TUGAS AKHIR – EE184801

DISAIN DIAGRAM *LADDER* UNTUK BANYAK OBJEK PADA *FACTORY AUTOMATIC TRAINER* MENGGUNAKAN METODE *STATE DIAGRAM*

Andhiko Palito F
NRP 07111745000046

Dosen Pembimbing
Dr. Ir. Mochammad Rameli
Eka Iskandar, S.T., M.T.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh November
Surabaya 2019

FINAL PROJECT – EE184801

LADDER DIAGRAM DESIGN FOR MULTIPLE OBJECTS ON FACTORY AUTOMATIC TRAINER USING STATE DIAGRAM METHOD

Andhiko Palito F
NRP 07111745000046

Supervisor
Dr. Ir. Mochammad Rameli
Eka Iskandar, S.T., M.T.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Eletrical Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2019

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "Disain Diagram *Ladder* untuk Banyak Objek pada *Factory Automatic Trainer* Menggunakan Metode *State Diagram*" adalah merupakan hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia merima sangsi sesuai peraturan yang berlaku.

Surabaya, Juni 2019

Andhiko Palito F
NRP 07111745000046

DISAIN DIAGRAM *LADDER* UNTUK BANYAK OBJEK PADA *FACTORY AUTOMATIC TRAINER* MENGGUNAKAN METODE *STATE DIAGRAM*

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagai Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Teknik Sistem Pengaturan
Departemen Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui:

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Ir. Mochammad Rameli	Eka Iskandar, S.T., M.T.
NIP. 195412271981031002	NIP. 198005282008121001

SURABAYA
JULI, 2019

DISAIN DIAGRAM *LADDER* UNTUK BANYAK OBJEK PADA *FACTORY AUTOMATIC TRAINER* MENGGUNAKAN METODE *STATE DIAGRAM*

Andhiko Palito F-07111745000046

Pembimbing: 1. Dr. Ir. Mochammad Rameli
2. Eka Iskandar, S.T., M.T.

ABSTRAK

Programmable logic controller (PLC) merupakan salah satu perangkat yang umum digunakan sebagai perangakat kontrol di indutsri manufaktur. PLC banyak digunakan karena bahasa yang digunakan cukup mudah untuk dipahami dan perangkatnya cukup tangguh untuk digunakan karena mempunyai umur pakai yang relatif panjang. Bahasa yang digunakan untuk pemrograman PLC pada umumnya adalah *ladder diagram* yang juga digunakan bersamaan dengan *function block diagram*. Namun dalam pembuatan ladder diagram pada umumnya masih menggunakan cara konvensional yang menyebakan tidak adanya singkroniasi dalam pembuatan ladder. Akibat tidak ada singkronisasi adalah memperlambat dalam melanjutkan ladder diagram dan dalam melakukan analisa kesalahan. Tugas akhir ini membahas sebuah sistem yang bernama *factory automatic trainer* yang dapat memberikan gambaran dasar dari proses pembuatan *ladder diagram* yang ada di industri manufaktur. Pembuatan ladder diagram dirancang agar dapat memproses banyak benda kerja dalam satu waktu menggunakan metode *state diagram*. Hasil dari perancangan *ladder diagram* didapatkan waktu rata-rata 82,18 detik untuk sembilan benda kerja, yaitu masing-masing tiga buah benda metal, biru dan hitam. Hasil rata-rata yang didapat lebih cepat dibanding hanya memproses benda satu per satu hingga proses selesai, yaitu dengan rata-rata 26,06 detik.

Kata kunci: Industri, Perangkat kontrol, *Programmable logic controller, Ladder diagram, State diagram*.

LADDER DIAGRAM DESIGN FOR MULTIPLE OBJECTS ON FACTORY AUTOMATIC TRAINER USING STATE DIAGRAM METHOD

Andhiko Palito F-07111745000046

Supervisor: 1. Dr. Ir. Mochammad Rameli
2. Eka Iskandar, S.T., M.T.

ABSTARCT

Programmabel logic controller (PLC) is one of most used control device in manufacturing industry. The reason why using PLC is because the programming language relatifely easy to understand and has plus point in device durability. Ladder diagram and function block diagram are two of most used programming language of PLC. Nowadays, most of PLC programmer used their own logic to programming the PLC, which can cause unsynchronized in programming a ladder diagram. Without a synchronization when programming a PLC, it can slowing the programming and troubleshooting process. This final assignment will be used a system called factory automatic trainer which is able to explain and simulate basic ways to design a ladder diagram in manufacturing industry process. State diagram methode will be used to design a ladder diagram for multiple objects system. The result can reach average time about 82,18 seconds for each three metal, blue and black object sequentially. The processing time is faster than one object process at once, which is counted about 26,06 seconds in average.

***Key words**: Industry, Control device, Programmable logic controller, Ladder diagram, State diagram.*

KATA PENGANTAR

Puji syukur penulis ucapkan atas rahmat dan karunia yang telah diberikan oleh Allah Subhanahu wa Ta'ala sehingga penulis dapat menyelesaikan tugas akhir yang berjudul "Pembuatan *Diagram Ladder* untuk Banyak Proses pada *Factory Automatic Trainer* Menggunakan Metode *State Diagram*". Penulis ingin mengucapkan banyak terimakasih atas bantuan dan dukungan yang telah diberikan selama pengerjaan tugas akhir ini kepada:

1. Ayah Ahmad Fauzi dan Ibu Eni Tana selaku orang tua yang senantiasa memberikan semangat dan doa.
2. Bapak Mochammad Rameli selaku dosen pembimbing satu yang memberikan pengarahan, saran dan motivasi.
3. Bapak Eka Iskandar selaku dosen pembimbing dua yang telah memberikan ilmu dan bimbingan.
4. Seluruh rekan Lintas Jalur Teknik Sistem Pengaturan Elektro 2017 yang telah memberikan kerja sama yang baik.

Dalam penulisan tugas akhir ini dimungkinkan tidak luput dari kesalahan. Kritik dan saran sangat diharapkan untuk memperbaiki semua kesalahan. Harapan terakhir semoga tugas ini dapat bermanfaat bagi seluruh mahasiswa dan dapat dijadikan sebagai acuan di dunia insdustri.

Surabaya, Mei 2019

Andhiko Palito F
NRP 07111745000046

DAFTAR ISI

PERNYATAAN KEASLIAN v
TUGAS AKHIR v
ABSTRAK ix
ABSTARCT xi
KATA PENGANTAR xiii
DAFTAR ISI xv
DAFTAR GAMBAR xix
DAFTAR TABEL xxi
BAB 1 1
PENDAHULUAN 1
1.1. Latar Belakang 1
1.2. Permasalahan 2
1.3. Tujuan 2
1.4. Batasan Masalah 2
1.5. Metodologi 3
1.6. Sistematika 3
1.7. Relevansi 4
BAB 2 5
DASAR TEORI 5
2.1. *Factory Automatic Trainer*[1] 5
2.2. Sistem Kontrol[2] 7
2.3. *Programmable Logic Controller*[3] 8
2.3.1. Bagian PLC 9
2.3.2. Bahasa Pemrograman 11
2.4. Perangkat Masukan 12
2.4.1. *Push Button* 12
2.4.2. *Limit Switch* 13
2.4.3. *Pressure Switch* 13
2.4.4. *Photoelectric Switch* 14
2.4.5. *Proximity Switch* 14
2.5. Perangkat Keluaran 15
2.5.1. *Solenoid Valve* 15
2.5.2. Motor Listrik 15
2.5.3. *Vacuum* 16
2.6. *State Diagram*[4][5] 16
2.6.1. Penentuan Urutan Proses 17
2.6.2. Pendiskripsian Masukan dan keluaran 18

2.6.3. Pembuatan State Diagram I/O 18
2.6.4. Pembuatan Primitive Flow Table 19
2.6.5. Penyusunan *Merged Flow Table* 20
2.6.6. Pembuatan State Diagram R/O 22
2.6.7. Pembuatan *Switching Function[6]* 23
2.6.8. Pembuatan Diagram *Ladder* 25
BAB 3 27
PERANCANGAN SISTEM 27
3.1. Garis Besar Kerja Sistem 27
3.2. Rincian langkah kerja sistem 28
3.2.1. *Insert* 28
3.2.2. *Separator Metal* 28
3.2.3. *Pick and Place* 29
3.2.4. *Stopper* 29
3.2.5. *Conveyor* 1 29
3.2.6. *Conveyor* 2 30
3.2.7. *Line Movement* 30
3.3. Masukan dan Keluaran Sistem 31
3.4. Pembuatan State Diagram dan Ladder 34
3.4.1. Penentuan Urutan Proses dari Sistem 34
3.4.2. Pendiskripsian Masukan dan Keluaran dari Sistem 37
3.4.3. Pembuatan *State Diagram* I/O 38
3.4.4. Penyusunan *Primitive Flow Table* 39
3.4.5. Penyusunan *Merged Tlow Table* 40
3.4.6. Pembuatan *State Diagram* R/O 41
3.4.7. Pembuatan *Switching Function* 42
3.4.8. Pembuatan *Ladder Diagram* 43
3.5. Sub Sistem Eksternal 44
3.5.1. *Initial System* 44
3.5.2. *Lamp Indicator System* 45
3.5.3. Hitam Biru *System* 45
BAB 4 53
PENGUJIAN DAN ANALISA 53
4.1. Pengujian Sistem 53
4.1.1. Pengujian Masukan dan Keluaran 53
4.1.2. Validasi Hasil 55
4.1.3. Pengujian Digram L*adder* 56
4.1.4. Pengujian Sistem Eksternal 59
4.2. Analisa Hasil 60

4.2.1. Implementasi Metode *State Diagram* 60
4.2.2. Waktu proses 61
BAB 5 63
KESIMPULAN DAN SARAN 63
5.1. Kesimpulan 63
5.2. Saran 63
DAFTAR PUSTAKA xiii
LAMPIRAN xv
RIWAYAT HIDUP xiii

DAFTAR GAMBAR

Gambar 2.1. *Seperation modul* 5
Gambar 2.2. *Pick and place modul* 6
Gambar 2.3. *Stopper modul* 6
Gambar 2.4. *Line movement modul* 7
Gambar 2.5. Skematik dasar PLC 9
Gambar 2.6. Contoh diagram *ladder* 11
Gambar 2.7. Contoh blok fungsi 12
Gambar 2.8. Contoh teks terstruktur 12
Gambar 2.9. *Push button* 13
Gambar 2.10. *Limit switch* 13
Gambar 2.11. *Pressure switc*h 14
Gambar 2.12. *Photoelectric switch* 14
Gambar 2.13. *Proximity switch* 15
Gambar 2.14. *Solenoid valve* 15
Gambar 2.15. Motor listrik 16
Gambar 2.16. *Vacuum head* 16
Gambar 2.17. Contoh *state diagram* 17
Gambar 2.18. Contoh *state diagr*am I/O 19
Gambar 2.19. Contoh *state diagram* R/O 23
Gambar 2.20. Contoh diagram *ladder* hasil *switching function* 25
Gambar 3.1. Benda kerja (a) metal, (b) biru, (c) hitam 27
Gambar 3.2. *State diagram* I/O sub sistem 1 39
Gambar 3.3. *State diagram* R/O sub sistem 1 41
Gambar 3.4. *State diagram* R/O sub sistem 3 43
Gambar 3.5. *Diagram ladder* sub sistem 1 44
Gambar 3.6. Diagram *ladder initial system* 44
Gambar 3.7. Diagram *ladder relay* sub sistem 1 45
Gambar 3.8. *Flow chart step counter* 47
Gambar 3.9. *Flow chart step counter* 48
Gambar 3.10. *Flow chart blue step relay* 49
Gambar 3.11. *Flow chart end step counter* 50
Gambar 3.12. *Flow chart blue timer* 51
Gambar 3.13. *Flow chart blue step-relay timer* 52
Gambar 4.1. *Output* DC panel kontrol 53
Gambar 4.2. Panel modul 53
Gambar 4.3. Timer chart perencanaan 55
Gambar 4.4. Hasil simulasi 55

Gambar 4.5. Waktu proses.....62
Gambar 4.6. Ladder diagram konvensional sub proses 1 61
Gambar 4.7. Ladder diagram dengan metode sub proses 1 61

DAFTAR TABEL

Tabel 2.1. Penentuan urutan proses 18
Tabel 2.2. Pendeskripsian masukan dan keluaran..... 18
Tabel 2.3. Contoh pengisian kombinasi *input*..... 19
Tabel 2.4. Contoh pengisian stabil, tidak stabil dan *don't care* ... 20
Tabel 2.5. Contoh pengisian *output* 20
Tabel 2.6. Contoh penggabungan baris..... 21
Tabel 2.7. Contoh penentuan *relay* 21
Tabel 3.1. Masukan sistem..... 31
Tabel 3.2. Keluaran sistem..... 33
Tabel 3.3. Pembagian kerja sub sistem 34
Tabel 3.4. Deskripsi masukan dan keluaran sistem 37
Tabel 3.5. Sub sistem 1 38
Tabel 3.6. Primitive flow table sub sistem 1 39
Tabel 3.7. Merged flow table..... 40
Tabel 4.1. Pengujian masukan dan keluaran..... 54
Tabel 4.2. Hasil pengujian sistem 56
Tabel 4.3. Hasil pengujian 59
Tabel 4.4. Penggunaan relay..... 60
Tabel 4.5. Hasil pencatatan waktu tiap benda kerja dalam detik . 62

BAB 1
PENDAHULUAN

1.1. Latar Belakang

Tahun 2018 industri manufaktur sudah cukup berkembang di Indonesia, mulai dari industri makanan dan minuman, otomotif, elektronik dan beberapa jenis industri manufaktur lainnya. Industri manufaktur pada umumnya memiliki beberapa macam proses yaitu proses manual, semiotomatis, otomatis penuh dan otomasi yang sudah terhubung ke jaringan internet. Dari beberapa proses tersebut kecuali pada proses manual, proses-proses yang ada di industri manufaktur sudah menggunakan perangkat kontrol untuk menjalankan prosesnya. Perangkat kontrol tersebut dapat berupa PLC (*programmable logic controller*), DCS (*direct control system*) maupun perangkat-perangkat lainnya.

PLC merupakan salah satu perangkat proses yang sering digunakan sebagai perangkat proses dalam industri manufaktur. Alasan penggunaan PLC dikarenakan dalam pembuatan program PLC itu sendiri bahasanya lebih mudah untuk dipahami dan perangkatnya cukup tangguh untuk digunakan karena mempunyai umur pakai yang panjang. Selain alasan tersebut PLC juga dapat digunakan untuk skala proses yang kecil maupun besar dan semua proses itu dapat saling berkomunikasi, sehingga dapat menghasilkan proses yang diinginkan sesuai dengan spesifikasi dari masing-masing PLC yang digunakan.

Bahasa yang digunakan untuk memrogram PLC dinamakan dengan *ladder diagram*. Dinamakan *ladder diagram* dikerenakan memang bentuk dari bahasanya itu sendiri berbentuk *ladder* atau anak tangga. Untuk pembuatan dari *ladder diagram* itu sendiri dapat menggunakan logika dari masing-masing pembuat program (*developer*) atau dapat menggunakan metode-metode yang ada.

Alat atau perangkat yang dikontrol oleh PLC pada industri manufaktur dapat berdiri sendiri atau individu dan juga dapat berupa lini produksi yang terdiri dari beberapa mesin yang saling bekerja sama untuk mencapai tujuan tertentu. Pada dasarnya mesin yang ada di industri manufaktur memiliki proses atau prinsip kerja yang sama seperti pemilahan, pendistribusian, pemindahan dan lainnya atau secara teknisnya dapat berupa pendeteksian menggunakan sensor, penggerakan silinder penumatik menggunakan *solenoid valve*, mangaktifkan konveyor

dan proses lainnya. Jenis-jenis proses tersebut dapat disimulasikan menggunakan *factory automatic trainer* (FAT) yang sudah mencakup beberapa macam proses yang ada pada industri manufaktur. Tidak hanya mencakup beberapa macam proses, FAT juga memiliki sistem yang saling terhubung antara satu dan lainnya, sehingga juga dapat dipelajari bagaimana komunikasi antar proses tersebut.

1.2. Permasalahan

Proses pembuatan *ladder diagram* di industri manufaktur pada umumnya dilakukan hanya menggunakan logika dari *developer* masing-masing pabrik itu sendiri, sehingga tidak mempunyai aturan baku dalam pembuatan *ladder*. Penyebab masalah yang muncul dalam pembuatan *ladder* tersebut adalah tidak adanya singkronisasi antara *developer* yang satu dan lainnya. Ujung dari permasalahan ini dapat terjadi apabila *developer* pertama digantikan oleh *developer* lainnya yang dapat memperlambat dalam proses melanjutkan pemrograman *ladder*. Selain itu, masalah yang terjadi selanjutnya dapat terjadi pada saat proses *troubleshoot* jika mesin yang sudah ada di lapangan terjadi kendala, maka akan lebih lama untuk diidentifikasi jika orang yang melakukan *troubleshoot* berbeda dengan *developer* yang membuat programnya.

1.3. Tujuan

Tujuan dari tugas akhir ini adalah membuat *ladder diagram* untuk *factory automatic trainer* dengan menggunakan metode *state diagram*, sehingga konsep dari pembuatan *ladder diagram* pada *factory automatic trainer* ini dapat diterapkan di industri. Manfaat penggunaan metode dalam pembuatan *ladder diagram* juga berguna untuk menyamakan pola pikir dalam pembuatan dan pembacaan *ladder* sehingga dapat mempermudah dalam pengerjaan atau pembuatan dan menganalisa kesalahan *ladder diagram*.

1.4. Batasan Masalah

1. Tidak membahas prinsip kerja masing-masing perangkat secara detail.
2. Membahas bagaimana penerapan metode *state diagram* dalam pembuatan diagram *ladder*.
3. Tidak membandingkan dengan metode lain.
4. Hanya membahas sekilas tentang eksternal sistem.

1.5. Metodologi

Dalam pengerjaan tugas akhir ini, proses yang pertama kali dilakukan adalah melakukan studi literatur karena dalam tugas akhir ini akan diterapkan sebuah metode untuk merumuskan diagram *ladder* dari sistem itu sendiri. Studi literatur bertujuan untuk mendapatkan pemahaman bagaimana proses pengerjaan dari metode *state diagram* hingga mencapai tujuan akhir yaitu mendapatkan rumusan dari diagram *ladder*.

Penerapan metode *state diagram* dimulai dengan penentuan masukan dan keluaran sistem yang dilanjut dengan menentukan state-state dari masing-masing sub-sistem. Dari state yang sudah ada kemudian bisa diperoleh *primitive flow table* yang kemudian dilakukan penggabungan. Dalam penggabungan ini dibuatlah sebuah tabel yang bernama *merged flow table* dengan menambahkan kolom untuk penentuan jumlah dan relay apa saja yang akan aktif untuk masing-masing state. Dari *merged flow table* itu kemudian dapat ditentukan kembali state baru yang berguna untuk menentukan *switching function*-nya seperti apa untuk kebutuhan pembuatan *ladder diagram*.

Ladder diagram yang sudah selesai dirumuskan akan dilakukan uji coba, jika *ladder diagram* yang dibuat masih terdapat kesalahan maka akan dilakukan pengkajian ulang dari *ladder diagram* dan metode *state diagram* tersebut. Diagram *ladder* yang sudah dianggap berhasil akan dilakukan pengujian berkala hingga sudah dapat diyakini sistem tersebut sudah berjalan dengan lancar tanpa ada kesalahan. Tahap terakhir dari pengerjaan tugas akhir ini sendiri adalah membuat pembukuan untuk laporan dari proses dan hasil dari pembuatan tugas akhir ini.

1.6. Sistematika

Dalam penulisan tugas akhir akan dibagi menjadi lima bab. Penjelasan dari masing-masing bab adalah sebagai berikut:

Bab I Pendahuluan

Pendahuluann berisikan latar belakang, permasalahan, tujuan dan manfaat, batasan masalah, metodologi, sistematika dan relevansi.

Bab II Dasar Teori

Dasar teori berisikan teori-teori pendukung dalam pengerjaan tugas akhir. Teori pendukung berupa dasar-dasar dari *factory automatic trainer* hingga penjelasan dari tatacara pembuatan ladder menggunakan metode *state diagram*.

Bab III Perancangan Sistem

Perancangan sistem akan membahas pembagian sistem *factory automatic trainer* ke sistem yang lebih kecil dan bagaimana proses dari masing-masing sub-sistem. Serta membahas bagaimana pengaplikasian metode *state diagram* hingga menjadi sebuah diagram *ladder*.

Bab IV Hasil dan Analisa

Hasil dan analisa membahas hasil dari penerapan diagram ladder dan membandingkan dengan tugas akhir yang sebelumnya.

Bab V Penutup

Penutup berisikan kesimpulan dan saran dari hasil pembuatan tugas akhir.

1.7. Relevansi

Perancangan diagram *ladder* yang lebih terstruktur dan dapat dipertanggungjawabkan sangat dibutuhkan dalam pembuatan perangkat otomasi. Dengan menggunakan metode *state diagram* dalam pembuatan diagram *ladder*, pembuatan diharapkan akan menjadi lebih terarah dan pada saat terjadi masalah akan lebih mudah diidentifikasi. Hasil tugas akhir ini diharapkan dapat dijadikan sebagai acuan atau referensi dalam pembuatan diagram *ladder*, baik untuk proses pembelajaran maupun untuk proses di industri.

BAB 2
DASAR TEORI

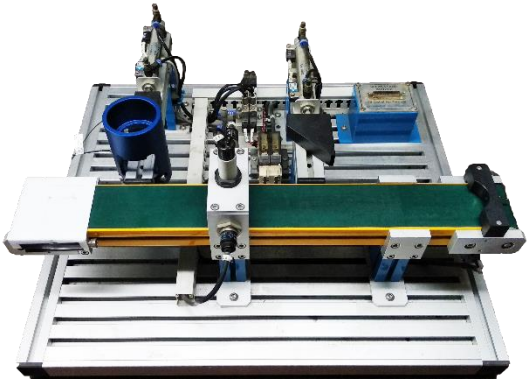
2.1. *Factory Automatic Trainer*[1]

Factory automatic trainer atau FAT merupakan sebuah gabungan sistem-sistem yang saling bekerja sama. FAT berfungsi sebagai media pembelajaran atau media simulasi dari proses-proses dasar yang ada di industri manufaktur. FAT terdiri dari modul-modul yang berisikan perangkat keras masukan dan keluaran, dan perangkat proses atau kontrol yang berupa *programmable logic controller*.

FAT memiliki empat buah modul yang saling bekerja sama. Berikut modul-modul yang ada pada FAT:

1. *Separation modul*

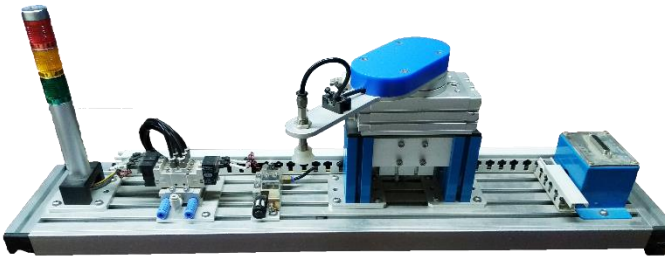
   Separation modul memiliki tiga buah fungsi utama, yaitu memasukan benda ke dalam proses, mendeteksi jenis benda dan memisahkan jenis benda kerja.



Gambar 2.1. *Seperation modul*

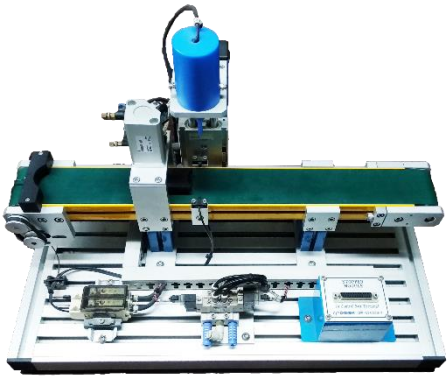
2. *Pick and place modul*

   Pick and place atau PNP memiliki tugas yang sederhana yaitu memindahkan benda kerja dari konveyor 1 (*separation*) ke konveyor 2 (*stopper*). Pemindahan benda kerja pada PNP menggunakan *vacuum* sebagai pengangkat benda kerja dan silinder untuk memindahkan posisi benda kerja.



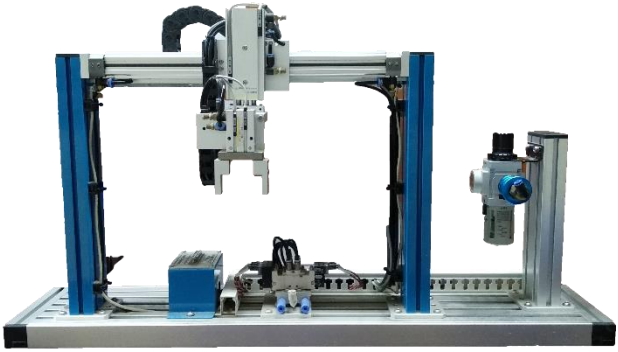
Gambar 2.2. *Pick and place modul*

3. *Stopper modul*
   Sesuai dengan namanya *stopper*, *stopper* modul memiliki tugas untuk menghentikan pergerakan benda yang kemudian dilakukan proses *drilling* atau pengeboran benda kerja.



Gambar 2.3. *Stopper modul*

4. *Line movement modul*
   Modul terakhir adalah *line movement* modul memiliki fungsi yang sama dengan PNP yaitu memindahkan benda kerja. Perbedaan dari *line movement* dan PNP tampak dari jenis *end effector* yang digunakan. Ketika PNP menggunakan *vacuum* sebagai *end effector*, *line movement* menggunakan mekanisme *grip* sebagai *end effector*.



Gambar 2.4. *Line movement modul*

2.2. Sistem Kontrol[2]

Definisi sistem secara bahasa adalah suatu kesatuan yang terdiri dari komponen atau elemen yang dihubungkan bersama untuk memudahkan aliran informasi, materi atau energi untuk mencapai suatu tujuan. Definisi sederhana dari sistem adalah kumpulan komponen-komponen yang saling bekerja sama untuk mencapai suatu tujuan tertentu. Kontrol biasanya diartikan sebagai pengawasan, pemeriksaan atau pengendalian. Dari pengertian sistem dan kontrol tersebut, maka sistem kontrol adalah suatu kumpulan komponen-komponen yang saling bekerja sama untuk mencapai suatu tujuan tertentu yang dapat mengendalikan atau mengatur diri sendiri atau sistem lainnya.

Sistem kendali terdiri dari sub sistem dan proses yang disusun untuk mendapatkan keluaran dan kinerja yang diinginkan dari masukan yang diberikan. Secara umum sistem kontrol dapat diklasifikasikan sebagai berikut:

1. Sistem kontrol manual dan otomatis
2. Sistem lingkar terbuka (*open loop*) dan lingkar tertutup (*close loop*)
3. Sistem kontrol kontinyu dan diskrit
4. Sistem menurut sumber penggerak: elektrik, mekanik, penumatik dan hidraulik.

Sistem kontrol manual adalah pengontrol yang dilakukan oleh manusia yang bertindak sebagai operator. Sistem kontrol otomatis adalah pengontrolan yang dilakukan oleh peralatan yang berkerja secara otomatis dan operasi dibawah pengawasan manusia.

Sistem kontrol lingkaran terbuka adalah suatu sistem yang keluarannya tidak mempunyai pengaruh terhadap aksi kontrol. Pada sistem ini keluaran dari sistem tidak digunakan kembali sebagai pembanding dengan masukan yang sudah diatur.

Berikut ciri-ciri sistem lingkaran terbuka:

1. Sederhana
2. Harga cenderung lebih murah
3. Kurang akurat karena tidak terdapat koreksi terhadap kesalahan

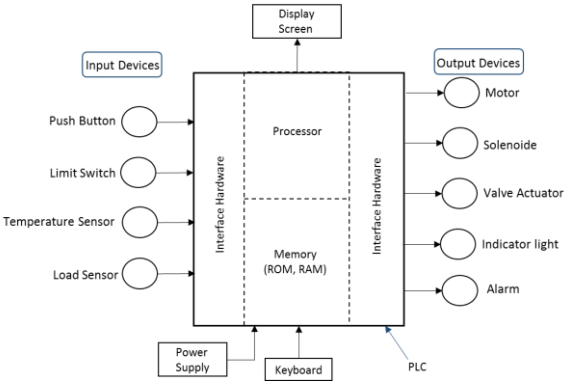
Sistem kontrol lingkaran tertutup adalah sistem kontrol yang sinyal keluarannya digunakan kembali sebagai pembanding dengan masukan yang sudah diatur. Tujuannya adalah untuk mendapatkan keluaran yang sesuai dengan yang diinginkan.

Berikut adalah komponen pada sistem kendali lingkaran tertutup:

1. Masukan, merupakan rangsangan yang diberikan kepada sistem, dapat berupa harga atau kisaran harga yang diinginkan. Masukan dapat berupa nilai digital atau analog.
2. Keluaran, merupakan tanggapan dari sistem. Jenis dari keluaran ini bermacam-macam tergantung dari jenis keluaran yang digunakan.
3. Alat kontrol, merupakan perangkat yang bertugas untuk mengontrol masukan yang diterima dan menghasilkan keluaran yang telah diproses.
4. Pendeteksi kesalahan, merupakan alat yang memberikan umpan balik dari keluaran. Pendeteksi kesalahan ini kemudian digunakan kembali sebagai perangkat masukan.
5. Gangguan, merupakan sinyal-sinyal tambahan dari luar sistem yang tidak diinginkan.

2.3. *Programmable Logic Controller*[3]

Programmabel logic controller (PLC) adalah sistem berbasis mikroprosesor yang menggunakan memori yang dapat diprogram untuk menyimpan instruksi-instruksi dan mengimplementasikan fungsi-fungsi logika, pengurutan, pewaktuan, pencacahan dan aritmatika untuk mengontrol mesin-mesin dan proses, serta dirancang untuk dioperasikan oleh teknisi yang mungkin memiliki kemampuan dan pengetahuan terbatas mengenai komputer dan bahasa pemrograman.



Gambar 2.5. Skematik dasar PLC

PLC pada dasarnya adalah sistem komputer digital yang mampu melaksanakan proses kerja yang kompleks. Penggunaan PLC dilingkungan industri terhitung lebih praktis dan efisien daripada sistem kontrol konvensional. Saat terjadi perubahan proses, pengguna hanya perlu mengubah instruksi-instruksi program yang dimasukkan kedalamnya tanpa perlu perubahan pengkabelan sistem. Selain itu dalam pengkabelan PLC mudah untuk dimodifikasi. PLC memiliki kemampuan untuk mengontrol sistem proses atau mesin yang kompleks.

2.3.1. Bagian PLC

Bagian-bagian dari sebuah sistem PLC terdiri dari lima komponen dasar yaitu unit prosesor atau CPU, memori, catu daya, modul masukan dan keluaran dan piranti pemrograman.

1. Unit prosesor

   Unit prosesor atau CPU (*central processing unit*) merupakan bagian yang mengandung komponen mikroprosesor. Berfungsi untuk mengontrol, mengolah, menyimpan dan mengawasi semua kegiatan pengoperasian dalam sebuah PLC, atau disebut juga otak dari PLC. CPU memproses semua sinyal masukan serta menjalankan aksi-aksi kontrol sesuai dengan program yang tersimpan didalam memori.
2. Memori

   Memori merupakan bagian dari prosesor sebagai penyimpan program dan data yang diakses oleh prosesor selama proses kerja berlangsung. Data yang tersimpan dalam bentuk biner yang

merupakan penerjemahaan *ladder diagram* yang telah dibuat oleh pengguna sebelumnya. Selain menyimpan program, memori menyimpan lokasi-lokasi dimana hasil perhitungan dapat disimpan didalamnya.

3. Unit catu daya
   Merupakan sumber daya yang masuk kedalam PLC. Berfungsi mengubah tegangan listrik bolak-balik dari PLN menjadi tegangan listrik searah yang sesuai dengan kebutuhan CPU atau modul masukan dan keluaran PLC.
4. Modul masukan dan keluaran
   Perangkat masukan dapat berupa saklar, sensor, tombol dan lainnya. Sedangkan perangkat keluaran dapat berupa motor, solenoid, lampu dan lainnya. Setiap perangkat masukan dan keluaran ini memiliki alamat tersendiri dalam sistem sehingga memudahkan dalam pembuatan program PLC.
   Terdapat 2 jenis tipe modul keluaran pada PLC, yaitu tipe *relay* dan transistor.
   a. Keluaran tipe *relay*
      Sinyal keluaran PLC digunakan untuk mengoperasikan sebuah *relay* sehingga dapat melewatkan arus sampai beberapa ampere di suatu tegangan eksternal. Penggunaan *relay* ini selain untuk mengalirkan arus yang lebih besar juga berguna untuk mengisolasi PLC dari rangkaian eksternal. Tipe *relay* bisa digunakan untuk pengsaklaran arus DC maupun AC, serta memiliki ketahanan yang lebih baik terhadap lonjatan arus. Namun keluaran tipe *relay* memiliki kekurangan dalam waktu *switching* yang relatif lambah dibanding tipe transistor.
   b. Keluaran tipe transistor
      Keluaran tipe transistor memiliki kecepatan *switching* yang lebih tinggi. Namun tipe ini hanya bisa digunakan untuk pensaklaran arus DC saja dan mudah rusak apabila arus lebih serta tegangan balik yang tinggi. Untuk itu diberikan perlindungan berupa sikring atau proteksi elektronik *build-in* yang terintegrasi.

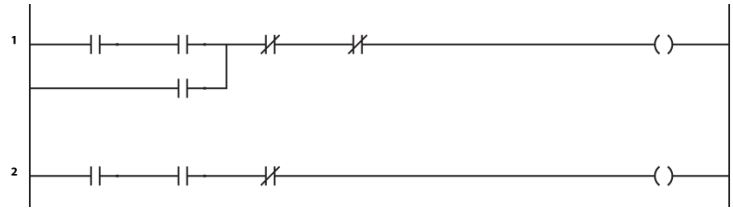
   Selain data digital, PLC juga dapat menerima atau menyampaikan sinyal analog pada masukan dan keluarnnya. Namun harus menggunakan modul khusus yaitu modul analog itu sendiri. Perbedaannya yaitu terdapat fungsi ADC (*analog*

digital converter) dari modul untuk mengkonversikan data analog menjadi data digital yang bisa diolah oleh prosesor PLC.

5. Piranti pemrograman
   Merupakan piranti yang digunakan untuk memasukan program kedalam memori prosesor. Piranti pemrograman PLC dapat dilakukan dengan 2 cara yaitu menggunakan *console* maupun menggunakan komputer tergantung dari jenis PLC yang digunakan. Bahasa pemrograman yang digunakan pada PLC dapat berupa diagram *ladder*, diagram blok fungsi (*function block*) atau menggunakan teks terstuktur (*skript*).

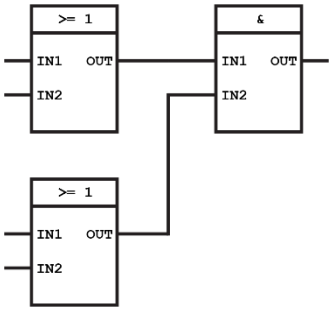
2.3.2. Bahasa Pemrograman

Ada beberapa bahasa yang digunakan dalam memrogram PLC, yaitu diagram *ladder*, blok fungsi dan teks terstruktur. Bahasa pemrograman diagram *ladder* terdiri dari kontak dan *relay* yang diatur sedemikian rupa dengan fungsi logika tertentu untuk menghasilkan fungsi tertentu. Dinamakan diagam *ladder* atau diagram anak tangga, karena bentuk dari diagram tersebut memang berbentuk anak tangga.



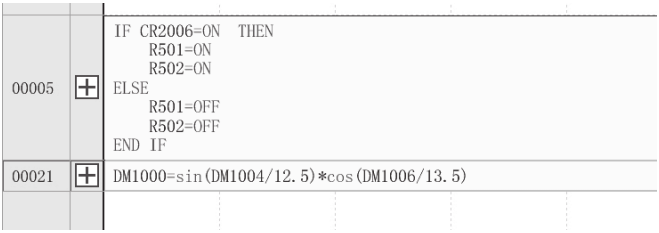
Gambar 2.6. Contoh diagram *ladder*

Diagram blok fungsi merupakan penyederhanaan dari fungsi logika yang berbentuk blok. Penggunaan blok fungsi ini digunakan bersamaan dengan diagram *ladder*. Banyak macam fungsi yang bisa digunakan seperti penghitung waktu mundur, perhitungan metematika dan fungsi-fungsi lainnya.



Gambar 2.7. Contoh blok fungsi

Teks terstruktur merupakan bahasa pemrograman PLC yang memiliki basis yang hampir sama dengan bahasa pemrograman seperti PHP, Phyton, C dan lainnya.



Gambar 2.8. Contoh teks terstruktur

2.4. Perangkat Masukan

2.4.1. *Push Button*

Push button merupakan perangkat keras yang dapat memberikan sinyal 1 pada saat kondisi ditekan. *Push button* ada yang kontaknya berupa *normally open* saja atau ada juga yang di lengkapi dengan kontak *normally close*. Selain itu *push button* juga ada yang memiliki mekasisme mengunci kondisi, seperti pada saat ditekan sekali akan mengunci ke posisi ditekan dan akan kembali ke kondisi awal jika ditekan sekali lagi.

Gambar 2.9. *Push button*

2.4.2. *Limit Switch*

Limit switch memiliki prinsip kerja yang sama denga *push button*, hanya saja *limit switch* bukan ditekan dengan tangan manusia melainkan detekan atau akan aktif pada saat tesentuh benda tertentu. Pada *limit switch* terdapat tuas yang jika tersentuh oleh benda tertentu akan menyebabkan penekan yang kemudian mengubah kondisi dari kontak *limit switch* itu sendiri. Untuk jenis kontaknya sama dengan *push button*.

Gambar 2.10. *Limit switch*

2.4.3. *Pressure Switch*

Pressure switch juga memiliki prinsip kerja yang sama dengan *push button*, namun pada *pressure switch* kontak akan berubah kondisi jika mekanisme yang ada pada *pressure switch* mendapat tekanan dengan nilai tertentu. Jika tekanan yang diterima memenuhi syarat minimal dari *pressure switch*, maka akan terjadi perubahan kondisi kontak. Untuk jenis kontaknya juga sama dengan *push button*.

Gambar 2.11. *Pressure switc*h

2.4.4. *Photoelectric Switch*

Photoelectric switch memiliki dua komponen utama yaitu *emitter* dan *receiver*. Prinsip kerjanya yaitu *emitter* akan memancarkan cahaya *infrared* atau sejenisnya yang kemudia akan diterima oleh *receiver*. Pada saat *receiver* bisa mendeteksi sinyal dari *emitter* maka kondisi *switch* ini akan aktif. Sebaliknya jika *receiver* tidak dapat menerima sinyal dari *emitter* dikarenakan terhalang suatu benda, maka kondisi switch akan tidak aktif.

Gambar 2.12. *Photoelectric switch*

2.4.5. *Proximity Switch*

Proximity switch merupakan sensor yang berfungsi untuk mendeteksi keberadaan suatu objek tanpa harus ada kontak fisik dengan *switch* tersebut. *Proximity switch* memancarkan gelombang elektromagnetik atau radiasi elektromagnetik yang jika benda melewati area gelombang tersebut maka akan menyebabkan perubahan pada medan elektromagnetik yang kemudian akan memberikan sinyal aktif.

Proximity switch dibagi menjadi dua yaitu induktif dan kapasistif. Jenis induktif pada dasarnya memanfaatkan perubahan induktansi akibat adanya bahan magnetik yang mendekat. Sehingga sensor induktif hanya mampu mendeteksi benda-benda logam saja. Jenis sensor konduktif

prinsip dasarnya yaitu adanya perubahan variabel bukan listrik yang menyebabkan perubahan kapasitansi dari kapasitor sehingga keadaan ini dapat dibaca oleh sensor. Perubahan kapasitasi dapat disebabkan oleh perubahan luas permukaan atau jarak dalam permitivitas dielektrik relatif. Semakin luar atau semakin dekat jarak benda dari sensor maka semakin besar juga perubahan kapasitansi dari sensor. Hal ini menyebakan jenis *proximity* ini dapat membaca benda logam maupun bukan logam.

Gambar 2.13. *Proximity switch*

2.5. Perangkat Keluaran

2.5.1. *Solenoid Valve*

Solenoid terdiri dari kumparan dan inti besi bergerak. Ketika kumparan diberi arus, inti besi akan bergerak sehingga menyebabkan terbukanya lubang udara yang semula tertutup. *Solenoid valve* ini sendiri memiliki beberapa jenis tipe yang ditentukan berdasarkan jenis katup dari masing-masing *valve*.

Gambar 2.14. *Solenoid valve*

2.5.2. Motor Listrik

Motor listrik yang digunakan pada perangkat keluaran PLC pada umumnya memiliki sumber arus yang terpisah dikarenakan daya yang dikeluarkan oleh keluaran PLC belum tentu kuat untuk menggerakkan

motor listrik. Biasanya PLC hanya mengirimkan sinyal aktif ke motor listrik melalui driver motor sebagai tanda motor aktif. Motor listrik yang digunakan bisa berupa motor DC maupun AC. Ketika motor DC menggunakan *driver* sebagai penghubung ke PLC, motor AC menggunakan *inverter* untuk berkomunikasi dengan PLC.

Gambar 2.15. Motor listrik

2.5.3. *Vacuum*

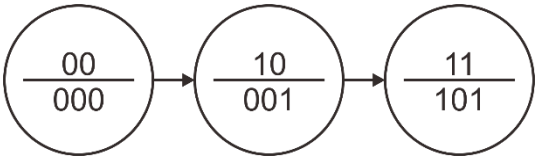
Vacuum pada dasarnya juga menggunakan solenoid untuk membuka katup udara yang akan disalurkan. Perbedaannya terdapat pada mekanisme akhir dari *vacuum* itu sendiri. Vacuum memanfaatkan efek Bernoulli untuk menghirup benda disekitarnya. Efek Bernoulli menyatakan bahwa udara yang memiliki kecepatan tinggi akan memiliki tekanan yang rendah. Sebaliknya udara yang memiliki kecepatan yang rendah akan memiliki tekanan yang tinggi.

Gambar 2.16. *Vacuum head*

2.6. *State Diagram*[4][5]

State diagram atau bisa disebut juga sebagai *flow table* merupakan sebuah grafik yang merepresentasikan suatu kondisi atau sebuah state yang disajikan dalam bentuk lingkaran. State yang ada didalam lingkaran tersebut berisikan informasi masukan dan keluaran dari suatu sistem.

Informasi masukan dan keluaran tersebut dituliskan dalam bentuk bilangan biner untuk membuat tampilan dari state agar lebih sederhana.



Gambar 2.17. Contoh *state diagram*

Ada beberapa tahapan dalam pembuatan *state diagram*. Berikut tahapan dalam pembuatan *state diagram*:

1. Penentuan urutan proses dari sistem
2. Pendiskripsian masukan dan keluaran dari sistem
3. Pembuatan *state diagram* I/O
4. Penyusunan *primitive flow table*
5. Penyusunan *merged flow table*
6. Pembuatan *state diagram* R/O
7. Pembuatan *switching function* (langkah tambahan)
8. Pembuatan *ladder diagram* (langkah tambahan)

Untuk tahapan nomor tujuh dan delapan merupakan tahapan tambahan agar bisa diaplikasikan ke PLC.

2.6.1. Penentuan Urutan Proses

Langkah pertama dalam menggunakan metode state diagram adalah menentukan urutan proses dari suatu sistem. Dalam langkah ini yang perlu diperhatikan adalah bagaimana sistem ini dapat bekerja secara normal dari satu langkah ke langkah selanjutnya sesuai dengan yang diinginkan. Selain itu, pada langkah ini juga ditentukan masukan dan keluaran yang aktif atau tidak aktif pada masing-masing langkah. Jika di suatu langkah memiliki kondisi masukan dan keluaran yang sama dengan langkah sebelumnya, maka dianggap sebagai satu state yang sama. Tujuannya adalah untuk dapat mempermudah melanjutkan ke proses pembuatan state diagram sistem.

Tabel 2.1. Penentuan urutan proses

State	Masukan State	Keluaran State	Syarat Perlu
S	Start on	Motor1 on	Kondisi diam
S+1	Input1 on	Output1 on	-
...	...	...	...
Sn	End on	Motor1 off	-

2.6.2. Pendiskripsian Masukan dan keluaran

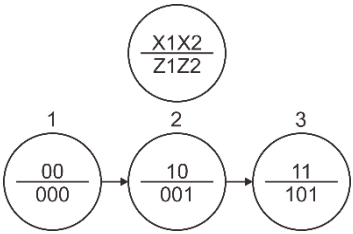
Langkah kedua, masukan dan keluaran yang sudah ditentukan pada langkah sebelumnya diberikan penamaan masing-masing. Pada umumnya penamaan untuk masukan sistem diberi nama X1 hingga Xn, sesuai jumlah masukan yang digunakan. Begitu juga dengan keluaran sistem, diberi penamaan dari Z1 hingga Zn. Tujuan dari penamaan masukan dan keluaran hanyalah untuk mempersingkat nama dari masukan dan keluaran tersebut.

Tabel 2.2. Pendeskripsian masukan dan keluaran

No	Nama Masukan	Kode Masukan
p	Input1	X1
p+1	Input1	X2
...	...	...
q	Input-q	Xq

2.6.3. Pembuatan State Diagram I/O

Langkah ketiga adalah penentuan state diagram dari sistem berdasarkan masukan dan keluaran yang sudah ditentukan dan dideskripsikan sebelumnya. Pada langkah ini dimulai dengan membuat state-state yang terdiri dari masukan dan keluaran sistem sesuai dengan urutan proses yang telah ditentukan ditahap pertama. Penulisan masukan dan keluaran yang ada didalam state akan bernilai “1” jika aktif dan sebaliknya akan bernilai “0” jika tidak aktif karena menggunakan bilangan biner. Masing-masing state tersebut akan dihubungkan dengan arah panah sebagai penanda ke arah mana state itu dapat lanjut atau berpindah kondisi dan tidak lupa juga untuk memberikan penomoran untuk masing-masing state tersebut.



Gambar 2.18. Contoh *state diagra*m I/O

Tampak pada gambar 2.18., keterangan masukan dari state terdapat dibagian atas. Sedangkan untuk keterangan keluaran state terdapat dibagian bawah.

2.6.4. Pembuatan Primitive Flow Table

Pembuatan *primitive flow table* bertujuan untuk menerjemahkan state diagram ke bentuk tabel. Tabel tesebut terdapat tiga kelompok kolom yaitu *row* atau urutan state, kombinasi *input* atau masukan sistem dan *output* atau keluaran sistem. Kolom-kolom tersebut kemudian diisi sesuati dengan kondisi state yang telah dibuat sebelumnya.

Langkah pertamanya adalah mengisi *row* atau urutan berdasarkan urutan penomoran state yang telah ditentukan sebanyak state yang ada pada *state diagram* I/O. Pada kolom *input* dibuat sebanyak kombinasi bit *input* yang ada. Misal pada *state diagram* I/O terdapat empat macam kombinasi *input* yaitu 1000, 0100, 0010 dan 0001, maka kemudian dibuat empat kolom yang masing-masing kolom diisi dengan kombinasi tersebut. Akan ada kemungkinan kondisi dimana kombinasi *input* disatu state sama dengan state lainnya. Dalam kondisi ini cukup membuat satu kolom kombinasi *input*. Contoh pengisian dapat dilihat di tabel 2.3.

Tabel 2.3. Contoh pengisian kombinasi *input*

ROW	INPUT				OUTPUT	
	1000	0100	0010	0001	...	...
S	...	...	...	...	...	...
S+1	...	...	...	...	...	...

Langkah selanjutnya adalah pengisian state stabil, state tidak stabil dan *don't care* untuk masing-masing *row*. State stabil adalah kondisi state yang aktif saat itu. Misal state yang aktif saat *row* S adalah state S dengan kondisi *input* 100, maka ditulislah state S di sel tabel yang sejajar dengan

baris *row* dan kolom *input* tersebut. State tidak stabil adalah state yang akan dituju oleh state aktif saat itu berdasarkan arah panah yang menghubungkan state-state tersebut. Misal state S memiliki arah panah ke state S+1 yang memiliki *input* 010. Maka ditulislah state S+1 disel tabel yang sejajar dengan baris *row* S dan kolom 010. Perbedaannya, state stabil biasanya ditulis dengan huruf cetak tebal sedangkan state tidak stabil tidak. Terakhir kondisi *don't care* adalah kondisi dimana tidak ada pengaruh apapun dari state yang aktif saat itu, yang di dalam tabel diberi tanda garis (-). Contoh pengisian dapat dilihat di tabel 2.4.

Tabel 2.4. Contoh pengisian stabil, tidak stabil dan *don't care*

ROW	INPUT				OUTPUT	
	1000	0100	0010	0001	...	...
S	**S**	S+1	-	-	...	...
S+1	-	**S+1**	-	-	...	...

Terakhir adalah pengisian kolom *output*. Pada tabel ini dibuat kolom sebanyak *output* yang digunakan. Dalam penulisan keterangan biasanya digunakan huruf Z1 hingga Zn, sesuai dengan jumlah *output* yang sudah ditentukan diawal proses. Sel-sel *output* tersebut diisi dengan bilangan biner yang menandakan "1" sebagai tanda aktif dan "0" sebagai tanda tidak aktif. Misal untuk state S dengan konfigurasi *output* $Z_1Z_2Z_3$ memiliki nilai *output* 101, maka pada sel-sel *output* yang sejajar dengan baris *row* S diisi nilai Z_1=1, Z_2=0 dan Z_3=1. Begitu selanjutnya hingga *output row* S+k.

Tabel 2.5. Contoh pengisian *output*

ROW	INPUT				OUTPUT		
	1000	0100	0010	0001	Z1	Z2	Z3
S	**S**	S+1	-	-	1	0	1
S+1	-	**S+1**	S+2	-	1	1	0
S+2			**S+2**	-	1	1	0

2.6.5. Penyusunan *Merged Flow Table*

Merged flow table adalah bentuk penyederhanaan dari *output* dan merupakan tabel untuk penentuan jumlah dan susunan *relay*. Penyederhanaan dari *output* dapat dilakukan jika kondisi *output* yang ada pada *primitive flow table* sama. Dibanding dengan penggunaan *relay* secara normal, ketika menggunakan *merged flow table* jumlah *relay* dapat diperkecil.

Berikut adalah aturan yang harus dipenuhi pada saat menggabungkan baris *merged flow table*:

1. Apabila pada saat penggabungan terdapat state stabil, tidak stabil dan atau kondisi *don't care* di dalam sel yang sama, maka sel penggabungan diisi dengan state stabil.
2. Apabila pada saat penggabungan terdapat state tidak stabil dan kondisi *don't care* di dalam sel yang sama, maka sel penggabungan diisi dengan state tidak stabil.
3. Apabila pada saat penggabungkan hanya ada kondisi *don't care*, maka tetap diisi dengan *don't care*.

Misal pada *row* S+2 memiliki kondisi *output* yang sama dengan *row* S+1, maka kedua baris tersebut dapat digabungkan dengan mengikuti aturan penggabungan tersebut.

Tabel 2.6. Contoh penggabungan baris

ROW	INPUT				OUTPUT		
	1000	0100	0010	0001	Z1	Z2	Z3
S	**S**	S+1	-	-	1	0	1
S+1,S+2	-	**S+1**	**S+2**	-	1	1	0

Selanjutnya adalah penentuan julah *relay* yang akan digunakan. *Relay* yang ada pada metode *state diagram* menggunakan teori RS *flip flop*. Alasan penggunaan teori RS *flip flop* dikarenakan satu *relay* dapat menghasil dua buah kombinasi bit yaitu "1" dan "0", dan jika menggunakan dua *relay* dapat menghasilkan empat kombinasi bit. Dari teori RS *flip flop* tersebut bisa didapat rumusan untuk menentukan jumlah kombinasi bit dengan jumlah *relay* tertentu. Rumusan tersebut adalah 2^x dimana x adalah jumlah *relay*. Jika y adalah jumlah baris dari *row*, maka jumlah *relay* yang harus dipenuhi adalah $2^x \geq y$, sehingga dapat memenuhi kebutuhan jumlah *row*.

Tabel 2.7. Contoh penentuan *relay*

ROW	INPUT				OUTPUT			RELAY	
	1000	0100	0010	0001	Z1	Z2	Z3	Y1	Y2
S	**S**	S+1	-	-	1	0	1	1	0
S+1,S+2	-	**S+1**	**S+2**	S+3	1	1	0	1	1
S+3	-	-	S+4	**S+3**	1	0	0	0	1
S+4	-	-	**S+4**	-	0	0	0	0	0

Dalam penentuan kombinasi *relay* harus mengikuti sebuah sistem yang bernama *reflected binary code* atau biasa dikenal dengan *reflected binary* atau *grey code*. Secara singkat, *grey code* adalah suatu sistem penomoran biner yang hanya boleh berubah satu bit tiap keadaannya. Tujuan dalam penggunaan *grey code* adalah untuk menghindari kesalahan dalam pembacaan *relay* pada state diagram. Misal suatu kombinasi *relay* 110 akan berpindah kondisi menjadi 101. Secara kasat mata mungkin akan tampak langsung berpindah dari 110 ke 101, namun jika waktu diperlambat tidak seperti itu. Ketika perpindahan 110 ke 101 akan ada kondisi dimana perubahannya dimulai dari 110, 100, baru kemudian menjadi 101, atau dimulai dari 110, 111, baru kemudian 101. Dengan menggunakan sistem *grey code* tersebut, kondisi state sistem harus berubah dari 110 ke 100 untuk mencegah kesalahan.

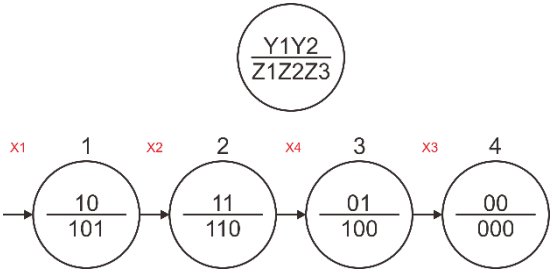
2.6.6. Pembuatan State Diagram R/O

Langkat terakhir dalam metode state diagram adalah pembuatan *state diagram* berdasarkan *relay* dan *output* (R/O). perbedaan yang tampak antara state I/O dan R/O adalah jumlah statenya jika mengalami penggabungan pada *merged flow table*, dan posisi keterangan *input* yang terletak dipanah penghubung state karena digantikan oleh keterangan *relay* sistem. Input yang digunakan tidak semuanya, hanya *input* yang memiliki pengaruh dalam perubahan state saja atau bisa disebut sebagai *trigger* dari perubahan state.

Perubahan state I/O ke R/O mengartikan bahwa setiap perubahan *output* akan bergantung kepada perubahan kombinasi *relay*. Tujuan dari pemosisian *input* dipanah yang menuju state adalah sebagai tanda bahwa perubahan kondisi *relay* akan dipengaruhi *input* yang menuju state tersebut.

Cara pembuatan state diagram R/O sangat mudah. Berikut tahapan pembuatan state diagram R/O:

1. Masukkan kombinasi *relay* dan kombinasi *output* ke state yang sama.
2. Penentuan arah panah dibuat sesuai dengan perubahan kondisi stabil ke tidak stabil yang ada di tabel *merged flow table*.
3. Memberikan keterangan *input* yang menjadi *trigger* dipanah yang menuju state yang akan di-*trigger*.



Gambar 2.19. Contoh *state diagram* R/O

2.6.7. Pembuatan *Switching Function[6]*

Switching function adalah persamaan yang digunakan untuk membuat diagram *ladder*. Pembuatan *switching function* dibagi menjadi dua, yaitu *swicthing function* untuk *relay* dan untuk *output*. Untuk *input* biasanya akan digunakan huruf X1 hingga Xn, *relay* menggunakan huruf Y1 hingga Yn dan *output* akan menggunakan huruf Z1 hingga Zn, masing-masing tergantung berapa jumlah yang digunakan. *Relay* dan *output* memiliki dua kondisi yaitu *set* yang memiliki perubahan nilai dari "0" ke "1" dan *reset* memiliki perubahan nilai dari "1" ke "0".

Ada beberapa tahapan dalam pembuatan *switching function relay*, sebagai berikut:

1. Menentukan *input* yang mengubah *relay*-m menjadi kondisi *set.*
2. Menentukan *input* yang mengubah *relay*-m menjadi kondisi *reset*, kemudian mengganti kondisi *input*. Misal *input*-n pada sistem adalah X1, maka di ganti kondisi menjadi *not*-X1.
3. Menggabungkan semua *input* yang mengubah *relay*-m menjadi kondisi *set* dengan logika OR (jika ada).
4. Menambahkan *relay*-m ke gabungan *input* sebagai *safe holding relay*-m.
5. Menggabungkan semua *input* yang mengubah *relay*-m menjadi kondisi *set* (jika ada).

Saat penulisan *switching function*, logika AND ditulis dengan perkalian, dengankan untuk logika OR ditulis dengan penjumlahan. Dalam penggabungan *input* untuk kondisi *reset* memiliki syarat sebagai berikut:

1. Jika *input*-n yang me-*reset relay*-m aktif kembali atau tetap aktif ketika *relay*-m kembali ke kondisi *set* hingga di *reset* oleh *input* lainnya, maka penggabungan *input* menggunakan logika OR.
2. Jika *input*-n yang me-*reset relay*-m tidak aktif kembali ketika *relay*-m kembali ke kondisi *set* hingga di *reset* oleh *input* lainnya, maka penggabungan *input* menggunakan logika AND.

Berikut contoh *switching function relay*:

$$Y1 = (X1 + Y1) * \overline{X4}$$
$$Y2 = (X2 + Y2) * \overline{X3}$$

Ada kondisi dimana perlu ditambahkan suatu input yang menggunakan logikan OR dengan *input reset*. Kondisi tersebut adalah ketika *input*-n yang akan me-*reset relay*-m aktif dan tidak aktif saat kondisi *relay*-m belum seharusnya *reset*. Pada kondisi ini ditambahkan *input* lain yang sama-sama aktif dengan *relay*-m, namun sudah mati pada saat *input*-n akan me-*reset relay*-m. pada kondisi ini boleh menambahkan beberapa *input* tambahan dengan tujuan *relay*-m tetap aktif saat *relay*-m diharuskan aktif.

Setelah membuat *switching function relay*, kemudian dilanjutkan dengan pembuatan *switching function output*. Berikut tahapan pembuatan *switching function output*:

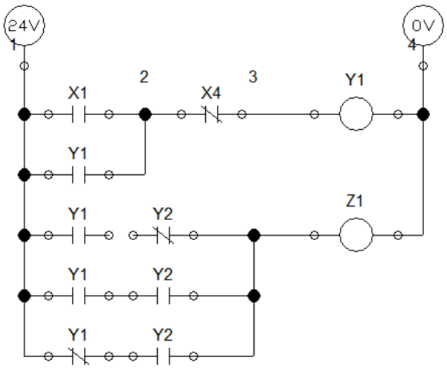
1. Menentukan saat state keberapa saja *output*-n aktif.
2. Mencari tau kombinasi *relay* apa saja yang ada di state yang telah ditentukan.
3. Kombinasi *relay* yang ada disatu state ditulis dengan logika AND.
4. Menggabungkan kombinasi *relay* yang memiliki *output*-n yang sama dengan logika OR.

Berikut contoh *switching function output*:

$$Z1 = (Y1 * \overline{Y2}) + (Y1 * Y2) + (\overline{Y1} * Y2)$$
$$Z2 = (Y1 * \overline{Y2})$$
$$Z3 = (Y1 * Y2)$$

2.6.8. Pembuatan Diagram *Ladder*

Pembuatan diagram *ladder* merupakan bentuk pengaplikasian dari *switching function* yang telah dirumuskan pada tahap sebelumnya. Pada sebuah fungsi atau persamaan, disebelah kiri tanda sama dengan (=) merupakan *output*, sedangkan disebelah kanan merupakan *input*. Begitu juga dengan *switching function*, dibagian sebelah kiri adalah *ouput* atau *relay*, sedangkan disebelah kanan merupakan *input* atau kontak.



Gambar 2.20. Contoh diagram *ladder* hasil *switching function*

BAB 3
PERANCANGAN SISTEM

3.1. Garis Besar Kerja Sistem

Dalam tugas akhir ini digunakan sebuah sistem yang bernama *factory automatic trainer* atau FAT. FAT memiliki prinsip dasar dari proses yang ada di industri manufaktur seperti penyeleksian, pendistribusian, pengepakan dan lainnya. Prinsip-prinsip dasar yang ada pada FAT menggunakan empat buat modul, yaitu *separation module, pick and place module, stopper module* dan *line movement module*.

Modul-modul yang ada pada FAT akan memproses tiga jenis benda kerja yaitu satu benda logam dan dua benda bukan logam. Benda bukan logam ini terdiri dari benda bewarna biru dan hitam. Dalam proses ini semua benda akan diproses bergantian secara kontinyu. Kontinyu yang dimaksud adalah ketika benda kerja pertama sudah mulai di proses, benda kerja selanjutnya akan dimasukan ke dalam proses setelah beberapa detik berlalu. Jadi dalam sistem ini benda kerja selanjutnya akan ikut di proses tanpa harus menunggu proses benda kerja yang pertama selesai dahulu.

Benda kerja yang terdiri dari tiga buah tersebut akan diproses secara berbeda pada modul tertentu. Untuk benda logam, setelah benda memasuki area proses separation module, benda akan dipisahkan langsung. Sehingga proses benda logam berhenti di *separation module*. Benda bukan logam akan diproses sama yaitu melewati *pick and place module, stopper module* sampai modul terakhir yaitu *line movement module*. Namun pada modul terakhir dua buah benda bukan logam tersebut diletakkan di kotak yang berbeda.

Gambar 3.1. Benda kerja (a) metal, (b) biru, (c) hitam

3.2. Rincian langkah kerja sistem

Dalam tugas akhir ini, *factory automatic trainer* dibagi kedalam tujuh sub sistem dan tiga sub sistem eksternal. Berikut urutan dari ke tujuh sub sistem:

1. *Insert*
2. *Separator metal*
3. *Pick and place*
4. *Stopper*
5. *Conveyor 1*
6. *Conveyor 2*
7. *Line movement*

3.2.1. *Insert*

Sub sistem ini memiliki fungsi untuk memasukan benda kerja ke konveyor dan mendeteksi jenis benda kerja yang melewati sensor. Berikut rincian langkah kerja sub sistem:

1. Sensor tidak mendeteksi, tidak ada proses.
2. Memasukkan benda kerja ke *magazine*, ketika mengenai sensor *magazine*, *timer* menghitung mundur.
3. Ketika *timer* aktif, benda didorong oleh silinder ke konveyor.
4. Ketika sensor *insert* mendeteksi, silinder *insert* mundur ke posisi semula.
5. Ketika benda melewati sensor *capacitive* dan sensor *magazine* mendeteksi benda, *timer* menghitung mundur kembali.
6. Ketika benda melewati sensor *capavitive* dan sensor *magazine* tidak mendeteksi benda, proses berakhir.

3.2.2. *Separator Metal*

Separator metal berfungsi untuk memisahkan benda kerja metal dari proses. Berikut rincian langkah kerja sub sistem:

1. Sensor *eject return* mendeteksi saat kondisi awal, tidak ada proses.
2. Ketika benda kerja metal melewati sensor *proximity*, *timer* menghitung mundur.
3. Ketika *timer* aktif, silinder pemisah benda kerja metal akan mendorong benda keluar dari *conveyor*.
4. Ketika sensor *eject* mendeteksi, silinder pemisah kembali ke posisi awal.
5. Ketika *sensor eject* mendeteksi kembali, proses berakhir.

3.2.3. *Pick and Place*

Pick and place (PNP) berfungsi untuk memindahkan benda kerja dari konveyor 1 ke konveyor 2. Berikut rincian langkah kerja sub sistem:

1. Pada saat posisi diam, *pick and place* berada di posisi up dan sudah bergerak ke arah *clockwise*.
2. Ketika *end separation* mendeteksi benda, PNP akan ke posisi *down*.
3. Ketika sensor *down* mendeteksi, *vacuum* aktif.
4. Ketika sensor *vacuum* mendeteksi, posisi *up*.
5. Ketika sensor *up* mendeteksi, posisi PNP berubah ke *counter clockwise*.
6. Ketika sensor *counter clockwise* mendeteksi, PNP akan ke posisi down.
7. Ketika sensor *down* mendeteksi, *vacuum* mati dan PNP ke posisi *up.*
8. Ketika sensor *up* mendeteksi, posisi PNP berubah ke posisi *clockwise*.
9. Ketika sensor *clockwise* mendeteksi, proses selesai.

3.2.4. *Stopper*

Pada proses *stopper* dilakukan proses *drilling* atau pengeboran benda kerja. Berikut rincian langkah kerja sub sistem:

1. Sensor *stopper down* dan *drill up* mendeteksi pada kondisi awal.
2. Ketika *work point* mendeteksi benda, *drill* akan aktif dan turun.
3. Ketika sensor *drill down* mendeteksi, *timer* akan menghitung mundur.
4. Ketika *timer* aktif, *drill* akan naik.
5. Ketika sensor *drill up* mendeteksi, *stopper* akan naik.
6. Ketika sensor *stopper up* mendeteksi, *timer* lain akan menghitung mundur.
7. Ketika *timer* aktif, *stopper* akan turun.
8. *Timer* tidak aktif, proses selesai.

3.2.5. *Conveyor* 1

Pada proses ini hanya akan di atur hidup dan mati dari *conveyor* 1. Berikut rincian langkah kerja sub sistem:

1. Ketika tombol *start (initial system)* ditekan, *conveyor* akan langsung menyala.

2. Ketika *end separator* mendeteksi benda, *timer* akan menghitung mundur.
3. Ketika *timer* aktif dan *timer* pada subsistem 2 sedang menghitung mundur, *conveyor* tetap menyala.
4. Ketika *timer* aktif dan *timer* pada subsistem 2 tidak sedang menghitung mudur, *conveyor* akan mati.
5. Ketika *end separator* tidak mendeteksi lagi, *conveyor* akan menyala kembali.

3.2.6. *Conveyor* 2

Sama dengan *coveyor* 1, pada sistem ini hanya akan diatur hidup dan mati dari *coveyor* 2. Berikut rincian langkah kerja sub sistem:

1. Ketika tombol *start (initial system)* ditekan, *conveyor* akan langsung menyala.
2. Ketika *work area* mendeteksi, maka *timer* akan menghitung mundur.
3. Ketika *timer* aktif, *conveyor* akan mati.
4. Ketika *stopper up* mendeteksi, *conveyor* akan menyala kambali.
5. Semua masukan mati, tidak ada proses.

3.2.7. *Line Movement*

Proses yang ada pada *line movement* adalah mengambil dan memindahkan benda kerja ke salah satu kotak. Berikut rincian langkah kerja sub sistem:

1. Pada saat posisi diam, kondisi *gripper* dari *line movement* akan ada di atas, terbuka, dan berada diposisi dekat *coveyor* 2.
2. Ketika *end stopper* mendeteksi, *grip* akan turun.
3. Ketika *grip down* mendeteksi, *grip* akan aktif.
4. Ketika *finger grip* mendeteksi, *grip* akan naik.
5. Ketika *grip up* mendeteksi, *grip* akan ke posisi *forward*.
6. Ketika *forward* mendeteksi, *grip* akan turun.
7. Ketika sensor *down* mendeteksi, *grip* akan *open* dan naik.
8. Ketika sensor *up* mendeteksi, *grip* akan *backward* ke posisi awal.
9. Ketika sensor *backward* mendeteksi, proses selesai.

3.3. Masukan dan Keluaran Sistem

Dalam pengerjaan tugas akhir, digunakan sebanyak 36 buah masukan yang terdiri dari masukan dari FAT itu sendiri dan masukan yang didapat dari kontak beberapa *timer relay*. Keluaran yang digunakan sebanyak 29 buah yang terdiri dari keluaran FAT dan beberapa *relay* yang digunakan mengaktifkan *timer*. Pada tabel 3.1. akan dijelaskan masukan-masukan yang digunakan.

Tabel 3.1. Masukan sistem

No	Nama	Alamat	Keterangan
1	*Magazine*	IX0.0.6	Sensor yang medeteksi keberadaan benda kerja untuk awal proses
2	*Proximity*	IX0.0.8	Sensor untuk mendeteksi benda logam
3	*Capacitive*	IX0.0.9	Sensor untuk mendeteksi benda
4	*Insert*	IX0.0.10	Sensor yang mendeteksi silinder *insert* dalam posisi *insert*
5	*Timer 1*	AUTO	Kontak dari *timer* 1
6	*Eject*	IX0.0.12	Sensor yang mendeteksi silinder *eject* dalam posisi *eject*
7	*Eject return*	IX0.0.13	Sensor yang mendeteksi silinder *eject* dalam posisi *return*
8	*Timer 2*	AUTO	Kontak dari *timer* 2
9	*End separator*	IX0.0.14	Sensor yang mendeteksi benda ketika sudah berada di akhir *conveyor* 1
10	*Rotary CCW*	IX0.1.6	Sensor yang mendeteksi PNP ada di posisi berlawanan arah jarum jam
11	*Rotary CW*	IX0.1.7	Sensor yang mendeteksi PNP ada di posisi searah jarum jam
12	*Up PNP*	IX0.1.8	Sensor yang mendeteksi PNP ada di posisi atas
13	*Down PNP*	IX0.1.9	Sensor yang mendeteksi PNP ada di posisi bawah
14	*Vacuum*	IX0.1.10	Sensor yang mendeteksi vacuum PNP sedang aktif

15	*Stopper up*	IX0.1.0	Sensor yang mendeteksi stopper ada di posisi atas
16	*Stopper down*	IX0.1.1	Sensor yang mendeteksi stopper ada di posisi bawah
17	*Work point*	IX0.1.2	Sensor yang mendeteksi benda kerja berada di area *drilling*
18	*Drill up*	IX0.1.3	Sensor yang mendeteksi *drill* ada di posisi atas
19	*Drill down*	IX0.1.4	Sensor yang mendeteksi *drill* ada di posisi bawah
20	*Timer 3*	AUTO	Kontak dari *timer* 3
21	*Timer 4*	AUTO	Kontak dari *timer* 4
22	*Timer 1 act*	AUTO	Kontak dari pengaktif *timer* 1
23	*Timer 5*	AUTO	Kontak dari *timer* 5
24	*Timer 6*	AUTO	Kontak dari *timer* 6
25	*End stopper*	IX0.1.5	Sensor yang mendeteksi benda ketika sudah berada di akhir *conveyor* 2
26	*Forward*	IX0.0.0	Sensor yang mendeteksi *gripper* ada di posisi *forward* atau di dekat *conveyor* 1
27	*Backward*	IX0.0.1	Sensor yang mendeteksi *gripper* ada di posisi *forward* atau di dekat *conveyor* 2
28	*Up gripper*	IX0.0.2	Sensor yang mendeteksi *gripper* ada di posisi atas
29	*Down gripper*	IX0.0.3	Sensor yang mendeteksi *gripper* ada di posisi bawah
30	*Finger open*	IX0.0.4	Sensor yang mendeteksi *gripper* sedang membuka
31	*Finger grip*	IX0.0.5	Sensor yang mendeteksi *gripper* sedang menutup
32	*Photosensor*	IX0.0.7	Sensor yang mendeteksi benda selain hitam
33	*Timer 7*	Auto	Kontak dari *timer* 7

Pada tabel 3.2. akan dijelaskan keluaran-keluaran yang digunakan.

Tabel 3.2. Keluaran sistem

No	Nama	Alamat	Keterangan
1	*Insert*	QX0.2.6	Pergerakan silinder yang mendorong benda kerja ke konveyor
2	*Insert return*	QX0.2.7	Pergerakan silinder pendorong kembali ke posisi awal
3	*Timer 1 act*	AUTO	Pengaktif waktu mundur *timer* 1
4	*Eject*	QX0.2.8	Pergerakan silinder yang memisahkan benda kerja metal
5	*Eject return*	QX0.2.8	Pergerakan silinder pemisah kembali ke posisi awal
6	*Timer 2 act*	AUTO	Pengakif waktu mundur *timer* 2
7	*Rotary CCW*	QX0.3.6	Pergerakan silinder PNP ke posisi konveyor 2
8	*Rotary CW*	QX0.3.7	Pergerakan silinder PNP ke posisi konveyor 1
9	*Up PNP*	QX0.3.8	Pergerakan silinder PNP ke posisi atas
10	*Down PNP*	QX0.3.9	Pergerakan silinder PNP ke posisi bawah
11	*Vacuum*	QX0.3.10	Mengaktifkan vacuum penghisap benda kerja
12	*Stopper up*	QX0.3.0	Pergerakan silinder stopper ke posisi atas
13	*Drill up*	QX0.3.1	Pergerakan silinder *drill* ke posisi atas
14	*Drill down*	QX0.3.2	Pergerakan silinder *drill* ke posisi bawah
15	*Drill on*	QX0.3.3	Pengaktif *drill*
16	*Timer 3 act*	AUTO	Pengaktif waktu mundur 3
17	*Timer 4 act*	AUTO	Pengaktif waktu mundur 4
18	*Conveyor 1*	QX0.2.10	Pengaktif konveyor 1
19	*Timer 5 act*	AUTO	Pengaktif waktu mundur 5
20	*Conveyor 2*	QX0.3.4	Pengaktif konveyor 2
21	*Timer 6 act*	AUTO	Pengaktif waktu mundur 6

22	*Forward*	QX0.2.0	Pergerakan silinder *gripper* ke posisi konveyor 1
23	*Backward*	QX0.2.1	Pergerakan silinder *gripper* ke posisi konveyor 2
24	*Up Gripper*	QX0.2.2	Pergerakan silinder *gripper* ke posisi atas
25	*Down gripper*	QX0.2.3	Pergerakan silinder *gripper* ke posisi bawah
26	*Finger grip*	QX0.2.4	Pergerakan silinder *gripper* menjepit benda kerja
27	*Sign red*	QX0.3.11	Lampu indikator merah
28	*Sign yellow*	QX0.3.12	Lampu indikator kuning
29	*Sign green*	QX0.3.13	Lampu indikator hijau

3.4. Pembuatan State Diagram dan Ladder

Berdasarkan landasan teori pada bab ii, terdapat enam buah langkah kerja *state diagram* dan ditambah dua langkah agar bisa menjadi sebuah diagram *ladder*.

3.4.1. Penentuan Urutan Proses dari Sistem

Langkah pertama dalam penerapan metode *state diagram* untuk merumuskan sebuah diagram *ladder* adalah menentukan urutan proses dari sistem. Untuk mempermudah pengerjaan, sistem dibagi ke sub-sub tertentu yang langkah kerjanya sudah disesuaikan dengan keinginan. Dalam pengerjaan *factory automatic trainer* dibagi menjadi tujuh buah sub sistem dan tiga buah sub sistem eksternal yang tidak menggunakan metode *state diagram* dalam pembuatannya. Pada tabel 3.3. akan ditentukan pembagian sub sistem dan urutan kerja dalam masing-masing sub sistem tersebut.

Tabel 3.3. Pembagian kerja sub sistem

Sub Proses	State	Masukan	Keluaran
Insert	1	-	-
	2	*Magazine on*	*Timer 1 act on*
	3	*Timer 1 on*	*Insert on*
	4	*Magazine off, Insert on*	*Insert return on, Insert off, Timer 1 act off*

	5	*Magazine on, Capacitive on, Insert off, Timer 1 off*	*Timer 1 act on, Insert return off*
	6	*Capacitive on, Insert off, Timer 1 off*	*Insert return off*
Separator metal	1	*Eject return on*	-
	2	*Proximity on*	*Timer 2 act on*
	3	*Timer 2 on, Proximity off*	*Eject on*
	4	*Eject on, Eject return off*	*Eject return on, Eject off, Timer 2 act off*
	1	*Eject return on, Eject off, timer 2 off*	-
Pick and Place	1	*Rotari CW on, Up on*	-
	2	*End separator on*	*Down PNP on*
	3	*Down PNP on, Up PNP off*	*Vacuum on, Down PNP off*
	4	*Vacuum on*	*Up PNP on*
	5	*Up PNP on, Down PNP off*	*Rotary CCW on, Up PNP off*
	6	*Rotary CCW on, Rotary CW off*	*Down PNP on, Rotary CCW off*
	7	*Down PNP on, Up PNP off*	*Up PNP on,Vacuum off, Down PNP off*
	8	*Up PNP on, Down PNP off, Vacuum off*	*Rotary CW on, Up PNP off*
	1	*Rotary CW on, Rotary CCW off*	-
Stopper	1	*Stopper down on, Drill up on*	-
	2	*Work area on*	*Drill on on, Drill down on*
	3	*Driil down on, Drill up off*	*Timer 4 act on, Drill down off*
	4	*Timer 4 on*	*Drill up on*
	5	*Drill up on, drill down off*	*Stopper up on, Drill up off, Drill on off, Timer 4 act off*

	6	*Stopper up on, Stopper down off, Work area off, Timer 4 off*	*Timer 3 act on*
	7	*Timer 3 on*	*Stopper up off, timer 3 act off*
	8	*Timer 3 off,*	
Conveyor 1	1	-	*Conveyor 1 on*
	2	*End separator on*	*Timer 4 act*
	3	*Timer 4 on*	*Conveyor 1 off*
	4	*End separator on, Timer 1 act on*	*Timer 4 act*
	5	*Timer 4 on, Timer 1 act on*	-
	6	*End separator off*	*Conveyor 2 on, Timer 4 act off*
Conveyor 2	1	-	*Conveyor on*
	2	*Work area on*	*Timer 6 act on*
	3	*Timer 6 on*	*Conveyor 2 off*
	4	*Stopper up on*	*Conveyor on, Timer 4 act off*
	1	*Stopper up off, Work area off, Timer 6 off*	-
Line movement	1	*Backward on, Up grip on, Grip open on*	-
	2	*End stopper on*	*Down grip on*
	3	*Down grip on, Up grip off*	*Finger grip on, Down grip off*
	4	*Finger grip on, Finger open off*	*Up grip on*
	5	*Up grip on, Down grip off*	*Forward on*
	6	*Forward on, Backward off*	*Down grip on, Forward off*
	7	*Down grip on, up Grip off*	*Up grip on, Down grip off, Finger grip off*

	8	*Up grip on, Finger open on, Down grip off, Finger grip off*	*Backward on, Up off*
	1	*Backward on, Forward off*	-

3.4.2. Pendiskripsian Masukan dan Keluaran dari Sistem

Berikut adalah tabel yang mendeskripsikan masukan dan keluaran dari sistem.

Tabel 3.4. Deskripsi masukan dan keluaran sistem

No	Masukan		Keluaran	
	Nama	Simbol	Nama	Simbol
1	*Magazine*	X1	*Insert*	Z1
2	*Proximity*	X2	*Insert return*	Z2
3	*Capacitive*	X3	*Timer 1 act*	Z3
4	*Insert*	X4	*Eject*	Z4
5	*Timer 1*	X5	*Eject return*	Z5
6	*Eject*	X6	*Timer 2 act*	Z6
7	*Eject return*	X7	*Rotary CCW*	Z7
8	*Timer 2*	X8	*Rotary CW*	Z8
9	*End separator*	X9	*Up PNP*	Z9
10	*Rotary CCW*	X10	*Down PNP*	Z10
11	*Rotary CW*	X11	*Vacuum*	Z11
12	*Up PNP*	X12	*Stopper up*	Z12
13	*Down PNP*	X13	*Drill up*	Z13
14	*Vacuum*	X14	*Drill down*	Z14
15	*Stopper up*	X15	*Drill on*	Z15
16	*Stopper down*	X16	*Timer 3 act*	Z16
17	*Work point*	X17	*Timer 4 act*	Z17
18	*Drilll up*	X18	*Conveyor 1*	Z18
19	*Drill down*	X19	*Timer 5 act*	Z19
20	*Timer 3*	X20	*Conveyor 2*	Z20
21	*Timer 4*	X21	*Timer 6 act*	Z21
22	*Timer 1 act contact*	X22	*Forward*	Z22
23	*Timer 5*	X23	*Backward*	Z23
24	*Timer 6*	X24	*Up grip*	Z24
25	*End stopper*	X25	*Down grip*	Z25

26	*Forward*	X26	*Finger grip*	Z26
27	*Backward*	X27	*Red sign*	Z27
28	*Up grip*	X28	*Yellow sign*	Z28
29	*Down grip*	X29	*Green sign*	Z29
30	*Finger open*	X30		
31	*Finger grip*	X31		
32	*Photosensor*	X32		
33	*Timer 7*	X33		

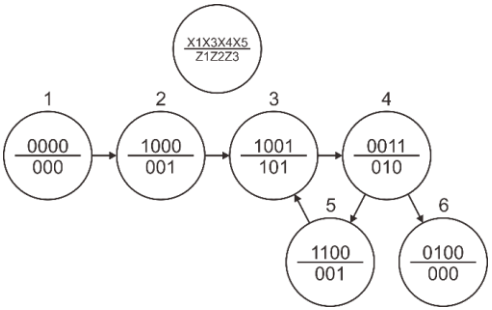
3.4.3. Pembuatan *State Diagram* I/O

Pembuatan *state diagram* I/O untuk sistem *factory automatic trainer* dibagi menjadi tujuh sub proses berdasarkan langkah penentuan urutan proses atau langkah pertama. Selanjutnya akan dibahas proses pembuatan dari diagram I/O untuk sub sistem pertama yaitu *insert*. Pada sub ini terdapat empat masukan dan tiga keluaran. Berikut tabel sub sistem 1.

Tabel 3.5. Sub sistem 1

Sub Proses	State	Masukan	Keluaran
Insert	1	-	-
	2	*Magazine on*	*Timer 1 act on*
	3	*Timer 1 on*	*Insert on*
	4	*Magazine off, Insert on*	*Insert return on, Insert off, Timer 1 act off*
	5	*Magazine on, Capacitive on, Insert off, Timer 1 off*	*Timer 1 act on, Insert return off*
	6	*Capacitive on, Insert off, Timer 1 off*	*Insert return off*

Pada tabel 3.5. untuk kolom masukan dan keluaran merupakan kondisi masukan dan keluaran yang berubah. Perubahan tersebut bisa dari mati ke hidup maupun dari hidup ke mati. Perubahan dari mati ke hidup ditandai dengan angka satu untuk masing-masing masukan dan keluarannya, begitu juga sebaliknya untuk perubahan dari hidup ke mati ditandai dengan angka nol. Berikut *state diagram* I/O dari sub sistem 1.



Gambar 3.2. *State diagram* I/O sub sistem 1

Pada tabel 3.5. tampak pada kolom masukan dan keluaran state 1 tidak terdapat kerangan masukan dan keluaran. Hal itu menandakan tidak ada perubahan pada state, yang berarti semua masukan dan keluaran bernilai nol jika dimasukan ke dalam *state digram* gambar 3.2.. Selanjutnya pada state kedua pada tabel tampak masukan dan keluaran memiliki keterangan *magazine on* dan *timer 1 act on*, maka pada *state diagram* state kedua ditulis masukan dan keluar bernilai satu untuk X1 dan Z3. Begitu seterusnya hingga semua state selesai dibuat.

Arah panah dari *state diagram* dapat ditentukan sesuai keinginan. Maksud dari sesuai keinginan yaitu ditentukan dengan bagaimana pergerakan normal dari sistem sesuai dengan yang dibutuhkan. Seperti pada state 1 secara normal bergerak ke state 2, atau seperti state 4 bisa bergerak normal ke state 5 dan 6.

3.4.4. Penyusunan *Primitive Flow Table*

State diagram yang ada pada gambar 3.2. dapat langsung di terjemahkan ke dalam bentuk tabel. Berikut adalah tabel dari *primitive flow table*.

Tabel 3.6. Primitive flow table sub sistem 1

Row	Input (X1X3X4X5)						Output		
	0000	1000	1001	0011	1100	0100	Z1	Z2	Z3
1	**1**	2	-	-	-	-	0	0	0
2	-	**2**	3	-	-	-	0	0	1
3	-	-	**3**	4	-	-	1	0	1
4	-	-	-	**4**	5	6	0	1	0
5	-	-	3	-	**5**	-	0	0	1
6	-	-	-	-	-	**6**	0	0	0

Pada gambar 3.2. terdapat enam buah state yang kebetulan memiliki kombinasi masukan yang berbeda-beda, maka dapat langsung dimasukan ke dalam *primitive flow table*. Untuk state-state stabil yang ada dapat langsung dimasukan dengan tulisan yang dicetak tebal ke dalam kolom masukan (*input*) dari masing-masing state yang sejajar dengan baris *row* state. Contoh untuk *row* 1, state 1 stabil memiliki kombinasi masukan 0000 pada *state diagram*, maka dapat langung dituliskan state 1 di baris *row* 1 dan di kolom input 0000.

Untuk state tidak stabil cukup melihat arah panah yang dituju dari state stabil yang sedang diproses. Misal di baris *row* 1 memiliki state 1 sebagai state stabil, jika dilihat dari *state digram*-nya tampak state 1 akan menuju state 2. Dari kondisi tersebut maka dituliskan state tidak stabil 2 di baris *row* 1 dan dikolom masukan dari state 2 tersebut, yaitu 1000. Kondisi *don't care* merupakan cara yang paling mudah yaitu dengan menulis garis mendatar atau biasa disebut dengan strip pada sel-sel yang tidak diisi dengan state stabil dan state tidak stabil.

Langkah terakhir dari pembuatan *primitive flow table* adalah menentukan keluaran (*output*). Menentukan keluaran juga sangat mudah cukup melihat keluaran masing-masing state dan mencocokan dengan masukan dibaris *row* yang sama. Misal pada *state diagram* state 1 memiliki keluran 000, maka dituliskan Z1=0, Z2=0 dan Z3=0.

3.4.5. Penyusunan *Merged Tlow Table*

Pada tabel 3.6. terdapat *row* yang memiliki keluaran yang sama yaitu *row* 1 dengan 6, dan *row* 2 dengan 5. Maka dalam pembutan *merged flow table* dapat dilakukan penyederhanaan baris. Tabel 3.7. berikut merupakan *merged flow table*.

Tabel 3.7. Merged flow table

Row	Input						Output			Relay	
	0000	1000	1001	0011	1100	0100	Z1	Z2	Z3	Y1	Y2
1,6	**1**	2	-	-	-	**6**	0	0	0	0	0
2,5	-	**2**	3	-	**5**	-	0	0	1	1	0
3	-	-	**3**	4	-	-	1	0	1	1	1
4	-	-	-	**4**	5	6	0	1	0	0	1

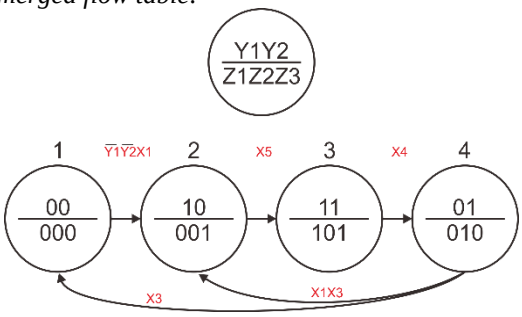
Tampak pada tabel 3.7., baris yang memiliki keluaran yang sama sudah digabungkan mengikuti aturan dari penggabungan pada dasar teori. Dalam penggabungan, jumlah baris berubah menjadi empat baris,

sehingga dibutuhkan dua buah *relay* untuk memenuhi kebutuhan dari jumlah kombinasi *relay*.

Selama pengerjaan tugas akhir didapat aturan tambahan dalam penentuan kombinasi *relay* untuk masing-masing *row*. Aturan tambahan tersebut adalah aturan untuk penempatan kombinasi relay 0, 00, 000 dan seterusnya atau kondisi dimana semua *relay* dalam keadaan mati. Ketika kombinasi relay dalam keadaan mati semua, maka harus diposisikan di keluaran state yang bernilai 0, 00, 000 atau dalam keadaan mati semua juga. Tujuannya untuk membuat sistem tersebut tidak langsung menyala ketika belum diberi perubahan sinyal apapun.

3.4.6. Pembuatan *State Diagram* R/O

Setelah dilakukan *merging* pada tabel 3.7., dihasilkan empat buah baris yang juga berarti akan ada empat buah state pada *state diagram* I/O. Berikut adalah gambar dari *state digram* I/O yang didapat dari hasil pembuatan *merged flow table*.



Gambar 3.3. *State diagram* R/O sub sistem 1

Untuk pengisian state, cukup dengan mengisi kombinasi *relay* dibagian atas dan dibagian bawah diisi kombinasi keluaran (*output*) yang sebaris dengan *relay* tersebut. Seperti baris 1,6 meliki *relay* 00 dan *output* 000, maka digabungkan menjadi satu di-state yang sama.

Dalam penentuan arah panah dapat dilihat dari pergerakan state stabil ke state tidak stabil dimasing-masing baris. Pada tabel 3.7. tampak baris 1,6 (state 1 R/O) meliki state stabil yang 1 dan 6 mengarah kearah state 2 dibaris 2,5 (state 2 R/O). Maka dibuatlah arah panah dari state 1 R/O ke state 2 R/O. Begitu seterusnya hingga semua arah panah selesai dibuat.

Langkah selanjutnya adalah penempatan masukan (*input*) yang memiliki pengaruh terhadap perubahan dari satu state ke state selanjutnya. Pada gambar 3.3., state 1 berubah ke state 2 dengan mengalami perubahan *relay* dari 00 menjadi 10. Jika dilihat kembali ke tabel 3.7., perubahan *relay* 00 ke 10 diakibatkan oleh perubahan state stabil 1 menuju state tidak stabil 2 dan kemudian sampai ke state stabil 2. Maka dilihatlah masukan yang berubah aktif dari state 1 dan 2 tersebut. Pada tabel 3.7. tampak masukan yang berubah adalah X1, maka ditulislah X1 dipanah penghubung state 1 dan state 2. Begitu seterusnya hingga proses selesai.

Ada kondisi tambahan dalam pembuatan keterangan yang ada dipanah penghubung. Kondisi tersebut yaitu ketika suatu *input* hanya boleh memberi sinyal untuk mengubah state ke state lain dalam suatu kondisi saja. Contoh pada pada sub sistem 1, X1 dapat hidup beberapa kali selama prosesnya, namun X1 hanya boleh aktif dalam keadaan tertentu yaitu ketika sistem dalam keadaan diam atau *relay* dalam keadaan mati semua. Maka dalam pembuatan state diagaram R/O nya ditambahkan kondisi relay Y1 dan Y2 keduanya mati ke *input* X1 seperti gambar 3.3.

3.4.7. Pembuatan *Switching Function*

Jika berpatokan dengan gambar 3.3. atau *state diagram* R/O untuk sub sistem 1, maka *switching function* yang harus dicari adalah untuk *relay* Y1 Y2 dan *output* Z1 Z2 Z3. Perbedaan dari pembuatan *switching function relay* dan *output*, yaitu ketika untuk *switching function relay* akan dicari sinyal *set* dan *reset*, tapi untuk *output* hanya dicari sinyal *set*.

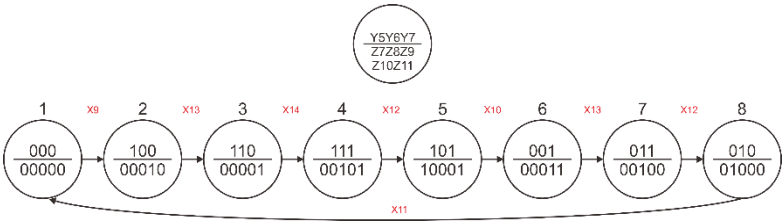
Dimulai dengan *relay* Y1, Y1 akan aktif dari state 2 hingga state 3 dan kemudian mati di state 4. Dari kondisi tersebut jika dilihat di gambar 3.3., Y1 akan aktif saat perpindahan state 1 ke state 2 dan state 4 ke state 2. Maka ditetapkan lah *input* yang ada pada arah panah 1 ke 2 dan arah panah 4 ke 2 sebagai *set* dari Y1. Selanjutnya adalah menentukan *reset* dari Y1. Sesuai dengan kondisi Y1 mati ketika perpindahan dari state 3 ke state 4. Maka *input* yang ada di panah 3 ke 4 merupakan *reset* dari Y1. Langkah terakhir dari *switching function* Y1 adalah menambahkan *safe holding* oleh Y1 itu sendiri dibagian *set*.

Selanjutnya untuk pembuatan *output*, Z1 akan aktif di state 3 saja yang memiliki kondisi *relay* Y1Y2 aktif keduanya. Maka *relay* Y1 dan Y2 tersebut merupakan *set* dari Z1. Begitu tahap selanjutnya hingga

semua *swtiching function* selesai. Berikut *switching function* untuk sub sistem 1.

$$Y1 = [(\overline{Y1} * \overline{Y2} * X1) + (X1 * X3) + Y1] * \overline{X4}$$
$$Y2 = (X5 + Y2) * \overline{X3}$$
$$Z1 = Y1 * Y2$$
$$Z2 = \overline{Y1} * Y2$$
$$Z3 = (Y1 * \overline{Y2}) + (Y1 * Y2)$$

Berbeda dengan sub sistem 1, sub sistem 3 memiliki kondisi tambahan yaitu ada sebuah *relay* yang mengalami kondisi *set* dan *reset*-nya selama dua kali dalam satu lingkaran proses. Gambar 3.4. merupakan gambar dari *state diagram* R/O dari sub sistem 3.



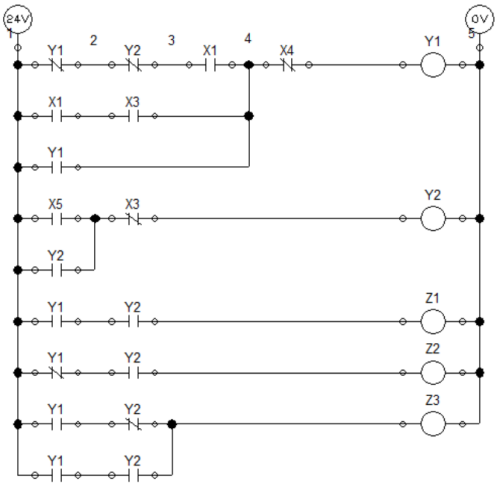
Gambar 3.4. *State diagram* R/O sub sistem 3

Pada *state diagram* R/O sub sistem 3, dapat dilihat untuk Y6 mengalami dua kali kondisi *set* dan *reset*. Berdasarkan aturan yang ada pada dasar teori, untuk sinyal yg mengubah kondisi menjadi *set* cukup digabungkan dengan menggunakan logika AND. Namun tidak dengan penggabungan sinyal yang mengubah menjadi *reset*, harus disesuaikan dengan syarat yang ada pada dasar teori. Berdasarkan sub bab 2.6.7 tentang syarat penggabungan sinyal *reset*, maka syarat yang memenuhi adalah poin pertama. Berikut adalah *switching function* dari *relay* Y6.

$$Y6 = (X13 + Y2) * (X12 + X11)$$

3.4.8. Pembuatan *Ladder Diagram*

Langkah terakhir adalah pembuatan *diagram ladder*. *Switching function* sub sistem 1 yang sudah dibuat, dapat langsung diterjemahkan menjadi *diagram ladder*. Gambar 3.5. adalah *diagram ladder* dari sub sistem 1.



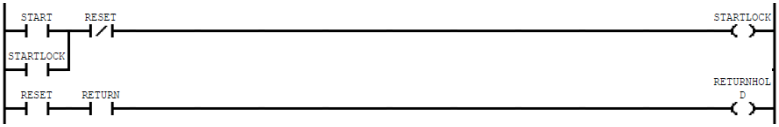
Gambar 3.5. *Diagram ladder* sub sistem 1

3.5. Sub Sistem Eksternal

Selain sistem yang dibuat dengan metode *state diagram*, ada tiga buah sistem tambahan yang tidak menggunakan metode *state diagram*. Sistem yang terdiri dari *initial system, lamp indicator system* dan hitam biru *system*.

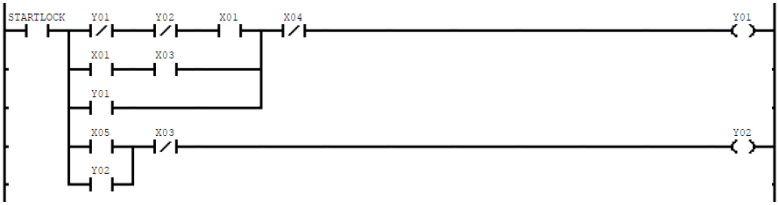
3.5.1. *Initial System*

Initial system berfungsi sebagai syarat awal dari sistem berjalan dan sebagai sistem yang dapat menghentikan sistem keseluruhan ketika saat darurat. Selain itu, juga ditambahkan sebuah sistem tambahan untuk mengembalikan pergerakan sistem ke posisi awal. Ada tiga buah tombol yaitu *start, reset* dan *return*. Tombol *start* berfungsi sebagai syarat awal semua sistem berjalan, *reset* sebagai tombol emergensi sistem dan *return* sebagai tombol untuk mengembalikan sistem ke posisi awal yang di tekan serentak dengan tombol *reset*.



Gambar 3.6. Diagram *ladder initial system*

Gambar 3.6. merupakan ladder dari *initial system* yang jika *start* ditekan akan mengaktifkan *relay startlock* dan mati ketika *reset* ditekan. *Startlock* tersebut digunakan untuk setiap awalan dari semua sub sistem yang ada. Contoh untuk digram *ladder relay* sub sistem 1 akan di tunjukkan di gambar 3.7.



Gambar 3.7. Diagram *ladder relay* sub sistem 1

3.5.2. *Lamp Indicator System*

Lamp indicator system hanya berfungsi sebagai indikator dari sistem itu sendiri. Terdiri dari tiga warna lampu yaitu merah, kuning dan hijau. Merah memberikan keterangan bahwa sistem sedang berjalan atau beroperasi. Kuning memberikan keterangan sistem sedang aktif namun tidak sedang beroperasi. Hijau memberikan keterangan sistem tidak aktif dan tidak beroperasi, maka aman untuk didekati.

Diagram *ladder* akan ditampilkan dilampiran tugas akhir.

3.5.3. Hitam Biru *System*

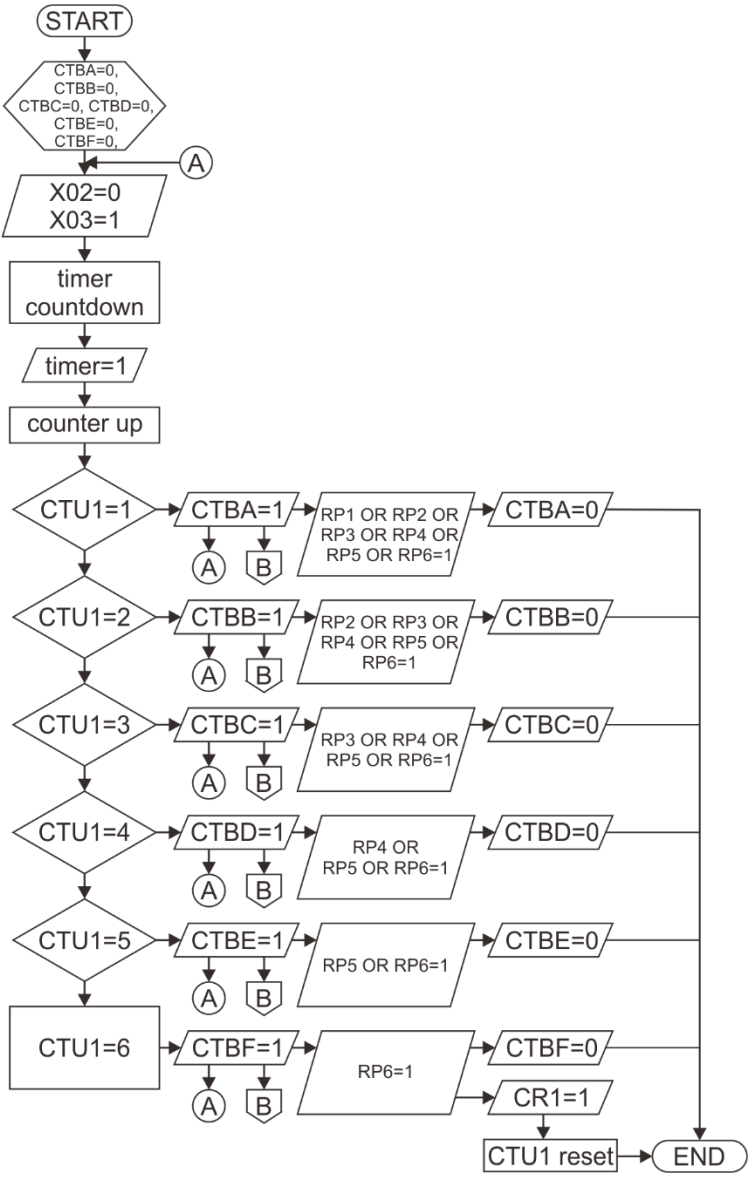
Hitam biru *system* adalah sebuah sistem yang menentukan akhir dari proses benda kerja hitam dan biru. Berdasarkan keterangan dari sub bab 3.1, benda kerja bukan logam diletakkan diletakkan di kotak yang berbeda. Kotak untuk benda kerja yang berwarna hitam diletakkan sejajar dengan *conveyor* 1 dan benda kerja biru diletakkan sebelum *conveyor* 1.

Ketika benda kerja hitam diberi sinyal oleh sensor *forward* dari *line movement* untuk berhenti dan turun untuk melepaskan benda, maka benda kerja biru menggunakan sebuah *timer* untuk turun dan melepaskan benda. Hitam biru *system* inilah yang menentukan kapan *timer* boleh aktif dan tidak aktif ketika benda sedang diproses di sub sistem 7, yaitu *line movement.*

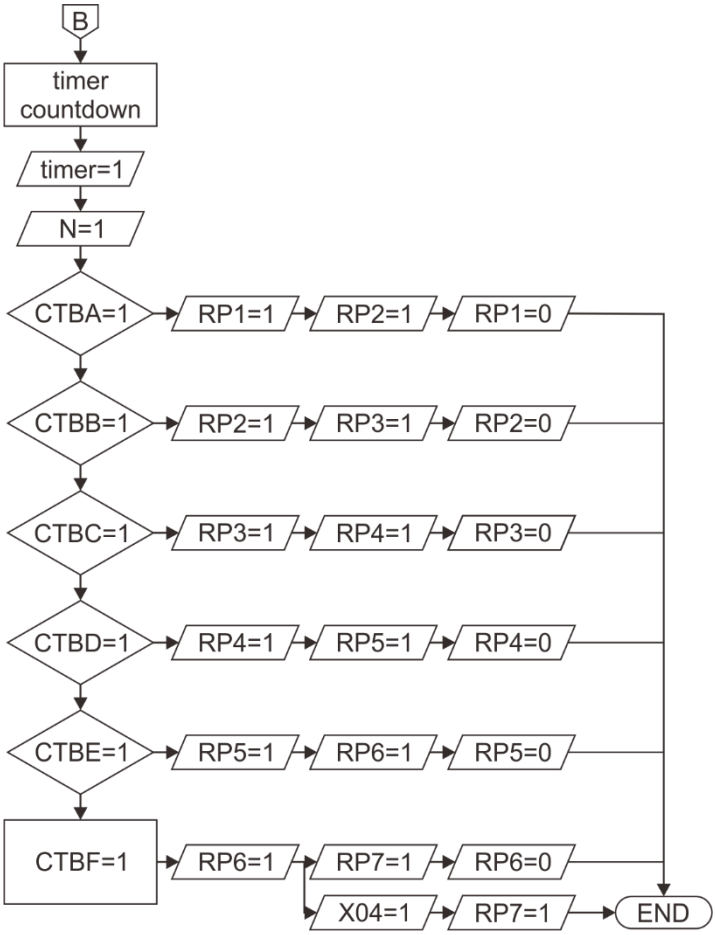
Ada lima tahapan dalam hitam biru *system* ini:

1. *Step counter*
   Step counter adalah sebuah proses yang menghitung proses dari pendeteksian benda kerja hitam dan biru. Tujuannya adalah untuk mengaktifkan *relay* pembanding untuk proses selanjutnya.
2. *Blue step relay*
   Blue step relay adalah *relay* yang menentukan langkah ke berapa saja benda kerja biru dideteksi. Kontak *relay* pembanding dari *step counter* akan mengaktifkan *relay* urutan dari warna biru jika benda yang dideteksi adalah warna biru. Misal pada urutan pembacaan kedua atau ketika *step counter* membaca nilai 2, jika benda yang dideteksi adalah benda kerja biru, maka akan mengaktifkan *blue step relay* kedua.
3. *End step counter*
   End step counter merupakan *relay* pembanding kedua ketika benda kerja mengenai *end stopper. Relay* pembanding tersebut akan digunakan untuk langkah selanjutnya.
4. *Blue timer*
   Blue timer akan aktif ketika *end step counter* memiliki nilai yang sama dengan *blue step relay*. Misal pada *blue step relay* kedua aktif dan *end step counter* membaca nilai 2, maka *blue timer* akan aktif. Kontak *blue timer* yang aktif akan digunakan untuk proses *line movement*, yaitu sebagai sinyal untuk kapan akan melepaskan benda kerja biru ke kotak untuk benda kerja biru.
5. *Blue step-timer reset*
   Berfungsi untuk mereset kondisi dari *blue step relay* dan *blue timer*, karena pada saat sedang berlangsung sistem akan dikunci dan akan di-*reset* ketika proses tersebut telah selesai. Dengan di-*reset*-nya *blue step relay* dan *blue timer*, proses akan dapat diulangi secara terus menerus.

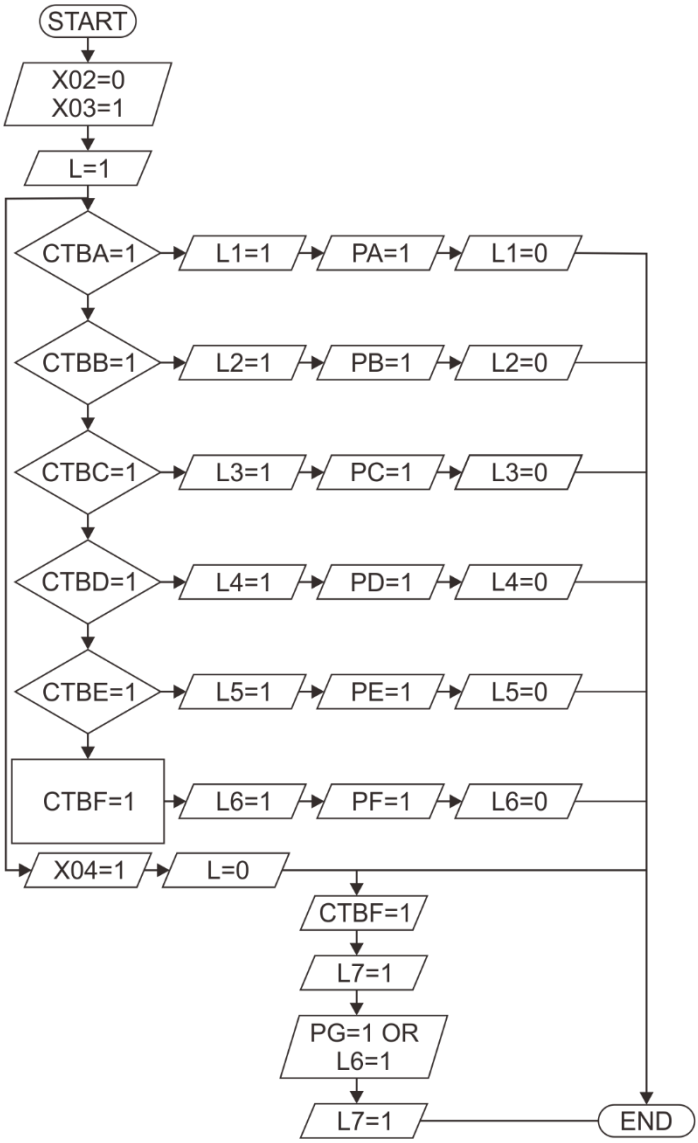
Gambar 3.8. sampai gambar 3.13. adalah *flow chart* dari proses hitam biru *system.*



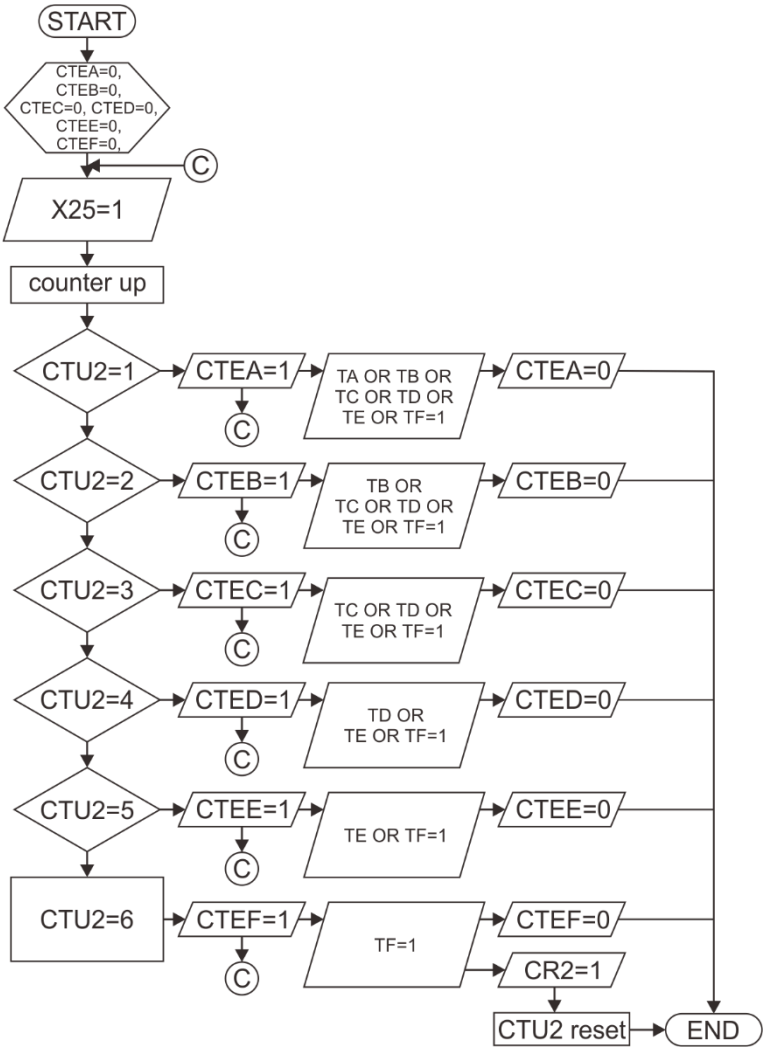
Gambar 3.8. *Flow chart step counter*



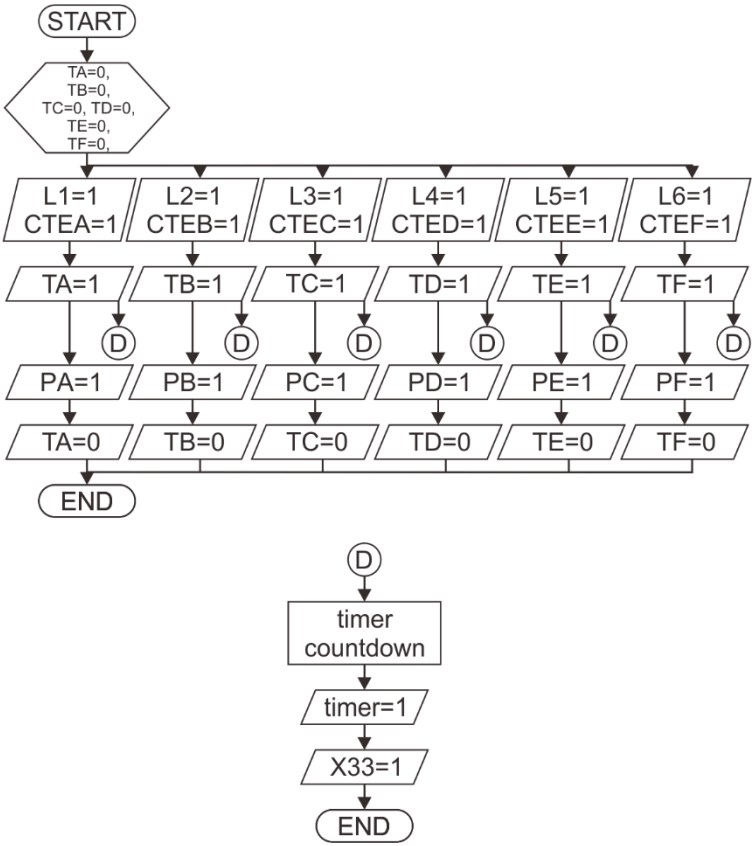
Gambar 3.9. *Flow chart step counter*



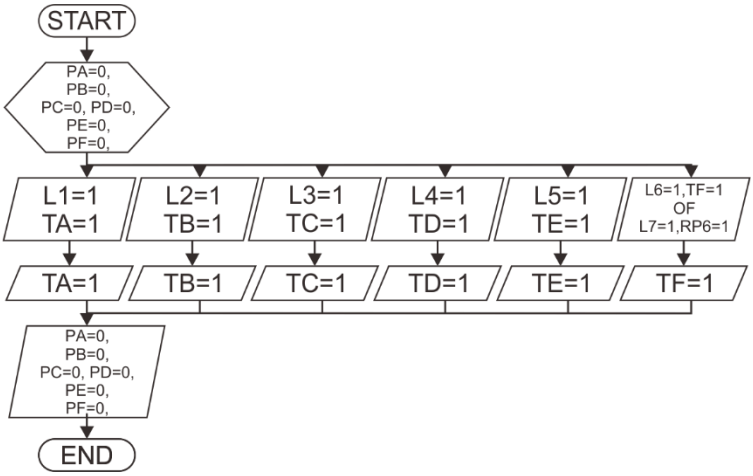
Gambar 3.10. *Flow chart blue step relay*



Gambar 3.11. *Flow chart end step counter*



Gambar 3.12. *Flow chart blue timer*



Gambar 3.13. *Flow chart blue step-relay timer*

BAB 4
PENGUJIAN DAN ANALISA

4.1. Pengujian Sistem

Pengujian sistem dilakukan mulai dari tahap sebelum pembuatan program hingga pengujian sistem setelah ditanamkan program yang sudah melewati proses metode *state diagram*.

4.1.1. Pengujian Masukan dan Keluaran

Tahapan dalam pengujian ini adalah melakukan pemasangan kabel-kabel *jumper*, menyalakan semua MCB, menghubungkan sistem ke pipa udara bertekanan dan mengaktifkan *power* yang ada dipanel modul. yang ada dipanel kontrol FAT. Ada dua macam kabel *jumper* yang harus dihubungkan, sebagai berikut:

1. Kabel *jumper* dari sumber arus DC ke com PLC yang ada dipanel kontrol.

Gambar 4.1. *Output* DC panel kontrol

2. Kabel *jumper* dari arus DC ke panel modul.

Gambar 4.2. Panel modul

Dalam tahap pengujian masukan dan keluaran dari sistem dilakukan secara manual. Pada panel masing-masing modul sudah terdapat tombol yang berguna untuk meggerakkan atau menghidupkan semua perangkat keluaran yang ada pada panel. Pengujian ini dapat dilakukan secara serentak untuk masukan dan keluaran.

Ketika tombol untuk menggerakkan silinder dan menyalakan konveyor ditekan, akan dilihat apakah perangkat keluaran sistem berjalan dengan seharusnya. Selain itu juga dilihat dari pembacaan dari sensor yang ada pada silinder, apakan mendeteksi atau tidak. Untuk sensor-sensor yang akan mendeteksi benda kerja dilakukan secara manual dengan meletakkan benda kerja di posisi yang nantinya akan membaca keberadaan benda kerja tersebut.

Tabel 4.1. Pengujian masukan dan keluaran

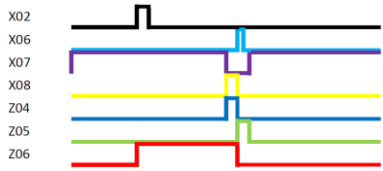
No	Masukan		Keluaran	
	Nama	Normal	Nama	Normal
1	*Magazine*	√	*Insert*	√
2	*Proximity*	√	*Insert return*	√
3	*Capacitive*	√	*Eject*	√
4	*Insert*	√	*Eject return*	√
5	*Eject*	√	*Rotary CCW*	√
6	*Eject return*	√	*Rotary CW*	√
7	*End separator*	√	*Up PNP*	√
8	*Rotary CCW*	√	*Down PNP*	√
9	*Rotary CW*	√	*Vacuum*	√
10	*Up PNP*	√	*Stopper up*	√
11	*Down PNP*	√	*Drill up*	√
12	*Vacuum*	√	*Drill down*	√
13	*Stopper up*	√	*Drill on*	√
14	*Stopper down*	√	*Conveyor 1*	√
15	*Work point*	√	*Conveyor 2*	√
16	*Drilll up*	√	*Forward*	√
17	*Drill down*	√	*Backward*	√
18	*End stopper*	√	*Up grip*	√
19	*Forward*	√	*Down grip*	√
20	*Backward*	√	*Finger grip*	√
21	*Up grip*	√	*Red sign*	√
22	*Down grip*	√	*Yellow sign*	√

23	*Finger open*	√	*Green sign*	√
24	*Finger grip*	√		
25	*Photosensor*	√		

Sesuai dengan tabel 4.1., menunjukkan hasil dari pengujian perangkat masukan dan keluaran yang dibutuhkan berjalan dengan seharusnya. Maka pengujian dapat dilanjutkan ke proses selanjutnya.

4.1.2. Validasi Hasil

Validasi hasil dilakukan guna menyamakan antara perencanaan dan hasil yang dicapai. Dalam melakukan validasi digunakan sebuah software simulasi bernama FluidSim dengan membuat rangkaian penumatik, elektrikal dan diagram ladder nya sesuai dengan yang sudah direncanakan. Dari software tersebut kemudian diambil timechart dari proses yang kemudian dibandingkan dengan timer chart yang sudah dirancang sebelumnya.



Gambar 4.3. Timer chart perencanaan

Gambar 4.3. merupakan hasil perencanaan awal dari sub sistem dua yaitu metal separator. Kemudian hasil simulasi yang ada pada gambar 4.4. dibandingkan dengan perencanaan. Berdasarkan perencanaan dan simuliasi dapat disimpulkan bahwa sistem yang dirancang sudah sesuai dengan hasil yang diinginkan.

Gambar 4.4. Hasil simulasi

4.1.3. Pengujian Digram L*adder*

Pengujian ini dilakukan setelah program selesai dirancang menggunakan metode *state diagram*. Pengujian dilakukan dengan membandingkan hasil dari program diagram *ladder* dengan rencana awal seperti apa sistem berjalan dengan normal. Jika hasil dan rencana sesuai maka dapat dikatakan pembuatan diagram *ladder* dengan metode *state diagram* berjalan dengan lancar dan berhasil.

Tabel 4.2. Hasil pengujian sistem

Sub Proses	State	Masukan	Keluaran	Normal
Insert	1	-	-	√
	2	*Magazine on*	*Timer 1 act on*	√
	3	*Timer 1 on*	*Insert on*	√
	4	*Magazine off, Insert on*	*Insert return on, Insert off, Timer 1 act off*	√
	5	*Magazine on, Capacitive on, Insert off, Timer 1 off*	*Timer 1 act on, Insert return off*	√
	6	*Capacitive on, Insert off, Timer 1 off*	*Insert return off*	√
Separator metal	1	*Eject return on*	-	√
	2	*Proximity on*	*Timer 2 act on*	√
	3	*Timer 2 on, Proximity off*	*Eject on*	√
	4	*Eject on, Eject return off*	*Eject return on, Eject off, Timer 2 act off*	√
	1	*Eject return on, Eject off, timer 2 off*	-	√
Pick and Place	1	*Rotari CW on, Up on*	-	√

	2	*End separator on*	*Down PNP on*	√
	3	*Down PNP on, Up PNP off*	*Vacuum on, Down PNP off*	√
	4	*Vacuum on*	*Up PNP on*	√
	5	*Up PNP on, Down PNP off*	*Rotary CCW on, Up PNP off*	√
	6	*Rotary CCW on, Rotary CW off*	*Down PNP on, Rotary CCW off*	√
	7	*Down PNP on, Up PNP off*	*Up PNP on,Vacuum off, Down PNP off*	√
	8	*Up PNP on, Down PNP off, Vacuum off*	*Rotary CW on, Up PNP off*	√
	1	*Rotary CCW on*	-	√
Stopper	1	*Stopper down on, Drill up on*	-	√
	2	*Work area on*	*Drill on on, Drill down on*	√
	3	*Driil down on, Drill up off*	*Timer 4 act on, Drill down off*	√
	4	*Timer 4 on*	*Drill up on*	√
	5	*Drill up on, drill down off*	*Stopper up on, Drill up off, Drill on off, Timer 4 act off*	√
	6	*Stopper up on, Stopper down off, Work area off, Timer 4 off*	*Timer 3 act on*	√
	7	*Timer 3 on*	-	√
	1	-	*Conveyor 1 on*	√

Conveyor 1	2	*End separator on*	*Timer 4 act*	√
	3	*Timer 4 on*	*Conveyor 1 off*	√
	4	*End separator on, Timer 1 act on*	*Timer 4 act*	√
	5	*Timer 4 on, Timer 1 act on*	-	√
	6	*End separator off*	*Conveyor 2 on, Timer 4 act off*	√
Conveyor 2	1	-	*Conveyor on*	√
	2	*Work area on*	*Timer 6 act on*	√
	3	*Timer 6 on*	*Conveyor 2 off*	√
	4	*Stopper up on*	*Conveyor on, Timer 4 act off*	√
	1	*Stopper up off, Work area off, Timer 6 off*	-	√
Line movement	1	*Backward on, Up grip on, Grip open on*	-	√
	2	*End stopper on*	*Down grip on*	√
	3	*Down grip on, Up grip off*	*Finger grip on, Down grip off*	√
	4	*Finger grip on, Finger open off*	*Up grip on*	√
	5	*Up grip on, Down grip off*	*Forward on*	√
	6	*Forward on, Backward off*	*Down grip on, Forward off*	√
	7	*Down grip on, up Grip off*	*Up grip on, Down grip off, Finger grip off*	√
	8	*Up grip on, Finger open on, Down grip*	*Backward on, Up off*	√

		off, Finger grip off		
	1	*Backward on, Forward off*		√

Dari hasil pengujian berdasarkan tabel 4.2., dapat disimpulkan bahwa perancangan diagram *ladder* menggunakan metode *state diagram* bejalan dengan lancar dan berhasil.

4.1.4. Pengujian Sistem Eksternal

Sistem eksternal terdiri dari tiga buah sistem tambahan, yaitu *intial system lamp indicator system* dan hitam biru *system*.

Tabel 4.3. Hasil pengujian

No	Pengujian	Cara pengujian	Normal
1	*Start*	Menekan tombol start dan melihat apakah sistem sudah berjalan	√
2	*Reset*	Menekan tombol reset dan melihat apakah sistem akan berhenti	√
3	*Return*	Menekan tombol return dengan tombol reset dan melihat apakasi sistem kembali ke posisi awal sistem	√
4	Indikator merah	Melihat apakah indikator menyala ketika sistem sedang aktif dan beroperasi	√
5	Indikator kuning	Melihat apakah indikator menyala ketika sistem sedang aktif dan tidak beroperasi	√
6	Indikator hijau	Melihat apakah indikator menyala ketika sistem sedang tidak aktif dan tidak beroperasi	√
7	Hitam	Melihat apakah benda kerja hitam dijatuhkan di tempat seharusnya	√

8	Biru	Melihat apakah benda kerja biru dijatuhkan di tempat seharusnya	√

Berdasarkan tabel 4.3., dapat disimpulkan bahwa sistem eksternal berjalan sesuai dengan keinginan.

4.2. Analisa Hasil

Analisa hasil dilakukan untuk menganalisa hasil dari program diagram ladder yang telah dibuat. Hal-hal yang dianalisa mencakup jumlah relay yang digunakan, pencatatan waktu dalam memproses benda kerja dan membandingkan dengan hasil dari tugas akhir tahun sebelumnya.

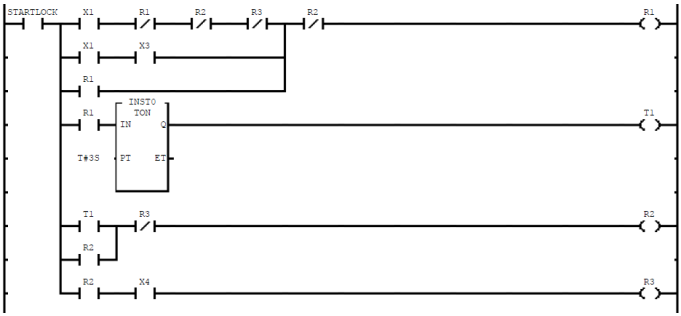
4.2.1. Implementasi Metode *State Diagram*

Implementasi dari metode *state diagram* untuk *factory automatic trainer* sesuai dengan yang telah dirancang *diawal* proses. Total *relay* yang digunakan dalam pembuatan diagram *ladder* FAT adalah sebanyak 17 *relay* (tidak termasuk sistem eksternal). Berikut adalah penjabaran dari penggunaan *relay*.

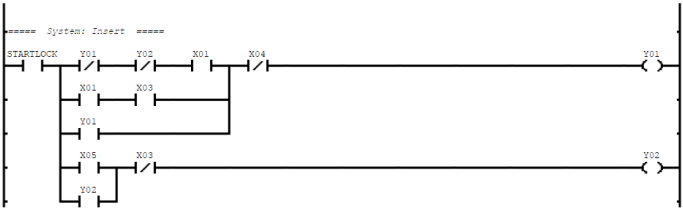
Tabel 4.4. Penggunaan relay

No	Sub proses	State diagram	Konvensional
1	*Insert*	2	4
2	*Non metal separator*	2	4
3	*Pick and place*	3	7
4	*Stopper*	3	6
5	*Conveyor 1*	2	2
6	*Conveyor 2*	2	2
7	*Line movement*	3	7
Total		17	32

Jika dibandingkan dengan hasil pembuatan ladder diagram yang tidak menggunakan metode atau secara konvensional, didapatkan jumlah penggunaan relay sebanyak 32 relay yang berarti 15 relay lebih banyak dibanding dengan menggunakan metode. Begitu juga jika dibandingkan dengan tugas akhir sebelumnya yaitu dengan judul "Konstruksi *Ladder Diagram* Menggunakan Metode *State Diagram* Pada *Factory Automatic Trainer"*, oleh Gustafian Fawally Al-Barr[7] yang hanya memproses satu benda satu per satu, tugas akhir tersebut mencatatkan penggunaan *relay* sebanyak 23 buah yang berarti tugas akhir saat ini menggunakan *relay* yang lebih sedikit.



Gambar 4.5. Ladder diagram konvensional sub proses 1



Gambar 4.6. Ladder diagram dengan metode sub proses 1

Tampak pada gambar 4.6. dan gambar 4.7. perbandingan penggunaan relay ladder diagram konvensional dan ladder diagram yang menggunakan metode. Pada ladder diagram konvensional digunakan sebanyak empat relay, sedangkan ladder diagram dengan metode hanya menggunakan dua relay saja. Ladder diagram konvensional keseluruhan akan ditampilkan dilampiran sebagai pembanding dengan ladder diagram lainnya.

Begitu juga jika dibandingkan dengan tugas akhir sebelumnya yaitu dengan judul "Konstruksi *Ladder Diagram* Menggunakan Metode *State Diagram* Pada *Factory Automatic Trainer",* oleh Gustafian Fawally Al-Barr[7] yang hanya memproses satu benda satu per satu, tugas akhir tersebut mencatatkan penggunaan *relay* sebanyak 23 buah yang berarti tugas akhir saat ini menggunakan *relay* yang lebih sedikit.

4.2.2. Waktu proses

Pencatatan waktu proses dalam pengerjaan tugas akhir ini dilakukan dalam beberapa tahap, yaitu:

1. Benda kerja biru satu per satu
2. Benda kerja hitam satu per satu
3. Benda kerja metal satu per satu

4. Multi proses berurutan
5. Multi proses tidak berurutan (*random*)

Masing-masing tahap tersebut dilakukan sebanyak lima kali.

Tabel 4.5. Hasil pencatatan waktu tiap benda kerja dalam detik

Data	Tahap 1	Tahap 2	Tahap 3	Tahap 4	Tahap 5
1	26,07	27.82	9.32	83,22	83,38
2	26,11	27,92	9,72	82,93	77,73
3	26,21	27,71	9,14	83,65	83,52
4	26,53	27,84	9,67	83,07	80,58
5	26,44	27,98	9,78	83,08	80,72
Rata-rata	26,27	27,85	9,52	83,19	81,18

Gambar 4.7. Waktu proses

Berdasarkan gambar 4.5., rata-rata waktu yang dibutuhkan untuk benda hitam dan biru jika diproses satu per satu adalah sekitar 27 detik. Benda kerja metal mendapatkan waktu rata-rata sekitar 9 detik. Sedangkan untuk multi proses mendapatkan waktu rata-rata sekitar 82 detik.

Hasil pencatatan waktu untuk benda kerja biru jika dibandingkan dengan tugas akhir sebelumnya mencatatkan waktu yang lebih lama yaitu sekitar 3-4 detik. Untuk benda kerja hitam mencatatkan waktu lebih lama sekitar satu detik. Namun jika dilakukan pengerjaan sebanyak tiga buah benda kerja hitam dan biru (total 6 benda kerja) untuk tugas akhir sebelumnya akan mencatatkan waktu sekitar 149,51 detik atau 2 menit 29,51 detik. Jika dibandingkan dengan pengerjaan sebanyak tiga buah benda hitam, biru dan metal (total 9 benda kerja) untuk tugas akhir saat ini akan mencatatkan waktu hanya sekitar 81,18-83,19 detik atau 1 menit 21,18 detik sampai 1 menit 23,19 detik. Dari hasil tersebut dengan kondisi benda yang lebih banyak dan dalam tugas akhir ini mendapatkan selisih waktu sekitar 67 detik lebih cepat dibanding dengan tugas akhir sebelumnya. Hal tersebut disebabkan pada tugas akhir ini *factory automatic trainer* dapat memproses benda terus menerus tanpa harus menunggu benda sebelumnya selesai diproses.

BAB 5
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Penggunaan metode state diagram dalam pembuatan *ladder diagram* akan membuat *developer* lebih terarah dalam mengerjaan perangkat otomasi yang sedang diprogram. Hal tersebut disebabkan adanya teori yang membuat pengerjaan ladder diagram lebih pasti, tanpa harus menerka-nerka dalam pembuatan *ladder diagram*. Selain itu dalam pembuatan ladder diagram factory automatic trainer juga dapat meminimalisir penggunaan *relay* yaitu sebanyak 17 *relay*, yang jika dibandingkan dengan membuat tanpa menggunakan metode sebanyak 32 *relay*.

Waktu rata-rata proses kontinyu yang didapatkan sekitar 82 detik, yang jika dibandingkan dengan tugas akhir tahun 2018 yang miliki waktu rata-rata sekitar 149 detik. Rata-rata tersebut lebih cepat sekitar 1 menit 22 detik, karena pada tugas akhir ini benda yang diproses oleh factory automatic trainer lebih dari satu benda dalam satu kali proses, tanpa harus menunggu satu benda selesai dari awal sampai akhir.

5.2. Saran

Dalam pembuatan diagram *ladder* terdapat beberapa macam metode yang masing-masing mempunyai keunggulannya tersendiri. Hal selanjutnya yang dapat dilakukan adalah membandingkan beberapa macam metode tersebut. Dengan membandingkan beberapa metode, maka bisa diketahui sisi positif dan negatif untuk masing-masing metode untuk sebuah sistem yang sama. Sehingga dapat diketahui metode mana yang lebih efektif dalam pembuatan sebuah digram *ladder*.

DAFTAR PUSTAKA

[1] Chungpa EMT Co., Ltd, ”CPE-AT8030N Factory Automatic Trainer User’s Manual”, Korea: Chungpa EMT Co., Ltd.

[2] Palito, Andhiko F, ”Pembuatan Sistem Kontrol Mesin *Detect Apply Grease* K21 Berbasis PLC Mitsubishi FX3U di PT Aisin Indonesia”, Jakarta: Politeknik Manufaktur Astra, 2016.

[3] Bolton, W, ”Sistem Instrumentasi dan Sistem Kontrol”, Jakarta: Erlangga, 2009.

[4] Hanssen, Dag H., “Programmable Logic Controller: A Practical Approach to IEC 61131-3 Using Codesys”, United Kingdom: John Wiley & Sons, 2015.

[5] Iskandar, Eka, Mochammad Rameli dan Nicco, “Ladder Diagram based on State Diagram for Selection and Assembling Part on Dual Conveyor”, Surabaya: Sepuluh Nopember Institute of Tecnologi, 2017.

[6] Menesis, Stamatios and George Nikolakopoulos, ”Introduction to Industrial Automation”, Boca Raton: Taylor & Francis Group, 2018.

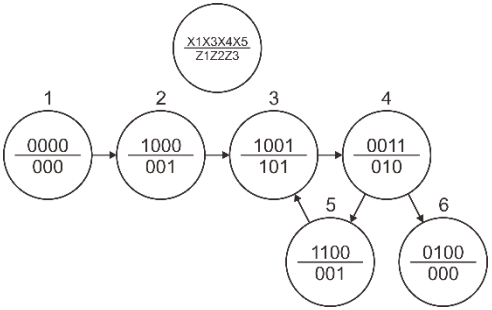
[7] Al-Barr, Gustafian Fawally, “Konstuksi *Ladder Diagram* Menggunakan Metode *State Diagram* pada *Factory Automatic Trainer*”, Surabaya: Institut Teknologi Sepuluh Nopember, 2018.

LAMPIRAN

1. Sub sistem 1
2. Sub sistem 2
3. Sub sistem 3
4. Sub sistem 4
5. Sub sistem 5
6. Sub sistem 6
7. Sub sistem 7
8. Validasi Sub sistem 1
9. Validasi Sub sistem 2
10. Validasi Sub sistem 3
11. Validasi Sub sistem 4
12. Validasi Sub sistem 5
13. Validasi Sub sistem 6
14. Validasi Sub sistem 7
15. Lamp indicator system
16. Hitam biru system
17. *Ladder diagram* konvensional

Sub Sistem 1

State Diagram I/O



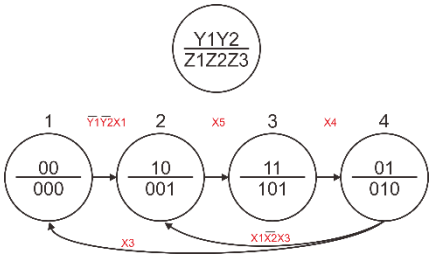
Primitive Flow Table

Row	Input						Output		
	0000	1000	1001	0011	1100	0100	Z1	Z2	Z3
1	**1**	2	-	-	-	-	0	0	0
2	-	**2**	3	-	-	-	0	0	1
3	-	-	**3**	4	-	-	1	0	1
4	-	-	-	**4**	5	6	0	1	0
5	-	-	3	-	**5**	-	0	0	1
6	-	-	-	-	-	**5**	0	0	0

Merged Flow Tabel

Row	Input						Output			Relay	
	0000	1000	1001	0011	1100	0100	Z1	Z2	Z3	Y1	Y2
1,6	**1**	2	-	-	-	**6**	0	0	0	0	0
2,5	-	**2**	3	-	**5**	-	0	0	1	1	0
3	-	-	**3**	4	-	-	1	0	1	1	1
4	-	-	-	**4**	5	6	0	1	0	0	1

State Diagram R/O



Switching Function

$Y1 = [(\overline{Y1} * \overline{Y2} * X1) + (X1 * X3) + Y1] * \overline{X4}$

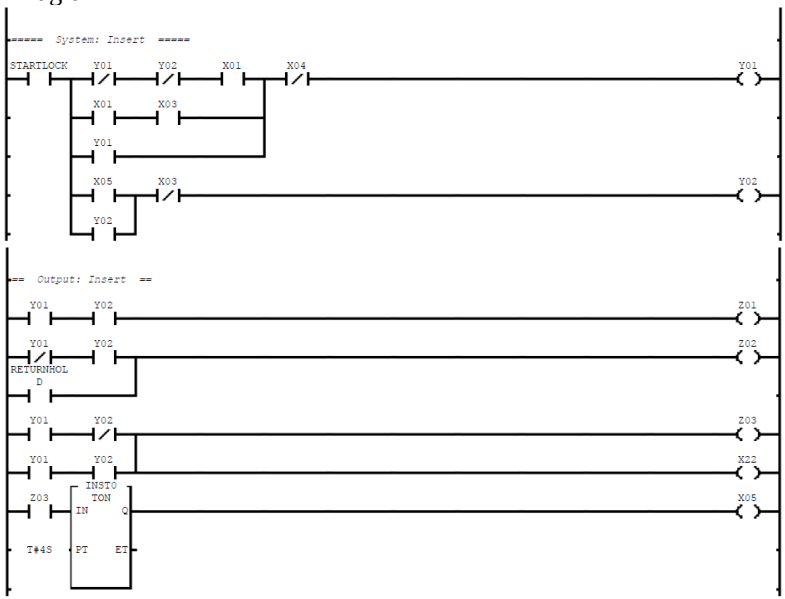
$Y2 = (X5 + Y2) * \overline{X3}$

$Z1 = Y1 * Y2$

$Z2 = \overline{Y1} * Y2$

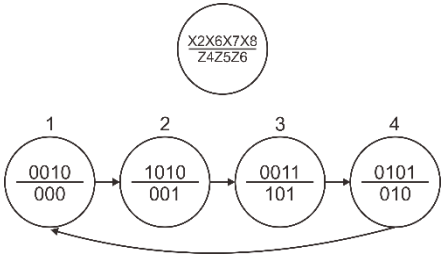
$Z3 = (Y1 * \overline{Y2}) + (Y1 * Y2)$

Ladder Diagram



Sub Sistem 2

State Diagram I/O



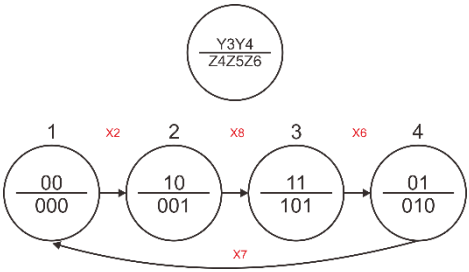
Primitive Flow Table

Row	Input				Output		
	0010	1010	0011	0101	Z4	Z5	Z6
1	**1**	2	-	-	0	0	0
2	-	**2**	3	-	0	0	1
3	-	-	**3**	4	1	0	1
4	1	-	-	**4**	0	1	0

Merged Flow Tabel

Row	Input				Output			Relay	
	0010	1010	0011	0101	Z4	Z5	Z6	Y3	Y4
1	**1**	2	-	-	0	0	0	0	0
2	-	**2**	3	-	0	0	1	1	0
3	-	-	**3**	4	1	0	1	1	1
4	1	-	-	**4**	0	1	0	0	1

State Diagram R/O



Switching Function

$Y3 = (X2 + Y3) * \overline{X6}$

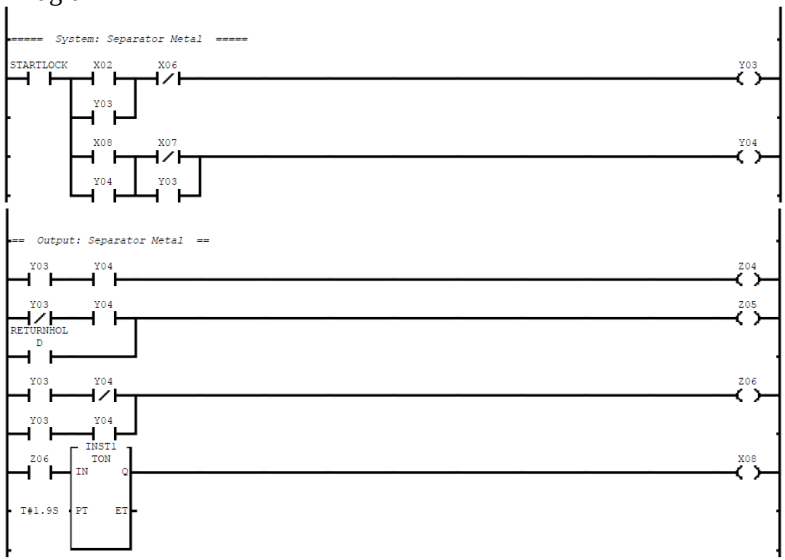
$Y2 = (X8 + Y4) * (\overline{X7} + Y3)$

$Z4 = Y3 * Y4$

$Z5 = \overline{Y3} * Y4$

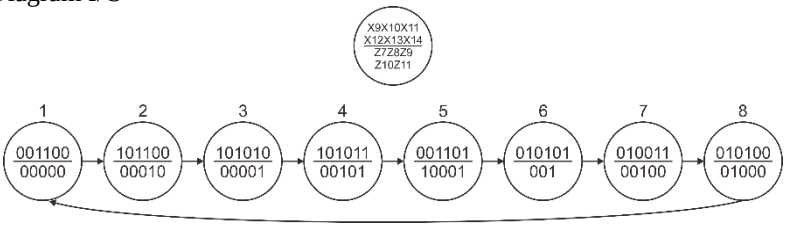
$Z6 = (Y3 * \overline{Y4}) + (Y3 * Y4)$

Ladder Diagram



Sub Sistem 3

State Diagram I/O



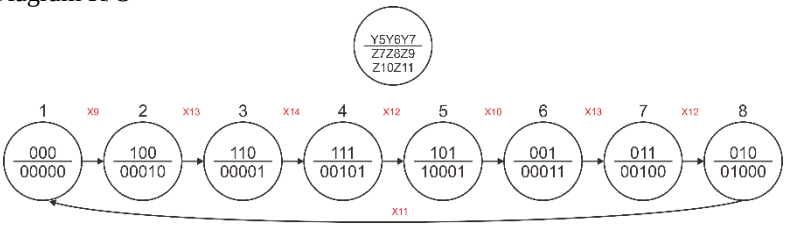
Primitive Flow Table

Row	Input								Output				
	001100	101100	101010	101011	001101	010101	010011	010100	Z7	Z8	Z9	Z10	Z11
1	**1**	2	-	-	-	-	-	-	0	0	0	0	0
2	-	**2**	3	-	-	-	-	-	0	0	0	1	0
3	-	-	**3**	4	-	-	-	-	0	0	0	0	1
4	-	-	-	**4**	5	-	-	-	0	0	1	0	1
5	-	-	-	-	**5**	6	-	-	1	0	0	0	1
6	-	-	-	-	-	**6**	7	-	0	0	0	1	1
7	-	-	-	-	-	-	**7**	8	0	0	1	0	0
8	1	-	-	-	-	-	-	**8**	0	1	0	0	0

Merged Flow Tabel

Row	Input								Output					Relay		
	001100	101100	101010	101011	001101	010101	010011	010100	Z7	Z8	Z9	Z10	Z11	Y5	Y6	Y7
1	**1**	2	-	-	-	-	-	-	0	0	0	0	0	0	0	0
2	-	**2**	3	-	-	-	-	-	0	0	0	1	0	1	0	0
3	-	-	**3**	4	-	-	-	-	0	0	0	0	1	1	1	0
4	-	-	-	**4**	5	-	-	-	0	0	1	0	1	1	1	1
5	-	-	-	-	**5**	6	-	-	1	0	0	0	1	1	0	1
6	-	-	-	-	-	**6**	7	-	0	0	0	1	1	0	0	1
7	-	-	-	-	-	-	**7**	8	0	0	1	0	0	0	1	1
8	1	-	-	-	-	-	-	**8**	0	1	0	0	0	0	1	0

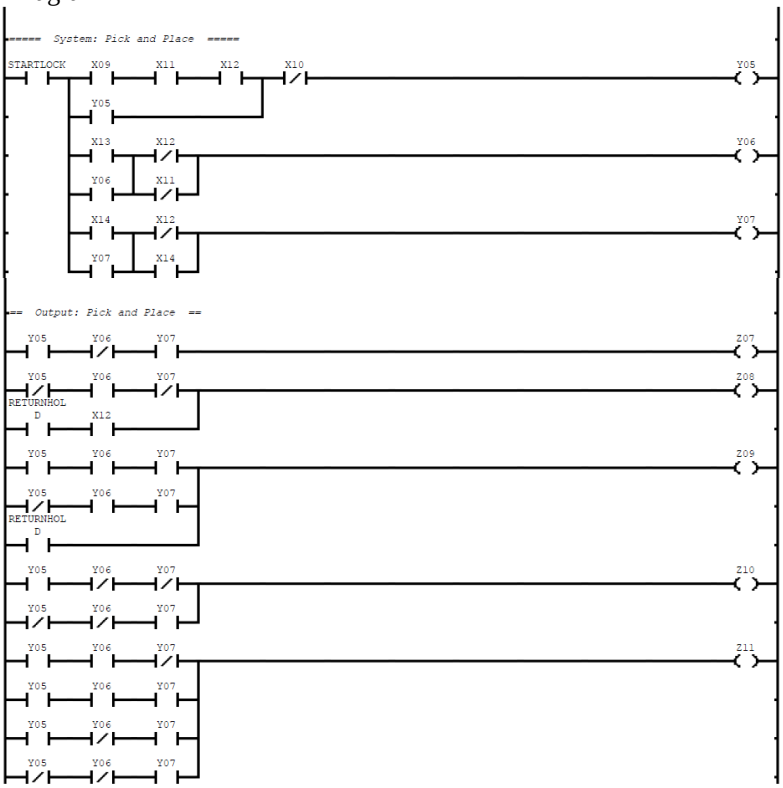
State Diagram R/O



Switching Function

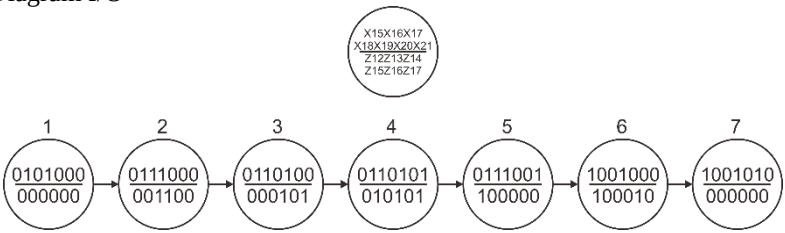
$$Y5 = [(X9 * X11 * X12) + Y5] * \overline{X10}$$
$$Y6 = (X13 + Y6) * (\overline{X12} + \overline{X11})$$
$$Y7 = (X14 + Y7) * (\overline{X12} + X14)$$
$$Z7 = Y5 * \overline{Y6} * Y7$$
$$Z8 = \overline{Y5} * Y6 * Y7$$
$$Z9 = (Y5 * Y6 * Y7) + (\overline{Y5} * Y6 * Y7)$$
$$Z10 = (Y5 * \overline{Y6} * \overline{Y7}) + (\overline{Y5} * \overline{Y6} * Y7)$$
$$Z11 = (Y5 * Y6 * \overline{Y7}) + (Y5 * Y6 * Y7) + (Y5 * \overline{Y6} * Y7) + (\overline{Y5} * \overline{Y6} * Y7)$$

Ladder Diagram



Sub Sistem 4

State Diagram I/O



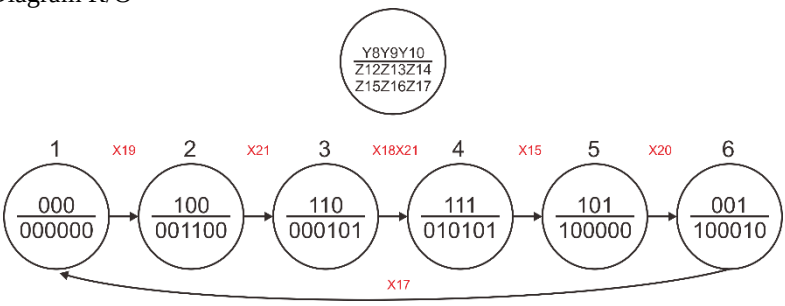
Primitive Flow Table

Row	Input (X15X16X17X18X19X20X21)							Output					
	1001010	0111000	0110100	0110101	0111001	1001000	1001010	Z12	Z13	Z14	Z15	Z16	Z17
1	**1**	2	-	-	-	-	-	0	0	0	0	0	0
2	-	**2**	3	-	-	-	-	0	0	1	1	0	0
3	-	-	**3**	4	-	-	-	0	0	0	1	0	1
4	-	-	-	**4**	5	-	-	0	1	0	1	0	1
5	-	-	-	-	**5**	6	-	1	0	0	0	0	0
6	-	-	-	-	-	**6**	7	1	0	0	0	1	0
7	1	-	-	-	-	-	**7**	0	0	0	0	0	0

Merged Flow Tabel

Row	Input (X15X16X17X18X19X20X21)							Output						Relay		
	1001010	0111000	0110100	0110101	0111001	1001000	1001010	Z12	Z13	Z14	Z15	Z16	Z17	Y8	Y9	Y10
1	**1**	2	-	-	-	-	**7**	0	0	0	0	0	0	0	0	0
2	-	**2**	3	-	-	-	-	0	0	1	1	0	0	1	0	0
3	-	-	**3**	4	-	-	-	0	0	0	1	0	1	1	1	0
4	-	-	-	**4**	5	-	-	0	1	0	1	0	1	1	1	1
5	-	-	-	-	**5**	6	-	1	0	0	0	0	0	1	0	1
6	-	-	-	-	-	**6**	7	1	0	0	0	1	0	0	0	1

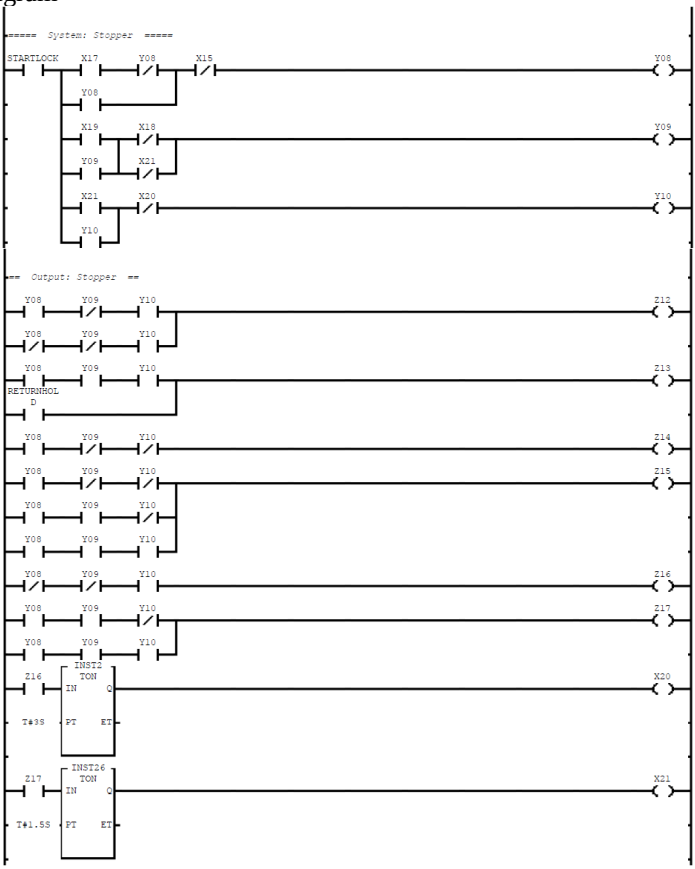
State Diagram R/O



Switching Function

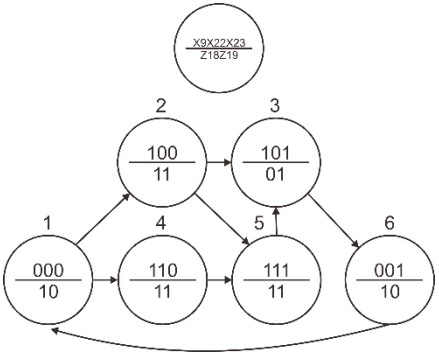
$$Y8 = [(X17 * \overline{Y08}) + Y8] * \overline{X15}$$
$$Y9 = (X19 + Y9) * (\overline{X18} + \overline{X21})$$
$$Y10 = (X21 + Y10) * \overline{X20}$$
$$Z12 = (Y8 * \overline{Y9} * Y10) + \left(\overline{Y5} * \overline{Y6} * Y7\right)$$
$$Z13 = Y8 * Y9 * Y10$$
$$Z14 = Y8 * \overline{Y9} * \overline{Y10}$$
$$Z15 = (Y8 * \overline{Y9} * \overline{Y10}) + (Y8 * Y9 * \overline{Y10}) + (Y8 * Y9 * Y10)$$
$$Z16 = \overline{Y8} * \overline{Y9} * Y10$$
$$Z16 = (Y8 * Y9 * \overline{Y10}) + (Y8 * Y9 * Y10)$$

Ladder Diagram



Sub Sistem 5

State Diagram I/O



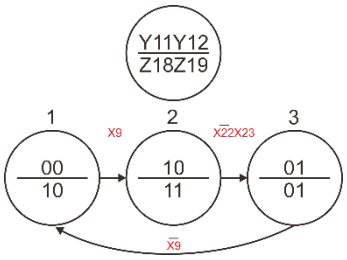
Primitive Flow Table

Row	Input						Output		
	0000	1000	1001	0011	1100	0100	Z1	Z2	Z3
1	**1**	2	-	-	-	-	0	0	0
2	-	**2**	3	-	-	-	0	0	1
3	-	-	**3**	4	-	-	1	0	1
4	-	-	-	**4**	5	6	0	1	0
5	-	-	3	-	**5**	-	0	0	1
6	-	-	-	-	-	**5**	0	0	0

Merged Flow Tabel

Row	Input						Output			Relay	
	0000	1000	1001	0011	1100	0100	Z1	Z2	Z3	Y1	Y2
1,6	**1**	2	-	-	-	**6**	0	0	0	0	0
2,5	-	**2**	3	-	**5**	-	0	0	1	1	0
3	-	-	**3**	4	-	-	1	0	1	1	1
4	-	-	-	**4**	5	6	0	1	0	0	1

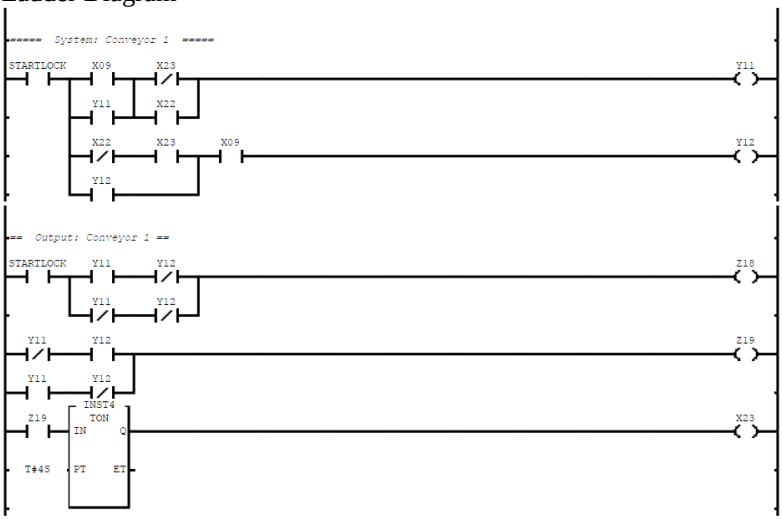
State Diagram R/O



Switching Function

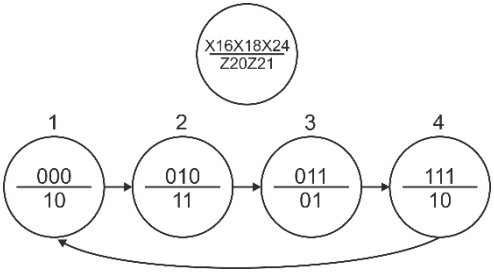
$$Y11 = (X9 + Y11) * (\overline{X6} + X22)$$
$$Y12 = [(\overline{X22} * X23) + Y12] * X9$$
$$Z18 = Y11 * \overline{Y12}$$
$$Z19 = \overline{Y11} * \overline{Y12}$$

Ladder Diagram



Sub Sistem 6

State Diagram I/O



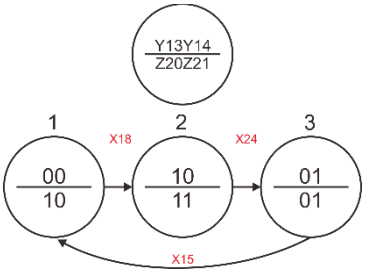
Primitive Flow Table

Row	Input				Output	
	000	010	011	111	Z20	Z21
1	**1**	2	-	-	1	0
2	-	**2**	3	-	1	1
3	-	-	**3**	4	0	1
4	1	-	-	**4**	1	0

Merged Flow Tabel

Row	Input				Output		Relay	
	000	010	011	111	Z20	Z21	Y13	Y14
1,4	**1**	2	-	**4**	1	0	0	0
2	-	**2**	3	-	1	1	1	0
3	-	-	**3**	4	0	1	0	1

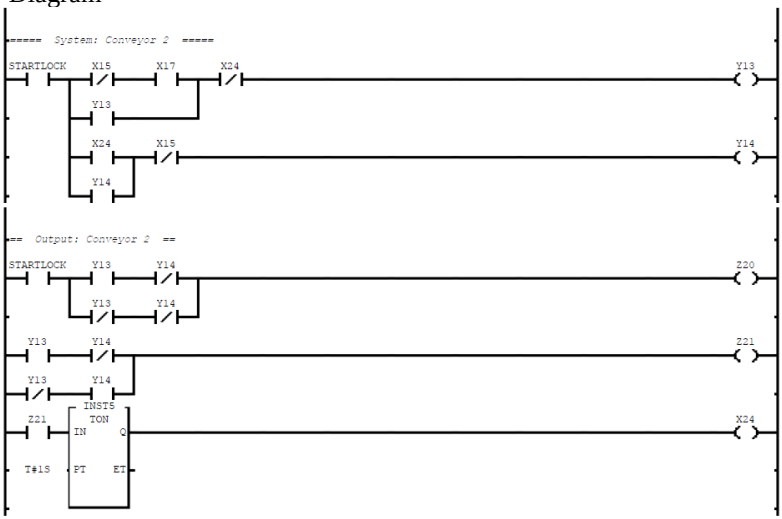
State Diagram R/O



Switching Function

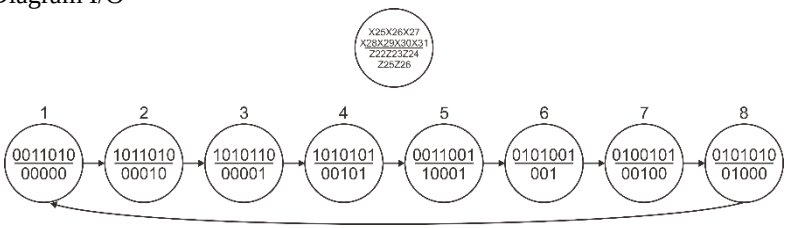
$$Y13 = \left[\left(\overline{X15} * X17\right) + Y13\right] * X24$$
$$Y14 = (X24 + Y14) * \overline{X15}$$
$$Z18 = Y13 * \overline{Y14}$$
$$Z19 = \overline{Y13} * \overline{Y14}$$

Ladder Diagram



Sub Sistem 7

State Diagram I/O



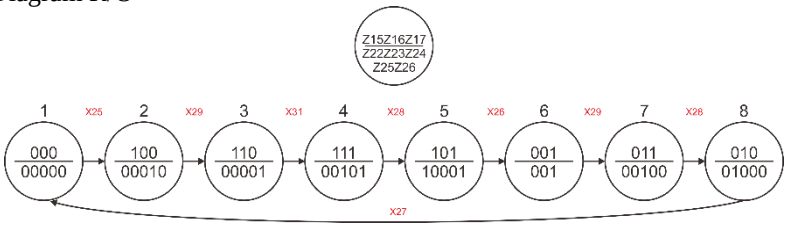
Primitive Flow Table

Row	Input								Output				
	0011010	1011010	1010110	1010101	0011001	0101001	0100101	0101010	Z22	Z23	Z24	Z25	Z26
1	**1**	2	-	-	-	-	-	-	0	0	0	0	0
2	-	**2**	3	-	-	-	-	-	0	0	0	1	0
3	-	-	**3**	4	-	-	-	-	0	0	0	0	1
4	-	-	-	**4**	5	-	-	-	0	0	1	0	1
5	-	-	-	-	**5**	6	-	-	1	0	0	0	1
6	-	-	-	-	-	**6**	7	-	0	0	0	1	1
7	-	-	-	-	-	-	**7**	8	0	0	1	0	0
8	1	-	-	-	-	-	-	**8**	0	1	0	0	0

Merged Flow Tabel

Row	Input								Output					Relay		
	0011010	1011010	1010110	1010101	0011001	0101001	0100101	0101010	Z22	Z23	Z24	Z25	Z26	Y15	Y16	Y17
1	**1**	2	-	-	-	-	-	-	0	0	0	0	0	0	0	0
2	-	**2**	3	-	-	-	-	-	0	0	0	1	0	1	0	0
3	-	-	**3**	4	-	-	-	-	0	0	0	0	1	1	1	0
4	-	-	-	**4**	5	-	-	-	0	0	1	0	1	1	1	1
5	-	-	-	-	**5**	6	-	-	1	0	0	0	1	1	0	1
6	-	-	-	-	-	**6**	7	-	0	0	0	1	1	0	0	1
7	-	-	-	-	-	-	**7**	8	0	0	1	0	0	0	1	1
8	1	-	-	-	-	-	-	**8**	0	1	0	0	0	0	1	0

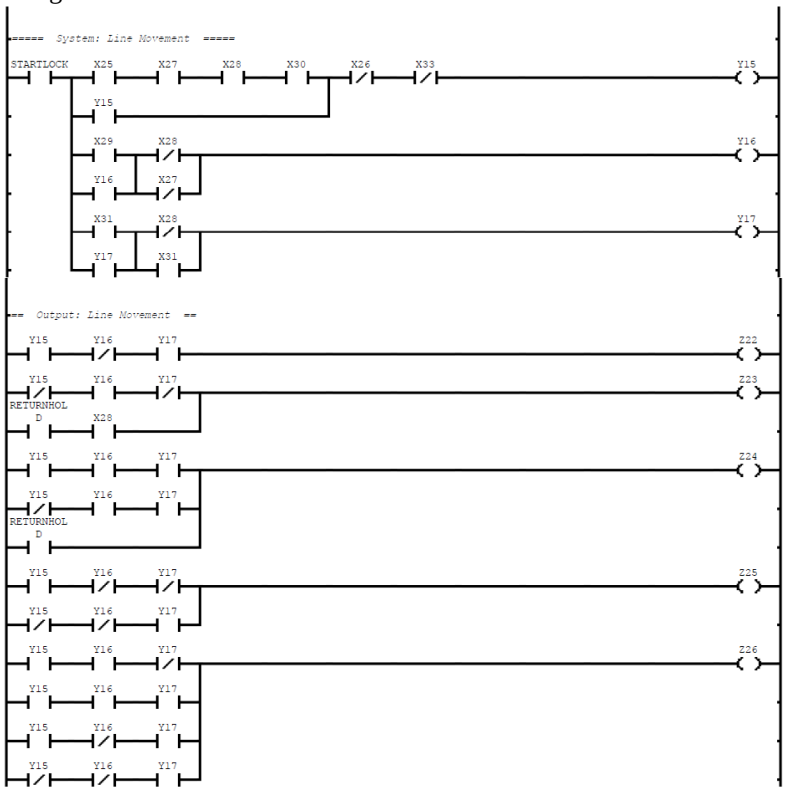
State Diagram R/O



Switching Function

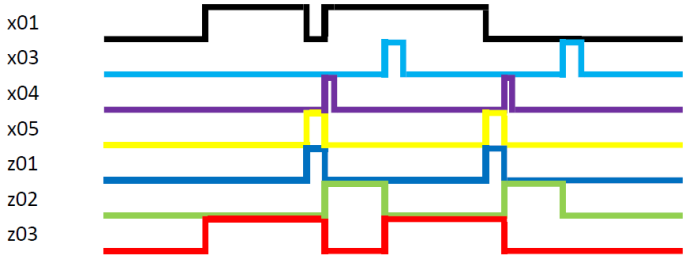
$$Y15 = [(X25 * X27 * X28 * X30) + Y15] * \overline{X26} + \overline{X33}$$
$$Y16 = (X29 + Y16) * (\overline{X28} + \overline{X27})$$
$$Y17 = (X31 + Y17) * (\overline{X28} + X31)$$
$$Z22 = Y15 * \overline{Y16} * Y17$$
$$Z23 = \overline{Y15} * Y16 * Y17$$
$$Z24 = (Y15 * Y16 * Y17) + (\overline{Y15} * Y16 * Y17)$$
$$Z25 = (Y15 * \overline{Y16} * \overline{Y17}) + (\overline{Y15} * \overline{Y16} * Y17)$$
$$Z26 = (Y15 * Y16 * \overline{Y17}) + (Y15 * Y16 * Y17) + (Y15 * \overline{Y16} * Y17) + (\overline{Y15} * \overline{Y16} * Y17)$$

Ladder Diagram

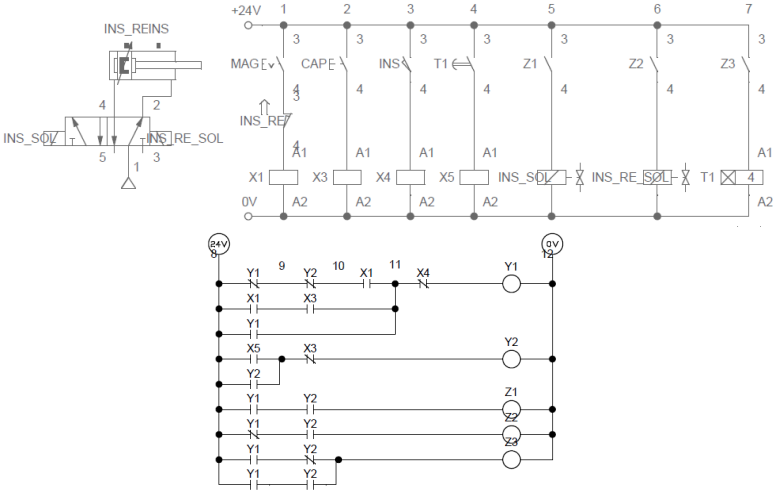


Validasi Sub Sistem 1

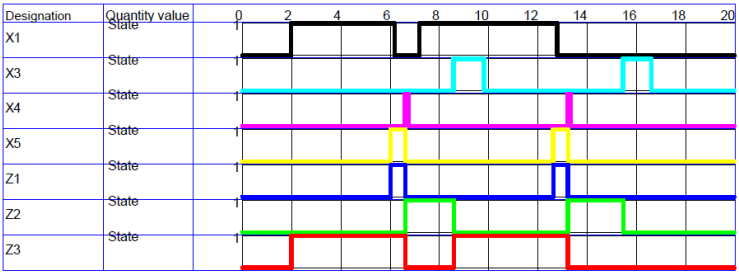
Time chart perancangan



Tampilan pada software validasi

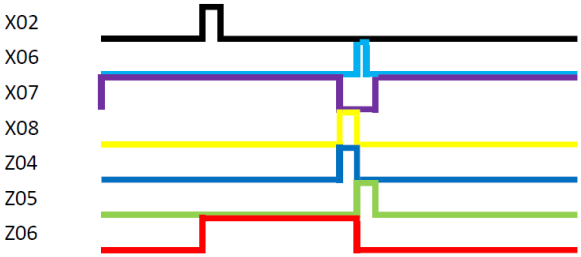


Time chart hasil validasi

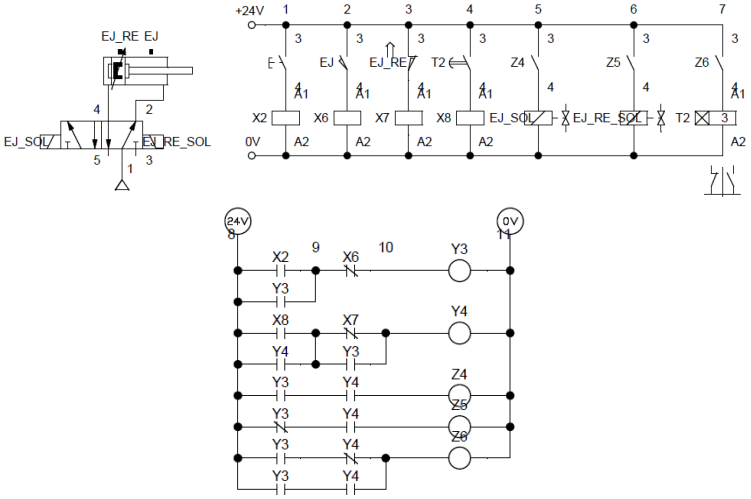


Validasi Sub Sistem 2

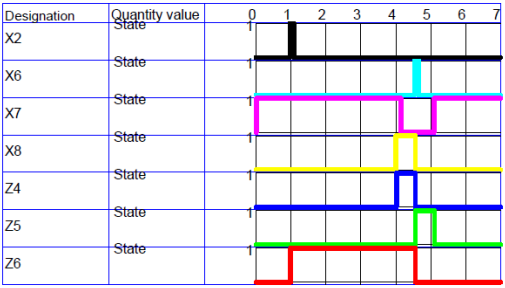
Time chart perancangan



Tampilan pada software validasi

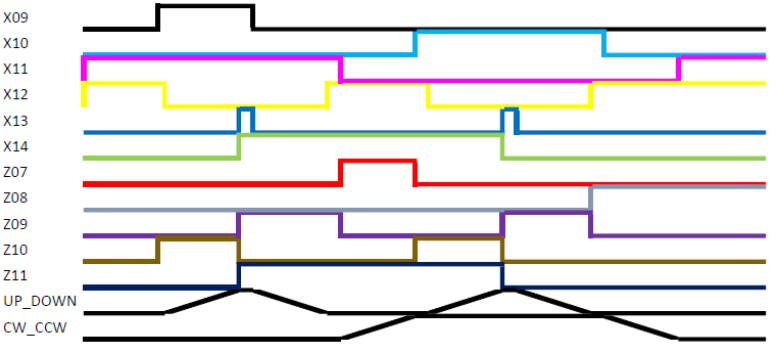


Time chart hasil validasi

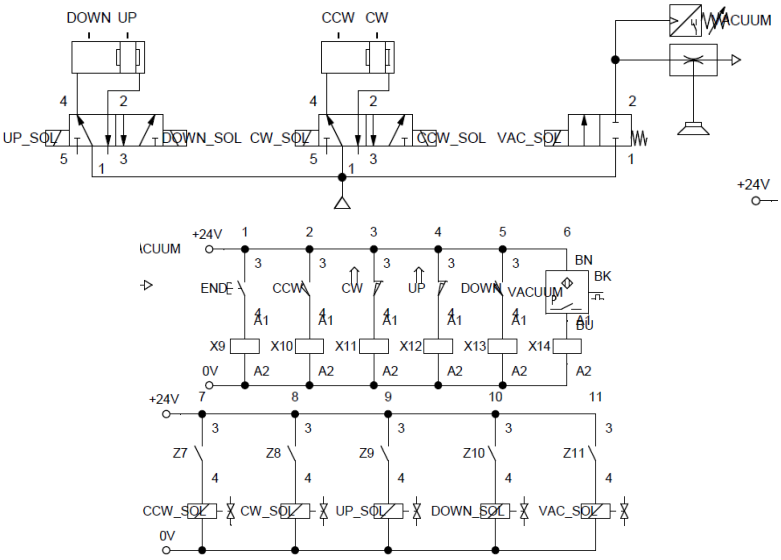


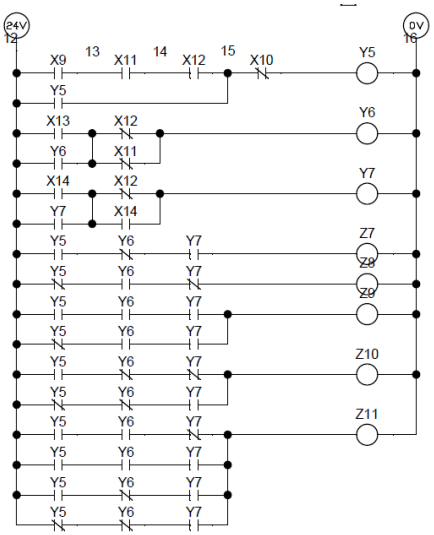
Validasi Sub Sistem 3

Time chart perancangan

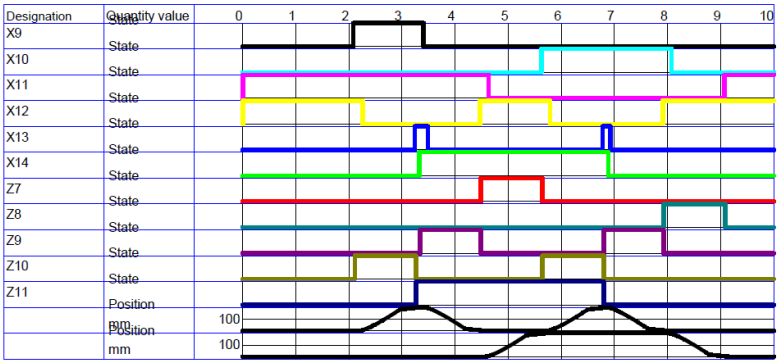


Tampilan pada software validasi



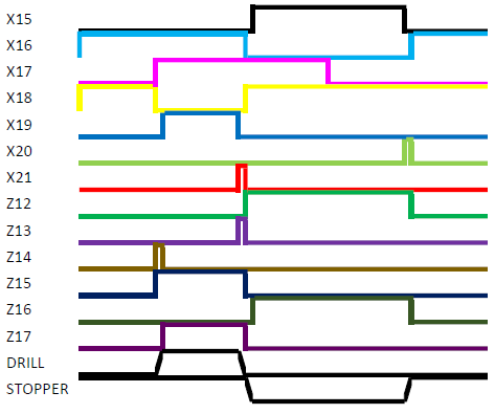


Time chart hasil validasi

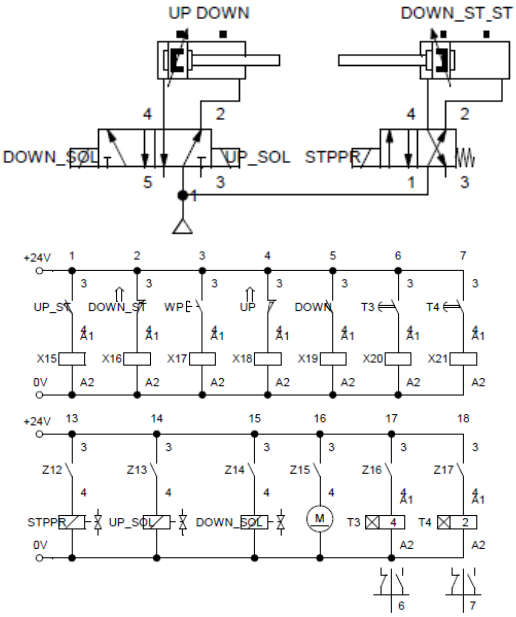


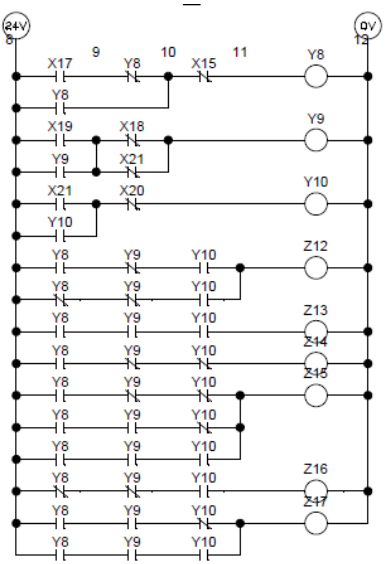
Validasi Sub Sistem 4

Time chart perancangan

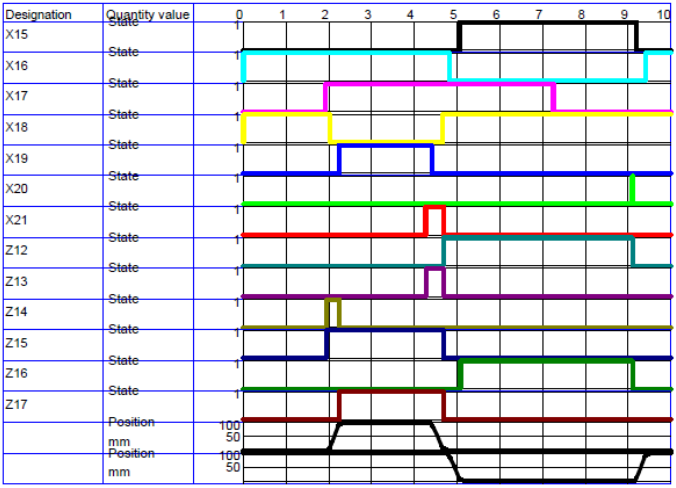


Tampilan pada software validasi





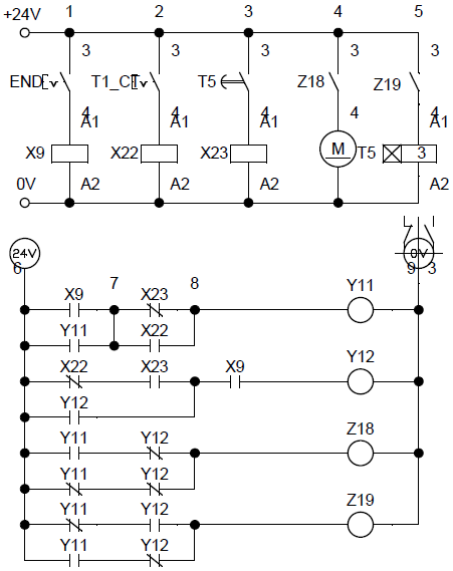
Time chart hasil validasi



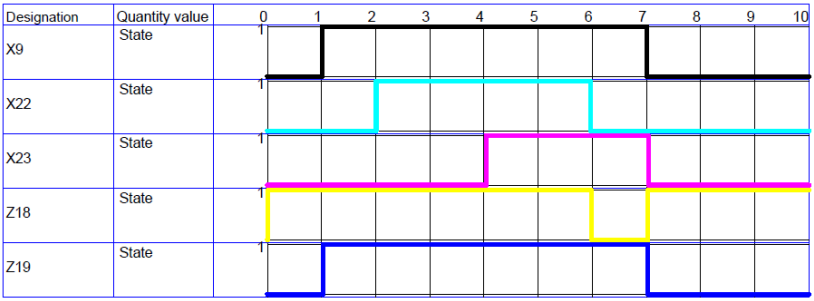
Validasi Sub Sistem 5

Time chart perancangan

Tampilan pada software validasi

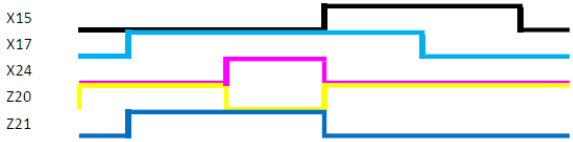


Time chart hasil validasi

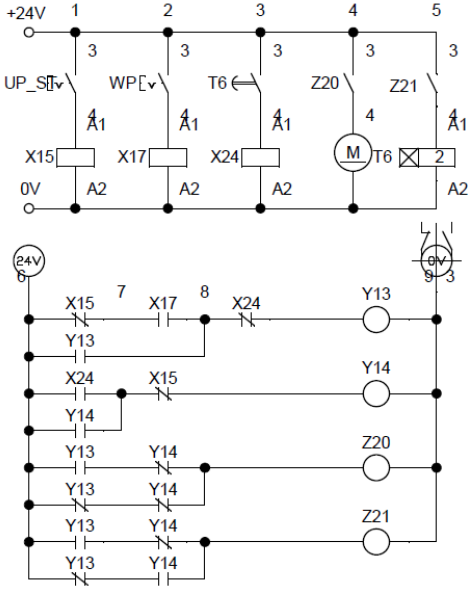


Validasi Sub Sistem 6

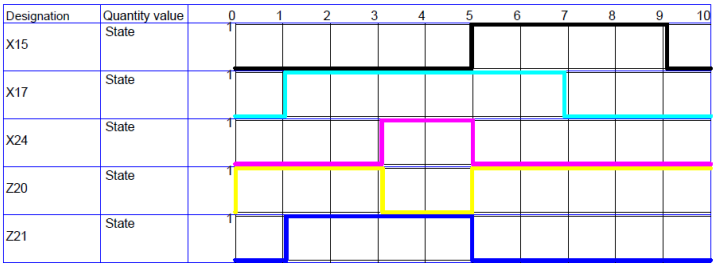
Time chart perancangan



Tampilan pada software validasi

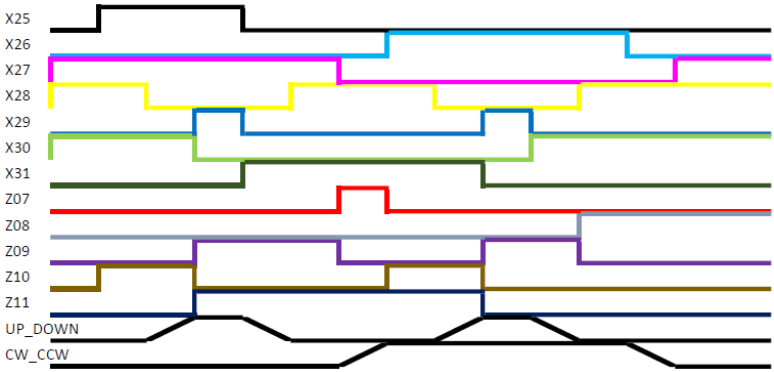


Time chart hasil validasi

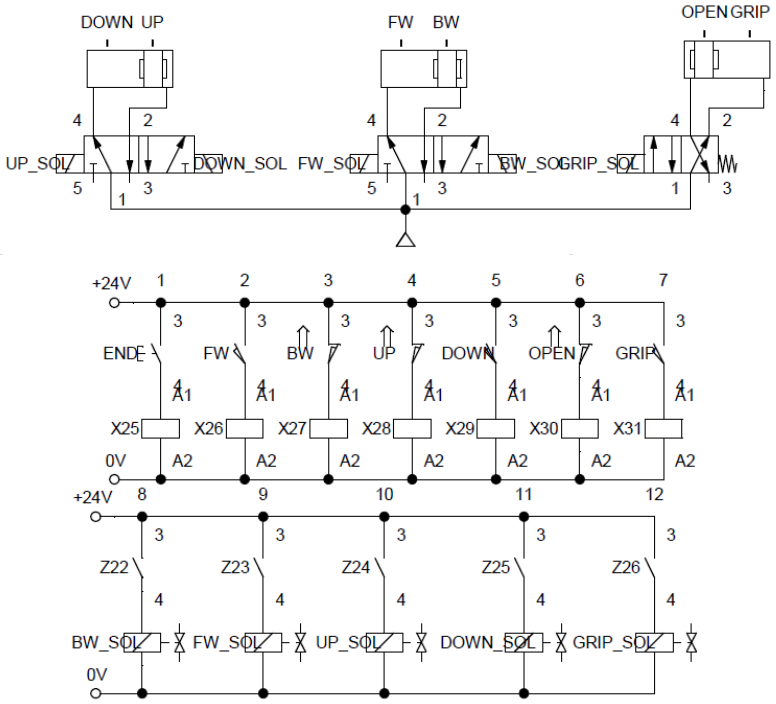


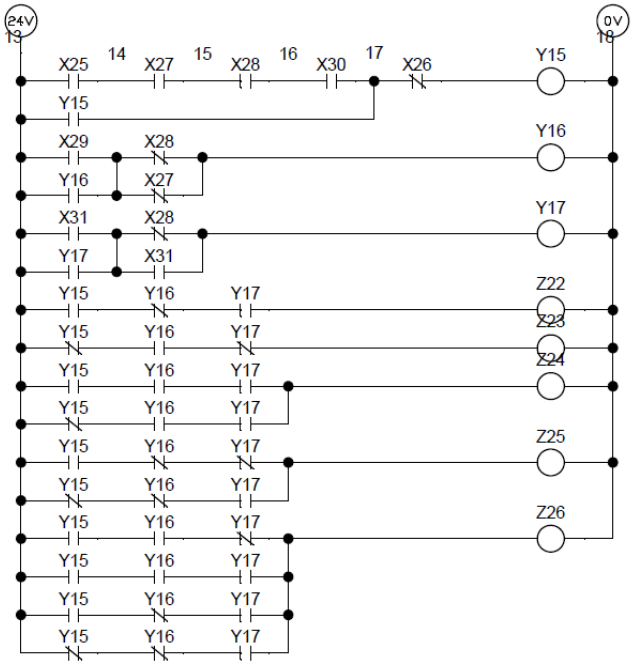
Validasi Sub Sistem 7

Time chart perancangan

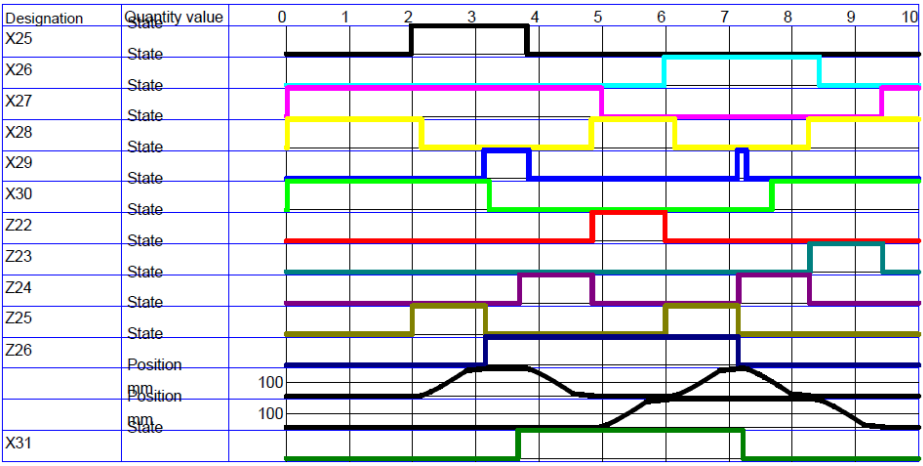


Tampilan pada software validasi

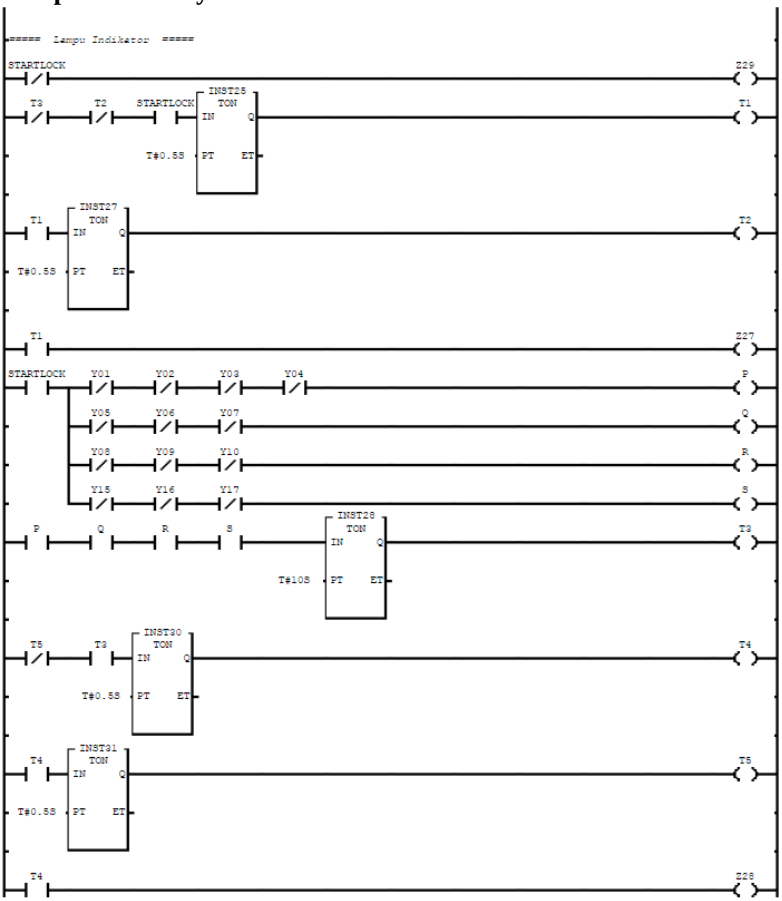




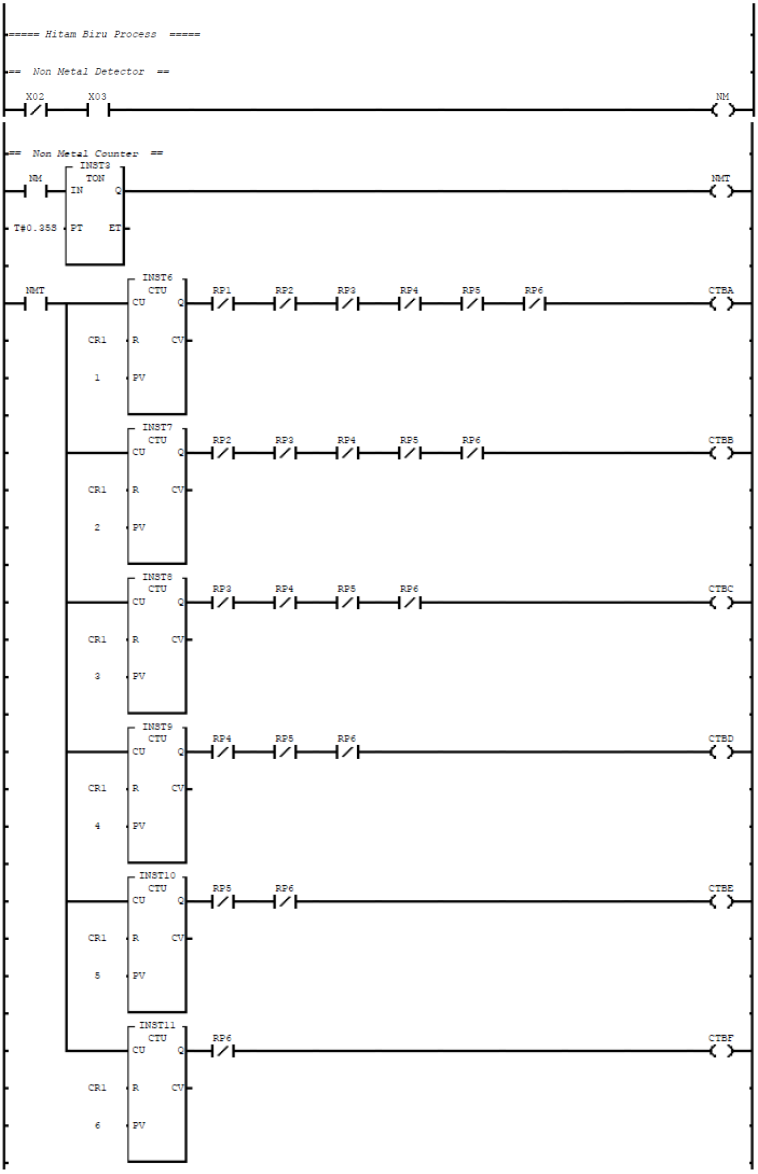
Time chart hasil validasi

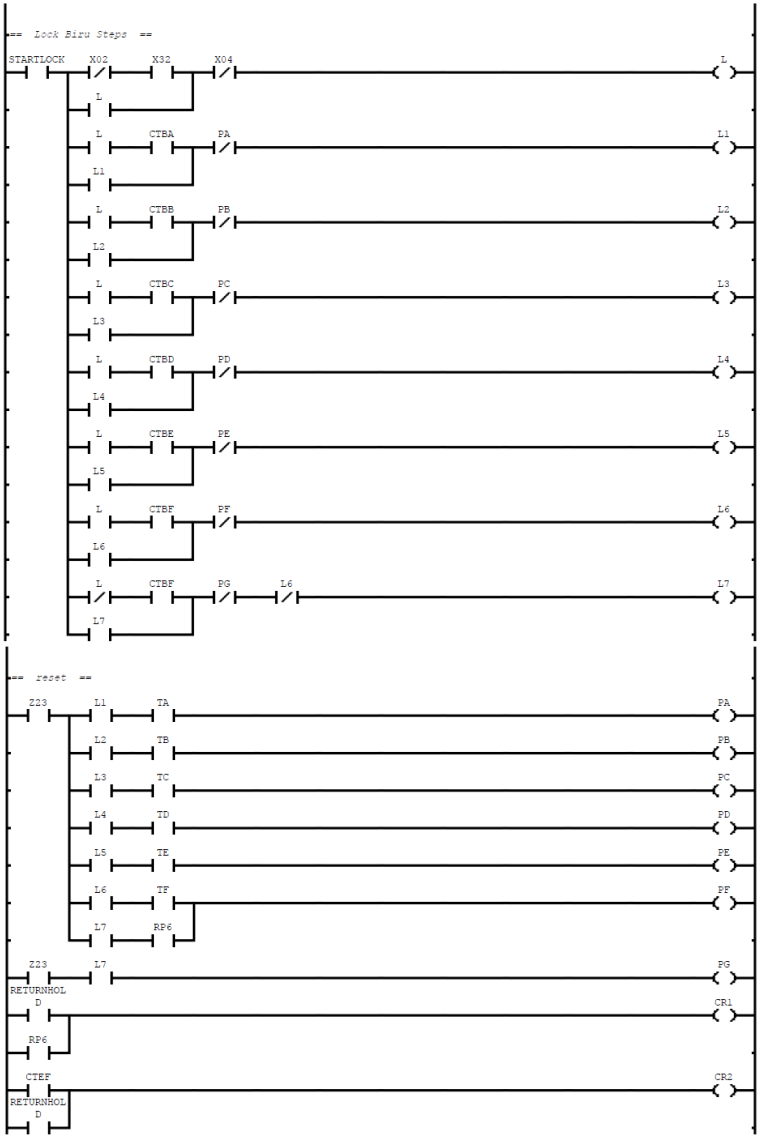


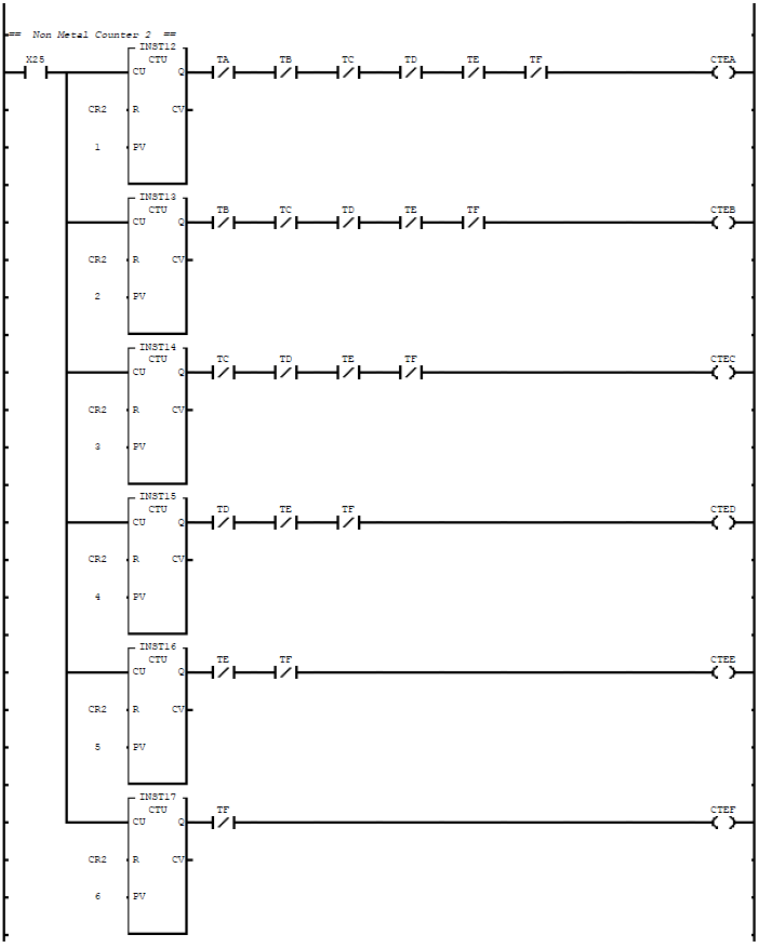
Lamp indicator system

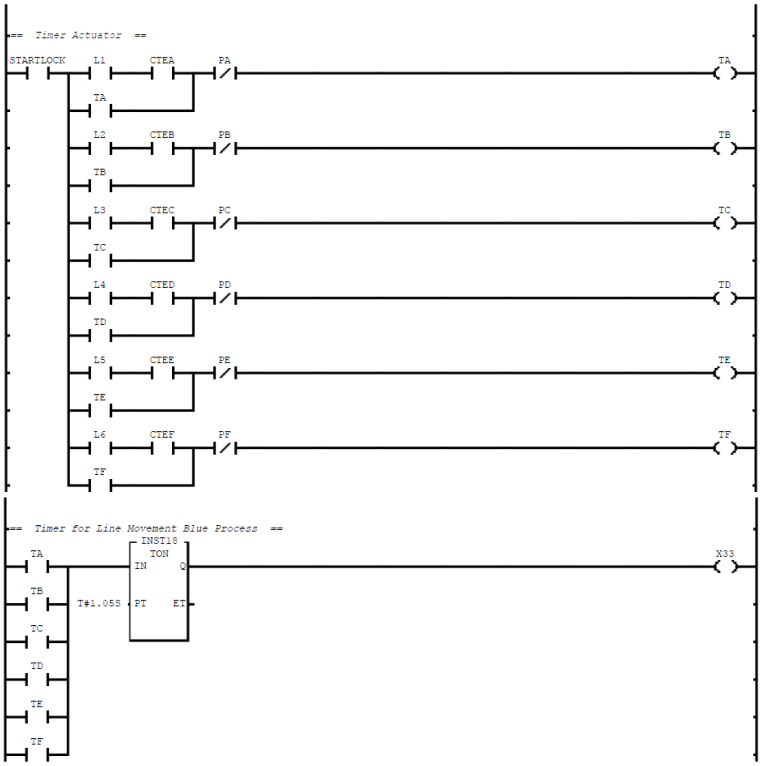


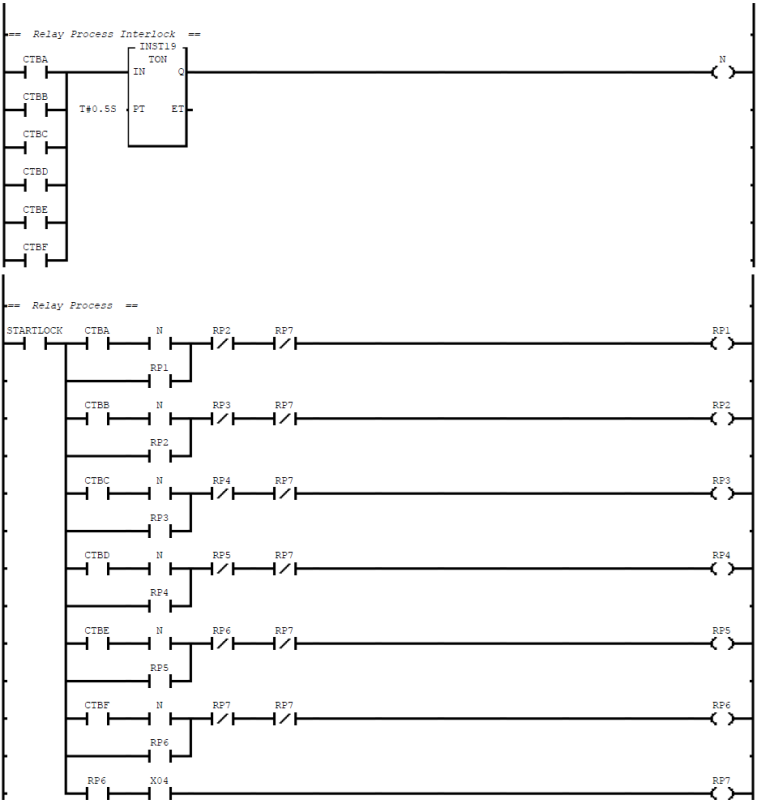
Hitam biru system



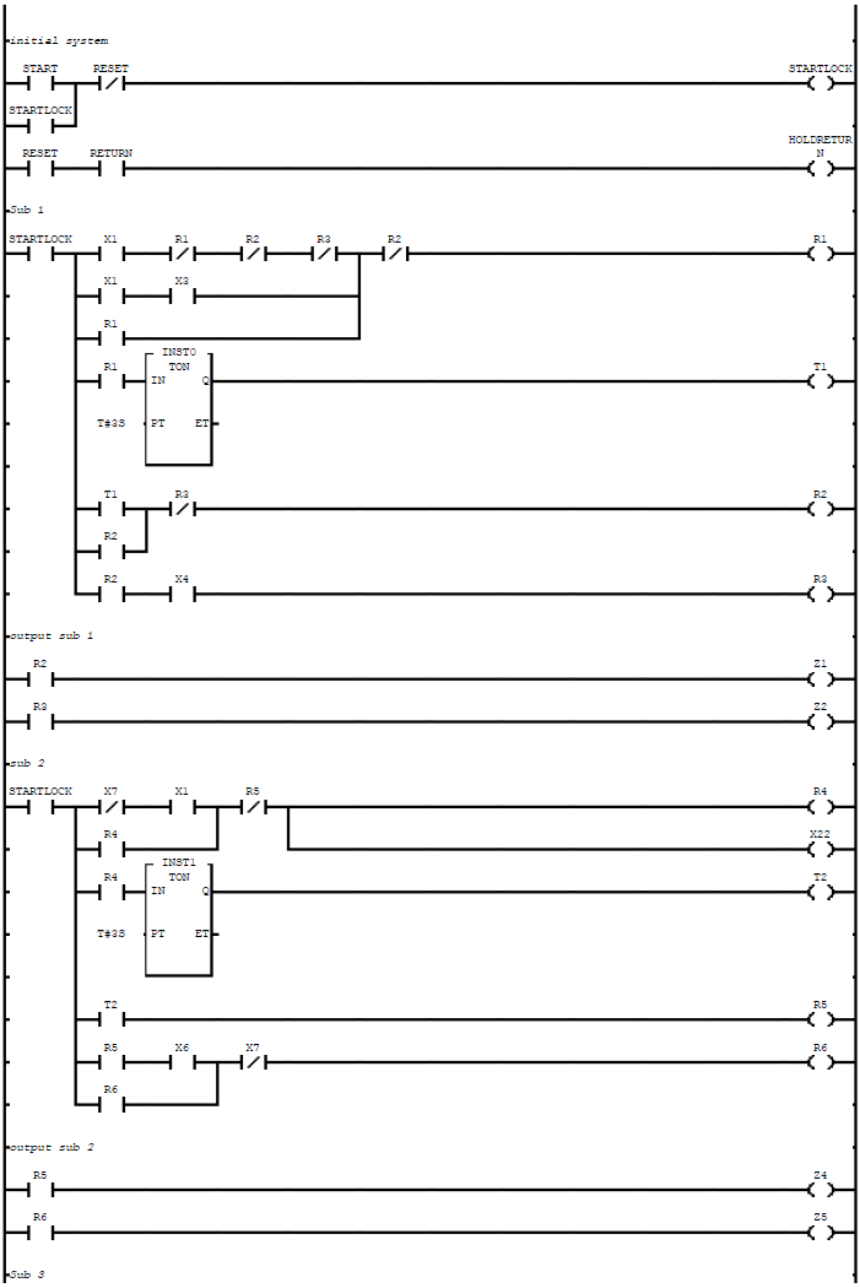

== Lock Biru Steps ==
STARTLOCK
X02
X32
X04
L
L
CTBA
PA
L1
L1
CTBB
PB
L2
L2
CTBC
PC
L3
L3
CTBD
PD
L4
L4
CTBE
PE
L5
L5
CTBF
PF
L6
L6
CTBF
PG
L6
L7
L7
== reset ==
Z23
L1
TA
PA
L2
TB
PB
L3
TC
PC
L4
TD
PD
L5
TE
PE
L6
TF
PF
L7
RP6
Z23
L7
PG
RETURNHOL
D
CR1
RP6
CTEF
CR2
RETURNHOL
D

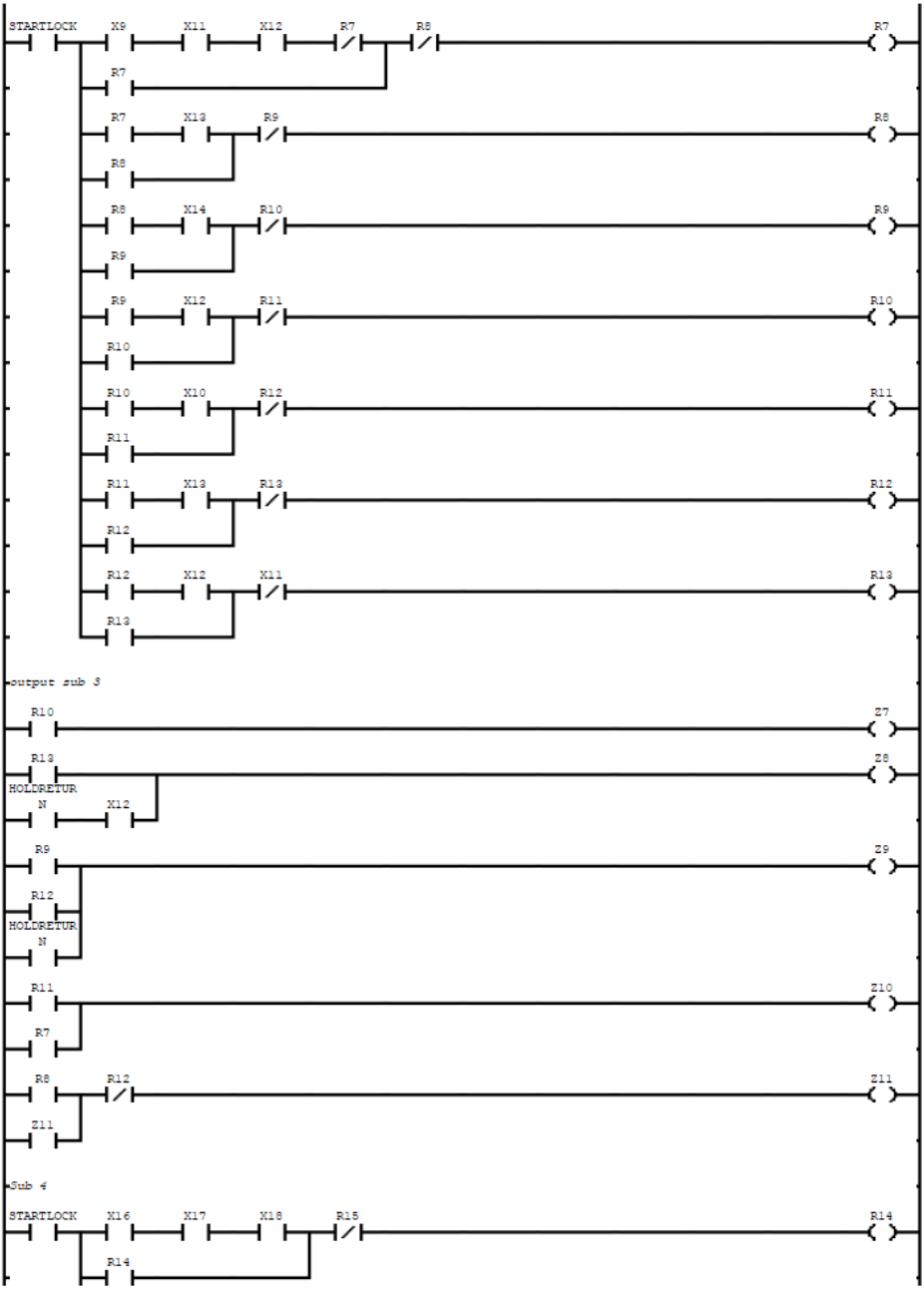

== Non Metal Counter 2 ==
X25
INST12
CTU
CU
Q
R
CV
PV
CR2
1
TA
TB
TC
TD
TE
TF
CTEA
INST13
CTU
CU
Q
R
CV
PV
CR2
2
TB
TC
TD
TE
TF
CTEB
INST14
CTU
CU
Q
R
CV
PV
CR2
3
TC
TD
TE
TF
CTEC
INST15
CTU
CU
Q
R
CV
PV
CR2
4
TD
TE
TF
CTED
INST16
CTU
CU
Q
R
CV
PV
CR2
5
TE
TF
CTEE
INST17
CTU
CU
Q
R
CV
PV
CR2
6
TF
CTEF

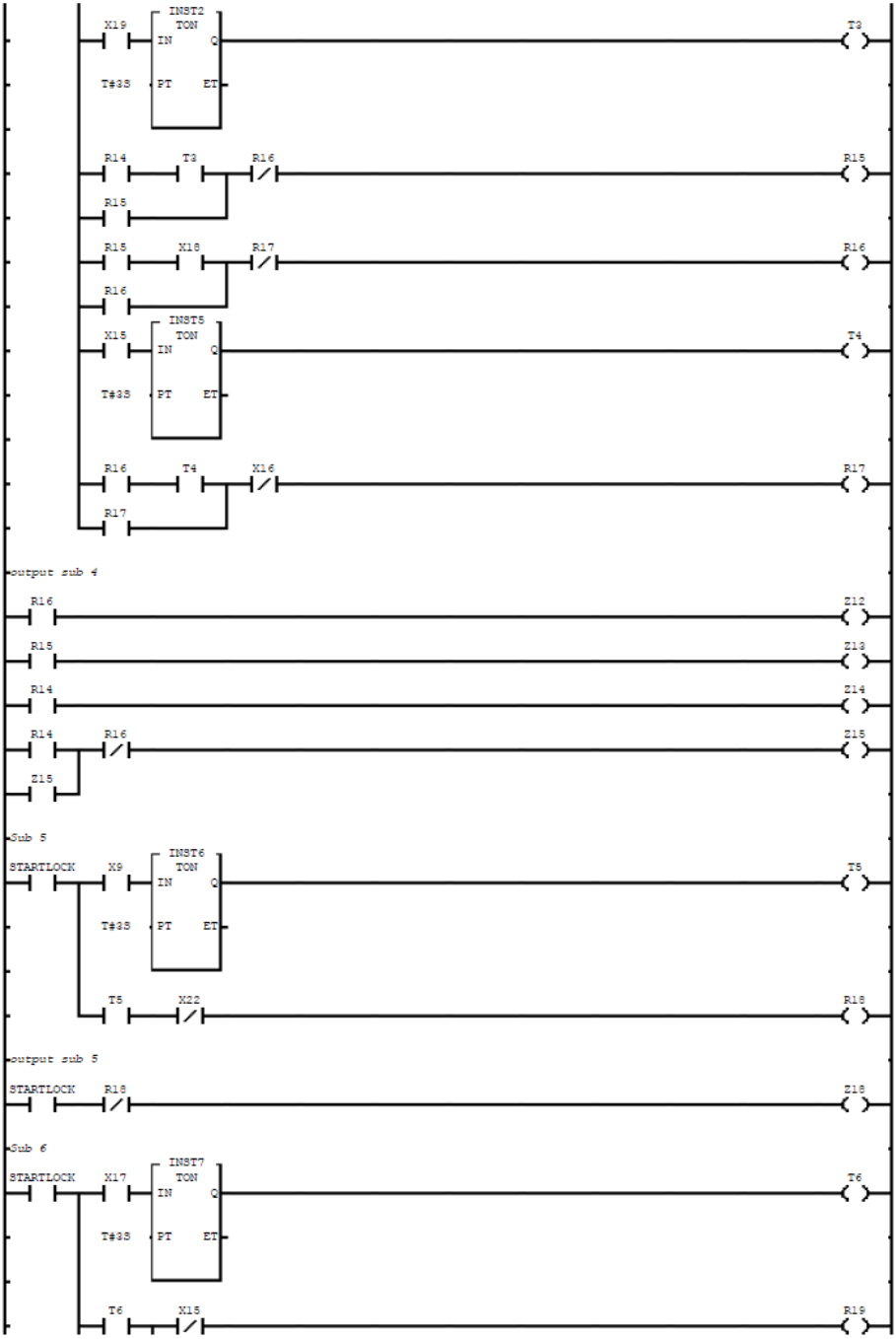

== Timer Actuator ==
STARTLOCK
L1
CTEA
PA
TA
TA
L2
CTEB
PB
TB
TB
L3
CTEC
PC
TC
TC
L4
CTED
PD
TD
TD
L5
CTEE
PE
TE
TE
L6
CTEF
PF
TF
TF
== Timer for Line Movement Blue Process ==
TA
TB
TC
TD
TE
TF
INST18
TON
IN
Q
T#1.05S
PT
ET
X33

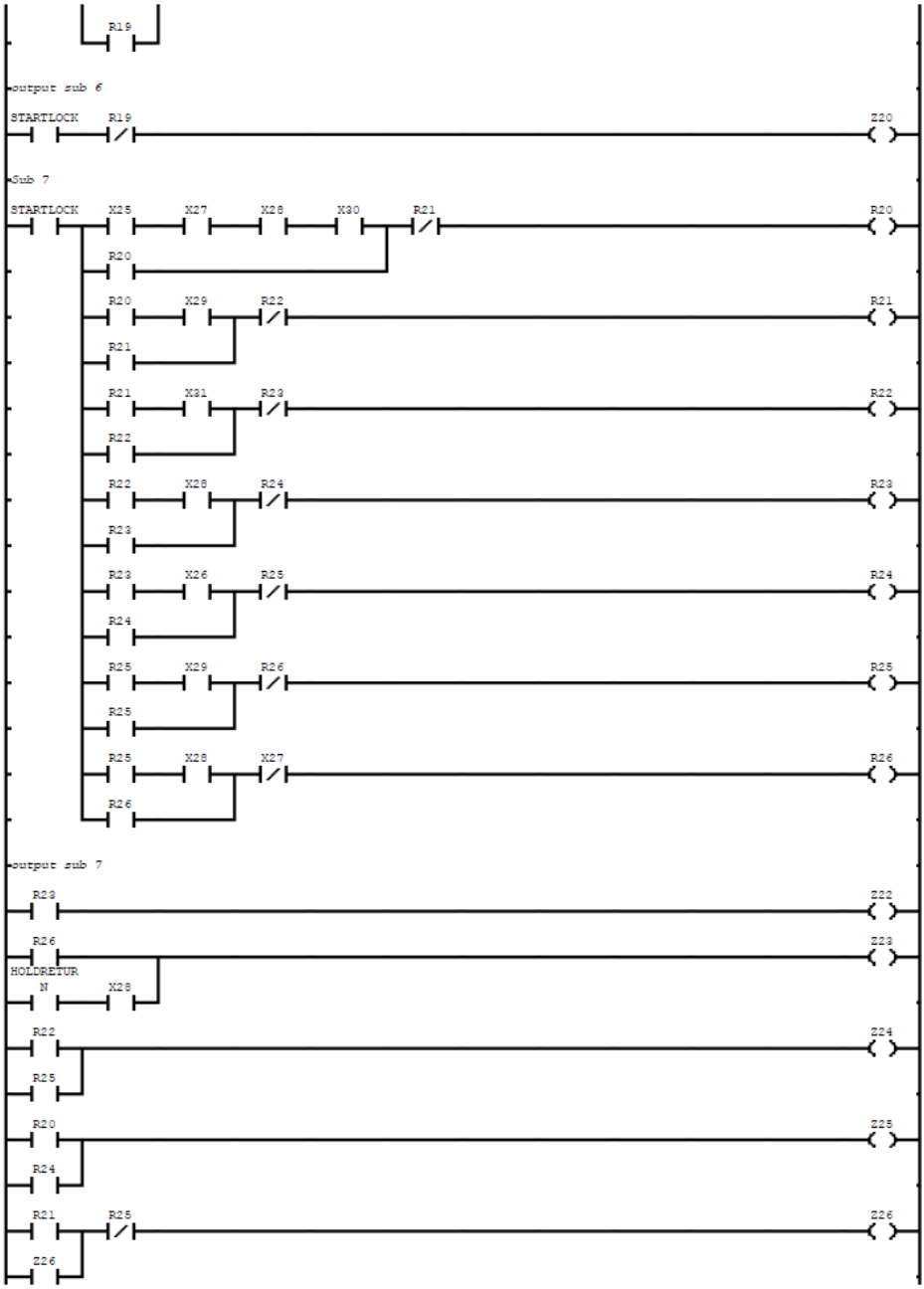

== Relay Process Interlock ==
CTBA
CTBB
CTBC
CTBD
CTBE
CTBF
INST19
TON
IN
Q
T#0.5S
PT
ET
N
== Relay Process ==
STARTLOCK
CTBA
N
RP2
RP7
RP1
RP1
CTBB
N
RP3
RP7
RP2
RP2
CTBC
N
RP4
RP7
RP3
RP3
CTBD
N
RP5
RP7
RP4
RP4
CTBE
N
RP6
RP7
RP5
RP5
CTBF
N
RP7
RP7
RP6
RP6
RP6
X04
RP7

Ladder diagram konvensional




STARTLOCK
X9
X11
X12
R7
R8
R7
R7
R7
X13
R9
R8
R8
R8
X14
R10
R9
R9
R9
X12
R11
R10
R10
R10
X10
R12
R11
R11
R11
X13
R13
R12
R12
R12
X12
X11
R13
output sub 3
R10
Z7
R13
Z8
HOLDRETUR
N
X12
R9
Z9
R12
HOLDRETUR
N
R11
Z10
R7
R8
R12
Z11
Z11
Sub 4
STARTLOCK
X16
X17
X18
R15
R14
R14


X19
INST2
TON
IN
Q
T#3S
PT
ET
T3
R14
T3
R16
R15
R15
R15
X18
R17
R16
R16
INST5
TON
X15
IN
Q
T#3S
PT
ET
T4
R16
T4
X16
R17
R17
output sub 4
R16
Z12
R15
Z13
R14
Z14
R14
R16
Z15
Z15
Sub 5
STARTLOCK
X9
INST6
TON
IN
Q
T#3S
PT
ET
T5
T5
X22
R18
output sub 5
STARTLOCK
R18
Z18
Sub 6
STARTLOCK
X17
INST7
TON
IN
Q
T#3S
PT
ET
T6
T6
X15
R19


R19
output sub 6
STARTLOCK
R19
Z20
Sub 7
STARTLOCK
X25
X27
X28
X30
R21
R20
R20
R20
X29
R22
R21
R21
R21
X31
R23
R22
R22
R22
X28
R24
R23
R23
R23
X26
R25
R24
R24
R25
X29
R26
R25
R25
R25
X28
X27
R26
R26
output sub 7
R23
Z22
R26
Z23
HOLDRETUR
N
X26
R22
Z24
R25
R20
Z25
R24
R21
R25
Z26
Z26

RIWAYAT HIDUP

Andhiko Palito F adalah seorang anak pertama dari Ayah Ahmad Fauzi dan Ibu Eni tana yang lahir dan tinggal di Bukittinggi sejak 22 Januari 1995 hingga menamatkan sekolah menengah atas pada umur 19 tahun. Setelah SMA merantau ke Jakarta untuk melanjutkan studi program beasiswa Astra di Politeknik Manufaktur Astra dengan bidang studi Mekatronika dan selesai pada tahun 2016. Selama menempuh studi D3 juga mengikuti kegiatan magang industri di PT. Aisin Indonesia di bagian Equipment Development selama 6 bulan. Selama kegiatan magang industri berhasil menyelesaikan 2 mesin pendukung lini produksi dan melakukan analisa kesalahan (troubleshooting) 2 buah mesin. Melanjutkan studi kembali di Institut Teknologi Sepuluh Nopember, dengan jurusan Teknik Eletro, bidang studi Teknik Sistem Pengaturan. Selama studi S1 juga mengikuti program magang industri di PT. Semen Padang selama 2 bulan, dan selama magang juga ikut membantu analisa kesalahan program sebuah mesin.

Kontak
Ponsel : +62 823 923 10727
Email : andhikopalitof@gmail.com