

Fundamentos de Electrónica para ingenieros en Inteligencia Artificial



**Aprendizaje guiado
con scripts en Python**

Dr. Javier Salas García

Fundamentos de Electrónica para ingenieros en Inteligencia Artificial: Aprendizaje guiado con scripts en Python

Autor:

Salas-García, Javier

Detalles de publicación:

Editorial: KDP Amazon

ISBN: 9798244017144

Fecha de publicación: Enero 2026

Cómo citar esta obra:

Salas-García, J. (2026). *Fundamentos de Electrónica para ingenieros en Inteligencia Artificial: Aprendizaje guiado con scripts en Python*. KDP Amazon.

Aviso legal:

Todos los derechos reservados. Ninguna parte de esta publicación puede ser reproducida, distribuida o transmitida de ninguna forma o por cualquier medio, incluyendo fotocopias, grabaciones u otros métodos electrónicos o mecánicos, sin el permiso previo por escrito del autor, excepto en el caso de breves citas incorporadas en revisiones críticas y ciertos otros usos no comerciales permitidos por la ley de derechos de autor.

Impreso por Amazon.

Esta página se dejó en blanco intencionalmente.

Índice general

Prefacio	9
1. Instrumentación y Fundamentos de Circuitos	11
1.1. El Multímetro: De la Analogía Hidráulica al Dataset	11
1.1.1. Voltaje: La Altura de la Cascada (Potencial)	11
1.1.2. Corriente: El Caudal del Río (Flujo)	13
1.1.3. Mediciones Avanzadas: True RMS y la Mentira del Promedio	13
1.2. El Osciloscopio: La Máquina del Tiempo para Señales . . .	15
1.2.1. Del Tubo de Rayos Catódicos al Dominio Digital . .	16
1.2.2. La Anatomía de la Observación: Escalas y Sincronización	17
1.2.3. El Teorema de Nyquist en Acción: Cuando el Instrumento Miente	18
1.2.4. Los Asesinos Silenciosos de Datos: Glitches y Jitter .	19
1.2.5. Más Allá del Trigger Básico: Cazando Anomalías . .	20
1.2.6. Decodificación de Protocolos: Del Ruido al Significado	21
1.3. Fuentes de Alimentación: El Corazón Invisible de tus Sistemas	22
1.3.1. Lineal vs. Conmutada: El Compromiso Fundamental	23
1.3.2. Ripple: La Ondulación que Envenena tus ADCs . . .	23
1.3.3. Regulación de Línea y Carga: Cuando el Voltaje No es Constante	24
1.3.4. El PSRR Olvidado: Tu Última Línea de Defensa . .	26
1.3.5. Transitorios de Encendido: El Caos de los Primeros Milisegundos	27
1.3.6. Medición de Calidad de Fuentes: Más Allá del Multímetro	28
1.4. Componentes Reactivos: Cuando la Frecuencia Importa . .	29
1.4.1. La Resistencia: El Componente que No Discrimina .	30
1.4.2. El Capacitor: Memoria de Voltaje y Filtro de Frecuencia	31
1.4.3. El Inductor: Inercia Electromagnética y Supresión de Transitorios	32
1.4.4. Filtros RC: La Primera Línea de Defensa Contra el Caos	33
1.4.5. Respuesta Transitoria: Cuando Enciendes el Sistema	34
1.4.6. Resonancia LC: Cuando Dos Reactivos Combaten .	36
1.5. Leyes Fundamentales: La Física que Gobierna tus Datos . .	37
1.5.1. Ley de Ohm: Más que una Simple Ecuación	37

1.5.2.	Ley de Watt: Donde el Voltaje y la Corriente se Encuentran con la Termodinámica	38
1.5.3.	Leyes de Kirchhoff: Conservación de la Carga y la Energía	40
1.5.4.	Análisis de Mallas y Nodos: Ingeniería Sistemática vs Adivinanza	41
1.5.5.	Divisores de Voltaje: La Configuración Más Útil y Más Malinterpretada	41
1.5.6.	Medición de Corriente: El Problema del Shunt	43
1.6.	Carga, Tensión y Corriente: Los Fundamentos Microscópicos	44
1.6.1.	Carga Eléctrica: La Propiedad Cuantizada que Gobierna Todo	45
1.6.2.	Tensión Eléctrica: Energía Potencial por Unidad de Carga	46
1.6.3.	Corriente Eléctrica: Flujo de Carga en Movimiento	48
1.6.4.	Densidad de Corriente y Calentamiento por Efecto Joule	48
1.6.5.	Capacitancia e Inductancia: Almacenamiento de Energía en Campos	50
1.7.	Arreglos Serie-Paralelo: Arquitecturas de Circuitos y Distribución de Señales	51
1.7.1.	Resistencias en Serie: Arquitectura de Cadena Vulnerable	52
1.7.2.	Resistencias en Paralelo: Arquitectura Redundante con División de Corriente	54
1.7.3.	Redes Mixtas Serie-Paralelo: Simplificación Sistemática	57
1.7.4.	Teorema de Thévenin y Norton: Abstracción de Redes Complejas	58
1.7.5.	Topologías de Sensores: Serie vs Paralelo en Redes Distribuidas	59
1.8.	Medición en Arreglos Complejos: Técnicas de Diagnóstico de Circuitos	61
1.8.1.	Medición de Voltaje: Mapeo Topológico	61
1.8.2.	Medición de Corriente: Rastreo del Flujo de Carga	63
1.8.3.	Medición de Resistencia: Detección de Fallas Pasivas	65
1.8.4.	Técnicas Avanzadas: Medición de Ruido y Oscilaciones Parásitas	66
2.	Semiconductores y Componentes Activos	69
2.1.	El Diodo: Rectificación y No-Linealidad Fundamental	69
2.1.1.	La Curva I-V del Diodo: Exponencial y Asimétrica	70
2.1.2.	Rectificación de Media Onda y Onda Completa	71
2.1.3.	Diodos de Protección y Clamp Circuits	72
2.2.	El Transistor BJT: Amplificación y Control de Corriente	75
2.2.1.	Curvas Características y Regiones de Operación	76

2.2.2.	Amplificador en Configuración de Emisor Común . .	77
2.2.3.	BJT como Switch: Saturación Profunda y Tiempos de Conmutación	78
2.3.	El Transistor MOSFET: Switching Rápido y Baja Potencia	82
2.3.1.	Curvas Características y Resistencia On-State	83
2.3.2.	MOSFET como Switch de Potencia: PWM y Control de Velocidad	84
2.3.3.	Comparación BJT vs MOSFET y Selección de Tecnología	85
2.4.	El Amplificador Operacional: Bloque Universal de Procesamiento Analógico	88
2.4.1.	El Amplificador Ideal y las Reglas de Oro	89
2.4.2.	Circuitos de Acondicionamiento: Sumador, Restador, e Instrumentación	91
2.4.3.	Filtros Activos y Limitaciones de Frecuencia	93
3.	Interfaz Digital-Analógico	97
3.1.	Conversión Analógico-Digital: Cuantización y Muestreo	97
3.1.1.	Resolución, Cuantización, y Ruido	98
3.1.2.	Muestreo y Teorema de Nyquist	100
3.1.3.	Arquitecturas de ADC: SAR, Sigma-Delta, y Flash	102
3.2.	Conversión Digital-Analógico: Reconstrucción y Síntesis de Señales	104
3.2.1.	Arquitecturas de DAC: Resistor String, R-2R, y PWM	105
3.2.2.	Reconstrucción de Señales y Glitches	106
3.2.3.	DACs en Sistemas de Control y Síntesis de Audio	107
3.3.	Modulación por Ancho de Pulso: Control de Potencia y Generación de Señales	111
3.3.1.	Generación de PWM con Microcontroladores	112
3.3.2.	Control de Velocidad de Motores DC y Torque	113
3.3.3.	PWM para LEDs y Percepción Visual	115
3.4.	Protocolos de Comunicación: UART, SPI e I2C	117
3.4.1.	UART: Comunicación Asíncrona Punto-a-Punto	118
3.4.2.	SPI: Comunicación Síncrona de Alta Velocidad	119
3.4.3.	I2C: Bus Multi-Dispositivo con Direcccionamiento	119
4.	Gestión de Potencia y Diseño de Sistemas	123
4.1.	Fuentes de Alimentación y Regulación de Voltaje	123
4.1.1.	Reguladores Lineales	123
4.1.2.	Reguladores Conmutados	124
4.1.3.	Gestión de Ruido	124
4.2.	Baterías y Gestión de Energía	125
4.2.1.	Curvas de Descarga y Capacidad Efectiva	126
4.2.2.	Sistemas de Gestión de Batería (BMS)	126
4.2.3.	Cálculo de Autonomía y Presupuesto de Potencia	127

4.3.	Ruido, Blindaje, y Compatibilidad Electromagnética	128
4.3.1.	Fuentes de Ruido y Mecanismos de Acoplamiento . .	129
4.3.2.	Técnicas de Grounding y Segregación	129
4.3.3.	Filtrado y Blindaje	129
4.4.	Diseño de PCB y Layout para Alta Integridad de Señal . .	131
4.4.1.	Cálculo de Ancho de Trazas y Gestión Térmica . . .	131
4.4.2.	Capas, Stackup, y Impedancia Controlada	131
4.4.3.	Reglas de Diseño y DFM	132
5.	Integración de Sistemas para IA	135
5.1.	Acondicionamiento de Señales Analógicas	135
5.1.1.	Amplificación y Ajuste de Ganancia	135
5.1.2.	Offset y Nivel DC	136
5.1.3.	Filtrado Antialiasing	136
5.2.	Microcontroladores y Arquitectura de Sistemas Embebidos	138
5.2.1.	Arquitectura ARM Cortex-M y Periféricos	139
5.2.2.	ESP32: WiFi, Bluetooth, y Dual-Core	140
5.2.3.	Tiempo Real y Determinismo	140
5.3.	Sensores para Sistemas de IA: IMU, Cámaras, y Micrófonos	142
5.3.1.	IMU (Inertial Measurement Unit)	142
5.3.2.	Cámaras para Visión por Computadora	142
5.3.3.	Micrófonos MEMS y Audio	143
5.4.	Edge Computing y TinyML: Optimización para Microcon- troladores	144
5.4.1.	Cuantización y Compresión de Modelos	145
5.4.2.	Frameworks para Edge: TensorFlow Lite Micro . . .	146
5.4.3.	Performance y Trade-offs	146
A.	Cronograma de Actividades y Prácticas	149
B.	Actividades de Aula Invertida	153
B.1.	Unidad 1: Instrumentación	154
B.1.1.	1.1: El Multímetro (30 min)	154
B.1.2.	1.2: El Osciloscopio (25 min)	154
B.1.3.	1.3: Fuentes de Alimentación (20 min)	154
B.1.4.	1.4: Filtros Pasivos (35 min)	155
B.1.5.	1.5: Leyes de Kirchhoff (20 min)	155
B.1.6.	1.6: Carga, Tensión y Corriente (25 min)	155
B.1.7.	1.7: Serie y Paralelo (30 min)	156
B.1.8.	1.8: Medición en Arreglos de Resistencias (30 min) .	156
B.2.	Unidad 2: Semiconductores	157
B.2.1.	2.1: Diodos (25 min)	157
B.2.2.	2.2: Transistor Bipolar versus Transistor de Efecto de Campo (30 min)	157
B.2.3.	2.3: Fallo de Transistor de Potencia (25 min)	157

B.2.4.	2.4: Amplificador con Amplificador Operacional (35 min)	158
B.3.	Unidad 3: Conversión ADC/DAC	159
B.3.1.	3.1: Arquitecturas de Convertidor Analógico-Digital (30 min)	159
B.3.2.	3.2: Solapamiento Espectral en Sistema Real (30 min)	159
B.3.3.	3.3: Convertidor Digital-Analógico para Audio (25 min)	160
B.3.4.	3.4: Control de Motor por Modulación de Ancho de Pulso (35 min)	161
B.3.5.	3.5: Protocolo de Comunicación (30 min)	162
B.4.	Unidad 4: Potencia y Diseño PCB	163
B.4.1.	4.1: Regulador Lineal versus Regulador Conmutado (30 min)	163
B.4.2.	4.2: Batería se Drena Rápido (30 min)	164
B.4.3.	4.3: Ruido Electromagnético (25 min)	165
B.4.4.	4.4: Diseño de Placa de Circuito Impreso (35 min)	165
B.5.	Unidad 5: Microcontroladores e IA	167
B.5.1.	5.1: Arquitecturas de Microcontrolador (30 min)	167
B.5.2.	5.2: Fluctuación de Sistema Operativo en Tiempo Real (30 min)	168
B.5.3.	5.3: Tubería de Procesamiento de Sensores (30 min)	169
B.5.4.	5.4: Sistema de Cómputo en el Borde (35 min)	171

Esta página se dejó en blanco intencionalmente.

Prefacio

Modelos que alcanzan precisiones del 99% colapsan cuando los datos son ruidosos o capturados con hardware inadecuado. Un dataset corrupto destruye un proyecto de machine learning más rápido que un algoritmo deficiente. Este libro nace para cerrar la brecha entre el mundo físico de los sensores y el mundo digital de los modelos: antes de entrenar redes neuronales, es crucial garantizar que el ruido no entierre la información.

«Fundamentos de Electrónica para Ingenieros de IA» no pretende formar diseñadores de circuitos integrados, sino ingenieros capaces de entender qué están midiendo. El objetivo es pragmático: diagnosticar si un modelo falla por el algoritmo o por un sensor saturado, y entender cómo las decisiones de hardware afectan la calidad del dataset.

La estructura del texto avanza desde la instrumentación básica hasta sistemas completos de Edge AI. La Unidad 1 y 2 cubren las herramientas y el acondicionamiento de señales. La Unidad 3 aborda la conversión digital-analógica y sus errores físicos. La Unidad 4 integra microcontroladores y protocolos de comunicación, y la Unidad 5 culmina con Edge Computing y TinyML. El enfoque es práctico: teoría mínima, ejemplos cuantificados y scripts de Python para simular cada concepto. No se trata solo de leer ecuaciones, sino de modificar código y ver los resultados.

Este texto es un puente que conecta el dominio analógico con el digital. Si al terminar el curso puedes leer un datasheet, diseñar el acondicionamiento de un sensor, capturar datos limpios y entrenar un modelo robusto, el objetivo se habrá cumplido. Bienvenido al lado analógico de la inteligencia artificial.

Recursos Digitales: Todos los scripts de Python mencionados en este libro, necesarios para generar las figuras y realizar las simulaciones, pueden descargarse gratuitamente desde la página: <https://electrovigia.net/libros>

Javier Salas-García

Enero de 2025

Esta página se dejó en blanco intencionalmente.

Capítulo 1

Instrumentación y Fundamentos de Circuitos

1.1 El Multímetro: De la Analogía Hidráulica al Dataset

Objetivo de Aprendizaje

Objetivo: **Analizar** el comportamiento de instrumentos de medición eléctrica **mediante** la experimentación con simulaciones en Python y el uso físico del multímetro, **para** garantizar la integridad de los datos adquiridos y evitar daños por mala operación en sistemas de Inteligencia Artificial.

Ingeniería de Datos desde el Hardware

En Inteligencia Artificial, solemos decir «Basura entra, basura sale». Si entrenas una red neuronal con datos de sensores mal medidos, el modelo aprenderá ruido. El multímetro es tu herramienta para validar la «Verdad Terrestre»(Ground Truth). Antes de confiar en un archivo CSV, debes confiar en tus electrones.

Seguramente has escuchado que la electricidad se parece al agua fluyendo por tuberías. Aunque esta analogía tiene límites, es extraordinariamente útil para entender por qué medimos el voltaje y la corriente de formas tan distintas, y por qué confundirlas puede costarte un fusible o todo el instrumento.

1.1.1 Voltaje: La Altura de la Cascada (Potencial)

El voltaje es, en esencia, presión o fuerza potencial. Imagina una cascada: el agua en la cima tiene una energía potencial alta respecto al agua en el fondo. Si quieres saber «cuánta energía»tiene esa cascada, no necesitas detener el agua; solo necesitas medir la altura desde la cima hasta el fondo.

Esto nos lleva a la primera regla de oro: **El voltaje se mide en paralelo**. Tu multímetro, configurado como voltímetro, actúa como una regla de medir extremadamente ligera que tocas en dos puntos del circuito sin alterar lo que sucede en él. Para lograr esto, el voltímetro tiene una **Impedancia de Entrada (Z_{in})** altísima, típicamente de $10\text{ M}\Omega$. Es como si la regla fuera etérea para no molestar al agua.

Pero, ¿qué pasa si tu regla no es tan «ligera»? Aquí entramos en un concepto de ingeniería crítica llamado **Efecto de Carga**. Si la resistencia interna de tu voltímetro no es infinitamente mayor que la resistencia del circuito que mides, el instrumento «robará» corriente para funcionar, alterando el voltaje que intentas leer.

Observa la siguiente figura generada con nuestros scripts de simulación. Muestra cómo el error de medición se dispara cuando intentas medir circuitos de alta impedancia (como sensores biológicos) con un multímetro estándar.

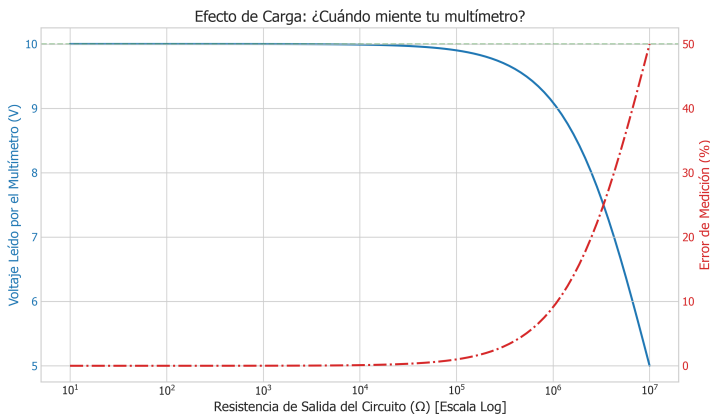


Figura 1.1: Gráfica del error porcentual de medición en función de la resistencia del circuito. Se observa que cuando la resistencia del circuito se acerca a la del instrumento ($10^7\Omega$), el error es catastrófico. (Figura generada con el script `s1_1_e1.py`)

Reto Computacional: Experimenta con el Script

Abre el script `s1_1_e1.py`. Modifica la variable `R_meter` cambiándola de `10e6` (Multímetro digital bueno) a `10e3` (Multímetro analógico viejo de $10k\Omega$). Ejecuta el script.

Pregunta: ¿Cómo cambia la curva de error? ¿Qué nos dice esto sobre usar equipos antiguos para medir sensores modernos de IA?

1.1.2 Corriente: El Caudal del Río (Flujo)

Ahora hablemos de la corriente (Amperios). Esto es la cantidad de agua que pasa por el tubo cada segundo. Para medir esto, no puedes quedarte mirando desde afuera como con la altura (voltaje). Tienes que «cortar» la tubería e insertar un contador de flujo en medio para que toda el agua pase a través de él.

Por eso, la segunda regla de oro es: **La corriente se mide en serie**. Debes abrir físicamente el circuito e insertar el multímetro para que forme parte del camino de los electrones.

Para no estorbar el flujo del río, el amperímetro debe tener una resistencia interna cercana a cero (idealmente 0Ω). Es un tubo muy ancho y liso. Y aquí radica el peligro mortal para tu equipo. Muchos estudiantes, acostumbrados a medir voltaje (paralelo), olvidan cambiar las puntas o la configuración y conectan el amperímetro en paralelo a una fuente de energía.

¿Qué ocurre? Estás conectando un cable de casi cero resistencia directo a la toma de corriente. Según la Ley de Ohm ($I = V/R$), si R tiende a cero, la corriente tiende a infinito.

¡Peligro de Daño al Equipo!

¡Detente y piensa! Si conectas el multímetro en modo Amperímetro (resistencia interna $\approx 0\Omega$) directamente a una batería (paralelo), estás creando un cortocircuito directo.

Analicemos gráficamente este desastre. La Figura 1.2 muestra cómo la corriente se dispara exponencialmente a medida que la resistencia del circuito baja, cruzando rápidamente el umbral de destrucción del fusible.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_1_e2.py`. Cambia el valor de `V_fuente` a 120V (como si lo conectaras a un enchufe de casa por error).

Tarea: Observa el eje Y. ¿De qué magnitud es la corriente teórica? ¿Crees que el pequeño fusible de vidrio del multímetro pueda contener esa explosión de energía?

1.1.3 Mediciones Avanzadas: True RMS y la Mentira del Promedio

Finalmente, hablemos de la calidad del dato. En electrónica básica se usan ondas senoidales perfectas. Pero en IA, analizamos vibraciones de motores, voces o señales PWM de robots, que son ruidosas y complejas.

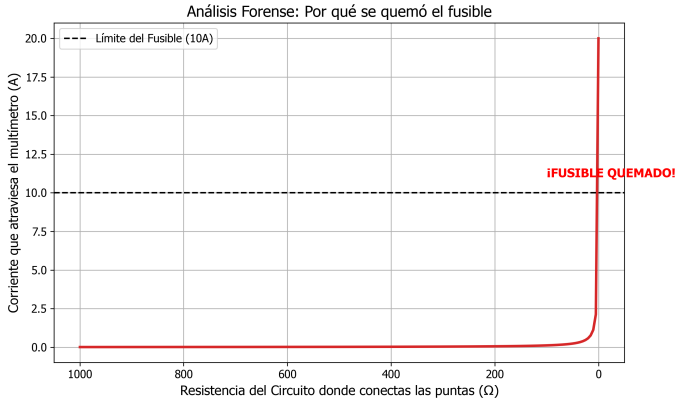


Figura 1.2: Simulación de un error de conexión. Al conectar el amperímetro en paralelo, la resistencia efectiva cae drásticamente, provocando un pico de corriente que destruye el fusible instantáneamente. (Figura generada con el script `s1_1.e2.py`)

Los multímetros económicos asumen que todo es una onda senoidal. Miden el promedio y lo multiplican por una constante (1.11) para adivinar el valor eficaz (RMS). Pero si mides una señal cuadrada de un sensor digital o ruido blanco con estos aparatos, te darán un número falso. Para ingeniería de datos, necesitas un multímetro **True RMS**, que calcula matemáticamente la integral de la onda en tiempo real:

$$V_{RMS} = \sqrt{\frac{1}{T} \int_0^T [v(t)]^2 dt}$$

La Figura 1.3 expone esta discrepancia. Un ingeniero que use un multímetro barato reportará un voltaje menor al real, lo que podría llevarte a diseñar mal la fuente de alimentación de tu GPU o procesador neuronal.

Reto Computacional: Experimenta con el Script

En el script `s1_1.e3.py`, aumenta la variable `ruido` (la desviación estándar de la distribución normal).

Reto: Encuentra qué tanto ruido se necesita para que el error entre el multímetro barato y el True RMS sea mayor a 1 Voltio. Este experimento simula un entorno industrial ruidoso.

Como ves, medir no es solo leer un número. Es entender la interacción física entre tu herramienta y el mundo. En la siguiente sesión, llevaremos esto al laboratorio para caracterizar el error real de nuestros equipos.

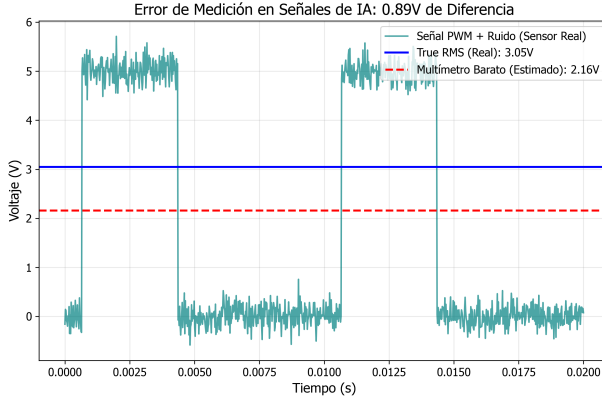


Figura 1.3: Comparativa entre el valor True RMS (real) y el valor estimado por un multímetro de bajo costo ante una señal PWM con ruido. La diferencia representa el error de «ceguera» del instrumento. (Figura generada con el script `s1_1.e3.py`)

1.2 El Osciloscopio: La Máquina del Tiempo para Señales

Objetivo de Aprendizaje

Objetivo: Analizar señales eléctricas dinámicas en el dominio del tiempo mediante el uso del osciloscopio y la interpretación de formas de onda, para diagnosticar problemas en sistemas de adquisición de datos y validar la calidad de señales antes de su digitalización en aplicaciones de IA.

Ingeniería de Datos desde el Hardware

Cuando entrenas un modelo de IA para reconocimiento de voz o detección de anomalías en motores, no solo importa el «qué» mides, sino el «cuándo». El osciloscopio es tu ventana temporal: te permite ver si una señal tiene ruido, distorsión o retardos que tu ADC convertirá en basura digital. Si tu dataset tiene glitches de 10 microsegundos que el osciloscopio revela pero tu código ignora, estás entrenando con datos corruptos.

El multímetro te da un número estático, un promedio congelado en el tiem-

po. Pero la realidad eléctrica es dinámica. Las señales oscilan, pulsan, se deforman. Imagina intentar diagnosticar un motor usando solo una fotografía: podrías ver que existe, pero no sabrías si gira, vibra o está atascado. El osciloscopio es la cámara de alta velocidad del mundo eléctrico, y en el contexto de la inteligencia artificial, se convierte en tu detector de mentiras para validar que los datos que alimentan a tus modelos no están envenenados desde el origen. Porque puedes tener la arquitectura de red neuronal más sofisticada del mundo, puedes afinar hiperparámetros durante semanas, pero si tus sensores entregan señales distorsionadas que nunca detectaste, tu modelo aprenderá artefactos en lugar de patrones reales.

1.2.1 Del Tubo de Rayos Catódicos al Dominio Digital

Históricamente, los osciloscopios analógicos usaban un haz de electrones que «pintaba» la forma de onda en una pantalla fosforescente. Los electrones golpeaban el fósforo depositado en el vidrio y generaban luz verdosa que persistía por fracciones de segundo, permitiendo que el ojo humano integrara visualmente las trazas de señales rapidísimas. Hoy, los osciloscopios digitales (DSO, por sus siglas en inglés) abandonaron esa tecnología casi mágica y adoptaron un enfoque computacional: muestrean la señal a velocidades vertiginosas-millones o miles de millones de veces por segundo- y reconstruyen matemáticamente la forma de onda en una pantalla LCD de alta resolución. Este proceso de muestreo y reconstrucción no es solo una mejora técnica superficial; es idéntico conceptualmente al que ocurre en un convertidor analógico-digital (ADC) cuando digitalizas datos de sensores para procesamiento con IA. Entender las limitaciones de este proceso aquí, con un osciloscopio en la mano, te salvará de cometer exactamente los mismos errores cuando diseñes tu pipeline de adquisición de datos para machine learning.

La especificación más importante de un osciloscopio- y esto sorprende a muchos estudiantes- no es su número de canales ni el tamaño glamoroso de su pantalla táctil a color. Es su **tasa de muestreo** (medida en MS/s o GS/s, mega o giga-muestras por segundo) y su **ancho de banda** (medido en MHz o GHz). Estas dos cifras definen la fidelidad temporal con la que el instrumento puede capturar la realidad eléctrica. Si intentas capturar una señal de 100 MHz con un osciloscopio cuyo ancho de banda es de apenas 50 MHz, verás en la pantalla una versión suavizada, atenuada y fundamentalmente mentirosa de lo que realmente está ocurriendo en el circuito. Los componentes de alta frecuencia de la señal- los flancos agudos, los glitches rápidos, las oscilaciones parásitas- serán filtrados por las limitaciones del instrumento, exactamente igual a cuando usas un ADC demasiado lento para capturar eventos transitorios rápidos en tus sensores de vibración o acelerómetros. La señal que ves es una ilusión filtrada, y cualquier decisión de diseño que tomes basándote en ella será defectuosa.

1.2.2 La Anatomía de la Observación: Escalas y Sincronización

Un osciloscopio tiene tres grupos de controles que debes dominar con la misma soltura con la que ajustas los hiperparámetros de tu red neuronal. Pero a diferencia de los hiperparámetros de software, equivocarte aquí no te costará solo tiempo de entrenamiento; te costará quemar componentes, perder señales únicas, o peor aún, tomar decisiones de diseño basadas en información incorrecta. El primero de estos controles es la **escala vertical (Volts/Div)**, que controla cuántos voltios representa cada división vertical de la cuadrícula en la pantalla. Si tu señal tiene una amplitud de 5 voltios pico a pico y configuras el osciloscopio en 10 V/div, la forma de onda quedará aplastada en una pequeña fracción de la pantalla, perdiendo todo detalle fino de su morfología. Por otro lado, si configuras 500 mV/div para esa misma señal de 5 voltios, la traza se saldrá literalmente de los límites visibles, y solo verás fragmentos incomprensibles en los bordes superior e inferior. Este ajuste es conceptualmente análogo a la normalización de features en machine learning: si no escalas tus datos correctamente antes de entrenar, características importantes quedan comprimidas en rangos numéricos donde el gradiente descendente no puede detectarlas, mientras que otras explotan a valores tan grandes que dominan artificialmente la función de pérdida.

El segundo control fundamental es la **escala horizontal (Time/Div)**, que determina cuánto tiempo cronológico abarca cada división horizontal de la pantalla. Aquí entra en juego tu capacidad de pensar en órdenes de magnitud temporales. Una señal de reloj de 1 kHz tiene un periodo de 1 milisegundo. Si configuras el osciloscopio en 10 ms/div, verás aproximadamente 10 ciclos completos de la onda desplegados de izquierda a derecha, lo cual es útil para observar modulaciones lentas o variaciones en el duty cycle. Pero si ajustas la base de tiempo a 100 ns/div (nanosegundos por división), estarás haciendo un zoom temporal tan extremo que verás solo una fracción microscópica de un único ciclo, revelando detalles del flanco de subida que serían invisibles en la escala de milisegundos. Esta es tu «ventana de observación» temporal, y elegirla incorrectamente es como intentar detectar anomalías en una serie temporal de producción industrial mirando solo las últimas tres muestras o promediando datos de todo un año en un solo punto.

El tercero y más sutil de los controles es el **sistema de disparo (trigger)**, que es probablemente el concepto que separa a un usuario novato de un ingeniero competente con el osciloscopio. Sin un trigger configurado correctamente, la pantalla mostraría una señal que «corre» continuamente de derecha a izquierda, renovándose tan rápido que el ojo humano solo vería un borrón incomprensible y tembloroso. El trigger le dice al osciloscopio: «congela la adquisición de datos y dibuja la forma de onda en la pantalla

cuando detectes que la señal cruza X voltios con pendiente ascendente». Este punto de referencia estabiliza la imagen, permitiéndote observar patrones repetitivos como si estuvieran congelados en el tiempo, o capturar eventos únicos y transitorios en modo single-shot, donde el instrumento espera pacientemente-a veces durante horas-hasta que ocurra exactamente el fenómeno que quieres capturar. El trigger es lo que convierte al osciloscopio en una trampa para electrones fugaces.

1.2.3 El Teorema de Nyquist en Acción: Cuando el Instrumento Miente

Aquí es donde la teoría abstracta de procesamiento de señales-que probablemente estudiaste en un curso previo con transformadas de Fourier en la pizarra-se vuelve visceral y dolorosamente tangible. El Teorema de Nyquist-Shannon establece que para reconstruir fielmente una señal, debes muestrearla a una tasa de al menos el doble de su componente de frecuencia más alta. Esta no es una sugerencia pedagógica; es un límite matemático fundamental de la teoría de la información. Cuando lo violas, no obtienes una versión degradada de la señal original. Obtienes una señal completamente diferente, una ilusión matemática llamada **aliasing**, que es indistinguible de la realidad si no conoces la verdad subyacente.

Por ejemplo, cuando intentas capturar una señal senoidal de 90 MHz usando un osciloscopio digital cuya tasa de muestreo es de 100 MS/s- apenas 1.1 veces la frecuencia de la señal, muy por debajo del mínimo teórico de Nyquist-el osciloscopio solo captura muestras discretas y al conectar esos puntos, reconstruye erróneamente una señal aparente de solo 10 MHz. Esta señal falsa-el alias-es lo que verías en la pantalla. No hay advertencia, no hay mensaje de error. El instrumento te miente con total confianza, y si no comprendes el fenómeno, tomarás decisiones de diseño basadas en datos ficticios.

Este fenómeno no es una curiosidad académica confinada a los laboratorios de electrónica. Es exactamente el mismo error que cometes cuando diseñas un sistema de adquisición de datos para IA y decides muestrear acelerómetros a 100 Hz esperando capturar vibraciones de maquinaria que ocurren a 80 Hz. Los componentes armónicos de esa vibración-que pueden estar en rangos de kilohertz-se plegarán (folding es el término técnico) hacia frecuencias más bajas, contaminando tu espectro de frecuencias con artefactos fantasma que tu modelo de machine learning aprenderá como si fueran reales. Estarás entrenando un detector de anomalías que responde a frecuencias que nunca existieron.

1.2.4 Los Asesinos Silenciosos de Datos: Glitches y Jitter

El osciloscopio moderno viene equipado con capacidades de medición automática que habrían parecido magia hace tres décadas. Pulsa un botón y te despliega la frecuencia de la señal, su periodo, el tiempo de subida de los flancos, el ancho de pulso, el duty cycle-métricas que antes requerían posicionar manualmente dos cursores en la pantalla y calcular diferencias a mano. Pero estas mediciones estadísticas, por útiles que sean, pueden ocultar dos fenómenos letales que solo la observación directa de la forma de onda puede revelarte, y que destruirán silenciosamente la integridad de tus datos de IA sin dejar rastro obvio en tus archivos CSV.

El primero de estos fenómenos son los **glitches**, pulsos espurios de duración microscópica-típicamente en el rango de nanosegundos-que aparecen y desaparecen tan rápido que un multímetro los promedia aritméticamente como si no existieran, y un ADC puede o no capturarlos dependiendo del timing exacto de su reloj de muestreo. Un glitch de 50 nanosegundos en la línea de alimentación de un sensor puede causar que un ADC de 12 bits entregue una lectura completamente errónea-quizás saturada a su valor máximo-durante esa muestra particular, sin que tu código de Python tenga la menor sospecha de que algo salió mal. Si ese sensor está montado en un brazo robótico industrial y el glitch ocurre justo cuando el sistema de control está decidiendo si cerrar o abrir las pinzas, el resultado puede ser una pieza dañada o un accidente. Pero más insidioso aún: si estás recolectando datos de entrenamiento para un modelo de IA y estos glitches ocurren aleatoriamente, tu dataset contendrá valores atípicos (outliers) que no representan el fenómeno físico que intentas modelar. Estos outliers contaminarán las estadísticas de normalización de tus datos, sesgarán las métricas de validación, y en el peor caso, el modelo aprenderá a replicar estos errores como si fueran características válidas del sistema.

El segundo fenómeno es el **jitter**, que es más sutil y filosóficamente más perturbador porque no se trata de valores incorrectos sino de tiempo incierto. El jitter es la variación aleatoria en el timing de eventos que deberían ser periódicos. Si tu señal de reloj de muestreo tiene jitter, estás capturando datos en momentos impredecibles que varían microsegundo a microsegundo respecto a donde deberían estar. Esto introduce ruido temporal-una forma de corrupción que ninguna técnica de denoising espacial puede limpiar, porque el error no está en la magnitud de la medición sino en su ubicación en el eje del tiempo. Para sistemas de adquisición de datos que luego alimentan modelos de series temporales o redes neuronales recurrentes, el jitter destruye la coherencia temporal que el modelo necesita para aprender dependencias causales correctas. Es como intentar aprender a bailar con música cuyo tempo fluctúa aleatoriamente: nunca sincronizarás correctamente.

En una señal de reloj digital real, el ideal platónico sería una onda cuadrada perfecta con flancos infinitamente agudos y periodos exactamente iguales. La realidad muestra flancos que varían su posición temporal debido al jitter, y picos espurios-glitches-que aparecen donde no deberían existir pulsos. Los glitches pueden resetear flip-flops en sistemas digitales, causar lecturas erróneas en ADCs, o hacer que un protocolo de comunicación interprete ruido como datos válidos. El jitter, por su parte, acumula incertidumbre temporal que degrada progresivamente la calidad de sincronización en sistemas complejos.

Ambos fenómenos son completamente invisibles para un multímetro estándar, que solo te dirá el valor RMS promediado en el tiempo. Pero para sistemas digitales-y eso incluye cualquier sistema de adquisición de datos moderno-son potencialmente letales. Un ingeniero que no use el osciloscopio para validar la calidad temporal de sus señales está volando a ciegas.

1.2.5 Más Allá del Trigger Básico: Cazando Anomalías

El modo de disparo básico-edge trigger, que captura cuando la señal cruza un umbral de voltaje con pendiente ascendente o descendente-funciona perfectamente para señales periódicas y bien comportadas. Pero el mundo real, especialmente en sistemas industriales, robótica y sensores para IA, está lleno de eventos únicos, patológicos y profundamente anómalos que solo ocurren una vez cada mil millones de ciclos. Para capturar estos fantasmas eléctricos, los osciloscopios modernos implementan modos de disparo avanzados que transforman el instrumento de un simple visualizador en un detective forense especializado.

El **Pulse Width Trigger** dispara la captura solo cuando detecta un pulso cuya duración es más ancha o más angosta que un rango especificado. Esto es perfecto para detectar timeouts en protocolos de comunicación, donde un dispositivo que debería responder en 10 microsegundos de repente tarda 50 debido a un bug de firmware, o para capturar el momento exacto en que un circuito PWM mal diseñado genera un pulso anormalmente corto que no logra activar la carga. El **Runt Trigger** es aún más especializado: captura pulsos que no alcanzan su amplitud lógica completa, quedando atrapados en un estado intermedio inválido. Estos runts son síntomas de problemas de integridad de señal-quizás causados por reflexiones en líneas de transmisión mal terminadas-o de transitorios en la alimentación que colapsan momentáneamente el voltaje durante una transición lógica.

Pero el más sofisticado de todos es el **Setup/Hold Violation Trigger**, diseñado para capturar errores de timing en sistemas digitales síncronos. En estos sistemas, los datos deben estar estables durante una ventana temporal crítica: un tiempo mínimo antes del flanco activo del reloj (setup time) y un tiempo mínimo después (hold time). Violar cualquiera de estas

restricciones-aunque sea por nanosegundos-cause que flip-flops y registros capturen valores incorrectos, generando errores aleatorios e intermitentes que son imposibles de debuggear con métodos de software porque el código ejecutándose en el procesador es correcto; el problema está en el sustrato físico de timing sobre el cual corre ese código. Un osciloscopio con este trigger puede permanecer conectado durante horas o días, esperando pacientemente a que ocurra la violación de timing de un nanosegundo que está causando que tu sistema embebido se cuelgue una vez por semana sin patrón aparente.

1.2.6 Decodificación de Protocolos: Del Ruido al Significado

Los osciloscopios modernos de gama media y alta incluyen capacidades de decodificación de protocolos seriales en tiempo real, y esto transforma radicalmente la experiencia de debugging. En lugar de observar pulsos incomprensibles en las líneas SDA y SCL de un bus I2C, el osciloscopio te muestra directamente los bytes transmitidos con sus valores hexadecimales superpuestos en la pantalla, identifica las direcciones de los dispositivos, marca los bits de ACK/NACK, y hasta detecta violaciones de protocolo como condiciones de start sin el correspondiente stop. Cuando tu sensor ESP32 se niega tercamente a responder por I2C y tu código de Arduino reporta genéricamente «Error de comunicación», el osciloscopio te dirá con precisión forense si el problema es un bit de ACK perdido porque el esclavo no está presente en esa dirección, un glitch en la línea SDA que corrompe los datos, o jitter en SCL que viola las especificaciones de timing del protocolo.

Esta capacidad no solo acelera el debugging; cambia cualitativamente cómo piensas sobre los sistemas embebidos. En lugar de tratar la comunicación como una caja negra que o funciona o no funciona, puedes observar la conversación bit por bit entre microcontrolador y sensor, viendo exactamente dónde se rompe el diálogo. Para aplicaciones de IA que dependen de sensores externos-acelerómetros, giroscopios, sensores de temperatura, cámaras-esta visibilidad es la diferencia entre pasar semanas adivinando por qué obtienes lecturas corruptas ocasionales versus identificar en una tarde que el problema es un capacitor de desacoplamiento faltante que permite glitches de 200 mV en la línea de datos durante las transiciones.

Como ves, el osciloscopio no es simplemente un voltímetro que grafica voltaje versus tiempo. Es una herramienta forense que revela la «salud» temporal de tus señales, detecta anomalías que otros instrumentos promedian hasta la invisibilidad, y te permite validar que los datos que alimentan a tus sistemas de IA no están contaminados desde el origen. En el mundo del machine learning, donde solemos asumir ingenuamente que los datos en nuestros ar-

chivos CSV son limpios, precisos y bien formados, el osciloscopio es tu primera línea de defensa contra la realidad ruidosa, glitchy y profundamente caótica del mundo analógico. Porque un modelo de IA entrenado con datos corruptos no aprende el mundo real; aprende los artefactos de tus instrumentos de medición. Y eso, en ingeniería, no es inteligencia artificial. Es amplificación de errores.

1.3 Fuentes de Alimentación: El Corazón Invisible de tus Sistemas

Objetivo de Aprendizaje

Objetivo: **Caracterizar** el comportamiento de fuentes de alimentación lineales y conmutadas **mediante** mediciones de ripple, regulación de línea y carga, **para** garantizar la estabilidad energética de sistemas de adquisición de datos y prevenir fallas silenciosas en aplicaciones de IA que procesan señales analógicas sensibles.

Ingeniería de Datos desde el Hardware

Tu GPU de entrenamiento puede costar \$3000 dólares, tu arquitectura Transformer puede tener mil millones de parámetros, pero si la fuente de alimentación de tu sistema de sensores introduce 200 mV de ripple en las líneas analógicas, tus datos de entrada serán basura. La IA no detecta que el problema está en los voltios; simplemente aprende el ruido como si fuera señal.

En los cursos introductorios de electrónica, las fuentes de alimentación se representan como símbolos ideales: círculos perfectos con signos de más y menos, entregando voltajes constantes e inmutables como si fueran leyes de la física. Pero en el mundo real, y especialmente en sistemas de adquisición de datos para inteligencia artificial-, las fuentes de alimentación son componentes activos, ruidosos y profundamente imperfectos que pueden ser la diferencia entre un sistema que funciona con precisión milimétrica y uno que genera datos corruptos que destruyen silenciosamente la validez de tus modelos entrenados. Porque aquí está la verdad incómoda: todos tus circuitos analógicos, todos tus ADCs, todos tus amplificadores operacionales y sensores, usan la línea de alimentación como referencia implícita. Si esa referencia vibra, se distorsiona o colapsa bajo carga, toda tu cadena de medición se contamina desde la raíz.

1.3.1 Lineal vs. Conmutada: El Compromiso Fundamental

Existen dos filosofías antagónicas para construir fuentes de alimentación, y cada una representa un conjunto diferente de compromisos ingenieriles que afectarán directamente la calidad de tus datos. Las **fuentes lineales** son conceptualmente simples y brutalmente ineficientes: toman el voltaje de entrada (típicamente de un transformador conectado a la red eléctrica), lo rectifican a corriente continua, y luego usan un transistor de potencia operando en su región activa como una resistencia variable controlada que «quemar» el exceso de voltaje en forma de calor. Si necesitas 5 voltios y la entrada rectificada es de 12 voltios, el transistor regulador disipa esos 7 voltios sobrantes como si fuera una estufa. La eficiencia energética es terrible—típicamente entre 30 % y 60 %—pero a cambio obtienes algo precioso: silencio eléctrico. Una fuente lineal bien diseñada produce voltaje DC casi puro, con niveles de ruido en el rango de microvoltios, porque no hay conmutación de alta frecuencia que genere armónicos parásitos.

Las **fuentes conmutadas** (SMPS, por sus siglas en inglés: Switched-Mode Power Supply) adoptaron la filosofía opuesta. En lugar de quemar el exceso de voltaje, lo convierten eficientemente mediante conmutación rápida—típicamente a frecuencias entre 50 kHz y varios MHz—usando transistores que operan solo en dos estados: completamente encendidos (saturación) o completamente apagados (corte). En estos estados extremos, los transistores disipan muy poca potencia, logrando eficiencias energéticas del 80 % al 95 %. Esto las hace ideales para aplicaciones de alta potencia como alimentar GPUs, servidores, o cargar baterías. Pero esa conmutación rápida tiene un precio: generan **ripple** (ondulación residual de voltaje) y ruido electromagnético en frecuencias que pueden acoplarse a tus circuitos analógicos, contaminando señales débiles que intentas medir. Para sistemas de IA que procesan audio, bioseñales o vibraciones de baja amplitud, este ripple puede ser letal.

1.3.2 Ripple: La Ondulación que Envenena tus ADCs

El ripple es la ondulación residual de voltaje que permanece en la salida de una fuente después de la rectificación y filtrado. En fuentes lineales, el ripple proviene principalmente de la frecuencia de la red eléctrica (60 Hz en México, 50 Hz en Europa) y sus armónicos, y puede ser reducido agresivamente con capacitores de filtrado grandes y reguladores de baja caída (LDO). En fuentes conmutadas, el ripple tiene dos componentes: uno a la frecuencia de conmutación (decenas o cientos de kHz) y otro a frecuencias más bajas debido a la modulación del ciclo de trabajo del controlador PWM. Ambos componentes son peligrosos, pero por razones diferentes.

El ripple de baja frecuencia (50-120 Hz) es particularmente insidioso en aplicaciones de IA que procesan señales biomédicas como electrocardiogramas o electromiogramas, porque cae exactamente en el rango de frecuencias de interés biológico. Si tu fuente introduce 100 mV de ripple a 60 Hz en la línea de alimentación de un amplificador de instrumentación conectado a electrodos ECG, ese ripple se sumará directamente a la señal cardíaca que intentas medir (que típicamente es de solo 1-2 mV). Tu ADC digitalizará fielmente esta mezcla contaminada, y tu modelo de machine learning para detectar arritmias aprenderá que todos los pacientes tienen un componente sinusoidal de 60 Hz en su corazón-lo cual es falso, es solo un artefacto de tu fuente barata.

El ripple de alta frecuencia, por otro lado, puede ser filtrado más fácilmente con componentes pasivos (inductores y capacitores), pero tiene una propiedad más sutil y peligrosa: puede modular no linealmente con las señales que estás midiendo. Si tu fuente conmutada opera a 100 kHz y estás muestreando un acelerómetro a 10 kHz, productos de intermodulación pueden generar frecuencias espurias que aparecen dentro de tu banda de interés. Esto es aliasing inyectado desde la fuente de alimentación, y es casi imposible de eliminar por software porque está mezclado en el dominio analógico antes de la digitalización.

La Figura 1.4 muestra la diferencia brutal entre el ripple de una fuente lineal de calidad versus una fuente conmutada económica midiendo la salida con un osciloscopio configurado en AC coupling para amplificar las variaciones.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1.3_e1.py`. Observa las amplitudes del ripple en ambas fuentes.

Experimento: Modifica la variable `amplitud_ripple_switching` de 0.15 V a 0.5 V, simulando una fuente conmutada con filtrado deficiente. Si tu ADC tiene 12 bits de resolución y un rango de 0-3.3V, ¿cuántos bits efectivos pierdes debido a este ripple? Calcula el ENOB (Effective Number of Bits).

1.3.3 Regulación de Línea y Carga: Cuando el Voltaje No es Constante

Más allá del ripple, las fuentes reales sufren de dos formas de inestabilidad que rara vez se enseñan explícitamente pero que destrazan sistemas en producción. La **regulación de línea** mide cuánto cambia el voltaje de salida cuando el voltaje de entrada varía. En México, el voltaje de la red eléctrica nominalmente es 127 V, pero en la realidad oscila entre 110 V y 135 V dependiendo de la hora del día, la carga en el transformador del barrio, y

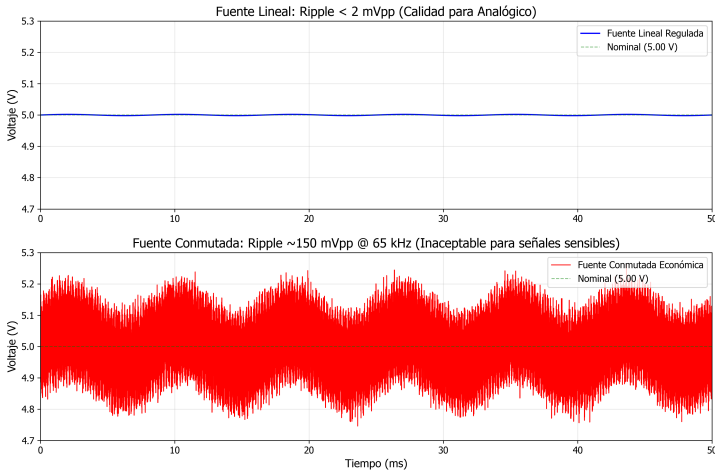


Figura 1.4: Comparación de ripple en fuentes de alimentación. Superior: Fuente lineal regulada con ripple de 2 mVpp a 120 Hz. Inferior: Fuente conmutada económica con ripple de 150 mVpp a 65 kHz más componentes de baja frecuencia. Para un sistema de adquisición de bioseñales, la diferencia es la línea entre datos utilizables y basura electrónica. (Figura generada con el script `s1_3_e1.py`)

eventos transitorios como el arranque de aires acondicionados industriales. Una fuente con pobre regulación de línea puede variar su salida en 200-300 mV cuando la entrada fluctúa, y si esa salida alimenta la referencia de voltaje de tu ADC, todas tus mediciones absolutas estarán contaminadas con esta deriva. Tu modelo de IA que intenta detectar cambios sutiles de temperatura de 0.1°C en un proceso industrial estará viendo variaciones fantasma causadas por la inestabilidad de la red eléctrica.

La **regulación de carga** es aún más traicionera porque ocurre dinámicamente durante la operación. Mide cuánto cae el voltaje de salida cuando la corriente demandada aumenta. Imagina un sistema embebido que alimenta simultáneamente un microcontrolador ESP32, un sensor de distancia ultrasónico, y un módulo de comunicación WiFi. Durante la transmisión WiFi, el módulo de radio puede demandar picos de corriente de 300-500 mA durante microsegundos. Si la fuente de alimentación tiene alta impedancia de salida o capacitores de desacoplamiento insuficientes, el voltaje colapsará momentáneamente-quizás en 400-600 mV-durante esos picos. Este colapso puede causar que el ADC integrado del ESP32 entregue lecturas erróneas justo en el momento en que el sistema está muestreando sensores para transmitir, sin que tu código tenga la menor indicación de que algo salió mal.

La Figura 1.5 simula este comportamiento mostrando cómo el voltaje de salida se degrada a medida que aumenta la corriente demandada en una fuente con regulación pobre versus una bien diseñada.

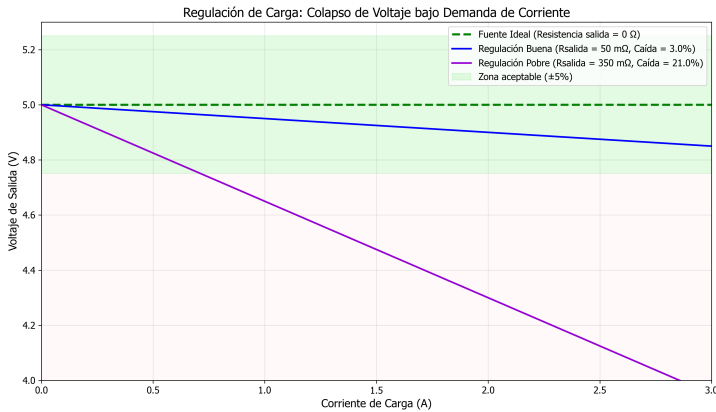


Figura 1.5: Regulación de carga en fuentes reales. La fuente ideal mantiene 5.00 V independiente de la corriente. Una fuente con regulación pobre (línea roja) colapsa hasta 4.3 V bajo carga máxima, una caída del 14 % que viola las especificaciones de casi todos los circuitos digitales de 5V. (Figura generada con el script `s1_3_e2.py`)

Reto Computacional: Experimenta con el Script

En el script `s1_3_e2.py`, modifica el valor de `resistencia_salida_mala` a diferentes valores.

Pregunta: ¿Cuál es la resistencia de salida máxima tolerable si necesitas mantener el voltaje dentro de $\pm 2\%$ bajo una carga de 2A? Si tu sistema de IA depende de sensores alimentados por esta fuente, ¿qué tipo de errores aparecerán en tus datos cuando la carga varíe dinámicamente?

1.3.4 El PSRR Olvidado: Tu Última Línea de Defensa

Existe una especificación crítica de amplificadores operacionales y reguladores de voltaje que la mayoría de los estudiantes ignora hasta que un sistema falla misteriosamente en producción: el **PSRR** (Power Supply Rejection Ratio, o Razón de Rechazo de Fuente de Alimentación). El PSRR mide cuánto del ruido presente en las líneas de alimentación logra acoplarse a la salida del dispositivo. Se expresa típicamente en decibeles: un PSRR de 80 dB significa que el ruido en la alimentación es atenuado 10,000 veces antes de aparecer en la salida. Esto suena impresionante, pero hay una

trampa brutal: el PSRR degrada dramáticamente con la frecuencia.

Un amplificador operacional estándar puede tener PSRR de 100 dB a frecuencias de DC y baja frecuencia, pero a 100 kHz-exactamente donde operan las fuentes conmutadas-ese PSRR puede haber caído a apenas 40 dB, que es una atenuación de solo 100 veces. Si tu fuente conmutada tiene 200 mV de ripple a 100 kHz, hasta 2 mV de ese ripple aparecerán directamente en la salida de tu amplificador, sin importar qué tan cuidadosamente diseñaste el circuito de señal. Para un sistema que intenta medir señales de termopares del orden de decenas de microvoltios, esos 2 mV de ripple residual son un desastre total.

La solución no es trivial. Requiere una estrategia de «defensa en capas»: usar fuentes lineales para etapas analógicas críticas, agregar reguladores LDO locales con alto PSRR cerca de cada circuito sensible, implementar filtrado LC agresivo entre la fuente conmutada principal y los subsistemas analógicos, y diseñar planos de tierra y alimentación en el PCB que minimicen los loops de corriente que actúan como antenas captando el ruido electromagnético de la conmutación. Cada capa de defensa atenúa el problema, pero ninguna sola técnica es suficiente.

1.3.5 Transitorios de Encendido: El Caos de los Primeros Milisegundos

Existe un fenómeno que destruye componentes y corrompe estados iniciales pero que rara vez se simula en los laboratorios académicos: los **transitorios de encendido**. Cuando conectas una fuente de alimentación, no salta instantáneamente de 0V a su valor nominal. Hay un periodo de rampa que puede durar desde microsegundos hasta milisegundos, y durante ese tiempo, el voltaje puede oscilar, sobrepasar (overshoot) su valor final, o comportarse de formas caóticas dependiendo del diseño del regulador y la impedancia de la carga. Circuitos digitales mal diseñados pueden interpretar estos transitorios como señales de reloj espurias, causando que microcontroladores ejecuten instrucciones aleatorias antes de que el voltaje se estabilice. ADCs pueden capturar valores saturados o indefinidos que corrompen sus registros de calibración interna.

Peor aún, si tienes múltiples voltajes en tu sistema (por ejemplo, 12V para motores, 5V para lógica digital, 3.3V para microcontroladores, y 1.8V para circuitos analógicos de precisión), el orden y timing en que estos voltajes alcanzan sus valores nominales es crítico. Si el voltaje de E/S (I/O) de un microcontrolador sube antes que su voltaje de núcleo (core), puedes inyectar corriente inversa a través de los diodos de protección internos del chip, causando «latch-up»-un estado catastrófico donde corrientes parásitas activan estructuras tiristor parasitarias dentro del silicio, llevando el consumo de corriente a niveles destructivos que pueden quemar literalmente el

circuito integrado.

La Figura 1.6 muestra una simulación de transitorios de encendido en una fuente con pobre control de overshoot, comparada con una fuente con «soft-start» que limita la velocidad de subida del voltaje.

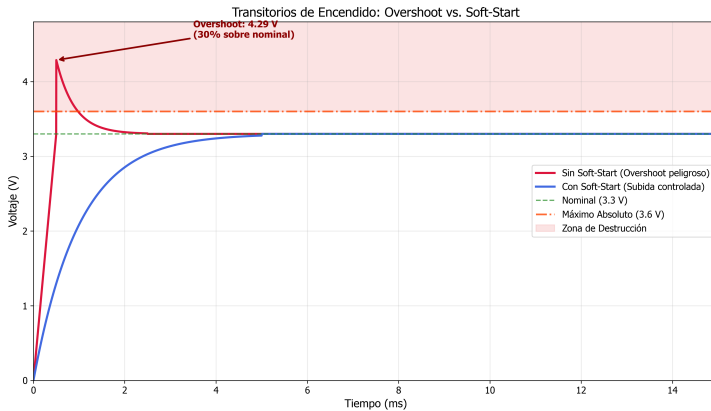


Figura 1.6: Transitorios de encendido en fuentes de alimentación. La fuente sin soft-start (roja) presenta un overshoot del 30 % que puede exceder los límites absolutos de componentes sensibles. La fuente con soft-start (azul) sube de forma controlada, protegiendo la carga. (Figura generada con el script `s1_3_e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_3_e3.py`. Modifica la variable `overshoot_percent` a diferentes valores.

Análisis: Si tu microcontrolador tiene un voltaje máximo absoluto de 3.6V y la fuente nominal es 3.3V, ¿cuánto overshoot porcentual puedes tolerar antes de entrar en la zona de destrucción? ¿Qué componente agregarías al circuito para proteger contra estos transitorios?

1.3.6 Medición de Calidad de Fuentes: Más Allá del Multímetro

Medir una fuente de alimentación con un multímetro es como evaluar la salud de un atleta tomándole una fotografía: obtienes un número estático que oculta toda la dinámica relevante. Para caracterizar realmente una fuente, necesitas el osciloscopio configurado correctamente. Primero, mide en **AC coupling** para amplificar el ripple sin que el componente DC sature la pantalla. Configura una base de tiempo lo suficientemente lenta para

capturar varios ciclos del ripple de baja frecuencia (10 ms/div para ver 60 Hz), pero también realiza capturas rápidas en modo single-shot para detectar transitorios de alta frecuencia. Segundo, mide bajo condiciones de carga realistas usando una **carga electrónica** o al menos resistencias de potencia adecuadas-nunca caracterices una fuente en vacío y asumas que se comportará igual bajo carga.

Tercero, y esto es crítico: mide en el punto de uso, no en los terminales de salida de la fuente. Los cables de alimentación, las pistas del PCB, y los conectores tienen resistencia e inductancia parásitas que causan caídas de voltaje adicionales proporcionales a la corriente. Un sistema que mide 5.00V en la fuente puede tener solo 4.7V en el microcontrolador ubicado a 30 cm de distancia si los cables son delgados y la corriente es alta. Esta caída es dinámica: cuando la carga pulsa corriente, el voltaje en el punto de uso colapsa momentáneamente, y ninguna cantidad de regulación en la fuente puede compensar la inductancia del cableado que está entre la fuente y la carga.

La fuente de alimentación es el corazón de tu sistema, y como todo corazón, puede latir de forma irregular, debilitarse bajo estrés, o colapsar completamente en el momento más crítico. Un modelo de IA entrenado con datos contaminados por fuentes defectuosas no aprende a detectar fallos en maquinaria; aprende a detectar los transitorios de tu fuente conmutada barata. Y eso, en un contexto industrial donde las decisiones basadas en IA pueden costar millones de dólares en paros de producción, no es un detalle técnico menor. Es la diferencia entre ingeniería competente e incompetencia disfrazada de machine learning.

1.4 Componentes Reactivos: Cuando la Frecuencia Importa

Objetivo de Aprendizaje

Objetivo: Caracterizar el comportamiento dependiente de la frecuencia de resistencias, inductores y capacitores **mediante** mediciones de impedancia y análisis de respuesta transitoria, **para** diseñar filtros anti-aliasing y circuitos de acondicionamiento de señales en sistemas de adquisición de datos para IA.

Ingeniería de Datos desde el Hardware

Cada sensor que conectas a tu sistema de IA tiene capacitancia parásita. Cada cable que usas tiene inductancia. Cuando muestreas a 100 kHz y tu cableado forma un filtro pasa-bajas no intencional con frecuencia de corte a 20 kHz, estás digitalizando una versión suavizada y retardada de la realidad. Tu modelo de IA aprenderá correlaciones temporales falsas porque el hardware está distorsionando la causalidad física de los eventos.

En los circuitos de corriente continua, el mundo es simple y estático. Una resistencia de 1 k Ω opone exactamente 1 k Ω al flujo de corriente, sin importar si mides con una batería o con un generador de señales. Pero cuando introduces señales que varían con el tiempo - y en sistemas de adquisición de datos para IA, todas las señales interesantes varían con el tiempo - el comportamiento de los componentes se vuelve radicalmente dependiente de la frecuencia. Los capacitores, que bloquean completamente la corriente continua, se convierten en cortocircuitos a altas frecuencias. Los inductores, que son simples cables enrollados para señales lentas, se transforman en barreras impenetrables para transitorios rápidos. Esta dependencia de frecuencia no es un detalle técnico esotérico; es la razón por la cual tus datos de alta velocidad llegan distorsionados, retardados y contaminados con reflexiones que ningún algoritmo de machine learning puede compensar correctamente.

1.4.1 La Resistencia: El Componente que No Discrimina

Las resistencias son los únicos componentes pasivos verdaderamente democráticos: oponen la misma resistencia a todas las frecuencias, desde corriente continua hasta gigahertz. Esta propiedad aparentemente trivial las hace invaluableles como referencias de impedancia conocida en circuitos de medición. Cuando usas una resistencia de precisión de 10 k Ω como carga de un sensor, esperas que esos 10 k Ω sean exactamente 10 k Ω sin importar si el sensor está midiendo temperatura que varía lentamente a milihertz o vibraciones de maquinaria a kilohertz. Pero incluso esta simplicidad tiene trampas ocultas. Las resistencias reales - especialmente las de potencia o las montadas en circuitos impresos con pistas largas - tienen inductancia y capacitancia parásitas que se vuelven significativas a altas frecuencias. Una resistencia de 1 k Ω puede comportarse como si tuviera 10 nH de inductancia serie y 1 pF de capacitancia en paralelo, y aunque estos valores parecen microscópicos, a 100 MHz esa inductancia genera una reactancia adicional de 6 Ω que contamina tus mediciones.

La potencia disipada en resistencias sigue la famosa ecuación $P = I^2 R =$

V^2/R , y esta disipación es puramente térmica (la energía eléctrica se convierte irreversiblemente en calor). Esta conversión unidireccional es lo que diferencia fundamentalmente a las resistencias de los componentes reactivos. Un capacitor o inductor almacena energía temporalmente y la devuelve al circuito; una resistencia la destruye. En sistemas de adquisición de datos de alta precisión, este calentamiento puede ser un problema silencioso: una resistencia de muestreo de corriente de $0.1\ \Omega$ disipando $2\ \text{W}$ puede calentarse 40°C sobre la temperatura ambiente, y ese gradiente térmico introduce efectos termoelectrónicos que generan voltajes parásitos del orden de microvoltios a milivoltios, contaminando señales de sensores de baja amplitud.

1.4.2 El Capacitor: Memoria de Voltaje y Filtro de Frecuencia

Un capacitor es, en esencia, un dispositivo que se opone a cambios bruscos de voltaje. Su impedancia es inversamente proporcional a la frecuencia: $Z_C = 1/(j\omega C)$, donde $\omega = 2\pi f$. Esto significa que a frecuencias bajas, un capacitor presenta alta impedancia y bloquea el paso de corriente; a frecuencias altas, su impedancia colapsa y se comporta casi como un cortocircuito. Esta propiedad lo convierte en el componente fundamental para construir filtros que separan señales por frecuencia, y en sistemas de adquisición de datos, los filtros son la línea de defensa crítica contra el aliasing - ese fenómeno letal donde frecuencias por encima de la tasa de Nyquist se «pliegan» hacia abajo y contaminan tu espectro de interés.

Considera un sistema que muestrea acelerómetros a $10\ \text{kHz}$ para detectar vibraciones de maquinaria industrial. Según Nyquist, solo puedes capturar fielmente frecuencias hasta $5\ \text{kHz}$. Pero el mundo real está lleno de vibraciones de alta frecuencia-resonancias a $30\ \text{kHz}$, ruido electromagnético a megahertz-que, sin filtrado, serán capturadas por el ADC y aparecerán falsamente como frecuencias bajas. La solución es un filtro anti-aliasing RC pasa-bajas antes del ADC, típicamente con frecuencia de corte entre $3\text{-}4\ \text{kHz}$, que atenúa agresivamente todo lo que está por encima. Pero diseñar este filtro requiere entender profundamente la respuesta en frecuencia de capacitores reales, porque los capacitores tienen algo llamado ESR (Equivalent Series Resistance) y ESL (Equivalent Series Inductance) que degrada su comportamiento a altas frecuencias.

La Figura 1.7 muestra cómo la impedancia de un capacitor real diverge del modelo ideal a medida que la frecuencia aumenta, debido a parásitos internos.

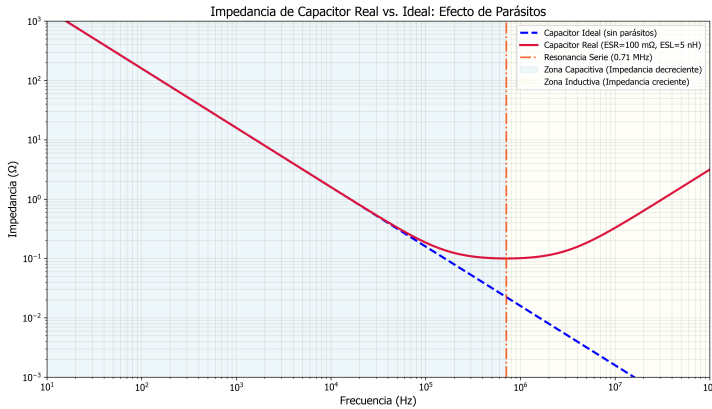


Figura 1.7: Impedancia de un capacitor de 10 μF cerámico (real vs. ideal) en función de la frecuencia. El capacitor ideal muestra impedancia decreciente monótonicamente. El capacitor real exhibe resonancia serie a 1 MHz debido a la inductancia parásita, más allá de la cual se comporta como un inductor. Para filtrado de alta frecuencia, esta divergencia destruye la efectividad del componente. (Figura generada con el script `s1.4_e1.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1.4_e1.py`. Modifica la variable `ESL` (inductancia serie) de 1 nH a 10 nH, simulando capacitores de diferentes tecnologías.

Pregunta: ¿A qué frecuencia ocurre la resonancia serie? Si estás diseñando un filtro anti-aliasing para un ADC de 1 MS/s, ¿qué tipo de capacitor (cerámico, electrolítico, film) deberías usar y por qué? ¿Cómo afecta la resonancia parásita a la efectividad del filtro?

1.4.3 El Inductor: Inercia Electromagnética y Supresión de Transitorios

Si el capacitor se opone a cambios de voltaje, el inductor se opone a cambios de corriente. Su impedancia es directamente proporcional a la frecuencia: $Z_L = j\omega L$. A corriente continua, un inductor ideal es simplemente un cable-resistencia cercana a cero. Pero cuando intentas cambiar la corriente rápidamente, el inductor genera un voltaje de oposición proporcional a la velocidad del cambio: $V = L dI/dt$. Esta propiedad lo convierte en un supresor natural de transitorios y un componente esencial en fuentes conmutadas, donde se usa para suavizar las corrientes pulsantes y almacenar energía magnética durante los ciclos de conmutación.

En sistemas de adquisición de datos, los inductores aparecen frecuentemente como parásitos no deseados: la inductancia de las pistas del PCB, la inductancia de los cables de conexión, la inductancia mutua entre pistas paralelas. Un cable de alimentación de 10 cm puede tener fácilmente 100 nH de inductancia. Si por ese cable circula una corriente que cambia a razón de 1 A/ μ s-algo común cuando un microcontrolador conmuta su módulo WiFi-el inductor generará un transitorio de voltaje de $V = L dI/dt = 100 \times 10^{-9} \times 10^6 = 0,1$ V, que es 100 mV. Ese pico de 100 mV en la línea de alimentación se inyecta directamente a todos los circuitos analógicos, contaminando lecturas de ADC justo en el momento en que el sistema está transmitiendo datos.

La combinación de inductancia parásita con capacitancia parásita forma circuitos LC resonantes no intencionales que pueden oscilar a frecuencias específicas cuando son excitados por transitorios. Estas oscilaciones - conocidas como «ringing» - aparecen en las formas de onda como sobrepicos amortiguados que el osciloscopio revela pero que tu código de software nunca ve directamente. Sin embargo, si esas oscilaciones caen dentro de la banda de paso de tu filtro anti-aliasing, serán digitalizadas y aparecerán en tus datos como eventos espurios que tu modelo de IA aprenderá como características válidas del sistema.

1.4.4 Filtros RC: La Primera Línea de Defensa Contra el Caos

El filtro RC pasa-bajas es probablemente el circuito más simple y más crítico en sistemas de adquisición de datos. Consiste en una resistencia en serie seguida de un capacitor a tierra, formando un divisor de voltaje cuya atenuación depende de la frecuencia. La frecuencia de corte se define como $f_c = 1/(2\pi RC)$, el punto donde la señal se atenúa a $1/\sqrt{2} \approx 0,707$ de su amplitud original, equivalente a -3 dB. Por encima de esta frecuencia, la atenuación aumenta a razón de 20 dB por década - cada vez que multiplicas la frecuencia por 10, la amplitud se divide por 10.

Pero aquí está la parte que los libros de texto rara vez enfatizan con suficiente urgencia: un filtro de primer orden (un solo RC) tiene una pendiente de atenuación de solo 20 dB/década, que es peligrosamente suave para aplicaciones anti-aliasing. Si tu ADC muestrea a 100 kHz y configuras el filtro con $f_c = 10$ kHz, a la frecuencia de Nyquist (50 kHz), la atenuación será de aproximadamente 14 dB, que es solo un factor de 5. Si hay ruido electromagnético de 100 mV a 50 kHz, todavía pasarán 20 mV al ADC, y esos 20 mV serán indistinguibles de señal válida. Para aplicaciones críticas, necesitas filtros de orden superior - segundo, tercero, cuarto - que proveen pendientes de 40, 60, 80 dB/década, respectivamente, aplastando el ruido de alta frecuencia antes de que llegue al ADC.

La Figura 1.8 compara la respuesta en frecuencia de filtros RC de primer, segundo y tercer orden, mostrando cómo la pendiente de atenuación aumenta dramáticamente con cada etapa adicional.

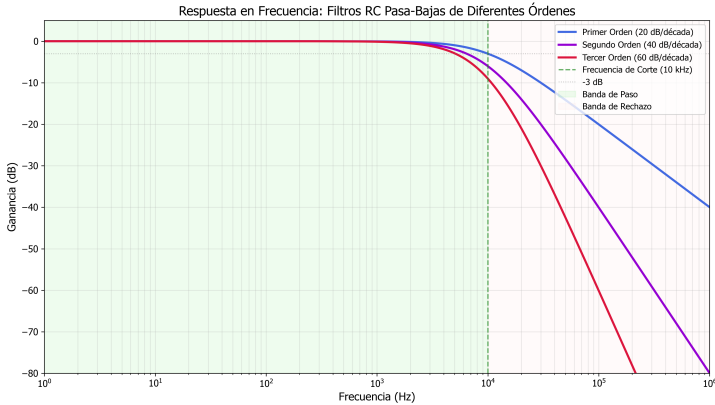


Figura 1.8: Respuesta en frecuencia (diagrama de Bode de magnitud) de filtros pasa-bajas RC de diferentes órdenes, todos con frecuencia de corte nominal de 10 kHz. El filtro de primer orden (azul) atenúa suavemente a 20 dB/década. El segundo orden (violeta) duplica la pendiente a 40 dB/década. El tercer orden (rojo oscuro) alcanza 60 dB/década, proveyendo supresión agresiva de alta frecuencia necesaria para anti-aliasing efectivo. (Figura generada con el script `s1_4_e2.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_4_e2.py`. Observa la diferencia de atenuación a 100 kHz entre los tres órdenes de filtro.

Análisis: Si tu ADC muestrea a 50 kHz (Nyquist a 25 kHz) y hay interferencia electromagnética de 200 mV a 100 kHz, ¿cuánto voltaje de interferencia llegará al ADC con cada tipo de filtro? ¿Cuál es el mínimo orden necesario para atenuar la interferencia por debajo de 1 mV?

1.4.5 Respuesta Transitoria: Cuando Enciendes el Sistema

La respuesta en frecuencia te dice cómo se comporta un circuito en estado estacionario con señales sinusoidales continuas. Pero el mundo real está lleno de eventos transitorios: el momento en que enciendes la alimentación, cuando un sensor cambia bruscamente de estado, cuando un pulso digital transita. La respuesta transitoria describe cómo evoluciona el voltaje o la

corriente en función del tiempo cuando aplicas un cambio escalón. Para un filtro RC, cuando aplicas un voltaje escalón a la entrada, la salida no salta instantáneamente - sube exponencialmente con una constante de tiempo $\tau = RC$, alcanzando el 63 % del valor final en un tiempo τ , el 86 % en 2τ , el 95 % en 3τ , y así sucesivamente.

Este retardo es crítico en sistemas de adquisición de datos multiplexados, donde un solo ADC muestrea múltiples canales secuencialmente. Si cada canal tiene un filtro RC anti-aliasing, cuando el multiplexor cambia del canal 1 al canal 2, el voltaje en el capacitor del filtro del canal 2 debe establecerse al nuevo valor antes de que el ADC tome la muestra. Si la constante de tiempo del filtro es $10\text{ }\mu\text{s}$ y solo esperas $1\text{ }\mu\text{s}$ antes de muestrear, capturarás un valor transitorio que es una mezcla entre el voltaje anterior y el nuevo - un error de settling que corrompe tus datos. La regla práctica es esperar al menos 5τ para que el voltaje se establezca dentro del 1 % del valor final, pero esto limita la velocidad máxima de muestreo multicanal.

La Figura 1.9 muestra la respuesta transitoria de un filtro RC cuando se aplica un escalón de voltaje, comparando diferentes constantes de tiempo.

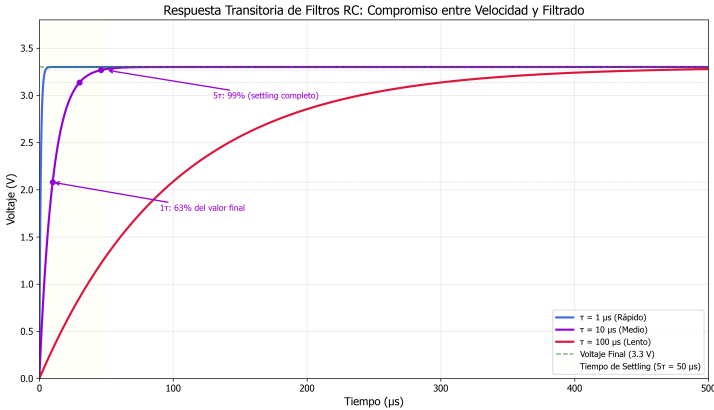


Figura 1.9: Respuesta transitoria de filtros RC con diferentes constantes de tiempo ante un escalón de voltaje de 0 a 3.3V. Filtros con τ pequeño ($1\text{ }\mu\text{s}$, azul) responden rápido pero filtran menos ruido. Filtros con τ grande ($100\text{ }\mu\text{s}$, rojo oscuro) filtran agresivamente pero introducen retardos significativos. El compromiso entre velocidad de respuesta y supresión de ruido es inevitable. (Figura generada con el script `s1_4.e3.py`)

Reto Computacional: Experimenta con el Script

En el script `s1_4_e3.py`, modifica los valores de las constantes de tiempo.

Experimento: Si tienes un ADC multiplexado de 8 canales y cada canal tiene un filtro RC con $\tau = 10 \mu\text{s}$, ¿cuál es la tasa de muestreo máxima por canal si debes esperar 5τ para settling? ¿Cómo cambiaría tu diseño si necesitas muestrear cada canal a 1 kHz mínimo?

1.4.6 Resonancia LC: Cuando Dos Reactivos Combaten

Cuando conectas un inductor y un capacitor en serie o en paralelo, formas un circuito resonante que tiene una frecuencia natural de oscilación: $f_0 = 1/(2\pi\sqrt{LC})$. A esta frecuencia, la reactancia inductiva ωL y la reactancia capacitiva $1/(\omega C)$ se cancelan exactamente, y el circuito exhibe comportamientos extremos dependiendo de la configuración. En serie, la impedancia cae a un mínimo (limitada solo por resistencias parásitas), convirtiéndose en un cortocircuito resonante. En paralelo, la impedancia sube a un máximo, actuando como un circuito abierto.

Esta resonancia no es solo una curiosidad académica; aparece constantemente en sistemas reales como parásito destructivo. La inductancia de las pistas de un PCB combinada con la capacitancia de desacoplamiento forma circuitos LC que pueden oscilar a decenas o cientos de megahertz. Cuando un circuito digital conmuta, excita estas resonancias, generando oscilaciones amortiguadas (ringing) que se propagan por las líneas de alimentación y se acoplan a circuitos analógicos sensibles. Un diseñador inexperto que coloca capacitores de desacoplamiento sin considerar la inductancia de las vías del PCB puede crear inadvertidamente resonadores que amplifican el ruido en lugar de suprimirlo.

Los componentes reactivos no son simples bloques de construcción pasivos; son elementos activos que transforman señales en función del tiempo y la frecuencia, introduciendo retardos, resonancias y distorsiones que ningún algoritmo de procesamiento digital puede corregir perfectamente si no entiendes su origen físico. En sistemas de IA que dependen de señales temporales - series de tiempo, audio, vibraciones - ignorar la física de capacitores e inductores es garantizar que tu modelo aprenderá artefactos de tu hardware en lugar de patrones del mundo real. Y eso no es machine learning. Es amplificación de ignorancia.

1.5 Leyes Fundamentales: La Física que Governa tus Datos

Objetivo de Aprendizaje

Objetivo: Aplicar las leyes de Ohm, Watt y Kirchhoff para analizar circuitos de acondicionamiento de señales **mediante** el cálculo de corrientes, voltajes y potencias en redes resistivas, **para** diseñar divisores de voltaje, redes de polarización de sensores y circuitos de medición de corriente en sistemas de adquisición de datos para IA.

Ingeniería de Datos desde el Hardware

Cada sensor que conectas a tu ADC es un divisor de voltaje entre su impedancia de salida y la impedancia de entrada del ADC. Si no entiendes la Ley de Ohm lo suficientemente profundo como para calcular el efecto de carga, tus mediciones serán sistemáticamente incorrectas, y tu modelo de IA aprenderá una función de mapeo distorsionada entre el fenómeno físico y los valores digitalizados.

Las leyes de Ohm, Watt y Kirchhoff no son recomendaciones pedagógicas ni aproximaciones útiles. Son consecuencias directas de las leyes de conservación de la física - conservación de la carga y conservación de la energía - aplicadas a circuitos eléctricos. Violarlas no es una opción; ignorarlas solo garantiza que tus cálculos serán incorrectos y tus diseños fallarán de formas misteriosas y difíciles de debuggear. En el contexto de sistemas de adquisición de datos para inteligencia artificial, estas leyes determinan si tus mediciones reflejan fielmente la realidad física o si están contaminadas por efectos de carga, caídas de voltaje parásitas y disipación de potencia que introduce errores térmicos. Un ingeniero que no domina estas leyes al nivel de aplicación automática e instintiva está construyendo sistemas sobre fundamentos de arena.

1.5.1 Ley de Ohm: Más que una Simple Ecuación

La Ley de Ohm establece que el voltaje a través de un conductor es proporcional a la corriente que fluye por él: $V = I \cdot R$. Esta ecuación de tres variables aparentemente trivial esconde profundidades conceptuales que separan a un técnico que memoriza fórmulas de un ingeniero que entiende física. Primero, la resistencia R no es una propiedad arbitraria; es una medida de cuánta oposición presenta el material al flujo de electrones debido a colisiones con la red cristalina. A nivel microscópico, cuando aplicas un

campo eléctrico a un conductor, los electrones se aceleran, colisionan con átomos, ceden energía en forma de calor, y vuelven a acelerarse. Este proceso disipativo es lo que crea la resistencia, y es la razón por la cual los cables se calientan cuando circula corriente.

Segundo, la Ley de Ohm es fundamentalmente una relación local - se aplica en cada punto del circuito en cada instante de tiempo. Si tienes un circuito con múltiples resistencias, la Ley de Ohm se cumple simultáneamente en cada componente, y resolver el circuito completo requiere plantear un sistema de ecuaciones que respete tanto Ohm como las leyes de Kirchhoff. Tercero, y esto es crítico para sistemas de medición: la resistencia de entrada de tu instrumento (multímetro, osciloscopio, ADC) forma parte del circuito, y si no es infinitamente mayor que las resistencias del circuito que estás midiendo, el acto de medir altera el sistema que intentas observar. Este es el efecto de carga que ya discutimos con multímetros, pero se extiende a cualquier interfaz de medición.

Considera un sensor de temperatura tipo RTD (Resistance Temperature Detector) cuya resistencia varía de $100\ \Omega$ a 0°C hasta $138\ \Omega$ a 100°C . Para medir esta resistencia, pasas una corriente conocida a través del sensor y mides el voltaje resultante usando la Ley de Ohm: $R = V/I$. Pero si la corriente es demasiado alta, la potencia disipada ($P = I^2R$) calentará el sensor, aumentando su resistencia y creando un error de auto-calentamiento que contamina la medición. Si usas $10\ \text{mA}$ a través de $100\ \Omega$, disiparás $P = (0,01)^2 \times 100 = 0,01\ \text{W} = 10\ \text{mW}$, lo cual puede calentar el sensor varios grados centígrados dependiendo de su masa térmica. Tu sistema de IA que procesa estas lecturas aprenderá una no-linealidad espuria causada por tu método de medición, no por la física del sensor.

1.5.2 Ley de Watt: Donde el Voltaje y la Corriente se Encuentran con la Termodinámica

La Ley de Watt establece que la potencia eléctrica - la tasa de transferencia de energía - es el producto del voltaje y la corriente: $P = V \cdot I$. Combinando con la Ley de Ohm, obtienes las formas alternativas $P = I^2R$ y $P = V^2/R$, cada una útil en contextos diferentes. Esta ley es la conexión directa entre el mundo eléctrico y el mundo térmico: cada vatio disipado en una resistencia se convierte irreversiblemente en calor que debe ser evacuado del sistema. En circuitos de potencia, esto determina el tamaño de disipadores térmicos y la capacidad de refrigeración necesaria. En circuitos de medición de precisión, determina si el auto-calentamiento de componentes introducirá deriva térmica que degrada la exactitud.

Pero la Ley de Watt tiene implicaciones más sutiles en sistemas de adquisición de datos. Cuando diseñas un divisor de voltaje para escalar una señal de 10V a 3.3V para un ADC, estás disipando potencia continuamente en

las resistencias del divisor. Si usas resistencias de $10\text{ k}\Omega$, con 10 V de entrada, la corriente es $I = 10/(10k + 10k) = 0,5\text{ mA}$, y la potencia total disipada es $P = 10 \times 0,0005 = 5\text{ mW}$. Esto parece despreciable, pero si tu sistema tiene 16 canales de adquisición funcionando continuamente, esos 5 mW por canal se convierten en 80 mW de calor constante que eleva la temperatura del PCB. Y si las resistencias del divisor tienen coeficientes de temperatura de $100\text{ ppm}/^\circ\text{C}$ y el PCB se calienta 20°C sobre ambiente, la resistencia cambiará 0.2% , introduciendo un error sistemático en todas tus mediciones.

La Figura 1.10 muestra el compromiso entre consumo de potencia y precisión en divisores de voltaje con diferentes valores de resistencia.

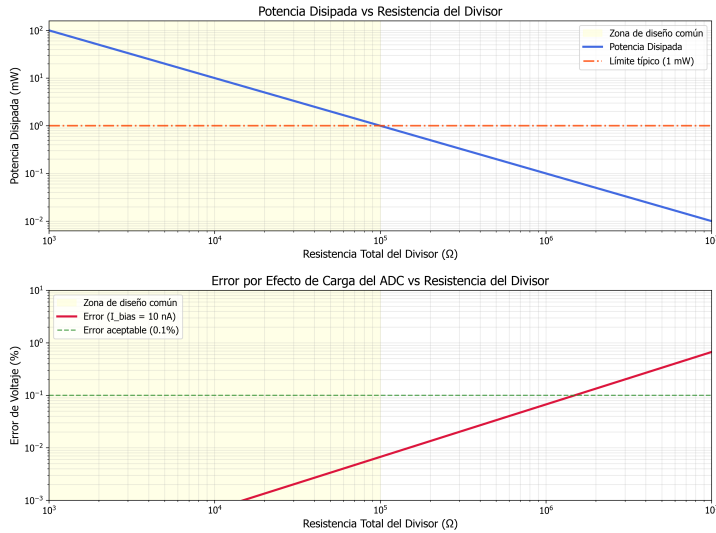


Figura 1.10: Compromiso entre potencia disipada y error por efecto de carga en divisores de voltaje. Resistencias bajas (línea azul) disipan mucha potencia pero minimizan el efecto de carga del ADC. Resistencias altas (línea roja oscuro) ahorran potencia pero son susceptibles a corrientes de polarización del ADC que introducen errores. El diseño óptimo (línea violeta) balancea ambos extremos. (Figura generada con el script `s1.5_e1.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_5_e1.py`. Modifica la variable `I_bias_ADC` (corriente de polarización del ADC) de 1 nA a 100 nA.

Análisis: ¿Cómo cambia el punto de diseño óptimo cuando aumenta la corriente de polarización? Si estás diseñando un sistema alimentado por batería que debe durar 1 año con una celda de 2000 mAh, ¿cuántos canales de adquisición puedes permitirte con divisores de 10 k Ω vs 1 M Ω ?

1.5.3 Leyes de Kirchhoff: Conservación de la Carga y la Energía

Las Leyes de Kirchhoff son consecuencias directas de dos principios fundamentales de la física. La **Ley de Corrientes de Kirchhoff (KCL)** establece que la suma algebraica de corrientes entrando a un nodo es cero: $\sum I_{in} = \sum I_{out}$. Esto es conservación de la carga eléctrica - los electrones no se crean ni se destruyen en un nodo, simplemente se redistribuyen. Si entran 100 mA por un cable y salen por otros tres, la suma de esas tres corrientes debe ser exactamente 100 mA. Si midiendo encuentras 98 mA, no estás violando la física; hay 2 mA fluyendo por algún camino que no consideraste - quizás una fuga a través de la resistencia de aislamiento del PCB o una corriente parásita por capacitancia acoplada.

La **Ley de Voltajes de Kirchhoff (KVL)** establece que la suma algebraica de voltajes alrededor de cualquier lazo cerrado es cero: $\sum V_{loop} = 0$. Esto es conservación de la energía - cuando un electrón completa un recorrido cerrado y regresa a su punto de partida, debe haber ganado y perdido exactamente la misma cantidad de energía potencial, de lo contrario estarías creando o destruyendo energía de la nada. Esta ley es lo que te permite analizar circuitos complejos descomponiéndolos en lazos independientes, escribiendo una ecuación KVL para cada lazo, y resolviendo el sistema resultante.

Estas leyes no son aproximaciones válidas «para circuitos de baja frecuencia» o «cuando las dimensiones son pequeñas». Son exactas y universales. Lo que cambia a altas frecuencias o en circuitos grandes es que debes incluir efectos que a bajas frecuencias eran despreciables: la inductancia de los cables (que genera caídas de voltaje proporcionales a $L di/dt$), la capacitancia entre pistas (que permite corrientes de desplazamiento), y el tiempo de propagación de señales (que hace que diferentes partes del circuito «vean» voltajes diferentes en el mismo instante). Pero KCL y KVL siguen siendo válidas si las aplicas correctamente incluyendo todos estos efectos.

1.5.4 Análisis de Mallas y Nodos: Ingeniería Sistemática vs Adivinanza

Hay dos métodos sistemáticos para analizar circuitos usando las Leyes de Kirchhoff: el **método de mallas** (basado en KVL) y el **método de nodos** (basado en KCL). Ambos son algoritmos que, aplicados mecánicamente, garantizan llegar a la solución correcta. El método de mallas define corrientes de malla que circulan por cada lazo cerrado del circuito, escribe ecuaciones KVL para cada malla, y resuelve para encontrar esas corrientes. Las corrientes reales en cada rama se obtienen como suma algebraica de las corrientes de malla que pasan por esa rama. El método de nodos asigna voltajes a cada nodo del circuito (tomando uno como referencia/tierra), escribe ecuaciones KCL para cada nodo expresando las corrientes en términos de esos voltajes, y resuelve para encontrar los voltajes nodales.

¿Cuál método usar? Depende de la topología del circuito. Si el circuito tiene muchas ramas en paralelo entre pocos nodos, el método de nodos produce menos ecuaciones. Si el circuito tiene muchos lazos con pocas ramas comunes, el método de mallas es más eficiente. En la práctica, un ingeniero competente alterna entre ambos métodos dependiendo del problema específico, y frecuentemente usa una combinación híbrida donde analiza parte del circuito con mallas y parte con nodos, conectando ambos análisis en las fronteras.

La Figura 1.11 compara el número de ecuaciones necesarias para analizar el mismo circuito usando método de mallas versus método de nodos.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1.5.e2.py`. Observa cómo el número de ecuaciones cambia con la topología del circuito.

Experimento: Para un circuito con 5 nodos y 8 ramas, ¿cuántas ecuaciones independientes necesitas en cada método? Verifica que ambos métodos den la misma solución final (como debe ser, si ambos son correctos).

1.5.5 Divisores de Voltaje: La Configuración Más Útil y Más Malinterpretada

El divisor de voltaje es probablemente el circuito más ubicuo en electrónica: dos resistencias en serie entre un voltaje de entrada y tierra, con la salida tomada del punto intermedio. La ecuación es inmediata por KVL: $V_{out} = V_{in} \times R_2 / (R_1 + R_2)$. Pero esta ecuación solo es válida cuando la corriente que sale del nodo central es despreciable comparada con la corriente que fluye por el divisor. En cuanto conectas una carga - un ADC, un amplificador,

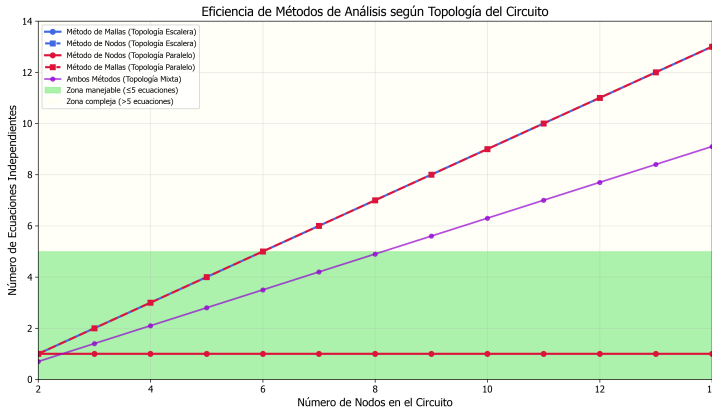


Figura 1.11: Comparación de eficiencia entre método de mallas y método de nodos para diferentes topologías de circuitos. Circuitos con topología tipo «escalera»(azul) se analizan eficientemente con mallas. Circuitos con múltiples ramas en paralelo (rojo) favorecen el método de nodos. La elección del método correcto puede reducir el trabajo de cálculo a la mitad. (Figura generada con el script `s1_5_e2.py`)

otro circuito - con impedancia finita, la corriente se desvía hacia esa carga, cambiando el voltaje de salida.

La Figura 1.12 muestra cómo el voltaje de salida de un divisor colapsa cuando aumenta la corriente de carga, comparando divisores con diferentes valores de resistencia total.

Este efecto de carga es la razón por la cual los ADCs modernos tienen amplificadores de entrada con impedancia de entrada extremadamente alta (típicamente $> 1 \text{ M}\Omega$ para ADCs tipo SAR, y efectivamente infinita para ADCs con buffer de seguidor). Pero «alta impedancia» no significa «impedancia infinita». Un ADC típico de 12 bits puede tener corrientes de polarización de entrada de 1-100 nA. Si tu divisor de voltaje tiene impedancia total de $10 \text{ M}\Omega$, esos 100 nA generarán una caída de voltaje de $V = 100 \times 10^{-9} \times 10 \times 10^6 = 1 \text{ V}$, que es un error masivo. Por eso los divisores para ADCs de alta resolución rara vez usan resistencias mayores a $100 \text{ k}\Omega$ total, balanceando consumo de potencia contra efecto de carga.

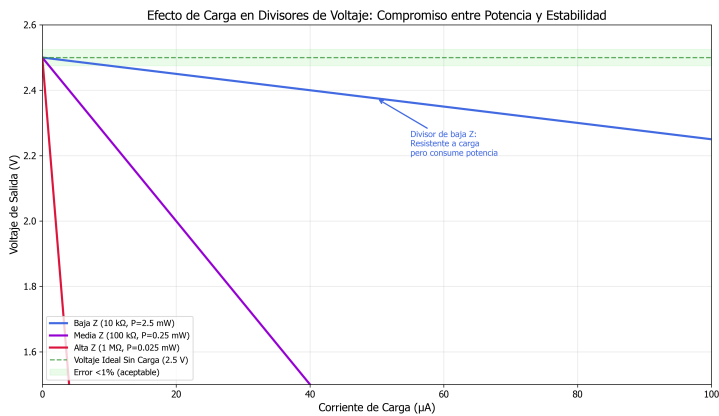


Figura 1.12: Efecto de carga en divisores de voltaje con diferentes impedancias. Divisor de baja impedancia (10 kΩ total, azul) mantiene el voltaje estable bajo carga moderada pero disipa 5 mW continuamente. Divisor de alta impedancia (1 MΩ total, rojo) ahorra potencia (50 µW) pero colapsa dramáticamente bajo carga de solo 1 µA. El diseño correcto depende del contexto de aplicación. (Figura generada con el script `s1_5_e3.py`)

Reto Computacional: Experimenta con el Script

En el script `s1_5_e3.py`, modifica la corriente de carga y observa el error en el voltaje de salida.

Pregunta crítica: Si estás diseñando un divisor 2:1 (reduce 5V a 2.5V) para un ADC con corriente de polarización de 50 nA y necesitas precisión de 0.1 %, ¿cuál es la resistencia máxima total que puedes usar? ¿Cuánta potencia disipará el divisor continuamente?

1.5.6 Medición de Corriente: El Problema del Shunt

Medir voltaje es relativamente fácil porque los voltímetros tienen alta impedancia y no alteran significativamente el circuito. Medir corriente es intrínsecamente invasivo porque debes insertar el medidor en serie con el circuito, y cualquier resistencia que agregues causará caídas de voltaje. La técnica estándar es usar una resistencia de bajo valor conocida (llamada «shunt») en serie con el circuito, medir el voltaje a través de ella, y calcular la corriente usando Ohm: $I = V/R_{shunt}$. El problema es el compromiso entre sensibilidad y caída de voltaje. Un shunt de 0.01 Ω midiendo 1 A genera solo 10 mV de señal, que requiere amplificación cuidadosa y es susceptible a ruido. Un shunt de 1 Ω genera 1 V de señal fácil de medir, pero introduce 1 V de caída en el circuito, alterando significativamente su operación.

En sistemas de adquisición de datos para IA que monitorean consumo de corriente - por ejemplo, detectando anomalías en motores industriales mediante análisis de corriente - la elección del shunt determina la relación señal-ruido de tus mediciones. Un shunt demasiado pequeño da señales enterradas en ruido; demasiado grande perturba el sistema que intentas medir. Y el shunt debe tener coeficiente de temperatura extremadamente bajo (típicamente ≤ 50 ppm/°C) porque disipa potencia ($P = I^2 R_{shunt}$) que lo calienta, cambiando su resistencia y introduciendo errores que varían con la corriente medida, una no-linealidad que tu modelo de IA aprenderá como característica del motor cuando en realidad es un artefacto de tu instrumentación.

Las leyes de Ohm, Watt y Kirchhoff no son contenido de un curso introductorio que se memoriza para un examen y luego se olvida. Son las herramientas fundamentales de razonamiento que usarás diariamente al diseñar, debuggear y optimizar cualquier sistema que involucre electricidad. En el contexto de IA, donde los algoritmos de machine learning reciben datos del mundo físico a través de cadenas de sensores, acondicionadores, ADCs y circuitos de interfaz, cada uno de esos bloques está gobernado por estas leyes. Ignorarlas es garantizar que tus datos tendrán errores sistemáticos y distorsiones que ningún preprocesamiento de software puede eliminar completamente. Y entrenar modelos de IA con datos sistemáticamente incorrectos no produce inteligencia artificial. Produce estupidez artificial con precisión de 32 bits en punto flotante.

1.6 Carga, Tensión y Corriente: Los Fundamentos Microscópicos

Objetivo de Aprendizaje

Objetivo: Comprender la naturaleza física de la carga eléctrica, la tensión y la corriente a nivel microscópico **mediante** el análisis de fenómenos de conducción y almacenamiento de energía, **para** interpretar correctamente las limitaciones físicas de sensores, actuadores y componentes electrónicos en sistemas de IA embebida.

Ingeniería de Datos desde el Hardware

Cuando un sensor de corriente te reporta 3.2 amperios, no estás midiendo un número abstracto. Estás contando aproximadamente 2×10^{19} electrones por segundo fluyendo por un conductor. Si no entiendes que esa corriente genera campos magnéticos, disipa calor, y causa caídas de voltaje proporcionales a la resistencia del camino, diseñarás sistemas de IA que fallan misteriosamente cuando la carga aumenta.

La electricidad no es magia ni abstracción matemática. Es física tangible: electrones moviéndose a través de materiales, campos eléctricos ejerciendo fuerzas sobre cargas, energía almacenándose en configuraciones de campos electromagnéticos. Cuando mides un voltaje con un multímetro, estás cuantificando la energía potencial por unidad de carga entre dos puntos. Cuando mides una corriente, estás contando cuánta carga fluye por segundo a través de una sección transversal de un conductor. Y cuando hablas de «carga eléctrica», estás refiriéndote a una propiedad fundamental de la materia que no puede ser creada ni destruida, solo redistribuida. Entender estos conceptos al nivel microscópico-no solo como definiciones memorizadas sino como fenómenos físicos reales-es lo que separa a un ingeniero que diseña sistemas robustos de uno que ensambla componentes esperando que funcionen por suerte.

1.6.1 Carga Eléctrica: La Propiedad Cuantizada que Gobierna Todo

La carga eléctrica es una propiedad fundamental de ciertas partículas subatómicas. El electrón tiene carga negativa de magnitud $e = 1,602 \times 10^{-19}$ coulombs, y el protón tiene carga positiva de exactamente la misma magnitud. Esta cuantización significa que toda carga observable es un múltiplo entero de la carga elemental-no puedes tener media carga de electrón. Cuando frotas un globo contra tu cabello y se pega a la pared, has transferido del orden de 10^{10} electrones del cabello al globo, creando un desbalance neto de carga que genera fuerzas electrostáticas. Cuando conectas una batería a un circuito, no estás «creando» electrones; estás usando reacciones químicas para bombear electrones de un terminal al otro, generando un desbalance de carga que los electrones intentan neutralizar fluyendo por el circuito externo.

La conservación de la carga eléctrica es absoluta: en cualquier proceso físico, la carga total antes y después del proceso es idéntica. Puedes separar cargas positivas de negativas (como en un capacitor), puedes mover electrones de un lugar a otro (corriente eléctrica), pero no puedes crear ni aniquilar carga neta. Esta conservación es la base física de la Ley de Corrientes de

Kirchhoff-si entran 5 coulombs por segundo a un nodo por un cable, exactamente 5 coulombs por segundo deben salir por otros cables, porque los electrones no pueden acumularse indefinidamente en un punto del circuito sin violar conservación de carga. Cuando diseñas sistemas de medición de corriente para monitoreo de consumo energético en aplicaciones de IA, estás literalmente contando electrones que fluyen, y cualquier discrepancia entre entrada y salida indica una fuga de corriente (por resistencias de aislamiento del PCB, humedad, contaminación) que contamina tus mediciones.

1.6.2 Tensión Eléctrica: Energía Potencial por Unidad de Carga

El voltaje o tensión eléctrica entre dos puntos es la energía potencial eléctrica por unidad de carga entre esos puntos, medida en voltios, donde $1 \text{ volt} = 1 \text{ joule/coulomb}$. Cuando dices que una batería tiene «9 voltios», estás diciendo que por cada coulomb de carga que la batería mueve de su terminal negativo al positivo a través del circuito externo, entrega 9 joules de energía. Esta energía no aparece de la nada; proviene de reacciones químicas que convierten energía química almacenada en energía eléctrica. Cuando esa carga regresa al terminal negativo dentro de la batería, la reacción química «recarga» su energía potencial, listo para otro ciclo.

El voltaje no es una propiedad de un punto único; siempre es una diferencia entre dos puntos. Decir «el voltaje en el nodo A es 5V» es impreciso-lo correcto es «el voltaje en A respecto a tierra es 5V». Esta dependencia de la referencia es crítica en sistemas de medición diferencial. Si intentas medir la salida de un sensor que está flotando (sin conexión directa a tierra) usando un ADC referido a tierra, obtendrás lecturas erróneas porque estás midiendo voltaje respecto a una referencia que el sensor no comparte. En sistemas de IA que procesan señales de múltiples sensores distribuidos físicamente, problemas de referencia de tierra (ground loops, diferencias de potencial entre tierras) pueden introducir voltajes de modo común de decenas o cientos de milivoltios que contaminan señales de baja amplitud.

La Figura 1.13 muestra una analogía hidráulica del voltaje como diferencia de altura potencial gravitatoria.

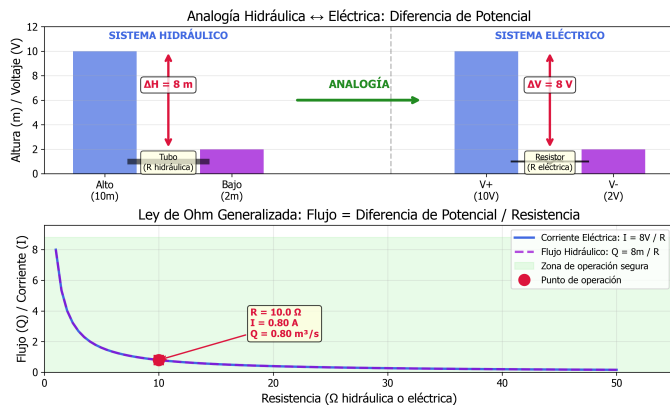


Figura 1.13: Analogía hidráulica del voltaje eléctrico. Así como el agua fluye de mayor a menor altura gravitatoria disipando energía potencial, la corriente eléctrica fluye de mayor a menor potencial eléctrico. La «altura» es análoga al voltaje, el «flujo de agua» a la corriente, y la «resistencia del tubo» a la resistencia eléctrica. (Figura generada con el script `s1_6_e1.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_6_e1.py`. El script incluye un deslizador interactivo que te permite variar la resistencia en tiempo real (de 1 a 50 Ω) y observar cómo el punto de operación se mueve a lo largo de las curvas de Ley de Ohm. Cuando disminuyes la resistencia (tubo más ancho en la analogía hidráulica), el flujo/corriente aumenta proporcionalmente. Cuando aumentas la resistencia (tubo más angosto), el flujo disminuye.

Experimento interactivo: Mueve el deslizador y observa tres cosas simultáneamente:

1. El valor de corriente eléctrica (puntos azules en la curva sólida)
2. El valor de flujo hidráulico (cuadrados violetas en la curva punteada)
3. Cómo ambos siguen exactamente la misma ley matemática: $I = \Delta V / R$ y $Q = \Delta H / R$

Pregunta conceptual: Si tienes dos reservorios de agua a diferentes alturas conectados por tuberías de diferente diámetro en serie (análogo a resistencias en serie), ¿cuál determina la velocidad de flujo-la diferencia de altura total o el diámetro del tubo más angosto? Traduce esta respuesta al dominio eléctrico: en un circuito con múltiples resistencias en serie, ¿cuál limita la corriente?

1.6.3 Corriente Eléctrica: Flujo de Carga en Movimiento

La corriente eléctrica es la tasa de flujo de carga a través de una sección transversal de un conductor, medida en amperios, donde $1 \text{ ampere} = 1 \text{ coulomb/segundo}$. Cuando dices que por un cable circulan 2 amperios, estás diciendo que 2 coulombs de carga pasan por cualquier punto de ese cable cada segundo. Dado que cada electrón porta $1,6 \times 10^{-19} \text{ C}$, 1 ampere corresponde a aproximadamente $6,24 \times 10^{18}$ electrones por segundo—un número astronómico que ilustra por qué podemos tratar la corriente como una cantidad continua en lugar de eventos discretos de electrones individuales.

Pero aquí está el detalle que sorprende a muchos: los electrones en un conductor no se mueven rápido. En un cable de cobre típico conduciendo 1 A, la velocidad de deriva promedio de los electrones es del orden de milímetros por segundo—más lento que una hormiga caminando. ¿Cómo entonces una señal eléctrica puede viajar a casi la velocidad de la luz por un cable? Porque lo que se propaga rápidamente no son los electrones individuales, sino el campo eléctrico que los empuja. Es como una manguera llena de agua: cuando abres la llave en un extremo, el agua sale casi instantáneamente del otro extremo no porque las moléculas de agua viajaron instantáneamente, sino porque el impulso de presión se propagó a través del fluido. En circuitos eléctricos, el campo electromagnético se propaga a velocidades cercanas a c , pero los portadores de carga (electrones) se arrastran lentamente.

Esta distinción es crítica cuando diseñas trazas de PCB para señales de alta velocidad. La corriente que lleva la señal puede ser pequeña (microamperios), pero si la frecuencia es alta (megahertz o gigahertz), los efectos de impedancia de línea de transmisión, reflexiones, y acoplamiento electromagnético dominan el comportamiento. Un diseñador inexperto que solo piensa en «corriente DC» fallará al diseñar trazas para buses SPI a 10 MHz que conectan sensores a un microcontrolador en un sistema de adquisición de datos para IA.

1.6.4 Densidad de Corriente y Calentamiento por Efecto Joule

Cuando la corriente fluye por un conductor de sección transversal finita, la densidad de corriente $J = I/A$ (amperios por metro cuadrado) determina cuánta potencia se disipa por unidad de volumen del conductor. La potencia disipada volumétricamente es $P_v = J^2 \rho$, donde ρ es la resistividad del material. Si la densidad de corriente es demasiado alta, el calentamiento por efecto Joule puede elevar la temperatura del conductor hasta el punto de fundir el aislamiento (en cables), degradar las uniones de soldadura (en PCBs), o activar mecanismos de electromigración que literalmente mueven

átomos de metal creando circuitos abiertos (en chips integrados).

La Figura 1.14 muestra cómo la temperatura de un conductor aumenta con la densidad de corriente para diferentes materiales.

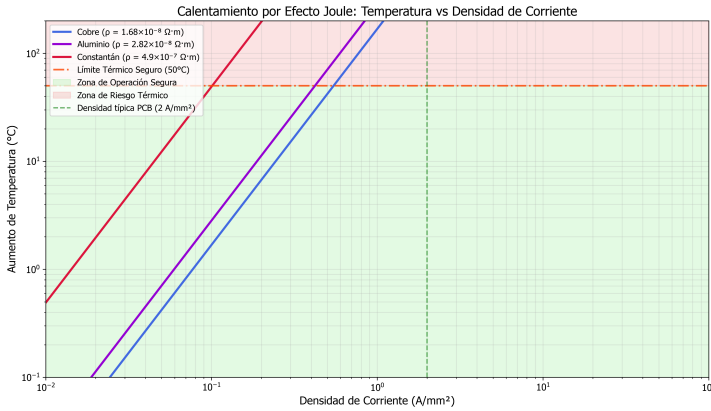


Figura 1.14: Elevación de temperatura en conductores debido a densidad de corriente. Cobre (línea azul) tiene baja resistividad y soporta mayor densidad de corriente que aluminio (violeta) o constantán (rojo). Exceder la densidad de corriente máxima causa fallas térmicas que pueden ser intermitentes y difíciles de diagnosticar. (Figura generada con el script `s1.6_e2.py`)

En sistemas embebidos para IA que operan en ambientes industriales (motores, actuadores de alta corriente), un diseño que funciona perfectamente en el laboratorio a $25^{\circ}C$ puede fallar catastróficamente en campo a $60^{\circ}C$ ambiente con corrientes de pico que exceden las especificaciones porque el diseñador no calculó correctamente la capacidad de corriente de las pistas del PCB. Una pista de 1 mm de ancho en cobre de 1 oz puede manejar aproximadamente 2-3 A antes de calentarse peligrosamente, pero este límite disminuye dramáticamente si el PCB está dentro de un gabinete sin ventilación adecuada.

Reto Computacional: Experimenta con el Script

En el script `s1.6_e2.py`, modifica la densidad de corriente y observa cómo cambia la temperatura.

Cálculo práctico: Si tienes una pista de PCB de 0.5 mm de ancho, 35 μm de espesor (1 oz de cobre), y necesitas conducir 5 A continuamente, ¿cuál será el aumento de temperatura? ¿Es seguro? Si no lo es, ¿qué soluciones tienes (aumentar ancho, usar múltiples capas en paralelo, agregar vías de costura térmica)?

1.6.5 Capacitancia e Inductancia: Almacenamiento de Energía en Campos

Los capacitores almacenan energía en campos eléctricos; los inductores en campos magnéticos. Esta afirmación simple esconde profundidades físicas. Cuando cargas un capacitor conectándolo a una fuente de voltaje, estás forzando electrones de una placa hacia la otra, creando un exceso de carga negativa en una placa y un déficit (carga positiva) en la otra. El campo eléctrico entre las placas almacena energía dada por $E = \frac{1}{2}CV^2$. Esta energía está literalmente en el espacio vacío (o dieléctrico) entre las placas, no en las cargas mismas—una manifestación de que los campos electromagnéticos son entidades físicas reales que portan energía.

Cuando haces circular corriente por un inductor, los electrones en movimiento generan un campo magnético circular alrededor del conductor. Cambiar esa corriente requiere cambiar el campo magnético, y el campo magnético se opone al cambio mediante la generación de un voltaje inducido ($V = L dI/dt$). La energía almacenada es $E = \frac{1}{2}LI^2$, y está distribuida en el espacio donde existe el campo magnético. Cuando cortas súbitamente la corriente en un inductor (por ejemplo, abriendo un interruptor mecánico), el campo magnético colapsa, induciendo voltajes que pueden alcanzar cientos o miles de voltios en microsegundos—suficiente para generar arcos eléctricos, destruir transistores, o inducir transitorios en circuitos adyacentes.

La Figura 1.15 compara el almacenamiento de energía en capacitores versus inductores.

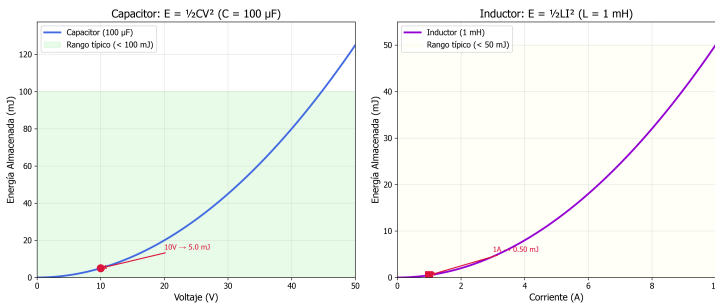


Figura 1.15: Energía almacenada en capacitores vs inductores. Los capacitores almacenan energía proporcional a V^2 , favoreciendo alto voltaje. Los inductores almacenan energía proporcional a I^2 , favoreciendo alta corriente. Para la misma energía almacenada, las estrategias de diseño son radicalmente diferentes. (Figura generada con el script `s1_6_e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_6_e3.py`. Compara cuánta energía almacena un capacitor de 100 μF a 10V versus un inductor de 1 mH a 1A.

Diseño: Si necesitas almacenar 1 joule de energía en un espacio pequeño, ¿usarías un capacitor de alto voltaje o un inductor de alta corriente? ¿Qué factores adicionales considerarías (densidad energética volumétrica, tasa de descarga, eficiencia de conversión)?

Carga, tensión y corriente no son abstracciones matemáticas. Son manifestaciones de campos electromagnéticos, movimiento de partículas cargadas, y transferencia de energía que ocurren en el mundo físico tangible. Cuando diseñas un sistema de adquisición de datos para IA y eliges un ADC, estás determinando cuántos coulombs por segundo pueden fluir por tus líneas de entrada sin saturar el amplificador de entrada. Cuando seleccionas cables de alimentación, estás calculando si la densidad de corriente causará calentamiento que cambie la resistencia y genere caídas de voltaje variables con la carga. Y cuando decides si usar capacitores de desacoplamiento de 100nF o 10 μF , estás especificando cuánta energía puede almacenarse localmente en campos eléctricos para suavizar transitorios de corriente. Ignorar la física subyacente es diseñar sistemas que funcionan por accidente en condiciones ideales y fallan misteriosamente en el mundo real.

1.7 Arreglos Serie-Paralelo: Arquitecturas de Circuitos y Distribución de Señales

Objetivo de Aprendizaje

Objetivo: Analizar el comportamiento de resistencias en configuraciones serie, paralelo y mixtas **mediante** el cálculo de resistencias equivalentes y distribución de voltajes y corrientes, **para** diseñar redes de sensores robustas y sistemas de distribución de potencia en plataformas de IA embebida.

Ingeniería de Datos desde el Hardware

Cuando conectas múltiples sensores a un sistema de adquisición de datos, la topología de conexión-serie versus paralelo-determina si una falla en un sensor tumba todo el sistema o solo afecta una lectura. En redes de sensores industriales para IA, esta distinción es la diferencia entre downtime de milisegundos (reentrenar el modelo sin ese sensor) versus horas (esperar mantenimiento porque toda la cadena de medición colapsó).

Las resistencias en serie y paralelo no son solo ejercicios académicos de circuitos simplificados. Son patrones arquitectónicos fundamentales que aparecen en cada sistema electrónico real: divisores de voltaje para acondicionar señales de sensores, arreglos de LEDs en paralelo para iluminación, resistencias de pull-up/pull-down en buses digitales, shunts de corriente en configuraciones de cuatro terminales, y redes de terminación para líneas de transmisión de alta velocidad. Cuando diseñas un sistema de IA embebida, cada decisión sobre cómo conectar componentes—si poner sensores en cadena (serie) o distribuidos independientemente (paralelo), si alimentar módulos desde una fuente común o dedicadas, si usar divisores resistivos o buffers activos—está determinando la confiabilidad, el consumo de potencia, y la inmunidad al ruido de todo el sistema. Y entender por qué una topología es superior a otra en cada contexto requiere dominar el comportamiento de arreglos serie-paralelo a un nivel que va más allá de memorizar fórmulas.

1.7.1 Resistencias en Serie: Arquitectura de Cadena Vulnerable

Cuando conectas resistencias en serie—un extremo de una conectado al extremo de la siguiente—estás creando un camino único para la corriente. La corriente que fluye por cada resistencia es idéntica porque los electrones no tienen otro camino; es como una fila de personas atravesando una puerta estrecha, donde la tasa de flujo está limitada por el cuello de botella pero todos avanzan a la misma velocidad. La resistencia total del arreglo es simplemente la suma de las resistencias individuales: $R_{total} = R_1 + R_2 + R_3 + \dots$. Esta aditividad tiene consecuencias profundas. Si tienes tres resistencias de 1 k Ω en serie, la resistencia total es 3 k Ω , y si aplicas 9V a través de toda la cadena, la corriente será $I = 9V/3k\Omega = 3$ mA, y cada resistencia de 1 k Ω caerá exactamente 3V.

Esta distribución de voltaje es el fundamento de los divisores de voltaje—la herramienta más básica y más abusada en electrónica. Un divisor resistivo de dos resistencias en serie permite generar un voltaje intermedio entre la fuente y tierra: $V_{out} = V_{in} \cdot \frac{R_2}{R_1 + R_2}$. Parece simple, pero esconde trampas mortales. Primero, el voltaje de salida es válido solo cuando no extraes

corriente significativa del punto medio; si conectas una carga, estás poniendo una resistencia en paralelo con R_2 , reduciendo su resistencia efectiva y colapsando el voltaje de salida. Segundo, el divisor disipa potencia continuamente ($P = V^2/R_{total}$), desperdiciando energía incluso cuando no estás midiendo nada. Tercero, la impedancia de salida del divisor es la resistencia equivalente de Thévenin vista desde el punto medio, que puede ser lo suficientemente alta como para cargar capacitancias parásitas lentamente, limitando el ancho de banda de la señal.

En sistemas de adquisición de datos para IA, los divisores resistivos se usan constantemente para escalar voltajes de sensores al rango de entrada de ADCs (por ejemplo, convertir 0-10V de un sensor industrial a 0-3.3V para un ADC de microcontrolador). Pero si el divisor tiene impedancia total de 100 k Ω y el ADC tiene capacitancia de entrada de 10 pF más 50 cm de traza de PCB con 2 pF/cm adicionales, la constante de tiempo de carga es $\tau = R \cdot C \approx 50k\Omega \cdot 20pF = 1\mu s$. Si intentas muestrear a 1 MSPS (1 microsegundo por muestra), el voltaje en el capacitor del ADC no alcanzará a estabilizarse antes de la conversión, introduciendo errores de muestreo que tu modelo de IA interpretará como ruido aleatorio cuando en realidad es distorsión sistemática causada por tu diseño de acondicionamiento de señal.

La Figura 1.16 muestra cómo se distribuye el voltaje en cadenas resistivas serie y el efecto de carga cuando conectas impedancias al punto medio.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_7_e1.py`. Modifica el valor de `R_carga` para simular diferentes impedancias de entrada de ADCs o buffers.

Diseño crítico: Si tienes un divisor de 10 k Ω + 10 k Ω para generar 1.65V desde 3.3V, y conectas un ADC con impedancia de entrada de 50 k Ω , ¿cuál será el voltaje real medido? ¿Cuánto error introduces? Si este divisor alimenta un canal de un sensor de temperatura calibrado que espera exactamente 1.65V para 25°C, ¿cuántos grados Celsius de error aparente generas?

Otra consecuencia de las configuraciones serie es la vulnerabilidad a fallas en cadena. Si tienes cinco LEDs en serie alimentados por una fuente de corriente constante y uno de ellos falla en circuito abierto (el filamento se rompe), toda la cadena se apaga porque la corriente no puede fluir. En sistemas de iluminación esto es molesto; en cadenas de sensores críticos para seguridad (por ejemplo, termopares en serie monitoreando temperatura en un reactor industrial) es catastrófico. Esta es la razón por la cual las topologías de sensores industriales modernos usan conexiones paralelas con multiplexación, no cadenas serie.

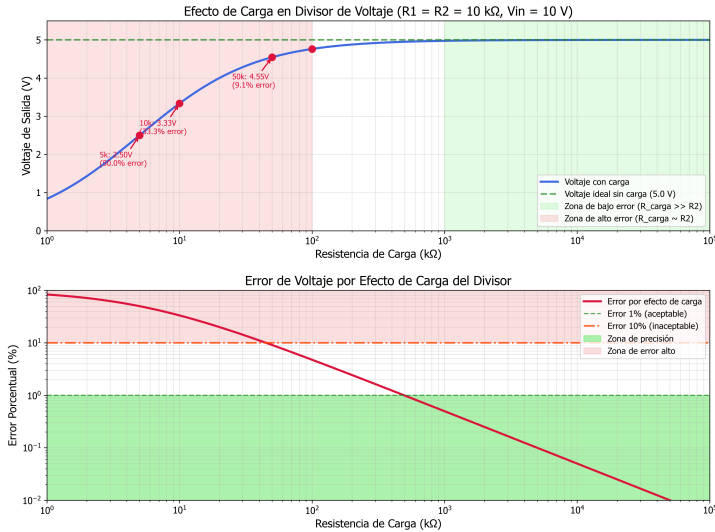


Figura 1.16: Distribución de voltaje en resistencias serie y efecto de carga. Izquierda: divisor ideal sin carga muestra distribución proporcional perfecta. Derecha: al conectar una carga (resistencia en paralelo), el voltaje colapsa proporcionalmente a la impedancia de carga. Si la carga es $10\times$ más alta que la resistencia del divisor, el error es 9%; si es solo $2\times$, el error supera el 30 %. (Figura generada con el script `s1.7_e1.py`)

1.7.2 Resistencias en Paralelo: Arquitectura Redundante con División de Corriente

Cuando conectas resistencias en paralelo-todos los extremos superiores unidos, todos los inferiores unidos-estás creando múltiples caminos para que la corriente fluya. El voltaje a través de cada resistencia es idéntico (están conectadas a los mismos dos nodos), pero la corriente se divide entre ellas inversamente proporcional a sus resistencias: resistencias más pequeñas conducen más corriente, como tuberías más anchas conducen más agua. La resistencia equivalente de resistencias en paralelo es siempre menor que la más pequeña de ellas, dada por: $\frac{1}{R_{eq}} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \dots$

Esta reducción de resistencia equivalente es contraintuitiva al principio: ¿cómo agregar más resistencias puede reducir la resistencia total? Porque estás agregando más caminos para la corriente. Es análogo a abrir más carriles en una autopista: la “resistencia” al flujo de autos disminuye. Para dos resistencias iguales en paralelo, la resistencia equivalente es la mitad de una sola; para N resistencias iguales R en paralelo, $R_{eq} = R/N$. Pero si las resistencias son diferentes, la equivalente está dominada por la más

pequeña. Si pones $1\text{ k}\Omega$ en paralelo con $10\ \Omega$, la equivalente es aproximadamente $9.9\ \Omega$ -el camino de baja resistencia cortocircuita efectivamente el de alta resistencia.

Esta propiedad es fundamental en el diseño de fuentes de corriente de referencia para bias de sensores. Si necesitas generar 1 mA estable para alimentar un fotodiodo, puedes usar una resistencia de $10\text{ k}\Omega$ a través de 10 V . Pero si la resistencia tiene tolerancia de 5% , tu corriente variará 5% . Si en lugar de eso pones dos resistencias de $20\text{ k}\Omega$ en paralelo (dando $10\text{ k}\Omega$ nominal), las variaciones individuales se promedian parcialmente, reduciendo el error. Con N resistencias idénticas en paralelo, la desviación estándar de la resistencia equivalente disminuye por factor de $\sqrt{N}$ (si las variaciones son independientes). Es una forma primitiva de averaging que mejora precisión a costa de área de PCB y componentes.

La división de corriente en arreglos paralelos también determina cómo se distribuye la potencia disipada. Si conectas tres resistencias de $10\ \Omega$, $20\ \Omega$, y $50\ \Omega$ en paralelo a través de 5 V , la de $10\ \Omega$ conduce 0.5 A y disipa 2.5 W , la de $20\ \Omega$ conduce 0.25 A y disipa 1.25 W , y la de $50\ \Omega$ conduce 0.1 A y disipa 0.5 W . La resistencia más pequeña sufre el mayor estrés térmico. En arreglos de LEDs en paralelo sin resistencias de limitación individuales, las variaciones de V_f (voltaje forward) entre LEDs hacen que algunos conduzcan mucho más corriente que otros, calentándose más, reduciendo su V_f , conduciendo aún más corriente-un proceso de realimentación térmica positiva que eventualmente destruye los LEDs más débiles. Por eso los diseños robustos de arreglos de LEDs usan resistencias de ballast individuales en serie con cada LED, no un solo resistor compartido.

La Figura 1.17 muestra la división de corriente en configuraciones paralelas y el efecto de desbalance de carga.

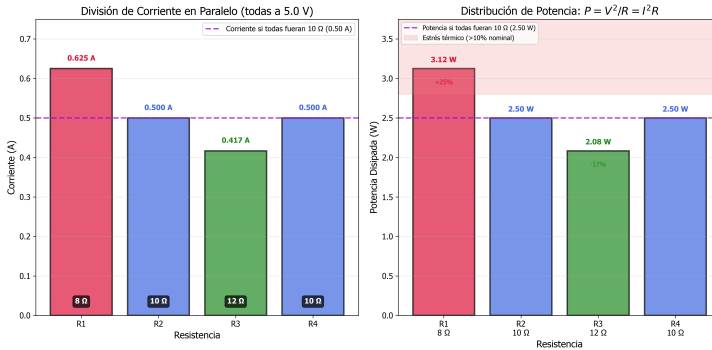


Figura 1.17: División de corriente en resistencias en paralelo. Cuando múltiples caminos están disponibles, la corriente se distribuye inversamente proporcional a las resistencias. Un camino de baja resistencia “roba” corriente de caminos de alta resistencia. En sistemas de distribución de potencia, esto significa que módulos con impedancia de entrada baja consumen desproporcionadamente más corriente que módulos de alta impedancia conectados en paralelo a la misma fuente. (Figura generada con el script `s1_7_e2.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_7_e2.py`. El script muestra tres casos de cuatro resistencias en paralelo conectadas al mismo voltaje (5V): resistencias idénticas (ideal), con variación de $\pm 10\%$, y con variación de $\pm 20\%$. Observa cómo pequeñas diferencias en valores de resistencias causan desbalances significativos en la corriente y potencia disipada por cada resistencia.

Pregunta cuantitativa: En el caso de variación $\pm 20\%$, la resistencia más baja es $8\ \Omega$ y la nominal es $10\ \Omega$. Ambas están conectadas a 5V. Calcula:

1. La corriente por cada una (usa Ley de Ohm: $I = V/R$)
2. La potencia disipada por cada una (usa $P = V^2/R$ o $P = I^2R$)
3. ¿Qué porcentaje más de potencia disipa la de $8\ \Omega$ comparada con la de $10\ \Omega$?

Aplicación: Si estas fueran resistencias de ballast para LEDs en paralelo, ¿por qué el LED con la resistencia más baja se quemaría primero debido al desbalance térmico?

1.7.3 Redes Mixtas Serie-Paralelo: Simplificación Sistemática

La mayoría de los circuitos reales no son puramente serie ni paralelo, sino combinaciones complejas de ambos. Resolver redes mixtas requiere reducción sistemática: identificar grupos de resistencias que están inequívocamente en serie o paralelo, reemplazarlos por su equivalente, y repetir hasta obtener una sola resistencia equivalente total. Este proceso no es único-diferentes secuencias de simplificación pueden llegar al mismo resultado-pero requiere reconocimiento de patrones y cuidado para no confundir topologías. Dos resistencias están en serie solo si comparten un nodo común al cual no está conectada ninguna otra resistencia. Dos resistencias están en paralelo solo si ambos extremos están conectados a los mismos dos nodos. Si no cumple ninguno de estos criterios, no puedes simplificarlas directamente; necesitas transformaciones como delta-estrella o análisis nodal.

Un ejemplo común de red mixta es un divisor resistivo cargado: R_1 en serie con R_2 , pero con R_L (resistencia de carga) en paralelo con R_2 . No puedes combinar R_1 y R_2 directamente porque no están en serie puro (hay una tercera conexión en el nodo intermedio). Primero combinas R_2 y R_L en paralelo: $R_{eq} = \frac{R_2 \cdot R_L}{R_2 + R_L}$. Luego combinas esta equivalente con R_1 en serie: $R_{total} = R_1 + R_{eq}$. La corriente total es $I_{total} = V_{in}/R_{total}$, el voltaje en el nodo intermedio es $V_{out} = I_{total} \cdot R_{eq}$, y de ahí puedes calcular cuánta corriente va por R_2 versus R_L usando división de corriente.

Este análisis es exactamente lo que haces cuando diseñas interfaces de sensores. Un sensor resistivo (como un RTD de temperatura) actúa como R_2 en un divisor, pero el ADC que lo lee tiene impedancia de entrada finita R_L (típicamente del orden de megaohms, pero no infinita). Si $R_L \gg R_2$, su efecto es despreciable. Pero si tu sensor tiene impedancia de salida comparable a la impedancia de entrada del ADC-por ejemplo, un sensor piezorresistivo de alta impedancia conectado a un ADC con amplificador de entrada simple-el efecto de carga introduce errores que varían con la temperatura (porque tanto el sensor como los buffers del ADC tienen coeficientes térmicos). Tu modelo de IA aprenderá estas correlaciones espurias entre temperatura ambiente y lecturas del sensor, reduciendo la capacidad de generalización cuando despliegas el sistema en un ambiente con temperatura diferente a la de entrenamiento.

La Figura 1.18 muestra un ejemplo de red mixta y su proceso de simplificación paso a paso.

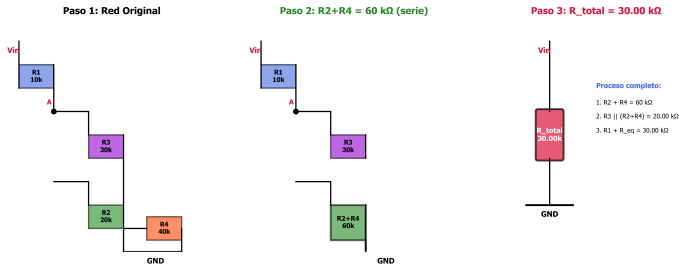


Figura 1.18: Reducción sistemática de una red mixta serie-paralelo. Izquierda: circuito original con topología compleja. Centro: primer paso de simplificación combinando resistencias que están claramente en serie o paralelo. Derecha: resistencia equivalente final. El proceso requiere reconocimiento cuidadoso de patrones para evitar errores. (Figura generada con el script `s1_7_e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_7_e3.py`. El script muestra interactivamente el proceso de reducción de una red mixta, iluminando cada paso de simplificación.

Desafío de reconocimiento: El script genera redes aleatorias. Antes de ver la solución animada, identifica mentalmente qué resistencias puedes combinar primero. ¿Cuántos pasos de reducción necesitas? ¿Existe más de una secuencia válida de simplificación?

1.7.4 Teorema de Thévenin y Norton: Abstracción de Redes Complejas

Cualquier red resistiva con fuentes independientes, vista desde dos terminales externos, es eléctricamente indistinguible de una sola fuente de voltaje V_{th} en serie con una resistencia R_{th} (equivalente de Thévenin), o equivalentemente, una fuente de corriente I_N en paralelo con R_N (equivalente de Norton). Estos teoremas no son trucos matemáticos abstractos; son herramientas de abstracción que te permiten reemplazar circuitos complejos de acondicionamiento de señal con un modelo simple de dos parámetros que captura completamente su comportamiento hacia cargas externas. Cuando conectas un sensor a un ADC, el sensor puede tener internamente decenas de componentes (amplificadores, filtros, divisores de referencia), pero desde la perspectiva del ADC, es solo un equivalente de Thévenin: un voltaje de salida con cierta impedancia de salida.

El voltaje de Thévenin V_{th} es el voltaje de circuito abierto entre los termi-

nales (lo que mides con un voltímetro de alta impedancia). La resistencia de Thévenin R_{th} es la resistencia vista desde los terminales con todas las fuentes independientes reemplazadas por sus resistencias internas (fuentes de voltaje reemplazadas por cortocircuitos, fuentes de corriente por circuitos abiertos). En el equivalente de Norton, $I_N = V_{th}/R_{th}$ es la corriente de cortocircuito, y $R_N = R_{th}$. Estos modelos son intercambiables; cuál uses depende de si piensas en el circuito como fuente de voltaje (Thévenin) o fuente de corriente (Norton).

La importancia práctica es que una vez que conoces el equivalente de Thévenin de tu circuito de acondicionamiento de señal, puedes diseñar la etapa siguiente sin necesidad de analizar toda la red aguas arriba. Si tu sensor tiene $V_{th} = 2,5V$ y $R_{th} = 10k\Omega$, sabes inmediatamente que conectarlo a un ADC con impedancia de entrada de $100\text{ k}\Omega$ causará un error de carga de aproximadamente $10k/(10k + 100k) \approx 9\%$ en el peor caso, y que la constante de tiempo de carga con capacitancia de entrada de 20 pF será $\tau = 10k \cdot 20pF = 200ns$. Sin necesidad de analizar el interior del sensor.

Esta abstracción es la base del diseño modular: cada bloque de un sistema de adquisición de datos se especifica por su impedancia de entrada (como carga) y su impedancia de salida (como fuente). Cuando encadenas bloques, verificas que cada etapa tiene impedancia de entrada al menos $10\times$ mayor que la impedancia de salida de la etapa anterior. Si cumple esta regla, el efecto de carga es menor al 10% , típicamente aceptable. Si no cumple, necesitas agregar un buffer de alta impedancia de entrada (como un seguidor de voltaje con OpAmp) entre las etapas.

1.7.5 Topologías de Sensores: Serie vs Paralelo en Redes Distribuidas

Cuando diseñas una red de múltiples sensores para un sistema de IA, la topología de conexión afecta profundamente la confiabilidad, el consumo de potencia, y la complejidad del cableado. En configuraciones serie (daisy-chain), conectas cada sensor al siguiente en cadena, con solo dos cables recorriendo toda la red. Esto minimiza cableado pero crea punto único de falla: si cualquier sensor falla en abierto, toda la cadena se interrumpe. Además, en cadenas largas, la caída de voltaje acumulativa en los cables puede hacer que los sensores al final de la cadena operen con voltaje insuficiente. Protocolos industriales como $4-20\text{ mA}$ usan topología serie (loop de corriente) precisamente porque la corriente es constante en toda la cadena independientemente de las caídas de voltaje, y cada sensor modula la corriente del loop según su medición.

En configuraciones paralelas, cada sensor tiene conexión directa e independiente al concentrador central (topología estrella). Esto requiere más cableado (cada sensor necesita un par de cables dedicado) pero ofrece ais-

lamiento de fallas: un sensor fallando no afecta a otros. También permite multiplexación: puedes muestrear sensores selectivamente, apagar sensores para ahorro de energía, o reemplazar sensores individuales sin interrumpir el sistema completo. Esta es la arquitectura dominante en sistemas modernos de adquisición de datos multicanal para IA.

Las topologías mixtas-como buses compartidos con derivaciones (I2C, CAN, RS-485)-intentan balancear cableado reducido con cierta robustez a fallas. Cada sensor se conecta a un bus común mediante una derivación corta (stub). Si un sensor falla en cortocircuito, puede tirar el bus completo; si falla en abierto, solo ese sensor se pierde. El diseño cuidadoso de resistencias de terminación y limitación de corriente en derivaciones mejora la robustez.

En el contexto de IA embebida, donde despliegas decenas o cientos de sensores en un ambiente industrial, la topología determina no solo la confiabilidad sino también la calidad de los datos. Un sensor fallando que corrompe datos de otros sensores en el mismo bus genera correlaciones espurias en tu dataset que el modelo aprenderá como patrones reales, reduciendo la capacidad de generalización. Separar sensores en canales independientes con aislamiento digital (usando optoacopladores o aisladores magnéticos) es costoso pero esencial en ambientes ruidosos donde los loops de tierra y acoplamientos electromagnéticos introducen ruido común que contamina múltiples canales simultáneamente.

Las configuraciones serie-paralelo no son solo ejercicios de circuitos; son decisiones arquitectónicas que afectan cada aspecto de tu sistema de IA embebida. Cuando reduces una red resistiva compleja a su equivalente de Thévenin, estás abstraendo detalles irrelevantes y focalizando en lo que importa para la interfaz. Cuando decides si usar divisores resistivos o buffers activos, estás balanceando consumo de potencia versus precisión y ancho de banda. Y cuando eliges topología serie versus paralelo para una red de sensores, estás definiendo el modo de falla dominante del sistema completo. Diseñar sin entender estas implicaciones es construir sobre fundamentos frágiles que colapsarán bajo condiciones reales de operación.

1.8 Medición en Arreglos Complejos: Técnicas de Diagnóstico de Circuitos

Objetivo de Aprendizaje

Objetivo: Aplicar técnicas sistemáticas de medición de voltaje y corriente en redes resistivas complejas **mediante** el uso estratégico de multímetros y osciloscopios, **para** diagnosticar fallas en circuitos de adquisición de datos y validar modelos de sensores en sistemas de IA embebida.

Ingeniería de Datos desde el Hardware

Un modelo de IA entrenado con datos de sensores defectuosos aprenderá las anomalías como patrones reales. Saber medir sistemáticamente voltajes, corrientes y resistencias en una red de sensores compleja es la diferencia entre detectar una falla en 5 minutos con un multímetro versus recolectar meses de datos contaminados que destruyen la capacidad de generalización de tu modelo.

Cuando un circuito no funciona como esperas - un sensor reporta valores erróneos, un módulo no enciende, una señal tiene distorsión inexplicable - la tentación es reemplazar componentes al azar esperando que el problema desaparezca. Esta estrategia de «shotgun debugging» es costosa, lenta, y frecuentemente falla porque el componente que reemplazas puede no ser la causa raíz del problema. La alternativa profesional es diagnóstico sistemático mediante mediciones estratégicas: medir voltajes en nodos clave para verificar que las fuentes de alimentación están funcionando, medir corrientes para detectar cortocircuitos o circuitos abiertos, medir resistencias para identificar soldaduras frías o pistas dañadas, y usar osciloscopios para detectar ruido, oscilaciones parásitas, o timing incorrecto. Esta sección te enseña no solo cómo usar los instrumentos, sino cómo pensar estratégicamente sobre qué medir, dónde medir, y cómo interpretar los resultados para converger rápidamente hacia la falla real.

1.8.1 Medición de Voltaje: Mapeo Topológico

El voltaje es relativo - siempre mides la diferencia de potencial entre dos puntos. Cuando conectas un multímetro en modo voltímetro, la punta roja mide el potencial en un nodo y la punta negra establece la referencia (típicamente tierra o común). La impedancia de entrada del voltímetro (típicamente $10\text{ M}\Omega$ para multímetros digitales) está en paralelo con el circuito

que estás midiendo, por lo que introduce una carga mínima en la mayoría de los casos. Pero si estás midiendo un divisor de voltaje de alta impedancia (resistencias de megaohms), incluso el voltímetro puede cargar suficiente el circuito como para cambiar el voltaje que estás intentando medir - un ejemplo de efecto del observador en medición.

La estrategia sistemática de medición de voltaje en un circuito con múltiples etapas es trabajar desde la fuente hacia la carga: primero verificas que el voltaje de alimentación está correcto en la entrada del circuito, luego mides en cada nodo intermedio siguiendo el flujo de señal o potencia. Si encuentras un nodo donde el voltaje es significativamente diferente del esperado, has acotado la región del problema. Por ejemplo, si un sensor debería entregar 2.5V a un ADC pero mides 0V en la entrada del ADC, el problema está entre el sensor y el ADC. Mide el voltaje directamente en la salida del sensor: si ahí lees 2.5V, la falla es el cable o la conexión intermedia. Si lees 0V también en el sensor, el problema está en el sensor mismo o su alimentación.

En redes resistivas complejas, puedes usar mediciones de voltaje para validar tus cálculos teóricos de divisores y caídas de voltaje. Si calculas que un nodo debería estar a 3.3V pero mides 2.8V, hay una resistencia parásita (soldadura fría, cable dañado, pista de PCB con fisura) consumiendo 0.5V que no está en tu modelo. Esta discrepancia te dice que tu modelo del circuito está incompleto, y que necesitas buscar resistencias no intencionales en serie con tu carga.

La Figura 1.19 muestra una estrategia sistemática para medir voltajes en una red de distribución de potencia con múltiples cargas.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1.8_e1.py`. Modifica la variable `R_cable` para simular diferentes resistencias de cables y observa cómo afectan el voltaje recibido por las cargas.

Diagnóstico real: Si diseñas un sistema con ESP32 (que requiere $3.3V \pm 0.3V$ para operar confiablemente) alimentado desde un regulador que entrega exactamente 3.3V en su terminal de salida, pero el ESP32 está a 50 cm de distancia conectado por cable AWG 26 ($0.13 \Omega/\text{metro}$), ¿cuál será el voltaje real en el ESP32 si consume 200 mA durante transmisión WiFi? ¿Está dentro de especificación? Si no, ¿qué soluciones tienes (aumentar calibre de cable, usar regulador local, aumentar voltaje de regulador)?

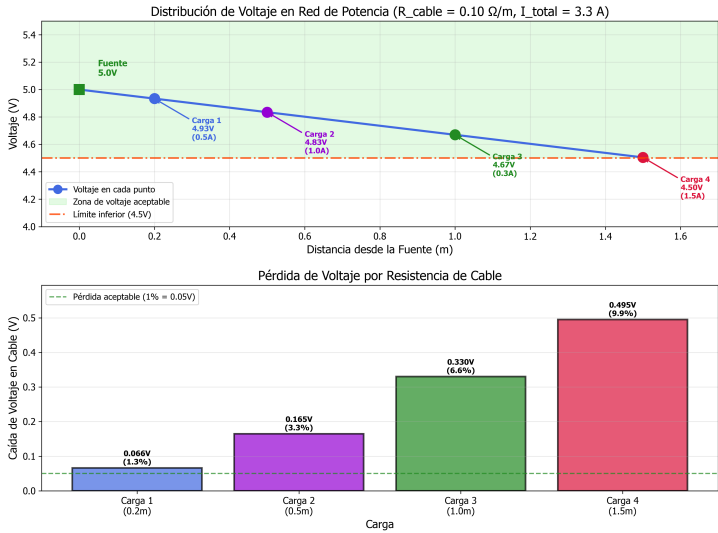


Figura 1.19: Estrategia de medición de voltaje en red de distribución de potencia. Medir en orden desde la fuente hacia las cargas permite detectar caídas de voltaje parásitas en cables, conectores y pistas de PCB. Si la fuente entrega 5.0V pero las cargas reciben 4.7V, hay 0.3V perdidos en resistencias de distribución que causan problemas de funcionamiento. (Figura generada con el script `s1_8.e1.py`)

1.8.2 Medición de Corriente: Rastreo del Flujo de Carga

A diferencia del voltaje, que se mide en paralelo entre dos puntos, la corriente se mide en serie insertando el amperímetro en el camino de la corriente. Esto significa que debes interrumpir físicamente el circuito, abrir la conexión en el punto donde quieres medir, y conectar el amperímetro en serie. La impedancia interna del amperímetro (típicamente $< 1 \Omega$ en multímetros digitales en rangos de amperios) introduce una pequeña caída de voltaje, pero en la mayoría de los circuitos es despreciable comparada con las resistencias de carga.

El problema con la medición directa de corriente es que es invasiva y arriesgada. Si accidentalmente conectas el amperímetro en paralelo (como si fuera voltímetro), su baja impedancia actúa como cortocircuito, conduciendo corrientes masivas que destruyen el fusible interno del multímetro y potencialmente dañan el circuito. Por eso, en sistemas de adquisición de datos en producción, la corriente se mide indirectamente usando resistencias shunt calibradas de bajo valor (0.01Ω a 0.1Ω) en serie con la carga, midiendo el

voltaje a través del shunt, y calculando $I = V_{shunt}/R_{shunt}$. Esta técnica no invasiva permite monitoreo continuo de corriente sin interrumpir el circuito.

En diagnóstico de circuitos con múltiples cargas en paralelo, medir la corriente total que sale de la fuente te dice si hay una carga anómala consumiendo más corriente de la esperada (indicando cortocircuito parcial o componente dañado), o menos corriente (indicando circuito abierto o componente que no está funcionando). Si calculas que tu circuito debería consumir 500 mA pero mides 2 A, hay un cortocircuito en alguna parte. Desconecta cargas una por una mientras monitorizas la corriente total; cuando la corriente cae dramáticamente al desconectar una carga específica, has identificado el módulo con falla.

La Figura 1.20 muestra el uso de resistencias shunt para medición no invasiva de corriente en múltiples ramas de un circuito.

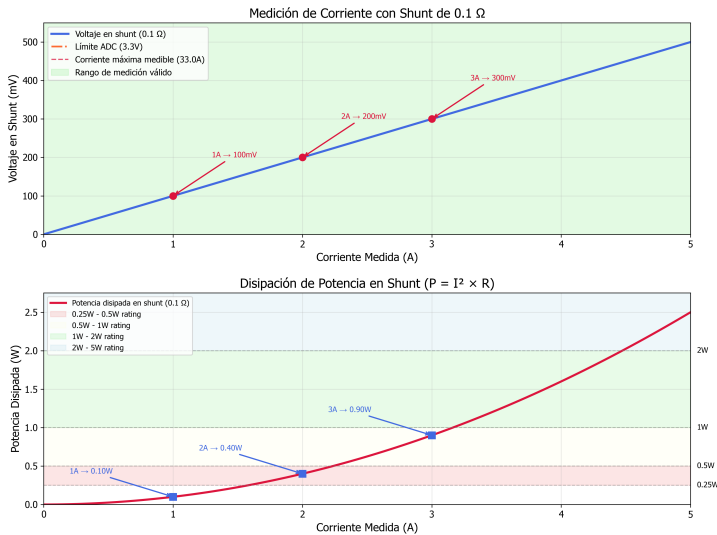


Figura 1.20: Medición de corriente usando resistencias shunt. Insertar un shunt de 0.1 Ω en serie con cada carga permite medir corrientes sin interrumpir el circuito. Un shunt conduciendo 1 A genera 100 mV fácilmente medibles con voltímetro. La desventaja es que disipa potencia ($P = I^2 R_{shunt}$), por lo que shunts para altas corrientes deben ser de potencia adecuada. (Figura generada con el script `s1_8_e2.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1.8_e2.py`. Observa cómo la caída de voltaje en el shunt y la potencia disipada aumentan cuadráticamente con la corriente.

Diseño de shunt: Si necesitas medir corrientes de hasta 5 A con un ADC de 12 bits que puede leer 0-3.3V, ¿qué valor de shunt usarías para maximizar la resolución sin exceder 3.3V a corriente máxima? ¿Cuánta potencia disipará el shunt a 5 A? ¿Qué rating de potencia necesita la resistencia (considera factor de seguridad $2\times$)?

1.8.3 Medición de Resistencia: Detección de Fallas Pasivas

La medición de resistencia te permite verificar valores de componentes, detectar soldaduras frías (resistencia anormalmente alta), identificar cortocircuitos (resistencia cercana a cero), y validar continuidad de cables y pistas de PCB. La regla fundamental es que **siempre debes medir resistencias con el circuito desenergizado y el componente desconectado de otros componentes activos**. Si intentas medir resistencia en un circuito energizado, puedes dañar el multímetro. Si mides resistencia de un componente mientras está conectado en paralelo con otros componentes, medirás la resistencia equivalente de todo el conjunto, no del componente individual.

La estrategia de diagnóstico con medición de resistencia es identificar componentes sospechosos (basados en mediciones de voltaje/corriente anómalas), desenergizar el circuito, desoldar uno de los terminales del componente sospechoso, y medir su resistencia. Si una resistencia marcada como 10 k Ω mide 12 k Ω , está dentro de tolerancia típica (5-10 %); si mide 100 k Ω , está dañada (probablemente sobrecalentada o excedida en voltaje). Si un cable que debería tener resistencia cercana a cero ($< 1\Omega$) mide 50 Ω , hay una fractura interna en el conductor que crea alta resistencia de contacto intermitente.

En sistemas de sensores resistivos (RTDs, termistores, potenciómetros), la medición directa de resistencia con el sensor desconectado te permite validar su calibración. Un RTD PT100 a 0°C debería medir exactamente 100.0 Ω ; a 25°C, aproximadamente 109.7 Ω . Si mide valores fuera de este rango, el sensor está descalibrado o dañado, y los datos que genere serán sistemáticamente incorrectos. Ningún algoritmo de IA puede corregir errores de calibración si no sabes que existen; necesitas validar físicamente los sensores antes de confiar en sus datos.

La Figura 1.21 muestra ejemplos de diagnóstico de fallas mediante medición

de resistencia.

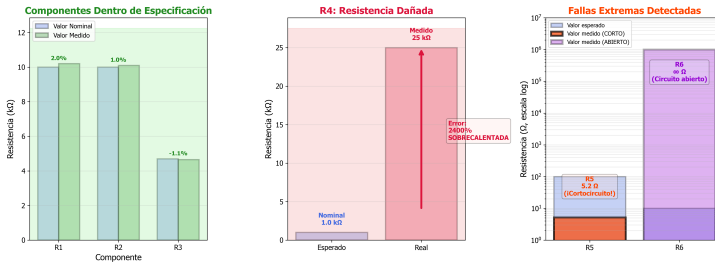


Figura 1.21: Diagnóstico de fallas mediante medición de resistencia. Izquierda: componentes dentro de especificación muestran valores esperados. Centro: resistencias anormalmente altas indican sobrecalentamiento o soldaduras frías. Derecha: resistencias cercanas a cero indican cortocircuitos. Comparar mediciones reales con valores esperados revela fallas que no son evidentes midiendo solo voltajes. (Figura generada con el script `s1_8_e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s1_8_e3.py`. El script simula mediciones de resistencia en componentes con diferentes tipos de fallas.

Caso de estudio: Tienes un divisor de voltaje que debería generar 1.65V desde 3.3V usando dos resistencias de 10 kΩ, pero mides 1.2V en la salida. Desconectas las resistencias y mides: R1 = 10.2 kΩ (normal), R2 = 45 kΩ (¡anormal!). ¿Cómo explica esto el voltaje bajo? Si R2 se sobrecalentó debido a corriente excesiva, ¿qué pudo haber causado esa corriente (cortocircuito temporal, conexión incorrecta del ADC)?

1.8.4 Técnicas Avanzadas: Medición de Ruido y Oscilaciones Parásitas

El multímetro mide valores DC o RMS promediados en el tiempo, pero no te dice nada sobre componentes AC, transitorios rápidos, o ruido de alta frecuencia superpuesto a señales DC. Para eso necesitas osciloscopio. En un sistema de adquisición de datos para IA, ruido en las líneas de alimentación puede contaminar señales analógicas, jitter en relojes puede causar errores de muestreo, y oscilaciones parásitas en salidas de OpAmps pueden saturar ADCs intermitentemente. Ninguna de estas fallas es visible con multímetro - el voltaje DC promedio puede parecer correcto mientras el ruido AC destruye tu relación señal-ruido.

La técnica es medir con osciloscopio en modo AC (bloqueando el componente DC para amplificar solo las variaciones AC) en puntos críticos: rieles

de alimentación (buscando ripple excesivo), salidas de sensores (buscando ruido de alta frecuencia), entradas de ADCs (buscando transitorios durante switching de canales), y líneas de reloj digitales (buscando jitter o ringing). Si encuentras ruido de 100 mVpp superpuesto a una señal de sensor de 1V, tu SNR efectivo es solo 20 dB-insuficiente para resolución de 12 bits. Necesitas agregar filtros, mejorar el layout de PCB, o usar blindaje para reducir acoplamientos electromagnéticos.

En sistemas digitales de alta velocidad (buses SPI, I2C, UART conectando sensores a microcontroladores), el osciloscopio te permite verificar que las señales cumplen timing: setup time, hold time, tiempos de rise/fall. Si las señales tienen slew rate demasiado lento (debido a capacitancia parásita o resistencias de pull-up demasiado altas), puedes tener errores de comunicación intermitentes que aparecen solo cuando el bus está ocupado o la temperatura cambia. Estas fallas son imposibles de diagnosticar sin osciloscopio porque el sistema funciona «casi siempre», fallando solo en condiciones de borde que son difíciles de reproducir.

Las mediciones no son solo números en una pantalla; son evidencia física que valida o refuta tus modelos mentales del circuito. Cuando mides un voltaje inesperado, no estás encontrando un «error» del circuito - estás descubriendo que tu modelo del circuito era incompleto o incorrecto. Cuando mides corriente anómala, estás detectando un camino conductivo que no consideraste. Y cuando mides ruido con osciloscopio, estás revelando dinámicas de alta frecuencia que las ecuaciones de DC ignoran completamente. En sistemas de IA embebida, donde la calidad de los datos determina la calidad del modelo entrenado, saber diagnosticar problemas de hardware mediante medición sistemática es tan crítico como saber escribir algoritmos de machine learning. Un modelo entrenado con datos de sensores ruidosos, descalibrados, o con offsets causados por caídas de voltaje en cables aprenderá patrones espurios que no generalizan. Y debuggear un modelo de IA que falla en producción porque los datos de entrenamiento estaban contaminados por problemas de hardware es órdenes de magnitud más difícil que haber detectado esos problemas con un multímetro en 5 minutos durante la integración inicial del sistema.

Esta página se dejó en blanco intencionalmente.

Capítulo 2

Semiconductores y Componentes Activos

2.1 El Diodo: Rectificación y No-Linealidad Fundamental

Objetivo de Aprendizaje

Objetivo: Comprender el comportamiento no-lineal del diodo semiconductor mediante el análisis de su curva característica I-V y aplicaciones en rectificación, **para** diseñar circuitos de acondicionamiento de señal que protejan sensores y sistemas de adquisición de datos contra sobrevoltajes inversos y transitorios.

Ingeniería de Datos desde el Hardware

Los ADCs en sistemas de IA embebida no toleran voltajes negativos en sus entradas. Un sensor que genera señales bipolares ($\pm 5V$) conectado directamente a un ADC que acepta solo 0-3.3V destruirá el chip. Los diodos de protección son la primera línea de defensa contra estas fallas, pero solo si entiendes su curva I-V y sus limitaciones de corriente y velocidad.

El diodo semiconductor es el componente no-lineal más simple y más ubicuo en electrónica. A diferencia de una resistencia, donde el voltaje y la corriente mantienen una relación lineal proporcional (Ley de Ohm), el diodo exhibe una relación exponencial entre voltaje y corriente que lo hace conducir masivamente en una dirección (polarización directa) y bloquear casi completamente en la dirección opuesta (polarización inversa). Esta asimetría fundamental lo convierte en la herramienta básica para rectificación (convertir AC a DC), protección de circuitos (clamp diodes), y manipulación de señales (detección de pico, multiplexación). Pero la no-linealidad del diodo no es una idealización matemática limpia - es un fenómeno físico con consecuencias prácticas que afectan velocidad de conmutación, caídas de

voltaje, generación de calor, y distorsión de señales. Cuando diseñas sistemas de adquisición de datos para IA, cada sensor analógico probablemente tiene diodos de protección en su salida, cada fuente de alimentación tiene diodos de rectificación, y cada bus digital tiene diodos de clamp para suprimir transitorios. Ignorar cómo funcionan realmente estos diodos es diseñar sistemas que fallan de maneras sutiles y difíciles de diagnosticar.

2.1.1 La Curva I-V del Diodo: Exponencial y Asimétrica

La relación entre corriente y voltaje en un diodo semiconductor ideal está gobernada por la ecuación de Shockley: $I = I_S (e^{V_D/(nV_T)} - 1)$, donde I_S es la corriente de saturación inversa (típicamente del orden de picoamperios a nanoamperios), V_D es el voltaje a través del diodo, n es el factor de idealidad (típicamente 1-2), y $V_T = kT/q \approx 26$ mV a temperatura ambiente es el voltaje térmico. Esta ecuación captura la asimetría fundamental: en polarización directa ($V_D > 0$), la exponencial domina y la corriente crece explosivamente; en polarización inversa ($V_D < 0$), la exponencial tiende a cero y la corriente se satura en $-I_S$, que es despreciablemente pequeña.

En la práctica, esto significa que un diodo de silicio típico (como el 1N4148) comienza a conducir apreciablemente cuando el voltaje directo supera aproximadamente 0.6-0.7 V, y a partir de ahí un pequeño incremento de voltaje causa un gran incremento de corriente. Por ejemplo, a 0.65 V el diodo puede conducir 1 mA, pero a 0.75 V (solo 100 mV más) puede conducir 10 mA o más, dependiendo del diodo específico. Esta sensibilidad exponencial hace que el voltaje directo del diodo sea relativamente constante en un rango amplio de corrientes - una propiedad que se explota en reguladores de voltaje de referencia basados en diodos Zener y diodos LED.

La Figura 2.1 muestra la curva I-V característica de un diodo de silicio típico, destacando las regiones de conducción directa, bloqueo inverso, y ruptura por avalancha.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2.1_e1.py`. El script grafica la curva I-V usando la ecuación de Shockley y permite modificar parámetros como la corriente de saturación I_S y el factor de idealidad n .

Experimento numérico: Usando la ecuación de Shockley con $I_S = 1$ nA, $n = 1$, y $V_T = 26$ mV, calcula la corriente del diodo a voltajes de 0.5 V, 0.6 V, 0.7 V, y 0.8 V. Observa cómo la corriente aumenta aproximadamente por un factor de $10\times$ por cada 60 mV de incremento de voltaje (a temperatura ambiente). Esta es la base del comportamiento logarítmico del diodo cuando se usa como detector de señal.

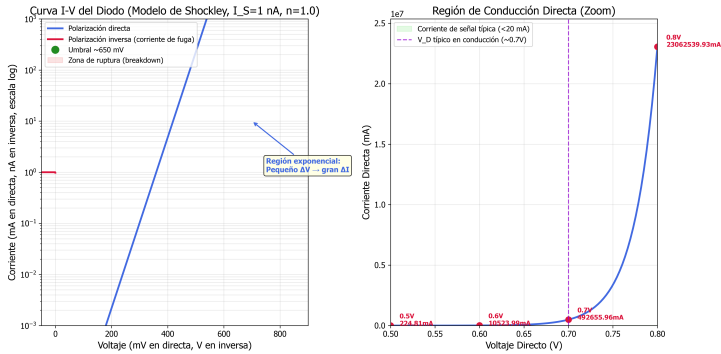


Figura 2.1: Curva característica I-V de un diodo semiconductor. Región I: Bloqueo inverso ($V < 0$), corriente despreciable hasta alcanzar el voltaje de ruptura. Región II: Umbral de conducción directa ($0 < V < 0,6 \text{ V}$), corriente crece exponencialmente. Región III: Conducción plena ($V > 0,7 \text{ V}$), corriente limitada solo por resistencia externa. La no-linealidad extrema significa que pequeños cambios de voltaje causan grandes cambios de corriente. (Figura generada con el script `s2.1_e1.py`)

2.1.2 Rectificación de Media Onda y Onda Completa

La aplicación más básica del diodo es la rectificación: convertir una señal AC (alterna, con voltaje positivo y negativo) en una señal DC pulsante (solo positiva o solo negativa). En un rectificador de media onda, un solo diodo bloquea los semiciclos negativos de la señal AC, dejando pasar solo los semiciclos positivos. El resultado es una señal pulsante a la frecuencia de la AC original (60 Hz en México), con valor promedio distinto de cero. El problema es que estás desperdiciando la mitad de la energía disponible (los semiciclos bloqueados) y la señal rectificada tiene un ripple enorme (oscilación residual) que requiere filtrado masivo.

En un rectificador de onda completa, cuatro diodos configurados en puente (rectificador de puente) o dos diodos con transformador de tap central invierten los semiciclos negativos para que también contribuyan a la salida DC. Esto duplica la frecuencia de ripple (a 120 Hz en línea de 60 Hz) y duplica el voltaje DC promedio, haciendo el filtrado mucho más eficiente. Pero cada diodo introduce una caída de voltaje de aproximadamente 0.7 V en conducción directa, así que un puente de diodos pierde 1.4 V del voltaje de entrada (dos diodos conducen simultáneamente en cualquier semiciclo). En aplicaciones de baja potencia donde cada voltio cuenta, esta caída puede ser inaceptable, motivando el uso de rectificadores síncronos con MOSFETs que tienen caídas de voltaje mucho menores.

La Figura 2.2 muestra las formas de onda de entrada y salida para recti-

ficadores de media onda y onda completa, incluyendo el efecto del filtrado con capacitor.

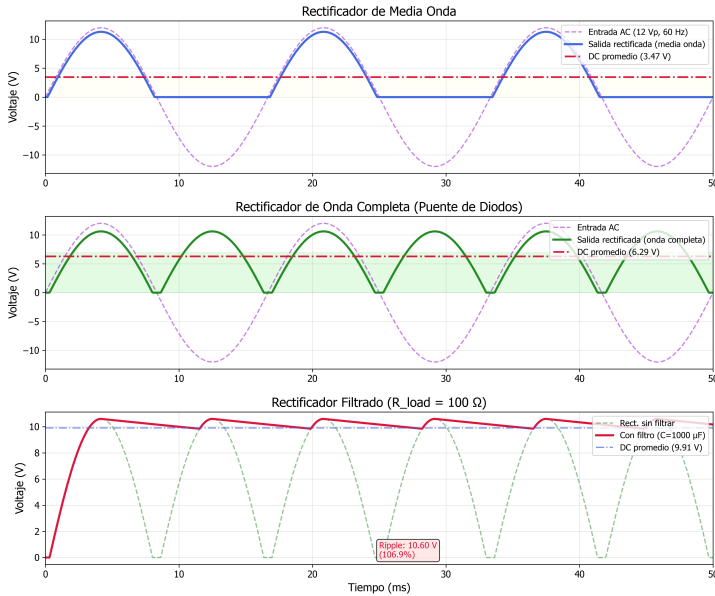


Figura 2.2: Rectificación con diodos. Arriba: rectificador de media onda elimina semiciclos negativos, generando ripple del 100 % a 60 Hz. Medio: rectificador de onda completa invierte semiciclos negativos, generando ripple a 120 Hz con mejor factor de forma. Abajo: filtrado con capacitor reduce ripple pero introduce corrientes de pico elevadas durante la recarga del capacitor. (Figura generada con el script `s2_1.e2.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2_1.e2.py`. Modifica el valor del capacitor de filtrado y observa cómo cambia el ripple en la salida DC.

Diseño práctico: Si necesitas generar 12 VDC con ripple < 100 mV desde un transformador de 12 VAC @ 60 Hz usando un rectificador de onda completa, ¿qué capacitancia mínima necesitas? Usa la aproximación $\Delta V = I_{carga} / (f \cdot C)$, donde $f = 120$ Hz es la frecuencia de ripple. Si la carga consume 500 mA, ¿qué capacitor usarías?

2.1.3 Diodos de Protección y Clamp Circuits

En circuitos de adquisición de datos, los diodos de protección previenen que voltajes fuera de rango destruyan componentes sensibles como ADCs

y microcontroladores. La configuración más común es un par de diodos conectados entre la entrada de señal y los rieles de alimentación (V_{cc} y GND): si la señal intenta ir por encima de $V_{cc} + 0.7$ V, el diodo superior conduce y clampea la señal a $V_{cc} + 0.7$ V. Si la señal intenta ir por debajo de $GND - 0.7$ V, el diodo inferior conduce y clampea la señal a $GND - 0.7$ V.

Estos diodos no son ideales - conducen corriente que debe ser limitada por una resistencia en serie para evitar destruir los diodos mismos. Si conectas una fuente de señal de 10 V directamente a una entrada protegida por diodos diseñados para 3.3 V sin resistencia limitadora, la corriente de clamp puede ser $(10 \text{ V} - 3.3 \text{ V})/R_{\text{diodo}}$ que puede alcanzar varios amperios si R_{diodo} es pequeña, excediendo la capacidad de corriente del diodo y quemándolo.

Otra aplicación crítica de diodos de clamp es en entradas digitales con protección ESD (descarga electrostática). Cuando tocas un pin de microcontrolador después de caminar sobre alfombra en invierno, puedes tener varios kilovoltios de potencial estático que se descargarán en nanosegundos, generando corrientes de pico de cientos de amperios. Los diodos de protección ESD dentro del chip desvían esta corriente hacia los rieles de alimentación, salvando las compuertas lógicas internas. Pero si los rieles de alimentación están desconectados (chip apagado), los diodos no tienen adónde drenar la corriente, y la ESD puede atravesar el óxido de gate del MOSFET, destruyendo permanentemente el dispositivo. Esta es la razón por la cual nunca debes aplicar señales a un chip antes de energizarlo.

La Figura 2.3 muestra esquemas de protección con diodos en entradas analógicas y digitales.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2.1_e3.py`. El script simula un circuito de protección y muestra la corriente de clamp vs voltaje de entrada para diferentes valores de resistencia serie.

Cálculo de resistencia serie: Si proteges una entrada de ADC de 3.3 V con diodos Schottky (caída directa de 0.3 V) y la señal de entrada puede llegar hasta 10 V, ¿qué resistencia serie necesitas para limitar la corriente de clamp a 10 mA (máximo recomendado para diodos pequeños)? Si la señal tiene impedancia de salida de 1 k Ω , ¿cómo afecta esto al diseño?

Los diodos son la puerta de entrada al mundo de los semiconductores no-lineales. Comprender su curva I-V exponencial, sus limitaciones de corriente y voltaje, y sus aplicaciones en rectificación y protección es esencial para diseñar sistemas de adquisición de datos robustos. Cuando conectas sensores del mundo real a ADCs, estás interfazando señales analógicas ruidosas

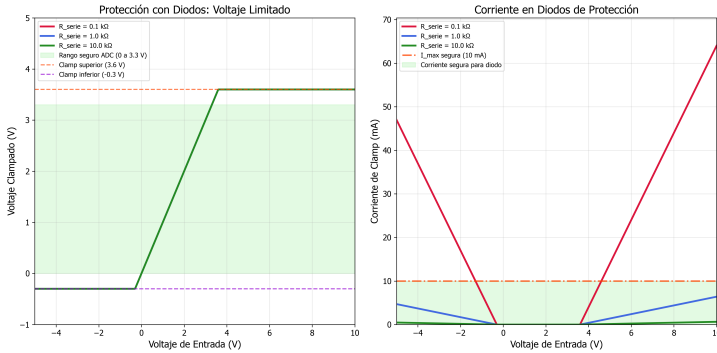


Figura 2.3: Circuitos de protección con diodos. Izquierda: diodos de clamp en entrada analógica limitan voltaje a rango seguro del ADC (0 a V_{ref}). Resistencia serie limita corriente de clamp cuando la señal excede los rieles. Derecha: protección ESD en entrada digital con diodos internos del chip. Si la señal excede V_{cc} o cae debajo de GND, los diodos conducen y disipan la energía. (Figura generada con el script `s2_1_e3.py`)

y potencialmente fuera de rango con circuitos digitales frágiles que se destruyen con sobrevoltajes de microsegundos. Los diodos de protección son tu primera línea de defensa, pero solo si sabes diseñar correctamente las resistencias limitadoras de corriente y entiendes que los diodos tienen capacitancia parásita que limita su velocidad de respuesta a transitorios rápidos. En sistemas de IA donde recoges millones de muestras de sensores y las usas para entrenar modelos, un solo transitorio que sature o dañe el ADC puede contaminar todo un batch de datos con outliers que degradan la precisión del modelo entrenado. Y ningún algoritmo de machine learning puede distinguir datos corruptos por fallas de hardware de datos reales con patrones inusuales sin información adicional sobre la física del sistema de medición.

2.2 El Transistor BJT: Amplificación y Control de Corriente

Objetivo de Aprendizaje

Objetivo: Analizar el funcionamiento del transistor bipolar de juntura (BJT) como dispositivo controlado por corriente en sus tres regiones de operación (corte, activa, saturación), **para** diseñar circuitos amplificadores lineales y switches de potencia que permitan acondicionar señales de sensores de bajo nivel y controlar actuadores desde microcontroladores.

Ingeniería de Datos desde el Hardware

Los sensores de temperatura, presión, y aceleración producen señales del orden de milivolts. Un ADC de 10 bits con rango de 0-3.3V tiene resolución de 3.2 mV por bit, lo que significa que señales de sensores de 10-50 mV se pierden en el ruido de cuantización. Necesitas amplificar estas señales antes de digitalizarlas, y el transistor BJT en configuración de emisor común es el amplificador discreto más simple y ubicuo para esta tarea. Pero la ganancia del BJT depende exponencialmente de la temperatura y de las tolerancias de fabricación, lo que significa que no puedes simplemente conectar un BJT y esperar ganancia predecible sin diseño cuidadoso del punto de operación y polarización.

El transistor bipolar de juntura (BJT, del inglés Bipolar Junction Transistor) es un dispositivo semiconductor de tres terminales que puede amplificar corriente o funcionar como switch controlado por corriente. A diferencia del diodo que solo tiene dos regiones de operación (conducción y bloqueo), el BJT tiene tres regiones fundamentales: corte (ambas junturas en inversa, transistor apagado), activa (juntura base-emisor en directa, juntura base-colector en inversa, operación lineal para amplificación), y saturación (ambas junturas en directa, transistor completamente encendido como switch cerrado). La región activa es donde el BJT exhibe su propiedad más poderosa: la corriente de colector I_C es proporcional a la corriente de base I_B multiplicada por un factor β (ganancia de corriente DC) que típicamente está entre 50 y 300 dependiendo del transistor específico. Esta relación $I_C = \beta I_B$ permite que una pequeña corriente de base controle una corriente de colector mucho mayor, lo que es la esencia de la amplificación de corriente. Pero esta relación solo se mantiene mientras el transistor permanece en región activa-cuando la corriente de colector es tan grande que el voltaje

colector-emisor cae por debajo de aproximadamente 0.2-0.3 V, el transistor entra en saturación y la ganancia β colapsa. Entender las curvas características del BJT y cómo diseñar el punto de operación (DC bias point) para mantener el transistor en región activa durante toda la excursión de la señal es fundamental para diseñar amplificadores que no distorsionen.

2.2.1 Curvas Características y Regiones de Operación

Las curvas características del BJT grafican corriente de colector I_C vs voltaje colector-emisor V_{CE} para diferentes valores fijos de corriente de base I_B . Cada curva corresponde a un valor específico de I_B , y la familia de curvas muestra cómo la corriente de colector aumenta casi linealmente con I_B en región activa, mientras que se satura (se vuelve plana, horizontal) cuando V_{CE} es suficientemente grande. La pendiente de estas curvas en región activa no es exactamente cero (como sería en un transistor ideal) sino que tiene una ligera inclinación positiva debido al efecto Early, que modela la modulación del ancho de la base por el voltaje colector-emisor. Este efecto causa que la ganancia de corriente efectiva aumente ligeramente con V_{CE} , lo que se manifiesta como una resistencia de salida finita en el modelo de pequeña señal del transistor.

En región de corte ($I_B = 0$ o $V_{BE} < 0,6$ V para silicio), el transistor está apagado y la corriente de colector es despreciable (solo corriente de fuga, del orden de nanoamperios). En región activa ($V_{BE} \approx 0,7$ V, $V_{CE} > V_{CE,sat} \approx 0,2$ V), el transistor amplifica: $I_C = \beta I_B$, y el voltaje V_{BE} es relativamente constante (0.7 V para silicio, similar a un diodo en conducción). En región de saturación ($V_{BE} \approx 0,8 - 0,9$ V, $V_{CE} < 0,2$ V), el transistor está completamente encendido como switch cerrado, y la relación $I_C = \beta I_B$ ya no se cumple-la corriente de colector está limitada por la resistencia externa en el circuito, no por la corriente de base. Cuando diseñas un amplificador, quieres operar en región activa. Cuando diseñas un switch digital para controlar un LED o un relé, quieres operar en saturación (para minimizar la caída de voltaje V_{CE} y por lo tanto la disipación de potencia en el transistor).

La Figura 2.4 muestra las curvas características de un BJT NPN típico (como el 2N3904), destacando las tres regiones de operación.

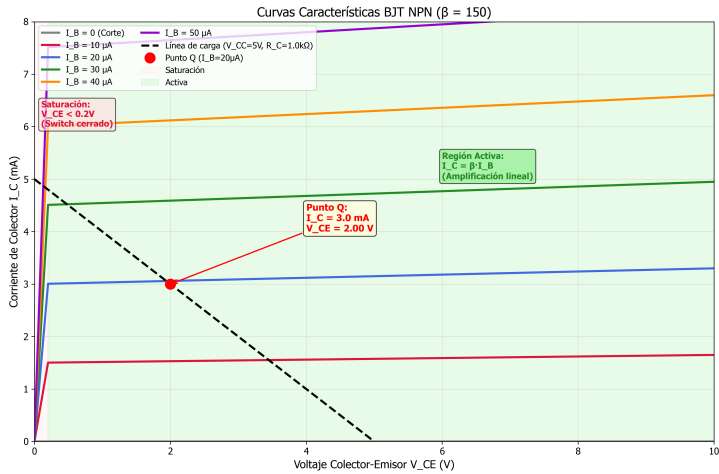


Figura 2.4: Curvas características de un transistor BJT NPN. Cada curva corresponde a una corriente de base constante. Región I: Corte ($I_B = 0$), transistor apagado. Región II: Activa (líneas casi horizontales para $V_{CE} > 0,2$ V), $I_C = \beta I_B$ con $\beta \approx 100 - 200$. Región III: Saturación ($V_{CE} < 0,2$ V), transistor completamente encendido como switch. La línea de carga superpuesta muestra los puntos de operación posibles para un circuito dado. (Figura generada con el script `s2_2.e1.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2_2.e1.py`. El script grafica las curvas I_C vs V_{CE} para diferentes corrientes de base y superpone una línea de carga para un circuito resistivo típico.

Experimento numérico: Usando las curvas características con $\beta = 150$, determina el punto de operación (Q-point) para un circuito con $V_{CC} = 5$ V, $R_C = 1$ k Ω , e $I_B = 20$ μ A. Calcula I_C , V_{CE} , y verifica que el transistor está en región activa. ¿Qué sucede si aumentas I_B a 40 μ A? ¿El transistor entra en saturación?

2.2.2 Amplificador en Configuración de Emisor Común

La configuración de emisor común es la topología de amplificador más fundamental con BJT: la señal de entrada se aplica a la base, la salida se toma del colector, y el emisor está conectado a tierra (común a entrada y salida). Esta configuración proporciona ganancia de voltaje y ganancia de corriente simultáneamente, lo que la hace ideal para amplificar señales de sensores débiles. La ganancia de voltaje de pequeña señal en emisor común

está dada aproximadamente por $A_v = -g_m R_C$, donde $g_m = I_C/V_T$ es la transconductancia del transistor ($V_T \approx 26$ mV a temperatura ambiente) y R_C es la resistencia de colector. El signo negativo indica que la salida está invertida respecto a la entrada (cuando la entrada aumenta, la salida disminuye). La ganancia de corriente es simplemente β , y la ganancia de potencia es el producto $A_v \times \beta$.

El problema crítico del amplificador de emisor común es que la ganancia A_v depende de I_C , que a su vez depende exponencialmente de V_{BE} , que a su vez depende de la temperatura (V_{BE} disminuye aproximadamente 2 mV por cada grado Celsius de aumento de temperatura). Esto significa que si polarizas el transistor aplicando un voltaje fijo a la base, el punto de operación se moverá drásticamente con la temperatura, causando distorsión o incluso empujando al transistor fuera de región activa. La solución estándar es usar un divisor resistivo de voltaje en la base combinado con una resistencia de emisor R_E que proporciona retroalimentación negativa: cuando la temperatura aumenta y V_{BE} disminuye, I_E tiende a aumentar, lo que aumenta la caída de voltaje en R_E , lo que reduce el voltaje efectivo aplicado a la base, estabilizando I_C . Esta técnica de polarización por divisor de voltaje con resistencia de emisor es el caballo de batalla de los amplificadores discretos de BJT, y es lo que permite diseños robustos que funcionan sobre un rango de temperatura y tolerancias de componentes.

La Figura 2.5 muestra un amplificador de emisor común con polarización estabilizada y las formas de onda de entrada y salida.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2_2.e2.py`. El script simula un amplificador de emisor común y muestra las formas de onda de entrada y salida, destacando la inversión de fase y la ganancia.

Diseño práctico: Diseña un amplificador de emisor común para amplificar la señal de un sensor de temperatura (salida de 10 mV a 50 mV) a un rango de 0.5 V a 2.5 V adecuado para un ADC de 3.3 V. Asume $V_{CC} = 5$ V, $\beta = 150$, y establece el punto Q en $I_C = 1$ mA, $V_{CE} = 2.5$ V. Calcula R_C , R_E , y las resistencias del divisor de base (R_1 , R_2) asumiendo que la corriente del divisor es 10 veces mayor que I_B para minimizar el efecto de carga.

2.2.3 BJT como Switch: Saturación Profunda y Tiempos de Conmutación

Cuando usas un BJT como switch digital para controlar cargas como LEDs, relés, o motores pequeños desde un microcontrolador, quieres operar en dos estados: corte (transistor completamente apagado, $I_C \approx 0$) y saturación

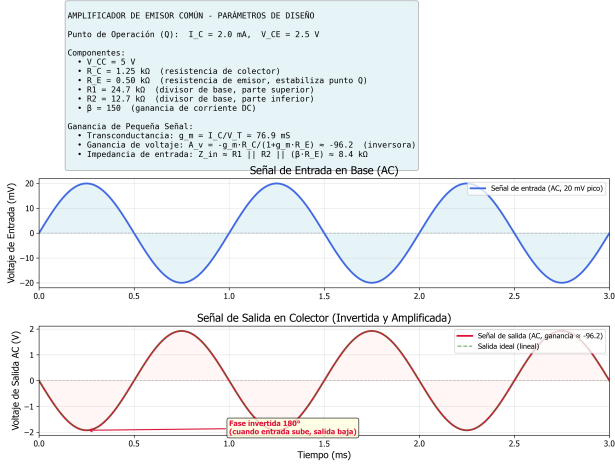


Figura 2.5: Amplificador de emisor común con polarización por divisor de voltaje. Arriba: esquema del circuito mostrando resistencias de polarización (R_1 , R_2), resistencia de colector (R_C), y resistencia de emisor (R_E) con capacitor de bypass (C_E) para AC. Abajo: formas de onda mostrando inversión de fase entre entrada (base) y salida (colector), y ganancia de voltaje. La región lineal se limita por el rango de voltajes entre corte y saturación. (Figura generada con el script `s2.2_e2.py`)

profunda (transistor completamente encendido, $V_{CE} \approx 0.2 \text{ V}$). Para asegurar saturación profunda, debes proporcionar suficiente corriente de base para que $I_B > I_{C,sat} / \beta_{forced}$, donde β_{forced} (ganancia forzada) es típicamente mucho menor que β en región activa (típicamente 10-20 en saturación profunda). En la práctica, esto significa que si quieres conmutar 100 mA de corriente de colector con un 2N3904 ($\beta = 150$ típico), necesitas proporcionar al menos $I_B = 100 \text{ mA} / 10 = 10 \text{ mA}$ de corriente de base para asegurar saturación, no los $100 \text{ mA} / 150 = 0.67 \text{ mA}$ que calcularías usando β de región activa.

El problema con usar BJTs como switches es la velocidad de conmutación. Cuando el transistor está en saturación profunda, hay un exceso de portadores minoritarios almacenados en la base que deben ser removidos antes de que el transistor pueda apagarse. Este tiempo de almacenamiento (storage time) es del orden de microsegundos y limita la frecuencia máxima de conmutación a decenas o cientos de kHz dependiendo del transistor específico. Para aplicaciones de switching de alta frecuencia (cientos de kHz o MHz), los MOSFETs son superiores porque no almacenan carga en la misma manera y pueden conmutar mucho más rápido. Pero para switching

de baja frecuencia (menos de 100 kHz) donde costo y simplicidad importan, los BJTs siguen siendo ampliamente usados, especialmente en configuraciones como el Darlington que proporciona muy alta ganancia de corriente para controlar cargas grandes desde señales de microcontrolador de baja corriente.

La Figura 2.6 muestra un BJT configurado como switch para controlar un LED desde un pin de microcontrolador, incluyendo las formas de onda de voltaje de base y colector durante la conmutación.

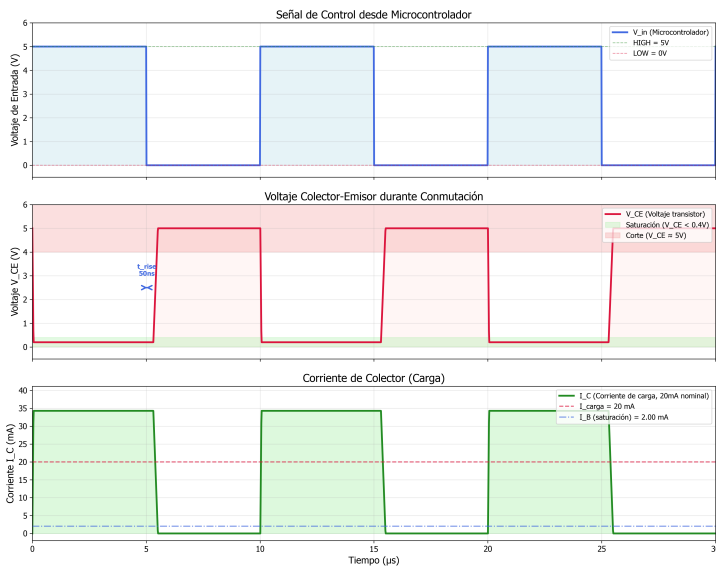


Figura 2.6: BJT como switch controlado por microcontrolador. Izquierda: circuito mostrando resistencia de base para limitar I_B y diodo flyback para proteger contra picos inductivos si la carga es un relé o motor. Derecha: formas de onda mostrando transición entre corte y saturación. Cuando V_{in} (señal del microcontrolador) sube a HIGH, el transistor entra en saturación y V_{CE} cae a 0.2 V, encendiendo la carga. El tiempo de conmutación incluye tiempo de subida (rise time) y tiempo de almacenamiento (storage time) durante el apagado. (Figura generada con el script `s2_2.e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2_2_e3.py`. El script simula un BJT como switch y muestra las formas de onda de conmutación, incluyendo los tiempos de retardo y almacenamiento.

Cálculo de resistencia de base: Quieres controlar un LED de 20 mA desde un pin de Arduino (5 V) usando un 2N3904. El LED tiene caída de voltaje de 2 V. Calcula la resistencia de base necesaria para asegurar saturación profunda ($\beta_{forced} = 10$). Si el pin del Arduino solo puede proporcionar 20 mA máximo, ¿es viable este diseño? Si no, ¿qué modificación harías?

El transistor BJT es el amplificador de corriente fundamental y el precursor de toda la electrónica moderna de semiconductores. Aunque los circuitos integrados modernos usan mayormente MOSFETs por su menor consumo de potencia y mejor escalabilidad a dimensiones nanométricas, los BJTs siguen siendo críticos en aplicaciones de alta frecuencia (amplificadores de RF), alta corriente (drivers de potencia), y bajo ruido (amplificadores de entrada en instrumentación de precisión). Cuando diseñas un sistema de adquisición de datos para IA, es probable que tengas un amplificador operacional (que internamente es una configuración compleja de BJTs o MOSFETs) como primera etapa después del sensor, pero entender cómo funciona un amplificador de emisor común discreto te da intuición sobre ganancia, impedancia de entrada/salida, ancho de banda, ruido, y distorsión que aplica directamente al diseño de sistemas completos. Y cuando debuggeas por qué tu sensor de temperatura tiene offset que deriva con temperatura ambiente, o por qué tu ADC se satura inesperadamente cuando la señal excede cierto nivel, la comprensión profunda de las curvas características del BJT, sus regiones de operación, y sus limitaciones térmicas y de frecuencia es lo que te permite diagnosticar y corregir el problema en lugar de solo ajustar parámetros aleatoriamente hasta que funcione por casualidad.

2.3 El Transistor MOSFET: Switching Rápido y Baja Potencia

Objetivo de Aprendizaje

Objetivo: Comprender el funcionamiento del transistor MOSFET como dispositivo controlado por voltaje en sus regiones de operación (corte, óhmica, saturación), **para** diseñar circuitos de switching de alta eficiencia en fuentes conmutadas, drivers de motores, y sistemas de control de potencia donde la velocidad de conmutación y el bajo consumo son críticos.

Ingeniería de Datos desde el Hardware

Un microcontrolador moderno consume 50-100 mA a 3.3V durante operación activa, pero solo 1-10 μA en modo de bajo consumo (sleep). Esta diferencia de cuatro órdenes de magnitud es posible porque los circuitos digitales CMOS (Complementary Metal-Oxide-Semiconductor) usan MOSFETs que no consumen corriente estática - solo consumen energía durante las transiciones de switching. En contraste, los circuitos equivalentes con BJTs consumirían corriente de base continuamente, haciendo inviable la operación con batería. Cuando diseñas sistemas de IA embebida que deben operar meses o años con batería, entender por qué los MOSFETs dominan la electrónica digital y de potencia moderna es fundamental.

El transistor de efecto de campo metal-óxido-semiconductor (MOSFET) es el dispositivo semiconductor más ubicuo en electrónica moderna. A diferencia del BJT que se controla por corriente de base, el MOSFET se controla por voltaje de gate, y teóricamente no consume corriente de gate en DC (solo corriente de fuga del orden de nanoamperios). Esta característica de control por voltaje sin consumo de corriente lo hace ideal para circuitos digitales donde miles de millones de transistores pueden conmutar sin generar corrientes de polarización masivas, y para switching de potencia donde el driver puede ser un simple pin de microcontrolador sin necesidad de amplificación de corriente. El MOSFET tiene tres regiones de operación análogas al BJT pero con terminología diferente: corte (gate-source voltage V_{GS} por debajo del voltaje umbral V_{th} , transistor apagado), región óhmica o lineal (transistor encendido como resistencia variable, V_{DS} pequeño), y región de saturación (transistor encendido con corriente constante, V_{DS} grande). Confusamente, la «saturación» del MOSFET corresponde a la región «activa» del BJT (amplificación lineal), mientras que la región «óhmica» del MOSFET

corresponde a la «saturación» del BJT (switch cerrado). Esta inversión de terminología es fuente de confusión perpetua, pero la física subyacente es clara: en región óhmica el MOSFET se comporta como una resistencia controlada por voltaje ($R_{DS,on}$), y en región de saturación la corriente de drain es proporcional al cuadrado del voltaje gate-source.

2.3.1 Curvas Características y Resistencia On-State

Las curvas características del MOSFET grafican corriente de drain I_D vs voltaje drain-source V_{DS} para diferentes valores de voltaje gate-source V_{GS} . Para $V_{GS} < V_{th}$, el transistor está en corte y $I_D \approx 0$. Para $V_{GS} > V_{th}$, el MOSFET conduce, y la relación entre I_D y V_{DS} tiene dos regímenes: en región óhmica ($V_{DS} < V_{GS} - V_{th}$), la corriente crece linealmente con V_{DS} como una resistencia $R_{DS} = V_{DS}/I_D$ que disminuye con V_{GS} ; en región de saturación ($V_{DS} > V_{GS} - V_{th}$), la corriente se satura (se vuelve plana, horizontal) y está dada aproximadamente por $I_D = \frac{1}{2}k_n(V_{GS} - V_{th})^2$, donde k_n es el parámetro de transconductancia que depende de las dimensiones del transistor y de la movilidad de portadores en el canal.

Para aplicaciones de switching de potencia (convertidores DC-DC, inversores, drivers de motor), quieres operar el MOSFET en región óhmica como switch cerrado con resistencia $R_{DS,on}$ lo más baja posible para minimizar pérdidas de conducción ($P = I^2 R_{DS,on}$). Los MOSFETs de potencia modernos tienen $R_{DS,on}$ del orden de miliohmios para corrientes de decenas de amperios, lo que genera caídas de voltaje de solo decenas de milivolts incluso a corrientes muy altas. En contraste, un BJT en saturación tiene $V_{CE,sat} \approx 0,2 - 0,3$ V independiente de la corriente, lo que genera pérdidas mucho mayores a altas corrientes. Esta es la razón por la cual los MOSFETs han reemplazado completamente a los BJTs en switching de potencia moderno.

La Figura 2.7 muestra las curvas características de un MOSFET de canal N (como el IRLZ44N), destacando las regiones de operación y la resistencia $R_{DS,on}$.

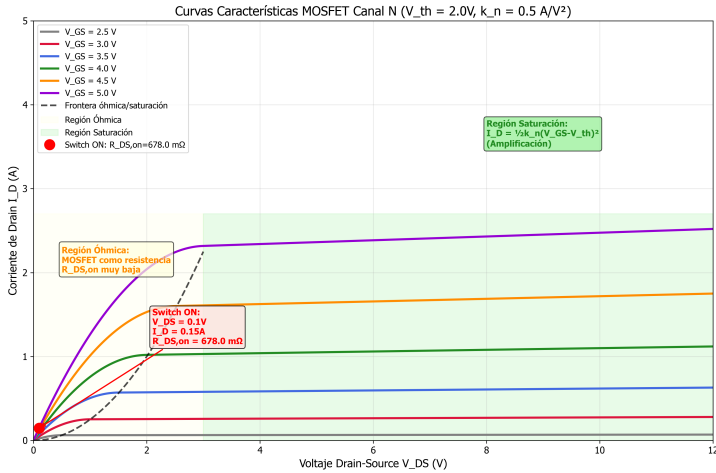


Figura 2.7: Curvas características de un MOSFET de canal N. Cada curva corresponde a un voltaje gate-source constante. Región I: Corte ($V_{GS} < V_{th}$), transistor apagado. Región II: Óhmica (V_{DS} pequeño), transistor como resistencia variable $R_{DS,on}$. Región III: Saturación (V_{DS} grande), corriente constante $I_D \propto (V_{GS} - V_{th})^2$. Para switching, operar en región óhmica minimiza pérdidas. Para amplificación, operar en saturación (análogo a activa en BJT). (Figura generada con el script `s2.3_e1.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2.3_e1.py`. El script grafica las curvas I_D vs V_{DS} para diferentes voltajes de gate y muestra la transición entre región óhmica y saturación.

Experimento numérico: Usando las curvas características con $V_{th} = 2 V$ y $k_n = 0,5 A/V^2$, calcula la corriente de drain para $V_{GS} = 5 V$ y $V_{DS} = 0,1 V$ (región óhmica). Luego calcula para $V_{GS} = 5 V$ y $V_{DS} = 10 V$ (región de saturación). Compara las corrientes y explica por qué en región óhmica la resistencia efectiva $R_{DS} = V_{DS}/I_D$ es baja.

2.3.2 MOSFET como Switch de Potencia: PWM y Control de Velocidad

La aplicación más común del MOSFET en sistemas embebidos es como switch de potencia controlado por PWM (modulación de ancho de pulso) para controlar la velocidad de motores DC, el brillo de LEDs de alta potencia, y el voltaje de salida en convertidores DC-DC. En esta configu-

ración, el gate del MOSFET se conecta a un pin de microcontrolador a través de una resistencia de gate opcional (para limitar corriente de pico durante switching), el source se conecta a GND, y el drain se conecta a la carga y a la alimentación a través de una resistencia o inductor. Cuando el microcontrolador aplica HIGH (3.3V o 5V) al gate, V_{GS} supera V_{th} y el MOSFET enciende, conectando la carga a GND. Cuando aplica LOW (0V), el MOSFET apaga y la carga queda desconectada. Variando el duty cycle de la señal PWM (porcentaje de tiempo en HIGH vs LOW), controlas la potencia promedio entregada a la carga.

El problema crítico del switching con MOSFET es que la velocidad de conmutación está limitada por la capacitancia de gate C_{GS} . Aunque el MOSFET no consume corriente DC de gate, durante las transiciones de switching debes cargar y descargar esta capacitancia (típicamente decenas a cientos de picofaradios para MOSFETs pequeños, nanofaradios para MOSFETs de potencia). La corriente de pico durante switching es $I_{peak} = C_{GS} \cdot dV/dt$, y si el microcontrolador no puede proporcionar suficiente corriente, el MOSFET conmutará lentamente, pasando tiempo significativo en la región lineal donde tanto V_{DS} como I_D son no - cero simultáneamente, causando disipación de potencia $P = V_{DS} \times I_D$ que puede destruir el transistor por sobrecalentamiento. Para switching de alta frecuencia (> 100 kHz) o MOSFETs de potencia con alta C_{GS} , necesitas un driver de gate dedicado que pueda proporcionar corrientes de pico de varios amperios para cargar/descargar rápidamente la capacitancia de gate.

La Figura 2.8 muestra un MOSFET controlando un motor DC mediante PWM, incluyendo las formas de onda de gate, drain, y corriente de motor.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2.3_e2.py`. El script simula control PWM de un motor DC y muestra cómo el duty cycle afecta la potencia entregada.

Diseño práctico: Quieres controlar un motor de 12V / 2A desde un Arduino (5V) usando un IRLZ44N (MOSFET lógico, $V_{th} = 1 - 2V$, $R_{DS,on} = 0,022 \Omega @ V_{GS} = 5V$). Calcula la caída de voltaje en el MOSFET cuando el motor consume 2A. Calcula la potencia disipada en el MOSFET. ¿Necesitas disipador térmico? (Asume resistencia térmica junction-to-ambient de $62^\circ C/W$ sin disipador).

2.3.3 Comparación BJT vs MOSFET y Selección de Tecnología

La decisión entre usar BJT o MOSFET en un diseño depende de varios factores: control por corriente vs voltaje, velocidad de switching, resistencia/voltaje de saturación, costo, disponibilidad, y complejidad del circuito

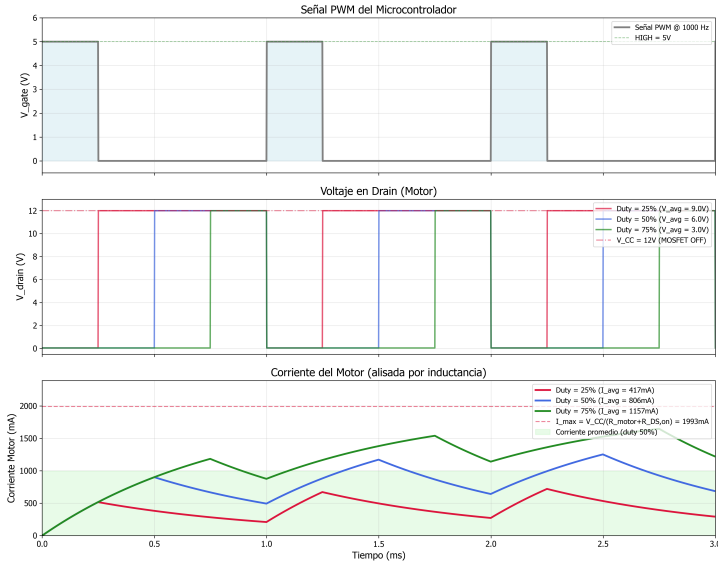


Figura 2.8: Control de motor DC con MOSFET mediante PWM. Arriba: circuito mostrando MOSFET de canal N, diodo flyback para protección contra picos inductivos del motor, y resistencia de gate. Medio: señal PWM del microcontrolador (duty cycle variable). Abajo: voltaje en drain y corriente de motor. Cuando PWM = HIGH, MOSFET enciende y motor recibe V_{CC} . Cuando PWM = LOW, MOSFET apaga y corriente del motor circula por diodo flyback. El duty cycle determina velocidad promedio del motor. (Figura generada con el script `s2_3_e2.py`)

de drive. Los BJTs requieren corriente de base continua para mantenerse encendidos, lo que puede ser problemático si el driver tiene capacidad de corriente limitada (como pines de microcontrolador limitados a 20-40 mA). Los MOSFETs solo requieren voltaje de gate (sin corriente DC), pero tienen capacitancia de gate que debe ser cargada rápidamente para switching veloz, requiriendo corrientes de pico altas durante transiciones. Los BJTs tienen $V_{CE,sat}$ relativamente alto (0.2-0.3V) independiente de corriente, mientras que los MOSFETs tienen $V_{DS} = I_D \times R_{DS,on}$ que escala con corriente pero puede ser mucho menor que 0.2V si $R_{DS,on}$ es suficientemente baja. Los BJTs son generalmente más baratos para potencias bajas (< 1W), mientras que los MOSFETs dominan en switching de potencia media y alta (> 1W) debido a su mayor eficiencia.

Para amplificación de señales analógicas de baja frecuencia (< 100 kHz), los BJTs siguen siendo preferidos por su mayor transconductancia (ganancia) y menor ruido a bajas frecuencias. Para switching digital y control de potencia, los MOSFETs son la opción estándar por su control simple por voltaje,

baja resistencia on-state, y capacidad de switching a altas frecuencias (> 1 MHz). En circuitos integrados digitales (microcontroladores, FPGAs, memorias), los MOSFETs CMOS dominan completamente porque permiten densidades de integración masivas (miles de millones de transistores por chip) con consumo de potencia razonable, algo imposible con tecnología BJT.

La Figura 2.9 compara las características de switching de BJT y MOSFET lado a lado, mostrando diferencias en tiempos de conmutación, caídas de voltaje, y requisitos de drive.

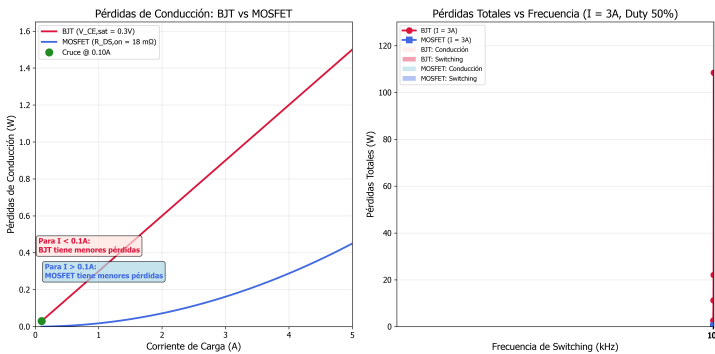


Figura 2.9: Comparación BJT vs MOSFET como switches de potencia. Izquierda: BJT requiere corriente de base continua ($I_B > I_C/\beta_{forced}$), tiene $V_{CE,sat} \approx 0,2 - 0,3V$, y tiempo de almacenamiento limita velocidad de apagado. Derecha: MOSFET requiere solo voltaje de gate (sin corriente DC), tiene $V_{DS} = I_D \cdot R_{DS,on}$ (puede ser $< 0,1V$), pero capacitancia de gate requiere corrientes de pico altas para switching rápido. Para potencias $> 1W$ y frecuencias > 100 kHz, MOSFET es superior. (Figura generada con el script `s2_3_e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2_3_e3.py`. El script compara pérdidas de potencia en BJT vs MOSFET para diferentes corrientes de carga y frecuencias de switching.

Análisis de eficiencia: Para un switch de 12V / 5A operando a 50 kHz (PWM para control de motor), compara las pérdidas de conducción y switching entre un BJT (2N3055, $V_{CE,sat} = 0,3V$, $t_{storage} = 2 \mu s$) y un MOSFET (IRFZ44N, $R_{DS,on} = 0,018 \Omega$, $Q_g = 72$ nC). Asume duty cycle de 50%. ¿Qué dispositivo es más eficiente? ¿Cómo cambia la respuesta si la frecuencia aumenta a 500 kHz?

El MOSFET es el transistor dominante en electrónica moderna, no porque

sea universalmente superior al BJT en todas las métricas, sino porque sus características de control por voltaje sin consumo de corriente DC y baja resistencia on-state lo hacen ideal para las dos aplicaciones que dominan el mercado de semiconductores: circuitos digitales CMOS (procesadores, memorias, FPGAs) y switching de potencia (fuentes conmutadas, inversores, drivers de motor). Cuando diseñas sistemas de IA embebida, prácticamente todo el circuito digital está construido con MOSFETs CMOS a nivel de transistor, y cualquier control de potencia (reguladores DC-DC para diferentes rieles de voltaje, backlight de display, motores de actuadores) usará MOSFETs de potencia. Pero entender las limitaciones del MOSFET - especialmente la capacitancia de gate que requiere corrientes de pico para switching rápido, y la dependencia de $R_{DS,on}$ con temperatura (aumenta con temperatura, opuesto al BJT)-es crítico para evitar fallas térmicas y problemas de integridad de señal en diseños de alta frecuencia. Y cuando debuggeas por qué tu motor se sobrecalienta o por qué tu convertidor DC-DC genera ruido que corrompe las mediciones de tu ADC, la comprensión profunda de las curvas características del MOSFET, sus tiempos de switching, y sus modos de falla es lo que te permite diagnosticar el problema raíz en lugar de solo aplicar «fixes»empíricos sin entender por qué funcionan.

2.4 El Amplificador Operacional: Bloque Universal de Procesamiento Analógico

Objetivo de Aprendizaje

Objetivo: **Analizar** el amplificador operacional como bloque de construcción universal para circuitos de acondicionamiento de señal, aplicando las reglas del amplificador ideal y retroalimentación negativa, **para** diseñar circuitos de ganancia programable, filtros activos, y convertidores de señal que preparen las salidas de sensores para digitalización en sistemas de adquisición de datos.

Ingeniería de Datos desde el Hardware

Un acelerómetro MEMS típico genera señales de $\pm 1.5g$ como voltajes de salida de $1.5V \pm 300mV$. Un ADC de 10 bits con rango de 0-3.3V desperdicia la mitad de su rango dinámico (todos los bits por debajo de 1.2V y por encima de 1.8V nunca se usan), reduciendo la resolución efectiva a aproximadamente 8 bits. Necesitas restar el offset de 1.5V (centering), amplificar la señal de $\pm 300mV$ a $\pm 1.65V$ (ganancia de $5.5\times$), y sumar un offset de 1.65V para llevar la señal de salida de 0-3.3V (level shifting). Estas tres operaciones-suma, amplificación, y resta-son exactamente las operaciones que los amplificadores operacionales realizan con precisión, linealidad, y bajo ruido cuando se configuran en las topologías estándar que dominas en esta sección.

El amplificador operacional (op-amp) es probablemente el circuito integrado analógico más importante en electrónica práctica. Es un amplificador diferencial de alta ganancia (típicamente $100,000\times$ o más) con entrada diferencial de alta impedancia y salida de baja impedancia que puede proporcionar corriente significativa a una carga. El poder del op-amp no viene de usarlo en lazo abierto (sin retroalimentación), donde su ganancia masiva y falta de linealidad lo hacen inestable e impredecible, sino de configurarlo con retroalimentación negativa, donde la ganancia efectiva del circuito está determinada solo por las resistencias externas y es independiente de la ganancia intrínseca del op-amp (siempre que esta sea suficientemente alta). Esta independencia de parámetros internos del dispositivo es lo que hace que los circuitos con op-amp sean robustos, predecibles, y fáciles de diseñar: calculas la ganancia deseada, seleccionas resistencias según la fórmula de retroalimentación negativa, y el circuito funciona según lo calculado sin necesidad de ajustar o trimear componentes para compensar variaciones de fabricación. Las configuraciones fundamentales del op-amp-amplificador inversor, no-inversor, seguidor de voltaje, sumador, restador, integrador, diferenciador-son los bloques de construcción que combinamos para implementar cualquier función de procesamiento analógico: acondicionamiento de sensores, filtrado, generación de formas de onda, conversión de señal, y control en lazo cerrado.

2.4.1 El Amplificador Ideal y las Reglas de Oro

El modelo del amplificador operacional ideal captura las propiedades esenciales del dispositivo real en la mayoría de las aplicaciones: ganancia de lazo abierto infinita ($A_{OL} \rightarrow \infty$), impedancia de entrada infinita (las terminales + y - no consumen corriente), impedancia de salida cero (la salida puede proporcionar cualquier corriente sin caída de voltaje), ancho de banda infi-

nito (respuesta instantánea a cambios de entrada), y voltaje de offset cero (cuando las entradas son iguales, la salida es cero). Aunque ningún op-amp real cumple exactamente estas condiciones, los dispositivos modernos se aproximan suficientemente para que el modelo ideal sea útil en análisis de primer orden. El LM358 (op-amp de propósito general, muy barato) tiene ganancia de lazo abierto de $100,000\times$, impedancia de entrada de $1\text{ M}\Omega$ (no infinita, pero mucho mayor que las resistencias típicas del circuito), e impedancia de salida de $75\ \Omega$ (no cero, pero mucho menor que cargas típicas de $\text{k}\Omega$). Dispositivos más sofisticados como el TL072 (JFET input) o el OPA2134 (audio precision) tienen impedancias de entrada de $\text{T}\Omega$ (teraohms) y distorsión armónica total de 0.0001% , acercándose aún más al ideal.

Las dos reglas de oro del análisis de op-amp con retroalimentación negativa simplifican radicalmente el análisis de circuitos: (1) *La diferencia de voltaje entre las terminales + y - es cero* ($V_+ = V_-$), porque el op-amp ajusta su salida para forzar esta condición cuando hay retroalimentación negativa. (2) *Las terminales + y - no consumen corriente* ($I_+ = I_- = 0$), porque la impedancia de entrada es (casi) infinita. Estas dos reglas permiten analizar cualquier configuración estándar de op-amp escribiendo ecuaciones de nodos en las terminales + y - sin necesidad de conocer la ganancia interna del op-amp o sus parámetros detallados. La ganancia del circuito completo emerge simplemente de las relaciones entre las resistencias de retroalimentación y las resistencias de entrada.

La Figura 2.10 muestra las configuraciones básicas de amplificador inversor y no-inversor con sus fórmulas de ganancia derivadas de las reglas de oro.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2.4_e1.py`. El script simula las configuraciones básicas de op-amp y muestra las relaciones entrada-salida para diferentes valores de resistencias de retroalimentación.

Diseño práctico: Diseña un amplificador inversor con ganancia de $-10\times$ para amplificar la salida de un sensor de temperatura ($0\text{--}100\text{ mV}$) a un rango de $0\text{--}1\text{ V}$ adecuado para un ADC. Selecciona R_{in} y R_f apropiadas. ¿Por qué no conviene usar $R_{in} = 1\ \Omega$ y $R_f = 10\ \Omega$? (Pista: piensa en corrientes y disipación de potencia). ¿Por qué no conviene usar $R_{in} = 10\text{ M}\Omega$ y $R_f = 100\text{ M}\Omega$? (Pista: piensa en corrientes de bias del op-amp y ruido).

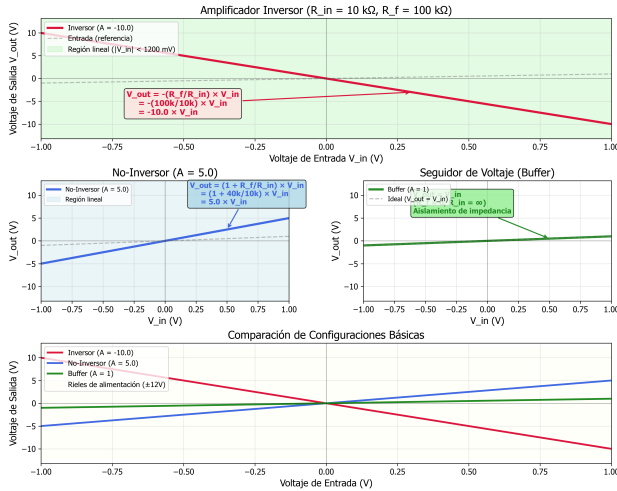


Figura 2.10: Configuraciones fundamentales del op-amp. Izquierda: Amplificador inversor ($V_{out} = -(R_f/R_{in}) \cdot V_{in}$), ganancia determinada por relación de resistencias, entrada virtual ground. Derecha: Amplificador no-inversor ($V_{out} = (1 + R_f/R_{in}) \cdot V_{in}$), ganancia siempre ≥ 1 , alta impedancia de entrada. Centro inferior: Seguidor de voltaje (buffer, ganancia = 1), aísla fuente de carga sin amplificación. Las fórmulas son independientes de parámetros del op-amp gracias a retroalimentación negativa. (Figura generada con el script `s2_4_e1.py`)

2.4.2 Circuitos de Acondicionamiento: Sumador, Restador, e Instrumentación

Los circuitos de acondicionamiento de señal combinan múltiples entradas con ganancias programables y offsets para transformar señales de sensores en formatos compatibles con ADCs. El amplificador sumador usa múltiples resistencias de entrada conectadas al nodo de suma (terminal inversora) para implementar $V_{out} = -(R_f/R_1)V_1 - (R_f/R_2)V_2 - \dots$, permitiendo mezclar señales con pesos controlados. El amplificador restador (diferencial) implementa $V_{out} = (R_f/R_{in})(V_2 - V_1)$, rechazando señales de modo común (ruido que afecta ambas entradas por igual) mientras amplifica la diferencia. Esta configuración es fundamental para sensores de puente (strain gauges, termopares, celdas de carga) donde la señal útil es la diferencia entre dos voltajes que flotan sobre un voltaje de modo común grande.

El amplificador de instrumentación es una configuración especializada de

tres op-amps que proporciona ganancia diferencial programable con impedancia de entrada extremadamente alta en ambas entradas (a diferencia del restador simple que carga las fuentes de entrada de manera asimétrica). Es el circuito estándar para amplificar señales de sensores de puente de alta impedancia en instrumentación de precisión. La ganancia se ajusta con una sola resistencia externa, y el rechazo de modo común (CMRR, Common-Mode Rejection Ratio) puede superar 100 dB en dispositivos integrados especializados como el INA128 o AD620, lo que significa que señales de ruido de 1 V de modo común se atenúan a $10 \mu\text{V}$ mientras que señales diferenciales de 1 mV se amplifican sin degradación.

La Figura 2.11 muestra circuitos de sumador, restador, y amplificador de instrumentación con análisis de ganancia y ejemplos de aplicación.

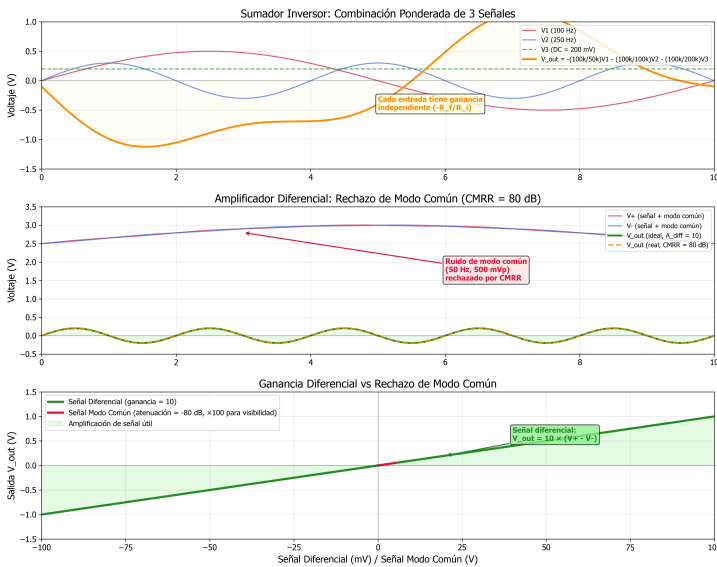


Figura 2.11: Circuitos de acondicionamiento de señal con op-amps. Arriba izquierda: Sumador inversor combina múltiples señales con pesos independientes, útil para mixing de audio y offset adjustment. Arriba derecha: Restador (diferencial) amplifica diferencia entre dos entradas, rechaza ruido de modo común, crítico para sensores de puente. Abajo: Amplificador de instrumentación (3 op-amps) proporciona alta impedancia de entrada en ambas terminales y ganancia ajustable con una resistencia, estándar en instrumentación biomédica y sensores de precisión. (Figura generada con el script `s2_4_e2.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2_4_e2.py`. El script simula circuitos de sumador y restador, mostrando cómo se combinan señales y cómo se rechaza ruido de modo común.

Aplicación de sensor de puente: Un strain gauge en configuración de puente de Wheatstone genera señales diferenciales de ± 10 mV sobre un voltaje de modo común de 2.5 V (mitad del voltaje de excitación de 5 V). Diseña un amplificador diferencial con ganancia de $100\times$ para llevar la señal de salida a ± 1 V. Calcula las resistencias necesarias. Si el op-amp tiene CMRR de 80 dB, ¿cuánto voltaje de modo común aparece en la salida?

2.4.3 Filtros Activos y Limitaciones de Frecuencia

Los op-amps permiten implementar filtros activos (pasa-bajas, pasa-altas, pasa-banda, rechaza-banda) sin necesidad de inductores, que son grandes, caros, y no-ideales (tienen resistencia parásita significativa). Un filtro pasa-bajas de primer orden se implementa simplemente agregando un capacitor en paralelo a la resistencia de retroalimentación de un amplificador inversor, creando un polo en frecuencia $f_c = 1/(2\pi R_f C_f)$. Para frecuencias mucho menores que f_c , el capacitor es un circuito abierto y la ganancia es $-R_f/R_{in}$ (como amplificador inversor DC). Para frecuencias mucho mayores que f_c , el capacitor es un cortocircuito y la ganancia cae a cero con pendiente de -20 dB/década. Filtros de mayor orden (segundo, tercero, etc.) con respuestas más agudas (Butterworth, Chebyshev, Bessel) se implementan cascadeando etapas de op-amp o usando topologías especializadas como Sallen-Key o multiple feedback.

Pero los op-amps reales no tienen ancho de banda infinito. El producto ganancia-ancho de banda (GBW, Gain-Bandwidth Product) es una figura de mérito clave: un LM358 tiene GBW de 1 MHz, lo que significa que a ganancia unitaria ($\times 1$) el ancho de banda es 1 MHz, pero a ganancia de $\times 100$ el ancho de banda cae a 10 kHz. Si intentas amplificar señales de 100 kHz con ganancia de $100\times$ usando un LM358, la ganancia real será mucho menor de lo calculado porque estás operando más allá del límite de GBW. Para aplicaciones de audio (20 Hz - 20 kHz), el LM358 es adecuado. Para acondicionamiento de sensores ultrasónicos (40 kHz - 200 kHz) o video (hasta varios MHz), necesitas op-amps de mayor velocidad como TL072 (GBW = 3 MHz) o LM6172 (GBW = 100 MHz). El slew rate (tasa de cambio máxima de voltaje de salida, en V/ μ s) es otra limitación crítica: el LM358 tiene slew rate de 0.5 V/ μ s, lo que significa que para una señal de salida de 10 Vpp a 10 kHz (que requiere $dV/dt = 2\pi f V_{pk} \approx 0.31$ V/ μ s), el op-amp puede seguir la señal, pero a 100 kHz (requiere 3.1 V/ μ s) la salida

se distorsionará en forma triangular (slew-rate limiting).

La Figura 2.12 muestra implementaciones de filtros pasa-bajas y pasa-altas con op-amp, incluyendo respuestas en frecuencia y limitaciones de GBW.

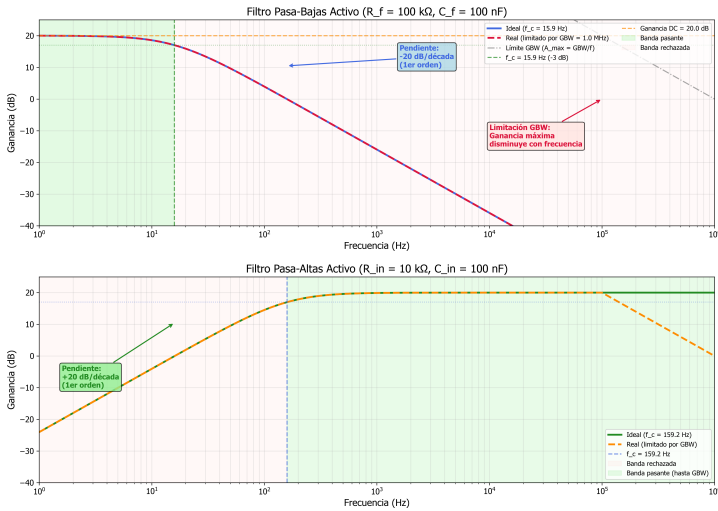


Figura 2.12: Filtros activos con op-amps. Izquierda: Filtro pasa-bajas de primer orden (RC en retroalimentación), frecuencia de corte $f_c = 1/(2\pi R_f C_f)$, pendiente de atenuación -20 dB/década. Derecha: Filtro pasa-altas (RC en entrada), atenúa frecuencias bajas, permite frecuencias altas. Centro: Respuesta en frecuencia mostrando ganancia en banda pasante, frecuencia de corte, y roll-off. La línea punteada muestra limitación por GBW del op-amp real: ganancia máxima alcanzable disminuye con frecuencia según $A_{max} = GBW/f$. (Figura generada con el script `s2_4_e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s2_4_e3.py`. El script grafica respuestas en frecuencia de filtros activos y muestra el efecto de la limitación de GBW en la ganancia alcanzable.

Diseño de filtro anti-aliasing: Necesitas digitalizar señales de audio (0-20 kHz) con un ADC que muestrea a 44.1 kHz (estándar CD). Diseña un filtro pasa-bajas de segundo orden con frecuencia de corte de 20 kHz para evitar aliasing. Si usas un LM358 (GBW = 1 MHz), ¿puedes lograr ganancia de $\times 10$ en la banda pasante? Si no, ¿qué op-amp usarías?

Este ejercicio es análogo a diseñar la pipeline de preprocesamiento de datos en un sistema de machine learning: así como normalizas y escalas features de entrada (min-max scaling, z-score normalization) para que caigan en rangos óptimos para el algoritmo de aprendizaje, aquí escalas y filtras señales analógicas para que caigan en rangos óptimos para el ADC, maximizando SNR y resolución efectiva. En ambos casos, la calidad del preprocessing determina directamente la calidad de los resultados downstream-basura entra, basura sale (GIGO), ya sea en datos de sensores o en datasets de entrenamiento.

El amplificador operacional es el componente que cierra la brecha entre el mundo analógico continuo de sensores y actuadores y el mundo digital discreto de microcontroladores y procesamiento. Casi todos los sistemas de adquisición de datos tienen al menos una etapa de op-amp entre el sensor y el ADC: amplificación para maximizar el uso del rango dinámico del ADC, filtrado para prevenir aliasing y rechazar ruido fuera de banda, buffering para aislar la alta impedancia del sensor de la carga capacitiva del ADC, y level shifting para adaptar señales bipolares o con offset a los rangos unipolares de ADCs comunes. Cuando diseñas sistemas de IA embebida que dependen de datos de sensores de alta calidad-acelerómetros para detección de gestos, micrófonos para reconocimiento de voz, sensores de corriente para monitoreo de energía-la cadena de acondicionamiento analógico basada en op-amps determina el SNR, la linealidad, el ancho de banda, y la estabilidad térmica de las mediciones, que a su vez determinan el límite fundamental de accuracy alcanzable por cualquier modelo de machine learning entrenado con esos datos. Ningún algoritmo puede recuperar información destruida por ruido, saturación, o aliasing en la etapa de adquisición analógica. Por eso, entender profundamente cómo diseñar circuitos de op-amp robustos y de bajo ruido no es una habilidad opcional para ingenieros de IA embebida-es un prerequisite para construir sistemas que funcionen fuera del laboratorio en entornos ruidosos, con variaciones de temperatura, y alimentación imperfecta.

Esta página se dejó en blanco intencionalmente.

Capítulo 3

Interfaz Digital-Analógico

3.1 Conversión Analógico-Digital: Cuantización y Muestreo

Objetivo de Aprendizaje

Objetivo: Comprender los principios fundamentales de conversión analógico-digital (muestreo, cuantización, y codificación), las limitaciones impuestas por resolución y frecuencia de muestreo, **para** diseñar sistemas de adquisición de datos que maximicen la calidad de información capturada de sensores analógicos mientras respetan las restricciones de ancho de banda, ruido, y aliasing en aplicaciones de IA embebida.

Ingeniería de Datos desde el Hardware

Un acelerómetro que muestrea a 100 Hz puede capturar vibraciones de hasta 50 Hz (teorema de Nyquist). Si un motor genera vibraciones a 75 Hz y muestreas a 100 Hz, la señal aliaseada aparecerá como una vibración falsa de 25 Hz en tus datos, contaminando cualquier modelo de machine learning entrenado para detectar patrones de falla mecánica. El modelo aprenderá a reconocer un artefacto de tu sistema de medición, no el fenómeno físico real. No hay algoritmo de post-procesamiento que pueda recuperar la frecuencia original una vez que el aliasing ha ocurrido - la información se destruye irreversiblemente en el momento del muestreo. Por eso, entender los límites fundamentales de conversión ADC y diseñar filtros anti-aliasing apropiados no es opcional cuando construyes sistemas de IA que toman decisiones basadas en datos de sensores.

La conversión analógico-digital (ADC, Analog-to-Digital Conversion) es el proceso de transformar una señal continua en tiempo y amplitud (voltaje analógico del mundo real) en una secuencia discreta de números binarios que un microcontrolador o procesador digital puede almacenar, transmitir, y procesar. Este proceso tiene tres pasos fundamentales: muestreo (sam-

pling), cuantización (quantization), y codificación (encoding). El muestreo captura el valor instantáneo de la señal analógica a intervalos regulares determinados por la frecuencia de muestreo f_s . La cuantización redondea cada muestra a uno de 2^N niveles discretos permitidos, donde N es el número de bits del ADC. La codificación asigna un código binario de N bits a cada nivel cuantizado. Cada uno de estos pasos introduce limitaciones y distorsiones que afectan la fidelidad de la representación digital: el muestreo introduce aliasing si la frecuencia de muestreo es insuficiente, la cuantización introduce ruido de cuantización proporcional a $1/2^N$, y la codificación puede introducir errores de linealidad si el ADC real no es ideal. Cuando diseñas un sistema de adquisición de datos para IA, debes balancear resolución (número de bits), velocidad de muestreo (samples por segundo), costo (ADCs de alta resolución y alta velocidad son caros), consumo de potencia (ADCs rápidos consumen más corriente), y complejidad del acondicionamiento analógico (filtros anti-aliasing, amplificación, buffering).

3.1.1 Resolución, Cuantización, y Ruido

La resolución de un ADC se especifica en bits y determina cuántos niveles discretos puede representar. Un ADC de 8 bits tiene $2^8 = 256$ niveles, un ADC de 10 bits tiene $2^{10} = 1024$ niveles, un ADC de 12 bits tiene $2^{12} = 4096$ niveles, y así sucesivamente. La resolución en voltaje (tamaño del paso de cuantización o LSB, Least Significant Bit) es $Q = V_{ref}/2^N$, donde V_{ref} es el voltaje de referencia del ADC. Por ejemplo, un ADC de 10 bits con $V_{ref} = 3,3$ V tiene resolución de $Q = 3,3/1024 \approx 3,22$ mV por bit. Esto significa que cambios de voltaje menores a 3.22 mV son invisibles para el ADC - dos voltajes de entrada que difieren por menos de un LSB se codifican al mismo valor digital. Este efecto de redondeo introduce ruido de cuantización, que tiene características estadísticas de ruido blanco uniforme con potencia $\sigma_q^2 = Q^2/12$ si la señal de entrada tiene suficiente variabilidad (dithering natural). El SNR (relación señal-ruido) teórico de un ADC ideal de N bits es aproximadamente $SNR_{dB} \approx 6,02N + 1,76$ dB, lo que significa que cada bit adicional de resolución mejora el SNR en aproximadamente 6 dB (factor de 2 en potencia).

Pero la resolución nominal en bits no cuenta toda la historia. Los ADCs reales tienen errores de no-linealidad diferencial (DNL, Differential Non-Linearity) y no-linealidad integral (INL, Integral Non-Linearity) que causan que algunos códigos sean más anchos o estrechos que el LSB ideal, y que la función de transferencia real se desvíe de la línea ideal. Un ADC de 12 bits con INL de ± 4 LSB tiene resolución efectiva de aproximadamente 10 bits en el peor caso. Además, el ruido del circuito de acondicionamiento analógico (ruido térmico de resistencias, ruido de op-amps, ruido de switching de fuentes conmutadas) se suma al ruido de cuantización y puede dominar el error total si no se diseña cuidadosamente. Un ADC de 16 bits con

resolución teórica de $3,3V/65536 \approx 50 \mu V$ por bit es inútil si el ruido del circuito de entrada es de varios milivolts RMS - estás digitalizando ruido con alta precisión, no señal útil.

La Figura 3.1 muestra el proceso de cuantización, el error de cuantización resultante, y cómo la resolución en bits afecta la representación digital de una señal analógica.

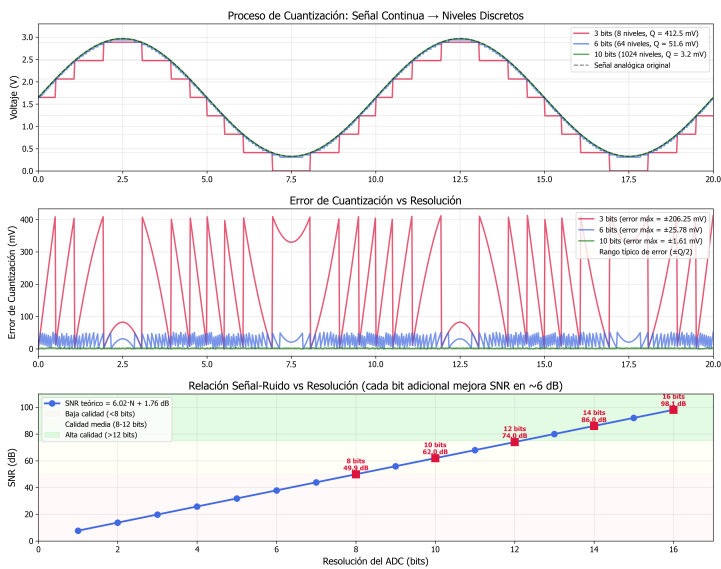


Figura 3.1: Proceso de cuantización en ADC. Arriba: señal analógica continua y niveles de cuantización para ADC de 3 bits (8 niveles). La señal analógica se redondea al nivel más cercano. Medio: señal cuantizada (escala) mostrando pérdida de información entre niveles. Abajo: error de cuantización (diferencia entre señal original y cuantizada), limitado a $\pm Q/2$ donde Q es el tamaño del LSB. El error tiene distribución uniforme y actúa como ruido añadido a la señal. Comparación entre 3, 6, y 10 bits muestra reducción dramática del error con mayor resolución. (Figura generada con el script `s3_1.e1.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s3_1_e1.py`. El script simula el proceso de cuantización para diferentes resoluciones de ADC y muestra el error de cuantización resultante.

Experimento numérico: Una señal de sensor tiene amplitud de 1 Vpp y ruido de 10 mV RMS. Si usas un ADC de 8 bits con rango de 0-3.3 V, ¿cuál es el LSB? ¿Cuántos LSBs de ruido hay en la señal? Si aumentas a 12 bits, ¿mejora significativamente el SNR o el ruido del sensor sigue dominando? Calcula el ENOB (Effective Number of Bits) considerando que $\text{SNR}_{\text{total}} = \text{SNR}_{\text{cuantización}} + \text{SNR}_{\text{ruido_sensor}}$ (en escala lineal, no dB).

3.1.2 Muestreo y Teorema de Nyquist

El teorema de muestreo de Nyquist-Shannon establece que para reconstruir perfectamente una señal continua desde sus muestras discretas, la frecuencia de muestreo f_s debe ser al menos el doble de la máxima frecuencia contenida en la señal: $f_s > 2f_{\text{max}}$. La frecuencia $f_N = f_s/2$ se llama frecuencia de Nyquist y representa el límite superior de frecuencias que pueden ser representadas sin ambigüedad con una frecuencia de muestreo dada. Si la señal contiene componentes de frecuencia por encima de f_N , estos componentes se «doblan» (fold back) al espectro base y aparecen como frecuencias falsas (alias) que contaminan las mediciones. Por ejemplo, si muestreas a 1 kHz ($f_N = 500$ Hz) una señal que contiene un componente de 700 Hz, este componente aparecerá como 300 Hz en los datos digitalizados (la diferencia $f_s - f_{\text{señal}} = 1000 - 700 = 300$ Hz). Este efecto de aliasing es irreversible - una vez que has muestreado la señal, no hay forma de distinguir entre un componente real de 300 Hz y un alias de 700 Hz, porque ambos generan exactamente la misma secuencia de muestras.

La solución es usar un filtro anti-aliasing analógico antes del ADC: un filtro pasa-bajas que atenúa todas las frecuencias por encima de f_N antes de que la señal sea muestreada. Este filtro debe ser suficientemente agudo para proporcionar atenuación de al menos $6N$ dB (donde N es el número de bits del ADC) en la frecuencia de Nyquist, de manera que las componentes aliaseadas queden por debajo del ruido de cuantización. Para un ADC de 12 bits, necesitas al menos 72 dB de atenuación, lo que típicamente requiere un filtro de 4to orden o superior (cada orden adicional contribuye aproximadamente 20 dB/década de roll-off). Filtros Butterworth, Chebyshev, o elípticos se usan dependiendo de si priorizas respuesta plana en banda pasante, roll-off agudo, o fase lineal. En aplicaciones de audio, los filtros Bessel son preferidos por su respuesta de fase lineal que minimiza distorsión de forma de onda.

La Figura 3.2 muestra el efecto del muestreo en el dominio del tiempo y frecuencia, incluyendo ejemplos de aliasing cuando la frecuencia de muestreo es insuficiente.

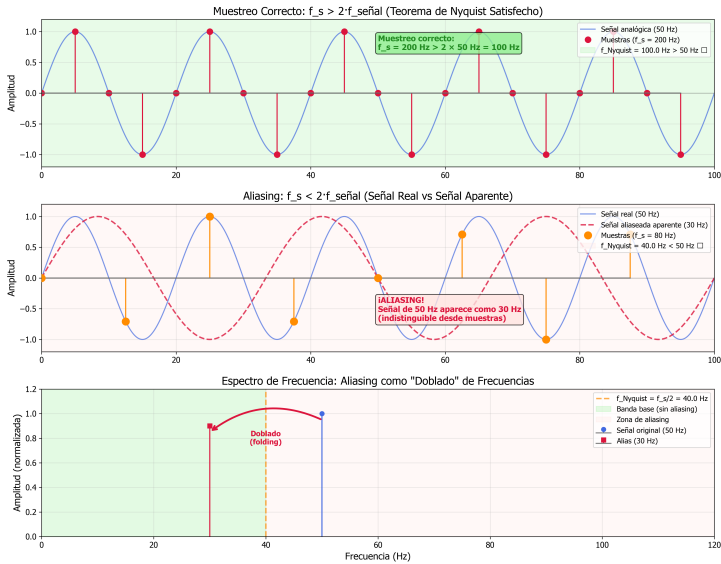


Figura 3.2: Muestreo y aliasing. Izquierda: señal continua de 50 Hz muestreada a 200 Hz ($f_s > 2f_{\text{señal}}$, correcto) reproduce fielmente la señal original. Derecha: misma señal de 50 Hz muestreada a 80 Hz ($f_s < 2f_{\text{señal}}$, insuficiente) produce alias de 30 Hz, señal falsa que no existe en la entrada. Abajo: espectro de frecuencia mostrando cómo componentes por encima de frecuencia de Nyquist se «doblan» al espectro base, contaminando mediciones. Filtro anti-aliasing atenúa estas componentes antes del muestreo. (Figura generada con el script `s3_1_e2.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s3_1_e2.py`. El script demuestra aliasing en el dominio del tiempo para diferentes relaciones entre frecuencia de señal y frecuencia de muestreo.

Análisis de aliasing: Un sensor de vibración puede capturar frecuencias de 0-500 Hz. Si muestras a 800 Hz, ¿qué frecuencias se aliasen? Si una vibración de 600 Hz está presente, ¿a qué frecuencia aparecerá en los datos digitalizados? Diseña un filtro anti-aliasing pasa-bajas con frecuencia de corte de 350 Hz (margen de seguridad) usando una topología Sallen-Key de 2do orden. ¿Qué atenuación proporciona este filtro a 400 Hz (frecuencia de Nyquist)?

3.1.3 Arquitecturas de ADC: SAR, Sigma-Delta, y Flash

Existen múltiples arquitecturas de ADC, cada una con trade-offs específicos entre velocidad, resolución, consumo de potencia, y costo. Los ADCs de aproximaciones sucesivas (SAR, Successive Approximation Register) son los más comunes en microcontroladores embebidos porque ofrecen buen balance entre velocidad (decenas a millones de muestras por segundo), resolución (8-16 bits típico), y consumo de potencia (micro a milivatios). El SAR funciona mediante búsqueda binaria: compara la entrada con $V_{ref}/2$, luego con $V_{ref}/2 \pm V_{ref}/4$, y así sucesivamente durante N ciclos para un ADC de N bits. Cada microcontrolador típico (Arduino, STM32, ESP32) tiene ADCs tipo SAR integrados con 10-12 bits de resolución y velocidades de hasta 1 MSPS (mega-samples por segundo).

Los ADCs sigma-delta ($\Sigma\Delta$) usan sobremuestreo (muestran a frecuencias mucho mayores que la frecuencia de Nyquist de la señal) combinado con filtrado digital para lograr resoluciones muy altas (16-24 bits) a velocidades relativamente bajas (decenas a miles de SPS). Son ideales para instrumentación de precisión (básculas, termómetros de alta resolución, medidores de corriente) donde la resolución y linealidad son críticas pero la velocidad no. Los ADCs flash son los más rápidos (gigamuestras por segundo) pero tienen resolución limitada (6-8 bits típico) y alto consumo de potencia, usándose en osciloscopios digitales, receptores de radio de alta velocidad, y captura de señales de video.

La Figura 3.3 compara las características de diferentes arquitecturas de ADC en términos de velocidad, resolución, y consumo de potencia.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s3_1_e3.py`. El script grafica el espacio de diseño de diferentes arquitecturas de ADC en función de resolución vs velocidad.

Selección de ADC: Para cada aplicación, selecciona la arquitectura más apropiada y justifica: (a) Monitoreo de temperatura de CPU con actualización cada 100 ms, precisión de 0.1°C en rango de $0\text{-}100^\circ\text{C}$. (b) Adquisición de audio de calidad CD (44.1 kHz, 16 bits). (c) Osciloscopio digital para señales de 100 MHz. (d) Acelerómetro para detección de gestos, rango $\pm 2g$, 100 Hz de ancho de banda. Para cada caso, especifica resolución mínima en bits y frecuencia de muestreo necesaria.

La conversión ADC es el cuello de botella fundamental entre el mundo analógico continuo y el procesamiento digital discreto. Todas las mediciones que alimentan modelos de IA embebida-temperatura, aceleración, audio, voltaje de baterías, corriente de motores-pasan por ADCs, y las limitaciones de resolución, ruido, linealidad, y aliasing de estos ADCs definen el límite teórico superior de accuracy alcanzable por cualquier algoritmo

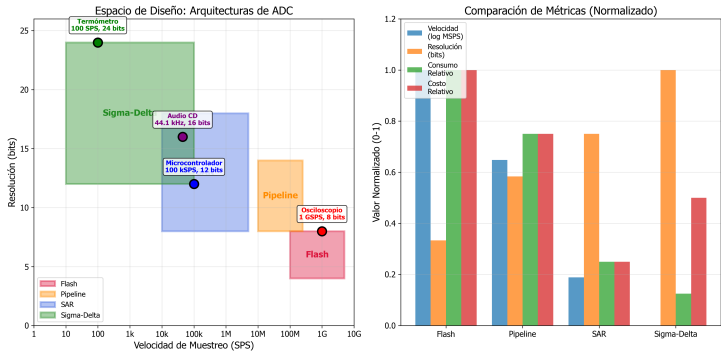


Figura 3.3: Arquitecturas de ADC y sus trade-offs. SAR: balance entre velocidad (10 kSPS - 5 MSPS) y resolución (8-16 bits), bajo consumo, ideal para aplicaciones generales en microcontroladores. Sigma-Delta: alta resolución (16-24 bits) mediante sobremuestreo, baja velocidad (10-100 kSPS), ideal para instrumentación de precisión. Flash: máxima velocidad (> 100 MSPS) pero baja resolución (6-8 bits) y alto consumo, usado en osciloscopios y comunicaciones de alta velocidad. Pipeline y SAR asíncrono son intermedios. Selección depende de requisitos de aplicación específica. (Figura generada con el script `s3.1_e3.py`)

downstream. Un modelo de machine learning entrenado para clasificar gestos desde un acelerómetro de 8 bits nunca podrá igualar la performance de un modelo entrenado con datos de 16 bits, porque literalmente hay menos información en los datos de entrada. Y un sistema de monitoreo de vibración que aliasa frecuencias críticas debido a muestreo insuficiente generará falsos positivos y negativos sin importar cuán sofisticado sea el algoritmo de detección de anomalías. Por eso, el diseño cuidadoso de la cadena de adquisición analógica-acondionamiento de señal, filtrado anti-aliasing, selección de ADC apropiado, y configuración de frecuencia de muestreo - no es una tarea secundaria delegable a «los de hardware», es una responsabilidad central de cualquier ingeniero que construye sistemas de IA que interactúan con el mundo físico.

3.2 Conversión Digital-Analógico: Reconstrucción y Síntesis de Señales

Objetivo de Aprendizaje

Objetivo: Analizar el proceso de conversión digital-analógico y las técnicas de reconstrucción de señales continuas desde datos discretos, **para** diseñar sistemas de generación de formas de onda, control analógico, y síntesis de audio que transformen comandos digitales de microcontroladores en voltajes y corrientes que actúan sobre el mundo físico.

Ingeniería de Datos desde el Hardware

Un sistema de control de temperatura con PID digital calcula una salida de control numérica (0-100 % de potencia de calefacción) cada 10 ms. Pero el elemento calefactor requiere un voltaje analógico continuo de 0-10 V para modular su potencia. El DAC es el puente que convierte la secuencia discreta de números de 0-4095 (ADC de 12 bits) en voltajes analógicos continuos que el driver de potencia puede usar para controlar la resistencia calefactora. Si el DAC tiene baja resolución (8 bits), la temperatura oscilará visiblemente porque el control solo puede tomar 256 valores discretos. Si el DAC es lento (actualización a 10 Hz), el sistema no puede responder rápidamente a perturbaciones. Y si el DAC tiene glitches de switching, transitorios de voltaje durante la actualización corromperán la señal de control, causando inestabilidad. Entender las limitaciones de conversión DAC es tan crítico como entender ADC cuando diseñas sistemas de control en lazo cerrado.

La conversión digital-analógico (DAC, Digital-to-Analog Conversion) es el proceso inverso a ADC: toma una secuencia de números binarios y genera voltajes o corrientes analógicas proporcionales. Mientras que el ADC es pasivo (observa el mundo analógico), el DAC es activo (actúa sobre el mundo analógico). Aplicaciones de DAC incluyen generación de formas de onda arbitrarias (AWG, Arbitrary Waveform Generators) para pruebas y caracterización de circuitos, síntesis de audio digital, control analógico de actuadores (motores, válvulas, calefactores), y calibración de sistemas de medición. Los DACs se especifican por resolución (bits), velocidad de actualización (samples por segundo), tiempo de establecimiento (settling time, cuánto tarda la salida en estabilizarse después de un cambio de código), glitches de switching (transitorios de voltaje durante actualizaciones), y li-

nealidad (DNL/INL, análogo a ADCs). Los DACs de alta velocidad (cientos de MSPS) se usan en transmisores de radio y generadores de señales de RF. Los DACs de alta resolución (16-24 bits) se usan en audio de calidad estudio y sistemas de control de precisión. Los DACs integrados en microcontroladores típicamente tienen resolución modesta (8-12 bits) y velocidades de decenas a cientos de kSPS, adecuados para control de propósito general pero insuficientes para aplicaciones de audio o RF de alta fidelidad.

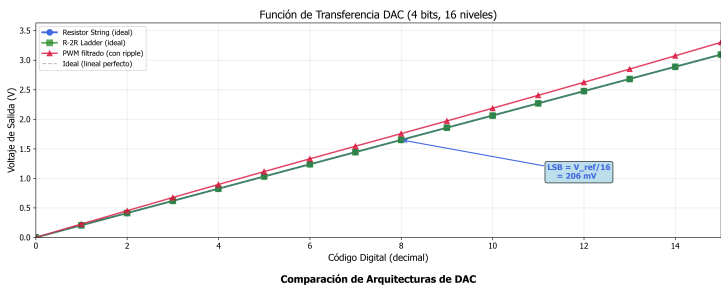
3.2.1 Arquitecturas de DAC: Resistor String, R-2R, y PWM

La arquitectura más simple de DAC es el divisor resistivo (resistor string): 2^N resistencias idénticas conectadas en serie entre V_{ref} y GND crean 2^N niveles de voltaje igualmente espaciados. Un multiplexor selecciona el nivel correspondiente al código binario de entrada. Esta arquitectura es intrínsecamente monótona (no hay glitches) y tiene excelente linealidad, pero el número de resistencias crece exponencialmente con la resolución, haciéndola impráctica para más de 8-10 bits. Se usa en DACs flash de baja resolución y alta velocidad.

La arquitectura R-2R usa una red en escalera de resistencias con solo dos valores (R y 2R) para crear una red de división de corriente que combina ponderadamente los bits de entrada. Esta topología requiere solo $2N$ resistencias para un DAC de N bits y es la más común en DACs integrados de resolución media (10-16 bits). La precisión depende del matching entre resistencias: un error de 0.1 % en una resistencia introduce error de DNL. DACs de precisión usan resistencias laser-trimmed en el proceso de fabricación para minimizar mismatch.

Para microcontroladores sin DAC dedicado, la técnica más común es usar PWM (Pulse Width Modulation) filtrado. El microcontrolador genera una señal PWM de alta frecuencia (decenas a cientos de kHz) con duty cycle proporcional al valor digital deseado. Un filtro pasa-bajas RC extrae el valor DC promedio, que es proporcional al duty cycle: $V_{out} = V_{CC} \times D$, donde D es el duty cycle (0-1). La resolución está determinada por la resolución del timer PWM (típicamente 8-16 bits). El ancho de banda está limitado por la frecuencia de corte del filtro: si necesitas actualizar la salida a 1 kHz, el filtro debe tener $f_c > 1$ kHz, lo que requiere frecuencia PWM $\gg 10$ kHz para evitar ripple residual en la salida. Esta técnica es barata (sin componentes adicionales excepto R y C) pero lenta y con ripple residual que puede ser problemático en aplicaciones de bajo ruido.

La Figura 3.4 muestra esquemas de diferentes arquitecturas de DAC y sus características comparativas.



Característica	Resistor String	R-2R Ladder	PWM Filtrado
Resolución práctica	4-8 bits	8-18 bits	8-16 bits (timer)
Velocidad	Alta (MSPS)	Media-Alta (KSPS-MSPS)	Baja (1Hz-KHz)
Monotonía	Garantizada	Depende matching	Garantizada
Complejidad	2^N resistencias	2N resistencias	R + C + software
Costo	Alto (>8 bits)	Medio	Muy bajo

Figura 3.4: Arquitecturas de DAC. Arriba izquierda: Resistor string (divisor resistivo) usa 2^N resistencias para N bits, monótono y lineal pero exponencialmente complejo. Arriba derecha: R-2R ladder usa red en escalera con solo 2 valores de resistencia, común en DACs integrados de 10-16 bits. Abajo: PWM filtrado genera voltaje analógico desde duty cycle, barato pero con ripple y ancho de banda limitado. Tabla comparativa muestra trade-offs entre resolución, velocidad, complejidad, y costo. (Figura generada con el script `s3_2_e1.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s3_2_e1.py`. El script simula las arquitecturas de DAC y muestra las salidas analógicas generadas para diferentes códigos digitales de entrada.

Diseño de DAC con PWM: Necesitas generar un voltaje de control de 0-3.3 V con resolución de 8 bits usando PWM de un Arduino (5V). El voltaje debe poder actualizarse a 100 Hz sin ripple visible. Diseña el filtro RC: elige frecuencia PWM de 10 kHz (100× mayor que tasa de actualización), calcula R y C para frecuencia de corte de 500 Hz (margen entre actualización y PWM). ¿Cuánto ripple residual esperas en la salida?

3.2.2 Reconstrucción de Señales y Glitches

Cuando un DAC actualiza su salida desde un código digital al siguiente, la transición no es instantánea. Durante el tiempo de establecimiento (settling

time), la salida puede tener sobrepicos, oscilaciones, o glitches (picos breves de voltaje) antes de estabilizarse en el valor final. Los glitches ocurren porque diferentes bits del código digital no conmutan simultáneamente-hay skew temporal entre bits que causa que el DAC pase brevemente por códigos intermedios no deseados. Por ejemplo, al pasar de código 0111 (7) a 1000 (8), si el bit más significativo conmuta antes que los otros, el DAC puede pasar momentáneamente por 1111 (15), generando un glitch positivo. Los DACs de alta velocidad usan técnicas como segmentación (dividir en DAC grueso + DAC fino) y current-steering (conmutar corrientes en lugar de voltajes) para minimizar glitches.

Para aplicaciones donde los glitches son inaceptables (como control de motores o síntesis de audio), se usa un sample-and-hold (S/H) en la salida del DAC: el DAC actualiza mientras el S/H está en modo «hold»(salida desconectada del DAC), y luego el S/H captura la salida estable en modo «sample». Alternativamente, se usa un filtro pasa-bajas para atenuar los glitches de alta frecuencia, aunque esto limita el ancho de banda de la señal reconstruida. La reconstrucción ideal de una señal desde sus muestras requiere un filtro de reconstrucción ideal (sinc interpolation), pero en la práctica se usa un simple zero-order hold (ZOH, la salida se mantiene constante entre muestras) combinado con un filtro anti-imaging pasa-bajas que atenúa las réplicas espectrales (images) en múltiplos de la frecuencia de muestreo.

La Figura 3.5 muestra el proceso de reconstrucción de señal desde muestras digitales, incluyendo efectos de ZOH y filtrado anti-imaging.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s3.2_e2.py`. El script simula reconstrucción de señal desde DAC con diferentes frecuencias de muestreo y muestra el efecto del filtro anti-imaging.

Análisis de glitches: Un DAC de 12 bits con tiempo de establecimiento de $1\ \mu\text{s}$ actualiza a 100 kSPS. Durante la transición del código 2047 a 2048 (mitad de escala), el DAC pasa momentáneamente por código 0 debido a skew entre bits, generando un glitch de -50 % de escala que dura 10 ns. Si la aplicación es control de motor a 1 kHz, ¿el glitch afecta significativamente? Si la aplicación es síntesis de audio a 20 kHz, ¿el glitch es audible? Diseña un filtro para atenuar glitches de 10 ns sin afectar señal de 20 kHz.

3.2.3 DACs en Sistemas de Control y Síntesis de Audio

En sistemas de control en lazo cerrado, el DAC genera la señal de actuación calculada por el algoritmo de control digital (PID, control óptimo, MPC).

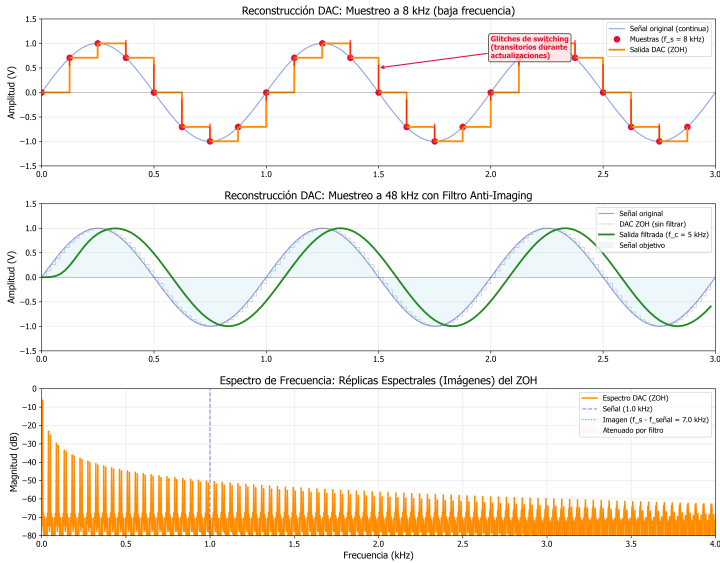


Figura 3.5: Reconstrucción de señal analógica desde DAC. Arriba: muestras digitales discretas. Medio: salida DAC con zero-order hold (ZOH), señal tipo escalera. Abajo: señal reconstruida después de filtro pasa-bajas anti-imaging, que suaviza transiciones y atenúa componentes espectrales en múltiplos de frecuencia de muestreo. Glitches de switching (transitorios durante actualizaciones) se muestran exagerados en rojo. S/H o filtrado adicional puede eliminarlos. Comparación entre diferentes frecuencias de muestreo muestra trade-off entre suavidad de reconstrucción y complejidad de filtrado. (Figura generada con el script `s3_2_e2.py`)

La resolución del DAC determina la granularidad del control: un DAC de 8 bits solo puede tomar 256 valores discretos, causando que la variable controlada (temperatura, posición, velocidad) oscile alrededor del setpoint con amplitud proporcional al LSB del DAC. Para control de precisión, se requieren DACs de 12-16 bits. La velocidad del DAC debe ser mayor que el ancho de banda del sistema controlado: para un sistema térmico con constante de tiempo de 10 s, un DAC de 1 Hz es suficiente; para un servo de posición con ancho de banda de 100 Hz, necesitas un DAC de varios kHz. La latencia total (procesamiento del controlador + actualización del DAC + tiempo de establecimiento) debe ser pequeña comparada con el período de muestreo para evitar degradación del margen de fase y potencial inestabilidad.

En síntesis de audio, DACs de 16-24 bits operando a 44.1-192 kHz reconstruyen señales de audio desde datos digitales almacenados o generados por algoritmos de síntesis. El SNR del DAC debe ser > 90 dB para audio de ca-

lidad profesional (16 bits teóricos dan 96 dB, pero DACs reales logran 90-95 dB debido a ruido térmico y distorsión armónica). La distorsión armónica total (THD+N) debe ser $< 0,01\%$ para audio de alta fidelidad. Los DACs de audio modernos usan arquitecturas sigma-delta con sobremuestreo (convertir 44.1 kHz a 2.8 MHz internamente) combinado con noise shaping para empujar ruido de cuantización fuera de la banda audible, logrando resoluciones efectivas de 20+ bits.

La Figura 3.6 muestra aplicaciones típicas de DAC en generación de formas de onda, control analógico, y síntesis de audio.

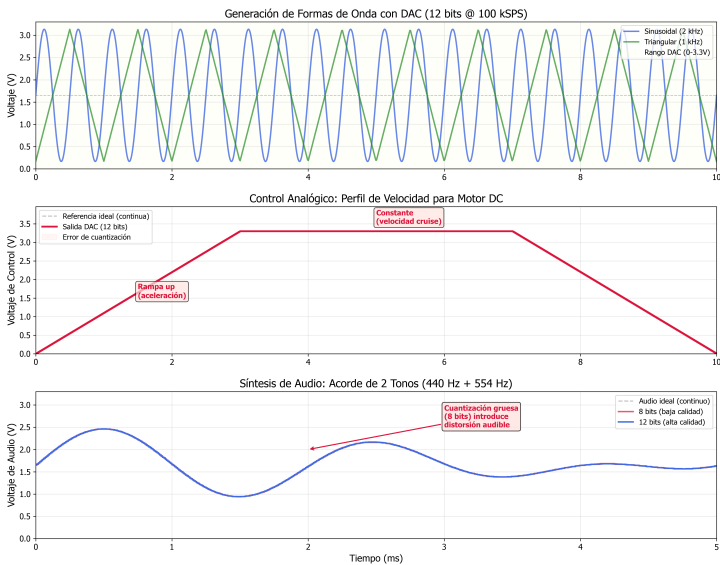


Figura 3.6: Aplicaciones de DAC. Izquierda: Generación de forma de onda arbitraria (sinusoidal, triangular, cuadrada) desde tabla lookup en memoria. Centro: Control analógico de motor DC mediante señal de referencia generada por DAC, procesada por amplificador de potencia. Derecha: Síntesis de audio desde muestras PCM (Pulse Code Modulation), con filtro anti-imaging para eliminar componentes espectrales fuera de banda audible. Requisitos de resolución y velocidad varían dramáticamente entre aplicaciones. (Figura generada con el script `s3.2_e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s3_2_e3.py`. El script genera diferentes formas de onda usando DAC simulado y muestra aplicaciones en síntesis de señales y control.

Diseño de generador de funciones: Usando un DAC de 12 bits @ 1 MSPS, diseña un generador de sinusoidales de frecuencia variable (1 Hz - 100 kHz). Almacena una tabla de 1024 muestras de un ciclo de senoide en memoria. Para generar frecuencia f , lee la tabla a rata $R = f \times 1024$ SPS. ¿Cuál es la máxima frecuencia alcanzable antes de violar Nyquist? ¿Cómo afecta la cuantización de 12 bits a la pureza espectral (SFDR, Spurious-Free Dynamic Range)?

Los DACs son el canal de salida de cualquier sistema digital que debe actuar sobre el mundo analógico. Desde control de motores y actuadores hasta generación de señales de test y síntesis de audio, los DACs convierten decisiones y cálculos digitales en voltajes y corrientes que causan efectos físicos medibles. Cuando diseñas un sistema de control de IA embebida-un robot que ajusta su trayectoria basado en sensores, un sistema de climatización que optimiza temperatura basado en ocupación predicha, un amplificador de audio que compensa distorsión mediante predistorsión digital-el DAC es tan crítico como el ADC en el extremo de entrada. Un DAC lento introduce latencia que puede desestabilizar lazos de control. Un DAC de baja resolución causa oscilaciones tipo limit-cycle alrededor del setpoint deseado. Un DAC con glitches inyecta transitorios que pueden excitar resonancias mecánicas o generar clicks audibles en audio. Y un DAC mal filtrado genera componentes espectrales fuera de banda (imaging) que violan especificaciones de emisiones electromagnéticas o interfieren con otros sistemas. Por eso, seleccionar y configurar DACs apropiados, diseñar filtros de reconstrucción correctos, y caracterizar linealidad y tiempo de establecimiento no es una tarea opcional-es parte integral del diseño de sistemas de control e interfaz humano-máquina robustos.

3.3 Modulación por Ancho de Pulso: Control de Potencia y Generación de Señales

Objetivo de Aprendizaje

Objetivo: Dominar los principios de modulación por ancho de pulso (PWM) para control de potencia, generación de señales analógicas, y modulación de actuadores, **para** diseñar sistemas eficientes de control de motores, LEDs, y elementos calefactores que maximicen eficiencia energética mientras minimizan ruido electromagnético y distorsión.

Ingeniería de Datos desde el Hardware

Un motor DC de 12V/2A consume 24W cuando opera a velocidad máxima. Si usas un resistor variable para controlar velocidad (reducir voltaje efectivo), a 50 % de velocidad la potencia total sigue siendo 24W-12W van al motor, 12W se disipan como calor en el resistor. Pero con control PWM al 50 % de duty cycle, el switch (MOSFET) está completamente ON (pérdidas de conducción mínimas, $P = I^2 R_{DS,on} \approx 0,1W$) o completamente OFF (pérdidas cero) durante cada ciclo. La potencia promedio al motor es 12W, pero las pérdidas en el switch son solo 0.1W, logrando eficiencia de 99 % vs 50 % con control resistivo. Esta diferencia entre disipar 12W como calor vs 0.1W es la razón fundamental por la cual todo control de potencia moderno-desde cargadores de teléfono hasta inversores solares de MW-usa switching PWM en lugar de regulación lineal. Cuando diseñas sistemas de IA embebida alimentados por batería, cada vatio ahorrado en eficiencia se traduce directamente en horas adicionales de operación autónoma.

La modulación por ancho de pulso (PWM, Pulse Width Modulation) es una técnica para codificar un valor analógico como una señal digital binaria (ON/OFF) variando la fracción de tiempo que la señal permanece en estado HIGH vs LOW. El duty cycle D (ciclo de trabajo) es la fracción de tiempo en estado HIGH durante un período: $D = t_{ON}/T = t_{ON}/(t_{ON} + t_{OFF})$, donde T es el período de la señal PWM. El valor promedio de la señal PWM es $V_{avg} = D \times V_{HIGH}$, lo que permite controlar indirectamente un valor analógico (voltaje promedio, potencia promedio, intensidad luminosa promedio) usando solo switching digital rápido. La frecuencia PWM $f_{PWM} = 1/T$ debe ser suficientemente alta para que la carga (motor, LED, filtro RC) responda al valor promedio en lugar de a las oscilaciones indi-

viduales, pero no tan alta que las pérdidas de switching en el transistor dominen sobre las pérdidas de conducción. Para control de motores DC, frecuencias de 1-20 kHz son típicas (por encima del rango audible para evitar zumbido acústico). Para LEDs de iluminación, 100 Hz - 1 kHz es suficiente (por encima de la frecuencia de parpadeo visible humano de 50 Hz). Para conversión DAC mediante PWM filtrado, frecuencias de 10-100 kHz permiten filtros RC compactos.

3.3.1 Generación de PWM con Microcontroladores

Los microcontroladores generan PWM usando timers/counters de hardware que cuentan ciclos de reloj y comparan el valor del contador con un registro de comparación, alternando la salida cuando coinciden. El período PWM está determinado por el valor del registro de auto-recarga (TOP value), y el duty cycle por el valor del registro de comparación (COMPARE value): $D = \frac{COMPARE}{TOP}$. La resolución del PWM en bits es $N = \log_2(TOP + 1)$: un timer de 8 bits permite duty cycles de 0-255 (256 niveles), un timer de 16 bits permite 0-65535 (65536 niveles). La frecuencia PWM es $f_{PWM} = f_{clock} / (prescaler \times (TOP + 1))$, donde *prescaler* divide la frecuencia de reloj del microcontrolador para extender el rango de períodos alcanzables. Existe un trade-off fundamental entre resolución y frecuencia: para un reloj fijo, aumentar resolución (TOP más grande) disminuye frecuencia, y viceversa. Por ejemplo, con un reloj de 16 MHz y prescaler de 1, un PWM de 8 bits opera a $16 \text{ MHz} / 256 \approx 62,5 \text{ kHz}$, pero un PWM de 16 bits opera a $16 \text{ MHz} / 65536 \approx 244 \text{ Hz}$.

Los timers de microcontroladores típicamente soportan múltiples modos PWM: Fast PWM (cuenta hasta TOP y reinicia, generando señal asimétrica), Phase-Correct PWM (cuenta hasta TOP y luego cuenta hacia abajo, generando señal simétrica con mejor comportamiento espectral para control de motores), y Phase and Frequency Correct PWM (similar a Phase-Correct pero con control independiente de frecuencia). La elección del modo afecta la frecuencia efectiva, la resolución, y el contenido armónico de la señal PWM. Para aplicaciones de DAC filtrado, Fast PWM es preferido por su mayor frecuencia. Para control de motores, Phase-Correct PWM reduce corrientes de pico y ruido EMI.

La Figura 3.7 muestra la generación de PWM por hardware usando timers de microcontrolador, incluyendo formas de onda para diferentes duty cycles y modos.

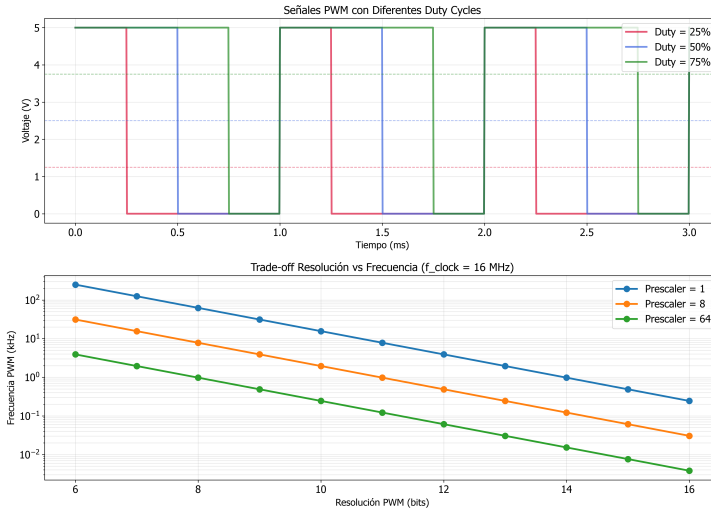


Figura 3.7: Generación de PWM con timer de hardware. Arriba: diagrama conceptual mostrando contador que incrementa cada ciclo de reloj, comparador que detecta coincidencia con valor COMPARE, y flip-flop que alterna salida. Medio: formas de onda para duty cycles de 25 %, 50 %, 75 % mostrando relación entre t_{ON} y período T . Abajo: Fast PWM vs Phase-Correct PWM, destacando diferencias en simetría y frecuencia efectiva. Resolución limitada por bits del timer: 8 bits $\rightarrow$ 256 niveles, 10 bits $\rightarrow$ 1024 niveles, 16 bits $\rightarrow$ 65536 niveles. (Figura generada con el script `s3_3_e1.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s3_3_e1.py`. El script simula generación de PWM por timer de hardware y muestra formas de onda resultantes para diferentes configuraciones.

Configuración de timer PWM: Un Arduino Uno tiene reloj de 16 MHz. Usando timer de 8 bits con prescaler de 64, calcula: (a) Frecuencia PWM resultante. (b) Si cambias a prescaler de 1 para aumentar frecuencia, ¿cuál es la nueva frecuencia? (c) Si necesitas exactamente 10 kHz de PWM con máxima resolución posible, ¿qué valores de prescaler y TOP debes usar? (Pista: usa modo CTC o Fast PWM con TOP programable).

3.3.2 Control de Velocidad de Motores DC y Torque

El control PWM de motores DC funciona porque la inductancia del motor actúa como filtro pasa-bajas: la corriente (y por tanto el torque) no puede

cambiar instantáneamente cuando el voltaje se conmuta. Durante la fase ON del PWM, el voltaje aplicado es V_{CC} y la corriente aumenta exponencialmente con constante de tiempo $\tau = L/R$. Durante la fase OFF, la corriente disminuye, circulando por el diodo flyback en paralelo al motor. Si la frecuencia PWM es suficientemente alta ($f_{PWM} \gg 1/\tau$), la corriente tiene ripple pequeño alrededor de un valor promedio $I_{avg} \propto D \times V_{CC}$, y el torque promedio es proporcional a la corriente promedio. La velocidad del motor en régimen permanente es proporcional al voltaje promedio para cargas constantes: $\omega \propto D \times V_{CC}$.

Pero el ripple de corriente no es cero, y este ripple causa pérdidas adicionales (heating del motor por corriente RMS mayor que corriente DC equivalente), ruido acústico (vibración mecánica a frecuencia PWM), y estrés en los devanados. Para minimizar ripple, se aumenta la frecuencia PWM o se agrega un filtro LC pasa-bajas entre el driver y el motor. Sin embargo, frecuencias muy altas aumentan pérdidas de switching en el MOSFET (cada transición ON/OFF tiene pérdidas de energía $E_{sw} \propto V_{CC} \times I_{load} \times t_{sw}$, donde t_{sw} es el tiempo de conmutación). El diseño óptimo balancea ripple de corriente, pérdidas de switching, ruido acústico, y EMI.

Para motores brushless (BLDC, Brushless DC) y motores paso a paso, se usa PWM multi-fase: 3 señales PWM con desfase de 120° para BLDC trifásico, o 4-8 señales con secuencias específicas para steppers. El microcontrolador debe generar estas señales sincronizadas con resolución de microsegundos, típicamente usando timers dedicados con salidas complementarias (para drivers de medio puente o puente completo) y dead-time insertion (para evitar shoot-through durante conmutación).

La Figura 3.8 muestra control PWM de motor DC incluyendo ripple de corriente, efecto de inductancia, y comparación con control lineal resistivo.

Reto Computacional: Experimenta con el Script

Ejecuta el script `s3.3_e2.py`. El script simula control PWM de motor DC y muestra ripple de corriente, eficiencia, y comparación con control lineal.

Diseño de control de motor: Motor DC de 24V / 3A, inductancia de 8 mH, resistencia de 2 Ω . Usas PWM a 20 kHz para controlar velocidad. Calcula: (a) Constante de tiempo $\tau = L/R$. (b) Ripple de corriente pico-pico a duty cycle de 50% (usa aproximación $\Delta I \approx (V_{CC} \times D \times (1 - D)) / (L \times f_{PWM})$). (c) Pérdidas de conducción en MOSFET con $R_{DS,on} = 20 \text{ m}\Omega$ a corriente promedio de 1.5 A. (d) Pérdidas de switching si tiempo de conmutación es 100 ns y frecuencia es 20 kHz.

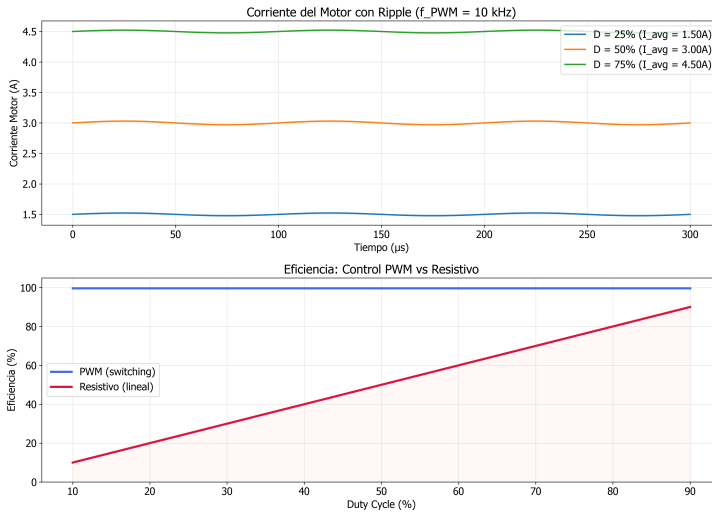


Figura 3.8: Control PWM de motor DC. Izquierda: circuito con MOSFET de potencia, diodo flyback, y motor. Centro: formas de onda mostrando voltaje PWM aplicado y corriente del motor con ripple filtrado por inductancia. Duty cycle de 50 % genera velocidad de 50 % de máximo. Derecha: comparación de eficiencia entre control resistivo (lineal) y PWM (switching). Control resistivo disipa potencia proporcional a $(1 - D)^2$, mientras PWM solo disipa pérdidas de conducción y switching, logrando eficiencia >95 % en amplio rango de duty cycles. (Figura generada con el script `s3_3_e2.py`)

3.3.3 PWM para LEDs y Percepción Visual

Los LEDs responden instantáneamente a cambios de corriente (sin inercia térmica o mecánica como lámparas incandescentes), lo que los hace ideales para control PWM. El ojo humano tiene persistencia de visión de aproximadamente 15-20 ms (50-60 Hz), así que frecuencias PWM por encima de 100 Hz aparecen como intensidad constante sin parpadeo perceptible. Sin embargo, efectos estroboscópicos pueden ocurrir con frecuencias bajas cuando objetos en movimiento son iluminados por LED PWM: un ventilador girando parece congelado o moverse lentamente cuando la frecuencia del PWM coincide con la frecuencia de rotación. Por esto, iluminación de calidad usa frecuencias de varios kHz.

Un desafío adicional es la percepción logarítmica de brillo humano: el ojo percibe cambios de intensidad luminosa en escala logarítmica, no lineal. Si generas PWM linealmente (duty cycles de 0 %, 10 %, 20 %, ..., 100 %), los pasos perceptuales de brillo no son uniformes-los cambios en bajas intensidades se perciben como grandes saltos, mientras que en altas intensida-

des los cambios son imperceptibles. Para brillo percibido uniforme, debes aplicar gamma correction: mapear duty cycle según una función no-lineal $D = (I_{deseado}/I_{max})^\gamma$, donde $\gamma \approx 2 - 2,5$ para LEDs típicos. Esto significa que para 50 % de brillo perceptual, necesitas aproximadamente 25 % de duty cycle.

Para control de color RGB con LEDs, se usan 3 canales PWM independientes (rojo, verde, azul) con mixing óptico. La resolución efectiva de color depende de la resolución de cada canal: 8 bits por canal (256 niveles) da $256^3 \approx 16,7$ millones de colores. Pero debido a gamma correction y limitaciones de los LEDs (especialmente en bajo brillo donde ruido y no-linealidad dominan), la resolución efectiva es menor, típicamente equivalente a 6-7 bits por canal en implementaciones de bajo costo.

La Figura 3.9 muestra control PWM de LEDs incluyendo percepción de brillo, gamma correction, y mixing de color RGB.

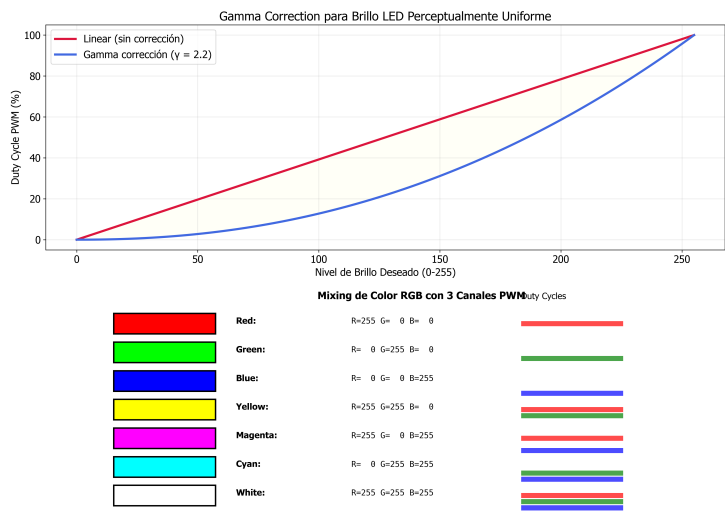


Figura 3.9: Control PWM de LEDs. Arriba izquierda: duty cycle lineal (0-100 %) vs brillo percibido (no-lineal debido a respuesta logarítmica del ojo). Arriba derecha: gamma correction aplica mapeo no-lineal para brillo percibido uniforme. Abajo: mixing de color RGB con 3 canales PWM independientes genera espectro completo de colores visibles. Frecuencia PWM ¿200 Hz evita parpadeo perceptible. Resolución de 8 bits por canal permite 16.7M colores teóricos, pero efectivo es 6-7 bits debido a gamma y limitaciones de LEDs en bajo brillo. (Figura generada con el script `s3_3_e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta el script `s3_3_e3.py`. El script simula control PWM de LEDs con gamma correction y mixing de color RGB.

Implementación de gamma correction: Quieres controlar brillo de un LED con 256 niveles perceptualmente uniformes usando PWM de 8 bits. Genera tabla lookup de 256 entradas donde entrada i (0-255) mapea a duty cycle $D_i = (i/255)^{2.2} \times 255$ (redondeado a entero). Implementa en Arduino como array en PROGMEM para ahorrar RAM. Compara brillo perceptual usando tabla vs duty cycle lineal: enciende LED a niveles de 10, 20, 30, ..., 250 y observa uniformidad de pasos percibidos.

PWM es la técnica universal de control de potencia en sistemas embebidos modernos porque ofrece eficiencia inalcanzable por métodos lineales (resistivos) mientras mantiene simplicidad de implementación (solo requiere un timer de hardware y un transistor de potencia). Desde control de velocidad de ventiladores en laptops hasta drivers de motores en drones y robots, desde regulación de brillo en displays hasta cargadores de baterías de vehículos eléctricos, PWM es ubicuo precisamente porque minimiza disipación de potencia en el elemento de control, permitiendo sistemas compactos sin disipadores térmicos masivos. Cuando diseñas sistemas de IA embebida que deben operar con restricciones de energía severas-robots autónomos con batería limitada, sensores IoT alimentados por solar, drones con presupuesto de peso crítico-entender cómo diseñar drivers PWM eficientes, cómo seleccionar frecuencias que balancean ripple vs pérdidas de switching, y cómo mitigar EMI generado por transiciones rápidas de corriente no es conocimiento opcional, es requisito fundamental para construir sistemas que funcionan en el mundo real en lugar de solo en simulación o prototipo de laboratorio.

3.4 Protocolos de Comunicación: UART, SPI e I2C

Objetivo de Aprendizaje

Objetivo: Comparar los protocolos de comunicación serie UART, SPI, e I2C en términos de velocidad, complejidad, topología, y robustez, **para** seleccionar e implementar el protocolo apropiado para interfaz con sensores, displays, y módulos de comunicación en sistemas embebidos de IA.

Ingeniería de Datos desde el Hardware

Un acelerómetro I2C puede transmitir datos de 3 ejes (6 bytes) en $50 \mu\text{s}$ @ 400 kHz. El mismo sensor con interfaz SPI puede hacerlo en $6 \mu\text{s}$ @ 8 MHz. Pero I2C usa solo 2 pines (SDA, SCL) y permite hasta 127 dispositivos en el mismo bus, mientras SPI requiere 4+ pines (MISO, MOSI, SCK, CS) y necesita un chip-select adicional por cada dispositivo. Cuando diseñas un sistema con 10+ sensores y pines limitados del microcontrolador, I2C gana por simplicidad de cableado. Cuando necesitas streaming continuo de IMU a 1 kHz para control de dron, SPI gana por velocidad. No hay protocolo universalmente mejor-cada uno tiene trade-offs específicos que debes balancear según restricciones de tu aplicación.

Los protocolos de comunicación serie son el mecanismo estándar para transferir datos entre microcontroladores, sensores, módulos de comunicación (WiFi, Bluetooth, LoRa), displays, y memorias. Los tres protocolos más comunes en sistemas embebidos son UART (Universal Asynchronous Receiver-Transmitter), SPI (Serial Peripheral Interface), e I2C (Inter-Integrated Circuit). UART es asíncrono punto-a-punto, simple pero lento (típicamente $< 1 \text{ Mbps}$). SPI es síncrono maestro-esclavo, rápido (decenas de MHz) pero requiere muchos pines para múltiples esclavos. I2C es síncrono multi-maestro con direccionamiento, moderadamente rápido (100-400 kHz estándar, hasta 3.4 MHz en modo HS), y eficiente en pines (solo 2 líneas compartidas). La selección del protocolo depende de velocidad requerida, número de dispositivos, distancia de cableado, disponibilidad de pines, y complejidad de implementación.

3.4.1 UART: Comunicación Asíncrona Punto-a-Punto

UART transmite datos de forma asíncrona: no hay señal de reloj compartida, cada dispositivo tiene su propio reloj interno y se sincroniza detectando los flancos de los bits de start y stop. Un frame típico UART consiste en: 1 bit de start (siempre LOW), 5-9 bits de datos (típicamente 8), bit de paridad opcional, y 1-2 bits de stop (siempre HIGH). La velocidad de transmisión (baud rate) debe ser idéntica en ambos extremos (tolerancia típica de $\pm 2\%$). Baud rates comunes son 9600, 19200, 38400, 57600, 115200 bps.

UART es full-duplex (transmisión y recepción simultáneas en líneas separadas TX/RX), pero solo punto-a-punto. Para interfaz con PC o módulos que usan RS-232, se requiere convertidor de niveles (chip MAX232). La Figura 3.10 muestra el timing de frames UART.

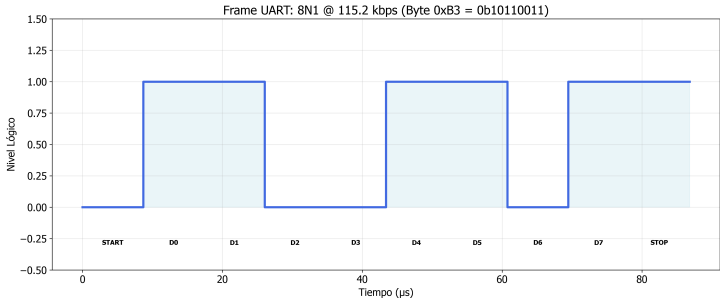


Figura 3.10: Protocolo UART. Frame mostrando bit de start, 8 bits de datos (LSB primero), paridad opcional, y stop bits. Sincronización basada en detección de start bit. Baud rate debe coincidir ($\pm 2\%$ tolerancia). (Figura generada con `s3_4_e1.py`)

3.4.2 SPI: Comunicación Síncrona de Alta Velocidad

SPI usa 4 líneas: SCLK (reloj), MOSI (Master Out Slave In), MISO (Master In Slave Out), y CS (Chip Select). Velocidades llegan a decenas de MHz. Full-duplex: datos en ambas direcciones simultáneamente. Problema: cada esclavo requiere línea CS dedicada. SPI tiene 4 modos (CPOL, CPHA) que definen polaridad y fase de reloj. La Figura 3.11 muestra timing de SPI.

3.4.3 I2C: Bus Multi-Dispositivo con Direcccionamiento

I2C usa solo 2 líneas: SDA (datos) y SCL (reloj), ambas con pull-ups. Soporta hasta 127 dispositivos en mismo bus. Comunicación inicia con START, dirección de 7 bits + R/W, datos con ACK, y STOP. Velocidades: 100 kHz (estándar), 400 kHz (rápido), hasta 3.4 MHz (HS). Half-duplex. La Figura 3.12 muestra protocolo I2C.

Reto Computacional: Experimenta con el Script

Ejecuta los scripts `s3_4_e1.py`, `s3_4_e2.py`, y `s3_4_e3.py` para simular timing de UART, SPI, e I2C.

Comparación: Calcula tiempo para transferir 1 kB: (a) UART @ 115200 bps con 8N1. (b) SPI @ 8 MHz. (c) I2C @ 400 kHz con overhead. ¿Cuál es más rápido?

Los protocolos serie conectan todos los componentes del sistema embebido. La selección afecta latencia, pines disponibles, y robustez contra ruido. Cuando integras múltiples sensores, entender timing de protocolo es crítico

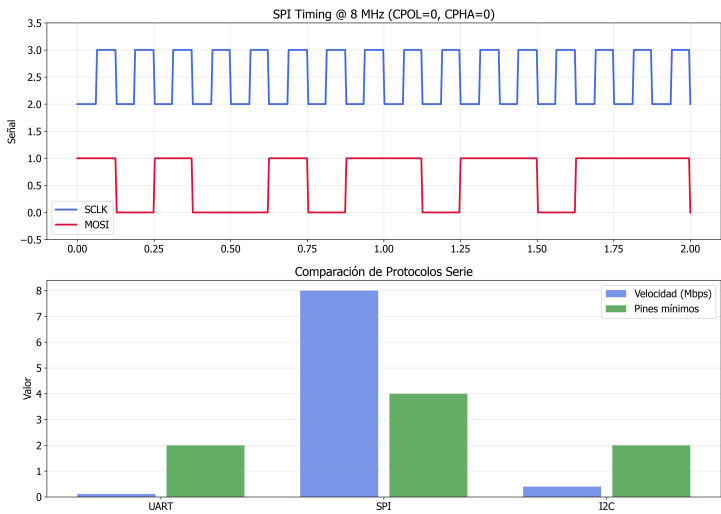


Figura 3.11: Protocolo SPI. Maestro genera SCLK y controla CS. Datos síncronos en MOSI/MISO simultáneamente. Full-duplex, alta velocidad. Requiere CS dedicada por esclavo. (Figura generada con `s3_4_e2.py`)

para diagnosticar problemas de comunicación.

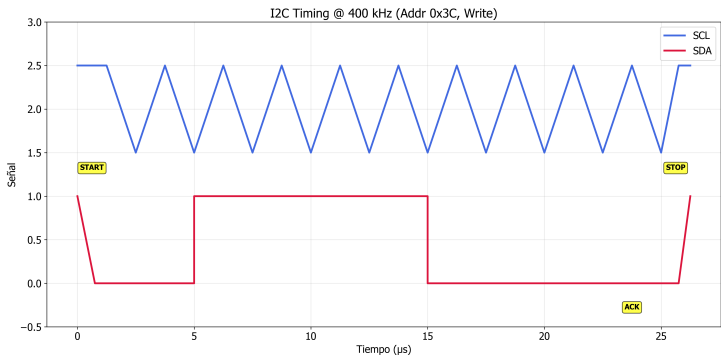


Figura 3.12: Protocolo I2C. START (SDA↓ mientras SCL alta), dirección + R/W, ACK, datos con ACK, STOP (SDA↑ mientras SCL alta). Pull-ups permiten operación multi-dispositivo. (Figura generada con `s3_4.e3.py`)

Esta página se dejó en blanco intencionalmente.

Capítulo 4

Gestión de Potencia y Diseño de Sistemas

4.1 Fuentes de Alimentación y Regulación de Voltaje

Objetivo de Aprendizaje

Objetivo: Analizar arquitecturas de fuentes de alimentación lineales y conmutadas, **para** seleccionar sistemas que provean voltajes estables minimizando ruido y maximizando eficiencia en sistemas embebidos alimentados por batería.

Ingeniería de Datos desde el Hardware

ESP32 consume 500 mA @ 3.3V. Desde batería de 12V con regulador lineal: eficiencia 27 %, duración 5.5h. Con buck converter: eficiencia 90 %, duración 13h. Esta diferencia determina viabilidad del sistema en robótica móvil.

Las fuentes convierten voltajes de entrada en voltajes regulados. Reguladores lineales son simples pero ineficientes ($\eta = V_{out}/V_{in}$). Reguladores conmutados logran 85-95 % eficiencia pero generan ruido EMI. El diseño balancea eficiencia, ruido, tamaño, y costo.

4.1.1 Reguladores Lineales

Funcionan como resistencias variables. Disipación: $P = (V_{in} - V_{out}) \times I_{out}$. LDOs permiten dropout bajo (100-300 mV). Ventajas: sin ruido EMI, respuesta rápida. Desventajas: baja eficiencia con grandes diferencias de voltaje. La Figura 4.1 muestra eficiencia vs dropout.

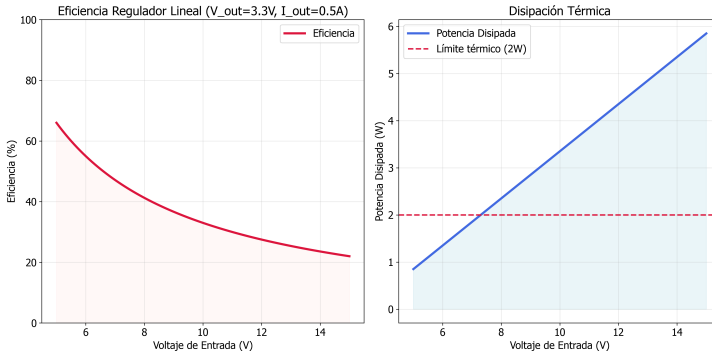


Figura 4.1: Reguladores lineales. Eficiencia vs dropout. LDO mejora eficiencia con V_{in} cercano a V_{out} . Gestión térmica crítica: $T_j = T_{amb} + P_{diss} \times \theta_{JA}$. (s4_1_e1.py)

4.1.2 Reguladores Conmutados

Switching a 100 kHz - 2 MHz con inductor y capacitores. Buck: $V_{out} = D \times V_{in}$. Boost: $V_{out} > V_{in}$. Eficiencia 85-95 %. Pérdidas: conducción, switching, DCR inductor, ESR capacitores. La Figura 4.2 compara topologías.

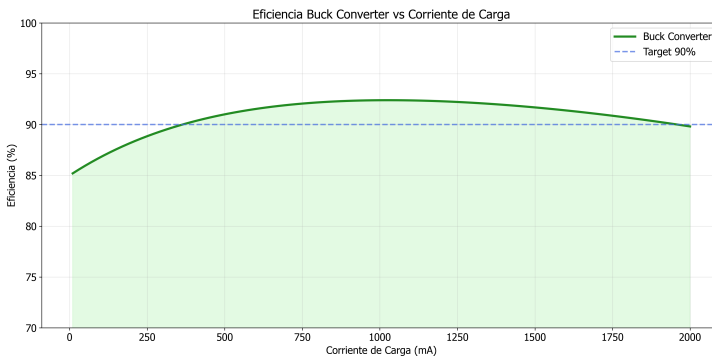


Figura 4.2: Reguladores conmutados. Buck, boost, buck-boost. Eficiencia vs carga. Selección de inductor: mayor L $\rightarrow$ menor ripple, mayor tamaño. (s4_1_e2.py)

4.1.3 Gestión de Ruido

Ripple típico: 10-100 mV. Para ADCs sensibles: post-regulación con LDO. Layout crítico: minimizar loops de corriente, ground plane continuo. La Figura 4.3 muestra espectro de ruido.

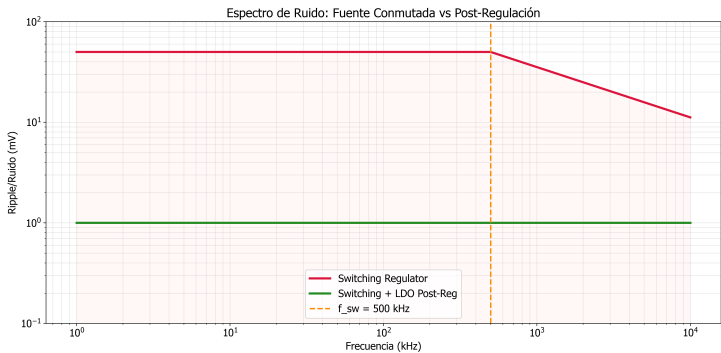


Figura 4.3: Ruido en fuentes conmutadas. Espectro con fundamental y armónicos. Mitigación: capacitores bajo ESR, filtros LC, post-regulación LDO. (s4_1_e3.py)

Reto Computacional: Experimenta con el Script

Ejecuta `s4_1_e1.py`, `s4_1_e2.py`, `s4_1_e3.py`. Diseña fuente para ESP32 (3.3V @ 500 mA) desde 12V. Compara LDO, buck+LDO, y buck directo.

Fuentes determinan autonomía y calidad de señal. Cada punto de eficiencia es crucial para duración de batería. Ripple contamina ADCs si no se gestiona.

4.2 Baterías y Gestión de Energía

Objetivo de Aprendizaje

Objetivo: Comprender características de baterías (Li-ion, LiPo, NiMH) y sistemas de gestión de energía, **para** diseñar sistemas autónomos con autonomía predecible y carga segura.

Ingeniería de Datos desde el Hardware

Batería LiPo 3.7V/2000 mAh descargada a 1C (2A) entrega 7 Wh. Descargada a 2C (4A), entrega solo 6 Wh por calentamiento interno. A -10°C, capacidad cae 30 %. Sistemas IoT en exteriores deben diseñar para peor caso, no especificaciones nominales.

Baterías almacenan energía química convertible en eléctrica. Li-ion: 3.7V nominal, alta densidad energética (150-250 Wh/kg), sensibles a sobrecarga/descarga profunda (BMS obligatorio). NiMH: 1.2V nominal, menor densidad (60-100 Wh/kg), más robustas. Capacidad en mAh $\times$ voltaje = energía en mWh. Tasa C: 1C significa descarga en 1 hora (batería de 2000 mAh a 1C = 2A).

4.2.1 Curvas de Descarga y Capacidad Efectiva

Voltaje de batería no es constante: Li-ion va de 4.2V (cargada) a 3.0V (descargada). Sistemas deben operar en todo el rango o usar buck-boost. Capacidad efectiva depende de corriente de descarga: a 2C, solo 80 % de capacidad nominal disponible. Temperatura afecta: a 0°C, capacidad cae 20 %. La Figura 4.4 muestra curvas típicas.

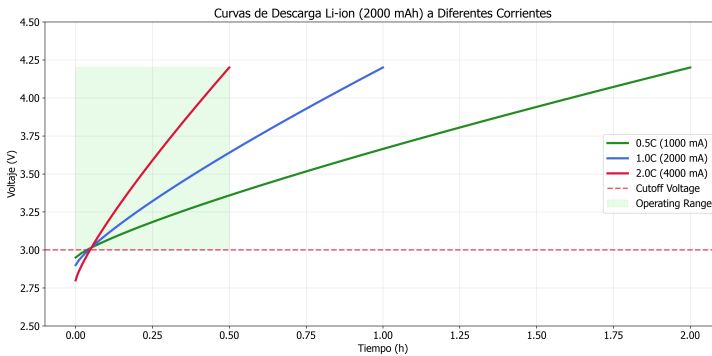


Figura 4.4: Curvas de descarga Li-ion a diferentes corrientes (0.5C, 1C, 2C). Voltaje cae no-linealmente. Capacidad efectiva disminuye con corriente alta. (s4_2_e1.py)

4.2.2 Sistemas de Gestión de Batería (BMS)

BMS protege contra: sobrecarga ($> 4.2V$ causa incendio), descarga profunda ($< 2.5V$ daña permanentemente), sobrecorriente, sobrecalentamiento. Incluye: monitoreo de voltaje por celda, balanceo (ecualizar carga entre celdas en serie), protección con MOSFETs. Estimación de estado de carga (SOC) mediante coulomb counting o voltaje. La Figura 4.5 muestra arquitectura típica.

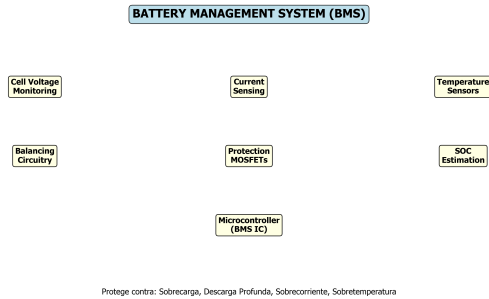


Figura 4.5: Sistema BMS. Monitoreo por celda, balanceo pasivo/activo, protección MOSFETs. SOC estimation por integración de corriente. (s4_2_e2.py)

4.2.3 Cálculo de Autonomía y Presupuesto de Potencia

Autonomía = (Capacidad × Voltaje × Eficiencia) / Potencia_Promedio.
Ejemplo: 2000 mAh @ 3.7V (7.4 Wh), sistema consume 500 mW promedio, eficiencia regulador 90 %: autonomía = $7.4 \times 0.9 / 0.5 = 13.3\text{h}$. Presupuesto debe incluir picos (WiFi transmit 300 mA) y promediar duty cycle. La Figura 4.6 muestra análisis.

Reto Computacional: Experimenta con el Script

Ejecuta s4_2_e1.py, s4_2_e2.py, s4_2_e3.py. Calcula autonomía para ESP32 con batería 2000 mAh: sleep 10 mA (90 % tiempo), active 80 mA (9 %), TX 250 mA (1 %).

Baterías determinan autonomía. BMS es crítico para seguridad. Presupuesto de potencia debe considerar todos modos de operación, no solo promedio simple.

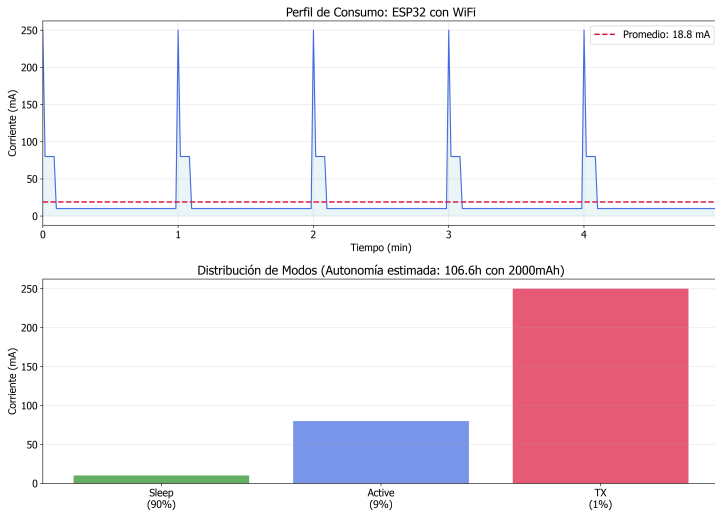


Figura 4.6: Presupuesto de potencia. Corriente vs tiempo mostrando picos (TX WiFi), sleep modes. Cálculo de promedio ponderado determina autonomía real. (`s4_2_e3.py`)

4.3 Ruido, Blindaje, y Compatibilidad Electromagnética

Objetivo de Aprendizaje

Objetivo: Identificar fuentes de ruido electromagnético y técnicas de mitigación (blindaje, filtrado, grounding), **para** diseñar sistemas que cumplan normativas EMC y mantengan integridad de señal en entornos ruidosos.

Ingeniería de Datos desde el Hardware

ADC de 12 bits con LSB de 0.8 mV midiendo señal de termopar (40 mV para 100°C) pierde 3 LSBs (2.4 mV) por ruido de switching a 10 cm del regulador DC-DC sin filtrado. Equivale a error de $\pm 6^{\circ}\text{C}$. Ningún algoritmo de IA puede corregir datos corruptos en origen.

Ruido electromagnético corrompe señales analógicas y digitales. Fuentes: switching de fuentes de alimentación, transitorios digitales, motores, RF externo. Acoplamiento por: conducción (ground loops, impedancia com-

partida), capacitancia (dV/dt alto), inductancia (dI/dt alto), radiación. Mitigación requiere diseño integral: PCB layout, grounding, shielding, filtrado.

4.3.1 Fuentes de Ruido y Mecanismos de Acoplamiento

Ruido conducido: corrientes de switching en ground crean caída de voltaje por impedancia finita, afectando referencias de otros circuitos. Ground loops: múltiples paths de retorno crean corrientes circulantes. Ruido radiado: loops de corriente alta frecuencia actúan como antenas. Crosstalk: señales rápidas en trazas adyacentes inducen voltaje por acoplamiento capacitivo. La Figura 4.7 muestra mecanismos.

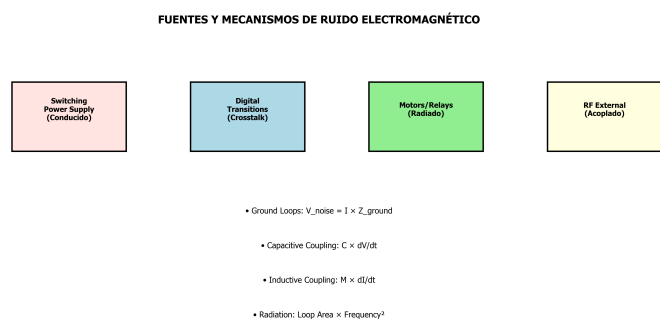


Figura 4.7: Fuentes y mecanismos de ruido. Ground loops, acoplamiento capacitivo/inductivo, EMI radiado. Impedancia de ground no es cero: $V_{noise} = I \times Z_{ground}$. (s4_3_e1.py)

4.3.2 Técnicas de Grounding y Segregación

Star grounding: un solo punto de conexión para dominios aislados (analog, digital, power). Minimiza ground loops pero requiere layout cuidadoso. Ground planes: capa continua de cobre provee baja impedancia, shielding parcial. Split grounds: separar analog/digital, conectar en un punto cerca de ADC. Ferrite beads: alta impedancia a RF, baja a DC, aísla dominios. La Figura 4.8 compara técnicas.

4.3.3 Filtrado y Blindaje

Capacitores de bypass: 100 nF cerca de cada IC filtra ruido de alta frecuencia. Bulk capacitors (10-100 μF) cerca de fuente filtran baja frecuencia.

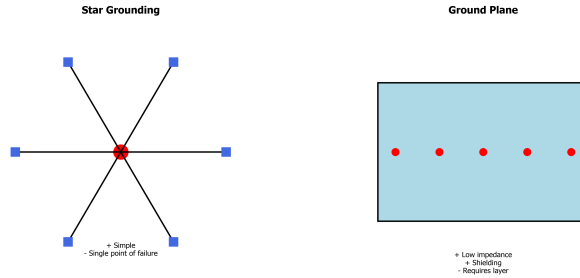


Figura 4.8: Estrategias de grounding. Star ground vs ground plane. Split grounds con ferrite beads. Trade-offs: star simple pero frágil, plane robusto pero requiere vias. (s4_3_e2.py)

Filtros LC: atenúan ripple de fuentes conmutadas. Blindaje: caja metálica conectada a ground bloquea RF (efectivo ¡30 MHz). Para campos magnéticos de baja frecuencia (<1 MHz), se requiere μ -metal. La Figura 4.9 muestra técnicas.

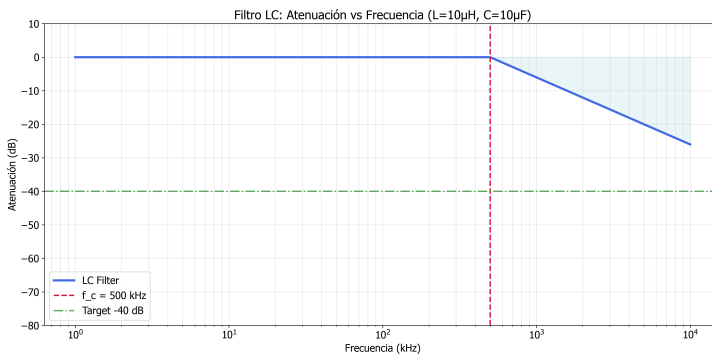


Figura 4.9: Filtrado y blindaje. Capacitores de bypass cerca de ICs. Filtros LC en fuentes. Blindaje metálico para RF. Efectividad vs frecuencia. (s4_3_e3.py)

Reto Computacional: Experimenta con el Script

Ejecuta `s4_3_e1.py`, `s4_3_e2.py`, `s4_3_e3.py`. Diseña filtro LC para atenuar switching noise de 500 kHz en riel de 3.3V. Target: -40 dB @ 500 kHz.

Ruido es inevitable pero gestionable. Ground planes, filtrado, y segregación son esenciales. Cada mV de ruido en ADC es LSBs perdidos permanentemente.

4.4 Diseño de PCB y Layout para Alta Integridad de Señal

Objetivo de Aprendizaje

Objetivo: Aplicar principios de diseño de PCB (capas, ancho de trazas, vias, clearance) **para** crear layouts que minimicen interferencia, maximicen confiabilidad, y cumplan restricciones de manufactura.

Ingeniería de Datos desde el Hardware

Traza de 10 mil (0.25 mm) a 1 oz/ft² cobre transporta 0.5A continuo sin sobrecalentamiento. Para 2A, necesitas 50 mil. Motor de 5A requiere 150+ mil o múltiples capas en paralelo. Subestimar ancho causa fallas térmicas intermitentes imposibles de debuggear.

PCB (Printed Circuit Board) interconecta componentes mediante trazas de cobre. Diseño afecta: impedancia (señales de alta velocidad), EMI (radiación), térmica (disipación calor), manufacturabilidad (costo). Reglas básicas: ancho de traza según corriente, clearance según voltaje, vias para conexiones entre capas, ground/power planes para baja impedancia.

4.4.1 Cálculo de Ancho de Trazas y Gestión Térmica

Ancho mínimo depende de corriente y elevación de temperatura permitida. Fórmula IPC-2221: $A = \frac{I}{(k \times \Delta T^{0.44})^{1/0.725}}$ donde A en mil², I en A, ΔT en °C, $k = 0,024$ para trazas externas, $k = 0,048$ internas. Regla práctica: 1 mil por 1 mA para $\Delta T=10^\circ\text{C}$ (externas). Para corrientes > 1A, usar polígonos o múltiples capas. Vias conducen calor entre capas. La Figura 4.10 muestra diseño térmico.

4.4.2 Capas, Stackup, y Impedancia Controlada

PCB multicapa: alternancia signal/ground/power/signal. Stackup 4-layer típico: Top (signal) / Ground / Power / Bottom (signal). Ground/power

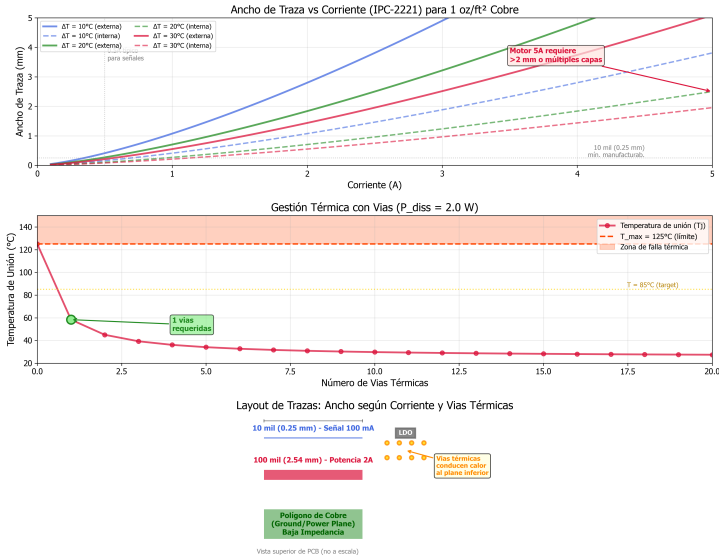


Figura 4.10: Ancho de trazas vs corriente para diferentes ΔT . Vías térmicas conducen calor de componentes a ground plane. Polígonos para corrientes altas ($> 1 \text{ A}$). (s4.4_e1.py)

planes proveen: retorno de baja impedancia, shielding entre señales, capacitancia distribuida (desacople). Para señales $> 50 \text{ MHz}$ o impedancias controladas ($50\Omega \text{ RF}$, $90\Omega \text{ USB}$ differential), se requiere cálculo preciso de geometría (ancho, altura dieléctrico). La Figura 4.11 muestra configuraciones.

4.4.3 Reglas de Diseño y DFM

Clearance mínimo entre trazas: 6-8 mil para bajo voltaje ($< 50 \text{ V}$), mayor para alto voltaje (1 mil por 60V aproximado). Via diameter: drill + 10 mil (ej: drill 12 mil $\rightarrow$ via 22 mil). Soldermask: 4 mil clearance alrededor de pads. Silkscreen: texto mínimo 40 mil alto, 6 mil ancho. DFM (Design for Manufacturing): panelización, fiduciales para pick-and-place, test points. La Figura 4.12 muestra checklist.

Reto Computacional: Experimenta con el Script

Ejecuta `s4.4_e1.py`, `s4.4_e2.py`, `s4.4_e3.py`. Calcula ancho de traza para 3A continuo, $\Delta T = 20^\circ\text{C}$, cobre 1 oz. Diseña stackup 4-layer para sistema mixto analog/digital.

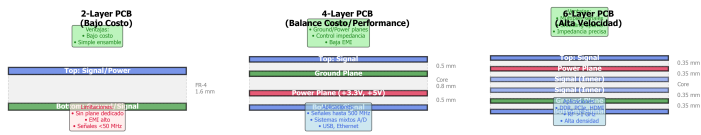


Figura 4.11: Stackups de 2-layer (bajo costo, limitado), 4-layer (balance costo/performance), 6-layer (alta velocidad). Ground planes críticos para integridad de señal. (s4_4.e2.py)

Layout de PCB no es arte - es ingeniería con reglas cuantificables. Cada decisión (ancho, clearance, capas) tiene trade-offs medibles en costo, performance, confiabilidad.

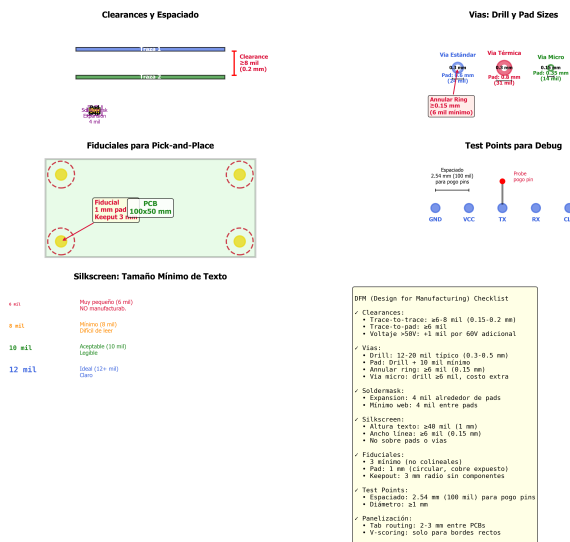


Figura 4.12: Reglas DFM. Clearances, via sizes, soldermask expansion. Fiduciales para ensamble automatizado. Test points para debug. (s4_4_e3.py)

Capítulo 5

Integración de Sistemas para IA

5.1 Acondicionamiento de Señales Analógicas

Objetivo de Aprendizaje

Objetivo: Diseñar circuitos de acondicionamiento (amplificación, filtrado antialiasing, offset) para adaptar señales de sensores al rango de entrada de ADCs, maximizando SNR y previniendo aliasing en sistemas de adquisición de datos para IA.

Ingeniería de Datos desde el Hardware

Sensor de temperatura LM35 entrega 10 mV/°C (0-100°C = 0-1V). ADC de 12 bits @ 3.3V tiene resolución 0.8 mV. Sin acondicionamiento: usas solo 30 % del rango, SNR efectivo baja de 72 dB a 62 dB. Con amplificación x3.3 y offset para centrar señal: aprovechas 100 % del rango, SNR recupera 72 dB, precisión sube de ±0.5°C a ±0.15°C. El acondicionamiento no es opcional-determina la calidad de tus datos de entrenamiento.

Acondicionamiento transforma señales de sensores para optimizar digitalización. Etapas típicas: (1) Amplificación: ajustar ganancia para llenar rango de ADC. (2) Offset: centrar señal bipolar en rango unipolar del ADC. (3) Filtrado antialiasing: eliminar frecuencias $\geq f_{Nyquist}$ para prevenir aliasing. Sin acondicionamiento, desperdicias bits del ADC y introduces ruido.

5.1.1 Amplificación y Ajuste de Ganancia

Sensor típico entrega 10-100 mV. ADC requiere 0-3.3V o 0-5V. Ganancia óptima: $G = V_{ADC_max} / V_{sensor_max}$. Amplificador no inversor: $V_{out} = (1 +$

$R_f/R_i) \times V_{in}$. SNR mejora al usar rango completo: cada bit desaprovechado reduce SNR en 6 dB. La Figura 5.1 muestra diseño de amplificación.

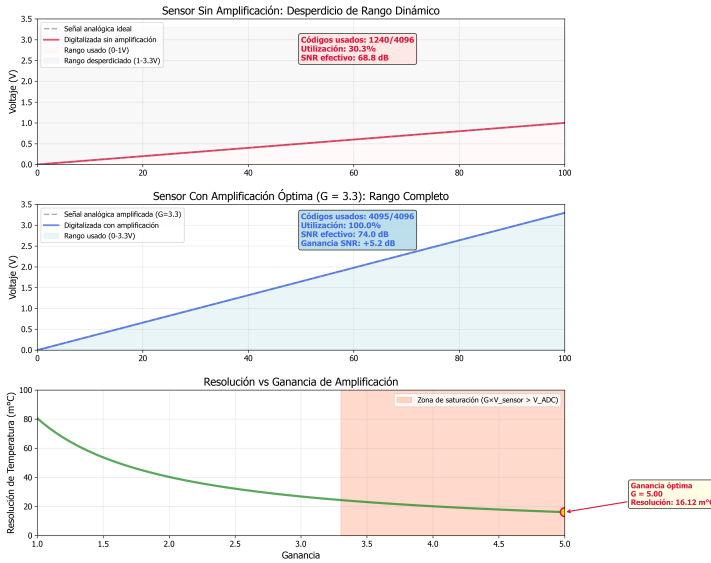


Figura 5.1: Amplificación de señal. Sensor LM35 (0-1V) amplificado x3.3 para rango 0-3.3V. Sin amplificación: solo 1240 códigos de 4096 usados. Con amplificación: 4096 códigos, SNR +10 dB. (`ss5_1_e1.py`)

5.1.2 Offset y Nivel DC

Señales bipolares (acelerómetro: $\pm 2g$, ECG: ± 5 mV) requieren offset para ADC unipolar. Sumador con opamp: $V_{out} = V_{signal} + V_{offset}$. Offset típico: $V_{ref}/2$ para centrar. Rail-to-rail opamps (MCP6001) maximizan rango dinámico. La Figura 5.2 muestra circuito de offset.

5.1.3 Filtrado Antialiasing

Teorema de Nyquist: $f_{sample} \geq 2 \times f_{signal}$. Frecuencias $\geq f_{Nyquist}$ se replican como aliasing. Filtro antialiasing (lowpass) antes de ADC: atenúa ≥ 40 dB componentes sobre $f_{Nyquist}$. Sallen-Key 2do orden Butterworth: pendiente 40 dB/década, f_{cutoff} típicamente $0.4 \times f_{sample}$. La Figura 5.3 muestra respuesta de filtro.

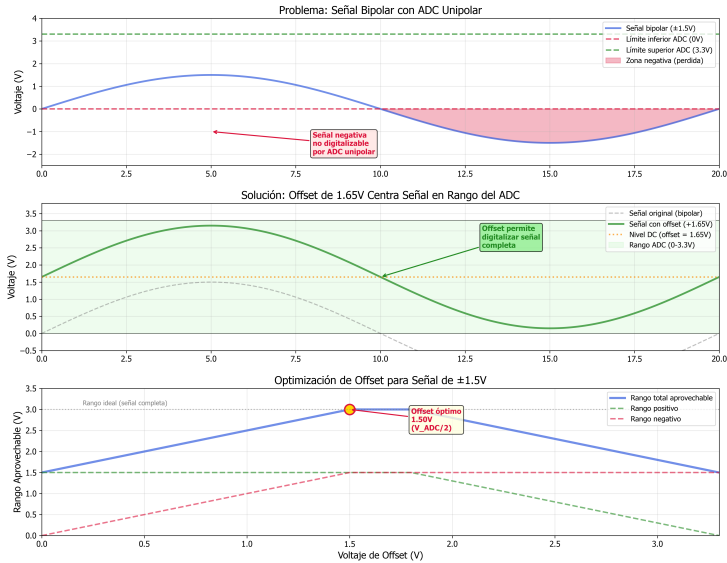


Figura 5.2: Ajuste de offset. Señal bipolar $\pm 1.65V$ centrada a rango 0-3.3V mediante sumador con $V_{offset} = 1.65V$. Permite digitalizar señales AC con ADC unipolar. (s5_1_e2.py)

Reto Computacional: Experimenta con el Script

Ejecuta `s5_1_e1.py`, `s5_1_e2.py`, `s5_1_e3.py`. Diseña cadena de acondicionamiento para micrófono electret (-20 mV a $+20\text{ mV}$) @ ADC 12-bit 3.3V, $f_{sample} = 16\text{ kHz}$. Calcula ganancia, offset, y f_{cutoff} de filtro antialiasing.

Acondicionamiento determina resolución efectiva del sistema de adquisición. Cada dB de SNR perdido por mal acondicionamiento equivale a perder 0.17 bits de resolución. Para datasets de IA, garbage in = garbage out.

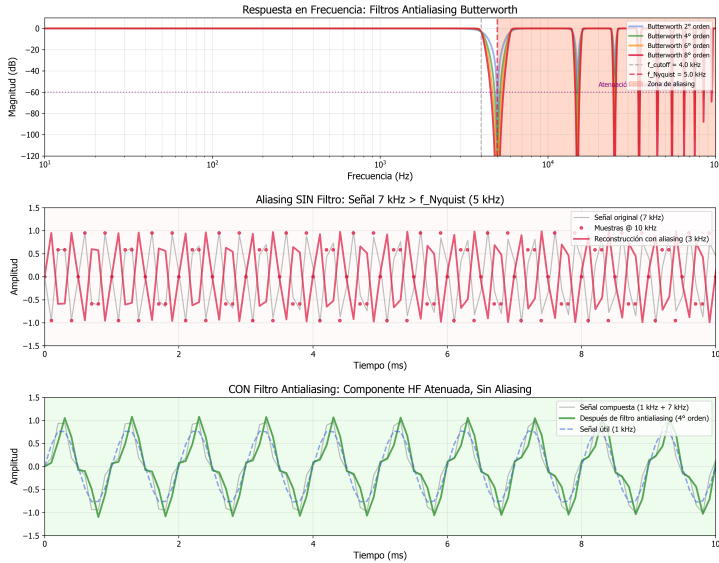


Figura 5.3: Filtro antialiasing. Butterworth 4to orden con $f_{\text{cutoff}} = 4$ kHz para $f_{\text{sample}} = 10$ kHz. Atenuación ≥ 60 dB @ f_{Nyquist} previene aliasing. Trade-off: mayor orden $\rightarrow$ mejor rechazo pero phase distortion. (s5_1_e3.py)

5.2 Microcontroladores y Arquitectura de Sistemas Embebidos

Objetivo de Aprendizaje

Objetivo: Analizar arquitecturas de microcontroladores (ARM Cortex-M, ESP32), periféricos (ADC, timers, DMA), y restricciones de tiempo real **para** diseñar sistemas embebidos que adquieran, procesen, y transmitan datos de sensores con latencia predecible para aplicaciones de IA en edge.

Ingeniería de Datos desde el Hardware

ESP32 @ 240 MHz ejecuta inferencia de red neuronal de 10 capas en 45 ms. Mismo modelo en ESP8266 @ 80 MHz toma 180 ms. Para sistema de detección de caídas que requiere respuesta < 100 ms, ESP32 funciona, ESP8266 falla. Frecuencia de reloj no es vanidad- es capacidad de procesamiento. Cada MHz extra reduce latencia, permite modelos más complejos, aumenta throughput de datos.

Microcontrolador integra CPU, memoria (Flash, RAM), y periféricos (ADC, UART, SPI, timers) en un chip. Arquitecturas comunes: ARM Cortex-M (STM32, nRF), RISC-V (ESP32-C3), Xtensa (ESP32). Selección depende de: velocidad de CPU, tamaño de memoria, periféricos disponibles, consumo de energía, ecosistema de desarrollo.

5.2.1 Arquitectura ARM Cortex-M y Periféricos

Cortex-M0+: 48 MHz, bajo consumo (< 10 μ A sleep). Cortex-M4: 80-180 MHz, FPU, DSP. Cortex-M7: 400+ MHz, cache. Periféricos críticos: ADC (12-16 bits, DMA), timers (PWM, capture), comunicación (UART, SPI, I2C @ DMA). DMA permite transferencias sin CPU, liberando ciclos para procesamiento. La Figura 5.4 muestra arquitectura típica.

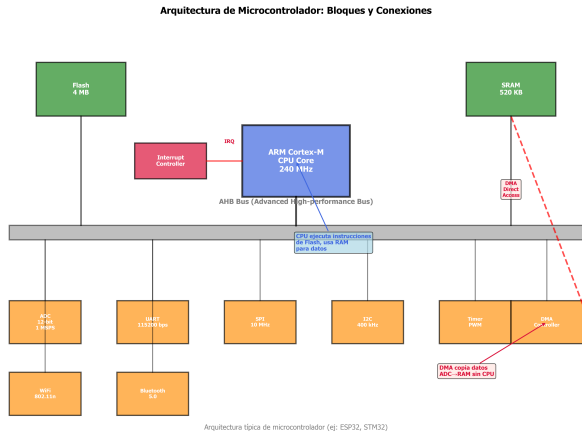


Figura 5.4: Arquitectura de microcontrolador. CPU conectada vía bus AHB a memoria y periféricos. DMA permite transferencias ADC→RAM sin intervención de CPU. Interrupt controller gestiona eventos asíncronos. (s5_2_e1.py)

5.2.2 ESP32: WiFi, Bluetooth, y Dual-Core

ESP32: dual-core Xtensa @ 240 MHz, 520 kB RAM, WiFi 802.11n, Bluetooth 5.0. Ideal para edge AI con conectividad. Core 0: tareas de red. Core 1: procesamiento de IA. ADC: 12-bit, 8 canales, hasta 1 MSPS con DMA. Consumo: 160 mA activo, < 10 μ A deep sleep. La Figura 5.5 compara performance.

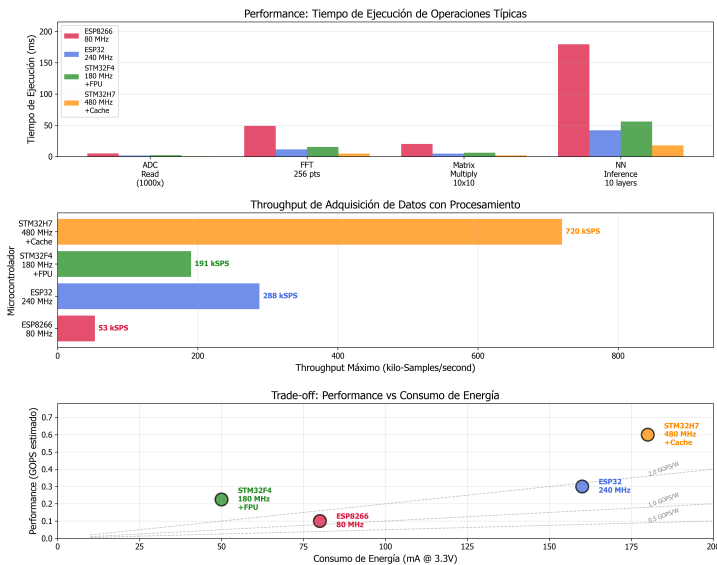


Figura 5.5: Performance de microcontroladores. ESP32 procesa 1000 muestras de FFT en 12 ms. STM32F4 @ 180 MHz en 8 ms. ESP8266 @ 80 MHz en 50 ms. Latencia determina viabilidad para aplicaciones de tiempo real. (s5.2.e2.py)

5.2.3 Tiempo Real y Determinismo

Sistema de tiempo real: respuesta predecible dentro de deadline. Hard real-time: falla si deadline se pierde (control de motor, airbag). Soft real-time: degradación gradual (procesamiento de audio, video). FreeRTOS provee scheduler para tareas con prioridades. Jitter típico: < 1 ms. La Figura 5.6 muestra timing de tareas.

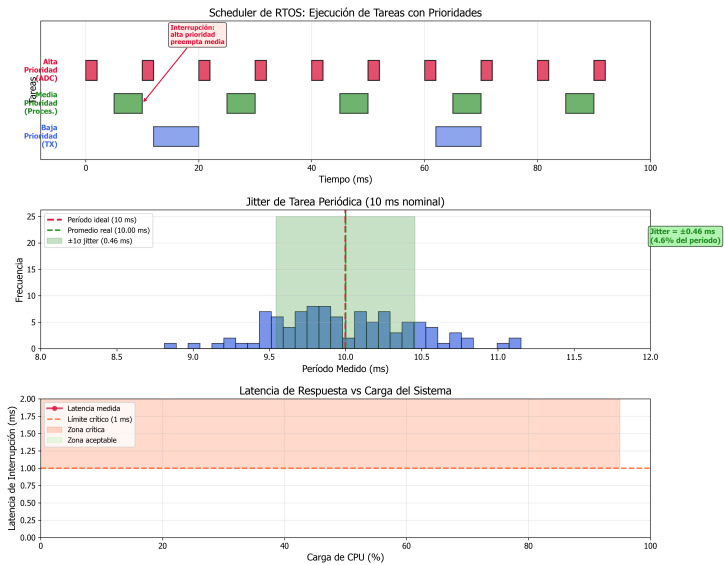


Figura 5.6: Timing de RTOS. Tarea de alta prioridad (adquisición ADC) interrumpe tarea de baja prioridad (procesamiento). Scheduler asegura dead-lines. Jitter medido con osciloscopio. (`s5_2_e3.py`)

Reto Computacional: Experimenta con el Script

Ejecuta `s5_2_e1.py`, `s5_2_e2.py`, `s5_2_e3.py`. Calcula throughput de sistema: ADC @ 10 kSPS, procesamiento FFT 256 puntos cada 25.6 ms, transmisión WiFi de resultados. ¿Qué frecuencia de CPU mínima se requiere?

Arquitectura de microcontrolador determina capacidad de procesamiento, latencia, y consumo. Para edge AI, balance entre performance y energía es crítico. ESP32 es sweet spot para prototipado; STM32 para producción optimizada.

5.3 Sensores para Sistemas de IA: IMU, Cámaras, y Micrófonos

Objetivo de Aprendizaje

Objetivo: Integrar sensores especializados (IMU, cámaras, micrófonos MEMS) con microcontroladores mediante protocolos I2C/SPI para adquirir datos multimodales (movimiento, visión, audio) que alimenten modelos de IA en aplicaciones de robótica, wearables, y monitoreo ambiental.

Ingeniería de Datos desde el Hardware

IMU MPU6050 genera 6 canales (accel XYZ + gyro XYZ) @ 1 kHz = 12 kB/s via I2C. Cámara OV7670 entrega 640x480 RGB @ 30 FPS = 27 MB/s. Micrófono MEMS @ 16 kHz PCM = 32 kB/s. Para sistema de detección de actividad humana, IMU es suficiente y consume < 1 mA. Para reconocimiento facial, cámara es necesaria pero consume 100+ mA. Selección de sensor determina bandwidth, energía, y complejidad de procesamiento.

Sensores modernos integran transductores, ADC, y comunicación digital. Ventaja: interfaz simple (I2C/SPI), calibración de fábrica, bajo ruido. Desventaja: menos control sobre adquisición, latencia de protocolo.

5.3.1 IMU (Inertial Measurement Unit)

IMU combina acelerómetro ($\pm 2g$ a $\pm 16g$) y giroscopio (± 250 a $\pm 2000^\circ/s$). MPU6050/6500 común en drones, wearables. I2C @ 400 kHz, rango configurable, FIFO interno. Aplicaciones: detección de gestos, conteo de pasos, orientación. La Figura 5.7 muestra señales de IMU.

5.3.2 Cámaras para Visión por Computadora

Cámaras de bajo costo: OV7670 (VGA), OV2640 (2MP). Interfaz: DVP (Digital Video Port, paralelo 8-bit) o MIPI CSI. Throughput alto requiere DMA o procesador dedicado (ESP32-CAM). Preprocesamiento: downsampling, conversión a grayscale reduce datos 10-20x. La Figura 5.8 muestra pipeline.

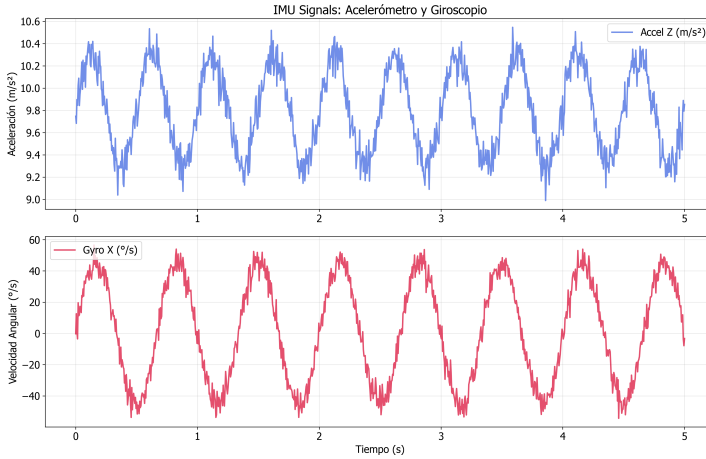


Figura 5.7: Señales de IMU durante actividad. Acelerómetro detecta movimiento, giroscopio detecta rotación. Frecuencia de muestreo típica: 100-1000 Hz. Filtro complementario fusiona ambos para orientación estable. (s5_3_e1.py)

5.3.3 Micrófonos MEMS y Audio

Micrófonos MEMS (ej: INMP441, SPH0645) con I2S digital. 16-bit PCM @ 16 kHz estándar para voz. Preprocesamiento: FFT para espectrograma, MFCC (Mel-Frequency Cepstral Coefficients) para speech recognition. La Figura 5.9 muestra extracción de features.

Reto Computacional: Experimenta con el Script

Ejecuta `s5_3_e1.py`, `s5_3_e2.py`, `s5_3_e3.py`. Calcula bandwidth para sistema multimodal: IMU @ 200 Hz + Audio @ 16 kHz + Cámara 96x96 @ 10 FPS. ¿Cabe en UART @ 115200 bps?

Sensores digitales simplifican interfaz pero agregan latencia de protocolo. Para tiempo real crítico (<1 ms), sensores analógicos con ADC directo son preferibles.

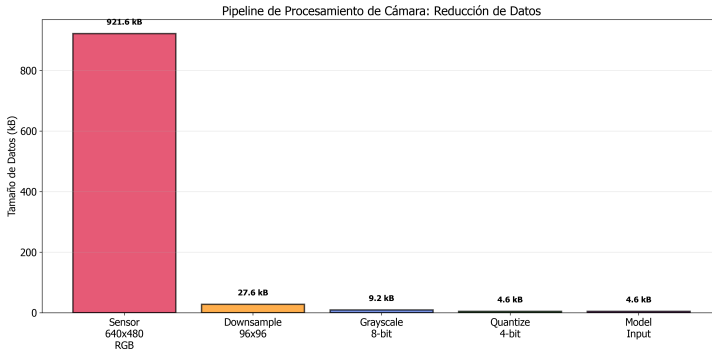


Figura 5.8: Pipeline de cámara. Sensor→ISP→DMA→RAM→Procesamiento. Downsampling de 640x480 a 96x96 reduce datos 40x. Quantización a 8-bit grayscale. (s5_3_e2.py)

5.4 Edge Computing y TinyML: Optimización para Microcontroladores

Objetivo de Aprendizaje

Objetivo: Implementar modelos de Machine Learning en microcontroladores mediante cuantización, pruning, y frameworks (TensorFlow Lite Micro, Edge Impulse) para ejecutar inferencia local con latencia < 100 ms y consumo < 10 mW, eliminando dependencia de cloud para aplicaciones críticas.

Ingeniería de Datos desde el Hardware

Red neuronal de 10 capas en float32: 400 kB modelo, 50 ms inferencia @ ESP32. Cuantizada a int8: 100 kB modelo, 12 ms inferencia, precisión baja solo 2%. Memoria se reduce 4x, velocidad aumenta 4x, energía baja 5x. Cuantización no es compresión lossy - es optimización específica para hardware entero de microcontroladores.

TinyML ejecuta ML en dispositivos con < 1 MB RAM, < 1 MHz. Ventajas: latencia ultra-baja (< 10 ms), privacidad (datos no salen del dispositivo), operación offline. Limitaciones: modelos pequeños, precisión reducida vs cloud.

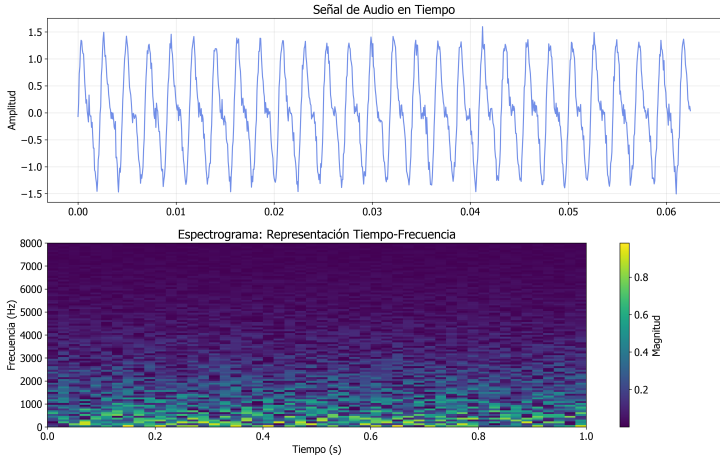


Figura 5.9: Audio features. Señal temporal $\rightarrow$ FFT $\rightarrow$ espectrograma. MFCC captura características de voz. Window típico: 25 ms con overlap de 10 ms. (s5_3_e3.py)

5.4.1 Cuantización y Compresión de Modelos

Cuantización convierte pesos float32 (4 bytes) a int8 (1 byte). Quantization-aware training (QAT) entrena con ruido de cuantización para mantener precisión. Post-training quantization (PTQ) cuantiza modelo ya entrenado. Típicamente: $< 3\%$ pérdida de precisión. La Figura 5.10 compara formatos.

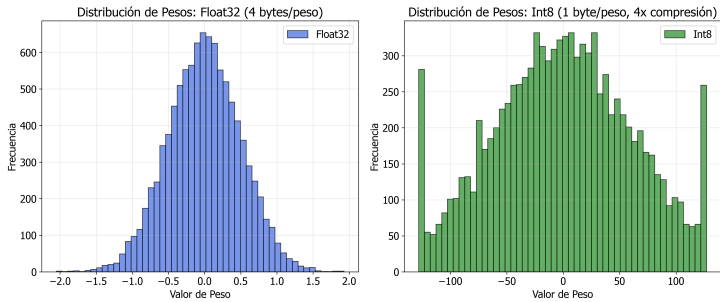


Figura 5.10: Cuantización de modelo. Float32 vs int8: tamaño 4x menor, inferencia 3-5x más rápida en MCU sin FPU. Distribución de pesos se mapea a rango int8 con factor de escala. (s5_4_e1.py)

5.4.2 Frameworks para Edge: TensorFlow Lite Micro

TFLite Micro: intérprete de ML para MCU. Modelos convertidos de TensorFlow/Keras. Operadores soportados: Conv2D, Dense, ReLU, MaxPool. Memoria estática: sin malloc. Footprint: < 20 kB código + modelo. La Figura 5.11 muestra workflow.

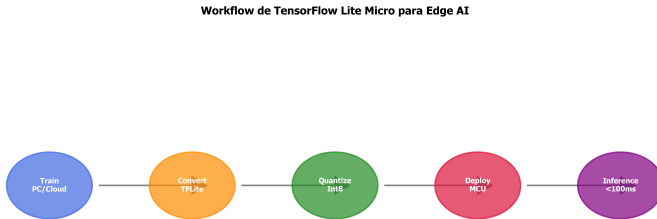


Figura 5.11: Workflow de TFLite Micro. Entrenar en PC → Convertir a .tflite → Cuantizar → Deploy en MCU. Inferencia ejecuta en < 100 ms para modelos típicos (10-20 capas). (`s5_4_e2.py`)

5.4.3 Performance y Trade-offs

Latencia de inferencia depende de: tamaño de modelo (capas, parámetros), tipo de operaciones (conv2d costoso), frecuencia de CPU. Throughput típico: 5-50 inferencias/segundo en MCU @ 200 MHz. Energía: 10-100 mJ por inferencia. La Figura 5.12 muestra trade-offs.

Reto Computacional: Experimenta con el Script

Ejecuta `s5_4_e1.py`, `s5_4_e2.py`, `s5_4_e3.py`. Calcula consumo de energía para clasificador de gestos: inferencia cada 100 ms @ 20 mJ/inferencia. Batería 500 mAh @ 3.7V. ¿Duración?

Edge computing permite IA distribuida: sensores inteligentes que procesan localmente y envían solo resultados (no raw data) a cloud. Reduce bandwidth 100-1000x, mejora privacidad, tolera desconexión de red.

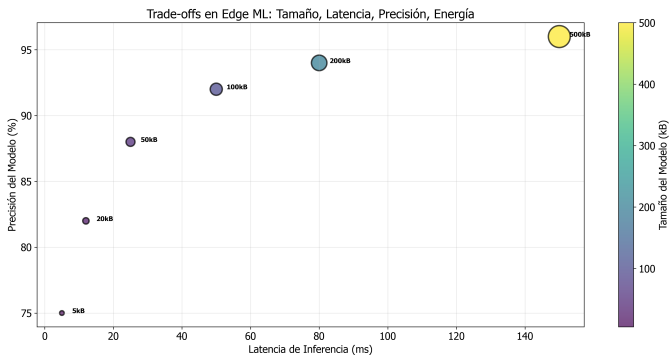


Figura 5.12: Trade-offs en edge ML. Precisión vs latencia vs energía. Modelos pequeños (10 kB) ejecutan en < 10 ms pero precisión < 85 %. Modelos grandes (500 kB) alcanzan > 95 % pero requieren > 100 ms. (s5_4_e3.py)

Apéndice A

Cronograma de Actividades y Prácticas

La siguiente planificación temporal está diseñada para cubrir las 15 semanas del periodo escolar, equilibrando la fundamentación teórica con la experimentación práctica orientada a la Ingeniería en Inteligencia Artificial.

El curso se divide en sesiones teóricas y sesiones de laboratorio. Las semanas de evaluación (7 y 15) están reservadas exclusivamente para exámenes y revisión de proyectos.

Distribución Semanal

Semana	Sesiones	Temas	Actividad
1	1, 2	1.1 Multímetro (Impedancia, RMS)1.2 Osciloscopio	Práctica 1
2	3, 4	1.3 Fuentes de Alimentación1.4 RLC y Resp. en Frecuencia	
3	5, 6	1.5 Leyes de Circuitos (Nodos/Mallas)1.6 Carga y Corriente	
4	7, 8	1.7 Teoremas de Redes (Thevenin)2.1 Diodo (Rectificación)	Práctica 2
5	9, 10	2.2 Transistor BJT (Zona Activa/Sat.)2.3 Transistor MOSFET	
6	11, 12	2.4 Amplificador Operacional	Práctica 3
7	13, 14	Primera Evaluación Parcial	Examen
8	15, 16	3.1 ADC: Conversión A/D3.2 DAC: Conversión D/A	Práctica 4
9	17, 18	3.3 PWM: Control de Potencia3.4 Protocolos (UART, SPI, I2C)	
10	19, 20	4.1 Fuentes y Regulación4.2 Baterías y Gestión Energía	
11	21, 22	4.3 Ruido Electromagnético4.4 Diseño de PCB	Práctica 5
12	23, 24	5.1 Acondicionamiento Señales5.2 Microcontroladores y Arq.	
13	25, 26	5.3 Sensores para IA (IMU, Cámaras)5.4 Edge Computing y TinyML	Práctica 6
14	27, 28	Integración Proyecto Final	Proyecto
15	29, 30	Segunda Evaluación Parcial	Examen

Descripción de las Prácticas de Laboratorio

Práctica 1: Caracterización Estadística del Ruido Térmico

Objetivo: Evaluar la distribución probabilística del ruido en resistencias de alto valor mediante mediciones masivas y análisis de histogramas para modelar el error gaussiano en sensores.

Vinculación con IA: Introducción a la naturaleza estocástica de las mediciones reales y comprensión del ruido como variable aleatoria que afecta la calidad de los datos de entrenamiento.

Práctica 2: Ingeniería Inversa de la No-Linealidad

Objetivo: Modelar matemáticamente la curva I-V de un diodo real mediante la recolección de puntos de operación para entrenar un modelo de regresión no lineal.

Vinculación con IA: Aplicación práctica de técnicas de machine learning para caracterizar componentes electrónicos y comprender funciones de activación no lineales.

Práctica 3: El Perceptrón Analógico (Hardware)

Objetivo: Construir un sumador ponderado con amplificadores operacionales mediante resistencias de precisión para implementar físicamente la ecuación $y = \sum w_i x_i + b$ antes de digitalizar.

Vinculación con IA: Implementación física de la operación fundamental de una neurona artificial, demostrando que las redes neuronales tienen una contraparte analógica en circuitos electrónicos.

Práctica 4: Fusión de Sensores y Calibración

Objetivo: Diseñar un arreglo multisensorial (Luz + Temperatura) mediante circuitos de acondicionamiento para crear un dataset etiquetado que correlacione variables ambientales.

Vinculación con IA: Generación de datasets multimodales y comprensión de la importancia del preprocesamiento analógico antes de la digitalización.

Práctica 5: Caracterización de Ruido y SNR en Cadenas de Adquisición

Objetivo: Medir y cuantificar el ruido electromagnético en diferentes configuraciones de circuitos de adquisición (cables sin blindar vs blindados, ground loops, filtrado capacitivo), calcular la relación señal-ruido (SNR) mediante osciloscopio y análisis estadístico, y diseñar estrategias de mitigación para maximizar la calidad de datos en sistemas de sensado para IA.

Vinculación con IA: Comprensión del impacto directo del SNR en la calidad de datasets de entrenamiento: un sistema con SNR de 40 dB vs

60 dB puede significar la diferencia entre un modelo de clasificación con 85 % vs 95 % de accuracy, demostrando que el ruido del hardware limita fundamentalmente el desempeño alcanzable por cualquier algoritmo de ML.

Práctica 6: Digitalización de Alta Fidelidad

Objetivo: Configurar un ADC externo y transmisión UART mediante microcontrolador para evitar el aliasing y la pérdida de información en la adquisición.

Vinculación con IA: Comprensión del teorema de muestreo de Nyquist y su impacto en la calidad de los datos que alimentan modelos de IA.

Proyecto Final: Smart Edge Sensor

Descripción: Sistema completo que integra: Sensor → Filtro Analógico → ADC → Preprocesamiento Digital → Modelo de IA en PC.

Objetivo: Integrar todos los conocimientos del curso en un sistema funcional de adquisición y procesamiento inteligente de datos, desde el mundo físico hasta la inferencia con modelos de IA.

Apéndice B

Actividades de Aula Invertida

CONFIDENCIAL - MATERIAL EXCLUSIVO PARA DOCENTES

Este apéndice contiene actividades diseñadas para mantener a los estudiantes comprometidos mediante dinámicas que no requieren equipo de laboratorio y que no pueden resolverse consultando una inteligencia artificial. Cada actividad desarrolla habilidades profesionales críticas: argumentación técnica, comunicación efectiva, pensamiento sistémico y toma de decisiones bajo presión. Los estudiantes no tienen acceso a este material para preservar el elemento sorpresa y evitar que preparen respuestas prefabricadas.

Las actividades están diseñadas con mecanismos que dificultan el uso de inteligencia artificial. El tiempo limitado de diez a treinta minutos no permite consultas extensivas. El componente oral obligatorio mediante presentaciones y defensas presenciales no es delegable. Los casos específicos son creados por el profesor y no están disponibles en internet. La interdependencia entre equipos hace que el trabajo de uno alimente al siguiente. La evaluación valora el proceso de argumentación más que la respuesta final correcta.

B.1 Unidad 1: Instrumentación

B.1.1 1.1: El Multímetro (30 min)

Los estudiantes se dividen inicialmente en cuatro grupos de expertos donde cada grupo estudia durante diez minutos un aspecto del multímetro usando material preparado previamente: medición de voltaje continuo y alterno con el concepto de valor eficaz verdadero, medición de corriente y fusibles de protección, medición de resistencia e impedancia de entrada del instrumento, y prueba de continuidad y diodos. Luego se reorganizan en grupos mixtos de cuatro personas donde cada experto enseña su tema a los demás en tres o cuatro minutos usando pizarra o papel para diagramas. Finalmente un cuestionario rápido de cuatro preguntas evalúa al grupo completo donde todos deben saber todo porque la calificación es grupal, creando presión social para que cada estudiante preste atención durante la fase de enseñanza mutua.

B.1.2 1.2: El Osciloscopio (25 min)

Seis estaciones distribuidas en el aula presentan afirmaciones técnicas sobre especificaciones del osciloscopio donde algunas son correctas y otras contienen errores sutiles. Los equipos rotan por las estaciones dedicando tres minutos a cada una donde discuten si la afirmación es verdadera o falsa y escriben su justificación en una nota adhesiva. Al final una plenaria de diez minutos revela las respuestas correctas y discute cada estación evaluando la profundidad de la justificación técnica y no solo el acierto. Las afirmaciones incluyen trampas como «un osciloscopio de cien megahertz puede medir perfectamente una señal cuadrada de cincuenta megahertz» que es falsa porque la señal cuadrada tiene armónicos hasta cinco o siete veces la frecuencia fundamental y necesitarías más de trescientos cincuenta megahertz para reproducirla fielmente.

B.1.3 1.3: Fuentes de Alimentación (20 min)

Cada equipo recibe una especificación técnica de fuente y debe explicarla a otro equipo que actúa como gerente de proyecto sin formación en electrónica, luego se invierte donde el cliente describe una necesidad en lenguaje coloquial y el equipo técnico debe traducir a especificaciones precisas. Por ejemplo «necesito que no haga ruido eléctrico que afecte mis sensores» se traduce a especificación de rizado menor a cincuenta milivolts con posible filtrado adicional y aislamiento galvánico. El equipo cliente califica la claridad de la explicación y el profesor califica la precisión técnica desarrollando una habilidad crítica que ninguna inteligencia artificial puede replicar: adaptar el lenguaje técnico a la audiencia en tiempo real mediante interacción

humana.

B.1.4 1.4: Filtros Pasivos (35 min)

Similar a la actividad del multímetro pero con mayor complejidad donde los grupos de expertos estudian durante quince minutos filtros pasa-bajas, pasa-altas, resonante serie y resonante paralelo. En grupos mixtos durante quince minutos cada experto enseña su filtro incluyendo qué frecuencias pasan o bloquean, cuándo usarlo y cálculo rápido de componentes. El desafío integrador final requiere conocimiento de los cuatro filtros: diseñar una cascada de filtros para adquisición de señal cardíaca donde la señal útil está en cero punto cinco a cien hertz pero tienes ruido de cincuenta o sesenta hertz de línea eléctrica, voltaje continuo de electrodos y ruido de conmutación mayor a diez kilohertz. La respuesta esperada combina un pasa-altas de cero punto cinco hertz para bloquear el voltaje continuo, un filtro de rechazo de banda de cincuenta o sesenta hertz con resonante serie a tierra, y un pasa-bajas de cien hertz para bloquear el ruido de conmutación.

B.1.5 1.5: Leyes de Kirchhoff (20 min)

Cinco estaciones con circuitos dibujados y afirmaciones incorrectas sobre aplicación de las leyes de tensión y corriente de Kirchhoff donde los equipos rotan, discuten, justifican por escrito y finalmente hay plenaria. Las trampas incluyen «en circuitos en serie la corriente en la primera resistencia es el doble que en la segunda porque la primera resistencia es la mitad» que es falso porque en serie la corriente es igual en todos los componentes, y «en paralelo la resistencia total es la suma de resistencias individuales» cuando en realidad la resistencia equivalente es menor que la menor resistencia del arreglo.

B.1.6 1.6: Carga, Tensión y Corriente (25 min)

Los equipos deben diseñar en veinte minutos un sistema de distribución de potencia para un robot móvil con cuatro subsistemas: motores de doce volts a dos amperes pico, computadora de cinco volts a tres amperes, sensores de tres punto tres volts a quinientos miliamperes, y servos de seis volts a un ampere pico. Las restricciones incluyen presupuesto de trescientos pesos, eficiencia mayor a setenta y cinco por ciento, protección contra cortocircuito en cada línea y placa de circuito impreso de diez por diez centímetros máximo. Los entregables son diagrama de bloques, especificación de cada regulador con topología y componente específico, cálculo de corriente total y autonomía esperada, y lista de materiales estimada con costos. Cada equipo presenta en dos minutos y otros equipos hacen preguntas desafiantes como «qué pasa si los cuatro motores arrancan simultáneamente, tu

batería puede dar el pico de corriente»o «usaste regulador lineal para doce volts a cinco volts, cuántos watts se disipan como calor, necesitas disipador». La votación final elige el mejor diseño según robustez, eficiencia, costo y simplicidad.

B.1.7 1.7: Serie y Paralelo (30 min)

Un caso de resolución colaborativa en relevos con tres equipos trabajando secuencialmente donde el problema es una cadena de cincuenta diodos emisores de luz donde solo se encienden treinta. El primer equipo tiene diez minutos para analizar la topología del circuito, identificar cómo están conectados los grupos y plantear hipótesis inicial sobre si el fallo está en conexión serie o paralelo entregando análisis escrito al segundo equipo. El segundo equipo usa mediciones de voltaje simuladas para localizar el grupo fallido y determinar si es diodo abierto o resistencia quemada entregando diagnóstico al tercer equipo. El tercer equipo propone solución de reparación y calcula nueva resistencia limitadora si reemplaza un grupo completo. El giro es que cada equipo debe usar el trabajo del anterior y si el primer equipo se equivoca los equipos dos y tres deben detectarlo y corregir.

B.1.8 1.8: Medición en Arreglos de Resistencias (30 min)

Un equipo presenta en cinco minutos un diseño de circuito de monitoreo de temperatura con ocho termistores distribuidos en un cuarto de servidores que debe detectar sobrecalentamiento mayor a treinta y cinco grados. Otros equipos actúan como interesados: el gerente de tecnología pregunta cómo saber qué termistor detectó sobrecalentamiento y qué pasa si un termistor falla, el ingeniero de manufactura pregunta cómo calibras ocho sensores con tolerancia de cinco por ciento y si la resistencia del cable afecta la medición, finanzas pregunta qué justifica el costo extra versus competidores, y seguridad pregunta qué pasa si el circuito falla y no detecta sobrecalentamiento y si puede causar chispa en ambiente con polvo. La votación final requiere mayoría para aprobar el diseño para producción evaluando claridad de presentación, solidez técnica del diseño, capacidad de responder bajo presión y justificación de compromisos.

B.2 Unidad 2: Semiconductores

B.2.1 2.1: Diodos (25 min)

Seis estaciones con trampas sobre características de diodos donde las afirmaciones erróneas incluyen «un diodo conduce perfectamente a cero voltios», «diodo Schottky es mejor que diodo de señal para todas las aplicaciones», «en rectificador de onda completa la frecuencia del rizado es igual a la frecuencia de línea», y «puedo conectar diodos en serie para aumentar corriente soportada» que es especialmente engañosa porque serie aumenta voltaje soportado pero la corriente es la misma limitada por el diodo más débil mientras que para aumentar corriente se conectan en paralelo con resistencias de balance.

B.2.2 2.2: Transistor Bipolar versus Transistor de Efecto de Campo (30 min)

Un debate donde dos equipos defienden topologías opuestas para control por modulación de ancho de pulso de motor de corriente continua de doce volts a cinco amperes. El primer equipo defiende transistor bipolar tipo Darlington argumentando costo menor, no necesita manejador complejo y es más robusto contra descarga electrostática pero acepta disipación alta de cinco watts que necesita disipador grande y ganancia variable con temperatura. El segundo equipo defiende transistor de efecto de campo argumentando eficiencia alta con solo medio watt de disipación, conmutación rápida para modulación de alta frecuencia y control por voltaje que no carga la salida digital pero acepta costo mayor y sensibilidad al voltaje de compuerta. El moderador introduce giro a mitad del debate: el sistema debe funcionar de menos diez a sesenta grados centígrados donde los equipos deben explicar cómo la temperatura afecta cada diseño porque el transistor bipolar pierde ganancia con temperatura mientras el transistor de efecto de campo aumenta resistencia pero sigue siendo más eficiente. La audiencia vota por el equipo más convincente según solidez de argumentos técnicos y capacidad de adaptar argumento ante nueva información.

B.2.3 2.3: Fallo de Transistor de Potencia (25 min)

Un análisis forense donde equipos compiten por diagnosticar primero una fuente elevadora de cinco volts a doce volts que funcionó tres días y luego el transistor se quemó literalmente con encapsulado abierto y silicio visible. La evidencia incluye diagrama del circuito, foto del transistor destruido, hoja de datos, registro de voltajes y forma de onda de la compuerta capturada antes del fallo mostrando onda cuadrada de cero a cinco volts. Las pistas

revelan que el sistema funciona con batería de polímero de litio que puede dar picos de corriente muy altos, temperatura ambiente de cuarenta y cinco grados en caja plástica al sol sin disipador, y ciclo de trabajo medido de ochenta y cinco por ciento. La causa raíz es saturación incompleta del transistor porque voltaje compuerta-fuente de cinco volts es insuficiente donde la hoja de datos muestra que la resistencia especificada de cero punto cero dos ocho ohms es a voltaje compuerta-fuente de diez volts pero a cinco volts es típicamente cero punto uno ohms generando con corriente de cinco amperes una disipación de dos punto cinco watts que a cuarenta y cinco grados sin disipador alcanza más de ciento cincuenta grados en la unión causando escape térmico donde la resistencia aumenta con temperatura generando más disipación y más temperatura hasta fallo catastrófico. Las soluciones propuestas incluyen usar transistor de nivel lógico, agregar manejador de compuerta que genere voltaje compuerta-fuente de diez volts desde entrada de cinco volts con circuito auxiliar, agregar disipador y reducir ciclo de trabajo objetivo.

B.2.4 2.4: Amplificador con Amplificador Operacional (35 min)

Un equipo presenta en ocho minutos diseño de circuito de amplificación para micrófono de condensador con salida típica de diez milivolts pico a pico que alimentará convertidor analógico-digital de doce bits con rango de cero a tres punto tres volts para reconocimiento de voz en dispositivo embebido. El diseño debe incluir topología inversor o no-inversor o diferencial con ganancia calculada y justificada, polarización para señal alterna con alimentación simple de tres punto tres volts, filtrado y rechazo de ruido fuera de banda de voz de trescientos a tres mil cuatrocientos hertz, componentes específicos con amplificador operacional justificado según producto ganancia-ancho de banda, velocidad de respuesta, ruido y costo, y lista de materiales. Los interesados atacan: el arquitecto de aprendizaje automático pregunta cuál es el ruido del circuito porque el modelo necesita relación señal-ruido mayor a cuarenta decibels y si el filtro distorsiona formantes de voz, el ingeniero de firmware pregunta si el convertidor recibe señal centrada en uno punto sesenta y cinco volts o en cero volts y qué pasa si el usuario grita con cien milivolts de entrada, confiabilidad de hardware pregunta cuál es la vida útil de capacitores electrolíticos a cincuenta grados y qué pasa si el usuario toca el micrófono sin protección contra descarga electrostática, y costos pregunta por qué no usar amplificador operacional de cinco pesos en lugar de uno que cuesta veinticinco pesos. La votación requiere mayoría para aprobar diseño para prototipo y si rechazan el equipo tiene cinco minutos para proponer modificación y re-presentar.

B.3 Unidad 3: Conversión ADC/DAC

B.3.1 3.1: Arquitecturas de Convertidor Analógico-Digital (30 min)

Un debate sobre selección de convertidor analógico-digital para monitoreo de vibración en motor industrial donde la señal es acelerómetro analógico de cero a cinco volts con corriente continua hasta cinco kilohertz, el objetivo es detectar patrones de fallo como desgaste de rodamiento y desbalanceo para crear conjunto de datos de modelo predictivo, el costo objetivo es menor a cien pesos por canal y se necesitan cuatro canales simultáneos. El primer equipo defiende convertidor de aproximaciones sucesivas de doce bits a cien mil muestras por segundo argumentando velocidad alta que da margen sobre las diez mil muestras necesarias por teorema de muestreo, bajo costo de treinta pesos y fácil interfaz serial pero acepta resolución limitada donde doce bits dan uno punto dos dos milivolts por bit a cinco volts que puede perder detalles finos y relación señal-ruido típica de setenta y dos decibels. El segundo equipo defiende convertidor sigma-delta de dieciséis bits a diez mil muestras por segundo argumentando alta resolución de setenta y seis microvolts por bit que detecta vibraciones sutiles, relación señal-ruido típica de noventa decibels que genera conjunto de datos más limpio para aprendizaje automático y filtro digital integrado pero acepta velocidad justa donde diez mil muestras da límite de teorema de muestreo de cinco kilohertz exacto sin margen, costo mayor de ochenta pesos y latencia de grupo por retardo de filtro de un milisegundo. El giro es que el motor tiene variador de frecuencia que inyecta ruido de conmutación a veinte kilohertz en las líneas de alimentación donde el convertidor de aproximaciones sucesivas es más sensible a fluctuación de reloj y ruido en referencia de voltaje mientras sigma-delta con filtro digital integrado rechaza ruido fuera de banda automáticamente por sobremuestreo y decimación. La votación valora argumentos técnicos cuantitativos con cálculos de relación señal-ruido, costos y márgenes más que argumentos cualitativos.

B.3.2 3.2: Solapamiento Espectral en Sistema Real (30 min)

Un análisis forense de modelo de clasificación de señal cardíaca con sesenta y cinco por ciento de exactitud cuando debería superar noventa por ciento donde el sistema usa convertidor de doce bits a quinientos hertz que es suficiente para señal cardíaca de cero punto cinco a cien hertz, filtro pasa-bajas simple con frecuencia de corte de ciento cincuenta hertz, conjunto de datos de diez mil muestras etiquetadas y red neuronal convolucional unidimensional entrenada con exactitud en conjunto de validación de solo sesenta y cinco por ciento. La evidencia muestra transformada rápida de Fourier

de señal capturada con pico anómalo a ciento cincuenta hertz que no debería existir en señal cardíaca real, ambiente de hospital con iluminación de diodos emisores con conmutación a treinta kilohertz y equipos médicos, osciloscopio en entrada del convertidor muestra señal visualmente limpia, y al agregar filtro de cuarto orden a cien hertz el pico a ciento cincuenta hertz desaparece. Los equipos deben identificar que la fuente del pico es solapamiento espectral de ruido de conmutación de diodos donde treinta kilohertz muestreados a quinientos hertz generan alias según frecuencia alias igual a valor absoluto de n por frecuencia de muestreo más o menos frecuencia de señal donde con n igual a sesenta el alias es treinta kilohertz módulo quinientos hertz igual a ciento cincuenta hertz exactamente. El filtro de primer orden a ciento cincuenta hertz atenúa solo doce decibeles por octava dando solo cuarenta y seis decibeles a treinta kilohertz que es insuficiente mientras el osciloscopio no mostraba el problema porque treinta kilohertz puede estar fuera de su ancho de banda o el display promedia. La solución es filtro de cuarto orden a frecuencia de corte de cien hertz dando atenuación de ochenta decibeles a treinta kilohertz suficiente para reducir ruido por debajo de un bit del convertidor. El impacto en aprendizaje automático es que el pico a ciento cincuenta hertz es característica espuria no relacionada con condición cardíaca donde el modelo aprende correlación falsa entre ruido de diodos y etiqueta, al desplegar en ambiente diferente sin esos diodos específicos la exactitud colapsa, y con conjunto de datos limpio después de solución la exactitud sube a noventa y dos por ciento.

B.3.3 3.3: Convertidor Digital-Analógico para Audio (25 min)

Un diseño rápido de generador de tonos de advertencia para dispositivo médico portátil que debe generar tonos de doscientos hertz a cuatro kilohertz con calidad suficiente para ser inteligibles donde las especificaciones son microcontrolador a ciento ochenta megahertz con convertidor digital-analógico interno de doce bits, altavoz de ocho ohms a medio watt, alimentación batería de tres punto siete volts, presupuesto de componentes externos menor a cincuenta pesos y consumo menor a cincuenta miliamperes promedio porque batería de quinientos miliamperes-hora debe durar más de diez horas. Las restricciones de diseño incluyen distorsión armónica total menor a diez por ciento para inteligibilidad, latencia menor a diez milisegundos desde activación hasta sonido audible y no usar amplificadores de audio integrados por costo. Los entregables en veinte minutos son diagrama de bloques desde convertidor pasando por filtro, amplificador y altavoz, cálculo de frecuencia de muestreo del convertidor justificado según teorema de muestreo y calidad, diseño de filtro pasa-bajas con frecuencia de corte y valores de componentes, selección de amplificador como amplificador operacional más transistor o clase AB o modulación de ancho de pulso

directo, cálculo de potencia y eficiencia, y lista de materiales con costos. El jurado hace preguntas como «tu frecuencia de muestreo es ocho kilohertz para generar cuatro kilohertz no viola teorema de muestreo» donde la respuesta es que el teorema dice mayor a dos veces pero ocho mil es apenas dos veces y menor a diez veces puede tener distorsión alta, «usas amplificador operacional más transistor en configuración complementaria cómo eliminas distorsión de cruce» donde la respuesta es polarización de clase AB con diodos, «tu filtro es primer orden es suficiente para atenuar imágenes del convertidor a frecuencia de muestreo» donde depende de calcular atenuación específica, y «consumo de cuarenta y cinco miliamperes promedio incluye consumo del microcontrolador generando muestras» donde debe incluir costo computacional.

B.3.4 3.4: Control de Motor por Modulación de Ancho de Pulso (35 min)

Un caso colaborativo en relevos de motor de corriente continua con modulación de ancho de pulso que tiene vibración anómala a ciclo de trabajo de cincuenta a setenta por ciento cuando no debería donde el sistema es motor de doce volts a dos amperes con inductancia de cinco milihenries, manejador de transistor controlado por salida digital de microcontrolador, modulación a un kilohertz con ciclo de trabajo variable de cero a cien por ciento, y el problema es que a ciclo de trabajo de cincuenta a setenta por ciento el motor vibra y hace ruido audible. La evidencia incluye forma de onda de la modulación en compuerta del transistor, forma de onda de corriente en motor medida con resistor de medición, espectro de frecuencia de corriente por transformada rápida de Fourier y código del firmware con configuración de temporizador. El primer equipo en diez minutos examina formas de onda, identifica anomalías como frecuencia incorrecta o sobrepico, plantea hipótesis sobre si es problema eléctrico o mecánico y entrega análisis al segundo equipo. El segundo equipo en diez minutos examina código de configuración del temporizador, identifica configuración incorrecta, propone solución y entrega al tercer equipo. El tercer equipo en diez minutos valida que la solución resuelve problema, propone mejoras adicionales como diodo de protección o capacitor de filtrado y presenta solución final. La causa raíz es frecuencia de modulación a un kilohertz está en rango audible de veinte hertz a veinte kilohertz donde motor y estructura mecánica actúan como transductor acústico, a ciclo de trabajo de cincuenta a setenta por ciento los armónicos de la modulación a dos kilohertz y tres kilohertz excitan resonancia mecánica del motor, transformada de Fourier muestra pico a un kilohertz y armónicos evidenciando conmutación audible, y corriente tiene fluctuación alta por inductancia pequeña del motor causando vibración electromagnética. El análisis de código revela temporizador configurado para un kilohertz pero por error de redondeo la frecuencia real es

novecientos ochenta hertz. Las soluciones son aumentar frecuencia de modulación a mayor a veinte kilohertz fuera de rango audible reconfigurando temporizador a veinticinco kilohertz eliminando vibración acústica, agregar diodo de protección en paralelo al motor con polaridad correcta para proteger transistor de pico de voltaje cuando conmutación apaga, y capacitor de desacople de cien microfaradios más cien nanofaradios en paralelo en alimentación del motor para reducir caída de voltaje durante conmutación generando corriente más suave.

B.3.5 3.5: Protocolo de Comunicación (30 min)

Un equipo presenta en cinco minutos diseño de sistema de sensado distribuido para agricultura de precisión con veinte nodos sensores de temperatura, humedad y pH de suelo distribuidos en área de cien por cien metros que reportan datos cada diez minutos, puerta de enlace central recolecta datos y sube a nube, y alimentación solar más batería con autonomía de siete días sin sol. El diseño debe especificar protocolo seleccionado entre opciones de corto alcance serial, radio de largo alcance, WiFi o Bluetooth con justificación de alcance, consumo, tasa de transferencia, costo y complejidad, topología como estrella, bus, malla o punto a punto, formato de paquete con estructura de datos y suma de verificación, consumo energético con cálculo de energía por transmisión y autonomía, y lista de materiales por nodo. Los interesados son ingeniero de firmware preguntando cómo manejas colisiones si dos nodos transmiten simultáneamente, si un nodo falla se cae toda la red o solo ese nodo, y si implementaste retransmisión automática con cuántos reintentos, administración de energía preguntando si siete días de autonomía incluye días nublados con carga parcial, si el módulo de comunicación entra en suspensión profunda entre transmisiones con cuánto consumo en suspensión, y cuánta energía consume despertar de suspensión y hacer establecimiento de conexión de red, despliegue preguntando cómo agrego un nodo nuevo al sistema sin reconfigurar todos los demás, cómo diagnostico remotamente si la antena de un nodo se daña, y si el protocolo soporta actualización de firmware por aire, y costos preguntando si el módulo de radio a ciento cincuenta pesos por nodo con veinte nodos da tres mil pesos mientras WiFi no es más barato, y si necesitas licencia o suscripción para usar el protocolo o frecuencia. La votación requiere mayoría para aprobar para despliegue piloto y si rechazan el equipo propone modificación rápida.

B.4 Unidad 4: Potencia y Diseño PCB

B.4.1 4.1: Regulador Lineal versus Regulador Conmutado (30 min)

Un debate sobre topología de regulador para dispositivo de internet de las cosas alimentado por batería de ion de litio de cuatro punto dos volts cargada a tres punto cero volts descargada que debe generar tres punto tres volts a quinientos miliamperes para microcontrolador y sensores. El primer equipo defiende regulador de baja caída argumentando simplicidad extrema de tres pines y dos capacitores externos, ruido ultra-bajo menor a cien microvolts eficaces ideal para convertidor analógico-digital y radiofrecuencia, costo bajo de cinco pesos y respuesta rápida a transientes de carga pero acepta eficiencia variable donde eficiencia igual a voltaje de salida sobre voltaje de entrada da setenta y nueve por ciento a cuatro punto dos volts a tres punto tres volts e imposible a tres punto cero volts porque caída de uno punto dos volts típico, disipación de diferencia de voltaje por corriente igual a cero punto nueve volts por cero punto cinco amperes igual a cero punto cuarenta y cinco watts que necesita disipador, y no funciona cuando batería baja de tres punto tres volts más caída. El segundo equipo defiende convertidor reductor conmutado argumentando alta eficiencia mayor a noventa por ciento en todo el rango de voltaje de entrada, bajo calor sin necesidad de disipador, y funciona hasta que batería llega a protección de bajo voltaje de dos punto cinco volts típico usando más capacidad de batería pero acepta costo mayor de treinta y cinco pesos más inductor de ocho pesos más capacitores de tres pesos dando cuarenta y seis pesos total, ruido de conmutación con fluctuación de cincuenta milivolts a frecuencia de conmutación generando interferencia electromagnética, y complejidad requiriendo inductor como componente más grande y diseño de placa crítico. El moderador introduce que el dispositivo debe funcionar en ambiente industrial de menos veinte a setenta grados centígrados donde el regulador de baja caída tiene voltaje de caída que aumenta con temperatura baja pudiendo fallar a menos veinte grados con batería parcialmente descargada mientras convertidor reductor tiene eficiencia que disminuye levemente pero sigue funcional y el inductor es más robusto térmicamente. La batería pierde capacidad a menos veinte grados quedando con cincuenta a setenta por ciento de capacidad nominal lo que favorece convertidor reductor porque usa más capacidad disponible. El segundo giro es que el microcontrolador tiene modo de suspensión profunda de diez microamperes noventa por ciento del tiempo y activo de quinientos miliamperes diez por ciento del tiempo donde el regulador de baja caída tiene corriente de reposo típica de cinco miliamperes desperdiciando energía en suspensión mientras convertidor reductor con buen diseño tiene corriente de reposo de veinte microamperes en modo de baja potencia a baja carga favoreciendo aún más el convertidor

reductor en cálculo de energía promedio.

B.4.2 4.2: Batería se Drena Rápido (30 min)

Un análisis forense de reloj inteligente con autonomía de seis horas cuando la especificación es cuarenta y ocho horas donde el dispositivo usa batería de polímero de litio de trescientos miliamperes-hora a tres punto siete volts, microcontrolador con Bluetooth de bajo consumo, pantalla de diodos orgánicos de una punto tres pulgadas, sensores acelerómetro y frecuencia cardíaca óptico, y especificación de cuarenta y ocho horas con uso típico donde pantalla encendida cinco por ciento, Bluetooth conectado y frecuencia cardíaca cada diez minutos. Los prototipos drenan batería en seis horas y usuarios reportan dispositivo tibio al tacto. La medición de corriente total muestra suspensión profunda con cinco microamperes esperado pero dos punto cinco miliamperes medido, pantalla encendida con quince miliamperes esperado pero dieciocho miliamperes medido, Bluetooth transmitiendo con ocho miliamperes pico esperado pero nueve miliamperes pico medido, y sensor de frecuencia cardíaca activo con cinco miliamperes esperado pero veinticinco miliamperes medido. La foto térmica infrarroja muestra componente específico más caliente diez grados sobre ambiente. El consumo anómalo principal es sensor de frecuencia cardíaca a veinticinco miliamperes que es cinco veces esperado donde la causa raíz es diodo emisor infrarrojo del sensor no está en atenuación por modulación de ancho de pulso sino encendido continuo. El sensor óptico usa diodo emisor infrarrojo más fotodiodo y la hoja de datos dice «diodo emisor debe operarse en modulación a cien hertz con ciclo de trabajo ajustable según señal»pero el código tiene error donde diodo emisor está configurado como salida digital alta continua generando corriente de veinte miliamperes a cien por ciento de ciclo de trabajo mientras con modulación a diez por ciento de ciclo de trabajo suficiente para señal la corriente promedio es dos miliamperes. El consumo anómalo secundario es suspensión profunda a dos punto cinco miliamperes que es quinientas veces esperado donde la causa raíz es resistencia de elevación en línea de bus serial nunca deshabilitada conectando acelerómetro que tiene resistencias de elevación de diez kilohms a tres punto tres volts y cuando microcontrolador y acelerómetro entran en suspensión las líneas del bus flotan donde las resistencias de elevación mantienen líneas a tres punto tres volts pero acelerómetro en suspensión tiene pines en alta impedancia generando corriente de fuga a través de resistencias de elevación de tres punto tres volts sobre diez kilohms igual a trescientos treinta microamperes por línea por dos igual a seiscientos sesenta microamperes más acelerómetro en suspensión con bus activo consumiendo otros dos miliamperes en modo incorrecto. Las soluciones son corregir firmware con modulación para diodo emisor infrarrojo con ciclo de trabajo diez por ciento, poner acelerómetro en verdadero modo de apagado no solo suspensión, y opcional agregar

transistores para desconectar resistencias de elevación durante suspensión profunda mejorando autonomía de seis horas a aproximadamente cincuenta horas.

B.4.3 4.3: Ruido Electromagnético (25 min)

Seis estaciones con trampas sobre interferencia electromagnética, lazos de tierra y blindaje. Las afirmaciones erróneas incluyen «cable blindado elimina cien por ciento del ruido electromagnético» cuando en realidad el blindaje atenúa campos electromagnéticos típicamente veinte a sesenta decibeles dependiendo de frecuencia y calidad del blindaje y si el blindaje no está correctamente conectado a tierra en un solo punto puede empeorar ruido, «lazo de tierra se resuelve conectando todas las tierras juntas en múltiples puntos» cuando lazo de tierra se causa por múltiples conexiones a tierra y se resuelve usando conexión de tierra en un solo punto o plano de tierra con cuidado en trazado porque múltiples puntos crean lazos donde corrientes de tierra circulan inducidas por campos magnéticos, «cuenta de ferrita bloquea ruido de alta frecuencia actuando como capacitor» cuando cuenta de ferrita actúa como resistor a alta frecuencia disipando energía como calor por pérdidas magnéticas no como capacitor porque a baja frecuencia es inductor y a alta frecuencia es resistor con resistencia aproximadamente cien ohms a cien megahertz típico, «par trenzado rechaza ruido porque las inductancias se cancelan» cuando par trenzado rechaza ruido de modo común porque las áreas de lazo se cancelan donde campo magnético induce voltajes opuestos en cada vuelta del par que se cancelan en modo diferencial no por inductancias sino por áreas de lazo, y «agregar más capacitores de desacople siempre mejora el ruido» cuando capacitores en paralelo pueden resonar a ciertas frecuencias en anti-resonancia empeorando impedancia y la estrategia correcta es usar valores diferentes como cien nanofaradios más diez microfaradios más cien microfaradios para cubrir rangos de frecuencia sin resonancias peligrosas.

B.4.4 4.4: Diseño de Placa de Circuito Impreso (35 min)

Un equipo presenta en ocho minutos diseño de placa de circuito impreso de dos capas para registrador de datos de internet de las cosas con microcontrolador, convertidor analógico-digital externo de dieciséis bits por bus serial, convertidor reductor conmutado de cinco volts a tres punto tres volts a un ampere, conector para cuatro sensores analógicos externos con cables hasta dos metros, y antena WiFi integrada en placa. El diseño en papel o pizarra debe mostrar apilamiento con asignación de capas entre superior de señal y inferior de plano de tierra versus señal, colocación de ubicación de componentes críticos, trazado de líneas de potencia y señales sensibles

y alta velocidad, estrategia de tierra con plano de tierra continua o tierra en estrella o plano dividido, desacople con ubicación de capacitores de desacople, y zona libre alrededor de antena. Los interesados son ingeniero de compatibilidad electromagnética preguntando si tu convertidor reductor con conmutación a quinientos kilohertz tiene plano de tierra debajo para minimizar área de lazo, por qué la línea del inductor del convertidor es larga de tres centímetros cuando debería estar junto al circuito integrado, y si hay cuentas de ferrita en las líneas de entrada de sensores para bloquear ruido inyectado desde cables largos, ingeniero de radiofrecuencia preguntando si tu antena WiFi tiene plano de tierra debajo cuando la hoja de datos de antena integrada requiere zona libre de tierra, si la línea del convertidor reductor pasa cerca de la antena porque ruido de conmutación va a degradar sensibilidad del WiFi, y si hiciste cálculo de impedancia de la línea de alimentación de radiofrecuencia como cincuenta ohms, ingeniero de integridad de señal preguntando cómo aseguras que rebote de tierra del convertidor reductor no inyecta ruido en tierra del convertidor analógico-digital de dieciséis bits con bit menos significativo de setenta y seis microvolts, si pusiste capacitores de desacople en alimentación del convertidor analógico-digital menor a cinco milímetros del pin, y si líneas de entrada analógica están trazadas lejos de líneas digitales con líneas de guarda a tierra, y manufactura preguntando si tus vías son cero punto tres milímetros cuando capacidades de tu fabricante tienen mínimo típicamente cero punto cuatro milímetros y si separación entre línea de tres punto tres volts y tierra es cero punto quince milímetros cuando para cobre de dos onzas el mínimo recomendado es cero punto dos milímetros. La votación requiere mayoría para aprobar para fabricación y la evaluación considera comprensión de consideraciones de interferencia electromagnética, integridad de señal y radiofrecuencia, justificación técnica de compromisos de diseño, y capacidad de identificar y corregir errores bajo presión.

B.5 Unidad 5: Microcontroladores e IA

B.5.1 5.1: Arquitecturas de Microcontrolador (30 min)

Un debate sobre selección de microcontrolador para clasificación de gestos en tiempo real usando acelerómetro más modelo de aprendizaje automático ejecutado directamente en el dispositivo donde los requisitos son clasificación de diez gestos diferentes, latencia menor a cien milisegundos desde sensor hasta resultado, alimentación batería con autonomía mayor a veinticuatro horas, conectividad Bluetooth de bajo consumo para enviar resultados a teléfono inteligente y costo objetivo menor a ciento cincuenta pesos. El primer equipo defiende microcontrolador de doble núcleo a doscientos cuarenta megahertz con WiFi más Bluetooth de bajo consumo integrado sin necesidad de módulo externo, doble núcleo donde un núcleo para aprendizaje automático y otro para pila de Bluetooth, ecosistema maduro con muchos ejemplos y costo bajo de sesenta pesos pero acepta consumo alto de ciento sesenta miliamperes activo y diez microamperes en suspensión profunda dando batería de quinientos miliamperes-hora durando solo tres horas en uso continuo, sin unidad de punto flotante de precisión simple haciendo inferencia de red neuronal lenta por emulación de punto flotante, y fluctuación alta porque pila de WiFi y Bluetooth interfiere con sincronización determinista. El segundo equipo defiende microcontrolador de arquitectura diferente a ochenta megahertz con unidad de punto flotante argumentando unidad de punto flotante de precisión simple acelera inferencia de red neuronal tres a cinco veces, ultra-bajo consumo de seis miliamperes a ochenta megahertz y menor a un microampere en suspensión profunda dando batería durando más de veinticuatro horas, determinismo completo sin sistema operativo multitarea complejo dando latencia predecible, y instrucciones de procesamiento digital de señales acelerando preprocesamiento de señales de acelerómetro pero acepta sin conectividad integrada necesitando módulo Bluetooth externo más cincuenta pesos, reloj menor de ochenta megahertz versus doscientos cuarenta megahertz cuestionando si es suficiente para inferencia en menor a cien milisegundos, y ecosistema menos amigable para principiantes. El moderador introduce datos adicionales donde el modelo de red neuronal tiene cincuenta mil parámetros ocupando doscientos kilobytes y requiere diez millones de operaciones por inferencia. El cálculo de viabilidad para primera opción sin unidad de punto flotante da emulación de punto flotante de cincuenta ciclos por operación totalizando diez millones por cincuenta igual a quinientos millones de ciclos que a doscientos cuarenta megahertz da dos punto cero ocho segundos superando latencia de cien milisegundos mientras con cuantización de ocho bits de diez ciclos por operación da cero punto cuarenta y dos segundos aún muy lento. El cálculo para segunda opción con unidad de punto flotante da uno a cinco ciclos por operación con promedio de tres ciclos totalizando diez millones

por tres igual a treinta millones de ciclos que a ochenta megahertz da cero punto trescientos setenta y cinco segundos superando cien milisegundos mientras con optimización cuantizada de ocho bits de un ciclo por operación da cero punto ciento veinticinco segundos dando ciento veinticinco milisegundos cercano pero superando límite. El segundo giro pregunta sobre tercera opción de alto rendimiento a cuatrocientos ochenta megahertz con unidad de punto flotante más memoria caché donde treinta millones de ciclos sobre cuatrocientos ochenta megahertz da sesenta y dos punto cinco milisegundos cumpliendo latencia pero costo aproximadamente doscientos cincuenta pesos superando presupuesto y consumo de ciento ochenta miliamperes similar a primera opción sin ventaja en batería. La conclusión es que ninguna opción cumple todos los requisitos perfectamente y necesitan argumentar qué requisito es negociable entre relajar latencia a doscientos milisegundos para segunda opción viable, reducir complejidad del modelo con menos parámetros para ambos viables, aumentar presupuesto para tercera opción viable, o sacrificar autonomía de batería para primera opción con cuantización más batería grande evaluando capacidad de argumentar compromisos cuantitativos y proponer soluciones creativas.

B.5.2 5.2: Fluctuación de Sistema Operativo en Tiempo Real (30 min)

Un análisis forense de sistema de control de motor paso a paso usando sistema operativo en tiempo real donde el motor pierde pasos esporádicamente una a dos veces por minuto causando pérdida de precisión en posicionamiento acumulativo. El sistema usa microcontrolador de doble núcleo a doscientos cuarenta megahertz, sistema operativo en tiempo real con frecuencia de tic de cien hertz dando período de diez milisegundos, tarea de control de motor con prioridad alta y período de un milisegundo generando pulsos de paso, tarea de comunicación serial con prioridad media procesando comandos entrantes, y tarea de registro con prioridad baja escribiendo a tarjeta de memoria cada un segundo. La medición con osciloscopio muestra período de pulsos de paso variando entre cero punto noventa y ocho milisegundos y uno punto quince milisegundos con nominal de uno punto cero cero milisegundos dando fluctuación medida de más menos ciento cincuenta microsegundos. El motor especifica pulso mínimo de dos microsegundos y frecuencia máxima de dos kilohertz con período mínimo de quinientos microsegundos. El registro de sistema muestra coincidencia temporal entre pérdida de pasos y escritura a tarjeta de memoria y el análisis de código revela que tarea de registro deshabilita interrupciones durante escritura a bus serial de tarjeta de memoria por doscientos microsegundos. La primera causa raíz es prioridad de interrupción del temporizador de motor no configurada correctamente porque el sistema operativo en tiempo real usa interrupciones de nivel uno a cinco, tareas de sistema operativo corren en

contexto de interrupción de nivel uno más bajo, temporizador de motor configurado como tarea de sistema operativo con prioridad alta pero no como interrupción de hardware dedicada, y cuando tarea de registro deshabilita interrupciones nivel uno a tres el temporizador de motor se bloquea dando solución de configurar temporizador de motor como interrupción de nivel cinco más alta que sistema operativo. La segunda causa raíz es contención de procesador por pila de WiFi donde núcleo cero corre pila de WiFi que aunque WiFi no esté en uso activo la pila hace mantenimiento ocasionalmente bloqueando interrupciones por cien a doscientos microsegundos dando solución de fijar tarea de motor a núcleo uno dedicado con pila de WiFi en núcleo cero o alternativamente deshabilitar WiFi completamente si no se usa. La tercera causa raíz es frecuencia de tic de sistema operativo demasiado baja donde frecuencia de tic de cien hertz da período de diez milisegundos, tarea de motor tiene período nominal de un milisegundo, planificador solo puede despertar tarea en múltiplos de tic como cero milisegundos diez milisegundos veinte milisegundos, para lograr un milisegundo la tarea debe despertar cada tic y usar retardo interno pero esto introduce fluctuación de hasta un tic de diez milisegundos en peor caso dando solución de aumentar frecuencia de tic a mil hertz de un milisegundo o usar temporizador de hardware dedicado. Las soluciones propuestas son usar temporizador de hardware configurando temporizador de hardware en modo de recarga automática a un kilohertz con rutina de servicio de interrupción generando pulsos de paso directamente sin pasar por planificador de sistema operativo, aumentar prioridad de interrupción a nivel cinco si se usa temporizador, fijar a núcleo dedicado donde núcleo uno solo para motor y núcleo cero para resto, eliminar deshabilitación de interrupciones en registro usando acceso directo a memoria para escritura a bus serial de tarjeta sin bloquear procesador, y aumentar frecuencia de tic de cien hertz a mil hertz aunque aumenta sobrecarga de sistema operativo aproximadamente cinco por ciento produciendo resultado esperado con temporizador de hardware dedicado más acceso directo a memoria para tarjeta de memoria de fluctuación reducida a menor a cinco microsegundos determinado por latencia de interrupción de hardware con motor funcionando sin pérdida de pasos.

B.5.3 5.3: Tubería de Procesamiento de Sensores (30 min)

Un diseño rápido de sistema embebido de detección de anomalías en máquina industrial usando vibración con acelerómetro que debe clasificar tres estados como normal, desgaste de rodamiento y desbalanceo donde las especificaciones son acelerómetro de bus serial de más menos dieciséis gravedades tres ejes hasta tres punto dos kilohertz de ancho de banda, microcontrolador a elegir con justificación, modelo pre-entrenado de treinta kilobytes

con inferencia requiriendo aproximadamente cinco millones de operaciones, latencia de clasificación cada un segundo no siendo tiempo real crítico, alimentación de pared sin batería, y presupuesto de trescientos pesos. Los entregables en veinticinco minutos son diagrama de bloques desde sensor pasando por acondicionamiento, convertidor analógico-digital, microcontrolador, inferencia hasta salida, frecuencia de muestreo justificada según análisis espectral de vibración, preprocesamiento incluyendo si se necesita filtrado dado que acelerómetro ya tiene filtro interno, si se usa ventaneo, si se usa transformada rápida de Fourier y si se usa normalización, selección de microcontrolador comparando dos opciones y eligiendo una, estimación de carga de procesador incluyendo costo de adquisición por bus serial, costo de transformada de Fourier si se usa, costo de inferencia y margen de seguridad, y lista de materiales con costos. Las consideraciones técnicas a discutir incluyen frecuencia de muestreo donde vibración de fallo de rodamiento es típicamente uno a diez kilohertz, por teorema de muestreo necesitas mayor a veinte kilohertz pero ancho de banda máximo del acelerómetro es tres punto dos kilohertz dando decisión de muestrear a seis punto cuatro kilohertz siendo dos veces ancho de banda del sensor o aceptar solapamiento espectral, tamaño de ventana para transformada de Fourier donde transformada de doscientos cincuenta y seis puntos a seis punto cuatro kilohertz da cuarenta milisegundos de datos con resolución en frecuencia de seis mil cuatrocientos hertz sobre doscientos cincuenta y seis igual a veinticinco hertz cuestionando si es suficiente para distinguir fallo de rodamiento con picos finos en espectro y compromiso donde más puntos da mejor resolución pero más cómputo, y opciones de microcontrolador comparando primera opción a doscientos cuarenta megahertz sin unidad de punto flotante dando transformada de Fourier lenta versus segunda opción a ciento ochenta megahertz con unidad de punto flotante dando transformada de Fourier rápida con biblioteca optimizada donde punto de referencia de transformada de Fourier de doscientos cincuenta y seis puntos de punto flotante da primera opción aproximadamente ocho milisegundos y segunda opción con unidad de punto flotante más biblioteca optimizada aproximadamente cero punto ocho milisegundos. La presentación en cinco minutos recibe preguntas desafiantes como «tu transformada de Fourier usa doscientos cincuenta y seis puntos pero el modelo espera características de transformada de Fourier o forma de onda cruda», «muestreo a seis punto cuatro kilohertz continuo da seis mil cuatrocientas muestras por segundo por dos bytes igual a doce punto ocho kilobytes por segundo y tu microcontrolador tiene memoria de acceso aleatorio suficiente para buffer», «dijiste cinco millones de operaciones de inferencia pero incluiste costo de transformada de Fourier más normalización», y «cómo detectas que el acelerómetro falló por cable suelto o sensor muerto con autoprueba» evaluando completitud técnica más justificación cuantitativa más viabilidad de implementación.

B.5.4 5.4: Sistema de Cómputo en el Borde (35 min)

Un equipo presenta en ocho minutos arquitectura completa de cámara inteligente de conteo de personas para comercio minorista con aforo en tiempo real donde el sistema usa cámara de seiscientos cuarenta por cuatrocientos ochenta píxeles a treinta cuadros por segundo, modelo de detección de objetos cuantizado para detección de personas, salida de contador de personas entrando y saliendo, conectividad WiFi para enviar estadísticas a panel de control, y restricción de menor a dos watts de consumo total por alimentación sobre Ethernet. El diseño debe especificar arquitectura de hardware eligiendo entre microcontrolador o sistema en chip o microcontrolador más acelerador externo de inteligencia artificial e interfaz de cámara como paralelo, serial de alta velocidad o bus serial universal, tubería de procesamiento desde captura pasando por cambio de tamaño, inferencia, post-procesamiento hasta conteo, optimizaciones incluyendo si se usa cuantización de ocho bits, si se reduce resolución, y si se hace salto de cuadros no procesando treinta cuadros por segundo completos, estimación de rendimiento con latencia de inferencia, cuadros por segundo alcanzable y consumo de potencia, y lista de materiales con costo. Los interesados son ingeniero de aprendizaje automático preguntando si modelo cuantizado con exactitud de sesenta y cinco por ciento es suficiente para conteo fiable, cómo manejas oclusión de personas solapadas y con qué conjunto de datos fue entrenado el modelo, y si usas salto de cuadros procesando uno de cada tres cuadros puedes perder personas que pasan rápido, ingeniero de sistemas embebidos preguntando si microcontrolador con solo quinientos veinte kilobytes de memoria de acceso aleatorio puede contener modelo que necesita aproximadamente trescientos kilobytes para activaciones, si dijiste quince cuadros por segundo de inferencia incluiste tiempo de captura de cámara y post-procesamiento, y si usas acceso directo a memoria para transferencia de imagen desde cámara porque si no procesador está bloqueado copiando píxeles, ingeniero de energía preguntando cómo microcontrolador a doscientos cuarenta megahertz más cámara activa consume aproximadamente quinientos miliwatts y WiFi en transmisión consume otros quinientos miliwatts pico suma menor a dos watts total pero consideraste calentamiento en caja hermética, y gerente de producto preguntando por qué tu solución a ochocientos pesos cuando competidor ofrece sensor de profundidad por tiempo de vuelo a seiscientos pesos con noventa y cinco por ciento de exactitud, si la cámara se satura con alguien pasando la mano frente a ella repetidamente el contador se vuelve loco, y si guardas imágenes de personas o solo metadatos de conteo por privacidad. La votación requiere mayoría para luz verde para prototipo y la evaluación considera realismo técnico con cálculos de rendimiento, memoria y potencia, anticipación de casos extremos y problemas de despliegue, y compromisos bien justificados.

Notas de Implementación

La preparación de actividades requiere material impreso para actividades tipo galería de conceptos donde las estaciones se imprimen en hojas separadas y se colocan en distintas mesas del aula. Los casos de análisis forense incluyen evidencia simulada como formas de onda generadas con simuladores de circuitos o bibliotecas de graficación, código con fragmentos reales con errores introducidos intencionalmente, y registros simulados en archivos de texto. Para prevenir uso de inteligencia artificial modifica valores numéricos y nombres de componentes en cada semestre cambiando voltajes como doce volts a quince volts, nombres de componentes y parámetros de aplicación como motor de dos amperes a motor de tres punto cinco amperes.

Las actividades están diseñadas para veinte a treinta y cinco minutos. Si dispones de sesiones más largas puedes combinar dos actividades complementarias como galería más debate, extender fase de plenaria con discusión más profunda, o agregar ronda de corrección y re-presentación en revisiones de diseño. Para clases remotas usa salas separadas que rotan cada tres minutos para galería de conceptos, un equipo presenta en sala principal mientras otro en sala separada prepara réplica para debates, pizarra colaborativa digital para que equipo dibuje diseño en vivo en revisiones de diseño, manteniendo tiempo límite estricto más presentación oral obligatoria para efectividad contra uso de inteligencia artificial.

Las actividades progresan en complejidad de Unidad 1 a 5. La Unidad 1 presenta conceptos fundamentales con problemas de solución única y poca ambigüedad. La Unidad 2 introduce compromisos simples de costo versus rendimiento con dos a tres opciones viables. La Unidad 3 presenta compromisos complejos con múltiples variables requiriendo análisis cuantitativo. La Unidad 4 involucra problemas multidisciplinarios combinando interferencia electromagnética más integridad de señal más costo más manufactura. La Unidad 5 presenta sistemas completos con optimización bajo múltiples restricciones y diseño abierto.

Una actividad es exitosa si más de ochenta por ciento de estudiantes participan activamente sin que solo dos o tres dominen, discusiones técnicas emergen espontáneamente sin que profesor tenga que jalar, estudiantes hacen preguntas entre ellos no solo a profesor, y al final pueden articular por qué una opción es mejor que otra en lugar de solo porque sí. El diagnóstico común incluye que si estudiantes terminan muy rápido en diez minutos de actividad de veinticinco minutos agrega giro a mitad de actividad que invalida solución inicial forzando reanálisis, si estudiantes se atorán y no avanzan prepara pistas progresivas que revelas cada cinco minutos donde primera pista es sutil y última pista casi da la respuesta, si un equipo domina debate y otros no participan implementa regla de cada integrante debe hablar al menos una vez o rotación obligatoria de vocero cada cinco minutos, y si es-

tudiantes usan teléfono celular discretamente aplica tiempo límite agresivo de quince minutos en vez de veinticinco más componente oral improvisado donde explica tu decisión en treinta segundos sin leer notas.