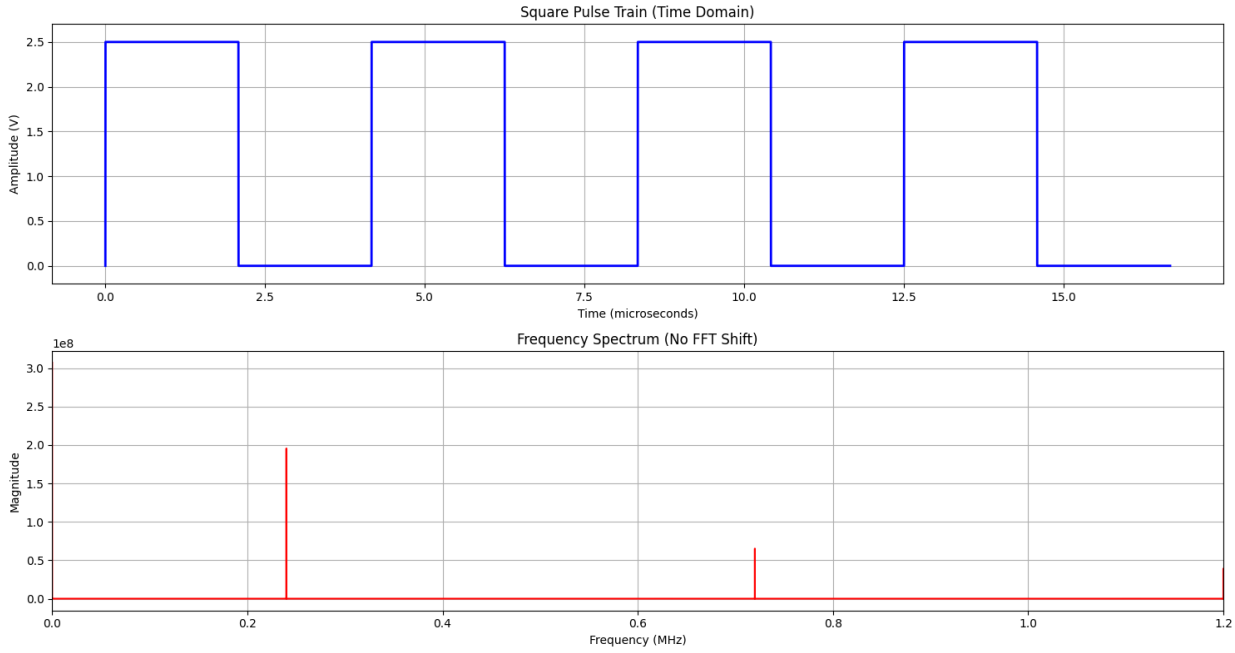
Name: Justin Ngo
PSU ID: jvn5439
Professor: Mark Mahon
Class: CompEn 462
Date: 02.17.2025

Problem 1

```
# Name: Justin Ngo
# PSU Email: jvn5439
# PSUID: 933902868
# Class: CMPEN 462
# Assignment: Homework 2
# Date: 02.17.2025

import numpy as np
import matplotlib.pyplot as plt

'''
References:
https://numpy.org/doc/stable/reference/routines.fft.html
https://www.geeksforgeeks.org/plotting-a-square-wave-using-matplotlib-numpy-and-
    scipy/
https://stackoverflow.com/questions/66000468/plot-square-wave-in-python
https://psu.instructure.com/courses/2378968/files/174067072?module_item_id
    =44321606 (Sines_FFT_example.m)
https://psu.instructure.com/courses/2378968/files/174066175?module_item_id
    =44321556 (pulse_trains.m)
https://psu.instructure.com/courses/2378968/files/174066916?module_item_id
    =44321597 (Example_tone_FFT.m)
'''


# Signal parameters
fo = 240e3      # Fundamental frequency (240 KHz)
fs = 4096 * fo  # Sample frequenct (~983MHz)
t = 250e-3      # Duration in seconds (250 ms)
A_max = 2.5     # Max Amplitude (2.5V)
A_min = 0.0     # Min Amplitude (0V)

# Time vector
t = np.arange(0, t, 1/fs)
samples = int(fs/fo) # samples per period

# Generate square pulse train
signal = A_max * (np.sign(np.sin(2 * np.pi * fo * t)) > 0)

# Calculate FFT
n = len(signal)  # Number of points
freqs = np.fft.fftfreq(n, d=1/fs)  # Frequency vector
fft_signal = np.fft.fft(signal)  # Compute FFT
pulses = 4
t_display = t[:int(pulses * fs / fo)] # Time vector for 4 pulses
signal_display = signal[:int(pulses * fs / fo)] # Signal vector for 4 pulses

# Graphs
plt.figure(figsize=(15, 8))

# Time domain plot
plt.subplot(2, 1, 1)
plt.plot(t_display * 1e6, signal_display, 'b-', linewidth=2)
plt.grid(True)
plt.xlabel('Time (microseconds)')
plt.ylabel('Amplitude (V)')
plt.title('Square Pulse Train (Time Domain)')
```

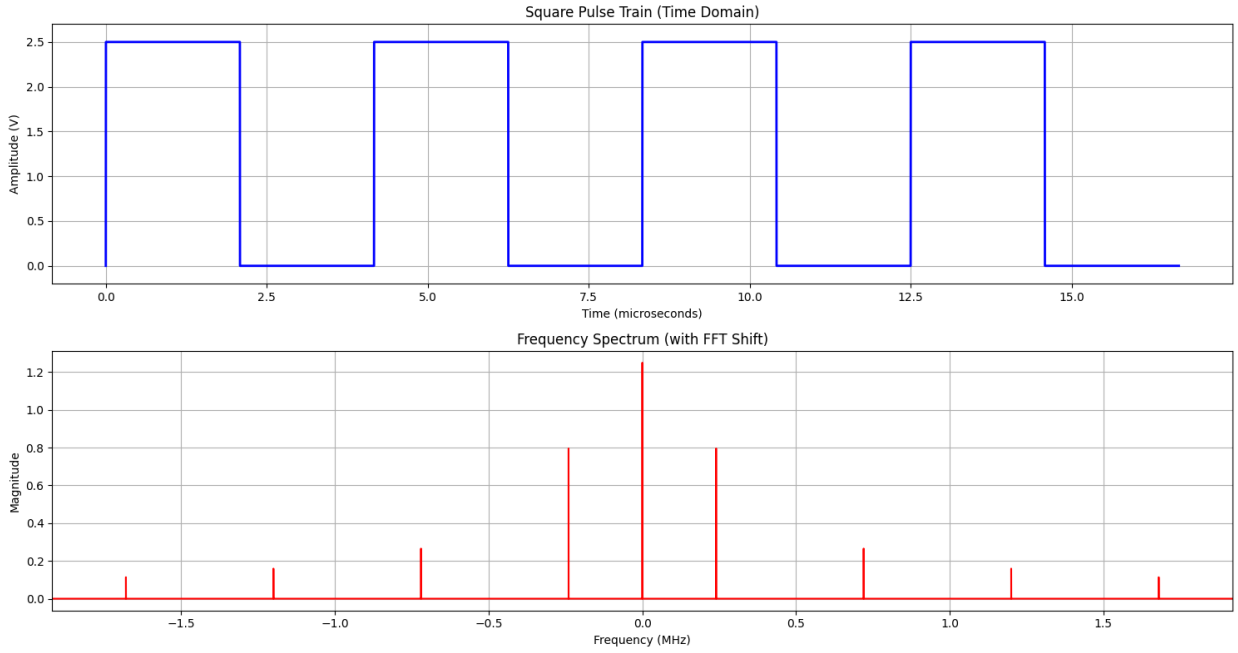
```
plt.ylim([A_min - 0.2, A_max + 0.2])

# Frequency domain plot
plt.subplot(2, 1, 2)
plt.plot(freqs[:len(freqs)//2]/1e6, np.abs(fft_signal[:len(freqs)//2]), 'r-')
plt.grid(True)
plt.xlabel('Frequency (MHz)')
plt.ylabel('Magnitude')
plt.title('Frequency Spectrum (No FFT Shift)')
plt.xlim([0, 5*fo/1e6])  # Show up to 5th harmonic

plt.tight_layout()
plt.show()
```



Problem 2

```
# Name: Justin Ngo
# PSU Email: jvn5439
# PSUID: 933902868
# Class: CMPEN 462
# Assignment: Homework 2
# Date: 02.17.2025

import numpy as np
import matplotlib.pyplot as plt

'''
References:
https://numpy.org/doc/stable/reference/routines.fft.html
https://www.geeksforgeeks.org/plotting-a-square-wave-using-matplotlib-numpy-and-
    scipy/
https://stackoverflow.com/questions/66000468/plot-square-wave-in-python
https://psu.instructure.com/courses/2378968/files/174067072?module_item_id
    =44321606 (Sines_FFT_example.m)
https://psu.instructure.com/courses/2378968/files/174066175?module_item_id
    =44321556 (pulse_trains.m)
https://psu.instructure.com/courses/2378968/files/174066916?module_item_id
    =44321597 (Example_tone_FFT.m)
'''


# Signal parameters
fo = 240e3      # Fundamental frequency (240 KHz)
fs = 4096 * fo  # Sample frequenct (~983MHz)
t = 250e-3      # Duration in seconds (250 ms)
A_max = 2.5     # Max Amplitude (2.5V)
A_min = 0.0     # Min Amplitude (0V)


# Time vector
t = np.arange(0, t, 1/fs)
samples = int(fs/fo) # samples per period

# Generate square pulse train
signal = A_max * (np.sign(np.sin(2 * np.pi * fo * t)) > 0)

# Calculate FFT
n = len(signal)  # Number of points
freqs = np.fft.fftfreq(n, d=1/fs)  # Frequency vector
fft_signal = np.fft.fft(signal)  # Compute FFT

# Apply fftshift and normalize FFT
freqs_shifted = np.fft.fftshift(freqs)
fft_signal_shifted = np.fft.fftshift(fft_signal)
fft_signal_normalized = fft_signal_shifted / n  # Remove FFT gain by dividing by n

pulses = 4
t_display = t[:int(pulses * fs / fo)] # Time vector for 4 pulses
signal_display = signal[:int(pulses * fs / fo)] # Signal vector for 4 pulses

# Graphs
plt.figure(figsize=(15, 8))
```

```
# Time domain plot
plt.subplot(2, 1, 1)
plt.plot(t_display * 1e6, signal_display, 'b-', linewidth=2)
plt.grid(True)
plt.xlabel('Time (microseconds)')
plt.ylabel('Amplitude (V)')
plt.title('Square Pulse Train (Time Domain)')
plt.ylim([A_min - 0.2, A_max + 0.2])

# Frequency domain plot
plt.subplot(2, 1, 2)
plt.plot(freqs_shifted/1e6, np.abs(fft_signal_normalized), 'r-')
plt.grid(True)
plt.xlabel('Frequency (MHz)')
plt.ylabel('Magnitude')
plt.title('Frequency Spectrum (with FFT Shift)')
# Adjust x-axis limits to show frequencies centered around 0
plt.xlim([-8*fo/1e6, 8*fo/1e6])  # Show up to +-5th harmonic

plt.tight_layout()
plt.show()
```



Problem 3

Frequencies and Amplitudes:

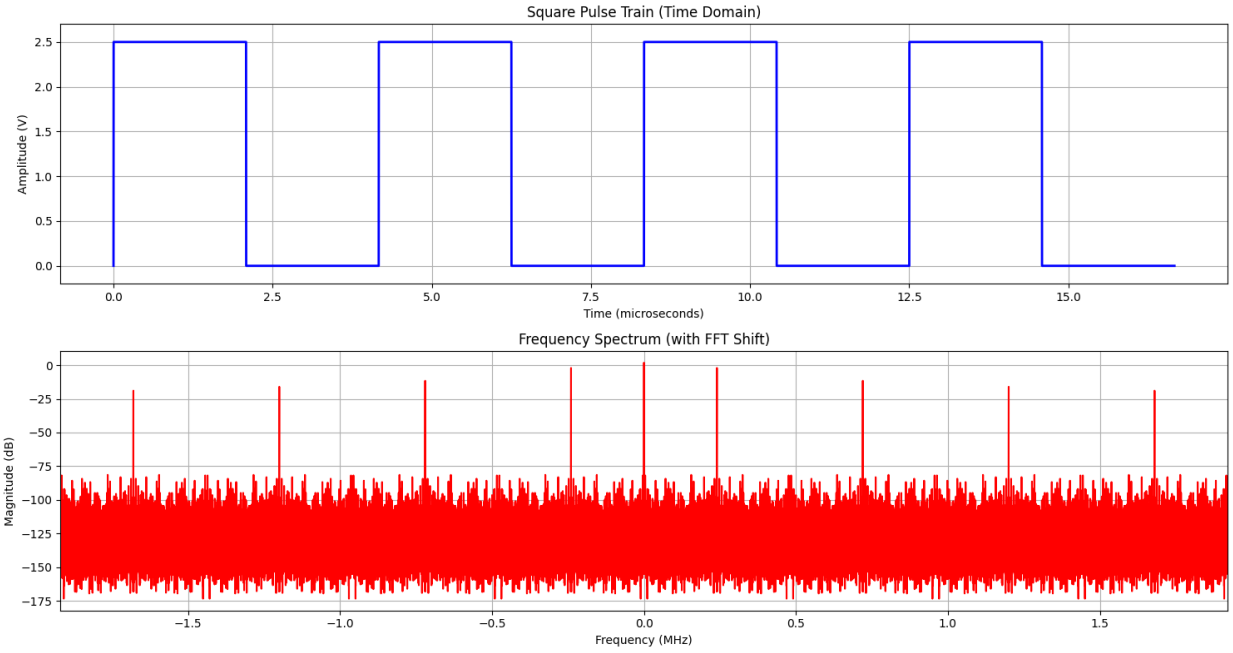
Harmonic Freqency Formula: $k \cdot \mathrm{f}_o$.
Harmonic Amplitude Formula: $\frac{A_{max}}{k\pi}$

1. DC Term (k = 0)
   (a) Frequency: 0
   $0 \cdot 240KHz = 0$
   (b) Amplitude: 1.25V
   $\frac{2.5V}{2} = 1.25V$
2. 1st Harmonic (k = 1)
   (a) Frequency: 240KHz
   $1 \cdot 240KHz = 240KHz$
   (b) Amplitude: 0.8V
   $\frac{2.5V}{\pi} \approx 0.8V$
3. 2nd Harmonic (k = 2)
   (a) Frequency: 480KHz
   $2 \cdot 240KHz = 480KHz$
   (b) Amplitude: .27V
   $\frac{2.5V}{3\pi} \approx .27$
4. 3rd Harmonic (k = 3)
   (a) Frequency: 720KHz
   $3 \cdot 240KHz = 720KHz$
   (b) Amplitude: .16V
   $\frac{2.5V}{3\pi} \approx .16$
5. 4th Harmonic (k = 4)
   (a) Frequency: 960KHz
   (b) Amplitude:

Problem 4

```
# Name: Justin Ngo
# PSU Email: jvn5439
# PSUID: 933902868
# Class: CMPEN 462
# Assignment: Homework 2
# Date: 02.17.2025

import numpy as np
import matplotlib.pyplot as plt

'''
References:
https://numpy.org/doc/stable/reference/routines.fft.html
https://www.geeksforgeeks.org/plotting-a-square-wave-using-matplotlib-numpy-and-
    scipy/
https://stackoverflow.com/questions/66000468/plot-square-wave-in-python
https://psu.instructure.com/courses/2378968/files/174067072?module_item_id
    =44321606 (Sines_FFT_example.m)
https://psu.instructure.com/courses/2378968/files/174066175?module_item_id
    =44321556 (pulse_trains.m)
https://psu.instructure.com/courses/2378968/files/174066916?module_item_id
    =44321597 (Example_tone_FFT.m)
'''


# Signal parameters
fo = 240e3      # Fundamental frequency (240 KHz)
fs = 4096 * fo  # Sample frequenct (~983MHz)
t = 250e-3      # Duration in seconds (250 ms)
A_max = 2.5     # Max Amplitude (2.5V)
A_min = 0.0     # Min Amplitude (0V)


# Time vector
t = np.arange(0, t, 1/fs)
samples = int(fs/fo) # samples per period

# Generate square pulse train
signal = A_max * (np.sign(np.sin(2 * np.pi * fo * t)) > 0)

# Calculate FFT
n = len(signal)  # Number of points
freqs = np.fft.fftfreq(n, d=1/fs)  # Frequency vector
fft_signal = np.fft.fft(signal)  # Compute FFT

# Apply fftshift and normalize FFT
freqs_shifted = np.fft.fftshift(freqs)
fft_signal_shifted = np.fft.fftshift(fft_signal)
fft_signal_normalized = fft_signal_shifted / n  # Remove FFT gain by dividing by n

pulses = 4
t_display = t[:int(pulses * fs / fo)] # Time vector for 4 pulses
signal_display = signal[:int(pulses * fs / fo)] # Signal vector for 4 pulses

# Graphs
plt.figure(figsize=(15, 8))
```

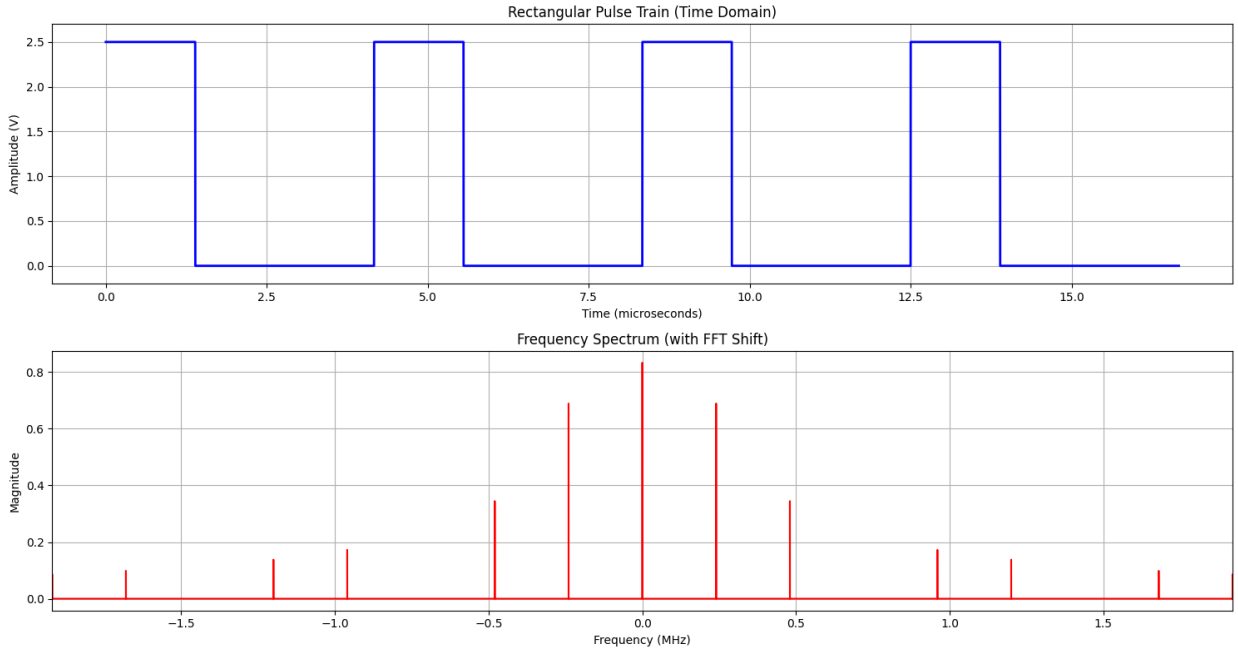
```
# Time domain plot
plt.subplot(2, 1, 1)
plt.plot(t_display * 1e6, signal_display, 'b-', linewidth=2)
plt.grid(True)
plt.xlabel('Time (microseconds)')
plt.ylabel('Amplitude (V)')
plt.title('Square Pulse Train (Time Domain)')
plt.ylim([A_min - 0.2, A_max + 0.2])

# Frequency domain plot with log scale for magnitude
plt.subplot(2, 1, 2)
plt.plot(freqs_shifted/1e6, 20 * np.log10(np.abs(fft_signal_normalized)), 'r-')
plt.grid(True)
plt.xlabel('Frequency (MHz)')
plt.ylabel('Magnitude (dB)')
plt.title('Frequency Spectrum (with FFT Shift)')
# Adjust x-axis limits to show frequencies centered around 0
plt.xlim([-8*fo/1e6, 8*fo/1e6])  # Show up to +-5th harmonic

plt.tight_layout()
plt.show()
```



Problem 5

```
# Name: Justin Ngo
# PSU Email: jvn5439
# PSUID: 933902868
# Class: CMPEN 462
# Assignment: Homework 2
# Date: 02.17.2025

import numpy as np
import matplotlib.pyplot as plt

'''
References:
https://numpy.org/doc/stable/reference/routines.fft.html
https://www.geeksforgeeks.org/plotting-a-square-wave-using-matplotlib-numpy-and-
    scipy/
https://stackoverflow.com/questions/66000468/plot-square-wave-in-python
https://psu.instructure.com/courses/2378968/files/174067072?module_item_id
    =44321606 (Sines_FFT_example.m)
https://psu.instructure.com/courses/2378968/files/174066175?module_item_id
    =44321556 (pulse_trains.m)
https://psu.instructure.com/courses/2378968/files/174066916?module_item_id
    =44321597 (Example_tone_FFT.m)
'''


# Signal parameters
fo = 240e3      # Fundamental frequency (240 KHz)
fs = 4096 * fo  # Sample frequency (~983MHz)
t = 250e-3      # Duration in seconds (250 ms)
A_max = 2.5     # Max Amplitude (2.5V)
A_min = 0.0     # Min Amplitude (0V)
duty_cycle = 1/3  # Duty cycle (1/3)

# Time vector
t = np.arange(0, t, 1/fs)
samples = int(fs/fo)  # samples per period

# Generate rectangular pulse train
signal = A_max * ((np.mod(t, 1/fo) < (duty_cycle / fo)).astype(float))

# Calculate FFT
n = len(signal)  # Number of points
freqs = np.fft.fftfreq(n, d=1/fs)  # Frequency vector
fft_signal = np.fft.fft(signal)  # Compute FFT

# Apply fftshift and normalize FFT
freqs_shifted = np.fft.fftshift(freqs)
fft_signal_shifted = np.fft.fftshift(fft_signal)
fft_signal_normalized = fft_signal_shifted / n  # Remove FFT gain by dividing by n

pulses = 4
t_display = t[:int(pulses * fs / fo)]  # Time vector for 4 pulses
signal_display = signal[:int(pulses * fs / fo)]  # Signal vector for 4 pulses

# Graphs
plt.figure(figsize=(15, 8))
```

```
# Time domain plot
plt.subplot(2, 1, 1)
plt.plot(t_display * 1e6, signal_display, 'b-', linewidth=2)
plt.grid(True)
plt.xlabel('Time (microseconds)')
plt.ylabel('Amplitude (V)')
plt.title('Rectangular Pulse Train (Time Domain)')
plt.ylim([A_min - 0.2, A_max + 0.2])

# Frequency domain plot
plt.subplot(2, 1, 2)
plt.plot(freqs_shifted/1e6, np.abs(fft_signal_normalized), 'r-')
plt.grid(True)
plt.xlabel('Frequency (MHz)')
plt.ylabel('Magnitude')
plt.title('Frequency Spectrum (with FFT Shift)')
# Adjust x-axis limits to show frequencies centered around 0
plt.xlim([-8*fo/1e6, 8*fo/1e6])  # Show up to +-5th harmonic

plt.tight_layout()
plt.show()
```



The main difference between part 2 and part 4's time graphs are that the active state in part 4 is shorter than in part 2. The difference between their frequency graphs is that the magnitudes of the later harmonics are higher, and the freqencies are more condensed and closer together.