기능 모듈 명칭	유형	개발 상황	세부정보	중요도 상태	파일	메모, 세부 설정 (예)
PWM DC 릴레이 출력	Ouput 출력 모듈	기본 릴레이 제어만 가능 PWM 미작동	DC 릴레이를 통해 전자밸브를 제어하는 것을 주 목적으로 한다. 보드에서 PWM (Pulse Width Modulation)을 통해 밸브의 열림과 닫힘을 제어하며, 펄스 폭을 조절하여 Duty 값으로 전압을 제어하는 방식입니다. 이를 통해 밸브의 열림 정도를 정밀하게 조절할 수 있다. 기본 명령은 0 과 100% 로 제어 된다.	필수 IO 프로그램	파일	PWM_Frequency (float): PWM 신호의 주파수, 밸브의 반응 속도 결정 (예: PWM_Frequency = 1000;) PWM_DutyCycle (float): PWM 신호의 듀티 사이클, ON/OFF 비율로 밸브 개도 조절 (예: PWM_DutyCycle = 50;) Relay_OutputPin (int): DC 릴레이 출력이 연결된 핀 번호 (예: Relay_OutputPin = 5;) Valve_State (bool): 밸브의 현재 상태, 열림(ON) 또는 닫힘(OFF) 기록 (예: Valve_State = true;) Valve_TargetPosition (float): 밸브 목표 위치, 설정된 개도 값(%) (예: Valve_TargetPosition = 75.0;) Relay_CurrentStatus (bool): 릴레이의 현재 상태, 활성화 여부(ON/OFF) 추적 (예: Relay_CurrentStatus = false;)
알람 버즈 출력	Ouput 출력 모듈	작동	기본 알림 버저 프로그램은 시스템 오류, 경고, 프로세스 완료 등의 이벤트 시 청각적 피드백을 제공하는 기능이다. 디지털 핀을 통해 소리의 길이와 주파수를 제어하여 다양한 알림을 설정할 수 있다. 콘트롤 보드 부팅시 1음 1회 버즈 발생, 오류 발생시 3음 회 반복 발생 기타 상황에 따라 설정 예정	필수 IO 프로그램	파일	Buzzer_Pin (int): 버저가 연결된 디지털 핀 번호 (예: Buzzer_Pin = 9;) Buzzer_Frequency (int): 버저 주파수 (Hz 단위) (예: Buzzer_Frequency = 1000;) Buzzer_Duration (int): 버저 울림 시간 (ms 단위) (예: Buzzer_Duration = 500;) Buzzer_State (bool): 버저 상태 (ON/OFF) (예: Buzzer_State = true;) Buzzer_Volume (int): PWM을 통한 버저 음량 조절 (0-255) (예: Buzzer_Volume = 128;) Alert_Type (int): 경고 유형에 따른 알림 설정 (예: Alert_Type = 1;)
AC SSR 제어 출력	Ouput 출력 모듈	작동	AC 히터 제어를 위한 SSR(솔리드 스테이트 릴레이) 출력 제어 프로그램은 AC 히터의 전력을 제어하는 프로그램입니다. PWM 최대 듀티 Duty 를 설정하고 PID 분석을 통해 작동한다. PID와 연동하지만 별도로 Duty를 설정하여 히터의 전력을 줄일 수 있다.	필수 IO 프로그램	파일	SSR_Pin (int): SSR이 연결된 디지털 출력 핀 (예: SSR_Pin = 10;) PWM_Frequency (float): SSR 제어를 위한 PWM 신호 주파수 (예: PWM_Frequency = 1000;) PWM_DutyCycle (float): SSR 제어를 위한 PWM 듀티 사이클, 전력 공급량 결정 (예: PWM_DutyCycle = 50;) Control_LoopTime (int): 제어 루프 주기 (예: Control_LoopTime = 1000;)
K-Type 센서 입력	Input 입력 모듈	작동	K타입 온도 센서는 열전대(thermocouple) 방식으로, 두 금속의 접합부에서 발생하는 열기전력을 통해 온도를 측정합니다. 이 센서의 경우 보정값과 온도 변환 공식이 필요하며, 이를 처리하기 위한 몇 가지 변수가 필요합니다.	필수 IO 프로그램	파일	Thermocouple_Voltage (float): K타입 센서에서 발생한 열전대 전압 (mV) (예: Thermocouple_Voltage = (ADC_Value / 1023.0) * V_Ref;) Cold_Junction_Temp (float): 냉접합 보상 온도 (예: Cold_Junction_Temp = 25.0;) Temperature (float): 변환된 온도 값 (예: Temperature = Thermocouple_Voltage / K_Factor;) K_Factor (float): K타입 열전대 변환 계수 (예: K_Factor = 0.041276;) ADC_Value (int): ADC로부터 읽은 디지털 값 (예: ADC_Value = analogRead(ADC_Pin);) V_Ref (float): ADC 참조 전압 (예: V_Ref = 5.0;) Calibrated_Temperature (float): 보정된 최종 온도 값 (예: Calibrated_Temperature = (Temperature * Gain_Correction) + Offset_Correction;) Offset_Correction (float): 온도 오프셋 보정 값 (예: Offset_Correction = 2.0;) Gain_Correction (float): 온도 이득 보정 값 (예: Gain_Correction = 1.01;)
PT100 센서 입력	Input 입력 모듈	작동	Pt100 온도 센서를 사용해 온도 값을 읽어오는 프로그램은 저항 온도 감지기(RTD)인 Pt100 센서로부터 저항 값을 읽고 이를 온도로 변환하는 과정을 포함합니다. Pt100은 저항이 온도에 비례해 변하며, 이를 기반으로 정확한 온도 측정이 가능합니다	필수 IO 프로그램	파일	ADC_Pin (int): Pt100 센서가 연결된 아날로그 입력 핀 (예: ADC_Pin = A0;) Reference_Resistance (float): 기준 저항 값, 보통 100Ω (예: Reference_Resistance = 100.0;) ADC_Value (int): ADC로부터 읽어들인 디지털 값 (예: ADC_Value = analogRead(ADC_Pin);) Temperature (float): 변환된 온도 값 (예: Temperature = (Resistance - Reference_Resistance) / 0.00385;) Resistance (float): 센서의 현재 저항 값 (예: Resistance = (ADC_Value / 1023.0) * V_Ref / I;) Offset_Correction (float): 센서의 온도 보정 값 (예: Offset_Correction = 1.5;) Gain_Correction (float): 센서의 이득 보정 값 (예: Gain_Correction = 1.02;) Calibrated_Temperature (float): 보정 후 최종 온도 값 (예: Calibrated_Temperature = (Temperature * Gain_Correction) + Offset_Correction;)
ToF 거리 감지 센서 I2C	Input 입력 모듈	작동하지만 로직 미흡	ToF(Time of Flight) 거리 센서를 이용해 포타필터를 감지합니다. ToF 센서는 빛이 물체에 반사되어 돌아오는 시간을 측정해 거리를 계산하는 방식으로, 매우 정밀한 거리 측정이 가능합니다. 포타필터와 같은 물체를 감지하기 위해서는 최대 감지 거리와 민감도를 조정해야 합니다.	필수 IO 프로그램	파일	Detection_MinRange (float): 포타필터 감지를 위한 최소 거리 (예: Detection_MinRange = 5.0;) Detection_MaxRange (float): 포타필터 감지를 위한 최대 거리 (예: Detection_MaxRange = 15.0;) Sensitivity (int): ToF 센서의 감도 설정 (0 ~ 100) (예: Sensitivity = 80;) Threshold_Distance (float): 포타필터 감지 임계 거리 (예: Threshold_Distance = 10.0;) Ambient_Light_Compensation (bool): 주변광 보정 활성화 여부 (예: Ambient_Light_Compensation = true;) Integration_Time (int): 거리 측정 통합 시간 (ms) (예: Integration_Time = 100;) Measured_Distance (float): ToF 센서로 측정된 거리 (cm) (예: Measured_Distance = sensor.getDistance();) Object_Detected (bool): 포타필터 감지 여부 (True/False) (예: Object_Detected = (Measured_Distance < Threshold_Distance);)
4~20mA 압력 센서 입력	Input 입력 모듈	작동하지만 설정 환경 미흡	4~20mA 출력을 지원하는 압력 센서를 사용해 보일러 압력을 측정하는 모듈 프로그램은 센서로부터 전류 신호를 읽어 이를 압력 값으로 변환하고, 보일러의 실시간 압력 상태를 모니터링하는 기능을 제공합니다. 0 ~ 16Bar 를 기준으로 하고 있습니다.	필수 IO 프로그램	파일	ADC_Pin (int): 4~20mA 신호가 연결된 아날로그 핀 (예: ADC_Pin = A0;) ADC_Value (int): 아날로그 핀에서 읽어들인 ADC 값 (예: ADC_Value = analogRead(ADC_Pin);) V_Ref (float): 참조 전압 값 (예: V_Ref = 5.0;) Shunt_Resistor (float): 4~20mA 신호를 전압으로 변환하기 위한 분로 저항 값 (예: Shunt_Resistor = 250.0;) Current_mA (float): 센서로부터 읽은 전류 값(mA) (예: Current_mA = Voltage / Shunt_Resistor * 1000;) Pressure_Min (float): 센서의 최소 압력 값 (예: Pressure_Min = 0.0;) Pressure_Max (float): 센서의 최대 압력 값 (예: Pressure_Max = 150.0;) Pressure (float): 센서에서 계산된 실제 압력 값 (예: Pressure = ((Current_mA - 4.0) / 16.0) * (Pressure_Max - Pressure_Min) + Pressure_Min;) Calibration_Offset (float): 압력 값에 대한 오프셋 보정 (예: Calibration_Offset = 0.0;) Calibration_Scale (float): 이득 보정 값 (예: Calibration_Scale = 1.0;)
근접 센서 NPN 입력 (금속 감지)	Input 입력 모듈	미작동	NPN 방식의 근접 센서는 센서가 물체를 감지할 때 NPN 트랜지스터를 사용하여 신호가 LOW(0V)로 떨어지는 방식입니다. 즉, 근접 센서가 활성화되면 센서 출력이 GND로 연결되며, 이를 통해 디지털 입력 핀에서 LOW 신호를 읽어 센서가 물체를 감지했음을 알 수 있습니다. 이 기능을 통해 패들의 춰를 판단하여 엔코더 모터 작동의 시작 지점을 판단한다.	필수 IO 프로그램	파일	Sensor_Pin (int): NPN 근접 센서가 연결된 디지털 핀 번호 (예: Sensor_Pin = 2;) Sensor_State (bool): 센서의 현재 상태를 저장하는 변수 (예: Sensor_State = digitalRead(Sensor_Pin);) Object_Detected (bool): 물체가 감지되었는지 여부를 나타내는 변수 (예: Object_Detected = (Sensor_State == LOW);) Last_State (bool): 이전 상태를 저장하여 센서 상태 변화를 기록하는 변수 (예: Last_State = HIGH;)
플로우메타 카운터 입력 OC	Input 입력 모듈	작동하지만 로직 미흡	Open Collector 방식의 플로우메타는 회전 속도에 따라 펄스를 발생시키며, 이를 통해 유량을 측정하는 센서입니다. Open Collector 방식은 센서가 펄스 신호를 디지털 핀으로 보내 유량을 측정할 수 있도록 하며, 저항을 통해 외부 전원을 연결해 신호를 수집합니다.	필수 IO 프로그램	파일	Flow_Pin (int): 플로우메타에서 펄스를 읽어올 디지털 핀 (예: Flow_Pin = 2;) Pulse_Count (volatile int): 발생한 펄스 수를 저장하는 변수 (예: Pulse_Count = 0;) Pulses_Per_Liter (float): 리터당 발생하는 펄스 수 (예: Pulses_Per_Liter = 450.0;) Flow_Rate (float): 현재 측정된 유량 (L/min) (예: Flow_Rate = Pulse_Count / Pulses_Per_Liter;) Flow_Volume (float): 누적 유량 (리터) (예: Flow_Volume += Flow_Rate * Time_Interval;) Calibration_Factor (float): 플로우메타 보정 계수 (예: Calibration_Factor = 1.0;) Last_Time (unsigned long): 마지막 유량 측정 시간이 저장되는 변수 (예: Last_Time = millis();)
OC 수위, 누수 감지 센서	Input 입력 모듈	작동 설정 값 미흡	Open Collector 방식으로 수위를 감지하는 센서는 디지털 신호를 통해 수위가 특정 수준에 도달했는지 감지한다. 수위는 Low 낮은 수위, High 높은 수위로 구분하고 급수와 히팅제어와 함께 사용된다.	필수 IO 프로그램	파일	감도 설정
버튼 입력, 버튼 LED 출력 CAN	I/O 통합 모듈	LED 작동 미흡	CAN 통신을 통해 연결된 3 버튼에 추가로 1버튼 모듈을 연결 할 수 있다. 연결 개수와 작동 목적이 정의되어야 한다. 각 버튼은 싱글, 더블, 롱 푸쉬를 구분하고 동시 입력을 지원한다. 상태에 따라 버튼에 연결된 LED의 색상 및 밝기 조정이 가능하다.	필수 IO 프로그램	파일	*메모, 세부 설정 (예)*
패들 제어 엔코더 모터 입출력 CAN	I/O 통합 모듈	미흡	엔코더가 내장된 BLDC 모터를 이용하여 패들(Paddle)의 위치에 따라 모터와 밸브를 제어하는 시스템을 설계한다. 엔코더를 이용해 패들의 위치를 정확하게 측정하고 그 위치에 따라 모터와 밸브를 제어하는 알고리즘을 구현해야 합니다. 이를 위해 BLDC 모터 제어, 엔코더 값 읽기, 그리고 밸브 제어가 핵심 기능이 됩니다.	필수 IO 프로그램	파일	*메모, 세부 설정 (예)*
BLDC 모터 펌프 드라이버 구동 ModBus	I/O 통합 모듈	초기화, 애러 관리 안됨	추출 압력을 만들기 위한 BLDC 펌프 모터 구동으로 모드버스를 통해 제어한다. 별도의 드라이버 모듈이 있어 모듈 초기화 애러 감지, 가감속 설정을 지원해야 한다.	필수 IO 프로그램	파일	*메모, 세부 설정 (예)*

기능 모듈 명칭	유형	개발 상황	세부정보	중요도 상태	파일	메모, 세부 설정 (예)
라즈베리파이 LCD 구동 CAN	I/O 통합 모듈	기본만 구동	콘트롤 보드가 CAN 통신을 통해 라즈베리파이에 연결된 LCD 디스플레이로 정보를 전달하고, 이를 통해 추출 시간, 추출 온도, 추출 압력, 유량, 유속 등의 정보를 표시합니다. 콘트롤 보드의 작동 상황에 따라 LCD 밝기 조절 및 자동 꺼짐 기능을 수행 할 수 있다. 10분 동안 사용하지 않는 경우 꺼짐, 5분 동안 사용하지 않는 경우 어두워짐.	필수 IO 프로그램	파일	추출 시간: Time: XX.XX s 추출 온도: Temp: XX.XX C 추출 압력: Pressure: XX.XX bar 유량: Flow: XX.XX L/min 유속: Speed: XX.XX m/s
LED 조명 출력 제어	Ouput 출력 모듈	작동	LED 제어 시스템을 통해 추출 조명, 대기 조명을 위한 LED 종류, LED 수, 밝기, 작동 방식을 설정하고 제어하려면, PWM 제어를 통해 LED 밝기를 조절하고, 여러 작동 모드를 설정하는 프로그램 모듈을 작성할 수 있습니다.	필수 IO 프로그램	파일	LED 종류와 개수를 사용자가 설정할 수 있으며, 해당 정보를 기반으로 LED를 제어. LED 밝기는 PWM 핀을 통해 조절되며, 0에서 255까지 설정할 수 있음. LED 모드에 따라 LED를 켜거나, 끄거나, 밝기를 변화시키는 방식으로 제어. 페이드(fade) 모드는 LED 밝기가 점점 커졌다가 줄어드는 효과를 제공. 펄스(pulse) 모드는 일정한 주기로 LED를 깜빡임.
RS485, 모드 버스 모듈	통신 모듈	작동	RS485 통신, 모드버스 통신을 통해 외부 제어 보드와 통신을 진행한다. 추가 옵션 모듈로는 수질 측정기, 전력 측정기 등이 있다.	필수 통신 프로그램	파일	*메모, 세부 설정 (예)*
I2C 모듈	통신 모듈	작동	I2C 통신을 통해 외부 보드와 입출력 통신을 한다.	필수 통신 프로그램	파일	*메모, 세부 설정 (예)*
CAN 통신 모듈 (디스플레이)	통신 모듈	작동	CAN 통신을 통해 센서, 제어 보드, 디스플레이 보드간 통신을 지원한다.	필수 통신 프로그램	파일	*메모, 세부 설정 (예)*
PID 온도 제어	제어 모듈	설정값 미흡	PID (Proportional-Integral-Derivative) 제어는 온도 제어에 자주 사용되는 자동 제어 시스템으로, 목표 온도에 도달하고 오차를 최소화하는 데 효과적입니다. PID 제어는 세 가지 제어 매개변수(비례, 적분, 미분)를 사용하여 온도를 안정적으로 제어합니다. 온도 센서(예: PT100, K타입 열전대 등)로부터 측정된 온도와 목표 온도 간의 오차를 기반으로 히터를 제어하는 방식입니다. 전력 관리 프로그램과 함께 연동하여 에너지 절약 및 온도 안정화를 위한 여러개의 최적화 PID 값을 활용해야한다.	필수 제어 프로그램	파일	Setpoint (float): 목표 온도 값 (예: Setpoint = 100.0;) Input (float): 현재 측정된 온도 값 (예: Input = readTemperature();) Output (float): PID 계산 결과 출력 값 (예: Output = 0.0;) Kp (float): 비례 제어 게인 (예: Kp = 2.0;) Ki (float): 적분 제어 게인 (예: Ki = 0.5;) Kd (float): 미분 제어 게인 (예: Kd = 1.0;) Error (float): 목표 온도와 현재 온도 간의 차이 (예: Error = Setpoint - Input;) Integral (float): 오차의 누적 합 (예: Integral += Error * dt;) Derivative (float): 오차의 변화율 (예: Derivative = (Error - Last_Error) / dt;) Last_Error (float): 이전 오차 값 (예: Last_Error = Error;) dt (float): 시간 변화량 (예: dt = millis() - lastTime;)
콘트롤 보드간 동기화 CAN	제어 모듈	미적용	콘트롤 보드간 제어값 동기화, 표시 값 동기화, 제어 상태 공유를 지원한다. 2~3개 모듈이 한 기기에 설치되어 서로 로그 값을 주고 받아 라즈베리파이로 전송한다.	필수 제어 프로그램	파일	*메모, 세부 설정 (예)*
스팀 보일러 수위 관리 제어	제어 모듈	미흡	수위 센서에서 수집한 정보를 바탕으로 급수 시간, 급수 인터벌을 설정한다. Low 낮은 수위의 접촉이 없는 경우 히터를 멈추거나 경고를 발생시킨다. 급수하여도 수위가 감지가 안되는경우 경고 알람과 히터 차단등의 안전 기능을 수행한다.	필수 제어 프로그램	파일	Low_Level_Sensor_Pin (int): Low 수위 감지 센서 핀 번호 (예: Low_Level_Sensor_Pin = 2;) High_Level_Sensor_Pin (int): High 수위 감지 센서 핀 번호 (예: High_Level_Sensor_Pin = 3;) Solenoid_Valve_Pin (int): 솔레노이드 밸브 핀 번호 (예: Solenoid_Valve_Pin = 4;) Low_Level_State (bool): Low 수위 감지 센서 상태 (LOW일 때 물 부족, 예: Low_Level_State = digitalRead(Low_Level_Sensor_Pin);) High_Level_State (bool): High 수위 감지 센서 상태 (LOW일 때 물 가득, 예: High_Level_State = digitalRead(High_Level_Sensor_Pin);) Valve_State (bool): 솔레노이드 밸브 상태 (ON/OFF, 예: Valve_State = false;)
근접센서 / 엔코더 / 모터 패들 제어	제어 모듈	미흡	패들(핸들) 제어를 통해 추출 명령을 구동하는 방식으로 근접센서를 통해 패들의 원점을 찾고 핸들의 움직임을 엔코더로 받아 들여 모터를 통해 장력과 원점 이동을 제어한다. 기본 동작은 2단 알로그 스위치로 1단 밸브 열기 2단 모터 기동 으로 작동한다.	필수 제어 프로그램	파일	*메모, 세부 설정 (예)*
AC 모터 제어 SSR 제어 (AC SSR 사용)	Ouput 출력 모듈	작동	BLDC 모터 펌프와 AC 모터 펌프의 선택에 따라 AC 모터 제어가 작동한다.	필수 제어 프로그램	파일	*메모, 세부 설정 (예)*
추출 프로파일 관리 제어 (버튼 프로그램)	제어 모듈	기본 작동	이 프로그램은 3개의 버튼을 통해 유량에 따라 커피 추출을 제어합니다. 각 버튼은 원클릭과 더블클릭을 감지하여 다양한 유량 값을 미리 설정하고, 추출 과정에서 이를 기준으로 자동으로 제어됩니다. 버튼을 눌러 유량을 선택하고, 그에 따라 솔레노이드 밸브나 펌프 등의 장치를 제어하여 원하는 유량에 도달하면 추출을 멈춥니다.	필수 제어 프로그램	파일	Button_Pins[] (int): 3개의 버튼이 연결된 핀 번호 (예: Button_Pins[3] = {2, 3, 4};) Single_Click_Flow[] (float): 각 버튼의 원클릭으로 설정된 유량 값 (mL) (예: Single_Click_Flow[3] = {50.0, 100.0, 150.0};) Double_Click_Flow[] (float): 각 버튼의 더블클릭으로 설정된 유량 값 (mL) (예: Double_Click_Flow[3] = {75.0, 125.0, 200.0};) Flow_Rate (float): 실시간 유량 값 (L/min 단위) (예: Flow_Rate = 2.0;) Total_Flow (float): 현재 추출된 총 유량 (mL) (예: Total_Flow = 0.0;) Solenoid_Valve_Pin (int): 솔레노이드 밸브를 제어할 출력 핀 번호 (예: Solenoid_Valve_Pin = 5;) Debounce_Time (unsigned long): 버튼 입력을 디바운스하는 시간 (ms) (예: Debounce_Time = 50;)
BME680 환경 정보 센서 입력	Input 입력 모듈	미적용	BME680은 온도, 습도, 기압, 공기 품질(가스 저항) 등의 환경 정보를 수집할 수 있는 환경 센서입니다. 이 센서는 실내 공기 질 분석 및 환경 모니터링에 주로 사용됩니다. BME680을 통해 온도, 습도, 기압, VOC(휘발성 유기 화합물) 데이터를 측정할 수 있으며, 이를 기반으로 실내 공기질을 평가하는 데 사용할 수 있습니다. 옵션으로 I2C 를 통해 사용이 가능하다.	옵션 프로그램	파일	Temperature (float): 센서로부터 측정된 온도 (°C) (예: Temperature = bme.readTemperature();) Humidity (float): 센서로부터 측정된 습도 (%) (예: Humidity = bme.readHumidity();) Pressure (float): 센서로부터 측정된 기압 (hPa) (예: Pressure = bme.readPressure() / 100.0F;) Gas_Resistance (float): 공기 중 VOC 농도를 나타내는 가스 저항 (옴) (예: Gas_Resistance = bme.readGas();) Air_Quality_Index (float): 가스 저항 값과 환경 데이터를 종합한 공기 질 지수 (AQI) Altitude (float): 기압을 기준으로 계산된 고도 (미터) (예: Altitude = bme.readAltitude(SEALEVELPRESSURE_HPA);)
BH1750FVI 조도 센서 입력	Input 입력 모듈	미적용	BH1750FVI 조도 센서는 옵션으로 조명 밝기(조도)를 측정할 수 있는 디지털 광센서입니다. 이 센서는 I2C 인터페이스를 통해 연결되며, 실시간 조도 값을 간편하게 읽을 수 있습니다. BH1750FVI는 lx(lux) 단위로 밝기를 측정하며, 조도 자동 조정 시스템이나 조명 제어 시스템에 사용됩니다.	옵션 프로그램	파일	BH1750 측정 모드: CONTINUOUS_HIGH_RES_MODE: 연속 측정, 고해상도 모드 (1 lx 분해능). CONTINUOUS_HIGH_RES_MODE_2: 연속 측정, 고해상도 모드 (0.5 lx 분해능). CONTINUOUS_LOW_RES_MODE: 연속 측정, 저해상도 모드 (4 lx 분해능). ONE_TIME_HIGH_RES_MODE: 단일 측정, 고해상도 모드. ONE_TIME_LOW_RES_MODE: 단일 측정, 저해상도 모드. 변수 설명: lux (float): 센서로 측정한 조도 값 (lux 단위) (예: lux = lightMeter.readLightLevel();) I2C_Address (int): BH1750의 I2C 주소 (예: I2C_Address = 0x23;) Measurement_Mode (int): 조도 센서의 측정 모드 설정 (예: Measurement_Mode = BH1750::CONTINUOUS_HIGH_RES_MODE;) Sensor_Status (bool): 센서가 정상적으로 작동 중인지 나타내는 상태 (예: Sensor_Status = lightMeter.begin();)
리셋 부팅, 초기화 버튼 제어	제어 모듈	미적용	콘트롤 보드를 CAN 버튼을 통해 리부팅, 팩토리 초기화 할 수 있는 제어 프로그램으로 3개의 버튼을 동시에 5초 이상 누르고 있으면 리부팅 된다. 10초 이상 누르고 있으면 초기화 된다.	보조 프로그램	파일	*메모, 세부 설정 (예)*
4~20mA 제어 신호 출력	Ouput 출력 모듈	미흡	위부 밸브 제어 보드를 구동하기위해 4~20mA 제어 신호를 추력한다	2순위 프로그램	파일	*메모, 세부 설정 (예)*
로그 관리 모듈	제어 모듈	미적용	사용자의 작동 로그를 기록하고 관리하는 것으로 추출량, 추출시간, 총 횟수, 설정 값등의 로그를 관리하여 분석할 수 있게 IoT에 연동되어야 한다.	필수 제어 프로그램	파일	*메모, 세부 설정 (예)*
절전 및 PID 관리 모듈	제어 모듈	미적용	로그를 분석하여 절전 및 슬립 모드를 자동으로 제어하고 PID 값을 적용하는 제어 모듈	보조 프로그램	파일	*메모, 세부 설정 (예)*
OTA 펌웨어 업데이트	제어 모듈	미적용	라즈베리파이를 통해서 원격으로 펌웨어를 업데이트 하는 기능으로 CAN을 통해 업데이트가 진행된다.	보조 프로그램	파일	*메모, 세부 설정 (예)*
스마트 플로우 메타 ModBus	I/O 통합 모듈	미적용	ModBus 통신을 통해 수질, 유량, 유속을 판단하는 플로우메타와 연결한다.	2순위 프로그램	파일	*메모, 세부 설정 (예)*