Jurnal Fokus Elektroda : Energi Listrik, Telekomunikasi, Komputer, Elektronika dan Kendali)
Volume 8 No 1, Tahun 2023: Hal. 13-20.
e-ISSN: 2502-5562. Open Access at: https://elektroda.uho.ac.id/
Penerbit : Jurusan Teknik Elektro Universitas Halu Oleo Kendari Sulawesi Tenggara,

Perancangan Pengendali Gerakan *Stepper Motor* menggunakan Mikrokontroler STM32 dengan Tampilan Grafis TouchGFX

Mada Jimmy Fonda Arifianto[1] , *Heru Suprapto*[2]

[1,2]*Program Studi Mekatronika, Politeknik Astra*

mada.jimmy@polytechnic.astra.ac.id, heru.suprapto@polytechnic.astra.ac.id

Correspondent Author: Mada Jimmy Fonda Arifianto

***Abstract* — Stepper Motor is an actuator, which is often used in electromechanical systems of automation equipment or machines that require precision movement. Physical quantities in motion control are position, velocity and acceleration. To facilitate the control and monitoring of these quantities, a graphical display interface with a touch screen is proposed. In this paper, a motion controller design that integrates the STM32H743 microcontroller, TMC5160 and a 7 inch LCD touch screen will be presented. The supporting software for this system is TouchGFX as a Graphical User Interface on a touch screen. In addition, FreeRTOS is used as a microcontroller operating system to support the concept of multi-tasking in motion control applications. Hopefully, this controller design can be run without the need additional controller devices or computers. The experiment results show that the motor can be driven simultaneously on a task, while the microcontroller can perform display control with 30fps, touch screen and data communication which are handled by the scheduler in real time. The application requires 16.26% RAM and 24.14% flash of total capacity.**

***Keyword* — Stepper motor, motion control, microcontroller, GUI, FreeRTOS, STM32.**

Abstrak — *Stepper Motor* merupakan aktuator yang sering digunakan pada sistem elektromekanik peralatan otomasi atau mesin yang membutuhkan gerakan presisi. Besaran fisika dalam pengendalian gerakan adalah posisi, kecepatan dan percepatan. Untuk memudahkan pengendalian dan pemantauan besaran tersebut, diusulkanlah antarmuka tampilan grafis dengan layar sentuh. Pada tulisan ini akan dikemukakan rancangan pengendali gerakan yang mengintegrasikan mikrokontroler STM32H743, TMC5160 dan LCD 7 inch layar sentuh. Perangkat lunak pendukung sistem ini yaitu TouchGFX sebagai platform *Graphical User Interface* pada layar sentuh. Selain itu, FreeRTOS juga digunakan sebagai sistem operasi mikrokontroler untuk menunjang konsep multi-tugas (*multitasking*) pada aplikasi kendali gerakan. Diharapkan, rancangan pengendali ini dapat dijalankan tanpa memerlukan perangkat kontroler atau komputer tambahan. Hasil uji coba menunjukkan bahwa motor dapat digerakkan secara simultan pada suatu tugas (*task*), sementara mikrokontroler dapat melakukan tugas kendali tampilan dengan laju bingkai 30fps, layar sentuh dan komunikasi data yang ditangani oleh *scheduler* secara *realtime*. Aplikasi membutuhkan RAM sebesar 16.26% dan Flash sebesar 24.14% dari kapasitas total.

Kata kunci — Stepper Motor, kendali gerakan, mikrokontroler, GUI, FreeRTOS, STM32.

I. PENDAHULUAN

Pengendali gerakan *(motion control)* pada peralatan atau mesin presisi memegang peranan penting dalam menghasilkan produk yang berkualitas. *Stepper motor* adalah salah satu motor elektrik yang sering digunakan untuk keperluan ini. Contoh mesin yang menggunakan *stepper motor* untuk menghasilkan gerakan yaitu pencetak tiga dimensi, pemotong laser, mesin CNC, robot lengan, dan sebagainya. Pengendali gerakan yang terdapat pada mesin-mesin tersebut biasanya membutuhkan kontroler atau komputer terpisah, beserta perangkat lunak sesuai aplikasinya. Penelitian bertema kendali mesin *engraving* berbasis mikrokontroler juga telah dilakukan untuk mengurangi biaya[1]. Mesin CNC milling mini juga dikembangkan menggunakan mikrokontroler[2].

Saat ini teknologi mikrokontroler berkembang semakin pesat dengan meningkatnya frekuensi *clock* pada prosesornya, bertambah besarnya memori serta adanya fitur-fitur yang semakin kaya. Sementara itu teknologi tampilan (*display technology*) juga semakin banyak pilihan yang dapat digunakan untuk aplikasi sistem tertanam, contohnya *Liquid Crystal Display – Thin Film Transistor* (LCD-TFT) dengan layar sentuh yang tersedia dalam bermacam ukuran. Dengan latar belakang ini, maka penulis memberikan usulan untuk mengembangkan alternatif pengendali gerakan yang terintegrasi pada satu peralatan yang dilengkapi dengan layar tampilan sebagai antarmuka dengan pengguna. Diharapkan, dengan adanya konsep baru ini, aplikasi yang berkaitan dengan sistem pergerakan mekanik dapat dilakukan dengan praktis dan cepat namun tetap akurat.

Tujuan penelitian yang hendak dicapai yaitu membuat alat pengendalian gerakan berbasis mikrokontroler dan LCD layar sentuh. Dari tujuan tersebut, maka rumusan masalah yang dinyatakan yaitu bagaimana membuat sebuah purwarupa aplikasi pengendali *stepper motor* berbasis mikrokontroler dengan antarmuka LCD layar sentuh dengan konsep *multi-tasking*. Tampilan *Graphics User Interface* (GUI) diharapkan memiliki konsep desain modern yang menarik seperti perangkat gawai pada umumnya (misalnya tampilan pada ponsel certas dan komputer tablet)

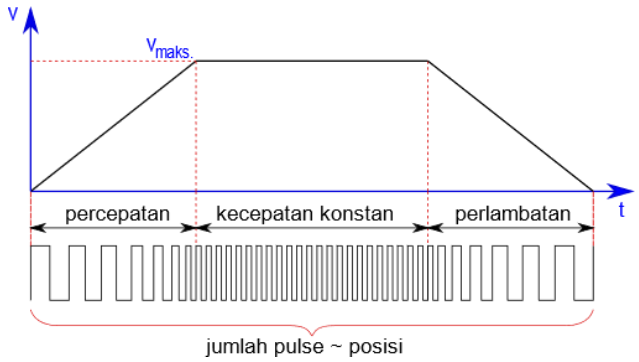
Metodologi penelitian yang digunakan yaitu ekperimen. Penelitian dimulai dengan studi pustaka mengenai pengembangan GUI, *multi-tasking* dan pengenalian *stepper motor*. Setelah itu, dilakukanlah perancangan alat dan penentuan kebutuhan alat. Berikutnya adalah pemprograman mikrokontroler dan perancangan tampilan GUI pada layar. Uji coba dilakukan tahap demi tahap sampai diperoleh hasil

seperti pada tujuan penelitian. Selanjutnya hasil penelitian didokumentasikan sebagai referensi penelitian selanjutnya.

II. TINJAUAN PUSTAKA

A. Pengendalian Gerakan

Stepper motor merupakan motor dengan pergerakan secara diskret (langkah per langkah) dengan sudut putar tertentu. Arus yang mengalir ke koil pada motor menyebabkan terjadinya medan magnet yang akan menggerakkan magnet pada rotor. Dengan mengatur urutan koil yang aktif, maka rotor akan mulai berputar[3]. Contohnya, *bipolar stepper motor* dengan sudut langkah (*step angle*) sebesar 1.8° membutuhkan 200 langkah untuk berputar satu putaran penuh. Pengaturan motor dilakukan dengan memberikan sinyal dari mikrokontroler ke penggerak (penguat arus) yang berfungsi menguatkan arus menuju ke motor. Ada dua macam sinyal yang harus diberikan ke penggerak yaitu sinyal yang digunakan untuk menambah langkah (*step* atau *pulse*) dan sinyal untuk menentukan arah (*direction*). Supaya motor dapat berputar, maka perlu diberikan *pulse* dengan frekuensi tertentu.



Gambar 1 Profil kecepatan pada Stepper Motor

Ilustrasi profil kecepatan (*velocity profile*) pada *stepper motor* dapat dilihat di Gambar 1. Jumlah pulsa yang diberikan ke penggerak akan menentukan langkah sudut gerakan poros motor. Sedangkan frekuensi pulsa pada sinyal ini akan menentukan kecepatan sudut motor. Perubahan frekuensi dari 0 Hz menuju frekuensi tertentu akan menentukan percepatan sudut ketika motor mulai bergerak. Demikian juga sebaliknya, perubahan frekuensi menuju 0 Hz akan menentukan perlambatan sudut ketika motor sedang berhenti atau sedang mencapai posisi yang diinginkan. Pengaturan percepatan dan perlambatan ini bertujuan agar gerakan motor menjadi halus atau tidak menghentak.

Sebuah penggerak (*stepper motor driver*) yang memiliki kemampuan *microstepping* dapat mengatur sudut motor kurang dari *step angle*. Dengan demikian putaran motor dapat lebih halus. *Microstep* dilakukan dengan cara mengatur modulasi arus yang mengalir pada koil-koil sedemikian rupa sehingga motor dapat bergerak di antara *full step angle*. Sudut *microstep* tergantung dari resolusi *Digital-to-Analog converter* pada penguatan arus, biasanya sebesar 1/2, /1/4, 1/8, … 1/64 dari *full step angle*. Contoh: dengan *full step angle* sebesar 1.8° dan faktor *microstep* sebesar ¼ maka langkah motor dapat mencapai sudut 0,45°.

Besaran penting pada *stepper motor* yang juga perlu diperhatikan adalah torsi motor. Untuk menahan posisi rotor pada posisi tertentu maka diperlukan arus pada koil yang dapat menghasilkan *holding torque*. Besarnya torsi ini dipengaruhi oleh batasan arus yang dapat diatur *driver*. Torsi juga dipengaruhi oleh pengaturan faktor *microstep*, di mana pengaturan *full step* akan memberikan torsi yang besar[4].

B. Sistem Tertanam

Teknologi sistem tertanam (*embedded system*) saat ini berkembang cukup pesat baik dari sisi perangkat keras maupun perangkat lunak. Contohnya adalah mikrokontroler STM32H743 yang sudah ditanamkan prosesor Arm® Cortex®-M7 core dengan kecepatan *clock* mencapai 480MHz. Mikrokontroler ini juga terdapat periferal yang cukup lengkap yaitu antarmuka komunikasi data (I2C, SPI, UART, Ethernet, CAN), memori RAM dan FLASH yang ukurannya cukup besar serta jumlah I/O *port* yang banyak.

Fitur yang cukup penting lainnya adalah mikrokontroler ini memiliki *LCD controller (LTDC)* tersendiri, sehingga dapat diintegrasikan dengan sebuah LCD layar sentuh untuk memberikan interaksi kepada pengguna. Tampilan layar sentuh dapat dikembangkan secara menarik dan memukau seperti tampilan pada ponsel cerdas, karena didukung oleh TouchGFX *framework* yang disediakan secara gratis. *Source code* untuk RTOS (*Real Time Operating System*) juga disediakan untuk mendukung mikrokontroler agar memiliki kemampuan *multitasking*. Peranan RTOS pada sistem tertanam merupakan hal yang lumrah karena banyak digunakan pada sistem kendali industri [5][6]. Dengan kemampuan fitur tersebut maka aplikasi terkait kendali gerakan (*motion control*) dapat dikembangkan dengan performa cukup tinggi atau dengan kata lain dapat menggantikan fungsi komputer. Bahkan sistem tertanam berbasis mikrokontroler ini memiliki keunggulan dibanding dengan komputer (PC) yaitu waktu booting yang sangat cepat (hampir seketika).

C. Antarmuka Grafis

Perkembangan teknologi tampilan grafis *(graphics display*) di bidang sistem tertanam semakin meningkat dan semakin banyak pilihan yang dapat digunakan. Tampilan yang sering digunakan saat ini adalah *Liquid Crystal Display* (LCD). Pada teknologi terdahulu, LCD hanya bisa menampilkan informasi dalam satu warna saja

(*monochrome*). Dengan adanya teknologi LCD-TFT saat ini layar bisa menampilkan banyak warna dengan warna dasar merah, hijau dan biru (*red*, *green*, *blue* atau RGB). Ukuran LCD juga semakin bervariasi, baik dari dimensi (ukuran diagonal yang biasanya diukur dalam satuan inchi) maupun ukuran jumlah *pixel*.

Warna pada LCD dihasilkan dengan mengatur intensitas tiap kanal warna RGB pada sebuah pixel. Dengan demikian, maka akan diperoleh kombinasi warna yang banyak. Jika tiap kanal memiliki resolusi 8 bit (2^8=256 kombinasi intensitas warna), maka pada tiga kanal warna akan memiliki 24 bit atau setara dengan 2^{24} = 16.777.216 warna. Format seperti ini disebut juga RGB888. Pengendalian LCD dengan format RGB888 membutuhkan kontroler yang setidaknya memiliki 24 terminal output digital untuk dikoneksikan ke kanal warna LCD. Jika jumlah output digital pada kontroler tidak mencukupi, maka bisa digunakan format RGB565 dengan jumlah bit pada masing-masing warna yaitu 5bit merah, 6 bit hijau dan 5 bit biru, dengan kata lain ada 16 bit data yang digunakan atau setara dengan 2^{16} = 65.536 warna. Pada kebanyakan bahasa pemprograman, 16 bit bisa diwakili dengan variabel dengan tipe data *integer* atau *word* (2 Byte).

Teknologi panel LCD terus berkembang dengan diciptakannya layar sentuh yang terintegrasi dengan layar LCD. Teknologi ini memungkinkan adanya interaksi secara langsung antara manusia dengan peralatan atau disebut juga *Human Machine Interface* (HMI). Ada dua tipe layar sentuh yang umumnya digunakan yaitu layar sentuh bertipe resistif (*resistive touch screen*) dan bertipe kapasitif (*capacitive touch screen*). Pada layar sentuh resistif, terdapat dua lembaran transparan tipis yang memiliki nilai resistansi, masing-masing searah horisontal dan searah vertikal (aksis x,y). Nilai resistansi akan berubah sesuai posisi jari saat menekan layar. Dengan menghubungkan terminal layar sentuh ke pembagi tegangan, maka akan diperoleh tegangan analog yang selanjutnya diperhitungkan untuk mengetahui posisi penekanan pada layar. Berbeda dengan tipe resistif yang harus memberikan sedikit tekanan ke layar, pada layar sentuh tipe kapasitif, jari pengguna tidak perlu menekan tapi hanya menyentuh permukaan kacanya saja. Tipe ini memiliki lapisan elektroda di antara lapisan kaca dan lapisan layar sentuh untuk mengindera konduktifitas yang biasa dihasilkan oleh jari. Tipe kapasitif memungkinkan untuk menerima lebih dari satu sentuhan (*multi-touch*).

Peralatan kontroler industri banyak yang menggunakan layar LCD warna untuk memberikan informasi sekaligus menjadi masukan sinyal ketika layar sentuhnya ditekan. Jenis-jenis obyek yang dapat ditampilkan pada layar adalah teks, gambar, grafik data, tombol, saklar, *slider*, *list box* dan sebagainya.

Teknologi perangkat lunak dalam mendukung perancangan antarmuka grafis juga mengalami perkembangan. TouchGFX *framework* adalah salah satu perangkat lunak yang akan dibahas pada tulisan ini. TouchGFX dikembangkan oleh STMicroelectronics yang secara khusus memiliki kompabilitas dengan mikrokontroler STM32 agar LCD-TFT yang dikendalikannya dapat menghasilkan gambar, tulisan, grafik, animasi yang menarik. Dengan demikian revolusi *HMI (Human Machine Interface) of Things* dapat mudah dicapai[7].

III. Pengumpulan Data dan Perancangan

A. Pengumpulan Data

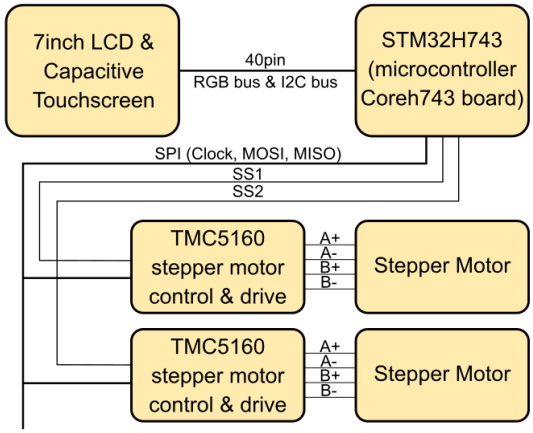
Bagian ini mendefinisikan spesifikasi sistem yang akan dikembangkan berdasarkan kebutuhan-kebutuhan pengguna terkait dengan kendali motor yang sering digunakan di industri. Sebagai acuan, studi sistem kendali gerakan difokuskan pada pengaturan posisi di mesin CNC. Gerakan yang dihasilkan pada prinsipnya adalah kumpulan kordinat titik-titik yang telah diprogram lalu dijalankan sesuai urutan titik tersebut.

Kriteria yang akan digunakan sebagai dasar perancangan sistem yaitu:

1. Pengaturan pengendalian *stepper motor* berbasis mikrokontroler dilakukan melalui LCD layar sentuh
2. Besaran yang diatur adalah posisi (sudut), kecepatan, arah putar stepper motor dan *microstep*
3. Layar LCD memberikan informasi nilai posisi aktual secara realtime.
4. Pengaturan posisi motor dapat dilakukan secara tunggal (satu nilai posisi) dengan memberi masukan pada layar.
5. Pengaturan tujuan posisi dapat dilakukan berdasarkan tabel nilai yang dibuat pengguna melalui masukan dari LCD layar sentuh

B. Perancangan Perangkat Keras

Berdasarkan kebutuhan dan tujuan penelitian, maka sistem kendali gerakan perlu dirancang terlebih dahulu sebelum dilakukan pembuatan alat. Rancangan dalam bentuk diagram blok dituangkan pada gambar berikut.



Gambar 2 Diagram blok sistem kontrol

Komponen dan modul elektronik dipilih berdasarkan teknologi yang relatif baru sehingga fungsi-fungsi yang tersedia dapat dikembangkan lebih lanjut. Masing-masing komponen dapat dijelaskan sebagai berikut:

1. Papan CoreH743I / OpenH743-C dari Waveshare yang berfungsi sebagai pengendali utama [8]. Spesifikasi papan CoreH743 adalah:
   a. Kecepatan *clock* mikrokontroler STM32H743 yaitu 480MHz
   b. Terdapat IC RAM IC42S16400J (64Mbit)
2. LCD layar sentuh tipe kapasitif yang dihubungkan ke mikrokontroler melalui *Flexible Flat Cable* (FFC) 40 pin. Pada kabel ini terdapat jalur pengiriman data RGB (*Red Green Blue*) dan jalur I2C untuk mengambil data layar sentuh. Spesifikasi modul LCD adalah:
   a. Ukuran layar: 7 inch
   b. Ukuran pixel: 1024x600
   c. Layar sentuh tipe kapasitif
   d. Komunikasi RGB 24bit
   e. Komunikasi I2C untuk *touchscreen*
3. Penggerak motor tipe TMC5160 yang dihubungkan ke mikrokontroler menggunakan jalur I2C sebagai sarana komunikasinya. Karena pada papan TMC5160 sudah terpasang rangkaian mosfet sebagai penguat arus maka *stepper motor* dapat langsung dihubungkan. Spesifikasi TMC5160 *breakout board* [9][10] adalah:
   a. Tegangan masukan motor: 9…36V
   b. Arus fasa: RMS 2.8A
   c. *Microsteps*: 1…256
   d. Komunikasi SPI
4. Stepper motor 17HS4401 yang dihubungkan ke TMC5160 melalui 4 kabel. Spesifikasi motor adalah:
   a. Ukuran: NEMA 17
   b. Tegangan: 12v
   c. Arus: 1.5A
   d. Sudut Langkah: 1.8°
5. Catu daya tegangan 5 volt DC untuk mikrokontroler dan tegangan 12volt DC untuk motor (melalui TMC5160)

C. Perancangan Perangkat Lunak

Pemrograman mikrokontroler berperan penting dalam membangun aplikasi dengan antarmuka grafis untuk pengendali *stepper motor*. Sebelum membuat program, maka perlu dirancang terlebih dahulu dalam bentuk diagram alir seperti pada Gambar 3.

Diagram alir menunjukkan urutan fungsi-fungsi yang dilaksanakan oleh program utama (`main.c`). Bagian inisiasi adalah proses mengaktifkan beberapa antarmuka antara lain *General Purpose Input Output* (GPIO), *Universal Synchronous Receiver-Transmitter* (UART), *Flexible Memory Controller* (FMC), *LCD-TFT Display Controller* (LTDC), *Direct Memory Access 2D* (DMA2D), *Serial Peripheral Interface* (SPI), *Inter-integrated Circuit* (I2C), *Timer*, *Cyclic Redundancy Check* (CRC) dan TouchGFX. Fungsi-fungsi inisiasi ini dihasilkan secara otomatis oleh perangkat lunat STM32CubeMX.

Setelah inisiasi, selanjutnya mendefinisikan tugas-tugas (*tasks*) ke dalam *thread* pada sistem operasi FreeRTOS. Tugas tersebut yaitu:

1. TouchGFX_Process

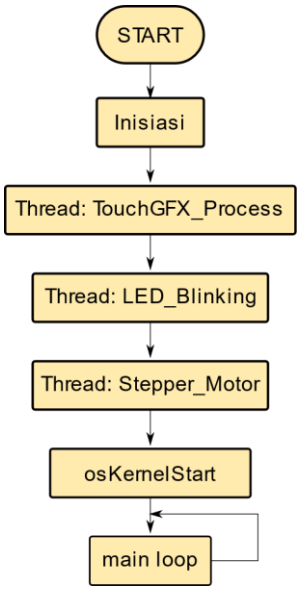
Thread ini berfungsi untuk menjalankan aplikasi GUI dengan menambahkan fungsi yang sudah dihasilkan oleh TouchGFX yaitu `MX_TouchGFX_Process()`.

2. LED_Blinking

Thread ini untuk mengaktifkan salah satu LED yang terdapat pada PCB OpenSTM32 agar menyala berkedip sekaligus sebagai indikator bahwa tugas pada RTOS sudah berjalan secara independen.

3. Stepper Motor Initialization

Tugas yang berkaitan dengan pengaturan motor diatur di *thread* ini. Inisialisasi yang dilakukan pada bagian ini yaitu konfigurasi chopper, PWM, batasan arus, batasan kecepatan dan pengaturan percepatan.



Gambar 3 Diagram alur program utama

Setelah tugas-tugas didefinisikan, maka sistem operasi dijalankan melalui proses `osKernelStart` untuk mengaktifkan *scheduler*. Bagian terakhir yaitu *main loop*, yang tidak mengerjakan apapun selain iterasi tanpa henti namun berguna untuk menjaga program utama tetap berjalan.

IV. PEMBUATAN

A. Persiapan Bahan dan Alat

Modul-modul elektronik seperti yang telah disebutkan pada bagian perancangan dihubungkan seperti pada diagram blok (Gambar 2). Pada pembuatan sistem ini, modul mikrokontroler CoreH743I diletakkan pada papan pengembangan OpenSTM32I pada slot yang sesuai. Di papan ini sudah disediakan juga konektor-konektor yang dipisahkan menurut fitur-fitur mikrokontroler.

Koneksi antara mikrokontroler STM32H743 dengan LCD dan STM32H743 dengan TMC5160 dijelaskan seperti pada Tabel 1. Konfigurasi pin ini akan digunakan lebih lanjut pada saat pembuatan program.

Tabel 1 Koneksi mikrokontroler dengan LCD dan TMC5160

STM32 Port	*LCD Signal*	*STM32 Port*	*LCD Signal*
PG6	R7	PI7	B7
PB1	R6	PI6	B6
PC0	R5	PI5	B5
PH10	R4	PI4	B4
PH9	R3	PG11	B3
PH8	R2	PD6	B2
PH3	R1	PG12	B1
PH2	R0	PE4	B0
PI2	G7	PI10	HSYNC
PI1	G6	PI9	VSYNC
PI0	G5	PG7	CLK
PH15	G4	PF10	DE
PG10	G3	PF6	GT911_RST
PH13	G2	PF7	GT911_SDA
PE6	G1	PF8	GT911_SCL
PE5	G0	PD7	GT911_INT
STM32 Port	*TMC5160*	*STM32 Port*	*TMC5160*
PA4	CSN	PA6	MISO
PA5	SCK	PA7	MOSI

B. Pemprograman

Pembuatan program kendali gerakan berbasis mikrokontroler STM32 dilakukan dengan tahapan berikut:

1. Konfigurasi pin dan fitur pada mikrokontroler.

Mikrokontroler STM32 memiliki banyak fitur yang dapat diaktifkan (menggunakan perangkat lunak STM32CubeMX versi 6.5.0) seperti ditunjukkan pada Gambar 4. Fitur-fitur yang digunakan pada proyek ini sesuai dengan sub program inisiasi pada diagram alir. Hal penting yang perlu diperhatikan pada tahap ini adalah memastikan konfigurasi fungsi masing-masing pin pada mikrokontroler harus sesuai dengan pin pada perangkat yang akan dihubungkan ke mikrokontroler.

2. Konfigurasi frekuensi *clock.*

Frekuensi *clock* pada CPU perlu diatur agar mendapatkan kinerja optimal. Pengaturan *clock* yang tinggi akan meningkatkan kinerja, sementara penggunaan energi menjadi lebih besar. Namun karena catu daya yang digunakan pada proyek ini tidak menggunakan baterai melainkan adaptor AC-DC, maka frekuensi *clock* diatur pada nilai semaksimal mungkin.

Pada tahap ini, pengaturan *clock* disesuaikan dengan perangkat luar yang terhubung sehingga mendapatkan hasil: frekuensi *clock* untuk CPU sebesar 400Mhz, *clock* untuk FMC sebesar 200Mhz, *clock* untuk LDTC sebesar 22,5Mhz, *clock* untuk SPI dan UART sebesar 100Mhz.

Gambar 4 Konfigurasi mikrokontroler di STM32CubeMX

3. Pembuatan format program

Setelah tahap 1 dan 2 dilakukan, maka format program (*template*) dapat dihasilkan oleh STM32CubeMX pada bagian *generate code*. Tahapan ini menghasilkan program bahasa C/C++ yang dilengkapi dengan pustaka (*library*) sesuai tipe mikrokontroler. *Firmware* yang digunakan pada proyek ini yaitu STM32Cube FW_H7 V1.10.0. Selanjutanya program C/C++ ini dapat disunting sesuai kebutuhan sistem yang dikembangakan menggunakan STM32CubeIDE

4. Pembuatan rancangan GUI.

Antarmuka dibuat dengan menggunakan TouchGFX Designer 4.18. Tahap ini adalah penyusunan tayangan (*screen*) yang merupakan tata letak *widget* GUI (*button*, *slider*, *text*, grafik, latar belakang dan sebagainya) yang akan ditampilkan pada layar LCD layar sentuh. *Widget* tersebut diletakkan pada sebuah *Canvas*.

Pada panel sebelah kanan terdapat *properties* yang mengtur ciri-ciri obyek seperti posisi, ukuran, warna, nilai dan sifat tertentu. Pada panel ini juga terdapat *interactions*

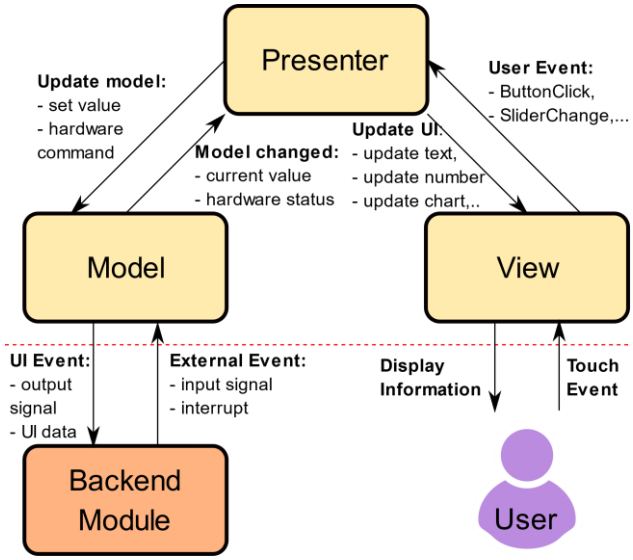
yang mengatur apabila ada *trigger* pada obyek tertentu (misalnya disentuh atau digeser) maka akan memanggil fungsi tertentu.

Sebuah aplikasi TouchGFX dapat disusun dengan beberapa *screen* dengan tata letak yang berbeda. Perubahan dari satu *screen* ke *screen* lainnya dapat dilakukan dengan cepat atau dapat menggunakan transisi berupa animasi tayangan.

Gambar 5 Penyusunan letak *widget* pada TouchGFX Designer

5. Pengembangan program mikrokontroler.

Perangkat lunak yang digunakan untuk mengembangkan program mikrokontroler yaitu STM32CubeIDE. Tahap ini merupakan pembuatan program utama pada `main.c`, mengintegrasikan LCD dan layar sentuh dengan TouchGFX framework. Antarmuka pengguna (*user interface*, UI) pada TouchGFX menghasilkan *template* pemprograman dalam Bahasa C/C++ dengan pola arsitektur MVP (Model-View-Presenter) yang diwujudkan dalam bentuk kelas (*class*).



Gambar 6 Pola *Model-View-Presenter* (MVP) pada TouchGFX

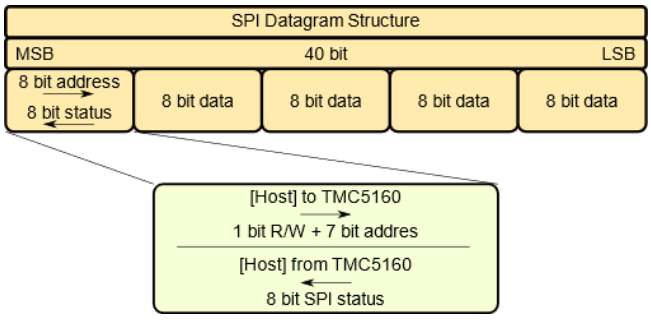
Model merupakan antarmuka yang mendefinisikan data yang hendak ditampilkan dan data yang akan dikirim ke *backend*. Pada aplikasi yang dikembangkan ini, *Model* dapat berinteraksi dengan *class* lain untuk mengaktifkan fungsi-fungsi yang berkaitan dengan modul penggerak TMC5160 untuk mengendalikan motor. Contohnya adalah pengiriman perintah dari model ke TMC5160 untuk menentukan posisi target dan kecepatan target sebuah motor. Pada TMC5160 juga ada fungsi untuk membaca posisi aktual, kecepatan aktual motor dan kondisi apakah motor sudah mencapai posisi atau belum.

View merupakan antarmuka yang menampilkan data ke layar tampilan dan meneruskan peristiwa (*event*) yang dilakukan pengguna terhadap layar sentuh. Contoh tampilan pada layar yaitu posisi aktual dan kecepatan aktual motor yang ditampilkan pada obyek teks. Contoh masukan pengguna melalui layar sentuh yaitu obyek *Button* (tombol) untuk memulai menjalankan motor dan obyek *Slider* untuk menentukan kecepatan dan posisi. Interaksi obyek-obyek tersebut dinyatakan dengan fungsi-fungsi kustom yang terlebih dahulu diatur melalui TouchGFX Designer.

Presenter bekerja berdasarkan *model* dan *view*. *Presenter* mengambil data dari model lalu memformat data untuk dapat ditampilkan ke *view*. Demikian juga sebaliknya, jika ada aktivitas pengguna ketika menekan obyek pada layar, maka *view* akan mengaktifkan fungsi pada *presenter* untuk mengakses variabel atau perintah di *model*.

6. Pembuatan *library* untuk TMC5160 *Breakout board.*

Modul TMC5160 menggunakan SPI (*Serial Peripheral Interface*) untuk berkomunikasi dengan mikrokontroler (sebagai *host*). Supaya transmisi data berhasil maka pin NCS harus dalam keadaan aktif (*low*).



Gambar 7 Struktur datagram SPI pada chip TMC5160

Komunikasi diawali dengan mengirim 8 bit terdiri dari 1 bit untuk menentukan mode apakah *Read* atau *Write*, diikuti dengan 7 bit alamat. Setelah alamat, berikutnya adalah mengirim 4 x 8 bit data. Total yang terkirim adalah 40 bit datagram (dijelaskan pada Gambar 7). Contoh-contoh perintah pada datagram tersebut antara lain untuk konfigurasi, pengaturan posisi dan kecepatan, pembacaan posisi dan kecepatan, mulai menjalankan motor, pengaturan *microstep* dan sebagainya[9].

Pembuatan *library* diperlukan agar program dapat terstruktur dengan baik dan mudah dikembangkan. Fungsi-fungsi yang dibuat sendiri adalah yang sering dipanggil

dalam program yaitu: baca tulis data sepanjang 5 byte dari/ke TMC5160. Caranya yaitu:

a. Memberikan nilai 0 pada pin CSN (*chip select*) menggunakan fungsi HAL_GPIO_WritePin().
b. Baca tulis data SPI menggunakan fungsi HAL_SPI_TrainsmitReceive(). Byte pertama menentukan perintah, nilai 0x21 untuk membaca posisi, 0x22 untuk membaca kecepatan.
c. Memberikan nilai 1 pada pin CSN (*chip select*) menggunakan fungsi HAL_GPIO_WritePin().

Library yang dikembangkan ini selanjutnya akan dimasukkan (*include*) ke *Model* (`model.cpp`) yang *template*-nya dihasilkan oleh TouchGFX.

7. Transfer program ke mikrokontroler

Program yang akan ditransfer ke mikrokontroler perlu dilakukan kompilasi terlebih dahulu. Setelah tidak ada kesalahan, lalu file (*.HEX) dapat dihasilkan, maka berikutnya dilakukan proses *debugging* atau pengetesan untuk memastikan hasil sesuai dengan rancangan program. Transfer program dengan format (*.HEX) ke mikrokontroler bisa dilakukan dengan STM32CubeProgrammer.

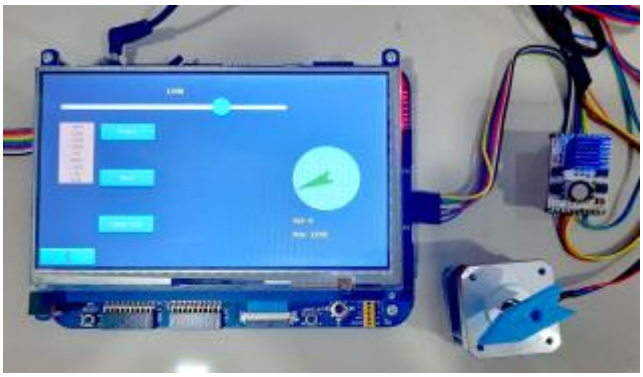
V. PENGUJIAN DAN ANALISIS HASIL

Pengujian dilakukan meliputi dua hal, yang pertama adalah menguji keberhasilan fungsi-fungsi *hardware* maupun *software* (Gambar 8) dan ujicoba keakuratan tampilan grafis dengan kondisi aktual motor.

Gambar 8 Ujicoba pada tayangan 1 (*screen 1*)

Ujicoba fungsi yang telah berhasil pada tayangan pertama yaitu:

1. Tampilan layar LCD-TFT 7 inch sesuai dengan rancangan TouchGFX
2. Fungsi layar sentuh sudah berhasil, ditunjukkan adanya respon saat *button* ditekan dan *slider* digeser. Perpindahan antar *screen* juga berhasil
3. Pengendalian posisi dan kecepatan motor diatur dengan menggunakan *slider*.
4. Pengaturan *microstep* dilakukan dengan menekan *button* pada pilihan *full step*, 2,4,8,…64
5. Terdapat teks untuk keperluan *debugging* yang berisi informasi perintah baca tulis SPI ke TMC5160 dengan format bilangan heksadesimal. Terdapat juga informasi tentang *counter* yang menandakan proses *multi-tasking* berjalan dengan baik
6. Pengaturan motor dengan multi target dapat dilakukan secara berurutan (*sequential*) yang disertai indikator posisi berupa jarum (Gambar 9).



Gambar 9 Ujicoba pada tayangan 2 (*screen 2*)

Pengujian kedua dilakukan untuk mengukur bagaimana kinerja sistem dilihat dari ketepatan informasi grafis terhadap aktual posisi motor (tanpa beban). Cara mengujinya adalah secara visual dengan bantuan kamera video dengan kecepatan 120 fps yang membandingkan tayangan indikator dan teks pada LCD dan posisi petunjuk pada motor.

Tabel 2 Pengukuran posisi dan selisih tiap *frame*

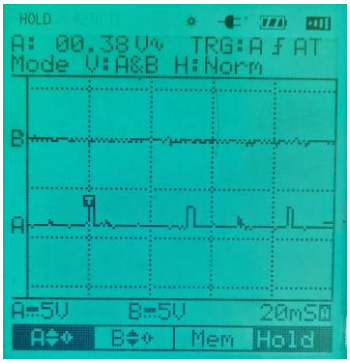
	k: **88 step/s**		k: **1075 step/s**		k: **2350 step/s**	
i	**p**	**$p_{(i+1)} - p_i$**	**p**	**$p_{(i+1)} - p_i$**	**p**	**$p_{(i+1)} - p_i$**
1	45	2	48	25	47	55
2	47	2	73	25	102	54
3	49	2	98	25	156	55
4	51	2	123	25	211	55
5	53		148		266	

i = nomor *frame*; p = posisi (satuan: step);
k= kecepatan (satuan: step/s)

Pengamatan dengan video dilakukan tiga kali percobaan masing-masing dengan kecepatan berbeda. Ujicoba ini menghasilkan data selisih posisi antara satu pembacaan dan pembacaan sebelumnya yang berkaitan dengan komunikasi SPI pada TMC5160 (data dapat dilihat di Tabel 2).

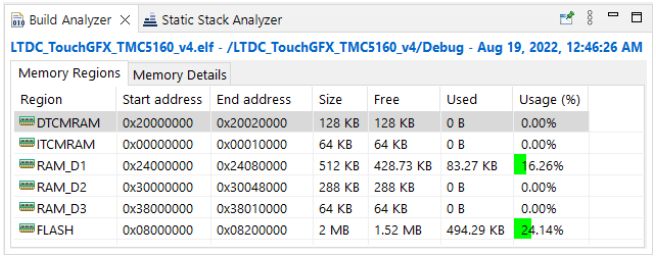
Pengukuran *frame rate* dilakukan dengan bantuan osiloskop pada sebuah kaki GPIO terhadap GND. Sebelumnya, pin GPIO ini telah diprogram untuk memberi tanda berapa lama tampilan TouchGFX membutuhkan waktu untuk *render* (proses memperbarui *pixel* demi *pixel* pada layar). Hasil pengukuran menunjukkan periode *render*

adalah setiap 33ms (Gambar 10). Di kurun waktu ini, komunikasi ke TMC5160 dilakukan untuk mendapatkan nilai posisi dan kecepatan aktual (di bagian kelas *View*, seperti dijelaskan sebelumnya). Dengan kata lain, pembaruan nilai posisi dan kecepatan motor yang ditampilkan di layar dilakukan dalam satu detik sebanyak 1000ms/33ms=30 kali.



Gambar 10 Tampilan osiloskop saat mengukur frame rate

Penggunaan memori pada mikrokontroler dilakukan untuk melihat sejauh mana aplikasi dapat dikembangkan lebih lanjut. Hasil analisis penggunaan memori ditangkap dari panel *Build Analyzer* di STM32CubeIDE pada saat selesai dikompilasi (Gambar 11), menunjukkan penggunaan RAM sebesar 16.26% dan penggunaan FLASH untuk penyimpanan program sebesar 24.14%.



Gambar 11 Penggunaan memori pada aplikasi mikrokontroler

VI. KESIMPULAN

Berdasarkan ujicoba dan hasil analisis, maka dapat ditarik kesimpulan sebagai berikut:

1. Pengendalian stepper motor dengan mikrokontroler STM32H743 yang terintegrasi dengan LCD-TFT layar sentuh kapasitif 7 inch telah berhasil dibuat. Motor dapat dikendalikan posisi dan kecepatannya. Informasi posisi dan kecepatan aktual dapat dibaca dan ditampilkan berupa teks. Aplikasi membutuhkan RAM 16.26% dan flash 24.14% dari total kapasitas.
2. Aplikasi GUI dengan konsep *multi-tasking* telah berhasil berjalan dengan *frame rate* sebesar 30fps. Aplikasi dikembangkan menggunakan TouchGFX Designer versi 4.18.1, STM32CubeMX versi 6.5 dan STM32CubeIDE versi 1.9.0

VII. SARAN

Berdasarkan kapasitas memori yang masih tersedia cukup banyak, maka prototip aplikasi pengendalian gerakan dapat dikembangkan lebih lanjut. Saran yang disampaikan yaitu:

1. Perhitungan interpolasi untuk beberapa motor dapat dilakukan melalui mikrokontroler
2. Integrasi dengan *external storage* menggunakan SDMMC dapat dilakukan untuk penyimpanan program CNC misalnya G-Code.
3. UX (*user experience*) dapat dikembangkan pada antarmuka GUI untuk mendapatkan relevansi yang lebih besar terhadap kebutuhan dan pengalaman pengguna.

DAFTAR ACUAN

[1] L. Shangyang, D. Zhekang, L. Huipin, G. Mingyu, “STM32 and UCOSIII based Engraving Machine Motion Control System”, *Conference: Chinese Control And Decision Conference (CCDC)*, 2020

[2] S Ganesh Kumar, Nitish Kumar B V, Ranjith D, Prakash S, Roshan S V, "Prototyping of Mini CNC Milling Machine using Microcontroller", *International Research Journal of Modernization in Engineering Technology and Science*, vol. 03, issue:04,April -2021

[3] I. Virgala, M. Kelemen, A. Gmiterko, and T. Lipták, “Control of Stepper Motor by Microcontroller.” *Journal of Automation and Control*, vol. 3, no. 3, pp. 131-134, 2015. doi:10.12691/automation-3-3-19.

[4] Bednarski B, Jackiewicz K, Gałecki A. “*Influence of Microstepping Signal Shape on Shaft Movement Precision and Torque Variation of the Stepper Motor”*. Energies. 2021; 14(19):6107. https://doi.org/10.3390/en14196107

[5] Y.H. Hee, M.K. Ishak, M.S.M. Assari, M.T.A. Seman. "Embedded operating system and industrial applications: a review", *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 3, pp 1687-1700, May 2021

[6] STMicroelectronics, *STM32H743/753*. Available: https://www.st.com/en/microcontrollers-microprocessors/stm32h743-753.html

[7] STMicroelectronics. *X-CUBE-TOUCHGFX*. Available: https://www.st.com/en/development-tools/touchgfxdesigner.html

[8] Waveshare, OpenH743I-C, Available: https://www.waveshare.com/wiki/OpenH743I-C

[9] B. Dwersteg, *TMC5160/TMC5160A Datasheet Rev 1.17*, Trinamic, 2022.

[10] Trinamic Blog, *TMC5160 Breakout Board*, Available: https://www.trinamic.com/support/eval-kits/details/tmc5160-bob/