

REPORT

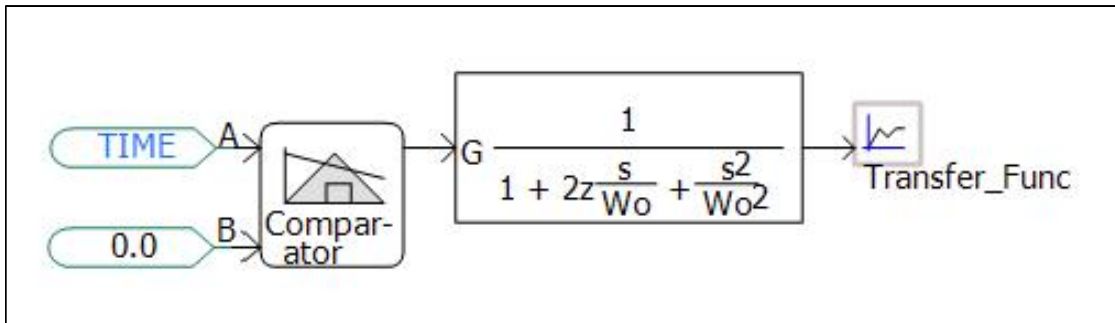
PSCAD 설계 보고서4



과목명	전력변환디바이스
담당교수	심재웅 교수님
학과	융합전자공학과
학년	3학년
학번	201910906
이름	이학민
제출일	2023.11.09.

1. Example1

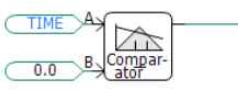
1) PSCAD 설계

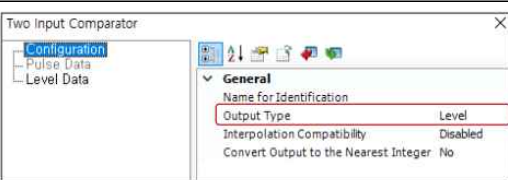
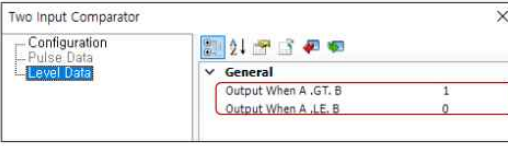


2차 시스템 $G(s) = \frac{100}{(s^2 + s + 1)27}$ 설계

2) 시뮬레이션 환경 설정

① Two Input Comparator 설정



의미: 시간과 숫자 0을 비교한다. (그림은 0초에 신호를 보내겠다는 의미)

출력신호형태: Level (Time이 높으면 신호가 계속 나옴 - PWM방식)

신호: A가 더 큰 경우 1이 출력되고, B가 더 큰 경우 0이 출력됨

위와 같이 Comparator를 설정하여 0초부터 신호가 출력되도록 설정하였다.

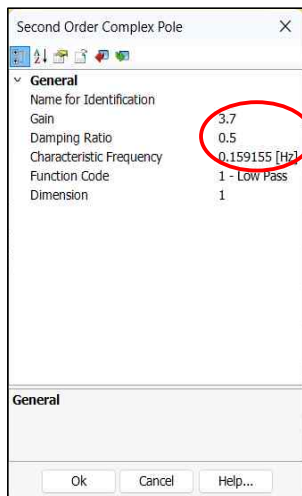
② 전달함수 설정(이득, 감쇠비, 고유 주파수)

주어진 문제의 전달함수에서 이득, 감쇠비, 고유 주파수를 계산하는 과정은 다음과 같다.

$$G(s) = \frac{100}{(s^2 + s + 1)27} = \left(\frac{1}{s^2 + s + 1} \right) 3.7 \rightarrow \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

제동비 ζ 및 고유 주파수 $\omega_n \rightarrow 2\zeta\omega_n = 1$ 이고, $\zeta = \frac{1}{2\omega_n} = 0.5$, $\omega_n^2 = 1$ 이므로, $\omega_n = 1 \rightarrow$ PSCAD입력 시 : $f_n = \frac{1}{2\pi} [\text{Hz}]$

계산된 값을 PSCAD에 적용하면 다음과 같다.

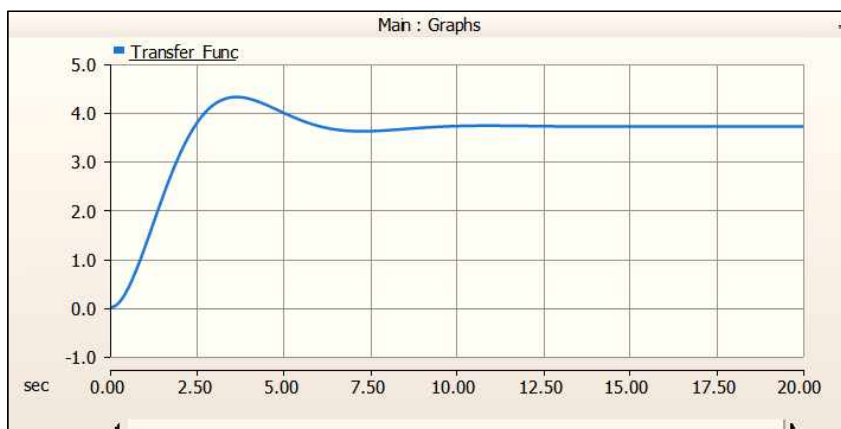


$$Gain = 3.7$$

$$Damping Ratio = 0.5$$

$$Characteristic Frequency = \frac{1}{2\pi} = 0.159155 [\text{Hz}]$$

3) PSCAD 시뮬레이션 결과



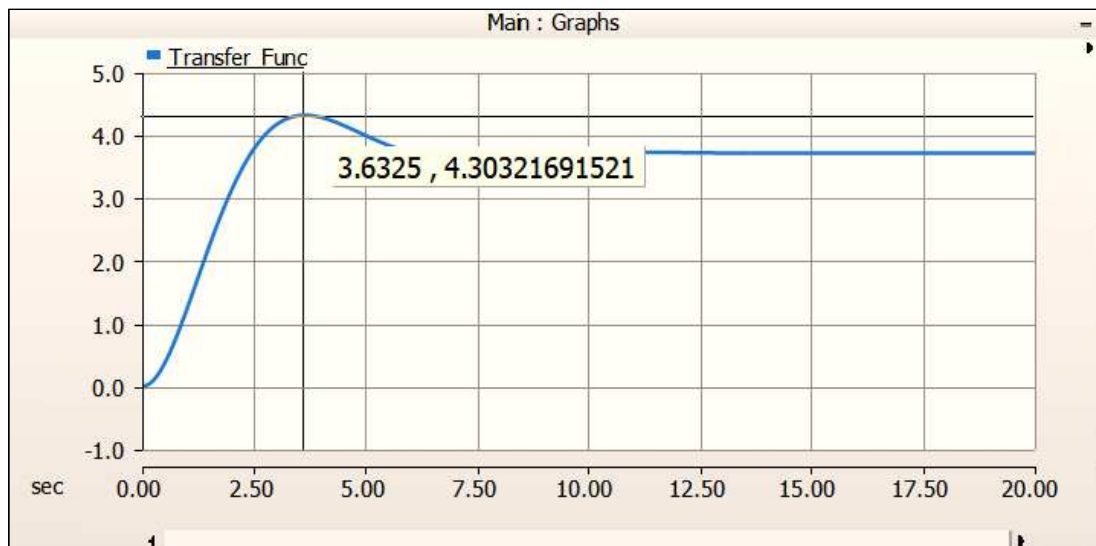
시뮬레이션을 실행하여 얻은 그래프는 다음과 같다.

4) 결과 분석 및 결론

문제에서 주어진 성능지표를 계산하면 다음과 같다.

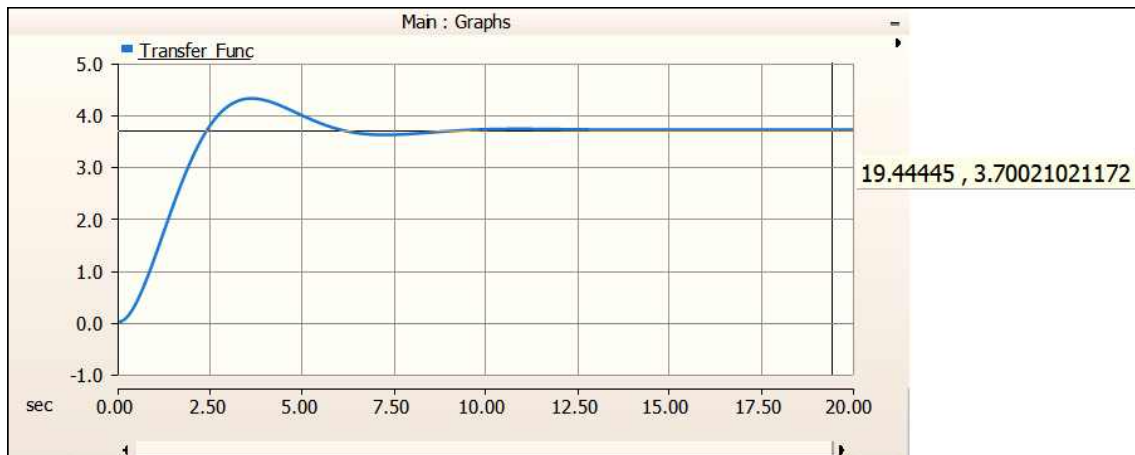
최대 오버슈트 시간 :	$t_{\max} = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}} = 3.628$
백분률 오버슈트 :	Percent maximum overshoot = $100e^{-\pi\zeta/\sqrt{1-\zeta^2}} = 16.3\%$
지연시간 :	$t_d \cong \frac{1.1+0.125\zeta+0.469\zeta^2}{\omega_n} \quad 0 < \zeta < 1.0 = 1.27975$
상승시간 :	$t_r = \frac{1-0.4167\zeta+2.917\zeta^2}{\omega_n} \quad 0 < \zeta < 1 = 1.5209$
정정시간 (5%) :	$t_s \cong \frac{3.2}{\zeta\omega_n} \quad 0 < \zeta < 0.69 = 6.4$

① 최대 오버슈트 시간



그래프로부터 최대 오버슈트 시간 t_{\max} 는 3.6325 [s]이다.

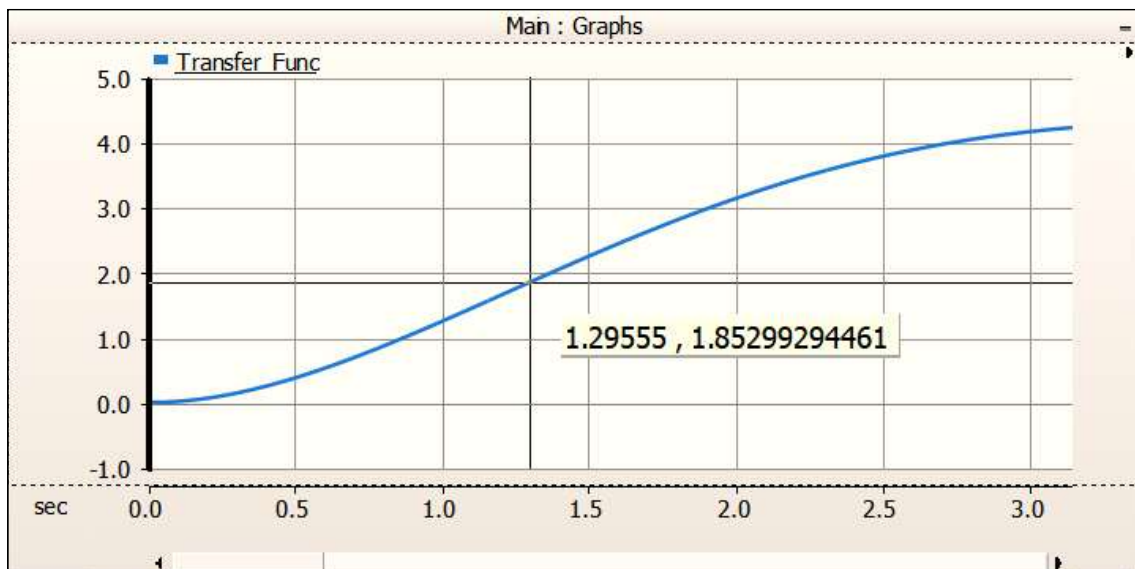
② 백분율 오버슈트



$$y_{ss} = 3.70021$$

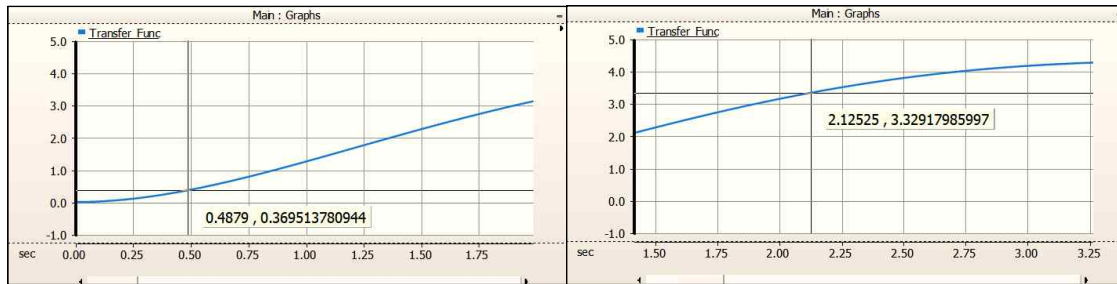
$$\begin{aligned} \text{백분율 오버슈트} &= \frac{\text{최대오버슈트}}{y_{ss}} \times 100 \\ &= \frac{4.30322 - 3.70021}{3.70021} \times 100 = 16.29964 [\%] \end{aligned}$$

③ 지연시간



지연시간은 최종값의 50%에 도달하는 시간이므로 최종값의 50%인 1.85에 도달하는 시간을 측정하였다. 따라서 $t_d = 1.29555$ [s]이다.

④ 상승시간

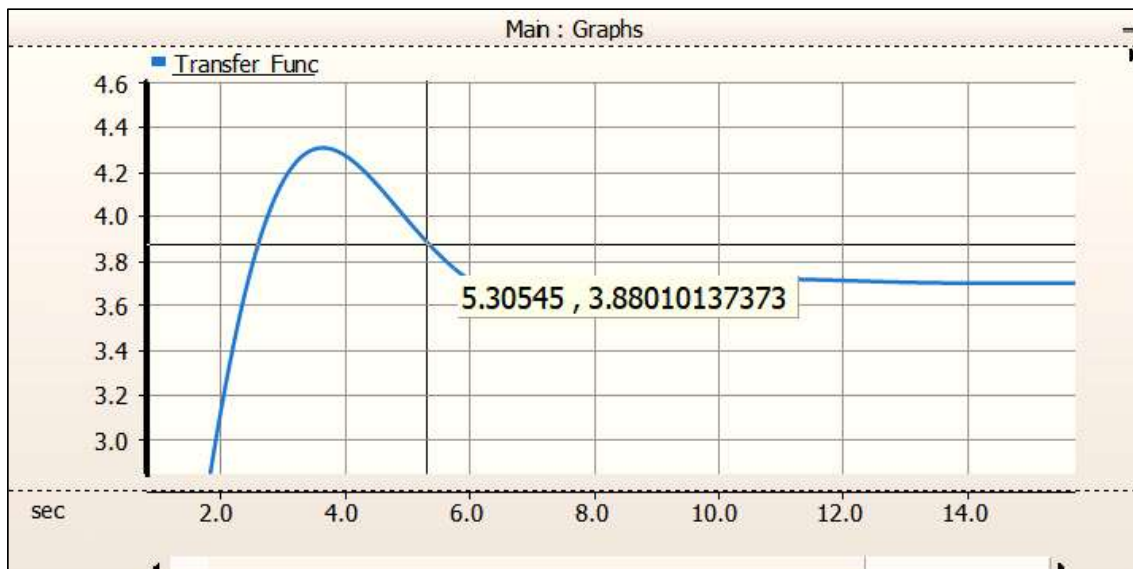


상승시간은 최종값의 10%에서 90%까지 도달하는 데 필요한 시간으로 정의한다. 최종값이 3.7이므로 10%는 0.37, 90%는 3.33이다.

그래프의 값에 따라 $t_r = 2.12525 - 0.4879 = 1.63735$ [s]이다. 근사식으로 계산한 이론값인 1.5209와 근소한 차이를 보인다.

⑤ 정정시간 (5%)

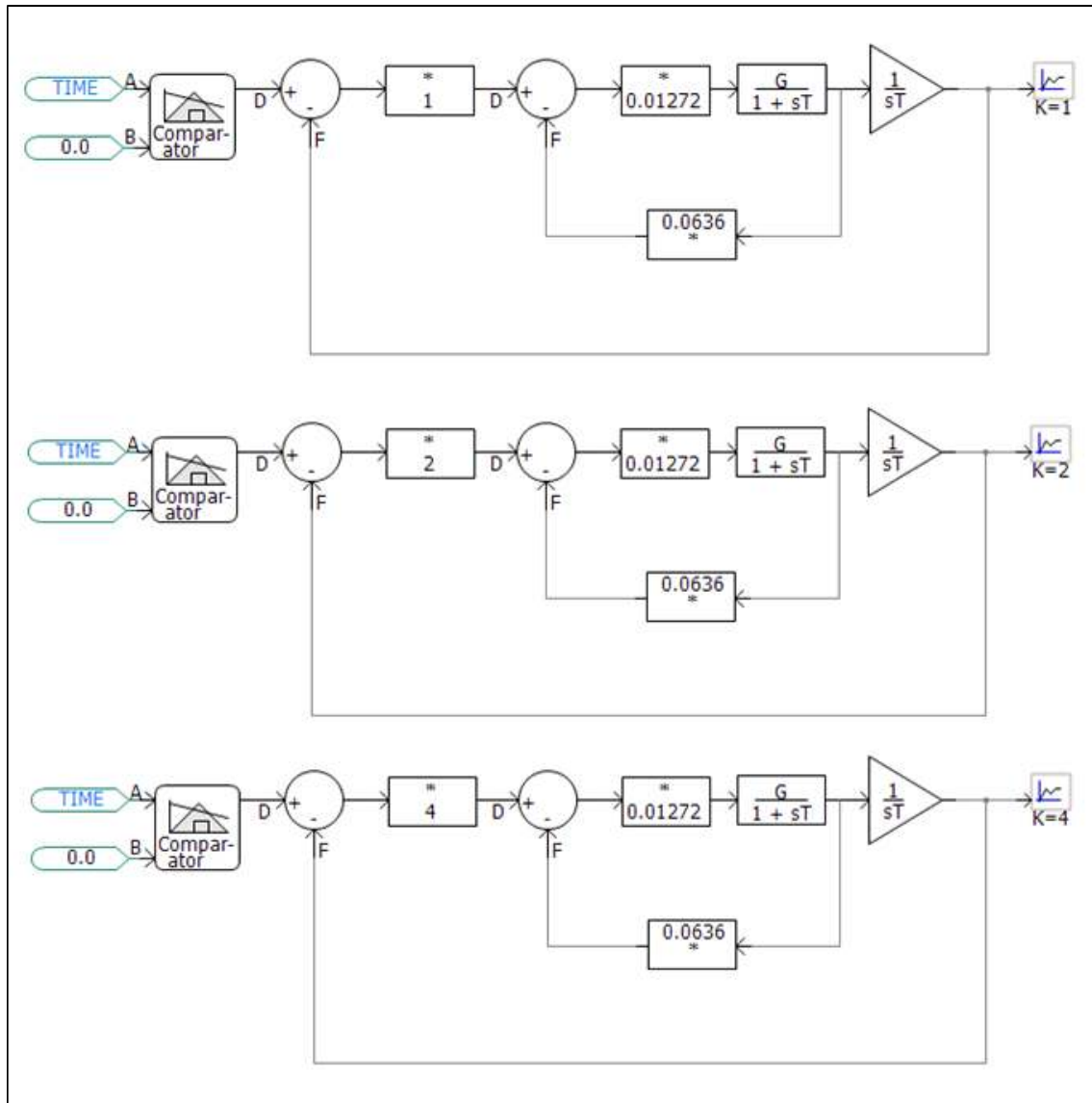
정정시간은 응답이 최종값의 5% 이내에 들어가는 데 필요한 시간이다. 최종값이 3.7이므로 응답이 3.515 ~ 3.885의 값을 가지는 시간을 측정하였다.



한편, 그래프가 최종값에 수렴하며 진동하는 동안 항상 3.515 이상의 값을 가진다. 따라서 정정시간 $t_s = 5.30545$ [s]이다. 근사식으로 계산한 이론값인 6.4와 크게 차이 나는 결과를 얻었다.

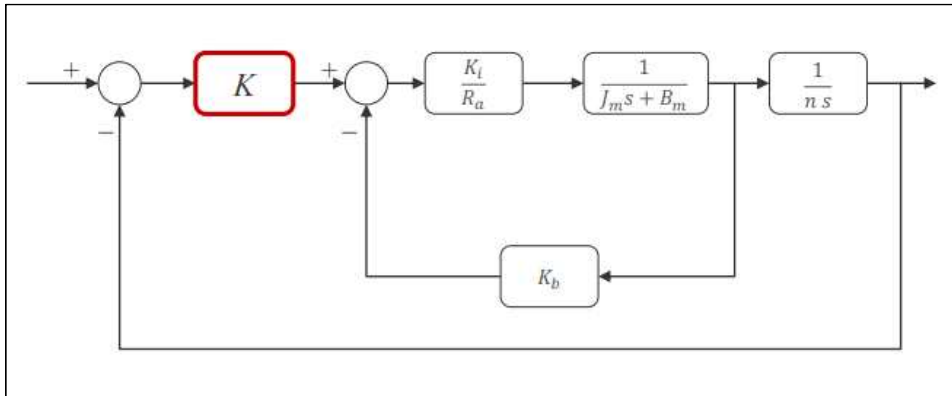
2. Example2

1) PSCAD 설계



K값을 1, 2, 4로 변화할 때 DC 모터에 대한 전달함수

2) 시뮬레이션 환경 설정



모터여자저항	$R_a = 5 [\Omega]$
모터 토크 상수	$K_i = 0.0636 [\text{Nm/A}]$
모터 역기전력 상수	$K_b = 0.0636 [\text{V/rad/s}]$
모터 회전자 관성능률	$J_m = 7.0616 \times 10^{-7} [\text{Nms}^2]$
모터의 점성마찰계수	$B_m = 3.5308 \times 10^{-5} [\text{Nms}]$
기어비	$n = 0.05$

$$\frac{K_i}{R_a} = \frac{0.0636}{5} = 0.01272$$

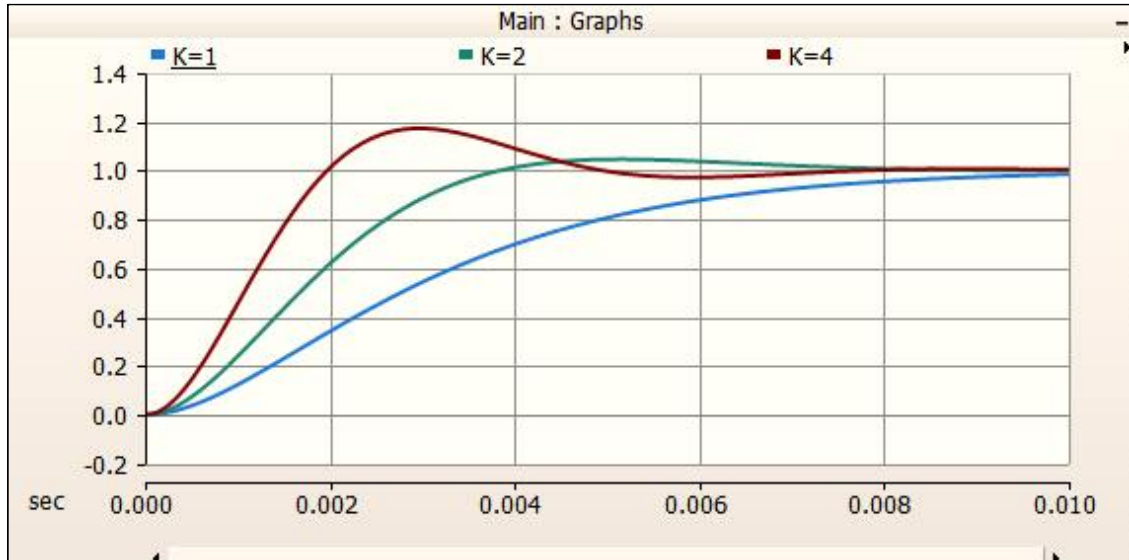
$$\frac{1}{J_m s + B_m} = \frac{1}{(7.0616 \times 10^{-7})s + (3.5308 \times 10^{-5})} = \frac{28322.19327}{1 + 0.02s}$$

$$\frac{1}{ns} = \frac{1}{0.05s}, K_b = 0.0636$$

$$K = 1, 2, 4$$

주어진 상수를 이용해 Gain과 Time Constant를 계산하여 DC 모터에 대한 전달함수를 완성하였다. Two Input Comparator은 Example1과 똑같이 설정하였다.

3) PSCAD 시뮬레이션 결과



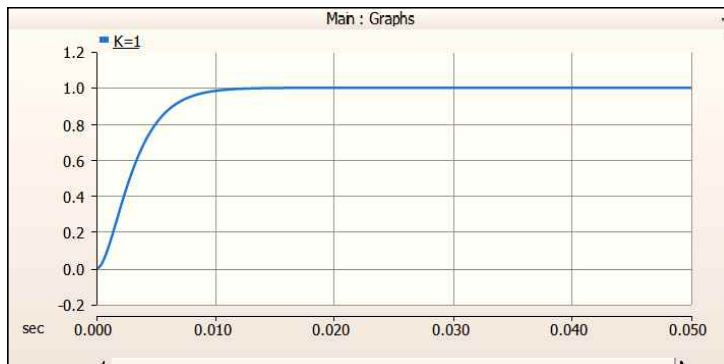
시뮬레이션을 실행하여 얻은 그래프는 다음과 같다.

4) 결과 분석 및 결론

Controller Gain K	Damping Ratio ζ	Natural Frequency ω_n	Response
$K = 1.0$	$\zeta = 1.0$		Critically damped
$K = 2.0$	$\zeta = 0.707$	$\omega_n = 844.8 \text{ rad/s}$	Underdamped Settling Time (5%) $t_s = \frac{4.5\zeta}{\omega_n} = 0.0038s$ Settling Time (2%) $t_s = \frac{4}{\zeta\omega_n} = 0.0067s$ Rise Time $t_r = \frac{1-0.416\zeta-2.917\zeta^2}{\omega_n} = 0.0026s$ Max Overshoot Time $t_{max} = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}} = 0.0053s$ %Overshoot $PO = 100e^{-\pi\zeta/\sqrt{1-\zeta^2}} = 4.3$
$K = 4.0$	$\zeta = 0.5$	$\omega_n = 1,194.8 \text{ rad/s}$	Underdamped Settling Time (5%) $t_s = \frac{4.5\zeta}{\omega_n} = 0.0054s$ Settling Time (2%) $t_s = \frac{4}{\zeta\omega_n} = 0.0067s$ Rise Time $t_r = \frac{1-0.416\zeta-2.917\zeta^2}{\omega_n} = 0.0013s$ Max Overshoot Time $t_{max} = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}} = 0.003s$ %Overshoot $PO = 100e^{-\pi\zeta/\sqrt{1-\zeta^2}} = 16.3$

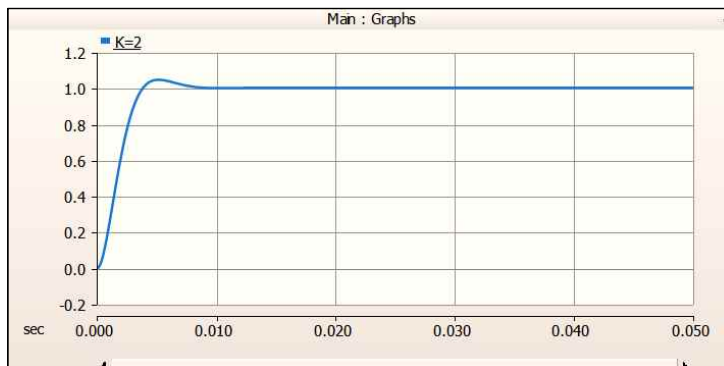
각 경우의 응답 특성의 이론값은 다음과 같다.

(1) $K = 1$



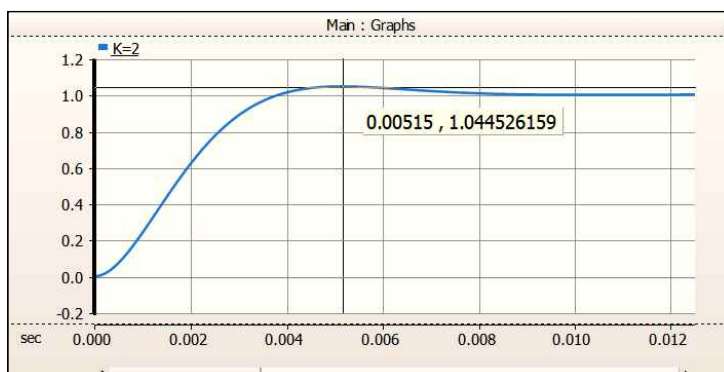
$K = 1$ 일 때, 감쇠비는 1.0이므로 Critically damped case이다. 따라서 오버슈트가 발생하지 않는 그래프 양상을 보인다.

(2) $K = 2$



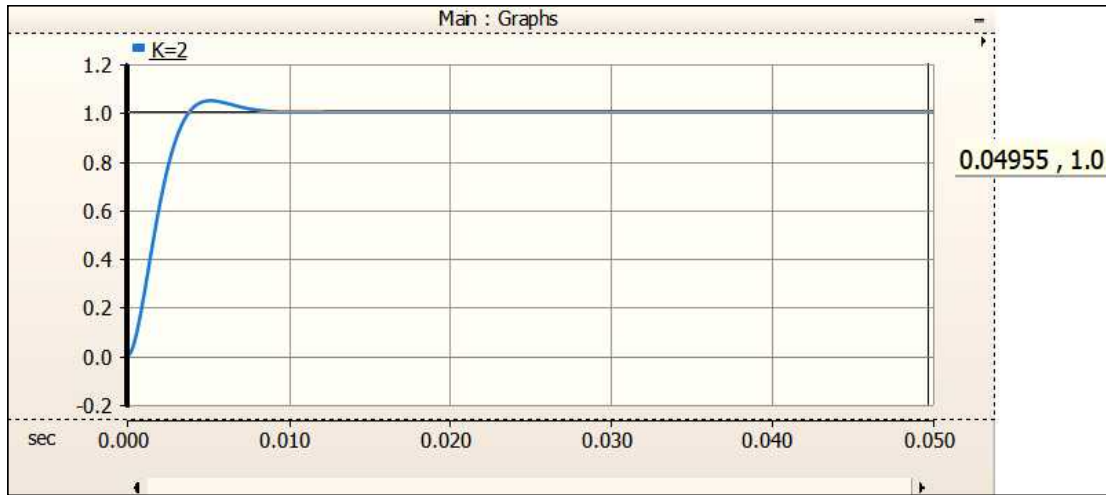
$K = 2$ 일 때, 감쇠비는 0.707이므로 Underdamped case이다. 따라서 오버슈트가 발생한다.

① Max Overshoot Time



그래프를 읽으면, $t_{\max} = 0.00515$ [s]이다.

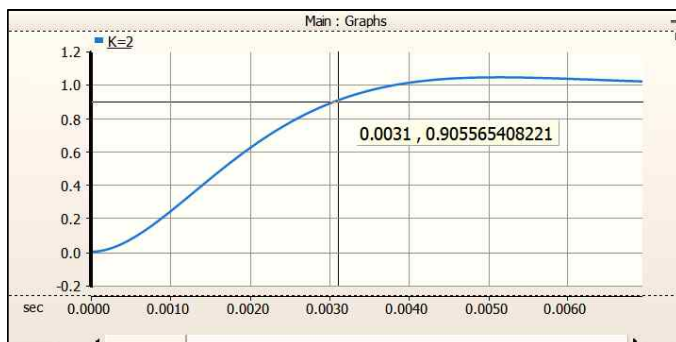
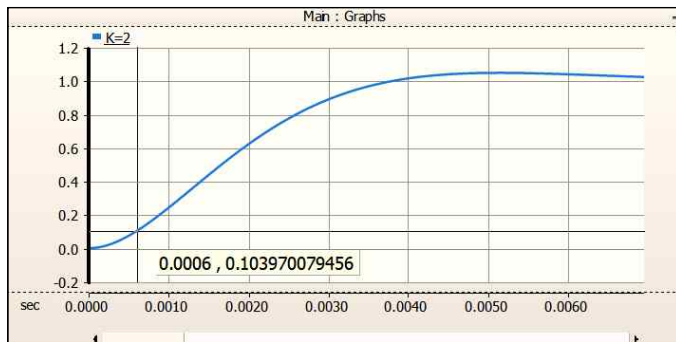
② %Overshoot



$$y_{ss} = 1.0$$

$$\begin{aligned} \text{백분율 오버슈트} &= \frac{\text{최대오버슈트}}{y_{ss}} \times 100 \\ &= \frac{1.044526 - 1.0}{1.0} \times 100 = 4.4526 [\%] \end{aligned}$$

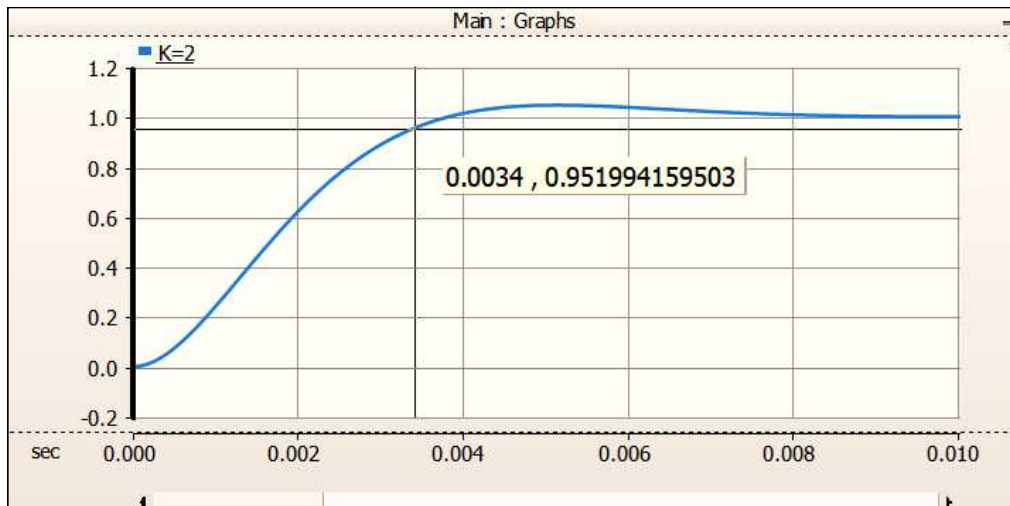
③ Rise Time



그래프의 값에 따라 계산을 하면 $t_r = 0.0031 - 0.0006 = 0.0025$ [s]이다.

④ Settling Time (5%)

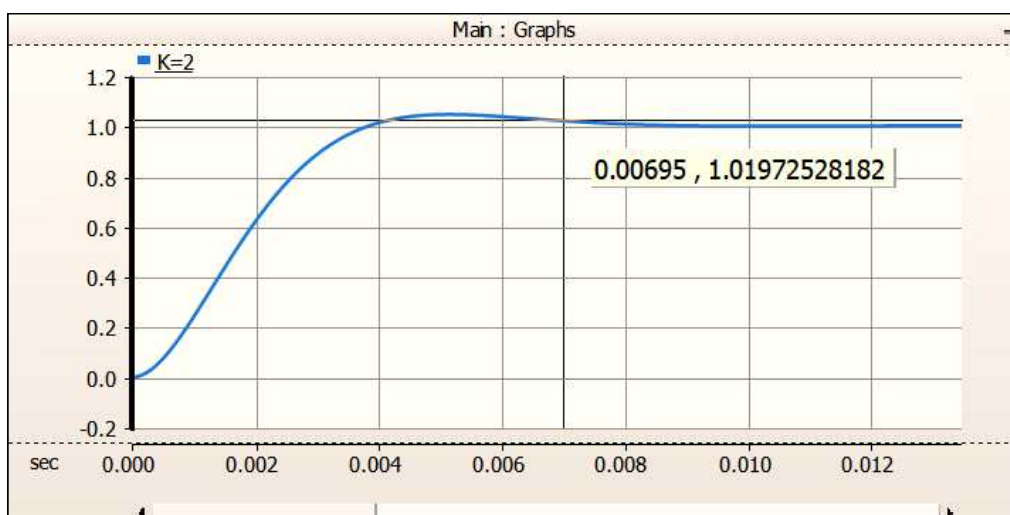
정정시간 5%는 그래프의 값이 0.95 ~ 1.05일 때이다. 그래프의 첨두값이 1.05를 넘어가지 않으므로 0.95에 도달한 시간을 측정하였다.



그래프에 따라 $t_s = 0.0034$ [s]이다.

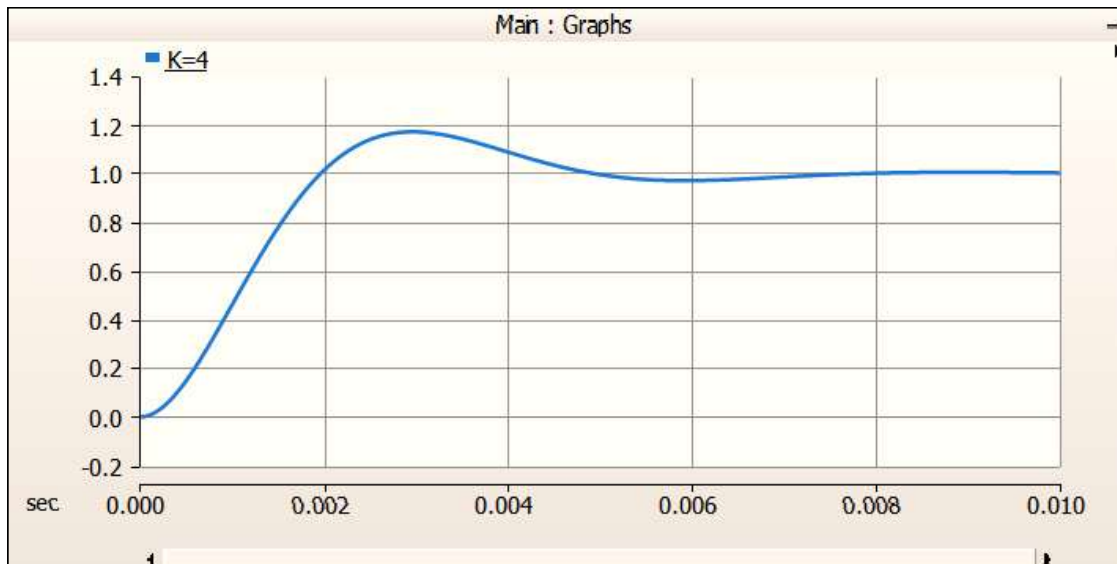
⑤ Settling Time (2%)

정정시간 2%는 그래프의 값이 0.98 ~ 1.02일 때이다. 그래프의 첨두값이 1.02를 초과하고 첨두값 이후에 0.98 이하로 낮아지는 구간이 없으므로 첨두값을 지나 1.02에 도달하는 시간을 측정하였다.



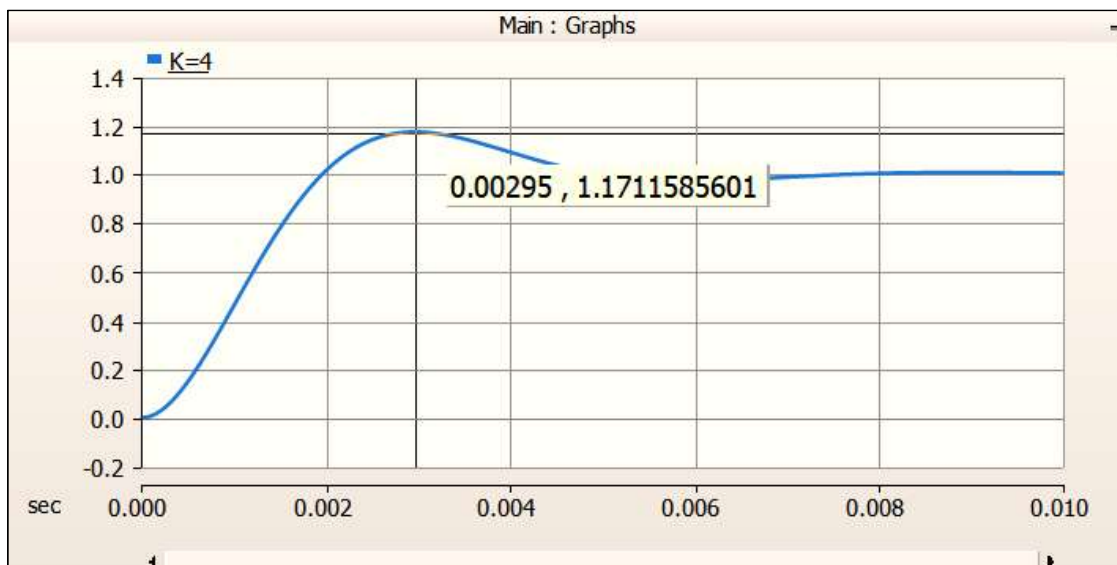
그래프에 따라 $t_s = 0.00695$ [s]이다.

(3) $K = 4$



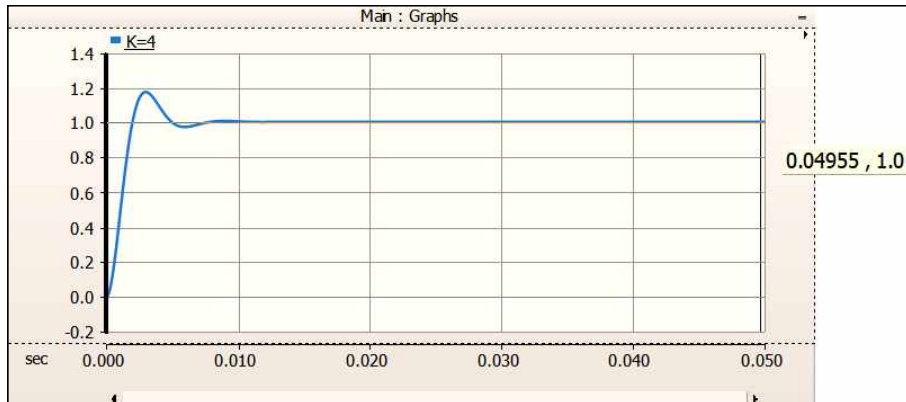
$K = 4$ 일 때, 감쇠비는 0.5이므로 Underdamped case이다. 따라서 오버슈트가 발생한다.

① Max Overshoot Time



그래프를 읽으면, $t_{\max} = 0.00295$ [s]이다.

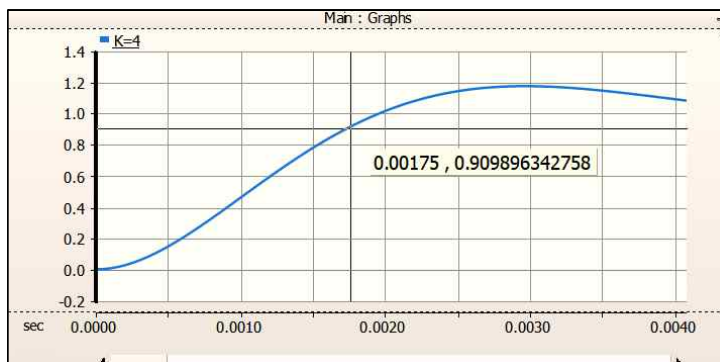
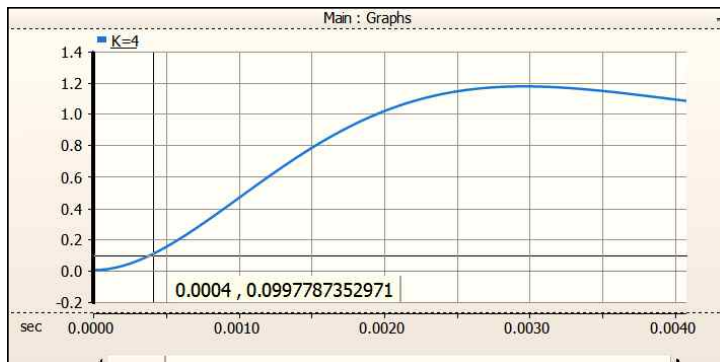
② %Overshoot



$$y_{ss} = 1.0$$

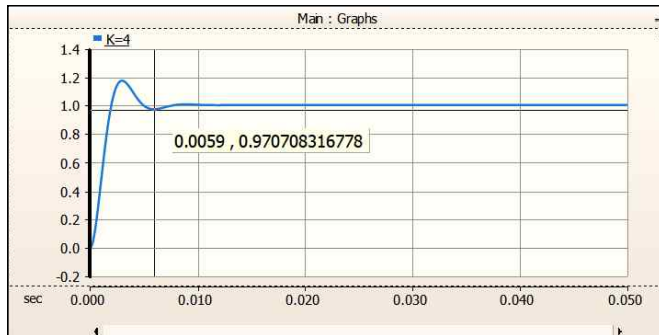
$$\begin{aligned} \text{백분율 오버슈트} &= \frac{\text{최대오버슈트}}{y_{ss}} \times 100 \\ &= \frac{1.17116 - 1.0}{1.0} \times 100 = 17.116 [\%] \end{aligned}$$

③ Rise Time



그래프의 값에 따라 계산을 하면 $t_r = 0.00175 - 0.0004 = 0.00135$ [s]이다.

④ Settling Time (5%)



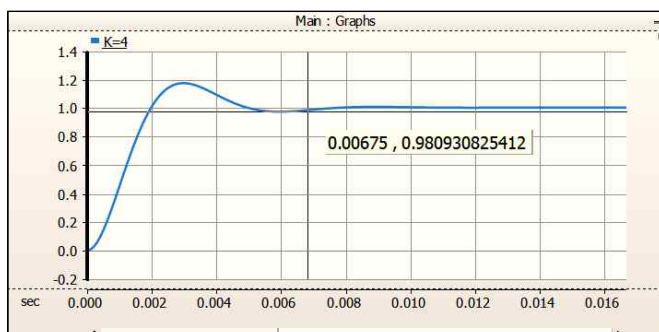
정정시간 5%는 그래프의 값이 0.95 ~ 1.05일 때이다. 첨두값을 지난 후의 최소값은 0.97로 0.95보다 크기 때문에 첨두값 이후 1.05에 도달한 시간을 측정하였다.



그래프에 따라 $t_s = 0.00435$ [s]이다.

⑤ Settling Time (2%)

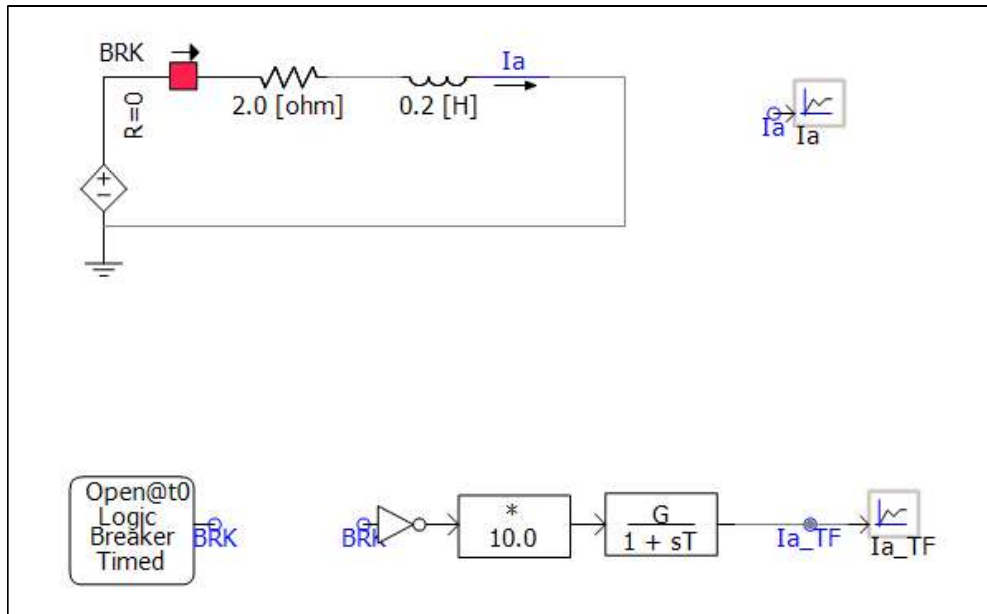
정정시간 2%는 그래프의 값이 0.98 ~ 1.02일 때이다. 첨두값 이후의 최소값을 지나 0.98 이상에 도달할 때의 시간을 측정하였다.



그래프에 따라 $t_s = 0.00675$ [s]이다.

3. Example4

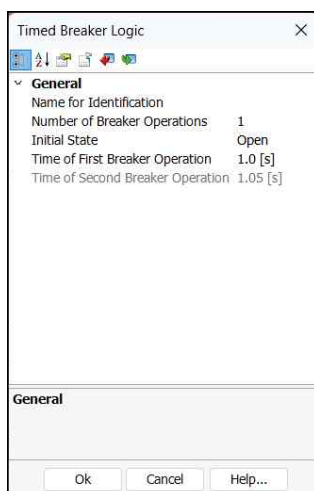
1) PSCAD 설계



PSCAD로 설계한 회로 및 전달함수

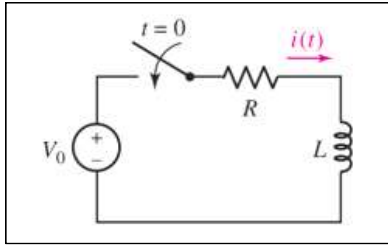
2) 시뮬레이션 환경 설정

① Time Breaker Logic 설정



Closed 상태로 만들기 위해서 Initial State를 Open으로 설정하고 Number of Breaker Operations를 1로 설정하였다.

② 전달함수의 Gain과 Time Constant

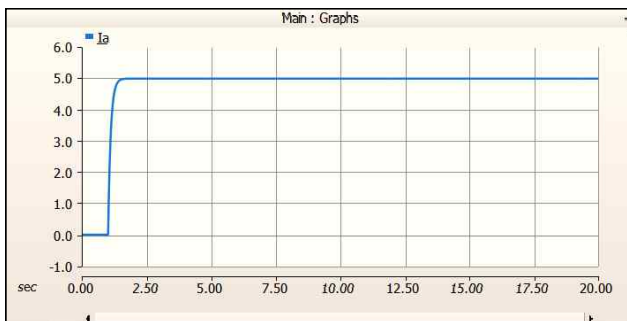


$$\begin{aligned}
 Ri + L \frac{di}{dt} &= v \\
 \frac{di}{dt} + \frac{R}{L}i &= \frac{v}{L} \\
 I(s)\left(s + \frac{R}{L}\right) &= \frac{V(s)}{L} \\
 G(s) = \frac{I(s)}{V(s)} &= \frac{1}{L} \frac{1}{s + \frac{R}{L}} = \frac{1}{R} \frac{1}{1 + s\frac{L}{R}} \leftrightarrow G \frac{1}{1 + sT} \rightarrow G = \frac{1}{R}, T = \frac{L}{R}
 \end{aligned}$$

$$\begin{aligned}
 G &= \frac{1}{2.0} = 0.5 \\
 T &= \frac{L}{R} = \frac{0.2}{2.0} = 0.1
 \end{aligned}$$

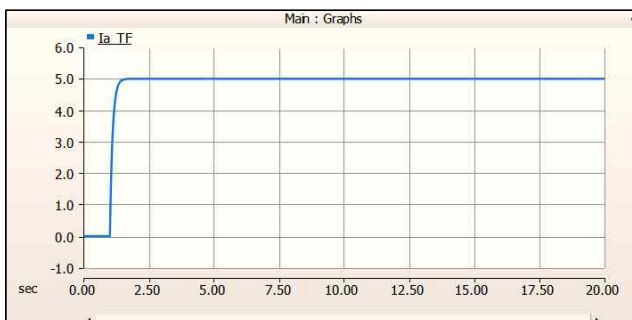
3) PSCAD 시뮬레이션 결과

① 회로에서 측정한 전류



$$I_a = 5.0 \text{ [A]이다.}$$

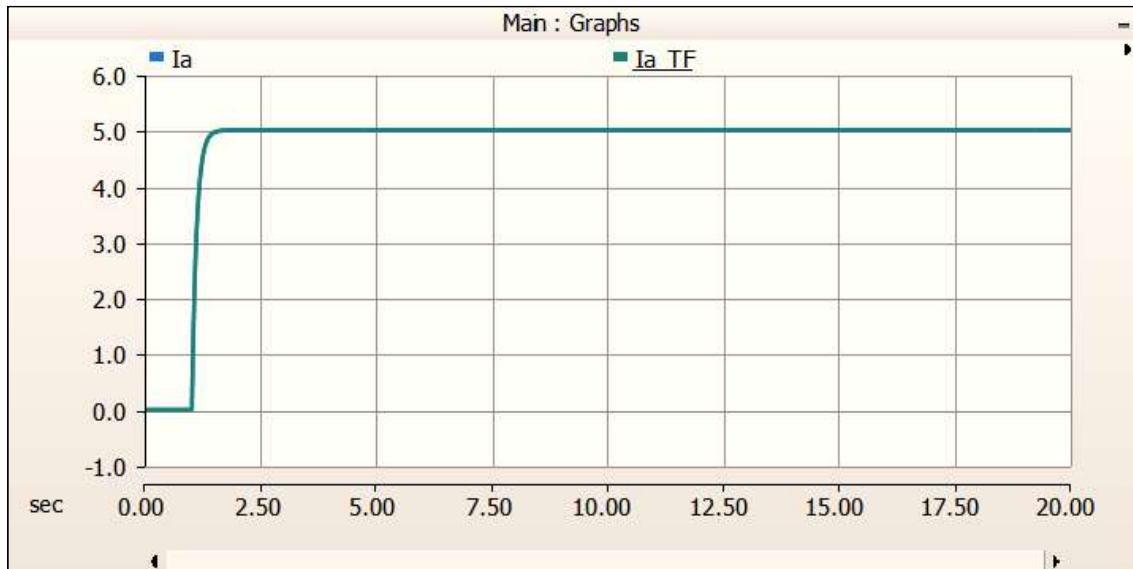
② 전달함수로 구성한 출력 (전류)



$$I_{a_{TF}} = 5.0 \text{ [A]}$$

4) 결과 분석 및 결론

회로에서 측정한 전류와 전달함수로 구성된 출력을 비교하면 다음과 같다.

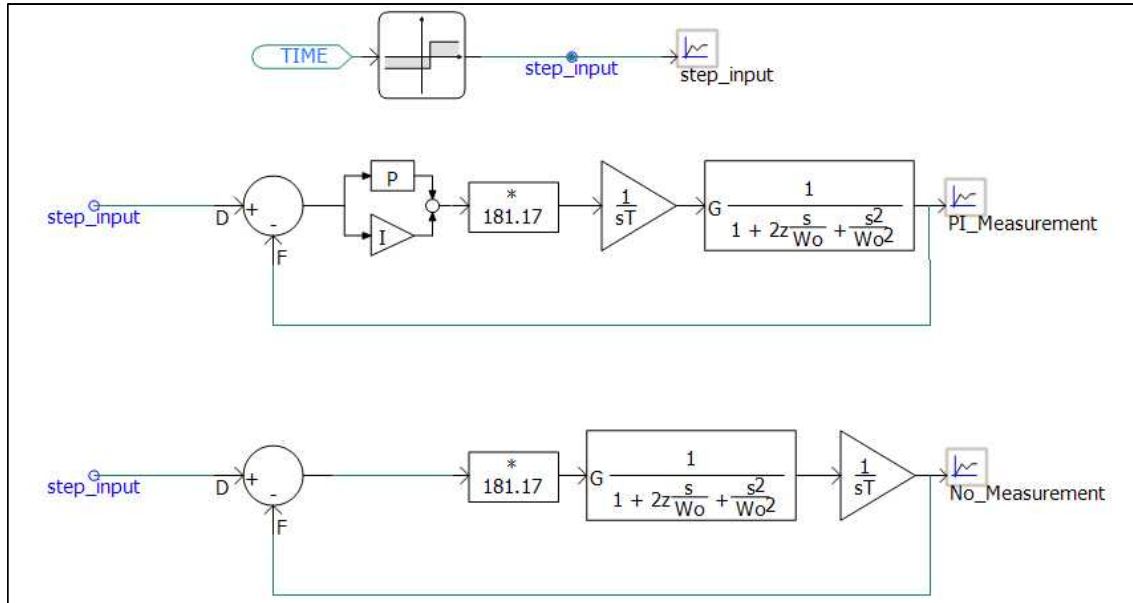


회로에서 측정한 전류와 전달함수로 구성된 출력(전류) 그래프는 정확하게 일치한다.

따라서 회로에서 얻은 전달함수를 통하여 실제 값과 똑같은 결과가 나옴을 알 수 있었다.

4. Example4

1) PSCAD 설계

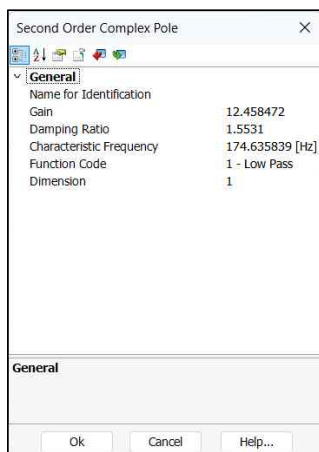


비행기 자세 제어 시스템 모델 설계

2) 시뮬레이션 환경 설정

① 감쇠비와 고유주파수 설정

$G(s) = \frac{1.5 \times 10^7 K}{s(s^2 + 3408.3s + 1204000)}$ 에서 다음과 같은 결과를 얻을 수 있다.



$$\omega_n^2 = 1204000, \omega_n = 1097.269338, f_n = \frac{\omega_n}{2\pi} = 174.635839 [Hz]$$

$$2\zeta\omega_n = 3408.3, \zeta = 1.5531$$

$$1.5 \times 10^7 = G \times \omega_n^2, G = 12.458472$$

② PI제어기 설정

PI 제어기 설계방법

- 0. 배경지식: PSCAD의 PI제어 블록에서 $K_p = \text{Gain}$, $\text{Time Constant} = K_p/K_i$ 이다. (TC: Time Constant)
- 1. K_p 값을 먼저 설정한 후 TC를 설정한다.
- 2. K_p 값을 먼저 설정하기 위하여 Time Constant 값을 매우 큰 값 (혹은 K_i 를 0)으로 설정한다.
- 3. Ref에 Step 함수를 적용하여 측정값과 비교한다. 측정값과 Ref값이 비슷해 지는 K_p 를 찾는다.
- 4. K_p 를 찾으면 응답속도를 개선할 수 있는 TC값을 찾는다.
- 문제해결방법을 위한 특징 이해
 - K_p 값이 너무 크면 ref에 Step을 넣었을 때 값이 갑자기 커지는 현상이 있다. (Overshoot가 발생 할 수 있다.)

문제해결방법을 위한 특징 이해

- 기본 특징
 - K_p : ref의 변화 시 측정값에 즉각적인 영향을 주는 것이 일반적이다. 변화에 즉각 영향을 준다. (K_p 라는 상수 곱해서 입력 되기 때문)
 - 너무 작은 숫자를 넣으면, 초반 출력 변화가 작고, TC에만 의지하게 된다. 너무 큰 숫자를 넣으면 초반 변화가 크기 때문에 Overshoot이 발생할 수 있다.
 - TC: 서서히 변화하여 정상상태 오차를 줄인다. TC를 최대한 작게 하는게 유리하다.
 - 변화율에 영향을 주기 때문에 너무 작은 숫자를 넣으면 Oscillation이 발생하고 너무 큰 숫자를 넣으면 변화가 작다.
- Overshoot 발생 시
 - K_p 값을 줄인다: K_p 값이 너무 크면 ref변화 시 값이 갑자기 커지는 현상이 있다. 즉, Overshoot가 발생 할 수 있다.
 - TC 값을 증가시킨다: TC값이 너무 작아도 Overshoot가 발생 할 수 있다.
- 응답이 느릴 때
 - K_p 값을 증가시킨다: K_p 값이 너무 작으면 ref변화 시 초반 변화가 매우 적다.
 - TC 값을 감소시킨다: TC값이 너무 크면 값이 변화하는데 시간이 오래 걸린다.
- Steady State에서 과도하게 흔들릴 때
 - K_p 값을 줄인다: K_p 값이 너무 크면 과도하게 흔들리는 경향이 생기기도 한다.
 - TC 값을 증가시킨다: TC값이 너무 작으면, 심하게 흔들릴 수 있다.
 - Sampling Time을 줄인다: Simulation의 Sampling Time이 크면 과도하게 흔들릴 수 있다.

최대오버슈트 5%이하

상승시간 0.01초 이하

정정시간 0.02초 이하 (5%)

다음과 같이 주어진 성능 사양을 만족하기 위한 첨두값 조건은 다음과 같이 계산할 수 있다.

$$y_{ss} = 1.0$$

$$\text{백분율 오버슈트} = \frac{\text{최대오버슈트}}{y_{ss}} \times 100$$

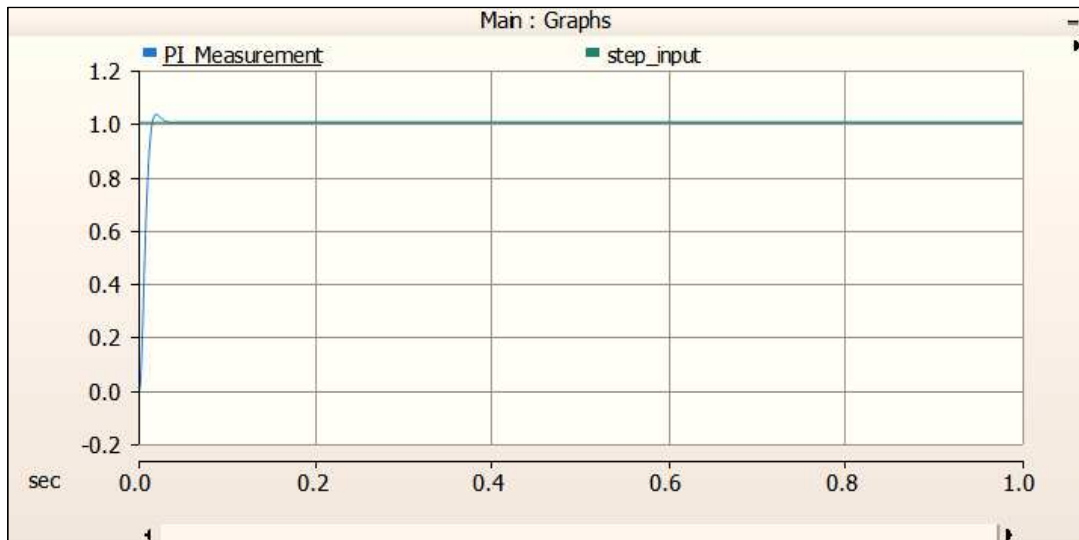
$$= \frac{y_{peak} - 1.0}{1.0} \times 100 \leq 5 [\%]$$

$$\therefore y_{peak} \leq 1.05$$

3) PSCAD 시뮬레이션 결과

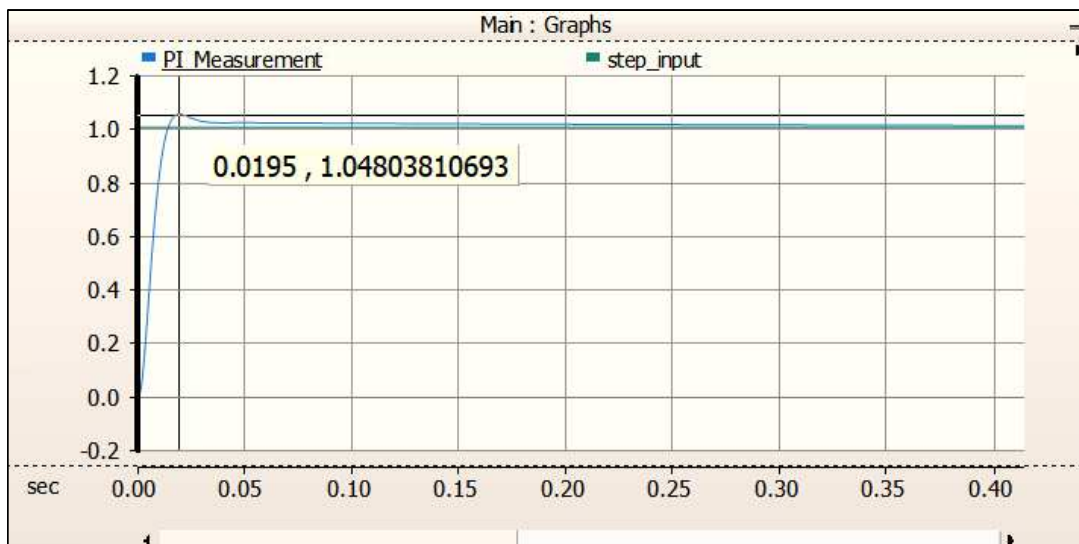
(1) PI제어기가 있을 때

PI제어기의 Time Constant를 매우 큰 수인 1000000으로 설정하고 측정값과 ref가 비슷해지는 Proportional Gain로 0.07을 얻었다.



$$K_p = 0.07, TC = 1000000$$

앞서 구한 침투값 조건을 만족하면서 최대한 Time Constant를 줄인 결과, $TC = 5.0$ 을 얻었다.

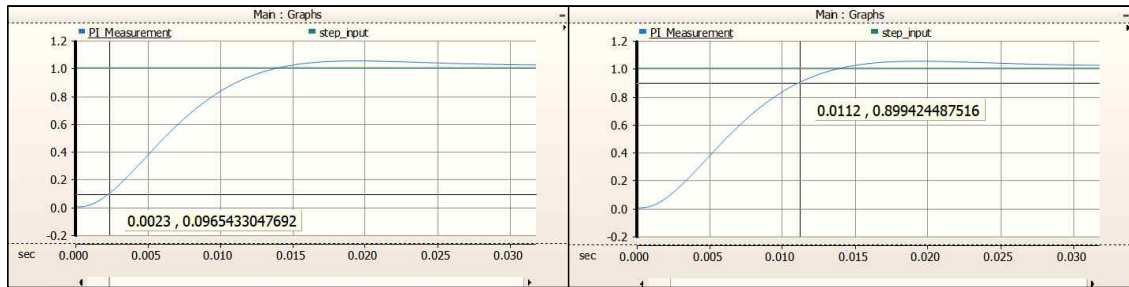


$$K_p = 0.07, TC = 5.0$$

① 백분율 오버슈트

$$\text{백분율오버슈트} = \frac{1.048 - 1.0}{1.0} \times 100 = 4.8 [\%]$$

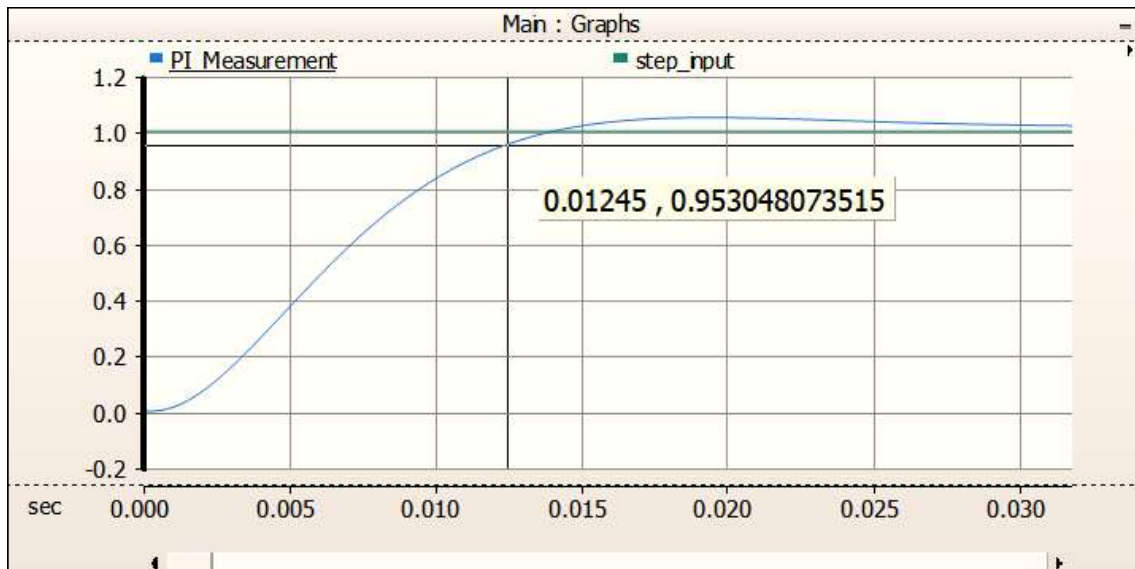
② 상승시간



그래프로부터 $t_r = 0.0112 - 0.0023 = 0.0089$ [s]이다.

③ 정정시간 (5%)

최종값이 1.0이므로 0.95까지 도달하는 데에 걸리는 시간을 측정하였다.

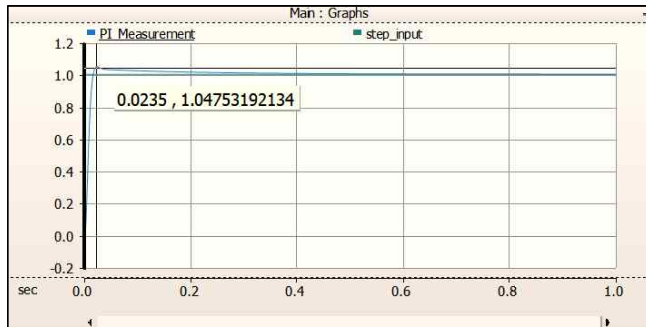


그래프로부터 $t_s = 0.01245$ [s]임을 알 수 있다.

최대오버슈트가 5% 이하, 상승시간이 0.01초 이하, 정정시간(5%)이 0.02초 이하 3가지 조건을 모두 만족하므로 PI제어기의 이득과 시정수 값이 적절하게 설정되었다.

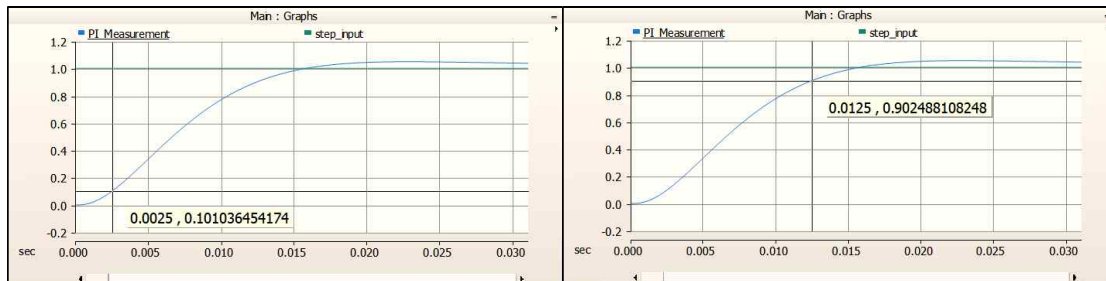
시정수 값을 줄이기 위해서 K_p 값을 조절하였고, 조건을 만족하는 결과로 $TC = 3.5$, $K_p = 0.062$ 를 얻었다.

① 백분율 오버슈트



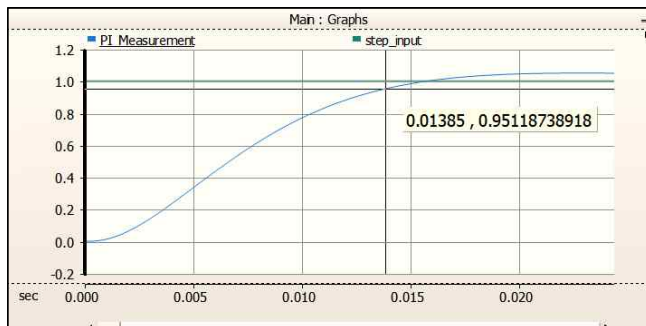
$$\text{백분율오버슈트} = \frac{1.0475 - 1.0}{1.0} \times 100 = 4.75 [\%]$$

② 상승시간



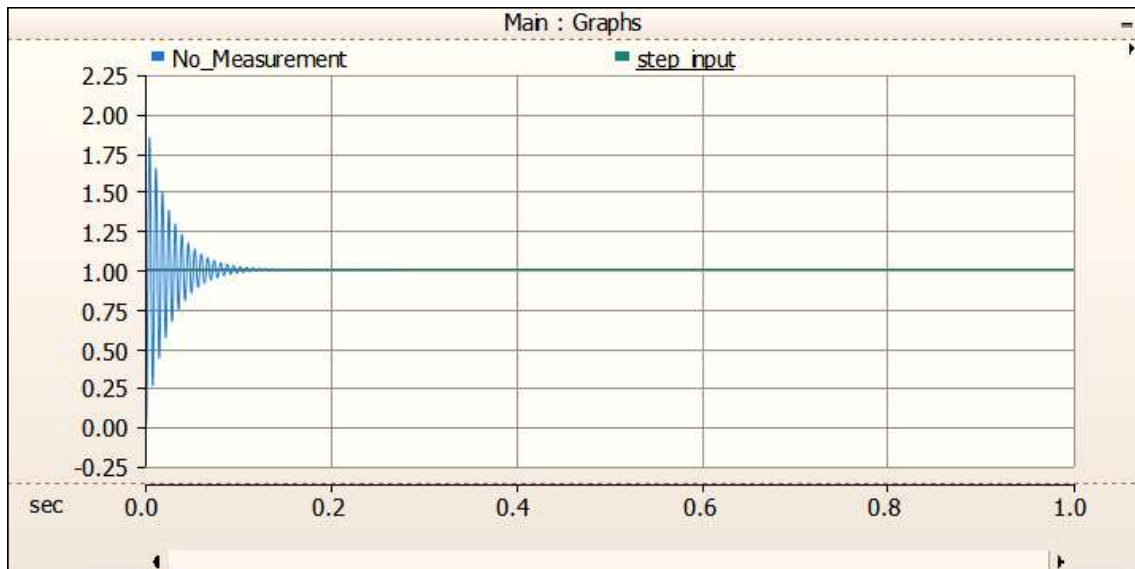
그래프로부터 $t_r = 0.0125 - 0.0025 = 0.01$ [s]이다.

③ 정정시간 (5%)



그래프로부터 $t_s = 0.01385$ [s]임을 알 수 있다.

(2) PI제어기가 없을 때

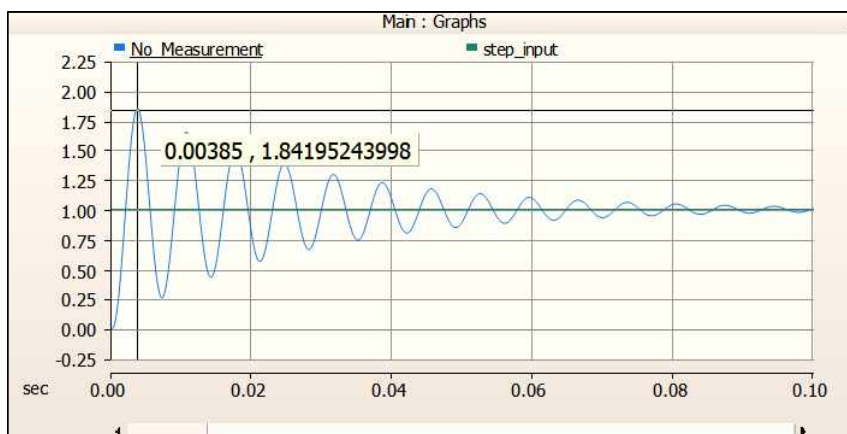


PI제어기가 없을 때의 그래프는 다음과 같다.

4) 결과 분석 및 결론

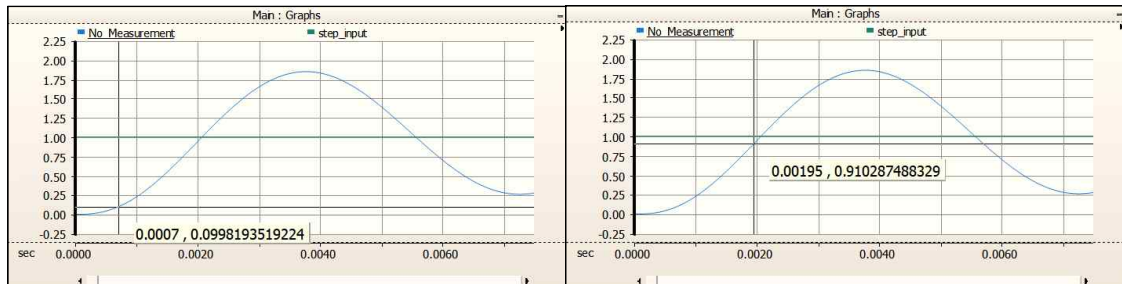
PI제어기의 유무에 따라 변화하는 그래프를 분석하기 위하여 PI제어기가 없을 때의 백분율 오버슈트, 상승시간, 정정시간(5%)을 분석하였다.

① 백분율 오버슈트



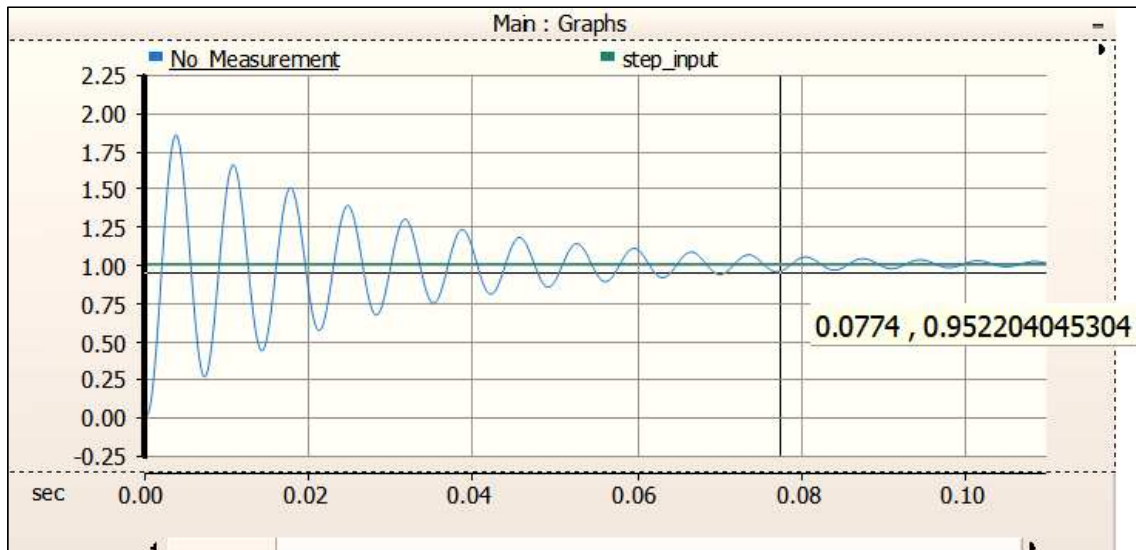
$$\text{백분율오버슈트} = \frac{1.84195 - 1.0}{1.0} \times 100 = 84.195 [\%]$$

② 상승시간



그래프로부터 $t_r = 0.00195 - 0.0007 = 0.00125$ [s]이다.

③ 정정시간 (5%)

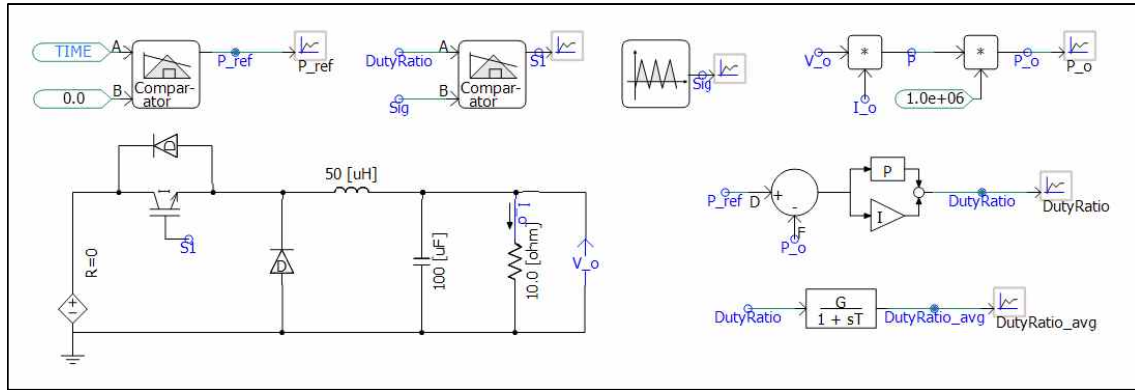


그래프로부터 $t_s = 0.0774$ [s]임을 알 수 있다.

따라서 PI제어기를 추가 설치하면 수렴하는 과정에서 진동이 줄어들고, 백분율 오버슈트가 작아지며 정정시간이 줄어들어 원하는 값에 더 빠르게 도달하는 결과를 얻을 수 있다.

5. Example5

1) PSCAD 설계



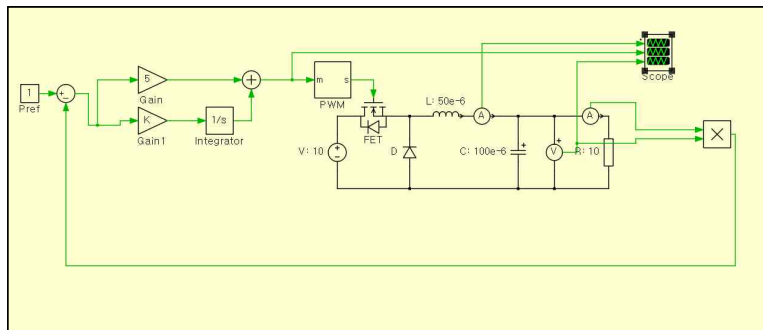
Buck-Converter와 PI제어기 PSCAD 설계

2) 시뮬레이션 환경 설정

DC-DC 컨버터의 조건은 3번째 설계 보고서의 Buck-Converter와 같다. Signal Generator의 주파수는 100[kHz]로 값이 매우 크기 때문에 정확한 데이터를 얻기 위하여 Solution Time Step과 Channel Plot Step을 0.1[μs]로 낮춰주었다. Buck-Converter의 입력 전압 V_i 는 100[V]이다.

Duration of Run (s)	0.1
Solution Time Step (us)	0.1
Channel Plot Step (us)	0.1
Runtime	

Buck Converter 자체가 전달함수이고 이 때의 입력이 Duty Ratio, 출력이 Load Power이면, 출력 Load Power(P_{mea})가 기준 Power(P_{ref})에 수렴하도록 PI제어기를 통해 Duty Ratio를 제어하는 것이 목표이다.

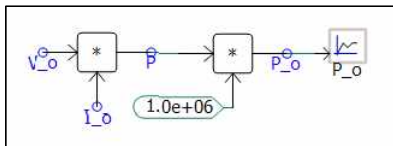


제어기 전체 모습 (PLECS)

3) PSCAD 시뮬레이션 결과

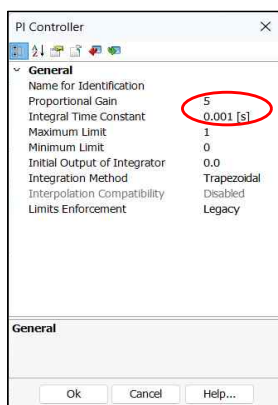
3번째 설계 실습 때 Buck-Converter의 Duty Ratio를 0.6으로 설정하고 시뮬레이션한 결과는 다음과 같은 결과를 얻을 수 있었다. $V_o = 60[V]$, $I_o = 6[A]$ $\therefore P_o = 360[W]$

P_{ref} 에는 임의의 값이 들어갈 수 있으므로 $P_{ref} = 360[W]$ 로 설정한 후 P_o 가 P_{ref} 의 값에 수렴해갈 때 Duty Ratio가 0.6이 되는지 역으로 확인해보는 과정을 수행하였다.

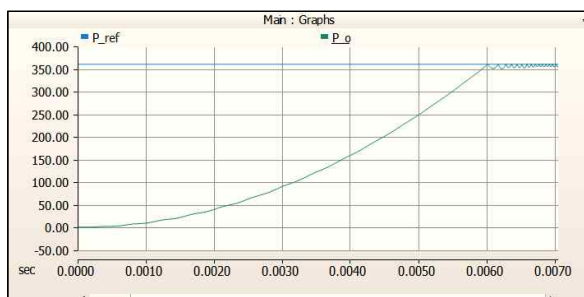


V_o 와 I_o 의 기본 단위는 각각 kV, kA 이므로 P에 저장되는 결과는 실제값의

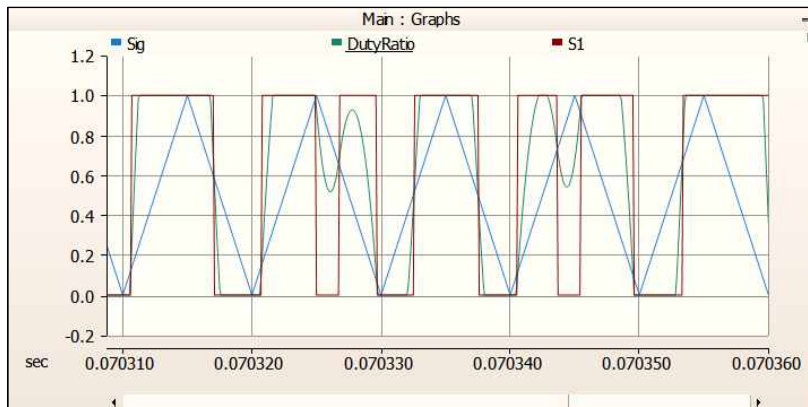
$\frac{1}{(10^{-3})^2} = \frac{1}{10^{-6}}$ 배가 된다. 따라서 10^6 을 곱하여 실제 P_o 값을 얻었다.



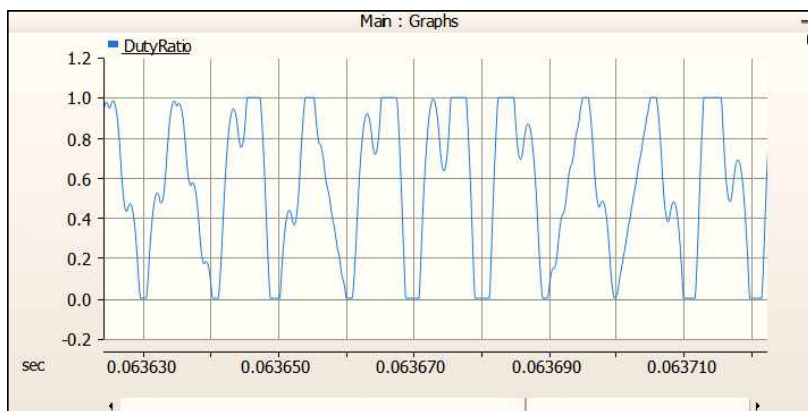
P_o 가 P_{ref} 의 값에 수렴하도록 PI제어기의 Gain과 Time Constant를 각각 1000, 0.001[s]로 조절하였다.



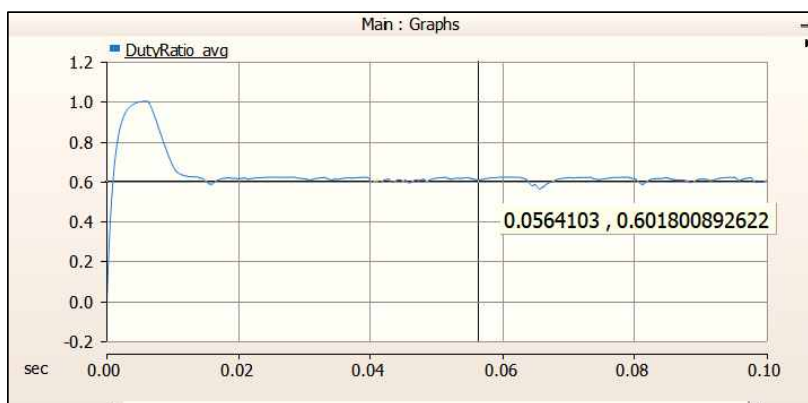
P_o 의 파형 그래프는 P_{ref} 의 값에 수렴하는 것을 알 수 있다.



S1은 $\text{DutyRatio} > \text{Sig}$ 일 때 1이고 이외의 경우 0이다.



P_o 가 P_{ref} 의 값에 가까워질 수 있도록 Duty Ratio가 실시간으로 계속 변화하는 모습을 나타낸 그래프이다.

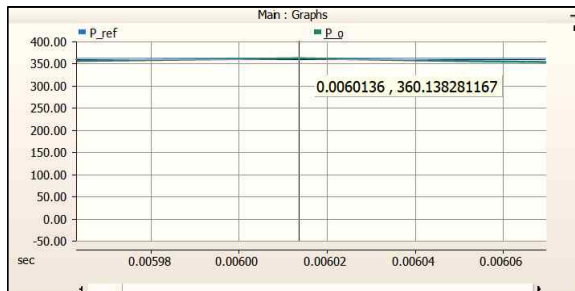


P_o 가 $P_{ref} = 360[W]$ 에 수렴하기 위한 Duty Ratio는 0.6임을 저번 설계 실습 내용을 통해 알 수 있는데, Low Pass Filter를 통해 Duty Ratio의 평균값을 구한 결과 0.6이 나옴을 확인하였다.

4) 결과 분석 및 결론

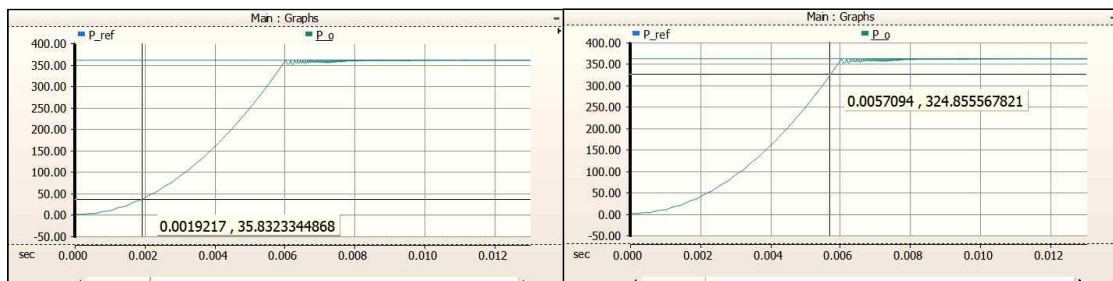
(1) PI제어기의 Gain = 5, Time Constant = 0.001[s]일 때

① 최대오버슈트 (5% 이하)



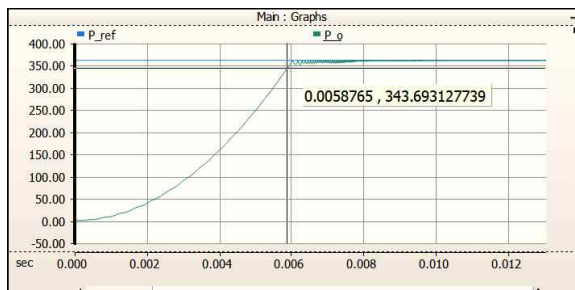
$$\text{백분율오버슈트} = \frac{360.13828 - 360}{360} \times 100 = 0.03841 [\%]$$

② 상승시간 (0.01초 이하)



그래프로부터 $t_r = 0.00571 - 0.00192 = 0.00379[s]$ 이다.

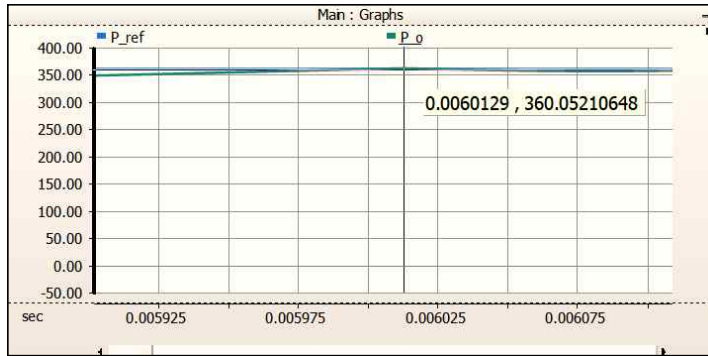
③ 정정시간(5%) (0.03초 이하)



그래프로부터 $t_s = 0.0058765 [s]$ 임을 알 수 있다.

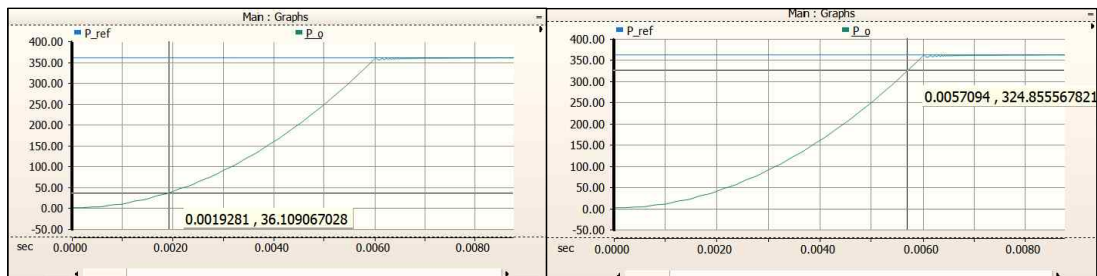
(2) PI제어기의 Gain = 1000, Time Constant = 5[s]일 때

① 최대오버슈트 (5% 이하)



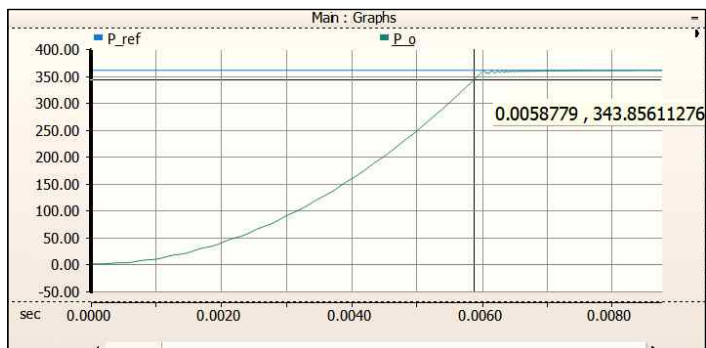
$$\text{백분율오버슈트} = \frac{360.0521 - 360}{360} \times 100 = 0.01447 [\%]$$

② 상승시간 (0.01초 이하)



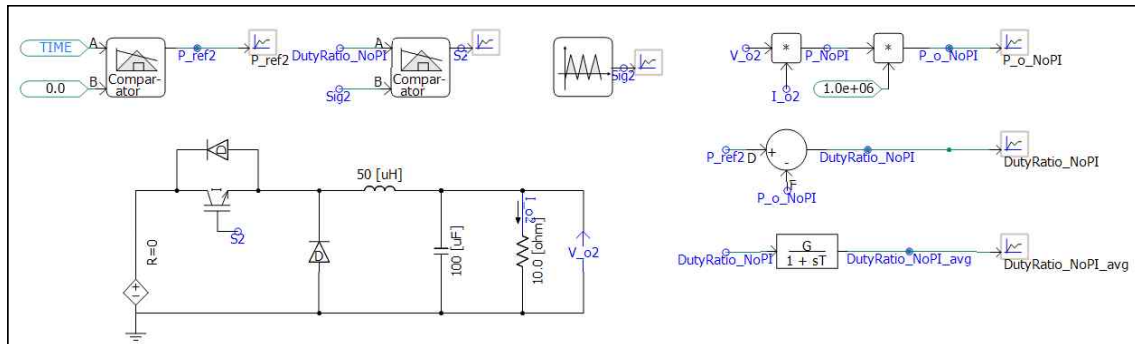
그래프로부터 $t_r = 0.00571 - 0.00193 = 0.00378[s]$ 이다.

③ 정정시간(5%) (0.03초 이하)

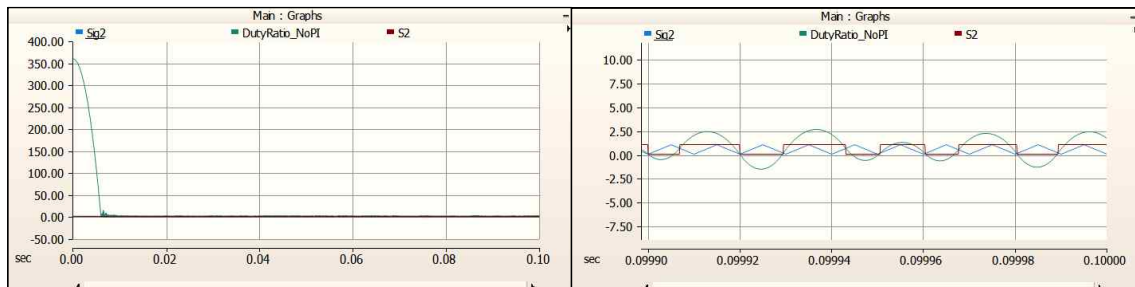


그래프로부터 $t_s = 0.0058779 [s]$ 임을 알 수 있다.

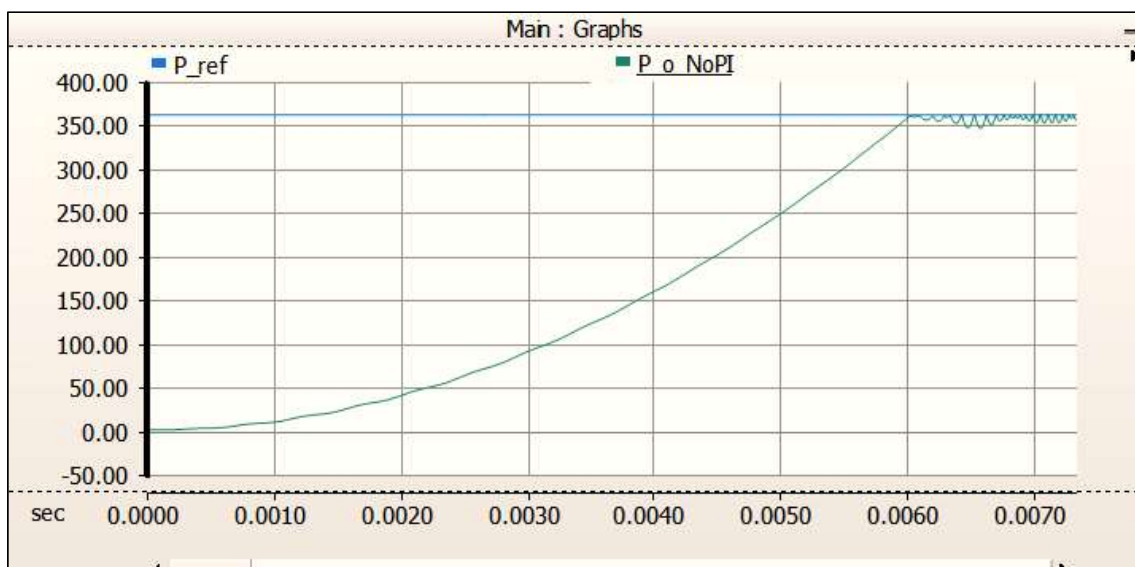
(3) PI제어기가 없을 때



Duty Ratio가 0에서 1 사이의 값을 갖게 제어할 수 있는 요소가 없기 때문에 Duty Ratio가 1 이상의 값도 가지면서 비교적 심하게 요동치는 모습을 확인할 수 있었다.



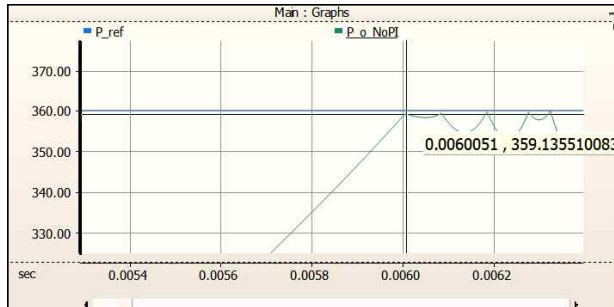
PI제어기가 없을 때의 P_o 도 P_{ref} 값에 수렴한다.



PI제어기가 없을 때의 P_o 파형은 다음과 같다.

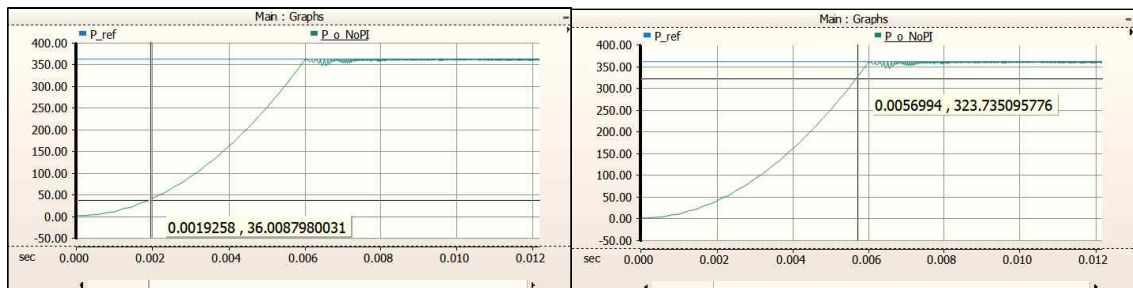
PI제어기가 있을 때와 비교하기 위하여 최대오버슈트, 상승시간, 정정시간을 분석하였다.

① 최대오버슈트 (5% 이하)



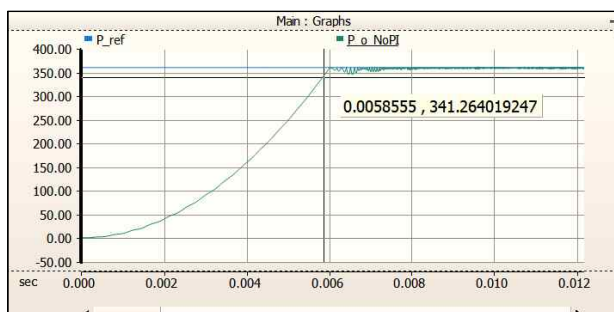
파형의 최댓값이 360을 넘지 않으므로 최대오버슈트는 발생하지 않는다.

② 상승시간 (0.01초 이하)



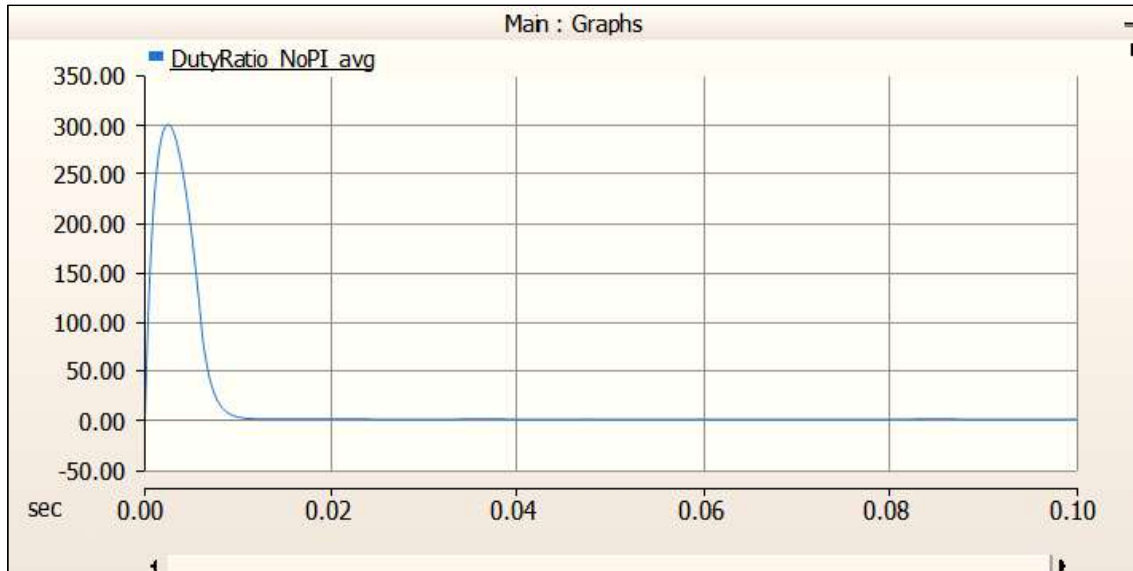
그래프로부터 대략 $t_r = 0.0056994 - 0.0019258 = 0.0037736[s]$ 이다.

③ 정정시간(5%) (0.03초 이하)

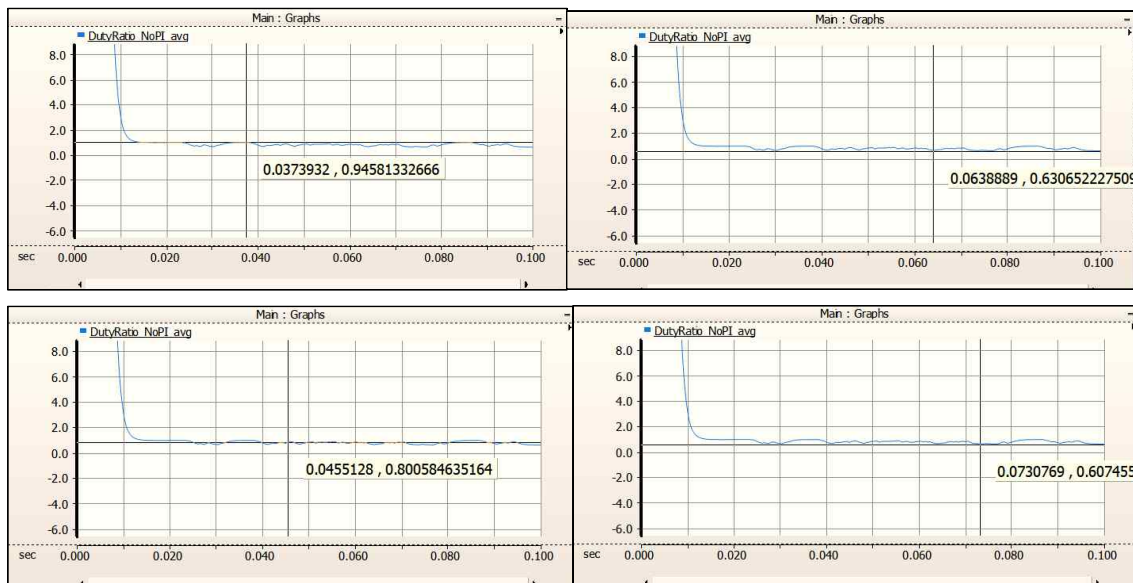


그래프로부터 대략 $t_s = 0.0058555 [s]$ 임을 알 수 있다.

PI제어기가 없을 때 Duty Ratio의 평균값을 구하기 위해 Low Pass Filter를 적용하여 얻은 결과 그래프를 출력하면 다음과 같다.



우선, Duty Ratio가 0과 1 사이로 제어되지 않아 약 300까지 급격하게 치솟고 정상값을 향해 다시 급격하게 낮아지는 현상이 나타났다.



또한 Duty Ratio는 PI제어기가 있을 때보다 0.6 부근에서 비교적 더 큰 오차를 가지며 진동한다. 이러한 현상은 P_o 가 P_{ref} 에 정밀하게 수렴하는데 방해하는 요소가 된다.

결과적으로 PI제어기가 설치된 경우에서 PI제어기가 없을 때보다 더 정확한 결과를 얻을 수 있었다.