#include "init.h"

// Variable para guardar el valor de temperatura

float temperatura = 0.0;

// Variable para guardar el valor del setpoint

float setpoint = 0.0;

/\*

 \* @brief Inicializacion de perifericos

 \*/

 void init(void) {

  // Inicializacion de UART

  stdio\_init\_all();

  // Creo un repeating timer

  struct repeating\_timer timer;

  // Creo un callback para la interrupcion del timer

  add\_repeating\_timer\_ms(ADC\_DELAY\_MS, muestreo\_periodico, NULL, &timer);

  // Configuro el I2C0 a 100 KHz de clock

  i2c\_init(i2c0, 100 \* 1000);

  // Elijo GPIO4 como linea de SDA

  gpio\_set\_function(SDA\_GPIO, GPIO\_FUNC\_I2C);

  // Elijo GPIO5 como linea de SCL

  gpio\_set\_function(SCL\_GPIO, GPIO\_FUNC\_I2C);

  // Activo pull-up en ambos GPIO, son debiles por lo que

  // es recomendable usar pull-ups externas

  gpio\_pull\_up(SDA\_GPIO);

  gpio\_pull\_up(SCL\_GPIO);

  // Inicializo LCD

  lcd\_init();

  // Limpio LCD

  lcd\_clear();

  // Escribo lo basico

  lcd\_string("TEMP=");

  lcd\_set\_cursor(1, 0);

  lcd\_string("SETP=");

  // Inicializo ADC

  adc\_init();

  // Inicializo el GPIO del NTC como entrada analogica

  adc\_gpio\_init(NTC\_GPIO);

  // Inicializo el GPIO del potenciometro como entrada analogica

  adc\_gpio\_init(SP\_GPIO);

  // Inicializacion de PWM

  // - 100 Hz de frecuencia

  // - Ancho de pulso inicial del 0%

}

/\*

 \* @brief Callback para la interrupcion de timer

 \* @param t: puntero a repeating\_timer

 \*/

bool muestreo\_periodico(struct repeating\_timer \*t) {

  // Canal elegido, no se pierde el valor aunque sea local (static)

  static uint8\_t selected\_channel = 0;

  // Constante de proporcionalidad de NTC

  const uint16\_t beta = 3950;

  // Actualizar el canal seleccionado

  selected\_channel = selected\_channel ^ 1;

  // Cambio de canal 0 a 1 o viceversa

  adc\_select\_input(selected\_channel);

  // Lectura analogica (variable adc\_value)

  uint16\_t adc\_value = adc\_read();

  // Verifico que canal lei

  if(selected\_channel == 0) {

    // Canal 0: termistor. Calcular valor de temperatura (variable temperatura)

    temperatura = 1 / (log(1 / (4095. / adc\_value - 1)) / beta + 1.0 / 298.15) - 273.15;

  }

  else {

    // Canal 1: potenciometro. Calcular el valor de temperatura equivalente (4095 = 35 grados)

    setpoint = 9 \* 4095 / 35

    // Calcular error (temperatura - setpoint)

    float error = temperatura - setpoint;

    // Llamar al control proporcional

    float control\_p = control\_proporcional(error);

    // Verificar el signo del control

    if(control\_p < 0) {

      // Si es negativo: temperatura > setpoint -> enfrio

      // Apagar un canal de PWM

      // Encender el otro

    }

    else {

      // Si es positivo: temperatura < setpoint -> caliento

      // Apagar un canal de PWM

      // Encender el otro

    }

  }

}