ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

Харківський національний університет
радіоелектроніки

Кафедра ЕОМ

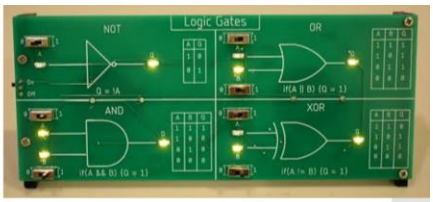
Веб-застосунок для автоматизації проєктування логічних схем

Кваліфікаційна робота
перший (бакалаврський рівень)

Здобувач:
Денис ОЛЬХОВИК
КІУКІ-21-3

Керівник:
ас. Олег ЖУРИЛО

Актуальність теми

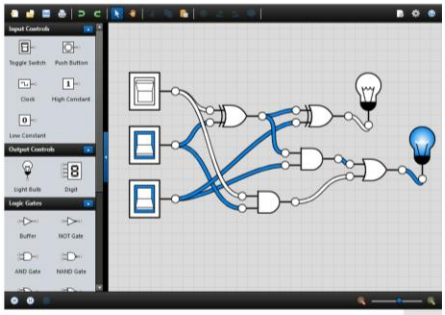


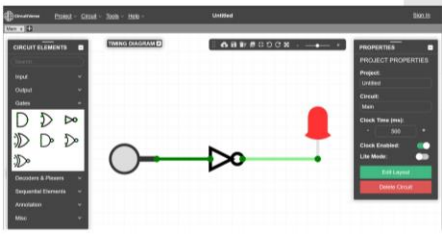
- Складність побудови логічних схем
- Відсутність інструментів для збереження та повторного використання схем
- Обмежені можливості кастомізації
- Відсутність підтримки командної роботи
- Платний або обмежений доступ

Порівняння аналогів

Проблеми існуючих симуляторів:

- складність
- застарілий інтерфейс
- обмежений функціонал
- відсутність збереження результатів роботи



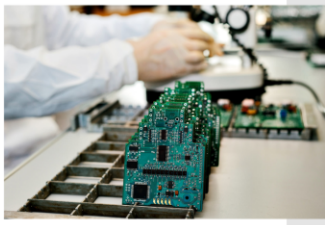


3

Задачі

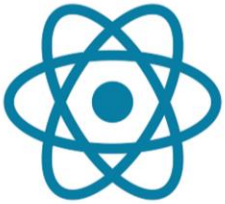
Створити застосунок для побудови й симуляції логічних схем, у якому буде реалізовано:

- підтримку власних елементів
- збереження у хмару
- доступ з браузера
- інтуїтивний інтерфейс



4

Архітектура застосунку

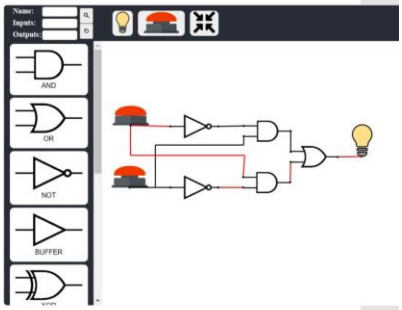


- React + JS + Vite
- Збереження: Google Drive, LocalStorage
- Публікація: GitHub Pages

5

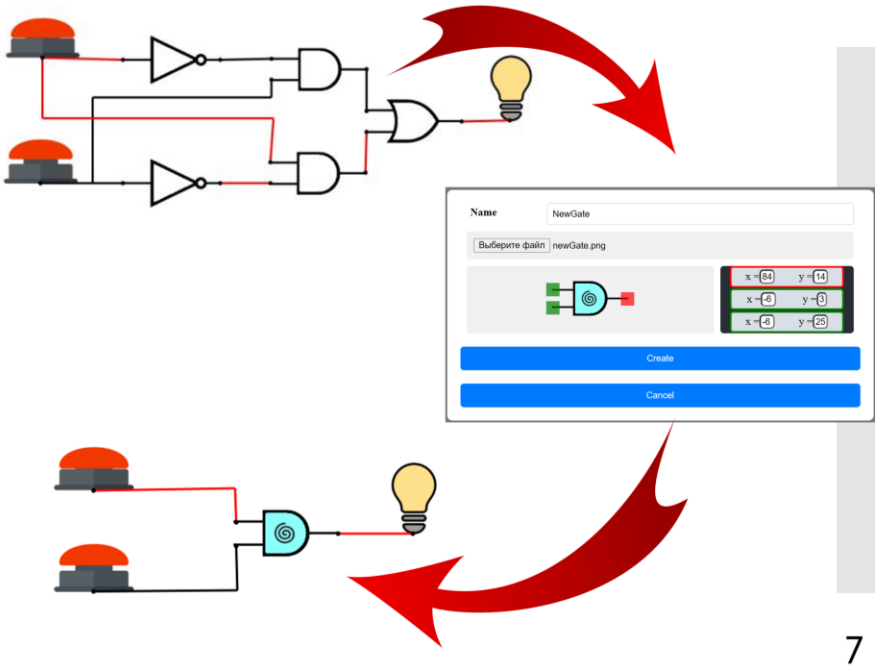
Основні можливості

- Простота використання
- Створення та симуляція схем
- Відкритість для кастомізації
- Експорт у Google Drive



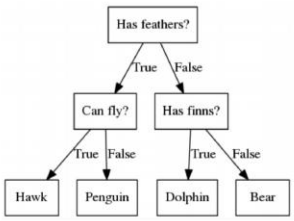
6

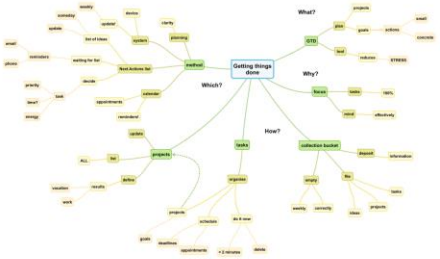
Створення власного вентиля

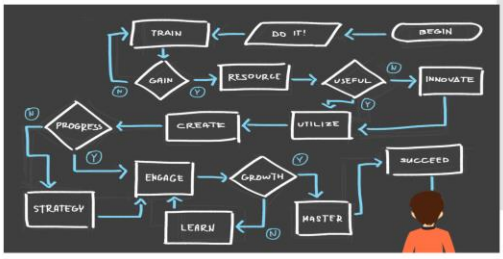


7

Використання застосунку за межами комп'ютерної логіки

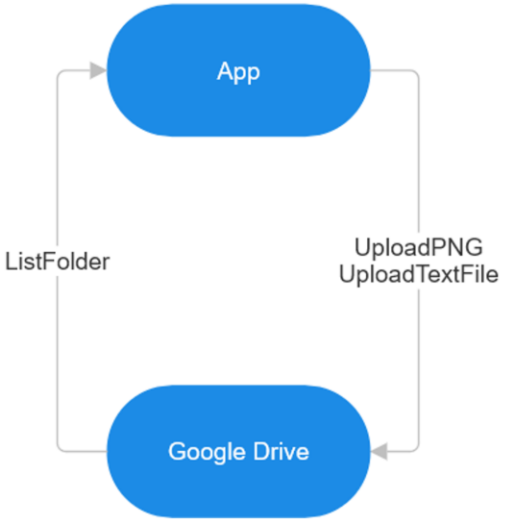






8

Робота з Google Drive



9

Висновки

- Logic Gates Lab — зручний та гнучкий інструмент для проєктування логічних схем
- Може використовуватись як сучасна альтернатива традиційним симуляторам
- Сприяє кращому опануванню цифрової логіки через практичну роботу з логічними схемами

10

ДОДАТОК Б

Вихідний код системи збереження

Б.1 Файл скрипту Google Apps Script ListFolder

```
function doGet(e) {
  try {
    const folderId = e.parameter.folderId; // ID папки переданий
як параметр
    if (!folderId) {
      throw new Error("Missing folderId parameter");
    }

    const folder = DriveApp.getFolderById(folderId);
    const files = folder.getFiles(); // Отримуємо всі файли в
папці
    const contentList = [];

    while (files.hasNext()) {
      const file = files.next();

      // Перевіряємо тип файлу (працюємо лише з Google Docs або
текстовими файлами)
      if (file.getMimeType() === MimeType.GOOGLE_DOCS) {
        const doc = DocumentApp.openById(file.getId());
        const body = doc.getBody().getText();
        contentList.push({
          title: file.getName(),
          content: body,
        });
      } else if (file.getMimeType() === MimeType.PLAIN_TEXT) {
        const textContent = file.getBlob().getDataAsString();
        contentList.push({
          title: file.getName(),
          content: textContent,
        });
      }
    }

    // Повертаємо JSON з вмістом файлів
    return
ContentService.createTextOutput(JSON.stringify(contentList))
      .setMimeType(ContentService.MimeType.JSON);

  } catch (error) {
    return ContentService.createTextOutput(JSON.stringify({
      success: false,
```

```
      error: error.message,
    })).setMimeType(ContentService.MimeType.JSON);
  }
}
```

Б.2 Файл скрипту Google Apps Script UploadTextFile

```
function doPost(e) {
  try {
    const folderId = e.parameter.folderId; // ID папки на Google
Drive
    const fileName = e.parameter.fileName || 'file.txt'; // Ім'я
текстового файлу
    const content = e.parameter.content || '"name": "AND"'; //
Текст для запису в файл

    if (!folderId || !content) {
      return ContentService.createTextOutput("Missing folderId
or content")
        .setMimeType(ContentService.MimeType.TEXT);
    }

    // Створюємо текстовий файл з вказаним контентом
    const folder = DriveApp.getFolderById(folderId);
    const file = folder.createFile(fileName, content,
MimeType.PLAIN_TEXT);

    return ContentService.createTextOutput(JSON.stringify({
      success: true,
      fileUrl: file.getUrl(),
      fileId: file.getId(),
    })).setMimeType(ContentService.MimeType.JSON);
  } catch (error) {
    return ContentService.createTextOutput(JSON.stringify({
      success: false,
      error: error.message,
    })).setMimeType(ContentService.MimeType.JSON);
  }
}
```

Б.3 Файл скрипту Google Apps Script UploadPNG

```
function doPost(e) {
  try {
    const folderId = e.parameter.folderId; // ID папки на Google
Drive
    const contentType = e.parameter.contentType || 'image/png';
// Тип контенту (за замовчуванням PNG)
```

```
    const fileName = e.parameter.fileName ||
'uploaded_image.png'; // Ім'я файлу
    const base64Data = e.parameter.fileData; // Base64-дані
файлу

    if (!folderId || !base64Data) {
      return ContentService.createTextOutput("Missing folderId
or fileData")
        .setMimeType(ContentService.MimeType.TEXT);
    }

    // Декодуємо Base64 дані
    const decodedBlob =
Utilities.newBlob(Utilities.base64Decode(base64Data),
contentType, fileName);

    // Завантажуємо файл до вказаної папки
    const folder = DriveApp.getFolderById(folderId);
    const file = folder.createFile(decodedBlob);

    return ContentService.createTextOutput(JSON.stringify({
      success: true,
      fileUrl: file.getUrl(),
      fileId: file.getId(),
    })).setMimeType(ContentService.MimeType.JSON);
  } catch (error) {
    return ContentService.createTextOutput(JSON.stringify({
      success: false,
      error: error.message,
    })).setMimeType(ContentService.MimeType.JSON);
  }
}
```

Б.4 GoogleDriveSaving.jsx

```
export async function saveTxtToGoogleDrive(element) {
  const uploadUrl =

"https://script.google.com/macros/s/AKfycbzxAmUKqd_Jdz3rba2Yy_iX
7f3H00rjv5UgOb7glvUkCgYBX2nouG9TH8kGgObiM7pg/exec";
  const fileName = element.name;
  const content = JSON.stringify(element, null, 2);
  const match =
localStorage.getItem("savedURL").match(/\/folders\/([^/?]+)/);
  const folderId = match ? match[1] : null;

  if (folderId !== null) {
    const response = await fetch(uploadUrl, {
      method: "POST",
      headers: {
        "Content-Type": "application/x-www-form-urlencoded",
```

```
      },
      body: new URLSearchParams({
        folderId: folderId,
        fileName: fileName,
        content: content,
      }),
    });

    const result = await response.json();
    if (!result.success) {
      alert(`Upload failed: ${result.error}`);
    }
  }
}

export async function saveImgToGoogleDrive(name, gateName) {
  const uploadUrl =

"https://script.google.com/macros/s/AKfycbzoi8w9JJHmQqDkuO4oIDYD
Pk9Wv2POMtkr6ZlJ5ylonssx5RPMwBG0mXHTlWkOUw49cw/exec";
  const match =
localStorage.getItem("savedURL").match(/\/folders\/([^/?]+)/);
  const folderId = match ? match[1] : null;

  const isUrl = (str) => {
    try {
      const url = new URL(str);
      return url.protocol === "http:" || url.protocol ===
"https:";
    } catch (_) {
      return false;
    }
  };

  if (folderId !== null) {
    try {
      if (isUrl(name)) {
        return name;
      }

      // Завантажуємо зображення як Blob
      const response = await fetch(name);
      if (!response.ok) {
        alert(`Failed to fetch image: ${response.statusText}`);
        return null;
      }
      const blob = await response.blob();

      // Читаємо Blob як Base64
      const base64Data = await new Promise((resolve, reject) =>
{
        const reader = new FileReader();
        reader.onload = () =>
```

```
resolve(reader.result.split(",")[1]); // Видалити префікс
        reader.onerror = () => reject(reader.error);
        reader.readAsDataURL(blob);
      });

      // Відправляємо запит на завантаження
      const uploadResponse = await fetch(uploadUrl, {
        method: "POST",
        headers: {
          "Content-Type": "application/x-www-form-urlencoded",
        },
        body: new URLSearchParams({
          folderId: folderId,
          fileName: gateName,
          fileData: base64Data,
        }),
      });

      const result = await uploadResponse.json();
      if (result.success) {
        const match = result.fileUrl.match(/\/d\/([^/]+)/);
        if (match && match[1]) {
          const fileId = match[1];
          return `https://lh3.google.com/u/0/d/${fileId}`;
        } else {
          alert("Invalid Google Drive link format.");
        }
      } else {
        alert(`Upload failed: ${result.error}`);
        return null;
      }
    } catch (error) {
      console.error("Error fetching or uploading static image:",
error);
      alert("Failed to fetch or upload static image. See console
for details.");
      return null;
    }
  }
}

export async function loadFromGoogleDrive() {
  const webAppUrl =

"https://script.google.com/macros/s/AKfycbxTOOhKhaqCSw7hJJKDIcKl
NCt6TMToAjPqnEItno4JHv2q0zkY7iaavUFWvVom6Lcukw/exec";
  const savedURL = localStorage.getItem("savedURL");
  if (!savedURL) {
    return 0;
  }
  const match = savedURL.match(/\/folders\/([^/?]+)/);
  const folderId = match ? match[1] : null;
```

```
  if (folderId === null) {
    return 0;
  } else {
    try {
      const response = await
fetch(`${webAppUrl}?folderId=${folderId}`);
      if (!response.ok) {
        throw new Error(`HTTP error! Status:
${response.status}`);
      }
      const result = await response.json();
      const parsedArray = result.map((item) => ({
        title: item.title,
        content: JSON.parse(item.content),
      }));
      return parsedArray;
    } catch (error) {
      console.error("Fetch error:", error);
      console.log(`Error: ${error.message}`);
    }
  }
}
```