

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій
(повне найменування інституту, назва факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ГІБРИДНОЇ ЛАБОРАТОРІЇ GOLDi
GOLDi HYBRID LABORATORY SOFTWARE

Виконав: студент(ка) 4 курсу, групи КНТ-218сп
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні науки

Медков В.М.

(прізвище та ініціали)

Керівник Гладкова О.М.

(прізвище та ініціали)

Рецензент Терещенко Е.В.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ІРЕ, ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
« » 2021 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Медкова Всеволода Максимовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення гібридної лабораторії GOLDi, GOLDi hybrid laboratory software

керівник проєкту (роботи) Гладкова Ольга Миколаївна, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від «30» березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) 25 травня 2021 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Дослідження особливостей гібридної лабораторії GOLDi. 3. Програмна реалізація віддаленого експерименту на Arduino. 4. Керівництво користувача.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1–4 Основна частина	Гладкова О.М., доцент		
Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання “28” лютого 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2–3 тижні	Розділ 1
3	Розробка архітектури системи	4 тиждень	Розділ 2
4	Розробка програми	5–6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 3
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Розділ 4, Додатки
7	Захист роботи	9 тиждень	

Студент(ка)

(підпис)

Медков В.М.

(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис)

Гладкова О.М.

(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 95 с., 73 рис., 5 табл., 2 дод., 12 джерел.

ДИСТАНЦІЙНЕ НАВЧАННЯ, ГІБРИДНА ЛАБОРАТОРІЯ, ФІЗИЧНІ ЕКСПЕРИМЕНТИ, ARDUINO, СЕРЕДОВИЩЕ ПРОГРАМУВАННЯ, ФУНКЦІЯ.

Об'єкт дослідження – реалізація експериментів гібридної лабораторії GOLDi.

Предмет дослідження – програмне та апаратне забезпечення гібридної лабораторії GOLDi.

Мета роботи – розробка програмного забезпечення гібридної лабораторії GOLDi для дистанційного управління експериментами за допомогою мікроконтролера Arduino.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, мова програмування C++, персональний комп'ютер з процесором Intel Core 2 Duo під управлінням операційної системи Microsoft Windows 7, плата Arduino, Arduino IDE.

Результати. Розроблено програмне забезпечення для управління експериментом 3-Axis-Portal з використанням мікроконтролера Arduino.

Висновки. Розроблене програмне забезпечення розширює можливості дистанційного управління експериментами гібридної лабораторії GOLDi.

Галузь використання – гібридна лабораторія GOLDi.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 95 pages, 73 figures, 5 tables, 2 appendixes, 12 sources.

DISTANCE LEARNING, HYBRID LABORATORY, PHYSICAL EXPERIMENTS, ARDUINO, PROGRAMMING ENVIRONMENT, FUNCTION.

Object of study is the implementation of experiments of the GOLDi hybrid laboratory.

Subject of research are the software and hardware of the GOLDi hybrid laboratory.

The goal of the work is to develop software for the GOLDi hybrid laboratory for remote control of experiments using Arduino microcontroller.

Materials, methods and technical tools: structural and object-oriented programming, C ++ programming language, personal computer with Intel Core 2 Duo processor running Microsoft Windows 7 operating system, Arduino board, Arduino IDE.

Results. Software has been developed to control the 3-Axis-Portal experiment using an Arduino microcontroller.

Conclusions. The developed software expands the possibilities of remote control of GOLDi hybrid laboratory experiments.

Field of application is GOLDi hybrid laboratory.

ЗМІСТ

С.

Перелік скорочень та умовних позначок.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Вбудовані системи.....	10
1.2 Огляд платформ для вбудованих систем.....	12
1.2.1 Freeduino.....	12
1.2.2 Raspberry Pi.....	14
1.3 Аналіз платформи Arduino.....	20
1.3.1 Загальна характеристика.....	20
1.3.2 Огляд та порівняння існуючих платформ Arduino.....	24
2 Дослідження особливостей гібридної лабораторії GOLDi.....	32
2.1 Структура мережі GOLDi.....	32
2.2 Структура лабораторії.....	35
2.3 Приклади використання лабораторії.....	39
2.3.1 Експеримент 3-Axis-Portal.....	39
2.3.2 Експеримент Elevator B.....	40
2.3.3 Експеримент Production Cell.....	41
2.4 Методика використання гібридної лабораторії.....	43
2.4.1 Як користуватися ЕСР.....	43
2.4.2 Ручне управління.....	49
2.4.3 Використання Digital Demo Board.....	51
2.4.4 Програмований логічний пристрій.....	55
2.4.5 Скінчений автомат.....	62
2.5 Постановка задач дипломної роботи.....	68
3 Програмна реалізація віддаленого експерименту на Arduino.....	70
3.1 Алгоритм експерименту.....	70
3.2 Вибір інструментарію розробки.....	71

3.2.1 Arduino IDE.....	71
3.2.2 Visual Micro.....	72
3.2.3 Pluto.....	74
3.3 Реалізація програмного забезпечення.....	74
4 Керівництво користувача.....	78
Висновки.....	82
Перелік джерел посилання.....	83
Додаток А Текст програми.....	84
Додаток Б Слайди презентації.....	91

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ARM – Advanced RISC Machine;

CPLD – Complex Programmable Logic Device;

FTDI – Future Tehnology Device International;

HDMI – High Definition Multimedia Interface;

SoC – System-on-a-chip;

UART – Universal asynchronous receiver/transmitter;

ЕОМ – електронна обчислювальна машина;

ПЛІС – програмована логічна інтегральна схема;

ШІМ – широтно-імпульсна модуляція.

ВСТУП

Гібридна лабораторія GOLDi розроблена для того щоб, студенти та викладачі могли дистанційно виконувати та проводити лабораторні та практичні роботи.

Під час карантину цеє актуальним, оскільки можна працювати дистанційно, заходячи на сайт GOLDi. Користувачі можуть досліджувати різноманітні фізичні системи та керувати ними на відстані за допомогою різних пристроїв управління. У GOLDi присутні пристрої управління (Control unit), які вибираються для управління фізичними системами, що виконують написаний код, або команди, які їм прописують. Це дуже зручно, адже кожен студент та викладач може виконувати роботу з тією фізичною системою і пристроями управління, які вони оберуть, згідно з чергою.

Але до одного і того ж фізичного пристрою не може підключитися декілька користувачів, і для цього в GOLDi присутнє часове обмеження на виконання експериментів, що дозволяє організувати роботу системи, для використання декількома користувачами. Якщо не виходить отримати доступ до фізичної системи, то кожен може без черги підключитися до віртуальної системи, що дуже зручно і швидко.

Мета роботи – розробка програмного забезпечення гібридної лабораторії GOLDi для дистанційного управління експериментами за допомогою мікроконтролера Arduino.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Вбудовані системи

Вбудовувані системи (Embedded Systems) – це результат міждисциплінарного проектування, в якому умовно можна виділити три основних складових:

- етап вирішення завдання на прикладному рівні, коли необхідно знайти правильні методи та алгоритми без деталей реалізації. Це сфера діяльності прикладних фахівців з відповідних областей (фізика, енергетика, медицина, лінгвістика, біологія та ін.);

- процес програмування, в ході якого потрібно відобразити отримане прикладне рішення на технологічну базу інформатики та обчислювальної техніки. Це робота фахівців з області інформатики, сьогодні все частіше її називають архітектурним, високо рівними або системним проектуванням;

- фаза реалізації, в ході якої інженери, програмісти та прикладні фахівці забезпечують виконання раніше сформульованих вимог, таких як необхідна функціональність, динаміка поведінки, надійність і безпека функціонування, габарити, енергоспоживання, вартість і технологічність при тиражуванні.

Перші дослідження використання цифрових обчислювальних машин для управління фізичними об'єктами почалися в 60-х роках ХХ ст [1].

У наступному десятилітті ЕВМ починають активно застосовуватися у складі інформаційно-керуючих систем (ІУС) для завдань збору інформації про стан різних технічних систем, а також для управління ними [1].

Це визначається розвитком інтегральних мікросхем і, особливо, появою мікропроцесорів. На даному етапі розвитку для ІУС з невеликими габаритами, конструктивно об'єднаними з фізичним об'єктом контролю (управління), вдалою стала назва "вбудовані системи управління реального часу". Термін "реальний час" тут означає, що вбудована система повинна

виконувати певні дії, наприклад зчитування даних з датчиків і видачу команд на виконавчі пристрої, не раніше і не пізніше, а в строго задані моменти і інтервали часу [1].

До складу простих вбудованих систем входять:

- мікропроцесорний модуль пам'яттю;
- периферійна система;
- система електроживлення;
- об'єднуючий конструктив (шасі, корпус);
- керуюче програмне забезпечення (ПЗ).

Вбудовані системи продовжують стрімко проникати в усі області життя людини. Паралельно зростають і вимоги до якості їх роботи, що проявляється в необхідності правильного виконання функцій, у надійності зберігання та захисту інформації, у тривалості автономної роботи та багато в чому іншому [1].

Різноманітність додатків і збільшення їх складності змушують розробників активно вдосконалювати технології проектування вбудованих систем на всіх рівнях: компонентної бази, архітектури процесорів, засобів програмування, моделей організації обчислень [1].

Сьогоднішні вбудовані системи в переважній більшості створюються як багатопроцесорні системи, з процесорними ядрами hard і soft, з просторово зосередженою і/або просторово розподіленою архітектурою [1].

Стрімко набирає темпи створення динамічно реконфігурованих обчислювальних платформ, які дозволять різко знизити енергоспоживання вбудованих систем при зростанні їх функціональності та продуктивності [1].

1.2 Огляд платформ для вбудованих систем

1.2.1 Freeduino

Freeduino – програмується через USB порт мікро-ЕОМ (електронна обчислювальна машина), в яку можна завантажити будь-яку програму і отримати компактний автономний мікропроцесорний пристрій із закладеними в нього певними функціями [2].

Freeduino – сумісна з Ардуїно відкрита платформа (рис 1.1), популярна в усьому світі, і найбільш відома саме під ім 'ям Ардуїно. Зустрічаються також і російські варіанти написання: "Arduino" і " Freeduino " [2].



Рисунок 1.1 – Плата Freeduino [2]

Можливості платформи не обмежені наявною "на борту" периферією – існують модулі (рис. 1.2), що додають до Freeduino нові функції та інтерфейси, такі, наприклад, як модуль Ethernet, або модуль M-Shield, що спрощує підключення двигунів постійного струму і крокових двигунів:

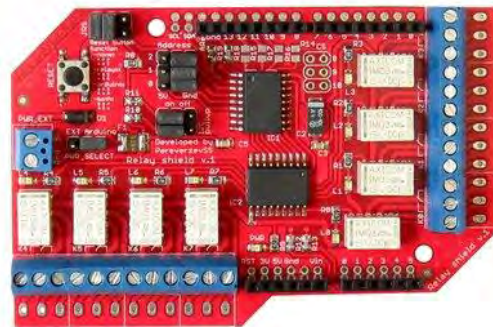


Рисунок 1.2 – Вигляд плати Freeduino [2]

Переваги:

- зв'язок пристрою з USB портом комп'ютера;
- не потрібно навичок програмування мікроконтролерів;
- не потрібні спеціалізовані програматори і компілятори;
- програмується дуже простою мовою, зрозумілою неспеціалісту;
- ручне і просте середовище розробки програм для мікроконтролера;
- широке поширення в мережі Internet: сотні сайтів з прикладами готових проєктів і бібліотек додаткових функцій;
- проєкт є 100% відкритим – доступні всі вихідні тексти;
- пристрій досить функціонально – 14 цифрових входів/виходів, з них 6 виходів з широтно-імпульсним модулюванням, 6 аналогових входів;
- багатоплатформенність. Середовище розробки програм для Freeduino працює на Windows, Macintosh OS X, Linux та інших операційних системах, оскільки є відкритою і реалізована на Java.

Freeduino можна використовувати практично по всіх областях електроніки, де потрібне управління системою за заданим алгоритмом з можливістю реагування на зовнішні сигнали [2].

За допомогою Freeduino Ви можете легко виготовити системи управління різних електронних пристроїв: світломузика; сигналізація; кроковий двигун; рідкокристалічна панель, можливе застосування у вимірі фізичних величин спільно з датчиками рівня, руху, ваги та багато іншого [2].

Електричні схеми мікроконтролерів Arduino і Freeduino не мають жодних принципових відмінностей, і пристрої 100% сумісні один з одним. Незважаючи на те, що весь проєкт Arduino (включаючи середовище розробки, компілятор, прошивки завантажувача, електричну схему та інше) є відкритим, є обмеження на використання назви Arduino. Тому ентузіасти й організували проєкт Freeduino [2].

1.2.2 Raspberry Pi

Raspberry Pi – це одноплатний комп'ютер, розміром з кредитну карту, Розроблений в Великобританії з метою використання в процесі навчання базовим знанням в сфері комп'ютерів. Одним з його серйозних переваг є рекордно низька ціна 25 \$ за модель А (спрощену) і 35 \$ за модель В (рис 1.3) [3].

Ідея створення маленького і дешевого комп'ютера для навчання з'явилася в 2006-му році у Ебена Аптона і його колег по Лабораторії обчислювальної техніки Кембриджського університету, які спантеличені тим фактом, що з року в рік знання студентів першокурсників факультету обчислювальної техніки ставали все гірше і гірше. Наприклад, якщо в 90-х роках більшість абітурієнтів приходили з хорошими знаннями на рівні програмістів-ентузіастів, то вже в 2000-х картина змінилася і типовий абітурієнт мав хіба що початковими знаннями в області веб дизайну [3].

З 2006 по 2008 роки Ебен розробив кілька зразків комп'ютера. Найперші зразки були зроблені на недорогих мікроконтролерах і дозволяли тільки навчатися програмуванню на мові Python. Після 2008 року всі ширше стали розповсюджуватися процесори для мобільних пристроїв, які були досить потужними і мали хороші засоби мультимедіа. Останнє виявилось особливо важливим, тому що підліткам не цікаво програмування саме по собі, але цікаві мультимедіа можливості комп'ютерів. В цей же час Ебен влаштувався працювати в компанію Broadcom, яка займається виробництвом таких процесорів. Після приєднання до його команди Девіда Брейбена, співавтора відомої гри Elite, було прийнято рішення про утворення некомерційної організації Pi Foundation, в складі якої вони продовжили розробку [3].

Raspberry Pi часто використовується як мозок робота, домашній сервер або просто комп'ютер [3].

Спочатку проект створювався як освітній, Raspberry Pi відмінно підходить для вивчення основ електроніки. На основі Raspberry Pi створено безліч комп'ютерних класів. Однак сьогодні його призначення вийшло за тільки освітнє [3].

Щоб почати роботу з Raspberry Pi крім самої плати вам знадобиться:

- SD-карта, з якою ви завантажите операційну систему; виробник рекомендує використовувати карту від 8 до 32 Gb, реально працює і на менших картах;
- монітор або телевізор з роз'ємами HDMI (High Definition Multimedia Interface), DVI або RCA (тільки для моделей A і B), і, відповідно, кабель HDMI-HDMI, HDMI-DVI або RCA-RCA, також можна використовувати HDMI-VGA перетворювач;
- USB-клавіатура, USB-миша, кабель живлення або акумулятор micro-USB;
- Raspberry Pi поставляється без ОС, її потрібно завантажити з сайту виробника і завантажити на SD карту.



Рисунок 1.3 – Raspberry Pi 4 Model B [4]

Попри збережені габарити попередньої серії (85,6 мм × 56,5 мм), в решті система зазнала істотних змін. Серед іншого, нове розташування кріплень та портів вводу-виведення робить нову серію несумісною з корпусами від попередньої серії. Проте, була збережена сумісність з платами розширення від попередньої серії [4]. Огляд версій плат Raspberry наведено в таблиці 1.1.

Таблиця 1.1 – Огляд версій плат Raspberry [4]

Версія	Дата початку продаж	Процесор	Частота	Ядер	ОЗП	GPIO	USB	Ethernet	WiFi	Bluetooth	Ціна
A	Лютий 2013	ARM1176JZ-F	700 МГц	1	256 МБ	26 пінів	1 порт				\$20
A+	Листопад 2014	ARM1176JZ-F	700 МГц	1	256 МБ	40 пінів	1 порт				\$25
B	Квітень 2012	ARM1176JZ-F	700 МГц	1	512 МБ	26 пінів	2 порти	Так			\$35
B+	Червень 2014	ARM1176JZ-F	700 МГц	1	512 МБ	40 пінів	4 порти	Так			\$30
2B	Лютий 2015	ARM Cortex-A7	900 МГц	4	1 ГБ	40 пінів	4 порти	так			\$35
Zero	Листопад 2015	ARM1176JZ-F	1 ГГц	1	512 МБ	40 пінів	1 порт				\$5
3B	Лютий 2016	ARM Cortex-A53 x64	1,2 ГГц	4	1 ГБ	40 пінів	4 порти	Так	802.11n	4.1	\$35
3B+	Березень 2018	ARM Cortex-A53 x64	1,4 ГГц	4	1 ГБ	40 пінів	4 порти	Так	двосмуговий 2,4/5 ГГц 802.11b/g/n/ac Wi-Fi	4.2	\$35
4B	Червень 2019	ARM Cortex-A72 x64	1,5 ГГц	4	1-4 ГБ	40 пінів	4 порти	Так	2,4 ГГц та 5,0 ГГц IEEE 802.11ac	5.0	\$35+
400	Листопад 2020	Broadcom BCM2711 ARM Cortex-A72 x64	1,8 ГГц	4	4 ГБ	40 пінів	2 × USB 3.0 та 1 × USB 2.0 порти типу A	Так	2,4 ГГц та 5,0 ГГц IEEE 802.11b/g/n/ac	5.0, BLE	\$70 або \$100

Модель Raspberry Pi 4B була представлена 23 червня 2019 року. Попри збережені габарити попередньої серії (85,6 мм × 56,5 мм), в решті система зазнала істотних змін. Серед іншого, нове розташування кріплень та портів вводу-виведення робить нову серію несумісною з корпусами від попередньої серії. Проте, була збережена сумісність з платами розширення від попередньої серії [4].

Нова серія побудована на основі системи на кристалі (SoC (System-on-a-chip)) BCM2711 в якій використаний чотирьох ядерний процесор 1.5GHz ARM (Advanced RISC Machine) Cortex-A72 з тактовою частотою 1,5 ГГц, виготовлений за технологічним процесом 28 нм [4].

Цей мікропроцесор має підтримку стандартного ARM GIC (Generic Interrupt Controller), що істотно пришвидшує роботу систем віртуалізації [4].

Підтримка Wi-Fi лишилась на рівні стандарту 802.11ac, але на відміну від попередньої серії додана підтримка Bluetooth версії 5.0 [4].

Четверта серія доступна в трьох конфігураціях оперативної пам'яті: 1, 2, та 4 ГБ ОЗУ стандарту LPDDR4. в червні 2020 року була випущена модель з 8 ГБ ОЗУ [4].

На відміну від попередньої серії, мережевий адаптер Gigabit Ethernet тепер працює на повну пропускну здатність, та не обмежений 300 Мбіт/с шлюзу USB 2.0 в третій серії [4].

Як і попередня серія, Raspberry Pi 4B підтримує технологію живлення через Ethernet (PoE) з встановленням додаткової плати розширення [4].

Замість одного порту HDMI нова серія має два порти micro-HDMI. Здатна виводити сигнал в форматі HDMI 2.0 4K@60 Гц на один монітор, або 4K@30 Гц на два монітори, програвати відео в форматі 4K [4].

Завдяки наявному в системі прискорювачу VideoCore 6 новий комп'ютер буде здатен відтворювати відео в форматі H.265 та 4K@60 кадри на секунду [4].

Живлення системи забезпечує порт USB Type-C. Однак, розробники системи припустились помилки й перші партії мають адаптер порту не

повністю відповідний стандарту USB Type-C. Через це можуть виникати проблеми з деякими стандартними кабелями [4].

Модель Raspberry Pi 400 (рис. 1.4) була представлена 2 листопада 2020 року.



Рисунок 1.4 – Комплект поставки Pi 400 [4]

Ця модель основана на Raspberry Pi 4B з 4 ГБ оперативної пам'яті, але має трохи швидший центральний процесор: Broadcom BCM2711 4-ядерний ARM Cortex-A72 (ARM v8) 64-bit SoC з тактовою частотою 1,8 ГГц [4].

Було запропоновано два варіанти цієї моделі: власне сам обчислювальний модуль, орієнтовною вартістю 70 доларів США, та цей же модуль вбудований у спеціальний корпус – клавіатуру. Як взірць для такого форм-фактору послужили «класичні» мікрокомп'ютери – BBC Micro, ZX Spectrum, Commodore Amiga та інші [4].

З тильного боку клавіатури розташовані роз'єми: 40-піновий роз'єм вводу–виводу загального призначення, слот для карток microSD, два порти micro-HDMI, роз'єм живлення USB-C, два порти USB 3.0 тип-A та один порт USB 2.0 тип-A, Gigabit Ethernet, замок Kensington [4].

21 січня 2021 року було представлено мікроконтролер Raspberry Pi Pico (рис. 1.5).



Рисунок 1.5 – Нижня та верхня сторони Raspberry Pi Pico [4]

Він побудований на основі 2-х ядерного мікропроцесора RP2040 архітектури ARM з тактовою частотою 133 МГц. Має 264 КБ ОЗУ, 26 пінів (контактів) введення–виведення (із них 3 – аналогових піна введення), порт мікро-USB та датчик температури. На платі відсутня підтримка як WiFi, так і Bluetooth. Орієнтовна ціна системи – 4 долара США [4].

Система має 2 МБ флеш пам'яті. Її можна програмувати із використанням MicroPython, CircuitPython, та C. Фонд Raspberry Foundation розпочав співпрацю з Adafruit, Pimoroni, Arduino та Sparkfun для створення периферійних пристроїв та інших аксесуарів для Raspberry Pi Pico та інших мікроконтролерів побудованих на основі чіпа RP2040 [4].

1.3 Аналіз платформи Arduino

1.3.1 Загальна характеристика

Arduino – це величезний конструктор, в якому немає кінцевого, певного набору деталей, і немає обмежень в розмаїтті того, що можна зібрати. Все обмежено лише вашою фантазією. Це новий світ, забійне хобі і відмінний подарунок [5].

У процесор Arduino можна завантажити програму, яка буде керувати всіма цими пристроями за заданим алгоритмом. Таким чином, можна створити

незліченну кількість оригінальних пристроїв, зроблених своїми руками і за власною задумом [5].

Користувач сучасного комп'ютера не замислюється про функціонування окремих частин ПК. Він просто запускає потрібні програми і працює з ними. Точно так само і Arduino дозволяє користувачеві зосередитися на розробці проектів, а не на вивченні пристрою і принципів функціонування окремих елементів. Немає потреби і в створенні закінчених плат і модулів. Розробник може використовувати готові плати розширення або просто безпосередньо підключити до Arduino необхідні елементи [5].

Всі інші зусилля будуть спрямовані на розробку і налагодження керуючої програми на мові високого рівня. В результаті доступ до розробки мікропроцесорних пристроїв отримали не тільки професіонали, але і просто любителі щось зробити своїми руками. Наявність готових модулів і бібліотек програм дозволяє непрофесіоналам в електроніці створювати готові працюючі пристрої для вирішення своїх завдань. А варіанти використання Arduino обмежені тільки можливостями мікроконтролера і наявного варіанту плати, ну і, звичайно, фантазією розробника [5].

Крім того, щоб плата заробила всього лише достатньо підключити її до комп'ютера через USB [6].

Додаткові плюси я перерахую нижче, але не лякайтеся, якщо якісь терміни ви не зрозумієте, з часом ви з ними розберетеся:

- плата має вбудований програматором (не потрібен компілятор);
- використання мови програмування близького до C / C ++, що робить її простий у використанні і вивченні;
- наявність безлічі бібліотек для модулів, сенсорів, дисплеїв і т.п., доступних для вільного використання;
- для збору пристрою не потрібно пайки, компоненти з'єднуються за допомогою спеціальної макетної плати, перемичок і проводів;
- можливість автономної роботи, тобто використання батарейок або акумуляторів;

- наявність версії для роботи з популярною мобільною операційною системою Android;
- величезна кількість різних модулів, сенсорів, дисплеїв, датчиків;
- наявність великої кількості плат розширення - shields або «Шілд».

Є ще один дуже суттєвий плюс Arduino - просто неймовірно величезне співтовариство любителів цих плат і любителів постворювали різні пристрої на її основі своїми руками [6].

Можна створити пристрій як стаціонарним, так і автономним або переносним. Наявність портативних акумуляторів і бездротових джерел передачі даних сприяють створенню ряду цікавих проектів [6].

Плата Arduino (рис. 1.6) складається з мікроконтролера Atmel AVR (рис. 1.7), а також елементів обв'язки для програмування та інтеграції з іншими пристроями. На багатьох платах наявний лінійний стабілізатор напруги +5В або +3,3В. Тактування здійснюється на частоті 16 або 8 МГц кварцовим резонатором. У мікроконтролер записаний завантажувач (bootloader), тому зовнішній програматор не потрібен [7].



Рисунок 1.6 – Процесорна плата Ардуіно з платами макетування [7]



Рисунок 1.7 – Мініатюрна процесорна плата без встановлених деталей [7]

На концептуальному рівні усі плати програмуються через RS-232 (послідовне з'єднання), але реалізація даного способу різниться від версії до версії. Новіші плати програмуються через USB, що можливо завдяки мікросхемі конвертера USB-to-Serial FTDI (Future Tehnology Device International) FT232R. У версії платформи Arduino Uno як конвертер використовується контролер Atmega8 у SMD-корпусі. Дане рішення дозволяє програмувати конвертер таким чином, щоб платформа відразу розпізнавалася як миша, джойстик чи інший пристрій за вибором розробника зі всіма необхідними додатковими сигналами керування. У деяких варіантах, таких як Arduino Mini або неофіційній Boarduino, для програмування потрібно підключити до контролера окрему плату USB-to-Serial або кабель [7].

Плати Arduino дозволяють використовувати значну кількість виводів мікроконтролера як входні/вихідні контакти у зовнішніх схемах. Наприклад, у платі Decimila доступно 14 цифрових входів/виходів, 6 із яких можуть генерувати ШІМ (широтно-імпульсна модуляція) сигнал, і 6 аналогових входів. Ці сигнали доступні на платі через контактні площадки або штирові роз'єми. Також існує багато різних зовнішніх плат розширення, які називаються «shields», які приєднуються до плати Arduino через штирові роз'єми [7].

Ардуіно і Ардуіно–сумісні плати спроектовані таким чином, щоб їх можна було при необхідності розширювати, додаючи в пристрій нові компоненти («shields»). Ці плати розширень підключаються до Ардуіно за допомогою встановлених на них штирових роз'ємів. Існує ряд уніфікованих плат, що допускає конструктивно жорстке з'єднання процесорної плати та плат розширення в стопку через штирові лінійки. Крім того, випускаються плати зі зменшеним (наприклад, Nano, Lilypad) і спеціальним (для задач робототехніки) форм-фактором [7].

Сторонніми виробниками випускається велика гамма всіляких датчиків і виконавчих пристроїв, в тій чи іншій мірі сумісних між собою і з процесорними платами Ардуіно [7].

Сторонніми виробниками також випускаються набори електромеханічних елементів, орієнтованих на роботу спільно з платами Ардуіно (як правило, через спеціальні плати «драйвери») – двигуни, електромагніти тощо [7].

У концепцію Ардуіно не входить корпусних або монтажних деталей. Розробник вибирає метод установки і механічного захисту процесорних плат та компонентів розширення самостійно [7].

1.3.2 Огляд та порівняння існуючих платформ Arduino

Arduino Uno (рис. 1.8) є стандартною платою Arduino і можливо найбільш поширеною. Вона заснована на чіпі ATmega328, що має на борту 32 КБ флеш-пам'яті, 2 КБ SRAM і 1 Кбайт EEPROM пам'яті. На периферії має 14 дискретних (цифрових) каналів введення / виводу і 6 аналогових каналів введення / виводу, це дуже різнобічно-корисні девайси, що дозволяють перекривати більшість аматорських завдань в області мікроконтроллерной техніки. Дана плата контролера є однією з найдешевших і найбільш часто використовуваних. При плануванні нового проєкту, якщо ви не знайомі, з платформою Arduino, радимо почати з Uno а інші моделі представлені в таблиці 1.2 [8].



Рисунок 1.8 – Arduino Uno [8]

Таблиця 1.2 – Порівняння плат Arduino [8]

Arduino	Процесор	Напруга живлення	Флеш пам'ять, КБ	EEPROM, КБ	SRAM, КБ	Двійкові входи/виходи	...з ШІМ	Аналогові входи	USB-інтерфейс	Інші інтерфейси	Розміри, мм
ADK	ATmega2560	5 В	256	4	8	54	14	16	ATmega8U2	MAX3421EUSB хост	101.6 × 53.3
BT (Bluetooth)	ATmega328	5 В	32	1	2	14	4	6	Відсутній	Bluegiga WT11 Bluetooth	
Diecimila	ATmega168	5 В	16	0.5	1	14	6	6	FTDI		68.6 × 53.3
Due	ATMEL SAM3U		256	0	50	54	16	16			
Duemilanove	ATmega168/328P	5 В	16/32	0.5/1	44228	14	6	6	FTDI		68.6 × 53.3
Ethernet	ATmega328	5 В	32	1	2	14	4	6	Відсутній	Wiznet Ethernet MicroSD	
Fio	ATmega328P	3.3 В	32	1	2	14	6	8	Відсутній		40.6 × 27.9
Leonardo	Atmega32u4	5 В	32	1	2	14	6	12	Atmega32u4		68.6 × 53.3
LilyPad	ATmega168V or ATmega328V	2.7-5.5 В	16	0.5	1	14	6	6	Відсутній		50 ∅
Mega	ATmega1280	5 В	128	4	8	54	14	16	FTDI		101.6 × 53.3
Mega2560	ATmega2560	5 В	256	4	8	54	14	16	ATmega8U2 ATmega16U2		101.6 × 53.3
Nano	ATmega168 or ATmega328	5 В	16/32	0.5/1	44228	14	6	8	FTDI		43 × 18
Pro Mini	ATmega328P	5 В або 3.3 В	32	1	2	14	6	8	FTDI	UART	33 × 18
Uno	ATmega328P	5 В	32	1	2	14	6	6	ATmega8U2 ATmega16U2		68.6 × 53.3
MK-duino	ATmega8	5 В							[Відсутній]	[RS-232]	

Arduino Leonardo (рис. 1.9) та ж Arduino Uno, але з іншим мікроконтролером, який знаходиться в тому ж класі, але має деякі відмінності позитивного характеру. Більша кількість аналогових входів (12 проти 6) для сенсорів, більше каналів ШІМ (7 проти 6), більше пінів з апаратним перериванням (5 проти 2), роздільні незалежні serial-інтерфейси для USB і UART (Universal asynchronous receiver/transmitter). Arduino Leonardo може прикидатися клавіатурою або мишею (HID-пристроєм) для комп'ютера. Це дозволяє легко зробити своє власне пристрій введення. Через терморегулятори трохи відмінною від Arduino Uno можлива несумісність з деякими платами розширення [8].



Рисунок 1.9 – Arduino Leonardo [8]

Arduino Nano (рис. 1.10) – це функціональний аналог Arduino Uno, але розміщений на мініатюрній платі. Відмінність полягає у відсутності власного гнізда для зовнішнього живлення, використанням чіпа FTDI FT232RL для USB-Serial перетворення (або CH340G, потрібно встановити відповідні драйвера) і застосуванням mini-USB кабелю для взаємодії замість стандартного. В іншому, начинка і способи взаємодії збігаються з базовою моделлю. Платформа має Штиркові контакти, що дозволяє легко встановлювати її на макетну плату. Використовуйте Arduino Nano там, де важлива компактність, а можливостей Arduino Mini чи мало, або не хочеться займатися паянням [8].

Arduino Mega (рис. 1.11) як Arduino Uno, але на базі більш потужного мікроконтролера тієї ж архітектури.



Рисунок 1.10 – Arduino Nano [8]

Відмінний вибір «на виріст» або якщо Arduino Uno перестала справлятися. В рази більше пам'яті: 256 КБ проти 32 КБ постійної і 8 КБ проти 2 КБ оперативної. В рази більше портів: 60 з них 16 аналогових і 15 з ШІМ. Трохи довше базової Arduino Uno: 101×53 мм проти 69×53 мм [8].



Рисунок 1.11 – Arduino Mega [8]

Arduino Due (рис. 1.12) одна з найбільш продуктивних плат від Arduino на мікроконтролері Cortex-M3 по форм-фактору аналогічна Arduino Mega. Процесор на 84 МГц і 512 КБ пам'яті. 66 пінів введення-виведення, з яких 12 можуть бути аналоговими входами, 12 підтримують ШІМ і всі 66 можуть бути налаштовані, як апаратні переривання. Вбудований контролер шини CAN дозволяє створювати мережу з Due або взаємодіяти з автомобільною електронікою. Два каналу ЦАП дозволяють синтезувати стереозвук з

дозволом в 4,88 Гц. Рідним напругою для плати є 3.3 В, а не традиційні 5 В. Необхідно стежити, щоб обрана периферія підтримувала роботу з цим рівнем або ставити перетворювачі рівнів напруги [8].



Рисунок 1.12 – Arduino Due [8]

Arduino Mini (рис. 1.13) та ж Arduino Uno, але в іншому форм-факторі. Компактна: всього 30×18 мм. Через форм-фактора не можна без хитрощів встановлювати плати розширення Arduino. Передбачається з'єднання з додатковими модулями проводами і / або через макетну плату. На платі немає USB-порту, тому прошивати потрібно через окремий USB-Serial адаптер [8].

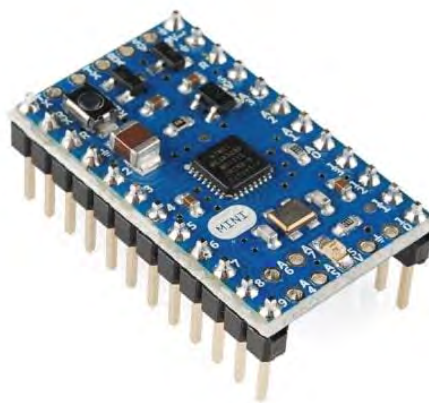


Рисунок 1.13 – Arduino Mini [8]

Arduino Micro (рис. 1.14) – це Arduino Leonardo, виконаний на компактній платі. Відмінність полягає у відсутності власного гнізда для

зовнішнього живлення, але воно може бути підведено безпосередньо до контакту Vi. В іншому, начинка і способи взаємодії збігаються з Arduino Leonardo. Він також має один мікроконтролер ATmega32u4 і для прошивки через USB, і для виконання програм; також може виступати в ролі клавіатури або миші; надає таку саму кількість пам'яті, цифрових, аналогових і ШІМ-портів [8].

Плата Arduino M0 виконувати складні математичні розрахунки, отримувати більш точні аналогові значення і при цьому слухати музику безпосередньо з мікроконтролера. Arduino M0 заснована на 32-бітному ARM-процесорі ATSAM21G18 від Atmel з обчислювальним ядром Cortex® M0. Мікроконтролер працює на частоті 48 МГц [8].



Рисунок 1.14 – Arduino Micro [8]

А завдяки своїй 32-бітній архітектурі він виконує більшість операцій над цілими числами всього за один такт. На відміну від більшості плат Arduino, рідним напругою Arduino M0 (рис. 1.15) Pro є 3.3 В, а не 5 В. Відповідно, виходи для логічної одиниці видають 3.3 В, а в режимі входу очікують приймати не більше 3.3 В. Arduino M0 дивиться в бік USB через віртуальний serial-порт, не через апаратний. Це означає, що 0-й і 1-й контакти апаратного порту залишаються вільними і ви можете використовувати їх одночасно з комунікацією з комп'ютером. Віртуальний serial-порт доступний через об'єкт SerialUSB, а апаратний через об'єкт Serial1 [8].



Рисунок 1.15 – Arduino M0 [8]

Arduino LilyPad (рис. 1.16) – досить цікавий пристрій. Воно випадає зі звичних стереотипів про звичайний Arduino, тому що має не прямокутну, а круглу форму. По-друге, воно не підтримує механічні з'єднання з ШІлд. Воно призначене для, невеликих автономних пристроїв. Кругла форма продиктувала те, що роз'єми рівномірно розподілені по колу, і його невеликий розмір (2 дюйма в діаметрі) робить його ідеальним для переносних пристроїв. Це пристрій легко заховати, і кілька виробників розробили пристрої, спеціально для LilyPad: екрани, датчики світла, навіть коробки для батарей харчування, які можуть бути зашиті в тканину. Для того, щоб зробити LilyPad якомога менше і якомога легше, на скільки можливо, були принесені деякі жертви. У LilyPad немає регулятора напруги на борту, так що йому для харчування буде необхідно забезпечити принаймні 2.7 вольт, і не більше 5.5 вольт ці плати та їх характеристики зображені в таблиці 1.3 [8].

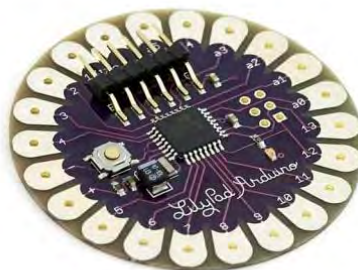


Рисунок 1.16 – Arduino LilyPad [8]

Таблиця 1.3 – Порівняльний аналіз плат Arduino [8]

Плата Arduino	Мікроконтролер	Робоча напруга (В)	Цифрові входи/виходи	Виходи з ШИМ	Аналогові входи/виходи	Максимальний струм з Піна введення-виведення (мА)	Flash-пам'ять (КБ)	RAM (КБ)	EEPROM-пам'ять (КБ)	Тактова частота (МГц)	Габарити (мм)	USB-роз'єм
Uno	ATmega328	5	14	6	6	40	32	2	1	16	69×53	USB A-B
Leonardo	ATmega32u4	5	20	7	12	40	32	2.5	1	16	75×53	micro-USB
Nano	ATmega328	5	14	6	8	40	32	2	1	16	18×45	mini-USB
Mega	ATmega2560	5	54	14	16	40	256	8	4	16	102×53	USB A-B
Due	Atmel SAM3X8E ARM Cortex-M3	3.3	54	12	12	800	512	96	×	84	102×53	micro-USB
Mini	ATmega328	5	16	6	6	40	32	2	1	16	30×18	USB-Serial
Micro	ATmega32u4	5	20	7	12	40	32	2.5	1	16	48×18	micro-USB
Mo	Atmel SAMD21	3.3	20	12	6	7	256	32	×	48	69×53	micro-USB
LilyPad	ATmega328p	2.7 - 5.5	20	6	6	40	32	2	1	8	Ø 50	USB-Serial

2 ДОСЛІДЖЕННЯ ОСОБЛИВОСТЕЙ ГІБРИДНОЇ ЛАБАРАТОРІЇ GOLDi

2.1 Структура мережі GOLDi

Група з комплексних комунікаційних систем у Технічному університеті Ільменау, Німеччина, розробила апаратне та програмне забезпечення гібридної лабораторії GOLDi. Ця лабораторія використовується для проведення практичних занять, а також надання практичного досвіду для розробки вбудованої електроніки. Це виконується не з використанням на місці уроку, а дистанційно через Інтернет. Перевага у тому, що курси можуть пропонуватися на міжнародному рівні у всьому світі, що дає студентам з різних країн, які розмовляють різними мовами, однаковий доступ і рівні можливості в лабораторії. Крім того, навіть для місцевих студентів лабораторії пропонують розширені години роботи порівняно зі звичайною лабораторією. Крім переваг для студентів, це також знижує витрати на академічне навчання і підвищує якість, пропонуючи розширення можливостей практичного навчання.

Повністю дистанційні лабораторії GOLDi працюють в десяти університетах Вірменії, Грузії, Німеччини, Австралії та України (рис.2.1). Це означає, що кожен партнер отримує повністю дистанційну лабораторію. Одна з проблем реалізації – це обслуговування кожної дистанційної лабораторії партнера:

- будь-яка нова функціональність або нове вбудоване ПЗ повинні бути впровадженими вручну на кожному веб-сайті партнера;
- кожна установа має конкретні зміни відповідно до своїх власних вимог;
- кожен університет має різну архітектуру мережі;
- кожен університет має свої налаштування дистанційної лабораторії;

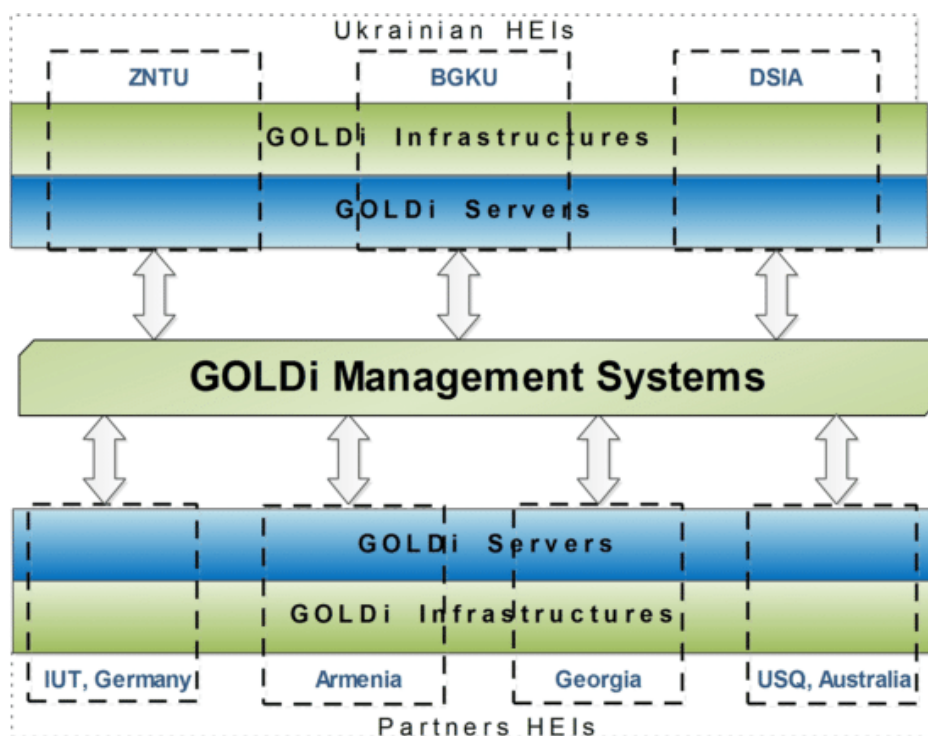


Рисунок 2.1 – Загальна архітектура GOLDI

– обліковий запис користувача на відповідному вебсайті надає доступ до експериментів в лабораторіях партнерів.

Кожен партнер GOLDi повинен виконувати технічне обслуговування саме за підтримки проєктної команди Імпапу GOLDi. Ця процедура дуже неефективна. Тому модернізували мережу всіх дистанційних лабораторій GOLDi до хмарної системи GOLDi. Доступний Сервери GOLDi (це означає доступні партнерські дистанційні лабораторії) зареєстровані в хмарі GOLDi. Вони підсвічені як готові для використання на центральному веб сайті GOLDI (рис. 2.2).

Студенти можуть використовувати будь-яке обладнання, доступне в режимі онлайн за адресою будь-яких з цих університетів. Вони можуть спостерігати, що студенти з інших країн мають однакові навчальні завдання, однакове обладнання та однакову навчальну базу, і вона виступає як сильна мотивація для студентів.

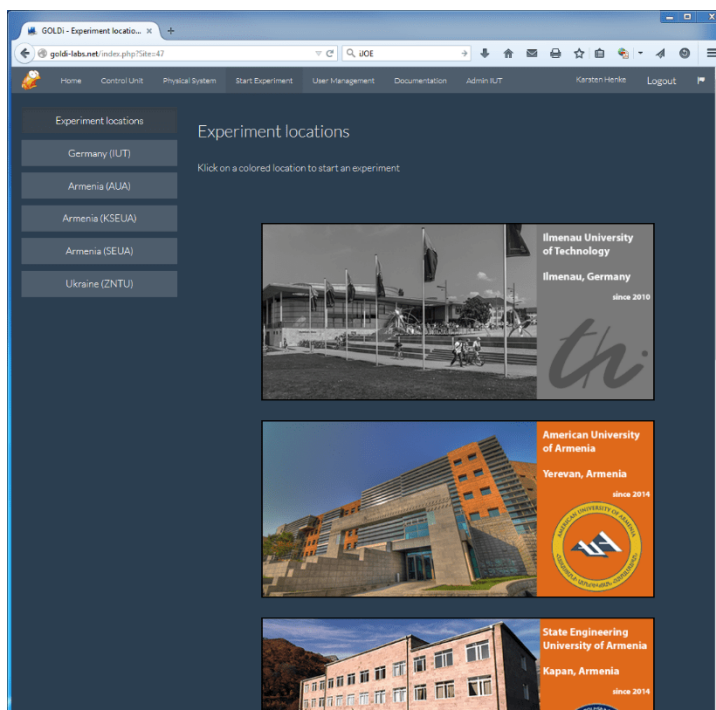


Рисунок 2.2 – Вебсайт GOLDi

Хмарна архітектура лабораторії (рис. 2.3). має такі характеристики:

- обслуговування всієї системи в одному центральному місці: goldi-labs.net;
- усі університети – партнери мають однакову версію GOLDi;
- нові функціональні можливості доступні для всіх партнерів;
- використання всіх лабораторій партнерів з одним центральним обліковим записом користувача goldi-labs.net;
- користувач може визначити, яка лабораторія має найменшу затримку (найкращий час реакції) відносно його поточному географічному положенню і підключенню до Інтернету;
- огляд всіх поточних експериментів у всіх лабораторіях партнерів по всьому світу з можливістю спостерігати за цими експериментами;
- кожен клієнт GOLDi взаємодіє з хмарою GOLDi для централізованого управління: користувачами, експериментами та бронюванням;

- всі дані хмари, що відносяться до конфіденційних, знаходяться в обчислювальному відділі Ілменау технологічного університету;
- кожен клієнтський додаток обмінюється тільки сигналами датчика/приводу безпосередньо з локальною інфраструктурою GOLDi.

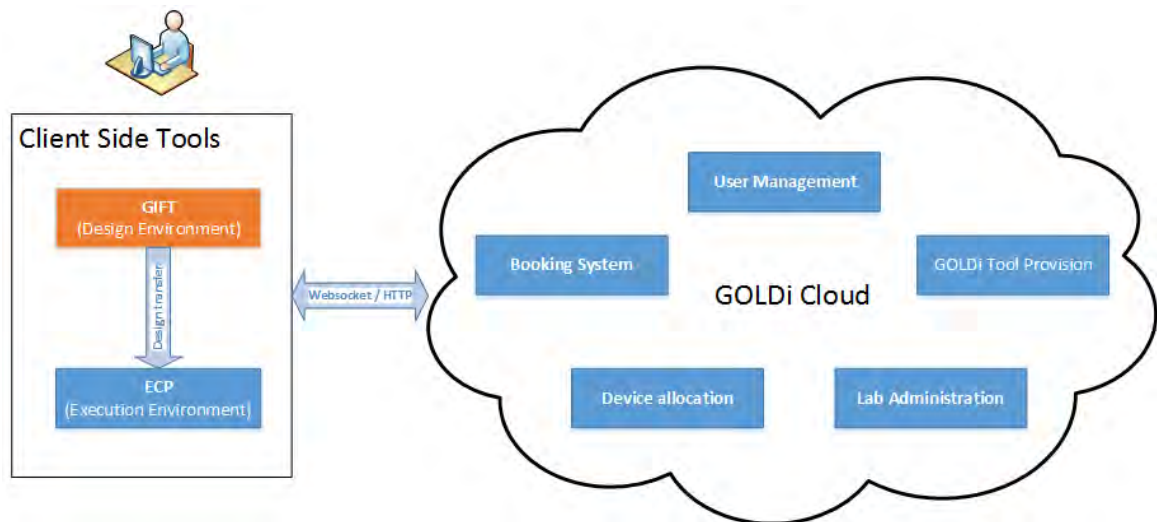


Рисунок 2.3 – Хмарна архітектура GOLDi

2.2 Структура лабораторії

Структуру лабораторії наведено на рис. 2.4.

Пристрій управління, пристрій керування процесором – компонент апаратного забезпечення центрального процесора комп'ютерів, що управляє роботою та взаємодією функціональних вузлів ПК. Являє собою кінцевий дискретний автомат [9].

Структурно пристрій управління можна поділити на такі основні складові:

- дешифратор команд / операцій;
- регістровий файл;
- вузол обчислення поточної адреси, що виконується;
- лічильник команд;

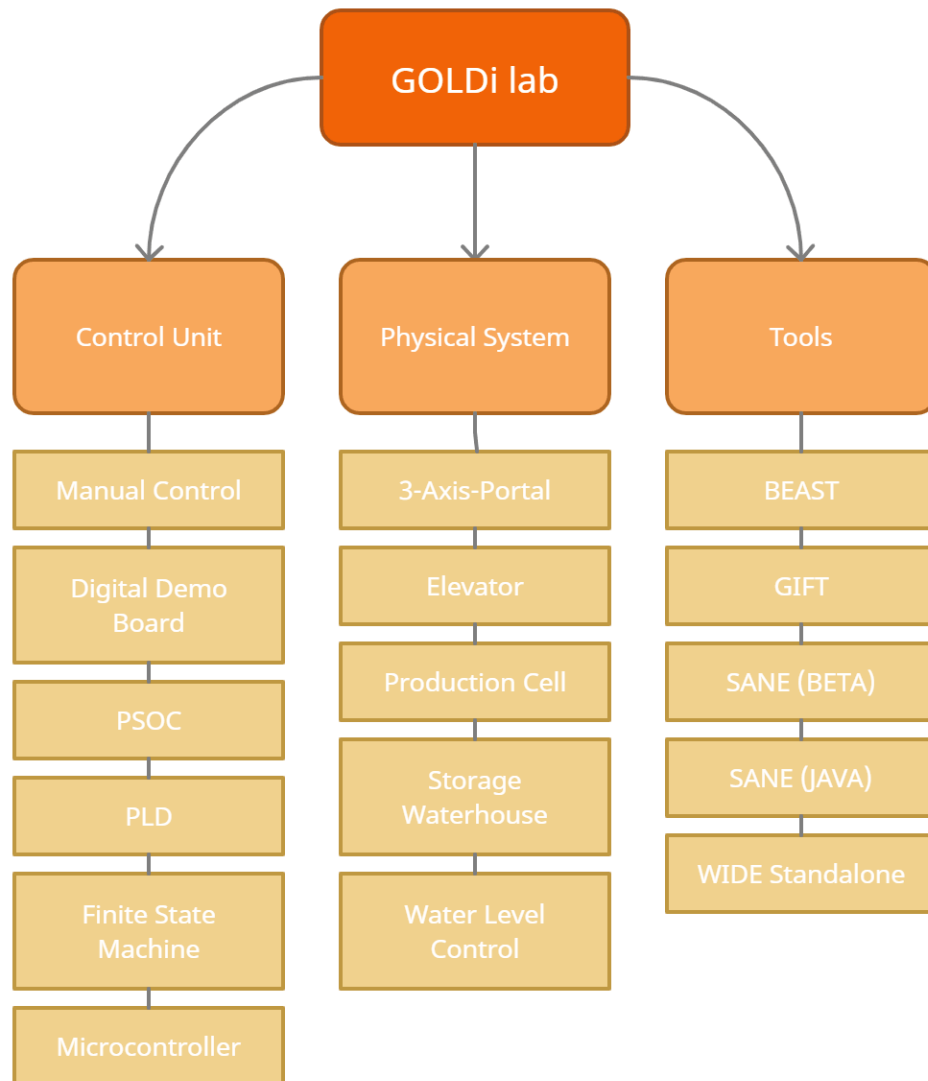


Рисунок 2.4 – Структура GOLDi

При виконанні програми пристрій управління отримує чергову інструкцію (команду програми), стан регістра ознак результату виконання попередньої команди (регістра прапорів), стан керуючих входів (наприклад, сигналів переривань або готовності периферійних пристроїв) і на вихід видає послідовності імпульсів управління (міжрегістрових пересилками, адресацією і сигналами пересилань даних практично між всіма блоками комп'ютера – арифметико-логічним пристроєм, підсистемою введення / виводу, оперативною пам'яттю [9].

У більшості випадків для виконання команди / інструкції пристрій управління видає кілька керуючих імпульсів або їх комбінацій в

послідовності, що визначається кодом поточної інструкції (команди). Після завершення видачі всіх імпульсів управління для виконання поточної команди зчитує код наступної команди і т.д [9].

Пристрій управління зазвичай синхронізується зовнішніми тактовими імпульсами, при цьому перехід кінцевого автомата з одного стану в інший відбувається на кожному тактовому імпульсі або його фазі [9].

Послідовність станів пристрою управління відповідно до інструкцій може визначатись апаратно або мікропрограмно [9].

Сюди перенаправляється запит «Жорстка логіка». Пристрої керування, реалізовані апаратно, забезпечують зміну станів та сигнали автомата апаратними засобами. Як спрощену модель такого управління можна розглядати мікросхему постійної пам'яті, в якій записані дані, що визначають стан автомата відповідно до адреси ПЗП, причому, ці дані застосовуються також і для формування наступної адреси ПЗП (тобто, наступного стану). Подібні «автомати» використовуються при необхідності реалізації відносно простої логіки з числом керуючих сигналів до декількох сотень, крім того, мікросхеми постійної пам'яті можуть мати недостатню швидкодію. В більш складних випадках логіка автомата забезпечується окремими логічними вентилями, тригерами, лічильниками тощо (так звана жорстка, або фіксована логіка), що дозволяє створювати автомати з на порядки більшим числом керуючих сигналів [9].

Апаратне керування дозволяє істотно підвищити швидкість роботи автомата, але призводить до ускладнення його конструкції. Зміна логіки автомата досягається його фізичним чи структурним перепроєктуванням / переробкою [9].

Мікропрограмне забезпечення дозволяє значно спростити пристрій керування, переклавши частину логіки автомата з обладнання на мікропрограми. Мікропрограмне керування було запропоновано в 1951 році британським вченим Морісом Вілком [9].

Ідея мікропрограмного керування полягає у можливості керувати станами та сигналами автомата через спеціальний мікрокод, що складається з мікроінструкцій. При надходженні в пристрій управління нової інструкції (команди процесора) обладнання та мікрокод виконують дешифрування інструкції процесора і забезпечують її виконання відповідно до типу інструкції та попереднього стану пристрою управління [9].

Мікропрограми також забезпечують обробку переривань, коригування помилок та інші операції процесора, що характеризуються досить складною логікою [9].

Введення мікрокоду дозволило істотно зменшити складність пристрою керування. Обсяг мікрокоду на різних платформах може істотно відрізнитись залежно від того, яка частина логіки автомата реалізована через мікрокод [9].

Interactive hybrid online lab – лабораторія під назвою GOLDi (Grid Online Lab Devices Ilmenau) – позначає сітку концепції створення універсальної віддаленої лабораторної інфраструктури.

Внутрішня організація лабораторії (рис. 2.5), складається з серверної сторони та трьох частин:

- внутрішня послідовна шина віддаленої лабораторії;
- блок резервування шини (BPU) для сполучення обраного блоку керування з віддаленою лабораторною шиною і для захисту її від неправильного використання та пошкодження;
- блок фізичного захисту системи (PSPU), який захищає фізичні системи (електромеханічний моделі у віддаленій лабораторії) від навмисного пошкодження або випадкового неправильного командування та управління, яке пропонує різні механізми доступу та користування;

Взаємозв'язок між блоками керування Web і вибраної моделі фізичного обладнання під час віддаленої роботи з ними (експеримент), а також керування користувачами виконується віддаленим сервером лабораторії, який також обробляє веб-камери.

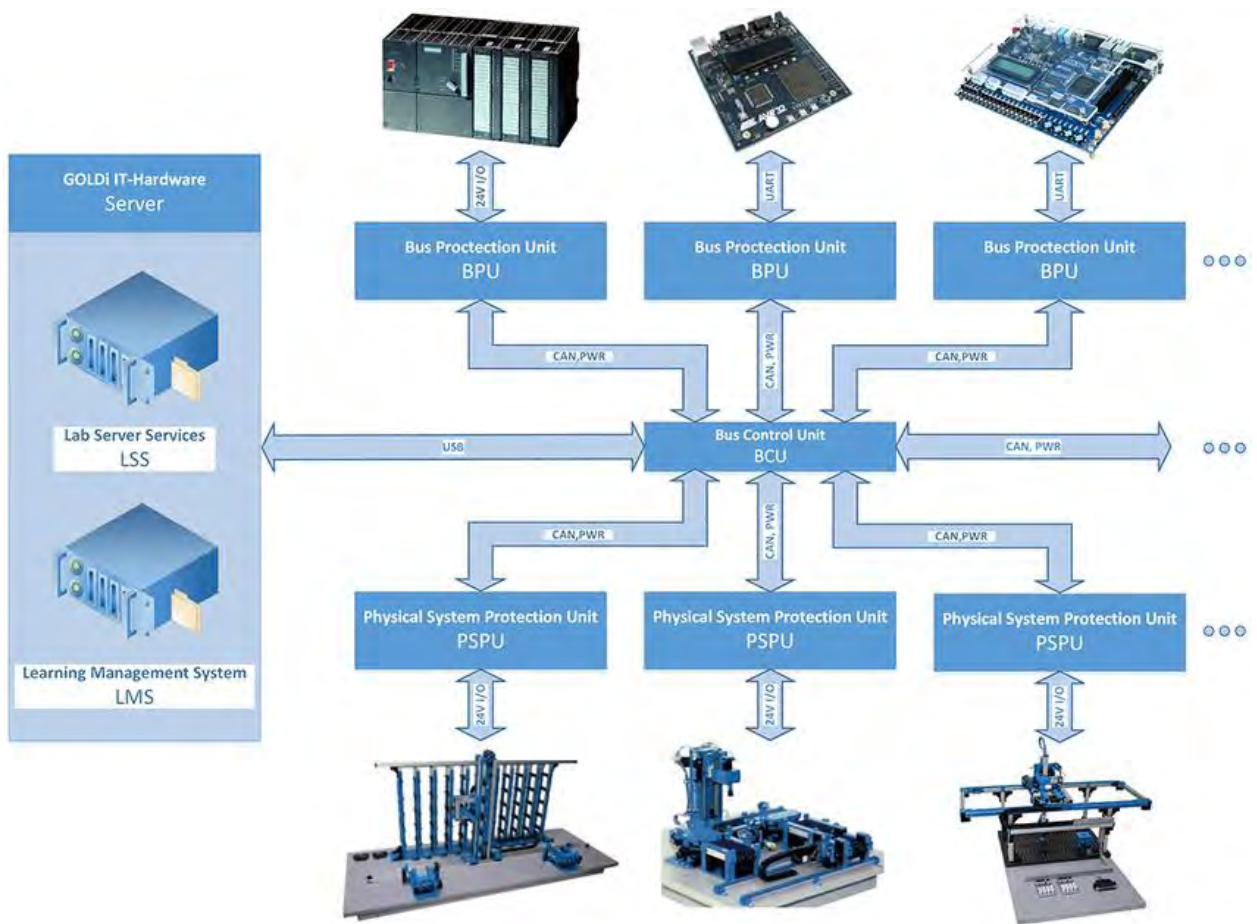


Рисунок 2.5 – Внутрішня організація GOLDi

2.3 Приклади використання лабораторії

2.3.1 Експеримент 3-Axis-Portal

Модель 3-Axis-Portal моделює стаціонарний маніпуляційний робот з ортогональним робочим простором, що використовується для передачі заготовок у блок обробки або сортування, наприклад, на заводах, що автоматизуються великою мірою. Модель складається з портального робота, здатного рухатися в трьох лінійних напрямках і електромагнітного захоплення, придатного для переміщення в Z-напрямку, штучного магазину і розряду станції.

Кожне з кінцевих положень декількох рухомих частин розпізнається програмними перемикачами кінцевого положення. Модельований процес показує, що металеві заготовки витягуються з магазину електромагнітним

захопленням, переміщуються на розряд і опускаються: на початку послідовності рухомі частини робота виконують еталонний обхід, щоб зрівняти системи вимірювання приросту відстані осей X і Y з їх дійсними положеннями. Контрольний обхід виконується із захопленням, що знаходиться у верхньому кінцевому положенні. Після цього захоплення переміщується в напрямку X і Y до тих пір, поки воно не досягне необхідного положення над сховищем деталей. Захоплення переміщується в напрямку $-Z$, поки не торкнеться заготовки. Магніт вмикається, і деталь прикріплюється до захоплення. Захоплення переміщується в напрямку $+Z$ до тих пір, поки воно не досягне свого верхнього кінцевого положення. Після цього він виконує рух у напрямку X і Y до тих пір, поки не досягне положення над станцією розряду. Досягнувши цього стану, захоплення знову переміщується в напрямку $-Z$ до тих пір, поки прикріплена деталь не буде поміщена на розряд станцію, де вона розпізнається індуктивним вимикачем близькості. Заготівлю витягують відключенням електромагнітного захоплення.

Для того щоб позбавити весь блок від пошкодження в результаті переміщення з допустимого робочого простору, викликаного помилкою при програмуванні блоку управління, лінійні напрямки додатково забезпечені апаратними вимикачами кінцевого положення, які викликають негайний останок відповідної осі у разі спрацювання. Електронна концепція моделі дозволяє тільки перезавантажити роботу, здійснюючи рух до робочого простору.

2.3.2 Експеримент Elevator B

Elevator B (4 floors) ліфт складається з клітини з противагами, котловану і чотирьох підлогових блоків, кожен з яких містить пневматичні розсувні двері, кнопки виклику і кольорові лампи управління для зазначення напрямку руху клітини. Крім цього є пульт управління, що реалізує варіанти роботи зсередини обойми.

По суті, це кнопки вибору для вибору підлоги, тривожна кнопка, аварійний останок і можливість вибору режиму роботи, де підйомник управляється виключно зовні клітини. Змодельований процес показує, що ліфт переміщується з основного положення на один з поверхів за допомогою панелі керування або однієї з кнопок виклику і після відкриття і закриття розсувних дверей готовий до наступної послідовності: Після спрацювання однієї з кнопок виклику, що індикується сигнальною лампою, клітина переміщується повільно-швидко-повільно, керуючись механічними перемикачами залежно від відстані, на обраний поверх.

Розсувні двері відкриваються і залишається відкритими до закінчення запрограмованого часу завантаження. Один шлях світловий бар'єр контролює вхід, щоб запобігти, в реальному випадку, заподіяння шкоди людям або речам, що знаходяться в небезпечній зоні дверей.

Після закриття розсувних дверей клітина переміщується на наступний обраний поверх, де послідовність відкривання і закривання розсувних дверей відбувається таким же чином. У модель вбудовано мініатюрний компресор для пневматичних розсувних дверей.

2.3.3 Експеримент Production Cell

Production Cell модельна технологічна комірка моделює технологічну комірку з інтегрованим круговим накопичувачем, як використовується, наприклад, у промислових процесах різання металу. Технологічний осередок складається з рейкової виводної каретки з конвеєрною стрічкою, двох поворотних столів з конвеєрними ланцюгами, вертикального фрезерного верстата і трьох конвеєрних стрічок.

Різні транспортні засоби розташовані в замкнутому контурі. Механічні вимикачі керують кінцевими положеннями рухомих вузлів транспортування та індуктивні безконтактні вимикачі перевіряють положення деталей. Змодельований процес показує, що деталь подається в кільцевий накопичувач,

потім по годинниковій стрілці транспортується у вертикальний фрезерний верстат, де обробляється, і, нарешті, виводиться на розвантажувальну станцію.

Немеханізовану частину укладають на конвеєрну стрічку і підводять до рейкової свинцевої каретки. Каретка підводить його до наступної конвеєрної стрічки. Там деталь транспортується до поворотного столу, який виконує поворот на 90° (+ С-напрямок) для переміщення деталі на наступну конвеєрну стрічку. Ця конвеєрна стрічка транспортує її на фрезерний верстат. Вертикальний корпус виконує рух уздовж його - Y-напрямку, щоб намалювати біля деталі. Шпиндель починає обертатися і переміщається вздовж його -Z-напрямку, щоб інтимно послідовність обробки.

Після завершення цієї послідовності вертикальний корпус і шпиндель повертаються у вихідне положення, і механічно оброблена частина передається на другий поворотний стіл, який знову повертає її на першу конвеєрну стрічку, готову до вилучення.

Storage Warehouse High Level Місце зберігання Warehouse моделює систему зберігання даних високого рівня, яка використовується, наприклад, у багатьох галузях промисловості. Модель складається зі стійки, розділеної на 5 x 10 місць зберігання, пристрою управління складом, портативного в напрямку X, і двох станцій зарядки/розвантаження.

До пристрою управління складом прикріплена клітина, що є переносною в Z-напрямку і включає телескопічний носій палітри, переносний в Y-напрямку. Моделюваний процес показує, що палітри зберігаються і витягуються з місця зберігання високого рівня: У разі, якщо один платіж станція зайнята палітрою, телескопічний носій палітри переміщується до станції і захоплює палітру. Це розпізнається відбиваючим світловим перемикачем. Слідом за цим операційний пристрій складу виводить палітру на передбачуваний займати місце зберігання в оптимізованому способі шляхом одночасного переміщення в напрямку X і Z. Займаюче місце зберігання розпізнається програмним забезпеченням. Для швидкого переміщення до місця зберігання, з одного боку, і безпечного переміщення, з іншого боку,

горизонтальні положення стійки забезпечені вдосконаленими механічними перемикачами, які дозволяють затримувати робочий пристрій складу до досягнення заданого положення. Видалення палітр відбувається таким же чином, виконується в зворотному хронологічному порядку. Під час додавання осей Y до двох інших осей переміщення утримувача панелі в напрямку X і Z можливе, лише якщо утримувач панелі знаходиться в середньому положенні. Крім того, осі X оснащені апаратним вимикачем кінцевого положення для запобігання фатальних помилок всієї складської системи при використанні конвеєра або програмуванні блоку управління. Склад зберігання високого рівня може бути об'єднаний з додатковими модулями і стандартними моделями для автоматизації периферії складу.

2.4 Методика використання гібридної лабораторії

2.4.1 Як користуватися ЕСР

Крок 1. Спочатку потрібно створити обліковий запис або увійти за допомогою вже наявного облікового запису, та розпочати експеримент (рис. 2.6).



Рисунок 2.6 – Інтерфейс входу

Перш ніж розпочати вибраний експеримент, вам слід підготувати відповідний файли для фізичної системи. Ви можете знайти більше інформації про те, як адаптуватися файли з фізичних систем на блоці управління краном.

Блоки управління:

- ручне управління;
- цифрова демонстраційна дошка;
- PLD;
- кінцевий автомат;
- MicroController.

Крок 2. Клацніть на вкладці Почати експеримент та виберіть місце розташування експериментів (рис. 2.7).

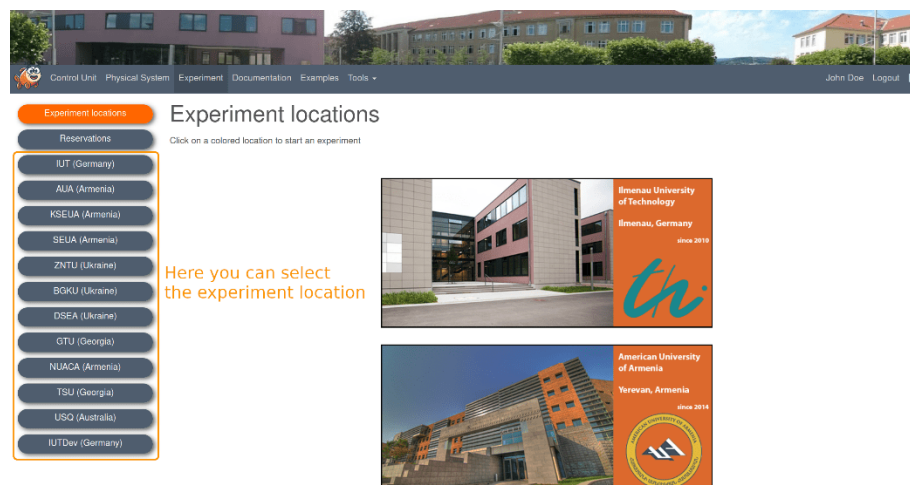


Рисунок 2.7 – Вибір місця розташування експериментів

Виберіть фізичну систему та блок управління, які ви хочете використовувати (рис. 2.8).

При цьому доступні такі опції:

- ви можете вибрати між реальним та віртуальним, коли ви вибираєте фізичний блок системи та управління;
- вибравши Real, ви можете спробувати свій експеримент на реальній системі;

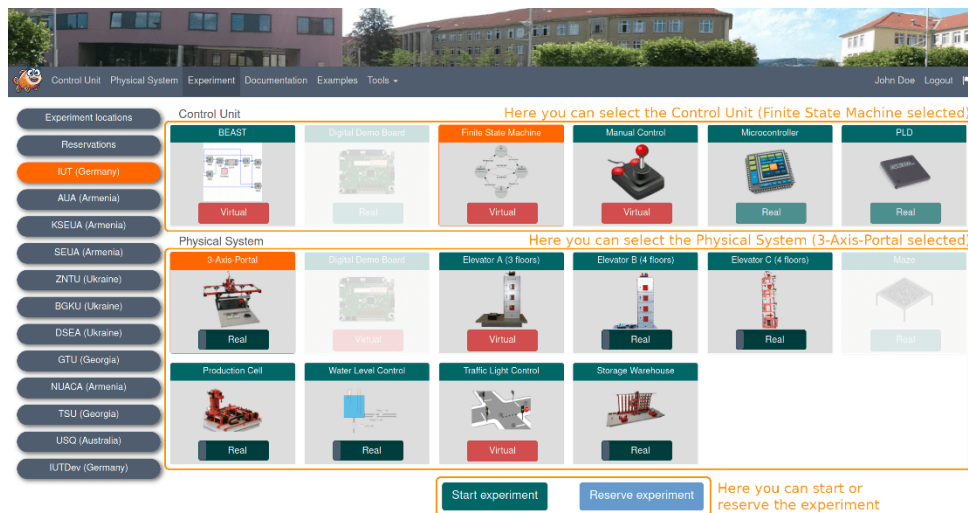


Рисунок 2.8 – Виберіть фізичну систему та блок управління

- вибравши Virtual, ви можете спробувати свій експеримент на симуляції;
- ви можете поєднати кожну фізичну систему з кожним блоком управління, за винятком цифрова демонстраційна дошка. Ви можете поєднувати цифрову демонстраційну плату лише з цифрова демонстраційна дошка;
- вибравши фізичну систему та блок управління, ви можете натиснути на кнопку почати експеримент у кінці сторінки.

Крок 3. Тепер експеримент буде відкрито в новій вкладці (рис. 2.9).

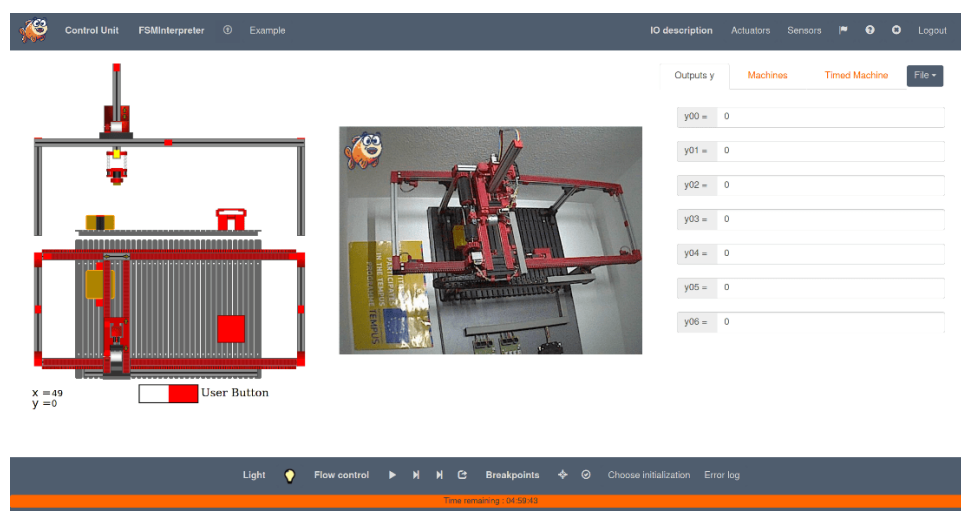


Рисунок 2.9 – Інтерфейс експерименту

Угорі та в кінці сайту є панелі інструментів (рис. 2.10). На верхній панелі інструментів сайту ви можете знайти кнопки для зміни налаштувань, завантаження файлів та відображення опису вводу-виводу зображені в таблиці 2.1.

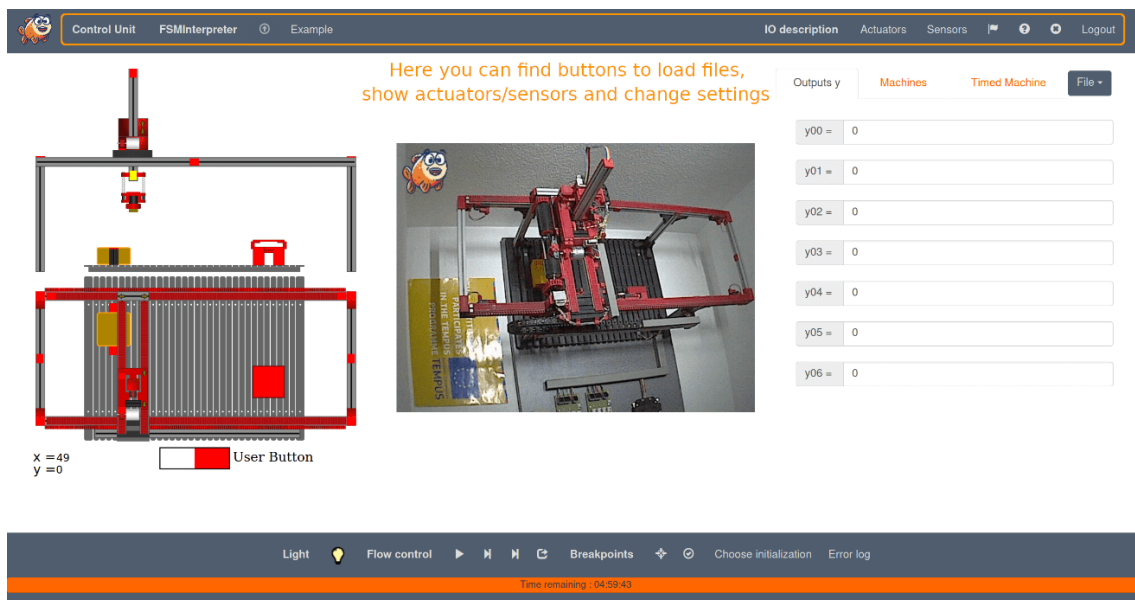


Рисунок 2.10 – Верхня панель інструментів

Сторінку експерименту наведено на рис 2.11.

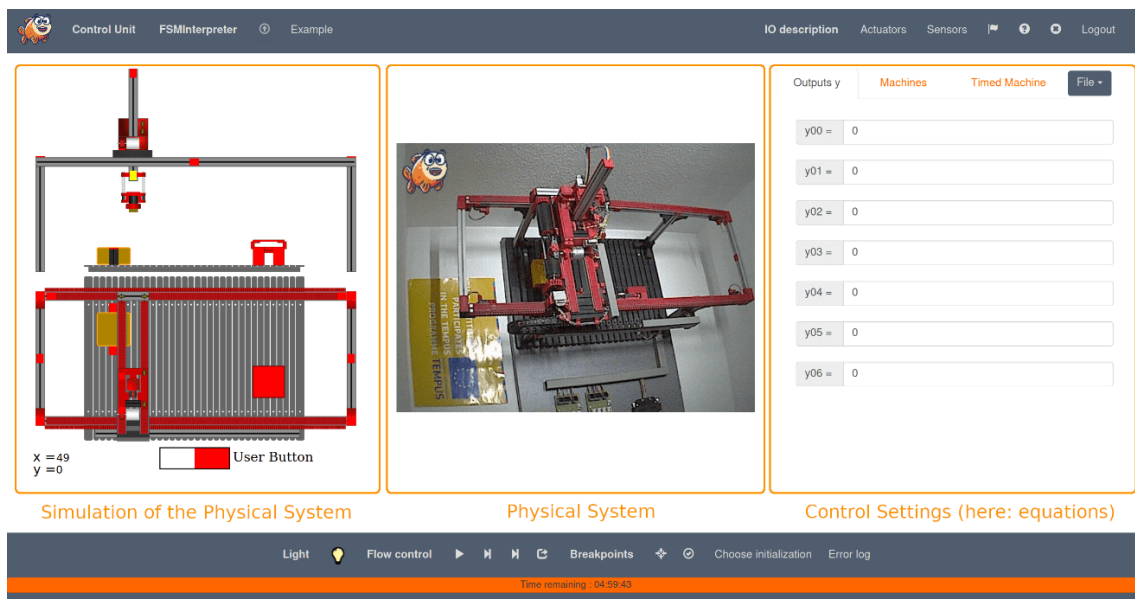
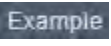
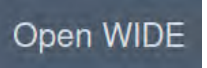
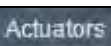
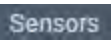
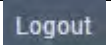


Рисунок 2.11 – Сторінка експерименту

Таблиця 2.1 – Кнопки верхньої панелі інструментів

Позначення	Опис
	За допомогою цієї кнопки ви можете завантажити збережений файл. Зверніть увагу на формат файлу для кожного блоку управління
	За допомогою цієї кнопки ви можете завантажити окремий приклад для обраного блоку управління
	За допомогою цієї кнопки ви можете відкрити інтегроване середовище розробки GOLDi (доступно лише для мікроконтролера та PLD)
	Клацніть на цю кнопку, щоб побачити огляд приводів. Назви виконавчих механізмів мають зелений або червоний колір. Зелений вказує на значення актуатора 1. Червоний означає значення актуатора 0. Коли ви переходите мишею по одному з перелічених виконавчих механізмів, відповідний привід буде виділено у віртуальній системі.
	Клацніть на цю кнопку, щоб побачити огляд датчиків. Назви датчиків мають зелений або червоний колір. Зелений вказує на значення датчика 1. Червоний вказує на значення датчика 0. Коли ви переходите мишею по одному з перерахованих датчиків, відповідний датчик буде виділено у віртуальній системі.
	За допомогою цієї кнопки ви можете змінити мову цілого веб-сайт. Можливо, не все буде перекладено на вибрану вами мову. Ці частини залишатимуться англійською мовою.
	За допомогою цієї кнопки ви можете відкрити довідковий посібник.
	Клацніть на цю кнопку, щоб закрити поточний експеримент.
	Натисніть на цю кнопку, і ви вийдете з GOLDi-веб-сайт. Експеримент також буде закритий.

На нижній панелі інструментів ви можете знайти кнопки для управління експериментом (табл. 2.2). Фізична система зображена на рисунку 2.12.

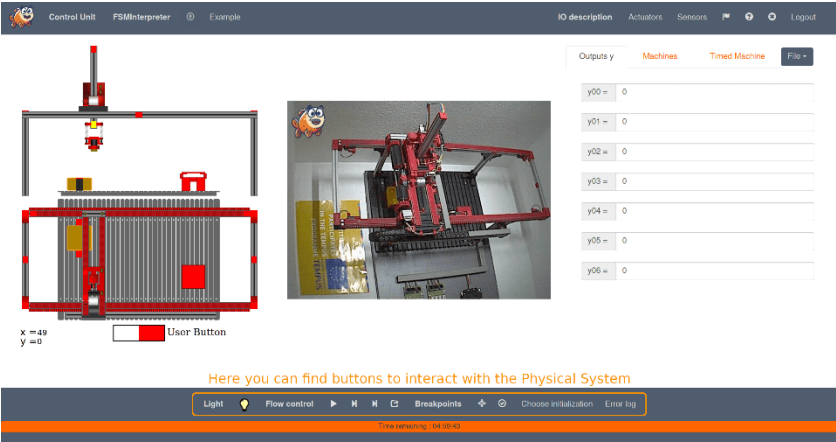


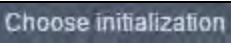
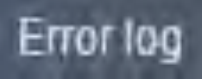
Рисунок 2.12 – Нижня панель інструментів

В середині сторінки ви можете знайти моделювання фізичної системи зліва – реальна модель фізичної системи посередині та управління налаштування (тут: рівняння) для експерименту праворуч.

Таблиця 2.2 – Нижні кнопки панелі інструментів

Позначення	Опис
1	2
	За допомогою цієї кнопки ви можете вмикати та вимикати світло з реальної системи.
	За допомогою цієї кнопки ви можете проводити завантажену програму у фізичній системі.
	За допомогою цієї кнопки моделювання робить крок за кроком вперед через завантажену програму. Він зупиняється кожного разу, коли доходить до датчика.
	За допомогою цієї кнопки моделювання робить крок за кроком вперед через завантажену програму. Він зупиняється кожного разу, коли доходить до виконавчого механізму.

Продовження табл.2.2

1	2
	За допомогою цієї кнопки ви можете скинути налаштування контролера (FPGA або MicroController).
	With this button you can set breakpoints at which the program stops every time when the system reaches them.
	За допомогою цієї кнопки ви можете встановити точки зупинку, на яких програма зупиняється кожного разу, коли система досягає їх.
	За допомогою цієї кнопки ви можете вибрати початковий стан для фізичної системи. Кожна фізична система має свій власний набір можливих початкових станів.
	Натисніть на цю кнопку, і ви побачите огляд помилок. Існують два різні типи помилок, які можуть виникнути: Помилка управління: – Помилка на основі проєкту вашого експерименту Час очікування: Виникає, коли система не реагує протягом певного часу. Це може статися, якщо ваш Інтернет занадто повільний або ваш дизайн має помилку циклу.

2.4.2 Ручне управління

За допомогою ручного управління можна керувати фізичними системами за допомогою натискання на кнопки. Ви можете використовувати ці кнопки для управління рухом системні елементи в різних напрямках та для управління декількома специфічними особливостями систему (рис 2.13).

Якщо ви вибрали 3-осьовий портал як фізичну систему, ви зможете побачити анімацію цієї системи з лівого боку та кнопок управління з правого боку. Портального робота можна переміщати по осі x і по осі y, його

електромагнітний захоплювач можна переміщати в напрямку z. Крім того, магніт захоплювача можна вмикати і вимикати. Для кожного можливого руху є кнопки управління. Якщо на них клацнули, ви можете побачити відповідний рух в анімації.

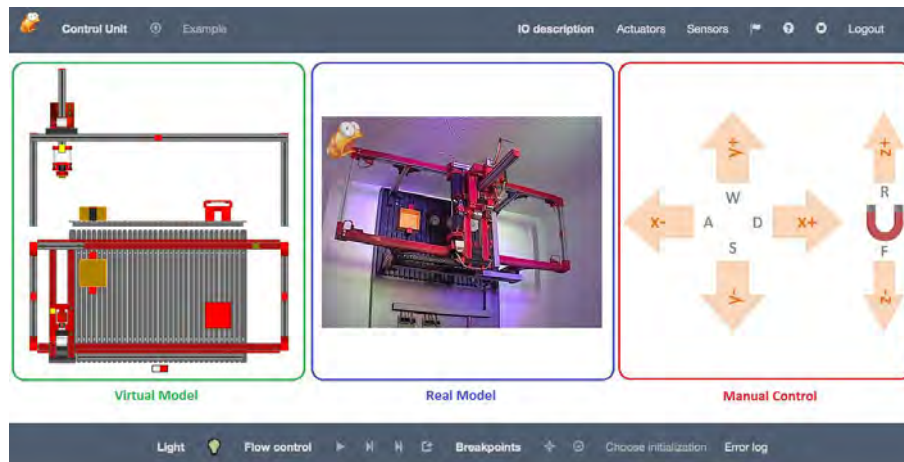


Рисунок 2.13 – Ручне управління 3-Axis-Portal

Замість того, щоб натискати кнопку управління, ви також можете використовувати свою панель очей. Рух у напрямку x та у відповідає клавішам "W-A-S-D", напрямком z відповідає клавішам "R-F", а магніт можна вмикати та вимикати за допомогою клавіші "E" (рис 2.14).

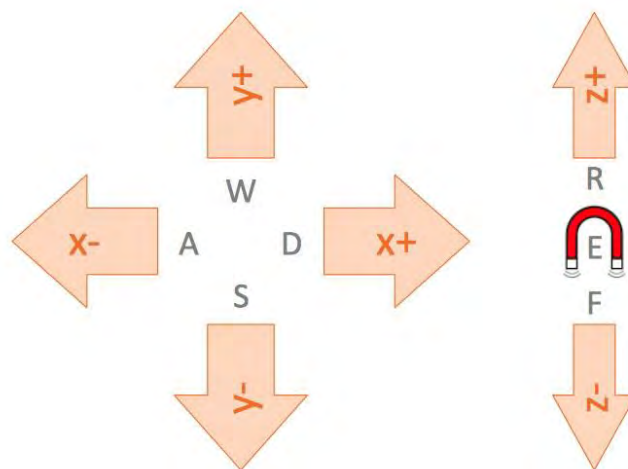


Рисунок 2.14 – Кнопки управління та відповідні клавіші

2.4.3 Використання Digital Demo Board

Digital Demo Board – це спеціальна дошка для швидкого створення прототипів, яка і була розроблена з навчальною метою. Наша дошка швидких прототипів базується на MAX V - 5M1270Z CPLD (Complex Programmable Logic Device) від ALTERA. Крім того, він складається з вводу кнопки, світлодіодні виходи та кілька інших компонентів. Можна скористатися швидким прототипуюча дошка як окрема або веб-версія.

Дошка цифрових демонстрацій складається з таких компонентів:

- MAX V - 5M1270Z (CPLD від Altera (рис 2.15));
- кнопки введення: 8 повзункових перемикачів, 8 кнопок, 2 поворотні шестигранні кодер (рис 2.16);
- LED виходи: 4 7-сегментні дисплеї, 1 LED дисплей (рис 2.17);

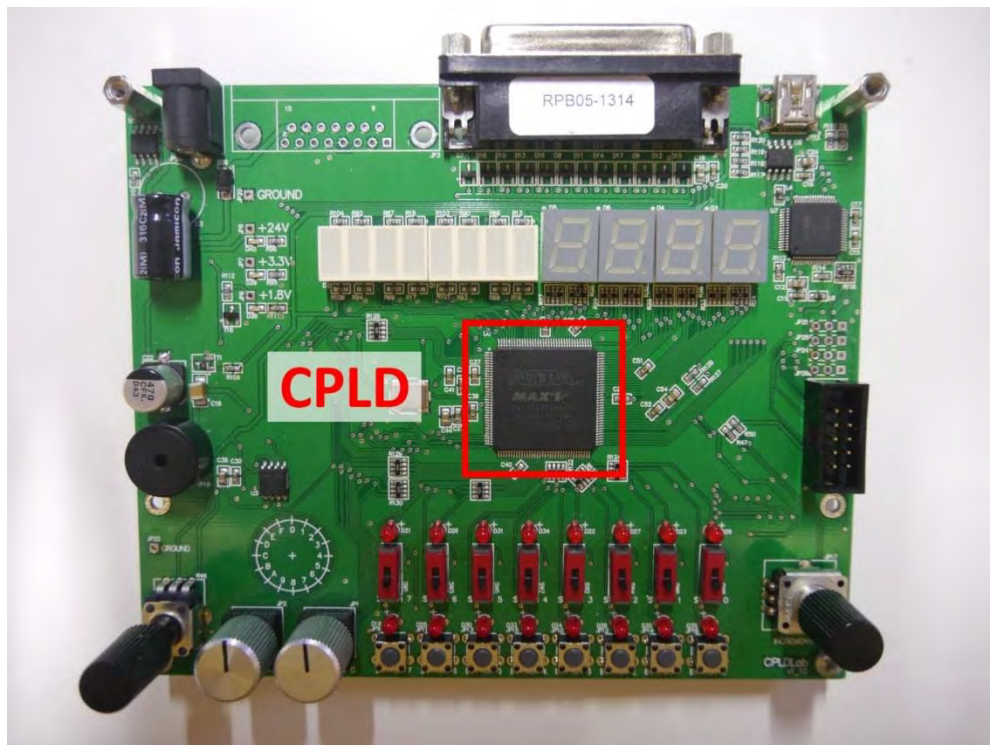


Рисунок 2.15 – CPLD

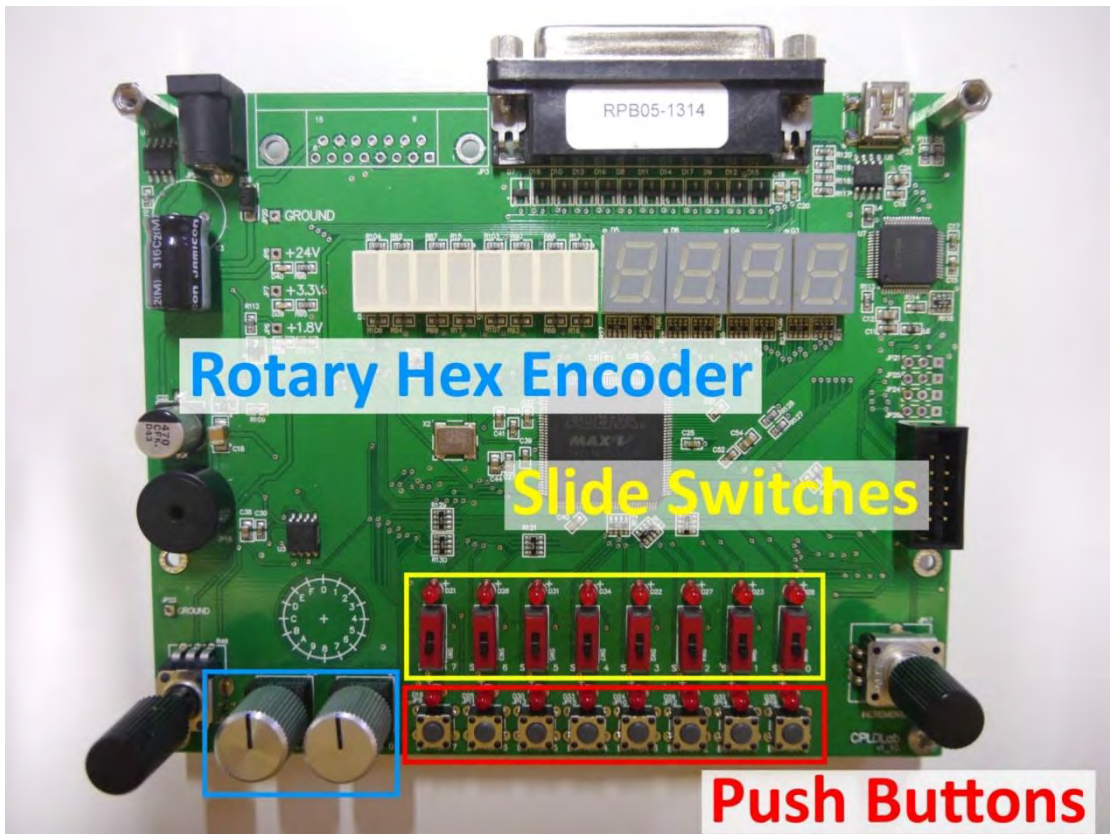


Рисунок 2.16 – Кнопки введения

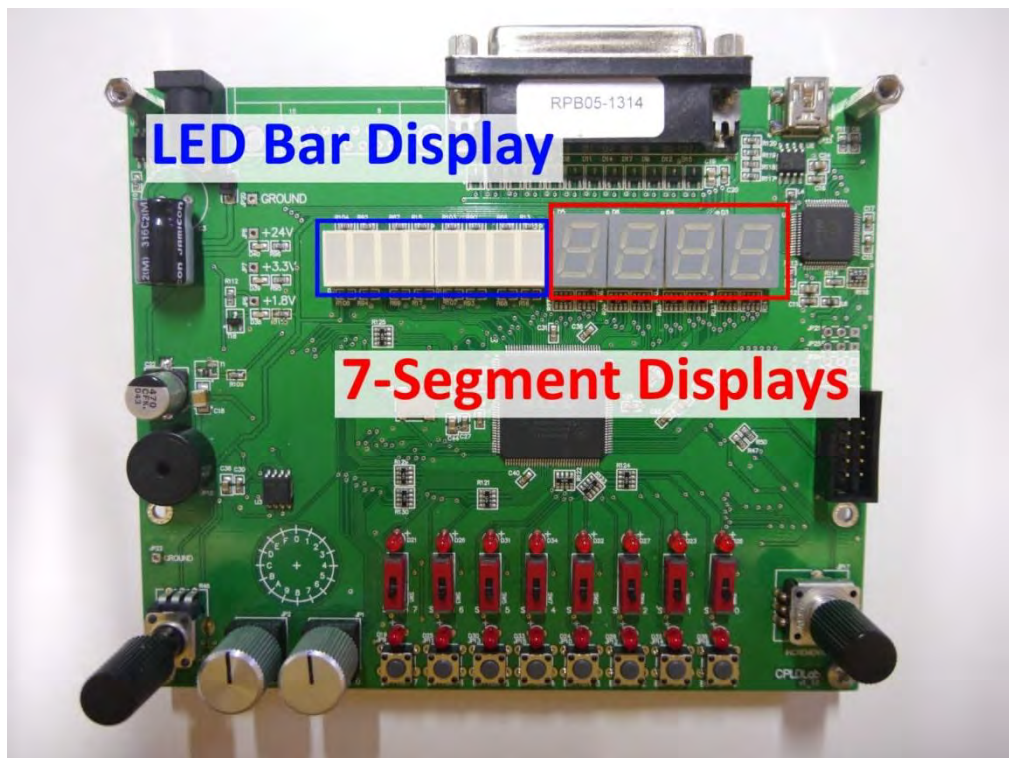


Рисунок 2.17 – LED виходи

– інші компоненти: кристалогенератор 10 МГц, синтезатор частоти, п'єзоелектричний генератор, інкрементний кодер, UART (через USB) (рис 2.18).

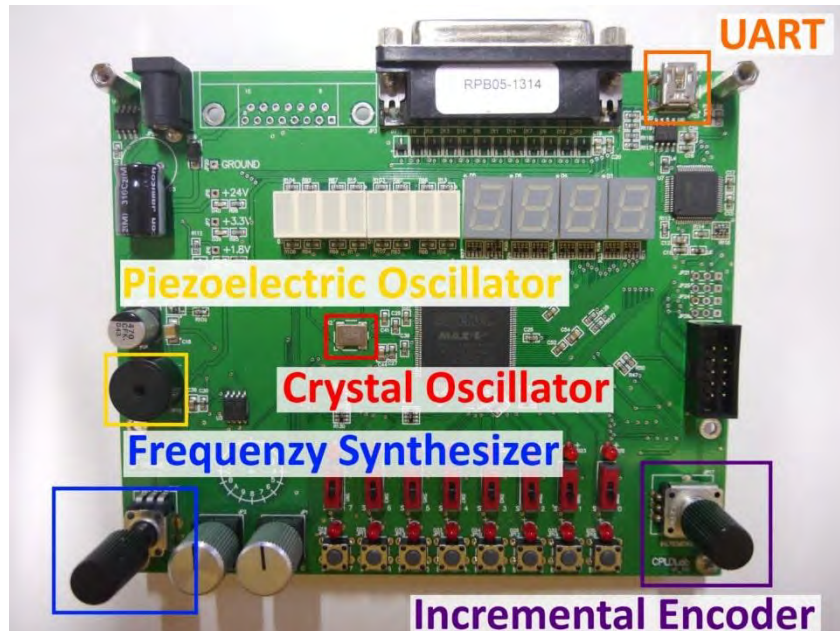


Рисунок 2.18 – Інші компоненти

Призначення контактів різних цифрових демонстраційних плат:

- Pinout-file for Version v1 00: Pinout DDB v100;
- Pinout-file for Version v1 10: Pinout DDB v110;
- Pinout-file for Version v1 11: Pinout DDB v111;
- Pinout-file for Version v1 20: Pinout DDB v120;
- Pinout-file for Version v1 21: Pinout DDB v121;

Тут ви можете завантажити збірник усіх призначень Pin: PDF-файл.

Для створення файлів для програмування CPLD на цифровій демонстраційній платі вам потрібно програмне забезпечення Quartus II Web-Edition від ALTERA. Після створення власного облікового запису користувача на веб-сайті ALTERA можна безкоштовно завантажити ліцензійну версію Quartus II тут: <http://dl.altera.com/?edition=web>

Завантажте останню версію Quartus II (версія 15.0) або принаймні версію 12.0 або пізнішу. Ви можете вирішити, чи хочете ви завантажити комбіновані файли або окремі файли. За допомогою Індивідуальних файлів потрібно лише завантажити на Пристрій підтримку пристроїв MAX II, MAX V, зберегти все в одному каталозі. Після завершення завантаження все буде встановлено, запустивши програму Quartus Setup.

Якщо ви хочете отримати детальний вступ про те, як встановити Quartus II та як поводитися з програмним забезпеченням, ви можете перейти за цими посиланнями:

- Quartus II – Вступ;
- довідник Quartus II;
- встановлення та ліцензування програмного забезпечення Altera.

Найпростіший спосіб використовувати розроблену апаратну платформу - це самостійне рішення для швидкого прототипування цифрових систем (без підключення до Інтернету). Для програмування MAX V CPLD на цій окремій версії плати швидкого прототипу на платі був розміщений додатковий чіп FTDI. Вам потрібно підключити плату прототипування через USB до ПК або ноутбука, щоб завантажити синтезований дизайн CPLD до мікросхеми FTDI на платі прототипування. Цей чіп автоматично програмує CPLD через інтерфейс JTAG.

За допомогою веб-версії цифрової демонстраційної дошки можна завантажити синтезований код CPLD дизайну для автоматичного програмування CPLD на платі швидкого прототипування у віддаленій лабораторії. Тому виберіть Digital Demo Board як блок управління та як фізичну систему, щоб розпочати експеримент, щоб ви могли бачити веб-інтерфейс плати в ЕСП. Цей веб-інтерфейс дозволяє віртуально маніпулювати всіма входами плати швидкого прототипування (повзунковими перемикачами, перемикачами шістнадцяткового кодування, кнопками та інкрементним кодером) у візуальній моделі плати прототипування (у верхній лівій частині). Всі входи реалізовані як елементи керування HTML5 і можуть бути активовані

за допомогою миші взаємодіюче. Зміни негайно надсилаються на дошку швидкого прототипування в лабораторії theremote, і відповідні результати знову відображаються всередині візуальної моделі. Крім того, веб-камера буде використана для спостереження за дошкою швидкого прототипування (у верхній правій частині), щоб спостерігати за результатами дій користувача безпосередньо як реакція у віддаленій лабораторії.

Для використання в Інтернеті плати швидкого прототипу на нижній стороні друкованої плати розміщується додатковий FPGA "для взаємодії" для реалізації зв'язку з віддаленою лабораторною інфраструктурою. Всі входи плати (кнопки, синтезатор, інкрементний кодер та генератор) повинні бути вилучені з друкованої плати і замінені прямим підключенням до виходів "взаємозв'язку" ПЛІС (програмована логічна інтегральна схема), який буде встановлювати всі вхідні сигнали відповідно до вводу користувача через користувацький інтерфейс на клієнтському ПК. Згенеровані виходи плати з прототипуванням можуть бути зчитувані безпосередньо за допомогою "взаємозв'язку" FPGA, не відключаючи жодного світлодіода. FPGA з'єднаний з BPU в пульті дистанційного керування лабораторна інфраструктура.

2.4.4 Програмований логічний пристрій

Програмований логічний пристрій (PLD) є електронним компонентом для інтегральних схем. PLD має невизначену функцію на момент виготовлення, тому перед тим, як PLD може бути використаний в ланцюзі, він повинен бути запрограмований.

У цьому випадку ми використовуємо один з MAX V CPLD (MAX V 5M570ZT100C5).

Архітектура цих CPLD інтегрує зовнішні функції, такі як ash, Оперативна пам'ять, генератори та фазові петлі. Крім того, CPLD ми використовуємо 570 логічних елементів та 100 контактів для певного пакету.

Щоб адаптувати файли для програмування PLD в експерименті, вам потрібно програмне забезпечення Quartus II Web-Edition від ALTERA.

Завантажте останню версію Quartus II (версія 15.0) або принаймні версію 12.0 або пізніше. Ви можете вирішити, чи хочете ви завантажити комбіновані файли або

індивідуальні файли. З Індивідуальними файлами вам потрібно лише завантажити

Пристрій пристроїв MAX II, MAX V підтримує, зберігайте все в одному каталозі. Після завершення завантаження все буде встановлено, запустивши програму Quartus Setup.

Якщо ви хочете отримати докладний вступ, як встановити Quartus II і як це зробити для роботи з програмним забезпеченням ви можете перейти за розділами:

- Quartus II – Вступ;
 - довідник Quartus II;
 - встановлення та ліцензування програмного забезпечення Altera;
 - щоб розпочати експеримент з PLD на GOLDi, вам слід підготувати наступне;
 - Quartus проєкти для обраної вами фізичної системи. У вас теж є вибір;
 - почати з текстового дизайну або з графічного дизайну в Quartus.
- Ось вам;
- можна завантажити приклади проєктів для фізичної системи "3AxisPortal": Textual Design: PortalCrane Mealy, Graphical Design: PortalCrane Mealy.

Ці проєкти включають усі необхідні експерименти. Тільки типовинні змінити UserDesign-файли та TopLevelDesign-файли для адаптації завантажений проєкт у ваш експеримент. Для отримання додаткової інформації прочитайте наступне вступ до проєкту дизайну Quartus на базі "PortalCrane-Mealy" приклад.

У цьому прикладі ми використовуємо проєкт графічного дизайну, щоб продемонструвати, як використовувати Quartus II для GOLDi.

Крок 1. Створення проєкту в Quartus II:

- завантажити rar-архів Графічний дизайн: PortalCrane_Mealy та вилучити файли. Створена папка "Example_TextualDesign" це папка проєкту;
- запустити Quartus II і натисніть на Open Project (рис 2.19). Виберіть qpf-файл "BPU ProgrammableLogicDevice.qpf" у папці проєкту та натисніть "Відкрити";

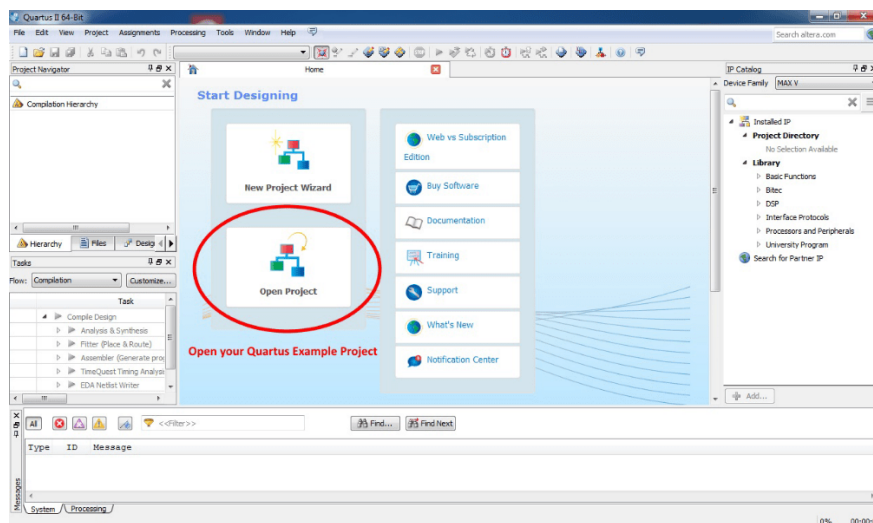


Рисунок 2.19 – Quartus II інтерфейс користувача

- у Project Navigator ви можете побачити файли проєкту Quartus (рис 2.20). Він має в основному таку структуру;
- папка GOLDiInterface містить усі необхідні файли для спілкування в рамках інфраструктури GOLDi. Ці файли є готовий до використання - модифікації не потрібні. Вони будуть включені автоматично;
- папка UserDesign містить фактичний алгоритм управління. Це може використовувати звичайний опис VHDL (наприклад, PortalCrane Mealy FSM.vhd), дизайн як автоматичний графік за допомогою вбудованого редактора графіків FSM (наприклад, PortalCrane Mealy FSM.smf) або

схематичне оформлення (наприклад, Portal-Crane Mealy Schem.bdf). У цьому випадку ми використовуємо схематичне проектування;

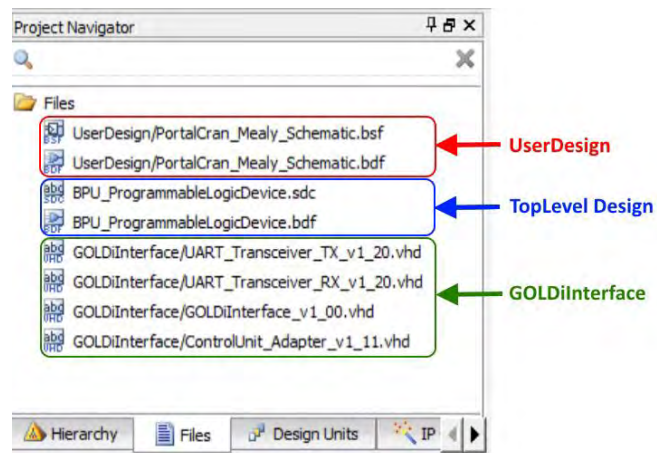


Рисунок 2.20 – Quartus II навігатор проекту

– дизайн TopLevel - це зв'язок між GOLDiInterface та UserDesign. Він може бути реалізований як графічний дизайн блоку (наприклад, BPU ProgrammableLogicDevice.bdf) або як звичайний опис VHDL (наприклад, BPU ProgrammableLogicDevice.vhd). У цьому випадку ми використовуємо графічний дизайн блоку.

Крок 2. Адаптуйте файли до свого проекту. Наступні файли повинні бути адаптовані відповідно до фактичного алгоритму управління та вибрана фізична система:

- UserDesign файл відповідно до використаної специфікації (* .vhd, * .smf або * .bdf);
- файл TopLevelDesign BPU ProgrammableLogicDevice (* .vhd або * .bdf).

Файл UserDesign ./UserDesign/PortalCrane Mealy Schematic.bdf. У цьому файлі ви повинні адаптувати схематичний дизайн до вашого алгоритму управління (рис 2.21). Файл bdf генерує відповідний символ у файлі ./UserDesign/PortalCrane Mealy Schematic.bsf.

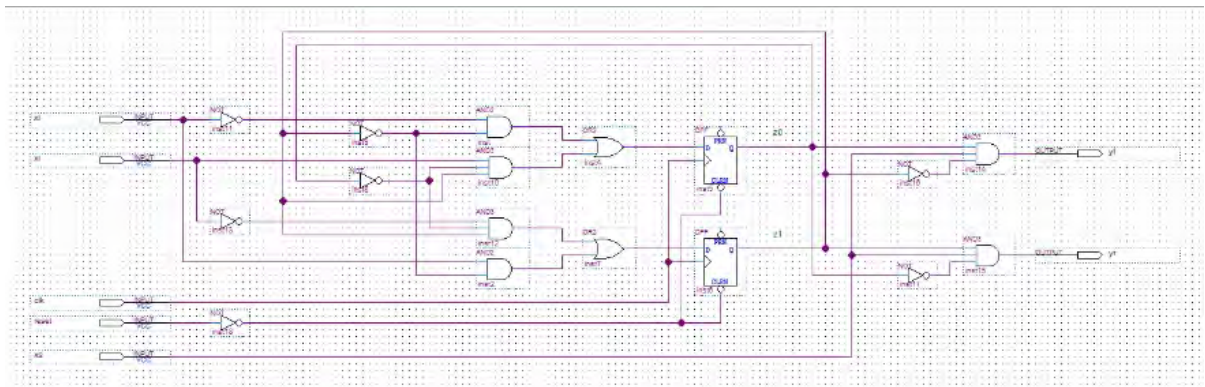


Рисунок 2.21 – Схема UserDesign (*.bdf)

Файл TopLevelDesign ./BPU ProgrammableLogicDevice.bdf. Всередині цього файлу ви повинні вставити символ, який був згенерований з UserDesign у файлі ./UserDesign/PortalCrane Mealy Schematic.bsf (рис 2.22).

Крім того, входи та виходи символу повинні бути взаємопов'язані з відповідними бітами сигналу датчика GOLDiInterface v1 00 та біти приводу відповідно до висновку опису фізичної системи.

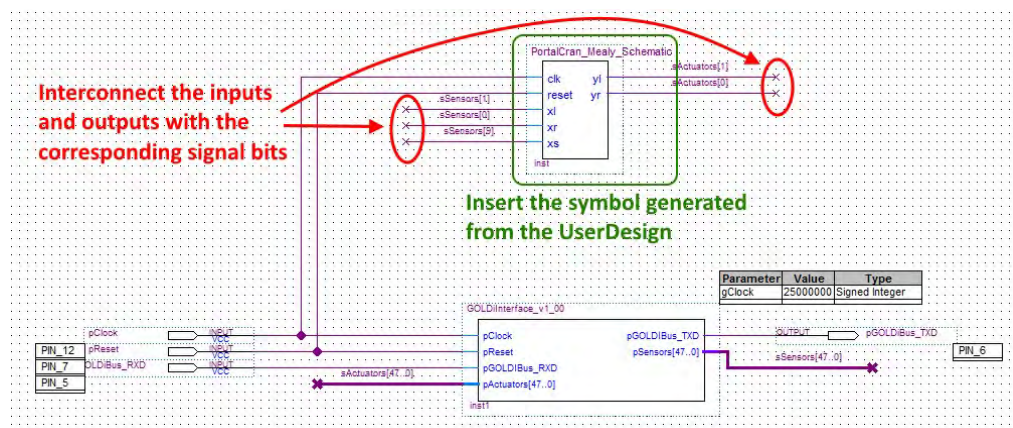


Рисунок 2.22 – TopLevel файл для схематичного проектування (*.bdf)

Файл UserDesign ./UserDesign/PortalCrane Mealy FSM.vhd. Наступний опис VHDL було автоматично сформовано з графіку автоматичного проектування користувача (рис 2.23). UserDesign/PortalCrane Mealy FSM.smf, який ви можете створити за допомогою вбудованого майстра стану машини (рис 2.24).

```

-- Copyright (C) 1991-2014 Altera Corporation. All rights reserved.
-- Generated by Quartus II Version 14.1.0 Build 186 12/03/2014 SJ Web Edition
-- Created on Fri Aug 07 13:12:32 2015

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY PortalCrane_Mealy_FSM IS
    PORT (
        clock    : IN STD_LOGIC;
        reset    : IN STD_LOGIC := '0';
        xl       : IN STD_LOGIC := '0';
        xr       : IN STD_LOGIC := '0';
        xs       : IN STD_LOGIC := '0';
        yl       : OUT STD_LOGIC;
        yr       : OUT STD_LOGIC
    );
END PortalCrane_Mealy_FSM;

ARCHITECTURE BEHAVIOR OF PortalCrane_Mealy_FSM IS
    TYPE type_fstate IS (Z0,Z1,Z2);
    SIGNAL fstate : type_fstate;
    SIGNAL reg_fstate : type_fstate;
BEGIN
    PROCESS (clock,reg_fstate)
    BEGIN
        IF (clock='1' AND clock'event) THEN
            -- State Machine Logic
        END IF;
    END PROCESS;
END BEHAVIOR;

```

Рисунок 2.23 – Текстовий опис UserDesign в VHDL

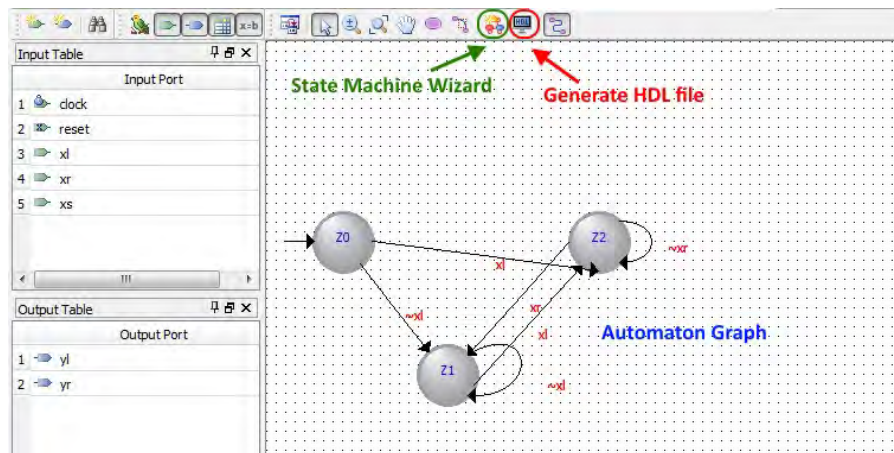


Рисунок 2.24 – UserDesign графік автоматів

Файл TopLevelDesign./BPU ProgrammableLogicDevice.vhd. У файлі TopLevelDesign потрібно змінити опис компонента реалізованого алгоритму управління та відображення портів вхідних та вихідних сигналів у відповідні датчики та біти виконавчого механізму відповідно до опису інтерфейсу фізичного система (рис 2.25).

Крок 3. Завершіть проєкт. Після адаптації файлів вам доведеться скопіювати свій проєкт у Quartus II. Якщо виникають помилки, вам доведеться їх виправити відповідно до повідомлення про помилку.

```

-- #####
-- #####
-- ## User design component description => your implementation part!
-- ##
-- #####

component PortalCrane_Mealy_FSM IS
  PORT
  (
    clock      : IN STD_LOGIC;
    reset      : IN STD_LOGIC := '0';
    xl         : IN STD_LOGIC := '0';
    xr         : IN STD_LOGIC := '0';
    xs         : IN STD_LOGIC := '0';
    yl         : OUT STD_LOGIC;
    yr         : OUT STD_LOGIC;
  );
  component
  description of your
  control algorithm
  END component;
begin
  -- #####
  -- #####
  -- ## User state machine => your implementation part!
  -- ##
  -- #####

  PortalCrane_Instance : PortalCrane_Mealy_FSM
  PORT MAP (
    clock => pClock,
    reset => pReset,
    xl   => sSensors(1),
    xr   => sSensors(0),
    xs   => sSensors(9),
    yl   => sActuators(1),
    yr   => sActuators(0);
  );
  sActuators(47 downto 2) <= (others => '0');

```

Рисунок 2.25 – TopLevel файл для текстового дизайну в VHDL

Якщо ваші дані правильні, ви можете завантажити вихідні дані (рис 2.26) (*.pof) з нової папки "вихідні файли" до експерименту (рис 2.27).

Для отримання додаткової інформації про тестування проєкту у вибраній вами фізичній системі ви можете прочитати опис до "Як користуватися ЕСП".



Рисунок 2.26 – POF-файл у папці "вихідні файли"

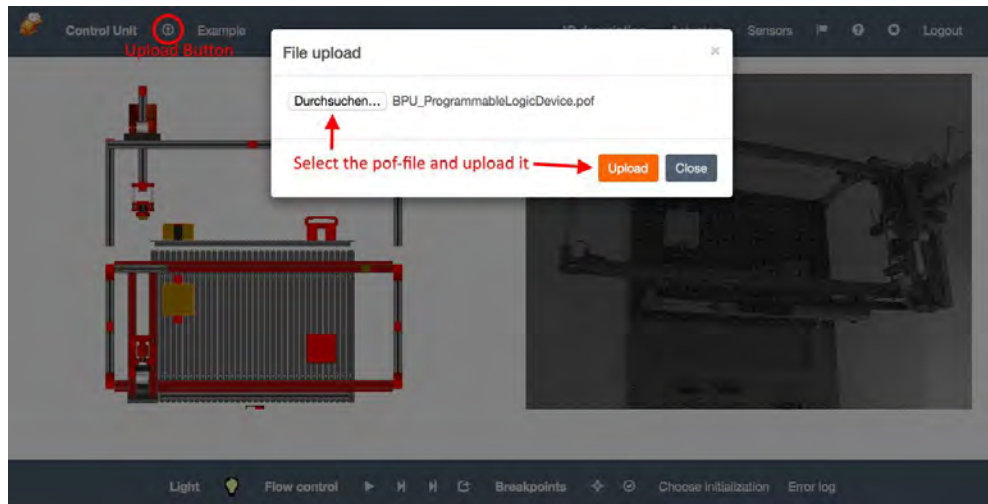


Рисунок 2.27 – Завантаження вихідного файлу до експерименту

2.4.5 Скінчений автомат

Скінчений автомат (FSM) – це модель поведінки, складена з кінцевого набору станів і переходів. У цьому контексті FSM описує послідовну поведінку цифрової системи. Залежно від внутрішнього стану система може реагувати на один і той же вхід різними виходами.

У цьому прикладі одного FSM недостатньо для опису цифрової системи, тому для реалізації вимог ми використовуємо паралельну машину з двома деталями машини.

Для проєктування кінцевого автомата ми використовуємо графічний інтерактивний інструмент FSM

Інструмент GIFT може створити FSM у формі автомата стану, переходу таблиця, машинна таблиця або z-рівняння. Для отримання додаткової інформації про GIFT та різні форми FSM, ви можете прочитати посібник користувача для GIFT. Клацніть тут, щоб відкрити графічний інтерактивний інструмент FSM (GIFT). "3-осьовий портал".

У цьому прикладі ми створюємо паралельну машину, яка описує наступне: поведінки "3-осевого порталу". Кран запускається в положенні X-, Y-положення. По-перше весь кран рухається вправо, поки не досягне потрібного положення на осі X.

Потім кран їде вниз і вгору знову в напрямку Z. Як тільки кран має закінчивши рух вгору, він рухається вліво, поки не досягне лівого положення на осі X.

Нам потрібно створити дві окремі машини в GIFT, які ми згодом адаптуємо один до одного. Перша машина керує рухом по осі X (ліворуч-праворуч). За допомогою діаграми стану та таблиці переходів ми створюємо машину та формуємо відповідні рівняння z- / y. Вхідна змінна "x3" є лише заповнювачем для стану Z0 ("вгору") від машини A1, який буде визначено пізніше. Інші змінні відповідають датчикам і виконавчим механізмам фізичної системи "3AxisPortal" (рис 2.28). Перший апарат A0 (рис 2.29) можна імпортувати в експеримент з ECP Export від GIFT.

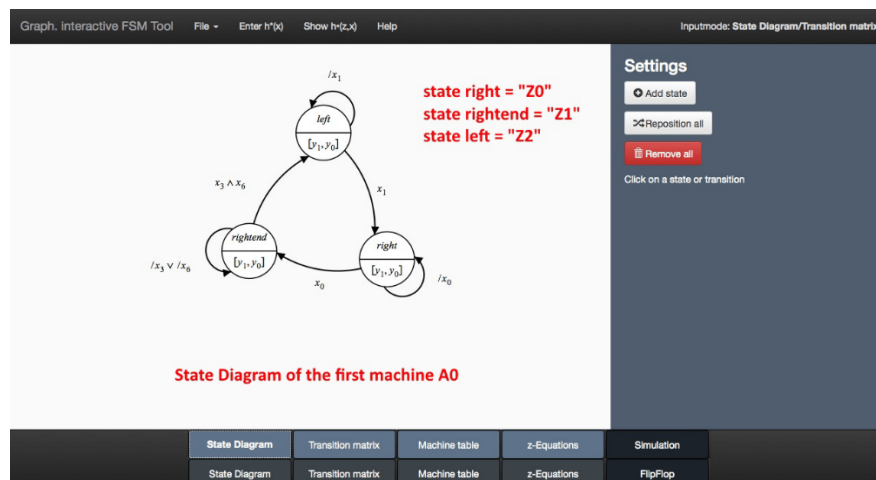


Рисунок 2.28 – Діаграма стану машини A0

	Z0	Z1	Z2	Z3	Y0	Y1
Z0	x0	x0			1	0
Z1		x3 ^ x5	x3 ^ x5		0	0
Z2	x1		x1		0	1
Z3						

Рисунок 2.29 – Таблиця переходів машини A0

Тому натисніть на ECP Export у меню File у GIFT та натисніть на кнопці Імпорт із GIFT в меню Файл в експерименті. Тоді ви повинні назвати свою машину та адаптувати змінні з GIFT до фізичної системи у спливаючому вікні (рис 2.30).

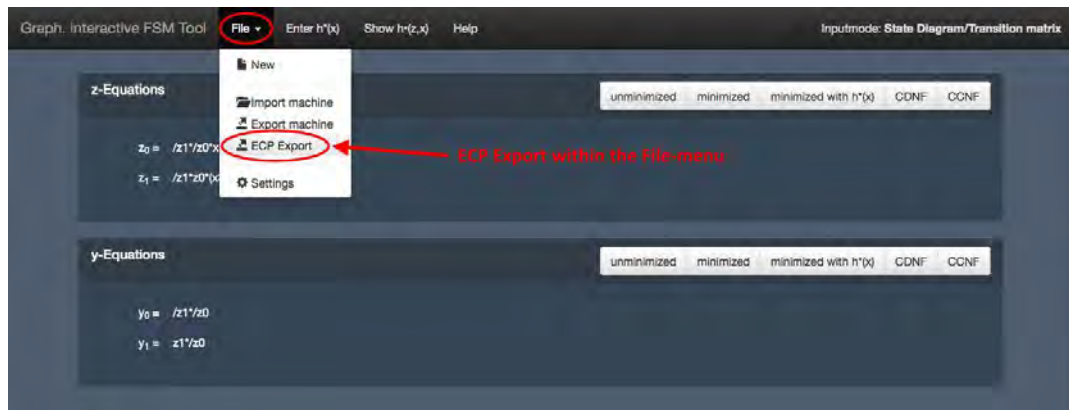


Рисунок 2.30 – ECP- Експорт у GIFT

Назвіть першу машину "A0" і призначте змінні датчикам та торси з системи. У цьому прикладі зв'яжіть змінні x_6 на X_6 , x_1 на X_1 , x_0 до X_0 , y_1 до Y_1 і y_0 до Y_0 . Змінна x_3 все ще є заповнювачем, тому ви можете призначити її будь-якому непотрібному вам датчику системи (тут: x_3 - X_9). Після імпорту першої машини A0 в експеримент (рис 2.31), налаштуйте імпортовану машину (рис 2.32).

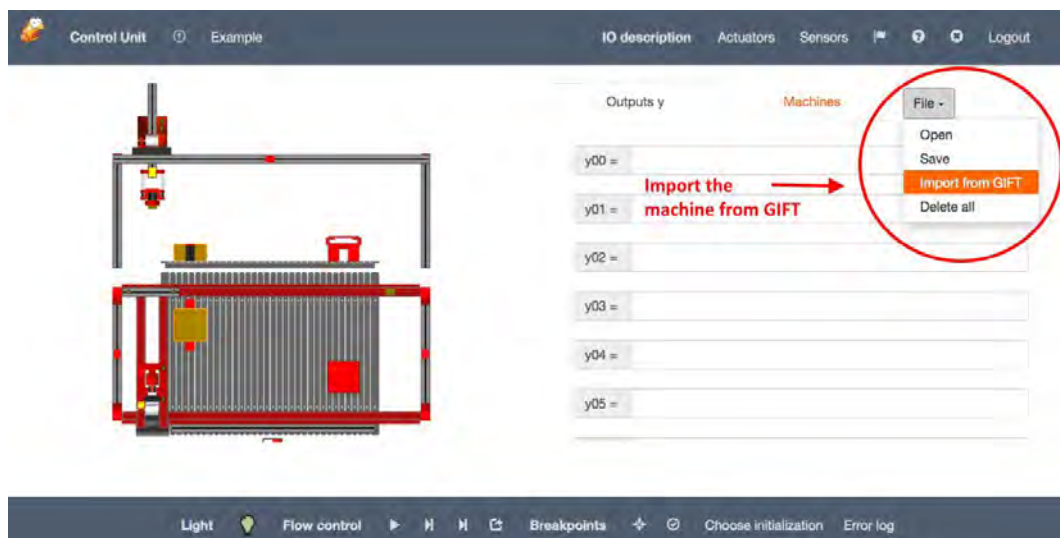


Рисунок 2.31 – Імпорт із GIFT в експерименті

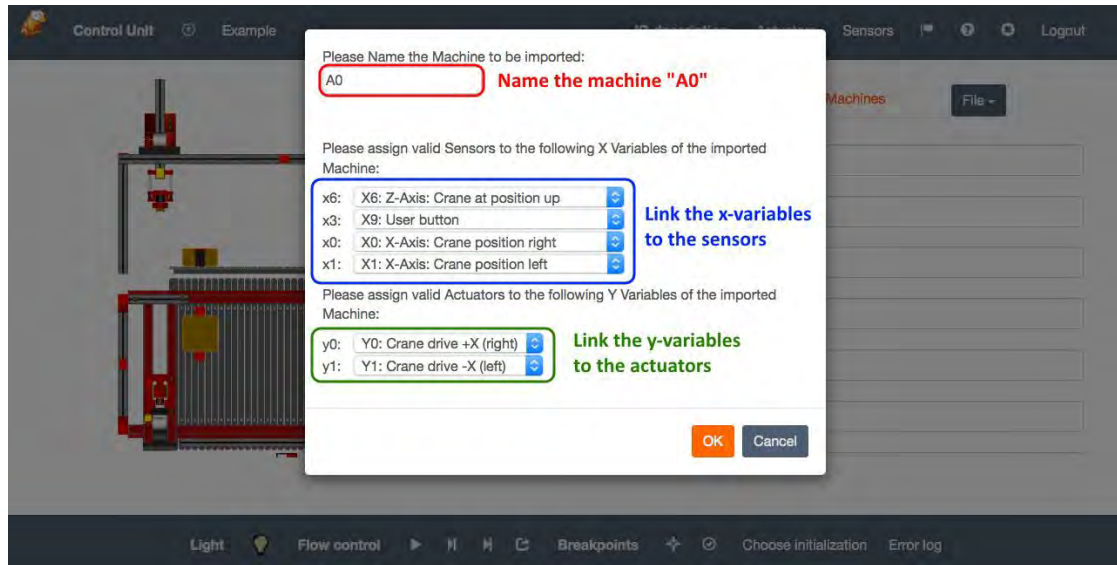


Рисунок 2.32 – Налаштування імпортованої машини в експерименті

Додайте нову машину A1 (рис 2.33) і знову перейдіть на GIFT (рис 2.34), щоб створити другу машину A1 (рис 2.35), подібну до машини A0, яка контролює рух по осі Z (вниз-вгору). Цього разу ви не можете імпортувати машину через ЕСР в експеримент, тому вам потрібно скопіювати рівняння z- / у з GIFT та вставити їх у правильне поле введення в експерименті (рис 2.36).

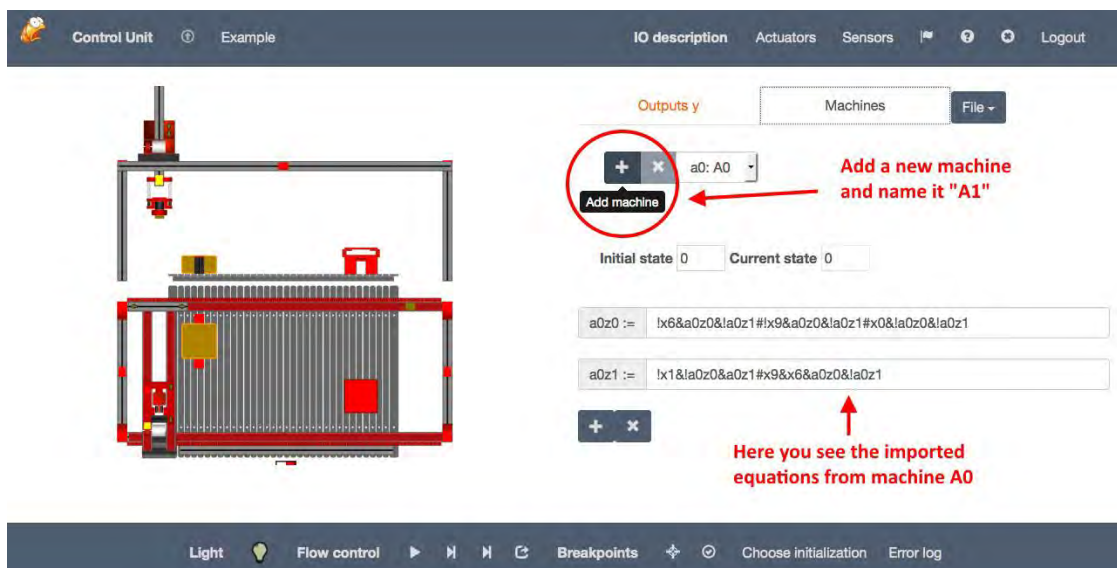


Рисунок 2.33 – Додайте нову машину (A1)

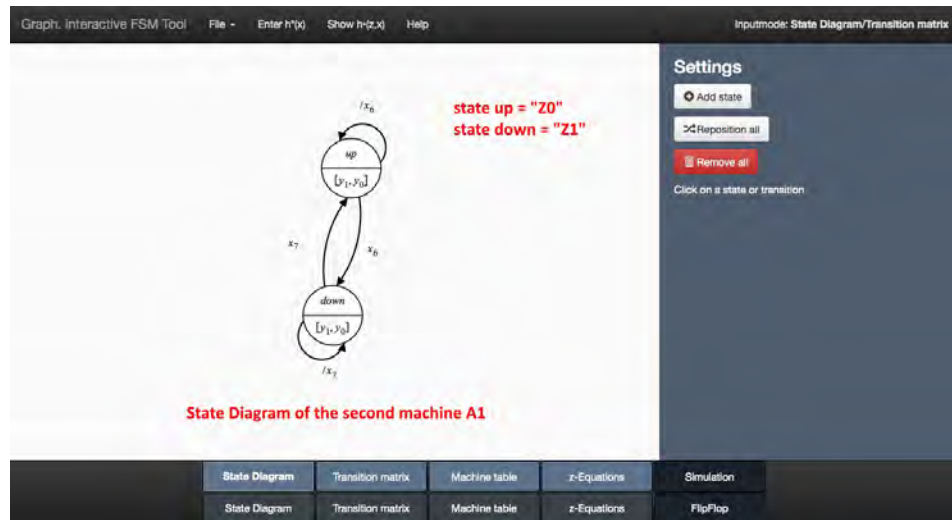


Рисунок 2.34 – Діаграма стану машини A1

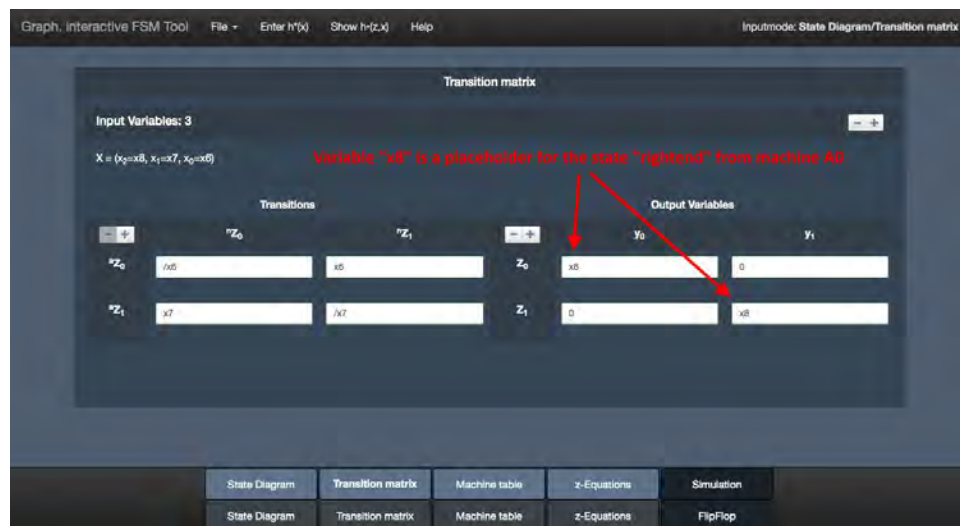


Рисунок 2.35 – Таблиця переходів машини A1

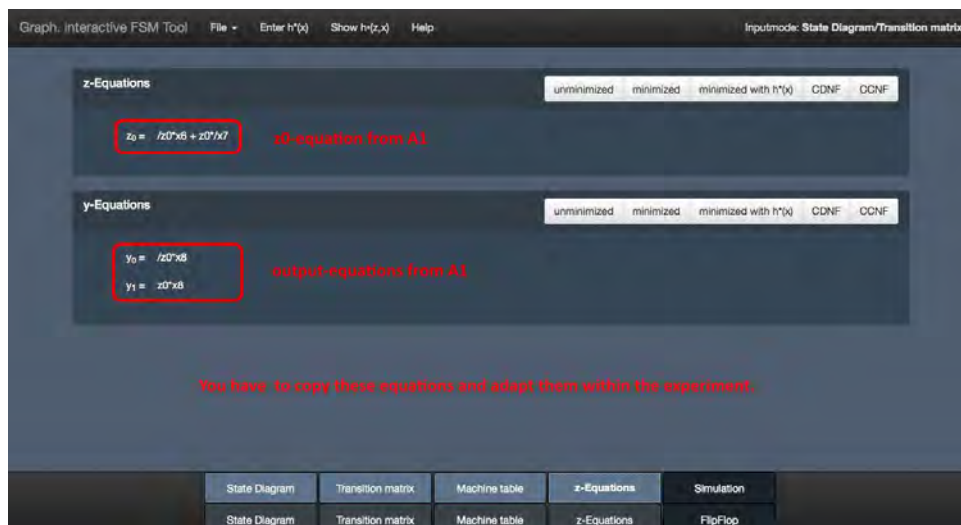


Рисунок 2.36 – Z / Y-рівняння машини A1

Тепер вам потрібно скопіювати рівняння $z0$ у поле " $a1z0$ " - з машини A1 (рис. 2.37), рівняння $y0$ - у поле " $y04$ ", а рівняння $y1$ - у поле " $y05$ " із "Виходів у" та адаптувати рівняння (рис. 2.38). Змініть " $z0$ " на " $a1z0$ ", "/" на "!", "*" на "&", "+" на "#" та " $x8$ " на "! $A0z1$ & $a0z0$ ". Крім того, вам потрібно змінити заповнювач-змінну " $x9$ " з машини A0 на "! $A1z0$ " (рис. 2.39). Коли ви створили та адаптували обидві машини A0 та A1 в експерименті (рис 2.40), ви можете перевірити працездатність паралельної машини, натиснувши кнопку "Пуск".



Рисунок 2.37 – Скопіюйте $z0$ -рівняння з A1

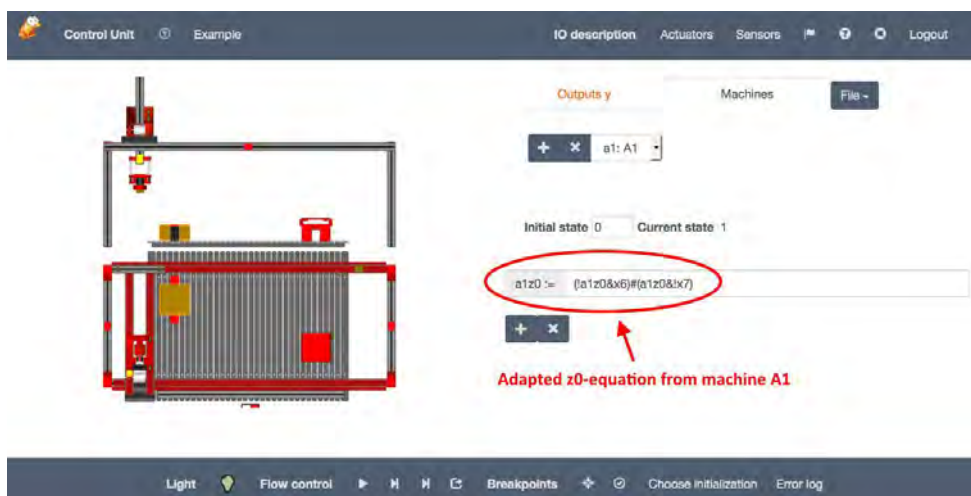


Рисунок 2.38 – Адаптуйте $z0$ -рівняння з A1

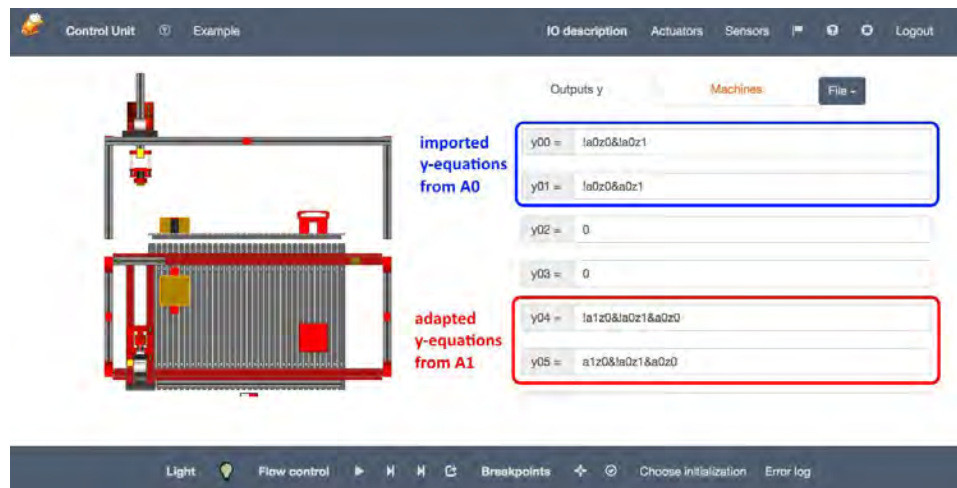


Рисунок 2.39 – Скопіюйте та адаптуйте рівняння вихідних даних

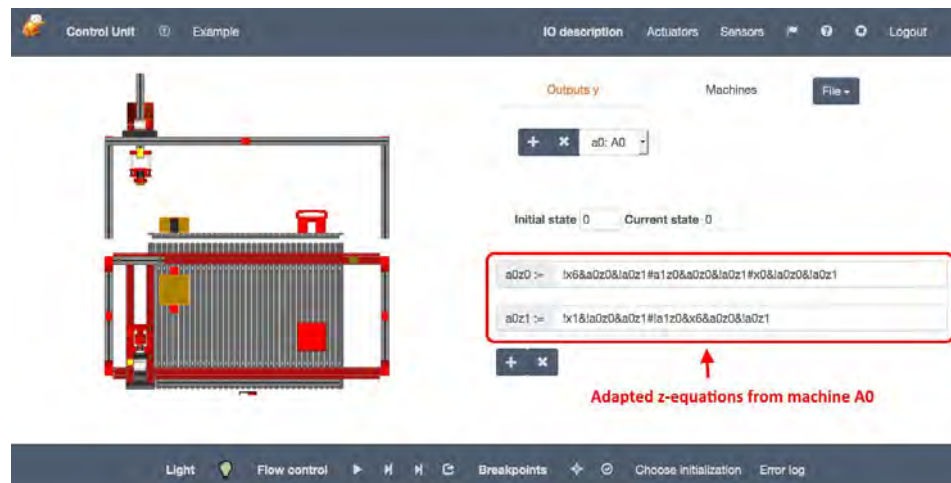


Рисунок 2.40 – Адаптовані z-рівняння машини A0

Якщо все правильно, ви можете побачити описаний рух "3-осьового порталу" в анімації ліворуч.

2.5 Постановка задач дипломної роботи

Гібридна лабораторія GOLDi використовується для проведення досліджень і для надання практичного досвіду для розробки вбудованої електроніки.

Експеримент 3-Axis-Portal, Elevator, Production Cell, Digital Demo Board та інші можуть керуватися за допомогою різних керуючих пристроїв. Але

керування за допомогою мікроконтролерів не реалізовано. Тому, тема роботи є актуальною.

Мета роботи - розробка програмного забезпечення гібридної лабораторії GOLDi для дистанційного управління експериментами за допомогою мікроконтролера Arduino.

Задачі роботи:

- розробити алгоритм мікроконтролерного управління експериментом;
- обрати апаратні та програмні засоби розробки;
- виконати програмну реалізацію;
- розробити програмну документацію.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ВІДДАЛЕНОГО ЕКСПЕРИМЕНТУ НА ARDUINO

3.1 Алгоритм експерименту

Розроблений алгоритм (рис.3.1) ілюструє основні етапи реалізації експерименту 3AxisPortal. По перше треба створити датчики та актуатори, прописати стани машини Мура, створити потрібні функції, інструкції та визначитися з додатковими змінними.

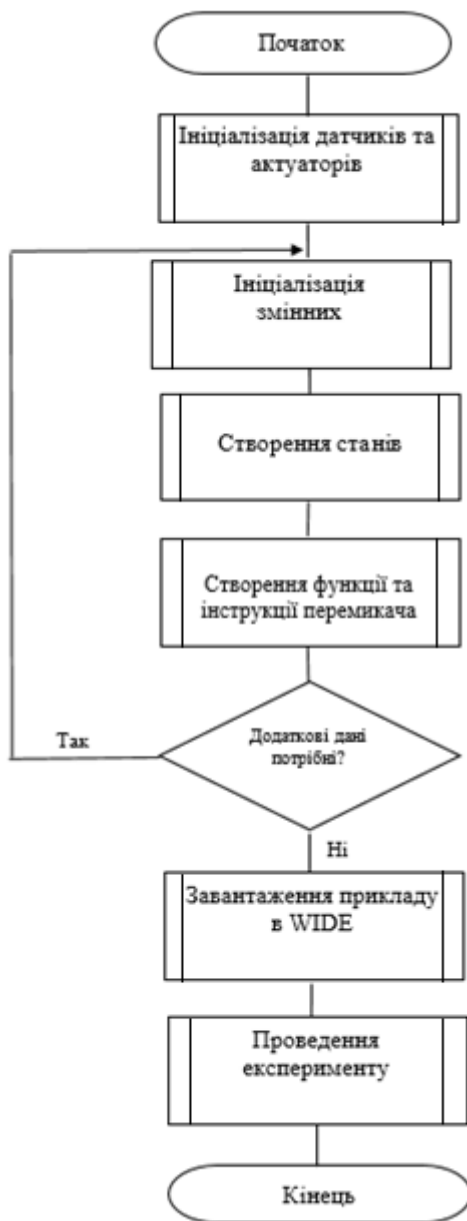


Рисунок 3.1 – Алгоритм програми

3.2 Вибір інструментарію розробки

Інтегроване середовище розробки Arduino це багатоплатформовий додаток на Java, що включає в себе редактор коду, компілятор і модуль передачі прошивки в плату. Середовище розробки засноване на мові програмування Processing та спроектоване для програмування новачками, не знайомими близько з розробкою програмного забезпечення. Мова програмування аналогічна мові Wiring. Загалом, це C++, доповнений деякими бібліотеками. Програми обробляються за допомогою препроцесора, а потім компілюються за допомогою AVR-GCC [7].

3.2.1 Arduino IDE

Arduino IDE – інтегроване середовище розробки для Windows, MacOS і Linux, розроблена на Cі і C ++, призначена для створення і завантаження програм на Arduino-сумісні плати, а також на плати інших виробників (рис.3.2) [10].

Вихідний код для середовища випущений під загальнодоступною ліцензією GNU версії 2. Підтримує мови Cі і C ++ з використанням спеціальних правил структурування коду. Arduino IDE надає бібліотеку програмного забезпечення з проекту Wiring, яка надає безліч загальних процедур введення і виведення. Для написаного користувачем коду потрібні тільки дві базові функції для запуску ескізу і основного циклу програми, які скомпільовані і пов'язані з заглушкою програми `main ()` в виконувану циклічну програму з ланцюжком інструментів GNU, також включеною в дистрибутив IDE. Використовує програму `avrdude` для перетворення коду, в текстовий файл в шістнадцятковій кодуванні, який завантажується в плату Arduino програмою-завантажувачем у вбудованому програмному забезпеченні плати [10].

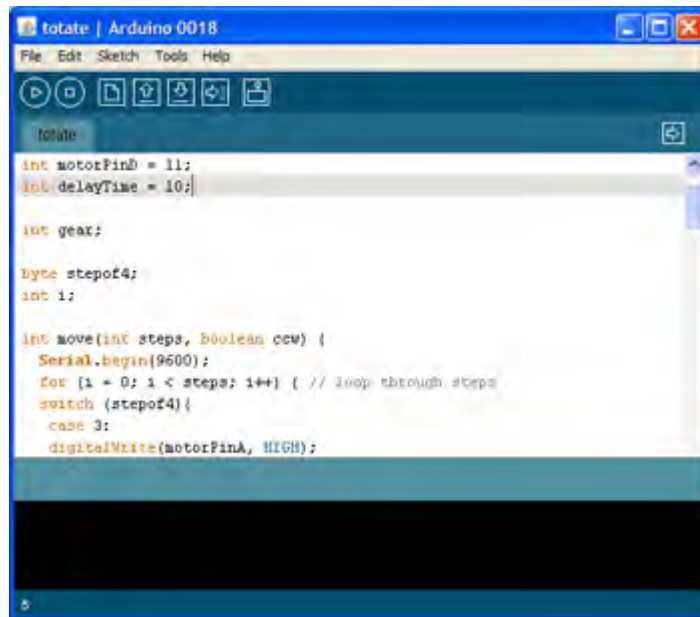


Рисунок 3.2 – Інтерфейс Arduino IDE

З ростом популярності Arduino інші постачальники в якості програмної платформи почали впроваджувати призначені для користувача компілятори і інструменти з відкритим вихідним кодом (ядра), які можуть створювати і завантажувати ескізи в інші мікроконтролери, що не підтримуються офіційною лінійкою мікроконтролерів Arduino [10].

3.2.2 Visual Micro

Visual Micro – це просте в установці розширення для Microsoft Visual Studio (рис. 3.3), яка веде інтегрованого середовища розробки на платформі Windows [11].

Легко інтегрується в Visual Studio і дозволяє почати програмування Arduino за секунди. Функції:

- повністю сумісний з Arduino IDE, кожен проект Arduino IDE є проектом Visual Micro, і навпаки;
- за лаштунками використовуються оригінальні інструменти збірки від Arduino;

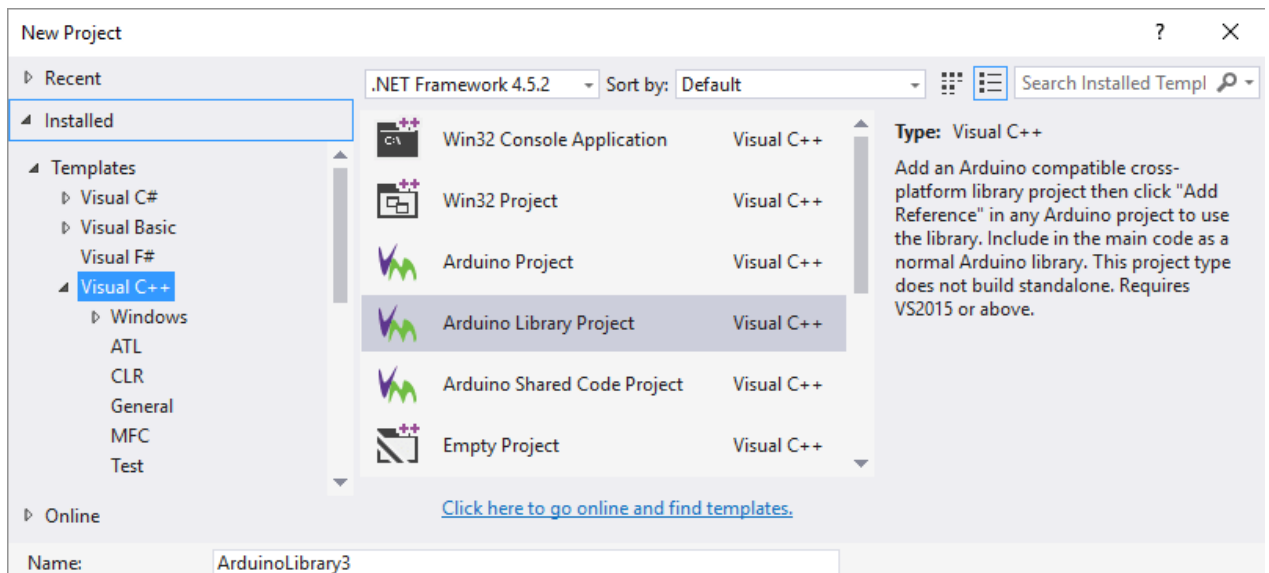


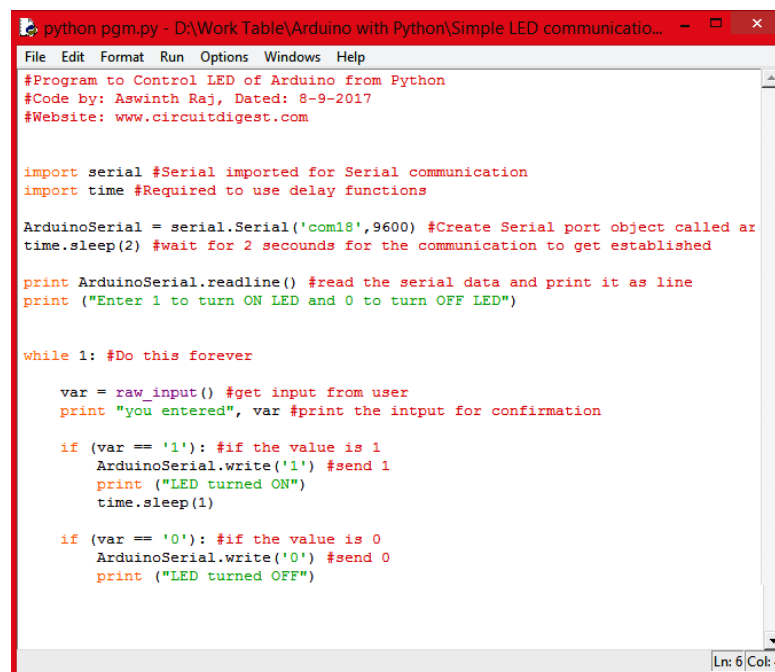
Рисунок 3.3 – Інтерфейс Visual Micro

- настраюється призначений для користувача інтерфейс;
- підтримка Intellisense: автоматично пропонує, автоматично доповнює імена функцій і класів у міру введення (приклад);
- своєчасна перевірка синтаксису: виявляє, виділяє і пояснює синтаксичні помилки негайно, без компіляції;
- підтримка декількох файлів .ino, .cpp для кожного проекту, вкладених папок проекту для більших проектів;
- підтримує широкий спектр плат: Arduinos і сумісні, Teensy, Sparkfun, Adafruit, Intel;
- унікальний відладчик (точки зупину, точки трасування, відстеження і зміна змінних);
- не потрібно спеціального устаткування, працює з USB або Wi-Fi
- потужний менеджер бібліотек;
- завантажити сторонні бібліотеки і інтегруйте їх в Visual Micro;
- шукати бібліотеки в Інтернеті;
- board Manager: Завантажити підтримку для нових плат прямо з IDE, установка не потрібно;
- працюйте з декількома дошками одночасно;

- створення нових бібліотек Arduino в один клік;
- підтримка GIT і TFS.

3.2.3 Pluto

Pluto – це бібліотека Python, побудована поверх Firmata, але з більш простими, інтуїтивно зрозумілими API та іншими магічними методами для швидкого створення прототипів. Python – це найпростіша мова для вивчення, і Pluto має намір допомогти початківцям, таким як студенти та викладачі, навчитися програмуванню та електроніці, а дизайнерам, які працюють із засобами масової інформації та фізичними обчисленнями, легко прототипувати свої ідеї без необхідності щоразу завантажувати та складати код (рис. 3.4) [11].



```
python pgm.py - D:\Work Table\Arduino with Python\Simple LED communicatio...
File Edit Format Run Options Windows Help

#Program to Control LED of Arduino from Python
#Code by: Aswinth Raj, Dated: 8-9-2017
#Website: www.circuitdigest.com

import serial #Serial imported for Serial communication
import time #Required to use delay functions

ArduinoSerial = serial.Serial('com18',9600) #Create Serial port object called ar
time.sleep(2) #wait for 2 seconds for the communication to get established

print ArduinoSerial.readline() #read the serial data and print it as line
print ("Enter 1 to turn ON LED and 0 to turn OFF LED")

while 1: #Do this forever

    var = raw_input() #get input from user
    print "you entered", var #print the input for confirmation

    if (var == '1'): #if the value is 1
        ArduinoSerial.write('1') #send 1
        print ("LED turned ON")
        time.sleep(1)

    if (var == '0'): #if the value is 0
        ArduinoSerial.write('0') #send 0
        print ("LED turned OFF")
```

Рисунок 3.4 – Приклад коду на мові Python [11]

3.3 Реалізація програмного забезпечення

З Arduino можна використовувати додаткові функції, включаючи digitalRead () та digitalWrite (). Вони були адаптовані до експериментів GOLDi

Remotelab наступним чином: `digitalRead (i)`: повертає, чи активовано датчик `xi` `digitalWrite (i, HIGH / LOW)`: деактивує / активує привід `yi` Перелік усіх датчиків та учасників усіх експериментів можна знайти на веб-сайті www.goldi-labs.net у розділі Фізична система. Там також можна знайти файли PDF з інформацією про експерименти.

Крім того, є функції `setup ()` і `loop ()`, які потрібні в кожна програма Arduino. Вони використовуються наступним чином:

`setup ()`: виконується один раз на початку програми і може бути використаний, наприклад ініціалізувати змінні.

`loop ()`: викликається неодноразово і використовується для основної логіки програми.

При створенні Arduino-проекту буде створений `.ino`-файл із цим самим назва проекту. Цей файл буде містити пусту версію функцій `setup ()` та `loop ()`.

Якщо приклад базується на автоматі, має бути наступна структура, що використовується. для `3AxisPortal`. На початку, всі датчики та актуатори повинні бути визначені з відповідними назвами (рис. 3.5).

```
// #####
// # Sensors of 3AxisPortal for use with digitalWrite(Sensor)
// #####
#define Crane_Position_Right 0
#define Crane_Position_Left 1
#define Crane_Reference_Position_X 2
#define Crane_Position_Back 3
#define Crane_Position_Front 4
#define Crane_Reference_Position_Y 5
#define Crane_Position_Up 6
#define Crane_Position_Down 7
#define Proximity_Sensor 8
#define User_Button 9

// #####
// # Actuators of 3AxisPortal for use with digitalWrite(Actuator, LOW/HIGH)
// #####
#define Drive_Right 0
#define Drive_Left 1
#define Drive_Back 2
#define Drive_Front 3
#define Crane_Up 4
#define Crane_Down 5
#define Magnet 6
```

Рисунок 3.5 – Оголошення датчиків та актуаторів

Потім слід створити перелік станів, необхідних для автомата, а також його об'єкт (рис. 3.6).

```
// #####
// # States of the Mealy machine
// #####
typedef enum
{
    Z0_WaitForFallingEdge,           // Z0 – wait for 1/0 edge
    Z1_DriveXMinusYMinus,           // Z1 – drive to left/down
    Z2_DriveYPlusXPlus               // Z2 – drive to up/right
} AutomatStates_t;

AutomatStates_t State;
```

Рисунок 3.6 – Створення станів

У функції `setup ()` значення згаданого об'єкта встановлюється в початковий стан автомата (рис. 3.7).

```
// #####
// # Initialize the state of the Mealy machine with start state
// #####
void setup()
{
    State = Z0_WaitForFallingEdge;
}
```

Рисунок 3.7 – Створення функції

У функціональному циклі `()` повинна бути інструкція перемикача, яка реалізує автомат (рис. 3.8).

```
// #####
// # Update the current state of the Mealy machine
// #####
void loop()
{
    switch (State)
    {
        case Z0_WaitForFallingEdge:
        {
            digitalWrite(Drive_Right, LOW);
            digitalWrite(Drive_Left, LOW);
            digitalWrite(Drive_Back, LOW);
            digitalWrite(Drive_Front, LOW);
            digitalWrite(Crane_Up, LOW);
            digitalWrite(Crane_Down, LOW);
            digitalWrite(Magnet, LOW);

            if (!digitalRead(User_Button))
            {
                State = Z1_DriveXMinusYMinus;
            }

            break;
        }

        case ...
    }
}
```

Рисунок 3.8 – Інструкція перемикача

Для прикладів, які не засновані на автоматах, все-таки рекомендується перерахувати виконавчі механізми та сенсори на початку файлу, щоб полегшити користувачеві роботу. Після цих визначень можуть бути створені додаткові змінні для використання в програмі. Їх можна ініціалізувати за допомогою функції `setup()`. Функціональний цикл `()` як і раніше виконує ті самі цілі, що і раніше, але не потрібно дотримуватися наведених вище вказівок. Могуть бути створені додаткові функції.

Для того, щоб завантажити новий приклад в WIDE, Zip-файл, що містить файли прикладу, потрібно помістити у правильний підкаталог прикладів. Приклади каталогів містять підкаталоги для кожного експерименту, для якого існують приклади. Ці каталоги знову мають підкаталоги з іменем мови програмування, на якій написаний приклад. Тож Zip-файл прикладу, написаного в Arduino для 3AxisPortal, слід помістити в каталог `examples / 3AxisPortal / Arduino /`. Zip-файл не повинен містити папку, оскільки папка створюється з ім'ям Zip-файлу, коли користувач завантажує відповідний приклад.

Текст програми наведено в Додатку А.

4 КЕРІВНИЦТВО КОРИСТУВАЧА

Заходимо до сайту GOLDi та входимо до облікового запису (рис 4.1).



Рисунок 4.1 – Головна сторінка GOLDi

Реєструємося або вводимо пароль і логін і заходимо в обліковий запис (рис. 4.2).

Please login with a registered GOLDi account or a TU Ilmenau LDAP account.

You have been logged in successfully!

Username	goldi
Password	Password

Register
Forgot password?
Login

Рисунок 4.2 – поле вводу логіну та пароля

Вибираємо Tools і WIDE Standalone (рис. 4.3)

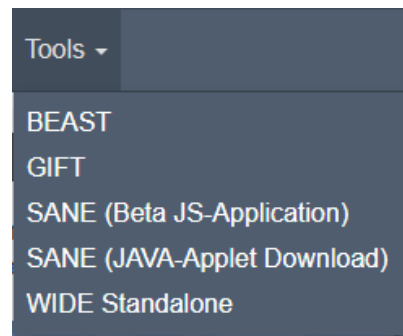


Рисунок 4.3 – Поле Tools

Створюємо новий проєкт (рис. 4.4).

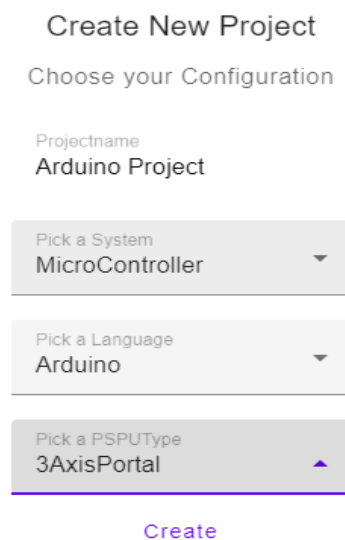


Рисунок 4.4 – Вікно створення проєкту

Завантажуємо проєкт (рис. 4.5).

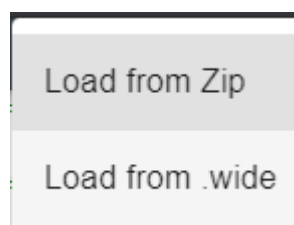


Рисунок 4.5 – Вікно вибору файлу для завантаження

Завантажений код у WIDE наведено на рис. 4.6.

```

1 // =====
2 // # Sensors of 3AxisPortal for use with digitalRead(Sensor) #
3 // =====
4 #define Crane_Position_Right 0
5 #define Crane_Position_Left 1
6 #define Crane_Reference_Position_X 2
7 #define Crane_Position_Back 3
8 #define Crane_Position_Front 4
9 #define Crane_Reference_Position_Y 5
10 #define Crane_Position_Up 6
11 #define Crane_Position_Down 7
12 #define Proximity_Sensor 8
13 #define User_Button 9
14
15 // =====
16 // # Actuators of 3AxisPortal for use with digitalWrite(Actuator, LOW/HIGH) #
17 // =====
18 #define Drive_Right 0
19 #define Drive_Left 1
20 #define Drive_Back 2
21 #define Drive_Front 3
22 #define Crane_Up 4
23 #define Crane_Down 5
24 #define Magnet 6
25
26 // =====
27 // # States of the Moore machine #
28 // =====
29 typedef enum
30 {
31     Z0_WaitForFallingEdge, // Z0 - wait for 1/0 edge
32     Z1_DriveXMinusVMinus, // Z1 - drive to X- and Y-
33     Z2_DriveXMinus, // Z2 - drive to X-
34     Z3_DriveVMinus, // Z3 - drive to Y-
35     Z4_DriveVPlus, // Z4 - drive to Y+
36     Z5_DriveXPlus, // Z5 - drive to X+
37     Z6_WaitForUserSwitch // Z6 - wait for UserSwitch = 1
38 } AutomatStates_t;
39
40 AutomatStates_t State;
41
42 // =====
43 // # Initialize the state of the Moore machine with start state #
44 // =====
45 void setup()
46 {

```

Рисунок 4.6 – Завантажений код програми

Компілюємо та завантажуюємо проєкт (рис. 4.7).

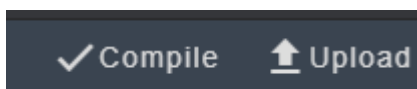


Рисунок 4.7 – Кнопки компіляції та завантаження проєкту

Чекаємо завантаження файлів (рис. 4.8).

Uploading file to control unit...



Flashing control unit...



Рисунок 4.8 – Вікно завантаження проєкту

Запускаємо проєкт (рис. 4.9).

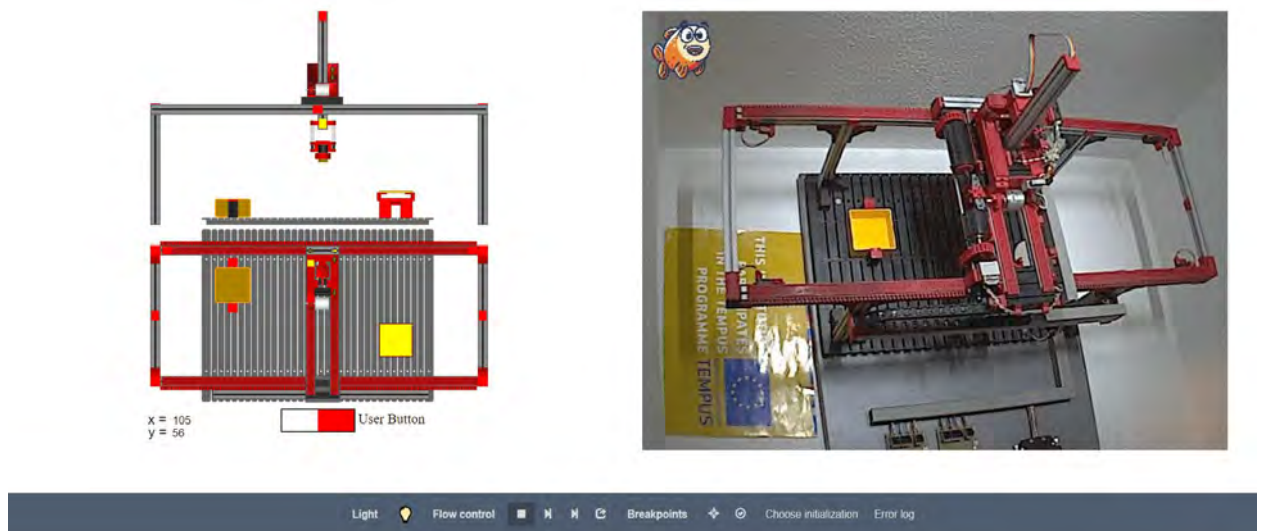


Рисунок 4.9 – Результат роботи

ВИСНОВКИ

В результаті виконання дипломної роботи виконано аналіз особливостей реалізації вбудованої електроніки на різноманітних платформах, зокрема, Freeduino, Raspberry Pi, Arduino.

Проведено порівняння функціональних можливостей та характеристик існуючих версій платформи Arduino.

Зроблено огляд побудови мережі GOLDi, особливостей гібридної лабораторії GOLDi її структури, прикладів використання лабораторії для проведення експериментів 3-Axis-Portal, Elevator B, Production Cell.

Розроблено програмний код для мікроконтролерного управління експериментами дистанційної гібридної лабораторії GOLDi на Arduino, зокрема для експерименту 3-Axis-Portal.

В подальшій роботі перелік експериментів може бути розширений.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Встраиваемые системы [Электрон. ресурс]. – Режим доступа: <https://controleng.ru/programmnye-sredstva/vstraivaemy-e-sistemy-upravleniya/>.
2. Freeduino – Arduino совместимый микроконтроллер [Электрон. ресурс]. – Режим доступа: <http://freeduino.ru/arduino/index.html>.
3. Что такое Raspberry Pi [Электрон. ресурс]. – Режим доступа: <http://www.armlinux.ru/что-такое-raspberry-pi/>.
4. Raspberry Pi [Электрон. ресурс]. – Режим доступа: https://uk.wikipedia.org/wiki/Raspberry_Pi.
5. Что такое Arduino и как начать изучение? [Электрон. ресурс]. – Режим доступа: <https://ampermarket.kz/base/arduino/>.
6. Что такое Arduino [Электрон. ресурс]. – Режим доступа: <https://arduinoplus.ru/chto-takoe-arduino/>.
7. Arduino [Электрон. ресурс]. – Режим доступа: <https://uk.wikipedia.org/wiki/Arduino>.
8. Обзор контроллеров семейства Arduino [Электрон. ресурс]. – Режим доступа: https://ampermarket.kz/base/arduino_family/.
9. Пристрій керування [Электрон. ресурс]. – Режим доступа: https://uk.wikipedia.org/wiki/Пристрій_керування.
10. Arduino IDE [Электрон. ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Arduino_IDE.
11. Development Tools [Electronic resource]. – Access mode: <https://playground.arduino.cc/Main/DevelopmentTools/#VMVS>.
12. The Grid of Online Laboratory Devices Ilmenau (GOLDi) [Electronic resource]. – Access mode: <https://www.goldi-labs.net/index.php?Site=1&Hint=SessionTimeout>.

ДОДАТОК А

Текст програми

```

//#####
//#####
// # Sensors of 3AxisPortal for use with digitalRead(Sensor) #
//#####
//#####

#define Crane_Position_Right      0
#define Crane_Position_Left      1
#define Crane_Reference_Position_X 2
#define Crane_Position_Back      3
#define Crane_Position_Front     4
#define Crane_Reference_Position_Y 5
#define Crane_Position_Up        6
#define Crane_Position_Down      7
#define Proximity_Sensor        8
#define User_Button              9

//#####
//#####
// # Actuators of 3AxisPortal for use with digitalWrite(Actuator, LOW/HIGH) #
//#####
//#####

#define Drive_Right      0
#define Drive_Left      1
#define Drive_Back      2
#define Drive_Front     3
#define Crane_Up        4
#define Crane_Down      5
#define Magnet          6

//#####
//#####
// # States of the Moore machine #
//#####
//#####

typedef enum
{
    Z0_WaitForFallingEdge, // Z0 - wait for 1/0
    edge
    Z1_DriveXMinusYMinus, // Z1 - drive to X-
    and Y-
    Z2_DriveXMinus, // Z2 - drive to X-
    Z3_DriveYMinus, // Z3 - drive to Y-
    Z4_DriveYPlus, // Z4 - drive to Y+
    Z5_DriveXPlus, // Z5 - drive to X+
    Z6_WaitForUserSwich // Z6 - wait for
    UserSwich = 1
} AutomatStates_t;

```

```
AutomatStates_t State;
```

```

//#####
//#####
//#   Initialize the state of the Moore machine with start state           #
//#####
//#####

```

```

void setup()
{
    State = Z0_WaitForFallingEdge;
}

```

```

//#####
//#####
// #   Update the current state of the Moore machine                       #
//#####
//#####

```

```
//#####
```

```

void loop()
{
    switch (State)
    {
        case Z0_WaitForFallingEdge:
// Z0
        {
            digitalWrite
(Drive_Right,  LOW);
            digitalWrite
(Drive_Left,   LOW);
            digitalWrite
(Drive_Back,   LOW);
            digitalWrite
(Drive_Front,  LOW);
            digitalWrite
(Crane_Up,     LOW);
            digitalWrite
(Crane_Down,   LOW);
            digitalWrite
(Magnet,       LOW);

            if(!digitalRead(User_Button) && !digitalRead(Crane_Position_Left) &&
!digitalRead(Crane_Position_Front))
            {
                State = Z1_DriveXMinusYMinus;
            }
        }
    }
}

```

```

        else if(!digitalRead(User_Button) &&
!digitalRead(Crane_Position_Left) && digitalRead(Crane_Position_Front))
        {
            State = Z2_DriveXMinus;
        }
        else if(!digitalRead(User_Button) && digitalRead(Crane_Position_Left)
&& !digitalRead(Crane_Position_Front))
        {
            State = Z3_DriveYMinus;
        }
        else if(!digitalRead(User_Button) && digitalRead(Crane_Position_Left)
&& digitalRead(Crane_Position_Front))
        {
            State = Z4_DriveYPlus;
        }

        break;
    }

    case Z1_DriveXMinusYMinus:
// Z1
    {
        digitalWrite
(Drive_Right,    LOW);
        digitalWrite
(Drive_Left,    HIGH);
        digitalWrite
(Drive_Back,    LOW);
        digitalWrite
(Drive_Front,   HIGH);
        digitalWrite
(Crane_Up,      LOW);
        digitalWrite
(Crane_Down,    LOW);
        digitalWrite
(Magnet,        LOW);

        if (!digitalRead(Crane_Position_Left) &&
digitalRead(Crane_Position_Front))
        {
            State = Z2_DriveXMinus;
        }
        else if (digitalRead(Crane_Position_Left) &&
!digitalRead(Crane_Position_Front))
        {
            State = Z3_DriveYMinus;
        }
        else if (digitalRead(Crane_Position_Left) &&
digitalRead(Crane_Position_Front))
        {
            State = Z4_DriveYPlus;

```

```

        }

        break;
    }

    case Z2_DriveXMinus:
// Z2
    {
        digitalWrite
(Drive_Right, LOW);
        digitalWrite
(Drive_Left, HIGH);
        digitalWrite
(Drive_Back, LOW);
        digitalWrite
(Drive_Front, LOW);
        digitalWrite
(Crane_Up, LOW);
        digitalWrite
(Crane_Down, LOW);
        digitalWrite
(Magnet, LOW);

        if (digitalRead(Crane_Position_Left))
        {
            State = Z4_DriveYPlus;
        }

        break;
    }

    case Z3_DriveYMinus:
// Z3
    {
        digitalWrite
(Drive_Right, LOW);
        digitalWrite
(Drive_Left, LOW);
        digitalWrite
(Drive_Back, LOW);
        digitalWrite
(Drive_Front, HIGH);
        digitalWrite
(Crane_Up, LOW);
        digitalWrite
(Crane_Down, LOW);
        digitalWrite
(Magnet, LOW);

        if (digitalRead(Crane_Position_Front))
        {

```

```

        State = Z4_DriveYPlus;
    }

    break;
}

case Z4_DriveYPlus:
// Z4
{
    digitalWrite
(Drive_Right,  LOW);
    digitalWrite
(Drive_Left,   LOW);
    digitalWrite
(Drive_Back,   HIGH);
    digitalWrite
(Drive_Front,  LOW);
    digitalWrite
(Crane_Up,     LOW);
    digitalWrite
(Crane_Down,   LOW);
    digitalWrite
(Magnet,       LOW);

    if (digitalRead(Crane_Position_Back))
    {
        State = Z5_DriveXPlus;
    }

    break;
}

case Z5_DriveXPlus:
// Z5
{
    digitalWrite(Drive_Right,  HIGH);
    digitalWrite
(Drive_Left,   LOW);
    digitalWrite
(Drive_Back,   LOW);
    digitalWrite
(Drive_Front,  LOW);
    digitalWrite
(Crane_Up,     LOW);
    digitalWrite
(Crane_Down,   LOW);
    digitalWrite
(Magnet,       LOW);

    if (digitalRead(Crane_Position_Right) &&
digitalRead(User_Button))

```

```

        {
            State = Z0_WaitForFallingEdge;
        }
        else if(digitalRead(Crane_Position_Right) &&
!digitalRead(User_Button))
        {
            State = Z6_WaitForUserSwich;
        }

        break;
    }

    case Z6_WaitForUserSwich:
// Z6
    {
        digitalWrite
(Drive_Right,    LOW);
        digitalWrite
(Drive_Left,    LOW);
        digitalWrite
(Drive_Back,    LOW);
        digitalWrite
(Drive_Front,   LOW);
        digitalWrite
(Crane_Up,      LOW);
        digitalWrite
(Crane_Down,    LOW);
        digitalWrite
(Magnet,        LOW);

        if (digitalRead(User_Button))
        {
            State = Z0_WaitForFallingEdge;
        }

        break;
    }
}
}
}

```

ДОДАТОК Б

Слайди презентації

Національний університет "Запорізька політехніка"

Кафедра програмних засобів

Дипломна кваліфікаційна робота

Програмне забезпечення гібридної лабораторії GOLDi

Виконав
ст. гр. КНТ-218сп

В.М. Медков

Керівник
к.т.н., доцент

О.М. Гладкова

Рисунок Б.1 – Слайд 1

Актуальність роботи



Вебсайт GOLDi



Структура GOLDi

Рисунок Б.2 – Слайд 2

Мета та задачі дипломної роботи

Мета роботи – розробка програмного забезпечення гібридної лабораторії GOLDi для дистанційного управління експериментами за допомогою мікроконтролера Arduino.

Задачі роботи:

- розробити алгоритм мікроконтролерного управління експериментом;
- обрати апаратні та програмні засоби розробки;
- виконати програмну реалізацію;
- розробити програмну документацію.

3

Рисунок Б.3 – Слайд 3

Вибір апаратної платформи

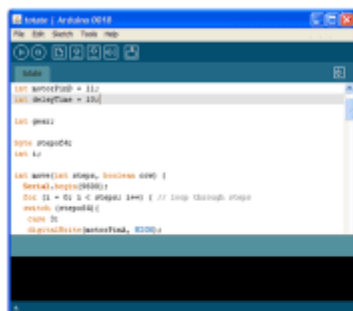
Плата Arduino	Мікроконтролер	Робоча напруга (В)	Цифрові порти/піни	Внутрішній ШІМ	Аналогові порти/каналів	Максимальна швидкість Пито живлення-аналогові (кА)	Розмір чипа (мм)	RAM (КБ)	EEPROM-пам'ять (КБ)	Тактова частота (МГц)	Габарити (мм)	USB-порт
Uno	ATmega328	5	34	6	6	40	32	2	1	16	69x33	USB A-B
Leonardo	ATmega32u4	5	20	7	12	40	32	2.5	1	16	73x33	micro-USB
Nano	ATmega328	5	34	6	8	40	32	2	1	16	18x45	mini-USB
Mega	ATmega2560	5	54	14	16	40	256	8	4	16	102x53	USB A-B
Due	Atmel SAM3X8E ARM Cortex-M3	3.3	54	12	12	800	512	96	×	84	102x53	micro-USB
Mini	ATmega328	5	16	6	6	40	32	2	1	16	30x18	USB-Serial
Micro	ATmega32u4	5	20	7	12	40	32	2.5	1	16	48x18	micro-USB
Pro	Atmel SAMD11	3.3	20	12	6	7	256	32	×	48	66x33	micro-USB
Lo-Power	ATmega328p	2.7 - 5.5	20	6	6	40	32	2	1	8	Ø 30	USB-Serial



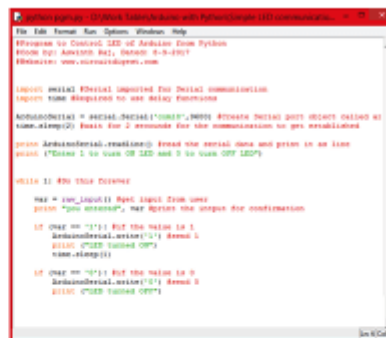
4

Рисунок Б.4 – Слайд 4

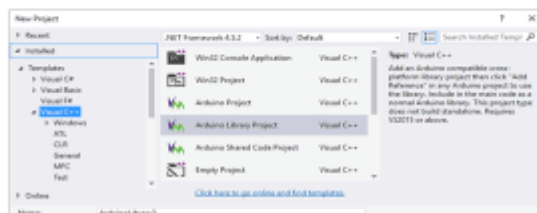
Вибір інструментарію розробки



Arduino IDE



Pluto Python



Visual Micro

5

Рисунок Б.5 – Слайд 5

Алгоритм управління експериментом



6

Рисунок Б.6 – Слайд 6

Особливості програмної реалізації

```
// =====
// # Sensors of 3AxisPortal for use with digitalRead(Sensor)
// =====
#define Crane_Position_Right 0
#define Crane_Position_Left 1
#define Crane_Reference_Position_X 2
#define Crane_Position_Back 3
#define Crane_Position_Front 4
#define Crane_Reference_Position_Y 5
#define Crane_Position_Up 6
#define Crane_Position_Down 7
#define Possibility_Sensor 8
#define User_Buttons 9

// =====
// # Actuators of 3AxisPortal for use with digitalWrite(Actuator, LOW/HIGH)
// =====
#define Drive_Right 0
#define Drive_Left 1
#define Drive_Back 2
#define Drive_Front 3
#define Crane_Up 4
#define Crane_Down 5
#define Magnet 6

// =====
// # Initialize the state of the Mody machine with start state
// =====
void setup()
{
    State = Z0_WaitForFallingEdge;
}

// =====
// # Status of the Mody machine
// =====
typedef enum
{
    Z0_WaitForFallingEdge, // Z0 - wait for 1/8 edge
    Z1_DriveXMotorYMotor, // Z1 - drive to left/down
    Z2_DriveYMotorXStop, // Z2 - drive to up/right
} AutomataStates_t;

AutomataStates_t State;

// =====
// # Update the current state of the Mody machine
// =====
void loop()
{
    switch (State)
    {
        case Z0_WaitForFallingEdge:
        {
            digitalWrite(Drive_Right, LOW);
            digitalWrite(Drive_Left, LOW);
            digitalWrite(Drive_Back, LOW);
            digitalWrite(Drive_Front, LOW);
            digitalWrite(Crane_Up, LOW);
            digitalWrite(Crane_Down, LOW);
            digitalWrite(Magnet, LOW);

            if (!digitalRead(User_Buttons))
            {
                State = Z1_DriveXMotorYMotor;
            }
            break;
        }
        case ...
    }
}
```

Оголошення датчиків та актуаторів

Створення функцій

Створення станів

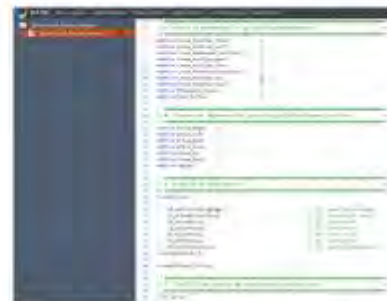
Інструкція перемикача

Рисунок Б.7 – Слайд 7

Результат проведення експерименту



Вікно створення проекту



Завантажений код програми



Результат роботи

Рисунок Б.8 – Слайд 8

Висновки

В результаті виконання дипломної роботи виконано аналіз особливостей реалізації вбудованої електроніки на різноманітних платформах, зокрема, Freeduino, Raspberry Pi, Arduino.

Проведено порівняння функціональних можливостей та характеристик існуючих версій платформи Arduino.

Зроблено огляд побудови мережі GOLDi, особливостей гібридної лабораторії GOLDi її структури, прикладів використання лабораторії для проведення експериментів 3-Axis-Portal, Elevator B, Production Cell.

Розроблено програмний код для мікроконтролерного управління експериментами дистанційної гібридної лабораторії GOLDi на Arduino, зокрема для експерименту 3-Axis-Portal.

В подальшій роботі перелік експериментів може бути розширений.

9

Рисунок Б.9 – Слайд 9