnały C_C, B_C, A_C, wytwarza sygnały sterujące C_j. Jego sygnały wyjściowe: C_{RI}, C_A, C_B, C_C, C_{R1}, C_{R2}, C_{WY}, podawane są na wejścia zapisu (zegarowe) poszczególnych odbiorników. Wyznaczają one moment zapisu informacji z szyny danych do poszczególnych odbiorników. W zestawie laboratoryjnym wszystkie odbiorniki przyłączone do szyny danych to rejestry z równoległym wpisem danych.

Sposób generowania sygnałów sterujących C_j na wyjściach dekodera 2 przedstawiono w tabeli 9.3.

C_C	B_C	A_C	C_{RI}	C_A	C_B	C_C	C_{R1}	C_{R2}	C_{WY}
0	0	0	1	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0
0	1	1	0	0	0	1	0	0	0
1	0	0	0	0	0	0	1	0	0
1	0	1	0	0	0	0	0	1	0
1	1	0	0	0	0	0	0	0	1

Tabela 9.3. Tablica pracy dekodera 2 – wytwarzającego sygnały sterujące odbiornikami.

Rozdzielenie zakresu pracy obu dekoderów powoduje, że w jednej chwili czasu do szyny danych podłączony jest tylko jeden nadajnik i jeden odbiornik. Pozwala to wyeliminować niejednoznaczności w wyborze urządzenia, które ma być aktualnym nadajnikiem, a które odbiornikiem.

2) Jednostka arytmetyczno-logiczna (ALU)

Układ ALU realizuje operacje arytmetyczne i logiczne na 4-bitowych liczbach zapisanych w rejestrach R_1 i R_2. Rodzaj wykonywanej operacji określa pięciobitowe słowo stanu: $M S_3 S_2 S_1 S_0$, które jest zapisywane w dwóch 3-bitowych rejestrach przesuwających.

Na Rys.9.1. przedstawiono układ połączeń ALU z wyjściami w/w rejestrów. Na wejścia informacyjne rejestrów przesuwających podawane są z pamięci RAM sygnały W_1 i W_2, zaś na ich wejścia zegarowe podawany jest sygnał = $C_{RI} + C_{R1} + C_{R2}$. Z tych zależności wynika, że w czasie projektowania instrukcji z użyciem jednostki arytmetyczno-logicznej należy koniecznie przewidzieć wygenerowanie w trzech pierwszych mikrocyklach sygnałów C_{RI}, C_{R1} i C_{R2}.

W celu ustalenia prawidłowych stanów wejść sterujących, określających wybór operacji, jaką ma wykonać ALU należy także we właściwej kolejności zapisać ich wartości w pamięci RAM, na pozycjach W_1 i W_2 w kolejnych mikrocyklach. Sposób kodowania funkcji realizowanej przez ALU przedstawiono w tabeli 9.4.

I_2	I_1	I_0	C_1	C_0	C_T	B_T	A_T	C_C	B_C	A_C	W_1	W_2
X	X	X	0	0							M	S_3
			0	1							S_2	S_1
			1	0							X	S_0
			1	1							X	X

Tabela 9.4. Zasada kodowania funkcji realizowanej przez ALU.

Jednostka arytmetyczno-logiczna jest układem kombinacyjnym i wynik na jej wyjściu pojawia się po podaniu danych na wejścia. Wystawienie wyniku operacji na magistralę wymaga otwarcia wyjść trójstanowych ALU poprzez podanie sygnału T_{ALU}. W tym układzie można zrealizować po 16 operacji logicznych i arytmetycznych, które wyspecyfikowano w tabeli 9.5.

Lp.	S_3	S_2	S_1	S_0	Operacje arytmetyczne **M = 0**	Operacje logiczne **M = 1**
1.	0	0	0	0	$Y = 0$	$Y = \overline{R1}$
2.	0	0	0	1	$Y = R1 \div R2$	$Y = \overline{R1 \wedge R2}$
3.	0	0	1	0	$Y = R1 \times R2 - 1$	$Y = \overline{R1} \vee R2$
4.	0	0	1	1	$Y = R1 + R2$	$Y = 1$
5.	0	1	0	0	$Y = R1^2 - R2^2$	$Y = R1 \oplus R2$
6.	0	1	0	1	$Y = R1 \times 2$	$Y = \overline{R1} \Leftrightarrow \overline{R2}$
7.	0	1	1	0	$Y = \frac{R1 + R2}{2}$	$Y = R1 \Rightarrow R2$
8.	0	1	1	1	$Y = R1 \times R2$	$Y = R1 \vee R2$
9.	1	0	0	0	$Y = R1 \,\%\, R2$	$Y = \overline{R1} \Rightarrow \overline{R2}$
10.	1	0	0	1	$Y = R1 \div 2$	$Y = \overline{R1 \wedge R2}$
11.	1	0	1	0	$Y = (R1 \times R2) - (R1+R2)$	$Y = R1 \wedge R2$
12.	1	0	1	1	$Y = R1 - R2$	$Y = \overline{R1 \oplus R2}$
13.	1	1	0	0	$Y = (R1 - R2)^2$	$Y = \overline{R2}$
14.	1	1	0	1	$Y = R1$	$Y = \overline{R1 \vee R2}$
15.	1	1	1	0	$Y = R1^2$	$Y = R1 \Leftrightarrow R2$
16.	1	1	1	1	$Y = R1 - 1$	$Y = \overline{R1} \oplus R2$

Tabela 9.5. Lista operacji dokonywanych przez ALU

UWAGA: *w tabeli 9.5. symbole danych umieszczonych w rejestrach* $\mathbf{R_1}$ *i* $\mathbf{R_2}$ *zapisano jako* R1 *i* R2.

3) Rejestry pomocnicze

Do tej grupy należą rejestry: $\mathbf{R_{WY}}$, $\mathbf{R_A}$, $\mathbf{R_B}$, $\mathbf{R_C}$, $\mathbf{R_1}$, $\mathbf{R_2}$, $\mathbf{R_P}$, $\mathbf{R_I}$. Dwa z nich: $\mathbf{R_P}$ i $\mathbf{R_I}$ są rejestrami 3-bitowymi, pozostałe są 4-bitowe. Wszystkie są połączone z szyną danych.

Można wśród nich wyróżnić:

- układy dwukierunkowe, które mogą pracować jako nadajnik i jako odbiornik danych;
- układy jednokierunkowe, które są wykorzystywane tylko jako nadajnik, albo tylko jako odbiornik danych.

◀ Do grupy układów dwukierunkowych należą rejestry: $\mathbf{R_A}$, $\mathbf{R_B}$, $\mathbf{R_C}$. Zapis informacji z szyny danych do tych rejestrów nastąpi po podaniu na ich wejścia zegarowe odpowiednio sygnałów $\mathbf{C_A}$, $\mathbf{C_B}$, $\mathbf{C_C}$. Odczyt informacji z tych rejestrów i wystawienie jej na szynę danych nastąpi po podaniu na ich wejścia sterujące odpowiednio sygnałów $\mathbf{T_A}$, $\mathbf{T_B}$, $\mathbf{T_C}$.

Podgląd na LED-ach zawartości każdego z rejestrów dwukierunkowych możliwy jest poprzez wciśnięcie odpowiedniego przycisku na klawiaturze $\mathbf{R_A}$, $\mathbf{R_B}$, $\mathbf{R_C}$. W danym momencie możliwy jest podgląd zawartości tylko jednego z rejestrów.

◀ Do grupy układów jednokierunkowych należą rejestry: $\mathbf{R_I}$, $\mathbf{R_1}$, $\mathbf{R_2}$, $\mathbf{R_{WY}}$, $\mathbf{R_P}$, przy czym $\mathbf{R_P}$ może pracować tylko jako nadajnik, a pozostałe – tylko jako odbiorniki.

- Rejestr $\mathbf{R_I}$ służy do przechowywania kodu instrukcji, która jest aktualnie wykonywana przez zestaw laboratoryjny. Ten kod jest jednocześnie adresem obszaru w pamięci RAM, gdzie zapisane są 4 słowa z zakodowanymi poleceniami (mikrocyklami). Mają one być wykonane w czasie realizacji tej instrukcji. Wyjścia rejestru $\mathbf{R_I}$ są połączone bezpośrednio z odpowiednimi wejściami adresowymi RAM-u. Zapis informacji do rejestru następuje po podaniu na jego wejście zegarowe sygnału $\mathbf{C_{RI}}$. Przez cały czas jego zawartość jest wyświetlana na LED-ach umieszczonych na głównej płycie czołowej zestawu.
- Rejestry $\mathbf{R_1}$ i $\mathbf{R_2}$ są układami pełniącymi rolę bufora dla danych wejściowych jednostki arytmetyczno-logicznej. Zapis informacji do nich realizowany jest poprzez podanie na wejście zegarowe tych rejestrów odpowiednio sygnałów $\mathbf{C_{R1}}$ i $\mathbf{C_{R2}}$.
- Rejestr $\mathbf{R_{WY}}$ służy do prezentacji wyników operacji wykonywanych przez zestaw laboratoryjny. Wpisanie do niego informacji nastąpi po podaniu na jego wejście zegarowe sygnału $\mathbf{C_{WY}}$. Zawartość tego rejestru także jest wyświetlana w sposób ciągły na LED-ach umieszczonych na płycie czołowej zestawu laboratoryjnego.
- Rejestr $\mathbf{R_P}$ to zespół ośmiu rejestrów połączonych tak, że możliwy jest sekwencyjny odczyt danych w nich zapisanych. Przechowywany jest w nim program, czyli zbiór kodów instrukcji, które ma wykonać zestaw laboratoryjny i to w takiej kolejności, w jakiej zostały zapisane do tego rejestru. Odczyt kolejnej, zapisanej w nim instrukcji, i wystawienie jej na szynę danych nastąpi każdorazowo po podaniu na jego wejście sterujące sygnału $\mathbf{T_I}$. Na LED-ach: **adres** ($\mathbf{a_2}$, $\mathbf{a_1}$, $\mathbf{a_0}$) wyświetlany jest numer aktualnie wykonywanej instrukcji. Pobieranie kolejnej instrukcji z rejestru programu i wyświetlanie je numeru jest sterowane z licznika, uruchamianego w momencie rozpoczęcia pracy zestawu w trybie **Program**.

9.2.4. ZASADA DZIAŁANIA ZESTAWU

Zestaw laboratoryjny *STEROWANIE SZYNĄ DANYCH* umożliwia modelowanie transmisji danych pomiędzy wybranymi układami pełniącymi rolę nadajników i odbiorników informacji. Przepływ informacji odbywa się za pośrednictwem szyny danych. Wszystkie nadajniki, odbiorniki, jak i sama szyna danych są układami 4 bitowymi, w których zastosowano logikę dodatnią (jedynka logiczna oznacza stan aktywny).

Nadajnikiem może być wyjście jednego z następujących układów:

- rejestr $\mathbf{R_A}$,
- rejestr $\mathbf{R_B}$,
- rejestr $\mathbf{R_C}$,
- jednostka arytmetyczno-logiczna ALU,
- zespół przełączników do zadawania danych wejściowych **dane** ($\mathbf{d_3}$, $\mathbf{d_2}$, $\mathbf{d_1}$, $\mathbf{d_0}$),
- zespół przełączników do wprowadzania kodu instrukcji **instrukcja** ($\mathbf{I_2}$, $\mathbf{I_1}$, $\mathbf{I_0}$) - przy pracy w trybie **Cykl** lub **Mikrocykl**,
- rejestr $\mathbf{R_P}$ - przy pracy w trybie **Program**.

Odbiornikiem może być wejście jednego z następujących układów:

- rejestr $\mathbf{R_A}$,
- rejestr $\mathbf{R_B}$,
- rejestr $\mathbf{R_C}$,
- rejestr pomocniczy $\mathbf{R_1}$,
- rejestr pomocniczy $\mathbf{R_2}$,
- rejestr wyjściowy $\mathbf{R_{WY}}$,
- rejestr instrukcji $\mathbf{R_I}$.

Do szyny danych, którą fizycznie stanowią cztery przewody, dołączone są wszystkie wyjścia nadajników i wejścia odbiorników. Wspomniano już, że układy będące nadajnikami informacji, albo mają wyjścia trójstanowe, albo są podłączone do szyny poprzez bramki z wyjściami trójstanowymi i są sterowane sygnałami $\mathbf{T_i}$ (przy $\mathbf{T_i = 0}$ wyjście i-tego nadajnika jest w stanie wysokiej impedancji).

Zapis oraz odczyt informacji odbywa się poprzez podanie w odpowiedniej sekwencji czasowej sygnałów sterujących: na wejście $\mathbf{T_i}$ i-tego nadajnika i wejście $\mathbf{C_j}$ j-tego odbiornika. Odczyt informacji z nadajnika odbywa się poprzez podanie na wejście wybranego nadajnika stanu wysokiego sygnału sterującego „**T**". Zapis informacji do odbiornika nastąpi po podaniu na wejście sterujące odpowiedniego odbiornika stanu wysokiego sygnału sterującego „**C**".

Sygnały te są generowane przez układ sterujący, który składa się z pamięci RAM, generatora sygnałów GS oraz dwóch dekoderów sygnałów wyjściowych.

W zestawie laboratoryjnym przesyłanie informacji sterowane jest trzybitową instrukcją, zadawaną bezpośrednio z przycisków **instrukcja** ($\mathbf{I_2}$, $\mathbf{I_1}$, $\mathbf{I_0}$) lub przez jej odczyt z rejestru $\mathbf{R_P}$.

Instrukcja, która ma być wykonana przez zestaw laboratoryjny, tzn. wybór rodzaju wykonywanej operacji, wyznaczenie układów nadajnika i odbiornika pomiędzy którymi na nastąpić przepływ danych, wybór źródła danych, itp. definiowana jest przez zestaw czterech 8-bitowych słów sterujących. Są one zapisywane z przełączników C_T, B_T, A_T, C_C, B_C, A_C, W_1, W_2 bezpośrednio do pamięci RAM. Organizacja pamięci RAM i sposób jej programowania został omówiony w p.9.2.3.

Realizowanie przesłań pomiędzy elementami przyłączonymi do szyny danych jest oczywiście możliwe dopiero po zapisaniu pamięci RAM.

Podstawowym trybem pracy zestawu jest **Cykl**. Prześledźmy, jak w tym trybie działa zestaw laboratoryjny.

Ustawiamy na klawiaturze **instrukcja** żądany kod instrukcji, która ma wykonać zestaw. Następnie przez wciśnięcie przycisku **START**, znajdującego się na płycie czołowej zestawu laboratoryjnego inicjujemy rozpoczęcie wykonywania operacji (start pracy układu). Wyjścia generatora GS C_1, C_0, które są podłączone do wejść adresowych RAM-u, będą przyjmowały kolejno wartości 00, 01, 10, 11. Każde przejście pomiędzy stanami oznacza wykonanie przez układ jednego mikrocyklu. W jednej instrukcji, określonej przez słowo adresowe zadane z klawiatury **instrukcja**, mamy więc cztery mikroinstrukcje, a każda z nich wykonywana jest podczas jednego mikrocyklu.

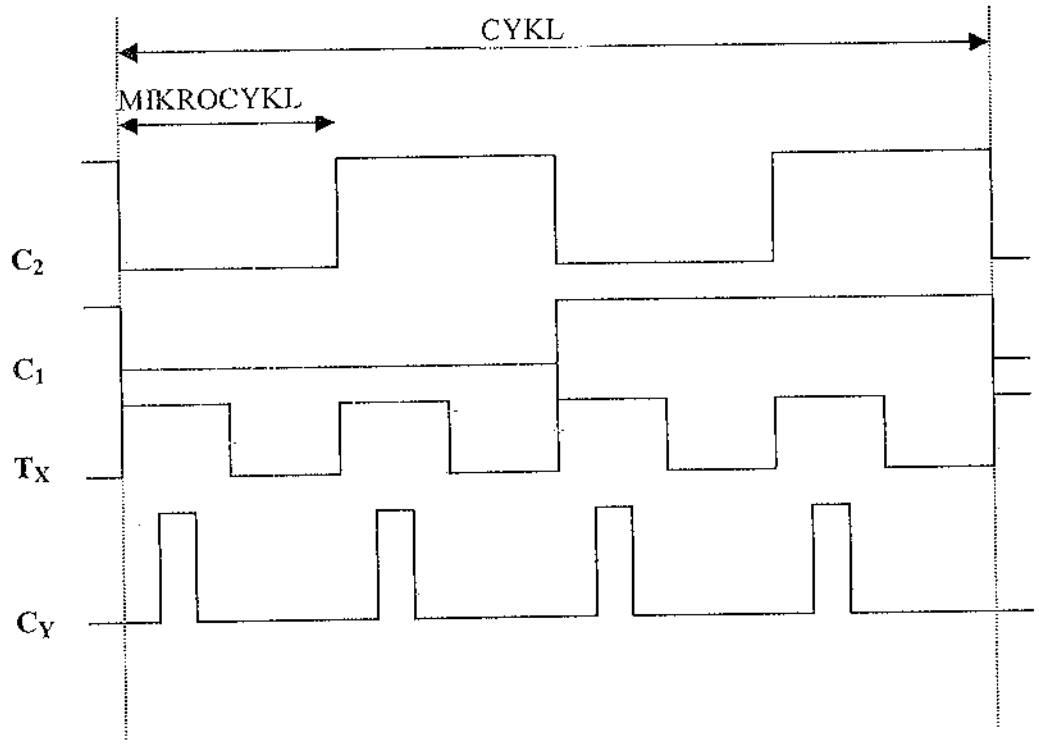
Przyjmijmy, że w pamięci RAM pod adresami xxx00 zostało zapisane słowo 000 000 xx (a tak powinno być zawsze!!!). W takiej sytuacji stan wejść adresowych $C_1 = C_0 = 0$ (niezależnie od stanu pozostałych wejść adresowych) wywołuje z pamięci słowo 000 000 xx. Spowoduje ono pojawienie się na wyjściach dekoderów **1** i **2** (patrz tablice 9.2. 9.3.) sygnałów T_I i C_{RI}. Sygnały te spowodują przesłanie instrukcji ustawionej na klawiaturze **instrukcja**, poprzez szynę danych do rejestru instrukcji R_I, którego wyjścia połączone są z wejściami adresowymi I_2, I_1, I_0 pamięci RAM.

Dalsze trzy stany wejść C_1, C_0 (01, 10, 11) wywołują z pamięci RAM trzy słowa, które sterują trzema przesłaniami informacji zapisanymi w kolejnych mikrocyklach.

W trybie **Cykl** każda instrukcja jest wykonywana w czterech mikrocyklach. W pierwszym mikrocyklu zawsze przesyłana jest instrukcja z klawiatury do rejestru instrukcji, zaś w pozostałych trzech mikrocyklach wykonywane są przesłania nakazane instrukcją.

Przy wykonywaniu instrukcji z użyciem ALU w pierwszym mikrocyklu generowany jest sygnał C_{RI}, tak jak i przy wykonywaniu każdej innej instrukcji. W drugim i trzecim mikrocyklu generowane są sygnały C_{R1} i C_{R2}, które ładują rejestry R_1 i R_2. W tych trzech mikrocyklach impulsy zegarowe są doprowadzane 3-bitowych rejestrów przesuwających i do nich zostają wpisywane wartości bitów W_1 i W_2. W ten sposób do rejestrów przesuwnych zostaje wpisane słowo stanu ALU, które determinuje funkcję, jaką ma wykonać ALU. W ostatnim, czwartym mikrocyklu, generowany jest sygnał dołączający ALU do szyny danych i sygnał zapisu informacji z szyny do wybranego odbiornika – różnego od R_1 i R_2.

Wyjścia dekoderów **1** i **2**.które generują sygnały T_i i C_j są strobowane sygnałami T_X i C_Y generowanymi w takt sygnałów C_1 i C_0, jak to pokazano to na Rys.9.2. Eliminuje to wpływ hazardów na pracę układów w zestawie.



Rys.9.2. Zależności czasowe pomiędzy impulsami C_1, C_0 a impulsami T_X, C_Y strobującymi dekodery.

9.2.5. ZASADY WYKORZYSTYWANIA ZESTAWU

Wykorzystywanie zestawu laboratoryjnego *STEROWANIE SZYNĄ DANYCH* odbywa się zawsze w dwóch, następujących po sobie, etapach:
- etap pierwszy: PROGRAMOWANIE;
- etap drugi: PRACA.

♦ PROGRAMOWANIE

Realizacja tego etapu wymaga wykonania następujących czynności:
- zapisania pamięci RAM,
- zapisania programu w rejestrze programu, czyli w $\mathbf{R_P}$.

1) Zapisanie RAM-u

Już wcześniej zaznaczono, że wykorzystywanie zestawu laboratoryjnego do modelowania różnych przesłań za pośrednictwem szyny danych będzie możliwe dopiero po zapisaniu RAM-u, czyli zaprogramowaniu sposobu działania tego urządzenia.

Przypomnijmy, że w każdych czterech kolejnych wierszach pamięci RAM są zapisane kody mikroinstrukcji, jakie muszą zostać wykonane po wywołaniu instrukcji wybranej do realizacji przez zestaw.

Przy projektowaniu poszczególnych instrukcji należy pamiętać, że ze względu na konstrukcję zestawu trzeba spełnić wymagania co do kolejności umieszczania niektórych mikroinstrukcji w poszczególnych mikrocyklach.

Z analizy tabel pracy dekoderów **1** i **2** (Tabela.9.1. i Tabela.9.2.) wynika, że aby w pierwszym mikrocyklu została pobrana instrukcja z klawiatury **instrukcja** (albo z rejestru programu $\mathbf{R_P}$) i wpisana do rejestru instrukcji $\mathbf{R_I}$, muszą w nim być wygenerowane sygnały $\mathbf{T_I}$ i $\mathbf{C_{RI}}$. W każdym pierwszym mikrocyklu do pamięci RAM musi być wpisane słowo 000 000 xx. W pozostałych mikrocyklach wpisujemy kody mikroinstrukcji, których wykonanie jest niezbędne do realizacji programowanej instrukcji.

Specjalne warunki należy spełnić przy projektowaniu instrukcji, w których wykorzystuje się jednostkę arytmetyczno-logiczną. Zagadnienie to zostało wyjaśnione w p.9.2.4., podp.2)., a sposób kodowania funkcji realizowanej przez ALU przedstawiono w tabeli 9.4. Wymagania tam podane muszą być bezwzględnie spełnione, albowiem przy innym sposobie zakodowania mikrocykli nie zostanie poprawnie ustawione słowo stanu, które ustala operację jaką wykona ALU.

2) Zapisanie programu

Polega ono na zapisaniu w pamięci programu, czyli w $\mathbf{R_P}$ zestawu instrukcji, które zestaw ma wykonać jednorazowo. Jednorazowo można wpisać 8 instrukcji. Oczywiście nie trzeba do tego rejestru wpisywać wszystkich instrukcji, które zostały zaprogramowane. Można także zażądać, aby jedna instrukcja została wykonania kilka razy -- wtedy jej kod wpisujemy kilkukrotnie. Instrukcje zostaną wykonane w kolejności wpisania do $\mathbf{R_P}$.

Zapisanie rejestru programu nie jest konieczne, aby przystąpić do etapu **PRACA**, ale zestaw nie będzie mógł wtedy pracować w trybie **Program**.

Po zakończeniu etapu programowania możemy przystąpić do drugiego etapu, tzn. do właściwej pracy z zestawem.

♦ **PRACA**

W tym etapie możliwe jest wykonywanie zapisanych wcześniej instrukcji oraz programów, a także wykonywanie prostych obliczeń z wykorzystaniem jednostki arytmetyczno-logicznej i modelowanie różnych przesłań za pomocą szyny danych.

Zestaw laboratoryjny *STEROWANIE SZYNĄ DANYCH* ma przewidziane trzy tryby pracy:

- **Mikrocykl**,
- **Cykl**,
- **Program**.

Wyboru trybu pracy zestawu dokonuje się przez wciśnięcie jednego z trzech przełączników zależnych, które znajdują się na płycie pulpitu i są opisane nazwami podanymi powyżej.

1) Mikrocykl

W tym trybie realizacja instrukcji zadanej z klawiatury **instrukcja** odbywa się na raty – po wciśnięciu przycisku **START** nastąpi tylko jedna zmiana stanów wyjść C_1, C_0 generatora GS. Zestaw wykonana jedną mikroinstrukcję i będzie oczekiwał na ponowne wciśnięcie przycisku **START**.

2) Cykl

W tym trybie, po wciśnięciu przycisku **START** zostanie zrealizowana cała instrukcja zadana z klawiatury **instrukcja**. Nastąpią kolejne cztery zmiany stanów wyjść C_1, C_0 generatora GS.

3) Program

W tym trybie, po wciśnięciu przycisku **START** zostanie zrealizowane 8 instrukcji w takiej kolejności, w jakiej zostały zapisane w rejestrze programu R_P. Układ wygeneruje 32 kolejne stany wyjść C_1, C_0.

Jeżeli w czasie pracy zestawu pojawi się instrukcja czytania z wejścia, układ sterowania zatrzymuje się. Zapala się LED **Gotów**, który zgaśnie po wciśnięciu przycisku **CZYTAJ**. Nastąpi wtedy przepisanie danych z klawiatury **dane** do zaprogramowanego odbiornika i układ zostanie ponownie uruchomiony.

9.3. OBSŁUGA ZESTAWU LABORATORYJNEGO

Zestaw laboratoryjny musi zostać przygotowany do pracy. Należy w tym celu zdefiniować listę instrukcji, które mają być wykonywane przez zestaw laboratoryjny *STEROWANIE SZYNĄ DANYCH*, a następnie trzeba zapisać w pamięci RAM odpowiednie kody, zapewniające realizacje zaprojektowanych operacji.

Listę rozkazów najlepiej jest umieścić w tabeli, o kształcie jak Tabela.9.1. Następnie możemy przystąpić do sekwencyjnego zapisu danych do pamięci RAM, a w dalszej części do wykonania zapisanego programu.

9.3.1. ZAPIS PROGRAMU DO PAMIĘCI RAM

W tym celu należy wykonać następujące czynności:

a) Przełącznik **zapis/praca** ustawić w pozycji **zapis** - *wciśnięty*.

b) Wyzerować adres w pamięci RAM ($\mathbf{I_2}$, $\mathbf{I_1}$, $\mathbf{I_0}$) oraz stan wyjść adresowych generatora GS ($\mathbf{C_1}$, $\mathbf{C_0}$) poprzez naciśnięcie odpowiadającym im odpowiednich przycisków **zer.**

c) Na klawiaturze ustawić wartość słowa sterującego $\mathbf{C_T}$, $\mathbf{B_T}$, $\mathbf{A_T}$, $\mathbf{C_C}$, $\mathbf{B_C}$, $\mathbf{A_C}$, $\mathbf{W_1}$, $\mathbf{W_2}$, a następnie dokonać wpisu tego słowa do pamięci, pod adresem wyświetlanym przez diody $\mathbf{I_2}$, $\mathbf{I_1}$, $\mathbf{I_0}$, $\mathbf{C_1}$, $\mathbf{C_0}$, poprzez naciśnięcie przycisku **wpisz**. Stan wyjść adresowych $\mathbf{I_2}$, $\mathbf{I_1}$, $\mathbf{I_0}$, $\mathbf{C_1}$, $\mathbf{C_0}$ zostanie automatycznie zwiększony o jeden.

d) Powtórzyć czynności z pkt. c) aż do zapisania wszystkich instrukcji.

Zapis programu do rejestru programu $\mathbf{R_P}$ odbywa się analogicznie jak zapis do pamięci RAM.

9.3.2. WYKONANIE INSTRUKCJI ZAPISANYCH W PAMIĘCI

W tym celu należy wykonać następujące czynności:

a) Przełącznik **zapis/praca** ustawić w pozycji **praca** - *wyciśnięty*.

b) W zależności od interesującego nas trybu pracy programu wybrać **Mikrocykl**, **Cykl** lub **Program**.

c) Wyzerować adres w pamięci RAM (($\mathbf{I_2}$, $\mathbf{I_1}$, $\mathbf{I_0}$) oraz stan wyjść adresowych generatora GS ($\mathbf{C_1}$, $\mathbf{C_0}$) poprzez naciśnięcie odpowiadających im przycisków **zer.**

d) Wcisnąć przycisk **START**. Układ rozpocznie wykonywanie zapisanego programu.

e) W razie potrzeby (świecąca dioda **Gotów**) dokonywać wpisu danych wejściowych z klawiatury **Dane** przez naciśnięcie przycisku **CZYTAJ**.

f) Po zakończeniu wykonywania **Mikrocyklu** bądź **Cyklu** wznowienie wykonywania programu następuje przez ponowne naciśnięcie przycisku **START**. Należy pamiętać o ustawieniu odpowiedniej zawartości rejestru instrukcji $\mathbf{R_I}$.

9.3.3. KOREKTA PROGRAMU ZAPISANEGO W PAMIĘCI RAM

W przypadku wystąpienia błędu w zapisanym programie istnieje możliwość zapisu poprawnego słowa z klawiatury C_T, B_T, A_T, C_C, B_C, A_C, W_1, W_2 i korekta nieprawidłowej części programu. Aby tego dokonać należy postępować jak w przypadku schematu zapisu nowego programu przedstawionego w pkt.9.3.1., z tą różnicą, że zamiast dokonywać zerowania adresu w pamięci RAM, na klawiaturze I_2, I_1, I_0 ustawiamy adres pamięci pod którym chcemy zapisać poprawną wartość słowa sterującego.

9.4. PROGRAM ĆWICZENIA

9.4.1. PRZYGOTOWANIE DO ZAJĘĆ

1. W domu należy rozwiązać zadania otrzymane od prowadzącego. W przypadku, gdy zadany problem wymaga wykonania ciągu kilku złożonych obliczeń z wykorzystaniem ALU, wskazane jest przygotowanie algorytmu wykonywanych obliczeń w postaci schematu operacyjnego (jak przy programowaniu).
2. Następnie należy przygotować listę rozkazów potrzebnych do realizacji programu. Najlepiej jest przygotować ją w postaci podanej w Tab.9.1.

9.4.1. PRZEBIEG ĆWICZENIA W LABORATORIUM

1. W laboratorium, po zaakceptowaniu przez prowadzącego przygotowanego w domu rozwiązania problemu należy zapisać zaprojektowane instrukcje w pamięci RAM, oraz program w pamięci R_P. Postępowanie w trakcie tych czynności opisano w p. 9.3.
2. Po zapisaniu instrukcji w RAM i programu w R_P należy sprawdzić poprawność działania programu na przygotowanych przez siebie danych, a następnie zademonstrować go prowadzącemu. Prowadzący może wymagać przeprowadzenia demonstracji przy wykorzystaniu danych podanych przez niego.
3. Po zaakceptowaniu przez prowadzącego przedstawionego rozwiązania, należy wykonać jego ewentualne dodatkowe polecenia. Po ich wykonaniu należy uzyskać potwierdzenie wykonania ćwiczenia.

9.5. PRZYKŁADOWE ZADANIA PROJEKTOWE

1. Przygotować taki zestaw instrukcji (maksymalnie 8), który będzie wczytywał dwie liczby 4 - bitowe $A(a_3a_2a_1a_0)$ i $B(b_3b_2b_1b_0)$ podawane przez prowadzącego zajęcia, a następnie wykona na nich następującą operacje (jedną z podanych wybiera prowadzący):
   a. $C = \frac{(A \cdot B) - (A + B) + B}{4}$
   b. $C = (A \oplus B) + (\bar{A} \vee \bar{B})$
   c. $C = (A^2 - B^2 + (A - B)^2) \wedge B$

   Liczbę wyjściową $C(c_3c_2c_1c_0)$ zapisać do rejestru wyjściowego $\mathbf{R_{WY}}$. Należy przygotować zestaw danych, na których będzie można sprawdzić poprawność działania programu. Sprawdzić działanie w dwóch trybach pracy: CYKL i PROGRAM.
2. Przygotować taki zestaw instrukcji (maksymalnie 8), który będzie wczytywał dwie liczby 4 - bitowe $A(a_3a_2a_1a_0)$ i $B(b_3b_2b_1b_0)$ podawane przez prowadzącego zajęcia, a następnie będzie wpisywało do rejestru wyjściowego $\mathbf{R_{WY}}$ liczby tworzące ciąg Fibonacciego (kolejna liczba jest sumą dwóch poprzednich): najpierw C=A+B, potem D=B+C, następnie E=C+D, etc. Określić, ile można wygenerować takich liczb za pomocą 8 instrukcji (wliczając w nie wczytanie liczb z klawiatury). W przypadku przepełnienia – przeniesienie należy zaniedbać. Sprawdzić działanie w dwóch trybach pracy: CYKL i PROGRAM.
3. Przygotować taki zestaw instrukcji (maksymalnie 8), który będzie wczytywał liczbę 4 - bitową $A(a_3a_2a_1a_0)$ podawaną przez prowadzącego zajęcia, a następnie będzie wpisywał do rejestru wyjściowego $\mathbf{R_{WY}}$ liczby będące o 1 mniejszą niż poprzednia: A, A-1, A-2, A-3, A-4, etc. Program ma działać w nieskończonej pętli. Sprawdzić działanie w dwóch trybach pracy: CYKL i PROGRAM.
4. Przygotować taki zestaw instrukcji (maksymalnie 8), który będzie wczytywał liczbę 4 - bitową $A(a_3a_2a_1a_0)$ podawaną przez prowadzącego zajęcia, a następnie będzie wpisywał do rejestru wyjściowego $\mathbf{R_{WY}}$ liczby będące o 2 większe niż poprzednia: A, A+2, A+4, A+6, A+8, etc. Program ma działać w nieskończonej pętli. Sprawdzić działanie w dwóch trybach pracy: CYKL i PROGRAM.
5. Przygotować taki zestaw instrukcji (maksymalnie 8), który będzie wczytywał dwie liczby 4 - bitowe $A(a_3a_2a_1a_0)$ i $B(b_3b_2b_1b_0)$ podawane przez prowadzącego zajęcia. Następnie należy sprawdzić, czy liczba B jest równa kwadratowi liczby A. Jeżeli $B=A^2$, należy B przesłać do rejestru $\mathbf{R_A}$, w przeciwnym przypadku – do rejestru $\mathbf{R_A}$ trzeba wpisać 0. Sprawdzić działanie w dwóch trybach pracy: CYKL i PROGRAM.
6. Przygotować taki zestaw instrukcji (maksymalnie 8), który będzie wczytywał liczbę 4 - bitową $A(a_3a_2a_1a_0)$ podawaną przez prowadzącego zajęcia, a następnie, jeżeli $a_3=0$, będzie wpisywał do rejestru wyjściowego $\mathbf{R_{WY}}$ liczbę zanegowaną względem A; natomiast jeżeli $a_3=1$, to będzie wpisywał liczbę 2 razy mniejszą niż A. Sprawdzić działanie w dwóch trybach pracy: CYKL i PROGRAM.
7. Przygotować taki zestaw instrukcji (maksymalnie 8), który będzie wczytywał liczbę 4 - bitową $A(a_3a_2a_1a_0)$ podawaną przez prowadzącego zajęcia, a następnie, jeżeli $A \geq 8$ to będzie dzielił tę liczbę na dwa i wpisywał do rejestru wyjściowego $\mathbf{R_{WY}}$, jeżeli $A < 8$ – będzie wpisywał A do rejestru wyjściowego $\mathbf{R_{WY}}$. Następnie, prowadzący zajęcia musi podać

liczbę, która zostanie dodana do ostatniego wyniku i ta liczba ma zostać wpisana do rejestru wyjściowego R_{WY}. Sprawdzić działanie w dwóch trybach pracy: CYKL i PROGRAM.

8. Zdefiniować taki zestaw instrukcji, aby możliwe było wykonanie następujących operacji:

1) **dane** → R_A,
2) **dane** → R_B,
3) R_A → R_1,
4) R_B → R_2,
5) $R_1 + R_2$ (arytm)→ R_C,
6) R_C → R_{WY},
7) Wyzeruj R_C

- Przed przystąpieniem do testowania instrukcji należy przygotować zestawy danych, które będą podawane jako argumenty przy wykonywaniu poszczególnych instrukcji i wykonać obliczenia kontrolne dla instrukcji 5).
- Sprawdzić działanie zaprojektowanych instrukcji w trzech trybach pracy zestawu laboratoryjnego: CYKL, MIKROCYKL i PROGRAM.

9.6. PRZYKŁADOWE ROZWIĄZANIE ZADANIA PROJEKTOWEGO

ZADANIE

Wczytać dwie liczby 4-bitowe. Na dwóch młodszych bitach obydwu liczb należy wykonać operację sumy logicznej, a na dwóch starszych bitach – sumy arytmetycznej. Wyniki tych dwóch działań skleić w liczbę 4-bitową i przesłać do rejestru wyjściowego.
W ramach rozwiązania należy przygotować wszystkie dane potrzebne do zaprogramowania pamięci RAM – najlepiej w postaci tabeli, wg wzoru podanego w tabeli 9.1.
UWAGA: przy dodawaniu nie uwzględniamy ewentualnego przeniesienia z najstarszej pozycji, bo w zestawie *nie jest ono wyprowadzone z ALU.*

ROZWIĄZANIE

1) Uwagi wstępne

Zadanie to można rozwiązać na kilka sposobów. Przy wyborze wariantu do realizacji trzeba wziąć przede wszystkim pod uwagę to, jakie instrukcje są dostępne w jednostce arytmetyczno-logicznej znajdującej się w zestawie.

Dodatkowym ograniczeniem jest to, aby tak zaplanować obliczenia, by do ich zrealizowania wystarczyło użycie 8 różnych instrukcji – tyle można maksymalnie zapisać w pamięci RAM zestawu laboratoryjnego.

Ewentualnie można zaakceptować takie rozwiązanie, w którym zostało zaprojektowanych 8 różnych instrukcji, ale do wykonania zadanych obliczeń trzeba byłoby niektóre z nich użyć więcej niż jeden raz. Oczywiście wtedy nie można byłoby takich obliczeń realizować w trybie **Program**. Należałoby wtedy wykorzystać tryb **Cykl**.

2) Założenia wstępne

Struktura danych i wyniku

- Dane są dwie liczby, **X** i **Z**, które można zapisać w postaci: **X** (x_3, x_2, x_1, x_0) i **Z** (z_3, z_2, z_1, z_0):
- Wynik będzie miał postać: **W** (w_3, w_2, w_1, w_0).

3) Algorytm obliczeń

Poniżej przedstawiono w punktach przykładowy algorytm obliczeń przy realizacji zadania. Odrabiającym ćwiczenie pozostawia się narysowanie tego algorytmu w postaci schematu operacyjnego.

a) Wczytać dane do rejestrów R_A i R_B: **X** → R_A, **Z** → R_B.

b) Obliczyć sumę logiczną liczb **X** i **Z**, a wynik zapisać w rejestrze R_C.

c) Nałożyć maskę 0011 na liczbę w rejestrze R_C. *(W tej sytuacji w rejestrze R_C zostanie zapisana liczba o postaci (0, 0, w_1, w_0), gdzie dwa młodsze bity to wynik żądanej w zadaniu operacji na młodszych bitach liczb wejściowych).*

d) Nałożyć maskę 1100 na liczbę w rejestrze $\mathbf{R_A}$. *{Po wykonaniu tej operacji w rejestrze R_A zostanie zapisana liczba o postaci $(x_3, x_2, 0, 0)$}.*

e) Nałożyć maskę 1100 na liczbę w rejestrze $\mathbf{R_B}$. *{Po wykonaniu tej operacji w rejestrze R_B zostanie zapisana liczba o postaci $(z_3, z_2, 0, 0)$}.*

f) Obliczyć sumę arytmetyczną liczb znajdujących się aktualnie w rejestrach $\mathbf{R_A}$ i $\mathbf{R_B}$. Wynik operacji ma zostać zapisany w rejestrze $\mathbf{R_B}$. *{W tej sytuacji w rejestrze R_B zostanie zapisana liczba o postaci $(w_3, w_3, 0, 0)$, gdzie dwa starsze bity to wynik żądanej w poleceniu zadaniu operacji na starszych bitach liczb wejściowych}.*

g) Wyzerować rejestr wyjściowy $\mathbf{R_{WY}}$.

h) Skleić dwie liczby: z rejestrów $\mathbf{R_B}$ i $\mathbf{R_C}$ w jedna liczbę 4-bitową i przesłać ja do rejestru wyjściowego $\mathbf{R_{WY}}$. *{W tej sytuacji w rejestrze R_{WY} zostanie wynik żądanych w poleceniu zadaniu operacji}.*

4) Kodowanie poszczególnych instrukcji

Na podstawie algorytmu obliczeń należy podzielić zaplanowane operacje na mikroinstrucje, które mają być wykonane w poszczególnych mikrocyklach i zakodować je zgodnie z tabelami 9.2., 9.3. i 9.5.

Sposób zakodowania mikroinstrukcji (w poszczególnych mikrocyklach), potrzebnych do wykonania działań przedstawionych powyżej podano w tablicy 9.6.

5) Przygotowanie programu

Zadane obliczenia można wykonać w zestawie laboratoryjnym w różnych trybach pracy. Jeżeli chcemy to zrobić w trybie **Program**, musimy wcześniej dokonać zapisu w pamięci programu $\mathbf{R_P}$. W tym przykładowym rozwiązaniu instrukcje zostały tak zaprojektowane, że aby zrealizować zadane polecenia, należy wszystkie instrukcje wykonać w takiej kolejności, w jakiej zostały zapisane w pamięci RAM. W takiej też kolejności należy je zapisać do pamięci programu $\mathbf{R_P}$.

Dane do zapisania w pamięci programu $\mathbf{R_P}$ przedstawiono w tabeli 9.7.

UWAGA:
Instrukcje w pamięci programu R_P nie muszą być zapisywane w takiej kolejności, jak zostały zapisane w pamięci RAM. Kolejność ich przywoływania w R_P może być inna, w zależności od potrzeb i oczywiście od sposobu ich zaprojektowania.

operacja	I_2	I_1	I_0	C_1	C_0	C_T	B_T	A_T	C_C	B_C	A_C	W_1	W_2
a	0	0	0	0	0	0	0	0	0	0	0	X	X
				0	1	1	0	1	0	1	0	X	X
				1	0	0	1	0	0	0	1	X	X
				1	1	1	0	1	0	1	0	X	X
b	0	0	1	0	0	0	0	0	0	0	0	1	0
				0	1	0	0	1	1	0	0	1	1
				1	0	0	1	0	1	0	1	X	1
				1	1	1	0	0	0	1	1	X	X
c	0	1	0	0	0	0	0	0	0	0	0	1	1
				0	1	0	1	1	1	0	0	0	1
				1	0	1	0	1	1	0	1	X	0
				1	1	1	0	0	0	1	1	X	X
d	0	1	1	0	0	0	0	0	0	0	0	1	1
				0	1	0	0	1	1	0	0	0	1
				1	0	1	0	1	1	0	1	X	0
				1	1	1	0	0	0	0	1	X	X
e	1	0	0	0	0	0	0	0	0	0	0	1	1
				0	1	0	1	0	1	0	0	0	1
				1	0	0	1	0	1	0	0	X	0
				1	1	1	0	0	0	1	0	X	X
f	1	0	1	0	0	0	0	0	0	0	0	0	0
				0	1	0	0	1	1	0	0	0	1
				1	0	0	1	0	1	0	1	X	1
				1	1	1	0	0	0	1	0	X	X
g	1	1	0	0	0	0	0	0	0	0	0	1	0
				0	1	0	0	1	1	0	0	1	0
				1	0	0	0	1	1	0	1	X	0
				1	1	1	0	0	1	1	0	X	X
h	1	1	1	0	0	0	0	0	0	0	0	1	0
				0	1	0	1	0	1	0	0	1	1
				1	0	0	1	1	1	0	1	0	1
				1	1	1	0	0	1	1	0	X	X

Tabela.9.6. Tabela z danymi do zaprogramowania pamięci RAM – dla przykładowego rozwiązania zadania projektowego.

NR operacji	kod instrukcji		
	I_2	I_1	I_0
1	0	0	0
2	0	0	1
3	0	1	0
4	0	1	1
5	1	0	0
6	1	0	1
7	1	1	0
8	1	1	1

Tabela.9.7. Tabela z danymi do zapisania w pamięci programu R_P – dla przykładowego rozwiązania zadania projektowego.