DOI: 10.22146/jnteti.v14i1.7058

# Mesin Pengisian Cairan Otomatis Menggunakan Arduino dan LabVIEW

**Syafriyadi Nor[1], Zaiyan Ahyadi[1]**

[1] Jurusan Teknik Elektro, Program Studi Elektronika, Politeknik Negeri Banjarmasin, Banjarmasin, Kalimantan Selatan 70123, Indonesia

[Diserahkan: 5 Februari 2024, Direvisi: 3 Mei 2024, Diterima: 24 Februari 2025]
Penulis Korespondensi: Zaiyan Ahyadi (email: z.ahyadi@poliban.ac.id)

**INTISARI** — Mesin pengisian cairan otomatis merupakan elemen krusial dalam meningkatkan efisiensi dan produktivitas industri modern, khususnya pada sektor manufaktur dan pengemasan. Namun, implementasi teknologi ini sering kali menghadapi kendala seperti biaya yang tinggi, kebutuhan akan perangkat yang kompleks, serta keterbatasan pemahaman teknis. Penelitian ini bertujuan untuk merancang sistem praktikum edukatif yang sederhana, terjangkau, dan mudah diimplementasikan guna membantu mahasiswa memahami konsep dasar sekaligus aplikasi nyata mesin pengisian cairan otomatis. Sistem ini mengintegrasikan perangkat lunak LabVIEW untuk pemrosesan visual dan mikrokontroler Arduino Nano dengan protokol komunikasi Modbus RTU, yang menyimulasikan standar komunikasi industri. Program LabVIEW mengendalikan proses pergerakan *conveyor belt*, pengisian (*filling*), dan penutupan (*capping*) menggunakan *ladder logic*, serta mencatat jumlah botol yang diisi. Mikrokontroler Arduino berfungsi untuk mengelola kendali tombol *on-off conveyor belt* melalui LabVIEW serta *keypad* untuk menentukan takaran volume dan jumlah botol. Komunikasi antara LabVIEW dan Arduino dilakukan secara serial menggunakan protokol Modbus RTU, memberikan pengalaman langsung kepada mahasiswa dalam mengonfigurasi komunikasi industri. Uji eksperimental dilakukan dengan berbagai skenario operasional untuk mengevaluasi kinerja sistem. Hasil pengujian menunjukkan bahwa sistem mampu mengisi botol dengan akurat pada rentang volume cairan 250–1.000 ml, dengan kecepatan pengisian hingga 10 ml/s dan kapasitas maksimal lima botol per siklus. Sistem berjalan stabil tanpa gangguan selama pengujian. Penelitian ini berkontribusi pada pengembangan praktikum instrumentasi sistem kendali; menghadirkan solusi pembelajaran yang efektif, interaktif, dan terjangkau. Penggunaan Modbus RTU terbukti menjadi pendekatan komunikasi yang andal untuk mendukung implementasi mesin pengisian cairan otomatis sekaligus meningkatkan pemahaman mahasiswa tentang aplikasi teknologi industri.

**KATA KUNCI** — Instrumentasi, Arduino, LabVIEW, *Filling Machine*, *Capping*, Modbus.

## I. PENDAHULUAN

Mesin pengisian cairan otomatis banyak digunakan di berbagai industri, seperti industri makanan, farmasi, kimia, dan kosmetik, untuk mengisi cairan dengan volume dan jumlah yang berbeda ke dalam wadah. Ketepatan dalam mengisi cairan merupakan kunci untuk mencegah kerugian finansial, sedangkan penggunaan otomatisasi dapat meningkatkan efisiensi dan kualitas dalam sektor industri [1]. Pengisian cairan secara manual membutuhkan banyak tenaga dan waktu, sehingga mengurangi efisiensi dan produktivitas [2]. Selain itu, otomatisasi bertujuan untuk meningkatkan mutu produk, meningkatkan tingkat produksi, mengurangi penggunaan tenaga kerja, menurunkan biaya tenaga kerja, mengurangi pemborosan bahan baku, serta meminimalkan waktu tidak produktif [3].

Mesin pengisian cairan otomatis berfungsi untuk mengisi cairan ke dalam botol, gelas, atau kaleng. Selain itu, mesin ini juga dapat mengemas beragam produk cair, seperti air mineral, minyak, minuman ringan, dan anggur, ke dalam kemasan botol, gelas, atau kaleng [4]. Mesin ini dapat mengisi cairan ke dalam botol, gelas, atau kaleng secara otomatis sesuai dengan takaran dan jumlah yang diatur sebelumnya. Perkembangan industri 4.0 membuat mesin ini tidak membutuhkan banyak tenaga manusia dalam proses kendali [5], sehingga dapat mengatasi masalah kekurangan staf yang dihadapi [6].

Proses pengisian (*filling*) dan penutupan (*capping*) dalam produksi produk cair sering menghadapi tantangan, seperti masalah kendali ketika terjadi kesalahan atau gangguan pada sistem [7]. Selain itu, implementasi sistem mesin pengisian cairan otomatis memerlukan konfigurasi yang kompleks pada perangkat keras dan lunak untuk mengubah takaran volume dan jumlah botol. Dengan menggunakan kombinasi teknologi perangkat lunak dan keras, seperti LabVIEW dan mikrokontroler Arduino, implementasi dapat memberikan gambaran dalam proses pengisian cairan ke dalam kemasan. Selain itu, perangkat yang digunakan juga lebih terjangkau. LabVIEW adalah bahasa pemrograman grafis yang pertama kali dirilis pada tahun 1986 oleh National Instruments [8]. Pemrograman LabVIEW tidak ditulis dalam bentuk teks, tetapi menggunakan *dataflow* atau direpresentasikan secara grafis menggunakan diagram blok. LabVIEW memiliki banyak fitur dan keunggulan, seperti mudah digunakan, antarmuka yang intuitif, dan dapat memproses data secara *real-time*. Sementara itu, mikrokontroler Arduino adalah sebuah platform gratis yang yang dapat digunakan untuk mengendalikan berbagai jenis mesin dan sistem [9]. Selain itu, mikrokontroler Arduino memiliki harga yang terjangkau.

Penelitian mengenai mesin pengisian cairan otomatis dengan integrasi teknologi LabVIEW dan mikrokontroler Arduino masih terbatas. Penerapan *programmable logic controller* (PLC) sebagai proses kendali telah dilakukan sebelumnya [10]–[14]. Desain dan implementasi sistem pengisian cairan otomatis menggunakan mikrokontroler Arduino dan LabVIEW juga sudah pernah diteliti [15]–[18]. Pada penelitian lain, sebuah sistem dibuat untuk melakukan pemantauan kekeruhan, suhu, dan pH air dengan antarmuka LabVIEW dan Arduino [19]. Selain menggunakan Arduino, integrasi dengan LabVIEW juga dapat dilakukan dengan PLC.

Sebuah penelitian membuat sistem *windscreen washer filling machine*, yang dirancang menggunakan PLC dan LabVIEW [20]. Antarmuka komunikasi antara PLC dan LabVIEW menggunakan *open process control* (OPC). Pemantauan sistem dilakukan dengan *supervisory control and data acquisition* (SCADA), yang dibuat menggunakan modul *datalogging and supervisory control* (DSC) dari LabVIEW. Namun, penelitian tersebut belum membahas mengenai penentuan data masukan untuk takaran volume dan jumlah botol. Integrasi antara LabVIEW dan Arduino dilakukan menggunakan protokol modbus *remote terminal unit* (RTU) [20]. Penggunaan protokol modbus sendiri telah menjadi standar industri. Oleh karena itu, banyak perangkat keras dan lunak yang mendukung protokol ini. Adanya standar ini memudahkan integrasi dan kompatibilitas antarperangkat dari berbagai produsen, tidak terkecuali pada Arduino dan LabVIEW, yang telah mendukung protokol tersebut. Maka, penelitian ini bertujuan merancang *industrial plant system* pada mesin pengisian cairan otomatis dengan masukan takaran volume dan jumlah botol melalui *keypad* menggunakan LabVIEW dan mikrokontroler Arduino dengan meniru proses sistem nyata yang ada pada industri.

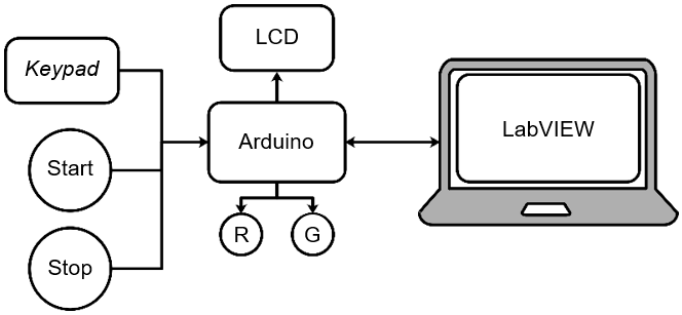
Hal yang baru dalam penelitian ini adalah integrasi antara LabVIEW dan Arduino melalui protokol modbus RTU. Mikrokontroler Arduino berperan sebagai pengendali fisik yang mengendalikan *plant* pada LabVIEW. Di penelitian serupa, umumnya digunakan perangkat keras PLC. Kombinasi ini menjadi solusi atas keterbatasan akses dan ketersediaan peralatan yang relevan dengan industri. Penelitian ini sangat penting untuk membantu mahasiswa dalam praktikum instrumen sistem kendali dan memberikan gambaran tentang proses pada sistem mesin pengisian cairan otomatis.

## II. PERANCANGAN SISTEM

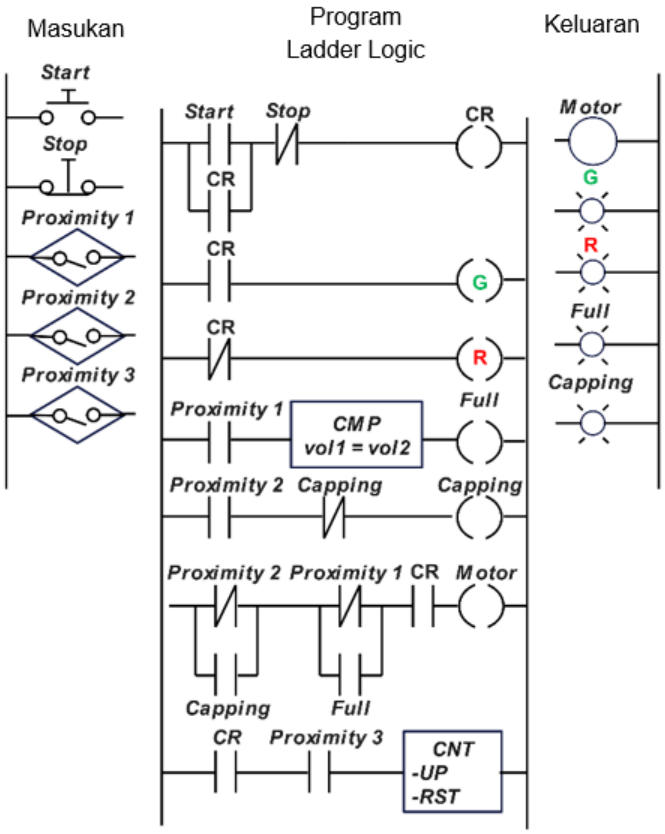
Dalam penenelitian ini, dirancang sebuah *plant* simulasi sistem mesin pengisian cairan otomatis pada aplikasi LabVIEW 2017. Gambar 1 menunjukkan diagram blok sistem pengisian cairan otomatis, yang terdiri atas mikrokontroler Arduino, *push button*, *keypad*, LCD, LED, dan kendali LabVIEW. Arduino digunakan untuk memproses data masukan *keypad* dan *push button*, sedangkan LCD menampilkan tampilan menu dan data masukan dari *keypad* dan LED digunakan sebagai indikator perintah dari *push button*. Kendali LabVIEW akan menerima sinyal kendali dan menjalankan proses pengisian cairan.

Sistem ini menggunakan modul mikrokontroler Arduino Nano ATMega 328. Arduino Nano memiliki 22 *pin* digital untuk I/O, 8 *pin* analog, dan beberapa protokol komunikasi serial seperti UART, SPI dan I2 [21]. Pada Arduino Nano dipasang dua *push button* untuk *start* dan *stop* serta dua LED sebagai indikator. LED berwarna hijau untuk menunjukkan *conveyor belt* berjalan dan LED warna merah untuk menunjukkan *conveyor belt* berhenti. Untuk memasukkan jumlah takaran dan jumlah botol, digunakan *keypad* 3 × 4 beserta LCD 16 × 2 dengan modul i2c. Untuk menghubungkan masukan *push button* dan *keypad* ke LabVIEW, protokol serial UART Arduino dihubungkan dengan protokol serial LabVIEW menggunakan National Instrument Virtual Instrument Software Architecture (NI-VISA). NI-VISA adalah sebuah *application programming interface* (API) yang menyediakan antarmuka pemrograman komunikasi serial pada aplikasi National Instrument (NI) [22].

Pada kendali LabVIEW, perancangan terdiri atas *conveyor belt*, sensor *proximity* untuk mendeteksi keberadaan botol saat




**Gambar 1.** Diagram blok sistem mesin pengisian cairan otomatis.

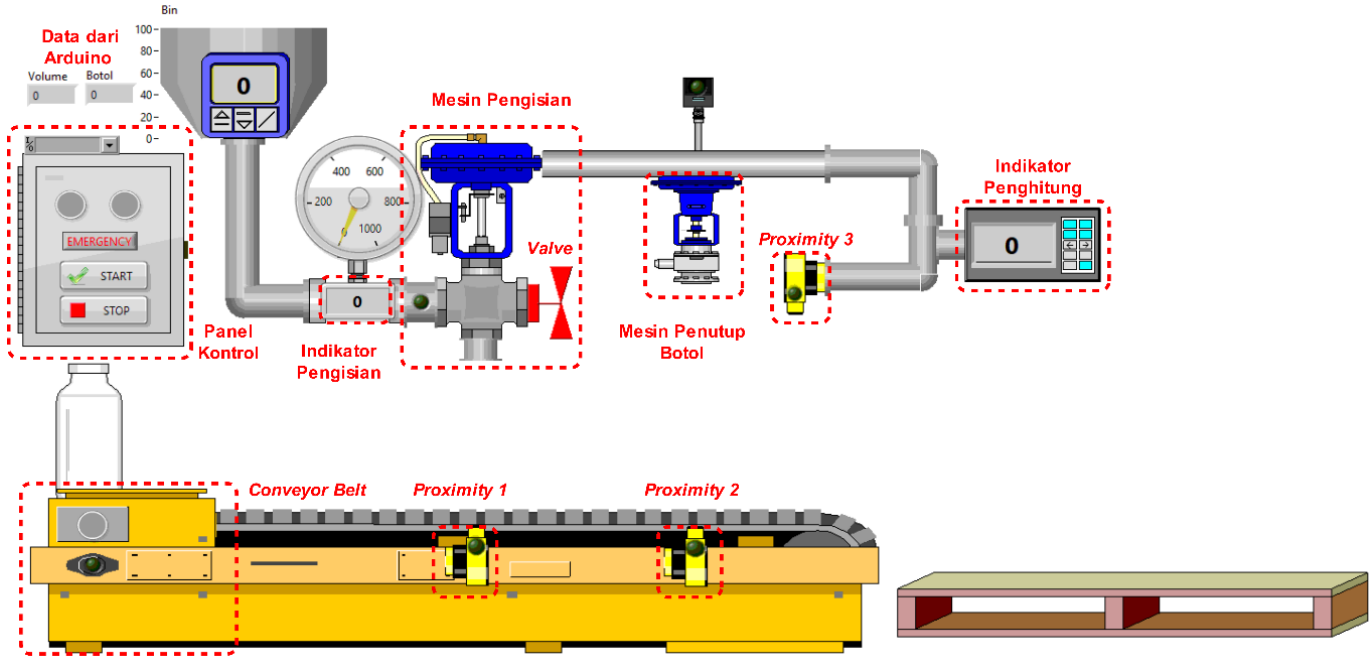



**Gambar 2.** *Ladder logic* pada sistem mesin pengisian cairan otomatis.

*conveyor belt* berjalan, mesin pengisian cairan (*FILLING*), dan mesin untuk menutup botol (*CAPPING*). Mesin tersebut dibuat menggunakan modul DSC, yaitu modul perangkat lunak *add-on* untuk LabVIEW untuk mengembangkan *human machine interface* (HMI).

Meskipun LabVIEW adalah pemrograman berbasis grafis atau diagram blok, dalam penelitian ini proses kendali simulasi sistem mesin pengisian cairan otomatis dilakukan menggunakan *ladder logic*. *Ladder logic* dipilih karena digunakan untuk membangun perangkat lunak dari PLC dan implementasinya telah banyak digunakan dalam aplikasi kendali industri. Dalam hal ini, *ladder logic* digunakan untuk membangun logika yang runtun pada proses *conveyor belt* sampai dengan pengisian, penutupan botol, dan penghitungan jumlah botol. Algoritma kendali dapat diprogram dan disusun berdasarkan *ladder logic*, sedangkan fungsi *counter* dan *timer* disesuaikan dengan aturan program pada LabVIEW.

Gambar 2 menunjukkan *ladder logic* yang digunakan untuk kendali sistem pada diagram blok LabVIEW. Cara kerja *plant* ini adalah sebagai berikut. Saat tombol *start* ditekan, *control relay* (CR) akan aktif dan menyalakan lampu hijau (*green*, G)




**Gambar 3.** Komponen sistem mesin pengisian cairan otomatis pada *Front Panel* LabVIEW.

sebagai indikator bahwa *conveyor belt* aktif untuk memindahkan botol masuk dan keluar dari stasiun pengisian. Saat sensor *proximity* 1 mendeteksi keberadaan botol, *conveyor belt* berhenti mengisi cairan. Takaran yang akan dimasukkan ke dalam botol ditentukan oleh masukan *keypad* yang dikirimkan Arduino melalui komunikasi serial.

Setelah terisi sesuai takaran, motor kembali menjalankan *conveyor belt*. Ketika sensor *proximity* 2 mendeteksi keberadaan botol, *conveyor belt* berhenti kembali dan melakukan proses penutupan pada botol. Setelah proses penutupan selesai, *conveyor belt* kembali aktif untuk memindahkan botol. Saat sensor *proximity* 3 mendeteksi keberadaan botol, botol akan dihitung.

Total *counter* akan direset kembali ke 0 saat mencapai *set point* yang ditentukan. Proses berhenti jika tombol *Stop* ditekan; lampu G akan padam dan lampu merah (*red*, R) akan menyala. Seluruh komponen pada sistem mesin pengisian cairan otomatis pada *Front Panel* LabVIEW ditunjukkan dalam Gambar 3.

Panel kendali digunakan untuk mengendalikan komponen yang ada pada *front panel* LabVIEW. Indikator LED yang terdapat pada panel akan menyala bersamaan dengan LED yang terhubung ke Arduino. *Conveyor belt* digunakan untuk memindahkan botol ke tempat pengisian hingga proses penutupan selesai. Sensor *proximity* berfungsi untuk mendeteksi keberadaan botol saat *conveyor belt* berjalan. *Display* pengisian memberikan informasi tentang laju aliran fluida yang sedang mengalir dalam pipa atau saluran tertentu ke dalam botol.

Data dari Arduino dikirim ke LabVIEW menggunakan komunikasi serial. Mesin pengisian bertugas mengisi cairan ke dalam botol saat sensor *proximity* 1 mendeteksi keberadaan botol, sedangkan mesin penutupan digunakan untuk menutup botol saat botol terdeteksi oleh sensor *proximity* 2. *Counter display* memberikan informasi jumlah botol yang terdeteksi oleh sensor *proximity* 3. Katup (*valve*) digunakan sebagai indikator untuk status menutup atau membuka katup saat proses pengisian berlangsung.

Tahap selanjutnya adalah membuat program di Arduino IDE untuk mengambil data dari pembacaan *push button* dan *keypad*, kemudian menampilkan data tersebut ke LCD. Kemudian dilakukan pembuatan menu pilihan pada LCD, yang terdiri atas menu *VIEW* dan *SET*.

Menu *SET* adalah fungsi yang dipanggil untuk memasukkan takaran volume pada botol [0 – 1.000] ml. Proses memasukkan takaran volume adalah dengan menekan angka pada *keypad* [‘0’ – ‘9’]. Karena keluaran fungsi *keypad* mengembalikan nilai dengan tipe data *char*, perlu dilakukan konversi ke integer. Konversi dilakukan dengan (1).

$$x = x \times 10 + (key - '0'). \quad (1)$$

Variabel $x$ adalah tipe data *integer*, angka 10 adalah konstanta yang dikalikan dengan $x$ untuk menambahkan 1 setiap kali tombol pada *keypad* ditekan, dan *key* adalah angka pada keypad [‘0’ – ‘9’] yang dikurangi dengan karakter ‘0’ untuk mendapatkan hasil bilangan desimal dari kode ASCII (*American Standard Code for Information Interchange*). Karakter angka [‘0’ – ‘9’] diubah menjadi bilangan desimal [48–57].

Setelah nilai takaran dimasukkan, jika simbol ‘*’ pada *keypad* ditekan, nilai takaran akan disimpan ke dalam sebuah variabel. Kemudian, proses selanjutnya adalah memasukkan jumlah botol yang akan diisi. Jika masukan jumlah botol telah dimasukkan, simbol ‘*’ pada *keypad* ditekan kembali, sehingga variabel yang menampung takaran dan jumlah botol akan disimpan. Kemudian, tampilan akan kembali ke pilihan *VIEW*. Di dalam pilihan *VIEW*, variabel takaran dan jumlah botol akan tertampil pada LCD 16 × 2. Saat *push button START* (1) atau *STOP* (0) ditekan, data yang terkandung dalam variabel tersebut dikirim ke LabVIEW meggunakan protokol Modbus RTU.

Proses ini memungkinkan LabVIEW untuk menerima dan memproses data takaran dan jumlah botol dari sistem, memberikan fleksibilitas dan kendali yang lebih lanjut dalam pemantauan dan pengaturan parameter secara *real-time*. Dengan integrasi Modbus RTU, informasi yang dikumpulkan

dapat dengan efisien diakses dan dimanfaatkan dalam aplikasi LabVIEW. Untuk berkomunikasi menggunakan protokol modbus pada Arduino, diperlukan *library* yang mendukung protokol modbus tersebut. Dalam penelitian ini, digunakan *library* ModbusRTUSlave [23].

Gambar 4 menunjukkan pesan Modbus RTU, yang terdiri atas alamat perangkat *Slave ID*, *function code*, *data*, dan *cyclic reduncany check* (CRC). Jika bagian *Slave* ID dan CRC dari suatu protokol Modbus dihilangkan, akan diperoleh unit data protokol (*protocol data unit*, PDU).

*Slave* ID (*Identification*) adalah bagian dari protokol Modbus yang menunjukkan alamat perangkat Modbus *slave* yang akan berkomunikasi dengan perangkat Modbus *master*. Setiap perangkat Modbus *slave* diidentifikasi oleh alamat unik yang dikenal sebagai *Slave* ID. selain itu, *Slave* ID juga digunakan oleh perangkat Modbus *master* untuk menentukan perangkat tujuan dari permintaan atau instruksi yang dikirimkan.

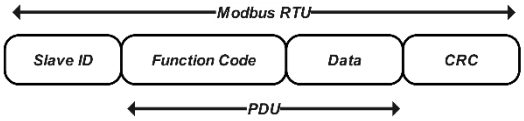
*Function code* adalah nilai numerik yang mengidentifikasi jenis operasi yang diminta oleh perangkat Modbus *master* atau yang dilakukan oleh perangkat Modbus *slave*. Misalnya, *Function code* 03 digunakan untuk membaca *holding register*, sedangkan *function code* 06 digunakan untuk menulis ke *holding register*. *Function code* memberikan petunjuk kepada perangkat Modbus tentang tindakan yang harus dilakukan. Setiap *function code* memiliki makna dan fungsi tertentu dalam konteks operasi Modbus. Tabel I menyajikan *function code* umum pada protokol modbus.

Data dalam konteks Modbus merujuk pada informasi yang dikirimkan atau diterima melalui protokol Modbus. Data ini dapat mencakup nilai-nilai dari register atau informasi lain yang dikirimkan antara perangkat Modbus *master* dan *slave*. Data tersebut menyimpan nilai-nilai yang diperlukan untuk menjalankan operasi sesuai dengan *function code* yang dikirimkan. Sebagai contoh, jika *function code* adalah 03 (*holding register*), data mungkin berisi informasi tentang alamat register yang akan dibaca dan jumlah register yang akan diambil.

CRC adalah metode pengecekan kesalahan yang digunakan untuk memastikan integritas data dalam komunikasi Modbus. Proses ini melibatkan penghitungan nilai CRC dari paket data dan penyertaan nilai CRC tersebut di dalam paket.

Karena Arduino berfungsi sebagai *slave*, Arduino akan merespons pesan yang diminta oleh LabVIEW yang berfungsi sebagai *master*. Data pembacaan *coil*, takaran volume, dan jumlah botol disimpan ke dalam variabel. Variabel CR adalah tipe data Boolean yang merepresentasikan dua kondisi, yaitu nilai 0, yang menandakan perintah *STOP*, dan nilai 1, yang menunjukkan perintah *START*. Untuk mengakses dan membaca data ini, diterapkan *function code* 01. Sementara itu, variabel volume dan jumlah botol memiliki tipe data integer. Variabel volume memiliki rentang nilai antara 0 hingga 1.000 ml, sedangkan variabel botol memiliki rentang nilai antara 0 hingga 10. Meskipun rentang nilai ini dapat ditetapkan secara fleksibel sesuai kebutuhan, dalam konteks penelitian ini kedua variabel tersebut dibatasi agar nilainya berada dalam rentang nilai yang telah dijelaskan.

Untuk menyimpan data dari variabel volume dan botol, digunakan struktur data berupa larik (*array*) yang berfungsi sebagai register. Dengan implementasi ini, data yang terkandung dalam larik dapat diakses dan dibaca melalui penggunaan *function code* 03.




**Gambar 4.** Pesan modbus RTU.

TABEL I
*FUNCTION CODE* UMUM PADA PROTOKOL MODBUS

| *Function Code* | Deskripsi | Fungsi |
|---|---|---|
| 01 | *Read Coils* | Membaca nilai dari beberapa *coil*. |
| 02 | *Read Discrete Inputs* | Membaca nilai dari beberapa masukan diskret. |
| 03 | *Read Holding Registers* | Membaca nilai dari beberapa *holding register*. |
| 04 | *Read Input Registers* | Membaca nilai dari beberapa masukan register. |
| 05 | *Write Single Coil* | Menulis nilai ke satu *coil*. |
| 06 | *Write Single Register* | Menulis nilai ke satu register. |
| 15 | *Write Multiple Coils* | Menulis nilai ke beberapa *coil*. |
| 16 | *Write Multiple Registers* | Menulis nilai ke beberapa register. |

Selanjutnya, LabVIEW akan mengirim permintaan ke *slave* (Arduino) untuk mengakses dan membaca data yang tersimpan di dalam register. Untuk membaca data *coil*, blok yang digunakan adalah *read coil*, dengan alamat awal 0 (00001), sedangkan untuk membaca data takaran volume dan jumlah botol, digunakan blok *Read Holding Registers*, dengan alamat awal 0 (40001 untuk takaran volume, 40002 untuk jumlah botol).

Pada tahap awal pemrograman, *library* pendukung dan variabel untuk menyimpan nilai takaran volume dan jumlah botol dideklarasikan. Ketika pengguna menekan simbol '*' pada *keypad*, pilihan akan beralih ke pilihan *SET* untuk mengatur nilai takaran volume dan jumlah botol. Setelah itu, pilihan akan kembali ke *VIEW*. Dari pilihan *VIEW*, pengguna dapat menekan tombol *START* atau *STOP* untuk mengirimkan data ke LabVIEW.

Alur pemrograman mikrokontroler Arduino dimulai dengan deklarasi *library* pendukung serta variabel yang digunakan untuk menyimpan nilai takaran volume dan jumlah botol. Pada tahap awal, sistem berada dalam mode *VIEW* sebagai mode *default*. Dalam mode ini, pengguna dapat memantau nilai takaran volume dan jumlah botol yang telah diatur sebelumnya.

Jika pengguna menekan simbol '*' pada *keypad*, sistem akan beralih ke mode SET. Pada mode ini, pengguna dapat memasukkan nilai baru untuk takaran volume cairan (dalam ml) dan jumlah botol yang diinginkan. Setelah nilai-nilai ini disimpan, sistem secara otomatis kembali ke mode *VIEW*.

Dari mode *VIEW*, pengguna memiliki opsi untuk memulai atau menghentikan pengoperasian sistem. Menekan tombol *START* pada *push button* akan mengirimkan nilai *coil* 1 (*ON*) ke perangkat lunak LabVIEW melalui protokol komunikasi Modbus RTU untuk memulai proses pengisian cairan. Sebaliknya, menekan tombol *STOP* akan mengirimkan nilai *coil* 0 (*OFF*) untuk menghentikan proses yang sedang berlangsung. Alur ini memastikan bahwa pengguna memiliki

kendali penuh atas pengaturan dan pengoperasian sistem dengan antarmuka yang sederhana dan terstruktur.

## III. HASIL PERANCANGAN DAN SIMULASI

Simulasi mesin pengisian cairan otomatis telah dirancang pada LabVIEW, dengan menggunakan perangkat keras seperti mikrokontroler Arduino, *push button*, LED, LCD 16 × 2, dan *keypad* 3 × 4. Tahap pertama adalah merangkai komponen dan perangkat keras yang telah disebutkan. Kemudian, program diunggah ke mikrokontroler Arduino ATMega 328P. Setelah program diunggah, tampilan awal akan menunjukkan menu *VIEW*, yaitu nilai takaran volume dan jumlah botol. Saat simbol '*' pada *keypad* ditekan, menu akan berpindah ke menu *SET*. Di dalam menu *SET*, pengguna dapat memasukkan nilai takaran atau volume dengan rentang [0 – 1.000] ml. Setelah itu, simbol '*' ditekan untuk menyimpan volume, kemudian menampilkan jumlah botol atau total botol. Selanjutnya, simbol '*' ditekan kembali untuk menyimpan data masukan dan tampilan akan kembali ke menu *VIEW*.

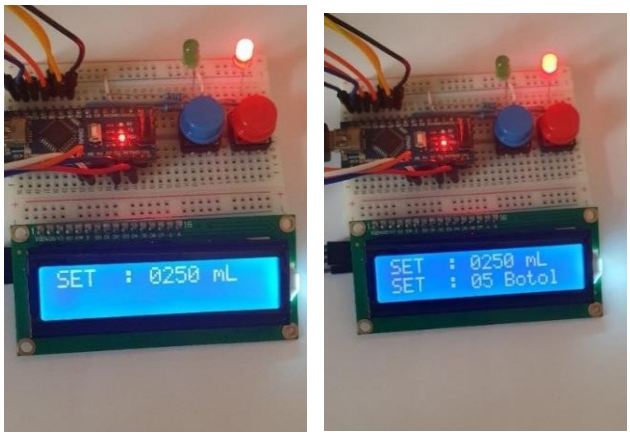
Proses memasukkan nilai takaran dan jumlah botol menggunakan *keypad* ditunjukkan pada Gambar 5. Terdapat tiga variabel yang dikirim melalui serial Arduino: variabel CR berisi *state* 0 atau 1 dari *push button*, variabel volume [0 – 1.000] ml dan variabel jumlah botol [0 – 10] botol. Namun, dalam penelitian ini jumlah botol dibatasi hanya lima botol.

Proses penerimaan data oleh LabVIEW dilakukan melalui Modbus RTU dan Virtual Instrument Software Architecture (VISA) pada LabVIEW. Program mesin pengisian cairan otomatis menggunakan diagram blok pada LabVIEW. Setiap blok dihubungkan membentuk *dataflow* sesuai dengan *ladder logic*. Gambar 6 hanya menampilkan tangkapan layar sebagian dari diagram blok secara keseluruhan.

*Modbus Create Master Instance* adalah salah satu fungsi atau blok dalam LabVIEW yang digunakan untuk membuat instansiasi Modbus *master*. Instansiasi ini dapat menjadi Modbus *master* serial atau TCP. Parameter yang digunakan meliputi jenis koneksi serial (seperti RTU atau ASCII), alamat perangkat Modbus *slave*, *resource name* VISA untuk komunikasi serial, kecepatan transfer data (*baud rate*), dan metode pemeriksaan kesalahan (*parity*) untuk memastikan integritas data.

*Read Coils* adalah blok yang digunakan untuk membaca status dari sejumlah *coil* (relai atau sakelar) di perangkat Modbus *slave*. Dalam penelitian ini, blok ini digunakan untuk membaca nilai dari variabel CR. Modbus *master* dapat mendapatkan informasi tentang keadaan relai atau sakelar pada perangkat Modbus *slave* dengan mengatur parameter seperti alamat awal dari *coil* yang akan dibaca (*starting address*), jumlah *coil* yang akan dibaca (*number of coils*), dan nilai-nilai yang merepresentasikan status *on/off* dari *coil* yang telah dibaca (*coil values*).

*Index Array Function* digunakan untuk membaca nilai berdasarkan indeks larik. *Read Holding Registers* digunakan untuk membaca nilai-nilai dari *holding register* pada perangkat Modbus *slave*. *Holding register* adalah lokasi memori yang dapat digunakan untuk menyimpan data yang dapat diakses dan dimodifikasi oleh Modbus *master*. Parameter yang digunakan meliputi alamat awal dari *holding register* yang akan dibaca (*starting address*), jumlah *holding register* yang akan dibaca (*number of holding registers*), dan nilai-nilai dari *holding registers* yang telah dibaca (*register values*). Keluaran dari blok ini biasanya berupa larik atau kumpulan nilai, dengan setiap elemen larik mengandung nilai dari satu *holding register*.




(a) (b)

**Gambar 5**. Proses memasukkan nilai, (a) takaran volume, (b) jumlah botol.

*Array to Cluster Function* mengonversi larik 1D menjadi sekelompok elemen yang bertipe sama dengan elemen larik, sedangkan *Unbundle Function* digunakan untuk membagi *cluster* menjadi elemen-elemen individualnya. Elemen-elemen tersebut (seperti volume dan jumlah botol) akan menjadi *set point*. Fungsi-fungsi logika seperti *AND*, *OR*, dan *NOT* direpresentasikan dalam kendali pada Gambar 2 dalam diagram blok.

*Case Structure* digunakan untuk mengatur aliran eksekusi program berdasarkan kondisi tertentu. Dalam konteks ini, subdiagram digunakan untuk menjalankan botol pada *conveyor belt.*

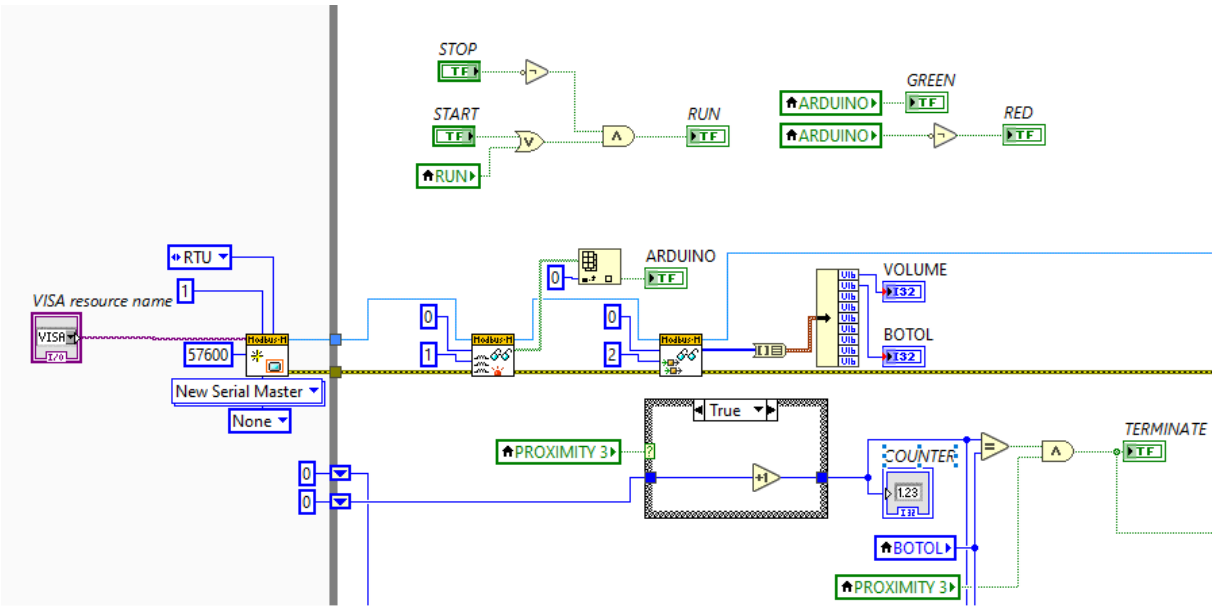
Gambar 7 menunjukkan hasil uji coba pada *front panel* LabVIEW dengan takaran volume 250 ml. Saat tombol *START* ditekan, Arduino akan mengirimkan sinyal perintah (1) untuk mengaktifkan LED hijau, kemudian menjalankan *conveyor belt* . Botol-botol diatur dan disusun pada *conveyor belt* yang bergerak menuju mesin pengisian. Saat sensor *proximity* 1 mendeteksi keberadaan botol, *conveyor belt* berhenti. Mesin pengisi botol bekerja dengan memasukkan cairan ke dalam botol secara otomatis dengan kecepatan 10 ml/s. Katup akan terbuka dan cairan akan dimasukkan ke dalam botol melalui *nozzle* yang terhubung dengan tangki cairan. Volume cairan yang diisikan sesuai dengan takaran yang sudah ditentukan. Untuk menentukan waktu respons yang diperlukan untuk mengisi volume 250 ml, digunakan (2).

$$waktu = \frac{volume}{kecepatan} = \frac{250\ ml}{10\ ml/s} = 25\ s. \qquad (2)$$

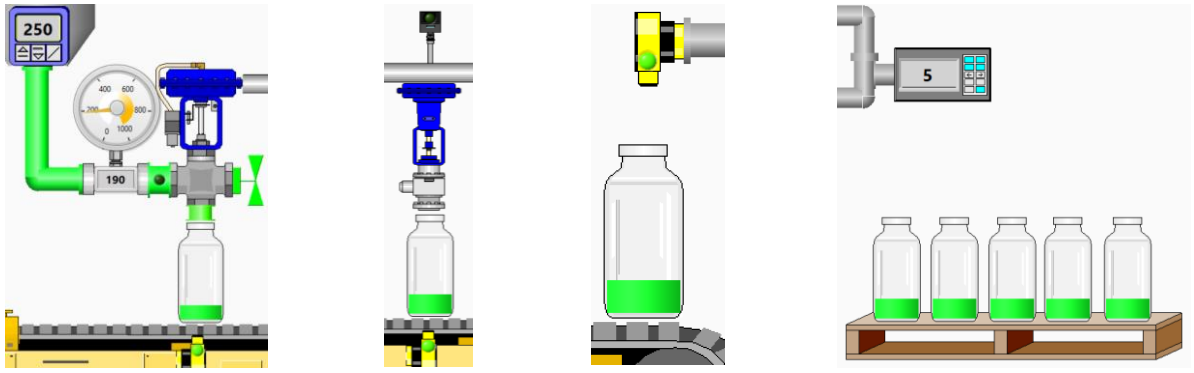
Setelah botol terisi sesuai dengan volume yang ditentukan, *conveyor belt* kembali bergerak menuju mesin penutup botol. Saat sensor *proximity* 2 mendeteksi keberadaan botol, mesin penutup botol bekerja dengan menempatkan tutup botol di atas leher botol dan menekan tutup tersebut secara otomatis hingga rapat. Setelah itu, *conveyor belt* kembali bergerak menuju proses penghitungan botol.

Botol yang melewati sensor *proximity* 3 akan dihitung jumlahnya. Sensor *proximity* 3 diprogram untuk menghitung jumlah botol yang telah diisi dan ditutup. *Plant* akan berhenti saat jumlah botol yang diinginkan tercapai sesuai dengan data jumlah botol yang ditentukan sebelumnya dan dikirimkan oleh Arduino.

Gambar 8 menunjukkan waktu yang dibutuhkan simulasi untuk mengisi cairan ke dalam botol dengan empat takaran volume yang berbeda. Pengujian dilakukan dengan masukan

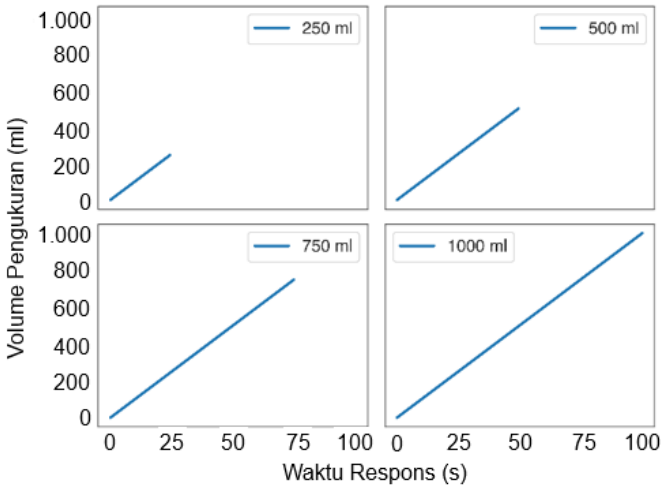



**Gambar 6**. Diagram blok Modbus pada LabVIEW.




(a) (b) (c) (d)

**Gambar 7**. Proses hasil simulasi pada *front panel* LabVIEW, (a) pengisian, (b) penutupan, (c) *counting*, (d) jumlah botol.




**Gambar 8**. Waktu respons terhadap takaran volume.

takaran volume mulai dari 250 ml, 500 ml, 750 ml, dan 1.000 ml, dengan jumlah sebanyak lima botol. *Set point* pada *front panel* LabVIEW telah sesuai dan akurat dengan takaran volume dan jumlah botol yang telah ditentukan melalui *keypad*. Hasil yang ditunjukkan pada *front panel* LabVIEW tidak membahas jenis cairan tertentu yang digunakan saat pengisian botol. Sistem kendali yang dibuat menggambarkan proses penentuan data masukan takaran volume dan jumlah botol melalui *keypad* serta menampilkan proses kendali *conveyor belt*, pengisian, penutupan, dan penghitungan pada *front panel* LabVIEW. Kendali *on-off* melalui *push button* dan komunikasi antara Arduino dengan LabVIEW menggunakan protokol Modbus RTU berjalan dengan baik sesuai dengan perancangan sistem yang dibuat untuk menjalankan *plant* pada *front panel* LabVIEW.

## IV. KESIMPULAN

Penelitian ini telah menunjukkan bahwa integrasi LabVIEW dan mikrokontroler dapat memberikan gambaran sistem mesin pengisian cairan otomatis yang efektif. Sistem ini memberikan kendali yang tepat atas takaran volume dan jumlah botol, memastikan setiap wadah terisi sesuai dengan masukan yang ditentukan, yaitu 250 ml, 500 ml, 750 ml, dan 1.000 ml. Penggunaan LabVIEW sebagai alat perancangan *plant* yang menyerupai proses industri serta integrasi yang mudah dengan mikrokontroler Arduino telah terbukti berhasil. Komunikasi antara LabVIEW dan Arduino dalam penelitian ini dilakukan menggunakan protokol Modbus RTU, yang memberikan kestabilan dan keandalan dalam pertukaran data, memastikan komunikasi yang akurat dan efisien. Hal ini menunjukkan potensi sistem ini sebagai sarana pendukung praktikum

instrumentasi sistem kendali. Pengembangan lebih lanjut dan optimalisasi sistem dapat mengarah pada adopsi yang lebih luas di berbagai aplikasi *plant* industri lainnya.

## KONFLIK KEPENTINGAN

Penulis menyatakan bahwa tidak ada konflik kepentingan dalam penelitian dan penyusunan makalah ini.

## KONTRIBUSI PENULIS

Konseptualisasi, Syafriyadi Nor dan Zaiyan Ahyadi; metodologi, Syafriyadi Nor; perangkat lunak, Syafriyadi Nor; validasi, Syafriyadi Nor dan Zaiyan Ahyadi; analisis formal, Syafriyadi Nor; investigasi, Syafriyadi Nor dan Zaiyan Ahyadi; sumber daya, Syafriyadi Nor dan Zaiyan Ahyadi; kurasi data, Syafriyadi Nor dan Zaiyan Ahyadi; penulisan—penyusunan draf asli, Syafriyadi Nor; penulisan—peninjauan dan penyuntingan, Zaiyan Ahyadi; visualisasi, Syafriyadi Nor dan Zaiyan Ahyadi.

## UCAPAN TERIMA KASIH

Terima kasih disampaikan kepada Laboratorium Mikroprosessor Politeknik Negeri Banjarmasin yang telah memfasilitasi penelitian ini.

## REFERENSI

[1] S.K. Das dkk., "Design, development and FEA analysis of multi-sized bottle filling system," dalam *2023 Int. Conf. Integr. Comput. Intell. Syst. (ICICIS)*, 2023, hal. 1–5, doi: 10.1109/ICICIS56802.2023.10430283.

[2] R.L.W. Koggalage, A.G.M.I.S. Wijesinghe, H.P.S.S. Caldera, dan R.R. Samarawickrama, "Design and implementation of an automated multi-purpose filling and capping machine," dalam *2021 From Innov. To Impact (FITI)*, 2021, hal. 1–5, doi: 10.1109/FITI54902.2021.9833035.

[3] M.M. Khan dkk., "Simulation of PLC ladder logic programming for an automated glass bottle molding and refilling plant," dalam *4th Smart Cities Symp. (SCS 2021)*, 2021, hal. 114–119, doi: 10.1049/icp.2022.0324.

[4] M.F. Rahaman, S. Bari, dan D. Veale, "Flow investigation of the product fill valve of filling machine for packaging liquid products," *J. Food Eng.*, vol. 85, no. 2, hal. 252–258, Mar. 2008, doi: 10.1016/j.jfoodeng.2007.07.020.

[5] K.S. Kiangala dan Z. Wang, "An Industry 4.0 approach to develop auto parameter configuration of a bottling process in a small to medium scale industry using PLC and SCADA," dalam *2nd Int. Conf. Sustain. Mater. Process. Manuf. (SMPM 2019)*, 2019, hal. 725–730, doi: 10.1016/j.promfg.2019.06.015.

[6] G. Selvaraj, R. Karthikeyan, S. Brindha, dan K. Kumar S, "Low cost assorted sized bottles automated liquid filling system using SCADA," dalam *2023 Intell. Comput. Control Eng. Bus. Syst. (ICCEBS)*, 2023, hal. 1–4, doi: 10.1109/ICCEBS58601.2023.10448914.

[7] A. Mahrez dkk., "Design a PLC-based automated and controlled liquid filling-capping system," dalam *2022 Int. Eng. Conf. Elect. Energy Artif. Intell. (EICEEAI)*, 2022, hal. 1–5, doi: 10.1109/EICEEAI56378.2022.10050478.

[8] C. Elliott, V. Vijayakumar, W. Zink, dan R. Hansen, "National instruments LabVIEW: A programming environment for laboratory automation and measurement," *J. Lab. Autom.*, vol. 12, no. 1, hal. 17–24, Feb. 2007, doi: 10.1016/j.jala.2006.07.012.

[9] Arduino. "What is Arduino? | Arduino." Tanggal akses: 20-Feb-2023. [Online]. Tersedia: https://www.arduino.cc/en/Guide/Introduction

[10] M.L. Ahmed, S. Kundu, dan M. Rafiquzzaman, "Automatic bottle filling system using PLC based controller," *J. Adv. Mech.*, vol. 4, no. 1, hal. 17–24, Mar. 2019.

[11] R. Sureshkumar dkk., "IoT based bottle filling system using PLC," dalam *2023 Int. Conf. Energy Mater. Commun. Eng. (ICEMCE)*, 2023, hal. 1–5, doi: 10.1109/ICEMCE57940.2023.10433948.

[12] A. Kumar M dan H.P. Kumar, "Automatic bottle filling system using PLC," *Int. J. Trend Sci. Res. Dev. (IJTSRD)*, vol. 2, no. 1, hal. 361–364, Nov./Des. 2017, doi: 10.31142/ijtsrd5953.

[13] G.A. Laksmana, P. Santoso, dan F. Pasila, "Aplikasi untuk memonitor PLC pada mesin filling dan capping," *J. Tek. Elekt.*, vol. 10, no. 2, hal. 48–53, Sep. 2017, doi: 10.9744/jte.10.2.48-53.

[14] D. Patil, "Automatic bottle filling, capping and labelling system using PLC based controller," *Ilkogretim*, vol. 20, no. 1, hal. 5750–5761, 2021, doi: 10.17051/ilkonline.2021.01.604.

[15] O.I. Abdullah, W.T. Abbood, dan H.K. Hussein, "Development of automated liquid filling system based on the interactive design approach," *FME Trans.*, vol. 48, no. 4, hal. 938–945, Agu. 2020, doi: 10.5937/fme2004938A.

[16] M. Aria dkk., "Virtual simulation system with various examples and analysis tools for programmable logic controller training," dalam *3rd Int. Conf. Inform. Eng. Sci. Technol. (INCITEST 2020)*, 2020, hal. 1–7, doi: 10.1088/1757-899X/879/1/012108.

[17] A. El Hammoumi dkk., "Real-time virtual instrumentation of Arduino and LabVIEW based PV panel characteristics," dalam *Int. Conf. Renew. Energ. Energy Effic. (REEE'2017)*, 2018, hal. 1–11, doi: 10.1088/1755-1315/161/1/012019.

[18] R.M. Shrenika dkk., "Non-contact water level monitoring system implemented using LabVIEW and Arduino," dalam *2017 Int. Conf. Recent Adv. Electron. Commun. Technol. (ICRAECT)*, 2017, hal. 306–309, doi: 10.1109/ICRAECT.2017.51.

[19] Y.K. Taru dan A. Karwankar, "Water monitoring system using Arduino with LabVIEW," dalam *2017 Int. Conf. Comput. Methodol. Commun. (ICCMC)*, 2017, hal. 416–419, doi: 10.1109/ICCMC.2017.8282722.

[20] M. Gharte, "Automation of soap windscreen washer filling machine with PLC and LabVIEW," dalam *Int. Conf. Autom. Control Dyn. Optim. Tech. (ICACDOT)*, 2016, hal. 469–472, doi: 10.1109/ICACDOT.2016.7877630.

[21] Arduino. "Arduino ® Nano Arduino ® Nano Features." Tanggal akses: 6-Feb-2024. [Online]. Tersedia: https://docs.arduino.cc/hardware/nano/

[22] A. Analysis and U. Manual, LabVIEW User Manual, vol. 3304, no. Januari. 2012.

[23] C.M. Bulliner. "ModbusRTUSlave." Tanggal akses: 6-Feb-2024. [Online]. Tersedia: https://github.com/CMB27/ModbusRTUSlave