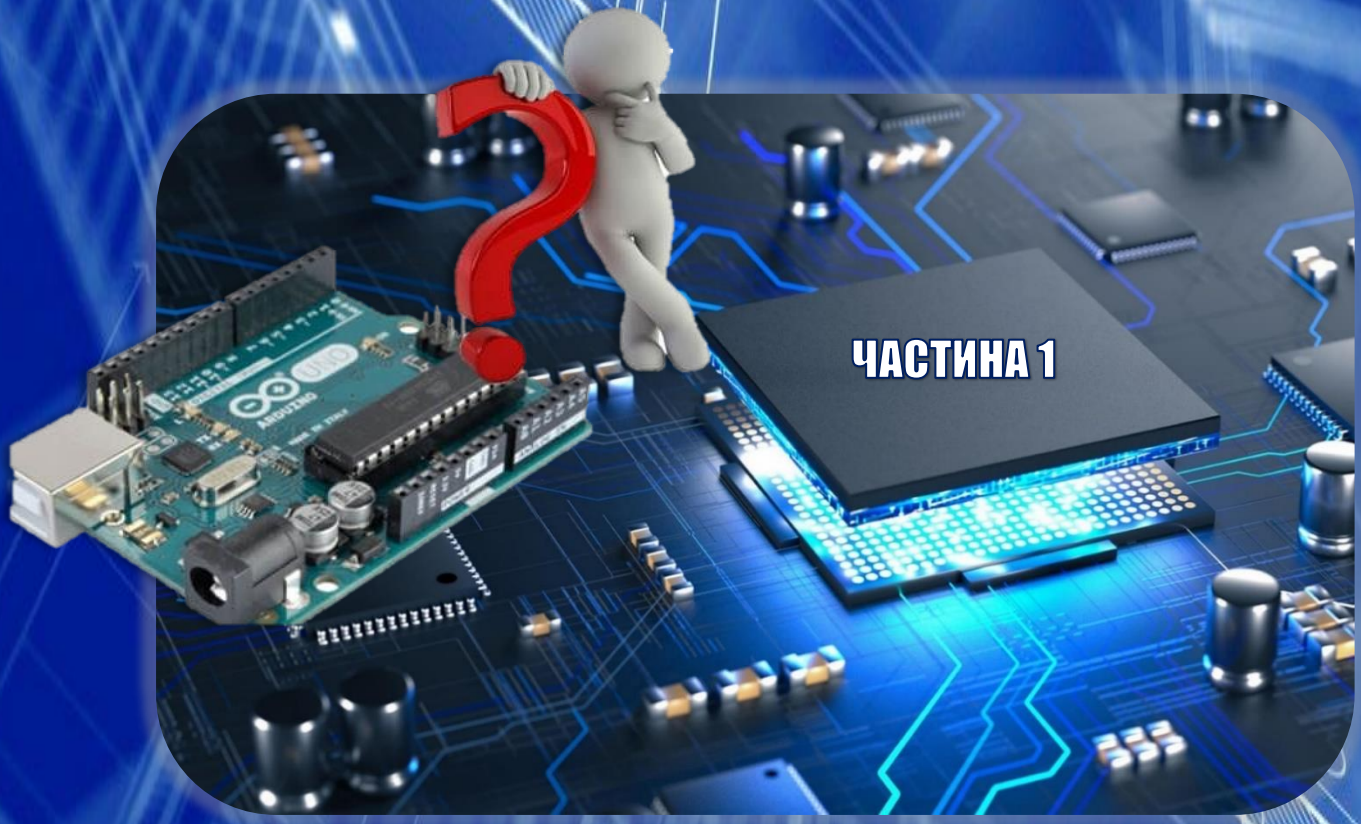
Міністерство освіти і науки України
Державний університет телекомунікацій

Кисіль Т.М., Фесенко М.А., Зінченко О.В.

ОСНОВИ МІКРОПРОЦЕСОРНОЇ ТЕХНІКИ



Методичні рекомендації до виконання практичних завдань
для здобувачів ступеня бакалавра освітньої програми
«Штучний інтелект» за спеціальністю 122 «Комп'ютерні науки»

Київ – 2022

Міністерство освіти і науки України

Державний університет телекомунікацій

Кисіль Т.М., Фесенко М.А., Зінченко О.В.

ОСНОВИ МІКРОПРОЦЕСОРНОЇ ТЕХНІКИ

ЧАСТИНА 1

Методичні рекомендації до виконання практичних завдань для здобувачів ступеня бакалавра освітньої програми «Штучний інтелект» спеціальності 122 «Комп'ютерні науки»

Київ – 2022

УДК 621.3

Рецензенти: **Іщеряков С.М.**, кандидат технічних наук, доцент, доцент кафедри Комп'ютерних наук Державного університету телекомунікацій.
Кузавков В.В., доктор технічних наук, доцент, начальник кафедри Побудови телекомунікаційних систем Військового інституту телекомунікацій та інформатизації імені Героїв Крут.

Кисіль Т.М., Фесенко М.А., Зінченко О.В.

Основи мікропроцесорної техніки: частина 1. – Методичні рекомендації до виконання практичних завдань для здобувачів ступеня бакалавра освітньої програми «Штучний інтелект» за спеціальністю 122 «Комп'ютерні науки». – К: ДУТ, 2022. – 125 с.

Методичні рекомендації містять практичні завдання основ мікропроцесорної техніки: приклади виконання завдань, тексти лабораторних та практичних робіт (контрольні питання, завдання до самостійного виконання). Методичні рекомендації використовуються на заняттях дисципліни «Основи мікропроцесорної техніки» для студентів спеціальності 122 «Комп'ютерні науки» освітньої програми «Штучний інтелект»

Рекомендовано на засіданні вченої ради Навчально-наукового інституту інформаційних технологій (Протокол № 2 від 12.09.2022 року)

© ДУТ 2022 рік

ЗМІСТ

ПЕРЕДМОВА 5

Практичне завдання 1. Представлення цифрових даних в позиційних системах числення. Побудова основних логічних елементів в Logic.Ly 6

Практичне завдання 2. Побудова основних логічних елементів за допомогою дискретних компонентів та їх представлення на електричних принципових схемах 20

Практичне завдання 3. Побудова інтегрованих логічних елементів та їх реалізація на макетних, печатних платах..... 30

Практичне завдання 4. Дослідження принципів роботи комбінаційних мікросхем та їх реалізація на макетних, печатних платах. 39

Практичне завдання 5. Машинні коди. Виконання арифметичних операцій з цілими та дійсними двійковими числами. 56

Практичне завдання 6. Дослідження принципів роботи комбінаційних цифрових приладів та їх реалізація на макетних, печатних платах..... 77

Практичне завдання 7. Моделювання пристроїв на базі мікроконтролера ArduinoUno..... 97

Практичне завдання 8. Вивчення принципів роботи портів мікроконтролера Arduino Nano..... 114

Список використаних джерел..... 125

ПЕРЕДМОВА

Методичні вказівки «Основи мікропроцесорної техніки» призначені для студентів освітньої програми «Штучний інтелект» спеціальності 122 «Комп'ютерні науки». Метою вивчення дисципліни «Основи мікропроцесорної техніки» є засвоєння студентами методів розробки мікропроцесорних пристроїв управління на базі мікроконтролерів. Основними завданнями навчальної дисципліни «Основи мікропроцесорної техніки» є вивчення апаратної будови мікропроцесорних систем управління та основ їх програмування.

Метою практичних завдань з дисципліни «Основи мікропроцесорної техніки», які виконуються на базі програмного забезпечення в середовищі програмування мікропроцесорів ***Arduino Uno***, ***Arduino Mega***, ***Arduino NANO***, лабораторного макету «***Цифрова електроніка***» та/або віртуальної моделі лабораторного макету з використанням програмного та інформаційного забезпечення засобів ***Logic.Ly, EasyEDA***, ***TinkerCad***, ***Arduino 7.0*** є: проектування та симуляція печатних плат, вивчення принципів побудови електронних компонентів мікропроцесорних систем на основі процесорів Arduino; придбання навичок роботи з сенсорами та датчиками мікропроцесорного управлінням; придбання навичок підготовки, проведення та документування результатів вимірювання і функціонування мікропроцесорних систем.

В методичних рекомендаціях розглянуто програмно-апаратний комплекс Arduino. Описано особливості реалізації основних функцій для проектування мікропроцесорних систем на базі плат Arduino для управління об'єктом середньої складності, розглянуті основні функції, які використовуються для керування периферією, форматування даних та обміну даними. В методичних рекомендаціях наведено практичні завдання лабораторних робіт різного ступеня складності які базуються на програмно-апаратному комплексі Arduino. Студентам для початку пропонуються прості завдання для набуття практичних навичок роботи з компонентами електроніки та базових навичок програмування в середовищі Arduino. Надалі виконуються більш складні завдання. В результаті виконання курсу практичних завдань студенти зможуть створювати та налагоджувати програми у середовищі розробки Arduino.

Результати проведених досліджень повинні бути задокументовані та представлені на при кінці заняття викладачу. До виконаної роботи складається звіт, який повинен містити: титульний аркуш; короткі теоретичні відомості за виконаним дослідженням; схеми, програмний код, висновки за результатами досліджень; відповіді на контрольні питання.

Методичні рекомендації складено на основі лекційних матеріалів, які викладаються студентам відповідно до навчальної програми курсу «Основи мікропроцесорної техніки» напряму підготовки 122 «Комп'ютерні науки» освітньої програми «Штучний інтелект». Дисципліна «Основи мікропроцесорної техніки» є базовою та надає підґрунтя для вивчення таких дисциплін як, «Технології Smart-систем» та «Робототехніка» на старших курсах освітньої програми бакалаврського рівня освіти в Державному університеті телекомунікацій.

Практичне завдання 1.
Представлення цифрових даних в різних позиційних системах числення. Побудова основних логічних елементів в Logic.Ly

Мета роботи: опрацювання методів переведення цілих та дійсних чисел з однієї позиційної системи числення в інші: із 10-системи числення в 2-систему/8-систему/16-систему числення, із 2-системи числення в 8-систему/10-систему/16-систему числення; із 16-системи числення в 10-систему/8-систему числення та ін.; закріпити знання по основах роботи з булевими функціями І, АБО, НІ; набуття практичних навичок в складанні таблиць істинності, логічних виразів, простих логічних схем та в аналізі комбінаційних схем в ***Logic.Ly***.

Завдання 1

Виконати переведення цілих та дійсних чисел із однієї позиційної системи числення в іншу, відповідно до наведених в табл. 1 завдань.

Таблиця 1. – Варіанти до виконання завдання

Завдання	*Розрахунок варіанту завдання*
1. Перевести рік народження студента в двійкову систему числення	Якщо студент народився у 2002 році, тоді число $\mathbf{2002_{10} \rightarrow ?_2}$
2. Перевести дату народження студента в двійкову систему числення	Якщо студент народився у 17.08.2002 року, тоді число $\mathbf{17082002_{10} \rightarrow ?_2}$
3. Перевести число, сформоване за №групи та №студента в списку групи із 8–системи числення у десяткову	Якщо студент навчається в групі ШІД-12 і має номер 27 в списку групи, то необхідно перевести число $\mathbf{1227_8 \rightarrow ?_{10}}$
4. Перевести число та місяць народження студента із 10-системи числення у 8–систему	Якщо студент народився 17 липня, тоді необхідно перевести ціле число $\mathbf{1707_{10} \rightarrow ?_8}$
5. Перевести дату народження студента із 8-системи числення в двійкову систему	Якщо студент народився у 17.07.2002 року, тоді необхідно перевести дробове число $\mathbf{1707{,}2002_8 \rightarrow ?_{10}}$
6. Перевести число та місяць народження студента із двійкової системи числення у десяткову	Якщо студент народився 17 серпня, тоді необхідно перевести дійсне число $\mathbf{0{,}1708_{10} \rightarrow ?_2}$
7. Перевести поточну дату в двійкову систему числення методом ділення	Якщо поточна дата 8.02.2022, то перевести число $\mathbf{8022022_{10} \rightarrow ?_2}$
8. Виконати перевірку отриманого двійкового числа в ***завданні 7*** в десяткову систему числення.	Якщо поточна дата 8.02.2022, то перевести число $\mathbf{8022022_{10} \rightarrow ?_2}$, а також $\mathbf{?_2 \rightarrow ?_{10}}$
9. Перевести дату народження студента із 10-системи числення в двійкову за методом еквіваленту степені Виконати перевірку	Якщо студент народився у 17.08.2002 року, тоді необхідно перевести дійсне число $\mathbf{1708{,}2002_{10} \rightarrow ?_2}$; виконати

Завдання	*Розрахунок варіанту завдання*
отриманого двійкового числа в десяткову систему числення	перевірку отриманого результату $?_2 \rightarrow ?_{10}$
10. Перевести дату народження студента перевести в 16-систему числення. Використовуючи тетради, отриманий результат, перевести в 2-систему числення	Якщо студент народився у 17.08.2002 року, тоді число $\mathbf{1708{,}2002_{10} \rightarrow ?_{16}}$ отриманий результат перевести $?_{16} \rightarrow ?_2$
11. Перевести дату народження студента в 8-систему числення. За допомогою тріад, отриманий результат перевести в двійкову систему числення	Якщо студент народився у 17.07.2002 року, тоді число $\mathbf{1707{,}2002_{10} \rightarrow ?_8}$; отриманий результат перевести $?_8 \rightarrow ?_2$
12. Перевести поточну дату в 16-систему числення. Виконати перевірку отриманого 16-числа в десяткову систему числення	Якщо поточна дата 31.01.2022 року, тоді необхідно перевести число $\mathbf{31012022_{10} \rightarrow ?_{16}}$ та виконати перевірку отриманого результату $?_{16} \rightarrow ?_{10}$
13. Перевести число, сформоване за №групи та №студента в списку групи із десяткової системи числення у 16-систему числення, використовуючи метод ділення	Якщо студент навчається в групі ШІД-12 і має номер 17 в списку групи, то необхідно перевести число $\mathbf{1217_{16} \rightarrow ?_{10}}$
14. Перевести число, сформоване за №групи та №студента в списку групи із десяткової системи числення у двійкову. Отриманий результат перевести із двійкової системи у 16-систему числення	Якщо студент навчається в групі ШІД-12 і має номер 30 в списку групи, то необхідно перевести число $\mathbf{1220_{10} \rightarrow ?_2}$ отриманий результат перевести $?_2 \rightarrow ?_{16}$

Завдання 2

1. Дослідити функціональність основних логічних елементів, побудувати їх таблиці істинності, записати логічні вирази, перевірити функціональність в ***Logic.Ly*** відповідно до варіанту (***табл. 2***):
   - ✓ Кон'юктора;
   - ✓ Диз'юктора***;***
   - ✓ Інвертора***;***
   - ✓ Логічного елемента І-НІ, АБО-НІ, І-АБО-НІ, І-АБО***;***
2. За заданим логічним виразом, відповідно до варіанту (***табл. 3***), побудувати структурну схему, скласти її таблицю істинності, перевірити функціональність схеми в ***Logic.Ly***
3. За заданою структурною схемою, відповідно до варіанту (***табл. 4***), скласти таблицю істинності, записати логічний вираз, перевірити функціональність схеми в ***Logic.Ly***

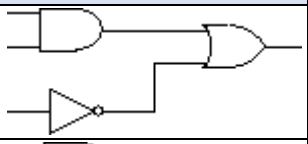
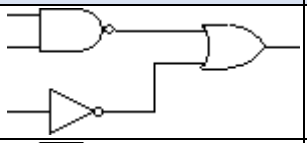
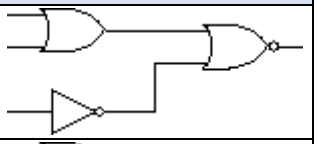
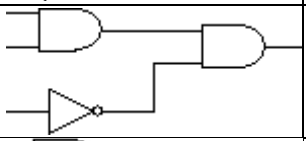
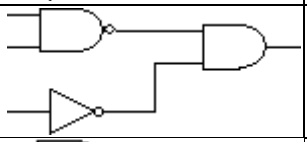
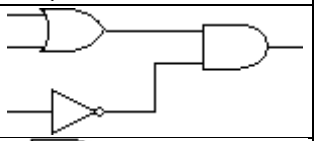
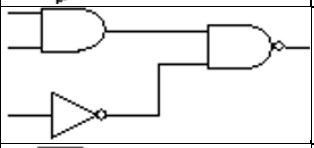
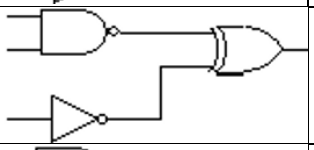
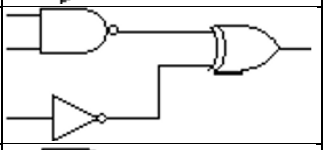
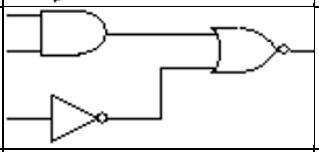
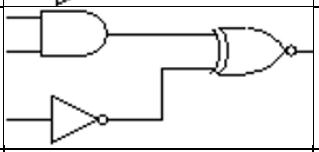
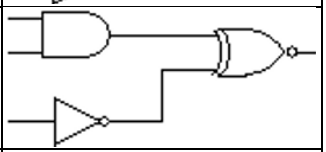
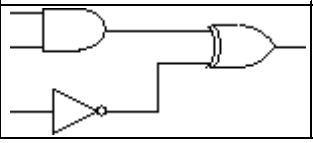
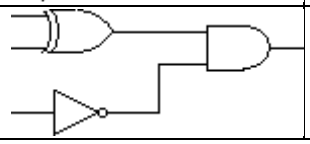
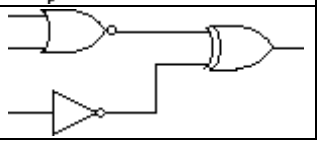
Таблиця 2. - Варіанти логічних елементів

№ вар.	Логічний елемент	Кількість входів	№ вар.	Логічний елемент	Кількість входів
1	кон'юктор	2	**6**	кон'юктор	3
	дизюнктор	3		дизюнктор	3
	інвертор	4		інвертор	3
	І-НІ	3		І-НІ-АБО-НІ	4
2	кон'юктор	3	**7**	кон'юктор	2
	дизюнктор	5		дизюнктор	2
	інвертор	3		інвертор	1
	АБО-НІ	2		АБО-І-НІ	4
3	кон'юктор	2	**8**	кон'юктор	3
	дизюнктор	2		дизюнктор	2
	інвертор	3		інвертор	1
	І-АБО	4		АБО-НІ-І	3
4	кон'юктор	3	**9**	кон'юктор	4
	дизюнктор	2		дизюнктор	4
	інвертор	2		інвертор	2
	І-АБО-НІ	3		АБО-НІ-АБО	5
5	кон'юктор	2	**10**	кон'юктор	2
	дизюнктор	4		дизюнктор	4
	інвертор	2		інвертор	2
	І-АБО-І	6		І-НІ-І	4

Таблиця 3. - Варіанти логічних виразів

№ варіанту	Логічний вираз
1	$F = \overline{\overline{A} \vee B \vee C}$
2	$F = \overline{\overline{A} \& \overline{B}}$
3	$F = \overline{\overline{A} \& A \& \overline{B} \& B \& \overline{C} \& C}$
4	$F = \overline{A \& B \vee C}$
5	$F = \overline{\overline{A} \& \overline{B} \vee \overline{C} \& \overline{D}}$
6	$F = \overline{A \& B \vee C \& D}$
7	$F = \overline{A \& B \vee \overline{C}}$
8	$F = \overline{\overline{A} \& A \& \overline{B} \& B}$
9	$F = \overline{\overline{A} \vee B \& \overline{C}}$
10	$F = \overline{\overline{A \& B} \vee \overline{C \& D}}$
11	$F = \overline{\overline{A} \& B}$
12	$F = \overline{\overline{A} \vee B \vee \overline{C}}$
13	$F = \overline{\overline{A} \& \overline{B} \& C \& D}$
14	$F = \overline{\overline{A} \vee \overline{B}}$
15	$F = \overline{\overline{A} \& \overline{B} \& \overline{C} \& \overline{D}}$

Таблиця 4. - Варіанти завдань принципових схем

Варіант	Схема	Варіант		Варіант	
1.		6.		11.	
2.		7.		12.	
3		8.		13.	
4.		9.		14.	
5.		10.		15.	

Зміст звіту

1. Титульна сторінка, тема та мета практичного завдання.
2. Опис виконаних завдань з наведеними рисунками, графіками, таблицями істинності, логічними виразами, структурними, логічними та електричними схемами, а також часовими діаграмами.
3. Висновки, що відображають результати виконання роботи та їх експериментальний аналіз.

Теоретичні відомості

Загальне правило переводу цілих чисел.

Для переводу цілого числа з однієї позиційної системи в іншу його треба послідовно поділити на основу ***q*** тієї системи, у яку воно переводиться, доти, доки не вийде частка, менше ***q.*** Число в новій системі запишеться у вигляді залишків ділення, починаючи з останнього. Остання частка надає більше значення числа. Розглянемо приклад на наведеним правилом.

Приклад 1. Перевести число **118** із десяткової системи числення у двійкову (рис. 1).

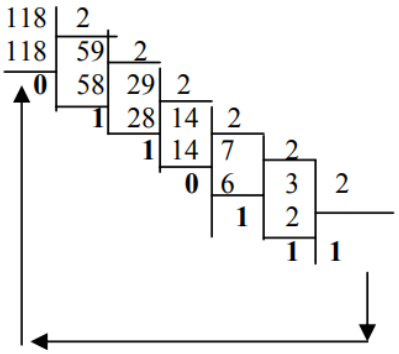


Рисунок 1. Переведення десяткового числа у двійкову систему числення.

В результаті отримаємо: **118_{10}=1110110_2**

Приклад 2. Десяткове число **83** перевести у двійкову систему числення (рис. 2) та записати результат.

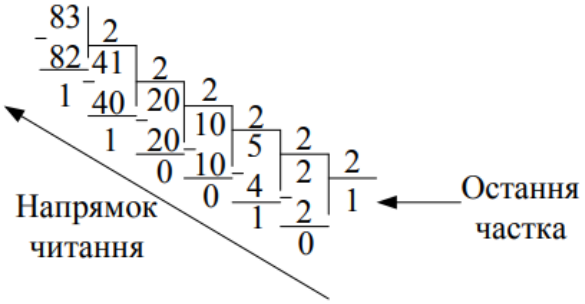


Рисунок 2. - Переведення десяткового числа у двійкову систему числення.

В результаті отримаємо: **83_{10}=1010011_2**

Приклад 3. Перевести число **372_8** із восьмирічної системи числення у десяткову.

Для переведення числа з однієї позиційної системи числення в іншу використовується наступна формула:

$$f = \sum k \cdot q^n$$

де,

k – розряди чисел в системі числення;

q – основа позиційної системи числення;

n – степеневе значення за основою в системі числення.

Використовуючи формулу переведення чисел (1) з однієї системи числення в іншу, можна записати наступний вираз:

$$372_8 = 3 \cdot (8^2) + 7 \cdot (8^1) + 2 \cdot (8^0) = 3 \cdot 64 + 7 \cdot 8 + 2 \cdot 1 = 192 + 56 + 2 = 250_{10}$$

В результаті отримаємо: **372_8=250_{10}**

Приклад 4. Переведемо число **135_{10}** у 8-систему числення (рис. 3).

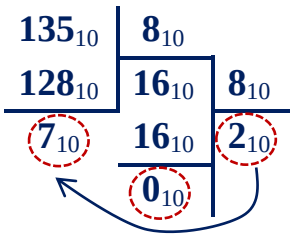


Рисунок 3. - Переведення числа із 10-системи числення у 8-систему.

В результаті отримаємо: **135_{10}=207_8**

Приклад 5. Переведемо число $24{,}6_8$ у десяткову систему числення.

Використовуючи формулу переведення чисел (1) з однієї системи числення в іншу, можна записати наступний вираз:

$$24{,}6_8 = 2 \cdot (8^1) + 4 \cdot (8^0) + 6 \cdot (8^{-1}) = 2 \cdot 8 + 4 \cdot 1 + 6 \cdot 0{,}8 = 16 + 4 + 0{,}75 = 20{,}75_{10}.$$

В результаті отримаємо: **$24{,}6_8$=$20{,}75_{10}$**

Загальне правило переведення дробових чисел.

Для переведення правильного дробу з однієї позиційної системи в іншу його потрібно послідовно множити на основу тієї системи, у яку він переводиться. Перемножуються тільки дробові частини; дріб у новій системі запишеться у вигляді цілих частин добутків.

Приклад 6. Десятковий дріб **0,3125** перевести у двійкову систему числення. Двійкове число може бути розкладене за ступенями 2 у будь-якій системі числення. Для переведення числа скористаємось схемою, яка відображена на рис. 4.

В результаті отримаємо: $\mathbf{0,3125_{10} = 0,0101_2}$.

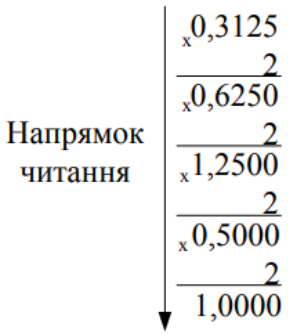


Рисунок 4. - Переведення дробового числа із 10–системи числення у двійкову.

Приклад 7. Переведення цілого число з двійкової системи числення в десяткову:

$$100111_2 = 1\cdot2^5 + 0\cdot2^4 + 0\cdot2^3 + 1\cdot2^2 + 1\cdot2^1 + 1\cdot2^0 = 32+4+2+1 = 39_{10}.$$

Переведення дробу з двійкової системи числення в іншу проводиться за допомогою послідовного множення дробу на основу його системи.

Приклад 8. Перевести дробове число з двійкової системи числення в десяткову:

$$1011,01101_2 = 1\cdot2^3 + 0\cdot2^2 + 1\cdot2^1 + 1\cdot2^0 + 0\cdot2^{-1} + 1\cdot2^{-2} + 1\cdot2^{-3} + 0\cdot2^{-4} + 1\cdot2^{-5}$$

$$= 8+2+1+\frac{1}{4}+\frac{1}{8}+\frac{1}{32} = 11\frac{13}{32} = 11,40625_{10}$$

Переведення двійкового числа в десятковий його еквівалент можна виконати за допомогою такого дуже простого способу. Перетворення здійснюється шляхом підсумовування значень ступенів числа 2, що відповідають тим розрядам двійкового числа, що переводиться, у яких містяться одиниці. Наприклад, необхідно перетворити двійкове число $\mathbf{10111010_2}$ у десяткове число, тоді складемо таблицю переведення значень розрядів (табл. 5)

Таблиця 5. - Переведення числа із двійкової системи числення в десяткове за його еквівалентом.

Розряд числа	7	6	5	4	3	2	1	0
Задане число	**1**	**0**	**1**	**1**	**1**	**0**	**1**	**0**
Ступінь числа	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Результат	128	0	32	16	8	0	2	0

В результаті отримаємо: $\mathbf{10111010_2 = 272_{10}}$.

Шістнадцятирічна система числення – це система з основою **16**, в якій для зображення чисел служать наступні символи:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Буквені позначення введено тому, що незручно зображувати одну цифру двома символами.

Приклад 9. Шістнадцятирічне число $IFA,C24_{16}$ перевести у двійкову систему числення.

Для переведення шістнадцятирічного числа скористаємось схемою відповідності чисел (тетради) 16-система/2-система, яка наведена на рис. 5.

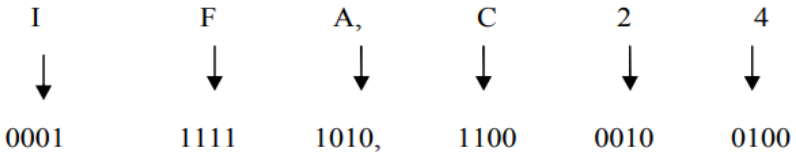


Рисунок 5. – Схема переведення відповідності чисел 16-система/2-система (тетрадами)

В результаті отримаємо: $\mathbf{1FA,C24_{16}=0001\ 1111\ 1010,1100\ 0010\ 0100_2}$.

Приклад 10. Задане число $763,224_8$, яке задане у 8-системі числення необхідно перевести у двійкову систему числення, використовуючи тріади.

Для переведення заданого числа скористаємось схемою відповідності чисел 8-система/2-система, яка наведена на рис. 6.

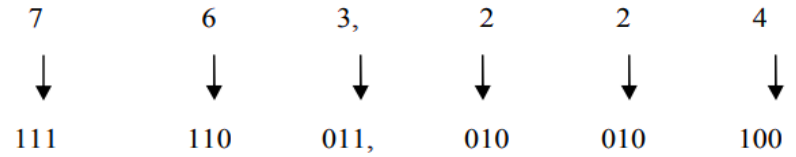


Рисунок 6. – Схема переведення відповідності чисел 8-система/2-система (тріадами)

В результаті отримаємо: $\mathbf{763,224_{16}=0001\ 1111\ 0011,1001\ 0100_2}$.

Для переведення чисел з двійкової системи числення у шістнадцятирічну (восьмирічну) необхідно двійкові числа, згрупувати по чотири (***тетради***) або (***тріади***) в обидві сторони від коми. За необхідності неповні групи цифр доповнити 0. Кожну з одержаних груп двійкових цифр замінити відповідною їй цифрою у новій системі числення, а одержані шістнадцятирічні (восьмирічні) цифри розташувати на місцях замінюваних двійкових кодів.

Наприклад, переведемо число $\mathbf{111111010,1100001_2}$ у шістнадцятирічне. Розбиваємо двійкове число на тетради та отримаємо результат переведення:

0001	**1111**	**1010,**	**1100**	**0010**	**0100**
1	**F**	**A**	**C**	**2**	**4**

Приклад 11. Перевести з шістнадцятирічної системи числення число **BA** у десяткову систему:

$$BA=B\cdot16^1+A\cdot16^0=11\cdot16+10\cdot1=186.$$

Переведення з десяткової системи в шістнадцятирічну проводиться за

загальним правилом переведення цілих чисел. В результаті отримаємо: **$BA_{16}=186_{10}$.**

Приклад 12. Перевести число 165 з десяткової системи числення в шістнадцятирічну.

Для переведення десяткового числа в шістнадцятирічну систему, необхідно поділити число 165 на основу 16 (рис. 7)

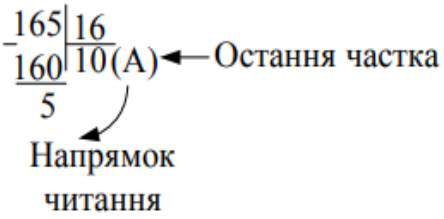


Рисунок 7. – Переведення числа із десяткової системи числення у шістнадцятирічну.

В результаті отримаємо: **$165_{10}=A5_{16}$**

Для того, щоб перевести з двійкової системи числення в шістнадцятирічну, достатньо подати це число у вигляді умовно поділеного на чотирьох розрядні двійкові коди (***тетради***) відповідно до таблиці 6.

Таблиця 6. – Відповідність значень в різних системах числення.

Десяткова система числення	*Двійкова система числення*	*Вісімкова система числення*	*Шіснадцяткова система числення*
0	**0000**	**0**	**0**
1	**0001**	**1**	**1**
2	**0010**	**2**	**2**
3	**0011**	**3**	**3**
4	**0100**	**4**	**4**
5	**0101**	**5**	**5**
6	**0110**	**6**	**6**
7	**0111**	**7**	**7**
8	**1000**	**10**	**8**
9	**1001**	**11**	**9**
10	**1010**	**12**	**A**
11	**1011**	**13**	**B**
12	**1100**	**14**	**C**
13	**1101**	**15**	**D**
14	**1110**	**16**	**E**
15	**1111**	**17**	**F**

Приклад 13. Перевести число 10110110001101_2 з двійкової системи числення в шістнадцятирічну.

Поділивши задане двійкове число на тетради, отримаємо наступний результат переведення:

$10\ 1101\ 1000\ 1101_2 = \underline{0010}\ \underline{1101}\ \underline{1000}\ \underline{1101} = 2D8D_{16}$.

Для полегшення переведення чисел з однієї позиційної системи числення в іншу можна скористатись таблицею 7. Додаткові приклади переведення із однієї позиційної системи числення в іншу наведено в табл. 8.

Таблиця 7. - Переведення десяткових чисел у шістнадцятирічні числа

Множник	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	1	2	3	4	5	6	7	8	9
10	A	14	1E	28	32	3C	46	50	5A
10^2	64	C8	12C	190	1F4	258	2BC	320	384
10^3	3E8	7D0	BB8	FA0	1388	1770	1B58	1F40	2328
10^4	2710	4E20	7530	9C40	C350	EA60	11170	13880	15F90
10^5	186A0	30D40	493E0	61A80	7A120	927C0	AAE60	C3500	DBBA0
Приклад: $1234=1000+200+30+4=(3E8)_{16}+(C8)_{16}+(1E)_{16}+(4)_{16}=(4D2)_{16}$									

Таблиця 8. - Приклади перетворення значень з різних системи числення

1	**10→2** $37_{10}=100101_2$ 37 | 2 36 | 18 | 2 1 | 18 | 9 | 2 0 | 8 | 4 | 2 1 | 4 | 2 | 2 0 | 2 | 1 0	5	**2→10** 5 4 3 2 1 0 $100101_2=1\cdot2^5+0\cdot2^4+0\cdot2^3+1\cdot2^2+$ $+0\cdot2^1+1\cdot2^0=37$
		6	**2→16** $100101_2=25_{16}$ 2 5
2	**10→8** 37 | 8 32 | 4 5 $37_{10}=45_8$	7	**8→2** $41_8=100101_2$ 100 101
		8	**16→2** $25_{16}=00100101_2$ 0010 0101
3	**10→16** 37 | 16 32 | 2 5 $37_{10}=25_{16}$	9	**8→10** 1 0 $45_8=4\cdot8^1+5\cdot8^0=37_{10}$
		10	**16→10** 1 0 $25_{16}=2\cdot16^1+5\cdot16^0=37_{10}$
4	**2→8** $100101_2=45_8$ 4 5	11	**16→8** $25_{16}=00100101_2$ $25_{16}=00100101_2=45_8$ 0010 0101 0 4 5
		12	**8→16** $41_8=100101_2=25_{16}$ 100 101 2 5

Основні логічні елементи

Теоретичною основою цифрової техніки є алгебра логіки. Основним предметом булевої алгебри є просте твердження, яке або істинне (позначають символом **1**) або хибне (позначають символом **0**).

Прості висловлювання позначають буквами, які у цифровій техніці називають змінними (***аргументами***). Змінну зі скінченним числом значень називають перемикальною, аз двома значеннями – булевою.

Функція, яка має скінченне число значень називається перемикальною (***логічною***). Логічна функція, число можливих значень якої дорівнює двом, називається булевою. Булеві функції двох змінних подано в табл. 9.

Таблиця 9.- Двозначні логічні функції

Функція	*Набір аргументів x_1, x_2*				*Назва логічної функції*	*Визначення логічної функції*
	0 0	0 1	1 0	1 1		
f_1	0	0	0	0	Константа нуль	0
f_2	0	0	0	1	Кон'юнкція	$X_1 X_2$
f_3	0	0	1	0	Заборона X_2	$X_1 \overline{X}_2$
f_4	0	0	1	1	Повторення X_1	X_1
f_5	0	1	0	0	Заборона X_1	$\overline{X}_1 X_2$
f_6	0	1	0	1	Повторення X_2	X_2
f_7	0	1	1	0	Виняткове АБО	$X_1 \oplus X_2 = X_1 \overline{X}_2 \vee \overline{X}_1 X_2$
f_8	0	1	1	1	Диз'юнкція	$X_1 \vee X_2$
f_9	1	0	0	0	Стрілка Пірса	$\overline{X_1 \vee X_2}$
f_{10}	1	0	0	1	Рівнозначність	$X_1 \sim X2 = X_1 X_2 \vee \overline{X}_1 \overline{X}_2$
f_{11}	1	0	1	0	Інверсія X_2	$\overline{X}_2$
f_{12}	1	0	1	1	Імплікація від X_2 до X_1	$X_1 \equiv \overline{X}_2$
f_{13}	1	1	0	0	Інверсія X_1	$\overline{X}_1$
f_{14}	1	1	0	1	Імплікація від X_1 до X_2	$\overline{X}_1 \equiv X_2$
f_{15}	1	1	1	0	Штрих Шеффера	$\overline{X_1 X_2}$
f_{16}	1	1	1	1	Константа одиниця	1

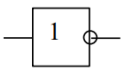
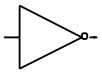
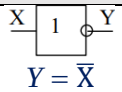
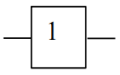
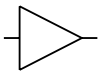
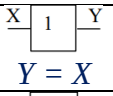
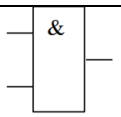
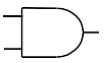
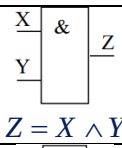
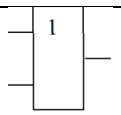
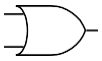
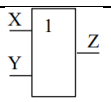
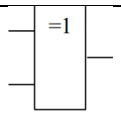
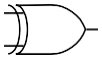
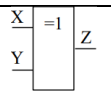
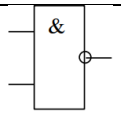
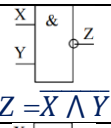
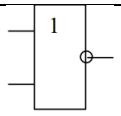
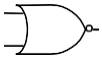
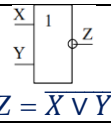
Фундаментальними логічними схемами є логічні елементи, з яких складаються складні схеми і цифрові системи. Назви та позначення логічних елементів наведено в табл. 10

Таблиця істинності – це спосіб подання залежності вихідних сигналів логічної схеми від логічних рівнів її вхідних сигналів. Таблиці істинності двозначних логічних функцій наведені у табл. 10.

Дослідження основних логічних операцій

Розглянемо принципи роботи основних логічних елементів та перевіримо їх функціональність в онлайн-додатку ***Logic.Ly*** https://logic.ly/

Таблиця 10. – Схеми логічних елементів

Назва логічного елемента	Позначення		Логічна функція	Назва в розділі Standard Gates
	ГОСТ	ANSI		
НІ (інвертор)				Inverters
буферний елемент				Buffers
І кон'юктор				And Gates
АБО диз'юнктор				Or Gates
Виключене АБО				Xor Gates
І-НІ				Nand Gates
АБО-НІ				Nor Gates

Приклад 14. Дослідити функціональність кон'юктора (рис. 8): побудувати таблицю істинності, записати логічний вираз, перевірити функціональність елемента в ***Logic.Ly.***

Розв'язок. Для кон'юктора побудуємо таблицю істинності, яка має два входи A, B та вихід F (табл. 11) Логічний вираз реалізованого кон'юктора має наступний вигляд: **F=A ∧ B.** Функціональність кон'юктора наведено на рис. 8 при активних сигналах x_1 та x_2.

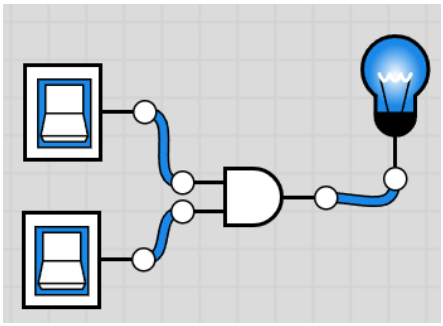


Рисунок 8. – Функціональна схема кон'юктора

Таблиця 11. – Таблиця істинності кон'юктора

A	*B*	*F*
0	0	0
0	1	0
1	0	0
1	1	1

Приклад 15. Дослідити функціональність диз'юктора (рис. 9): побудувати таблицю істинності, записати логічний вираз, перевірити функціональність елемента в ***Logic.Ly.***

Розв'язок. Для диз'юктора побудуємо таблицю істинності, яка має два входи A, B та вихід F (табл. 12). Логічний вираз реалізованого диз'юктора має вигляд: **F=A v B.** Функціональність диз'юктора наведено на рис. 9 при активному сигналі x_1.

Таблиця 12. – Таблиця істинності диз'юктора

A	B	F
0	0	1
0	1	1
1	0	1
1	1	1

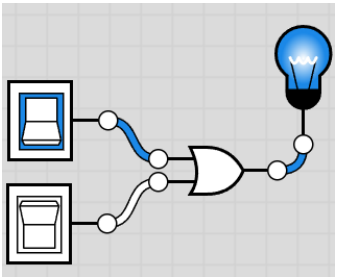


Рисунок 9. – Функціональна схема диз'юктора

Приклад 16. Дослідити функціональність інвертора (рис. 10): побудувати таблицю істинності, записати логічний вираз, перевірити функціональність елемента в ***Logic.Ly.***

Розв'язок. Для інвертора побудуємо таблицю істинності, яка має один вхід A та вихід F (табл. 13). Логічний вираз реалізованого інвертора має вигляд: **F=$\overline{A}$**

Таблиця 13. – Таблиця істинності інвертора

A	F
0	1
1	0

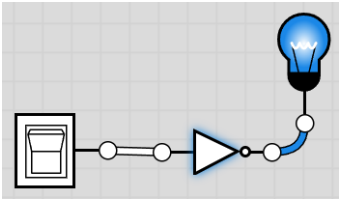


Рисунок 10. – Функціональна схема інвертора

Приклад 17. Дослідити функціональність логічного елемента І-НІ (рис. 11): побудувати таблицю істинності, записати логічний вираз, перевірити функціональність елемента в ***Logic.Ly.***

Розв'язок. Для логічного елемента І-НІ побудуємо таблицю істинності, яка має два входи A, A та вихід F (табл. 14). Логічний вираз реалізованого елемента І-НІ має вигляд: **F=$\overline{A}$**

Таблиця 14. – Таблиця істинності логічного елемента І-НІ

A	A&A	F=$\overline{A\&A}$
0	0	1
1	1	0

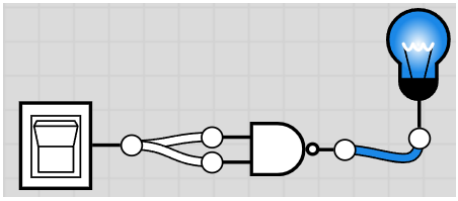


Рисунок 11. – Функціональна схема логічного елемента **І-НІ**

Приклад 18. Дослідити функціональність логічної схеми (рис. 12): побудувати таблицю істинності, записати логічний вираз, перевірити функціональність схеми в ***Logic.Ly.***

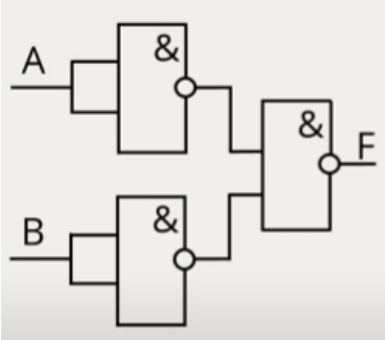


Рисунок 12. – Структурна схема логічних елементів

Розв'язок. Для логічної схеми побудуємо таблицю істинності, яка має два входи A, B та вихід F (***табл. 15***). Логічний вираз реалізованої схеми має наступний вигляд: $\mathbf{F=\overline{A}\&\overline{B}}$ або $\mathbf{F=\overline{A\&A}\&\overline{B\&B}}$. Функціональну схему визначеної функції наведено на рис. 13

Таблиця 15. – Таблиця істинності логічної схеми

A	B	A&A	B&B	$\overline{A}$	$\overline{B}$	$\overline{A}\&\overline{B}$	$F=\overline{A\&B}$
0	0	0	0	1	1	1	0
0	1	0	1	1	0	0	1
1	0	1	0	0	1	0	1
1	1	1	1	0	0	0	1

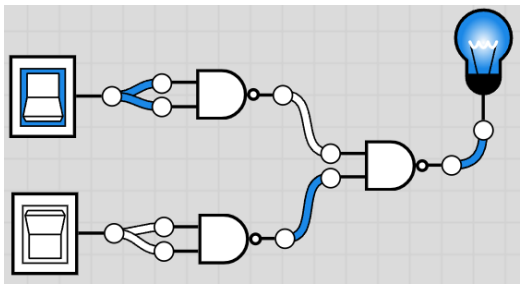


Рисунок 13. – Функціональна схема заданої функції в ***Logic.Ly***

Приклад 19. Дослідити функціональність логічного елемента АБО-НІ (рис. 13): записати логічний вираз, побудувати таблицю істинності, перевірити функціональність елемента в ***Logic.Ly.***

Розв'язок. Для логічного елемента АБО-НІ, який має два входи A, B та вихід F складемо логічний вираз: $\mathbf{F=\overline{A \vee B}}$ За складеним виразом побудуємо таблицю істинності (табл. 16). Логічний вираз реалізованого елемента АБО-НІ має вигляд, який наведено на рис. 14.

Таблиця 16. – Таблиця істинності логічного елемента АБО-НІ

A	B	A∨B	$F=\overline{A \vee B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

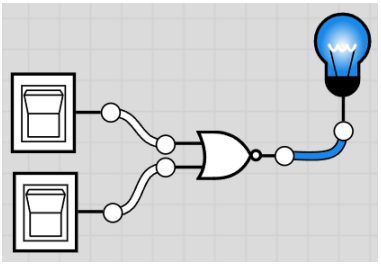


Рисунок 14. – Структурна схема логічного елемента АБО-НІ

Приклад 20. За заданим логічним виразом $F=\overline{A \,\&\, B \vee C}$ побудувати структурну схему та скласти таблицю істинності, перевірити функціональність схеми в ***Logic.Ly***

Розв'язок. Побудуємо структурну схему, яка має три входи A, B, C та вихід F (рис. 15). За заданим виразом побудуємо таблицю істинності (табл. 17).

Таблиця 17. – Таблиця істинності побудованою логічної схеми

A	B	C	A&B	A & B ∨ C	$F=\overline{A \,\&\, B \vee C}$
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	0	1	0
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	1	1	0
1	1	1	1	1	0

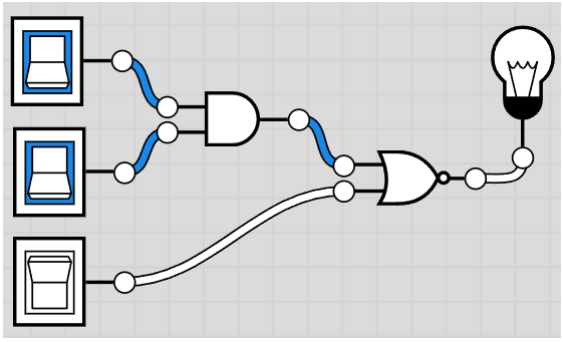


Рисунок 15. – Структурна схема побудованою логічної схеми

Контрольні запитання

1. Привести загальне правило переведення цілих чисел.
2. Як перевести число з двійкової у будь-яку систему числення?
3. Що являє собою шістнадцятирічна система числення?
4. Як проводиться переведення числа із шістнадцятирічної позиційної системи у іншу систему числення?
5. З якою метою у запису чисел використовуються тетради?
6. Привести загальне правило переведення дробових значень.
7. Яке значення для сприйняття числа має напрямок читання його розрядів при виконанні переведення з іншої системи числення?
8. Дати означення логічних функцій.
9. Зобразити умовні позначення логічних елементів, таблиці істинності логічних функцій.

Практичне завдання 2.
Побудова основних логічних елементів за допомогою дискретних компонентів та їх представлення на електричних принципових схемах

Мета роботи: побудова електричних схем основних логічних елементів І, АБО, НІ за допомогою дискретних компонентів в ***EasyEDA***; набуття практичних навичок по створенню макетів печатних плат реалізованих логічних елементів в ***TinkerCad***.

Задання 1. Створити електричні-принципові схеми логічних елементів за допомогою простих дискретних компонентів в ***EasyEDA***:

- ✓ Логічного елемента ***І*** (***AND***) на діодах;
- ✓ Логічного елемента ***АБО*** (***OR***) на діодах;
- ✓ Логічного елемента ***І*** (***AND***) на транзисторах;
- ✓ Логічного елемента ***АБО*** (***OR***) на транзисторах;
- ✓ Логічного елемента ***НІ*** (***NOT***) на транзисторах;
- ✓ Логічного елемента ***І-НІ*** (***NAND***) на транзисторах;
- ✓ Логічного елемента ***АБО-НІ*** (***NOR***) на транзисторах.

Побудувати печатні плати створених логічних елементів та зберегти їх макети в ***2D*** та ***3D*** в онлайн конструкторі ***EasyEDA***.

Побудувати схеми логічних елементів на макетних платах в ***TinkerCad*** та перевірити принцип їх функціонування.

Завдання 2. Використовуючи дискретні компоненти, створити в **EasyEDA** електричні схеми логічних елементів відповідно до заданого варіанту (табл. 18).

Таблиця 18. - Варіанти для створення логічних елементів

№ вар.	*Логічний елемент*	*Кількість входів*	*№ вар.*	*Логічний елемент*	*Кількість входів*
1	кон'юктор І	2	**5**	кон'юктор І	4
	диз'юктор АБО	4		диз'юктор АБО	4
	інвертор НІ	2		інвертор НІ	3
	елемент І-НІ	4		елемент І-НІ	2
	елемент АБО-НІ	2		елемент АБО-НІ	2
2	кон'юктор І	4	**6**	кон'юктор І	2
	диз'юктор АБО	2		диз'юктор АБО	2
	інвертор НІ	3		інвертор НІ	1
	елемент І-НІ	4		елемент І-НІ	4
	елемент АБО-НІ	2		елемент АБО-НІ	6
3	кон'юктор І	2	**7**	кон'юктор І	4
	диз'юктор АБО	2		диз'юктор АБО	2
	інвертор НІ	1		інвертор НІ	1
	елемент І-НІ	2		елемент І-НІ	4
	елемент АБО-НІ	4		елемент АБО-НІ	2
4	кон'юктор І	6	**8**	кон'юктор І	4
	диз'юктор АБО	4		диз'юктор АБО	4
	інвертор НІ	2		інвертор НІ	2
	елемент І-НІ	4		елемент І-НІ	6
	елемент АБО-НІ	2		елемент АБО-НІ	6

Побудувати печатні плати створених логічних елементів та зберегти їх макети в ***2D*** та ***3D*** в онлайн конструкторі ***EasyEDA***.

Побудувати схеми логічних елементів на макетних платах в ***TinkerCad*** та перевірити принцип їх функціонування.

Зміст звіту

1. Титульна сторінка, тема та мета практичного завдання.
2. Опис виконаних завдань з наведеними рисунками, графіками, таблицями істинності, логічними виразами, структурними, логічними та електричними схемами створені в онлайн конструкторі ***EasyEDA***.
3. Печатні плати створених логічних елементів та зберегти їх макети в 2D та 3D в онлайн конструкторі ***EasyEDA***.
4. Схеми логічних елементів на макетних платах в ***TinkerCad.***
5. Висновки, що відображають результати виконання роботи та їх експериментальний аналіз.

Теоретичні відомості

На прикладі простої електричної схеми зберемо на макетній платі схему в ***TinkerCad*** для перевірки роботи цифрових та аналогових сигналів.

Приклад 1. Створити електрично-принципову схему в ***EasyEDA*** з використанням цифрових приладів, яка наведена на рис. 16 та містить наступні дискретні елементи: кнопку, резистор ємністю 330 Ом, світлодіод.

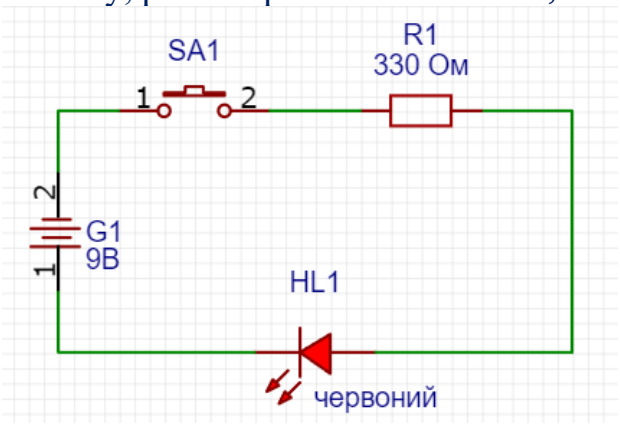


Рисунок - 16. Електрична принципова схема цифрового приладу.

Зберемо наведену електричну схему на макетній платі (рис. 17) в ***TinkerCad*** та перевіримо її функціональність.

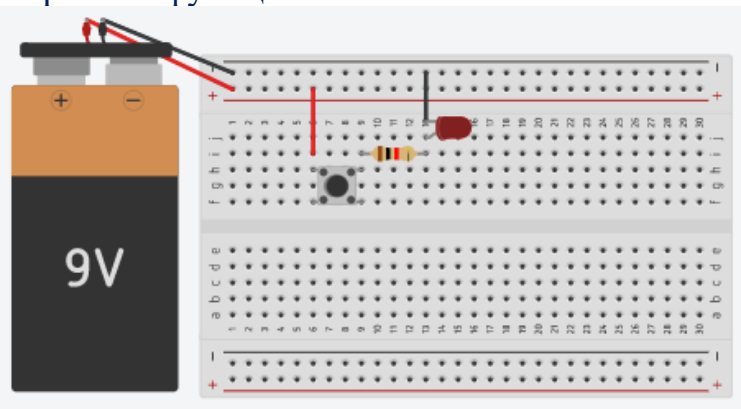


Рисунок - 17. - Схема цифрового приладу на макетній платі

В даній схемі світлодіод через обмежуючий резистор та кнопку підключений до джерела живлення. Коли кнопка натиснута, на світлодіоді присутня напруга – сигнал високого рівня. Такий сигнал на виході схеми виводить логічну одиницю. Після відпускання кнопки світлодіод гасне, що відповідає присутності сигналу низького рівня – логічному нулю.

Логічні елементи можна створити за допомогою простих дискретних компонентів на: діодах, транзисторах, резисторах. Розглянемо приклади зборки схем основних логічних елементів на печатних та макетних платах.

Приклад 2. В ***EasyEDA*** створити електрично-принципову схему логічного елементу «І» (AND) на діодах (рис. 18), яка містить наступні елементи: 2 резистори ємністю 10 кОм; резистор ємністю 75 кОм; 2 випрямних діоди; світлодіод; DIP-перемикач.

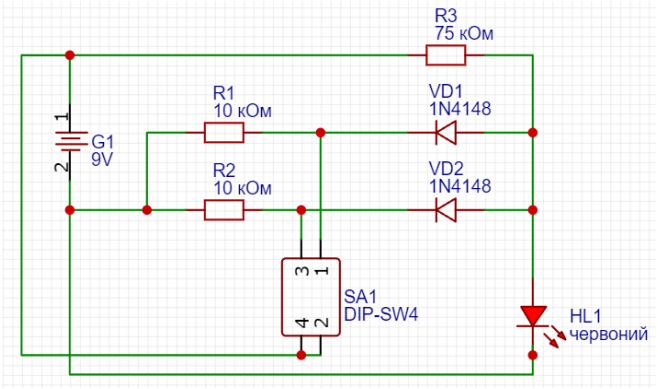


Рисунок 18. - Схема електрична принципова логічного елементу І на діодах

Після проектування електричної схеми в ***EasyEDA*** створимо макет печатної плати (рис. 19) та перевіримо її у форматі **2D** (рис. 20), **3D** (рис. 21). Виконаємо зборку схеми на макетній платі в ***TinkerCad*** відповідно до рис. 22.

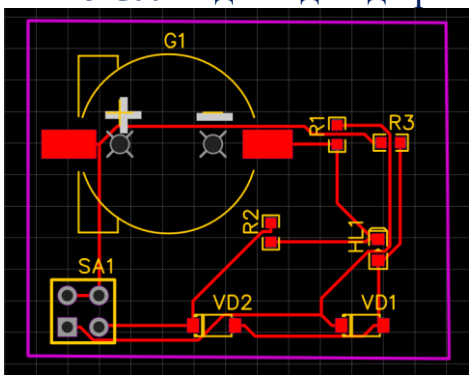


Рисунок 19. – Логічний елемент І на печатній платі

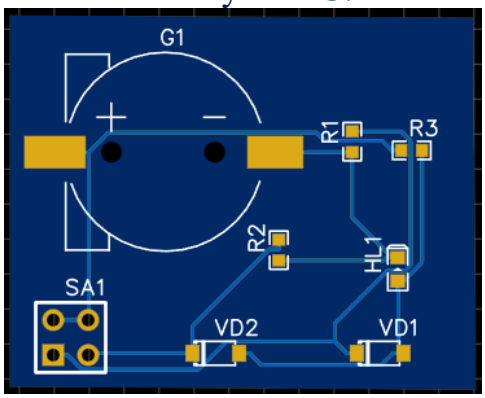


Рисунок 20. - Печатні плата в 2D

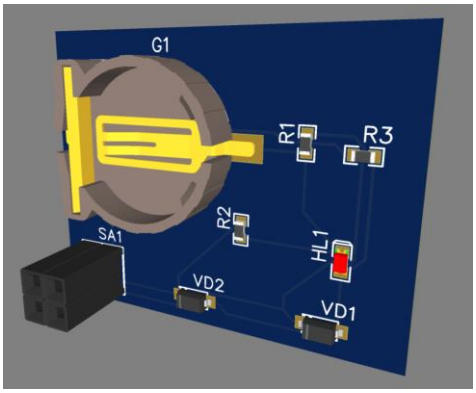


Рисунок 21. - Печатна платі в 3D

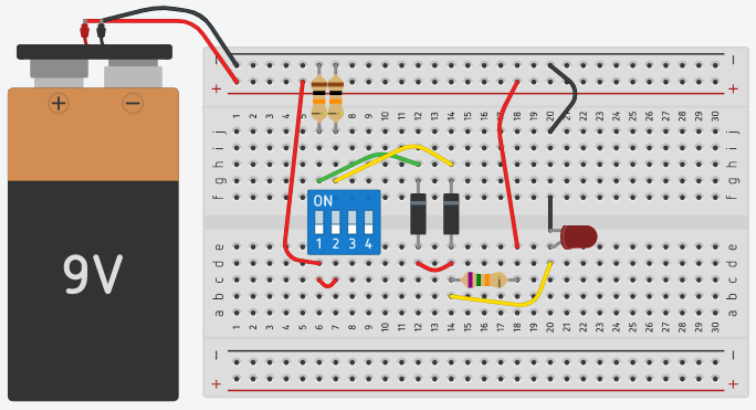


Рисунок 22.- Схема елементу І на діодах зібрана на макетній платі.

В даній схемі застосовуються тумблери, оскільки, на відміну від кнопок, ними зручно фіксувати необхідне значення вхідного сигналу. Якщо переводити тумблери №1 та №2, які підключені до входів А та В. почергово включаючи тумблери можна перевірити працездатність за таблицею істинності логічного елемента І на схемі.

Розглянемо принцип роботи даної схеми. Резистори R1 та R2 необхідні для забезпечення на входах елементу стабільного логічного нуля, коли тумблери вимкнені. Їх опори достатньо великі, що забезпечує подачу сигналу високого рівня та упереджує короткого замикання джерела живлення.

Якщо встановити обидва тумблери в положення OFF, то світлодіод не світиться так як катоди діодів VD1 та VD2 через резистори R1 та R2 під'єднані до мінусу джерела живлення. За такого підключення діоди відкриваються і крізь них поступає електричний струм. Оскільки опір діодів набагато менший від опору світлодіоду, то весь струм передано діодам, а світлодіод не буде світитись. Падіння напруги на діодах дорівнює 0,7В, що відповідає напрузі логічного «0». Логічна «1» з'явиться в тому випадку, коли напруга буде подана на обидва діоди від «+» джерела живлення. Це буде тоді, коли обидва тумблери будуть в положенні ON.

Приклад 3. Створити електрично-принципову схему логічного елементу «І» (AND) на транзисторах, яка наведена на рис. 23 та містить наступні елементи: 2 резистори ємністю 20 кОм; 2 резистори ємністю 10 кОм; резистор ємністю 4,7 кОм; резистор ємністю 330 Ом; 2 біполярних NPN транзистори; світлодіод; DIP-перемикач.

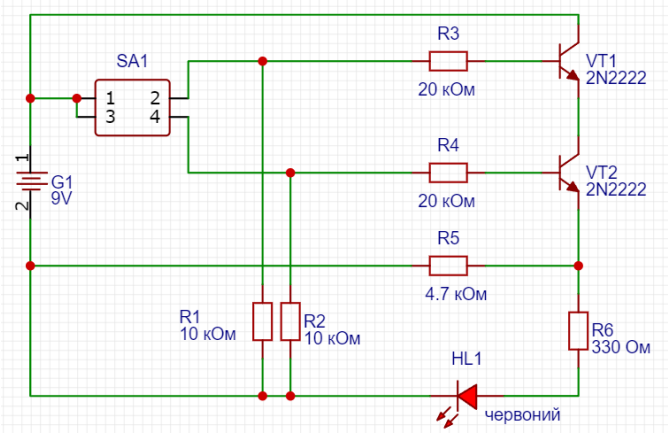


Рисунок 23.- Схема логічного елементу І на транзисторах.

Зібрати схему логічного елемента **І** в ***TinkerCad*** з використання транзисторів, можна макетній платі відповідно до рис. 24

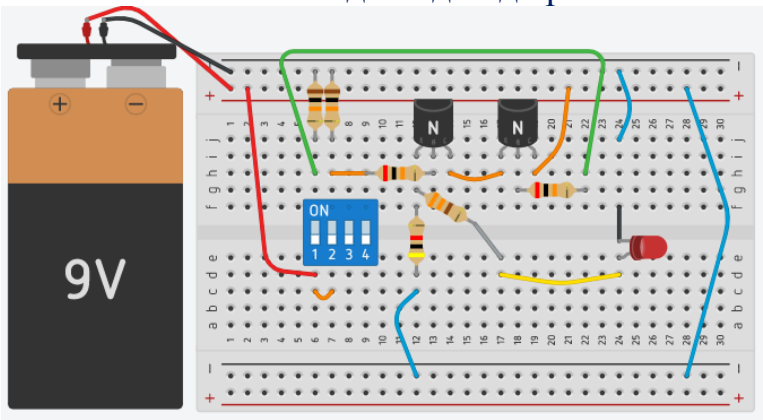


Рисунок 24.- Схема логічного елементу І на транзисторах на макетній платі.

На виході такого елементу буде логічна «1» лише тоді, коли на обох входах А та В буде присутній сигнал високого рівня. Тобто, коли транзистори пропускають струм, то світлодіод світитися. Якщо на вході елементу присутній хоча б один логічний «0», то відповідний транзистор не пропускає струм, що призведе до розриву ланцюга і світлодіод не буде світитись.

Приклад 4. Створити електрично-принципову схему логічного елементу «АБО» (OR) на випрямних діодах, яка наведена на рис. 25 та містить наступні елементи: 2 резистори ємністю 10 кОм; резистор ємністю 75 кОм; 2 випрямних діоди; резистор ємністю 330 Ом; світлодіод; DIP-перемикач.

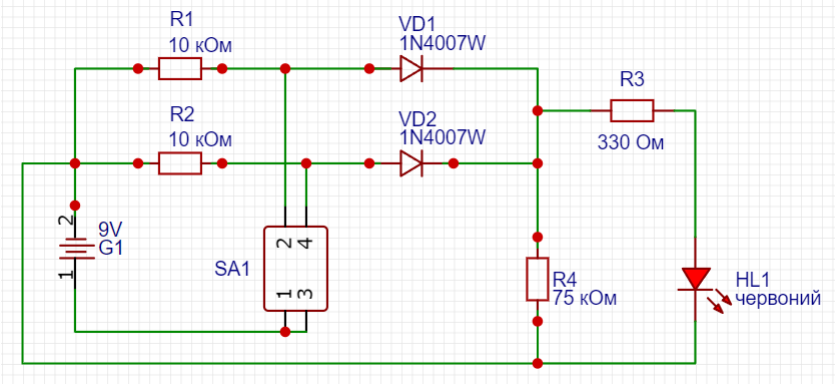


Рисунок 25. - Схема логічного елементу АБО на діодах

Зібрати схему в ***TinkerCad*** на макетній платі можна відповідно схеми, наведеної на рис. 26

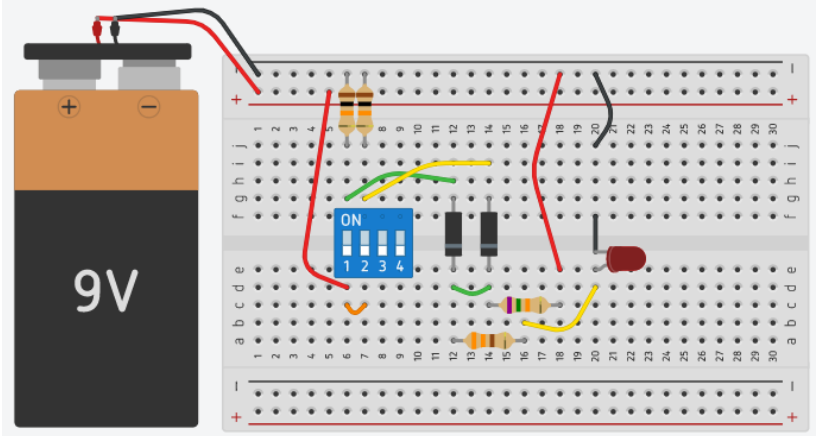


Рисунок 26. - Схема елементу АБО на діодах зібрана на макетній платі.

В даній схемі, якщо на обидва входи подавати логічного «0», то світлодіод не буде загоратись, так як діоди VD1 та VD2 закриті, оскільки на їх анодах присутній низький рівень сигналу, що унеможливлює протікання електричного струму. Якщо включити хоча б один тумблер, то буде подано високий рівень сигналу і анод відповідного діода підключається до «+» джерела живлення та пропускає струм у схемі. На вихід схеми передається логічна «1». Тобто, для появи на виході логічної «1» достатньо активувати хоча б один із діодів.

Приклад 5. Створити електрично-принципову схему логічного елементу «АБО» (OR) на транзисторах, яка наведена на рис. 27 та містить наступні елементи: 2 резистори ємністю 20 кОм; 2 резистори ємністю 10 кОм; резистор ємністю 4,7 кОм; резистор ємністю 330 Ом; 2 біполярних NPN транзистори; світлодіод; DIP-перемикач.

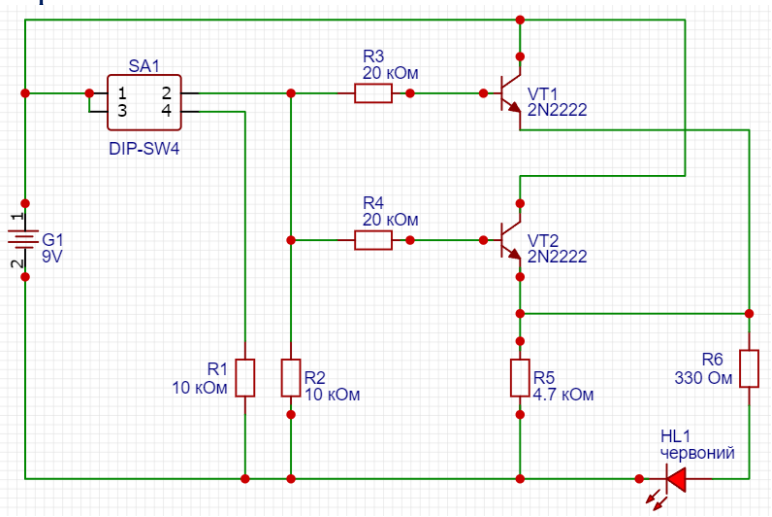


Рисунок 27. - Схема логічного елементу АБО на транзисторах

Зібрати схему в ***TinkerCad*** на макетній платі можна відповідно до наведеної на рис. 28 схеми. Дану схему можна представити у вигляді двох паралельно ввімкнених перемикачів, які в будь якому разі пропускають сигнал на виходи схеми, що відповідає наявності логічної «1».

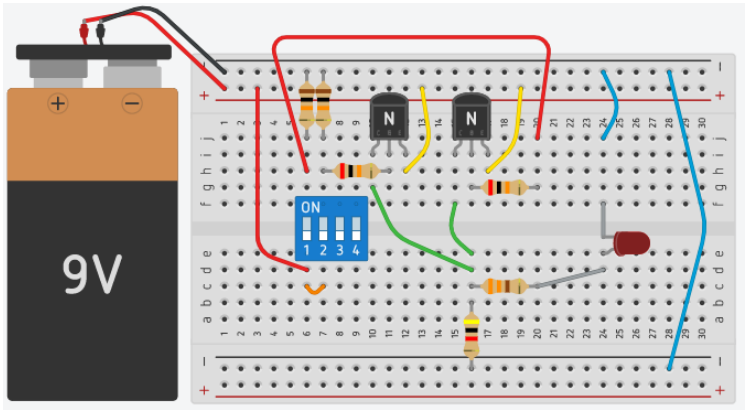


Рисунок 28. - Схема елементу АБО на транзисторах.

За наявності логічної «1» хоча б на одному із входів, відповідний транзистор відкривається та пропускає струм на виході схеми. Отже, коли присутня логічна «1» на одному із входів, то розриву ланцюга немає. Розрив буде лише тоді, коли на обох входах буде логічний «0». Функціональність роботи елемента можна перевірити у відповідності до таблиці істинності (табл. 12)

Приклад 6. Створити електрично-принципову схему логічного елементу «HI» (NOT) на біполярному транзисторі, яка наведена на рис. 29 та містить наступні елементи: 2 резистори ємністю 10 кОм; резистор ємністю 4,7 кОм; біполярний NPN транзистор; світлодіод; DIP-перемикач.

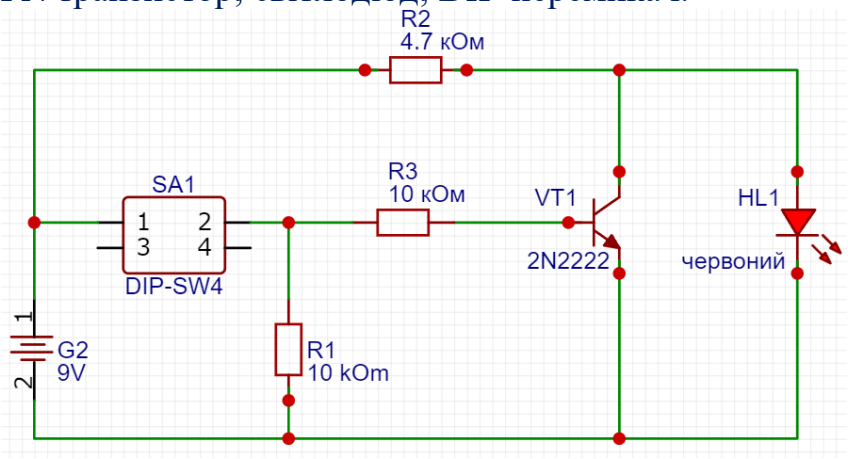


Рисунок 29. - Схема електрична принципова логічного елементу HI.

В ***TinkerCad*** зібрати схему логічного елементу HI на макетній платі можна відповідно до наведеної на рис. 30 структури. На даній схемі роль елемента HI виконує біполярний NPN-транзистор VT1, ввімкнений у схему із загальним емітером. Виходом даного елементу є колектор транзистора.

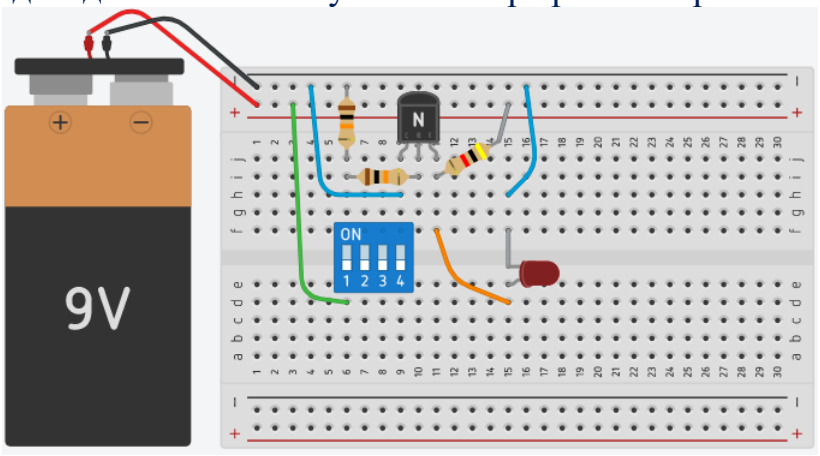


Рисунок 30. - Схема логічного елементу НЕ створена на макетній платі.

За наявності високого рівня сигналу на вході A, на виході Y буде логічний «0», тому світлодіод не загориться. Це відбудеться тому, що в даному випадку відкриється транзистор і струм протікає між колектором та емітером за рахунок великого опору на резисторі R2, оминаючи світлодіод.

За наявності сигналу низького рівня на вході A, на виході Y буде логічна «1», то світлодіод засвітиться. Струм від «+» джерела живлення до «-» протікає через резистор R2 та світлодіод HL1, оминаючи транзистор, так як він буде у закритому стані, на його базу подано низький рівень сигналу.

Приклад 7. Створити електрично-принципову схему логічного елементу «I-HI» (NAND) на транзисторах, яка наведена на рис. 31 та містить наступні елементи: 2 резистори ємністю 20 кОм; 2 резистори ємністю 10 кОм; резистор ємністю 4,7 кОм; резистор ємністю 330 Ом; 2 біполярних NPN транзистори; світлодіод; DIP-перемикач.

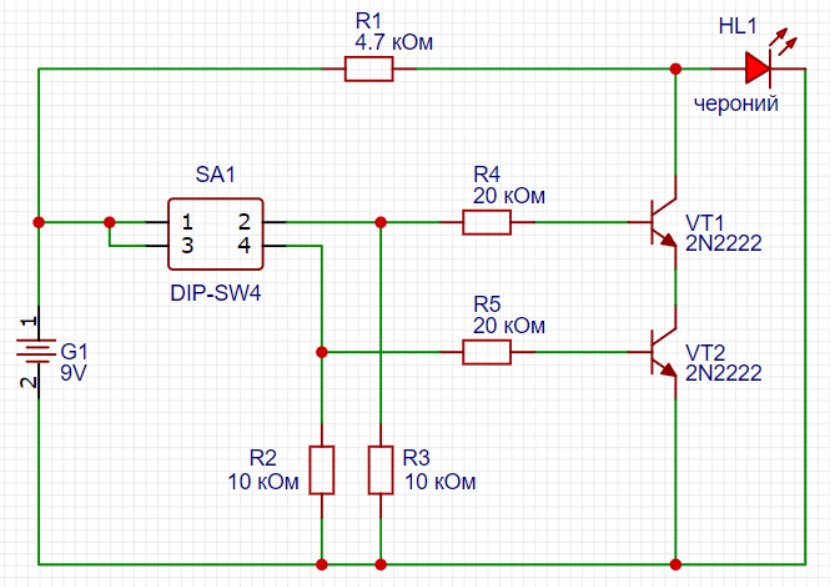


Рисунок 31. – Електрична схема логічного елементу І-НІ

Зібрати схему логічного елемента І-НІ в ***TinkerCad*** на макетній платі можна відповідно до структури, наведеної на рис. 32. Схема логічного елемента І-НІ схожа до роботи схеми логічного елемента І.

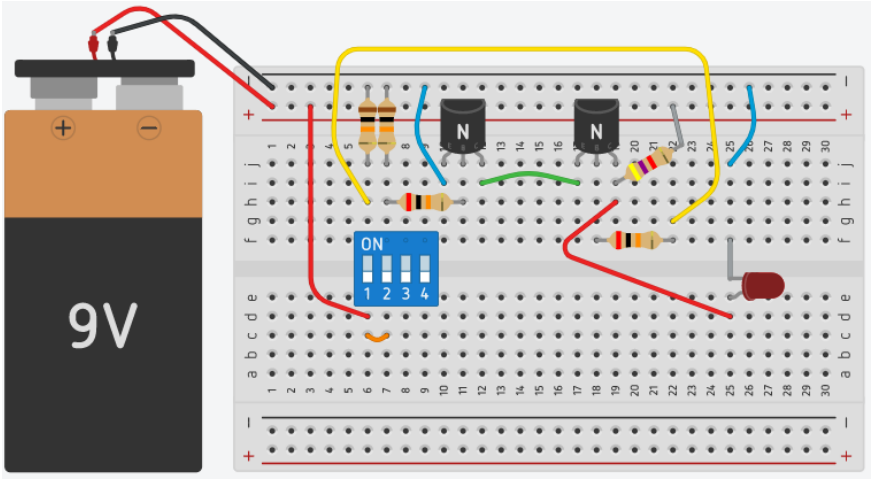


Рисунок 32. - Схема логічного елементу І-НІ створена на макетній платі.

На виході даної схеми з’явиться логічний «0» лише за присутності двох логічних «1» на входах А та В. При цьому транзистори VT1 та VT2 будуть відкриті та пропустять через себе струм, з’єднуючи вихід елементу з «-» джерела живлення. Якщо на будь-який вхід елементу буде подано сигнал логічного «0», то відповідний транзистор буде закритим. Через цей вихід потрібно підєднати до «+» джерела живлення, тобто буде сформовано сигнал логічної «1» і світлодіод загориться в схемі. Дослідити функціонування зібраної схеми можна у відповідності до його таблиці істинності.

Приклад 8. Створити електрично-принципову схему в ***EasyEDA*** логічного елементу «АБО-НІ» (NOR) на транзисторах, яка наведена на рис. 33 та містить наступні елементи: 2 резистори ємністю 20 кОм; 2 резистори ємністю 10 кОм; резистор ємністю 4,7 кОм; резистор ємністю 330 Ом; 2 біполярних NPN транзистори; світлодіод; DIP-перемикач.

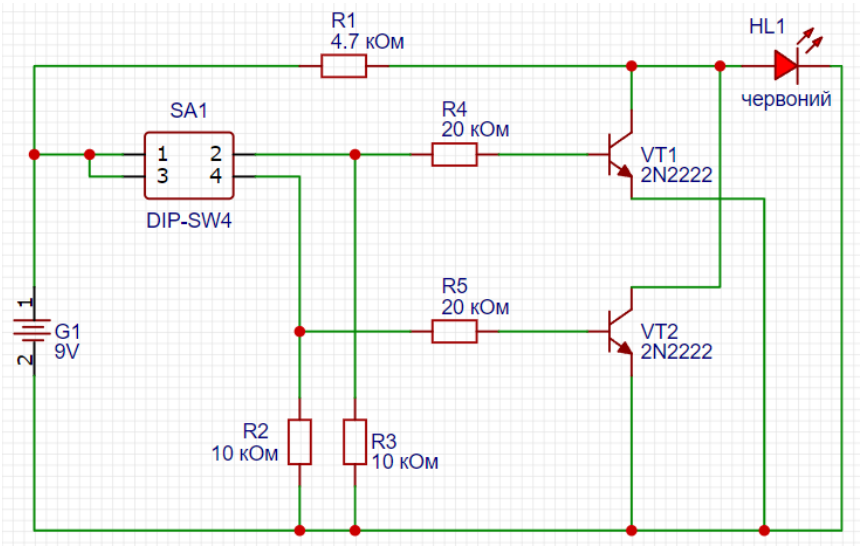


Рисунок 33. - Схема електрична логічного елементу АБО-НІ

Після проектування електричної схеми в ***EasyEDA*** створимо макет печатної плати (рис. 34) та переглянемо її у форматі **2D** з обох сторін плати (рис. 35), **3D** з обох сторін (рис. 36 а, б). Як видно з наведених рисунків, на платі печатні треки подані з обох сторін: червоним кольором показано припаяні треки на верхній стороні плати (рис. 35а); синім - на нижній стороні плати (рис. 35б).

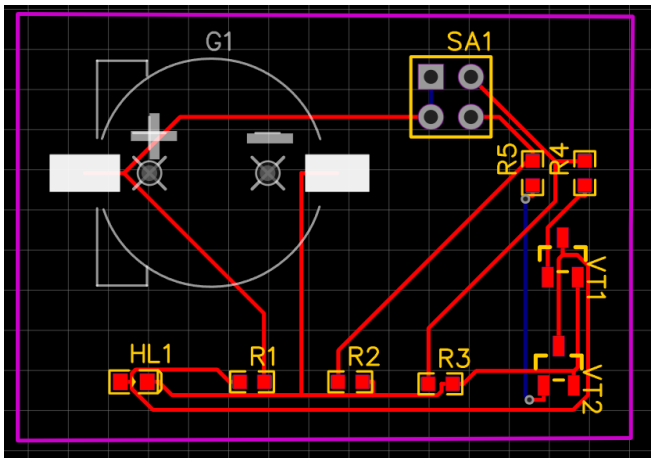
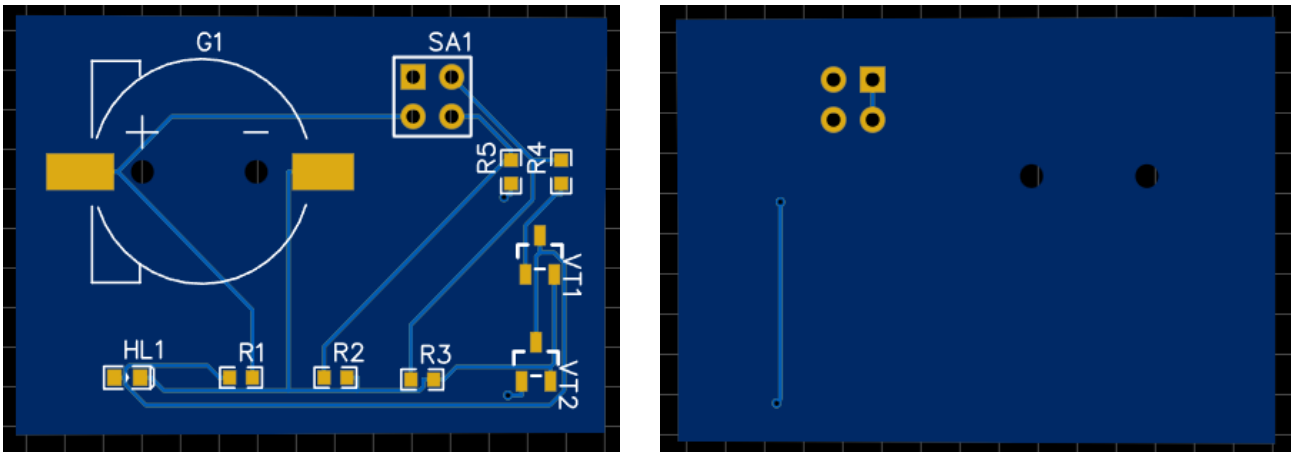
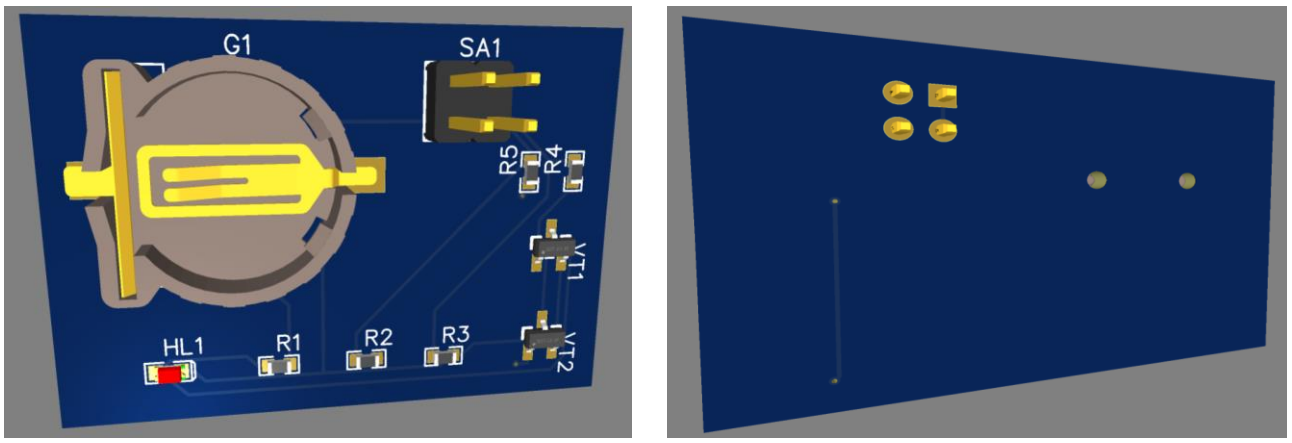


Рисунок 34. - Макет логічного елементу АБО-НІ на транзисторах



а) б)

Рисунок 35. – Макети печатної плати логічного елементу АБО-НІ в 2D



а) б)

Рисунок 36. - Макети печатної плати логічного елементу АБО-НІ в 3D

Виконати зборку схеми логічного елемента АБО-НІ на макетній платі в ***TinkerCad*** можна відповідно до рис. 37. Для побудови логічного елемента АБО-НІ на транзисторах потрібно взяти схему логічного елемента АБО та перемістити її вихід з точки з'єднання емітерів транзисторів VT1 та VT2 в точку з'єднання колекторів.

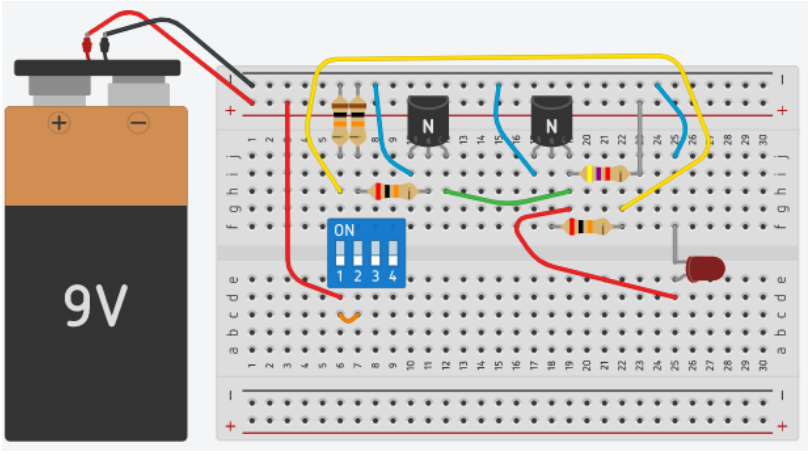


Рисунок 37. - Схема логічного елементу АБО-НІ на макетній платі

Дослідити принцип роботи даної схеми та перевірити її результати можна у відповідності до її таблиці істинності.

Контрольні запитання

1. Дайте визначення цифровому сигналу?
2. Які рівні сигналів містять основні логічні елементи?
3. Побудуйте структурну схему логічного елементу І
4. Побудуйте структурну схему логічного елементу АБО
5. Побудуйте структурну схему логічного елементу І-НІ
6. Дайте визначення аналоговому сигналу?
7. Побудуйте структурну схему логічного елементу АБО-НІ
8. Побудуйте електричну схему логічного елементу АБО
9. Побудуйте електричну схему логічного елементу НІ
10. Побудуйте електричну схему логічного елементу І-НІ
11. Побудуйте макет печатної плати схеми логічного елементу І на діодах
12. Побудуйте макет печатної плати елемента АБО-НІ на транзисторах

Практичне завдання 3.
Побудова інтегрованих логічних елементів та їх реалізація на макетних, печатних платах.

Мета роботи: побудова електричних схем на основі мікросхем, які містять логічні елементи І, АБО, НЕ в онлайн-конструкторі **EasyEDA**; набуття практичних навичок по створенню макетів печатних плат реалізованих логічних елементів з використанням мікросхем серії 74HC в ***TinkerCad***.

Задання 1. Створити електричні-принципові схеми логічних елементів з використанням мікросхем серії 74HC в ***EasyEDA***:

- ✓ Логічного елемента ***І*** (***AND***) тип мікросхеми 74HC08;
- ✓ Логічного елемента ***АБО*** (***OR***) тип мікросхеми 74HC32;
- ✓ Логічного елемента ***І-НІ*** (***NAND***) тип мікросхеми 74HC00;
- ✓ Логічного елемента ***АБО-НІ*** (***NOR***) тип мікросхеми 74HC02.

Побудувати печатні плати створених логічних елементів та зберегти їх макети в ***2D*** та ***3D*** в онлайн конструкторі ***EasyEDA***.

Побудувати схеми логічних елементів на макетних платах в ***TinkerCad*** та перевірити принцип їх функціонування відповідно до таблиць істинності.

Завдання 2. Використовуючи дискретні компоненти, створити в **EasyEDA** електричні схеми логічних елементів відповідно до заданого варіанту (табл. 19).

Таблиця 19. - Варіанти для створення логічних елементів

№ вар.	Тип мікросхеми	Кількість входів	№ вар.	Тип мікросхеми	Кількість входів
1	Мікросхема 74HC08	2	**6**	Мікросхема 74HC08	4
	Мікросхема 74HC32	4		Мікросхема 74HC32	4
	Мікросхема 74HC00	6		Мікросхема 74HC00	8
	Мікросхема 74HC02	8		Мікросхема 74HC02	6
2	Мікросхема 74HC08	8	**7**	Мікросхема 74HC08	2
	Мікросхема 74HC32	2		Мікросхема 74HC32	4
	Мікросхема 74HC00	6		Мікросхема 74HC00	6
	Мікросхема 74HC02	4		Мікросхема 74HC02	8
3	Мікросхема 74HC08	4	**8**	Мікросхема 74HC08	4
	Мікросхема 74HC32	2		Мікросхема 74HC32	2
	Мікросхема 74HC00	8		Мікросхема 74HC00	6
	Мікросхема 74HC02	6		Мікросхема 74HC02	8
4	Мікросхема 74HC08	6	**9**	Мікросхема 74HC08	4
	Мікросхема 74HC32	4		Мікросхема 74HC32	4
	Мікросхема 74HC00	2		Мікросхема 74HC00	8
	Мікросхема 74HC02	4		Мікросхема 74HC02	6
5	Мікросхема 74HC08	8	**10**	Мікросхема 74HC08	2
	Мікросхема 74HC32	4		Мікросхема 74HC32	4
	Мікросхема 74HC00	2		Мікросхема 74HC00	8
	Мікросхема 74HC02	6		Мікросхема 74HC02	6

Побудувати печатні плати створених логічних елементів та зберегти їх макети в ***2D*** та ***3D*** в онлайн конструкторі ***EasyEDA***.

Побудувати схеми логічних елементів на макетних платах в ***TinkerCad*** та перевірити принцип їх функціонування відповідно та таблиць істинності.

Зміст звіту

1. Титульна сторінка, тема та мета практичного завдання.
2. Опис виконаних завдань з наведеними рисунками, графіками, таблицями істинності, логічними виразами, структурними, логічними та електричними схемами створені в онлайн конструкторі ***EasyEDA***.
3. Побудувати печатні плати створених логічних елементів та зберегти їх макети в 2D та 3D в онлайн конструкторі ***EasyEDA***.
4. Побудувати схеми логічних елементів на макетних платах в онлайн конструкторі ***TinkerCad.***
5. Висновки, що відображають результати виконання роботи та їх експериментальний аналіз.

Теоретичні відомості

Існують окремі типи цифрових мікросхем, які містять логічні елементи лише одного типу. Наприклад мікросхема 74HC00 містить в своїй конструкції 4 логічні елементи І-НІ (рис. 38)

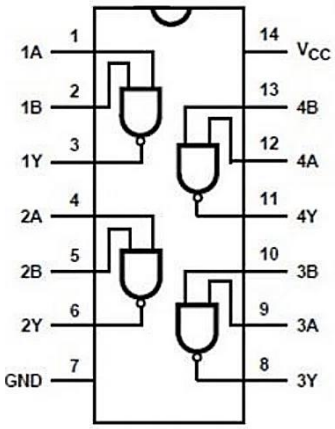


Рисунок 38. – Умовно-графічне зображення мікросхеми 74HC00.

Кожен з чотирьох логічних елементів І-НІ має по два вхідних та один інтегрований вихідний сигнали. Існують мікросхеми, які містять інші логічні елементи: АБО; І; І-НЕ; АБО-НЕ та інші. Існують такі мікросхеми, логічні елементи яких мають не по два, а по чотири та більше вхідних сигнали (рис. 39).

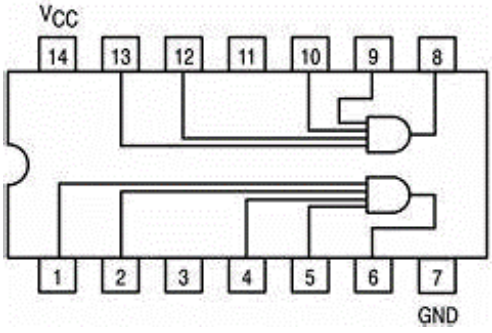


Рисунок 39. – Умовно-графічне зображення мікросхеми DM74ALS21A.

Мікросхеми побудовані на інтегрованих логічних елементах, використовуються для створення складних цифрових пристроїв, які зручніші у своїй реалізації в порівнянні з логічними елементами створеними на діодній, транзисторній, діодно-транзисторній логіці.

Достатньо встановити мікросхему в електричний ланцюг та подати живлення на необхідні вхідні сигнали. Важливо правильно розмістити мікросхему в макетні плату. Маркування на корпусі мікросхеми у вигляді ключа (рис. 40) вказує на розташування 1-го контакту.

Рисунок 40. – Розташування ключа на мікросхемах.

Розглянемо мікросхеми серії 74, в якій міститься маркування про інформацію виробника, тип мікросхеми та її функціонування.

Логічний елемент І (AND). На рис. 41 зображено схематичне позначення мікросхеми 74HC08 та розташування логічних елементів в її корпусі. Дана мікросхема містить 4 незалежних логічних елементів І, контакти якої:

- **1** (1A) та **2** (1B) – входи першого елемента;
- **3** (1Y) – вихід першого логічного елемента;
- **4** (2A) та **5** (2B) – входи другого елемента;
- **6** (2Y) – вихід другого логічного елемента;
- **9** (3A) та **10** (3B) – входи третього елемента;
- **8** (3Y) – вихід третього логічного елемента;
- **12** (3A) та **13** (3B) – входи четвертого елемента;
- **11** (3Y) – вихід четвертого логічного елемента;
- **7** – підключення до джерела живлення (полюс «-»)
- **14** – підключення до джерела живлення (полюс «+»)

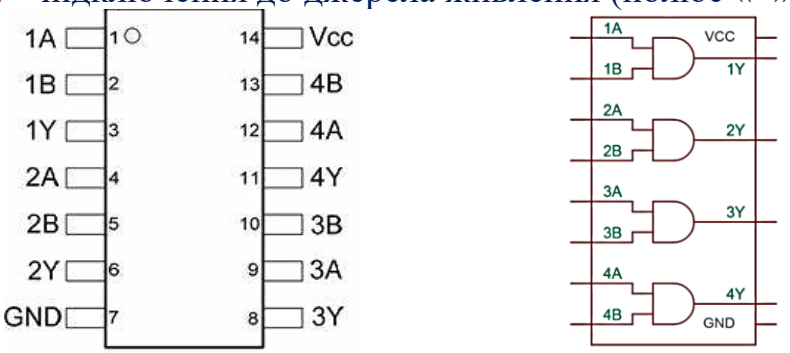


Рисунок 41. - Схематичне позначення мікросхеми 74HC08.

Максимальна дозволена напруга живлення даної мікросхеми складає 6В, рекомендована 5В. На рис. 42 представлена схема підключення мікросхеми 74HC08, яка забезпечує принцип роботи інтегрального логічного елементу І (AND). Оскільки в якості джерела живлення застосовується 9-вольтова батарея, то живлення на мікросхему подається через дільник напруги, виконаний на

резисторах R1 та R2. Крім того, для стабілізації напруги живлення даної мікросхеми можна застосувати додаткову мікросхему типу L7805.

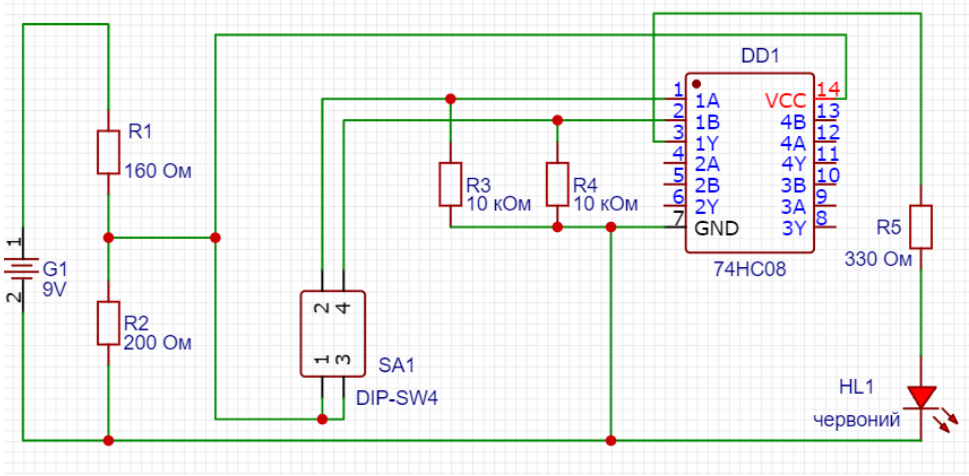


Рисунок 42. – Схема підключення мікросхеми 74HC08 (елемент AND)

Вхідні сигнали подаються за допомогою двох тумблерів DIP-перемикача SA1. Тумблер №1 підключено до входу 1A, тумблер №2 – до входу 1B. Коли тумблери вимкнені, то на відповідні входи мікросхеми подається логічний «0», оскільки вони через резистори R3 та R4 підключені до мінуса джерела живлення, який формує нульовий потенціал. Таке підключення забезпечує стійкий рівень логічного нуля за відсутності будь-якого вхідного сигналу. Зібрати схему на макетній платі в ***TinkerCad*** на основі мікросхеми 74HC08 можна відповідно до рис. 43

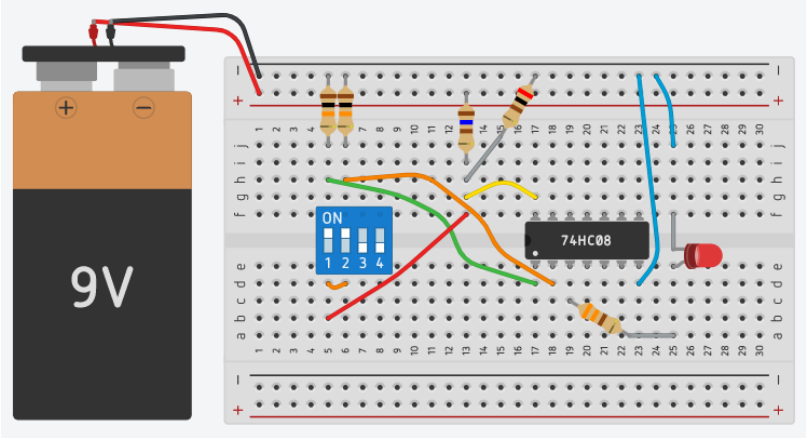


Рисунок 43. - Схема інтегрованого логічного елементу І (AND)

При включенні тумблера (положення ON) на відповідний вхід мікросхеми подається високий рівень сигналу логічна «1», а через те, що резистори R3 та R4 мають великий опір, не відбувається замикання плюса джерела живлення на мінус. Вихід логічного елементу 1Y (контакт №3) через обмежуючий резистор підключено до світлодіоду HL1, який засвітиться при логічній «1» та погасне при логічному «0».

Логічний елемент АБО (OR). На рис. 44 представлено схематичне зображення мікросхеми 74HC32, яка складається з чотирьох логічних елементів АБО. Розглянемо контакти мікросхеми:

- **1**, **2**, **4**, **5**, **9**, **10**, **12**, **13** - вхідні контакти;

- **3**, **6**, **8**, **11** - вихідні контакти;
- **7** – «-» джерела живлення;
- **14** – «+» джерела живлення.

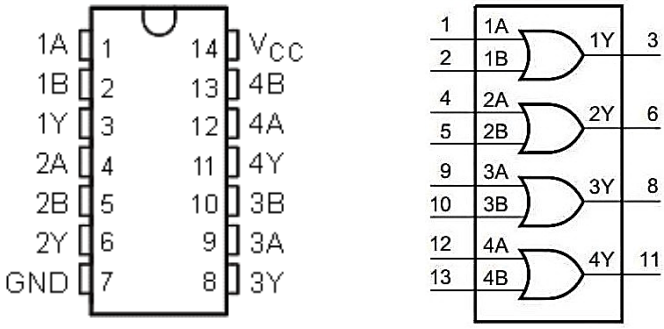


Рисунок 44. - Схематичне позначення мікросхеми 74HC32.

Максимальна дозволена напруга живлення даної мікросхеми складає 7В, рекомендована - 5В. кожен з чотирьох логічних елементів мікросхеми має по 2 входи та по 1 виходу. Інтегровані логічні елементи мікросхеми побудовані на транзисторно-діодній логіці.

На рис. 45 зображено схему підключення мікросхеми 74HC32, живлення в якій подається через дільник напруги, а вхідні контакти через резистори підключені до мінуса джерела живлення, що забезпечує стабільну роботу схеми. Світлодіод HL1 служить індикатором стану для виходів:

- **0** – не світиться;
- **1** - світиться.

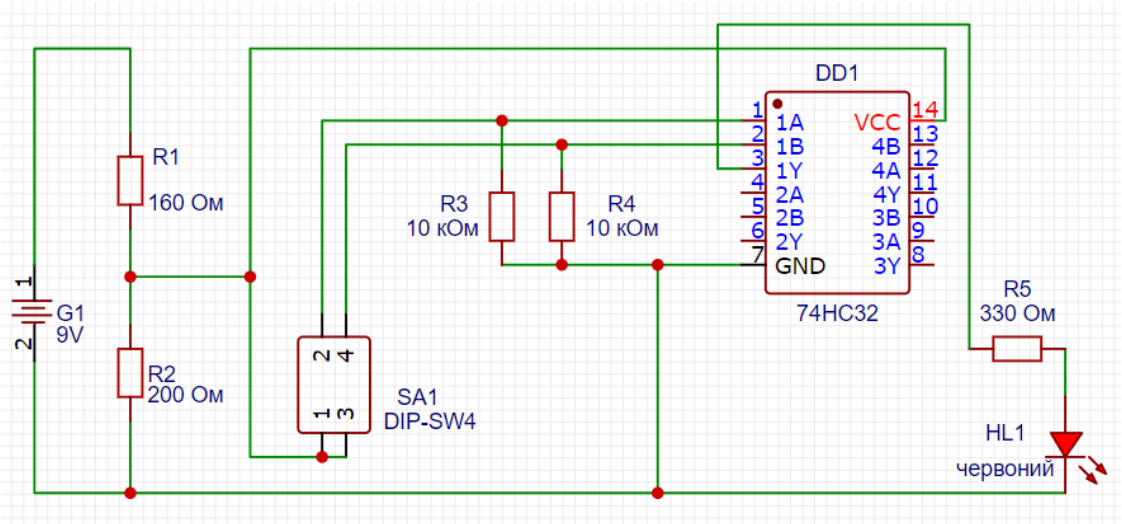


Рисунок 45. – Схема підключення мікросхеми 74HC32 (елемент OR)

При подачі сигналів A та B за допомогою тумблерів 1, 2 DIP-перемикача можна перевірити всі можливі комбінації вхідних сигналів у відповідності до таблиці істинності логічного елементу АБО, який реалізовано в мікросхемі даного типу. Для реалізації елемента АБО на мікросхемі можна замінити контакти 1, 2 та 3 на:

- **4**, **5**, **6**;
- **13**, **12**, **11**;
- **10**, **9**, **8**.

Можливо реалізувати 4-ри елементи АБО, тоді потрібно задіяти всі

контакти мікросхеми, тоді потрібно подавати на входи сигнали від окремих тумблерів, які підключені до виходів світлодіоду. У схемі підключення мікросхеми 74HC32 для перевірки принципу її роботи передбачити наступні дискретні компоненти:

- R1 резистор ємністю 160 Ом;
- R2 резистор ємністю 200 Ом;
- R3, R4 резистори ємністю 10 кОм;
- R5 резистор ємністю 330 Ом;
- HL1 світлодіод та DIP-перемикач SW4.

Зібрати схему на макетній платі в ***TinkerCad*** на основі мікросхеми 74HC32 можна відповідно до рис. 46

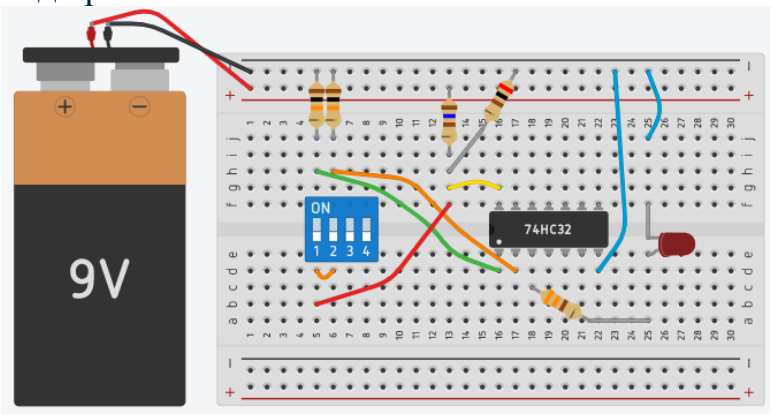


Рисунок 46. - Схема інтегрованого логічного елементу АБО (OR)

Логічний елемент І-НІ (NAND). Мікросхема типу 74HC00 містить 4-ри інтегральних логічних елементи І-НІ (NAND) кожен з яких містить два входи та один інвертований вихід (рис. 47).

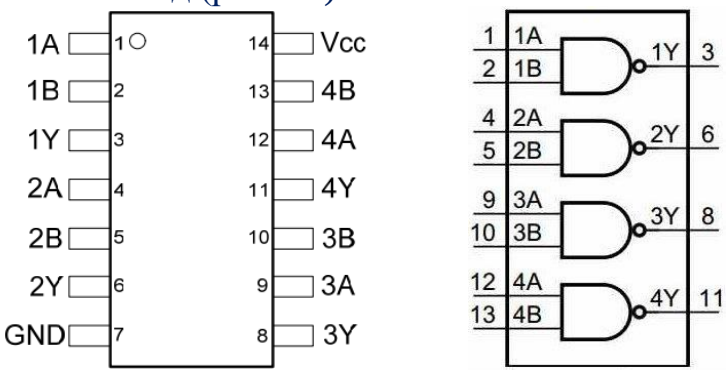


Рисунок 47. - Схематичне позначення мікросхеми 74HC00.

Електрична принципова схема підключення даної мікросхеми (рис. 48) ідентична до схем логічних елементів І та АБО. Подаючи за допомогою тумблерів DIP-перемикача вхідні сигнали, можна перевірити працездатність кожного елемента І-НІ або одночасно 4-х елементів І-НІ мікросхеми.

У схемі підключення мікросхеми 74HC00 для перевірки принципу її роботи передбачити наступні дискретні компоненти:

- R1 резистор ємністю 160 Ом;
- R2 резистор ємністю 200 Ом;
- R3, R4 резистори ємністю 10 кОм;
- R5 резистор ємністю 330 Ом;

- HL1 світлодіод;
- перемикач DIP-SW4.

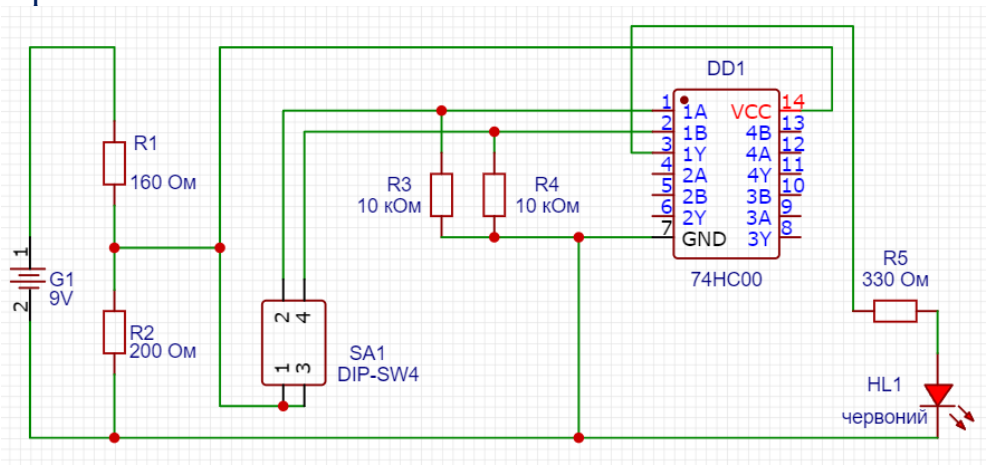


Рисунок 48. – Схема підключення мікросхеми 74HC00 (елемент NAND)

Зібрати схему в ***TinkerCad*** на макетній платі на основі мікросхеми 74HC00 можна відповідно до рис. 49

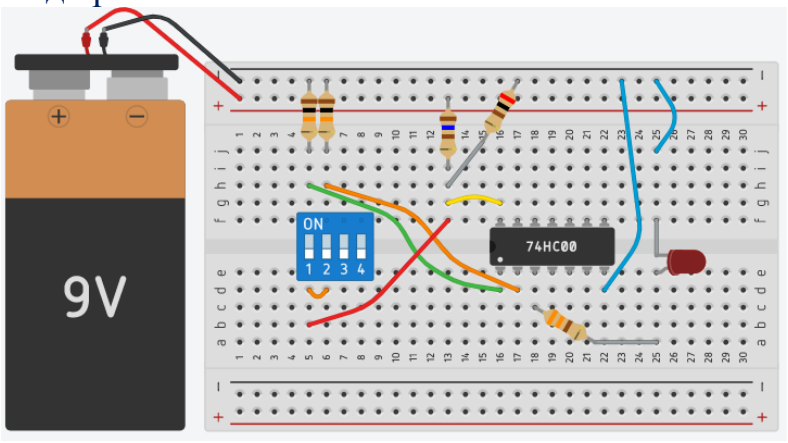


Рисунок 49. - Схема інтегрованого логічного елементу І-НІ (NAND)

Логічний елемент АБО-НІ (NOR). Схематичне позначення мікросхеми 74HC02, що містить 4-ри логічні елементи АБО-НІ, зображено на рис. 50. Кожен логічний елемент, що входить до складу мікросхеми, має по два входи та по одному інвертованому виходу.

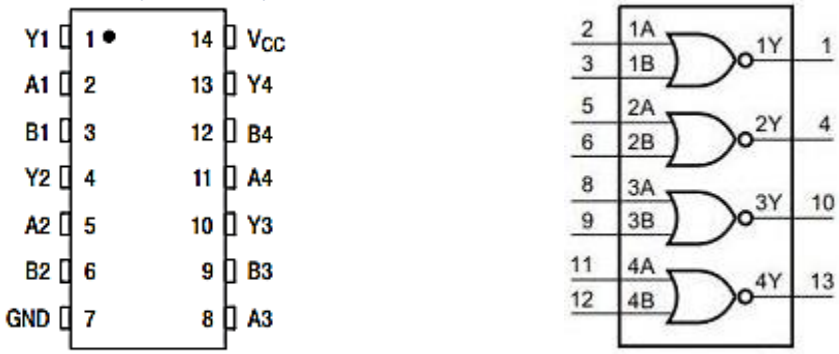


Рисунок 50. - Схематичне позначення мікросхеми 74HC02.

На відміну від мікросхем інших логічних елементів, дана мікросхема містить інший порядок підключення вхідних/вихідних контактів, а саме:

- Вхідні контакти мікросхеми: **2**, **3**, **5**, **6**, **8**, **9**, **11**, **12**.
- Вихідні контакти мікросхеми: **1**, **4**, **10**, **13**.

- Контакти підключення до джерела живлення: **7**, **14**.

Схема підключення мікросхеми показана на рис. 51. Користуючись таблицею істинності АБО-НІ, можна перевірити їх працездатність за допомогою тумблерів DIP-перемикача. У схемі підключення мікросхеми 74HC02 для перевірки принципу її роботи передбачити наступні дискретні компоненти:

- R1 резистор ємністю 160 Ом;
- R2 резистор ємністю 200 Ом;
- R3, R4 резистори ємністю 10 кОм;
- R5 резистор ємністю 330 Ом;
- HL1 світлодіод та перемикач DIP SW4.

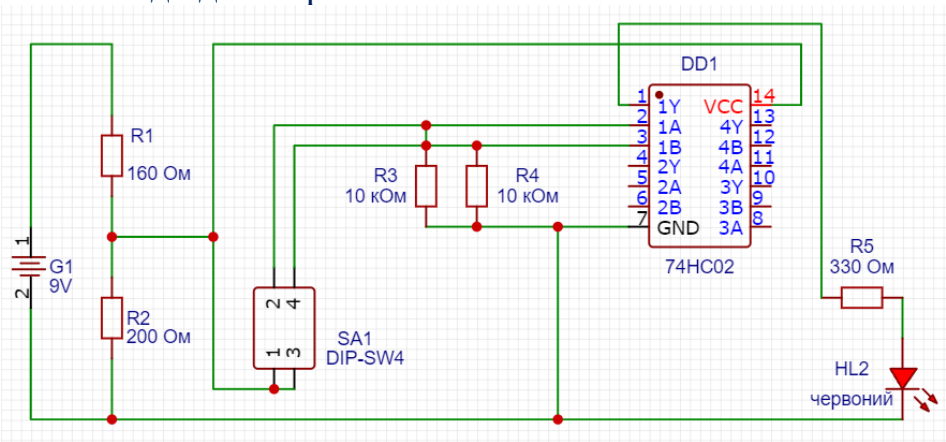


Рисунок 51. – Схема підключення мікросхеми 74HC02 (елемент NOR)

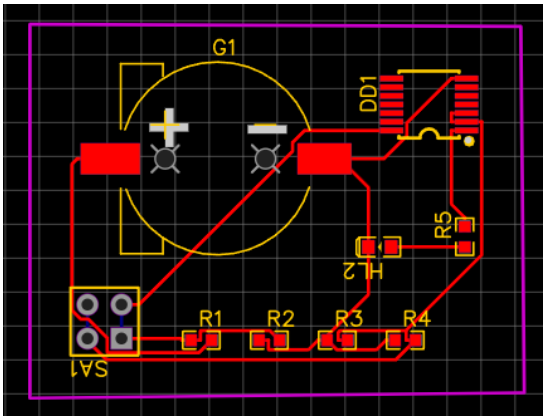
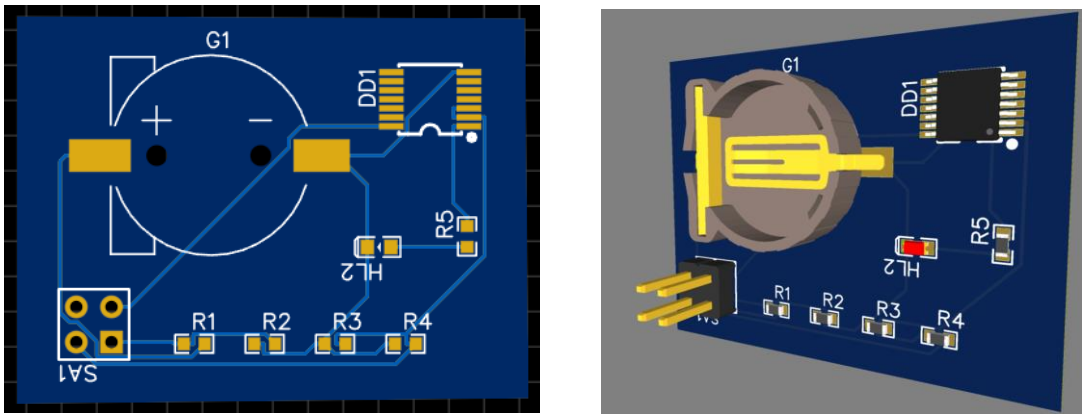


Рисунок 52. - Макет логічного елементу АБО-НІ на транзисторах



а) б)

Рисунок 53. – Макети печатної плати логічного елементу АБО-НІ в 2D/3D

Після проектування електричної схеми в ***EasyEDA*** створимо макет печатної плати (рис. 52) та переглянемо її у форматі **2D** (рис. 53а) та **3D** (рис. 53б). Як видно з наведених рисунків, на платі печатні треки подані тільки з однієї сторони, що свідчить про оптимальне розміщення інтегрованих компонентів на платі. Зібрати схему на макетній платі в ***TinkerCad*** на основі мікросхеми 74HC02 можна відповідно до рис. 54.

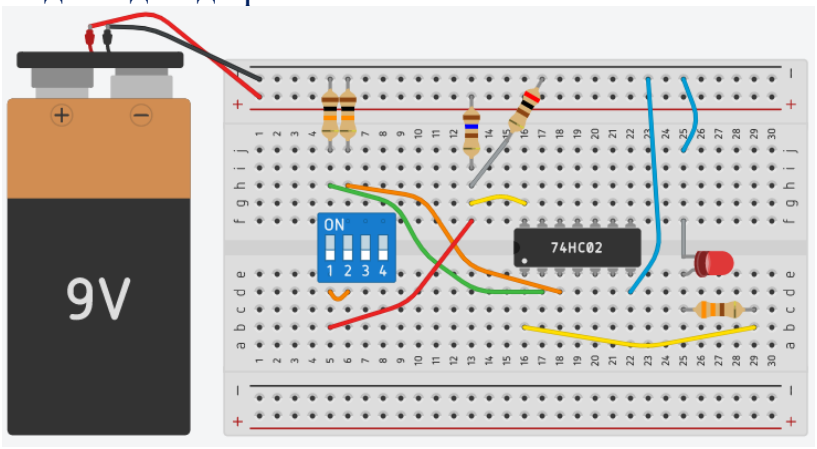


Рисунок 54. - Схема інтегрованого логічного елементу АБО-НІ (NOR)

Задаючи за допомогою тумблерів різні вхідні сигнали та спостерігаючи за поведінкою світлодіоду, можна скласти таблицю істинності відповідних елементів.

Контрольні запитання

1. Побудуйте структурну схему інтегрованого елементу І
2. Побудуйте структурну схему інтегрованого елементу АБО
3. Побудуйте структурну схему інтегрованого елементу І-НЕ
4. Побудуйте структурну схему інтегрованого елементу АБО-НЕ
5. Яку функцію виконують резистори **R1, R2** в електричних принципових схемах, реалізованих на основі інтегрованих елементів?
6. Яка мікросхема відповідає логічному елементу **І?**
7. Яка мікросхема відповідає логічному елементу **АБО?**
8. Яка мікросхема відповідає логічному елементу **І-НЕ?**
9. Яка мікросхема відповідає логічному елементу **АБО-НЕ?**
10. Приведіть умовно-графічне зображення світлодіоду.
11. Приведіть умовно-графічне зображення мікросхеми 74HC08.
12. Приведіть умовно-графічне зображення мікросхеми 74HC32.
13. Приведіть умовно-графічне зображення мікросхеми 74HC00.
14. Приведіть умовно-графічне зображення мікросхеми 74HC02.

Практичне завдання 4.
Дослідження принципів роботи комбінаційних мікросхем та їх реалізація на макетних, печатних платах.

Мета роботи: ознайомлення з принципами роботи комбінаційних мікросхем: дешифратора, шифратора, мультиплексора та демультиплексора; побудова електричних схем в ***EasyEDA*** на основі мікросхем серії 74: шифратора 74HC148, дешифратора 74HC238, мультиплексора 74HC151, демульплексора 74HC238; набуття практичних навичок по створенню макетів печатних плат з використанням реалізованих комбінаційних мікросхем в онлайн-конструкторах ***EasyEDA***; набуття практичних навичок у створенні макетів печатних плат реалізованих логічних елементів в онлайн конструкторі ***TinkerCad***; експериментальне дослідження комбінаційних мікросхем в пакетах ***Electronics Workbench v5.12, Multisim.***

Задання 1. Створити електричні-принципові схеми комбінаційних пристроїв в ***EasyEDA*** з використанням мікросхем 74-ї серії:

- ✓ шифратора на мікросхемі типу 74HC148;
- ✓ дешифратора на мікросхемі типу 74HC238;
- ✓ мультиплексора на мікросхемі типу 74HC151;
- ✓ демультиплексора на мікросхемі типу 74HC238;

Побудувати печатні плати створених комбінаційних пристроїв та зберегти їх макети в ***2D*** та ***3D*** в онлайн конструкторі ***EasyEDA***.

Побудувати схеми комбінаційних пристроїв на макетних платах конструктора «Цифрова електроніка» та/або в ***TinkerCad***, перевірити принцип їх функціонування відповідно до таблиць істинності.

Створити розгорнуту структуру шифратора на логічних елементах в онлайн конструкторі в ***Logic.ly***.

Завдання 2. Використовуючи дискретні компоненти, створити в ***EasyEDA*** електричні схеми комбінаційних пристроїв відповідно до заданого варіанту (табл. 20).

Таблиця 20. - Варіанти для побудови комбінаційних пристроїв

№ вар.	Комбінаційний пристрій	Кількість розрядів	№ вар.	Комбінаційний пристрій	Кількість розрядів
1	шифратор	2	**6**	шифратор	4
	дешифратор	3		дешифратор	3
	мультиплексор	3		мультиплексор	2
	демультиплексор	4		демультиплексор	1
2	шифратор	3	**7**	шифратор	2
	дешифратор	2		дешифратор	4
	мультиплексор	3		мультиплексор	3
	демультиплексор	4		демультиплексор	1
3	шифратор	4	**8**	шифратор	3
	дешифратор	2		дешифратор	2
	мультиплексор	3		мультиплексор	2
	демультиплексор	2		демультиплексор	1

№ вар.	*Комбінаційний пристрій*	*Кількість розрядів*	*№ вар.*	*Комбінаційний пристрій*	*Кількість розрядів*
4	шифратор	2	**9**	шифратор	3
	дешифратор	4		дешифратор	4
	мультиплексор	2		мультиплексор	2
	демультиплексор	3		демультиплексор	6
5	шифратор	1	**10**	шифратор	2
	дешифратор	3		дешифратор	4
	мультиплексор	2		мультиплексор	1
	демультиплексор	4		демультиплексор	3

Побудувати печатні плати створених комбінаційних пристроїв та зберегти їх макети в ***2D*** та ***3D*** в онлайн конструкторі ***EasyEDA***.

Побудувати схеми комбінаційних пристроїв на макетних платах конструктора «Цифрова електроніка» та/або в ***TinkerCad***, перевірити принцип їх функціонування відповідно до таблиць істинності.

Завдання 3. Створити в пакеті ***Multisim*** та/або ***Electronics Workbench*** електричні принципові схеми з використанням мікросхем 74-ї серії типу 74HC148, 74HC238, 74HC151 для перевірки принципів роботи шифратора, дешифратора, мультиплексора, демультиплексора у відповідності до їх таблиць істинності. Зберегти файли комбінаційних пристроїв, записати результати їх роботи.

Зміст звіту

1. Титульна сторінка, тема та мета практичного завдання.
2. Опис виконаних завдань з наведеними рисунками, графіками, таблицями істинності, логічними виразами, структурними, логічними та електричними схемами створені в онлайн конструкторі ***EasyEDA***.
3. Побудувати печатні плати створених комбінаційних пристроїв та зберегти їх макети в 2D та 3D в онлайн конструкторі ***EasyEDA***.
4. Побудувати схеми комбінаційних пристроїв на макетних платах в онлайн конструкторі ***TinkerCad.***
5. Розгорнуті структурні схеми комбінаційних приладів на логічних елементах, створені в онлайн конструкторі в ***Logic.ly***.
6. Висновки, що відображають результати виконання роботи та їх експериментальний аналіз.

Теоретичні відомості

За принципом роботи цифрові логічні прилади поділяються на два типи:

- *комбінаційні;*
- *послідовні.*

Комбінаційними приладами називаються такі, виходи яких безпосередньо залежать від стану входів. До таких приладів відносяться:

- *шифратори;*
- *дешифратори;*
- *мультиплексори;*

- *демультиплексори;*
- *суматори;*
- *цифрові компаратори.*

Послідовними називаються такі логічні цифрові прилади, виходи яких залежать не тільки від вхідних сигналів, а й від внутрішніх станів приладів. До послідовних пристроїв відносяться:

- *тригери;*
- *регістри;*
- *лічильники.*

Розглянемо комбінаційну логіку відомих комбінаційних приладів.

Комбінаційний пристрій: шифратор. *Шифратор* - цифровий пристрій, що виконує операцію перетворення коду із однієї системи числення в іншу. Найчастіше в цифровій електроніці використовуються шифратори, що перетворюють десяткові числа в двійковий код.

Як правило, шифратор має n-виходів та 2^n входів. Наприклад, шифратор має 3-ри виходи, отже, кількість його входів дорівнює $2^3 = 8$. Це означає, що даний шифратор може зашифрувати десяткові числа від 0 до 7 у відповідний двійковий код. На виході зашифроване число буде представлене трьома двійковими розрядами.

Десяткове число задається шляхом подачі нуля або одиниці на відповідний вхід шифратора (рівень сигналу, який необхідно подати залежить від типу шифратора). На рис. 55 показано схематичне позначення мікросхеми шифратора типу 74HC148.

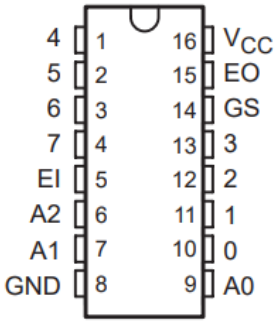


Рисунок 55. - Схематичне позначення мікросхеми 74HC148.

Входами від 0 до 7 є контакти: **10**, **11**, **12**, **13**, **1**, **2**, **3**, **4** відповідно. ***Вихідними*** контактами від **A0** до **A2** є: **9**, **7**, **6**. Контакти підключення джерела живлення - **8** та **16**. Додано ще один вхідний контакт E1 (**5**). Для початку роботи шифратора слід подати на цей контакт логічний «0» - сигнал дозволу роботи. Оскільки даний шифратор побудований на простих логічних елементах (рис. 56), то можна скласти таблицю істинності режимів його роботи (табл. 21). Умовні позначення, які наведено в таблиці істинності шифратора означають наступне:

- **A0** - молодший розряд двійкового числа;
- **A2** – старший розряд двійкового числа;
- **L** – низький рівень сигналу (low - 0);
- **H** - високий рівень сигналу (high - 1);
- **X** – нейтральний рівень сигналу.

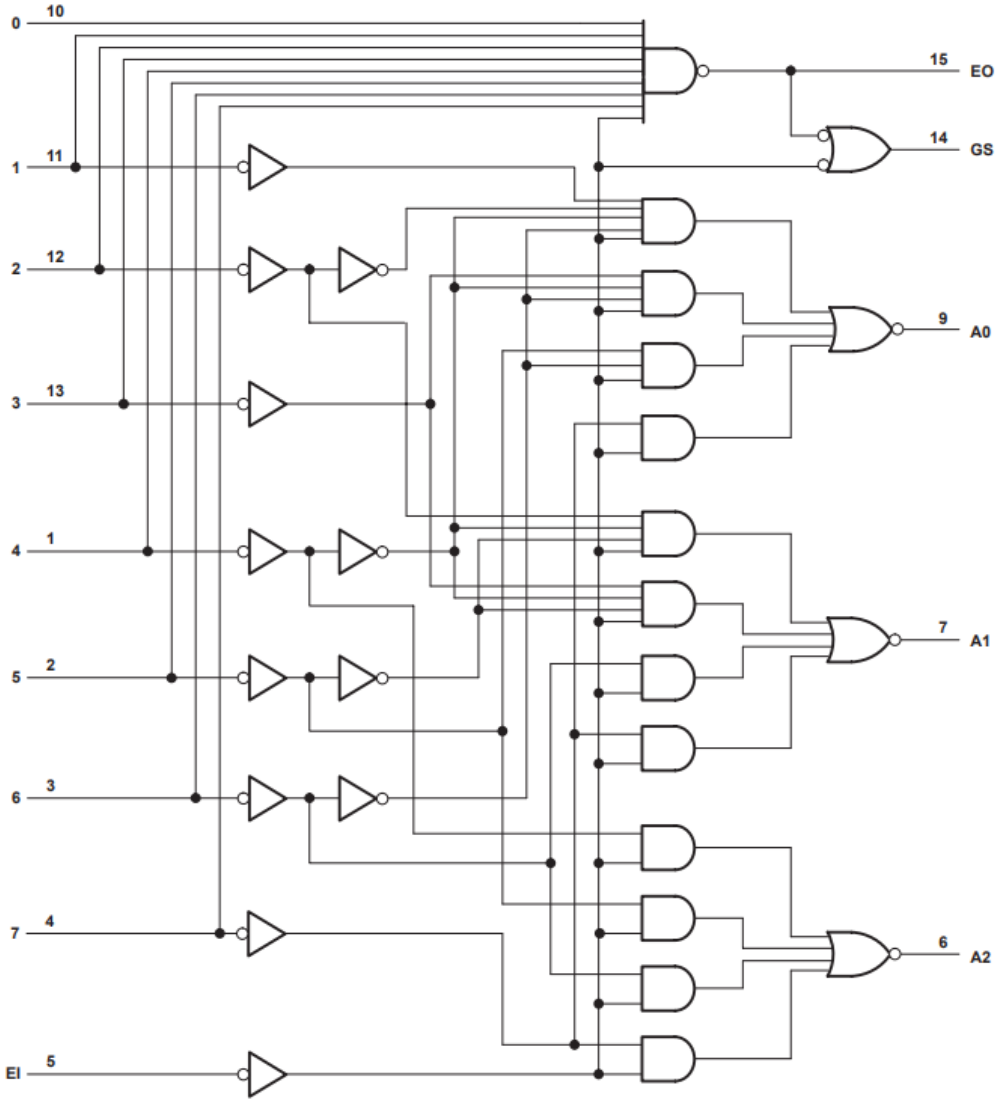


Рисунок 56. – Розгорнути структура шифратора на логічних елементах.

Таблиця 21. – Таблиця істинності шифратора 74HC148.

Число	Вхідні сигнали									Вихідні сигнали				
	E1	0	1	2	3	4	5	6	7	A2	A1	A0	GS	E0
	H	X	X	X	X	X	X	X	X	H	H	H	H	H
	L	H	H	H	H	H	H	H	H	H	H	H	H	L
0	L	L	X	X	X	X	X	X	X	H	H	H	L	H
1	L	X	L	H	H	H	H	H	H	H	H	L	L	H
2	L	X	X	L	H	H	H	H	H	H	L	H	L	H
3	L	X	X	X	L	H	H	H	H	H	L	L	L	H
4	L	X	X	X	X	L	H	H	H	L	H	H	L	H
5	L	X	X	X	X	X	L	H	H	L	H	L	L	H
6	L	X	X	X	X	X	X	L	H	L	L	H	L	H
7	L	X	X	X	X	X	X	X	L	L	L	L	L	H

Для того щоб створити модель шифратора, необхідно підключити до джерела живлення контакти мікросхеми 74HC148 та подати вхідні сигнали відповідно до рис. 57.

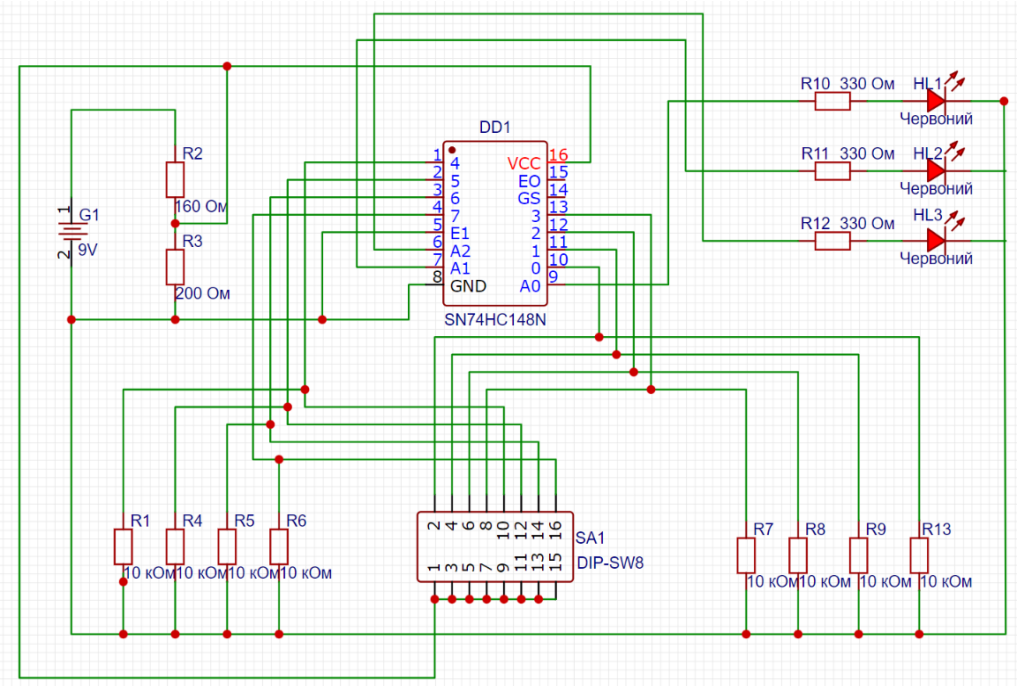


Рисунок 57. – Схема підключення мікросхеми 74HC148 (шифратора)

У схемі підключення мікросхеми 74HC148 для перевірки принципу її роботи передбачити наступні дискретні компоненти:

- мікросхема DD1 типу 74HC148 (SN74HC148N)
- R1, R4, R5, R6, R7, R8, R9, R13 резистори ємністю 10 кОм;
- R2 резистор ємністю 160 Ом;
- R3 резистор ємністю 200 Ом;
- R10, R11, R12 резистори ємністю 330 Ом;
- HL1, HL2, HL3 червоні світлодіоди;
- SA1 DIP-перемикач SW8;
- батарея живлення G1 9В.

Входи мікросхеми підключені до DIP-перемикача із вісьмома тумблерами, та через резистори Rl, R4-R9 та R13 номіналом 10 кОм, що підключені до мінуса джерела живлення. Коли тумблери вимкнені, на входах шифратора присутні логічні нулі. Другий ряд контактів DIP-перемикача підключено до середньої точки дільника напруги. тобто на них подається високий рівень напруги (логічна «1»). Виходи шифратора через обмежуючі резистори підключені до світлодіодів. **HL1** відповідає молодшому розряду вихідного двійкового коду (**A0**), **HL2** - **A1**, **HL3** - старший розряд двійкового коду (**A2**). Схема підключення шифратора наведена на рис. 58.

Для перевірки принципу роботи шифратора, необхідно встановити тумблер №1 DIP-перемикача у положення ON. На нульовий вхід мікросхеми (контакт **10**) подається низький рівень напруги (логічний «0»), на всі інші контакти – високий рівень напруги (логічна «1»). Це відповідає даним третього рядка таблиці істинності, яку наведено в табл. 22, тобто відповідає десятковому числу 0.

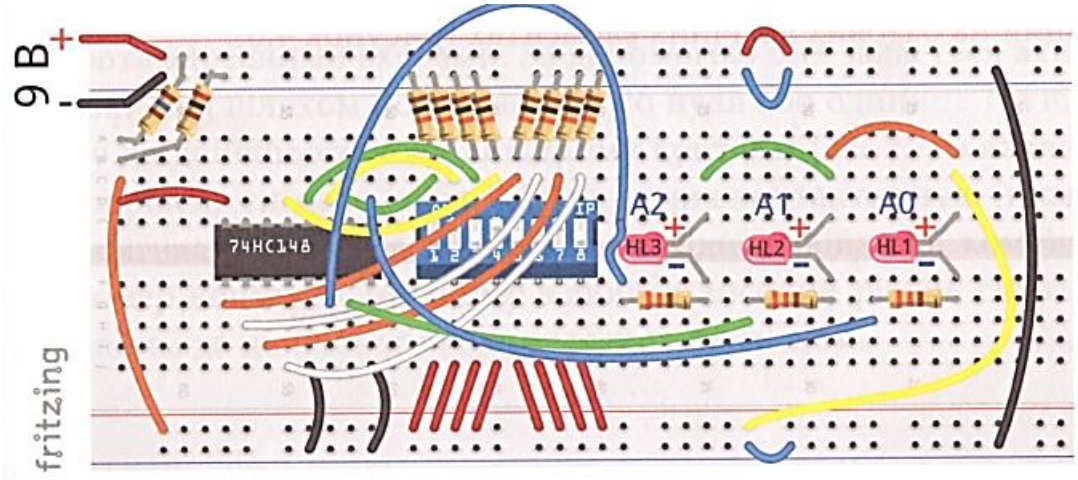


Рисунок 58. - Схема підключення шифратора на мікросхемі 74HC148.

Таблиця 22. – Таблиця істинності функціонування шифратора.

Число	Вхідні сигнали									Вихідні сигнали		
	E1	0	1	2	3	4	5	6	7	A2	A1	A0
	1	X	X	X	X	X	X	X	X	1	1	1
	0	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1	1	1
1	0	X	0	1	1	1	1	1	1	1	1	0
2	0	X	X	0	1	1	1	1	1	1	0	1
3	0	X	X	X	0	1	1	1	1	1	0	0
4	0	X	X	X	X	0	1	1	1	0	1	1
5	0	X	X	X	X	X	0	1	1	0	1	0
6	0	X	X	X	X	X	X	0	1	0	0	1
7	0	X	X	X	X	X	X	X	0	0	0	0

На виході всі світлодіоди засвітились, тобто на всіх трьох виходах **A0-A2** присутні логічні одиниці: **A2=1**, **A1=1**, **A0=1**, що не відповідає дійсності. Поданий сигнал на нульовий вхід (контакт **10**), повинен видавати на виході двійковий код нуля (**000**), а отримано (**111**). В чому ж помилка? Помилки немає. Для даної мікросхеми-шифратора, для отримання двійкового коду вхідного десяткового числа, потрібно інвертувати результат на виводах **A0-A2:** $\overline{\mathbf{111}}$ **= 000**. Отже, в результаті інверсії отримаємо двійковий код **000**, який відповідає десятковому числу **0**.

Переведемо тумблер №1 DIP-перемикача в положення ON, а тумблер №2 в положення OFF, то на перший вхід (контакт **11**) подається логічний «0», а на всі інші – логічні «1». Засвітились світлодіоди **HL2** (**A1=1**), **HL3** - (**A2=1**), світлодіод HL1 не світиться (A0=0). Отже вихідна комбінація **A2 A1 A0 = 110** (див. четвертий рядок табл. 23).

Якщо інвертувати даний код $\overline{\mathbf{110}}$**= 001**, то отримаємо двійковий код, який відповідає десятковій одиниці. За такою методикою відповідно до таблиці 1 почергово можна перевірити інші можливі комбінації двійкового коду шифратора.

Таблиця 23. – Таблиця істинності функціонування шифратора.

Число	*Вхідні сигнали*									*Вихідні сигнали*		
	E1	0	1	2	3	4	5	6	7	A2	A1	A0
	1	X	X	X	X	X	X	X	X	1	1	1
	0	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1	1	1
1	0	X	0	1	1	1	1	1	1	1	1	0
2	0	X	X	0	1	1	1	1	1	1	0	1
3	0	X	X	X	0	1	1	1	1	1	0	0
4	0	X	X	X	X	0	1	1	1	0	1	1
5	0	X	X	X	X	X	0	1	1	0	1	0
6	0	X	X	X	X	X	X	0	1	0	0	1
7	0	X	X	X	X	X	X	X	0	0	0	0

ВАЖЛИВО! Якщо випадково подати на два чи більше входи логічний «0», то на виході отримаєте двійковий код, який відповіде найбільшому номеру задіяного входу. Наприклад, якщо на 7-й га на 0-й входи подати логічний нуль (тумблери №8 та №1 в положення OFF), то на виході не засвітиться жоден світлодіод - **000**. Цей код в інвертованому вигляді відповідає десятковому числу **7**. Якщо подати на 3-й і на 5-й входи логічні «0» (тумблери №4 та №6 в положення OFF), то на виході буде отримано код **010** $\rightarrow$ $\overline{\mathbf{010}}$**= 101**, а це код числа **5** в десятковій системі числення.

Для даної мікросхеми для шифрування необхідного десяткового числа, на відповідний цьому числу вхідний контакт потрібно подавати логічний «0», щоб закодувати десяткове число 1; при подачі на 1-й вхідний контакт логічного «**0**»; якщо число **7** – то на 7-й вхідний контакт. Існують інші типи шифраторів, які розпізнають десяткові числа при надходженні на відповідний контакт логічної «**1**». Причому вихідний двійковий код не потрібно інвертувати для перевірки результату роботи шифратора.

Якщо світлодіоди, які встановлені на виході мікросхеми, підключити навпаки (катод через резистор до виходу мікросхеми, а анод до середньої точки дільника напруги), то результат, який видає шифратор ***не*** потрібно інвертувати. Таким чином, світлодіоди автоматично інвертують результат роботи шифратора (рис. 58)

Комбінаційний пристрій дешифратор. Дешифратор - цифровий пристрій, що виконує обернену до шифратора функцію - перетворює вхідний двійковий код у відповідне йому десяткове число.

Кількість контактів шифратора обернена до дешифратора. Якщо дешифратор має n-входів, то кількість його виходів дорівнює $\mathbf{2^n}$. Кількість вхідних контактів визначає розрядність двійкового коду, який він здатен дешифрувати. Наприклад, якщо входів всього 3-ри, то максимально можливе значення числа, яке можна передати за його допомогою, дорівнює: $\mathbf{111_2=7_{10}}$. Якщо кількість входів збільшити до 4-х, то кількість виходів збільшиться до 16-ти, а максимально можливе значення числа, що здатен обробляти дешифратор

становить: $\mathbf{1111_2=15_{10}}$

При подачі на вхідні контакти двійковий, коду, на відповідному вихідному контакті з'являється високий рівень сигналу. Номер контакту в десятковому вигляді відповідає двійковому коду, поданому на вхід. На всіх інших виходах буде подано низький рівень сигналу, а на інших високий. Це залежить від того інвертовані виходи дешифратора чи прямі.

Наприклад, дешифратор має три вхідні контакти: **A0**, **A1**, **A2**. На них подана комбінація **001**, тоді **A0** - молодший розряд, **A2** – старший розряд (табл. 24).

Таблиця 24. – Нумерація розрядності дешифратора.

A2	*A1*	*A0*
0	0	1

Пам'ятаючи про спосіб переведення двійкового коду в десятковий, отримуємо відповідне десяткове число **1**. Це означає, що на першому виході дешифратора з'явиться логічний нуль або одиниця (залежить від типу дешифратора). Всі інші виходи мають протилежний до нього рівень сигналу.

На відміну від шифратора, дешифратор виконує обернену операцію – дешифрує двійковий код у відповідне десяткове число. Дешифратор належить до комбінаційних пристрої та реалізований в мікросхемі типу **74HC238**, схематичне зображення якої наведено на рис. 59.

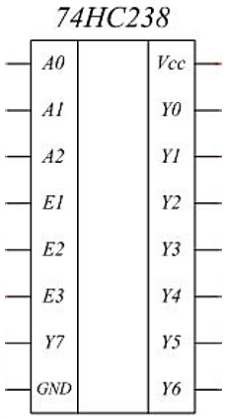


Рисунок 59. - Схематичне позначення мікросхеми 74HC238.

Мікросхема-дешифратор **74HC238** має 3 входи **A0**, **A1**, **A2**, які називаються адресними входами. За допомогою них задається вхідний двійковий код шляхом подачі логічного «**0**» або «**1**». На відповідному до двійкового коду вихідному контакті (**Y0-Y7**) з'являється високий рівень сигналу (табл. 25), на інших контактах «**0**» (низький рівень сигналу). Це і є декодування вхідної двійкової комбінації, адже номер контакту, на якому з'явився високий рівень сигналу, є відповідним до двійкового коду десятковим числом.

Умовні позначення, які наведено в таблиці істинності дешифратора означають наступне:

- **A0** - молодший вхідний розряд десяткового числа;
- **A2** – старший вхідний розряд десяткового числа;
- **L** – низький рівень сигналу (low - 0);
- **H** - високий рівень сигналу (high - 1);
- **X** – нейтральний рівень сигналу;

- **Y0** - молодший вихідний розряд двійкового числа;
- **Y7** - старший вихідний розряд двійкового числа.

Таблиця 25. – Таблиця істинності дешифратора 74HC238.

Число	*Вхідні сигнали*						*Вихідні сигнали*							
	E1	E2	E3	A2	A1	A0	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
	H	X	X	X	X	X	L	L	L	L	L	L	L	L
	X	H	X	X	X	X	L	L	L	L	L	L	L	L
	X	X	L	X	X	X	L	L	L	L	L	L	L	L
0	L	L	H	L	L	L	L	L	L	L	L	L	L	H
1	L	L	H	L	L	H	L	L	L	L	L	L	H	L
2	L	L	H	L	H	L	L	L	L	L	L	H	L	L
3	L	L	H	L	H	H	L	L	L	L	H	L	L	L
4	L	L	H	H	L	L	L	L	L	H	L	L	L	L
5	L	L	H	H	L	H	L	L	H	L	L	L	L	L
6	L	L	H	H	H	L	L	H	L	L	L	L	L	L
7	L	L	H	H	H	H	H	L	L	L	L	L	L	L

Входи **E1-E3** - керуючі входи, які заведені на логічний елемент «**І**». Ці входи задають режим роботи мікросхеми **74HC238**: дешифратор або демультиплексор. Для реалізації дешифратора можна створити електричну схему, яку наведено на рис. 60. Керуючі входи **E1** та **E2** підключені до мінуса джерела живлення, а **E3** – до середньої точки дільника напруги (подача високого рівня напруги, як і на контакт **Vcc**).

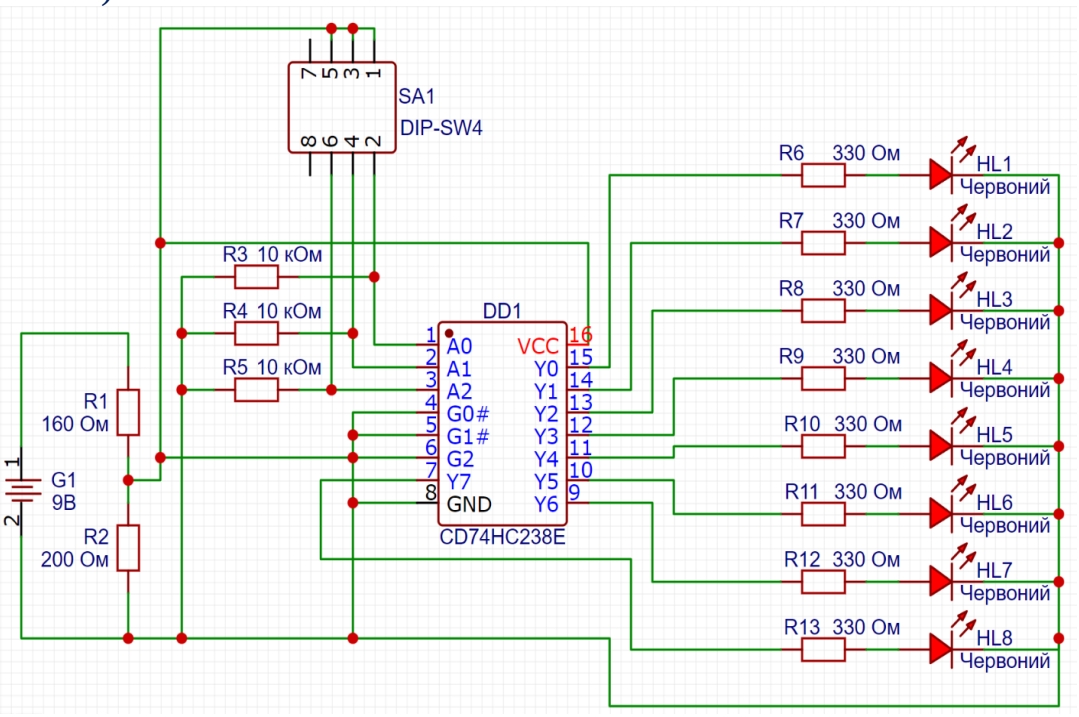


Рисунок 60. – Схема підключення мікросхеми 74HC238 (дешифратора)

У схемі підключення мікросхеми 74HC238 для перевірки принципу її роботи передбачити наступні дискретні компоненти:

- мікросхема DD1 типу 74HC238 (CD74HC238E)
- R1 резистор ємністю 160 Ом;

- R2 резистор ємністю 200 Ом;
- R3, R4, R5 резистори ємністю 10 кОм;
- R6- R13 резистори ємністю 330 Ом;
- HL1-HL8 червоні світлодіоди;
- SA1 DIP-перемикач SW4;
- батарея живлення G1 9В.

Адресні входи підключені до DIP-перемикача. У вимкненому стані хоча б одного тумблера, на відповідний контакт мікросхеми подається стабільний логічний «0» (низький рівень сигналу), адже всі входи через резистори **R3-R5** підключені до мінуса джерела живлення. Таке підключення забезпечує стабільну роботу мікросхеми. При переведенні тумблера в положення ON на відповідний вхід подається логічна «1». Тумблер №1 заведено на вхід **A0**, тумблер №2 – на вхід **A1**, тумблер №3 – на вхід **A2**.

Вхідні контакти дешифратора підключені до світлодіодів **HL1-HL8**. При подачі на вхід певної двійкової комбінації, на виході засвітиться той світлодіод, який підключено до виходу, чий порядковий номер в десятковій системі числення відповідає двійковому коду на вході. Для розуміння принципу роботи дешифратора, можна зібрати на макетній платі схему, яку наведено на рис. 61.

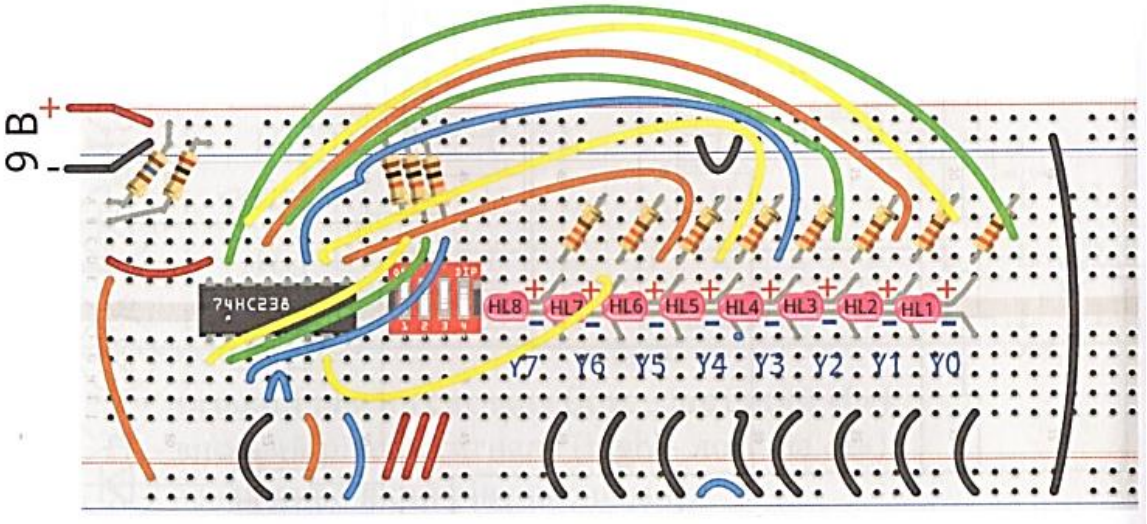


Рисунок 61. - Схема підключення дешифратора.

За допомогою тумблерів №1, 2, 3 DIP-перемикача, можна задати вхідний двійковий код. Наприклад, нехай всі тумблери знаходяться у вимкненому стані, що означає комбінацію 000, що відповідає десятковому 0. Тоді загориться світлодіод HL1, який підключено до виходу Y0.

Коли тумблер №1 перевести в положення ON, а інші залишаються у вимкненому стані, то подається комбінація двійкового коду **001**, що відповідає десятковій одиниці. Тоді загориться світлодіод **HL2**, який підключено до виходу **Y1**, і т. д. За допомогою трьох тумблерів, що підключені до входів **A0-A2** дешифратора, можна декодувати числа від 0 до 7 (табл. 26).

Самостійно можна перевірте результати роботи дешифратора та записати їх у таблицю 26.

Таблиця 26. – Результати дослідження роботи дешифратора.

Вхідний контакт	**A2**	**A1**	**A0**	*Відповідне двійковому коду десяткове число*	*Порядковий номер засвіченого світлодіоду*	*Номер виходу дешифратора*
№ тумблера	**3**	**2**	**1**			
Двійкова комбінація на вході дешифратора	**0**	**0**	**0**	**0**	**HL1**	**Y0**
	0	**0**	**1**	…	…	…
	0	**1**	**0**	…	…	…
	0	**1**	**1**	…	…	…
	1	**0**	**0**	…	…	…
	1	**0**	**1**	…	…	…
	1	**1**	**0**	…	…	…
	1	**1**	**1**	…	…	…

Комбінаційний пристрій мультиплексор. *Мультиплексор* - цифровий прилад, що має декілька входів та один вихід і призначений для передачі на вихід сигналу з одного із входів мікросхеми. Номер входу, з якого буде передано сигнал на вихід, визначається комбінацією на керуючих (адресних) контактах. Тобто керуючі входи визначають, який з інформаційних входів потрібно підключити до виходу.

Як правило, мультиплексор має m-керуючих (адресних) входи та $\mathbf{2^m}$ інформаційні входи. Такі мультиплексори називають ***повними***. Наприклад, мікросхема мультиплексора має 2 керуючих входи, отже, вона повинна мати $2^2=4$ інформаційні входи. На рис. 62 представлено схематичне позначення мікросхеми типу 74HC151 мультиплексора.

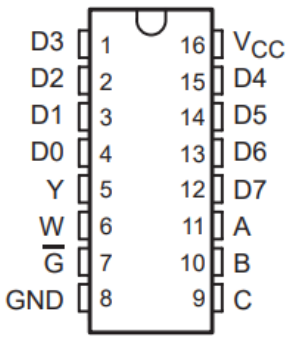


Рисунок 62. - Схематичне позначення мікросхеми 74HC151.

Мікросхема має 3-ри керуючих входи S0-S2, які вказують на те, з якого із 8-ми інформаційних входів (**I0-I7**) сигнал буде передано на вихід **Y**. Даний мультиплексор має інверсний вихід,сигнал на якому протилежний до виходу **Y**. Інверсний вхід $\overline{E}$ вмикає/вимикає мультиплексор. Активним рівнем сигналу для входу **E** є логічний «**0**», тобто для початку роботи з мікросхемою, потрібно подати на цей контакт низький рівень сигналу, відповідно до наведеної таблиці істинності (табл. 27).

Умовні позначення, які наведено в таблиці істинності дешифратора означають наступне:

- **L** – низький рівень сигналу (low - 0);
- **H** - високий рівень сигналу (high - 1);

- X – нейтральний рівень сигналу.

Таблиця 27. – Таблиця істинності мультиплексора 74HC151.

№ рядка	Вхідні сигнали												Вихідні сигнали	
	$\overline{E}$	S2	S1	S0	I0	I1	I2	I3	I4	I5	I6	I7	Y	$\overline{Y}$
1	H	X	X	X	X	X	X	X	X	X	X	X	L	H
2	L	L	L	L	L	X	X	X	X	X	X	X	L	H
3	L	L	L	L	H	X	X	X	X	X	X	X	H	L
4	L	L	L	H	X	H	X	X	X	X	X	X	L	H
5	L	L	L	H	X	H	X	X	X	X	X	X	H	L
6	L	L	H	L	X	X	H	X	X	X	X	X	L	H
7	L	L	H	L	X	X	H	X	X	X	X	X	H	L
8	L	L	H	H	X	X	X	L	X	X	X	X	L	H
9	L	L	H	H	X	X	X	H	X	X	X	X	H	L
10	L	H	L	L	X	X	X	X	L	X	X	X	L	H
11	L	H	L	L	X	X	X	X	H	X	X	X	H	L
12	L	H	L	H	X	X	X	X	X	L	X	X	L	H
13	L	H	L	H	X	X	X	X	X	H	X	X	H	L
14	L	H	H	L	X	X	X	X	X	X	L	X	L	H
15	L	H	H	H	X	X	X	X	X	X	H	X	H	L
16	L	H	H	H	X	X	X	X	X	X	X	L	L	H
17	L	H	H	H	X	X	X	X	X	X	X	H	H	L

Схема електричну-принципову підключення мультиплексора на мікросхемі 74HC151 наведено на рис. 63. Аналогічно до попередніх схем, сигнал з плюса джерела живлення на контакт живлення мікросхеми заводиться через дільник напруги, оскільки максимально допустима напруга для даного мультиплексора становить 6В.

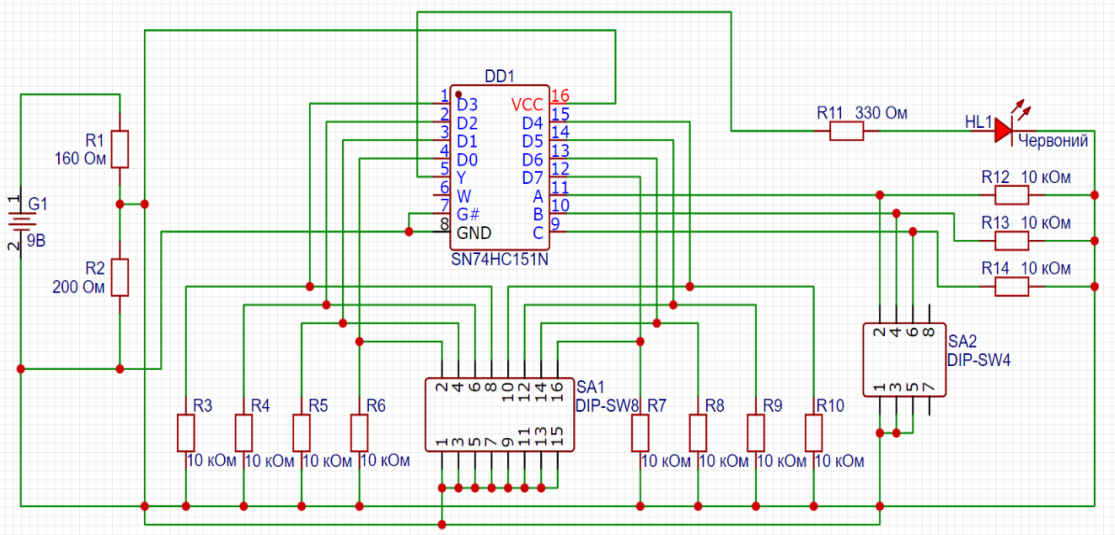


Рисунок 63. – Схема підключення мікросхеми 74HC151 (мультиплексора)

У схемі підключення мікросхеми **74HC151** для перевірки принципу її роботи передбачити наступні дискретні компоненти:

- мікросхема DD1 типу 74HC151 (**SN74HC151N**)
- R1 резистор ємністю 160 Ом;
- R2 резистор ємністю 200 Ом;
- R3-R10, R12-R14 резистори ємністю 10 кОм;
- R11 резистор ємністю 330 Ом;
- HL1 червоний світлодіод;
- SA1 DIP-перемикач SW4;
- SA2 DIP-перемикач SW8;
- батарея живлення G1 9В.

На інформаційні входи сигнали подаються за допомогою DIP-перемикача **SA1**: вимкнене положення відповідає логічному «0», увімкне положення - логічна «**1**». Інформаційні входи **I0-I7** заведені не тільки на контакти DIP-перемикача, але й через резистори до мінуса джерела живлення. Таке підключення забезпечує наявність стабільного логічного «**0**» при виключених контактах тумблера. Адресні входи заводяться на інший DIP-перемикач SA2. До прямого виходу Y підключено світлодіод HL1. Зібрати схему мультиплексора на макетній платі можна відповідно до рис. 64.

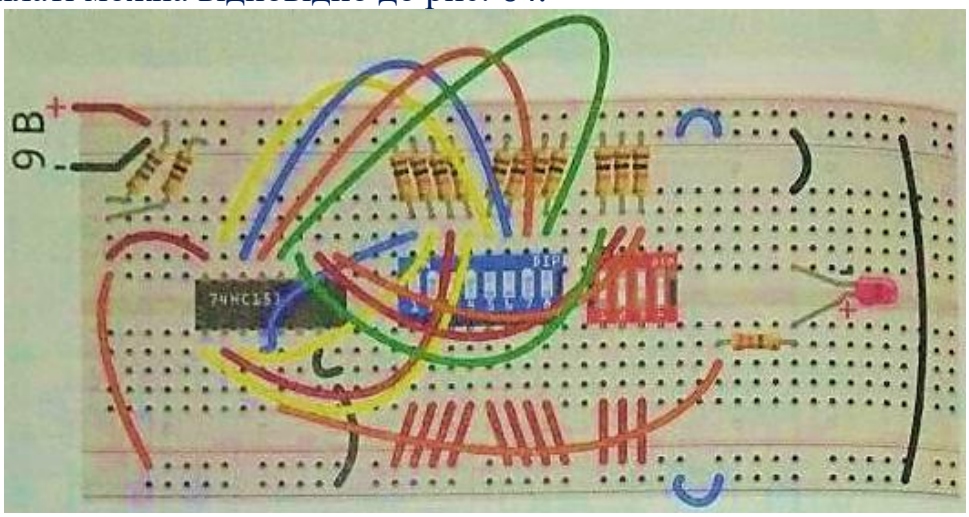


Рисунок 64. - Схема підключення мультиплексора.

На адресному DIP-перемикачі **SA2** необхідно встановити перші три тумблери в **0** (вимкнене положення). Таким чином встановлено нульову адресу на вході. Почергово, вмикаючи тумблери №1-8 інформаційного DIP-перемикача **SA1** можна спостерігати за світлодіодом **HL1**.

Якщо включити тумблер №1, який підключено до входу **I0**, означає що на схемі задано нульову адресу, і на вихід мікросхеми підключився нульовий вхід мультимплексора. Якщо встановити тумблер №2 адресного DIP-перемикача **SA2** в положення **ON** (увімкнений), а тумблери №1 і №3 залишити вимкненими, то на мультиплексорі задана другу адресу, що відповідає другому інформаційному входу **I2**, який підключено до виходу. Перемикаючи тумблер №3 інформаційного DIP-перемикача SA1 в різні положення, можна спостерігати за поведінкою світлодіоду. Коли на другий інформаційний вхід подається логічна «**1**» (тумблер №3 включений), на виході **Y** також з'являється логічна «**1**», тоді світлодіод спрацьовує. І навпаки, коли на вході **I2** логічний «**0**», то світлодіод

HL1 не світиться. Можна перевірити інші комбінації на адресних входах мультиплексора та записати їх результати до табл. 28.

Таблиця 28. – Результати дослідження роботи мультиплексора.

Адресний вхід	**S2**	**S1**	**S0**	***Відповідне десяткове число***	***Номер виходу дешифратора***
№ тумблера DIP-перемикача SA2	**3**	**2**	**1**		
Двійкова комбінація на вході мультиплексора	0	0	0	0	I0
	0	0	1	…	…
	0	1	0	…	I2
	0	1	1	…	…
	1	0	0	…	…
	1	0	1	…	…
	1	1	0	…	…
	1	1	1	…	…

На адресні (керуючі) входи **S0-S2** подано двійковий код **100**, що відповідає десятковому числу 4. На інформаційному вході **I4** присутня логічна «**1**», яка повинна передатись на вихід мультиплексора.

Сигнали від інформаційних входів через повторювачі (буфери, які на схемі позначено трикутником) заходять на логічні елементи «**5I**» (мають 5-ть входів і 1-н вихід). Туди ж спрямовуються сигнали від адресних входів, які пройшли через **1** або **2** інвертори, а також на елементи «**5I**» надходить інвертований сигнал від входу дозволу роботи E. Лише на одному елементі «**5I**», який підключено до інформаційного входу **I4**.на всіх інших елементах «**5I**» на виході буде логічний «0», оскільки на їх входах присутній хоча б один нуль.

Виходи елементів «5I» заходять на вхід елементу «**8АБО-НІ**», який має 8 входів та 1 інвертований вихід. Звичайний елемент «**АБО**» видає на своєму виході логічну «**1**». Якщо на одному з входів присутня «**1**», то елемент «**АБО-НІ**» в такому випадку видає на виході «**0**». Так і в цьому випадку, коли на вході «**8АБО-НІ**» присутня лише одна логічна «1», то на виході буде логічний «**0**». Це значення подається на інверсний вихід мультиплексора, а на прямий вихід Y подається ще один інвертор, що призводить до появи на прямому виході мультиплексора логічної «**1**». Цей сигнал поступає із інформаційного входу I4, номер якого задали адресні входи. Якщо подати на якийсь інший вхід одиницю, а на **I4** – логічний «**0**», то адреса залишається на *прямому* виході **Y** - логічний «**0**».

Комбінаційний пристрій демультиплексор. Демультиплексор - цифровий прилад, що має один інформаційний m керуючих входів та 2^m виходів. Він виконує передачу сигналу з вхідного каналу на один із вихідних, номер якого задається керуючими входами.

В якості демультиплексора використовується мікросхема 74-ї серії **74HC238** (рис. 65), яка використовувалась при побудові дешифратора. Окремої мікросхеми демультиплексора не існує, тому використовуючи інші режими роботи даної мікросхеми можна створити модель роботи демультиплесора.

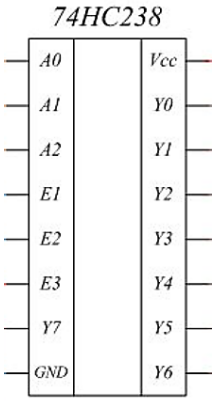


Рисунок 65. - Схематичне позначення мікросхеми 74HC238.

Адресними входами демультиплексора будуть контакти **A0-A2**, а інформаційним входом контакт **E3**. В залежності від комбінації на адресних контактах, на один із виходів мікросхеми 74HC238 передається сигнал, який присутній на контакті **E3**. І якщо у випадку дешифратора на відповідний до набраного двійкового коду вихід подавався тільки високий рівень сигналу, то зараз на відповідному виході можна буде спостерігати як логічний нуль так, логічну одиницю. Подаючи на вхід E3 сигнал з'являється аналоговий цифровий сигнал. Схему підключення мікросхеми **74HC238** в якості демультиплексора наведено на рис. 66.

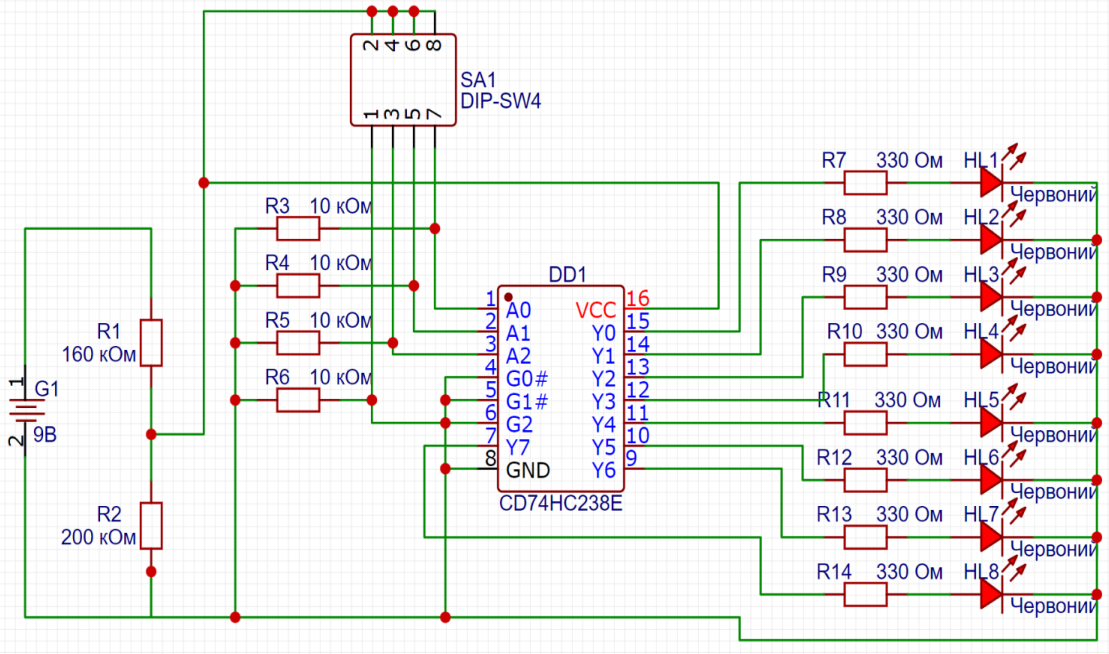


Рисунок 66. – Схема підключення мікросхеми 74HC238 (демультиплексора)

У схемі підключення мікросхеми 74HC238 для перевірки принципу її роботи передбачити наступні дискретні компоненти:

- мікросхема DD1 типу 74HC238 (CD74HC238E)
- R1 резистор ємністю 160 Ом;
- R2 резистор ємністю 200 Ом;
- R3-R6 резистори ємністю 10 кОм;
- R7-R14 резистори ємністю 330 Ом;
- HL1- HL8 червоні світлодіоди;

- SA1 DIP-перемикач SW4;
- батарея живлення G1 9В.

Дана схема практично ідентична до схеми електричної принципової дешифратора (рис. 61), проте на відміну від дешифратора, вхід **E3** використовується в якості інформаційного входу і підключений до тумблера №4. При встановленні на входах **A0-A2** адреси в двійковому вході сигнал з входу **E3** буде передаватись на вихід, номер якого дорівнює встановленій адресі в десятковій системі числення.

Задавати вхідний сигнал можна за допомогою тумблера №4 DIP-перемикача **SA1**. Якщо встановити мікросхему 74HC238 на макетну плату (рис. 67), можна за допомогою тумблерів №1,2,3 DIP-перемикача встановити будь-яку адресу. Зібрати схему демультиплексора на макетній платі на основі мікросхеми 74HC238 відповідно до рис. 12 та перевірити режими його функціонування.

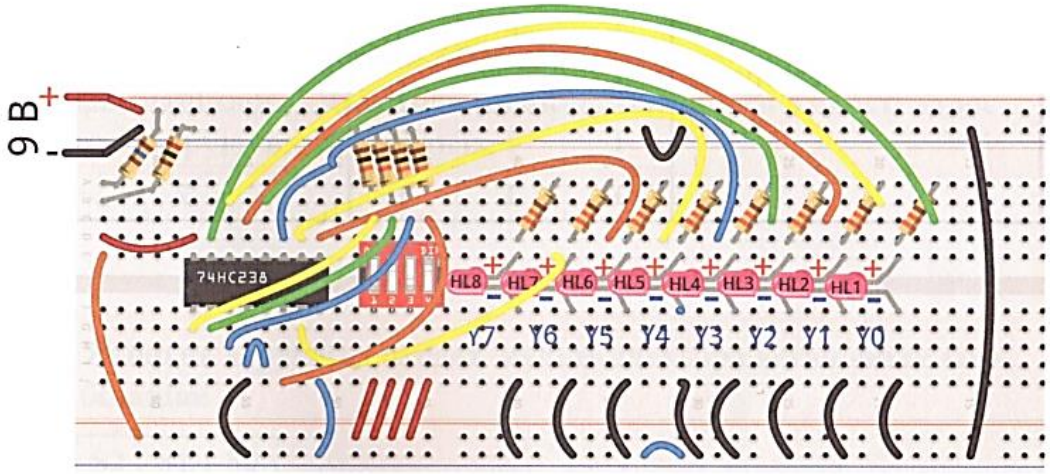


Рисунок 67. - Схема підключення демультиплексора.

Перемикаючи різні положення тумблера №4, можна подавати сигнал на інформаційний вхід **E3**. При цьому, світлодіоди, які підключені до вихідних контактів можуть почергово засвічуватись та гаснути. Самостійно з'ясуйте до якого вихідного контакту мікросхеми підключено цей вихідний контакт?

Користуючись табл. 29, можна встановити на адресних входах всі можливі комбінації. Самостійно можна перевірити перевірте результати роботи дешифратора та записати його результати у таблицю 29.

Таблиця 29. – Результати дослідження роботи демультиплексора.

Вхідний контакт	**A2**	**A1**	**A0**	***Номер підключеного вихідного контакту***
№ тумблера	**3**	**2**	**1**	
Двійкова комбінація на вході дешифратора	**0**	**0**	**0**	**0**
	0	**0**	**1**	…
	0	**1**	**0**	…
	0	**1**	**1**	…
	1	**0**	**0**	…
	1	**0**	**1**	…
	1	**1**	**0**	…
	1	**1**	**1**	…

Після проектування електричної схеми в ***EasyEDA*** створимо макет печатної плати (рис. 68) та переглянемо її у форматі **3D** (рис. 69). Як видно з наведених рисунків, на платі треки подані з двох сторін (червоним кольором показані треки на верхній стороні плати, синім – на нижній).

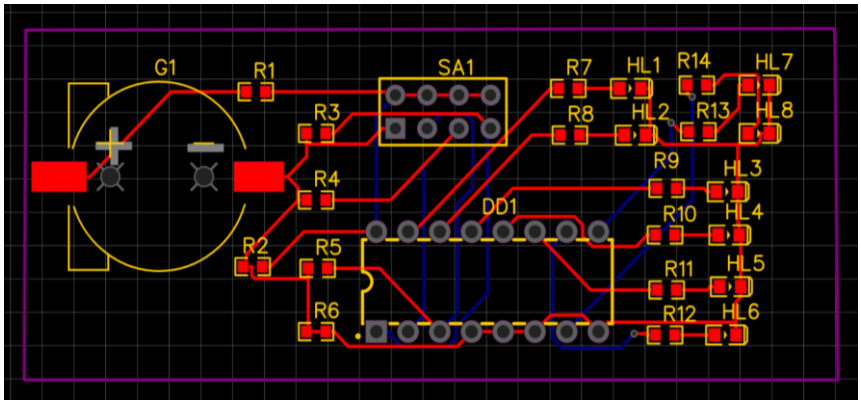


Рисунок 68. – Макет печатної плати демультиплексора.

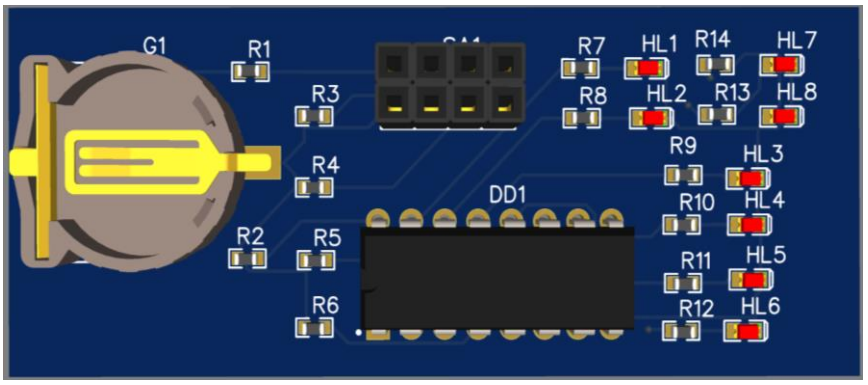


Рисунок 69. - Схема підключення демультиплексора.

Контрольні запитання

15.На які групи поділяються цифрові пристрої? Яка між ними різниця?
16.Яку функцію виконує шифратор?
17.Яку функцію виконує дешифратор?
18.В чому відмінність між шифратором та дешифратором?
19.Яку функцію виконує мультиплексор?
20.Яку функцію виконує демультиплексор?
21.В чому відмінність між мультиплексором та демультиплексором?
22.Яку мікросхему можна використати в роботі шифратора?
23.Яку мікросхему можна використати в роботі дешифратора?
24.Яку мікросхему можна використати в роботі демультиплексора?
25. До якого типу цифрових пристроїв належить шифратор?
26. До якого типу цифрових пристроїв належить дешифратор?
27. До якого типу цифрових пристроїв належить мультиплексор?
28.Запишіть таблицю істинності шифратора 4-х розрядного шифратора?
29. Запишіть таблицю істинності шифратора 3-х розрядного дешифратора?
30. Який вхід мікросхеми 74HC238 вважається інформаційним?
31. Які входи мікросхеми 74HC238 вважаються адресними (керуючими)?
32. Дайте визначення комбінаційних пристроїв.

Практичне завдання 5.
Машинні коди. Виконання арифметичних операцій з цілими та дійсними двійковими числами.

Мета роботи: представлення двійкових чисел в прямому, зворотному та додатковому кодах; застосування машинних кодів для виконання операцій додавання, віднімання, множення, ділення цілих та дійсних двійкових чисел; виконання операцій з прямими (ПК) та додатковими (ДК) двійковими кодами; реалізація лінійних та циклічних зсувів; використання стану ознаки CF, яка міститься в регістрах ознак процесора; виконання операції ділення з використанням двійкових кодів за відповідними схемами без відновлення та з відновленням залишку; ознайомлення з принципами роботи комбінаційних цифрових суматорів**.**

Завдання 1. Відповідно до варіанту **(**табл. 30) подати значення **X** та **Y** машинними кодами: десятковим, двійковим, прямим, зворотним та додатковим для **+X, –X, +Y, –Y**. В табл. 30 значення X подано в додатковому коді та Y – в зворотному коді. При розрахунку, значення X та Y прийняти довжиною в 1 байт. Результати виконання записати в табл. 31 відповідно до наведеного зразка.

Таблиця 30. – Індивідуальні завдання[1]

Варіант	*X (Додатковий код)*	*Y (Зворотний код)*	*Варіант*	*X (Додатковий код)*	*Y (Зворотний код)*
1	1,110111	0,110111	16	0,1101010	1,110110
2	1,1010111	0,10111	17	0,000111	1,1110001
3	1,1110110	0,101011	18	1,11111	0,110000
4	0,101011	1,1011001	19	0,1001011	1,100110
5	0,1111101	1,10111	20	0,01011	1,1011011
6	0,100111	1,0111101	21	1,1011001	0,1100100
7	1,1001111	0,100010	22	0,11011	1,1001011
8	0,1001100	1,1101101	23	1,0111100	0,1101
9	1,0111	0,100010	24	0,101110	1,1010111
10	0,10111	1,101110	25	0,1101101	1,110
11	0,1101011	1,100011	26	1,1111	0,1011101
12	1,100011	0,111001	27	0,1001011	1,111101
13	1,01110	0,111010	28	0,111011	1,100111
14	0,110001	1,0110110	29	1,111	0,101111
15	1,10111	0,10011	30	0,1011	1,1100000

**** в таблиці червоним кольором наведена знакова позиція відповідного числа.***

Виконати операції додавання в додатковому та зворотному кодах двійкових чисел X і Y, використовуючи різні комбінації знаків цих чисел, а саме (+X,+Y), (+X,–Y), (–X,–Y) і (–X,+Y). Результати додавання записати в табл. 32, відповідно до наведеного зразка.

[1] *Варіант вибирається у відповідності з порядковим номером студента в списку академічної групи*

Таблиця 31. – Приклад отриманих машинних кодів, відповідно до варіанту №30

Коди	*+X*	*–X*	*+Y*	*–Y*
Десятковий	+11	–11	+31	–31
Двійковий	+1011	–1011	+11111	–11111
Прямий	0,0001011	1,0001011	0,0011111	1,0011111
Зворотний	0,0001011	1,1110100	0,0011111	**1,1100000**
Додатковий	**0,0001011**	1,1110101	0,0011111	1,1100001

Таблиця 32. – Приклад отриманих результатів, відповідно до варіанту №30

Зображення заданих чисел в різних кодах	*Додавання при різних комбінаціях знаків*		*Результат в заданому коді*	*Результат в прямому коді*	*Результат в десятковому коді*
X в десятковому коді (+11)	В додатковому коді	*+X + (–Y)*	1,1101100	1,0010100	–20
Y в десятковому коді (–31)		*+X + (+Y)*	0,0101010	0,0101010	+42
X в прямому коді (0,0001011)		*–X + (–Y)*	1,1010110	1,0101010	–42
Y в прямому коді (1,0011111)		*–X + (+Y)*	0,0010100	0,0010100	+20
X в зворотному коді (0,0001011)	В зворотному коді	*+X + (–Y)*	1,1101011		
Y в зворотному коді (**1,1100000**)		*+X + (+Y)*	0,0101010		
X в додатковому коді (**0,0001011**)		*–X + (–Y)*	1,1010101		
Y в додатковому коді (1,1100001)		*–X + (+Y)*	0,0010100		

Завдання 2. Відповідно до варіанту (табл. 33), виконати переведення вихідних чисел A_{10} та B_{10}, поданих в десятковій системі числення, в прямий код двійкової системи числення. Точність двійкових чисел A_2 і B_2 – дев'ять знаків після коми.

Таблиця 33. – Вихідні дані для виконання лабораторної роботи[1]

Варіант	A_{10}	B_{10}	Варіант	A_{10}	B_{10}
1	16.53	–36.29	16	36.63	–57.75
2	23.47	53.67	17	–59.97	37.73
3	–65.38	15.74	18	38.12	68.59
4	42.89	33.52	19	–71.17	39.93
5	–76.74	17.44	20	40.14	–56.65
6	–32.19	–80.88	21	–72.27	43.36
7	99.57	–15.97	22	44.51	–69.96
8	51.68	28.79	23	63.85	46.64
9	–35.49	77.39	24	–48.84	–18.81
10	64.25	–19.43	25	61.16	50.15
11	–47.31	29.34	26	–52.29	28.82
12	20.52	45.78	27	26.63	54.47
13	30.32	–55.56	28	58.85	–24.42
14	67.76	31.41	29	22.72	60.76
15	34.54	–87.77	30	–70.73	21.12

Отримані двійкові числа A_2 і B_2 представити в форматі з плаваючою комою в прямому коді – **A[пк]** та **B[пк]**. Вирівняти порядки і отримати зворотний (ЗК) і додатковий (ДК) коди –відповідно **A[зк]**, **B[зк]**, **A[дк]** та **B[дк].** Прямий, зворотний і додатковий коди отримати для $+A_{10}$ і $-A_{10}$, а також для $+B_{10}$ і $-B_{10}$.

[1] *Варіант вибирається у відповідності з порядковим номером студента в списку академічної групи*

Виконати додавання (віднімання) мантис двійкових чисел ***A*[дк]** і ***B*[дк]** в додатковому модифікованому коді. Додавання виконати для різних комбінацій знаків чисел ***A*[дк]** і ***B*[дк]**, а саме для (+,+), (+,–), (–,–) і (–,+).

Виконавши нормалізацію мантиси результату додавання, коли в цьому є необхідність, відтворити результат додавання в ДК, ЗК і ПК, а також в двійковому з фіксованою комою і десятковому коді.

Завдання 3. Відповідно до варіанту (табл. 34), отримати цілі двійкові числа A_2 і B_2 у форматі з фіксованою комою в прямому (**ПК**) і додатковому (**ДК**) кодах. При цьому використати різні знаки у чисел: $+A_2$, $+B_2$, $-A_2$ і $-B_2$.

***Таблиця 34. – Варіанти до виконання завдання*[1]**

Варіант	Множене A_{10}	Множник B_{10}	Варіант	Множене A_{10}	Множник B_{10}
1	16	–36	16	3	–27
2	28	33	17	–5	17
3	–25	15	18	8	20
4	12	33	19	–4	19
5	–36	17	20	4	–26
6	–32	–10	21	–17	13
7	9	–15	22	14	–19
8	11	28	23	23	6
9	–35	27	24	–8	–5
10	4	–19	25	21	2
11	–7	29	26	–5	28
12	20	15	27	3	35
13	30	–5	28	9	–12
14	6	14	29	13	6
15	34	–8	30	–7	21

Виконати множення двійкових чисел A_2 і B_2. Використати при множенні додаткові коди і різні комбінації знаків: **($+A_2$,$+B_2$), ($+A_2$,$-B_2$), ($-A_2$,$+B_2$), ($-A_2$,$-B_2$).**

Завдання 4. Відповідно до варіанту (табл. 35), виконати ділення двох заданих пар чисел, використавши відповідно схему ділення без відновлення залишку і з відновленням залишку. Результат ділення записати в прямому двійковому коді та в десятковому коді.

***Таблиця 35. – Варіанти до виконання завдання*[2]**

№ варіанту	*Вихідні числа для виконання операції ділення*			
	Без відновлення залишку		*З відновленням залишку*	
	Ділене	*Дільник*	*Ділене*	*Дільник*
	A	***B***	***A***	***B***
0	23	7	33	10
1	24	5	34	11
2	25	7	35	8
3	26	8	36	10
4	27	6	37	9

[1] ***Варіант вибирається у відповідності з порядковим номером студента в списку академічної групи***

[2] ***Варіант вибирається у відповідності з порядковим номером студента в списку академічної групи +5***

№ варіанту	*Вихідні числа для виконання операції ділення*			
	Без відновлення залишку		*З відновленням залишку*	
	Ділене	*Дільник*	*Ділене*	*Дільник*
	A	***B***	***A***	***B***
5	28	6	38	8
6	29	8	39	9
7	30	9	40	12
8	31	8	41	13
9	32	7	42	12
10	17	4	27	8
11	18	5	28	9
12	19	6	29	8
13	20	6	30	9
14	21	6	31	8
15	22	5	32	9
16	23	8	17	5
17	24	7	18	4
18	25	8	19	5
19	26	6	20	5
20	27	8	21	5
21	28	9	22	6
22	29	7	43	14
23	30	8	44	13
24	31	7	25	6
25	32	6	26	7
26	33	9	27	7
27	34	8	12	3
28	35	9	13	4
29	36	8	14	4
30	37	9	19	4

Завдання 5. Використовуючи дискретні компоненти, створити в ***EasyEDA*** електричні схеми комбінаційних пристроїв відповідно до заданого варіанту (табл. 36).

Таблиця 36. - Варіанти для побудови цифрових пристроїв

№ вар.	*Цифровий пристрій*	*Кількість розрядів*	*№ вар.*	*Цифровий пристрій*	*Кількість розрядів*
1	напівсуматор	1	**6**	напівсуматор	2
	суматор	2		суматор	1
2	напівсуматор	2	**7**	напівсуматор	3
	суматор	1		суматор	4
3	напівсуматор	3	**8**	напівсуматор	4
	суматор	4		суматор	3
4	напівсуматор	4	**9**	напівсуматор	1
	суматор	3		суматор	2
5	напівсуматор	1	**10**	напівсуматор	2
	суматор	2		суматор	1

Побудувати печатні плати створених цифрових пристроїв в ***EasyEDA*** та зберегти їх макети в форматі ***2D*** та ***3D***.

Побудувати схеми цифрових пристроїв на макетних платах конструктора

«Цифрова електроніка» та/або в ***TinkerCad*** та записати відповідні таблиці істинності режимів їх роботи.

Зміст звіту

1. Титульна сторінка, тема та мета практичного заняття.
2. Опис виконаних завдань з наведеними рисунками, графіками, таблицями істинності, логічними виразами, структурними, логічними та електричними схемами створені в онлайн конструкторі ***EasyEDA***.
3. Побудувати печатні плати створених комбінаційних пристроїв та зберегти їх макети в 2D та 3D в онлайн конструкторі ***EasyEDA***.
4. Висновки, що відображають результати виконання роботи.

Теоретичні відомості

Правила додавання та віднімання цілих двійкових чисел. Основні правила додавання (віднімання) цілих чисел із знаком можна сформулювати наступним чином:

а). Операція віднімання замінюється операцією додавання з від'ємним числом.

б). Доданки повинні мати однакову кількість розрядів. Для вирівнювання розрядних сіток доданків треба дописувати нулі зліва до значущої частини числа.

в). Приписані незначущі нулі змінюють своє значення при перетвореннях кодів за загальними правилами.

г). Знакові розряди чисел враховуються при додаванні нарівні зі значущими розрядами.

д). При появі одиниці переносу із старшого знакового розряду, в разі використання зворотного коду, ця одиниця підсумовується зі значущою частиною числа. При використанні додаткового коду одиниця переносу не враховується.

е). Знак результату формується автоматично.

є). Результат представляється в тому коді, в якому представлені доданки.

Правила додавання двійкових цифр чисел ***A*** і ***B*** представлені в табл. 77. Тут показані правила додавання двійкових цифр a_i та b_i однойменних розрядів чисел A і B з урахуванням можливого переносу P_{i-1} із попереднього розряду.

Таблиця 37. – Правила додавання двійкових цифр

Значення i-х двійкових розрядів чисел A, B і P_{i-1}			***Розряд суми S_i***	***Перенос в наступний розряд Pi***
a_i	***b_i***	***P_{i-1}***		
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Слід мати на увазі, що операція додавання виконується в модифікованих зворотному та додатковому кодах. Операція віднімання замінюється операцією додавання з від'ємним числом. Надалі використовуються наступні умовні позначення:

- ***пк*** – прямий код;
- ***зк*** – зворотний код;
- ***мзк*** – модифікований зворотний код;
- ***дк*** – додатковий код;
- ***мдк*** – модифікований додатковий код.

При зображенні цілих чисел зі знаком використовуються один або два (для модифікованих кодів) старших двійкових розрядів для зображення коду позиційного знаку (0 або 00 – для додатного числа і 1 або 11 – для від'ємного числа).

Прямий двійковий код цілого числа із знаком відрізняється від прямого двійкового коду цілого числа без знаку використанням одного старшого двійкового розряду для зображення коду знаку (0 – для додатного числа і 1 – для від'ємного числа). Наприклад:

A = +1011 $[A]_{пк}$ = 0,0001011
B = –11111 $[B]_{пк}$ = 1,0011111

Зворотний код цілого додатного двійкового числа співпадає з його прямим кодом. Зворотний код цілого від'ємного числа можна отримати із прямого коду відповідного цілого додатного числа, якщо в знаковому розряді вказати 1, а решту цифр прямого коду змінити на протилежні. Якщо маємо зворотний код цілого від'ємного числа, то можемо отримати прямий код відповідного цілого додатного числа, якщо в знаковому розряді вказуємо 0, а решту цифр зворотного коду змінимо на протилежні.

В зворотному коді знаковий розряд і значуща частина числа вважаються єдиним цілим. Знак суми отримується автоматично під час додавання вмісту знакових розрядів доданків і одиниці переносу із значущої частини доданків, якщо одиниця переносу є.

Особливістю використання зворотного коду при виконанні операції додавання цілих двійкових чисел із знаком є те, що при появі одиниці переносу із знакового розряду має бути виконана корекція суми. Для здійснення корекції необхідно виконати операцію додавання одиниці переносу зі значущою частиною числа.

Приклад 1:

A = + 11101 (+29); B = – 01011 (–11)

0,11101 $[A]_{зк}$
+1,10100 $[B]_{зк}$
10,10001 –> 0,10010 –> + 10010 (+18)

Приклад 2:

A = – 000110 (–6); B = – 010111 (–23)

1,111001 $[A]_{зк}$
+ 1,101000 $[B]_{зк}$
11,100001 –> 1,100010 –> 1, 011101 –> – 011101 (–29)

При виконанні операції додавання з використанням зворотного коду отримуваний знак суми може не відповідати очікуваному, тобто замість знака “плюс” з’являється знак “мінус” або навпаки. Тому операцію додавання треба виконувати з використанням модифікованого зворотного коду, в якому під знак числа відводяться два двійкових розряди.

Приклад 3:

$A = -10011$ (–19) $B = -10001$ (–17)

$$\begin{array}{r} 11{,}01100 \quad [A]_{мзк} \\ +\ 11{,}01110 \quad [B]_{мзк} \\ \hline 110{,}11010 \end{array}$$

Зліва з’явилась одиниця переносу, тому має бути виконана корекція суми. Для здійснення корекції необхідно виконати операцію додавання одиниці переносу зі значущою частиною числа.

В результаті отримаємо:

$$110{,}11010 \rightarrow 10{,}11011$$

Відмінність цифр в знакових розрядах свідчить про те, що треба виконати модифікацію суми. Для цього отримане число разом із знаковими розрядами зрушують на один розряд вправо, а у звільненому лівому знаковому розряді дублюють значення отриманого після зсуву правого знакового розряду. В результаті отримаємо:

$$10{,}11011 \rightarrow 11{,}011011$$

Отже, загальний результат:

$$11{,}011011 \rightarrow 1{,}100100 \rightarrow -100100 \quad (-36)$$

Додатковий код цілого додатного двійкового числа співпадає з його прямим кодом. Додатковий код цілого від’ємного числа можна отримати із прямого коду відповідного цілого додатного числа, якщо в знаковому розряді вказати 1, а решту цифр прямого коду змінити на протилежні, а потім виконати операцію додавання отриманого коду з одиницею.

Якщо маємо додатковий код цілого від’ємного числа, то можемо отримати прямий код відповідного цілого додатного числа, якщо виконаємо вказані вище дії, але в знаковий розряд впишемо 0.

В додатковому коді знаковий розряд і значуща частина числа вважаються єдиним цілим. Знак суми отримується автоматично під час додавання вмісту знакових розрядів доданків і одиниці переносу із значущої частини доданків, якщо одиниця переносу є.

Приклад 4:

$[A]_{пк} = 1{,}110100$

$[A]_{дк} = 1{,}001011 + 0{,}000001 = 1{,}001100$

$[A]_{дк} = 1{,}001100 \rightarrow 0{,}110011 + 0{,}000001 = 0{,}110100$

Особливістю використання додаткового коду при виконанні операції додавання цілих двійкових чисел із знаком є те, що поява одиниці переносу із знакового розряду не враховується. При виконанні операції додавання з використанням додаткового коду отримуваний знак суми може не відповідати очікуваному, тобто замість знаку “плюс” з’являється знак “мінус” або навпаки, як показано в прикладі 5. Тому операцію додавання треба виконувати з

використанням модифікованого додаткового коду, в якому під знак числа відводяться два двійкових розряди, причому для зображення знаку “плюс” використовується код 00, а для зображення знаку “мінус”– код 11.

Приклад 5:

A = +10101 B = +1110 D = –10101 E = –01110

$[A]_{дк}$ = 0,10101	$[D]_{дк}$ = 1,01011
+$[B]_{дк}$ = 0,01110	+$[E]_{дк}$ = 1,10010
$[C]_{дк}$ = 1,00011	$[F]_{дк}$ = 10,11101 –> 0,11101

При виконанні операції додавання з використанням додаткового коду отримуваний знак суми може не відповідати очікуваному, тобто замість знаку “плюс” з’являється знак “мінус” або навпаки, як показано в прикладі 5. Тому операцію додавання треба виконувати з використанням модифікованого додаткового коду, в якому під знак числа відводяться два двійкових розряди, причому для зображення знаку “плюс” використовується код 00, а для зображення знаку “мінус”– код 11.

Відмінність цифр в знакових розрядах, а саме якщо з’являться коди 01 або 10, свідчить про те, що треба виконати модифікацію суми. Для цього отримане число разом із знаковими розрядами зрушують на один розряд вправо, а у звільненому лівому знаковому розряді дублюють значення отриманого після зсуву правого знакового розряду.

Приклад 6:

A = +10010 B = +11001

$[A]_{мдк}$ = 00,10010
+$[B]_{мдк}$ = 00,11001
$[C]_{мдк}$ = 01,01011 –> 00,101011

Перевірка: 10010_2 –> 18_{10} 11001_2 –> 25_{10} 101011_2 –> 43_{10}

Приклад 7:

A = –01011 B = –01101

$[A]_{мдк}$ = 11,10101
+$[B]_{мдк}$ = 11,10011
$[C]_{мдк}$ = 11,01000 –> 0,10111 + 0,00001 = 0,11000 –> C = –11000

Перевірка:

1011_2 –> 11_{10} 1101_2 –> 13_{10} 11000_2 –> 24_{10}

Приклад 8:

+101001 (41) + (+110100 (52)) = +1011101 (93)
–11001 (–25) + (+01001 (9)) = 11,00111 + 00,01001 =
= 11,10000 –> 0,01111 + 0,00001 –> –10000 (–16)
–00110 (–6) + (+10100 (20)) = 11,11010 + 00,10100 = 100,01110 –> 00,01110 (14)

Відображення чисел у форматі з плаваючою комою. У чисел з плаваючою комою (точкою) відділяє цілу частину числа від його дробової частини. Числа в форматі з плаваючою комою в оперативній пам’яті у полі

заданої довжини часто містять ***знак порядку***, ***порядок***, ***знак мантиси*** і ***мантису*** (рис. 70).

Рисунок 70. – Один із форматів зображення чисел з плаваючою комою

Для зображення знаків порядку і мантиси резервують по одному біту: 0 – ознака додатного значення; 1 – ознака від'ємного значення. Знак числа співпадає зі знаком мантиси.

Порядок – це число, що відтворює степінь, в яку треба піднести число 2. На число "2 в степені "порядок" треба помножити мантису щоб отримати значення двійкового числа з плаваючою комою. Таким чином, порядок визначає місцезнаходження точки в двійковому числі в форматі з фіксованою комою.

Мантиса містить як цілу, так і дробову частини числа, що було зображене в форматі з фіксованою комою, але при умові, що двійкова кома перенесена з метою отримання нормалізованого значення мантиси. В мантисі кількість двійкових розрядів, які займає ціла частина числа з фіксованою комою, визначається значенням порядку, а решта розрядів містять дробову частину числа з фіксованою комою.

Для отримання нормалізованого числа з плаваючою комою порядок необхідно змінити таким чином, щоб лівий (старший) біт мантиси був не 0. Наприклад, замість мантиси 00001011 ... 11 треба використати мантису 1011 ... 110000, але при цьому порядок треба зменшити на чотири.

Приклад 9:

$A_{10} = -20.125$

$A_2 = -10100.0001 = -0.101000001*2^{+5}$

В форматі з плаваючою комою: мантиса $M_2 = 101000001_2$; порядок $P_{10} = 5_{10}$, $P_2 = 101_2$. Таким чином, для одного із варіантів зображення двійкового числа в форматі з плаваючою комою (наприклад, в 4-байтовому полі) будемо мати наступний формат, який наведено на рис. 71.

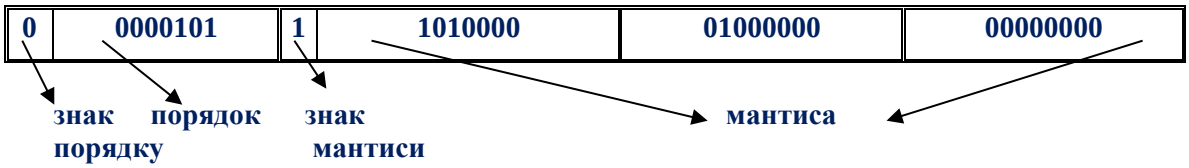


Рисунок 71. – Формат числа з плаваючою комою

Правила додавання (віднімання) чисел у форматі з плаваючою комою. Основні правила додавання (віднімання) чисел у форматі з плаваючою комою можна сформулювати наступним чином:

1) Треба виконати операцію віднімання над значеннями порядків вихідних чисел *A*[пк] і *B*[пк]. Якщо різниця порядків не дорівнює нулю, мають бути виконані вирівнювання порядків і корекція відповідної мантиси.

3) При вирівнюванні порядків вміст розрядів мантиси числа з меншим порядком треба зрушити вправо на різницю порядків. При цьому молодші

розряди мантиси числа з меншим порядком зникають, в її старші розряди вписуються нулі, а порядок даного числа збільшується на отриману різницю порядків.

4) Треба вирівняти розрядну сітку мантиси того числа, у якого менша кількість розрядів в мантисі – в мантису цього числа мають бути дописані нулі справа.

5) Виконати операцію додавання мантис (операція віднімання замінюється операцією додавання з від'ємним числом). Дії над доданками (мантисами) виконуються в зворотному (ЗК) або додатковому (ДК) модифікованому коді.

6) Порядок результату береться рівним більшому порядку.

7) Якщо мантиса результату не нормалізована, то виконуються нормалізація мантиси і корекція значення порядку.

Нормалізація мантиси результату виконується в двох випадках:

а) цифри в знакових розрядах мантиси *не співпадають*;

б) цифри в знакових розрядах однакові і співпадають з цифрою старшого розряду мантиси.

Якщо цифри в знакових розрядах мантиси не співпадають, то *мантиса зрушується* на один розряд *вправо,* а *порядок збільшується* на одиницю.

Якщо цифри знакових розрядів мантиси однакові і співпадають з цифрою старшого розряду мантиси, то *мантиса зрушується* на один розряд *вліво*, а *порядок зменшується* на одиницю.

8). Врешті решт результат, отриманий при виконанні операції додавання, треба відобразити в прямому коді, виконавши округлення мантиси результату, а потім – в десятковому коді.

Приклад 10:

$A_{10} = 6{,}25,\ B_{10} = 20{,}5$

1. Записуємо числа в двійковому коді:

$A_2 = 110{,}01 = 0.11001*2^{+3}$

$B_2 = 10100{,}1 = 0.101001*2^{+5}$

2. Записуємо числа в форматі з плаваючою комою (див. п.4.1):

$[A]_{пк} = 0\ \ 011\ \ 0\ \ 11001$

$[B]_{пк} = 0\ \ 101\ \ 0\ \ 101001$

3. Порівнюємо порядки: $P_B - P_A = 5 - 3 = 2$.

4. Різниця порядків не дорівнює нулю, тому виконуємо вирівнювання порядку числа *A* і відповідну корекцію мантиси числа *A* (порядок числа *A* менше порядку числа *B* на 2, тому робимо зрушення мантиси числа *A* на два розряди вправо і збільшуємо порядок числа *A* на два):

$[A]_{пк} = 0\ \ 101\ \ 0\ \ 0011001$

$[B]_{пк} = 0\ \ 101\ \ 0\ \ 101001$

5. Вирівнюємо кількість двійкових розрядів в мантисах чисел *A* і *B* (розрядна сітка мантиси числа *B* на одиницю менша, тому в мантису числа *B* дописуємо один нуль справа):

$[A]_{пк} = 0\ \ 101\ \ 0\ \ 0011001$

$[B]_{пк} = 0\ \ 101\ \ 0\ \ 1010010$

6. Виконуємо операцію додавання мантис в додатковому модифікованому коді (знаки мантис відображаємо двома двійковими розрядами):

$[M_A]_{дк}$ = 00 0011001
$+[M_B]_{дк}$ = 00 1010010
$[M_C]_{дк}$ = 00 1101011

Мантиса результату додавання не потребує нормалізації, тому що розряди знаку мають однакові значення і значення старшого розряду мантиси не дорівнює значенню молодшого розряду знаку.

7. Записуємо результат у форматі з плаваючою комою: **0 101 0 1101011**

8. Переводимо результат в двійковий і десятковий коди

$$C_2 = \left(\frac{1}{2}+\frac{1}{4}+\frac{1}{16}+\frac{1}{64}+\frac{1}{128}\right)\cdot 2^5 = \frac{64+32+8+2+1}{128}\cdot 32 = \frac{107}{128}\cdot 32 = \frac{107}{4} = 26{,}75$$

C_2 = 11010,11 C_{10} = 16+8+2+0,5+0,25 = 26,75

Слід мати на увазі, що часто через невелику кількість двійкових розрядів, виділених для зберігання мантиси, виникає похибка відтворення дробової частини вихідного числа в двійковій системі числення. Значення абсолютної похибки Δ відтворення числа X обчислюється по формулі:

$$\Delta = X_{вихідне} - X_{отримане},$$

де, ***Хвихідне*** – вихідне число,
Хотримане – отримане число.

Значення відносної похибки δ відтворення числа X обчислюється по формулі:

$$\delta = (\Delta / X_{вихідне})*100\% .$$

Правила множення цілих двійкових чисел. Множення цілих двійкових чисел зі знаком в прямому і додатковому коді виконується шляхом покрокового додавання значення множеного, яке зсувається, та часткової суми, якщо у відповідному розряді множника присутня одиниця.

При множенні чисел в ***прямому коді*** знакові та значущі розряди обробляються нарізно. Для визначення знаку результату виконується додавання цифр, записаних в знакових розрядах операндів.

Множення цілих двійкових чисел в ***додатковому коді*** виконується по алгоритму, аналогічному множенню чисел в прямому коді, але знак результату отримується автоматично, при чому в операції множення приймають участь не тільки значущі, але і знакові розряди чисел. При цьому слід мати на увазі дві особливості. Якщо виконується множення множеного A на знакову одиницю від'ємного множника B, то необхідно для відповідного додавання використати не A, а $[-A]_{дк}$. Якщо ж від'ємним числом являється множене, то в кожному додаванні приймають участь одиниці, які отримуються внаслідок дописування зліва перед множеним незначущих нулів і наступної інверсії множеного.

Приклад 11.

$A = +10_{10} = 0\ 1010_2$ (множене); $[A]_{дк}$ = 0 1010
$B = -13_{10} = 1\ 1101_2$ (множник); $[B]_{дк}$ = 1 0011

01010
10011
01010
01010
00000
00000
1 0110
1 01111110

Під час множення на знакову одиницю множника *B* була виконана корекція – в якості останнього доданка використано $[-A]_{дк} = 1\ 0110$.

Оскільки в знаковому розряді результату маємо 1, то це означає, що результатом множення є двійкове число в додатковому коді. Для переходу до прямого коду результат потрібно інвертувати і виконати операцію додавання над інвертованим кодом і одиницею:

1 10000001+1= 1 10000010.

Результат:

$D = 1\ 10000010_2 \rightarrow -(1*2^7+0*2^6+0*2^5+0*2^4+0*2^3+0*2^2+1*2^1+0*2^0) = -(128+2) = -130_{10}$.

Приклад 12.

$A = -10_{10}$ (множене); $[A]_{дк} = 1\ 0110$
$B = +13_{10}$ (множник); $[B]_{дк} = 0\ 1101$

1 0110
0 1101
111110110
00000000
1110110
110110
00000
101111110

Оскільки множене являється від'ємним числом, то при додаванні використані одиниці множеного, які отримані шляхом інверсії незначущих нулів в старших розрядах, при наявності ненульового розряду множника.

В знаковому розряді добутку маємо 1, отже результат множення отримали в додатковому коді. Для переходу до прямого коду результат потрібно інвертувати і виконати операцію додавання над інвертованим кодом і одиницею:

1 10000001+1= 1 10000010.

Результат:

$D = 1\ 10000010_2 = -(1*2^7+0*2^6+0*2^5+0*2^4+0*2^3+0*2^2+1*2^1+0*2^0) = -(128+2) = -130_{10}$

Машинний алгоритм характеризується тим, що для множеного, множника і часткової суми використовуються регістри однакової довжини (наприклад, 8-бітові). Результат (добуток) зберігається в парі регістрів, в одному з яких зберігалась часткова сума, а в другому – множник. При цьому старша частина результату розміщується в регістрі часткової суми, а молодша – в регістрі множника. В даному алгоритмі виконуються наступні дії:

1. Зсув множника на 1 розряд в сторону молодших розрядів з урахуванням в старшому розряді поточного вмісту ознаки CF, тобто виконання операції *RCR*.
2. Якщо значення, висунуте з правого розряду і відтворене в ознаці CF, не дорівнює 1, то перехід до 4-го пункту;
3. Додавання множеного з частковою сумою. Присвоєння ознаці CF значення 0.
4. Зсув часткової суми на 1 розряд в сторону молодших розрядів з урахуванням в старшому розряді поточного вмісту ознаки CF, тобто виконання операції *RCR*.
5. Якщо кількість зсувів не дорівнює кількості розрядів *n* у множника, то знову перейти до пункту 1.
6. Корекція значення молодшої частини результату множення шляхом зсуву множника на 1 розряд в сторону молодших розрядів з урахуванням в старшому розряді поточного вмісту ознаки CF, тобто виконання операції *RCR*.

Правила ділення цілих двійкових чисел. Операцію ділення ***без відновлення залишку*** потрібно виконувати за наступною схемою:

1) Вирівняти кількість розрядів у двійковому зображенні діленого *A* і дільника *B* (для цього треба дописати незначущі нулі зліва в двійковому числі з меншою кількістю розрядів).

2) Отримати додаткові коди для чисел +*A*2, –*A*2, +*B*2 і –*B*2.

3) Цілій частині частки *C* присвоїти початкове значення, рівне 0.

3) Виконати [A]дк: = [A]дк + [–B]дк

4) Якщо отримане значення [A]дк > 0, то збільшити на одиницю значення цілої частини частки C и перейти до пункту 3. Інакше перейти до пункту 5.

5) Якщо отримане значення [A]дк = 0, то збільшити на одиницю значення цілої частини частки C, закінчити формування частки, зафіксувавши положення коми, яка відділяє цілу частину частки від її дробової частини, що дорівнює 0 в цьому випадку, і перейти до пункту “кінець”. Інакше перейти до пункту 6.

6) Якщо отримане значення [A]дк < 0, то зафіксувати положення коми, яка відділяє цілу частину частки від її дробової частини, і перейти до формування дробової частини частки, починаючи з пункту 7.

7) Виконати лінійний зсув [*A*]дк на один розряд вліво (SHL A,1).

8) Якщо маємо [–A]дк , то виконати [A]дк: = [–A]дк + [+B]дк. Якщо маємо [+A]дк , то виконати [A]дк **:=** [+A]дк + [–B]дк.

9) Якщо отримане в п.8 значення [A]дк > 0 або [A]дк = 0, то *i*-му розряду дробової частини присвоїти значення 1, інакше – 0.

10) Якщо число розрядів після коми менше необхідного і отримане [A]дк ще не 0, то перейти до пункту 7, інакше закінчити формування дробової частини і перейти до пункту “кінець”.

11) Кінець формування частки *C*.

Приклад 13.

A = 10; B = 3; C10очік = 3,33...; [A]дк = 00 1010; [B]дк = 00 0011; [–B]дк = 11 1101

C_2поч = 0

00 1010
11 1101
00 0111 > 0 –> inc (C) –> C_2 = 001

00 0111
11 1101
00 0100 > 0 –> inc (C) –> C_2 = 010

00 1000
11 1101
00 0001 > 0 –> inc (C) –> C_2 = 011

00 0001
11 1101
11 1110 < 0 –> “,”

після зсуву: 11 1100 –> 11 1100
00 0011
11 1111 < 0 –> C_2 = 011,0

після зсуву: 11 1110 –> 11 1110
00 0011
00 0001 > 0 –> C_2 = 011,01

після зсуву: 00 0010 –> 00 0010
11 1101
11 1111 < 0 –> C_2 = 011,010$_2$ ... –> C_{10} = 3,25$_{10}$ …

Операцію ділення ***з відновленням залишку*** необхідно виконати за наступною схемою.

Щоб отримати значення цілої частини частки C по схемі з відновленням залишку, треба виконати кроки 1-6, розглянуті в п.4.1 для схеми без відновлення залишку.

Формування дробової частини частки С, як і в схемі без відновлення залишку, треба почати з кроку 7. Отже:

7) Виконати відновлення залишку, а саме [А]дк: = [–А]дк + [+В]дк.

8) Виконати лінійний зсув [А]дк на один розряд вліво (SHL A,1).

9) Виконати [А]дк := [+А]дк + [–В]дк.

10) Якщо отримане в п.9 значення [А]дк > 0 або [А]дк = 0, то i-му розряду дробової частини присвоїти значення 1, інакше – 0.

11) Якщо число розрядів після коми менше необхідного і отримане в п.9 значення [А]дк < 0, то перейти до пункту 7, але якщо отримане [А]дк > 0, то перейти до пункту 8, і, нарешті, якщо отримане [А]дк = 0, то закінчити формування дробової частини і перейти до пункту “кінець”.

12) Кінець формування частки C.

Приклад 14.

A = 12; B = 5; C10очік = 2,4; [A]дк = 00 1100; [B]дк = 00 0101; [–B]дк = 11 1011

C_2поч = 0

00 1100
11 1011
00 0111 >0 inc (C) –> C_2 = 001

00 0111
11 1011
00 0010 >0 inc (C) –> C_2 = 010

00 0010
11 1011
11 1101 <0 –> “,”

відновлення залишку: 11 1101 + 00 0101 = 00 0010 > 0
після зсуву: 00 0100 –> 00 0100 + 11 1011 = 11 1111 < 0 –> C_2 = 010,0
відновлення залишку: 11 1111 + 00 0101 = 00 0100 > 0
після зсуву: 00 1000 –> 00 1000 + 11 1011 = 00 0011 > 0 –> C_2 = 010,01
після зсуву: 00 0110 –> 00 0110 + 11 1011 = 00 0001 > 0 –> C_2 = 010,011

Таким чином,

$C_2 = 010{,}011_2$ ... –> $C_{10} = 2{,}345_{10}$ …

Комбінаційні пристрої. Комбінаційними називаються приладами, виходи яких безпосередньо залежать від стану входів. До таких приладів відносяться:

- *суматори (напівсуматори);*
- *цифрові компаратори.*

Суматор - це комбінаційний цифровий прилад, призначений для додавання чисел, що представлені у вигляді двійкових кодів.

Існують повні ***суматори*** та ***напівсуматори***. Напівсуматори додають два цифрові сигнали, не враховуючи перенос розряду, а повний суматор враховує переповнення в розряді. Для кращого розуміння нагадаємо спосіб додавання десяткових чисел:

$$\begin{array}{r} {\scriptstyle 1\ \ \ } \\ 124 \\ +\ 127 \\ \hline 251 \end{array}$$

В десятковій системі числення числа поділяються на: розряди, одиниці, десятки, сотні, тисячі і т. д. Якщо додавати числа 127 та 124, то спочатку додаються одиниці, з врахуванням переносу (***переповнення***), потім десятки, сотні і т. д. Аналогічно виконується додавання в двійковій системі числення. Важливо, при цьому враховувати перенесення (переповнення) в старший розряд. Напівсуматор не виконують додавання з врахуванням переповнення. Проте повні суматори мають окремий вхід, що враховує переповнення від попередньої операції додавання.

Комбінаційний пристрій: напівсуматор. Існують окремі мікросхеми напівсуматорів, що виконують роль в суматора. Однак, побудувати суматор можна на базових логічних елементах. Однорозрядний напівсуматор будуються на основі елементу XOR «виключене АБО» та елементу AND «логічне І», який зображено на рис. 72.

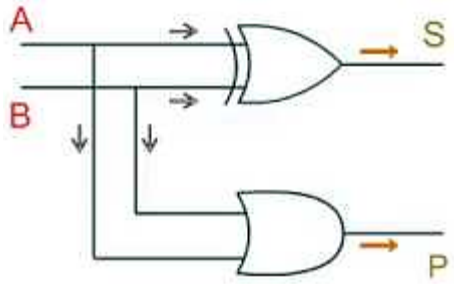


Рисунок 72. - Напівсуматор на логічних елементах.

Як і будь-який логічний цифровий пристрій, напівсуматор має таблицю істинності, яку наведено в табл. 38

Таблиця 38. - Таблиця істинності однорозрядного напівсуматора

A	B	S	P
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

де,

- **A**, **B** - входить суматора (доданки);
- **S** - вихід (результат додавання);
- **P** - вихід перенесення розряду.

Результат додавання двох чисел з'являється на виході логічного елементу XOR (рис. 72), що відповідає його таблиці істинності. Результат переповнення розряду виконує логічний елемент «І», на виході якого з'являється «логічна 1» при переповненні, інакше «логічний 0» при виконанні операції додавання.

На рис. 73 зображено схематичне позначення мікросхеми 74HC86, яка складається з 4-х логічних елементів XOR. В даній мікросхемі:

- вхідні контакти: 1, 2, 4, 5, 9, 10, 12, 13;
- вихідні контакти: 3, 6, 8, 11;
- мінус джерела живлення: контакт №7;
- плюс джерела живлення: контакт №14.

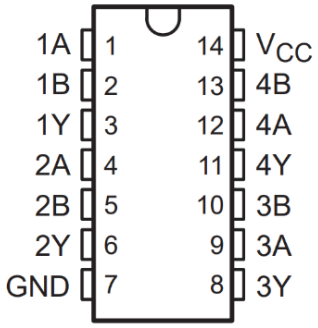


Рисунок 73. - Схематичне позначення мікросхеми 74HC86.

Схему напівсуматора, виконаного на інтегрованих логічних елементах зображено на рис. 74. На схемі DD1 зображено мікросхему 74HC86 (логічний елемент XOR), а DD2 мікросхема 74HC08 (логічний елемент AND «І»). Два сигнали, що додаються подано на вхід напівсуматора за допомогою DIP-перемикача. Якщо тумблер перемикача знаходиться у включеному положенні, тоді на відповідному вході (A чи B) присутній «логічний 0». Якщо тумблер знаходиться у вимкненому положенні, тоді на вході присутня логічна «1». Результат додавання відображається світлодіодами HL1 (червоний), а виникнення переносу світлодіодом HL2 (зелений). Напруга живлення на мікросхемі логіки подається через дільник напруги.

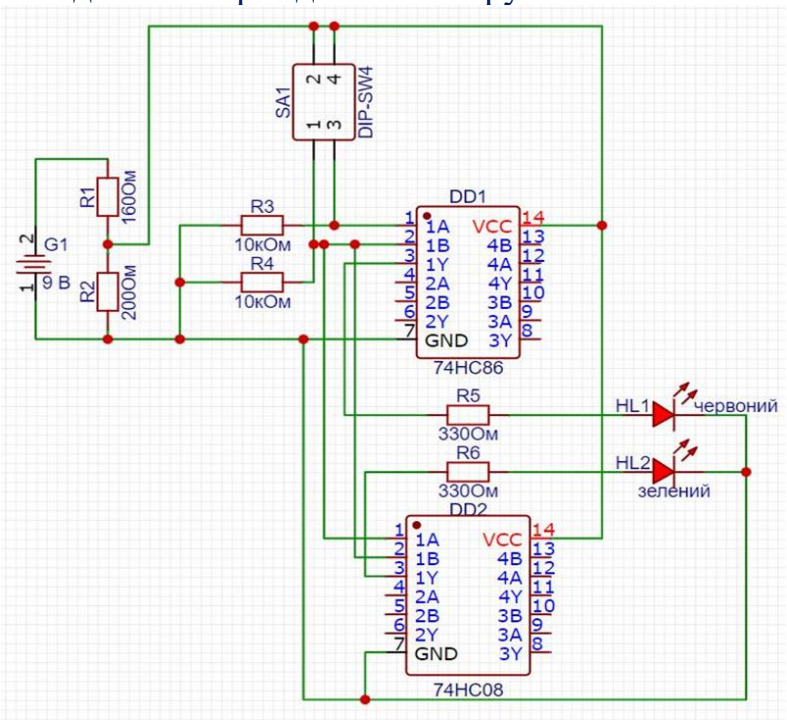


Рисунок 74. – Схема підключення мікросхеми 74HC86 (напівсуматора)

На макетній схемі (рис. 75), за допомогою тумблерів №1 (сигнал A) та №2 (сигнал B) DIP-перемикач подається на вхід напівсуматора різні комбінації сигналів, відповідно до табл. 38.

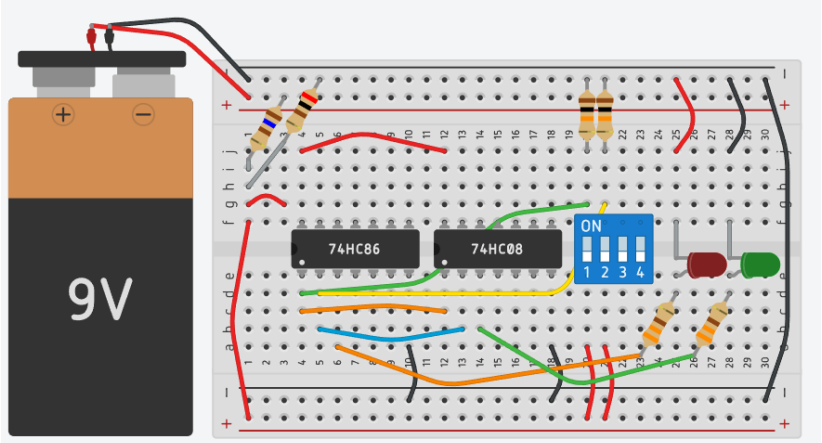


Рисунок 75. - Схема підключення однорозрядного напівсуматора.

Якщо на виході мікросхеми встановлюється логічна «1», то засвітиться відповідний світлодіод. Зелений світлодіод HL2 засвітився лише тоді, коли на обидва входи A та B подано логічні «1». Це означає, що потрібно виконати

перенос сигналу в старший розряд, але на схемі зібрано однорозрядний напівсуматор, то сигнал перенесення не впливає на результат додавання.

Комбінаційний пристрій: повний суматор. На відміну від напівсуматора, повний суматор має окремий вхід для врахування перенесення, яке може з'явитись при операції додавання молодших розрядів. Суматор, окрім вхідних сигналів додає попередній результат переповнення молодшого розряду. Таблиця істинності повного суматора (табл. 39) відрізняється від таблиці істинності напівсуматора.

Таблиця 39. - Таблиця істинності повного суматора.

A	B	Cin	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

де,

- **A**, **B** - входить суматора (доданки);
- **S** - вихід (результат додавання);
- $\mathbf{C_{in}}$ - вхід перенесення до молодшого розряду;
- $\mathbf{C_{out}}$ - вихід перенесення до старшого розряду.

Якщо на вхід повного суматора подано два «логічні 0», і при цьому на вході перенесення від попередньої операції додавання знаходиться «логічна 1», то результат додавання нулів прийме значення «1» (з врахуванням перенесення від молодшого розряду):

$$S = A + B + C_{in} = 0 + 0 + 1 = 1.$$

При додаванні двох «логічних 1», і за наявності на вході C_{in} одиниці, результат буде 1:

$$S = A + B + C_{in} = 1 + 1 + 1 = 1.$$

і на виході C_{out} з'явився «логічна 1». Схема повного суматора на звичайних логічних елементах виглядає так, як наведено на рис. 76.

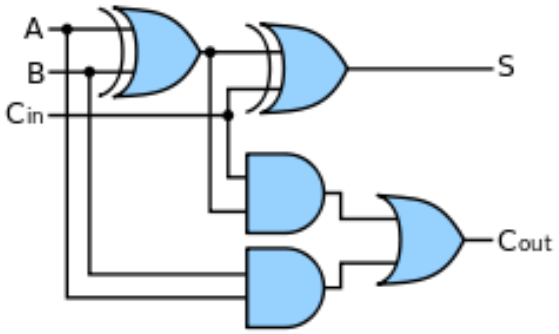


Рисунок 76. - Повний однорозрядний суматор на логічних елементах.

Два доданки A і B надходять на перший елемент XOR, де виводиться результат їх додавання. Два елементи AND (рис. 76) враховують можливість

появи переносу від сигналів на вході суматора (A, B) або від результату додавання (перший елемент XOR) і сигналу на вході C_{in} (вхідний перенос). Виходи цих елементів AND об'єднуються елементами OR, який формує вихідний сигнал перенесення C_{out} (рис. 77)

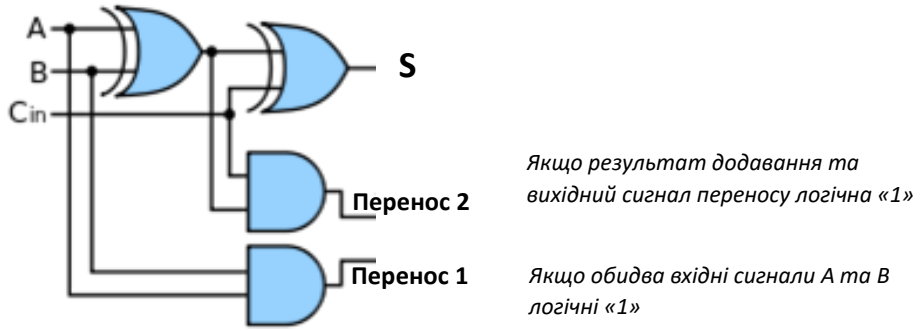


Рисунок 77. - Повний однорозрядний суматор на логічних елементах.

Вхідними сигналами другого елементу XOR є результат додавання доданків A, B та перенесення від молодшого розряду C_{in} На виході цього елементу отримується двійкова сума двох вхідних розрядів з врахуванням переносу від попереднього розряду. Схема повного однорозрядного суматора, виконаного на мікросхемах показано на рис. 78.

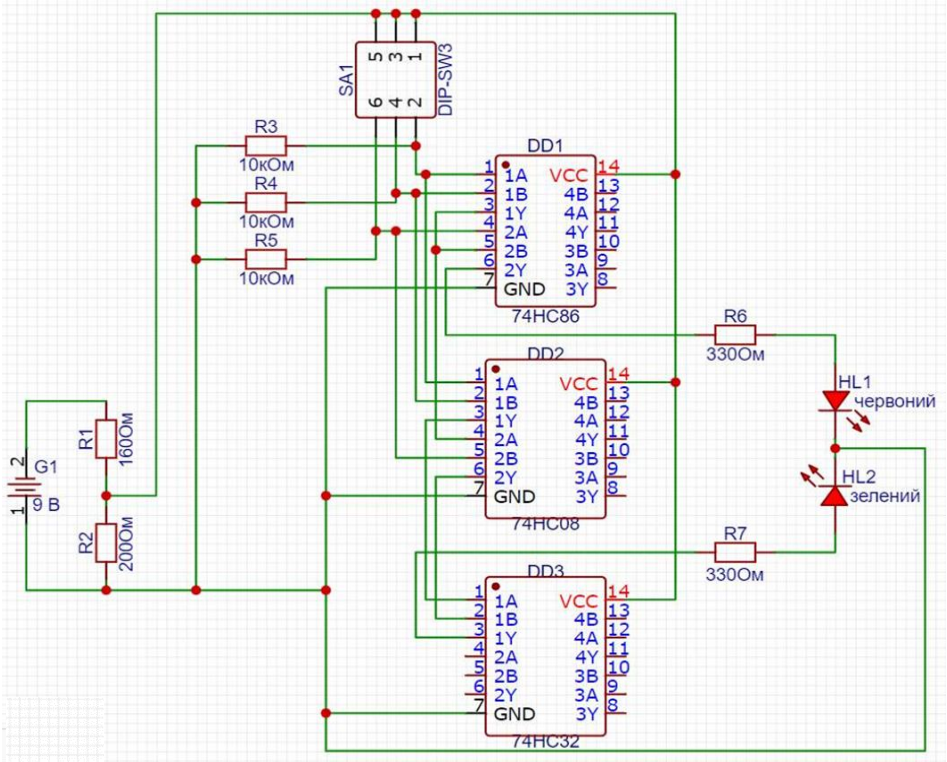


Рисунок 78. – Схема повного однорозрядного суматора

Вхідні цифрові сигнали подаються за допомогою DIP-перемикача: тумблер №1 (вхід A) та №2 (вхід B). За допомогою тумблера №3 можна імітувати надходження сигналу перенесення від молодшого розряду (вхід C_{in}). світлодіоди HL1 та HL2 використовується для відображення сигналів на виходах S (результат додавання) та C_{out} (наявність переносу до старшого розряду).

Після проектування електричної схеми в ***EasyEDA*** створимо макет печатної плати та переглянемо її у форматі **3D** (рис. 79).

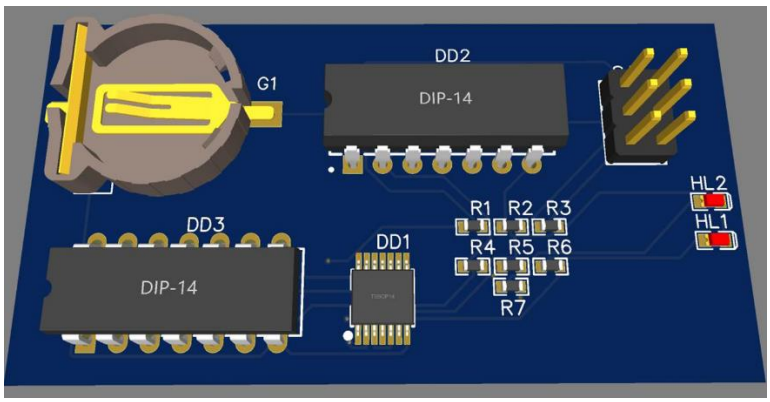


Рисунок 79. – Макет печатної плати повного однорозрядного суматора у форматі 3D.

Зібрану схема повного однорозрядного суматора на макетні платі в ***TinkerCad*** наведено на рис. 80.

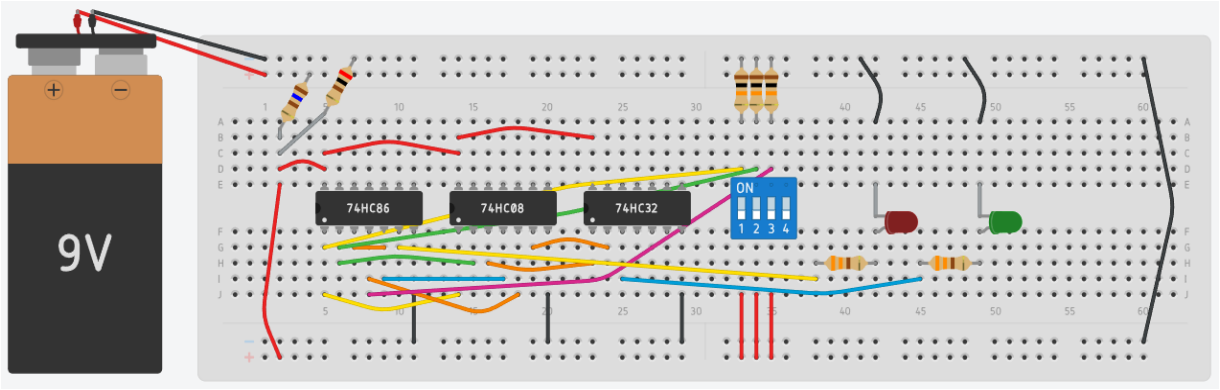


Рисунок 80. - Схема підключення однорозрядного повного суматора.

Користуючись таблицею істинності повного суматора (табл. 39), можна перевірити різні комбінації сигналів (доданки A, B та сигнал перенесення C_{in}). На схемі червоний світлодіод відображає роботу виходу S (результат додавання), а зелений - C_{out} (виникнення переносу до старшого розряду).

З'єднуючи послідовно повні однорозрядні суматори можна зібрати з 2-х, 3-х, 4-х і більше розрядні суматори. Для створення багаторозрядного суматора потрібно вхід C_{in} кожного наступного (старшого) розряду з'єднати з виходом C_{out} попереднього (молодшого) розряду. А на вхід C_{in} найпершого розряду подати «логічний 0» (рис. 81).

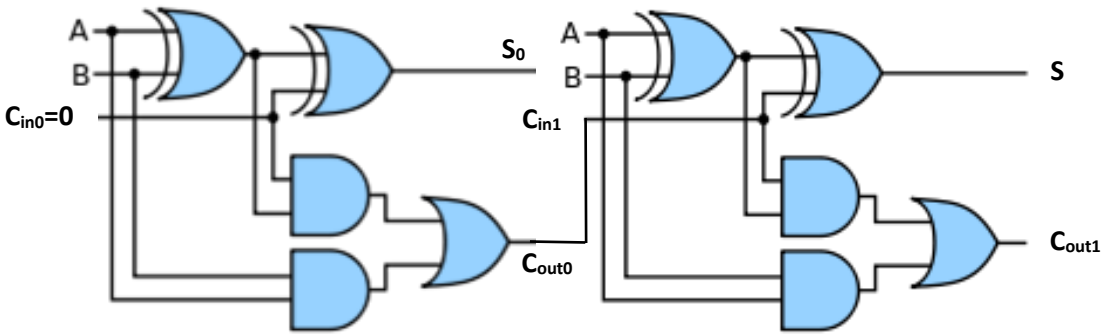


Рисунок 81. - Повний однорозрядний суматор на логічних елементах.

Розглянемо приклад додавання двох двійкових чотирьох розрядних чисел: 1010 + 1011.

$$\begin{array}{r} {\scriptstyle 1\ \ 1} \\ 1010 \\ +\ 1011 \\ \hline 10101 \end{array}$$

Перевірка результату додавання даних чисел у десятковій системі:

$$1010_2 \rightarrow 10_{10}$$
$$1011_2 \rightarrow 11_{10}$$
$$10101_2 \rightarrow 21_{10}$$

Результат додавання: $10_{10}+11_{10}=21_{10}$

Даний приклад можна перевірити на схемі зібраного повного суматора на макетній платі. Зауважимо, що повні суматори застосовуються в обчислювальні техніці, а також при створенні схем цифрових лічильників.

Контрольні запитання

1. Скільки бітів в одному та двох байтах?
2. Чи відрізняються прямий, зворотний і додатковий коди у додатного та у від'ємного числа?
3. Враховується чи ні одиниця переносу із старшого знакового розряду при використанні кодів зворотного та додаткового під час виконання операції додавання?
4. В чому особливість модифікованих кодів?
5. Що треба виконати, якщо в знакових розрядах при використанні модифікованих кодів під час виконання операції додавання з'являється значення 01 або 10?
6. Враховується чи ні одиниця переносу із старшого знакового розряду під час виконання операції додавання при використанні кодів зворотного та додаткового?
7. Чим відрізняється зображення числа в двійковому коді з фіксованою та з плаваючою комою?
8. Що потрібно виконати, якщо в знакових розрядах при використанні модифікованих кодів з'являється значення 01 або 10?
9. Особливості множення цілих двійкових чисел із знаком в ПК і ДК.
10. Що означає термін "часткова сума"?
11. Якщо множене або множник являється від'ємним числом, що треба мати на увазі при виконанні операції множення?
12. Чому отримане після ділення за наведеними вище схемами значення частки іноді не співпадає з очікуваним?
13. Чим відрізняється схема ділення без відновлення залишку від схеми ділення з відновленням залишку?
14. Відобразити схему алгоритму виконання ділення під час формування цілої частини частки.
15. Відобразити схему алгоритму виконання ділення без відновлення залишку, а також з відновленням залишку під час формування дробової частини частки.

Практичне завдання 6.
Дослідження принципів роботи комбінаційних цифрових приладів та їх реалізація на макетних, печатних платах.

Мета роботи: ознайомлення з принципами роботи комбінаційних цифрових приладів: суматорів, компараторів, тригерів та регістрів; побудова електричних схем в онлайн-конструкторі **EasyEDA** на основі мікросхем серії 74; набуття практичних навичок по створенню макетів печатних плат з використанням реалізованих комбінаційних пристроїв в онлайн-конструкторі **EasyEDA** та дослідження режимів їх роботи; створення розгорнутих структур комбінаційних пристроїв в Logic.ly.

Задання 1. Створити електричні-принципові схеми комбінаційних цифрових пристроїв в ***EasyEDA*** з використанням мікросхем 74-ї серії:

- ✓ компаратора на мікросхемі типу 74HC85;
- ✓ RS-тригера на мікросхемі типу 74HC02;
- ✓ D-тригера на мікросхемі типу 74HC74;
- ✓ паралельного регістра на мікросхемі типу 74HC574;
- ✓ реверсного 4-х розрядного лічильника на мікросхемі типу 74HC193;

Побудувати печатні плати створених комбінаційних пристроїв та зберегти їх макети в ***2D*** та ***3D*** в онлайн конструкторі ***EasyEDA***.

Побудувати схеми комбінаційних цифрових приладів на макетних платах конструктора «Цифрова електроніка» та/або в ***TinkerCad***, перевірити принцип їх функціонування відповідно до таблиць істинності.

Завдання 2. Використовуючи дискретні компоненти, створити в ***EasyEDA*** електричні схеми комбінаційних цифрових пристроїв відповідно до заданого варіанту (табл. 40).

Таблиця 40. - Варіанти для побудови комбінаційних пристроїв

№ вар.	*Комбінаційний пристрій*	*Кількість розрядів*	*№ вар.*	*Комбінаційний пристрій*	*Кількість розрядів*
1	компаратора	2	**6**	компаратора	4
	RS-тригера	3		RS-тригера	3
	D-тригера	3		D-тригера	2
	паралельного регістра	4		паралельного регістра	1
	лічильника	2		лічильника	6
2	компаратора	3	**7**	компаратора	2
	RS-тригера	2		RS-тригера	4
	D-тригера	3		D-тригера	3
	паралельного регістра	4		паралельного регістра	1
	лічильника	6		лічильника	8
3	компаратора	4	**8**	компаратора	3
	RS-тригера	2		RS-тригера	2
	D-тригера	3		D-тригера	2
	паралельного регістра	2		паралельного регістра	1
	лічильника	4		лічильника	4

№ вар.	*Комбінаційний пристрій*	*Кількість розрядів*	*№ вар.*	*Комбінаційний пристрій*	*Кількість розрядів*
4	компаратора	2	**9**	компаратора	3
	RS-тригера	4		RS-тригера	4
	D-тригера	2		D-тригера	2
	паралельного регістра	3		паралельного регістра	6
	лічильника	4		лічильника	4
5	компаратора	1	**10**	компаратора	2
	RS-тригера	3		RS-тригера	4
	D-тригера	2		D-тригера	1
	паралельного регістра	4		паралельного регістра	3
	лічильника	6		лічильника	2

Побудувати печатні плати створених комбінаційних цифрових пристроїв та зберегти їх макети в ***2D*** та ***3D*** в онлайн конструкторі ***EasyEDA***.

Побудувати схеми комбінаційних цифрових пристроїв на макетних платах конструктора «Цифрова електроніка» та/або в ***TinkerCad***, перевірити принцип їх функціонування відповідно до таблиць істинності.

Завдання 3. Створити електричні-принципові схеми комбінаційних цифрових пристроїв в ***EasyEDA*** з використанням мікросхем 74-ї серії:

- ✓ JK-тригера;
- ✓ Т-тригера;
- ✓ послідовного регістра4
- ✓ послідовно-паралельного регістра;
- ✓ додатного лічильника;
- ✓ від'ємного лічильника.

Створити в ***EasyEDA*** печатні плати комбінаційних цифрових приладів, подати її у форматі 2D і 3D та зберегти файли цифрових пристроїв.

Побудувати схеми комбінаційних цифрових пристроїв на макетних платах конструктора «Цифрова електроніка» та/або в ***TinkerCad***, перевірити принцип їх функціонування відповідно до таблиць істинності.

Зміст звіту

1. Титульна сторінка, тема та мета практичного заняття.
2. Опис виконаних завдань з наведеними рисунками, графіками, таблицями істинності, логічними виразами, структурними, логічними та електричними схемами створені в онлайн конструкторі **EasyEDA**.
3. Побудувати печатні плати створених логічних елементів та зберегти їх макети. Подати створені печатні плати у форматах **2D і 3D.**
4. Висновки, що відображають результати виконання роботи та їх експериментальний аналіз.

Теоретичні відомості

Комбінаційний пристрій: компаратор. Цифровий компаратор - комбінаційний цифровий пристрій, що виконує порівняння двох чисел у двійковому вигляді, які надходять на його входи.

Розрядність цифрових компараторів може бути різною. Найчастіше використовуються в 4-х розрядні компаратори, які порівнюють два двійкових числа, що складаються з чотирьох біт кожне. Для збільшення розрядності компаратора, їх об'єднують за спеціальними схемами. Як і для всіх попередніх комбінаційних цифрових приладів, компаратор складається з базових логічних елементів (рис. 82).

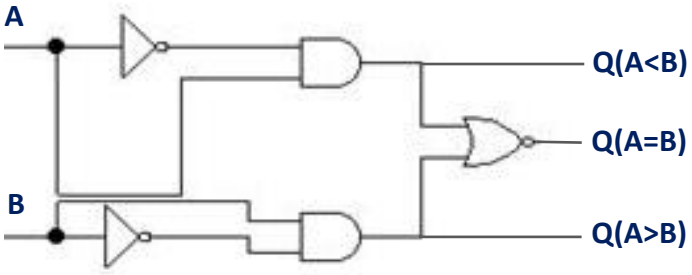


Рисунок 82. - Однорозрядний цифровий компаратор.

На вхід компаратора подаються два однорозрядні числа A та B. На одному з трьох виходів компаратора, в залежності від результату порівняння чисел, з'являються «логічна 1». Якщо сигнал на вході A більший за B, то в результаті «логічна 1» одиниця з'являється на виході $Q_{A>B}$, якщо B більше за A, то «логічна 1» буде на виході $Q_{A<B}$ У випадку, коли сигнали A та B рівні, «логічна 1» буде на виході $Q_{A=B}$.

Якщо виникає потреба порівняти двійкові числа, що мають більшу кількість розрядів, тоді об'єднують необхідну кількість однорозрядних композиторів. Найчастіше компаратори застосовуються в арифметико- логічних пристроях (АЛУ) для порівняння чисел, а також в системах, де обробляється великий потік даних і потрібно виявляти моменти часу, коли надходять однакові дані.

Для побудови чотирьох розрядного компаратора використовується мікросхема 74HC85, схематичне зображення якої наведено на рис. 83. Дана мікросхема містить:

- вхідні контакти: 10 (A0), 12 (A1), 13 (A2), 15 (A3) - призначені для подачі чотирьох розрядів числа A;
- вхідні контакти: 9 (B0), 11 (B1), 14 (B2), 1 (B3) - призначені для подачі чотирьох розряд числа B.

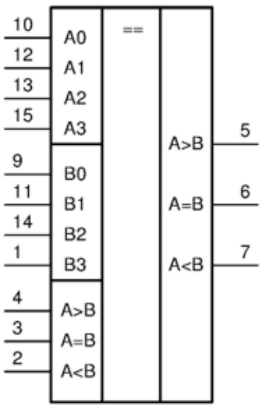


Рисунок 83. - Схематичне позначення мікросхеми 74HC85.

Ці контакти є інформаційними, на які подаються числа, що в подальшому порівнюються між собою. Мікросхема компаратор 74HC85 має три вихідні контакти:

- 5 ($Q_{A>B}$) – «логічна 1» сповіщає що число А більше за В;
- 6 ($Q_{A=B}$) – «логічна 1» сповіщає що число А дорівнює В;
- 7 ($Q_{A<B}$) – «логічна 1» сповіщає що число А менше за В;

Дана мікросхема має три каскадні входи:

- 2 $I_{A<B}$;
- 3 $I_{A=B}$;
- 4 $I_{A>B}$.

Дані контакти призначені для каскадного об'єднання декількох компараторів, при нарощенні розрядності двійкових чисел. До контакту 8 підключається мінус джерела живлення, до контакту 16 підключається плюс джерела живлення. Максимальна напруга живлення 7В. Принцип роботи цифрового компаратора 74HC85 наведено в табл. 41.

Таблиця 41.- Таблиця істинності компаратора 74HC85

Вхідні сигнали				***Вихідні сигнали***		
A3, B3	A2, B2	A1, B1	A0, B0	$Q_{A>B}$	$Q_{A<B}$	$Q_{A=B}$
A3>B3	X	X	X	H	L	L
A3<B3	X	X	X	L	H	L
A3=B3	A2>B2	X	X	H	L	L
A3=B3	A2<B2	X	X	L	H	L
A3=B3	A2=B2	A1>B1	X	H	L	L
A3=B3	A2=B2	A1<B1	X	L	H	L
A3=B3	A2=B2	A1=B1	A0>B0	H	L	L
A3=B3	A2=B2	A1=B1	A0<B0	L	H	L
A3=B3	A2=B2	A1=B1	A0=B0	H	L	H

Умовні позначення, які наведено в таблиці істинності шифратора означають наступне:

- L – низький рівень сигналу (low - 0);
- H - високий рівень сигналу (high - 1);
- X – нейтральний рівень сигналу.

Зауваження! Порівняння двох чисел компаратор розпочинає із значень двох старших розрядів (А3 та В3).

Принцип роботи цифрового компаратора наведено на рис. 84. Напруга живлення до мікросхеми подається через дільник, виконаний на резисторах R1 та R2. У схемі підключення мікросхеми 74HC85 перевірити принцип роботи цифрового компаратора з наступними дискретними компоненти:

- мікросхема DD1 типу 74HC85 (74HC85D,652);
- R1 резистор ємністю 160 Ом;
- R2 резистор ємністю 200 Ом;
- R3, R4, R5 резистори ємністю 330 Ом;
- R6 – R13 резистори ємністю 10 кОм;
- HL1 зелений світлодіод;
- HL2 жовтий світлодіод;
- HL3 червоний світлодіод;
- SA1 DIP-перемикач SW8 та G1 батарея живлення 9В.

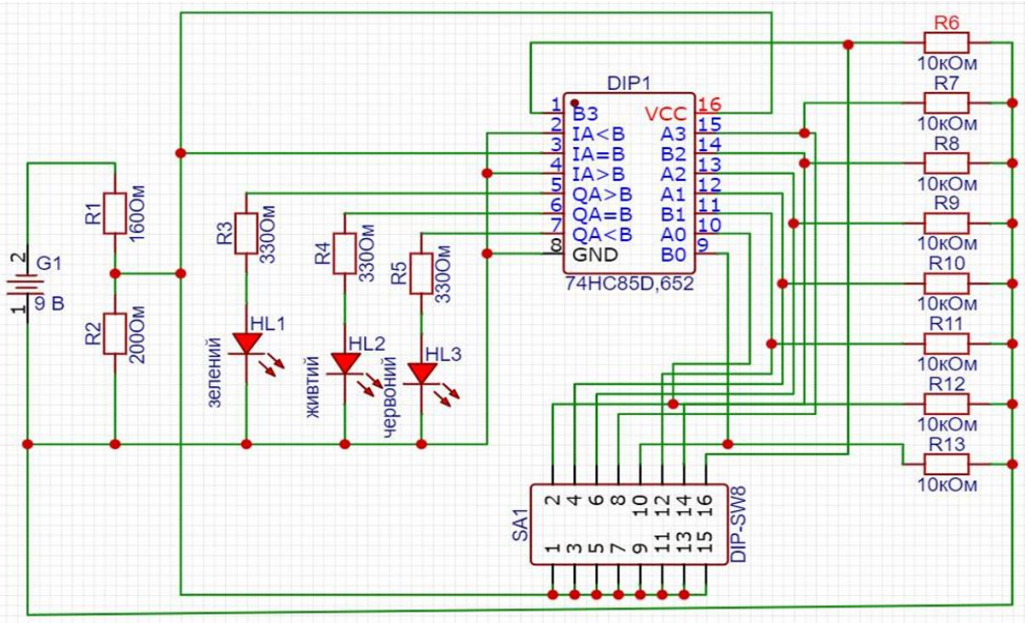


Рисунок 84. – Схема цифрового чотирьохрозрядного компаратора

Зібрати схему цифрового компаратора на макетній платі на основі мікросхем 74HC85 відповідно до рис. 85. Перевірити режими функціонування компаратора на макетній платі та записати відповідну таблицю істинності режимів його роботи.

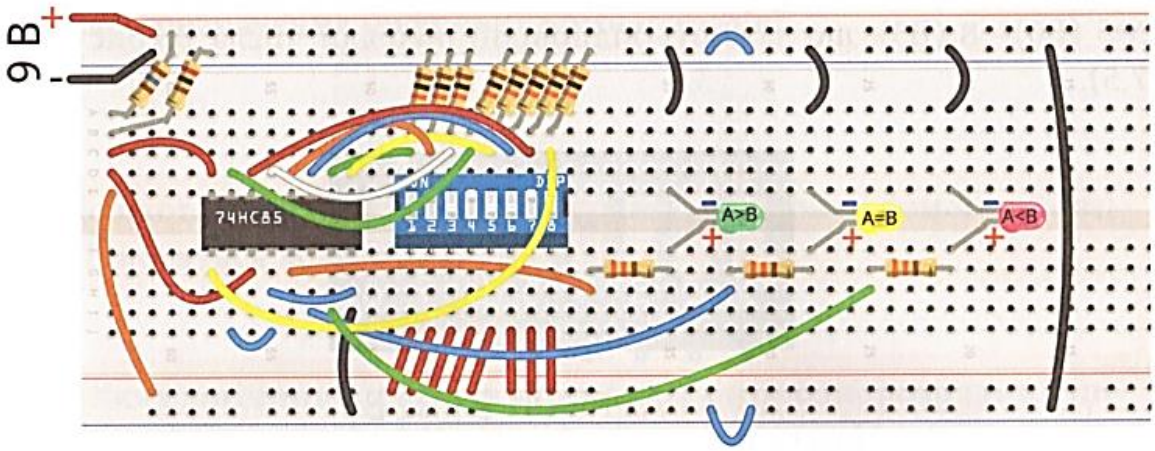


Рисунок 85. - Схема підключення цифрового компаратора.

Комбінація вхідних чисел задається за допомогою DIP-перемикача:

- тумблери №1-4 (A0-A3) задаються для першого числа;
- тумблери №5-8 (B0-B3) задаються для другого числа;

Якщо тумблер знаходиться в положенні ON, то на відповідний інформаційний вхід в компаратора (розряд числа) подається «логічна 1», в іншому випадку – «логічний 0». Зібрати схему на макетні платі можна скориставшись зразком рис. 8. Каскадні входи №2 та №4 необхідно підключити до мінус джерела живлення, а вхід №3 до середньої точки дільника напруги. Каскадів входи не використовуються в даній схемі, а рівень напруги на них обрано відповідно до документації виробника даної мікросхеми.

Зелений світлодіод підключено до виходу $Q_{A>B}$, жовтий - до $Q_{A=B}$, червоний - до $Q_{A<B}$. Порівнюючи два десяткові числа 8_{10} (1000_2) - число A та 5_{10} (0101_2) - число B, важливо врахувати, що в двійковій системі числення найменший розряд

числа знаходиться праворуч, а найстарший ліворуч. Розглянемо два числа за розрядами для їх порівняння (табл. 43).

Таблиця 42.- Порівняння двох чисел компаратором.

Число А (8_{10} - 1000_2)				***Число В (5_{10} - 0101_2)***			
A3	***A2***	***A1***	***A0***	***B3***	***B2***	***B1***	***B0***
1	0	0	0	0	1	0	1

Спочатку порівнюються найстаріші розряди А3 та В3: А3 більше В3. Судячи з таблиці 3, далі вже немає сенсу порівнювати інших розряди, оскільки можна одразу зробити висновок, що двійкове число А більше за В. Отже $8_{10} > 5_{10}$. При встановленні даних чисел на схемі компаратора, важливо пам’ятати, що тумблери №1-4 - число А, причому №1 відповідає молодшому розряду А0, №2 – А1, №3 – А2, №4 – А3. Аналогічно для числа В: тумблери №5 (В0) – 8 (В3).

Після того як числа встановлені, повинен засвітити зелений світлодіод тобто А>В. Самостійно порівняйте числа, які наведено в табл. 43. та запишіть їх результати.

Таблиця 43.- Порівняння двійкових чисел 4-х розрядним компаратором.

Число А	***Число В***	***Результат A>B, A<B, A=B***	***Результат роботи світлодіодів: зелений, червоний, жовтий.***
10_{10} (1010_2)	15_{10} (1111_2)		
3_{10} (0011_2)	3_{10} (0011_2)		
13_{10} (1101_2)	12_{10} (1100_2)		
7_{10} (0111_2)	4_{10} (0100_2)		

Мікросхеми компаратора мають чотири вхідні розряди для кожного числа, однак якщо задіяти каскадні входи та з'єднати декілька мікросхем, то можна отримати 8-ми та більше розрядні компаратори.

Комбінаційний пристрій: тригер. До послідовних цифрових приладів відносяться *тригери*. За функціональними можливостями (принципову зміну свого стану) тригери бувають наступних видів:

- *RS (SR) – тригер;*
- *D – тригер;*
- *JK (J) – тригер;*
- *T – тригер.*

В послідовних логічних пристроях вихідний сигнал залежить не тільки від вхідних сигналів, але й від внутрішнього стану пристрою. Тобто від тих сигналів, які були на вході цифрового пристрою в попередній моменти часу і призвели до зміни внутрішнього стану приладу. Отже цифрові прилади, які відносяться до послідовних, повинні мати елементи пам'яті для збереження інформації про попередній стани вхідних сигналів. До таких елементів належать тригери.

Тригер - цифровий прилад, що здатен тривалий час знаходитись в одному із сталих положень і перемикати в інше в інше під дією керуючих вхідних сигналів. *Сталим* положенням називається сформована логічна «1» або «0», що встановлюється на виході тригера і залишається там навіть після завершення дії керуючого вхідного сигналу. Тригер може зберігати (запам'ятовувати) один розряд двійкового числа «0» або «1». Тригери бувають:

- статичні;
- динамічні.

Статичні тригери перемикаються рівнем напруги керуючого сигналу, а динамічні перепадом (фронтом керуючого сигналу) з високою на низький рівень або навпаки. За реакцією керуючого сигналу тригери поділяються на:

- асинхронні;
- синхронні.

Асинхронні тригери змінюють свій стан в момент надходження керуючого сигналу. *Синхронні тригери* замінюють свій стан при надходженні керуючого сигналу лише в момент надходження спеціального синхронізуючого сигналу на особливому вході.

За способом реакції на синхронізуючий сигнал синхронні тригери поділяються на такі що:

- реагують на високий рівень сигналу;
- реагують на попередній (наростаючий) фронт сигналу;
- реагують на задній (спадаючий) фронт сигналу.

RS-тригер - зберігає на виході попереднє значення при наявності на обох керуючи входах логічного «0», і змінює своє вихідне значення на нове, при надходженні на один із входів логічної «1». Даний тригер має два керуючі входи:

- *S (set)* – встановлення;
- *R (reset)* – скидання.

Якщо на вхід S подана логічна «1», то на виході (Q) тригера встановиться «1», а при появі логічної «1» на вході R, вихід тригеру переключиться до «0». Якщо на обидва входи прийшли «1», то вихід тригеру буде знаходитись у невизначеному стані, який залежить від способу реалізації даного тригера.

Для більшості *RS*-тригерів така комбінація вхідних сигналів є забороненою. Також *RS*-тригер має інверсний вихід ($\overline{Q}$), стан якого протилежний до прямого.

Якщо на входах *R* та *S* присутній логічний «0», то стан тригеру не змінюється і залишається таким, яким був в попередній момент часу (до надходження обох нулів). Перевірити функціонування *RS*-тригеру можна за допомогою таблиці істинності (табл. 44).

Таблиця 44.- Таблиця істинності RS-тригера.

Входи		***Вихід в попередній момент часу***		***Вихід в наступний момент часу***	
R	***S***	***Q(t)***	$\overline{Q(t)}$	***Q(t+1)***	$\overline{Q(t+1)}$
0	0	0	1	0	1
0	0	1	0	1	0
0	1	1	0	1	0
0	1	0	1	1	0
1	0	1	0	0	1
1	0	0	1	0	1
1	1	1	0	не визначено	не визначено
1	1	0	1	не визначено	не визначено

RS-тригер можна реалізувати за допомогою логічних елементів (рис. 86). Існують синхронні *RS*-тригери (рис. 87), які мають додатковий вхід *C*.

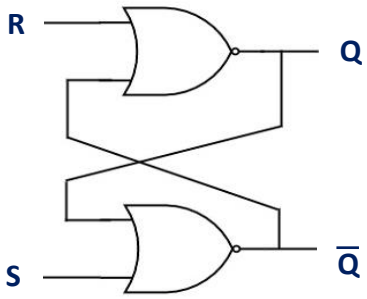


Рисунок 86.- *RS*-тригер на логічних елементах «АБО-НІ»

Надходження на даний вхід логічної «1», змусить *RS*-тригер змінити вихідний стан (за наявності на одному із керуючи входів логічної «1» (табл. 45).

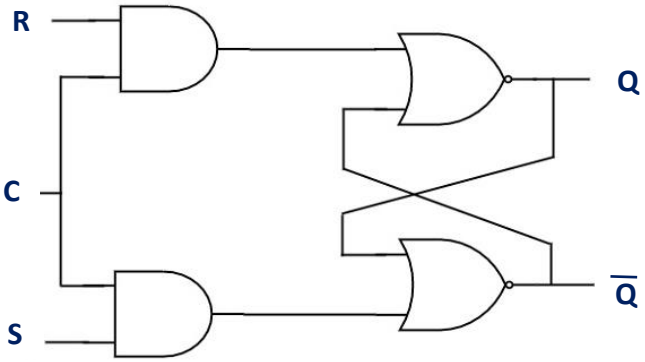


Рисунок 87.- Синхронний *RS*-тригер на логічних елементах/

Надходження на даний вхід логічної «1», змусить *RS*-тригер змінити вихідний стан (за наявності на одному із керуючи входів логічної «1» (табл. 45).

Таблиця 45.- Таблиця істинності синхронного RS-тригера.

Входи		***Синхронізуючий вхід***	***Вихід в попередній момент часу***		***Вихід в наступний момент часу***	
S	***R***	***C***	***Q(t)***	$\overline{Q(t)}$	***Q(t+1)***	$\overline{Q(t+1)}$
X	X	0	1	0	1	0
X	X	0	0	1	0	1
0	0	1	0	1	0	1
0	0	1	1	0	1	0
1	0	1	1	0	1	0
1	0	1	0	1	1	0
0	1	1	1	0	0	1
0	1	1	0	1	0	1
1	1	1	1	0	не визначено	не визначено
1	1	1	0	1	не визначено	не визначено

де, X – нейтральне значення сигналу.

За наявності на синхронізуючому вході С логічного «0» нуля, неважливо який сигнал подається на входи *S* та *R*, оскільки вихід тригеру не змінить свій стан і залишився таким, як і в попередній момент часу.

Схема *RS*-тригера, виконаного на логічних елементах «*АБО-НІ*» наведена на рис. 88. В якості елемента АБО-НІ застосовується мікросхема 74HC02. У схемі RS-тригера передбачити підключення з наступними дискретними та комбінаційними компонентами:

- мікросхема DD1 типу 74HC02 (SN74HC02D);
- R1 резистор ємністю 160 Ом;

- R2 резистор ємністю 200 Ом;
- R3, R4 резистори ємністю 10 кОм;
- R5, R6 резистори ємністю 330 Ом;
- HL1 червоний світлодіод;
- HL2 зелений світлодіод;
- SA1 DIP-перемикач SW4;
- G1 батарея живлення 9В.

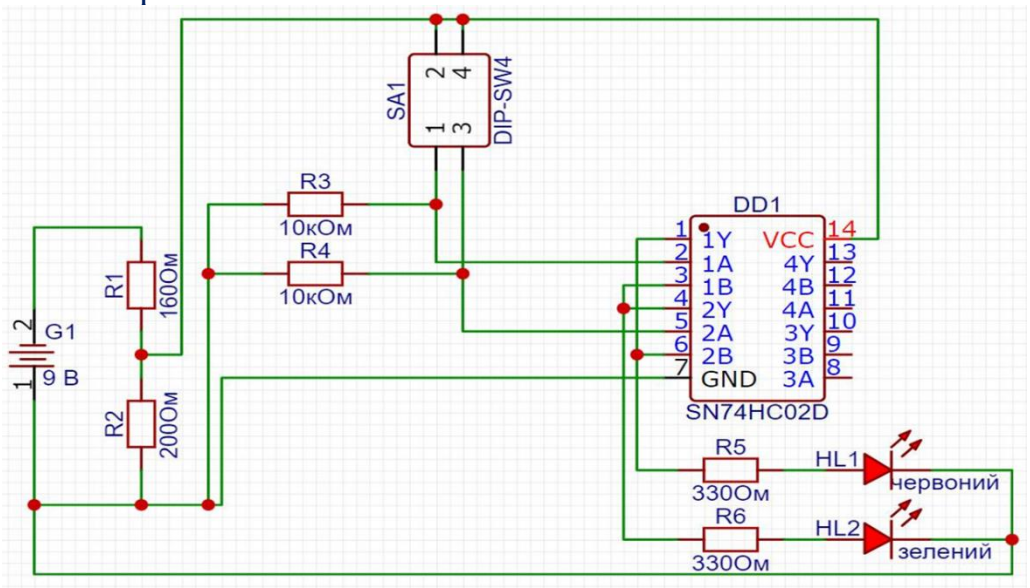


Рисунок 88. – Схема RS-тригера на логічних елементах АБО-НІ (NOR).

Зібрати схему RS-тригера на логічних елементах АБО-НІ на макетній платі на основі мікросхемb 74HC02 відповідно до рис. 89. Перевірити режими функціонування RS-тригера на макетній платі та записати відповідну таблицю істинності режимів його роботи.

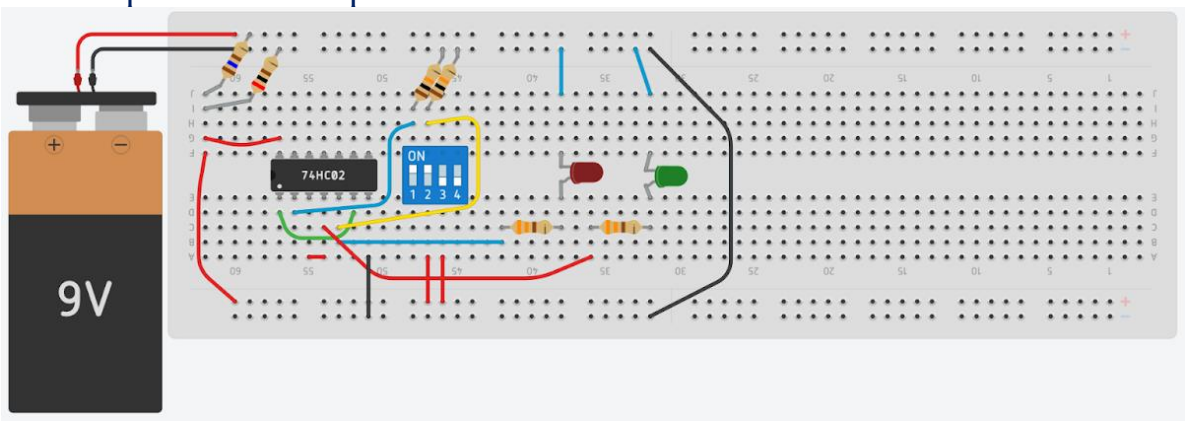


Рисунок 89. - Схема підключення RS-тригера на логічних елементах АБО-НІ.

Тумблер №1 *DIP*-перемикача відповідає за подачу вхідного сигналу R, а тумблер №2 – сигналу *S*. В мікросхемі 74HC02 задіяні два логічні елементи АБО-НІ. Вихід першого елементу 1Y відповідає прямому виходу *RS*-тригера і підключений до червоного світлодіоду HL1, а другий вихід підключений до зеленого світлодіоду HL2.

Коли тумблер увімкнений (ON), то на відповідний вхід тригера подається логічна «1», коли він вимкнений, то на вході подано логічний «0». Світлодіод на виході засвітиться, в разі наявності на ньому логічної «1». Змінюючи різні

комбінації вхідних сигналів за допомогою тумблерів: №1 (*R)* та №2 *(S) DIP*-перемикача, можна визначити, який світлодіод засвічується, позначаючи зміну відповідного виходу тригеру у стан логічної «1». Червоний світлодіод світиться у випадку коли на прямому вході Q встановлюється рівень логічної «1», при цьому на інверсному виході $\overline{Q}$ буде логічний «0», тому зелений світлодіод не буде світитися. А якщо на прямому виході Q буде логічний «0», то червоний світлодіод погасне і засвітиться зелений, це свідчить, що на інверсному виході $\overline{Q}$ присутня логічна «1».

Наприклад, початковий момент тумблери №1 та №2 *DIP*-перемикача знаходяться у вимкненому положенні. Після підключення батарейки світиться зеленим світлодіод – на інверсному $\overline{Q}$ логічна «1», а на прямому Q – «0». Якщо тумблер №2 перевести в положення (ON), то на вхід *S* подається «1», що призведе до зміни сигналів на виході: на Q тепер буде «1», а на $\overline{Q}$ – «0». Зелений світлодіод погасне, а червоної засвітиться. Повернувши тумблер №2 у вимкнене положення, червоний світлодіод буде світитись, а якщо увімкнути тумблер №1, засвітиться зеленим світлодіод, бо на вхід R подано логічну «1», що скинуло прямий вихід Q у стан логічного «0».

D-тригер - зберігає інформацію щодо стану входу *D* та передає її на вихід при надходженні активного рівня (логічної «1») синхронізуючого сигналу.

D-тригер зберігає стан входу при неактивному рівнів сигналу синхронізації. Даний тригер має два входи D, C і один вихід, таблиця істинності якого наведена в табл. 46.

Таблиця 46.-Таблиця істинності D-тригера.

Вхід	***Синхронізуючий вхід***	***Вихід в попередній момент часу***		***Вихід в наступний момент часу***	
D	***C***	***Q(t)***	$\overline{Q(t)}$	***Q(t+1)***	$\overline{Q(t+1)}$
0	0	1	0	1	0
0	0	0	1	0	1
1	0	1	0	1	0
1	0	0	1	0	1
0	1	0	1	0	1
0	1	1	0	0	1
1	1	0	1	1	0
1	1	1	0	1	0

Отже, коли активний рівень синхронізуючого сигналу відсутні (на вході C), то на виході *D*-тригера зберігається той рівень сигналу, що був у попередній момент часу. Як тільки на вході C з'являється високий рівень сигналу, те значення сигналу, що було на вході *D*-тригера, записується та з'являється на його виході. І немає значення, який рівень сигналу був до цього на виході. Даний тригер може мати інверсний вихід, рівень сигналу якого протилежний до прямого виходу. На рис. 90 представлена схема *D*-тригера на логічних елементах. Для тестування *D*-тригера використовується спеціалізована мікросхема 74HC74 (рис. 91). Дана мікросхема має два інтегровані *D*-тригери, входами яких є контакти №2 (1D) та №12 (2D). Відповідними їх видами є контакти 5 (1Q), 6 ($\overline{1Q}$) та 8($\overline{2Q}$), 9 (2Q).

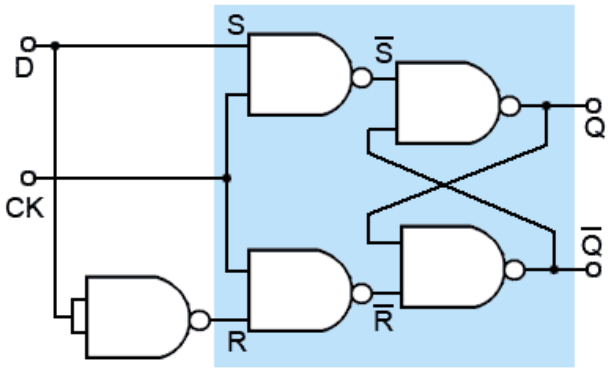


Рисунок 90.- *D*-тригер на логічних елементах.

Кожен тригер має інверсійні входи, встановлення ($\overline{\boldsymbol{SQ}}$), та скидання ($\overline{\boldsymbol{RQ}}$), які встановлюють на виході логічну «1» або «0» відповідно. Для цього на один із цих входів потрібно подати логічний «0». Одночасно подавати сигнали логічного «0» на ці входи не можна, оскільки це призведе до невизначеного стану, за якого на прямому та інверсному виходах будуть встановлені логічні «1».

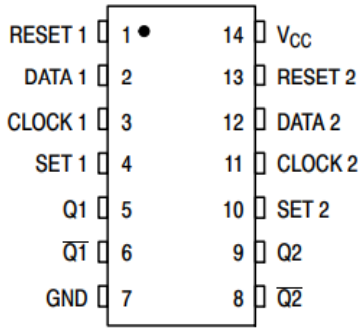


Рисунок 91. - Схематичне позначення мікросхеми 74HC74 (D-тригер).

Використовуючи контакти ($\overline{\boldsymbol{SQ}}$), ($\overline{\boldsymbol{RQ}}$), та виходи мікросхеми 74HC74 можна створити RS-тригер з інверсними входами. Крім того мікросхема 74HC74 має синхронізуючі входи 1CP та 2CP. Для того, щоб мікросхема 74HC74 працювала як синхронний *D*-тригер, необхідно входи ($\overline{\boldsymbol{RQ}}$) та ($\overline{\boldsymbol{SQ}}$), підключити до вихідного рівня напруги, а входи *D* та *C* до тумблерів *DIP*-перемикача (рис. 92).

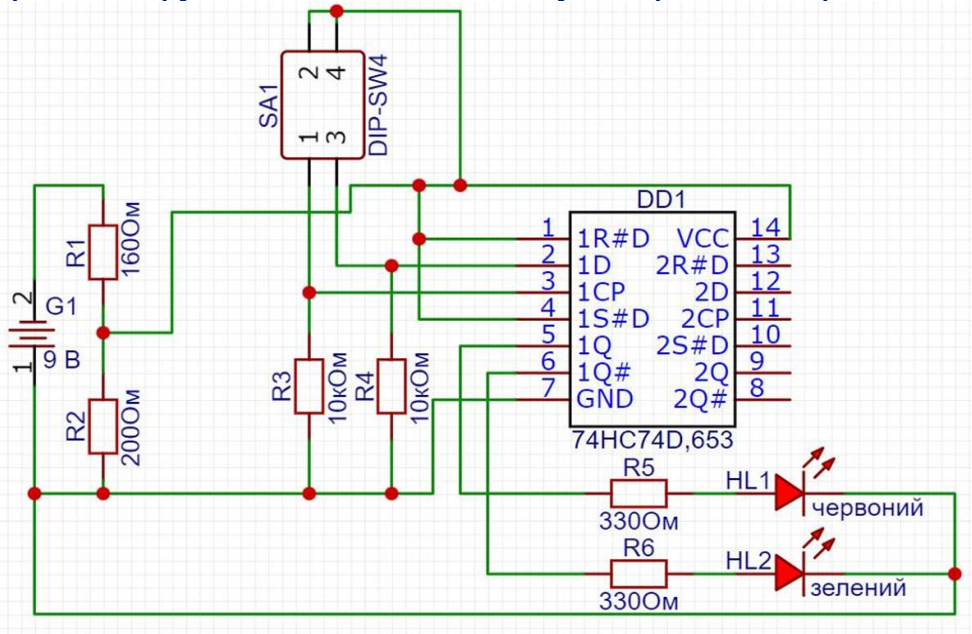


Рисунок 92. – Схема синхронного D-тригера.

У схемі підключення на основі мікросхеми 74HC74 перевірити принцип роботи D-тригера з наступними дискретними компоненти:

- мікросхема DD1 типу 74HC74 (74HC74D,653);
- R1 резистор ємністю 160 Ом;
- R2 резистор ємністю 200 Ом;
- R3, R4 резистори ємністю 10 кОм;
- R5, R6 резистори ємністю 330 Ом;
- HL1 червоний світлодіод;
- HL2 зелений світлодіод;
- SA1 DIP-перемикач SW4 та G1 батарея живлення 9В.

Зібрати схему D-тригера на макетній платі на основі мікросхем 74HC74 відповідно до рис. 93. Перевірити режими функціонування D-тригера на макетній платі та записати відповідну таблицю істинності режимів його роботи.

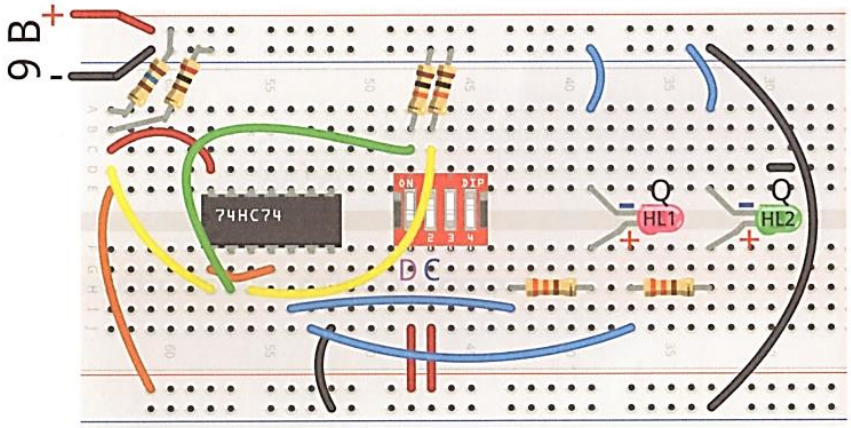


Рисунок 93. - Схема підключення синхронного D-тригера на макетній платі.

За наявності вході *D* логічної одиниці та наростаючого фронту синхронізуючого імпульсу *C*, на прямому виході *D*-тригера буде «1» - засвітитися червоний світлодіод. Якщо на вхід *D*-тригера подати логічний «0», а на *C* синхронізуючий сигнал, то на прямому виході буде «0» - червоний світлодіод погасне, засвітиться зелений.

Під наростаючим фронтом синхронізуйте сигналу слід розуміти момент перемикання сигналу із низького на високий рівень - передній фронт імпульсу. Тобто, якщо тумблер №2 був вимкнений, то для подачі синхронізуючого сигналу, його потрібно перевести в положення ON. І навпаки, якщо тумблер №2 був увімкнений (положення ON), то для повторної подачі синхронізуючого сигналу, тумблер потрібно вимкнути, а потім знову перевести положення ON. Режими роботи *D*-тригера наведено в табл. 46.

Вхід *D* під'єднано до тумблеру №1, а вхід *C* - до тумблеру №2. За вимкненого тумблера на вхід подається низький рівень сигналу, а якщо його перевести положення ON, то високий. Якщо залишити тумблер №1 у вимкненому положенні, а тумблер №2 включити, то таким чином можна подати імпульс на синхронізуючий вхід і сигнал з виходу *D* записується до тригеру та передається на вихід. Тоді засвітиться зелений світлодіод, який підключено до інверсного виходу і логічний «0» подано на прямий вихід, а червоний світлодіод не буде світитися, бо на інверсному вході з'являється логічна «1».

Коли включено тумблер №1, а потім вимкнути його та включити тумблер №2, то засвітиться червоний світлодіод, оскільки на вхід тригера подано і записано логічну «1». Якщо залишити тумблер №2 в такому ж положенні, а тумблер №1 перемикати, то змін не відбудеться, оскільки на вхід *C* не подається зростаючий фронт синхронізуючого сигналу (він залишається в незмінному стані).

T-тригер - перемикає вихідний сигнал (стану тригеру) до протилежного значення при надходженні на вхід керуючого сигналу високого рівня (табл. 47). Побудувати Т-тригер можна на базі *RS*-тригера або *D*-тригера.

Таблиця 47.-Таблиця істинності Т-тригера.

Вхід	***Вихід в попередній момент часу***	***Вихід в наступний момент часу***
D	***Q(t)***	***Q(t+1)***
0	0	0
0	1	1
1	0	1
1	1	0

JK-тригер – принцип його роботи аналогічний *RS*-тригеру: сигнал *J* встановлює тригер у стан логічної «1», а сигнал *K* – у стан логічного «0». За наявності двох одиниць на керуючий входах, вихідний сигнал тригеру змінюється на протилежне значення (табл. 48).

Таблиця 48.- Таблиця істинності JK-тригера.

Входи		***Вихід в попередній момент часу***		***Вихід в наступний момент часу***	
J	***K***	***Q(t)***	$\overline{Q(t)}$	***Q(t+1)***	$\overline{Q(t+1)}$
0	0	0	1	0	1
0	0	1	0	1	0
1	0	1	0	1	0
1	0	0	1	1	0
0	1	1	0	0	1
0	1	0	1	0	1
1	1	1	0	0	1
1	1	0	1	1	0

Комбінаційний пристрій: регістр. Регістр – послідовний цифровий прилад, призначений для зберігання інформації та виконання певних дій над нею. За способом завантаження даних регістри поділяються на:

- послідовні;
- паралельні;
- послідовно-паралельні (комбіновані).

Паралельні регістри призначені для збереження інформації. Їх можна створити використовуючи *RS*-тригера або *D*-тригера (рис. 94). Кількість тригерів дорівнює кількості двійкових розрядів числа, які необхідно занести до регістру.

Паралельні регістри одночасно приймають всі двійкові розряди числа та здатні одночасно подавати їх на вихід.

Послідовні регістри (регістри зсуву) мають один вхід, через який розряд за розрядом передаються двійковій біти числа. Дані регістри здатність зсувати вліво або вправо в залежності від керуючого сигналу.

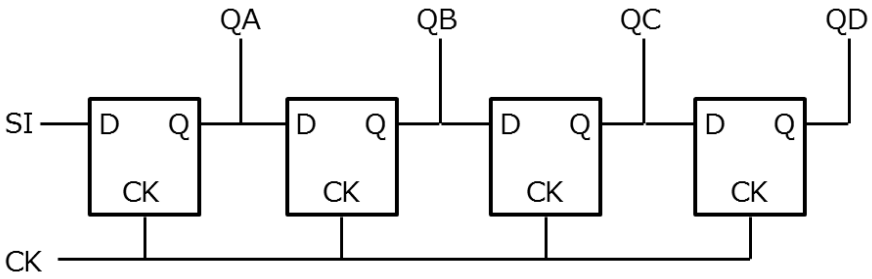


Рисунок 94.- Умовне позначення паралельного регістра на *D*-тригерах.

Інтервал часу, за який завантажується один біт двійкового числа у послідовний регістр, прийнято називати *тактом*. Кількість тактів, необхідних для завантаження даних в регістр, залежить від розрядності регістру.

Послідовно-паралельні регістри мають можливість завантажувати дані обома способами.

Паралельний регістр виконано на інтегральній мікросхемі 74HC574 (рис. 95). Дана мікросхема має 8 вході D0-D7 та 8 виходів Q0-Q7, два керуючих входи: інверсний $\overline{OE}$ (дозволяє/забороняє зчитувати числа), синхронізуючий CP.

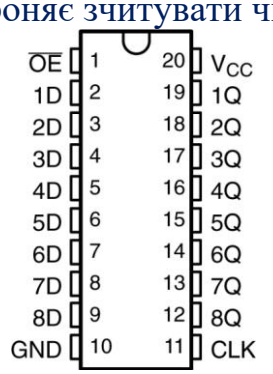


Рисунок 95. - Схематичне позначення мікросхеми 74HC574 (паралельного регістра).

Якщо на вході $\overline{OE}$ присутній високий рівень сигналу (логічна «1»), то зчитування числа з регістра не відбудеться - його виходи будуть вимкнені (перейдуть у високоімпедансний стан). Якщо в цей час змінювати вхідні дані та подавати синхронізуючий сигнал, то вони завантажаться до регістру. Коли на $\overline{OE}$ переходить логічний «0», то збережене значення відображається на виході. Для передачі збережених в реєстрі даних на вихідні контакти не обов'язково подавати синхронізованої сигнал CP.

Таким чином, даний регістр має синхронний вхід для запису даних та асинхронний вихід для зчитування результату. На рис. 96 представлено схему підключення реєстру 74HC574 для збереження на передачі даних. У схемі підключення на основі інтегральної мікросхеми 74HC574 перевірити принцип роботи паралельного регістра з наступними дискретними компоненти:

- мікросхема DD1 типу 74HC574 (74HC574PW,112);
- R1 резистор ємністю 160 Ом;
- R2 резистор ємністю 200 Ом;
- R3 – R11 резистори ємністю 10 кОм;
- R12 – R19 резистори ємністю 330 Ом;
- HL1 – HL8 червоні світлодіоди;

- SA1 DIP-перемикач SW8;
- SA1 кнопка замикання та G1 батарея живлення 9В.

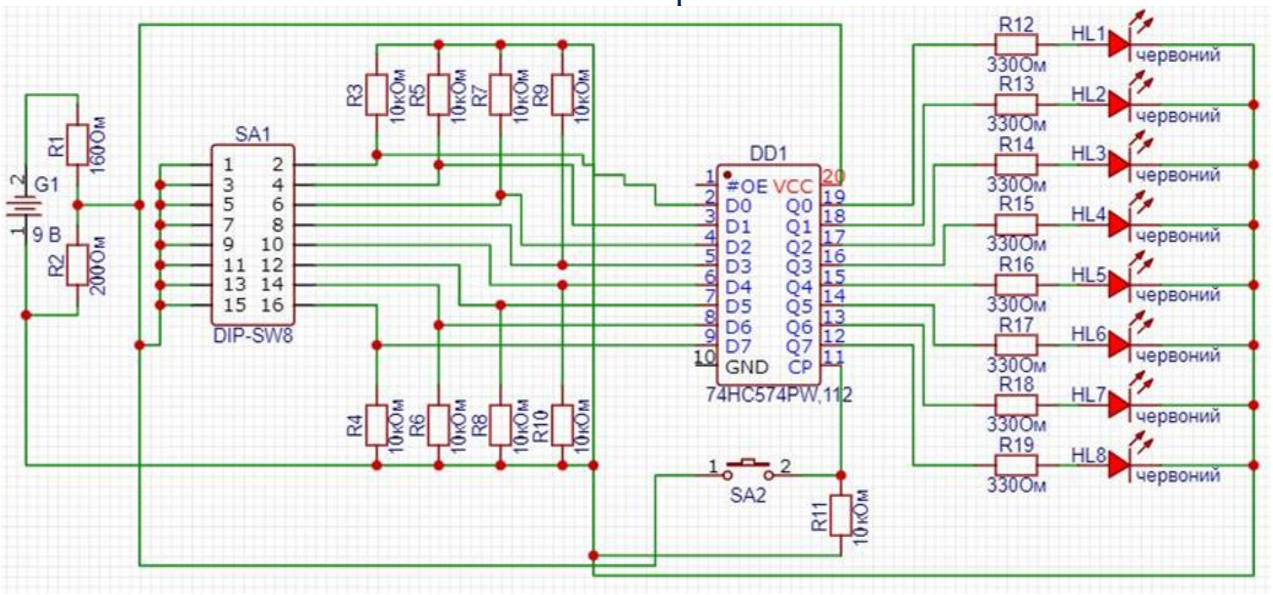


Рисунок 96. – Схема підключення мікросхеми 74HC574 (паралельного регістра).

Дані на вхід регістру подаються за допомогою *DIP*-перемикача. Входи Q0-Q7 підключенні до тумблерів №1-№8 відповідно. Виходи регістра підключені до світлодіодів HL1-HL8. Дозволяючий вхід підключено до логічного нуля (мінус джерела живлення). Сигнал на синхронізуючий вхід CP подається за допомогою кнопки SA2. Замикання кнопки SA2 формує зростаючий фронт синхронізуючого сигналу.

Зібрати схему на макетній платі можна відповідно до рис. 97. Після того, як схема буде зібрана і джерело живлення підключене, то можуть світитися деякі світлодіоди.

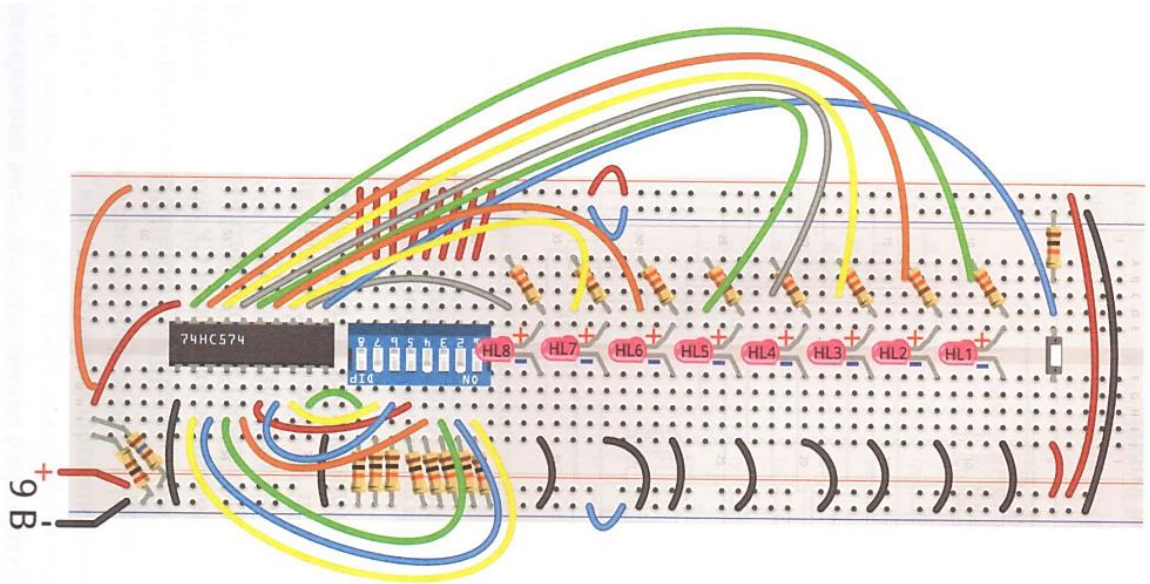


Рисунок 97. - Схема підключення паралельного регістра на макетній платі.

Це можуть бути дані, які з'явилися під час підключення живлення до схеми. За рахунок перехідних процесів у регістрі, тригери могли встановитися у довільне значення. Якщо встановити тумблер №1 та №3 *DIP*-перемикача в

положення ON, тоді дані готові до запису в регістр. Для того, щоб вони вже записались та з'явились на виході мікросхеми, потрібно активувати кнопку SA2, тоді засвітяться світлодіоди HL1 та HL3.

Перевірити режими функціонування паралельного регістра на макетній платі та записати відповідну таблицю істинності режимів його роботи.

Важливо! В даній схемі батарейка та дільник напруги, виконаний на резисторі R1 та R2 підключаються з іншої сторони макетної плати на відміну від попередніх схем. В порівнянні з іншими схемами на макетних платах *DIP*-перемикач повернутий на 180°. Позначка ON знаходиться з цієї сторони макетної плати, до якої підключена батарейка.

Можна встановлювати будь-яку комбінацію вихідних сигналів за допомогою *DIP*-перемикача і, активуючи кнопку, відображаються відповідні дані на виході регістра. Експериментуючи з різноманітними комбінаціями даних на вході регістра, можна перевірити його принцип роботи.

Комбінаційний пристрій: лічильник. Лічильник - послідовний цифровий прилад, призначений для підрахунку кількості вхідних імпульсів.

Лічильники мають складнішу, в порівнянні з регістрами, структуру та систему зв'язків, хоча вони і побудовані на основі тригерів різних типів. В основному всі лічильники будуються на *Т-тригерах.* Цифрові лічильники вміють рахувати на:

- збільшення;
- зменшення;
- в обох напрямках.

В залежності від напрямку підрахунку розрізняють наступні типи лічильників:

- додатні (рахують на збільшення);
- від'ємні (рахують на зменшення);
- реверсні (рахують в обох напрямках).

Реверсивні лічильники можуть змінювати напрям руху за допомогою зміни між розрядних внутрішніх зв'язків. За напрям рахунку відповідає спеціальний керуючий сигнал. Кількість двійкових розрядів лічильника визначається кількістю тригерів. Якщо лічильник має N тригері, то і двійкова розрядність його дорівнює N. Всі лічильники спрацьовують по фронту (зростаючому або спадаючому) тактового сигналу.

Додатні лічильники починають відлік від «0» до максимального значення, після чого знову починають відлік від «0». *Від'ємні лічильники* рахують на зменшенням від максимального значення «0», після чого знову повертаються до свого максимального значення і починають повторний відлік.

За модулем відліку лічильники поділяються на:

- двійкові;
- двійкового-десяткові (рахують від 0 до 9 і знову повертаються в 0);
- з довільним модулем відліку.

Двійкові лічильники, в залежності від кількості двійкових розрядів, можуть рахувати максимально до значення 2^N-1, де N - кількість розрядів. Якщо

лічильник має чотири двійкових розряди, то рахувати він може від 0 до 15 або від 15 до 0, в залежності від свого типу.

Двійково-десяткові лічильники мають 4 входи та здатні рахувати від 0 до 9, після чого знову починається рахунок з 0. Лічильники з довільним модулем відліку рахуючи вхідні імпульси до числа, яке задається окремо. Розглянем реверсивний 4-х розрядний з можливістю завантаження початкового значення (рис. 98).

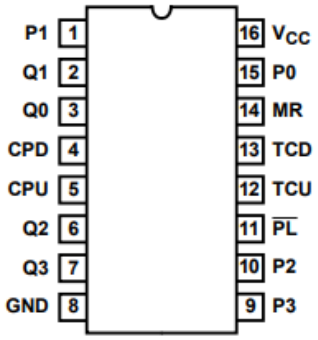


Рисунок 98. - Схематичне позначення мікросхеми 74HC193 (реверсного синхронного лічильника).

Даний лічильник має 4 входи D0-D3 для завантаження числа (в двійковому вигляді), з якого лічильник починає свій відлік. Якщо дані входи не використовуються, лічильник відраховує значення від 0 до 15 і навпаки. На виходах Q0-Q3 під час підрахунку вхідних імпульсів відображається результат у двійковому вигляді. Тобто, при кожному вхідному імпульсі лічильник додає або віднімає по одиниці до заданого вхідного числа, а результат відображається на виході. Вихід Q0 - наймолодший двійковий розряд результату, Q3 - найстарший двійковий розряд.

Режим роботи лічильника задається за допомогою вхідних сигналів CPD, CPU. Подаючи імпульси на вхід CPU, лічильник рахує на збільшення, тобто додає до поточного числа в пам'яті лічильника по 1 при кожному прийнятому імпульсі на вході CPU. На вході CPD при цьому потрібно встановити логічну «1» (високий рівень напруги). І навпаки, якщо потрібно рахувати на зменшення, то на вхід CPU подається логічна «1», а на CPD - вхідні імпульси. При надходженні кожного з них, від поточного значення в пам'яті лічильника віднімається по одиниці.

Крім того, мікросхема лічильника має керуючий вхід MR, що за наявності на ньому логічної «1», встановлює на всіх виходах Q0-Q3 лічильника логічні «0», незалежно від встановленого на вході числа. Є ще один керуючий інверсний вхід - $\overline{PL}$. Його функція полягає у завантаженні до лічильника встановленого на входах D0-D3 числа і передача його до виходів Q0-Q3. Для цього потрібно подати на цей вхід логічний «0».

На інверсних виходах $\overline{TCU}, \overline{TCD}$ завжди присутній високий рівень сигналу логічна «1», крім випадку, коли лічильник дорахував до одного з крайніх значень (0 або 15 відповідно). Тобто, коли лічильник рахує на збільшення і дорахував до 15, на виході $\overline{TCU}$ з'явився логічний «0», а на виході $\overline{TCD}$ – «1». І навпаки, коли лічильник рахує на зменшення, дійшов до 0 і знову починає рахувати з 15, на

виході $\overline{TCD}$ з'явиться логічний «0», а на виході $\overline{TCU}$ – логічна «1».

На рис. 99 представлена схема підключення лічильника в електричний ланцюг. Вхід MR через кнопку SA2 підключений до високого рівня напруги (середня точка дільника напруги). У схемі підключення на основі інтегральної мікросхеми 74HC193 перевірити принцип роботи реверсного синхронного лічильника з наступними дискретними компоненти:

- мікросхема DD1 типу 74HC193 (74HC193DB-Q100J);
- R1 резистор ємністю 160 Ом;
- R2 резистор ємністю 200 Ом;
- R3, R4 резистори ємністю 10 кОм;
- R5 – R8 резистори ємністю 330 Ом;
- HL1 – HL4 червоні світлодіоди;
- SA1 DIP-перемикач SW4;
- SA1 кнопка замикання;
- C1, C2 керамічні конденсатори 100 нФ;
- G1 батарея живлення 9В.

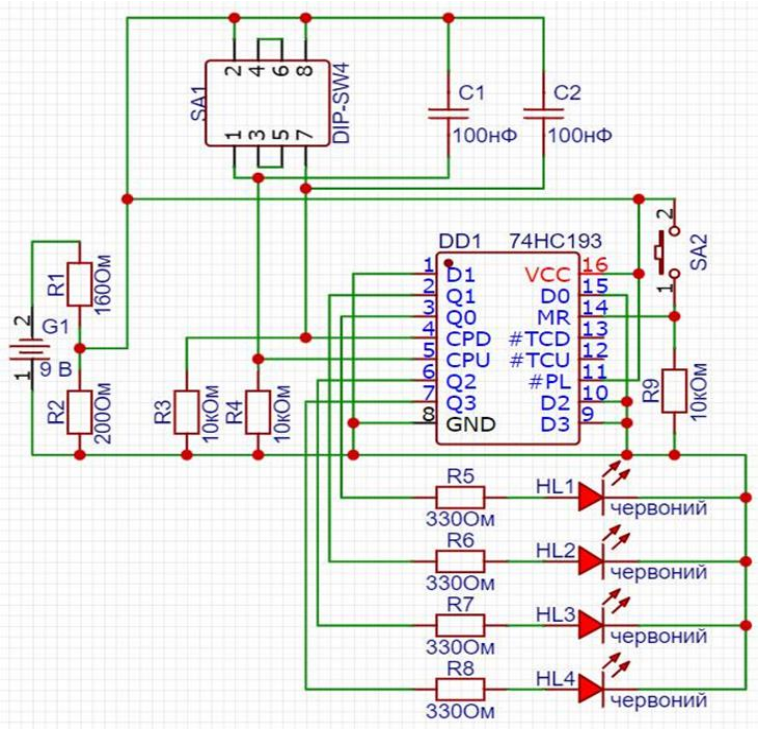


Рисунок 99. – Схема підключення мікросхеми 74HC193 (реверсного синхронного лічильника).

За розімкнутим контактів кнопки, на вхід MR подається логічний «0». Проте, за необхідності встановити на всіх виходах Q0-Q3 нулі, при натисканні на кнопку SA2 подається, таким чином на MR логічна «1». Вхід $\overline{PL}$ підключений до дільника напруги, що забезпечить на ньому стабільний високий рівень напруги логічна «1». Входи D0-D3 не задіяні тому підключенні до мінуса джерела живлення (нульовий потенціал). До виходів Q0-Q3 лічильника підключені світлодіоди для відображення результату підрахунку імпульсів у двійковому коді. Світлодіод HL1 -наймолодший двійковий розряд, HL4 – найстарший. При такому підключенні сигналів не використовується можливість

завантаження будь-якого значення до лічильника та його примусове скидання (запис усіх нулів до двійкових розрядів лічильника).

Конденсатори С1 та С2 необхідні у схемі для формування тактового імпульсу (рис. 100). Вхідні імпульси на входи CPD, CPU подаються шляхом перемикання тумблерів *DIP*-перемикача: №1 – CPD (рахунок на зменшення), №4 - CPU (рахунок на збільшення). Тумблери №2 та 3 не задіяні.

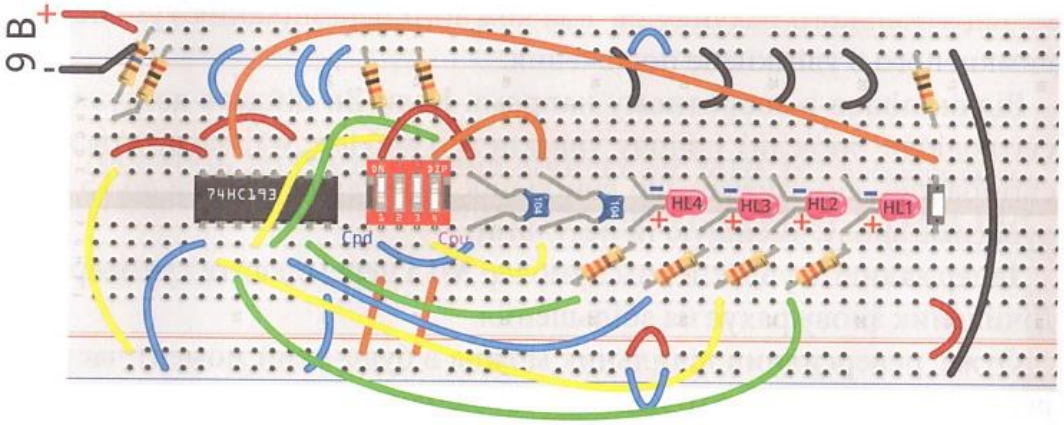


Рисунок 100. - Схема підключення реверсного синхронного лічильника на макетній платі.

Перевірити режими функціонування реверсного синхронного лічильника на макетній платі та записати відповідну таблицю істинності режимів його роботи.

Рахунок лічильника на збільшення. Після того, як схема зібрана на макетній платі та подане живлення, необхідно натиснути на кнопку SA2, щоб встановити на всі виходах лічильника логічні «0». Якщо перевести тумблер №1 *DIP*-перемикача в положення ON, то, таким чином, буде подана на лічильник логічна «1» і активізують режим рахунку на збільшення. Перемикаючи тумблер №4 між положеннями ON та OFF, можна подавати тактові імпульси на вхід CPU.

В результаті лічильник додає імпульси та відображає результат в двійковому вигляді на світлодіодах. При переключенні тумблерів, збільшується рахунок в лічильнику до тих пір поки всі світлодіодної не засвітяться. В результаті лічильник дорахує до числа 15_{10} (1111_2). Після чого необхідно скинули отриманий результат, переключаючи тумблери *DIP*-перемикача та натиснувши на кнопку SA2.

Рахунок на зменшення. Якщо залишити тумблер №4 в положенні ON, а тумблер №1 підключити до входу CPD, то можна перемикати різні положення *DIP*-перемикача. За такого підключення лічильник рахує на зменшення і віднімає від загального рахунку по одиниці за кожного перемикання тумблера із вимкненого або включеного положення. Відлік пішов у зворотному напрямі. Коли буде досягнуто нуля у відліку (всі світлодіоди погаснуть). Якщо знову включити тумблер №1, засвітяться всі світлодіоди, що свідчить про початок відліку на зменшення. Можна залишити включеним тумблер №1 та перемикати тумблер №4 лічильник знову розпочне рахунок на додавання.

Отже, реверсивний лічильник можна в будь-який момент часу перевести в інший режим відліку і він продовжить рахувати на збільшення або зменшення

від поточного значення. Також за допомогою кнопки SA2, що підключена до входу MR, можна встановити на виходах лічильника нулі.

Прикладом застосування лічильників можуть бути комплексні системи, де використовуються мікроконтролери та спеціалізоване програмне забезпечення, яке, наприклад, веде підрахунок кількості осіб, щоб відвідати певний захід або приміщення. Також лічильники застосовуються при створенні таймерів, або якості дільника частоти, тощо.

Контрольні запитання

1. На які групи поділяються цифрові пристрої? Яка між ними різниця?
2. Призначення та структура суматора.
3. Що таке однорозрядний напівсуматор, принцип дії?
4. Побудова повного суматора в різних логічних елементах.
5. Що таке повний суматор, принцип дії?
6. Побудова повного суматора в різних логічних елементах.
7. Паралельні, послідовні та паралельно-послідовні суматори
8. Переваги і недоліки паралельного суматора.
9. Послідовне та паралельне формування переносів.
10. Що таке суматор, принцип дії?
11. Що таке неповний суматор, принцип дії?
12. Скільки входів має напівсуматор та суматор?
13. Скільки розрядів складає напівсуматор?
14. Що таке тригер, принцип дії?
15. Які існують типи тригерів?
16. В чому особливість асинхронних тригерів?
17. Наведіть таблицю станів RS-тригера та поясніть його роботу.
18. Яка комбінація вхідних сигналів RS-тригера вважається недопустимою?
19. В чому принципова різниця між RS-тригером та JK-тригером?
20. Призначення регістрів, принцип дії?
21. Типи регістрів в залежності від способу обробки інформації.
22. Типи регістрів в залежності від способу введення інформації.
23. Принципи побудови паралельних регістрів.
24. Принципи побудови послідовних регістрів зі зсувом вправо.
25. Принципи побудови послідовних регістрів зі зсувом вліво.
26. Призначення лічильника. Його параметри.
27. Типи лічильників в залежності від лічильного модуля.
28. Типи лічильників в залежності від схеми побудови.
29. Принцип побудови паралельних лічильників.
30. Принцип побудови лічильників із наскрізним переносом.
31. Принцип побудови додатного, від'ємного та реверсного лічильника.
32. Як виконується нарощування розрядності лічильників.

Практичне завдання 7.
Моделювання пристроїв на базі мікроконтролера Arduino Uno

Мета роботи: створити моделі пристроїв на базі мікроконтролера ***ArduinoUno***; дослідити схему з'єднань, скласти програми (скетчі) управління мікроконтролером ***ArduinoUno***; дослідити характеристики та принципи їх керування в онлайн-редакторі ***Tinkercad***.

Завдання 1. Використовуючи порт введення-виведення мікроконтролера ***Arduino***, створити програми керування:

- ✓ світлодіодами;
- ✓ триколірним світлодіодом;
- ✓ фоторезистором;
- ✓ датчиком руху;
- ✓ датчиком температури.

Зібрати відповідні схеми на макетних платах ***Arduino*** та/або в ***Tinkercad***. Підключити схеми до комп'ютера через USB-порт плати Arduino. Скласти та завантажити скетч керування портом плати Arduino. Перевірити правильність роботи відповідного скетчу.

Завдання 2. Відповідно до заданого варіанту, на базі мікроконтролерів ***Arduino Uno, Arduino Mega*** створити модель[1] керування:

- ✓ світлофором;
- ✓ новорічною гірляндою;
- ✓ світлодіодною стрічкою;
- ✓ міні-синтезатором;
- ✓ міні-ігор («Ковбої», «Стрибки», «Вгадай код», «Автотрек») та інші.

Зібрати відповідну схему на макетній платі ***Arduino*** та/або в ***Tinkercad***. Підключити схеми до комп'ютера через USB-порт плати Arduino. Скласти та завантажити скетч керування проекту Arduino. Перевірити правильність роботи відповідного скетчу.

Зміст звіту

1. Титульна сторінка, тема та мета практичного заняття.
2. Подати зовнішній вигляд модулів підключених схем в Arduino Uno.
3. Тексти скетчів відповідних модулів.
4. Вивести дані моделювання досліджених режимів роботи.
5. Висновки, що відображають результати виконання роботи та їх експериментальний аналіз.

Теоретичні відомості

Ознайомлення з Arduino. До складу комплекту **Arduino** входять:

- мікроконтролер аналог мікропроцесора у звичайному ПК;
- кнопка скидання здійснює скидання мікроконтролера та повторний

[1] *Варіант та унікальність керованої моделі узгоджується з викладачем.*

запуск програми;

- порт USB забезпечує зв'язок з ПК та живлення пристрою;
- світлодіод №13 світлодіод, з'єднаний із цифровим виходом №13;
- живлення +9В додаткове живлення від зовнішнього джерела (батарея, блок живлення);

Підключення Arduino Uno до ПК. Для роботи з Arduino Uno знадобиться будь-який персональний комп'ютер із вільним USB-портом (рис. 101). У разі підключення Arduino до ПК зовнішнє живлення пристрою не потрібне.

Рисунок 101. - Основні елементи Arduino Uno.

Запуск Arduino Uno із зовнішнім джерелом. Для включення мікроконтролера, достатньо подати на нього живлення або від ПК через USB-кабель, або безпосередньо через спеціальний роз'єм зовнішнього живлення. При цьому, напруга зовнішнього живлення може змінюватись від +7В до +12В.

Макетний стенд та макетна плата. Макетний стенд складається із двох частин (рис. 102). При цьому є можливість нарощувати додаткові частини як зверху, так і знизу стенду.

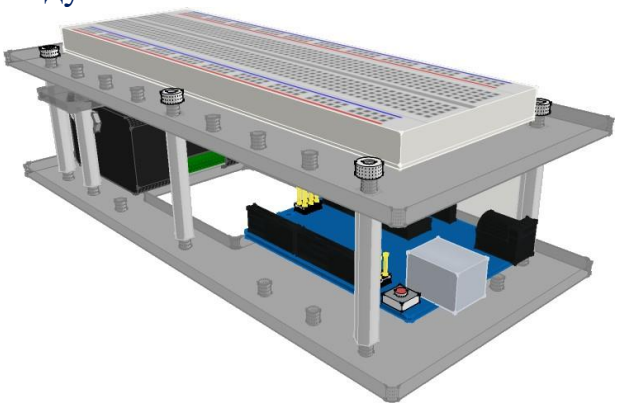


Рисунок 102. - Макетний стенд Arduino Uno.

На нижньому поверсі стенду розміщується сам контролер Arduino Uno і, опціонально, акумуляторна батарея для автомної роботи. На верхньому поверсі змонтовано макетну плату, яка дозволяє здійснювати складання електричних ланцюгів

без паяння. На рис. 103 наведена схема плати, на якій показані всі можливі з'єднання її провідників.

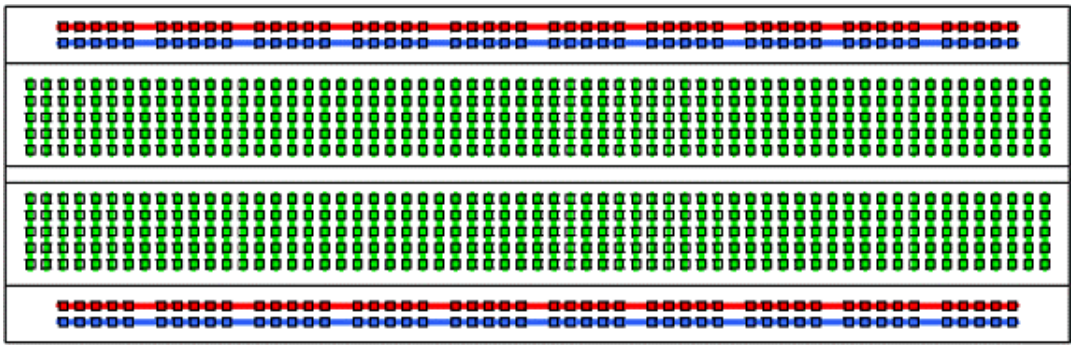


Рисунок 103. - Схема плати Arduino Uno.

По краях плати йдуть довгі провідники, які найчастіше називають шиною живлення. Провідники ж розміщені в середині плати, що з'єднуються у напрямку, зворотному до напряму шини живлення (рис. 104).

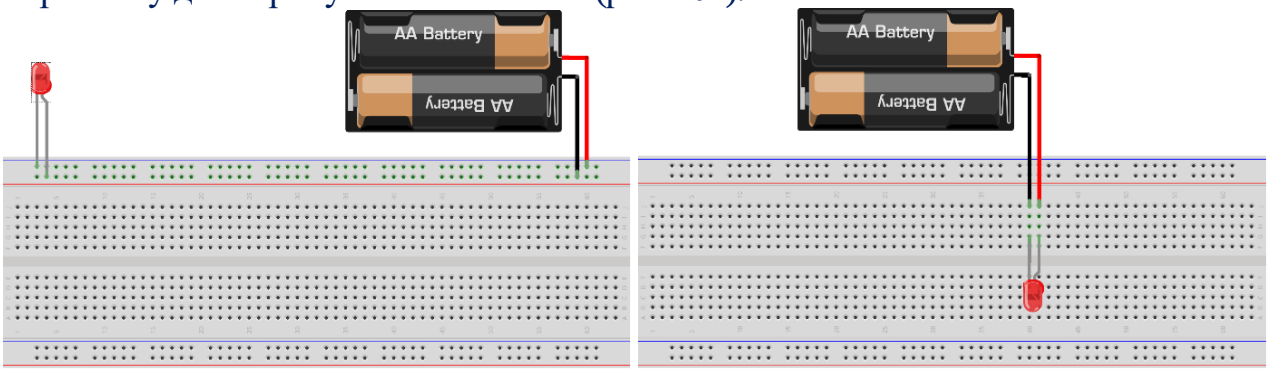
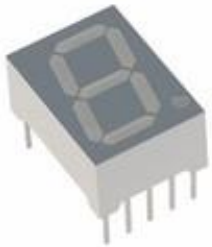
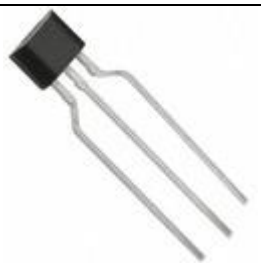
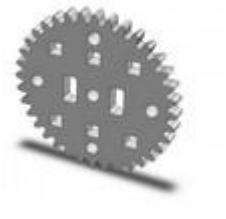


Рисунок 104. - Приклади використання макетної плати.

Розглянемо основні елементи навчального набору ***Arduino Uno***, перелік та зовнішній вигляд яких наведено в табл. 49

Таблиця 49. - Елементи навчального набору Arduino Uno

Світлодіод червоний - 2шт Світлодіод зелений - 2шт Світлодіод жовтий - 2шт	Резистор 220 Ом - 6 шт Резистор 10 кОм - 2шт Резистор 1кОм - 2шт	 Індикатор світлодіодний сегментний - 1шт
Потенцометр регулювальний 47 кОм - 1шт	Потенцометр підстрочний 10кОм - 1шт	Кнопка 2 шт

Дисплей жидкокристалічний - 1шт	Драйвер двигунів постійного струму - 1шт	Двигун постійного струму з редуктором - 1шт
Серво-двигун - 1шт	Датчик температури - 1шт	Датчик освітлення - 1шт
 Датчик Холла - 1шт	 Колесо з магнитом - 1шт	Ультразвуковой дальномер -1шт

Редактор програм Arduino IDE. Розглянемо основні компоненти редактору програм ***Arduino*** (рис. 105):

1. *Головне меню.* Меню редактора включає такі основні елементи: файл, правка, скетч, сервіс і довідка. Розглянемо докладніше кожен із них. У пункті ***Файл*** можна знайти команди, відповідальні створення нової програми, читання, збереження її змін, і команди завантаження програми мікроконтролера.

- ✓ створити нову програму (скетч);
- ✓ відкрити існуючу програму;
- ✓ папка зі скетчами відкрити програму із заданої папки;
- ✓ приклади відкрити приклад програми;
- ✓ закрити поточне вікно.
- ✓ зберегти зміни у раніше збереженій програмі;
- ✓ зберегти як зберегти нову програму, із зазначенням імені;
- ✓ завантажити програму в Arduino;
- ✓ завантажити за допомогою програматора;
- ✓ налаштування друку налаштування принтера;
- ✓ розпочати висновок на друк коду програми;
- ✓ налаштування редактора;
- ✓ вихід із Arduino IDE.

Пункт меню ***Правка*** містить команди, пов'язані з редагування тексту програми, включаючи копіювання, вставку, налаштування відступів і пошук за ключовим словом.

У розділі ***Скетч*** розміщуються команди для керування компіляцією програми:

- ✓ перевірити/компілювати програму;
- ✓ показати папку скетчів відкрити системну папку із програмами;
- ✓ додати файл додати до проекту файл із даними або програмою;
- ✓ імпортувати бібліотеку підключити до програми бібліотеку зі списку встановлених.

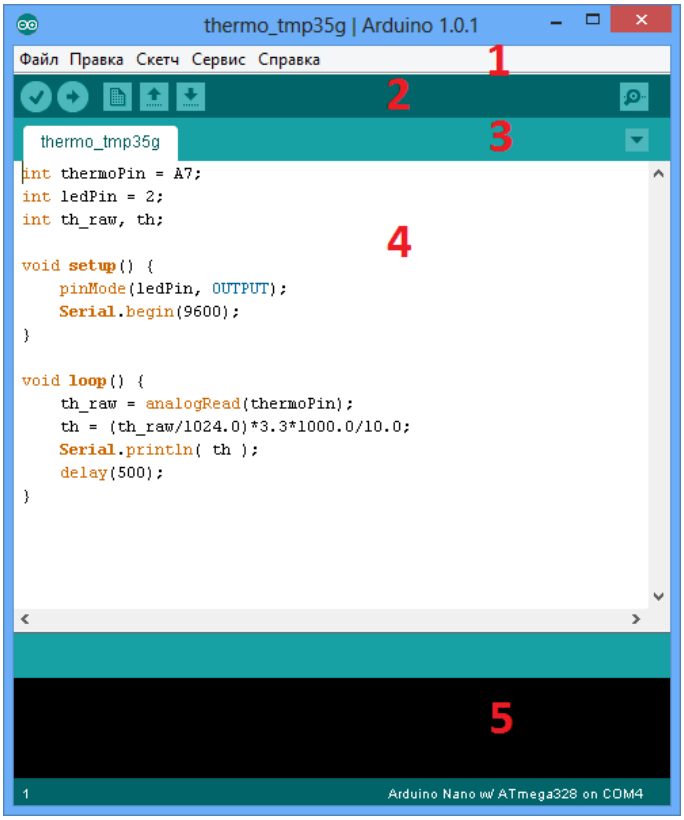


Рисунок 105. – Головне меню редактора ***Arduino***.

Пункт меню Сервіс включає допоміжні функції для роботи з самим мікроконтролером.

- автоматичне розміщення відступів, переносів рядків тощо;
- архівувати скетч архівація папки з програмою, та збереження архіву у вказане місце;
- виправити кодування та перезавантажити;
- монітор порту відкрити вікно для обміну даними з мікроконтролером;
- плата за вибір поточної плати (в даному випадку Arduino Uno);
- послідовний порт - вибір порту, до якого підключено пристрій;
- вибір програматора (не використовується в даному курсі);
- записати запис програми-завантажувача в мікроконтролер;

Меню ***Довідка*** містить докладний опис всіх функцій самого редактора Arduino IDE, а також всілякі команди та прийоми роботи з платформою Arduino.

2. Меню піктограм. На панелі інструментів розміщені наступні елементи керування:

Перевірити/Компілювати програму

Завантажити програму в Arduino

Створити нову програму

Відкрити існуючу програму

Зберегти програму

Монітор послідовного порту

3. Вкладки. Кожна програма Arduino може складатися з декількох файлів. Для перемикання між цими файлами використовується система вкладок у редакторі. Там же можна створити нову вкладку, і асоціювати з нею файл у папці з проектом.

4. Вікно програми. Безпосередньо текст програми створюється і редагується в головному вікні редактора. По суті, вікно редактора є типовим текстовим редактором, з підсвічуванням конструкцій коду.

5. Вікно повідомлень. В редакторі Arduino IDE є невелике вікно, яке служить для виведення повідомлень про помилки, що виникають у процесі компіляції програми, або під час завантаження програми в мікроконтролер.

Основи програмування Arduino. Програмування Arduino складає мові C++. Нижче наведені основні конструкції та умовності мови, необхідні для успішного проходження курсу.

1. Кожний вираз закінчується символом, крапка з комою. Наприклад: ***a = b+c;***
2. Тіло функцій та складових операторів (***if, else, for, while***) відокремлюється фігурними дужками. Наприклад:
```
if ( a>0 )
{
b = a+1;
}
```
3. Рядки відокремлюються звичайними подвійними лапками". Наприклад: ***println("some text");***
4. Символи відокремлюються одинарними лапками: ***symbol = 'a';***
5. Підключення бібліотек здійснюється за допомогою конструкції: ***#include <math.h>***
6. Кометарії у програмі починаються із символів // Як приклад: ***// це мій скетч***

Типи даних в C++ Arduino. Оголошення змінної мови c++ здійснюється за допомогою конструкції виду: ***тип_змінної ім'я_змінної;*** Приклад: ***int x, y;*** *// оголошено дві змінні x і y, які мають цілий тип*

1. Цілі числа

byte від 0 до 255
int від 32 768 до 32 767
word від 0 до 65535
long від 2 147 483 648 до 2 147 483 647

2. Дробові числа

float від 3.4028235E+38 до 3.4028235E+38
double еквівалентно float у поточній версії Arduino

3. Масиви. Масиви c++ задаються конструкцією типу:
тип_елемента ім'я_масиву [розмір];

Приклад: ***int numbers[10];*** *// задає масив з десяти цілих чисел*

4. Рядки та символи. Використовується тип даних ***char;***

Рядки в с++ є масиви з елементами типу char. Приклад:

char my_str[10]; *// рядок з десяти символів*

5. Інші типи

void порожній тип;

boolean false або ***true*** (хиба чи істина).

Оператори в С++ Arduino.

1. Оператори порівняння

== рівність

!= Нерівність

< менше

<= менше, або одно

> більше

>= більше, або одно

2. Умови

```
if (a>0) {
...
команди, за умовою істинності
}
else {
...
команди, які виконуються в іншому випадку
}
```

3. Цикли

```
for (k=0; k<3; k=k+1) {
...
команди, які виконуються на кожному кроці циклу
}
```

У дужках послідовно вказується:

- початкове значення ітератора k = 0;
- умова продовження циклу k<3 (поки що ітератор менше трьох);
- дія над ітератором під час кожного кроку k = k + 1 (збільшуємо на одиницю на кожному кроці)

Функції в С++ Arduino.

```
тип_функції імя_функції( аргументи )
{
команди, які виконуються в межах функції
return результат_функції;
}
```

де,

- ***тип_функції*** тип значення, що повертається. Наприклад, стандартна функція **sin** має тип значення **float**, що повертається.

- ***і'мя_функції*** будь-який рядок, що починається з літери, і містить лише літери та символи підкреслення.
- ***аргументи*** переліку аргументів, які функція використовує для своїх дій.
- ***результат_функції*** змінна або число, що визначає значення функції.

Нижче наведено приклад функції, що підсумовує два цілих числа.

```
int sum( int a, int b )
{ int result;
result = a+b;
return result;
}
```

Мінімальна програма. Мінімальна програма, яка може запустити на Arduino, складається з двох функцій:

```
void setup() {
}

void loop() {
}
```

Перша функція ***setup*** викликається лише один раз, після перезапуску Arduino.

Друга ***loop*** викликається нескінченну число разів під час роботи Arduino.

Приклади моделювання пристроїв на Arduino

Розглянемо приклади, з використання наступних функцій цифрового введення/виведення даних на мікроконтролері Arduino.

Приклад 1. Підключення світлодіоду на мікроконтролері Arduino та дослідження режимів його роботи.

Arduino може використовуватися як для створення автономних об'єктів, роботів, так і підключатися до програмного забезпечення, що виконується на комп'ютері. Плати мають аналогові та цифрові порти, до яких можна підключити практично будь-який простий пристрій: кнопка, датчик, двигун, екран. Для підключення світлодіода на Arduino та управління ним в роботі знадобиться:

1. *плата Arduino;*
2. *макет та 2 дроти;*
3. *світлодіод;*
4. *резистор.*

Всі комплектуючі входять до більшості початкових комплектів, їх також можна придбати окремо. Наведемо схему підключення світлодіода на Arduino в TinkerCad (рис. 106). Для роботи цієї моделі можна скористатись наступним скетчем в Arduino IDE:

```
int led = 8;
void setup()
{
pinMode(led, OUTPUT);
```

```
}
void loop()
{
digitalWrite(led, HIGH);
delay(1000);
digitalWrite(led, LOW);
delay(1000);
}
```

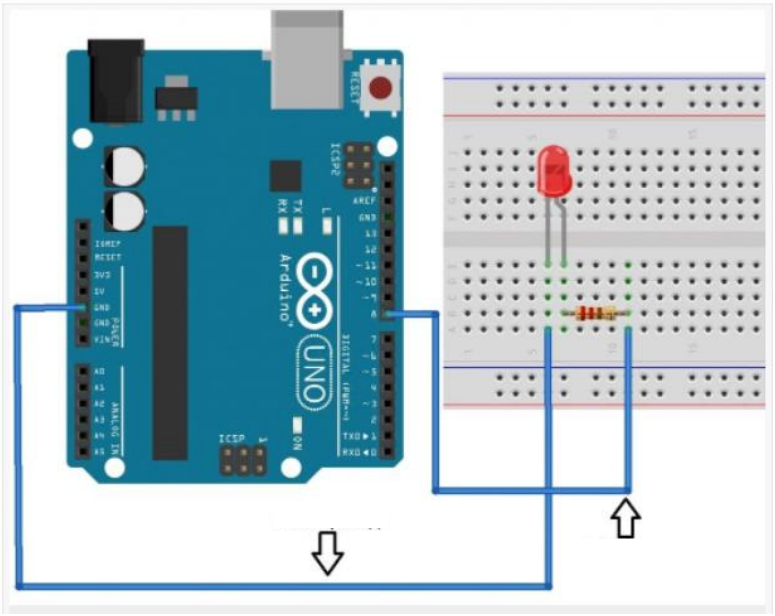


Рисунок 106. – Схема підключення світлодіода на Arduino.

Принцип роботи програми можна наводити з відповідними коментарями у виконуваному скетчі.

int led = 8; *//Оголошення змінної цілого типу, що містить номер порту якого ми підключили другий провід*

void setup() *//обов'язкова процедура setup, що запускається на початку програми; оголошення процедур починається словом void*

{

pinMode(led, OUTPUT); *// оголошення порту, led - номер порту, другий аргумент - тип використання порту - на вхід (INPUT) або на вихід (OUTPUT)*

}

void loop() *//обов'язкова процедура loop, що запускається циклічно після процедури setup*

{

digitalWrite(led, HIGH); *//Ця команда використовується для включення або вимкнення напруги на цифровому порту; led - номер порту, другий аргумент - включення (HIGH) або вимкнення (LOW)*

delay(1000); *//Ця команда використовується для очікування між діями, аргумент - час очікування в мілісекундах*

digitalWrite(led, LOW);

delay(1000);

}

Можна змінювати час очікування світлодіоду в мілісекундах та дослідити його функціонування за різними режимами роботи.

Приклад 2. Підключення до Arduino триколірного світлодіоду та дослідження режимів його роботи.

Триколірний світлодіод (rgb-led) - це три світлодіоди різних кольорів в одному корпусі. Для зборки моделі з триколірним світлодіодом використовуються наступні елементи (рис. 107):

- *плата Arduino;*
- *триколірний світлодіод.*

Рисунок 107. – Триколірний світлодіод на Arduino.

Якщо використовується світлодіод без резисторів, тоді знадобляться наступні елементи (рис. 108):

- *макет та 4 дроти;*
- *3 резистори на 220 Ом.*

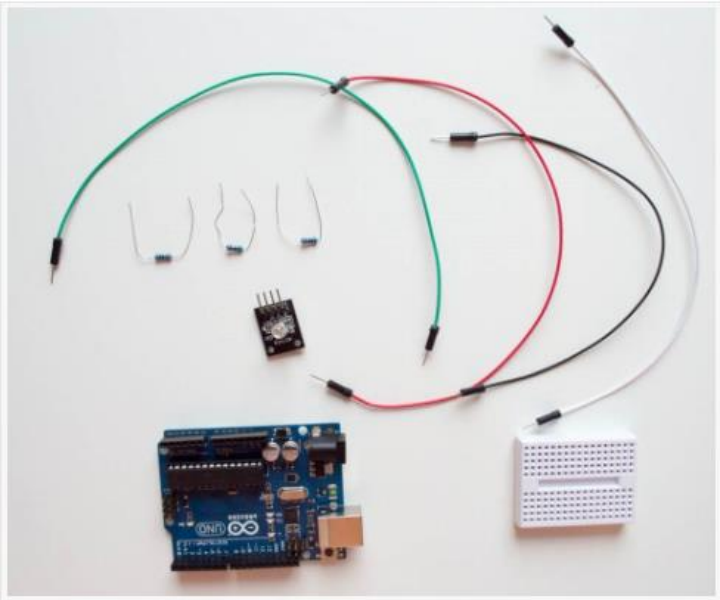


Рисунок 108. – Компоненти Arduino при реалізації світлодіодів без резисторів.

При роботі з триколірним світлодіодом без вбудованих резисторів необхідно мати на увазі, що призначення ніжки світлодіода можна визначити за її довжиною (рис. 109):

- найдовша – ***земля*** (**GND**),
- коротша – ***зелений*** (**G**),
- ще коротша – ***блакитний*** (**B**),
- найкоротша – ***червоний*** (**R**).

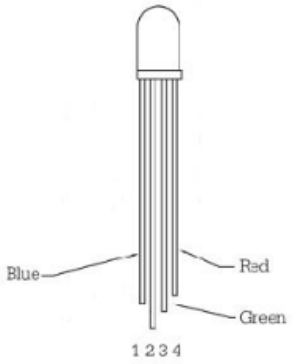


Рисунок 109. – Призначення контактів триколірного світлодіода.

Схема підключення моделі Arduino з триколірним світлодіодом із вбудованими резисторами на макетній платі наведено на рис. 110:

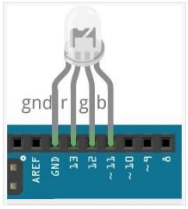


Рисунок 110. – Схема підключення триколірним світлодіодом із вбудованими резисторами

Схема підключення моделі Arduino з триколірним світлодіодом на макетній платі без вбудованих резисторів наведено на рис. 111.

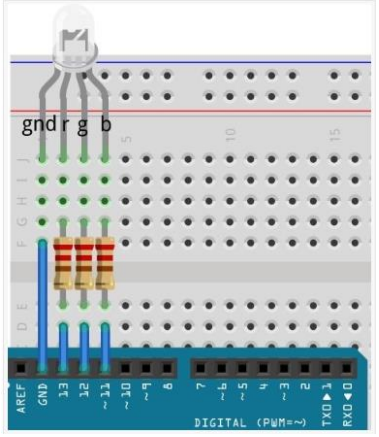


Рисунок 111. - Схема підключення триколірним світлодіодом без вбудованих резисторів

Для керування даною моделлю скористуємось наступною програмою (скетчем) в Arduino IDE:

```
//оголошуємо змінні з номерами контактів
int r = 13;
int g = 12;
int b = 11;
void setup () // Процедура setup
{
```

```
//оголошуємо порти
pinMode(r, OUTPUT);
pinMode(g, OUTPUT);
pinMode(b, OUTPUT);
}

void loop () // Процедура loop
{
digitalWrite(r, HIGH); //включаємо червоний
delay(500); / / Чекаємо 500 Мс
digitalWrite(r, LOW); //вимикаємо червоний
digitalWrite(g, HIGH); //включаємо зелений
delay(500); / / очікування 500 Мс
digitalWrite(g, LOW); //вимикаємо зелений
digitalWrite(b, HIGH); //включаємо синій
delay(500); / / оцікування 500 Мс
digitalWrite(b, LOW); //вимикаємо синій
}
```

Зібрана модель Arduino з триколірним світлодіодом без вбудованих резисторів на макетній платі наведено на рис. 112.

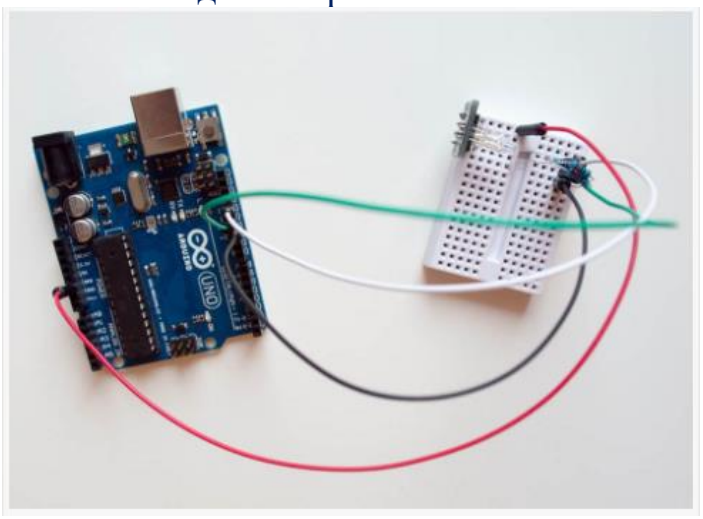


Рисунок 112. – Схема зібраної моделі Arduino із триколірним світлодіодом без вбудованих резисторів

Можна змінювати час очікування світлодіоду в мілісекундах та дослідити його функціонування за різними режимами роботи.

Приклад 3. Підключення фоторезистора (фотоелемента) до плати Arduino.

Фоторезистори використовуються в моделях як датчики освітленості. ***Фоторезистор*** - резистор, опір якого залежить від яскравості світла, що падає на нього.

У даній моделі світлодіод горить, тільки якщо яскравість світла над фоторезистором менша за задану. Яскравість можна регулювати програмно. Фоторезистори використовуються як датчики освітленості.

Вбудований фоторезистор дозволяє визначати ступінь освітленості, білі або чорні ділянки на поверхні та, у відповідності до цього, рух по лінії або інших

шляхів. Для складання даної моделі знадобляться наступні елементи (рис. 113):

- *плата Arduino та 6 проводів;*
- *фоторезистор;*
- *світлодіод4*
- *резистор на 220 Ом;*
- *резистор на 10 кОм.*

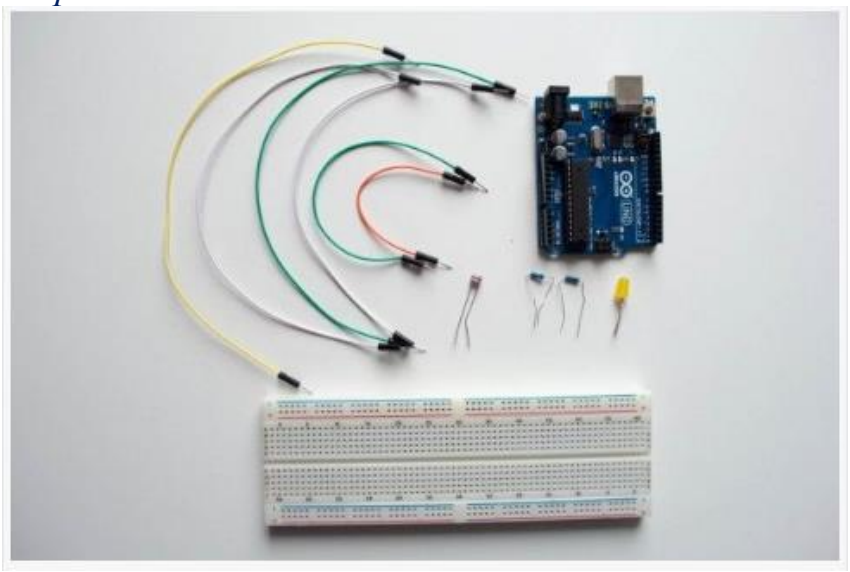


Рисунок 113. – Необхідні компоненти для підключення фоторезистора на Arduino

Схема підключення моделі Arduino з фоторезистором в TinkerCad наведена на рис. 114.

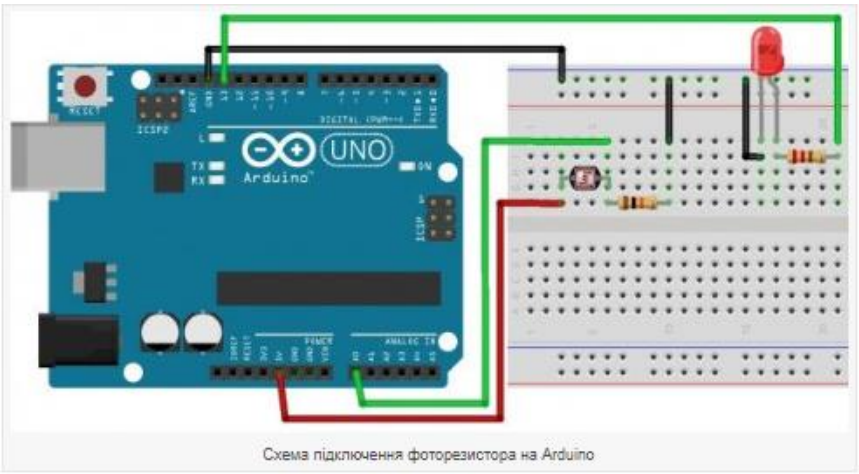


Рисунок 114. – Схема підключення фоторезистора на Arduino

Для роботи даної моделі можна скористатись наступною програмою (скетчем):

```
int led = 13; //Змінна за номером світлодіода
int ldr=0; // та фоторезистора
void setup () // Процедура setup
{
pinMode(led, OUTPUT); //Вказуємо пін вихід світлодіода
}
void loop () // Процедура loop
```

```
{
if (analogRead(ldr) < 800) digitalWrite(led, HIGH);
//якщо показник освітленості менше 800, включаємо світлодіод
else digitalWrite(led, LOW); //інакше вимикаємо
}
```

Зовнішній вигляд даної зібраної моделі з використання набору Arduino наведено на рис. 115.

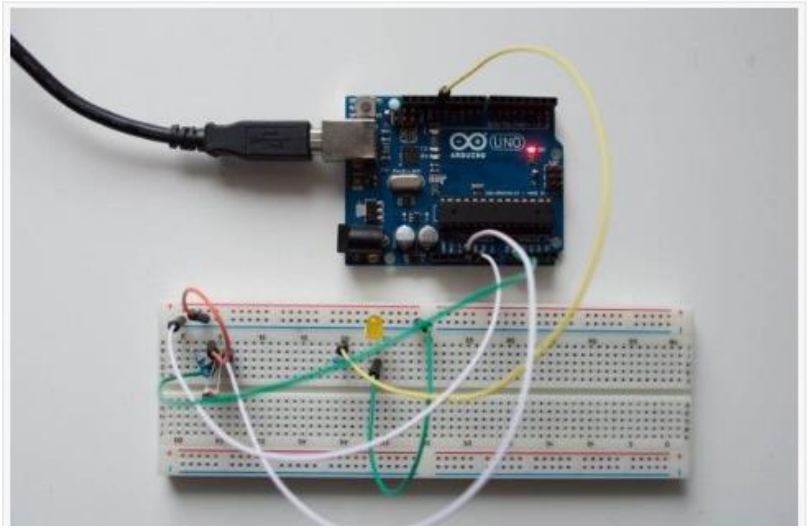


Рисунок 115. - Зібрана модель підключення фоторезистора на Arduino

Якщо світлодіод не реагує на зміну освітленості, то потрібно змінити значення на ***800*** у скетчі, якщо він постійно світиться, то значення зменшується, якщо не світиться - збільшується.

Приклад 4. Створити схему підключення світлофора на макетній платі набору Arduino. Скласти відповідний скетч його керуванням та дослідити режими його роботи.

Схему підключення світлофора на мікроконтролері Arduino Uno наведено на рис. 116

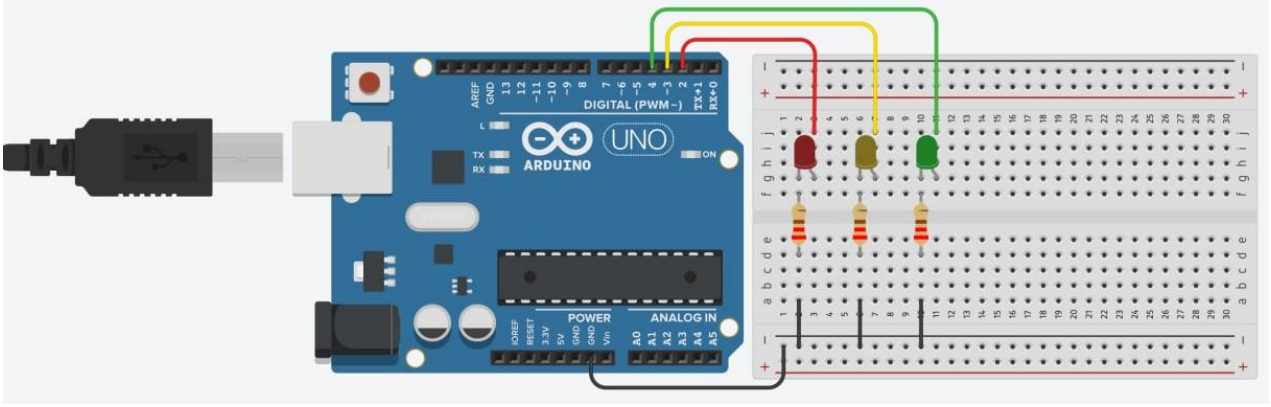


Рисунок 116. – Зібрана модель світлофора на Arduino Uno

Приклад скетчу світлофора

```
int led_red =2;
int led_yellow =3;
int led_green =4;
boolean led_on = HIGH;
void setup()
```

```
{
 pinMode(led_red, OUTPUT);
 pinMode(led_yellow, OUTPUT);
 pinMode(led_green, OUTPUT);
}
void loop()
{
 digitalWrite(led_red, HIGH);
 delay(2000);
 digitalWrite(led_red, LOW);
 digitalWrite(led_yellow, HIGH);
 delay(700);
 digitalWrite(led_yellow, LOW);
 digitalWrite(led_green, HIGH);
 delay(2000);
 for (int i=0; i<=3; i++) // зелений світлодіод миготить 3 рази
 {
 led_on = !led_on;
 digitalWrite(led_green, led_on);
 delay(500);
 }
 digitalWrite(led_green, LOW);
 digitalWrite(led_yellow, HIGH);
 delay(2000);
 digitalWrite(led_yellow, LOW);
}
```

Приклад 5. Створити схему підключення новорічної гірлянди на макетній платі набору Arduino. Скласти відповідний скетч його керуванням та дослідити режими його роботи.

Складемо програму керування новорічною гірляндою на мікроконтролері Arduino Uno.

Приклад скетчу новорічної гірлянди

```
volatile int mode = 0;  // режим роботи
const int buttonPin = 2;  // номер виходу з кнопкою
void setup()
{
 for(int i=3; i<14; i++)
   pinMode(i, OUTPUT);
 pinMode(buttonPin, INPUT);
 attachInterrupt(0, pin_ISR, RISING);
}
void pin_ISR()
```

```
{
  mode++;
  if(mode>4)
  {
    mode = 0;
  }
}
void loop()
{
  if (mode==0)
  // 0 початковий режим роботи
  // миготять одночасно всі світлодіоди
  {
    for(int i=3; i<14; i++)
      digitalWrite(i, HIGH);
      delay(500);
    for(int i=3; i<14; i++)
      digitalWrite(i, LOW);
      delay(300);
  }
  if (mode==1)
  // 1
  // рух вправо
  {
  for(int i=3; i<14; i++)
  digitalWrite(i, LOW);
  for(int i=13; i>2; i--)
  {
      digitalWrite(i, HIGH);
      delay(100);
  }
}
 if (mode==2)
 // 2
 // рух вліво
  {
    for(int i=3; i<14; i++)
    digitalWrite(i, LOW);
    for(int i=3; i<14; i++)
  {
      digitalWrite(i, HIGH);
      delay(100);
  }
}
```

```
 if (mode==3)
{
  // 3
  // random // світяться випадкові світлодіоди
  for(int i=3; i<14; i++)
    digitalWrite(i, LOW);
    for(int i=3; i<14; i++)
     if(random(128)%2==0) digitalWrite(i, HIGH);
   delay(200);
}
}
```

Схему підключення новорічної гірлянди на мікроконтролері Arduino Uno наведено на рис. 117

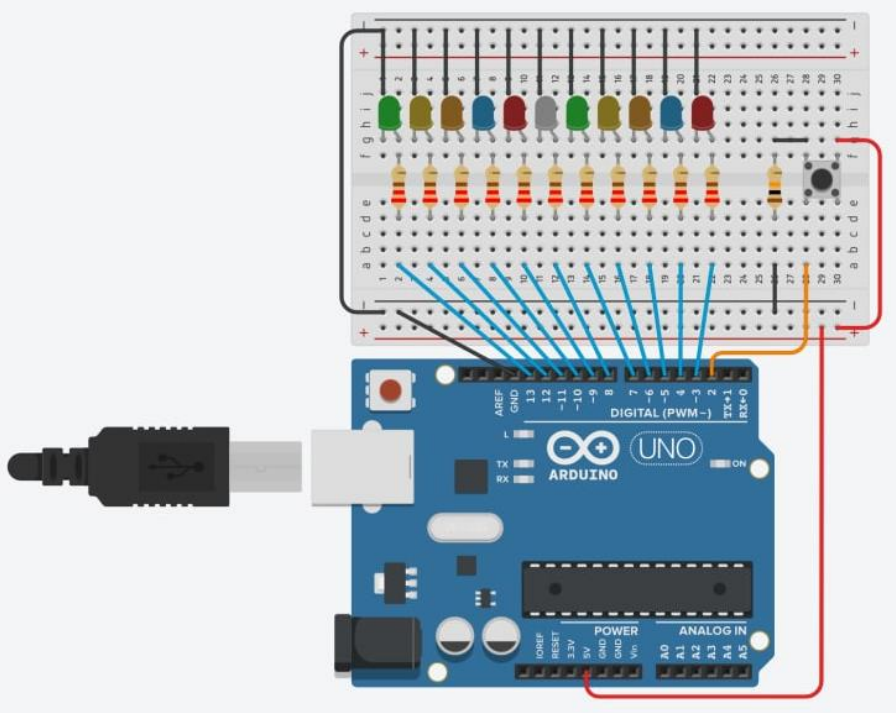


Рисунок 117. – Зібрана модель новорічної гірлянди на Arduino Uno

Контрольні запитання

1. Для чого призначена програмно-апаратна платформа Arduino?
2. Які плати Ардуіно найпоширеніші? Поясніть їх призначення.
3. Перерахуйте основні елементи плати Arduino Nano.
4. Опишіть інтерфейс середовища розробки Arduino.
5. Яка послідовність кроків при підключенні та установці Arduino Nano?
6. Як називається програма, написана для платформи Arduino?
7. Назвіть поняття функції? Тіло функції?
8. Якою є структура програми для платформи Arduino?
9. Поясніть синтаксис циклу for та while.
10. Які типи даних можна використовувати у програмі?
11. Які оператори аналізу умов застосовуються в програмування для Arduino?
12. Поясніть синтаксис оператора if – else.

Практичне завдання 8.
Вивчення принципів роботи портів мікроконтролерів Arduino Nano

Мета роботи*:** навчитись методам програмування на базі мікроконтролера ***Arduino Nano та дослідити роботу його портів вводу-виводу; дослідити роботу послідовного порту мікроконтролера Arduino з використанням класу Serial; дослідити роботу переривань, АЦП та ШІМ в міконтролерах Arduino.

Завдання 1. Зібрати схему пристрою, який наведено на рис. 118. Написати програму включення-виключення світлодіодів на виходах мікроконтролера в залежності від кількості натискань кнопки на вході. Всього натискань має бути три. При включенні світлодіоди починають миготіти із заданою частотою в порядку заданому у варіанті згідно з табл. 50. Після першого натискання кнопки частота збільшується вдвічі, після другого ще вдвічі. Після третього повертається до початкового значення. Варіанти підключення компонентів до входів-виходів наведено в табл. 50. Налагодити програму в середовищі Arduino і перевірити на макеті.

Таблиця 50. – Завдання відповідно до варіанту

№ варіанта	Завдання						
	Номера виводів плати Arduino					Початкова частота миготіння, Гц	Порядок включення світлодіодів
	Pin In	Pin Out 1	Pin Out 2	Pin Out 3	Pin Out 4		
1	7	11	6	10	9	0,5	одночасно
2	4	10	5	3	6	1	збільшення
3	12	9	3	6	11	2	зменшення
4	6	11	3	9	10	0,5	через один
5	5	10	6	8	11	1	одночасно
6	3	9	5	7	10	2	збільшення
7	8	9	10	6	11	0,5	зменшення

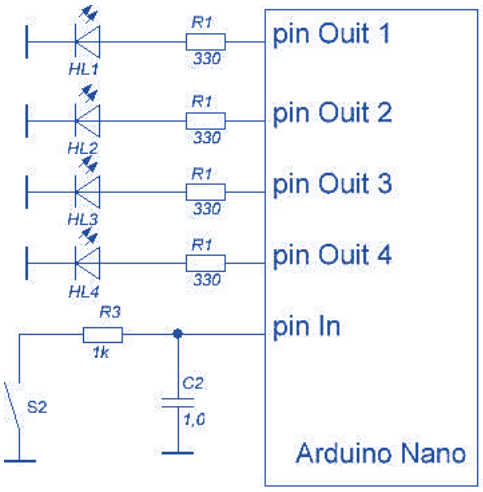


Рисунок 118. – Схема підключення світлодіодів на макетній платі Arduino

Завдання 2. Написати програму включення світлодіодів в різних режимах по команді комп'ютера. Початковий стан роботи світлодіодів контролера наведено в табл. 51. Зібрати схему, наведену на рис. 119.

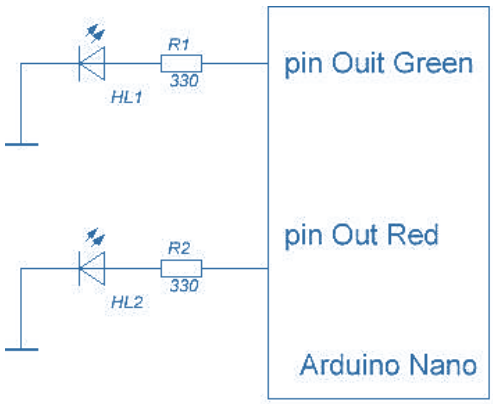


Рисунок 119. – Схема підключення на макетній платі Arduino.

Контролер Arduino повинен виконувати три команди:

1. Команду перевірки з'єднання між комп'ютером і контролером. На певний символ контролер повинен відповісти підтвердженням.
2. Команду управління світлодіодами. При отриманні якої встановлюється режим, зазначений в таблиці 4.2. Отримання команди підтвердити
3. Команду виключення світлодіодів, при отриманні якої світлодіоди вимикаються.

Команди складаються мінімум з одного символу. Індивідуальні завдання для вибору режимів роботи світлодіодів наведені в табл. 51.

Таблиця 51. – Варіанти індивідуальних завдань.

№ варіанта	*Режим до отримання команди*		*Режим після отримання команди*	
	Червоний світлодіод	*Зелений світлодіод*	*Червоний світлодіод*	*Зелений світлодіод*
1	ввімкнений постійно	мигтить	мигтить	ввімкнений постійно
2	вмикаються-вимикаються протифазно		ввімкнений постійно	мигтить
3	ввімкнений постійно	ввімкнений постійно	вмикаються-вимикаються протифазно	
4	мигтить	ввімкнений постійно	вмикаються-вимикаються одночасно	
5	вмикаються-вимикаються одночасно		ввімкнений постійно	ввімкнений постійно
6	мигтить	ввімкнений постійно	ввімкнений постійно	мигтить

Завдання 3. Написати програму завантаження 1-го байта даних в EEPROM а потім зчитування його, використовуючи окремі паролі на запис та зчитування. Зібрати схему, яку наведено на рис. 120. При написанні скетча необхідно врахувати:

1. Після перезавантаження плати світиться червоний світлодіод. Мікроконтролер отримує від комп'ютера пароль і команду на завантаження байта даних в EEPROM одним рядком. Для поділу пароля, команди і значення байта використовувати символ '/' або '\'. Для позначення закінчення повідомлення використовувати символ ‘?’.

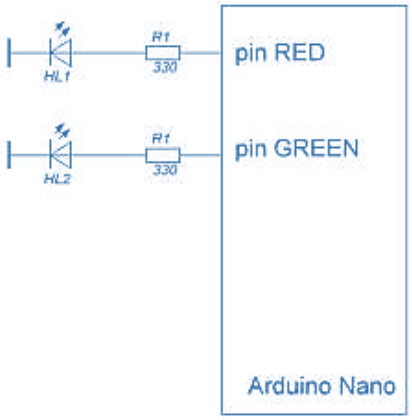


Рисунок 120. – Схема підключення на макетній платі Arduino.

2. Виконується верифікацію пароля.
3. Якщо пароль не підходить, то відсилається інформація про неправильний пароль і червоний світлодіод мигає декілька разів.
4. Якщо пароль збігається, то червоний світлодіод гасне, а зелений включається на 0,5 – 1,5 с, значення байта зчитується та записується в EEPROM.
5. Аналогічно для зчитування виконуються пункти 2-4 та використовується інший пароль. У результаті байт зчитується з EEPROM та передається в комп'ютер з супровідним повідомленням.

Запит (пароль) – ім'я студента або інше слово латиницею. Варіанти підключення світлодіодів наведено в табл. 52.

Таблиця 52. – Варіанти індивідуальних завдань

№ варіанта	*Завдання*	
	Pin RED	*Pin GREEN*
1	2	6
2	3	7
3	4	8
4	5	9
5	6	10
6	7	11

Завдання 4. Написати програму зміни довжини імпульсів на виході ШІМ за допомогою переривань і вимірювання середнього значення цієї напруги. Варіанти підключення входів-виходів взяти з табл. 53. Зібрати схему зображену на рис. 121.

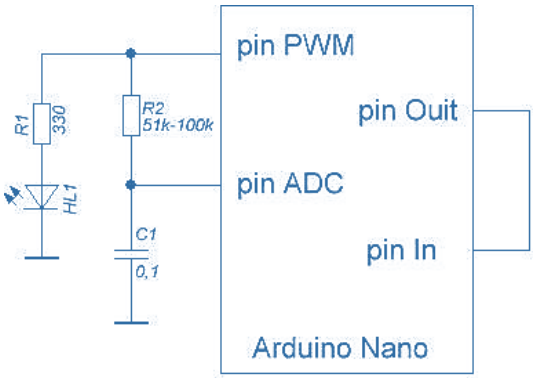


Рисунок 121. – Схема підключення на макетній платі Arduino.

Написати програму зміни довжини імпульсів на виході ШІМ за допомогою переривань і вимірювання середнього значення цієї напруги. Варіанти підключення входів-виходів взяти з табл. 53.

Таблиця 53. – Варіанти підключень входів-виходів

№ варіанта	*Виводи*				
	Pin PWM	*Pin ADC*	*Pin In*	*Pin Out*	*Тип переривання*
1	3	A1	int 0 (вихід 2)	13	LOW
2	5	A2	int 0 (вихід 2)	12	CHANGE
3	6	A3	int 0 (вихід 2)	11	RISING
4	9	A4	int 1 (вихід 3)	5	FALLING
5	10	A5	int 1 (вихід 3)	6	LOW
6	11	A6	int 1 (вихід 3)	7	FALLING

Зміст звіту

1. Титульна сторінка, тема та мета практичного заняття.
2. Подати зовнішній вигляд модулів підключених схем в ArduinoUno.
3. Тексти скетчів відповідних модулів.
4. Вивести дані моделювання досліджених режимів роботи.
5. Висновки, що відображають результати виконання роботи та їх експериментальний аналіз.

Теоретичні відомості.

Вивчення роботи портів введення-виведення плати Arduino. Наведемо основні функції цифрового введення / виведення:

1. ***pinMode (pin, mode):***
   Параметри: *pin:* номер виводу, режим роботи якого буде конфігуруватися; *mode*: приймає значення INPUT, OUTPUT або INPUT_PULLUP. Значень немає, які повинні повертатись після виконання.
2. ***digitalWrite (pin, value):***
   Параметри: pin: номер виводу; value: значення HIGH або LOW. Значень немає, які повинні повертатись після виконання.
3. ***digitalRead (pin):***
   Параметри: pin: номер цифрового виводу, з якого необхідно вважати значення. Значення, які повертаються HIGH або LOW.

Необхідно зібрати схему пристрою, який наведено на рис. 118. Написати програму включення-виключення світлодіодів на виходах мікроконтролера в залежності від кількості натискань кнопки на вході. Всього натискань має бути три. При включенні світлодіоди починають миготіти із заданою частотою Після першого натискання кнопки частота збільшується вдвічі, після другого ще вдвічі. Після третього повертається до початкового значення.

Приклад скетча, який задає конфігурацію виводів на платі має наступний вигляд:

```
int switchPin = 5;
int ledPin = 9;
```

```
void setup()
{
pinMode(switchPin, INPUT);
pinMode(ledPin, OUTPUT);
}
void loop()// Основний цикл програми
{
if(digitalRead(switchPin) == HIGH) digitalWrite(ledPin, LOW);
else
}
digitalWrite(ledPin, HIGH);
```

Надалі необхідно підключити зібрану схему до комп'ютера через USB порт плати Arduino. Завантажити програму в мікроконтролер Arduino та перевірити правильність роботи пристрою.

Вивчення роботи послідовного порту плати Arduino. Клас Serial використовується для зв'язку плати Arduino з комп'ютером або іншими пристроями. Плата Arduino Nano має один послідовний порт, також відомий як UART або USART. Він пов'язаний з цифровими виводами 0(RX) і 1 (TX), і використовується для зв'язку з комп'ютером через USB.Таким чином, під час використання послідовного порту, виводи 0 і 1 не можуть використовуватися в якості цифрових входів або виходів.

Для зв'язку з Arduino можна використовувати спеціальну програму моніторингу послідовного порту, вбудовану в програмне забезпеченняArduino. Для виклику програми натисніть відповідну кнопку на панелі інструментів і встановіть ту ж швидкість передачі, що вказується в програмі при виклику методу *begin ().*

Перелік функцій для роботи з послідовним портом плати Arduino:

1. Serial.begin(speed) Serial.begin(speed, config)

- *Параметри*:
  speed: швидкість в бітах на секунду (бодах) – long;
  ***config*:** задає кількість біт данних, перевірку чотності и стопові біти.
- *Опис*: задає швидкість передачі даних по послідовному інтерфейсу в бітах в секунду (бодах).

Для взаємодії з комп'ютером слід використовувати одну з попередньо встановлених швидкостей обміну: 300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 або 115200.

Другий аргумент цієї функції необов'язковий. Він дозволяє налаштувати кількість біт даних, перевірку парності і степових біти. За замовчуванням, посилка складається з 8 біт даних, без перевірки парності, з одним стоповим бітом.

2. Serial.available()

- *Параметри* – немає.
- *Значення*, які повертаються: кількість байт, доступних для зчитування.
- *Опис*: повертає кількість байт (символів) доступних для зчитування з

буфера послідовного порту. Під символами розуміються дані, які вже прийняті і зберігаються в послідовному приймальному буфері (який може зберігати максимум 64 байта).

3. Serial.read()

- *Параметри* – немає.
- *Значення* int, які повертаються: перший байт прийнятих даних (або -1, якщо таких немає)
- *Опис*: зчитує дані, що надходять по послідовному інтерфейсу.

4. Serial.print(val) Serial.print(val, format)

- *Параметри*:
  ***val*:** значення, яке необхідно вивести – будь-який тип даних;
  ***format*:** визначає систему числення (для цілочисельних типів), а також кількість десяткових знаків після коми (для чисел з плаваючою точкою).
- *Значення*, які повертаються: кількість виведених байт. Зчитування цього значення не обов'язково.
- *Опис*: функція виводить через послідовний порт заданий ASCII текст у вигляді, зрозумілому для людини. Ця команда може мати кілька різних форм.При виведенні числа кожній його цифрі відповідає один ASCII-символ. Дробові числа теж виводяться у вигляді ASCII-цифр, при цьому після коми за замовчуванням залишається два десяткових знака. Байти виводяться у вигляді окремих символів, а символи і рядки виводяться без змін.

Наведемо ***приклад скетчу для роботи з послідовним портом***.

```
int incomingByte = 0;
void setup()
{
    Serial.begin(9600);
}
void loop()
 {
    if (Serial.available() > 0) { incomingByte = Serial.read();
    Serial.print("I received: ");
    Serial.println(incomingByte, DEC);}
}
```

В даному випадку, мікроконтролер отримує команду від комп'ютера та виконує аналіз команди. В залежності від команди відсилає символ підтвердження, змінює алгоритм роботи світлодіодів або вимикає їх.

Надалі необхідно підключити зібрану схему до комп’ютера через USB порт плати Arduino. Завантажити програму в мікроконтролер Arduino та перевірити правильність роботи пристрою.

Робота з рядками в програмованих мікроконтролерах Arduino. Клас String дозволяє здійснювати більш складну обробку текстових рядків, ніж звичайні масиви символів. За допомогою класу String можна об’єднувати і

розширювати рядки, здійснювати пошук і заміну підрядків, і багато іншого.

На платі Arduino є 6-канальний (8-канальний – в Mini і Nano, 16 – в Mega) 10-бітний аналогово-цифровий перетворювач, який перетворює вхіднунапругу з діапазону 0 – 5 В у цілочисельні значення в межах від 0 до 1023 відповідно. Роздільна здатність АЦП становить: 5 В / 1024 значення або 0.0049 В (4.9 мВ) на одне значення. Вхідний діапазон і роздільна здатність можуть змінюватися за допомогою функції *analogReference ()*.

Для зчитування значення з аналогового входу потрібно близько 100 мікросекунд (0.0001 с), тому максимальна частота опитування виведення приблизно дорівнює 10 000 разів на секунду. Якщо аналоговий вхід ні до чого не підключений, значення, що повертається функцією *analogRead(),* буде випадковим. Змінюватиметься під впливом декількох факторів: величина напруги на інших аналогових входах, наведення від руки поблизу плати і т. д.

Перелік функцій Arduino для роботи з рядками:

1. String (val) String (val, base)

- *Параметри*:
  val: змінна, значення якої необхідно представити у вигляді об'єкта String - string, char, byte, int, long, unsigned int, unsigned long;
  base (не обов'язково параметра) - основа системи числення, в якому необхідно представити ціле число.
- *Значення*, які повертаються – відсутні.
- *Опис*: створює екземпляр класу String. Існує кілька різних версій цієї функції, які створюють об'єкт String з різних типів даних (іншими словами, формують рядок з послідовності символів). При створенні об'єкта String з числа, результуючий рядок буде містити ASCII-представлення цього числа.За замовчуванням вважається, що число зазначено в десятковій системі, тому:

2. String.indexOf (val) String.indexOf (val, from)

- Параметри:
  string: змінна типу String;
  val: шукане значення - char або String;
  from: початкова позиція для пошуку.
- *Значення*, які повертаються: індекс підрядка *val* в рядку String, або -1, якщо підрядок не знайдено.
- *Опис*: здійснює пошук символу або підрядка в рядку (String). За замовчуванням, пошук здійснюється з початку рядка, однак можна вказати і певну позицію, з якою необхідно почати пошук. Така можливість дозволяє знаходити всі входження підрядка в рядку.

Функції Arduino для запису/зчитування інформації з EEPROM:

1. read(). Зчитує байт з EEPROM. Ті байти, які ніколи не піддавалися записи, мають значення 255. Синтаксис:

EEPROM.read (address)

- *Параметри*: address – адреса байта, значення якого необхідно зчитати (int).

- *Значення*, які повертаються: значення зазначеного байта (byte).

2. write (). Записує байт в EEPROM. Синтаксис:

EEPROM.write (address, value)

- *Параметри*: ***address*** - адреса байта (нумерація з 0), значення якого необхідно записати (int); ***value*** - записується значення, від 0 до 255 (byte).
- Значення, які повертаються відсутні.

Надалі необхідно підключити зібрану схему до живлення (5В) через USB порт комп'ютера та плати Arduino. Завантажити програму в програмований мікроконтролер Arduino та перевірити правильність роботи пристрою.

Вивчення роботи переривань, ШІМ та АЦП програмованого мікроконтролера Arduino. Переривання – це сигнали, що переривають нормальний перебіг програми. Вони зазвичай використовуються для апаратних пристроїв, що вимагають негайної реакції на появу подій. Наприклад, послідовний порт або UART (універсальний асинхронний приймач) мікроконтролера повинен бути обслужений при отриманні нового символу. Якщо цього не зробити до приходу наступного символу, то даний символ може бути втрачений.

При надходженні нового символу UART генерує переривання. Мікроконтролер зупиняє виконання основної програми і переходить на програму обробки переривань (ISR), призначену для даного переривання. В даному випадку це переривання за отриманим символу. Програма обробки ISR зчитує символ з UART, поміщає в буфер, потім очищає прапор переривання і виконує повернення в основну програму, продовжуючи її з точки виклику. Все це відбувається у фоновому режимі і не впливає безпосередньо на основний код роботи програми.

Якщо в процесі роботи запускається багато переривань абопереривання генерує швидкодіючий таймер, основна програма буде виконуватися повільніше, так як мікроконтролер розподіляє свій машинний час між основною програмою і всіма функціями обробки переривань.

Arduino надає функції для роботи з перериваннями. Ці функції оголошені у файлі: *\ Hardware \ cores \ arduino \ wiring.h*

і реалізовані в файлі:

\ Hardware \ cores \ arduino \ WInterrupts.c

Розглянемо функції для роботи з перериванням:

1. *attachInterrupt (interrupt, function, mode)*

- *Параметри*:

interrupt*:* номер переривання (int). У більшості плат Arduino існує двазовнішніх переривання: номер 0 (цифровий вихід 2) і 1 (цифровий вихід 3).

function: функція, яку необхідно викликати при виникненні переривання; цяфункція повинна бути без параметрів і не повертати ніяких значень. Таку функцію іноді називають оброблювачем переривання.

mode: визначає умову, за якої має спрацьовувати переривання.
Може приймати одне з чотирьох визначених значень:

- ***LOW*** - переривання буде спрацьовувати щоразу, коли на виводі присутнійнизький рівень сигналу;
- ***CHANGE*** - переривання буде спрацьовувати щоразу, коли змінюється станвиводу;
- ***RISING*** - переривання спрацює, коли стан виводу зміниться з низького рівняна високий;
- ***FALLING*** - переривання спрацює, коли стан виводу зміниться з високогорівня на низький.
- *Опис:* ***attachInterrupt*** задає функцію обробки зовнішнього переривання, тобто функцію, яка буде викликана за зовнішніми перериваннями. Якщо до цього була задана інша функція, то призначається інша.

2. detachInterrupt (interrupt)

- *Параметри*: ***interrupt***: номер переривання **(0 або 1).**
- *Опис*: вимикає обробку зовнішнього переривання.

Розглянемо **Команди для роботи з АЦП та ШІМ:**

1. analogReference (type)

- *Параметри*: ***type*** – тип джерела опорної напруги (DEFAULT,INTERNAL, INTERNAL1V1, INTERNAL2V56 або EXTERNAL).
- *Значення*, які повертаються відсутні.
- *Опис*: встановлює джерело опорної напруги, що використовується при зчитуванні аналогового сигналу (іншими словами, задає максимальне значення вхідного діапазону). Для вибору джерела опорної напруги доступні наступні значення:
  - ✓ ***DEFAULT:*** опорна напруга за замовчуванням, рівна 5В (на 5В-платах Arduino) або 3.3В (на 3.3В-платах Arduino).
  - ✓ ***INTERNAL***: внутрішньо опорна напруга, рівна 1.1В мікроконтролера ATmega168 і ATmega328, або 2.56В в мікроконтролері ATmega8 (недоступно в Arduino Mega).
  - ✓ ***INTERNAL1V1:*** внутрішня опорна напруга 1.1В (тільки для Arduino Mega). *INTERNAL2V56:* внутрішня опорна напруга 2.56В (тільки для Arduino Mega).
  - ✓ ***EXTERNAL:*** як опорна напруги буде використовуватися напруга, прикладенадо виводу AREF (від 0 до 5В).

2. analogRead (pin)

- *Параметри*: ***pin***: номер виводу, з якого буде зчитуватися напруга (0 - 5 для більшості плат,0 - 7 для Mini і Nano, 0 - 15 для Mega).
- *Значення*, які повертаються: ціле число int (від 0 до 1023).
- *Опис*: зчитує величину напруги з зазначеного аналогового виводу.

3. analogWrite (pin, value)

- *Параметри*: ***pin*** – вивід, на якому буде формуватися напругу (ШІМ виходи 3, 5, 6, 9, 10 і 11);

- ***value*** – коефіцієнт заповнення – міститься в межах від 0 (завжди вимкнений) до255 (завжди включений).
- *Значення*, які повертаються відсутні.
- *Опис*: формує задану аналогову напругу на виводі у вигляді ШІМ-сигналу. Може використовуватися для варіювання яскравості світіння світлодіода або управління швидкістю обертання двигуна. Після виклику *analogWrite*(), на виводі буде безперервно генеруватися ШІМ-сигнал з заданим коефіцієнтом заповнення до наступного виклику функції *analogWrite()* (або до моменту виклику *digitalRead()* або *digitalWrite()*, взаємодіючих з цим же виводом).

Наведемо ***приклад скетчу*** по обробці переривань:

```
// світлодіод, підключений до digital pin 13 буде змінювати свій
// стан при зміні напруги на digital pin 2int pin = 13;
volatile int state = LOW;
void setup(){
      pinMode(pin, OUTPUT); // порт виходу
      attachInterrupt(0, blink, CHANGE); // прив'язуємо 0-е переривання
                                          до функції blink().
}
void loop(){
digitalWrite(pin, state); // виводимо state
}
void blink(){
state = !state;// міняємо значення на протилежне
}
```

Розглянемо алгоритм роботи наведеного скетчу:

1. При ініціалізації виставити на виході Pin Out рівень сигналу, необхідний для роботи по перериванню. Наприклад, HIGH, якщо використовується переривання LOW.
2. З комп’ютера передається команда та значення ШІМ (0-255) у вигляді рядка.
3. Після передачі команди з комп’ютера на виході PinOut встановлюється рівень сигналу, який запускає переривання.
4. У перериванні необхідно встановити ШІМ сигнал на виході PinPWM з прийнятим коефіцієнтом та виставити початковий рівень сигналу на виході PinOut.
5. Через 200 мс після закінчення обробки переривання АЦП зчитує напругу з конденсатора C1 (діапазон значень 0-1023).
6. Код з АЦП перераховується у вольти та у вигляді рядка: "Voltage: X.XX v" передається в комп’ютер.

Надалі необхідно підключити зібрану схему до живлення (5В) через USB порт комп’ютера та плати Arduino. Завантажити програму в програмований мікроконтролер Arduino та перевірити правильність роботи пристрою.

Контрольні питання

1. Зазначте основні сфери застосування Arduino?
2. Які є основні функції цифрового введення та виведення?
3. Які значення має параметр value функції *digitalWrite ()*?
4. Чи повертає значення функція *pinMode (pin, mode)*?
5. Описати алгоритм ініціалізації порту вводу-виводу Arduino?
6. Який клас та функції застосовують для роботи з послідовним портом плати Arduino?
7. Опишіть основні особливості функції *Serial.available()*.
8. Зазначте параметри функції *Serial.print( )*.
9. Які команди виконує контролер Arduino згідно з завданням?
10. Які дві характеристики має функція *Serial.begin( )*?
11. Описати основні характеристики класу String.
12. Які існують варіанти створення рядка в мові Arduino?
13. Які особливості має функція String.indexOf ( )?
14. Які є основні функції Arduino для запису/зчитування інформації з EEPROM та описати їх характеристики.
15. Що таке переривання?
16. Які значення може видавати функція attachInterrupt ()?
17. Чи повертаються будь-які значення при роботі з функцією analogReference()?
18. Зазначте основні команди для роботи з АЦП та ШІМ?
19. Опишіть, коротко, алгоритм роботи програми.

Список використаних джерел

Базова

1. Жабін В.І., Цифрові автомати. Практикум / Жабін В.І. – К.: ВЕК +, 2004. – 160 с.
2. Жабін В.І., Комп'ютерна логіка: [навч. посібник] / Жабін В.І., Жуков І.А., Клименко І.А., Ткаченко В.В. – К.: Книжкове вид-во НАУ, 2007. – 364 с.
3. Локазюк В. М. Мікропроцесори та мікроЕОМ у виробничих системах. Навчальний посібник для вузів. –Хмельницький: ТУП, 2001. – 244с.
4. Рябенький В.М., Жуйков В.Я., Гулий В.Д. Цифрова схемотехніка: Навч. посібник. – Львів: "Новий світ-2000", 2009. – 736 с.
5. Бойко В. І., Схемотехніка електронних систем: У 3 кн. Кн.3. Мікропроцесори та мікроконтролери: Підручник/ В. І. Бойко, А. М. Гуржій, В. Я. Жуйков, ін.
6. Цирульник С. М. Проектування мікропроцесорних систем/ С.М. Цирульник, Г.Л. Лисенко. – Вінниця: ВНТУ, 2012. – 191с. – ISBN 978-966-641-463-5
7. Бойко В.І., Цифрова схемотехніка електронних систем. Підручник / В.І., Бойко, В.Я. Жуйков, А.А. Зорі, В.В. Багрій, В.М. Співак, Т.О. Терещенко. 3-те вид. допов. і переробл.– К.: Вища школа, 2010. – 426 с. (гриф надано МОН України – протокол № 1.4/18-Г414 від 14.02.2008 р.). ISBN 966-642-193-3.
8. Чумаченко І.В., Кошовий М.Д., Лопатин В.В. Мікроконтролерні прилади: структура і використання: Навчальний посібник. – Харків: Нац. аерокосмічний ун-т «ХАІ», 2001. –277с.

Додаткова

9. AT90S2313 Embedded Microcontroller User's Manual. Atmel Corp. 1999.
10. Evans B. Arduino programming notebook. URL: http://engineering.nyu.edu/ gk12/ampscbri/pdf/ArduinoBooks/Ardui no%20Programming%20Notebook.pdf
11. Libmodbus documentation. URL: https://libmodbus.org/ documentation/.
12. OpenCV documentation. URL: https://docs.opencv. org/3.4.16/

Інформаційні ресурси

13. Arduino docs | Arduino documentation. URL: https://docs.arduino.cc/
14. Основи мікропроцесорної техніки: [електронний ресурс]. Режим доступу: http://dn.dut.edu.ua/course/MP (дата звернення 25.01.2022). – Назва з екрана.
15. Logic.Ly. Створення логічних схем. Режим доступу: https://logic.ly/
16. ***EasyEDA.*** Інструмент для створення та симуляції схем и плат. Режим доступу: https://easyeda.com
17. Офіційний сайт Arduino [Електронний ресурс]. – Режим доступу: https://www.arduino.cc.
18. Arduino.ua Плати Arduino Nano [Електронний ресурс]. - Режим доступу: https://doc.arduino.ua/ru/hardware/Nano
19. ServoAttach Програмування в Україні. URL: https://doc.arduino.ua/prog/Servo (дата звернення: 16.02.2022).
20. Arduino - ToneMelody. URL: https://www.arduino.cc/en/Tutorial/toneMelody (дата звернення: 16.03.2022).