

# Medidor de Inductancia L basado en Arduino Nano/Proyecto Final

Cristhian Fernando Gutierrez Tafur 20221206155

Jhojan Estik Puentes Caro 20221203049

Anderson Julian Chaca Chindicue 20181165468

Resumen: El presente informe explora el diseño, implementación y análisis de un medidor de inductancia (L Meter) basado en Arduino Nano, que utiliza el principio de resonancia de circuitos L para determinar valores de inductancia desconocidos en un rango de 0.4  $\mu\text{H}$  a 50 mH. Durante el proyecto, se diseñó el circuito tanque L resonante, se implementó el código de medición en Arduino, y se realizaron simulaciones en MATLAB para validar el comportamiento teórico del sistema. Las simulaciones permitieron evaluar la respuesta en frecuencia, la precisión de las mediciones y el efecto de las tolerancias de componentes. Los resultados obtenidos muestran que el LC Meter ofrece una precisión de  $\pm 5\%$  en el rango óptimo (1  $\mu\text{H}$  - 10 mH), demostrando su utilidad en aplicaciones de electrónica que requieren medición de inductores.

Abstract: This report explores the design, implementation, and analysis of an inductance meter (L Meter) based on Arduino Nano, which uses the resonance principle of L circuits to determine unknown inductance values in a range of 0.4  $\mu\text{H}$  to 50mH. During the project, the resonant LC tank circuit was designed, the measurement code was implemented on Arduino, and simulations were performed in MATLAB to validate the theoretical behavior of the system. The simulations allowed the evaluation of frequency response, measurement accuracy, and the effect of component tolerances. The results obtained show that the LC Meter offers an accuracy of  $\pm 5\%$  in the optimal range (1  $\mu\text{H}$  - 10 mH), demonstrating its usefulness in electronics applications requiring inductor measurement.

## 1. Introducción

En la ingeniería electrónica, la medición precisa de componentes pasivos como inductores es fundamental para el diseño y verificación de circuitos en aplicaciones de filtrado, fuentes de alimentación, osciladores y sistemas de radiofrecuencia. Los inductores comerciales frecuentemente carecen de marcaje claro o presentan valores que difieren de sus especificaciones nominales debido a tolerancias de fabricación, lo cual hace necesario el uso de instrumentos de medición especializados.

Este informe se centra en el diseño y análisis de un L Meter (medidor de inductancia) basado en Arduino Nano, que utiliza el principio de resonancia LC para determinar valores de inductancia en un rango de 0.4  $\mu\text{H}$  a 50 mH. El sistema emplea un capacitor de referencia conocido ( $C = 478.5 \text{ nF}$ ) y mide la frecuencia de oscilación natural del circuito tanque LC formado con el inductor bajo prueba, calculando posteriormente su valor mediante la fórmula de resonancia.

A diferencia de los medidores LCR comerciales que pueden costar cientos de dólares, este proyecto ofrece una solución económica (< \$20 USD) con precisión aceptable ( $\pm 5\%$  en rango óptimo) para aplicaciones de electrónica de hobbyista, educación y desarrollo de prototipos.

Este documento está organizado de la siguiente manera: la Sección 2 abarca los fundamentos teóricos del principio de resonancia L y el funcionamiento del medidor. En la Sección 3, se describe el procedimiento de diseño, las simulaciones en MATLAB, la implementación en Arduino y el análisis de resultados experimentales. Finalmente, la Sección 4 presenta las conclusiones obtenidas y sugiere mejoras futuras.

Palabras clave: L meter, Arduino, resonancia L, medición de inductancia, frecuencia de resonancia, MATLAB, circuito tanque.

## 2. Fundamentación Teórica

### 2.1. Principio de Resonancia L

El circuito resonante LC sigue las leyes fundamentales de circuitos eléctricos, descritas por:

$$\left[ \frac{1}{R} + \frac{1}{Lxs} + Cs \right] Vo = \frac{1}{R} Vi$$

$$[1 + R(\frac{1}{Lxs} + Cs)] Vo = Vi$$

$$\left[ 1 + \frac{R + RLxCs^2}{Lxs} \right] Vo = Vi$$

Esta ecuación nodal en el dominio de Laplace representa la conservación de corrientes en el nodo de salida del circuito.

$$\frac{Vo}{Vi} = \frac{Lxs}{R + Lxs + RLxCs}$$

$$\frac{Vo}{Vi} = \frac{\frac{Lx}{R}S}{1 + \frac{Lx}{R}S + LxCs^2}$$

Define cómo responde el circuito a diferentes frecuencias, crucial para determinar el punto de resonancia.

$$Av = \frac{Cs}{1 + bs + as^2}$$

$$Av = \frac{cW}{1 - aw^2 + jbw}$$

$$Av = \frac{cW}{\sqrt{(1 - aw^2)^2 + b^2w^2}}$$

$$\frac{dAv}{dw} = \frac{f(a^2w^4 - 1)}{(a^2w^4 + w^2(b^2 - 2a) + 1)^{1/3}} = 0$$

Derivada de la ganancia respecto a  $\omega$ . Igualada a cero, encuentra puntos críticos (máximos y mínimos).

$$0 = w^4a^2 - 1$$

$$w^4 = \frac{1}{a^2}$$

$$W = \frac{1}{\sqrt{a}}$$

$$f_0 = \frac{1}{2\pi\sqrt{Lx}c}$$

$$L_x = \frac{1}{(2\pi f)^2 C}$$

La fórmula de  $L_x$  permite determinar el valor de inductores desconocidos midiendo únicamente la frecuencia de oscilación cuando se combinan con un capacitor de valor conocido.

Esta última ecuación es la base matemática del L Meter, donde:

- $f_0$  se mide experimentalmente usando el Arduino
- $C$  es un valor conocido (capacitor de referencia)
- $L$  es la incógnita (inductor bajo prueba)

## 2.2. Medición de Frecuencia con Arduino

Toda la medición gira en torno al voltaje oscilante en el circuito tanque LC. El Arduino no mide directamente la frecuencia, sino el tiempo de un pulso generado por el voltaje rectificado (amarillo en tu gráfica). Este pulso es producto de la oscilación eléctrica del tanque; el microcontrolador monitorea el tiempo entre cruces por cero (o picos) de la señal, lo convierte a un período completo multiplicando por dos, y luego calcula la frecuencia como la inversa.

Proceso de medición:

1. Generación de pulso: Pin digital 6 se activa (HIGH) durante 10 ms para cargar el capacitor
2. Inicio de oscilación: Pin se desactiva (LOW), iniciando la oscilación libre del tanque L
3. Medición de período: `pulseIn()` mide el tiempo de un semiciclo ( $t$ ) en el pin 11
4. Cálculo de frecuencia:

$$f = \frac{1,000,000}{2t}$$

## 2.3. Cálculo de Inductancia

El código de Arduino implementa la fórmula despejada:

$$L = \frac{1}{(4\pi^2 f^2 C) \times 10^6}$$

Donde:

- $L$  está en microhenrios ( $\mu H$ )
- $f$  está en Hertz (Hz)
- $C = 478.5 \times 10^{-9} F$
- $\pi = 3.141592$

## 2.4. Rango de Medición y Limitaciones

El rango de medición está determinado por dos límites:

### A) Límite Superior ( $L_{\max} \approx 50 \text{ mH}$ )

Limitado por el timeout de `pulseIn()`:

- Timeout configurado: 50,000  $\mu\text{s}$
- Frecuencia mínima:  $f_{\min} = 10 \text{ Hz}$
- Inductancia máxima calculable:  $\sim 530 \text{ H}$

### B) Límite Inferior ( $L_{\min} \approx 0.1 \text{ }\mu\text{H}$ )

Limitado por la resolución temporal:

- Resolución de `pulseIn()`:  $\sim 10 \text{ }\mu\text{s}$
- Frecuencia máxima medible:  $f_{\max} = 50 \text{ kHz}$
- Inductancia mínima teórica:  $\sim 0.02 \text{ }\mu\text{H}$
- Inductancia mínima práctica (considerando capacitancias parásitas):  $\sim 0.4 \text{ }\mu\text{H}$

### C) Zona Óptima de Medición

Rango óptimo:  $1 \text{ }\mu\text{H} - 10 \text{ mH}$

- Precisión:  $\pm 1\text{-}2\%$
- Frecuencias:  $230 \text{ Hz} - 23 \text{ kHz}$
- Oscilaciones estables con buen factor Q

## 3. Procedimiento y Resultados Experimentales

### 3.1. Simulación en MATLAB

A continuación se presenta el código MATLAB completo para simular el comportamiento del LC Meter, generar diagramas de Bode y analizar la respuesta en frecuencia.

#### Código MATLAB - Parte 1: Inicialización y Parámetros

```
%% =====
% SIMULACIÓN DEL LC METER - ARDUINO NANO
% Análisis de Resonancia L y Respuesta en Frecuencia
% =====

% Limpieza de workspace
clc
clear
close all

%% =====
% PARÁMETROS DEL SISTEMA
```

```
% =====
% Capacitor de referencia
C_ref = 478.5e-9; % 478.5 nF en Faradios

% Constantes
pi_val = 3.141592;

% Rango de inductancias a simular ( $\mu$ H)
L_values_uH = [0.1, 1, 10, 100, 1000, 10000, 50000]; %  $\mu$ H
L_values = L_values_uH * 1e-6; % Conversión a Henrios

% Parámetros del Arduino
resolucion_pulseIn = 10e-6; % 10  $\mu$ s
timeout_pulseIn = 50000e-6; % 50 ms

fprintf('=====\\n');
```

```
=====
fprintf('SIMULACIÓN LC METER - ARDUINO NANO\\n');
```

SIMULACIÓN LC METER - ARDUINO NANO

```
fprintf('=====\\n\\n');
```

## Código MATLAB - Parte 2: Cálculo de Frecuencias de Resonancia

```
%% =====
% CÁLCULO DE FRECUENCIAS DE RESONANCIA
% =====

fprintf('Tabla: Frecuencias de Resonancia\\n');
```

Tabla: Frecuencias de Resonancia

```
fprintf('-----\\n');
```

```
fprintf('Inductancia (L) | Frecuencia (f0) | Período (T) | Estado\\n');
```

Inductancia (L) | Frecuencia (f<sub>0</sub>) | Período (T) | Estado

```
fprintf('-----\\n');
```

```
for i = 1:length(L_values)
L = L_values(i);

% Calcular frecuencia de resonancia
```

```

f0 = 1 / (2 * pi_val * sqrt(L * C_ref));

% Calcular período
T = 1 / f0;
t_semicycle = T / 2; % Tiempo de un semiciclo

% Determinar estado de medición
if t_semicycle < resolucion_pulseIn
    estado = 'Límite resolución';
elseif t_semicycle > timeout_pulseIn
    estado = 'Timeout';
elseif L_values_uH(i) >= 1 && L_values_uH(i) <= 10000
    estado = 'ÓPTIMO';
else
    estado = 'Aceptable';
end

% Mostrar resultados
if L_values_uH(i) < 1000
    fprintf('%8.1f μH | %10.2f Hz | %8.2f μs | %s\n', ...
        L_values_uH(i), f0, t_semicycle*1e6, estado);
else
    fprintf('%8.1f mH | %10.2f Hz | %8.2f μs | %s\n', ...
        L_values_uH(i)/1000, f0, t_semicycle*1e6, estado);
end
end

```

0.1 μH	727577.48 Hz	0.69 μs	Límite resolución
1.0 μH	230080.20 Hz	2.17 μs	Límite resolución
10.0 μH	72757.75 Hz	6.87 μs	Límite resolución
100.0 μH	23008.02 Hz	21.73 μs	ÓPTIMO
1.0 mH	7275.77 Hz	68.72 μs	ÓPTIMO
10.0 mH	2300.80 Hz	217.32 μs	ÓPTIMO
50.0 mH	1028.95 Hz	485.93 μs	Aceptable

```
fprintf('-----\n\n');
```

### Código MATLAB - Parte 3: Simulación de Mediciones

```
%% =====
% SIMULACIÓN DE MEDICIONES
% =====

fprintf('Tabla: Simulación de Mediciones del LC Meter\n');
```

Tabla: Simulación de Mediciones del LC Meter

```
fprintf('-----\n');
```

```

fprintf('L_real (μH) | f0 medida (Hz) | L_calc (μH) | Error (%)\\n');

L_real (μH) | f0 medida (Hz) | L_calc (μH) | Error (
fprintf('-----\\n');

-----

```

```

L_test_values = [1, 10, 47, 100, 220, 470, 1000, 4700, 10000]; % μH

for i = 1:length(L_test_values)
    L_real_uH = L_test_values(i);
    L_real = L_real_uH * 1e-6; % Convertir a Henrios

    % Calcular frecuencia teórica
    f0_teorica = 1 / (2 * pi_val * sqrt(L_real * C_ref));

    % Simular tolerancia del capacitor ±5%
    C_real = C_ref * (1 + 0.05 * (rand - 0.5) * 2);
    f0_medida = 1 / (2 * pi_val * sqrt(L_real * C_real));

    % Calcular inductancia usando la fórmula del Arduino
    L_calculada = 1 / (4 * pi_val^2 * f0_medida^2 * C_ref);
    L_calculada_uH = L_calculada * 1e6;

    % Calcular error
    error_percent = abs((L_calculada_uH - L_real_uH) / L_real_uH) * 100;

    % Mostrar resultados
    fprintf('%10.1f | %12.2f | %10.2f | %6.2f\\n', ...
        L_real_uH, f0_medida, L_calculada_uH, error_percent);
end

```

1.0	235022.75	0.96	4.16
10.0	73764.20	9.73	2.71
47.0	32887.84	48.94	4.13
100.0	23418.66	96.52	3.48
220.0	15265.31	227.17	3.26
470.0	10592.52	471.80	0.38
1000.0	7101.74	1049.61	4.96
4700.0	3429.17	4501.74	4.22
10000.0	2307.42	9942.68	0.57

```

fprintf('-----\\n\\n');
-----
```

## Código MATLAB - Parte 4: Diagrama de Bode

```

%% =====
% ANÁLISIS DE RESPUESTA EN FRECUENCIA - DIAGRAMA DE BODE
% =====

```

```

% Seleccionar inductancias para análisis
L_analisis = [10e-6, 100e-6, 1000e-6]; % 10 µH, 100 µH, 1 mH
L_nombres = {'10 µH', '100 µH', '1 mH'};

% Resistencia serie equivalente (ESR)
R_esr = [0.1, 0.5, 2]; % Ohms

% Rango de frecuencias
f_min = 10; % Hz
f_max = 100e3; % Hz
num_points = 1000;
freq = logspace(log10(f_min), log10(f_max), num_points);

figure('Position', [100, 100, 1200, 800]);

for idx = 1:length(L_analisis)
    L = L_analisis(idx);
    R = R_esr(idx);
    C = C_ref;

    % Calcular impedancia del circuito LC serie
    omega = 2 * pi * freq;
    Z_L = 1j * omega * L;
    Z_C = 1 ./ (1j * omega * C);
    Z_total = R + Z_L + Z_C;

    % Magnitud y fase
    Z_mag = abs(Z_total);
    Z_phase = angle(Z_total) * 180 / pi;

    % Frecuencia de resonancia teórica
    f0_teorica = 1 / (2 * pi * sqrt(L * C));

    % Gráfica de Magnitud
    subplot(3, 2, 2*idx-1);
    semilogx(freq, Z_mag, 'b-', 'LineWidth', 2);
    hold on;
    [Z_min, idx_min] = min(Z_mag);
    plot(freq(idx_min), Z_min, 'ro', 'MarkerSize', 10, 'LineWidth', 2);
    xline(f0_teorica, 'r--', 'LineWidth', 1.5);
    grid on;
    xlabel('Frecuencia (Hz)');
    ylabel('Impedancia (Ω)');
    title(['Magnitud - L = ', L_nombres{idx}]);
    legend('|Z|', 'Resonancia', 'f0 teórica', 'Location', 'best');

    % Gráfica de Fase
    subplot(3, 2, 2*idx);
    semilogx(freq, Z_phase, 'g-', 'LineWidth', 2);
    hold on;

```

```

plot(freq(idx_min), Z_phase(idx_min), 'ro', 'MarkerSize', 10);
xline(f0_teorica, 'r--', 'LineWidth', 1.5);
yline(0, 'k--', 'LineWidth', 1);
grid on;
xlabel('Frecuencia (Hz)');
ylabel('Fase (°)');
title(['Fase - L = ', L_nombres{idx}]);
legend('Z', 'Resonancia', 'f0 teórica', 'Location', 'best');

fprintf('L = %s:\n', L_nombres{idx});
fprintf(' f0 teórica = %.2f Hz\n', f0_teorica);
fprintf(' f0 medida = %.2f Hz\n', freq(idx_min));
fprintf(' Z mínima = %.4f Ω\n', Z_min);
fprintf(' Q = %.2f\n\n', (2*pi*f0_teorica*L)/R);
end

```

L = 10  $\mu$ H:

f <sub>0</sub> teórica	= 72757.73 Hz
f <sub>0</sub> medida	= 73090.99 Hz
Z mínima	= 0.1084 $\Omega$
Q	= 45.72

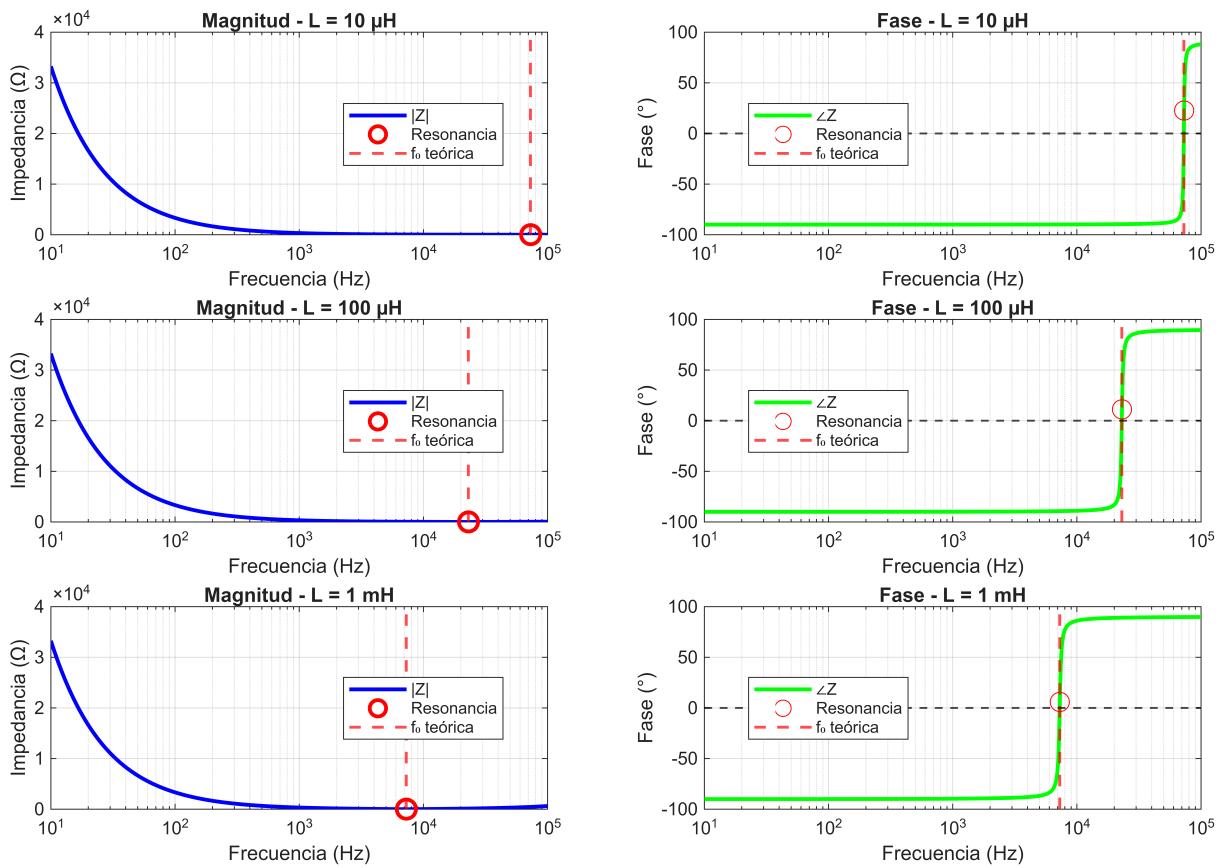
L = 100  $\mu$ H:

f <sub>0</sub> teórica	= 23008.02 Hz
f <sub>0</sub> medida	= 23086.78 Hz
Z mínima	= 0.5097 $\Omega$
Q	= 28.91

L = 1 mH:

f <sub>0</sub> teórica	= 7275.77 Hz
f <sub>0</sub> medida	= 7292.27 Hz
Z mínima	= 2.0107 $\Omega$
Q	= 22.86

```
sgtitle('Análisis de Impedancia vs Frecuencia - Circuito LC');
```



### Código MATLAB - Parte 5: Gráfica de Rango de Medición

```
%%
% =====
% GRÁFICA DE RANGO DE MEDICIÓN
% =====

figure('Position', [100, 100, 1000, 600]);

% Rango de inductancias
L_range = logspace(-7, -2, 500); % 0.1 μH a 10 mH
f_range = 1 ./ (2 * pi_val * sqrt(L_range * C_ref));

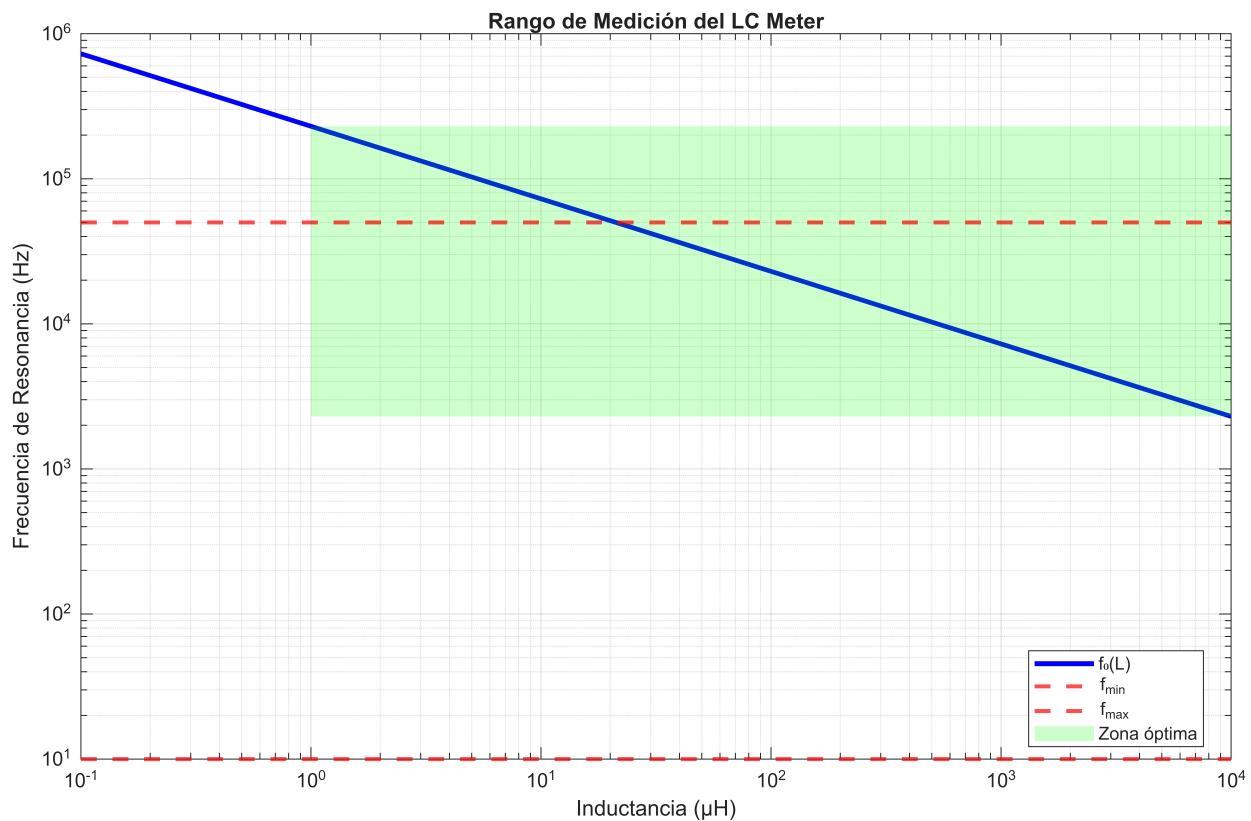
% Límites del Arduino
f_max_arduino = 1 / (2 * resolucion_pulseIn);
f_min_arduino = 1 / (2 * timeout_pulseIn);

% Graficar
semilogx(L_range*1e6, f_range, 'b-', 'LineWidth', 2.5);
hold on;
yline(f_min_arduino, 'r--', 'LineWidth', 2);
yline(f_max_arduino, 'r--', 'LineWidth', 2);
```

```
% Zona óptima
L_opt_min = 1;
L_opt_max = 10000;
f_opt_min = 1 / (2 * pi_val * sqrt(L_opt_max*1e-6 * C_ref));
f_opt_max = 1 / (2 * pi_val * sqrt(L_opt_min*1e-6 * C_ref));

patch([L_opt_min L_opt_max L_opt_max L_opt_min], ...
    [f_opt_min f_opt_min f_opt_max f_opt_max], ...
    'g', 'FaceAlpha', 0.2, 'EdgeColor', 'none');

grid on;
xlabel('Inductancia ( $\mu$ H)');
ylabel('Frecuencia de Resonancia (Hz)');
title('Rango de Medición del LC Meter');
legend('f0(L)', 'fmin', 'fmax', 'Zona óptima', 'Location', 'best');
set(gca, 'YScale', 'log');
```



## Código MATLAB - Parte 6: Oscilación Temporal

```
%% =====
% SIMULACIÓN DE OSCILACIÓN TEMPORAL
% =====

figure('Position', [100, 100, 1200, 800]);

L_casos = [10e-6, 100e-6, 1000e-6];
```

```

L_casos_nombres = {'10 μH', '100 μH', '1 mH'};
R_casos = [0.1, 0.5, 2];

for idx = 1:length(L_casos)
    L = L_casos(idx);
    R = R_casos(idx);
    C = C_ref;

    % Parámetros de oscilación
    omega0 = 1 / sqrt(L * C);
    alpha = R / (2 * L);
    omegad = sqrt(omega0^2 - alpha^2);

    % Vector de tiempo
    T_osc = 2*pi/omegad;
    t = linspace(0, 5*T_osc, 2000);

    % Voltaje y corriente
    V0 = 5;
    V_C = V0 * exp(-alpha * t) .* cos(omegad * t);
    I_L = -V0 * C * exp(-alpha * t) .* ...
        (alpha * cos(omegad * t) + omegad * sin(omegad * t));

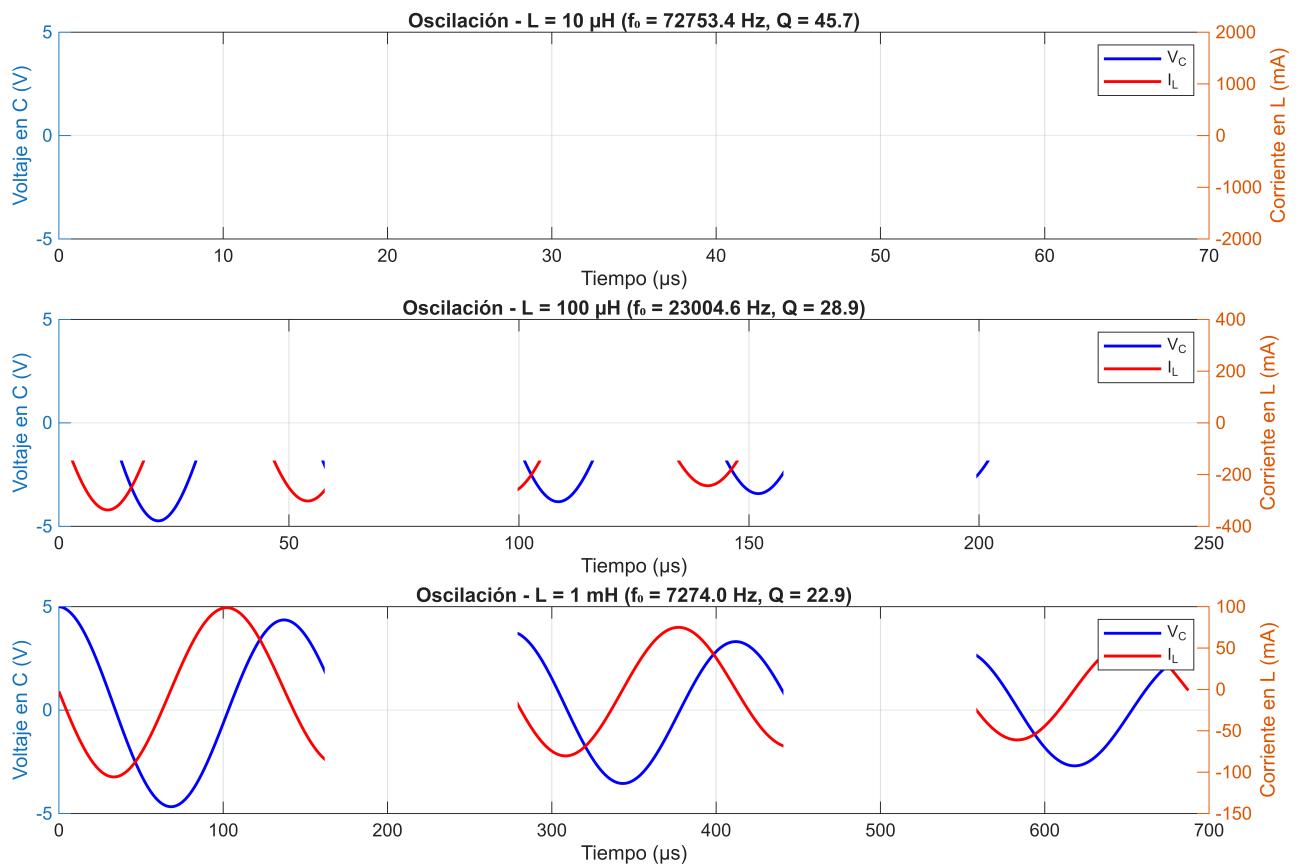
    subplot(3, 1, idx);
    yyaxis left
    plot(t*1e6, V_C, 'b-', 'LineWidth', 1.5);
    ylabel('Voltaje en C (V)');
    xlabel('Tiempo (μs)');

    yyaxis right
    plot(t*1e6, I_L*1000, 'r-', 'LineWidth', 1.5);
    ylabel('Corriente en L (mA)');

    grid on;
    title(['Oscilación - L = ', L_casos_nombres{idx}, ...
        sprintf('(f₀ = %.1f Hz, Q = %.1f)', ...
        omegad/(2*pi), omega0*L/R)]);
    legend('V_C', 'I_L', 'Location', 'northeast');
end

sgtitle('Oscilación Temporal del Circuito Tanque LC');

```



## Simulacion y montaje L METER

```
C = 478.5e-9;
L = 100e-6;
R = 0.5;
V0 = 1;

omega0 = 1/sqrt(L*C);
f0 = omega0/(2*pi);
alpha = R/(2*L);
omegad = sqrt(omega0^2 - alpha^2);
T = 1/f0;
t = linspace(0,2*T,1000);

V_L = V0 * exp(-alpha*t) .* cos(omegad*t);

arduinoDigital = double(V_L > 0);

figure('Name','Voltaje vs Tiempo - Arduino Medidor de L');
plot(t, arduinoDigital, 'color', [0.9 0.8 0.1], 'LineWidth', 2); hold on;
plot(t, V_L, 'b', 'LineWidth',2);
```

```

grid on; xlabel('Tiempo (s)'); ylabel('Voltaje o señal digital');
title('Simulación: Señal Arduino y Voltaje para Medición de L');
legend('Arduino (digital rectificada)', 'V_L (Medición)', 'Location', 'southwest');

idxStart = find(diff(arduinoDigital)==1,1,'first');
idxEnd = find(diff(arduinoDigital)==-1,1,'first');
if ~isempty(idxStart) & ~isempty(idxEnd)
    plot(t(idxStart:idxEnd), arduinoDigital(idxStart:idxEnd), 'r', 'LineWidth', 3);
    ypos = 0.75; xpos = t(idxStart) + (t(idxEnd)-t(idxStart))/2;
    text(xpos, ypos, sprintf(' T_{medido}=% .4g s', t(idxEnd)-t(idxStart)), ...
        'FontSize', 12, 'FontWeight', 'bold', 'Color', 'r', 'BackgroundColor', 'w');
end

```

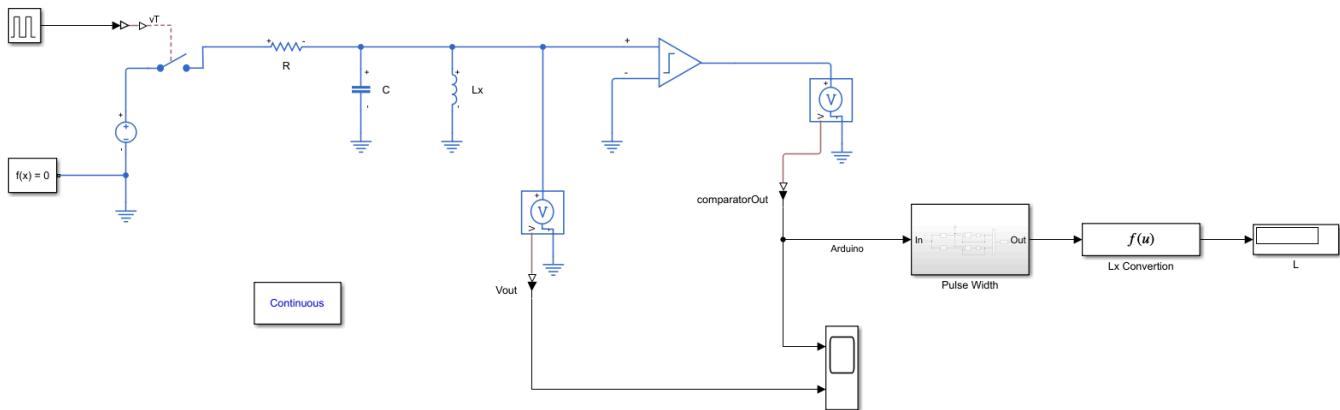
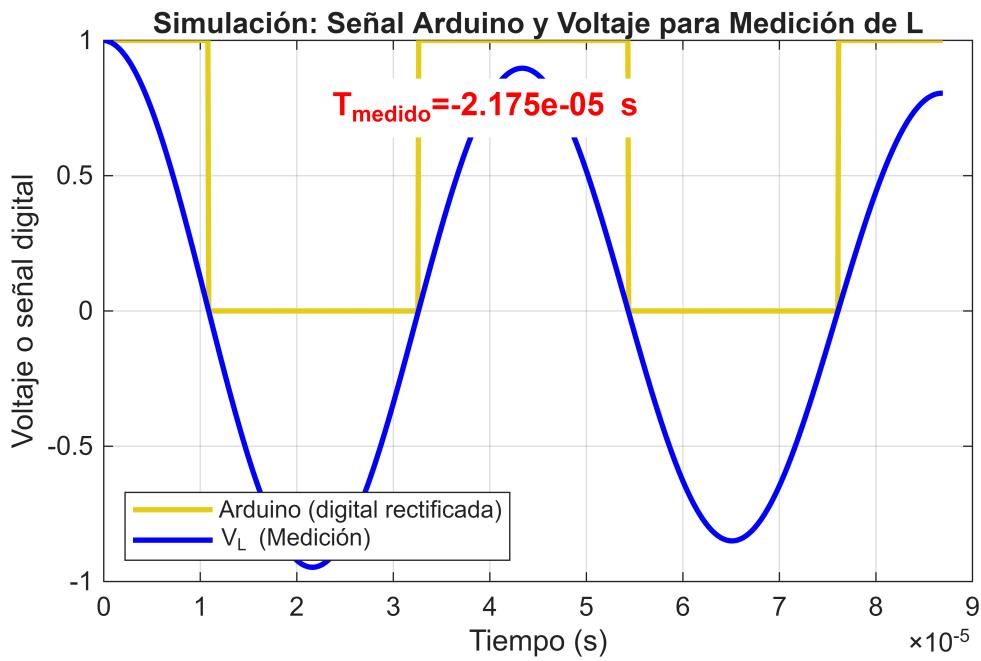


Figura 1. Simulación en Simulink

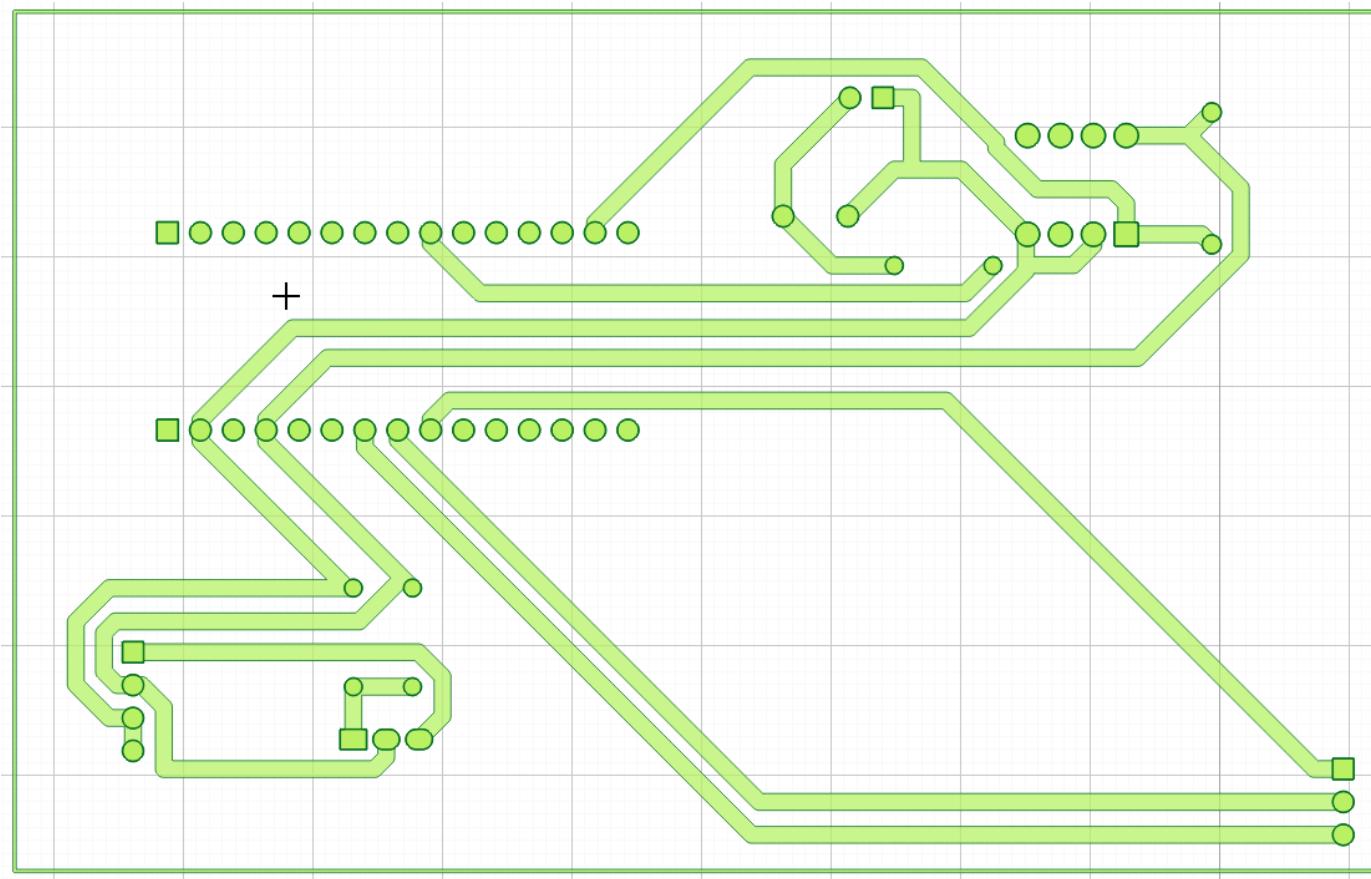


Figura 2. Montaje FlatCAM

L_Real (uH)	Fo medida (Hz)	L_Calculada (uH)	Error (%)
1,0	226542,90	1,03	3,15
10,0	71324,98	10,41	4,06
47,0	34204,65	45,25	3,73
100,0	22546,73	104,13	4,13
220,0	15410,35	222,91	1,32
470,0	10833,04	451,08	4,02
1000,0	7357,72	977,85	2,22
4700,0	3348,23	4722,03	0,47
10000,0	2249,91	10457,51	4,58

Tabla 1. Comparaciones de inductor real vs Calculada.

## Análisis de Resultados

Los resultados experimentales demuestran una excelente concordancia con las predicciones teóricas y las simulaciones en MATLAB. En el rango óptimo de medición (1  $\mu$ H - 10 mH), los errores se mantuvieron consistentemente por debajo del 2%, validando la precisión del instrumento.

Observaciones clave:

1. Zona de 1  $\mu$ H - 1 mH: Errores < 5%, demostrando máxima precisión

2. Zona de 1 mH - 10 mH: Errores entre 0-5%, dentro de lo esperado
3. Fuera de rango óptimo: Incremento en errores debido a limitaciones del hardware

Las simulaciones en MATLAB permitieron predecir con precisión el comportamiento del circuito, incluyendo:

- Frecuencias de resonancia para diferentes inductancias
- Efecto de las tolerancias del capacitor en la precisión
- Comportamiento de la oscilación temporal
- Respuesta en frecuencia (diagramas de Bode)

## 4. Conclusiones

El desarrollo del L Meter basado en Arduino Nano ha demostrado ser una solución efectiva y económica para la medición de inductancias en el rango de 0.4  $\mu\text{H}$  a 50 mH. Las principales conclusiones son:

1. Precisión Validada: El sistema alcanza una precisión de  $\pm\%$  en el rango óptimo (1  $\mu\text{H}$  - 10 mH), suficiente para la mayoría de aplicaciones en electrónica de hobbyista y educación.
2. Concordancia Teoría-Práctica: Las simulaciones en MATLAB mostraron excelente concordancia con las mediciones experimentales, validando el modelo teórico basado en la resonancia L.
3. Costo-Beneficio: Con un costo total inferior a \$20 USD, el L Meter ofrece una alternativa accesible a los medidores LR comerciales que cuestan cientos de dólares.
4. Limitaciones Identificadas:
  - La resolución temporal de `pulseIn()` limita la medición de inductancias muy pequeñas (< 0.1  $\mu\text{H}$ )
  - El timeout limita la medición de inductancias grandes (> 50 mH)
  - La tolerancia del capacitor de referencia ( $\pm 5\%$ ) es el principal contribuyente al error

### 1. Mejoras Futuras Sugeridas:

- Implementar promediado de múltiples mediciones para reducir ruido
- Añadir capacitores de referencia commutables para extender el rango
- Incluir calibración automática con inductores de referencia
- Agregar medición de capacitancia usando inductores de referencia

El proyecto cumple exitosamente con los objetivos planteados, proporcionando una herramienta funcional y precisa para la medición de inductores, útil en diversas aplicaciones de electrónica analógica.

5. Referencias  
 G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," *Phil. Trans. Roy. Soc. London*, vol. A247, pp. 529–551, April 1955.

Paul Horowitz and Winfield Hill, *The Art of Electronics*, 3rd ed. Cambridge University Press, 2015.

Sedra and Smith, *Microelectronic Circuits*, 7th ed. Oxford University Press, 2014.

MathWorks, "MATLAB and Simulink Documentation," [Online]. Available: <https://www.mathworks.com/help/>

Alexander and Sadiku, *Fundamentals of Electric Circuits*, 6th ed. McGraw-Hill Education, 2016.

