CURSO COMPLETO: FUNDAMENTOS DE ELECTRÓNICA

Video 03 — Parte 2A: Física de Semiconductores, Diodo de Unión, Zener, Regulador de Tensión, Diodos Optoelectrónicos y Análisis de Circuitos — Dopaje N y P, Unión PN, Ecuación de Shockley, Diodo Zener, Regulador, LED, Fotodiodo, Rectificadores y Recortadores

Parte del Curso: Parte 2A — Electrónica de Semiconductores: Física, Diodo de Unión, Zener, Regulador, Optoelectrónica y Circuitos con Diodos
Arquitectura / Controlador Estudiado: Semiconductores Tipo N y P, Ecuación de Shockley, Regulador Zener, LED/Fotodiodo, Rectificador de Onda Completa y Recortadores

```
n·p = n_i² | I_D = I_S·(e^(V_D / (n·V_T)) - 1) | R_S = (V_in(min) - V_Z) / (I_ZK + I_L(max)) | V_r(pp) ≈ I_L / (f_r·C)
```

1. Síntesis de Conceptos Clave y Reglas de Ingeniería

- Física de semiconductores: silicio intrínseco (E_g = 1.12 eV), dopaje tipo N (donadores pentavalentes) y tipo P (aceptores trivalentes), corrientes de arrastre y difusión.
- El diodo de unión PN: zona de empobrecimiento, barrera de potencial (~0.7 V en Si), polarización directa e inversa, ecuación exponencial de Shockley y modelos lineales por tramos.
- Diodo Zener en ruptura inversa controlada, diseño de reguladores de tensión frente a variaciones de línea y carga, y diodos optoelectrónicos (LED con resistencia limitadora y fotodiodo en polarización inversa).
- Análisis de circuitos con diodos: rectificadores de media onda y puente completo de Graetz con filtro capacitivo, recortadores (clippers), fijadores de nivel (clampers) y diodo de rueda libre (flyback).

2. Parámetros Nominales de Configuración del Controlador (Video 03)

Parámetro del Controlador / Interfase	Símbolo	Valor Nominal	Función en el Lazo de Control / Automatización
Ganancia Proporcional	`Kp (PB = 100/Kp)`	4	Esfuerzo inmediato proporcional al error actual e(t)
Tiempo Integral	`Ti [s]`	0.35 s	Elimina el error estacionario (Offset) acumulando la integral del error
Tiempo Derivativo y Filtro Divisor	`Td [s] / N`	Td = 0.05 s, N = 10	Anticipa la tendencia sobre -PV y atenúa el ruido de alta frecuencia
Periodo de Muestreo en Tiempo Real	`Ts [ms]`	1 ms	Intervalo determinista de conversión ADC, cálculo discreto y actualización PWM
Límites de Saturación Anti-Windup	`[u_min, u_max]`	[0, 12]	Cota física del actuador para integración condicional (Clamping)
Rango Escalado de Ingeniería (SCALE_X)	`[PV_min, PV_max]`	[0, 15]	Conversión lineal desde cuentas crudas del ADC (0..1023 o 0..27648)

3. Desarrollo Académico Diapositiva por Diapositiva

Diapositiva 01: Introducción a la Electrónica de Semiconductores y Estructura Cristalina

Conductores, Aislantes y Semiconductores (Silicio y Germanio), Enlaces Covalentes, Banda de Valencia y Banda de Conducción

- **Turno 1 (Masculina):** La electrónica moderna nace de la capacidad de controlar con precisión la conductividad eléctrica de materiales semiconductores como el silicio y el germanio, situados entre los metales conductores y los dieléctricos aislantes.
- **Turno 2 (Femenina):** En un cristal de silicio puro o intrínseco, cada átomo posee cuatro electrones de valencia que forman cuatro enlaces covalentes estables con sus átomos vecinos en la red cristalina.
- **Turno 3 (Masculina):** A temperatura ambiente, la agitación térmica permite que algunos electrones superen la brecha de energía prohibida de uno punto doce electronvoltios, saltando desde la banda de valencia hacia la banda de conducción.
- **Turno 4 (Femenina):** Cada electrón que salta deja tras de sí una vacante con carga efectiva positiva llamada hueco, de modo que en un semiconductor conducen simultáneamente dos tipos de portadores: electrones libres y huecos.

Diapositiva 02: Física de Semiconductores: Dopaje Tipo N, Tipo P y Transporte de Carga

Impurezas Donadoras Pentavalentes (Tipo N), Aceptoras Trivalentes (Tipo P), Ley de Acción de Masas y Corrientes de Arrastre y Difusión

- **Turno 1 (Masculina):** Para incrementar y controlar la conductividad del silicio introducimos impurezas controladas en el proceso denominado dopaje, creando semiconductores extrínsecos.
- **Turno 2 (Femenina):** Si añadimos átomos pentavalentes como fósforo o arsénico, el quinto electrón queda libre para conducir sin crear huecos, formando un semiconductor tipo ene donde los electrones son portadores mayoritarios y los huecos son minoritarios.
- **Turno 3 (Masculina):** Si dopamos con átomos trivalentes como boro o galio, falta un electrón para completar el enlace covalente y se genera una alta concentración de huecos libres, dando lugar a un semiconductor tipo pe.
- **Turno 4 (Femenina):** En estos materiales la corriente eléctrica fluye por dos mecanismos físicos fundamentales: la corriente de arrastre o deriva impulsada por un campo eléctrico externo, y la corriente de difusión provocada por gradientes de concentración de portadores.

Diapositiva 03: El Diodo de Unión PN: Región de Empobrecimiento y Polarización

Barrera de Potencial (~0.7 V en Silicio), Polarización Directa, Polarización Inversa y Capacitancia de Unión

- **Turno 1 (Masculina):** Al unir metalúrgicamente una región tipo pe llamada ánodo con una región tipo ene llamada cátodo, los electrones y huecos cercanos a la frontera se difunden y recombinan, dejando iones fijos descubiertos que forman la región de empobrecimiento.
- **Turno 2 (Femenina):** Esos iones crean un campo eléctrico interno que establece una barrera de potencial de aproximadamente cero punto siete voltios en silicio y cero punto tres voltios en germanio o diodos Schottky.
- **Turno 3 (Masculina):** Cuando aplicamos polarización inversa, conectando el positivo de la fuente al cátodo y el negativo al ánodo, la zona de empobrecimiento se ensancha y el diodo bloquea la conducción, permitiendo solo una diminuta corriente inversa de fuga.
- **Turno 4 (Femenina):** Cuando aplicamos polarización directa, con el ánodo positivo respecto al cátodo superando el umbral de cero punto siete voltios, la barrera se estrecha drásticamente y fluye una corriente exponencial de portadores mayoritarios.

Diapositiva 04: Ecuación de Shockley y Modelos de Aproximación del Diodo

Curva Característica Exponencial I_D = I_S·(e^(V_D / n·V_T) - 1), Tensión Térmica V_T ≈ 25.8 mV y Modelo de Caída Constante

- **Turno 1 (Masculina):** La relación matemática entre la corriente del diodo i sub de y su voltaje ve sub de está gobernada por la ecuación de Shockley: la corriente de saturación inversa multiplicada por la exponencial de ve sub de sobre ene veces el voltaje térmico, menos uno.
- **Turno 2 (Femenina):** El voltaje térmico ve sub te vale unos veintiséis milivoltios a temperatura ambiente, lo que implica que en conducción directa cada incremento de unos sesenta milivoltios en la tensión multiplica por diez la corriente del diodo.
- **Turno 3 (Masculina):** Para resolver circuitos de forma ágil sin ecuaciones trascendentes empleamos tres niveles de aproximación: el modelo ideal como interruptor perfecto, el modelo de caída constante de cero punto siete voltios y el modelo lineal por tramos que añade una resistencia dinámica serie.
- **Turno 4 (Femenina):** El modelo de caída constante de cero punto siete voltios resulta sumamente preciso siempre que la tensión de la fuente del circuito sea mucho mayor que el umbral de conducción del diodo.

Diapositiva 05: El Diodo Zener: Operación en la Región de Ruptura Controlada

Efecto Zener (< 5V) y Efecto Avalancha (> 5V), Tensión Zener V_Z, Corriente de Rodilla I_ZK e Impedancia Dinámica Z_Z

- **Turno 1 (Masculina):** A diferencia de un diodo rectificador convencional que se destruye si excedemos su tensión inversa de ruptura, el diodo Zener se diseña con un dopaje preciso para trabajar de forma continua y segura en la región de ruptura inversa.
- **Turno 2 (Femenina):** En tensiones inferiores a cinco voltios predomina el efecto Zener por efecto túnel cuántico, con coeficiente térmico negativo, mientras que por encima de cinco voltios predomina la multiplicación por avalancha con coeficiente térmico positivo.
- **Turno 3 (Masculina):** Curiosamente, los diodos Zener cercanos a cinco punto uno o cinco punto seis voltios combinan ambos fenómenos físicos y logran una estabilidad térmica excepcional frente a cambios de temperatura.
- **Turno 4 (Femenina):** Mientras la corriente inversa se mantenga entre la corriente mínima de rodilla i zeta ka y la corriente máxima admisible por potencia i zeta eme, el cátodo se mantiene a una tensión constante ve sub zeta respecto al ánodo.

Diapositiva 06: Regulador de Tensión con Diodo Zener: Análisis y Diseño

Cálculo de Resistencia Serie R_S, Regulación de Línea ante Variación de V_in y Regulación de Carga ante Variación de I_L

- **Turno 1 (Masculina):** La aplicación clásica del diodo Zener es el regulador de tensión en paralelo o shunt: conectamos una resistencia limitadora erre sub ese desde la fuente no regulada hacia el cátodo del Zener, y la carga en paralelo con el diodo.
- **Turno 2 (Femenina):** Por la Ley de Corrientes de Kirchhoff, la corriente que baja por la resistencia serie se reparte entre la corriente del diodo Zener y la corriente demandada por la carga.
- **Turno 3 (Masculina):** La condición crítica de salida de regulación ocurre cuando la tensión de entrada cae a su valor mínimo simultáneamente con la corriente de carga máxima; en ese punto extremo el Zener debe recibir al menos su corriente mínima de rodilla.
- **Turno 4 (Femenina):** En el extremo opuesto, cuando la tensión de entrada es máxima y la carga se desconecta por completo, toda la corriente atraviesa el Zener, por lo que debemos verificar que el producto de ve sub zeta por la corriente máxima no supere la potencia nominal del diodo.

Diapositiva 07: Diodos Optoelectrónicos: LED, Fotodiodo y Celdas Fotovoltaicas

Recombinación Radiativa en Compuestos III-V (LED), Caída Directa V_F según Longitud de Onda y Fotodiodo en Polarización Inversa

- **Turno 1 (Masculina):** Los diodos optoelectrónicos convierten energía eléctrica en radiación luminosa o viceversa mediante semiconductores de brecha de banda directa como el arseniuro de galio o el nitruro de galio.
- **Turno 2 (Femenina):** En un diodo emisor de luz o LED polarizado en directa, cada electrón que se recombina con un hueco libera un fotón cuya longitud de onda fija el color y también su caída de tensión directa: desde uno punto ocho voltios en rojo hasta más de tres voltios en azul y blanco.
- **Turno 3 (Masculina):** Dado que su curva es exponencial, jamás debemos conectar un LED directamente a una fuente de tensión sin una resistencia limitadora en serie o una fuente de corriente constante.
- **Turno 4 (Femenina):** El proceso inverso ocurre en el fotodiodo: los fotones incidentes sobre la región de empobrecimiento generan pares electrón-hueco proporcionales a la iluminancia, operando en polarización inversa para respuesta ultrarrápida o sin polarización en modo fotovoltaico.

Diapositiva 08: Análisis de Circuitos con Diodos I: Rectificadores y Filtrado Capacitivo

Rectificador de Media Onda, Puente de Onda Completa (4 Diodos), Tensión Inversa de Pico (PIV) y Rizado $V_r = I_L / (f_r \cdot C)$

- **Turno 1 (Masculina):** Para convertir tensión alterna en continua empleamos circuitos rectificadores: el rectificador de media onda utiliza un solo diodo y aprovecha únicamente los semiciclos positivos, presentando baja eficiencia.
- **Turno 2 (Femenina):** En cambio, el puente rectificador de onda completa de cuatro diodos conduce mediante dos diodos en serie durante el semiciclo positivo y mediante el par opuesto en el negativo, invirtiendo las crestas negativas hacia la carga.
- **Turno 3 (Masculina):** En un puente completo la tensión pico en la carga es igual al voltaje pico del secundario menos dos caídas de diodo, es decir, aproximadamente uno punto cuatro voltios menos, duplicando la frecuencia del rizado a cien o ciento veinte hercios.
- **Turno 4 (Femenina):** Al conectar un capacitor filtro en paralelo con la salida, la tensión pico a pico de rizado se reduce a la corriente de carga dividida entre la frecuencia de rizado multiplicada por la capacitancia.

Diapositiva 09: Análisis de Circuitos con Diodos II: Recortadores, Fijadores y Protección

Circuitos Recortadores Serie/Paralelo (Clippers), Fijadores de Nivel DC (Clampers) y Diodo de Rueda Libre (Flyback)

- **Turno 1 (Masculina):** En el acondicionamiento de señales, los circuitos recortadores o clippers utilizan diodos para limitar la excursión de una señal por encima o por debajo de un umbral de referencia, protegiendo las entradas de conversores analógico-digitales.
- **Turno 2 (Femenina):** Un par de diodos de señal conectados desde el pin de entrada hacia tierra y hacia la alimentación de tres punto tres voltios recorta instantáneamente cualquier sobretensión transitoria sin distorsionar la señal dentro de su rango válido.
- **Turno 3 (Masculina):** Por otro lado, los circuitos fijadores o clampers combinan un capacitor en serie con un diodo en paralelo para desplazar el nivel continuo de una forma de onda periódica sin alterar su amplitud pico a pico.
- **Turno 4 (Femenina):** Finalmente, en toda bobina de relé, válvula o motor debemos colocar en antiparalelo un diodo de rueda libre o flyback, el cual ofrece un camino de recirculación seguro para la corriente inductiva al apagarse el interruptor.

Diapositiva 10: Síntesis de la Parte 2A y Transición a los Transistores BJT y MOSFET

De la Unión PN de Dos Terminales al Control Activo de Corriente y Potencia con Dispositivos de Tres Terminales

- **Turno 1 (Masculina):** En esta Parte dos A hemos recorrido desde la física cuántica del silicio dopado tipo pe y tipo ene hasta el comportamiento real del diodo de unión en rectificación, regulación Zener, optoelectrónica y recorte de señales.
- **Turno 2 (Femenina):** Vimos que el diodo actúa como una válvula unidireccional controlada por la polaridad de su propia tensión entre ánodo y cátodo.
- **Turno 3 (Masculina):** Sin embargo, para amplificar señales débiles o conmutar cargas mediante una orden lógica externa necesitamos agregar una tercera terminal de control al cristal semiconductor.
- **Turno 4 (Femenina):** En la Parte dos B estudiaremos en detalle las dos grandes familias de transistores que gobiernan la electrónica: los transistores bipolares de unión BJT y los transistores de efecto de campo JFET y MOSFET.

4. Firmware C++/MCU: PID Discreto Tustin, FSM y CRC-16 (controlador_tiempo_real_video_03.ino)

```
/*
 * ============================================================================
 * CURSO COMPLETO: FUNDAMENTOS DE ELECTRÓNICA
 * VIDEO 03: Video 03 — Parte 2A: Física de Semiconductores, Diodo de Unión, Zener, Regulador de Tensión, Diodos Optoelectrónicos y Análisis de Circuitos
 * Archivo: controlador_tiempo_real_video_03.ino
 * ----------------------------------------------------------------------------
 * Descripción Técnica:
 *   Firmware determinista en C/C++ Embebido para Microcontrolador (compatible
 *   con arquitectura AVR / ARM Cortex-M STM32 / ESP32 / RP2040) que implementa:
 *     1. Acceso a nivel de registros con punteros 'volatile' y máscaras de bits.
 *     2. Controlador PID Discreto por Transformación Bilineal de Tustin con
 *        derivada sobre la variable medida (-PV), filtro derivativo N y
 *        esquema Anti-Windup por retro-cálculo y Clamping.
 *     3. Máquina de Estados Finitos (FSM de Moore) no bloqueante para
 *        secuenciación segura e interbloqueo de emergencia.
 *     4. Trama de telemetría con verificación CRC-16-CCITT hacia SBC/PLC.
 *
 * Enfoque del Módulo:
 *   Semiconductores Tipo N y P, Ecuación de Shockley, Regulador Zener, LED/Fotodiodo, Rectificador de Onda Completa y Recortadores
 * ============================================================================
 */

#include <Arduino.h>
#include <stdint.h>

// --- Definición de Pines de Interfase (Entradas Optoaisladas y Salidas PWM/MOSFET) ---
const uint8_t PIN_ADC_PV_SENSOR   = A0; // Entrada analógica PV (ej. Shunt 165 ohm de lazo 4-20 mA)
const uint8_t PIN_ADC_SP_CONSIGNA = A1; // Consigna local o referencia analógica (0..5V / 0..3.3V)
const uint8_t PIN_DI_MARCHA_NO    = 2;  // Entrada optoacoplada PC817: Pulsador de Marcha (NO)
const uint8_t PIN_DI_PARO_NC      = 3;  // Entrada optoacoplada PC817: Paro / Emergencia (Cableado NC)
const uint8_t PIN_PWM_ACTUADOR    = 9;  // Salida PWM hacia Driver MOSFET / Puente H
const uint8_t PIN_DO_SSR_ALARMA   = 8;  // Salida hacia Relé de Estado Sólido (SSR) / Piloto

// --- Parámetros Nominales del Controlador PID Discreto (Video 03) ---
const float KP_GAIN      = 4.000f;
const float TI_SEC       = 0.350f;
const float TD_SEC       = 0.050f;
const float N_FILT       = 10.0f;
const float TS_SEC       = 0.0010f; // Periodo de muestreo Ts = 1 ms
const float U_MIN_LIM    = 0.0f;
const float U_MAX_LIM    = 12.0f;
const float PV_MIN_ENG   = 0.0f;
const float PV_MAX_ENG   = 15.0f;

// --- Variables Volátiles Compartidas con Interrupción (ISR) ---
volatile bool     g_flagTickControl = false;
volatile uint32_t g_contadorPulsosEncoder = 0;

// --- Estructura de Estado del Controlador PID Discreto (Tustin + Anti-Windup) ---
struct ControladorPIDDiscreto {
  float kp;
  float ki;
  float kd;
  float n_divisor;
  float ts;
  float u_min;
  float u_max;
  float integral_acum;
  float termino_d_prev;
  float pv_prev;
  float error_prev;
  float u_salida;
};

ControladorPIDDiscreto g_pid;
```

```
// --- Máquina de Estados Finitos (FSM No Bloqueante de Moore) ---
enum EstadoControlador : uint8_t {
  ESTADO_REPOSO_SEGURO   = 0,
  ESTADO_PRECARGA_SUAVE  = 1,
  ESTADO_LAZO_PID_ACTIVO = 2,
  ESTADO_FALLA_PARO_NC   = 3
};

EstadoControlador g_estadoActual = ESTADO_REPOSO_SEGURO;
uint32_t g_prevMicrosLazo  = 0;
uint32_t g_timestampEstado = 0;
uint32_t g_pasoMuestreo    = 0;

// --- Rutina de Servicio de Interrupción Externa (Encoder / Sensor Rápido) ---
void isrPulsoEncoder() {
  g_contadorPulsosEncoder++;
}

// --- Inicialización de Coeficientes del PID Discreto ---
void inicializarPID(ControladorPIDDiscreto &c) {
  c.kp             = KP_GAIN;
  c.ki             = (TI_SEC > 1e-5f) ? (KP_GAIN / TI_SEC) : 0.0f;
  c.kd             = KP_GAIN * TD_SEC;
  c.n_divisor      = N_FILT;
  c.ts             = TS_SEC;
  c.u_min          = U_MIN_LIM;
  c.u_max          = U_MAX_LIM;
  c.integral_acum  = 0.0f;
  c.termino_d_prev = 0.0f;
  c.pv_prev        = 0.0f;
  c.error_prev     = 0.0f;
  c.u_salida       = 0.0f;
}

// --- Evaluación en Tiempo Real del PID por Regla Trapezoidal de Tustin ---
float evaluarPasoPID_Tustin(ControladorPIDDiscreto &c, float sp, float pv) {
  float error = sp - pv;

  // 1. Término Proporcional
  float p_term = c.kp * error;

  // 2. Término Derivativo sobre -PV filtrado por Tustin (Evita Derivative Kick)
  float tau_d  = (c.kp > 1e-6f) ? ((c.kd / c.kp) / c.n_divisor) : 0.0f;
  float denom  = 2.0f * tau_d + c.ts;
  float alpha1 = (2.0f * tau_d - c.ts) / denom;
  float beta0  = (2.0f * c.kd) / denom;
  float d_term = alpha1 * c.termino_d_prev - beta0 * (pv - c.pv_prev);

  // 3. Candidato de Integración Trapezoidal (Tustin)
  float delta_i = 0.5f * c.ki * c.ts * (error + c.error_prev);
  float i_cand  = c.integral_acum + delta_i;

  // 4. Suma Pre-Saturación y Esquema Anti-Windup Condicional (Clamping)
  float u_unsat = p_term + i_cand + d_term;
  float u_sat   = u_unsat;
  if (u_unsat > c.u_max) {
    u_sat = c.u_max;
    if (error < 0.0f) c.integral_acum = i_cand; // Solo integra si ayuda a desaturar
  } else if (u_unsat < c.u_min) {
    u_sat = c.u_min;
    if (error > 0.0f) c.integral_acum = i_cand;
  } else {
    c.integral_acum = i_cand;
  }

  // Actualizar memoria de estados discretos z^-1
  c.termino_d_prev = d_term;
```

```
  c.pv_prev        = pv;
  c.error_prev     = error;
  c.u_salida       = u_sat;
  return u_sat;
}

// --- Escalado Lineal Equivalente a NORM_X + SCALE_X ---
inline float escalarCuentasADC(int cuentasCrudas, int rawMin, int rawMax, float engMin, float engMax) {
  float u_norm = (float)(cuentasCrudas - rawMin) / (float)(rawMax - rawMin);
  if (u_norm < 0.0f) u_norm = 0.0f;
  if (u_norm > 1.0f) u_norm = 1.0f;
  return engMin + u_norm * (engMax - engMin);
}

// --- Cálculo de Verificación de Redundancia Cíclica CRC-16-CCITT (Polinomio 0x1021) ---
uint16_t calcularCRC16(const uint8_t *datos, uint8_t longitud) {
  uint16_t crc = 0xFFFFU;
  for (uint8_t i = 0; i < longitud; i++) {
    crc ^= ((uint16_t)datos[i] << 8);
    for (uint8_t b = 0; b < 8; b++) {
      if (crc & 0x8000U) {
        crc = (crc << 1) ^ 0x1021U;
      } else {
        crc <<= 1;
      }
    }
  }
  return crc;
}

void setup() {
  Serial.begin(115200);
  pinMode(PIN_DI_MARCHA_NO, INPUT_PULLUP);
  pinMode(PIN_DI_PARO_NC, INPUT_PULLUP);
  pinMode(PIN_PWM_ACTUADOR, OUTPUT);
  pinMode(PIN_DO_SSR_ALARMA, OUTPUT);

  attachInterrupt(digitalPinToInterrupt(PIN_DI_MARCHA_NO), isrPulsoEncoder, RISING);
  inicializarPID(g_pid);
  g_prevMicrosLazo  = micros();
  g_timestampEstado = millis();

  Serial.println(F("# ================================================================"));
  Serial.println(F("# CURSO CONTROLADORES EN MECATRONICA - VIDEO 03"));
  Serial.println(F("# Columnas: t_s,estado_fsm,SP_eng,PV_eng,u_PID,pulsos_enc,crc16"));
  Serial.println(F("# ================================================================"));
}

void loop() {
  uint32_t ahoraMicros = micros();
  uint32_t periodoMicros = (uint32_t)(TS_SEC * 1000000.0f);

  if (ahoraMicros - g_prevMicrosLazo >= periodoMicros) {
    g_prevMicrosLazo += periodoMicros;
    float t_sec = g_pasoMuestreo * TS_SEC;

    // 1. Lectura de Entradas Digitales (Seguridad de Paro NC y Marcha NO)
    bool paroCableadoOK = (digitalRead(PIN_DI_PARO_NC) == HIGH);
    bool ordenMarcha    = (digitalRead(PIN_DI_MARCHA_NO) == LOW);

    // 2. Lectura y Escalado de Entradas Analógicas (NORM_X -> SCALE_X)
    int rawPV = analogRead(PIN_ADC_PV_SENSOR);
    int rawSP = analogRead(PIN_ADC_SP_CONSIGNA);
    float pv_eng = escalarCuentasADC(rawPV, 0, 1023, PV_MIN_ENG, PV_MAX_ENG);
    float sp_eng = escalarCuentasADC(rawSP, 0, 1023, PV_MIN_ENG, PV_MAX_ENG);

    // 3. Evaluación de la Máquina de Estados Finitos (FSM de Moore No Bloqueante)
    if (!paroCableadoOK) {
```

```
    g_estadoActual = ESTADO_FALLA_PARO_NC;
  }

  float mandoActuador = 0.0f;
  switch (g_estadoActual) {
    case ESTADO_REPOSO_SEGURO:
      mandoActuador = 0.0f;
      digitalWrite(PIN_DO_SSR_ALARMA, LOW);
      if (ordenMarcha && paroCableadoOK) {
        g_timestampEstado = millis();
        g_estadoActual    = ESTADO_PRECARGA_SUAVE;
      }
      break;

    case ESTADO_PRECARGA_SUAVE:
      mandoActuador = 0.25f * U_MAX_LIM;
      if (millis() - g_timestampEstado >= 200UL) {
        // Transferencia sin saltos (Bumpless) al activar el lazo PID
        g_pid.integral_acum = mandoActuador;
        g_pid.pv_prev       = pv_eng;
        g_estadoActual      = ESTADO_LAZO_PID_ACTIVO;
      }
      break;

    case ESTADO_LAZO_PID_ACTIVO:
      mandoActuador = evaluarPasoPID_Tustin(g_pid, sp_eng, pv_eng);
      digitalWrite(PIN_DO_SSR_ALARMA, LOW);
      break;

    case ESTADO_FALLA_PARO_NC:
    default:
      mandoActuador = 0.0f;
      g_pid.integral_acum = 0.0f;
      digitalWrite(PIN_DO_SSR_ALARMA, HIGH);
      if (paroCableadoOK && ordenMarcha) {
        g_estadoActual = ESTADO_REPOSO_SEGURO;
      }
      break;
  }

  // 4. Escritura de Salida PWM hacia Etapa de Potencia (MOSFET / Puente H)
  float u_norm_pwm = (mandoActuador - U_MIN_LIM) / (U_MAX_LIM - U_MIN_LIM);
  int duty8bit     = constrain((int)(u_norm_pwm * 255.0f), 0, 255);
  analogWrite(PIN_PWM_ACTUADOR, duty8bit);

  // 5. Lectura Atómica de Contador de Interrupción y Telemetría con CRC-16
  if (g_pasoMuestreo % 10 == 0) {
    noInterrupts();
    uint32_t pulsosCopia = g_contadorPulsosEncoder;
    interrupts();

    uint8_t tramaBytes[6];
    tramaBytes[0] = (uint8_t)g_estadoActual;
    tramaBytes[1] = (uint8_t)duty8bit;
    tramaBytes[2] = (uint8_t)((rawPV >> 8) & 0xFF);
    tramaBytes[3] = (uint8_t)(rawPV & 0xFF);
    tramaBytes[4] = (uint8_t)((pulsosCopia >> 8) & 0xFF);
    tramaBytes[5] = (uint8_t)(pulsosCopia & 0xFF);
    uint16_t crc  = calcularCRC16(tramaBytes, 6);

    Serial.print(t_sec, 3);              Serial.print(',');
    Serial.print((int)g_estadoActual);   Serial.print(',');
    Serial.print(sp_eng, 2);             Serial.print(',');
    Serial.print(pv_eng, 2);             Serial.print(',');
    Serial.print(mandoActuador, 2);      Serial.print(',');
    Serial.print(pulsosCopia);           Serial.print(",0x");
    Serial.println(crc, HEX);
  }
```

```
    g_pasoMuestreo++;
  }
}
```

5. Script Python SBC: Sintonización, Índices IAE/ITAE y Tramas (supervision_sbc_y_sintonizacion_video_03.py)

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
================================================================================
CURSO COMPLETO: FUNDAMENTOS DE ELECTRÓNICA
VIDEO 03: Video 03 – Parte 2A: Física de Semiconductores, Diodo de Unión, Zener, Regulador de Tensión, Diodos
Optoelectrónicos y Análisis de Circuitos
Archivo: supervision_sbc_y_sintonizacion_video_03.py
--------------------------------------------------------------------------------
Descripción Técnica:
  Script integral en Python para Computadora de Placa Única (Raspberry Pi /
  SBC) y estación de ingeniería que implementa:
    1. Simulación en lazo cerrado de un controlador PID discreto (Tustin +
       Filtro Derivativo N + Anti-Windup por Clamping y Back-Calculation).
    2. Cálculo cuantitativo de métricas transitorias (tr, tp, Mp, ts) y de
       los cuatro índices integrales de desempeño (IAE, ISE, ITAE, ITSE).
    3. Reglas automáticas de sintonización de Ziegler-Nichols (1.º y 2.º
       método), Cohen-Coon y sintonización robusta IMC / Lambda.
    4. Construcción y validación de tramas binarias Maestro-Coprocesador
       (SBC <-> MCU/PLC) protegidas con suma de verificación CRC-16-CCITT.
================================================================================
"""

import math
import struct
from typing import Dict, List, Tuple

# Parámetros nominales del Video 03
KP_NOMINAL = 4
TI_NOMINAL = 0.35
TD_NOMINAL = 0.05
N_FILTRO   = 10
TS_SEG     = 0.0010
U_MIN      = 0
U_MAX      = 12


def sintonizar_pid_ziegler_nichols_y_lambda(
    K_planta: float, L_retardo: float, T_constante: float, lambda_deseado: float
) -> Dict[str, Dict[str, float]]:
    """
    Calcula las ganancias PID (Kp, Ti, Td) mediante:
      - Ziegler-Nichols 1.er Método (Curva de Reacción FOPDT)
      - Método de Cohen-Coon (para retardo significativo L/T)
      - Método IMC / Regla Lambda (respuesta sin sobreimpulso)
    """
    r = L_retardo / max(T_constante, 1e-6)

    # 1. Ziegler-Nichols Lazo Abierto (PID Estándar ISA)
    kp_zn = (1.2 * T_constante) / (K_planta * max(L_retardo, 1e-6))
    ti_zn = 2.0 * L_retardo
    td_zn = 0.5 * L_retardo

    # 2. Cohen-Coon (PID)
    kp_cc = (1.0 / (K_planta * r)) * (4.0 / 3.0 + r / 4.0)
    ti_cc = L_retardo * (32.0 + 6.0 * r) / (13.0 + 8.0 * r)
    td_cc = L_retardo * 4.0 / (11.0 + 2.0 * r)

    # 3. IMC / Lambda (PI robusto sin sobreimpulso + término D para retardo Padé)
    kp_imc = T_constante / (K_planta * (lambda_deseado + L_retardo))
    ti_imc = T_constante
    td_imc = (T_constante * L_retardo) / (2.0 * T_constante + L_retardo)

    return {
        "Ziegler_Nichols_FOPDT": {
            "Kp": round(kp_zn, 4), "Ti_s": round(ti_zn, 4), "Td_s": round(td_zn, 4)
        },
```

```
        "Cohen_Coon": {
            "Kp": round(kp_cc, 4), "Ti_s": round(ti_cc, 4), "Td_s": round(td_cc, 4)
        },
        "IMC_Lambda_Robusto": {
            "Kp": round(kp_imc, 4), "Ti_s": round(ti_imc, 4), "Td_s": round(td_imc, 4)
        },
    }


def simular_lazo_pid_discreto_tustin(
    kp: float, ti: float, td: float, n_filt: float, ts: float, pasos: int = 800
) -> Dict[str, float]:
    """
    Simula una planta mecatrónica de 2.º orden controlada por un PID discreto
    discretizado por Tustin con Anti-Windup y calcula tr, Mp, ts, IAE, ISE, ITAE.
    """
    ki = kp / ti if ti > 1e-6 else 0.0
    kd = kp * td
    tau_d = (td / n_filt) if n_filt > 0 else 0.0
    denom_d = 2.0 * tau_d + ts
    alpha_d = (2.0 * tau_d - ts) / denom_d
    beta_d  = (2.0 * kd) / denom_d

    # Estado de la planta (wn = 8 rad/s, zeta = 0.45, K = 1.0)
    wn = 8.0
    zeta = 0.45
    y_pos = 0.0
    y_vel = 0.0
    sp = 1.0

    integral = 0.0
    d_prev = 0.0
    y_prev = 0.0
    e_prev = 0.0

    iae = 0.0
    ise = 0.0
    itae = 0.0
    itse = 0.0
    y_max = 0.0
    t_pico = 0.0
    t_10 = None
    t_90 = None
    t_estable = 0.0

    for k in range(pasos):
        t = k * ts
        error = sp - y_pos

        # Acumulación de índices integrales de desempeño
        abs_e = abs(error)
        iae  += abs_e * ts
        ise  += (error ** 2) * ts
        itae += (t * abs_e) * ts
        itse += (t * (error ** 2)) * ts

        # Algoritmo PID Discreto con filtro derivativo sobre -PV
        p_term = kp * error
        d_term = alpha_d * d_prev - beta_d * (y_pos - y_prev)
        i_cand = integral + 0.5 * ki * ts * (error + e_prev)

        u_unsat = p_term + i_cand + d_term
        u_cmd = max(U_MIN, min(U_MAX, u_unsat))

        # Anti-Windup condicional (Clamping)
        if u_unsat == u_cmd or (u_unsat > U_MAX and error < 0) or (u_unsat < U_MIN and error > 0):
            integral = i_cand
```

```
        d_prev = d_term
        y_prev = y_pos
        e_prev = error

        # Integración de la planta física (Euler semi-implícito)
        acel = (wn ** 2) * u_cmd - 2.0 * zeta * wn * y_vel - (wn ** 2) * y_pos
        y_vel += acel * ts
        y_pos += y_vel * ts

        if y_pos >= 0.10 * sp and t_10 is None:
            t_10 = t
        if y_pos >= 0.90 * sp and t_90 is None:
            t_90 = t
        if y_pos > y_max:
            y_max = y_pos
            t_pico = t
        if abs(sp - y_pos) > 0.02 * sp:
            t_estable = t

    tr = (t_90 - t_10) if (t_10 is not None and t_90 is not None) else 0.0
    mp_pct = max(0.0, (y_max - sp) / sp * 100.0)

    return {
        "tiempo_subida_tr_s": round(tr, 4),
        "tiempo_pico_tp_s": round(t_pico, 4),
        "sobreimpulso_Mp_pct": round(mp_pct, 2),
        "tiempo_establecimiento_ts_2pct_s": round(t_estable, 4),
        "IAE": round(iae, 4),
        "ISE": round(ise, 4),
        "ITAE": round(itae, 4),
        "ITSE": round(itse, 4),
    }


def crc16_ccitt(data: bytes) -> int:
    """Calcula CRC-16-CCITT (0x1021, init 0xFFFF) para enlace serie SBC <-> MCU."""
    crc = 0xFFFF
    for byte in data:
        crc ^= (byte << 8)
        for _ in range(8):
            if crc & 0x8000:
                crc = ((crc << 1) ^ 0x1021) & 0xFFFF
            else:
                crc = (crc << 1) & 0xFFFF
    return crc


def empaquetar_trama_consigna_sbc(comando: int, setpoint_float: float) -> bytes:
    """Construye trama binaria: [0xAA, 0x55, CMD, LEN=4, float32_LE, CRC16_BE]."""
    cabecera_y_payload = struct.pack("<BBBf", 0xAA, 0x55, comando, 4, setpoint_float)
    crc = crc16_ccitt(cabecera_y_payload)
    return cabecera_y_payload + struct.pack(">H", crc)


if __name__ == "__main__":
    print("=" * 78)
    print(" CURSO: FUNDAMENTOS DE ELECTRÓNICA — VIDEO 03")
    print(" Enfoque:", "Semiconductores Tipo N y P, Ecuación de Shockley, Regulador Zener, LED/Fotodiodo,
Rectificador de Onda Completa y Recortadores")
    print("=" * 78)

    tablas = sintonizar_pid_ziegler_nichols_y_lambda(
        K_planta=1.25, L_retardo=0.15, T_constante=0.85, lambda_deseado=0.35
    )
    print("\n[1] Sintonización Comparativa (Ziegler-Nichols, Cohen-Coon e IMC Lambda):")
    for metodo, gan in tablas.items():
        print(f"    {metodo:24s} -> Kp = {gan['Kp']:6.3f} | Ti = {gan['Ti_s']:6.3f} s | Td = {gan['Td_s']:6.3f}
s")
```

```
metricas = simular_lazo_pid_discreto_tustin(KP_NOMINAL, TI_NOMINAL, TD_NOMINAL, N_FILTRO, 0.005)
print("\n[2] Desempeño Temporal e Índices Integrales (IAE, ISE, ITAE, ITSE):")
for k_m, v_m in metricas.items():
    print(f"    {k_m:34s} : {v_m}")

trama = empaquetar_trama_consigna_sbc(comando=0x01, setpoint_float=7.5)
print("\n[3] Trama Binaria Híbrida SBC -> MCU con CRC-16-CCITT:", trama.hex(" ").upper())
```

6. Programa PLC IEC 61131-3 (ST + IL): Enclavamiento, GRAFCET y Escalado 4-20 mA (automatizacion_plc_iec61131_video_03.st)

```
(* ================================================================================
 * CURSO COMPLETO: FUNDAMENTOS DE ELECTRÓNICA
 * VIDEO 03: Video 03 — Parte 2A: Física de Semiconductores, Diodo de Unión, Zener, Regulador de Tensión, Diodos
Optoelectrónicos y Análisis de Circuitos
 * Archivo: automatizacion_plc_iec61131_video_03.st
 * --------------------------------------------------------------------------------
 * Estándar: IEC 61131-3 — Texto Estructurado (ST) + Equivalencia en Lista de
 *           Instrucciones (IL) y Peldaños de Escalera (Ladder Diagram)
 * Descripción Técnica:
 *   Bloque de Función Industrial (FB_ControladorCeldaMecatronica_V03) que
 *   integra el Ciclo de Barrido (Scan), circuito de enclavamiento prioritario
 *   al paro (cableado NC), secuenciador GRAFCET de 4 etapas, temporizador TON,
 *   contador de lotes CTU, registro de corrimiento de rechazo de calidad SHRB,
 *   escalado analógico 4-20 mA (NORM_X / SCALE_X con detección de rotura de
 *   hilo < 3.6 mA) y algoritmo PID discreto con Anti-Windup.
 * ================================================================================ *)

FUNCTION_BLOCK FB_ControladorCeldaMecatronica_V03
VAR_INPUT
    (* Entradas Digitales de la Tabla de Imagen de Proceso PII (%IX) *)
    xPulsadorMarcha_NO   : BOOL; (* %IX0.0 - Contacto Normalmente Abierto *)
    xPulsadorParo_NC     : BOOL; (* %IX0.1 - Cableado Físicamente NC en Campo *)
    xReleTermico_NC      : BOOL; (* %IX0.2 - Guardamotor NC (1 = Sin Falla) *)
    xSensorPiezaEntrada  : BOOL; (* %IX0.3 - Sensor Inductivo PNP 24V DC *)
    xCamaraPiezaDefecto  : BOOL; (* %IX0.4 - 1 = Envase Defectuoso Detectado *)
    xPulsoEncoderBanda   : BOOL; (* %IX0.5 - Reloj de Desplazamiento de Banda *)
    xResetContadores     : BOOL; (* %IX0.6 - Reinicio Manual de Lote *)

    (* Entradas Analógicas Crudas del Convertidor ADC (%IW) *)
    iEntradaAnalogicaRaw : INT;  (* %IW64  - Lazo 4-20 mA: 0 a 27648 cuentas *)
    rSetpointIngenieria  : REAL := 9.0;
END_VAR

VAR_OUTPUT
    (* Salidas Digitales hacia la Tabla de Imagen de Salidas PIQ (%QX) *)
    xMotorBandaContactor : BOOL; (* %QX0.0 - Contactor Principal Enclavado *)
    xValvulaDosificadora : BOOL; (* %QX0.1 - Solenoide 24V DC con Diodo Flyback *)
    xCilindroExpulsor    : BOOL; (* %QX0.2 - Rechazo en Posición Bit 5 del SHRB *)
    xAlarmaRoturaCable   : BOOL; (* %QX0.3 - Corriente < 3.6 mA o Paro Activo *)
    xLoteCompletado      : BOOL; (* %QX0.4 - Salida Q del Contador CTU *)

    (* Salida Analógica Escalada y Variables de Diagnóstico HMI *)
    iSalidaAnalogicaRaw  : INT;  (* %QW80  - Mando 0..27648 hacia Variador/SSR *)
    rVariableProcesoPV   : REAL; (* Variable física escalada en unidades de ing. *)
    rSalidaControlPID    : REAL; (* Esfuerzo de control 0.0 a 100.0 % *)
    iEstadoGrafcet       : INT;  (* Etapa activa del GRAFCET: 0, 10, 20, 30 *)
END_VAR

VAR
    (* Instancias de Temporizadores y Contadores Estándar IEC 61131-3 *)
    tonDosificado        : TON;
    ctuContadorLotes     : CTU;

    (* Variables Internas de Escalado NORM_X / SCALE_X *)
    rValorNormalizado    : REAL;
    rMinIngenieria       : REAL := 0.0;
    rMaxIngenieria       : REAL := 15.0;

    (* Registro de Corrimiento de 16 Bits (Shift Register para Rechazo) *)
    wRegistroCorrimiento : WORD := 16#0000;
    xPulsoEncoderPrev    : BOOL := FALSE;
    xFlancoEncoder       : BOOL := FALSE;

    (* Memoria Retentiva del Bloque PID Discreto *)
    rKp                  : REAL := 4.000;
    rTi_s                : REAL := 0.350;
    rTd_s                : REAL := 0.050;
```

```
    rTs_s                : REAL := 0.0010;
    rIntegralAcum        : REAL := 0.0;
    rErrorPrev           : REAL := 0.0;
    rPvPrev              : REAL := 0.0;
END_VAR

(* ================================================================================
 * 1. EQUIVALENCIA EN LISTA DE INSTRUCCIONES (IL) DEL PELDAÑO DE ENCLAVAMIENTO:
 *      LD     xPulsadorMarcha_NO
 *      OR     xMotorBandaContactor
 *      AND    xPulsadorParo_NC
 *      AND    xReleTermico_NC
 *      ANDN   xAlarmaRoturaCable
 *      ST     xMotorBandaContactor
 * ================================================================================ *)
xMotorBandaContactor := (xPulsadorMarcha_NO OR xMotorBandaContactor)
                        AND xPulsadorParo_NC
                        AND xReleTermico_NC
                        AND NOT xAlarmaRoturaCable;

(* ================================================================================
 * 2. NORMALIZACIÓN (NORM_X) Y ESCALADO LINEAL (SCALE_X) DE LAZO 4-20 mA
 *    Detección de rotura de hilo si iEntradaAnalogicaRaw < -1728 (< 3.0 mA)
 * ================================================================================ *)
IF (iEntradaAnalogicaRaw < -1728) OR (NOT xPulsadorParo_NC) OR (NOT xReleTermico_NC) THEN
    xAlarmaRoturaCable := TRUE;
    rValorNormalizado  := 0.0;
ELSE
    xAlarmaRoturaCable := FALSE;
    (* NORM_X: Convierte 0..27648 cuentas a rango flotante 0.0..1.0 *)
    rValorNormalizado  := INT_TO_REAL(iEntradaAnalogicaRaw) / 27648.0;
    IF rValorNormalizado < 0.0 THEN rValorNormalizado := 0.0; END_IF;
    IF rValorNormalizado > 1.0 THEN rValorNormalizado := 1.0; END_IF;
END_IF;

(* SCALE_X: Convierte 0.0..1.0 a Unidades de Ingeniería [rMinIngenieria .. rMaxIngenieria] *)
rVariableProcesoPV := rMinIngenieria + rValorNormalizado * (rMaxIngenieria - rMinIngenieria);

(* ================================================================================
 * 3. REGISTRO DE CORRIMIENTO DE BITS (SHRB / BSL) SINCRONIZADO CON ENCODER
 * ================================================================================ *)
xFlancoEncoder    := xPulsoEncoderBanda AND NOT xPulsoEncoderPrev;
xPulsoEncoderPrev := xPulsoEncoderBanda;

IF xFlancoEncoder AND xMotorBandaContactor THEN
    wRegistroCorrimiento := SHL(IN := wRegistroCorrimiento, N := 1);
    IF xCamaraPiezaDefecto THEN
        wRegistroCorrimiento := wRegistroCorrimiento OR 16#0001;
    END_IF;
END_IF;

(* Si el bit desplazado llega a la estación 5 (bit 4, máscara 16#0010), activa expulsor *)
xCilindroExpulsor := (wRegistroCorrimiento AND 16#0010) <> 16#0000;

(* ================================================================================
 * 4. SECUENCIADOR GRAFCET / SFC DE 4 ETAPAS CON TEMPORIZADOR TON Y CONTADOR CTU
 * ================================================================================ *)
IF NOT xMotorBandaContactor THEN
    iEstadoGrafcet := 0;
END_IF;

CASE iEstadoGrafcet OF
    0: (* Etapa 0: Reposo esperando pieza en estación *)
        xValvulaDosificadora := FALSE;
        IF xMotorBandaContactor AND xSensorPiezaEntrada THEN
            iEstadoGrafcet := 10;
        END_IF;
```

```
    10: (* Etapa 10: Dosificación temporizada con bloque TON *)
        xValvulaDosificadora := TRUE;
        IF tonDosificado.Q THEN
            xValvulaDosificadora := FALSE;
            iEstadoGrafcet       := 20;
        END_IF;

    20: (* Etapa 20: Incremento de contador de lote y espera de liberación *)
        xValvulaDosificadora := FALSE;
        IF NOT xSensorPiezaEntrada THEN
            iEstadoGrafcet := 0;
        END_IF;
END_CASE;

tonDosificado(IN := (iEstadoGrafcet = 10), PT := T#1500MS);
ctuContadorLotes(CU := (iEstadoGrafcet = 20), R := xResetContadores, PV := 50);
xLoteCompletado := ctuContadorLotes.Q;

(* ================================================================================
 * 5. BLOQUE TECNOLÓGICO PID DISCRETO CON ANTI-WINDUP Y DES-ESCALADO ANALÓGICO
 * ================================================================================ *)
IF xMotorBandaContactor AND NOT xAlarmaRoturaCable THEN
    VAR
        rErrorActual : REAL;
        rTerminoP    : REAL;
        rTerminoD    : REAL;
        rCandI       : REAL;
        rSumaUnsat   : REAL;
    END_VAR
    rErrorActual := rSetpointIngenieria - rVariableProcesoPV;
    rTerminoP    := rKp * rErrorActual;
    rTerminoD    := -rKp * rTd_s * (rVariableProcesoPV - rPvPrev) / rTs_s;
    rCandI       := rIntegralAcum + (rKp / rTi_s) * rTs_s * 0.5 * (rErrorActual + rErrorPrev);

    rSumaUnsat   := rTerminoP + rCandI + rTerminoD;
    IF rSumaUnsat > 100.0 THEN
        rSalidaControlPID := 100.0;
    ELSIF rSumaUnsat < 0.0 THEN
        rSalidaControlPID := 0.0;
    ELSE
        rSalidaControlPID := rSumaUnsat;
        rIntegralAcum     := rCandI;
    END_IF;

    rErrorPrev := rErrorActual;
    rPvPrev    := rVariableProcesoPV;
ELSE
    rSalidaControlPID := 0.0;
    rIntegralAcum     := 0.0;
END_IF;

(* Conversión de 0.0..100.0 % a 0..27648 cuentas crudas de salida analógica %QW80 *)
iSalidaAnalogicaRaw := REAL_TO_INT((rSalidaControlPID / 100.0) * 27648.0);

END_FUNCTION_BLOCK
```