SIEMENS

SCE 교육 커리큘럼

Siemens Automation Cooperates with Education | 05/2017

TIA Portal Module 052-300

PID 컨트롤러

Cooperates with Education

Automation

SIEMENS

교육 커리큘럼에 따른 적합한 SCE 트레이너 패키지

SIMATIC 컨트롤러

- **SIMATIC ET 200SP Open Controller CPU 1515SP PC F 및 HMI RT SW**
  주문 번호: 6ES7677-2FA41-4AB1
- **SIMATIC ET 200SP Distributed Controller CPU 1512SP F-1 PN Safety**
  주문 번호: 6ES7512-1SK00-4AB2
- **SIMATIC CPU 1516F PN/DP Safety**
  주문 번호: 6ES7516-3FN00-4AB2
- **SIMATIC S7 CPU 1516-3 PN/DP**
  주문 번호: 6ES7516-3AN00-4AB3
- **SIMATIC CPU 1512C PN(소프트웨어 장착) 및 PM 1507**
  주문 번호: 6ES7512-1CK00-4AB1
- **SIMATIC CPU 1512C PN(소프트웨어 장착), PM 1507 및 CP 1542-5 (PROFIBUS)**
  주문 번호: 6ES7512-1CK00-4AB2
- **SIMATIC CPU 1512C PN(소프트웨어 장착)**
  주문 번호: 6ES7512-1CK00-4AB6
- **SIMATIC CPU 1512C PN(소프트웨어 장착) 및 CP 1542-5 (PROFIBUS)**
  주문 번호: 6ES7512-1CK00-4AB7

교육용 SIMATIC STEP 7 소프트웨어

- **SIMATIC STEP 7 Professional V14 SP1 - 단일 라이센스**
  주문 번호: 6ES7822-1AA04-4YA5
- **SIMATIC STEP 7 Professional V14 SP1 - 강의실 라이센스 (최대 인원 6명)**
  주문 번호: 6ES7822-1BA04-4YA5
- **SIMATIC STEP 7 Professional V14 SP1 - 업그레이드 라이센스 (최대 인원 6명)**
  주문 번호: 6ES7822-1AA04-4YE5
- **SIMATIC STEP 7 Professional V14 SP1 - 학생 라이센스 (최대 인원 20명)**
  주문 번호: 6ES7822-1AC04-4YA5

위 트레이너 패키지는 필요 시 후속 모델 패키지로 대체가 된다는 점에 유의하십시오. 현재 출시된 SCE 패키지에 대한 개요는 siemens.com/sce/tp에서 제공됩니다.

보충 교육

지멘스의 지역별 SCE 보충 교육에 대한 내용은 해당 지역의 SCE 고객 센터로 문의하시기 바랍니다.
siemens.com/sce/contact

SCE 관련 추가 정보

siemens.com/sce

사용 관련 정보

통합 자동화 솔루션인 TIA(Totally Integrated Automation)를 위한 SCE 교육 커리큘럼은 공교육 시설 및 R&D 기관 교육 목적의 "SCE(Siemens Automation Cooperates with Education) 프로그램을 위해 마련된 것입니다. Siemens AG는 프로그램의 내용을 보증하지 않습니다.

본 문서는 지멘스 제품/시스템을 초기 교육하는 용도로만 사용되어야 합니다. 따라서 교육 범위 내에서의 사용 목적으로 전체 또는 일부를 복사하여 교육생들에게 제공할 수 있습니다. 본 문서는 공공 교육 및 고등 교육 시설 내에서의 교육을 위한 목적으로의 배포, 복사 및 내용의 공유가 가능합니다.

예외적인 경우에는 Siemens AG 담당자의 서면 동의가 필요합니다. Roland Scheuerer roland.scheuerer@siemens.com.

해당 규정의 위반 시에는 그에 대한 책임이 부과될 수 있습니다. 특히 특허가 부여되었거나 실용신안 또는 의장등록이 된 경우, 번역을 포함한 제반 권리는 지멘스의 소유입니다.

산업체 고객을 위한 교육 과정의 사용은 명시적으로 금지됩니다. 지멘스는 교육 커리큘럼의 상업적 이용을 거부합니다.

드레스덴공대(TU Dresden), 특히 공학 박사 Leon Urbas 교수와 Michael Dziallas Engineering Corporation, 그리고 본 교육 커리큘럼을 준비하는 과정에서 도움을 주신 모든 관계자들께 감사의 말씀을 전합니다.

목차

TOC

SIMATIC S7-1500을 위한 PID 컨트롤러

1 목표

이 챕터에서는 TIA Portal 프로그래밍 툴을 통해 SIMATIC S7-1500에서 소프트웨어 PID 컨트롤러를 사용하는 방법에 대해 배워보겠습니다.

이 모듈에는 SIMATIC S7-1500에서 PID 컨트롤러를 호출, 연결, 구성 및 최적화하는 방법이 설명되어 있습니다. 또한, TIA Portal에서 PID 컨트롤러를 호출하여 사용자 프로그램에 삽입하기 위한 단계들이 나와 있습니다.

제3장에 기술된 SIMATIC S7 제어 장치를 사용할 수 있습니다.

2 전제 조건

SIMATIC S7 CPU1516F-3 PN/DP에서 챕터 "IEC 타이머 및 카운터"에서 학습한 내용을 토대로 합니다. 이 챕터에서는 예를 들어 SCE_EN_032-500_Analog_Values_R1508.zap13 같은 프로젝트를 이용할 수 있습니다.

3 필요한 하드웨어 및 소프트웨어

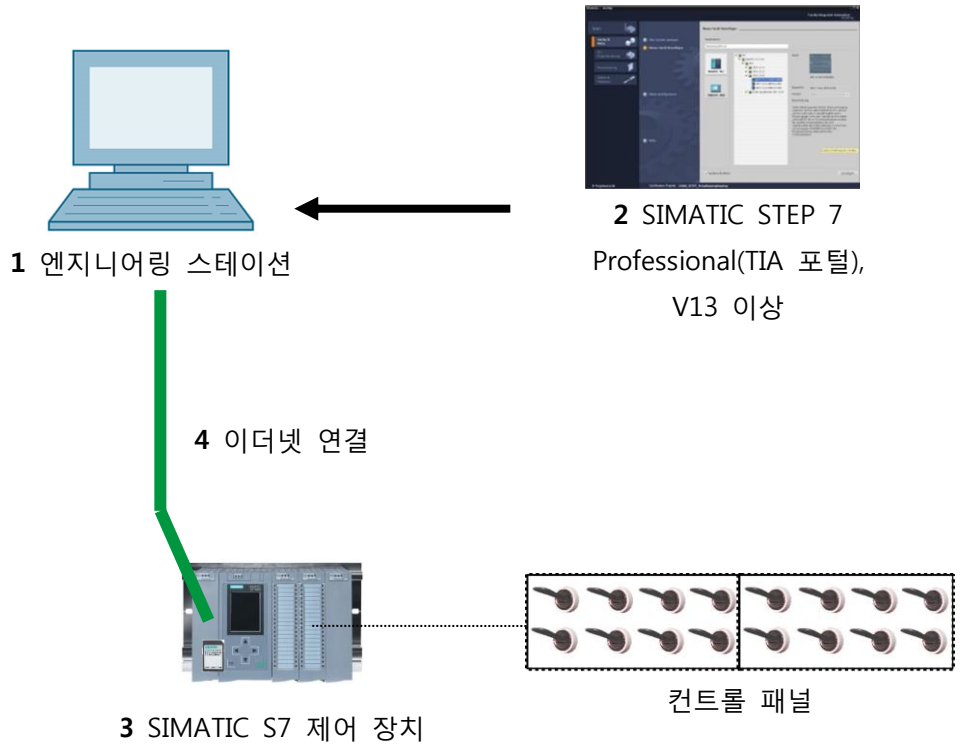
1 엔지니어링 스테이션: 하드웨어 및 운영 시스템이 필요합니다(자세한 정보는 TIA 포털의 설치 DVD Readme/Liesmich를 참조하세요).

2 TIA 포털의 소프트웨어 SIMATIC STEP 7 Professional – V13부터

3 SIMATIC S7-1500/S7-1200/S7-300 제어 장치, 예: CPU 1516F-3 PN/DP – 펌웨어 버전 V1.6 이상, 메모리 카드와 16DI/16DO 및 2AI/1AO 포함

참고: 디지털 입력과 아날로그 입력 및 출력은 컨트롤 패널에서 실행해야 합니다.

4 엔지니어링 스테이션과 제어 장치 간 이더넷 연결



4 이론

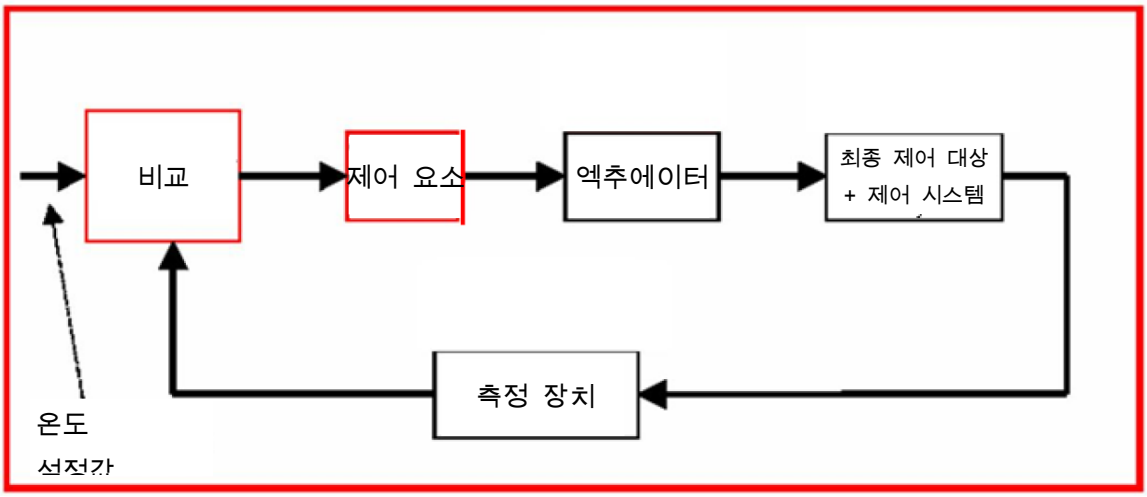
4.1 폐쇄 루프(Closed loop) 제어 태스크

폐쇄 루프 제어는 변수 측정값을 토대로 개입을 하여 변수의 값을 연속적으로 생성 및 유지하는 프로세스입니다.

프로세스는 변수 측정값을 토대로 실행이 되고, 변수는 프로세스 자체에 영향을 받기 때문에 이 모듈에서는 폐쇄 루프(제어 루프)에서 이루어지는 조치 경로를 만들어 보겠습니다.

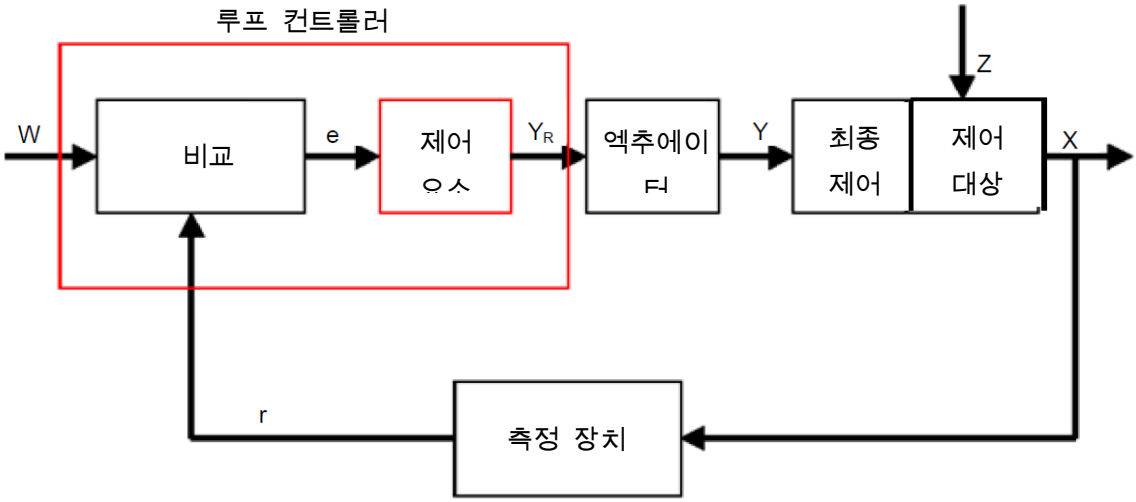
제어하고자 하는 변수를 연속적으로 측정해서 같은 유형의 사전 설정된 또 다른 변수와 비교합니다. 이러한 비교 결과에 따라 제어하고자 하는 변수가 사전 설정된 변수의 값에 맞게 조정이 됩니다.

폐쇄 루프 제어를 보여주는 다이어그램



4.2 폐쇄 루프의 구성 요소

폐쇄 루프 제어의 기본적인 개념이 아래에 자세히 설명되어 있습니다. 시작을 돕기 위해 다이어그램을 토대로 한 개요가 여기 나와 있습니다.



1. 제어 대상 변수(Controlled variable) x

폐쇄 루프 제어의 실제 "목표(Taget)"로, 제어에 영향을 받거나 일정하게 유지되는 변수입니다. 난방 시스템의 경우, 방안 온도가 여기에 해당됩니다. 특정 시점에서 제어 대상 변수의 순시 값을 이 시점에서의 "실제 값(actual value)"이라고 합니다.

2. 피드백 변수(Feedback variable) r

제어 루프에서는 원치 않는 변경에 응답할 수 있도록 제어 대상 변수를 연속적으로 확인합니다. 제어 대상 변수에 비례한 측정량을 피드백 변수라고 합니다. 예를 들어 "난방"에서 내부 온도계의 측정 전압이 여기에 해당됩니다.

3. 방해 변수(The disturbance variable) z

방해 변수는 원치 않는 방식으로 제어 대상 변수에 영향을 미치는 변수로서, 제어 대상 변수를 현재 설정값에서 벗어나게 만듭니다. 고정 설정값 제어의 경우, 방해 변수가 존재하기 때문에 제어의 시작 부분에만 방해 변수를 고려한 제어가 필요합니다. 난방 시스템 검토 시에는 외부 온도나 방안 온도가 이상적인 값에서 벗어나게 만드는 기타 모든 변수가 여기에 해당됩니다.

4. 설정값(Setpoint) w

특정 시점에서의 설정값은 이 시점에 제어 대상 변수가 이상적으로 가지고 있어야 하는 값입니다. 설정값은 종속 제어 내에서 따라 계속 변경될 수 있다는 점을 주의해야 합니다. 난방 시스템의 경우, 현재 원하는 목표 방안 온도가 여기에 해당됩니다.

5. 비교(Comparing element)

현재 측정된 제어 대상 변수와 비교할 피드백 변수의 순시 값의 비교가 이루어지는 지점입니다. 대부분의 경우, 두 변수 모두 전압으로 측정됩니다. 두 변수의 차이라면 "시스템 오류" e입니다. 이 값은 제어 요소로 전달되어 평가됩니다 (아래 참조).

6. 제어 요소(Controlling element)

제어 요소는 폐쇄 루프 제어의 실질적인 핵심입니다. 제어 요소는 입력 변수인 시스템 오류를 평가하는데, 이는 현제 설정값에서 제어 대상 변수가 어떻게, 얼마나 많이 벗어났는지, 그리고 실질적으로 제어 대상 변수에 근본적인 영향을 미치는 "컨트롤러 출력 변수" Y_R를 얻습니다. 난방 시스템의 경우, 컨트롤러 출력 변수는 믹서 모터를 위한 전압이 됩니다.

제어 요소가 시스템 오류로부터 컨트롤러 출력 변수를 결정하는 방식이 폐쇄 루프 제어의 주요 기준이 됩니다.

7. 엑추에이터(Actuator)

엑추에이터는 폐쇄 루프 제어의 '집행 기관'이라 할 수 있습니다. 제어 대상 변수가 어떤 영향을 받을 것인지에 대한 정보를 컨트롤러 출력 변수의 형태로 제어 요소로부터 수신하여 이를 "조작 변수(Manipulated variable)"로 변환합니다. 난방 시스템의 경우, 믹서 모터 컨트롤러가 여기에 해당됩니다.

8. 최종 제어 요소(Final controlling element)

조작 변수 Y의 펑션으로서 제어 대상 변수에 다소 직접적으로 영향을 미치는 제어 루프의 구성 요소입니다. 난방 시스템의 경우, 믹서, 난방 라인 및 라디에이터의 조합이 여기에 해당됩니다. 믹서 (조작 변수)의 조절은 믹서 모터 (엑추에이터)에 의해 이루어지며, 수온을 통해 방안 온도에 영향을 미칩니다.

9. 제어 대상 시스템(Controlled system)

제어 대상 시스템은 제어할 변수를 포함하고 있는 시스템으로서, 난방 시스템의 경우 주거 공간이 여기에 해당됩니다.

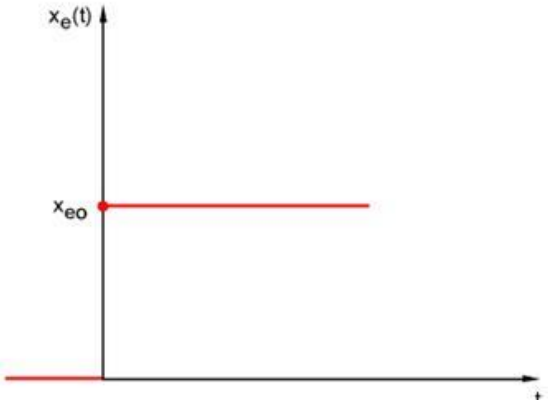
10. 정지 시간(Dead time)

정지 시간이란 컨트롤러 출력 변수가 변경되고 제어 대상 시스템에 측정 가능한 응답이 나타날 때까지의 시간을 뜻합니다. 난방 시스템의 경우, 믹서 모터의 전압이 바뀌고 이로 인한 방안 온도를 측정할 수 있는 시간이 여기에 해당됩니다.

4.3 제어 대상 시스템을 분석하기 위한 스텝 펑션

제어 대상 시스템, 컨트롤러 및 제어 루프의 응답을 분석하기 위해 입력 신호에 대해 통일된 형식의 펑션이 사용되는데 이를 스텝 펑션이라고 합니다.

제어 루프 요소를 분석하느냐 전체 제어 루프를 분석하느냐에 따라 제어 대상 변수 x(t), 조작 변수 y(t), 참조 변수 w(t) 또는 방해 변수 z(t)에 스텝 펑션으로 지정할 수 있습니다. 입력 신호에는 xe(t)가, 출력 신호에는 xa(t)가 지정됩니다.

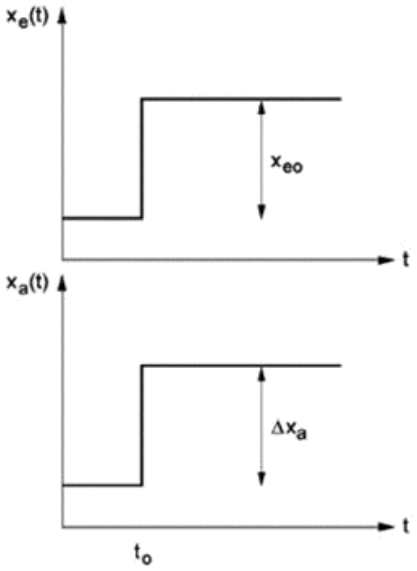


$$x_e(t) = \begin{cases} 0 & t < 0\text{인 경우} \\ x_{eo} \end{cases}$$

4.4 자기 조절(Self-regulation)이 되는 제어 대상 시스템.

4.4.1 시간 지연 없는 비례 시스템

이러한 제어 대상 시스템을 간단히 P 시스템이라고 합니다.



i t_0에서 입력 변수의 갑작스러운 변경

제어 대상 변수/조작 변수:

$$x = K_{ss} \bullet y$$

Kss: 조작 변수 변경을 위한 비례 계수

$$K_{ss} = \frac{\Delta x}{\Delta y} = \tan\alpha$$

제어 대상 변수/방해 변수:

$$x = K_{sz} \bullet z$$

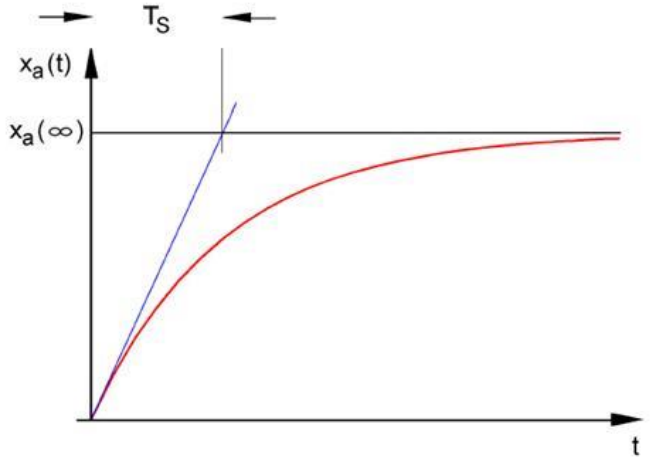
Ksz: 방해 변수 변경을 위한 비례 계수

범위: $y_h = y_{max} - y_{min}$

제어 범위: $x_h = x_{max} - x_{min}$

4.4.2 시간 지연 있는 비례 시스템

이러한 제어 대상 시스템을 간단히 P-T1 시스템이라고 합니다.



일반적인 입력 신호를 위한 미분 방정식 $x_e(t)$:

$T_S \cdot x_a(t) + x_a(t) = K_{PS} \cdot x_e(t)$

입력 신호의 스텝 펑션 (스텝 응답)를 위한 미분 방정식 해법

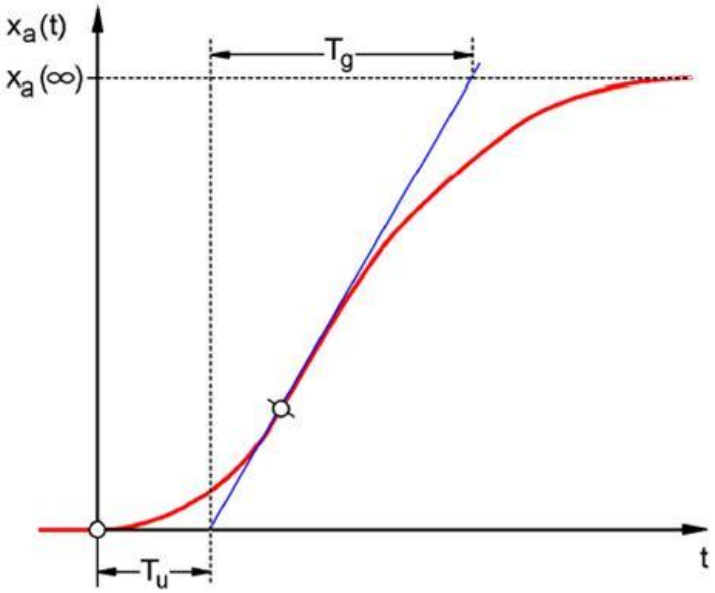
$x_a(t) = K_{PS} (1-e^{-t/TS}) \cdot x_{eo}$

$x_a (t = \infty) = K_{PS} \cdot x_{eo}$

T_S: 시간 상수

4.4.3 시간 지연이 2회 발생하는 비례 시스템

이러한 제어 대상 시스템을 간단히 P-T2 시스템이라고 합니다.



Tu: 지연 시간 Tg: 보상 시간

시간 상수 TS1 및 TS2를 갖는 두 P-T1 시스템을 반응로부터 자유롭도록 직렬 연결하여 시스템이 생성됩니다.

P-Tn 시스템의 제어 가능성:

$\frac{T_u}{T_g} < \frac{1}{10}$ → 손쉽게 제어 가능 $\frac{T_u}{T_g} \approx \frac{1}{6}$ → 여전히 제어 가능 $\frac{T_u}{T_g} > \frac{1}{3}$ → 제어가 어려움

Tu/Tg 비율이 증가할수록 시스템의 제어 가능성은 낮아집니다.

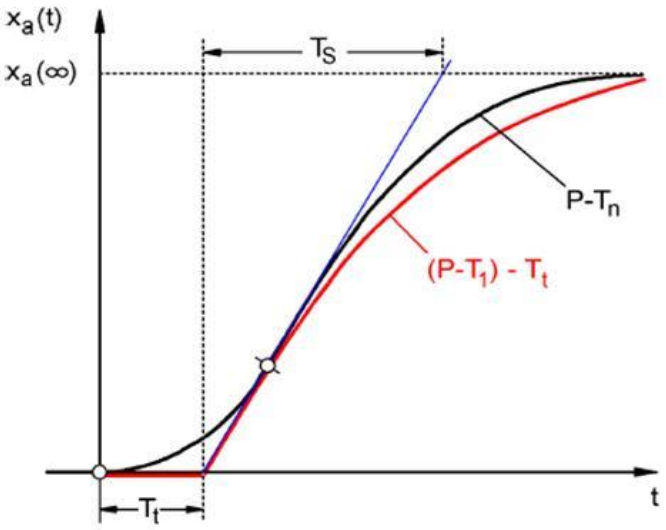
4.4.4 시간 지연이 n회 발생하는 비례 시스템

이러한 제어 대상 시스템을 간단히 P-Tn 시스템이라고 합니다.

시간 응답은 n차 미분 방정식으로 설명됩니다. 스텝 응답의 특성은 P-T2 시스템과 비슷합니다. 시간 응답은 Tu 및 Tg로 설명됩니다.

대체 옵션: 지연이 수없이 발생하는 시스템의 경우에는 P-T1 시스템을 정지 시간 시스템에 직렬 연결하는 것으로 대체할 수 있습니다.

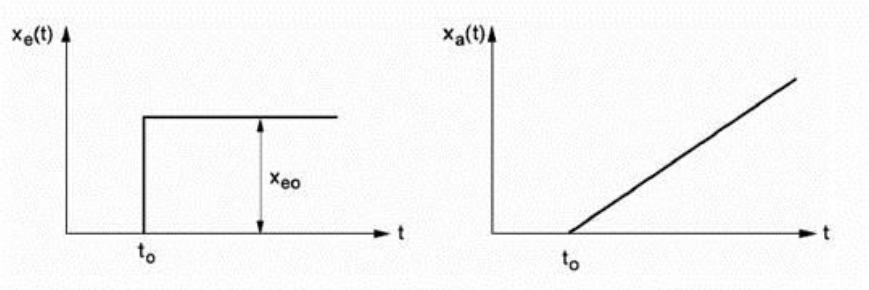
Tt ≫ Tu 및 TS ≫ Tg가 적용됩니다.



4.5 자기 조절이 되지 않는 시스템

이러한 제어 대상 시스템을 간단히 I 시스템이라고 합니다.

방해 이후에도 제어 대상 변수가 계속 꾸준히 증가하기 때문에 고정된 최종 값을 얻기 위해 노력할 필요가 없습니다.



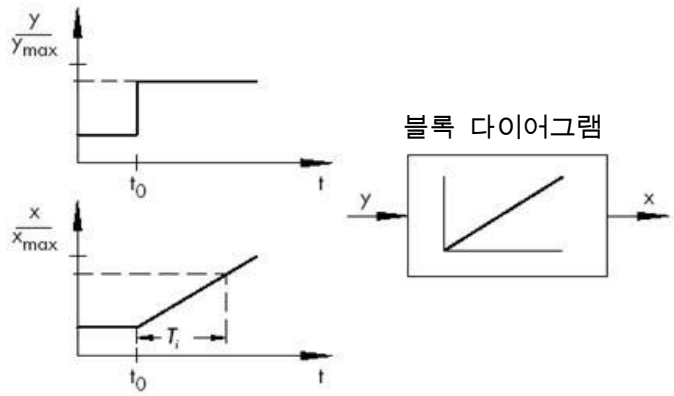
예: 수위 제어

들어오는 유량과 나가는 유량이 동일한 탱크의 경우에는 수위가 일정합니다. 유입되는 유량 또는 방출되는 유량에 변화가 생기면 수위가 증가 또는 감소합니다. 들어오는 유량과 나가는 유량의 차이가 증가하면 수위도 빠르게 변화합니다.

실질적으로 적분 기능에 한계가 있는 경우가 많다는 것을 이 예는 분명하게 보여주고 있습니다.

제어 대상 변수는 시스템 고유의 한계 값에 도달할 때까지만 증가 또는 감소합니다. 레벨 제어의 경우, 탱크가 넘치거나 비게되고, 압력 제어의 경우, 압력이 최대 또는 최소 수준에 도달합니다.

아래 그림에는 입력 변수에서 스텝 펑션 변화에 대한 I 시스템의 시간 응답을 비롯해 도출된 블록 다이어그램이 나와 있습니다.



입력 신호의 스텝 펑션이 xe(t) 펑션으로 변경되면 다음과 같이 계산이 됩니다.

$x_a(t)=K_{IS}\int x_e(t)\,dt \Rightarrow$ 제어 대상 시스템 적산

K_{IS}: 제어 대상 시스템의 적분 계수

* SAMSON 기술 정보 - L102 컨트롤러 및 제어 대상 시스템에서 나온 그림, 버전: 2000년 8월 (http://www.samson.de/pdf_en/l102en.pdf)

4.6 *연속 컨트롤러(Continuous controller)의 기본 유형*

1개 또는 2개의 조작 변수만을 활성화 및 비활성화하는 이산(discrete) 컨트롤러는 단순성의 이점이 있습니다. 두 컨트롤러 자체는 물론이고 엑추에이터 및 최종 제어 요소들이 연속 컨트롤러보다 훨씬 간단하고 따라서 비용도 저렴합니다.

하지만 이산 컨트롤러는 몇 가지 단점도 있습니다. 그 한 가지는 대형 전기 모터나 냉각 장치 같은 대형 시스템을 작동시켜야 하는 경우 스위치가 "온" 상태일 때 부하 피크가 발생해서 전원 공급에 과부하가 걸릴 수 있다는 점입니다. 이러한 이유로 "오프"와 "온" 간에 전환을 하기 보다는 최대 전력("전부하")과 이 보다 훨씬 낮은 엑추에이터 또는 최종 제어 요소의 전력("기저 부하") 간에 전환을 하는 경우가 종종 있습니다. 이러한 개선 노력에도 불구하고 여전히 이산 폐쇄 루프 제어가 적합하지 않은 경우가 많습니다. 속도가 이산적으로 제어되는 자동차 엔진을 생각해 봅시다.
이 경우에는 아이들링 상태와 최대 출력의 중간 상태가 없습니다.

갑작스러운 최대 출력으로부터 발생한 힘이 타이어를 통해 노면에 적절하게 전달될 수 없다는 사실 외에도, 이산 제어는 도로 교통에도 적합하지 않을 수 있습니다.

따라서 이러한 용도로는 연속 컨트롤러가 사용됩니다. 이론적으로는 시스템 오류와 컨트롤러 출력 변수 사이에 제어 요소를 설정하는 수학적 관계에는 제한이 거의 없습니다. 그러나 실제로는 3가지 전형적인 유형으로 구분이 됩니다. 이에 대해서는 아래에서 보다 자세히 설명하겠습니다.

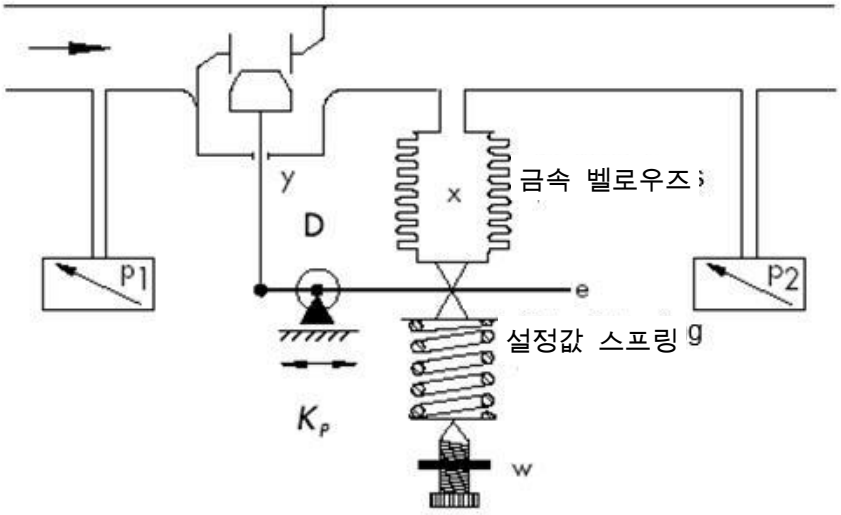
4.6.1 비례 컨트롤러 (P 컨트롤러)

P 컨트롤러의 조작 변수 y는 측정 오류 e에 비례합니다. 따라서 P 컨트롤러는 편차 발생 시 지체 없이 반응하며 조작 변수만을 생성한다고 추론할 수 있습니다.

그림에 나와 있는 비례 압력 컨트롤러는 설정값 스프링의 FS 힘과 압력 p2에 의해 탄성 금속 벨로우즈에서 생성된 FB 힘을 비교합니다. 두 힘이 균형을 이루고 있지 않으면 레버가 지점 D를 축으로 회전을 합니다. 이렇게 되면 밸브 플러그의 위치가 바뀌면서 새롭게 힘의 균형이 회복될 때까지 제어할 압력 p2가 바뀝니다.

오류 변수의 스텝 변화 이후 P 컨트롤러의 동적 작동이 그림에 나와 있습니다. 조작 변수 y의 진폭은 오류 e와 비례 조치 계수 Kp에 의해 결정됩니다.

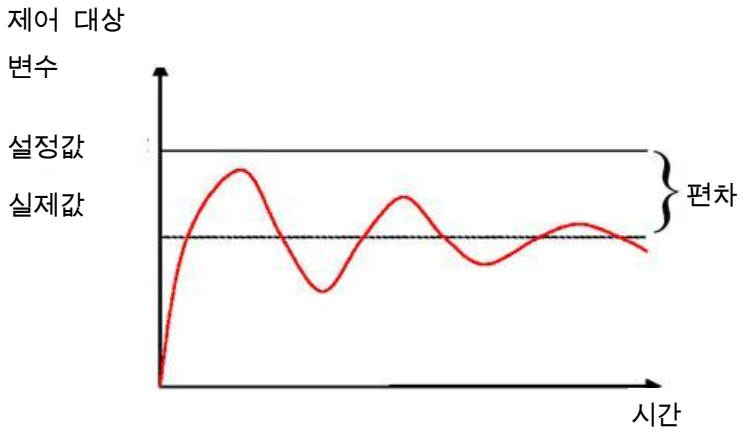
제어 편차를 가능한 작게 유지하려면 가능한 큰 값의 비례 조치 계수를 선택해야 합니다. 이 계수 값이 증가하면 컨트롤러의 반응 속도가 빨라집니다. 하지만 값이 너무 크면 오버슈트가 발생할 위험이 있고 컨트롤러의 "헌팅" 경향성이 커집니다.



$$y = K_p \cdot e$$

* SAMSON 기술 정보 - L102 컨트롤러 및 제어 대상 시스템에서 나온 그림과 텍스트, 버전: 2000년 8월 (http://www.samson.de/pdf_en/l102en.pdf)

다이어그램에서 P 컨트롤러의 응답을 확인할 수 있습니다.



이러한 컨트롤러 유형은 단순성이라는 장점이 있지만 (저항기만으로도 전자식 구현이 가능한 경우가 있음), 다른 한편으로는 다른 컨트롤러 유형에 비해 반응이 너무 즉각적이라는 단점도 있습니다.

P 컨트롤러의 가장 큰 단점은 끊임없이 시스템 편차가 발생한다는 것입니다. 따라서 장기적으로 보면 설정값에 완전히 도달할 수가 없습니다. 이러한 단점 외에도 응답 속도가 아직 최적화되지 않았기 때문에 비례 조치 계수를 늘리는 것으로 만족할 만한 수준까지 단점을 최소화하는 것이 불가능합니다. 왜냐하면 이로 인해 컨트롤러에 의한 오버슈트, 즉 과반응이 야기될 수 있기 때문입니다. 최악의 경우 컨트롤러는, 조작 변수가 아니라 컨트롤러 자체에 의해 제어 대상 변수가 설정값에서 주기적으로 벗어나는 영구적 발산(oscillation) 상태에 빠질 수 있습니다.

영구적인 제어 편차의 문제는 적분 컨트롤러를 추가하여 해결하는 것이 가장 좋습니다.

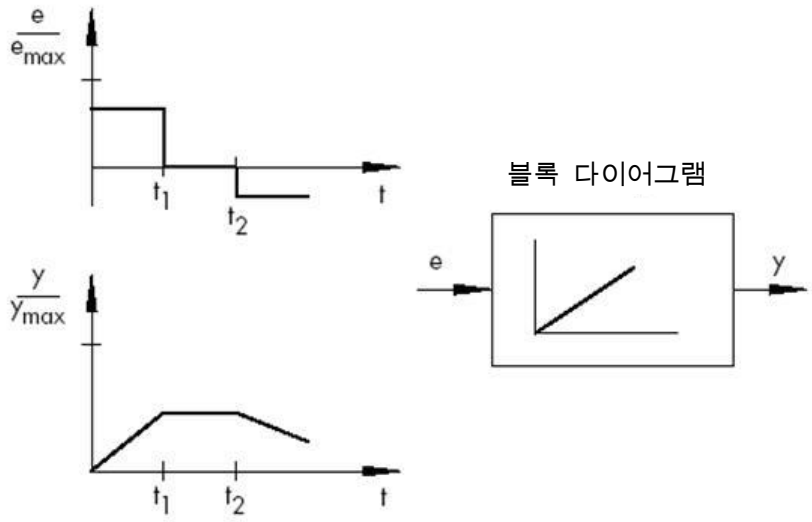
4.6.2 적분 컨트롤러 (I 컨트롤러)

적분 제어 조치를 사용하면 어떤 작동 지점에서든 시스템 편차를 완전히 교정할 수 있습니다. 오류가 0이 아닌 한, 적분 조치를 하면 조작 변수의 값이 변경됩니다. 참조 변수와 제어 대상 변수의 값이 똑같이 크면서도 조작 변수가 시스템 고유의 한계 값(Umax, pmax 등)에 도달할 때만 제어 프로세스가 균형을 이룹니다.

수학에서의 적분 조치에 따라 조작 변수의 값이 오류 e에 대한 적분에 비례해 변경됩니다.

$$y = K_i \int e\, dt \qquad \text{mit:} \quad K_i = \frac{1}{T_n}$$

조작 변수가 얼마나 빨리 증가/감소하느냐는 오류 및 적분 시간에 따라 다릅니다.

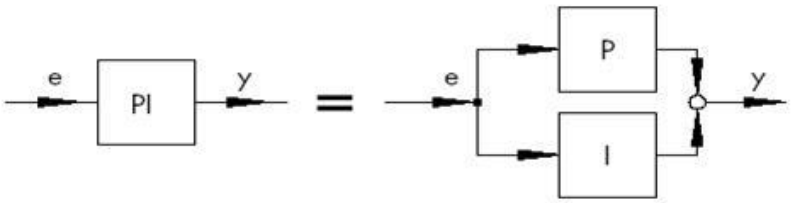


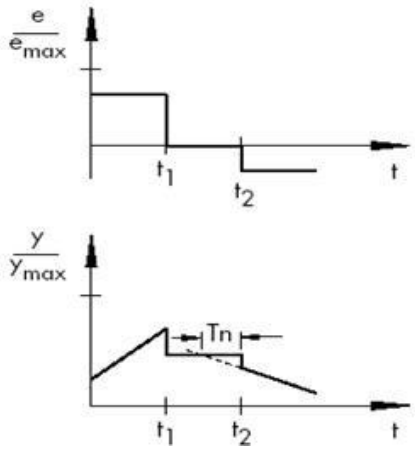
* SAMSON 기술 정보 - L102 컨트롤러 및 제어 대상 시스템에서 나온 그림과 텍스트, 버전: 2000년 8월 (http://www.samson.de/pdf_en/l102en.pdf)

4.6.3 PI 컨트롤러

PI 컨트롤러가 실제 사용되는 경우가 종종 있습니다. PI 컨트롤러는 하나의 P 컨트롤러와 하나의
I 컨트롤러가 병렬로 연결된 것입니다.

제대로 설계될 경우, PI 컨트롤러는 두 컨트롤러 유형의 장점(안정성 및 신속성과 정상 상태 오류 없음)을 하나로 결합할 수 있기 때문에 각자의 단점을 상쇄할 수 있습니다.





블록 다이어그램

동적 작동은 비례 조치 계수 Kp와 리셋 시간 Tn으로 표시가 됩니다. 비례 구성요소 덕분에 조작 변수는 오류 신호 e에 즉각 반응하지만, 적분 구성요소는 일정 시간이 지난 이후에만 영향력을 갖게 됩니다. Tn은 처음에 P 구성요소 (Kp)가 생성한 것과 동일한 제어 진폭을 I 구성요소가 생성할 때까지 경과된 시간을 표현합니다. I 컨트롤러에서와 마찬가지로 적분 조치 구성요소가 증폭될 경우 리셋 시간 Tn을 줄여야 합니다.

컨트롤러 차원화 (dimensioning):

Kp 값과 Tn 값을 조정하면 제어 대상 변수의 발산을 줄일 수 있지만, 제어의 역동성이 떨어진다는 점은 감수해야 합니다.

PI 컨트롤러의 용도: 정상 상태 오류가 발생하지 않는 신속한 제어 루프

예: 압력, 온도, 비율 제어 등

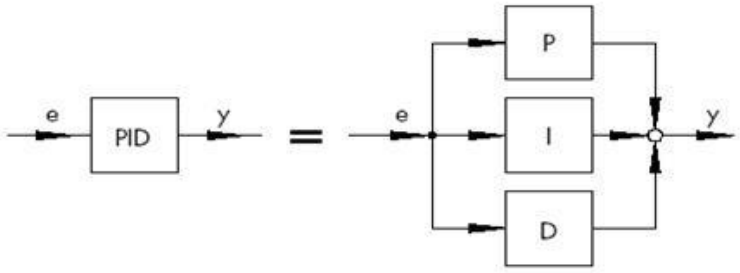
* SAMSON 기술 정보 - L102 컨트롤러 및 제어 대상 시스템에서 나온 그림과 텍스트, 버전: 2000년 8월 (http://www.samson.de/pdf_en/l102en.pdf)

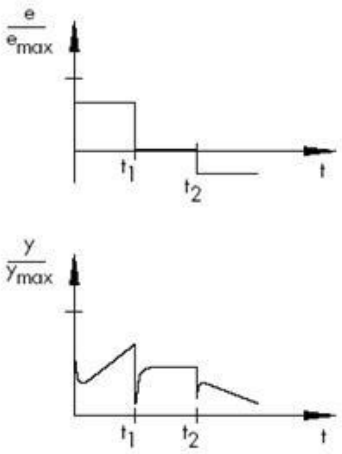
4.6.4 미분 컨트롤러 (D 컨트롤러)

D 컨트롤러는 P 컨트롤러 같이 진폭이 아니라 오류의 변화율을 토대로 조작 변수를 생성합니다. 따라서, 반응 속도가 P 컨트롤러보다 훨씬 빠릅니다. 오류가 아무리 작더라도 진폭의 변화가 발생하면 그 즉시 미분 컨트롤러가 큰 제어 진폭을 미리 생성합니다. 그러나 오류의 크기와 관계 없이 변화율은 0이기 때문에 정상 상태 오류 신호를 D 컨트롤러가 인식하지 못합니다. 따라서 미분 전용 컨트롤러는 실제로 거의 사용되지 않습니다. 보통은 다른 제어 요소들과 결합해 사용이 되며, 대부분은 비례 제어와 함께 사용됩니다.

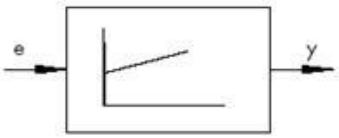
4.6.5 PID 컨트롤러

D 구성요소를 PI 컨트롤러에 추가하면 활용도가 뛰어난 PID 컨트롤러가 탄생합니다. PD 컨트롤러에서와 마찬가지로, 추가된 D 구성요소(적절하게 조정된 경우)는 제어 대상 변수가 설정값에 보다 빨리 도달하여 보다 신속하게 안정 상태가 되도록 해줍니다.





블록 다이어그램



$$y = K_P \cdot e + K_i \int e\,dt + K_D \frac{de}{dt} \quad \text{with} \quad K_i = \frac{K_P}{T_n};\ K_D = K_P \cdot T_V$$

* SAMSON 기술 정보 - L102 컨트롤러 및 제어 대상 시스템에서 나온 그림과 텍스트, 버전: 2000년 8월 (http://www.samson.de/pdf_en/l102en.pdf)

4.7 발산 테스트를 통한 컨트롤러 튜닝

만족할 만한 제어 결과를 위해서는 적절한 컨트롤러를 선택하는 것이 매우 중요합니다. 제어 파라미터 Kp에서는 더욱 그렇습니다. Tn 및 TV는 시스템 응답에 맞게 적절하게 조정이 되어야 합니다. 일반적으로 컨트롤러 파라미터의 조정은 안정성은 높이지만 제어 루프의 속도를 느리게 하거나, 매우 역동적이지만 제어 응답이 불규칙해서 발산이 쉽게 발생하고 이로 인해 제어 루프가 불안정해진다는 단점이 여전히 존재합니다.

항상 동일한 동작 값으로 작동해야 하는 비선형적 시스템의 경우(예: 고정 설정값 제어)에는 이러한 특정 동작 값에서 시스템 응답에 맞게 컨트롤러 파라미터를 조정해야 합니다. 후속 제어(follow-up) 시스템 ñ 에서와 같이 고정 동작 값을 정의할 수 없는 경우에는 전체 작동 범위 내에서 충분히 빠르고 안정적인 제어가 이루어질 수 있도록 컨트롤러를 튜닝해야 합니다.

실제로는 컨트롤러는 일반적으로 경험을 통해 확보한 값을 토대로 튜닝됩니다.

그러나 이러한 값들을 사용할 수 없는 경우에는 시스템 응답을 자세하게 분석하고 몇 가지 이론적/실용적 튜닝 방식을 적용해서 적절한 제어 파라미터를 결정해야 합니다.

Ziegler와 Nichols가 최초로 제안한 극한 방식(ultimate method)이라 불리는 접근 방식도 그 중 하나인데, 많은 경우에 적용할 수 있도록 간단한 튜닝 방식을 제공합니다. 그러나 제어 대상 변수에 대한 발산을 지속할 수 있는 제어 대상 시스템에만 적용이 가능합니다.

이 방식의 경우 다음과 같이 조정이 진행이 됩니다.

- 컨트롤러에서 Kp 및 Tv을 최하위 값으로, Tn을 최상위 값(컨트롤러의 영향력이 최소)으로 설정합니다.
- 원하는 동작 위치(제어 루프 스타트업)에 맞게 제어 대상 시스템을 수동으로 조정합니다.
- 컨트롤러의 조작 변수를 수동 조정된 값으로 설정하고 자동 작동 모드로 전환합니다.
- 제어 대상 변수가 조화 발산할 때까지 Kp를 계속해서 증가시킵니다 (Xp를 감소). 가능하다면 Kp 조정이 이루어지는 동안 설정값을 약간 단계적으로 변경하여 제어 루프에서 발산을 일으켜야 합니다.
- 조정된 Kp 값을 중요한 비례 조치 계수 Kp,crit까지 낮춥니다. 필요할 경우 몇몇 발산 시간의 평균을 계산해서 전체 발산 진폭에 대한 시간 범위를 Tcrit 값으로 결정합니다.
- 테이블에 따라 이 값에 Kp,crit 및 Tcrit의 값을 곱하고 컨트롤러에서 결정된 Kp, Tn 및 Tv 값을 입력합니다.

	K_p	T_n	T_v
P	0.50 x $K_{p.\ crit.}$	-	-
PI	0.45 x $K_{p.\ crit.}$	0.85 x $T_{crit.}$	-
PID	0.59 x $K_{p.\ crit.}$	0.50 x $T_{crit.}$	0.12 x $T_{crit.}$

* SAMSON 기술 정보 - L102 컨트롤러 및 제어 대상 시스템에서 나온 그림과 텍스트, 버전: 2000년 8월 (http://www.samson.de/pdf_en/l102en.pdf)

4.8 Tu-Tg 근사치 계산을 통한 컨트롤러 조정

여기서는 PT2 시스템을 예로 들어 제어 대상 시스템에 대한 조정을 수행해 보겠습니다.

Tu-Tg 근사치 계산

Ziegler-Nichols 방식과 Chien, Hrones 및 Reswick 방식은 Tu-Tg 근사치 계산을 토대로 하는데, 여기서는 시스템 스텝 응답을 기준으로 시스템 이동 계수 Ks, 지연 시간 Tu 및 균형 시간 Tg 파라미터가 결정됩니다.

아래 설명되어 있는 조정 규칙들은 아날로그 컴퓨터 시뮬레이션을 이용한 실험 결과를 토대로 만들어졌습니다.

소위 Tu-Tg 근사치 계산, 즉 P-T1-TL 시스템을 이용한 근사치 계산을 통해 충분한 정확성을 가지고 P-TN 시스템을 설명할 수 있습니다.

입력 스텝 높이가 K인 시스템 스텝 응답이 시작 지점이 됩니다. 그림에서와 같이 필요한 파라미터 (시스템 이동 계수 Ks, 지연 시간 Tu 및 균형 시간 Tg)가 결정됩니다.

계산에 필요한 시스템 이동 계수 Ks를 결정할 수 있으려면 이동 펑션이 최종적인 정상 상태 값 (K*Ks)에 도달해야 합니다.

이 방법의 가장 큰 장점은 시스템의 분석 설명이 불가능한 경우에도 근사치 계산을 이용할 수 있다는 점입니다.

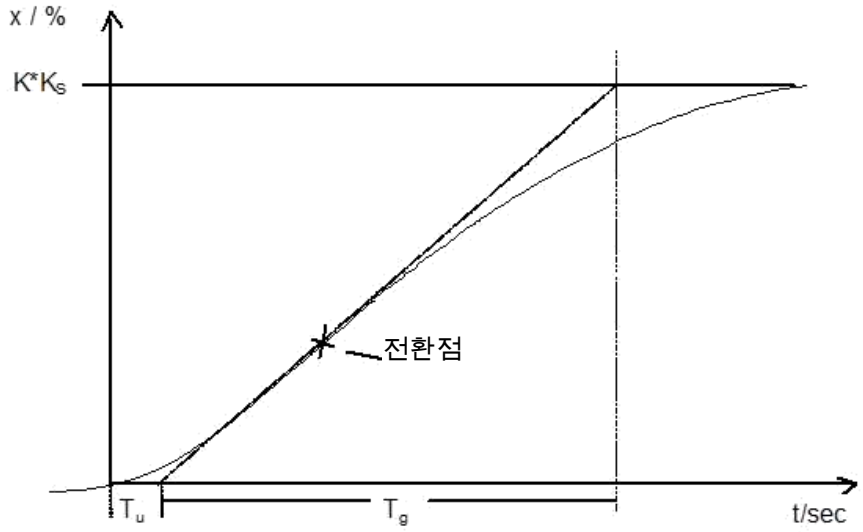


그림: Tu-Tg 근사치 계산

4.8.1 Ziegler-Nichols 방식에 따른 PI 컨트롤러 조정

P-T1-TL 시스템에서의 실험 결과를 토대로 Ziegler와 Nichols는 고정 설정값 제어를 위해 다음과 같은 컨트롤러 조정이 가장 최적이라는 것을 확인했습니다.

$$K_{PR} = 0{,}9 \frac{T_g}{K_S T_u}$$

$$T_N = 3{,}33\ T_u$$

이러한 조정 값을 이용하면 일반적으로 장애에 신속하게 응답할 수 있습니다.

4.8.2 Chien, Hrones 및 Reswick 방식에 따른 PI 컨트롤러 조정

가장 바람직한 컨트롤러 제어를 위해 장애에 대한 응답과 설정값 변경에 대한 응답을 모두 검토했습니다. 두 경우 서로 다른 값이 생성됩니다. 뿐만 아니라, 서로 다른 제어 성능 요구사항을 충족하도록 각각의 경우에 두 가지 조정이 지정되어 있습니다.

그 결과, 다음과 같이 조정이 이루어집니다.

- 장애 응답 시:

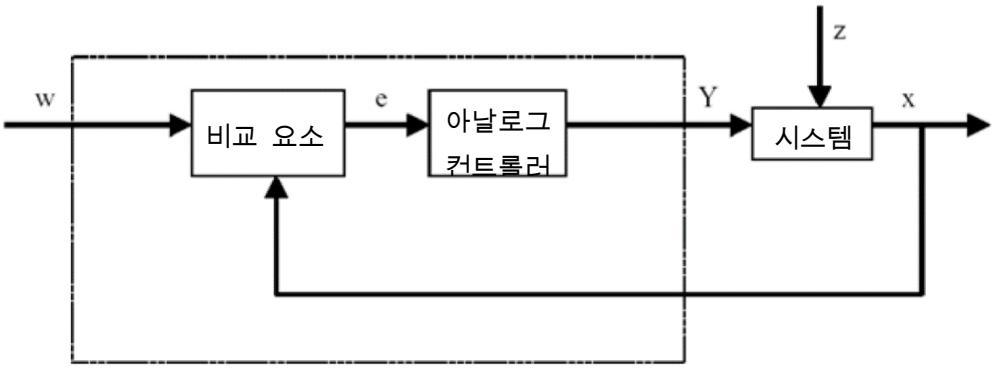
유지 기간이 가장 짧은 비주기적이고 일시적인 반응	20% 오버슈트 최소 발산 기간
$K_{PR} = 0.6 \frac{T_g}{K_S T_u}$	$K_{PR} = 0.7 \frac{T_g}{K_S T_u}$
$T_N = 4\ T_u$	$T_N = 2{,}3\ T_u$

- 설정값 변경 시:

유지 기간이 가장 짧은 비주기적이고 일시적인 반응	20% 오버슈트 최소 발산 기간
$K_{PR} = 0.35 \frac{T_g}{K_S T_u}$	$K_{PR} = 0.6 \frac{T_g}{K_S T_u}$
$T_N = 1{,}2\ T_g$	$T_N = T_g$

4.9 디지털 컨트롤러

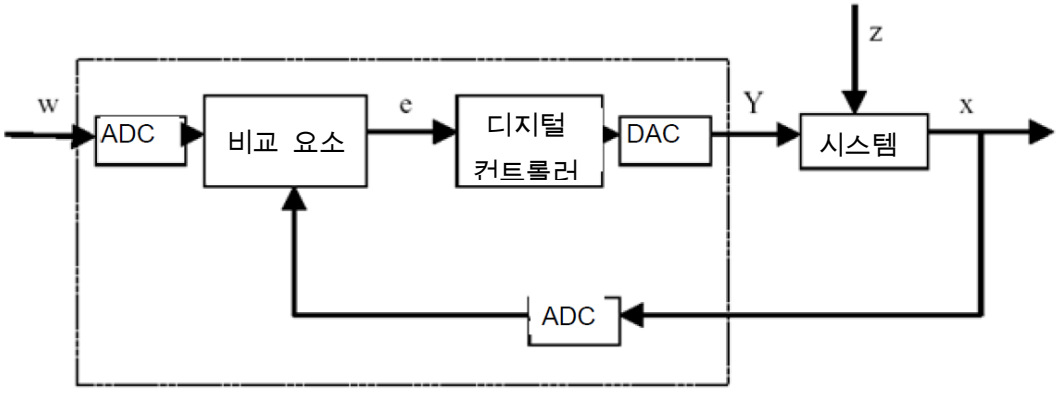
지금까지는 아날로그 컨트롤러, 즉 아날로그 값으로 존재하는 시스템 오류를 사용해 아날로그 방식으로 컨트롤러 출력 변수를 도출하는 컨트롤러에 주로 초점을 맞춰왔습니다. 이러한 유형의 제어 루프에 대한 다이어그램은 잘 알려져 있습니다.



그러나 시스템 오류에 대한 실제 평가를 디지털 방식으로 수행하는 것은 다음과 같은 장점이 있습니다. 먼저, 컴퓨터 프로그래밍에 사용할 수 있는 알고리즘이나 공식을 통해 정의를 할 수 있으면 아날로그 회로의 형태로 구현을 해야 하는 경우보다 시스템 오류와 컨트롤러 출력 변수의 관계를 훨씬 유연하게 정의할 수 있습니다. 또 하나는 디지털 기술이 여러 개의 컨트롤러를 최소한의 공간에 수용할 수 있도록 훨씬 강력한 회로 통합을 지원한다는 것입니다. 마지막으로, 컴퓨팅 용량이 충분할 때 컴퓨팅 시간을 분할하면 컴퓨터를 여러 제어 루프를 제어하는 하나의 컨트롤러로서 사용할 수도 있습니다.

변수에 대한 디지털 처리가 가능하도록 먼저 참조 변수와 피드백 변수가 모두 ADC(아날로그-디지털 컨버터)에서 디지털 값으로 변환됩니다. 이들 값은 디지털 비교 요소에 의해 서로 뺄셈이 되고, 그 차이가 디지털 제어 요소로 전달됩니다. 그런 다음, DAC(디지털-아날로그 컨버터)에서 컨트롤러 출력 변수가 다시 아날로그 값으로 변환됩니다. 외견 상 컨버터, 비교 요소 및 제어 요소가 하나로 결합된 장치는 아날로그 컨트롤러처럼 생겼습니다.

다이어그램을 토대로 디지털 컨트롤러의 구조를 검토해보겠습니다.



디지털 방식의 컨트롤러 구현은 장점만큼이나 다양한 문제를 수반합니다. 이러한 이유에서 폐쇄 루프 제어의 정확도로 인해 디지털화에 큰 차질이 발생하지 않도록 디지털 컨트롤러와 관련된 일부 변수의 크기를 충분히 크게 선택해야 합니다.

디지털 컴퓨터를 위한 품질 기준은 다음과 같습니다.

- DAC의 양자화 (quantization) 해상도

  이를 통해 연속적인 값 범위가 얼마나 섬세하게 디지털 방식으로 매핑되는지를 지정할 수 있습니다. 선택된 해상도는 폐쇄 루프 제어에 중요한 미세 포인트 중 어떤 것도 손실되지 않도록 충분히 높아야 합니다.

- ADC의 샘플링 속도

  이는 컨버터에 존재하는 아날로그 값이 측정 및 디지털화되는 빈도입니다. 컨트롤러가 제어 대상 변수의 단계적 변경에 시의 적절하게 응답할 수 있도록 이 값이 충분히 커야 합니다.

- 사이클 시간

  아날로그 폐쇄 루프 컨트롤러와 달리, 각각의 디지털 컴퓨터는 클록 사이클로 작동합니다. 단일 클록 사이클 동안(출력 값이 계산되지만 입력 값에 대한 쿼리는 수행되지 않는) 제어 대상 변수에 대한 대대적인 변경이 일어날 수 없을 정도로 사용 컴퓨터의 속도가 충분히 빨라야 합니다.

디지털 컨트롤러의 응답이 아날로그 컨트롤러만큼 신속하고 정확하게 이루어지려면 디지털 컨트롤러의 성능이 충분히 커야 합니다.

5 과제

이 챕터에서는 챕터 "SCE_EN_032-500 아날로그 값"에서 생성된 프로그램에 속도 제어를 위한 PID 컨트롤러를 추가해 보겠습니다. 이를 위해서는 호출한 "MOTOR_SPEEDCONTROL" [FC10] 펑션을 삭제해야 합니다.

6 계획 수립

TIA Portal에서는 폐쇄 루프 제어를 하기위해 PID_Compact 테크날러지 오브젝트를 사용할 수 있습니다.

"MOTOR_SPEEDCONTROL" [FC10] 블록을 대신하여 이 테크날러지 오브젝트가 모터 속도의 폐쇄 루프 제어를 합니다.

이는 "032-500_Analog_Values" 프로젝트의 연장선에서 이루어질 것입니다. 사전에 미리 이 프로젝트의 압축을 풀어야 합니다.

"Main" [OB1] 오거나이제이션 블록에서 호출한 "MOTOR_SPEEDCONTROL" [FC10] 펑션을 삭제한 뒤, 테크날러지 오브젝트를 호출 및 연결하여 주기적 인터럽트 OB에서 호출합니다.

그런 다음 PID_Compact 테크날러지 오브젝트를 구성해 시운전해야 합니다.

6.1 PID_Compact 폐쇄 루프 제어 블록

PID_Compact 테크날러지 오브젝트는 PID 컨트롤러에 비례 조치 최종 제어 요소를 위한 통합된 튜닝 기능을 제공합니다.

다음과 같은 작동 모드가 가능합니다.

- 비활성
- 사전 튜닝
- 미세 튜닝
- 자동 모드
- 수동 모드
- 오류 모니터링으로 대체 값(substitute value) 출력

여기에서는 자동 모드에서 이 컨트롤러에 대한 연결, 파라미터 지정 및 시운전을 해보겠습니다.

시운전 동안 통합 튜닝 알고리즘을 사용해 제어 대상 시스템에 대한 제어 응답을 기록하겠습니다.

PID_Compact 테크날러지 오브젝트는 항상 주기적 인터럽트 OB에서 호출되며, 여기에서 사이클 시간은 50 ms로 고정되어 있습니다.

속도 설정값은 분당 회전수 (범위: +/- 50 rpm)의 PID_Compact 테크날러지 오브젝트 "설정값"

입력에 상수로 설정됩니다. 데이터 타입은 32비트 부동 소수점 수 (REAL)입니다.

실제 속도 값 -B8 (모터의 센서 실제 속도 값 +/-10V는 +/- 50 rpm에 해당)은 "Input_PER"에 입력됩니다.

그런 다음, 컨트롤러 "Output_PER"의 출력이 신호 -U1 (2방향 모터의 조작 속도 값 +/- 10V는 +/- 50 rpm에 해당)에 연결됩니다.

출력 -Q3 (컨베이어 모터 -M1 가변 속도)가 설정되어 있는 경우에만 컨트롤러가 활성 상태가 됩니다. 이것이 설정되어 있지 않으면 "Reset" 입력 연결을 통해 컨트롤러가 비활성화됩니다.

6.2 기술 다이어그램

여기에는 과제를 위한 기술 다이어그램이 나와 있습니다.

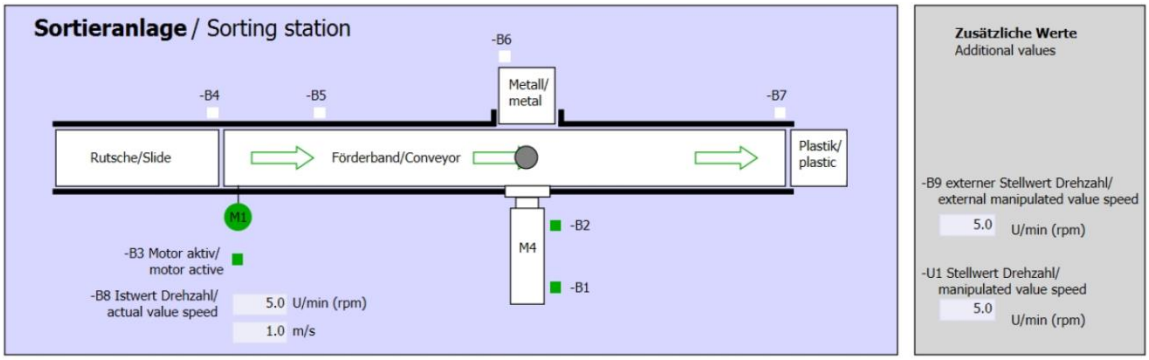


그림 1: 기술 다이어그램

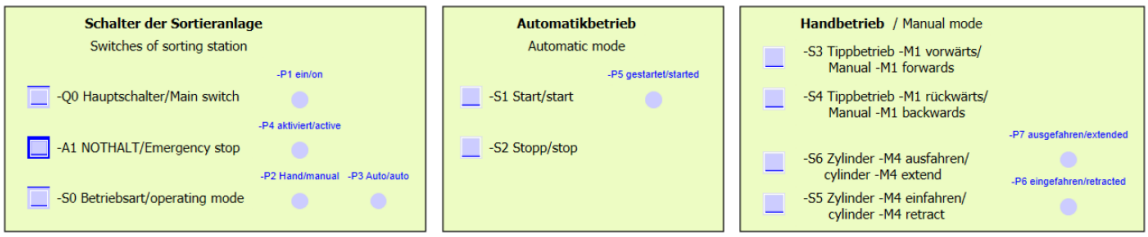


그림 2: 제어 패널

6.3 참조 목록

이 과제를 위한 글로벌 오퍼랜드로서 아래와 같은 신호들이 필요합니다.

DI	유형	식별자	기능	NC/NO
I 0.0	BOOL	-A1	반환 신호 비상 정지 OK	NC
I 0.1	BOOL	-K0	메인 스위치 "온"	NO
I 0.2	BOOL	-S0	모드 선택 수동 (0) / 자동 (1)	수동 = 0 자동 = 1
I 0.3	BOOL	-S1	푸시버튼 자동 시작	NO
I 0.4	BOOL	-S2	푸시버튼 자동 정지	NC
I 0.5	BOOL	-B1	센서 실린더 -M4 복귀	NO
I 1.0	BOOL	-B4	슬라이드의 센서	NO
I 1.3	BOOL	-B7	컨베이어 끝 센서	NO
IW64	BOOL	-B8	모터의 센서 실제 속도 값 +/- 10V는 +/- 50 rpm에 해당	

DO	유형	식별자	기능	
Q 0.2	BOOL	-Q3	컨베이어 모터 -M1 가변 속도	
QW 64	BOOL	-U1	2방향 모터의 조작 속도 값 +/- 10V는 +/- 50 rpm에 해당	

참조 목록 범례

DI	Digital Input	DO	Digital Output
AI	Analog Input	AO	Analog Output
I	Input	Q	Output
NC	Normally Closed		
NO	Normally Open		

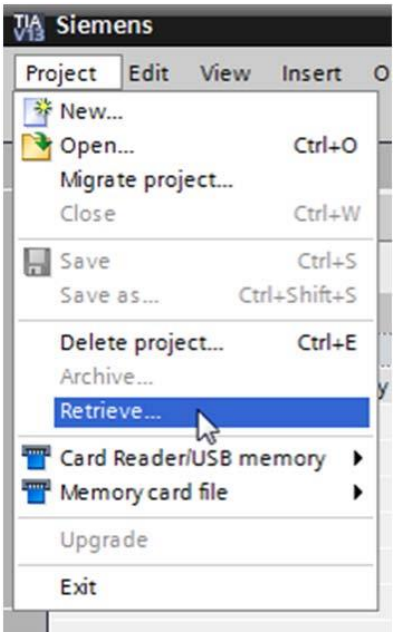
7 단계별 따라 해보기

아래에는 계획을 수립하는 방법에 대한 지침이 나와 있습니다. 모든 내용을 이미 충분히 숙지했다면 숫자가 표시된 단계로 넘어가도 좋습니다. 그렇지 않다면, 아래에 나와 있는 지침의 단계를 따라가면 됩니다.

7.1 기존 프로젝트 압축풀기

→ 챕터 "SCE_EN_032-500_Analog_Values"에서 생성된 "SCE_EN_032-500_Analog_Values_R1508.zap13" 프로젝트를 확장할 수 있으려면 먼저 아카이브에서 이 프로젝트의 압축을 풀어야 합니다. 아카이브된 기존 프로젝트의 압축을 풀려면 "Project"의 "Retrieve"로 가서 해당되는 아카이브를 선택해야 합니다. “Open”을 클릭해 선택합니다.

(→ Project → Retrieve → .zap 아카이브 선택 → Open)

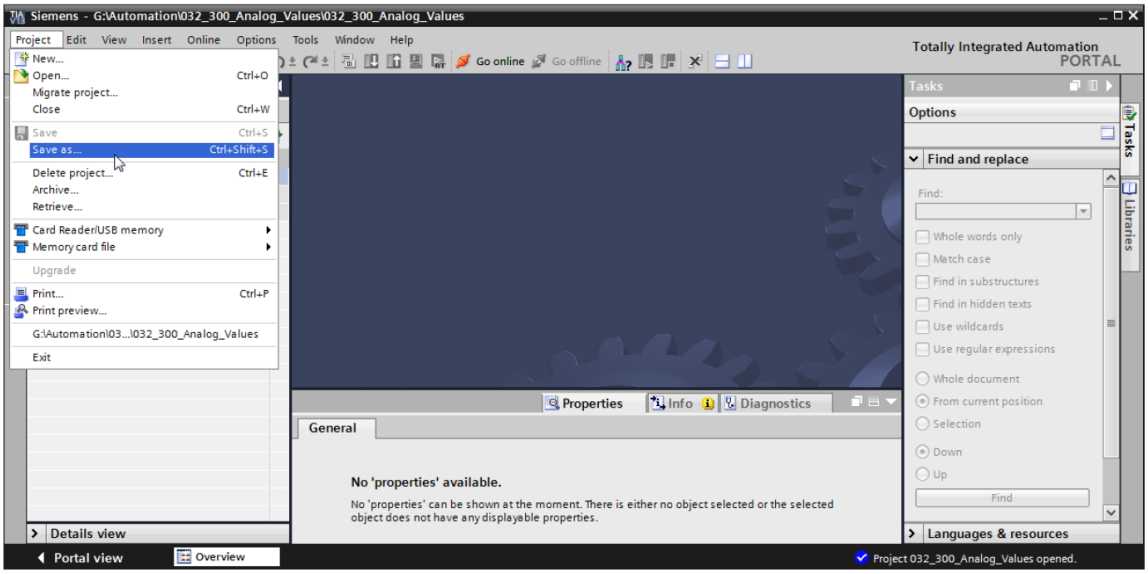


→ 그 다음으로 이 프로젝트가 저장될 대상 디렉토리를 선택합니다. "OK"를 눌러 선택합니다.

(→ Target directory → OK)

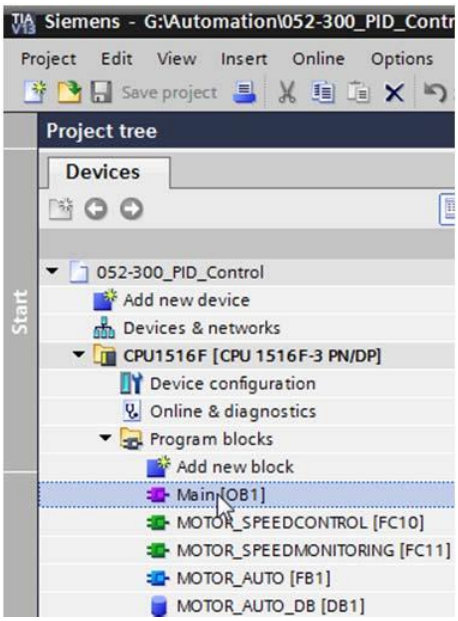
→ 열린 프로젝트를 052-300_PID_Controller라는 이름으로 저장을 합니다.

(→ Project → Save as.. → 052-300_PID_Controller → Save)



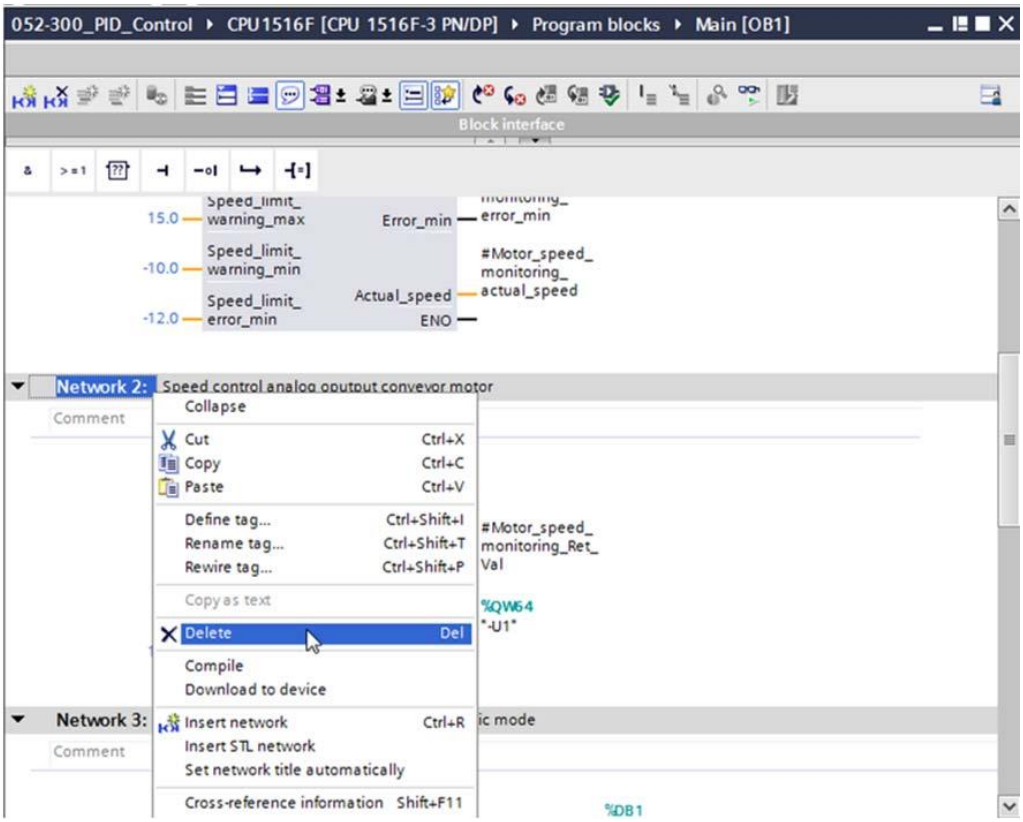
7.2 주기적 인터럽트 OB에서 PID_Compact 컨트롤러 호출

→ "Main [OB1]" 오거나이제이션 블록을 더블클릭해서 엽니다.



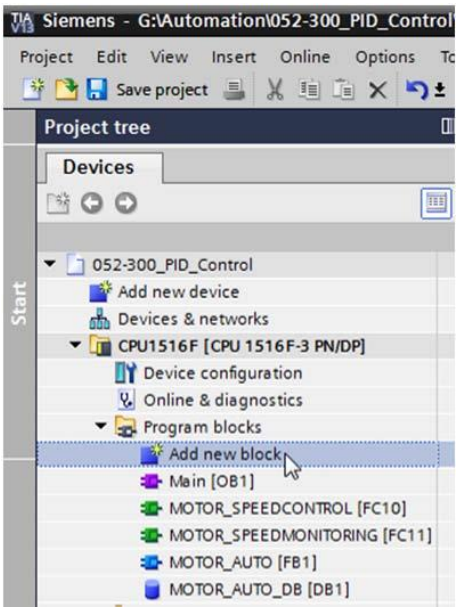
→ "MOTOR_SPEEDCONTROL" [FC10] 펑션 호출이 더 이상 필요하지 않기 때문에 네트워크 2를 삭제합니다.

(→ Network 2 → Delete)



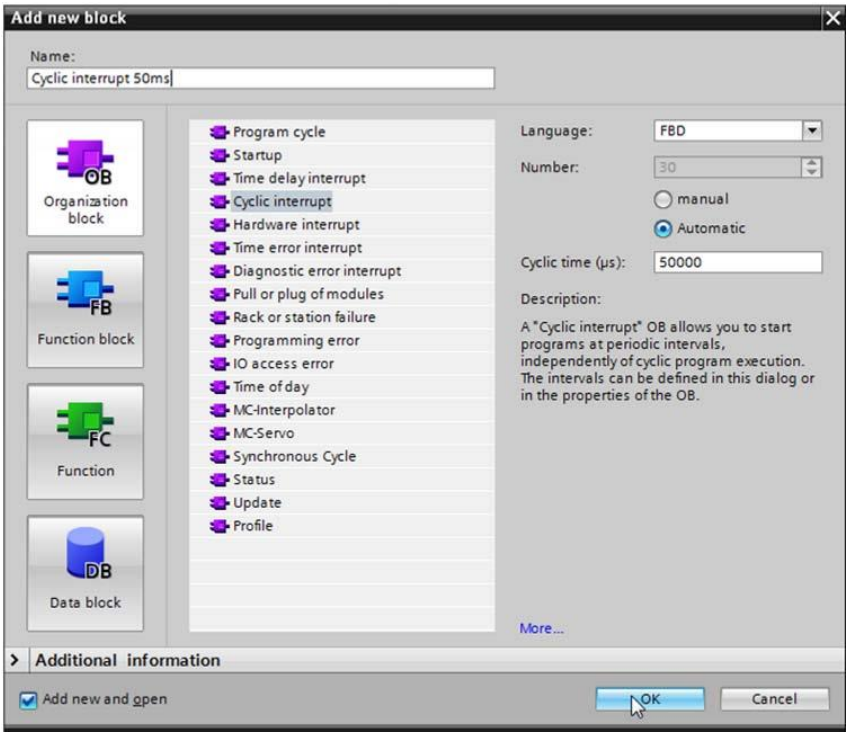
→ PID_Compact 컨트롤러를 호출하려면 주기적 인터럽트 OB가 필요합니다. 따라서 'Program blocks' 폴더에서 "Add new block" 항목을 선택합니다.

(→ Program blocks → Add new block)



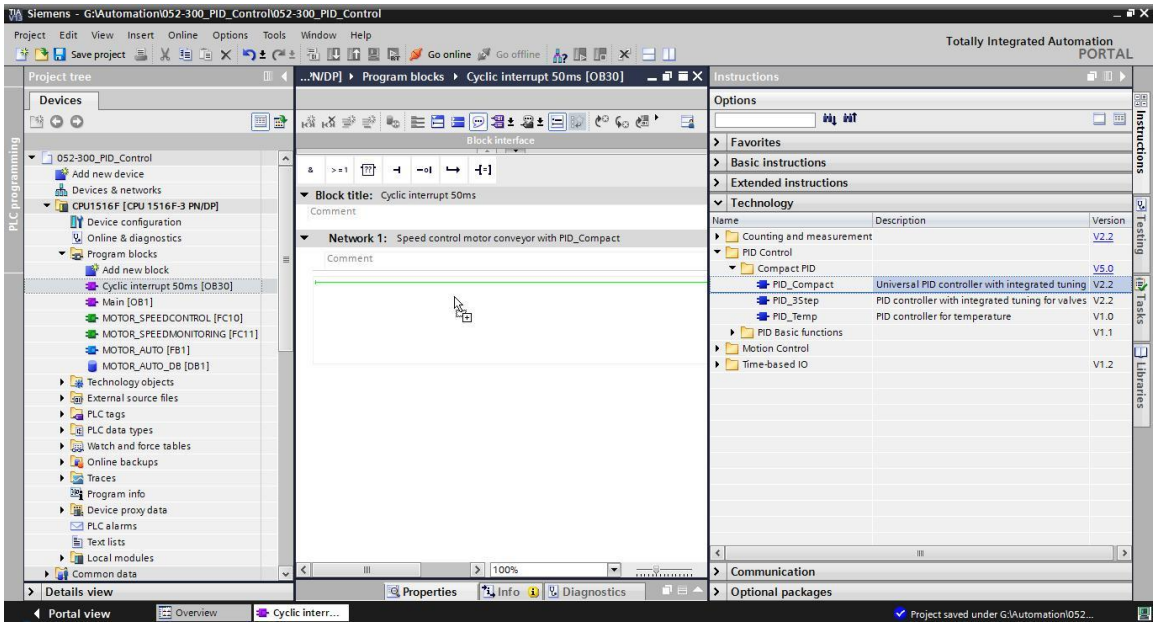
→ 다음 대화상자에서  를 선택하고 주기적 인터럽트 OB의 이름을 "Cyclic interrupt 50ms"로 변경합니다. 언어를 FBD로 설정하고 주기적 인터럽트 시간으로 "50000 µs"를 지정합니다. "Add new and open" 체크 박스를 선택합니다. 'OK'를 클릭합니다.

(→  → Name: Cyclic interrupt 50ms → Language: FBD → Cyclic time (ms): 50000 → ☑ Add new and open → OK)



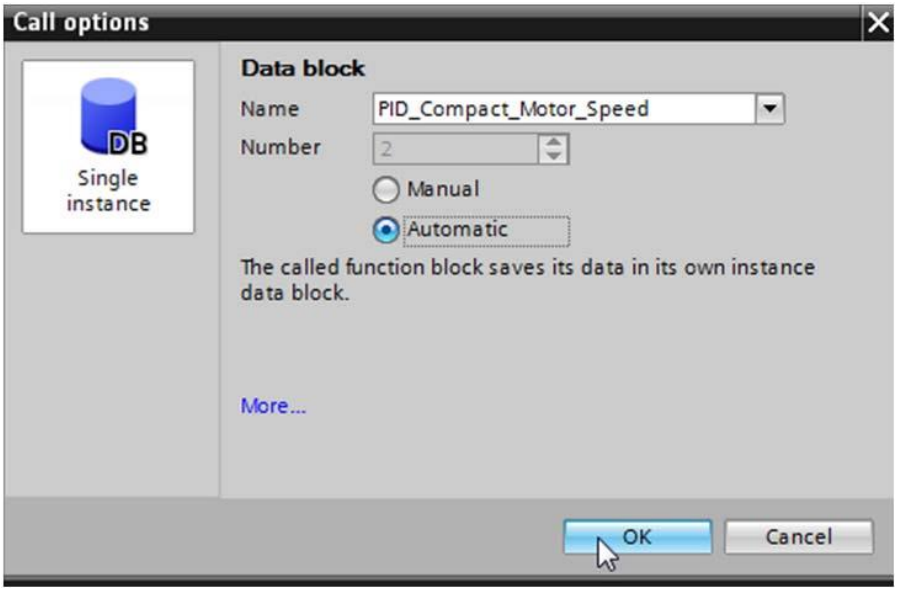
→ 이제 블록이 직접 열립니다. Network 1에 코멘트를 입력하고 끌어다 놓기 기능을 이용해 'PID_Compact' 테크날러지 오브젝트를 네트워크 1으로 이동시킵니다.

(→ Technology → PID Control → Compact PID → PID_Compact)



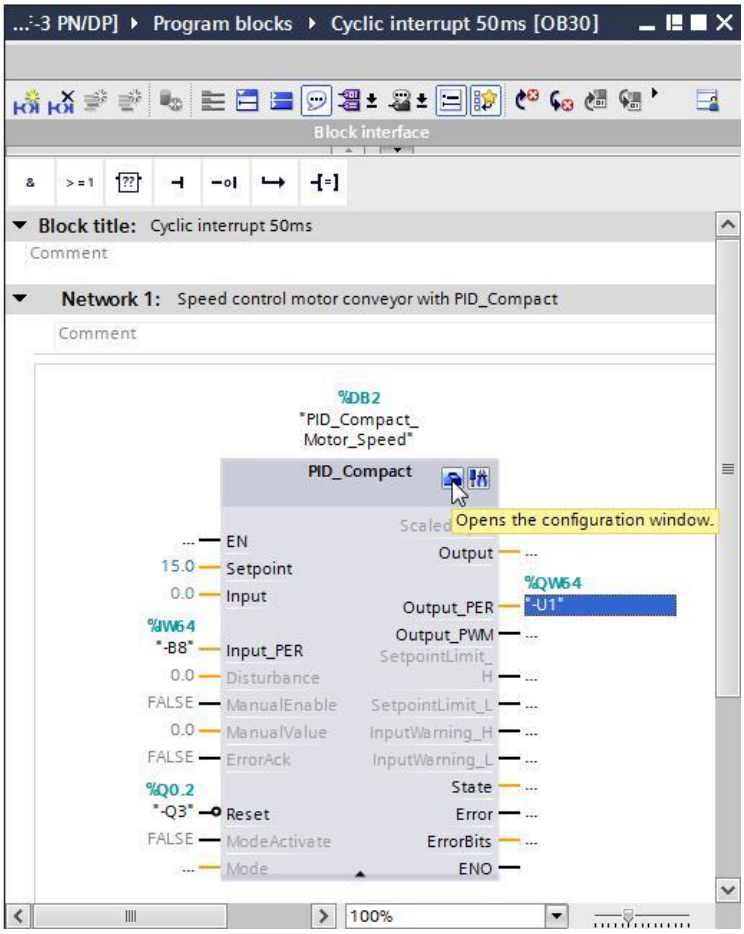
→ 인스턴스 데이터 블록에 대한 이름을 지정하고 OK를 클릭해 이를 적용합니다.

(→ PID_Compact_Motor_Speed → OK)



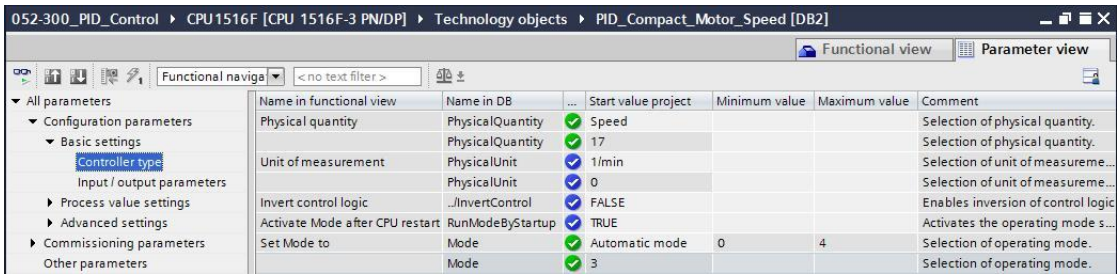
→ 화살표를 클릭해 블록의 뷰를 확장합니다. 아래와 같이 이 블록을 설정값 (상수: 15.0), 실제 값 (글로벌 태그 "-B8"), 조작 변수 (글로벌 태그 "-U1") 및 컨트롤러를 비활성화하기 위한 "Reset" 입력 (글로벌 태그 "-Q3")을 연결하고, "Reset" 입력을 부정화합니다. 그러면, 컨트롤러에 대한 구성 마스크 를 열 수 있습니다.

(→ → 15.0 → "-B8" → "-U1" → -Q3 → →)



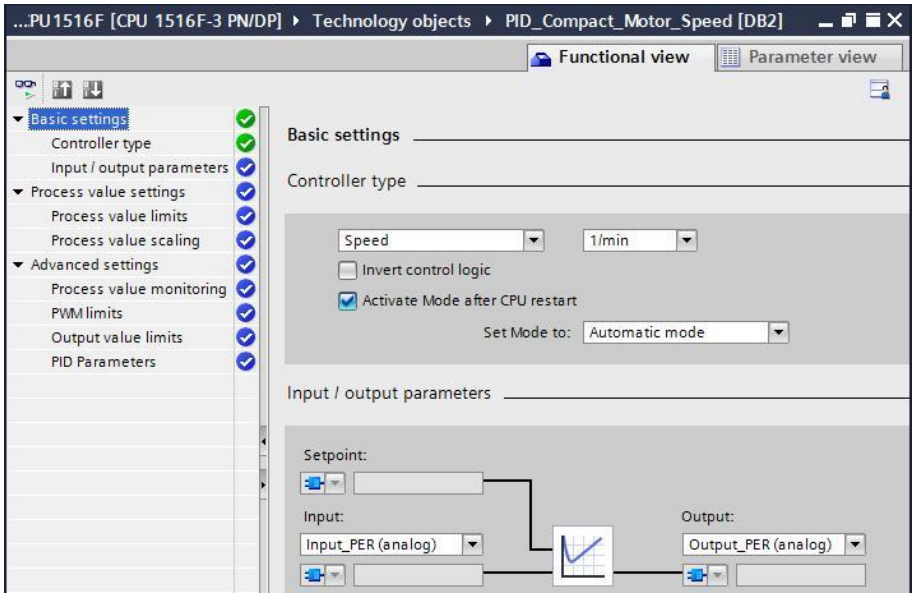
→ 컨트롤러 구성을 볼 수 있도록 "Parameter view"와 "Function view" 2개의 뷰가 제공됩니다. 여기서는 이해가 더 쉬운 "Function view"를 사용하겠습니다.

(→ 펑션 뷰)



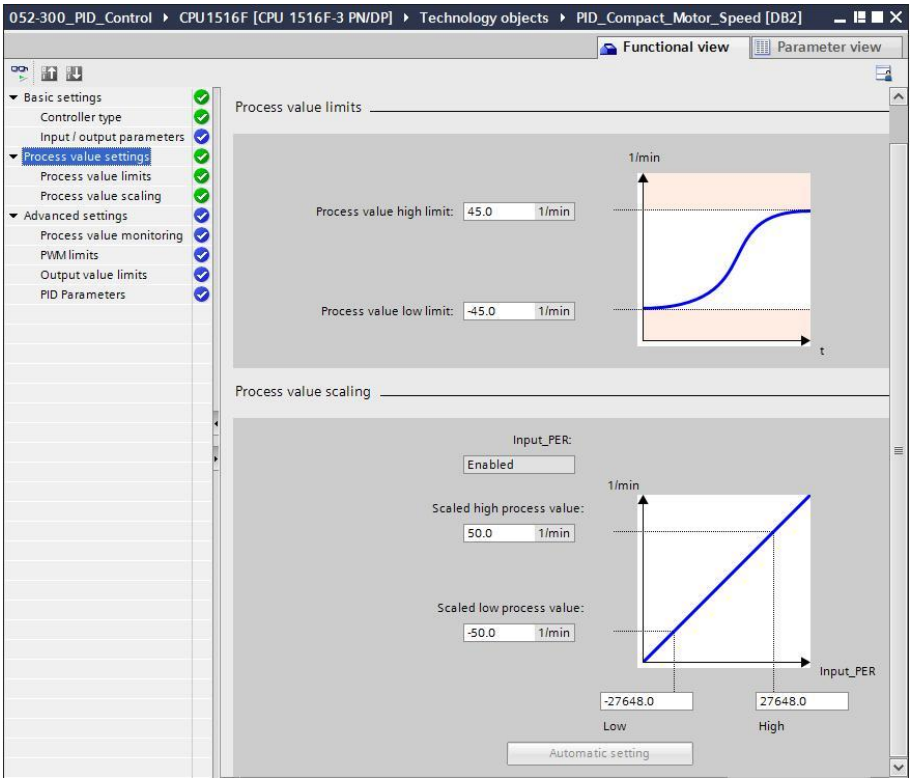
→ "Basic settings"에서 "Controller type" 및 "Input/output parameters"로 이동하여 아래와 같이 값을 설정합니다.

(→ Basic settings → Controller type → Input/output parameters)



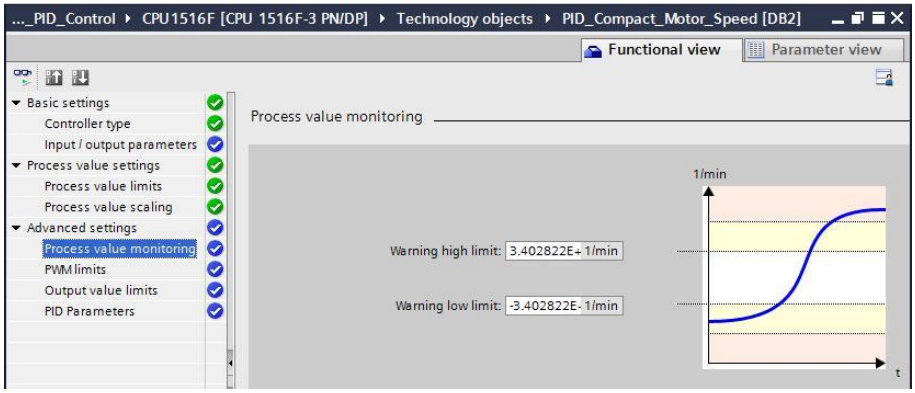
→ "Process value settings"에서 범위 +/- 50 rpm으로 스케일링을 수행하고 "Process value limits"을 +/- 45 rpm으로 설정합니다.

(→ Process value settings → Process value limits → Process value scaling)



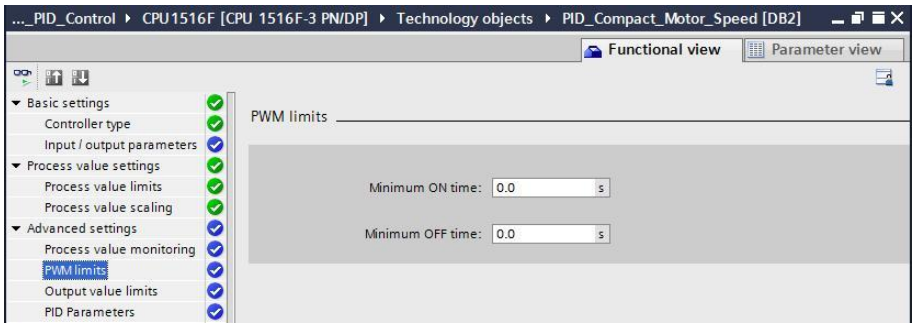
→ "Advanced settings"에서는 프로세스 값 모니터링이 가능하지만, 여기서는 이 내용을 다루지 않겠습니다.

(→ Advanced settings → Process value monitoring)



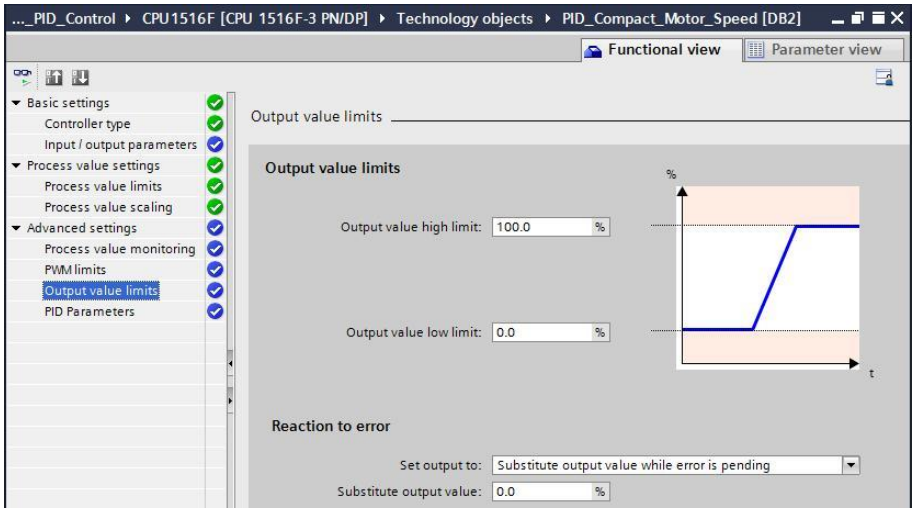
→ "Advanced settings"의 "PWM(Pulse Width Modulation)"는 기본 값으로 그대로 남겨 둡니다. 이 값에 대한 출력이 이 프로젝트에서는 필요하지 않기 때문입니다.

(→ 고급 설정 → PWM)



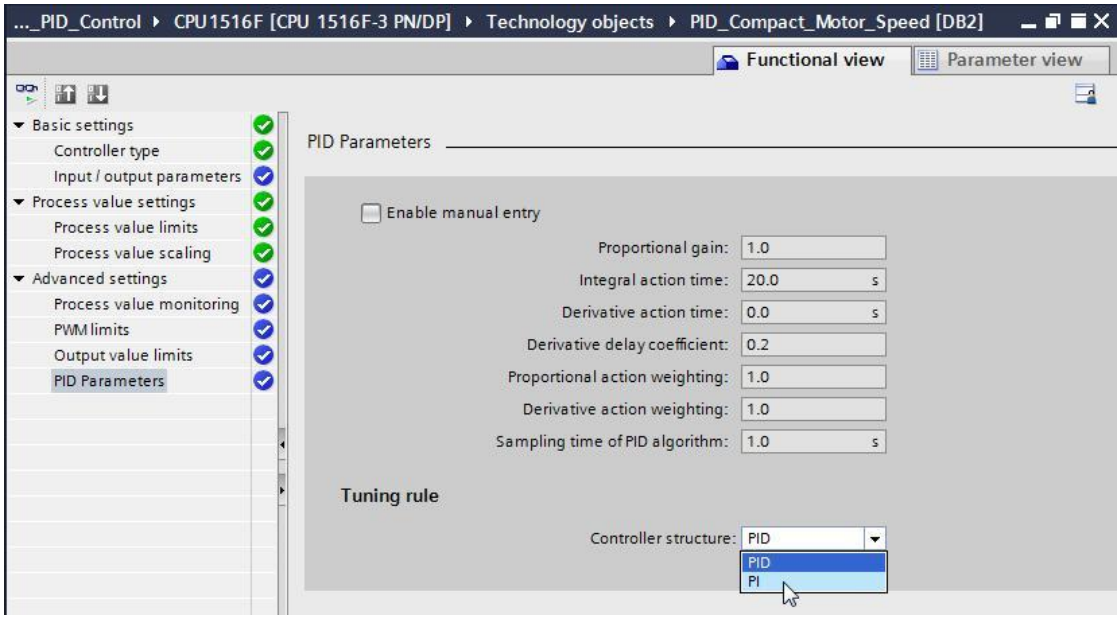
→ "Advanced settings"에서 "Output value limits"를 0.0 %에서 100.0 %로 정의합니다.

(→ Advance settings → Output value limits)



→ "Advance settings"에서는 "PID Parameter"에 대한 수동 설정도 가능합니다. 컨트롤러 구조를 'PI'로 바꾸고 ✕를 클릭하여 구성 창을 닫으면 펑션 PID 컨트롤러가 완성됩니다. 그러나 작동 중에 온라인으로 시운전 및 튜닝의 과정이 남아있습니다.

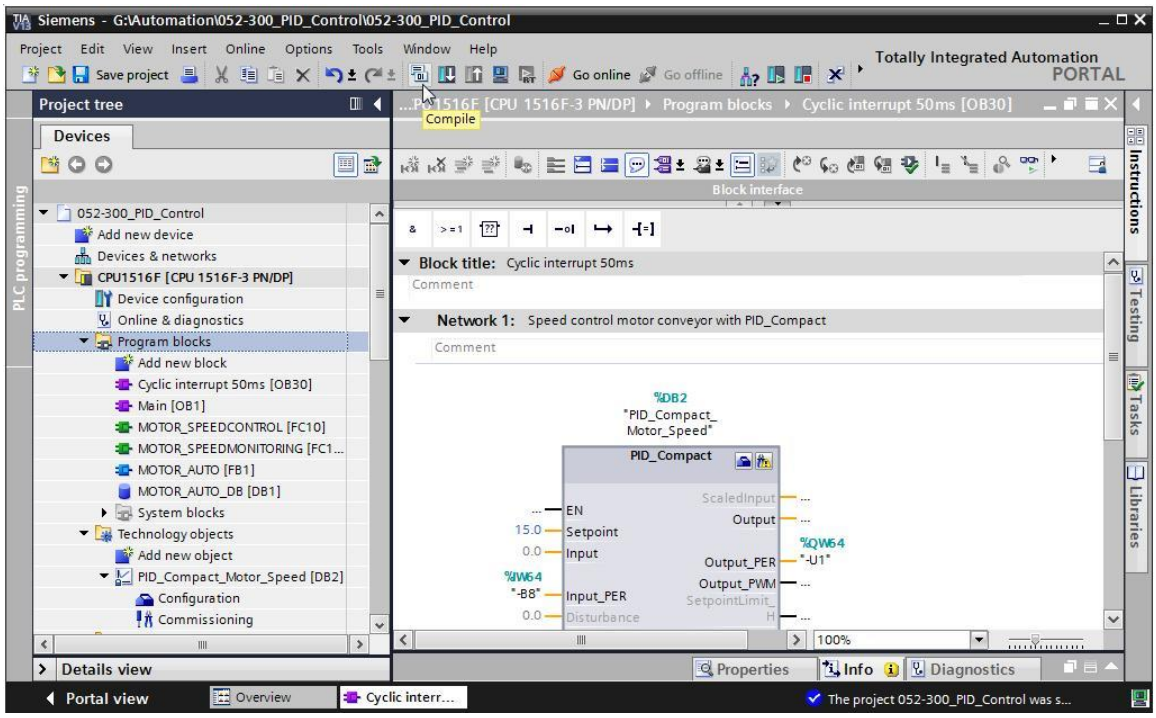
(→ Advanced settings → PID Prameters → Controller structure: PI → ✕)



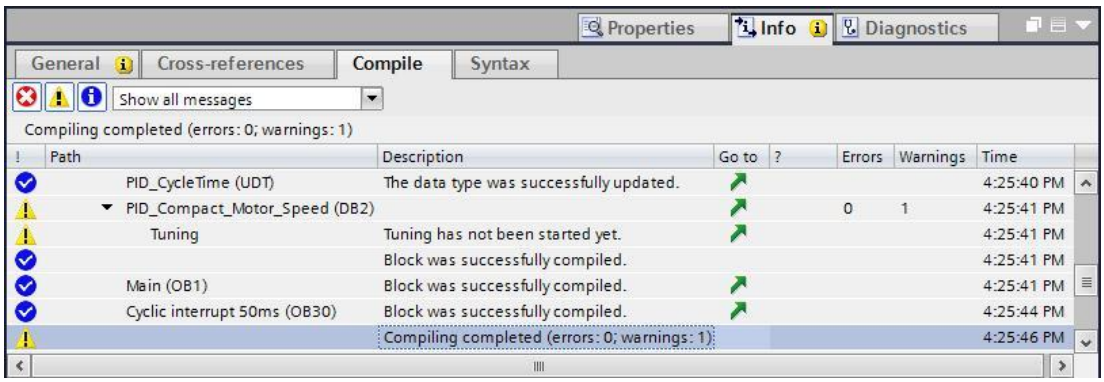
7.3 프로그램 저장 및 컴파일

→ 프로젝트를 저장하려면 메뉴에서 Save project 버튼을 선택합니다. 모든 블록을 컴파일하려면 "Program blocks" 폴더를 클릭하고 메뉴에서 컴파일을 위한 아이콘을 선택합니다.

(→ Save project → Program blocks →)

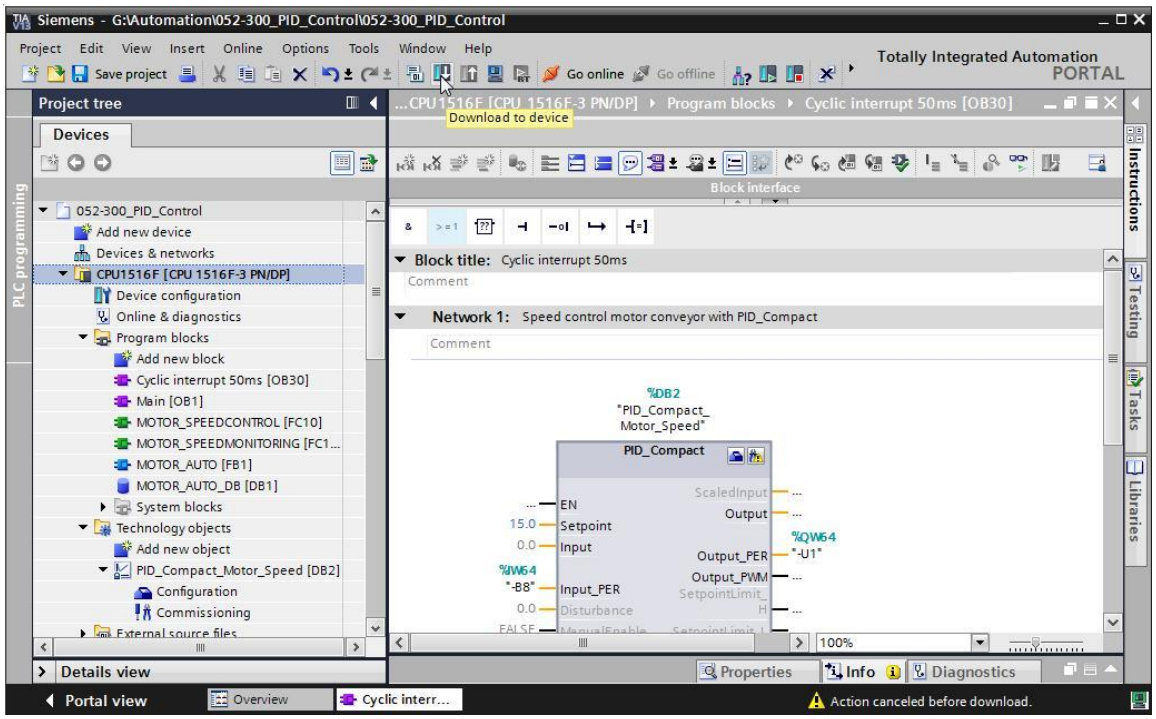


→ "Info" 아래의 "Compile" 영역에 블록이 성공적으로 컴파일이 되었는지 나타납니다.



7.4 프로그램 다운로드

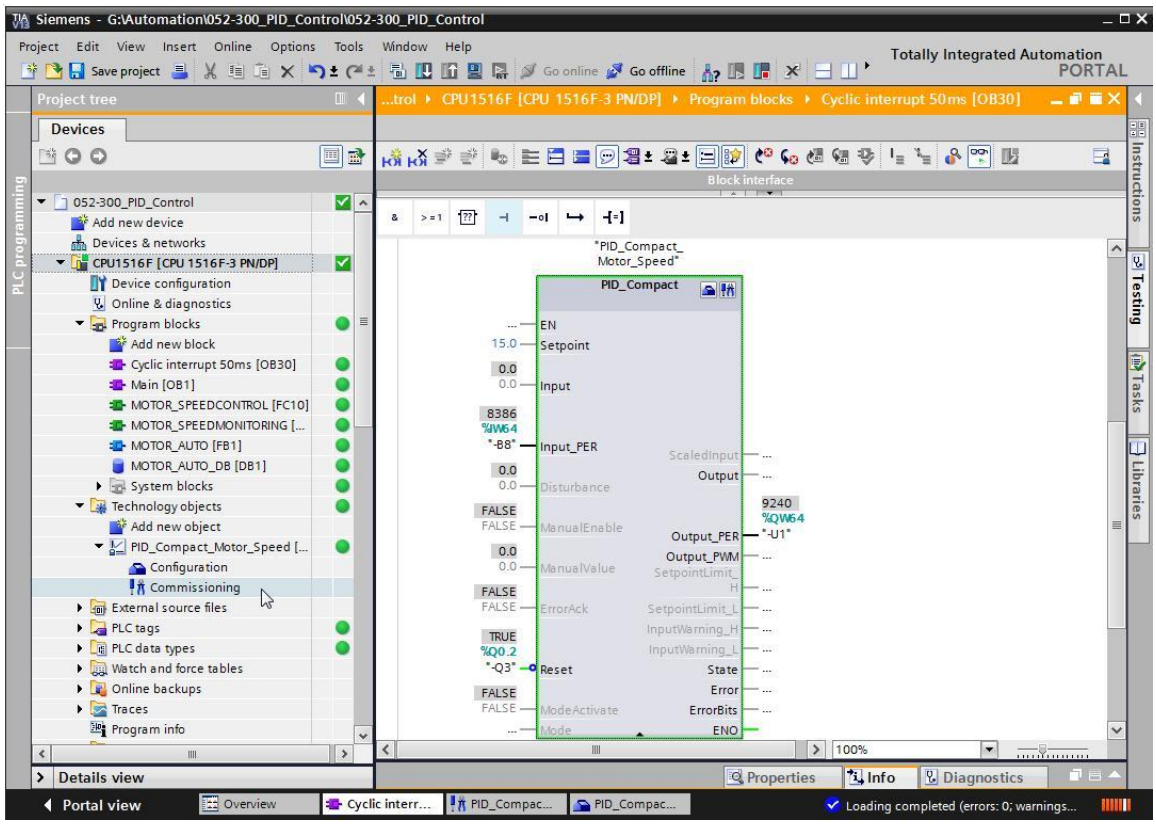
→ 컴파일이 성공적으로 완료되고 나면 앞서 설명한 하드웨어 구성을 포함해 생성된 프로그램과 함께 전체 컨트롤러를 다운로드할 수 있습니다. (→)



7.5 PID_Compact 모니터링

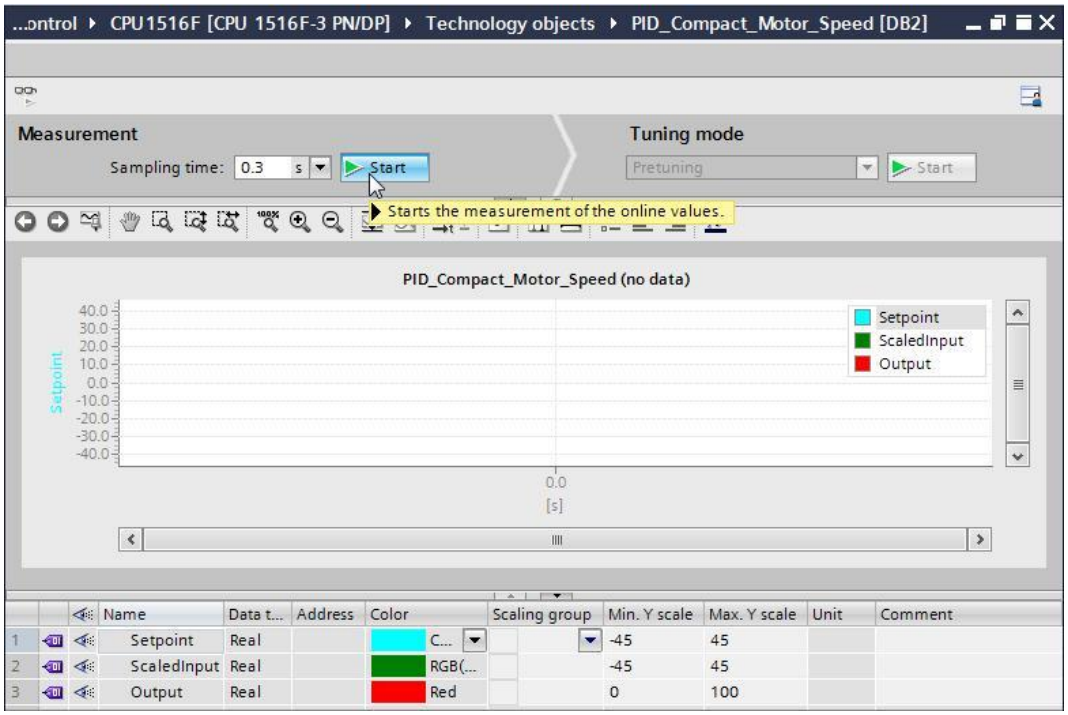
→ 모니터링 온/오프 아이콘  을 클릭하여 프로그램을 테스트할 때 블록 및 태그의 상태를 모니터링합니다. 그러나 CPU가 처음 시작될 때 'PID_Compact' 컨트롤러는 아직 튜닝이 되지 않은 상태입니다. 따라서 Commissioning 아이콘을 클릭하여 튜닝을 시작해야 합니다.

(→ Cyclic interrupt 50ms [OB30] →  → PID_Compact → Commissioning)



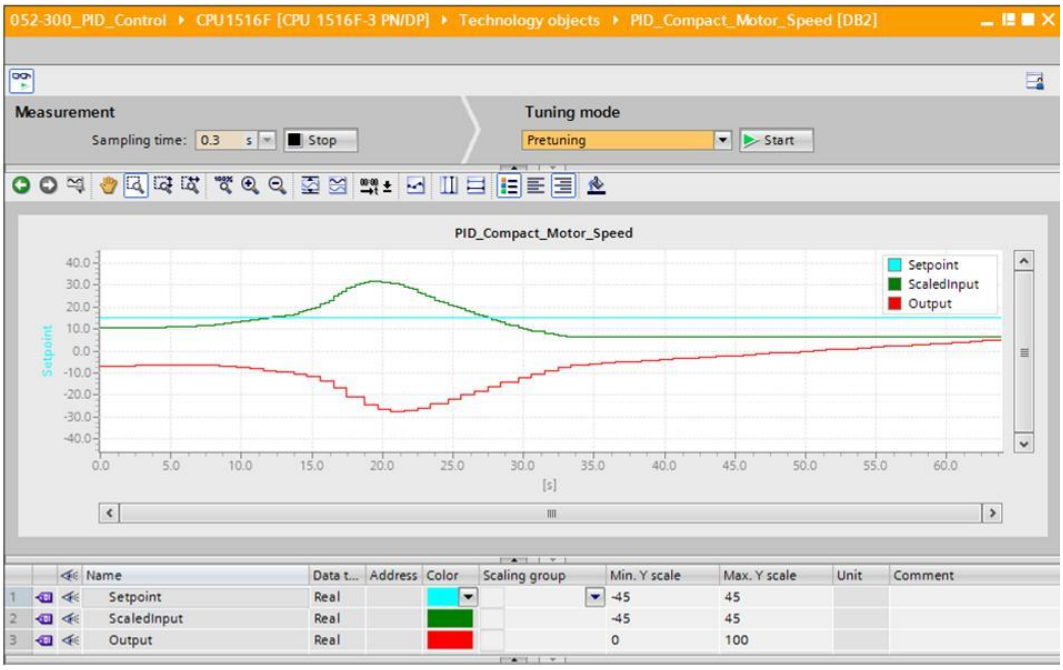
→ "Measurement" 아래의 Start 를 클릭하면 설정값 (Setpoint), 실제 값 (ScaledInput) 및 조작 변수 (Output)를 다이어그램에 표시하고 모니터링할 수 있습니다.

(→ Start)



→ Stop 를 클릭하면 측정을 정지시킬 수 있습니다.

(→ Stop)



7.6 PID_Compact 사전 튜닝

사전 튜닝은 출력 값의 단계적 변경에 대한 프로세스 응답을 판단해서 튜닝 지점을 찾습니다. 제어 대상 시스템의 최대 기울기와 정지 시간을 토대로 PID 파라미터가 계산됩니다. 사전 튜닝과 세부 튜닝을 수행하면 최적의 PID 파라미터가 확보됩니다.

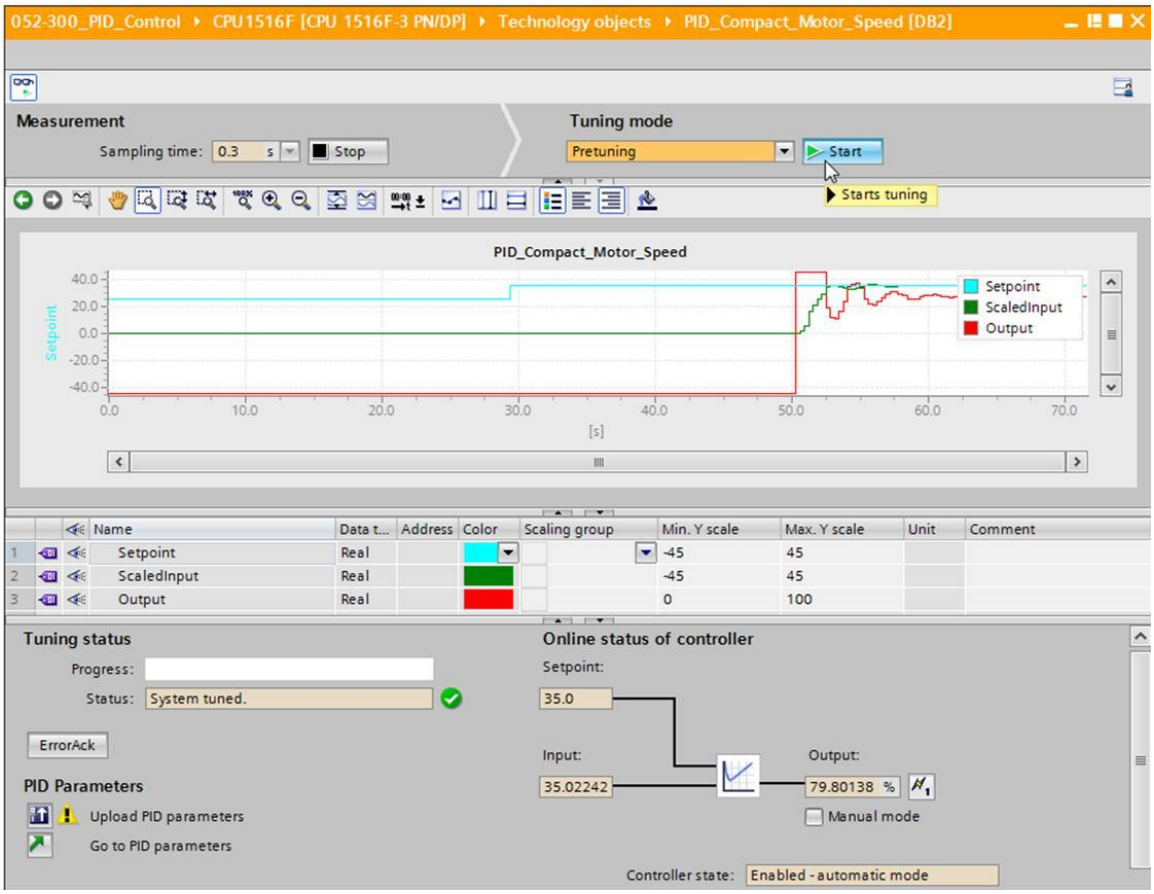
실제 값이 안정적일수록 보다 정확하게 PID 파라미터를 결정할 수 있습니다. 노이즈보다 실제 값의 증가량이 훨씬 큰 동안에는 실제 값 노이즈를 용인할 수 있습니다. 이는 작동 모드가 "비활성" 또는 "수동 모드"인 경우가 대부분 입니다. PID 파라미터는 다시 계산되기 앞서 백업이 됩니다.

다음 요구사항을 반드시 충족해야 합니다.

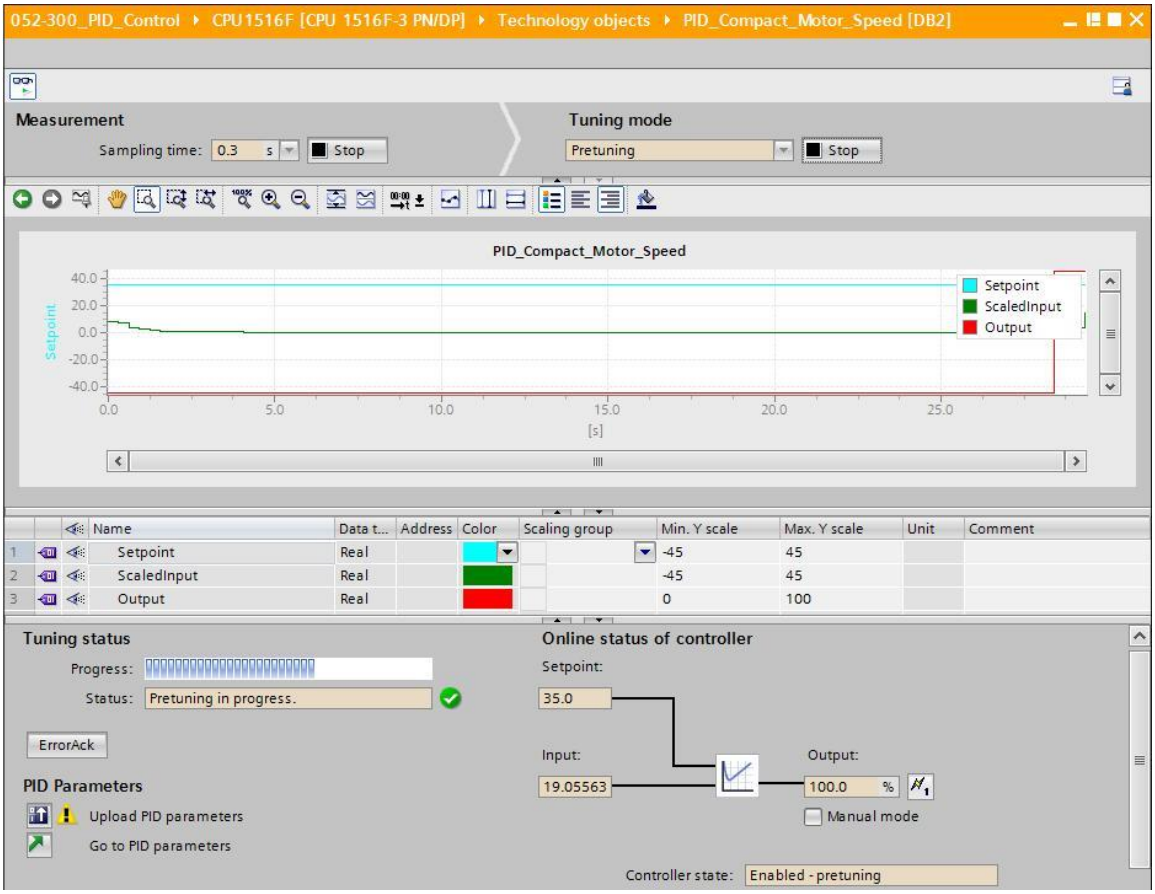
- 주기적 인터럽트 OB에서 "PID_Compact" 명령이 호출됩니다.
- ManualEnable = FALSE
- Reset = FALSE
- PID_Compact의 작동 모드는 "수동 모드", "비활성" 또는 "자동 모드"입니다.
- 설정값과 실제 값은 구성된 한계값 내에 있습니다 ("프로세스 값 모니터링" 구성을 참조).
- 설정값과 실제 값의 차이는 프로세스 값 상한과 하한 차이의 30% 이상입니다.
- 설정값과 실제 값의 차이는 설정값의 50 % 이상입니다.

→ "Tuning mode"에서 "Pretuning"을 선택합니다.

(→ 조정 모드 → 사전 조정 조정 → Start)



→ 사전 튜닝이 시작됩니다. 현재 작업 단계와 발생한 모든 오류가 "Tuning status" 필드에 표시됩니다. Progress 바에 현재 작업 단계의 진행 과정이 표시됩니다.



7.7 PID_Compact 미세 튜닝

미세 조정은 실제 값에 대해 일정하면서도 제한된 발산을 생성합니다. 이러한 발산의 진폭과 주기를 토대로 작동 지점에 맞게 PID 파라미터가 최적화됩니다. 그 결과에 따라 모든 PID가 다시 계산이 됩니다. 일반적으로 미세 튜닝의 결과로 나온 PID 파라미터가 사전 튜닝을 통해 나온 PID 파라미터보다 설정값 변경 및 방해에 훨씬 신속하게 응답합니다. 사전 튜닝과 세부 튜닝을 수행하면 최적의 PID 파라미터가 확보됩니다.

PID_Compact는 자동적으로 실제 값 노이즈보다 큰 발산을 생성하려고 합니다. 미세 튜닝은 실제 값의 안정성에 약간만 영향을 받습니다. PID 파라미터는 다시 계산되기 앞서 백업이 됩니다.

다음 요구사항을 반드시 충족해야 합니다.

- 주기적 인터럽트 OB에서 "PID_Compact" 명령이 호출됩니다.
- ManualEnable = FALSE
- Reset = FALSE
- 설정값과 실제 값은 구성된 한계값 내에 있습니다.
- 작동 지점에서 제어 루프는 안정적입니다. 실제 값이 설정값과 같을 때 작동 지점에 도달합니다.
- 방해는 발생하지 않습니다.
- PID_Compact의 작동 모드는 "수동 모드", "비활성" 또는 "자동 모드"입니다.

자동 모드에서 시작할 때는 다음과 같이 미세 튜닝이 실행됩니다.

튜닝을 통해 기존의 PID 파라미터를 개선하고 싶다면 자동 모드에서 미세 튜닝을 시작합니다.

제어 루프가 안정 상태가 되고 미세 튜닝을 위한 요구사항이 충족될 때까지 PID_Compact는 기존의 PID 파라미터를 사용해 제어를 수행합니다. 그래야만 미세 튜닝이 시작됩니다.

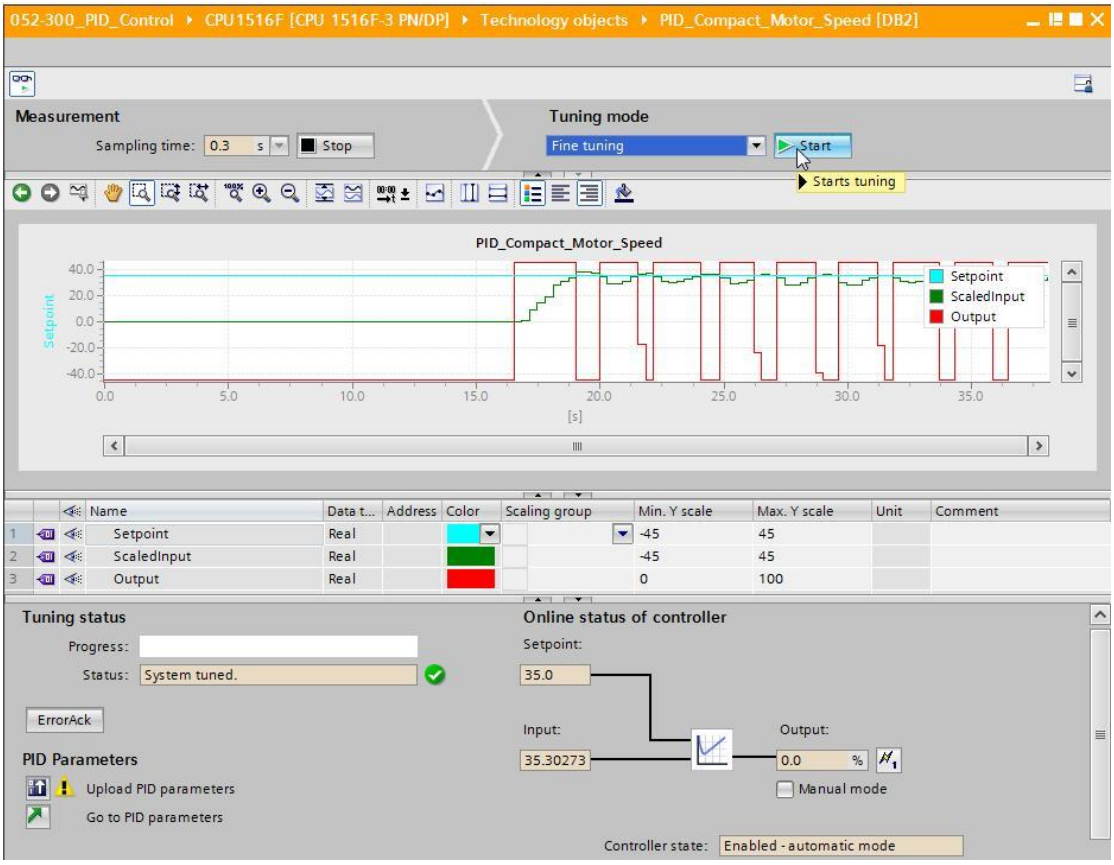
비활성 또는 수동 모드에서 시작할 때는 다음과 같이 미세 튜닝이 실행됩니다.

사전 튜닝을 위한 요구사항이 충족되면 사전 튜닝이 시작됩니다. 제어 루프가 안정 상태가 되고 미세 튜닝을 위한 요구사항이 충족될 때까지 PID_Compact는 결정된 PID 파라미터를 사용해 제어를 수행합니다. 그래야만 미세 튜닝이 시작됩니다. 사전 튜닝이 불가능한 경우에는 PID_Compact는 "Response to error"에 설정된 대로 응답을 합니다.

실제 값이 이미 사전 튜닝 설정값에 너무 가까운 경우에는 최소 또는 최대 출력 값을 통해 설정값에 도달하기 위한 시도가 이루어집니다. 이로 인해 오버슈트가 증가할 수 있습니다.

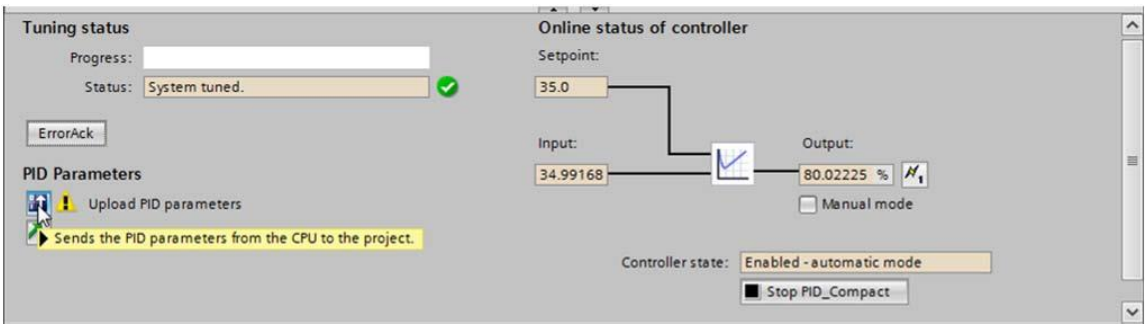
→ "Tuning mode"에서 "Fine tuning"을 선택합니다.

(→ Tuning mode → Fine tuning → Start)



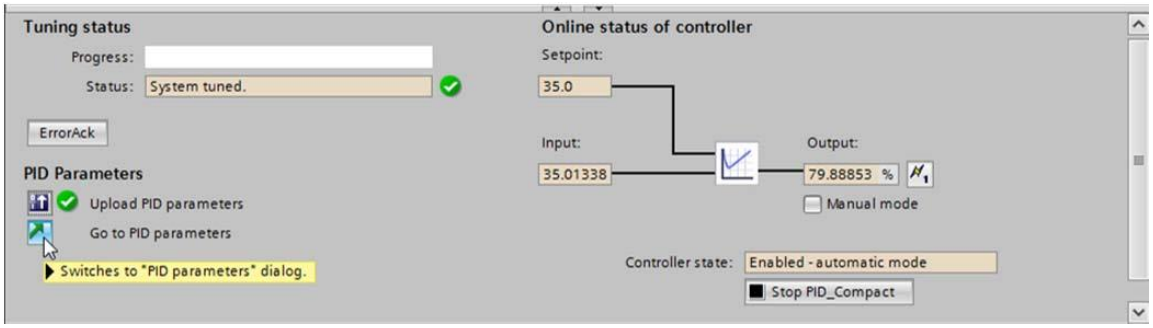
→ 세부 튜닝이 시작됩니다. 현재 작업 단계와 발생하는 모든 오류가 "Tuning status" 필드에 표시됩니다. 오류 메시지 없이 자체 조정이 완료되면 PID 파라미터가 튜닝된 것입니다. PID 컨트롤러는 자동 모드로 전환되면서 튜닝된 파라미터를 사용합니다. 튜닝된 PID 파라미터는 전원을 켜고 CPU를 재시작할 때도 그대로 유지됩니다. 버튼을 클릭해서 CPU에서 프로젝트로 PID 파라미터를 다운로드할 수 있습니다.

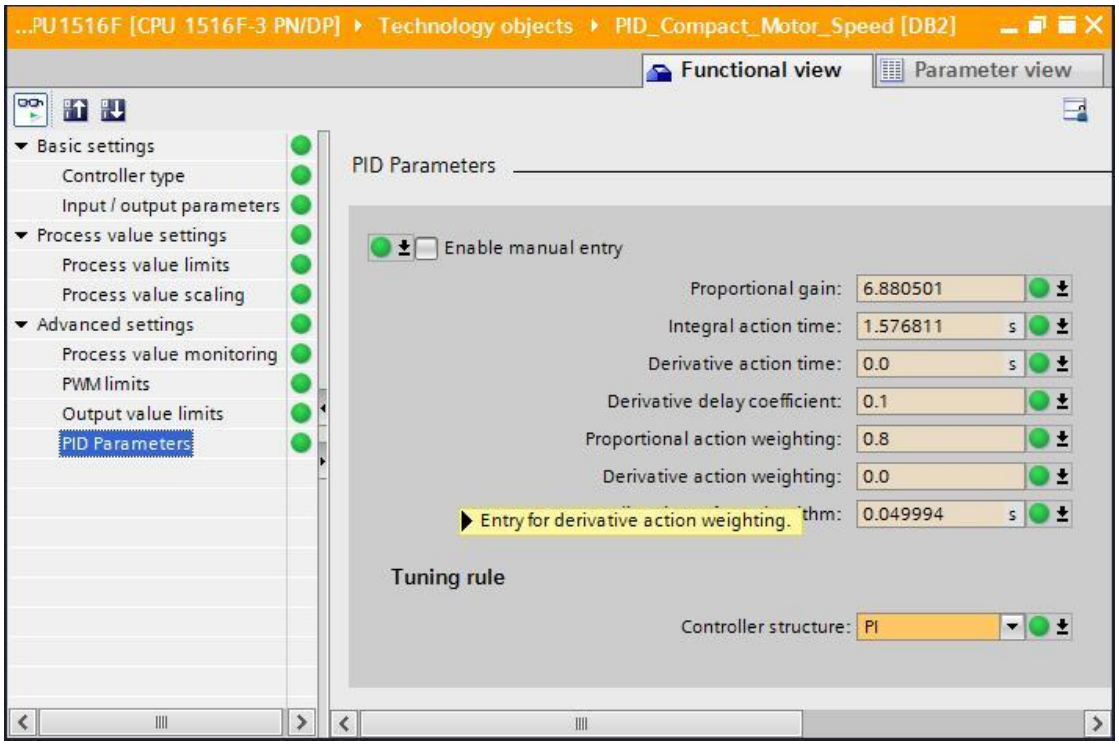
(→)



→  를 클릭하면 구성의 PID 파라미터를 표시할 수 있습니다.

(→ )





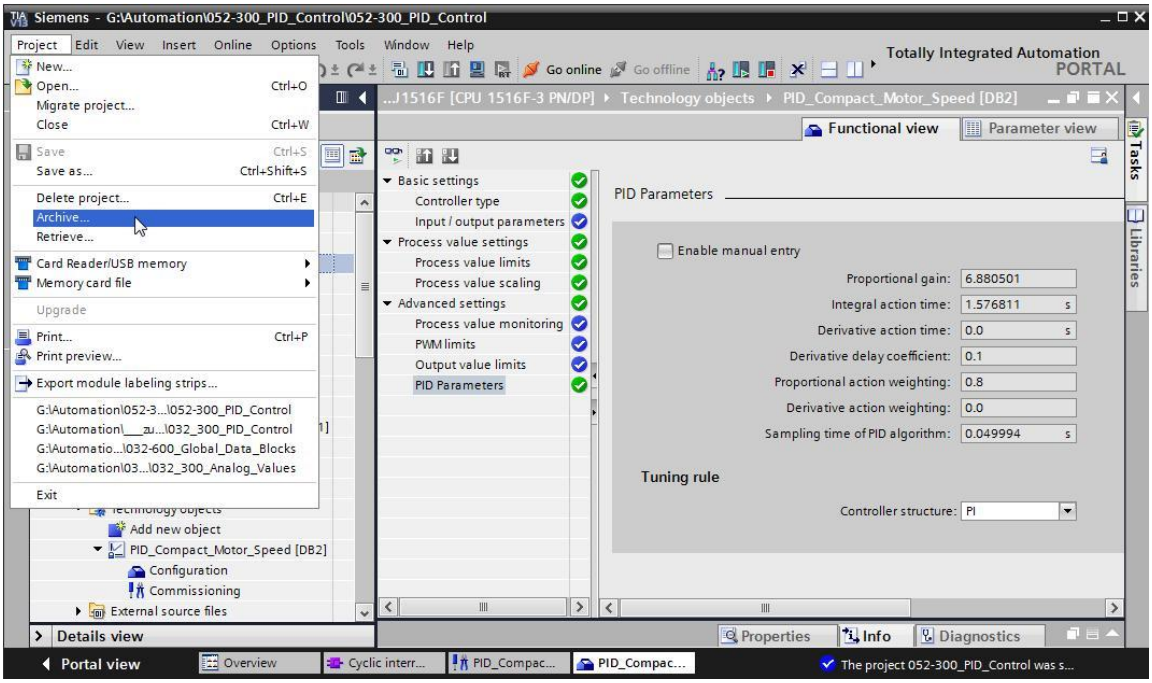
→ 마지막 단계로, 온라인 연결을 끊고 전체 프로젝트를 저장해야 합니다.

(→ Go offline → Save project)

7.8 프로젝트 아카이브

→ 이제 완료된 프로젝트를 아카이브하고자 합니다. "Project" 메뉴에서 "Archive..." 항목을 선택합니다. 프로젝트를 아카이브하고자 하는 폴더를 선택하고 "TIA Portal 프로젝트 아카이브" 파일 유형으로 이를 저장합니다.

(→ Project → Archive → TIA Portal project archive → 052-300_PID_Controller…. → Save)



8 체크리스트

번호	설명	완료
1	주기적 인터럽트 OB Cyclic interrupt 50ms [OB30]이 성공적으로 생성	
2	주기적 인터럽트 OB Cyclic interrupt 50ms [OB30]에서 PID_Compact 컨트롤러 호출 및 연결	
3	PID_Compact 컨트롤러 구성이 수행	
4	오류 메시지 없이 성공적으로 컴파일	
5	오류 메시지 없이 성공적으로 다운로드	
6	오류 메시지 없이 성공적으로 사전 조정	
7	오류 메시지 없이 성공적으로 세부 조정	
8	스테이션 전원 켜기 (-K0 = 1) 실린더 복귀 / 피드백 활성화 (-B1 = 1) 비상 정지 오프 (-A1 = 1)가 활성화되지 않음 자동 모드 (-S0 = 1) 푸시버튼 자동 정지가 구동되지 않음 (-S2 =1) 자동 시작 푸시버튼을 짧게 누르기 (-S2 = 1) 슬라이드의 센서 부분이 활성화되고 (-B4 = 1) 컨베이어 모터 -M1 가변 속도 (-Q3 = 1) 스위치를 켠 상태로 유지 속도는 +/- 50 rpm 범위에서의 속도 설정값에 해당	
9	컨베이어 끝의 센서 활성화 (-B7 = 1) → -Q3 = 0 (2초 후)	
10	자동 정지 푸시버튼을 짧게 누르기 (-S2 = 0) → -Q3 = 0	
11	비상 정지 오프를 활성화 (-A1 = 0) → -Q3 = 0	
12	수동 모드 (-S0 = 0) → -Q3 = 0	
13	스테이션 전원 끄기 (-K0 = 0) → -Q3 = 0	
14	실린더가 복귀되지 않음 (-B1 = 0) → -Q3 = 0	
15	속도 > Motor_speed_monitoring_error_max → -Q3 = 0	
16	속도 < Motor_speed_monitoring_error_min → -Q3 = 0	
17	프로젝트가 성공적으로 아카이브 됨	

9 추가 정보

초기 및 심화 교육에 방향을 제시하는 도구의 차원에서 TIA Portal 모듈에 대한 추가 정보를 활용할 수 있습니다. 시작하기, 동영상, 교재, 앱, 매뉴얼, 프로그래밍 지침, 체험용 소프트웨어/펌웨어 등을 아래 링크에서 찾아보실 수 있습니다.

www.siemens.com/sce/s7-1500