KONFIGURASI MIKROKONTROLER STM32 UNTUK MEMBACA PUSH BUTTON DENGAN ARDUINO IDE PADA PROTOTIPE SMART CHARGER DI PT. PASIFIK SATELIT NUSANTARA

Fajar Eka Maulana, Lela Nurpulaela
Teknik Elektro, Universitas Singaperbangsa Karawang
Jl. HS. Ronggo Waluyo, Puseur Jaya, Telukjambe Timur, Karawang, Jawa Barat 41361
Fajareka692@gmail.com

ABSTRAK

Mikrokontroler STM32, bagian dari keluarga ARM Cortex-M, telah menjadi populer dalam berbagai aplikasi elektronik karena kinerjanya yang tinggi, fleksibilitas, dan harga yang terjangkau. Permasalahan utama yang dihadapi adalah ketidakstabilan pembacaan sensor arus dan tegangan, yang dapat menyebabkan fluktuasi output daya dan potensi kerusakan perangkat. Untuk mengatasi hal ini, digunakan software Terminal untuk membaca data sensor serta push button untuk mengatur aliran listrik secara manual. Penelitian ini bertujuan untuk mengonfigurasi mikrokontroler STM32 pada prototipe smart charger di PT. Pasifik Satelit Nusantara agar mampu membaca input dari push button dengan menggunakan Arduino IDE. Metodologi yang digunakan mencakup pemilihan board STM32 yang sesuai, instalasi dukungan STM32 di Arduino IDE, serta implementasi konfigurasi pin dan teknik debouncing untuk memastikan pembacaan input yang akurat. Hasil penelitian menunjukkan bahwa konfigurasi yang tepat memungkinkan mikrokontroler STM32 untuk mengendalikan dan memantau proses pengisian daya dengan efisien. Penelitian ini menyimpulkan bahwa mikrokontroler STM32 dapat diandalkan untuk aplikasi smart charger, memberikan kemudahan bagi pengembang dalam memanfaatkan kemampuan canggih STM32 tanpa memerlukan toolchain yang kompleks.

Kata kunci : *Mikrokontroler STM32, Arduino IDE, smart charger, push button*

1. PENDAHULUAN

Mikrokontroler Mikrokontroler STM32, yang merupakan bagian dari keluarga mikrokontroler ARM Cortex-M, telah mendapatkan popularitas luas dalam berbagai aplikasi elektronik karena performanya yang tinggi, fleksibilitas, dan harga yang terjangkau. Salah satu aplikasi penting dari mikrokontroler ini adalah pada alat *smart charger*, di mana kebutuhan untuk mengontrol dan memantau proses pengisian daya secara efisien menjadi sangat krusial. Dalam konteks ini, kemampuan untuk membaca input dari push button menjadi salah satu fungsi dasar yang diperlukan untuk interaksi pengguna dan pengaturan mode operasi smart charger. Software Arduino IDE digunakan untuk memprogram mikrokontroler STM32 menawarkan pendekatan yang *user-friendly*, memungkinkan pengembang untuk memanfaatkan kekuatan STM32 dengan lebih mudah.[1].

Arduino IDE menawarkan lingkungan yang sederhana dan *user-friendly* untuk memprogram berbagai jenis mikrokontroler, termasuk STM32. Meskipun STM32 dan platform Arduino pada dasarnya berbeda dalam hal arsitektur dan kemampuan, didalamnya telah tersedia berbagai *library* dan *toolchain* yang memungkinkan STM32 untuk diprogram menggunakan Arduino IDE. Ini membuka peluang bagi pengembang untuk mengakses kemampuan canggih STM32 tanpa harus belajar *toolchain* yang lebih kompleks dan khusus [2].

Dalam konfigurasi mikrokontroler STM32 untuk membaca *push button*, langkah pertama adalah memilih *board* STM32 yang tepat dan menginstal dukungan STM32 di Arduino IDE. Ada berbagai model STM32 yang tersedia, seperti STM32F103, STM32F407, dan lainnya, masing-masing dengan fitur dan spesifikasi yang berbeda. Untuk pemula, STM32F103 ("*Blue Pill*") adalah pilihan populer karena ketersediaannya yang luas dan dokumentasi yang komprehensif[3].

Smart charger modern memerlukan kontrol yang cermat atas proses pengisian daya untuk memastikan efisiensi, keamanan, dan umur panjang baterai. Penggunaan *push button* pada alat ini memungkinkan pengguna untuk mengatur mode pengisian, memulai atau menghentikan proses pengisian, serta memilih parameter yang ingin ditampilkan. Untuk itu, mikrokontroler STM32 harus dikonfigurasi dengan tepat untuk membaca dan merespon input dari push button.

Membaca *push button* pada STM32 melalui Arduino IDE memerlukan beberapa konsep pemrograman mikrokontroler seperti konfigurasi pin, pembacaan digital, dan *debouncing*. Konfigurasi pin dilakukan dengan menggunakan fungsi pinMode() untuk menentukan pin yang terkait dengan tombol yang akan dimasukkan. Pembacaan digital dilakukan menggunakan fungsi *digitalRead*(). Hal ini memungkinkan mikrokontroler mendeteksi keadaan logika pin (HIGH atau LOW) ketika *push button* ditekan [4], [5].

Debouncing ialah teknik penting dalam membaca input dari push button. Push button mekanis sering menghasilkan *noise* atau getaran saat ditekan atau dilepaskan, yang dapat menyebabkan pembacaan

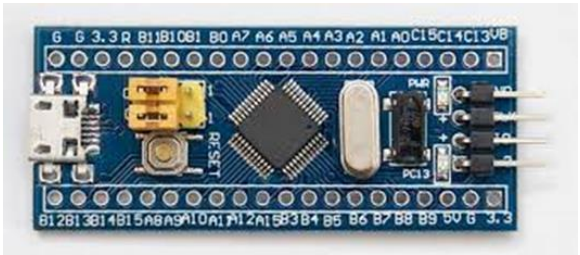
input yang tidak stabil. Untuk mengatasi masalah ini, teknik debouncing diterapkan untuk memastikan bahwa hanya pembacaan yang stabil yang diproses oleh mikrokontroler. Debouncing dapat dilakukan secara perangkat keras dengan menggunakan kapasitor atau secara perangkat lunak dengan menggunakan penundaan dan pengkondisian logika [6].

Jurnal ini akan menjelaskan setiap langkah yang diperlukan untuk mengonfigurasi STM32 untuk membaca push button menggunakan Arduino IDE termasuk penulisan kode. Dengan memahami dan mengikuti langkah-langkah ini, penulis dapat dengan mudah mengimplementasikan interaksi push button dalam projek – projek yang menggunakan mikrokontroler STM32, dan bermafaat dalam pengembangan berbagai aplikasi.

2. TINJAUAN PUSTAKA.

Pada penelitian yang berjudul ” Implementation of Smart Electric Vehicle Charging Station Driven Using Experimental Investigation”, Penelitian ini bertujuan untuk mengembangkan sistem pengisian daya pintar untuk kendaraan listrik yang dapat meningkatkan efisiensi dan keamanan pengisian daya. Fokus utama adalah pada penggunaan berbagai mikrokontroler dan algoritma kontrol untuk mencapai tujuan ini. Dengan menggunakan mikrokontroler yang tepat dan algoritma kontrol yang efektif, sistem pengisian daya pintar untuk kendaraan listrik dapat ditingkatkan secara signifikan. Ini tidak hanya meningkatkan efisiensi dan keamanan tetapi juga memberikan solusi praktis untuk tantangan pengisian daya yang ada saat ini [7]. Penelitian ini memberikan dasar yang kuat untuk pengembangan prototipe smart charger di PT. Pasifik Satelit Nusantara. Dengan menggunakan mikrokontroler STM32 dan konfigurasi yang tepat melalui Arduino IDE, pengembang dapat mengoptimalkan sistem pengisian daya dengan cara yang efisien dan aman.

2.1. STM32



Gambar 1. Mikrokontroller STM32

STM32 adalah keluarga mikrokontroler 32-bit yang dibuat oleh *STMicroelectronics*. Mikrokontroler ini khusus dirancang untuk aplikasi *Cortex-A* (aplikasi umum), *Cortex-M* (*embedded*), dan *Cortex-R* (*real-time*). Keunggulan utama STM32 terletak pada kombinasi antara performa tinggi dan periferal berkualitas tinggi, menjadikannya lebih unggul dibandingkan *Arduino* dalam hal kinerja.

Mikrokontroler ini sangat cocok untuk diterapkan dalam proyek-proyek yang bersifat terbenam (*embedded*). STM32 telah digunakan dalam beragam aplikasi, mulai dari perangkat pencetak sederhana hingga pengembangan perangkat lunak sistem dan sistem terbenam yang kompleks.

STM32 memiliki arsitektur Harvard yang memisahkan memori program dan memori data. Hal ini memungkinkan akses memori yang lebih cepat dan meningkatkan kinerja. Mikrokontroler ini juga dilengkapi dengan berbagai fitur periferal yang memperluas fungsinya, seperti:

a. Timer dan counter untuk mengukur waktu dan frekuensi
b. Antarmuka komunikasi serial seperti UART, SPI, dan I2C untuk terhubung dengan perangkat lain
c. Konverter analog-ke-digital (ADC) dan konverter digital-ke-analog (DAC) untuk memproses sinyal analog
d. Pengontrol DMA untuk transfer data yang efisien
e. Unit manajemen memori (MMU) untuk manajemen memori yang lebih canggih

Spesifikasi STM32 bervariasi tergantung pada seri dan modelnya. Berikut adalah beberapa spesifikasi umum STM32:

a. Inti: ARM Cortex-M0, M1, M3, M4, M7, M10
b. Frekuensi: 8 MHz hingga 216 MHz
c. Memori: Flash hingga 2 MB, RAM hingga 512 KB
d. Periferal: Timer, ADC, DAC, UART, SPI, I2C, DMA, MMU, dan banyak lagi
e. Tegangan: 1.6V hingga 3.3V
f. Suhu Operasi: -40°C hingga 85°C

Mikrokontroler ini memiliki keunggulan khusus dalam aplikasi yang membutuhkan daya rendah. Dengan fitur-fitur yang kuat dan fleksibilitasnya, STM32 menjadi pilihan yang populer dalam industri embedded dan banyak digunakan oleh pengembang dalam menciptakan solusi dan sistem elektronik yang andal dan efisien [8], [9].

2.2. Push Button

Gambar 2. Mikrokontroller STM32

Push button, atau tombol tekan, adalah saklar elektromekanis sederhana yang digunakan untuk mengontrol aliran arus listrik. *Push button* memiliki dua kontak, satu kontak yang terhubung secara permanen ke sumber arus, dan satu kontak yang hanya terhubung.

Push button berfungsi sebagai penghubung atau pemutus dalam dua kondisi, yaitu terbuka (*On*), tertutup (*Off*), yang direpresentasikan oleh 1 dan 0. Kedua kondisi ini sangat penting dalam mengoperasikan perangkat listrik yang membutuhkan sumber energi. *Push button* menjadi sangat penting dalam industri karena langsung terhubung dengan operator dan digunakan untuk memulai dan menghentikan mesin. Sebuah mesin, seberapa canggih pun, masih membutuhkan saklar seperti *push button* untuk mengontrol kondisi *On* dan *Off* yang menjadi dasar dari sistem kerjanya[10], [11].

Push button menjadi perangkat kunci yang umum digunakan dalam industri untuk memulai dan menghentikan operasi mesin karena langsung terhubung dengan operator dan sistem kerja yang mudah dioperasikan. Dalam konteks apapun, keberadaan push button atau saklar serupa yang mengontrol kondisi On dan Off adalah elemen esensial yang tidak dapat diabaikan dalam pengoperasian mesin, tidak peduli seberapa canggih mesin tersebut [12].

2.3. Arduino IDE

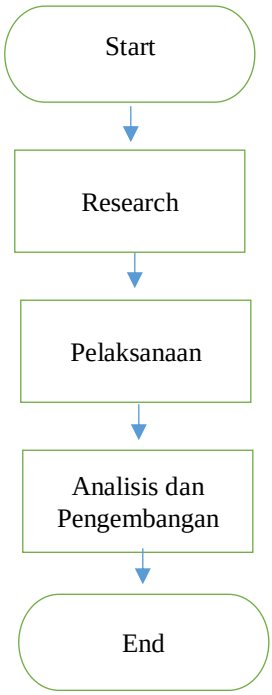
Arduino IDE merupakan perangkat lunak open-source yang digunakan untuk menulis, mengkompilasi, dan mengunggah kode program ke papan mikrokontroler. IDE ini didasarkan pada bahasa pemrograman Processing dan C/C++, dan menawarkan antarmuka yang mudah digunakan bagi pemula maupun programmer berpengalaman. Arduino IDE telah menjadi platform populer untuk prototyping elektronik, pendidikan, dan proyek seni interaktif [13],

Arduino IDE memiliki beberapa fitur utama, antara lain:

a. Editor teks: Digunakan untuk menulis kode program Arduino.
b. Compiler: Mengubah kode program Arduino menjadi kode mesin yang dapat dimengerti oleh mikrokontroler.
c. Uploader: Mengunggah kode program Arduino ke papan mikrokontroler.
d. Monitor serial: Digunakan untuk memantau komunikasi serial antara komputer dan papan mikrokontroler.
e. Library manager: Memungkinkan pengguna untuk menginstal dan menggunakan library pihak ketiga.
f. Debugger: Membantu pengguna dalam menemukan dan memperbaiki bug dalam kode program[2].

3. METODE PENELITIAN

Penelitian ini menggunakan pendekatan eksperimen untuk mengembangkan dan menguji prototipe Smart Charger berbasis IoT. Prototipe ini dirancang untuk mengontrol, memonitor, dan menampilkan arus, tegangan, serta daya keluaran pada layar TFT. Untuk mengaktifkan dan menonaktifkan port charger, digunakan tombol push button. Mikrokontroler STM32 dipilih sebagai komponen utama untuk mengatur dan membaca arus menggunakan software STM32CubeIDE.



Gambar 3. Flowchart Penelitian

a. Research : Melakukan identifikasi kebutuhan, dan studi literatur untuk menentukan kebutuhan fungsional dan non-fungsional dari sistem Smart Charger dan Mengevaluasi teknologi yang relevan dan praktik terbaik dalam pengembangan charger pintar.
b. Pelaksanaan : Melakukan pengamatan langsung di tempat kerja selama periode waktu yang ditentukan dan Merancang arsitektur perangkat.
c. Analisis : Melakukan analisis terkait prototipe smart charger dan memastikan bahwa sistem bekerja sesuai dengan spesifikasi yang telah ditentukan
d. Pengembangan : Meningkatkan performa dan fitur berdasarkan hasil analisis

Langkah-langkah dalam melakukan penelitian ini:

a. Perancangan Prototipe:
   - Desain dan implementasi hardware Smart Charger termasuk integrasi layar TFT dan push button.
   - Konfigurasi pin STM32 untuk berbagai fungsi, seperti push button enter (pin B13),

push button down (pin B14), dan push button up (pin B15).

b. Pengembangan Perangkat Lunak:
   - Pemrograman STM32 menggunakan STM32CubeIDE untuk membaca sensor arus, tegangan, dan daya.
   - Pemanfaatan perangkat lunak Terminal untuk memantau keluaran dari STM32 serta menguji fungsi *port charger* dengan menggunakan *USB Tester*.

4. HASIL DAN PEMBAHASAN

4.1. Penggunaan STM32 untuk Push Button pada Prototipe Smart Charger

Prototipe *Smart Charger* ini merupakan sebuah *charging stasion* berbasis *IOT*, yang mampu mengontrol, memonitoring arus, tegangan dan daya yang keluar dan memilih *port* yang digunakan. *Smart charger* ini juga dapat mengotrol keluaran melalui layar *TFT* yang dikendalikan oleh tombol *push button* untuk memilih *port* yang diaktifkan. Terdapat komponen mikrokontroler STM32 pada *Smart Charger* untuk mengatur dan membaca arus, tegangan, dan daya. Untuk melakukannya bisa menggunakan *software STM32CubeIDE*. Dan untuk mengecek *ouput* keluaran setiap *port* bisa memakai *software* Terminal atau bisa juga memakai *USB Tester*.

Tabel 1. Pin STM32

pin	STM32 BluePill	funtion	pin	STM32 BluePill	funtion
	A0	LCD-D0		B0	
	A1	LCD-D1		B1	
	A2	LCD-D2		B2	
	A3	LCD-D3		B3	LCD_Rd
	A4	LCD-D4		B4	LCD_Wr
	A5	LCD-D5		B5	LCD_Rs
	A6	LCD-D6		B6	LCD_Cs
	A7	LCD-D7		B7	LCD_Rst
	A8			B8	
	A9	Transmit to ESP32 (Tx1)		B9	Receive From STM32-1
	A10	Receive From ESP32 (Rx1)		B10	
	A11			B11	Receive From STM32-2 (Rx3)
	A12			B12	
	A13			B13	Push Button enter (S2)
	A15			B14	Push Button down (S3)
	C13	led status		B15	Push Button up (S4)

Gambar diatas menjelaskan mengenai prototipe smart charger dan pin STM32 yang digunakan pada Smart Charger. Pengunaan STM32 disini untuk mengaktifkan port USB yang terdapat pada layar TFT. Terlihat pada gambar digunakannya pin B13 sebagai *Push button enter*, pin B14 sebagai *Push button down*, pin B15 sebagai *Push button up*, dan lain sebagainya.

Pada prototipe ini, terdapat layar TFT digunakan untuk memonitoring arus, daya, dan tegangan, yang memungkinkan *user* untuk melakukan pemantauan secara real-time dan mendapatkan data yang akurat mengenai performa kelistrikan yang sedang diuji.

```
void halaman_satu() {
  tft.setFont(&FreeSansBold9pt7b);
  tft.setTextSize(1);
  tft.setCursor(10, 55);
  tft.setTextColor(GREEN);
  tft.println("Charger 1");

    tft.setCursor(25, 78);
    tft.println("V = ");
    tft.setCursor(29, 95);
    tft.println("I = ");
    tft.setCursor(25, 112);
    tft.println("W = ");
```

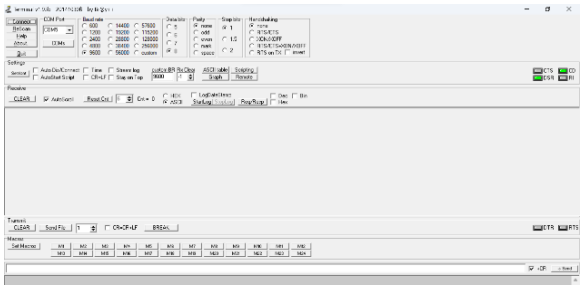
Gambar 4. Sintaks TFT

Sintaks di atas digunakan untuk menampilkan data yang diambil dari sensor arus, tegangan, dan daya ke layar TFT. Dalam hal ini, sensor arus akan memberikan informasi mengenai arus listrik yang mengalir melalui rangkaian, sedangkan sensor tegangan akan memberikan informasi tentang tegangan listrik yang ada pada titik tertentu dalam rangkaian tersebut. Data daya, yang merupakan hasil dari perkalian arus dan tegangan, juga akan ditampilkan di layar TFT. Dengan menggunakan sintaks ini, kita dapat memantau dan menganalisis kinerja dari sistem listrik secara real-time melalui tampilan visual yang ditunjukkan oleh layar TFT tersebut.

4.2. Pembacaan Data Tegangan, Arus, dan Daya Dari STM32

Pada bagian ini, Pembacaan Sensor tegangan, arus dan daya menggunakan Format Berikut: ($c1,0.00,0.00,0.00,c2,0.00,0.00,0.00,c3,0.00,0.00,0.00,c4,0.00,0.00,0.00,c5,0.00,0.00,0.00,c6,0.00,0.00,0.00,000000).

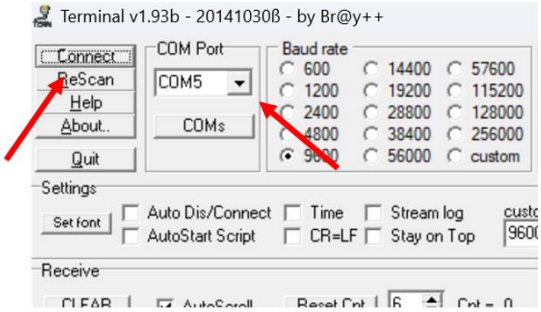
Pembacaan Sensor tegangan, arus dan daya disini penulis menggunakan *software* Terminal. Terminal merupakan program simulasi terminal *port serial* (COM) sederhana. Ini dapat digunakan untuk komunikasi dengan perangkat yang berbeda seperti modem, *router*, sistem *IC* tertanam, telepon *GSM*, modul *GPS*. Ini adalah alat debugging yang sangat berguna untuk aplikasi komunikasi serial.



Gambar 5. Tampilan awal *software* terminal

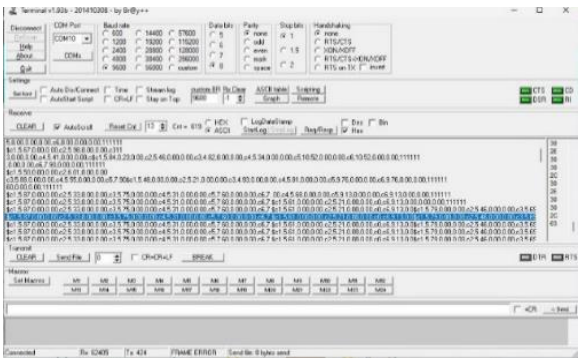
Langkah-langkah yang dilakukan untuk membaca sensor tegangan, arus dan daya Sebagai Berikut:

a. Buka *software* Terminal
b. Sambungkan laptop pada *IC* STM32 menggunakan *USB UART*.
c. Pada software Terminal *Connect* pada port usb yang digunakan.



Gambar 6. Menghubungkan Terminal Pada Port USB

d. Setelah Berhasil Tersambung maka lakukan atau klik "*Set Macro*" untuk menyalakan *port Charger* 1 hingga Charger 6 dengan kode "C11" untuk kode pertama C = *charger*, untuk kode kedua 1 = Mengatifkan *charger*, dan terkahir kode ketiga 1 = untuk memilih charger keberapa yang diaktifkan. Dan untuk mematikannya dengan kode "C01" untuk kode pertama C = *charger*, untuk kode kedua 0 = Menonaktifkan *charger*, dan terkahir kode ketiga 1 = untuk memilih *charger* keberapa yang dinonaktifkan.
e. Setelah dinyalakan maka nilai *Output* akan ditampilkan di *software*

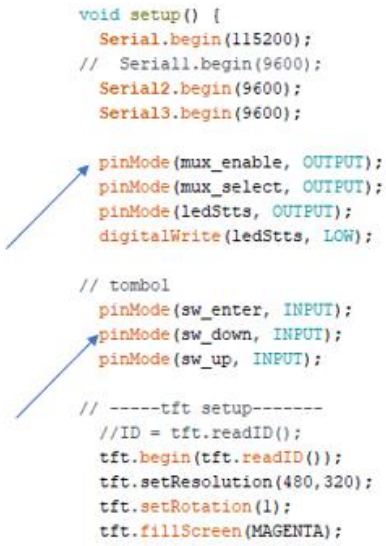


Gambar 7. Hasil Output Pembacaan Sensor

f. Lalu catat ouput yang dikeluarkan pada Terminal. didapatkan data sebagai berikut:
($c1,5.87,0.00,0.00,c2,5.23,0.92,0.00,c3,5.35,0.00,0.00,c4,7.73,0.00,0.00,c5,5.18,0.47,5.51,c5.18,0.00,0.00,111111).
Pada pembacaan sensor tegangan, arus dan daya, bisa kita lihat ada beberapa tegangan yang belum stabil dan ada beberapa tegangan yang dinyalakan akan mengalami *drop*. Kode "111111" paling belakang pada tahap 7, Bahwa kode itu pada pembacaan sensor merupakan tanda bahwa charger sedang *aktif* sedangkan bila *nonaktif* akan keluar tanda "000000".
Dengan demikian, setelah pembacaan data dari sensor tegangan, arus, dan daya dilakukan, dapat disimpulkan bahwa ada beberapa charger yang keluarannya belum stabil. Hal ini terlihat dari fluktuasi nilai tegangan dan arus yang terdeteksi oleh sensor, yang mengindikasikan bahwa output daya dari charger tersebut tidak konsisten. Ketidakstabilan ini dapat menyebabkan berbagai masalah, seperti kerusakan pada perangkat yang diisi dayanya atau efisiensi pengisian daya yang rendah. Oleh karena itu, penting untuk melakukan pengujian lebih lanjut dan mungkin melakukan perbaikan atau penggantian pada charger yang bermasalah untuk memastikan kinerja yang optimal dan aman.

4.3. Push button untuk mengaktifkan *Port Charger*

Selain itu, terdapat juga komponen *push button* yang digunakan untuk mengatur *on/off* dari *port USB* pada prototipe *Smart Charger*. Push button ini memungkinkan pengguna untuk mengendalikan aliran listrik ke *port USB* secara manual, yang berguna untuk menghemat energi dan meningkatkan keamanan perangkat. Dengan menekan *push button*, pengguna dapat dengan mudah menyalakan atau mematikan port USB, sehingga mencegah pengisian daya yang tidak diinginkan atau penggunaan yang berlebihan.



```
void setup() {
  Serial.begin(115200);
//  Serial1.begin(9600);
  Serial2.begin(9600);
  Serial3.begin(9600);

  pinMode(mux_enable, OUTPUT);
  pinMode(mux_select, OUTPUT);
  pinMode(ledStts, OUTPUT);
  digitalWrite(ledStts, LOW);

// tombol
  pinMode(sw_enter, INPUT);
  pinMode(sw_down, INPUT);
  pinMode(sw_up, INPUT);

// -----tft setup-------
  //ID = tft.readID();
  tft.begin(tft.readID());
  tft.setResolution(480,320);
  tft.setRotation(1);
  tft.fillScreen(MAGENTA);
```

Gambar 8. Sintaks untuk push button

Dapat dilihat dari sintaks diatas, pada bagian "*pinmode*" merupakan sintaks untuk *push button* untuk perintah *up*, *down*, dan *enter* pada menu dilayar monitor TFT. *Push button* ini mengatur aktif/mati dari *port* di *smart chager* dan kemudian di tampilkan di layar TFT. Pada bagian "*pinMode*(*mux_enable*, *OUTPUT*);" berguna untuk mengirim data dari sensor tegangan, arus, dan daya ke STM32 dan ditampilkan datanya di layar TFT.

Gambar 9. Tampilan sebelum di input program

Gambar 10. Tampilan setelah di input program

Bisa dilihat dari gambar 4sebelum dimasukan sintaks dari sensor, *LCD* hanya menampilkan tulisan V,I,W tanpa ada data yang keluar, maka sebab itu perlu program dari data sensor arus, tegangan, dan, arus. Dan hasil dari codingan tersebut dapat dilihat dalam gambar 4. Dapat dilihat dari gambar layar di *TFT* telah menampilkan data dari sensor dan dapat dlihat di bagian *charger* 2, *charger* 3, dan *charger* 5 berwarna merah itu menunjukkan *port* yang aktif pada prototipe smart chager.

Pada prototipe ini pula terdapat komponen ESP32 yang berfungsi untuk menghubungkan alat tersebut ke server ThingsBoard. Server ThingsBoard yang digunakan memanfaatkan protokol MQTT, yang dipilih sebagai protokol utama untuk pengiriman data dari sensor ke server ThingsBoard. ThingsBoard sendiri adalah platform IoT yang menyediakan infrastruktur di sisi server, mencakup database, broker MQTT, dan web server yang dilengkapi dengan dashboard.

Push button ini berfungsi untuk mengatur *port charger* melalui menu yang ditampilkan di layar TFT. Kemudian, data dari ESP32 digunakan untuk mengonfirmasi bahwa pengaktifan *charger* telah berhasil dilakukan. Selanjutnya, data dari ESP32 juga digunakan untuk mengirim informasi ke *database* MQTT.

5. KESIMPULAN DAN SARAN

Pembacaan sensor arus dan tegangan pada mikrokontroler STM32 menunjukkan beberapa hasil penting. Meskipun menggunakan software Terminal untuk membaca data sensor memberikan informasi yang cukup akurat mengenai arus, tegangan, dan daya yang dihasilkan oleh charger, terdapat beberapa ketidakstabilan dalam pembacaan tersebut. Fluktuasi nilai tegangan dan arus yang terdeteksi menunjukkan bahwa output daya dari beberapa charger tidak konsisten, yang dapat menyebabkan masalah seperti kerusakan perangkat yang diisi daya atau efisiensi pengisian yang rendah. Oleh karena itu, perlu dilakukan pengujian lebih lanjut dan kemungkinan perbaikan atau penggantian pada charger yang bermasalah untuk memastikan kinerja optimal dan aman dari sistem Smart Charger berbasis IoT ini.

Penggunaan push button pada prototipe Smart Charger memungkinkan pengguna untuk mengatur aliran listrik ke port USB secara manual, yang berguna untuk menghemat energi dan meningkatkan keamanan perangkat. Push button ini berfungsi untuk mengaktifkan atau menonaktifkan port USB, serta memonitor arus, tegangan, dan daya melalui layar TFT secara real-time. Hal ini memberikan kontrol lebih terhadap proses pengisian daya dan membantu dalam pemantauan performa kelistrikan yang diuji.

Secara keseluruhan, meskipun ada beberapa tantangan dalam pembacaan data sensor yang perlu diatasi, prototipe Smart Charger yang menggunakan mikrokontroler STM32 menunjukkan potensi yang baik dalam mengontrol dan memantau proses pengisian daya dengan efisien .

DAFTAR PUSTAKA

[1] M. Somantri, R. Pramudita, and M. A. Rizqulloh, “Workshop on STM32 Board Utilization to Elevate the Expertise of Electrical Vocational High School (SMK) Teachers in the Greater Bandung Region,” vol. 5, no. 1, pp. 21–30, 2024.

[2] K. Kartika, A. Asran, H. Erawati, E. Ezwarsyah, and ..., “Pelatihan Platform Arduino Bagi Siswa SMA Negeri 1 Baktiya Alue Ie Puteh Aceh Utara,” *J. Solusi …*, pp. 1–5, 2022, [Online]. Available: http://jsmd.dikara.org/jsmd/article/view/13%0A http://jsmd.dikara.org/jsmd/article/download/13/24

[3] J. Akbar, A. Suyitno, and J. F. N. Dethan, “Design Training Kit STM32F1 ARM Cortex on Microprocessor and Microcontroller System,” *JTEIN J. Tek. Elektro Indones.*, vol. 5, no. 1, pp. 127–137, 2024, doi: 10.24036/jtein.v5i1.585.

[4] Z. Oktarina and H. Habibullah, “Rancang Bangun Coffee Mix Berbasis Mikrokontroller,” *JTEIN J. Tek. Elektro Indones.*, vol. 4, no. 2, pp. 715–723, 2023, doi: 10.24036/jtein.v4i2.489.

[5] M. Sodikin, R. Corputty, R. Rusli, and A. Azizah, “Implementasi Perancangan Robot Line Follower Berbasis Mikrokontroler Atmega

32A," *Musamus J. Electro Mech. Eng.*, vol. 1, no. 02, pp. 53–60, 2019, doi: 10.35724/mjeme.v1i02.1491.

[6] G.-R. Miguel Angel and M. Pedro Cesar Santana, *DIY Microcontroller Projects for Hobbyists: The ultimate project-based guide to building real-world embedded applications in C and C++ programming*. Packt Publishing Ltd, 2021. [Online]. Available: http://ieeexplore.ieee.org/document/10163540

[7] K. Vinoth Kumar, P. Radhakrishnan, R. Kalaivani, V. Devadoss, L. D. Vijay Anand, and K. Vinodha, "Implementation of Smart Electric Vehicle Charging Station Driven Using Experimental Investigation," in *2021 2nd Global Conference for Advancement in Technology, GCAT 2021*, IEEE, 2021, pp. 1–5. doi: 10.1109/GCAT52182.2021.9587788.

[8] H. Ashari and M. Zakarijah, "Stm32 Arm Cortex-M Sebagai Media Pembelajaran Mikrokontroler," *J. Elektron. Pendidik. Tek. Elektron.*, vol. 8, no. 1, pp. 10–18, 2019.

[9] R. J. Franklin and Mohana, "Industry 4.0: Real Time Embedded System with Integrated Data Acquisition," in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2020, pp. 637–642. doi: 10.1109/ICSSIT48917.2020.9214251.

[10] A. Munandar, N. David, M. Veronika, D. Abdulllah, and E. Sahputra, "Miniature Design of Liquid Filling Machine Automatically Using ESP32 Based IOT (Internet of Things) Perancangan Miniatur Mesin Pengisi Cairan Otomatis Menggunakan ESP32 Berbasis IOT (Internet of Things)," vol. 3, no. 1, pp. 69–78, 2023.

[11] J. Li and et al., "A Flexible Piezoelectric Nanogenerator-Based Self-Powered Push Button for Personal Electronics," *Nano Energy*, vol. 63, 2019.

[12] B. R. Sinaga, "Rancang Bangun Gerbang dengan Menggunakan Kontrol Android Via Bluetooth Berbasis Arduino Uno R3," *J. Pendidik. Sains dan Komput.*, vol. 2, no. 02, pp. 312–316, 2022, doi: 10.47709/jpsk.v2i02.1737.

[13] N. Ningsih *et al.*, "Smart Agriculture untuk Mewujudkan Ketahanan Pangan Berbasis Lora di Desa Kalipadang-Benjeng Gresik," *J. Pengabdi. Nas. Indones.*, vol. 5, no. 1, pp. 221–233, 2024.