LAPORAN PERKULIAHAN
TEKNOLOGI KENDALI PROSES

CHAIDIR ANWAR_D411 12 012

TEKNIK KOMPUTER KENDALI ELEKTRONIKA (TKKE)

FAKULTAS TEKNIK JURUSAN ELEKTRO

UNHAS

BAB I
PENDAHULUAN

1.1. LATAR BELAKANG

Dalam rangka memenuhi tuntutan zaman sebagai mahasiswa teknik elektro, mahasiswa kini perlu memiliki kemampuan untuk dapat menerapkan ilmu yang dipelajarinya kedalam bidang kerjanya masing-masing. Dengan berbagai perkembangan teknologi, tentu perlu pembelajaran yang lebih banyak. Bagi seorang mahasiswa teknik elektro dengan konsentrasi Teknik Kendali, perlulah mendapat pembelajaran lebih mengenai cara mengendalikan dan merancang sebuah sistem fisik kedalam bentuk matematis yang dapat dianalisis, dalam hal kaitannya dengan suatu sistem fisik yang melakukan suatu proses, baik berupa proses fisika maupun kimia. Untuk itu penting bagi kita dalam memahami cara-cara yang diperlukan untuk membuat perancangan suatu sistem fisik kedalam model yang mudah untuk dianalisis –dalam hal ini adalah kestabilannya dan respon keluarannya–.

1.2. TUJUAN

Mampu mengubah suatu model sistem fisik kedalam model yang dapat dianalisis, baik matematis maupun dalam bagan kotak, sehingga dapat dianalisis kestabilan dan respon keluarannya. Dalam hal ini akan dilakukan dengan bantuan software *Simulink MATLAB R2013a (8.1.0.604).*

BAB II
PROYEK GUNT

Gunt merupakan sebuah perusahaan di Jerman yang membuat miniatur dari berbagai proses. Salah satunya adalah *level control, hsi*. Berikut adalah gambaran dari sistem fisik tersebut:

Level control merupakan sebuah proses yang berusaha menjaga kestabilan level air pada tangki penampungannya. Miniatur ini memiliki sebuah pompan untuk memompa air dari bawah ke dalam tangki penampungannya, sebuah katup proporsional untuk membuang kelebihan air dari yang diinginkan, serta sebuah sensor tekanan yang dapat mengukur ketinggian air di dalam tangki penampungan. Sistem ini juga dilengkapi dengan sebuah pipa pembuangan ketika terjadi kelebihan tampungan pada tangki. Berikut ini adalah data-data teknis yang dibutuhkan dalam perancangan model proses:

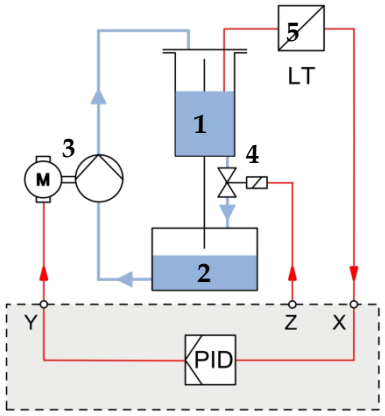
1. Level-controlled tank
- capacity: 1,2L

2. Storage tank
- capacity: 3,7L

3. Pump
- power consumption: 18W
- max. flow rate: 8L/min
- max. head: 6m

4. Proportional valve: Kvs: 0,7m³/h

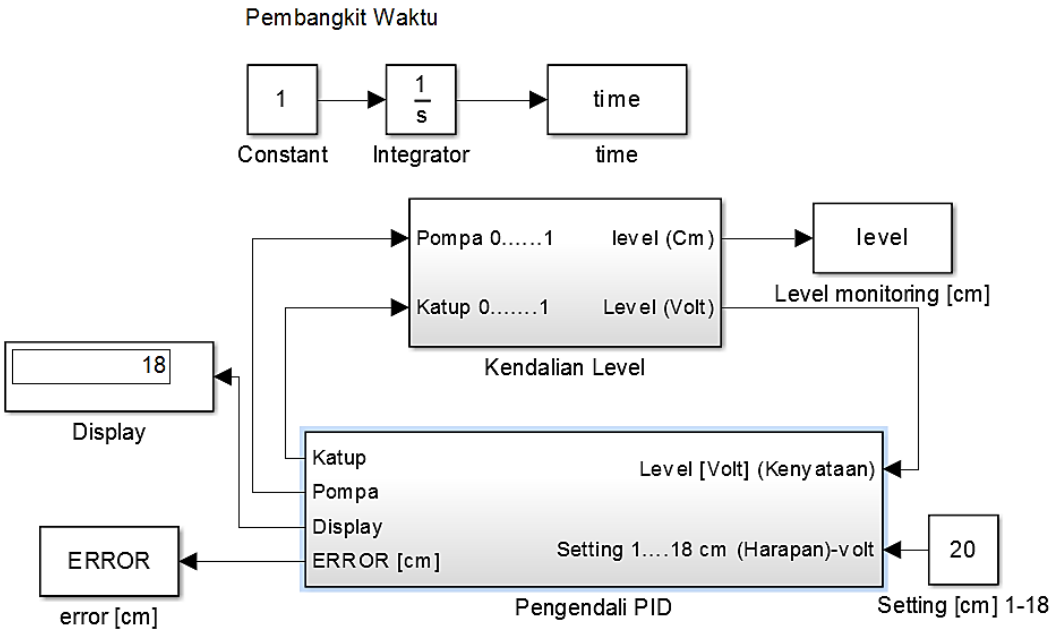
5. Pressure sensor: 0...30mbar (0...300mm)



Bagan skematik proses dari miniatur diperlihatkan pada gambar disamping. Sebagai informasi tambahan yang tidak diketahui, diasumsikan bahwa tinggi tangki adalah 20 cm dengan tinggi pipa pembuangan *overflow* adalah 18 cm.

Dari skematik proses diatas, dapat dilihat bahwa ada dua blok utama, yakni pengendali dan kendalian. Pengendali akan mendapat isyarat *setpoint*, yang berupa nilai pengaturan ketinggian air yang diinginkan. Pada pengendali, dibutuhkan dua buah masukan sebagai parameter acuannya, yaitu *setpoint* dan level ketinggian air yang telah dicapai, yang diambil dari pembacaan sensor yang sudah dalam satuan volt. Sebagai keluaran dari pengendali adalah isyarat kendalian untuk katup dan pompa. Keluaran lain dari pegendali disini sebagai tambahan adalah *display* yang akan menunjukkan hasil pembacaan dari *sensor* (lebih tepatnya dikatakan sebagai *tranducer*) dalam satuan *centimeter*. Keluaran lainnya adalah galat (*error*) yang disimpan pada sebuah variabel "*ERROR"* (*disebut to workspace)* yang akan menyimpan kondisi galat setiap waktu selama selang waktu simulasi sehingga respon dari sistem dapat teramati. Untuk kemudahan, selanjutnya untuk dilakukan *monitoring* akan digunakan bentuk variabel seperti ini.

Sedangkan untuk kendalian, akan menerima masukan berupa isyarat kendalian dari pengendali, yaitu isyarat kendalian pompa katup. Satu-satunya keluaran dari kendalian adalah hasil dari pembacaan sensor ketinggian dalam satuan *volt*. Namun sebagai tambahan, dipasang sebuah *variabel "level"* –yang dimaksud adalah *to workspace*, bertipe *array*– untuk level ketinggian air pada tangki. Untuk kemudahan *monitoring* terhadap waktu, dibuatlah pembangkit waktu dengan persamaan $\int \mathbf{1}\,\boldsymbol{dt} = \boldsymbol{t}$ untuk membantu respon sistem terhadap waktu (dengan nama variabel "*time").* Berikut ini adalah model umum yang telah kita capai hingga saat ini, dengan Kendalian Level dan Pengendali PID berupa *s*.



1. KENDALIAN LEVEL

Seperti yang telah dijelaskan sebelumnya, Kendalian atau *Plant* memiliki dua masukan, yaitu isyarat kendalian untuk pompa serta katup, serta dua buah keluaran, yakni umpan balik atau *feedback* dari sensor serta untuk *monitoring* ketinggian air pada tanki.

1.1. POMPA

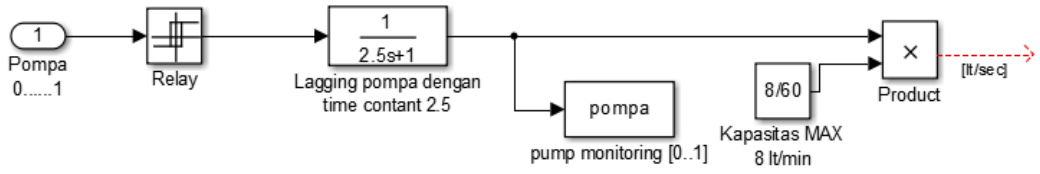
Pompa memiliki isyarat kendalian yang diasumsikan dari nol sampai satu [0..1], dimana nol adalah kondisi saat pompa dimatikan, sedangkan satu adalah saat pompa dinyalakan dengan kecepatan maksimum. Diketauhi bahwa pompa yang digunakan memiliki kecepatan aliran maksimum 8 liter/menit. Untuk itu dibuat sebuah blok *constant* yang digunakan sebagai pembanding terhadap sinyal kendalian. Sebagai catatan, pompa juga dapat diberi isayarat kendalian kurang dari 0 karna sifat *hysteresis* dari pompa yang harus diberi isyarat kendalian kurang dari 0 untuk mematikannya.

Karena pompa tentunya diputar oleh motor yang dikontrol oleh sebuah *relay*, maka pada pompa akan berlaku suatu model *hysteresis*, yakni kondisi dimana pompa tidak memiliki titik yang sama untuk on atau off dari keadaan sebelumnya. Untuk itu, dari input pompa dipasang sebuah blok *relay* yang memberikan efek *hysteresis* tersebut. Sebagai parameter pada blok relay, diberikan titik pergantian *on* pada 0.2, dan titik pergantian *off* pada -0.2, dengan keluaran adalah 1 saat *on* dan 0 saat *off*.

Setelah *on*, motor pompa tidak akan serta merta berputar dengan kecepatan maksimum sehingga aliran pompa maksimum tidak mungkin akan langsung tercapai. Untuk motor orde 1, motor pompa akan *lagging* terhadap respon masukannya dengan *time constant (τ)* tertentu. Untuk itu, digunakan sebuah blok *transfer function* untuk membuat *lagging* dengan *time constant* τ dengan persamaan $\frac{1}{\tau \cdot s+1}$.

Keluaran dari blok *tranfer function* ini dapat dipasang variabel *monitoring "pompa"* untuk mengetahui respon dari kecepatan motor pompa. Hasil dari *lagging* berupa kecepatan motor terhadap putaran maksimumnya[0..1], sehingga berdasarkan putaran motor maksimum, kita dapat membandingkan dengan kecepatan aliran maksimum –blok *constant* yang dibahas sebelumnya– untuk mendapatkan kecepatan aliran air sesaat yang dipompa ke dalam tangki.

Untuk membandingkan, kita dapat menggunakan sebuah blok *product* dengan keluaran berupa kecepatan aliran yang dipompa kedalam tangki dalam satuan *liter/detik.* Jika diasumsikan τ=2.5, maka secara keseluruhan blok diagram pompa dapat digambarkan sebagai berikut:



1.2. KATUP

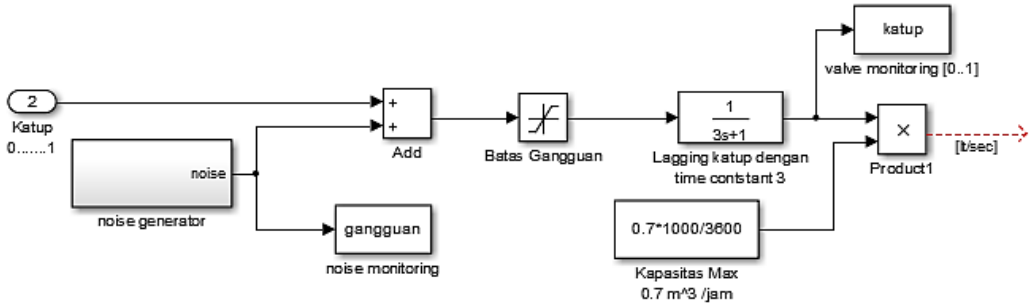
Katup yang digunakan adalah katup proporsional, artinya tidak hanya tertutup atau terbuka secara penuh, tapi memiliki *range*, seperti setengah tertutup, atau seperempat terbuka. Untuk itu, maka isyarat kendalian dapat berupa nilai-nilai dari nol sampai satu, dengan asumsi bahwa 0 adalah saat pompa tertutup penuh dan 1 saat pompa terbuka penuh. Sebagai catatan, katup tidak akan diberikan keadaan isyarat kendalian kurang dari 0 atau lebih dari 1.

Katup akan beroprasi dengan rentang isyarat kendalian dari 0 sampai 1 . Oleh sebab itu, isyarat kendalian diberi blok *saturation,* sehingga memberi batasan apabila ada gangguan yang muncul dan menyebabkan isyarat kendalian yang diterimanya bernilai tidak pada rentang tersebut.

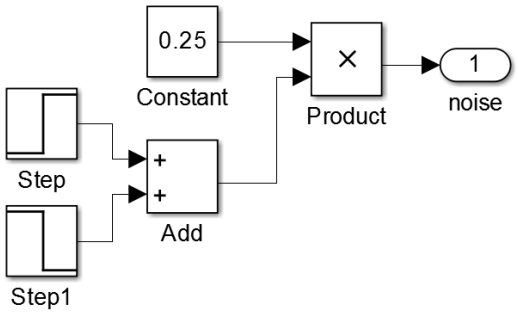
Seperti pada motor, katup juga tidak dapat langsung menutup dan membuka sekat berdasarkan isyarat kendaliannya, melainkan akan mengalami *lagging*. Oleh sebab itu, setelah blok *saturation* kembali dipasangkan sebuah blok *transfer function* yang memberikan sifat *lagging* tersebut pada katup. Setelah blok ini akan diberikan sebuah variabel *monitoring* dengan nama "*katup*" untuk memantau status dari katup.

Karena katup dapat membuang fluida dengan debit maksimum 0.7 m^3/hours, dan isyarat kendalian yang telah keluar dari blok *lagging* akan menghasilkan isyarat dari 0 sampai 1, maka kita dapat kembali membandingkan besarnya aliran yang keluar dari katup sesaat terhadap debit maksimumnya dengan kembali menggunakan blok *product*. Tentu saja dengan mengkonversi satuan dari debit maksimum dalam *liter/sec* terlebih dahulu. Keluarannya berupa debit aliran air yang dibuang dari tangki melalui katup dalam satuan *liter/second.*

Sebagai tambahan, sebelum melewati blok *saturation,* isyarat kendalian dapat diberikan gangguan untuk melihat respon sistem apabila terdapat gangguan. Akan ada sebuah yang dibuat dapat menghasilkan gangguan pada waktu tertentu sesuai keinginan, yang kemudian akan ditambahkan dengan blok *add* sehingga menghasilkan suatu isyarat kendalian yang terganggu. penghasil gangguan dapat dipantau dengan memasang variabel *monitoring "gangguan"*. Sehingga apabila keseluruhan sistem digambarkan, dengan *time constant* diasumsikan bernilai 3 detik, akan membentuk susunan blok diagram sebagai berikut:



Untuk *noise generator,* yakni penghasil gangguan, misalnya dapat dibuat dengan blok diagram di bawah ini. Tujuannya adalah menghasilkan gangguan berbentuk pulsa yang dapat diaktifkan selama waktu tertentu, misalnya dari 100 sampai 150 detik dengan amplitudo 0.25.

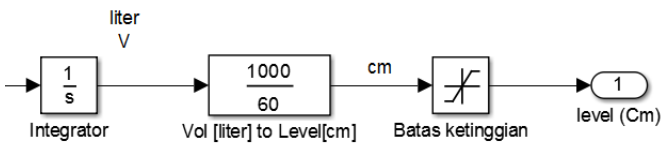


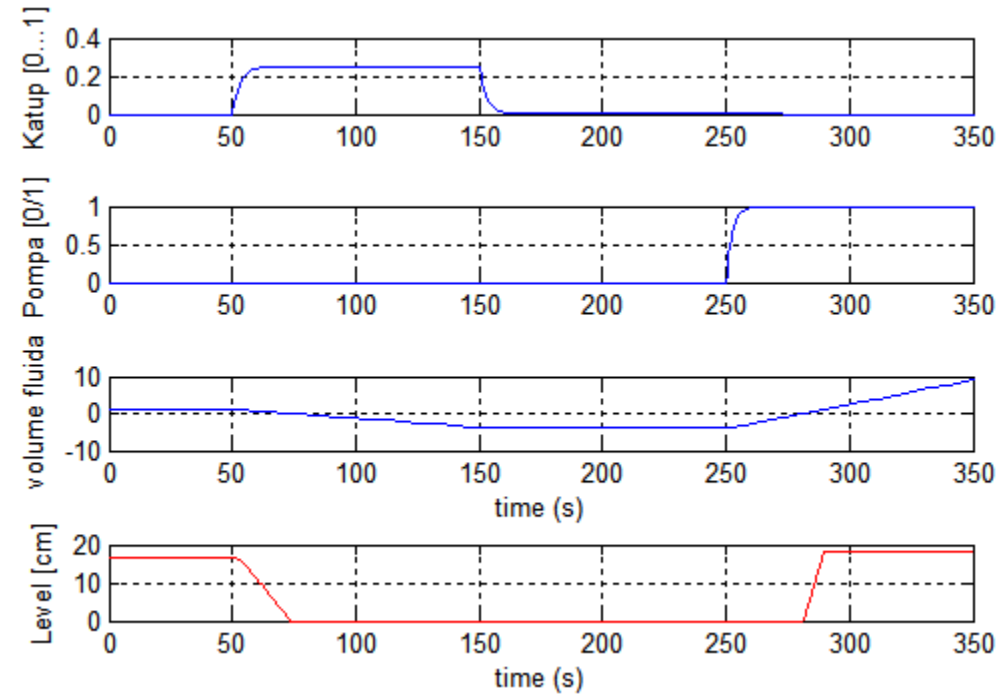
1.3. LEVEL CONTROLLED TANK

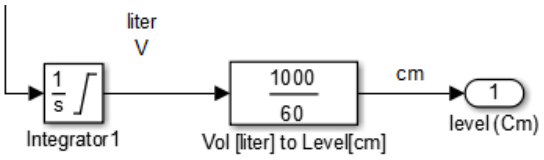
Level controlled tank merupakan tangki penampungan yang akan diatur ketinggiannya. Tangki ini akan mendapat tambahan fluida dari pompa, dan kelebihan fluida dapat dibuang melalui katup. Karena pompa bersifat menaikkan level fluida pada tangki, dan katup bersifat mengurangi aliran level fluida, maka aliran sesaat pada tangki adalah kecepatan aliran pada pompa dikurangi kecepatan aliran pada katup. Untuk itu dipasanglah suatu blok yang mengurangkan aliran pompa terhahap aliran katup tersebut, yakni blok *sum.*

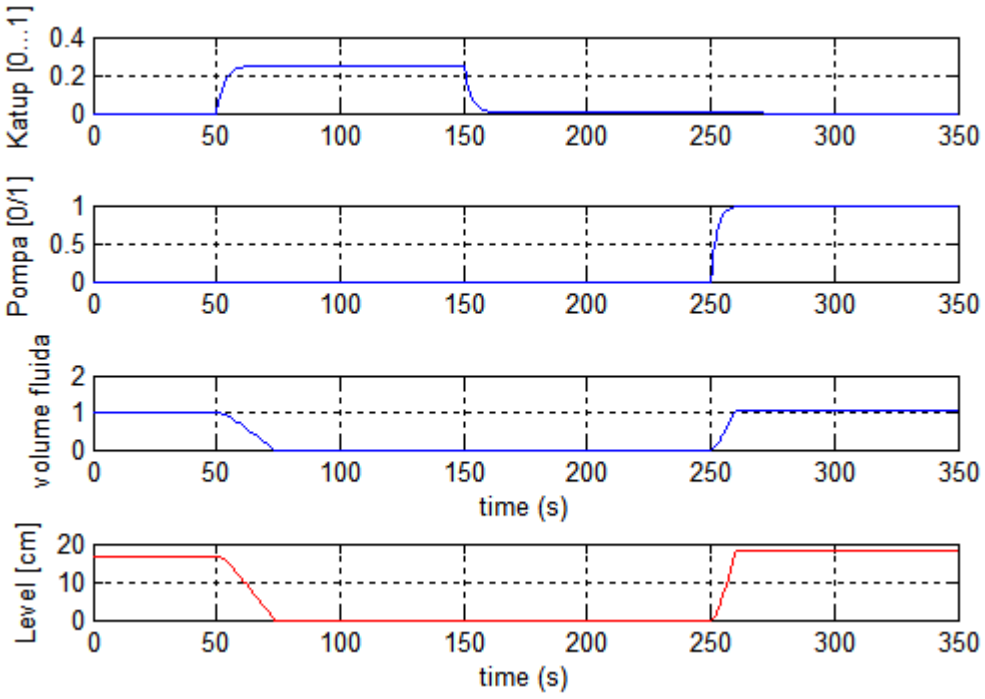
Hasil dari penjumlahan ini memiliki satuan *liter/detik* –turunan volume terhadap waktu–. Sehingga untuk mendapatkan volume sesaat pada tangki maka dilakukan pengintegralan sehingga menghasilkan satuan volume. Dengan asumsi luas penampang tangki adalah 60 cm^2, dengan tinggi pipa pembuangan *overflow* 18 cm –18.1 cm untuk toleransi 0.1cm– maka batas dari volume ini adalah dari 0 sampai 1.086 liter. Volume merupakan integrasi dari kecepatan aliran fluida setiap waktu. Untuk itu perlulah menggunakan suatu blok *integrator* yang dilengkapi *saturation* agar volume yang ada di dalam tangki tidak bernilai di luar rentang ini. Mengapa tidak menggunakan blok *saturasi* saja setelah diubah menjadi satuan cm agar lebih mudah perhitungannya? Sebab, apabila dilakukan setelah blok *integrator*, maka pada kondisi katup dibuka maksimum tanpa tambahan fluida dari pompa, hasil *integrator* tetap akan bernilai negatif walaupun pada *monitoring* nilai ini tetap terbaca sebagai level 0cm akibat pemasangan blok *saturation* setelah *integrator*.Namun, begitu pompa dinyalakan, level air tidak akan langsung naik, karena pompa perlu kembali menolkan volume fluida yang bernilai negatif tersebut. Itulah mengapa pembatasan volume fluida dilakukan langsung pada blok *integrator*, sehingga apapun yang terjadi, volume fluida akan bernilai kurang dari nol. Mari membandingkan grafik respon sistem terhadap perbedaan peletakan *saturation.*

Pada blok diagram yang menggunakan *saturation* setelah blok *integrator*, volume akan tetap bernilai negatif, dan level air akan mulai bertambah saat volume ini telah melewati titik nol. Berbeda dengan blok *integrator* yang telah dilengkapi dengan batas (saturasi) untuk volumenya. Saat pompa dinyalakan, maka saat itu juga level ketinggian fluida di dalam tangki akan naik. Secara fisik, model yang kedua inilah yang masuk akal. Blok *integrator with saturation* akan diberikan batas bawah 0 dan batas atas 1.086. Pada blok ini juga dapat diisi *initial condition*, yakni kondisi awal dari tangki dalam satuan liter (misalnya 1).







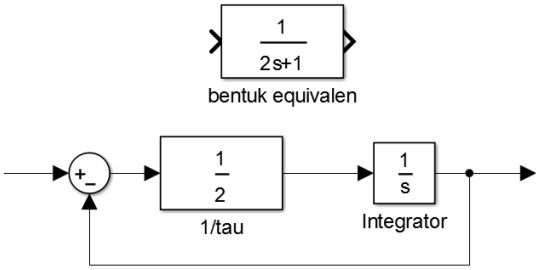


Karena keluaran yang akan di*monitoring* dari ini adalah level ketinggian tangki dalam satuan cm, perlu dilakukan perubahan dari volume menjadi ketinggian. Ketinggian dalam cm adalah volume dalam cm^3 dibagi luas penampangnya dalam cm^2. Untuk itu digunakan sebuah blok *transfer function* yang mengkonversi nilai ini sebagai keluaran. Sebuah *sensor* akan dipasang pula pada keluaran ini untuk mengawasi level ketinggian fluida dalam cm sehingga menjadi

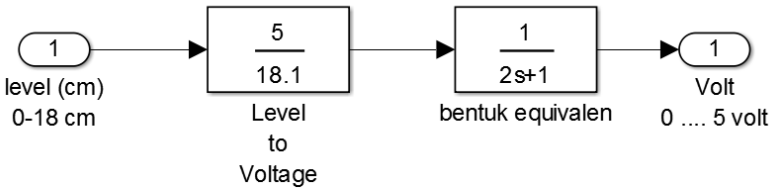
level ketinggian dalam *volt*.

1.4. SENSOR

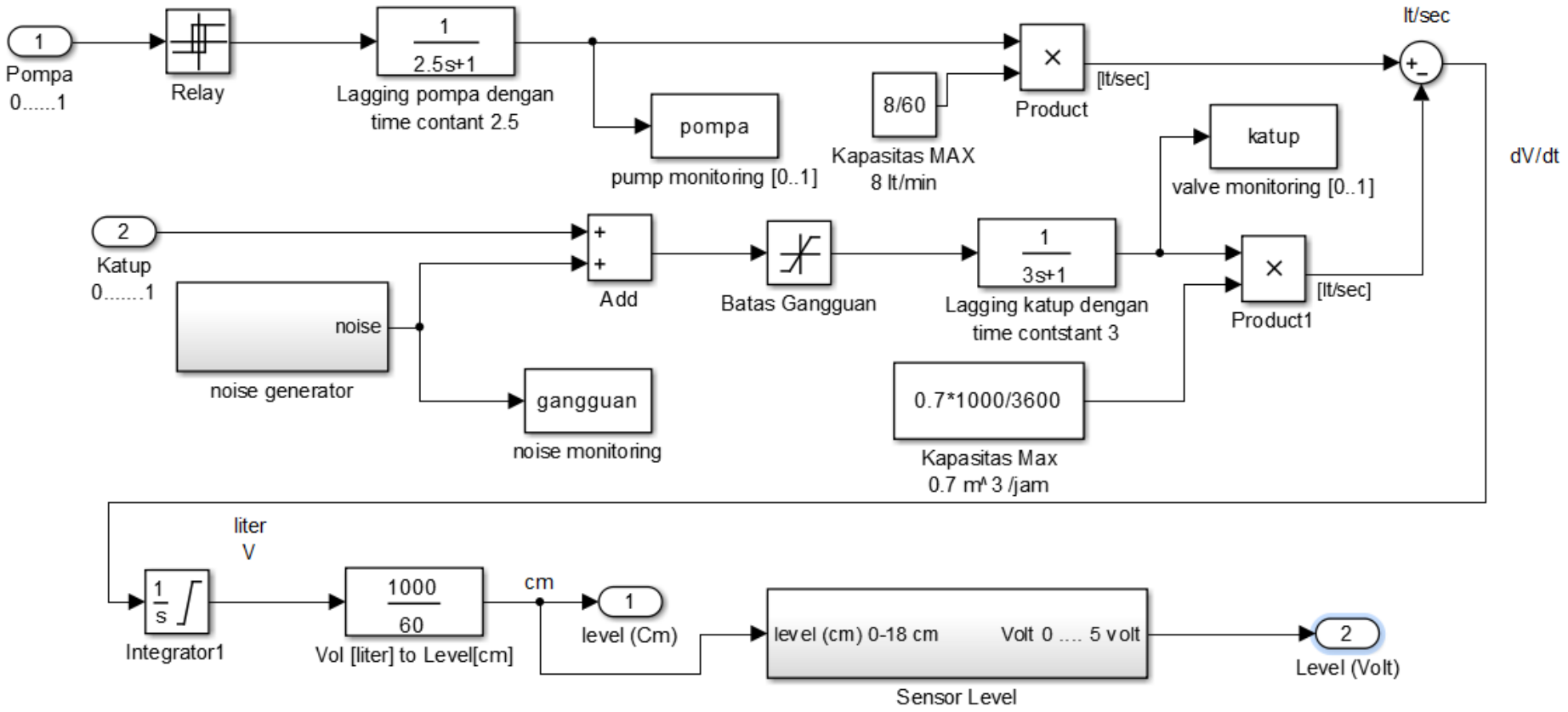
sensor bertujuan untuk membaca ketinggian air dalam cm, dan memberikan keluaran dalam satuan *volt*. Untuk ketinggian maksimum adalah 18.1 cm, dan akan dikonversi dalam satuan *volt* dari 0 sampai 5, maka dari satuan cm dapat diubah ke *volt* dengan mengalikannya dengan sebuah blok *transfer function* $\frac{5}{18.1}$.



Namun, seperti sensor pada umumnya, tentu pembacaan akan mengalami *lagging*. Misalnya saja dengan *time constant* 2, maka blok *lagging* dapat berbentuk seperti sebelumnya (gambar atas), atau dengan penjabaran (gambar bawah). Kelebihan dari gambar di bawah, kita dapat menentukan nilai awal pada blok *integrator*. Yang perlu diketauhi adalah penambahan nilai awal pada sensor akan merusak hasil pembacaan sensor, dan dapat menyebabkan masalah kemudian. Misalnys saja tidak akan pernah di dapatkan ketiggian fluida yang tepat jika dilihat dari pembacaan sensornya, sehingga output akan terus berosilasi atau ketinggian air tidak akan sesuai dengan *setpoint* yang diberikan. Untuk itu, pemberian nilai awal pada sensor tidak perlulah diberikan, karena sensor akan membaca nilai awal dari ketinggian fluida dengan sendirinya, walaupun tanpa memberikan nilai awal yang sama pada sensor. Secara lengkap, blok diagram sensor ditunjukkan pada gambar berikut:



Secara lengkap, blok kendalian level diberikan pada gambar di bawah ini:



2. PENGENDALI PID

Pengendali PID adalah yang memberikan isyarat kendalian pada kendalian agar level ketinggian fluida pada tangki tetap stabil sesuai dengan nilai *setpoint* yang diberikan. Masukan dari pengendali ada dua, yakni *setpoint* atau harapan dan hasil pembacaan sensor dalam *volt,* atau kenyataan. Sedangkan galat atau *error* adalah selisih antara harapan dan kenyataan.

Yang pertama adalah *display* yang menunjukkan hasil pembacaan sensor dalam satuan *centimeter*. Karena rentang keluaran sensor adalah 0 sampai 5 volt untuk pembacaan 0 sampai 18.1 cm, perlu diberikan blok pembalik dari volt ke cm, yakni *transfer fuction* dengan nilai $\frac{\mathbf{18.1}}{\mathbf{5}}$, sehingga dari blok ini akan langsung didapatkan level ketinggian fluida dalam satuan cm.

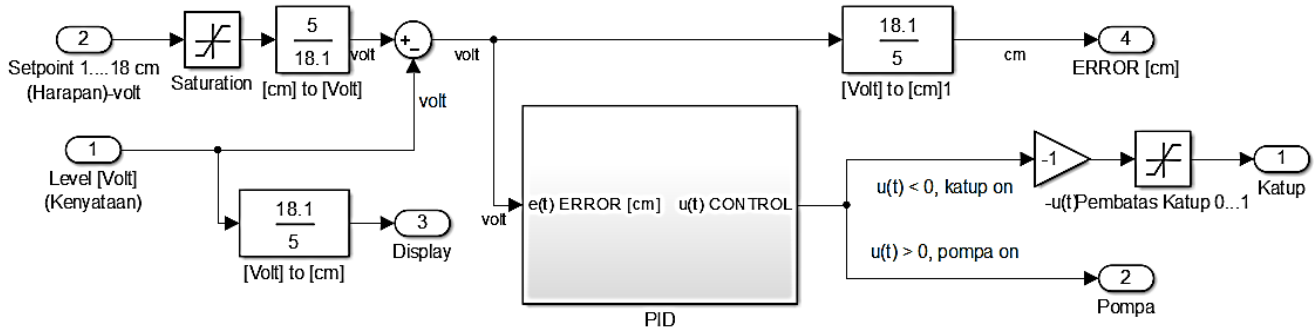
Karena hasil pembacaan sensor adalah dalam *volt*, maka untuk mencari galat, perlu dilakukan konversi *setpoint* terlebih menjadi satuan yang sama, *volt.* Namun, sebelum dilakukan konversi, perlu dilakukan pembatasan pada *setpoint,* agar nantinya *setpoint* berlebih, misalnya lebih besar dari tinggi pipa pembuangan *overflow*, maka akan mengakibatkan pompa terus *on* karena tinggi fluida pada tangki tidak akan mungkin melebihi tinggi pipa pembuangan *overflow* tersebut. Untuk itu, dari masukan *setpoint* dipasang sebuah blok *saturation* yang nilainya dibuat dari 0 sampai 18.1 cm. Barulah setelah itu sebuah blok *transfer function cm-to-volt* $\left(\frac{\mathbf{5}}{\mathbf{18.1}}\right)$ diletakkan.

Sekarang, keduanya –harapan dan kenyataan– telah dalam satuan yang sama. Barulah *error(volt)* bisa didapatkan dengan mengurangkan antara harapan dan kenyataan. Jika hasil bernilai negatif, artinya ketinggian fluida melebihi harapan sehingga katup perlu dibuka, dan apabila bernilai positif, artinya ketinggian fluida kurang dari harapan, sehingga pompa yang perlu di jalankan. Untuk memantau *error* dalam *centimeter* sebagai salah satu keluaran dari pengendali, kembali setelah *error* dalam *volt* didapatkan, diletakkan sebuah blok *transfer fuction volt-to-cm* $\left(\frac{\mathbf{18.1}}{\mathbf{5}}\right)$ yang keluarannya sudah merupakan *error* dalam satuan *centimeter.*

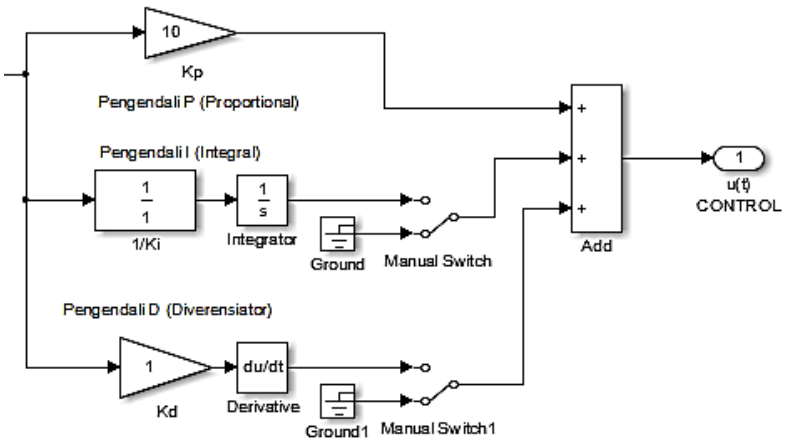
Pengendalian pompa dan katup dilakukan dengan menambahkan sebuah *PID* yang berisi pengendali PID berdasarkan error yang diterimanya. Oleh sebab itu, keluaran dari pengendali apabila bernilai positif akan dibuat untuk menggerakkan pompa dan apabila bernilai negatif akan dibuat untuk menggerakkan katup. Untuk itu dari keluaran *PID* langsung disambungkan ke keluaran yang menuju pompa, sehingga keluaran PID tersebut akan langsung mengendalikan pompa. Sedangkan untuk katup saat keluaran PID bernilai negatif, perlu dilakukan pembalikan dan pembatasan terlebih dahulu. Dilakukan pembalikan karena yang mengaktifkan katup adalah keluaran negatif dari PID, sedangkan katup akan menerima isyarat kendalian dari 0 sampai 1 (positif). Untuk itulah keluaran PID dilakukan pembalikan terlebih dahulu. Sedangkan dilakukan pembatasan dari 0 sampai 1 dilakukan karena agar isyarat kendalian tidak bernilai negatif ataupun bernilai diatas 1. Namun, untuk membuat katup tidak terlalu rentan terhadap gangguan, batasan atas yang dimasukkan bukanlah 1, melainkan 5-10. Batasan atas juga tidak boleh terlalu tinggi, sebab efek *lagging* pada katup dapat menyimpan isyarat yang terlalu besar sehingga dapat menyebabkan sistem malah menjadi lebih tidak stabil. Pembatasan diatas 1 ini tidaklah menjadi masalah, sebab pada kendalian sendiri sudah diberikan batasan sehingga isyarat yang diterimanya tidak melewati rentang 0..1.

Lalu, mengapa pompa tidak diberikan pembatasan yang sama? Pompa memiliki sifat *hysteresis,* sehingga butuh nilai isyarat kendalian negatif yang cukup untuk mematikan pompa. Jika diberikan batasan yang sama pada pompa, isyarat kendalian pompa tidak akan pernah bernilai negatif, sehingga pompa akan terus-menerus *on.* Inilah sebab mengapa pompa tidak diberikan batasan isyarat kendalian.

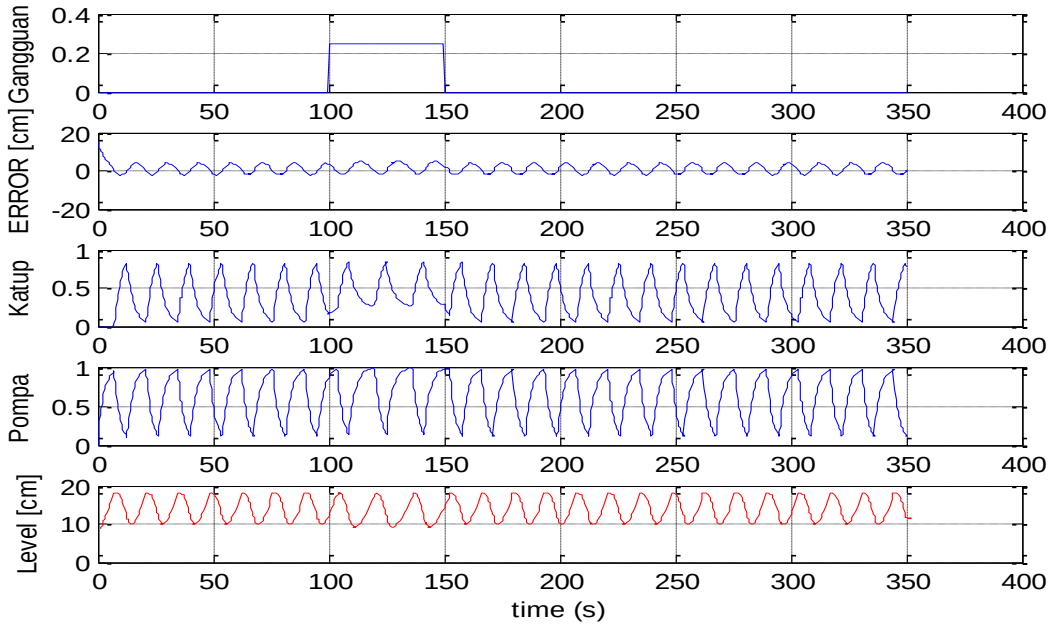
Berikut ini adalah blok diagram dari *PID* secara lengkap:



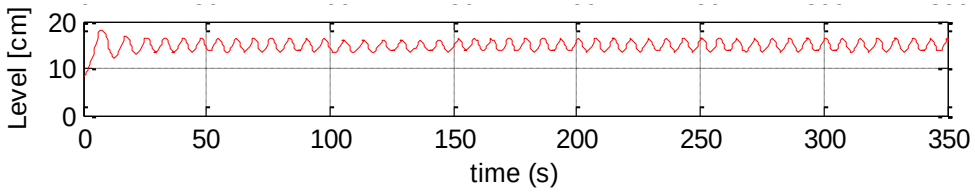
Selanjutnya adalah *PID.* Pendengali PID terdiri atas tiga buah pengendali; pengendali proportional (P), pengendali Differensial (D) dan pengendali Integral (I). Pengendali proportional bersifat memperbesar error sehingga akan lebih cepat dilakukan koreksi. Pengendali differensial bersifat meredam osilasi pada respon sistem. Sedangkan pengendali integral bersifat mempercepat respon sistem mencapai keadaan stabil, namun bersifat memberikan efek osilasi pada respon sistem tersebut. Berikut ini adalah blok diagram pengendali PID secara umum:



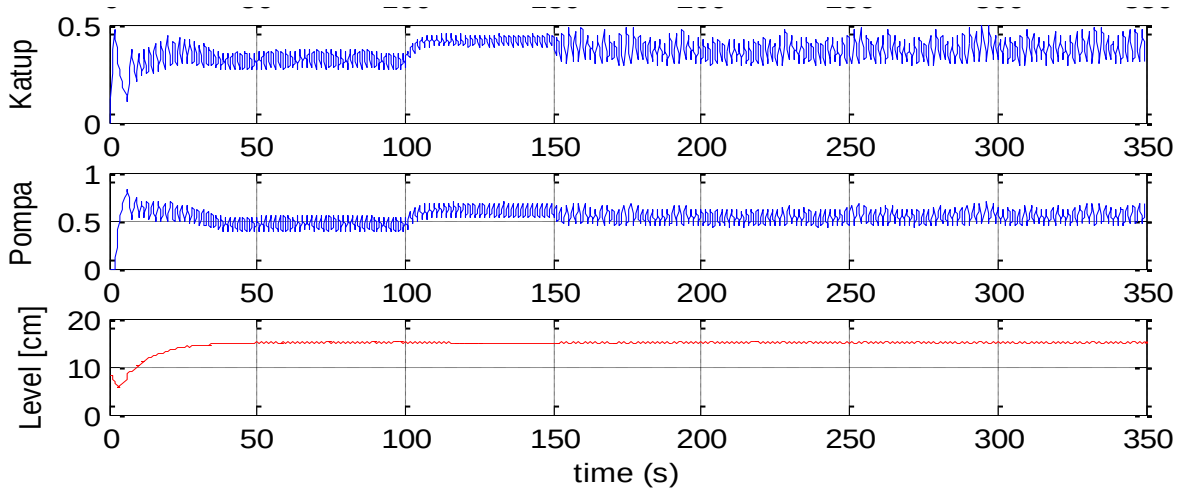
Terlihat bahwa keluaran pengendali PID u(t) merupakan akumulasi dari pengendali P, I dan D. Grakfik berikut akan menunjukkan bagaimana respon sistem (ketinggian fluida) dengan hanya menggunakan pengendali proportional dengan Kp=10 terhadap *setpoint* 15 cm (dengan volume awal 0.5):



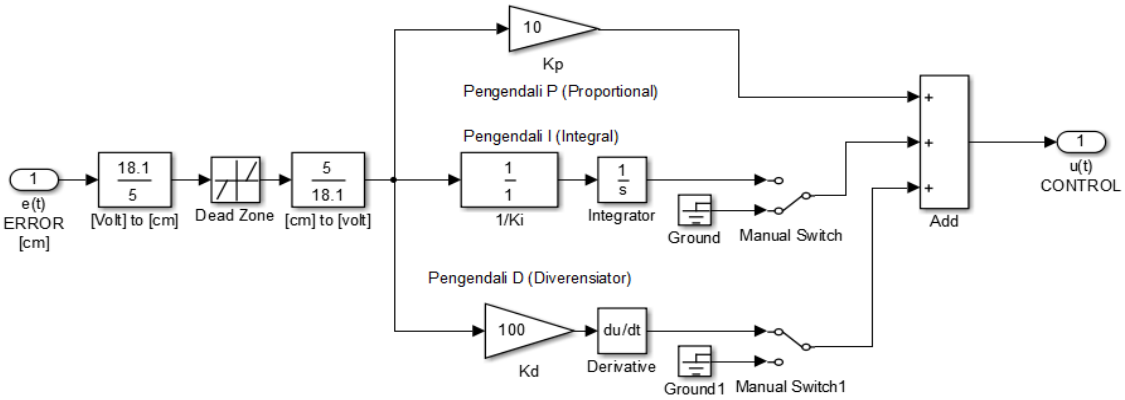
Dari respon sistem terlihat bahwa level ketinggian fluida hanya berosilasi di sekitar *setpoint.* Untuk meredam osilasi maka dipasanglah kombinasi pengendali PD (Kd=10), sehingga respon sistem menjadi seperti grafik di bawah ini:

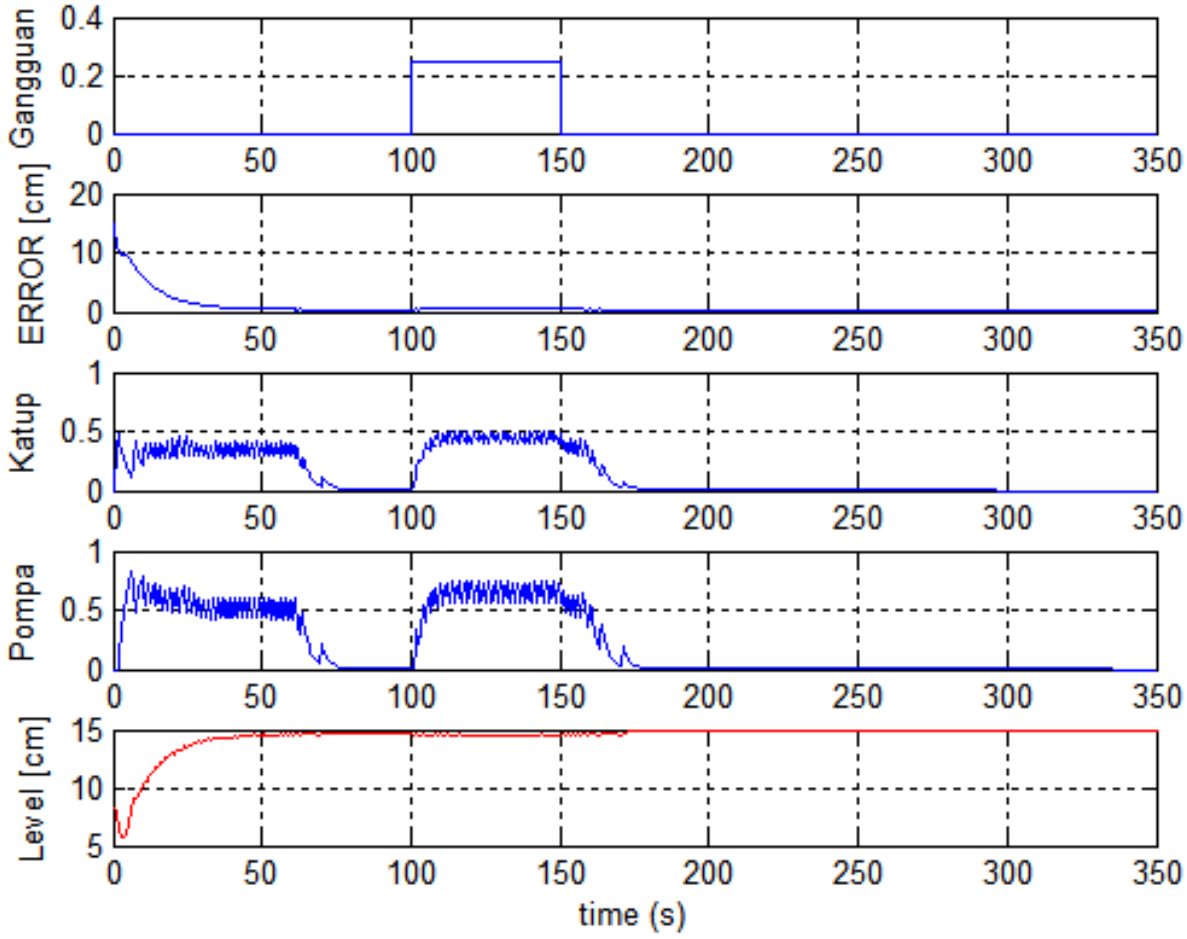


Jelas terlihat bahwa osilasi yang dihasilkan sudah cukup teredam. Untuk lebih meredam respon level ketinggian fluida pada tangki, kita hanya perlu menaikkan nilai koefisien differensial (Kd) menjadi cukup besar untuk meredam osilasi yang terjadi, misalnya Kd=50-100. Berikut adalah contoh respon sistem dengan nilai Kd=100:



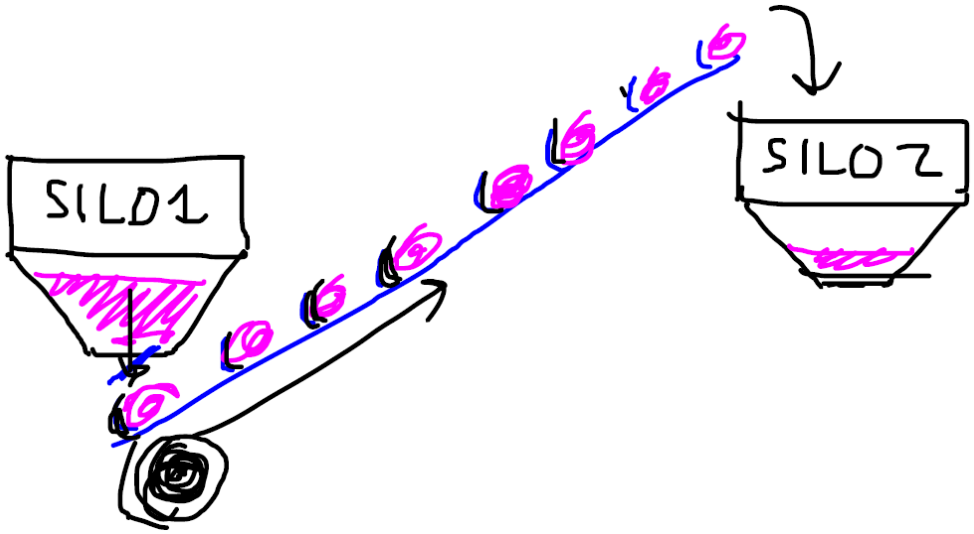
Sekarang, respon output sudah sangat stabil. Masalah berikutnya yang muncul adalah walaupun telah mencapai ketinggian fluida yang sangat mendekati *setpoint*, katup dan pompa terus bekerja selama ada *error* yang tercipta, walaupun nilainya sangat kecil. Ini bisa diatasi dengan memberikan koreksi *error,* berupa toleransi error yang masih dapat diterima. Misalnya untuk toleransi *error* yang baik untuk kondisi *hysteresis* relay dari -0.2 s/d 0.2 adalah ±0.4 cm. Sedangkan untuk kondisi *hysteresis* relay dari -0.1 s/d 0.1 toleransi ±0.1 cm sudah cukup baik. Berikut ini blok diagram lengkap dari PID beserta grafik respon sistem terhadap kondisi *hysteresis* relay dari -0.2 s/d 0.2 dengan toleransi ±0.4 cm:





Respon sistem seperti inilah yang dicari. Level ketinggian fluida dalam tangki akan tetap stabil (dengan toleransi *error* tertentu) walaupun ada gangguan yang muncul pada katup. Sedangkan apabila telah berada pada level ketinggian fluida yang ditentukan pada *setpoint*, maka katup dan pompa akan berhenti aktif, dan mencegah pompa serta katup beroprasi secara berlebihan.

BAB III
PROYEK SILO-TO-SILO



Proyek ini adalah membuat suatu model proses untuk memindahkan benda berupa pasir antara dua tempat penampung (disebut silo) menggunakan –lebih tepat disebut *elevator*– . ini dilengkapi cawan-cawan sebagai penahan tiap gundukan pasir. Silo yang digunakan berbentuk tabung pada bagian atas, dan kerucut terpancung terbalik pada bagian bawah. Ada dua silo yang digunakan, disebut Silo1 dan Silo2. Konfigurasi instalasi dapat dilihat pada gambar di atas.

Silo1 adalah tempat tampungan awal pasir. Pada bagian bawah Silo1 terdapat sebuah katup proporsional. Mulut katup akan langsung mengarah ke sebuah *conveyor*. *Conveyor* ini dilengkapi dengan cawan-cawan yang akan menahan gundukan pasir yang jatuh dari katup sehingga tidak jatuh selama perjalanan. *Conveyor* akan digerakkan oleh motor –dipisahkan oleh roda gigi–, sehingga cawan tersebut akan membawa setiap gundukan ke bagian ujung lainnya dari *conveyor* untuk ditumpahkan pada Silo2. Silo2 inilah yang akan kembali menampung setiap gundukan pasir yang telah dikeluarkan oleh Silo1. Pada Silo2, mulut katup akan tetap dibuat tertutup.

Berikut ini adalah data-data teknis yang dibutuhkan dalam perancangan model proses:

1. **Silo1**
   -Katup (proporsional; debit maks 1liter/20detik)
   -Penampung
   Diameter tabung 26 cm
   Tinggi tabung 20.5 cm
   Diameter atas kerucut terpancung terbalik 26 cm
   Diameter bawah kerucut terpancung terbalik 4.5 cm
   Tinggi kerucut terpancung terbalik 12 cm

2. **Silo2**
   -Katup (proporsional; debit maks 1liter/20detik)
   -Penampung
   Diameter tabung 26 cm
   Tinggi tabung 10.5 cm
   Diameter atas kerucut terpancung terbalik 26 cm

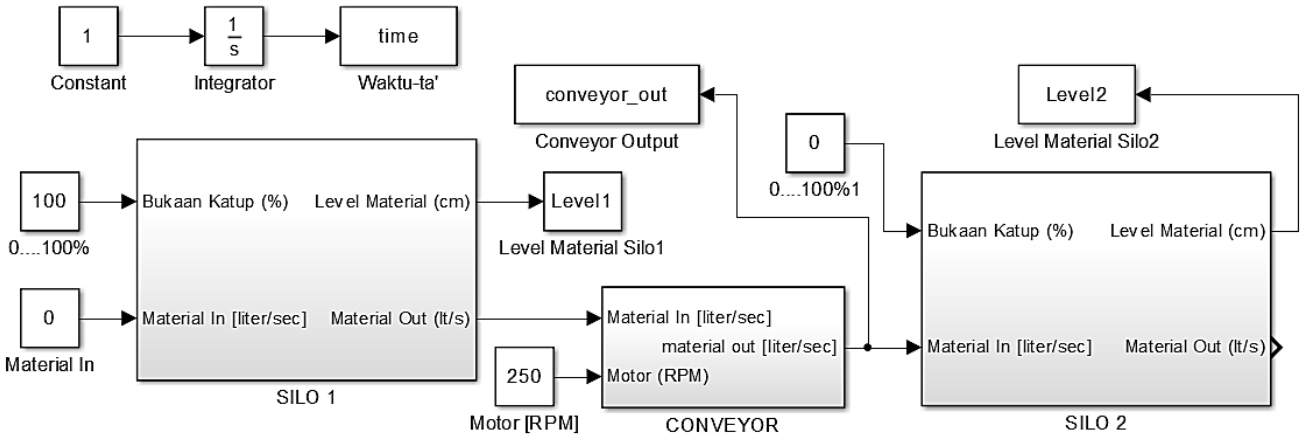
Diameter bawah kerucut terpancung terbalik 4.5 cm
Tinggi kerucut terpancung terbalik 12 cm

3. ***Conveyor***
   Kecepatan Motor 250rpm, (*adjustable*)
   Roda gigi (kecepatan *conveyor*) 1:5 motor (artinya sebanyak 250/5 cawan ditumpahkan setiap menit)
   Jumlah cawan di atas *conveyor* 10 cawan

Dari model di atas jelas terlihat bahwa ada tiga *subsystem* yang harus dibangun, Silo1, *conveyor* dan Silo2. Silo1 akan dapat diisi dengan material dari luar. Selain itu, perlu pula diatur seberapa besar katup pada Silo1 akan dibuka. Sebagai keluaran dari silo1, adalah material yang keluar dari katup. Untuk mengamati level material (dalam cm) pada Silo1, dapat pula ditambahkan sebuah variabel *monitoring* "*Level1"* pada keluaran lain dari Silo1.

Material keluaran dari Silo1 akan jatuh pada *conveyor* dan membentuk gundukan-gundukan pasir yang berjalan diatasnya. Selain banyaknya pasir dari Silo1 yang menjadi masukan dari *conveyor*, kecepatan motor perlu pula diatur untuk menentukan seberapa cepat gundukan material akan dibawa dan ditumpahkan ke Silo2. Untuk itu pada *conveyor* dibuat dua parameter masukan, yaitu *material_in* dan kecepatan motor untuk *conveyor*. Sedangkan hanya akan dibuat keluaran tunggal untuk *conveyor*, yaitu *material_out*, yakni berupa gundukan-gundukan pasir yang dibawa dan dijatuhkan pada Silo2. Akan dipasang sebuah variabel *monitoring* dengan nama "*_out"* untuk me*monitoring* bentuk keluaran dari *conveyor*.

Sedangkan untuk Silo2, secara blok diagram akan sama saja dengan Silo1, hanya saja akan ada sedikit perbedaan terkait dengan parameter-parameter saturasinya. Seperti halnya Silo1, pada Silo2 juga akan ditambahkan sebuah variabel *monitoring* dengan nama "*Level2"* untuk memantau ketinggian sesaat pasir pada Silo2.Sebagai catatan, katup pada Silo2 ini akan selalu dibuat tertutup, agar tidak ada material yang terbuang dengan percuma. Secara lengkap, model umum yang telah kita capai saat ini ditunjukkan pada gambar berikut:



1. **SILO1**

Telah dibahas sebelumnya bahwa bukaan katup merupakan salah satu parameter masukan dari Silo1. Selain bukaan katup, *Material_In* merupakan parameter masukan lainnya. Jika bukaan katup dinyatakan dalam persen[0..100], maka nilai bukaan ini perlu terlebih dahulu dikonversi ke nilai yang sebenarnya, yaitu dengan membagi nilai bukaan tersebut dengan 100. Untuk membagi, kita dapat menggunakan sebuah *transfer fucntion*. Dari data teknis diketahui bahwa katup memiliki debit 1 liter / 20 detik untuk bukaan penuh, atau setara dengan 0.05liter/detik. Seperti biasanya, kita akan membandingan nilai bukaan katup terhadap nilai maksimum untuk mendapatkan debit yang sebanding dengan buka
an katup tersebut. Untuk membandingkan, kita dapat kembali menggunakan sebuah blok *product.*Hasil dari membandingkan dengan blok ini akan memiliki keluaran berupa debit keluarnya material dari Silo1 dalam liter-per-detik, sebanding dengan nilai bukaan katup yang diberikan.

Jika *material_in* dalam liter/detik adalah debit masuknya material baru pada Silo, sedangkan bukaan katup menunjukkan debit keluarnya material dari Silo, maka debit pertambahan material pada Silo adalah *material_in* dikurangi debit bukaan katup. Untuk itu, dari parameter masukan *material_in* pada Silo langsung dapat dikurangkan dengan debit bukaan katup dalam satuan yang sama. Digunakan sebuah blok *sum.* Sekarang kita telah mendapatkan debit pertambahan material rata-rata sesaat dalam liter/detik pada Silo.

Seperti teknik pada proyek sebelumnya, jika debit adalah turunan volume terhadap waktu, maka volume merupakan integral dari debit terhadap waktu. Untuk itu, keluaran dari blok *sum* akan dimasukkan pada sebuah blok *integrator* untuk mendapatkan volume dari Silo dalam liter. Blok *integrator* yang digunakan adalah blok yang dilengkapi *saturation.* Alasannya masih sama, agar volume pada Silo tidak akan pernah bernilai negatif, sekalipun dilakukan pembatasan tinggi maretial pada bagian akhir . Penjelasan lebih rinci tentang blok ini dapat dilihat pada pembahasan sebelumnya (halaman 6 sub-bab 1.3. *level controlled tank*).

Sebagai referensi, untuk kerucut terpancung berlaku rumus $V = \frac{1}{3}\pi h(R^2 + Rr + r^2)$, dengan h adalah tinggi kerucut terpancung, *R* jari-jari tutup besar, dan *r* jari-jari tutup kecil. Dengan rumus ini ditambah dengan rumus volume tabung, didapatkan bahwa volume maksimum dari Silo1 adalah 13.4389 liter. Untuk itu pada batasan blok *integrator* yang digunakan diberi batasan saturasi dari 0 sampai 13.4389. Sedangkan *initial condition* merupakan volume awal dari Silo1 (misalnya 5 liter).

Variabel *monitoring* "*Level1"* yang dipasang bertujuan untuk memantau level ketinggian material pada Silo1 dalam satuan cm. Untuk itu perlu dilakukan konversi dari volume material dalam liter menjadi ketinggian material dalam cm. Dalam melakukan tugas ini, pilihan paling mudah adalah dengan menggunakan sebuah blok "*Interpreted MATLAB Fcn",* yakni sebuah blok yang memiliki respon keluaran berdasarkan *listing program* yang diketikkan oleh pengguna. Untuk melakukan tugas ini, berikut *listing program* yang digunakan:

```
function y = volume_to_level(x)
V = 1000*x; % mendefinisikan volume sebagai masukan dalam liter
R = 26/2; % Jari-jari silinder/tutup besar kerucut
r = 4.5/2; % Jari-jari katup /tutup kecil kerucut
h = 12; % Tinggi kerucut terpancung tersisa
Vk = (1/3)*(pi)*(R^2 + (R*r) + r^2)*h; % Volume kerucut terpancung
```

```
ho = (r/(R - r))*h; % Tinggi potongan kerucut (yang terpotong)
if V >= Vk %Jika kerucut telah penuh terisi material
    y = h + ((V-Vk)/(pi*(R^2))); % level material adalah tinggi kerucut
ditambah tinggi volume pada tabung
else % Jika kerucut belum penuh
    y = (((3*V*(ho^2)) + (pi*(r^2)*(ho^3)))/(pi*(r^2)))^(1/3) - ho;
end
```

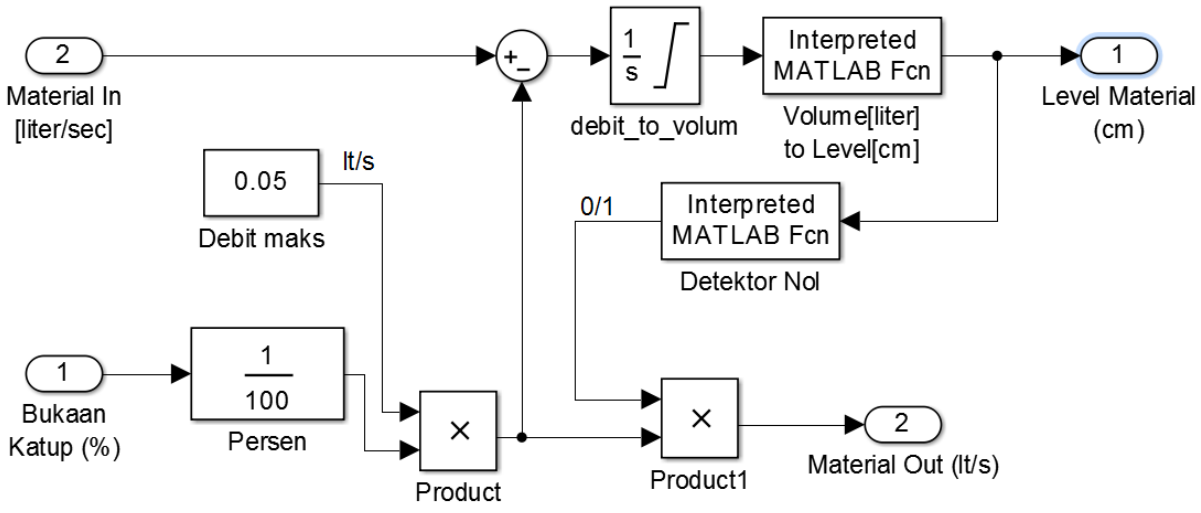
Untuk kerucut yang belum penuh, digunakan dari persamaan $t = \sqrt[3]{\frac{\pi\, h0^3\, r^2 + 3\, Vh0^2}{\pi\, r^2}} - h0.$

Keluaran dari blok fungsi terprogram ini dapat langsung menjadi keluaran dari *subsystem* Silo1 untuk dipantau melalui variabel *monitoring* "*Level1*". Namun, masalah yang muncul adalah bagaimana seandainya bila katup terus terbuka sedangkan material pada Silo1 telah habis. Pada *integrator* mungkin tidak akan menjadi masalah, karena telah diberi batasan sebelumnya. Masalah yang sebenarnya adalah pada material keluaran (*material_out*) dari Silo1 yang akan diterima oleh *conveyor*. Perlu ada sebuah blok pembatas yang me-nolkan material keluaran saat material di dalam Silo telah habis. Untuk itu, digunakan sebuah blok *Interpreted MATLAB Fcn(disebut detektor_nol)* dari level ketinggian material (dalam cm), yang keluarannya dikalikan dengan debit keluaran Silo1. Jika material belum habis, blok *Interpreted MATLAB Fcn* akan memberikan isyarat 1, sehingga tidak akan mengganggu nilai debit keluaran dari Silo1. Sedangkan apabila material sudah habis, akan diberikan isyarat 0, sehingga debit hasil perkalian akan ikut menjadi nol. Dari hasil perkalian inilah yang benar-benar sudah dapat menjadi keluaran dari *subsystem* Silo1 untuk ditujukan ke *conveyor*.

Berikut adalah *listring program* dari blok *detektor_nol*:

```
function y=detektor_nol(x)
if x==0
    y=0;
else
    y=1;
end
```

Berikut adalah blok diagram dari *subsystem* Silo1 secara utuh:



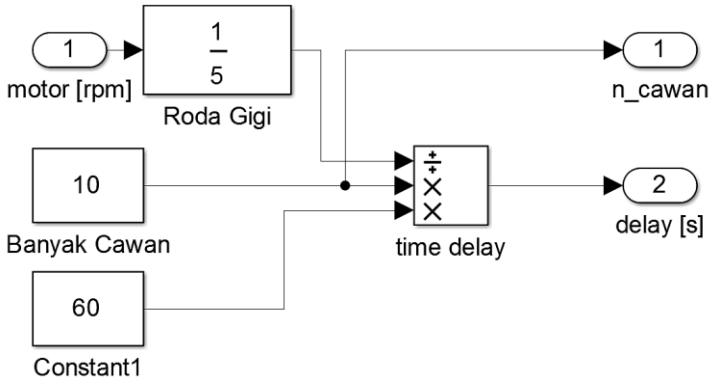
2. *CONVEYOR*

Subsystem ini bertujuan untuk mengubah volume yang masuk akan mengalami perubahan bentuk menjadi gundukan-gundukan material pada setiap cawannya, disertai dengan *delay* akibat dari proses memindahkan material. Kedengarannya sederhana, tapi *subsystem* inilah yang paling rumit.

Akan ada dua masukan dari *subsystem* ini, yakni *material_in* dan kecepatan motor *conveyor* dalam rpm. Akan kembali dibuat dua buah *subsystem*, yakni *subsystem* yang menerangkan data teknis dari *conveyor*, serta *subsystem* yang membentuk material masukan menjadi gundukan-gundukan material.

2.1. KETERANGAN SISTEM

Bagian inilah yang akan menerangkan data-data teknis dari *conveyor* sehingga menjadi hal berguna untuk bagian lainnya. Jika diketahui kecepatan motor sebagai parameter masukan adalah *x* rpm, maka *conveyor* akan menjatuhkan sebanyak *x/5* cawan dalam setiap menit, atau setiap cawan membutuhkan $60 : \frac{x}{5}$ detik untuk dijatuhkan. Karena ada 10 buah cawan di atas *conveyor*, maka dari mulai mengisi cawan sampai mulai menjatuhkannya pada ujung satunya, dibutuhkan waktu selama 10 kali dari waktu menjatuhkan 1 cawan, sebagai waktu dalam detik. Jika keluaran dari blok ini adalah banyak cawan (*n_cawan)* dan , maka sistem ini dapat direalisasikan dengan susunan blok diagram sebagai berikut:



2.2. PEMBENTUK GUNDUKAN

Subsystem inilah yang paling rumit. *Subsystem* ini dibuat dengan tiga buah parameter masukan; *material_in*, *n_cawan* (jumlah cawan), dan .

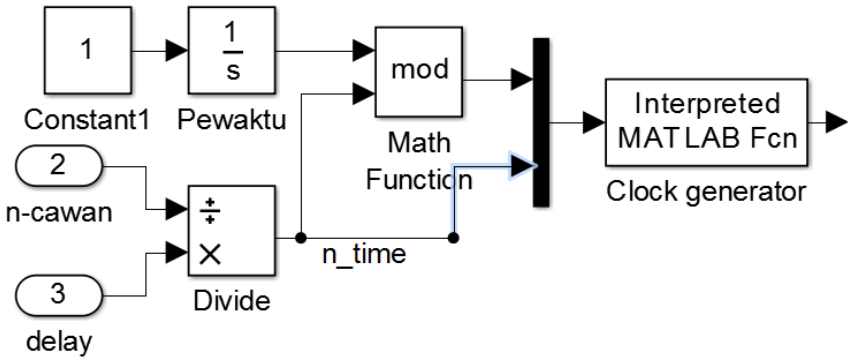
Pertama-tama, kita akan membutuhkan sebuah nilai yang menunjukkan waktu untuk mengisi setiap cawan dari tumpahan katup Silo1 (dinamakan *n_time). n_time* dapat dengan mudah didapatkan dengan membagi waktu totoal terhadap jumlah cawan yang ada (*n_cawan).* Waktu ini akan menjadi salah satu keluaran *subsystem*, karena setara dengan waktu yang dibutuhkan untuk memproses 1 buah gundukan pada setiap cawan, sehingga disebut juga *process time*.

Rangkaian pembentuk gundukan akan dibuat bekerja berdasarkan jumlah volume dari Silo1 yang diterima setiap waktu pengisian 1 cawan. Artinya yang kita butuhkan sekarang adalah sebuah pewaktu yang menentukan kapan volume dari setiap gundukan material akan mulai

dihitung, menahan volume tersebut sampai gundukan berikutnya, dan mengulang parameter waktu sebagai parameter fungsi untuk pembentuk gundukan –akan dibahas kemudian.

Sebagai pembuka, kita akan menghitung volume material yang diterima *conveyor* untuk mengisi satu buah cawan dan menjadi sebuah gundukan. Volume ini dapat kembali kita dapatkan dengan blok *integrator*. Namun, kita harus terus mengulang menghitung untuk gundukan-gundukan berikutnya. Untuk itu kita membutuhkan sebuah blok *integrator* yang dilengkapi dengan kemampuan *external* . Jika kita akan melakukan setiap selesai menghitung volume untuk satu gundukan, dan yang kita gunakan adalah mode reset tepi turun (*falling*), maka yang kita butuhkan adalah memiliki tepi turun setiap pergantian satu periode *clock*. Artinya *clock* yang akan dibuat adalah bernilia nol selama setengah periode awal *clock*, dan bernilai satu pada setengah periode berikutnya. Untuk melakukannya, kita dapat menggunakan sebuah blok *Interpreted MATLAB Fcn*, namun dengan beberapa parameter masukan.

Kita dapat menggunakan blok diagram beserta *listring program* berikut untuk membangkitkan *clock* yang tepat:




```
function y=fclock(x)
waktu=x(1); %waktu dari sisa bagi waktu terhadap waktu n_time
delay=x(2); %waktu untuk memproses setiap cawan (n_time)
banding=delay/2; %setengah periode n_time
if waktu<banding
    y=0; %output 0 selama setengah siklus pertama clock
else y=1; %output 1 selama setengah siklus berikutnya
end
end
```

Pertama kita akan membutuhkan sebuah pewaktu yang terus membangkitkan waktu selama simulasi dijalankan. Sisa hasil bagi waktu terhadap waktu setiap cawan (*n_time)* akan menghasilkan nilai dari 0 sampai *n_time*, yakni lamanya waktu setiap cawan dari mulai diisi. Bisa didapatkan dengan menggunakan fungsi “mod” dari blok *Math Function. n_time* juga menunjukkan lamanya waktu untuk 1 periode *clock*. Dari range nilai ini, kita dapat membuat keluaran akan bernilai 0 selama sisa hasil bagi lebih kecil dari setengah siklus *clock*, dan akan bernilai 1 selama setengah siklus berikutnya, yakni jika sisa hasil bagi telah lebih besar dari setengah siklus *clock*. Akhirnya kita telah mendapatkan sebuah sistem *clock* yang memberikan tepi turun setiap volume untuk 1 cawan telah selesai dihitung.

Setelah selesai dihitung volume untuk setiap cawan, perlulah membentuk gundukan material dengan volume yang tepat sama dengan nilai tersebut. Kita dapat membentuk gundukan

material namun dalam satuan liter/detik, karena sebagai masukan untuk silo berikutnya, dapat digunakan sebuah blok *Interpreted MATLAB Fcn.* Jika keluaran berbentuk debit yang dibentuk seperti gundukan-gundukan (seperti grafik positif fungsi sinus), namun dengan tetap memperhatikan volume yang diterima, dengan ilmu kalkulus dapat dibentuk persamaan berikut:

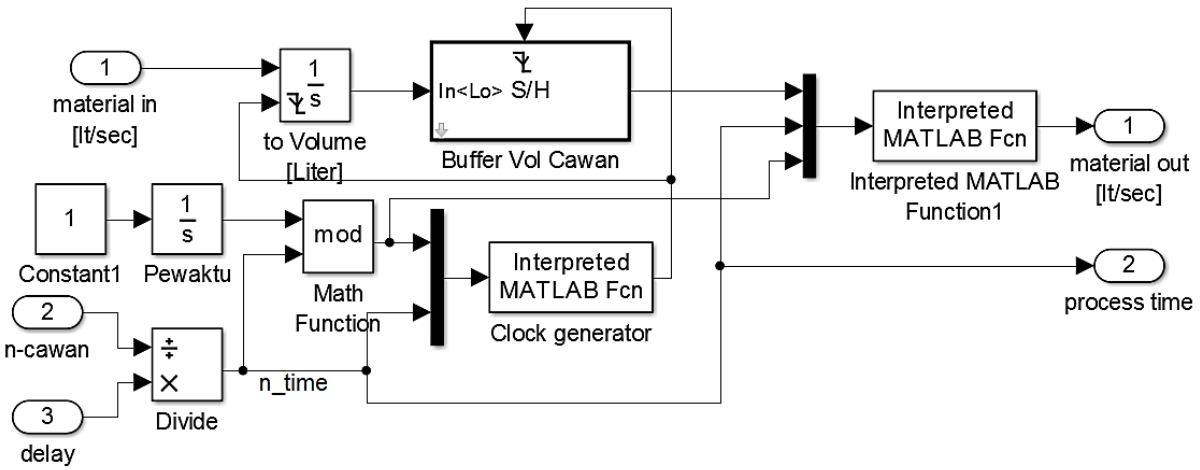
$$Vin = \int_0^{n_{time}} \mathrm{A} \cdot \sin(\omega \cdot t)\, dt \;\; ; dengan\ \omega = \frac{\pi}{n_{time}}$$

$$A = \frac{\pi \cdot Vin}{2 \cdot n\_time}$$

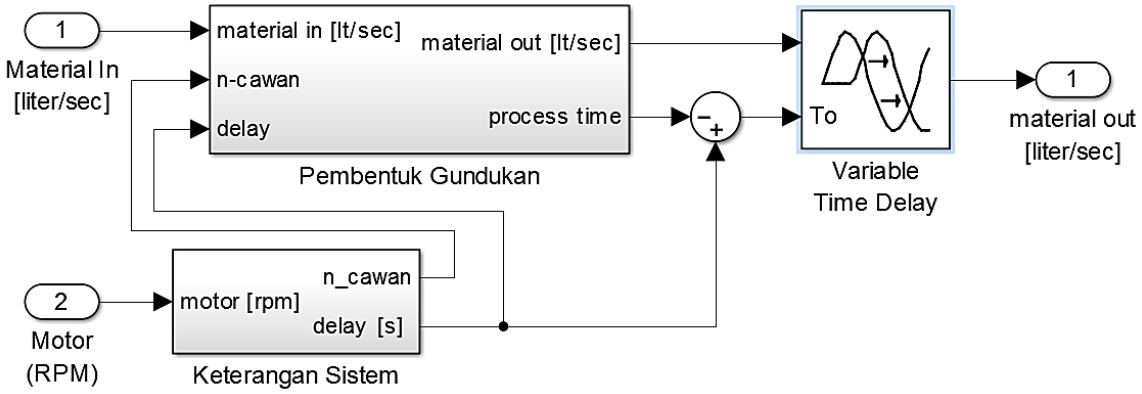
Dari persamaan ini, kita dapat membuat sebuah fungsi yang mendefiniskan volume yang tepat dari bentuk-bentuk gundukan yang dihasilkan, walau dalam satuan liter/detik. Namun, variabel *n_time* pada *listring program* berikut akan digantikan dengan variabel *"waktu"* yang ukurannya sengaja dibuat hanya 80% dari *n_time*, agar jarak antar satu gundukan dan gundukan berikutnya tetap terlihat. Berikut ini adalah *listring program*-nya:

```
function y=pembentuk_cawan(x)
vol=x(1); %volume untuk setiap cawan
n_time=x(2); %waktu pengisian tiap cawan
ftime=x(3); %parameter waktu
waktu=n_time-(n_time/5);  %panjang gundukan dalam detik
omega=pi/waktu; %kecepatan sudut fungsi
A=0.5*pi*vol/waktu; %menentukan amplitudo yang tepat agar volume
sesuai
if ftime<=waktu
    y=A*sin(omega*ftime);
else y=0;
end
end
```

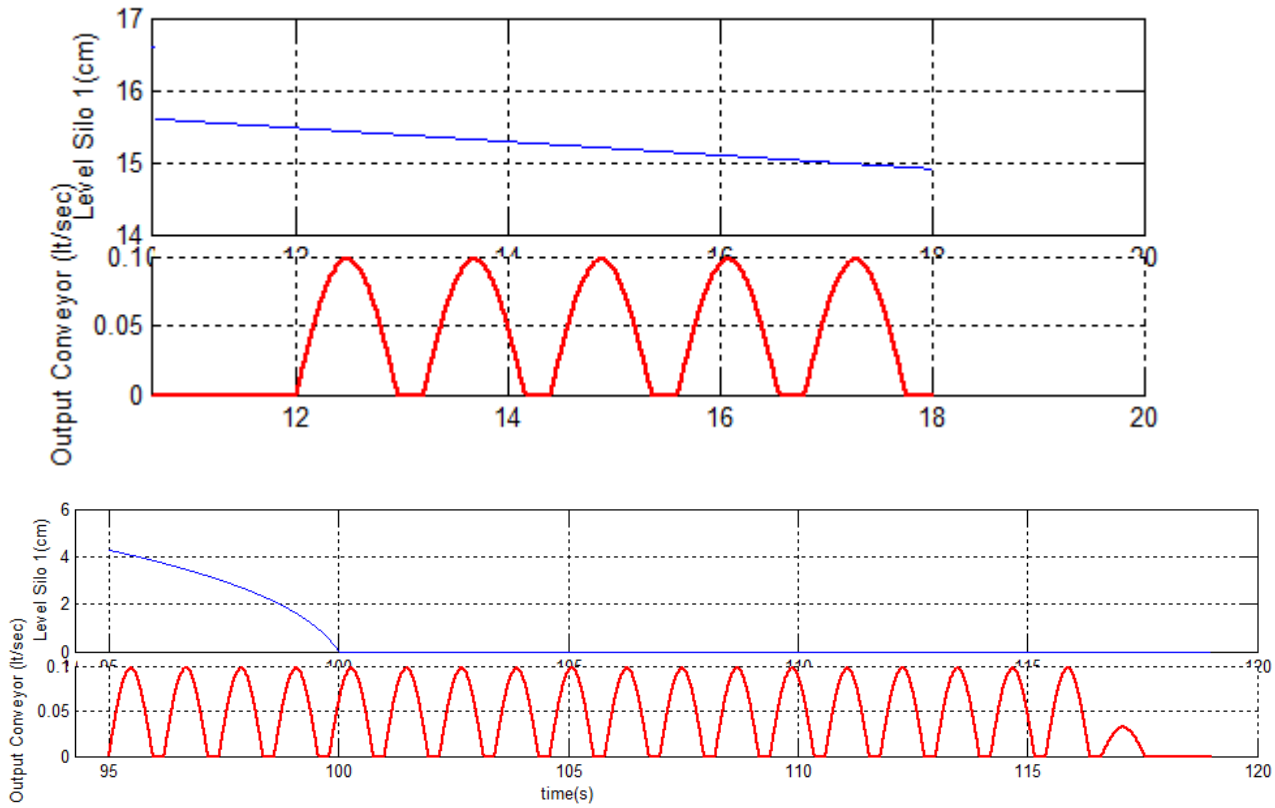
Namun, masalahnya adalah kita membutuhkan sebuah nilai masukan dari fungsi berupa volume yang akan dibuat gundukan berdasarkan hitungan dari blok *integrator*. Artinya, kita harus menahan hasil hitungan blok *integrator* tepat sebelum nilainya di- selama satu periode *clock*. Untuk melakukannya, kita membutuhkan sebuah blok yang tepat, *Sample and Hold,* dengan *clock* yang dibuat sama dengan *clock* reset dari blok *integrator* tersebut. Secara lengkap, berikut ditunjukkan blok diagram dari *subsystem* Pembentuk Gundukan:



Subsystem ini akan memakan waktu sebesar waktu untuk mengisi 1 cawan. Artinya, kita masih membutuhkan waktu sebesar 9 kali proses seperti ini lagi. Nilai *delay* sebesar itu adalah sama saja mengurangkan waktu *delay* dari *subsystem "Keterangan Sistem"* dengan waktu proses yang dipakai oleh blok *subsystem "Pembentuk Gundukan".* Untuk memberikan *delay*, kita dapat menggunakan sebuah blok *Variable Time* sebelum gundukan-gundukan material dibawa keluar , dengan nilai *delay* dari hasil pengurangan tersebut. Berikut ini adalah gambaran blok diagram *conveyor* secara lengkap:



Berikut ini respon keluaran dari *conveyor* dengan kecepatan motor 250rpm:

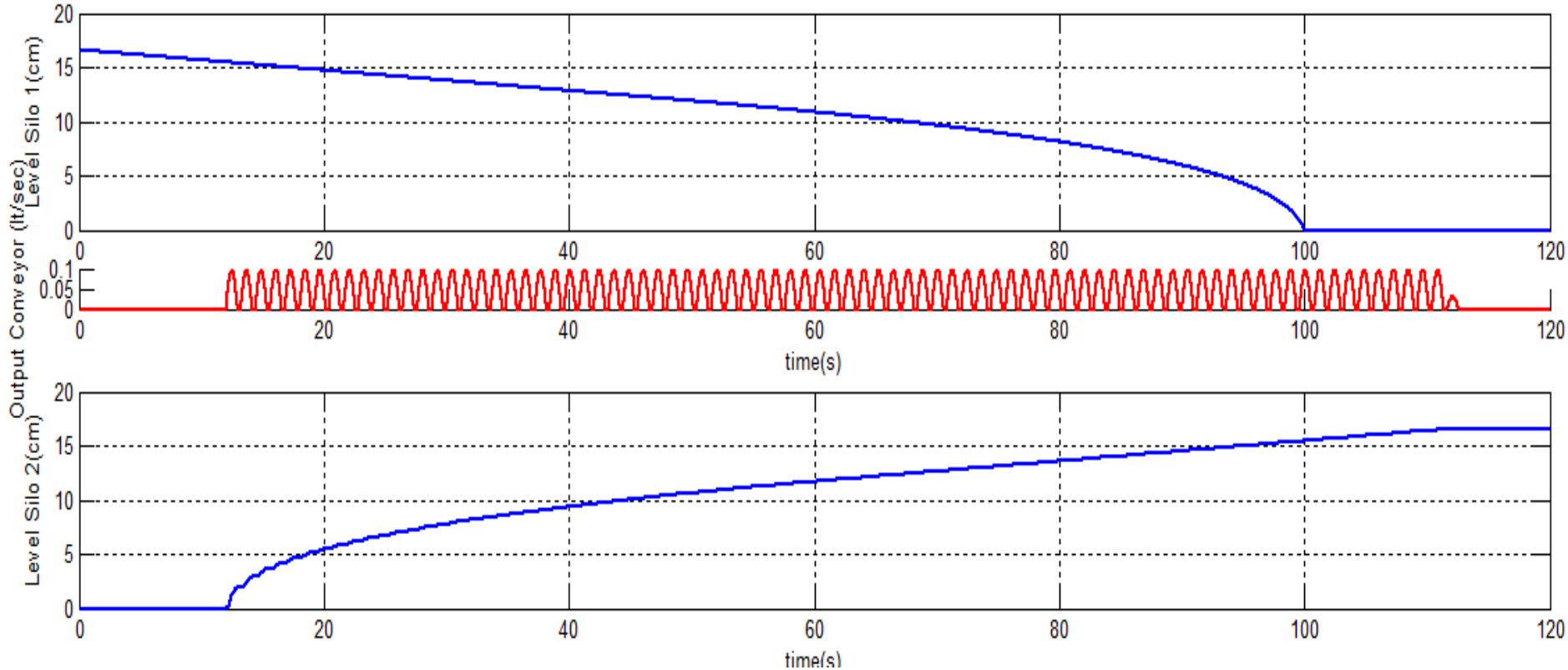


Terlihat bahwa respon keluaran dari *conveyor* berbentuk gundukan-gundukan material yang amplitudonya menyesuaikan dengan jumlah material yang masuk.

3. SILO2

Secara umum, blok diagram Silo2 akan sama saja dengan blok diagram pada Silo1, hanya saja Silo2 hanya dapat menampung material yang lebih sedikit dari Silo1. Untuk tinggi tabung pada Silo2, hanya mampu menampung material hingga 8.1296 liter. Untuk itu, perbedaan dari *subsystem* pada Silo2 hanya terletak pada batasan saturasi dari blok *integrator*-nya.

Berikut adalah grafik respon pemindahan material dari model Silo_to_Silo ini:



Respon yang baik adalah apabila ketinggian awal pada Silo1 akan sama dengan ketinggian akhir pada Silo2.

BAB IV
PENUTUP

KESIMPULAN

Sebagai mahasiswa elektro di bidang teknik kendali, mempelajari pemodelan sistem akan sangat penting. Selain untuk memonitoring sistem, pemodelan dapat membantu dalam melakukan analisis terhadap kesalahan-kesalahan sistem yang mungkin terjadi untuk meminimalisir kerugian dalam suatu proses. Namun, dibutuhkan keahlian yang baik dalam memodelkan sistem secara benar, karena pemodelan yang kurang tepat tentu saja dapat mengakibatkan kesalahan dalam memonitoring suatu proses.