Pemrograman Aplikasi Teknik:
Pembuatan SCADA dan Simulasinya

oleh:
Dian Artanto

PRAKATA

Seluruh tujuan pendidikan adalah mengubah cermin menjadi jendela. (Sydney J. Harris)

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa, karena hanya atas berkat dan rahmatNya saja, akhirnya buku ajar Pemrograman Aplikasi Teknik ini dapat selesai.

Latar belakang penulisan buku ajar ini adalah karena motivasi yang diberikan oleh PPIP Universitas Sanata Dharma kepada penulis untuk bisa menyiapkan materi ajar yang lebih baik dan "dipersiapkan". Dipersiapkan dalam arti betul-betul siap diberikan ke siswa/mahasiswa dan bisa dipahami. Untuk tujuan itu, maka buku ajar ini sengaja dibuat dalam bentuk tulisan yang singkat, dengan banyak gambar, langkah langkah praktik dan soal latihan. Karena dengan banyak praktik, penulis percaya siswa/mahasiswa akan lebih banyak belajar, seperti yang banyak didengungkan dalam metode pembelajaran, yaitu *Learning by Doing*.

Dalam masa sekarang ini, pembelajaran pemrograman sangat dibutuhkan, mengingat teknologi informasi yang telah berkembang sangat pesat. Dalam buku ini, pembelajaran pemrograman difokuskan pada pemrograman aplikasi teknik. Program aplikasi teknik yang masih populer hingga sekarang, dan menjadi bagian penting dalam Era Industri 4.0 ini adalah SCADA (*Supervisory Control And Data Acquisition*). SCADA penting dipelajari karena melibatkan banyak alat dan komponen, yang harus diintegrasikan, dimonitor dan dikontrol secara efektif dan efisien. Buku ini menyajikan cara pembuatan SCADA secara praktis dan sederhana, melibatkan penggunaan software: LabVIEW, Arduino IDE, Outseal Studio, Proteus, ModRSsim2, KEPServerEX dan Node-RED. Mengapa menggunakan banyak software? Agar pembaca buku ini mendapat banyak pengalaman dalam pemrograman, mulai dari pemrograman grafis (LabVIEW), teks (Arduino), ladder diagram (Outseal), simulasi (Proteus dan ModRSsim2), pengaturan konfigurasi (KEPServerEX), hingga pembuatan flow (Node-RED).

Aplikasi SCADA yang dibuat dimulai dari pembuatan tampilan SCADA, dilanjutkan dengan penyambungan alat secara virtual menggunakan Proteus, baik sambungan RS-232 maupun RS-485, dengan komunikasi Serial, Modbus dan OPC UA, dilanjutkan dengan pembuatan fitur SCADA, termasuk fitur Database, Alarm, Trend, dan Reporting, diakhiri dengan pembuatan IoT (Internet of Things) menggunakan Node-RED, yang dapat mengirimkan data alarm/peringatan melalui email dan Whatsapp, serta menampilkan user interface (Dashboard), yang dapat memonitor dan mengontrol data SCADA melalui Web (menggunakan REST Server) dan HP atau Smartphone Android (menggunakan MQTT).

Semua penjelasan dalam buku ini disertai dengan banyak gambar, dan dapat disimulasikan hasilnya, sehingga sangat cocok untuk diterapkan pada pembelajaran secara daring.

Secara ringkas, berikut ini isi dari buku ajar Pemrograman Aplikasi Teknik Pembuatan SCADA dan Simulasinya, yang terdiri dari 6 Bab, dengan keterangan setiap babnya sebagai berikut:

1. Pembuatan Tampilan

   Bab ini berisi pembuatan tampilan SCADA menggunakan LabVIEW. Tampilan berupa sebuah proses pengisian tangki dan animasinya.

2. Komunikasi Serial

   Bab ini berisi komunikasi antara tampilan SCADA LabVIEW dengan Arduino melalui komunikasi serial UART dan pengawatan rangkaiannya. Software Proteus digunakan untuk mensimulasikan rangkaian Arduino dan komponen input-outputnya dan komunikasinya dengan LabVIEW.

3. Protokol Modbus RTU

   Bab ini berisi komunikasi antara tampilan SCADA LabVIEW dengan 2 buah Arduino melalui komunikasi serial RS-485 dengan protokol Modbus RTU dan pengawatan rangkaiannya. Software ModRSsim2 digunakan untuk mensimulasikan komunikasi protokol Modbus. Software Proteus digunakan untuk mensimulasikan rangkaian RS-485 Arduino dan komunikasinya dengan LabVIEW. Outseal juga diperkenalkan di sini sebagai alternatif yang menarik untuk pemrograman Modbus RTU pada Arduino.

4. OPC Server

   Bab ini mengulangi isi dari Bab 3, hanya bedanya, di Bab ini digunakan OPC Server, yaitu NI OPC Server dan Kepware KEPServerEX, sebagai alternatif yang mudah untuk membuat protokol Modbus RTU di LabVIEW. Juga dibahas mengenai penerapan OPC UA, yang memiliki fitur koneksi yang lebih baik.

5. Database

   Bab ini berisi pembuatan fitur SCADA secara umum, yang memiliki Database, Alarm dan Trend, baik secara Real Time maupun Historical serta Reporting. Semua fitur tersebut dibuat menggunakan Datalogger KEPServerEX Kepware, yang dapat menyimpan data Tag, baik secara periodis maupun saat data berubah, dan software Microsoft Office (Access dan Excel).

6. IoT Gateway

   Bab terakhir ini berisi koneksi SCADA ke Internet of Things, dengan bantuan IoT Gateway KEPServerEX dan software Node-RED yang populer. Dengan REST Client, REST Server dan MQTT serta Node-RED, dapat dibuat alarm yang dapat mengirimkan peringatan melalui email dan Whatsapp, menampilkan dan mengubah data Tag melalui tampilan user interface (dashboard) di Web, dan melalui aplikasi MQTT Dashboard di smartphone (HP Android).

Akhir kata, masih banyak kekurangan dan keterbatasan dalam buku ini. Penulis berharap, seperti kata mutiara yang tertulis di atas, semoga buku ini dapat menjadi jendela yang membuka wawasan dan bermanfaat bagi pembaca.

Salam hangat.

DAFTAR ISI

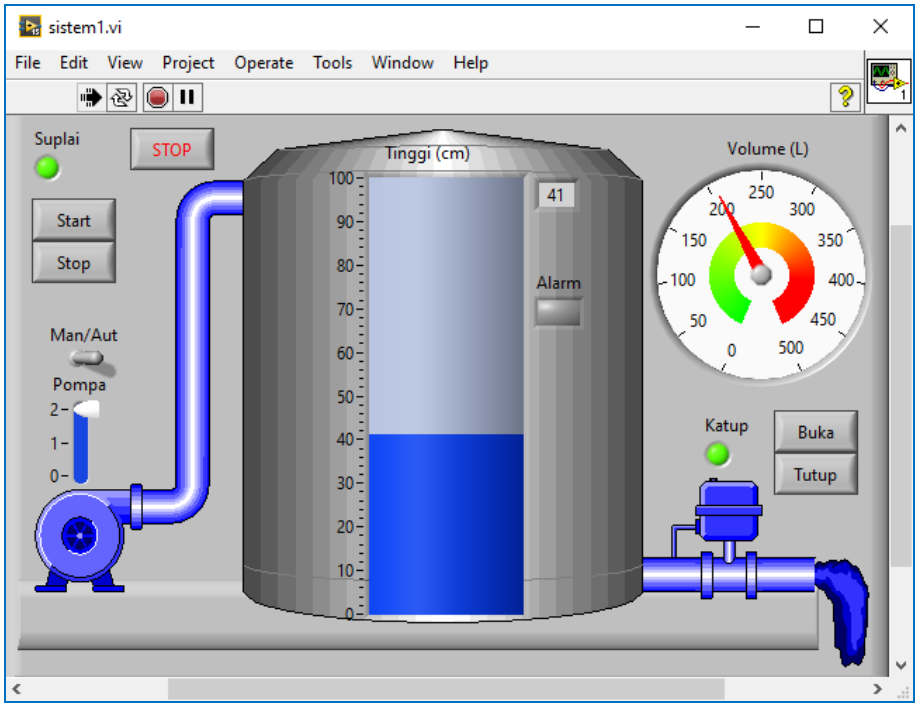
BAB 1
PEMBUATAN TAMPILAN SCADA

Target Materi:

- Menerapkan struktur pemrograman dasar LabVIEW.
- Membuat program tampilan dan animasinya di LabVIEW.

Tantangan:

➢ Pembaca dapat membuat tampilan program pengisian tangki seperti gambar berikut:



Gambar 1.1 Tampilan Proses Pengisian Tangki

1.1 Persiapan Perangkat yang Diperlukan

Bab ini memerlukan perangkat lunak (software) sebagai berikut:

1. Software LabVIEW
2. Software Add-On DSC Module

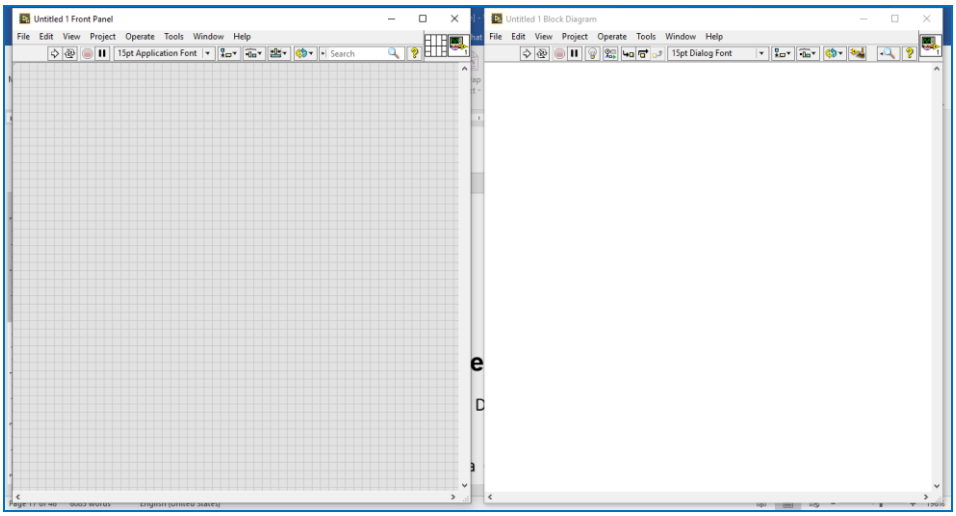
Software LabVIEW merupakan software untuk pembuatan program, sedangkan software Add-On DSC (*Datalogging and Supervisory Control*) Module merupakan software tambahan LabVIEW, yang memudahkan pembuatan program SCADA. Pembaca dapat mengunduh kedua software di atas di: ni.com/downloads. Pastikan software LabVIEW yang diunduh adalah versi yang 32 bit, agar kompatibel dengan software tambahan (*Add-On*) seperti DSC Module. Versi LabVIEW dan DSC Module yang penulis gunakan dalam buku ini adalah versi 2015. Untuk mudahnya, pembaca juga dapat mengunduh kedua software di atas di blog penulis: dianartanto@blogspot.com atau bisa menghubungi penulis lewat email: dian.artanto@gmail.com.

Setelah kedua software tersebut berhasil diunduh, jalankan hasil unduhan tersebut. Ketika diminta mengisi kode lisensi, abaikan dan tekan tombol *Next* (otomatis menjadi versi *Evaluation*). Tekan tombol *Next*, hingga muncul *Instal Hardware Support for LabVIEW*, pilih *Decline Support* (*Hardware Support* akan diinstal di Bab 2). Sekalipun versi *Evaluation*, namun fitur yang dimiliki adalah sama seperti versi *Professional*, hanya saja masa penggunaan versi evaluasi ini dibatasi. Trik yang dapat digunakan agar bisa memakai LabVIEW seterusnya, sekalipun masa *Evaluation* sudah berakhir adalah dengan cara, sebelum membuka LabVIEW, mundurkan waktu di komputer ke tahun 2000. Setelah waktu dimundurkan, buka LabVIEW, maka LabVIEW dapat digunakan lagi (jangan lupa untuk mengembalikan ke waktu sekarang setelah software LabVIEW terbuka).

Buka LabVIEW. Pada jendela yang terbuka, klik menu File, dan pilih New VI, maka akan muncul 2 jendela. Jendela yang depan, berwarna abu-abu kotak-kotak, disebut Front Panel, sedangkan jendela yang di belakang berwarna putih, disebut Block Diagram. Tekan tombol Control dan T bersama-sama, maka kedua jendela tersebut akan berdampingan, Front Panel di kiri dan Block Diagram di kanan.

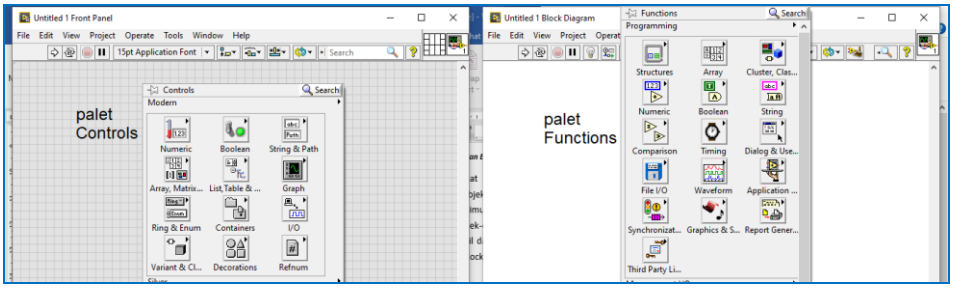
1.2 Pengenalan Struktur Pemrograman

Berikut ini tampilan Front Panel dan Block Diagram yang berdampingan:



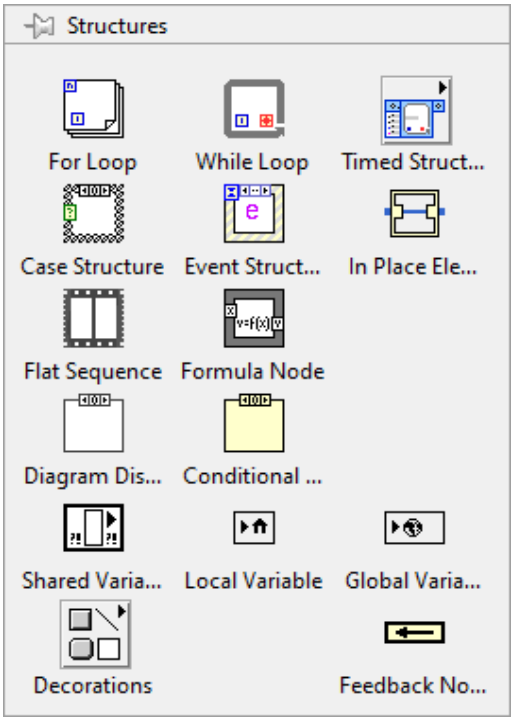
Gambar 1.2 Front Panel dan Block Diagram LabVIEW

Front Panel digunakan untuk membuat tampilan, sedangkan Block Diagram digunakan untuk membuat program. Objek-objek untuk tampilan di Front Panel dapat diambil dari palet Controls, yang dimunculkan dengan cara meng-klik kanan halaman Front Panel. Sedangkan objek-objek (berbentuk icon) untuk kode program di Block Diagram dapat diambil dari palet Functions, yang dimunculkan dengan cara meng-klik kanan halaman Block Diagram. Berikut tampilan Palet Controls di Front Panel dan palet Functions di Block Diagram.



Gambar 1.3 Palet Controls di Front Panel dan palet Functions di Block Diagram

Untuk lebih jelasnya mengenai penggunaan Tool-tool yang ada di Front Panel dan Block Diagram, pembaca dapat membaca buku penulis yang berjudul “Interaksi Arduino dan LabVIEW”, yang dapat pembaca unduh di halaman blog penulis: dian.artanto@blogspot.com. Sesuai dengan judul Sub Bab, maka berikut ini akan dijelaskan mengenai Struktur Pemrograman LabVIEW. Struktur Pemrograman LabVIEW dapat dilihat di palet Functions, di kategori Programming, di kategori Structures. Berikut isi kategori Structures:



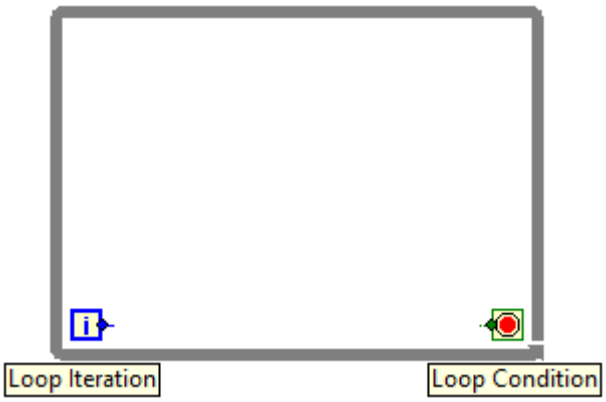
Gambar 1.4 Struktur pemrograman LabVIEW ada di palet Functions di kategori Structures

Terlihat ada bermacam-macam Struktur Pemrograman tersedia. Di Sub Bab ini, hanya dibahas 4 Struktur Pemrograman saja, sedangkan Struktur yang lain akan dibahas di bab-bab berikutnya. Empat Struktur yang akan dibahas adalah:

1. While Loop
2. Case Structure
3. Local Variable
4. Shift Register

1.2.1 While Loop

Struktur While Loop memiliki 3 komponen utama, yaitu sebuah kotak yang dapat diatur luasannya, sebuah terminal Loop Condition (kotak dengan bulatan merah di dalamnya) dan sebuah terminal Loop Iteration (kotak dengan huruf i).



Gambar 1.5 Tiga komponen While Loop: Kotak While Loop, Loop Iteration dan Loop Condition

Ketika program dijalankan, Struktur While Loop ini akan menjalankan kode program atau icon-icon di dalam kotaknya secara berulang terus-menerus, dan hanya berhenti ketika terminal Loop Condition (*Stop If True*) diberi nilai True.

Terminal Loop Condition tersebut dapat diubah dari mode default *Stop If True* menjadi mode *Continue If True.* Apabila mode *Stop If True* berhenti melakukan perulangan apabila diberi True, sebaliknya mode *Continue If True* akan melakukan perulangan apabila diberi True, dan hanya berhenti ketika diberi False.

Terminal Loop iteration (kotak dengan huruf i) akan menghasilkan nilai jumlah perulangan yang dilakukan, yang nilainya dimulai dari 0.

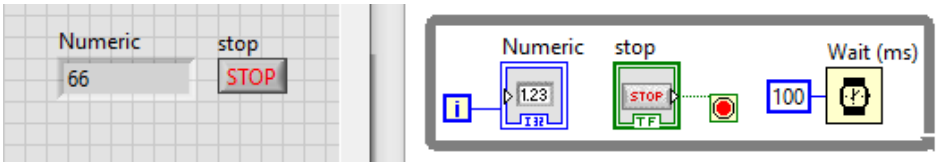
Untuk mendapatkan gambaran mengenai penggunaan While Loop, buat program seperti Gambar 1.6. Untuk membuatnya, ikuti langkah-langkah berikut:

1. Pilih While Loop di Structures, di kategori Programming di palet Functions, dan kemudian buat kotak While Loop di halaman Block Diagram.
2. Klik kanan pada kaki input terminal Loop Condition, dan pilih Create, pilih Control, maka akan muncul icon Stop terhubung dengan Loop Condition.

3. Klik kanan pada kaki output terminal Loop Iteration, dan pilih Create, pilih Indicator, maka muncul icon Numeric Indicator.
4. Perhatikan di Front Panel, ternyata ada 2 objek yang tampak di Front Panel, yaitu objek tombol Stop dan objek kotak Numeric Indicator. Jadi ketika icon Control dan icon Indicator dibuat (*Create*) di Block Diagram, secara otomatis muncul objeknya di Front Panel.
5. Jalankan program dengan menekan tombol Run (tombol paling kiri pada Toolbar), dan perhatikan bahwa angka di kotak Numeric Indicator di Front Panel, bertambah sangat cepat. Tekan tombol Stop untuk menghentikan perulangan.

Gambar 1.6 Contoh penggunaan Struktur While Loop untuk melakukan pengulangan

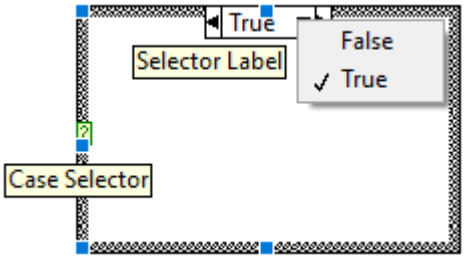
Keterangan: Gambar 1.6 di atas menunjukkan tampilan Front Panel dan Block Diagram setelah program dijalankan. Kotak Numeric di Front Panel menampilkan nilai Loop Iteration (i) atau jumlah perulangan. Tampak bahwa perulangan yang dilakukan sangat banyak, hingga 16.786.966, dan itu dihasilkan hanya dalam waktu beberapa detik saja. Tentunya dengan jumlah perulangan yang besar ini membuat prosesor komputer bekerja keras. Untuk menghemat sumber daya prosesor ini, biasanya ditambahkan sebuah "delay" atau waktu tunggu di dalam While Loop, yaitu dengan icon Wait(ms) dengan nilai input 100. Icon Wait(ms) ini diambil dari palet Functions di kategori Programming, di kategori Timing. Untuk memberi input 100, klik kanan kaki input icon Wait(ms), pilih Create, pilih Constant, dan ganti angka 0 dengan angka 100.



Gambar 1.7 Untuk menghemat resource komputer, Wait(ms) ditambahkan pada While Loop

1.2.2 Case Structure

Struktur Case bisa memiliki 2 atau lebih kotak, namun hanya satu saja yang ditampilkan, dan juga hanya satu saja yang bisa dijalankan pada satu waktu. Untuk melihat kotak yang lain di Struktur Case, tekan tombol panah di kiri dan kanan Selector Label. Selain kotak Case dan Selector Label, ada 1 komponen penting yang dimiliki oleh Struktur Case, yaitu Terminal Case Selector (kotak dengan tanda tanya). Case Selector ini digunakan untuk menentukan kotak mana yang dipilih untuk dijalankan. Jadi Struktur Case ini sama seperti Instruksi percabangan If, di mana input Case Selector menjadi syarat pada instruksi If tersebut.

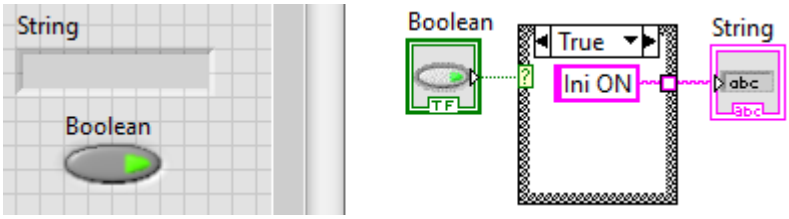


Gambar 1.8 Tiga komponen Case Structure: Kotak Case, Selector Labe, dan Case Selector

Untuk mendapatkan gambaran mengenai penggunaan Struktur Case, buat program seperti Gambar 1.10. Untuk membuatnya, ikuti langkah-langkah berikut:

1. Pilih Case Structure di Structures, di kategori Programming di palet Functions, dan kemudian buat kotak di halaman Block Diagram.
2. Klik kanan pada kaki input Terminal Case Selector (kotak tanda tanya), dan pilih Create, pilih Control, maka muncul icon Boolean terhubung ke Case Selector, dan juga muncul objek Tombol Boolean di Front Panel.
3. Di Front Panel, tempatkan sebuah objek String Indicator, yang diambil dari palet Controls, di kategori Modern, di kategori String & Path. Penempatan objek String Indicator di Front Panel ini akan memunculkan icon String Indicator di Block Diagram.
4. Di Block Diagram, tempatkan icon String Indicator di luar kotak Case. Klik kanan pada kaki input icon String, dan pilih Create, pilih Constant, maka muncul kotak String kosong (*Empty String*). Isi kotak String tersebut

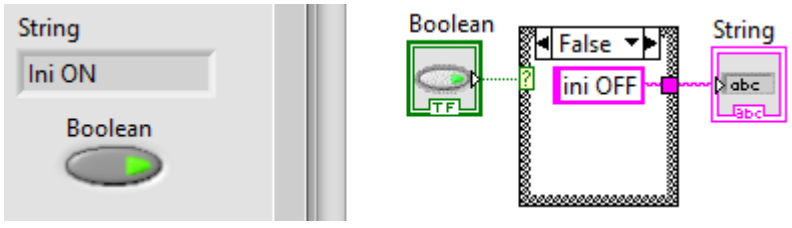
dengan tulisan “ini ON”. Pindahkan kotak String “ini ON” ke dalam kotak Case True, dan kemudian hubungkan kembali kotak String “ini ON” dengan icon String Indicator di luar kotak Case.



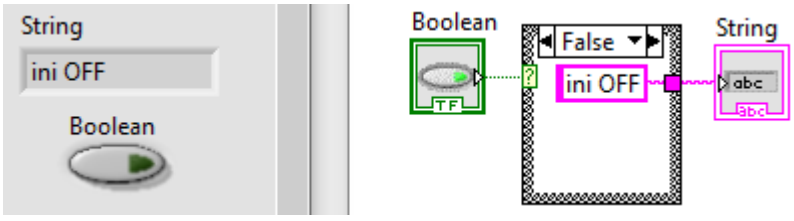
Gambar 1.9 Menempatkan “ini ON” di Case True dan menghubungkannya dengan icon String

5. Berikutnya, pindah ke kotak Case False dengan menekan tombol tanda panah di samping Selector Label. Di Case False, klik kanan pada kotak kecil kosong di dinding kotak Case, dan pilih Create, pilih Constant. Pada kotak String kosong yang muncul, isi dengan “ini OFF”.
6. Karena tidak ada Struktur perulangan While Loop dalam program, maka jalankan program dengan menekan tombol Run Continuously (tombol kedua dari kiri di Toolbar, bergambar 2 panah yang melingkar). Kemudian tekan tombol Boolean, perhatikan ketika LED di tombol menyala, muncul tulisan “ini ON” pada String Indicator. Ketika tombol Boolean ditekan lagi hingga LED di tombol Boolean padam, muncul tulisan “ini OFF”.

(a).



(b).



Gambar 1.10(a) Ketika Boolean True, String “ini ON”, (b) ketika Boolean False, String “ini OFF”

Keterangan: Gambar 1.10 (a) di atas menunjukkan tampilan Front Panel dan Block Diagram ketika tombol Boolean ditekan hingga bernilai True (LED pada tombol menyala), membuat Struktur Case menjalankan Case True, yang mengisi String Indicator dengan tulisan “ini ON”. Gambar 1.10 (b) di atas menunjukkan tampilan Front Panel dan Block Diagram ketika tombol Boolean ditekan hingga bernilai False (LED pada tombol padam), membuat Struktur Case menjalankan Case False, yang mengisi String Indicator dengan tulisan “ini OFF”.

1.2.3 Local Variable

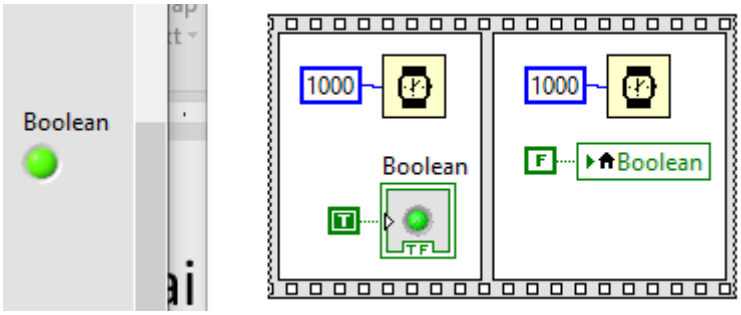
Struktur Local Variable berfungsi sama seperti sebuah kawat, yaitu untuk meneruskan data dari objek Control maupun Indicator ke terminal objek lain, dengan mengacu pada nama objek yang datanya diteruskan tersebut. Selain sebagai penerus data, Struktur Local Variable juga digunakan untuk merepresentasikan sebuah objek di beberapa tempat di dalam kode program,

Struktur Local Variable membuat tampilan menjadi lebih mudah dibaca karena menghilangkan kawat, namun kekurangannya, dibanding dengan pengawatan langsung adalah Local Variable membutuhkan satu kali siklus perulangan untuk meng-“update” datanya. Karena itu Local Variable tidak dapat digunakan untuk meneruskan data yang bersifat *momentary* (sesaat), seperti tombol OK button.

Untuk mendapatkan gambaran mengenai penggunaan Local Variable, buat program LED berkedip seperti Gambar 1.11. Untuk membuat program tersebut, ikuti langkah-langkah berikut:

1. Di Front Panel, tempatkan sebuah Round LED yang diambil dari palet Controls, di kategori Modern, di kategori Boolean.
2. Di Block Diagram, buat kotak Struktur Flat Sequence (berbentuk pita film), yang diambil dari palet Functions, di kategori Programming, di Structures.
3. Tempatkan icon Boolean Round LED ke dalam kotak Flat Sequence tersebut. Beri nilai Boolean tersebut dengan nilai True, dengan cara meng-klik kanan kaki input Boolean tersebut, dan pilih Create, pilih Constant, maka akan muncul nilai False Constant. Klik pada nilai default False tersebut agar berubah menjadi True.

4. Pada kotak Flat Sequence yang sama, tempatkan sebuah icon Wait (ms), yang diambil dari palet Functions, di kategori Programming, di kategori Timing. Klik kanan kaki input icon Wait (ms), pilih Create, pilih Constant. Ganti angka 0 dengan angka 1000 (nilai 1000 dalam ms berarti 1 detik).
5. Berikutnya, tambahkan Frame di kanan kotak Flat Sequence dengan cara mengklik kanan tepi kotak Flat Sequence, dan pilih Add Frame After.
6. Pada kotak Frame yang kedua, tempatkan icon Wait(ms) dengan nilai input 1000, dan sebuah Local Variable dari Boolean Round LED. Untuk membuat Local Variable Boolean Round LED ini dilakukan dengan cara meng-klik kanan Boolean Round LED, pilih Create, pilih Local Variable.
7. Beri nilai input Local Variable Boolean Round LED dengan nilai False Constant, yang dibuat dengan cara meng-klik kanan kaki input Local Variable, pilih Create, pilih Constant.
8. Karena di dalam program tidak ada Struktur perulangan While Loop, jalankan program dengan menekan tombol Run Continuously. Perhatikan bahwa Round LED dapat bergantian menyala dan padam setiap detik.



Gambar 1.11 Membuat Boolean Round LED berkedip dengan bantuan Local Variable

Keterangan: Gambar 1.11 di atas menunjukkan tampilan Front Panel dan Block Diagram untuk program LED berkedip. Struktur Flat Sequence memaksa program menjalankan instruksi dari kotak paling kiri ke kotak di kanannya secara berurutan. Kotak di kiri dijalankan selama 1000ms atau 1 detik, dan kotak di kanannya juga dijalankan selama 1 detik. Kotak di kiri membuat LED (Boolean) menyala, sedang kotak di kanan membuat LED padam. Untuk merujuk pada objek yang sama, di kotak yang kanan digunakan Local Variable dari objek LED tersebut.

1.2.4 Shift Register

Sebuah Struktur Pemrograman di LabVIEW yang tidak ada di palet Functions adalah Shift Register. Struktur Shift Register ini tidak tersedia di palet Functions, karena penggunaannya hanya bisa diterapkan pada struktur perulangan For Loop dan While Loop saja. Untuk menggunakan struktur Shift Register ini, klik kanan dinding kotak For Loop atau While Loop, dan kemudian pilih Add Shift Register dari menu popup yang muncul.

Shift Register ini dapat digunakan untuk mendapatkan data sebelum perulangan, dan untuk menghantarkan data dari satu perulangan ke perulangan berikutnya. Bentuk Shift Register ini berupa pasangan terminal di dinding kiri dan kanan blok, bergambar tanda panah ke bawah dan ke atas.

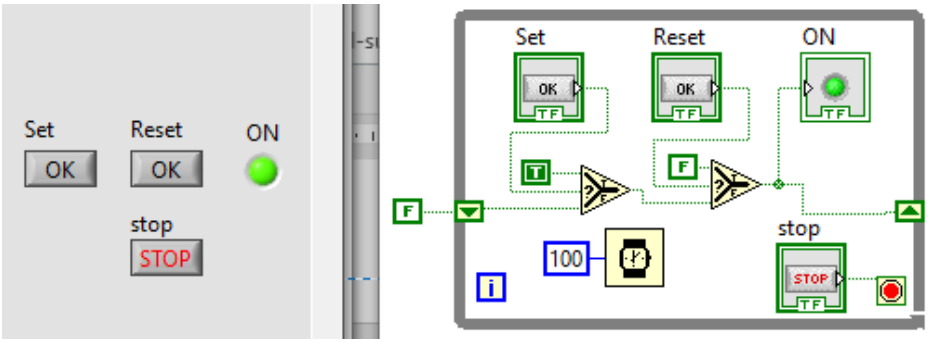
Untuk mendapatkan gambaran mengenai penggunaan Shift Register, buat program Set Reset di Gambar 1.13, dengan mengikuti langkah-langkah berikut:

1. Di Front Panel, tempatkan 2 buah objek OK Button dan sebuah Round LED yang diambil dari palet Controls, di kategori Modern, di kategori Boolean. Penempatan ketiga objek tersebut di Front Panel secara otomatis akan memunculkan icon-icon ketiga objek tersebut di Block Diagram.
2. Beri nama ketiga objek tersebut secara berturut-turut: Set, Reset dan ON.

Gambar 1.12 Menempatkan tiga buah objek di Front Panel: Set, Reset dan ON

3. Di Block Diagram, ambil Struktur While Loop dari palet Functions, di kategori Programming, di Structures. Buat kotak While Loop, dan masukkan ketiga icon Set, Reset dan ON ke dalam kotak While Loop.
4. Klik kanan pada kaki input Terminal Loop Condition, pilih Create, pilih Controls, maka akan muncul icon tombol Stop.
5. Tambahkan di dalam kotak While Loop, icon Wait(ms) yang diambil dari palet Functions, di kategori Programming, di kategori Timing. Beri nilai 100 pada icon Wait(ms).

6. Berikutnya, klik kanan pada dinding kotak While Loop, pilih Add Shift Register, maka muncul 2 buah Terminal Shit Register, berbentuk kotak dengan panah ke bawah dan ke atas, di kiri kanan dinding While Loop.
7. Beri nilai awal Terminal Shift Register yang di kiri dengan nilai False Constant. Untuk mendapatkan icon False Constant, ambil dari palet Functions, di kategori Programming, di Boolean.
8. Berikutnya, ambil 2 buah icon Select, dari palet Functions, di kategori Programming, di kategori Comparison. Hubungkan kaki input tengah icon Select pertama dengan tombol Set, dan kaki input tengah icon Select kedua dengan tombol Reset.
9. Hubungkan kaki input yang lain seperti Gambar 1.13.
10. Jalankan program dengan menekan tombol Run (jangan menekan tombol Run Continuously, karena di dalam program sudah ada Struktur Perulangan). Tekan tombol Set dan lepaskan tombol, maka Round LED akan menyala. Tekan tombol Reset dan lepaskan tombol, maka Round LED akan padam. Tekan tombol Stop, maka program akan berhenti.



Gambar 1.13 Program Set Reset dengan icon Select, Shift Register dan While Loop

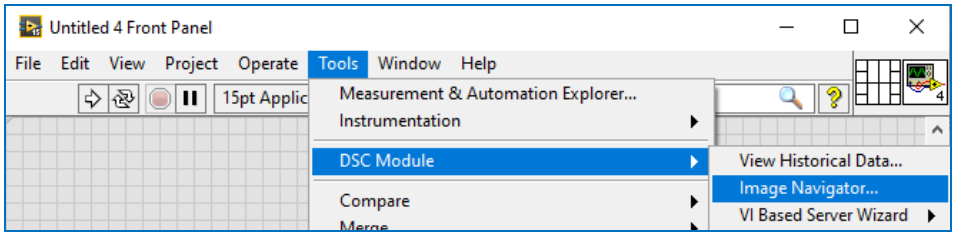
Keterangan: Dari program pada Gambar 1.13 di atas, mula-mula Shift Register berisi nilai False. Ketika tombol Set ditekan, icon Select meneruskan nilai True, menbuat Shift Register berisi nilai True. Ketika tombol Reset ditekan, icon Select meneruskan nilai False, membuat Shift Register berisi nilai False. Nilai True atau False di Shift Register ini diterima oleh LED, yang membuat LED nyala atau padam.

1.3 Pembuatan Tampilan dan Animasinya

Setelah mengetahui beberapa Struktur Pemrograman LabVIEW, maka pada Sub Bab ini, akan ditunjukkan penerapan dari Struktur Pemrograman tersebut. Untuk memudahkan penjelasan, Sub Bab ini dibagi menjadi 3 tahap; tahap pertama membuat tampilan, tahap kedua membuat objek animasi, dan tahap ketiga membuat program pengisian Tangki.

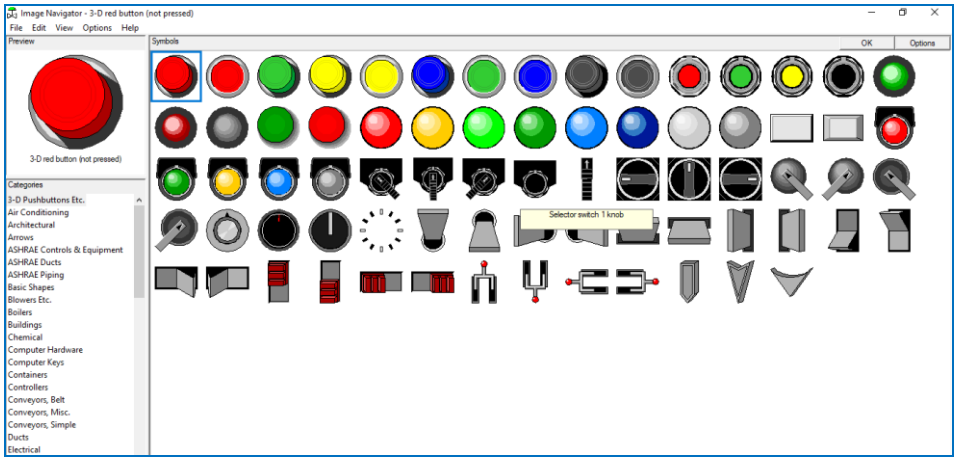
1.3.1 Pembuatan Tampilan Pengisian Tangki

Pembuatan Tampilan Pengisian Tangki ini memerlukan Image Navigator, yang ada di menu Tools, di kategori DSC Module.

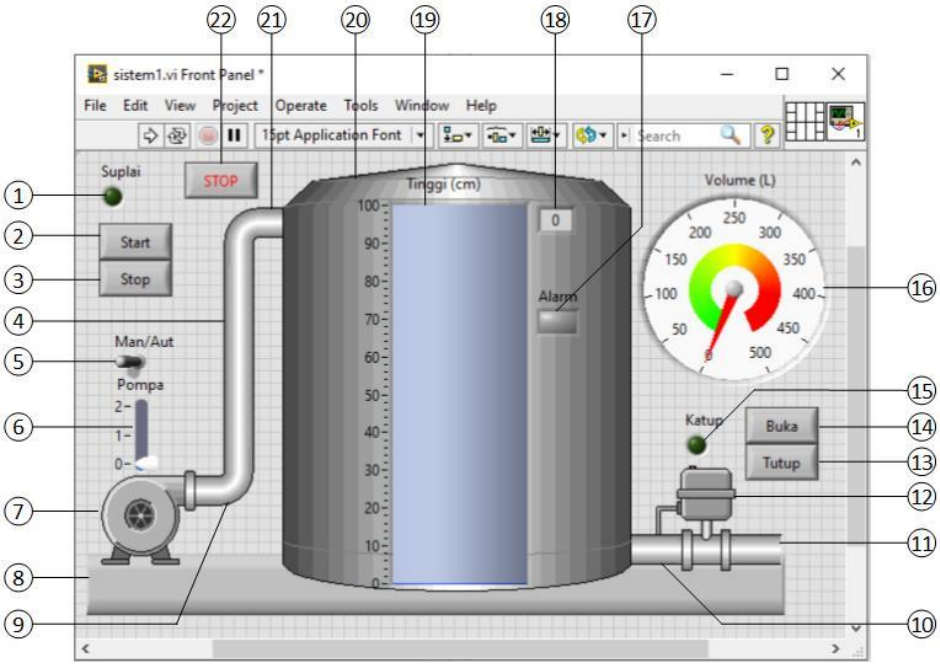


Gambar 1.14 Membuka Image Navigator di DSC Module di menu Tools

Buka Image Navigator, maka akan muncul jendela yang berisi bermacam-macam gambar objek yang menarik, yang dapat digunakan untuk membuat Tampilan.



Gambar 1.15 Kumpulan gambar pada Image Navigator



Gambar 1.16 Tampilan Pengisian Tangki

Gambar 1.16 di atas menunjukkan tampilan yang akan dibuat. Ada 22 objek yang diperlukan. Berikut ini tabel daftar objek dan keterangan perolehannya:

Tabel 1.1 Daftar Objek pada Tampilan Pengisian Tangki

No.	Nama Objek	Lokasi Perolehan	Ukuran (grid)
1.	Round LED Suplai	Palet Controls\Modern\Boolean	Default
2.	Tombol Start	Palet Controls\Modern\Boolean	6 x 3
3.	Tombol Stop	Palet Controls\Modern\Boolean	6 x 3
4.	Short Vertical Pipe	Image Navigator\Pipes	16 x 2,5
5.	Toggle Man/Aut	Palet Controls\Modern\Boolean	Default
6.	Vertical Pointer Slide	Palet Controls\Modern\Numeric	4 x 6
7.	Cool Pump	Image Navigator\Pumps	8 x 8
8.	Vertical Smooth Box	Palet Controls\Modern\Decorations	57 x 5

No.	Nama Objek	Lokasi Perolehan	Ukuran (grid)
9.	90^0 curve 4	Image Navigator\Pipes	4,5 x 4,5
10.	Short Horizontal Pipe	Image Navigator\Pipes	3,5 x 2,5
11.	Short Horizontal Pipe	Image Navigator\Pipes	3,5 x 2,5
12.	Control Valves – Hor.	Image Navigator\Valves	7 x 9
13.	Tombol Buka	Palet Controls\Modern\Boolean	6 x 3
14.	Tombol Tutup	Palet Controls\Modern\Boolean	6 x 3
15.	Round Led Katup	Palet Controls\Modern\Boolean	Default
16.	Gauge	Palet Controls\Modern\Numeric	16 x 16
17.	Square LED Alarm	Palet Controls\Modern\Boolean	3 x 2
18.	Digital Display Tank	Objek ini bagian dari Tank	Default
19.	Tank	Palet Controls\Modern\Numeric	11 x 31
20.	Very Smooth Tank 1	Image Navigator\Tanks	28 x 35
21.	90^0 curve 4	Image Navigator\Pipes	4,5 x 4,5
22.	Tombol Stop	Create Control pada Loop Condition	6 x 3

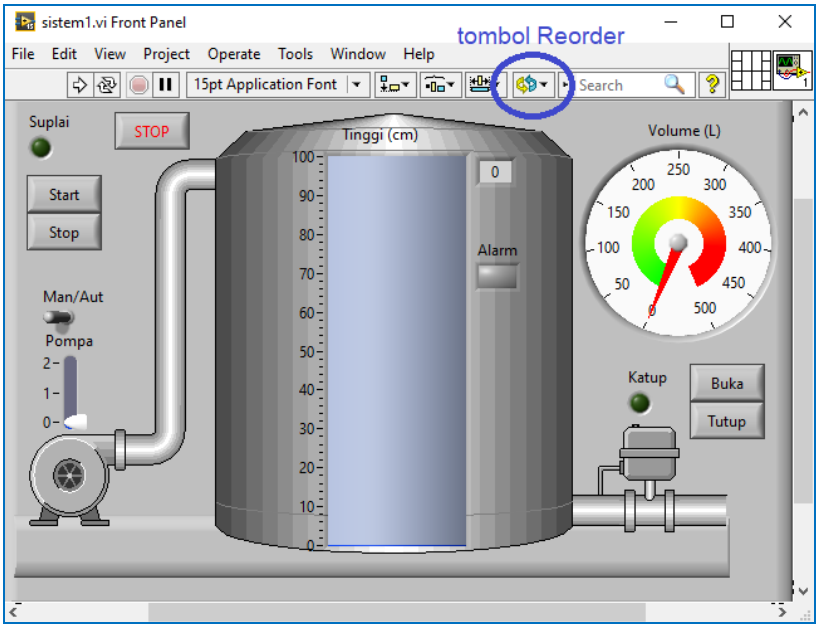
Catatan: Ukuran didasarkan pada kotak-kotak atau grid di halaman Front Panel. Ukuran 6 x 3 berarti lebar (horisontal) 6 kotak dan tinggi (vertikal) 3 kotak. Ukuran Default berarti ukuran tidak diubah.

Langkah-langkah pembuatan Tampilan Pengisian Tangki:

1. Buka sebuah vi yang baru dengan memilih New VI di menu File.
2. Ambil semua objek sesuai tabel daftar objek di atas, dan tempatkan di Front Panel. Untuk objek yang diambil dari Image Navigator, pengambilan dilakukan dengan memilih objek, kemudian pada kotak Preview, klik kanan gambar objek, pilih Copy as Picture (wmf) Only. Kemudian di Front Panel, tekan tombol Control dan P untuk menempelkan gambar objek.

Catatan: Pilih Copy as Picture (wmf) Only, agar gambar latar belakang objek menjadi transparan atau tidak terlihat.

3. Untuk objek no. 18, Digital Display Tank, objek ini dimunculkan dengan cara meng-klik kanan objek Tank, pilih Visible Items, pilih Digital Display.
4. Untuk Objek no. 16, Gauge, agar muncul warna gradasi, klik kanan pada objek, pilih Properties. Pada jendela Properties, buka Tab Scale, dan kemudian centang pada Show color ramp dan Interpolate Color.
5. Atur posisi semua objek hingga sesuai dengan Gambar 1.16 di atas.
6. Secara default, objek no. 6 (Vertical Pointer Slide), objek no. 16 (Gauge) dan objek no. 19 (Tank), memiliki nilai maksimum 10. Untuk mengubahnya, klik 2 kali pada angka maksimum tersebut, ganti angkanya sesuai dengan nilai maksimum yang diinginkan. Buat objek Vertical Pointer Slide memiliki nilai maksimum 2, objek Gauge memiliki nilai maksimum 500, dan objek Tank memiliki nilai maksimum 100.
7. Tambahkan objek Decoration Horizontal Smooth Box, lebarkan hingga semua objek terlingkupi. Agar objek Decoration ini bisa berada di belakang, tekan tombol Reorder di Toolbar, dan pilih Move To Back.



Gambar 1.17 Penambahan Horizontal Smooth Box di belakang Tampilan Pengisian Tangki

8. Sampai di sini pembuatan Tampilan Pengisian Tangki selesai.

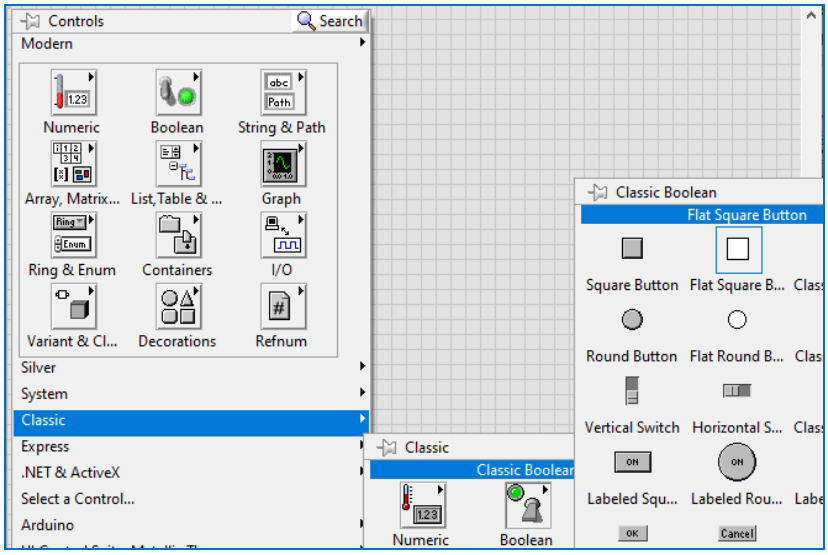
1.3.2 Pembuatan Objek Animasi

Animasi tampilan yang mudah dibuat adalah dengan mengubah warna objek. Diinginkan agar saat pompa (*Cool Pump*) bekerja mengisi tangki, warna pompa dan pipa berubah warna, untuk menunjukkan terjadi proses pengisian. Begitu pula ketika katup dibuka, diinginkan warna katup dan pipa di kanan tangki berubah warna, untuk menunjukkan terjadi proses pengosongan.

Sebaliknya ketika tidak ada pengisian maupun pengosongan, warna objek-objek tersebut kembali ke semula. Untuk bisa mengubah warna objek di LabVIEW, dari warna A ke warna B, dan sebaliknya, dapat dilakukan dengan meng-"*customize*" sebuah objek tipe Boolean.

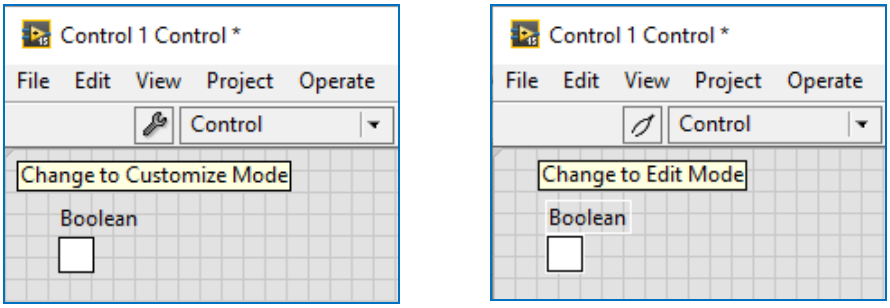
Berikut ini langkah-langkah untuk membuat warna objek pompa, pipa dan katup dapat berubah warna ketika diberi nilai True, dan kembali ke warna semula ketika diberi nilai False:

1. Buka sebuah vi baru. Jangan lupa simpan vi yang sebelumnya telah dibuat, beri nama yang unik, misalnya Tampilan Pengisian Tangki.
2. Di vi yang baru, di Front Panel, ambil objek Flat Square Button, dari palet Controls, di kategori Classic, di Boolean, dan tempatkan di Front Panel.



Gambar 1.18 Mengambil objek Flat Square Button dari palet Controls, Classic, Boolean

3. Tekan atau klik objek Flat Square Button tersebut, perhatikan bahwa ketika di-klik, objek tersebut berubah warna dari putih menjadi hitam, dan sebaliknya dari hitam ke putih ketika di-klik berikutnya. Dengan meng-"customize" bentuk objek Flat Square Button menjadi objek Pompa yang berwarna "original" saat warnanya putih, dan saat warnanya hitam diubah menjadi objek Pompa dengan warna "shaded blue", maka animasi perubahan warna objek Pompa bisa dibuat.
4. Klik kanan pada objek Flat Square Button, pilih Advanced, pilih Customize, maka muncul jendela Control. Secara default, jendela Control berada pada mode Edit. Mode Edit hanya bisa digunakan untuk mengubah kondisi True menjadi False atau sebaliknya, tetapi tidak bisa untuk mengubah bentuk. Agar objek kotak putih bisa berubah bentuk menjadi objek Pompa, mode Edit harus diubah menjadi mode Customize. Untuk itu klik tombol bergambar kunci pas, agar dapat berpindah ke mode Customize. Tanda bahwa mode Customize sudah aktif adalah gambar kunci pas pada tombol berubah menjadi gambar tang penjepit.



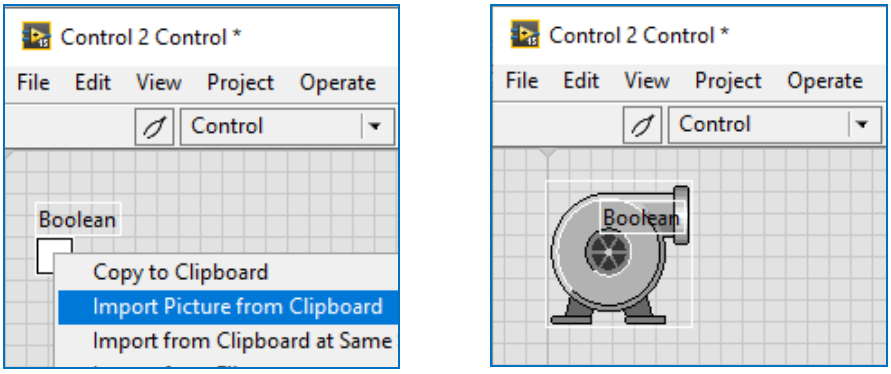
(a) (b)

Gambar 1.19 (a) Gambar kunci pas: Mode Edit, (b) Gambar tang penjepit: Mode Customize

5. Setelah Mode Customize aktif, seperti Gambar 1.19(b), klik kanan pada objek Boolean, pilih Import Picture from Clipboard. Hanya saja, pilihan Import Picture from Clipboard tersebut masih *disable* (abu-abu), karena belum ada gambar yang di-copy di Clipboard. Diinginkan gambar objek Pompa (Cool Pump) yang di-import. Untuk itu, buka vi yang lama, pada gambar Pompa, pilih gambar tersebut, kemudian tekan Control dan C

untuk meng-copy gambar. Setelah itu, kembali ke jendela Control, klik kanan objek Boolean, pilih Import Picture from Clipboard, maka gambar kotak putih berubah menjadi gambar Pompa (lihat Gambar 1.20).

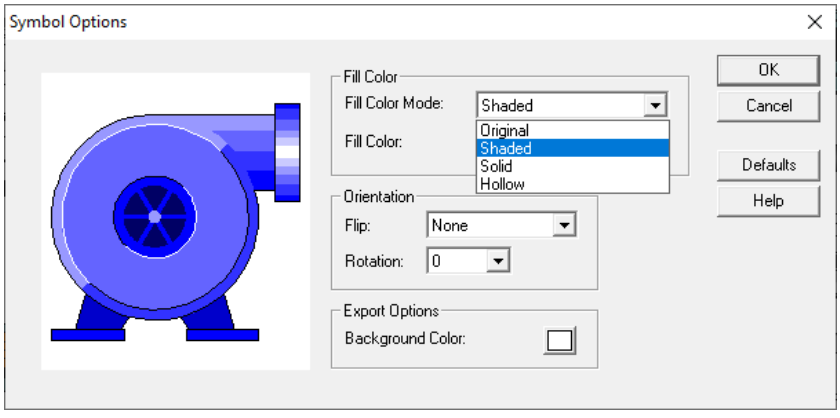
6. Setelah berhasil mengubah bentuk kotak putih menjadi gambar Pompa, berikutnya diinginkan mengubah bentuk kotak hitam menjadi gambar Pompa dengan warna biru, untuk menandakan Pompa sedang aktif.



(a) (b)

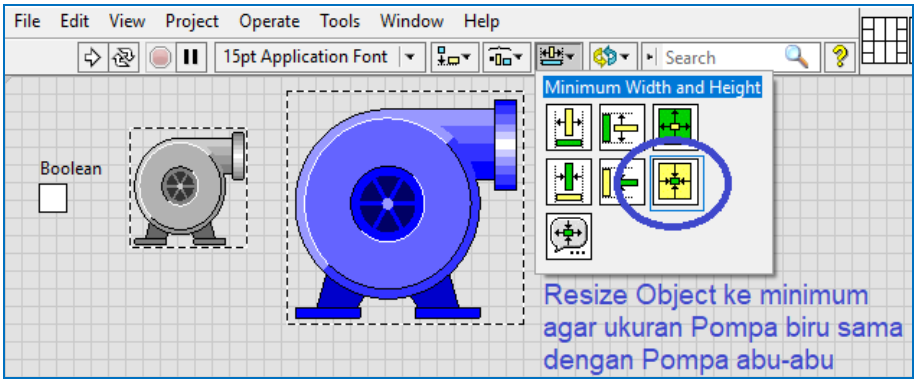
Gambar 1.20 (a) Import Picture from Clipboard, (b) gambar kotak berubah menjadi pompa

7. Pada jendela Image Navigator, pilih Cool Pump, yang ada di kategori Pumps. Klik kanan pada gambar di kotak Preview, pilih Symbol Options. Pada jendela Symbol Options, di kolom Fill Color Mode, pilih Shaded.



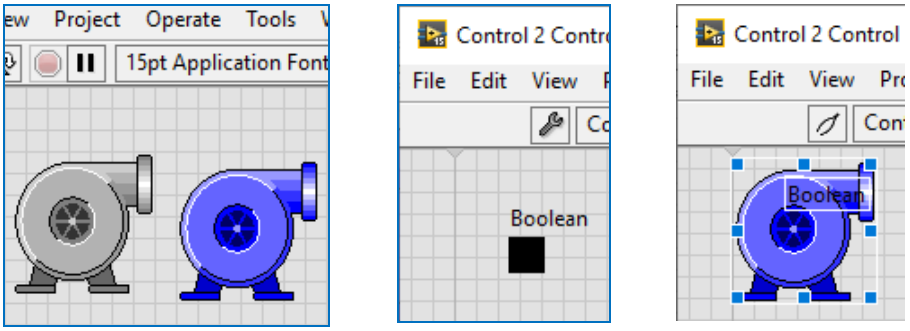
Gambar 1.21 Mengatur Fill Color Mode pada objek Pompa dari Original menjadi Shaded

8. Klik OK, muncul gambar Pompa biru di kotak Preview. Klik kanan gambar Pompa biru tersebut, pilih Copy as Picture (wmf) Only. Tempelkan gambar Pompa biru ini di Front Panel dengan menekan tombol Control dan V.
9. Karena ukuran Pompa biru ini lebih besar dari Pompa abu-abu, maka agar ukurannya sama, gunakan tool Resize Object, yang ada di Toolbar. Pilih kedua Pompa, buka Tool Resize Objects, pilih Minimum Widht and Height, maka ukuran gambar Pompa biru dan abu-abu menjadi sama-sama kecil.



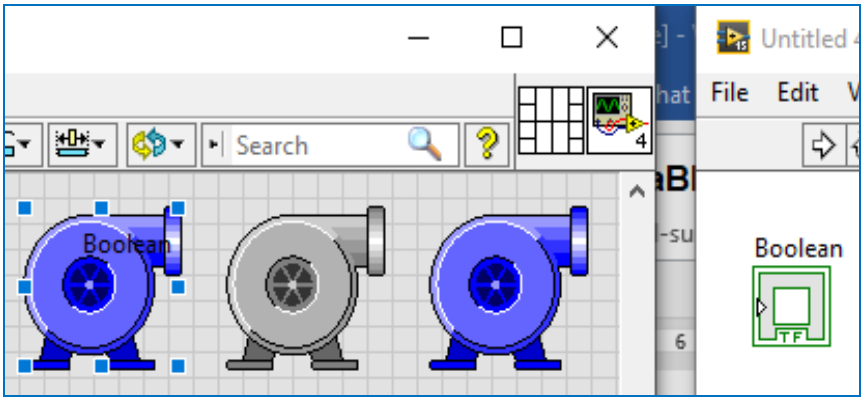
Gambar 1.22 Pilih kedua Pompa, pilih Minimum Widht and Height di Tool Resize Objects

10. Di jendela Control, ubah mode Customize ke mode Edit. Di mode Edit, klik gambar objek agar gambar False menjadi gambar True. Gambar False adalah gambar Pompa, dan gambar True adalah gambar kotak hitam.
11. Setelah gambar kotak hitam terlihat, ubah mode Edit menjadi mode Customize. Kemudian copy gambar Pompa biru. Setelah itu, klik kanan gambar kotak hitam, pilih Import Picture from Clipboard, maka gambar kotak hitam berubah menjadi gambar Pompa biru.
12. Tutup jendela Control, pilih Yes pada pertanyaan Replace …?, kemudian tekan tombol Save, dan beri nama Pompa. Animasi Pompa telah berhasil dibuat. Objek Boolean Flat Square Button yang semula berbentuk kotak putih, sekarang menjadi Pompa abu-abu, dan ketika objek tersebut di-klik, yang semula kotak hitam, sekarang menjadi Pompa biru. Dengan memberi nilai True pada icon Boolean akan membuat Pompa berwarna biru, dan ketika diberi nilai False, akan membuat Pompa berwarna abu-abu.



(a) (b) (c)

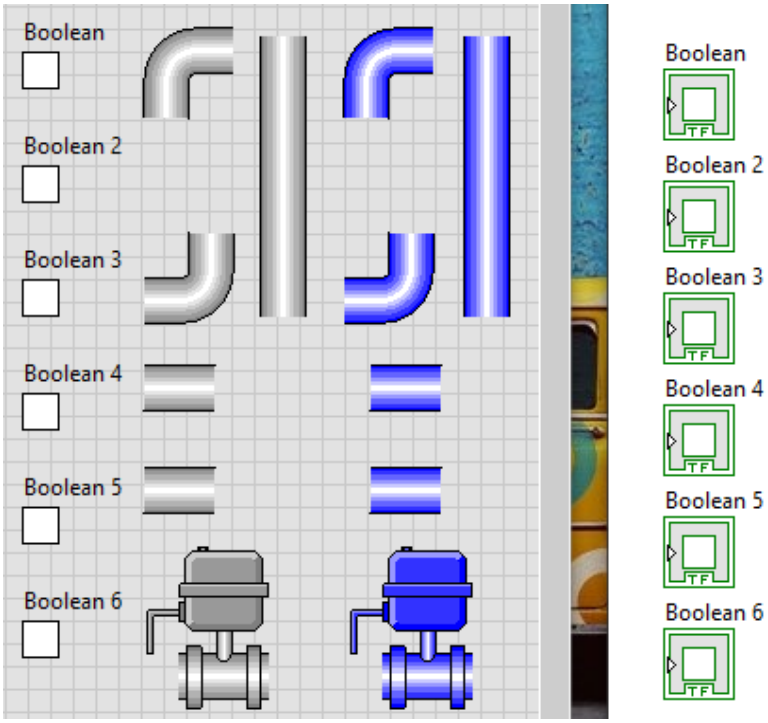
Gambar 1.23 (a) ukuran Pompa biru dan abu-abu sama, (b) di menu Edit, ubah gambar ke gambar kotak hitam, (c) di menu Customize, ubah gambar kotak hitam menjadi Pompa biru



Gambar 1.24 Ketika objek Boolean di-klik, maka gambar objek berubah menjadi Pompa biru, ketika di-klik lagi, maka gambar objek berubah menjadi Pompa abu-abu

13. Setelah berhasil membuat animasi perubahan warna pada objek Pompa, lanjutkan dengan cara yang sama, untuk membuat animasi pada objek Pipa dan Katup. Ada 5 buah objek Pipa, yang terdiri dari sebuah pipa vertikal, 2 buah pipa siku dan 2 buah pipa horisontal. Untuk membuat semua objek tersebut dapat berubah warna, perlu 6 buah Boolean Flat Square Button. Tempatkan 6 buah Flat Square Button di Front Panel.
14. Berikutnya, tempatkan keenam gambar objek yang akan dibuat animasinya di samping 6 buah objek Flat Square Button. Ambil gambar keenam objek tersebut dengan meng-copy dari gambar yang sudah dibuat di vi Tampilan Pengisian Tangki.

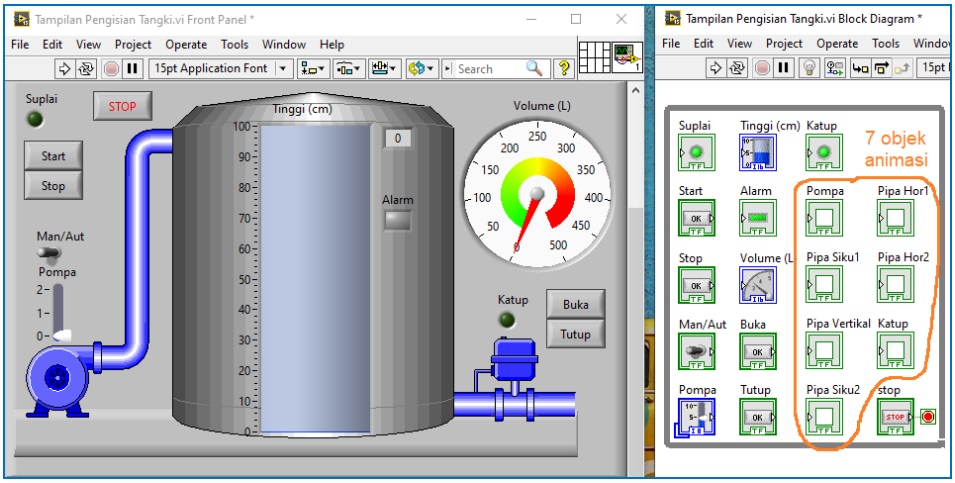
15. Di Image Navigator, cari gambar keenam objek tersebut satu persatu. Gunakan Symbol Options, dan pilih Shaded pada kolom Fill Color Mode untuk mendapatkan gambar biru dari keenam objek tersebut.
16. Gunakan Copy as Picture (wmf) Only, agar gambar background keenam objek tersebut tidak terlihat/transparan. Tempatkan gambar biru keenam objek-objek tersebut di samping gambarnya masing-masing, yang di-copy dari vi Tampilan Pengisian Tangki.
17. Buat ukuran gambar biru dari keenam objek tersebut berukuran sama dengan gambarnya masing-masing, gunakan Tool Resize Objects di Toolbar, dan pilih Minimum Width and Height.



Gambar 1.25 Enam Flat Square Button dengan gambar keenam objek warna abu-abu dan biru

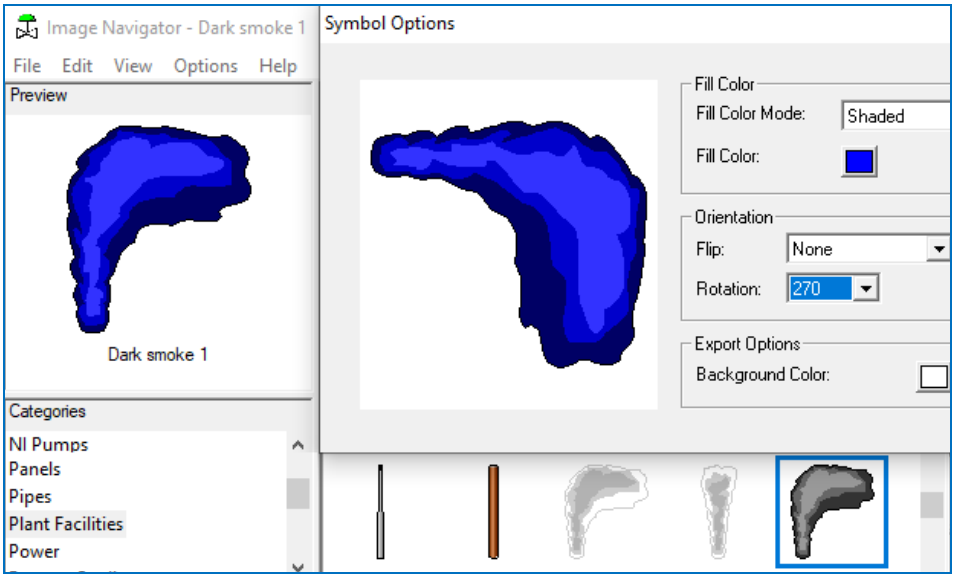
18. Satu persatu, buat keenam objek Flat Square Button berubah bentuk menjadi gambar di sampingnya, yaitu ketika False menjadi gambar berwarna abu-abu, dan ketika True menjadi gambar berwarna biru. Klik kanan objek, pilih Advance, pilih Customize.

19. Di mode Customize, pilih Import Picture from Clipboard, namun sebelum itu, copy gambar yang akan di-Import. Gunakan mode Edit dan klik pada gambar, untuk mengubah gambar False menjadi True, atau sebaliknya.
20. Setelah semua objek Boolean Flat Square Button telah diubah bentuknya, buka kembali vi Tampilan Pengisian Tangki. Ganti gambar Pompa, Pipa dan Katup dengan ketujuh objek Boolean Flat Square Button yang telah berubah bentuk. Tempatkan objek sesuai dengan posisi gambarnya.



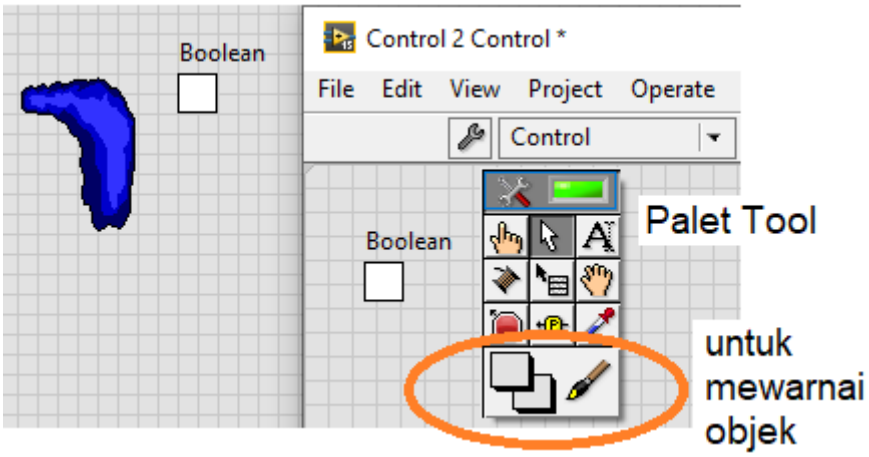
Gambar 1.26 Menempatkan ketujuh objek Boolean Flat Square Button sesuai posisi gambar

21. Berikutnya, agar saat pengosongan Tangki bisa keluar air dari pipa, seperti Gambar 1.1, tambahkan gambar Dark Smoke1, yang diambil dari Image Navigator, di kategori Plant Facilities. Klik kanan pada gambar Dark Smoke1 di kotak Preview, pilih Symbol Options. Di kolom Fill Color Mode, pilih Shaded, dan pada kolom Rotation di Orientation, pilih 270^0. Klik OK, kemudian klik kanan lagi pada gambar, pilih Copy as Picture (wmf) Only. Tempelkan gambar tersebut di Front Panel dengan menekan tombol Control dan V. Perkecil ukurannya menjadi 6 x 8 kotak.
22. Berikutnya tambahkan objek Boolean Flat Square Button untuk membuat gambar air tersebut bisa dianimasikan. Diinginkan ketika bernilai False, gambar air menghilang (diberi warna transparan), dan ketika bernilai True, gambar air muncul.



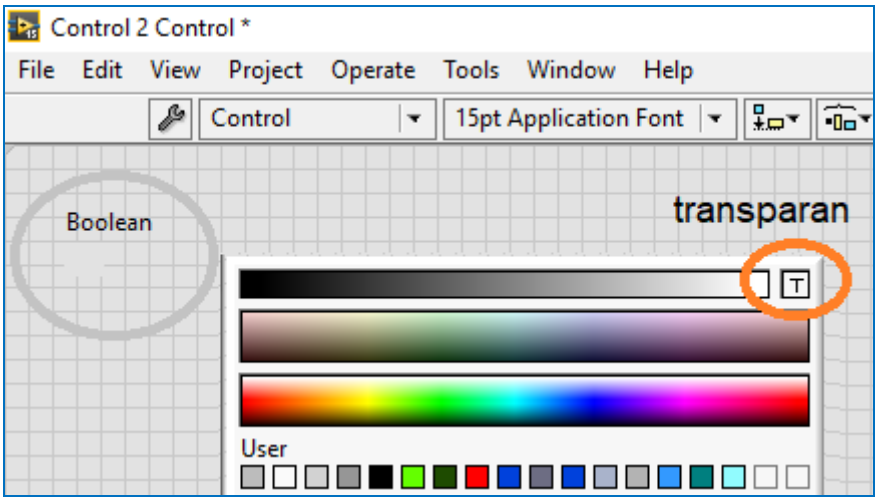
Gambar 1.27 Animasi air dibuat dengan gambar Dark Smoke1, Shaded biru dan diputar 270^0

23. Klik kanan pada objek Boolean Flat Square Button, pilih Advanced, pilih Customize. Pada jendela Control yang muncul, di mode Edit, munculkan Tool Palet untuk mengubah warna objek Flat Square Button menjadi transparan. Memunculkan Tool Palet dengan cara meng-klik kanan halaman jendela Control, sembari menekan tombol Shift.



Gambar 1.28 Di mode Edit, di gambar False (kotak putih), warnai Boolean dengan transparan

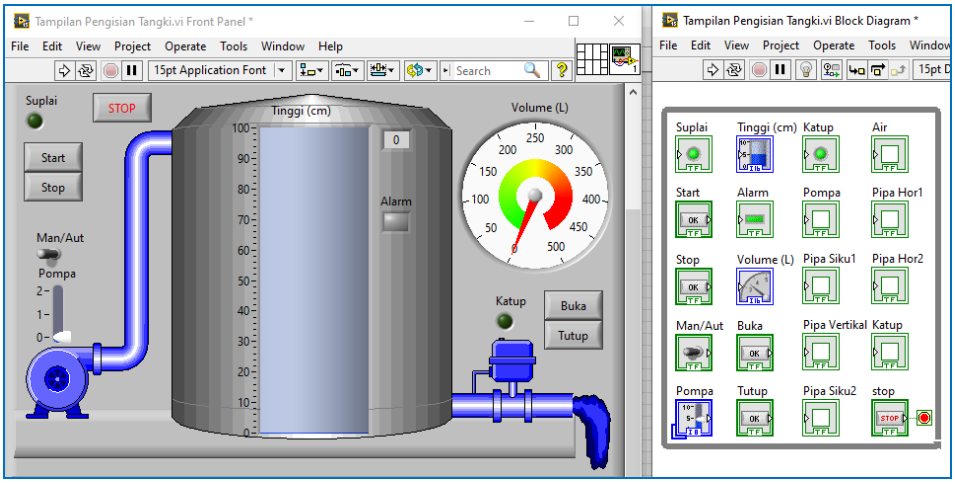
24. Setelah Tool Palet muncul, berikutnya pilih gambar kuas, kemudian sentuhkan kuas pada objek Boolean, kemudian klik kanan hingga muncul kotak pilihan warna. Pada kotak pilihan warna yang muncul, klik pada tulisan T yang berarti Transparan, maka gambar kotak putih pada objek Boolean akan menghilang, karena menjadi transparan.



Gambar 1.29 Sentuhkan kuas ke objek, kotak warna muncul, klik T, objek menjadi transparan

25. Agar pointer mouse yang berbentuk kuas bisa kembali ke mode otomatis, buka lagi Tool Palet dengan meng-klik kanan sambil menekan tombol Shift. Di Tool Palet, klik gambar LED di bagian atas agar menyala kembali, maka mode otomatis pointer mouse kembali aktif.
26. Berikutnya, klik pada objek Boolean yang sudah transparan, agar berubah menjadi gambar True yang berbentuk kotak hitam. Setelah gambar kotak hitam muncul, klik pada tombol kunci pas agar berpindah dari mode Edit ke mode Customize.
27. Di mode Customize, klik kanan gambar kotak hitam, pilih Import Picture from Clipboard, namun sebelum itu, copy dulu gambar Dark Smoke1. Setelah itu klik Import, maka gambar kotak hitam akan berubah menjadi gambar Dark Smoke1.
28. Tutup jendela Control, pilih Yes pada pertanyaan Replace …?, kemudian tekan tombol Save, dan beri nama Air.

29. Animasi air telah berhasil dibuat. Ketika objek ini diberi nilai True, muncul gambar air, ketika diberi nilai False, gambar air menghilang. Tempatkan objek bergambar air ini di ujung pipa pengosongan.



Gambar 1.30 Objek Flat Square Button dengan gambar Air ditempatkan di ujung pipa

30. Sampai di sini pembuatan objek animasi selesai.

1.3.3 Pembuatan Program Pengisian Tangki

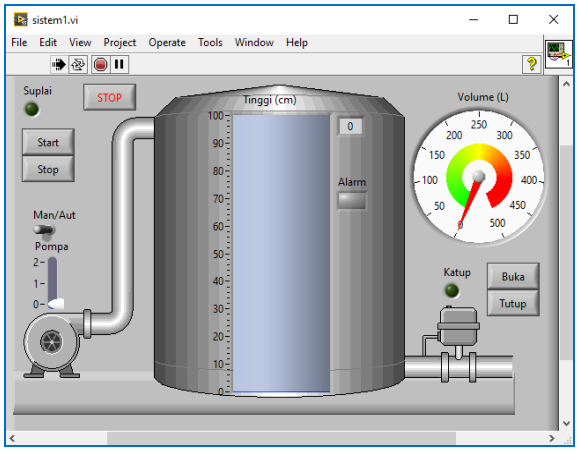
Perhatikan Blok Diagram pada Gambar 1.30 di atas. Ada 20 buah icon yang harus dihubungkan (diprogram) untuk menghasilkan proses pengisian dan pengosongan Tangki. Sebelum membuat program tersebut, perlu kiranya memahami dulu ketentuan dan gambaran Proses Pengisian dan Pengosongan Tangki ini.

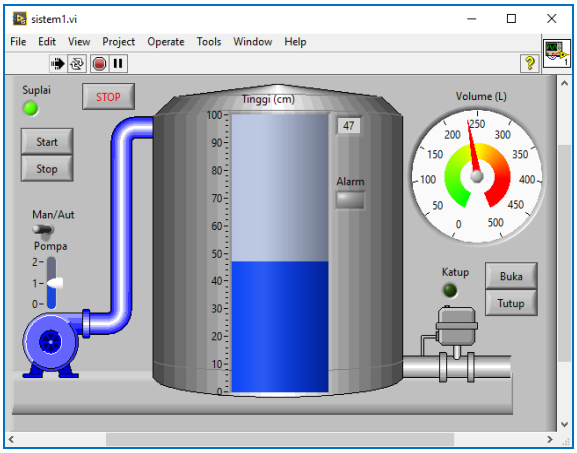
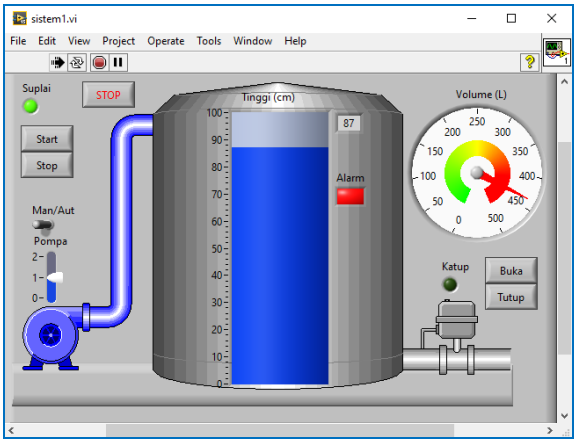
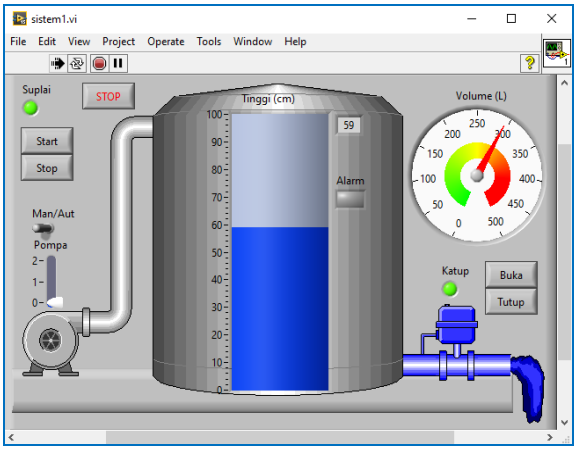
Berikut ini ketentuan proses Pengisian dan Pengosongan Tangki:

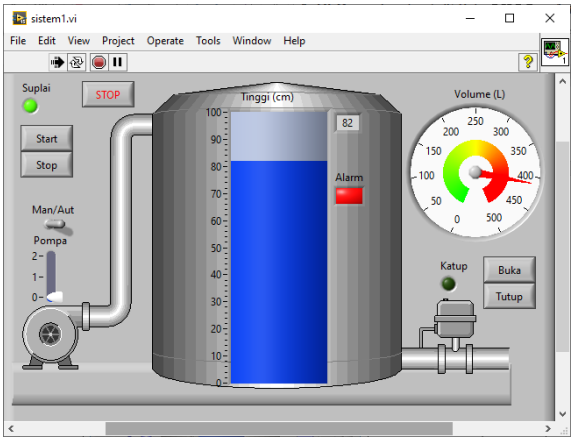
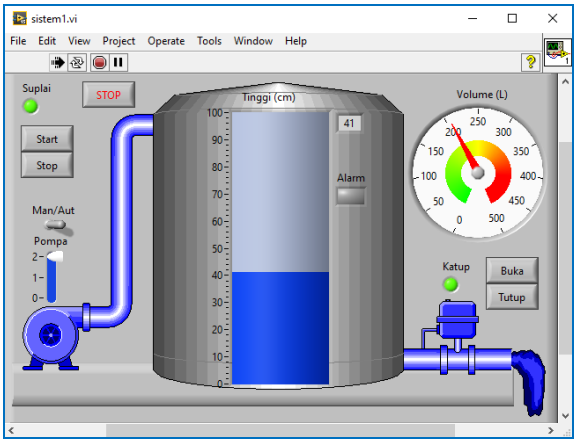
1. Semua proses pada pengisian dan pengosongan Tangki tidak akan berjalan apabila indikator LED Suplai tidak menyala.
2. Indikator LED Suplai menyala ketika tombol Start ditekan, dan mati ketika tombol Stop ditekan.
3. Tinggi tangki: 100 cm, Luas penampang: 5000 cm^2, Volume Tangki: 500 L.
4. Pengisian Tangki dilakukan dengan menghidupkan Pompa Air. Pompa Air ini dapat dihidupkan dengan mode Manual (MAN) atau Otomatis (AUT).

5. Apabila mode Manual dipilih, pengguna harus menggerakkan Slider. Bila Slider digerakkan ke angka 0, Pompa akan mati. Bila Slider digerakkan ke angka 1, Pompa akan mengisi Tangki dengan debit air 50 L air/detik. Bila Slider digerakkan ke angka 2, Pompa akan mengisi Tangki dengan debit pengisian dua kali lipat, yaitu 100 L air/detik.
6. Apabila mode Otomatis dipilih, maka Pompa akan secara otomatis hidup ketika tinggi air dalam Tangki kurang dari 20 cm, dan otomatis mati ketika tinggi air lebih dari 80 cm. Debit pengisian air dalam mode otomatis adalah debit maksimum, yaitu 100 L air/detik.
7. Agar Tangki tidak rusak karena pengisian air yang melebihi kapasitasnya, sebuah Alarm ditambahkan, yang akan aktif ketika tinggi air dalam Tangki melebihi 80 cm, dan mematikan pompa, bila tinggi air mencapai 100 cm.
8. Debit pengosongan Tangki sebesar 50 L air/detik.
9. Pengosongan Tangki dilakukan dengan membuka Katup. Katup hanya dapat dibuka apabila LED Suplai menyala, dan tombol Buka ditekan. Katup tertutup ketika LED Suplai padam, atau ketika tombol Tutup ditekan.
10. Indikator LED Katup akan menyala apabila dalam kondisi terbuka, dan mati ketika Katup dalam kondisi tertutup.

Berikut ini gambaran Pengisian dan Pengosongan Tangki pada tampilan program:

Tampilan Program:	Keterangan
	1. Saat program dijalankan pertama kali, proses pengisian dan pengosongan tidak berjalan, menunggu LED Suplai menyala. 2. Ketika tombol Start ditekan, LED Suplai menyala.

Tampilan Program:	Keterangan
	3. Ketika mode Manual dipilih, dan Slider digerakkan ke angka 1, Pompa mengisi Tangki dengan debit 50 L/detik. Bila Slider digerakkan ke angka 2, Pompa mengisi Tangki dengan debit 100 L/detik.
	4. Ketika tinggi air melebihi 80 cm, Alarm aktif, Lampu Alarm berkedip. 5. Ketika tinggi air mencapai 100 cm, Pompa mati.
	6. Ketika tombol Buka ditekan, Katup terbuka, LED Katup menyala, pengosongan air terjadi, dengan debit 50 L/detik.

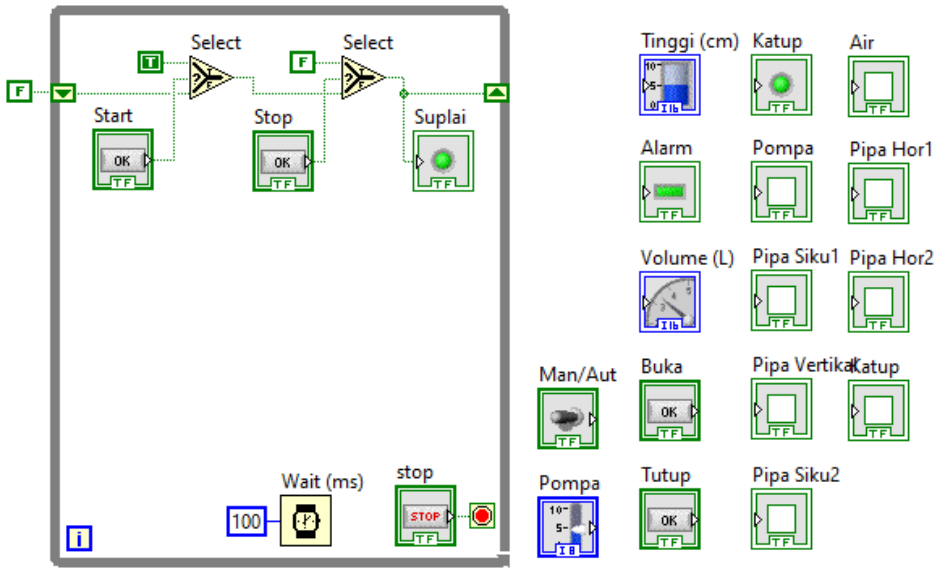
Tampilan Program:	Keterangan
	7. Ketika mode Otomatis dipilih, apabila tinggi air kurang dari 80 cm, Pompa akan mengisi Tangki dengan debit 100 L/detik (Slider posisi 2) 8. Begitu tinggi air lebih dari 80 cm, Pompa berhenti (Slider posisi 0)
	9. Ketika Katup dibuka, terjadi pengosongan Tangki dengan debit 50 L air/detik. 10. Ketika tinggi air kurang dari 20 cm, Pompa otomatis hidup kembali, mengisi Tangki dengan debit 100 L/detik.

Dari gambaran Proses Pengisian dan Pengosongan Tangki di atas, berikut ini tahapan pembuatan program untuk memenuhi gambaran proses tersebut:

1. Membuat program Set Reset LED Suplai.
2. Membuat program Set Reset LED Katup.
3. Membuat program Pengisian Tangki oleh Pompa.
4. Membuat program Pengosongan Tangki oleh Katup.
5. Membuat program Otomatis Pompa.
6. Membuat program Alarm Berkedip.
7. Membuat program Animasi

1.3.3a Program Set Reset LED Suplai

Program Set Reset LED Suplai ini sama seperti program Set Reset pada Gambar 1.13, di mana tombol Start sebagai Set dan tombol Stop sebagai Reset. Berikut ini program untuk Set Reset LED Suplai.



Gambar 1.31 Program Set Reset LED Suplai yang sama seperti program pada Gambar 1.13

Jalankan program di atas dengan menekan tombol Run (jangan menekan tombol Run Continuously, karena akan membuat perulangan di dalam perulangan). Setelah program berjalan, tekan tombol Start, maka LED Suplai menyala, dan terus menyala hingga tombol Stop ditekan.

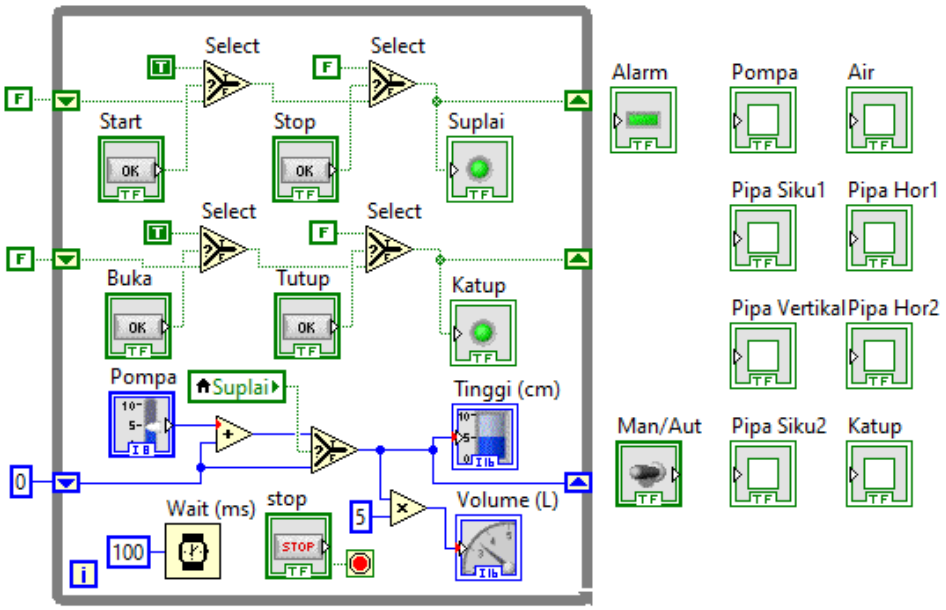
1.3.3b Program Set Reset LED Katup

Program Set Reset LED Katup ini sama seperti Program Set Reset LED Suplai di atas, di mana tombol Buka sebagai tombol Set dan tombol Tutup sebagai Reset, seperti ditunjukkan pada Gambar 1.32.

1.3.3c Program Pengisian Tangki oleh Pompa

Sama seperti program Set Reset yang memerlukan Shift Register dan icon Select, program untuk Pengisian Tangki ini juga memerlukan Shift Register dan Select.

Hanya bedanya, data di Shift Register untuk program Set Reset bertipe Boolean, sedangkan data di Shift Register untuk program Pengisian Tangki bertipe Integer. Lebih jelasnya lihat gambar Block Diagram berikut ini:



Gambar 1.32 Program Set Reset LED Suplai, Set Reset LED Katup dan Pengisian Tangki

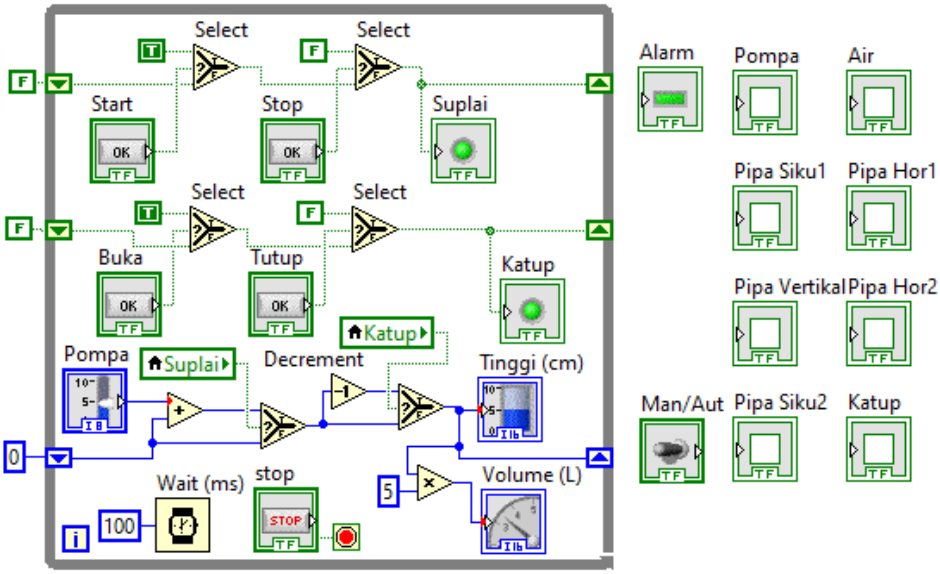
Gambar 1.32 di atas, dari atas ke bawah, mengikuti urutan Shift Register, secara berturut-turut menunjukkan program Set Reset LED Suplai, program Set Reset LED Katup dan program Pengisian Tangki. Ketiga program tersebut memerlukan Shift Register untuk meneruskan data dari satu loop ke loop berikutnya. Ketiga program tersebut juga memerlukan icon Select, untuk meneruskan data apabila mendapat input False, dan meng-"update" data apabila inputnya True.

Untuk program pengisian Tangki, icon Select akan meng-"update" data Shift Register dengan menambahkan nilai yang ada dengan nilai Slider apabila LED Suplai menyala. Apabila LED Suplai menyala dan Slider digerakkan ke angka 1, Tinggi air dalam Tangki akan bertambah 1 cm setiap 100 milidetik (=10 cm/detik), dan Volume air dalam Tangki bertambah 50 L/detik. Apabila LED Suplai menyala dan Slider digerakkan ke angka 2, Tinggi air dalam Tangki akan bertambah 2 cm setiap 100 milidetik (=20 cm/detik) dan Volume air Tangki bertambah 100 L/detik.

Catatan: Progam Gambar 1.32 di atas menggunakan Local Variable LED Suplai untuk memberi input pada icon Select Shift Register Pengisian Tangki. Local Variable LED Suplai ini bisa diganti dengan menghubungkan garis langsung dari kaki input LED Suplai ke kaki input icon Select.

1.3.3d Program Pengosongan Tangki oleh Katup

Pengosongan Tangki bekerja berlawanan dengan Pengisian Tangki. Apabila Pengisian Tangki menambah Tinggi dan Volume air Tangki, Pengosongan Tangki sebaliknya mengurangi Tinggi dan Volume air Tangki. Pengosongan Tangki ini hanya terjadi ketika LED Katup menyala. Berikut ini program Pengosongan Tangki.



Gambar 1.33 Program Pengosongan Tangki: bila LED Katup ON, Tinggi air Tangki dikurangi 1

Program Pengosongan Tangki pada Gambar 1.33 sudah bisa bekerja seperti yang diharapkan, yaitu hanya mengurangi Tinggi air dalam Tangki ketika LED Katup menyala, dengan pengurangan tinggi air 1 cm setiap 100 milidetik. Namun demikian, program di atas menimbulkan 3 kejanggalan (“bug”), yaitu:

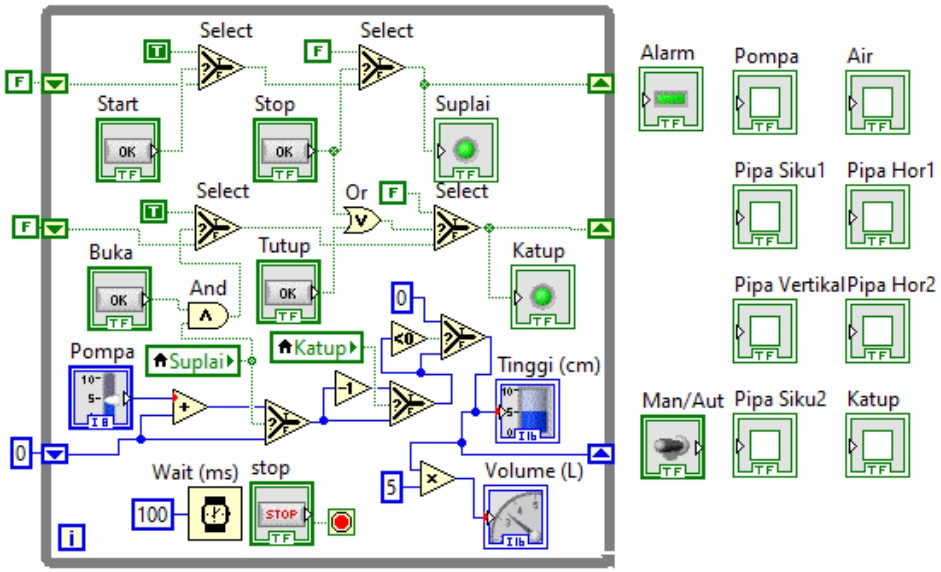
a. Kejanggalan yang pertama: Tinggi air Tangki bisa menjadi negatif apabila LED Katup masih menyala saat Tinggi air dalam Tangki sudah benilai 0.

b. Kejanggalan yang kedua: harusnya ketika tombol Buka ditekan, apabila LED Suplai belum menyala, LED Katup juga tidak menyala, karena tidak ada suplai listrik ke Katup.
c. Kejanggalan yang ketiga: harusnya ketika LED Suplai mati, Katup otomatis juga menutup, karena suplai listrik ke Katup berhenti.

Berikut ini alternatif perbaikannya:

a. Untuk mengatasi kejanggalan pertama, agar Tinggi air dalam Tangki tidak bisa negatif, tambahkan icon Select, dengan kaki input t diberi angka 0, sehingga bila inputnya negatif, akan mengeluarkan nilai 0.
b. Untuk mengatasi kejanggalan kedua, diatasi dengan menambah fungsi AND antara LED Suplai dengan tombol Buka, sehingga LED Katup hanya bisa menyala apabila LED Suplai sudah menyala dan tombol Buka ditekan.
c. Untuk mengatasi kejanggalan ketiga, diatasi dengan menambah fungsi OR antara tombol Tutup dengan tombol Stop, sehingga baik tombol Tutup maupun tombol Stop, keduanya bisa membuat LED Katup mati.

Berikut ini program perbaikan untuk mengatasi 3 kejanggalan di atas:

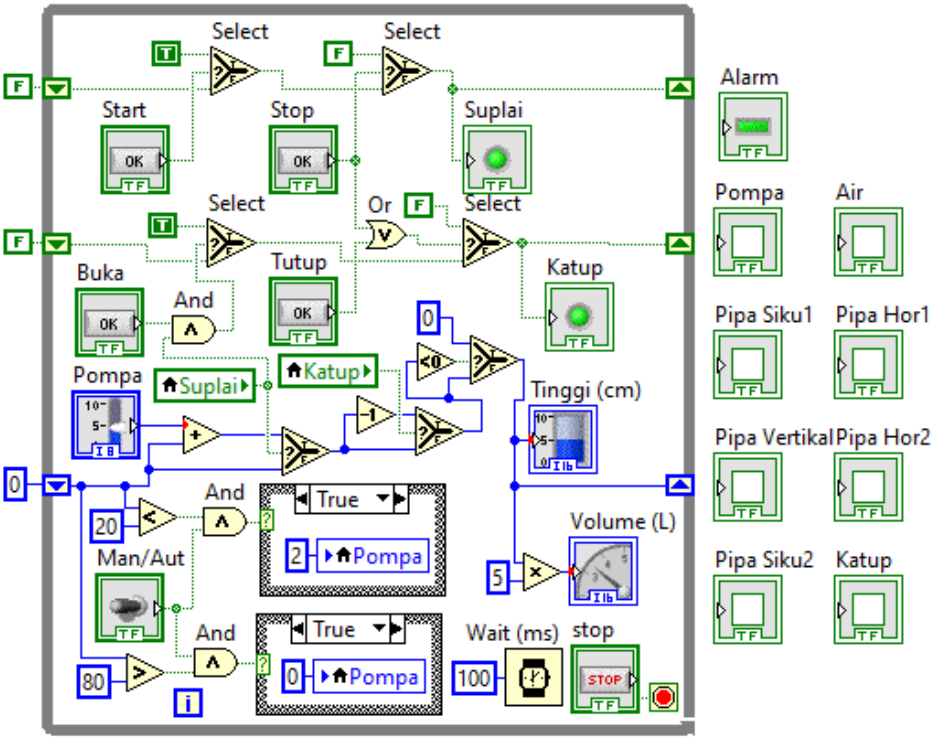


Gambar 1.34 Perbaikan: Tinggi air Tangki tidak pernah negatif, LED Katup hanya bisa dinyalakan apabila LED Suplai menyala, dan LED Katup akan mati bila LED Suplai mati

1.3.3e Program Otomatis Pompa

Ketika Pompa dihidupkan dalam mode Otomatis, diinginkan pengisian akan dilakukan secara otomatis ketika Tinggi air Tangki kurang dari 20 cm, dan otomatis mati ketika Tinggi air Tangki lebih dari 80 cm. Pengisian dilakukan dengan debit air 100 L/detik, atau Slider = 2, dengan kenaikan tinggi air 20 cm/detik,

Berikut ini program Otomatis Pompa:

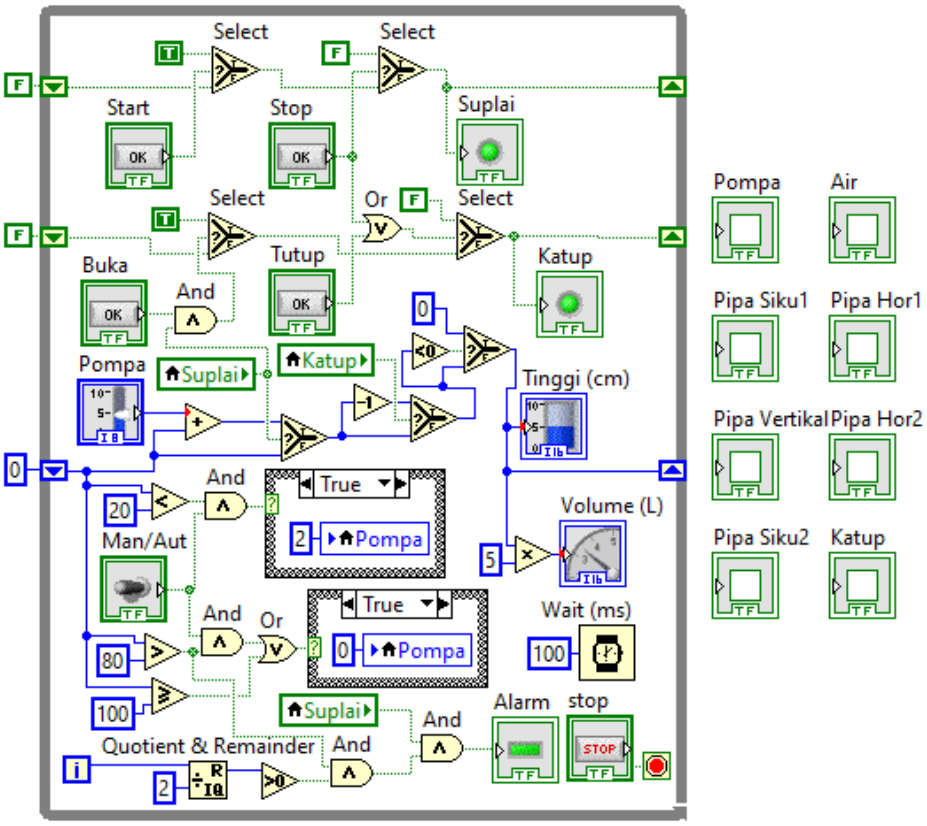


Gambar 1.35 Ketika mode Otomatis dipilih, dan Tinggi air < 20 cm, Pompa hidup, (Slider = 2), ketika mode Otomatis dipilih, dan Tinggi air > 80 cm, Pompa mati (Slider = 0)

Dalam Gambar 1.35 di atas, dua buah Struktur Case digunakan untuk menghidupkan Pompa (input 2), dan mematikan Pompa (input 0). Kotak Case False untuk kedua Struktur Case tersebut tidak ada isinya, alias kosong. Itulah sebabnya mengapa dalam program Otomatis Pompa ini digunakan Struktur Case, bukan Select, karena dengan Struktur Case, tidak perlu semua kotak Case harus diisi, sedangkan dengan Select, semua nilai input harus diisi.

1.3.3f Program Alarm berkedip

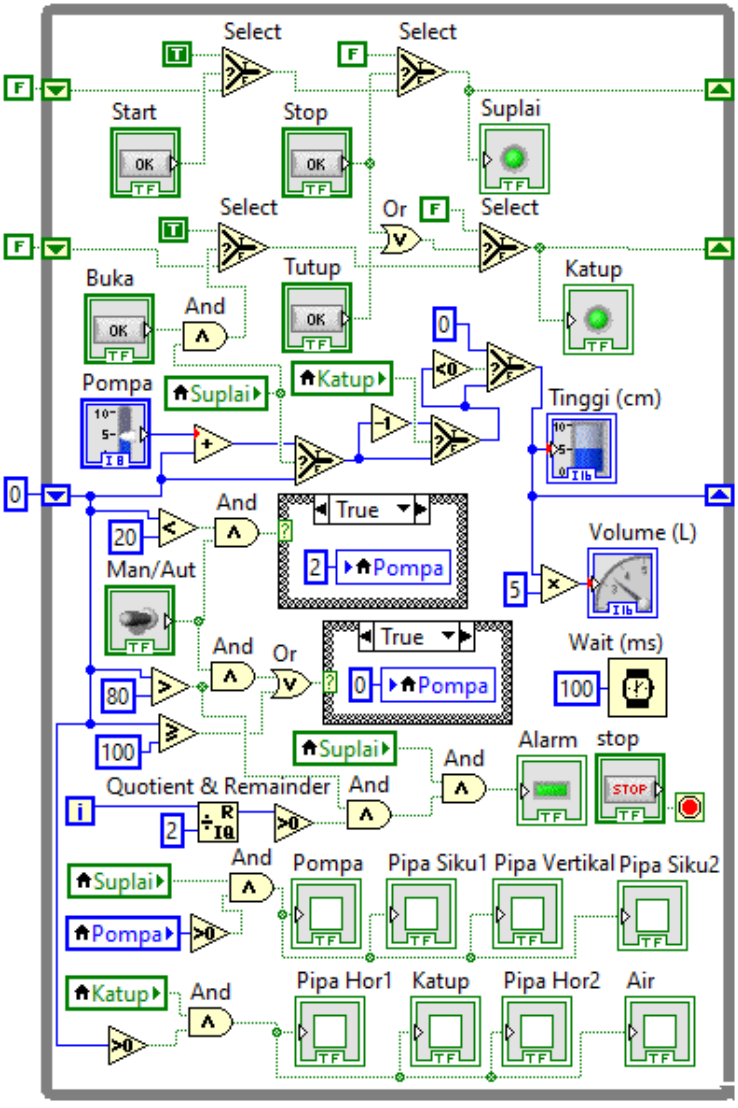
Berbeda dengan LED berkedip pada Gambar 1.11 yang menggunakan Struktur Flat Sequence, LED Alarm berkedip juga dapat dibuat dengan Loop Iteration (i) yang dikombinasikan dengan fungsi Quotient & Remainder. Kombinasi keduanya bisa menghasilkan nilai yang berulang. Untuk membuat LED Alarm berkedip dibutuhkan nilai 0, 1, 0, 1, … secara berulang. Untuk menghasilkan nilai tersebut, hubungkan input angka yang dibagi dengan Loop Iteration (i), dan angka pembagi diberi 2, maka output Remainder dari fungsi Quotient & Remainder akan menghasilkan nilai perulangan tersebut. Diinginkan LED Alarm berkedip ketika Tinggi air Tangki lebih dari 80 cm. Kemudian ketika Tinggi air Tangki mencapai 100 cm, diinginkan Pompa mati. Berikut ini program untuk membuat kedua hal itu:



Gambar 1.36 Ketika Tinggi air Tangki > 80 cm dan LED Suplai menyala, LED Alarm berkedip, ketika Tinggi air Tangki >= 100 cm maka Pompa mati

1.3.3g Program Animasi

Agar proses Pengisian dan Pengosongan Tangki di atas terlihat lebih baik, perlu animasi perubahan warna Pompa, Pipa, dan Katup, untuk menunjukkan adanya aliran air, dan juga gambar air keluar dari pipa saat Pengosongan Tangki. Berikut ini contoh program penambahan animasi.



Gambar 1.37 Ketika Pengisian: Pompa, Pipa Siku1, Pipa Vertikal, Pipa Siku2 menjadi biru, ketika Pengosongan: Pipa Hor1, Katup, Pipa Hor2 menjadi biru, dan Air muncul

Perhatikan program animasi Gambar 1.37 di atas. Objek Pompa, Pipa Siku1, Pipa Vertikal dan Pipa Siku2 akan berwarna biru untuk menunjukkan adanya aliran air, hanya ketika LED Suplai menyala dan nilai Slider Pompa lebih besar dari 0. Di luar kondisi itu, keempat objek tersebut akan berwarna abu-abu.

Objek Pipa Hor1, Katup, Pipa Hor2 akan berwarna biru, dan gambar Air akan muncul, ketika LED Katup menyala dan Tinggi air dalam Tangki lebih besar dari 0. Di luar kondisi itu, ketiga objek tersebut berwarna abu-abu, dan tidak muncul Air.

Sampai di sini pembuatan program Tampilan Pengisian Tangki sudah selesai.

1.4 Soal Latihan

1. Mengapa saat menjalankan program LabVIEW, tidak boleh menekan tombol Run Continuously ketika di dalam program menggunakan While Loop?
2. Mengapa ketika menggunakan Struktur perulangan While Loop, perlu ditambahkan instruksi delay seperti Wait(ms) pada program?
3. Apa kelebihan dan kelemahan Local Variable dibandingkan pengawatan secara langsung?
4. Struktur Case dan fungsi Select sama-sama digunakan untuk program percabangan. Namun demikian, Struktur Case memiliki kelebihan dan kelemahan dibandingkan fungsi Select. Sebutukan kelebihan dan kelemahan tersebut.
5. Apabila Luas penampang Tangki yang semula 5000 cm^3 diperbesar menjadi 10.000 cm^3, maka butuh waktu berapa lama Pengisian Tangki harus dilakukan, agar Tinggi air dalam Tangki mencapai 100 cm? Anggap bahwa Katup Pengosongan tidak dibuka, dan debit pengisian Pompa adalah 50 L/detik.
6. Tambahkan pada program sebuah LED Alarm yang akan aktif berkedip ketika Tinggi Air dalam Tangki kurang dari 20 cm.
7. Untuk menghemat suplai daya, modifikasi program sehingga saat tidak ada air dalam Tangki (Tinggi air = 0 cm), Katup otomatis menutup (LED Katup mati).

8. Tambahkan pada program sebuah Numeric Indicator yang menampilkan catatan waktu yang telah lewat, dalam satuan menit dan detik, yang dimulai dari saat tombol Start ditekan, dan berhenti ketika tombol Stop ditekan. Gunakan bantuan Loop Iteration (i), dikombinasikan dengan fungsi Quotient & Remainder, dengan angka pembagi 60, baik untuk menit maupun detik.

9. Tambahkan pada program sebuah Numeric Indicator yang menampilkan volume atau banyaknya air dalam Liter, yang telah dikeluarkan dari pipa pengosongan.

10. Diinginkan mengganti Slider Pompa dengan tiga buah tombol (OK Button), menggantikan Slider 0, 1 dan 2 secara berturut-turut. Modifikasi program sehingga penggantian Slider dengan 3 buah tombol tersebut tetap dapat menjalankan pompa seperti ketika menggunakan Slider. Karena ketiga tombol tersebut bersifat momentary (kembali ke posisi semula setelah ditekan), maka tambahkan 3 buah LED yang menandai tombol mana yang terakhir ditekan.

1.5 Refleksi

1. Menurut Anda, dari isi yang diuraikan, apakah **Target Materi** dari Bab 1 buku ini tercapai? Jika belum tercapai, apakah ada kesulitan dalam memahami materi yang berkaitan dengan Target Materi tersebut? Apakah Anda memiliki saran dan masukan untuk memperbaiki materi tersebut?

2. Apakah **Tantangan** yang diberikan dalam Bab 1 buku ini dapat Anda selesaikan? Jika belum, kesulitan apa yang membuat Anda tidak bisa menyeleaikan Tantangan tersebut? Apakah Anda sudah mencoba alternatif lain untuk menemukan sendiri solusinya (mencari di Google misalnya)?

3. Apakah ada **Manfaat** yang Anda dapatkan setelah mempelajari Bab 1 dari buku ini? Apakah ada yang ingin Anda pelajari lebih lanjut? Apakah ada **Ide** yang menarik yang Anda ingin kembangkan?

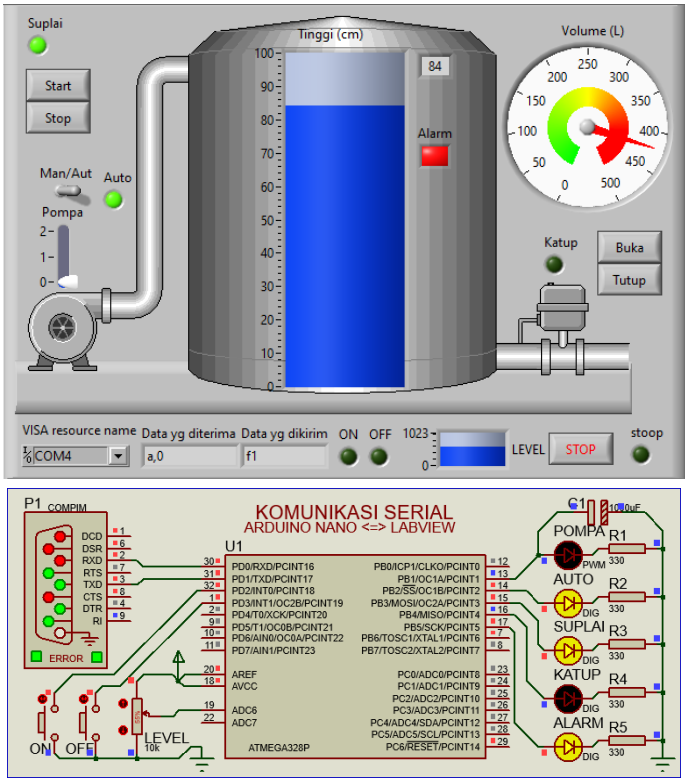
BAB 2
KOMUNIKASI SERIAL

Target Materi:

- Menerapkan komunikasi serial antara LabVIEW dengan Arduino
- Menggunakan Proteus untuk mensimulasikan rangkaian Arduino

Tantangan:

➢ Pembaca dapat menghubungkan Tampilan SCADA di LabVIEW dengan Arduino melalui komunikasi serial RS-232.



Gambar 2.1 Komunikasi Serial RS-232 antara Tampilan SCADA dengan Arduino

2.1 Persiapan Perangkat yang Diperlukan

Di samping menggunakan perangkat lunak di Bab 1, Bab 2 ini memerlukan tambahan perangkat lunak/software sebagai berikut:

1. Software Driver NI-VISA (*Hardware support*)
2. Software Arduino
3. Software Proteus
4. Virtual Serial Port Driver (VSPD)

Software Driver NI-VISA (*Virtual Instrument Software Architecture*) digunakan untuk menghubungkan LabVIEW dengan hardware, termasuk hardware yang menggunakan komunikasi serial seperti Arduino. Software Arduino digunakan untuk membuat program dan menanamkan program ke hardware Arduino. Software Proteus digunakan untuk mensimulasikan rangkaian Arduino. Software VSPD (*Virtual Serial Port Driver*) digunakan untuk membuat pasangan COM virtual. Keempat software di atas dapat pembaca unduh di halaman blog penulis: dianartanto@blogspot.com atau bisa menghubungi penulis langsung lewat email: dian.artanto@gmail.com. Setelah semua software di atas bisa diunduh, silahkan diinstal dan ikuti langkah-langkah di Sub Bab berikut.

2.2 Komunikasi Serial LabVIEW

Komunikasi serial adalah proses pengiriman data 1 bit per waktu secara berurutan (serial) melalui sebuah saluran komunikasi. Lawan dari komunikasi serial adalah komunikasi paralel, yang mengirimkan seluruh bit data dalam satu waktu sekaligus melalui beberapa saluran secara paralel. Komunikasi paralel secara hardware tidak banyak diterapkan karena pertimbangan biaya kabel yang mahal, sinkronisasi yang kompleks, dan instalasi yang lebih sulit. Komunikasi serial lebih dipilih, dan telah banyak diterapkan, contohnya pada komunikasi dengan USB, PS/2, Ethernet, MIDI, PCI Express, Profibus, RS-232, RS-422, RS-485, I2C, SPI, CAN, 1-Wire, dan lain-lain.

Secara umum komunikasi serial memiliki 3 saluran, yaitu TX, RX dan Ground. TX untuk mengirim data dan RX untuk menerima data. Ada 4 parameter penting pada komunikasi serial, yaitu baud rate, bit data, bit stop dan bit parity.

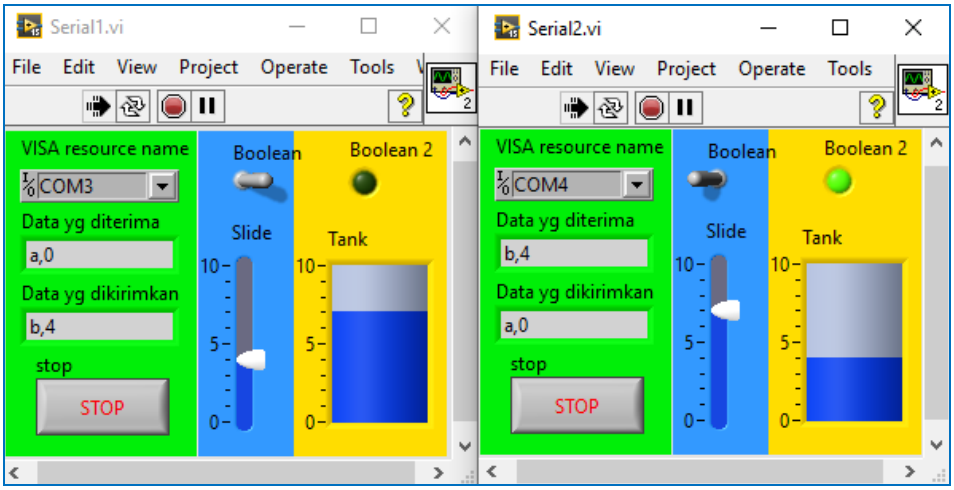
Baud rate merupakan kecepatan transfer data dalam bit per detik. Sebagai contoh, 9600 bps berarti 9600 bit per detik. Semakin tinggi baud rate komunikasi serial, semakin pendek jarak hantarnya. Untuk baud rate 9600 bps, panjang kabel maksimum yang direkomendasikan adalah 15 m, sedangkan untuk baud rate 56 kbps, panjang kabel maksimum adalah 2,6 m. Secara umum, data dikirimkan dalam bentuk paket data per 1 byte, yang terdiri dari bit start/stop, bit data (bisa berukuran 5, 7 atau 8 bit), dan bit parity. Bit start/stop digunakan untuk sinkronisasi. Bit parity digunakan untuk pengecekan error, untuk memastikan bahwa bit data yang dikirimkan adalah benar.

Standar penyambungan komunikasi serial yang cukup terkenal adalah RS-232, yang diciptakan oleh IBM untuk perangkat komputer. RS-232 ini digunakan untuk menghubungkan komputer dengan keyboard, mouse, printer, modem dan juga diterapkan pada alat instrumentasi di industri. Kelebihan dari komunikasi serial RS-232 ini adalah bisa melakukan komunikasi Full-duplex, yaitu kirim dan terima data pada waktu yang bersamaan. Kelemahan dari komunikasi serial RS-232 ini hanya bisa untuk komunikasi antar 2 alat saja, dengan jarak maksimum 15 m.

Agar bisa melakukan komunikasi serial lebih dari 2 alat dengan jarak yang lebih jauh, diciptakanlah standar RS-422. Standar RS-422 menggunakan 2 jalur, baik untuk saluran pengiriman maupun untuk penerimaan data. Dengan menggunakan 2 jalur, sinyal yang diterima lebih kebal gangguan dan kabel bisa lebih panjang, karena nilainya tidak lagi diukur terhadap Ground, tetapi diukur dari perbedaan sinyal antara kedua jalur tersebut (perbedaan hingga 0,2 V masih bisa dideteksi). Standar RS-485 merupakan penyempurnaan dari RS-422, yang bisa menghubungkan hingga 32 alat dengan jarak maksimum 1,2 km melalui komunikasi serial (biasanya menggunakan protokol Modbus). Kelemahan dari RS-485 ini adalah komunikasi hanya bisa secara Half-duplex, yaitu kirim dan terima data harus bergantian, tidak bisa dilakukan secara bersamaan. Lebih jauh, komunikasi serial RS-485 ini akan dibahas pada Bab 3.

Sejak tahun 2000, port serial untuk RS-232 (konektor DB9) di komputer telah digantikan dengan port USB. Sekalipun tidak ada port serial, namun komunikasi serial di komputer tetap dapat dilakukan melalui port USB.

LabVIEW juga menyediakan fungsi untuk komunikasi serial, dengan driver NI VISA yang menghubungkan LabVIEW dengan hardware port serial komputer. Gambar 2.2 berikut menunjukkan contoh penerapan fungsi komunikasi serial dari 2 buah program LabVIEW yang sama.



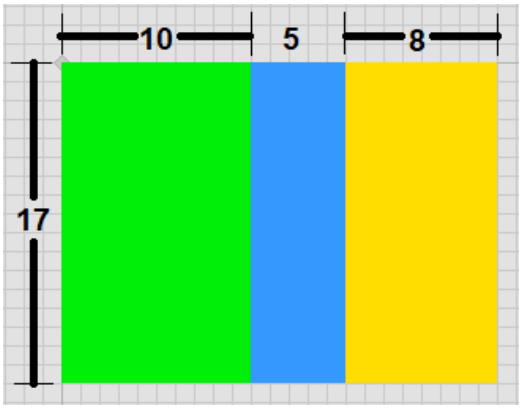
Gambar 2.2 Komunikasi Serial antara 2 buah program LabVIEW

Keterangan Gambar: Data yang dikirimkan dari program LabVIEW yang kiri, akan diterima oleh program LabVIEW yang kanan, begitu juga sebaliknya. Toggle Switch Boolean akan mengirimkan data dengan huruf awalan a, Slide Integer akan mengirimkan data dengan huruf awalan b. Data yang diterima, apabila diawali huruf a, akan meng-hidup/mati-kan LED Boolean2, apabila diawali huruf b, akan me-naik/turun-kan tinggi isi Tank. Karena komunikasi dilakukan secara software, maka port COM yang digunakan harus merupakan pasangan COM virtual. Gunakan software VSPD untuk membuat pasangan COM virtual ini (dalam contoh di sini, penulis menggunakan COM3 dan COM4 sebagai pasangan COM Virtual).

Berikut ini langkah-langkah untuk membuat program di atas:

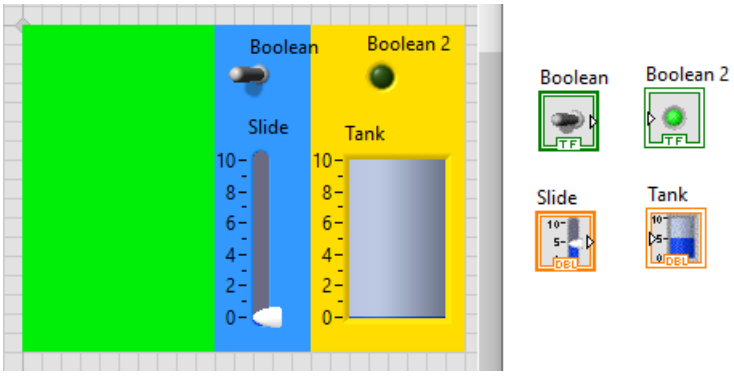
1. Buka LabVIEW, pada menu File pilih New VI.
2. Tekan tombol Control dan T untuk membuat Front Panel dan Block Diagram berdampingan.

3. Klik kanan Front Panel, muncul Palet Controls. Pilih Modern, pilih Decorations, pilih Flat Box. Buat 3 buah kotak dengan Flat Box, dengan ukuran seperti terlihat pada Gambar 2.3.



Gambar 2.3 Ukuran Flat Box berturut-turut: 10x17, 5x17, 8x17

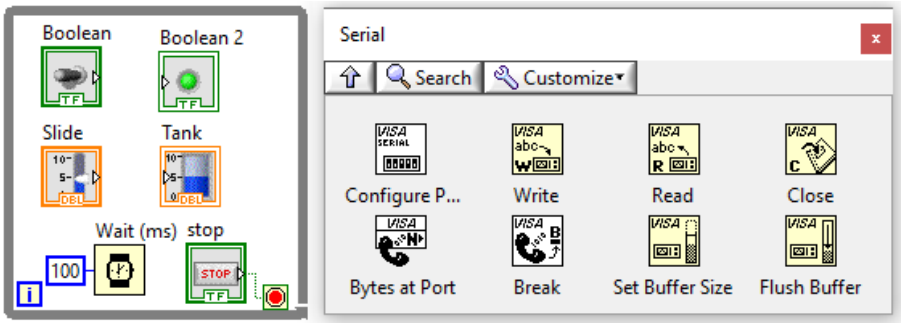
4. Untuk membuat warna, klik kanan sembari menekan tombol Shift. Pada Palet Tools yang muncul, pilih tool kuas, sentuhkan pada Flat Box, klik kanan untuk membuka kotak warna. Pilih sembarang warna, maka warna Flat Box akan sesuai dengan warna yang dipilih. Setelah selesai, jangan lupa untuk mengembalikan mode Tool menjadi otomatis lagi, dengan cara membuka kembali Palet Tools, dan nyalakan LED di bagian atas Palet.
5. Di Front Panel, tempatkan sebuah Horizontal Toggle Switch, sebuah Round LED, sebuah Vertical Pointer Slide, dan sebuah Tank.



Gambar 2.4 Toggle Switch, Round LED, Pointer Slide dan Tank di Front Panel

6. Di Block Diagram, lingkupi keempat icon dengan Struktur While Loop.
7. Tambahkan tombol Stop (Create Control) pada Terminal Loop Condition.
8. Tambahkan icon Wait(ms), beri nilai 100 pada kaki inputnya.
9. Berikutnya, tambahkan fungsi-fungsi Komunikasi Serial pada program. Untuk membuka fungsi-fungsi Komunikasi Serial ini, klik kanan pada halaman Block Diagram sehingga muncul Palet Functions. Pada Palet Functions, tekan tombol Search. Pada kolom Search, ketik kata "serial" diikuti Enter. Klik 2 kali pada hasil pencarian di baris paling atas, maka akan terbuka 8 fungsi Komunikasi Serial seperti Gambar 2.5 berikut.

Catatan: Cara lain untuk membuka fungsi Komunikasi Serial di LabVIEW adalah dengan membuka Palet Functions, pilih Instrument I/O, pilih Serial.

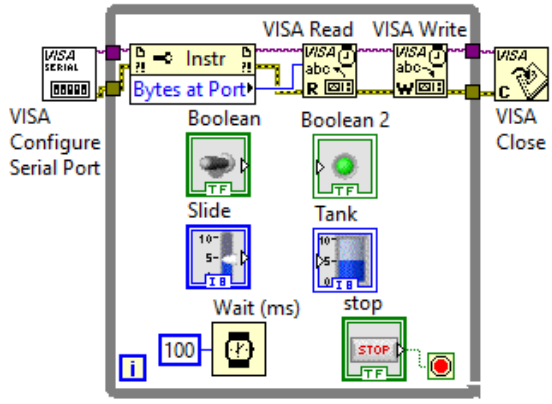


Gambar 2.5 Delapan buah fungsi Komunikasi Serial

Keterangan:

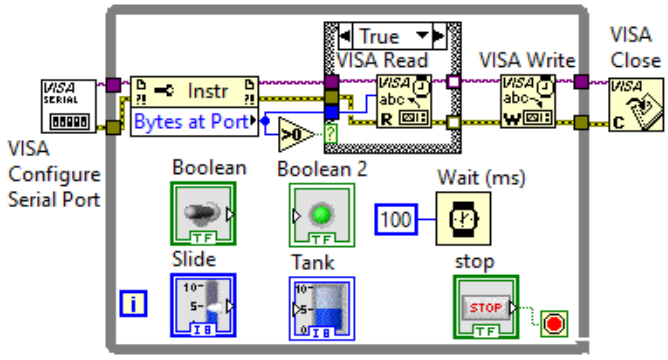
- VISA Configure Port untuk membuka dan mengatur konfigurasi port.
- VISA Write untuk mengirimkan data melalui komunikasi serial.
- VISA Read untuk menerima data melalui komunikasi serial.
- VISA Close untuk menutup komunikasi serial.
- VISA Bytes at Port untuk mengetahui banyaknya data (byte) di port serial.
- VISA Break untuk memutus komunikasi sementara.
- VISA Set Buffer Size untuk mengatur ukuran buffer.
- VISA Flush Buffer untuk membersihkan data di dalam buffer.

10. Tempatkan icon VISA Configure Port, Bytes at Port, Read, Write dan Close, dan hubungkan seperti Gambar 2.6 berikut:



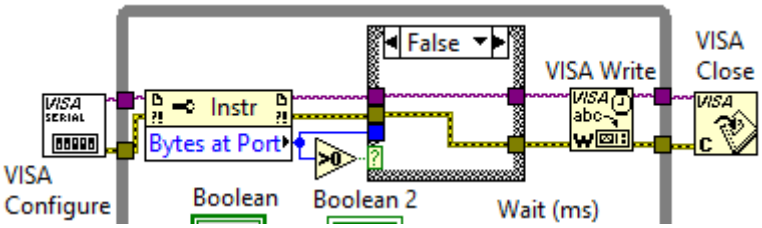
Gambar 2.6 Menghubungkan icon VISA Configure Port, Bytes at Port, Read, Write dan Close

11. Icon VISA Configure Port dan VISA Close diletakkan di luar While Loop karena hanya perlu dijalankan sekali, sedangkan VISA Bytes at Port, VISA Read dan VISA Write ditempatkan di dalam While Loop karena harus dijalankan berulang kali.
12. Agar program tidak menjalankan fungsi VISA Read terus-menerus, tetapi hanya menjalankan fungsi Read apabila ada data di port serial, maka tambahkan sebuah Struktur Case melingkupi VISA Read, dan hubungkan Terminal Case Selector (?) dengan output VISA Bytes at Port, dengan tambahan fungsi Greater than 0, seperti Gambar 2.7 berikut:



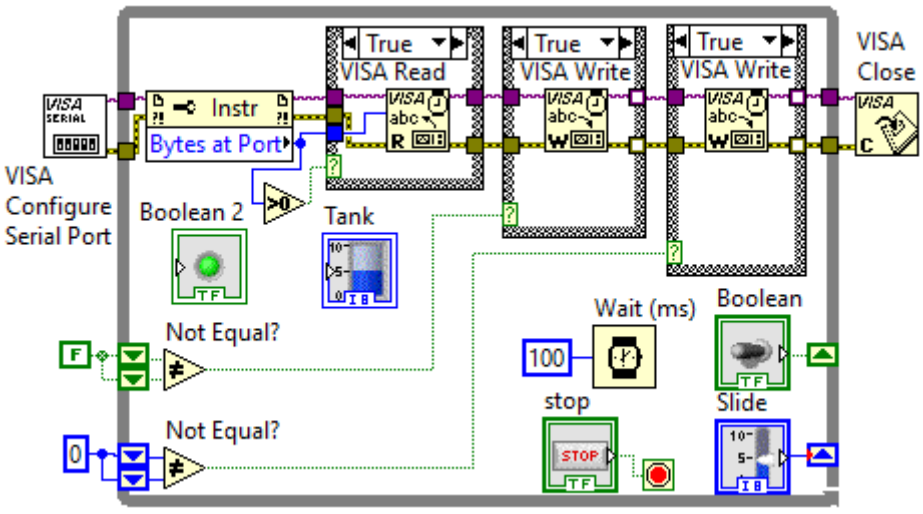
Gambar 2.7 Menambahkan Case Structure agar fungsi Read tidak dijalankan terus-menerus

13. Untuk isi kotak False pada Struktur Case, tidak ada fungsi Read, teruskan garis data di kiri dengan di kanan.



Gambar 2.8 Ketika tidak ada data di port serial, tidak dilakukan pembacaan

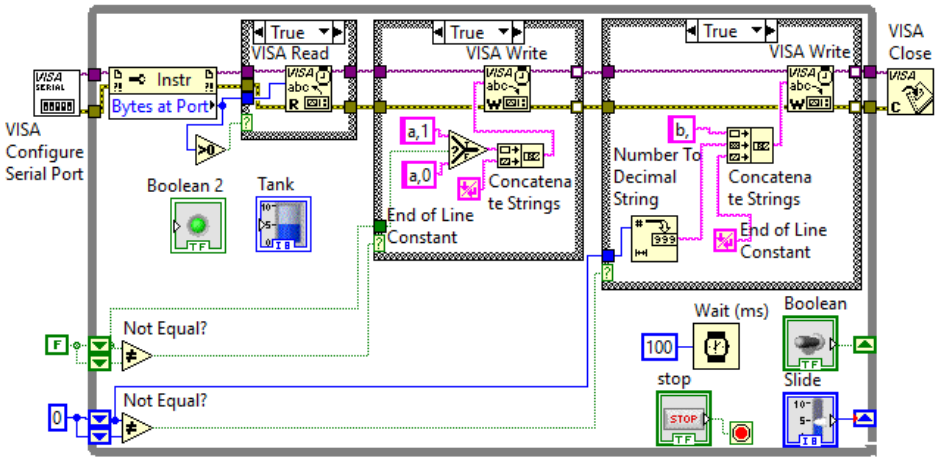
14. Ada 2 data yang akan dikirimkan, yaitu data Horizontal Toggle Switch dan Vertical Pointer Slide. Untuk menghemat "resource" atau sumber daya, maka hanya data yang berubah saja yang dikirimkan. Perubahan data ini dapat diketahui dengan menggunakan Shift Register, yaitu dengan membandingkan data yang sekarang dengan data yang sebelumnya. Apabila tidak sama (*Not Equal?*), barulah data tersebut dikirimkan. Untuk itu, tambahkan 2 buah Shift Register dan 2 buah Case Structure, dengan masing-masing Case Structure berisi sebuah icon VISA Write. Hubungkan seperti Gambar 2.9 berikut:



Gambar 2.9 Membuat program agar hanya data yang berubah saja yang dikirimkan

Catatan: Mulai Gambar 2.6 di atas, icon Slide (Vertical Pointer Slide) dan Tank telah diubah tipe datanya, dari yang semula bertipe data Double (DBL), ditandai dengan warna garis oranye, telah diubah menjadi Integer dengan warna garis biru. Untuk mengubah tipe data: klik kanan pada icon Slide, pilih Representation, pilih U8 (*unsigned integer* 8 bit, jangkauan nilai: 0 sd 255) atau U16 (*unsigned integer* 16 bit, jangkauan nilai: 0 sd 65535). Lakukan hal yang serupa untuk icon Tank.

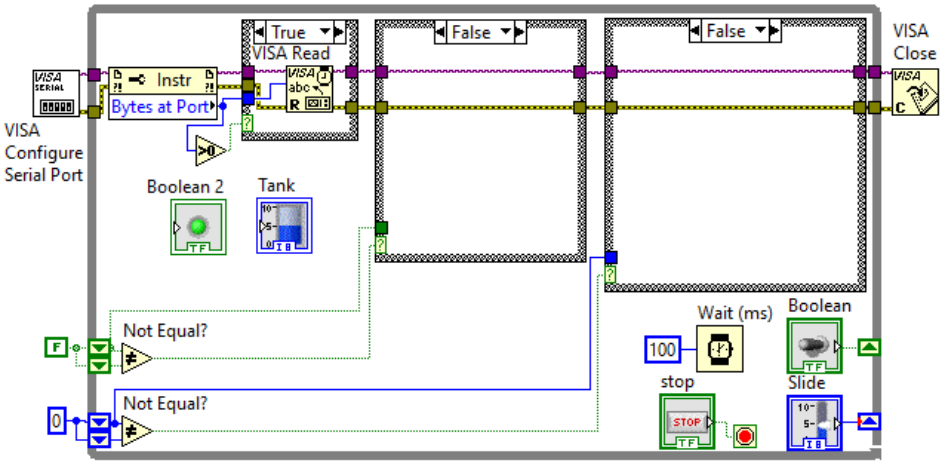
15. Agar komunikasi antara pengirim dan penerima bisa berjalan dengan baik, dibutuhkan kesepakatan format data. Format data yang umum dipakai adalah penggunaan koma sebagai pemisah data, dan Enter sebagai penanda akhir data. Kemudian untuk membedakan data dari Toggle Switch dengan data dari Pointer Slide, data dari Toggle Switch diberi huruf awalan a, sedangkan data dari Pointer Slide diberi huruf awalan b. Ketika Toggle Switch diubah nilainya dari False menjadi True, maka data yang dikirimkan menjadi a,1 diikuti Enter. Sebaliknya ketika Toggle Switch diubah dari True menjadi False, maka data yang dikirimkan akan menjadi a,0 diikuti Enter. Begitu pula ketika Pointer Slide diubah nilainya dari 0 menjadi 10, maka data yang dikirimkan menjadi b,10 diikuti Enter. Karena input VISA Write bertipe data String, sedangkan Pointer Slide bertipe data Integer, maka perlu diubah dengan fungsi Number To Decimal String.



Gambar 2.10 Menerapkan format data pengiriman dengan koma dan Enter (End of Line)

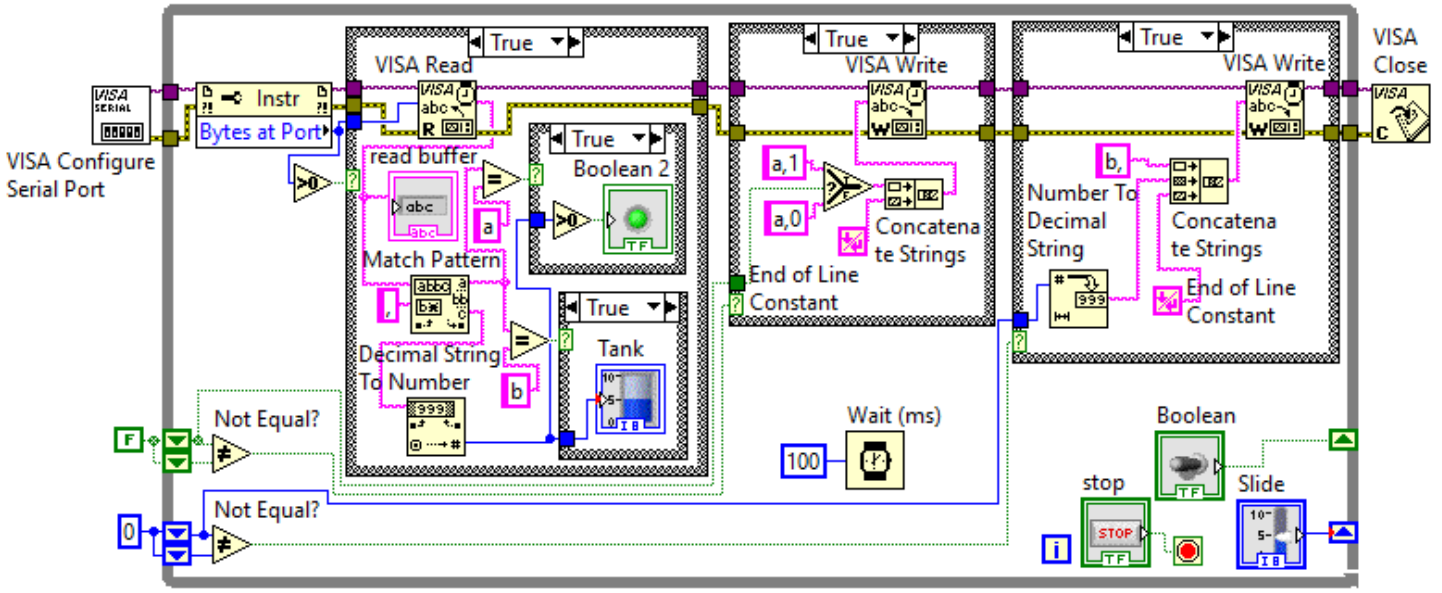
Catatan: Pada Gambar 2.10 di atas, diperlukan fungsi Concatenate String yang dapat menggabungkan data String, agar sesuai format data, yaitu dipisahkan dengan koma dan diakhiri Enter, untuk kemudian diberikan ke input VISA Write.

16. Agar pengiriman data dilakukan hanya bila terjadi perubahan data, maka pada kotak False untuk kedua Struktur Case tidak ada fungsi Write, garis data di kiri diteruskan ke kanan.



Gambar 2.11 Ketika nilai Switch dan Slide tidak berubah, tidak ada data yang dikirimkan

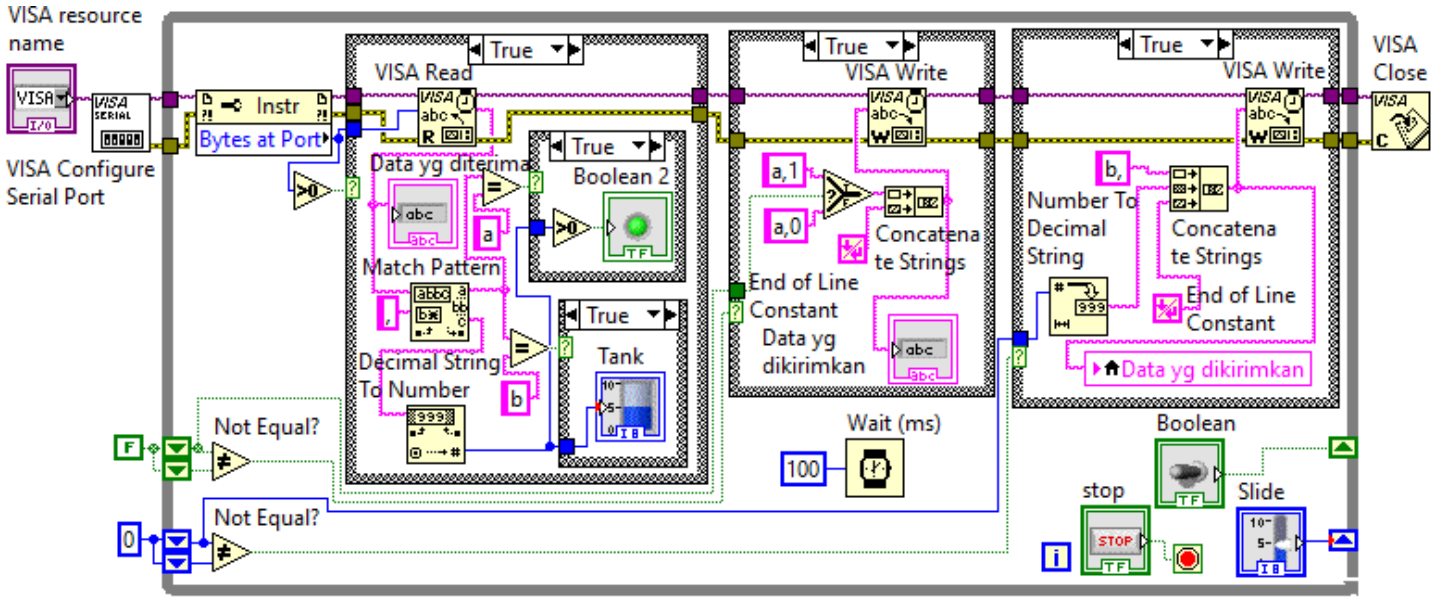
17. Kembali ke VISA Read. Agar data yang diterima VISA Read dapat terlihat, klik kanan kaki output read buffer, pilih Create, pilih Indicator.
18. Diinginkan data yang diterima oleh VISA Read tersebut dapat ditampilkan dalam bentuk nyala/padam Round LED dan tinggi isi Tank. Apabila datanya dari Toggle Switch, maka akan ditampilkan dalam bentuk nyala/padam Round LED. Apabila datanya Vertical Pointer Slide, maka akan ditampilkan dalam bentuk tinggi isi Tank. Karena data dipisahkan dengan koma, maka untuk mendapatkan data sebelum koma dan setelah koma, digunakan fungsi Match Pattern, yang ada di Palet Functions, di Programming, di String. Hubungkan input String Match Pattern dengan kaki output read buffer, dan pada kaki input regular expression, klik kanan, dan pilih Create, pilih Constant, dan isi dengan koma.



Gambar 2.12 Indicator read buffer akan menampilkan data yang diterima VISA Read, sedangkan Match Pattern akan memisahkan data tersebut. Data sebelum koma berupa huruf a atau b. Bila huruf a, berarti data dari Toggle Switch, sedangkan bila huruf b, berarti data dari Vertical Pointer Slide. Bila data dari Toggle Switch, maka data setelah koma bisa 1 atau 0. Bila 1, Round LED menyala, bila 0 Round LED padam. Bila data dari Vertical Pointer Slide, data setelah koma ditampilkan dalam bentuk Tinggi isi Tank. Baik data untuk Round LED maupun data untuk Tinggi isi Tank, keduanya diperoleh dari fungsi Decimal String to Number, yang mengubah String menjadi angka.

19. Seperti terlihat pada Gambar 2.12 di atas, kaki output before substring Match Pattern terhubung dengan 2 buah icon Equal; icon Equal yang pertama diberi huruf a, dan icon Equal yang kedua diberi huruf b. Masing-masing output icon Equal tersebut dihubungkan dengan Case Selector (?) Struktur Case. Case True pada Struktur Case pertama berisi Round LED, sedangkan Case True pada Struktur Case kedua berisi Tank. Berikutnya, kaki output after substring Match Pattern dihubungkan dengan icon Decimal String to Number, yang akan mengubah tipe data String menjadi Integer. Output icon yang sudah menjadi angka ini, dihubungkan dengan icon Tank dan juga Round LED, dengan diberi sisipan Greater than 0.
20. Berikutnya, tambahkan Port pada VISA Configure Port. Klik kanan pada kaki input VISA Resource Name, pilih Create, pilih Control.
21. Di Front Panel, tempatkan kotak VISA Resource Name, Indicator read buffer, tombol Stop pada kotak Flat Box yang paling kiri.
22. Agar tidak hanya data yang diterima saja yang ditampilkan, tetapi juga data yang dikirimkan, duplikasi (copy paste) Indicator read buffer, dan letakkan di bawah read buffer. Ubah nama read buffer menjadi Data yang diterima, dan nama Indicator yang baru dengan Data yang dikirimkan.
23. Di Block Diagram, hubungkan icon Indicator yang baru (Data yang dikirimkan) dengan output icon Concatenate String. Karena ada 2 buah icon Concatenate String, untuk Concatenate String yang kedua, gunakan Local Variable Indicator tersebut, seperti ditunjukkan pada Gambar 2.14.

Gambar 2.13 Tampilan program Komunikasi Serial



Gambar 2.14 Program LabVIEW untuk komunikasi serial. VISA Read hanya dijalankan apabila ada data di port Serial, yang bisa diketahui dengan bantuan VISA Bytes at Port. VISA Write hanya dijalankan ketika ada perubahan data, yang bisa diketahui dengan bantuan Shift Register. Data yang dikirimkan maupun yang diterima terdiri dari 2 buah tipe data, yaitu tipe Boolean dan tipe Integer. Data Bolean dihasilkan oleh Toggle Switch, dan data Integer dihasilkan oleh Vertical Pointer Slide. Data yang diterima dengan tipe Boolean, ditampilkan dalam bentuk nyala/padam Round LED, sedangkan data tipe Integer ditampilkan dalam bentuk Tinggi isi Tank.

24. Beri nama program LabVIEW yang telah dibuat ini dengan nama Serial1.
25. Kemudian Save As program ini, dan beri nama Serial2.
26. Buka program Serial1 dan Serial2, jalankan keduanya. Namun sebelum dijalankan, pada kotak VISA Resource Name, pilih port COM yang berpasangan. Dalam contoh di sini, penulis menggunakan COM3 di program Serial1, dan COM4 di program Serial2.

Catatan: Apabila pada kotak VISA Resource Name tidak muncul pilihan port COM, bisa karena driver NI VISA belum diinstal, atau karena belum adanya pasangan COM virtual. Gunakan software VSPD untuk membuat pasangan COM virtual.

27. Pindahkan tuas Toggle Switch di program Serial1, dan amati nyala Round LED di program Serial2. Naik turunkan Slide di program Serial1, dan amati Tinggi isi Tank di program Serial2. Lakukan sebaliknya. Harusnya setiap kali pengubahan kondisi pada Toggle Switch dan Slide di program yang satu, akan mengakibatkan perubahan kondisi pada Round LED dan Tank di program yang lain, seperti ditunjukkan pada Gambar 2.2.
28. Sampai di sini Komunikasi Serial LabVIEW selesai.

2.3 Komunikasi Serial Arduino

Sama seperti LabVIEW, Arduino juga memiliki fungsi untuk komunikasi serial. Ada banyak fungsi Komunikasi Serial di Arduino (hingga 21 fungsi), pembaca dapat melihatnya di software Arduino IDE, di menu Help, di Reference, di topik Serial. Dari fungsi-fungsi komunikasi serial yang tersedia tersebut, di Sub Bab ini, hanya akan dipakai 5 fungsi, yaitu:

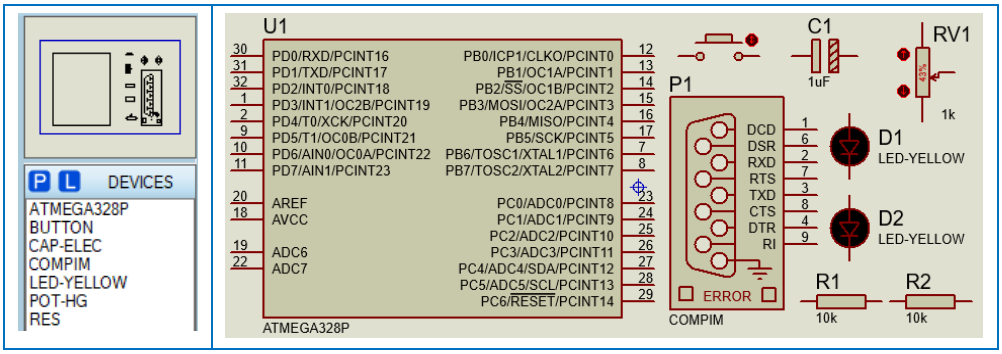
1. Serial.begin(): fungsi ini sama seperti VISA Configure Port di LabVIEW, yaitu mengatur kecepatan dan konfigurasi port komunikasi serial, namun bedanya, fungsi ini tidak mengatur saluran port yang digunakan. Untuk saluran port yang digunakan, diatur di menu Tools, di pilihan Port.
2. Serial.available(): fungsi ini sama seperti VISA Bytes at Port di LabVIEW, yaitu untuk mengetahui banyaknya data (byte) yang tersedia untuk dibaca di buffer penerimaan port serial.

3. SerialEvent(): merupakan subrutin khusus yang akan dipanggil setiap kali ada data di buffer penerimaan port serial.
4. Serial.read(): fungsi ini sama seperti VISA Read di LabVIEW, yaitu untuk menerima atau menangkap data. Setiap kali Serial.read() dijalankan, data di buffer penerimaan akan berkurang 1 byte.
5. Serial.print(): fungsi ini sama seperti VISA Write di LabVIEW, yaitu untuk mengirimkan data melalui komunikasi serial. Apabila di LabVIEW menggunakan End of Line Constant (Enter) sebagai penanda akhir data, di Arduino, instruksi Serial.print() diberi tambahan ln, sehingga menjadi Serial.println(), yang berarti data yang dikirimkan diikuti dengan Enter. Enter ini terdiri dari 2 karakter tidak tercetak, yaitu char(13) atau Carriage Return dan char(10) atau New Line.

Di samping fungsi-fungsi komunikasi serial tersebut, di software Arduino IDE, disediakan 2 buah alat (di menu Tools) yang sangat berguna, yaitu Serial Monitor yang dapat menampilkan data yang diterima maupun yang dikirimkan melalui komunikasi serial, dan Serial Plotter, yang dapat menampilkan data yang diterima tersebut dalam bentuk grafik.

Untuk memudahkan pembuatan komunikasi serial di Arduino ini, di sini digunakan software Proteus, yang dapat mensimulasikan rangkaian Arduino dengan baik, tanpa resiko kerusakan komponen, dan dapat disimpan. Berikut ini langkah-langkah pembuatan komunikasi serial di Arduino dengan Proteus:

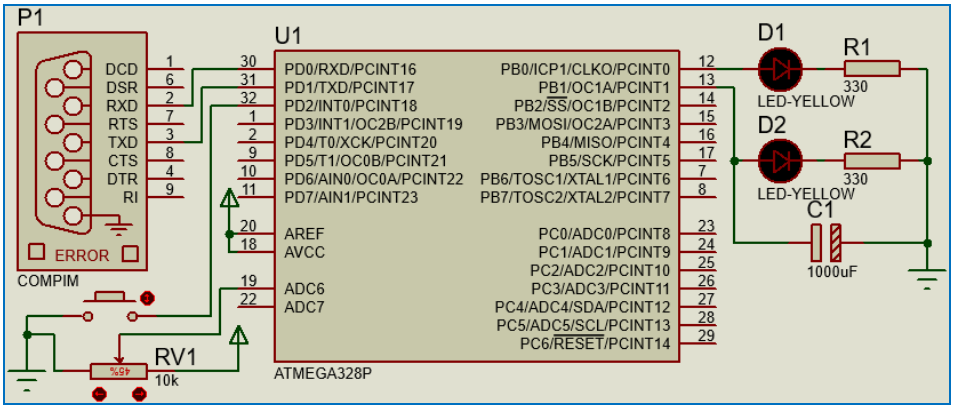
1. Buka software Proteus, ambil komponen seperti Gambar 2.15 berikut:



Gambar 2.15 Komponen: ATmega328P, Button, Cap-Elec, Compim, 2 LED-Yellow, Pot-HG, 2 Res

Catatan: Untuk mengambil komponen di Proteus, pertama buat dulu New Project, klik Next sampai selesai. Pada halaman New Project, klik kanan, pilih Place, pilih Component, pilih From Libraries. Pada kolom Keywords, ketik ATmega328P, klik Enter, klik OK. Ulangi untuk komponen yang lain. Untuk memutar komponen, klik kanan komponen, pilih Rotate atau Mirror.

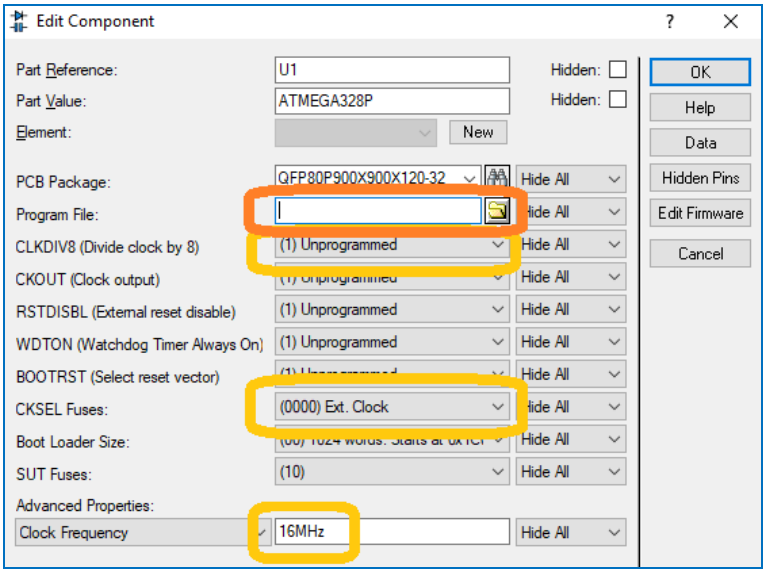
2. Rangkai komponen di atas sehingga seperti Gambar 2.16 berikut:



Gambar 2.16 Membuat rangkaian komponen dari Gambar 2.15

Catatan: Beberapa komponen perlu diubah nilainya. Untuk mengubah nilai, klik komponen, pilih Edit Properties. Ubah nilai Resistance RV1 dari 1k menjadi 10k. Ubah pula nilai Resistance R1 dan R2, dari 10k menjadi 330 ohm. Ubah pula nilai Capacitance C1, dari 1uF menjadi 1000uF. Perhatikan tanda panah ke atas. Tanda panah ke atas adalah simbol dari tegangan 5V. Simbol ini bisa diperoleh dengan cara klik kanan halaman, pilih Place, pilih Terminal, pilih Power. Tempatkan di kaki AREF dan AVCC, serta di RV1. Selain Power, di Terminal juga ada simbol Ground. Tempatkan Ground ini di kaki RV1 dan Button, serta di C1, R1 dan R2.

3. Untuk mensimulasikan Arduino, yang dalam hal ini digunakan komponen ATmega328P (ATmega328P merupakan mikrokontroler yang digunakan di Arduino Uno dan Arduino Nano), perlu mengatur beberapa parameter di Edit Properties ATmega328P seperti Gambar 2.17 berikut:



Gambar 2.17 Empat parameter yang harus diatur di Edit Properties ATmega328P

Keterangan: Empat parameter yang harus diatur adalah Program File, CLKDIV8, CKSEL Fuses, dan Clock Frequency. Untuk Program File, harus diisi dengan lokasi direktori file Hex, hasil kompilasi program Arduino, yang akan dibuat di langkah berikutnya. Untuk CLKDIV8, pilih Unprogrammed. Untuk CKSEL Fuses, pilih Ext Clock. Untuk Clock Frequency, isi dengan 16MHz.

4. Kolom Program File harus diisi dengan lokasi file Hex, yang dihasilkan oleh kompilasi program Arduino. Untuk itu, buka software Arduino IDE, dan ketik program Arduino berikut ini.

No.	Kode Program 01: Serial_1
01.	`int a0 = 0;`
02.	`int b0 = 0;`
03.	`char pesan[8];`
04.	`int i = 0;`
05.	`char pesan0;`
06.	`char pesan1;`
07.	
08.	`void setup() {`
09.	`  Serial.begin(9600);`
10.	`  pinMode(2, INPUT_PULLUP);//Button`
11.	`  pinMode(8, OUTPUT);//LED D1`

```
12.   pinMode(9, OUTPUT);//LED D2 PWM
13. }
14.
15. void loop() {
16.   int a = !digitalRead(2);//Button
17.   if (a != a0) {
18.     if (a) Serial.println("a,1");
19.     else Serial.println("a,0");
20.     delay(50);
21.   }
22.   a0 = a;
23.   int b = map(analogRead(6), 0, 1023, 0, 255);//RV1
24.   if (b != b0) {
25.     Serial.print("b,");
26.     Serial.println(b);
27.     delay(50);
28.   }
29.   b0 = b;
30. }
31.
32. void serialEvent() {
33.   while (Serial.available()) {
34.     pesan1 = Serial.read();
35.     pesan[i] = pesan1;
36.     i++;
37.     if (pesan1 == '\n') {
38.       pesan[i] = '\0';
39.       pesan0 = pesan[0];
40.       pesan[0] = '0';
41.       pesan[1] = '0';
42.       int angka = atoi(pesan);
43.       if (pesan0 == 'a') digitalWrite(8, angka);
44.       if (pesan0 == 'b') analogWrite(9, angka);
45.       i = 0;
46.     }
47.   }
48. }
```

Keterangan program:

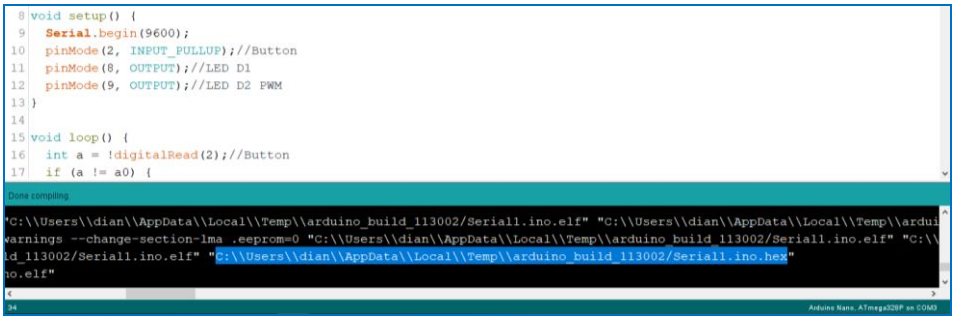
1. Baris 1-2: deklarasi variable a0 dan b0 yang digunakan untuk mendeteksi apakah ada perubahan kondisi pada Tombol dan Potensio.
2. Baris 3-6: deklarasi variable yang digunakan untuk menampung data yang diterima melalui instruksi Serial.read(). Karena instruksi Serial.read() mengambil data 1 byte demi 1 byte, maka diperlukan sebuah variable array. Array adalah sebuah kelompok data, dengan setiap anggota atau data

didalamnya memiliki nomor urut. Baris ke-3 menunjukkan deklarasi variable array, yang dinamai pesan, dengan tipe data char, dengan maksimum anggota sebanyak 8. Baris ke-4 menunjukkan deklarasi variable nomor urut.

3. Baris 8-13: blok void setup, yang berisi pengaturan seting serial dan penugasan kaki input output. Kaki D2 yang terhubung dengan tombol, ditugaskan sebagai input. Kaki D8 yang terhubung dengan LED D1, ditugaskan sebagai output. Kaki D9 yang terhubung dengan LED D2, ditugaskan sebagai output PWM. Agar nilai PWM yang dihasilkan menyerupai nilai analog, pada rangkaian ditambahkan sebuah kapasitor 1000uF. Kaki A6 yang terhubung dengan Potensio tidak perlu ditugaskan, karena kaki A6 secara default sudah merupakan kaki input analog.
4. Baris 15-30: blok void loop, yang berisi program untuk pengiriman data.
5. Baris 32-48: blok void serialEvent(), yang berisi program penerimaan data.
6. Untuk pengiriman data, data hanya dikirimkan apabila ada perubahan nilai pada Tombol atau pada Potensio. Data yang dari Tombol, diberi awalan huruf a, sedangkan data yang dari Potensio, diberi awalan huruf b. Ketika Tombol ditekan, nilai yang dikirimkan adalah 1, sedangkan ketika Tombol dilepas, nilai yang dikirimkan adalah 0. Data yang dari Potensio bisa bervariasi, dari 0-255. Karena ADC di Arduino memiliki resolusi 10 bit, nilai pembacaan analog bisa bervariasi dari 0-1023, maka agar bisa menghasilkan nilai 0-255, digunakan fungsi map, yang memetakan nilai analog Potensio dari 0-1023 menjadi 0-255.
7. Untuk penerimaan data, data ditampung ke dalam variable array pesan. Baris 34 – 36 menunjukkan bagaimana data yang diambil oleh Serial.read(), diisikan satu byte demi satu byte ke variable array pesan, mulai dari indeks (nomor urut) pertama, yang akan bertambah setiap kali pengulangan.
8. Data yang diterima terdiri dari sebuah huruf awalan, diikuti koma, diikuti nilai dari Tombol atau dari Potensio, diakhiri Enter. Sebagai contoh, anggap data yang diterima: “b,255” diikuti Enter. Enter adalah sebuah gabungan karakter yang terdiri dari karakter CR (‘\r’) dan LF (‘\n’).
9. Untuk data “b,255” diikuti Enter, di variable array pesan akan berisi data sebagai berikut: pesan[0]=’b’, pesan[1]=’,’, pesan[2]=’2’, pesan[3]=’5’, pesan[4]=’5’, pesan[5]=’\r’, pesan[6]=’\n’.

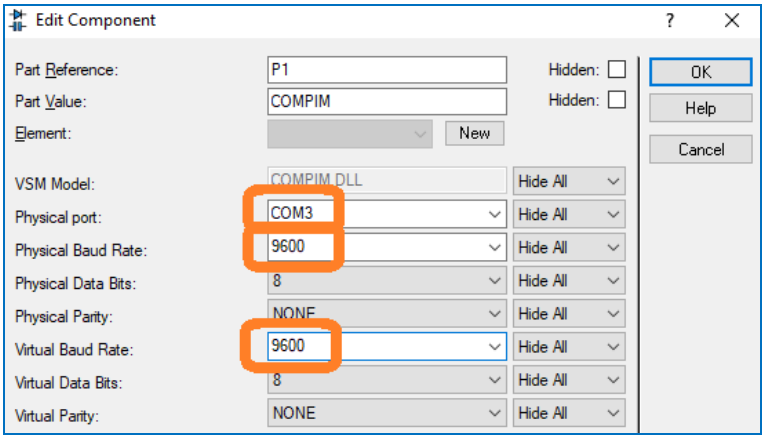
10. Untuk variable array dengan tipe data char, dianggap lengkap datanya apabila diakhiri dengan karakter null (‘\0’). Itulah sebabnya, pada baris ke-38, data terakhir (pesan[i]) diisi dengan null (‘\0’). Dengan penambahan null ini, maka total byte variable array pesan adalah 8 byte. Itulah sebabnya, pada deklarasi variable array pesan, dibuat isinya 8 (byte).
11. Baris 37 – 48 menunjukkan apabila akhiran data diterima (karakter LF atau ‘\n’), maka buat LED D1 atau LED D2 padam atau menyala sesuai nilai yang diterima. Nilai yang diterima ini bisa diperoleh dengan menggunakan fungsi **atoi**, yang akan mengubah data array char menjadi integer. Hanya saja, fungsi **atoi** hanya bisa mengubah apabila data di variable tersebut berbentuk angka. Dalam contoh data “b,255”, fungsi **atoi** tidak akan bisa mengubah data ini. Agar data “b,255” bisa diubah menjadi integer, huruf b dan tanda koma harus diganti dengan char ‘0’, sehingga menjadi “00255”, seperti ditunjukkan pada baris 40 dan 41, di mana pesan[0] dan pesan[1] diisi dengan ‘0’.

5. Berikutnya, kompilasi program di atas dengan menekan tombol Verify di Toolbar Arduino IDE, yaitu tombol paling kiri bergambar tanda centang. Namun sebelum melakukan kompilasi, buat pengaturan pada Preferences. Buka menu File, pilih Preferences. Pada kotak Preferences, pada kolom *Show verbose output during:*, beri tanda centang pada Compilation. Setelah itu, pada menu Tools, pilih Board, pilih Arduino Nano. Kemudian lakukan kompilasi. Harusnya lokasi file Hex akan muncul pada kotak di bagian bawah, seperti ditunjukkan Gambar 2.18 berikut ini:



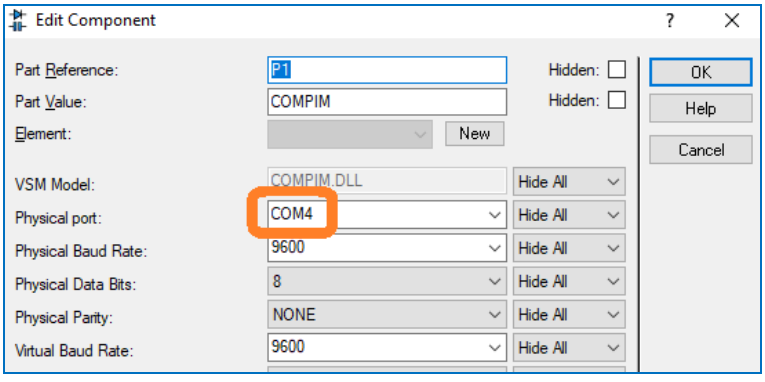
Gambar 2.18 Lokasi file Hex muncul di kotak bawah software Arduino IDE

6. Setelah lokasi file Hex muncul di kotak bawah software Arduino IDE, pilih lokasi tersebut, tekan Control dan C untuk meng-copy, dan tempelkan (tekan Control dan V) di kolom Program File Edit Properties ATmega328P.
7. Klik 2 kali pada komponen COMPIM untuk membuka Edit Properties COMPIM. Isi kolom Physical port dengan salah satu COM dari pasangan COM. Dalam contoh di sini, penulis menggunakan COM3. Ubah isi Physical Baud Rate dan Virtual Baud Rate dari angka 2400 menjadi 9600.



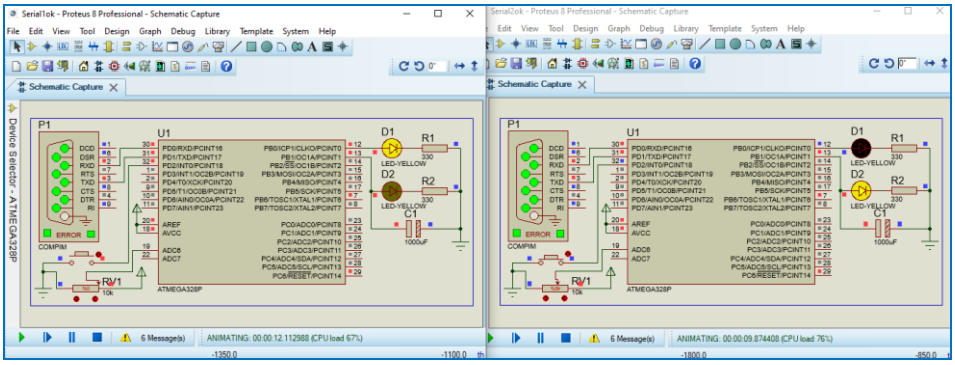
Gambar 2.19 Pengaturan isi Physical Port, Physical dan Virtual Baud Rate COMPIM

8. Berikutnya, simpan Project di Proteus dengan nama Serial1, dan Save As Project tersebut dengan nama Serial2. Pada Project Serial2, klik 2 kali pada komponen COMPIM. Pada kolom Physical Port, isi pasangan COM.



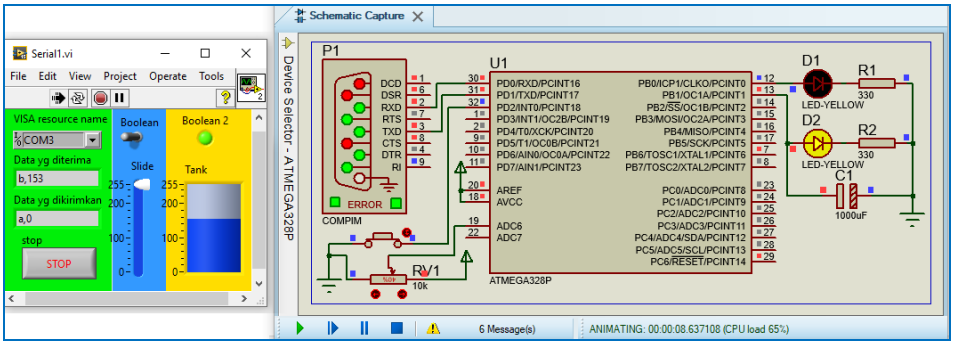
Gambar 2.20 Isi Physical Port dengan pasangan COM

9. Buka Project Proteus Serial1 dan Serial2 dalam satu layar. Jalankan kedua Project tersebut bersama-sama. Tekan Tombol dan geser Potensio dari Serial1, dan amati kondisi LED1 dan LED2 di Serial2. Lakukan hal sebaliknya. Harusnya setiap kali pengubahan kondisi pada Tombol dan Potensio, akan mengakibatkan perubahan kondisi pada LED1 dan LED2 di rangkaian lain, seperti ditunjukkan pada Gambar 2.21.



Gambar 2.21 Komunikasi serial antara 2 buah Arduino yang disimulasikan di Proteus

10. Stop simulasi kedua Project Proteus di atas. Diinginkan komunikasi serial antara LabVIEW dengan Arduino. Buka program LabVIEW Serial1.vi. Pada kotak VISA Resource Name, isi dengan pasangan COM, dalam contoh disini menggunakan COM3. Agar nilai Slide dan Tank bisa setara dengan nilai PWM dan analog Potensio, maka ubah angka maksimum Slide Tank dari 10 menjadi 255. Setelah itu, jalankan program LabVIEW.
11. Jalankan Project Serial2 Proteus. Pastikan COM yang digunakan di COMPIM merupakan pasangan COM dengan COM di program LabVIEW.
12. Klik Toggle Switch dan naik turunkan Slide di LabVIEW, amati kondisi LED1 dan LED2 di Project Serial2 Proteus. Berikutnya, tekan Tombol dan geser Potensio di Project Serial2 Proteus, dan amati kondisi Round LED dan Tank di LabVIEW. Harusnya setiap kali pengubahan kondisi objek Control di LabVIEW, akan mengakibatkan perubahan kondisi pada komponen output di Proteus, dan pengubahan kondisi pada komponen input di Proteus, akan mengakibatkan perubahan kondisi objek Indicator di LabVIEW, seperti ditunjukkan pada Gambar 2.22.



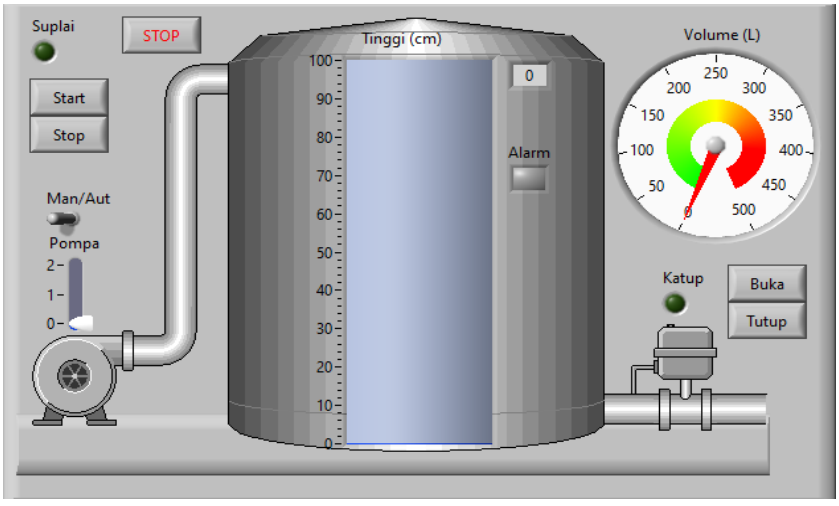
Gambar 2.22 Komunikasi serial antara LabVIEW dengan Arduino di Proteus

13. Sampai di sini Komunikasi Serial Arduino selesai.

2.4 Komunikasi Serial pada Tampilan Pengisian Tangki

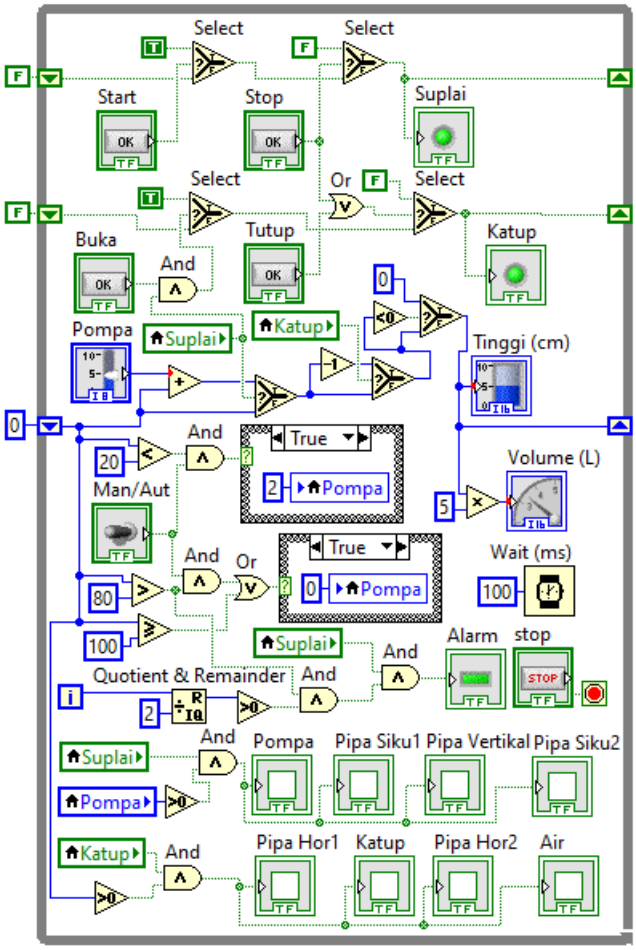
Pada Gambar 2.22 di atas menunjukkan komunikasi serial antara LabVIEW dengan Arduino yang disimulasikan dengan software Proteus. Diinginkan untuk membuat komunikasi serial yang sama, yaitu antara LabVIEW dengan Arduino, namun dengan menggunakan tampilan Pengisian Tangki. Berikut ini langkah-langkahnya:

1. Buka kembali program Tampilan Pengisian Tangki seperti berikut:



Gambar 2.23 Front Panel program Tampilan Pengisian Tangki

2. Block Diagram program Tampilan Pengisian Tangki:

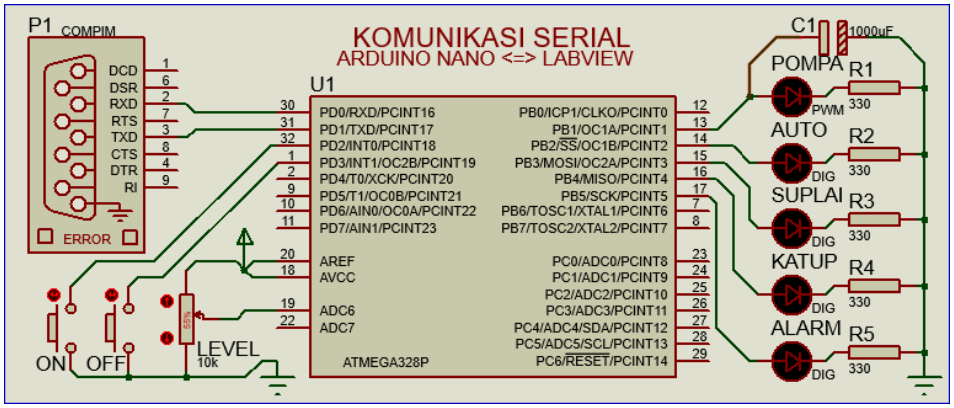


Gambar 2.24 Block Diagram program Tampilan Pengisian Tangki

3. Dengan menambahkan komunikasi serial antara LabVIEW dengan Arduino pada program di atas, berikut ini beberapa hal yang dapat diwujudkan:
   a. Arduino dapat menampilkan kondisi Indikator Suplai, mode Auto, Katup dan Alarm dalam bentuk nyala/padam 4 buah LED.
   b. Arduino dapat menampilkan nilai Slide Pompa (0-1-2), dalam bentuk intensitas cahaya LED.
   c. Tombol di Arduino dapat menghidup/matikan Indikator LED Suplai.

d. LabVIEW dapat menampilkan nilai potensio yang mensimulasikan sensor level untuk mengukur tinggi air dalam Tangki.

4. Untuk memenuhi keempat hal di atas, buat rangkaian di Proteus seperti gambar berikut:



Gambar 2.25 Rangkaian Arduino di Proteus untuk Tampilan Pengisian Tangki

5. Berikutnya, buat program Arduino seperti berikut:

No.	Kode Program 02: Serial_2
01.	`int a0 = 0;`
02.	`int b0 = 0;`
03.	`int h0 = 0;`
04.	`void setup() {`
05.	`  Serial.begin(9600);`
06.	`  pinMode(2, INPUT_PULLUP); //Start=a`
07.	`  pinMode(3, INPUT_PULLUP); //Stop=b`
08.	`  pinMode(10, OUTPUT); //LED Auto=c`
09.	`  pinMode(11, OUTPUT); //LED Suplai=d`
10.	`  pinMode(12, OUTPUT); //LED Katup=e`
11.	`  pinMode(13, OUTPUT); //LED Alarm=f`
12.	`  pinMode(9, OUTPUT); //PWM Pompa=g`
13.	`}`
14.	`void loop() {`
15.	`  int a = !digitalRead(2);//Tombol ON`
16.	`  if (a != a0) {`
17.	`    if (a) Serial.println("a,1");`
18.	`    else Serial.println("a,0");`
19.	`    delay(50);`
20.	`  }`
21.	`  a0 = a;`
22.	`  int b = !digitalRead(3);//Tombol OFF`
23.	`  if (b != b0) {`

```
24.     if (b) Serial.println("b,1");
25.     else Serial.println("b,0");
26.     delay(50);
27.   }
28.   b0 = b;
29.   int h = analogRead(6);//Sensor Level
30.   if (h != h0) {
31.     Serial.print("h,");
32.     Serial.println(h);
33.     delay(50);
34.   }
35.   h0 = h;
36. }
37. void serialEvent() {
38.   while (Serial.available()) {
39.     char data1 = Serial.read();
40.     int data2 = Serial.parseInt();
41.     if (Serial.read() == char(13)) {
42.       switch (data1) {
43.         case 'c':
44.           digitalWrite(10, data2);
45.           break;
46.         case 'd':
47.           digitalWrite(11, data2);
48.           break;
49.         case 'e':
50.           digitalWrite(12, data2);
51.           break;
52.         case 'f':
53.           digitalWrite(13, data2);
54.           break;
55.         case 'g':
56.           analogWrite(9, data2);
57.           break;
58.       }
59.     }
60.   }
61. }
```

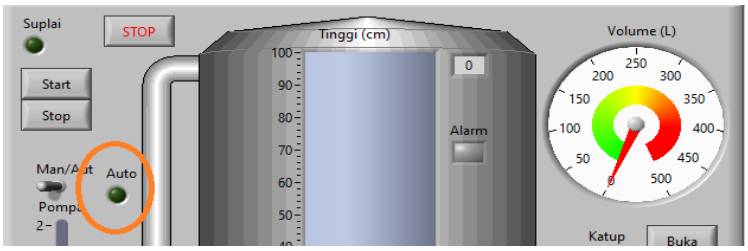
Keterangan program:

1. Baris 1-3: deklarasi variable a0, bo dan h0, yang digunakan untuk mendeteksi apakah ada perubahan kondisi pada tombol ON, OFF dan Potensio (LEVEL).
2. Baris 5-14: blok void setup, yang berisi program untuk seting komunikasi dan penugasan kaki input output.
3. Baris 16-38: blok void loop: berisi program untuk pengiriman data. Data yang dari Tombol ON akan dikirimkan dengan huruf awalan a. Data yang dari

Tombol OFF akan dikirimkan dengan huruf awalan b. Data yang dari Potensio akan dikirimkan dengan huruf awalan b. Data hanya dikirimkan apabila terjadi perubahan data. Tombol ditekan akan mengirimkan nilai 1, tombol dilepas akan mengirimkan 0. Antara data huruf dengan data nilai diberi pemisah koma, dan diakhiri dengan Enter.

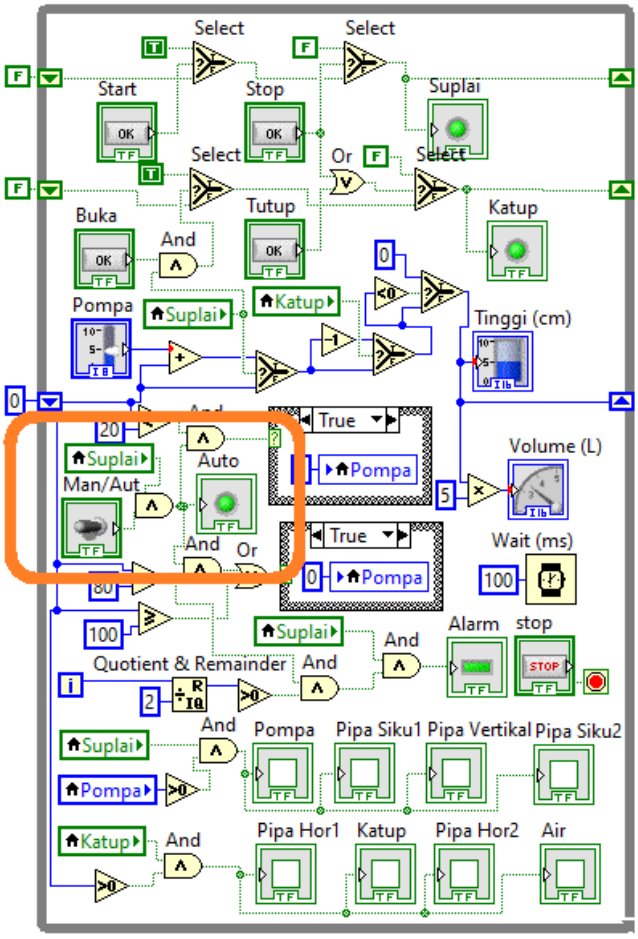
4. Baris 40-64: blok void serialEvent(): berisi program untuk penerimaan data. Berbeda dengan program Arduino sebelumnya, yang menggunakan variable array char, di program Arduino yang kedua ini, data yang diterima ditangkap dengan Serial.read() dan Serial.parseInt(). Serial.read() akan mengambil data huruf awalan, sedangkan Serial.parseInt() akan mengambil data yang berbentuk angka saja, dan langsung diubah ke dalam integer. Berikutnya, untuk meringankan ukuran data, format data antara huruf awalan dan angka, tidak ada koma, dan data diakhiri dengan CR, char(13) saja, tanpa LF.

6. Kompilasi program Arduino di atas, namun sebelumnya, pastikan Board di menu Tools pada pilihan Arduino Nano. Setelah kompilasi selesai, copy lokasi file Hex, dan tempatkan di kolom Program File Edit Properties ATmega328P. Atur juga parameter yang lain pada jendela Edit Properties ATmega328P, seperti ditunjukkan pada Gambar 2.17.
7. Berikutnya, klik 2 kali komponen COMPIM, dan atur parameter di dalamnya seperti Gambar 2.19. Pastikan Physical Port berisi salah satu pasangan COM, dan Baud Rate nya diisi 9600.
8. Kembali ke program LabVIEW (Front Panel) pada Gambar 2.23, tambahkan objek Indicator Round LED, tempatkan di samping Toggle Switch Man/Aut, dan beri nama Round LED Auto.



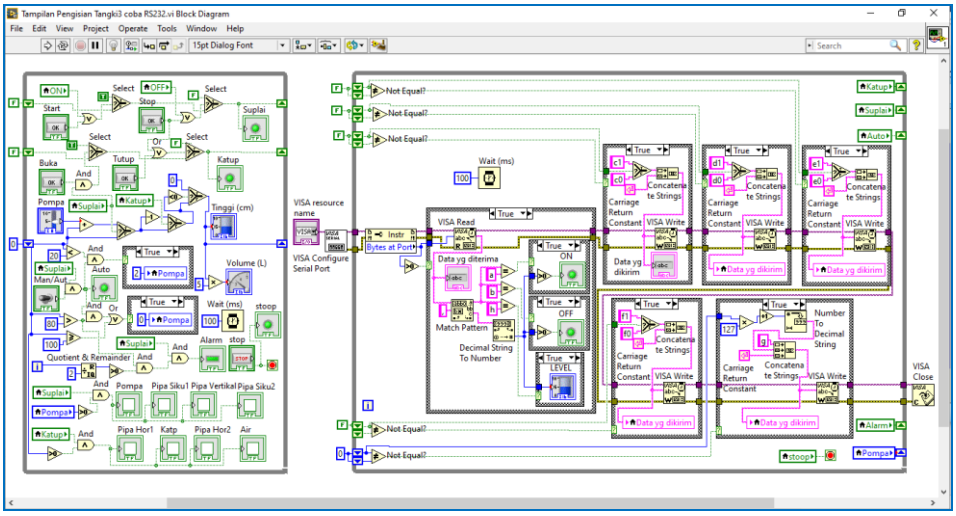
Gambar 2.26 Penambahan objek Indicator Round LED Auto

9. Di Block Diagram, hubungkan icon Auto dengan kaki output icon Toggle Switch Man/Aut. Karena diinginkan LED Auto hanya bisa menyala apabila LED Suplai menyala dan Toggle Switch bernilai True, maka sisipkan fungsi AND pada kaki output Toggle Switch Man/Aut, dan hubungkan kaki AND yang lain dengan Local Variable LED Suplai.



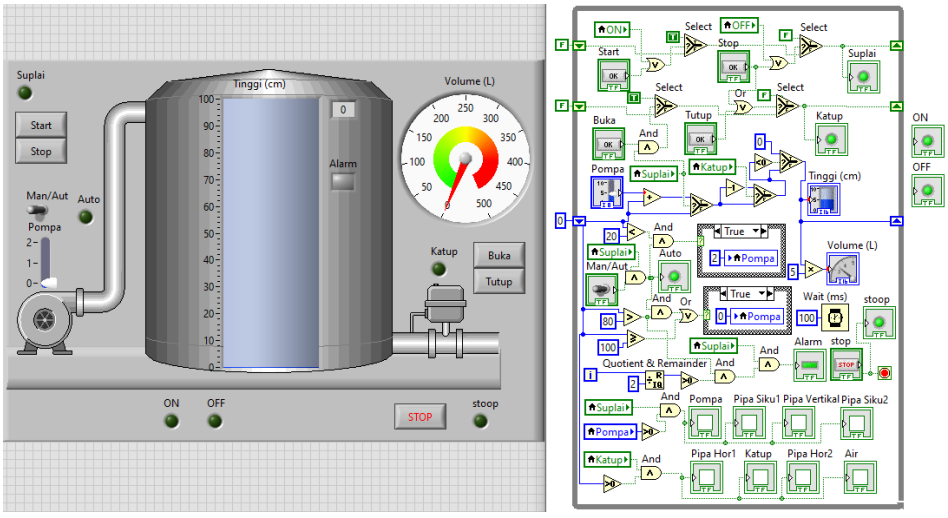
Gambar 2.27 Icon Auto terhubung dengan Toggle Switch Man/Aut dan Local Variable Suplai

10. Program LabVIEW dapat menjalankan beberapa While Loop sekaligus. Untuk itu, di Block Diagram, dapat ditambahkan sebuah While Loop kedua, untuk komunikasi serial dengan Arduino, seperti terlihat pada Gambar 2.28. Bagaimana cara membuatnya, akan dijelaskan lebih lanjut.



Gambar 2.28 Menambahkan While Loop kedua untuk komunikasi serial dengan Arduino

11. Sebelum membuat program seperti Gambar 2.28 di atas, di While Loop pertama (seperti Gambar 2.27), perlu ditambahkan 3 buah icon, yaitu icon Local Variable ON, Local Variable OFF dan icon stoop. Gambar 2.29 berikut menunjukkan tampilan di Front Panel dan Block Diagram setelah penambahan ketiga buah icon tersebut.



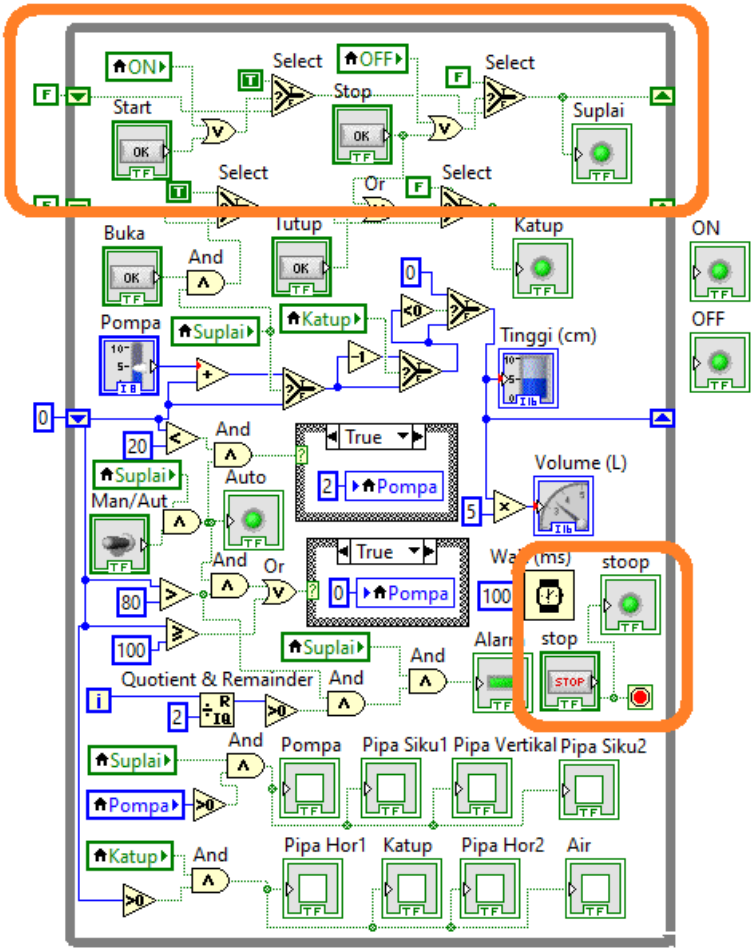
Gambar 2.29 Di bagian bawah Tampilan, ada 3 objek Indicator tambahan (ON, OFF, stoop)

12. Untuk mendapatkan tambahan 3 buah icon tersebut, bisa diambil dari palet Controls di Front Panel. Klik kanan Front Panel, hingga muncul Palet Controls, kemudian ambil objek Round LED dari kategori Boolean, sebanyak 3 buah dan tempatkan di bagian bawah Tampilan Tangki, dan beri nama secara berturut-turut ON, OFF dan stoop. Untuk tombol STOP merah, tombol STOP ini sudah ada dari semula, hanya dipindahkan dari bagian atas Tangki, ke bagian bawah.
13. Dengan penempatan 3 buah objek Round LED di Front Panel, secara otomatis akan muncul 3 buah icon di Block Diagram (ON, OFF dan stoop). Di Block Diagram, tempatkan icon stoop di dalam While Loop, di dekat icon tombol STOP merah. Hubungkan icon stoop ini dengan icon STOP merah, yang terhubung dengan Terminal Loop Condition.

Catatan: Apa fungsi icon stoop ini? Icon stoop ini digunakan untuk mematikan While Loop kedua, yaitu ketika tombol STOP merah ditekan. Jadi di While Loop kedua, Terminal Loop Condition dihubungkan dengan Local Variable icon stoop. Setiap kali tombol STOP ditekan, membuat icon stoop dan Local Variable-nya bernilai True, sehingga tidak hanya While Loop pertama saja yang berhenti, tetapi While Loop kedua juga berhenti.

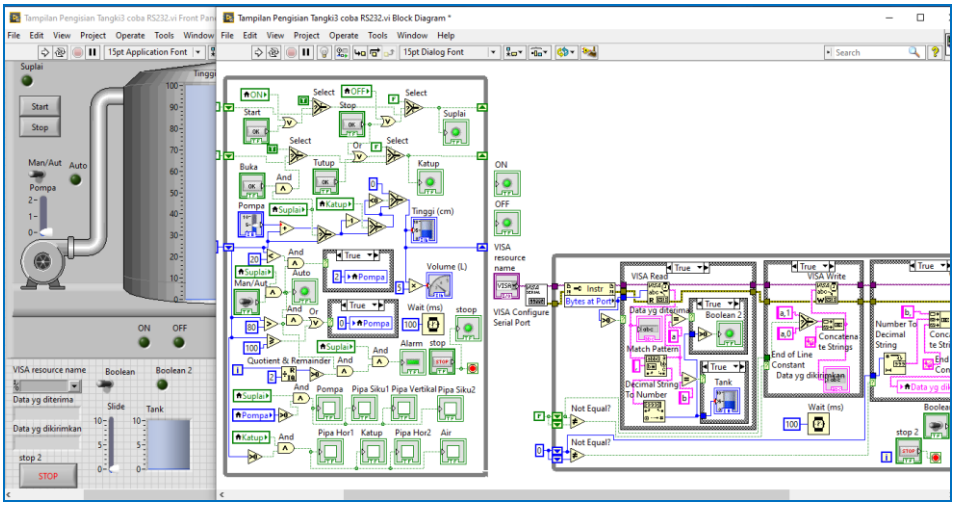
14. Berikutnya, buat Local Variable dari icon LED ON dan icon LED OFF, dan tempatkan keduanya di dekat icon tombol Start dan icon tombol Stop, di dalam While Loop. Cara membuat Local Variable, klik kanan icon, pilih Create, pilih Local Variable. Kemudian buat icon tombol Start di-OR-kan dengan Local Variable LED ON, dan buat icon tombol Stop di-OR-kan dengan Local Variable LED OFF, seperti ditunjukkan pada Gambar 2.30.

Catatan: Apa fungsi Local Variable ON dan OFF ini? Fungsi keduanya untuk membuat agar Indicator LED Suplai tidak hanya bisa di-hidup/mati-kan dari tombol Start dan Stop di LabVIEW, tetapi bisa juga di-hidup/mati-kan dari tombol ON dan OFF di rangkaian Arduino. Untuk itu, agar bisa saling menggantikan, tombol Start harus di-OR-kan dengan Local Variabel LED ON, dan tombol Stop harus di-OR-kan dengan Local Variable LED OFF.



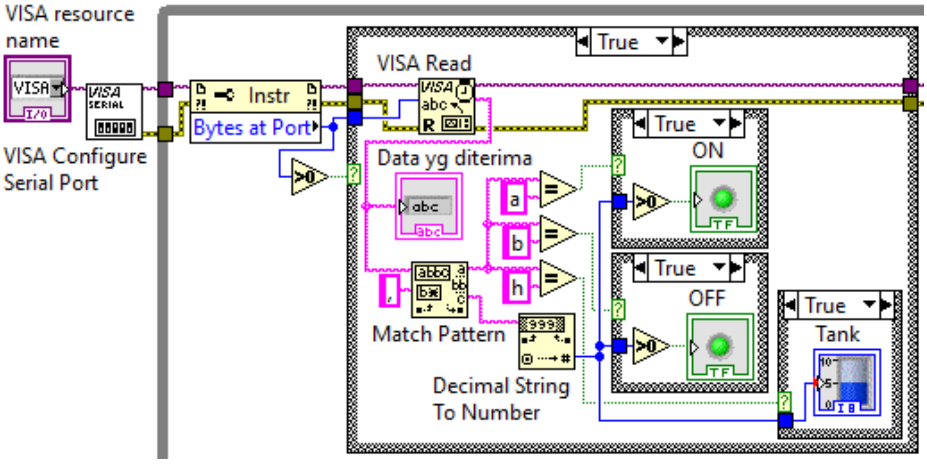
Gambar 2.30 Di While Loop pertama, ada penambahan 3 icon dari Gambar 2.27, yaitu Local Variable ON, OFF dan stoop. Local Variable ON di-OR dengan icon tombol Start, Local Variable OFF di-OR dengan tombol Stop dan icon stoop dihubungkan dengan icon tombol STOP merah.

15. Berikutnya, setelah While Loop pertama selesai, tambahkan While Loop kedua. Buka program LabVIEW Serial1, yang telah dibuat di Sub Bab Komunikasi Serial LabVIEW. Buka Block Diagram program Serial1 tersebut, dan copy semua kode programnya, dan tempelkan di Block Diagram program yang baru ini, di samping kotak While Loop pertama. Gambar 2.31 menunjukkan penambahan While Loop kedua, hasil meng-copy Block Diagram Serial1.



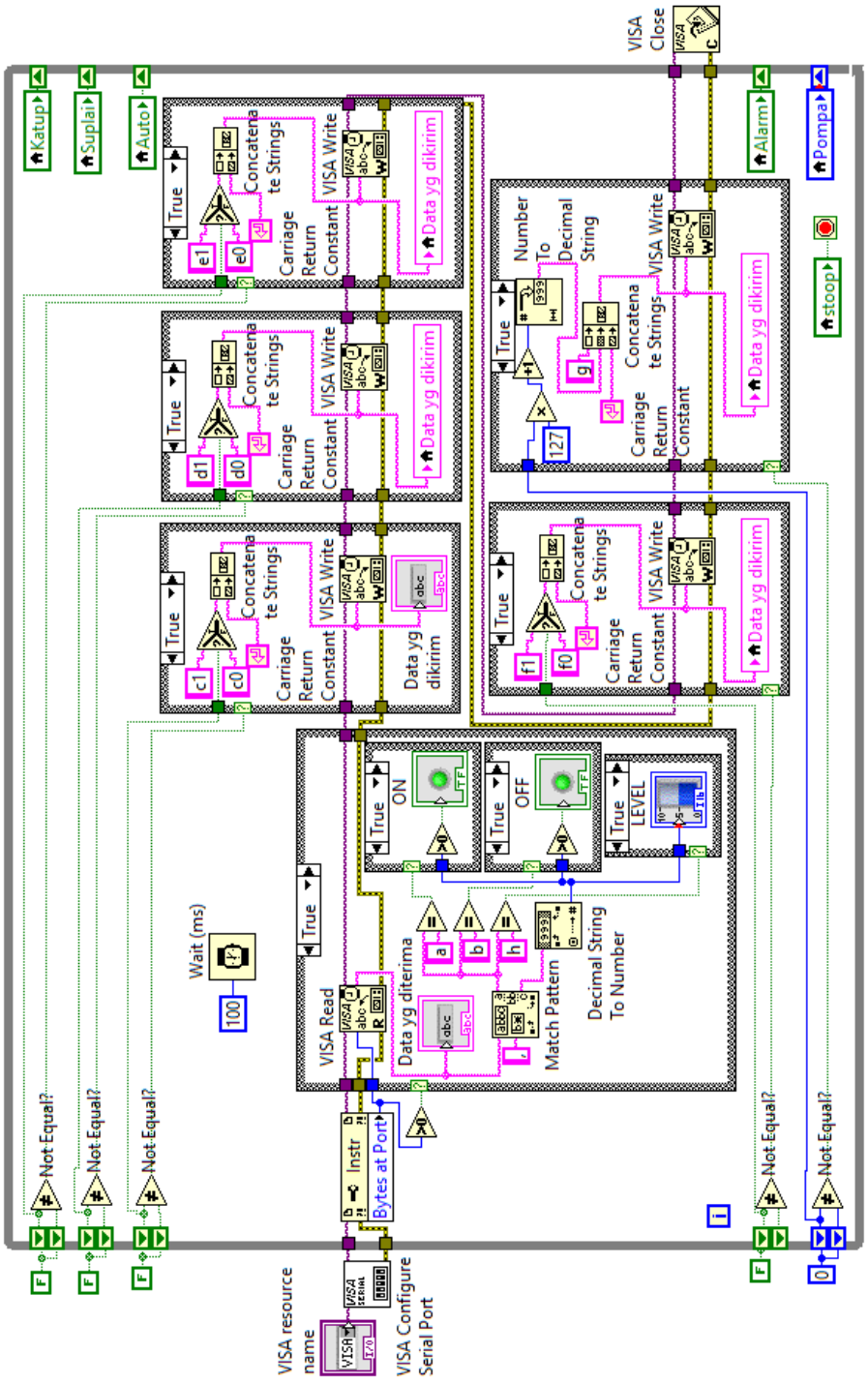
Gambar 2.31 Menambahkan While Loop kedua, hasil meng-copy program Serial1

16. Untuk kotak While Loop kedua, perlu dilakukan modifikasi. Modifikasi mulai dari kotak Struktur Case paling kiri, yang berisi VISA Read. Ada 3 data yang dikirimkan dari Arduino ke LabVIEW, yaitu Tombol ON yang mengirimkan data Boolean dengan huruf awalan “a”, Tombol OFF yang mengirimkan data Boolean dengan huruf awalan “b”, dan Potensio yang mengirimkan data Integer dengan huruf awalan “h”. Untuk itu, modifikasi Struktur Case VISA Read menjadi seperti berikut:



Gambar 2.32 Memodifikasi isi Struktur Case VISA Read yang di-copy dari program Serial1

17. Modifikasi isi Struktur Case VISA Read seperti berikut:
    a. Hapus kotak Struktur Case icon LED Boolean2 beserta isinya.
    b. Tambahkan icon Equal yang terhubung dengan kaki output aa Match Pattern, dan kaki yang lain diberi huruf “h”.
    c. Tambahkan 2 buah kotak Struktur Case. Pada Case True, masing-masing diberi icon LED ON dan icon LED OFF.
    d. Hubungkan Terminal Case LED ON dengan output Equal “a”.
    e. Hubungkan Teminal Case LED OFF dengan output Equal “b”.
    f. Hubungkan Terminal Case Tank dengan output Equal “h”.
18. Berikutnya, modifikasi kotak Struktur Case VISA Write. Ada 5 LED di rangkaian Arduino yang dikontrol dari LabVIEW dengan VISA Write, yaitu:
    a. LED Auto, yang di-hidup/mati-kan dengan huruf awalan “c”.
    b. LED Suplai, yang di-hidup/mati-kan dengan huruf awalan “d”.
    c. LED Katup, yang di-hidup/mati-kan dengan huruf awalan “e”.
    d. LED Alarm, yang di-hidup/mati-kan dengan huruf awalan “f”.
    e. LED Slide Pompa, yang dikontrol intensitas cahayanya dengan nilai 0, 1 dan 2, dengan huruf awalan “g”.
19. Kelima kotak Struktur Case VISA Write tersebut diaktifkan secara berturut-turut oleh Local Variable icon LED Auto, LED Suplai, LED Katup, LED Alarm dan LED Slide Pompa.
20. Hilangkan icon Boolean Toggle Switch dan icon Integer Slide yang di-copy dari program Serial1. Kemudian tambahkan 3 buah Shift Register lagi, sehingga total ada 5 buah Shift Register. Buat input kelima Shift Register tersebut diperoleh dari 5 buah Local Variabel yang mengaktifkan kelima Struktur Case VISA Write di atas. Agar Struktur Case hanya aktif apabila ada perubahan nilai Local Variable, maka munculkan kotak Shift Register di bawahnya, tambahkan fungsi Not Equal, dan hubungkan masing-masing outputnya dengan Terminal Case kelima Struktur Case VISA Write.
21. Untuk meringankan data komunikasi, format data pengiriman dari LabVIEW ke Arduino tidak menggunakan tanda koma dan hanya diakhiri dengan satu karakter akhiran, yaitu Carriage Return (CR), atau char(13). Gambar 2.33 menunjukkan hasil modifikasi ke 5 Struktur Case VISA Write.



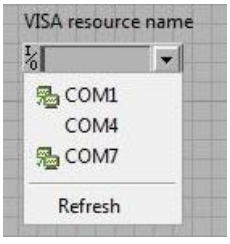
Gambar 2.33 While Loop kedua untuk komunikasi serial yang menghubungkan Tampilan Pengisian Tangki dengan Arduino.

Keterangan program:

1. Perhatikan program LabVIEW pada Gambar 2.33 di atas. Program tersebut adalah program untuk While Loop kedua, yang digunakan untuk berkomunikasi secara serial dengan Arduino.
2. Program di atas secara garis besar terdiri dari 2 bagian; bagian penerimaan, yang menerima data dari Arduino, dikerjakan oleh VISA Read, dan bagian pengiriman, yang mengirimkan data ke Arduino, dikerjakan oleh VISA Write.
3. Agar VISA Read tidak bekerja terus-menerus, ditambahkan Struktur Case dan Bytes at Port, yang dapat mendeteksi apakah ada data di buffer penerimaan port serial. Jika ada data di port serial, Bytes at Port akan mengaktifkan VISA Read, yang akan mengambil data dari port serial. Jika tidak ada data di port serial, VISA Read tidak aktif. Ketika ada data dengan huruf awalan a, Struktur Case LED ON diaktifkan. Apabila data berikutnya adalah angka 1, maka LED ON akan menyala, apabila angka 0, maka LED ON padam. Ketika ada data dengan huruf awalan b, Struktur Case LED OFF diaktifkan. Apabila data berikutnya adalah angka 1, maka LED OFF akan menyala, apabila angka 0, maka LED OFF padam. Ketika ada data dengan huruf awalan h, Struktur Case LEVEL diaktifkan. Data berikutnya akan me-naik/turun-kan tinggi isi LEVEL.
4. Agar VISA Write tidak bekerja terus-menerus, ditambahkan Shift Register dan Case Structure. Ada 5 data yang dikirimkan ke Arduino. Data hanya dikirimkan apabila ada perubahan data. Data dengan huruf awalan c akan mengirimkan kondisi LED Auto ke Arduino. Data dengan huruf awalan d akan mengirimkan kodisi LED Suplai ke Arduino. Data dengan huruf awalan e akan mengirimkan kondisi LED Katup ke Arduino. Data dengan huruf awalan f akan mengirimkan kondisi LED Alarm ke Arduino. Data dengan huruf awalan g akan mengirimkan kondisi Slide Pompa ke Arduino. Perhatikan bahwa untuk menghemat ukuran data, di sini huruf awalan dengan nilainya tidak dipisahkan dengan koma, dan hanya diakhiri dengan karakter Carriage Return (CR).
5. Terakhir, agar While Loop kedua ini bisa berhenti bersamaan dengan While Loop pertama saat tombol STOP merah ditekan, tambahkan Local Variable dari icon LED stoop ke Terminal Loop Condition While Loop kedua. Di While Loop perama, icon LED stoop harus terhubung dengan icon tombol STOP.

22. Di Front Panel, tempatkan kotak VISA Resource Name, kotak Data yang diterima, kotak Data yang dikirim pada baris yang sama dengan LED ON seperti Gambar 2.1 di awal Bab ini.
23. Pilih di kotak VISA Resource Name, nama COM dari salah satu pasangan COM, dalam contoh di sini, digunakan COM3 dan COM4. COM3 untuk COMPIM di Prroteus, dan COM4 untuk VISA Resource Name di LabVIEW.
24. Sebelum dijalankan, klik kanan objek LEVEL, pilih Representation, pilih U16 (*unsigned integer* 16 bit). Mengapa U16? Karena nilai maksimum LEVEL hingga 1023, di mana tipe data U8 tidak lagi memadai.
25. Setelah itu jalankan LabVIEW dan jalankan rangkaian Arduino di Proteus (Gambar 2.25). Perhatikan bahwa Indikator LED Suplai bisa di-hidup/mati-kan tidak hanya dari tombol Start dan Stop di LabVIEW, tetapi bisa juga dari tombol ON dan OFF di Arduino. Geser potensio di Arduino, perhatikan tinggi isi LEVEL. Perhatikan pula kondisi LED Auto, LED Suplai, LED Katup, LED Alarm dan LED Pompa di Arduino, yang kondisinya sama dengan kondisi Control dan Indicator pada Tampilan Pengisian Tangki.
26. Sampai di sini Komunikasi Serial antara Tampilan Pengisian Tangki di LabVIEW dengan Arduino selesai

Catatan: Ketika terjadi error pada komunikasi serial, maka akan muncul tanda error di samping nama port, seperti ditunjukkan pada Gambar 2.34 berikut ini. Cara untuk menghilangkan error ini adalah dengan menutup semua program LabVIEW, kemudian membukanya lagi, maka tanda error akan menghilang.



Gambar 2.34 Ketika komunikasi serial error, akan muncul tanda di samping nama port. Hilangkan error ini dengan menutup semua program LabVIEW, kemudian buka kembali.

2.5 Soal Latihan

1. Komunikasi serial antara LabVIEW dengan Arduino dalam Bab ini menggunakan data ASCII. Apa yang Anda ketahui tentang ASCII? Apa saja karakter ASCII yang tidak tercetak, dan karakter ASCII yang tercetak?

2. Perhatikan kode program Arduino Serial_1 di halaman 55-56. Apa icon-icon fungsi komunikasi serial di LabVIEW yang mirip dengan instruksi berikut ini: (a). Serial.begin()? (b). Serial.available()? (c). Serial.read()? (d). Serial.print()?

3. Pada baris 40-41 dari kode program Arduino Serial_1, mengapa pesan[0] dan pesan[1] harus diisi dengan data char '0'?

4. Pada baris 39 dari kode program Arduino Serial_1, mengapa isi dari pesan[0] diisikan ke variable pesan0?

5. Dari kode program Arduino Serial_2 di halaman 63-64, apa isi variabel data1 dan data2 apabila LED Katup di Tampilan Pengisian Tangki, berubah dari kondisi menyala menjadi padam.

6. Di Proteus, dengan rangkaian Arduino yang sama seperti Gambar 2.22, tambahkan sebuah komponen LCD 16x2 (LM106L). Buat agar data yang dikirimkan dari program Serial1 LabVIEW ke Arduino dapat ditampilkan di baris pertama, sedangkan data yang dikirim dari Arduino ke LabVIEW dapat ditampilkan di baris kedua.

7. Pengggunaan LCD 16x2 membutuhkan minimal 7 kaki I/O Arduino. Untuk menghemat kaki I/O Arduino, buat agar LCD 16x2 menjadi LCD I2C, dengan menambahkan IC PCF8754 ke rangkaian LCD, dan menambahkan library Wire.h ke program Arduino. Buat hal yang sama seperti Soal no. 6, yaitu baris pertama LCD menampilkan data yang dikirimkan dari LabVIEW ke Arduino, dan baris kedua menampilkan data yang dikirimkan dari Arduino ke LabVIEW.

8. Ulangi Soal no. 7 di atas untuk rangkaian Arduino di Proteus yang sama seperti Gambar 2.25, dengan program LabVIEW Tampilan Pengisian Tangki Serial.

9. Ulangi Soal no. 8, dan buat agar Tinggi air Tangki dapat ditampilkan di baris pertama di LCD 16x2, di samping data yang dikirimkan dari LabVIEW ke Arduno, diberi pemisah tanda koma.

10. Untuk mengetahui apakah simulasi Proteus sama seperti kenyataan, ulangi Soal no. 9, namun dengan menggunakan hardware Arduino Nano secara riil, begitu pula dengan komponen-komponen lainnya, termasuk 2 buah Tombol, sebuah Potensio 10 kohm, 5 buah LED, 5 buah resistor 330 ohm dan LCD I2C (LCD 16x2 dengan IC PCF8574). Di kotak VISA Resouce Name LabVIEW, gunakan nama COM yang muncul saat USB Arduino Nano dihubungkan ke komputer. Jalankan program LabVIEW, dan amati kondisi input output rangkaian Arduino dan juga objek-objek di Tampilan Pengisian Tangki.

2.5 Refleksi

1. Menurut Anda, dari isi yang diuraikan, apakah **Target Materi** dari Bab 2 buku ini tercapai? Jika belum tercapai, apakah ada kesulitan dalam memahami materi yang berkaitan dengan Target Materi tersebut? Apakah Anda memiliki saran dan masukan untuk memperbaiki materi tersebut?

2. Apakah **Tantangan** dalam Bab 2 buku ini dapat Anda selesaikan? Jika belum, kesulitan apa yang membuat Anda tidak bisa menyelesaikannya? Apakah Anda mencoba alternatif lain untuk menemukan sendiri cara penyelesaiannya (mencari di Google misalnya)?

3. Apakah ada **Manfaat** yang Anda dapatkan setelah mempelajari Bab 2 dari buku ini? Apakah ada yang ingin Anda pelajari lebih lanjut? Apakah ada **Ide** yang menarik yang ingin Anda kembangkan?

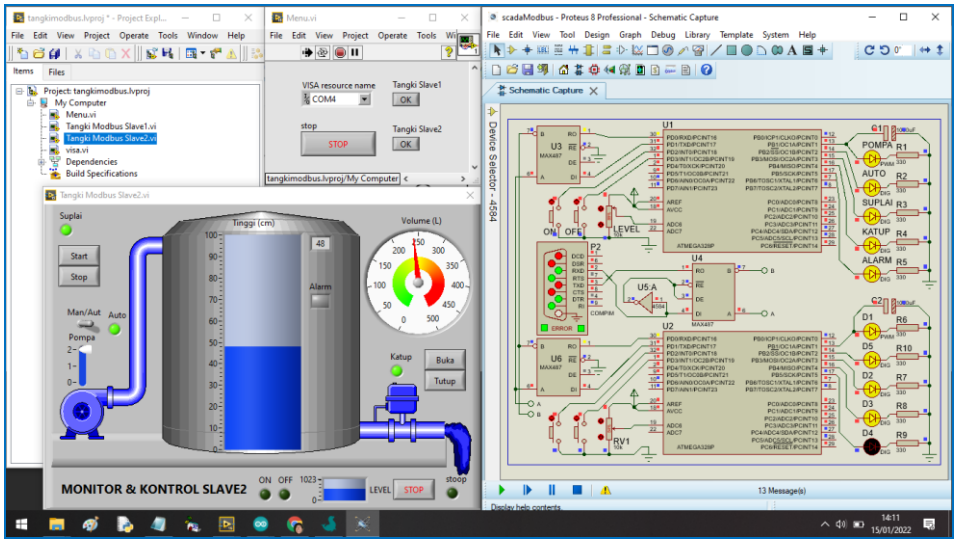
BAB 3
PROTOKOL MODBUS RTU

Target Materi:

- Menerapkan protokol Modbus antara LabVIEW dengan Arduino
- Menerapkan protokol Modbus antara LabVIEW dengan Outseal

Tantangan:

➢ Pembaca dapat membuat komunikasi serial RS-485 dengan protokol Modbus RTU untuk menghubungkan Tampilan SCADA di LabVIEW dengan 2 buah rangkaian Arduino yang disimulasikan di Proteus.



Gambar 3.1 Komunikasi serial RS-485 dengan protokol Modbus RTU antara Tampilan SCADA di LabVIEW dengan 2 Arduino yang disimulasikan di Proteus

3.1 Persiapan Perangkat yang Diperlukan

Di samping menggunakan perangkat lunak di Bab sebelumnya, Bab ini memerlukan tambahan perangkat lunak/software sebagai berikut:

1. Software VIPM versi 2018 ke atas.
2. Library Modbus NI (ni_lib_modbus_library)
3. Software QModMaster
4. Software ModRSsim2
5. Outseal Studio V2.5

Software VIPM (VI Package Manager) adalah software yang digunakan untuk menginstal library tambahan pada LabVIEW. Software VIPM ini mirip seperti Google Playstore pada Android, di mana pembaca dapat menemukan daftar library yang tersedia, yang bisa didownload dan diinstal. Sebenarnya setiap kali menginstal software LabVIEW, software VIPM juga terinstal. Hanya saja, karena software LabVIEW yang penulis instal adalah versi 2015, sementara Library Modbus NI ini hanya bisa diinstal oleh software VIPM versi 2018 ke atas, maka bagi pembaca yang VIPM nya masih 2015, silahkan menginstal software VIPM ini.

Library Modbus NI adalah library tambahan untuk pemrograman Modbus di LabVIEW. Library ini dapat diinstal secara offline maupun online dengan software VIPM versi 2018 ke atas. Instalasi secara online memerlukan koneksi dengan internet. Ketika komputer terkoneksi internet, software VIPM akan menampilkan daftar library yang dapat diinstal, termasuk NI Modbus Library. Klik 2 kali nama library tersebut, dan pilih Instal, maka proses instalasi secara online akan berjalan. Apabila komputer tidak terhubung dengan internet, buka software VIPM, pilih Open Package File(s) di menu File, dan arahkan pencarian ke file ni_lib_modbus_library, klik OK, maka proses instalasi secara offline akan berjalan.

Software QmodMaster dan ModRSsim2 adalah software simulator untuk penerapan protokol Modbus, baik untuk Modbus RTU (serial) maupun Modbus TCP. Software QmodMaster untuk mensimulasikan Modbus Master, sedangkan ModRSsim2 untuk mensimulasikan Modbus Slave. Lebih jauh mengenai Master dan Slave pada Modbus akan dijelaskan di Sub Bab Protokol Modbus.

Software Outseal adalah software pemrograman untuk Arduino, dengan bahasa pemrograman berbentuk Ladder Diagram, mirip seperti pemrograman pada PLC. Hal yang menarik dari software Outseal ini adalah, fungsi-fungsi Modbus dapat diterapkan dengan mudah, baik untuk dijadikan Master maupun Slave, dan software ini adalah karya anak bangsa sendiri, yang dapat didownload secara gratis di alamat: https://www.outseal.com/site/download.html. Hanya saja versi software Outseal di alamat tersebut adalah versi 3.2, sedangkan yang penulis pakai dalam buku ini adalah versi 2.5.

Agar memudahkan pembaca, pembaca dapat mendownload semua software yang diperlukan di atas, di halaman blog penulis, yaitu di: dianartanto@blogspot.com.

3.2 Protokol Modbus

Setelah semua software yang diperlukan di atas berhasil diinstal, maka sebelum menerapkan pemrograman Modbus di LabVIEW, pembaca perlu mengetahui beberapa hal mengenai protokol Modbus seperti berikut ini:

1. Komunikasi serial dengan sambungan RS-232, seperti yang telah diuraikan di Bab 2, hanya dapat diterapkan untuk komunikasi antar 2 alat saja, dengan jarak kabel yang pendek, yaitu kurang dari 15 meter. Sedangkan sambungan RS-485, yang akan diuraikan di Bab 3 ini, dapat diterapkan untuk komunikasi hingga 32 alat, dengan jarak kabel bisa mencapai 1,2 km. Namun demikian, apabila komunikasi RS-232 bisa Full Duplex, yaitu bisa mengirim dan menerima pada waktu bersamaan, komunikasi RS-485 ini hanya bisa Half Duplex, yaitu mengirim dan menerima secara bergantian, tidak bisa melakukan keduanya pada waktu yang bersamaan.
2. Protokol Modbus telah dikembangkan sejak tahun 1979, yang semula hanya digunakan untuk menghubungkan perangkat melalui sambungan serial (sambungan RS-232, RS-422, RS-485), kemudian meningkat hingga mencakup sambungan melalui TCP/IP, dan UDP. Saat ini, protokol Modbus telah menjadi protokol umum yang digunakan oleh banyak perangkat untuk komunikasi yang sederhana, andal, dan efisien di berbagai jaringan modern.

3. Protokol Modbus merupakan protokol request - respon. Protokol Modbus hanya akan bekerja ketika ada permintaan (request) dari Master. Slave tidak dapat menginisiasi komunikasi, hanya Master yang bisa memulai komunikasi. Setiap permintaan Master harus direspon oleh Slave, apabila tidak ada respon dari Slave, maka Master akan memberikan indikasi error, dan terus-menerus mengirim permintaan hingga Slave merespon.
4. Ada 3 tipe Modbus, yaitu Modbus ASCII, Modbus RTU dan Modbus TCP. Modbus ASCII dan RTU sama-sama digunakan untuk sambungan serial, hanya bedanya, Modbus ASCII menggunakan data dengan karakter ASCII, sedangkan Modbus RTU menggunakan biner, yang membuat datanya menjadi lebih kecil ukurannya dibandingkan Modbus ASCII. Modbus TCP menggunakan format data yang lebih baik, yang memungkinkan penanganan request - respon menjadi lebih efisien, yang digunakan pada jaringan Ethernet.
5. Ada 4 tipe data pada protokol Modbus, seperti terlihat pada tabel berikut:

Tabel 3.1 Tipe Data pada Protokol Modbus

No	Blok Memori	Jangkauan alamat	Tipe Data	Akses data
1	Coil	000000 – 065535	Boolean	Read/Write
2	Input Diskrit	100000 – 165535	Boolean	Read Only
3	Input Register	300000 – 365535	16 bit	Read Only
4	Holding Register	400000 – 465535	16 bit	Read/Write

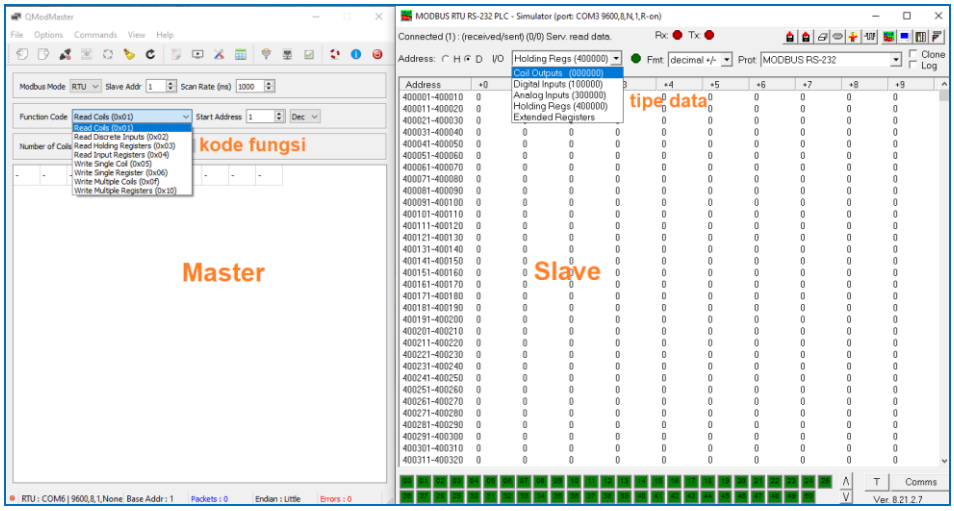
6. Secara umum, ada 8 kode Fungsi request yang dikirimkan Master ke Slave.

Tabel 3.2 Kode Fungsi pada Protokol Modbus

No	Kode Fungsi (Hex)	Keterangan
1	0x01	Read Coils
2	0x02	Read Discrete Inputs
3	0x03	Read Holding Registers
4	0x04	Read Input Registers

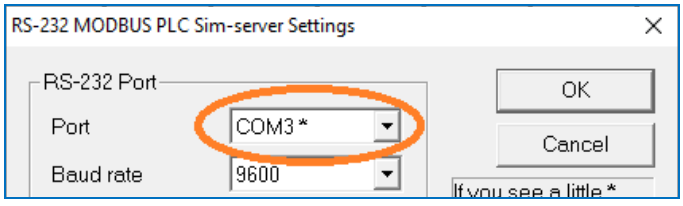
No	Kode Fungsi (Hex)	Keterangan
5	0x05	Write Single Coil
6	0x06	Write Single Register
7	0x0F	Write Multiple Coils
8	0x10	Write Multiple Registers

7. Untuk memahami lebih dalam mengenai Protokol Modbus, buka software QmodMaster dan ModRSsim2 seperti ditunjukkan pada gambar berikut ini:



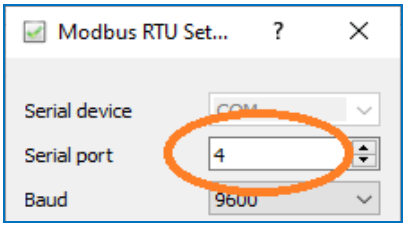
Gambar 3.2 QmodMaster sebagai Modbus Master dan ModRSsim2 sebagai Modbus Slave

8. Pada software ModRSsim2, pilih Protokol (Prot): Modbus RS-232. Kemudian tekan tombol Setup (bergambar konektor DB9, tombol keempat dari kiri), dan atur Port dengan salah satu COM yang berpasangan, seperti berikut:



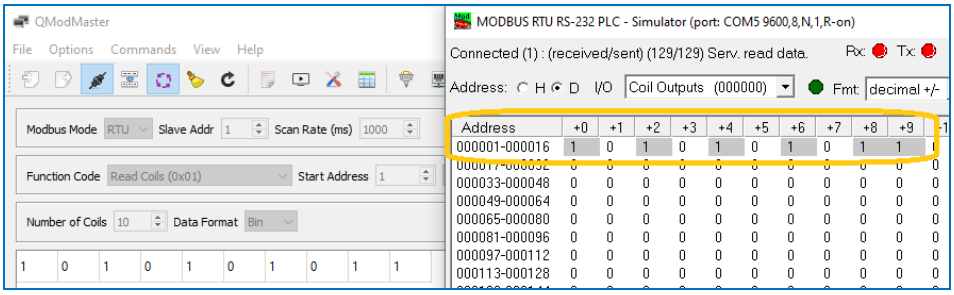
Gambar 3.3 Atur Port pada tombol setup dengan salah satu COM yang berpasangan

9. Berikutnya, pada software QModMaster, pilih Modbus Mode: RTU. Kemudian tekan tombol Modbus RTU di Toolbar (tombol keenam dari kanan), dan atur Serial Port dengan COM pasangan dari COM yang digunakan di ModRSsim2.



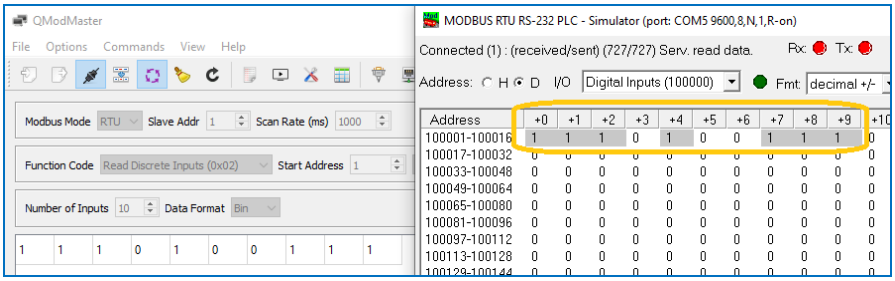
Gambar 3.4 Atur Serial Port di Mobus RTU dengan COM pasangan dari COM di ModRSsim2

10. Berikutnya, di software QmodMaster, tekan tombol Connect (bergambar kontak yang disambungkan, tombol ketiga dari kiri). Apabila tidak muncul error, berarti koneksi QModMaster dengan ModRSsim2 telah berhasil.
11. Di software QModMaster, pilih Function Code: Read Coils (0x01), Start Address: 1, Number of Coils: 10. Tekan tombol Scan (tombol kelima dari kiri), maka QmodMaster akan menampilkan hasil pembacaan data Coil di ModRSsim2, dari 000001 sampai 000010. Lakukan pengubahan data di ModRSsim2 di alamat 000001 sampai 000010 itu, dan perhatikan bahwa data Coils di ModRSsim2 dapat dibaca dan ditampilkan di QmodMaster.



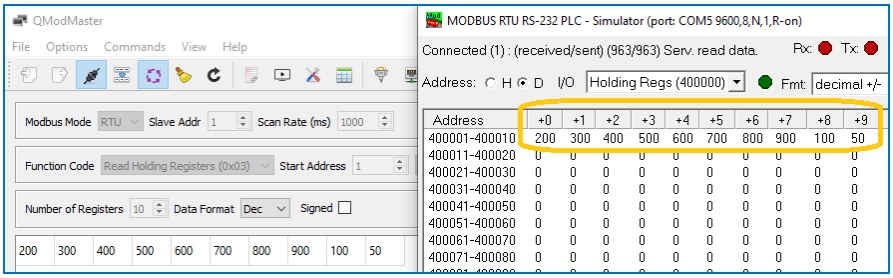
Gambar 3.5 Data Coil Outputs di ModRSsim2 dapat dibaca dan ditampilkan di QModMaster

12. Di QModMaster, pilih Function Code: Read Discrete Inputs (0x02), Start Address: 1, Number of Coils: 10. Kemudian tekan tombol Scan, maka QmodMaster akan menampilkan hasil pembacaan data Input Diskrit (Digital) dari ModRSsim2, dengan alamat mulai dari 100001 sampai 100010.



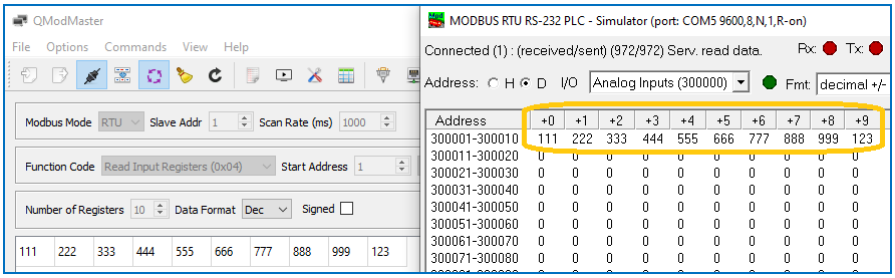
Gambar 3.6 Data Input Digital di ModRSsim2 dapat dibaca dan ditampilkan di QModMaster

13. Berikutnya di QModMaster, pilih Function Code: Read Holding Registers (0x03), Start Address: 1, Number of Coils: 10, dan kemudian tekan tombol Scan, maka QmodMaster akan menampilkan hasil pembacaan data Holding Registers dari ModRSsim2, dengan alamat mulai dari 400001 sampai 400010.



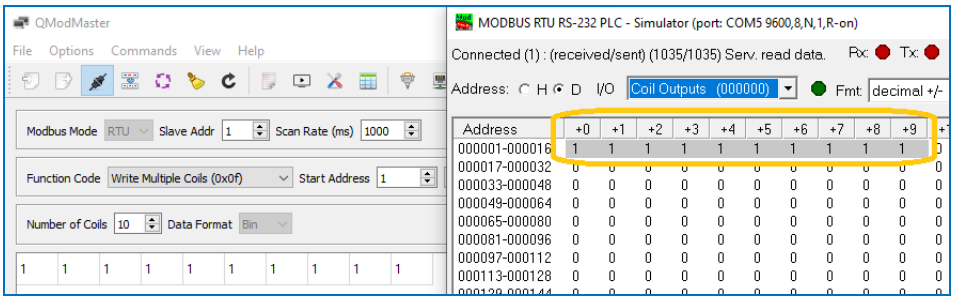
Gambar 3.7 Data Holding Registers di ModRSsim2 dapat dibaca ditampilkan di QModMaster

14. Berikutnya di QModMaster, pilih Function Code: Read Input Registers (0x04), Start Address: 1, Number of Coils: 10, dan kemudian tekan tombol Scan, maka QmodMaster akan menampilkan hasil pembacaan data Input Register (Analog) dari ModRSsim2, dengan alamat mulai dari 300001 sampai 300010.



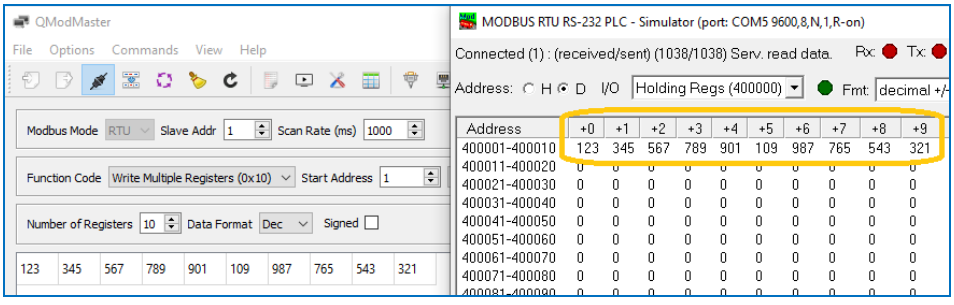
Gambar 3.8 Data Input Analog di ModRSsim2 dapat dibaca dan ditampilkan di QModMaster

15. Berikutnya di QModMaster, pilih Function Code: Write Multiple Coils (0x0F), dengan Start Address: 1, Number of Coils: 10. Lakukan pengaturan nilai pada kotak data di QmodeMaster, sebagai contoh buat semua bernilai 1. Tekan tombol Read/Write (tombol keempat dari kiri), maka nilai Coils di ModRSsim2 di alamat 000001 - 000010 akan berubah sesuai dengan data di QModMaster.



Gambar 3.9 Data Coil Outputs di ModRSsim2 dapat diatur dari QModMaster

16. Berikutnya di QModMaster, pilih Function Code: Write Multiple Registers (0x10), dengan Start Address: 1, Number of Coils: 10. Lakukan pengaturan nilai pada kotak data di QModeMaster. Tekan tombol Read/Write (tombol keempat dari kiri), maka nilai Holding Registers di ModRSsim2 di alamat 000001 - 000010 akan berubah sesuai dengan nilai data di QModMaster.



Gambar 3.10 Data Holding Registers di ModRSsim2 dapat diatur dari QModMaster

17. Dari Gambar 3.2 sampai 3.10 di atas, pembaca dapat melihat bagaimana protokol Modbus diterapkan. Master yang disimulasikan oleh QModMaster, selalu menginisiasi komunikasi dengan memberikan kode fungsi dan alamat yang dituju ke Slave, yang disimulasikan oleh ModRSsim2.

18. Jadi secara umum protokol Modbus memiliki 2 kode fungsi, yaitu fungsi Read dan Write. Fungsi Read membuat Master dapat mengambil data dari Slave, sedangkan fungsi Write membuat Master dapat mengubah data di Slave. Ada 4 jenis fungsi Read, sesuai dengan jumlah tipe data pada protokol Modbus, satu tipe data satu fungsi Read. Sedangkan jenis fungsi Write hanya 2, karena tipe data yang bisa di-Write hanya ada 2, yaitu Coil dan Holding Register.
19. Sampai di sini pengenalan protokol Modbus telah selesai. Sub Bab berikut menunjukkan langkah-langkah pembuatan Modbus Master dengan LabVIEW, dan Modbus Slave dengan Arduino.

3.3 Protokol Modbus pada LabVIEW dan Arduino

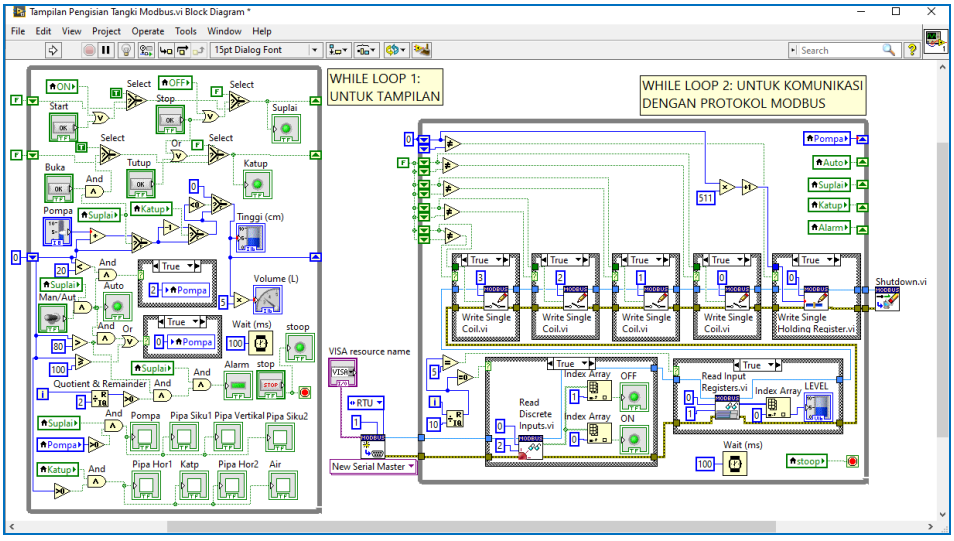
Setelah pada Sub Bab 3.2 di atas, diberikan gambaran mengenai penerapan protokol Modbus, dengan software simulasi QModMaster dan ModRSsim2, maka pada Sub Bab 3.3 ini, akan ditunjukkan bagaimana membuat LabVIEW sebagai Master dan Arduino sebagai Slave, menggantikan QModMaster dan ModRSsim2. Pertimbangan LabVIEW dijadikan Master di sini yaitu agar ada tampilan komunikasi yang menarik dengan objek-objek yang interaktif. Pertimbangan Arduino dijadikan Slave, agar bisa menunjukkan ketersambungan banyak perangkat yang bisa terhubung dengan protokol Modbus.

Di Sub Bab 3.3 ini, sebagai awalan, akan ditunjukkan bagaimana menghubungkan sebuah Arduino sebagai Slave dengan LabVIEW sebagai Master, dengan protokol Modbus dan sambungan RS-232. Di Sub Bab 3.4 barulah akan ditunjukkan bagaimana menghubungkan 2 buah Arduino sebagai Slave dengan LabVIEW sebagai Master dengan protokol Modbus dan sambungan RS-485.

Berikut ini langkah-langkah menghubungkan LabVIEW sebagai Master dengan Arduino sebagai Slave, menggunakan protokol Modbus dan sambungan RS-232:

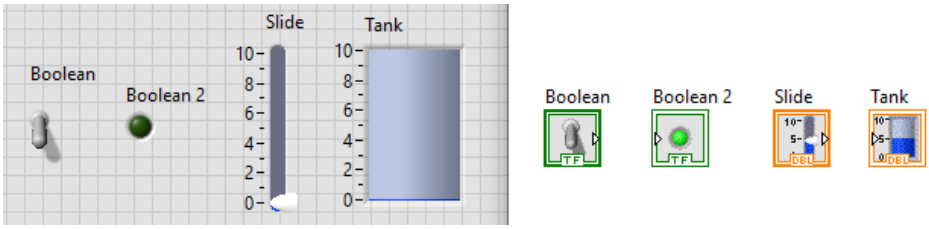
1. Buka kembali program LabVIEW Tampilan Pengisian Tangki dengan komunikasi serial yang telah dibuat di Sub Bab 2.4. Ada 2 buah While Loop pada program LabVIEW tersebut. While Loop pertama berfungsi untuk mengatur tampilan (Gambar 2.30), sedangkan While Loop kedua berfungsi untuk menjalankan komunikasi serial UART (Gambar 2.33).

2. Save As program LabVIEW yang telah dibuat di Sub Bab 2.4 tersebut dengan nama “Tampilan Pengisian Tangki Modbus”.
3. Setelah Save As berhasil, hapus While Loop kedua yang menjalankan komunikasi serial UART, sementara While Loop pertama yang mengatur tampilan tetap digunakan. While Loop kedua dihapus agar dapat diganti dengan While Loop yang menjalankan Modbus Master, seperti berikut:



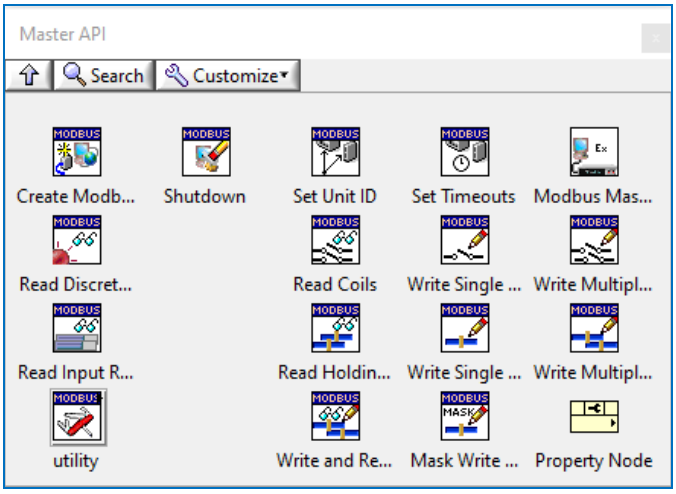
Gambar 3.11 Program LabVIEW Tampilan Pengisian Tangki dengan Protokol Modbus

4. Untuk bisa memahami program LabVIEW pada Gambar 3.11 di atas, pembaca perlu memahami program modbus LabVIEW yang lebih sederhana terlebih dulu. Untuk itu buka sebuah program baru (new vi), dan buat tampilan empat buah objek di Front Panel, yang terdiri dari Toggle Switch, Round LED, Vertical Pointer Slide dan Tank, seperti berikut:



Gambar 3.12 Empat buah objek: Toggle Switch, Round LED, Vertical Pointer Slide dan Tank

5. Diinginkan keempat objek di LabVIEW tersebut dapat melakukan:
   a. Toggle Switch dapat menyalakan/memadamkan LED di Arduino.
   b. Round LED dapat membaca kondisi tombol di Arduino, apabila ditekan, Round LED menyala, bila tidak ditekan, Round LED padam.
   c. Pointer Slide dapat mengatur intensitas cahaya LED di Arduino.
   d. Objek Tank dapat membaca nilai analog dari Potensio di Arduino.
6. Untuk membuat keempat hal di atas, di Block Diagram, ambil While Loop, dan masukkan keempat icon objek tersebut ke dalam While Loop.
7. Tambahkan tombol Stop pada While Loop, dengan meng-klik kanan icon Terminal Loop Condition, dan pilih Create, pilih Control.
8. Tambahkan juga icon Wait(ms), dan isi dengan nilai 100.
9. Berikutnya, klik kanan di Block Diagram hingga palet Functions muncul. Di palet Functions, pilih Data Communication, pilih Modbus Library, pilih Master API, maka akan muncul kotak fungsi Master API seperti berikut:

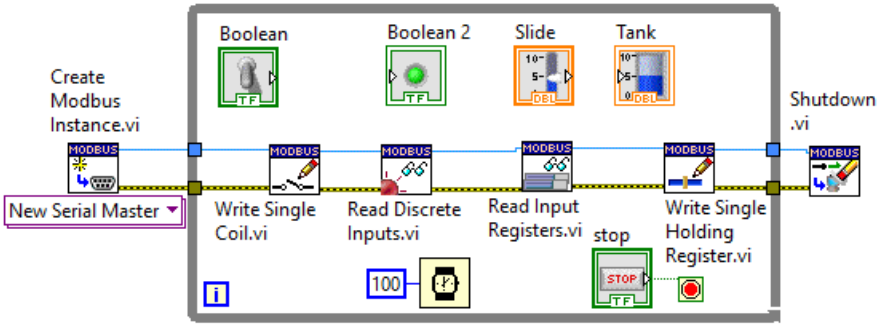


Gambar 3.13 Fungsi-fungsi Master API di Block Diagram LabVIEW

Catatan: Fungsi Create Modbus digunakan untuk mengatur komunikasi Modbus, mencakup tipe Modbus yang digunakan (Master atau Slave, Serial atau TCP, Serial ASCII atau RTU), saluran port yang digunakan, ID Slave yang dituju, baudrate, dll. Fungsi Create Modbus ini hanya perlu dijalankan sekali di awal, untuk itu tempatkan fungsi ini di luar While Loop.

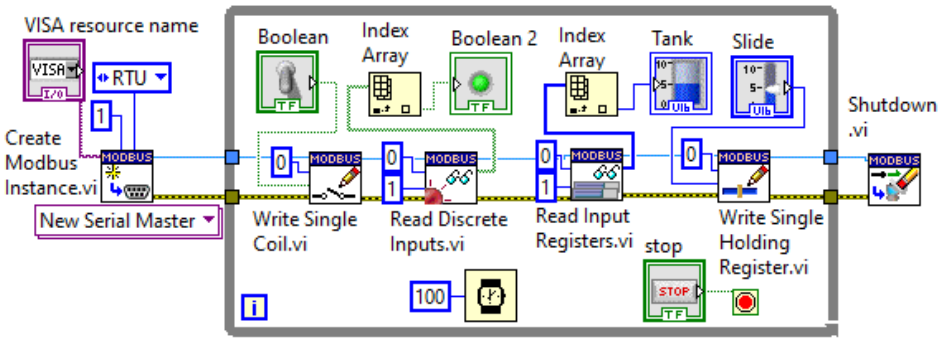
Begitu pula fungsi Shutdown, tempatkan fungsi ini di luar While Loop karena hanya perlu dijalankan sekali di akhir program, yang digunakan untuk menutup komunikasi modbus. Sedangkan fungsi-fungsi Read dan Write perlu dijalankan berulang-kali, untuk itu tempatkan fungsi-fungsi Read Write ini didalam While Loop.

10. Dari kotak Master API, ambil Create Modbus, Write Single Coil, Read Discrete Input, Read Input Register, Write Single Register dan Shutdown, tempatkan dan hubungkan seperti gambar berikut:



Gambar 3.14 Tempatkan dan hubungkan fungsi-fungsi Modbus Master seperti gambar

11. Atur setiap fungsi dan beri input dengan cara meng-klik kaki input tersebut dan pilih Create, pilih Constant. Khusus untuk kaki input VISA Resource name di Create Modbus, gunakan Create Control, agar pilihan port COM bisa dimasukkan oleh pengguna dari Front Panel.

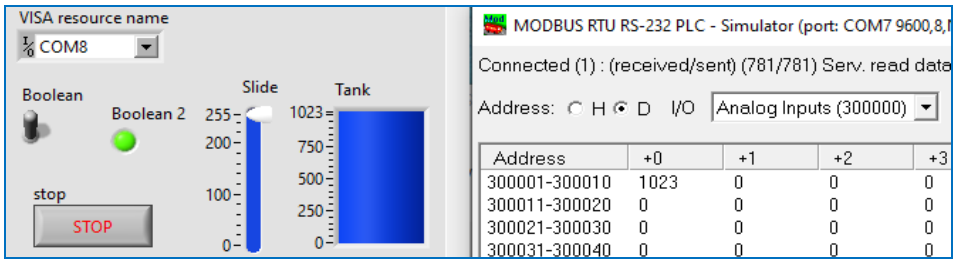


Gambar 3.15 Atur setiap fungsi dan beri input sesuai dengan yang diperlukan

Catatan: Di Create Modbus, pilih tipe Modbus New Serial Master, unit ID = 1, serial type = RTU. Untuk semua fungsi Read dan Write, beri input address = 0. Mengapa diberi 0? Karena alamat yang digunakan di LabVIEW mengacu pada indeks dalam pemrograman, yang selalu dimulai dari angka 0. Jadi address 0 ini mengacu pada alamat memori 1. Untuk Coil, berarti alamat 000001, untuk Input Diskrit, berarti alamat 100001, untuk Input Register berarti alamat 300001, dan untuk Holding Register berarti alamat 400001.

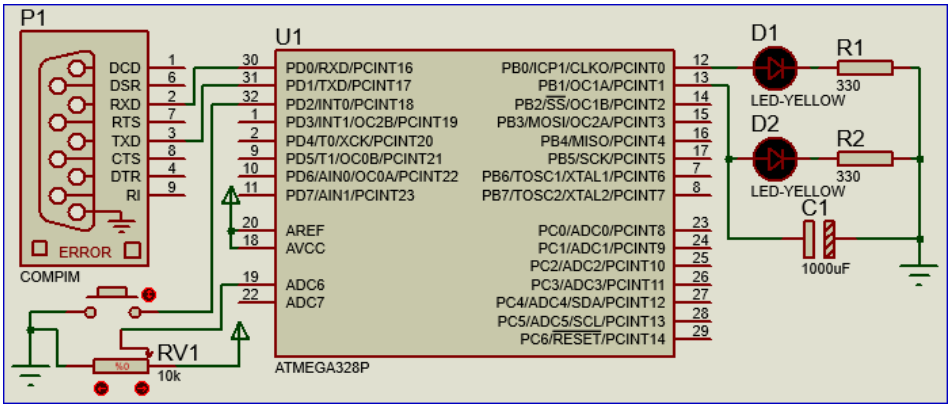
Berikutnya, hubungkan Toggle Switch ke input coil to write, dan Pointer Slide ke input register to write. Hubungkan Round LED dengan output fungsi Read Discrete Inputs. Karena output fungsi ini berupa array, tambahkan fungsi Index Array untuk mengambil satu nilai saja. Kaki input index Index Array tidak diisi tidak masalah, karena secara default input index ini adalah 0, yang berarti mengambil data yang pertama. Begitu pula dengan icon Tank, hubungkan icon ini dengan output fungsi Read Input Registers, dengan menyisipkan Index Array. Agar data yang digunakan tidak pecahan, ubah tipe data Pointer Slide dan Tank, dari yang semula tipe double (DBL), ubah menjadi tipe unsigned interger 16 bit (U16), dengan cara meng-klik kanan icon, pilih Representation, pilih U16, maka garis data yang semula berwarna oranye, akan berubah menjadi biru. Kemudian ubah skala maksimum Pointer Slide dari 10 menjadi 255, dan skala maksimum Tank dari 10 menjadi 1023.

12. Jalankan program LabVIEW Gambar 3.15 di atas. Atur VISA Resource Name dengan salah satu port COM yang berpasangan. Kemudian jalankan software ModRSsim2. Pilih Prot: Modbus RS-232, dan atur Port dengan port COM yang berpasangan dengan port COM di VISA LabVIEW.



Gambar 3.16 Program Modbus Master LabVIEW mengatur dan membaca data di ModRSsim2

13. Perhatikan bahwa setiap kali tuas Toggle Switch dan Pointer Slide di LabVIEW dinaik-turunkan, data di alamat pertama di Coil dan di Holding Register ModRSsim2 akan berubah mengikuti Toggle Switch dan Holding Register. Begitu pula ketika nilai di alamat pertama Input Digital dan Input Analog di ModRSsim2 diubah nilainya, maka objek Round LED dan objek Tank juga akan berubah mengikuti nilai di ModRSsim2.
14. Berikutnya, diinginkan agar rangkaian Arduino menggantikan ModRSsim2, yaitu dibuat menjadi Modbus Slave. Untuk itu buat rangkaian Arduino di Proteus seperti gambar berikut (rangkaian ini sama seperti rangkaian Arduino pada Gambar 2.16):



Gambar 3.17 Rangkaian Arduino sebagai Modbus Slave

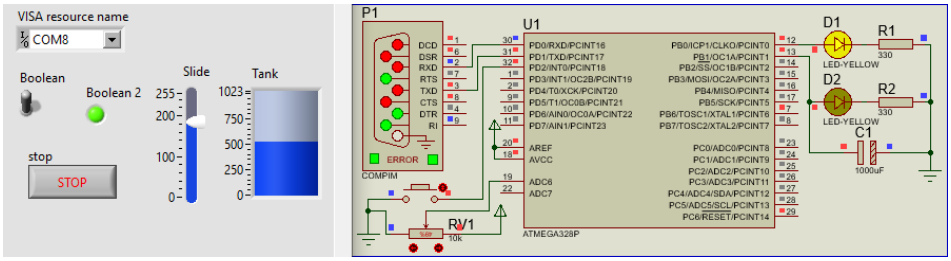
15. Sama seperti LabVIEW yang membutuhkan tambahan library untuk komunikasi Modbus, Arduino juga memerlukan tambahan library. Ada banyak library modbus yang bisa digunakan. Di sini penulis menggunakan library modbus yang dibuat oleh Andre Sarmento Barbosa dkk., yang dapat diunduh di alamat: github.com/andresarmento/modbus-arduino, atau di blog penulis, dengan nama file modbus-arduino-master.zip.
16. Lakukan penambahan library di Arduino dengan cara membuka menu Sketch di software IDE Arduino, pilih Include Library, pilih Add .ZIP library, dan arahkan ke lokasi file modbus-arduino-master.zip. Tunggu beberapa saat hingga di kolom bawah muncul pesan: library telah ditambahkan.
17. Di software IDE Arduino, ketik program berikut ini:

No.	Kode Program 03: ModbusSlave
01.	#include <Modbus.h>
02.	#include <ModbusSerial.h>
03.	const int S1 = 0;
04.	const int tombol = 2;
05.	const int L1 = 0;
06.	const int led = 8;
07.	const int P1 = 0;
08.	const int potensio = A6;
09.	const int H1 = 0;
10.	const int ledpwm = 9;
11.	ModbusSerial mb;
12.	
13.	void setup() {
14.	mb.config(&Serial, 9600, SERIAL_8N1);
15.	mb.setSlaveId(1);
16.	pinMode(tombol, INPUT_PULLUP);
17.	mb.addIsts(S1);
18.	pinMode(led, OUTPUT);
19.	mb.addCoil(L1);
20.	mb.addIreg(P1);
21.	pinMode(ledpwm, OUTPUT);
22.	mb.addHreg(H1);
23.	}
24.	
25.	void loop() {
26.	mb.task();
27.	mb.Ists(S1, digitalRead(tombol));
28.	digitalWrite(led, mb.Coil(L1));
29.	mb.Ireg(P1, analogRead(potensio));
30.	analogWrite(ledpwm, mb.Hreg(H1));
31.	delay(15);
32.	}

Keterangan program:

1. Baris 1-11: memasukkan library Modbus, deklarasi variabel untuk data modbus dan kaki input output Arduino, serta membuat objek modbus (mb).
2. Baris 13-23: blok void setup, berisi program pengaturan komunikasi modbus, mengatur mode input output dan mengatur alamat data modbus (coil, input diskrit, input register dan holding register), semua diisi 0.
3. Baris 25-32: blok void loop, berisi program yang menjalankan komunikasi modbus, yang memberikan respon setiap kali ada request dari Master. Data tombol dan potensio dikirimkan ke Master, sedangkan data coil dan holding register yang diterima, digunakan untuk menyalakan LED di D8 dan D9.

18. Sebelum melakukan kompilasi program di atas, pilih menu Tools, pilih Board = Arduino Nano. Kemudian lakukan kompilasi program dengan menekan tombol Verify (bergambar centang). Tunggu hingga muncul pesan Done Compiling yang menunjukkan kompilasi selesai.
19. Setelah kompilasi selesai, klik 2 kali pada komponen U1 (ATmega328P) di Proteus untuk membuka kotak Edit Properties ATmega328P. Pada kolom Program File, isi dengan lokasi file Hex hasil kompilasi program Arduino. Atur juga parameter yang lain pada kotak Edit Properties ATmega328P ini seperti ditunjukkan pada Gambar 2.17.
20. Berikutnya, klik 2 kali komponen COMPIM, dan atur parameter di dalamnya seperti Gambar 2.19. Pastikan Physical Port berisi pasangan COM dengan COM di VISA LabVIEW, dan isi Baud Rate = 9600.
21. Jalankan program Modbus Master LabVIEW Gambar 3.15 dan rangkaian Arduino Modbus Slave Gambar 3.12 di Proteus. Naik-turunkan tuas Toggle Switch dan Pointer Slide di LabVIEW, dan amati perubahan nyala kedua LED di rangkaian Arduino. Tekan Button dan geser RVI1 di rangkaian Arduino di Proteus, dan amati Round LED dan level Tank di LabVIEW.



Gambar 3.18 Modbus Master LabVIEW dan Modbus Slave Arduino di Proteus

22. Sampai di sini pembuatan Modbus Master dengan LabVIEW, dan Modbus Slave dengan Arduino telah selesai. Sambungan komunikasi serial antara port serial komputer dengan rangkaian Arduino di atas adalah sambungan RS-232. Untuk sambungan RS-485, yang bisa menghubungkan 2 atau lebih rangkaian Arduino pada satu saluran port serial, akan dibahas pada Sub Bab selanjutnya.

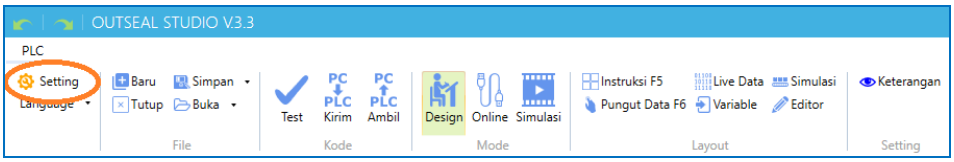
3.4 Protokol Modbus RTU dengan Outseal

Sebelumnya, penulis mengucapkan terimakasih dan apresiasi kepada pencipta Outseal PLC ini, yang termasuk "orang baik" di dunia maya, karena memberikan karyanya secara gratis (open source), dan penulis ikut bangga, karena merupakan karya anak bangsa. Bagi pembaca yang belum mengenal apa itu Outseal, Outseal adalah sebuah software dan juga hardware, yang bisa menjadi alternatif pengganti alat kontrol di industri (PLC). Mengapa Outseal menarik? Karena biayanya yang jauh lebih murah dibandingkan dengan PLC, dengan bahasa pemrograman yang juga berbentuk Ladder Diagram. Lebih jauh mengenai Outseal, pembaca dapat mengunjungi web resminya, di **outseal.com**. Di halaman tersebut, pembaca dapat mendownload software Outseal, dan juga buku panduan penggunaan Outseal, serta tersedia Tutorial yang menarik dan jelas.

Software Outseal yang digunakan dalam buku ini adalah versi 3.3. Versi ini adalah versi terbaru saat buku ini ditulis. Apabila pembaca sudah tidak menemukannya di halaman download, karena sudah muncul versi yang lebih baru, pembaca dapat mengunduhnya di blog penulis.

Di sini Outseal digunakan sebagai Slave. Sekalipun bisa juga diprogram menjadi Master, namun agar ada tampilan komputer yang menarik yang dibuat dengan LabVIEW, maka LabVIEW yang dijadikan Master, sedangkan Outseal dijadikan Slave. Berikut ini langkah-langkah pemrograman Outseal sebagai Slave menggunakan rangkaian Gambar 3.17.

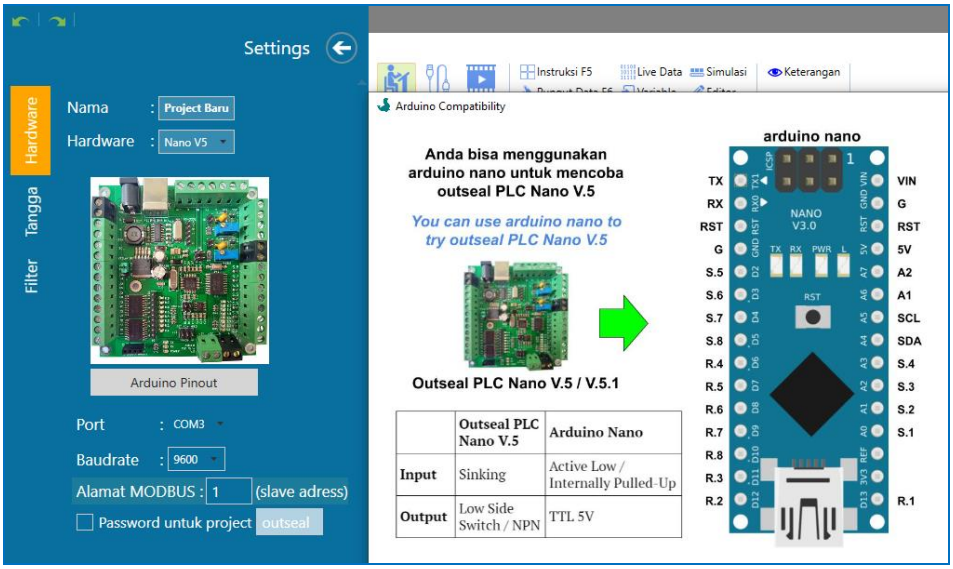
1. Instal software Outseal Studio versi 3.3.
2. Setelah selesai, buka software, tekan tombol Setting di Toolbar.



Gambar 3.19 Tampilan Toolbar Outseal Studio V3.3

3. Pada kotak Setting yang muncul di kiri, pilih Hardware = Nano V5, dengan Baudrate = 9600 dan alamat Modbus = 1 (ini alamat untuk modbus slave).

4. Tekan tombol Arduino Pinout, di bawah gambar hardware Nano V5, maka akan muncul gambar Arduino Nano dan penamaan kakinya di Outseal.



	Outseal PLC Nano V.5	Arduino Nano
Input	Sinking	Active Low / Internally Pulled-Up
Output	Low Side Switch / NPN	TTL 5V

Gambar 3.20 Pinout atau kaki input output di Outseal Nano V5

Lebih jauh, berikut ini Tabel hubungan antara kaki komponen ATmega328, dengan Arduino Nano, dan dengan Outseal Nano V5.

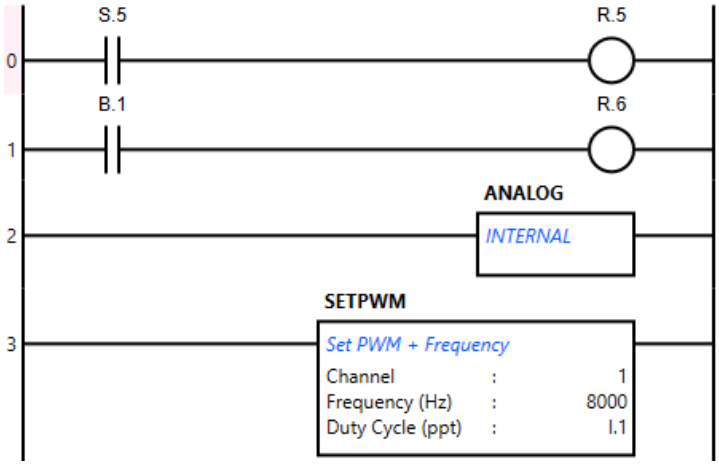
Tabel 3.3 Daftar Kaki Input Output Outseal Nano V5

No.	ATmega328	Arduino Nano	Nano V5	Fungsi di Nano V5
1.	PD0	D0	RX	Komunikasi serial
2.	PD1	D1	TX	Komunikasi serial
3.	PD2	D2	S5	Input Digital 5
4.	PD3	D3	S6	Input Digital 6
5.	PD4	D4	S7	Input Digital 7
6.	PD5	D5	S8	Input Digital 8
7.	PD6	D6	R4	Output Digital 4
8.	PD7	D7	R5	Output Digital 5

No.	ATmega328	Arduino Nano	Nano V5	Fungsi di Nano V5
9.	PB0	D8	R6	Output Digital 6
10.	PB1	D9	R7	Output Digital 7/PWM
11.	PB2	D10	R8	Output Digital 8
12.	PB3	D11	R3	Output Digital 3
13.	PB4	D12	R2	Output Digital 2
14.	PB5	D13	R1	Output Digital 1
15.	PC0	A0	S1	Input Digital 1
16.	PC1	A1	S2	Input Digital 2
17.	PC2	A2	S3	Input Digital 3
18.	PC3	A3	S4	Input Digital 4
19.	PC4	A4	SDA	Komunikasi i2C
20.	PC5	A5	SCL	Komunikasi I2C
21.	PC6	A6	A1	Input Analog 1
22.	PC7	A7	A2	Input Analog 2

5. Dari Tabel di atas, diketahui Outseal Nano V5 memiliki 8 input digital, 8 output digital, 2 input analog, 1 output analog (PWM), TX-RX untuk komunikasi serial (RS-232/RS-485), dan SDA-SCL untuk komunikasi I2C.
6. Setelah mengetahui kaki I/O Outseal, berikutnya membuat program yang dapat menjalankan rangkaian pada Gambar 3.17, menggantikan program Arduino sebelumnya. Program untuk bisa membaca Tombol, membaca Potensio, mengatur nyala/padam LED dan intensitas cahaya LED, dapat ditunjukkan seperti Gambar 3.21 di halaman berikut.

Catatan: Apabila pembaca belum mengetahui bagaimana membuat program di Outseal, silahkan pembaca bisa mengikuti tutorial pengoperasian Outseal, yang dapat dilihat di video youtube. Buka youtube, ketik kata kunci: “outseal”, buka salah satu video yang berjudul “Tutorial singkat dasar pengoperasian outseal”.



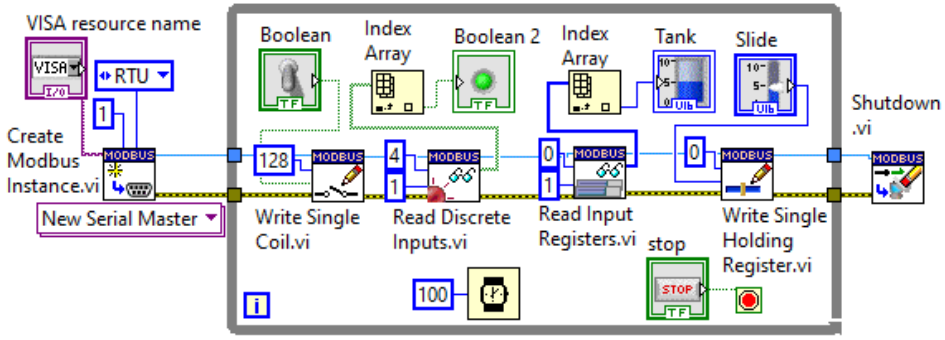
Gambar 3.21 Program Outseal untuk pengganti program Arduino di Rangkaian Gambar 3.17

Keterangan program:

1. Anak tangga 0: setiap kali kontak S5 close, koil R5 akan bernilai 1, sebaliknya ketika kontak S5 open, koil R5 akan bernilai 0. Kontak dan koil ini bisa diambil dengan menekan tombol Instruksi, kemudian pilih kategori Bit. Kontak S5 ini sama dengan kaki D2 Arduino, yang terhubung dengan tombol. Kontak close berarti tombol ditekan, kontak open berarti tombol tidak ditekan. Kontak S5 ini juga dapat dibaca dengan protokol Modbus, yaitu di alamat 5 Input Diskrit.
2. Anak tangga 1: setiap kali kontak B1 close, koil R6 bernilai 1, sebaliknya ketika B1 open, R6 akan bernilai 0. Berbeda dengan kontak S5 yang memiliki kaki I/O fisik, kontak B1 ini tidak memiliki kaki I/O fisik, namun bisa dibaca dan bisa ditulis. Kontak B1 ini dapat dibaca dan ditulis dengan protokol Modbus, yaitu di alamat 129 Modbus Coil. Koil R6 memiliki kaki I/O fisik. Koil R6 ini sama dengan kaki D8 Arduino, yang terhubung dengan LED. Jadi ketika R6 bernilai 1, LED di D8 menyala, sebaliknya ketika R6 bernilai 0, LED di D8 padam.
3. Anak tangga 2: mengaktifkan input analog internal, yaitu A1 dan A2. Instruksi Analog Internal ini dapat diambil di tombol Instruksi, di kategori Data. Input Analog A1 dan A2 ini memiliki kaki I/O secara fisik, yaitu di A6 dan A7 Arduino Nano. Di samping memiliki kaki I/O fisik, Input Analog A1 dan A2 ini juga dapat dibaca dengan protokol Modbus, di alamat memori 1 dan 2 Input Register.

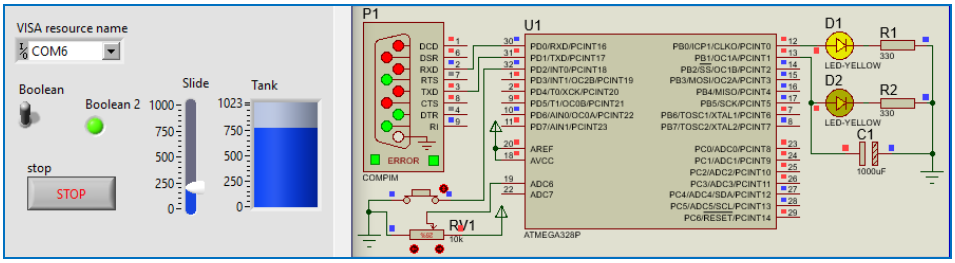
4. Anak tangga 3: mengaktifkan output analog (PWM). Untuk mengaktifkan output analog di R7, isi Channel = 1. Untuk nilai Frequency (Hz), bisa diisi dari 0 – 8000 Hz. Semakin besar nilai frequency, semakin bagus output PWM yang dihasilkan (semakin mendekati nilai analog). Untuk nilai Duty Cycle, bisa diisi dengan angka konstan, atau variabel, dengan nilai 0 - 1000. Kaki R7 Outseal ini sama dengan kaki D9 Arduino, yang terhubung dengan LED. Semakin besar nilai Duty Cycle, semakin besar intensitas cahaya LED. Agar nilai Duty Cycle ini bisa diubah-ubah, di program Gambar 3.21 di atas, nilai Duty Cycle diisi dengan variabel Integer I.1. Nilai variabel Integer I.1 ini menempati alamat 1 di Holding Register data Modbus, yang artinya isinya bisa dibaca dan ditulis.

7. Setelah selesai membuat program di atas, lakukan kompilasi program dengan menekan tombol Test (bergambar tanda centang) pada Toolbar Outseal. Apabila kompilasi berhasil, akan muncul tulisan “Tidak terdapat kesalahan”. Kompilasi ini akan menghasilkan file Hex (Hasil.hex), yang nantinya diperlukan untuk menjalankan simulasi di Proteus.
8. Setelah kompilasi berhasil, buka rangkaian Arduino di Proteus, yang berisi rangkaian Arduino seperti Gambar 3.17. Klik 2 kali pada komponen ATmega328P, untuk membuka jendela Edit Component. Pada jendela Edit Component yang terbuka, isi kotak Program File dengan alamat lokasi file Hex, hasil kompilasi program Outseal, yang selalu berada di lokasi: C:\Users\.....\AppData\Local\Temp\Outseal\Hasil.hex. Ganti titik-titik dengan nama user pada komputer Anda.
9. Sebelum program LabVIEW dan rangkaian Arduino di Proteus dijalankan, program LabVIEW perlu diperbaiki dulu karena lokasi Coil dan Input Diskrit pada program Outseal berbeda dengan program Arduino sebelumnya. Pada kaki address icon Write Single Coil, ganti angka 0 dengan 128. Mengapa 128? Karena alamat Modbus yang digunakan di LabVIEW dimulai dari angka 0, sedangkan alamat Modbus yang digunakan di Outseal dimulai dari angka 1. Maka untuk alamat kontak B1 yang seharusnya 129, di LabVIEW menjadi 128. Begitu pula untuk kaki address di Read Discrete Input, ganti angka 0 dengan angka 4, bukan angka 5.



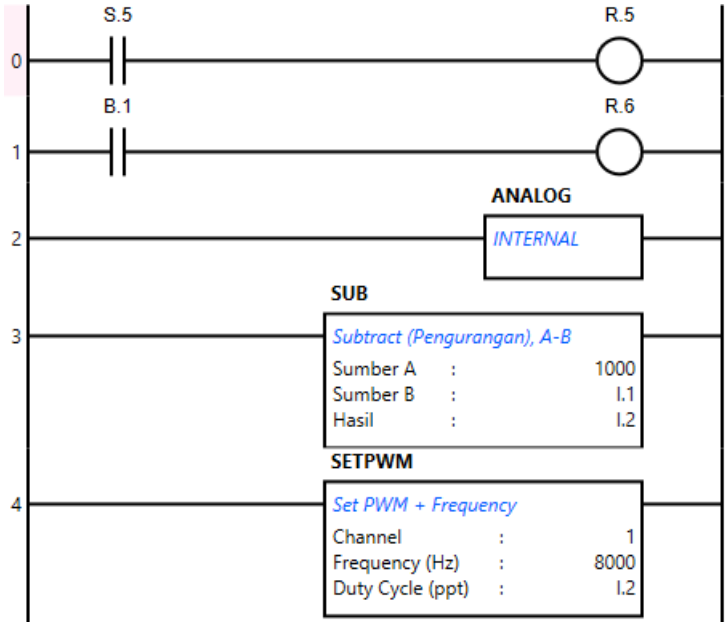
Gambar 3.22 Perbaikan program LabVIEW di alamat Write single Coil dari 0 menjadi 128 (kontak B1), dan di alamat Read Discrete Inputs, dari 0 menjadi 4 (kontak S5)

10. Setelah mengganti nilai kaki input address di Write Single Coil dan Read Discrete Inputs, berikutnya jalankan program LabVIEW dan rangkaian Arduino di Proteus. Naik Turunkan Toggle Switch dan Pointer Slide di LabVIEW (ubah nilai maksimum Pointer Slider dari 1023 menjadi 1000), dan amati perubahan nyala LED di D8 dan di D9 Arduino. Begitu pula tekan tombol dan geser Potensio di rangkaian Arduino, dan amati perubahan nyala Round LED dan Tank di LabVIEW.



Gambar 3.23 Program Modbus Master di LabVIEW dan Modbus Slave Outseal di Proteus

11. Perhatikan, nilai Pointer Slide di LabVIEW ternyata terbalik dengan nilai intensitas cahaya LED di D9 Arduino. Agar nilainya tidak terbalik, tambahkan instruksi SUB di program Outseal (ambil instruksi SUB di kategori instruksi Hitung), yang mengubah nilai output Duty Cycle dengan persamaan Duty Cycle = I.2 = 1000 – I.1, di mana I.1 adalah nilai yang diterima dari protokol Modbus, dan I.2 adalah nilai untuk diberikan ke Duty Cycle, sehingga programnya menjadi seperti Gambar 3.24 berikut.

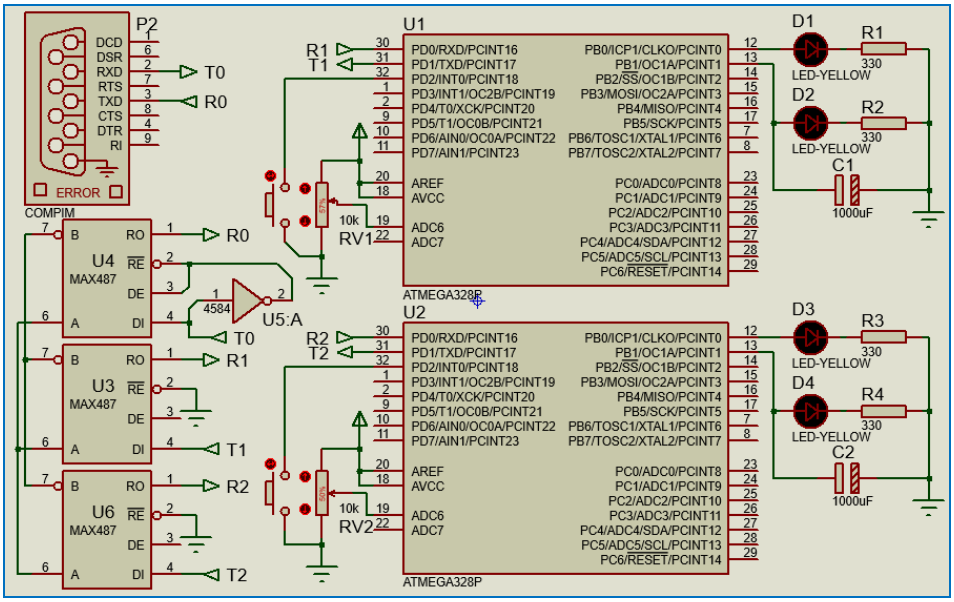


Gambar 3.24 Penambahan SUB untuk membuat nilai PWM sesuai dengan input data Modbus

12. Kompilasi program di atas, dan jalankan kembali program LabVIEW dan rangkaian Arduino di Proteus, maka seharusnya nilai Pointer Slide yang semakin besar akan membuat intensitas cahaya LED di D9 Arduino semakin menyala terang.
13. Berikutnya, diinginkan ada 2 buah rangkaian Arduino yang dihubungkan dengan sambungan RS-485 menggunakan protokol Modbus dengan LabVIEW sebagai Master, dan kedua Arduino sebagai Slave1 dan Slave2.

Catatan: Untuk membuat sambungan RS-485, dibutuhkan IC MAX 485, yang mengubah jalur komunikasi serial RX dan TX di setiap Arduino menjadi jalur A dan B, di mana data di jalur A selalu berkebalikan dengan data di jalur B. Data di jalur A dan B ini dibuat selalu berkebalikan, selain untuk pengecekan bahwa data yang diterima benar, juga untuk membuat jarak komunikasi menjadi jauh, hingga 1,2 km, karena yang dideteksi bukan lagi level tegangan terhadap 0, tetapi nilai selisih antara A dan B. Namun demikian, dengan jalur A dan B ini membuat komunikasi pada sambungan RS-485 ini hanya bisa Half Duplex, yaitu dalam satu waktu hanya bisa terima saja (RX) atau kirim saja (TX).

14. Buat rangkaian 2 Arduino dengan sambungan RS485 seperti berikut ini:



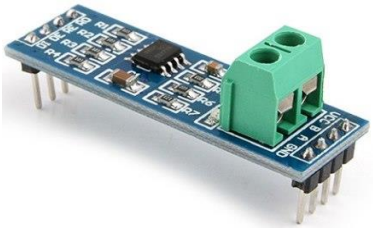
Gambar 3.25 Rangkaian 2 buah Arduino dengan sambungan RS-485

15. Berikut ini daftar komponen yang digunakan dalam rangkaian di atas:

Tabel 3.4 Daftar Komponen Rangkaian Gambar 3.25

No.	Komponen	Jumlah	Fungsi
1.	ATmega328	2	Representasi Arduino Nano
2.	Button	2	Alat input Digital
3.	COMPIM	1	Representasi port serial
4.	Cap-Elec 1000uF	2	Penghalus tegangan PWM
5.	IC 4584	1	Kontrol RE DE MAX-487
6.	LED-Yellow	4	Output Digital 1
7.	IC MAX-487	3	Input Digital 1
8.	Pot-HG 10K	2	Input Digital 2
9.	Res 330 Ohm	4	Input Digital 3

Catatan: IC MAX-487 adalah keluarga dari IC MAX-485 yang biasa digunakan untuk sambungan RS-485. Di pasaran, komponen ini dijual dalam bentuk break-out seperti Gambar 3.26. Dalam rangkaian Gambar 3.25 dibutuhkan 3 buah IC MAX-487, karena ada 3 unit yang dihubungkan, yaitu 1 Master dan 2 Slave. Jadi setiap unit, termasuk Master, harus dihubungkan dengan MAX-485 atau MAX-487, agar bisa tersambung secara RS-485. Kaki RX dan TX setiap unit harus dihubungkan dengan kaki RO dan DI, kemudian output A dan B setiap MAX-485 disatukan, A dengan A, dan B dengan B. Kaki RE dan DE digunakan untuk kontrol, apabila RE dan DE diberi LOW, berarti MAX-485 difungsikan untuk menerima data, sebaliknya bila RE dan DE diberi HIGH, berarti MAX-485 difungsikan untuk mengirim data (seperti telah disebutkan di atas, RS-485 bersifat Half-Duplex, yang hanya bisa mengirim saja atau menerima saja dalam satu waktu). Dalam rangkaian di atas, untuk membuat Slave otomatis menerima data dan merespon balik, kaki RE diberi LOW, dan kaki DE tidak diberi tegangan. Kemudian agar Master bisa menginisiasi komunikasi (Master dihadirkan dengan COMPIM), kaki RE dan DE MAX-487 disatukan dan diberi output NOT, dengan input NOT diambil dari kaki output Master (TO). Di COMPIM, kaki RX diberi TO dan kaki TX diberi DI, karena COMPIM terhubung silang dengan port Master (komputer dengan LabVIEW).



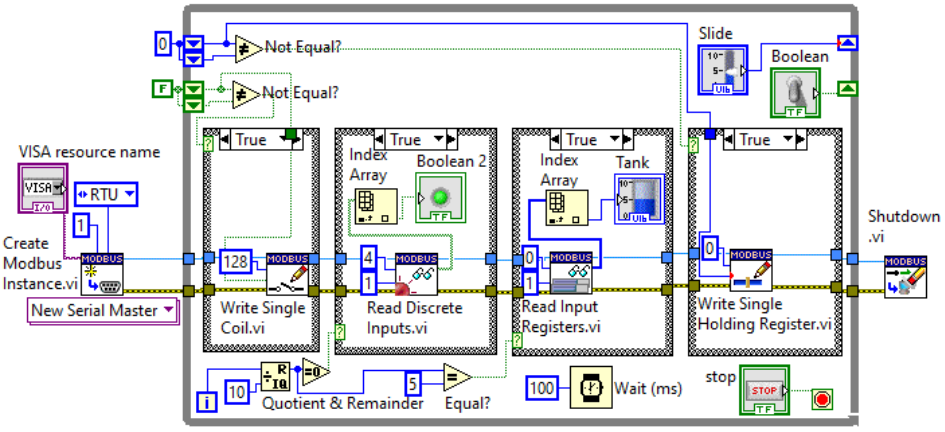
Gambar 3.26 Break-out MAX-485, terminal kiri: DI, DE, RE, RO, terminal kanan: Vcc, A, B, Gnd

14. Setelah rangkaian Gambar 3.25 di atas selesai dibuat, karena rangkaian setiap unit Slavenya sama dengan rangkaian Gambar 3.17, maka gunakan program yang sama untuk kedua Slave, hanya berbeda pada alamat Modbusnya saja. Jadi isi Program File komponen U1, atau ATmega328P yang atas dengan file Hasil.hex yang dibuat dengan alamat Modbus = 1.

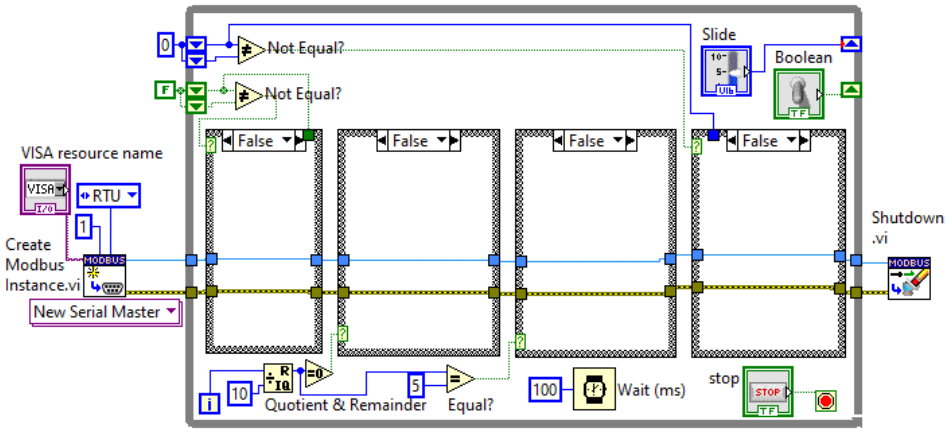
15. Sedangkan komponen U2, atau ATmega328P yang bawah, isi dengan file Hasil.hex yang dibuat dengan alamat Modbus = 2. Penggantian alamat Modbus ini dapat dilakukan di kotak Settings Outseal.

Catatan: setiap kali dilakukan kompilasi di Outseal, file hasil kompilasi sebelumnya ditimpa dengan yang baru. Untuk itu, setelah melakukan kompilasi, copy atau pindahkan file Hasil.hex tersebut ke lokasi yang lain. Akan lebih baik bila file Hasil.hex tersebut dipindahkan ke lokasi yang sama dengan file Proteus yang berisi rangkaian yang sedang dibuat. Jangan lupa mengganti nama file tersebut dengan nama yang berbeda. Sebagai contoh, untuk kompilasi program Modbus 1, beri nama Hasil1.hex, dan kompilasi program Modbus 2, beri nama Hasil2.hex.

16. Setelah komponen U1 dan U2 ATmega328P pada rangkaian di Proteus secara berturut-turut diisi dengan file Hasil1.hex dan Hasil2.hex, berikutnya modifikasi program Modbus Master di LabVIEW agar bisa mengatur kedua Slave di Proteus. Karena ada 2 Slave, maka program Master di LabVIEW harus diperbaiki agar data yang dikirimkan ke Slave menjadi lebih efisien, yaitu hanya mengirimkan data apabila ada perubahan nilai, dan menerima data secara periodis, seperti ditunjukkan pada gambar berikut ini, yang merupakan perbaikan dari Gambar 3.22.

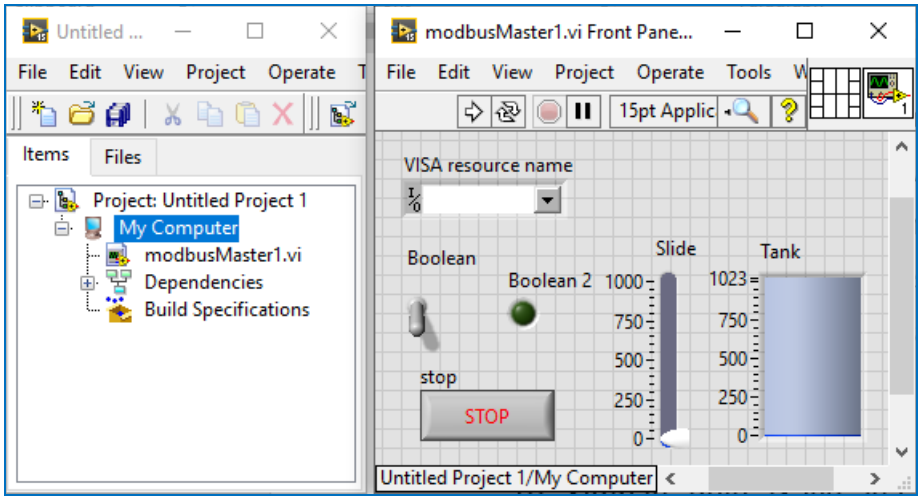


Gambar 3.27 Perbaikan program Master Gambar 3.22 setiap fungsi diberi Struktur Case



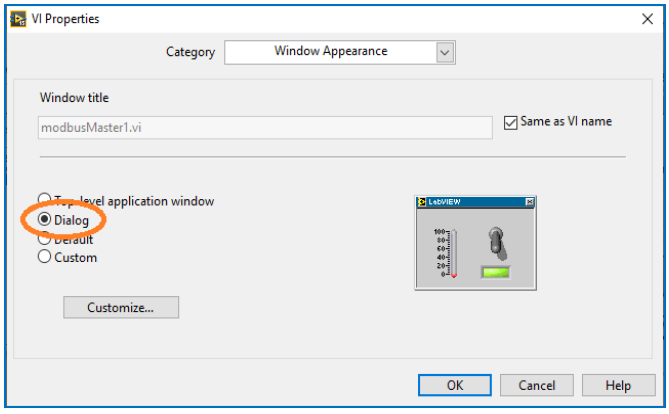
Gambar 3.28 Ketika nilai tidak berubah, tidak ada data yang dikirimkan, ketika waktu perulangan belum di milidetik ke 500 atau ke 1000, maka tidak ada data yang diterima

17. Setelah program Master LabVIEW di atas selesai dibuat, jalankan program LabVIEW tersebut, dan jalankan juga Proteus.
18. Perhatikan pengubahan Toggle Switch dan Pointer Slide di LabVIEW akan dapat mengatur nyala LED di D8 dan di D9 Slave1 (U1). Begitu pula sebaliknya, pengubahan kondisi tombol dan potensio Slave1, membuat kondisi Round LED dan Tank di LabVIEW berubah.
19. Untuk pengaturan di Slave2 (U2), pembaca perlu membuat lagi program Master LabVIEW yang sama dengan Gambar 3.27 dan 3.28 di atas, hanya unit ID di Create Modbus, harus diganti dari angka 1 menjadi 2. Namun sebelum membuat program ini, agar nantinya programnya tidak menjadi ruwer, silahkan menggunakan fasilitas Project di LabVIEW. Berikut ini langkah-langkahnya:
20. Simpan dulu (Save As) program Gambar 3.27 di atas dengan nama modbusMaster1.vi. Agar tampilan program Master ini tidak memenuhi layar, perkecil ukuran tampilan Front Panel.
21. Buka menu Project, pilih Create Project. Pada jendela Create Project yang muncul, pilih Blank Project, dan tekan tombol Finish. Akan muncul pesan “You have one or more …”, tekan pilihan Add. Maka akan muncul kotak Untitled Project 1 yang berisi daftar isi Project, dengan nama modbusMaster1.vi berada di daftarnya.



Gambar 3.29 Untitled Project 1 dengan program modbusMaster1.vi berada di dalamnya

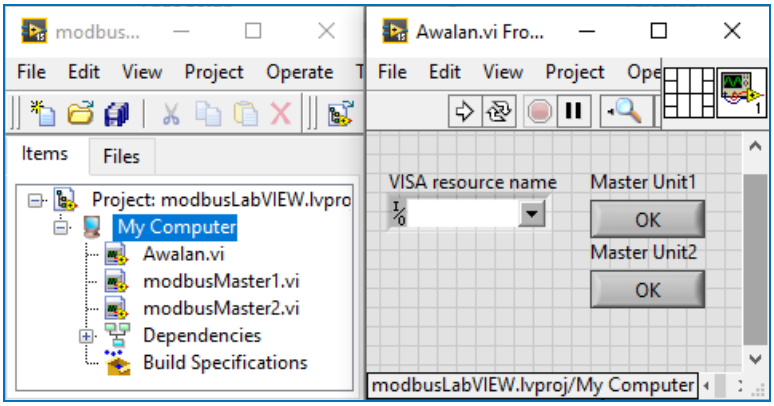
22. Agar tampilan Toolbar Front Panel modbusMaster1.vi tidak menampilkan tombol Run Continuously, yang bila ditekan bisa mengakibatkan program tidak bisa dihentikan (karena ada perulangan di dalam perulangan), maka di Front Panel, pilih menu File, pilih VI Properties. Pada kotak VI Properties yang muncul, pilih Category: Window Appearance, pilih Dialog, klik OK.



Gambar 3.30 Pilih menu File, pilih VI Properties, pilih Window Appearance, pilih Dialog

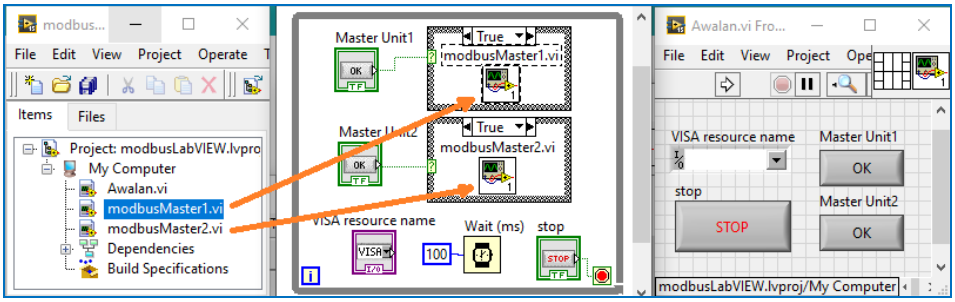
23. Berikutnya Save As program modbusMaster1.vi ini, dan beri nama modbusMaster2.vi. Maka di kotak Untitled Project 1, akan muncul nama modbusMaster2.vi menggantikan nama modbusMaster1.vi.

24. Agar modbusMaster1.vi kembali muncul di daftar Project, klik kanan pada nama My Computer, pilih Add, cari nama modbusMaster1.vi, klik OK, maka modbusMaster1.vi akan muncul kembali di daftar nama Project.
25. Berikutnya, pada Toolbar kotak Untitled Project 1, tekan tombol Save All (this Project), beri nama Project dengan nama modbusLabVIEW.
26. Berikutnya, buka modbusMaster2.vi dengan meng-klik 2 kali pada nama modbusMaster2.vi di daftar Project. Setelah muncul Front Panel modbusMaster2.vi, tekan tombol Control dan E di keyboard, maka akan muncul Block Diagram.
27. Di Block Diagram modbusMaster2.vi, ubah angka 1 di Unit ID Create Modbus menjadi angka 2. Tekan Control dan S di keyboard untuk menyimpan program, dan tekan Control dan W di keyboard untuk menutup Block Diagram. Kemudian tekan Control dan W sekali lagi untuk menutup Front Panel modbusMaster2.vi.
28. Berikutnya, diinginkan agar ada sebuah program LabVIEW, yang menjadi sebuah tampilan pembuka, dengan 2 buah tombol. Bila tombol pertama ditekan, maka akan memunculkan tampilan modbusMaster1, apabila tombol kedua ditekan, akan memunculkan tampilan modbusMaster2. Di samping 2 buah tombol, diinginkan juga sebuah kotak VISA Resource Name, yang digunakan untuk memilih saluran Port COM. Mengapa kotak pilihan Port COM ini diletakkan di program pembuka? Karena sebelum menjalankan program modbusMaster1 dan modbusMaster2, port COM harus sudah ditentukan dulu. Jadi pengguna akan membuka program pembuka, mengatur port COM (yang akan digunakan oleh program modbusMaster1 dan program modbusMaster2), baru kemudian menekan salah satu tombol untuk memilih tampilan modbusMaster1 atau modbusMaster2. Untuk membuat hal ini, klik kanan pada nama My Computer di kotak Project, pilih New, pilih VI.
29. Pada Front Panel VI yang baru, tempatkan 2 buah OK Button, yang diambil dari kategori Modern, Boolean, dan sebuah kotak VISA Resource Name, yang diambil dari kategori Modern, kategori I/O.
30. Save atau simpan program VI yang baru ini dengan nama Awalan.vi.



Gambar 3.31 Program Awalan.vi berisi Kotak VISA Resource Name dan 2 buah OK Button

31. Sama seperti pada program modbusMaster1 dan modbusMaster2, hilangkan tombol Run Continuouly pada program Awalan.vi dengan memilih File → VI Properties → Window Appearance → Dialog, klik OK.
32. Berikutnya, buka Block Diagram dengan menekan tombol Control dan E.
33. Di Block Diagram, ambil Struktur While Loop, dari palet Functions, dari kategori Programming, dari kategori Structures, dan tempatkan ketiga icon yang ada (2 OK Button dan 1 Visa Resource) ke dalam While Loop.
34. Tambahkan sebuah tombol Stop di Terminal Loop Condition dengan meng-klik kanan Terminal Loop Condition, pilih Create, pilih Control.
35. Tambahkan icon Wait(ms), beri inputnya nilai Constant 100, dengan cara klik kanan icon Wait(ms), pilih Create, pilih Constant, ganti 0 dengan 100.
36. Ambil dan tempatkan 2 buah Struktur Case (Case Structure), yang diambil dari palet Functions, dari kategori Programming, dari kategori Structure, dan tempatkan keduanya di dalam While Loop.
37. Hubungkan kaki Case Selector (bergambar ?) di Case Structure pertama, dengan icon OK Button pertama (Master Unit1), dan Case Selector di Case Structure kedua, dengan icon OK Button kedua (Master Unit2).
38. Kemudian tarik atau “drag” nama modbusMaster1.vi dari kotak Project ke Block Diagram, hingga masuk ke dalam Case Structure pertama.
39. Ulangi untuk nama modbusMaster2.vi, tarik atau “drag” hingga masuk ke dalam Case Structure kedua di Block Diagram.



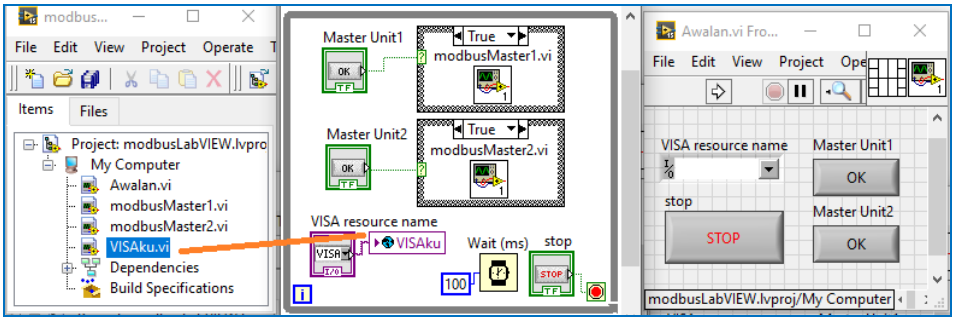
Gambar 3.32 Tarik modbusMaster1 dari kotak Project ke Struktur Case Unit1 di Block Diagram

40. Berikutnya, agar pilihan di kotak VISA Resource Name yang ada di program Awalan.vi dapat diteruskan ke program modbusMaster1.vi dan program modbusMaster2.vi, dibutuhkan sebuah Global Variable.

> **Catatan**: apabila Local Variable bisa digunakan untuk menghadirkan nilai variable di beberapa tempat dalam satu Block Diagram, bahkan untuk While Loop yang berbeda sekalipun, maka Global Variable lebih besar lagi ruang lingkupnya, yaitu bisa menghadirkan nilai variable dari satu program vi ke program vi yang lain. Seperti halnya variable VISA Resource Name yang ada di program Awalan.vi, agar bisa dibaca di program modbusMaster1.vi dan program modbusMaster2.vi, maka variable VISA Resource Name harus dijadikan Global Variable.

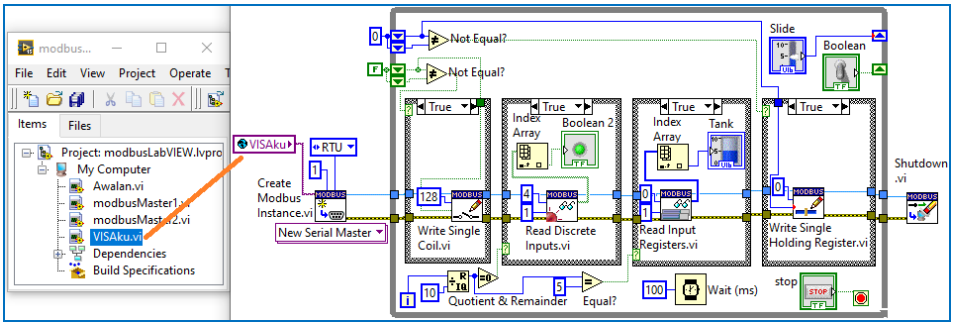
41. Berikut ini cara pembuatan Global Variable VISA Resource Name:
42. Pada kotak Project, klik kanan pada nama My Computer, kemudian pilih New, kemudian pilih New lagi, maka akan muncul kotak Create New. Pada daftar di kotak Create New, pilih Global Variable, klik OK, maka pada kotak Project, akan muncul nama Global 1.
43. Global 1 ini sebenarnya sebuah program vi, tetapi tidak memiliki Block Diagram, hanya Front Panel saja. Untuk membuat VISA Resource Name menjadi Global Variable, di Front Panel Global 1, ambil dan tempatkan kotak VISA Resource Name, dan ubah nama labelnya, misal VISAku.
44. Simpan Global 1 (tekan Control dan S) dan beri nama misalnya VISAku.
45. Tekan tombol Control dan W di Keyboard untuk menutup VISAku.

46. Di kotak Project, tarik nama VISAku dan masukkan ke While Loop di Block Diagram program Awalan.vi, hinga muncul icon VISAku. Kemudian hubungkan icon VISA Resource Name dengan icon VISAku.



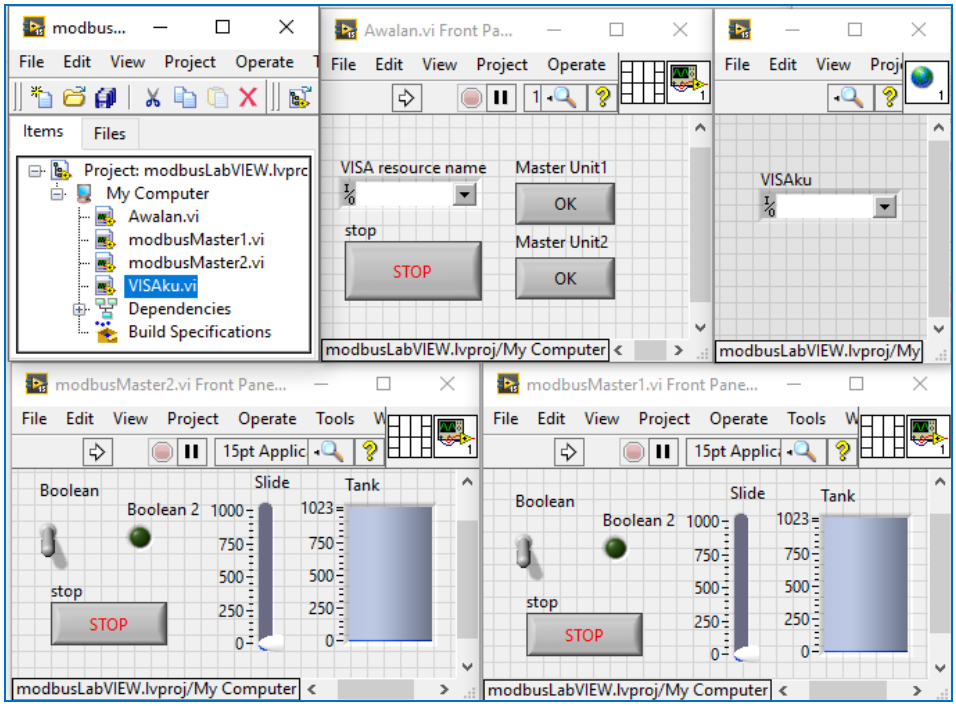
Gambar 3.33 Tarik Global Variable VISAku dari kotak Project ke Block Diagram Awalan.vi

47. Hilangkan kotak VISA Resource Name di program modbusMaster1.vi dan modbusMaster2.vi, dan ganti dengan icon VISAku. Icon VISAku diperoleh dengan cara menarik (men-drag) nama VISAku dari kotak Project, hingga muncul di Block Diagram.
48. Secara default icon VISAku memiliki kaki input. Agar bisa memberikan output ke icon Create Modbus, perlu mengubah kaki input menjadi kaki output. Untuk itu klik kanan icon VISAku, pilih Change to Read, maka kaki input yang ada di kiri, sekarang berubah menjadi kaki output di kanan.
49. Setelah itu hubungkan kaki output VISAku dengan kaki VISA Create Modbus, seperti ditunjukkan gambar berikut:



Gambar 3.34 Men-"drag" VISAku, dan mengganti icon VISA Resource Name dengan VISAku

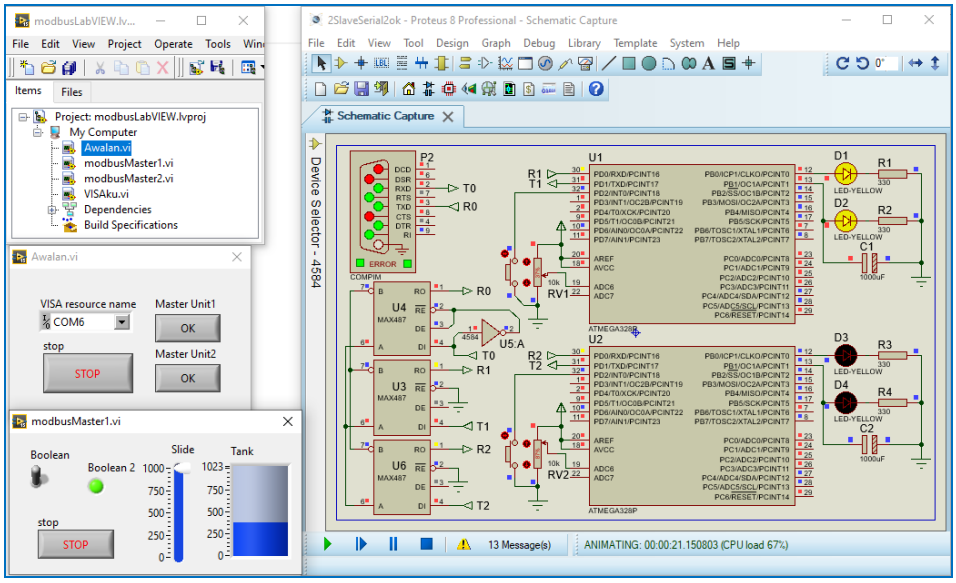
50. Gambar berikut ini menunjukkan tampilan Front Panel keempat program yang ada di kotak Project berturut-turut dari kiri ke kanan, atas ke bawah: program Awalan.vi, Global variable VISAku, modbusMaster2.vi dan modbusMaster1.vi. Perhatikan bahwa tidak ada tombol Run Continuously di semua Front Panel, dan juga di program modbusMaster1.vi dan modbusMaster2.vi, sudah tidak ada lagi kotak VISA Resource Name.



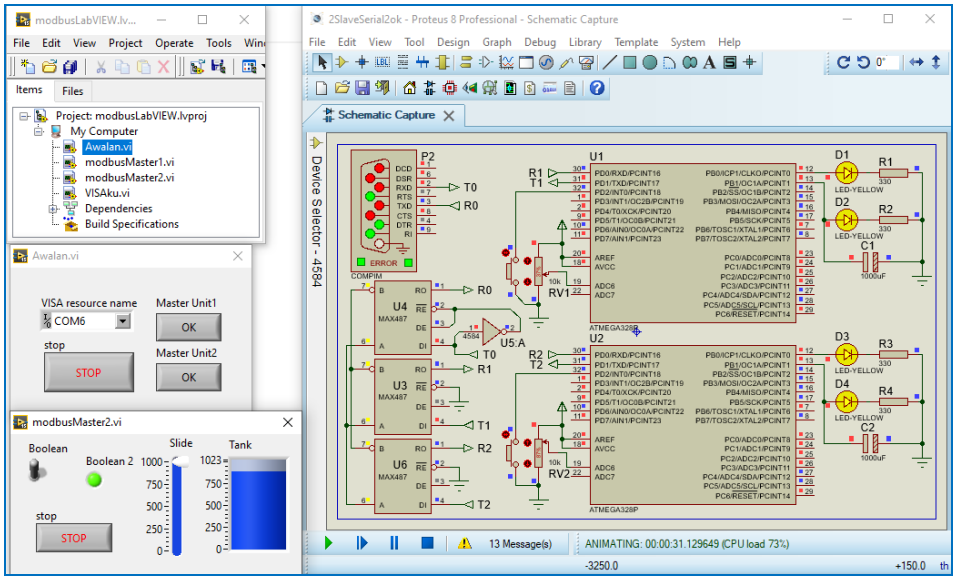
Gambar 3.35 Kotak VISA Resource Name yang sebelumnya ada di program modbusMaster1.vi dan modbusMaster2.vi, sudah dipindahkan di program Awalan.vi, dengan bantuan Global Variable VISAku, yang bisa meneruskan isi variable VISA Resource dari program ke program

51. Berikutnya, tutup semua program LabVIEW di atas, kecuali program Awalan.vi. Kemudian tekan tombol Run pada program Awalan.vi.
52. Pada tampilan program Awalan.vi, pilih salah satu COM yang berpasangan pada kotak VISA Resource Name. Di sini, Port COM yang dipilih akan menjadi port COM program modbusMaster1.vi dan program modbusMaster2.vi. Berikutnya, tekan tombol Master Unit 1, maka akan muncul tampilan modbusMaster1.vi.

53. Sementara program modbusMaster1.vi masih berjalan, jalankan rangkaian 2 buah Arduino di Proteus (Gambar 3.25). Namun sebelum menjalankan rangkaian tersebut, pastikan port COM pada COMPIM merupakan pasangan COM dengan COM di program Awalan.vi. Setelah pasangan COM tersebut benar, jalankan Proteus.
54. Lakukan pengubahan pada Toggle Switch dan Pointer Slide, dan amati kondisi kedua LED di D8 dan D9 yang terhubung dengan U1 (rangkaian ATmega328 yang atas). Seharusnya setiap kali Toggle Switch dinaikkan ke atas, LED di D8 menyala, dan sebaliknya padam bila Toggle Switch diturunkan. Begitu pula untuk Pointer Slide, apabila dinaikkan ke atas, LED di D9 akan menyala semakin terang, dan sebaliknya akan meredup apabila Pointer Slide diturunkan (lihat Gambar 3.36).
55. Di Proteus, lakukan penekanan Tombol dan penggeseran Potensio yang terhubung dengan U1 (ATmega328 yang atas). Maka harusnya objek Round LED dan Tank akan berubah mengikuti nilai Tombol dan Potensio.
56. Setelah objek-objek di modbusMaster1 dan di Proteus berjalan dengan benar, tutup program modbusMaster1.vi dengan menekan tombol STOP.
57. Di program Awalan.vi, tekan tombol Master Unit 2, untuk menjalankan program modbusMaster2.vi. Ulangi hal yang sama seperti langkah di atas, yaitu lakukan pengubahan pada Toggle Switch dan Pointer Slide, dan amati LED di D8 dan D9 yang terhubung dengan U2 (rangkaian ATmega328 yang bawah). Seharusnya setiap kali Toggle Switch dinaikkan ke atas, LED di D8 menyala, dan sebaliknya padam bila Toggle Switch diturunkan. Begitu pula untuk Pointer Slide, apabila dinaikkan ke atas, LED di D9 akan menyala semakin terang, dan sebaliknya akan meredup apabila Pointer Slide diturunkan (lihat Gambar 3.37).
58. Di Proteus, lakukan penekanan Tombol dan penggeseran Potensio yang terhubung dengan U2 (ATmega328 yang bawah). Maka harusnya objek Round LED dan Tank akan berubah mengikuti nilai Tombol dan Potensio.
59. Setelah objek-objek di modbusMaster2.vi dan di Proteus berjalan dengan benar, tutup program modbusMaster2.vi dengan menekan tombol STOP.
60. Sampai di sini protokol Modbus RTU dengan Outseal telah selesai.



Gambar 3.36 Program modbusMaster1 memantau dan mengontrol rangkaian Arduino U1, pengubahan Toggle Switch dan Pointer Slide di LabVIEW, mengubah nyala LED D1 dan D2, pengubahan Tombol dan Potensio RV1, mengubah nyala Round LED dan Tank di LabVIEW.

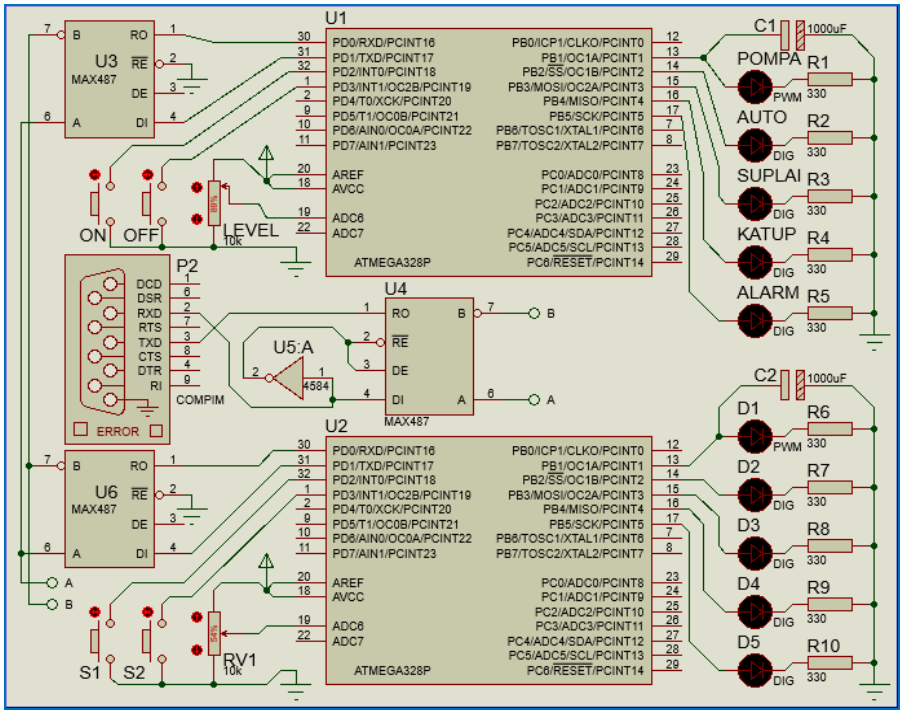


Gambar 3.37 Program modbusMaster2 memantau dan mengontrol rangkaian Arduino U2, pengubahan Toggle Switch dan Pointer Slide di LabVIEW, mengubah nyala LED D3 dan D4, pengubahan Tombol dan Potensio RV2, mengubah nyala Round LED dan Tank di LabVIEW.

3.5 Tampilan SCADA 2 Slave dengan Modbus RTU

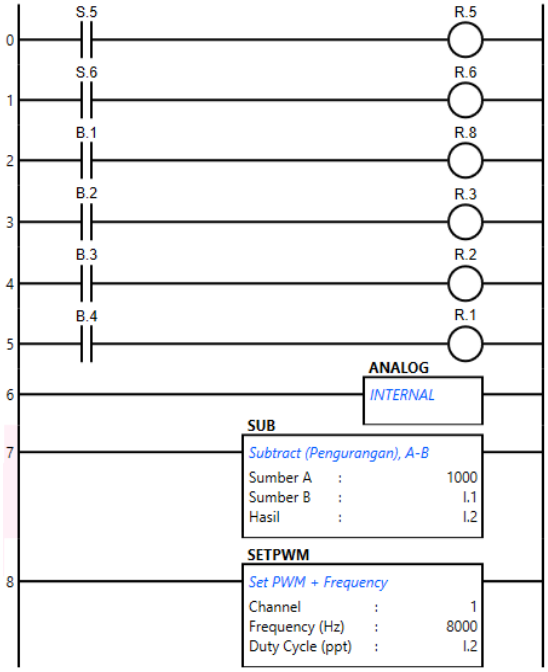
Dari Sub Bab 3.4, telah dicontohkan bagaimana penerapan protokol Modbus RTU pada LabVIEW dan Arduino. Selain itu juga telah diberikan contoh penggunaan Outseal dan pemrogramannya menggunakan Ladder Diagram, yang dapat menjadi alternatif pengganti program Arduino. Di samping itu, dari pembuatan program di LabVIEW, telah diuraikan juga bagaimana membuat program LabVIEW menjadi lebih mudah diorganisir, yaitu dengan menggunakan Project. Juga telah diuraikan bagaimana menggunakan Global Variable untuk meneruskan data dari satu program ke program lain. Program di Sub Bab 3.5 ini pada dasarnya sama dengan program yang dibuat di Sub Bab 3.4, hanya tampilan Front Panel di program Master Unit 1 dan program Master Unit 2 menggunakan Tampilan Pengisian Tangki. Berikut ini langkah-langkah pembuatan Tampilan SCADA Pengisian Tangki LabVIEW dengan 2 buah Slave Arduino (Outseal) dengan protokol Modbus RTU:

1. Buat rangkaian 2 buah Arduino di Proteus seperti berikut:



Gambar 3.38 Rangkaian 2 buah Arduino (Outseal) untuk Modbus Tampilan Pengisian Tangki

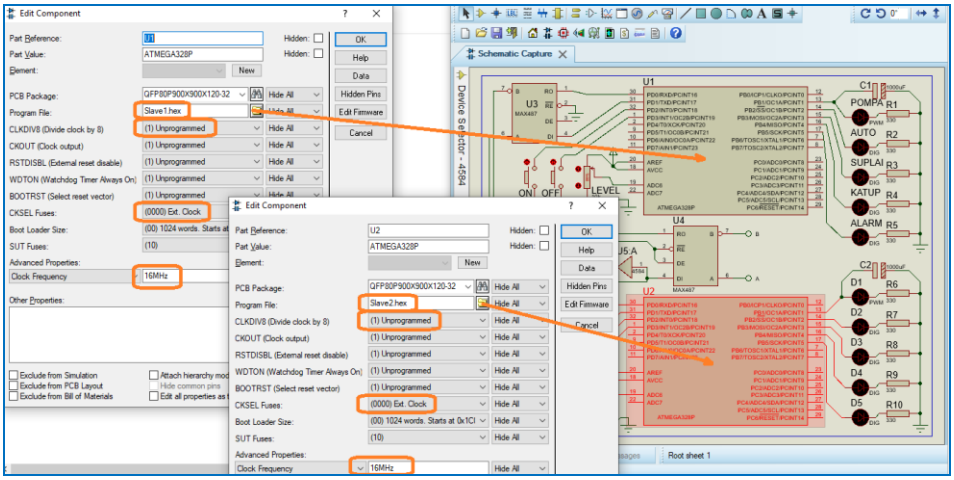
2. Buat program untuk Slave (U1) dengan alamat Modbus = 1 menggunakan program Outseal seperti berikut:



Gambar 3.39 Program Outseal untuk Modbus Slave Tampilan Pengisian Tangki

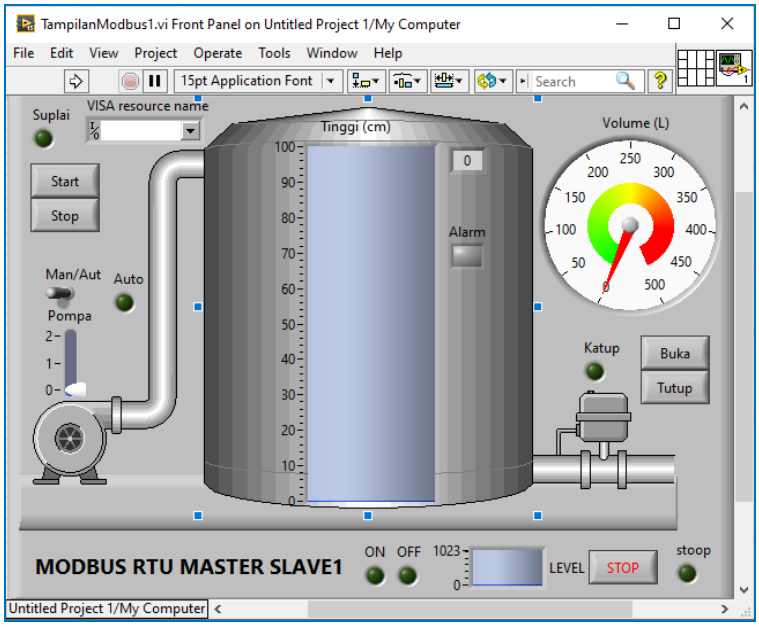
Keterangan: Program Outseal di atas hampir sama seperti program Outseal pada Gambar 3.24, hanya di sini ada penambahan kontak S6, memori internal B2, B3, B4, dan Relay R8, R1, R2, dan R3, mengikuti rangkaian Proteus pada Gambar 3.38. Instruksi Subtract juga ditambahkan untuk membalik nilai PWM.

3. Kompilasi program di atas, dan pindahkan file hex hasil kompilasi ke lokasi yang sama dengan file Proteus Gambar 3.38 di atas, dan ubah namanya menjadi unik, misalnya Slave1.hex, kemudian isikan nama Slave1.hex itu ke dalam kolom Program File komponen U1 rangkaian Gambar 3.38.
4. Ulangi langkah di atas untuk alamat Modbus = 2. Kompilasi program, ubah namanya menjadi Slave2.hex, dan pindahkan file hex hasil kompilasi ke lokasi yang sama dengan file Slave1.hex, dan isikan nama Slave2.hex itu ke dalam kolom Program File komponen U2 rangkaian Gambar 3.38.

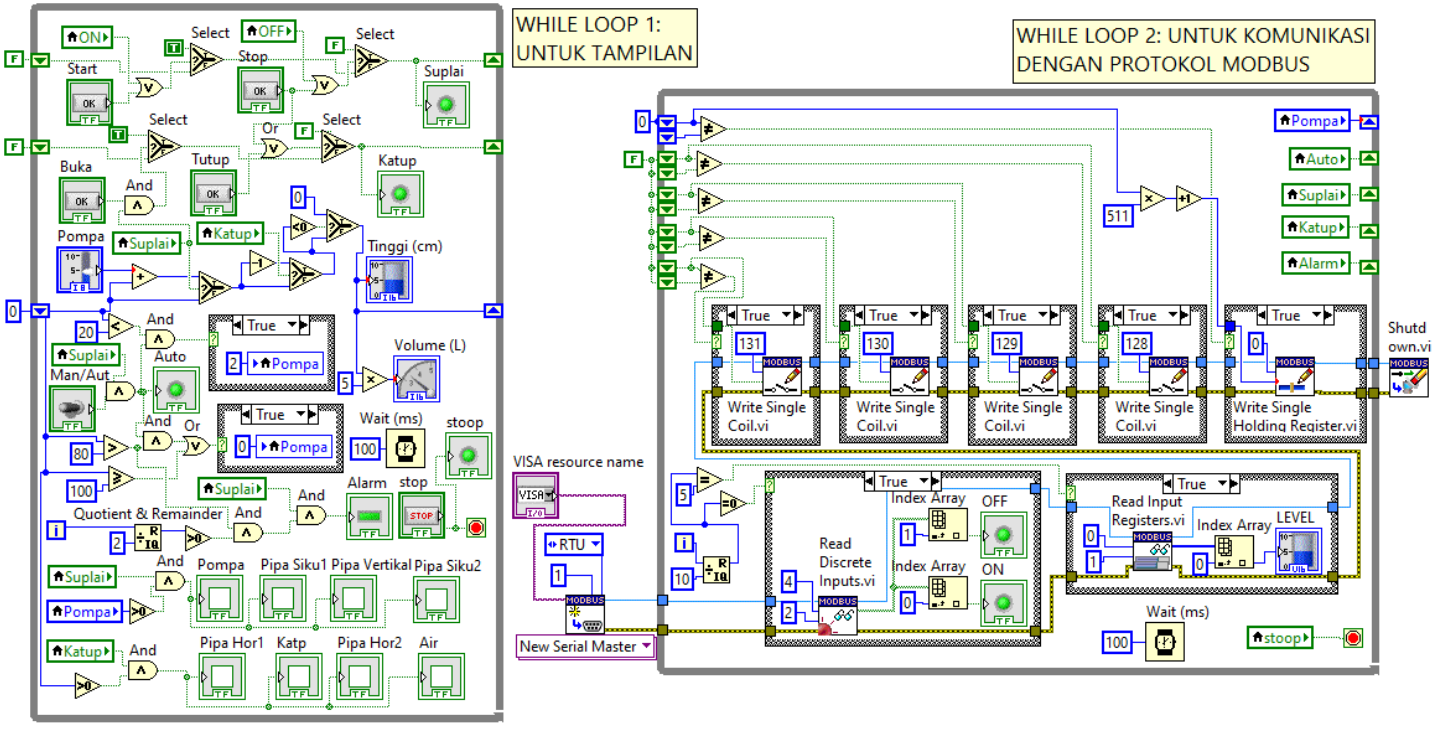


Gambar 3.40 Kolom Program File U1 diisi Slave1.hex, dan Program File U2 diisi Slave2.hex

5. Setelah rangkaian Slave di Proteus selesai diprogram dengan program Outseal, maka berikutnya membuat program Master LabVIEW. Berikut ini program Tampilan Pengisian Tangki menggunakan protokol Modbus Master RTU, dengan unit ID diisi 1 (untuk pengaturan Slave1).



Gambar 3.41 Front Panel program Modbus RTU Master untuk Slave1



Gambar 3.42 Block Diagram program Modbus RTU Master untuk Slave1, dengan 2 buah While Loop. While Loop yang kiri untuk mengatur Tampilan Pengisian Tangki, sedangkan While Loop yang kanan untuk mengatur komunikasi Modbus RTU antara Master dengan Slave1. Terlihat di While Loop kedua, setiap fungsi Modbus RTU diberi Struktur Case, agar komunikasi menjadi lebih efisien, yaitu program hanya mengirimkan data objek yang nilainya berubah saja, dan tidak sepanjang waktu menunggu data, tetapi memberikan sedikit waktu secara berkala untuk menerima data.

Keterangan program: Terlihat pada program Gambar 3.42 di atas, setiap fungsi modbus RTU di While Loop yang kanan, diberi struktur Case. Untuk Case False pada semua Struktur Case tersebut, buat seperti Gambar 3.28, yaitu garis data di kiri diteruskan di kanannya. Penambahan struktur Case membuat komunikasi modbus menjadi lebih efisien, karena program hanya mengirimkan data yang nilainya berubah saja, dan tidak sepanjang waktu menunggu data, tetapi memberikan sedikit waktu secara berkala untuk menerima data.

Ada 4 buah fungsi Write Single Coil, dengan alamat berturut-turut 128, 129, 130 dan 131. Keempat fungsi Write Single Coil ini digunakan untuk mengatur nilai B1, B2, B3 dan B4 pada program Outseal, yang harusnya memiliki alamat 129, 130, 131 dan 132. Berhubung program Modbus RTU di LabVIEW dimulai dari alamat 0, maka alamat 129, 130, 131 dan 132 dikurangi 1 menjadi 128, 129, 130 dan 131.

Fungsi Read Discrete Input digunakan untuk membaca kontak S5 dan S6 pada program Outseal. Seharusnya S5 dan S6 ada di alamat 5 dan 6 memori Input Diskrit, namun karena LabVIEW memulai indeks dari 0, maka alamatnya menjadi 4 dan 5, untuk itu alamat di fungsi Read Discrete Input diisi 4, dengan jumlah 2.

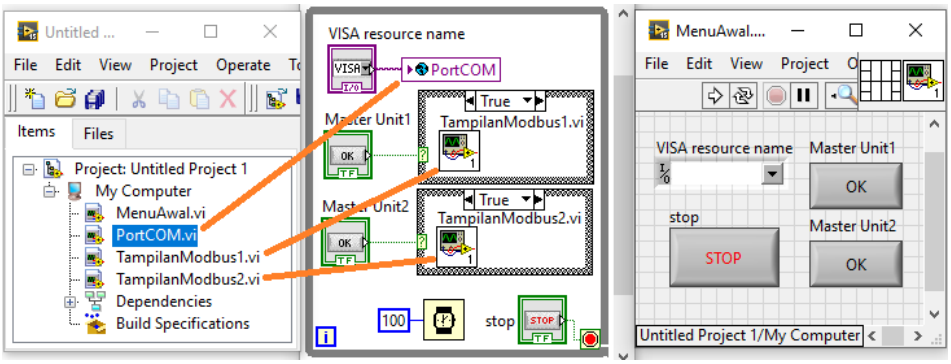
Fungsi Read Input Register digunakan untuk membaca input analog A.1 Outseal, yang merupakan kaki A6 Arduino, yang terhubung dengan potensio. Karena LabVIEW memulai alamatnya dari 0, maka A.1 ada di alamat 0 Input Register.

Fungsi Write Single Register digunakan untuk mengatur nilai Duty Cycle PWM di alamat I.1. Karena dimulai dari 0, maka I.1 ada di alamat 0 Holding Register.

6. Setelah program LabVIEW Gambar 3.41 dan 3.42 selesai dibuat, simpan program dengan nama TampilanModbus1.vi.
7. Berikutnya, buka menu Project, pilih Create Project. Pada jendela Create Project, pilih Blank Project. Muncul pesan, “You have one or more …”, pilih Add, muncul kotak Project dengan TampilanModbus1 dalam daftar.
8. Save As TampilanModbus1.vi, kemudian beri nama TampilanModbus2.vi.
9. Di Block Diagram TampilanModbus2.vi, ubah angka 1 di kaki input unit ID icon Create Modbus, menjadi angka 2, kemudian simpan program.

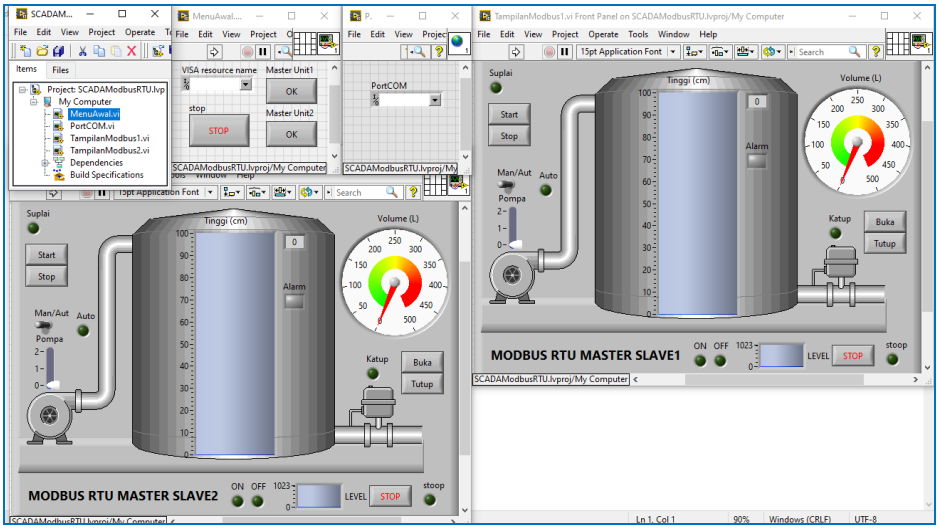
10. Pada kotak Project, klik kanan My Computer, pilih Add, pilih File, arahkan pencarian pada file TampilanModbus1.vi, klik Add File, maka di kotak Project muncul nama TampilanModbus1.vi dan TampilanModbus2.vi.
11. Berikutnya buat program MenuAwal.vi, yang sama seperti program Awalan.vi di Sub Bab 3.4 (lihat Gambar 3.31). Untuk membuatnya, klik kanan My Computer di kotak Project, pilih New, pilih VI.
12. Di Front Panel MenuAwal.vi, tempatkan sebuah VISA Resource Name yang diambil dari palet Controls → Modern → I/O. Ambil dan tempatkan juga 2 buah OK Button, beri nama Master Unit1 dan Master Unit2.
13. Di Block Diagram MenuAwal.vi, ambil dan tempatkan sebuah While Loop, dan masukkan ketiga icon (VISA Resource Name dan 2 OK Button) ke dalam While Loop. Tambahkan tombol Stop pada Terminal Loop Condition, dan icon Wait(ms) dengan input 100.
14. Kemudian tambahkan 2 buah Struktur Case, dan hubungkan Case Selector Struktur Case pertama dengan tombol Master Unit1 dan Case Selector Struktur Case kedua dengan Master Unit2 (lihat Gambar 3.32).
15. Isi Struktur Case pertama dengan icon TampilanModbus1.vi, dan Struktur Case kedua dengan TampilanModbus2.vi, yang dibuat dengan cara menarik nama keduanya dari kotak Project hingga ke Block Diagram.
16. Berikutnya, untuk icon VISA Resource Name, diinginkan isi data VISA Resource Name ini dapat diteruskan ke program TampilanModbus1.vi dan TampilanModbus2.vi. Agar dapat meneruskan data ke program vi lain tersebut, dibutuhkan sebuah Global Variable. Buat Global Variable, dengan meng-klik kanan My Computer di kotak Project, pilih New, pilih New lagi, dan kemudian pada kotak Create New, pilih Global Variable.
17. Di Front Panel Global Variable, tempatkan VISA Resource Name yang diambil dari palet Controls → Modern → I/O. Ubah label VISA Resource Name menjadi PortCOM. Tutup Front Panel Global Variable dengan menekan tombol silang di pojok kanan, dan simpan dengan nama PortCOM, maka akan muncul nama PortCOM pada kotak Project.
18. Berikutnya, tarik nama PortCOM dari kotak Project, hingga ke Block Diagram MenuAwal.vi, dan tempatkan di dalam While Loop.

19. Hubungkan icon VISA Resource Name dengan icon PortCOM.



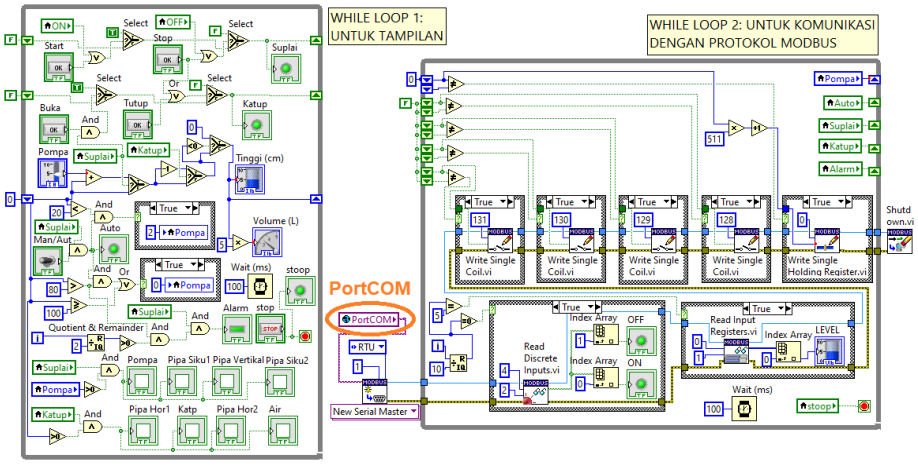
Gambar 3.43 Program MenuAwal.vi, menarik PortCOM dari kotak Project ke Block Diagram

20. Berikutnya, hilangkan atau hapus kotak VISA Resource Name di Front Panel program TampilanModbus1.vi dan TampilanMobdus2.vi.
21. Sebagai ganti VISA Resource Name yang telah dihapus, gunakan PortCOM yang ada di kotak Project. Caranya, buka Block Diagram, tarik PortCOM dari kotak Project dan tempatkan pada lokasi yang sama dengan icon VISA Resource Name yang telah dihapus. Berikutnya ubah kaki input icon PortCOM menjadi kaki output, dengan cara meng-klik kanan icon, pilih Change to Read. Selanjutnya, hubungkan kaki output ini dengan kaki input VISA Resource Name icon Create Modbus. Lakukan penggantian ini pada program TampilanModbus1.vi dan TampilanModbus2.vi.
22. Berikutnya, agar tombol Run Continuously hilang dari tampilan program MenuAwal.vi, TampilanModbus1.vi dan TampilanModbus2.vi, maka di ketiga program tersebut, ubah Window Appearance dari Default menjadi Dialog. Untuk membuka Window Appearance, pilih menu File, pilih VI Properties, pilih Category Window Appearance.
23. Terakhir, di kotak Project, tekan tombol Save All (this project), dan beri nama Project: ScadaModbusRTU.lvproj.
24. Gambar berikut menunjukkan tampilan Front Panel keempat program yang ada di kotak Project berturut-turut dari kiri ke kanan, atas ke bawah: program MenuAwal.vi, Global variable PortCOM, TampilanModbus1.vi dan TampilanModbus2.vi.



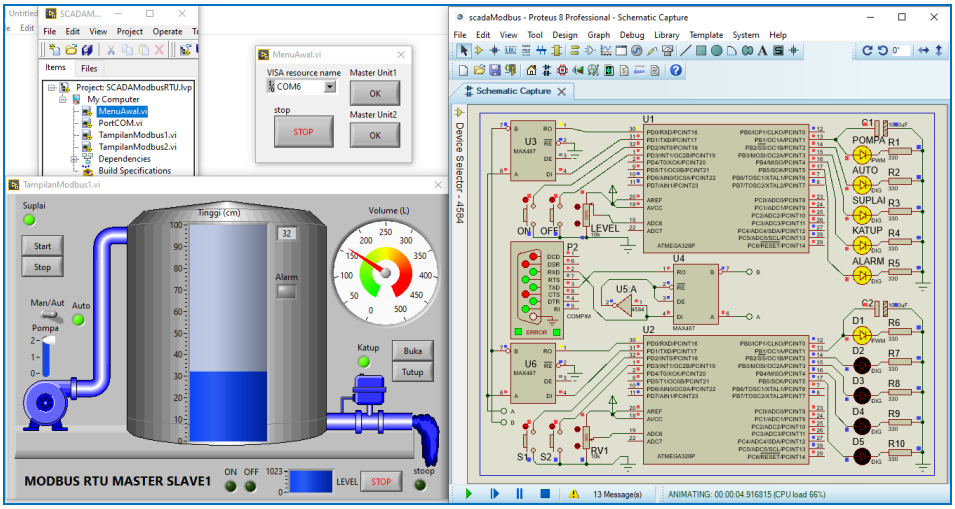
Gambar 3.44 Front Panel program-program yang ada dalam daftar Project SCADAMobusRTU

25. Perhatikan Gambar 3.44 di atas, sudah tidak ada tombol Run Continuously di semua Front Panel, dan juga sudah tidak ada lagi kotak VISA Resource Name di program TampilanModbus1.vi dan TampilanModbus2.vi.
26. Gambar berikut menunjukkan Block Diagram dari TampilanModbus1.vi, yang sama seperti Gambar 3.42, hanya beda pada icon VISA Resource Name, yang sudah diganti dengan icon PortCOM.



Gambar 3.45 Block Diagram TampilanModbus1.vi dengan icon VISA Resource Name yang telah digantikan dengan icon Global Variable PortCOM

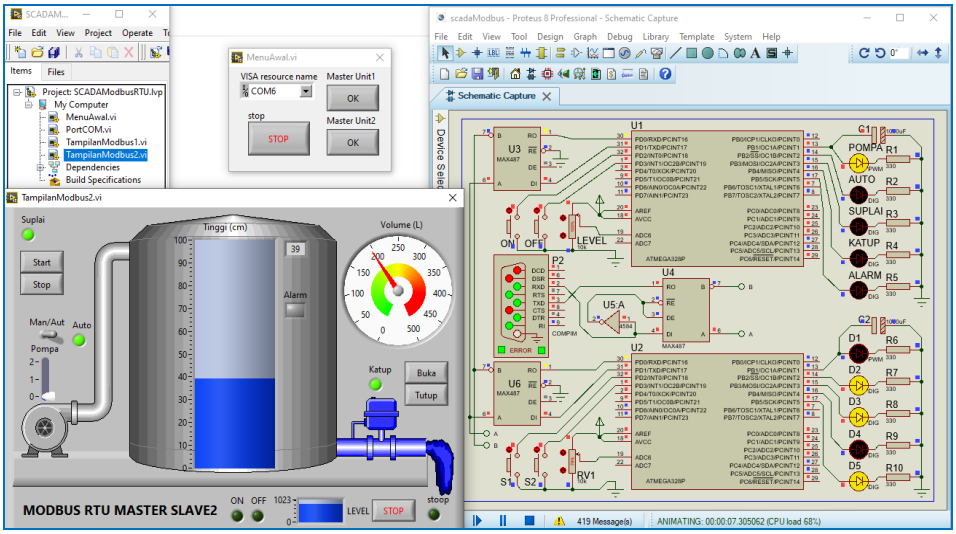
27. Berikutnya, jalankan program, baik rangkaian di Proteus maupun program LabVIEW. Namun sebelum menjalankan program LabVIEW, tutup semua program LabVIEW, kecuali program MenuAwal.vi.
28. Tekan tombol Run program MenuAwal.vi, pilih PortCOM yang merupakan pasangan COM, dan kemudian tekan tombol Master Unit1, maka TampilanModbus1.vi akan muncul.
29. Sementara TampilanModbus1.vi masih berjalan, sebelum menjalankan Proteus, pastikan port COM pada COMPIM adalah COM yang berpasangan dengan COM yang diisikan pada program MenuAwal.vi LabVIEW. Apabila sudah benar, jalankan Proteus.
30. Lakukan pengaturan pada tombol Start, Stop, Mode Manual/Auto Pompa, pengaturan slider pompa, buka/tutup Katup, dan amati perubahan pada rangkaian U1. Lakukan pula pengubahan pada kedua tombol dan potensio di rangkaian U1, dan amati perubahan pada TampilanModbus1.vi.



Gambar 3.46 Ketika tombol Master Unit1 ditekan, muncul TampilanModbus1.vi. Pengaturan pada TampilanModbus1.vi membuat perubahan pada rangkaian U1 di Proteus, sebaliknya penekanan tombol dan potensio di rangkaian U1 mengubah TampilanModbus1.vi

31. Tutup program TampilanModbus1.vi dengan menekan tombol Stop. Kemudian tekan tombol Master Unit2 di program MenuAwal.vi, maka TampilanModbus2.vi akan muncul.

32. Lakukan pengaturan pada tombol Start, Stop, Mode Manual/Auto Pompa, pengaturan slider pompa, buka/tutup Katup, dan amati perubahan pada rangkaian U2. Lakukan pula pengubahan pada kedua tombol dan potensio di rangkaian U2, dan amati perubahan pada TampilanModbus2.vi.



Gambar 3.47 Ketika tombol Master Unit2 ditekan, muncul TampilanModbus2.vi. Pengaturan pada TampilanModbus2.vi membuat perubahan pada rangkaian U2 di Proteus, sebaliknya penekanan tombol dan potensio di rangkaian U2 mengubah TampilanModbus2.vi

33. Sampai di sini Tampilan SCADA 2 Slave dengan Modbus RTU selesai.

3.6 Soal Latihan

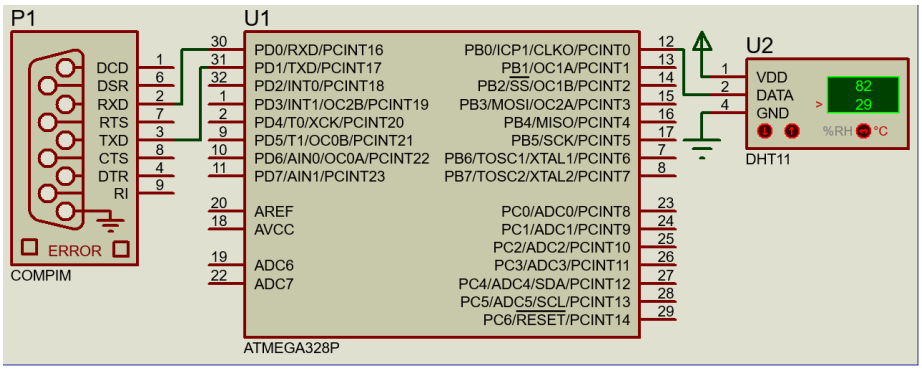
1. Komunikasi serial dengan protokol Modbus antara LabVIEW dengan Arduino dalam Bab ini menggunakan Modbus RTU. Apa yang Anda ketahui tentang Modbus RTU? Apa kelebihannya dibandingkan dengan Modbus ASCII?

2. Berkaitan dengan blok memori pada Modbus:
   a. Tipe blok memori apa yang bisa dibaca dan ditulis?
   b. Tipe blok memori apa yang hanya bisa dibaca?
   c. Tipe blok memori apa yang ukurannya bit?
   d. Tipe blok memori apa yang ukurannya Word?

3. 10 buah sensor suhu akan disambung dengan komunikasi serial RS-485. Agar bisa terbaca oleh Master, setiap sensor harus memiliki alamat atau nomor ID yang berbeda. Secara default (dari pabrik pembuatnya), nomor ID semua sensor tersebut adalah 1, yang ada di alamat memori 100 di Holding Register. Apa kode fungsi yang dapat digunakan untuk mengubah nomor ID kesepuluh sensor tersebut, sehingga kesepuluh sensor tersebut memiliki nomor ID masing-masing secara berurutan dari angka 1 sampai 10 (Gunakan ModRSsim2 sebagai representasi sensor suhunya, dan gunakan QModMaster sebagai Master yang mengatur nomor ID sensor).

4. Sensor suhu pada soal no. 3 di atas, bisa menghasilkan output data suhu dan kelembaban, yang ditempatkan di alamat 100 dan 101 di Input Register. Apa kode fungsi yang dapat digunakan untuk membaca data suhu dan kelembaban tersebut (Gunakan ModRSsim2 sebagai representasi sensor suhunya, dan gunakan QModMaster sebagai Master yang membaca data suhu dan kelembaban).

5. Buat rangkaian berikut ini di Proteus (COMPIM, ATmega328P, DHT11).



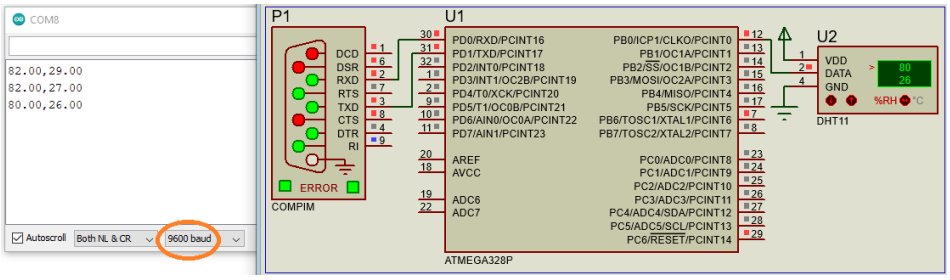
Gambar 3.48 Rangkaian Arduino dengan sensor DHT11 di kaki D8

Gunakan kode program Arduino berikut ini untuk diisikan pada kolom Program File komponen U1 di atas.

No.	Kode Program 04: Baca DHT11
01.	`#include "DHT.h"`
02.	`#define DHTPIN 8`
03.	`#define DHTTYPE DHT11`

```
04.  DHT dht(DHTPIN, DHTTYPE);
05.  void setup() {
06.    Serial.begin(9600);
07.    dht.begin();
08.  }
09.  void loop() {
10.    delay(2000);
11.    float h = dht.readHumidity();
12.    float t = dht.readTemperature();
13.    if (isnan(h) || isnan(t)) return;
14.    Serial.print(h);
15.    Serial.print(',');
16.    Serial.println(t);
17.  }
```

Sebelum melakukan kompilasi, program di atas membutuhkan Library tambahan, yang bernama DHT sensor library. Download file ini dari internet, atau dapat juga pembaca download dari blog penulis. Untuk menambahkan library ini ke software Arduino, buka menu Sketch, pilih Include Library, pilih Add ZIP. Library. Arahkan pencarian pada file DHT_sensor_library. Tunggu beberapa saat hingga muncul pesan Library telah ditambahkan. Setelah itu, lakukan kompilasi. Kemudian isikan lokasi file hex ke dalam kolom Program File komponen U1 Gambar 3.48 di atas. Setelah terisi file hex dan parameter lainnya (seperti Gambar 2.17), jalankan Proteus. Gunakan Serial Monitor di software Arduino untuk melihat data yang dibaca. Agar bisa terhubung, pastikan Port COM di COMPIM dan Port COM yang dipilih di software Arduino (pilih menu Tools, pilih Port) merupakan COM pasangan. Setelah itu buka Serial Monitor, harusnya Serial Monitor menampilkan data kelembaban diikuti data suhu, yang dipisahkan dengan koma, seperti gambar berikut:



Gambar 3.49 Serial Monitor menampilkan data kelembaban dan suhu DHT11

Sebagai latihan, buatlah program LabVIEW yang menggantikan Serial Monitor Gambar 3.49 di atas, sehingga data kelembaban maupun data suhu dapat ditampilkan di LabVIEW dalam bentuk objek Numeric, misalnya objek Gauge untuk data kelembaban (0-100%) dan Thermometer untuk data suhu.

6. Ulangi Soal Latihan no. 5 di atas, namun dengan menerapkan protokol Modbus, dimana QModMaster sebagai Modbus Master, dan Arduino sebagai Modbus Slave. Modifikasi kode program Arduino 03 (ModbusSlave) di Sub Bab 3.3 dan gabungkan dengan kode program Arduino 04 (Baca DHT11), sehingga data kelembaban dapat diletakkan di alamat 100 Input Register, dan data suhu di alamat 101 Input Register. Gunakan library Modbus Serial yang dibuat oleh Andre Sarmento Barbosa dkk., yang dapat diunduh di alamat: github.com/andresarmento/modbus-arduino, atau di blog penulis, dengan nama file modbus-arduino-master.zip.

7. Ulangi Soal Latihan no. 6 di atas, namun sebagai ganti QModMaster, gunakan program Modbus Master LabVIEW yang dapat membaca data di alamat 100 dan 101, dan menampilkan datanya dalam bentuk objek Numeric, seperti Gauge untuk data kelembaban (0-100%) dan Thermometer untuk data suhu.

8. Ulangi Soal Latihan no. 6 di atas, namun dengan 2 buah rangkaian Arduino, dan masing-masing Arduino terhubung dengan sensor DHT11. Diinginkan masing-masing sensor DHT11 di kedua Arduino tersebut dapat dibaca oleh QModMaster. Gunakan sambungan RS-485 untuk menghubungkan QModMaster dan dua Arduino sebagai Slave1 (ID=1) dan Slave2 (ID=2).

9. Ulangi Soal Latihan no. 8 di atas, namun sebagai ganti QModMaster, gunakan 2 buah program Modbus Master LabVIEW; Master1 untuk membaca Slave1 dan Master2 untuk membaca Slave2, dan tampilkan data kelembaban dan suhu tersebut dalam bentuk objek Numeric, seperti Gauge dan Thermometer.

10. Ulangi Soal Latihan no. 9 di atas, namun dengan menambahkan Project, yang menampung kedua program Modbus Master LabVIEW di atas, dan program menu pembuka, serta sebuah Global Variable, seperti Sub Bab 3.5.

3.7 Refleksi

1. Menurut Anda, dari isi yang diuraikan, apakah **Target Materi** dari Bab 3 buku ini tercapai? Jika belum tercapai, apakah ada kesulitan dalam memahami materi yang berkaitan dengan Target Materi tersebut? Apakah Anda memiliki saran dan masukan untuk memperbaiki materi tersebut?
2. Apakah **Tantangan** dalam Bab 3 buku ini dapat Anda selesaikan? Jika belum, kesulitan apa yang membuat Anda tidak bisa menyelesaikannya? Apakah Anda mencoba alternatif lain untuk menemukan sendiri cara penyelesaiannya (mencari di Google misalnya)?
3. Apakah ada **Manfaat** yang Anda dapatkan setelah mempelajari Bab 3 dari buku ini? Apakah ada yang ingin Anda pelajari lebih lanjut? Apakah ada **Ide** yang menarik yang ingin Anda kembangkan?

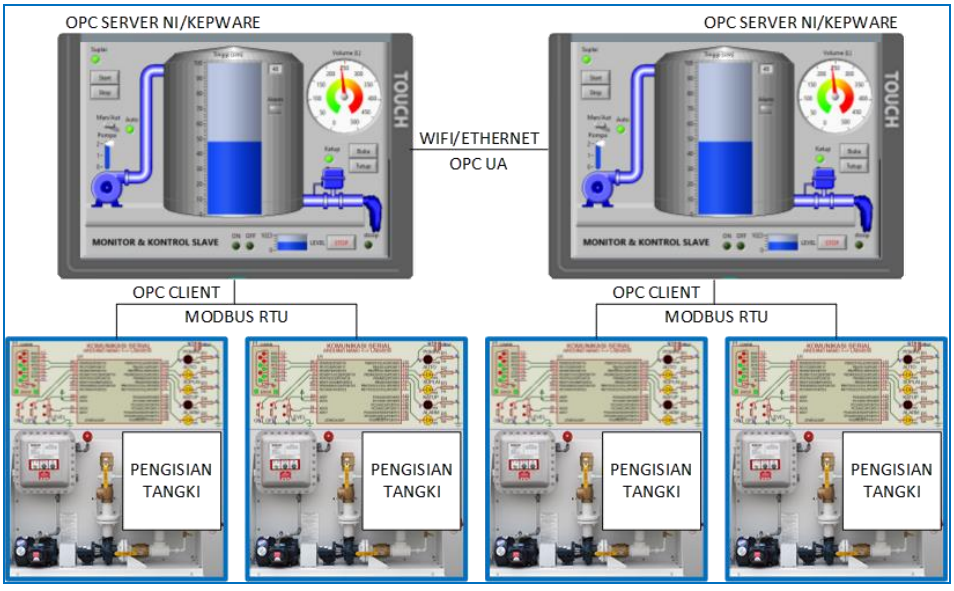
BAB 4
OPC SERVER

Target Materi:

- Menggunakan OPC Server NI untuk penerapan Modbus RTU dan OPC UA.
- Menggunakan OPC Server Kepware untuk Modbus RTU dan OPC UA.

Tantangan:

- Menghubungkan 2 buah jaringan Modbus RTU menggunakan OPC UA, yang bisa terhubung melalui WiFi atau Ethernet (LAN).



Gambar 4.1 Menghubungkan 2 buah jaringan Modbus RTU dengan OPC UA

4.1 Persiapan Perangkat yang Diperlukan

Di samping menggunakan perangkat lunak di Bab sebelumnya, Bab ini memerlukan tambahan perangkat lunak/software sebagai berikut:

5. Software NI OPC Server
6. Software Kepware KEPServerEX

Software NI OPC Server ini sebenarnya sudah diinstal di Bab 1. Ketika pembaca menginstal software Add-On DSC (*Datalogging and Supervisory Control*) Module, NI OPC Server ini otomatis terinstal. Software NI OPC Server ini dibuat oleh National Instruments dengan bantuan Kepware, agar memudahkan pengguna LabVIEW dalam menghubungkan dan meng-integrasikan berbagai perangkat kontrol di LabVIEW. Terdapat 150 lebih protokol atau driver perangkat yang tersedia, termasuk protokol Modbus, yang dapat menghubungkan perangkat Modbus tanpa perlu pemrograman yang rumit.

Software kedua adalah software OPC Server KEPServerEX yang dibuat oleh Kepware. Karena pembuatnya sama, software KEPServerEX ini mirip sekali dengan NI OPC Server. Sekalipun mirip dengan NI OPC Server, namun software KEPServerEX ini menyediakan lebih banyak layanan dibandingkan NI OPC Server. Salah satu contoh layanannya adalah IoT Gateway (mencakup REST Server, REST Client dan MQTT), yang akan dibahas di Bab 6. Versi demo dari Software KEPServerEX ini dapat didownload di alamat ini: www.kepware.com/en-us/products/kepserverex atau bisa juga didownload dari halaman blog penulis.

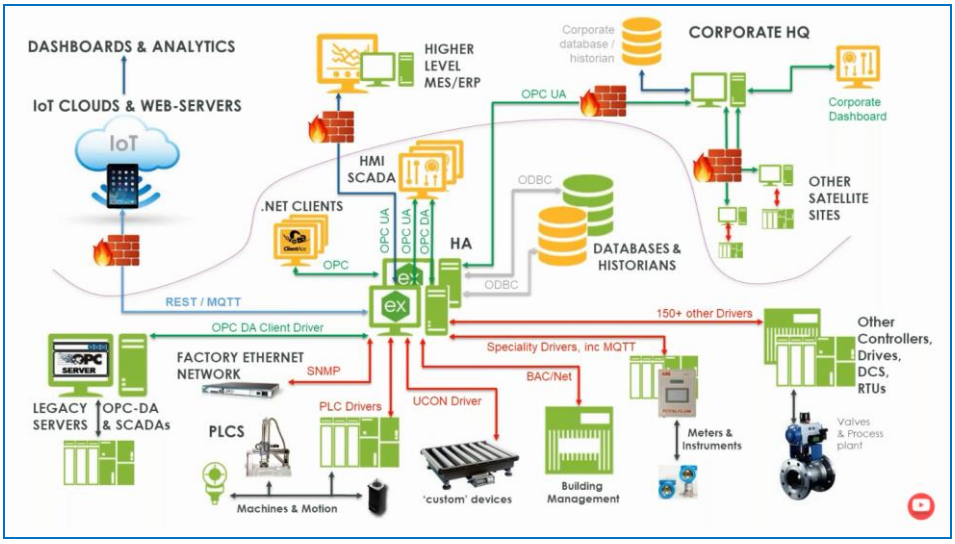
4.2 OPC dengan NI OPC Servers

Apa itu OPC? OPC merupakan singkatan dari *Open Platform Communications*, yaitu sebuah standar industri otomasi untuk interoperabilitas. Interoperabilitas adalah kemampuan dari dua atau lebih sistem atau komponen untuk berbagi pakai data atau informasi.

OPC pertama kali dikembangkan di tahun 1996, yang mula-mula merupakan singkatan dari *OLE for Process Control*. OPC ini digunakan untuk mengatur

komunikasi data secara riil antara berbagai perangkat kontrol dari pabrikan yang berbeda. Karena dalam perkembangannya, OPC tidak hanya digunakan pada bidang "Process Control" tetapi juga di banyak aplikasi, di tahun 2011, OPC Foundation mengubah nama OPC dari OLE for Process Control menjadi *Open Platform Communications*. Perubahan nama tersebut juga mencerminkan teknologi yang digunakan, yang sudah mengintegrasikan berbagai aplikasi seperti .NET Framework, XML, SNMP, ODBC, Web Services, dll.

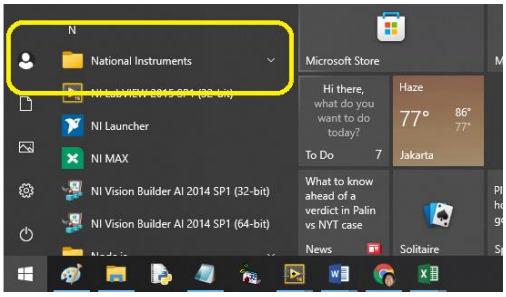
OPC menggunakan teknologi Client-Server. OPC Server berlaku sebagai penyedia data, sedangkan OPC Client berlaku sebagai pengguna data. Seperti diketahui, setiap vendor perangkat otomasi memiliki protokol dan driver perangkatnya sendiri. Sehingga penggabungan perangkat dari vendor yang berbeda menjadi sulit karena begitu beragamnya protokol dan driver. Dengan adanya OPC membuat penggabungan perangkat dari vendor-vendor yang berbeda tersebut menjadi mudah. Setiap vendor akan memasukkan driver dan protokolnya di OPC Server, sehingga OPC Server dapat terhubung dengan semua perangkat yang berbeda-beda tersebut. Pengguna kemudian hanya perlu mengakses data atau perangkat yang diperlukan menggunakan OPC Client. Lebih jauh, teknologi OPC memungkinkan penggabungan semua perangkat.



Gambar 4.2 Teknologi OPC dari Kepware (ex) bisa menggabungkan semua perangkat

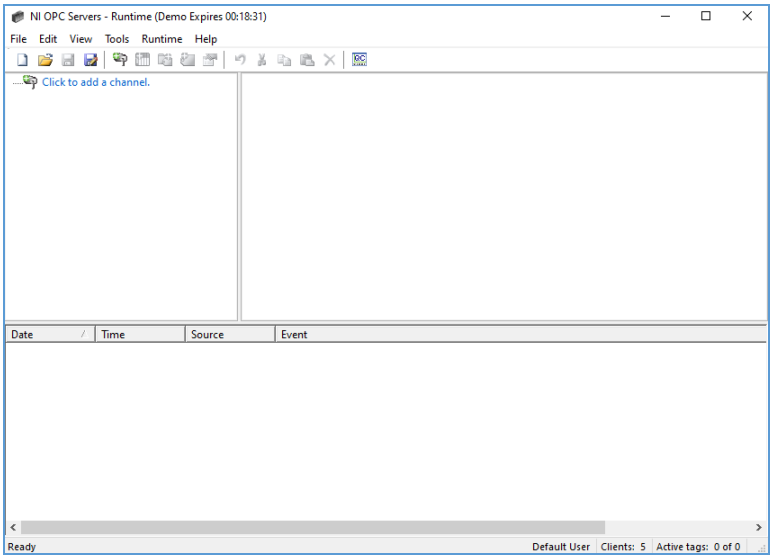
Agar pembaca lebih jelas mengenai penggunaan teknologi OPC ini, berikut ini contoh penerapan teknologi OPC untuk protokol Modbus RTU pada software LabVIEW. Berikut ini langkah-langkahnya:

1. Di Start Menu Program, buka OPC Servers Configuration (ada di dalam folder National Instruments).



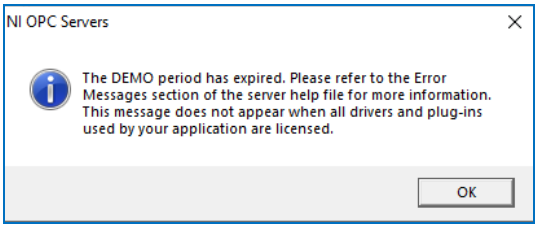
Gambar 4.3 Membuka OPC Server Configuration di folder National Instruments di Start Menu

2. Setelah OPC Server Configuration di-klik, akan muncul NI OPC Servers.
3. Pilih menu File, pilih New, maka akan muncul pertanyaan "*This operation will cause …*". Pilih *Yes, update*, maka isi channel default di kolom kiri akan dikosongkan seperti gambar berikut ini.



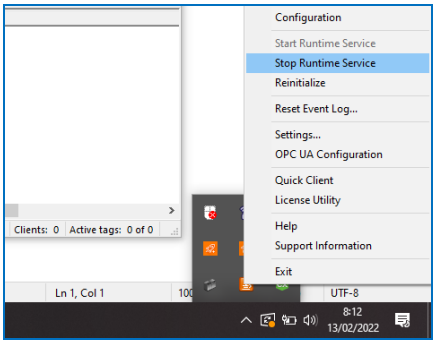
Gambar 4.4 Jendela NI OPC Servers – Runtime, dengan sisa waktu 18 menit 31 detik

4. Runtime NI OPC Server ini dibatasi hanya bisa dijalankan selama 2 jam. Gambar berikut ini menunjukkan tampilan ketika waktu Runtime habis.



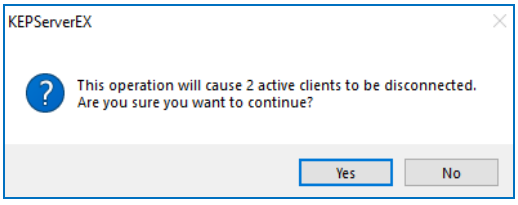
Gambar 4.5 Pesan muncul ketika waktu Runtime sudah habis

5. Apabila waktu Runtime sudah habis, agar bisa dijalankan kembali, perlu dilakukan Reset waktu. Caranya, buka OPC Servers Administration, yang ada di Start Menu, di dalam folder National Instruments.
6. Ketika OPC Servers Administration di Start Menu dibuka, akan muncul icon bergambar kotak hitam di pojok kiri bawah. Klik kanan icon tersebut, maka muncul menu pilihan seperti berikut. Pilih Stop Runtime Service.



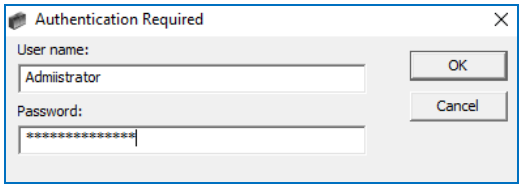
Gambar 4.6 Klik kanan icon OPC Servers Administration, pilih Stop Runtime Service

7. Muncul peringatan pemutusan Client seperti berikut. Pilih Yes.



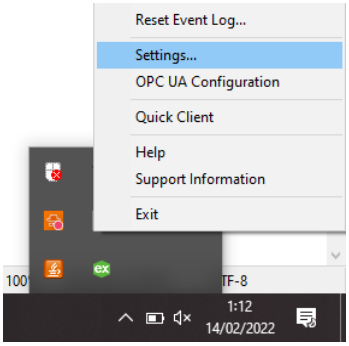
Gambar 4.7 Muncul peringatan akan terjadi pemutusan Client

8. Di NI OPC Server ini, terdapat pembedaan wewenang. Hanya Administrator saja yang berhak melakukan pengubahan seting. Setelah peringatan di atas dipilih Yes, sebelum reset waktu runtime bisa dilakukan, muncul permintaan Otentikasi seperti berikut ini



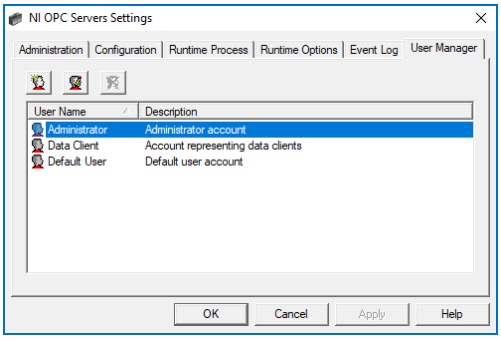
Gambar 4.8 Muncul permintaan Otentikasi untuk bisa mengubah seting

9. Apabila permintaan Otentikasi di atas tidak muncul, berarti NI OPC Server pembaca belum diseting password untuk Administrator. Apabila pembaca tidak menginginkan pengaturan password, silahkan melanjutkan ke langkah no. 12, namun apabila pembaca ingin mengatur password untuk Administrator, silahkan ikuti langkah 10-11 berikut ini.
10. Klik kanan icon kotak hitam di pojok kanan bawah, lalu pilih Settings.

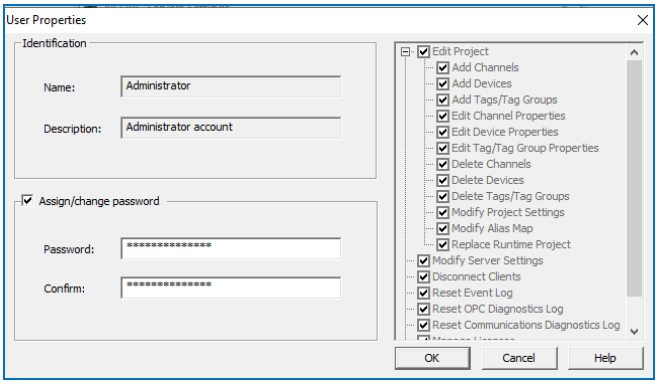


Gambar 4.9 Klik kanan icon kotak hitam (OPC Servers Administration), pilih Settings

11. Muncul jendela NI OPC Servers Settings seperti Gambar 4.10. Klik 2 kali pada User Name Administrator, maka muncul jendela User Properties seperti Gambar 4.11. Beri tanda centang pada Assign/change Password, dan kemudian isi kolom Password dan kolom Confirm dengan password yang sama. Catat password ini, karena akan selalu diminta setiap kali melakukan pengubahan seting pada OPC. Klik OK.

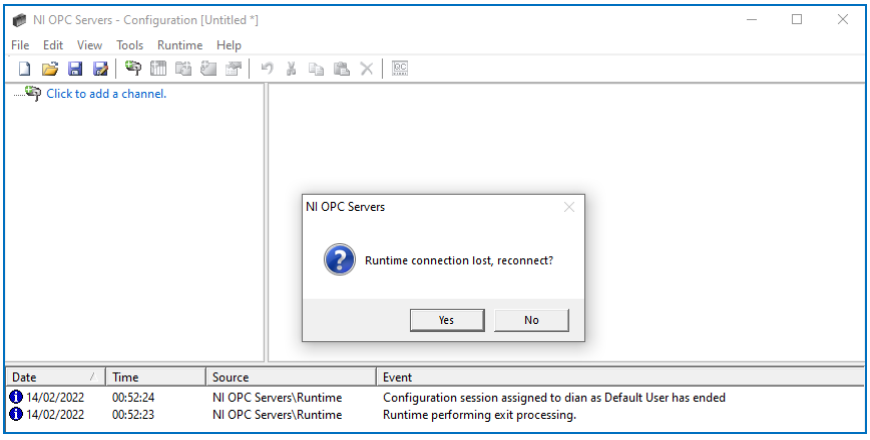


Gambar 4.10 Klik 2 kali pada User Name Administrator



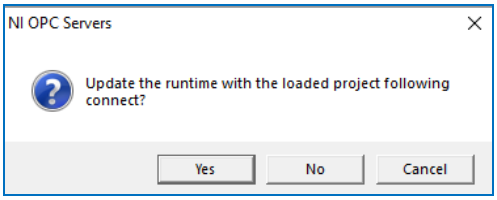
Gambar 4.11 Beri centang Assign/change password, kemudian isi Password dan Confirm

12. Muncul pertanyaan Runtime connection lost, reconnect? Pilih Yes.



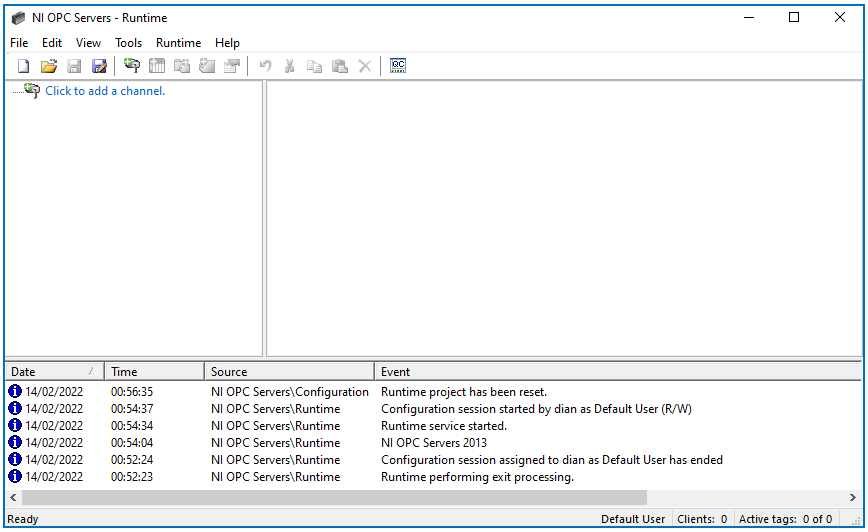
Gambar 4.12 Pilih Yes pada pertanyaan reconnect.

13. Pilih Yes pada pertanyaan "*Update the runtime …*"



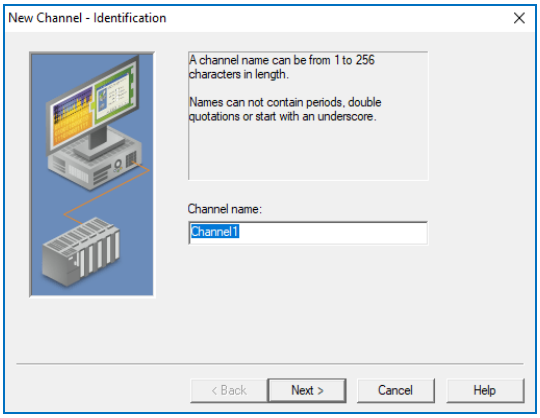
Gambar 4.13 Pilih Yes pada pertanyaan Update the runtime

14. Setelah Yes, maka muncul pesan di Log bahwa Runtime telah direset.

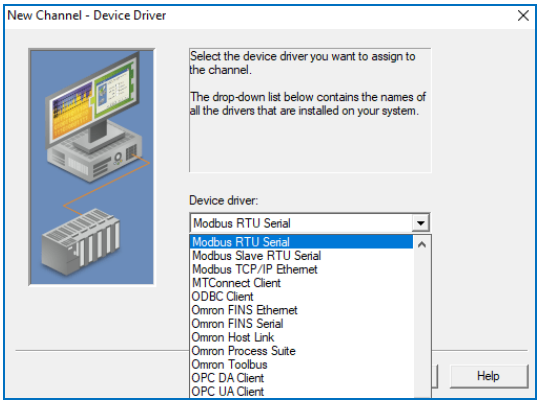


Gambar 4.14 Muncul pesan di Log, Runtime project has been reset

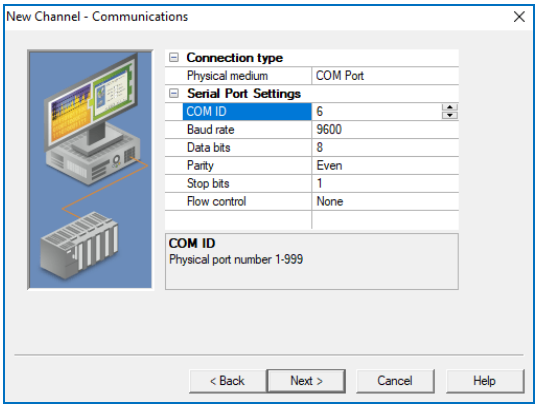
15. Berikutnya, klik pada tulisan *Click to add a channel*, muncul jendela New Channel-Identification, seperti Gambar 4.15. Klik Next.
16. Di kolom Device driver, pilih Modbus RTU Serial, seperti Gambar 4.16.
17. Klik Next, hingga di jendela New Channel-Communications, isi COM ID dengan salah satu dari pasangan COM virtual. Dalam contoh di sini menggunakan pasangan COM5 dan COM6. COM5 diisikan di ModRSSim2, sedangkan COM6 diisikan di kolom COM ID di sini, seperti Gambar 4.17.
18. Klik Next, hingga selesai (Finish), maka akan muncul tulisan Channel1 di kolom kiri jendela NI OPC Servers-Runtime, seperti Gambar 4.18.



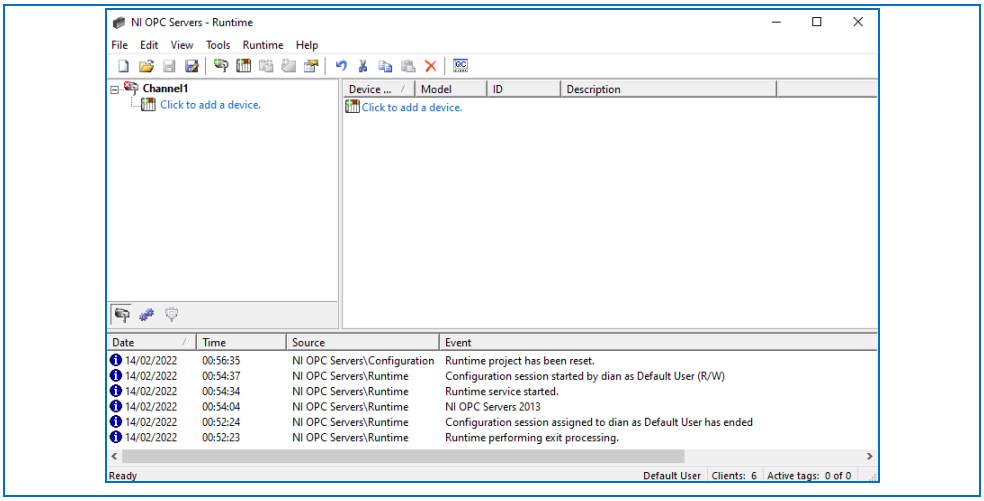
Gambar 4.15 Menambahkan Channel baru, klik Next



Gambar 4.16 Pilih Device driver: Modbus RTU Serial

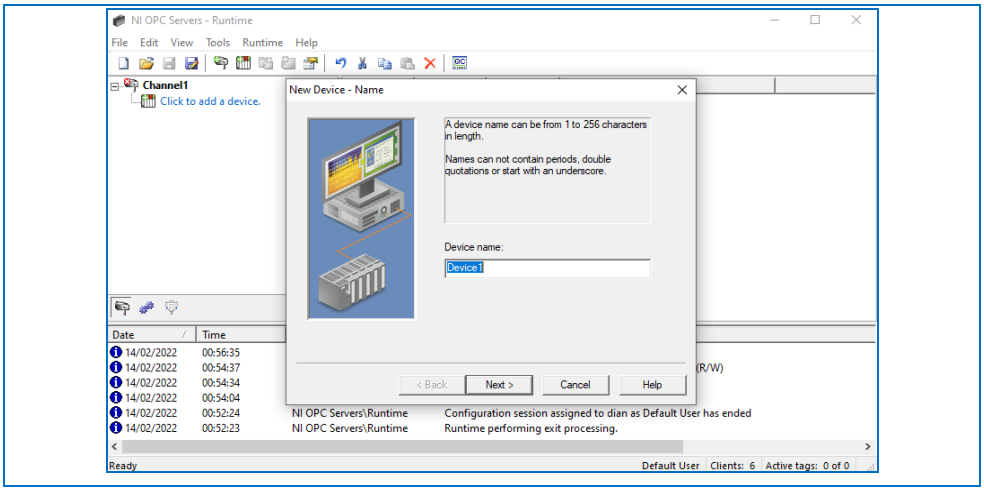


Gambar 4.17 Mengisi COM ID dengan salah satu pasangan COM virtual



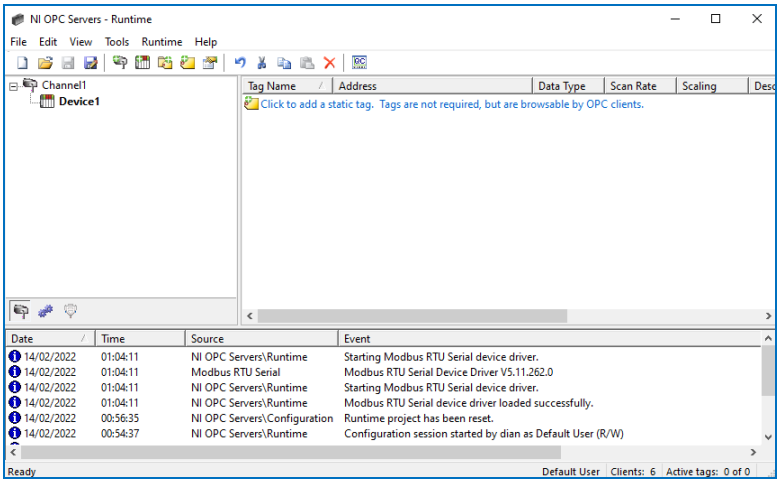
Gambar 4.18 Sebuah Channel baru dengan nama Channel1 muncul di kolom kiri

19. Berikutnya, klik pada tulisan *Click to add a device*, maka akan muncul jendela New Device-Name seperti gambar berikut. Klik Next.

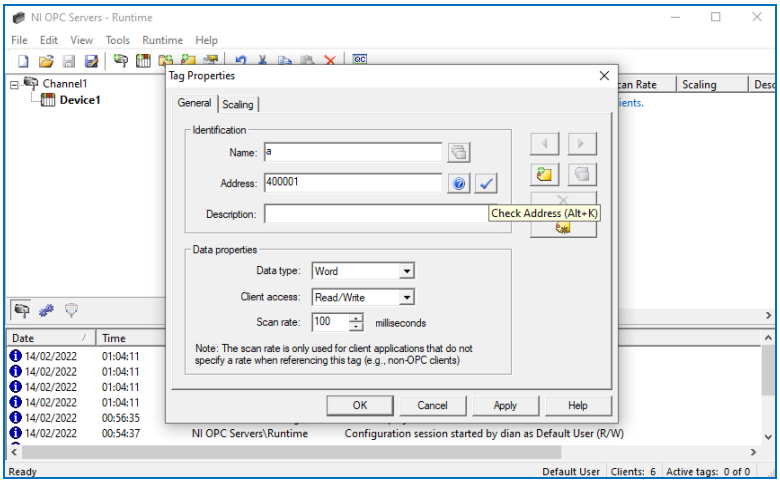


Gambar 4.19 Sebuah Channel baru dengan nama Channel1 muncul di kolom kiri

20. Klik Next hingga selesai (Finish), maka akan muncul tulisan Device1 di bawah nama Channel1 di kolom kiri, seperti Gambar 4.20.
21. Berikutnya, klik pada tulisan *Click to Add a static tag* di kolom sebelah kanan, maka akan muncul jendela Tag Properties, seperti Gambar 4.21.

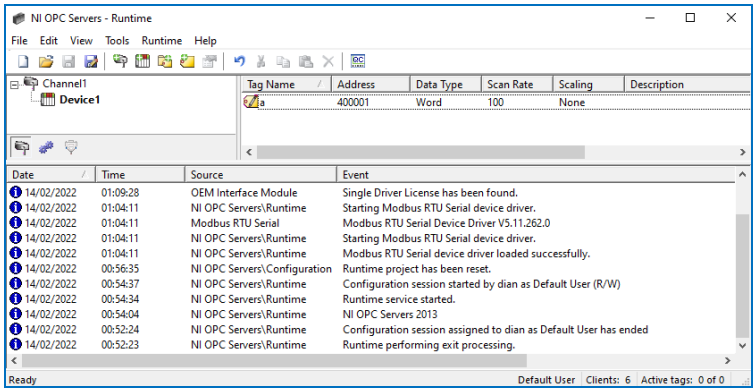


Gambar 4.20 Sebuah Device baru dengan nama Device1 muncul di bawah nama Channel1

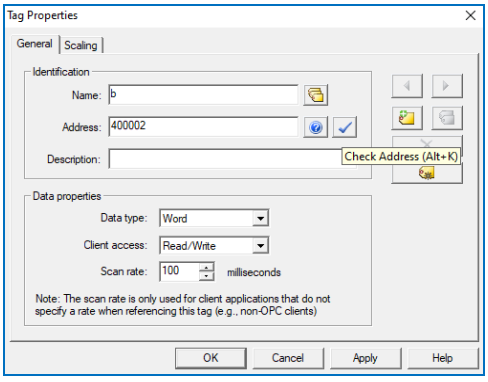


Gambar 4.21 Klik pada tulisan Click to add a static tag, maka muncul jendela Tag Properties

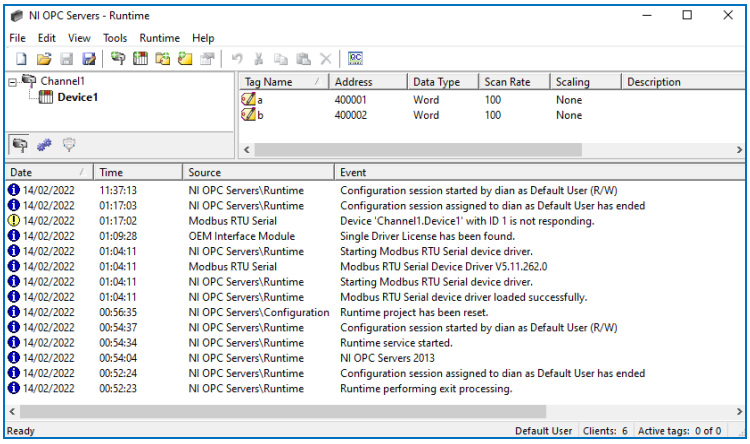
22. Pada jendela Tag Properties, isi Name = a, dan Address = 400001. Tekan tombol centang untuk memastikan format Address benar. Klik OK, maka muncul Tag Name a diikuti data alamat dll., seperti Gambar 4.22.
23. Klik kanan pada kolom di bawah Tag a, pilih New Tag. Pada jendela Tag Properties, isi Name = b, dan Address = 400002, seperti Gambar 4.23.
24. Tekan tombol centang untuk memastikan format Address benar. Klik OK, maka muncul Tag b, di bawah Tag a, seperti Gambar 4.24.



Gambar 4.22 Tag Name a muncul di kolom sebelah kanan

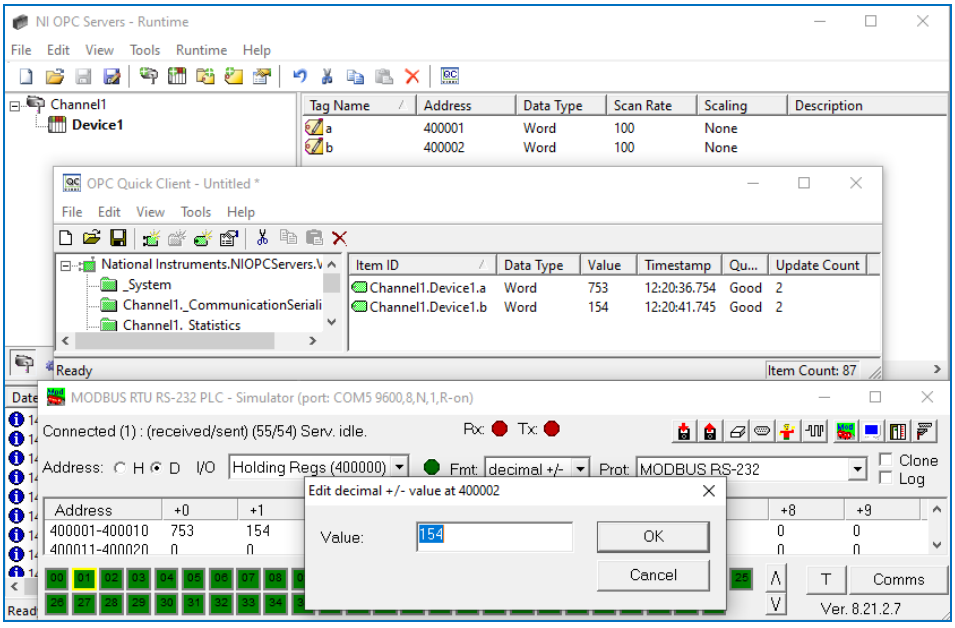


Gambar 4.23 Tambahkan lagi sebuah Tag, beri nama = b dan alamat = 400002, klik OK



Gambar 4.24 Tag b muncul di bawah Tag a, di kolom sebelah kanan

25. Berikutnya, lakukan akses data (*read/write*) kedua Tag ini dengan software simulator ModRSSim2. Buka ModRSSim2, pilih Protokol Modbus RS232. Tekan tombol Set up komunikasi, atur Port sesuai dengan pasangan COM di langkah no. 17.
26. Untuk bisa menampilkan data di OPC, tekan tombol Quick Client, yang ada di ujung kanan Toolbar NI OPC Servers. Maka muncul jendela OPC Quick Client, yang menampilkan nilai data dari Tag-tag di OPC Servers.

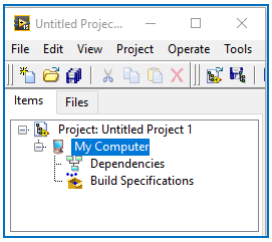


Gambar 4.25 OPC Server untuk mengatur protokol komunikasi, OPC Client untuk mengakses dan menampilkan data OPC, dan ModRSsim2 sebagai simulator perangkat yang terhubung

Catatan: apabila jendela OPC Client tidak menampilkan data, atau kolom nilai data di kolom kanan kosong, hal ini bisa disebabkan oleh karena waktu Runtime habis. Lihat langkah no 6-14 untuk me-reset waktu Runtime.

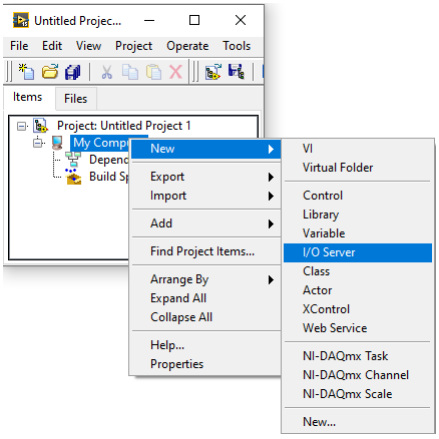
27. Berikutnya, diinginkan agar seting protokol atau driver Modbus RTU Serial di OPC Server, dapat dimasukkan (di-*deploy*) di LabVIEW, sehingga nantinya perangkat Modbus yang disimulasikan oleh ModRSsim2, bisa berkomunikasi dengan LabVIEW, dengan jembatan OPC Server dan Client.

28. Berikut ini langkah-langkahnya: Buka software LabVIEW. Di halaman awal LabVIEW, pilih New atau Create Project, pilih Empty atau Blank Project, maka muncul kotak Project seperti berikut:

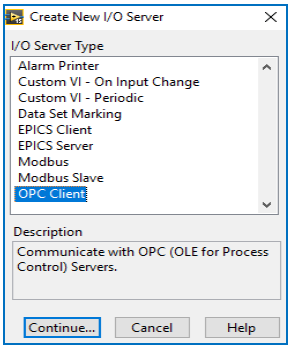


Gambar 4.26 Kotak Project baru

29. Klik kanan My Computer, pilih New, pilih I/O Server, pilih OPC Client.

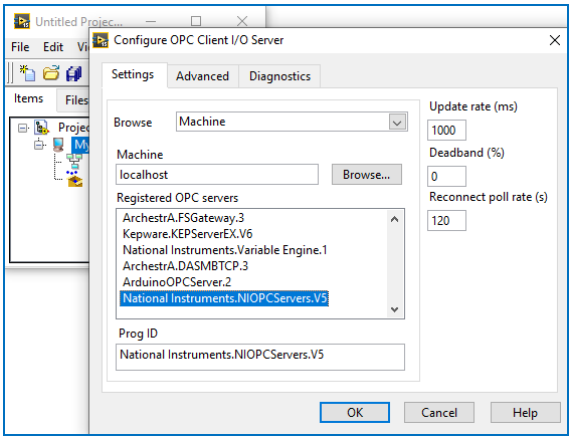


Gambar 4.27 Klik kanan My Computer, pilih New, pilih I/O Server



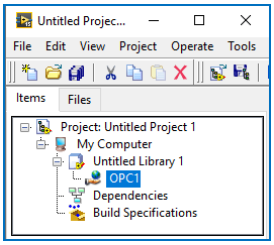
Gambar 4.28 Di jendela Create New I/O Server, pilih OPC Client

30. Di jendela Configure OPC Client I/O Server, pada kolom Registered OPC servers, pilih National Instruments.NIOPCServers.V5.



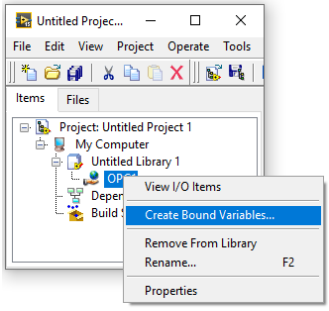
Gambar 4.29 Di Configure OPC Client I/O Server, pilih National Instruments.NIOPCServers.V5

31. Klik OK, maka di kotak Project, di dalam Untitled Library1, muncul OPC1.



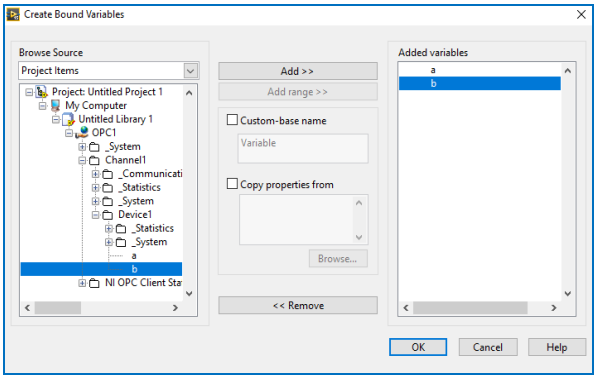
Gambar 4.30 Muncul I/O Server OPC1 di dalam folder Untitled Library1

32. Klik kanan OPC1, pilih Create Bound Variables.



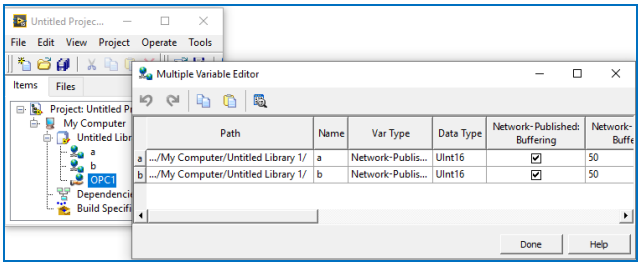
Gambar 4.31 Membuat Create Bound Variables yang terhubung dengan OPC1

33. Di jendela Create Bound Variables, buka Tag a dan b, klik Add.



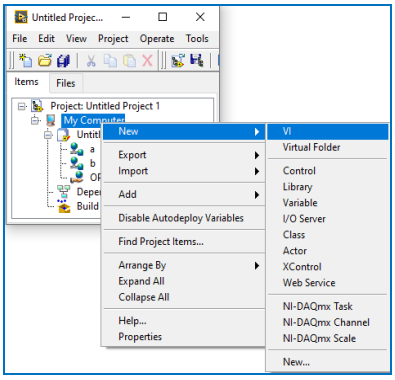
Gambar 4.32 Menambahkan variable untuk program LabVIEW dari Tag a dan b

34. Klik OK, maka LabVIEW menciptakan variable I/O Server dari Tag a dan b.



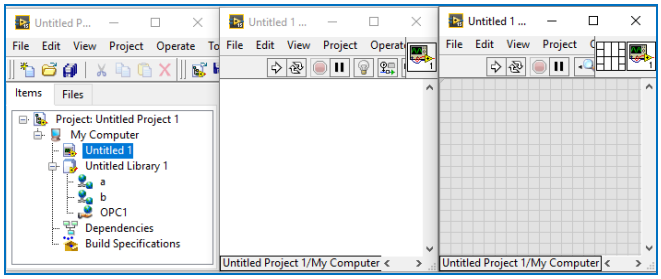
Gambar 4.33 LabVIEW menciptakan variable I/O Server dari Tag a dan b

35. Berikutnya, di kotak Project, klik kanan My Computer, pilih New, pilih VI.



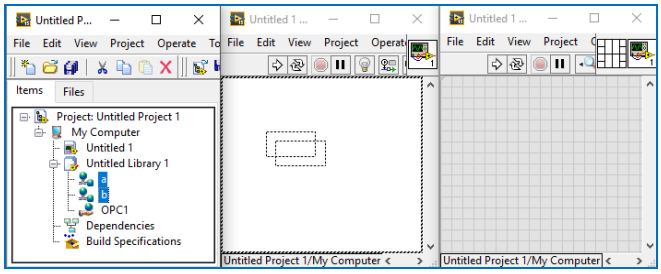
Gambar 4.34 Menambahkan sebuah VI di kotak Project untuk pembuatan program

36. Tempatkan Block Diagram dan Front Panel VI di dekat kotak Project.



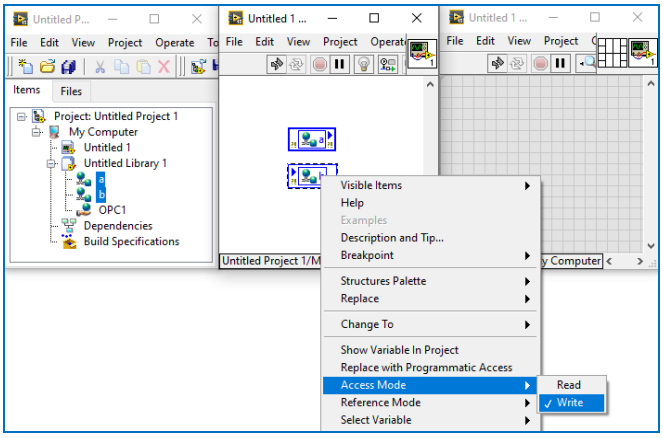
Gambar 4.35 Menempatkan Block Diagram dan Front Panel VI di dekat kotak Project

37. Tarik (*drag*) variable a dan b dari kotak Project ke Block Diagram.



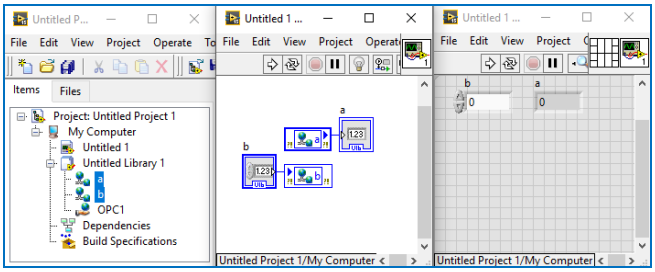
Gambar 4.36 Men-drag variable a dan b dari kotak Project ke Block Diagram

38. Atur mode akses variable b sehingga dapat di-write, dengan klik kanan variable b, pilih Access Mode, pilih Write.



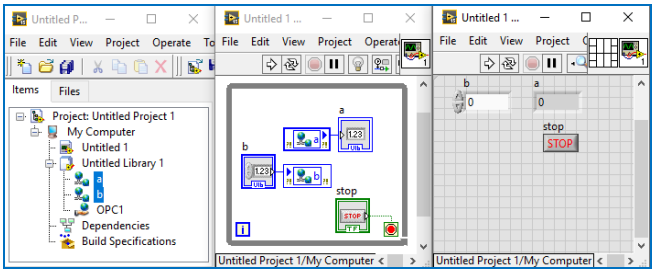
Gambar 4.37 Mengatur mode akses untuk variable b dapat di-Write

39. Klik kanan kaki output variable a, pilih Create, pilih Indicator.
40. Klik kanan kaki input variable b, pilih Create, pilih Control.



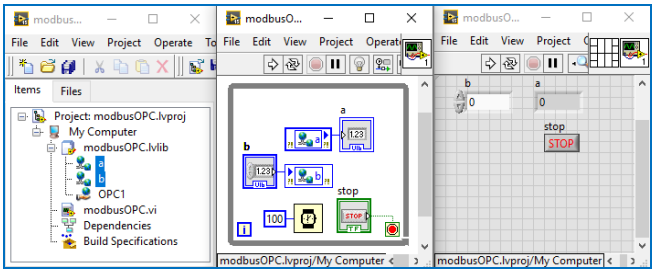
Gambar 4.38 Menambahkan Indicator pada variable a, dan Control pada variable b

41. Agar program bisa berjalan terus-menerus, tambahkan While Loop.
42. Tambahkan tombol Stop agar program bisa dihentikan.



Gambar 4.39 Menambahkan Struktur While Loop dan tombol Stop

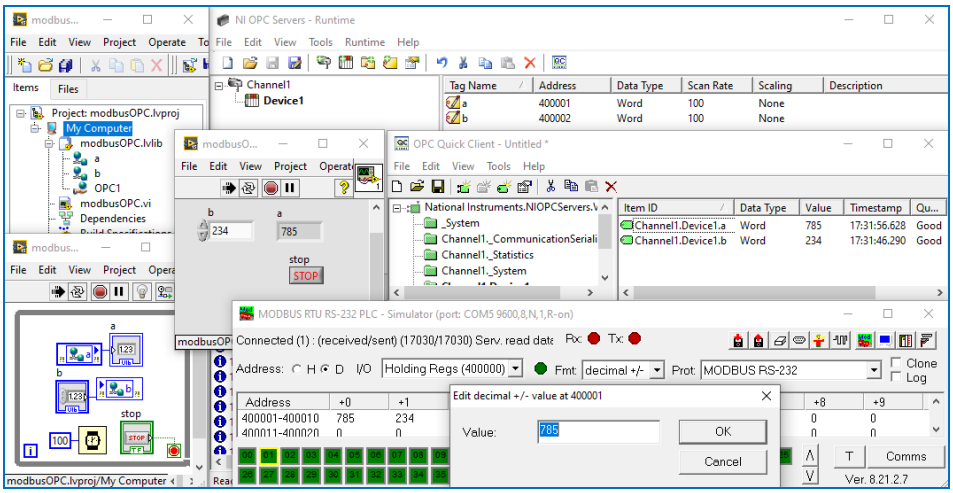
43. Tambahkan waktu tunda dengan fungsi Wait(ms), isi nilai input = 100 ms.



Gambar 4.40 Menambahkan tunda 100 ms untuk menyediakan waktu respon yang cukup

44. Jalankan progam LabVIEW, dan lakukan pengubahan nilai variable b, dan perhatikan nilai Tag b di OPC Quick Client dan ModRSsim2.

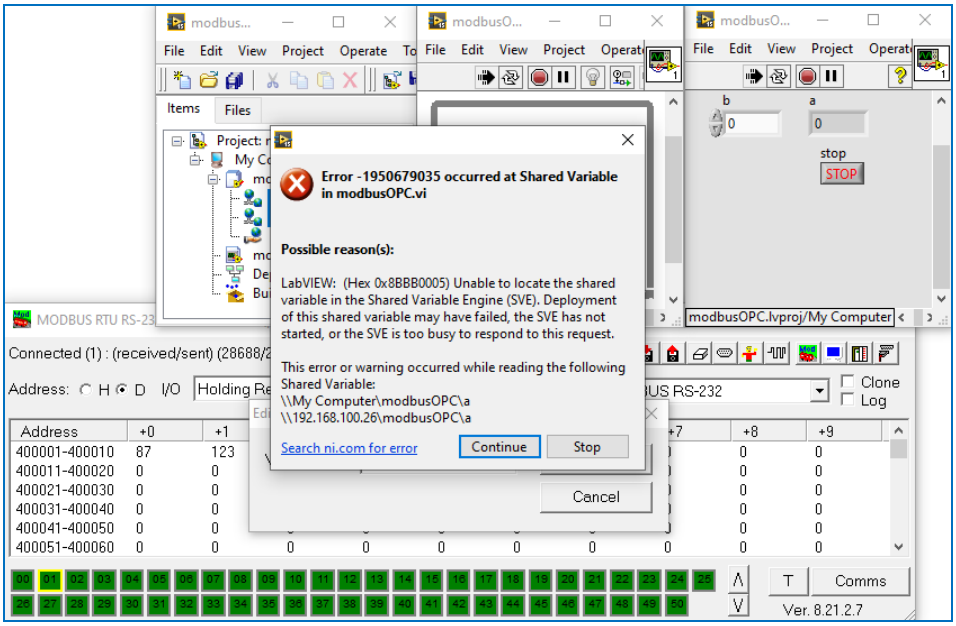
45. Lakukan pengubahan juga pada Tag a dan b di ModRSsim2. Perhatikan bahwa nilai Tag a dapat diubah dari ModRSsim2, hanya nilai Tag b tidak dapat diubah dari ModRSsim2. Pengubahan nilai Tag b hanya dapat dilakukan di program LabVIEW, karena objek untuk variable b di LabVIEW menggunakan objek Control.



Gambar 4.41 Program LabVIEW dapat membaca dan mengatur nilai Tag, baik nilai Tag di OPC Quick Client maupun di ModRSsim2 (yang mensimulasikan perangkat Modbus RTU)

Catatan: perhatikan, bahwa dengan bantuan NI OPC Servers, program LabVIEW untuk komunikasi dengan protokol Modbus RTU menjadi sangat mudah dan sederhana, serta lebih handal, karena driver dan protokol perangkat, serta komunikasi datanya, sudah ditangani dengan sangat baik oleh OPC Server. Program LabVIEW hanya perlu memanggil OPC Client untuk bisa mengakses data Tag, dengan menghubungkan Tag tersebut dengan variable I/O Server di LabVIEW.

Apabila saat program LabVIEW dijalankan, pembaca menemui error pada variable I/O Server (*Shared Variable Engine*) seperti ditunjukkan pada Gambar 4.42, pastikan OPC Quick Client dijalankan. Apabila OPC Quick Client sudah dijalankan, namun error masih muncul, maka cara mudah untuk mengatasi adalah dengan menghapus atau me-remove program DSC dengan bantuan NI Package Manager, dan kemudian instal ulang program DSC tersebut, maka error terkait *Shared Variable Engine* tersebut bisa dihilangkan.



Gambar 4.42 Apabila muncul error terkait SVE, pastikan OPC Quick Client dijalankan. Apabila OPC Quick Client sudah jalan, tapi error masih muncul, hapus dan instal ulang program DSC

46. Simpan program dan project LabVIEW. Sampai di sini, program LabVIEW Modbus RTU dengan NI OPC Server selesai.

4.3 OPC UA dengan NI OPC Servers

OPC UA (*Unified Architecture*) merupakan perbaikan dari OPC DA (*Data Access*). Sebelum membahas mengenai OPC UA, akan lebih baik mengetahui sejarah teknologi OPC. Teknologi OPC dimulai dengan OPC DA. OPC DA dibangun di atas teknologi DCOM Microsoft di tahun 1995. Sebelumnya, di tahun 90-an, komputer dengan sistem operasi Windows sangat populer digunakan sebagai platform otomasi industri berbiaya rendah. Saat itu Microsoft juga merilis COM, yang dilanjutkan dengan DCOM. DCOM adalah teknologi Microsoft eksklusif untuk komunikasi antar perangkat lunak pada jaringan komputer. DCOM mengizinkan pengembang dan vendor perangkat lunak untuk bekerja secara independen. Untuk bisa mengakses perangkat yang dibuatnya, vendor hanya perlu menyediakan driver yang mengimplementasikan interface OPC DA.

Semua SCADA, HMI dan perangkat lunak lainnya dapat terhubung dengan perangkat vendor tersebut melalui interface OPC DA. Untuk memastikan interface yang dibuat vendor memenuhi standar, OPC Foundation nirlaba dibentuk, untuk memvalidasi interface vendor. OPC DA dengan demikian menjadi satu-satunya standar yang diterima secara universal untuk pertukaran data antara sistem otomasi industri yang berbeda.

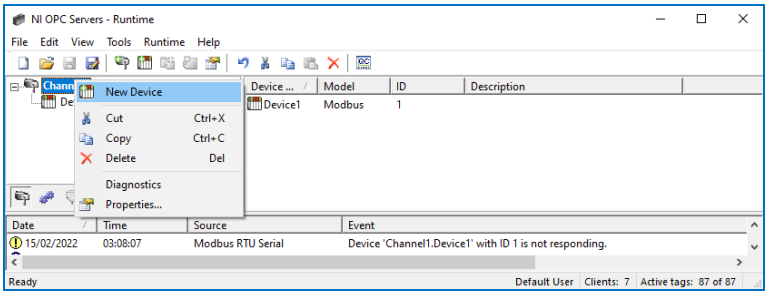
Sekalipun teknologi DCOM yang digunakan pada OPC DA ini sukses, namun sebenarnya banyak masalah yang mengikutinya. Beberapa di antaranya adalah sulitnya mengelola keamanan dengan interface DCOM, karena tidak mempertimbangkan adanya firewall dan keamanan internet (firewall harus dinonaktifkan agar komunikasi berjalan). Konfigurasi DCOM juga cukup sukar, seringkali menimbulkan masalah yang aneh. Teknologi DCOM ini juga tidak multiplatform, hanya bisa bekerja pada sistem operasi Windows. Oleh karena masalah itu, mulailah dikembangkan OPC UA.

OPC UA dibangun di atas protokol standar dan bukan pada teknologi Windows-centric seperti COM dan DCOM, sehingga OPC UA bisa bekerja pada berbagai sistem operasi (multiplatform). OPC UA mengimplementasikan layanan Web, XML, HTTPS, dan komunikasi internet lainnya yang memungkinkan akses ramah pada firewall. Agar sesuai dengan standar industri, OPC UA menggunakan standar keamanan internet seperti sertifikat X.509. Karena OPC UA adalah arsitektur berorientasi layanan terbuka, vendor mana pun bebas membangun model informasi data mereka sendiri yang sesuai dengan standar OPC. Spesifikasi OPC UA tersedia secara bebas dan dapat diterapkan di bawah lisensi GPL 2.0. Setiap platform SCADA modern memiliki dukungan luar biasa untuk OPC UA, seperti Industri 4.0, IIoT semuanya berkontribusi pada OPC UA.

Agar lebih jelas memahami teknologi OPC UA ini, berikut ini langkah-langkah penerapan OPC UA pada program LabVIEW menggunakan NI OPC Servers, yang akan menghubungkan jaringan 2 buah Modbus RTU di 2 buah komputer (OPC UA bisa menghubungkan lebih dari 2 komputer, namun agar tidak memperbanyak halaman buku, contoh di sini hanya menggunakan 2 komputer):

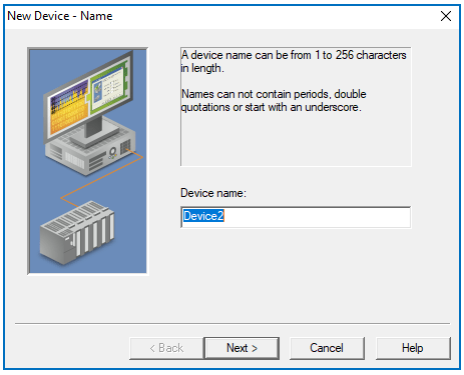
1. Siapkan 2 buah komputer.

2. Buat kedua komputer tersebut terhubung dalam satu jaringan (bisa menggunakan kabel LAN atau WiFi; cara mudahnya adalah menggunakan WiFi dari hotspot yang sama).
3. Komputer pertama akan disebut komputerA, dan komputer kedua akan disebut komputerB. Mulai dari komputerA, buka OPC Servers Administration, dan dilanjutkan buka OPC Servers Configuration.
4. Apabila tidak diubah, maka harusnya OPC Servers Configuration masih sama seperti Gambar 4.24, yaitu berisi Channel1 dengan sebuah Device (Device1), yang berisi 2 buah Tag, a dan b. Diinginkan untuk menambahkan sebuah Device lagi (Device2), dengan nomor ID = 2, dengan isi Tag yang sama seperti Device1. Untuk menambah Device2 ini, klik kanan Channel1, pilih New Device.



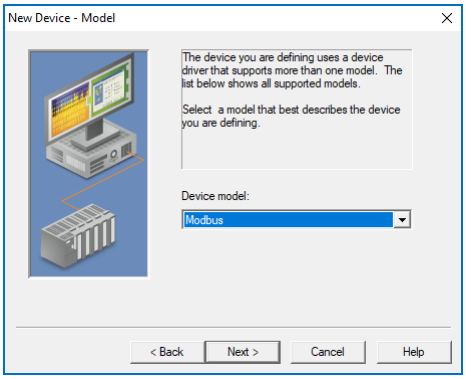
Gambar 4.43 Klik kanan Channel1, pilih New Device

5. Muncul jendela New Device-Name, Device2. Klik Next.



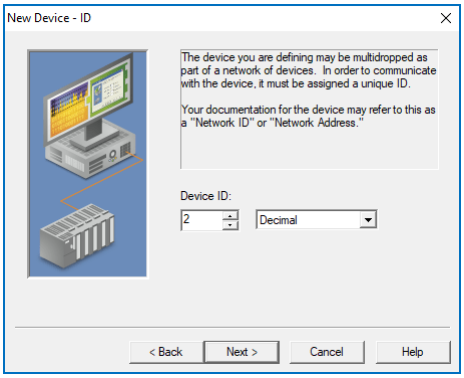
Gambar 4.44 Device Name = Device2

6. Pilih Device model: Modbus, klik Next.



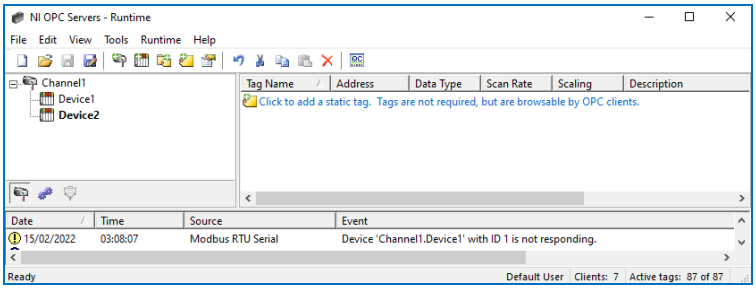
Gambar 4.45 Device model = Modbus

7. Ubah Device ID dari 1 menjadi 2, klik Next.



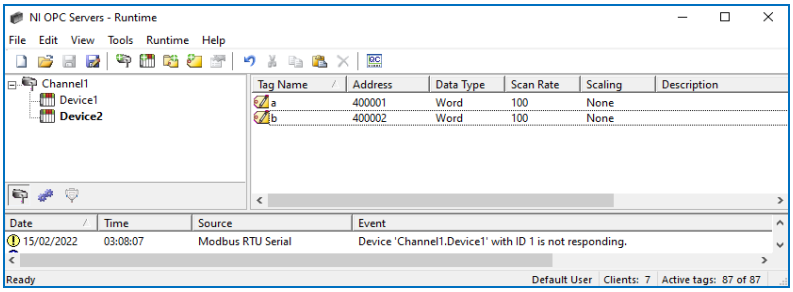
Gambar 4.45 Device ID = 2

8. Klik Next hingga selesai (Finish), maka akan muncul nama Device2.



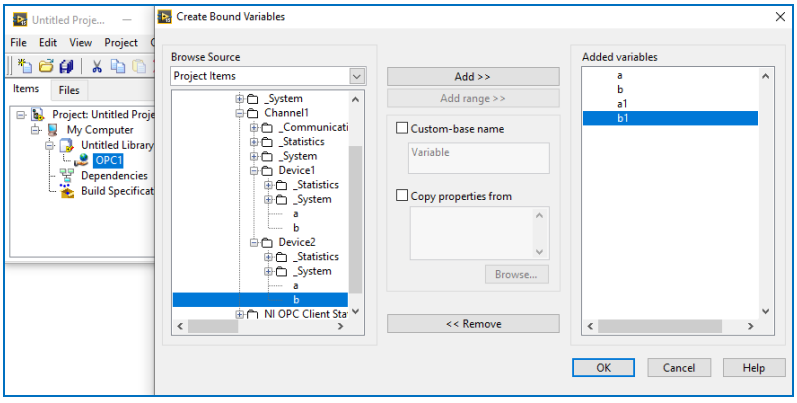
Gambar 4.46 Muncul nama Device2 di bawah Device1

9. Berikutnya, tambahkan Tag pada Device2. Agar cepat, copy Tag a dan b di Device1, pilih kedua Tag, dan tekan tombol Control+C, dan kemudian tempel di Device2 dengan menekan tombol Control+V.



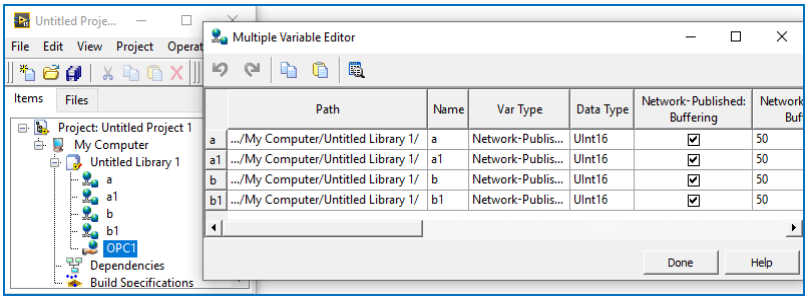
Gambar 4.47 Copy Tag a dan b di Device1, dan tempelkan di Device2

10. Berikutnya, buka LabVIEW, buat project baru dengan Create Project.
11. Pada kotak Project, klik kanan My Computer, pilih New, pilih I/O Server.
12. Pada kotak Create New I/O Server yang muncul, pilih OPC Client.
13. Pada kotak Configure OPC Client I/O Server yang muncul, di kolom Registered OPC Servers, pilih National Instruments.NIOPCServers.V5.
14. Klik OK, maka di kotak Project muncul OPC1, di folder Untitled Library1.
15. Berikutnya, klik kanan OPC1, dan pilih Create Bound Variables.
16. Pada kotak Create Bound Variables, di kolom Browse Source, buka Tag a dan b di Device1 dan di Device2, dan klik tombol Add untuk memasukkan keempat Tag tersebut pada kolom Added variables.



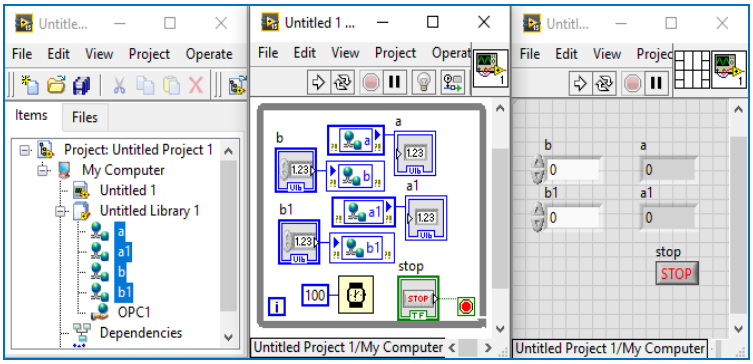
Gambar 4.48 Membuat Tag a dan b di Device1 dan Device2 menjadi variable I/O Server

17. Perhatikan, bahwa Tag a dan b dari Device2, ketika ditambahkan ke kolom Added variables, otomatis diubah menjadi a1 dan b1. Klik OK, maka akan muncul variable a, a1, b, dan b1 di dalam folder Untitled Library1.



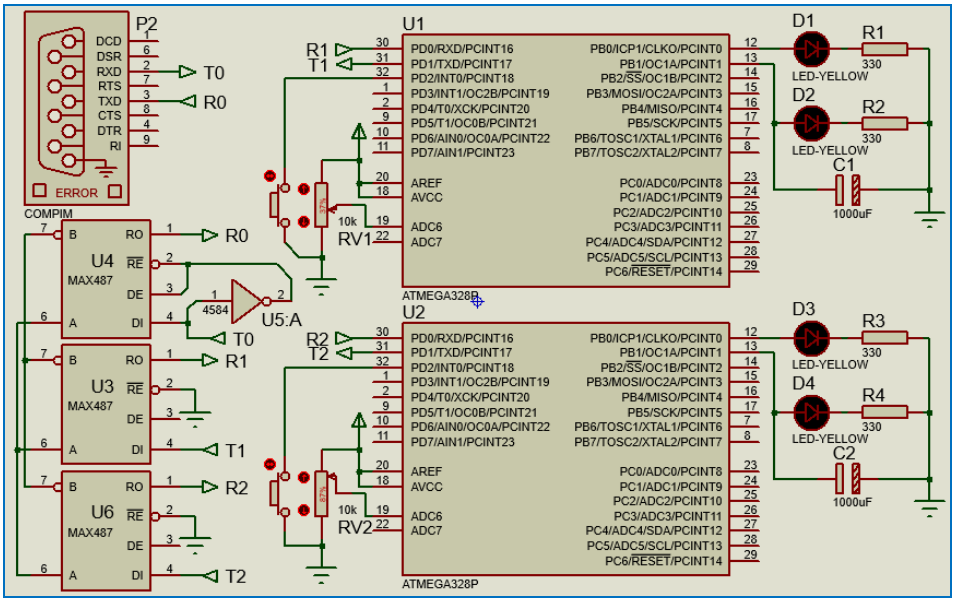
Gambar 4.49 Empat buah variable I/O Server yang dihasilkan dari Create Bound Variables

18. Klik Done. Berikutnya, klik kanan My Computer, pilih New, pilih VI.
19. Tempatkan Block Diagram dan Front Panel VI yang baru berdampingan. Kemudian drag keempat variable (a, a1, b, b1) ke dalam Block Diagram.
20. Tambahkan While Loop, tombol Stop, dan icon Wait(ms) dengan nilai 100.
21. Ubah Access Mode variable b dan b1, dari Read menjadi Write.
22. Tambahkan Indicator pada variable a dan a1, dan Control pada variable b dan b1 (klik kanan kaki input/output, pilih Create, pilih Control/Indicator).



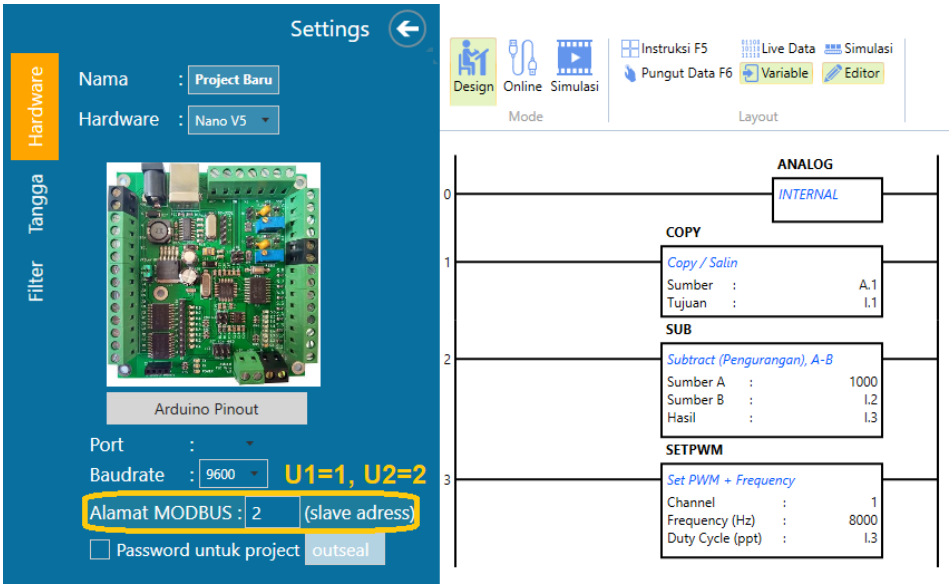
Gambar 4.50 Program LabVIEW sebagai Master dari 2 buah Modbus RTU (Device1 & Device2)

23. Berikutnya, karena ada 2 buah perangkat Modbus Slave RTU, simulator ModRSsim2 tidak bisa digunakan. Sebagai gantinya, digunakan software Proteus, dengan rangkaian seperti Gambar 3.25.



Gambar 4.51 Rangkaian 2 buah Arduino (Outseal) di Proteus sebagai Modbus RTU Slave

24. Gunakan program Outseal yang sama berikut ini untuk Arduino U1 dan U2 di atas. Buat alamat modbus U1 = 1 dan alamat modbus U2 = 2.



Gambar 4.52 Program Outseal untuk U1 dan U2, U1 diberi alamat 1, dan U2 diberi alamat 2

Keterangan program:

a. Rangkaian 2 buah Arduino yang terhubung dengan jaringan RS-485 di Proteus pada Gambar 4.51 di atas, masing-masing memiliki input output Tombol, Potensio dan 2 buah LED. Namun demikian, hanya Potensio dan sebuah LED yang digunakan untuk program Modbus kali ini, mengikuti seting di NI OPC Servers, yang diseting menggunakan 2 buah Tag di alamat Holding Register, yaitu di alamat 400001 dan 400002, baik di Device1 maupun Device2.

b. Lebih jauh, kedua memori tersebut dibuat variable di program LabVIEW, di mana untuk alamat 400001, diseting dapat di-Read, sehingga terhubung dengan objek Numeric Indicator, dan untuk alamat 400002, diseting dapat di-Write, sehingga terhubung dengan objek Numeric Control.

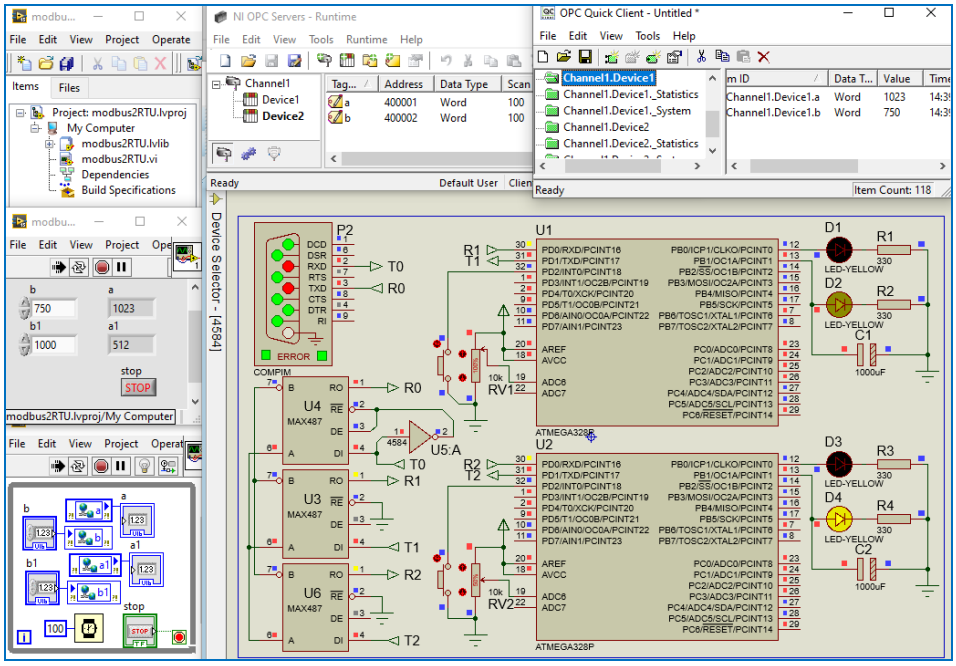
c. Menyesuaikan dengan program LabVIEW tersebut, di program Outseal, untuk alamat 400001, adalah sama dengan data Integer1 (I.1), dan alamat 400002 adalah sama dengan data Integer2 (I.2). Karena variable 400001 di LabVIEW harus dapat di-Read, maka di program Outseal, I.1 harus dibuat mengeluarkan data, di mana data ini dikeluarkan oleh Potensio, melalui variable A.1. Untuk itu, agar A1 bisa menjadi I.1, digunakan instruksi Copy.

d. Berikutnya, karena variable 400002 di LabVIEW dapat di-Write, maka di program Outseal, I.2 harus dibuat membaca data, dan menampilkan datanya, yang ditampilkan dalam bentuk nyala LED di kaki D9 Arduino (atau kaki R7 Outseal), di mana kaki D9 ini adalah Channel = 1 di SetPWM.

e. Karena nilai Duty Cycle ternyata terbalik, yaitu nilai 0 membuat LED nyala terang, dan nilai 1000 malah membuat LED padam, maka perlu pembalikan, menggunakan fungsi Substract, sehingga bila nilai I.2 = 1000, maka Duty Cycle (I.3) menjadi 0, dan sebaliknya bila I.2 = 0, nilai Duty Cycle (I.3) menjadi 1000.

f. Dengan program yang sama, atur alamat Modbus Slave = 1 untuk U1, dan alamat Modbus Slave = 2 untuk U2, yang diatur lewat Settings, seperti ditunjukkan pada Gambar 4.52 di atas. Lakukan kompilasi untuk mendapatkan file Hex, dengan menekan tombol Test. Agar tidak tertumpuk filenya, ambil file Hex untuk U1, tempatkan di dekat file Proteus, dan ubah namanya, misalnya modbusRTU1.hex. Ulangi untuk U2, ambil file Hex, tempatkan di dekat file Proteus, dan ubah namanya menjadi modbusRTU2.hex.

25. Berikutnya, jalankan program LabVIEW dan Proteus. Lakukan pengubahan nilai b dan b1 dari 0 - 1000, dan perhatikan nyala LED PWM di kaki D9, U1 dan U2. Seharusnya ketika 0, LED padam, dan nyala terang ketika 1000.
26. Di Proteus, naik turunkan Potensio di kaki A6 di U1 dan U2, dan perhatikan nilai Numeric Indicator a dan a1. Seharusnya ketika posisi minimum, nilainya = 0, dan ketika posisi maksimum, nilainya = 1023.



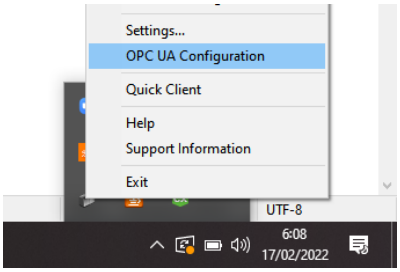
Gambar 4.53 Program LabVIEW membaca nilai Potensio (Tag a = 400001) dan mengatur LED PWM (Tag b = 400002) di U1 dan U2 melalui protokol Modbus RTU di NI OPC Servers

Catatan: perhatikan Gambar 4.53 di atas, ada 3 jenis koneksi yang ada di gambar tersebut. Jenis pertama adalah koneksi fisik berupa rangkaian RS-485 yang menghubungkan 2 buah rangkaian Arduino di Proteus sebagai Slave dengan komputer sebagai Master. Jenis kedua adalah koneksi virtual berupa protokol Modbus RTU, yang dibuat oleh OPC Server sebagai Modbus Master dan program Outseal sebagai Modbus Slave. Jenis ketiga adalah koneksi interface, antara pengguna dengan tampilan OPC Quick Client dan juga Front Panel LabVIEW.

OPC Quick Client adalah software bantu OPC Kepware, yang menyediakan semua fitur yang diperlukan untuk aplikasi OPC Client, meliputi akses semua data di OPC Server, baca dan tulis data Tag, dan menguji dan mendiagnosis kinerja server.

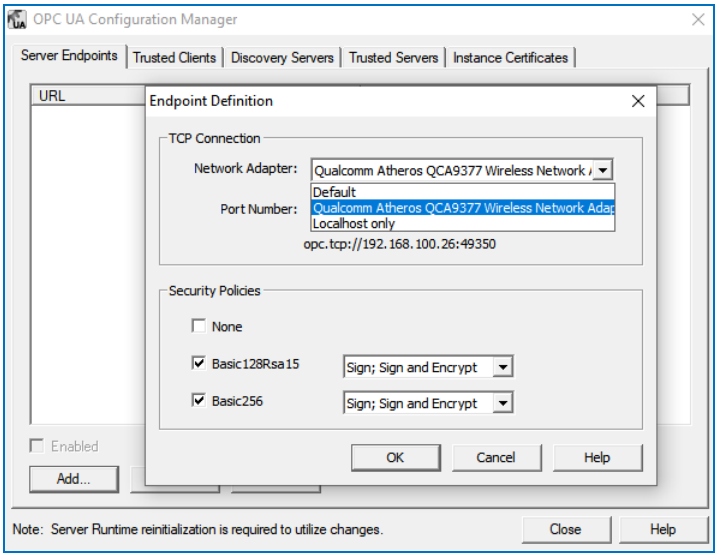
Ada 150 lebih protokol atau driver yang disediakan oleh NI OPC Server (juga OPC Kepware). Ketika 1 atau lebih protokol dipasang di NI OPC Servers, Interface OPC Quick Client akan menampilkan status dan kondisi Device serta Tag pada protokol tersebut, dan juga bisa mengubah datanya. Status dan kondisi komunikasi ditampilkan melalui keterangan di kolom sisi kanan, dan juga Log data di kolom bawah. OPC Quick Client bekerja seperti OPC DA, yang bisa menjadi interface dari sebuah OPC Server. Hanya saja, kelemahan Interface OPC DA adalah hanya bisa berjalan di sistem operasi Windows. Untuk mengatasi kelemahan OPC DA ini, dikembangkanlah OPC UA, yang bisa menjadi jembatan atau interface pada sistem operasi yang berbeda (multiplatform), serta memiliki tingkat keamanan yang lebih tinggi dan "ramah" terhadap Firewall. Ramah dalam arti, Firewall tidak perlu dimatikan. Hanya perlu membuka 1 port khusus untuk OPC UA, sehingga data di OPC Server tetap bisa diakses oleh OPC UA tanpa Firewall harus dimatikan.

27. Setelah program Modbus RTU, baik di LabVIEW maupun di Proteus bisa berjalan dengan baik, diinginkan rangkaian Proteus di komputer ini (komputerA) dapat juga dibaca dan dikontrol dari program LabVIEW di komputer lain (komputerB). Jadi di sini, komputerA dibuat sebagai OPC UA Server, sedangkan komputerB dibuat sebagai OPC UA Client.
28. Untuk membuat komputerA menjadi OPC UA Server, klik kanan icon OPC Server Administration, pilih OPC UA Configuration.

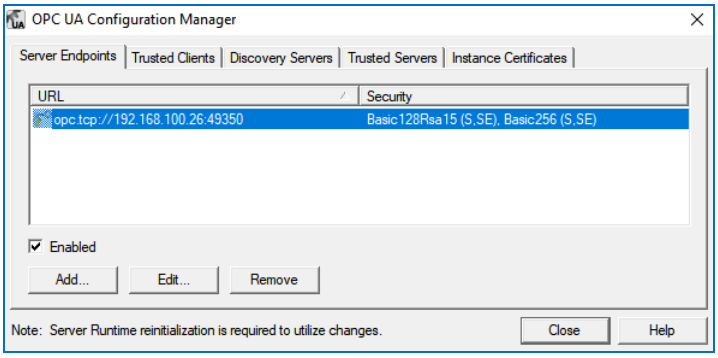


Gambar 4.54 Klik kanan icon OPC Server Administration, pilih OPC UA Configuration

29. Pada jendela OPC UA Configuration, pada Tab Server Endpoints, klik tombol Add. Maka akan muncul jendela Endpoint Definition.
30. Pada Network Adapter, pilih Adapter yang digunakan, dalam contoh di sini menggunakan Qualcomm (catatan: sebaiknya jangan memilih Default atau Localhost only, tetapi pilih Adapter supaya bisa memunculkan URL).
31. Klik OK, maka akan muncul URL di Server Endpoints (Gambar 4.56).



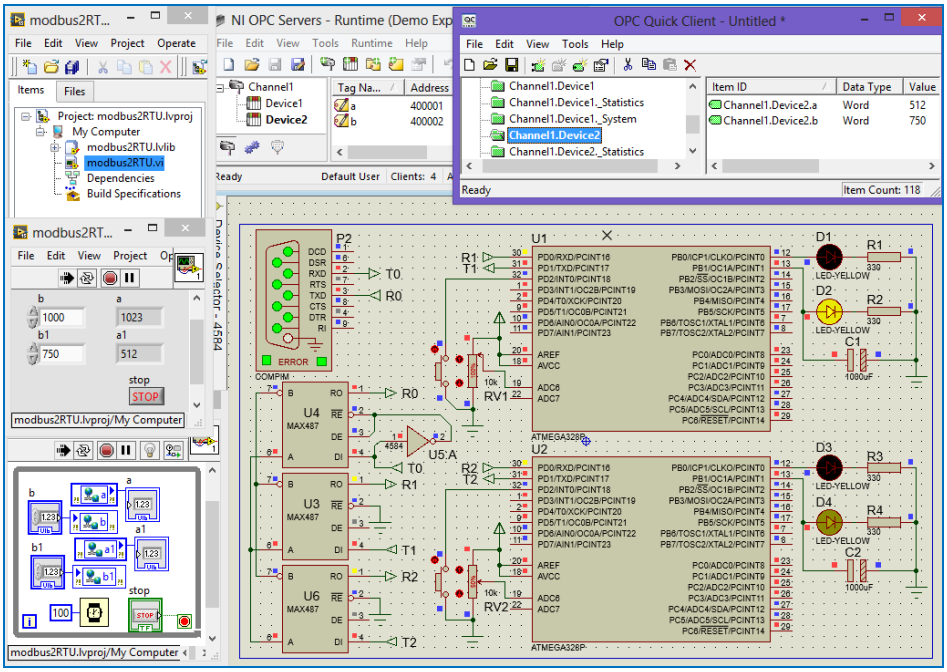
Gambar 4.55 Klik Add, di Endpoint Definition, pilih Network Adapter, klik OK



Gambar 4.56 Di Tab Server Endpoints muncul URL Server Endpoints

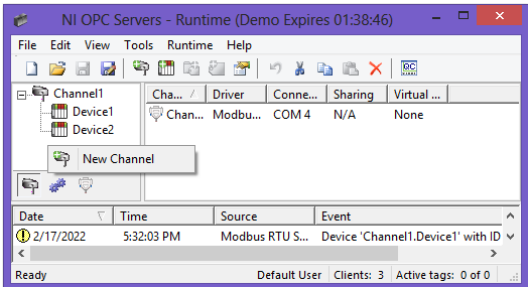
32. URL di atas merupakan alamat OPC UA Server (komputerA). Klik Close.
33. Berikutnya, seting OPC UA Client di komputerB.

34. Di komputerB, ulangi langkah no. 4 sampai 26 di atas, sehingga di komputerB dapat menampilkan hasil yang sama seperti Gambar 4.53, yaitu komunikasi antara program Modbus RTU Master di LabVIEW dengan rangkaian 2 buah Arduino (dengan program Outseal) di Proteus sebagai Modbus RTU Slave, dengan penghubung NI OPC Servers, seperti berikut:



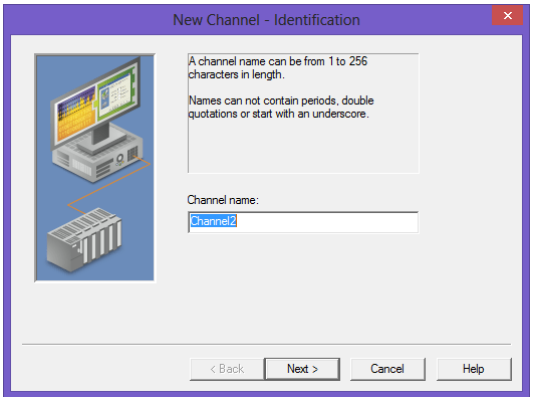
Gambar 4.57 Buat komputerB juga dapat menampilkan hasil yang sama seperti komputerA

35. Berikutnya, di NI OPC Servers Configuration, tambahkan sebuah channel untuk OPC UA Client. Klik kanan kolom di kiri, pilih New Channel.



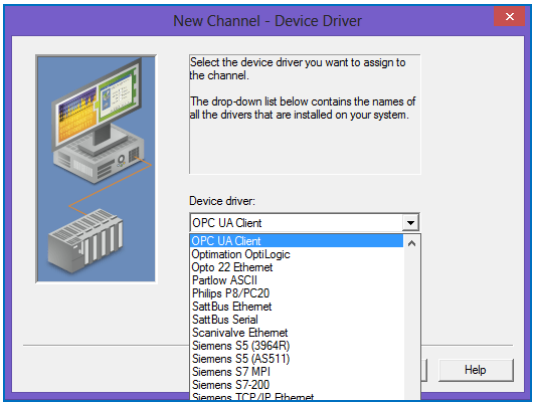
Gambar 4 .58 Buat channel baru dengan meng-klik kanan kolom di kiri, pilih New Channel

36. Muncul jendela New Channel-Identification, name = Channel2. Klik Next.



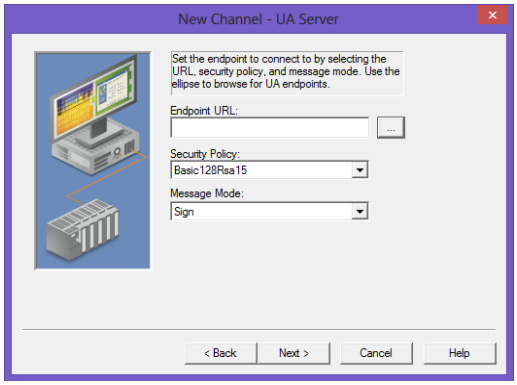
Gambar 4.59 Jendela New Channel, Channel name = Channel2, klik Next

37. Pada jendela New Channel-Device Driver, pilih OPC UA Client. Klik Next.

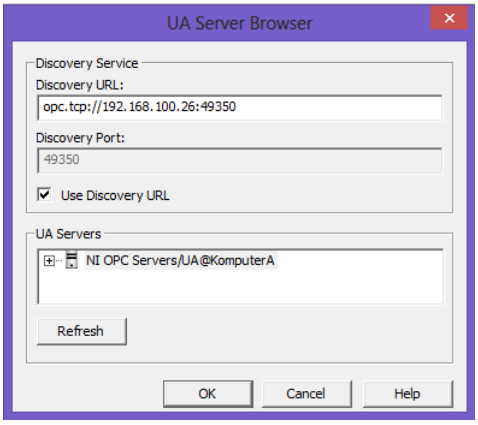


Gambar 4.60 Pilih Device driver = OPC UA Client, klik Next

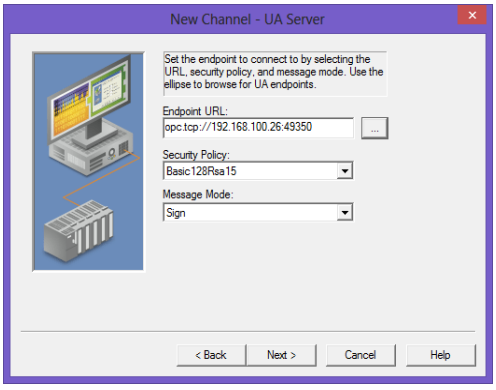
38. Klik Next hingga sampai di jendela New Channel-UA Server. Tekan tombol di samping kanan kolom Endpoint URL, maka akan muncul jendela UA Server Browser. Isi kolom Discovery URL dengan URL komputerA (lihat langkah no. 31 - 32, dan Gambar 4.56. Dalam contoh di sini, URL komputerA adalah opc.tcp://192.168.100.26: 49350). Ketik URL tersebut di kolom Discovery URL, kemudian beri centang pada Use Discovery URL. Klik OK. Apabila kolom Endpoint URL belum terisi, ulangi langkah ini.



Gambar 4.61 Di jendela New Channel UA Server, klik tombol di kanan kolom Endpoint URL

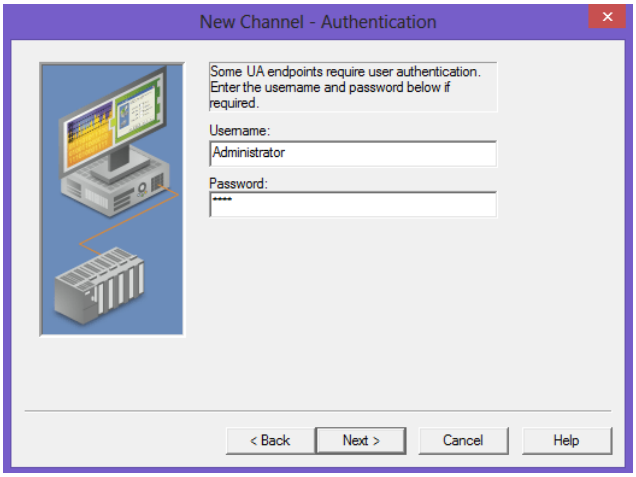


Gambar 4.62 Isi Discovery URL dengan URL di OPC UA Configuration komputerA, beri centang di Use Discovery URL, klik OK. Buka kembali UA Server Browser, harusnya UA Servers muncul



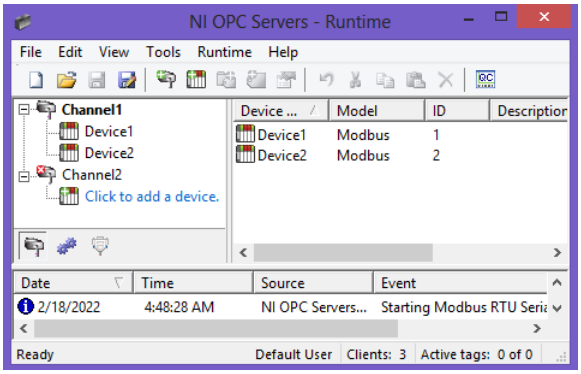
Gambar 4.63 Endpoint URL sudah terisi dengan URL komputerA, klik Next

39. Klik Next hingga sampai di jendela Authentication. Karena OPC UA memiliki tingkat keamanan yang tinggi, selain kode keamanan, pengguna harus memiliki otentikasi tingkat Administrator. Ulangi kembali langkah no. 10 – 11 di Sub Bab 4.2, tentang pembuatan password untuk Administrator. Setelah password berhasil dibuat, isi kolom Username dengan Administrator, dan kolom Password dengan password tersebut.



Gambar 4.64 Isi Username dengan Administrator, dan Password sesuai seting Administrator

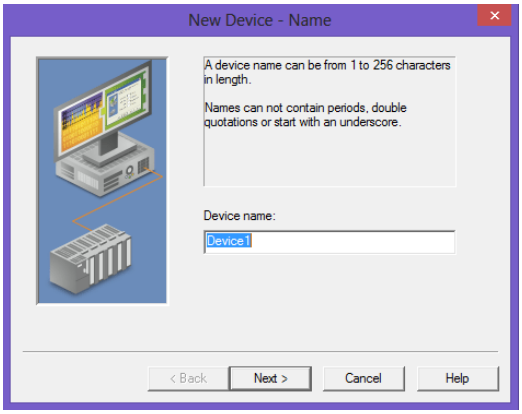
40. Klik Next hingga selesai (Finish), maka pada NI OPC Servers Configuration, akan muncul Channel2, yang merupakan OPC UA Client.



Gambar 4.65 Muncul Channel2 (OPC UA Client) di NI OPC Servers Configuration komputerB

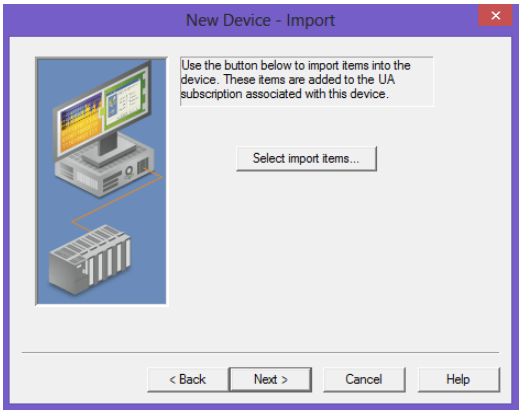
41. Klik pada *Click to add a device* untuk menambahkan Device.

42. Muncul jendela New Device-Name. Klik Next.



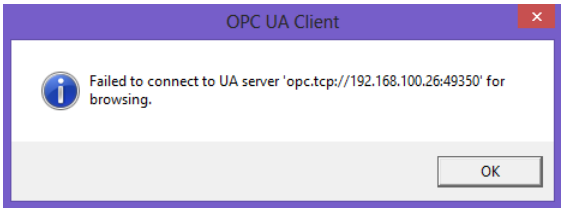
Gambar 4.66 Jendela New Device muncul, Device name = Device1

43. Klik Next hingga di New Device-Import. Klik tombol Select Import Items.



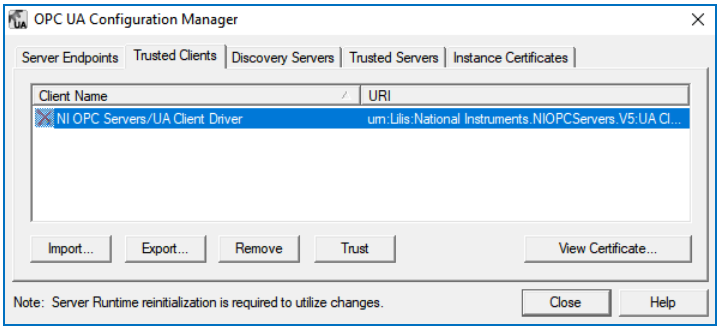
Gambar 4.67 Di jendela New Device – Import, klik tombol Select Import Items

44. Karena belum terhubung, muncul pesan Failed to connect to UA server.

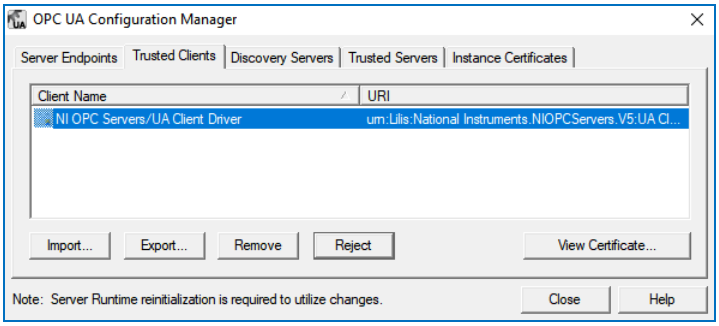


Gambar 4.68 Ketika tombol Select Import Items ditekan, muncul pesan Failed to connect

45. Agar bisa terhubung, di komputerA, buka OPC UA Configuration. Di Tab Trusted Clients, terlihat tanda silang pada Client Name. Tekan tombol Trust untuk menghubungkan OPC UA Server ini (komputerA) dengan OPC UA Client (komputerB), harusnya tanda silang akan hilang. Klik Close.

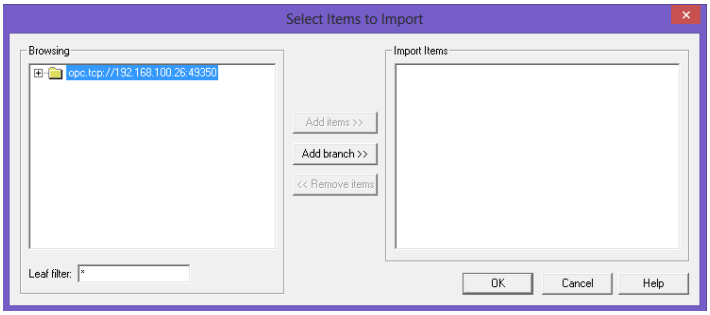


Gambar 4.69 Di Tab Trusted Clients OPC UA Configuration komputerA, nama Client disilang



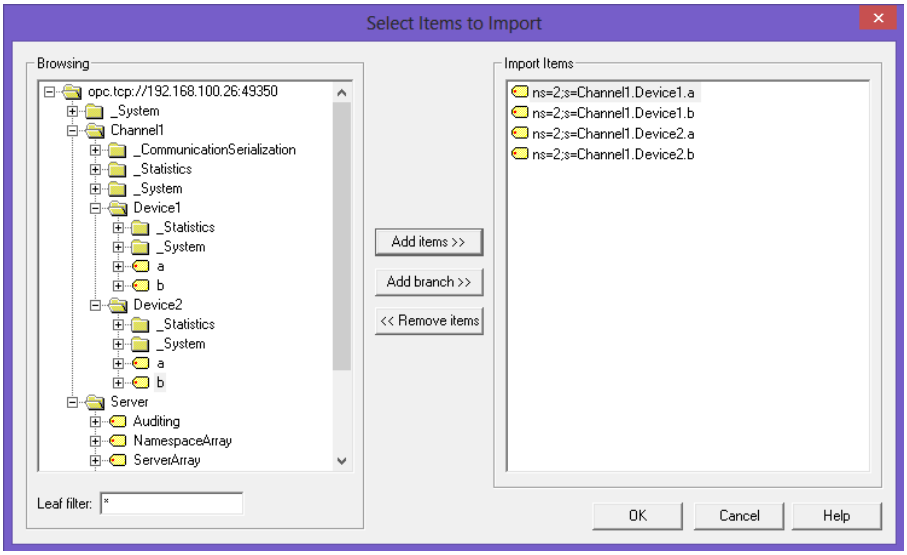
Gambar 4.70 Setelah tombol Trust ditekan, tanda silang di nama Client (komputerB) hilang

46. Di komputerB, ulangi tekan tombol Select Import Items, maka muncul:



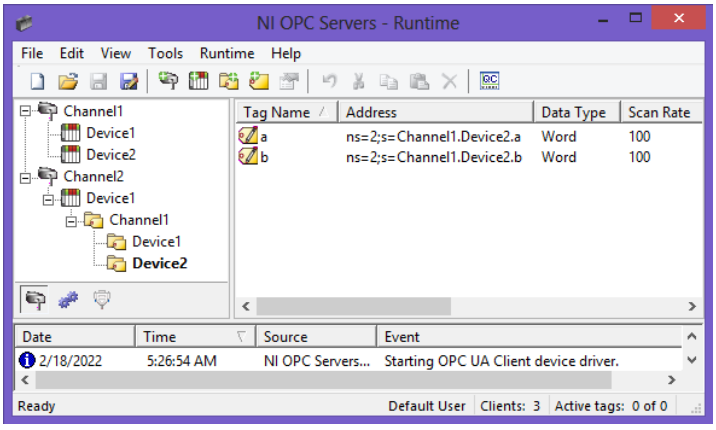
Gambar 4.71 Jendela Select Items to Import bisa muncul setelah OPC UA Server terhubung

47. Import Tag a dan b di Device1 dan Device2 dari URL OPC UA Server.



Gambar 4.72 Add items Tag a dan b di Device1 dan 2 dari URL OPC UA Server (komputerA)

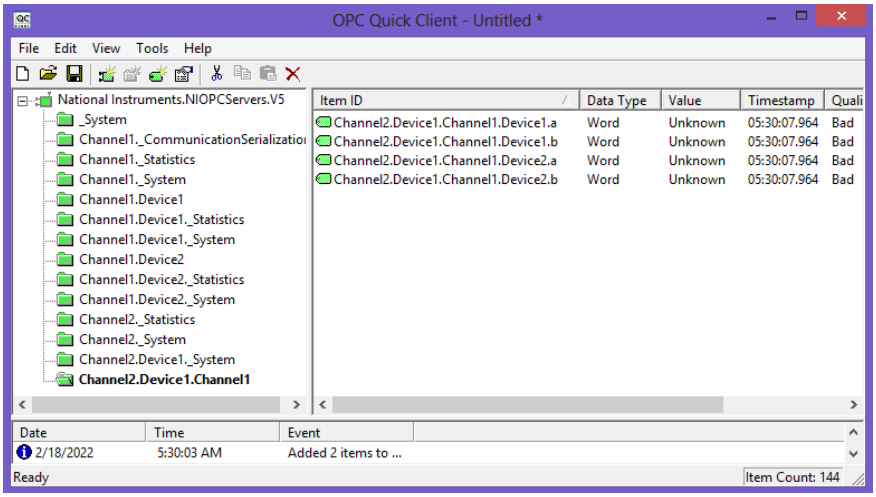
48. Klik OK, maka di Channel2, di Device1, muncul Channel, Device dan Tag dari OPC Server komputerA. Perhatikan bahwa dengan OPC UA ini, semua Channel dan Device beserta protokol dan drivernya dari OPC Server di komputerA, dapat dimonitor dan dikontrol dari OPC Server di komputerB.



Gambar 4.73 Ketika Import berhasil, maka muncul Channel, Device dan Tag dari komputerA

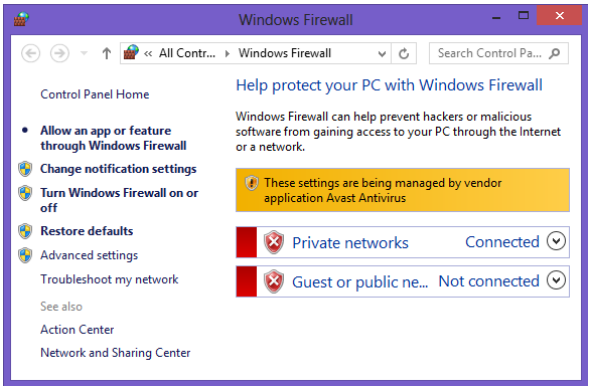
49. Klik tombol OPC Quick Client untuk menjalankan OPC UA Client.

50. Apabila pembaca mendapati kualitas komunikasi OPC Quick Client bernilai Bad seperti gambar berikut ini, bisa disebabkan karena 2 hal: yang pertama, karena data tidak bisa terkirim karena terhalang oleh Windows Firewall, dan yang kedua, karena rangkaian Arduino di Proteus, yang menjadi perangkat Modbus RTU Slave, belum dijalankan.



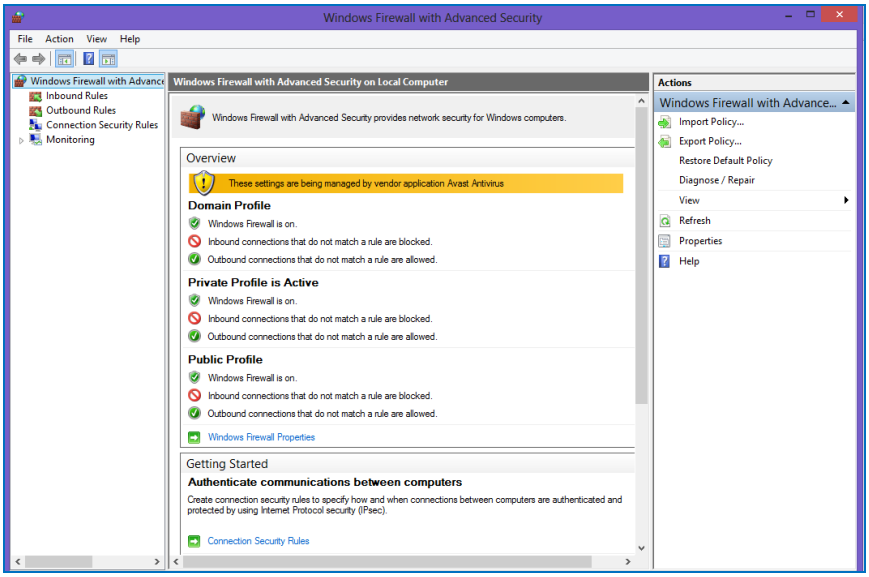
Gambar 4.74 Quality komunikasi di OPC Client bernilai "Bad" bisa karena 2 hal, pertama karena Windows Firewall, kedua karena perangkat (Modbus Slave) di Server tidak aktif

51. Untuk mengatasi Windows Firewall ini, di komputerB, buka Windows Firewall, dengan meng-klik kanan pada Start Window, pilih Search, ketikkan Firewall, pilih nama Windows (Defender) Firewall.



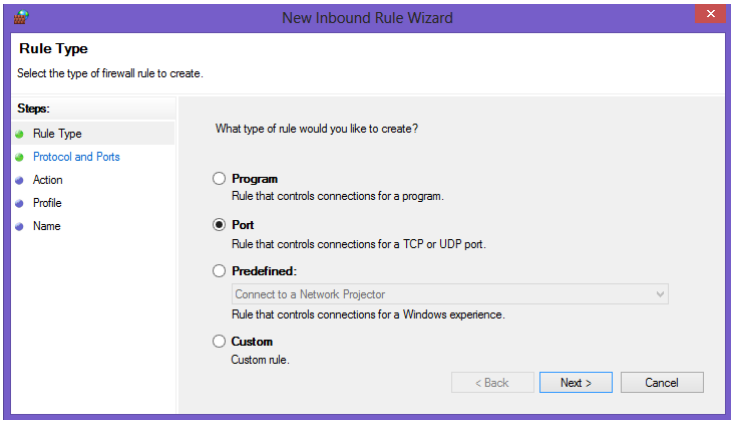
Gambar 4.75 Buka Windows (Defender) Firewall, klik Advanced settings

52. Di Windows Firewall with Advanced Security, klik Inbound Rules.



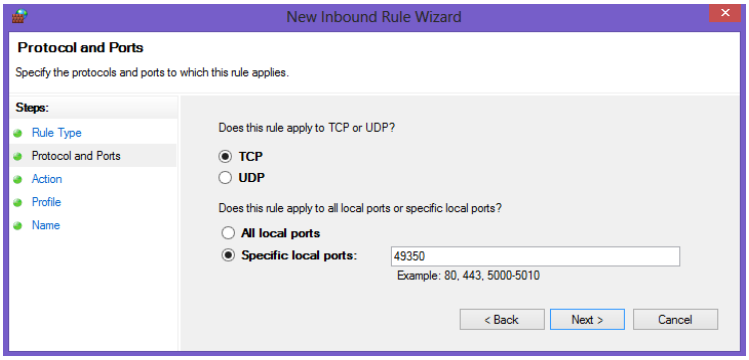
Gambar 4.76 Di Windows Firewall with Advanced Security, klik Inbound Rules

53. Di Inbound Rule (kolom yang kanan), klik New Rule, maka muncul jendela New Inbound Rule Wizard. Di Steps: Rule Type, pilih Port, klik Next.



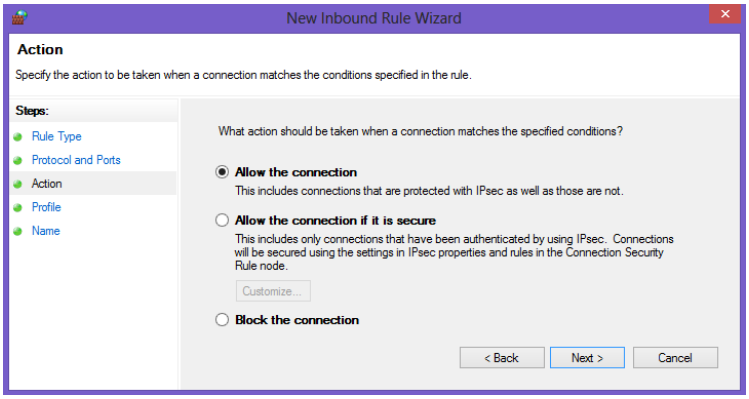
Gambar 4.77 Di jendela New Inbound Rule Wizard, pilih Port, klik Next

54. Di Steps: Protocol and Ports, pilih TCP, isi Specific local ports: 49350 (49350 adalah Port untuk OPC UA National Instruments), klik Next.



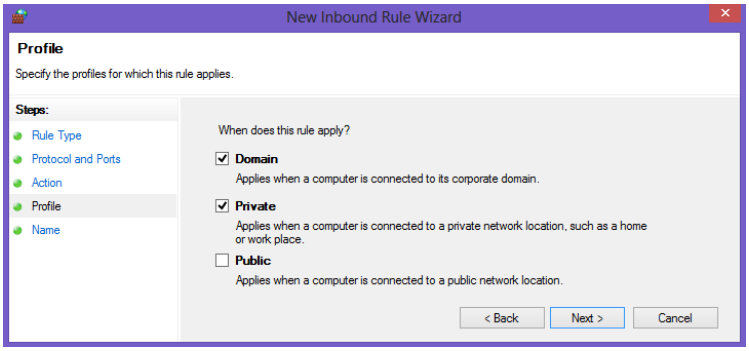
Gambar 4.77 Di Steps: Protocol and Ports, pilih TCP, isi Specific local ports: 49350

55. Di Steps: Action, pilih Allows the connection



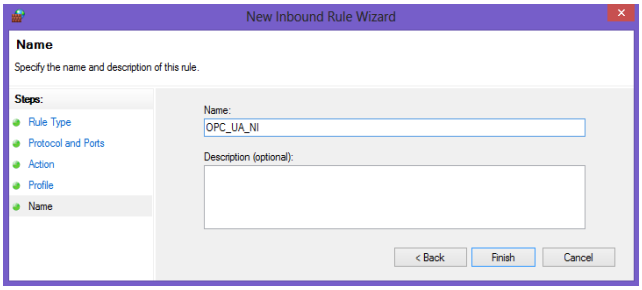
Gambar 4.78 Di Steps: Action, pilih Allows the connection

56. Di Steps: Profile, hilangkan tanda centang pada Public.



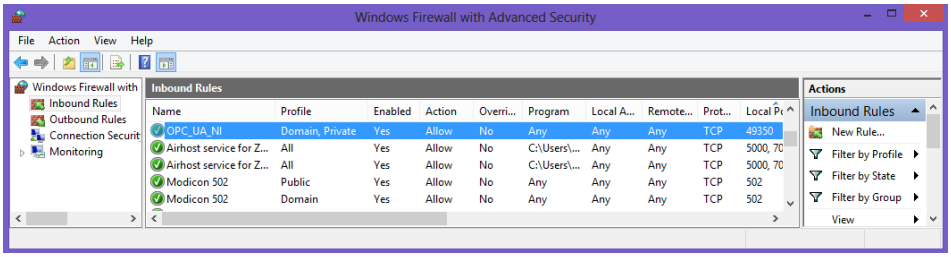
Gambar 4.78 Di Steps: Profile, hilangkan tanda centang pada Public

57. Di Steps: Name, beri nama sembarang, klik Finish.



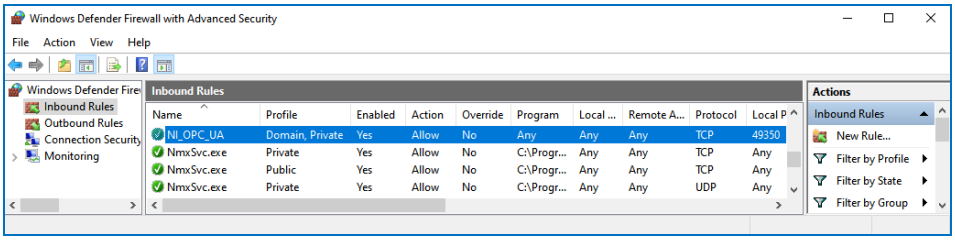
Gambar 4.79 Di Steps: Name, beri nama bebas, klik tombol Finish

58. Di jendela Windows Firewall, muncul nama Inbound Rules yang baru, yang mengijinkan koneksi dilakukan oleh setiap perangkat yang berada di Domain yang sama atau jaringan Private, yang melalui Port 49350.



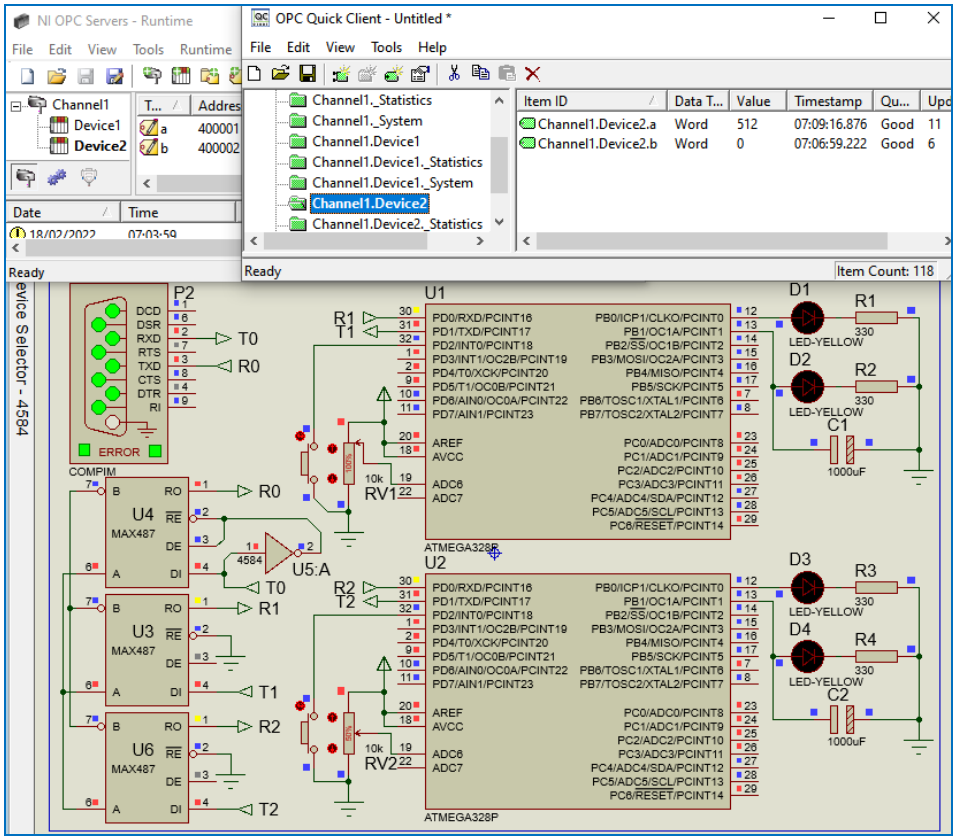
Gambar 4.79 Tampak nama Inbound Rule yang baru, yang mengijinkan koneksi setiap perangkat dalam Domain yang sama atau jaringan Private yang melalui Port 49350

59. Dengan membuat Port khusus untuk komunikasi OPC UA ini, Windows Firewall tidak perlu dimatikan, sehingga keamanan sistem tetap terjaga. Agar komputerA juga bisa dihubungkan dengan OPC UA ini tanpa perlu mematikan Windows Firewall, ulangi langkah no. 51 hingga 58 di atas.



Gambar 4.80 Di komputerA, juga telah dibuat Inbound Rule yang baru seperti komputerB

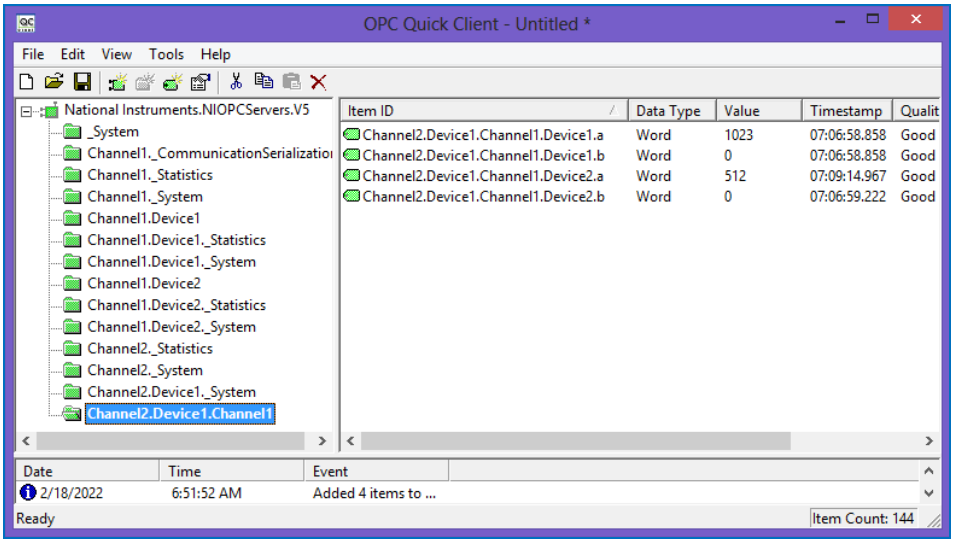
60. Berikutnya, setelah Windows Firewall diatasi dengan mengkhususkan sebuah Port untuk komunikasi OPC UA, hal kedua yang dilakukan adalah menjalankan Device yang terhubung ke OPC UA Server (komputerA). Untuk itu, di komputerA, jalankan rangkaian 2 buah Arduino (dengan program Modbus RTU Outseal) di Proteus, dan buka OPC Quick Client, perhatikan bahwa nilai Quality OPC Quick Client dari Bad menjadi Good.



Gambar 4.81 Ketika rangkaian 2 buah Arduino di Proteus, yang menjadi Device1 dan 2 di OPC UA Server (komputerA) dijalankan, maka nilai Quality OPC Quick Client telah menjadi Good

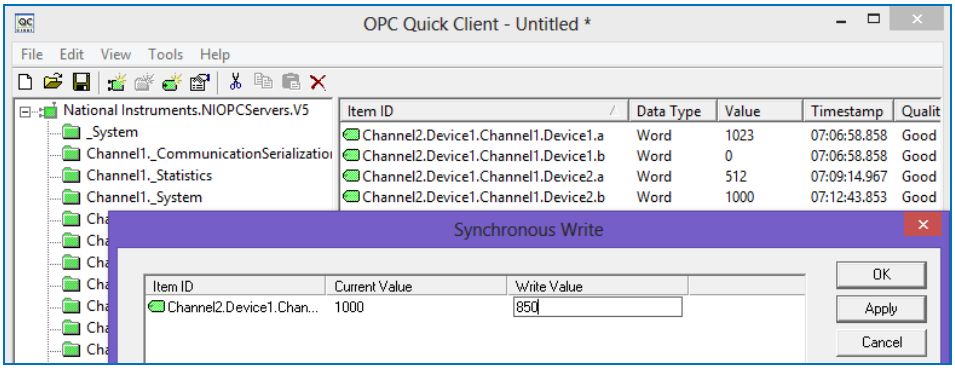
61. Ketika perangkat OPC UA Server (komputerA) sudah dijalankan, dan kualitas komunikasi antara perangkat dengan OPC UA Server bernilai Good (terlihat dari OPC Quick Client), maka seharusnya, OPC Quick Client di komputerB juga menampilkan kualitas komunikasi yang bernilai Good.

62. Buka OPC Quick Client di komputerB, dan perhatikan nilai dari Channel2 (OPC UA Client), yang menampilkan nilai Potensio (Tag a) di Device1 dan Device2, dan juga nilai kontrol untuk LED PWM (Tag b).



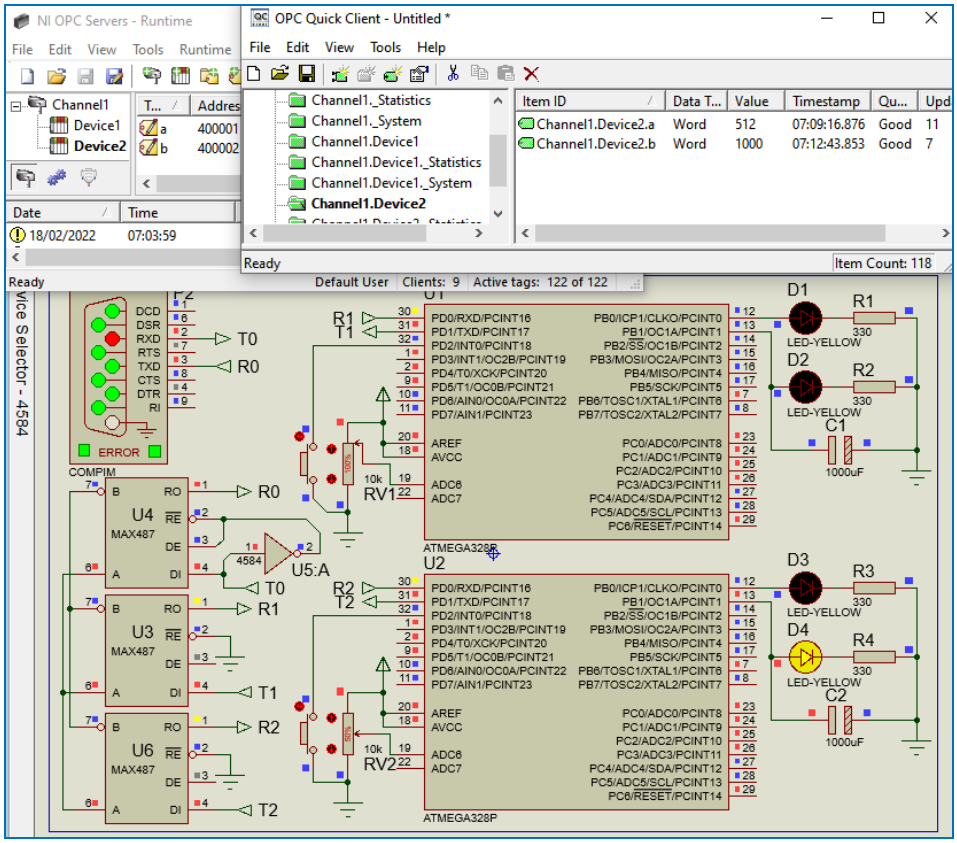
Gambar 4.82 Tampak Quality komunikasi bernilai Good, sehingga Nilai Tag a dan b di kedua Device di komputerA dapat dimonitor dari komputerB melalui OPC UA

63. Tag b di komputerA, baik di Device1 maupun 2, dapat diatur nilainya dari OPC Quick Client di komputerB, yang menjadi interface OPC UA Client. Contoh pengaturan nilai PWM dari 1000 menjadi 850, dilakukan dengan cara: klik kanan nilai 1000, pilih Synchronous Write, isi Write Value = 850.



Gambar 4.83 Pengaturan nilai LED PWM (Tag b) di Device2 dengan Synchronous Write

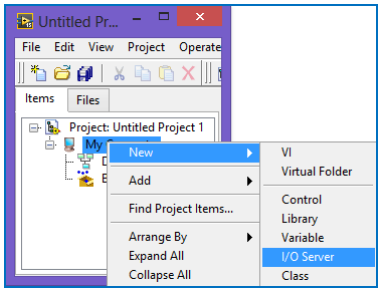
64. Ketika Tag b di Channel2 (OPC UA Client) di OPC Quick Client di komputerB diubah angkanya menjadi 1000, maka Tag b di Device2 di komputerA menjadi bernilai 1000 juga. Nilai Tag b Device 2 ini terhubung dengan LED PWM (D4) di Arduino U2, sehingga LED menyala maksimum.



Gambar 4.84 Pengaturan nilai LED PWM (Tag b) membuat LED PWM di U2 menyala terang

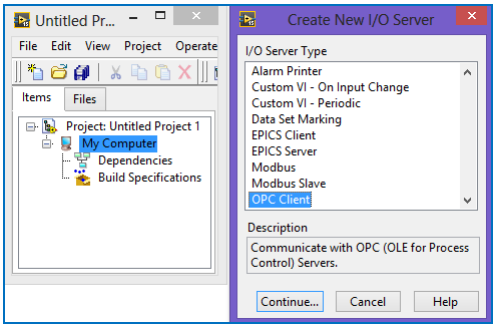
65. Setelah komunikasi antara OPC UA Client di komputerB dengan OPC UA Server di komputerA berjalan lancar (baik untuk Read maupun Write), melalui OPC Quick Client masing-masing, maka langkah berikutnya adalah men-*deploy* OPC UA Client di program LabVIEW, sehingga perangkat yang terhubung dengan OPC UA Server di komputerA, dapat dimonitor dan dikontrol dengan program LabVIEW di komputerB. Untuk itu buka LabVIEW di komputerB, Create Project, pilih Blank Project.

66. Di kotak Project, klik kanan My Computer, pilih New, pilih I/O Server



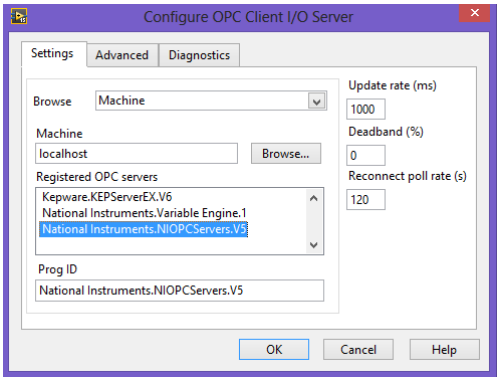
Gambar 4.85 Klik kanan My Computer, pilih New, pilih I/O Server

67. Pada kotak Create New I/O Server, pilih OPC Client. Klik Continue.



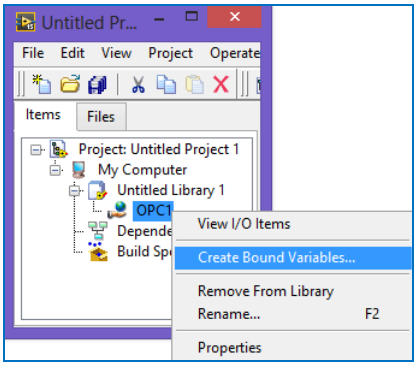
Gambar 4.86 Pilih Create New I/O Server = OPC Client

68. Pada Registered OPC servers di kotak Configure OPC Client I/O Server, pilih National Instruments.NIOPC Servers.V5. Klik OK.



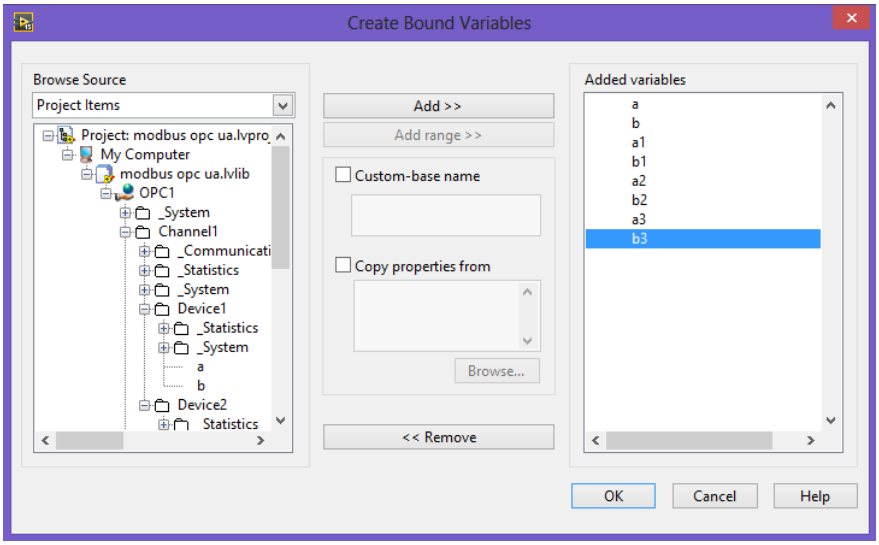
Gambar 4.87 Pilih National Instruments.NIOPCServers.V5

69. Di kotak Project, muncul OPC1, di dalam folder Untitled Library1. Klik kanan OPC1, pilih Create Bound Variables.



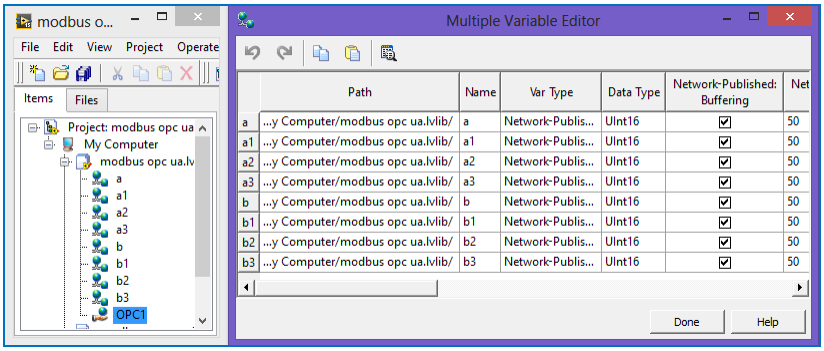
Gambar 4.88 Klik kanan OPC1, pilih Create Bound Variables

70. Muncul jendela Create Bound Variables. Tambahkan Tag a dan b, mulai dari Device1 dan Device2 di Channel1, dilanjutkan dengan Tag a dan b dari Device1 dan Device2 di Channel2 secara berurutan, sehingga muncul 8 buah variable di kolom Added variables. Klik OK.



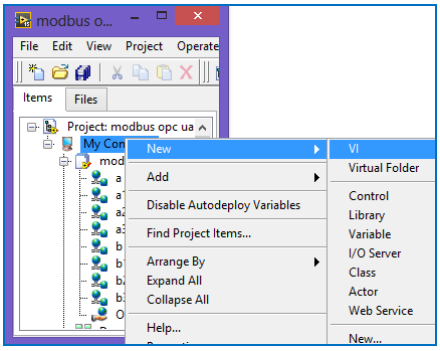
Gambar 4.89 Tambahkan Tag a dan b dari Device1 dan Device2 di setiap Channel

71. Muncul jendela Multiple Variable Editor. Klik Done.



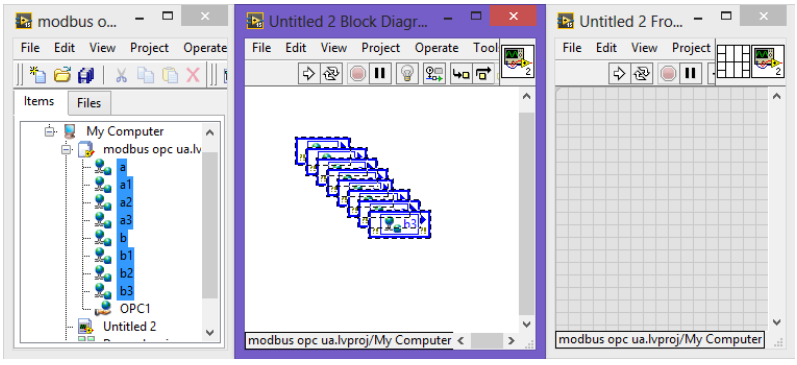
Gambar 4.90 Tag a dan b di setiap Device dibuat variable untuk program LabVIEW

72. Berikutnya, tambahkan program LabVIEW pada Project. Klik kanan My Computer, pilih New, pilih VI.



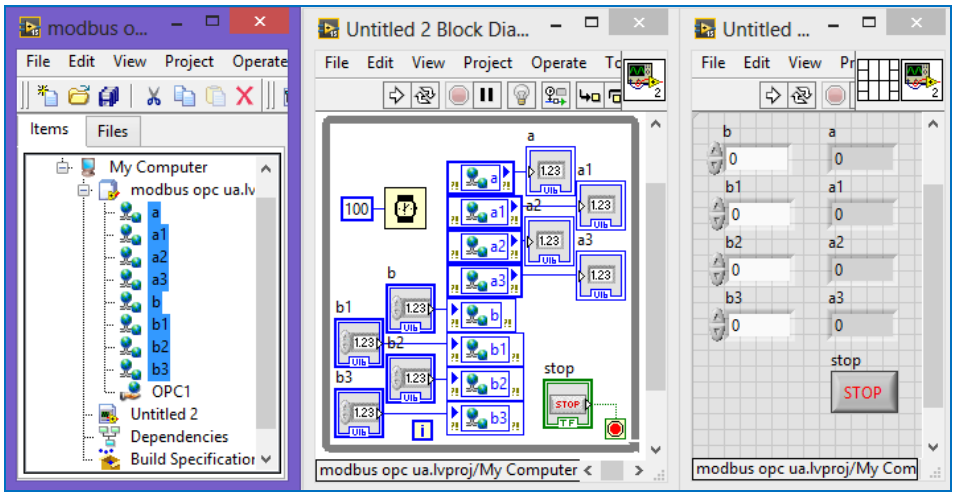
Gambar 4.91 Tambahkan program LabVIEW dengan klik kanan My Computer, pilih New, VI

73. Tarik dan drag kedelapan variable dari kotak Project ke Block Diagram.



Gambar 4.92 Drag kedelapan variable dari kotak Project ke Block Diagram

74. Buat variable a, a1, a2 dan a3 diberi Indicator, sedangkan variable b, b1, b2 dan b3 diberi Control. Agar dapat diberi Control, ubah Access Mode keempat variable tersebut dari Read menjadi Write. Untuk menambahkan Control/Indicator, klik kanan kaki icon, pilih Create, pilih Control/Indicator.
75. Tambahkan Struktur While Loop, Tombol Stop dan Wait(ms) = 100.

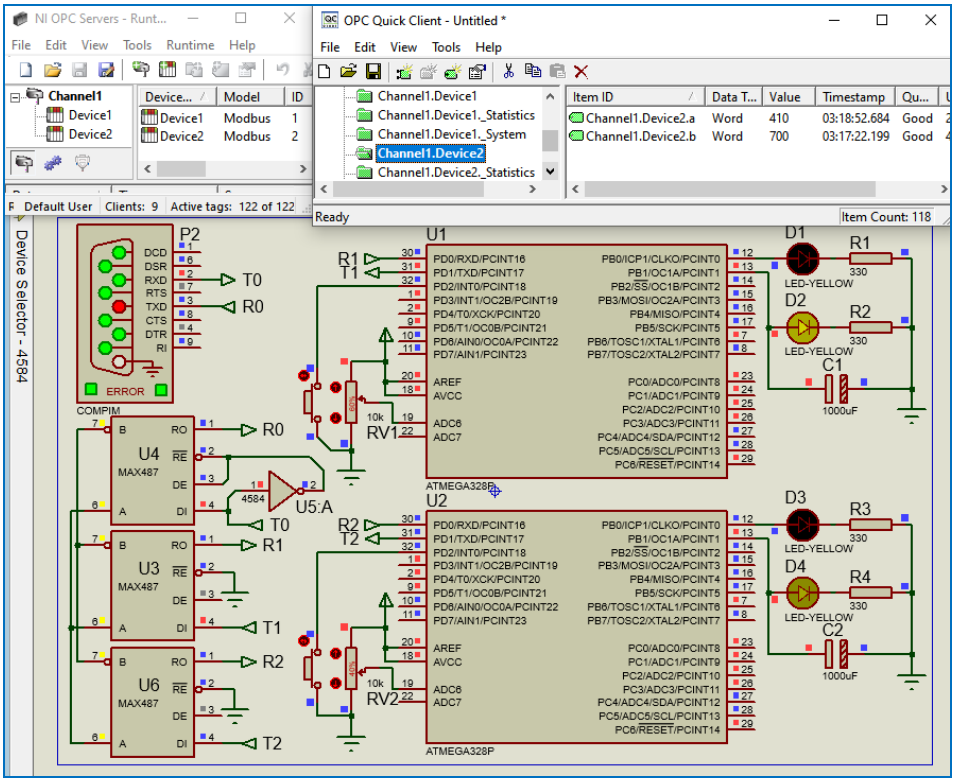


Gambar 4.93 Program LabVIEW untuk Read dan Write Tag pada OPC Server

Keterangan program:

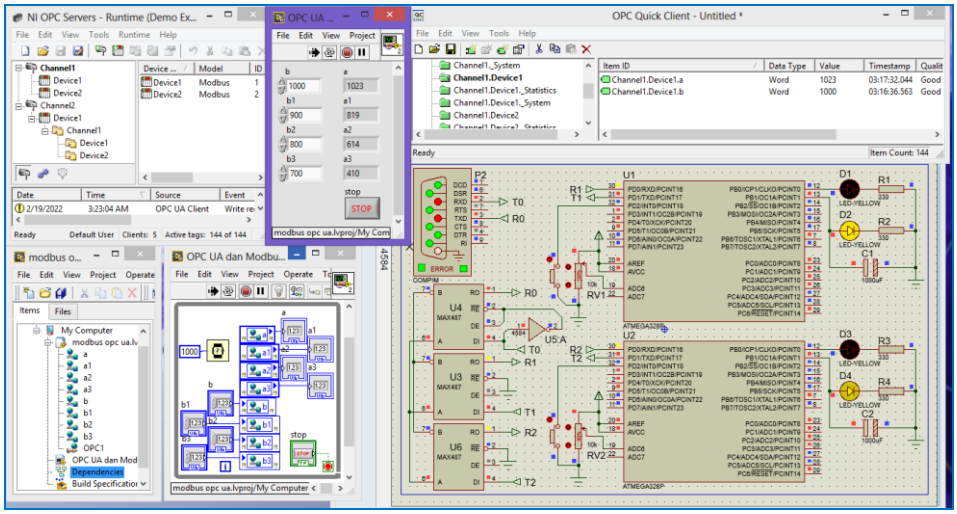
1. Indicator a akan menampilkan data dari Tag a di Device1 Channel1.
2. Control b akan mengatur data Tag b di Device1 Channel1.
3. Indicator a1 akan menampilkan data dari Tag a di Device2 Channel1.
4. Control b1 akan mengatur data Tag b di Device2 Channel1.
5. Indicator a2 akan menampilkan data dari Tag a di Device1 Channel2.
6. Control b2 akan mengatur data Tag b di Device1 Channel2.
7. Indicator a3 akan menampilkan data dari Tag a di Device2 Channel2.
8. Control b3 akan mengatur data Tag b di Device2 Channel2.
9. Channel1 menggunakan protokol Modbus RTU, sedangkan Channel2 menggunakan OPC UA. Dengan demikian, program LabVIEW pada Gambar 4.93 di atas adalah program Modbus Master sekaligus OPC UA Client. Dengan bantuan NI OPC Server, program gabungan tersebut menjadi sederhana.

76. Sebelum program LabVIEW di atas dijalankan, semua rangkaian Modbus RTU di Proteus, baik yang di komputerA maupun di komputerB harus dijalankan, begitu juga semua OPC Quick Client juga dijalankan, seperti terlihat pada gambar berikut ini. Pastikan NI OPC Servers Runtime belum habis waktunya, apabila sudah habis, silahkan direset kembali (lihat langkah no. 4-14 di Sub Bab 4.2 untuk mereset waktu Runtime). Pastikan kualitas komunikasi di OPC Quick Client untuk setiap Device bernilai Good.



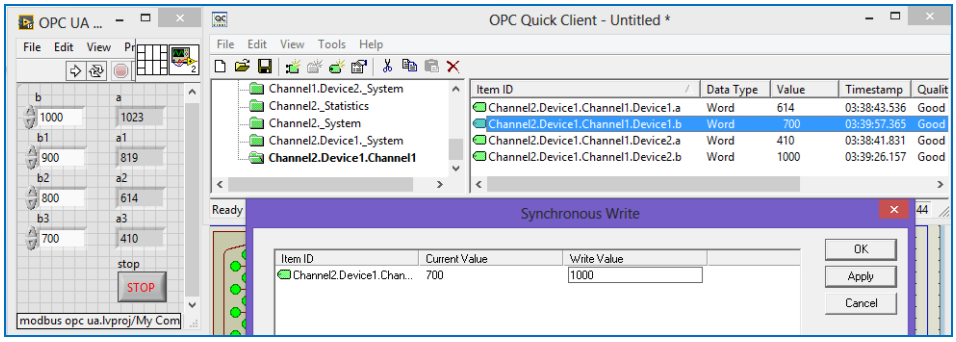
Gambar 4.94 Rangkaian Modbus RTU di Proteus di komputerA (UA Server), Quality = Good

77. Setelah semua rangkaian di Proteus di komputerA dan di komputerB dijalankan, begitu juga OPC Quick Client dijalankan, dan semua Quality komunikasi untuk setiap Device bernilai Good, maka jalankan program LabVIEW. Seharusnya Indicator a, a1, a2 dan a3 menampilkan nilai potensio di setiap rangkaian Modbus RTU di komputerA dan komputerB.

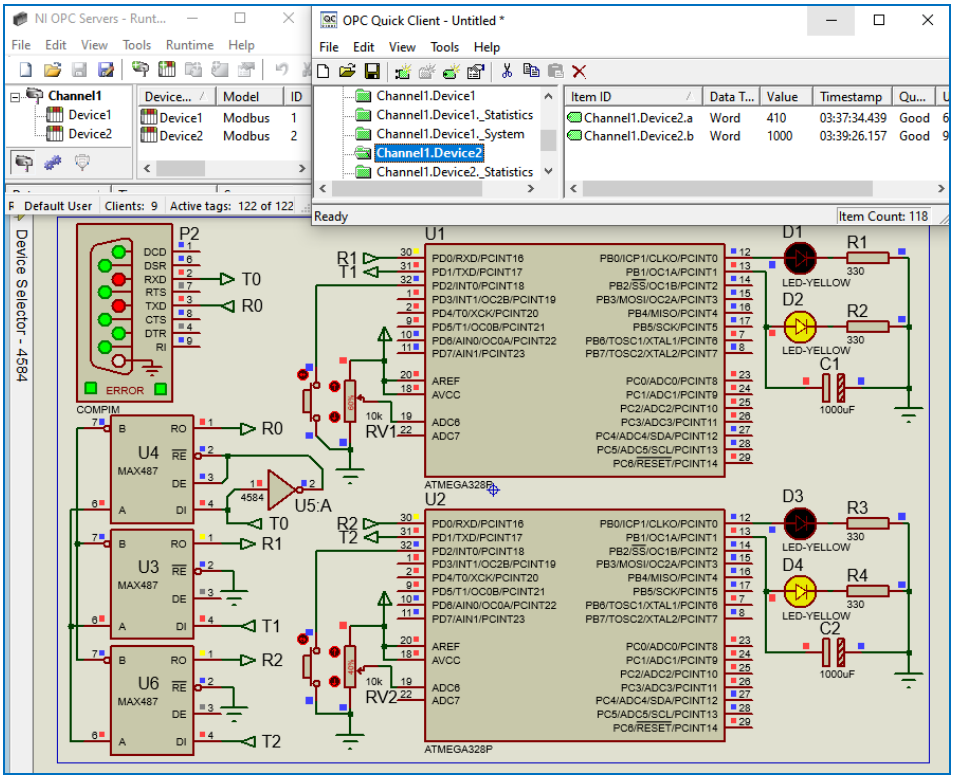


Gambar 4.95 Front Panel LabVIEW menampilkan nilai potensio dari 4 rangkaian Modbus RTU yang disimulasikan di Proteus (2 Modbus RTU di komputerA dan 2 Modbus RTU di komputerB)

78. Atur Control b, b1, b2 dan b3 di Front Panel LabVIEW berturut-turut menjadi 1000, 900, 800 dan 700. Perhatikan bahwa LED PWM di keempat rangkaian Modbus RTU menyala dengan intensitas cahaya yang berbeda (lihat rangkaian di Proteus di Gambar 4.94 dan 4.95 di atas).
79. OPC Quick Client perlu dijalankan untuk mendapatkan akses data Tag, sehingga membuat variable (di LabVIEW) yang berkaitan dengan Tag tersebut, ikut ter-update nilainya. OPC Quick Client selain untuk mendapatkan data Tag (Read), juga dapat mengatur (Write) data Tag.



Gambar 4.96 Nilai Tag b di Device1 Channel2 di komputerA (OPC UA Server) diubah dari 700 menjadi 1000 dengan Synchronous Write OPC Quick Client di komputerB



Gambar 4.97 Hasil pengubahan nilai Tag b Device1 dan 2 di Channel2 (komputerA) dengan OPC Quick Client di komputerB, tampak bahwa kedua LED PWM menyala terang (nilai 1000)

80. Simpan program dan project LabVIEW. Sampai di sini, pembuatan program LabVIEW OPC UA dengan NI OPC Server selesai.

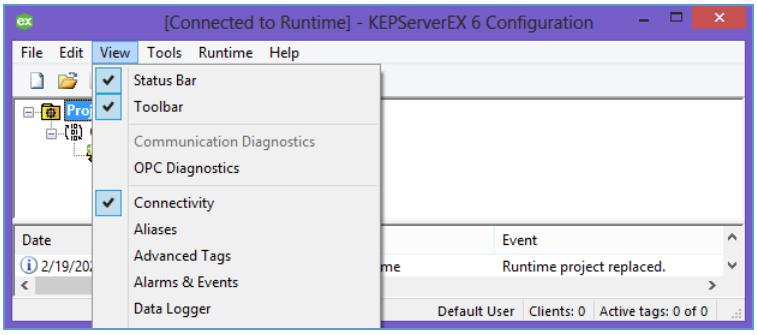
4.4 OPC UA dengan KEPServerEX Kepware

Agar menjawab soal Tantangan pada Gambar 4.1, maka pada Sub Bab ini, akan diperlihatkan bagaimana menghubungkan dua buah jaringan Modbus Slave RTU menggunakan OPC UA, dengan tampilan Pengisian Tangki seperti yang dibuat pada Bab 1. Berikutnya, sesuai dengan judul Sub Bab ini, maka di Sub Bab ini akan diperlihatkan bagaimana LabVIEW juga dapat men-"*deploy*" OPC Server dari Kepware (sebenarnya NI OPC Server juga dibuat oleh Kepware), yang diberi nama KEPServerEX. Mengapa menggunakan KEPServerEX? Karena KEPServerEX lebih

banyak fiturnya dibandingkan NI OPC Server. Salah satu fitur KEPServerEX yang belum ada di NI OPC Server adalah IoT Gateway. IoT Gateway ini bisa menghubungkan Device dengan layanan IIoT (Industrial Internet of Things). Di samping itu, KEPServerEX ini sama seperti NI OPC Server, sehingga memudahkan pengguna yang sudah terbiasa dengan NI OPC Server.

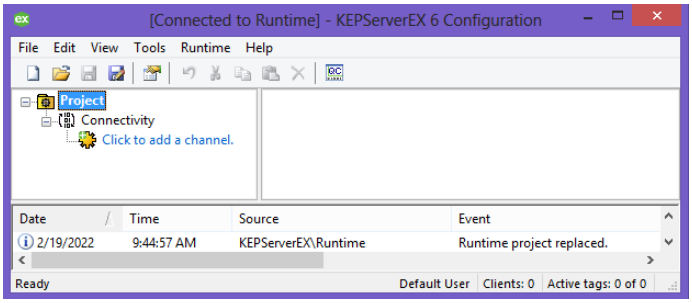
Berikut ini langkah-langkah pembuatan program LabVIEW untuk menghubungkan 2 buah jaringan Modbus RTU Pengisian Tangki di 2 buah komputer menggunakan OPC UA KEPServerEX:

1. Instal software KEPServerEX di komputerA dan komputerB.
2. KomputerA akan dibuat menjadi OPC UA Client, sedangkan komputerB akan dibuat menjadi OPC UA Server. Mulai dari komputerB (OPC UA Server), buat rangkaian di Proteus di komputerB, seperti Gambar 3.38.
3. Buat program Outseal seperti Gambar 3.39. Kompilasi program Outseal tersebut, dengan seting alamat Modbus untuk Arduino U1 adalah 1, dan seting alamat Modbus untuk Arduino U2 adalah 2.
4. Setelah rangkaian Modbus RTU Slave di Proteus selesai dibuat, langkah berikutnya adalah mengatur seting modbus RTU Master di OPC Server KEPServerEX. Buka KEPServerEx Administration dan KEPServerEX Configuration. Pada KEPServerEX Configuration, pilih menu File, pilih New. Muncul pertanyaan "*This operation will cause …*", pilih Yes, Update, untuk memulai dari awal (membersihkan Channel dan Device sebelumnya).
5. Berikutnya, buka menu View, hilangkan semua tanda centang, kecuali pada Status Bar, Toolbar dan Connectivity.



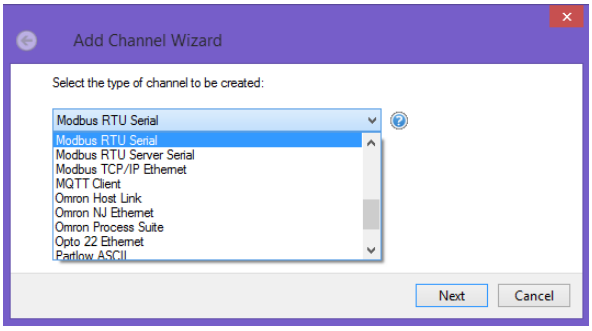
Gambar 4.98 Hilangkan semua tanda centang kecuali di Status Bar, Toolbar dan Connectivity

6. Berikutnya, Click to add a channel.



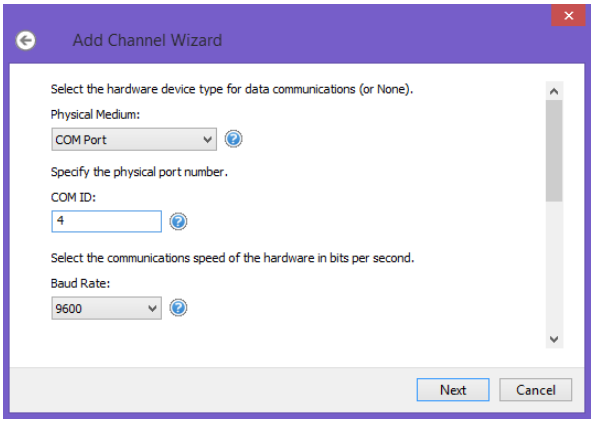
Gambar 4.99 Buat Channel untuk Device Modbus RTU Master dengan Click to add a channel

7. Pilih Modbus RTU Serial.



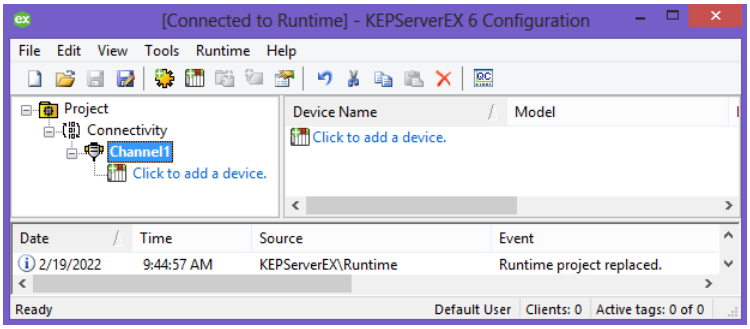
Gambar 4.100 Pilih Modbus RTU Serial

8. Atur Port COM sesuai dengan pasangan COM di COMPIM Proteus.



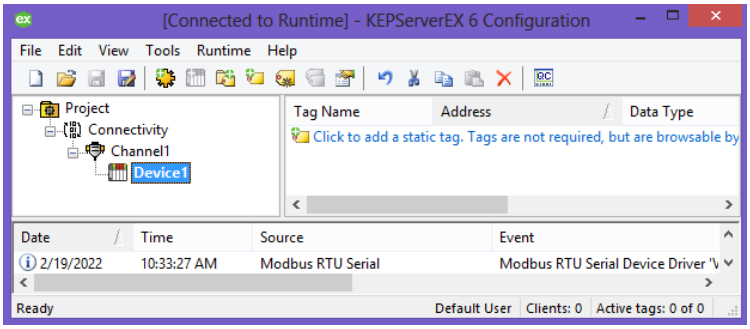
Gambar 4.101 Atur Port COM sesuai dengan psangan COM di COMPIM Proteus

9. Klik Next hingga selesai (Finish), maka Channel1 muncul di bawah Connectivity. Tambahkan Device pada Channel1, Click to add a device.



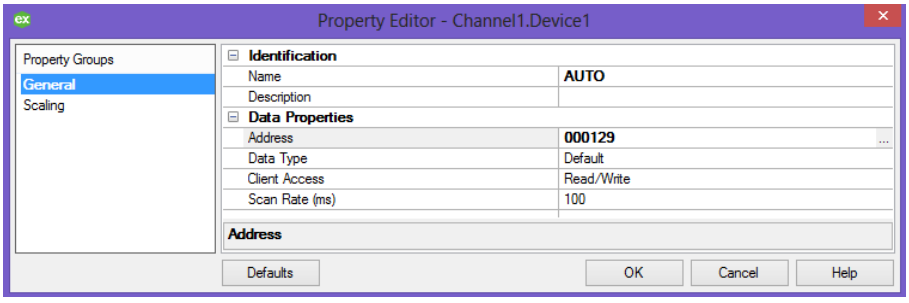
Gambar 4.101 Setelah Channel1 muncul, tambahkan Device dengan Click to add a device

10. Klik Next hingga selesai (Finish), maka Device1 di Channel1 muncul.



Gambar 4.102 Klik Next hingga selesai, maka muncul Device1 di bawah Channel1

11. Tambahkan Tag pada Device1 dengan meng-klik *Click to add a static tag*. Beri nama Tag = AUTO, dengan Address = 000129.



Gambar 4.103 Menambahkan Tag AUTO (000129) pada Device1

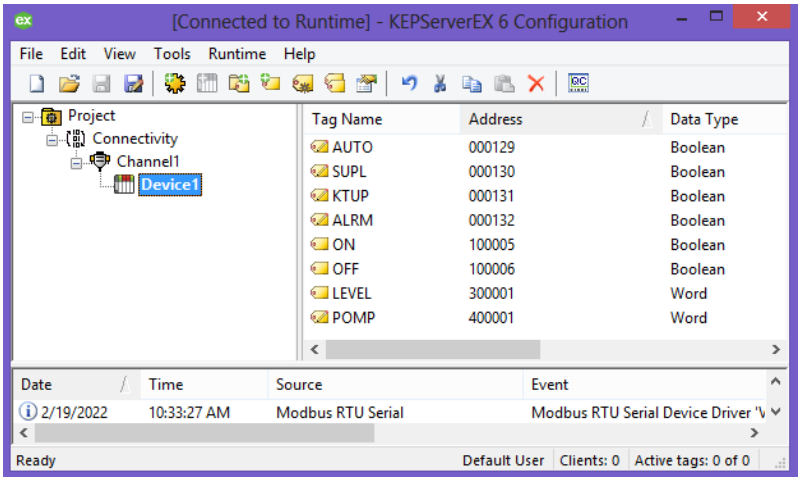
12. Ulangi penambahan Tag pada Device1, satu-persatu sesuai tabel berikut:

Tabel 4.1 Daftar Tag pada Device1

No.	Tag Name	Address	Client Access
1.	AUTO	000129	Read/Write
2.	SUPL	000130	Read/Write
3.	KTUP	000131	Read/Write
4.	ALRM	000132	Read/Write
5.	ON	100005	Read Only
6.	OFF	100006	Read Only
7.	LEVEL	300001	Read Only
8.	POMP	400001	Read/Write

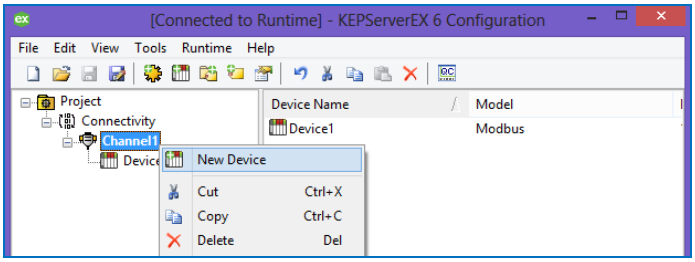
Catatan: tidak perlu mengatur isi Client Access, karena isinya akan secara otomatis mengikuti Address. Klik OK pada Tag, maka bila angka pertama Address: 1 atau 3 Client Access otomatis terisi Read Only, bila 0 atau 4: otomatis terisi Read/Write.

13. Tambahkan Tag pada Device1 sesuai Tabel 4.1 sehingga seperti berikut:



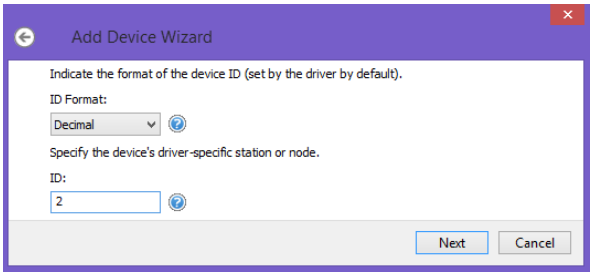
Gambar 4.104 Menambahkan Tag pada Device1 sesuai Tabel 4.1

14. Buat Device2 dengan meng-klik kanan Channel1, dan pilih New Device.



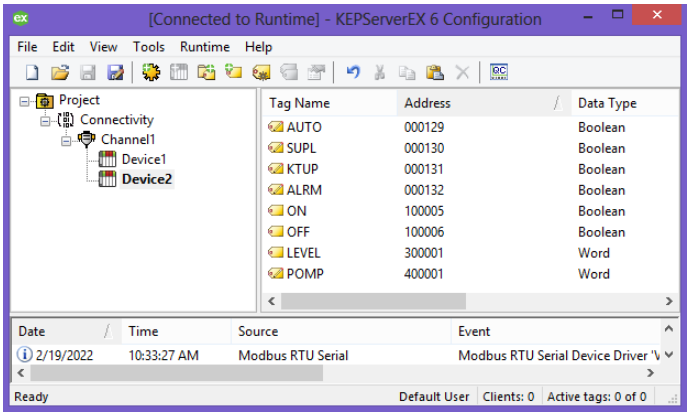
Gambar 4.105 Tambahkan Device2 di Channel1

15. Di jendela Add Device Wizard, Klik Next hingga sampai ID Device, buat agar ID Device2 = 2. Kemudian klik Next hingga selesai.



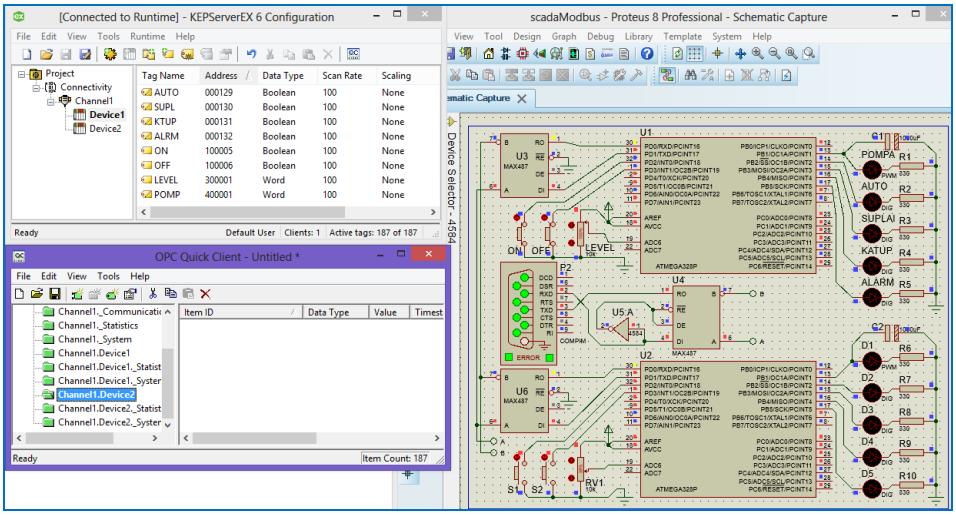
Gambar 4.106 Tambahkan Device2 di Channel1

16. Muncul Device2 di bawah Device1. Copy (Ctrl+C) semua Tag di Device1, dan Paste (Ctrl+V) di Device2, sehingga menjadi seperti berikut:



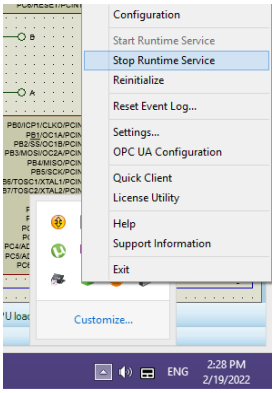
Gambar 4.107 Copy Paste semua Tag di Device1 ke Device2

17. Berikutnya, jalankan rangkaian Proteus Gambar 3.38, yang sudah diisi dengan file Hex hasil kompilasi program Outseal di Gambar 3.39. Jalankan juga OPC Quick Client dengan menekan tombol QC di KEPServerEX.

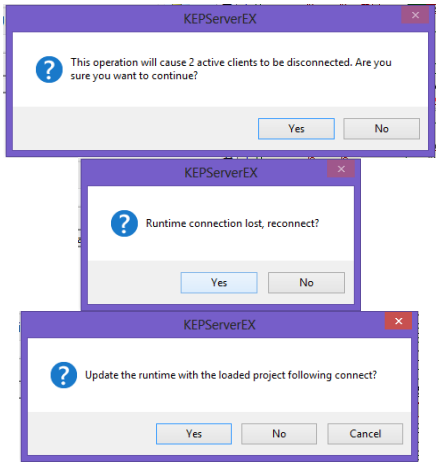


Gambar 4.108 Jalankan rangkaian 2 buah Modbus RTU Slave di Proteus dan Modbus RTU Serial (Master) Device1 dan Device2 di OPC Server (KEPServerEX)

Catatan: Ketika data Tag pada OPC Quick Client kosong, ini berarti waktu Runtime telah habis. Agar waktu Runtime bisa berjalan lagi, reset Runtime dengan cara meng-klik kanan icon KEPServerEX Administration, dan pilih Stop Runtime Service, seperti Gambar 4.109 dan jawab semua pertanyaan dengan klik Yes.

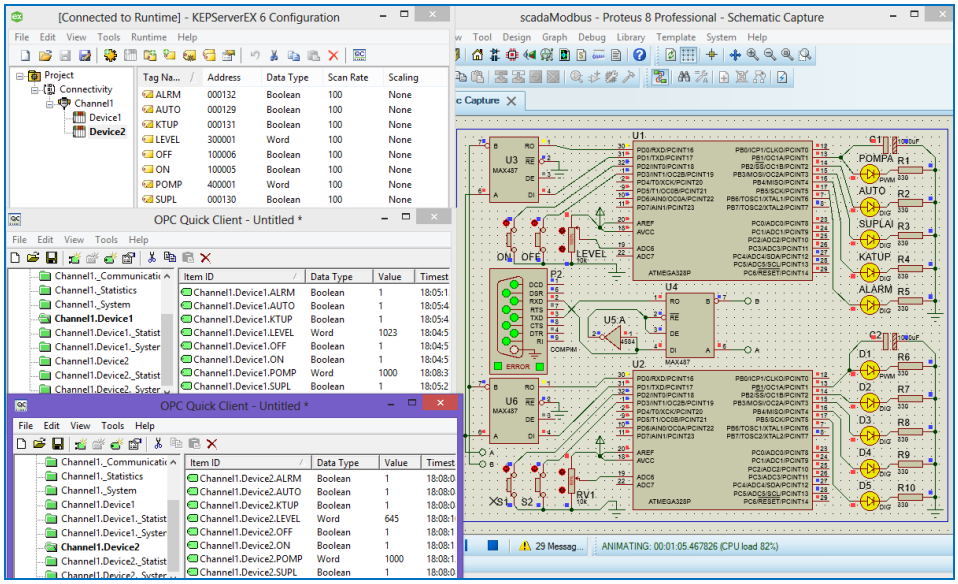


Gambar 4.109 Klik kanan icon KEPServerEX Administration, pilih Stop Runtime Service



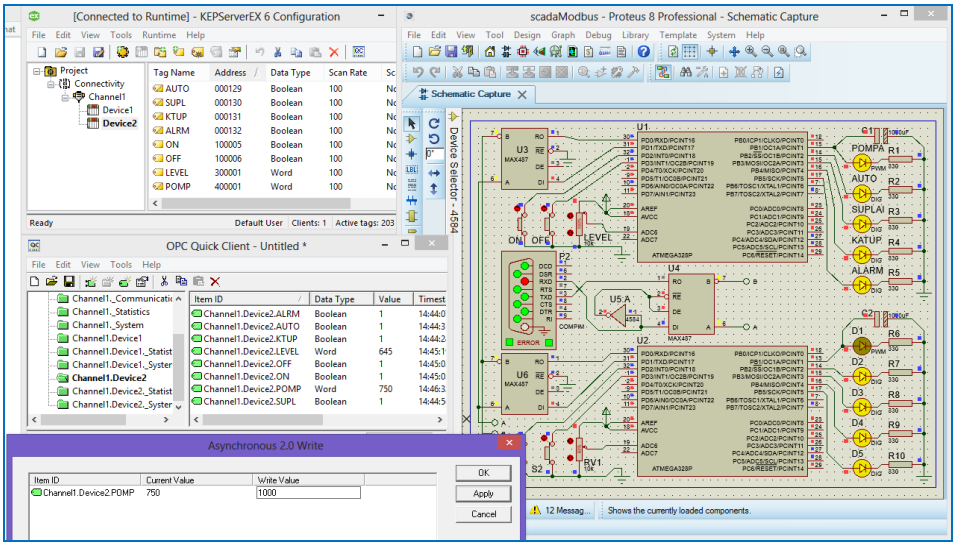
Gambar 4.110 Pilih Yes pada semua pertanyaan yang muncul (kecuali Save)

18. Ketika waktu Runtime telah direset, maka OPC Quick Client akan dapat menampilkan nilai semua Tag di Device1 dan Device2.



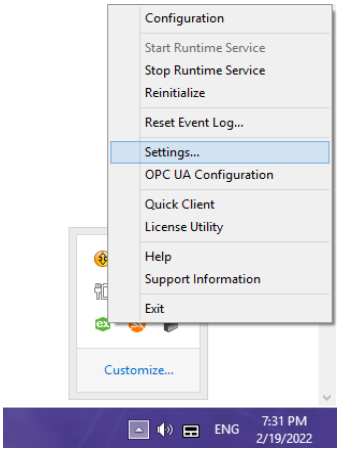
Gambar 4.111 Ketika Runtime sudah direset, maka OPC Quick Client menampilkan nilai Tag

19. Lakukan pengubahan nilai Tag dengan meng-klik kanan nilai Tag (khusus jenis Tag yang Read/Write, angka pertama 0 atau 4), pilih Synchronous atau Asynchronous Write, isikan nilai yang baru pada kolom, klik OK.



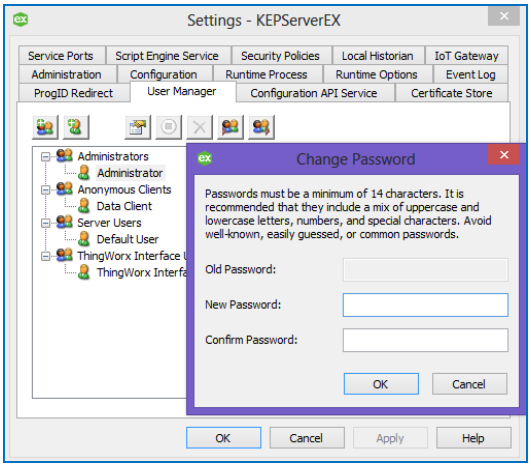
Gambar 4.112 Pengubahan nilai Tag dari OPC Quick Client dengan Asynchronous Write

20. Setelah kedua Device Modbus RTU Master di Channel1 berhasil dijalankan, berikutnya membuat komputerB menjadi OPC UA Server. Untuk membuat OPC UA Server, diperlukan otentikasi pengguna tingkat Administrator. Pengaturan password untuk pengguna tingkat Administrator ini dilakukan di Settings KEPServerEX Administration. Klik kanan icon KEPServerEX Administration, pilih Settings.



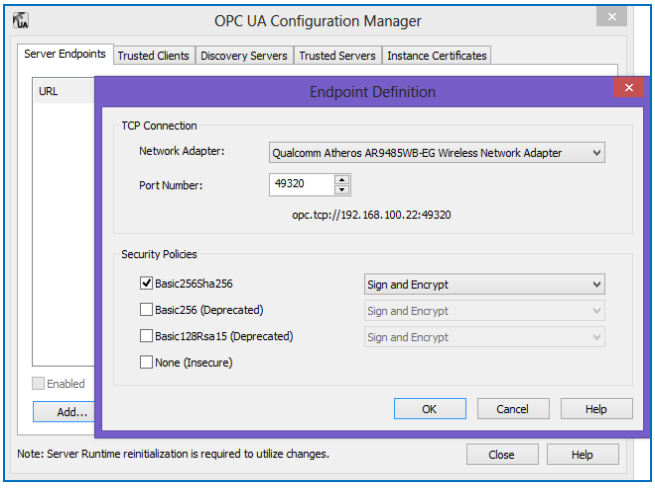
Gambar 4.113 Untuk pengaturan password user Administrator, klik kanan icon KEPServerEX Administration (ada di pojok kanan bawah, icon dengan tulisan "ex"), pilih Settings

21. Pada jendela Settings KEPServerEX, pilih Tab User Manager, kemudian klik 2 kali pada Administrator, maka muncul kotak Change Password. Masukkan password pada kolom dengan minimal 14 karakter.



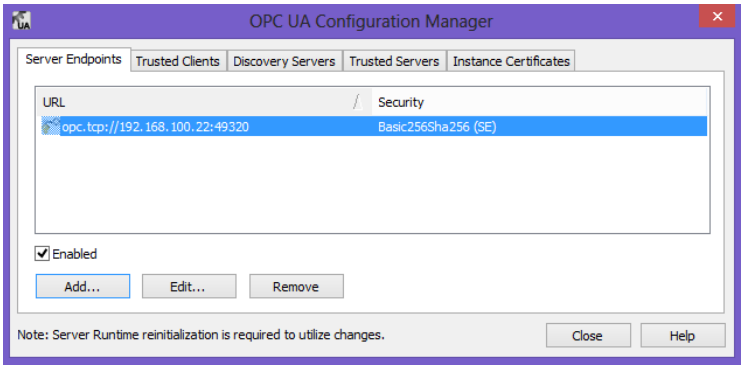
Gambar 4.114 Pilih User Manager, klik 2 kali Administrator, masukkan password 14 karakter

22. Berikutnya, buka OPC UA Configuration. Klik kanan icon KEPServerEX Administration, pilih OPC UA Configuration. Pada Tab Server Endpoints, klik tombol Add. Pilih Network Adapter sesuai nama Adapter (jangan memilih Default atau Localhost, agar bisa muncul alamat IP), klik OK.



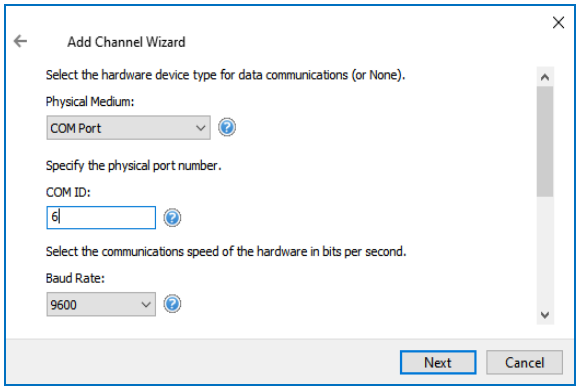
Gambar 4.115 Di Tab Server Endpoints, klik tombol Add, pilih Network Adapter, klik OK

23. Muncul URL di Server Endpoints, dengan port 49320 (port untuk NI OPC Server = 49350). Pastikan opsi Enabled tercentang.



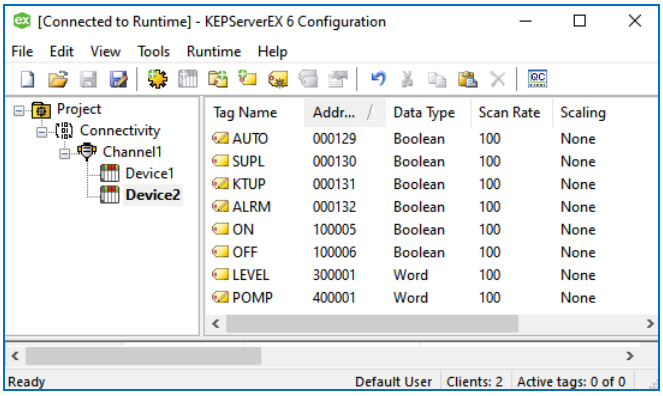
Gambar 4.116 Muncul URL di Server Endpoints dengan port 49320, Enabled tercentang

24. Setelah pengaturan OPC UA Configuration di komputerB (seperti Gambar 4.116 di atas) selesai, berikutnya beralih ke komputerA.
25. Di komputerA, buat Channel1, dengan tipe Modbus RTU Serial. Atur Port COM sesuai dengan COM yang berpasangan dengan COMPIM di Proteus.



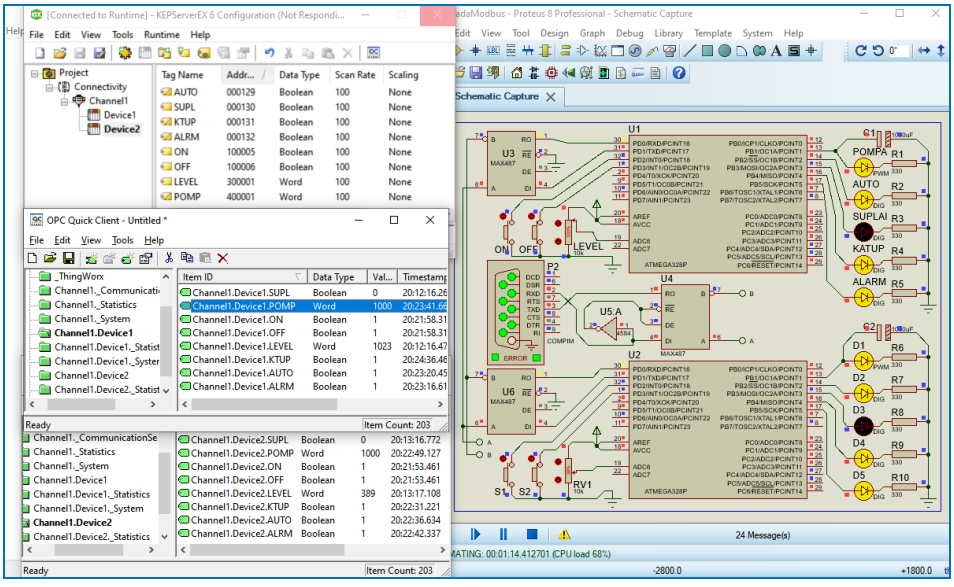
Gambar 4.117 COM ID diisi dengan COM yang berpasangan dengan COMPIM di Proteus

26. Tambahkan Device1 dan Device2 pada Channel1 tersebut, jangan lupa mengubah nomor ID Device2 dengan angka 2 (default ID adalah 1).
27. Tambahkan 8 buah Tag pada Device1 seperti Tabel 4.1 di atas.
28. Ulangi untuk Device2, gunakan Copy dan Paste untuk mempercepat penambahan Tag pada Device2.



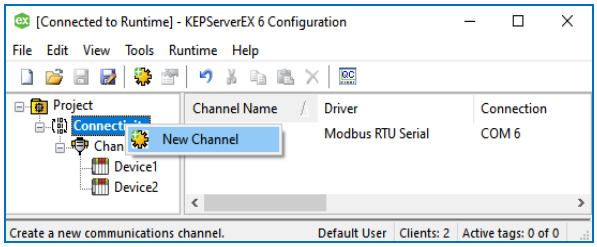
Gambar 4.118 Device1 dan Device2 terisi dengan 8 buah Tag sesuai Tabel 4.1

29. Berikutnya, di komputerA, buka rangkaian 2 buah Arduino di Proteus sebagai simulasi perangkat Modbus RTU Slave, seperti Gambar 3.38. Isi kedua Arduino dengan file Hex hasil kompilasi program Outseal, dengan alamat Modbus Slave di U1 adalah 1 dan di U2 adalah 2. Jalankan rangkaian Proteus, dan juga buka OPC Quick Client. Seharusnya OPC Quick Client dapat menampilkan nilai semua Tag di Device1 dan Device2.



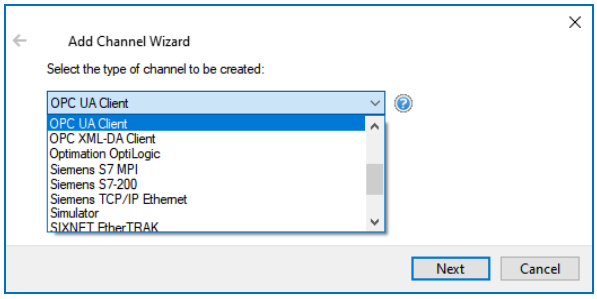
Gambar 4.119 OPC Quick Client menampilkan nilai data Tag Modbus RTU Slave Device1 dan 2

30. Apabila OPC Quick Client tidak menampilkan data apapun, berarti waktu Runtime habis, silahkan direset dengan meng-klik Stop runtime Service.
31. Setelah Channel1 dengan protokol Modbus RTU Serial dapat dijalankan di komputerA, maka langkah berikutnya adalah menambahkan Channel2 dengan tipe OPC UA Client. Klik kanan Connectivity, pilih New Channel.



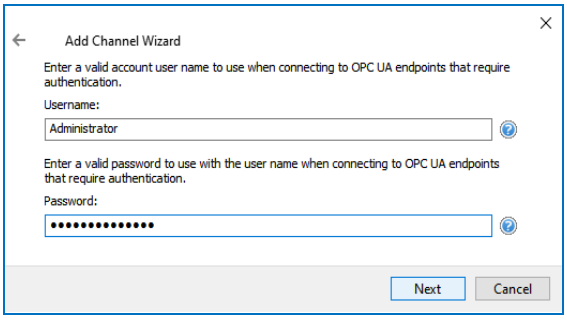
Gambar 4.120 Tambahkan Channel2 untuk OPC UA Client

32. Klik Next pada jendela Channel Wizard. Pilih tipe OPC UA Client. Klik Next.



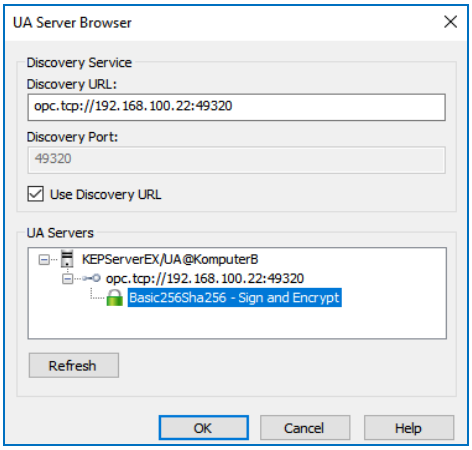
Gambar 4.121 Pilih OPC UA Client

33. Isi Username = Administrator dan password sesuai langkah no. 21 di atas.



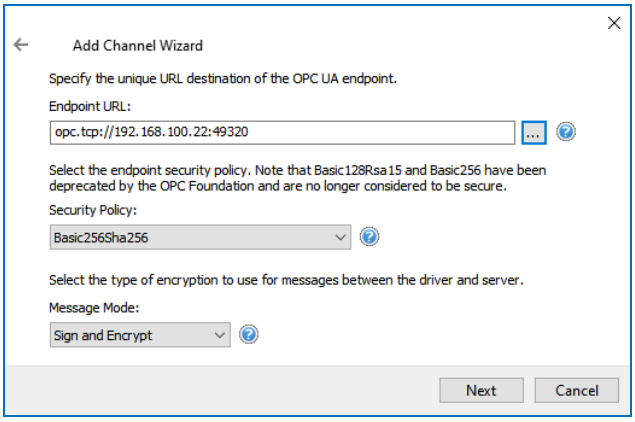
Gambar 4.121 Isi Username = Administrator, dan password sesuai langkah no. 21

34. Klik Next hingga sampai di isian Endpoint URL. Tekan tombol di samping kolom (…), hingga muncul UA Server Browser. Tulis URL komputerB pada kolom Discovery URL, beri tanda centang pada Use Discovery URL, tekan tombol Refresh. Seharusnya di kolom UA Servers muncul nama URL dan kode keamanan. Apabila belum muncul, di komputerB, buka OPC UA Configuration, di Tab Server Endpoints, ulangi langkah no. 22-23.



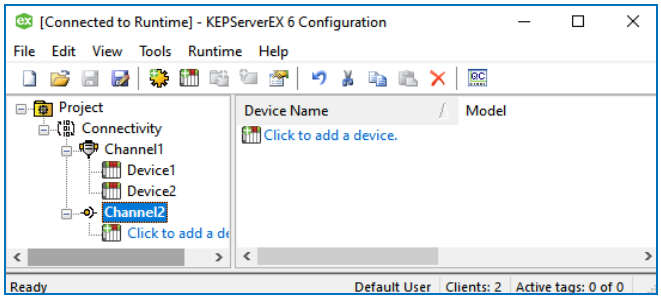
Gambar 4.122 Menggunakan Use Discovery URL untuk menemukan Endpoint URL

35. Setelah URL komputerB muncul di kolom UA Servers, klik pada kode keamanan (Basic256…), klik OK, maka kolom Endpoint URL akan terisi dengan kode URL komputerB (OPC UA Server).



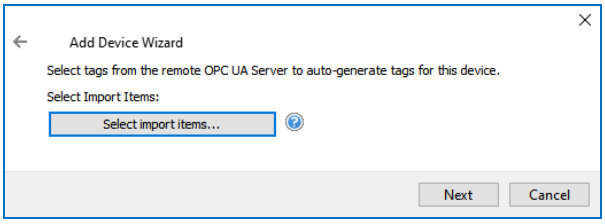
Gambar 4.123 Endpoint URL berisi URL komputerB yang menjadi OPC UA Server

36. Klik Next hingga selesai (Finish), maka muncul Channel2.



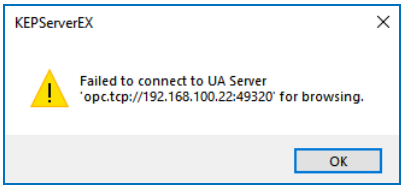
Gambar 4.124 Muncul Channel2 dengan tipe Channel: OPC UA Client

37. Tambahkan Device di Channel2 dengan meng-klik *Click to add a device*.
38. Pada jendela Add Device Wizard yang muncul, klik Next hingga sampai di Select import items.



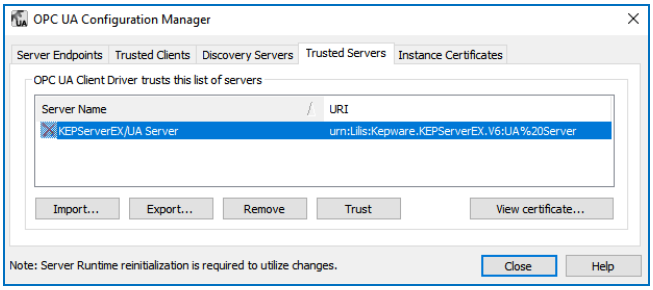
Gambar 4.125 Di jendela Device Wizard, klik Next hingga sampai di Select Import Items

39. Tekan tombol Select import items. Bila muncul pesan error "Failed to connect …", berarti OPC UA Client belum terhubung dengan UA Server.

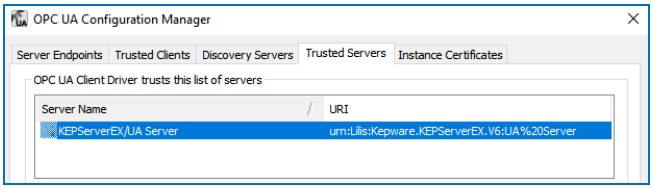


Gambar 4.126 Pesan error muncul karena OPC UA Client belum terhubung dengan UA Server

40. Untuk menghubungkan OPC UA Client dengan OPC UA Server, keduanya harus saling "Trust". Buka OPC UA Configuration di komputerA (OPC UA Client). Di Tab Trusted Servers, tampak nama Server dengan tanda silang di sampingnya, pilih nama Server tersebut dan tekan tombol Trust.

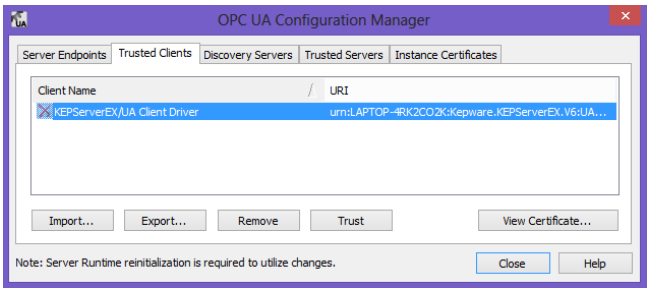


Gambar 4.127 Buka OPC UA Configuration di komputerA, di Tab Trusted Servers, terlihat ada UA Server (komputerB) dengan tanda silang di kirinya

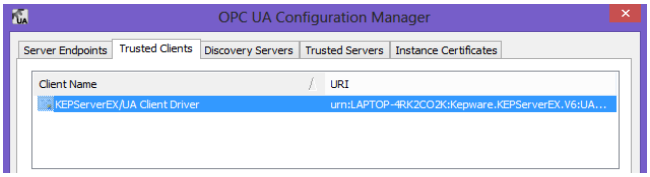


Gambar 4.128 Pilih UA Server tersebut, tekan tombol Trust, maka tanda silang hilang

41. Berikutnya buka OPC UA Configuration di komputerB. Di Tab Trusted Clients, muncul nama UA Client (komputerA) dengan tanda silang. Pilih UA Client tersebut, klik tombol Trust, maka tanda silang akan hilang.

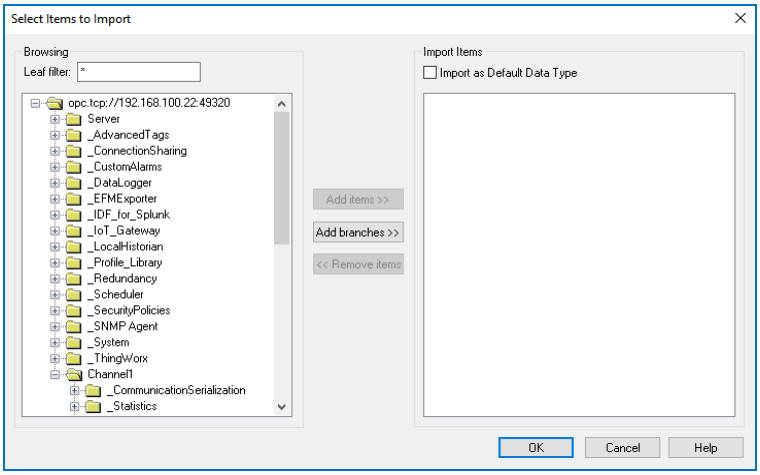


Gambar 4.129 Buka OPC UA Configuration di komputerB, di Tab Trusted Clients, terlihat ada UA Client (komputerA) dengan tanda silang di kirinya



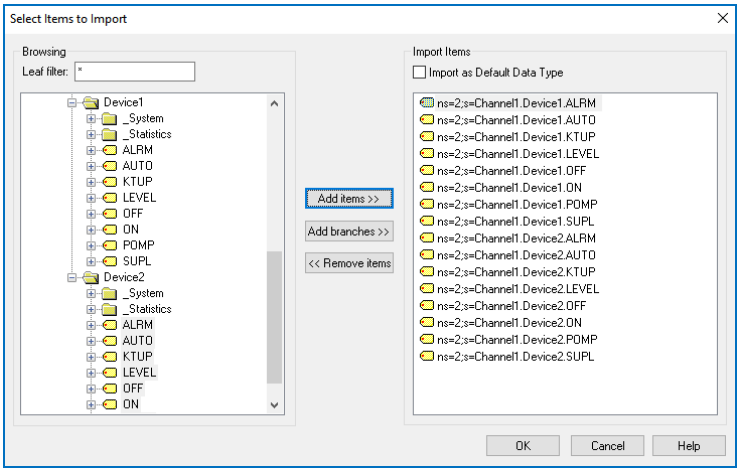
Gambar 4.130 Pilih UA Client tersebut, tekan tombol Trust, maka tanda silang hilang

42. Ulangi langkah no. 39, tekan tombol Select import items, maka akan muncul jendela Select Items to Import. Perhatikan bahwa dengan OPC UA, semua isi dari OPC Server dapat diimport menjadi Tag di OPC Client, termasuk semua protokol, driver dan layanan yang diatur di OPC Server.



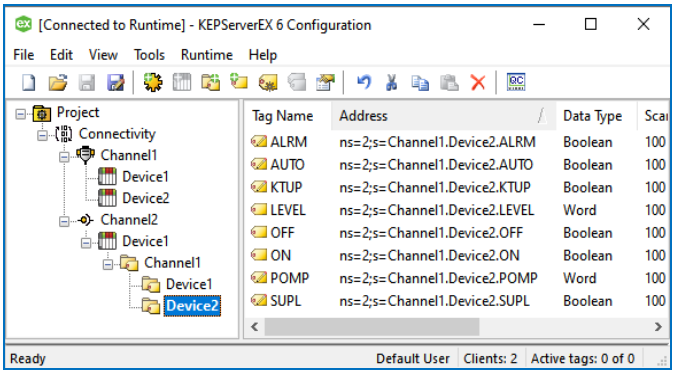
Gambar 4.131 Dengan OPC UA, semua isi OPC Server di komputerB yang menjadi Server, dapat di-import menjadi Tag di komputerA yang menjadi Client

43. Import semua Tag Device1 dan Device2, pilih Tag, tekan Add Items.



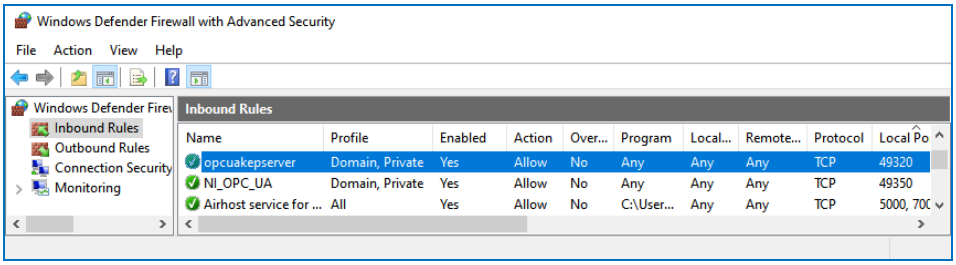
Gambar 4.132 Mengimport semua Tag Device1 dan Device2 ke Client (komputerA)

44. Klik OK, maka semua Items yang diimport muncul di Device1 di Channel2.



Gambar 4.133 Di KEPServerEX komputerA, di Channel2, tepatnya di Device1, berisi Channel1 dengan semua Device dan Tagnya, yang diimport dari komputerB selaku OPC UA Server

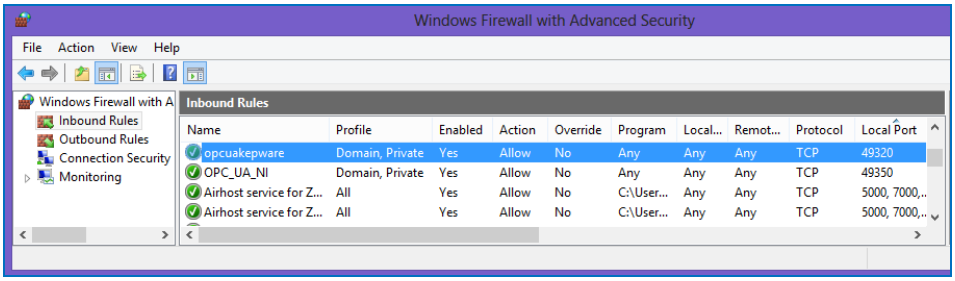
45. Setelah semua Tag yang diimport muncul di KEPServerEX komputerA, berikutnya jalankan OPC Quick Client. Apabila OPC Quick Client tidak dapat menampilkan nilai Tag Device Channel2, ada 2 kemungkinan penyebabnya: yang pertama karena waktu Runtime telah habis, yang kedua karena Firewall. Untuk yang pertama, solusinya dengan mereset waktu Runtime. Untuk yang kedua, solusinya dengan menambahkan New Rule di Inbound Rules Windows Firewall, yang mengijinkan port 49320 dapat diakses oleh OPC UA, seperti pada langkah 51-58 di Sub Bab 4.3.



Gambar 4.134 Buat New Rule di Inbound Rules Windows Firewall komputerA, untuk mengijinkan port 49320 dapat diakses oleh OPC UA, seperti langkah 51-58 di Sub Bab 4.3

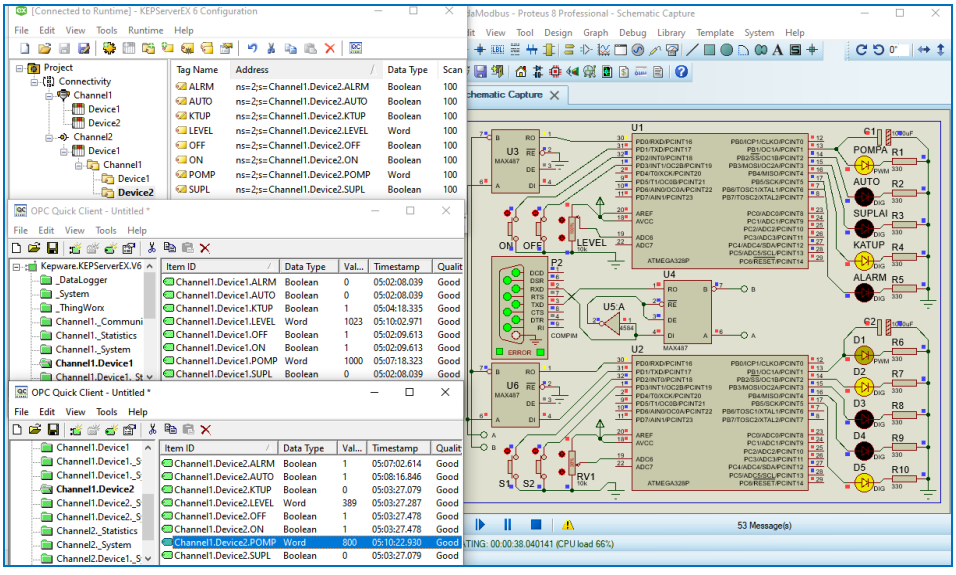
Catatan: perhatikan bahwa port untuk OPC UA Kepware KEPServerEX adalah 49320, sedangkan port untuk OPC UA National Instruments adalah 49350.

46. Buat New Rule yang sama di Inbound Rules Windows Firewall komputerB, yang mengijinkan port 49320 di komputerB dapat diakses oleh OPC UA.



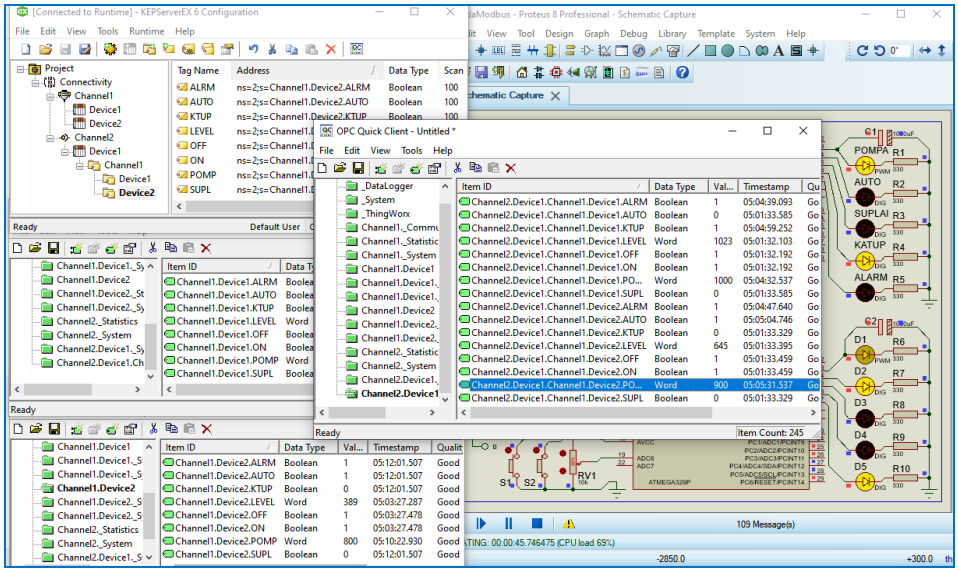
Gambar 4.135 Buat New Rule yang sama, yang mengijinkan akses port 49320 di komputerB

47. Setelah pembuatan New Rule untuk akses port 49320 selesai, jalankan kembali rangkaian Proteus, dan OPC Quick Client di komputerA dan komputerB. Seharusnya OPC Quick Client dapat menampilkan data Tag Device1 dan Device2 di Channel1. Lakukan juga pengubahan nilai Tag dengan meng-klik kanan nilai Tag, kemudian pilih Synchronous atau Asynchronous Write. Tag yang bisa diubah nilainya adalah Tag tipe Read/Write, yaitu Tag ALRM, AUTO, KTUP, SUPL dan POMP.



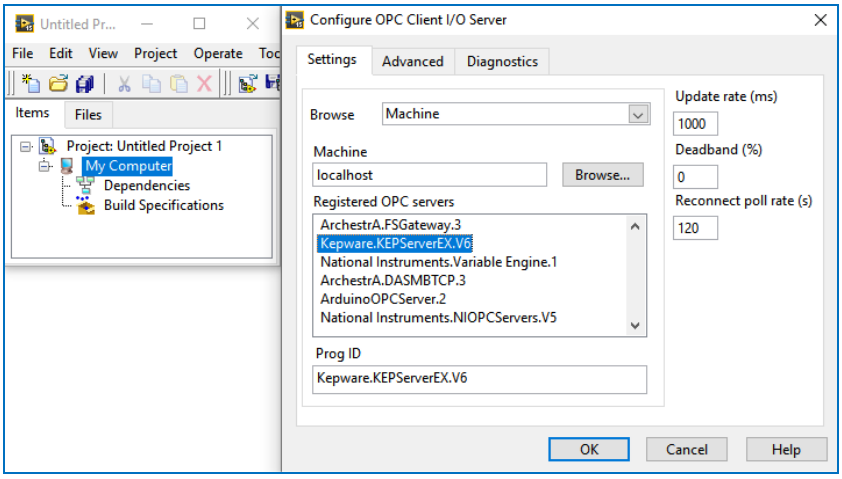
Gambar 4.136 Nilai Tag Device1 dan 2 di Channel1 dapat ditampilkan oleh OPC Quick Client. Nilai Tag juga dapat diatur dengan Synchronous/Asynchronous Write. Ubah Tag ALRM, AUTO, KTUP, SUPL dari 0 menjadi 1, dan Tag POMP dari 0 menjadi 1000, baik di Device1 maupun di Device2, maka seharusnya LED yang bersesuaian di rangkaian Proteus menyala.

48. Seharusnya OPC Quick Client di komputerA juga dapat menampilkan data semua Tag di Device1 dan Device2 di Channel2, yang merupakan data dari 2 buah rangkaian Modbus RTU di Proteus di komputerB. Lakukan juga pengaturan nilai Tag untuk Tag ALRM, AUTO, KTUP, SUPL dari 0 menjadi 1, dan Tag POMP dari 0 menjadi 1000, seharusnya LED yang bersesuaian di rangkaian Proteus di komputerB dapat menyala.

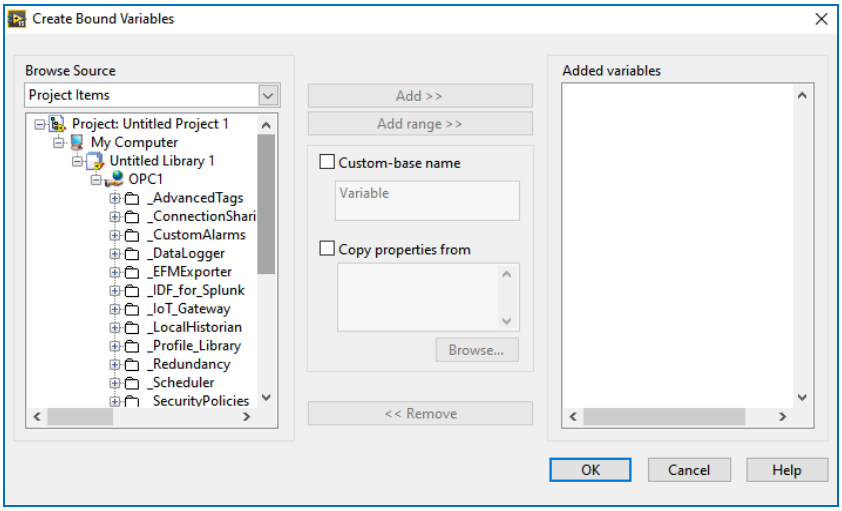


Gambar 4.137 OPC Quick Client menampilkan nilai Tag Device1 dan 2 Channel2 (komputerB)

49. Setelah nilai semua Tag di Device1 dan Device2 di Channel1 dan juga Device1 dan Device2 di Channel2 dapat ditampilkan (dan diubah nilainya) oleh OPC Quick Client, maka langkah berikutnya adalah membuat program LabVIEW yang dapat menampilkan dan mengatur nilai semua Tag pada keempat Device tersebut. Buat New Project program LabVIEW.
50. Di kotak Project, klik kanan My Computer, pilih New, pilih I/O Server, pilih OPC Client. Pada jendela Configure OPC Client I/O Server, di kotak Registered OPC servers, pilih Kepware.KEPServerEX.V6 (Gambar 4.138).
51. Klik OK, maka di kotak Project akan muncul Untitled Library1 dengan isi OPC1. Klik kanan OPC1, pilih Create Bound Variables, maka muncul jendela Create Bound Variables (lihat Gambar 4.139).

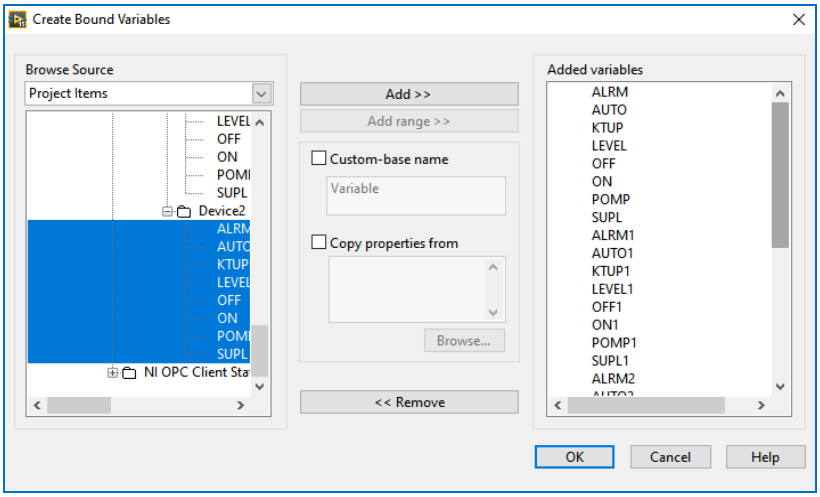


Gambar 4.138 Pilih OPC server = Kepware.KEPServerEX.V6

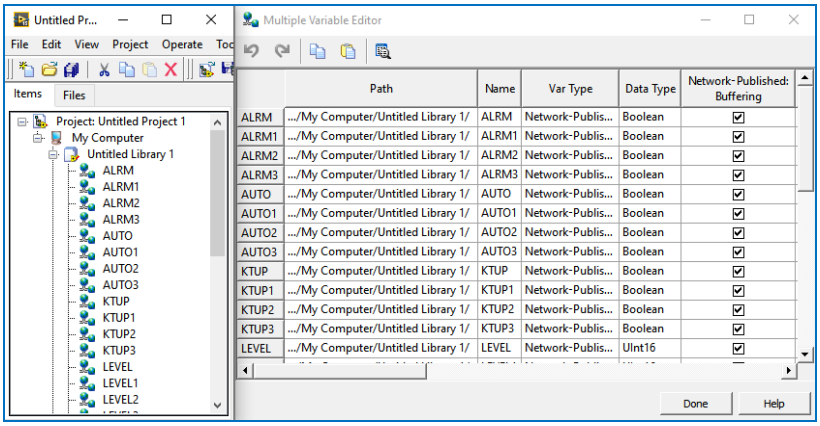


Gambar 4.139 Create Bound Variables, untuk membuat Tag menjadi variable di LabVIEW

52. Buat semua Tag di Device1 dan Device2 di Channel1 dan di Channel2 ditambahkan menjadi variable di LabVIEW, dengan memilih semua Tag tersebut, dan menekan tombol Add (lihat Gambar 4.140).
53. Klik OK, maka muncul jendela Multiple Variable Editor, yang menampilkan daftar nama variable yang telah dibuat. Nama-nama variable ini juga muncul di kotak Project, di bawah Untitled Library1 (lihat Gambar 4.141).



Gambar 4.140 Add semua Tag di Device1 dan Device2 di Channel1 dan Channel2



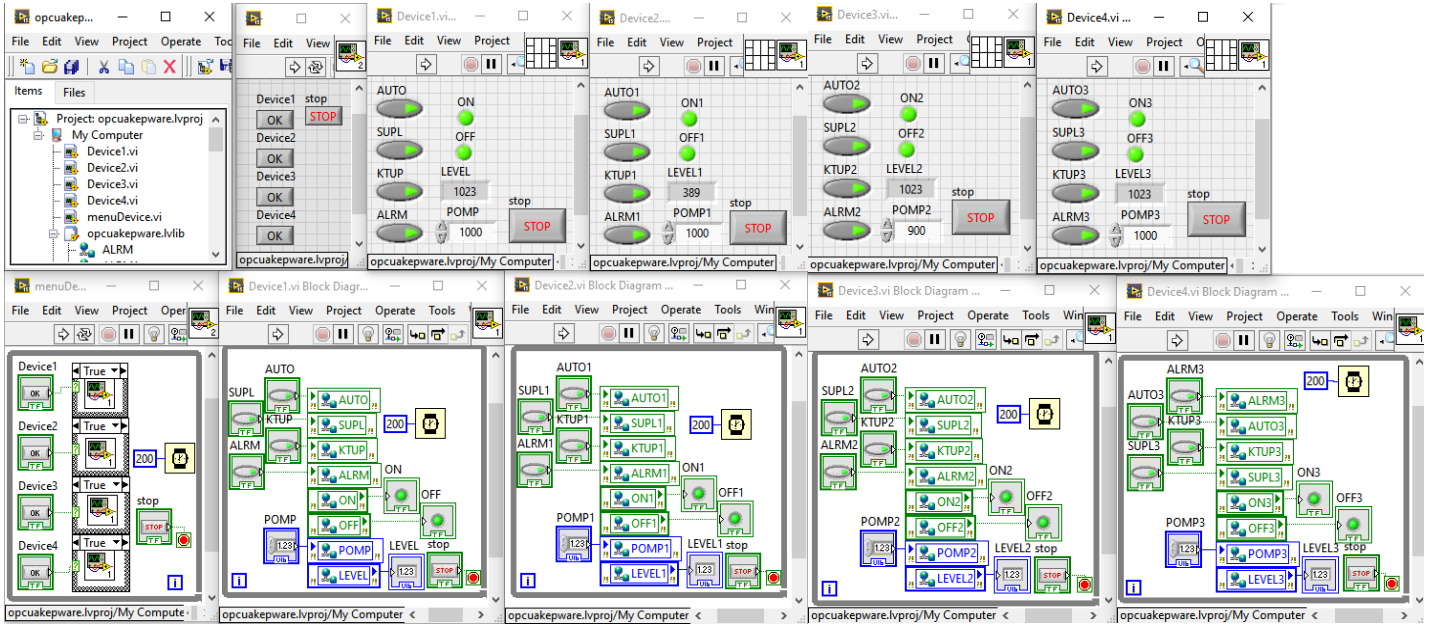
Gambar 4.141 Klik OK, semua variable (ada 32 variable) dimunculkan di kotak Project

54. Di kotak Project, tekan tombol Save All (this Project). Beri nama library dan project dengan nama yang sama, contoh “opcuakepware”.
55. Berikutnya, klik kanan My Computer, pilih New, pilih VI untuk menambahkan program LabVIEW. Beri nama Device1.
56. Tambahkan 3 buah program LabVIEW lagi, dan beri nama Device2, Device3 dan Device4.
57. Pilih 8 buah variable dari kotak Project, dan tarik ke Block Diagram Device1 hingga Device4, dengan pengaturan seperti tabel berikut:

Tabel 4.2 Pembagian 32 Variable ke dalam 4 Program LabVIEW

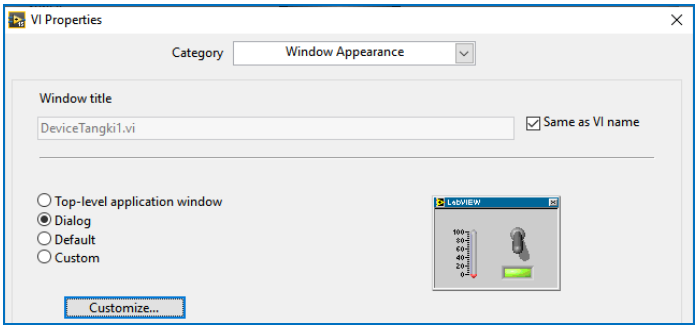
No.	Nama program	Variable
1.	Device1.vi	AUTO, SUPL, KTUP, ALRM, ON, OFF, LEVEL, POMP
2.	Device2.vi	AUTO1, SUPL1, KTUP1, ALRM1, ON1, OFF1, LEVEL1, POMP1
3.	Device3.vi	AUTO2, SUPL2, KTUP2, ALRM2, ON2, OFF2, LEVEL2, POMP2
4.	Device4.vi	AUTO3, SUPL3, KTUP3, ALRM3, ON3, OFF3, LEVEL3, POMP3

58. Setelah setiap Block Diagram dari keempat program terisi dengan 8 buah variable sesuai tabel di atas, berikutnya tambahkan While Loop, dan masukkan semua icon variable tersebut ke dalam While Loop. Tambahkan fungsi Wait(ms) dengan input 200 di dalam While Loop dan tombol Stop dengan meng-klik kanan terminal Condition, pilih Create, pilih Control.
59. Di Block Diagram program Device1, ubah Access Mode variable AUTO, SUPL, KTUP, ALRM dan POMP, dari yang semula Read, ubah menjadi Write. Kemudian satu persatu klik kanan kaki input variable-variable tersebut, pilih Create, pilih Control. Berikutnya, satu persatu klik kanan kaki output variable ON, OFF dan LEVEL, dan pilih Create, pilih Indicator
60. Ulangi langkah 59 di atas untuk program Device2, Device3 dan Device4.
61. Buat lagi sebuah program LabVIEW, dengan meng-klik kanan My Computer, pilih New, pilih VI, dan beri nama menuDevice.
62. Diinginkan setiap program, dari Device1 hingga Device4 dapat dipanggil dengan menekan tombol di menu Device. Untuk itu tambahkan 4 buah OK Button di Front Panel menuDevice, dan beri nama label berturut-turut Device 1, Device2, Device3 dan Device4.
63. Kemudian di Block Diagram menuDevice, ambil While Loop, dan tempatkan semua icon OK Button ke dalam While Loop. Tambahkan juga tombol Stop dan fungsi Wait(ms) dengan input 200 di dalam While Loop.
64. Berikutnya, tempatkan 4 buah Struktur Case di dalam While Loop, dan hubungkan terminal Case Selector (?) setiap Struktur Case ke OK Buton satu persatu. Kemudian tarik nama program Device1 hingga Device4 dari kotak Project, ke dalam kotak Struktur Case satu persatu.



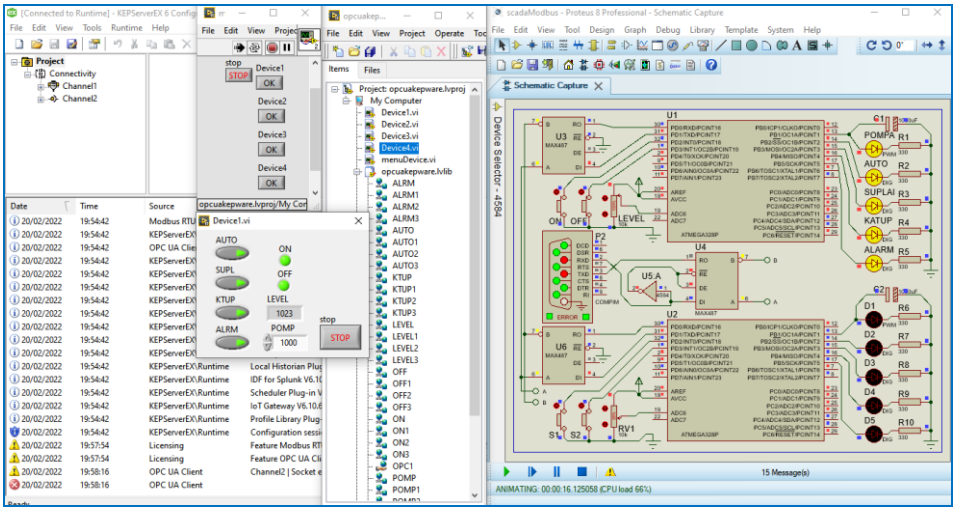
Gambar 4.142 Berturut-turut dari kiri ke kanan dan dari atas ke bawah: Kotak Project, Front Panel dan Block Diagram menuDevice.vi, Device1.vi, Device2.vi, Device3.vi, dan Device4.vi. Semua icon variable di Block Diagram diambil dari kotak Project dengan cara men-"drag" nama variable ke dalam Block Diagram. Begitu pula untuk icon Device1.vi, Device2.vi, Device3.vi dan Device3.vi, yang ada di kotak Struktur Case di Block Diagram menuDevice, semua icon tersebut diambil dari kotak Project dengan cara men-"drag" nama program vi tersebut ke dalam kotak Struktur Case di Block Diagram menuDevice.vi

65. Terakhir, agar Front Panel setiap program dari Device1 hingga Device4 dapat muncul setiap kali tombol di menuDevice ditekan, buat Window Appearance di VI Properties, di menu File, untuk program Device1.vi hingga Device4.vi, diubah dari pilihan Default menjadi Dialog.



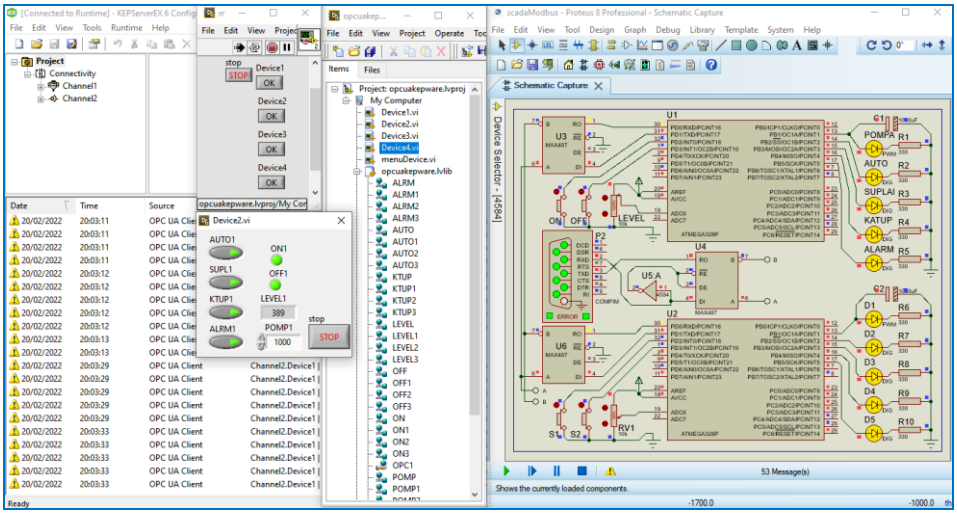
Gambar 4.143 Ubah Window Appearance dari Default menjadi Dialog

66. Berikutnya, jalankan rangkaian Proteus, OPC Quick Client dan program menuDevice LabVIEW. Tekan tombol Device1, maka seharusnya Front Panel Device1.vi muncul. Tekan tombol dan geser Potensio di Proteus, dan amati Indicator ON, OFF dan LEVEL. Tekan tombol AUTO, SUPL, KTUP, ALRM, dan atur nilai POMP, dan lihat perubahan pada rangkaian Proteus.



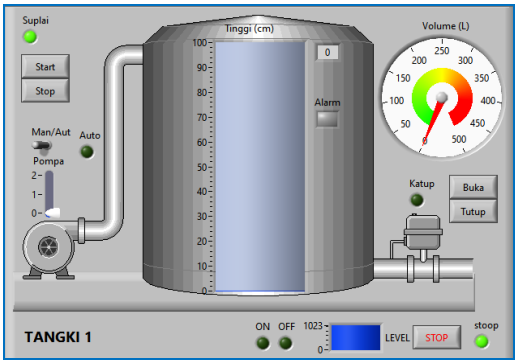
Gambar 4.144 Amati perubahan rangkaian U1 setiap kali pengaturan nilai Control Device1.vi, dan amati Indicator Device1.vi setiap kali tombol dan potensio di rangkaian U1 diubah

67. Tekan tombol Stop untuk menutup Device1.vi. Ulangi langkah 66 di atas untuk program Device2.vi.

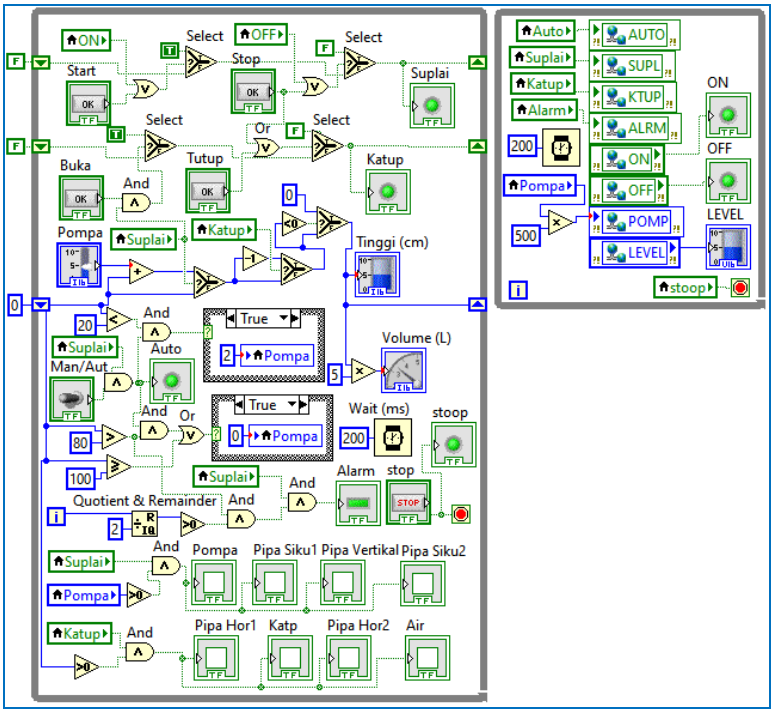


Gambar 4.145 Amati perubahan rangkaian U2 setiap kali pengaturan nilai Control Device2.vi, dan amati Indicator Device2.vi setiap kali tombol dan potensio di rangkaian U2 diubah

68. Tekan tombol Stop untuk menutup Device2.vi. Sebelum menjalankan program Device3.vi, pastikan rangkaian Proteus di komputerB dijalankan, begitu juga OPC Quick Client harus dijalankan. Ulangi langkah 66 di atas untuk program Device3.vi. Program Device3.vi ini akan dapat memonitor dan mengontrol rangkaian U1 Proteus di komputerB (UA Server).
69. Tekan tombol Stop Device3.vi. Lanjutkan dengan program Device4.vi.
70. Berikutnya, agar menjawab soal Tantangan Bab 4 ini, maka tambahkan tampilan Pengisian Tangki di setiap program Device1.vi hingga Device4.vi. Gambar 4.146 menunjukkan tampilan Front Panel Device1.vi. Gambar 4.147 menunjukkan Block Diagram Device1.vi.
71. Untuk mempercepat pembuatan program, setelah Front Panel dan Block Diagram program Device1.vi selesai dibuat, Save As program tersebut menjadi Device2.vi, Device3.vi dan Device4.vi. Berhubung variable I/O Server di setiap program berbeda, ubah variable tersebut dengan cara meng-klik variable tersebut, pilih Select Variable, pilih My Computer, pilih opcuakepware.lvlib, pilih satu dari 32 nama variable yang muncul.



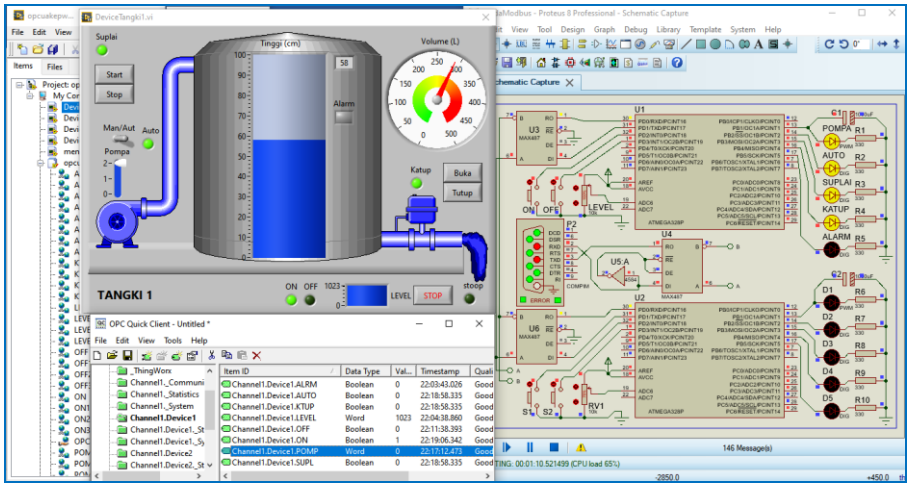
Gambar 4.146 Front Panel Device1.vi (lihat kembali program Bab 2)



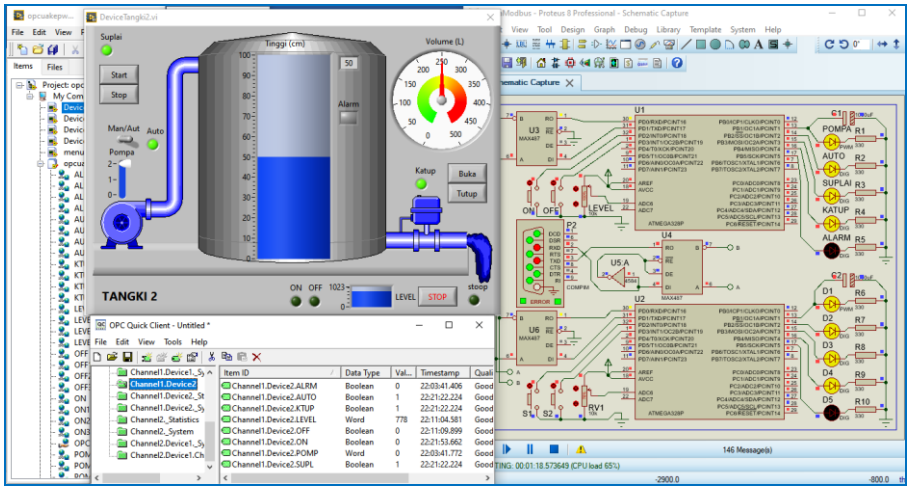
Gambar 4.147 Block Diagram Device1.vi (lihat kembali program Bab 2)

72. Setelah program Device1.vi hingga Device4.vi selesai dibuat dengan tambahan tampilan Pengisian Tangki, maka langkah berikutnya adalah, jalankan rangkaian Proteus, dan OPC Quick Client, baik di komputerA maupun di komputerB. Kemudian jalankan program menuDevice.

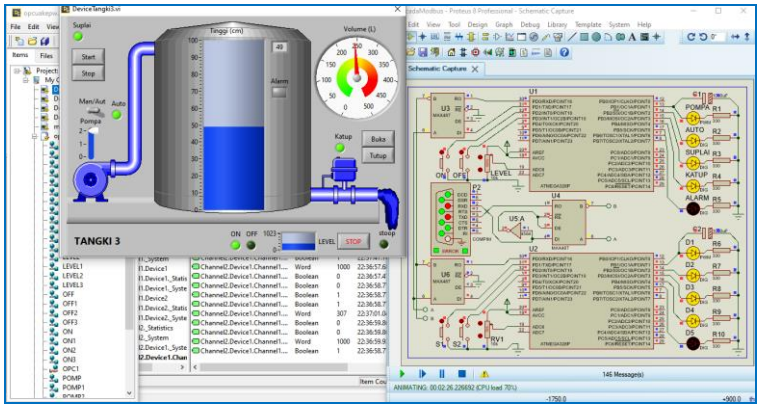
73. Tekan tombol Device1 di menuDevice, maka muncul Front Panel Device1. Lakukan penekanan tombol ON di rangkaian U1 Proteus hingga indikator Suplai di Device1 dan LED SUPL menyala. Geser Toggle Switch Man/Aut ke kanan, seharusnya LED AUTO menyala. Tekan tombol Buka di Device1, seharusnya LED KTUP menyala. Naik-turunkan Potensio di rangkaian U1, seharusnya indikator LEVEL mengikuti posisi Potensio. Tekan tombol Stop.
74. Ulangi langkah no. 73 di atas untuk Device2.vi, Device3.vi dan Device4.vi.



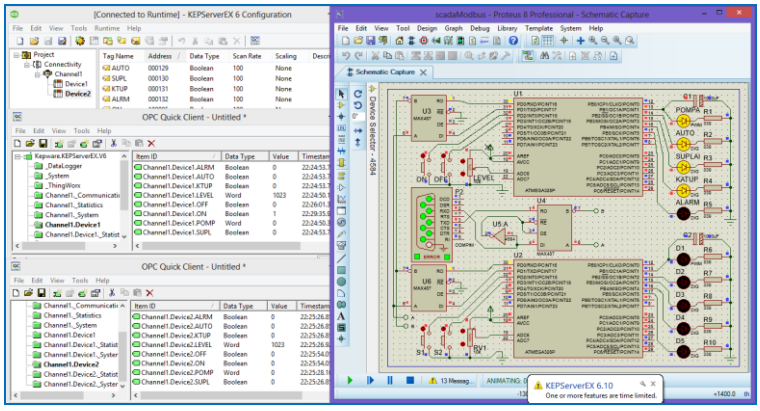
Gambar 4.148 Device1 dapat memonitor dan mengontrol rangkaian U1 Proteus komputerA



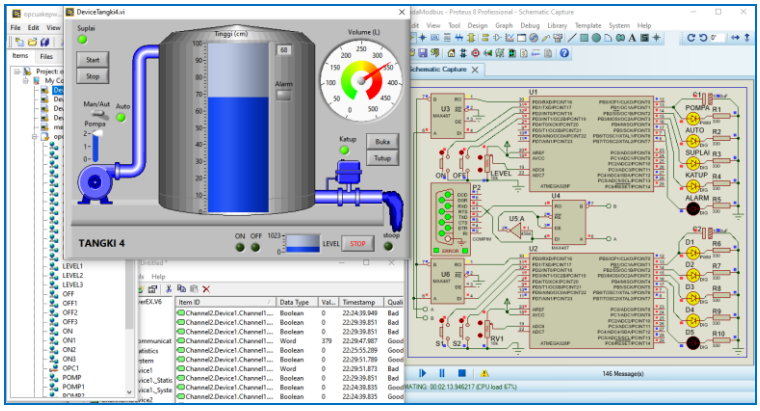
Gambar 4.149 Device2 dapat memonitor dan mengontrol rangkaian U2 Proteus komputerA



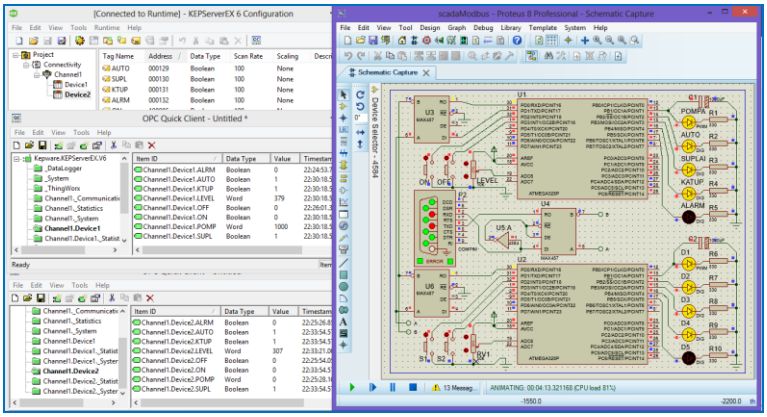
Gambar 4.150 Device3 dapat memonitor dan mengontrol rangkaian U1 Proteus di komputerB



Gambar 4.151 Tampilan rangkaian U1 Proteus di komputerB hasil pengaturan Device3



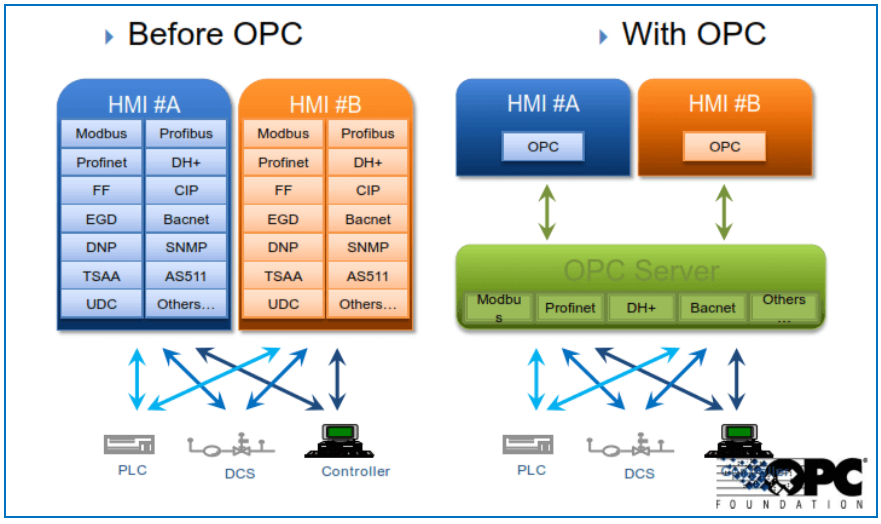
Gambar 4.152 Device4 dapat memonitor dan mengontrol rangkaian U2 Proteus di komputerB



Gambar 4.153 Tampilan rangkaian U2 Proteus di komputerB hasil pengaturan Device4

75. Simpan program dan project LabVIEW. Sampai di sini, pembuatan program LabVIEW OPC UA dengan KEPServerEX Kepware selesai.

4.5 Soal Latihan



Gambar 4.154 Keuntungan OPC (Sumber gambar: theautomization.com/opc-ua-vs-da)

1. Lihat Gambar 4.154 di atas. Dari gambar itu, apa saja keuntungan menggunakan OPC, dalam kaitannya dengan pembuatan program HMI? (HMI = *Human Machine Interface* dalam buku ini menggunakan program LabVIEW).

2. Apa itu OPC Server, dan apa fungsinya?
3. Apa itu OPC Client, dan apa fungsinya?
4. Apa itu OPC Quick Client dan apa fungsinya?
5. Apa itu OPC UA? Apa fungsi dan kelebihannya dibanding OPC Classic (DA)?
6. Buatlah sebuah OPC dengan NI OPC Servers, yang memiliki sebuah Channel dengan protokol Modbus RTU, dengan sebuah Device dan 4 buah Tag. Tag terdiri dari Coil, Input Diskrit, Input Register dan Holding Register, yang memiliki alamat berturut-turut 000001, 100001, 300001, dan 400001. Gunakan ModRSSim2 sebagai Slave dan OPC Quick Client sebagai Interface Master. Lakukan pengubahan nilai semua Tag di ModRSSim2 dan amati nilai Tag di OPC Quick Client. Lakukan pengubahan nilai Tag Coil dan Holding Register di OPC Quick Client, dan amati perubahan nilai di ModRSSim2.
7. Ulangi Soal Latihan no. 6 di atas, namun ganti ModRSSim2 dengan rangkaian Arduino di Proteus seperti Gambar 3.17. Gunakan program Outseal seperti Gambar 3.21. Buat agar OPC Quick Client dapat membaca penekanan tombol dan pengubahan nilai potensio pada rangkaian, dan dapat mengontrol nyala hidup LED (D1) dan mengatur intensitas cahaya LED PWM (D2).
8. Ulangi Soal Latihan no. 7 di atas, dan tambahkan program LabVIEW (sebagai HMI) yang dapat membaca penekanan tombol dan pengubahan nilai potensio pada rangkaian, dan dapat mengontrol nyala hidup LED (D1) dan mengatur intensitas cahaya LED PWM (D2).
9. Ulangi Soal Latihan no. 8 di atas, namun dengan rangkaian 2 buah Arduino, seperti Gambar 4.51. Diinginkan program LabVIEW dapat membaca penekanan tombol dan pengubahan nilai potensio di kedua rangkaian Arduino (U1 dan U2), serta dapat mengontrol nyala hidup LED dan mengatur intensitas cahaya LED PWM di kedua rangkaian Arduino (U1 dan U2).
10. Ulangi Soal Latihan no. 9 di atas, namun dengan rangkaian 2 buah Arduino di 2 buah komputer. Diinginkan program LabVIEW dapat membaca dan mengubah input output di kedua rangkaian Arduino yang ada di 2 buah komputer.

4.6 Refleksi

1. Menurut Anda, dari isi yang diuraikan, apakah **Target Materi** dari Bab 4 buku ini tercapai? Jika belum tercapai, apakah ada kesulitan dalam memahami materi yang berkaitan dengan Target Materi tersebut? Apakah Anda memiliki saran dan masukan untuk memperbaiki materi tersebut?
2. Apakah **Tantangan** dalam Bab 4 buku ini dapat Anda selesaikan? Jika belum, kesulitan apa yang membuat Anda tidak bisa menyelesaikannya? Apakah Anda mencoba alternatif lain untuk menemukan sendiri cara penyelesaiannya (mencari di Google misalnya)?
3. Apakah ada **Manfaat** yang Anda dapatkan setelah mempelajari Bab 4 dari buku ini? Apakah ada yang ingin Anda pelajari lebih lanjut? Apakah ada **Ide** yang menarik yang ingin Anda kembangkan?

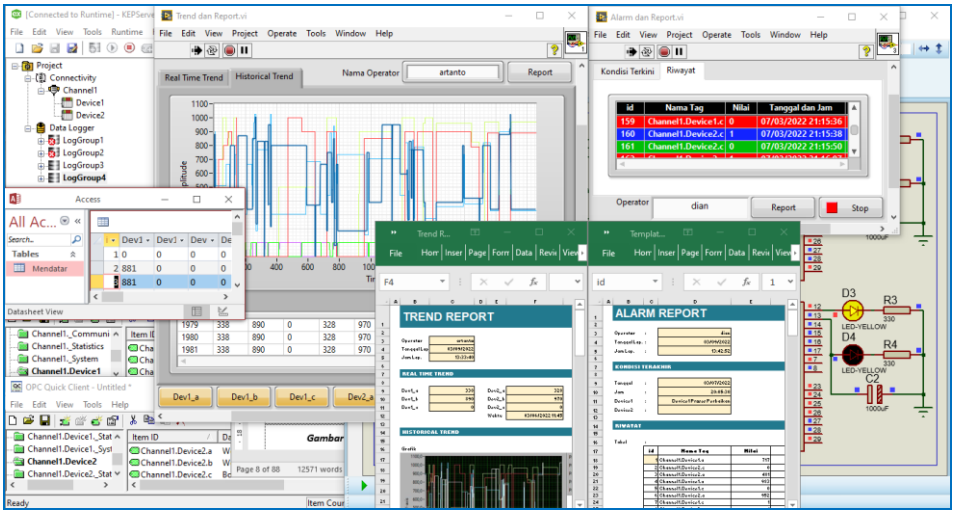
BAB 5
DATABASE

Target Materi:

- Menggunakan Data Logger KEPServerEX untuk membuat Database.
- Menggunakan Database dan LabVIEW untuk membuat Alarm dan Trend.
- Menggunakan Database, Excel dan LabVIEW untuk membuat Report.

Tantangan:

- Buat program HMI dengan LabVIEW yang dapat menampilkan data Alarm, dan grafik Trend secara Real Time dan Historical, serta Report keduanya.



Gambar 5.1 Program HMI dengan fitur Alarm, Trend dan Report di Excel

5.1 Persiapan Perangkat yang Diperlukan

Bab ini memerlukan tambahan perangkat lunak/software sebagai berikut:

1. Software Add-On LabVIEW Database Connectivity Toolkit
2. Software Office (Microsoft Access dan Microsoft Excel)
3. Software Add-On LabVIEW Report Generation Toolkit

Software Add-On LabVIEW Database Connectivity Toolkit ini berisi fungsi-fungsi untuk pembuatan database dan penerapan operasi database secara umum. Pembaca dapat mengunduh software Add-On ini di halaman blog penulis.

Software kedua adalah software yang umumnya sudah terinstal apabila pembaca menggunakan sistem operasi Windows, yaitu Microsoft Access dan Microsoft Excel. Di Bab ini, Microsoft Access digunakan untuk pembuatan database, sedangkan Microsoft Excel digunakan untuk pembuatan laporan (*reporting*).

Software ketiga adalah software Add-On yang sudah terinstal di LabVIEW Profesional. Tanda bahwa Add-On ini sudah terinstal, di Palet Functions, di kategori Programming, ada Report Generation. Apabila pembaca tidak menemukan, pembaca dapat mengunduh software Add-On ini di blog penulis.

5.2 Database dengan Data Logger KEPServerEX

Sebelum membahas mengenai pembuatan database, akan lebih baik apabila me-review kembali materi yang telah diuraikan di Bab sebelumnya:

- Di Bab 1, telah diuraikan mengenai pembuatan Tampilan SCADA dan animasinya menggunakan LabVIEW.
- Di Bab 2, telah diuraikan mengenai pembuatan program LabVIEW untuk komunikasi serial (ASCII) sederhana antara 2 perangkat.
- Di Bab 3, telah diuraikan mengenai pembuatan program LabVIEW untuk menghubungkan lebih dari 2 perangkat (1 Master dan 2 Slave), menggunakan protokol Modbus RTU dan sambungan RS-485.
- Di Bab 4, telah diuraikan mengenai penggunaan OPC Server (NI OPC Server dan KEPServerEX) yang memudahkan penyambungan berbagai perangkat, dan juga pembuatan program HMI (LabVIEW) menjadi mudah.

Membaca review di atas, secara garis besar materi di keempat Bab sebelumnya dapat diringkas menjadi 2 hal; hal pertama adalah Pembuatan Tampilan (Bab 1), dan hal kedua adalah Penyambungan Perangkat (Bab 2 – 4). Sementara itu, terkait dengan judul dari buku ini, yang mengambil topik mengenai pembuatan SCADA, secara umum fitur SCADA harus memiliki minimal 7 fitur berikut ini:

a. *Proces Visualization* (sama dengan Pembuatan Tampilan)
b. *IO Server* (sama dengan Penyambungan Perangkat)
c. Database
d. Alarm
e. *Trending* (*Real Time and Historical*)
f. Reporting
g. *Security*

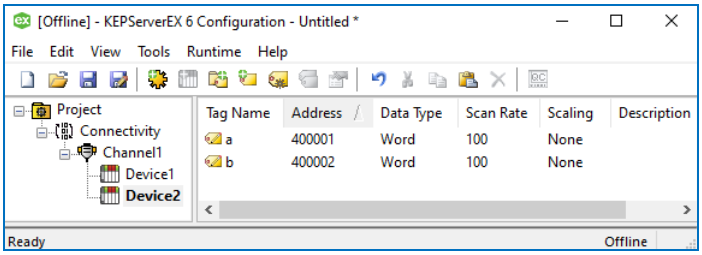
Agar ketujuh fitur SCADA di atas dapat diimplementasikan dalam buku ini, maka Bab 5 ini akan menguraikan pembuatan keempat fitur berikutnya, yaitu *Database*, *Alarm*, *Trending,* dan *Reporting*. Berikutnya di Bab 6 akan diuraikan mengenai Teknologi IoT Gateway yang mengembangkan ruang lingkup SCADA menjadi lebih luas, sehingga terkait erat dengan fitur terakhir, yaitu *Security*.

Apa itu database dan apa fungsinya? Secara umum, database adalah sarana penyimpanan data yang menyimpan data secara sistematik, dan dapat diakses secara mudah. Database berfungsi seperti layaknya sebuah perpustakaan yang menyimpan banyak buku – yang ditata dan diurutkan berdasarkan kategori, judul, nama pengarang, dll, sehingga memudahkan pencarian buku.

Sekalipun banyak jenis aplikasi database tersedia, seperti MySQL, SQL Server, Oracle, dll., namun di sini penulis memilih Microsoft Access. Salah satu alasannya adalah, supaya tidak repot dengan masalah instalasi. Microsoft Access ini biasanya terinstal sekaligus ketika menginstal Microsoft Office. Di Sub Bab 5.2 ini akan diperlihatkan bagaimana melakukan penyimpanan data Tag dari perangkat yang terhubung dengan OPC Server ke dalam database Microsoft Access, dengan bantuan layanan Data Logger dari KEPServerEX. Berikut ini langkah-langkahnya:

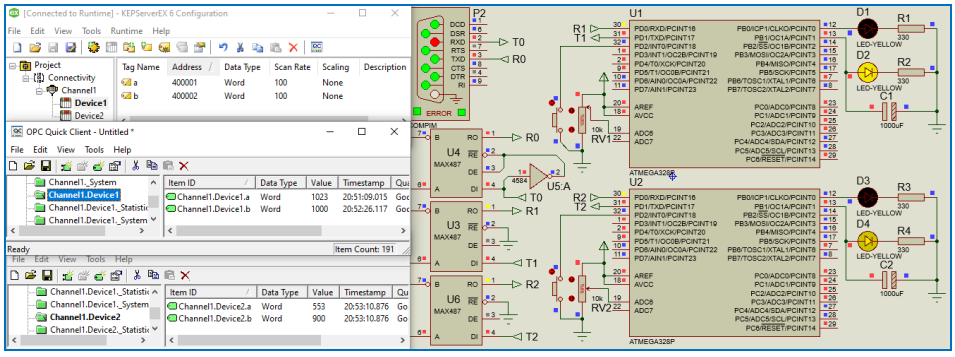
1. Buka KEPServerEX Configuration, buat sebuah Channel, dengan protokol Modbus RTU Serial, dengan 2 buah Device (ID = 1 dan ID = 2).

2. Masing-masing Device memiliki 2 buah Tag, dengan nama Tag a dan b, yang secara berurut memiliki alamat 400001 dan 400002.



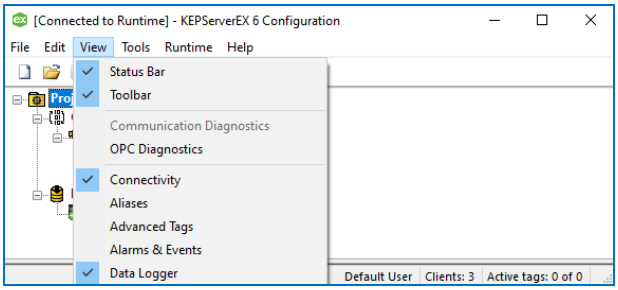
Gambar 5.2 Sebuah Channel dengan 2 buah Device, dengan 2 buah Tag di setiap Device

3. Sebagai perangkat Modbus RTU Slave untuk Device1 dan Device2, gunakan rangkaian 2 buah Arduino di Proteus seperti Gambar 4.51.
4. Gunakan program Outseal Gambar 4.52 untuk kedua Arduino. Buat alamat Modbus = 1 untuk U1, dan alamat Modbus = 2 untuk U2.
5. Jalankan OPC Quick Client untuk Device1 dan Device2, dan juga Proteus. Perhatikan bahwa Tag a (400001) di Device1 dan Device2 dapat membaca nilai potensio di U1 dan U2 rangkaian Proteus, dan Tag b (400002) di Device1 dan Device2 dapat mengatur nyala LED PWM di U1 dan di U2.



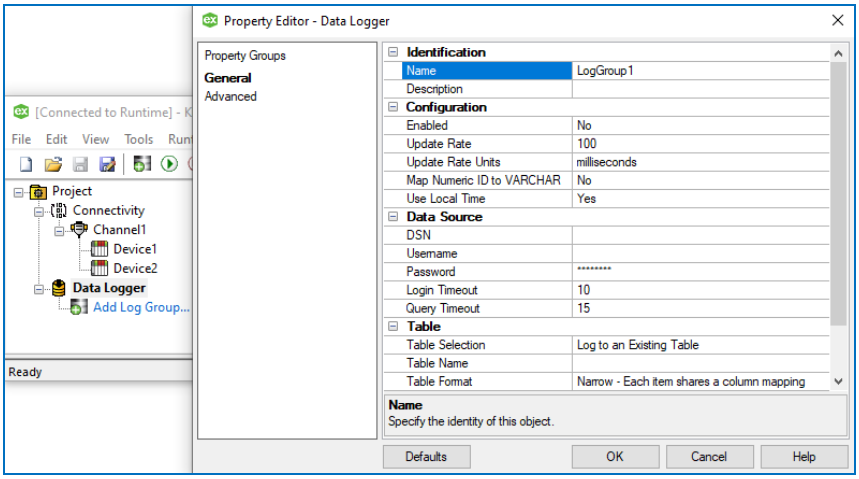
Gambar 5.3 Tag a membaca nilai potensio (0-1023), Tag b mengatur nyala LED PWM (0-1000)

6. Diinginkan nilai Tag a dan Tag b di kedua Device dapat disimpan secara berkala ke dalam Microsoft Access. Penyimpanan data ini dapat dilakukan dengan mudah dengan bantuan Data Logger. Buka menu View di KEPServerEX Configuration, dan centang pada Data Logger.



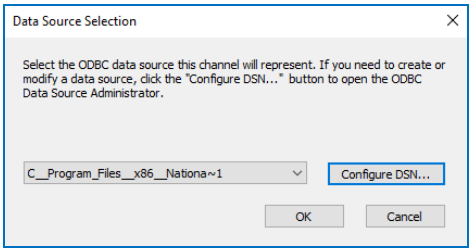
Gambar 5.4 Centang Data Logger di menu View agar terlihat di kolom kiri Configuration

7. Setelah Data Logger muncul di kolom kiri, klik pada Add Log Group.



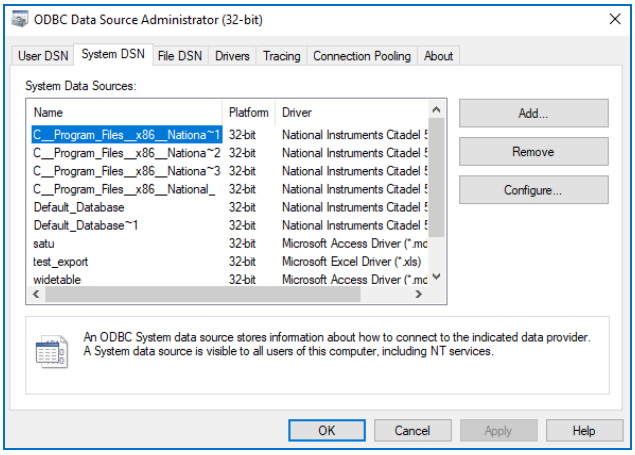
Gambar 5.5 Klik pada Add Log Group, maka muncul Property Editor Data Logger

8. Ada 4 item yang harus diisi, yaitu DSN, Table Selection, Table Name dan Table Format. Mulai dari DSN, klik pada kolom isian DSN, tekan tombol yang muncul di kanan, maka terbuka kotak Data Source Selection.



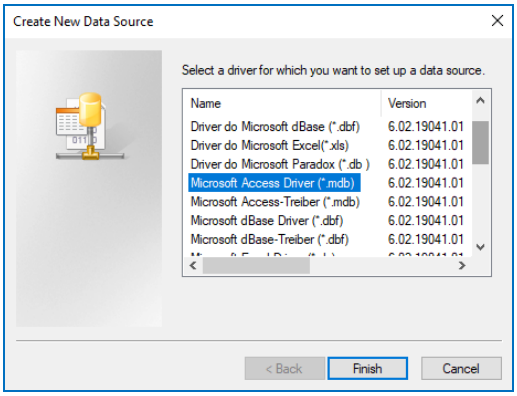
Gambar 5.6 Kotak Data Source Selection, klik tombol Configure DSN

9. Di kotak Data Source Selection, klik pada tombol Configure DSN, maka muncul kotak ODBC Data Source Administrator (32-bit). Pilih Tab System DSN (User DSN untuk akses database ke 1 pengguna, sedangkan System DSN untuk akses database ke banyak pengguna). Klik tombol Add.



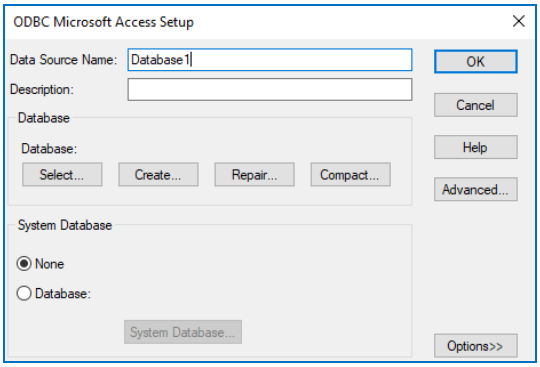
Gambar 5.7 Di kotak ODBC Data Source Administrator, pilih Tab System DSN, klik tombol Add

10. Muncul kotak Create New Data Source. Pilih Microsoft Access Driver (*.mdb). Klik Finish.



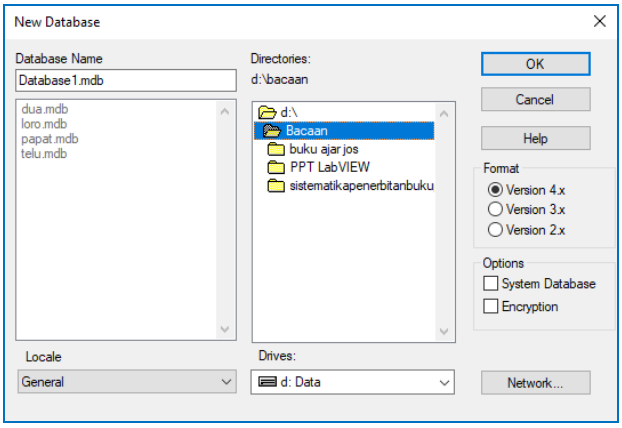
Gambar 5.8 Di kotak Create New Data Source, pilih Microsoft Access Driver (*.mdb), klik Finish

11. Muncul kotak ODBC Microsoft Access Setup. Pada kolom Data Source Name, isi kolom dengan sebuah nama (nama ini akan muncul di Property Editor – Data Logger). Sebagai contoh, isi dengan nama Database1.



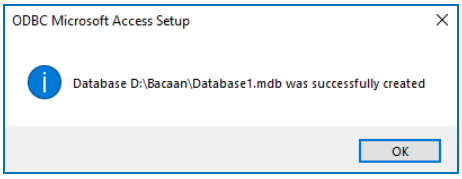
Gambar 5.9 Isi kolom Data Source Name, kemudian klik tombol Create

12. Setelah kolom Data Source Name diisi, berikutnya klik tombol Create, maka muncul kotak New Database. Beri nama file database Microsoft Access yang akan dibuat. Untuk memudahkan, beri nama yang sama dengan nama DSN. Tempatkan file pada lokasi yang mudah ditemukan.



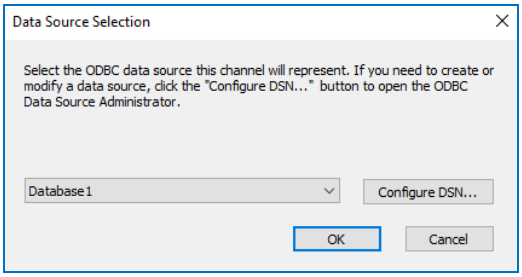
Gambar 5.10 Beri nama file database (*.mdb), tempatkan di lokasi yang tepat, klik OK

13. Klik OK, maka muncul pesan bahwa file database sudah selesai dibuat.



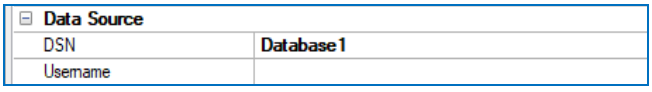
Gambar 5.11 Muncul pesan bahwa file database Microsoft Access sudah selesai dibuat.

14. Klik OK pada kotak ODBC Microsoft Access Setup.
15. Pada kotak ODBC Data Source Administrator, pilih nama DSN yang telah dibuat. Klik OK, maka nama DSN yang telah dibuat muncul di kolom pilihan di kotak Data Source Selection, seperti gambar berikut ini.



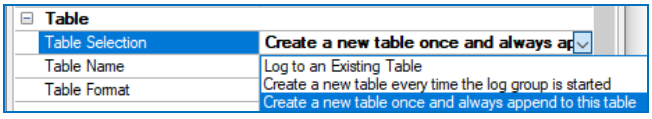
Gambar 5.12 Nama DSN muncul pada kolom pilihan di Data Source Selection, klik OK

16. Klik OK, maka nama DSN yang telah dibuat, muncul di kolom DSN di kotak Property Editor Data Logger.



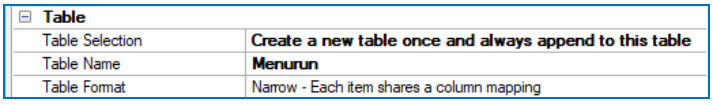
Gambar 5.13 Di kolom DSN di Property Editor – Data Logger, muncul nama DSN yang dibuat

17. Berikutnya di kolom Table Selection Property Editor, pilih pilihan ketiga, yaitu ***Create a new table once and always append to this table***.



Gambar 5.14 Di Table Selection, pilih Create a new table once and always append to this table

18. Di kolom Table Name, beri nama tabel yang baru. Dalam contoh di sini, diisi dengan nama **Menurun**. Mengapa dinamai **Menurun**? Karena ini terkait dengan pilihan Table Format, yang diisi dengan pilihan **Narrow**.



Gambar 5.15 Di Table Name, isi dengan nama tabel yang baru, dan Table Format = Narrow

Catatan: pilihan Narrow pada Table Format akan membuat semua Tag yang disimpan di database akan ditampilkan menurun satu persatu. Sebagai contoh di sini, ada 4 buah Tag, yaitu Tag a Device1, Tab b Device1, Tag a Device2, dan Tag b Device2. Maka bila Table Format dipilih Narrow, keempat Tag tersebut akan disimpan dan ditampilkan satu persatu, mulai dari Tag a Device1 di baris pertama, Tag b Device1 di baris kedua, dan Tag seterusnya di baris berikutnya.

Sebaliknya bila pilihan Wide pada Table Format dipilih, maka keempat Tag tersebut akan disimpan dan ditampilkan sekaligus dalam 1 baris mendatar, baru pada penyimpanan data berikutnya ditampilkan di baris berikutnya.

id	Item	Value
1	Tag a Device1	500
2	Tag b Device1	600
3	Tag a Device2	700
4	Tag b Device2	800
5	Tag a Device1	501

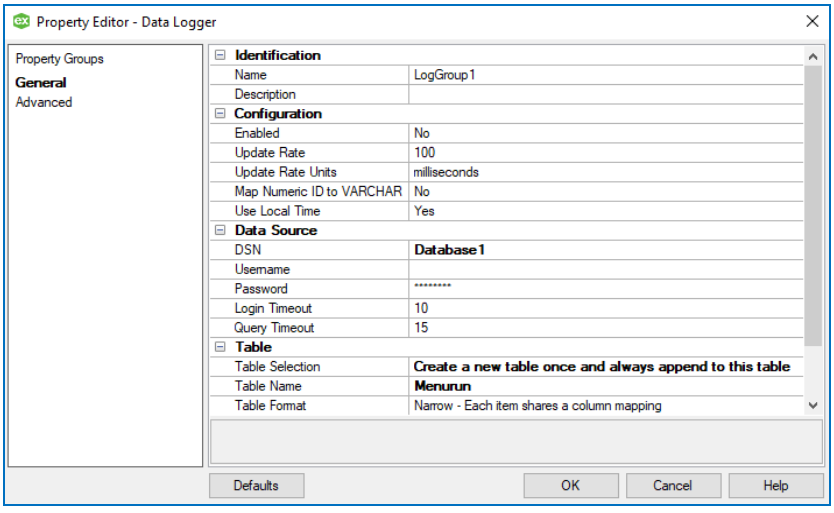
(a)

id	Tag a D1	Tag b D1	Tag a D2	Tag b D2
1	500	600	700	800
2	501	601	701	801
3	502	602	702	802
4	503	603	703	803
5	504	604	704	804

(b)

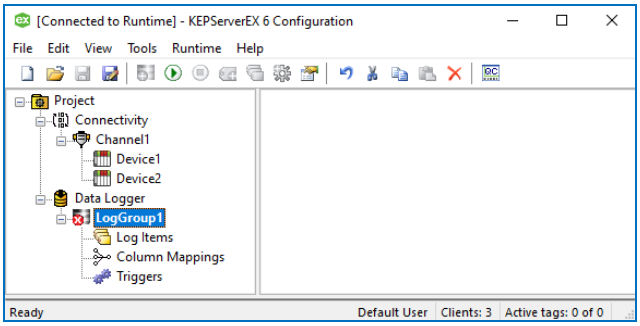
Gambar 5.16 (a) Table Format = Narrow, (b) Table Format = Wide

19. Pastikan 4 item di Property Editor Data Logger terisi, yaitu DSN, Table Selection, Table Name dan Table Format, seperti gambar berikut ini.



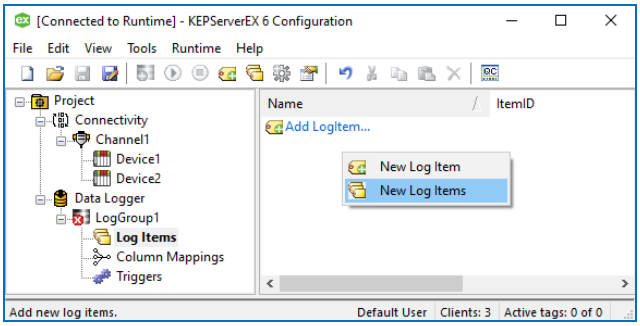
Gambar 5.17 DSN, Tabel Selection, Table Name dan Table Format di Property Editor telah diisi

20. Klik OK, maka muncul nama LogGroup1 di bawah Data Logger.



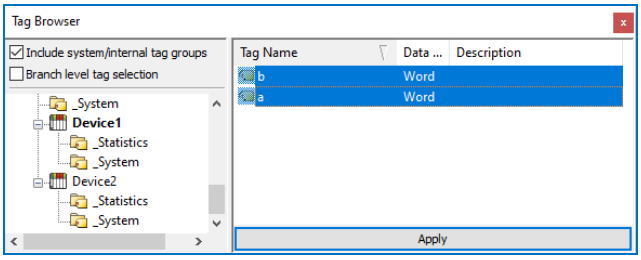
Gambar 5.18 Muncul LogGroup1 di bawah Data Logger

21. Diinginkan nilai Tag a dan Tag b di Device1 dan Device2 dapat disimpan di database Microsoft Access. Pilih Log Items, kemudian klik kanan kolom di kanan, pilih New Log Items.



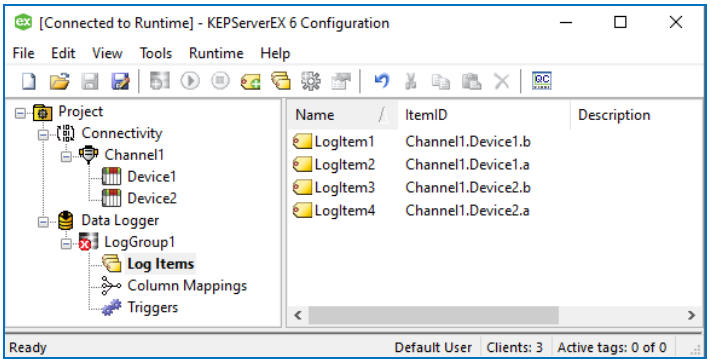
Gambar 5.19 Klik Log Items, klik kanan kolom di kanan, pilih New Log Items

22. Muncul kotak Tag Browser. Klik pada Device1. Muncul Tag a dan b di kolom kanan. Pilih kedua Tag, dan tekan tombol Apply.



Gambar 5.20 Pilih Device1, pilih kedua Tag di kolom kanan, tekan tombol Apply

23. Ulangi langkah 20-22 untuk Tag a dan b di Device2. Pilih Log Items, klik kanan di kolom kanan, pilih New Log Items. Di Tag Browser, pilih Device2, pilih Tag a dan b di kolom kanan, tekan tombol Apply, maka seharusnya di Log Items ada 4 LogItem, sesuai jumlah Tag yang nilainya akan disimpan.



Gambar 5.21 Di Log Items berisi 4 buah Tag, yang nilainya akan disimpan ke database

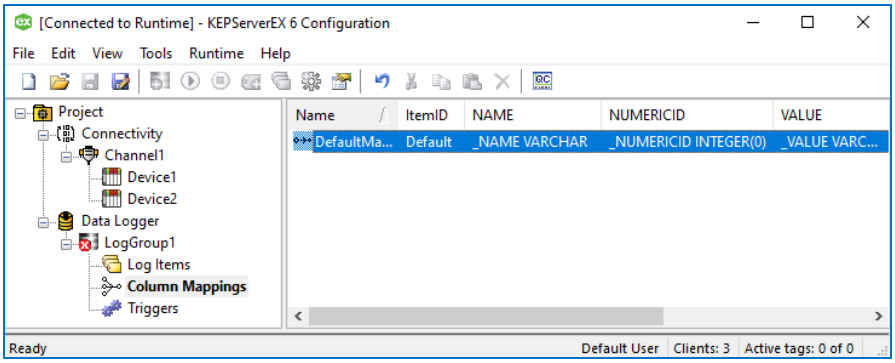
Catatan: Setiap LogGroup memiliki 3 objek pengaturan, yaitu Log Items, Column Mappings dan Triggers, dengan penjelasan sebagai berikut:

a. Log Items digunakan untuk menampung Tag yang nilainya akan disimpan.

b. Column Mappings digunakan untuk mengatur tampilan data di database. Secara default, setiap Tag memiliki 5 kolom data, yaitu nama data, nomor ID, kualitas data, waktu (*timestamp*), dan nilai data. Kelima data ini dapat diatur, mana data yang tidak ditampilkan, dengan cara menghapus nama kolomnya (*Database Field Name*) di kotak Property Editor Column Mapping.

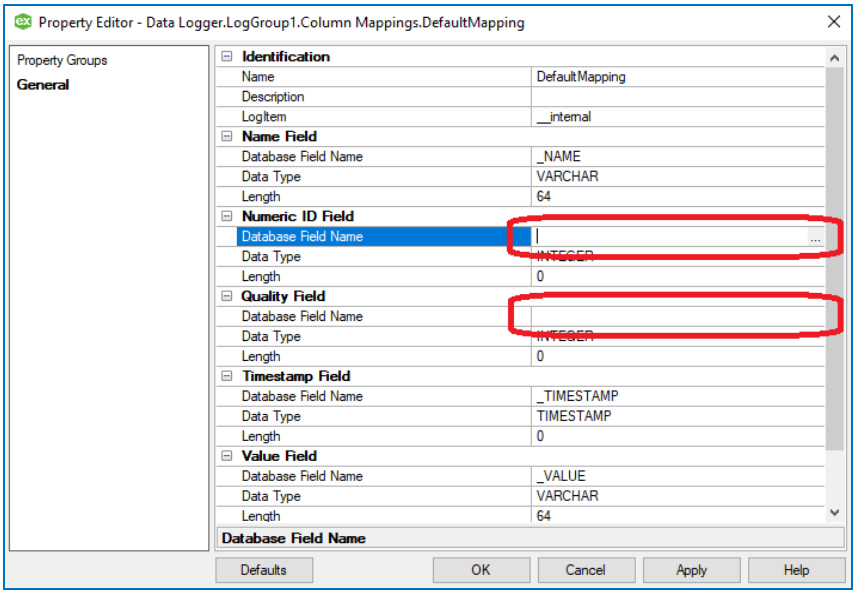
c. Triggers digunakan untuk mengatur kapan data akan disimpan. Ada 3 tipe Trigger, yaitu Always Triggered, Time Based dan Condition Based. Always Triggered adalah pilihan default, yang berarti penyimpanan akan selalu dilakukan selama LogGroup di-Enabled dan OPC Server aktif. Sedang Time Based berarti penyimpanan hanya dilakukan selama waktu tertentu. Condition Based berarti penyimpanan hanya dilakukan selama syarat kondisi dipenuhi. Di samping pilihan Trigger, kondisi penyimpanan juga dapat dipilih, apakah akan disimpan dengan waktu interval tertentu, atau hanya disimpan ketika ada data Tag yang berubah, atau gabungan dari kedua kondisi.

24. Setelah LogItems terisi dengan Tag yang akan disimpan ke database, berikutnya ke pengaturan tampilan database. Klik Column Mappings, maka muncul sebaris nama-nama kolom data di kolom kanan.



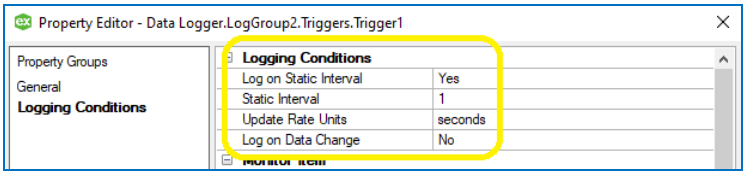
Gambar 5.22 Klik pada Column Mappings, maka muncul nama-nama kolom data

25. Klik 2 kali pada Default Mapping di kolom kanan, maka muncul Property Editor Column Mappings. Agar tidak banyak kolom data di database, maka hapus isi *Database Field Name* untuk *Numeric ID Field* dan *Quality Field*, yang ditunjukkan dengan kotak merah kosong pada gambar berikut ini.



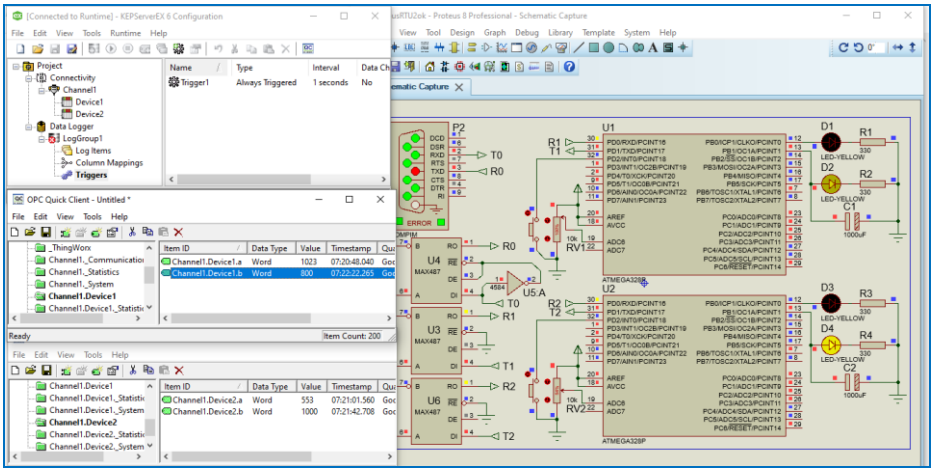
Gambar 5.23 Hapus nama kolom data untuk nomor ID dan kualitas data, klik OK

26. Setelah pengaturan tampilan data di Column Mappings, berikutnya adalah pengaturan Triggers. Klik pada Triggers, kemudian klik 2 kali pada Trigger1 di kolom kanan, maka muncul Property Editor Trigger. Diinginkan penyimpanan dilakukan setiap detik sekali. Untuk itu di Property Editor, klik *Logging Conditions*, ubah *Static Interval* dari 500 menjadi 1, dan *Update Rate Units* dari *miliseconds* menjadi *seconds*. Juga ubah isi *Log on Data Change* dari *Yes* menjadi *No*. Mengapa dibuat No? Agar penyimpanan hanya dilakukan setiap detik sekali (*Log on Static Interval = Yes*), dan tidak ada penyimpanan di tengah interval, sekalipun ada perubahan data (*Data Change*) di waktu tersebut.



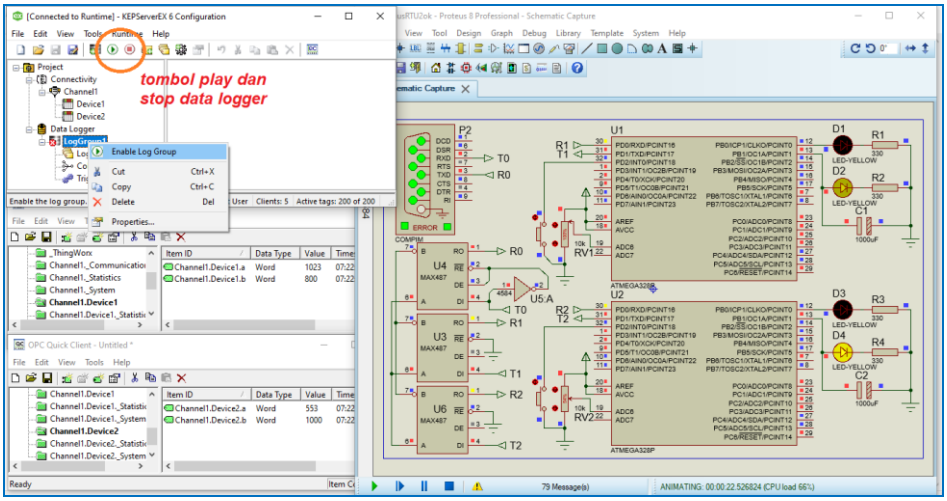
Gambar 5.24 Atur Logging Conditions Trigger agar bisa menyimpan setiap detik sekali

27. Setelah ketiga objek pengaturan selesai, jalankan OPC Quick Client dan rangkaian di Proteus. Pastikan Tag a dan b di kedua Device dapat ditampilkan nilainya di OPC Quick Client. Apabila nilai Tag di kedua Device tidak muncul di OPC Quick Client, maka reset Runtime OPC Server.



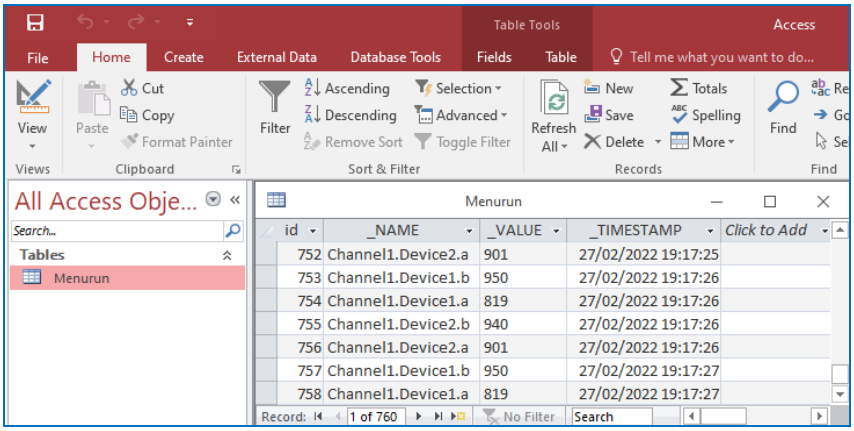
Gambar 5.25 Jalankan OPC Quick Client dan rangkaian Proteus, pastikan nilai Tag muncul

28. Setelah nilai kedua Tag di kedua Device dapat terlihat di OPC Quick Client, jalankan penyimpanan data ke database (data logger) dengan cara meng-klik kanan LogGroup1, pilih Enable Log Group. Cara lain untuk menjalankan data logger ini dengan memilih LogGroup1, dan tekan tombol “play” warna hijau di Toolbar KEPServerEX.



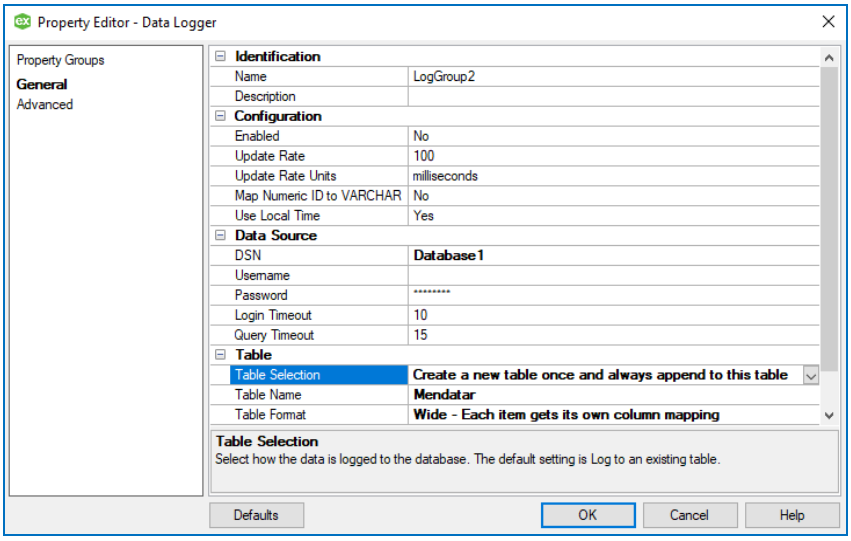
Gambar 5.26 Jalankan data logger dengan klik kanan LogGroup1, pilih Enable Log Group

29. Lakukan pengubahan nilai Tag a dengan menggeser potensio di U1 dan U2 di Proteus. Untuk Tag b, lakukan pengubahan nilai dengan meng-klik kanan nilai Tag tersebut di OPC Quick Client, dan pilih Synchronous atau Aysnchronous Write, dan isi kolom Write Value dengan nilai 0-1000.
30. Setelah beberapa saat, hentikan data logger dengan menekan tombol Stop di Toolbar KEPServerEX, atau cara lain dengan meng-klik kanan LogGroup1, pilih Disable Log Group.
31. Berikutnya buka file database tempat penyimpanan data. Gambar 5.27 menunjukkan tampilan hasil penyimpanan data di file Database1.mdb, di dalam tabel Menurun. Perhatikan bahwa ada 4 kolom data. Kolom data pertama id (index), dibangkitkan secara otomatis oleh Microsoft Access, sedangkan kolom _NAME, _VALUE dan _TIMESTAMP mengikuti pengaturan Column Mapping di Gambar 5.23. Perhatikan nilai keempat Tag ditampilkan secara berurutan menurun di kolom _NAME.



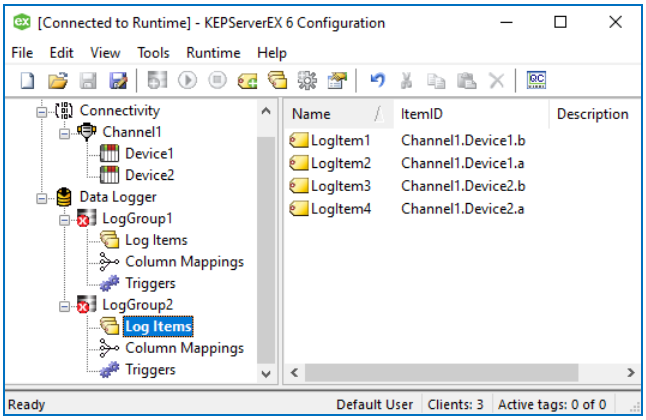
Gambar 5.27 Tampilan data di Database1.mdb, keempat Tag ditampilkan secara menurun

32. Diinginkan untuk membuat keeempat Tag ditampilkan secara mendatar dalam 1 baris. Untuk itu, buat lagi sebuah LogGroup, dengan meng-klik kanan Data Logger di KEPServerEX Configuration, pilih New Log Group, maka muncul Property Editor Data Logger LogGroup2.
33. Isi DSN = *Database1*, Table Selection = *Create a new table once and always append to this table*, Table Name = *Mendatar*, Table Format = *Wide*, seperti ditunjukkan pada gambar berikut ini.



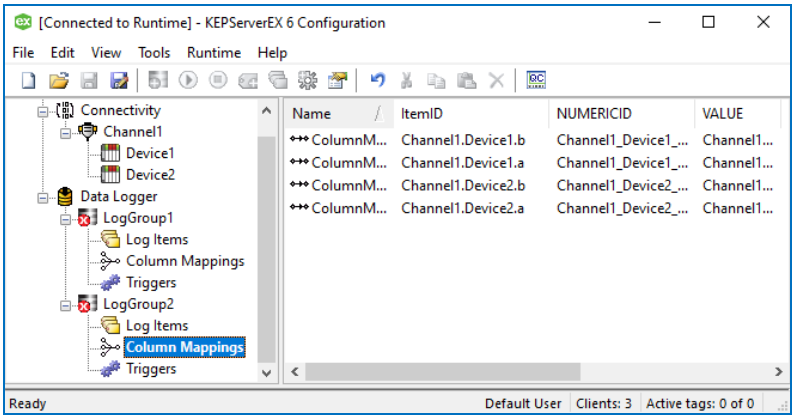
Gambar 5.28 Buat lagi LogGroup2, dengan Table Format = Wide

34. Klik OK, maka muncul nama LogGroup2 di bawah LogGroup1.
35. Isi Log Items LogGroup2 sama seperti isi Log Items LogGroup1.



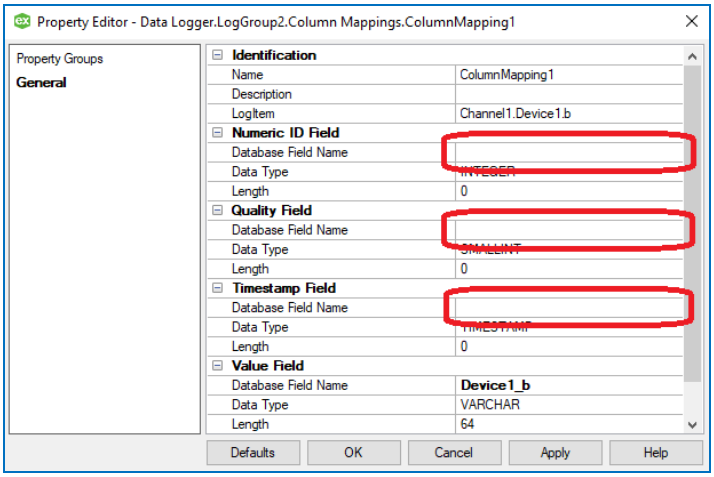
Gambar 5.29 Isi Log Items Log Group2 sama seperti isi Log Items Log Group1

36. Karena Table Format = Wide, maka di Column Mapping ada 4 buah nama Column Mapping, sesuai dengan jumlah Tag yang diisikan di Log Items.



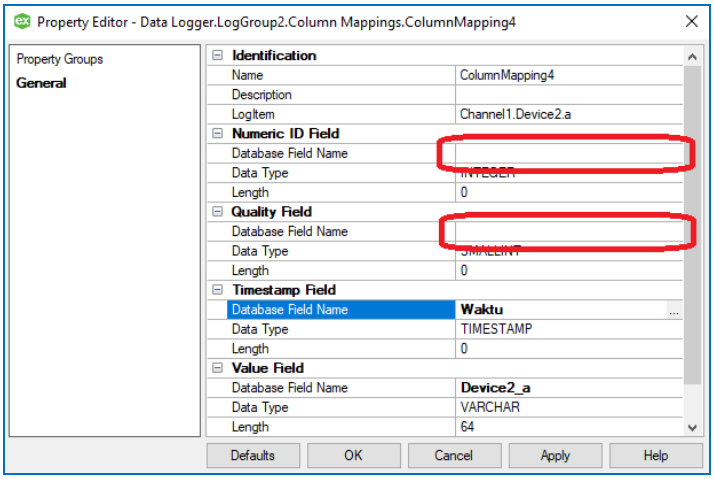
Gambar 5.30 Karena Table Format = Wide, muncul 4 Column Mapping sesuai jumlah Tag

37. Agar tidak terlalu banyak kolom (apabila semua kolom ditampilkan, maka akan ada 4 x 4 + 1 = 17 kolom), maka di Property Editor Column Mapping, hapus nama kolom Nomor ID, Kualitas, dan Timestamp, hingga tinggal kolom Nilai. Agar tidak terlalu panjang nama kolomnya, hilangkan nama Channel, hingga hanya nama Device dan Tag-nya saja, contoh Device1_b.



Gambar 5.31 Agar tidak banyak kolom, hapus semua nama kolom, kecuali kolom Nilai data

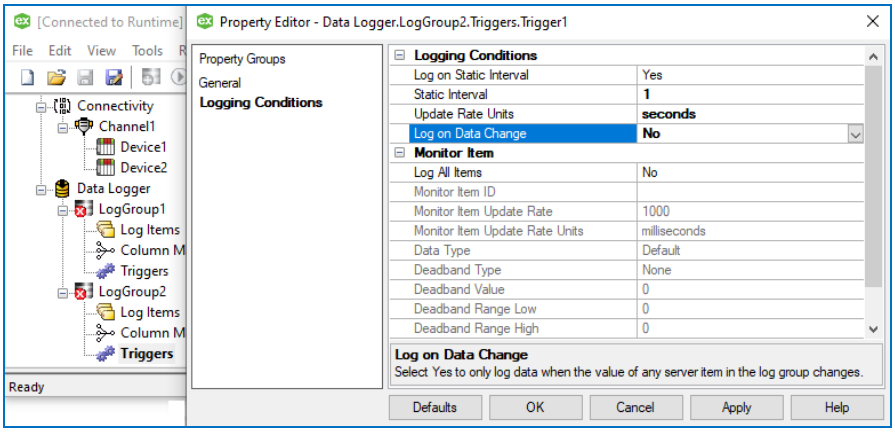
38. Buat pengaturan langkah no. 37 di atas untuk Column Mapping1 sampai Column Mapping3. Untuk Column Mapping4, buat agar kolom Timestamp dan Nilai tidak dihapus, sedang kolom Nomor ID dan Kualitas dihapus. Agar nama kolom tidak terlalu panjang, ubah nama Timestamp menjadi Waktu, dan nama kolom Nilai menjadi nama Device diikuti nama Tag.



Gambar 5.32 Khusus Column Mapping4, buat agar kolom Timestamp dan Nilai tidak dihapus

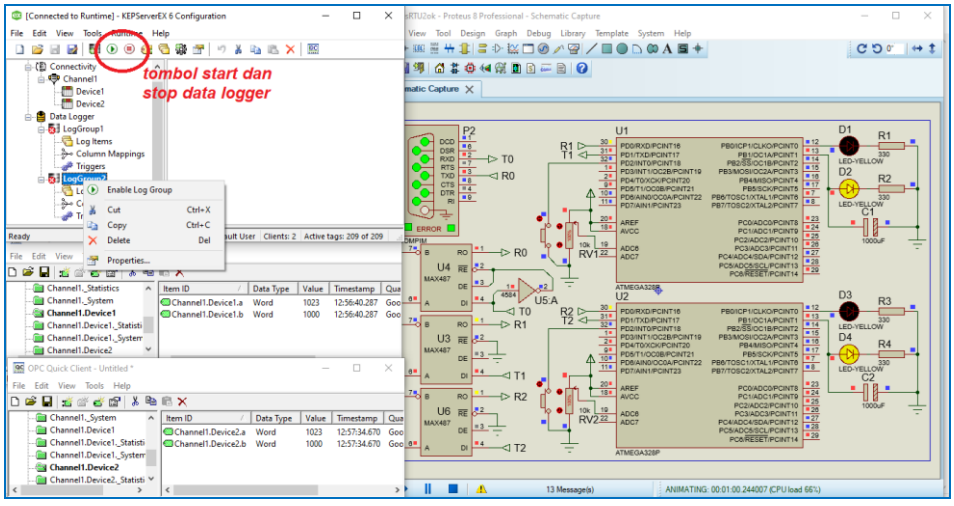
39. Klik OK. Berikutnya pilih Triggers. Di kolom kanan, klik 2 kali Trigger1 hingga muncul Property Editor Trigger1.

40. Di Logging Conditions, buat Log on Static Interval = Yes, Static Interval = 1, Update Rate Units = seconds, dan Log on Data Change = No. Klik OK.



Gambar 5.33 Atur Logging Conditions agar penyimpanan data dilakukan setiap detik sekali

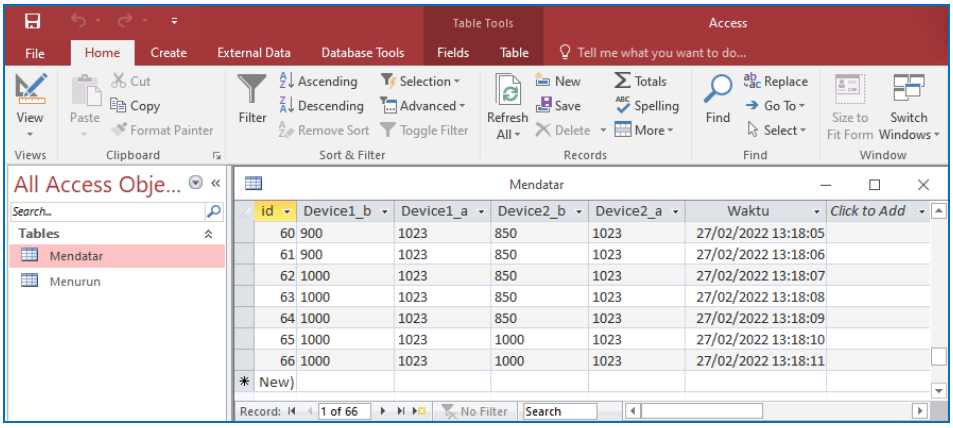
41. Jalankan OPC Quick Client dan rangkaian Modbus RTU Slave di Proteus. Setelah nilai Tag di kedua Device terlihat di OPC Quick Client, jalankan penyimpanan data dengan meng-klik kanan LogGroup2, dan pilih Enable Log Group atau dengan menekan tombol Start di Toolbar KEPServerEX.



Gambar 5.34 Jalankan OPC Quick Client, rangkaian Proteus, dan data logger

42. Lakukan pengubahan nilai pada Tag a dan b di kedua Device.

43. Setelah beberapa saat, hentikan data logger.
44. Berikutnya buka file database tempat penyimpanan data. Di Table Mendatar, terlihat 6 kolom data. Kolom data pertama adalah kolom id (index), yang dibangkitkan secara otomatis oleh Microsoft Access. Empat kolom berikutnya, merupakan nilai 4 buah Tag, yang ditampilkan secara berurutan mendatar. Kolom keenam adalah kolom Waktu (Timestamp).



Gambar 5.35 Tampilan data di Database1.mdb, keempat Tag ditampilkan secara mendatar

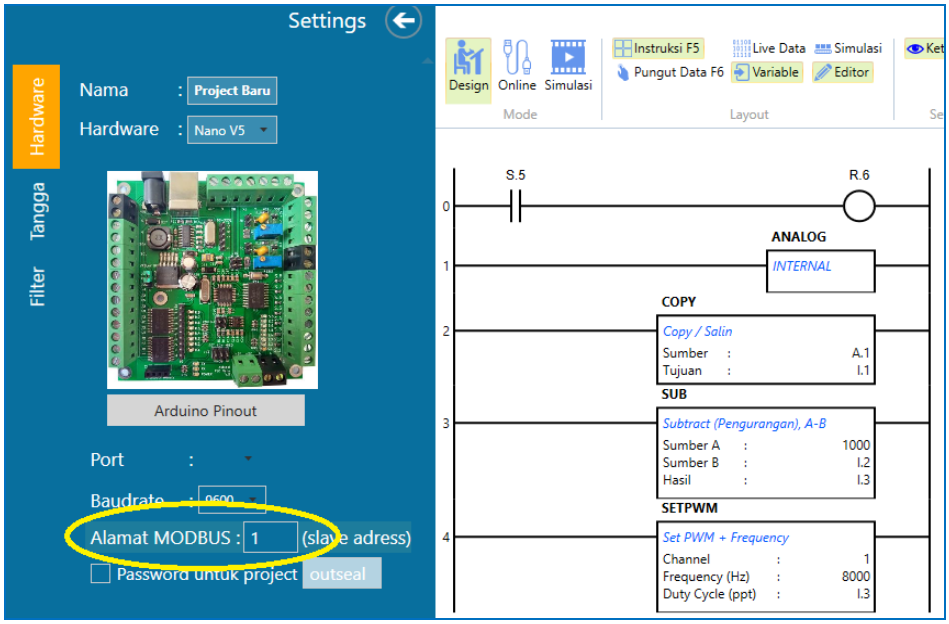
45. Sampai di sini pembuatan database dengan bantuan Data Logger KEPServerEX telah selesai.

5.3 Alarm dengan Data Logger dan LabVIEW

Fitur Alarm merupakan fitur SCADA yang penting. Alarm memungkinkan pihak yang berkaitan dapat segera menangani kondisi abnormal yang terjadi. Pemberitahuan alarm kepada operator dan kepada pihak yang berkaitan dapat disajikan dalam berbagai bentuk, bisa dalam bentuk sirine, lampu indikator, tampilan HMI, email/SMS, atau notifikasi di HP. Dalam Sub Bab ini, akan disajikan langkah-langkah pembuatan alarm dalam bentuk tampilan HMI LabVIEW, baik untuk alarm *real time* maupun *historical*. Berikut langkah-langkahnya:

1. Gunakan rangkaian 2 buah Arduino di Proteus seperti Gambar 4.51, seperti yang juga digunakan di Sub Bab 5.2.

2. Gunakan program Outseal berikut ini untuk kedua Arduino.

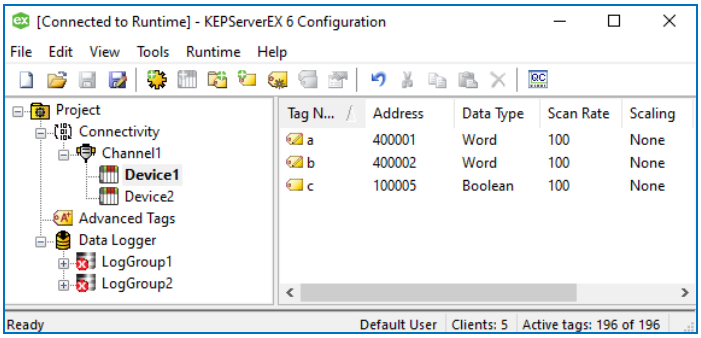


Gambar 5.36 Program Outseal untuk kedua Arduino, atur alamat Modbus U1 = 1, dan U2 = 2

3. Program di atas, selain untuk membaca nilai analog Potensio, mengatur nyala LED PWM di D9, juga membaca tombol yang terhubung di D2. Setiap kali tombol ditekan, LED di D8 akan menyala.
4. Kompilasi program di atas dengan menekan tombol Test. Namun sebelum kompilasi, buat alamat Modbus Arduino U1 = 1, dan alamat Modbus Arduino U2 = 2. Setelah kompilasi, isikan file Hex di U1 dan di U2.

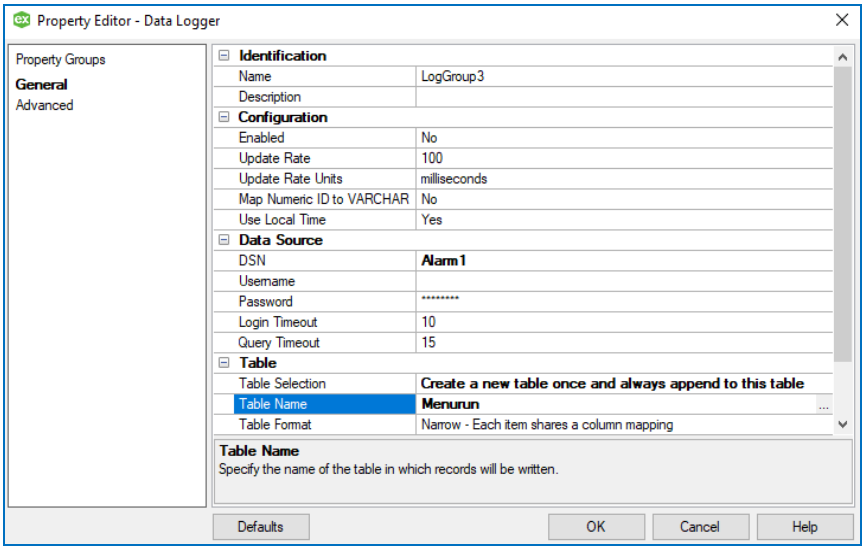
Catatan: Rangkaian Proteus dan program Outseal di atas digunakan untuk mensimulasikan Alarm. Diinginkan, setiap kali nilai analog Potensio lebih dari 800 (nilai ini mensimulasikan tinggi air dalam tangki mendekati batas maksimum), maka sebuah peringatan bahaya dikirimkan ke HMI. Seseorang (teknisi) kemudian akan menangani alarm ini dengan menekan tombol di D2 sebagai tanda proses perbaikan sedang dilakukan, dan ketika tombol D2 dilepas, proses perbaikan telah seleai. Ketika level air dalam tangki telah turun (nilai analog Potensio kurang dari 800), maka sebuah informasi kondisi Normal dikirimkan ke HMI

5. Berikutnya, buka KEPServerEX Configuration. Lanjutkan konfigurasi yang sudah dibuat di Sub Bab 5.2. Tambahkan sebuah Tag dengan nama c, dengan alamat 100005, di Device1 dan Device2.



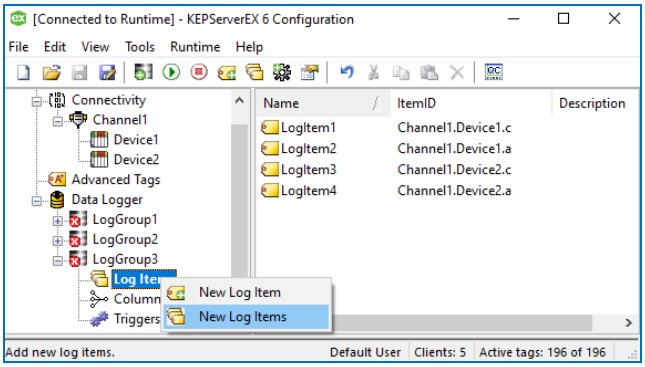
Gambar 5.37 Menambahkan sebuah Tag c (100005) di Device1 dan Device2

6. Berikutnya, tambahkan sebuah Data Logger dengan meng-klik kanan Data Logger, pilih New Log Group, maka muncul Property Editor Data Logger.
7. Di Property Editor Data Logger, isi kolom DSN. Lihat kembali langkah no. 8 – 16 di Sub Bab 5.2 untuk membuat DSN. Isi Table Selection = *Create a new table once …*, Table Name = Menurun, Table Format = *Narrow*.



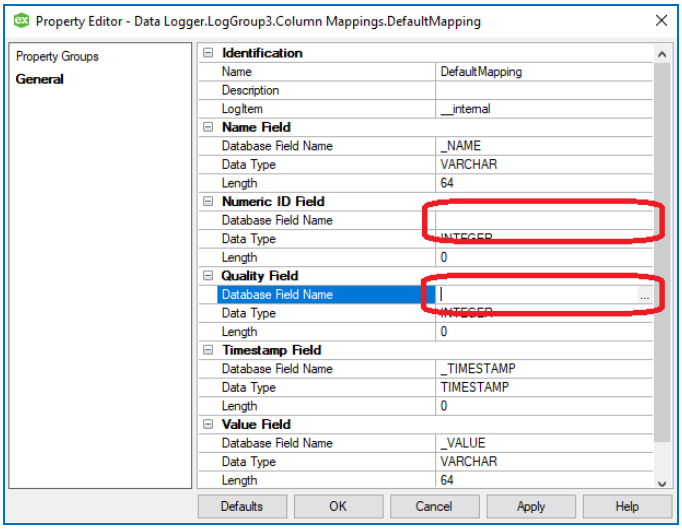
Gambar 5.38 Buat DSN dan file database (*.mdb) yang baru, atur Table Selection = Create a new Table once …, Table Name = Menurun, Table Format = Narrow

8. Klik OK, maka muncul LogGroup3. Klik kanan pada Log Items, pilih New Log Items. Tambahkan Tag a dan c dari Device1 dan Device2, sehingga ada 4 buah LogItem.



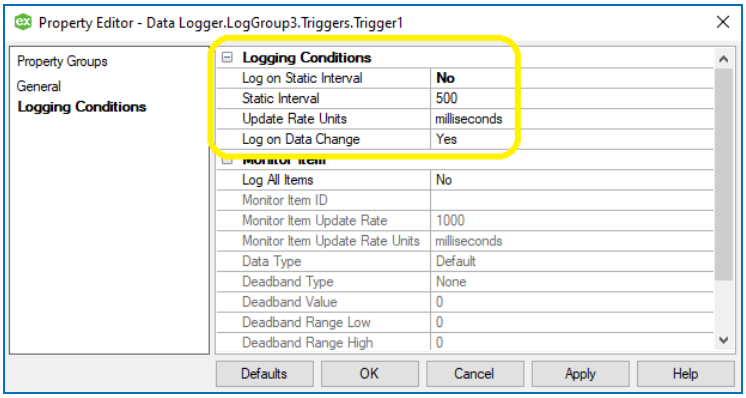
Gambar 5.39 Tambahkan Tag a dan c dari Device1 dan Device2 di Log Items LogGroup3

9. Di Property Editor Column Mappings, hapus nama kolom nomor ID dan kualitas data, sehingga hanya tinggal kolom nama, nilai dan waktu.



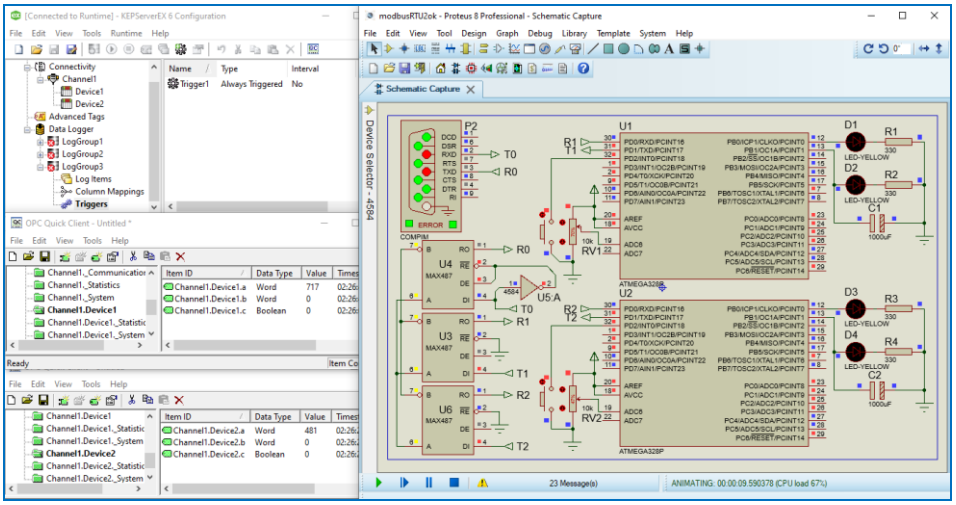
Gambar 5.40 Di Property Editor Column Mappings, hapus nama kolom nomor ID dan kualitas

10. Di Property Editor Triggers, buat agar penyimpanan data dilakukan hanya saat data berubah saja, tidak periodik, dengan cara membuat Log on Static Interval diisi No, dan Log on Data Change diisi Yes.



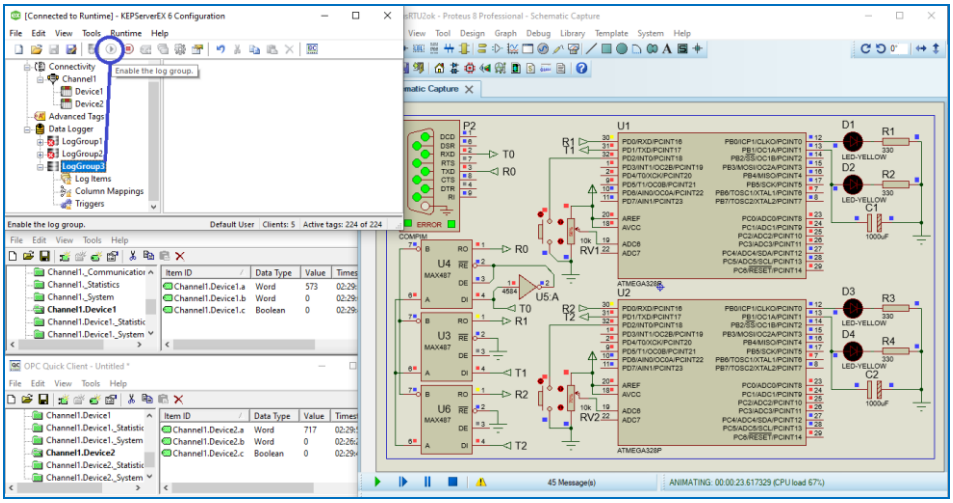
Gambar 5.41 Di Property Editor Triggers, isi Log on Static Interval = No

11. Setelah pengaturan LogGroup3 selesai, jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client bisa menampilkan data Tag a dan c di Device1 dan Device2. Apabila OPC Quick Client tidak bisa menampilkan nilai Tag, reset waktu Runtime OPC Server.



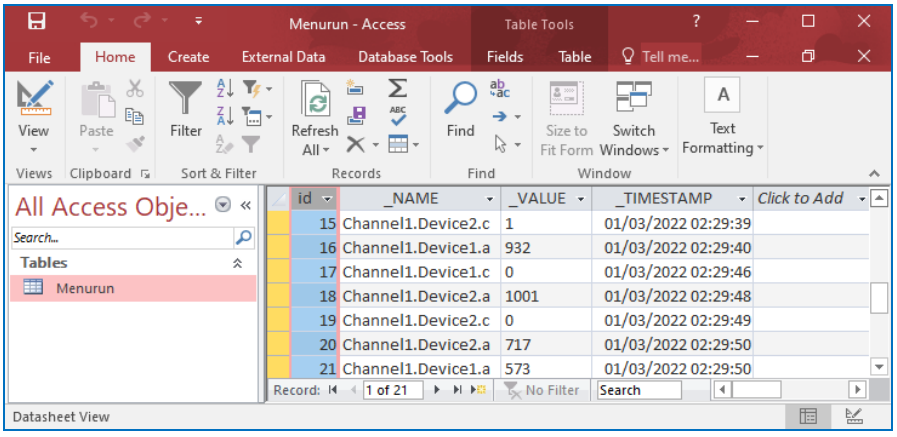
Gambar 5.42 Pastikan OPC Quick Client bisa menampilkan nilai Tag a dan c di Device1 dan 2

12. Apabila data Tag sudah bisa ditampilkan, maka berikutnya, jalankan Data Logger dengan meng-klik kanan LogGroup3, dan pilih Enable Log Group, atau dengan memilih LogGroup3, dan tekan tombol Start di Toolbar.
13. Lakukan pengubahan kondisi Tombol dan Potensio di U1 dan di U2.



Gambar 5.43 Menjalankan Data Logger LogGroup3 dengan menekan tombol Start di Toolbar

14. Setelah beberapa saat, hentikan Data Logger, dan buka file database yang diatur di Property Editor Data Logger dengan nama Table Menurun.



Gambar 5.44 Hasil penyimpanan LogGroup3 di database Microsoft Access

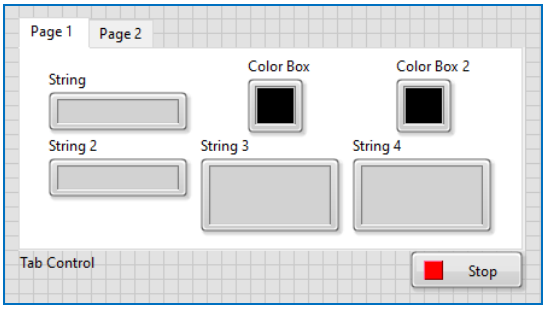
15. Tampak pada Gambar 5.44 di atas, ada 4 kolom data; kolom id, nama, nilai dan waktu. Kolom id dibangkitkan secara otomatis oleh Access, sedangkan kolom nama, nilai dan waktu sesuatu pengaturan di Column Mappings. Setelah penyimpanan database berhasil, langkah selanjutnya mengatur tampilan Alarm di LabVIEW. Buka LabVIEW, buat VI yang baru.

16. Di Front Panel VI yang baru, tempatkan 9 buah objek yang diambil dari Palet Controls, di kategori sesuai Tabel berikut ini.

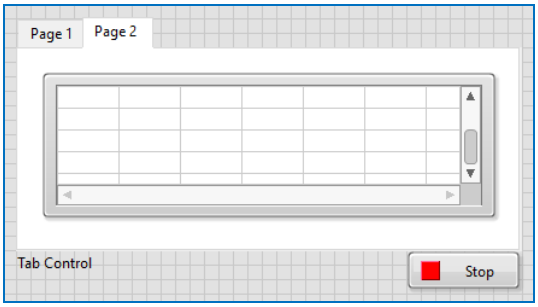
Tabel 5.1 Objek untuk Tampilan Alarm

No.	Nama Objek	Jumlah	Kategori Objek
1.	System Tab Control	1	System >> Containers
2.	String Indicator	4	Silver >> String & Path
3.	Color Box	2	Silver >> Numeric
4.	Stop Button	1	Silver >> Boolean
5.	Table	1	Silver >> List, Table & Tree

17. Mula-mula tempatkan Tab Control. Di Page 1, tempatkan 4 buah String Indicator dan 2 buah Color Box. Di Page 2, tempatkan Table. Tempatkan tombol Stop di luar Tab Control, di bagian bawah.



Gambar 5.45 Tempatkan 4 buah String Indicator dan 2 buah Color Box di Page 1

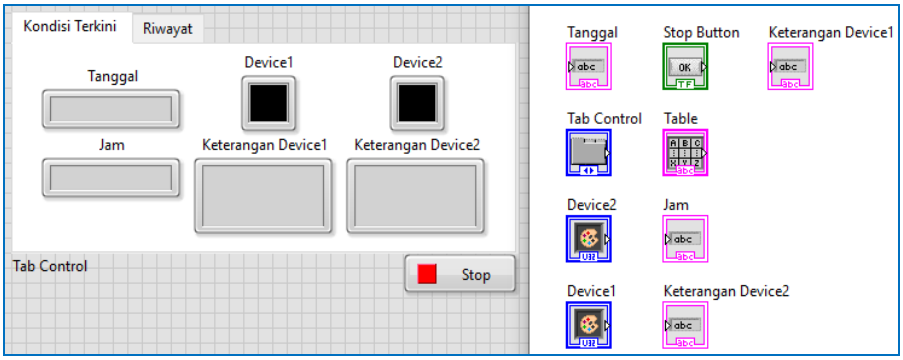


Gambar 5.46 Tempatkan Table di Page 2, dan tombol Stop di bawah kotak Tab Control

18. Ubah label semua objek seperti Tabel berikut.

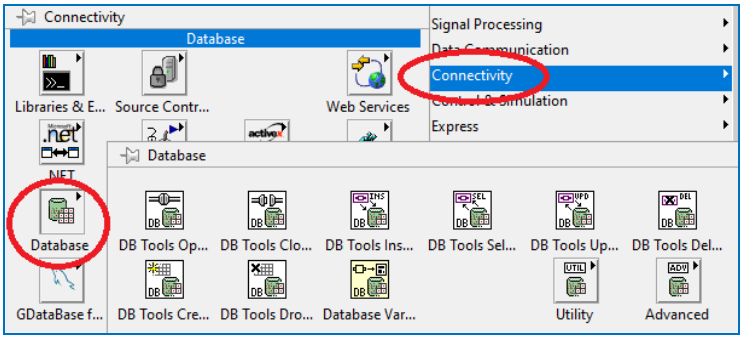
Tabel 5.2 Pengubahan Label Objek

No.	Label Lama	Label Baru
1.	Page 1, Page 2	Kondisi Terkini, Riwayat
2.	String, String 2	Tanggal, Jam
3.	String 3	Keterangan Device1
4.	String 4	Keterangan Device2
5.	Color Box, Color Box 2	Device1, Device2



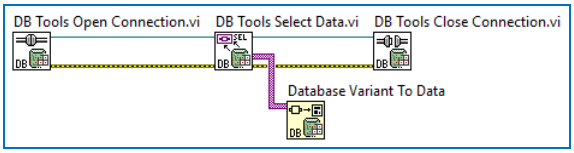
Gambar 5.47 Hasil pengubahan nama label pada Objek

19. Setelah objek pada Front Panel selesai ditempatkan, tambahkan fungsi Database, yang diambil dari Palet Functions, di kategori Connectivity.



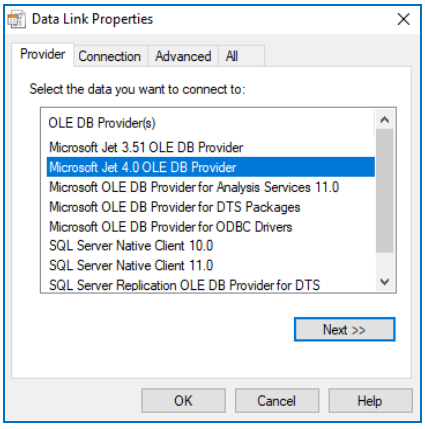
Gambar 5.48 Buka palet Functions, pilih Connectivity, pilih Database

20. Tempatkan 4 buah fungsi Database, yaitu DB Tools Open Connection, DB Tools Select Data, DB Tools Close Connection dan Database Variant To Data. Hubungkan seperti gambar berikut ini.



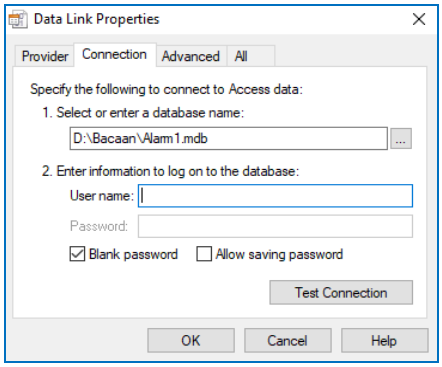
Gambar 5.49 Gunakan 4 fungsi: Open Connection, Select Data, Database, Close Connection

21. Berikutnya, buat link ke file database, dengan cara pilih menu Tools, pilih Create Data Link, maka muncul kotak Data Link Properties. Pilih di Tab Provider: Microsoft Jet 4.0 OLE DB Provider. Klik Next.

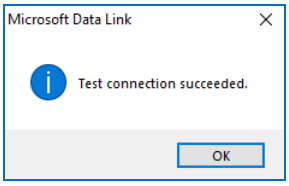


Gambar 5.50 Pilih Create Data Link di menu Tools, pilih Microsoft Jet 4.0 OLE DB Provider

22. Di Tab Connection, di kolom isian pertama, tekan tombol di kanan, arahkan ke lokasi file database, klik OK, maka muncul nama database beserta dengan direktorinya.
23. Hilangkan kata Admin di User name. Tekan tombol Test Connection. Apabila muncul pesan Test Connection Succeeded, maka koneksi berhasil.
24. Klik OK, maka muncul pesan yang memberitahu bahwa data link (file *.udl) berhasil dibuat. Catat nama file beserta dengan direktorinya.
25. Di fungsi DB Tools Open Connection, klik kanan kaki kiri atas, pilih Create, pilih Constant,kemudian isikan nama file data link beserta direktorinya.

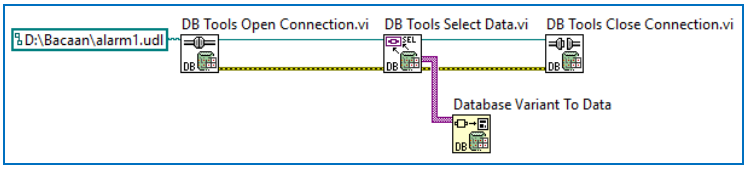


Gambar 5.51 Isi kolom pertama dengan nama database beserta dengan direktorinya



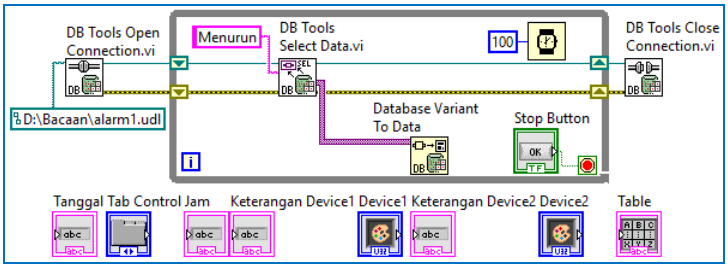
Gambar 5.52 Tekan tombol Test Connection, bila muncul pesan di atas berarti koneksi berhasil

Gambar 5.53 Klik OK, maka file data link (*.udl) berhasil dibuat



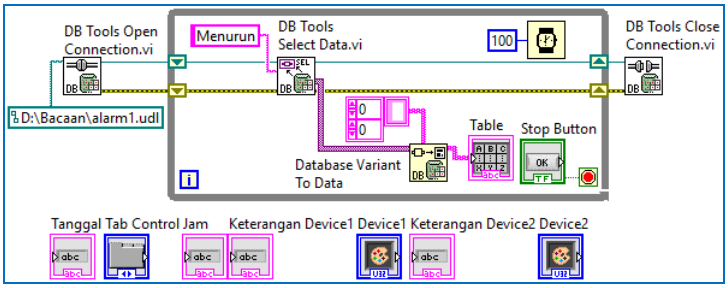
Gambar 5.54 Klik kanan kaki kiri atas fungsi DB Tools Open Connection, pilih Create, pilih Constant, di kotak yang muncul, isikan nama file data link beserta dengan direktorinya

26. Berikutnya, tambahkan Struktur While Loop, buat kotak While Loop melingkupi fungsi DB Tools Select Data dan Database Variant to Data.
27. Masukkan icon Stop Button yang telah dibuat ke dalam Struktur While Loop, hubungkan ke terminal Loop Condition While Loop.
28. Tambahkan fungsi Wait(ms) di dalam Struktur While Loop, beri input 100.



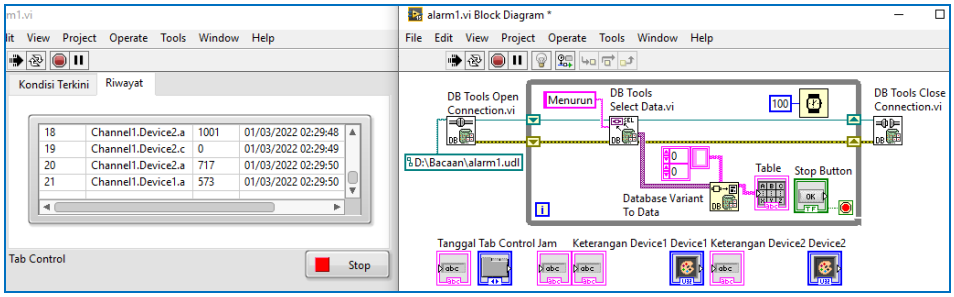
Gambar 5.55 Tambahkan Struktur While Loop pada program, dan fungsi Wait(ms)

29. Diinginkan output fungsi Database Variant to Data ditampilkan di objek Table. Karena tipe data pada objek Table adalah String (warna garisnya pink, yang menunjukkan tipe String), maka di kaki input type Database Variant to Data, perlu mendapat Array 2D Constant yang bertipe String. Cara membuatnya, ambil Array Constant dari Palet Functions di kategori Array, kemudian masukkan sebuah String Constant ke dalam kotak Array tersebut, sehingga kotak Array berubah warnanya, dari hitam menjadi pink. Karena default Array Constant adalah Array 1D, agar menjadi Array 2D, tarik ke bawah kotak kecil di kiri, hingga menjadi 2 kotak.
30. Setelah Array Constant menjadi Array 2D bertipe String, hubungkan ke kaki type Database Variant To Data. Kemudian hubungkan kaki output Database Variant to Data ke icon Table.



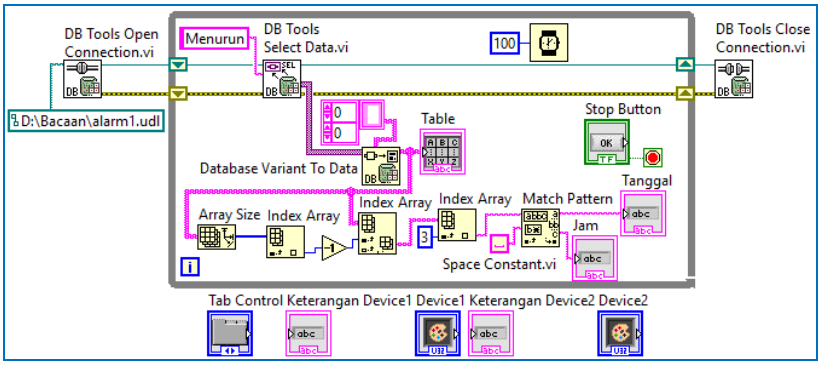
Gambar 5.56 Buat Array 2D String Constant, hubungkan icon tersebut ke kaki input type Database Variant to Data, dan kemudian kaki output data ke icon Table

31. Setelah itu, jalankan program LabVIEW, tekan tombol Run, dan kemudian tekan tombol Stop. Perhatikan bahwa objek Table sekarang telah terisi dengan data yang sama seperti isi database di Gambar 5.44.



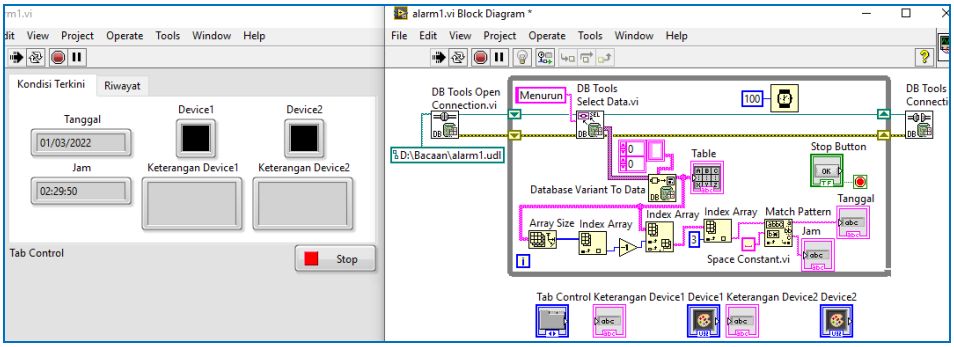
Gambar 5.57 Jalankan LabVIEW, tampak Table telah terisi data yang sama dengan database

32. Diinginkan data yang paling bawah atau yang terakhir, dapat ditampilkan di Kondisi Terkini. Untuk itu, gunakan fungsi Index Array. Dengan memberi input nilai baris yang terakhir di kaki input row Index Array, membuat output Index Array ini menghasilkan sebaris data yang paling bawah.
33. Dari sebaris data yang paling bawah tersebut, dapat diambil data di kolom tertentu, dengan menggunakan fungsi Index Array juga. Untuk mendapatkan data Tanggal dan jam di baris terakhir, beri nilai 3 di kaki input Index Array. Mengapa 3? Karena kolom Tanggal dan Jam di kolom keempat, di mana index selalu dimulai dari angka 0.
34. Untuk memisahkan data Tanggal dengan Jam, gunakan fungsi Match Pattern, dan beri input regular expression dengan Space Constant (spasi), karena data Tanggal dan Jam terpisah dengan sebuah spasi. Kemudian hubungkan icon Tanggal dengan kaki output a Match Pattern, dan icon Jam dengan kaki output c Match Pattern.



Gambar 5.58 Membuat icon Tanggal dan Jam mendapatkan data paling terkini (terakhir)

35. Jalankan program, maka objek Tanggal dan Jam di Tab Kondisi Terkini akan menampilkan data Tanggal dan Jam dari data yang terakhir.



Gambar 5.59 Jalankan program, maka Tanggal dan Jam menampilkan data yang terakhir

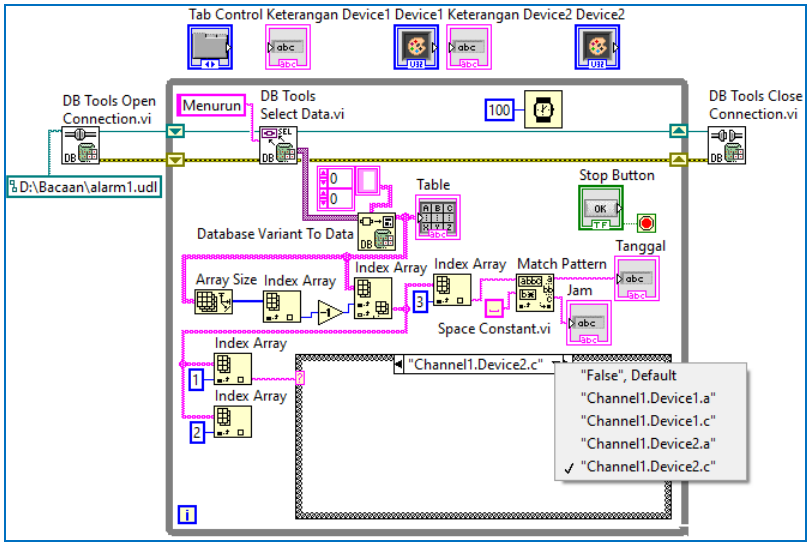
36. Berikutnya, diinginkan agar kotak (*Color Box*) Device1 dan Device2 dapat menampilkan warna sesuai dengan kondisinya. Ada 4 buah warna yang berbeda, yang menunjukkan kondisi Device, seperti terlihat dalam tabel di bawah ini. Ketika nilai batas (800) terlampaui, maka warna Device akan menjadi merah. Apabila normal, maka warna Device menjadi hijau.
37. Selain menampilkan warna sesuai kondisi, di kotak Keterangan Device, akan muncul penjelasan mengenai kondisi yang terjadi.

Tabel 5.3 Tampilan Keterangan dan Warna Device1

No.	Kondisi	Keterangan Device	Warna Device
1.	Device1.a > 800	Level Device1 Bahaya!	Merah
2.	Device1.a < 801	Level Device1 Normal	Hijau
3.	Device1.c = 1	Device1 Proses Perbaikan	Biru
4.	Device1.c = 0	Perbaikan Device1 selesai	Hijau Biru
5.	Device2.a > 800	Level Device2 Bahaya!	Merah
6.	Device2.a < 801	Level Device2 Normal	Hijau
7.	Device2.c = 1	Device2 Proses Perbaikan	Biru
8.	Device2.c = 0	Perbaikan Device2 selesai	Hijau Biru

Catatan: Device1.a adalah Potensio di U1, Device2.a adalah Potensio di U2, Device1.c adalah Tombol di U1, dan Device2.c adalah Tombol di U2. Tombol ditekan bernilai 1, sedangkan bila dilepas bernilai 0. Bila tombol ditekan, warna Device adalah Biru, sedangkan bila dilepas, warna Device adalah Hijau Biru.

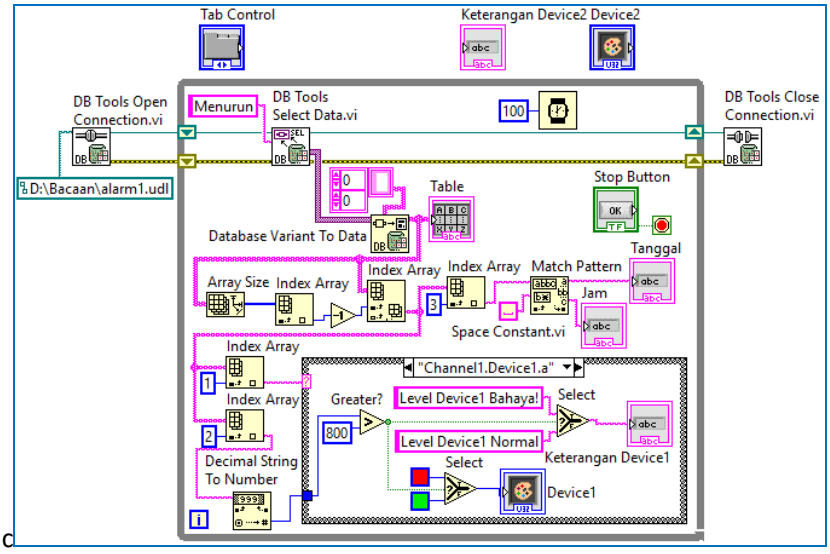
38. Untuk mendapatkan data terakhir untuk nama Device dan Tag, gunakan Index Array dengan input 1 (kolom kedua). Kemudian untuk mendapatkan nilai Tag, gunakan Index Array dengan input 2 (kolom ketiga). Karena warna Device dan keterangan untuk setiap Device dan Tag berbeda-beda, maka tambahkan sebuah Struktur Case. Hubungkan input Struktur Case ke output Index Array yang diberi input 1. Karena output Index Array ini berupa String, maka label Selector Struktur Case akan mendapat tanda petik. Secara default, ada 2 Case, yaitu "True" dan "False" Default. Ubah label "True" menjadi "Channel1.Device1.a". Kemudian tambahkan 3 Case lagi: "Channel1.Device1.c", "Channel1.Device2.a", "Channel1.Device2.c".



Gambar 5.60 Menambahkan Struktur Case dengan input nama Tag

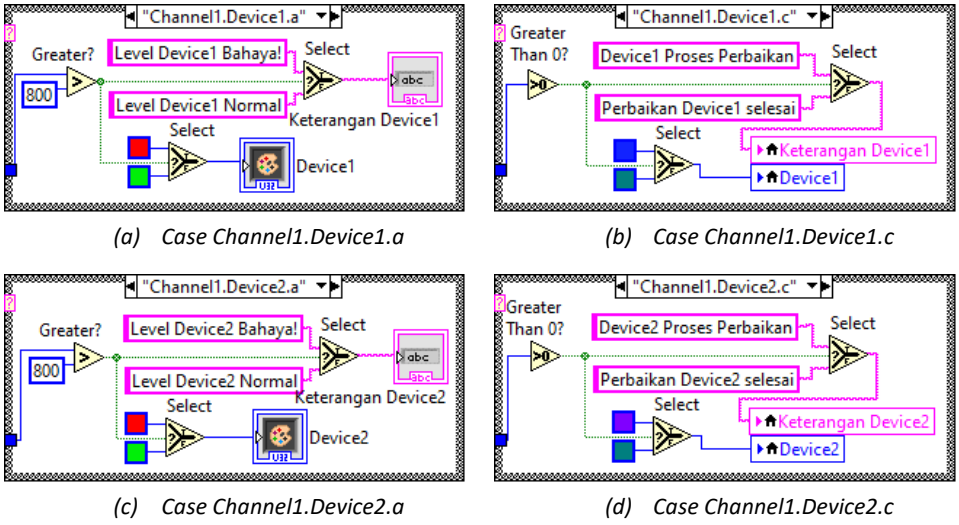
39. Di Case "Channel1.Device1.a", masukkan icon String Indicator Keterangan Device1 dan Color Box Device1.

40. Tambahkan String Constant dan Color Constant. Cara membuat String Constant, klik kanan input Keterangan Device1, pilih Create, pilih Constant. Isi dengan teks “Level Device1 Bahaya!”. Buat lagi String Constant, dan tulis “Level Device1 Normal”. Untuk membuat Color Constant, klik kanan input Color Box Device1, dan pilih Create, pilih Constant. Klik kotak Color Constant, dan ubah warnanya menjadi merah. Buat lagi Color Constant, dan ubah warnanya menjadi hijau. Tambahkan sebuah icon Greater, dan 2 buah Select, buat seperti gambar berikut ini.



Gambar 5.61 Ketika nama Tag = “Channel1.Device1.a” dan nilai Tag lebih dari 800, Keterangan Device1 = “Level Device1 Bahaya!” dan warna Device1 merah. Bila nilai Tag kurang dari 800, Keterangan Device1 = “Level Device1 Normal” dan warna Device1 hijau

41. Berikutnya, copy isi Channel1.Device1.a, tempelkan di semua Case, kecuali Case “False” Default (dikosongkan saja Case “False” Default ini).
42. Untuk Case “Channel1.Device1.c”, perbaiki hasil dari copy paste dengan menghapus icon Keterangan Device1x, dan icon Device1x. Ganti icon ini dengan Local Variable Keterangan Device1 dan Local Variable Device1. Perbaiki juga teks di kedua String Constant dan warna di kedua Color Constant, seperti terlihat pada Gambar 5.62. Perbaiki juga untuk “Channel1.Device2.a” dan “Channel1.Device2.c” seperti Gambar 5.62.

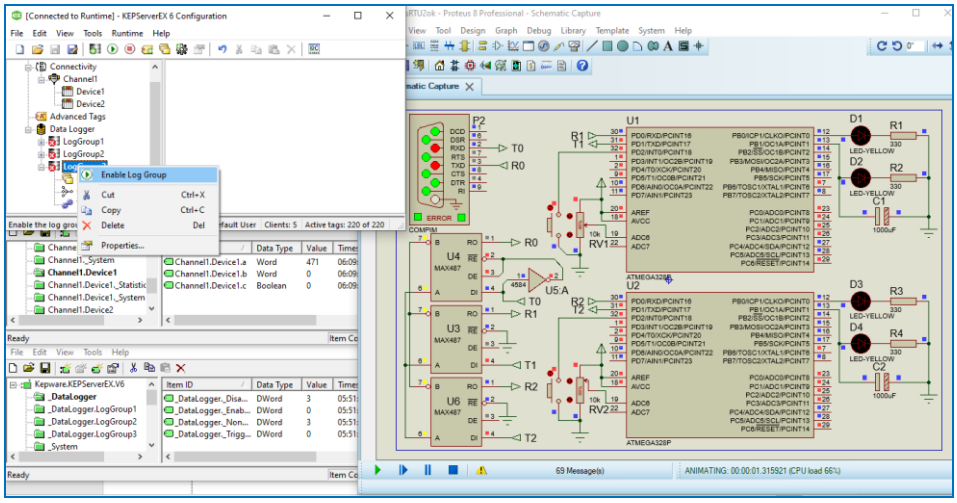


(a) Case Channel1.Device1.a *(b) Case Channel1.Device1.c*

(c) Case Channel1.Device2.a *(d) Case Channel1.Device2.c*

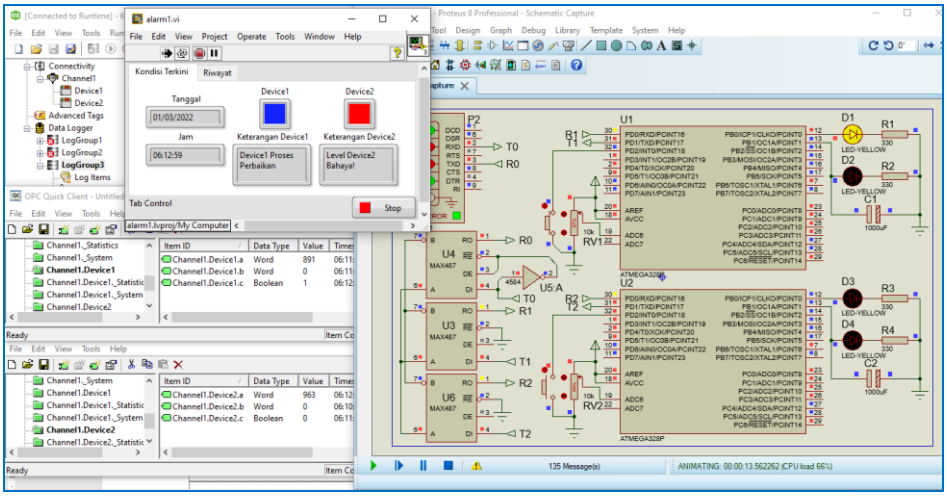
Gambar 5.62 Isi Struktur Case pada setiap Case (untuk Case "False" Default isinya kosong). Khusus untuk Channel1.Device1.c dan Channel1.Device2.c, gunakan Local Variable sebagai ganti icon String Indicator Keterangan Device dan icon Color Box Device hasil copy paste

43. Berikutnya jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client dapat menampilkan nilai Tag di kedua Device. Setelah bisa muncul nilainya, jalankan Data Logger dengan meng-Enable LogGroup3.



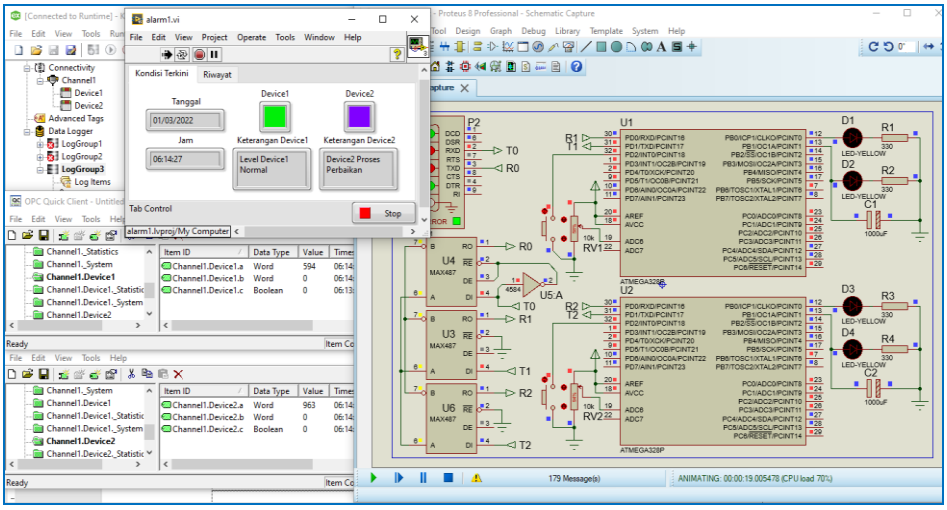
Gambar 5.63 Jalankan rangkaian Proteus, dan pastikan OPC Quick Client bisa menampilkan nilai Tag, setelah itu jalankan Data Logger dengan meng-klik kanan LogGroup3, pilih Enable Log Group atau dengan menekan tombol Start di Toolbar KEPServerEX Configuration

44. Setelah LogGroup3 di-Enable, jalankan program LabVIEW. Tekan Tombol dan geser Potensio di rangkaian U1 dan U2, dan amati warna pada kotak Device1 dan Device2, serta Keterangan Device1 dan Device2.



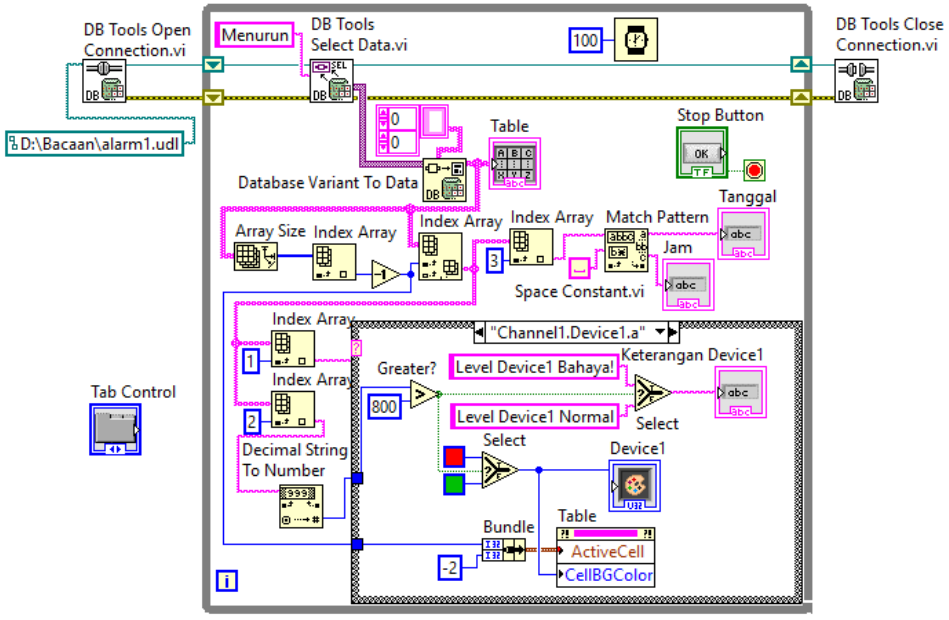
Gambar 5.64 Warna berubah dan muncul Keterangan setiap kali tombol dan potensio digeser

45. Ketika Potensio digeser turun, warna Device menjadi hijau, dan muncul keterangan Level Normal. Ketika tombol ditekan, warna Device menjadi biru, dan muncul keterangan Proses Perbaikan.



Gambar 5.65 Warna Device hijau saat Potensio digeser turun, dan biru saat tombol ditekan

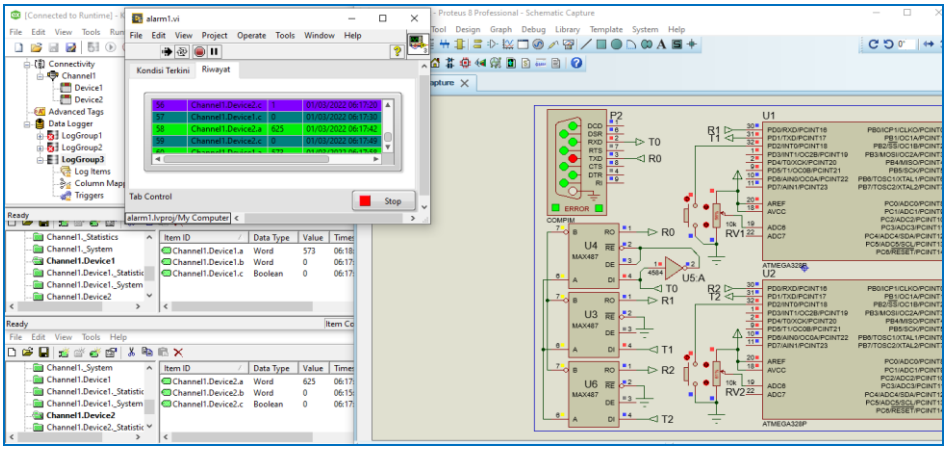
46. Diinginkan agar objek Table tidak hanya menampilkan data, tetapi juga menampilkan warna background pada baris data sesuai nilainya, yang mengikuti Tabel 5.3. Untuk itu, tambahkan Property Node CellBGColor Table di setiap Case, yang mengatur warna background pada baris. Untuk menentukan baris yang akan diwarnai, gunakan Property Node ActiveCell Table. Lebih jelasnya, perhatikan gambar Block Diagram berikut ini:



Gambar 5.66 Penambahan icon Property Node Active Cell dan CellBGColor Table di setiap Case, agar bisa menampilkan warna pada baris sesuai dengan nilai datanya

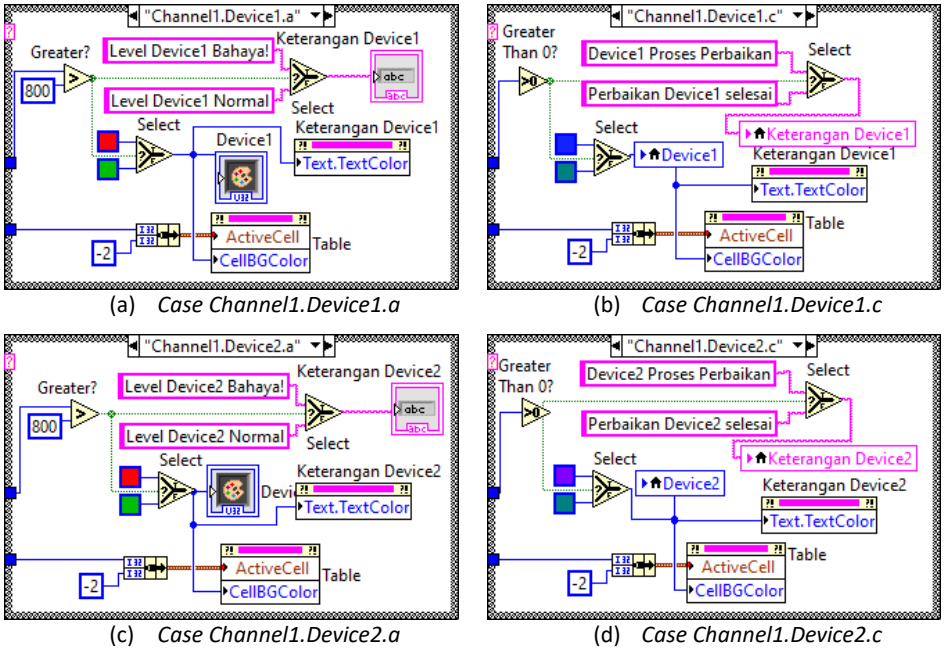
47. Untuk memunculkan Property Node ActiveCell Table, klik kanan pada icon Table, pilih Create, pilih Property Node, pilih ActiveCell, pilih ActiveCell.
48. Setelah muncul icon Property Node ActiveCell, tarik ke bawah icon tersebut, sehingga muncul Property Node CellBGColor.
49. Agar Property Node Active Cell dan CellBGColor dapat diberi input, klik kanan Property Node ini, pilih Change All to Write.
50. Berikutnya, ambil icon Bundle dari Palet Functions, di Programming, di Cluster, Class & Variant. Ada 2 kaki input icon Bundle. Input pertama untuk nomor baris dan input kedua untuk nomor kolom.

51. Untuk mendapatkan baris terakhir Table, ambil nilai baris dari Array Size, yang dikurangi 1. Untuk memilih semua kolom, beri nilai input -2. Setelah itu hubungkan output icon Bundle ini dengan Property Node Active Cell.
52. Agar warna baris Table sesuai dengan nilainya, yang mana sama dengan warna kotak Device, maka beri nilai input Property Node CellBGColor dari output Select Color Constant di setiap Case.
53. Ulangi langkah 46-52 di atas untuk Case yang lain, kecuali Case “False”.
54. Setelah program di Block Diagram Gambar 5.66 di atas selesai dibuat, jalankan OPC Quick Client, rangkaian Proteus dan LogGroup3, serta program LabVIEW. Tekan tombol dan geser potensio di rangkaian U1 dan U2, dan perhatikan objek Table di Tab Riwayat LabVIEW. Seharusnya objek Table tidak hanya menampilkan nilai, tetapi juga menampilkan warna pada setiap baris sesuai data nilai, seperti gambar berikut ini.



Gambar 5.67 Objek Table menampilkan warna di setiap baris sesuai dengan data nilai

55. Berikutnya, untuk memperbaiki tampilan di Tab Kondisi Terkini, pilih objek String Indicator Tanggal, tekan tombol Application Font di Toolbar, pilih Size = 24 pt, Style = Bold dan Justify = Center.
56. Buat hal yang sama untuk objek String Indicator Jam.
57. Berikutnya, diinginkan untuk membuat warna tulisan di Keterangan Device sama dengan warna kotak Device. Untuk itu tambahkan di setiap Case, icon Property Node TextColor Keterangan Device, seperti berikut:

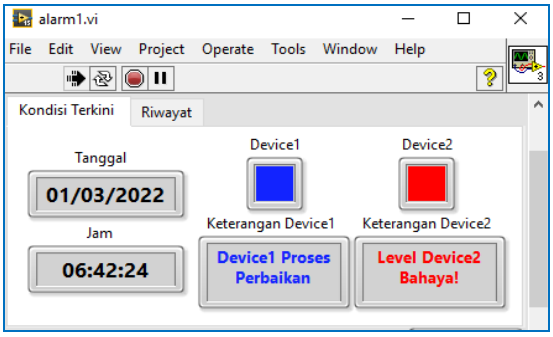


(a) *Case Channel1.Device1.a* (b) *Case Channel1.Device1.c*

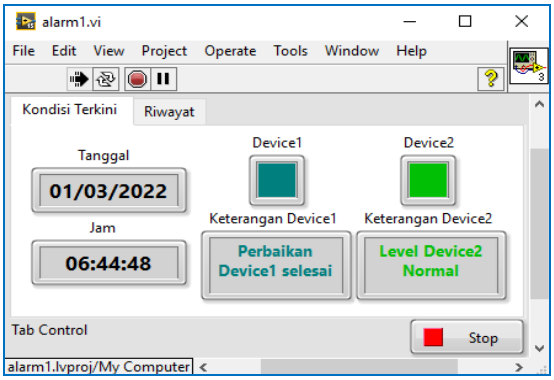
(c) *Case Channel1.Device2.a* (d) *Case Channel1.Device2.c*

Gambar 5.68 Menambahkan Property Node Text.TextColor untuk mengubah warna tulisan di Keterangan Device menjadi sama dengan warna di kotak warna Device

58. Untuk membuat icon Property Node TextColor Keterangan Device1, klik kanan, pilih Create >> Property Node >> Text >> TextColors >> TextColor.
59. Ulangi untuk icon Property Node TextColor Keterangan Device2.
60. Berikutnya, jalankan OPC Quick Client, rangkaian Proteus, LogGroup3, dan program LabVIEW. Seharusnya tulisan pada Keterangan Device berwarna sama dengan kotak warna Device (buat Size = 18 pt, Justify = Center).

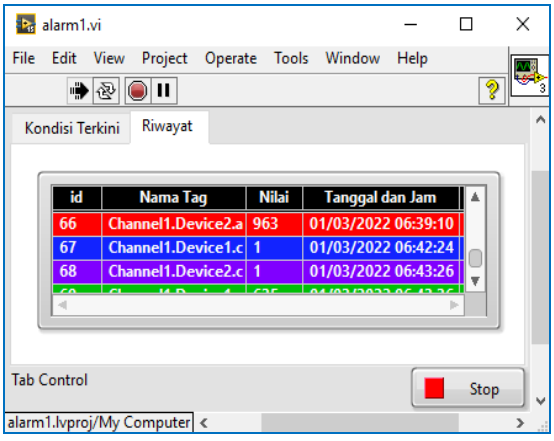


Gambar 5.69 Tulisan pada Keterangan Device berwarna sama dengan kotak warna Device



Gambar 5.70 Kondisi terkini Alarm menampilkan kondisi terkini (real time) dan waktu kejadiannya, serta tampilan warna untuk menunjukkan kategori kondisi

61. Agar tampilan Riwayat lebih jelas, buat tulisan pada Table menjadi putih, dengan cara meng-klik Table, kemudian tekan tombol Application Font, pilih Color, pilih warna putih.
62. Tambahkan juga kepala kolom pada Table dengan cara meng-klik kanan objek Table, pilih Visible Items, beri tanda centang pada Column Headers. Untuk memberi warna pada kepala kolom, tekan tombol Shift dan klik kanan hingga muncul Tool Palet. Gunakan Tool warna berbentuk kuas, sentuhkan pada kepala kolom, dan pilih warna. Jangan lupa untuk mengembalikan mode Auto pada Tool Palet ini (LED Auto harus menyala).



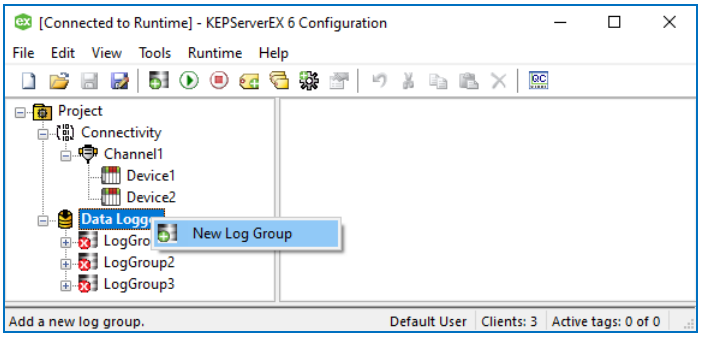
Gambar 5.71 Riwayat Alarm (historical) menampilkan semua kondisi yang pernah terjadi, dengan catatan waktunya, dan tampilan warna untuk menunjukkan kategori kondisi.

63. Sampai di sini pembuatan Alarm yang menampilkan kondisi terkini (*real time*) dan riwayat (*historical*) melalui tampilan HMI dengan bantuan Data Logger dan LabVIEW telah selesai.

5.4 Trend dengan Data Logger dan LabVIEW

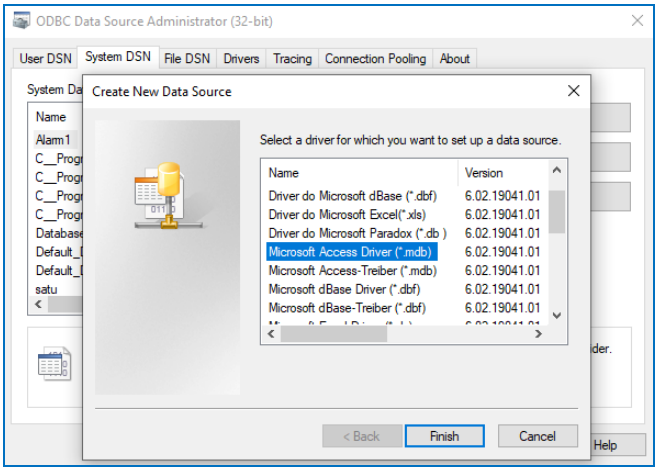
Sesuai dengan namanya, Trend berarti kecenderungan. Dengan adanya Trend, dapat dilihat kecenderungan atau pola dari suatu sistem. Trend sama dengan Alarm, yang merupakan fitur SCADA untuk penyampaian data tertentu. Data pada Alarm adalah data kondisi abnormal, sedangkan data pada Trend adalah data kondisi dari waktu ke waktu dalam bentuk grafik. Dengan tampilan grafik, kecenderungan atau karakteristik perangkat dapat diamati dengan lebih baik. Sama seperti Alarm, Trend juga terdiri dari 2 tipe, yaitu tipe Real Time (data terkini), dan Historical (data dari awal hingga saat ini). Berikut ini langkah-langkah pembuatan Trend menggunakan Data Logger dan LabVIEW:

1. Buka KEPServerEX Configuration. Lanjutkan konfigurasi KEPServerEx pada Sub Bab 5.3, yang memiliki sebuah Channel, dengan 2 buah Device, dan setiap Device memiliki 3 buah Tag, yaitu Tag a di alamat 400001, Tag b di alamat 400002 dan Tag c di alamat 100005.
2. Diinginkan untuk menampilkan semua Tag tersebut (6 buah Tag) dalam bentuk grafik Trend. Untuk itu tambahkan sebuah Data Logger lagi.

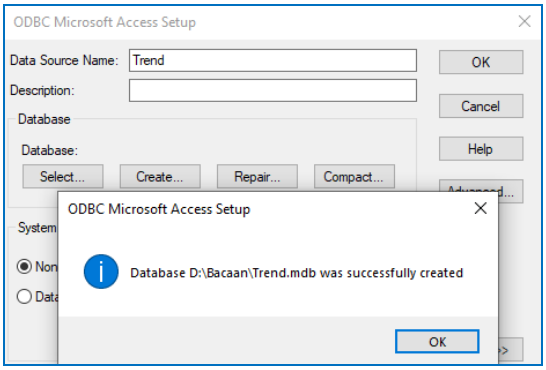


Gambar 5.72 Klik kanan Data Logger, pilih New Log Group untuk membuat LogGroup4

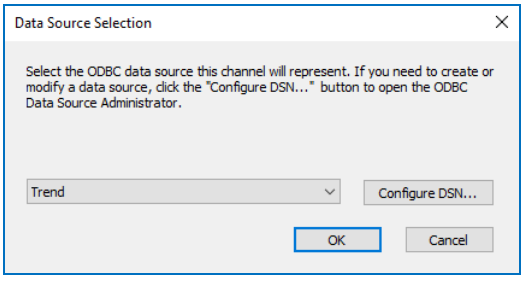
3. Buat sebuah DSN baru dengan nama Trend, dan file database baru.



Gambar 5.73 Klik Configure DSN, pilih Tab System DSN, klik Add, pilih Microsoft Access Driver

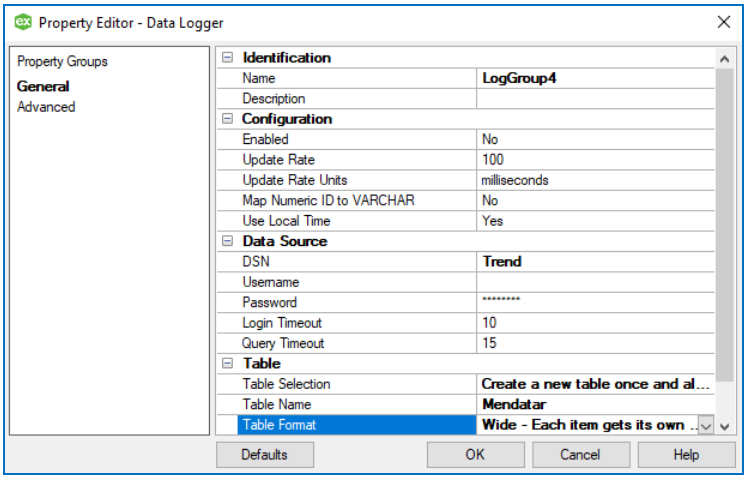


Gambar 5.74 Isi kolom DSN = Trend, klik Create, buat file database baru (contoh Trend.mdb)



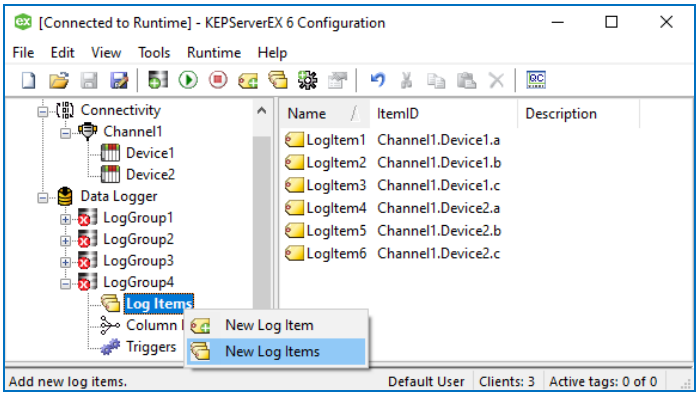
Gambar 5.75 Klik OK, pada Data Source Selection, pilih nama Trend dari daftar, klik OK

4. Pada Property Editor Data Logger, atur Table Selection = ***Create a new table once …***, Table Name = ***Mendatar***, dan Table Format = ***Wide***.



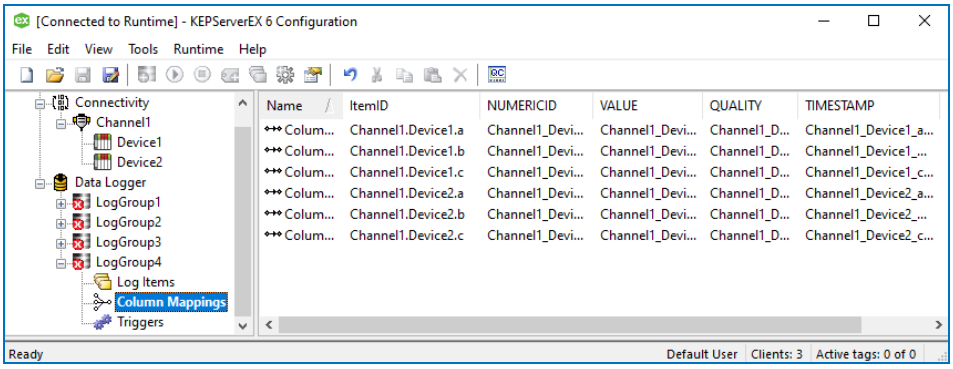
Gambar 5.76 Di Property Editor Data Logger, isi Table Selection = Create a new table once and always append to this table, Table Name = Mendatar, Table Format = Wide

5. Klik OK, maka muncul LogGroup4. Tambahkan Tag yang akan ditampilkan di Trend, dengan cara klik kanan Log Items, pilih New Log Items, pilih Tag a, b dan c di Device1. Klik Apply. Ulangi untuk Tag a, b dan c di Device2.

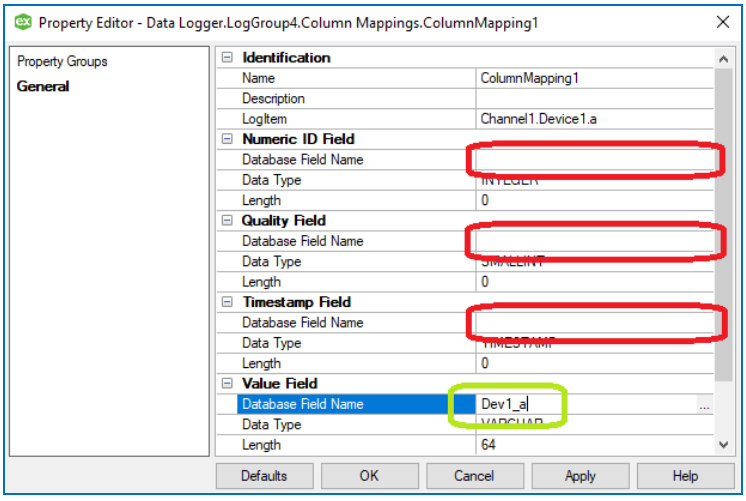


Gambar 5.77 Klik kanan Log Items, pilih New Log Items, pilih Tag a, b, c di Device1 dan Device2

6. Di Column Mappings, terlihat 6 baris pengaturan kolom, jadi satu pengaturan untuk setiap Tag. Buat semua pengaturan tersebut hanya menampilkan nilai, dan untuk Tag yang terakhir, selain nilai, tambahkan tampilan waktu (*Timestamp*). Untuk menghilangkan tampilan yang tidak dikehendaki, hapus namanya di kolom data (*Database Field Name*).

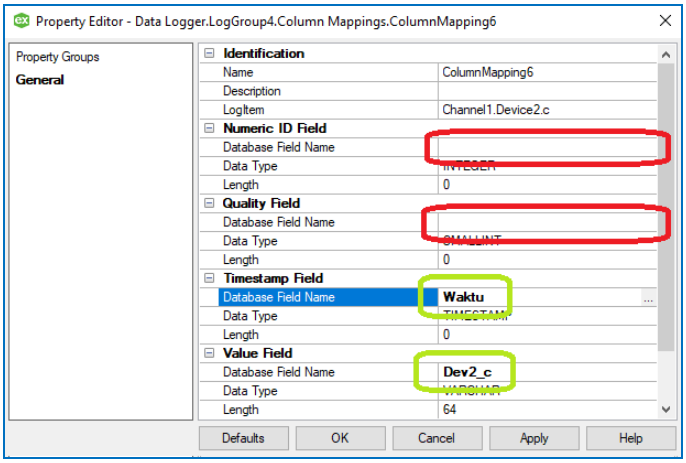


Gambar 5.78 Terlihat 6 buah pengaturan kolom di Column Mappings

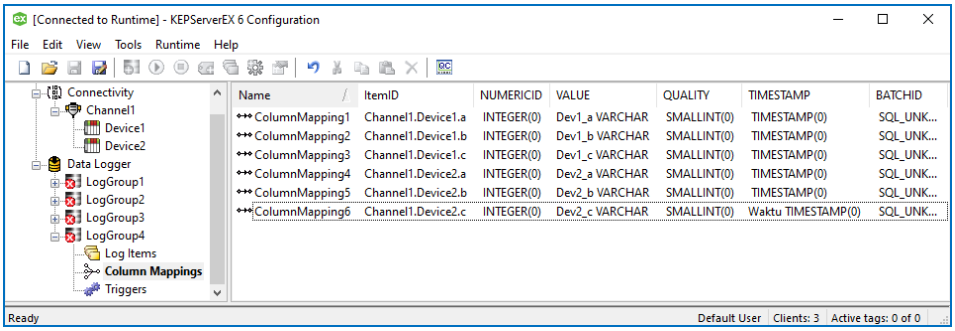


Gambar 5.79 Hapus tiga nama kolom di Database Field Name, yaitu nama kolom Numeric ID, Quality dan Timestamp, sisakan nama kolom Value (Nilai). Agar tidak terlalu panjang, di Database Field Name Value, ubah nama Channel1.Device1.a_VALUE menjadi Dev1_a.

7. Buat keenam pengaturan kolom seperti Gambar 5.79 di atas, kecuali untuk pengaturan yang terakhir (yang keenam), selain nama kolom Nilai tidak dihapus, juga nama kolom Timestamp tidak perlu dihapus.
8. Agar nama kolom tidak terlalu panjang, ubah nama Channel1.Device1.a_ VALUE menjadi Dev1_a. Lakukan juga untuk Tag yang lain. Untuk nama Timestamp, ubah nama Channel1.Device2.c_TIMESTAMP, menjadi Waktu.
9. Setelah keenam pengaturan kolom (Column Mappings) selesai seperti terlihat pada Gambar 5.81, lanjutkan dengan pengaturan Triggers.

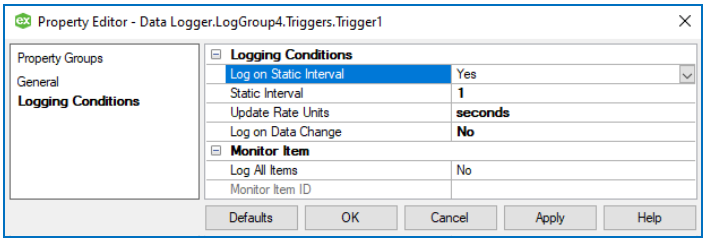


Gambar 5.80 Untuk pengaturan yang terakhir, hapus nama kolom Numeric ID dan Quality, ubah nama kolom Timestamp menjadi Waktu, dan nama kolom Value menjadi Dev2_c



Gambar 5.81 Pengaturan keenam Column Mapping yang hanya menampilkan nilai dari keenam Tag ditambah Timestamp (Waktu) di Tag terakhir

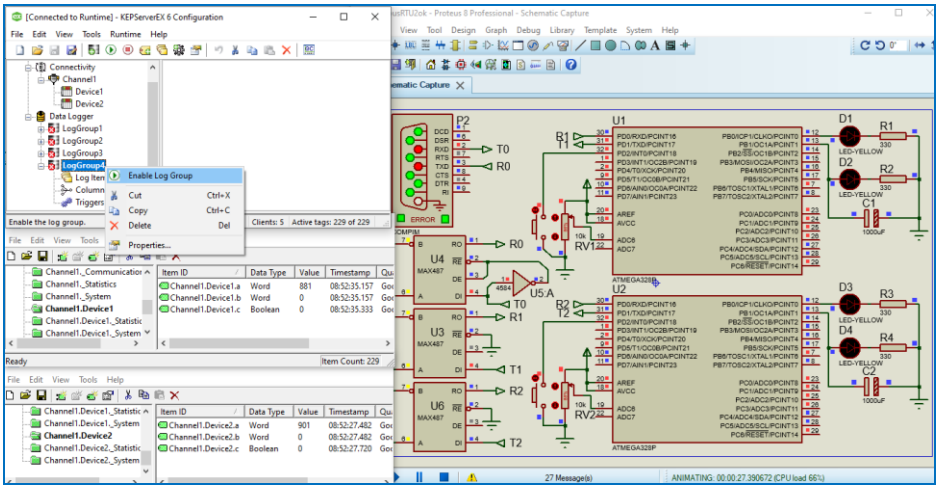
10. Di Triggers, atur Logging Conditions: Log on Static Interval = Yes, Static Interval = 1, Update Rate Units = seconds, Log on Data Change = No.



Gambar 5.82 Atur Logging Conditions: Log on Static Interval = Yes, Static Interval = 1, Update Rate Units = seconds, Log on Data Change = No

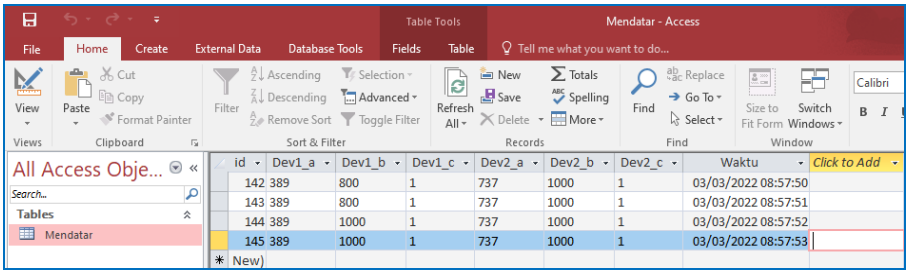
Catatan: Perhatikan perbedaan antara Alarm dan Trend pada Table Format dan Trigger Data Logger. Table Format untuk Alarm adalah Narrow (1 baris untuk 1 Tag), sedangkan Table Format untuk Trend adalah Wide (1 baris untuk semua Tag). Trigger untuk Alarm adalah Log on Data Change (hanya menyimpan data yang nilainya berubah), sedangkan Trigger untuk Trend adalah Log on Static Interval (menyimpan semua data secara berkala). Pengaturan Table Format dan Trigger yang berbeda tersebut dikarenakan fungsi dari kedua fitur tersebut berbeda. Alarm lebih difungsikan untuk memonitor kondisi abnormal dari setiap Tag dan waktu terjadi kondisi tersebut. Sedangkan Trend lebih difungsikan untuk mengamati karakteristik semua Tag dari waktu ke waktu.

11. Jalankan OPC Quick Client dan rangkaian Proteus, dan Enable LogGroup4. Lakukan pengaturan pada Tag a dan c di Device1 dan 2, dengan menekan tombol dan menggeser potensio di rangkaian U1 dan U2, serta mengatur Tag b dengan Asynchronous/Synchronous Write di OPC Quick Client.



Gambar 5.83 Jalankan OPC Quick Client, rangkaian Proteus dan Enable LogGroup4

12. Setelah beberapa saat, Disable LogGroup4, dan buka file database (dalam contoh ini Trend.mdb). Perhatikan, pada Table Mendatar, data semua Tag ditampilkan dalam 1 baris, diawali dengan nomor id, yang dibangkitkan secara otomatis oleh Microsoft Access, dan diakhiri dengan data Waktu.



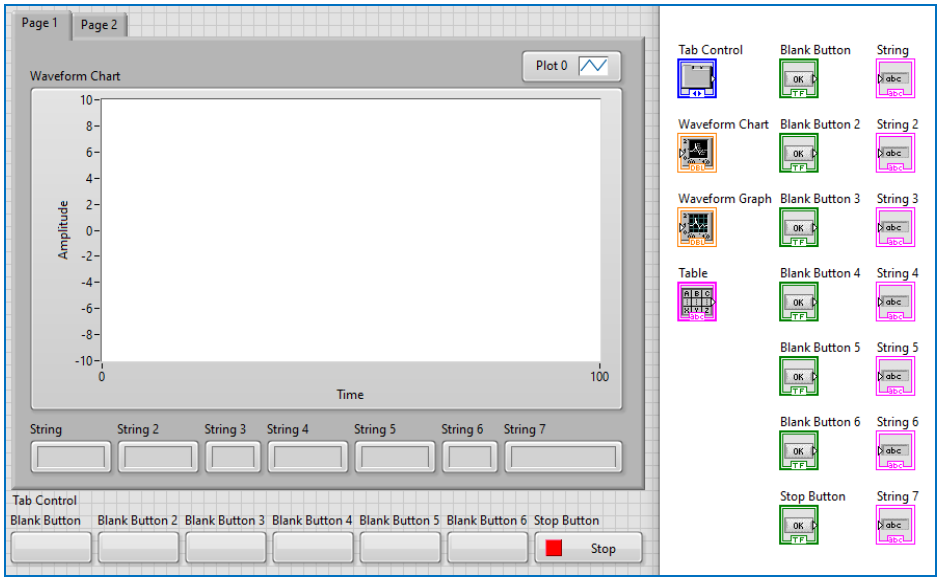
Gambar 5.84 Table Mendatar menampilkan nilai keenam Tag dalam 1 baris dan waktunya

13. Diinginkan data keenam Tag dalam Table Mendatar tersebut ditampilkan dalam bentuk grafik. Untuk itu buka LabVIEW, buat VI yang baru.
14. Di Front Panel, ambil objek dari Palet Controls seperti tabel berikut:

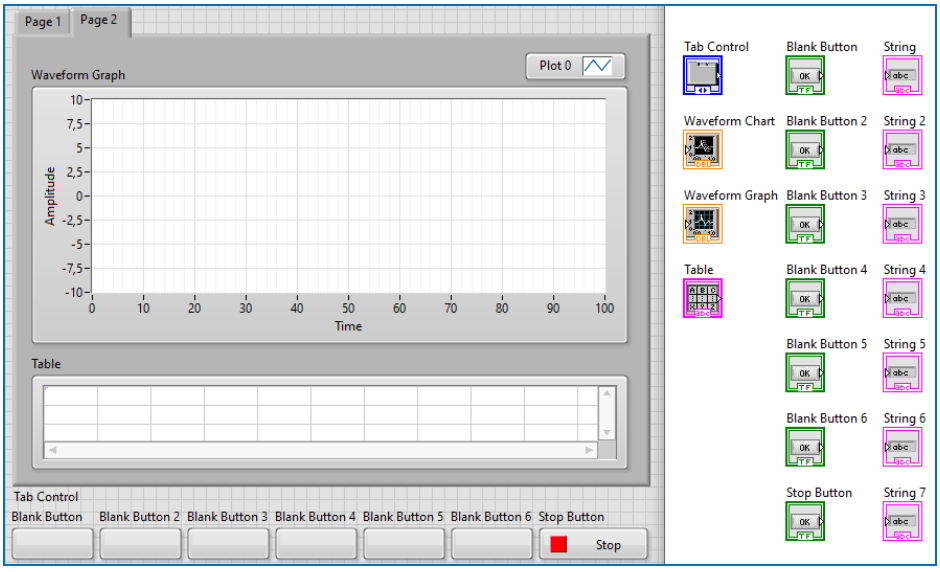
Tabel 5.4 Objek untuk Tampilan Trend

No.	Nama Objek	Jumlah	Kategori Objek
1.	Tab Control	1	Modern >> Containers
2.	Waveform Chart	1	Silver >> Graph
3.	Waveform Graph	1	Silver >> Graph
4.	Table	1	Silver >> List, Table & Tree
5.	Blank Button	6	Silver >> Boolean
6.	Stop Button	1	Silver >> Boolean
7	String Indicator	7	Silver >> String & Path

15. Tempatkan objek-objek yang tertulis di tabel di atas seperti Gambar 5.85 untuk objek di Page1 Tab Control dan Gambar 5.86 untuk objek di Page2.
16. Berikutnya, ubah nama Tab “Page1” Tab Control menjadi “Real Time Trend”, dan nama Tab “Page2” Tab Control menjadi “Historical Trend”.
17. Ubah label ketujuh String Indicator berturut-turut menjadi Dev1_a, Dev1_b, Dev1_c, Dev2_a, Dev2_b, Dev2_c dan Waktu.
18. Munculkan Boolean Text pada keenam Blank Button (klik kanan, pilih Visible Items, beri centang pada Boolean Text), dan secara berturut-turut beri nama Dev1_a, Dev1_b, Dev1_c, Dev2_a, Dev2_b, dan Dev2_c.

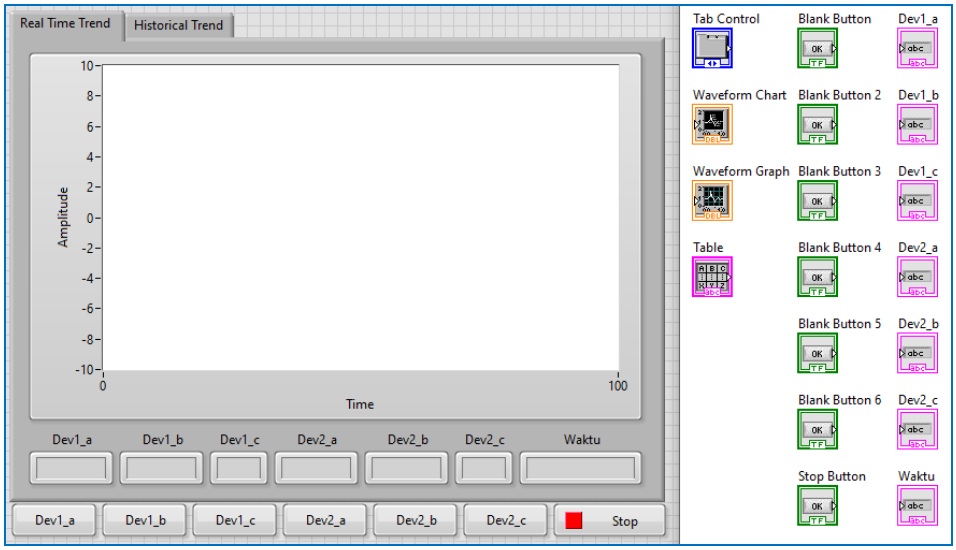


Gambar 5.85 Penempatan objek di Page1 Tab Control dan di luar Tab Control

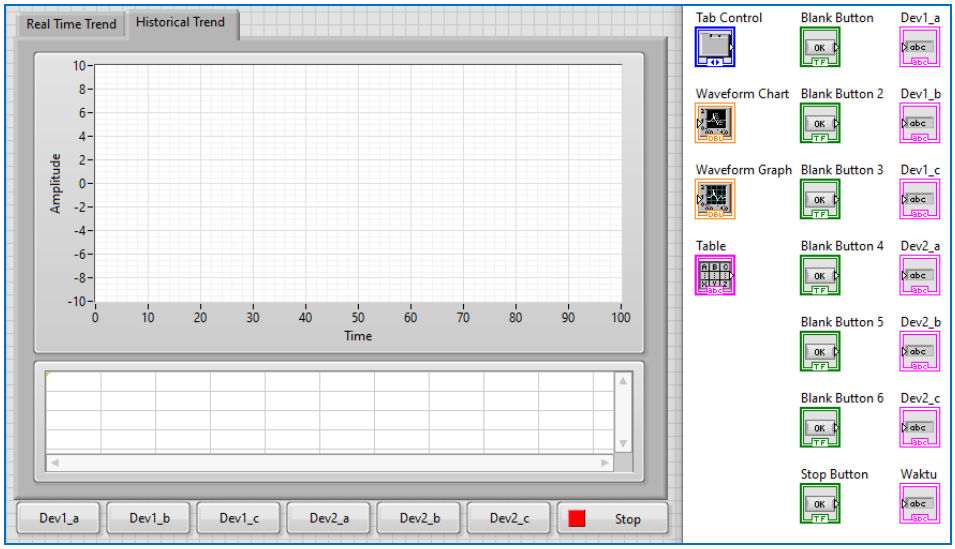


Gambar 5.86 Penempatan objek di Page2 Tab Control dan di luar Tab Control

19. Setelah setiap Blank Button diberi nama di Boolean Text, berikutnya hilangkan label Blank Button (klik kanan Blank Button, pilih Visible Items, hilangkan centang pada Label). Juga hilangkan label Tab Control.



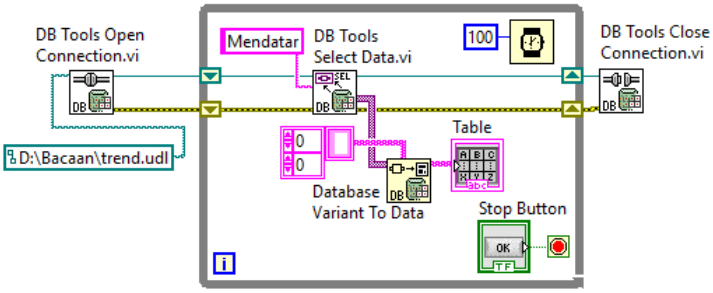
Gambar 5.87 Hilangkan label Waveform Chart dan label ketujuh Blank Button, ganti label String Indicator sesuai nama kolom, serta munculkan Boolean Text di keenam Blank Button, dan buat keenam Boolean Text tersebut menampilkan nama keenam Tag secara berurutan



Gambar 5.88 Di Page2 Tab Control, hilangkan label Waveform Graph dan Table, dan juga Plot Legend Waveform Graph (serta Plot Legend Waveform Chart di Page1 Tab Control)

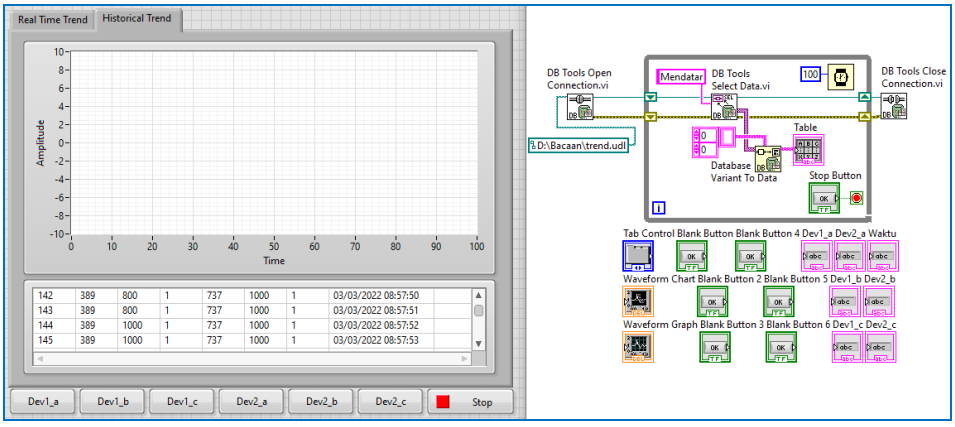
20. Hilangkan juga label Waveform Chart, Waveform Graph, dan Table, serta Plot Legend pada Waveform Chart dan Waveform Graph.

21. Pastikan ada 18 icon objek di Block Diagram, seperti Gambar 5.88 di atas.
22. Berikutnya, buat Block Diagram seperti berikut ini. Untuk langkah-langkah membuatnya, lihat kembali langkah no. 18 – 30 di Sub Bab 5.3. Jangan lupa membuat data link (trend.udl) yang menghubungkan program ini dengan file database trend.mdb, dan juga ganti nama input icon DB Tools Select Data, dari Table Menurun menjadi Table Mendatar.



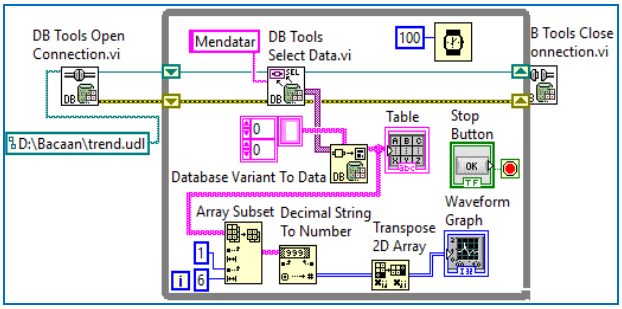
Gambar 5.89 Program mengambil data dari Database Access ditampilkan di objek Table

23. Setelah itu, jalankan program tersebut, tekan tombol Run, dan kemudian tekan tombol Stop. Perhatikan bahwa objek Table sekarang telah terisi dengan data yang sama seperti isi database di Gambar 5.84.



Gambar 5.90 Ketika program dijalankan, maka Table terisi data dari file database

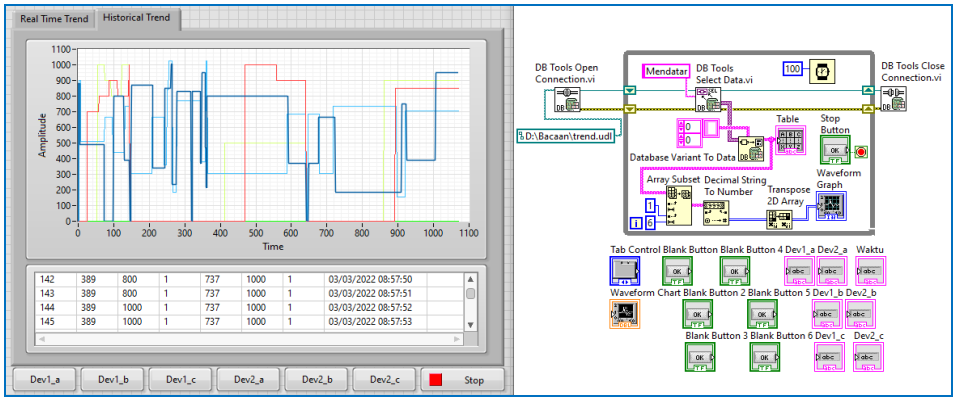
24. Berikutnya, untuk membuat data di Table ditampilkan sebagai grafik di Waveform Graph, tambahkan program berikut ini.



Gambar 5.91 Agar data Table bisa ditampilkan sebagai grafik, tambahkan Array Subset (0,0,1,6), Decimal String to Number, Transpose 2D Array dan Waveform Graph

Catatan: Beberapa icon di Block Diagram, kaki inputnya memiliki nilai default. Dalam Gambar 5.91 di atas, kaki input icon Array Subset yang nomor 2 dan 3 dari atas, memiliki nilai default 0, artinya bila kaki input tersebut tidak dihubungkan dengan suatu nilai Constant, maka input kedua kaki tersebut adalah 0.

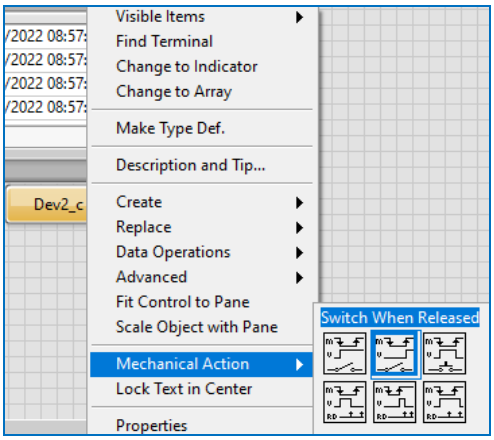
25. Jalankan program di atas, dan perhatikan tampilan Waveform Graph.



Gambar 5.92 Ketika program dijalankan, Waveform Graph menampilkan sejumlah grafik

26. Diinginkan, agar setiap grafik tersebut, dapat dimunculkan dan dihilangkan satu persatu, dengan menggunakan tombol-tombol di samping kiri tombol Stop. Jika tombol Dev1_a ditekan, maka grafik Dev1_a akan muncul. Jika hanya 3 tombol yang ditekan, maka grafik yang bersesuaian dengan ketiga tombol tersebut akan muncul. Jika keenam tombol ditekan, maka keenam grafik akan muncul.

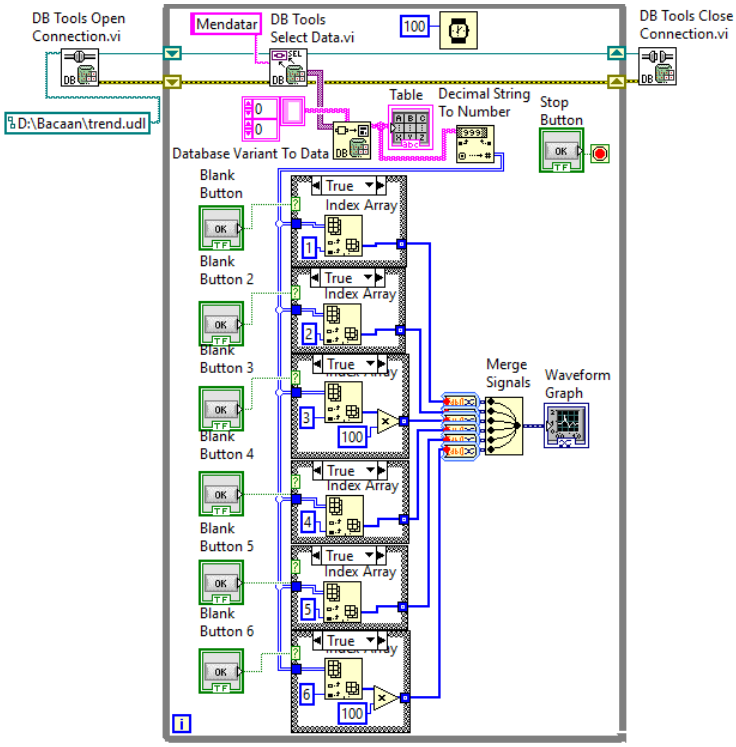
27. Sebelum menambahkan program di Block Diagram, agar keenam tombol yang bersifat *momentary* (aktif sesaat) bisa menjadi *permanent* (terkunci), lakukan pengaturan berikut ini. Klik kanan salah satu tombol, pilih Mechanical Action, ubah *Latch When Released*, menjadi *Switch When Released*, seperti gambar berikut. Ulangi untuk kelima tombol yang lain.



Gambar 5.93 Ubah Mechanical Action tombol dari Latch menjadi Switch agar permanent

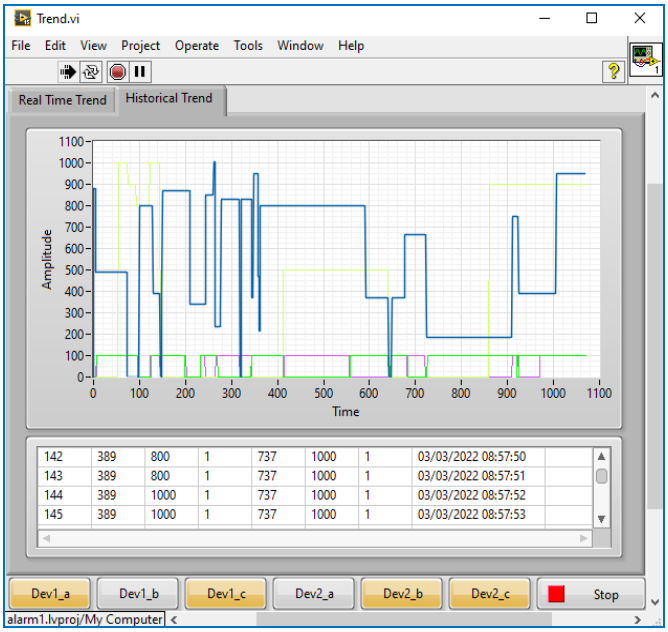
28. Untuk membuat setiap grafik dapat diatur kemunculannya oleh masing-masing tombol, gunakan fungsi Index Array. Namun sebelum menambahkan Index Array, hapus dulu icon Array Subset dan icon Transpose 2D Array. Hubungkan data Table ke icon Decimal String to Number, agar data Array String (Table) menjadi Array Numeric.
29. Berikutnya tambahkan Index Array. Karena ada 6 tombol, tempatkan 6 buah Index Array. Fungsi Index Array ini dapat mengambil sebagian Array Numeric dengan memberikan angka pada input indeks baris atau indeks kolomnya. Bila indeks kolom diberi angka 0, maka Index Array akan menghasilkan semua data Array Numeric di kolom pertama. Bila indeks kolom diberi angka 1, maka Index Array akan menghasilkan semua data Array Numeric di kolom kedua, dan seterusnya (lihat Gambar 5.94).
30. Karena nilai data di kolom keempat (Dev1_c) dan kolom ketujuh (Dev_2c) terlalu kecil, maka agar bisa terlihat pada tampilan grafik bersama, output dari Index Array untuk kedua kolom ini, dikalikan dengan angka 100.

31. Berikutnya, agar output Index Array hanya ditampilkan ketika tombol ditekan, maka masukkan Index Array ke dalam Struktur Case, satu Struktur Case untuk satu Index Array, dan hubungkan masing-masing input Struktur Case (?) dengan sebuah Tombol, satu persatu.
32. Agar semua data output Index Array tersebut dapat ditampilkan dalam satu Waveform Graph, tambahkan Merge signals. Tarik ke bawah Merge signals hingga muncul 6 buah kaki. Hubungkan keenam kaki satu persatu dengan output Index Array. Saat garis data dari Index Array terhubung dengan Merge signals, secara otomatis muncul Dynamic Data Type.
33. Berikutnya, hubungkan output Merge signals ke Waveform Graph.
34. Karena garis data dari Index Array menembus Struktur Case, muncul kotak kecil di dinding Struktur Case. Buat agar kotak tersebut tidak kosong, dengan meng-klik kanan kotak kecil tersebut, pilih Use Default if Unwired.



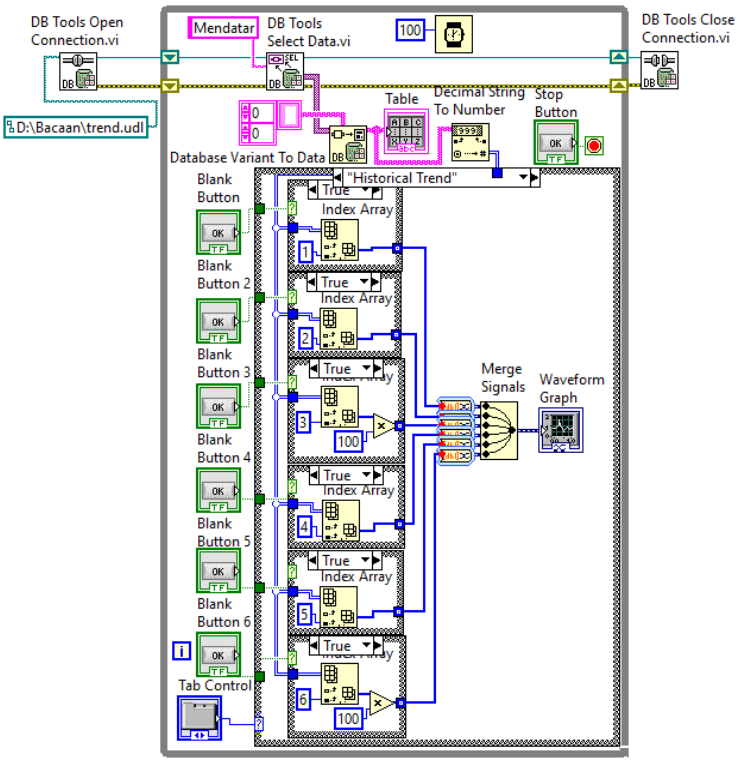
Gambar 5.94 Menggunakan Index Array agar dapat mengambil sebagian data Array, menggunakan Struktur Case, agar setiap Grafik dapat diatur kemunculannya dengan tombol

35. Berikutnya, jalankan program LabVIEW tersebut, dan perhatikan tampilan Waveform Graph. Tekan salah satu tombol, dan perhatikan grafik yang muncul. Tekan keenam tombol, seharusnya keenam grafik juga muncul.



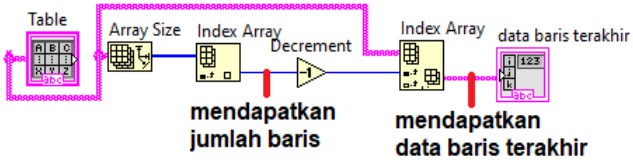
Gambar 5.95 Ketika 4 tombol ditekan, muncul 4 grafik pada Waveform Graph

36. Waveform Graph menampilkan data Riwayat atau Historical Trend, yang menampilkan grafik data dari awal penyimpanan hingga saat ini. Untuk menampilkan grafik data terkini, Real Time, digunakan Waveform Chart.
37. Sebelum menambahkan Waveform Chart, diinginkan agar keenam tombol juga dapat dipakai untuk memunculkan satu persatu grafik di Waveform Chart, untuk itu tambahkan Struktur Case, dan lingkupi keenam Index Array beserta Merge Signals dan Waveform Graph.
38. Kemudian hubungkan input Struktur Case (?) yang baru ini dengan icon Tab Control, maka di Label Struktur Case akan muncul 2 label Case, yaitu “Historical Trend” dan “Real Time Trend”, Default.
39. Pastikan program Waveform Graph di dalam Case dengan label “Historical Trend”. Apabila ternyata berada di Case “Real Time Trend”, silahkan klik kanan label itu, dan pilih Make This Case “Historical Trend”.



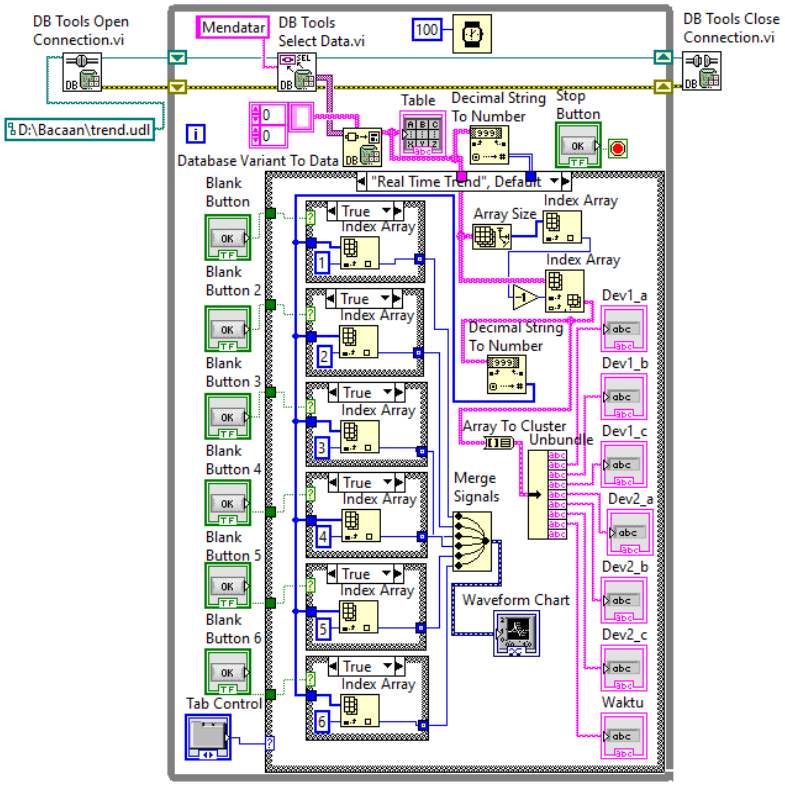
Gambar 5.96 Tambahkan Struktur Case yang melingkupi keenam Index Array, Merge Signals dan Waveform Graph, hubungkan inputnya dengan Tab Control, maka muncul label Historical Trend dan Real Time Trend, sesuai nama Tab pada objek Tab Control di Front Panel

40. Untuk mendapatkan data terkini dari database, dapat diperoleh dari mengambil data pada baris terakhir. Gunakan 2 buah Index Array untuk mendapatkan data baris terakhir. Index Array pertama dengan tambahan Array Size akan menghasilkan jumlah baris. Dari jumlah baris, diberikan ke input index baris Index Array kedua, akan dihasilkan data baris terakhir, seperti terlihat pada Gambar 5.97 berikut ini.

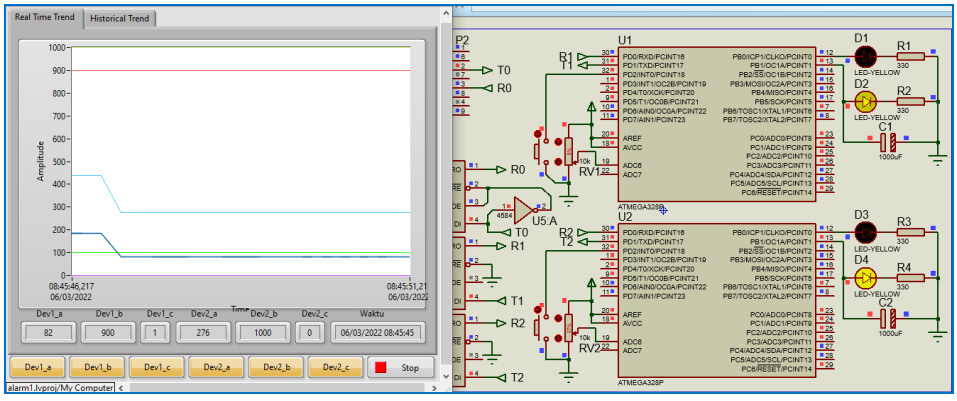


Gambar 5.97 Untuk mendapatkan data di baris terakhir, gunakan 2 buah Index Array, Index Array pertama untuk mengetahui jumlah baris, Index Array kedua menampilkan datanya

41. Tempatkan program Gambar 5.97 di atas ke dalam Case “Real Time Trend”. Tempatkan juga 6 buah Index Array, yang masing-masing ditempatkan ke dalam sebuah Struktur Case.
42. Agar bisa ditampilkan dalam bentuk grafik, data baris terakhir yang bertipe String harus diubah menjadi data Numeric. Karena itu tambahkan icon Decimal String to Number, dan hubungkan output icon ini ke input array keenam Index Array. Untuk mengambil angka di kolom tertentu, isi input indeks keenam Index Array tersebut dengan angka 1 sampai 6.
43. Tambahkan Merge Signals untuk menggabungkan semua output dari keenam Index Array tersebut.
44. Karena garis data keenam Index Array tersebut menembus Struktur Case, buat kotak kecil di setiap dinding Struktur Case tersebut tidak kosong, dengan cara klik kanan kotak tersebut, pilih Use Default if Unwired.
45. Berikutnya, hubungkan output Merge Signals ke Waveform Chart.
46. Agar nilai data terkini atau nilai data pada baris terakhir terlihat dengan jelas, tempatkan 7 buah String Indicator (Dev_1a hingga Waktu) ke dalam Case “Real Time Trend”. Untuk memisahkan atau mengambil sebagian data, selain menggunakan Index Array, cara lain adalah dengan menggunakan Unbundle Cluster. Untuk itu, ubah dulu data Array menjadi Cluster, kemudian gunakan Unbundle. Seperti terlihat pada Gambar 5.98, hubungkan output Index Array yang menghasilkan data baris terakhir ke icon Array to Cluster. Kemudian hubungkan icon Array to Cluster ke Unbundle. Di Unbundle, muncul 9 output. Hubungkan output kedua hingga kedelapan berturut-turut ke String Indicator Dev1_a, Dev1_b, Dev1_c, Dev2_a, Dev2_b, Dev2_c, dan Waktu.
47. Karena data yang ditampilkan adalah data terkini atau data baris terakhir, maka agar tampilan grafik bisa berubah, jalankan OPC Quick Client, rangkaian Proteus, dan Enable LogGroup4. Pastikan OPC Quick Client bisa menampilkan nilai dari keenam Tag. Kemudian lakukan pengubahan pada kondisi Tombol dan Potensio di rangkaian U1 dan U2 di Proteus. Amati grafik Waveform Chart di Tab Real Time Trend, dan juga tampilan nilai di ketujuh String Indicator.

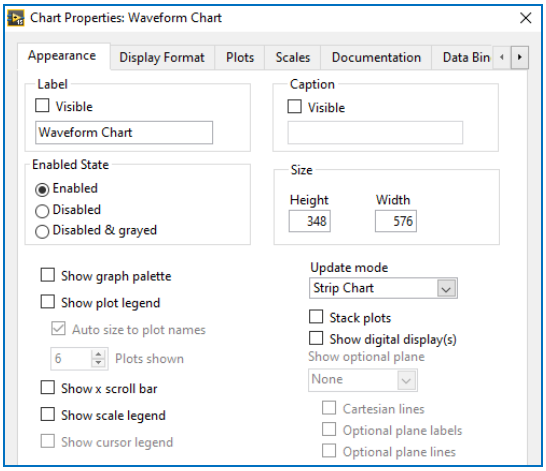


Gambar 5.98 Di Case "Real Time Trend", Waveform Chart mendapatkan 6 buah data terkini, atau data di baris terakhir, yang masing-masing datanya dapat diatur kemunculannya dengan tombol, begitu pula 7 String Indicator juga mendapatkan data terkini



Gambar 5.99 Jalankan OPC Quick Client, rangkaian Proteus, Enable LogGroup4, dan jalankan program LabVIEW. Lakukan pengubahan kondisi Tombol dan Potensio di rangkaian U1 dan U2 Proteus, kemudian lihat tampilan Waveform Chart dan ketujuh String Indicator.

48. Hal yang menarik di LabVIEW, ketika menggunakan Waveform Chart dan Waveform Graph adalah tersedianya fitur yang cukup lengkap dalam mengatur tampilan grafik. Untuk melihat fitur pengaturan tersebut, klik kanan Waveform tersebut, pilih Properties, dan pilih Tab Appearance.

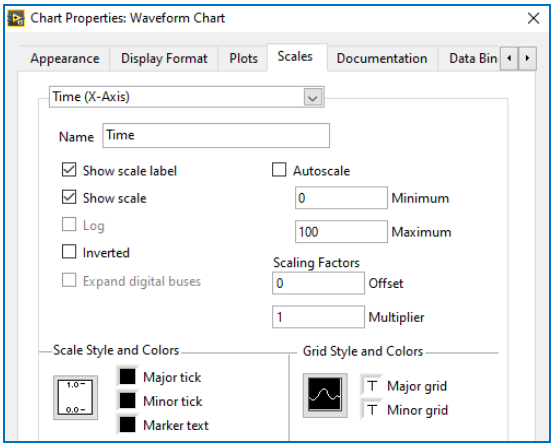


Gambar 5.100 Di Properties, di Tab Appearance, tersedia berbagai seting tampilan

49. Di Tab Appearance, terlihat berbagai seting yang bisa ditambahkan pada tampilan grafik. Mulai dari Update Mode, ada 3 pilihan, yaitu Strip Chart, Scope Chart dan Sweep Chart. Default Update Mode adalah Strip Chart. Ubah ke Scope atau Sweep Chart, dan perhatikan perubahan pada grafik.

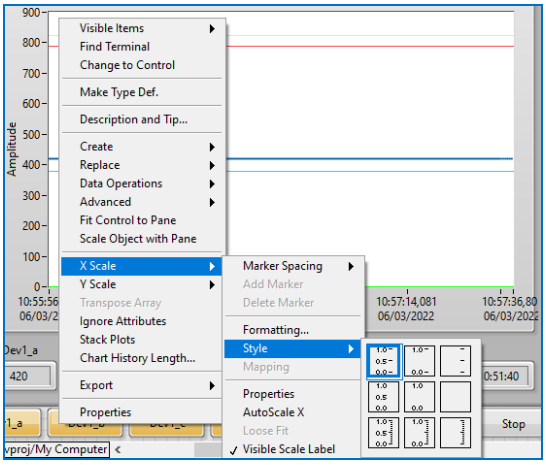
Catatan: Mode Strip Chart akan menampilkan grafik seolah seperti grafik pada pencatat gempa, yang menggeser tampilan grafiknya dari kanan ke kiri. Mode Scope Chart akan menampilkan grafik seolah seperti grafik pada osiloskop, yaitu grafik akan berjalan dari kiri ke kanan, kemudian setelah mencapai batas di kanan, grafik akan direfresh untuk mulai lagi dari kiri. Mode Sweep Chart akan menampilkan grafik seolah seperti grafik pada penampil EKG (Elektrocardiogram), yaitu mirip seperti grafik pada osiloskop, namun ketika grafik sudah mencapai batas kanan, grafik yang ada tidak dihilangkan, tetapi ditumpuk dengan grafik yang baru, dengan penanda sebuah garis melintang tegak (biasanya berwarna merah), yang berjalan dari kiri ke kanan, untuk menandai grafik, di mana grafik di kiri garis adalah grafik yang baru, sedangkan grafik di kanan garis adalah grafik yang lama.

50. Kembalikan Update Mode ke Strip Chart. Agar tampilan grafik tidak terlalu cepat bergeser, buat pengaturan di Properties, di Tab Scales, di Time (X-Axis), di bawah Autoscale, isi Minimum = 0, dan Maximum = 100.



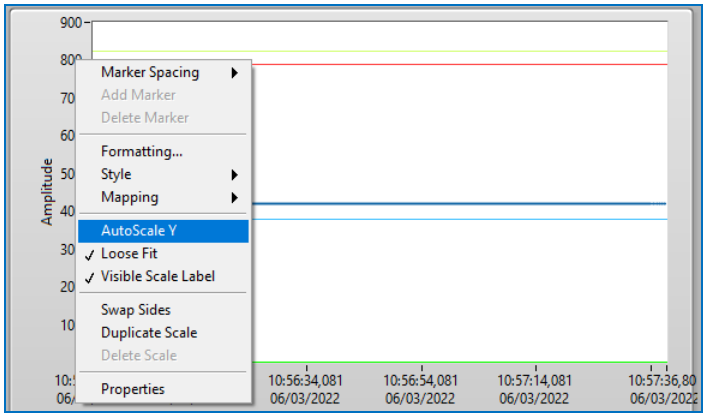
Gambar 5.101 Agar pergeseran grafik tidak terlalu cepat, atur nilai Minimum dan Maximum Time (X-Axis) di Tab Scales, di Properties grafik, isi dengan nilai 0 dan 100

51. Agar sumbu X menampilkan lebih banyak nilai waktu, klik kanan grafik, pilih X scale, pilih Style, pilih pilihan pertama, seperti gambar berikut.



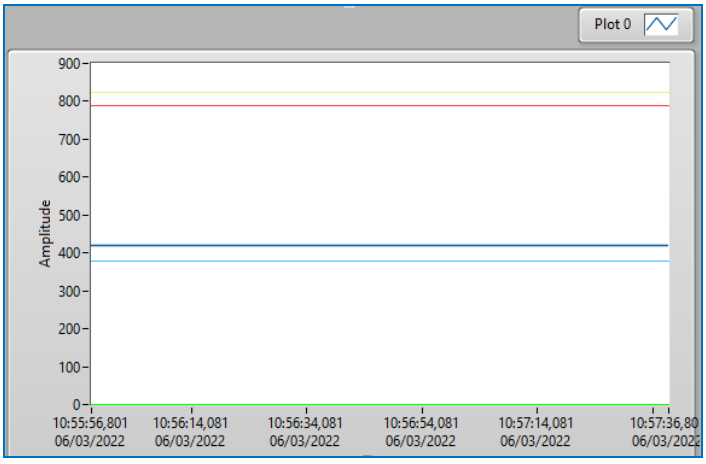
Gambar 5.102 Klik kanan grafik, pilih X Scale, pilih Style, pilih pilihan pertama

52. Agar sumbu Y tidak selalu berubah nilainya mengikuti data grafik, klik kanan sumbu Y, hilangkan tanda centang pada AutoScale Y.



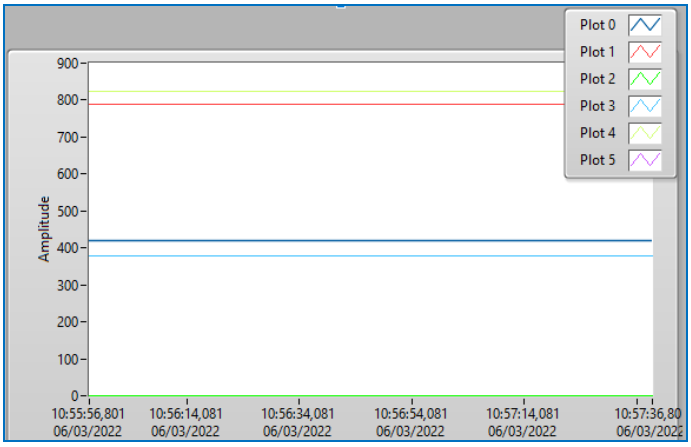
Gambar 5.103 Klik kanan sumbu Y, hilangkan centang pada AutoScale Y

53. Agar grafik terlihat lebih jelas, diinginkan untuk membuat satu grafik satu Plot. LabVIEW menyediakan fitur Stack Plots, yang bisa membuat 1 Waveform Chart menjadi banyak Plot yang disusun dari atas ke bawah, dengan jumlah Plot menyesuaikan jumlah Plot pada Plot Legend. Untuk itu, klik kanan grafik, pilih Visible Items, beri centang pada Plot Legend, maka akan muncul kotak Plot Legend seperti gambar berikut.



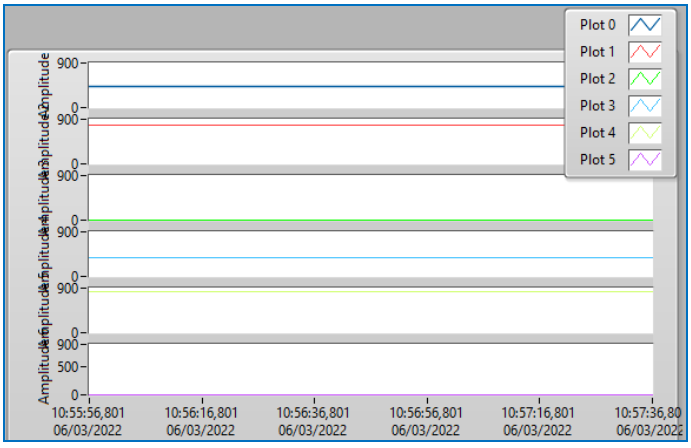
Gambar 5.104 Klik kanan grafik, pilih Visible Items, centang Plot Legend, muncul Plot Legend

54. Kemudian tarik ke bawah kotak Plot Legend, hingga muncul 6 buah Plot (Plot0 – Plot5). Dibuat 6 agar sesuai dengan grafik yang ingin ditampilkan.



Gambar 5.105 Tarik ke bawah Plot Legend hingga muncul 6 buah Plot: Plot0 sampai Plot5

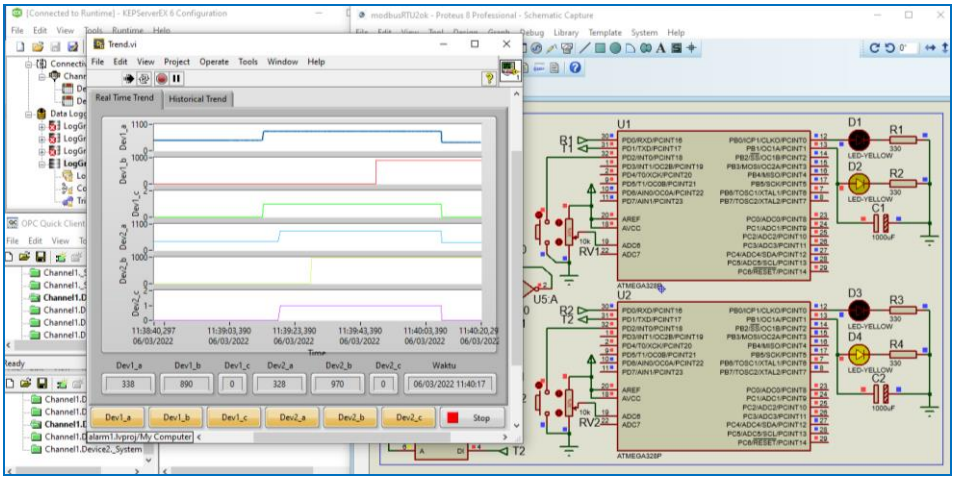
55. Berikutnya, klik kanan pada grafik, pilih Stack Plots, maka akan muncul 6 buah Plot yang bersusun dari atas ke bawah seperti gambar berikut.



Gambar 5.106 Setelah Plot Legend menampilkan 6 Plot, klik kanan grafik, pilih Stack Plots, maka Waveform Chart menampilkan 6 buah Plot yang bersusun dari atas ke bawah

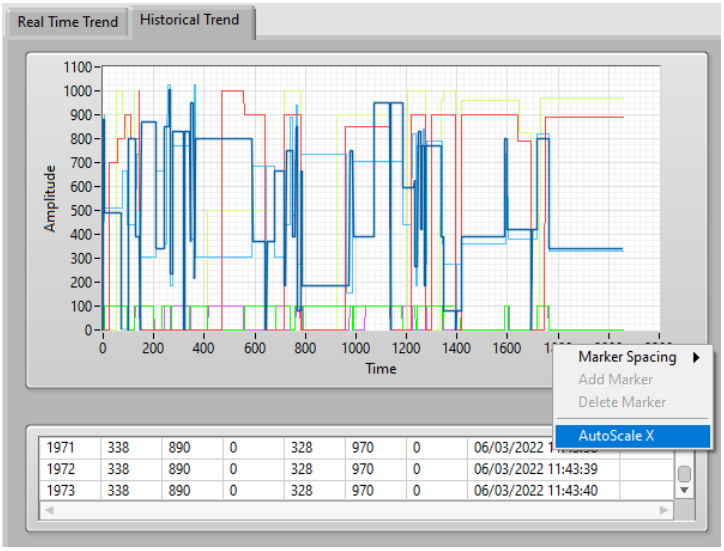
56. Ubah label sumbu Y di setiap Plot, satu persatu dari atas ke bawah, dari Amplitude menjadi Dev1_a, Dev1_b, Dev1_c, Dev2_a, Dev2_b, Dev2_c.
57. Ubah nilai maksimum sumbu Y di setiap Plot, satu persatu dari atas ke bawah, dari 900 menjadi 1100, 1000, 2, 1100, 1000, 2.
58. Hilangkan Plot Legend, dengan meng-klik Plot Legend, pilih Visible Items, hilangkan tanda centang pada Plot Legend.

59. Jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan nilai keenam Tag. Kemudian Enable LogGroup4, dan jalankan program LabVIEW penampil grafik (Trend) ini. Lakukan pengubahan kondisi Tombol dan Potensio di rangkaian U1 dan U2 Proteus, tekan keenam tombol di samping kiri Tombol Stop dan perhatikan tampilan grafik di Waveform Chart di Tab Real Time Trend, dan juga data di ketujuh String Indicator (Dev1_a sampai Waktu).

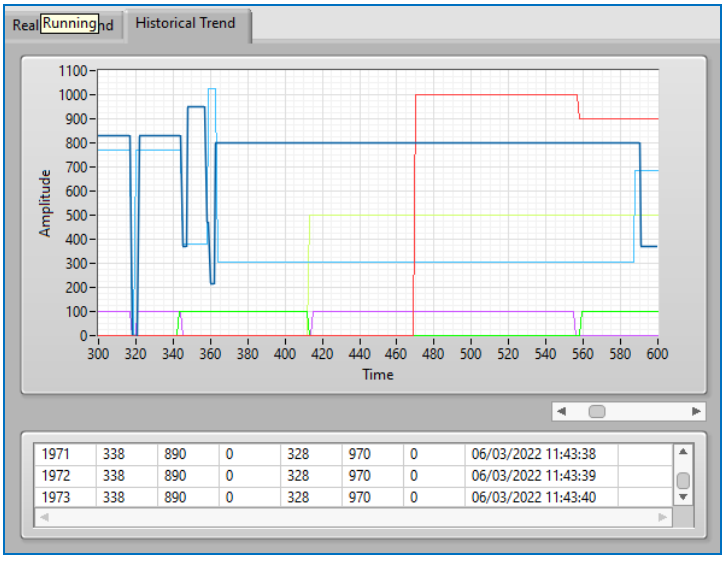


Gambar 5.107 Jalankan OPC Quick Client, rangkaian Proteus, Enable LogGroup4, dan program LabVIEW Trend. Tekan keenam tombol di kiri Tombol Stop, dan perhatikan keenam Plot di Waveform Chart di Tab Real Time Trend, dan juga perhatikan isi ketujuh String Indicator

60. Fitur grafik di LabVIEW yang menarik berikutnya adalah X scrollbar, yang dapat digunakan untuk menggeser tampilan grafik di sumbu X, sehingga dapat melihat grafik dengan lebih jelas. Munculkan fitur X scrollbar ini di Waveform Graph, di Tab Historical Trend.
61. Klik kanan Waveform Graph, pilih Visible Items, pilih X scrollbar, maka akan muncul sebuah Scrollbar di bawah Waveform Graph. Agar X scrollbar ini dapat berfungsi, klik kanan sumbu X, hilangkan tanda centang pada Autoscale X. Kemudian ubah nilai minimum dan nilai maksimum sumbu X dengan interval yang dekat. Dalam contoh Gambar 5.109, nilai interval dibuat 300, dari total 1973 baris data. Geser X scrollbar untuk melihat grafik dari 0 sampai 1973 dengan interval lebar grafik 300.



Gambar 5.108 Hilangkan tanda centang pada AutoScale X, agar nilai minimum dan maksimum Waveform Graph dapat diubah, sehingga X scrollbar dapat difungsikan



Gambar 5.109 Dengan X scrollbar, dan mengatur selisih nilai minimum dan maksimum = 300, dari total 1973 baris data, grafik dapat dilihat dari nilai 0 sampai 1973 dengan lebih jelas

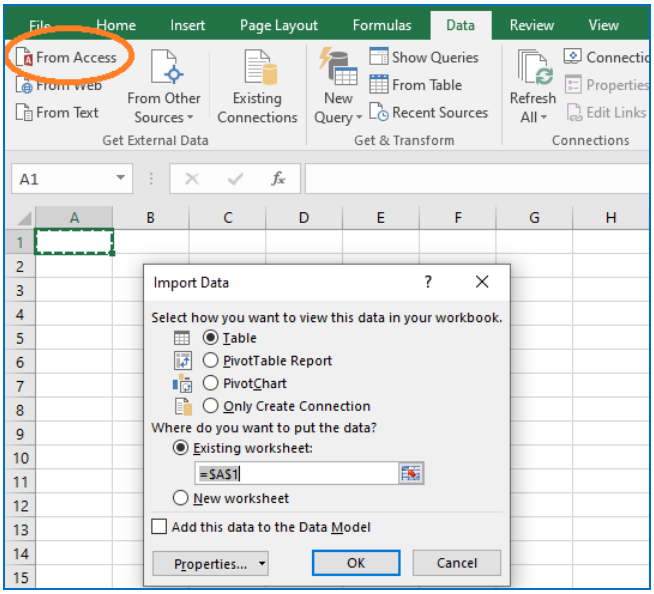
62. Sampai di sini pembuatan Trend yang menampilkan grafik Real Time dan Historical dengan bantuan Data Logger dan LabVIEW telah selesai.

5.5 Report dengan Excel dan LabVIEW

Report adalah laporan, yang bisa membuat orang menjadi “repot”. Apalagi Report ini harus dibuat secara rutin dan berkala, bisa harian, mingguan atau bulanan. Fitur SCADA memungkinkan pembuatan Report dengan hanya meng-klik sebuah tombol. Semakin mudah membuat Report, tentu akan semakin banyak menghemat waktu.

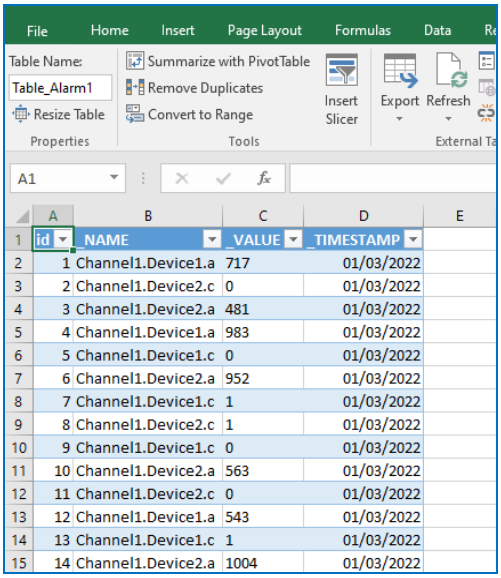
Microsoft Excel dipilih untuk pembuatan Report karena memungkinkan pengolahan data yang lebih mudah bila dibandingkan Microsoft Word. Berikut ini akan disajikan pembuatan Report dengan Excel, dan dilanjutkan dengan penambahan LabVIEW yang memungkinkan pembuatan Report di Excel dengan hanya menekan sebuah tombol. Diinginkan untuk membuat Report dari database Alarm dan Trend. Berikut langkah-langkah pembuatan Report dengan Excel:

1. Buka Microsoft Excel.
2. Pada menu Data, tekan tombol From Access (ada di pojok kiri atas).
3. Muncul kotak Select Data Source, arahkan ke lokasi file database Alarm.
4. Muncul kotak Import Data, pilih Table, dan pilih lokasi penempatan.



Gambar 5.110 Kotak Import Data, pilih Table dan pilih lokasi penempatan

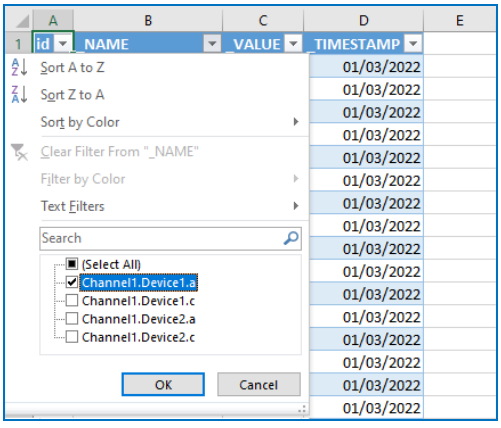
5. Excel menampilkan data dari database. Hal yang menarik di sini, setiap kolom data memiliki tombol Drop-down, yang jika ditekan, menampilkan daftar kategori data, yang bisa dipilih mana yang akan ditampilkan.



	A	B	C	D	E
1	id	_NAME	_VALUE	_TIMESTAMP	
2	1	Channel1.Device1.a	717	01/03/2022	
3	2	Channel1.Device2.c	0	01/03/2022	
4	3	Channel1.Device2.a	481	01/03/2022	
5	4	Channel1.Device1.a	983	01/03/2022	
6	5	Channel1.Device1.c	0	01/03/2022	
7	6	Channel1.Device2.a	952	01/03/2022	
8	7	Channel1.Device1.c	1	01/03/2022	
9	8	Channel1.Device2.c	1	01/03/2022	
10	9	Channel1.Device1.c	0	01/03/2022	
11	10	Channel1.Device2.a	563	01/03/2022	
12	11	Channel1.Device2.c	0	01/03/2022	
13	12	Channel1.Device1.a	543	01/03/2022	
14	13	Channel1.Device1.c	1	01/03/2022	
15	14	Channel1.Device2.a	1004	01/03/2022	

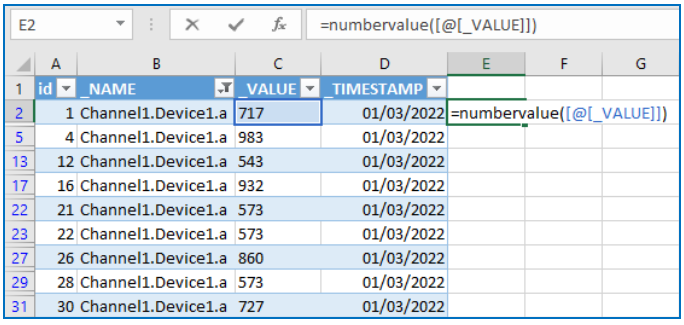
Gambar 5.111 Excel menampilkan data yang diimport dari database

6. Sebagai contoh, ingin ditampilkan data dari Channel1.Device1.a, dan dibuat grafik datanya. Untuk itu, klik tombol Drop-down di kolom NAME, beri tanda centang hanya pada pilihan Channel1.Device1.a. Klik OK.



Gambar 5.112 Tekan tombol Drop-down NAME, pilih Channel1.Device1.a

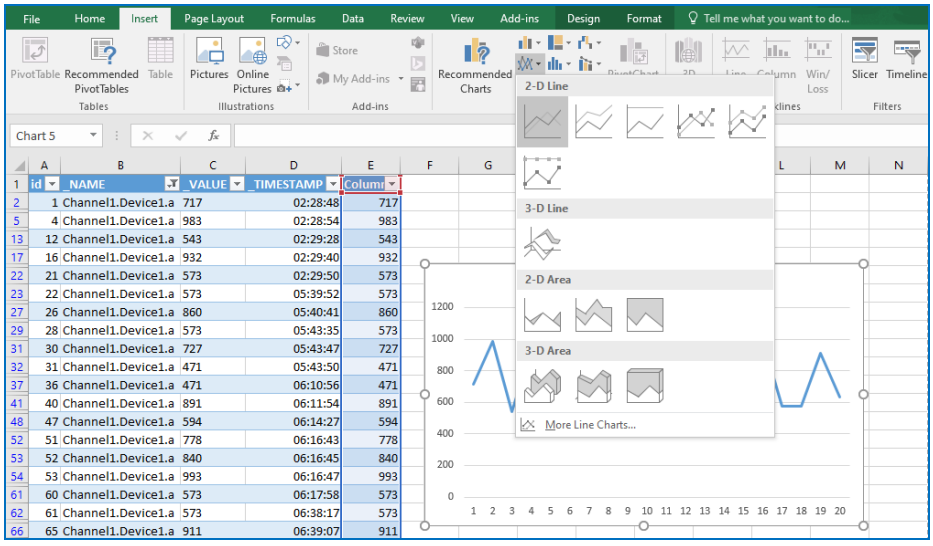
7. Karena grafik hanya bisa dibuat apabila datanya bertipe angka, bukan teks, maka perlu mengubah data teks yang ada di kolom VALUE menjadi angka. Untuk itu, di kolom yang kosong, isikan fungsi =numbervalue, diikuti kurung buka, kemudian letakkan kursor di cell yang akan diubah menjadi angka, kemudian beri kurung tutup. Tekan Enter, maka dari baris atas hingga baris terakhir akan terisi nilai angka dari teks di kolom VALUE.



	A	B	C	D	E	F	G
1	id	_NAME	_VALUE	_TIMESTAMP			
2	1	Channel1.Device1.a	717	01/03/2022	=numbervalue([@[_VALUE]])		
5	4	Channel1.Device1.a	983	01/03/2022			
13	12	Channel1.Device1.a	543	01/03/2022			
17	16	Channel1.Device1.a	932	01/03/2022			
22	21	Channel1.Device1.a	573	01/03/2022			
23	22	Channel1.Device1.a	573	01/03/2022			
27	26	Channel1.Device1.a	860	01/03/2022			
29	28	Channel1.Device1.a	573	01/03/2022			
31	30	Channel1.Device1.a	727	01/03/2022			

Gambar 5.113 Gunakan fungsi =numbervalue(terisi cell di mana kursor ditempatkan)

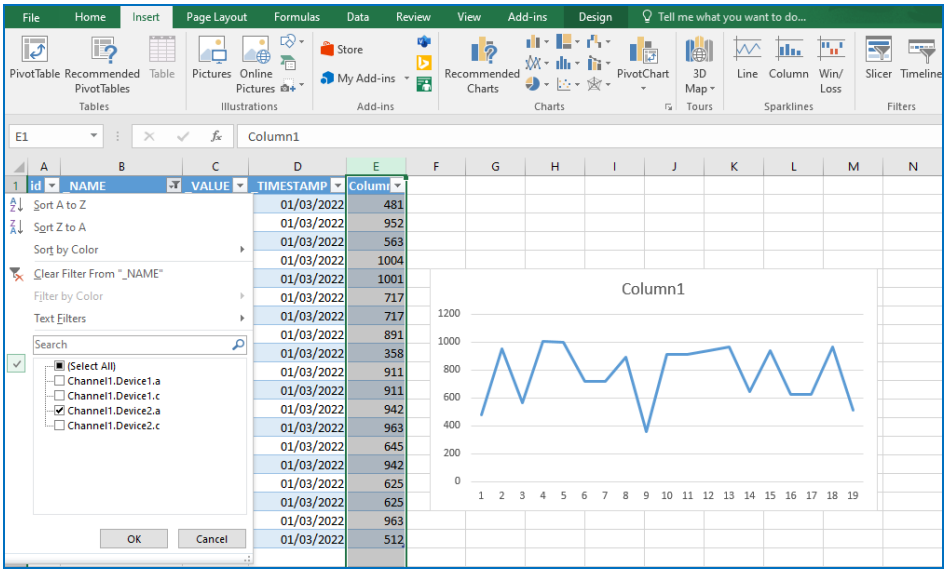
8. Klik pada kolom yang berisi angka hasil pengubahan teks, kemudian buka menu Insert, klik pada tombol Insert Line di kategori Chart, pilih 2-D Line, maka akan muncul grafik nilai Channel1.Device1.a.



	A	B	C	D	E
1	id	_NAME	_VALUE	_TIMESTAMP	Column
2	1	Channel1.Device1.a	717	02:28:48	717
5	4	Channel1.Device1.a	983	02:28:54	983
13	12	Channel1.Device1.a	543	02:29:28	543
17	16	Channel1.Device1.a	932	02:29:40	932
22	21	Channel1.Device1.a	573	02:29:50	573
23	22	Channel1.Device1.a	573	05:39:52	573
27	26	Channel1.Device1.a	860	05:40:41	860
29	28	Channel1.Device1.a	573	05:43:35	573
31	30	Channel1.Device1.a	727	05:43:47	727
32	31	Channel1.Device1.a	471	05:43:50	471
37	36	Channel1.Device1.a	471	06:10:56	471
41	40	Channel1.Device1.a	891	06:11:54	891
48	47	Channel1.Device1.a	594	06:14:27	594
52	51	Channel1.Device1.a	778	06:16:43	778
53	52	Channel1.Device1.a	840	06:16:45	840
54	53	Channel1.Device1.a	993	06:16:47	993
61	60	Channel1.Device1.a	573	06:17:58	573
62	61	Channel1.Device1.a	573	06:38:17	573
66	65	Channel1.Device1.a	911	06:39:07	911

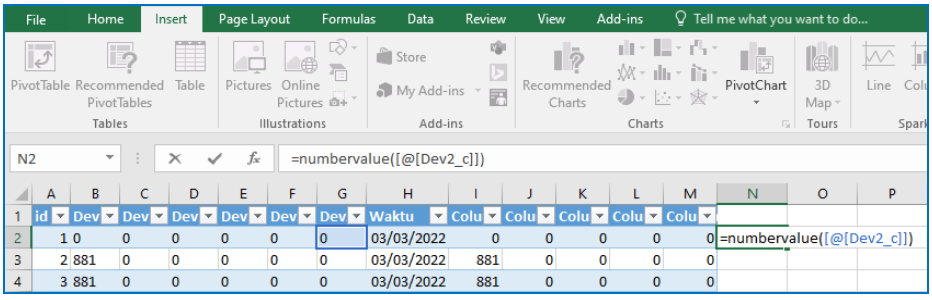
Gambar 5.114 Pilih kolom angka, di menu Insert, klik tombol Insert Line, pilih 2-D Line

9. Untuk mendapatkan grafik Alarm pada Tag yang lain, tekan tombol Drop-down NAME, ubah lokasi tanda centang ke nama Tag yang lain satu persatu, maka Report Alarm untuk semua Tag dapat diperoleh.



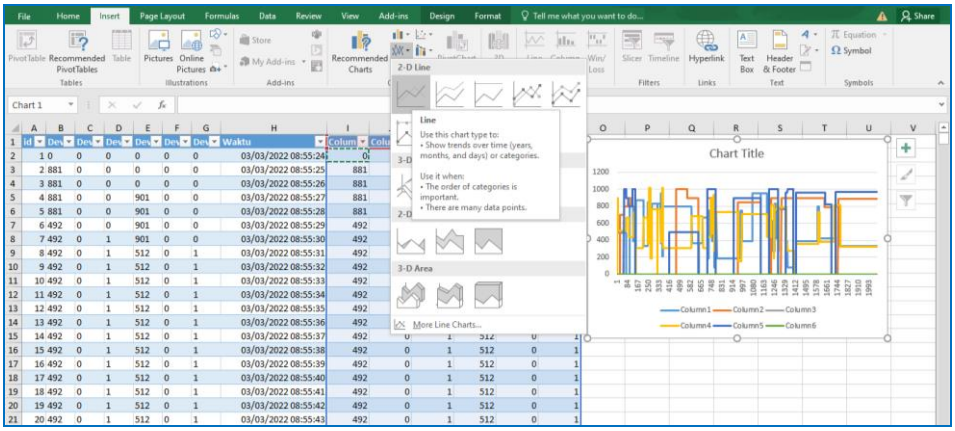
Gambar 5.115 Ubah nama Tag di kolom NAME, maka otomatis grafik berubah sesuai data

10. Sama seperti pada database Alarm, untuk database Trend, lakukan Import Data dengan memilih menu Data, tekan tombol From Access, arahkan ke lokasi file database Trend, klik Open, dan di kotak Import Data, pilih Table.
11. Setelah data muncul di Excel, ubah keenam kolom data Tag yang bertipe teks (tidak termasuk kolom id dan Waktu), menjadi angka dengan fungsi NUMBERVALUE, satu demi satu kolom.



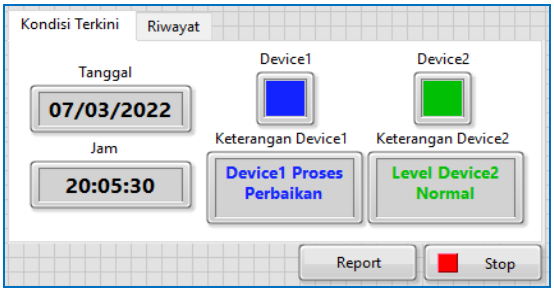
Gambar 5.116 Ubah semua data teks menjadi angka dengan fungsi numbervalue

12. Pilih keenam kolom tersebut, buka menu Insert, tekan tombol Insert Line, pilih 2-D Line, maka akan muncul grafik keenam Tag dalam 1 Chart.



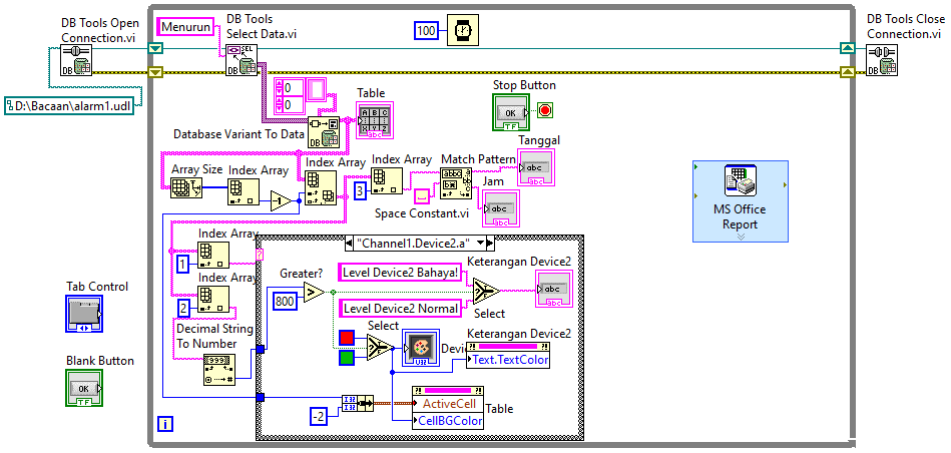
Gambar 5.117 Pilih keenam kolom angka, di menu Insert, klik tombol Insert Line, pilih 2-D Line

13. Untuk memperjelas grafik Trend, gunakan tombol Drop-down Waktu, pilih waktu tertentu, maka grafik akan terlihat lebih jelas.
14. Sampai di sini penggunaan Excel untuk pembuatan Report. Berikut ini, akan ditunjukkan cara pembuatan Report Excel yang lebih mudah, dengan hanya menekan sebuah tombol menggunakan program LabVIEW.
15. Buka kembali program LabVIEW Alarm.
16. Di Front Panel, ambil sebuah Blank Button dari Palet Controls, dari kategori Silver, Boolean, dan tempatkan di kiri tombol Stop.
17. Buat Labelnya menghilang (hilangkan centang Label di Visible Items), dan munculkan Boolean Text. Ubah nama Boolean Text menjadi Report.



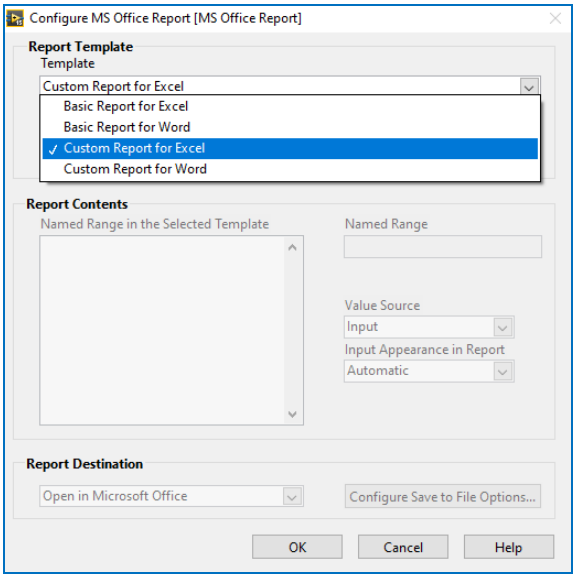
Gambar 5.118 Buka program LabVIEW Alarm, dan tambahkan sebuah tombol Report

18. Di Block Diagram, tempatkan icon MS Office Report di dalam While Loop, yang diambil dari kategori Programming, di Report Generation.



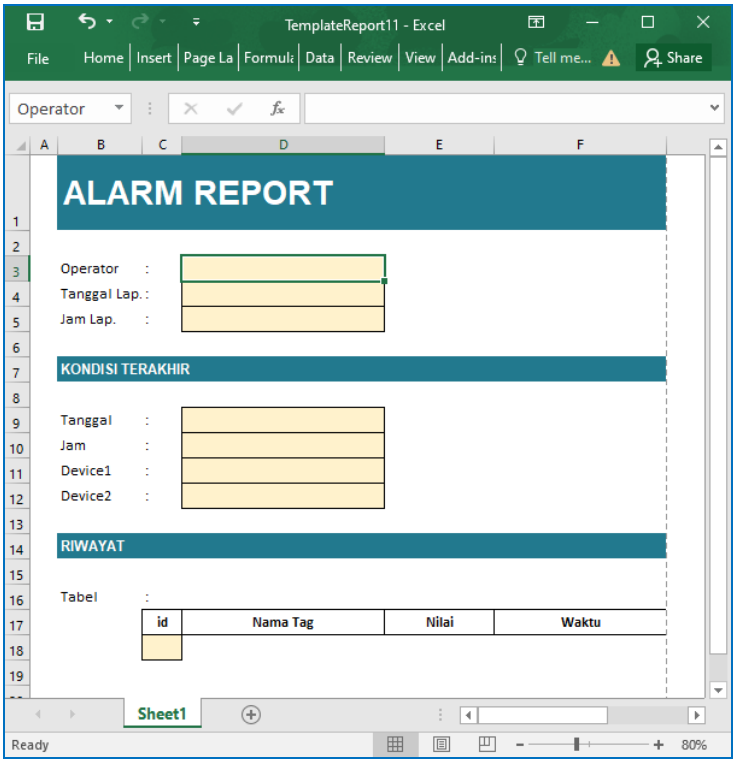
Gambar 5.119 Tempatkan icon MS Office Report di dalam While Loop, yang diambil dari Palet Functions, di kategori Programming, di kategori Report Generation

19. Muncul kotak Configure MS Office Report. Di isian Template, pilih Custom Report for Excel.



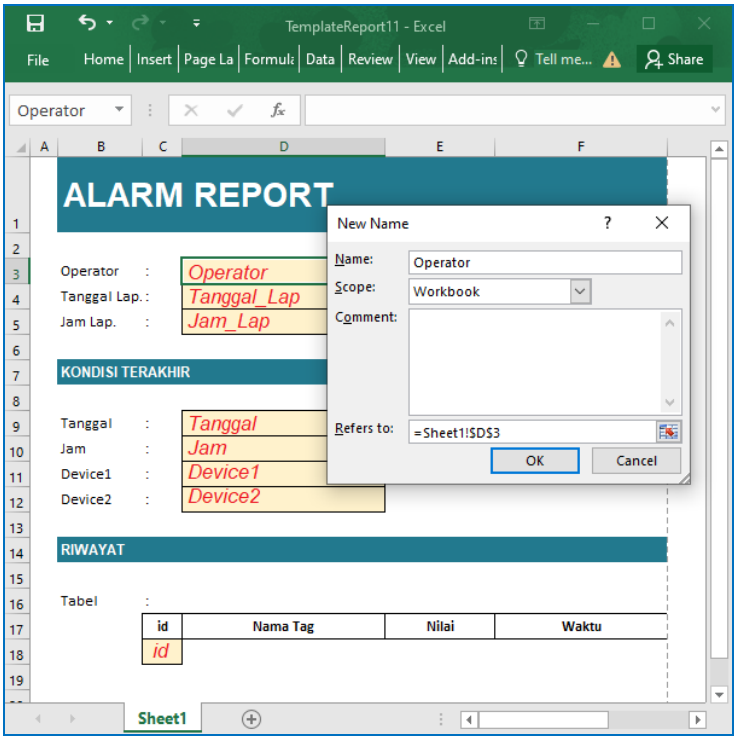
Gambar 5.120 Di kotak Configure MS Office Report, pilih Template: Custom Report for Excel

20. Di Path to Template, isi dengan lokasi file Template Excel. Untuk itu, perlu membuat Template Excel. Buka Excel, dan buat tampilan seperti berikut.



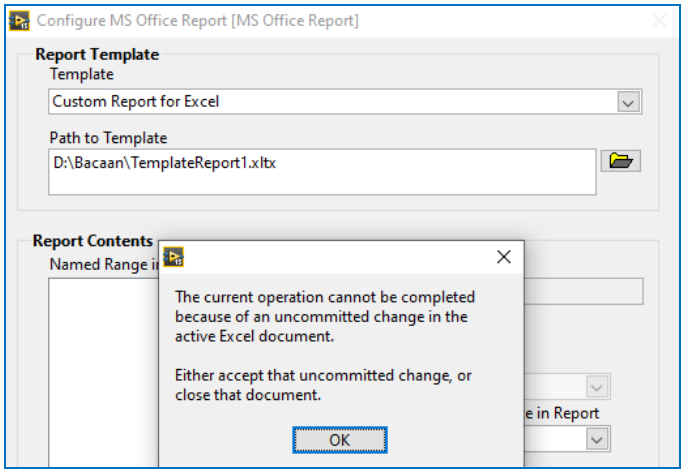
Gambar 5.121 Buat tampilan Template Excel seperti di atas

21. Agar data dari program LabVIEW Alarm bisa langsung diisikan ke cell tertentu, beri nama cell-cell tersebut dengan meng-klik kanan cell, pilih Define Name. Pada kotak New Name yang muncul, isi pada kolom Name sesuai dengan tulisan di sampingnya (atau atasnya), yaitu berturut-turut: Operator, Tanggal_Lap, Jam_Lap, Tanggal, Jam, Device1, Device2 dan id, seperti terlihat pada Gambar 5.122.
22. Setelah itu, simpan file Template tersebut dengan tipe Excel Template, yang memiliki ekstensi *.xltx, atau *.xlt.
23. Kemudian kembali lagi ke kotak Configure MS Office Report. Di kolom Path to Template, tekan tombol di samping kanan kolom, arahkan ke lokasi file Template. Klik OK.



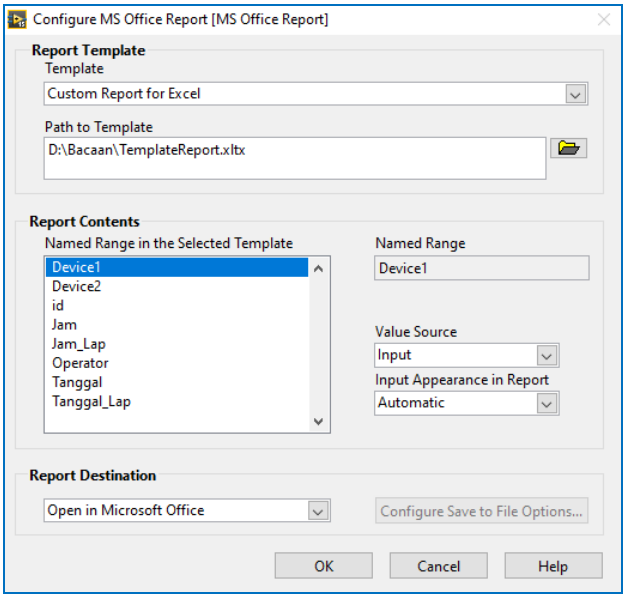
Gambar 5.122 Klik kanan cell yang diberi garis kotak, pilih Define Name, dan isi Name dengan tulisan berturut-turut: Operator, Tanggal_Lap, Jam_Lap, Tanggal, Jam, Device1, Device2, id

24. Apabila muncul pesan seperti gambar berikut ini, tutup file Template.



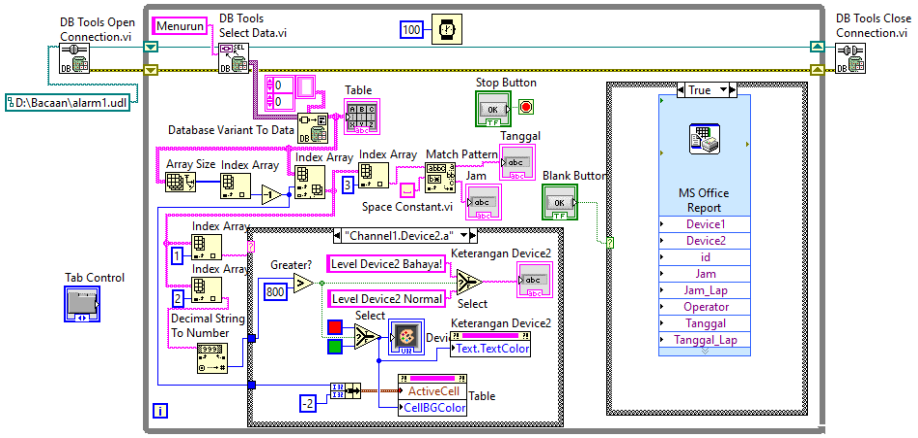
Gambar 5.123 Ketika file Template masih terbuka, muncul peringatan seperti gambar di atas

25. Setelah fiile Template ditutup, tekan kembali tombol di kanan kolom Path to Template, arahkan kembali ke file Template. Klik OK, maka di kolom Report Contents, muncul nama-nama yang telah di-Define di Template.



Gambar 5.124 Setelah file Template ditutup, tekan tombol di kolom Path to Template, arahkan ke lokasi file Template, klik OK, maka muncul nama-nama cell di Report Contents

26. Klik OK, maka icon MS Office Report akan memiliki kaki input dengan jumlah dan nama yang sama dengan isi Report Contents.
27. Tambahkan Struktur Case di dalam While Loop, dan masukkan icon MS Office Report ke dalam Struktur Case tersebut. Hubungkan input Struktur Case (terminal Case Selector) dengan Blank Button.
28. Tabel 5.5 menunjukkan hubungan antara objek di program Alarm dengan input MS Office Report. Mulai dari yang pertama, Device1 di MS Office Report terhubung dengan String Indicator Keterangan Device1. Agar garis data terlihat lebih rapi, gunakan Local Variable. Klik kanan icon Keterangan Device1, pilih Create, pilih Local Variable. Secara default, Local Variable Keterangan Device1 ini memiliki kaki di kiri (kaki input). Agar memiliki kaki di kanan (kaki output), klik kanan, pilih Change to Read. Kemudian hubungkan kaki output ini dengan Device1 MS Office Report.

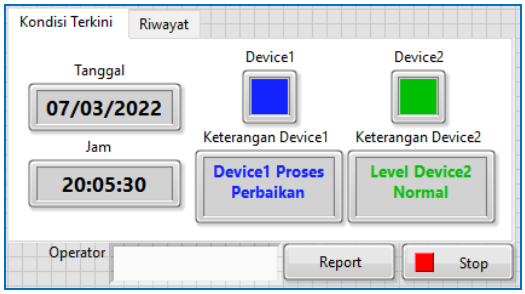


Gambar 5.125 Icon MS Office Report memiliki kaki-kaki input sesuai isi Report Contents. Tambahkan Struktur Case di dalam While Loop, masukkan icon MS Office Report ke dalam Case True, dan hubungkan input Struktur Case (terminal ?) dengan Blank Button

Tabel 5.5 Hubungan objek di program Alarm dengan input MS Office Report

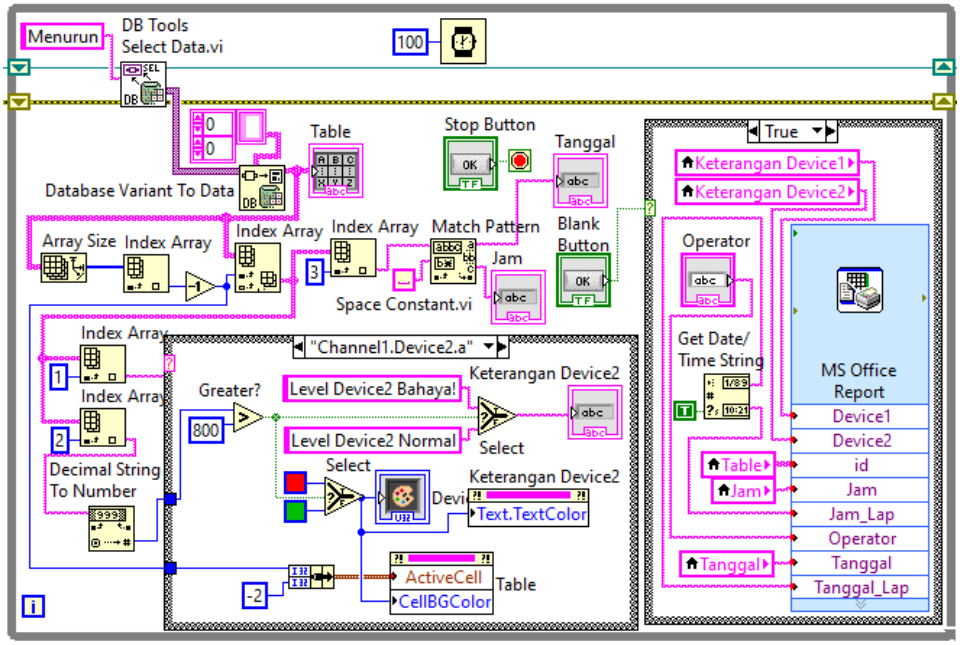
No.	Input MS Office Report	Objek di Program Alarm
1.	Device1	String Indicator Keterangan Device1
2.	Device2	String Indicator Keterangan Device2
3.	id	Table
4.	Tanggal	String Indicator Tanggal
5.	Jam	String Indicator Jam
6.	Operator	String Control Operator
7.	Tanggal_Lap	Tanggal saat laporan dibuat
8.	Jam_Lap	Jam saat laporan dibuat

29. Ulangi hal yang sama untuk String Indicator Keterangan Device2, Table, Tanggal dan Jam. Buat Local Variable untuk keempat objek tersebut, dan ubah kaki input menjadi kaki output, kemudian hubungkan Local Variable tersebut secara berturut-turut ke input Device2, id, Tanggal dan Jam.
30. Tambahkan sebuah String Control di Front Panel, beri nama Operator. Hubungkan icon String Control Operator ini dengan input Operator.



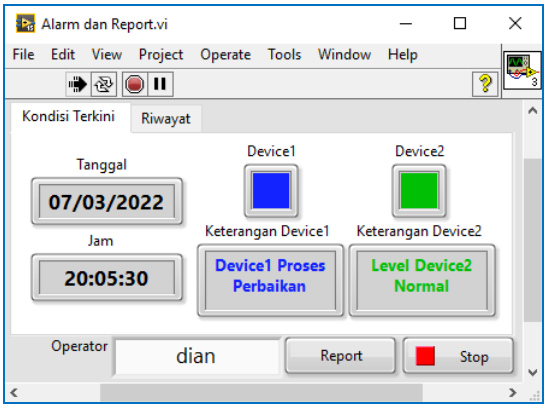
Gambar 5.126 Tambahkan sebuah String Control di Front Panel, beri nama label Operator

31. Berikutnya, untuk input Tanggal_Lap dan Jam_Lap, hubungkan dengan icon Get Date/Time String, yang diambil dari kategori Timing di Palet Functions. Untuk memunculkan detik pada fungsi tersebut, beri nilai True di kaki input want seconds? (?s). Lebih jelasnya lihat gambar berikut ini.

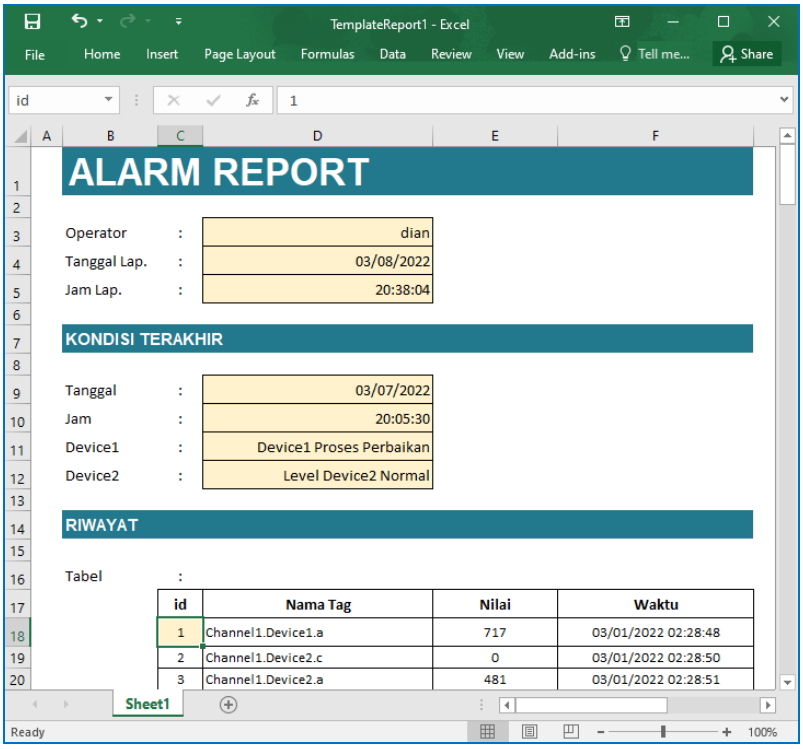


Gambar 5.127 Program Alarm dengan tambahan pembuatan Report di Excel

32. Jalankan program LabVIEW Alarm dan Report ini. Di kotak Operator, isi nama Operator. Kemudian tekan tombol Report, maka sebuah Report di Excel secara otomatis muncul.



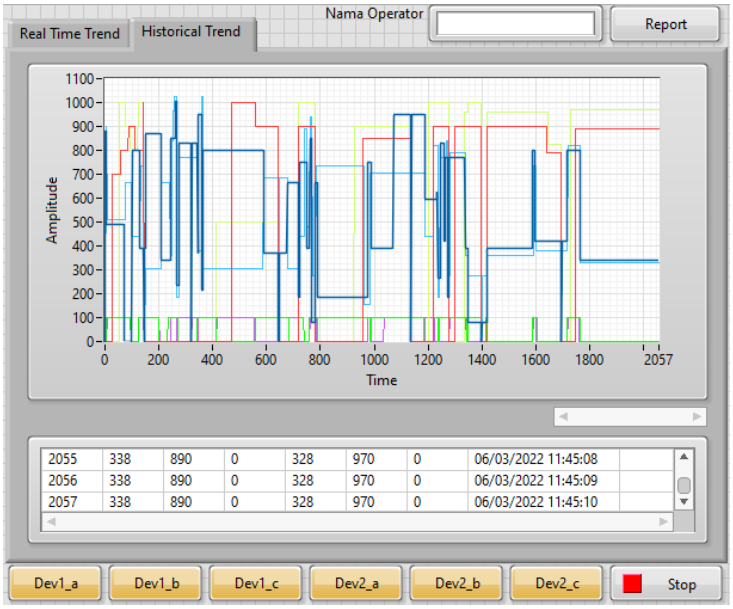
Gambar 5.128 Jalankan program LabVIEW, isi nama Operator, dan tekan tombol Report



Gambar 5.129 Muncul Report di Excel yang telah terisi data secara lengkap

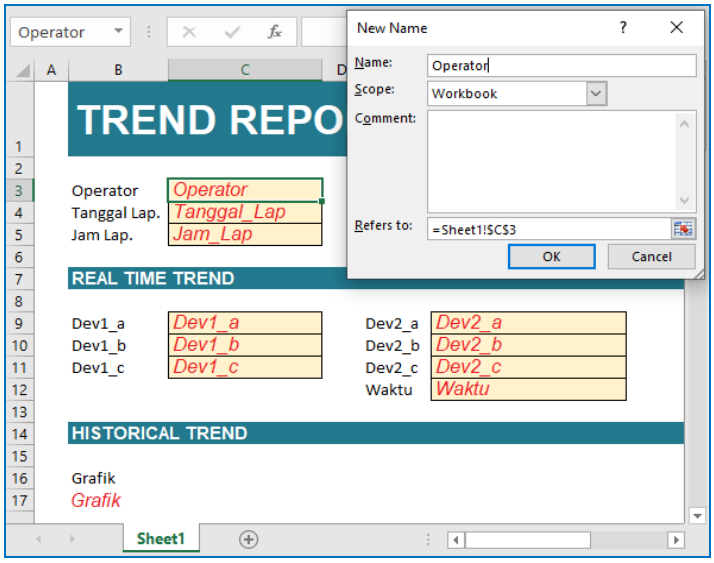
33. Ulangi pembuatan Report di atas untuk program LabVIEW Trend.
34. Buka program LabVIEW Trend.

35. Di Front Panel, tambahkan objek String Control dan tombol Report. Beri label String Control Nama Operator.



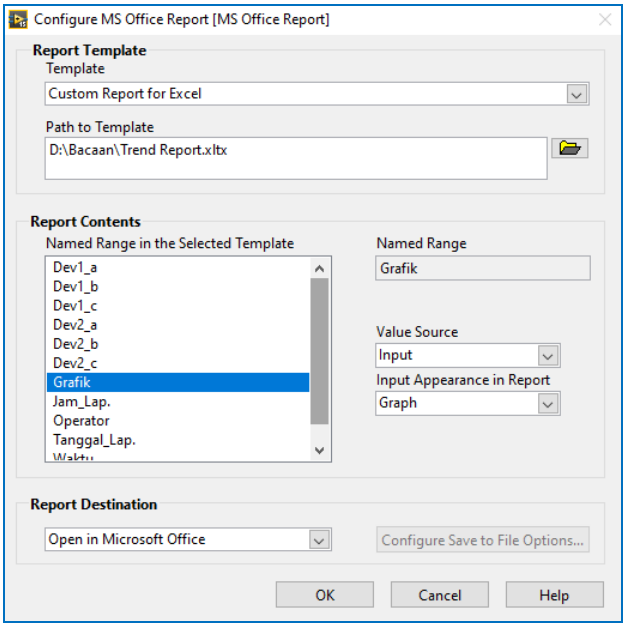
Gambar 5.130 Di Front Panel program Trend, tambahkan String Control dan Tombol Report

36. Di Block Diagram, tempatkan sebuah Struktur Case, dan hubungkan input Struktur Case (terminal Case Selector) dengan tombol Report.
37. Ambil icon MS Office Report dari Palet Function, di kategori Programming, di Report Generation, dan tempatkan di dalam Struktur Case yang baru.
38. Muncul kotak Configure MS Office Report. Di kolom Template, pilih Custom Report for Excel.
39. Untuk kolom Path to Template, perlu membuat dulu Template Excel. Untuk itu buka Excel, dan buat file Template Excel seperti Gambar 5.131.
40. Beri nama cell-cell di dalam garis kotak (kecuali cell Grafik) dengan meng-klik kanan cell, pilih Define Name, dan namai seperti Gambar 5.131. Ada 11 nama cell, yaitu Operator, Tanggal_Lap, Jam_Lap, Dev1_a, Dev1_b, Dev1_c, Dev2_a, Dev2_b, Dev2_c, Waktu, dan Grafik.
41. Simpan file Template Excel tersebut dengan tipe Excel Template, yang memiliki ekstensi *.xltx atau *.xlt. Tutup file Template.



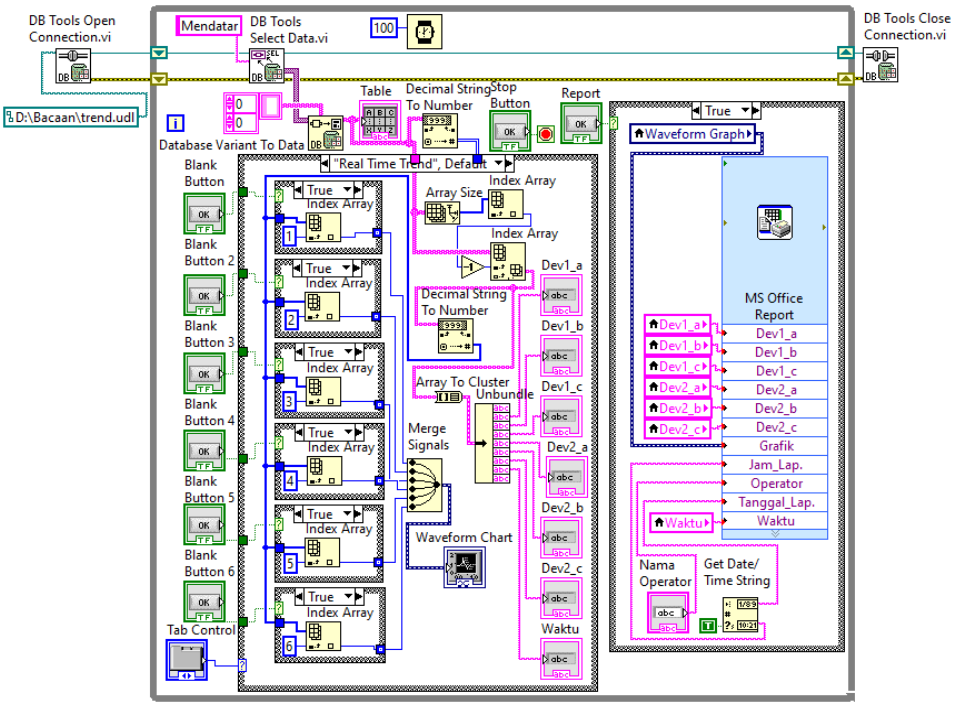
Gambar 5.131 Klik kanan cell yang ditandai garis kotak, pilih Define Name, isi sesuai gambar

42. Isikan lokasi file Template di kolom Path to Template. Klik OK, maka di kolom Report Contents akan muncul nama-nama cell yang dibuat.



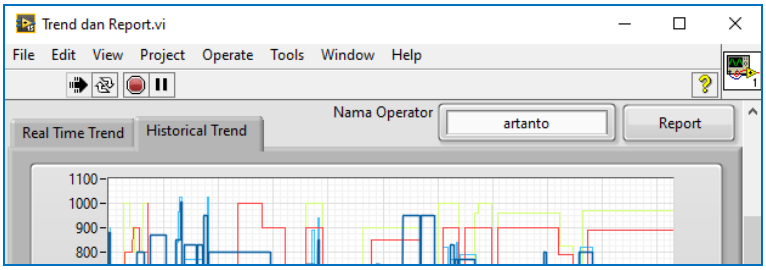
Gambar 5.132 Muncul nama-nama cell yang telah dibuat dengan Define Name di Template

43. Sebelum meng-klik OK pada kotak Configure MS Office Report Gambar 5.132 di atas, pilih nama Grafik di Report Contents, ubah isi kolom Input Appearance in Report, dari Automatic menjadi Graph. Karena bila tetap Automatic, yang muncul di Report bukan gambar grafik, tetapi tabel. Agar bisa muncul gambar grafiknya, maka harus dipilih Graph.
44. Setelah itu, tekan tombol OK, maka icon MS Office Report memiliki kaki input dengan jumlah dan nama yang sama dengan isi Report Contents.
45. Berikutnya, buat Local Variable untuk ketujuh icon String Indicator (Dev1_a, Dev1_b, Dev1_c, Dev2_a, Dev2_b, Dev2_c, Waktu), dan hubungkan dengan kaki input yang bernama sama di MS Office Report.
46. Untuk input Grafik, hubungkan dengan Local Variable Waveform Graph.
47. Untuk input Jam_Lap dan Tanggal_Lap, hubungkan dengan fungsi Get Date/ Time String. Beri input True pada kaki input want seconds.
48. Untuk input Operator, hubungkan dengan String Control Nama Operator.

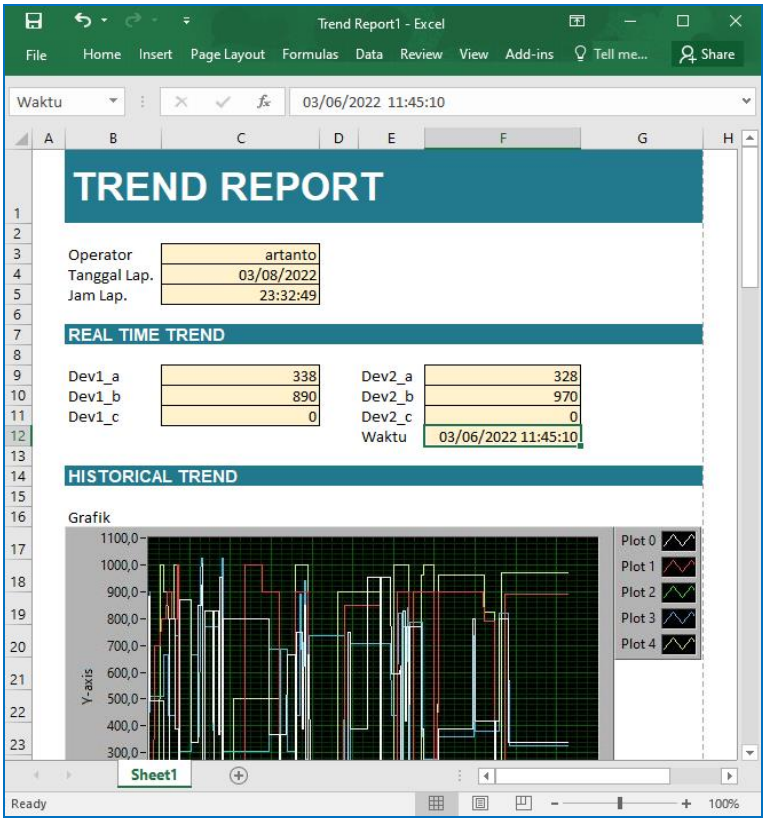


Gambar 5.133 Program Trend dengan tambahan Report menggunakan MS Office Report

49. Jalankan program LabVIEW Trend dengan tambahan program Report di atas. Di kotak Nama Operator, isi nama Operator. Kemudian tekan tombol Report, maka sebuah Report di Excel secara otomatis muncul.



Gambar 5.134 Jalankan program LabVIEW, isi Nama Operator, tekan tombol Report



Gambar 5.135 Muncul Report di Excel, dengan tampilan data teks dan grafik

50. Sampai di sini pembuatan Report dengan Excel dan LabVIEW selesai.

5.6 Soal Latihan

1. Apa fungsi Database di SCADA?
2. Apa yang Anda ketahui tentang fitur Alarm di SCADA? Apa saja jenis Alarm yang digunakan di SCADA dan fungsinya?
3. Apa yang Anda ketahui tentang fitur Trend di SCADA? Apa saja jenis Trend yang digunakan di SCADA dan fungsinya?
4. Ulangi pembuatan Database dengan Data Logger KEPServerEX seperti pada Sub Bab 5.2, namun dengan rangkaian Proteus seperti pada Gambar 3.38. Diinginkan 8 buah Tag (AUTO, SUPLAI, KATUP, ALARM, POMPA, ON, OFF dan LEVEL) di rangkaian U1 (Device1) dapat disimpan dalam Database setiap 10 detik sekali. Buat Table Format = Narrow.
5. Ulangi Soal Latihan 1 di atas, namun untuk rangkaian U2 (Device2), dengan Table Format = Wide.
6. Ulangi pembuatan program HMI Alarm menggunakan LabVIEW, yang akan menampilkan data terkini (Real Time) dan data Riwayat (Historical) untuk perubahan data pada Tag SUPLAI dan LEVEL di rangkaian U1 (Device1). Buat warna merah pada tampilan HMI apabila nilai LEVEL lebih dari 850, dan warna hijau apabila nilai LEVEL kurang dari 850. Buat warna tampillan HMI biru apabila indikator SUPLAI menyala, dan warna abu-abu apabila SUPLAI padam.
7. Ulangi Soal Latihan 3 di atas, namun untuk Tag SUPLAI dan LEVEL di rangkaian U1 dan U2, jadi ada 4 buah Tag yang ditampilkan perubahan datanya dan catatan waktunya oleh program HMI Alarm.
8. Ulangi pembuatan program HMI Trend menggunakan LabVIEW, yang akan menampilkan grafik data Real Time dan grafik data Historical untuk Tag SUPLAI dan LEVEL di rangkaian U1 dan U2.
9. Tambahkan tombol Report pada program HMI Alarm, yang bila ditekan, akan menampilkan Report Alarm di Excel, yang berisi data kondisi terakhir Alarm dan data Riwayatnya dalam bentuk tabel.

10. Tambahkan tombol Report pada program HMI Trend, yang bila ditekan, akan menampilkan Report Trend di Excel, yang berisi data Real Time Trend dan data Historical Trend dalam bentuk grafik.

5.7 Refleksi

1. Menurut Anda, dari isi yang diuraikan, apakah **Target Materi** dari Bab 5 buku ini tercapai? Jika belum tercapai, apakah ada kesulitan dalam memahami materi yang berkaitan dengan Target Materi tersebut? Apakah Anda memiliki saran dan masukan untuk memperbaiki materi tersebut?
2. Apakah **Tantangan** dalam Bab 5 buku ini dapat Anda selesaikan? Jika belum, kesulitan apa yang membuat Anda tidak bisa menyelesaikannya? Apakah Anda mencoba alternatif lain untuk menemukan sendiri cara penyelesaiannya (mencari di Google misalnya)?
3. Apakah ada **Manfaat** yang Anda dapatkan setelah mempelajari Bab 5 dari buku ini? Apakah ada yang ingin Anda pelajari lebih lanjut? Apakah ada **Ide** yang menarik yang ingin Anda kembangkan?

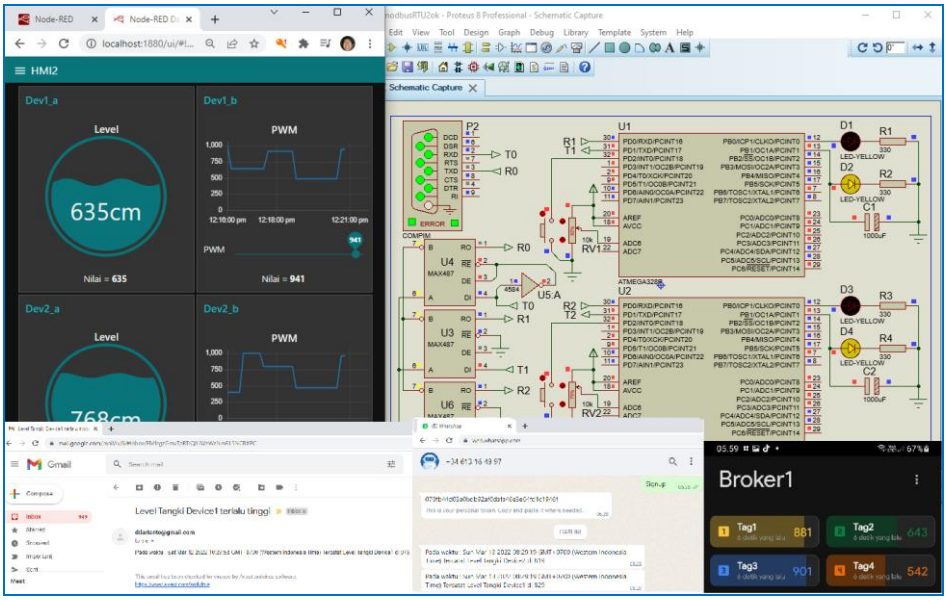
BAB 6
IoT GATEWAY

Target Materi:

- Menggunakan REST-Client dan Node-RED untuk mengirim Alarm.
- Menggunakan REST-Server dan Node-RED untuk kontrol data via HTTP
- Menggunakan MQTT Client dan Node-RED untuk kontrol data via TCP/IP

Tantangan:

➢ Buat program HMI yang dapat mengirimkan peringatan melalui email dan Whatsapp, memonitor dan mengontrol data Tag melalui HTTP dan TCP/IP.



Gambar 6.1 Program HMI Node-RED dengan IoT Gateway KEPServerEX, yang dapat mengirimkan alarm melalui email dan Whatsapp serta memonitor dan mengontrol data Tag

6.1 Persiapan Perangkat yang Diperlukan

Bab ini memerlukan tambahan perangkat lunak/software sebagai berikut:

1. Java JRE 32 bit
2. Node-RED

Software Java JRE 32 bit digunakan untuk menjalankan IoT Gateway KEPServerEX. Gunakan versi Java JRE yang 32 bit. Untuk pembaca yang menggunakan Windows, unduh software di: www.java.com/download/ie_manual.jsp, dan kemudian tekan tombol Agree and Start Free Download.

Software Node-RED digunakan untuk membuat program HMI dan sekaligus koneksinya dengan hardware dan layanan IoT. Untuk menginstal software Node-RED, harus menginstal dulu software Node.js. Untuk pembaca yang menggunakan Windows, pembaca dapat mengunduh software Node.js di: https://nodejs.org/en. Instalasi software Node.js ini perlu dijalankan dengan Run as Administrator. Setelah instalasi selesai, buka command prompt (cmd), ketikkan: **npm install -g --unsafe-perm node-red**. Agar instruksi tersebut bisa berjalan, komputer harus terhubung dengan internet. Instruksi tersebut akan mendownload software node-RED sekaligus menginstalnya. Setelah terinstal, software node-RED dapat dijalankan dengan mengetikkan: **node-red** di command prompt, akan muncul informasi mengenai node-red dan statusnya. Untuk membuka halaman program Node-RED, ketik **localhost:1880** di browser.

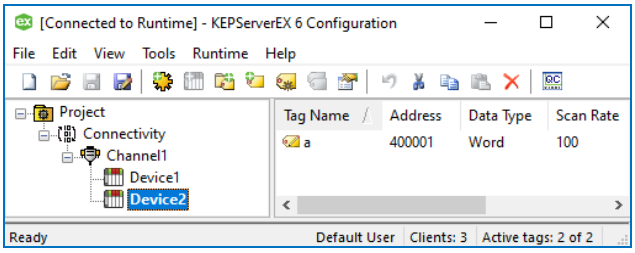
6.2 REST Client dan Node-RED

Bab terakhir dalam buku ini berkaitan dengan Internet of Things. Seperti pembaca ketahui, teknologi Internet of Things sudah banyak sekali diterapkan di sekitar kita. Mulai dari smartphone yang sudah banyak tersedia di mana-mana, hingga merambah ke benda-benda elektronik di sekitar kita (cctv, *smartwatch*, lampu, dll.), semua bisa terhubung dengan internet secara mudah. Tidak hanya di bidang elektronik saja, IoT juga telah diterapkan di banyak bidang. Apalagi di dunia industri, teknologi IIoT (*Industrial Internet of Things*) sudah menjadi keharusan bagi industri agar bisa berkompetisi dan bersaing di masa Revolusi Industri 4.0 ini.

Dengan berkembangnya teknologi IoT, software-software pemrograman untuk pembuatan aplikasi IoT cukup banyak tersedia di internet, dan salah satu software yang paling populer adalah Node-RED. Apa itu Node-RED? Node-RED adalah software pemrograman *open source* yang dapat menghubungkan perangkat keras, API, dan layanan online di internet, dengan pemrograman yang mudah.

Pemrograman di Node-RED mirip seperti pemrograman LabVIEW, yaitu hanya dengan menempatkan beberapa node dan menyambungkannya membentuk flow. Karena dukungan komunitas pengguna yang besar, saat ini telah tersedia lebih dari 225.000 node. Node yang tersedia tersebut termasuk node untuk komunikasi serial, node untuk komunikasi dengan protokol Modbus Serial, Modbus TCP, OPC DA, OPC UA, HTTP, MQTT dan layanan IoT lainnya. Di Sub Bab ini, akan dibahas penggunaan Node-RED, melibatkan REST Client KEPServerEX dan Whin untuk membuat pengiriman alarm ke email Gmail, dan Whatsapp, ketika nilai sensor melebihi nilai batasnya. Berikut ini langkah-langkah pembuatannya:

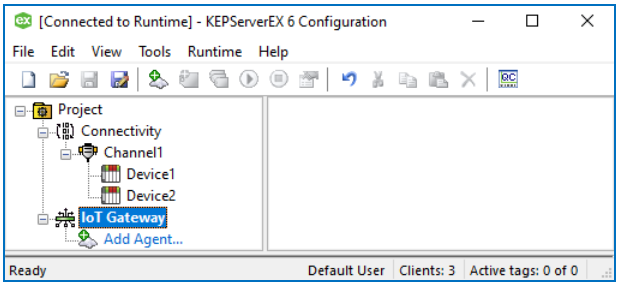
1. Buka KEPServerEX Configuration, buat sebuah Channel, dengan protokol Modbus RTU Serial, dengan 2 buah Device (ID = 1 dan ID = 2).
2. Buat masing-masing Device memiliki sebuah Tag, dengan alamat 400001.



Gambar 6.2 Sebuah Channel dengan 2 buah Device, 1 Tag di setiap Device, di alamat 400001

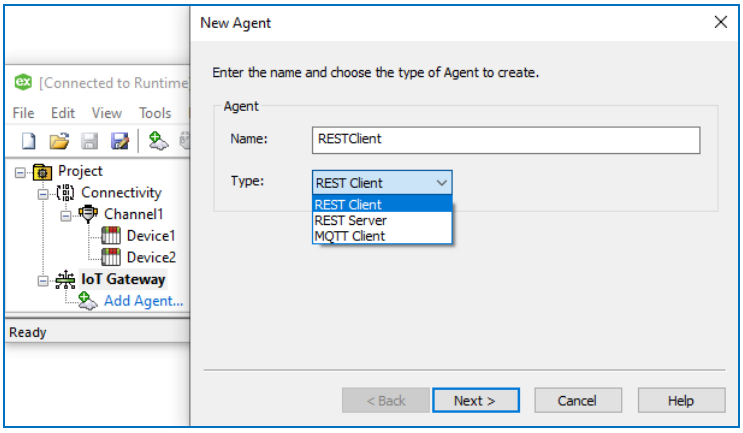
3. Sebagai perangkat Modbus RTU Slave untuk Device1 dan Device2, gunakan rangkaian 2 buah Arduino di Proteus seperti Gambar 4.51.
4. Gunakan program Outseal Gambar 4.52 untuk kedua Arduino. Buat alamat Modbus = 1 untuk U1, dan alamat Modbus = 2 untuk U2.
5. Jalankan OPC Quick Client untuk Device1 dan Device2, dan juga rangkaian Proteus. Pastikan bahwa Tag a (400001) di Device1 dan Device2 dapat menampilkan nilai Potensio di rangkaian U1 dan U2 Proteus.

6. Diinginkan nilai Tag a di Device1 dan Device2 dapat dipublikasikan datanya ke Cloud melalui HTTP. Untuk mempublikasikan data ke Cloud ini, KEPServerEX menyediakan IoT Gateway REST Client dan MQTT. Sub Bab ini hanya akan membahas REST Client, sedangkan untuk MQTT akan dibahas di Sub Bab selanjutnya. Untuk memunculkan IoT Gateway ini di KEPServerEX, buka menu View, beri centang pada IoT Gateway, maka muncul IoT Gateway di daftar Project.



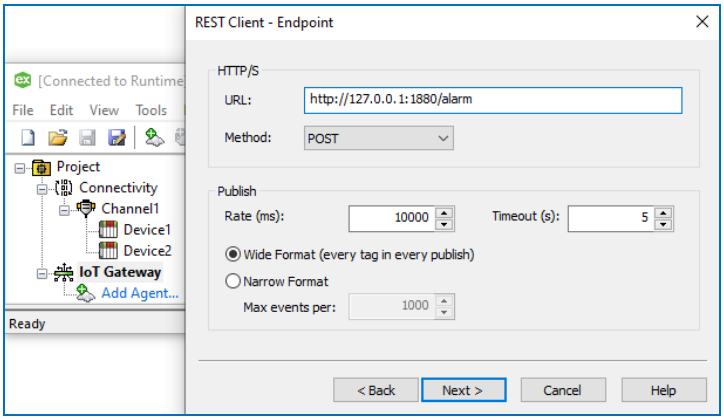
Gambar 6.3 Memunculkan IoT Gateway di Project di KEPServerEX

7. Klik Add Agent, muncul jendela New Agent. Isi kolom Name = RESTClient, dan pilih Type = REST Client. Klik tombol Next.



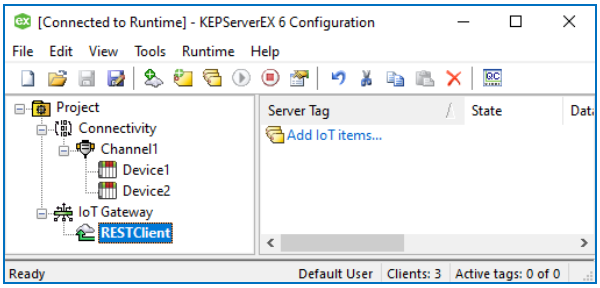
Gambar 6.4 Klik Add Agent, beri nama Agent, pilih Type = REST Client

8. Isi URL = http://127.0.0.1:1880/alarm. Perlu diketahui, alamat IP 127.0.0.1 adalah alamat IP localhost, dan port 1880 merupakan port khusus Node-RED. Di kolom Publish, pilih Wide Format (every tag in every publish).



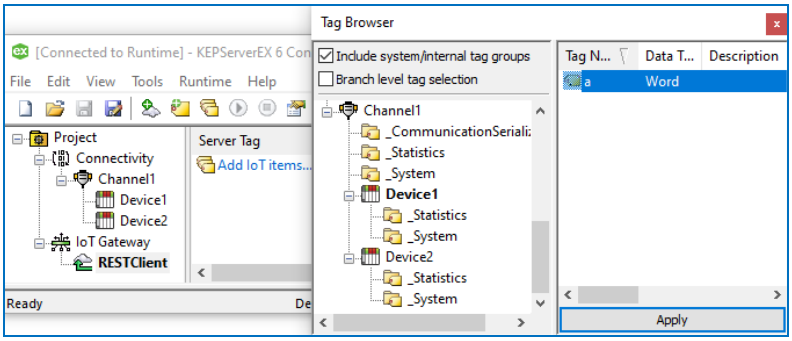
Gambar 6.5 Isi URL = http://127.0.0.1:1880/alarm dan di kolom Publish, pilih Wide Format

9. Klik Next hingga Finish, maka muncul RESTClient di bawah IoT Gateway.



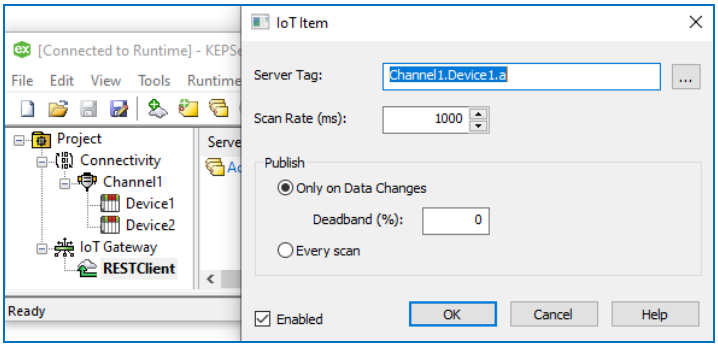
Gambar 6.6 Muncul RESTClient di bawah IoT Gateway

10. Klik pada Add IoT items. Di kotak Tag Browser yang muncul, pilih Tag a di Device1, dan tekan tombol Apply.



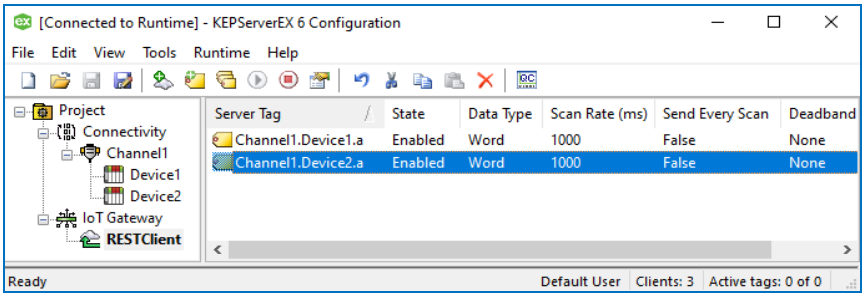
Gambar 6.7 Klik pada Add IoT items, muncul Tag Browser, pilih Tag a di Device1, pilih Apply

11. Muncul kotak IoT item Tag a Device1. Pilih Publish Only on Data Changes.



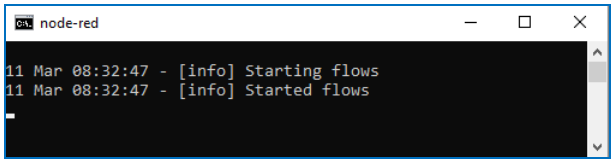
Gambar 6.8 Di kotak IoT item, di Tag a Device1, pilih Publish Only on Data Changes

12. Ulangi langkah 10-11 untuk menambahkan IoT Item Tag a di Device2.

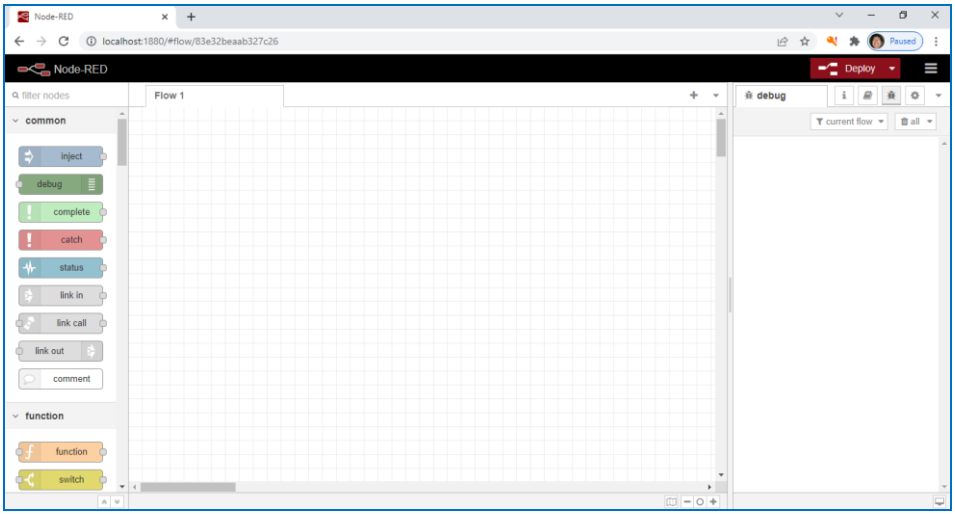


Gambar 6.9 Menambahkan Tag a di Device1 dan Device2 sebagai IoT item REST Client

13. Berikutnya, diinginkan agar kedua Tag yang dipublikasikan oleh REST Client melalui HTTP, dapat dibaca di Node-RED. Untuk itu buka Node-RED dengan cara mengetikkan **node-red** di Command Prompt, diikuti Enter.
14. Maka muncul pesan: **Welcome to Node-RED** di Command Prompt, yang diakhiri dengan pesan: **[info] Started flows**. Setelah muncul pesan ini, buka browser internet, ketikkan di kolom browser **localhost:1880**.

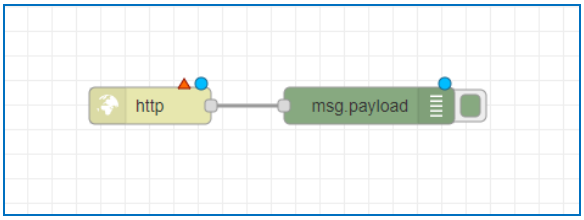


Gambar 6.10 [info] Started flows menandakan bahwa Node-RED sudah berjalan



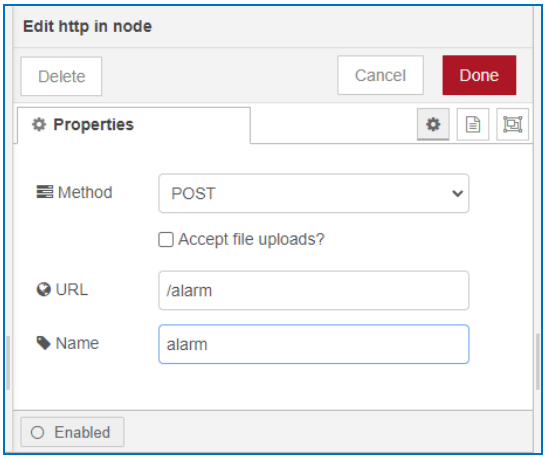
Gambar 6.11 Buka Editor Node-RED di browser dengan mengetikkan localhost: 1880

15. Untuk membaca data yang dipublikasikan REST Client melalui HTTP, ambil **node http in** dari palet kategori network, dan tempatkan di halaman Editor. Kemudian hubungkan dengan **node debug** (msg.payload) yang diambil dari palet kategori common untuk menampilkan data.

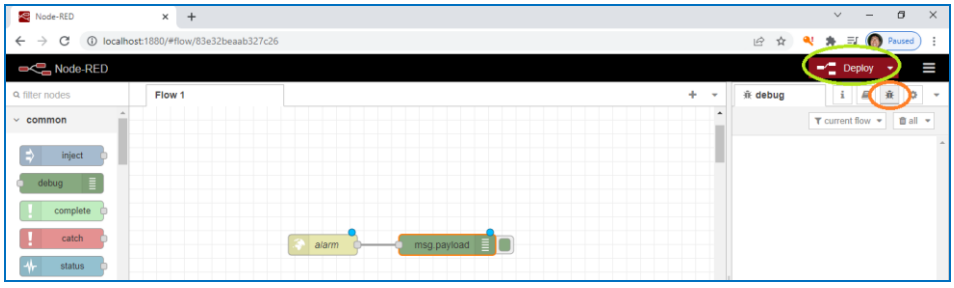


Gambar 6.12 Tempatkan node http in dan hubungkan dengan node debug

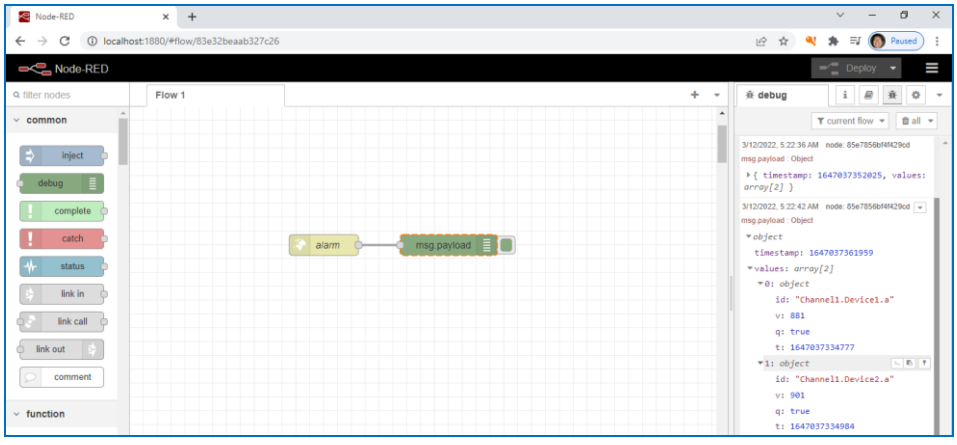
16. Ada tanda segitiga merah di **node http in**, menunjukkan bahwa node tersebut masih belum lengkap datanya. Klik 2 kali pada **node http in**, dan isi Method = POST, URL = /alarm, Name = alarm. Klik Done.
17. Tekan tombol Deploy di pojok kanan atas untuk menjalankan program, dan tekan tombol debug bergambar serangga untuk menampillkan data dari **node debug**. Apabila data masih belum muncul di kolom kanan, jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan data Tag a di Device1 dan Device2.



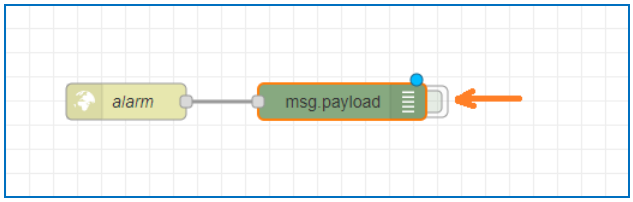
Gambar 6.13 Isi Method = POST, URL = /alarm, Name = alarm, klik Done



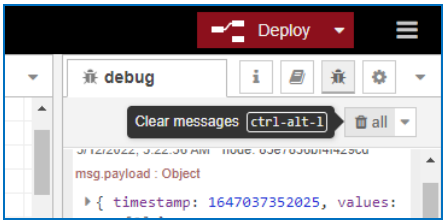
Gambar 6.14 Klik tombol Deploy untuk menjalankan program, dan tekan tombol debug (bergambar serangga) untuk menampilkan data dari node debug (msg.payload)



Gambar 6.15 Seharusnya kolom di kanan menampilkan data seperti gambar di atas. Apabila belum muncul data, pastikan OPC Quick Client menampilkan data Tag a di Device1 dan 2

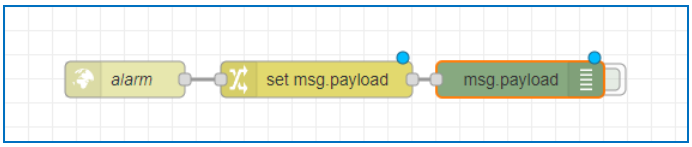


Gambar 6.16 Untuk menghentikan tampilan data di kolom kanan, non-aktifkan node debug dengan menekan tombol atau kotak di kanan node debug hingga masuk ke dalam node



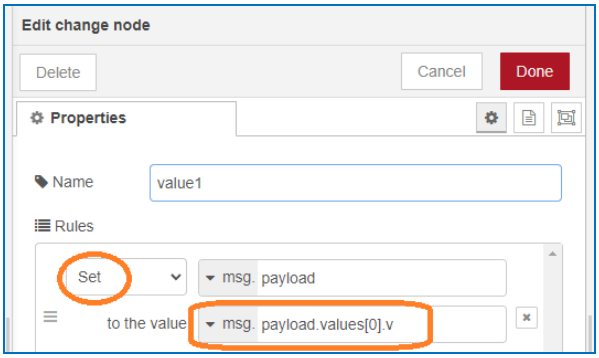
Gambar 6.17 Untuk membersihkan tampilan data di kolom kanan, tekan tombol bergambar kotak sampah dengan tulisan all di samping gambar kotak sampah

18. Perhatikan tampilan data yang dikirimkan oleh REST Client KEPServerEX dengan format JSON Standard seperti terlihat pada Gambar 6.15. Data tersebut cukup lengkap, namun untuk keperluan pengiriman data alarm, diinginkan hanya data value saja. Untuk membuat tampilan data hanya menampilkan data value saja, ambil **node change** dari palet kategori Function, dan sisipkan di tengah-tengah **node http in** dan **node debug**.

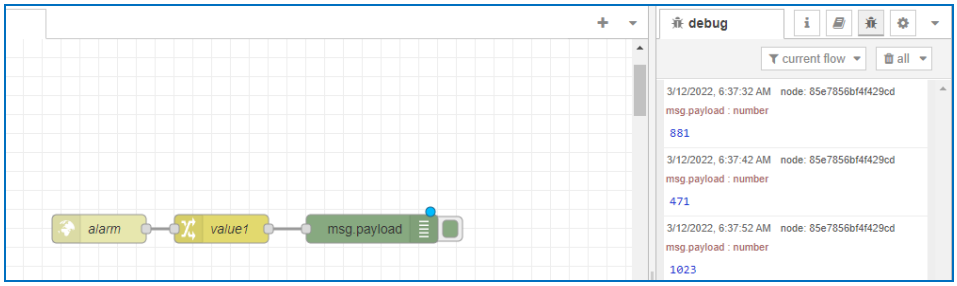


Gambar 6.18 Sisipkan node change di tengah-tengah node http in dan node debug

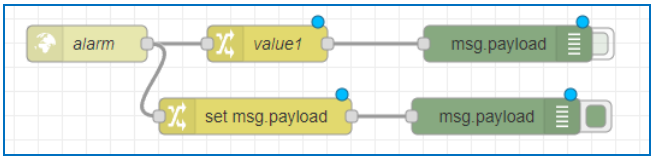
19. Klik 2 kali **node change**. Isi Name = **value1**. Atur Rules di pilihan **Set**, dan isi kolom to the value = **msg.payload.values[0].v**. Klik Done.
20. Tekan kembali tombol Deploy. Jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan data Tag. Seharusnya kolom di kanan menampilkan nilai Tag a di Device1 seperti Gambar 6.20.
21. Agar nilai Tag a di Device2 juga muncul di kolom kanan, tambahkan **node change** dan **node debug** seperti Gambar 6.21.



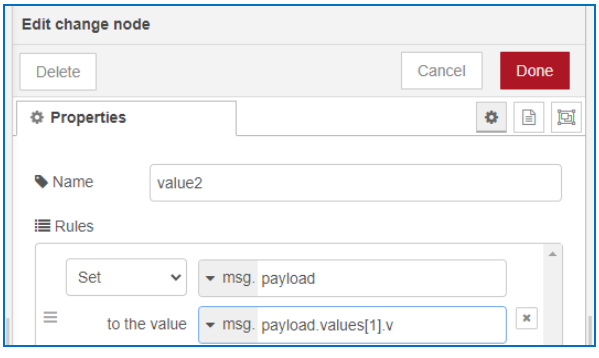
Gambar 6.19 Isi Name = value1, Rules: Set msg.payload to the value msg.payload.values[0].v



Gambar 6.20 Kolom debug di kanan menampilkan value saja, yaitu nilai Tag a di Device1

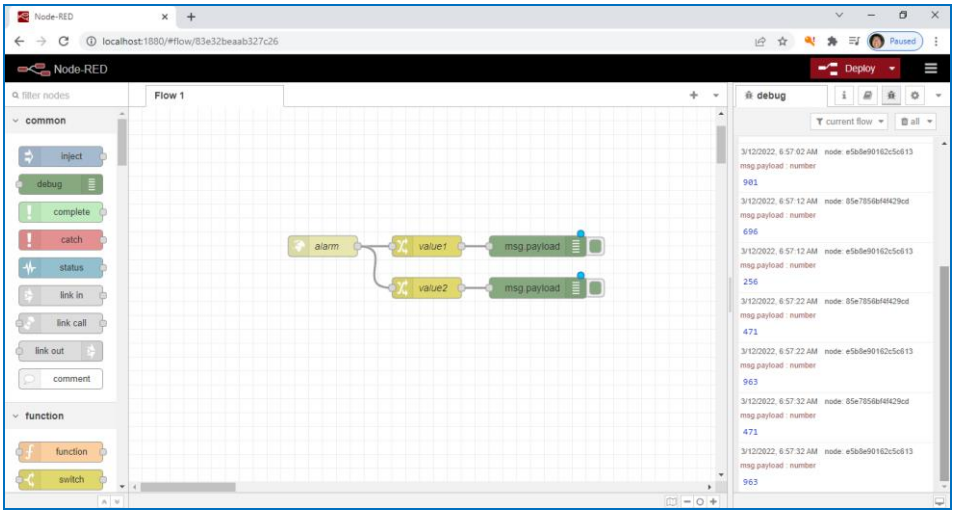


Gambar 6.21 Agar nilai Tag a Device2 juga muncul, tambahkan node change dan node debug



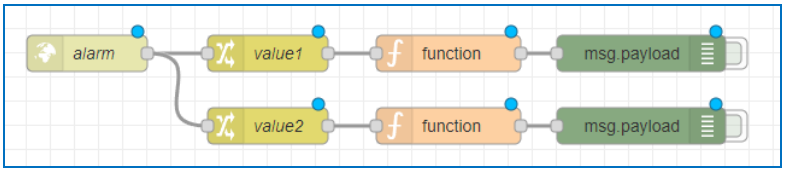
Gambar 6.22 Isi Name = value2, Rules: Set msg.payload to the value msg.payload.values[1].v

22. Klik 2 kali **node change** yang baru. Isi Name = **value2**. Atur Rules di pilihan **Set**, dan isi kolom to the value = **msg.payload.values[1].v**.
23. Klik Done. Tekan kembali tombol Deploy. Jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan data Tag. Seharusnya kolom di kanan menampilkan nilai Tag a Device1 dan Device2.

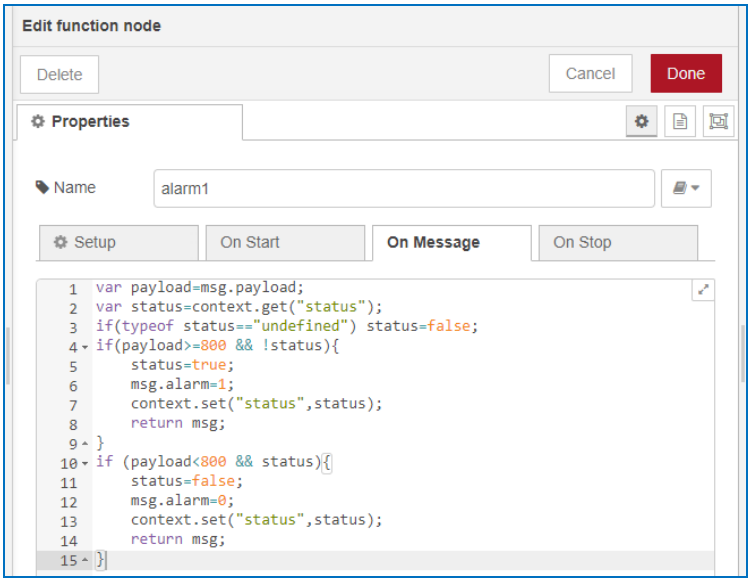


Gambar 6.23 Nilai Tag a Device1 dan Device2 ditampilkan di kolom kanan

24. Berikutnya, diinginkan agar data Tag a tersebut hanya ditampilkan sekali saja, yaitu ketika nilainya melebihi nilai batas (yang diatur di nilai 800), dan tidak akan ditampilkan lagi, kecuali bila nilainya sudah kembali normal (kurang dari 800), dan itupun juga hanya ditampilkan sekali. Pengaturan seperti ini diperlukan, agar alarm tidak selalu mengirimkan data. Jadi alarm hanya mengirimkan data sekali ketika nilai melebihi batas, dan sekali ketika nilai kembali normal. Sebagai contoh, anggap bahwa nilai Tag a berturut-turut adalah: 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 900, 800, 700, 600, 500, 400, 400, 300, 200, 100. Maka alarm akan mengirimkan data 2 kali saja, yaitu data 900 dan data 700, data yang lain tidak dikirimkan. Untuk membuat pengaturan seperti ini, sisipkan **node function** (di palet function) di antara **node change** dan **node debug**.
25. Klik 2 kali **node function**, dan isi dengan kode seperti Gambar 6.25.



Gambar 6.24 Sisipkan node function di antara node change dan node debug

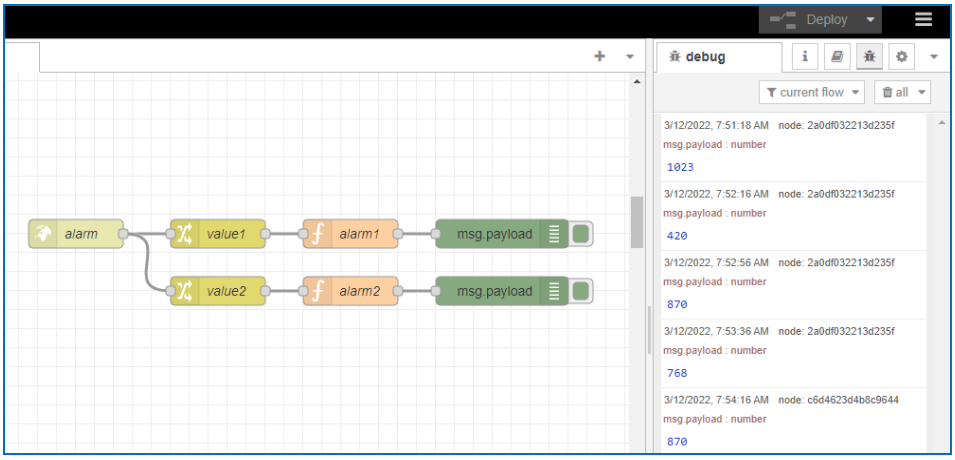


Gambar 6.25 Klik 2 kali pada node function, isi Name=alarm1, kode program di On Message

	Kode program di kolom On Message
01.	`var payload=msg.payload;`
02.	`var status=context.get("status");`
03.	`if(typeof status=="undefined") status=false;`
04.	`if(payload>=800 && !status){`
05.	`    status=true;`
06.	`    msg.alarm=1;`
07.	`    context.set("status",status);`
08.	`    return msg;`
09.	`}`
10.	`if (payload<800 && status){`
11.	`    status=false;`
12.	`    msg.alarm=0;`
13.	`    context.set("status",status);`
14.	`    return msg;`
15.	`}`

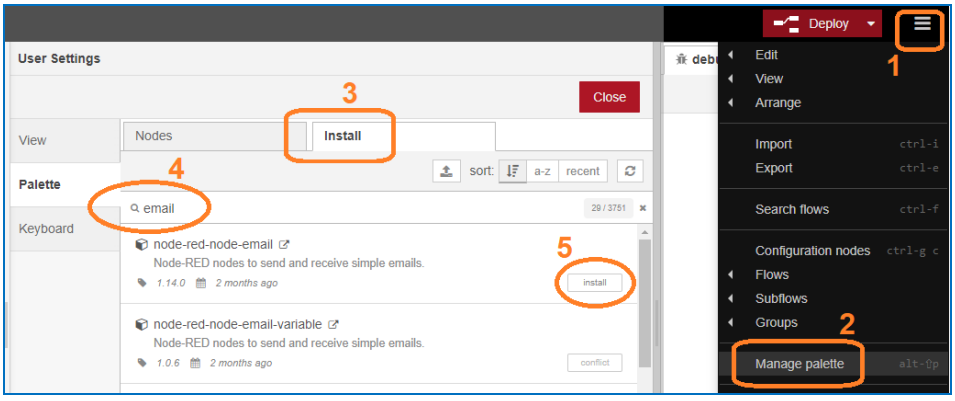
26. Klik Done. Ulangi untuk **node function** yang kedua, beri nama **alarm2**.

27. Tekan tombol Deploy. Jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan data Tag. Kemudian geser Potensio di rangkaian U1 dan U2, dan perhatikan tampilan data di kolom debug. Seharusnya data yang ditampilkan sesuai aturan di langkah no. 24.



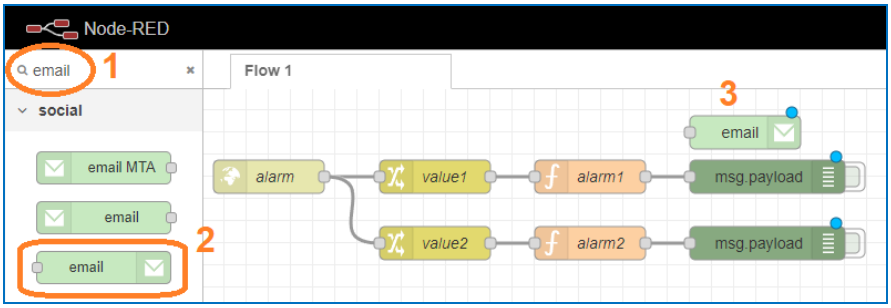
Gambar 6.26 Deploy program, dan geser Potensio di rangkaian U1 dan U2. Perhatikan bahwa data yang ditampilkan adalah data pertama, yang melebihi nilai batas, dan data yang pertama, yang nilainya kurang dari nilai batas, data yang lain tidak ditampilkan

28. Berikutnya, diinginkan data alarm tersebut dikirimkan melalui email. Tambahkan **node email** dengan cara menekan tombol ≡ di pojok kanan atas, pilih Manage palette, di kotak yang muncul, pilih Tab Install, ketik: email, dan pilih **node-red-node-email**, dan klik tombol install.



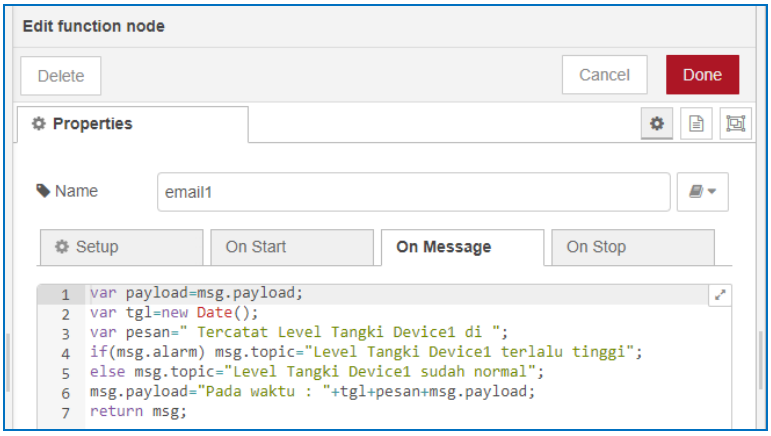
Gambar 6.27 Pilih Manage pallete, pilih Tab install, ketik email, instal node-red-node-email

29. Setelah terinstal (*installed*), klik tombol Close. Di pojok kiri atas, di kolom filter nodes, ketik: email. Ambil **node email** dan tempatkan di Editor.



Gambar 6.28 Ambil node email, dan tempatkan di Editor

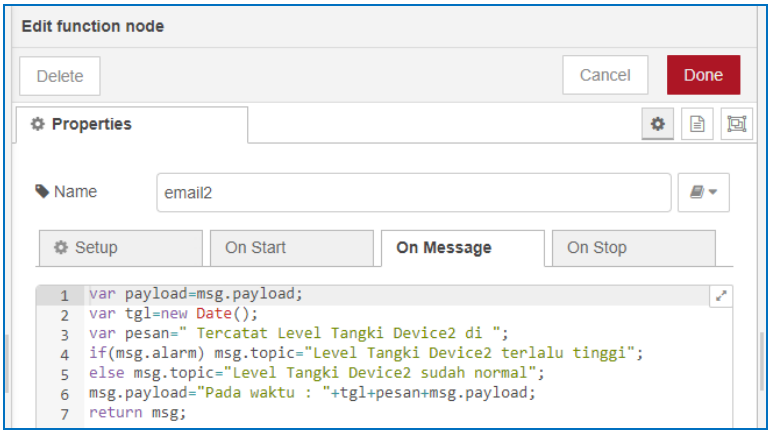
30. Diinginkan agar email yang terkirim memiliki subjek atau topic dan isi pesan. Untuk itu, tambahkan **node function**, dan sisipkan antara **node function alarm1** dengan **node debug**. Klik 2 kali **node function** yang baru ini, dan isi dengan nama dan kode program berikut ini:



Gambar 6.29 Klik 2 kali pada node function, isi Name=email1, kode program di On Message

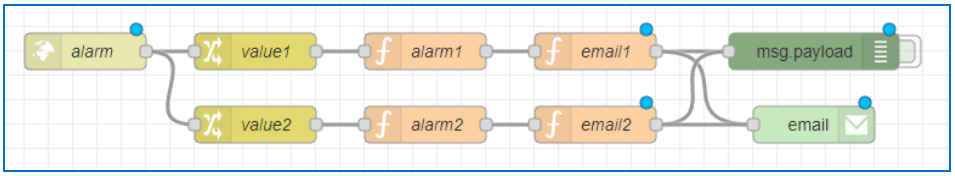
	Kode program di kolom On Message
01.	`var payload=msg.payload;`
02.	`var tgl=new Date();`
03.	`var pesan=" Tercatat Level Tangki Device1 di ";`
04.	`if(msg.alarm) msg.topic="Level Tangki Device1 terlalu tinggi";`
05.	`else msg.topic="Level Tangki Device1 sudah normal";`
06.	`msg.payload="Pada waktu : "+tgl+pesan+msg.payload;`
07.	`return msg;`

31. Duplikasi **node function email1** di atas. Ubah Name dari email1 menjadi email2, dan di kode programnya, ubah kata Device1 menjadi Device2.



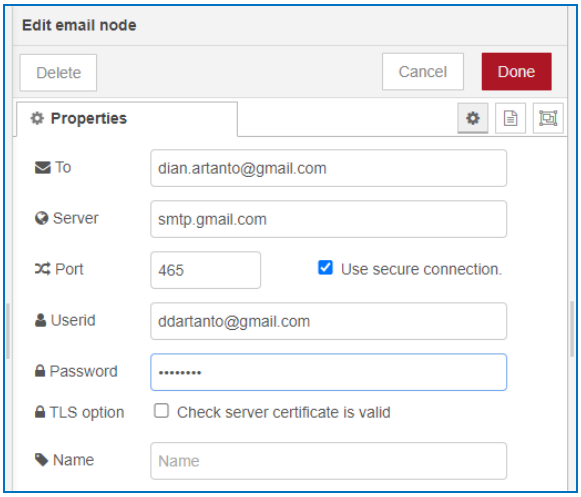
Gambar 6.30 Duplikasi node function email1, ubah namanya menjadi email2, dan ganti kata Device1 di kode program di On Message menjadi Device2

32. Agar flow menjadi sederhana, hapus **node debug** yang kedua, gabungkan output dari **node function email1** dengan **email2** ke **node debug** yang pertama, dan juga hubungkan output keduanya ke **node email**.



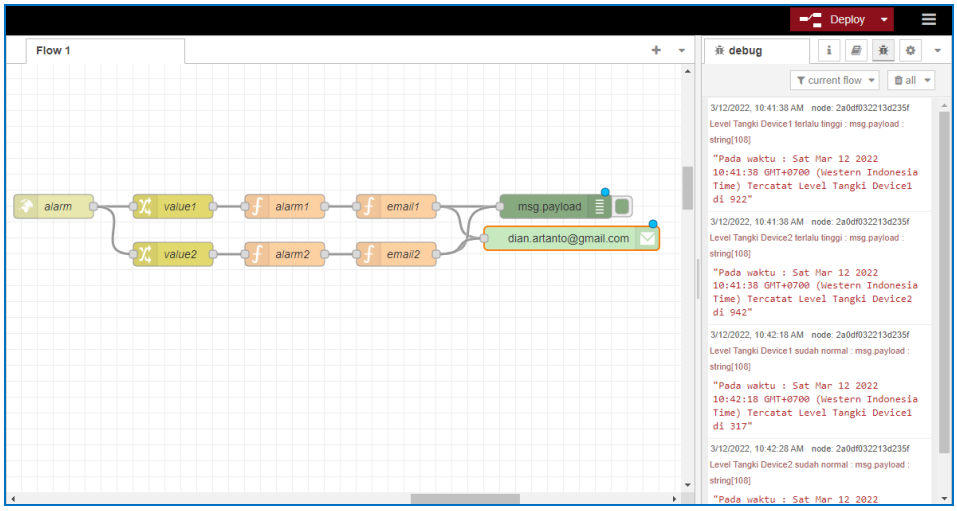
Gambar 6.31 Untuk menyederhanakan flow, hapus node debug kedua, dan buat node debug pertama dan node email mendapat output dari node function email1 dan email2

33. Berikutnya klik 2 kali pada **node email**. Pada kotak Edit email node, isi kolom To dengan alamat email yang dituju. Kolom server dan port sudah terisi secara default. Isi kolom userid dan password dengan akun gmail pembaca. Hilangkan tanda centang pada TLS Option.
34. Tekan tombol Deploy. Aktifkan **node debug** dengan menekan kotak di samping kanan node. Jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan data Tag. Geser Potensio di rangkaian U1 dan U2, dan perhatikan tampilan data di kolom debug.



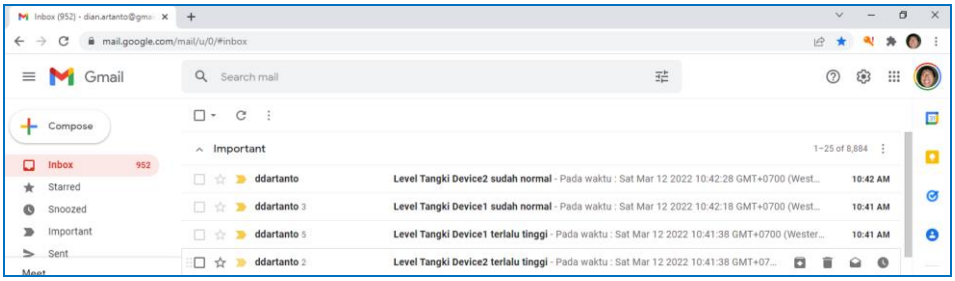
Gambar 6.32 Isi kolom To dengan email yang dituju, dan userid dan password dengan akun gmail pembaca, beri centang di Use secure connection, dan hilangkan centang di TLS option

35. Ketika rangkaian Proteus dijalankan, dan OPC Quick Client menampilkan nilai Tag, serta node debug diaktifkan, seharusnya kolom debug menampilkan data pertama kali, yaitu setiap kali nilai Potensio melebihi nilai batas, dan kurang dari nilai batas. Data tersebut disertai dengan topik (subjek email), catatan waktu pembacaan dan keterangan.

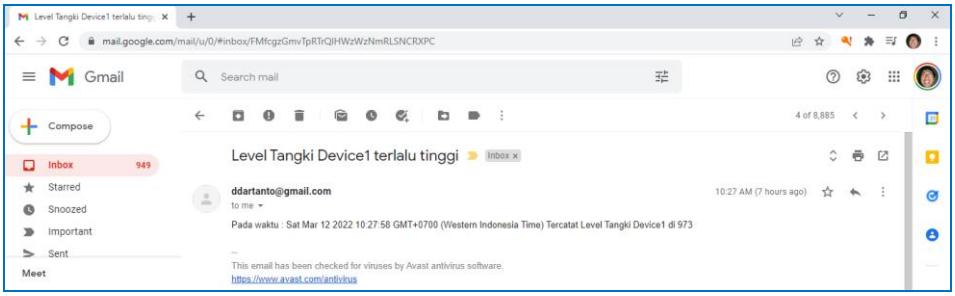


Gambar 6.33 Kolom debug menampilkan data sekali, yaitu ketika nilai Potensio melebihi nilai batas, dan ketika kurang dari nilai batas, dengan tambahan catatan waktu dan keterangan

36. Buka inbox di alamat email yang dituju, seharusnya ada email yang masuk, dengan subjek “Level Tangki Device…”. Buka email tersebut.

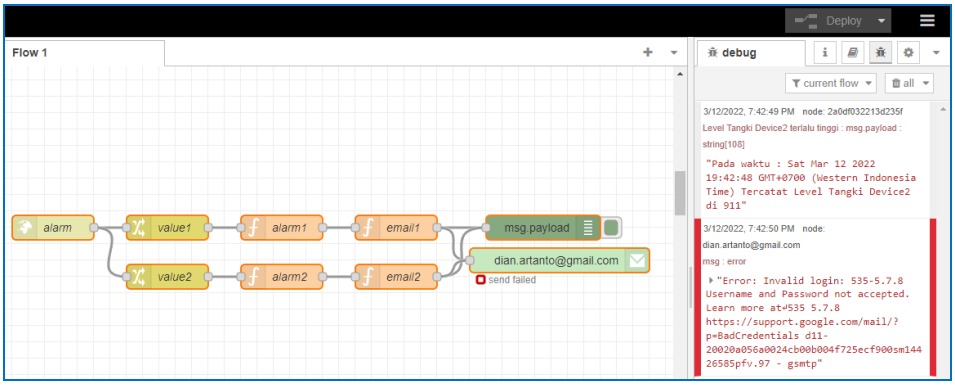


Gambar 6.34 Ada email yang masuk dengan subjek “Level Tangki Device…”



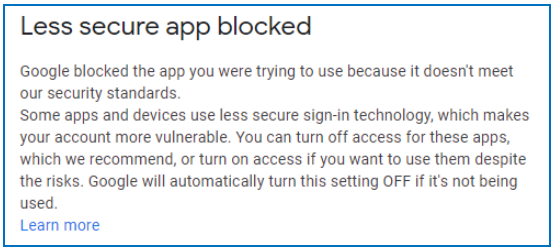
Gambar 6.35 Buka isi email, ada catatan waktu pembacaan dan nilai Tag

37. Apabila tidak ada email yang masuk, perhatikan kembali tampilan data di kolom debug. Apabila ada pesan Error di kolom debug, dan di **node email** muncul tulisan *send failed*, lakukan perbaikan pada isian di node email.

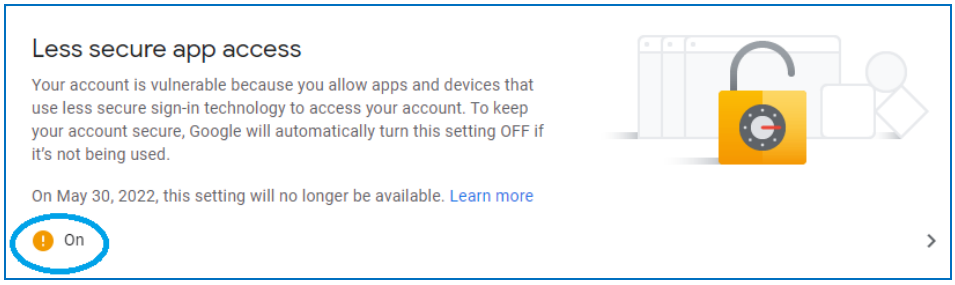


Gambar 6.36 Pengiriman email gagal karena Error

Catatan: Apabila di bawah node email muncul keterangan send failed, maka ada beberapa kemungkinan penyebab kegagalan pengiriman tersebut. Yang pertama, pastikan bahwa userid dan password terisi akun gmail yang benar. Kesalahan userid dan password bisa menyebabkan Error. Yang kedua, karena Google memblok akses ke email tersebut. Untuk membuat Google membuka akses ke email tersebut, hidupkan mode Less secure app access. Untuk menghidupkan mode Less secure app access ini, buka email tersebut. Klik pada nama akun di pojok kanan atas, pilih Manage your Google Account, pilih Security, pilih kolom Less secure app access, kemudian ubah OFF menjadi ON.

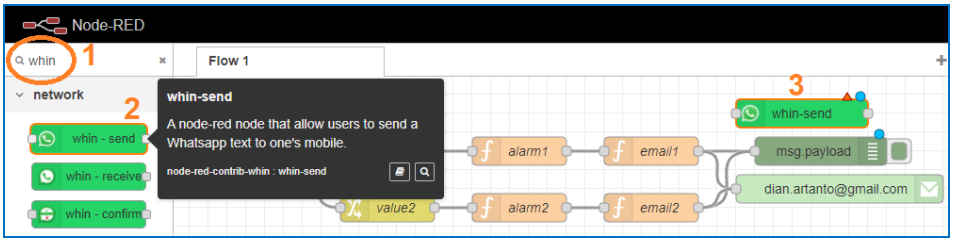


Gambar 6.37 Error bisa terjadi karena Google memblok akses ke email



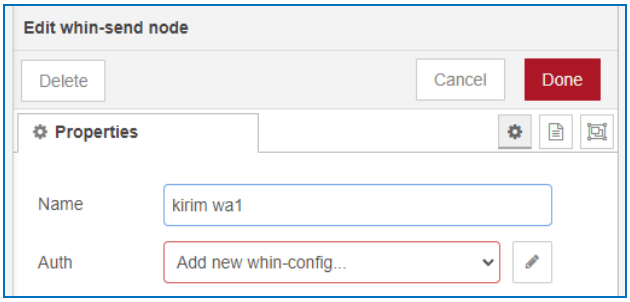
Gambar 6.38 Akses bisa dibuka dengan membuat mode Less secure app access menjadi ON

38. Berikutnya, diinginkan agar data alarm tersebut juga dikirimkan melalui Whatsapp. Untuk itu, tambahkan **node whin send**. Klik tombol ≡ di pojok kanan atas, pilih Manage palette. Di kotak yang muncul, pilih Tab Install, ketik: **whin**, dan pilih **node-red-contrib-whin**, dan klik tombol install.
39. Setelah **node-red-contrib-whin** berhasil terinstal, klik Close kotak Manage pallete. Kemudian di pojok kiri atas, ketik **whin**. Ambil **node whin send**.



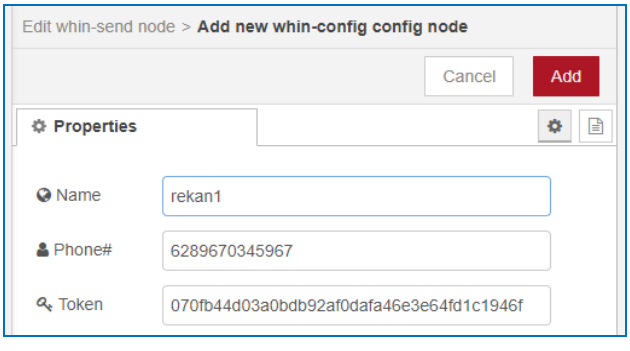
Gambar 6.39 Ketik whin di kolom filter nodes, ambil node whin-send

40. Hubungkan output **node function email1** dan **email2** ke input **node whin-send**. Klik 2 kali **node whin-send**, di kotak Edit whin-send node, isi Name. Klik tombol pensil di samping kanan kolom Auth.



Gambar 6.40 Klik 2 kali node whin-send, isi kolom Name dan klik tombol pensil di Auth

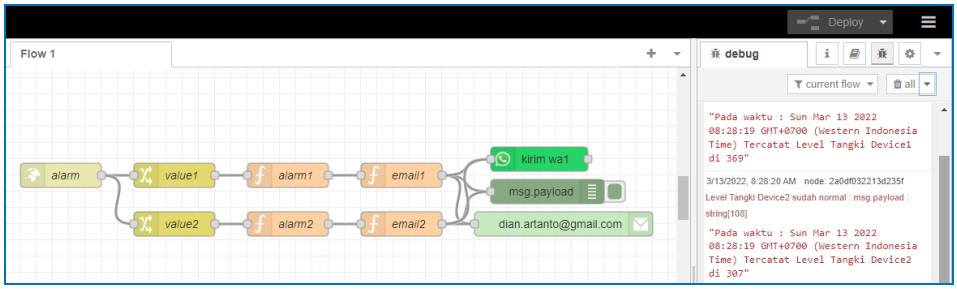
41. Muncul kotak Add new whin-config config node, isi Name, isi Phone dengan nomor WA yang akan dituju (tidak perlu tanda +, cukup kode area 62). Isi token (untuk mendapatkan token, silahkan melihat catatan di bawah). Setelah semua terisi, klik tombol Add, diikuti klik tombol Done.



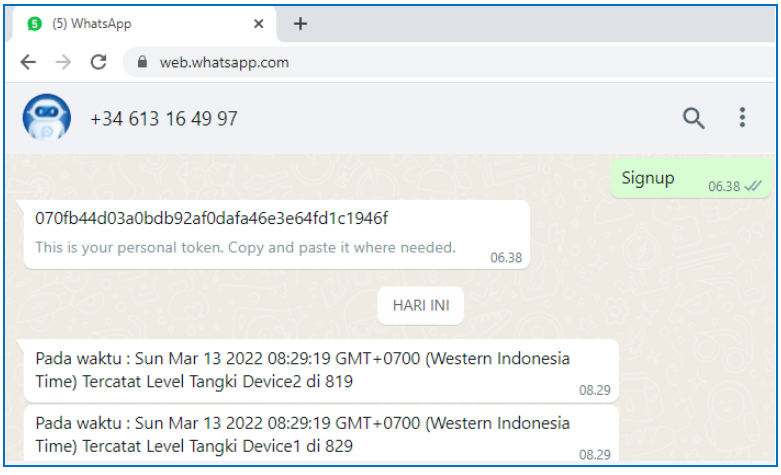
Gambar 6.41 Isi kolom Name, isi no WA yang dituju (tidak perlu tanda +) dan isi Token

Catatan: Untuk mendapatkan token, caranya mudah sekali, hanya dengan mengirimkan kata **signup** ke nomor: **+34 613 16 49 97**. Tunggu beberapa saat, maka nomor tersebut akan mengirimkan token berupa 40 digit karakter. Simpan nomor tersebut di Kontak HP (*handphone*), beri nama tertentu, misal **Alarm**. Ketika program node-RED di atas dijalankan, maka data alarm akan dikirimkan melalui Whatsapp, dari nomor tersebut ke nomor HP yang dituju.

42. Tekan tombol Deploy. Aktifkan juga **node debug**. Jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan nilai Tag. Geser Potensio di rangkaian U1 dan U2, dan perhatikan tampilan di kolom debug. Apabila data alarm muncul di kolom debug, seharusnya data alarm juga muncul di email dan di nomor Whatsapp yang dituju.



Gambar 6.42 Tekan Deploy, aktifkan node debug, seharusnya muncul data di kolom debug



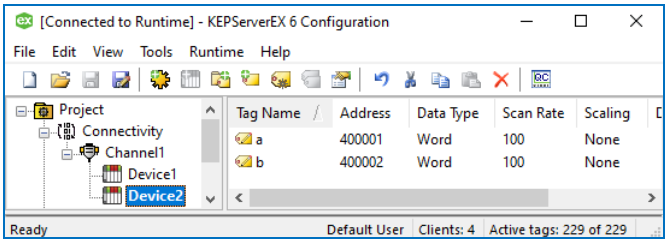
Gambar 6.43 Bila data muncul di kolom debug, muncul juga di nomor WA yang dituju

43. Sampai di sini pengiriman data alarm melalui email dan Whatsapp menggunakan REST Client, Node-RED dan **node whin** selesai.

6.3 REST Server dan Node-RED

Apabila REST Client KEPServerEX digunakan untuk mempublikasikan data Tag (Tag dari Device yang terhubung) ke Cloud melalui HTTP, maka REST Server KEPServerEX bisa digunakan untuk 2 arah, mempublikasikan data Tag sekaligus mengubah/mengatur data Tag tersebut melalui HTTP. Pada Sub Bab ini akan diperlihatkan bagaimana membuat tampilan web yang dapat memonitor dan mengontrol data Tag melalui HTTP, dengan bantuan REST Server KEPServerEX dan Node-RED Dashboard. Berikut ini langkah-langkah pembuatannya:

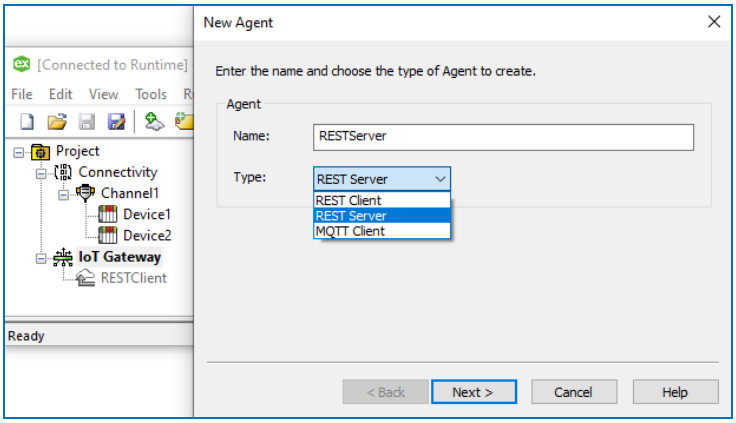
1. Buka KEPServerEX Configuration, buat sebuah Channel, dengan protokol Modbus RTU Serial, dengan 2 buah Device (ID = 1 dan ID = 2).
2. Buat masing-masing Device memiliki 2 buah Tag, a dan b, dengan alamat 400001 dan 400002.



Gambar 6.44 Channel1 dengan 2 buah Device, 2 Tag di setiap Device, di 400001 dan 400002

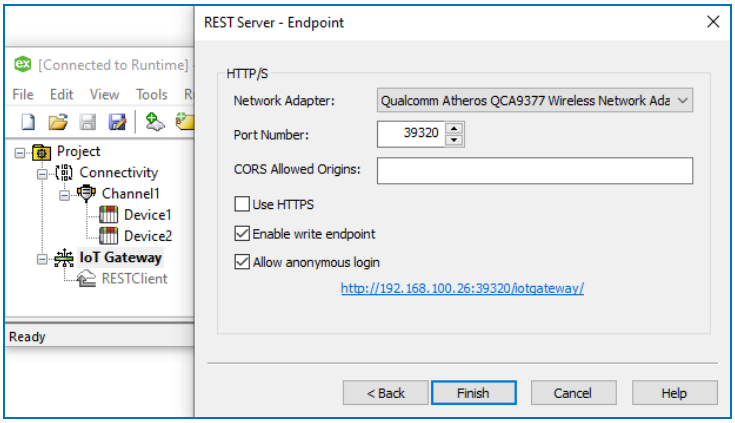
3. Sebagai perangkat Modbus RTU Slave untuk Device1 dan Device2, gunakan rangkaian 2 buah Arduino di Proteus seperti Gambar 4.51.
4. Gunakan program Outseal Gambar 4.52 untuk kedua Arduino. Buat alamat Modbus = 1 untuk U1, dan alamat Modbus = 2 untuk U2.
5. Jalankan OPC Quick Client untuk Device1 dan Device2, dan juga rangkaian Proteus. Pastikan bahwa OPC Quick Client dapat menampilkan nilai Tag a (400001) dan mengatur nilai Tag b (400002) di Device1 dan Device2.
6. Agar nilai Tag a dan Tag b di Device1 dan Device2 dapat dimonitor dan dikontrol melalui HTTP, tambahkan REST Server di IoT Gateway.

7. Namun sebelum menambahkan REST Server di IoT Gateway, klik kanan pada REST Client, pilih Disable. Kemudian klik kanan pada IoT Gateway, pilih New Agent, beri nama RESTServer, dan pilih Type REST Server.



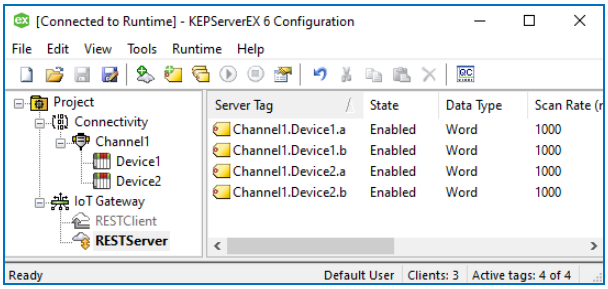
Gambar 6.45 Klik kanan IoT Gateway, pilih New Agent, Name=RESTServer, Type=REST Server

8. Klik Next. Pilih Network Adapter, hilangkan centang pada Use HTTPS, dan beri centang pada Enable write endpoint dan Allow anonymous login.



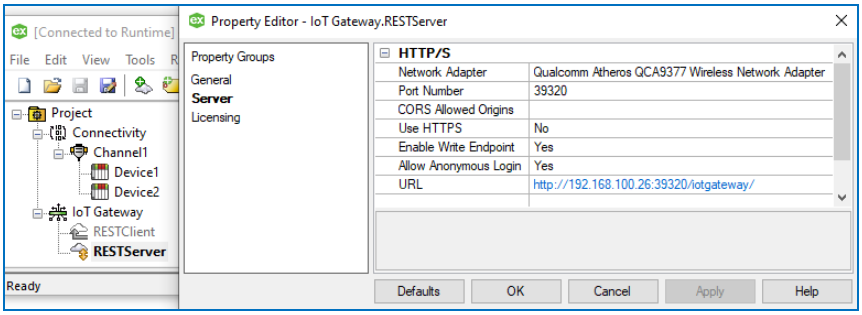
Gambar 6.46 Pilih Network Adapter, beri centang pada Enable write dan Allow anonymous

9. Klik Finish, maka muncul REST Server di IoT Gateway. Berikutnya tambahkan Item pada REST Server, dengan meng-klik Add IoT items. Di kotak Tag Browser, pilih Device1, pilih Tag a dan b, klik Apply. Muncul kotak Iot Items, klik OK. Ulangi untuk menambahkan Tag a dan b Device2.



Gambar 6.47 REST Server dengan 4 buah Tag

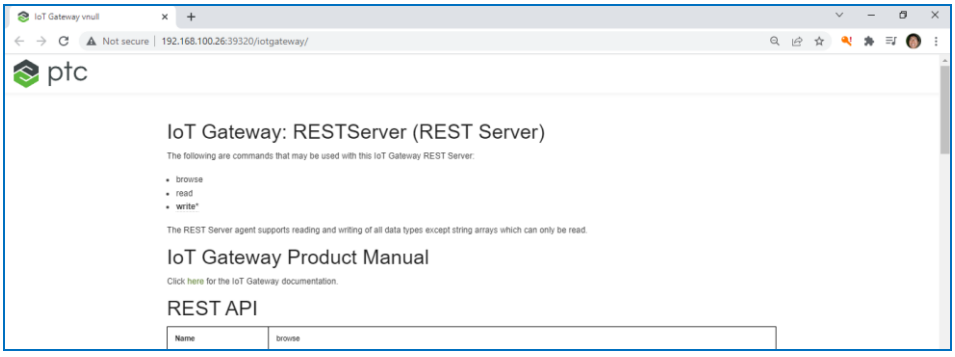
10. Klik 2 kali RESTServer. Pilih Server. Klik pada link URL.



Gambar 6.48 Klik 2 kali RESTServer, pilih Server, klik link URL (http://192.168.100.26 dst.nya)

Catatan: Untuk seterusnya, apabila pembaca menemukan alamat **192.168.100.26** dalam buku ini, silahkan diganti dengan IP di URL yg muncul di komputer pembaca.

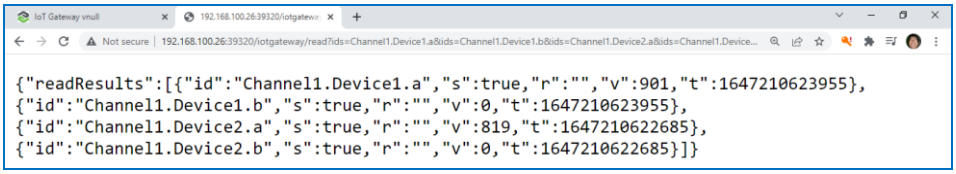
11. Begitu link di-klik, muncul halaman REST API IoT Gateway REST Server



Gambar 6.49 Muncul halaman REST API IoT Gateway REST Server

12. Sesuai REST API, untuk instruksi read (membaca nilai keempat Tag) dengan metode GET, dapat dilakukan dengan mengetikkan teks berikut ini di browser (**ganti 192.168.100.26 dengan alamat IP di URL yang muncul**)

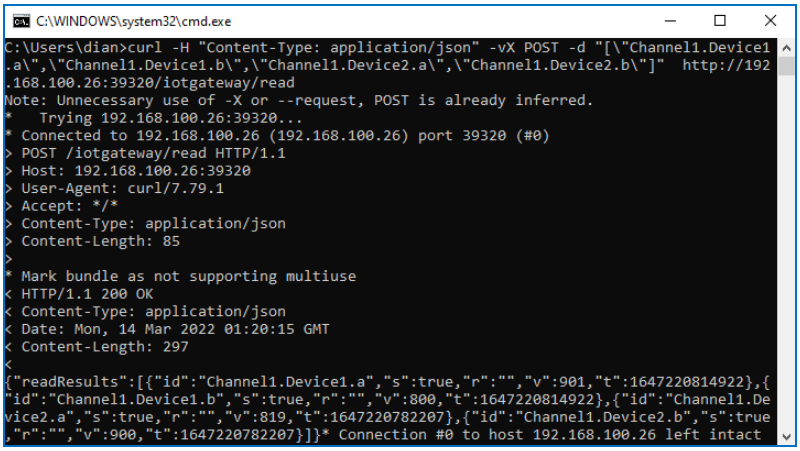
```
http://192.168.100.26:39320/iotgateway/read?
ids=Channel1.Device1.a&ids=Channel1.Device1.b&
ids=Channel1.Device2.a&ids=Channel1.Device2.b
```



Gambar 6.49 Hasil instruksi read, nilai keempat Tag dapat ditampilkan di halaman web

13. Di samping dengan browser, sesuai dengan petunjuk di halaman REST API, pembacaan data keempat Tag dapat juga dilakukan dengan metode POST, dengan mengetikkan teks berikut ini di command prompt:

```
curl -H "Content-Type: application/json" -vX POST
-d "[\"Channel1.Device1.a\",\"Channel1.Device1.b
\",\"Channel1.Device2.a\",\"Channel1.Device2.b\"]"
 http://192.168.100.26:39320/iotgateway/read
```

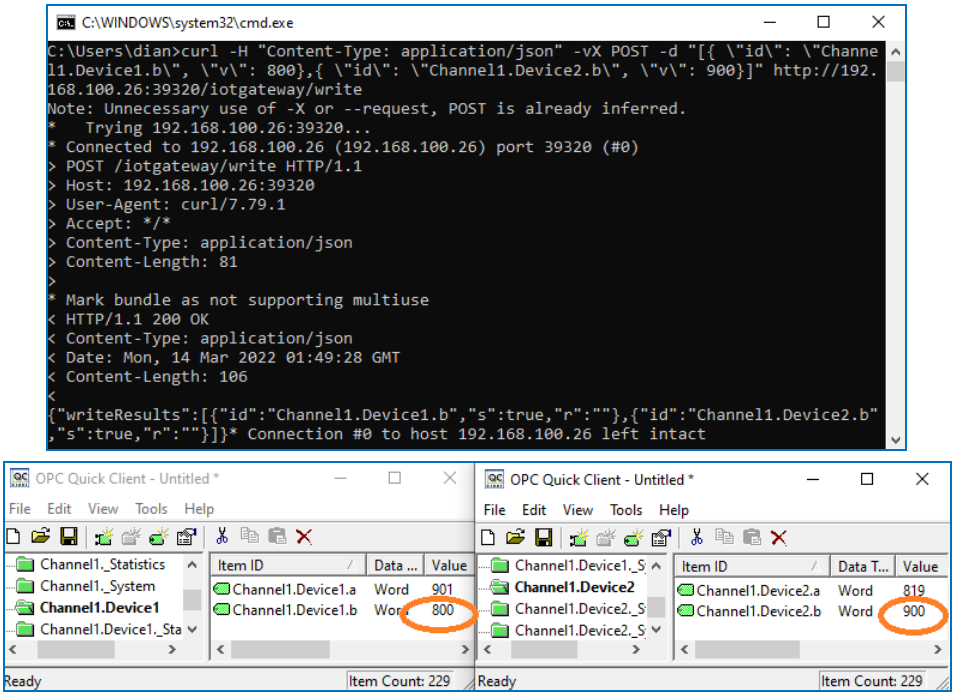


Gambar 6.50 Hasil instruksi read dengan metode POST di command prompt

Catatan: Apabila halaman URL REST API tidak terbuka, atau nilai Tag tidak dapat muncul di web maupun di command prompt, hal tersebut bisa disebabkan karena waktu Runtime demo sudah habis. Untuk itu reset kembali Runtime KEPServerEX.

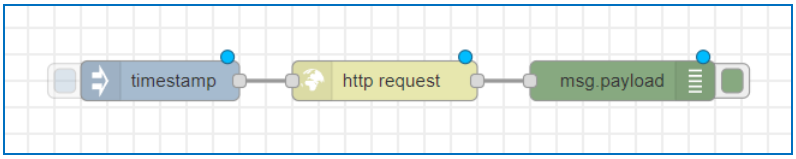
14. Sesuai dengan petunjuk di halaman REST API, dengan metode POST, dapat dilakukan pengubahan nilai Tag. Untuk itu ketikkan teks berikut ini di command prompt untuk mengatur nilai Tag b di Device1 dan Device2 (Tag a tidak bisa diubah karena terhubung dengan Potensio, sedangkan Tag b bisa diubah karena terhubung dengan LED).

```
curl -H "Content-Type: application/json" -vX POST -d "[{ \"id\": 
\"Channel1.Device1.b\", \"v\": 800},{ \"id\": \"Channel1.Device2.b
\", \"v\": 900}]" http://192.168.100.26:39320/iotgateway/write
```



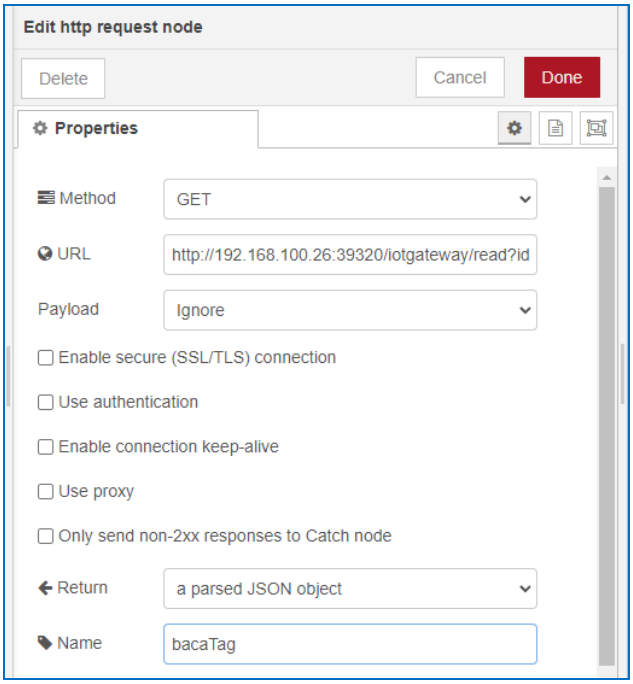
Gambar 6.51 Hasil instruksi write dengan metode POST di command prompt

15. Untuk memudahkan pembacaan dan pengaturan data Tag, berikut ini ditunjukkan pembuatan tampilan user interface di halaman web menggunakan dashboard Node-RED. Jalankan Node-RED dengan mengetikkan **node-red** di command prompt. Setelah muncul [info] Started flows, ketik di kolom browser, **localhost:1880.**
16. Di halaman Editor Node-RED, ambil dan tempatkan **node inject**, **node http request** dan **node debug**, dan hubungkan ketiganya.



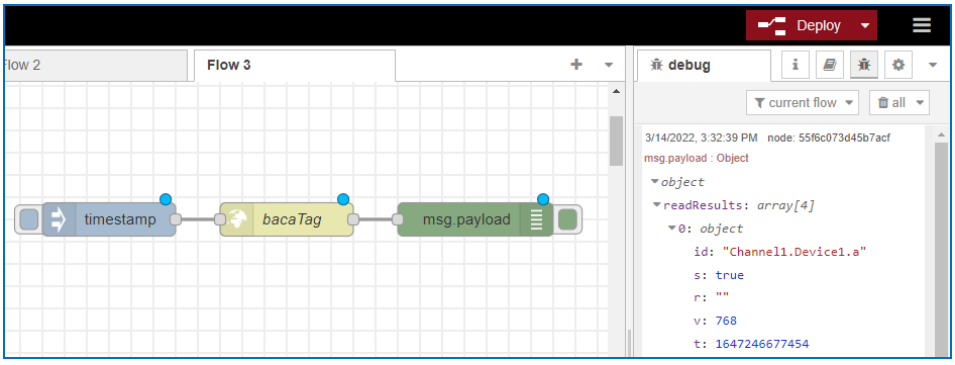
Gambar 6.52 Ambil node inject, node http request, node debug, hubungkan ketiganya

17. **Node inject** (secara otomatis berubah namanya menjadi timestamp), dapat digunakan untuk men-trigger jalannya program. Trigger bisa diaktifkan secara manual dengan cara menekan tombol kotak di kiri node, atau bisa diaktifkan secara otomatis pada waktu tertentu atau berulang dengan interval waktu tertentu. Sementara ini, gunakan trigger manual.
18. Klik 2 kali **node http request**. Di kotak Edit, isi Method = **GET**, URL = **http://192.168.100.26:39320/iotgateway/read?ids=Channel1.Device1.a&ids=Channel1.Device1.b&ids=Channel1.Device2.a&ids=Channel1.Device2.b**, Return = **a parsed JSON object**, dan Name = **bacaTag**.



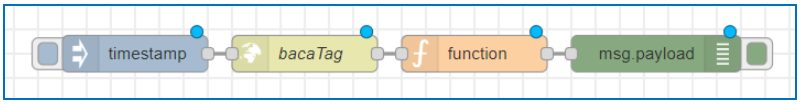
Gambar 6.53 Untuk membaca keempat Tag, gunakan instruksi read yang sama seperti di langkah no. 12, yaitu isi Method=GET, dan isi URL seperti Gambar 6.49

19. Klik Done. Tekan tombol Deploy. Jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan data Tag a dan b di Device1 dan Device2. Setelah itu, tekan tombol kotak di kiri node inject (timestamp). Seharusnya di kolom debug (tekan tombol bergambar serangga) menampilkan data keempat Tag dalam format JSON.



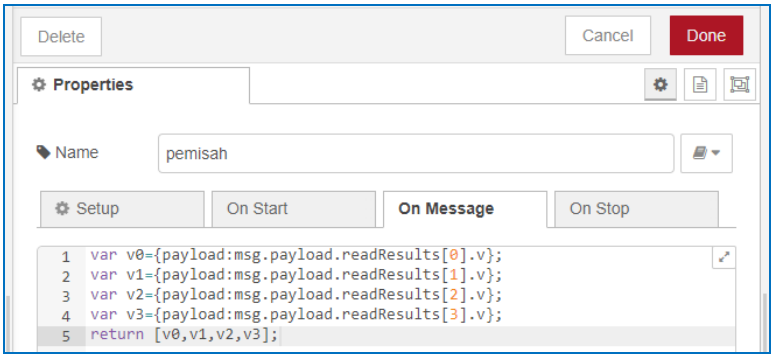
Gambar 6.54 Kolom debug menampilkan data ke-4 Tag setiap kali tombol node inject ditekan

20. Diinginkan keempat data Tag tersebut dipisahkan. Untuk itu sisipkan node function di antara node http request dan node debug.



Gambar 6.55 Sisipkan node function di antara node http request (bacaTag) dan node debug

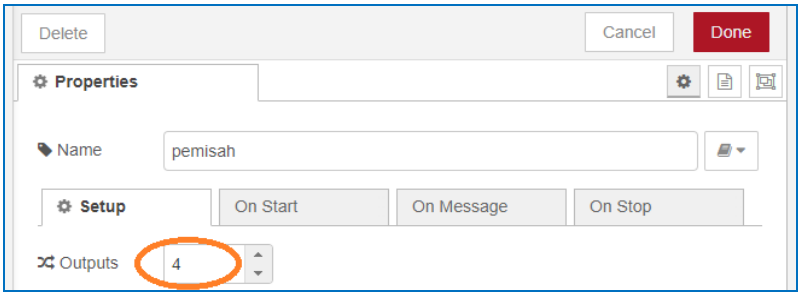
21. Klik 2 kali node function, isi Name = pemisah, dan kode di On Message.



Gambar 6.56 Di kotak Edit, isi Name = pemisah, dan kode program di kolom On Message

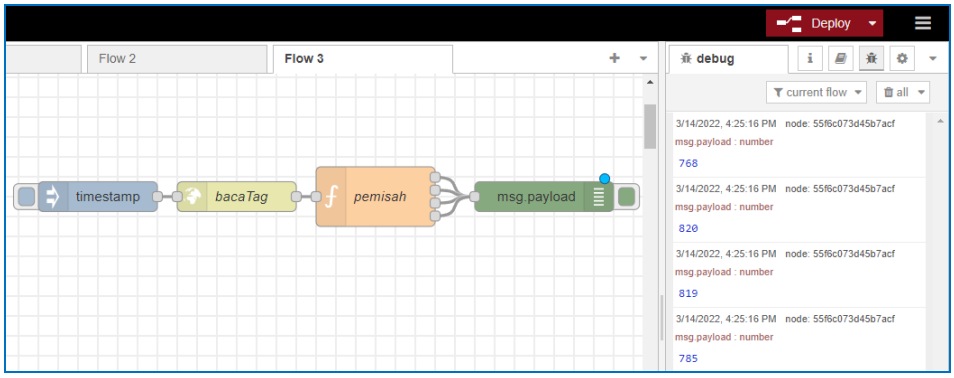
	Kode program di kolom On Message
01.	var v0={payload:msg.payload.readResults[0].v};
02.	var v1={payload:msg.payload.readResults[1].v};
03.	var v2={payload:msg.payload.readResults[2].v};
04.	var v3={payload:msg.payload.readResults[3].v};
05.	return [v0,v1,v2,v3];

22. Agar node function tersebut menghasilkan 4 saluran, klik Tab Setup, dan buat Outputs dari 1 menjadi 4. Klik Done untuk menutup kotak Edit.



Gambar 6.57 Di kotak Edit, di Tab Setup, ubah isi Outputs dari 1 menjadi 4

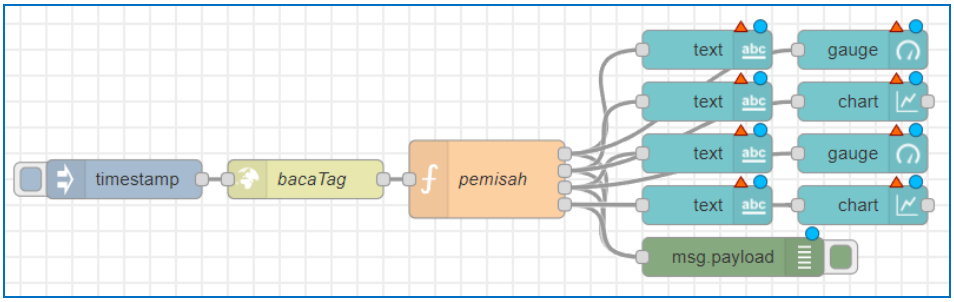
23. Hubungkan keempat output **node function** ke node **debug**. Tekan tombol Deploy, dan jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan nilai Tag. Kemudian tekan tombol di kiri **node inject**, dan perhatikan tampilan data Tag di kolom debug.



Gambar 6.58 Dengan node function pemisah, keempat Tag dipisahkan menjadi 4 saluran

24. Agar keempat Tag tersebut dapat ditampilkan di halaman web dengan tampilan user interface yang menarik, tambahkan node dashboard.

25. Klik tombol ≡, pilih Manage palette. Di kotak yang muncul, klik Tab Install. Di kolom search modules, ketik **dashboard**. Di daftar yang muncul, pilih **node-red-dashboard**, kemudian tekan tombol install. Klik Close.
26. Di kolom filter nodes, di pojok kiri atas, ketik **ui**. Muncul node-node dashboard. Ambil 4 buah **node text**, 2 buah **node chart** dan 2 buah **node gauge**. Hubungkan keempat **node text** dengan keempat output **node function** pemisah. Hubungkan 2 buah **node gauge** dengan output **node function** pertama dan ketiga. Hubungkan 2 buah **node chart** dengan output **node function** kedua dan keempat, seperti gambar berikut.



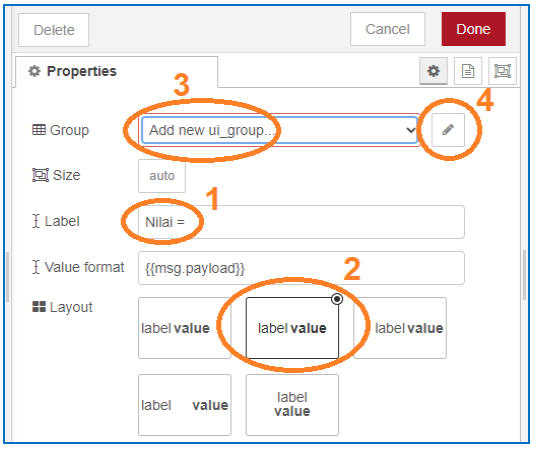
Gambar 6.59 Hubungkan 4 node text, 2 node gauge dan 2 node chart dengan node function

27. Klik 2 kali node-node dashboard, atur isian seperti tabel berikut:

Tabel 6.1 Pengaturan node-node dashboard Node-RED

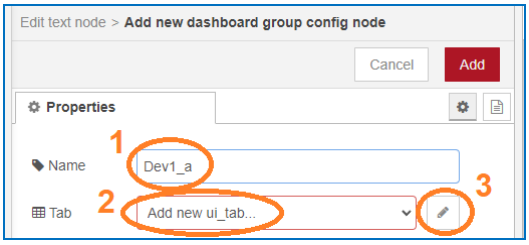
No.	Node	Tab	Group	Label	Lain-lain
1.	node text	HMI	Dev1_a	Nilai =	Layout: 1 baris, di tengah-tengah
2.	node text	HMI	Dev1_b	Nilai =	Layout: 1 baris, di tengah-tengah
3.	node text	HMI	Dev2_a	Nilai =	Layout: 1 baris, di tengah-tengah
4.	node text	HMI	Dev2_b	Nilai =	Layout: 1 baris, di tengah-tengah
5.	node gauge	HMI	Dev1_a	Level	Type: Level, Range 0 - 1023
6.	node gauge	HMI	Dev2_a	Level	Type: Level, Range 0 -1023
7.	node chart	HMI	Dev1_b	PWM	Type: Line Chart, x=5 mnt, y=1000
8.	node chart	HMI	Dev2_b	PWM	Type: Line Chart, x=5 mnt, y=1000

28. Sebagai contoh pengaturan, klik 2 kali **node text** yang di posisi teratas. Di kotak Edit, Label diisi **Nilai =** . Pilih Layout dengan tulisan di tengah, dan di kolom Group, pilih **Add new ui_group**, kemudian tekan tombol pensil.



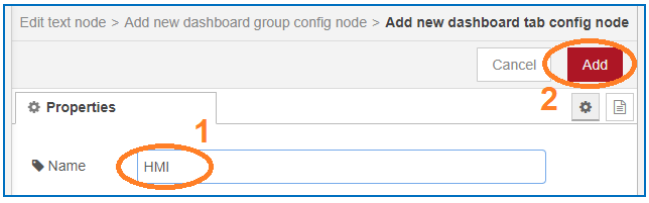
Gambar 6.60 Contoh pengaturan node text, isi dengan urutan sesuai gambar di atas

29. Muncul kotak pengaturan nama Group. Isi nama Group = **Dev1_a**. Di kolom Tab, pilih **Add new ui_tab**, kemudian tekan tombol pensil.



Gambar 6.61 Di kotak group, isi Name = Dev1_a, dan pilih Add new ui_tab, klik tombol pensil

30. Muncul kotak pengaturan nama Tab. Isi nama Tab = **HMI**, klik Add.

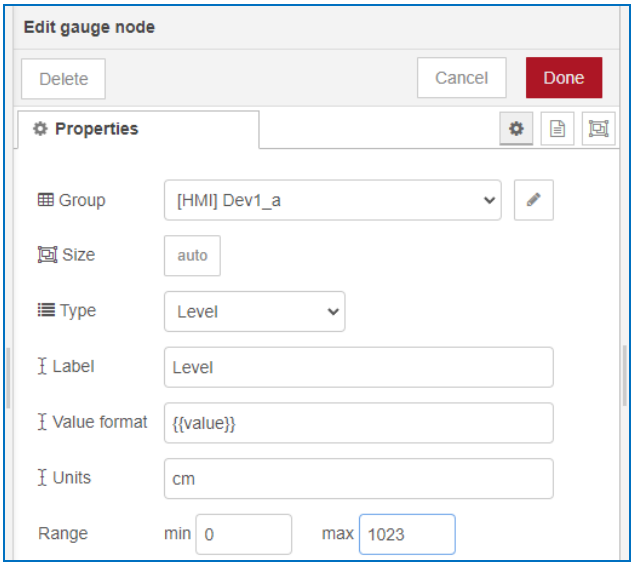


Gambar 6.62 Di kotak Tab, isi Name = HMI, klik tombol Add

31. Klik tombol Add sekali lagi di kotak Group, maka di kotak Edit akan muncul nama Tab dan Group: **[HMI] Dev1_a**. Klik Done untuk menutup kotak Edit.

Gambar 6.63 Setelah menambahkan nama Group = Dev1_a dan nama Tab = HMI, klik Done

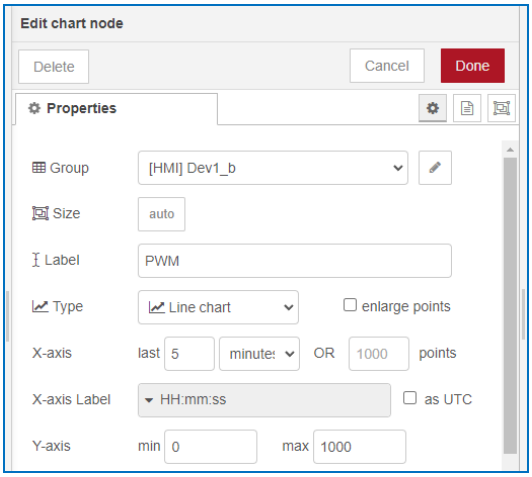
32. Untuk 3 **node text** yang lain, berturut-turut dari atas ke bawah, buat nama Group: **Dev1_b**, **Dev2_a**, **Dev2_b**, dengan nama Tab yang sama, **HMI**. Buat Label dan Layout ketiganya sama dengan node yang pertama.
33. Untuk **node gauge** yang pertama, pilih nama Group: **[HMI] Dev1_a**, Type = **Level**, Label = **Level**, Units = **cm**, Range min = **0**, max = **1023**.



Gambar 6.64 Untuk node gauge yang pertama, pilih nama Group = [HMI] Dev1_a

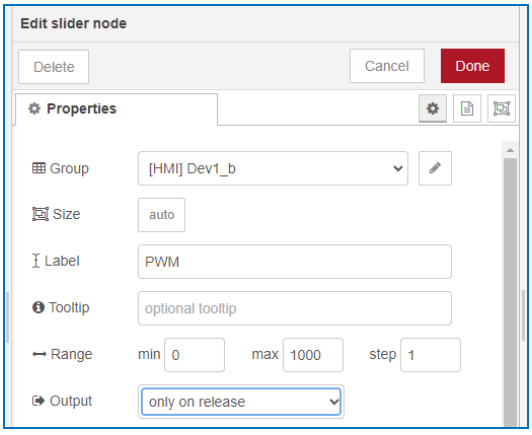
34. Buat **node gauge** yang kedua sama dengan yang pertama, hanya nama Groupnya berbeda. Pilih nama Group **node gauge** kedua = **[HMI] Dev2_a**.

35. Untuk **node chart** yang pertama, pilih nama Group: **[HMI] Dev1_b**, Label = **PWM**, Type = **Line chart**, X-axis last: **5 minutes**, Y-axis: **0** - **1000**.



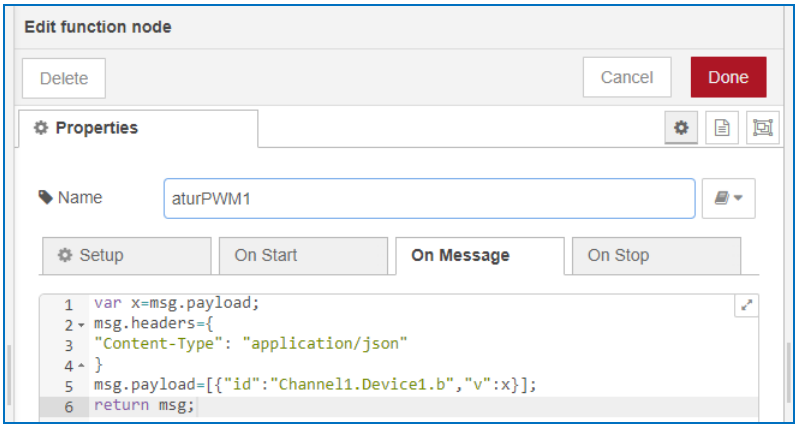
Gambar 6.65 Untuk node chart yang pertama, pilih nama Group = [HMI] Dev1_b

36. Buat **node chart** yang kedua sama dengan yang pertama, hanya nama Groupnya berbeda. Pilih nama Group **node chart** kedua = **[HMI] Dev2_b**.
37. Diinginkan menambahkan pengaturan Tag b di Device1 dan Device2 dari tampilan user interface. Untuk itu ambil 2 **node slider**. Atur **node slider** yang pertama. Pilih nama Group **node slider** pertama = **[HMI] Dev1_b**, Label = PWM, min = **0**, max = **1000**, Output = **only on release**.



Gambar 6.66 Untuk node slider pertama, pilih nama Group = [HMI] Dev1_b

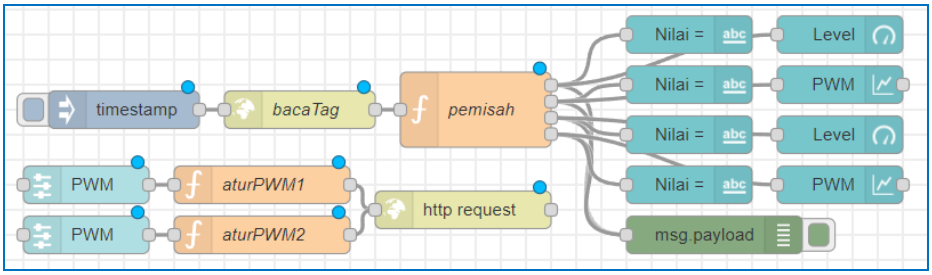
38. Buat **node slider** yang kedua sama dengan yang pertama, hanya nama Groupnya berbeda. Pilih nama Group **node slider** kedua = **[HMI] Dev2_b**.
39. Agar kedua node slider bisa mengubah nilai Tag, gunakan instruksi write seperti pada langkah no. 14, Gambar 6.51. Untuk itu tambahkan 2 node function. Hubungkan masing-masing ke output node slider. Klik 2 kali node function pertama, isi dengan kode program berikut ini:



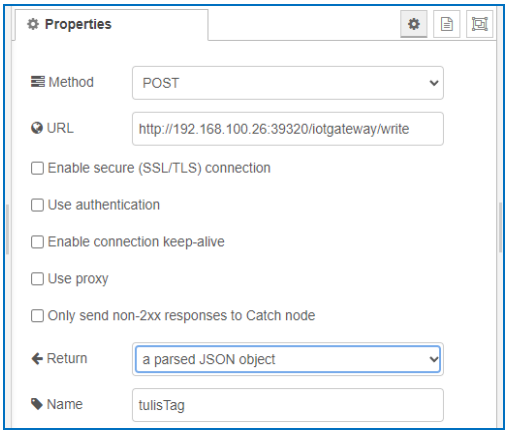
Gambar 6.67 Di kotak Edit, isi Name = aturPWM1, dengan kode program di On Message

	Kode program di kolom On Message
01.	`var x=msg.payload;`
02.	`msg.headers={`
03.	`"Content-Type":"application/json"`
04.	`}`
05.	`msg.payload=[{"id":"Channel1.Device1.b","v":x}];`
06.	`return msg;`

40. Ulangi kode program di atas untuk node function kedua, hanya ubah **Channel1.Device1.b** di baris kelima menjadi **Channel1. Device2.b**. Juga untuk membedakan, ubah nama **aturPWM1** menjadi **aturPWM2**.
41. Berikutnya, agar nilai dari **node slider** tersebut dapat dikirimkan ke REST Server melalui HTTP, tambahkan sebuah **node http request**. Hubungkan output dari kedua **node function** ke input **node http request**.
42. Klik 2 kali **node http request**, isi Method = **POST**, URL = **http://192.168.100.26:39320/iotgateway/write**, Return = **a parsed JSON object**, dan Name = **tulisTag** (sesuaikan alamat IP 192.168.100.26).

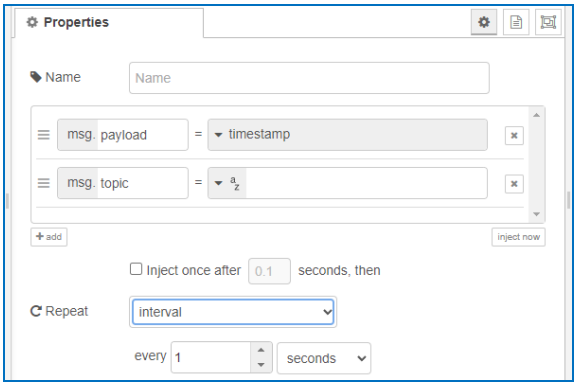


Gambar 6.68 Program (flow) keseluruhan: 4 node text, 2 node gauge dan 2 node chart untuk menampilkan data Tag di tampilan user interface, serta 2 node slider untuk mengubah Tag



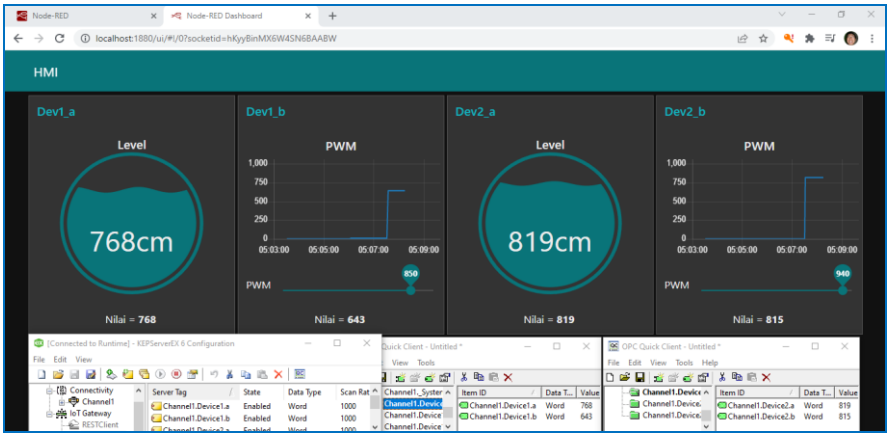
Gambar 6.69 Di node http request, Method=POST, URL sesuai REST API write, Name=tulisTag

43. Agar tampilan user interface selalu di-update per detik, klik 2 kali node inject (timestamp), di kolom Repeat, ubah **none** menjadi **interval**.



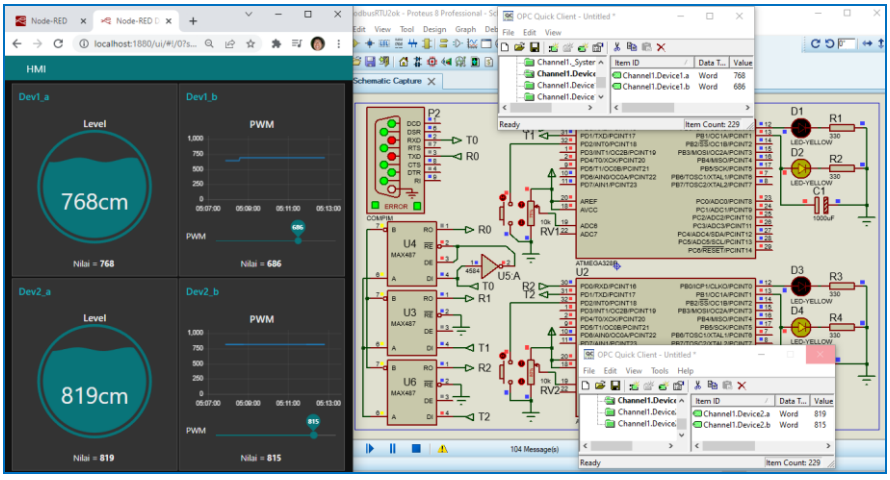
Gambar 6.69 Agar tampilan di-update setiap detik, buat Repeat = interval, setiap 1 detik

44. Tekan tombol Deploy. Jalankan OPC Quick Client dan rangkaian Proteus. Pastikan OPC Quick Client menampilkan data Tag. Kemudian buka browser yang baru, ketik: **localhost:1880/ui**. Maka muncul tampilan user interface dashboard Node-RED seperti gambar berikut:



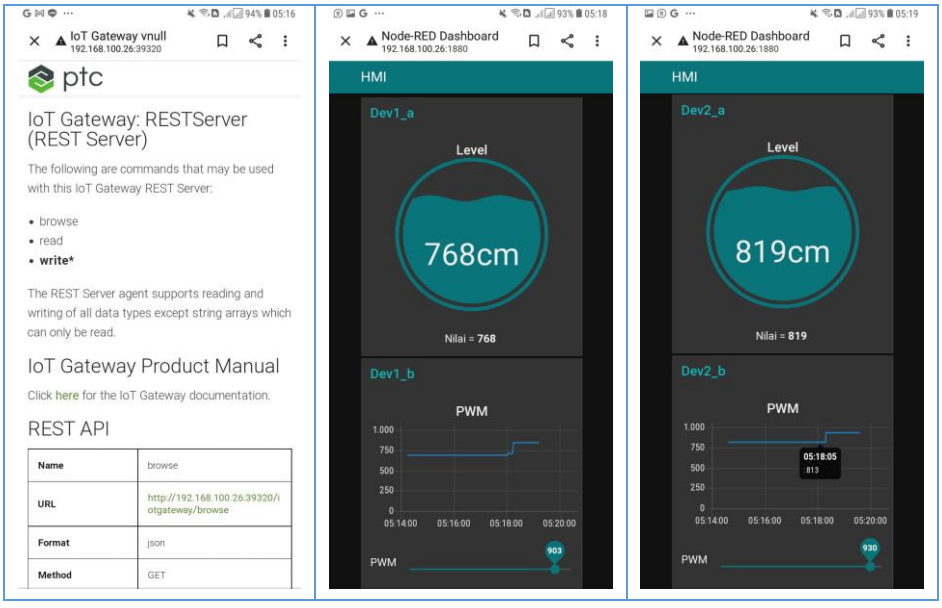
Gambar 6.70 Tampilan user interface dashboard Node-RED, di localhost:1880/ui

45. Lakukan pengubahan Potensio di rangkaian U1 dan U2 Proteus, dan perhatikan tampilan Level di Dev1_a dan Dev2_a. Kemudian lakukan pengubahan slider PWM di Dev1_b da Dev2_b, dan perhatikan nyala LED di rangkaian U1 dan U2 Proteus.



Gambar 6.71 User Interface Node-RED membaca nilai Potensio dan mengatur PWM di LED

46. Hal yang menarik dengan tampilan user interface dashboard Node RED ini adalah fleksibel. Jadi setiap kotak tampilannya dalam Group akan menyesuaikan posisinya setiap kali dilakukan pengubahan ukuran layar.
47. Hal menarik lainnya, apabila komputer pembaca menggunakan wifi yang sama jaringannya dengan HP pembaca, tampilan Node-RED tersebut dapat dibuka di HP, sehingga pengaturan juga dapat dilakukan lewat HP.



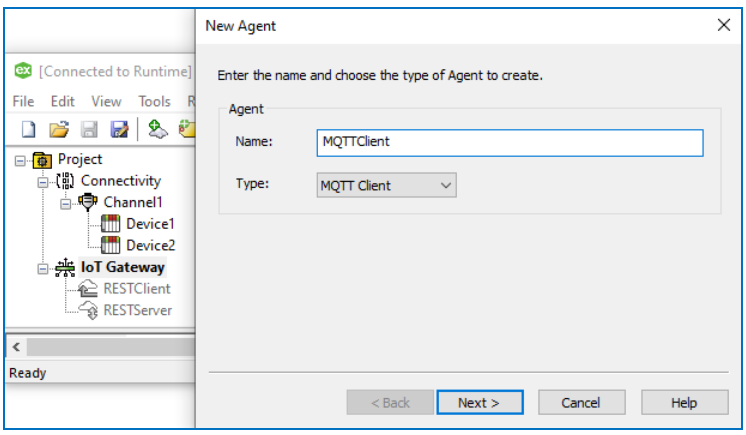
Gambar 6.72 Tampilan REST API dan ui Node-RED di HP, buka IP komputer di HP dengan tambahan port dan "/ui" (gunakan IP yang muncul di link URL REST Server Gambar 6.48)

48. Hal menarik berikutnya, pembaca dapat membuka, mengedit, membuat program (flow) dan men-Deploy serta menampilkan hasil program Node-RED yang ada di komputer dari HP pembaca, dengan cara memasukkan alamat IP komputer pada jaringan wifi lokal yang sama dengan HP.
49. Sampai di sini publikasi dan pengaturan Tag menggunakan REST Server KEPServerEX melalui HTTP, dan pembuatan tampilan user interface (HMI) dengan dashboard Node-RED. Kelemahan dari HTTP adalah ukuran data yang besar, yang membuat respon komunikasi menjadi lambat. Di Sub Bab berikut, akan diperkenalkan MQTT, yang berukuran kecil dengan respon komunikasi yang cepat, dan real time.

6.4 MQTT Client dan Node-RED

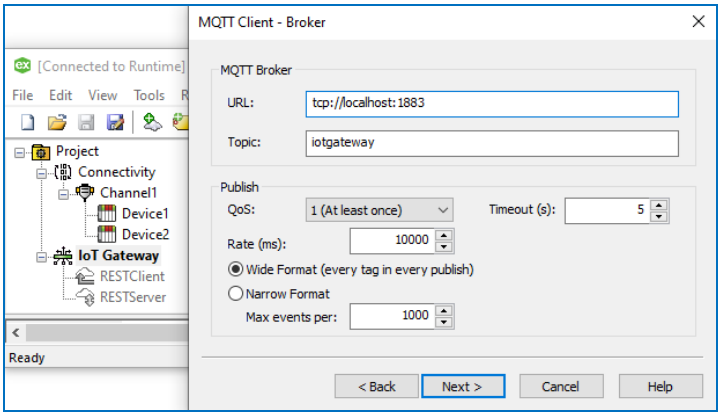
MQTT (*Message Queuing Telemetry Transport*) adalah komunikasi melalui TCP/IP, yang dirancang khusus untuk komunikasi *machine to machine* (M2M). Sistem kerja MQTT menerapkan prinsip Publish/Subscribe. Setiap alat dapat mem-publish sebuah topic dengan isinya, yang kemudian alat di tempat lain dapat men-subscribe topic tersebut kapanpun. Ada 4 istilah yang perlu dipahami di protokol MQTT, yaitu Publisher, Subscriber, Broker dan Topic. MQTT Broker bekerja sebagai makelar (Broker), yang akan meneruskan Topic dari Publisher ke semua Subscriber. Untuk bisa membandingkan hasil kerja MQTT dengan REST Server, di Sub Bab 6.4 ini, akan dilakukan hal yang sama seperti di Sub Bab 6.3, yaitu membuat tampilan user interface HMI dengan Node-RED, hanya bedanya, sebagai penghubung antara Node-RED dengan Device1 dan Device2, digunakan MQTT Client sebagai pengganti REST Server. Berikut ini langkah-langkahnya:

1. Buka KEPServerEX Configuration.
2. Klik kanan RESTServer di IoT Gateway, pilih Disable.
3. Klik kanan IoT Gateway, pilih New Agent.
4. Di kotak New Agent, isi Name = **MQTTClient**, pilih Type = **MQTT Client**.



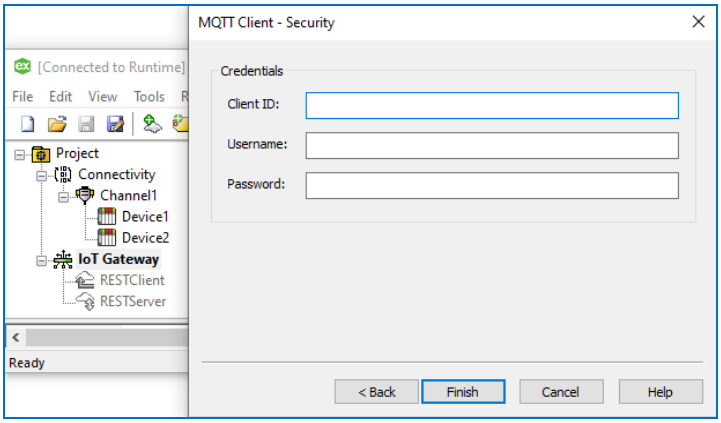
Gambar 6.73 Klik kanan IoT Gateway, pilih New Agent, di kotak New Agent, pilih MQTT Client

5. Klik Next. Ubah Narrow Format menjadi Wide Format. Isi URL di sini merupakan alamat MQTT Broker, gunakan default Broker localhost. Setelah MQTT bisa berjalan, nanti bisa diganti dengan Broker online.



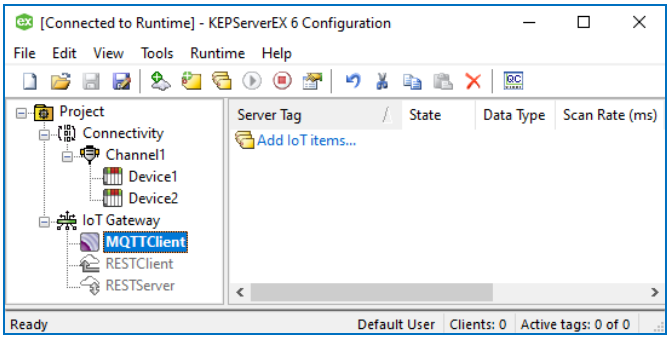
Gambar 6.74 Di halaman Broker, ubah Narrow Format menjadi Wide Format, sedangkan isian yang lain, gunakan default, dengan URL Broker = localhost:1880 dan Topic = iotgateway

6. Klik Next. Di halaman Security, untuk sementara isiannya dikosongkan. Apabila koneksi dengan MQTT sudah berhasil, silahkan pembaca mengisi ketiga kolom Security ini untuk membuat koneksi menjadi aman.

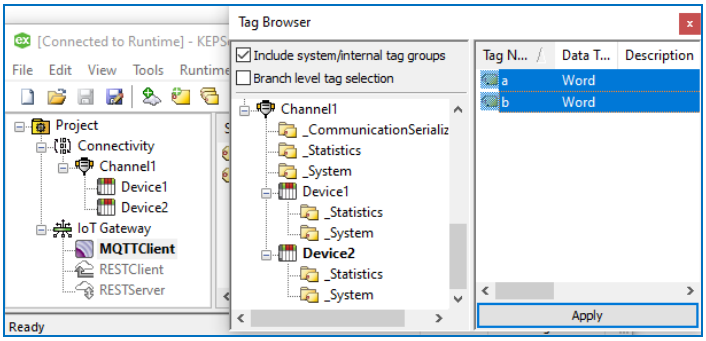


Gambar 6.75 Untuk sementara, ketiga kolom isian Credentials di halaman Security ini dikosongkan, setelah koneksi berhasil, silahkan diisi untuk meningkatkan keamanan

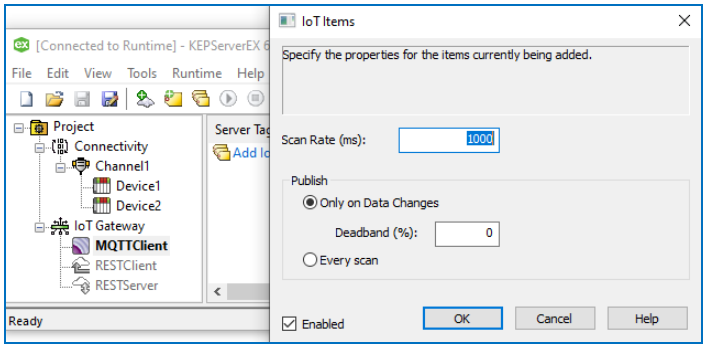
7. Klik Finish, maka muncul MQTTClient di IoT Gateway.
8. Klik Add IoT Items. Di kotak Tag Browser, pilih Device1, pilih Tag a dan b, tekan Apply. Pada kotak IoT Items yang muncul, gunakan seting default, klik OK. Ulangi untuk menambahkan Tag a dan b di Device2, sehingga ada 4 buah Tag di MQTTClient.



Gambar 6.76 Klik Finish, maka muncul MQTTClient di IoT Gateway

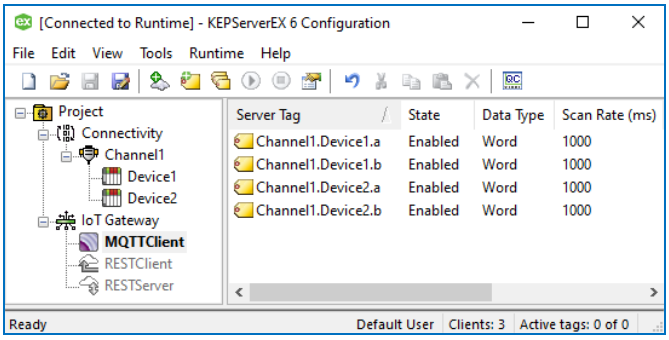


Gambar 6.77 Klik Add IoT Items, muncul Tag Browser, pilih Tag a dan b di Device1

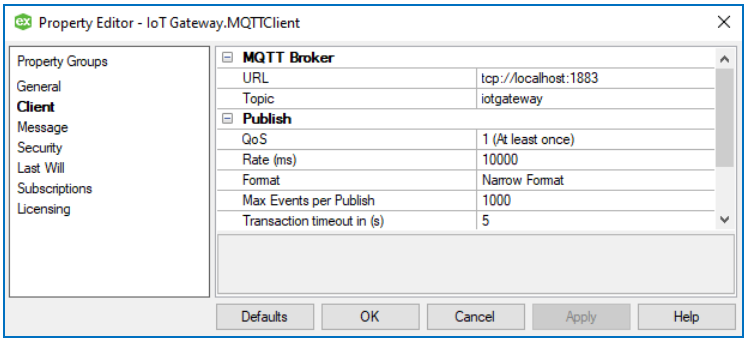


Gambar 6.78 Klik Apply, muncul kotak IoT Items, gunakan seting default, klik OK

9. Berikutnya, klik 2 kali MQTTClient hingga muncul kotak Property Editor. Ada 3 hal yang harus diperhatikan, yaitu alamat Broker, Topic untuk Publish dan Topic untuk Subcribe. Alamat Broker dan Topic untuk Publish ada di Property Editor Client. Secara default, Topic Publish = **iotgateway**.

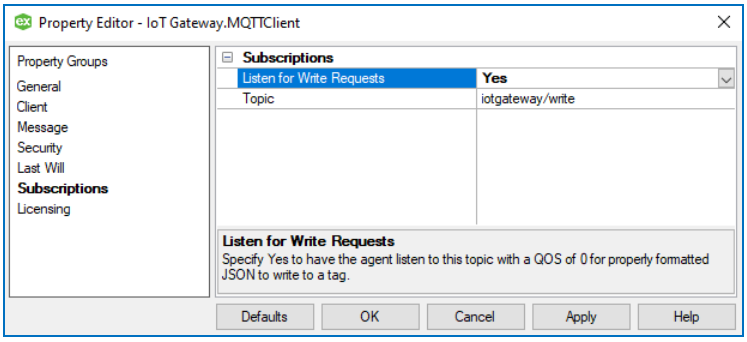


Gambar 6.79 Tambahkan Tag a dan b di Device2, sehingga di MQTTClient menjadi 4 Tag



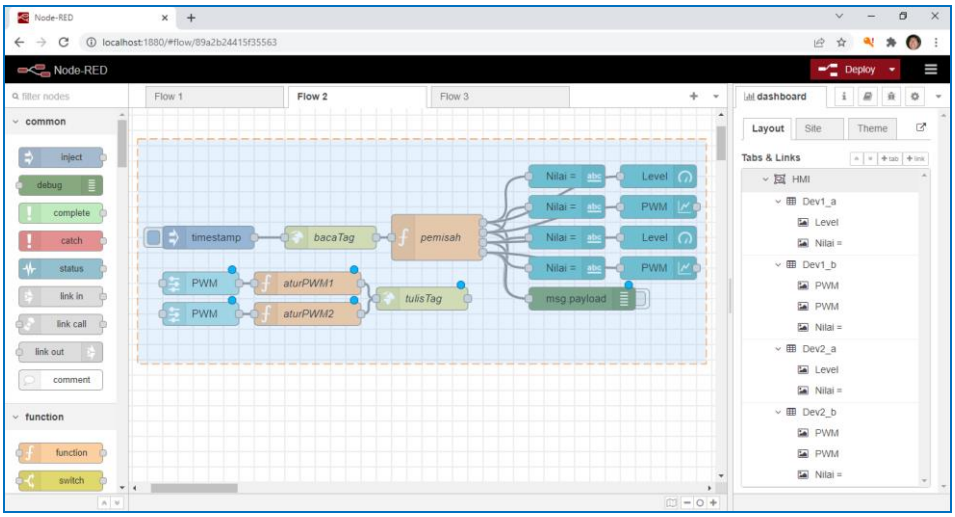
Gambar 6.80 Perhatikan alamat URL Broker dan Topic untuk Publish di Property Editor Client

10. Untuk Topic Subscribe, bisa dilihat di Property Editor Subscriptions. Secara default, Subscribe ini tidak aktif, untuk mengaktifkan, ubah isian Listen for Write Request, dari **No** menjadi **Yes**. Secara default, Topic Subscribe = **iotgateway/write**. Setelah koneksi MQTT berhasil, Topic ini dapat diubah.



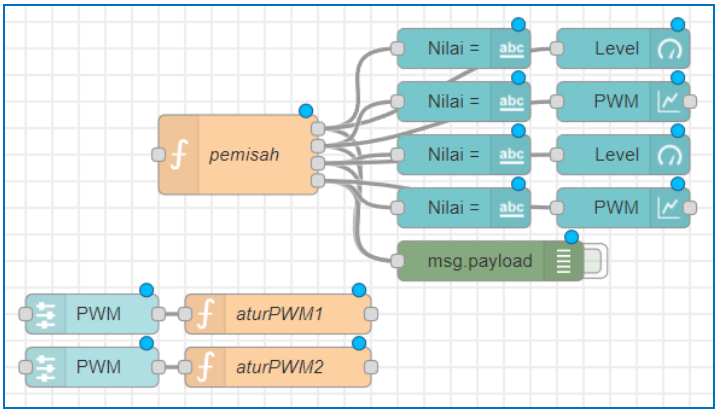
Gambar 6.81 Isi Listen for Write Request = Yes, Topic Subscribe = iotgateway/write

11. Klik OK. Pengaturan MQTTClient di KEPServerEX selesai. Berikutnya, buka Node-RED, dengan mengetikkan **node-red** di command prompt, dan mengetikkan **localhost:1880** di Browser.
12. Tambahkan Flow yang baru (Flow3). Pilih semua isi Flow2 (program Node-RED untuk REST Server), salin dan tempelkan (*copy paste*) di Flow3.



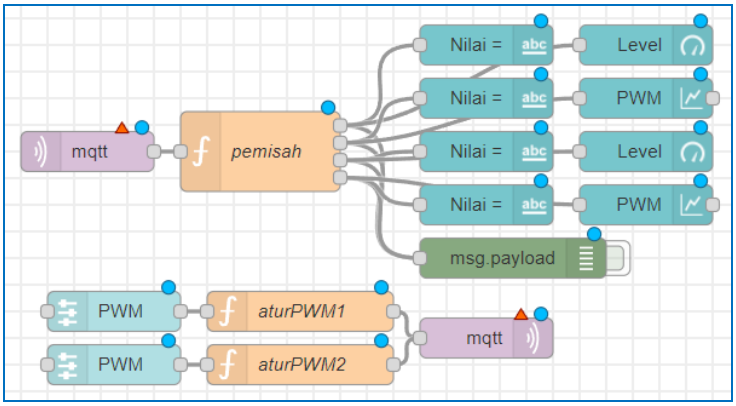
Gambar 6.82 Tambahkan Flow3, salin semua isi Flow2, dan tempel di Flow3

13. Berikutnya, setelah ditempel di Flow3, hapus **node inject** timestamp dan **node http request** bacaTag dan **node http request** tulisTag di Flow3.



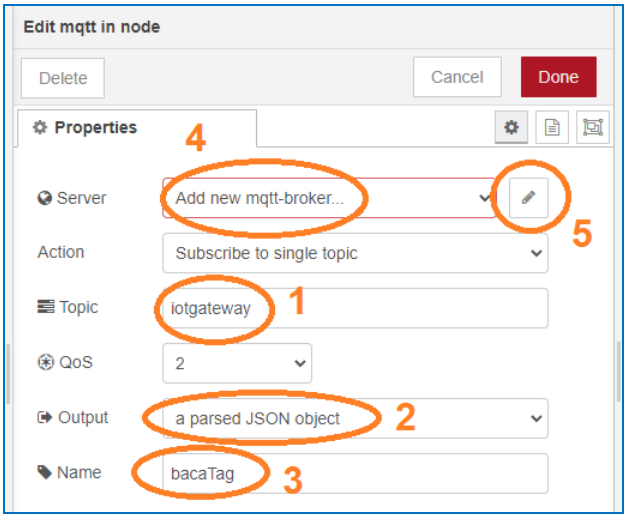
Gambar 6.83 Di Flow3, hapus node inject timestamp, node http request bacaTag dan tulisTag

14. Ambil **node mqtt in** dan hubungkan dengan **node function** pemisah.
15. Ambil **node mqtt out**, dan hubungkan output **node function** aturPWM1 dan atur PWM2 ke **node mqtt out**.



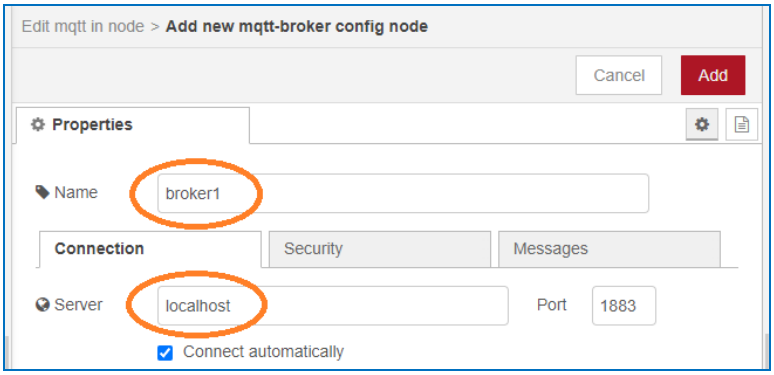
Gambar 6.84 Ambil node mqtt in dan node mqtt out, hubungkan node mqtt in ke node function pemisah, dan node function aturPWM1 dan aturPWM2 ke node mqtt out

16. Klik 2 kali **node mqtt in**, berturut-turut isi seperti gambar berikut, Topic = **iotgateway**, Output = a parsed JSON **Object**, Name = **bacaTag**, Server = **Add new mqtt-broker**, kemudian tekan tombol pensil.



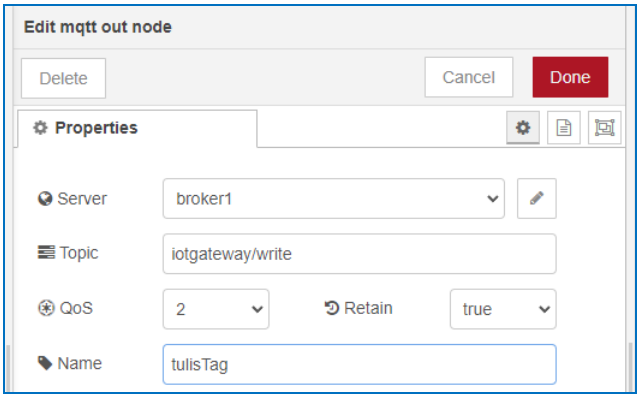
Gambar 6.85 Di node mqtt in, isi Topic = iotgateway (diisi dengan Topic Publish), Output = a parsed JSON object, Name = bacaTag, pilih Add new mqtt-broker, tekan tombol pensil

17. Muncul kotak pengaturan Broker. Isi Name = **broker1**, dan Server = **localhost**, kemudian tekan tombol Add.



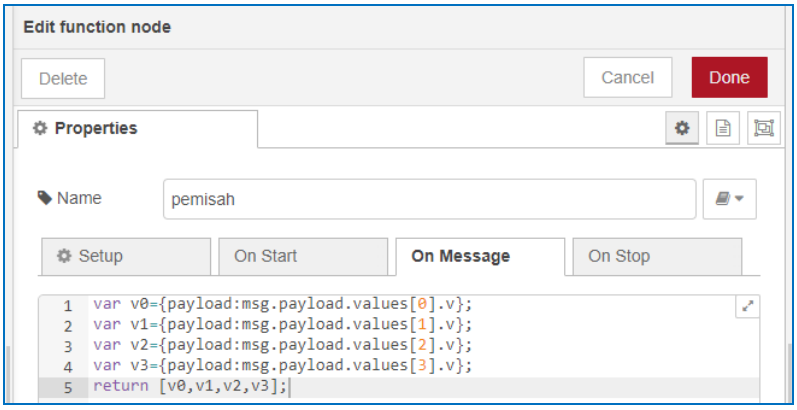
Gambar 6.86 Di kotak pengaturan Broker, isi Name = broker1, Server = localhost, klik Add

18. Kotak Edit node muncul kembali, dengan isian di kolom Server terisi **broker1**, yang merupakan nama broker untuk server **localhost**. Klik Done.
19. Berikutnya klik 2 kali **node mqtt out**, isi Server = broker1, Topic = iotgateway/write, QoS = 2, Retain = true, Name = tulisTag. Klik Done.



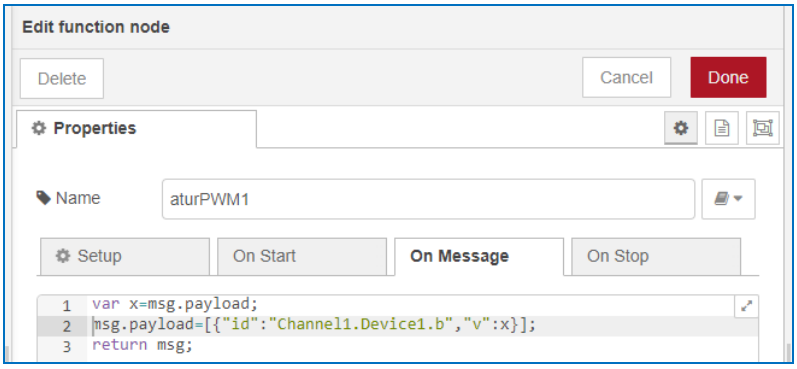
Gambar 6.87 Di node mqtt out, isi Server = broker1, Topic = iotgateway/write (diisi dengan Topic Subscribe), QoS = 2, Retain = True, Name = tulisTag, klik Done

20. Klik 2 kali pada **node function** pemisah. Di kode program di kolom On Message, ganti kata **readResults** di baris 1 – 4 dengan **values**. Klik Done.
21. Klik 2 kali pada node **function** aturPWM1. Hapus kode program dari baris 2 – 4, yang berisi msg.headers dengan isinya. Klik Done.



Gambar 6.88 Di node function pemisah, ganti kata readResults di baris 1-4 dengan kata values

	Kode program di kolom On Message
01.	`var v0={payload:msg.payload.values[0].v};`
02.	`var v1={payload:msg.payload.values[1].v};`
03.	`var v2={payload:msg.payload.values[2].v};`
04.	`var v3={payload:msg.payload.values[3].v};`
05.	`return [v0,v1,v2,v3];`

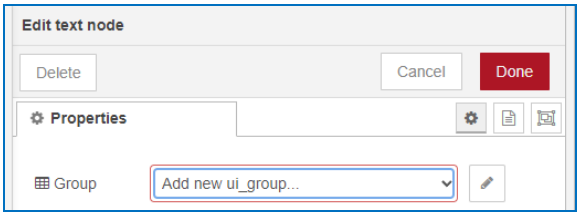


Gambar 6.89 Di Node function aturPWM1, hapus program msg.headers di baris ke 2-4

	Kode program di kolom On Message
01.	`var x=msg.payload;`
02.	`msg.payload=[{"id":"Channel1.Device1.b","v":x}];`
03.	`return msg;`

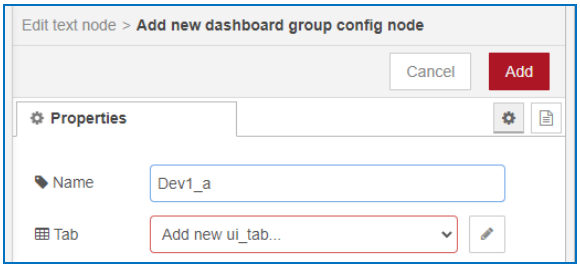
22. Ulangi untuk **node function** aturPWM2, hapus kode program msg.headers di baris ke 2-4. Klik Done.

23. Berikutnya, diinginkan untuk membuat tampilan user interface dengan 2 Tab (menu). Tab pertama diberi nama HMI, yang merupakan ui untuk RESTServer, sedangkan Tab kedua diberi nama HMI2, untuk tampilan ui MQTTClient. Untuk membuat 4 **node text**, 2 **node gauge**, 2 **node chart** dan 2 **node slider** di Flow3, yang saat ini berada di Tab HMI, menjadi berada di Tab HMI2, ikuti langkah-langkah berikut:
24. Klik 2 kali pada **node text** Nilai, yang di posisi teratas. Pada kotak Edit, di kolom Group, pilih **Add new ui_group**. Tekan tombol pensil.



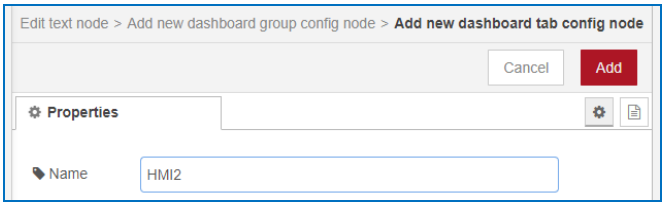
Gambar 6.90 Di kolom Group, pilih Add new ui_group, tekan tombol pensil

25. Muncul kotak pengaturan nama Group. Isi Name = **Dev1_a**. Kemudian di kolom Tab, pilih **Add new ui_tab**. Tekan tombol pensil.



Gambar 6.91 Di kotak pengaturan Group, isi Name = Dev1_a, dan pilih Tab Add new ui_tab

26. Muncul kotak pengaturan nama Tab. Isi Name = **HMI2**, klik tombol Add.



Gambar 6.92 Di kotak pengaturan Tab, isi Name = HMI2, klik tombol Add

27. Di kotak pengaturan nama Group, klik tombol Add lagi, maka di kotak Edit, di kolom nama Group, sekarang muncul nama **[HMI2] Dev1_a**. Klik Done.

Gambar 6.93 Setelah di pengaturan nama Group diisi Dev1_a dan di pengaturan nama Tab diisi HMI2, maka di kotak Edit, sekarang muncul nama [Group HMI2] Dev1_a

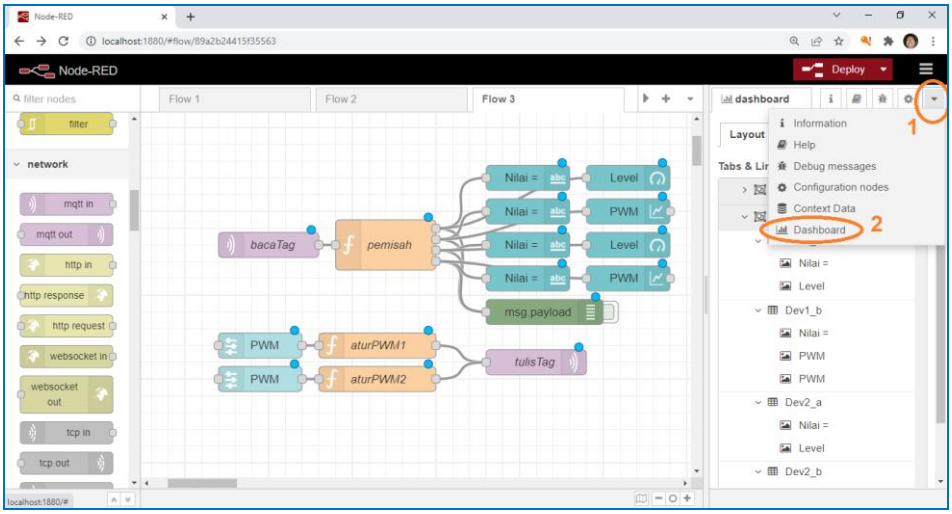
28. Ulangi langkah di atas untuk ketiga **node text** di bawahnya. Untuk pengaturan nama Group, silahkan diisi berturut-turut nama **Dev1_b**, **Dev2_a**, **Dev2_b**. Sedangkan untuk pengaturan nama Tab, karena nama HMI2 sudah dibuat, maka tidak perlu menambah Tab lagi, cukup memilih dari daftar yang muncul, dan pilih HMI2.
29. Berikutnya untuk 2 node gauge, 2 node chart, dan 2 node slider, karena nama Group dan Tab sudah dibuat, maka tidak perlu membuat lagi, cukup memilih dari daftar yang muncul, dan sesuaikan dengan Tabel berikut:

Tabel 6.2 Pengaturan nama Group node-node dashboard

No.	Nama Tab	Nama Group	Node-node Dashboard
1.	HMI2	Dev1_a	node text1 dan node gauge1
2.	HMI2	Dev1_b	node text2, node chart1, dan node slider1
3.	HMI2	Dev2_a	node text3, dan node gauge2
4.	HMI2	Dev2_b	node text4, node chart2, dan node slider2

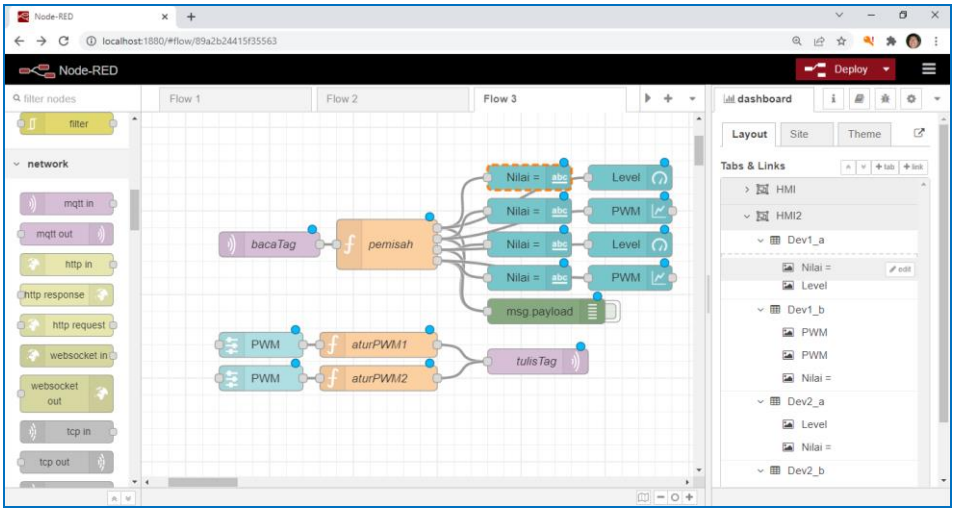
Catatan: Tambahan angka pada node menunjuk pada posisi node di program, angka 1 berarti posisi teratas, angka berikutnya berarti di bawahnya.

30. Untuk mengatur pengelompokan node dashboard, dapat juga dilakukan dengan membuka kolom dashboard. Caranya lihat gambar berikut:



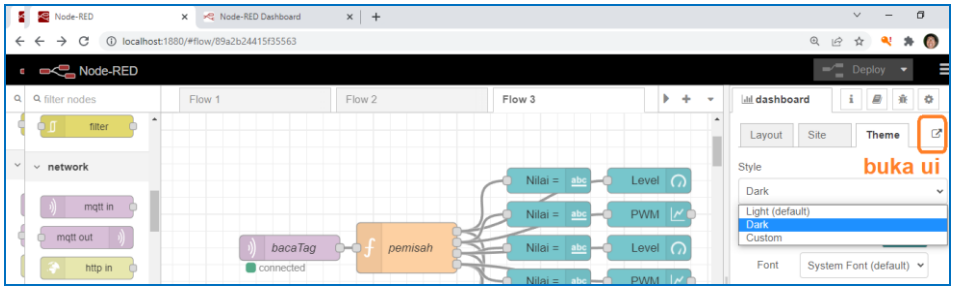
Gambar 6.94 Klik tombol panah ke bawah di pojok kanan atas, pilih Dashboard.

31. Maka terlihat daftar nama Tab, Group dan nama-nama node di setiap Group. Pengaturan dapat dilakukan dengan memindahkan (*drag*) nama-nama node tersebut ke posisi yang diinginkan. Seperti tampak pada gambar berikut, **node text** Nilai sedang dipindahkan. Di kolom dashboard ini bisa juga untuk menambah dan menghapus Tab, Group dan Link.



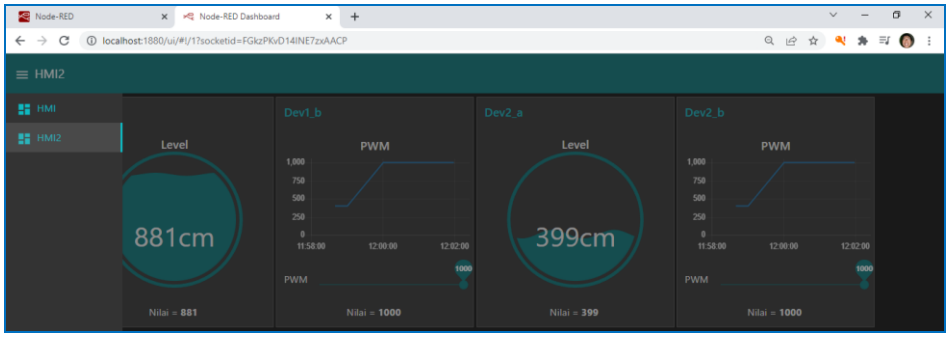
Gambar 6.95 Pengaturan Group dan Tab setiap node dapat dilakukan dengan mudah di kolom dashboard, dengan cara men-drag node tersebut ke posisi yang diinginkan

32. Untuk membuat tampilan ui dengan tema gelap, pilih Theme, dan pilih Dark di kolom Style. Untuk membuka tampilan ui, bisa juga dilakukan dengan cara meng-klik tombol yang diberi tanda pada gambar berikut ini.



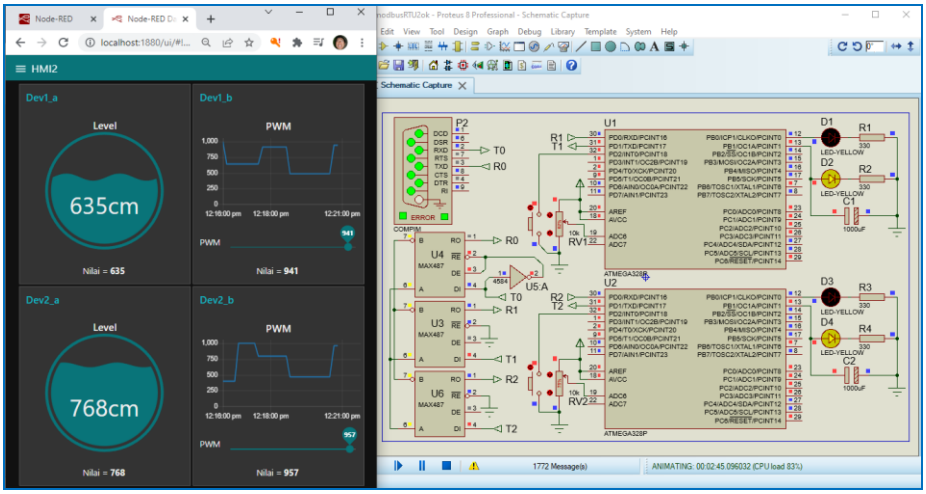
Gambar 6.96 Klik tombol yang diberi tanda untuk membuka ui

33. Tekan tombol Deploy, jalankan OPC Quick Client dan tampilan rangkaian Proteus. Pastikan OPC Quick Client menampilkan nilai Tag. Tekan tombol untuk membuka tampilan ui di kolom Dashboard. Di tampilan user interface, klik tombol ≡ di pojok kiri atas, dan pilih HMI2.



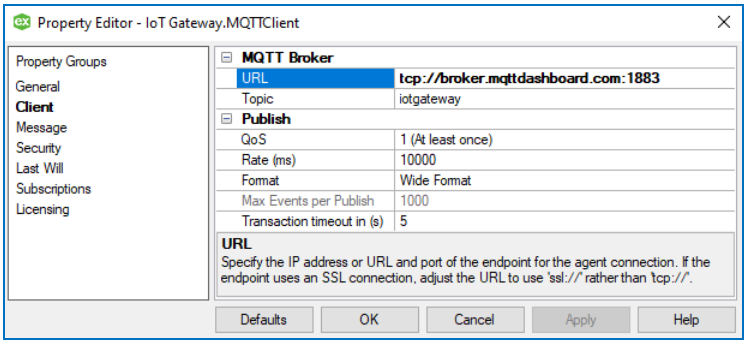
Gambar 6.97 Pilih HMI2, maka muncul tampilan ui dengan nilai keempat Tag dari MQTT Client

34. Lakukan pengubahan Potensio di rangkaian U1 dan U2, perhatikan Level Gauge dan nilainya di Dev1_a dan Dev2_a. Kemudian geser Slider di Dev1_b dan Dev2_b, perhatikan nyala LED PWM di rangkaian U1 dan U2.
35. Hal yang menarik dengan MQTT adalah, layanan MQTT Broker cukup banyak tersedia secara online dan gratis. Dengan demikian data yang dipublish dan disubscribe oleh tampilan ui HMI2 ini dapat ditampilkan di mana saja, asal bisa terhubung dengan MQTT Broker.



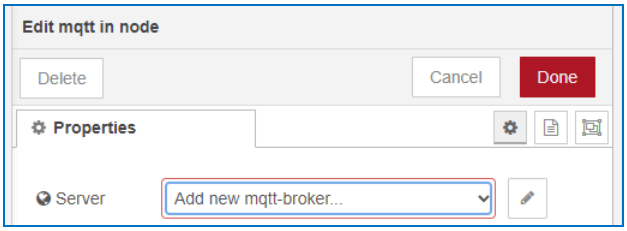
Gambar 6.98 Lakukan pengubahan kedua Potensio, dan perhatikan kedua Level di HMI2, berikutnya geser kedua slider di HMI2, dan perhatikan nyala kedua LED PWM di Proteus

36. Salah satu MQTT Broker online yang penulis gunakan adalah **broker.mqttdashbord.com**. Berikut ini cara mengganti broker localhost dengan MQTT Broker online. Klik 2 kali nama MQTTClient di KEPServerEX. Pada kotak Property Editor, pilih Client. Di MQTT Broker, isi kolom URL = **tcp://broker.mqttdashboard.com:1883**.

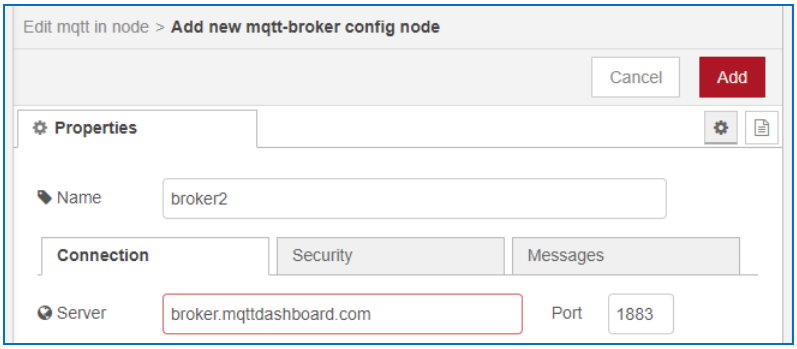


Gambar 6.99 Ganti Broker localhost dengan Broker online: broker.mqttdashboard.com

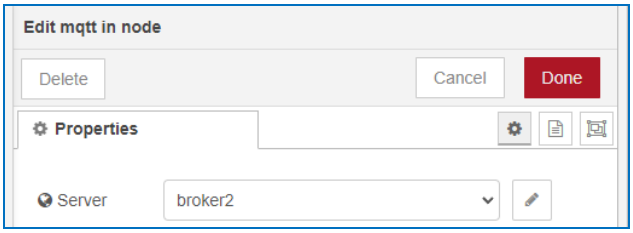
37. Berikutnya di Node-RED, klik 2 kali pada **node mqtt in** bacaTag. Di kotak Edit, di kolom Server, pilih **Add new mqtt-broker**, tekan tombol pensil.
38. Di kotak pengaturan broker, isi Server = **broker.mqttdashboard.com** dan isi Name = **broker2**. Klik Add, maka di kotak Edit muncul nama **broker2**.



Gambar 6.100 Di kotak Edit, pilih Add new mqtt-broker, tekan tombol pensil

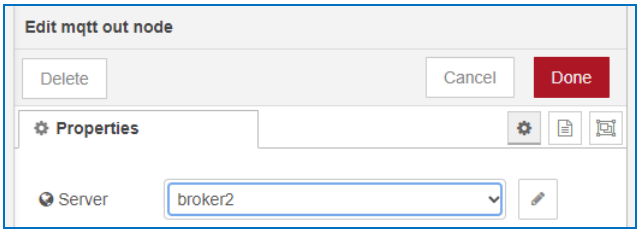


Gambar 6.101 Di pengaturan broker, isi Name=broker2, Server=broker.mqttdashboard.com



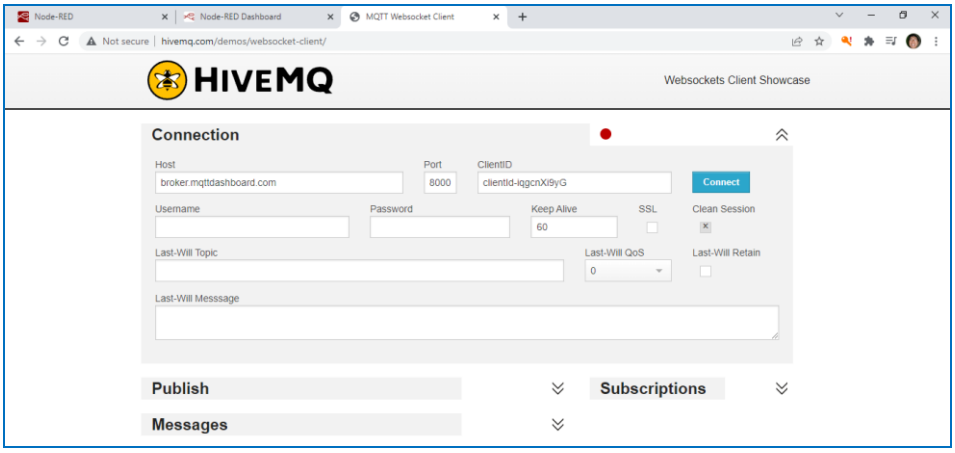
Gambar 6.102 Klik Add, maka muncul nama broker2 di kolom Server kotak Edit node mqtt in

39. Klik Done. Berikutnya, klik 2 kali pada **node mqtt out** tulisTag. Di kotak Edit, di kolom Server, pilih broker2 dari daftar yang muncul. Klik Done.



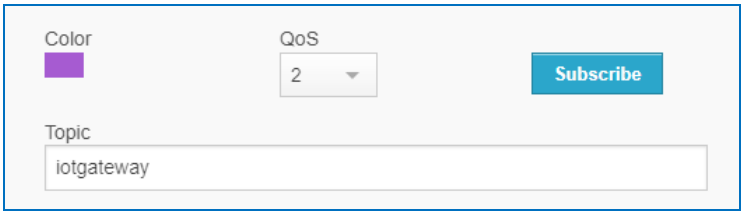
Gambar 6.103 Di kotak Edit node mqtt out, di kolom Server, pilih broker2 dari daftar

40. Berikutnya, pastikan komputer yang menjalankan Node-RED terhubung dengan internet. Tekan tombol Deploy, jalankan OPC Quick Client dan tampilan rangkaian Proteus. Pastikan OPC Quick Client menampilkan nilai Tag. Tekan tombol untuk membuka tampilan ui di kolom Dashboard. Di tampilan user interface, klik tombol ≡ di pojok kiri atas, dan pilih HMI2. Naik-turunkan Potensio di U1 dan U2, seharusnya kedua Level di HMI2 mengikuti nilai Potensi. Geser kedua Slider di HMI2, seharusnya LED PWM di U1 dan U2 menyala mengikuti nilai kedua Slider.
41. Untuk menunjukkan kemudahan komunikasi dengan MQTT, ketik "client hivemq" di browser, dan pilih **MQTT Websocket Client-HiveMQ**, maka akan muncul halaman Websocket-client HiveMQ. Tekan tombol Connect.

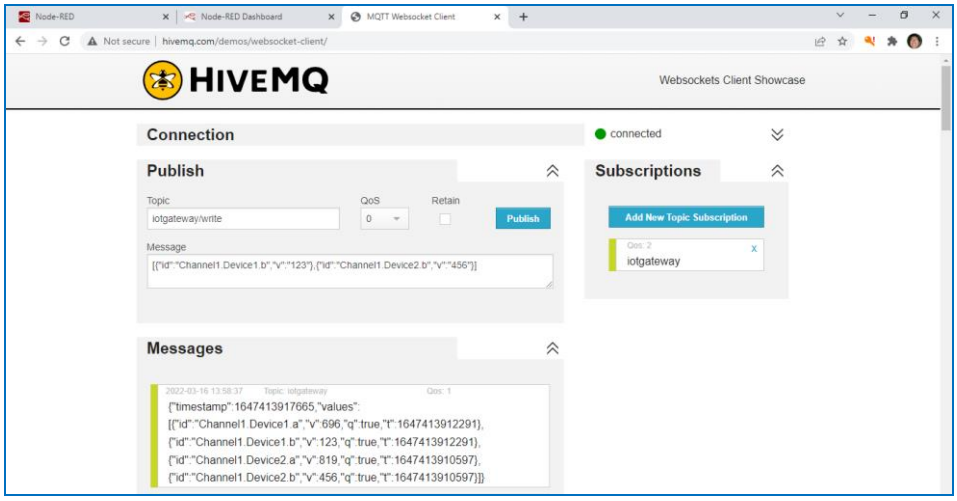


Gambar 6.104 Muncul halaman Websocket Client HiveMQ, tekan tombol Connect

42. Buka kolom **Subscriptions**. Klik tombol **Add New Topic Subscription**. Pada kotak yang muncul, isi Topic = **iotgateway**, tekan tombol Subscribe.

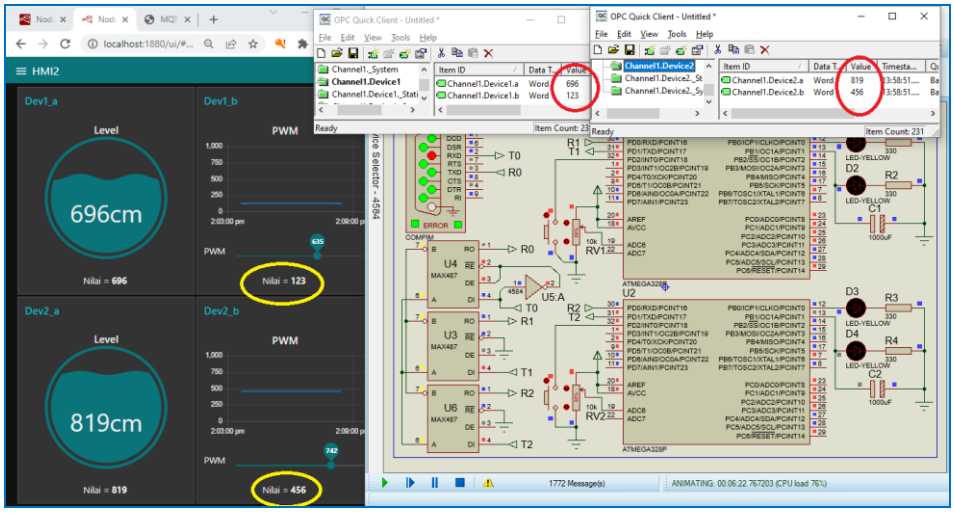


Gambar 6.105 Pada kotak yang muncul, isi Topic = iotgateway, tekan tombol Subscribe



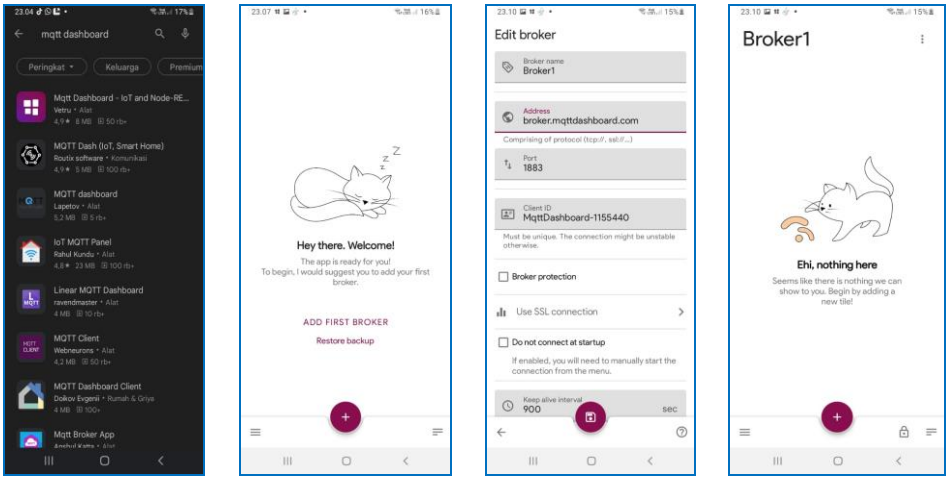
Gambar 6.106 Muncul halaman Websocket Client HiveMQ, tekan tombol Connect

43. Begitu tombol Subscribe ditekan, di kolom Messages muncul data keempat Tag yang dipublish oleh MQTTClient KEPServerEX.
44. Berikutnya, buka **Publish**, isi Topic = **iotgateway/write**, dan isi Message = **[{"id":"Channel1.Device1.b","v":"123"},{"id":"Channel1.Device1.b","v":"456"}]**. Kemudian tekan tombol Publish, maka data Tag b di Device1 dan Device2 berubah mengikuti nilai Tag yang dipublikasikan oleh HiveMQ.



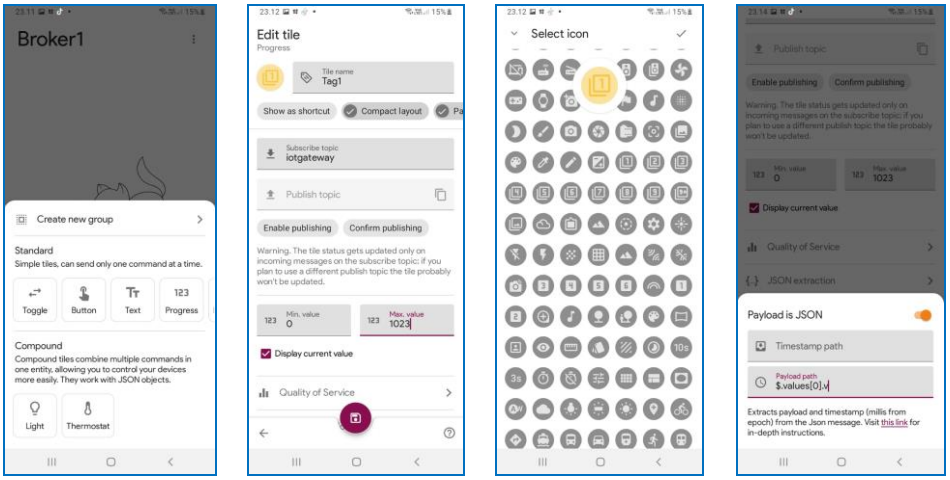
Gambar 6.107 Begitu tombol Publish ditekan, nilai Tag b di Device1 dan Device2 berubah

45. Selain menggunakan Websocket Client, tersedia juga aplikasi Android yang menyediakan layanan komunikasi MQTT yang gratis, salah satunya MQTT Dashboard - Vetru. Instal MQTT Dashboard - Vetru di Google Playstore, dan ikuti langkah-langkah berikut ini.
46. Setelah MQTT Dashboard diinstal, di halaman pertama tekan tombol + untuk mengatur alamat Broker. Isi Broker name = **Broker1**, dan Address = **broker.mqttdashboard.com**. Tekan Save, maka muncul halaman Broker1.



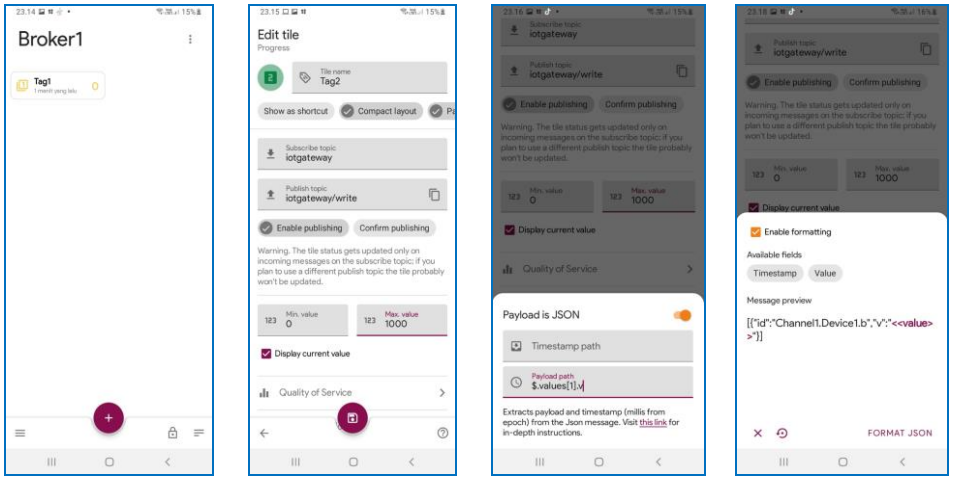
Gambar 6.108 Berturut-turut dari kiri ke kanan, (1) MQTT Dashboard-Vetru di Google Playstore, (2) tampilan pertama setelah diinstal, (3) Setelah tombol + ditekan, isi Broker name = Broker1, Address = broker.mqttdashboard.com, (4) Muncul Broker1 setelah Save ditekan

47. Tekan tombol + untuk menambahkan 4 Tag yang akan dibaca dan ditulis. Muncul pilihan tile, pilih tile Progress (123) di kategori Standard.
48. Muncul kotak Edit tile, isi Tile name = **Tag1**, Subscribe topic = **iotgateway**, Min. value = **0**, Max. value = **1023**.
49. Klik tombol tanda tanya di pojok kiri atas, pilih warna dan gambar icon. Klik tanda centang di pojok kanan atas untuk menutup Select icon.
50. Agar data JSON dari MQTTClient KEPServerEX bisa diterima hanya angka saja, maka di bagian bawah, di kolom JSON extraction, aktifkan **Payload is JSON**, dan isi di kolom Payload path = **$.values[0].v**. Tekan tombol navigation Back (<) untuk menutup kolom JSON.



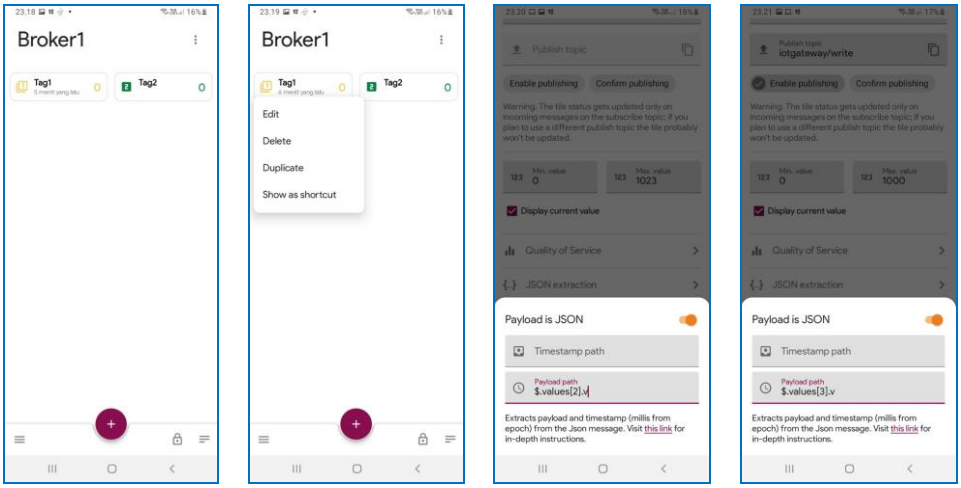
Gambar 6.109 Berturut-turut dari kiri ke kanan, langkah 47 - 50

51. Tekan tombol Save, maka di Broker1 muncul tile Tag1.
52. Tambahkan tile lagi dengan menekan tombol +. Di kotak Edit tile, isi Tile name = **Tag2**, isi Subscribe topic = **iotgateway**, isi Publish topic = **iotgateway/write**, centang **Enable publishing**, isi Min. = **0,** Max. = **1000**.
53. Di JSON Extraction, aktifkan **Payload is JSON**, isi Payload path = **$.values[1].v**. Di Edit output template, centang **Enable formatting**, Message preview = **[{"id":"Channel1.Device1.b","v":"<<value>>"}].**



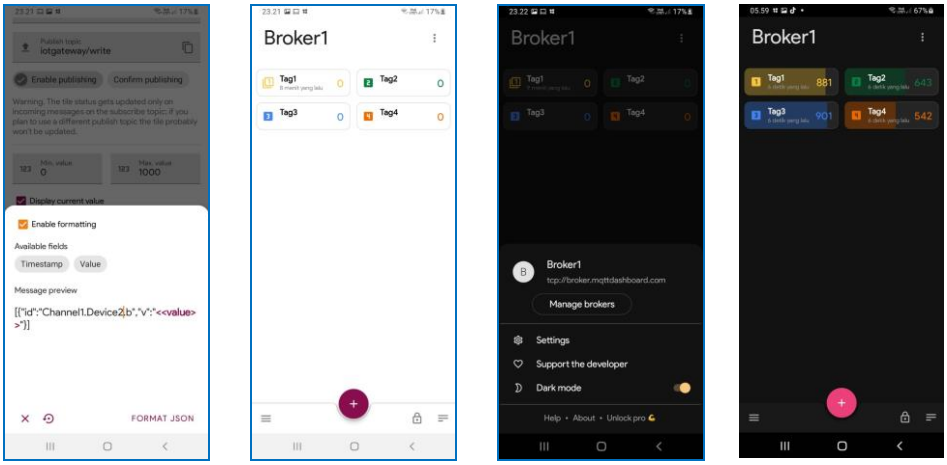
Gambar 6.110 Berturut-turut dari kiri ke kanan, langkah 51 - 53

54. Klik tombol navigation Back (<) untuk menutup Edit output template, kemudian tekan tombol Save, maka muncul tile Tag2 di Broker1.
55. Karena Tag3 sama seperti Tag1, maka klik kanan Tag1, dan pilih Duplicate. Begitu juga untuk Tag4, karena Tag4 sama seperti Tag2, maka klik kanan Tag2, dan pilih Duplicate. Jadi di Broker1 sudah ada 4 tile.
56. Perbaiki tile ketiga yang masih bernama Tag1, klik kanan, pilih Edit. di kotak Edit, ubah Tile name dari Tag1 menjadi Tag3. Di JSON Extraction, isi Payload path = **$.values[2].v**. Tutup JSON Extraction, dan tekan Save.
57. Perbaiki tile keempat yang masih bernama Tag2, klik kanan, pilih Edit. di kotak Edit, ubah Tile name dari Tag2 menjadi Tag4. Di JSON Extraction, isi Payload path = **$.values[3].v**. Tutup JSON Extraction dengan tombol Back.



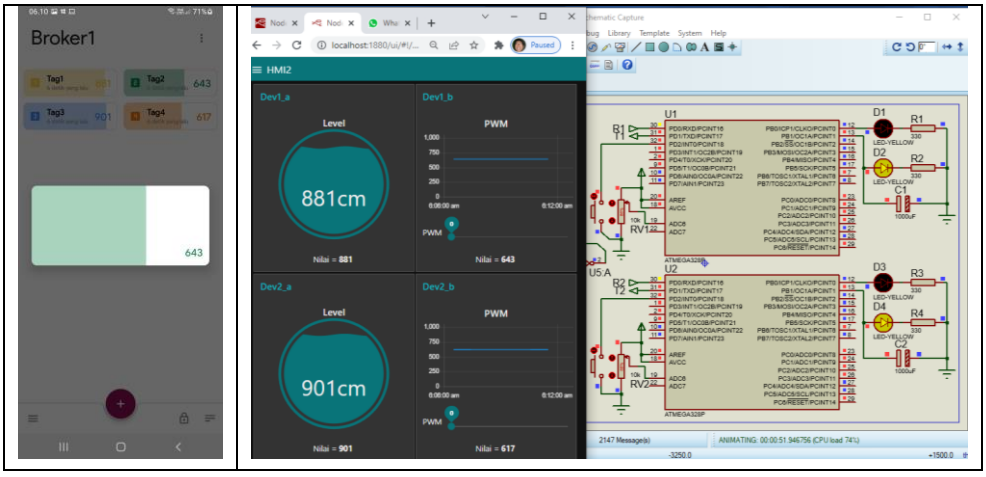
Gambar 6.111 Berturut-turut dari kiri ke kanan, langkah 54 - 57

58. Masih di tile keempat (Tag4), di Edit output template, isi Message preview = **[{"id":"Channel1.Device2.b","v":"<<value>>"}]**. Tutup dan tekan Save.
59. Maka di Broker1 akan ada 4 buah tile, dari Tag1 – Tag4.
60. Agar tampilan layar lebih menarik, ubah ke Dark mode dengan menekan tombol ≡ di pojok kiri bawah, aktifkan Dark mode.
61. Jalankan OPC Quick Client dan rangkaian Proteus. Juga tampilkan Node-RED Dashboard, pilih HMI2. Lakukan pengubahan Potensio di U1 dan U2, dan amati nilai tile di Tag1 dan Tag3 di Broker1.



Gambar 6.112 Berturut-turut dari kiri ke kanan, langkah 58 - 61

62. Di Broker1, lakukan pengubahan nilai tile Tag2 dan Tag4, dengan cara meng-klik tile hingga muncul kotak slider, geser ke nilai tertentu, dan perhatikan tampilan Node-RED Dashboard dan LED PWM di Proteus.



Gambar 6.113 Aplikasi MQTT Dashboard di Android dapat menampilkan dan mengubah nilai Tag dari MQTTClient KEPServerEX, yang ditampilkan di Node-RED Dashboard

63. Sampai di sini pembuatan tampilan user interface dengan Node-RED Dashboard, dengan data dari MQTTClient KEPServerEX. Telah ditunjukkan bagaimana menariknya komunikasi MQTT dibanding dengan REST Server, yaitu respon yang lebih cepat, dan dukungan aplikasi yang lebih banyak.

6.5 Penutup

Mengingat halaman buku yang sudah terlalu banyak, maka penulis terpaksa harus menutup di sini. Sebenarnya masih banyak hal yang belum tersampaikan dalam buku ini. Masih banyak hal yang menarik di "luar sana". Penulis bisa menulis buku ini karena banyaknya "orang-orang baik" di internet, yang membagikan ilmunya secara gratis. Penulis banyak belajar dari video-video di Youtube, yang menyajikan materi dengan sangat jelas dan menarik.

Bagi pembaca yang ingin mendalami SCADA, penulis sangat rekomendasikan untuk menguasai Node-RED, karena semua hardware dan software, serta layanan dan aplikasi online di web maupun di HP, bisa dihubungkan dengan Node-RED. Dengan OPC server seperti KEPServerEX, pembuatan SCADA memang menjadi mudah, namun masalahnya, KEPServerEX merupakan software berbayar, sementara Node-RED adalah software gratis berbasis Java Script. Sebenarnya pembuatan SCADA dalam buku ini bisa dibuat dengan hanya menggunakan Node-RED saja, tanpa perlu melibatkan OPC Server. Namun karena waktu dan tenaga penulis yang terbatas, dan jumlah halaman buku yang sudah terlalu banyak, maka penulis hanya bisa menyarankan pembaca untuk mempelajari pembuatan SCADA dengan Node-RED di Youtube. Banyak sekali video di Youtube yang mengajarkan cara pembuatan SCADA dengan Node-RED.

Terakhir, sebagai penutup, mengingat perkembangan teknologi Internet of Things – teknologi yang menghubungkan "segalanya", yang sudah luar biasa pesatnya, maka keamanan (*security*) sistem harus benar-benar diperhatikan. Tanpa keamanan yang baik, sistem koneksi yang canggih akan menjadi "malapetaka", karena dapat diretas. Terkait dengan sistem keamanan ini, ada 2 kata kunci yang penting, yaitu otentikasi dan otorisasi. Otentikasi adalah proses identifikasi apakah seorang user diijinkan untuk mengakses sistem. Otentikasi ini dapat diibaratkan seperti sebuah kunci untuk bisa membuka pintu sistem. Sedangkan otorisasi adalah proses verifikasi apakah seorang user berhak mengubah atau menerapkan aksi. Otorisasi ini dapat diibaratkan seperti kunci ke banyak ruangan. User dengan tingkat otorisasi rendah hanya bisa membuka sedikit ruangan, sedangkan user dengan tingkat otorisasi tinggi bisa membuka semua ruangan.

Otentikasi ini dapat pembaca temukan pada pengisian username dan password di setiap pembuatan aplikasi. Sedangkan Otorisasi dapat pembaca temukan di KEPServerEX, saat penggunaan username dan password Administrator untuk bisa menjalankan komunikasi OPC UA. Begitu juga untuk IoT Gateway, REST Client, REST Server dan MQTT Client, disarankan agar pembaca selalu menggunakan username dan password, agar tingkat keamanan sistem menjadi lebih baik.

Sampai di sini pembahasan mengenai SCADA dan IoT, selamat membuat aplikasi SCADA dan IoT yang menarik dan bermanfaat.

6.6 Soal Latihan

1. Apa yang Anda ketahui tentang Internet of Things?
2. Apa yang Anda ketahui tentang REST Client?
3. Apa yang Anda ketahui tentang REST Server?
4. Apa yang Anda ketahui tentang MQTT?
5. Ulangi Sub Bab 6.2, buatlah alarm yang dapat mengirimkan peringatan melalui email dan Whatsapp, dengan tambahan data Tag 2 buah tombol di rangkaian U1 dan U2 Gambar 4.51. Alarm untuk tombol ini hanya dikirimkan sekali ketika tombol ditekan, dan sekali ketika tombol dilepas. Sekalipun tombol ditekan terus, sekali sudah mengirimkan, tidak akan mengirimkan lagi.
6. Ulangi Sub Bab 6.3, buatlah tampilan user interface menggunakan Node-RED Dashboard, yang dapat menampilkan dan mengubah data Tag yang dipublikasikan oleh REST Server KEPServerEX, dengan tambahan data Tag 2 buah tombol dan data Tag 2 buah LED di rangkaian U1 dan U2 Gambar 4.51. Diinginkan data Tag kedua Tombol dan kedua LED dapat ditampilkan di tampilan ui Node-RED, di samping 4 data Tag yang sudah ada, dan juga diinginkan Tag kedua LED dapat dikontrol dari tampilan ui Node-RED.
7. Ulangi Soal Latihan no 6 di atas, namun sebagai ganti REST Server, gunakan MQTT Client KEPServerEX.

8. Ulangi Soal Latihan no. 7 di atas, diinginkan kedelapan data Tag di tampilan user interface Node-RED dapat juga ditampilkan dan dikontrol dari aplikasi MQTT Dashboard di HP Android.

9. Di Node-RED, buatlah tampilan user interface yang sama seperti tampilan Pengisian Tangki, di Gambar 1.1. Gunakan node-red-contrib-ui-svg (instal node di Manage palette) untuk membuat tampilan ui yang interaktif menggunakan file svg (vector graphics). Untuk mengetahui caranya, silahkan membuka youtube, dan mengetikkan kata kunci: node-red ui svg.

10. Buatlah aplikasi SCADA, yang bisa memonitor dan mengontrol sebuah alat, atau mesin atau sekelompok alat, contohnya seperti Home Automation, yang dapat dikontrol dan dimonitor dari tampilan web dan juga dari HP, yang dapat memberikan peringatan melalui email dan Whatsapp, dan menyimpan datanya di Database (pertimbangkan menggunakan file csv yang lebih ringan).

6.7 Refleksi

1. Menurut Anda, dari isi yang diuraikan, apakah **Target Materi** dari Bab 6 buku ini tercapai? Jika belum tercapai, apakah ada kesulitan dalam memahami materi yang berkaitan dengan Target Materi tersebut? Apakah Anda memiliki saran dan masukan untuk memperbaiki materi tersebut?

2. Apakah **Tantangan** dalam Bab 6 buku ini dapat Anda selesaikan? Jika belum, kesulitan apa yang membuat Anda tidak bisa menyelesaikannya? Apakah Anda mencoba alternatif lain untuk menemukan sendiri cara penyelesaiannya (mencari di Google misalnya)?

3. Apakah ada **Manfaat** yang Anda dapatkan setelah mempelajari Bab 6 dari buku ini? Apakah ada yang ingin Anda pelajari lebih lanjut? Apakah ada **Ide** yang menarik yang ingin Anda kembangkan?

DAFTAR PUSTAKA

[1]. Buku manual IoT Gateway, Kepware, PTC Inc., 2022, diunduh di www.kepware.com/en-us/support/manuals.

[2]. Buku manual Datalogger, Kepware, PTC Inc., 2022, diunduh di www.kepware.com/en-us/support/manuals.

[3]. Buku manual OPC UA, Kepware, PTC Inc., 2022, diunduh di www.kepware.com/en-us/support/manuals.

[4]. Buku manual Modbus Serial, Kepware, PTC Inc., 2022, diunduh di www.kepware.com/en-us/support/manuals.

[5]. Buku manual KEPServerEX, Kepware, PTC Inc., 2022, diunduh di www.kepware.com/en-us/support/manuals.

[6]. Taiji HagIno, Practical Node-RED Programming, 2021, Packt Publishing.

[7]. Agung Bakhtiar, Panduan Dasar Outseal PLC, 2020, www.outseal.com.

[8]. Gaston C. Hillar, MQTT Essentials - A Lighweight IoT Protocol, 2017, Packt Publishing.

[9]. Behzad Ehsani, Data Acquisition using LabVIEW, 2017, Packt Publishing.

[10]. Ronald W Larsen, LabVIEW for Engineers, 2011, Pearson Inc.