

Desain Perangkat Lunak Pengirim GCode Pada Mesin Mini
Computer Numerical Control Berbasis GRBL

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat
Untuk Memperoleh Gelar Sarjana Teknik Mesin



Disusun Oleh:

Nama : Himawan Akbar Mahendra
No. Mahasiswa : 17525026
NIRM : 2017023591

JURUSAN TEKNIK MESIN
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM INDONESIA
YOGYAKARTA

2021

PERNYATAAN KEASLIAN

Dengan ini saya menyatakan bahwa karya tulis ilmiah yang saya buat merupakan karya sendiri bukan hasil plagiarisme dari karya tulis yang dibuat oleh orang lain. Semua referensi dan kutipan yang saya tulis pada karya tulis ini saya cantumkan sitasi dan sumber pustakanya. Apabila dikemudian hari saya dianggap melakukan pelanggaran hak kekayaan intelektual dan yang saya tulis pada karya ilmiah ini tidak benar, maka saya bersedia menerima sanksi dan hukuman yang berlaku.

Yogyakarta, 4 Oktober 2021



Himawan Akbar Mahendra

LEMBAR PENGESAHAN DOSEN PEMBIMBING

Desain Perangkat Lunak Pengirim GCode Pada Mesin Mini *Computer Numerical Control* Berbasis GRBL

TUGAS AKHIR

Disusun Oleh:

Nama : Himawan Akbar Mahendra

No. Mahasiswa : 17525026

NIRM : 2017023591

Yogyakarta, 13 September 2021

Dosen Pembimbing



Mohammad Faizun, S.T., M.Eng., Ph.D.

NIP. 115250101

LEMBAR PENGESAHAN DOSEN PENGUJI

Desain Perangkat Lunak Pengirim GCode Pada Mesin Mini *Computer Numerical Control Berbasis GRBL*

TUGAS AKHIR

Disusun Oleh:

Nama : Himawan Akbar Mahendra
No. Mahasiswa : 17525026
NIRM : 2017023591

Tim Penguji

Mohammad Faizun, S.T., M.Eng., Ph.D.

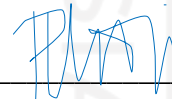
Ketua



Tanggal : 7 Oktober 2021

Purtojo, S.T., M.Sc.

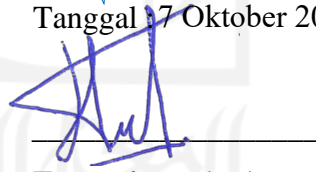
Anggota I



Tanggal : 7 Oktober 2021

Donny Suryawan, S.T., M.Eng.

Anggota II



Tanggal : 6 Oktober 2021

Mengetahui



Ketua Jurusan Teknik Mesin

Dr. Eng. Risdiyono, S.T., M.Eng.



HALAMAN PERSEMBAHAN

Alhamdulillah Rabbil'alamin Segala puji bagi Allah Subhanallahu wa Ta'ala atas Rahmat dan hidayah-Nya yang telah memberikan nikmat dan kemudahan yang luar biasa sehingga skripsi ini dapat diselesaikan. Karya sederhana ini dipersembahkan untuk:

Keluarga saya Bapak Ir. Suwanto, Ibu Dra. Wahyu Nuryani, Kakak Septo Aji Akbar Nugroho S.Ak. dan untuk saya Himawan Akbar Mahendra Terima kasih atas semua cinta, kasih sayang, perhatian, doa dan dukungan yang tiada henti yang selalu diberikan kepada saya. Terima kasih atas segala pengorbanan, waktu dan kesabaran yang telah diberikan kepada saya. Semoga Allah Subhanallahu wa Ta'ala membalas dengan kebaikan yang lebih banyak dan kemuliaan yang lebih tinggi
Aamiin.

HALAMAN MOTTO

“Kayu Gung Susuhing Angin”

*“Kayu kuwi tegese karep, gung tegese gedhe
susuh angin iku telenging napas
maknane karep sing gedhe bisane kasembadan
kudu sinartan alinging napas, weninging cipta,
meneping pancadriya, sumelehing rasa”*



KATA PENGANTAR ATAU UCAPAN TERIMA KASIH

Assalamu'alaikum Wr. Wb.

Segala puji dan syukur penulis panjatkan kehadirat Allah SWT atas segala rahmat serta hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul “Desain Perangkat Lunak Pengirim GCode Pada Mesin Mini *Computer Numerical Control* Berbasis GRBL” tepat pada waktunya. Tugas Akhir ini disusun sebagai salah satu persyaratan untuk meraih gelar sarjana strata 1 jurusan Teknik Mesin, Universitas Islam Indonesia.

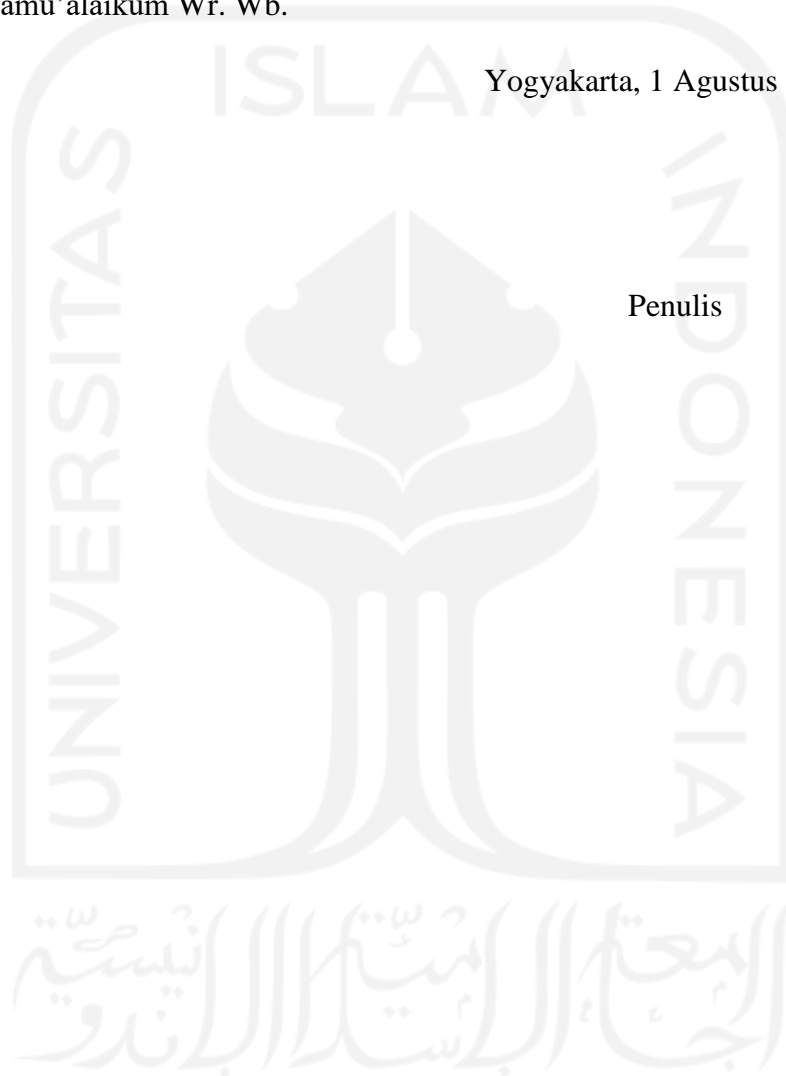
Keberhasilan penulis dalam menyelesaikan Tugas Akhir ini tidak terlepas dari bimbingan, bantuan, dukungan serta dorongan dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada :

1. Allah SWT yang telah melimpahkan segala rahmat-Nya, sehingga saya dapat menyelesaikan tugas akhir ini sebagai mana mestinya,
2. Kedua Orang Tua saya yang sudah mendidik dan membesarkan saya hingga saat ini,
3. Bapak Dr. Eng. Risdiyono, S.T., M.Eng. selaku Ketua Jurusan Teknik Mesin Fakultas Teknologi Industri Universitas Islam Indonesia,
4. Bapak Mohammad Faizun, S.T., M.Eng., Ph.D. selaku dosen pembimbing yang telah memberikan banyak bantuan, bimbingan, serta arahan dalam Tugas Akhir ini,
5. Bapak Arif Budi Wicaksono, S.T., M.T. selaku Dosen Pembimbing Akademik,
6. Seluruh Dosen dan karyawan di Jurusan Teknik Mesin Fakultas Teknologi Industri Universitas Islam Indonesia, yang tidak bisa disebutkan satu-satu, atas ilmu dan bimbingannya serta bantuannya selama penulis berkuliah di Jurusan Teknik Mesin FTI UII,
7. Teman-teman Teknik Mesin UII yang menjadi tempat bertukar pikiran, pandangan dan pengalaman serta saling membantu dalam kebaikan,
8. Widiati Anjarwani yang telah menjadi teman berbagi pikiran dan terus memberikan dukungan, doa dan semangat kepada penulis.

Dan lain-lain yang telah memberikan kelancaran dalam penyusunan laporan ini yang tidak dapat saya sebutkan satu persatu. Dalam menyusun laporan akhir ini, saya menyadari bahwa laporan ini jauh dari kata sempurna, oleh karena itu segala kritik dan saran yang bersifat membangun senantiasa diharapkan untuk menyempurnakan laporan akhir ini. Semoga laporan akhir ini dapat bermanfaat bagi penulis dan bagi pembaca pada umumnya. Aamiin
Wassalamu'alaikum Wr. Wb.

Yogyakarta, 1 Agustus 2021

Penulis



ABSTRAK

Penelitian ini bertujuan untuk merancang perangkat lunak dengan software Visual Studio 2019 dengan *Window Presentation Foundation* (WPF) untuk mengirim *file* GCode secara bergantian, runtut dan dapat dioperasikan secara manual untuk menentukan titik nol pada mesin mini CNC 3 axis berbasis GRBL dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 secara *serial port* dan *wireless*. Pada perancangan ini menggunakan metode *Agile Software Methods and Development*, yang kemudian dilakukan pengujian dengan metode *Black Box* untuk mengetahui fungsionalitas dari perangkat lunak yang dirancang atau dibangun.

Perancangan ini difokuskan dalam perancangan perangkat lunak untuk bertukar informasi antara perangkat lunak dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 pada mesin mini CNC 3 axis berbasis GRBL. Pertukaran informasi yang dilakukan secara *serial port* dengan *baudrate* serta *portname* yang tersedia dan *wireless* dengan *IP address* dari mikrokontroler.

Hasil akhir dari perancangan perangkat lunak menunjukkan bahwa *front end develop* (antarmuka), *back end develop* (sistem) dapat berfungsi dengan baik sesuai dengan konsep dan kriteria. Pengiriman *file* GCode dapat berfungsi dengan baik dan dikirim secara runtut sampai selesai.

Kata kunci: Perangkat lunak, CNC 3 axis berbasis GRBL, Wemos D1 R32 atau ESPDuino32, serial port, wireless

DAFTAR ISI

Halaman Judul	i
Lembar Pengesahan Dosen Pembimbing	iii
Lembar Pengesahan Dosen Penguji	iv
Halaman Persembahan	v
Halaman Motto	vi
Kata Pengantar atau Ucapan Terima Kasih	vii
Abstrak	ix
Daftar Isi	x
Daftar Tabel	xiii
Daftar Gambar	xiv
Daftar Notasi	xvi
Bab 1 Pendahuluan	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	3
1.4 Tujuan Penelitian atau Perancangan	3
1.5 Manfaat Penelitian atau Perancangan	4
1.6 Sistematika Penulisan	4
Bab 2 Tinjauan Pustaka	5
2.1 Kajian Pustaka	5
2.2 Dasar Teori	6
2.2.1 Mesin Mini CNC	6
2.2.2 GRBL CNC Shield	7
2.2.3 Wemos D1 R32 atau ESPDuino32	7
2.2.4 <i>Serial Communication</i>	7
2.2.5 <i>Wireless Communication</i>	8
2.2.6 Bahasa Pemrograman C#	8
2.2.7 Visual Studio	8
2.2.8 <i>Window Presentation Foundation (WPF)</i>	8
2.2.9 <i>Agile Software Methods and Development</i>	10

Bab 3 Metode Penelitian	11
3.1 Alur Perancangan.....	11
3.2 Peralatan dan Bahan.....	12
3.2.1 Alat	12
3.2.2 Bahan	12
3.3 Metode Perancangan.....	14
3.3.1 <i>Plan</i>	15
3.3.2 <i>Define</i>	15
3.3.3 <i>Develop</i>	15
3.3.4 <i>Test</i>	15
3.3.5 <i>Maintenance</i>	16
3.4 Perancangan	16
3.4.1 Perancangan Perangkat Lunak.....	16
3.4.2 Perancangan Sistem Kontrol	23
Bab 4 Hasil dan Pembahasan	27
4.1 Hasil Perancangan.....	27
4.1.1 Hasil <i>Front End Develop</i> (Antarmuka)	27
4.1.2 Hasil <i>Back End Develop</i> (Sistem)	32
4.2 Hasil Pengujian	37
4.2.1 Pengujian <i>Front End Develop</i> (Antarmuka).....	37
4.2.2 Pengujian <i>Back End Develop</i> (Sistem).....	43
4.2.3 Pengujian Pengiriman Data Terhadap Gerak CNC	47
4.2.4 Pengujian Usability	50
4.3 Analisis dan Pembahasan.....	53
4.3.1 Analisis <i>Front End Develop</i> (Antarmuka).....	53
4.3.2 Analisis <i>Back End Develop</i> (Sistem).....	54
4.3.3 Analisis Pengiriman Data Terhadap Gerak CNC	54
4.3.4 Analisis Usability	54
Bab 5 Penutup.....	56
5.1 Kesimpulan	56
5.2 Saran atau Penelitian Selanjutnya.....	56
Daftar Pustaka	57

Lampiran 1.....	60
Lampiran 2.....	67
Lampiran 3.....	69
Lampiran 4.....	80



DAFTAR TABEL

Tabel 3-1. Alat.....	12
Tabel 3-2. Bahan.....	13
Tabel 3-3. Pengaturan microstep driver pada CNC Shield V3.....	25
Tabel 4-1. <i>UI Element</i> dan fungsinya.....	29
Tabel 4-2. Hasil pengujian <i>Front End Develop</i>	37
Tabel 4-3. Hasil pengujian <i>Back End Develop</i>	43
Tabel 4-4. Hasil pengujian pengiriman data terhadap gerak CNC.....	47
Tabel 4-5. Gcode yang dikirim ke CNC.....	49
Tabel 4-6. Kriteria persentase kuisisioner.....	50
Tabel 4-7. Daftar pertanyaan kuisisioner.....	51
Tabel 4-8. Hasil pengujian usability.....	51
Tabel 4-9. Pengelolaan skala kuisisioner.....	53

DAFTAR GAMBAR

Gambar 2-1. Tampilan Visual Studio menggunakan <i>framework</i> WPF.....	9
Gambar 3-1. Diagram alir Perancangan	11
Gambar 3-2. Adaptor 24V	13
Gambar 3-3. CNCShield V3.....	13
Gambar 3-4. Wemos D1 R32/ESPDuino32	13
Gambar 3-5. Adaptor 5V	13
Gambar 3-6. <i>Firmware</i> GRBL_ESP32	14
Gambar 3-7. Jumper <i>connector</i> 2pin	14
Gambar 3-8. Kabel <i>micro</i> USB	14
Gambar 3-9. Motor Driver DRV8825	14
Gambar 3-10 <i>Agile Software Methods and Development</i>	15
Gambar 3-11. <i>Plan</i> diagram konsep dasar sistem	17
Gambar 3-12. Konsep rancangan desain antarmuka	19
Gambar 3-13. Konsep rancangan desain antarmuka <i>SplashScreen</i>	19
Gambar 3-14. Konsep rancangan desain antarmuka utama	20
Gambar 3-15. Konsep alur kerja <i>Back End Develop</i> koneksi melalui <i>Serial Port</i>	21
Gambar 3-16. Konsep alur kerja <i>Back End Develop</i> koneksi melalui <i>Wireless/Network</i>	21
Gambar 3-17. <i>Back End Develop</i> pengiriman <i>file</i> GCode.....	22
Gambar 3-18. <i>Back End Develop manual control</i>	23
Gambar 3-19. Skema diagram CNC Shield V3 dengan Arduino Uno.....	23
Gambar 3-20. Skema diagram perancangan perangkat keras.....	24
Gambar 3-21. Perancangan perangkat keras	24
Gambar 3-22. Library tambahan pada GRBL_ESP32 untuk Wemos D1 R32/ESPDuino32	25
Gambar 3-23 Konfigurasi <i>upload firmware</i> GRBL_ESP32.....	26
Gambar 4-1. Hasil perancangan antarmuka pengguna koneksi	27
Gambar 4-2. Hasil perancangan antarmuka pengguna koneksi (<i>Baudrate</i>).....	28
Gambar 4-3. Hasil perancangan antarmuka kondisi terhubung	28

Gambar 4-4. Hasil perancangan antarmuka pengiriman <i>file</i> GCode.....	29
Gambar 4-5. Hasil perancangan antarmuka pengguna akhir.....	29
Gambar 4-6. Hasil perancangan sistem koneksi <i>serial port</i> dan <i>wireless</i>	32
Gambar 4-7. Hasil perancangan sistem kondisi <i>connect</i> dan <i>disconnect</i>	33
Gambar 4-8. Hasil perancangan sistem definisi mode.....	34
Gambar 4-9. Hasil perancangan sistem fungsi 1 dalam pertukaran data dari sebuah <i>file</i>	34
Gambar 4-10. Hasil perancangan sistem fungsi 2 dalam pertukaran data dari sebuah <i>file</i>	35
Gambar 4-11. Hasil perancangan sistem SendLine.....	36
Gambar 4-12. Hasil perancangan sistem perpindahan manual.....	36
Gambar 4-13. Hasil perancangan sistem Textbox manual.....	37
Gambar 4-14. Tampilan ketika manual kontrol X- sebelum diklik.....	42
Gambar 4-15. Tampilan ketika manual kontrol X- cursor pada tombol.....	42
Gambar 4-16. Tampilan manual kontrol pada X- ketika diklik.....	43
Gambar 4-17. Menyambung dan memutus koneksi <i>serialport</i> atau <i>wireless</i>	46
Gambar 4-18. Sistem bekerja dengan koneksi <i>serial port</i>	46
Gambar 4-19. Sistem bekerja dengan koneksi <i>wireless</i>	47
Gambar 4-20. Hasil pengujian pengiriman data ke CNC secara <i>serial port</i>	48
Gambar 4-21. Hasil pengujian pengiriman data ke CNC secara <i>wireless</i>	49

DAFTAR NOTASI

CNC	= <i>Computer Numerical Control</i>
NC	= <i>Numerical Control</i>
UI	= <i>User Interface</i>
OPC	= <i>Open Platform Communications</i>
PC	= <i>Personal Computer</i>
TCP	= <i>Transmission Control Protocol</i>
IP	= <i>Internet Protocol</i>
ISO	= <i>The International Organization for Standardization</i>
MB	= <i>MegaByte</i>
UART	= <i>Universal Asynchronous Receiver Transmitter</i>
C#	= Bahasa pemrograman C#/C Sharp
C++	= Bahasa pemrograman C++
C	= Bahasa pemrograman C
WPF	= <i>Window Presentation Foundation</i>
XAML	= <i>Extensible Application Markup Language</i>
X	= Sumbu atau axis X
Y	= Sumbu atau axis Y
Z	= Sumbu atau axis Z

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Proses otomatisasi adalah sebuah proses yang dapat mempermudah, mempercepat, dan mampu menghasilkan produk yang lebih baik yang bertujuan untuk meminimalisasi keterlibatan manusia pada pekerjaan yang berulang. Otomatisasi telah diterapkan dalam dunia manufaktur dalam bentuk teknologi “*Industry 4.0*”. Dalam perkembangan teknologi otomasi didasari oleh teknologi baru dan mutakhir, elektronik, *Artificial Intelligence* (AI) dan kegiatan produksi yang terintegrasi dengan komputer. Otomatisasi ini mencakup kemampuan pemrosesan pada sistem, namun dalam mengintegrasikan manusia dengan sistem untuk mencapai otomatisasi bukan pekerjaan yang mudah dan sederhana. Faktor manusia terutama pada aspek kognitif sering disalahpahami dan diabaikan dalam desain sistem. (Madakam et al., 2018)

Desain sistem untuk menghasilkan, mengedit, mengeksekusi, memantau, dan menjalankan program aplikasi memerlukan logika sistem, gerakan dan proses kontrol sistem untuk mengendalikan mekanisme otomatisasi. (Sadre et al., 1996) Pada tahap desain sistem ini menentukan efisiensi dalam proses produksi secara otomatis. Efisiensi dapat diwujudkan dengan memilih solusi desain sistem terbaik untuk otomatisasi proses produksi berdasarkan parameter seperti kecepatan, pemrosesan, akurasi, suhu, dan mode manajemen. (Tomov, 2017)

Desain sistem otomatisasi proses produksi yang dikembangkan dengan beberapa parameter, tingkat presisi dan kecepatan merupakan bagian tuntutan terpenting. Proses mengintegrasikan manusia dengan sistem untuk mencapai otomatisasi memerlukan perangkat lunak yang dapat mempermudah proses produksi. Perangkat lunak berfungsi untuk mengintegrasikan komunikasi antara manusia dengan mesin. Sehingga mesin dapat dikontrol dan berfungsi sesuai dengan gerakan pada desain sistem otomatisasi yang telah dirancang. (Tomov, 2017) Dalam proses pengontrolan menggunakan perangkat lunak memerlukan PC (*Personal Computer*) sebagai media komunikasi dengan mesin. Pada saat ini,

dunia manufaktur telah menerapkan proses otomatisasi mesin dijalankan dengan bantuan komputer berdasarkan perintah yang diekstrak menjadi kode program, atau yang dikenal dengan mesin CNC.

Mesin CNC (*Computer Numerical Control*) dapat melakukan beberapa pekerjaan seperti, *drilling*, *milling*, dan *engraving*. Perkembangan teknologi yang signifikan pada PC (*Personal Computer*) memudahkan dalam penambahan fungsional pada mesin CNC (*Computer Numerical Control*) modern, dan dapat memudahkan untuk mengganti atau mengembangkan algoritma pengontrolan. (Rocha et al., 2010)

Mesin CNC yang terus berkembang dan semakin canggih menyebabkan Mesin CNC mempunyai harga dan biaya perawatan yang mahal, hal ini mewujudkan adanya alternatif yang dapat mengurangi biaya atau biasa disebut dengan mini CNC. Mesin mini CNC ini berbasis GRBL dengan sistem pengontrolan melalui Arduino dan 3 driver motor. Perkembangan *firmware* GRBL yaitu *firmware* GRBL yang dapat digunakan pada ESP32 sehingga memungkinkan pengiriman data melalui *wireless*, sehingga dapat meningkatkan efisiensi dalam proses integrasi antara manusia dengan sistem. Proses pemesanan dengan mesin mini CNC memerlukan perangkat lunak dalam pengiriman GCode dari *Personal Computer* (PC) ke mikrokontroler. (Parajuli et al., 2021)

Inovasi desain perangkat lunak pengiriman GCode pada mesin mini CNC ini diharapkan dapat meningkatkan efisiensi pada proses pemesanan dengan memanfaatkan jaringan *wireless* dan memiliki desain yang sederhana atau mudah dalam pengoperasiannya. Dalam proses pengiriman GCode untuk mengontrol mesin mini CNC menggunakan mikrokontroler Wemos D1 R32 yang sudah terintegrasi dengan modul *wifi*.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah disampaikan, maka diperoleh rumusan masalah yaitu: Bagaimana merancang desain perangkat lunak pengirim GCode yang sederhana dan dapat dioperasikan dengan jaringan *wireless* pada mesin mini CNC (*Computer Numerical Control*) berbasis GRBL?

1.3 Batasan Masalah

Berdasarkan rumusan masalah tersebut pada perancangan desain perangkat lunak pengirim GCode pada mesin mini CNC (*Computer Numerical Control*) berbasis GRBL ini akan dilakukan pembatasan masalah sebagai berikut:

1. Mendesain perangkat lunak yang dikembangkan dari bahasa pemrograman C# (CSharp) dan dapat berkomunikasi dengan mikrokontroler Wemos D1 R32 atau ESPDuino32.
2. Dalam proses perancangan dan pembuatan perangkat lunak menggunakan Aplikasi Visual Studio 2019 WPF (*Window Presentation Foundation*)
3. Perangkat lunak dapat mengirimkan GCode perbaris secara bergantian dan runtut.
4. Perangkat lunak hanya dapat mengirimkan GCode dari *file* dan dapat dioperasikan secara manual pada mesin mini CNC 3-axis berbasis GRBL.

1.4 Tujuan Penelitian atau Perancangan

Berdasarkan rumusan masalah tersebut, maka perancangan ini memiliki tujuan sebagai berikut:

1. Merancang dan membangun perangkat lunak dengan desain sederhana dan dapat mengirim GCode secara bergantian, runtut pada mesin mini CNC 3-axis berbasis GRBL dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 secara *serial port* maupun *wireless*.
2. Merancang dan membangun perangkat lunak dengan desain sederhana dan dapat dioperasikan secara manual untuk menentukan titik nol atau titik awal pada Mesin Mini CNC 3-axis berbasis GRBL dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 secara *serial port* maupun *wireless*.

1.5 Manfaat Penelitian atau Perancangan

Manfaat dalam proses perancangan ini diharapkan dapat menjadi kajian dalam proses penelitian atau perancangan terkait pengembangan perangkat lunak yang berkaitan dengan mesin mini CNC dan pengembangan sistem otomasi yang berkaitan dengan mesin produksi.

1.6 Sistematika Penulisan

Pada bagian sistematika penulisan dilakukan secara runtut untuk mempermudah dalam melakukan penelitian atau perancangan serta penyusunan pembahasan dan penarikan kesimpulan. Sistematika penulisan adalah sebagai berikut:

BAB I : PENDAHULUAN

Pada bab ini dijelaskan latar belakang, rumusan masalah, batasan, tujuan perancangan, manfaat, dan sistematika penulisan.

BAB II : TINJAUAN PUSTAKA

Pada bab ini dijelaskan teori-teori dan penelitian terdahulu yang digunakan sebagai acuan dan dasar dalam penelitian.

BAB III : METODOLOGI PERANCANGAN

Pada bab ini dijelaskan metode yang digunakan dalam perancangan meliputi alur perancangan, tahapan, dan peralatan yang digunakan.

BAB IV : HASIL DAN PEMBAHASAN

Pada bab ini dijelaskan hasil perancangan dan pembahasannya.

BAB V : KESIMPULAN DAN SARAN

Pada bab ini meliputi kesimpulan dari perancangan atau pembahasan dan saran untuk pengembangan perancangan selanjutnya.

BAB 2

TINJAUAN PUSTAKA

2.1 Kajian Pustaka

G. Rodríguez et al., 2014 melalui penelitiannya yang berjudul “*G-Code Interpreter Development Using Microsoft Visual Basic for ABL63 Control Systems*” mengenai pengembangan pada antarmuka pengguna untuk menerjemahkan GCode dengan Microsoft Visual Basic 2010 yang dibersamai dengan protokol komunikasi OPC. Antarmuka ini digunakan untuk menampilkan deteksi eror dalam verifikasi modul menurut parameter, identifikasi garis yang sesuai, dan memonitor posisi XY secara *Realtime*, dan mengijinkan untuk mengeksekusi GCode perbaris pada mode otomatis. Antarmuka tidak berisikan informasi terkait kecepatan pemotongan, kecepatan pahat, kecepatan *spindle* dari NC program, antarmuka dikembangkan menggunakan inkremen dan menggunakan kembali metodologi yang berorientasi.

(Alfatah, 2013) melalui penelitiannya yang berjudul “Rancangan Perangkat Lunak G Code Interpreter Untuk Pengendalian CNC 3 Aksis Berbasis Mikrokontroler” membahas terkait perancangan dan pembuatan GCode *interpreter* dengan Microsoft Visual Studio 2010. Mesin CNC 3-axis digunakan untuk pembuktian dalam menerjemahkan GCode menjadi gerak interpolasi linier dan gerak interpolasi melingkar. *Software* yang dirancang menggunakan ketelitian 1mm, dan melakukan interpolasi pada bidang XY. Perangkat lunak yang dirancang ada dua macam, program *Human Machine Interface* (HMI) dan perangkat lunak mikrokontroler sebagai *indexer*. HMI ditanam dalam *personal computer* (PC) yang dilengkapi rancangan perangkat lunak GCode *interpreter* untuk Pengendalian CNC 3 Aksis Berbasis Mikrokontroler.

(Rahmasari, 2017) melalui penelitiannya yang berjudul “Perancangan dan Pembuatan Antarmuka Berbasis Visual Studio (VB.NET) dan Komunikasi Berbasis TCP/IP Pada Mesin CNC *Routing 2.5D*” membahas terkait pembuatan perangkat lunak untuk mengoperasikan mesin CNC *routing 2.5D* dengan TCP/IP. Dalam perancangannya menggunakan Raspberry Pi sebagai mikrokontrolernya.

Perancangan ini menghasilkan perangkat lunak yang dapat melakukan perhitungan seperti jumlah *feedrate* dan mempunyai tampilan yang menarik untuk memudahkan pengguna dalam melakukan pengoperasian.

Yerra et al., 2017 melalui penelitiannya yang berjudul “*Development of an Open Type CNC System for a 3-Axis Micro CNC Machine*” membahas terkait pengembangan CNC berbiaya rendah yang mampu interpolasi 3-axis secara simultan operasi. Biaya yang lebih rendah dicapai dengan menggabungkan fitur antarmuka PC standar dengan sistem CNC berbasis mikrokontroler dalam sistem tertanam berbasis Arduino. Sistem yang digunakan adalah *offline GCode Parser* dan diinterpretasikan pada mikrokontroler dari USB.

Yakovlev et al., 2020 melalui penelitian yang berjudul “*Software Development for 3D Visualization of G-code When Working With CNC Machines*” melakukan pengembangan yang akan membantu operator dalam melakukan pekerjaan menggunakan mesin CNC. Dengan menggunakan aplikasi TwinCAT untuk mempresentasikan regenerasi aplikasi dari otomasi dan menggunakan proses operasi pada PC. TwinCAT adalah perangkat lunak terpadu untuk semua *Beckhoff operating systems*. TwinCAT terdiri dari sistem kontrol *run-time real-time*, pemrograman, diagnostik dan konfigurasi sistem.

Berdasarkan dari penelitian sebelumnya diatas maka dilakukan penelitian tentang perancangan perangkat lunak dengan desain sederhana dan dapat mengirim GCode secara bergantian, runtut dan dapat dioperasikan secara manual untuk menentukan titik nol atau titik awal pada mesin mini CNC 3-axis berbasis GRBL dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 secara *serial port* maupun *wireless*.

2.2 Dasar Teori

2.2.1 Mesin Mini CNC

Mesin CNC adalah mesin yang dikontrol dengan PC (*Personal Computer*) menggunakan bahasa pemrograman *Numerical Language* (perintah data dengan kode angka, huruf dan simbol) sesuai dengan standar ISO. Mesin mini CNC adalah mesin CNC dengan ukuran yang lebih kecil yang dikontrol dengan

mikrokontroler dan memiliki fungsi yang sama dengan Mesin CNC pada umumnya. Perbedaan Mesin mini CNC dengan Mesin CNC pada umumnya hanya pada ukuran komponen yang dioperasikan. (Mahmood et al., 2019)

2.2.2 GRBL CNC Shield

GRBL adalah sebuah *firmware* yang digunakan untuk mikrokontroler sebagai program kontrol yang berisikan *GCode command* sebagai bagian utamanya. GRBL adalah *firmware* yang *open-source* digunakan untuk kontrol pergerakan pada CNC *milling* dengan performa tinggi dan dengan biaya murah. (Sonny Jeon, 2021)

GRBL CNC Shield merupakan perangkat keras yang dapat digunakan oleh satu mikrokontroler. GRBL CNC Shield akan mengontrol driver motor pada motor stepper dengan perangkat lunak yang mendukung versi GRBL. (Mahmood et al., 2019)

2.2.3 Wemos D1 R32 atau ESPDuino32

ESP32 adalah sebuah perangkat keras mikrokontroler yang memiliki modul Wifi, Bluetooth dan memiliki kapasitas memory mencapai 4MB. perangkat canggih yang cocok untuk IoT (*Internet of Things*) aplikasi dalam hal properti dan harga. Sedangkan, Wemos D1 R32 adalah perangkat keras mikrokontroler yang berbentuk UNO dengan modul ESP32 atau dapat disebut ESP32 versi UNO. (Swati & Rao, 2019)

2.2.4 Serial Communication

Serial komunikasi terdapat dua tipe yaitu serial secara asinkron dan serial secara sinkron. Serial secara sinkron adalah komunikasi, dimana didalamnya hanya terdapat satu paket (pengirim dan penerima) yang menghasilkan waktu dan dikirim bersamaan dengan data. Serial secara asinkron adalah komunikasi, dimana didalamnya dapat diberikan beberapa paket (pengirim dan penerima) yang masing-masing menghasilkan waktu tetapi hanya data yang disalurkan tanpa waktu. *Universal Asynchronous Receiver Transmitter* (UART) adalah contoh dari serial secara asinkron yang biasanya digunakan dalam *serial port*

pada komputer (COM). UART adalah circuit yang terintegrasi dan memiliki peran paling penting pada komunikasi secara serial.(Laddha & Thakare, 2013)

2.2.5 Wireless Communication

Wireless Communication adalah sebuah jaringan data, telekomunikasi tanpa kabel atau nirkabel sehingga dapat diakses dengan jangkauan yang luas dengan sistem pengiriman (*transmitter*) dan sistem penerimaan (*receiver*) dengan sinyal. Berdasarkan dari kecepatan transfer data komunikasi nirkabel lebih baik dan lebih efektif dibanding dengan komunikasi serial. (Rappaport, 2002)

2.2.6 Bahasa Pemrograman C#

C# (C Sharp) adalah bahasa pemrograman level tinggi seperti bahasa Java dan C++. C# merupakan bahasa pemrograman yang bertujuan untuk membuat dan mengembangkan perangkat lunak yang berjalan pada .NET Framework. C# adalah salah satu yang paling populer karena sintaks berbasis C yang disederhanakan. (Nakov, 2013)

2.2.7 Visual Studio

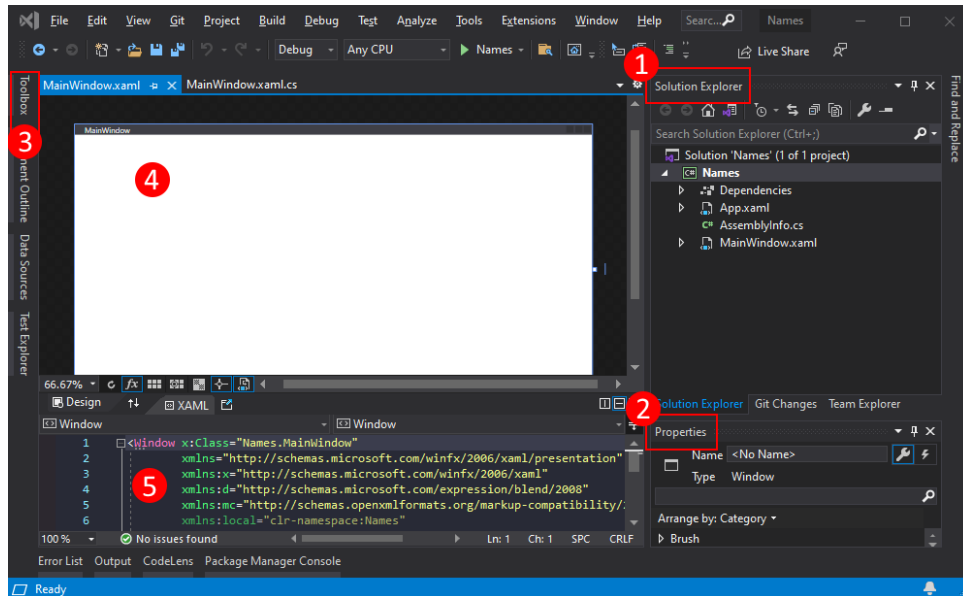
Visual Studio adalah sebuah *Integrated development environment* (IDE) yang digunakan untuk mengembangkan perangkat lunak pada Windows dan *platform* .NET Framework. Visual Studio (VS) mendukung bahasa pemrograman yang berbeda (misalnya C#, VB.NET dan C++) dan berbagai teknologi pengembangan perangkat lunak. Visual Studio digunakan untuk menulis kode atau program, mengkompilasi, mengeksekusi, menjalankan aplikasi dan menguji aplikasi, merancang antarmuka pengguna (form, dialog, halaman web, kontrol visual, dan lainnya). (Nakov, 2013)

2.2.8 Window Presentation Foundation (WPF)

Windows Presentation Foundation (WPF) adalah sebuah *framework* yang dibuat dengan *rendering* berbasis vektor. WPF menyediakan fitur pengembangan aplikasi yang mencakup *Extensible Application Markup Language* (XAML),

controls, data binding, layout, 2D dan 3D graphics, animation, styles, templates, documents, media, text, dan typography. WPF adalah bagian dari .NET, sehingga dapat membangun aplikasi yang menggabungkan elemen lain dari .NET API. (Microsoft, 2021)

Proses pengembangan aplikasi dengan menggunakan *framework* WPF, berikut ini merupakan penjabaran tampilan pada Visual Studio pada gambar 2-1



Gambar 2-1. Tampilan Visual Studio menggunakan *framework* WPF

Sumber: (Microsoft, 2021)

1. *Solution Explorer*

Solution Explorer merupakan panel berisi nama *project file*, nama *code*, dan *resources*.

2. *Properties*

Properties merupakan panel yang menampilkan pengaturan properti pada sebuah item yang dipilih.

3. *Toolbox*

Toolbox merupakan panel yang berisikan sebuah kontrol yang dapat ditambahkan atau digunakan pada *form*.

4. *XAML designer*

XAML designer merupakan sebuah panel yang berisikan objek tampilan (UI) yang dibangun oleh pengguna. Dalam proses desain menggunakan *toolbox*

yang dipilih akan ditampilkan di panel ini sehingga memudahkan pengguna untuk proses pengeditan desain.

5. XAML code editor

XAML code editor merupakan *code editor* dari *XAML designer*, yang digunakan untuk mendesain UI berdasarkan *code* yang dibuat tanpa harus mendesain dengan mengambil *toolbox* untuk menambahkan dan merubah kontrolnya tanpa *designer*. (Microsoft, 2021)

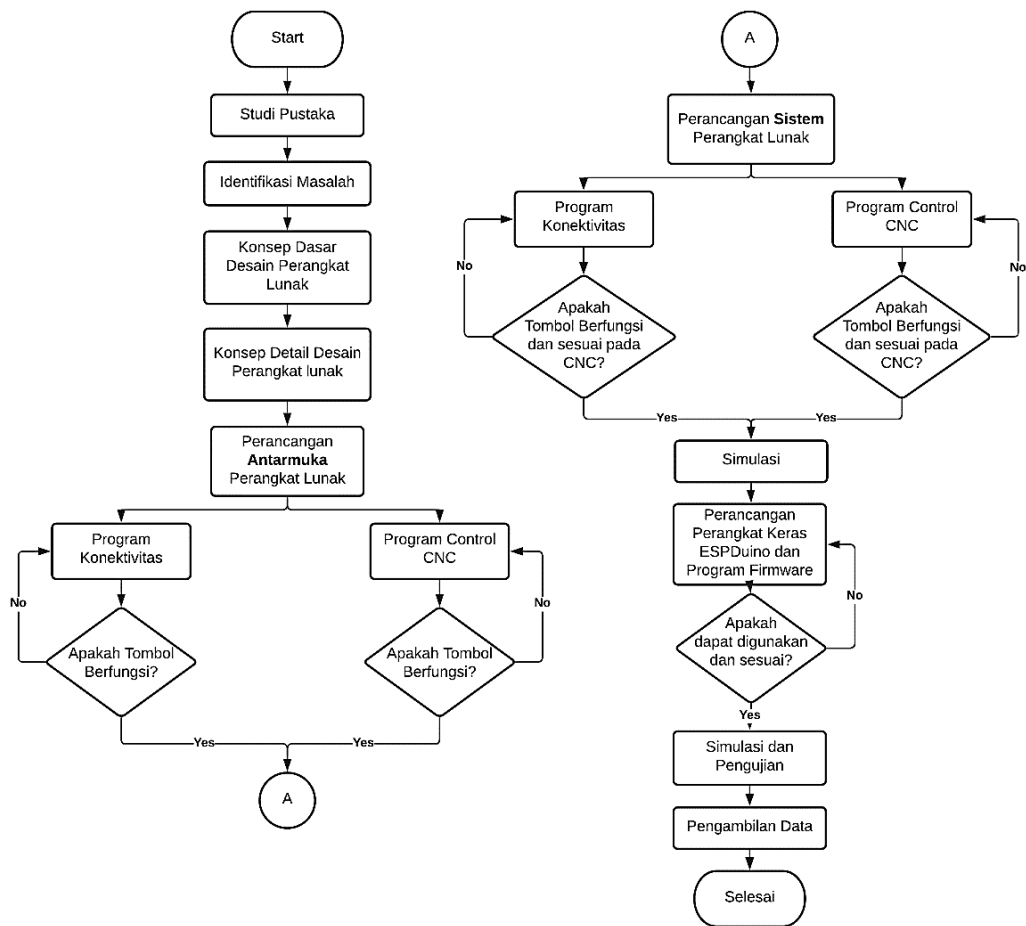
2.2.9 Agile Software Methods and Development

Agile Software Methods and Development adalah proses pengembangan perangkat lunak yang lebih mudah dan cepat berdasarkan pendekatan nilai, prinsip, dan praktik. Metode yang mencakup metode seperti pemrograman ekstrim, dan didorong oleh fitur pengembangan. Sehingga, metode ini dimasukkan dalam *life cycle of software development* karena metode pendekatan yang sangat cepat. (Frauke et al., 2003)

BAB 3 METODE PENELITIAN

3.1 Alur Perancangan

Dalam melakukan proses perancangan terdapat tahapan yang dilakukan. Tahapan-tahapan tersebut dapat dilihat melalui diagram alir pada Gambar 3-1 sebagai berikut:



Gambar 3-1. Diagram alir Perancangan

3.2 Peralatan dan Bahan

Dalam melakukan perancangan alat dan bahan yang akan digunakan dapat ditunjukkan sebagai berikut.

3.2.1 Alat

Alat yang akan digunakan dalam perancangan ini dapat dilihat pada tabel 3-1 sebagai berikut:



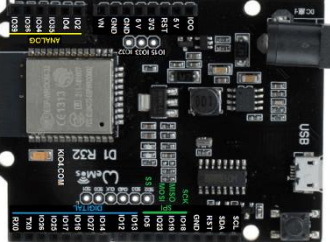

Tabel 3-1. Alat




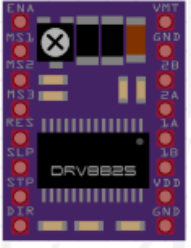
No	Nama Alat	Fungsi
1.	Laptop	Sebagai media untuk konsep desain dan membuat program
2.	CNC berbasis GRBL	Sebagai media simulasi dan pengujian
3.	<i>Software</i> Corel Draw 2020	Sebagai media proses konsep desain antarmuka pengguna
4.	<i>Software</i> Visual Studio 2019	Sebagai media untuk membuat perangkat lunak dengan Bahasa C#(CSharp)
5.	<i>Software</i> Arduino IDE 1.8.13	Sebagai media untuk membuat program pada mikrokontroler ESPDuino32

3.2.2 Bahan

Bahan yang digunakan dalam perancangan ini dapat dilihat pada tabel 3-2 sebagai berikut:

Tabel 3-2. Bahan

No	Nama Bahan	Gambar
1.	Adaptor 24V	 <p data-bbox="922 712 1262 748">Gambar 3-2. Adaptor 24V</p>
2.	CNC Shield V3	 <p data-bbox="911 1151 1273 1187">Gambar 3-3. CNCShield V3</p>
3.	Wemos D1 R32/ESPduino32	 <p data-bbox="938 1473 1251 1563">Gambar 3-4. Wemos D1 R32/ESPduino32</p>
4.	Adaptor 5V	 <p data-bbox="932 1910 1257 1946">Gambar 3-5. Adaptor 5V</p>

No	Nama Bahan	Gambar
5.	<i>Firmware</i> GRBL_ESP32	 <p>Gambar 3-6. <i>Firmware</i> GRBL_ESP32</p>
6.	Jumper <i>connector</i> 2pin	 <p>Gambar 3-7. Jumper <i>connector</i> 2pin</p>
7.	Kabel <i>micro</i> USB	 <p>Gambar 3-8. Kabel <i>micro</i> USB</p>
8.	Motor Driver DRV8825(4buah)	 <p>Gambar 3-9. Motor Driver DRV8825</p>

3.3 Metode Perancangan

Pada perancangan menggunakan metode *Agile Software Methods and Development* dapat dilihat pada gambar 3-10 sebagai berikut:



Gambar 3-10 *Agile Software Methods and Development*

Secara umum alur penelitian/perancangan merupakan pola dari *Agile Software Methods and Development*.

3.3.1 *Plan*

Pada proses ini adalah sebagai acuan dalam proses perancangan perangkat lunak yang berisikan kebutuhan atau fitur yang akan dimuat dalam perangkat lunak. Dalam proses ini secara keseluruhan meliputi dari rancangan perangkat keras dan perangkat lunak.

3.3.2 *Define*

Proses ini merupakan detail proses rancangan dari proses *plan*, dimana dalam tahapan ini berisi arah gerak untuk memenuhi kebutuhan atau fitur dalam perangkat lunak. Sehingga dalam proses perancangan perangkat lunak dapat dikerjakan secara runtut berdasarkan fungsi-fungsi yang akan diterapkan.

3.3.3 *Develop*

Tahap *develop* ini adalah proses penerapan dari tahapan sebelumnya menjadi sebuah kode pemrograman dengan parameter yang ditetapkan. Dalam proses *develop* ini berisikan *front end develop* dan *back end develop*. Pada tahap ini mengimplementasi dari *Agile Software Methods and Development* untuk melakukan pengembangan perangkat lunak.

3.3.4 *Test*

Proses ini adalah proses ujicoba yang dilakukan untuk mengetahui berfungsi atau tidaknya hasil dari proses-proses sebelumnya menggunakan *debug*

dan *diagnostics*. *Debug* dan *diagnostics* ini diimplementasikan dalam perangkat keras. Apabila ada fitur yang tidak berfungsi maka akan dilakukan proses *maintenance* untuk membenahi program agar dapat berfungsi sesuai yang diharapkan.

3.3.5 Maintenance

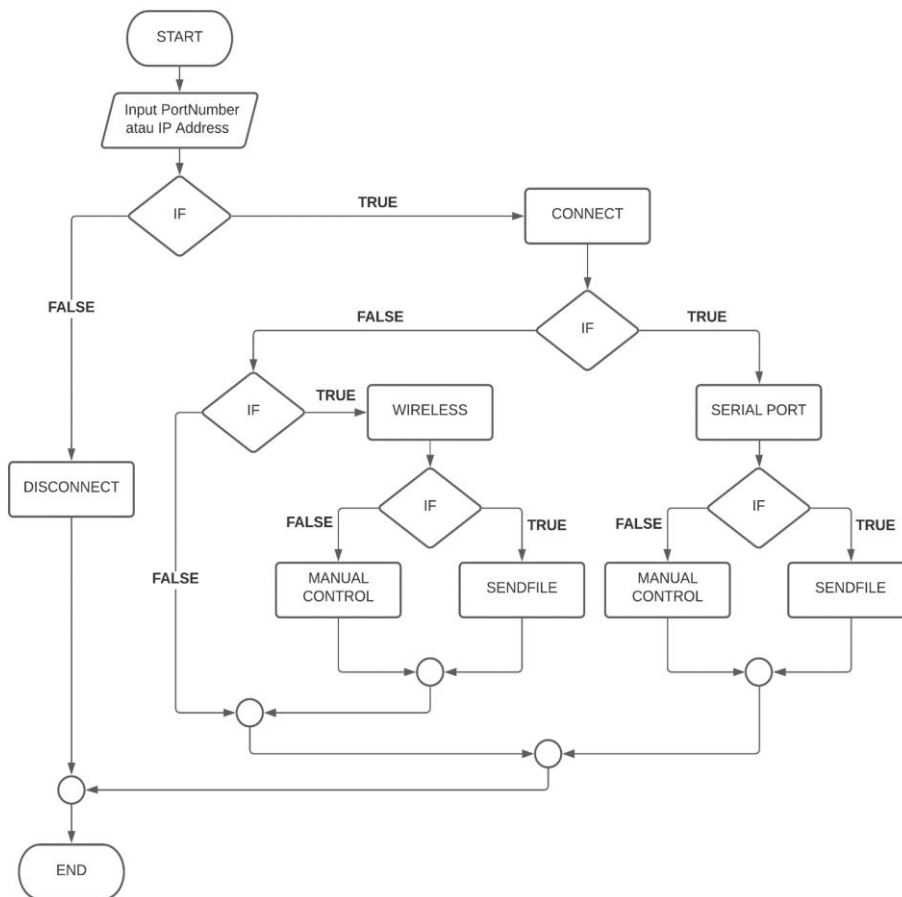
Tahap *maintenance* adalah tahap untuk membenahi program yang tidak dapat berjalan sesuai dari kebutuhan yang diinginkan. Tahap *maintenance* ini berdasarkan dari hasil proses *test*, apabila pada proses *test* sudah sesuai maka proses ini tidak dijalankan.

3.4 Perancangan

3.4.1 Perancangan Perangkat Lunak

3.4.1.1 Plan

Pada Tahap ini dilakukan pembuatan rancangan konsep dasar fitur yang akan dimuat dalam rancangan perangkat lunak. Untuk mempermudah dilakukan pembuatan konsep dasar dengan menggunakan diagram alir yang dapat dilihat pada gambar 3-11 sebagai berikut:



Gambar 3-11. *Plan* diagram konsep dasar sistem

Dari gambar 3-11 diatas, dalam proses perancangan perangkat lunak pengirim GCode pada mesin mini *Computer Numerical Control* berbasis GRBL memerlukan kebutuhan sistem sebagai berikut:

1. Dapat menyambung dan memutus koneksi/jaringan antara perangkat lunak dengan Wemos D1 R32 atau ESPDuino32.
2. Dapat membangun koneksi dengan menggunakan koneksi *serial port* antara perangkat lunak dengan Wemos D1 R32 atau ESPDuino32.
3. Dapat membangun koneksi dengan menggunakan koneksi *wireless* antara perangkat lunak dengan Wemos D1 R32 atau ESPDuino32.
4. Dapat melakukan pengiriman Gcode dalam bentuk *file* dari perangkat lunak ke Wemos D1 R32 atau ESPDuino32.
5. Dapat mengontrol secara manual dengan tombol dari perangkat lunak ke Wemos D1 R32 atau ESPDuino32.

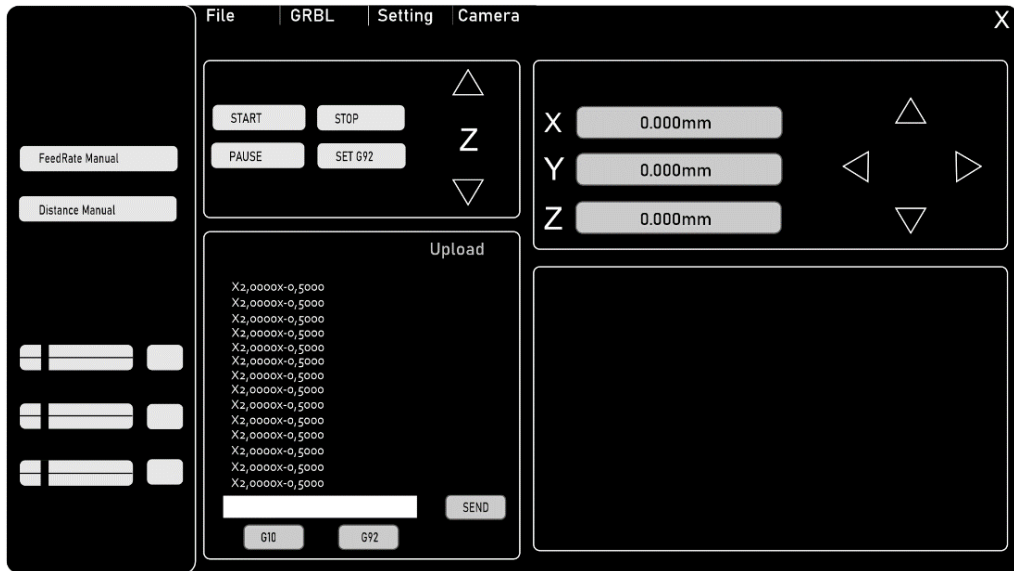
3.4.1.2 Define

Pada Tahap ini dilakukan pembuatan rancangan detail konsep dari konsep dasar untuk memetakan fitur yang akan dimuat dalam rancangan perangkat lunak.

Rancangan konsep secara detail yaitu sebagai berikut:

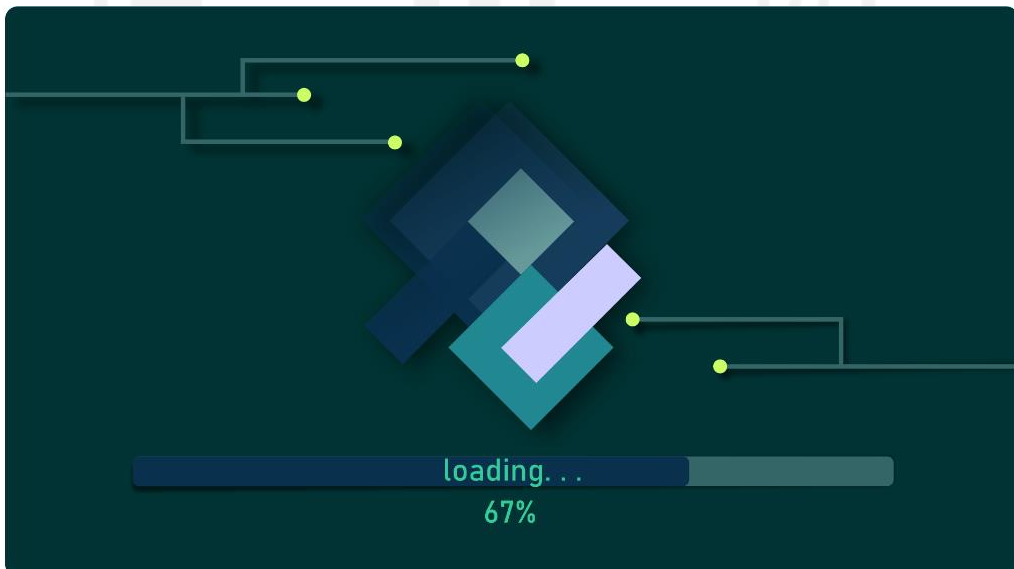
1. Pengguna dapat memilih tipe koneksi yaitu secara *serial port* dan *wireless*.
2. Pengguna dapat melakukan proses pemilihan koneksi secara *serial port* berdasarkan *port* yang tersedia.
3. Pengguna dapat melakukan pengaturan nilai *baudrate*.
4. Pengguna dapat melakukan proses pemilihan koneksi secara *wireless* dengan memberikan *Internet Protocol/IP Address* dan *Port*.
5. Pengguna dapat melakukan pengambilan *file* dari directory *Personal Computer* (PC) yang digunakan.
6. Pengguna dapat melihat meng-*upload file* pada perangkat lunak dan ditampilkan isi *file* tersebut.
7. Pengguna dapat mengirimkan *file* perbaris secara runtut dengan *control* untuk memulai, menjeda, dan menghentikan proses pemesinan di CNC dengan pola antrian.
8. Pengguna dapat melakukan *manual control* berdasarkan jarak(*distance*) dan gerak pemakanannya(*feedrate*) untuk menggerakkan sumbu X, Y, dan Z.
9. Pengguna dapat melakukan manual kontrol dengan mengetik GCode.
10. Pengguna dapat menentukan titik nol pada proses pemesinan di CNC.

Kemudian dilakukan pembuatan konsep rancangan desain antarmuka dan alur kerja perangkat lunak. Pembuatan konsep rancangan desain antarmuka dapat dilihat pada gambar 3-12 sebagai berikut:

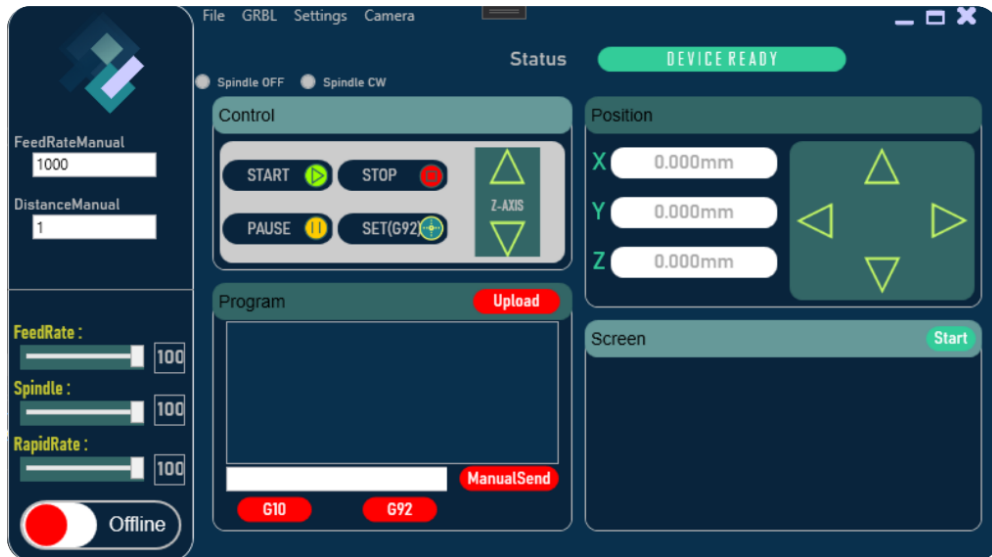


Gambar 3-12. Konsep rancangan desain antarmuka

Dengan adanya konsep rancangan desain antarmuka yang dilakukan dengan menggunakan *software* Corel Draw 2020 mempermudah dalam melakukan pengaturan format warna, ukuran, bentuk, dan peletakan. Dari konsep rancangan desain antarmuka pada gambar 3-12, kemudian dilakukan konsep rancangan akhir dengan menggunakan *software* Corel Draw 2020 dapat dilihat pada gambar 3-13 dan gambar 3-14 sebagai berikut:



Gambar 3-13. Konsep rancangan desain antarmuka *SplashScreen*



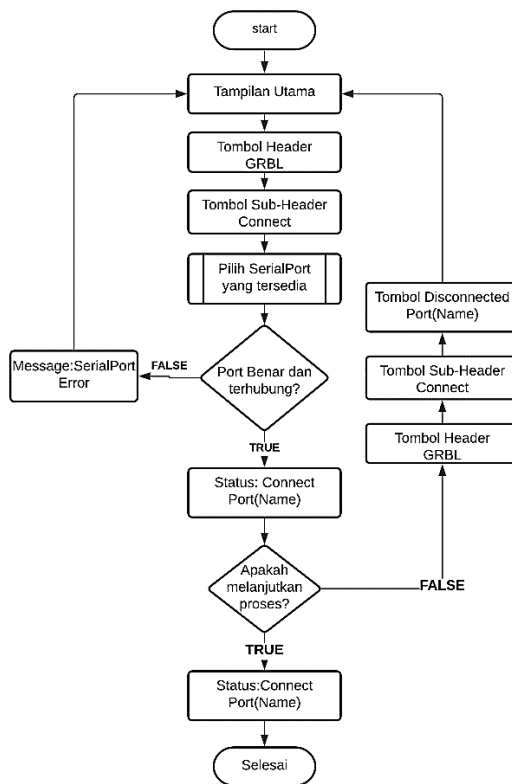
Gambar 3-14. Konsep rancangan desain antarmuka utama

3.4.1.3 Develop

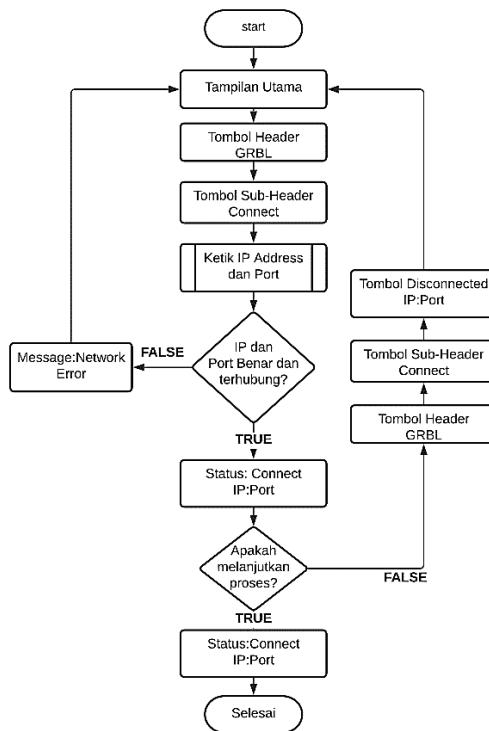
Pada Tahap ini dilakukan perancangan perangkat lunak, dengan menerapkan alur penelitian dari Metode *Agile Software Methods*. Tahap *develop* ini dibagi menjadi beberapa bagian yaitu:

1. *Front End Develop*
2. *Back End Develop*
 - a. *Back End Develop* koneksi melalui *Serial Port*
 - b. *Back End Develop* koneksi melalui *Wireless/Network*
 - c. *Back End Develop* pengiriman *file GCode*
 - d. *Back End Develop* *manual control*

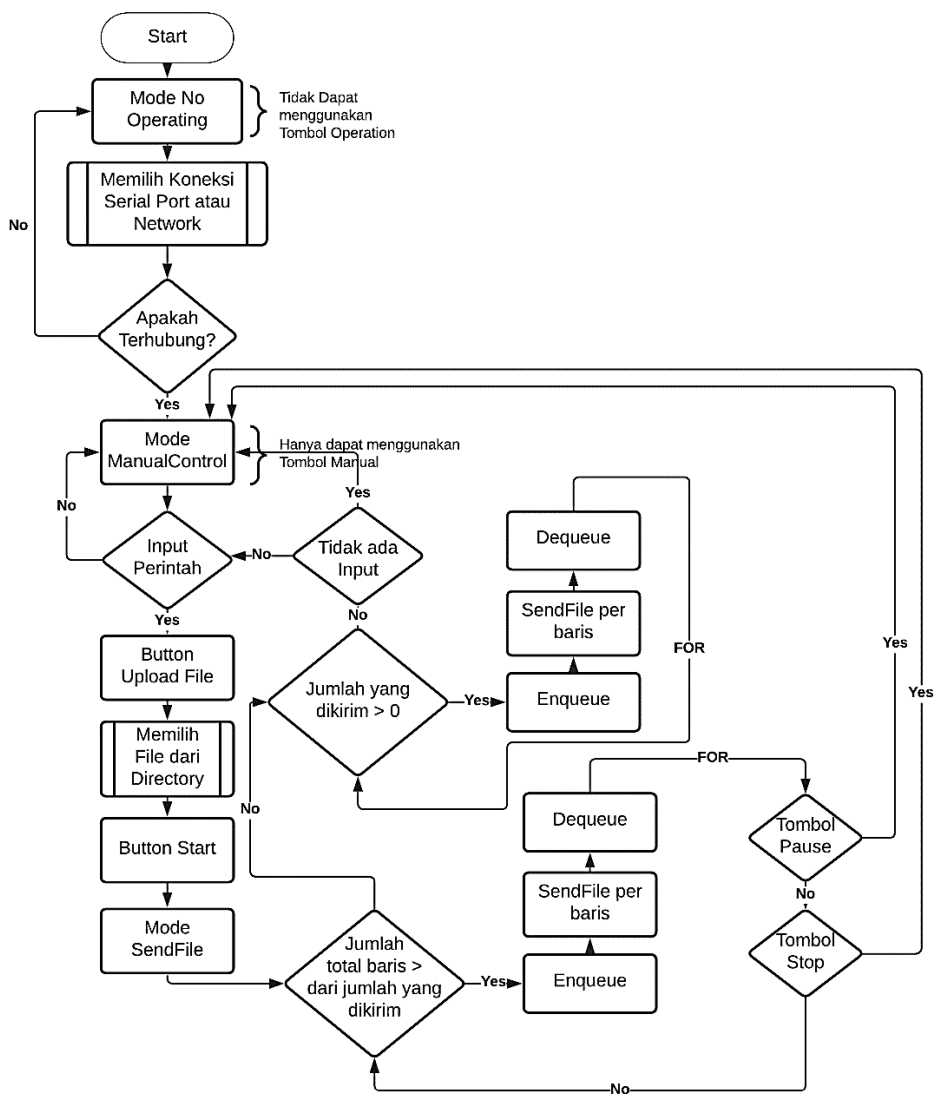
Hasil dan penjelasan pada tahap *develop* ini akan dimuat pada **Bab Hasil dan Pembahasan**. Konsep dalam pembuatan perangkat lunak pada bagian *Back End Develop* dapat dilihat pada gambar 3-15, gambar 3-16, gambar 3-17, dan gambar 3-18 sebagai berikut:



Gambar 3-15. Konsep alur kerja *Back End Develop* koneksi melalui *Serial Port*

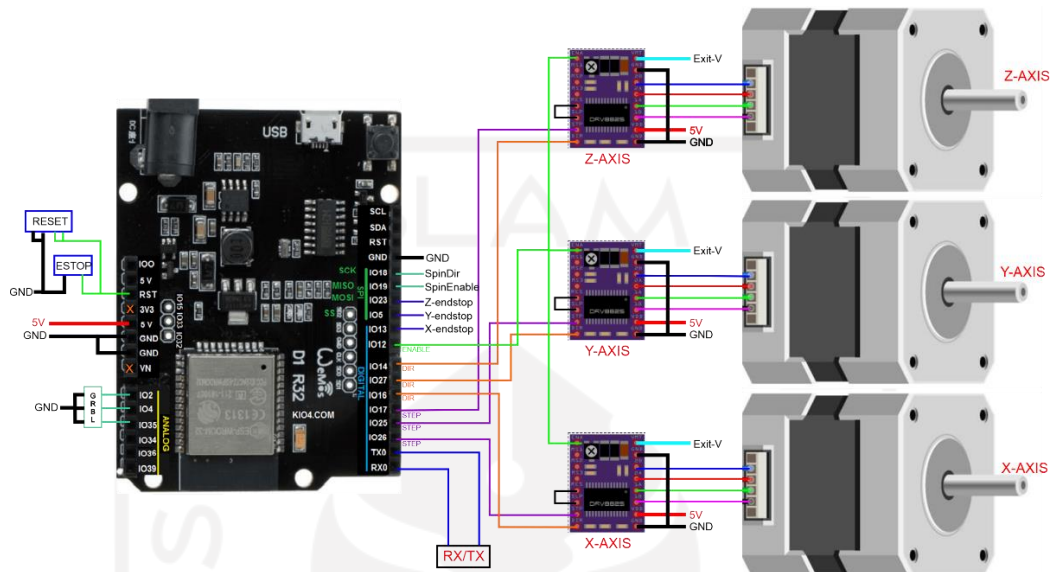


Gambar 3-16. Konsep alur kerja *Back End Develop* koneksi melalui *Wireless/Network*

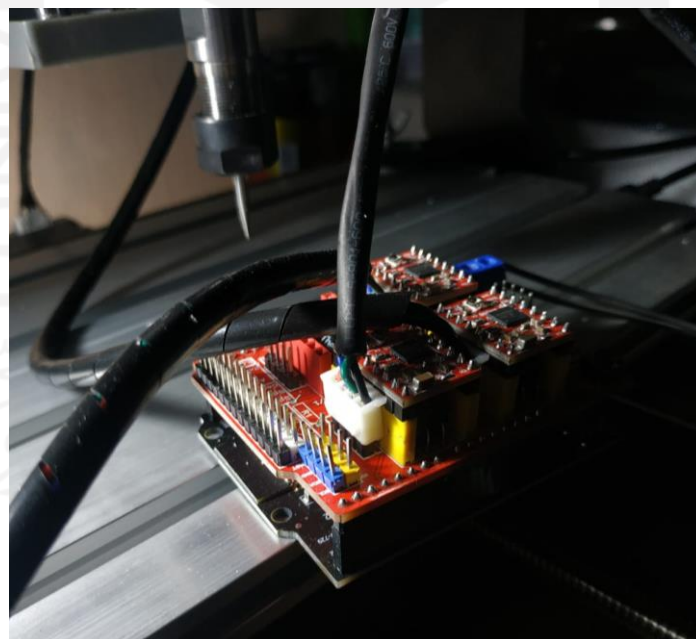


Gambar 3-17. Back End Develop pengiriman file GCode

Setelah pemetaan pada skema diagram CNC Shield dengan Arduino Uno dilakukan pembuatan skema diagram dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 dapat dilihat pada gambar 3-20 dan perancangan perangkat keras pada gambar 3-21 sebagai berikut:



Gambar 3-20. Skema diagram perancangan perangkat keras



Gambar 3-21. Perancangan perangkat keras

Proses perancangan perangkat keras ini menggunakan 2 buah adaptor yaitu adaptor 24volt dan adaptor 5volt dengan kabel usb yang dihubungkan ke lubang USB pada Wemos D1 R32 atau ESPDuino32. Proses perancangan sistem

kontrol pada *microstep* pada driver yang terpasang pada CNC Shield V3 menggunakan jumper pin dengan pengaturan pada tabel 3-3 sebagai berikut:

Tabel 3-3. Pengaturan microstep driver pada CNC Shield V3

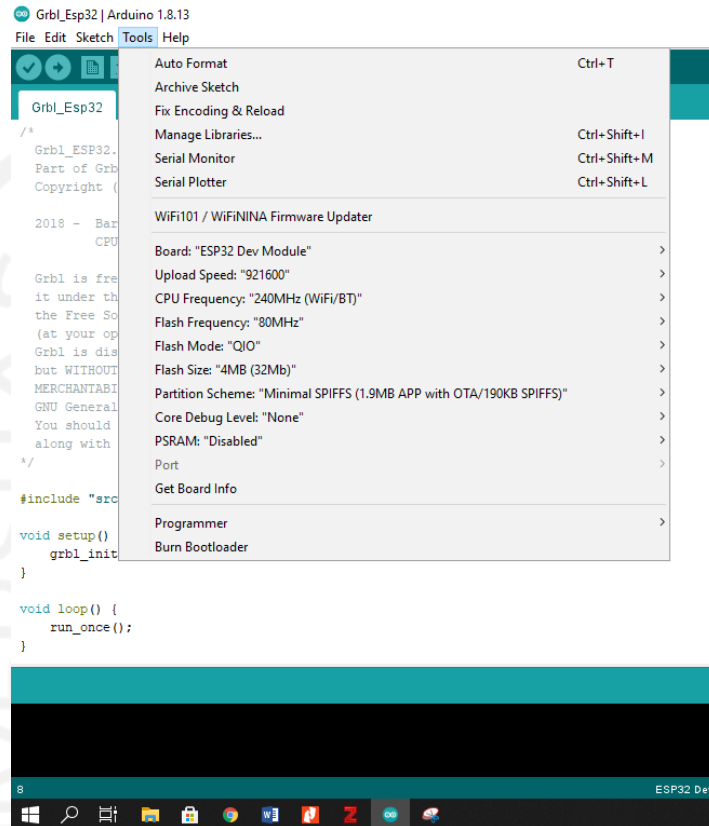
Mode0 (M0)	Mode1 (M1)	Mode2 (M2)	<i>Microstep Resolution</i>
Low	Low	Low	Full Step
High	Low	Low	Half Step
Low	High	Low	1/4 Step
High	High	Low	1/8 Step
Low	Low	High	1/16 Step
High	Low	High	1/32 Step
Low	High	High	1/32 Step
High	High	High	1/32 Step

Perancangan sistem kontrol ini memerlukan *upload* firmware GRBL_ESP32 dengan menggunakan *Software* Arduino IDE 1.8.13 dan pembuatan tambahan *library* pada *firmware* GRBL_ESP32 untuk agar dapat digunakan pada mikrokontroler Wemos D1 R32 atau ESPDuino32 dapat dilihat pada gambar 3-22 sebagai berikut:

```
#define MACHINE_NAME "MACHINE_ESPDUINO_32" // #define SPINDLE_TYPE SpindleType::PWM
#define X_STEP_PIN GPIO_NUM_26 #define SPINDLE_OUTPUT_PIN GPIO_NUM_23
#define X_DIRECTION_PIN GPIO_NUM_16 #define SPINDLE_ENABLE_PIN GPIO_NUM_18
#define Y_STEP_PIN GPIO_NUM_25 #define COOLANT_FLOOD_PIN GPIO_NUM_34
#define Y_DIRECTION_PIN GPIO_NUM_27 #define COOLANT_MIST_PIN GPIO_NUM_36
#define Z_STEP_PIN GPIO_NUM_17 #define X_LIMIT_PIN GPIO_NUM_13
#define Z_DIRECTION_PIN GPIO_NUM_14 // #define Z_LIMIT_PIN GPIO_NUM_23
#define A_STEP_PIN GPIO_NUM_15 #define LIMIT_MASK B111
#define A_DIRECTION_PIN GPIO_NUM_32 // #define PROBE_PIN GPIO_NUM_18
#define STEPPERS_DISABLE_PIN GPIO_NUM_12 #define CONTROL_RESET_PIN GPIO_NUM_2
#define USE_SPINDLE_RELAY #define CONTROL_FEED_HOLD_PIN GPIO_NUM_4
#define USE_SPINDLE_RELAY #define CONTROL_CYCLE_START_PIN GPIO_NUM_35
#ifdef USE_SPINDLE_RELAY #define USE_SPINDLE_RELAY
#define SPINDLE_TYPE SpindleType::RELAY #define DEFAULT_SPINDLE_RPM_MAX 1.0
#define SPINDLE_OUTPUT_PIN GPIO_NUM_19 #else
#else #define DEFAULT_SPINDLE_RPM_MAX 1000.0
#define SPINDLE_TYPE SpindleType::PWM #endif
#define SPINDLE_OUTPUT_PIN GPIO_NUM_23 #define DEFAULT_SPINDLE_RPM_MIN 0.0 // rpm
```

Gambar 3-22. Library tambahan pada GRBL_ESP32 untuk Wemos D1 R32/ESPDuino32

Proses *upload firmware* GRBL_ESP32 dengan menggunakan *Software* Arduino IDE 1.8.13 memerlukan *library TMCStepper by teemuatlut* yang dapat ditambahkan melalui *header menu sketch* pada *Software* Arduino IDE 1.8.13 konfigurasi *upload firmware* dapat dilihat pada gambar 3-23 sebagai berikut:



Gambar 3-23 Konfigurasi *upload firmware* GRBL_ESP32

BAB 4

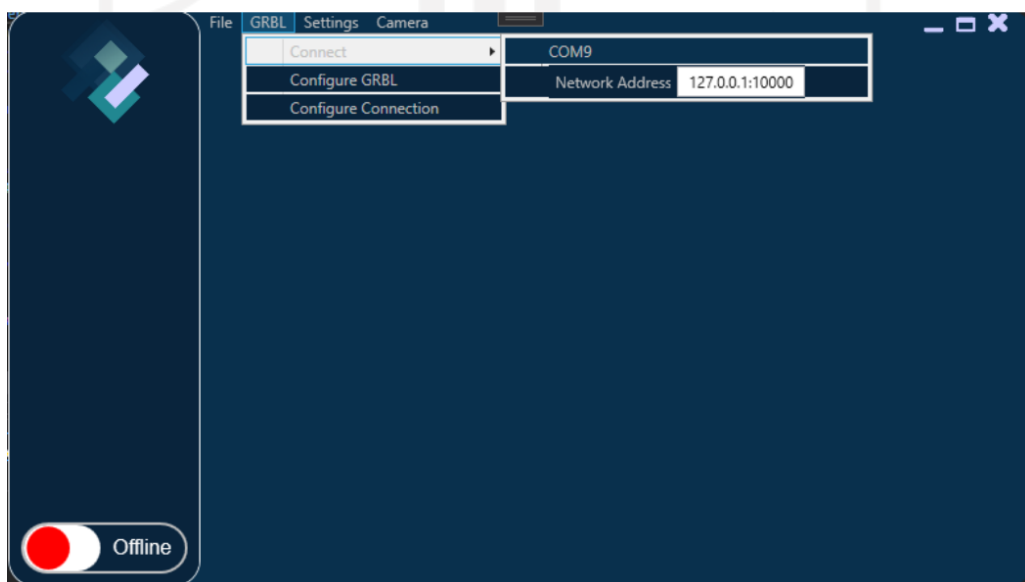
HASIL DAN PEMBAHASAN

4.1 Hasil Perancangan

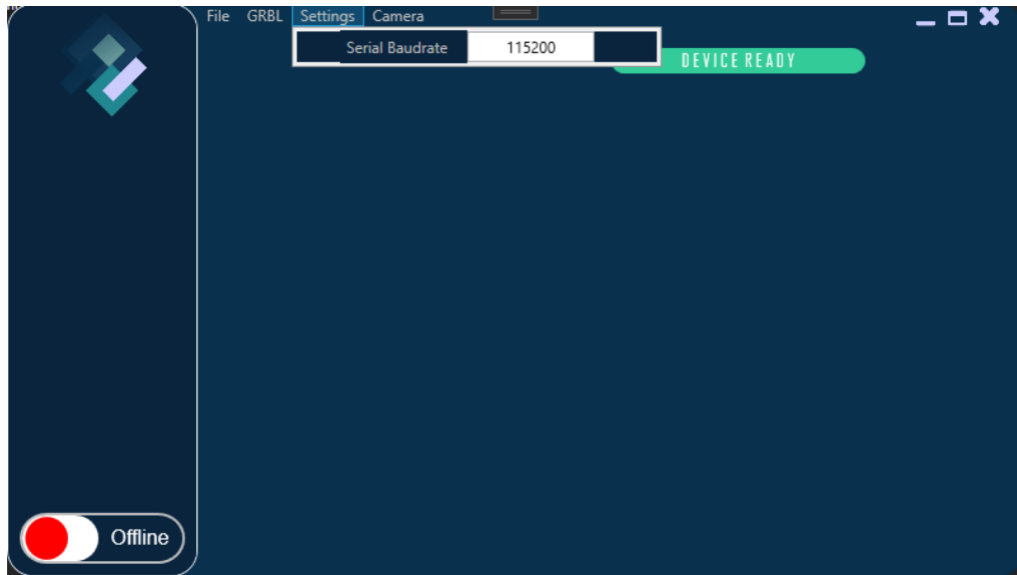
Perancangan perangkat lunak dimulai sesuai dengan alur konsep yang telah dibuat yaitu *Front End Develop* (antarmuka) yang dilanjutkan dengan *Back End Develop* (sistem).

4.1.1 Hasil *Front End Develop* (Antarmuka)

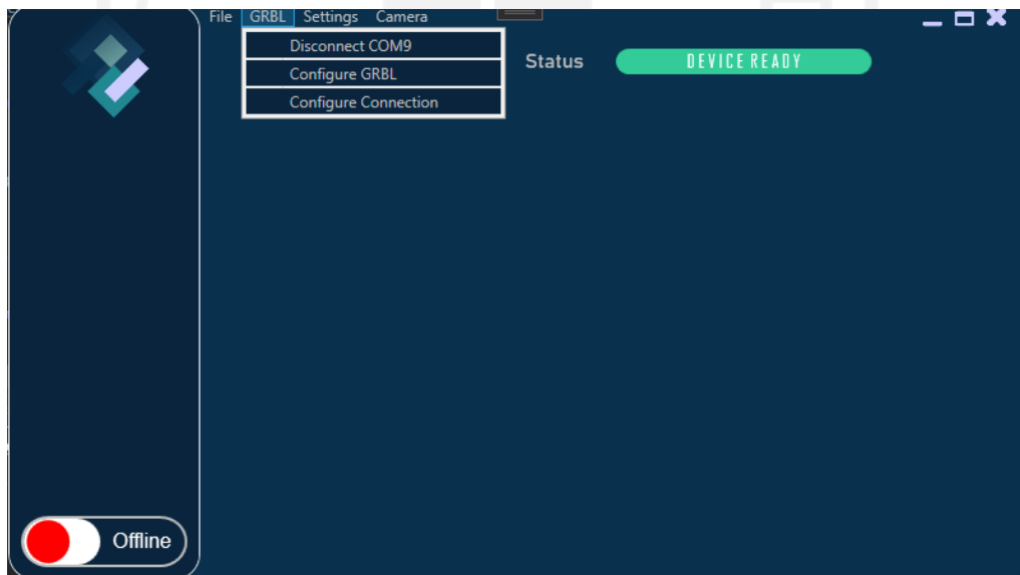
Pada tahap *Front End Develop* dengan menggunakan *Extensible Application Markup Language* (XAML) pada Software Visual Studio 2019 WPF App (.Net Framework) sesuai dengan kebutuhan. Proses perancangan antarmuka koneksi yang berfungsi sebagai antarmuka untuk berinteraksi dengan Wemos D1 R32 sehingga dapat terjadi perubahan tampilan saat adanya pertukaran data atau terhubungnya antara perangkat lunak dengan Wemos D1 R32. Hasil perancangan antarmuka koneksi secara *serial port* dan *wireless* dapat dilihat pada gambar 4-1, gambar 4-2, dan gambar 4-3 sebagai berikut:



Gambar 4-1. Hasil perancangan antarmuka pengguna koneksi

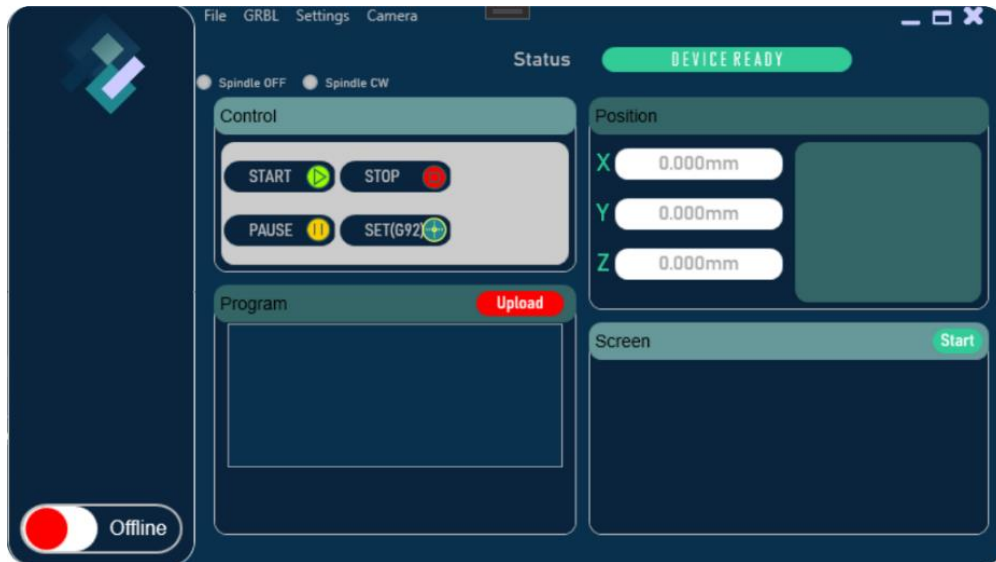


Gambar 4-2. Hasil perancangan antarmuka pengguna koneksi (*Baudrate*)

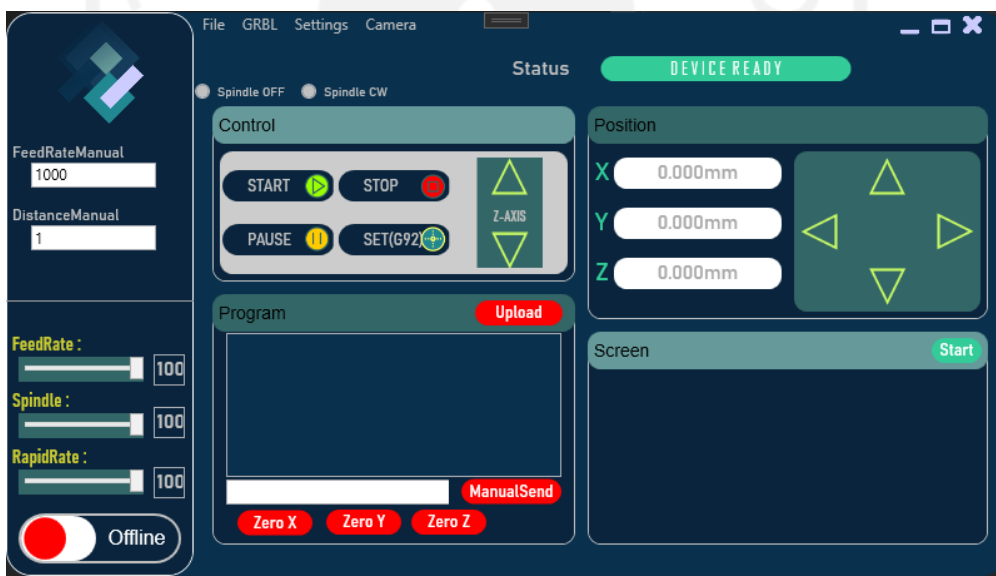


Gambar 4-3. Hasil perancangan antarmuka kondisi terhubung

Hasil perancangan antarmuka koneksi menggunakan *UI element MenuItem Header* sehingga dalam menghubungkan atau memutus koneksi tidak memakan banyak ruang karena dengan menggunakan *UI element MenuItem Header* dalam proses ini terdapat pada bagian atas tampilan. Setelah perancangan antarmuka koneksi dilakukan perancangan antarmuka pengiriman *file GCode* dan perancangan antarmuka *manual control* dapat dilihat pada gambar 4-4 dan 4-5 sebagai berikut:



Gambar 4-4. Hasil perancangan antarmuka pengiriman *file* GCode



Gambar 4-5. Hasil perancangan antarmuka pengguna akhir

Pada proses *Front End Develop* (antarmuka) terdapat berbagai *element* yang digunakan untuk menjalankan mekanisme gerak atau proses. Berikut ini adalah *UI element* yang digunakan beserta fungsinya (Tabel 4-1)

Tabel 4-1. *UI Element* dan fungsinya

No	Jenis <i>Toolbox</i>	Nama	Fungsi
1	Menu Item(Header)	File	Untuk melakukan proses mengambil dan menyimpan <i>file</i> di <i>directory Personal Computer (PC)</i>

No	Jenis <i>Toolbox</i>	Nama	Fungsi
2	Menu Item(Header)	GRBL	Untuk melakukan proses menghubungkan konektivitas secara <i>serial port/wireless</i>
3	Menu Item(Header)	Setting	Untuk melakukan penyetelan nilai <i>baudrate</i>
4	Radio Button	Spindle OFF	Untuk menonaktifkan <i>spindle</i>
5	Radio Button	Spindle CW	Untuk mengaktifkan <i>spindle</i> dengan putaran CW
6	Button	Start	Untuk memulai proses pengiriman GCode dalam <i>file</i>
7	Button	Stop	Untuk menghentikan proses pengiriman GCode dalam <i>file</i>
8	Button	Pause	Untuk menjeda proses pengiriman GCode dalam <i>file</i>
9	Button	Set G92	Untuk mendefinisikan titik nol mesin CNC
10	Textbox	Feedrate Manual	Untuk mengatur kecepatan pemakanan saat melakukan perpindahan secara manual
11	Textbox	Distance Manual	Untuk mengatur jarak pemakanan saat melakukan perpindahan secara manual
12	Button	Upload	Untuk mengunggah file GCode dalam aplikasi dari <i>directory Personal Computer (PC)</i>
13	Button	ManualSend	Untuk memulai proses pengiriman GCode dalam Textbox manual
14	Button	ZeroX	Untuk mendefinisikan titik nol sumbu X

No	Jenis <i>Toolbox</i>	Nama	Fungsi
15	Button	Zero Y	Untuk mendefinisikan titik nol sumbu Y
16	Button	Zero Z	Untuk mendefinisikan titik nol sumbu Z
17	Textbox	Manual	Untuk meng- <i>input</i> GCode yang dikirim secara manual
18	Button	Minimize	Untuk menyembunyikan tampilan
19	Button	Maximize	Untuk memperluas tampilan
20	Button	Close	Untuk keluar dari perangkat lunak
21	Button	X+	Untuk melakukan perpindahan posisi positif sumbu X dalam manual kontrol
22	Button	X-	Untuk melakukan perpindahan posisi negatif sumbu X dalam manual kontrol
23	Button	Y+	Untuk melakukan perpindahan posisi positif sumbu Y dalam manual kontrol
24	Button	Y-	Untuk melakukan perpindahan posisi negatif sumbu Y dalam manual kontrol
25	Button	Z+	Untuk melakukan perpindahan posisi positif sumbu Z dalam manual kontrol
26	Button	Z-	Untuk melakukan perpindahan posisi negatif sumbu Z dalam manual kontrol
27	ListView	File View	Untuk menampilkan isi <i>file</i> di perangkat lunak

4.1.2 Hasil *Back End Develop* (Sistem)

Pada tahap *Back End Develop* (sistem) menggunakan bahasa C# atau CSharp pada Software Visual Studio 2019 WPF App (.Net Framework). *Back End Develop* (sistem) ini berfungsi sebagai interaksi pengguna ketika antarmuka diberi sebuah aksi (klik, ketik, atau pilih) terjadi pemrosesan data atau akan adanya sistem yang berjalan.

Pertukaran data secara serial port memerlukan nilai *baudrate* untuk menentukan kecepatan dalam pengiriman sehingga tidak adanya data yang tertunda atau ditangguhkan. Sedangkan dalam pertukaran data secara *wireless* memerlukan *IP Address* atau alamat *Internet Protocol* dan *port number*.

Back End Develop (sistem) untuk menghubungkan dan dapat berinteraksi antara perangkat lunak secara *serial port* maupun *wireless* dengan Wemos D1 R32 atau ESPDuino32 dapat dilihat pada gambar 4-6 dan gambar 4-7 sebagai berikut:

```
public bool ConnectNetwork(string address)
{
    Disconnect();
    PortName = address;

    try
    {
        TcpClient tcpclient = new TcpClient();
        tcpclient.Connect(UtilityTCP.ParseIPEndPoint(address));

        ConnectionStream = tcpclient.GetStream();
        IsConnected = true;

        Connect();
        return true;
    }
    catch
    {
        return false;
    }
}

1 reference
public bool ConnectSerial(string portName)
{
    Disconnect();
    PortName = portName;

    try
    {
        SerialPort port = new SerialPort(portName, Settings.Default.SerialBaudRate);
        port.Open();

        ConnectionStream = port.BaseStream;
        IsConnected = true;
        Connect();

        return true;
    }
    catch
    {
        return false;
    }
}
```

Gambar 4-6. Hasil perancangan sistem koneksi *serial port* dan *wireless*


```

#region Connectivity
3 references
public void Disconnect()
{
    IsConnected = false;

    if (ConnectionStream != null)
    {
        ConnectionStream.Close();
        ConnectionStream = null;
    }

    Mode = OperatingMode.NoOperating;
    RaiseEvent(ConnectionStateChanged);
}
2 references
public void Connect()
{
    if (!IsConnected)
    {
        return;
    }
    ToSend.Clear();
    ToSendPriority.Clear();
    Sender.Clear();

    Mode = OperatingMode.ManualControl;
    RaiseEvent(ConnectionStateChanged);

    WorkerThread = new Thread(Running);
    WorkerThread.Priority = ThreadPriority.AboveNormal;
    WorkerThread.Start();
}

```

Gambar 4-7. Hasil perancangan sistem kondisi *connect* dan *disconnect*

Back End Develop (sistem) untuk membangun sebuah proses agar tidak adanya fungsi yang bertabrakan adalah dengan mendefinisikan tiga mode yaitu *NoOperating*, *ManualControl*, dan *SendFile*. Mode *NoOperating* adalah mode yang aktif ketika perangkat lunak dalam keadaan *disconnect* atau tidak terhubung. Mode *ManualControl* yaitu mode yang aktif ketika perangkat lunak dalam keadaan *connect* atau terhubung, sehingga dapat melakukan proses penentuan titik nol secara manual. Mode *SendFile* adalah mode yang aktif saat terjadinya pertukaran data dari perangkat lunak ke Wemos D1 R32 atau ESPDuino32. Kemudian saat perangkat lunak dalam keadaan *connect* atau terhubung perlu adanya pengiriman perintah untuk melakukan reset ke mode penerjemah GCode.

Pada proses pertukaran data dalam jumlah lebih dari 1 baris perlu dilakukan antrian (*Enqueue*). Data yang dikirim ke Wemos D1 R32 atau ESPDuino32 harus dalam keadaan tanpa spasi dan dipisah setiap barisnya dengan enter. *Back End Develop* (sistem) yang didefinisikan kedalam mode beserta fungsinya dapat dilihat pada gambar 4-8, gambar 4-9, dan gambar 4-10 sebagai berikut:

```

namespace CNC_Launcher.Core.Communication
{
    19 references
    class ModeControl
    {
        33 references
        public enum OperatingMode
        {
            ManualControl,
            SendFile,
            NoOperating
        }
        Action
        Field RunningFile
        Field Connectivity

        #region queue
        Queue Sender = Queue.Synchronized(new Queue()); //Contruction untuk melakukan antrian yang ak
        Queue ToSend = Queue.Synchronized(new Queue()); //Contruction untuk melakukan antrian yang ak
        Queue ToSendPriority = Queue.Synchronized(new Queue()); //Contruction untuk melakukan antrian
        #endregion queue

        #region Running
        1 reference
        public void Running()
        {
            try
            {
                StreamReader ConnectionReader = new StreamReader(ConnectionStream, Encoding.ASCII);
                StreamWriter ConnectionWriter = new StreamWriter(ConnectionStream, Encoding.ASCII);
                //Func untuk mengatur buffer size yang dikelola aplikasi dan untuk mendeskripsikan

                int ControllerBufferSize = Settings.Default.controllerBufferSize;
                BufferState = 0;

                ConnectionWriter.Write("\n$G\n");
                ConnectionWriter.Write("\n$#\n");
                ConnectionWriter.Flush();
            }
        }
    }
}

```

Gambar 4-8. Hasil perancangan sistem definisi mode

```

while (true)
{
    Task<string> lineTask = ConnectionReader.ReadLineAsync();

    while (!lineTask.IsCompleted)
    {
        if (!IsConnected)
        {
            return;
        }
        while (ToSendPriority.Count > 0)
        {
            ConnectionWriter.Write((char)ToSendPriority.Dequeue());
            ConnectionWriter.Flush();
        }
        if (Mode == OperatingMode.SendFile) //Func untuk mengatur format pengiriman gcode file per-line dengan posisi
        {
            if (File.Count > FilePosition && (File[FilePosition].Length + 1) < (ControllerBufferSize - BufferState))
            {
                string linesending = File[FilePosition].Replace(" ", ""); //Menghilangkan spasi line gcode yang akan
                ConnectionWriter.Write(linesending); //line yang siap kirim (sudah tidak ada spasi)
                ConnectionWriter.Write('\n'); //enter
                ConnectionWriter.Flush(); //untuk menghapus buffer data yang dikirim perline

                RaiseEvent(LineSent, linesending);

                BufferState += linesending.Length + 1; //jumlah bufferstate sama dengan jumlah kata line gcode siap k
                Sender.Enqueue(linesending); //mendaftarkan atau menambahkan antrian

                //Func untuk mnegatur proses pengiriman gcode secara manual dengan memilih mulainya line
                if (PauseLines[FilePosition] && Settings.Default.PauseFileOnMold) //line pada file yang di pause bert
                {
                    Mode = OperatingMode.ManualControl;
                }

                if (++FilePosition >= File.Count) //pre increment dari file line tertentu lebih dari sama dengan jum
                {
                    Mode = OperatingMode.ManualControl;
                }
            }
        }
    }
}

```

Gambar 4-9. Hasil perancangan sistem fungsi 1 dalam pertukaran data dari sebuah file

```

if (PauseLines[FilePosition] && Settings.Default.PauseFileOnHold) //line pada file yang di pause b
{
    Mode = OperatingMode.ManualControl;
}

if (++FilePosition >= File.Count) //pre increment dari file line tertentu lebih dari sama dengan j
{
    Mode = OperatingMode.ManualControl;
}
}

//Func untuk mendeskripsikan apabila fungsi if diatas tidak terpenuhi
//Func ini untuk melakukan proses berikutnya (progres line file)
else if (ToSend.Count > 0 && (((string)ToSend.Peek()).Length + 1) < (ControllerBufferSize - BufferState))
{
    string send_line = ((string)ToSend.Dequeue()).Replace(" ", ""); //line gcode yang dikirim sama dengan
    ConnectionWriter.Write(send_line); //line yang siap kirim yang sudah dihilangkan spasinya
    ConnectionWriter.Write('\n'); //enter
    ConnectionWriter.Flush(); //untuk menghapus buffer data yang dikirim perline
    RaiseEvent(LineSent, send_line);

    BufferState += send_line.Length + 1; //jumlah bufferstate sama dengan jumlah kata line gcode siap kirim

    Sender.Enqueue(send_line); //mendaftarkan atau menambahkan antrian
}

TimeSpan WaitTime = TimeSpan.FromMilliseconds(0.2);
Thread.Sleep(WaitTime); //lama waktu background sleep sebesar nilai jeda
}

string line = lineTask.Result; //string baru "line" adalah hasil line yang berjalan
}
}
catch
{
}
} //Method untuk proses berlangsungnya aplikasi
#endregion Running

```

Gambar 4-10. Hasil perancangan sistem fungsi 2 dalam pertukaran data dari sebuah *file*

Pada mode *SendFile* terdapat 2 fungsi didalamnya yaitu kondisi saat jumlah baris dalam sebuah *file* lebih besar daripada jumlah baris yang saat itu dikirim dan kondisi saat jumlah baris yang dikirim lebih besar daripada nol. Dalam pengiriman *file* perlunya penyesuaian jumlah karakter yang dikirim dengan ukuran dari *Buffer Size* agar dalam proses pengiriman dapat berlangsung lancar. *Buffer Size* adalah ukuran dalam proses pengiriman data dalam satuan waktu yang berfungsi untuk mengatur besaran pengiriman data agar tidak terjadi *delay* dan kinerja komputer dapat optimal.

Proses pertukaran data saat melakukan pengiriman baris GCode harus dalam kondisi *connect* atau terhubung sehingga dapat diantrikan untuk dikirim ke Wemos D1 R32 atau ESPDuino32. Hal ini dapat dilihat pada gambar 4-11 sebagai berikut:

```

public void SendLine(string line)
{
    if (!IsConnected)
    {
        return;
    }

    ToSend.Enqueue(line);
}

```

Gambar 4-11. Hasil perancangan sistem SendLine

Proses *Back End Develop* (sistem) dalam melakukan perpindahan secara manual dengan tombol X+, X-, Y+, Y-, Z+, dan Z- dengan parameter *rapid move* (kecepatan gerak perpindahan) dan *distance* (jarak perpindahan) dapat dilihat pada gambar 4-12 sebagai berikut:

```

#region JogManual
1 reference
private void Xplus_buttonclick(object sender, RoutedEventArgs e)
{
    if (!modeControl.IsConnected)
        return;
    if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
        return;

    if (TextBoxDistanceManual != null && TextBoxFeedRateManual.Text != null)
    {
        modeControl.SendLine(string.Format(DecimalOutputFormat, "G91F{0:0.##}X{1:0.###}",
            TextBoxFeedRateManual.Text, TextBoxDistanceManual.Text));
        Console.WriteLine("XplusButtonOK");
    }
}

1 reference
private void Xmin_buttonclick(object sender, RoutedEventArgs e)

1 reference
private void Ymin_buttonclick(object sender, RoutedEventArgs e)

1 reference
private void Yplus_buttonclick(object sender, RoutedEventArgs e)

1 reference
private void Zplus_buttonclick(object sender, RoutedEventArgs e)

1 reference
private void Zmin_buttonclick(object sender, RoutedEventArgs e)
#endregion JogManual

```

Gambar 4-12. Hasil perancangan sistem perpindahan manual

Proses perpindahan secara manual perlu beberapa kriteria yaitu harus dalam kondisi *connect* atau terhubung dan dalam kondisi mode ManualControl. Apabila dalam melakukan perpindahan membutuhkan input dengan GCode dapat dilakukan dengan Textbox yang dapat diketik sesuai dengan kebutuhan dapat dilihat pada gambar 4-13 sebagai berikut:

```

#region ButtonManual
private List<string> ManualCommands = new List<string>();
1 reference
void ManualSend()
{
    string tosend;
    tosend = TextBoxManualSender.Text;

    modeControl.SendLine(tosend);

    ManualCommands.Insert(0, tosend);

    TextBoxManualSender.Text = "";
}
1 reference
private void ButtonManualSend_Click(object sender, RoutedEventArgs e)
{
    ManualSend();
    Console.WriteLine("ManualSender_received");
}

```

Gambar 4-13. Hasil perancangan sistem Textbox manual

4.2 Hasil Pengujian

4.2.1 Pengujian *Front End Develop* (Antarmuka)

Pengujian *Front End Develop* dilakukan untuk mengetahui kinerja *element* yang digunakan dalam tampilan perangkat lunak secara fungsionalitas. Pada pengujian ini menggunakan metode *BlackBox* yang difokuskan pada fungsionalitas perangkat lunak. Sebagai contoh pada gambar 4-14, gambar 4-15 dan gambar 4-16. Berikut hasil pengujian *Front End Develop* terdapat pada tabel 4-2

Tabel 4-2. Hasil pengujian *Front End Develop*

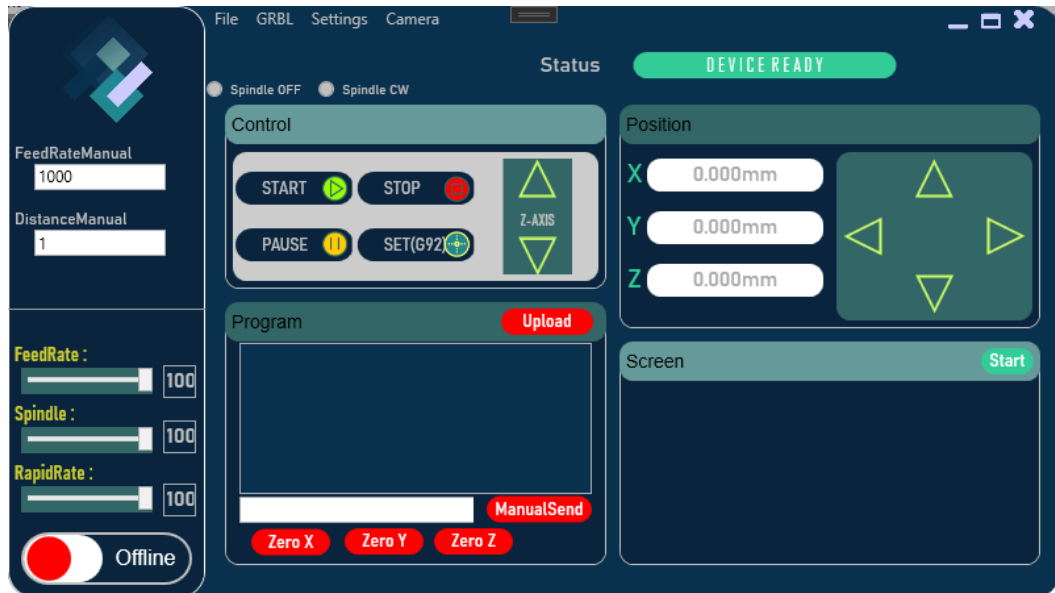
No	User Interface element	Fungsi yang diuji	Hasil yang didapat	Keterangan
1	Menu Item(Header) File	Pengguna dapat memunculkan sub menu dan terdapat perubahan warna tombol saat di klik	Dapat memunculkan submenu adanya perubahan warna saat di klik	Berhasil

No	User Interface element	Fungsi yang diuji	Hasil yang didapat	Keterangan
2	Menu Item(Header) GRBL	Pengguna dapat memunculkan sub menu, mengisikan <i>IP/Port</i> dan terdapat perubahan warna tombol saat di klik	Dapat memunculkan submenu dan terdapat perubahan warna saat di klik	Berhasil
3	Menu Item(Header) Setting	Pengguna dapat memunculkan sub menu, mengisikan <i>Serial Baudrate</i> dan terdapat perubahan warna tombol saat di klik	Dapat memunculkan submenu dan terdapat perubahan warna saat di klik	Berhasil
4	Radio Button Spindle OFF	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
5	Radio Button Spindle CW	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
6	Button Start	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
7	Button Stop	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil

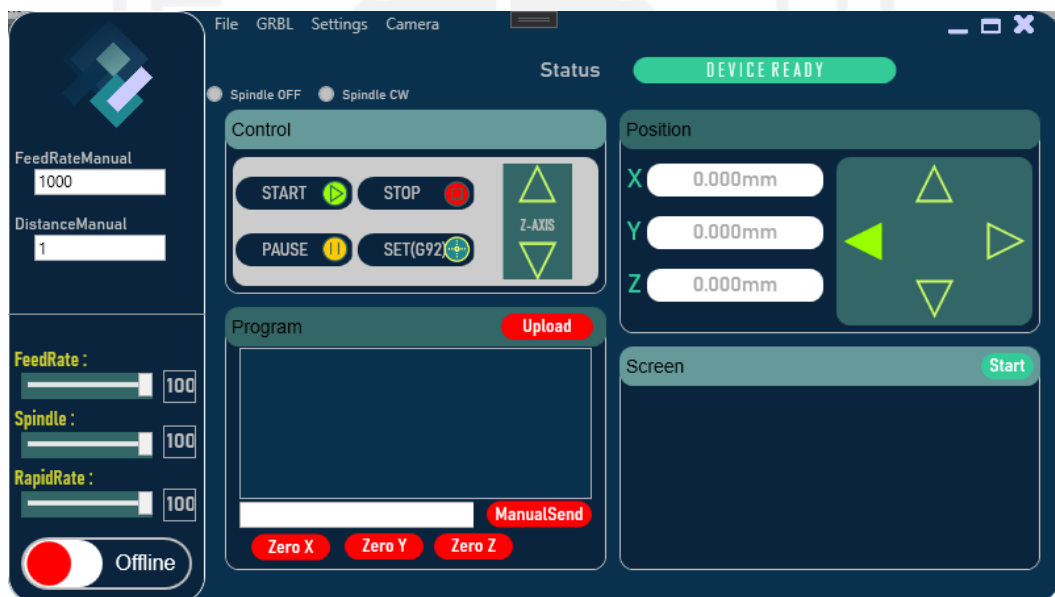
No	User Interface element	Fungsi yang diuji	Hasil yang didapat	Keterangan
8	Button Pause	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
9	Button Set92	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
10	TextBox Feedrate Manual	Pengguna dapat mengisikan nilai	Dapat mengisikan nilai dan dapat dilihat secara visual	Berhasil
11	TextBox Distance Manual	Pengguna dapat mengisikan nilai	Dapat mengisikan nilai dan dapat dilihat secara visual	Berhasil
12	Button Upload	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
13	Button Manual Send	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
14	Button ZeroX	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil

No	User Interface element	Fungsi yang diuji	Hasil yang didapat	Keterangan
15	Button ZeroY	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
16	Button ZeroZ	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
17	TextBox Manual	Pengguna dapat mengisikan nilai	Dapat mengisikan nilai dan dapat dilihat secara visual	Berhasil
18	Button Minimize	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
19	Button Maximize	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
20	Button Close	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
21	Button X+	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil

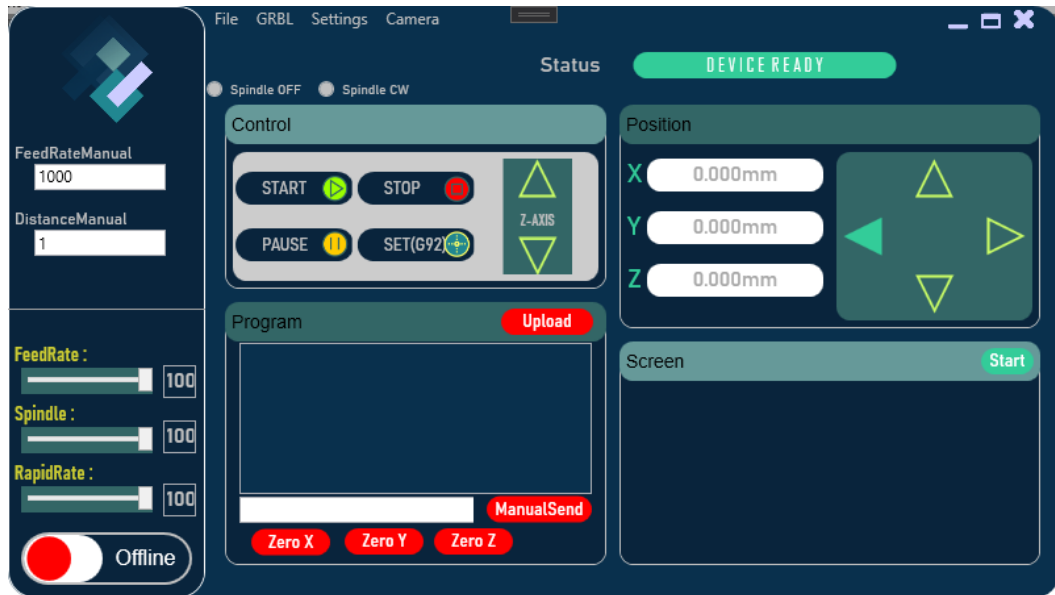
No	User Interface element	Fungsi yang diuji	Hasil yang didapat	Keterangan
22	Button X-	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
23	Button Y+	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
24	Button Y-	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
25	Button Z+	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
26	Button Z-	Pengguna dapat mengetahui adanya perubahan sebelum dan sesudah di klik	Adanya perubahan visual sebelum dan sesudah di klik	Berhasil
27	ListView File View	Pengguna dapat mengetahui/melihat isi <i>file</i> yang di unggah dalam perangkat lunak	Dapat mengetahui atau melihat isi <i>file</i> yang di unggah dalam perangkat lunak	Berhasil



Gambar 4-14. Tampilan ketika manual control X- sebelum diklik



Gambar 4-15. Tampilan ketika manual control X- kursor pada tombol



Gambar 4-16. Tampilan manual kontrol pada X- ketika diklik

4.2.2 Pengujian *Back End Develop* (Sistem)

Pengujian *Back End Develop* dilakukan untuk mengetahui sistem yang dibuat dapat berfungsi atau berjalan pada perangkat lunak. Pada pengujian ini menggunakan metode *BlackBox* yang difokuskan pada fungsionalitas perangkat lunak. Berikut hasil pengujian *Back End Develop* terdapat pada tabel 4-3

Tabel 4-3. Hasil pengujian *Back End Develop*

No	Fungsi yang diuji	Fungsi yang diharapkan	Hasil yang didapat	Keterangan
1	Fungsi Menghubungkan dan memutus jaringan secara <i>serial port</i>	Dapat mendeteksi <i>serial port</i> yang tersedia, menghubungkan dan memutus jaringan dengan mikrokontroler secara <i>serial port</i>	Sistem berhasil mendeteksi <i>serial port</i> yang tersedia, menghubungkan dan memutus jaringan dengan mikrokontroler secara <i>serial port</i>	Berhasil

No	Fungsi yang diuji	Fungsi yang diharapkan	Hasil yang didapat	Keterangan
2	Fungsi Menghubungkan dan memutuskan jaringan secara <i>wireless</i>	Dapat menghubungkan dan memutuskan jaringan secara <i>wireless</i> dari <i>IP/Port</i> yang dimasukan	Sistem berhasil menghubungkan dan memutuskan jaringan secara <i>wireless</i> dari <i>IP/Port</i> yang dimasukan	Berhasil
3	Fungsi memilih <i>file</i> dalam <i>directory</i> untuk diunggah ke perangkat lunak	Dapat memilih <i>file</i> dalam <i>directory</i> untuk diunggah <i>file</i> ke perangkat lunak	Sistem berhasil memilih <i>file</i> dalam <i>directory</i> untuk diunggah <i>file</i> ke perangkat lunak	Berhasil
4	Fungsi mengirimkan isi <i>file</i> ke mikrokontroler secara runtut per baris	Dapat mengirimkan isi <i>file</i> ke mikrokontroler secara runtut per baris	Sistem berhasil mengirimkan isi <i>file</i> ke mikrokontroler secara runtut per baris	Berhasil
5	Fungsi menghentikan pengiriman isi <i>file</i> ke mikrokontroler secara runtut per baris	Dapat menghentikan pengiriman isi <i>file</i> ke mikrokontroler secara runtut per baris	Sistem berhasil menghentikan pengiriman isi <i>file</i> ke mikrokontroler secara runtut per baris	Berhasil

No	Fungsi yang diuji	Fungsi yang diharapkan	Hasil yang didapat	Keterangan
6	Fungsi mengirim GCode secara manual	Dapat mengirim GCode secara manual dari GCode yang dimasukan oleh pengguna	Sistem berhasil mengirim GCode secara manual dari GCode yang dimasukan oleh pengguna	Berhasil
7	Fungsi menentukan titik nol	Dapat menentukan titik nol dengan menggerakkan tiap sumbu (sumbu x,y,z) sesuai dengan nilai gerakan pemakanan dan jaraknya	Sistem berhasil menggerakkan tiap sumbu berdasarkan nilai gerakan pemakanan dan jaraknya Hasil (G91F(VALUE)X(VALUE)) (G91F(VALUE)Y(VALUE)) (G91F(VALUE)Z(VALUE)) Dapat menentukan titik nol Hasil (G92X0Y0Z0) (G92X0) (G92Y0) (G92Z0)	Berhasil

Hasil respon sistem terkait menyambung dan memutus koneksi secara *serial port* maupun *wireless* dan melakukan pengiriman *file* GCode maupun manual kontrol dilihat pada gambar 4-17, gambar 4-18 dan gambar 4-19 sebagai berikut:

```
Output
Show output from: Debug
'CNC Launcher.exe' (CLR v4.0.3031
'CNC Launcher.exe' (CLR v4.0.3031
'CNC Launcher.exe' (CLR v4.0.3031
'CNC Launcher.exe' (CLR v4.0.3031
'CNC Launcher.exe' (CLR v4.0.3031
SerialConnect
The thread 0x22fc has exited with
The thread 0x3414 has exited with
Disconnected
NetworkConnect
The thread 0x350c has exited with
The thread 0x3ad8 has exited with
The thread 0x96c has exited with
Disconnected
```

Gambar 4-17. Menyambung dan memutus koneksi *serialport* atau *wireless*

```
Output
Show output from: Debug
SerialConnect
ManualSender_received
linesending
XminButtonOK
linesending
The thread 0x4ebc has exited with code 0 (0x0).
The thread 0x4324 has exited with code 0 (0x0).
The thread 0x3180 has exited with code 0 (0x0).
XplusButtonOK
linesending
YplusButtonOK
linesending
The thread 0x4bfc has exited with code 0 (0x0).
YminButtonOK
linesending
ZplusButtonOK
linesending
ZminButtonOK
linesending
The thread 0x3fe0 has exited with code 0 (0x0).
The thread 0x1e48 has exited with code 0 (0x0).
The thread 0x1f9c has exited with code 0 (0x0).
SetAllZero
linesending
The thread 0x35a4 has exited with code 0 (0x0).
FileStart
linesending
linesending
linesending
linesending
linesending
linesending
```

Gambar 4-18. Sistem bekerja dengan koneksi *serial port*

```

Output
Show output from: Debug
NetworkConnect
ManualSender_received
linesending
XminButtonOK
linesending
XplusButtonOK
linesending
YminButtonOK
linesending
YplusButtonOK
linesending
ZplusButtonOK
linesending
ZminButtonOK
linesending
The thread 0x18a8 has exited with code 0 (0x0).
G92 X0
linesending
G92 Y0
linesending
G92 Z0
linesending
FileStart
linesending
linesending
linesending
linesending
linesending
FileStop
linesending
linesending
The thread 0x2d8c has exited with code 0 (0x0).

```

Gambar 4-19. Sistem bekerja dengan koneksi *wireless*

4.2.3 Pengujian Pengiriman Data Terhadap Gerak CNC

Pengujian dilakukan untuk mengetahui kesesuaian pengiriman data dari perangkat lunak terhadap penerimaan data yang diproses oleh Wemos D1 R32 untuk mengoperasikan CNC GRBL 3 axis. Berikut hasil pengujian pengiriman data terhadap gerak CNC terdapat pada tabel 4-4, gambar 4-20, dan 4-21.

Bentuk *file* yang dikirim kedalam perangkat lunak dengan format *file* GCode. Terdapat tiga *file* yang dikirim yaitu *file* persegi, *file* segitiga dan *file* lingkaran, ketiga *file* tersebut berisikan kode GCode yang dapat dilihat pada tabel 4-5.

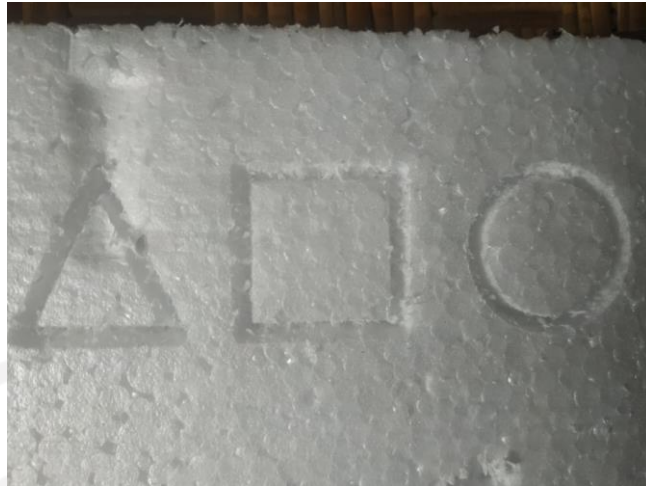
Tabel 4-4. Hasil pengujian pengiriman data terhadap gerak CNC

No	Mode Koneksi	Bentuk Uji	Hasil
1	<i>Serial Port</i>	Persegi, Panjang 50mm dan Lebar 50mm	Berhasil

No	Mode Koneksi	Bentuk Uji	Hasil
2	<i>Serial Port</i>	Segitiga, Alas 50mm, Tinggi 50mm	Berhasil
3	<i>Serial Port</i>	Lingkaran, Diameter 50mm	Berhasil
4	<i>Wireless</i>	Persegi, Panjang 50mm dan Lebar 50mm	Berhasil
5	<i>Wireless</i>	Segitiga, Alas 50mm, Tinggi 50mm	Berhasil
6	<i>Wireless</i>	Lingkaran, Diameter 50mm	Berhasil



Gambar 4-20. Hasil pengujian pengiriman data ke CNC secara *serial port*



Gambar 4-21. Hasil pengujian pengiriman data ke CNC secara *wireless*

Tabel 4-5. Gcode yang dikirim ke CNC

No	Variasi Bentuk	GCode
1	Persegi	F1000 G21 G91 G00 X0 Y0 Z0 G01 Z-10 G01 X0 Y50 G01 X50 Y0 G01 X0 Y-50 G01 X-50 Y0 G01 Z10 G00 X0 Y0 Z0
2	Segitiga	G21 G91 G00 X0 Y0 F1000 G01 Z-10 G01 X25 Y50 G01 X25 Y-50 G01 X-50 G01 Z10 G00 X0 Y0

No	Variasi Bentuk	GCode
3	Lingkaran	G21 G91 G00 X0 Y0 Z0 F1000 G01 Z-10 F1000 G02 X25 Y-25 R25 F500 G02 X-25 Y-25 R25 F500 G02 X-25 Y25 R25 F500 G02 X25 Y25 R25 F500 G01 Z10 F1000 G00 X0 Y0 Z0

4.2.4 Pengujian Usability

Pengujian usability dilakukan untuk mengetahui dan menilai kesesuaian perangkat lunak pada antarmuka dan keberhasilan sistem terhadap pengguna saat melakukan proses pemesinan dengan mesin mini CNC berbasis GRBL. Pengujian Usability mencakup 3 aspek yaitu: *Learnability* adalah tingkat kemudahan pengguna dalam memahami antarmuka perangkat lunak, *Effectiveness* adalah tingkat ketersediaan perangkat lunak terhadap pengguna dalam mencapai tujuannya, dan *Attitude* adalah tingkat kepuasan pengguna dalam menggunakan perangkat lunak. (Ola et al., 2016) Cara penghitungan hasil kuisioner yaitu dengan rumus sebagai berikut:

$$Persentase = \frac{\text{total skor}}{\text{skor} \times \text{jumlah responden}} \times 100\%$$

Kriteria persentase dapat dilihat pada tabel 4-6 sebagai berikut:

Tabel 4-6. Kriteria persentase kuisioner

Kategori	Keterangan
81%-100%	Sangat Setuju
61%-80%	Setuju
41%-60%	Cukup Setuju
21%-40%	Kurang Setuju
0%-20%	Tidak Setuju

Berikut adalah tabel daftar pertanyaan dalam kuisisioner yang dapat dilihat pada tabel 4-7 sebagai berikut:

Tabel 4-7. Daftar pertanyaan kuisisioner

No	Pertanyaan	Aspek
1	Apakah perangkat lunak pengirim GCode pada mesin mini <i>Computer Numerical Control</i> berbasis GRBL mudah dipahami dan digunakan?	<i>Learnability</i>
2	Apakah tombol perangkat lunak pengirim GCode pada Mesin Mini <i>Computer Numerical Control</i> berbasis GRBL dapat dilihat perubahan ketika di klik dan tidak?	<i>Learnability</i>
3	Apakah perangkat lunak dapat menentukan titik nol secara <i>serial port</i> maupun <i>wireless</i> ?	<i>Effectiveness</i>
4	Apakah perangkat lunak dapat melakukan pengiriman <i>file</i> GCode secara <i>serial port</i> maupun <i>wireless</i> ?	<i>Effectiveness</i>
5	Apakah perangkat lunak pengirim GCode pada mesin mini <i>Computer Numerical Control</i> berbasis GRBL yang dapat dihubungkan dengan <i>serial port</i> dan <i>wireless</i> di masa depan akan sering anda gunakan?	<i>Attitude</i>

Hasil pengujian usability dengan kuisisioner, dapat dilihat pada tabel 4-8 dan tabel 4-9 sebagai berikut:

Tabel 4-8. Hasil pengujian usability

Pertanyaan	Jawaban	Skor	Jumlah responden	Total Skor	Nilai Persentase (%)
1	Sangat Setuju	5	2	10	$\frac{14}{15} \times 100\% = 93,34\%$
	Setuju	4	1	4	
	Cukup Setuju	3	0	0	
	Kurang Setuju	2	0	0	
	Tidak Setuju	1	0	0	
Jumlah			3	14	

Pertanyaan	Jawaban	Skor	Jumlah responden	Total Skor	Nilai Persentase (%)
2	Sangat Setuju	5	0	0	$\frac{12}{15} \times 100\% = 80\%$
	Setuju	4	3	12	
	Cukup Setuju	3	0	0	
	Kurang Setuju	2	0	0	
	Tidak Setuju	1	0	0	
Jumlah			3	12	
3	Sangat Setuju	5	3	15	$\frac{15}{15} \times 100\% = 100\%$
	Setuju	4	0	0	
	Cukup Setuju	3	0	0	
	Kurang Setuju	2	0	0	
	Tidak Setuju	1	0	0	
Jumlah			3	15	
4	Sangat Setuju	5	3	15	$\frac{15}{15} \times 100\% = 100\%$
	Setuju	4	0	0	
	Cukup Setuju	3	0	0	
	Kurang Setuju	2	0	0	
	Tidak Setuju	1	0	0	
Jumlah			3	15	
5	Sangat Setuju	5	1	5	$\frac{12}{15} \times 100\% = 80\%$
	Setuju	4	1	4	
	Cukup Setuju	3	1	3	
	Kurang Setuju	2	0	0	
	Tidak Setuju	1	0	0	
Jumlah			3	12	

Tabel 4-9. Pengelolahan skala kuisisioner

Pertanyaan	Nilai Persentase	Keterangan
1	93,34%	Sangat Setuju
2	80%	Setuju
3	100%	Sangat Setuju
4	100%	Sangat Setuju
5	80%	Setuju
Rata-rata	$\frac{93,34\% + 80\% + 100\% + 100\% + 80\%}{5} = 90,67\%$	Sangat Setuju

4.3 Analisis dan Pembahasan

4.3.1 Analisis *Front End Develop* (Antarmuka)

Pengujian *Front End Develop* dilakukan untuk mengetahui tombol yang dibuat dapat berfungsi atau tidak dan pengguna dapat membedakan ketika tombol diklik dan tidak. Dari hasil pengujian dari *Front End Develop* yang dibuat dengan *software* Visual Studio 2019 WPF dengan Bahasa pemrograman XAML dapat berfungsi dan berhasil karena secara visual pengguna dapat membedakan tombol yang diklik, tombol yang tersentuh oleh kursor dan tombol yang tidak diklik. Hal ini berdasarkan dari hasil perancangan yang dapat dilihat dari gambar 4-1, gambar 4-2, gambar 4-3 yang menunjukkan hasil tampilan saat melakukan pemilihan dan pengaturan koneksi secara *serial port* maupun *wireless*, gambar 4-4 yang menunjukkan hasil tampilan untuk melakukan kontrol pengiriman *file*, dan gambar 4-5 yang menunjukkan hasil tampilan untuk melakukan kontrol secara manual untuk melakukan penentuan titik nol dan menggerakkan berdasarkan input GCode yang dilakukan oleh pengguna. Hasil pengujian berdasarkan *UI element* yang digunakan dan fungsi yang dihasilkan dapat dilihat pada tabel 4-1.

4.3.2 Analisis Back End Develop (Sistem)

Back End Develop dari perangkat lunak yang dibuat dengan *software* Visual Studio 2019 WPF dengan bahasa pemrograman C# atau CSharp. Pengujian *Back End Develop* yang dilakukan seperti dengan tabel 4-2 menunjukkan bahwa perancangan perangkat lunak berhasil dilakukan. Perangkat lunak yang dibuat dapat menghubungkan dan memutus koneksi secara *serial port* dan *wireless* yang digunakan sebagai perantara pengiriman data ke mikrokontroler Wemos D1 R32 dapat dilihat pada gambar 4-17. Dalam perangkat lunak yang dibuat berhasil mengirimkan *file* GCode perbaris secara runtut dan perangkat lunak berhasil mengirimkan GCode yang dimasukkan secara manual serta dapat menentukan titik nol setiap sumbu dengan variasi kecepatan gerak pemakanan dan jarak yang dimasukkan oleh pengguna secara *serial port* dan *wireless* yang dapat dilihat pada gambar 4-18 dan gambar 4-19.

4.3.3 Analisis Pengiriman Data Terhadap Gerak CNC

Pada hasil pengujian pengiriman data terhadap gerak CNC yang dilakukan dengan mesin CNC GRBL *milling* 3018 Pro. Dalam pengujian ini terdapat 3 bentuk yaitu persegi, lingkaran dan segitiga yang kemudian dirubah menjadi kode GCode, menghasilkan gerakan yang sesuai. Hasil pengujian ini dapat dilihat pada gambar 4-20 dengan pengiriman secara *serial port* dan gambar 4-21 dengan pengiriman secara *wireless*. Dari hasil pengujian dapat melakukan pengiriman data dengan 3 variasi GCode dengan pengiriman secara *serial port* maupun *wireless* berhasil dilakukan, kemudian dalam melakukan pengiriman setiap *file* GCode yang dikirim dapat berfungsi mengirimkan secara runtut sampai baris terakhir *file* ke mikrokontroler Wemos D1 R32.

4.3.4 Analisis Usability

Pada hasil pengujian usability dalam bentuk kuisisioner yang terdiri dari lima pertanyaan yang dibagikan kepada 3 responden setelah menggunakan perangkat lunak. Kuisisioner dilakukan dalam skala *Likert* atau skala satu sampai lima yang bertujuan untuk mengetahui jawaban setuju atau tidaknya responden

terhadap suatu pertanyaan. Berdasarkan dari hasil pengujian pada tabel 4-8 nilai persentase pada pertanyaan pertama yaitu 93,34%, nilai persentase pada pertanyaan kedua yaitu 80%, nilai persentase pada pertanyaan ketiga yaitu 100%, nilai persentase pada pertanyaan keempat yaitu 100%, nilai persentase pada pertanyaan kelima yaitu 80%. Hasil dari pengujian usability berdasarkan dari tabel 4-9 yaitu perangkat lunak Pengirim GCode pada mesin mini *Computer Numerical Control* berbasis GRBL mudah dipahami dan digunakan, tombol perangkat lunak pengirim GCode pada Mesin Mini *Computer Numerical Control* berbasis GRBL dapat dilihat perubahan ketika di klik, perangkat lunak dapat menentukan titik nol dan dapat melakukan pengiriman *file* GCode secara *serial port* maupun *wireless*.



BAB 5

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil perancangan dan hasil pengujian dapat diambil kesimpulan sebagai berikut:

1. Telah berhasil dirancang perangkat lunak yang dapat dioperasikan secara manual untuk menentukan titik nol atau titik awal pada Mesin Mini CNC 3 Axis berbasis GRBL dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 secara *serial port* maupun *wireless*.
2. Telah berhasil dirancang perangkat lunak yang dapat mengirim GCode secara bergantian, runtut pada mesin mini CNC 3-axis berbasis GRBL dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 secara *serial port* maupun *wireless*.

5.2 Saran atau Penelitian Selanjutnya

Berdasarkan hasil penelitian/perancangan yang telah dilakukan terdapat kekurangan dan memerlukan pengembangan lebih lanjut, yaitu sebagai berikut:

1. Melakukan pengembangan perangkat lunak yang dapat mengontrol lebih dari 3 axis dengan mikrokontroler Wemos D1 R32 atau ESPDuino32.
2. Melakukan pengembangan perangkat lunak yang dapat mengetahui titik koordinatnya.
3. Melakukan pengembangan perangkat lunak yang dapat memantau pergerakan CNC GRBL dengan kamera.
4. Melakukan pengembangan perangkat lunak yang dapat mengontrol CNC GRBL dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 untuk melakukan pergantian mata pahat secara otomatis.
5. Melakukan pengembangan perangkat lunak untuk mengontrol CNC GRBL dengan mikrokontroler Wemos D1 R32 atau ESPDuino32 yang dapat dikontrol jarak jauh dengan MQTT.

DAFTAR PUSTAKA

- Alfatah, F. A. (2013). Rancangan Perangkat Lunak G Code Interpreter untuk Pengendalian CNC 3 Aksis Berbasis Mikrokontroler. *UNIVERSITAS INDONESIA*.
- Frauke, P., Armin, E., & Frank, M. (2003). Requirements Engineering and Agile SoftwareDevelopment. *IEEE*. <https://doi.org/10.1109/ENABL.2003.1231428>
- G. Rodríguez, A., R. Vidal P, L., Miguel Díaz M, J., & Leonardo González Pinzón, C. (2014). G-Code interpreter development using Microsoft Visual Basic for ABL63 Control SystemS. *Electronic Vision*.
- Kruger, B. (2013). Arduino-CNC-Shield-Schematics. *Protoneer.Co.Nz Electronic Prototyping Specialists*. <https://blog.protoneer.co.nz/arduino-cnc-shield/arduino-cnc-shield-schematics/>
- Laddha, N. R., & Thakare, A. P. (2013). A Review on Serial Communication by UART. *International Journal of Advanced Research in Computer Science and Software Engineering*.
- Madakam, S., Holmukhe, R. M., & Jaiswal, D. K. (2018). The Future Digital Work Force: Robotic Process Automation (RPA). *Journal of Information Systems and Technology Management – Jistem USP*, 16, 2019, e201916001. <https://doi.org/10.4301/S1807-1775201916001>
- Mahmood, J. I., Hilmi, M. S., & Krishnan, P. (2019). Design And Development Of A Modular Computer Numerical Control (CNC) Machine Manipulator For Automation. *International Journal of Innovative Technology and*

ExploringEngineering(IJITEE).<https://doi.org/10.35940/ijitee.A3916.119>

119

Microsoft. (2021). *Desktop Guide (WPF .NET)*. <https://docs.microsoft.com/en-us/dotnet/desktop/wpf/overview/?view=netdesktop-5.0>

Nakov, S. (2013). *Fundamentals of Computer Programming With C# (The Bulgarian C# Programming Book)*. www.nakov.com.

Ola, Y. Y. O., Suyoto, & Purnomo, S. (2016). Pengujian Usability Antarmuka Aplikasi Mangente. *Seminar Nasional Teknologi Informasi Dan Komunikasi 2016 (SENTIKA 2016)*.

Parajuli, N., Thakuri, S. S., Shah, S. R., Bista, S., Shrestha, A., & Guragai, M. K. (2021). Modeling and Fabrication of Low Cost 3-Axis Computer Numerical Control (CNC) Machine. *International Journal of Advanced Engineering*, 4.

Rahmasari, A. (2017). *Perancangan dan Pembuatan Antarmuka Berbasis Visual Studio (VB.NET) dan Komunikasi Berbasis TCP/IP Pada Mesin CNC Routing 2.5D*. POLITEKNIK MANUFAKTUR NEGERI BANDUNG.

Rappaport, T. S. (2002). *Wireless Communications: Principle and Practice* (2nd ed.). NJ.

Rocha, P., de Souza, e S., Rogerio, & Tostes. (2010). *Prototype CNC machine design*. <https://doi.org/10.1109/INDUSCON.2010.5740068>

Sadre, A., Baechtel, D. F., & Graber, M. S. (1996). (Patent No. 5,485,620).

Sonny Jeon. (2021, January 23). GRBL. *Gnea / Grbl*. <https://github.com/gnea/grbl/wiki>

- Swati, T., & Rao, K. R. (2019). Industrial Process Monitoring System Using Esp32. *International Journal of Recent Technology and Engineering (IJRTE)*.
- Tomov, P. (2017). Increasing the Efficiency of Automation of Production Processes by Reporting the Parameters of the Parts' Flow. *TEM Journal*, 6(3), 484–487. <https://doi.org/10.18421/TEM63-08>
- Yakovlev, S. G., Keldibekov, J. K., & Gorbachenko, I. M. (2020). Software development for 3d visualization of g-code when working with CNC machines. *ICMSIT*. <https://doi.org/10.1088/1742-6596/1515/2/022082>
- Yerra, L., K, C., B, S., & Raju, P. R. (2017). Development of an Open Type CNC System for a 3-Axis Micro CNC Machine. *IAEME*.

LAMPIRAN 1

PROGRAM PERANGKAT LUNAK (MAINWINDOW)

```
MainWindow.xaml.cs  ModeControl.cs
CNC Launcher  CNC_Launcher.MainWindow
1  using CNC_Launcher.Core.Communication;
2  using Microsoft.Win32;
3  using System;
4  using System.Collections.Generic;
5  using System.Globalization;
6  using System.IO;
7  using System.Windows;
8  using System.Windows.Input;
9
10 namespace CNC_Launcher
11 {
12     /// <summary>
13     /// Interaction logic for MainWindow.xaml
14     /// </summary>
15     7 references
16     public partial class MainWindow : Window
17     {
18         1 reference
19         public MainWindow()
20         {
21             InitializeComponent();
22             modeControl.ConnectionStateChanged += Connection_ConnectionStateChanged;
23             modeControl.OperatingModeChanged += Main_OperatingMode_Changed;
24             Main_OperatingMode_Changed();
25         }
26
27         1 reference
28         public void Connection_ConnectionStateChanged()
29         {
30             menuItemConnect.Visibility = modeControl.IsConnected ? Visibility.Collapsed : Visibility.Visible;
31             menuItemDisconnect.Visibility = modeControl.IsConnected ? Visibility.Visible : Visibility.Collapsed;
32             menuItemDisconnect.Header = $"Disconnect {modeControl.PortName}";
33         }
34     }
35     2 references
```



```

MainWindow.xaml.cs  ModeControl.cs
CNC Launcher CNC_Launcher.MainWindow
31 private void Main_OperatingMode_Changed()
32 {
33     ButtonManualSend.IsEnabled = modeControl.Mode == ModeControl.OperatingMode.ManualControl;
34     ButtonZeroX.IsEnabled = modeControl.Mode == ModeControl.OperatingMode.ManualControl;
35     ButtonZeroY.IsEnabled = modeControl.Mode == ModeControl.OperatingMode.ManualControl;
36     ButtonZeroZ.IsEnabled = modeControl.Mode == ModeControl.OperatingMode.ManualControl;
37     ButtonFileUpload.IsEnabled = modeControl.Mode == ModeControl.OperatingMode.ManualControl;
38 }
39
40 public static NumberFormatInfo DecimalParseFormat = new NumberFormatInfo() { NumberDecimalSeparator = "." };
41
42 7 references
43 public static NumberFormatInfo DecimalOutputFormat
44 {
45     get
46     {
47         return new NumberFormatInfo() { NumberDecimalSeparator = ".", NumberDecimalDigits = 3 };
48     }
49 }
50 #region ButtonApp
51 0 references
52 private void Command_Open_Executed(object sender, ExecutedRoutedEventArgs e)
53 {
54     if (modeControl.Mode == ModeControl.OperatingMode.SendFile)
55         return;
56     OpenFileDialog UploadFile = new OpenFileDialog();
57     UploadFile.Filter = "GCode|*.tap;*.nc;*.ngc|All Files|*.*";
58     if (UploadFile.ShowDialog() == true)
59     {
60         Textbox_ProgramReadline.Text = File.ReadAllText(UploadFile.FileName);
61         modeControl.SetFile(File.ReadAllLines(UploadFile.FileName));
62     }
63 }

```

```

MainWindow.xaml.cs  ModeControl.cs
CNC Launcher CNC_Launcher.MainWindow
63 0 references
64 private void Command_SaveAs_Executed(object sender, ExecutedRoutedEventArgs e)
65 {
66     SaveFileDialog SaveFileGCode = new SaveFileDialog();
67     SaveFileGCode.ShowDialog();
68 }
69 0 references
70 private void MainWindow_Closed(object sender, EventArgs e)
71 {
72     Properties.Settings.Default.Save();
73 }
74 0 references
75 private void ToggleS_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)
76 {
77     if (ToggleS.Toggled1 == true)
78     {
79         ModeLabel.Content = "Online";
80     }
81     else
82     {
83         ModeLabel.Content = "Offline";
84     }
85 }
86 1 reference
87 private void ExitApp_Click(object sender, RoutedEventArgs e)
88 {
89     try
90     {
91         Application.Current.Shutdown();
92     }
93     catch (Exception ex)
94     {
95         MessageBox.Show(ex.Message);
96     }
97 }

```

```
MainWindow.xaml.cs  ModeControl.cs
C# CNC Launcher CNC_Launcher.MainWindow
95
1 reference
96 private void MaximizeApp_Click(object sender, RoutedEventArgs e)
97 {
98     try
99     {
100         this.WindowState = WindowState.Maximized;
101     }
102     catch (Exception ex)
103     {
104         MessageBox.Show(ex.Message);
105     }
106 }
107
1 reference
108 private void MinimizeApp_Click(object sender, RoutedEventArgs e)
109 {
110     try
111     {
112         this.WindowState = WindowState.Minimized;
113     }
114     catch (Exception ex)
115     {
116         MessageBox.Show(ex.Message);
117     }
118 }
119 #endregion ButtonApp
120
121 #region ButtonFile
1 reference
122 private void ButtonUpload_Click(object sender, RoutedEventArgs e)
123 {
124     if (modeControl.Mode == ModeControl.OperatingMode.SendFile)
125         return;

```



```

MainWindow.xaml.cs  X ModeControl.cs
CNC Launcher CNC_Launcher.MainWindow
121 #region ButtonFile
122 1 reference
123 private void ButtonUpload_Click(object sender, RoutedEventArgs e)
124 {
125     if (modeControl.Mode == ModeControl.OperatingMode.SendFile)
126         return;
127     OpenFileDialog UploadFile = new OpenFileDialog();
128     UploadFile.Filter = "GCode|*.tap;*.nc;*.ngc|All Files|*.*";
129     if (UploadFile.ShowDialog() == true)
130     {
131         Textbox_ProgramReadline.Text = File.ReadAllText(UploadFile.FileName);
132         modeControl.SetFile(File.ReadAllLines(UploadFile.FileName));
133     }
134 1 reference
135 private void StartFile_ButtonClick(object sender, RoutedEventArgs e)
136 {
137     modeControl.FileStart();
138     Console.WriteLine("FileStart");
139 }
140 1 reference
141 private void PauseFile_ButtonClick(object sender, RoutedEventArgs e)
142 {
143     modeControl.FilePause();
144     Console.WriteLine("FilePause");
145 }
146 1 reference
147 private void StopFile_ButtonClick(object sender, RoutedEventArgs e)
148 {
149     modeControl.FileStop();
150     Console.WriteLine("FileStop");
151 }

```

```

MainWindow.xaml.cs*  X ModeControl.cs
CNC Launcher CNC_Launcher.MainWindow
146 1 reference
147 private void StopFile_ButtonClick(object sender, RoutedEventArgs e)
148 {
149     modeControl.FileStop();
150     Console.WriteLine("FileStop");
151 }
152 1 reference
153 private void SetAllZero_ButtonClick(object sender, RoutedEventArgs e)
154 {
155     if (!modeControl.IsConnected)
156         return;
157     if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
158         return;
159     modeControl.SendLine("G92 X0 Y0 Z0");
160     Console.WriteLine("SetAllZero");
161 }
162 #endregion ButtonFile
163
164 #region JogManual
165
166 1 reference
167 private void Xplus_buttonclick(object sender, RoutedEventArgs e)
168 {
169     if (!modeControl.IsConnected)
170         return;
171     if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
172         return;
173     if (TextboxDistanceManual != null && TextboxFeedRateManual.Text != null)
174     {
175         modeControl.SendLine(string.Format(DecimalOutputFormat, "G91F{0:0.#}X{1:0.##}",
176             TextboxFeedRateManual.Text, TextboxDistanceManual.Text));

```

```

MainWindow.xaml.cs*  ModeControl.cs
CNC Launcher CNC_Launcher.MainWindow
180
181 1 reference
private void Xmin_buttonclick(object sender, RoutedEventArgs e)
182 {
183     if (!modeControl.IsConnected)
184         return;
185     if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
186         return;
187
188     if (TextBoxDistanceManual != null && TextBoxFeedRateManual.Text != null)
189     {
190         modeControl.SendLine(string.Format(DecimalOutputFormat, "G91F{0:0.##}X-{1:0.###}",
191             TextBoxFeedRateManual.Text, TextBoxDistanceManual.Text));
192     }
193     Console.WriteLine("XminButtonOK");
194 }
195
196 1 reference
private void Ymin_buttonclick(object sender, RoutedEventArgs e)
197 {
198     if (!modeControl.IsConnected)
199         return;
200     if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
201         return;
202
203     if (TextBoxDistanceManual != null && TextBoxFeedRateManual.Text != null)
204     {
205         modeControl.SendLine(string.Format(DecimalOutputFormat, "G91F{0:0.##}Y-{1:0.###}",
206             TextBoxFeedRateManual.Text, TextBoxDistanceManual.Text));
207     }
208     Console.WriteLine("YminButtonOK");
209 }
210

```

```

MainWindow.xaml.cs*  ModeControl.cs
CNC Launcher CNC_Launcher.MainWindow
211 1 reference
private void Yplus_buttonclick(object sender, RoutedEventArgs e)
212 {
213     if (!modeControl.IsConnected)
214         return;
215     if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
216         return;
217
218     if (TextBoxDistanceManual != null && TextBoxFeedRateManual.Text != null)
219     {
220         modeControl.SendLine(string.Format(DecimalOutputFormat, "G91F{0:0.##}Y{1:0.###}",
221             TextBoxFeedRateManual.Text, TextBoxDistanceManual.Text));
222     }
223     Console.WriteLine("YplusButtonOK");
224 }
225
226 1 reference
private void Zplus_buttonclick(object sender, RoutedEventArgs e)
227 {
228     if (!modeControl.IsConnected)
229         return;
230     if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
231         return;
232
233     if (TextBoxDistanceManual != null && TextBoxFeedRateManual.Text != null)
234     {
235         modeControl.SendLine(string.Format(DecimalOutputFormat, "G91F{0:0.##}Z{1:0.###}",
236             TextBoxFeedRateManual.Text, TextBoxDistanceManual.Text));
237     }
238     Console.WriteLine("ZplusButtonOK");
239 }
240
241 1 reference
private void Zmin_buttonclick(object sender, RoutedEventArgs e)

```



```

MainWindow.xaml.cs*  X ModeControl.cs
CNC Launcher CNC_Launcher.MainWindow
240
241 1 reference
private void Zmin_buttonclick(object sender, RoutedEventArgs e)
242  {
243     if (!modeControl.IsConnected)
244         return;
245     if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
246         return;
247
248     if (TextBoxDistanceManual != null && TextBoxFeedRateManual.Text != null)
249     {
250         modeControl.SendLine(string.Format(DecimalOutputFormat, "G91F{0:0.##}Z-{1:0.###}",
251             TextBoxFeedRateManual.Text, TextBoxDistanceManual.Text));
252     }
253     Console.WriteLine("ZminButtonOK");
254 }
255
256 #endregion JogManual
257
258 Camera
259
260 #region ButtonManual
261 private List<string> ManualCommands = new List<string>();
262 1 reference
263 void ManualSend()
264 {
265     string tosend;
266     tosend = TextBoxManualSender.Text;
267
268     modeControl.SendLine(tosend);
269
270     ManualCommands.Insert(0, tosend);
271 }
272
273
274
275
276
277
278

```

```

MainWindow.xaml.cs*  X ModeControl.cs
CNC Launcher CNC_Launcher.MainWindow
273
274     string tosend;
275     tosend = TextBoxManualSender.Text;
276
277     modeControl.SendLine(tosend);
278
279     ManualCommands.Insert(0, tosend);
280
281     TextBoxManualSender.Text = "";
282 }
283 1 reference
private void ButtonManualSend_Click(object sender, RoutedEventArgs e)
284 {
285     ManualSend();
286     Console.WriteLine("ManualSender_received");
287 }
288 1 reference
private void ButtonZeroX_Click(object sender, RoutedEventArgs e)
289 {
290     if (!modeControl.IsConnected)
291         return;
292     if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
293         return;
294
295     modeControl.SendLine("G92 X0");
296     Console.WriteLine("G92 X0");
297 }
298 1 reference
private void ButtonZeroY_Click(object sender, RoutedEventArgs e)
299 {
300     if (!modeControl.IsConnected)
301         return;
302     if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
303         return;

```

```
MainWindow.xaml.cs* X ModeControl.cs
C# CNC Launcher CNC_Launcher.MainWindow
289         if (!modeControl.IsConnected)
290             return;
291         if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
292             return;
293
294         modeControl.SendLine("G92 X0");
295         Console.WriteLine("G92 X0");
296     }
297
298     1 reference
299     private void ButtonZeroY_Click(object sender, RoutedEventArgs e)
300     {
301         if (!modeControl.IsConnected)
302             return;
303         if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
304             return;
305
306         modeControl.SendLine("G92 Y0");
307         Console.WriteLine("G92 Y0");
308     }
309     1 reference
310     private void ButtonZeroZ_Click(object sender, RoutedEventArgs e)
311     {
312         if (!modeControl.IsConnected)
313             return;
314         if (modeControl.Mode != ModeControl.OperatingMode.ManualControl)
315             return;
316
317         modeControl.SendLine("G92 Z0");
318         Console.WriteLine("G92 Z0");
319     }
320 #endregion ButtonManual
```



LAMPIRAN 2

PROGRAM PERANGKAT LUNAK (HEADERMENU)

```
HeaderMenu.cs  ModeControl.cs
C# CNC Launcher  CNC_Launcher.MainWindow
1  using CNC_Launcher.Core.Communication;
2  using System;
3  using System.ComponentModel;
4  using System.IO.Ports;
5  using System.Runtime.CompilerServices;
6  using System.Windows;
7  using System.Windows.Controls;
8  using System.Windows.Input;
9  using System.Windows.Media;
10
11 namespace CNC_Launcher
12 {
13     7 references
14     partial class MainWindow : INotifyPropertyChanged
15     {
16         ModeControl modeControl = new ModeControl();
17         public event PropertyChangedEventHandler PropertyChanged;
18
19         0 references
20         protected void OnPropertyChanged([CallerMemberName] string name = null)
21         {
22             PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(name));
23         }
24
25         1 reference
26         private void Menu_ListSerialports(object sender, RoutedEventArgs e)
27         {
28             menuItemConnect.Items.Clear();
29
30             foreach (string port in SerialPort.GetPortNames())
31             {
32                 MenuItem portItem = new MenuItem { Header = port };
33                 portItem.Click += PortItem Click;
```



```

HeaderMenu.cs*  X ModeControl.cs
C# CNC Launcher - CNC_Launcher.MainWindow
29         MenuItem portItem = new MenuItem { Header = port };
30
31         portItem.Click += PortItem_Click;
32         portItem.Background = new SolidColorBrush(Color.FromRgb(((int)((byte)9))),
33         ((int)((byte)48))), ((int)((byte)77))));
34         portItem.Foreground = new SolidColorBrush(Colors.Silver);
35
36         menuItemConnect.Items.Add(portItem);
37     }
38
39     menuItemConnect.Items.Add(menuItemNetwork);
40 }
41
42     1 reference
43     private void PortItem_Click(object sender, RoutedEventArgs e)
44     {
45         if (!modeControl.ConnectSerial((string)((MenuItem)sender).Header))
46         {
47             App.Message("Serial Error");
48         }
49         ToggleS.Toggled1 = false;
50         Console.WriteLine("SerialConnect");
51     }
52
53     3 references
54     private void MenuItemLostKeyboardFocus(object sender, KeyboardFocusChangedEventArgs e)
55     {
56         if (HeaderMenu.IsKeyboardFocusWithin)
57             e.Handled = true;
58     }
59
60     1 reference
61     private void MenuItem_Network_Click(object sender, RoutedEventArgs e)
62     {
63         if (HeaderMenu.IsKeyboardFocusWithin)
64             e.Handled = true;
65     }
66
67     1 reference
68     private void MenuItem_Disconnect_Click(object sender, RoutedEventArgs e)
69     {
70         modeControl.Disconnect();
71         Console.WriteLine("Disconnected");
72     }
73
74     1 reference
75     private void TextBoxNetworkAddress_Load(object sender, RoutedEventArgs e)
76     {
77         TextBoxNetworkAddress.Text = Properties.Settings.Default.NetworkAddress;
78     }
79
80     1 reference
81     private void TextBoxNetworkAddress_Unload(object sender, RoutedEventArgs e)
82     {
83         Properties.Settings.Default.NetworkAddress = TextBoxNetworkAddress.Text;
84     }
85 }

```

LAMPIRAN 3

PROGRAM PERANGKAT LUNAK (MODECONTROL)

```
ModeControl.cs  + X
C# CNC Launcher  CNC_Launcher.Core.Co
1  using CNC_Launcher.Properties;
2  using System;
3  using System.Collections;
4  using System.Collections.Generic;
5  using System.Collections.ObjectModel;
6  using System.IO;
7  using System.IO.Ports;
8  using System.Net.Sockets;
9  using System.Text;
10 using System.Threading;
11 using System.Threading.Tasks;
12 using System.Windows;
13
14 namespace CNC_Launcher.Core.Communication
15 {
16     21 references
17     class ModeControl
18     {
19         34 references
20         public enum OperatingMode
21         {
22             ManualControl,
23             SendFile,
24             NoOperating
25         }
26         #region Action
27         public event Action<string> LineSent;
28         public event Action BufferStateChanged;
29         public event Action OperatingModeChanged;
30         public event Action FileChanged;
31         public event Action ConnectionStateChanged;
32         #endregion Action
33     }
34 }
```



```
ModeControl.cs  X
C# CNC Launcher  CNC_Launcher.Core.Communication.ModeControl
31
32 #region field RunningFile
33 private ReadOnlyCollection<bool> _pauselines = new ReadOnlyCollection<bool>(new bool[0]);
34 public ReadOnlyCollection<bool> PauseLines
35 {
36     get { return _pauselines; }
37     private set { _pauselines = value; }
38 }
39
40 private ReadOnlyCollection<string> _File = new ReadOnlyCollection<string>(new string[0]);
41 public ReadOnlyCollection<string> File
42 {
43     get { return _File; }
44     private set
45     {
46         _File = value;
47         FilePosition = 0;
48
49         RaiseEvent(FileChanged);
50     }
51 }
52
53 private int _FilePosition = 0;
54 public int FilePosition
55 {
56     get { return _FilePosition; }
57     private set
58     {
59         _FilePosition = value;
60     }
61 }
```

```
ModeControl.cs  X
C# CNC Launcher  CNC_Launcher.Core.Communication.M
62 private int _bufferState;
63 public int BufferState
64 {
65     get { return _bufferState; }
66     private set
67     {
68         if (_bufferState == value)
69             return;
70
71         _bufferState = value;
72
73         RaiseEvent(BufferStateChanged);
74     }
75 }
76 #endregion field RunningFile
77
78 #region field Connectivity
79 private OperatingMode _mode = OperatingMode.NoOperating;
80 public OperatingMode Mode
81 {
82     get { return _mode; }
83     private set
84     {
85         if (_mode == value)
86             return;
87
88         _mode = value;
89         RaiseEvent(OperatingModeChanged);
90     }
91 } //Contruccion untuk mendeskripsikan lingkup mode karena terdapat 3mode
92 private bool _isconnected = false;
```

```

ModeControl.cs  X
CNC Launcher  CNC_Launcher.Core.Communication.ModeControl
90     }
91     } //Construction untuk mendeskripsikan lingkup mode karena terdapat 3mode
92     private bool _isconnected = false;
93     25 references
94     public bool IsConnected
95     {
96         get { return _isconnected; }
97         private set
98         {
99             if (value == _isconnected)
100                return;
101
102             _isconnected = value;
103
104             if (!IsConnected)
105                 Mode = OperatingMode.NoOperating;
106             RaiseEvent(ConnectionStateChanged);
107         }
108     } //Sebagai construction untuk mendeskripsikan mode tersambung dengan CNC
109
110     3 references
111     public string PortName { get; private set; } //Deskripsi untuk port yang tersedia dengan
112
113     public Stream ConnectionStream; // Construction untuk perubahan mode koneksi yang ditangk
114
115     private Thread WorkerThread; //Construction untuk tetap berjalan di background sehingga k
116     #endregion field Connectivity
117
118     #region queue
119     Queue<string> _sender = Queue.Synchronized(new Queue()); //Construction untuk melakukan antrian yan
120     Queue<string> _tosend = Queue.Synchronized(new Queue()); //Construction untuk melakukan antrian yan
121     Queue<string> _tosendpriority = Queue.Synchronized(new Queue()); //Construction untuk melakukan ant
122     #endregion queue

```

```

ModeControl.cs  X
CNC Launcher  CNC_Launcher.Core.Communication.ModeControl
122     #region Running
123     1 reference
124     public void Running()
125     {
126         try
127         {
128             StreamReader ConnectionReader = new StreamReader(ConnectionStream, Encoding.ASCII);
129             StreamWriter ConnectionWriter = new StreamWriter(ConnectionStream, Encoding.ASCII);
130             //Func untuk mengatur buffer size yang dikelola aplikasi dan untuk mendeskripsikan j
131
132             int ControllerBufferSize = Settings.Default.controllerBufferSize;
133             BufferState = 0;
134
135             ConnectionWriter.Write("\n$G\n");
136             ConnectionWriter.Write("\n#\n");
137             ConnectionWriter.Flush();
138
139             //Func untuk mendeskripsikan pengiriman gcode per-line dengan koneksi yang tersedia
140             while (true)
141             {
142                 Task<string> lineTask = ConnectionReader.ReadLineAsync();
143
144                 while (!lineTask.IsCompleted)
145                 {
146                     if (!IsConnected)
147                     {
148                         return;
149                     }
150                     while (ToSendPriority.Count > 0)
151                     {
152                         ConnectionWriter.Write((char)ToSendPriority.Dequeue());
153                         ConnectionWriter.Flush();
154                     }
155                 }
156             }

```

```
ModeControl.cs + X
CNC Launcher CNC_Launcher.Core.Communication.ModeControl LineSent
154 if (Mode == OperatingMode.SendFile) //Func untuk mengatur format pengiriman gcode file per-line dengan posisi
155 {
156     if (File.Count > FilePosition && (File[FilePosition].Length + 1) < (ControllerBufferSize - BufferState))
157     {
158         string linesending = File[FilePosition].Replace(" ", ""); //Menghilangkan spasi line gcode yang akan
159
160         ConnectionWriter.Write(linesending); //line yang siap kirim (sudah tidak ada spasi)
161         ConnectionWriter.Write('\n'); //enter
162         ConnectionWriter.Flush(); //untuk menghapus buffer data yang dikirim perline
163
164         RaiseEvent(LineSent, linesending);
165
166         BufferState += linesending.Length + 1; //jumlah bufferstate sama dengan jumlah kata line gcode siap k
167
168         Sender.Enqueue(linesending); //mendaftarkan atau menambahkan antrian
169         Console.WriteLine("linesending");
170
171         //Func untuk mnegatur proses pengiriman gcode secara manual dengan memilih mulainya line
172         if (PauseLines[FilePosition] && Settings.Default.PauseFileOnHold) //line pada file yang di pause berb
173         {
174             Mode = OperatingMode.ManualControl;
175         }
176
177         if (++FilePosition >= File.Count) //pre increment dari file line tertentu lebih dari sama dengan jumla
178         {
179             FilePosition = 0;
180             Mode = OperatingMode.ManualControl;
181         }
182     }
183 }
}

ModeControl.cs + X
CNC Launcher CNC_Launcher.Core.Communication.ModeControl LineSent
181     }
182 }
183 }
184 //Func untuk mendeskripsikan apabila fungsi if diatas tidak terpenuhi
185 //Func ini untuk melakukan proses berikutnya (progres line file)
186 else if (ToSend.Count > 0 && ((string)ToSend.Peek()).Length + 1 < (ControllerBufferSize - BufferState)) /
187 {
188     string linesending = ((string)ToSend.Dequeue()).Replace(" ", ""); //line gcode yang dikirim sama dengan
189
190     ConnectionWriter.Write(linesending); //line yang siap kirim yang sudah dihilangkan spasinya
191     ConnectionWriter.Write('\n'); //enter
192     ConnectionWriter.Flush(); //untuk menghapus buffer data yang dikirim perline
193     RaiseEvent(LineSent, linesending);
194
195     BufferState += linesending.Length + 1; //jumlah bufferstate sama dengan jumlah kata line gcode siap kir
196
197     Sender.Enqueue(linesending); //mendaftarkan atau menambahkan antrian
198     Console.WriteLine("linesending");
199 }
200
201 TimeSpan WaitTime = TimeSpan.FromMilliseconds(0.2);
202 Thread.Sleep(WaitTime); //lama waktu background sleep sebesar nilai jeda
203
204 BufferState = 0;
205 }
206 string line = lineTask.Result; //string baru "line" adalah hasil line yang berjalan
207 }
208 }
209
210 catch
211 {
212 }
213 }
```




```
ModeControl.cs  X
C# CNC Launcher  CNC_Launcher.Core.Communicati
215 #endregion Running
216
217 #region Connectivity
218 3 references
219 public void Disconnect()
220 {
221     IsConnected = false;
222
223     if (ConnectionStream != null)
224     {
225         ConnectionStream.Close();
226         ConnectionStream = null;
227     }
228
229     Mode = OperatingMode.NoOperating;
230     RaiseEvent(ConnectionStateChanged);
231 }
232 2 references
233 public void Connect()
234 {
235     if (!IsConnected)
236     {
237         return;
238     }
239
240     ToSend.Clear();
241     ToSendPriority.Clear();
242     Sender.Clear();
243
244     Mode = OperatingMode.ManualControl;
245     RaiseEvent(ConnectionStateChanged);
246
247     WorkerThread = new Thread(Running);
248     WorkerThread.Priority = ThreadPriority.AboveNormal;
```



```
ModeControl.cs → X
CNC Launcher CNC_Launcher.Core.Communic
243
244 WorkerThread = new Thread(Running);
245 WorkerThread.Priority = ThreadPriority.AboveNormal;
246 WorkerThread.Start();
247
248 1 reference
public bool ConnectNetwork(string address)
249 {
250     Disconnect();
251     PortName = address;
252
253     try
254     {
255         TcpClient tcpclient = new TcpClient();
256         tcpclient.Connect(UtilityTCP.ParseIPEndPoint(address));
257
258         ConnectionStream = tcpclient.GetStream();
259         IsConnected = true;
260
261         Connect();
262         return true;
263     }
264     catch
265     {
266         return false;
267     }
268 }
269
270 1 reference
public bool ConnectSerial(string portName)
271 {
272     Disconnect();
273     PortName = portName;
274
```



```
ModeControl.cs  X
CNC Launcher  CNC_Launcher.Core.Communication.ModeControl

270 public bool ConnectSerial(string portName)
271 {
272     Disconnect();
273     PortName = portName;
274
275     try
276     {
277         SerialPort port = new SerialPort(portName, Settings.Default.SerialBaudRate);
278         port.Open();
279
280         ConnectionStream = port.BaseStream;
281         IsConnected = true;
282         Connect();
283
284         return true;
285     }
286     catch
287     {
288         return false;
289     }
290 }
291 #endregion Connectivity
292
293 #region Accessibility
294 2 references
295 public void SetFile(IList<string> file)
296 {
297     if (Mode == OperatingMode.SendFile)
298     {
299         return;
300     }
301
302     bool[] pauselines = new bool[file.Count];
```



```
ModeControl.cs + X
C# CNC Launcher CNC_Launcher.Core.Communic
301 bool[] pauselines = new bool[file.Count];
302
303 File = new ReadOnlyCollection<string>(file);
304 Pauselines = new ReadOnlyCollection<bool>(pauselines);
305
306 FilePosition = 0;
307
308 }
309 15 references
310 public void SendLine(string line)
311 {
312     if (!IsConnected)
313     {
314         return;
315     }
316     ToSend.Enqueue(line);
317 }
318 1 reference
319 public void FileStop()
320 {
321     if (!IsConnected)
322     {
323         return;
324     }
325     Mode = OperatingMode.ManualControl;
326
327     ToSend.Clear();
328     ToSendPriority.Clear();
329     Sender.Clear();
330     ToSendPriority.Enqueue((char)0x18);
```



```
ModeControl.cs  X
CNC Launcher CNC_Launcher.Co
329     SendPriority();
330     ToSendPriority.Enqueue((char)0x18);
331
332     BufferState = 0;
333
334     SendLine("$G");
335     SendLine("#");
336 }
337 1 reference
338 public void FileStart()
339 {
340     if (!IsConnected)
341     {
342         return;
343     }
344     if (Mode != OperatingMode.ManualControl)
345     {
346         return;
347     }
348     Mode = OperatingMode.SendFile;
349 }
350 1 reference
351 public void FilePause()
352 {
353     if (!IsConnected)
354     {
355         return;
356     }
357     if (Mode != OperatingMode.SendFile)
358     {
359         return;
360     }
```



```
ModeControl.cs + X
CNC Launcher CNC_Launcher.Core.Communi
358     if (Mode != OperatingMode.SendFile)
359     {
360         return;
361     }
362
363     Mode = OperatingMode.ManualControl;
364 }
365     7 references
366     public void SendControl(byte controlchar)
367     {
368         if (!IsConnected)
369         {
370             return;
371         }
372         ToSendPriority.Enqueue((char)controlchar);
373     }
374     #endregion Accessibility
375
376     2 references
377     private void RaiseEvent(Action<string> action, string param)
378     {
379         if (action == null)
380             return;
381         Application.Current.Dispatcher.BeginInvoke(action, param);
382     }
383     6 references
384     private void RaiseEvent(Action action)
385     {
386         if (action == null)
387             return;
```



```
ModeControl.cs  X
CNC Launcher  CNC_Launcher.Core.Commu
363     Mode = OperatingMode.ManualControl;
364     }
365     7 references
366     public void SendControl(byte controlchar)
367     {
368         if (!IsConnected)
369         {
370             return;
371         }
372         ToSendPriority.Enqueue((char)controlchar);
373     }
374     #endregion Accessibility
375
376     2 references
377     private void RaiseEvent(Action<string> action, string param)
378     {
379         if (action == null)
380             return;
381         Application.Current.Dispatcher.BeginInvoke(action, param);
382     }
383     6 references
384     private void RaiseEvent(Action action)
385     {
386         if (action == null)
387             return;
388         Application.Current.Dispatcher.BeginInvoke(action);
389     }
390     }
391 }
392 }
```



LAMPIRAN 4

PROGRAM PERANGKAT LUNAK (ANTARMUKA)

```
MainWindow.xaml -> X
Window
1 <Window x:Class="CNC_Launcher.MainWindow"
2   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
3   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
4   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
5   xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
6   xmlns:theme="clr-namespace:CNC_Launcher.Theme"
7   mc:Ignorable="d"
8   Title="CNC PCB" Height="450" Width="800"
9   Background="Transparent"
10  AllowsTransparency="True"
11  ResizeMode="NoResize"
12  WindowStyle="None"
13  WindowStartupLocation="CenterScreen"
14  Loaded="Window_Loaded"/>
15
16 <Border CornerRadius="20"
17   Background="#09304d">
18   <Grid>
19     <Grid.ColumnDefinitions>
20       <ColumnDefinition Width="150"/>
21       <ColumnDefinition/>
22     </Grid.ColumnDefinitions>
23     <Grid Grid.Column="0">
24       <Border CornerRadius="20"
25         Background="#09243c"
26         BorderThickness="1"
27         BorderBrush="White">
28       </Border>
29       <Grid>
30         <Grid.RowDefinitions>
31           <RowDefinition Height="100"/>
32           <RowDefinition/>
33           <RowDefinition Height="60"/>
34           </Grid.RowDefinitions>
35         <StackPanel Grid.Row="0">
36           <Image Source="/Image/Logo CNC PCB.png"
37             Height="95"/>
```



```
MainWindow.xaml  X
Window
37  <StackPanel Grid.Row="0">
38      <Image Source="/Image/Logo CNC PCB.png"
39             Height="95"
40             Width="70"
41             HorizontalAlignment="Center"
42             VerticalAlignment="Center">
43      </Image>
44  </StackPanel>
45
46  <Label Grid.Row="1"
47         Content="FeedRateManual"
48         VerticalContentAlignment="Top"
49         HorizontalContentAlignment="Left"
50         Margin="0,0,0,0"
51         FontSize="12"
52         FontFamily="Bahnschrift"
53         Foreground="Silver"/>
54  <TextBox Name="TextBoxFeedRateManual"
55           Grid.Row="1"
56           Height="20"
57           Width="100"
58           VerticalContentAlignment="Top"
59           HorizontalAlignment="Left"
60           Margin="20,0,0,230"
61           Text="1000"/>
62
63  <Label Grid.Row="1"
64         Content="DistanceManual"
65         VerticalContentAlignment="Top"
66         HorizontalContentAlignment="Left"
67         Margin="0,50,0,0"
68         FontSize="12"
69         FontFamily="Bahnschrift"
70         Foreground="Silver"/>
71  <TextBox Name="TextBoxDistanceManual"
72           Grid.Row="1"
73           Height="20"
74           Width="100"
75           VerticalContentAlignment="Center">
```

المعهد الإسلامي للدراسات والبحوث

```
MainWindow.xaml  X
Window
76         HorizontalAlignment="Left"
77         Margin="20,0,0,130"
78         Text="1"/>
79     <Separator Grid.Row="1"
80         Margin="0,0,0,30"/>
81
82     <TextBlock Text="FeedRate :"
83         Margin="5,0,0,115"
84         FontSize="16"
85         FontFamily="Bahnschrift SemiBold Condensed"
86         Foreground="#cccc33"
87         Grid.Row="1"
88         HorizontalAlignment="Left"
89         VerticalAlignment="Bottom"/>
90     <TextBlock Text="Spindle :"
91         Margin="5,0,0,70"
92         FontSize="16"
93         FontFamily="Bahnschrift SemiBold Condensed"
94         Foreground="#cccc33"
95         Grid.Row="1"
96         HorizontalAlignment="Left"
97         VerticalAlignment="Bottom"/>
98     <TextBlock Text="RapidRate :"
99         Margin="5,0,0,25"
100        FontSize="16"
101        FontFamily="Bahnschrift SemiBold Condensed"
102        Foreground="#cccc33"
103        Grid.Row="1"
104        HorizontalAlignment="Left"
105        VerticalAlignment="Bottom"/>
106
107
108     <Slider Name="FeedRate_Slider"
109         Width="100"
110         Height="20"
111         HorizontalAlignment="Left"
112         VerticalAlignment="Center"
113         Margin="10,80,0,0"
114         Background="#336666"/>
```



```
MainWindow.xaml
Window
106
107
108 <Slider Name="FeedRate_Slider"
109         Width="100"
110         Height="20"
111         HorizontalAlignment="Left"
112         VerticalAlignment="Center"
113         Margin="10,80,0,0"
114         Background="#336666"
115         Minimum="0"
116         Maximum="100"
117         ValueChanged="FeedRate_Slider_ValueChanged"
118         IsSnapToTickEnabled="True"
119         TickFrequency="1"
120         Loaded="SliderFeedRate_Load"
121         Unloaded="SliderFeedRate_Unload"
122         Grid.Row="1"/>
123 <Slider Name="Spindle_Slider"
124         Width="100"
125         Height="20"
126         HorizontalAlignment="Left"
127         VerticalAlignment="Center"
128         Margin="10,170,0,0"
129         Background="#336666"
130         Minimum="0"
131         Maximum="100"
132         ValueChanged="Spindle_Slider_ValueChanged"
133         IsSnapToTickEnabled="True"
134         TickFrequency="1"
135         Loaded="SliderSpindle_Load"
136         Unloaded="SliderSpindle_Unload"
137         Grid.Row="1"/>
138 <Slider Name="RapidRate_Slider"
139         Width="100"
140         Height="20"
141         HorizontalAlignment="Left"
142         VerticalAlignment="Center"
143         Margin="10,260,0,0"
144         Background="#336666"
```



```

MainWindow.xaml  X
Window  mcIgnorable
142     VerticalAlignment="Center"
143     Margin="10,260,0,0"
144     Background="■" #336666"
145     Minimum="0"
146     Maximum="100"
147     ValueChanged="RapidRate_Slider_ValueChanged"
148     IsSnapToTickEnabled="True"
149     Ticks="25, 50, 100"
150     Loaded="SliderRapidRate_Load"
151     Unloaded="SliderRapidRate_Unload"
152     Grid.Row="1"/>
153
154
155     <TextBox Margin="110,80,0,0"
156             FontSize="16"
157             FontFamily="Bahnschrift SemiBold"
158             Height="25"
159             Width="25"
160             Background="■" Transparent"
161             Foreground="■" Silver"
162             Cursor="IBeam"
163             Name="TextBoxFeedRate"
164             Text="{Binding ElementName=FeedRate_Slider, Path=Value, UpdateSourceTrigger=PropertyChanged}"
165             HorizontalContentAlignment="Center"
166             VerticalContentAlignment="Center"
167             Grid.Row="1"/>
168
169     <TextBox Margin="110,165,0,0"
170             FontSize="16"
171             FontFamily="Bahnschrift SemiBold"
172             Height="25"
173             Width="25"
174             Background="■" Transparent"
175             Foreground="■" Silver"
176             Cursor="IBeam"
177             Name="TextBoxSpindle"
178             Text="{Binding ElementName=Spindle_Slider, Path=Value, UpdateSourceTrigger=PropertyChanged}"
179             HorizontalContentAlignment="Center"
180             VerticalContentAlignment="Center"
181             Grid.Row="1"/>

```

```

MainWindow.xaml  X
Window  mcIgnorable
181
182     <TextBox Margin="110,260,0,0"
183             FontSize="16"
184             FontFamily="Bahnschrift SemiBold"
185             Height="25"
186             Width="25"
187             Background="■" Transparent"
188             Foreground="■" Silver"
189             Cursor="IBeam"
190             Name="TextBoxRapidRate"
191             Text="{Binding ElementName=RapidRate_Slider, Path=Value, UpdateSourceTrigger=PropertyChanged}"
192             HorizontalContentAlignment="Center"
193             VerticalContentAlignment="Center"
194             Grid.Row="1"/>
195
196     <Grid Grid.Row="5">
197         <Border CornerRadius="20"
198                 Background="■" Transparent"
199                 Height="40"
200                 Width="130"
201                 BorderThickness="2"
202                 BorderBrush="■" Silver">
203             <Grid>
204                 <Grid.ColumnDefinitions>
205                     <ColumnDefinition/>
206                     <ColumnDefinition/>
207                 </Grid.ColumnDefinitions>
208                 <Label x:Name="ModelLabel"
209                         Grid.Column="1"
210                         Height="30"
211                         Width="60"
212                         Background="■" Transparent"
213                         BorderThickness="0"
214                         Content="Offline"
215                         FontFamily="Arial"
216                         FontSize="16"
217                         Margin="0,0,0,0"
218                         Foreground="■" White"
219                         HorizontalAlignment="Center"
220                         VerticalAlignment="Center"/>

```

```

MainWindow.xaml  X
Window  mc:Ignorable
220
221     <theme:Toggle_Switch x:Name="ToggleS"
222         HorizontalAlignment="Left"
223         VerticalAlignment="Center"
224         MouseLeftButtonDown="ToggleS_MouseLeftButtonDown"/>
225     </Grid>
226 </Border>
227 </Grid>
228 </Grid>
229 </Grid>
230
231 <Grid Grid.Column="1">
232     <Grid.RowDefinitions>
233         <RowDefinition Height="Auto"/>
234         <RowDefinition Height="50"/>
235         <RowDefinition/>
236     </Grid.RowDefinitions>
237
238     <Menu IsMainMenu="True"
239         Name="HeaderMenu"
240         Height="20"
241         Width="200"
242         HorizontalAlignment="Left"
243         Background="■"Transparent"
244         Foreground="■"Silver"
245         BorderBrush="■"Transparent">
246
247         <MenuItem Header="File"
248             Background="■"Transparent"
249             Foreground="■"Silver">
250             <MenuItem Header="Open (GCode)"
251                 Command="ApplicationCommands.Open"
252                 Background="□"#09243c"
253                 Foreground="■"Silver"/>
254             <MenuItem Header="Save (GCode)"
255                 Command="ApplicationCommands.Save"
256                 Background="□"#09243c"
257                 Foreground="■"Silver"/>
258             <Separator/>
259             <MenuItem Header="Exit"
260                 Command="ApplicationCommands.Close"
261                 Background="□"#09243c"
262                 Foreground="■"Silver"/>
263         </MenuItem>
264
265         <MenuItem Header="GRBL"
266             Background="■"Transparent"
267             Foreground="■"Silver">
268             <MenuItem Header="Connect"
269                 Name="menuItemConnect"
270                 SubmenuOpened="Menu_ListSerialports"
271                 Background="□"#09243c"
272                 Foreground="■"Silver">
273                 <MenuItem Name="menuItemNetwork"
274                     Click="Menu_Connect_Network_Click"
275                     Background="□"#09243c"
276                     Foreground="■"Silver">
277                     <MenuItem.Header>
278                         <StackPanel Orientation="Horizontal"
279                             Background="□"#09243c">
280                             <Label Content="Network Address"
281                                 Width="100"
282                                 Background="□"#09243c"
283                                 Foreground="■"Silver"/>
284                             <TextBox Name="textBoxNetworkAddress"
285                                 MinWidth="100"
286                                 Loaded="textBoxNetworkAddress_Load"
287                                 Unloaded="textBoxNetworkAddress_Unload"
288                                 PreviewLostKeyboardFocus="Menu_TextBoxLostKeyboardFocus"
289                                 TextAlignment="Center"
290                                 VerticalAlignment="Center"
291                                 Padding="3"/>
292                         </StackPanel>
293                     </MenuItem.Header>
294                 </MenuItem>
295             </MenuItem>
296             <MenuItem Header="Disconnect"
297                 Name="menuItemDisconnect"

```