ANNAMALAI UNIVERSITY

FACULTY OF ENGINEERING AND TECHNOLOGY

DEPT. OF ELECTRONICS AND INSTRUMENTATION ENGINEERING

M.E DEGREE (PC&I)- II SEMESTER

PCIP-207COMPUTER PROCESS CONTROL LAB

LIST OF EXPERIMENTS

1. **DESIGN AND SIMULATION OF SAMPLED DATA CONTROL SYSTEM WITH DEADBEAT CONTROLLER USING TUTSIM**

2. **a.DESIGN AND SIMULATION OF SAMPLED DATA CONTROL SYSTEM WITH DAHLIN'S CONTROLLER USING TUTSIM**
   b. DESIGN AND SIMULATION OF KALMAN'S CONTROLLER USING TUTSIM

3. **a. DEAD TIME COMPENSATION USING SMITH PREDICTOR ALGORITHM SIMULATION USING SIMULINK**
   b. **DESIGN AND TESTING OF INVERSE RESPONSE COMPENSATOR USING SIMULINK**

4. **AIR TEMPERATURE CONTROL SYSTEM**

5. a. **STUDY OF LABVIEW SOFTWARE**
   b. **SIMULATION OF A TEMPERATURE PROCESS USING LABVIEW**

6. a. **STUDY OF PROGRAMMABLE LOGIC CONTROLLER**
   b. **DIRECTION CONTROL OF A DC MOTOR USING PROGRAMMABLE LOGIC CONTROLLER**

7. **PROCESS IDENTIFICATION USING LEAST SQUARE ESTIMATION**

8. **STUDY OF SCADA SOFTWARE**

9. **PC BASED CONTROL OF SIMULATED PROCESS**

10. **STUDY OF FUZZY TOOL BOX AND DESIGN OF FUZZY LOGIC CONTROLLER FOR A GIVEN PROCESS**

(Dr.S.P.NATARAJAN)
HOD

(D.RATHIKARANI)
LAB INCHARGE

EX.NO:	DESIGN AND SIMULATION OF SAMPLED DATA CONTROL SYSTEM WITH DEADBEAT CONTROLLER USING TUTSIM
DATE:	

AIM

To design a deadbeat controller for a given process and to find the closed loop response of the sampled data system to a unit step change in the set point incorporating deadbeat algorithm.

THEORY

The block diagram of Direct Digital Control Loop (DDC) is as shown in Figure 1. For a set point change the closed loop response is given by

$$\ddot{Y}(z) = \frac{HG_P(z)D(z)}{1 + HG_P(z)} \ddot{Y}_{SP}(z) \qquad (1)$$

Where $HG_P(z)$ =Pulse transfer function of the process with Zero Order Hold.

In DDC for a given step change in the set point the discrete-time response should be specified. Then if $HG_P(z)$, $\ddot{Y}(z)$ and $\ddot{Y}_{sp}(z)$ are known we can solve equation (1) with respect to unknown D(z).

The transfer function of the digital controller is given by

$$D(z) = \frac{1}{HG_P(z)} * \frac{Y(z)/Y_{SP}(z)}{1 - Y(z)/Y_{SP}(z)} \qquad (2)$$

DESIGN OF DEADBEAT CONTROLLER FOR A GIVEN PROCESS

The closed loop response for a deadbeat control algorithm to a unit step change in the input exhibits no error at all sampling instants after the first instant. If the response is to have zero error at all sampling instants after the first, its discrete-time behaviour resembles that of a unit step delayed by one sampling instant. Therefore,

$$Y(z) = \frac{z^{-1}}{1 - z^{-1}} \qquad (3)$$

$$Y_{SP}(Z) = \frac{1}{1 - z^{-1}} \qquad \frac{Y(z)}{Y_{sp}(z)} = z^{-1} \qquad (4)$$

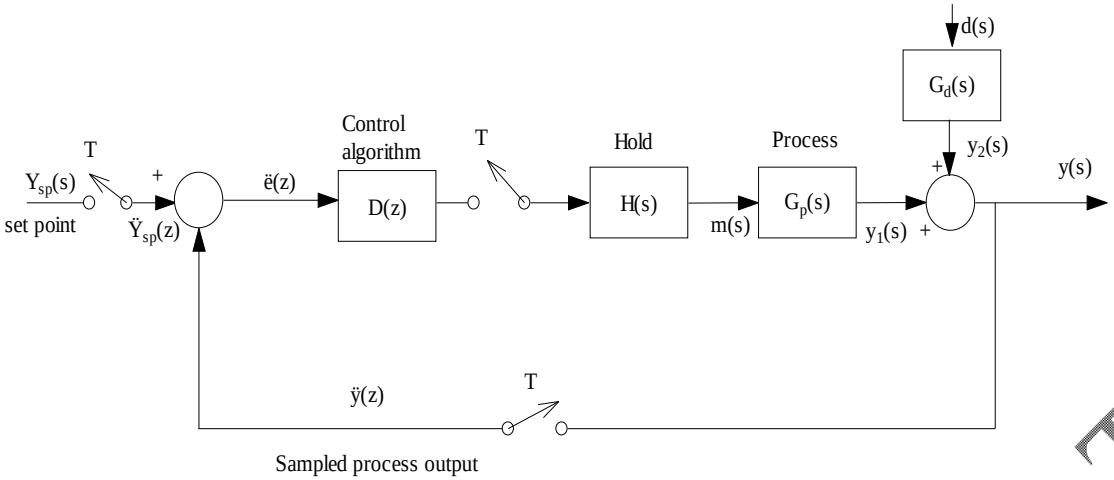


Figure 1. Block Diagram of Direct Digital Control Loop

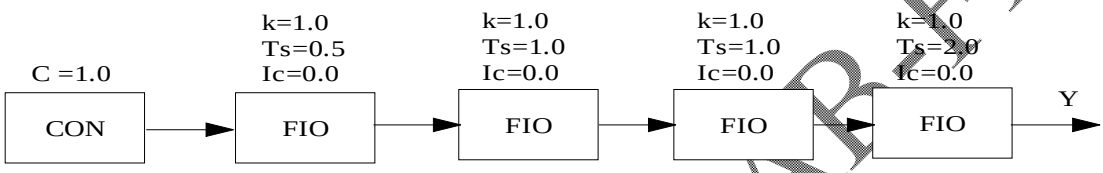


Figure 2. Block Diagram to obtain Open Loop Step Response

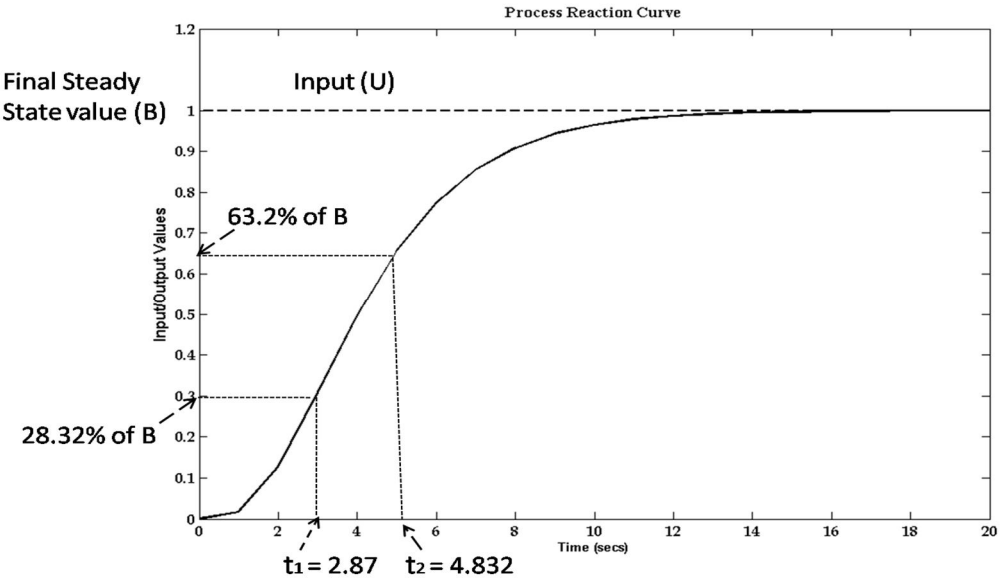


Figure 3. Open Loop Step Response of the higher order process for the step input of magnitude "u"

Substituting equations (3) and (4) in equation (2)

$$D(z) = \frac{1}{HG_P(z)} * \frac{z^{-1}}{1 - z^{-1}} \tag{5}$$

PROBLEM

Design a dead beat controller for the process

$$G_P(s) = \frac{1}{(0.5s+1)(s+1)(s+1)(2s+1)}$$ with Sampling Time T=5 seconds (6)

DESIGN

First approximate the given process into first order plus dead time model using process reaction curve method. The block diagram schematic is as shown in Figure 2 and the open loop response obtained is as shown in Figure 3.

The approximated first order time delay model is obtained from Figure 3 and is given by

$$G_P(s) = \frac{1e^{-1.889s}}{(2.943s+1)} \tag{7}$$

The pulse transfer function is given by

$$HG_P(z) = Z[H(s)\,G_P(s)]$$

where H(s) = zero order hold transfer function and is given by

$$H(s) = \left(\frac{1 - e^{-sT}}{s}\right)$$

$$HG_P(z) = Z\left[\left(\frac{1 - e^{-sT}}{s}\right)\left(\frac{1 * e^{-1.889s}}{(2.943s+1)}\right)\right]$$

$$= (1 - Z^{-1}) * Z\left[\frac{e^{-1.889s}}{S * (2.943S+1)}\right]$$

Sampling time T = 5 seconds

$$\lambda T = t_d$$

$$\lambda T = 1.889$$

$$\lambda = \frac{1.889}{5} = 0.3778$$

$$m = 1 - \lambda$$

$$m = 1 - 0.3778 = 0.6222$$

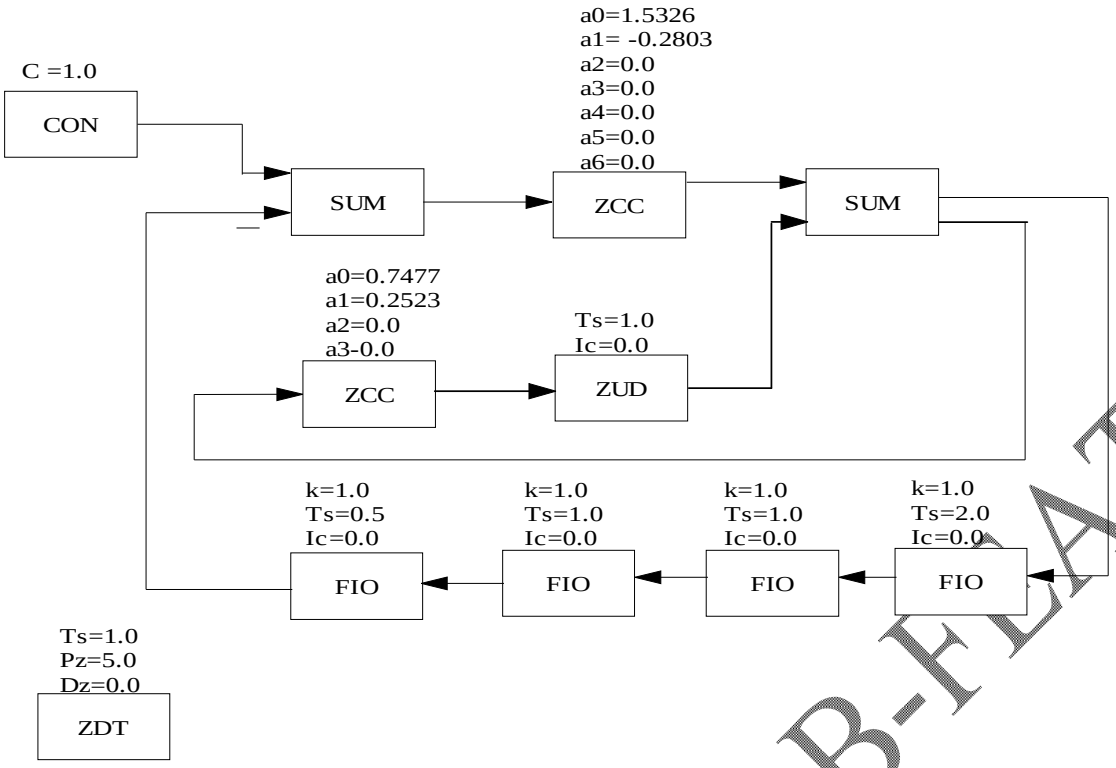


Figure 4. TUTSIM Block Diagram of Deadbeat Controller for actual process

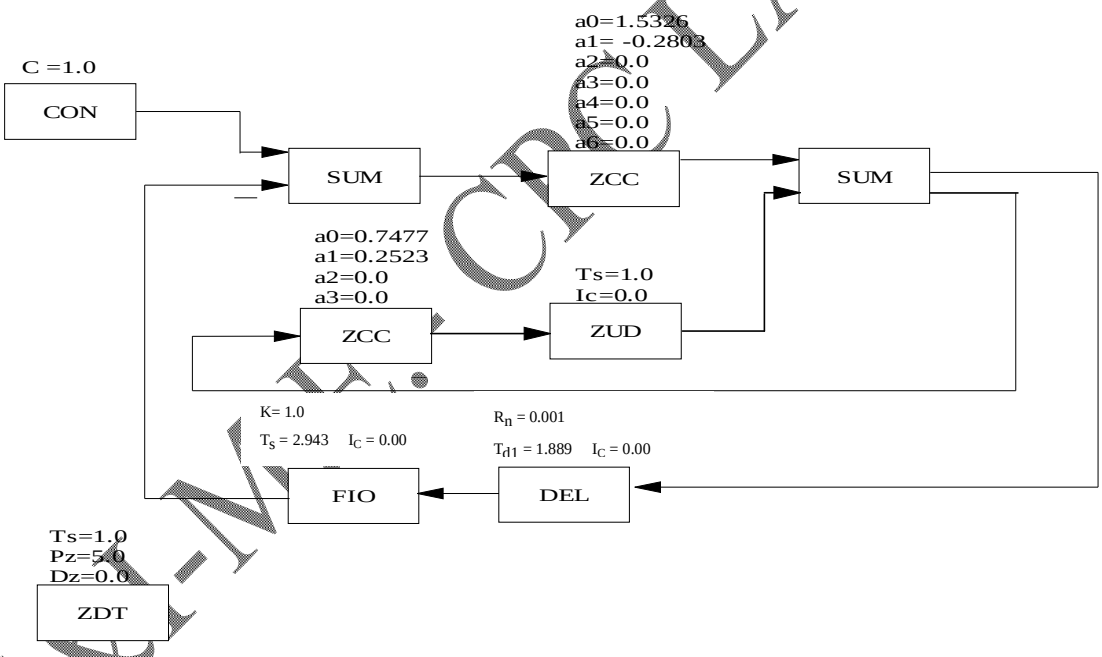


Figure 5. TUTSIM Block Diagram of Deadbeat Controller for approximated process

$$\therefore HG_p(z) = (1-z^{-1}) * Zm\left[\frac{1}{s(2.943s+1)}\right]_{m=0.6222}$$

$$= (1-z^{-1}) * Zm\left[\frac{1/2.943}{s(s+1/2.943)}\right]_{m=0.6222}$$

$$= (1-z^{-1}) * Zm\left[\frac{0.3398}{s(s+0.3398)}\right]_{m=0.6222,\ a=0.3398}$$

$$HG_p(z) = (1-z^{-1}) * Zm\left[\frac{0.3398}{s(s+0.3398)}\right]_{m=0.6222, a=0.3398}$$

We Know

$$Zm\left[\frac{a}{S(S+a)}\right] = \left[\frac{1}{z-1} - \frac{e^{-amT}}{z-e^{-aT}}\right]$$

$$HG_p(z) = (1-z^{-1}) * Zm\left[\frac{0.3398}{s(s+0.3398)}\right]_{m=0.6222, a=0.3398}$$

$$= (1-z^{-1})\left[\frac{1}{z-1} - \frac{e^{-(0.3398)\ (0.6222)\ (5)}}{z-e^{-(0.3398)\ (5)}}\right]$$

$$= (1-z^{-1})\left[\frac{1}{z-1} - \frac{0.3475}{z-0.1829}\right]$$

$$= \frac{z-1}{z}\left[\frac{1}{z-1} - \frac{0.3475}{z-0.1829}\right]$$

$$= \frac{1}{z} - \frac{(0.3475z - 0.3475)}{(z^2 - 0.1829z)}$$

$$= \frac{z^2 - 0.1829z - 0.3475z + 0.3475z}{z^3 - 0.1829z^2}$$

$$= \frac{0.6525z^2 + 0.1646z}{z^3 - 0.1829z^2}$$

$$= \frac{z(0.6525z + 0.1646)}{z(z^2 - 0.1829z)}$$

$$HG_p(z) = \frac{0.6525z + 0.1646}{z^2 - 0.1829z}$$

$$= \frac{z(0.6525 + 0.1646z^{-1})}{z^2(1-0.1829z^{-1})}$$

$$= \frac{0.6525 + 0.1646z^{-1}}{z - 0.1829}$$

$$= \frac{z^{-1}(0.6525 + 0.1646z^{-1})}{(1-0.1829z^{-1})}$$

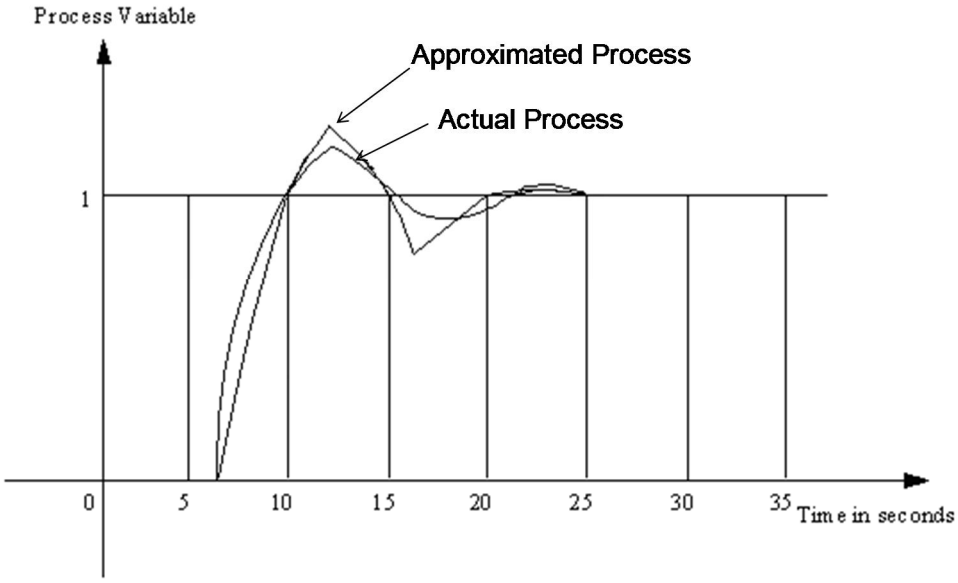


Figure 6. Response of Deadbeat Controller for both given process and approximated process

$$D(z)=\frac{1}{HG_p(z)}*\frac{z^{-1}}{1-z^{-1}}$$

$$D(z)=\frac{\left(1-0.1829z^{-1}\right)}{z^{-1}\left(0.6525+0.1646z^{-1}\right)}*\frac{z^{-1}}{1-z^{-1}}$$

$$D(z)=\frac{\left(1-0.1829z^{-1}\right)}{\left(0.6525-0.6525z^{-1}+0.1646z^{-1}-0.1646z^{2}\right)}$$

$$\frac{m(z)}{e(z)}=\frac{\left(1-0.1829z^{-1}\right)}{\left(0.6525-0.4879z^{-1}-0.1646z^{-2}\right)}$$

$$0.6525m(z)-0.4879Z^{1}m(z)-0.1646Z^{2}m(z)=e(z)-0.1829e(z)z^{-1}$$

Taking inverse Z-transform

$$m_n = 0.7477m_{n-1} + 0.2523m_{n-2} + 0.15326e_n - 0.2803e_{n-1}$$

EXPERIMENTAL PROCEDURE

(i) Give the step input of magnitude 'u' for the given higher order process as shown in Figure 2.

(ii) Obtain the process reaction (S- shaped) curve as shown in Figure 3.

(iii) From the process reaction curve find the steady state value(B), t_1 and $t_2$where t_1=time corresponds to 28.3% of B and t_2= time corresponds to 63.2% of B.

(iv) Approximate the first order plus dead time model using the given formulae as $G_p(s)=\frac{K_p e^{t_d s}}{\tau s+1}$ where K_P is the process gain, τ is the process time constant and t_d is the process dead time.

$$K=\frac{\text{Change in steady state value}}{\text{change in the input}}=\frac{B}{U}$$

$$\tau=1.5*(t_2-t_1) \qquad (8)$$

$$t_d=t_2-\tau$$

(v) Derive the pulse transfer function $HG_P(z)$.

(vi) Obtain the digital controller D(z) as given in the design.

(vii) Implement the transfer function of the dead beat controller by using Z-blocks in TUTSIM for actual and approximated process as shown in Figure 4 and Figure 5 respectively.

(viii) Analyze the response and comment on your results.

RESULT

The closed loop response of a sampled data system to a unit step change in the set point has been obtained after designing a dead beat controller.

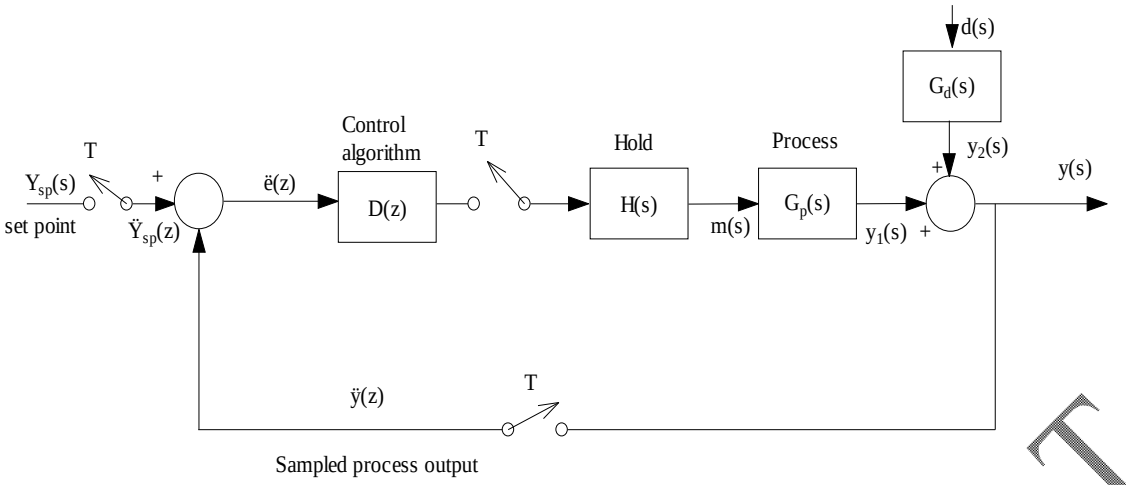


Figure 1. Block Diagram of Direct Digital Control Loop

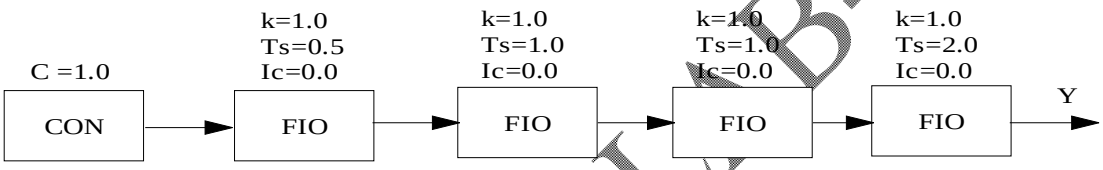


Figure 2. Block Diagram to obtain Open Loop Step Response

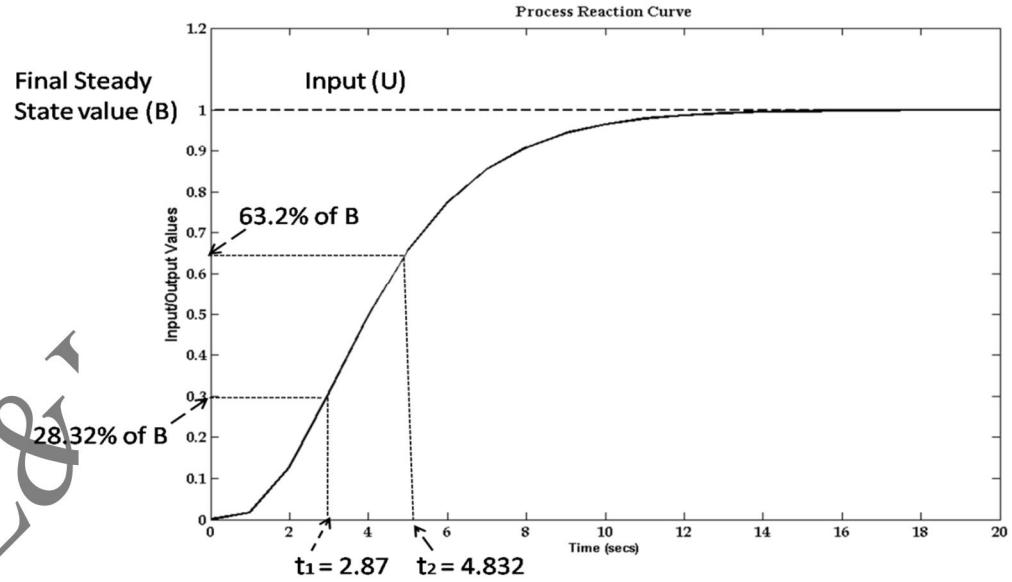


Figure 3. Open Loop Step Response of the higher order process for the step input of magnitude "u"

EX.NO:	DESIGN AND SIMULATION OF SAMPLED DATA CONTROL SYSTEM WITH DAHLIN'S CONTROLLER USING TUTSIM
DATE:	

AIM

To design a dahlin's controller for a given process and to find the closed loop response of the sampled data system to a unit step change in the set point incorporating dahlin's algorithm.

THEORY

The block diagram of Direct Digital Control Loop (DDC) is as shown in Figure 1. For a set point change the closed loop response is given by

$$\ddot{Y}(z) = \frac{HG_P(z)D(z)}{1 + HG_P(z)} \ddot{Y}_{SP}(z) \qquad (1)$$

Where $HG_P(z)$ =Pulse transfer function of the process with zero order hold.

In DDC for a given step change in the set point the discrete-time response should be specified. Then $HG_P(z)$, $\ddot{Y}(z)$ and $\ddot{Y}_{sp}(z)$ are known we can solve equation (1) with respect to unknown D (z).

The transfer function of the digital controller is given by

$$D(z) = \frac{1}{HG_P(z)} * \frac{Y(z)/Y_{SP}(z)}{1 - Y(z)/Y_{SP}(z)} \qquad (2)$$

PROBLEM

Design a Dahlin's controller for the process.

$$G_P(s) = \frac{1}{(0.5s+1)(s+1)(s+1)(2s+1)}$$ with Sampling Time T=5 seconds (3)

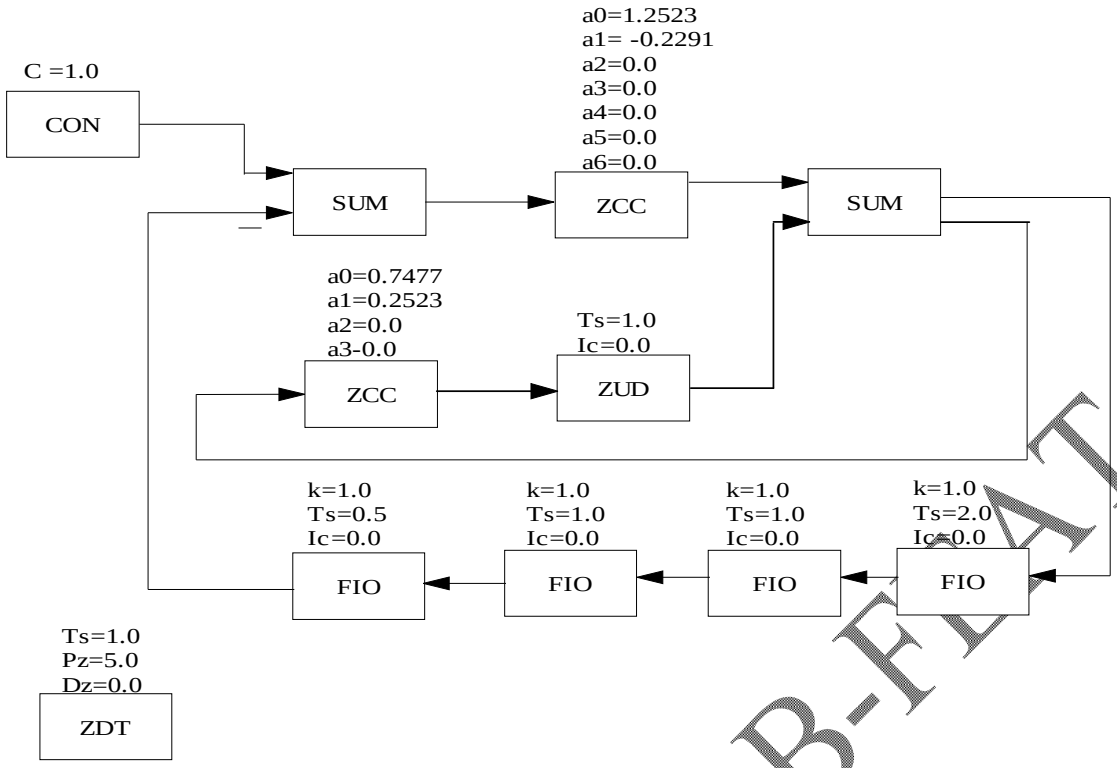


Figure 4. TUTSIM Block Diagram of Dahlin's Controller for actual process

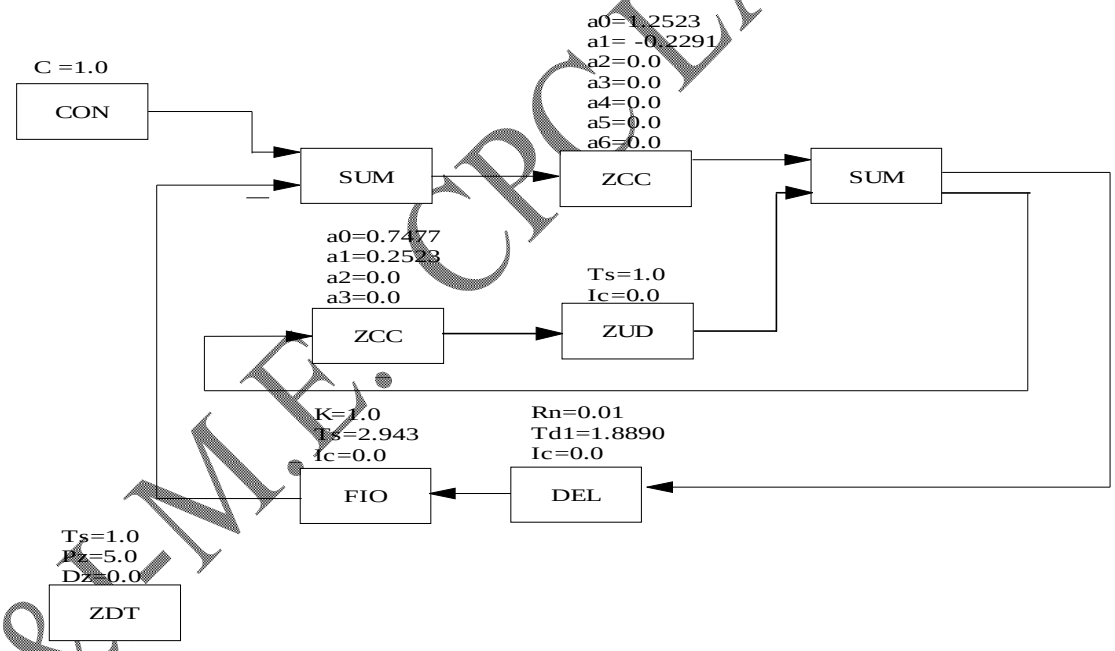


Figure 5. TUTSIM Block Diagram of Dahlin's Controller for approximated process

DESIGN

First approximate the given process into first order plus dead time model using process reaction curve method. The block diagram schematic is as shown in Figure 2 and the response obtained is as shown in Figure 3.

The approximated first order time delay model is obtained from Figure 3 and is given by

$$G_P(s) = \frac{1\, e^{-1.889s}}{(2.943s + 1)} \tag{7}$$

The pulse transfer function is given by

$$HG_P(z) = Z[H(s)\, G_P(s)]$$

where H(s) = zero order hold transfer function and is given by

$$H(s) = \left(\frac{1 - e^{-sT}}{s}\right)$$

$$HG_P(z) = Z\left[\left(\frac{1 - e^{-sT}}{s}\right)\left(\frac{1 * e^{-1.889s}}{(2.943s + 1)}\right)\right]$$

$$= (1 - Z^{-1}) * Z\left[\frac{e^{-1.889s}}{S * (2.943S + 1)}\right]$$

Sampling time T = 5 seconds

$$\lambda T = t_d$$

$$\lambda T = 1.889$$

$$\lambda = \frac{1.889}{5} = 0.3778$$

$$m = 1 - \lambda$$

$$m = 1 - 0.3778 = 0.6222$$

$$\therefore HG_P(z) = (1 - z^{-1}) * Zm\left[\frac{1}{s(2.943s+1)}\right]_{m=0.6222}$$

$$= (1 - z^{-1}) * Zm\left[\frac{1/2.943}{s(s + 1/2.943)}\right]_{m=0.6222}$$

$$= (1 - z^{-1}) * Zm\left[\frac{0.3398}{s(s + 0.3398)}\right]_{m=0.6222,\, a=0.3398}$$

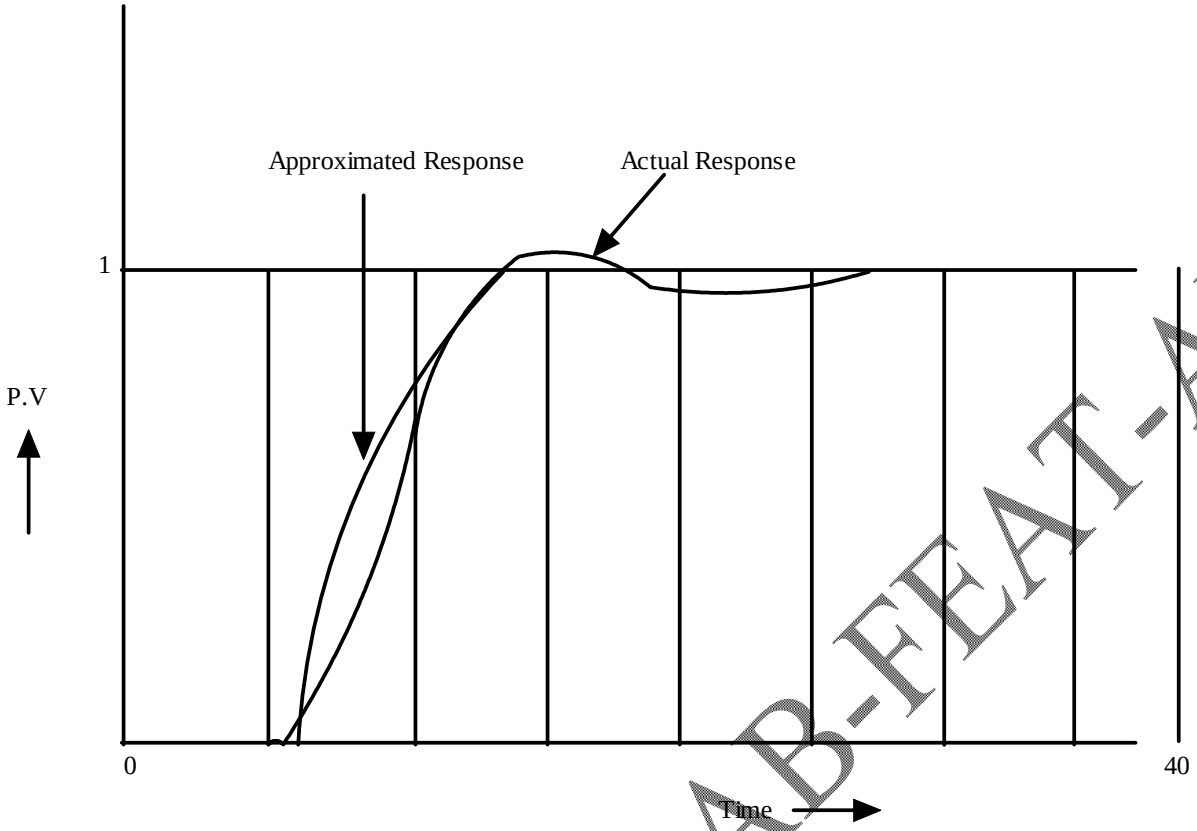


Figure 6. Closed loop Model Response of DAHLIN'S Controller for Actual process and Approximated process

$$HG_p(z) = (1 - z^{-1}) * Zm\left[\frac{0.3398}{s(s + 0.3398)}\right]_{m=0.6222, a=0.3398}$$

We Know

$$Zm\left[\frac{a}{S(S + a)}\right] = \left[\frac{1}{z - 1} - \frac{e^{-amT}}{z - e^{-aT}}\right]$$

$$HG_p(z) = (1 - z^{-1}) * Zm\left[\frac{0.3398}{s(s + 0.3398)}\right]_{m=0.6222, a=0.3398}$$

$$= (1 - z^{-1})\left[\frac{1}{z - 1} - \frac{e^{-(0.3398)(0.6222)(5)}}{z - e^{-(0.3398)(5)}}\right]$$

$$= (1 - z^{-1})\left[\frac{1}{z - 1} - \frac{0.3475}{z - 0.1829}\right]$$

$$= \frac{z - 1}{z}\left[\frac{1}{z - 1} - \frac{0.3475}{z - 0.1829}\right]$$

$$= \frac{1}{z} - \frac{(0.3475z - 0.3475)}{(z^2 - 0.1829z)}$$

$$= \frac{z^2 - 0.1829z - 0.3475z + 0.3475z}{z^3 - 0.1829z^2}$$

$$= \frac{0.6525z^2 + 0.1646z}{z^3 - 0.1829z^2}$$

$$= \frac{z(0.6525z + 0.1646)}{z(z^2 - 0.1829z)}$$

$$HG_p(z) = \frac{0.6525z + 0.1646}{z^2 - 0.1829z}$$

$$= \frac{z(0.6525 + 0.1646z^{-1})}{z^2(1 - 0.1829z^{-1})}$$

$$= \frac{0.6525 + 0.1646z^{-1}}{z - 0.1829}$$

$$= \frac{z^{-1}(0.6525 + 0.1646z^{-1})}{(1 - 0.1829z^{-1})}$$

$$D(z) = \frac{1}{HG_p(z)} * \frac{(1-\alpha)\, z^{-(N+1)}}{(1-\alpha\ z^{-1}) - (1-\alpha)z^{-(N+1)}} \quad where\ \alpha = e^{-(T/\tau)}$$

$$\alpha = e^{-(5/2.943)} = 0.1829$$

$N = td/T = 0$ (N is the integral multiples of deadtime)

$$D(z) = \frac{(z - 0.1829)}{(0.6525 + 0.1646\ z^{-1})}\ \frac{(1 - 0.1829\ )\, z^{-(0+1)}}{(1 - 0.1829\ z^{-1} - 0.8171\ z^{-1})}$$

$$D(z) = \frac{(z - 0.1829)}{(0.6525 + 0.1646\ z^{-1})}\left[\frac{0.8171\ z^{-1}}{1 - z^{-1}}\right]$$

$$\frac{m(z)}{e(z)} = \frac{(0.8171 - 0.1495\ z^{-1})}{(0.6525 - 0.4879\ z^{-1} - 0.1646\ z^{-2})}$$

Cross multiplying,

$$0.6525\ m(z) - 0.4879\ z^{-1} m(z) - 0.1646\ z^{-2} m(z) = 0.8171\ e(z) - 0.1495\ e(z)\ z$$

Taking inverse Z-transform

$$m_n = 0.7477 m_{n-1} + 0.2523 m_{n-2} + 1.2523 e_n - 0.2291 e_{n-1}$$

EXPERIMENTAL PROCEDURE

(i) Give the step input of magnitude 'u' for the given higher order process as shown in Figure 2.

(ii) Obtain the process reaction (S-shaped) curve as shown in Figure 3.

(iii) From the process reaction curve find the steady state value (B), t_1 and t_2 where t_1=time corresponds to 28.3% of B and t_2= time corresponds to 63.2% of B.

(iv) Approximate the first order plus dead time model using the given formulae as $G_p(s) = \frac{K_p e^{t_d s}}{\tau s + 1}$ where K_P is the process gain, τ is the process time constant and t_d is the process dead time.

$$K = \frac{\text{Change in steady state value}}{\text{change in the input}} = \frac{B}{U}$$

$$\tau = 1.5 * (t_2 - t_1) \qquad (8)$$

$$t_d = t_2 - \tau$$

(v) Derive pulse transfer function $HG_P(z)$.

(vi) Obtain the digital controller D(z) as given in the design.

(vii) Implement the transfer function of the dahlin's controller by using Z-blocks in TUTSIM for actual and approximated process as shown in Figure 4 and Figure 5 respectively.

(viii) Analyze the response and comment on your results.

INFERENCE

From the closed loop response it is observed that the dahlin's response is similar to a First Order response with dead time. (Without any Oscillations)

RESULT

The closed loop response of a sampled data system to a unit step change in the set point has been obtained after designing a Dahlin's controller.

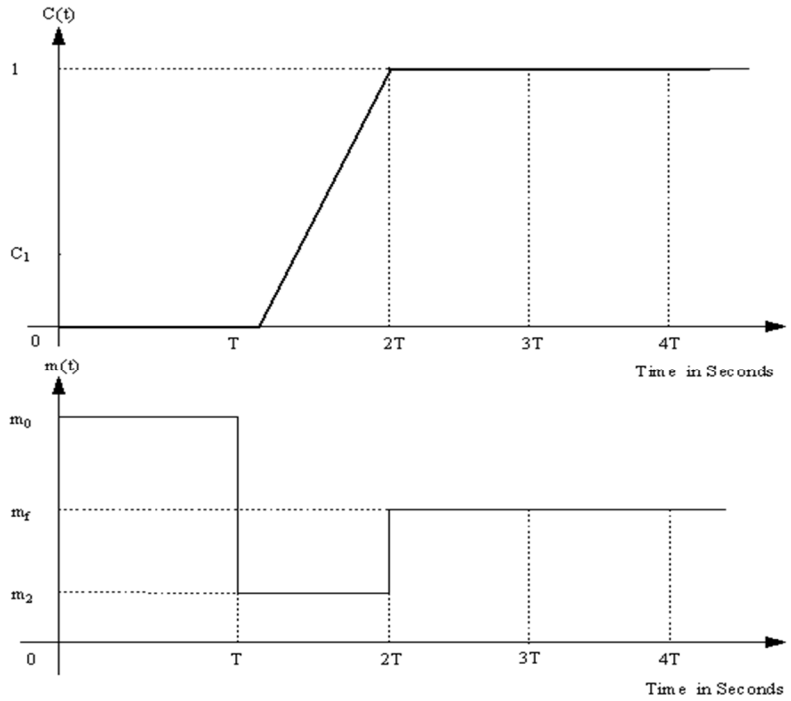


Figure 1. Desired response characteristics used to design Kalman's algorithm

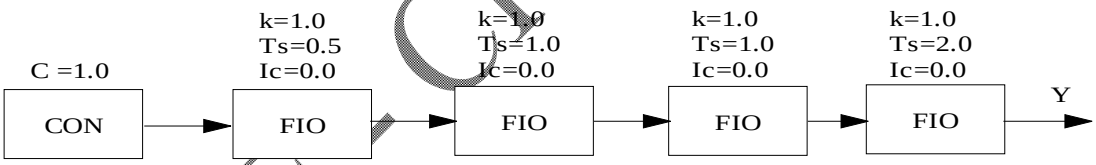


Figure 2. Block Diagram to obtain Open Loop Response

EX.NO:	DESIGN AND SIMULATION OF KALMAN'S CONTROLLER USING TUTSIM
DATE:	

AIM

To design a Kalman's controller for a given process and to find the closed loop response of the sampled data system to a unit step change in the set point incorporating Kalman's algorithm.

THEORY OF KALMAN CONTROLLER

To synthesis a digital control algorithm by Kalman's approach, we place restrictions on C and M, instead of usual C/R. Thus the derivation of the Kalman's algorithm assumes a specific C (z). That is the system response to a unit step input reaches the final value in two sampling time instants and remains at the final value thereafter as shown in Figure 1. The expression for C (z) is

$$C(z) = c_1 z^{-1} + z^{-2} + z^{-3} + \ldots \quad (1)$$

In order to accomplish this manipulated variable will assume two intermediate values and then assumes its final value thereafter as shown in Figure 1.

$$M(z) = m_0 + m_1 z^{-1} + m_f z^{-2} + m_f z^{-3} + \ldots \quad (2)$$

where no restrictions need be placed on the value of c_1 and m_f equals the reciprocal of the process steady state gain. The number of intermediate values of M equals the order of the process.

If the control algorithm is to be used for a unit step change in set point, then

$$R(z) = \frac{1}{1 - z^{-1}}$$

$$\frac{C(z)}{R(z)} = (1 - z^{-1})(c_1 z^{-1} + z^{-2} + z^{-3} + \ldots)$$

$$= C_1 z^{-1} + (1 - C_1) z^{-2} \quad (3)$$

$$= P_1 z^{-1} + P_2 z^{-2}$$

$$= P(z)$$

and $$\frac{M(z)}{R(z)} = (1 - z^{-1})(m_0 + m_1 z^{-1} + m_f z^{-2} + m_f z^{-3} + \ldots.)$$

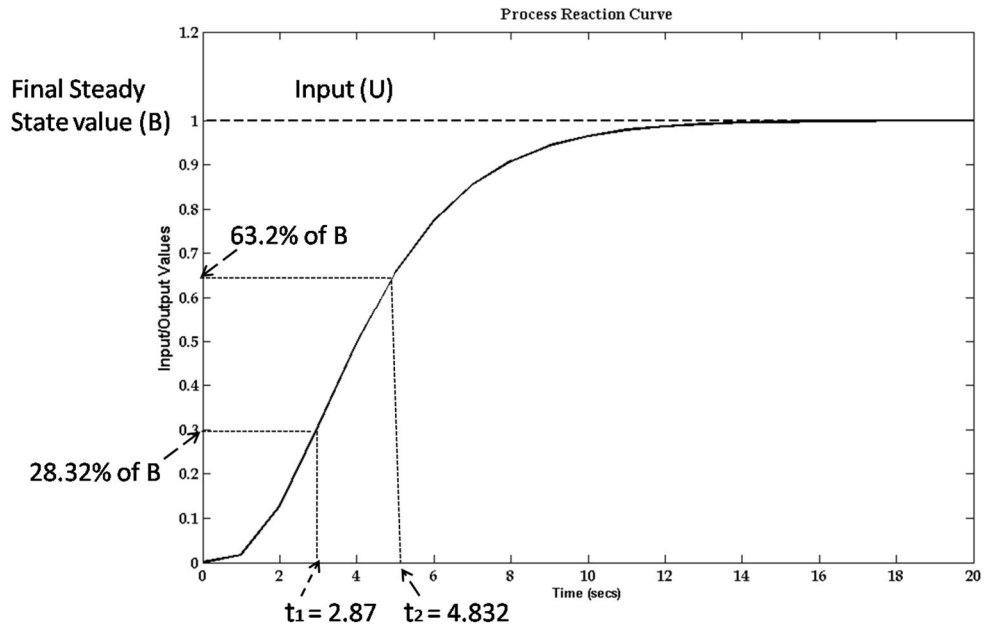


Figure 3. Open Loop Step Response of the higher order process for the step input of magnitude 'u'

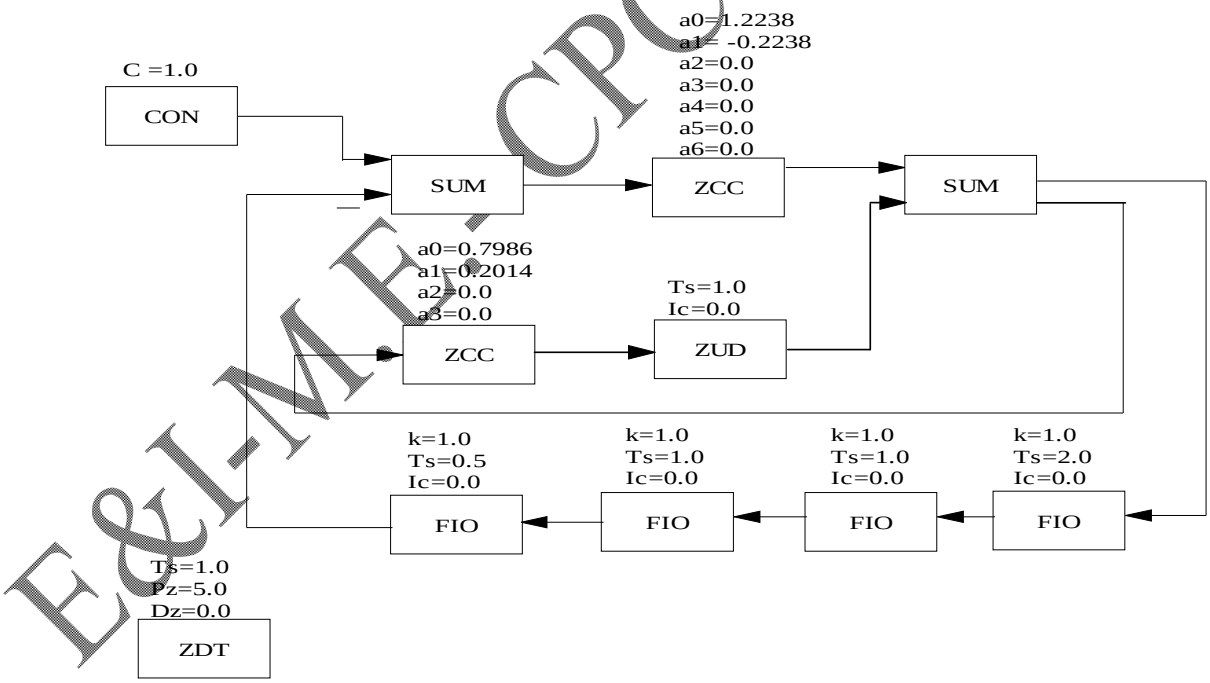


Figure 4. TUTSIM Block Diagram of Kalman's Controller for Actual Process

$$= m_0 + (m_1 - m_0)z^{-1} + (m_f - m_1)z^{-2}$$
$$= q_0 + q_1 z^{-1} + q_2 z^{-2}$$
$$= Q(z)$$

$$HG_P(z) = \frac{C(z)}{M(z)} = \frac{P(z)}{Q(z)} \quad (4)$$

Thus the coefficient in H $G_P(z)$ must equal those in P(z) and Q(z)

$$\sum_{i=1}^{2} P_i = P_1 + P_2 = 1 \text{ and} \quad (5)$$

$$\sum_{i=0}^{2} q_i = q_0 + q_1 + q_2 = \frac{1}{K_P} \quad (6)$$

These relationships do not generally hold good for the pulse transfer functions, but dividing by sum of the numerator coefficients will ensure that both equations (5) and (6) hold good. Since P(z) and Q(z) are known, the control algorithm D(z) can be derived as

$$D(z) = \frac{\frac{C(z)}{R(z)}}{1 - \left[\frac{C(z)}{R(z)}\right]} \frac{1}{HG_P(z)}$$

$$D(z) = \frac{P(z)}{1 - P(z)} \frac{Q(z)}{P(z)} = \frac{Q(z)}{1 - P(z)} \quad (7)$$

The equation holds good even if the process has a dead time of N sampling instants. In such case the specification of C(z) must include the appropriate number of zeros.

PROBLEM

Design a Kalman's controller for the process given by

$$G_P(s) = \frac{1}{(0.5s+1)(s+1)(s+1)(2s+1)} \quad (8)$$

Sampling Time T=5 second.

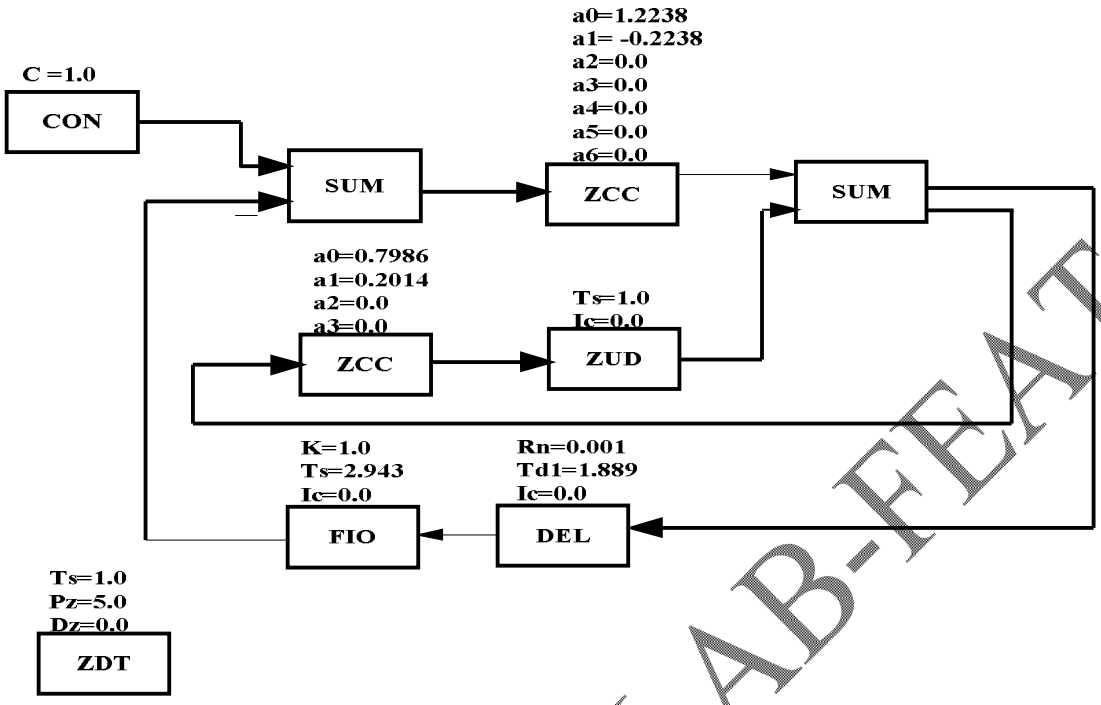


Figure 5. TUTSIM Block Diagram of Kalman's Controller for Approximated Process

DESIGN

First approximate the given process into first order plus dead time model using process reaction curve method. The block diagram schematic is as shown in Figure 2 and the open loop response obtained is shown in Figure 3.

The approximated first order plus time delay model is obtained from Figure 3 and is given by

$$G_P(s) = \frac{1e^{-1.889s}}{(2.943s+1)}$$

The pulse transfer function is given by

$$HG_P(z) = Z[H(s)G_P(s)]$$

$$\text{where } H(s) = \left(\frac{1-e^{-sT}}{s}\right) = \text{zero order hold transfer function}$$

$$HG_P(z) = Z\left[\left(\frac{1-e^{-sT}}{s}\right)\left(\frac{1*e^{-1.889s}}{(2.943s+1)}\right)\right]$$

$$HG_P(z) = Z[H(s)\,G_P(s)]$$

$$\text{where } H(s) = \left(\frac{1-e^{-sT}}{s}\right) = \text{zero order hold transfer function}$$

$$HG_P(z) = Z\left[\left(\frac{1-e^{-sT}}{s}\right)\left(\frac{1*e^{-1.889s}}{(2.943s+1)}\right)\right]$$

$$= (1-z^{-1})*Z\left[\frac{e^{-1.889s}}{s*(2.943s+1)}\right]$$

Sampling time T = 5 seconds

$$\lambda T = t_d$$

$$\lambda T = 1.889$$

$$\lambda = \frac{1.889}{5} = 0.3778$$

$$m = 1-\lambda$$

$$m = 1-0.292 = 0.6222$$

$$\therefore HG_P(z) = (1-z^{-1})*Zm\left[\frac{1}{s(2.943s+1)}\right]_{m=0.6222}$$

$$= (1 - z^{-1}) * Zm\left[\frac{1/2.943}{s(s + 1/2.943)}\right]_{m=0.6222}$$

$$= (1 - z^{-1}) * Zm\left[\frac{0.3398}{s(s + 0.3398)}\right]_{m=0.6222, a=0.3398}$$

$$HG_P(z) = (1 - z^{-1}) * Zm\left[\frac{0.3398}{s(s + 0.3398)}\right]_{m=0.6222, a=0.3398}$$

We Know

$$Zm\left[\frac{a}{s(s+a)}\right] = \left[\frac{1}{z-1} - \frac{e^{-amT}}{z - e^{-aT}}\right]$$

$$HG_P(z) = (1 - z^{-1}) * Zm\left[\frac{0.3398}{s(s + 0.3398)}\right]_{m=0.6222, a=0.3398}$$

$$= (1 - z^{-1})\left[\frac{1}{z-1} - \frac{e^{-(0.3398)(0.6222)(5)}}{Z - e^{-(0.3398)(5)}}\right]$$

$$= (1 - z^{-1})\left[\frac{1}{z-1} - \frac{0.3475}{z - 0.18293}\right]$$

$$= \frac{z-1}{z}\left[\frac{1}{z-1} - \frac{0.3475}{z - 0.1829}\right]$$

$$= \frac{1}{z} - \frac{(0.3475z - 0.3475)}{(z^2 - 0.1829z)}$$

$$= \frac{z^2 - 0.1829z - 0.3475z^2 + 0.3475z}{z^3 - 0.1829z^2}$$

$$= \frac{0.6525z^2 + 0.1646z}{z^3 - 0.1829z^2}$$

$$= \frac{z(0.6525z + 0.1646)}{z(z^2 - 0.1829z)}$$

$$HG_P(z) = \frac{0.6525z + 0.1646}{z^2 - 0.1829z}$$

$$= \frac{z(0.6525 + 0.1646z^{-1})}{z^2(1 - 0.1829z^{-1})}$$

$$= \frac{z^{-1}(0.6525 + 0.1646z^{-1})}{(1 - 0.1829z^{-1})}$$

NORMALISATION

For normalization add the numerator co-efficients then divide both numerators and denominator by the added numerator co-efficient value.

Added numerator co-efficient value = 0.6525+0.1646=0.8171

$$G(z)=\frac{P(z)}{Q(z)}=\frac{\frac{0.6525}{0.8171}+\frac{0.1646}{0.8171}z^{-1}}{\frac{z}{0.8171}-\frac{0.1829}{0.8171}}$$

$$\frac{P(z)}{Q(z)}=\frac{0.7986+0.2014z^{-1}}{1.2238z-0.2238}$$

$$=\frac{0.7986z^{-1}+0.2014z^{-2}}{1.2238-0.2238z^{-1}}$$

$$D(z) \quad =\frac{Q(z)}{1-P(z)}$$

$$D(z) \quad =\frac{1.2238-0.2238z^{-1}}{1-\left(0.7986z^{-1}+0.2014z^{-2}\right)}$$

$$D(z) \quad =\frac{\left(1.2238-0.2238z^{-1}\right)}{\left(1-0.7986z^{-1}-0.2014z^{-2}\right)}$$

$$\frac{m(z)}{e(z)} \quad =\frac{\left(1.2238-0.2238z^{-1}\right)}{\left(1-0.7986z^{-1}-0.2014z^{-2}\right)}$$

$$m(z)-0.7986z^{-1}m(z)-0.2014z^{-2}m(z)=1.2238e(z)-0.2238e(z)z^{-1}$$

Taking inverse Z-transform

$$m_n = 0.7986\, m_{n-1} + 0.2014\, m_{n-2} + 1.2238\, e_n - 0.2238\, e_{n-1}$$

EXPERIMENTAL PROCEDURE

i. Give the step input for the given higher order process as shown in Figure 2.
ii. Obtain the process reaction (S- shaped) curve as shown in Figure 3.
iii. From the process reaction curve find the steady state value(B), t_1 and t_2 where t_1=time corresponds to 28.32% of B and t_2= time corresponds to 63.2% of B.
iv. Approximate the first order plus dead time model using the given formulae as $G_P(s)=\frac{K_P e^{-tds}}{\tau s+1}$ where K_p is the process gain, τ is the process time constant and t_d is the process dead time.

$$K=\frac{\text{Change in steady state value}}{\text{change in the input}}=\frac{B}{U}$$

$$\tau=1.5*\left(t_2-t_1\right) \qquad (9)$$

$$t_d=t_2-\tau$$

v. Derive pulse transfer function $HG_P(z)$.
vi. Obtain the digital controller D(z) as given in the design.
vii. Implement the transfer function of the dead beat controller by using Z-blocks in TUTSIM for actual and approximated process as shown in Figure 5 and Figure 6 respectively.
viii. Analyse the response and comment on your results.

RESULT

The closed loop response of a sampled data system to a unit step change in the set point has been obtained after designing a Kalman's controller.

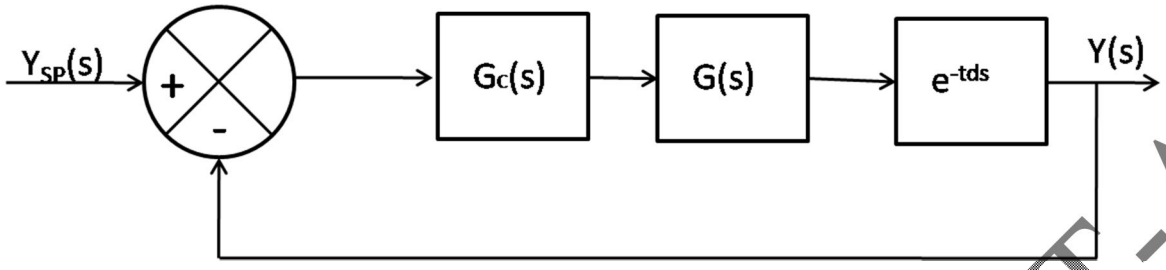


Figure 1(a). Block Diagram of Closed Loop System

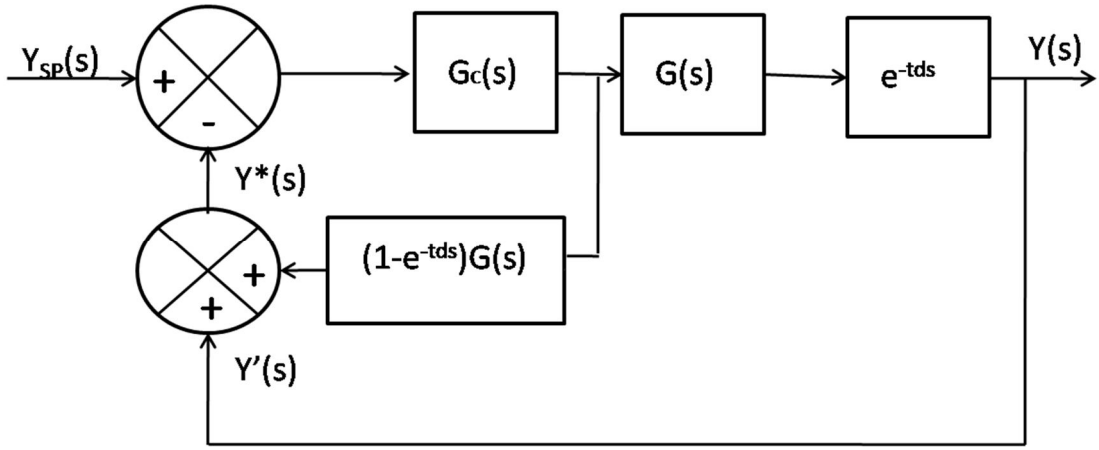


Figure 1(b). Block Diagram of System with Dead time and Compensation Scheme

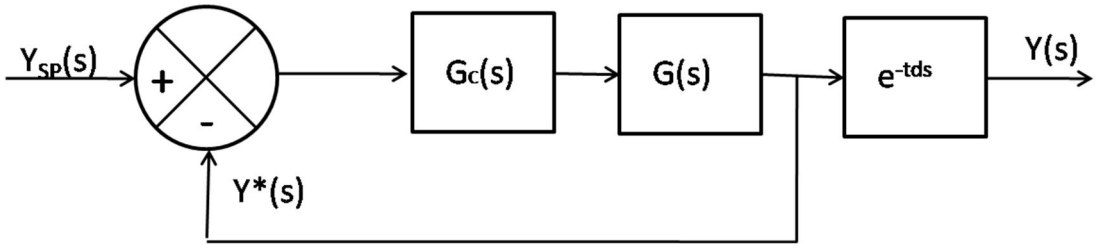


Figure 1(c). Block Diagram of System with Dead time Compensation

EX.NO:	DEAD TIME COMPENSATION USING SMITH PREDICTOR ALGORITHM SIMULATION USING SIMULINK
DATE:	

AIM

1. To design a dead-time compensator using Smith Predictor algorithm.
2. To obtain the closed loop response of the given system (with large dead time) with and without compensator.

THEORY

A conventional feedback controller would provide unsatisfactory closed loop response for a system having large dead-time, for the following reasons:

1. A disturbance entering the process will not be detected until after a significant period of time.
2. The control action that will be taken on the basis of the last measurement will be inadequate because it attempts to regulate a situation that originated before.
3. The control action will also take some time to make its effect felt by the process.

As a result of all the factors noted above, considerable dead time is a significant source of instability. Consider a simple feedback loop shown in Figure. 1(a) with the process transfer function,

$$G_p(s) = G(s)e^{-tds}$$

The open loop response to a change in set point for the process is given by

$$Y(s) = G_c(s) \{G(s)\, e^{-tds}\}\, Y_{sp}(s)$$

Here the response is delayed by t_d minutes.

In order to eliminate the undesired effects caused by dead time, the open loop feedback signal should carry the current information and not the delayed information such as

$$Y^*(s) = G_c G(s)\, Y_{sp}(s)$$

This is made possible, if Y'(s) is added with Y(s) as shown in Figure 1(b) and Y'(s) is given by,

$$Y'(s) = \{(1-e^{-tds})\, G(s)\}\, G_c Y_{sp}(s).$$

Now,

$$Y^*(s) = Y(s) + Y'(s)$$

$$= [(1-e^{-tds})\, G(s)\}\, G_c Y_{sp}(s)] + G(s)G_c(s)\, e^{-tds}$$

$$= G(s)G_c(s)Ysp(s) - e^{-tds}G(s)G_c(s)Ysp(s) + e^{-tds}\, G(s)G_c(s)Ysp(s)$$

Therefore, $Y^*(s) = G_c G(s)\, Y_{sp}(s)$. Here the feedback signal contains only the current information as shown in Figure 1(c).

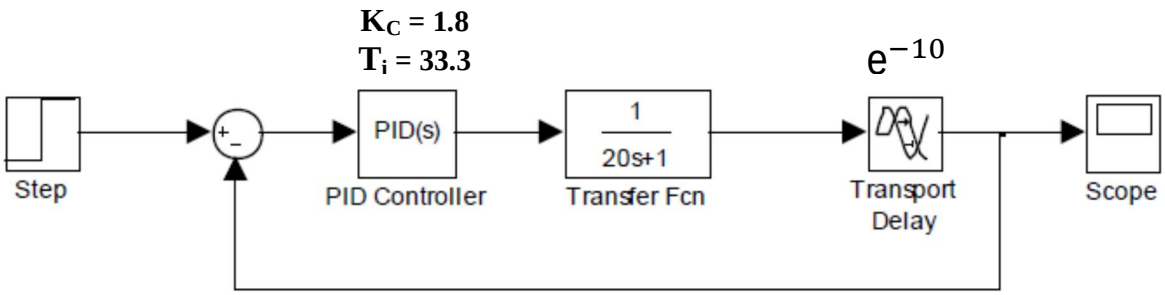


Figure 2. Block Diagram of propesses without compensator

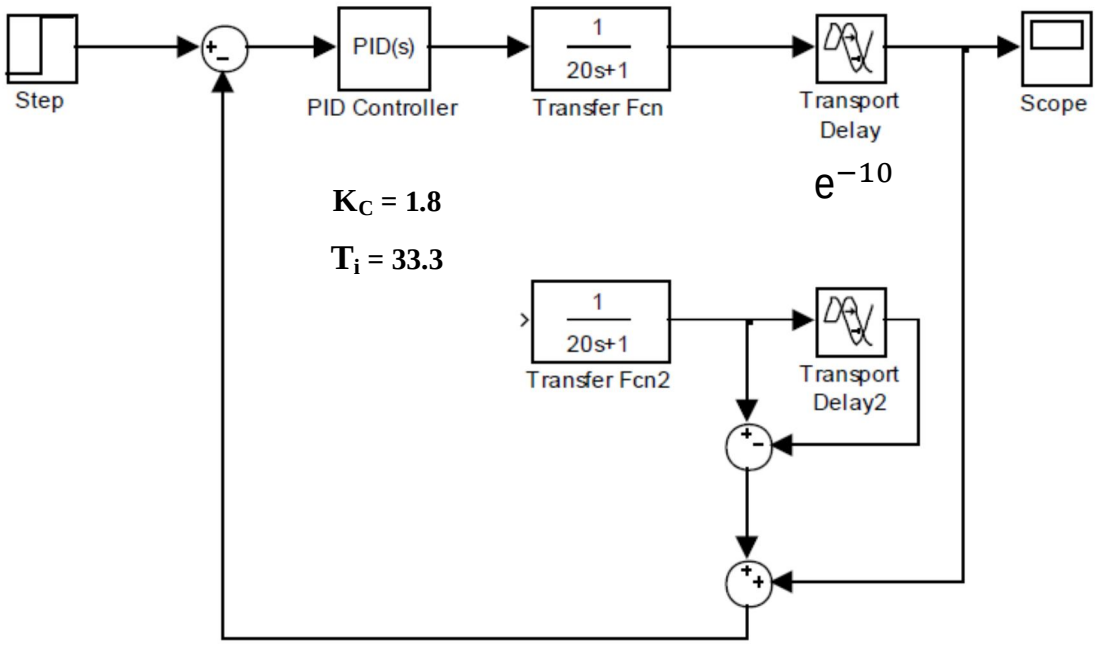


Figure 3. Block Diagram of propesses with compensator

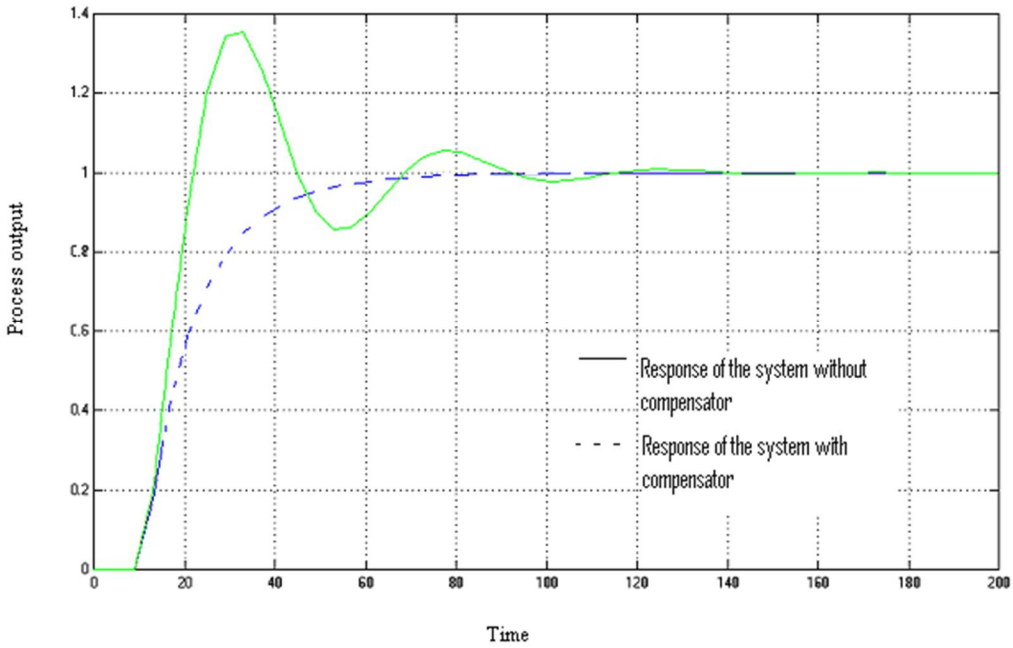


Figure 4. Response of the process with and without dead time compensator

The dead time compensator predicts the delayed effect that the manipulated variable will have on the process output. This prediction led to the term Smith predictor. Perfect compensation is possible only if the process model is perfectly known.

Dead time compensation is needed only when the ratio of dead time to time constant is greater than one. The controller should be conventional PID or PI controller.

EXAMPLE

Demonstrate the effect of dead time compensation for the process given by,

$G(s) = \{ e^{-10s}/(1/20s+1)\}$

Compare the response with and without dead time compensator.

PROCEDURE

1. For the given process $G(s) = \{ e^{-10s}/(1/20s+1)\}$ check the closed loop response by randomly setting different PI controller values.
2. Obtain the closed loop response of the given process with and without dead time compensator and analyze the robustness. (Refer Figure 2 and 3 respectively) using simulink software.
3. Analyze both the servo and regulatory responses with and without dead time compensator and comment on your result.

$$K_c = \frac{0.9\tau}{K_p t_d} = \frac{0.9 * 20}{1 * 10} = 1.8$$

$$Ti = 3.33 t_d = 33.3$$

RESULT

A dead time compensator using smith predictor algorithm was designed and the process was simulated using Simulink and the closed loop response incorporating dead time compensator and without dead time compensator was compared.

TABLE 1. SYTEMS WITH INVERSE RESPONSE

1. Pure capacitive minus first-order response

$$G(s) = \frac{K_2}{S} - \frac{K_1}{\tau_1 S + 1} = \frac{(K_2\tau_1 - K_1)S + K_2}{S(\tau_1 S + 1)}$$

For $K_2\tau_1 < K_1$ Zero = $-K_2/(K_2\tau_1 - K_1) > 0$

2. Difference between two first-order responses

$$G(s) = \frac{K_1}{\tau_1 S + 1} - \frac{K_2}{\tau_2 S + 1} = \frac{(K_1\tau_2 - K_2\tau_1)S + (K_1 - K_2)}{(\tau_1 S + 1)(\tau_2 S + 1)}$$

For $\frac{\tau_1}{\tau_2} > \frac{K_1}{K_2} > 1$ Zero = $-(K_1 - K_2)/(K_1\tau_2 - K_2\tau_1) > 0$

3. Difference between two first-order responses with dead time

$$G(s) = \frac{K_1 e^{-\tau_1 S}}{\tau_1 S+1} - \frac{K_2 e^{-\tau_2 S}}{\tau_2 S+1}$$

For $K_2 < K_1$ and $\tau_1 < \tau_2 \geq 0$

4. Second order-minus first order responses

$$G(s) = \frac{K_1}{\tau^2 S^2 + 2\delta\tau S + 1} - \frac{K_2}{\tau_2 S + 1}$$

For $K_1 > K_2$ and $t_1 > t_2 \geq 0$

5. Difference between two second-order responses

$$G(S) = \frac{K_1}{{\tau_1}^2 S^2 + 2\delta_1\tau_1 S + 1} - \frac{K_2}{{\tau_2}^2 S^2 + 2\delta_2\tau_2 S + 1}$$

For $\frac{{\tau_1}^2}{{\tau_2}^2} > \frac{K_1}{K_2} > 1$

6. Differences between two second-order responses with dead time

$$G(S) = \frac{K_1 e^{-\tau_1 S}}{{\tau_1}^2 S^2 + 2\delta_1\tau_1 S + 1} - \frac{K_2 e^{-\tau_2 S}}{{\tau_2}^2 S^2 + 2\delta_2\tau_2 S + 1}$$

For $K_1 > K_2$ and $t_1 > t_2 \geq 0$

EX.NO:	DESIGN AND TESTING OF INVERSE RESPONSE COMPENSATOR USING SIMULINK

AIM

To design and implement a inverse response compensator for a given process using Simulink.

THEORY

The dynamic behavior of certain processes moves in an opposite direction initially to where it eventually ends up for a step change at its input. Such behavior is called inverse response or non minimum phase response. Table.1 shows several such opposing effects between first order (or) second order systems. In all the cases, the system possesses an inverse response as its transfer function has a positive zero. Systems with inverse response are particularly difficult to control and require special attention.

INVERSE RESPONSE COMPENSATOR

There are two very popular ways to control systems with inverse response.

1. PID Controller with Ziegler Nichol's tuning.
2. Inverse response compensator.

1. SIMPLE PID CONTROL

From all types of feedback controllers only PID can be used effectively because of the simple reason the derivative control mode by its nature will anticipate the "wrong" direction of the system's response and will provide the proper corrective action to limit (never eliminate) the inverse response. The general schematic diagram is shown in Figure.1.

INVERSE RESPONSE COMPENSATOR:

A Smith Predictor (dead time compensator) cancels the effect of dead time. The same general concept of the predictor (compensator) can be used to cope with the inverse response of a process. To eliminate the inverse response it is enough to eliminate the positive zero of the above open-loop transfer function.

DESIGN AN INVERSE RESPONSE FROM TWO FIRST ORDER SYSTEMS

$$G(s) = \frac{K_1}{\tau_1 S+1} - \frac{K_2}{\tau_2 S+1}$$

Figure1 shows another possibility of inverse response. Two opposing effects result from two different first-order processes, yielding an overall response equal to

$$\bar{Y}(s) = \left(\frac{K_1}{\tau_1 S+1} - \frac{K_2}{\tau_2 S+1}\right)\bar{f}(s)$$

$$\bar{Y}(s) = \left(\frac{(K_1\tau_2 - K_2\tau_1)S+(K_1-K_2)}{(\tau_1 S+1)(\tau_2 S+1)}\right)\bar{f}(s)$$

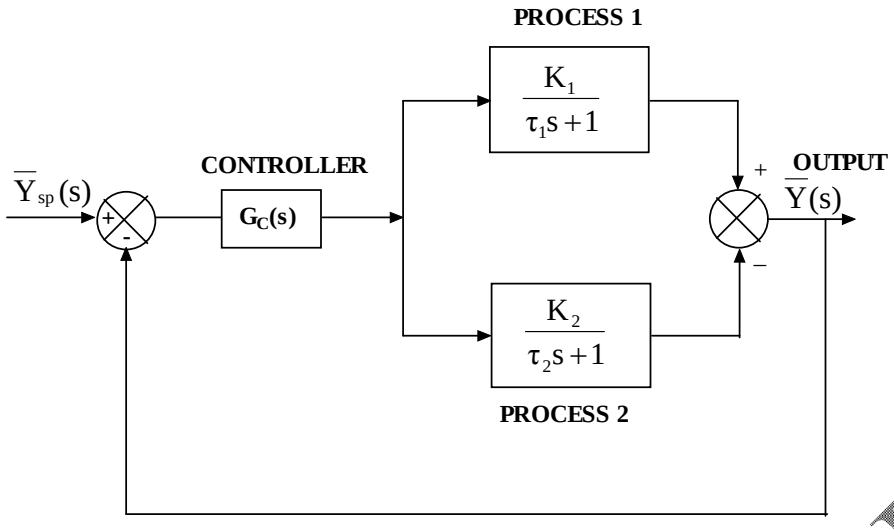


FIGURE 1. CONVENTIONAL FEEDBACK CONTROLLER OF PROCESS WITH INVERSE RESPONSE

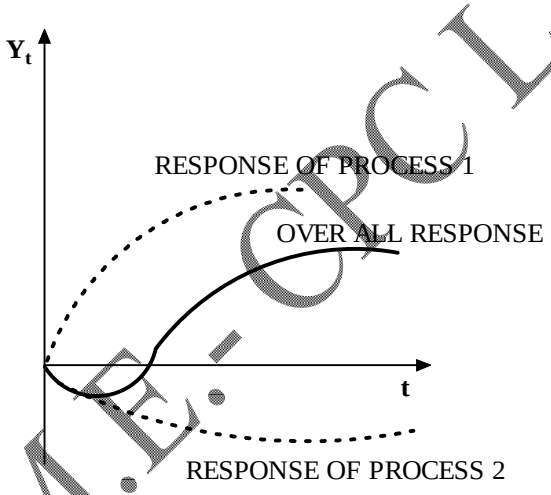


FIGURE 2. INVERSE RESPONSE

We have inverse response when initially (at t =0+) process 2 which reacts faster than process 1 ($K_2 / \tau_2 > K_1 / \tau_1$) dominates the response of the overall system.

But ultimately process 1 reaches a higher steady-state value than process 2 ($K_2 < K_1$) and forces the response of the overall system in the opposite direction.

Figure 3 shows the inverse response of the overall system.

Consider the feedback system of Figure 3. The controlled process exhibits inverse response when,

$$\frac{\tau_1}{\tau_2} > \frac{K_1}{K_2} > 1$$

The open-loop response of the system is

$$\bar{Y}(s) = G_C(s)\left(\frac{(K_1\tau_2 - K_2\tau_1)S + (K_1 - K_2)}{(\tau_1 S+1)(\tau_2 S+1)}\right)\bar{Y}_{sp}(s)$$

It has a positive zero at the point,

$$Z = \frac{(K_1 - K_2)}{(K_1\tau_2 - K_2\tau_1)} > 0$$

To eliminate the inverse response it is enough to eliminate the positive zero of the above open-loop transfer function. This is possible if in the open loop response $\bar{Y}(s)$ we add the quantity $\bar{Y}'(s)$ given by,

$$\bar{Y}'(s) = G_C(s)K\left(\frac{1}{\tau_1 S+1} - \frac{1}{\tau_2 S+1}\right)\bar{Y}_{sp}(s)$$

$$\bar{Y}^*(s) = \bar{Y}(s) + \bar{Y}'(s)$$

$$\bar{Y}^*(s) = G_C(s)\left(\frac{(K_1\tau_2 - K_2\tau_1) + K(\tau_1 - \tau_2)S + (K_1 - K_2)}{(\tau_1 S + 1)(\tau_2 S + 1)}\right)\bar{Y}_{sp}(s)$$

$$K \geq \frac{(K_2\tau_1 - K_1\tau_2)}{(\tau_1 - \tau_2)}$$

We find that the zero of the resulting open loop transfer function is non positive.

$$Z = -\frac{K_1 - K_2}{(K_1\tau_2 - K_2\tau_1) + K(\tau_1 - \tau_2)} \leq 0$$

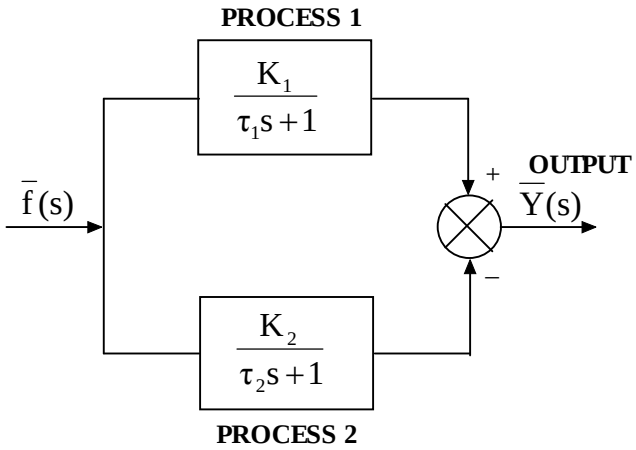


FIGURE 3. BLOCK DIAGRAM OF TWO OPPOSING FIRST - ORDER SYSTEMS

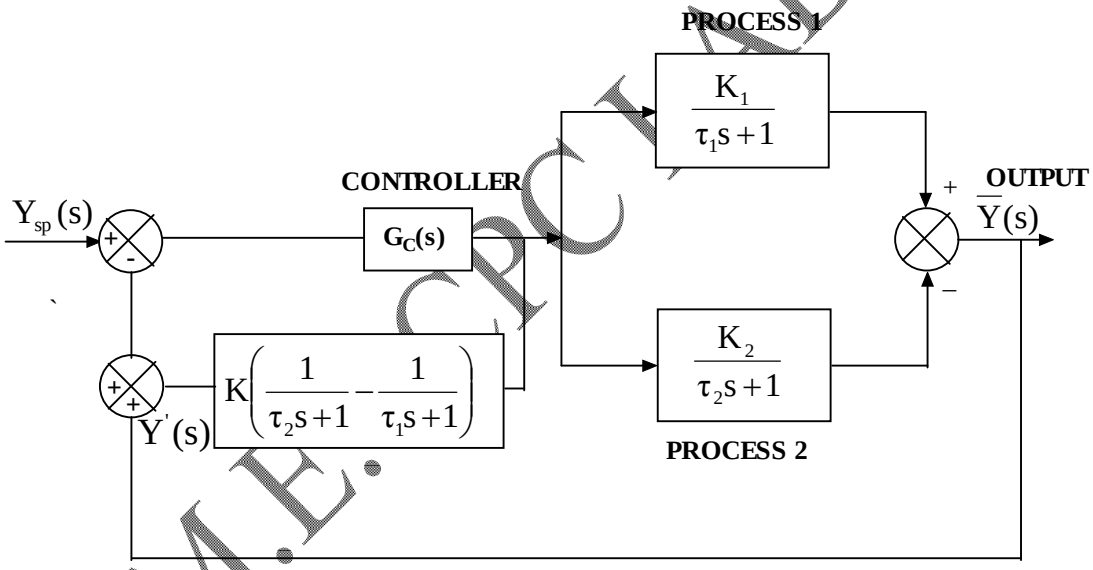


FIGURE 4. FEEDBACK CONTROL OF PROCESS WITH INVERSE RESPONSE WITH COMPENSATOR

Adding the signal $\bar{Y}'(s)$to the main feedback signal $\bar{Y}(s)$means the creation of the local loop around the controller as shown in Figure 4. The system in this local loop is the modified Smith Predictor and the actual compensator of the inverse response.

$$G_{Compensator} = K\left(\frac{1}{\tau_1 S+1} - \frac{1}{\tau_2 S+1}\right)$$

PROBLEM

Design an inverse response for the given transfer function.

$$G(s) = \frac{12}{2S+1} - \frac{9}{S+1}$$

PROCEDURE

1. Give the step input for the higher order process. Vary the gain until the sustained oscillation is produced as shown in Figure 5.
2. From the process response find the ultimate period pu and ultimate gain ku.
3. Find the PID controller settings from (Z-N closed loop tuning rules)

$$Kc = 0.6 * ku$$

$$T_i = \frac{pu}{2}$$

$$T_d = \frac{pu}{8}$$

1. Implement the Inverse response compensator as shown in Figure 5 and the response as shown in Figure 6.

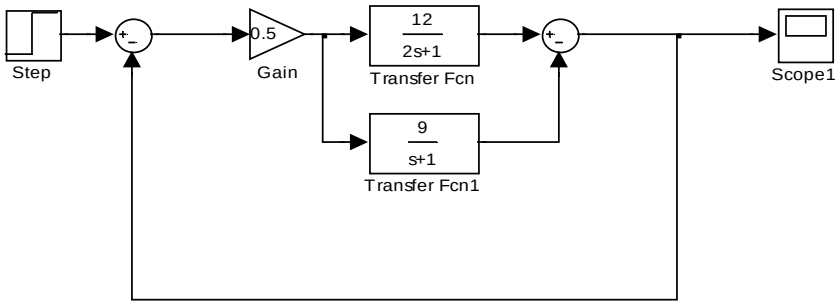


Figure 1. BLOCK DIAGRAM FOR CLOSED LOOP RESPONSE

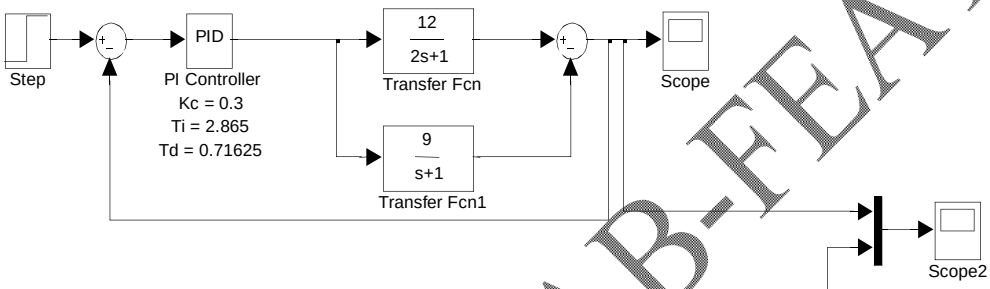


FIGURE 1. BLOCK DIAGRAM OF CONVENTIONAL FEEDBACK CONTROL OF PROCESS WITH INVERSE RESPONSE

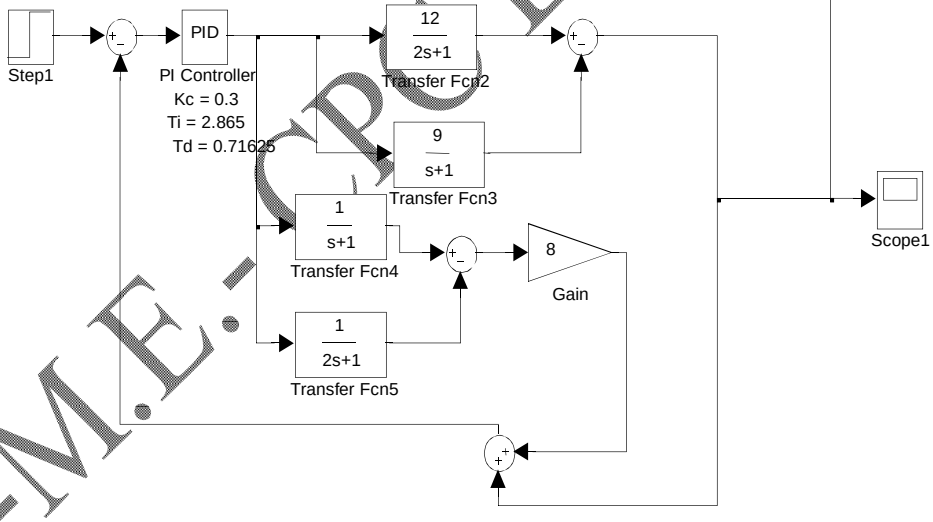


FIGURE 5. BLOCK DIAGRAM OF FEED BACK CONTROL OF PROCESS WITH INVERSE RESPONSE COMPENSATOR

CALCULATION

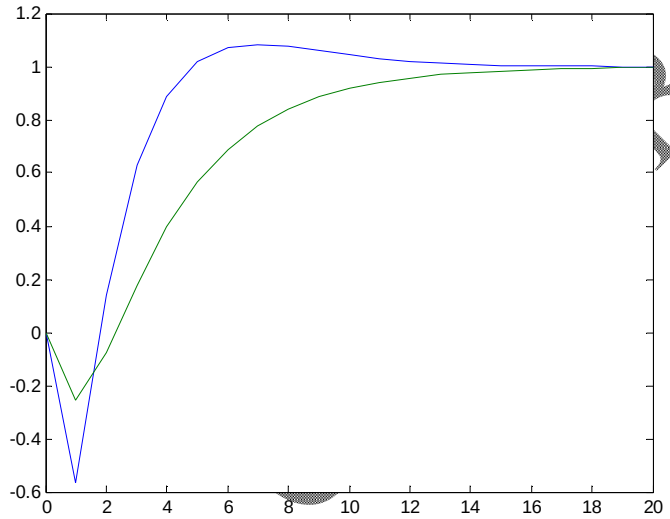
PID settings:

Ku = 0.5; Pu = 5.73

Kc = 0.6 * ku
Kc = 0.6 * 0.5 = 0.3

T_i = pu/2
T_i = 5.73/2 = 2.865
T_d =pu/8
T_d =5.73/8 = 0.71625

FIGURE.6 PROCESS WITH INVERSE RESPONSE /WITH INVERSE COMPENSATOR



RESULT

The inverse response compensator of a given transfer function has been obtained.

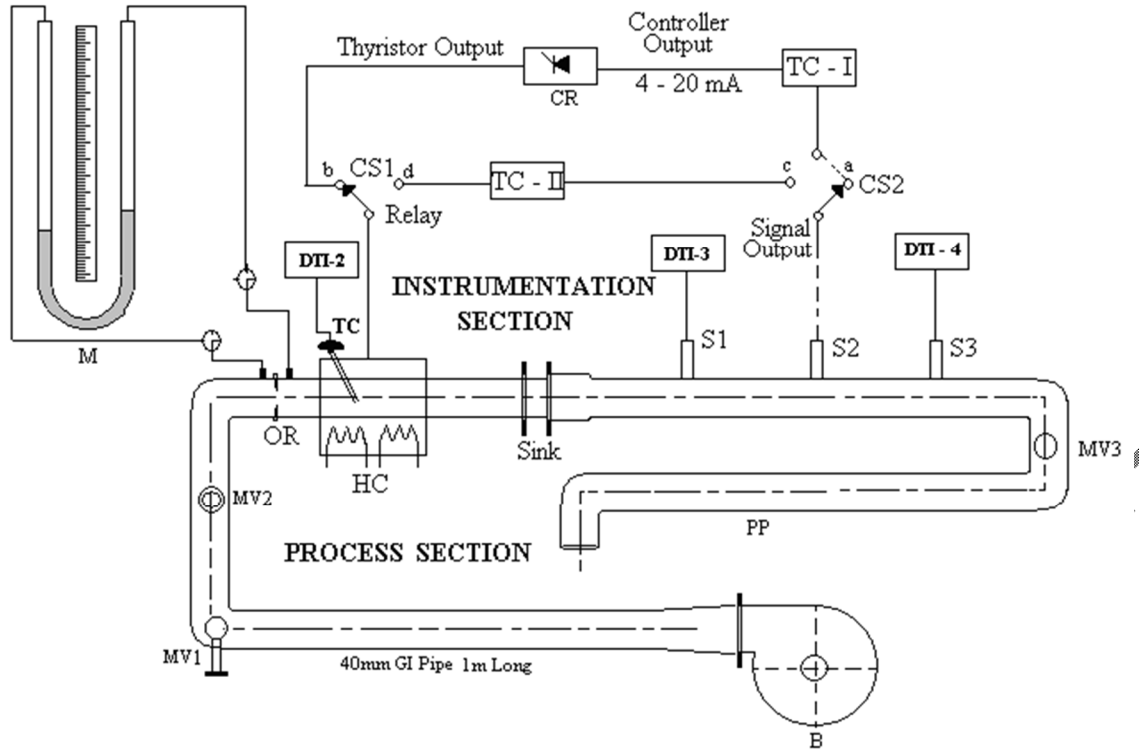


Figure 1. Piping and Instrumentation Diagram of Air Temperature process.

REFERENCE CODE	DESCRIPTION OF INSTRUMENTS
TC – 1	ELECTRONIC PID CONTOLLER
TC – 2	ON – OFF CONTROLLER
CR	THYRISTOR CONTROLLED RECTIFIER UNIT
CS1 & CS2	CHANGE OVER RELAYS
DTI 1,2,3,4	DIGITAL TEMPERATURE INDICATOR
M	MANOMETER
REFERNCE CODE	**DESCRIPTION OF THE PROCESS**
B	BLOWER
MV	MANUAL CONTROL VALVE
HC	HEATING CHAMBER
PP	POLY PROPYLENE TUBE
S1,S2 & S3	RTD SENSOR PROBES
OR	ORIFICE
CS2	THROUGH BANANA TERMINALS
a-b	HEATING CONNRCTION THROUGH ON – OFF
c-d	HEATER CONNECTION THROUGH PID
TC	THERMOCOUPLE

EX.NO:	AIR TEMPERATURE CONTROL SYSTEM
DATE	

AIM:

To obtain the closed loop response of the Air Temperature Process with P and PI controller.

PROCESS DESCRIPTION

The Piping and Instrumentation Diagram(P & ID) of a closedloop air temperature system is shown in Fig.1. It consists of two sections as

(i) Process section
(ii) Instrumentation section

The process bench consist of the following accessories like blower, cold air circulation line with manual control valve and bypass valve, orifice plate, heating chamber with thermo couple and process tube with RTD's.

The blower is mounted at the inlet of the cold air circulation line. The blower discharges air into the pipeline which is made of GI tube and the manual control valve (MV1) is provided to control the air flow to the process tube. The manual control bypass valve (MV2) is provided to let the air flow out. The function of MV2 is to protect the blower against damage due to back pressure.

The orifice mounted in the cold air circulation line enables to measure flow rate of the air in the process tube. The outlet of the orifice is connected to the U – tube of the glass manometer which indicates the differential pressure across the orifice.

The heating chamber is connected just after the orifice and temperature of the heating chamber is measured with a thermocouple. The process tube is fitted with three numbers of RTD's (S1, S2 and S3).

The instrumentation panel consists of the following accessories like PID controller, ON/OFF controller, temperature indicators, U – tube manometer, main power supply switch, main circuit board, auxiliary switches for individual instruments and heater, pair of banana terminals for input and output signals of instruments and sensors .

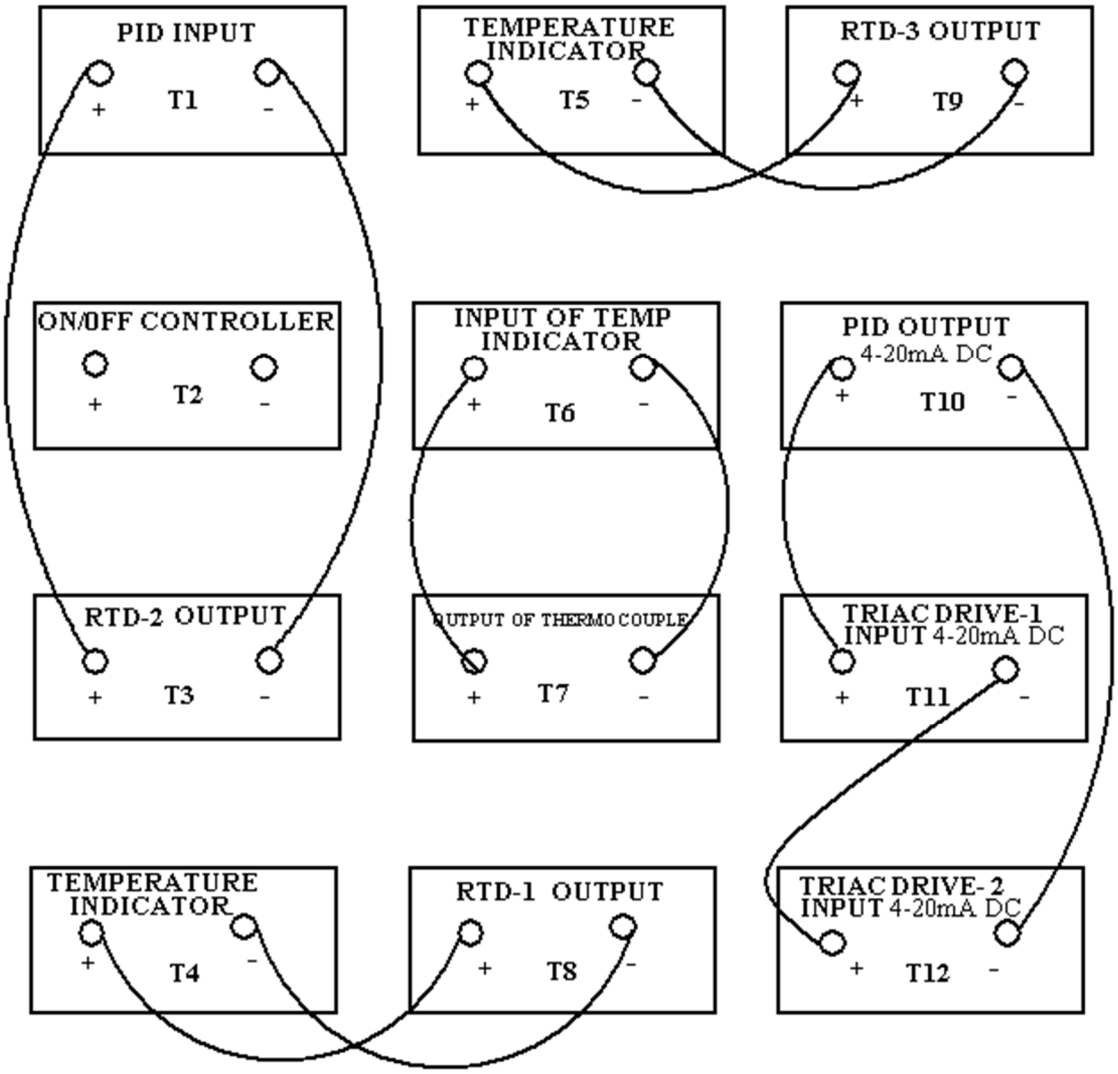


Figure 2. Closed loop connection diagram with PID controller

Table I. Controller settings

Mode	K_c (°C/mA)	T_i(sec)	T_d(sec)
P	1.137	-	-
PI	1.024	209.79	-
PID	1.365	126	31.5

PRECAUTIONS

The following precautions should be taken care before turning ON the process

1. Manual Valve (MV1) should be half open. MV2 and MV3 should be fully open.
2. PID controller should be in manual mode
3. During start up sequence, first the blower should be turned ON and then the heater should be ON.
4. To turn OFF the system, first the heater should be turned OFF and then the blower should be turned OFF.

CALCULATION OF PID CONTROLLER SETTINGS

The general form of a first order process with dead time is given by

$$G_p = \frac{k_p e^{-t_d s}}{\tau s + 1}$$

where k_P – process gain

τ – time constant of the process

t_d – time delay of the process

The mathematical model for Air temperature process is approximated to

$$G_P(s) = \frac{7.91}{567s + 1} e^{-63s}$$

The P, PI, PID controller settings for air temperature process using the Zeigler Nicholas tuning rules are given below

For Proportional mode (P)

$$K_c = \frac{\tau}{K_p t_d} \qquad (1)$$

For Proportional plus Integral mode(PI)

$$K_c = 0.9 * \frac{\tau}{K_p t_d}$$

Table2. Controlled Variable values for P mode

S.P = _____ %PB =_____ T_i = _____ and T_d = _____

S.No	Time (sec)	Temperature °C

Table3. Controlled Variable values for PI mode

S.P = _____ %PB =_____ T_i = _____ and T_d = _____

S.No	Time (sec)	Temperature°C

$$T_i = 3.33t_d \quad (3)$$

For Proportional plus Integral plus derivative mode (PID)

$$K_c = 1.2^* \frac{\tau}{K_p t_d} \quad (4)$$

$$T_i = 2^* t_d \quad (5)$$

$$T_d = 0.5^* t_d \quad (6)$$

Using the equations 1 through 6, the calculated P, PI and PID settings for the air temperature process are tabulated below in Table 1.

CLOSED LOOP CONTROL OF AIR TEMPERATURE PROCESS

The process medium used here is air. The temperature of the air is to be controlled and is allowed to flow through a polypropylene tube. The air is heated by the heating chamber.

The temperature of the air is sensed by a suitable sensor (RTD-2) whose output (4-20mA) is given to the PID controller. The controller compares this measured variable with its set point value and the error produced is manipulated through PID circuitry as controller's output (4-20mA).This signal is given as input to the triac driver circuit.

Triac driver in turn supplies variable AC voltage in the range of 0 – 230 V to the heater which is proportional to controller output. The final control element is the triac driver. The input to the heater controls the chamber temperature which in turn maintains the desired temperature of the air media to be controlled .

In this experiment, P and PI controller modes are implemented and K_c and T_i can be returned to the desired values so as to achieve the desired closed loop performance.

EXPERIMENTAL PROCEDURE

1. Connect the main power card to the incoming AC supply of 230 V/5A to enable power supply to the equipment which will be indicated by the neon light.

2. Give the connections as per the closed-loop connection diagram with PID controller (P/PI mode) termination panel as shown in Fig.2.

3. Select the control mode for PID control and accordingly connect T-1 (PID – input) to T-3 (RTD-2 output). Also connect the PID controller output T-10 in series with T-11 and T–12 as shown in Fig.2.

4. Switch ON the blower so as to enable the air flow to the process tube. Once the air flow stabilizes (which can be observed in the manometer) one can proceed with the experiment.

5. Switch ON the temperature indicators.

6. Switch ON the PID controller (Ensure its in manual mode).

7. Set the Set point value and %PB and other values from Table1.

8. Switch ON heater -1 and put the voltmeter knob in heater -1 position. Then change the controller from Manual mode to Auto Mode.

9. Indication on the voltmeter shows that power is transmitted to the heater.

10. Tabulate the controlled variable values for P and PI mode in Table.2 and able.3 respectively.

11. Plot the response of the process for various controller modes. Model graph is shown in Fig.3.

12. Calculate the performance measures from the observed data for P and PI mode and tabulate the results in Table 4.

RESULT

The closed loop response of the process with P and PI modes were studied for the air temperature process and the performance measure was tabulated.

EXERCISE

1. What is manipulated variable, process variable and controlled variable?
2. What is the difference between two wire and four wire RTD's?

Front panel

Block diagram

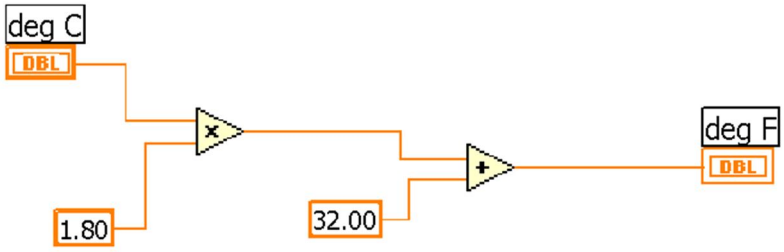


Figure 1. To convert temperature value in Celsius to Fahrenheit

EX.NO:	STUDY OF LABVIEW SOFTWARE
DATE:	

AIM

To familiarize the Lab VIEW software and to develop suitable blocks using LabVIEW for the given exercises.

INTRODUCTION TO LABVIEW:

Lab VIEW is a graphical programming environment. A program written in Lab VIEW is called Virtual Instrument (VI). Each VI must have a Front Panel and a Block Diagram. The Front Panel includes various controls and indicators, while the block diagram typically includes functions, terminals and sub VI's.

An object in the front panel has its counterpart called a terminal in the block diagram. All controls and indicators have their corresponding terminals in the block diagram. It is important to label the terminal to identify them.

The toolbar serves as the user interface, providing tools for editing and running the VI. It includes the text settings, move text menus as well as aligning and distributing objects and tools for debugging.

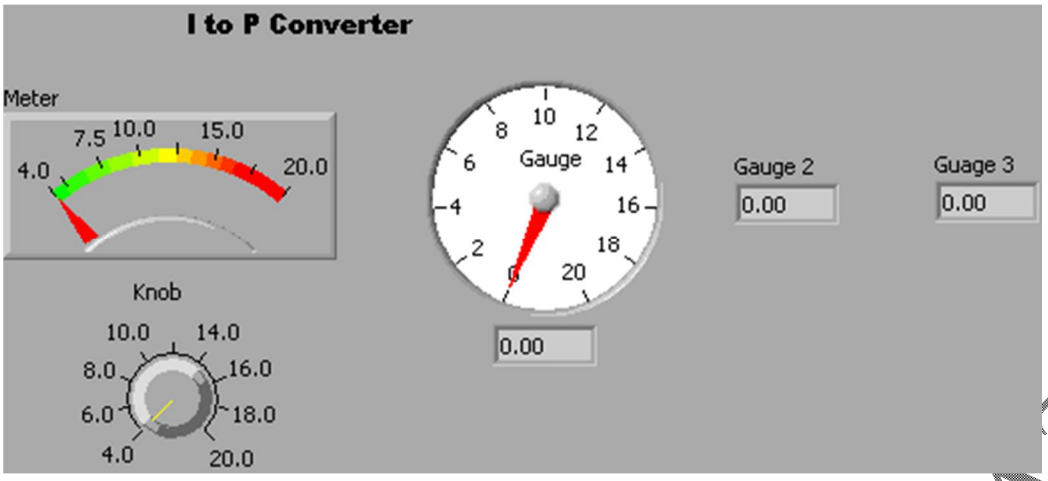
The menu bar includes options for file control, editing, operating a VI, window control and many options. The help options provide object description as well as in depth online help.

It is easy to troubleshoot and modify individual modules without affecting others. The Edit icon in the front panel is used to edit a VI. The icon editor has tools for drawing and entering text in color or black and white. Wires link objects in the block diagram and the wires carry data. The thickness and color of wire represent different data types. Numerical data is represented as floating point numbers, signed and unsigned integers or complex numbers. Format allows the user to express the numbers using floating point notation, scientific notation or engineering notation.

Trouble shooting a VI is necessary when the VI does not run because it has either syntax errors or run time errors. Single stepping, break points and probes help the trouble shooting task.

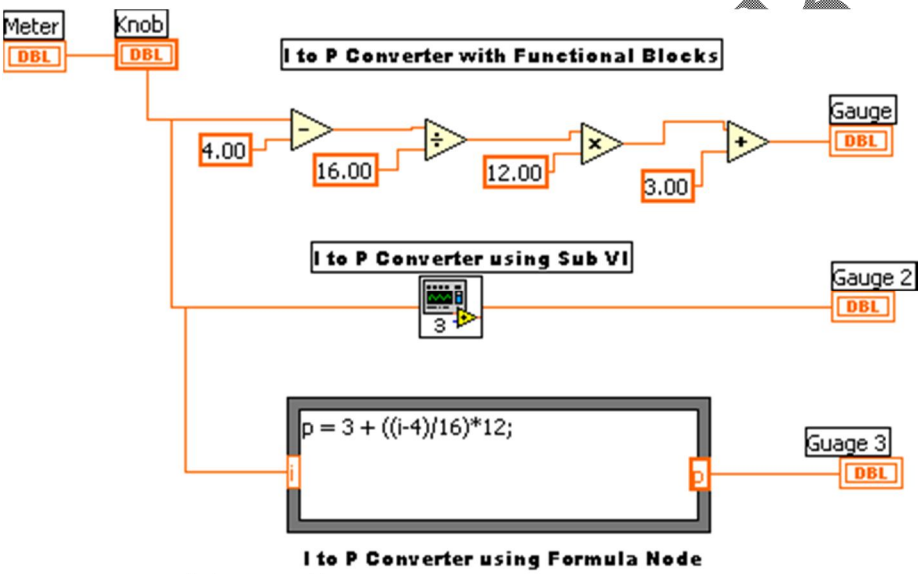
The timers are useful in providing a time delay or marking time. The 'tick count' timer may be used to time the operation, the 'wait' timer is useful in generating accurate time delays and the 'wait until next ms multiple' timer can also be used to provide time delays or time between loop iteration delays.

Front Panel



(A)

Block Diagram



(B)

Figure 2. To convert current signal (4 to 20mA) to pressure signal (3 to 15psi)
(A) Front panel (B) Block Diagram

An array is a collection of objects such as numbers, square LEDs, Boolean switches or other objects. In a one dimensional array objects are placed along a straight line. A two dimensional array is made up of rows and columns. In Lab VIEW an array can be made as an array control or array indicator.

A string is a collection of ASCII characters. Strings are used by Lab VIEW for text messages, instrument control and for storing data to disk. A string control is used to pass string data to the block diagram. A string indicator is used to display a string generated by the block diagram.

Waveform chart is used to display one or more waveforms. The data to be displayed on the waveform graph must be in the form of an array. The X-Y graph is to plot mathematical functions or curves using the Cartesian co-ordinates. The special tools in the charts and graphs are used to configure and operate them.

The text as well as data that the created VI generates can be saved to a file on the disk. In Lab VIEW special procedures must be followed to write text or data to a file on the disk or to retrieve information from the disk.

EXERCISE-1

To convert a temperature value in Celsius to Fahrenheit.

SOLUTION

F = (C * 1.8) + 32

Multiply the Celsius value by 1.8 and add 32.
Use the functions on the functions>> numeric palette to build a block diagram as shown in Figure 1.

STEP 1

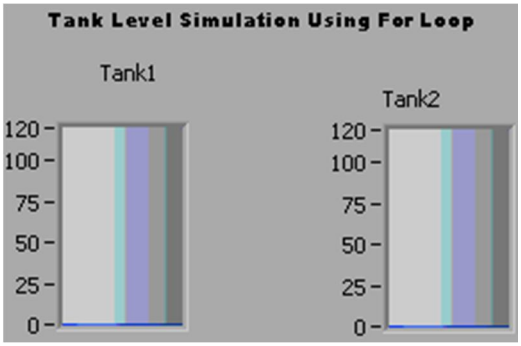
Select a 'multiply' function from the numeric palette and drag it to the block diagram. Select a 'wiring tool' from tools palette and place it over multiply block so the terminals are highlighted. Right click the mouse on one terminal of 'multiply' block and select 'create control'. A control block is created in front panel, so that we can vary the Celsius temperature from the front panel itself. On another terminal of 'multiply' block, right click the mouse and select 'create constant' and enter the value 1.8.

STEP 2

Select 'add' block from numeric palette and drag it to block diagram. Output of 'multiply' block is connected to one terminal of 'add' block and to another terminal, right click mouse and select 'create constant' and enter the value 32.

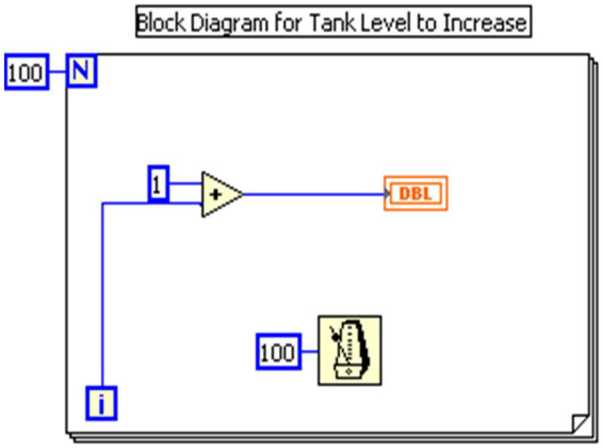
On output terminal of 'add' block 'create indicator'. So, Fahrenheit temperature value is indicated in the front panel.

Front Panel

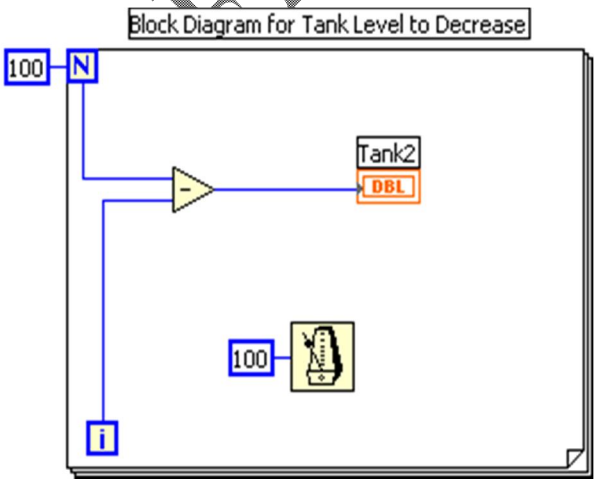


(A)

Block Diagram



(B)



(C)

Figure 3. Filling up the tank

(A) Front panel (B) Block Diagram to fill the tank up to 100 from 0 feet

(C) Block Diagram to drain the tank from 100 to 0 feet

EXERCISE-2

To convert current (I) in the range 4 to 20mA to pressure signal (P) in the range 3 to 15 psi.

SOLUTION

P = 3+ ((I – 4) / 16) * 12

Select ‘add’, ‘multiply’ and ‘divide’ blocks from numeric palette and configure as per exercise 1. Refer Figure 2.

Another method to solve this exercise is by using formula node.
Select functions >> structure palette.
Drag formula node to block diagram. Select ‘connect wire tool’ and on the frame of formula node right click mouse and select ‘add input’, ‘add output’. Declare input as I and output as P. Create ‘control’ and ‘indicator’. Pressure value will be displayed in the indicator.

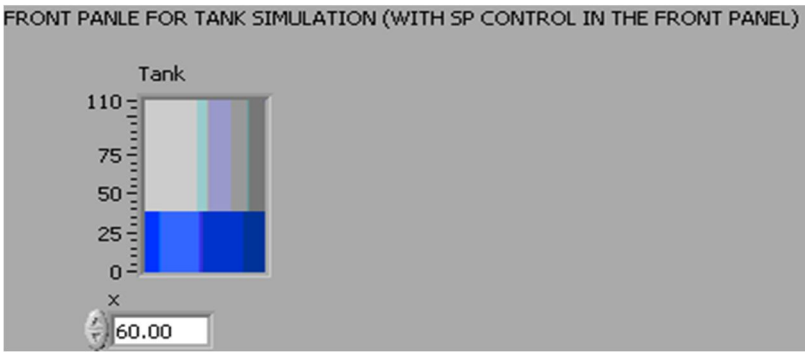
EXERCISE-3

a) To fill the tank level up to 100 from 0 feet.
b) To drain the tank level from 100 to 0 feet

a) Select functions >> structure palette.
   Drag ‘for’ loop to block diagram. Give 100 as number of iterations. On the ‘for’ loop add 1 to iteration, connect output of ‘add’ block to a tank. Tank will fill upto 100 feet. Time can be varied by using timer block. Refer Figure 3(a).

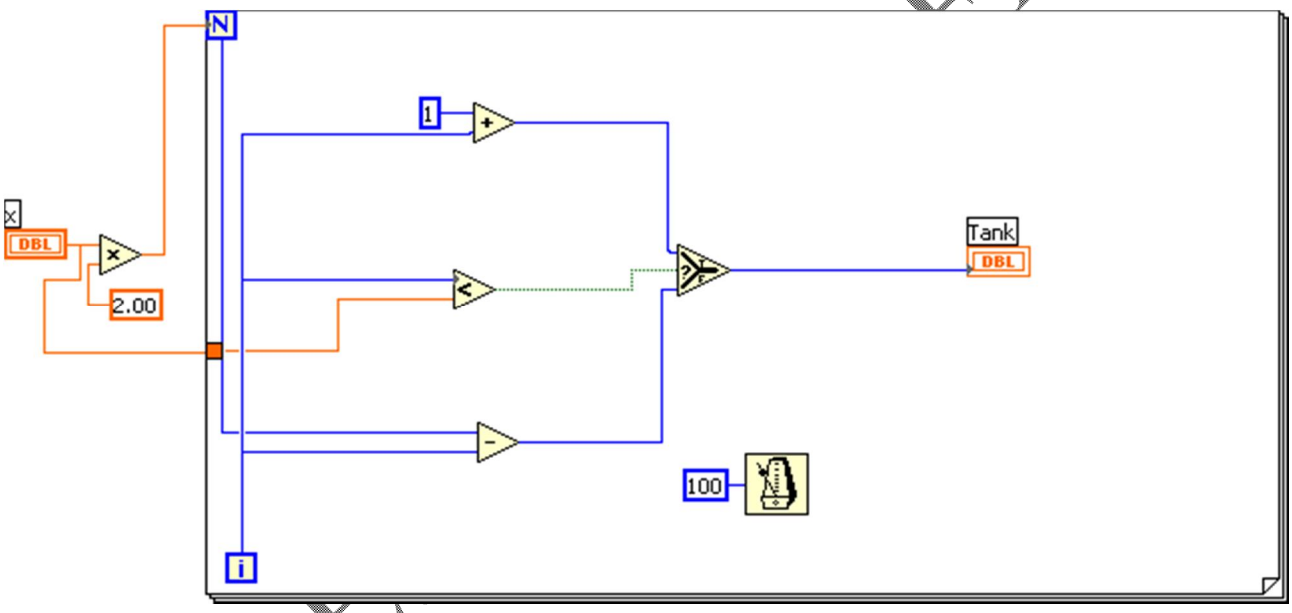
b) Select functions >> structure palette.
   Drag ‘for’ loop to block diagram. Give 100 as number of iterations. On the ‘for’ loop subtract number of iterations from every iteration. Connect output of subtract block to a tank. Tank will drain from 100 feet. Time can be varied by using timer block. Refer Figure 3(b).

Front Panel



(A)

Block Diagram



(B)

Figure 4. To increase the tank level to a desired set point and to decrease the level once it reaches the set point.
(A) Front panel (B) Block Diagram

EXERCISE-4

To increase the tank level to a desired set point and to decrease the level once it reaches the set point.

Configure blocks as shown in Figure 4 in similar way as in Exercises 1 to 3.

RESULT

The LabVIEW software was studied.

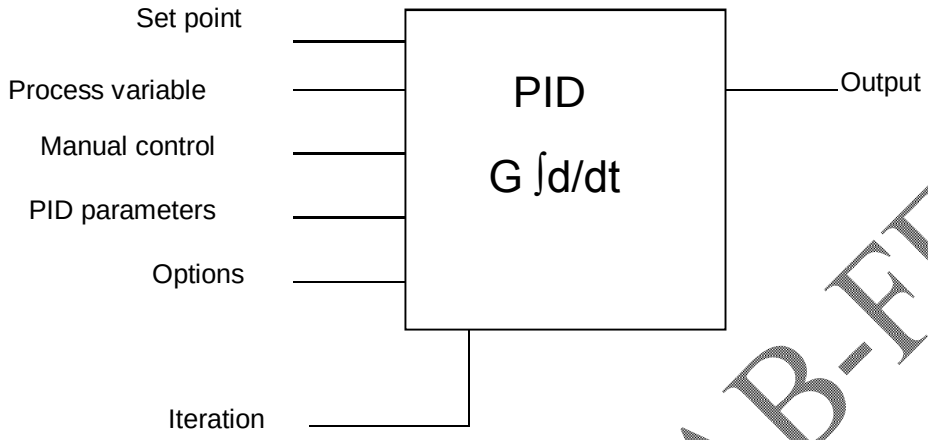


Figure 1. The PID software VI

EX.NO:	SIMULATION OF A TEMPERATURE PROCESS USING LABVIEW
DATE:	

Aim:

To simulate a temperature control system using Lab VIEW software and to study the PID toolkit accompanying Lab VIEW.

Introduction to PID control toolkit:

1. The PID toolkit adds sophisticated control algorithms to the instrumentation software development system.
2. By combing PID toolkit with math and logic functions in LabVIEW one can develop programs for the control system.
3. The closed loop or open loop tuning procedures can be adopted.
4. The PID function implement wide range of PID control algorithms.
5. The PID control algorithms incorporate

- Bump less auto/manual transfer
- Anti reset windup
- Direct or Inverse action
- Manual output adjustments

6. The PID control strategies can be designed and I/O values can be scaled from engineering units to percentage.
7. The National Instruments DAQ hardware can be integrated with the system for real world control.

The PID control algorithm:

In the PID controller, the set point is compared with the process variable to obtain the error.

$$e = sp - pv$$

The controller action can be basically calculated as

$$u(t) = K_c (e + 1/T_i \int e\,dt + T_d * de/dt) + bias$$

The required bias is to be added with u(t). Here K_c is the controller gain, T_i is the integral time in minutes and T_d is the derivative time in minutes. The PID VI's implement the 'position form PID algorithm'. The process variable filtering minimizes the effect of noise.

The controller output is limited to the range of specified controller outputs.

If

$u(k) >= u_{max}$ then $u(k) = u_{max}$

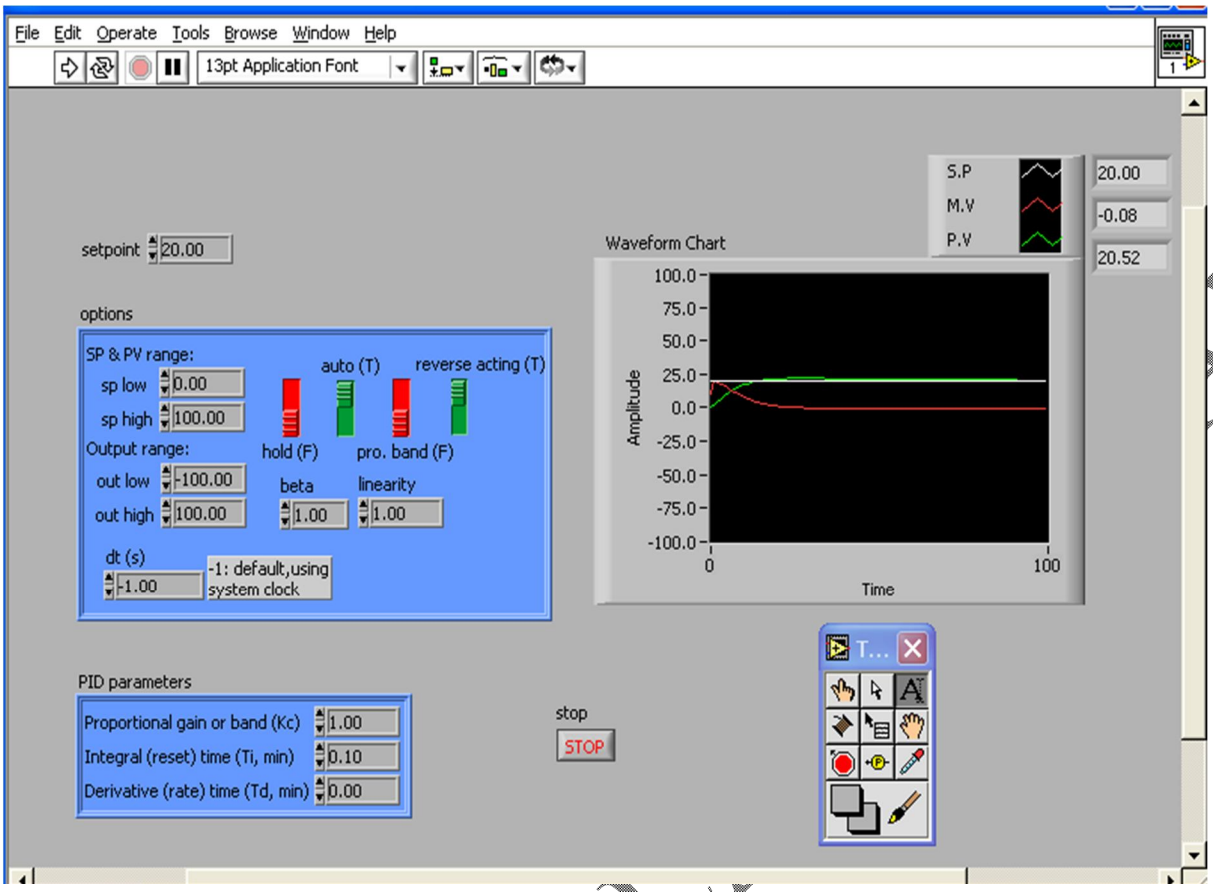


Figure 2.LabVIEW Front Panel for Temperature Process

$u(k) <= u_{min}$ then $u(k) = u_{min}$

One practical model of the PID controller is

$$u(k) = K_c[(\beta * sp - pv) + 1/T_i \int 1/ (sp - pv)dt - T_d \, dpv_f / dt] + bias$$

where β is the set point factor and PV_f is the value of process variable after filtering. The PID VI's use an integral sum correction algorithm that facilitates anti-reset windup and bump less transfer. Anti reset windup is the upper limit of the controller output, for example when 100% error immediately changes sign, the controller output starts to change appropriately in view of anti-windup feature. The algorithm prevents abrupt controller output changes when it is switched from manual to automatic mode.

Reverse action is the normal controller mode in which controller output decreases as process variable increases. Switching to 'hold' mode or 'manual' mode freezes the output at the current value of the input. All transfers are bumpless. These PID VI's can be called from inside a 'while loop' with fixed cycle time.

PID VI:

The PID VI is the basic PID algorithm. The PID VI has inputs for set point, process variable, manual control and PID parameters. The PID parameter input is cluster of three values proportional gain (K_c), integral time (T_i) and derivative time (T_d). The 'options' input specify additional parameters like hold, PB/K_c, Auto/ Manual and Reverse/Forward. The Boolean input 'hold' is used to hold the controller output at current state. The Boolean input PB/K_c specifies whether the proportional input parameter is the gain (K_c) or proportional band (PB). Gain is related to PB as

$$K_c = 100 / (\% PB)$$

Additionally the ranges for set point and controller output can be set.

Trapezoidal integration is used to avoid sharp changes in integral action when there is a PV or SP jump.

$$u_i(k) = K_c / T_i \sum[\{ e(i) + e(i - 1)\}/ 2] * \Delta t [1 / \{ 1 + (10 * e(i^2) / SP_{rng}^2)\}]$$

where SP_{rng} = set point range.

Because of the abrupt changes in the set point only filtered PV is applied to derivative action instead of error 'e' to avoid derivative kick.

$$u_d(k) = -K_c \, T_d / \Delta t \, \{ PV_f(k) - PV_f(k-1)$$

Now the controller output is

$u(k) = u_p(k) + u_i(k) + u_d(k)$

where

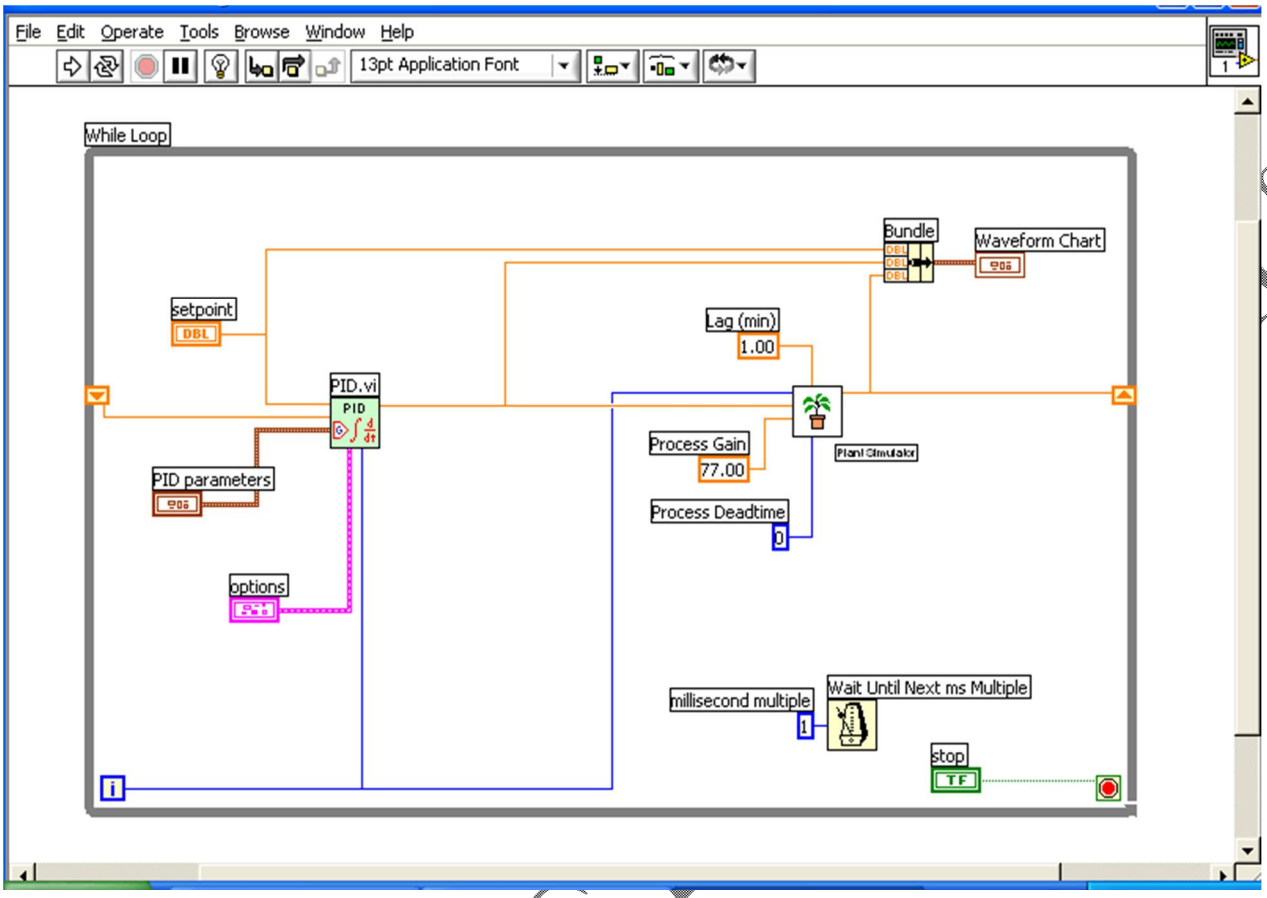


Figure 3.LabVIEW Block Diagram for Temperature Process

$u_p = K_c * e(k)$

The PID software VI:

This VI calculates the analog value of the process variable and set point using PID algorithm. The block diagram of a PID VI is shown in Figure 1.

Set point is the desired value for the process variable.

Process variable is the value of the feedback control loop.

Manual control is relative controller output value.

PID parameters are a cluster of Kc, Ti and Td where Kc is proportional gain,Ti is integral time in minutes and Td is the derivative time in minutes.

An 'option' is a cluster of 11 elements specifying optimal parameters for PID algorithm.

1. SP low is the minimum value for process variable and set point.
2. SP high is the maximum value for process variable and set point.
3. OUT low is the minimum value for the controller output.
4. OUT high is the maximum value for the controller output.
5. Hold (F), when 'true' places the controller in hold mode. Reset action stops and output freezes.
6. Auto (T), if 'true' selects automatic control and places the controller in manual mode when' false'. Bumpless transfer is used from auto to manual.
7. Proportional band(F), selects whether the proportional value of the PID parameters input is proportional gain or proportional band.
8. Reverse acting(T), if 'true' selects reverse action, the usual mode for controllers.
9. Beta is relative emphasis of disturbance rejection to set point tracking.
10. Linearity sets the linearity of error response, ranging from zero to one. '1' gives a plain linear response, while '0.1' gives an approximately parabolic (square law) response.
11. dt(s) is the interval in seconds at which the VI is called.

Iteration is the control loop iteration value.

Output is the output of control algorithm.

Exercise problem:

Simulate a temperature control system given by the transfer function

$$G(s) = K e^{-tds} / (\tau s + 1)$$

Where K = 77

τ = 1 min

td = 0

Design a PID controller for the given process.

Procedure:

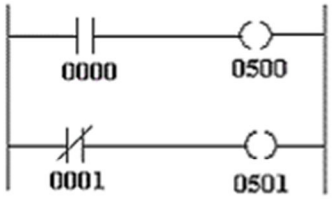
1. Construct the front panel and block diagram as shown in Figures 2 and 3.
2. Enter the PID parameters.
3. Give a step change in set point.
4. Run the VI and check the response.

Result:

Thus a temperature control system is simulated using Lab VIEW software.

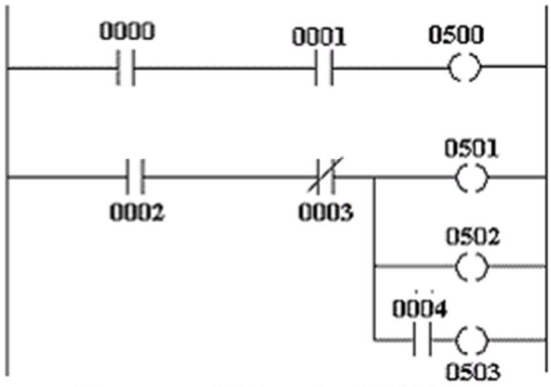
EXAMPLE LADDERS

1. To verify NO and NC contact



If 0000 is high, 0500 is on
If 0001 is low, 0501 is on

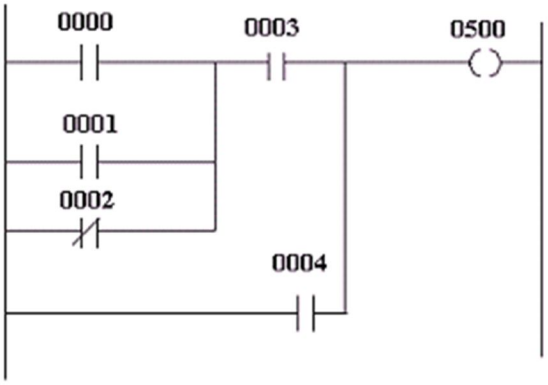
2. To verify AND logic



To turn on 0503, make 0002 high, 0003 low and 0004 high

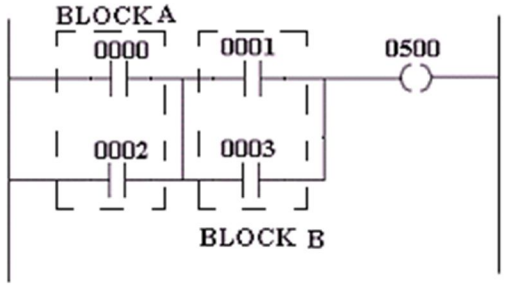
To verify OR,AND logic

3.

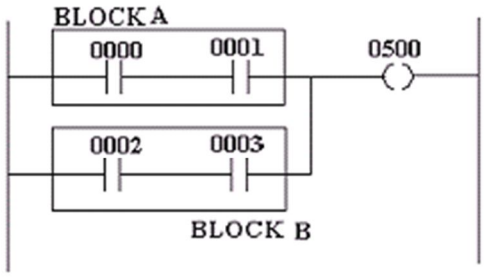


To turn 0500 on, 0003 must be closed and either 0000 or 0001 must be closed or 0002 must remain closed or 0004 must be closed

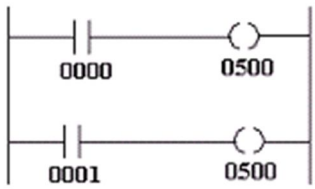
4.



5.



6. To verify Input priority



Though the output coils are same, inputs are different but priority is given to the input that is given last.

EX.NO:	STUDY OF PROGRAMMABLE LOGIC CONTROLLER

AIM

To study and familiarize the KV–16 (Keyence Make) ladder Builder software for PLC using examples.

THEORY

Programmable Logic Controller (PLC) is basically an assembly of solid-state digital logic elements designed to make logical decisions and provide outputs that were previously accomplished by electromechanical relays. PLC is used for the control and operation of manufacturing process equipment and machinery.

Programmable controllers offer several advantages over conventional relays. Relays have to be hard wired to perform specific functions. When requirement changes, the relay wiring has to be changed. But in case of programmable controller the hard wiring associated with the conventional relay control is eliminated. PLC is small, inexpensive, provides solid state reliability, low power consumption and ease of expandability.

The KV–16 contains 16 Input /Output modules, i.e. 10 input lines and 6 output lines. The KV–16 PLC is programmed using Ladder Builder software and KV Incrediware (based on MS-DOS). The PLC communicates with PC using RS–232C standard interface.

BASIC OPERATIONS USING LADDER BUILDER FOR KV-16

EDITOR:

The editor offers the following functions:

i. Creates a ladder diagram using diversified instructions of the ladder language.
ii. Performs editing functions.
iii. Registers comments to contacts and transfers them to a KV series CPU unit.
iv. Converts ladder diagrams and transfer the converted diagrams to a PLC.

The maximum number of lines that can be edited for one ladder program by the KV Incrediware is 9,999 only.

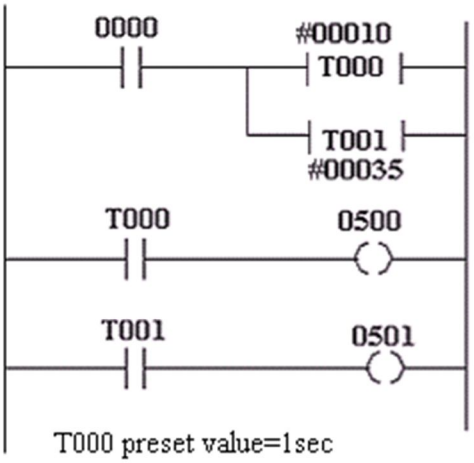
ENTERING/DELETING SYMBOLS AND CONNECTION LINES:

A ladder symbol is automatically entered when an instruction word is specified or by entering ‘A’ key, ‘B’ key in the keyboard or a coil in the current cursor position.

Some of the KV available Symbols are

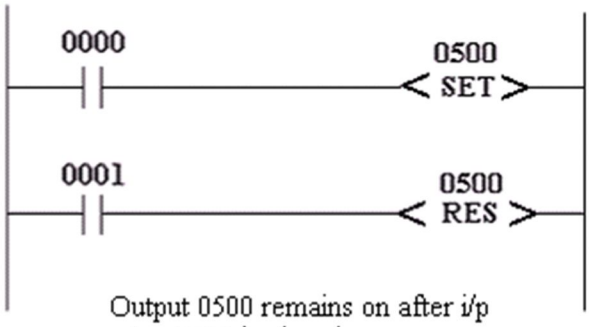
(i) NO (Normally Open) contact input - - - || - - - (F3) key.
(ii) NC (Normally Closed) contact input - - - |/| - - - (Shift+ F3) key.
(iii) NO Coil - - - ( ) - - - (F4) key.
(iv) NC Coil - - -()- - - (Shift + F4) key.

7. Timer (on delay)



T000 preset value=1sec
T001 preset value=3.5sec

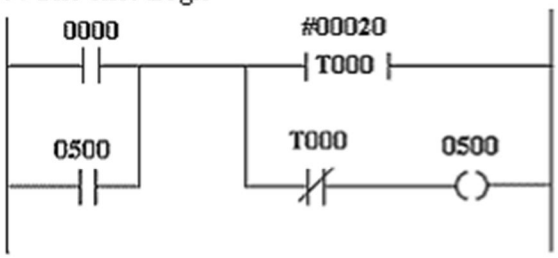
8. To set and reset a coil



Output 0500 remains on after i/p relay 0000 is closed.
output 0500 off when i/p 0001 is closed.

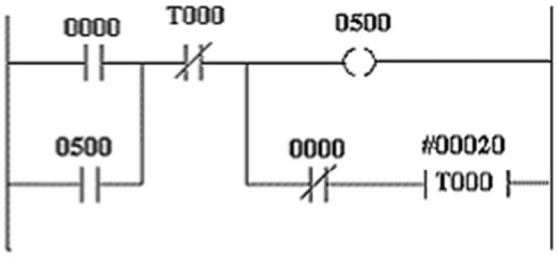
A timer is used to activate or deactivate a device after a preset time interval.
There are 2 types of timer, On delay and Off delay timer.

9. One shot Logic



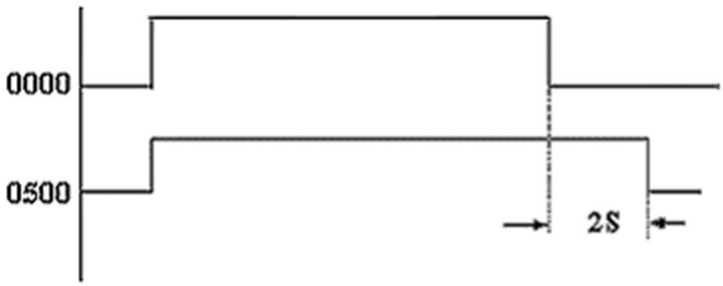
When 0000 is closed, timer T000 is on.
After 2sec 0500 is on forever.

10a. Off delay Timer



An Off delay timer delays turning Off.
When 0000 is closed, 0500 is on.
When 0000 is opened after 2sec 0500 is off.

10b. Off delay output



SIMULATOR

The Simulator offers the following functions:

(i) Confirms a program's operation status in real time using the ladder monitor.

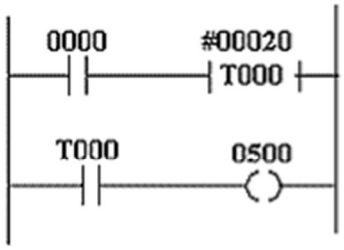
(ii) Performs simulations using diversified execution methods such as continuous scan, continuous step, one step and one scan.

(iii) Provides easy adjustment of present values and current values of timers, counters and devices as well as contact comments.

(iv) Sets and resets contacts forcibly.

The simulator can be started up by opening a ladder program created using the editor.

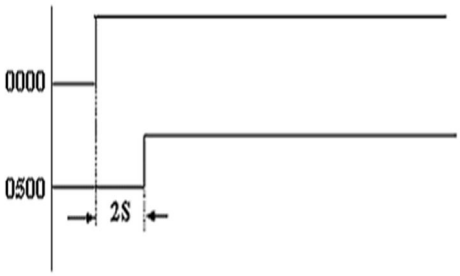
11a. On delay Timer



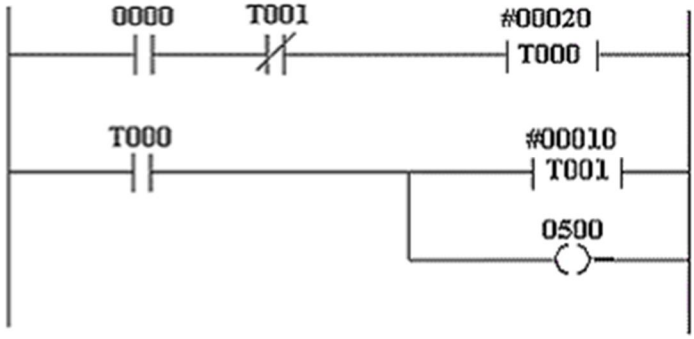
when 0000 is closed Timer is on. After 2sec 0500 is on.

An On delay timer delays turning On.

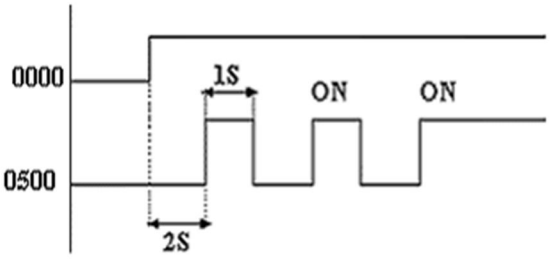
11b. On delay output



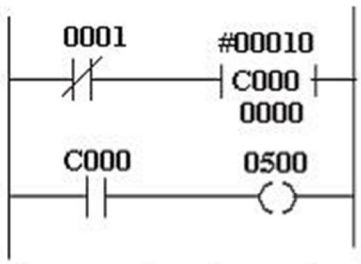
12a. Flicker Logic



12b. Flicker Output

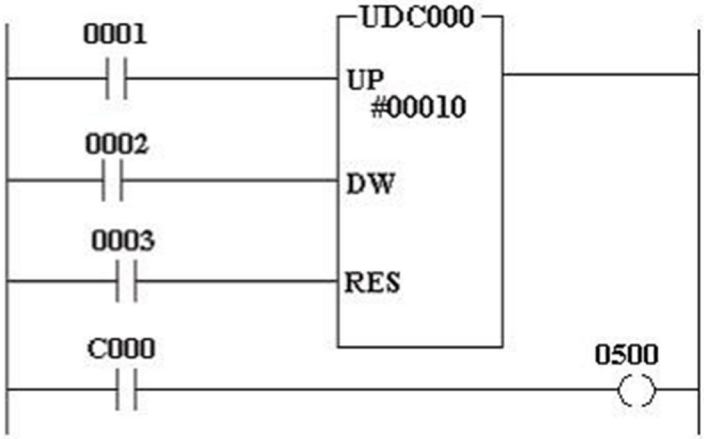


13 a. Counter (UP)



A counter is used to activate or deactivate a device after a preset count interval.

14. Counter (UP and DOWN)

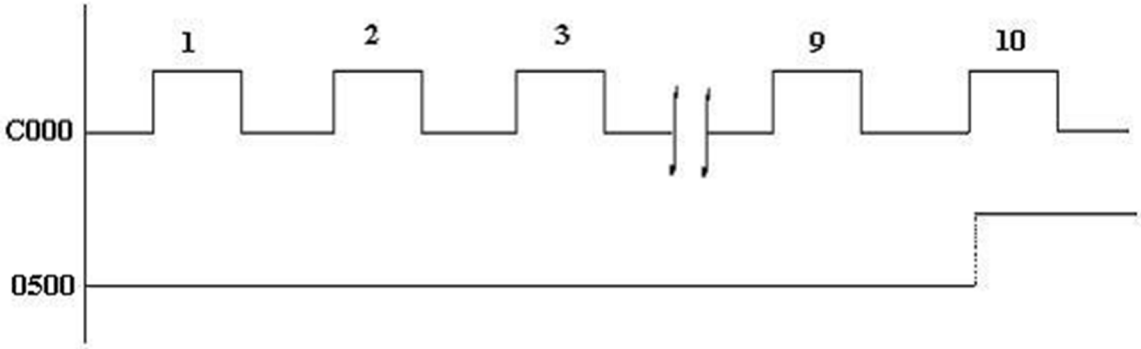


PROCEDURE

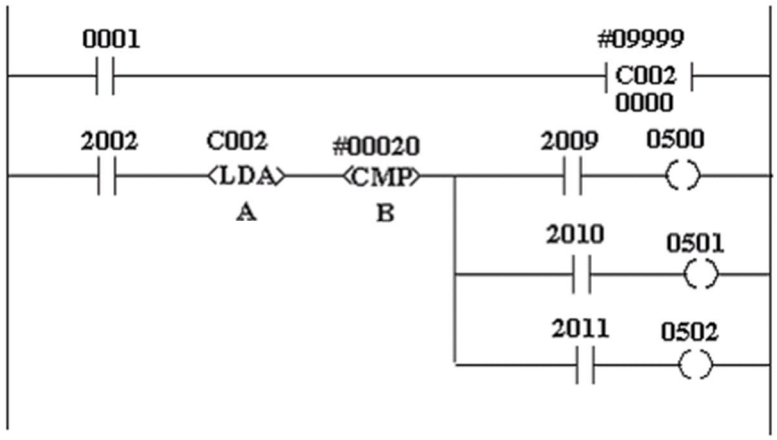
1. Switch on the PC
2. In the C:\> type CD KVLADDER . The following prompt appears on the screen C: \ KVLADDER>

   Then type “mnu” to enter into KVLADDER menu.
3. From the KVLADDER menu, select “Edit” option. In the “Edit” option, select “New Ladder File”. In that menu, type a name of your file with extension *.ldr. Then press “Enter” key to get editor screen for entering the program.
4. Enter the program.
5. To compile and simulate the entered program, select “Save >> Compile >> Simulator “menu by pressing “F1” key in the editor screen.
6. To simulate the program, give appropriate input and observe the result.

13 b.Counter

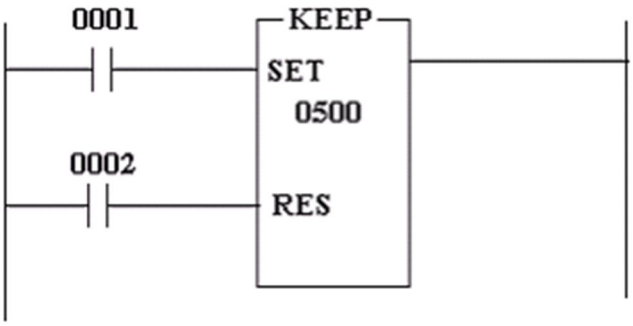


15. Multistep Counter



If A<B 0500 is on, A=B 0501 is on, A>B 0502 is ON

16. Keep Logic



RESULT

Thus the ladder builder software for Keyence PLC was studied and verified.

EXERCISE

1. Draw the general block diagram of PLC & explain its function.
2. Give the advantages of using PLC over hardwired relay contact type controller.
3. Differentiate between ON-LINE and OFF-LINE programming
4. List few applications of PLC.
5. Name any 3 standard proprietary models available in the market.

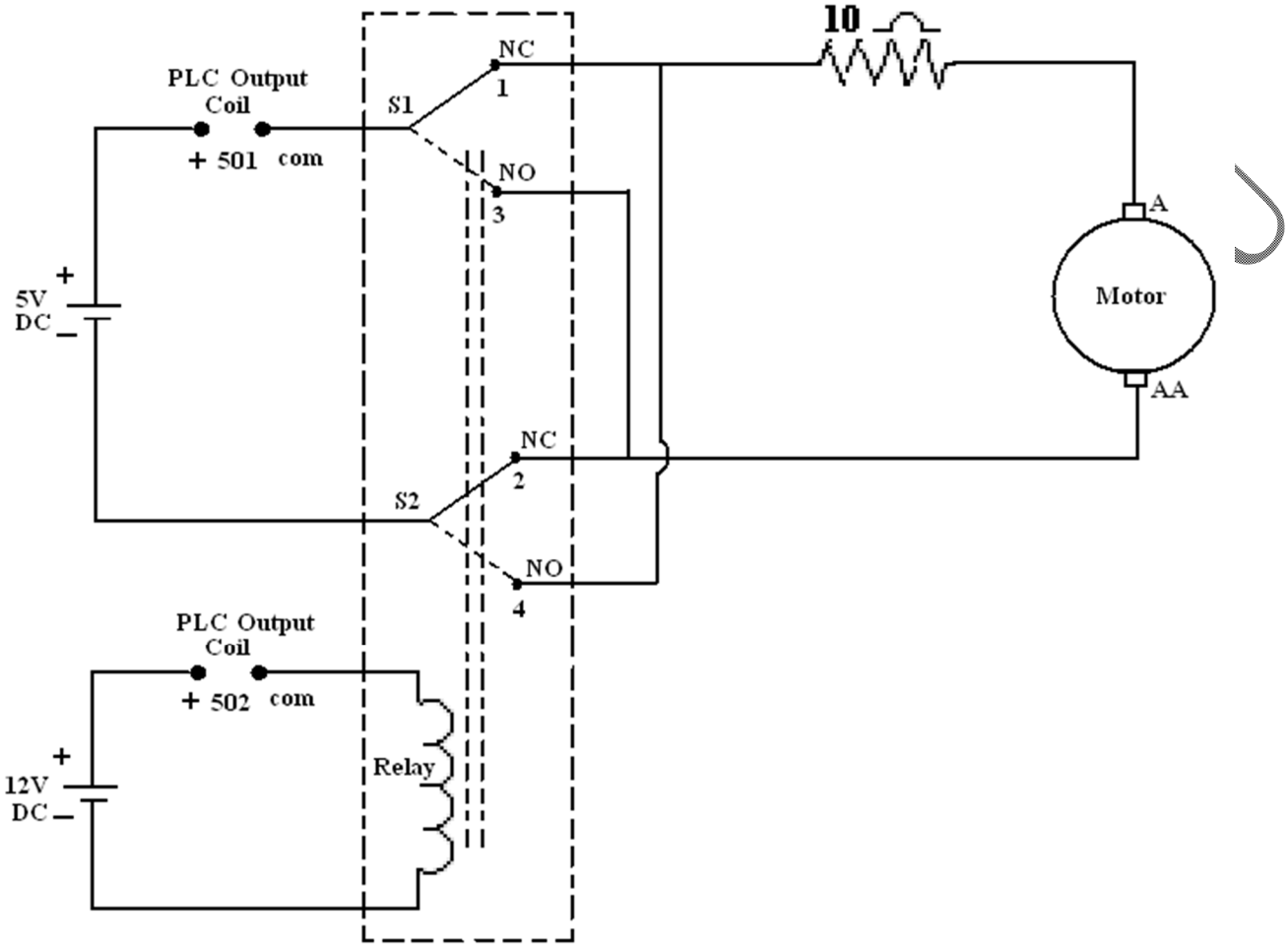


Figure 1. Connection diagram for motor control.

EX.NO:	DIRECTION CONTROL OF A DC MOTOR USING PROGRAMMABLE LOGIC CONTROLLER

AIM

To develop a ladder logic sequence using ladder builder software for a DC motor direction control problem and implement the same using Keyence PLC (KV - 16).

THEORY

In this experiment, it is proposed to change the direction of rotation of the motor in clockwise and in anticlockwise direction. Assuming that the field current of the motor is kept constant and the armature voltage is used to change the direction of the DC motor with the help of relay contacts and PLC outputs. The relay contact is energized to change the direction of voltage applied across the armature terminals of a DC motor.

PROBLEM STATEMENT

Develop a PLC ladder logic for the following sequence of operation:

1. Switch “ON” the process by pressing the START push button.
2. Rotate the motor in clockwise direction for 5 seconds.
3. Switch “OFF” the motor for 5 seconds.
4. Rotate the motor in anticlockwise direction for 5 seconds.
5. Switch “OFF” the motor for 5 seconds.
6. Repeat the sequence from step 2 until the STOP push button is pressed.

The above sequence of operation can be explained with the help of figure.1.

The power supply for the motor and the relay circuit should be switched ON after pressing the start Push Button (PB), the coil contact 501 is closed and the current flows through armature coil from A to AA. During this time, the coil contact 502 remains open and the relay coil is not yet energized. After 5 seconds 501 is opened and the motor stops running. After 5 seconds, both the contacts 501 and 502 are closed. Now the current flowing in the relay circuit will magnetize the relay coil in such a way that the contacts S1 and S2 changes from 1 to 3 and 2 to 4, respectively. Due to this change in the contact, the current in the motor circuit will flow in the opposite direction from AA to A. This reversal in the direction of current makes the motor to rotate in the anti clockwise direction for 5 seconds.

This sequence is repeated until the stop Push Button is pressed.

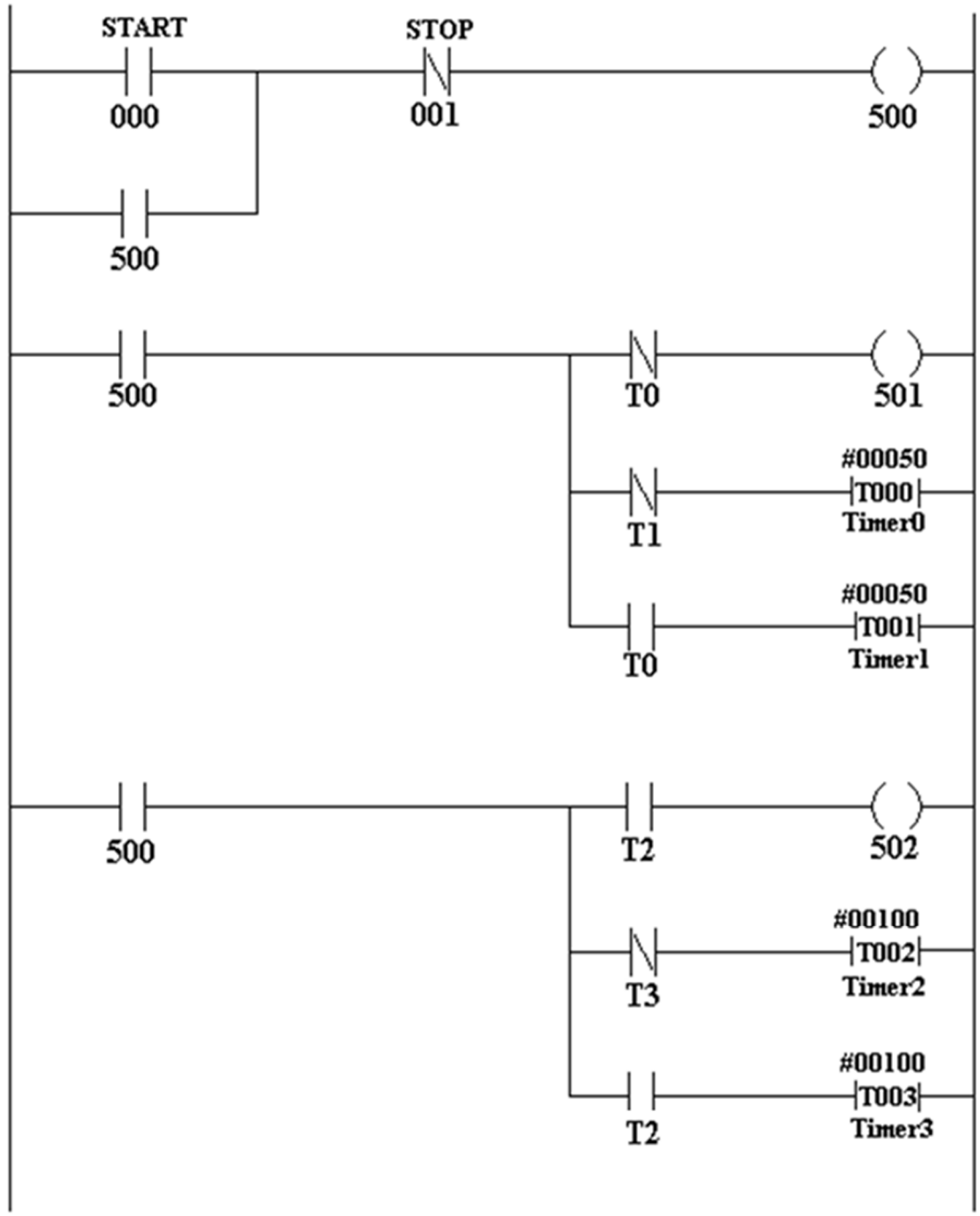


Figure.2. Ladder diagram for motor Control Application

LADDER LOGIC

The ladder logic sequences fro the given motor direction control problem is as shown in Fig.2. The hardware wiring diagram of motor with PLC is shown in Fig.1.

The timing diagram of the coil contacts and the output sequence is shown in Fig.3. Turning ON and OFF sequence of contacts 501 and 502 are listed in Table.1.

EXPERIMENTAL PROCEDURE

1. Using the Ladder builder software , develop the ladder logic as shown in Fig.2.
2. Set the necessary input voltage for motor(5V), relay circuit(12V) and PLC(24V).
3. Now connect the motor and relay circuit with PLC as per Fig.1.
4. Download the ladder logic to the PLC.
5. Execute the program in PLC and verify the results.

501	502	Output
0	0	OFF
0	1	OFF
1	0	Clockwise
1	1	Anticlockwise

Table1. Logic Table for Motor Control Problem

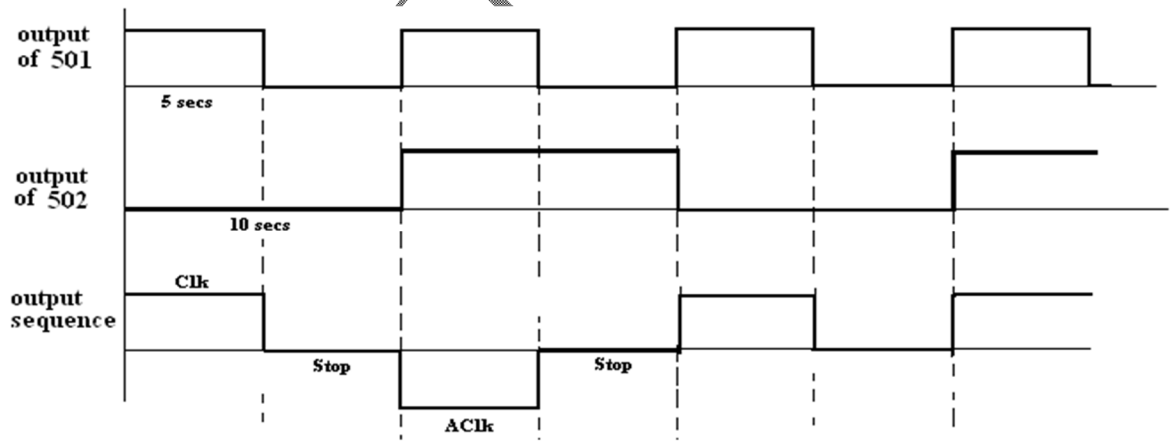


Figure 3. Timing diagram of PLC output coils

RESULT

The ladder logic sequence for direction control of a dc motor is developed and implemented using Keyence KV – 16 and the results are verified.

EXERCISE

1. Differentiate between ON delay and OFF delay timer.
2. What is an internal relay or internal coil?
3. Name the various blocks of PLC.
4. What is an intelligent I/O module

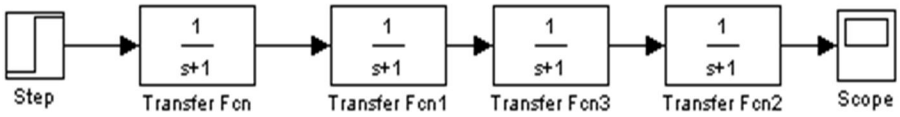


Figure1.Block diagram to get open loop response

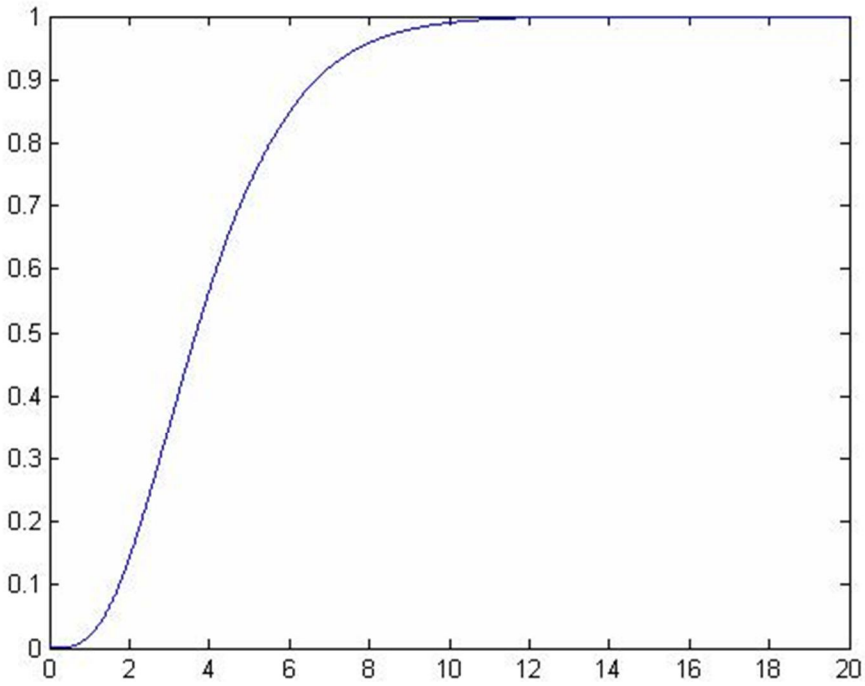


Figure 2 process response for $\frac{1}{(s+1)^4}$

From the response,

t_1 (23.6% of the final value) = 2.71secs

t_2 (63.2% of the final value) = 4.857secs

$\tau = 1.5*(t_2-t_1) = 1.5*(4.857-2.71) = 3.8$secs

$t_d = t_2-\tau = 4.85-3.8 = 1.28$secs

The approximated model is $\frac{1.04}{3.8s+1} e^{-1.28S}$

EX.NO:	PROCESS IDENTIFICATION USING LEAST SQUARE ESTIMATION

AIM

To identify the parameters of a linear discrete model for a given process using least square estimation algorithm.

INTRODUCTION

System identification is the art and exercise of identifying the various process variable or the underlying physical phenomena in a process (more often from a process data). The resulting expression is known as model.

Thus the model is a set of differential (or difference) and algebraic equations that can describe the process behavior.

The need for process model arises in many control applications for the use of timing methods. Process model are also needed in developing feed forward control algorithm, self-tuning and internal-model algorithm.

Process identification provides several forms that are useful in process control are

1. Process reaction curve (obtained by step input)
2. Frequency response diagram (obtained by sinusoidal input)
3. Pulse response (obtained by pulse input)

The n^{th} order linear discrete model is given by

$$\bar{y}_n = b_1 u_{n-1} + b_2 u_{n-2} ..... + b_{nb} u_{n-nb} - a_1 y_{n-1} - a_2 y_{n-2} - ..... - a_{na} y_{n-na} \quad (1)$$

Where $\bar{y}_n \rightarrow$ predicted value of the current output of the process.

$b_1, b_2, \ldots b_{nb}, a_1, a_2, a_{na} \rightarrow$ unknown parameters

$y_{n-1}, y_{n-2}, \ldots y_{n-na} \rightarrow$ previous values of output

$u_n, u_{n-1}, u_{n-2}, \ldots u_{n-nb} \rightarrow$ present and past values of input

Equ. (1) is usually written as

$$\bar{Y}(z) = \frac{B(z^{-1})}{A(z^{-1})} z^{-nk} u(z) \quad (2)$$

Now we define the parameter vector containing unknown parameters.

$$\theta = [b_1, b_2, \ldots b_{na}, a_1, a_2, \ldots a_{na}]^T \quad (3)$$

There are (n_b+n_a) unknown parameters.

Let us define NC as this total, so the vector has NC rows.

Let N=na, M=nb.

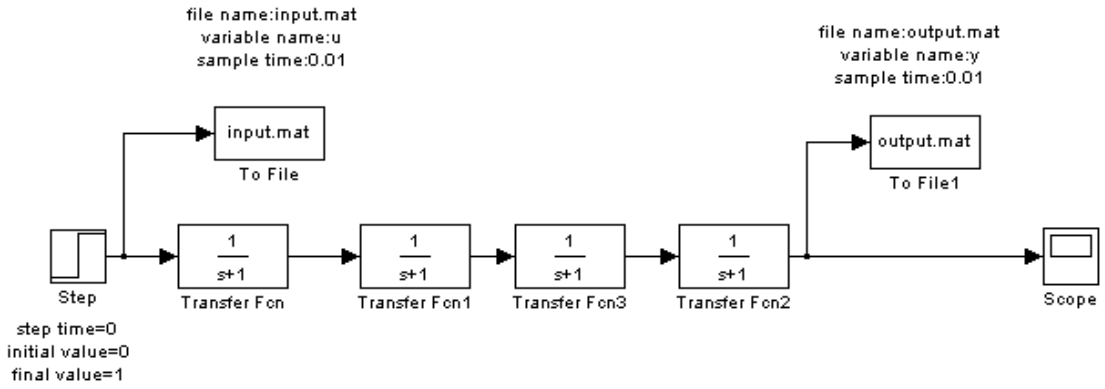


Figure 3.Simulink diagram for open loop response

MATLAB PROGRAM

```
loadoutput.mat;
loadinput.mat;
t=0:0.01:20;
y1=y(2,:);
plot(t,y1)
z=[y(2,:)' u(2,:)'];
na=1;nb=1;nk=125;
para=arx(z,[na,nb,nk],[],0.01);
[a,b]=th2tf(para);
k1=b(1,2);
k1=(-1.0)*k1;
k2=1-k1;
k3=a(1,nk+1);
kp=k3/k2
ts=0.01
deadtime= nk x ts
tau=(ts/log(k1))*(-1.0)
jmin=para(1,1)
```

Now suppose we have NP data points that give values of the output yn for known values of y_{n-1}, y_{n-2}, …y_{n-N}, u_n, u_{n-1}, … u_{n-M}.

The data could be grouped as follows.

	y_n values	y_{n-1} Values	L. y_{n-N} values	u_1 Values	u_{n-1} Values	L u_{n-M} Values
1	$y_{1,n}$	$y_{1,n-1}$	$y_{1,n-N}$	$u_{1,n}$	L	$u_{1,n-M}$
2	$y_{2,n}$	$y_{2,n-1}$	$y_{2,n-1}$	$u_{z,n-1}$	L	$u_{2,n-M}$
M	M	M	M	M		M
NP	$y_{NP,n}$	$y_{NP,n-1}$	$y_{NP,n-N}$	$u_{NP,n-1}$	L	$u_{NP,n-M}$

Our objective is to minimize the sum of the squares of the differences between actual measured data points ($y_{i,r}$) and those predicted by our model equation ($\overline{y}_{i,n}$) .

$$J = \sum_{i=1}^{NP} (y_{i,n} - \overline{y}_{i,n})^2 \qquad (4)$$

This is a least square problem that is solved by taking derivatives of J with respect to each of the unknown parameters (the NC elements of the θ vector) and setting these partial derivatives equal to zero. This gives NC equations i.e., NC unknowns. The solution is compactly written in matrix form.

$$\theta = [A^T A]^{-1} A^T Y \qquad (5)$$

where matrix A (with NP rows and NC columns) and the Y vector (with NP rows) are defined to represent the data points.

$$A\begin{bmatrix} y_{1,n-1} & y_{1,n-2} & L & y_{1,n-N} & u_{1,n} & L & u_{1,n-M} \\ y_{2,n-1} & y_{2,n-2} & L & y_{2,n-N} & u_{2,n} & L & u_{2,n-M} \\ M & M & & M & M & & \\ y_{NP,n-1} & y_{NPP,n-2} & L & y_{NP,n-N} & u_{NP,n} & L & u_{NP,n-M} \end{bmatrix} \qquad (6)$$

$$Y = \begin{bmatrix} y_{1,n} & y_{2,n} & y_{3,n} & K & y_{NP,n} \end{bmatrix}^T \qquad (7)$$

In this experiment the actual process is

$$G_P(s) = \frac{1}{(s+1)^4} \qquad (8)$$

The first order process with dead time can exactly model the above process as

$$G_p(s) = \frac{1.04e^{-1.28s}}{3.8s+1}$$

Let the first order model can be written in the form,

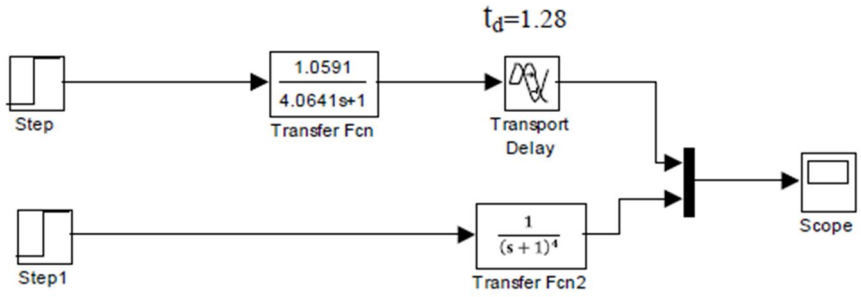


Figure 4 .Simulink diagram to verify the identified model with the original process

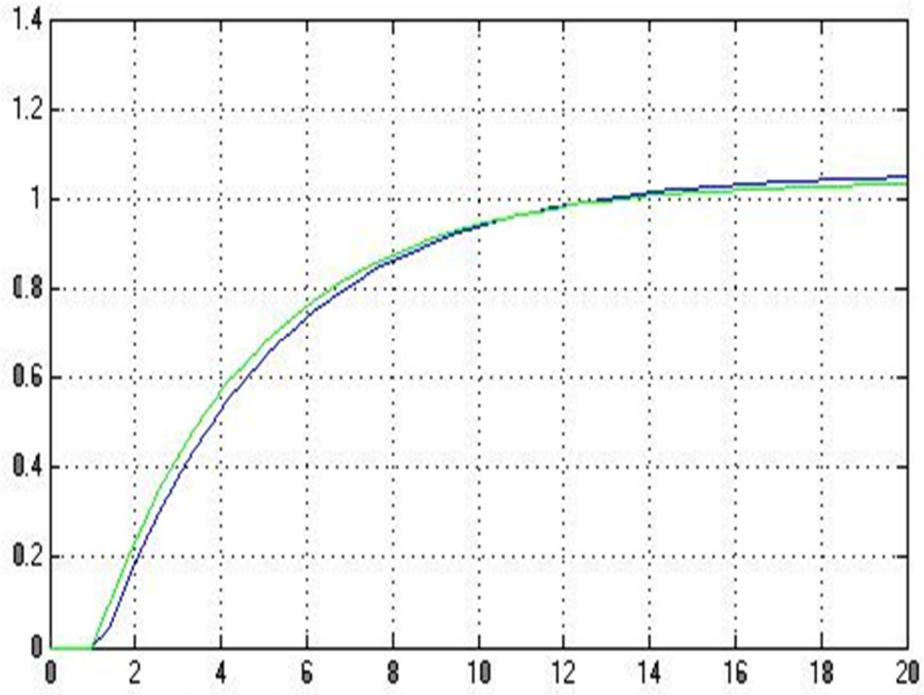


Figure 5 Estimated and Approximated model output

$$\frac{y(s)}{u(s)} = G_p(S) = \frac{K_P e^{-t_d s}}{\tau s + 1}$$

Let $G_o(s) = \frac{1 - e^{-sT}}{s}$

Where T $\rightarrow$ sampling period

The pulse transfer function of this process with zero order hold is

$$Z\,[G_o(s)\,G_o(s)] = Z\left[\frac{1-e^{-sT}}{s} \times \frac{kpe^{-tds}}{\tau s+1}\right]$$

Let td = nk* T

$$nk = \frac{td}{T}$$

$$Z\left[G_o(s)G_p(s)\right] = Z\left[\frac{1-e^{-ST}}{s} \times \frac{kpe^{-(nkTs)}}{\tau s+1}\right]$$

$$= z^{-nk}\,(1-z^{-1})\,Z\left[\frac{kp}{s(\tau s+1)}\right]$$

$$= z^{-nk}\,(1-z^{-1})\;Z\left[\frac{kp/\tau}{s(s+1/\tau)}\right]$$

$$\frac{kp/\tau}{s(s+1/\tau)} = \frac{A}{s} + \frac{B}{(s+1/\tau)}$$

$$A = \frac{kp/\tau}{s(s+1/\tau)} * s\,|_{s\to 0} = \frac{kp/\tau}{1/\tau} = kp$$

$$B = \frac{kp/\tau}{s\,(s+1/\tau)} * (s+1/\tau)\,|_{\,s\to -1/\tau}$$

$$= \frac{kp/\tau}{-1/\tau}$$

B = - Kp

$$Z\,[G_o(s)\,G_P(s)] = z^{-nk}\,(1-z^{-1})\,Z\left[\frac{kp}{s} - \frac{kp}{(s+1/\tau)}\right]$$

TABULATION:

nk	Kp	t_d	τ	J_{min}

$$= (1 - z^{-1})\, z^{-nk}\, kp\; Z\left[\frac{1}{s} - \frac{1}{s + 1/\tau}\right]$$

$$= (1 - z^{-1})\, z^{-nk}\, kp\left[\frac{z}{z-1} - \frac{z}{z - e^{-T/\tau}}\right]$$

$$= \frac{z-1}{z} = z^{-nk}\; kp\left[\frac{z^2 - z\, e^{-T/\tau} - z^2 + z}{(z-1)(z - e^{-T/\tau})}\right]$$

$$= z^{-nk}\, kp\left[\frac{1 - e^{-T/\tau}}{z - e^{-T/\tau}}\right]$$

Let $b = e^{-T/\tau}$

$$\log b = -T/\tau \quad \tau = -\frac{T}{\log b}$$

$$Z\,[G_o(s)\, G_p(s)] = z^{-nk}\, kp\left[\frac{1 - b}{z - b}\right]$$

$$Z\,[G_o(s)\, G_p(s)] = z^{-nk}\, \frac{kp\,(1 - b)}{z - b}$$

$$HGp(z) = \frac{y(z)}{u(z)} = \frac{kp(1 - b)}{z^{nk}\,(z - b)}$$

Multiply numerator and denominator by z^{-1}

$$\frac{y(z)}{u(z)} = \frac{kp(1-b)\, z^{-1}}{z^{nk}\,(z-b)\, z^{-1}}$$

$$HGp(z) = \frac{kp(1-b)\, z^{-1}}{z^{nk}\,(1 - b\, z^{-1})}$$

Let $b_1 = Kp*(1-b)$

$a_1 = -b$

$$HGp(z) = \frac{z^{-nk}\, b_1\, z^{-1}}{1 + a_1\, z^{-1}}$$

Assume $b_0 = 0$, we get the final value as,

$$\frac{y(z)}{u(z)} = HGp(z) = z^{-nk}\, \frac{(b_0 + b_1\, z^{-1})}{1 + a_1\, z^{-1}}$$

The discrete transfer function has three parameters that need to be identified; nk, b_1 and a_1.

PROCEDURE

1. Obtain the open loop response of actual process Gp (S)(fig.1,fig.2).
2. From the open loop response, process parameters are estimated using Two point method.
3. Load the data files from the simulink environment to matlab editor and then execute the program to estimate the values of a, b, nk.
4. Tabulate Kp, τ, t_d and $J_{min}$for various values of nk.
5. Select the Kp, τ and t_d from table with the low J_{min} criterion.
6. Compare the identified model with the actual process using (fig.3) simulink. Check the accuracy of the estimated model.

RESULT

Thus the parameter of a linear discrete model for a given process has been identified using least square estimation approach.

The identified model is =

The actual process is $\frac{1}{(S+1)^4}$

PROBLEM STATEMENTS

1. Switch on the lamp using Tag Name directory.

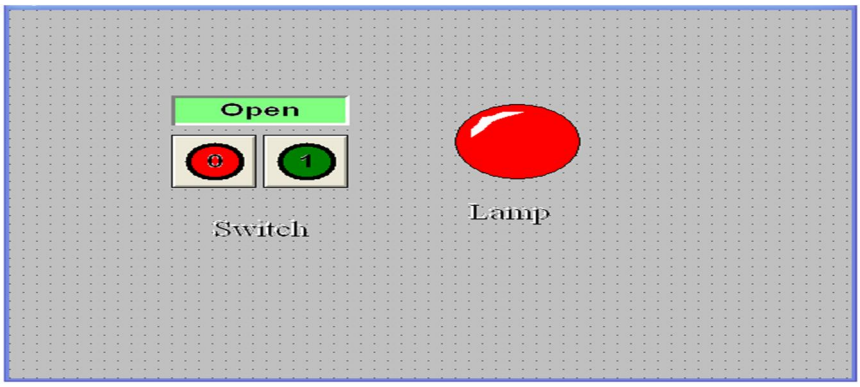


Figure 1: SCADA Mimic for Problem 1

2. Using the window script repeat the same example above.
3. Using symbols from the Symbols Factory develop a mimic to do the following using scripting:
   i. Start the process using a switch.
   ii. Increase the level in the Tank and indicate the level using a Analog Meter and a Tag Name Display unit.
   iii. Stop the process when the level reaches 95 cms.

EX.NO:	STUDY OF SCADA SOFTWARE

AIM

Study and implementation of Intouch wonderware SCADA software for a batch process.

SOFTWARE REQUIRED

SCADA Intouchwonderware software

INTRODUCTION TO SCADA

SCADA is an acronym that stands for Supervisory Control and Data Acquisition. SCADA refers to a system that collects data from various sensors at a factory, plant or in other remote locations and then sends this data to a central computer which then manages and controls the data.

As its name indicates, SCADA is not a full control system, but rather focuses on the supervisory level. As such, it is a pure software package that is positioned on top of hardware to which it is interfaced, in general via Programmable Logic Controllers (PLCs), or other commercial hardware modules.

SCADA is a term that is used broadly to portray control and management solutions in a wide range of industries. Some of the industries where SCADA is used are Water Management Systems, Electric Power, Traffic Signals, Mass Transit Systems, Environmental Control Systems, and Manufacturing Systems.

There are many parts of a working SCADA system. A SCADA system usually includes signal hardware (input and output), controllers, networks, user interface (HMI), communication equipments and software. All together, the term SCADA refers to the entire central system. The central system usually monitors data from various sensors that are either in close proximity or off site (sometimes miles away).
For the most part, the brains of a SCADA system are performed by the Remote Terminal Units (sometimes referred to as the RTU). The Remote Terminal Units consists of a programmable logic converter. The RTU are usually set to specific requirements.

One of key processes of SCADA is the ability to monitor an entire system in real time. This is facilitated by data acquisitions including meter reading, checking status of sensors, etc that are communicated at regular intervals depending on the system. Besides the data being used by the RTU, it is also displayed to a human that is able to interface with the system to override settings or make changes when necessary.

A SCADA system includes a user interface, usually called Human Machine Interface (HMI). The HMI of a SCADA system is where data is processed and presented to be viewed and monitored by a human operator he supervisory control system is a system that is placed on top of a real-time control

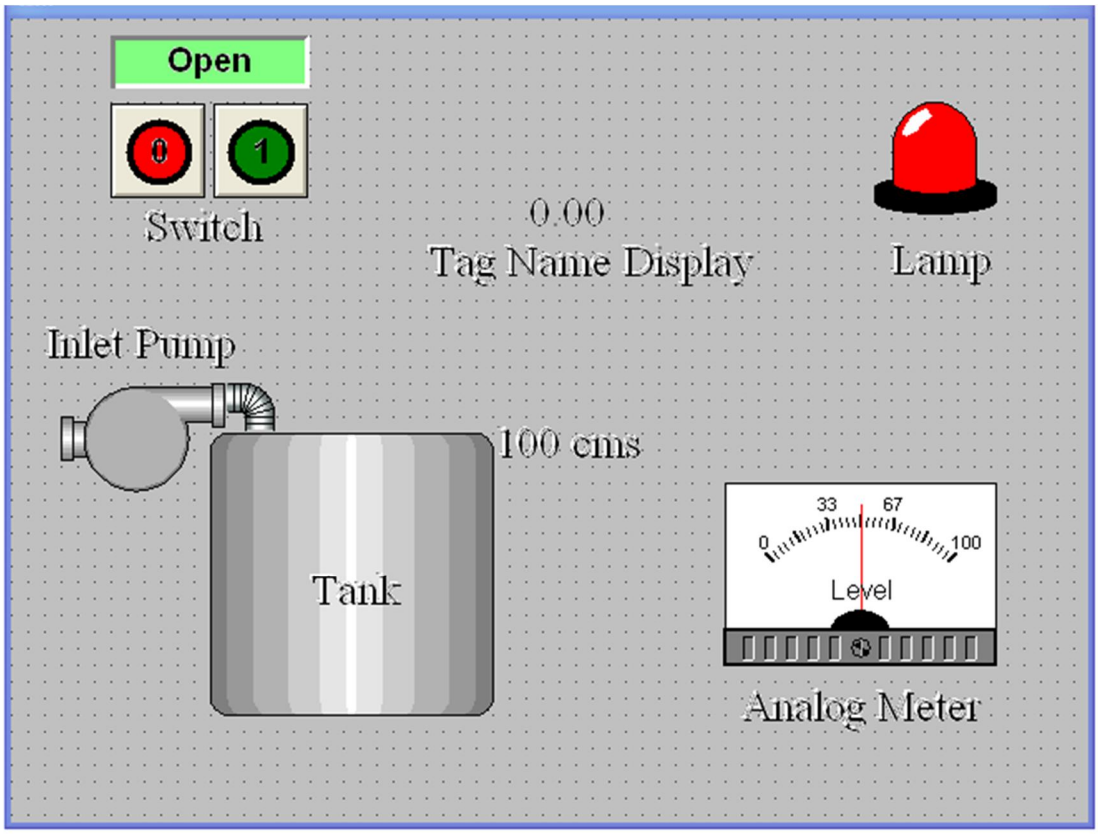


Figure 2: SCADA Mimic for Problem 3

4. using the symbol factory utility, develop a mimic for the following.

i. Have a Start/Stop button to run the process.

ii. Switch ON the pump when the process is started.

iii. When the level reaches a Set value, Stop the pump and start the Heater.

iv. When the temperature of the Fluid reaches a Set value, Stop the Heater and Open the Outlet Valve.

v. When the level in the tank reaches a minimum level, close the Outlet valve.

vi. Repeat the sequence until the Stop button is pressed.

system to control a process that is external to the SCADA system (i.e. a computer, by itself, is not a SCADA system even though it controls its own power consumption and cooling). This implies that the system is not critical to control the process in real-time, as there is a separate or integrated real-time automated control system that can respond quickly enough to compensate for process changes within the time-constants of the process.

A SCADA system is composed of the following:

1. Field Instrumentation
2. Remote Terminal Unit (RTU)
3. Communications Network
4. Master Terminal Unit (MTU)

1. Field Instrumentation

It refers to the sensors and actuators that are directly interfaced to the plant or equipment. They generate the analog and digital signals that will be monitored by the remote station. Signals are also conditioned to make sure they are compatible with the inputs/outputs of the (Remote Terminal Unit) RTU of PLC at the remote station. This part is also called as the local system. It is responsible for acquiring process status and actual value of process variables. They communicate with the control center at the lowest level of control hierarchy.

2. The Remote Terminal Unit

It is installed at the remote plant and controlled by the central host computer. This can be a Remote Terminal Unit (RTU) or a PLC. It gathers information from their remote site from various input devices, like valves, pumps, alarms, meters etc. Data is either analog or digital. The RTU's are designed for extremes in temperature, humidity and power outage conditions. This is one of the main advantages of the RTU. It can survive through adverse environmental conditions.

3. The Communications Network

It is the medium for transferring information from one location to another. It is quite possible that systems employ more than one means to communicate to remote sites. SCADA systems are capable of communicating using a wide variety of media such as fiber optics, dial-up, or dedicated voice grade telephone lines, or radio.

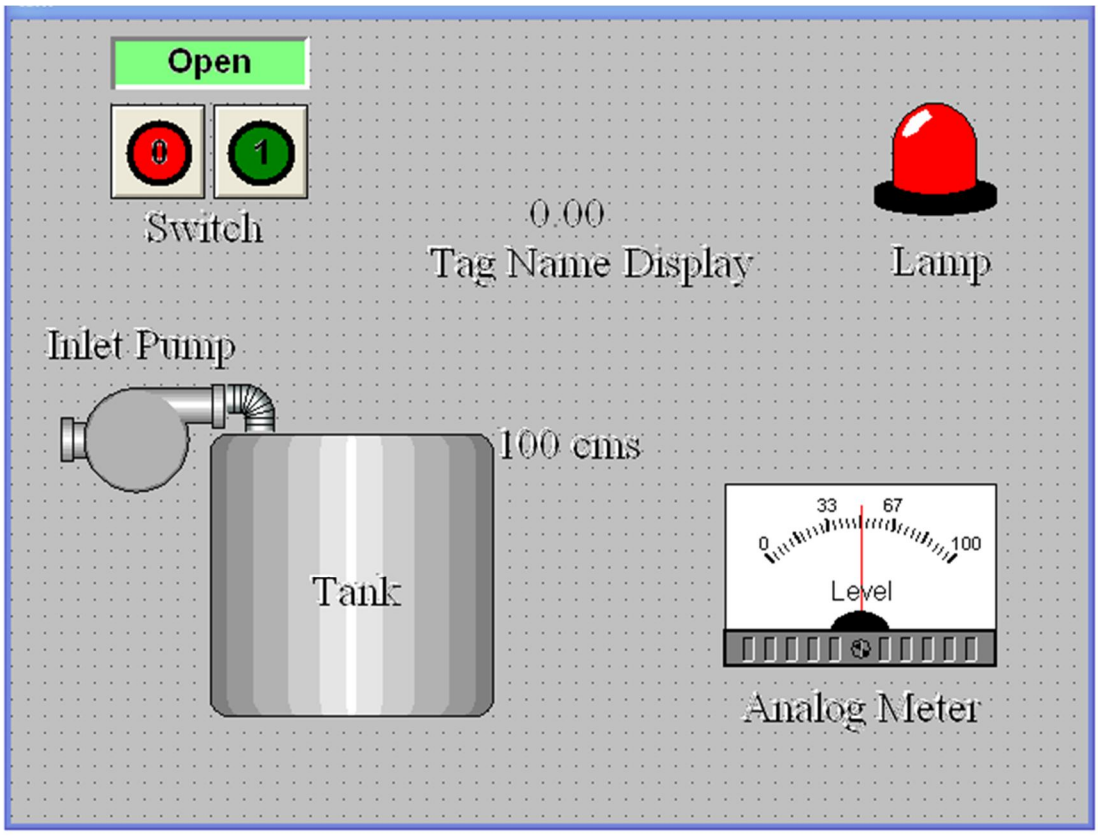


Figure 2: SCADA Mimic for Problem 3

4. using the symbol factory utility, develop a mimic for the following.

vii. Have a Start/Stop button to run the process.

viii. Switch ON the pump when the process is started.

ix. When the level reaches a Set value, Stop the pump and start the Heater.

x. When the temperature of the Fluid reaches a Set value, Stop the Heater and Open the Outlet Valve.

xi. When the level in the tank reaches a minimum level, close the Outlet valve.

xii. Repeat the sequence until the Stop button is pressed.

5. The Master Terminal Unit (MTU)

It refers to the location of the master or host computer. Several workstations may be configured on the MTU, if necessary. It uses a Man Machine Interface (MMI) program to monitor various types of data needed for the operation. Input to the system normally is initiated from the operator via MTU's keyboard. The MTU monitors information from remote sites and displays information for the operator. The relationship between MTU and RTU is analogous to master and slave.

A typical master unit consists of a base processing unit, operator interface, power supply, modem, input/output cards/relays, enclosure, battery backup, wiring terminals, lighting protection, programming and startup.

FEATURES OF SCADA SYSTEMS

A Smart Way to Operate SCADA systems are robust and "failure aware". The SCADA doesn't wait for things to go wrong; it continuously monitors itself for signs of trouble. By running multiple components across robust, redundant processors, when things do go wrong the system is programmed to take immediate steps to circumvent it. The remote telemetry units that perform most of the data acquisition and control employ similar redundant technology, so the personnel can stay focused on recovery of service, not on keeping the SCADA system up and running.

The SCADA system can be programmed to monitor its major physical components and to automatically reconfigure new "virtual" components as necessary whenever failures are detected. Virtual remote telemetry units, for example automatically are redeployed to a functioning server whenever a communication link server or a database server fails.

Well Connected

A SCADA connects to third parties at both the process level, where we have substations, signaling equipments and valves, and at the application level, where we have database, control and management functions. Using standard communication protocols, standard media interfaces, SCADA can interoperate with a wide range of different telemetry systems.

Alarm functions

It can use voice recording to communicate alarm conditions over telephones, pagers, two-way radios, or public address system. Remote Computers can dial into the SCADA system by modem and troubleshooting can be done remotely before traveling to the site.

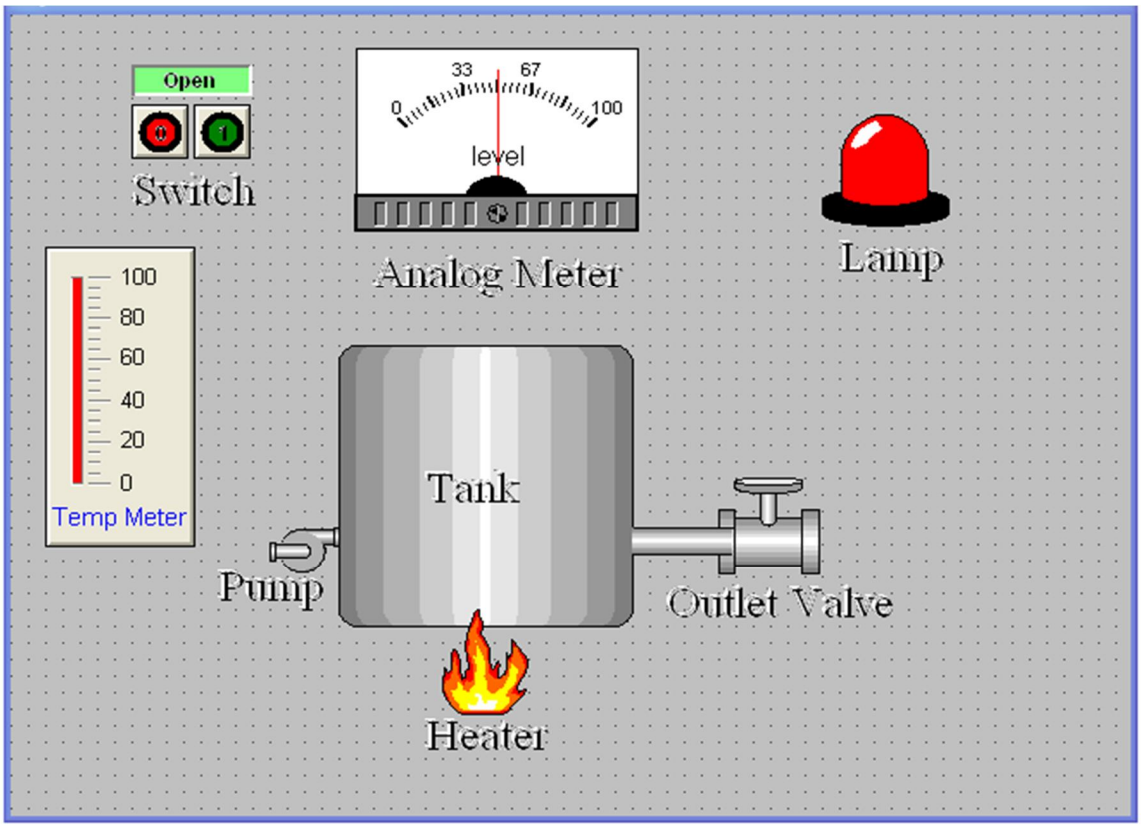


Figure 3: SCADA Mimic for Problem 4

SCADA SOFTWARE

Intouch is the quickest and easiest way to create man-machine interface applications for the Microsoft windows operating systems. The Intouch software consists of two major components- window maker and window viewer.

Intouch features

distributed alarm systems
distributed history
dynamic resolution conversion
dynamic reference addressing
network application development
factory focus
connectivity with more than 300 DDE servers
easily networked with wonderware net DDE
low cost process viewer solution
real time application process viewing
64 real time analog and discrete tags

Intouch program groups

Intouch runs the program that allows creating new Intouch applications open existing in touch applications and configure node specific settings.

Window maker runs Intouch in the development environment using the last – used Intouch application directory.

Window viewer runs Intouch in the runtime environment using the last used Intouch application directory.

His data runs the utility that acts as a DDE server for Intouch encrypted historical log files (*.LGH). His data is used with Intouch to retrieve requested historical data from the log file and if required converts it to a comma-separated variable (.CSV) file for use with spreadsheets or text editors.

SCRIPTS FOR BATCH PROCESS

Tag Names Used

SW - switch to turn the process ON/OFF

lmp – Lamp

pmp – Inlet Pump

out – Outlet Valve

heat – Heater

lev – Level in the Tank

temp – temperature of the fluid.

Window Script:

```
if sw == 1 then
lmp = 1
if level <= 5 then
pmp  =  1;
out   =  0;
heat =  0;
endif;

if level > 95 then
pmp  =  0;
out   =  0;
heat =  1;
endif;

if temp > 95 then
pmp = 0;
out = 1;
heat = 0;
endif;
if pmp = = 1 then
lev = lev + 1;
else
if out = = 1 then
temp = 0;
lev = lev – 1;
```

Tag name data dictionary

The tag name data dictionary (runtime database) is the heart of Intouch. At runtime it contains the current value of all of the items in the database. In order to create the runtime database, Intouch requires information about all of the variables being created. Each variable must be assigned a tag name and type. Intouch also requires additional information in order to be able to acquire the value and convert it for internal use. The tag name data dictionary is the mechanism used to enter this information.

Tags in wonderware and its types

A tag may be considered as a source of information from the real time devices like PLC, DCS, single or multi-loop controller or equipment. A tag typically may be a digital output, or an analog output.

Memory type tag-names

These exist internally within the Intouch application. They are used for creating system constants and simulations. There are four memory types.

- **Memory discrete** –internal discrete tag name with a value of either 0 or 1
- **Memory integer** –a 32 bit signed integer value between 0 to 2147483647
- **Memory real** – floating point memory tag name the floating point value between +/- 3.4 E38
- **Memory message** - text string tag name that can be upto 131 characters long.

Creating scripts in touch

In touch scripting capabilities allow you to execute commands and logical operations based on special criteria being met for example a key pressed a window being opened, a value changing etc. By using scripts a wide variety of customized and automated system functions can be created.

- Application scripts
- Window scripts
- Key scripts
- Condition scripts
- Data charge scripts
- Touch push button action scripts

```
else
if heat = = 1 then
temp = temp + 1;
endif;
endif;
endif;
else



lmp = 0;
lev  = 0;
pmp = 0;
heat = 0;
temp= 0;
out  = 0;
endif;
```

MIMIC DEVELOPMENT

Mimic screen development is an animated representation of the process undertaken. The mimic helps the user to get acquainted with the process and its working can be visualized by initiating the process in mimic screen itself. This type of developing mimics makes the job simpler for the use as he can control the process, verify the sequential execution of the same and also visualize text displays that tell about the current status of process.

Developing the mimic for the process depends on the P&I diagram and the process write up. A process can be a very complex system consisting of large number of field instruments like pressure gauge, valves, actuators, tanks and reactors. All these cannot be developed in a single mimic screen. Generally when a process is considered it will be composed of overall plant view, its individual stages, power distribution, alarms, trending and reports. All these cannot be viewed in a single mimic diagram. So they are divided according to the process being considered. When a number of mimics are developed, there must be parent window that gives the pathway of all the sub windows developed. This parent window must be such that it guides the operator to any part of the process.

To simulate the temperature process control here the various blocks are obtained from the wizards and pasted in the development environment. The blocks had properly assigned a tag name.

Each block is configured and then saved. The alarm conditions for this process are also given in the proper locations.

RESULT

A basic mimic is developed for a temperature process and it is simulated in the intouchwonderware environment.

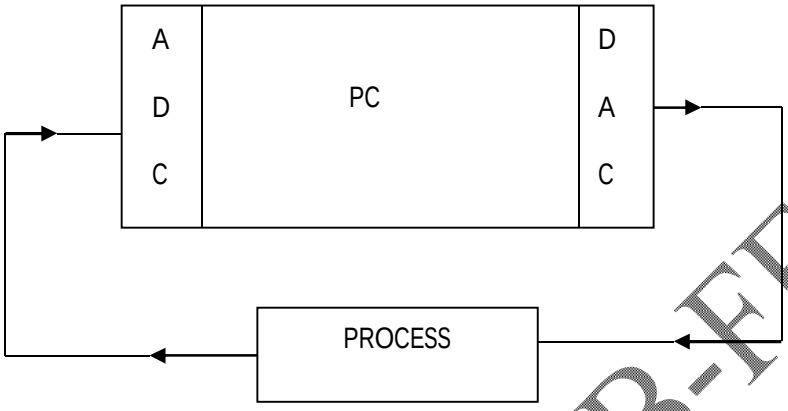


Figure 1 General Block Diagram

EX.NO:	PC BASED CONTROL OF SIMULATED PROCESS

Aim

(i) To study and interface PCS with personal computer.

(ii) To obtain the closed loop response of a first order system (or for a given process) using PID algorithm

APPARATUS REQUIRED:

Process control simulator, VMAT - 01 [ADC/DAC – ADD ON CARD], PC[PC-XT], patch chords.

THEORY:

The transfer function of the process is given by

$$G(s) = \frac{1}{(1 + \tau s)}$$

Where -----→τ=10 ms

The plant / process is simulated through OP-Amps and it satisfies the above transfer function. ADC/DAC card provides the interface between the plant and personal computer (or) between the analog & digital system.

The ADC allows the digital system to take in information from the analog system. [The o/p of the process]. The digital system [PC] can now rapidly analyze and process this information.

The DAC allows the result of such analysis to be communicated back to the analog system (process control simulator)

Position form algorithm:

$$m(k)=K_c\left[e(k)+\frac{(e(k)-e(k-1))T_d}{T}+\frac{T}{T_i}serror\right]+bias$$

where

K_c-----→ Gain

Serror -----> sum of the errors

e(k) ------> error at kth instant

m(k) ------> controller output at k th instant

T_d ----------> Derivative timeT_i -----------> Integral time.

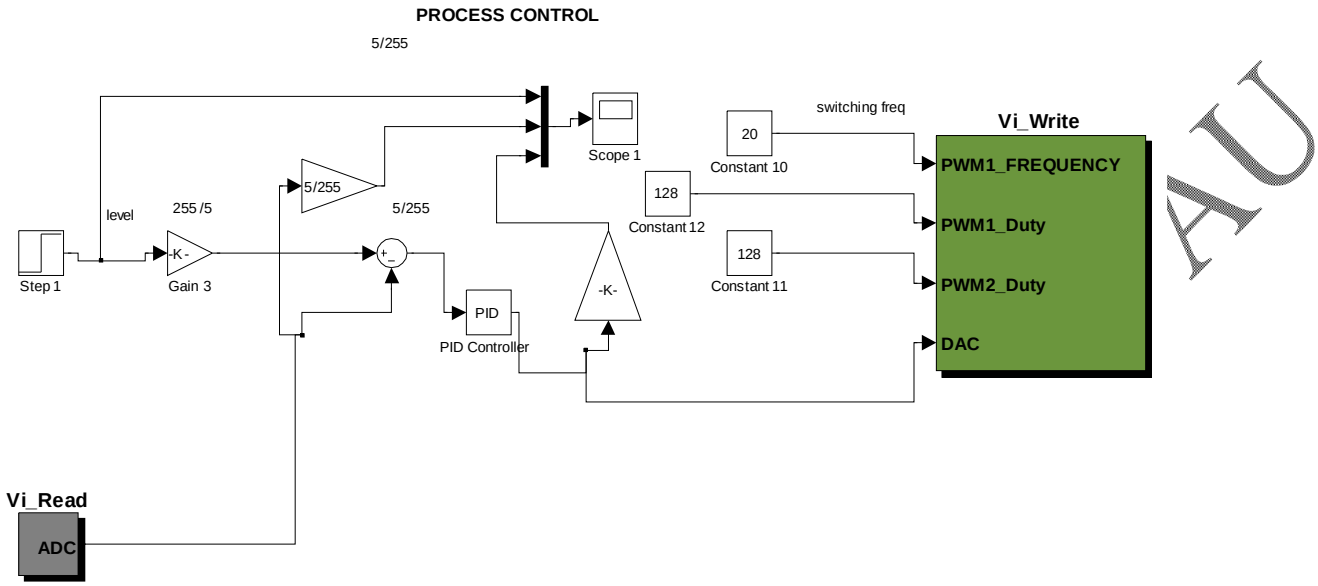


Figure 2.Simulink diagram of closed loop control

INTRODUCTION TO VMAT-01

In every educational field most of them are familiar with matlab because it is the environment to develop any algorithm easily. Our real time MATLAB interface card(VMAT - 01) that allows you to interface with MATLAB for controlling real time applications. We can implement our required closed loop algorithm by using MATLAB Simulink tools like PI, PID, FUZZY logic,etc.,

The VMAT - 01 Controller is specially designed for Real time interface for MATLAB simulink, Orcad simulation software. Based on advanced Risc Micro controller, a useful tool for researchers to interface their hardware with MATLAB to research in the fields like process control loop, DC - DC converter, DC motor controller, control Engineering etc., if we want to design this type of closed loop algorithms in any digital controller or some other controllers it is so difficult than MATLAB. Using our real time interface card (VMAT - 01) wecan gather real time signals and define our control technique in MATLAB then transfers the control signals to real time application.

FEATURES

* dSpic- Micro controller Based Controller.
* Can be used for instrumentation PID control Applications.
* Can be used for Power Electronics, DC-DC Converter, DC Motor control Applications.
* Single Channel ADC Input (0-5V).
* Single Channel DAC Output (0-5V).
* 2 Channel PWM Output (250 KHz).
* PC Interface Through RS232.
* +5 V DC Operation.

FRONT PANEL DESCRIPTION

Serial Port Interface - Interface PC and VMAT - 01 unit through RS232 cable.

5V Adaptor - To give supply to the unit.

GND - To connect the GND terminal.

ADC - To give the ADC input (0 - 5)V

DAC - To measure the DAC output (0 - 5)V

PWM1 - To measure the PWM1 signal.

PWM2 - To measure the PWM2 signal.

COMMUNICATION

Using serial port we can communicate our real time interface card (VMAT - 01) with MATLAB. For transmitting and receiving the data from the mat lab we provide some libraries. For receiving data from real time application use the **vi read** block. To transmit the control data to VMAT -01 card use the **vi write** block.

Figure 1. Front Panel Diagram

VI_READ

The ADC input of the VMAT-01 real time interface card can be gather into MATLAB as a output of VI_READ. The value of ADC input range 0-5V is proportional to the vi_read output integer value of (0-255). For example when we give 5V as a ADC input then the output of vi_read will be 255. By using this proportionality we can configure the real time data in Simulink Environment.

VI_WRITE

Above figure of vi_write is used to transfer the data to our VMAT-01 card. The main thing is we need to define our switching frequency (switching Time) of the pulse width modulation (PWM) signal. **DAC terminal** is used to control the output voltage of digital to analog converter from VMAT - 01 card.

255 data = 5V

For 1v output

= 255/5.

= 51.

For getting 1v output we have to give the input4 as 51.

Procedure

1. Calculate the controller parameter for the given process using synthesis formula Kc=τc/Kτ; Ti=τc where τc=closed loop time constant and τ=open loop time constant of process.
2. Connect the RS232 serial port with the PC and VMAT-01 real time MATLAB Interface card.
3. Connect the 5V power supply to the VMAT-01 real time MATLAB interface card.
4. Give your feedback voltage to the ADC terminal. That must be in 0 - 5V range.
5. Take DAC output from the VMAT-01 card.
6. Configure the Simulink diagram as shown in figure2 and execute it for various values of setpoint change and comment on the results

Result

Thus a simulated process in PCS is controlled by using VMAT -01 card from PC.

Membership Diagrams

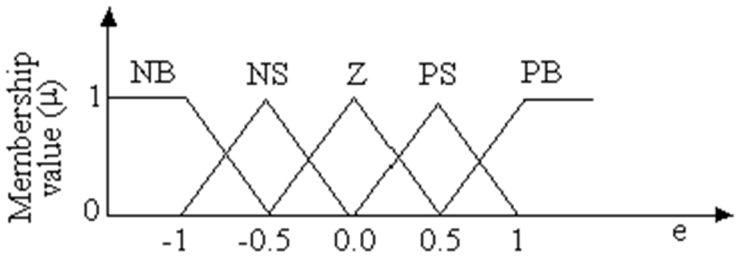


Figure 1. Membership diagram for error.

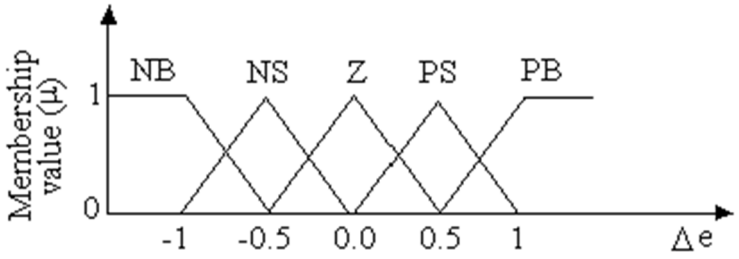


Figure 2. Membership diagram for change in error.

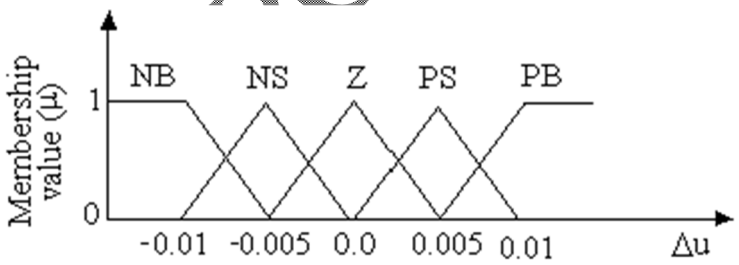


Figure 3. Membership diagram for change in output.

Table1.FAM (control rules) table

Δe / e	NB	NS	Z	PS	PB
NB	NB	NB	NS	NS	Z
NS	NB	NS	NS	Z	PS
Z	NS	NS	Z	PS	PS
PS	NS	Z	PS	PS	PB
PB	Z	PS	PS	PB	PB

EX.NO:	STUDY OF FUZZY TOOL BOX AND DESIGN OF
Date:	

AIM

1. To study the features of fuzzy logic toolbox.

2. To design a PI like Mamdani-type fuzzy logic controller (FLC) for

the given process $G(s) = \frac{e^{-2s}}{(15s+1)}$.

THEORY

The main paradigm of fuzzy control is that the control algorithm is a knowledge-based algorithm, described by methods of fuzzy logic. The fuzzy logic control system is a kind of expert knowledge-based system that contains the control algorithm in a simple rule-base. The knowledge encoded in the rule-base is derived from human experience and intuition and from the theoretical and practical understanding of the dynamics of the controlled system. This makes the fuzzy control special and conceptually different from conventional control. The operation of approximate reasoning converts the knowledge embedded in the rule-base (IF-THEN rules) into a crisp (nonfuzzy) control algorithm. Keeping this in mind, a PI like FLC is designed using fuzzy logic toolbox for the given process and its performances are compared with that of conventional PI controller. The operation of FLC can be explained as follows:

If the measured process output is a crisp quantity, it can be fuzzified into a fuzzy set. This fuzzy output is then considered as the fuzzy input to a fuzzy controller, which consists of linguistic rules. The output of the fuzzy controller is then another series of fuzzy sets. Since most physical systems cannot interpret fuzzy commands (fuzzy sets), the fuzzy controller output must be converted into crisp quantities using defuzzification methods. These crisp (defuzzified) control-output values then become the input values to the physical system and the entire closed-loop cycle is repeated.

PI-LIKE FLC

The FLC describes with the aid of fuzzy IF-THEN rules the relationship between the change in the control output $\Delta u(k) = u(k)-u(k-1)$ on one hand, and the error $e(k)$ and its change $\Delta e(k) = e(k)-e(k-1)$ on the other hand as given by

$$\Delta u(k) = F(e(k), \Delta e(k)) \quad (1)$$

The internal mechanism of the FLC translates it into a mapping:

$$\Delta u(k) = f(e(k), \Delta e(k)) \quad (2)$$

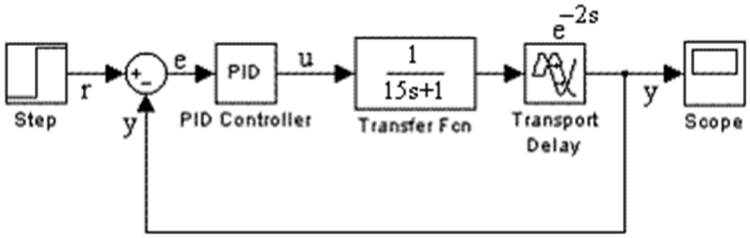


Figure 4.Closed-loop control implementing PI controller.

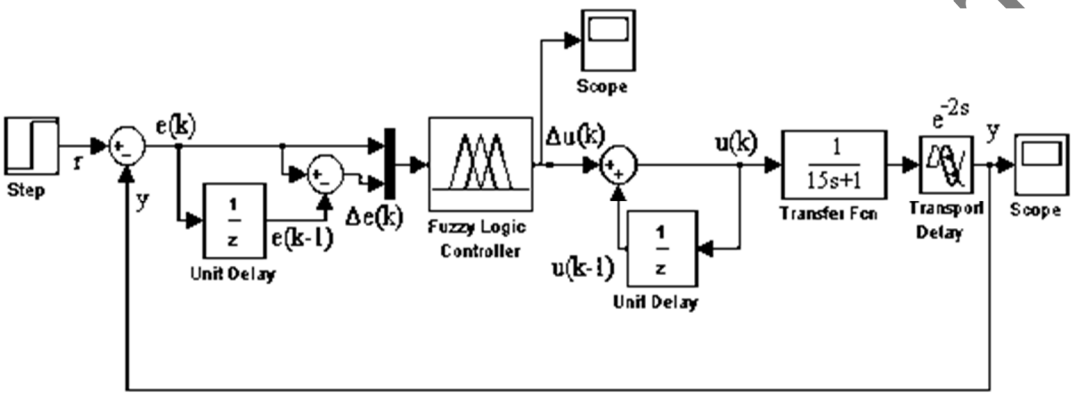


Figure 5. Closed-loop control implementing fuzzy logic controller.

Table 2. Performance Measures

Controller	Servo Response		Regulatory Response	
	Settling Time	% Overshoot	Settling Time	% Overshoot
PI Controller				
Fuzzy Logic Controller				

This relation is similar to conventional PI controller as

$$\Delta u(k) = K_p \Delta e(k) + K_I e(k) \qquad (3)$$

where K_p and K_I are the parameters of the PI controller. This FLC is called as a PI-like FLC. In case of PI controller, the equation (3) is linear, while in FLC, equation (2) is nonlinear.

DESIGN PROCEDURE OF FLC

Important steps in the design of FLC are listed as

Step1.*Determine the input and output variables of FLC*: By appropriate selection of the input and output variables the controller can be identified as PI-like, PD-like, or PID-like FLC.

Step2.*Define the parameters of FLC*: determine the scaling factors and the normalized universes of discourse and membership functions of the reference fuzzy sets associated with the linguistic labels of the main variables like e, Δe(k) and Δu(k).

Step3.*Determine the rule-base of the FLC*: The rule table is formed using the three meta rules given by

Step 4. *Computational realization of the FLC*: The fuzzy output of the FLC is

converted into a crisp output using defuzzification method.

1. If the error e (k) and its change Δe(k) are zero, then maintain present control setting.
2. If the error e (k) is tending to zero at a satisfactory rate, then maintain present control setting.
3. If the error e(k) is not self-correcting, then control action setting Δu(k) is not zero and depends on the sign and magnitude of e(k) and Δe(k).

FUZZY TOOLBOX COMMANDS

1. FIS editor - Using this command Mamdani or Sugeno type FIS can be selected.
2. Input / Output - Using this command number of inputs / Outputs can be selected.
3. MF editor - Using this command number of membership functions (MFs) can be added, modified or deleted. Also type of MFs can be selected.
4. Rule editor - Using this command rules can be edited, modified or deleted.
5. Rule viewer - Firing strength of the rules can be viewed.
6. Surf viewer - Output surface can be viewed.

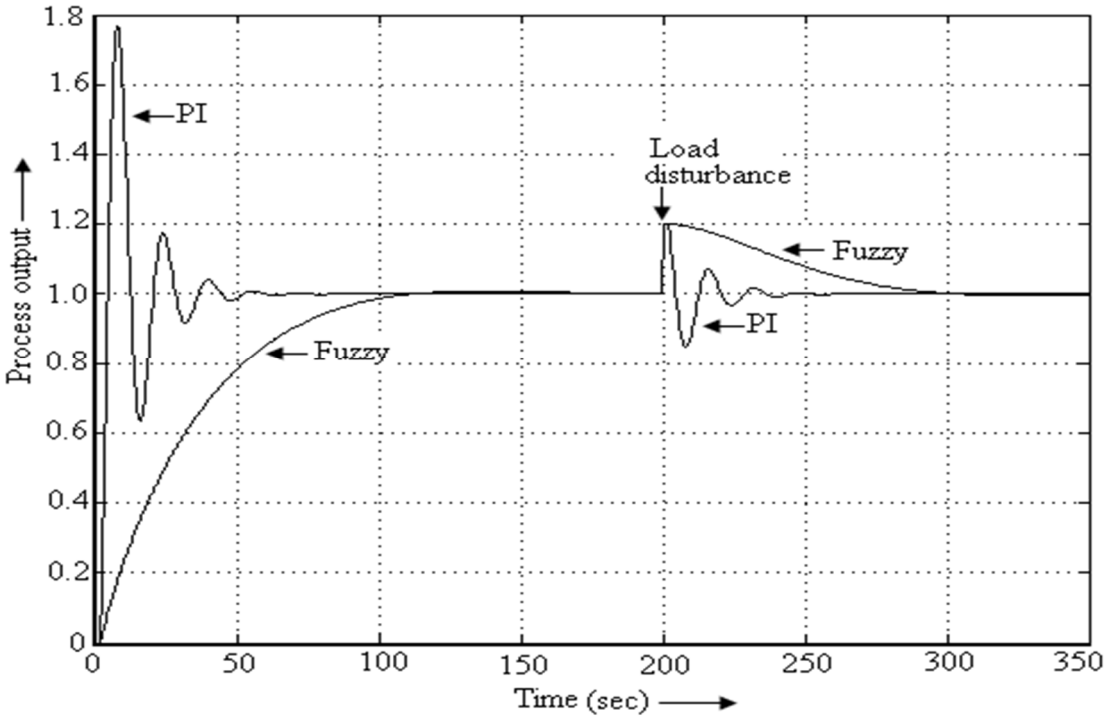


Figure 6. Model waveforms of regulated output voltage with PI and Fuzzy Logic Controller.

FLC DESIGN FOR THE GIVEN PROCESS

Reference input : r(k)

Process output : y(k)

Error e(k) : r(k)-y(k)

Change in error Δe(k) : e(k)-e(k-1)

Input variables : Error **e** and Change of error **Δe**.

Output variable : Change of output **Δu**.

Membership function type : Triangular with 50% overlapping.

Defuzzification method : Center of Gravity (COG) method.

The range of error : [-1 1].

The range of change in error : [-1 1].

The range of change in output : [-0.01 0.01].

The membership diagrams for e, Δe and Δu are shown in Figures 1,2 and 3. The fuzzy control rule table for the given process is given in Table1.

SIMULATION PROCEDURE

1. Develop the closed-loop control for the given process with conventional PI controller using Simulink as shown in Figure 4.

2. Similarly develop the closed-loop control for the given process with fuzzy logic controller using Simulink as shown in Figure 5.

3. Simulate individually for a unit step input and obtain the responses as shown in Figure 6.

4. Compare the performances of PI and fuzzy logic controllers from servo and regulatory responses and tabulate in Table 2.

RESULT

Thus the features of fuzzy toolbox are studied and a FLC is designed for the given system.

EXERCISE

1. Why Fuzzy Logic Controller is preferred for non-linear systems?
2. What are the different types of FLC?
3. Why triangular membership function is ideal over other types?