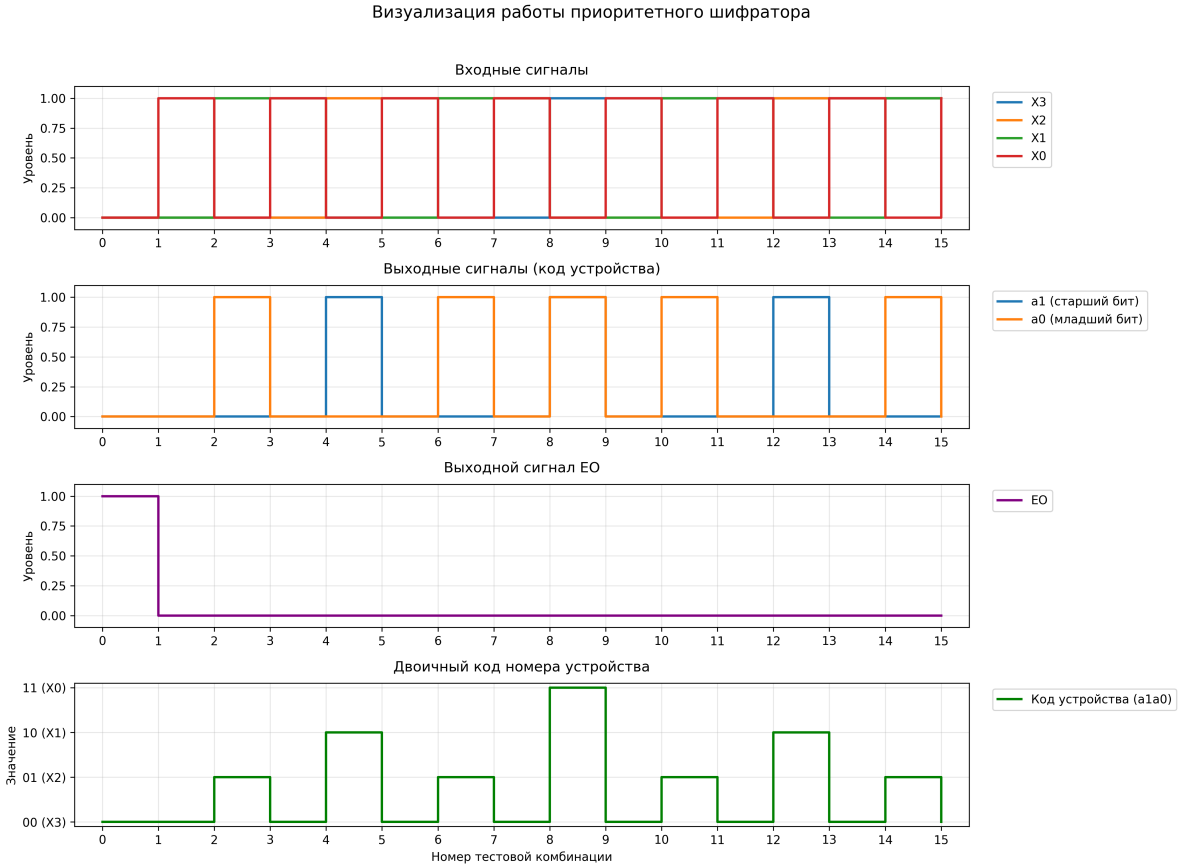
```
import matplotlib.pyplot as plt
import numpy as np
from IPython.display import display, Markdown

def priority_encoder(X):
    """
                  4      (X3, X2, X1, X0)
    X3              , X0 -
              (a1, a0, EO)
    """
    if X[3]:
        return (0, 0, 0)
    elif X[2]:
        return (0, 1, 0)
    elif X[1]:
        return (1, 0, 0)
    elif X[0]:
        return (1, 1, 0)
    else:
        return (0, 0, 1)  # EO=1           0

#                          (16      )
input_combinations = [(x3, x2, x1, x0)
                      for x3 in [0, 1]
                      for x2 in [0, 1]
                      for x1 in [0, 1]
                      for x0 in [0, 1]]

#
outputs = [priority_encoder(inp) for inp in input_combinations]

#
table_header = "| X3 | X2 | X1 | X0 | a1 | a0 | EO |\n|----|----|----|----|----|----|----|"
table_rows = []
for inp, out in zip(input_combinations, outputs):
    table_rows.append(f"| {inp[0]} | {inp[1]} | {inp[2]} | {inp[3]} | {out[0]} | {out[1]} |

display(Markdown("##                          "))
display(Markdown(table_header + "\n" + "\n".join(table_rows)))

#
plt.figure(figsize=(14, 10))
```

```
plt.suptitle('                          ', y=1.02, fontsize=14)

#
ax1 = plt.subplot(4, 1, 1)
for i in range(4):
    signal = [inp[i] for inp in input_combinations]
    plt.step(np.arange(16), signal, where='post', label=f'X{3-i}', linewidth=2)
plt.title('         ', pad=10)
plt.ylabel('    ')
plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left')
plt.grid(True, alpha=0.3)
plt.ylim(-0.1, 1.1)
plt.xlim(-0.5, 15.5)
plt.xticks(np.arange(16))

#           a1, a0
ax2 = plt.subplot(4, 1, 2, sharex=ax1)
a1_signal = [out[0] for out in outputs]
a0_signal = [out[1] for out in outputs]
plt.step(np.arange(16), a1_signal, where='post', label='a1 (      )', linewidth=2)
plt.step(np.arange(16), a0_signal, where='post', label='a0 (      )', linewidth=2)
plt.title('          (        )', pad=10)
plt.ylabel('    ')
plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left')
plt.grid(True, alpha=0.3)
plt.ylim(-0.1, 1.1)

#          EO
ax3 = plt.subplot(4, 1, 3, sharex=ax1)
eo_signal = [out[2] for out in outputs]
plt.step(np.arange(16), eo_signal, where='post', label='EO', linewidth=2, color='purple')
plt.title('          EO', pad=10)
plt.ylabel('    ')
plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left')
plt.grid(True, alpha=0.3)
plt.ylim(-0.1, 1.1)

#
ax4 = plt.subplot(4, 1, 4, sharex=ax1)
device_code = [out[0]*2 + out[1] for out in outputs]
plt.step(np.arange(16), device_code, where='post', label='         (a1a0)',
        linewidth=2, color='green')
```

```
plt.title('                    ', pad=10)
plt.xlabel('                ')
plt.ylabel('     ')
plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left')
plt.grid(True, alpha=0.3)
plt.ylim(-0.1, 3.1)
plt.yticks([0, 1, 2, 3], ['00 (X3)', '01 (X2)', '10 (X1)', '11 (X0)'])

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

X3	X2	X1	X0	a1	a0	EO
0	0	0	0	0	0	1
0	0	0	1	0	0	0
0	0	1	0	0	1	0
0	0	1	1	0	0	0
0	1	0	0	1	0	0
0	1	0	1	0	0	0
0	1	1	0	0	1	0
0	1	1	1	0	0	0
1	0	0	0	1	1	0
1	0	0	1	0	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	0
1	1	0	0	1	0	0
1	1	0	1	0	0	0
1	1	1	0	0	1	0
1	1	1	1	0	0	0


Визуализация работы приоритетного шифратора
Входные сигналы
Уровень
X3
X2
X1
X0
Выходные сигналы (код устройства)
Уровень
a1 (старший бит)
a0 (младший бит)
Выходной сигнал EO
Уровень
EO
Двоичный код номера устройства
Значение
11 (X0)
10 (X1)
01 (X2)
00 (X3)
Код устройства (a1a0)
Номер тестовой комбинации