Tugas Supervisory Control Pengendali Pintu Otomatis dengan Simulasi Hardware in The Loop

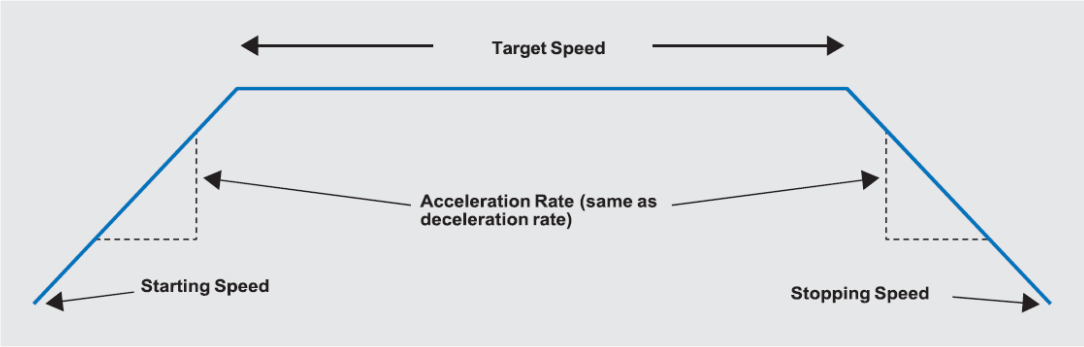
Contents

Daftar Gambar

1 Spesifikasi

Spesifikasi tugas ini adalah seperti pada poin-poin berikut.

- Time constant motor 1,2 detik
- Frekuensi sampling 100 Hz (perioda sampling 10 ms)
- Terdapat dua tombol yaitu tombol buka dan tombol tutup
- Di pintu ada sensor untuk mendeteksi orang / barang yang menghalangi, sinyal dari sensor dianggap sama dengan sinyal dari tombol buka
- Profil kecepatan pintu pada waktu membuka adalah seperti pada grafik berikut.



Gambar 1 Profil Kecepatan Pintu

Untuk mengerjakan tugas ini, mahasiswa telah membuat asumsi-asumsi seperti pada poin-poin berikut:

- Pintu merupakan sebuah pintu geser dengan lebar 1 meter
- Motor yang digunakan untuk membuka pintu dilengkapi dengan driver yang sedemikian rupa sehingga masukkan 1 volt akan menghasilkan rotasi 1 revolusi/detik
- Motor terhubung ke pintu sedemikian rupa sehingga satu rotasi motor akan menggeserkan pintu sejauh 1 meter
- Kecepatan pintu negatif menyatakan bahwa pintu sedang menutup
- Kecepatan pintu positif menyatakan bahwa pintu sedang membuka
- Posisi pintu pada nilai 1 meter menyatakan bahwa pintu sedang dalam keadaan terbuka

Perilaku dari pengendali pintu yang diinginkan adalah seperti pada poin-poin berikut.

- Jika pintu tertutup, pintu akan tetap tertutup
- Jika pintu tertutup dan tombol buka ditekan, pintu akan mulai membuka
- Jika pintu terbuka, pintu akan tetap terbuka selama 3 detik sebelum mulai menutup
- Jika pintu terbuka dan tombol tutup ditekan, pintu akan mulai menutup
- Jika pintu sedang menutup dan tombol buka ditekan, maka pintu akan mulai membuka
- Akselerasi dan deselerasi pintu terjadi selama 0,5 s

2 Perhitungan Transformasi Bilinear

Motor penggerak pintu dapat dimodelkan sebagai sistem orde 1 yang memiliki fungsi transfer seperti pada persamaan berikut.

$$H(s) = \frac{K}{T_c s + 1}$$

Dengan T_C adalah konstanta waktu motor dan K adalah *gain* dari motor. Diskritisasi fungsi transfer dapat dilakukan dengan metode transformasi bilinear yang dilakukan dengan substitusi berikut.

$$s = \frac{2}{T_s}\frac{z-1}{z+1}$$

Dengan T_s adalah perioda pencuplikan yang digunakan. Setelah melakukan substitusi, didapat versi diskrit fungsi transfer seperti pada persamaan berikut.

$$H(z) = \frac{KT_s(z+1)}{(T_s + 2T_c)z + (T_s - 2T_c)}$$

$$H(z) = \frac{KT_s(1+z^{-1})}{(T_s + 2T_c) + (T_s - 2T_c)z^{-1}}$$

$$\frac{Y(z)}{X(z)} = \frac{KT_s(1+z^{-1})}{(T_s + 2T_c) + (T_s - 2T_c)z^{-1}}$$

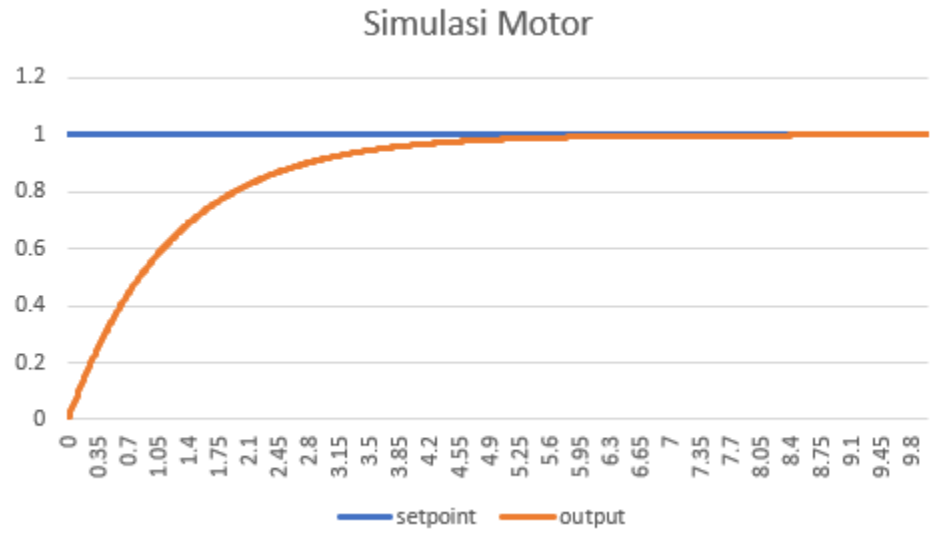
Dengan Y(z) adalah fungsi keluaran motor dan X(z) adalah fungsi masukkan motor dalam domain z. Menggunakan transformasi z invers, didapatkan bahwa motor dapat disimulasikan secara *real-time* dengan persamaan berikut.

$$y(n) = \frac{KT_s\big(x(n) + x(n-1)\big) - (T_s - 2T_c)y(n-1)}{T_s + 2T_c}$$

Dengan y(n) adalah fungsi keluaran motor dan x(n) adalah fungsi masukkan motor dalam domain waktu.

3 Simulasi Menentukan Time Constant

Menggunakan persamaan yang didapat pada bab sebelumnya, sebuah simulasi dapat dibuat untuk memastikan kesesuaian model motor dalam domain diskrit. Simulasi motor dilakukan menggunakan sebuah program C yang memberikan keluaran file csv. File csv yang dihasilkan memiliki empat kolom yaitu *time, setpoint, output,* dan *position* yang dapat diplot menggunakan *software* seperti MS Excel. Plot simulasi motor adalah seperti pada gambar berikut.



Gambar 2 Simulasi Respon Step Motor

Nilai-nilai yang digunakan adalah seperti pada poin-poin berikut.

- T_s = 0,01s
- T_c = 1,2s
- K = 1

Berdasarkan data pada file csv, dapat diamati bahwa nilai output yang paling mendekati nilai 63,2% dari nilai maksimal output berada pada sampel ke 121 atau pada *time* = 1,2 s. Dengan itu, dapat disimpulkan bahwa model motor diskrit sudah sesuai dengan spesifikasi yang dibuat.

4 Simulasi Kendali Kecepatan Pintu di Desktop

Karena pintu diinginkan dapat mencapai kecepatan maksimal dan minimal dari 0 dalam waktu 0,5 s, maka perlu dibuat sebuah pengendali yang dibuat sedemikian rupa sehingga respon step motor memiliki *settling time* sebesar 0,5 s dan nilai *steady state error* sebesar 0. Peletakan pole yang dapat menghasilkan s*ettling time* sebesar 0,5 s dapat dicari menggunakan persamaan berikut.

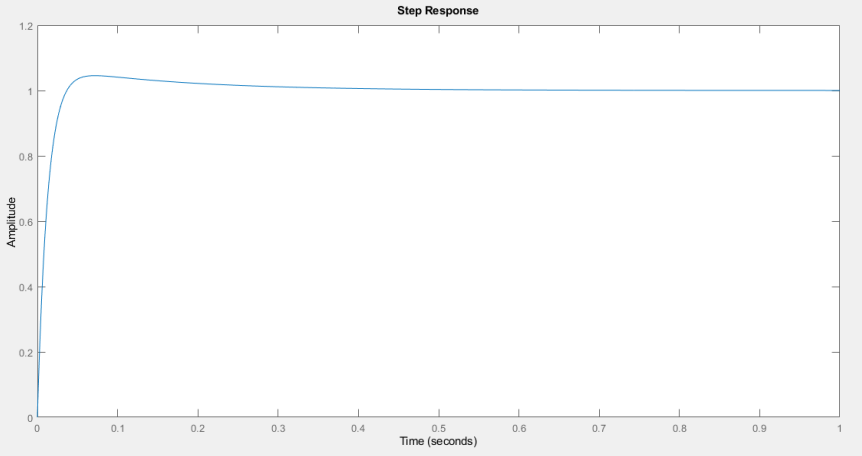
$$T_c = \frac{-0{,}5}{ln(0{,}05)}$$

$$pole = \frac{ln(0{,}05)}{0{,}5} = -5{,}991$$

Peletakan pole tersebut dapat dicapai dengan cara membuat pengendali PI dengan pole di origin dan zero di -6. Gain pengendali PI kemudian dapat diatur sehingga didapatkan pole dominan di sekitar -6. Melalui trial and error, mahasiswa mendapatkan bahwa pengendali PI dengan persamaan di bawah dapat memberikan respon yang diinginkan.

$$C(s)\frac{100(s+6)}{s}$$

Fungsi transfer pengendali menyiratkan bahwa pengendali PI memiliki K_p sebesar 100 dan K_i sebesar 600. Pengujian respon step sistem di MATLAB adalah seperti pada gambar berikut.

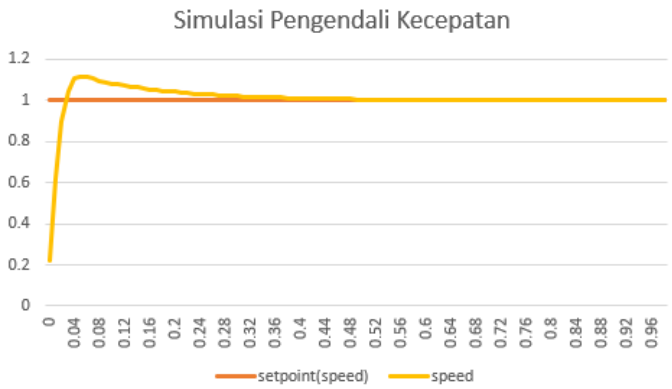


Gambar 3 Simulasi Respon Step Motor dengan Pengendali PI

Diskritisasi pengendali PI dilakukan dengan pendekatan integral trapezoid sehingga pengendali PI memiliki persamaan berikut.

$$c(n) = K_p e(n) + \frac{K_i T_s\big(e(n) + e(n-1)\big)}{2}$$

Dengan c(n) adalah sinyal kendali dan e(n) adalah sinyal galat dalam domain waktu. Simulasi pengendali PI dilakukan menggunakan program C. Sama seperti sebelumnya, program tersebut akan menghasilkan file csv. File csv tersebut memiliki lima kolom yaitu *time, speed setpoint, acceleration setpoint, speed,* dan *accelleration*. Hasil plotting data pada file csv adalah seperti pada gambar berikut.



Gambar 4 Simulasi Respon Step Kecepatan Motor dengan Pengendali PI Diskrit

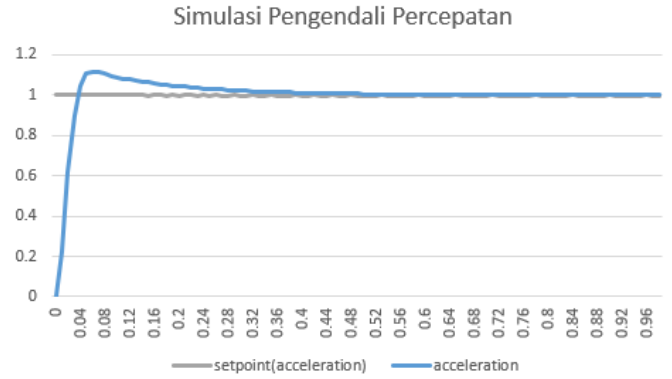
Nilai-nilai yang digunakan adalah seperti pada poin-poin berikut.

- $K_p = 50$
- $K_i = 300$

Nilai koefisien proporsional dan integral yang digunakan pada pengendali PI diskrit berbeda dengan pengendali PI kontinu. Hal tersebut disebabkan karena mahasiswa menemukan bahwa peneggunakan nilai $K_p = 100$ dan $K_i = 600$ akan mengashilkan respon dengan overshoot yang terlalu besar (sekitar 40%). Berdasarkan plot respon sistem, dapat disimpulkan bahwa pengendali PI telah memberikan respon kecepatan yang dibutuhkan.

5 Simulasi Kendali Percepatan Pintu di Desktop

Simulai pengendalian percepatan dilakukan dengan menggunakan program C yang sama dengan simulasi pengendali kecepatan. Untuk melakukan simulasi pengendalian percepatan, nilai setpoint yang diberikan ke sistem diinkremenkan pada setiap sampel untuk mensimulasikan percepatan yang kosntan. Dengan itu, hasil plot data simulasi adalah seperti pada gambar berikut.

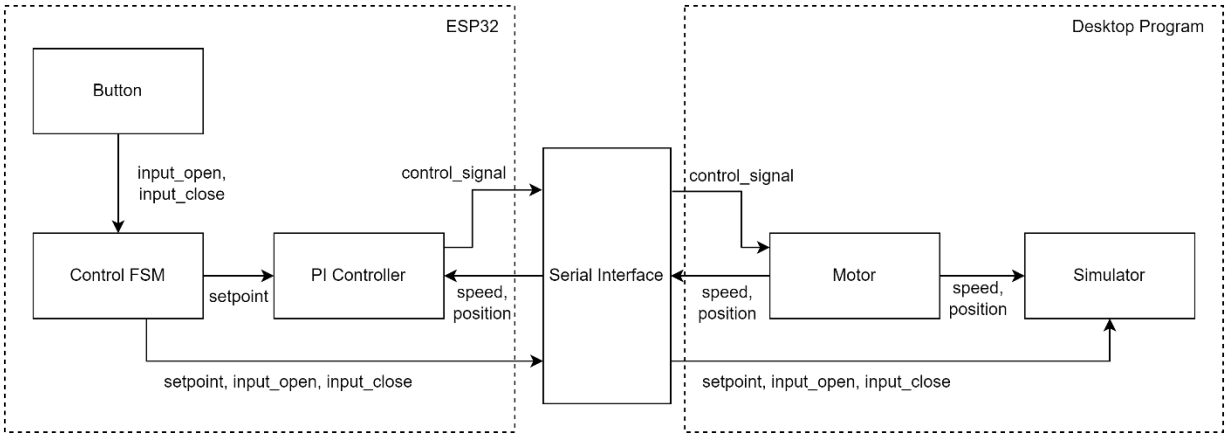


Gambar 5 Simulasi Respon Step Percepatan Motor dengan Pengendali PI Diskrit

Berdasarkan bentuk respon tersebut, dapat disimpulkan bahwa pengendali PI dapat mengendalikan percepatan dengan baik.

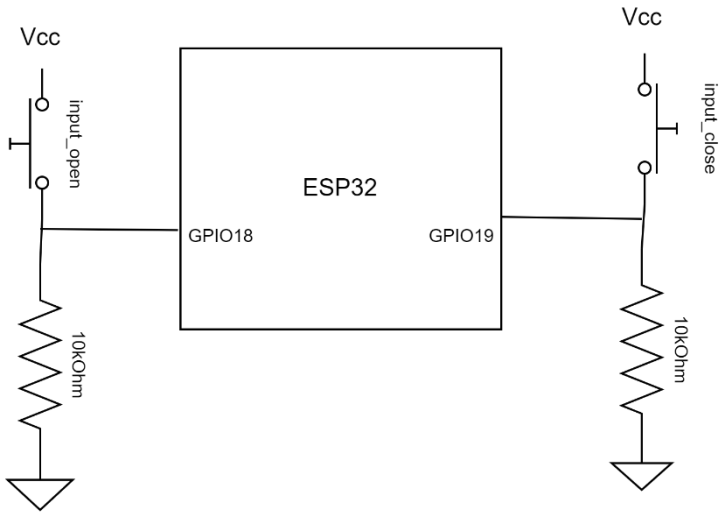
6 Simulasi HIL dengan Desktop dan Mikrokontroler

Simulasi HIL dibuat menggunakan mikrokontroler ESP32 dan program python yang berjalan di komputer desktop. Mikrokontroler berperan sebagai pengendali motor pintu sendangkan komputer desktop berperan sebagai motor dan pintu. Diagram blok simulator HIL adalah seperti pada gambar berikut.



Gambar 6 Diagram Blok Simulasi HIL

Rangkaian pada ESP32 adalah seperti pada gambar berikut.



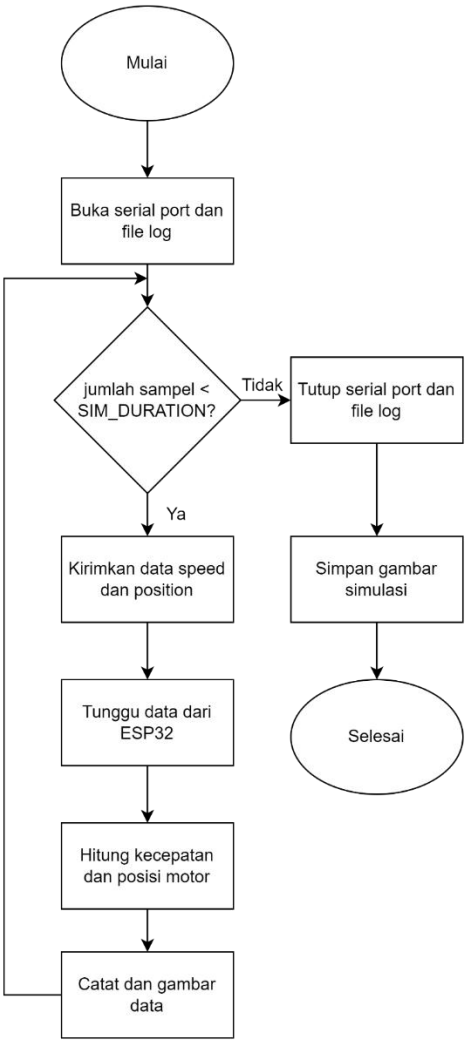
Gambar 7 Rangkaian Tombol di ESP32

Seluruh *source code* untuk tugas ini dapat dilihat di https://github.com/ubbeg2000/tugas2-el4121.

6.1 Desain Software di Desktop

Program komputer dibuat menggunakan python3. Blok Motor diimplementasikan dengan cara yang sama seperti pada bab 3 namun dengan bahasa pemrograman yang berbeda. Blok simulator berfungsi untuk menyimpan data simulasi ke dalam sebuah file csv dan menggambar data simulasi secara *real-time*. Selain

itu, simulator juga berfungsi untuk mengelola komunikasi serial diantara ESP32 dan komputer. Diagram alir simulator adalah seperti pada gambar berikut.



Gambar 8 Diagram Alir Blok Simulator

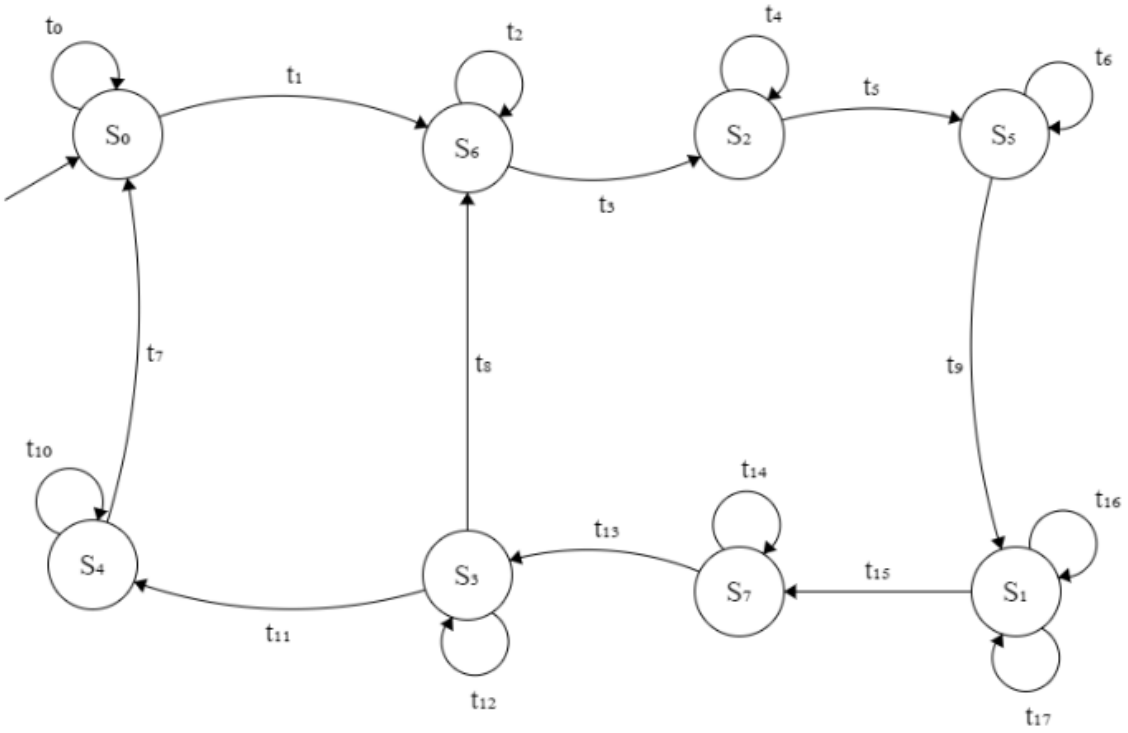
Terdapat beberapa pengaturan simulator yang dapat diubah, pengaturan-pengaturan tersebut adalah seperti pada poin-poin berikut.

- SIM_WINDOW_WIDTH : jumlah sampel yang ditampilkan oleh *display* simulasi
- SIM_DURATION : jumlah sampel yang disimpan dalam simulasi
- SIM_LOG_FILE_NAME : nama *default* dari file log simulasi
- SIM_PORT : port serial *default*
- SIM_REFRESH_RATE : jumlah sampel yang dibutuhkan untuk memperbarui *display* simulasi

Blok simulator diimplementasikan di file hil.py sedangkan blok motor diimplementasikan di file motor.py. Program simulator dapat dijalankan dengan menjalankan hil.py. Simuasi HIL dapat dihentikan dengan cara menekan Ctrl+C di komputer atau dengan cara menekan tombol EN di ESP32

6.2 Desain Software di Mikrokontroler

Perangkat lunak di ESP32 terdiri dari dua modul yaitu Control FSM dan PI Controller. PI Controller merupakan implementasi pengendali PI diskritp ada bab 4 dan bab 5. Blok FSM merupakan sebuah FSM yang berfungsi untuk menghasilkan setpoint yang sesuai dengan perilaku pengendali yang dijabarkan pada bab 1. Control FSM dibuat untuk menerima masukkan dari tombol-tombol yang terhubung ke ESP32, kecepatan motor, serta poisi pintu. Diagram FSM adalah seperti pada gambar berikut.



Gambar 9 Diagram State Control FSM

Keterangan State:

- S_0 : DOOR_CLOSE, merepresentasikan keadaan pintu tertutup
- S_1 : DOOR_OPEN, merepresentasikan keadaan pintu terbuka
- S_2 : DOOR_OPENING, merepresentasikan keadaan pintu membuka dengan kecepatan konstan
- S_3 : DOOR_CLOSING, merepresentasikan pintu menutup dengan kecepatan konstan
- S_4 : ACC_UP_0, merepresentasikan percepatan pintu ke kecepatan 0
- S_5 : ACC_DOWN_0, merepresentasikan perlambatan pintu ke kecepatan 0
- S_6 : ACC_UP_MAX, merepresentasikan percepatan pintu ke kecepatan maksimal
- S_7 : ACC_DOWN_MIN, merepresentasikan perlambatan pintu ke kecepatan minimal

Keterangan Input:

- input_open : sinyal murni yang merepresentasikan masukkan tombol buka
- input_close : sinyal murni yang merepresentasikan masukkan tombol tutup
- speed : sinyal bilangan riil yang merepresentasikan kecepatan pintu
- position : sinyal bilangan riil yang merepresentasikan posisi pintu

Keterangan Variabel:

- cnt : variabel yang digunakan untuk menahan pintu dalam keadaan terbuka

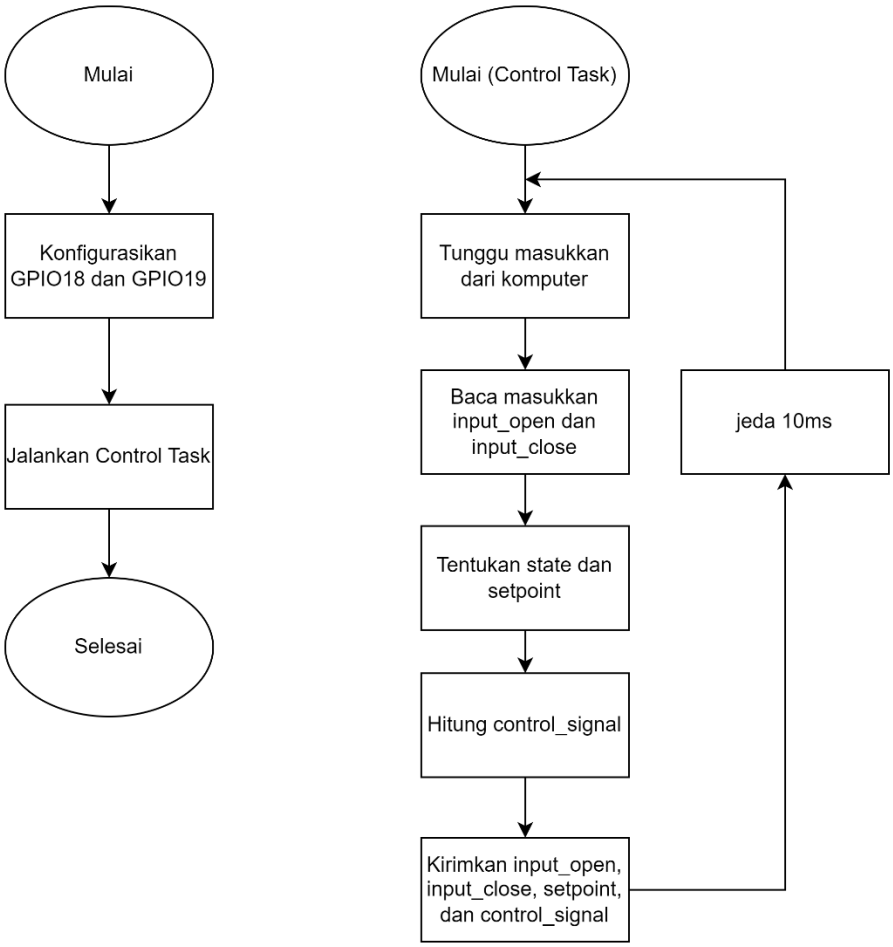
Keterangan Output:

- output : bilangan riil yang merepresentasikan setpoint untuk pengendali

Keterangan Transisi State:

- t_0 : input_open & ~input_close / output = 0
- t_1 : input_close & ~input_open / output = 0
- t_2 : speed >= MIN_SPEED / output -= ACC_RATE
- t_3 : speed < MIN_SPEED / output = MIN_SPEED
- t_4 : position < OPEN_TH
- t_5 : position >= OPEN_TH
- t_6 : speed > 0 / output -= ACC_RATE
- t_7 : speed >= 0 / output += ACC_RATE
- t_8 : input_open = 1
- t_9 : speed <= 0 / cnt = OPEN_DURATION, output = 0
- t_{10} : speed >= 0 / output += ACC_RATE
- t_{11} : ~input_open & position <= CLOSE_TH / output = MIN_SPEED
- t_{12} : ~input_open & position> CLOSE_TH / output = MIN_SPEED
- t_{13} : speed < MIN_SPEED / output = MIN_SPEED
- t_{14} : speed >= MIN_SPEED / output -= ACC_RATE
- t_{15} : input_close | (~input_open & cnt < 0) / output = 0
- t_{16} : ~input_close & cnt >= 0 / cnt -= 1, output = 0
- t_{17}: ~input_close & input_open / cnt = OPEN_DURATION, output = 0

Blok Control FSM merupakan implementasi dari FSM pada gambar di atas. Diagram alir perangkat lunak pada mikrokontroler adalah seperti pada gambar berikut.

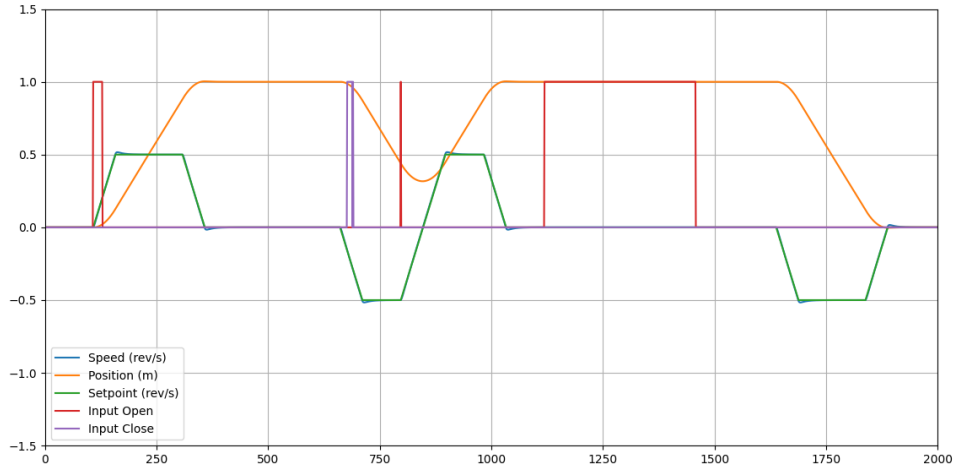


Gambar 10 Diagram Alir Program ESP32

Perangkat lunak pada mikrokontroler dibuat menggunakan bahasa C. Pemrograman ke mikrokontroler ESP32 dilakukan dengan menggunakan ESP-IDF.

6.3 Hasil Simulasi HIL

Salah satu hasil simulasi HL adalah seperti pada gambar berikut.



Gambar 11 Hasil Simulasi HIL

Dapat diamati bahwa hasil simulasi adalah sesuai dengan perilaku-perilaku pengendali yang diinginkan. Dapat diamati bahwa pintu akan membuka ketika input_open ditekan dan akan menutup ketika input_close ditekan. Dapat diamati pula bahwa percepatan dan perlambatan pintu memakan waktu kurang lebih 0,5 detik (sekitar 50 sampel). Selain itu ,dapat diamati bahwa pintu akan berada dalam keadaan terbuka selama 3 detik sebelum menutup kembali. Perilaku pembatalan penutupan pintu dengan cara menekan tombol input_open dapat diamati pula pada grafik tersebut.

7 Kesimpulan

Untuk menyimpulkan:

- FSM dapat digunakan untuk menerapkan *supervisory control*
- Pengendali PI berhasil melakukan pengendalian kecepatan motor
- Pengendali PI berhasil melakukan pengendalian percepatan motor
- Hasil simulasi HIl menunjukkan bahwa pengendali mampu mengendalikan pintu sehingga didapatkan perilaku yang diinginkan.