💻 CÓDIGO Y ALGORITMOS DE CONTROL MECATRÓNICO — MÓDULO 05

Curso: Curso Completo: Introducción a la Mecatrónica

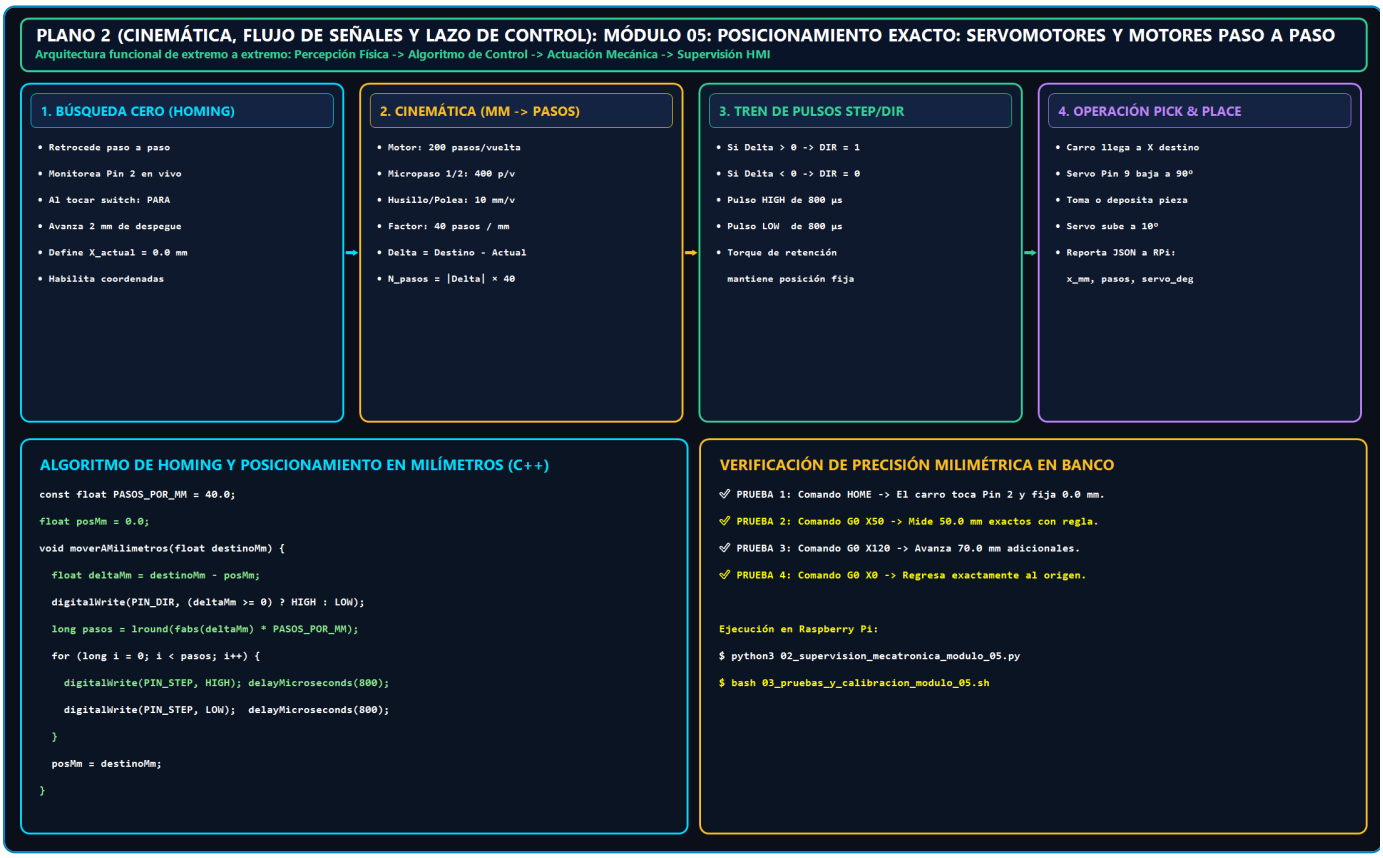
Módulo 05: Módulo 05: Posicionamiento Exacto: Servomotores y Motores Paso a Paso (Stepper + A4988)

Archivos en `codigo_fuente/`:

- `01_firmware_arduino_modulo_05.ino` (Control en tiempo real sobre Arduino UNO)
- `02_supervision_mecatronica_modulo_05.py` (Supervisión HMI y registro CSV/SQLite en Raspberry Pi)
- `03_pruebas_y_calibracion_modulo_05.sh` (Script automatizado de validación de banco)

---

1. Arquitectura de Control y Flujo de Señales (2200×1350)



Plano 2 — Arquitectura de Control y Cinemática del Módulo 05

---

2. Firmware Completo de Tiempo Real (`01_firmware_arduino_modulo_05.ino`)

```
/*
 * ============================================================================
 * CURSO COMPLETO: INTRODUCCIÓN A LA MECATRÓNICA — MÓDULO 05
 * Archivo: 01_firmware_arduino_modulo_05.ino
 * Prototipo: Eje Lineal CNC con Motor Paso a Paso NEMA 17 (A4988) + Servo
 * Hardware: Arduino UNO + Driver A4988 (STEP=3, DIR=4, EN=5)
 *           + Final de Carrera Homing (Pin 2) + Pinza Servomotor (Pin 9)
 * ============================================================================
 */

#include <Arduino.h>
#include <Servo.h>

const uint8_t PIN_HOMING_SW = 2; // Final de carrera en el origen X = 0.0 mm
const uint8_t PIN_STEP      = 3; // Tren de pulsos de paso
const uint8_t PIN_DIR       = 4; // Dirección del eje lineal
const uint8_t PIN_ENABLE    = 5; // Habilitación del driver A4988 (LOW = Activo)
const uint8_t PIN_SERVO     = 9; // Servomotor de la pinza / cabezal

// --- Constantes Cinemáticas del Eje Lineal ---
const float PASOS_POR_MM        = 40.0;  // 40 pasos = 1.0 mm de avance lineal
const float RECORRIDO_MAX_MM    = 200.0; // Longitud máxima útil del carril (mm)
const unsigned int PULSO_US     = 800;   // Semiperiodo del pulso STEP en microsegundos

Servo pinzaServo;
float posicionActualMm = 0.0;
long pasosAcumulados   = 0;
int anguloPinza        = 10;
bool ejeReferenciado   = false;

void emitirPasos(long cantidadPasos, bool sentidoPositivo, unsigned int retardoUs) {
  digitalWrite(PIN_DIR, sentidoPositivo ? HIGH : LOW);
  delayMicroseconds(20);
  for (long i = 0; i < cantidadPasos; i++) {
    // Seguridad: si retrocede y toca el tope de origen, detiene el movimiento
    if (!sentidoPositivo && digitalRead(PIN_HOMING_SW) == LOW) {
      break;
    }
    digitalWrite(PIN_STEP, HIGH);
    delayMicroseconds(retardoUs);
    digitalWrite(PIN_STEP, LOW);
    delayMicroseconds(retardoUs);
    pasosAcumulados += (sentidoPositivo ? 1 : -1);
  }
}

void ejecutarHoming() {
  digitalWrite(PIN_ENABLE, LOW); // Energizar bobinas del NEMA 17
  pinzaServo.write(10);          // Abrir pinza antes de mover el eje
  anguloPinza = 10;

  // Retroceder suavemente hasta accionar el microswitch del Pin 2
  digitalWrite(PIN_DIR, LOW);
  long limiteSeguridad = (long)(RECORRIDO_MAX_MM * PASOS_POR_MM * 1.2);
  for (long i = 0; i < limiteSeguridad; i++) {
    if (digitalRead(PIN_HOMING_SW) == LOW) {
      break;
```

```
    }
    digitalWrite(PIN_STEP, HIGH);
    delayMicroseconds(1100);
    digitalWrite(PIN_STEP, LOW);
    delayMicroseconds(1100);
  }

  // Avanzar 2.0 mm de despegue y fijar el cero absoluto de máquina
  emitirPasos((long)(2.0 * PASOS_POR_MM), true, 1000);
  posicionActualMm = 0.0;
  pasosAcumulados  = 0;
  ejeReferenciado  = true;

  Serial.println(F("{"modulo":5,"evento":"HOMING_COMPLETADO","x_mm":0.0,"pasos":0}"));
}

void moverACoordenadaMm(float destinoMm) {
  if (!ejeReferenciado) {
    ejecutarHoming();
  }
  destinoMm = constrain(destinoMm, 0.0, RECORRIDO_MAX_MM);
  float deltaMm = destinoMm - posicionActualMm;
  long pasosNecesarios = lround(fabs(deltaMm) * PASOS_POR_MM);

  if (pasosNecesarios > 0) {
    emitirPasos(pasosNecesarios, (deltaMm >= 0.0), PULSO_US);
    posicionActualMm = destinoMm;
  }

  // Operación Pick & Place con el servomotor al llegar a la coordenada
  pinzaServo.write(90);
  anguloPinza = 90;
  delay(250);
  pinzaServo.write(10);
  anguloPinza = 10;

  Serial.print(F("{"modulo":5,"homing_ok":"));
  Serial.print(ejeReferenciado ? 1 : 0);
  Serial.print(F(","x_mm":"));
  Serial.print(posicionActualMm, 2);
  Serial.print(F(","pasos":"));
  Serial.print(pasosAcumulados);
  Serial.print(F(","servo_deg":"));
  Serial.print(anguloPinza);
  Serial.println(F("}"));
}

void setup() {
  Serial.begin(115200);

  pinMode(PIN_HOMING_SW, INPUT_PULLUP);
  pinMode(PIN_STEP, OUTPUT);
  pinMode(PIN_DIR, OUTPUT);
  pinMode(PIN_ENABLE, OUTPUT);

  digitalWrite(PIN_ENABLE, LOW); // Activar driver A4988
  pinzaServo.attach(PIN_SERVO);
  pinzaServo.write(10);
```

```
  ejecutarHoming();
}

void loop() {
  if (Serial.available() > 0) {
    String cmd = Serial.readStringUntil('\n');
    cmd.trim();
    cmd.toUpperCase();

    if (cmd == "HOME" || cmd == "G28") {
      ejecutarHoming();
    } else if (cmd.startsWith("G0 X")) {
      float xDestino = cmd.substring(4).toFloat();
      moverACoordenadaMm(xDestino);
    } else if (cmd.startsWith("SERVO:")) {
      int deg = constrain(cmd.substring(6).toInt(), 0, 180);
      pinzaServo.write(deg);
      anguloPinza = deg;
    }
  }
}
```

---

3. Software de Supervisión y Telemetría (`02_supervision_mecatronica_modulo_05.py`)

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
================================================================================
CURSO COMPLETO: INTRODUCCIÓN A LA MECATRÓNICA — MÓDULO 05
Archivo: 02_supervision_mecatronica_modulo_05.py
Descripción: Controlador de coordenadas CNC / Pick-and-Place en Raspberry Pi.
             Envía órdenes G28 (Homing) y G0 X(mm) al eje lineal y audita
             la exactitud entre milímetros solicitados y pasos ejecutados.
================================================================================
"""

import csv
import json
import os
import sys
import time
from datetime import datetime

try:
    import serial
except ImportError:
    serial = None

PUERTO_SERIE = os.environ.get("MECATRONICA_PORT", "/dev/ttyACM0")
BAUDIOS      = 115200
ARCHIVO_CSV  = "posicionamiento_cnc_modulo_05.csv"


def inicializar_csv(ruta: str) -> None:
    if not os.path.exists(ruta):
        with open(ruta, mode="w", newline="", encoding="utf-8") as f:
            csv.writer(f).writerow(["timestamp", "homing_ok", "x_mm", "pasos", "servo_deg"])


def registrar_coordenada(d: dict) -> None:
    ts  = datetime.now().strftime("%H:%M:%S.%f")[:-3]
    hom = d.get("homing_ok", 1)
    xmm = d.get("x_mm", 0.0)
    pas = d.get("pasos", 0)
    srv = d.get("servo_deg", 10)
    print(f"[{ts}] Homing:{'OK' if hom else 'NO'} | Coordenada X:{xmm:6.2f} mm | Pasos:{pas:5d} |
Pinza Servo:{srv:3d}°")
    with open(ARCHIVO_CSV, mode="a", newline="", encoding="utf-8") as f:
        csv.writer(f).writerow([ts, hom, xmm, pas, srv])


def ejecutar_demo() -> None:
    print("ℹ️ Simulando rutina G28 (Homing) y secuencia Pick & Place CNC (40 pasos/mm)...")
    trayectoria = [
        {"homing_ok": 1, "x_mm": 0.00,   "pasos": 0,    "servo_deg": 10},
        {"homing_ok": 1, "x_mm": 45.00,  "pasos": 1800, "servo_deg": 90},
        {"homing_ok": 1, "x_mm": 120.50, "pasos": 4820, "servo_deg": 90},
        {"homing_ok": 1, "x_mm": 0.00,   "pasos": 0,    "servo_deg": 10},
    ]
    for p in trayectoria:
        registrar_coordenada(p)
```

```
        time.sleep(0.35)


def main() -> None:
    print("=" * 78)
    print(" CONTROLADOR CNC Y PICK & PLACE MÓDULO 05 — STEPPER NEMA 17 + SERVO")
    print("=" * 78)
    inicializar_csv(ARCHIVO_CSV)
    if "--demo" in sys.argv or serial is None:
        ejecutar_demo()
        return
    try:
        with serial.Serial(PUERTO_SERIE, BAUDIOS, timeout=1.0) as ser:
            while True:
                linea = ser.readline().decode("utf-8", errors="ignore").strip()
                if linea.startswith("{") and "x_mm" in linea:
                    registrar_coordenada(json.loads(linea))
    except Exception as e:
        print(f"⚠️ Sin conexión serie ({e}). Ejecutando trayectoria de demostración:")
        ejecutar_demo()


if __name__ == "__main__":
    main()
```