

4. 제어 시스템의 성능

4.7 실험 실습: Lab 4

4.7.1 2차 동적 시스템의 실시간 시뮬레이션

이 실험에서는 2차 시스템의 실시간 시뮬레이션을 위해서 전달 함수가 다음과 같은 시스템을 마이크로컨트롤러를 이용해서 구현하고 계단 응답을 관찰한다.

$$\frac{Y(s)}{U(s)} = \frac{1}{(s/\omega_n)^2 + (2\zeta s/\omega_n) + 1} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (4.220)$$

위의 전달 함수의 관계식은 다음과 같은 2차 미분 방정식으로 전환이 가능하다.

$$\frac{d^2 y(t)}{dt^2} + 2\zeta\omega_n \frac{dy(t)}{dt} + \omega_n^2 y(t) = \omega_n^2 u(t) \quad (4.221)$$

Lab2 의 1차 시스템의 실시간 시뮬레이션과 마찬가지로 미분 방정식은 컴퓨터에서 구현할 수 있는 디지털 형태로 변환되어야 한다. 여러 가지 방법이 가능할 수 있지만 상태 변수 방정식을 이용한 방법이 비교적 쉽고 간단하다. 다음과 같이 상태 변수를 정의해서 상태 변수 방정식을 만든 후 디지털 형태로 변환한다.

$$x_1(t) = y(t), \quad x_2(t) = \dot{y}(t) \quad (4.222)$$

위의 상태 변수 정의를 이용해서 2차 미분 방정식을 다음과 같이 2개의 1차 미분 방정식으로 바꿀 수 있다. 아래 식은 상태 변수 방정식이다.

$$\begin{aligned} \dot{x}_1(t) &= x_2(t) \\ \dot{x}_2(t) &= -\omega_n^2 x_1(t) - 2\zeta\omega_n x_2(t) + \omega_n^2 u(t) \end{aligned} \quad (4.223)$$

만약 위의 식으로 나타낸 시스템의 입력 함수 $u(t)$ 를 미리 알 수 있다면 위의 미분 방정식의 해를 수식의 형태로 구해서 $x_1(t)$ 과 $x_2(t)$ 를 구할 수 있다. 그러나 실제 시스템에서는 입력에 어떤 신호가 들어올지 미리 알지 못하는 경우가 있을 수 있으므로, 미분 방정식의 해를 수식의 형태로 구하는 것이 가능하지 않을 수 있다. 그러나 만약 컴퓨터를 이용해서 위의 미분 방정식을 실시간으로 구현한다면 임의의 입력 신호에 대해서 각 변수가 어떻게 응답하는지 관찰하는 것이 가능하다. 위의 식의 양변에 적분을 취하면 다음 식과 같다.

$$\begin{aligned}x_1(t) &= \int_0^t x_2(\tau) d\tau \\x_2(t) &= \int_0^t \left[-\omega_n^2 x_1(\tau) - 2\zeta \omega_n x_2(\tau) + \omega_n^2 u(\tau) \right] d\tau\end{aligned}\tag{4.224}$$

미분 방정식의 해를 구하는 것은 위의 적분을 계산하는 것이므로, 컴퓨터를 이용한 적분을 이용해서 위의 적분식을 계산할 수 있다. 이와 같은 방법은 2장의 실시간 시뮬레이션에 관한 설명에서 다루었다. 여러 가지 방법이 있지만 2장에서 사용했던 Euler의 방법을 이용하면 다음과 같은 근사식을 이용해서 적분을 구할 수 있다. 아래의 식에서 Δt 는 샘플링 주기이다.

$$\begin{aligned}x_1(t) &= x_1(t - \Delta t) + \Delta t \cdot x_2(t) \\x_2(t) &= x_2(t - \Delta t) + \Delta t \cdot \left[-\omega_n^2 x_1(t) - 2\zeta \omega_n x_2(t) + \omega_n^2 u(t) \right]\end{aligned}\tag{4.225}$$

위의 식을 컴퓨터를 이용해서 구현하려고 시도해 보면 위의 식 형태 그대로는 구현이 어려운 것을 알 수 있다. 첫번째 식에서 $x_1(t)$ 를 구하기 위해서는 $x_2(t)$ 의 값이 필요하며, 두번째 식에서 $x_2(t)$ 를 구하기 위해서는 $x_1(t)$ 의 값이 필요하다. 따라서 위의 식을 구현하기 위해서는 위의 식을 다시 재정리해서 구현 가능한 식으로 변환하는 과정이 필요하게 된다. 그러나 Euler의 적분 근사식은 다음과 같은 형태도 가능할 수 있다. 위의 식과 달리 아래의 식을 이용할 경우 그대로 구현이 가능하다.

$$\begin{aligned}x_1(t) &= x_1(t - \Delta t) + \Delta t \cdot x_2(t - \Delta t) \\x_2(t) &= x_2(t - \Delta t) + \Delta t \cdot \left[-\omega_n^2 x_1(t - \Delta t) - 2\zeta \omega_n x_2(t - \Delta t) + \omega_n^2 u(t - \Delta t) \right]\end{aligned}\tag{4.226}$$

예를 들어서, 첫번째 식에서 $x_1(t)$ 를 구하기 위해서는 $x_2(t - \Delta t)$ 의 값이 필요하지만, $x_2(t - \Delta t)$ 의 값을 이전 샘플링에서 계산해서 메모리에 저장해 둔다면 계산식에서 즉시 사용이 가능하다. 이 실험에서는 위의 식을 이용해서 2차 시스템을 구현한다.

Lab2의 **2.7.1 A/D 컨버터** 프로젝트를 만드는 과정과 동일한 방법으로 프로젝트를 만든다. 코드 생성을 실행한 후 아래의 코드를 입력한다.

코드 4.1

```
/* USER CODE BEGIN PV */
float u,uold,wn,zeta,delt,x1,x1old,x2,x2old;
/* USER CODE END PV */
```

코드 4.2

```
/* USER CODE BEGIN 1 */
wn = 2*3.14*20;zeta = 0.2;
delt = 0.001;
/* USER CODE END 1 */
```

코드 4.3

```
/* USER CODE BEGIN 2 */
  HAL_TIM_Base_Start_IT(&htim10);
  HAL_DAC_Start(&hdac, DAC_CHANNEL_2);
/* USER CODE END 2 */
```

코드 4.4

```
/* USER CODE BEGIN Callback 0 */
int da_value, ad_value;
if (htim->Instance == TIM10) {
  HAL_ADC_Start(&hadc1);
  if (HAL_ADC_PollForConversion(&hadc1, 10000) == HAL_OK)
  {
    ad_value = HAL_ADC_GetValue(&hadc1);
  }
  u = ad_value - 2048.0;
  x1 = x1old + delt*x2old;
  x2 = x2old + delt*(-wn*wn*x1old - 2.0*zeta*wn*x2old + wn*wn*uold);
  x1old = x1;
  x2old = x2;
  uold = u;
  da_value = x1 + 2048.0;
  HAL_DAC_SetValue(&hdac, DAC_CHANNEL_2, DAC_ALIGN_12B_R, (uint32_t)(da_value));
}
/* USER CODE END Callback 0 */
```

함수 발생기에서 주파수가 2Hz, peak-to-peak 전압이 2V인 구형파(square wave)가 발생되도록 설정한다. 실행 파일을 실습용 보드에 다운로드 해서 실행한 후, 오실로스코프의 채널1은 함수 발생기 출력에, 오실로스코프 채널2는 D/A 변환기 출력에 연결한다. 그림 4-50은 오실로스코프 화면을 보여준다. 이 실험에서 $\omega_n = 2\pi \cdot 20$, $\zeta = 0.2$ 이며, 이 값을 이용하면 오버슈트의 크기와 최고점 시간(peak time)을 계산할 수 있다. 오실로스코프 화면에서 오버슈트의 크기와 최고점 시간(peak time)을 읽어서 계산한 값과 비교해 본다.

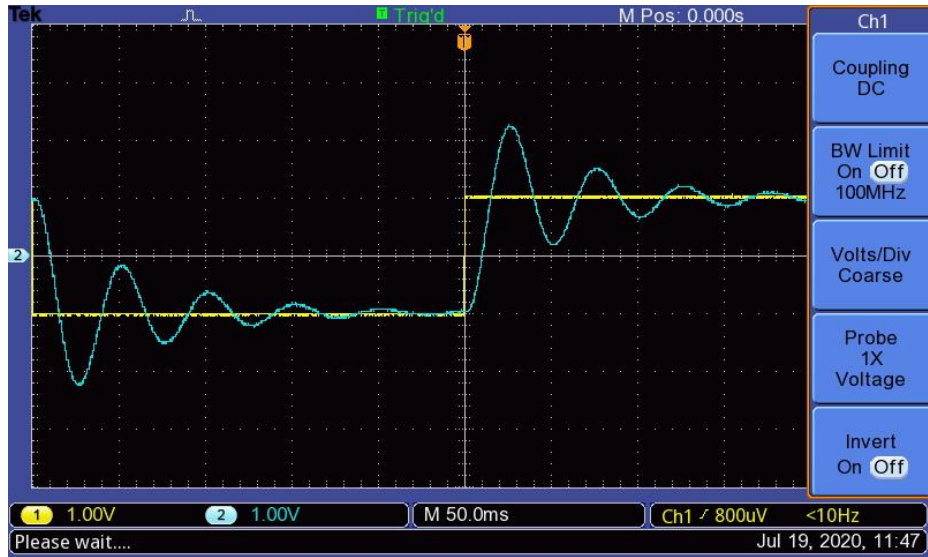


그림 4-50 오실로스코프 화면

실습 연습문제 4.1

위의 2차 시스템 실시간 시뮬레이션을 아래의 파라미터에 대해서 실시해 본 후, 오버슈트와 최고점 시간의 계산 값과 측정 값을 비교해 본다.

$$\begin{aligned}\omega_n &= 2\pi \cdot 20, \zeta = 0.5 \\ \omega_n &= 2\pi \cdot 20, \zeta = 0.8\end{aligned}\tag{4.227}$$

4.7.2 아날로그 다이내믹 시뮬레이터의 피드백 제어

일반적으로 제어 대상 시스템에서는 회전 운동이나 직선 운동 등의 움직임이 있는 경우가 많다. 제어기를 설계해서 제어 시스템을 구성하는 과정에는 제어기 프로그램의 오류 등 여러 가지 이유로 정상 작동하지 않는 경우가 발생할 수 있다. 실제 움직임이 있는 제어 시스템이 오작동 할 경우 파손되거나 위험한 상황이 발생할 수 있다. 따라서 제어기를 설계해서 실험할 경우에는 그런 위험이 없는 시스템을 구성해서 시험해 보는 과정이 필요하다.

아날로그 다이내믹 시뮬레이터는 OP 앰프를 이용해서 동적 시스템의 특성을 실시간으로 시뮬레이션 할 수 있는 회로이다. 매우 저렴한 비용으로 만들 수 있으며 파손될 위험도 적으므로 제어기를 시험하기 위한 대상으로 적합하다고 할 수 있다. 그림 4-51의 회로는 OP 앰프로 구성한 아날로그 다이내믹 시뮬레이터의 예를 보여준다.

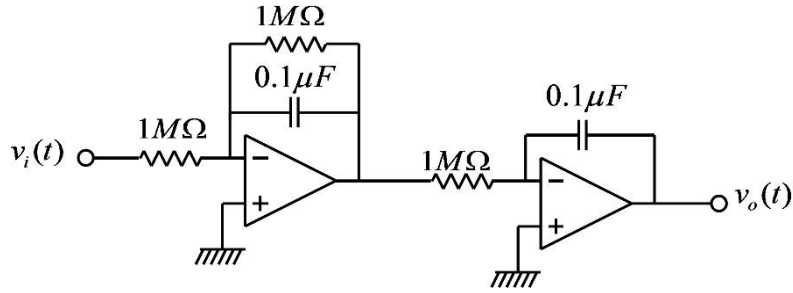


그림 4-51 아날로그 다이내믹 시뮬레이터

그림 4-51 회로의 전달 함수는 다음과 같다.

$$\frac{V_o(s)}{V_i(s)} = \left(\frac{-1}{0.1s+1} \right) \left(\frac{-10}{s} \right) = \frac{10}{s(0.1s+1)} \quad (4.228)$$

위의 전달 함수는 원점에 극점이 한 개 있으므로 불안정한 시스템이다. 따라서 피드백 제어에 의해서 적절하게 제어하지 않는다면 출력 값이 발산한다. 이 시스템에 대해서 그림 4-52와 같은 간단한 피드백 제어 시스템을 구성해 본다.

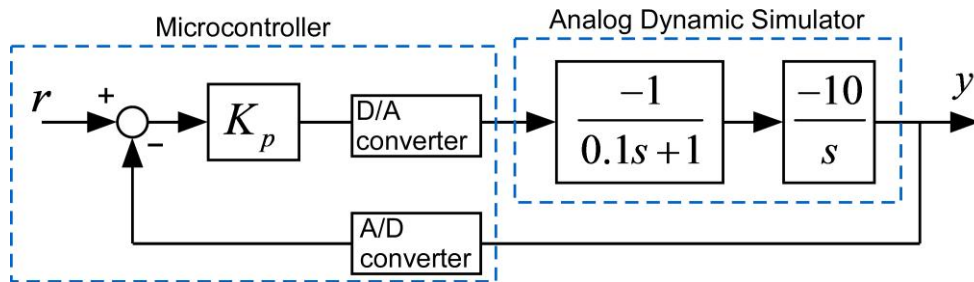


그림 4-52 아날로그 다이내믹 시뮬레이터의 피드백 제어

Lab2의 **2.7.1 A/D 컨버터** 프로젝트를 만드는 과정과 동일한 방법으로 프로젝트를 만든다. 코드 생성을 실행한 후 아래의 코드를 입력한다.

코드 4.5

```
/* USER CODE BEGIN PV */
float y,Kp,ref,control;
int interrupt_counter, sampling_frequency;
/* USER CODE END PV */
```

코드 4.6

```
/* USER CODE BEGIN 1 */
Kp=5.0;
sampling_frequency=1000;
/* USER CODE END 1 */
```

코드 4.7

```
/* USER CODE BEGIN 2 */
  HAL_TIM_Base_Start_IT(&htim10);
  HAL_DAC_Start(&hdac, DAC_CHANNEL_2);
/* USER CODE END 2 */
```

코드 4.8

```
/* USER CODE BEGIN Callback 0 */
int da_value, ad_value;
if (htim->Instance == TIM10) {
  HAL_ADC_Start(&hadc1);
  if (HAL_ADC_PollForConversion(&hadc1, 10000) == HAL_OK) {
    ad_value = HAL_ADC_GetValue(&hadc1);
  }
  y = ad_value - 2048.0;
  interrupt_counter++;
  if (interrupt_counter >= sampling_frequency*4) {
    interrupt_counter=0;
    ref=205;
  }
  if (interrupt_counter >= sampling_frequency*2) {
    ref=0;
  }
  control = Kp*(ref - y);
  if (control > 2047) control = 2047;
  if (control < -2048) control = -2048;
  da_value = control + 2048.0;
  HAL_DAC_SetValue(&hdac, DAC_CHANNEL_2, DAC_ALIGN_12B_R, (uint32_t)(da_value));
}
/* USER CODE END Callback 0 */
```

그림 4-52의 시스템에서 기준 입력(reference input) r 은 프로그램에서 생성된다. 기준 입력은 2초의 간격으로 0V의 값과 1V의 값을 교대로 발생한다. 즉, 기준 입력은 주기가 4초인 구형파(square wave)의 형태라고 볼 수 있다. 오실로스코프의 채널을 아날로그 다이내믹 시뮬레이터의 출력에 연결하고 프로그램을 실행하면 그림 4-53과 같은 계단 응답을 볼 수 있다. 그림에서 볼 수 있듯이 약간의 과도기를 지난 후 1V의 기준 입력과 0V의 기준 입력을 잘 따라가는 것을 볼 수 있다. 이 실험에서 제어기의 이득 값은 $K_p = 5$ 이다.

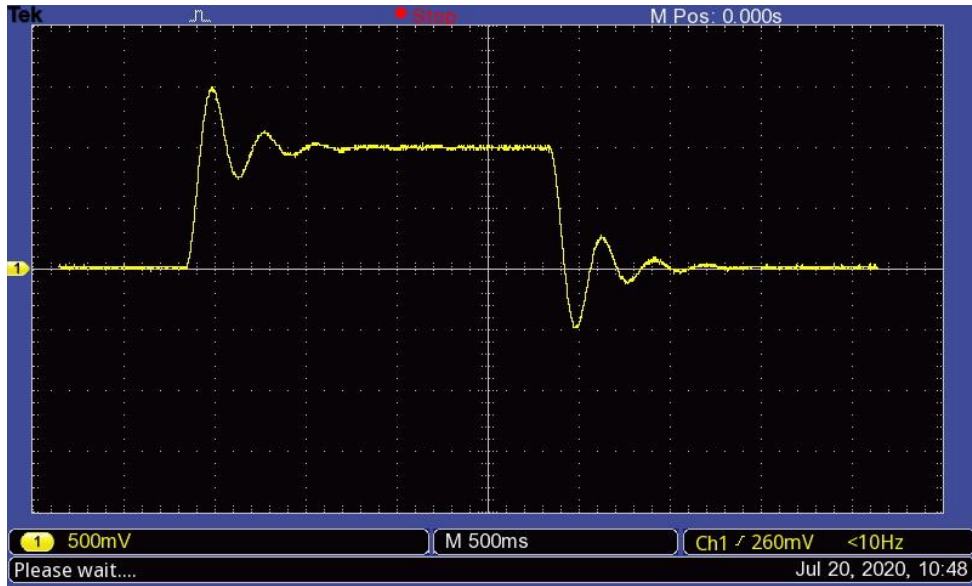


그림 4-53 페루프 시스템의 계단 응답

페루프 시스템의 전달 함수는 다음과 같다.

$$\frac{Y(s)}{R(s)} = \frac{\frac{10K_p}{s(0.1s+1)}}{1 + \frac{10K_p}{s(0.1s+1)}} = \frac{10K_p}{0.1s^2 + s + 10K_p} = \frac{100K_p}{s^2 + 10s + 100K_p} \quad (4.229)$$

위의 페루프 시스템 전달 함수와 2 차 표준 시스템의 전달 함수를 비교하면 ζ 와 ω_n 의 값을 다음과 같이 구할 수 있다.

$$\frac{100K_p}{s^2 + 10s + 100K_p} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (4.230)$$

$$\omega_n = \sqrt{100K_p}, \quad \zeta = \frac{10}{2\omega_n} = \frac{10}{2\sqrt{100K_p}} = \frac{1}{2\sqrt{K_p}} \quad (4.231)$$

식 (4.66)과 식 (4.68)을 이용하면 최고점 시간 t_p 와 오버슈트 M_p 를 계산할 수 있으며, 이 값들을 오실로스코프 화면에서 측정해서 비교해 본다.

실습 연습문제 4.2

아래의 제어 이득 값에 대해서 위의 피드백 제어 실험을 반복한다. 계산으로 얻은 최고점 시간 t_p 와 오버슈트 M_p 를 오실로스코프 화면에서 측정한 값과 비교해 본다.

$$K_p = 10, K_p = 2 \quad (4.232)$$