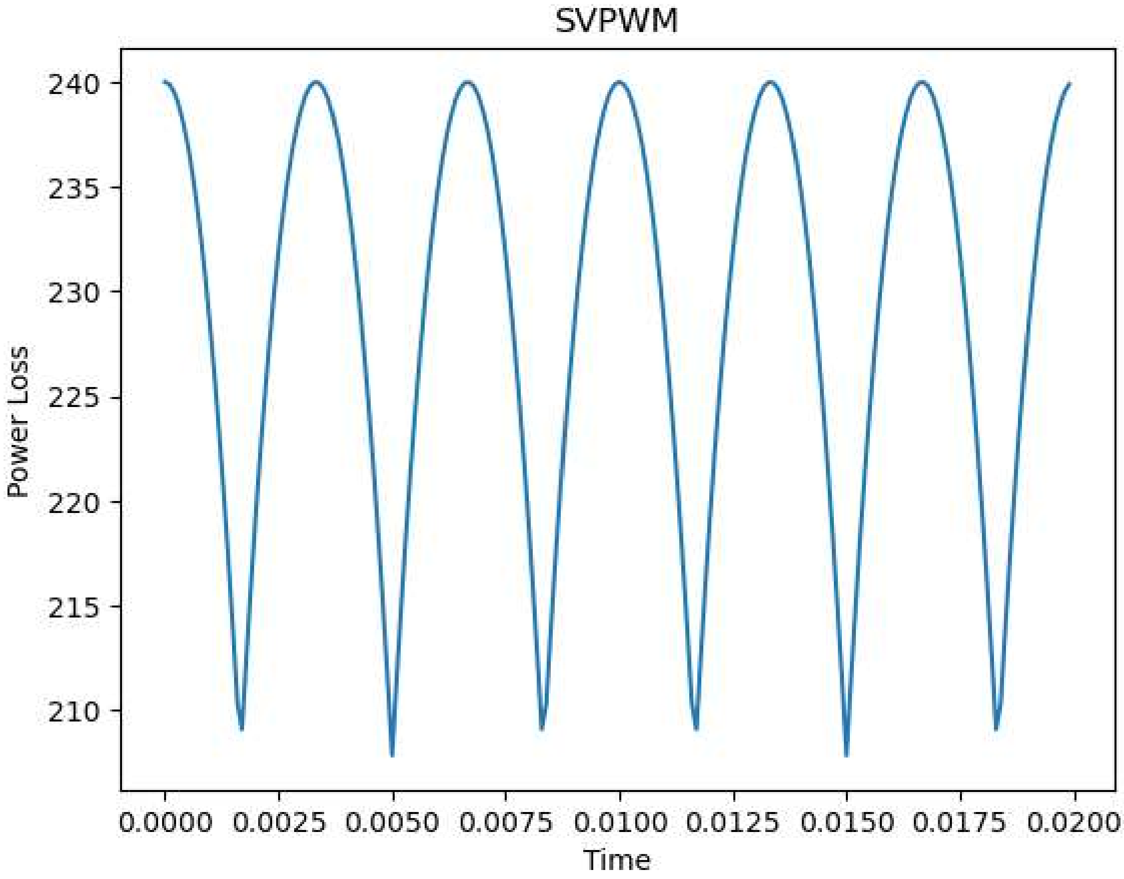
Switching loss curve for different switching techniques.

In [1]:

```
## SVPWM
import math
import numpy as np
from math import cos
import matplotlib.pyplot as plt
pi = math.pi

dic = {0:[0,0,0],1:[1,0,0],2:[1,1,0],3:[0,1,0],4:[0,1,1],5:[0,0,1],6:[1,0,1],7:[1,1,1]

def switching(a,b,c):
    temp = []
    a = dic[a];b = dic[b];c = dic[c]
    for i in range(3):
        if a[i] == b[i] == c[i]:
            temp.append(0)
        elif a[i] != b[i] and b[i] == c[i]:
            temp.append(1)
        elif a[i] == b[i] and b[i] != c[i]:
            temp.append(1)
        else:
            temp.append(2)
    return temp

ma = .8
Vdc = 600
w = 2*pi*50
t = 0
f = 50
theta = w*t
# ia = ma *(Vdc/2)*cos(w*t)
# ib = ma *(Vdc/2)*cos(w*t+2*pi/3)
# ic = ma *(Vdc/2)*cos(w*t-(2*pi/3))
fs = 10000
Tsw = fs #(fs/(2*fsw) not (fs/(2*Tsw))
Im = 1

Time = []
Power = []

na = nb = nc = 1

for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t)
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3)
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3))

    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("SVPWM")
x1 = Time
```

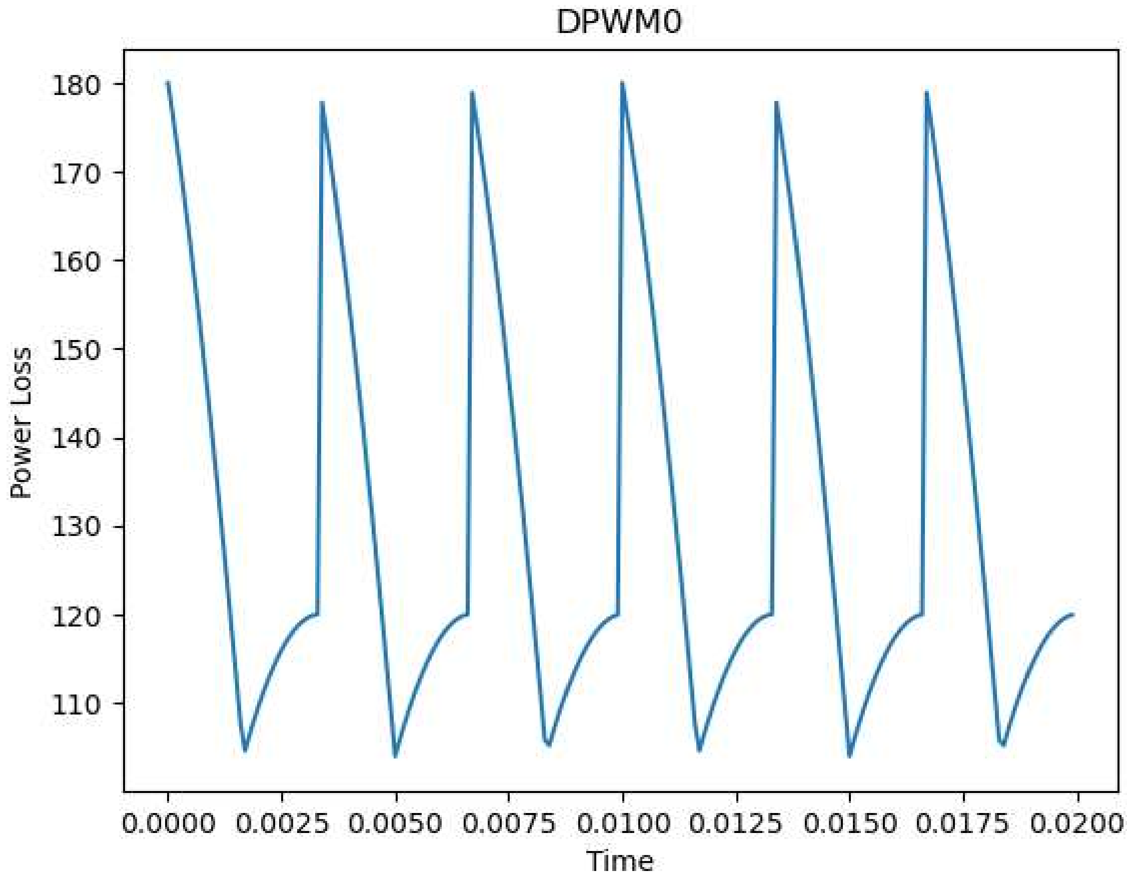
```
y1 = Power
plt.show()
```



In [2]:

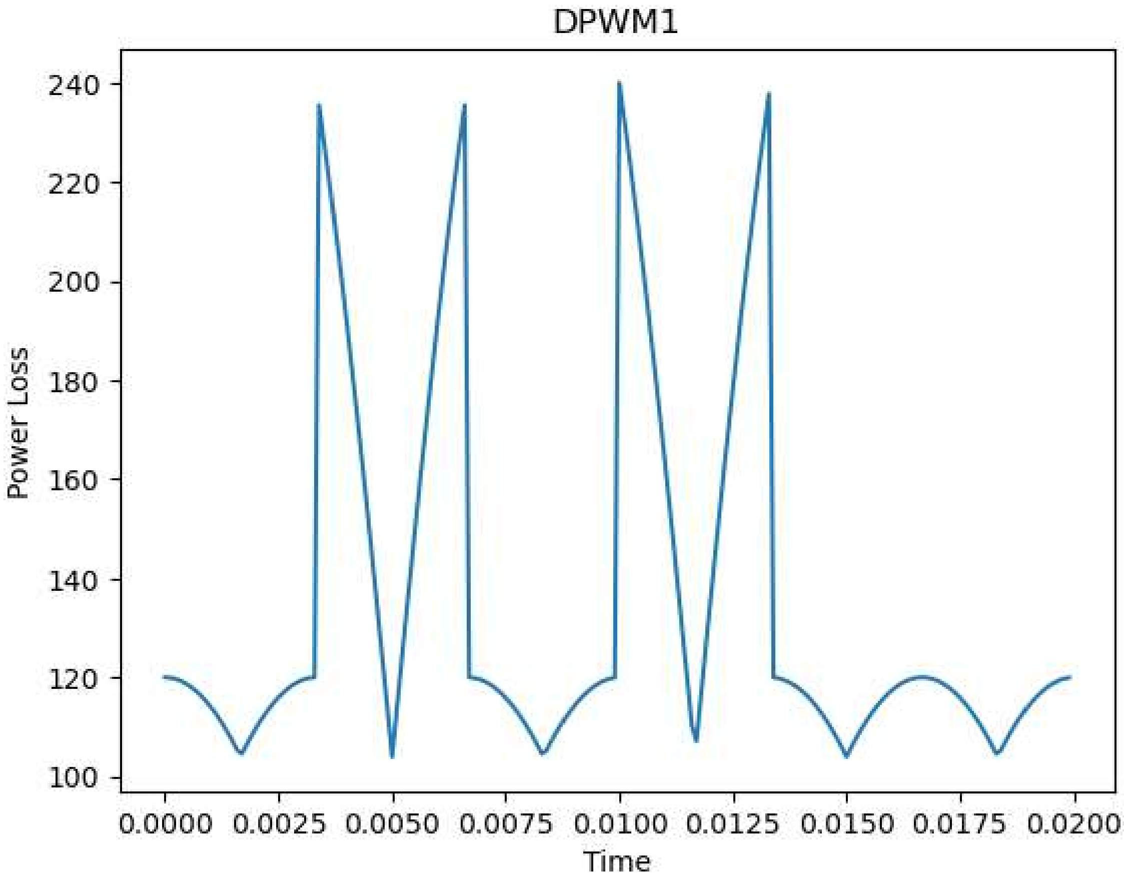
```
#for DPWM0
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t)
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3)
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3))
    if t<1/(6*f):
        na,nb,nc = switching(0,1,2)
    elif t<2/(6*f):
        na,nb,nc = switching(7,2,3)
    elif t<3/(6*f):
        na,nb,nc = switching(0,3,4)
    elif t<4/(6*f):
        na,nb,nc = switching(7,4,5)
    elif t<5/(6*f):
        na,nb,nc = switching(0,5,6)
    else:
        na,nb,nc = switching(7,6,1)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM0")
```

```
x2 = Time
y2 = Power
plt.show()
```



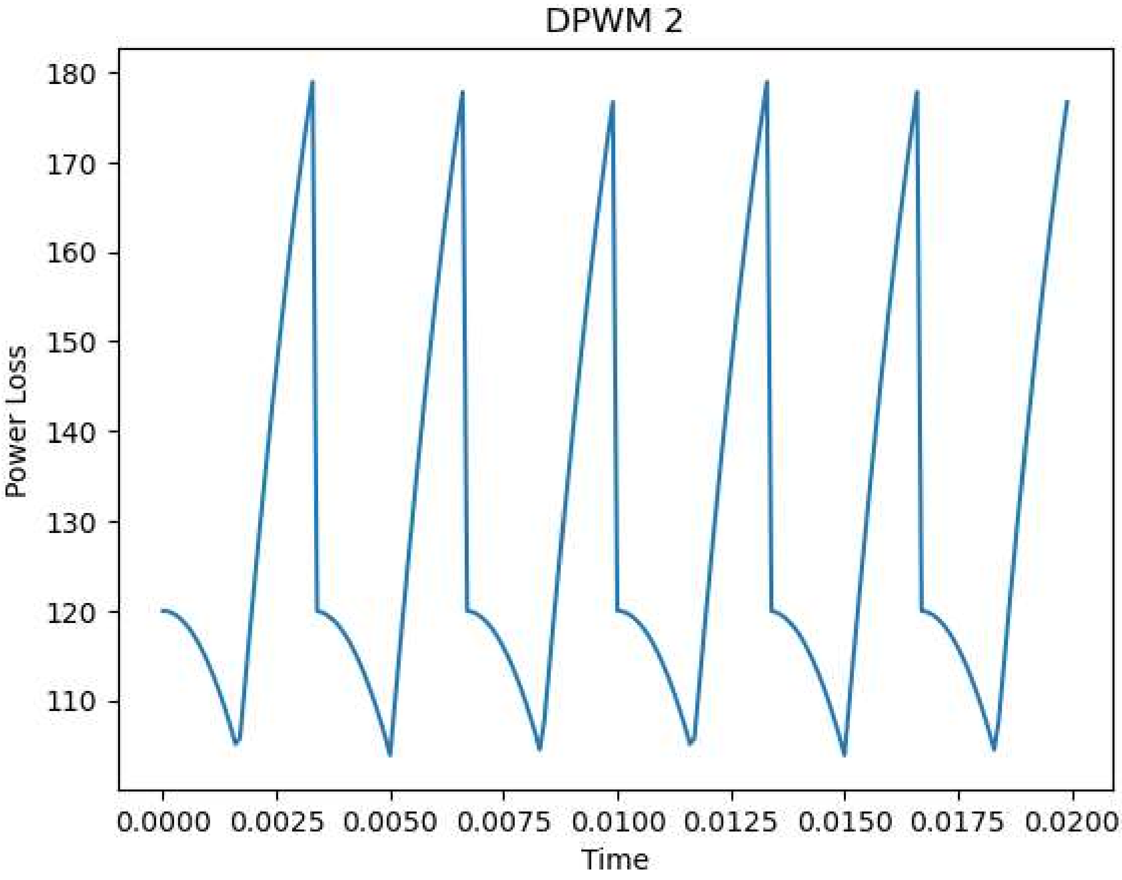
In [3]:

```
#DPWM 1
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t)
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3)
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3))

    if t<1/(12*f):
        na,nb,nc = switching(1,2,7)
    elif t<2/(12*f):
        na,nb,nc = switching(0,1,2)
    elif t<3/(12*f):
        na,nb,nc = switching(7,3,2)
    elif t<4/(12*f):
        na,nb,nc = switching(0,2,3)
    elif t<5/(12*f):
        na,nb,nc = switching(7,4,3)
    elif t<6/(12*f):
        na,nb,nc = switching(0,3,4)
    elif t<7/(12*f):
        na,nb,nc = switching(7,5,4)
    elif t<8/(12*f):
        na,nb,nc = switching(0,4,5)
    elif t<9/(12*f):
        na,nb,nc = switching(7,6,5)
    elif t<10/(12*f):
        na,nb,nc = switching(0,5,6)
    elif t<11/(12*f):
        na,nb,nc = switching(0,1,6)
    else:
        na,nb,nc = switching(7,6,1)

    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM1")
x3 = Time
y3 = Power
plt.show()
```



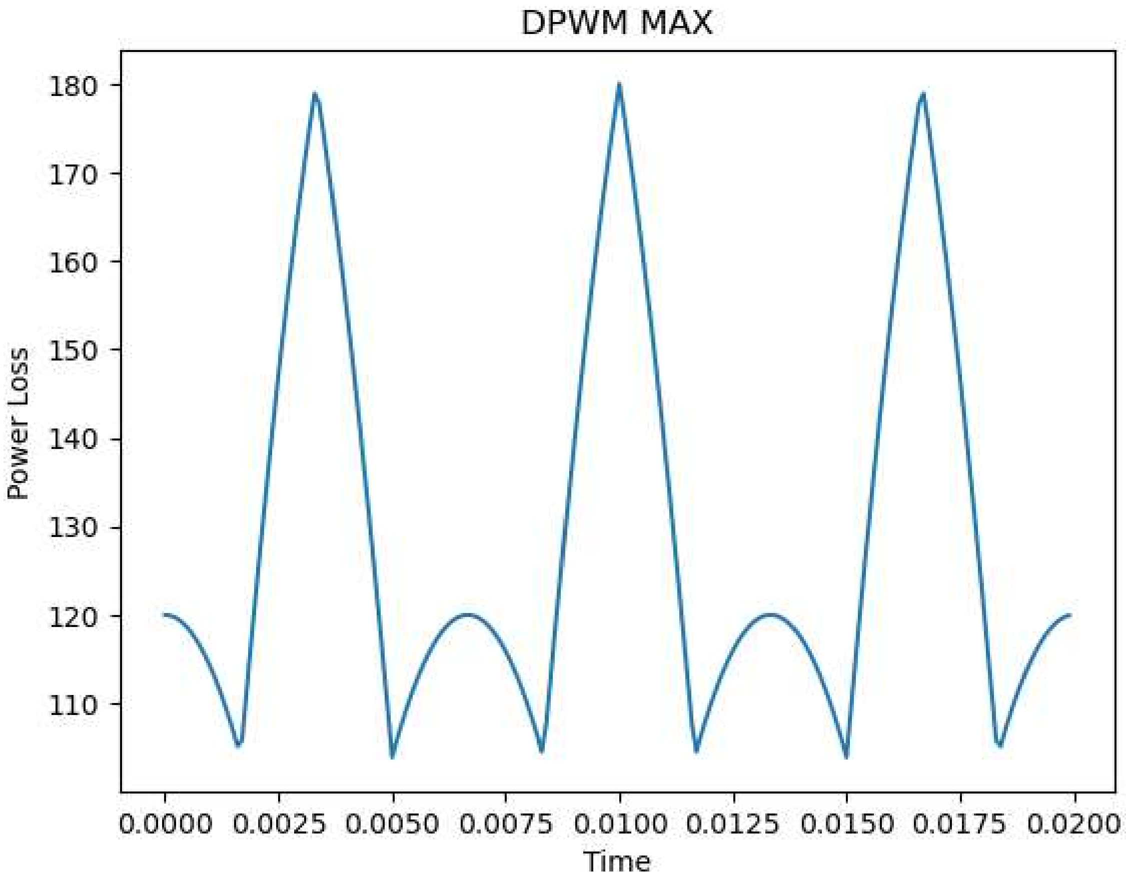
In [4]:

```
#for DPWM2
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t)
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3)
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3))
    if t<1/(6*f):
        na,nb,nc = switching(7,2,1)
    elif t<2/(6*f):
        na,nb,nc = switching(0,3,2)
    elif t<3/(6*f):
        na,nb,nc = switching(7,4,3)
    elif t<4/(6*f):
        na,nb,nc = switching(0,5,4)
    elif t<5/(6*f):
        na,nb,nc = switching(7,6,5)
    else:
        na,nb,nc = switching(0,1,6)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM 2")
x4 = Time
y4 = Power
plt.show()
```



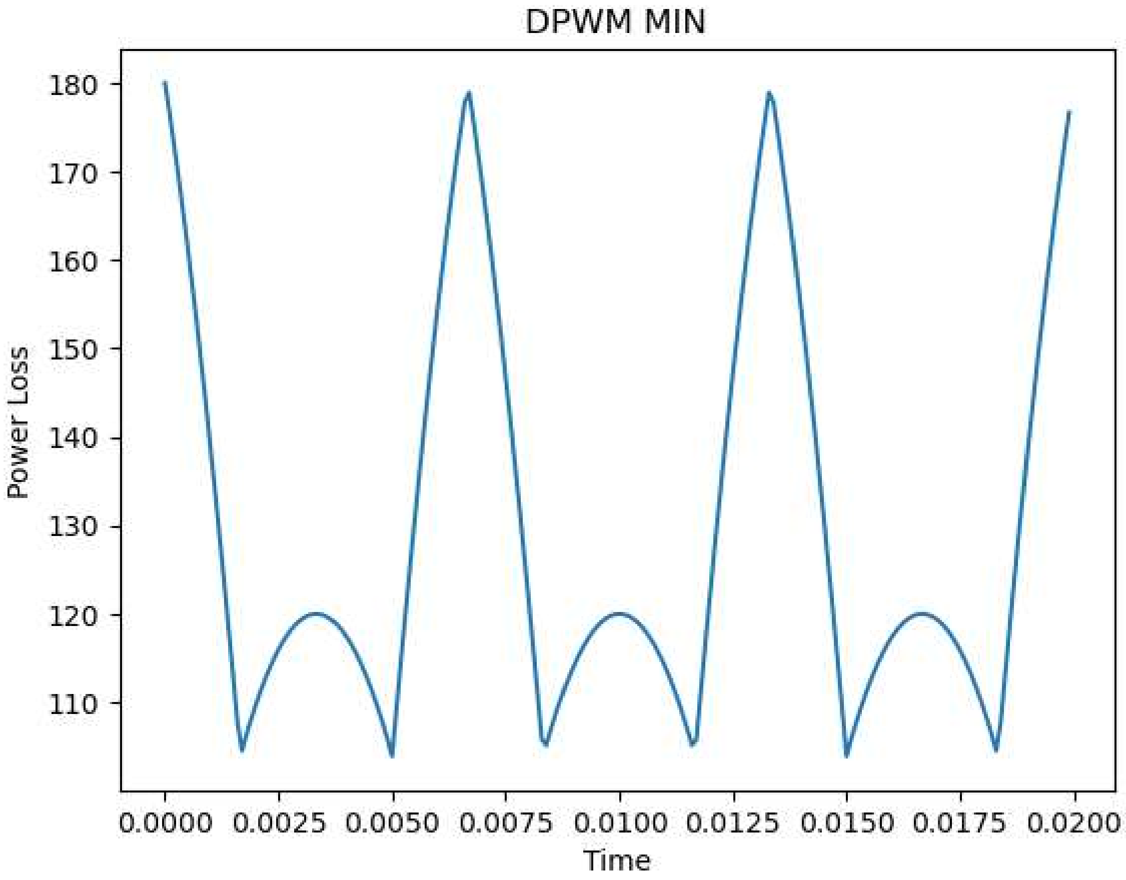
In [5]:

```
#for DPWM MAX
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t)
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3)
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3))
    if t<1/(6*f):
        na,nb,nc = switching(1,2,7)
    elif t<2/(6*f):
        na,nb,nc = switching(3,2,7)
    elif t<3/(6*f):
        na,nb,nc = switching(3,4,7)
    elif t<4/(6*f):
        na,nb,nc = switching(5,4,7)
    elif t<5/(6*f):
        na,nb,nc = switching(5,6,7)
    else:
        na,nb,nc = switching(1,6,7)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM MAX")
x5 = Time
y5 = Power
plt.show()
```



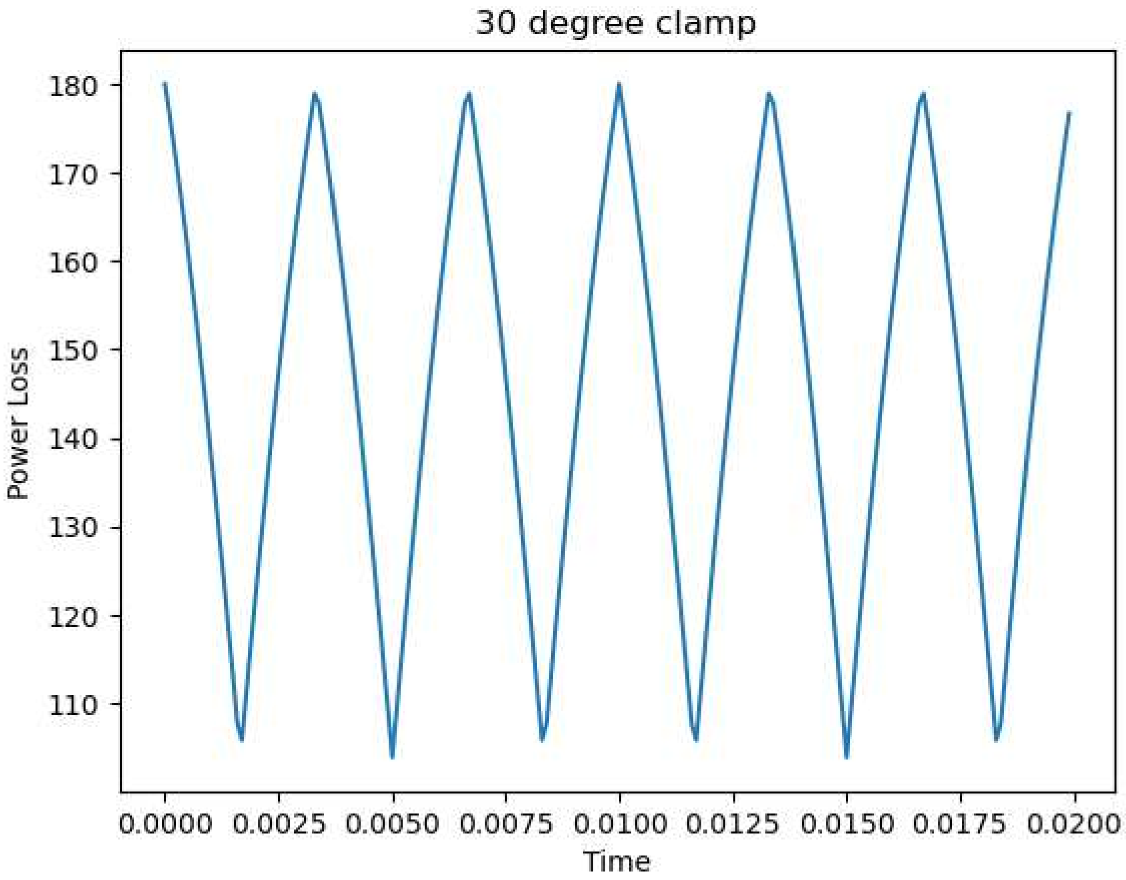
In [6]:

```
#for DPWM MIN
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t)
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3)
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3))
    if t<1/(6*f):
        na,nb,nc = switching(0,1,2)
    elif t<2/(6*f):
        na,nb,nc = switching(0,3,2)
    elif t<3/(6*f):
        na,nb,nc = switching(0,3,4)
    elif t<4/(6*f):
        na,nb,nc = switching(0,5,4)
    elif t<5/(6*f):
        na,nb,nc = switching(0,5,6)
    else:
        na,nb,nc = switching(0,1,6)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM MIN")
x6 = Time
y6 = Power
plt.show()
```



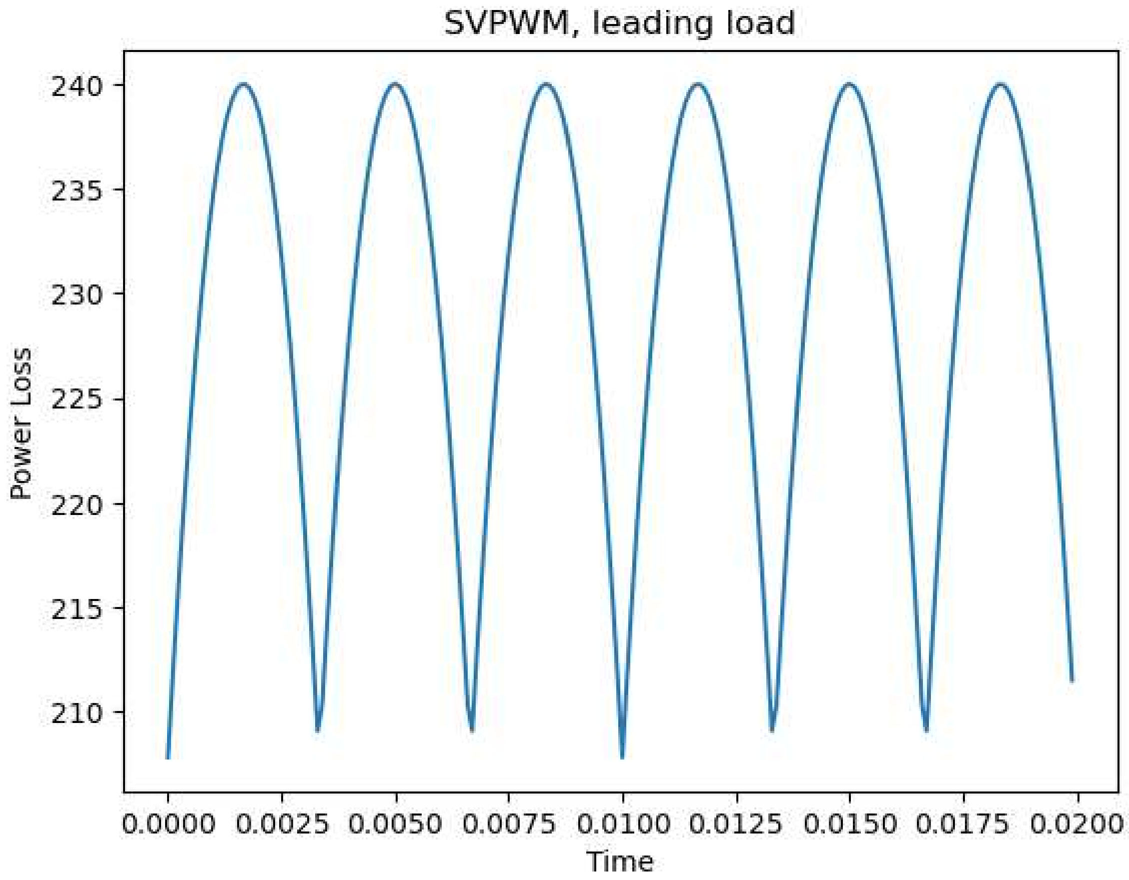
In [7]:

```
#30 degree clamp
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t)
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3)
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3))
    if t<1/(12*f):
        na,nb,nc = switching(0,1,2)
    elif t<2/(12*f):
        na,nb,nc = switching(7,2,1)
    elif t<3/(12*f):
        na,nb,nc = switching(0,2,3)
    elif t<4/(12*f):
        na,nb,nc = switching(7,3,2)
    elif t<5/(12*f):
        na,nb,nc = switching(0,3,4)
    elif t<6/(12*f):
        na,nb,nc = switching(7,4,3)
    elif t<7/(12*f):
        na,nb,nc = switching(0,4,5)
    elif t<8/(12*f):
        na,nb,nc = switching(7,5,4)
    elif t<9/(12*f):
        na,nb,nc = switching(0,5,6)
    elif t<10/(12*f):
        na,nb,nc = switching(7,6,5)
    elif t<11/(12*f):
        na,nb,nc = switching(0,6,1)
    else:
        na,nb,nc = switching(7,1,6)

    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("30 degree clamp")
x7 = Time
y7 = Power
plt.show()
```



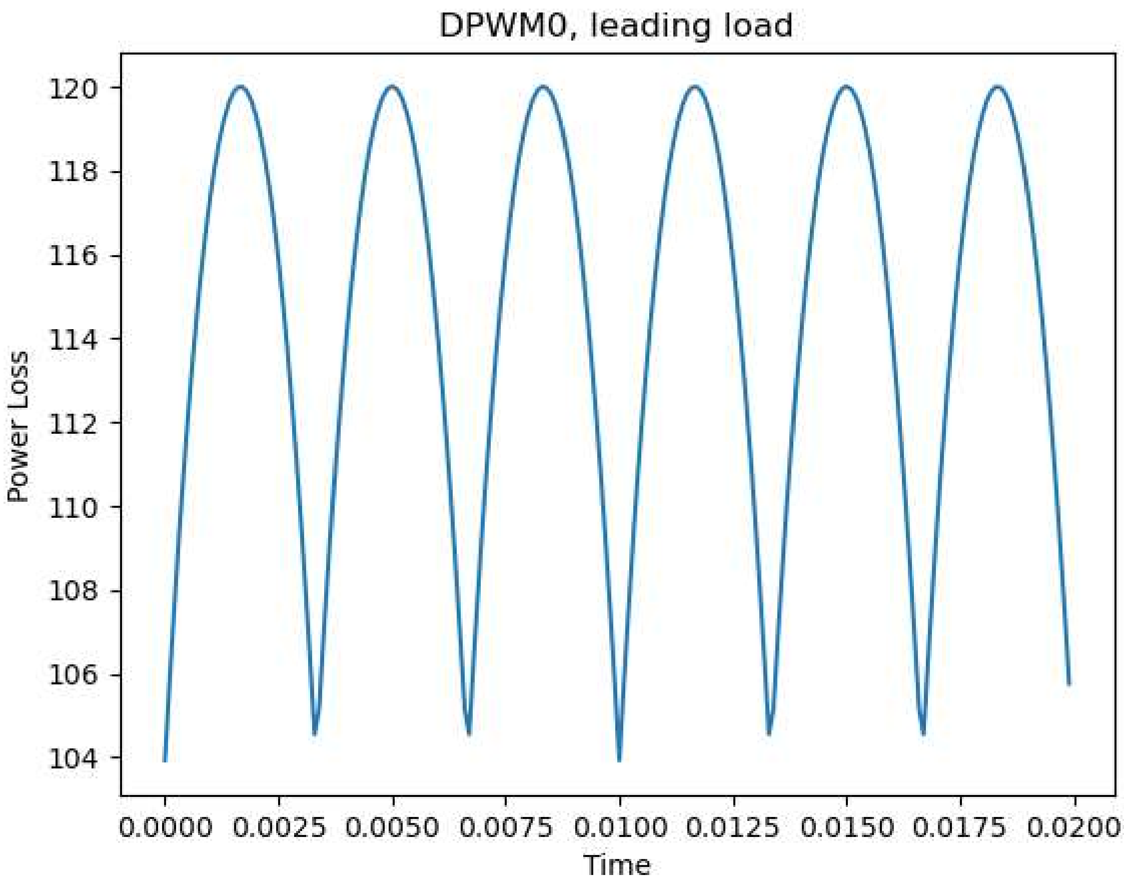
In [8]:

```
#Using SVPWM, 30 degree leading load condition
Time = []
Power = []

na = nb = nc = 1

for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t+(pi/6))
    ib = ma *(Vdc/2)*cos(w*t+(pi/6)-(2*pi/3))
    ic = ma *(Vdc/2)*cos(w*t+(pi/6)+(2*pi/3))

    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("SVPWM, leading load")
x8 = Time
y8 = Power
plt.show()
```



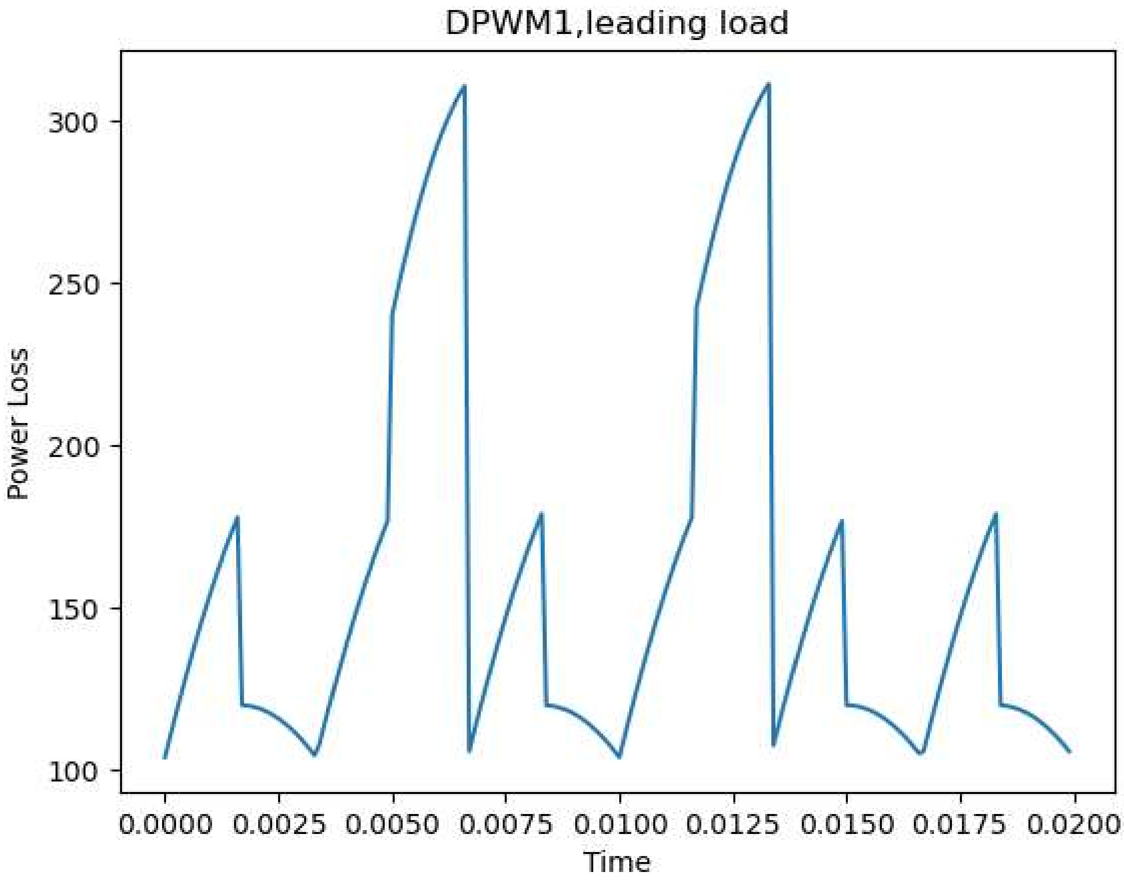
In [9]:

```
#Using DPWM0, 30 degree Leading Load condition
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t+(pi/6))
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3+(pi/6))
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3)+(pi/6))
    if t<1/(6*f):
        na,nb,nc = switching(0,1,2)
    elif t<2/(6*f):
        na,nb,nc = switching(7,2,3)
    elif t<3/(6*f):
        na,nb,nc = switching(0,3,4)
    elif t<4/(6*f):
        na,nb,nc = switching(7,4,5)
    elif t<5/(6*f):
        na,nb,nc = switching(0,5,6)
    else:
        na,nb,nc = switching(7,6,1)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM0, leading load")
x9 = Time
y9 = Power
plt.show()
```



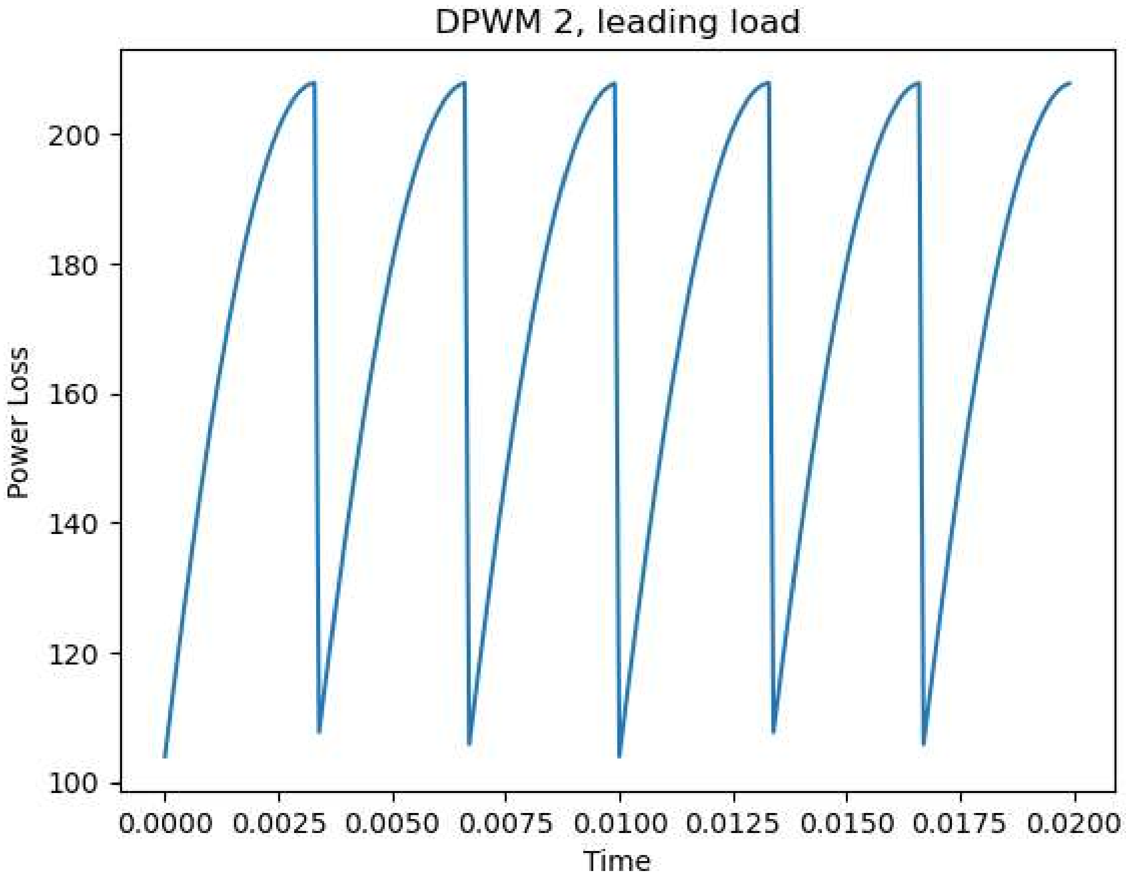
In [10]:

```
#DPWM 1
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t+(pi/6))
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3+(pi/6))
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3)+(pi/6))

    if t<1/(12*f):
        na,nb,nc = switching(1,2,7)
    elif t<2/(12*f):
        na,nb,nc = switching(0,1,2)
    elif t<3/(12*f):
        na,nb,nc = switching(7,3,2)
    elif t<4/(12*f):
        na,nb,nc = switching(0,2,3)
    elif t<5/(12*f):
        na,nb,nc = switching(7,4,3)
    elif t<6/(12*f):
        na,nb,nc = switching(0,3,4)
    elif t<7/(12*f):
        na,nb,nc = switching(7,5,4)
    elif t<8/(12*f):
        na,nb,nc = switching(0,4,5)
    elif t<9/(12*f):
        na,nb,nc = switching(7,6,5)
    elif t<10/(12*f):
        na,nb,nc = switching(0,5,6)
    elif t<11/(12*f):
        na,nb,nc = switching(0,1,6)
    else:
        na,nb,nc = switching(7,6,1)

    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM1,leading load")
x10 = Time
y10 = Power
plt.show()
```



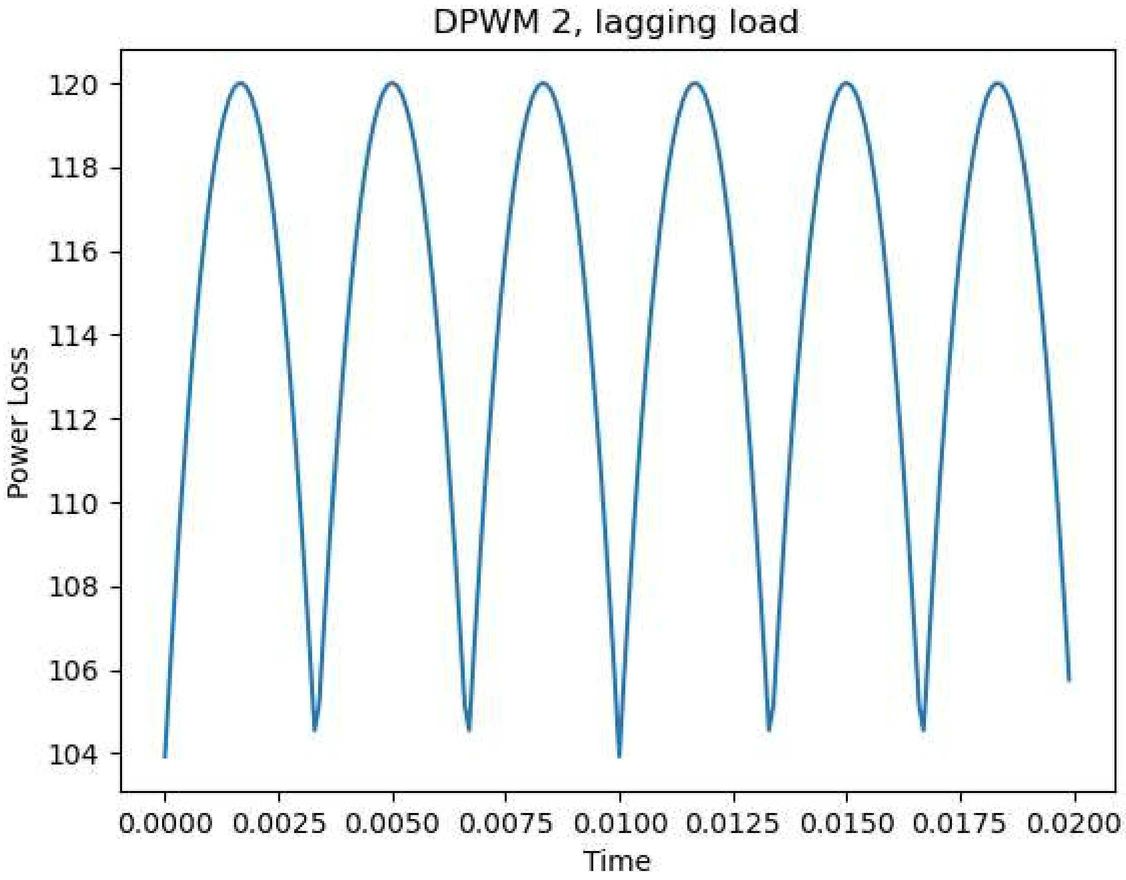
In [11]:

```
#for DPWM2
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t+(pi/6))
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3+(pi/6))
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3)+(pi/6))
    if t<1/(6*f):
        na,nb,nc = switching(7,2,1)
    elif t<2/(6*f):
        na,nb,nc = switching(0,3,2)
    elif t<3/(6*f):
        na,nb,nc = switching(7,4,3)
    elif t<4/(6*f):
        na,nb,nc = switching(0,5,4)
    elif t<5/(6*f):
        na,nb,nc = switching(7,6,5)
    else:
        na,nb,nc = switching(0,1,6)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM 2, leading load")
x11 = Time
y11 = Power
plt.show()
```



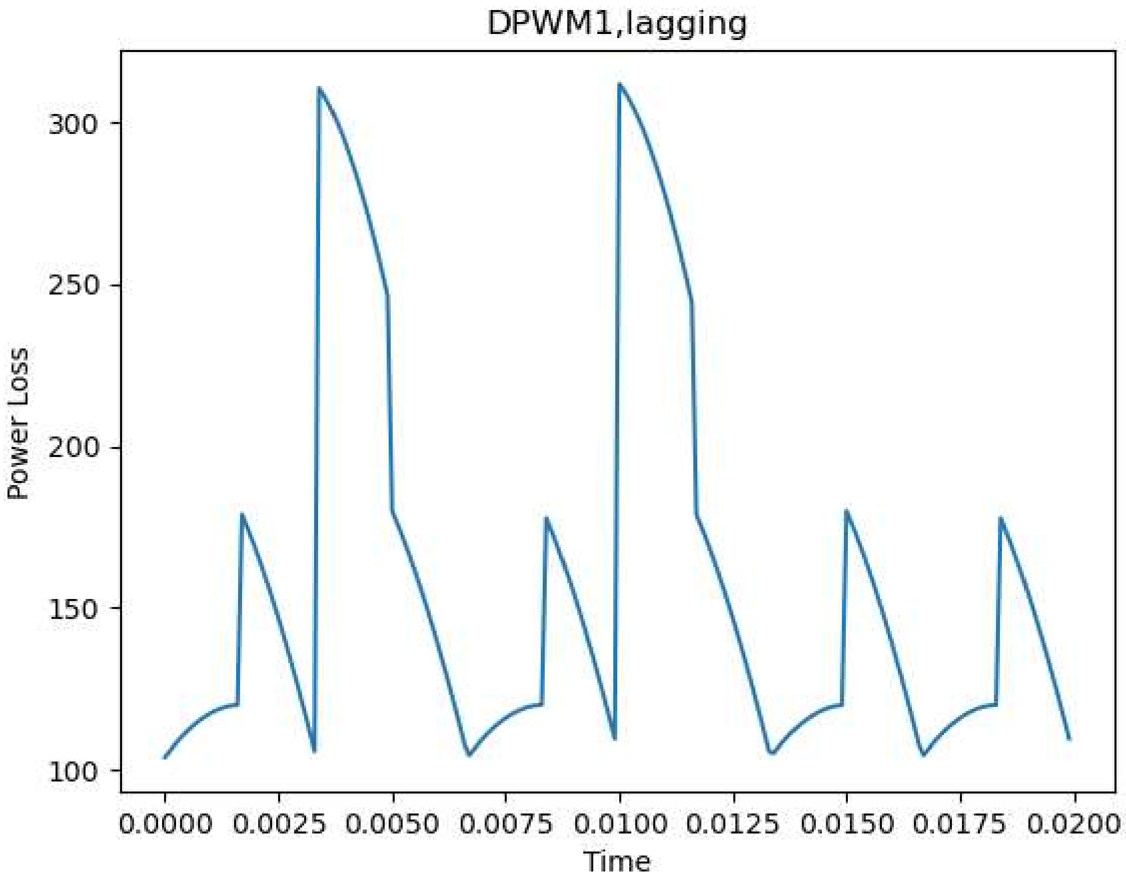
In [12]:

```
#for DPWM2
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t-(pi/6))
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3-(pi/6))
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3)-(pi/6))
    if t<1/(6*f):
        na,nb,nc = switching(7,2,1)
    elif t<2/(6*f):
        na,nb,nc = switching(0,3,2)
    elif t<3/(6*f):
        na,nb,nc = switching(7,4,3)
    elif t<4/(6*f):
        na,nb,nc = switching(0,5,4)
    elif t<5/(6*f):
        na,nb,nc = switching(7,6,5)
    else:
        na,nb,nc = switching(0,1,6)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM 2, lagging load")
x12 = Time
y12 = Power
plt.show()
```



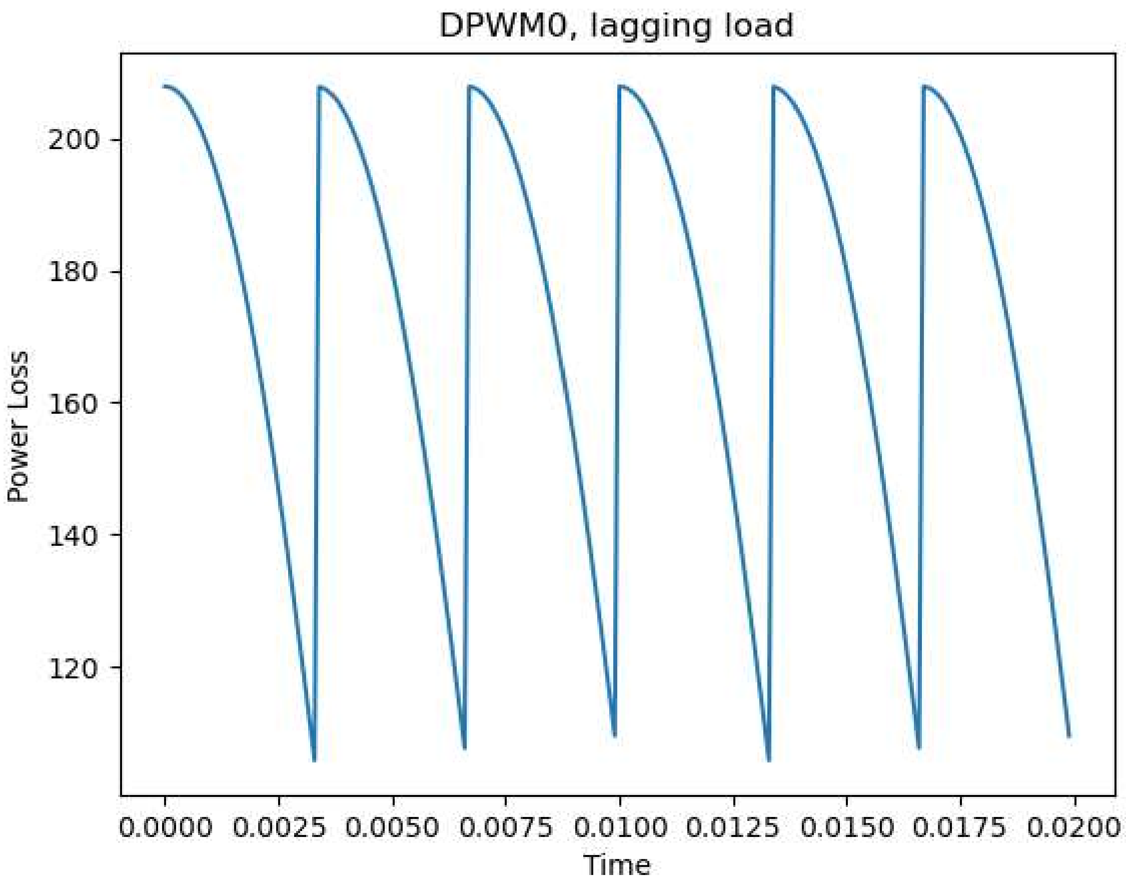
In [13]:

```
#DPWM 1
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t-(pi/6))
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3-(pi/6))
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3)-(pi/6))

    if t<1/(12*f):
        na,nb,nc = switching(1,2,7)
    elif t<2/(12*f):
        na,nb,nc = switching(0,1,2)
    elif t<3/(12*f):
        na,nb,nc = switching(7,3,2)
    elif t<4/(12*f):
        na,nb,nc = switching(0,2,3)
    elif t<5/(12*f):
        na,nb,nc = switching(7,4,3)
    elif t<6/(12*f):
        na,nb,nc = switching(0,3,4)
    elif t<7/(12*f):
        na,nb,nc = switching(7,5,4)
    elif t<8/(12*f):
        na,nb,nc = switching(0,4,5)
    elif t<9/(12*f):
        na,nb,nc = switching(7,6,5)
    elif t<10/(12*f):
        na,nb,nc = switching(0,5,6)
    elif t<11/(12*f):
        na,nb,nc = switching(0,1,6)
    else:
        na,nb,nc = switching(7,6,1)

    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM1,lagging")

x13 = Time
y13 = Power
plt.show()
```



In [14]:

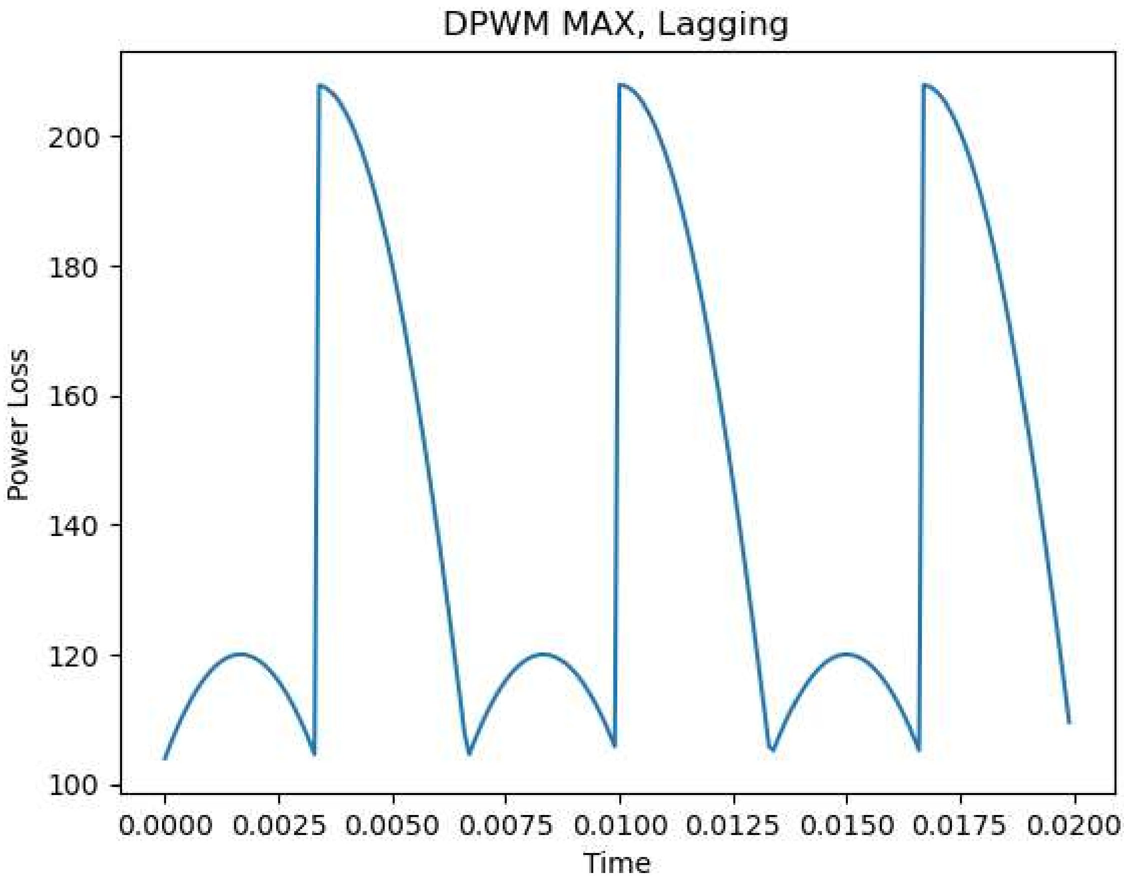
```python
#Using DPWM0, 30 degree Leading Load condition
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t-(pi/6))
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3-(pi/6))
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3)-(pi/6))
    if t<1/(6*f):
        na,nb,nc = switching(0,1,2)
    elif t<2/(6*f):
        na,nb,nc = switching(7,2,3)
    elif t<3/(6*f):
        na,nb,nc = switching(0,3,4)
    elif t<4/(6*f):
        na,nb,nc = switching(7,4,5)
    elif t<5/(6*f):
        na,nb,nc = switching(0,5,6)
    else:
        na,nb,nc = switching(7,6,1)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM0, lagging load")
x14 = Time
y14 = Power
plt.show()
```



In [15]:

```
#for DPWM MAX
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t-(pi/6))
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3-(pi/6))
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3)-(pi/6))
    if t<1/(6*f):
        na,nb,nc = switching(1,2,7)
    elif t<2/(6*f):
        na,nb,nc = switching(3,2,7)
    elif t<3/(6*f):
        na,nb,nc = switching(3,4,7)
    elif t<4/(6*f):
        na,nb,nc = switching(5,4,7)
    elif t<5/(6*f):
        na,nb,nc = switching(5,6,7)
    else:
        na,nb,nc = switching(1,6,7)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM MAX, Lagging")
x15 = Time
```

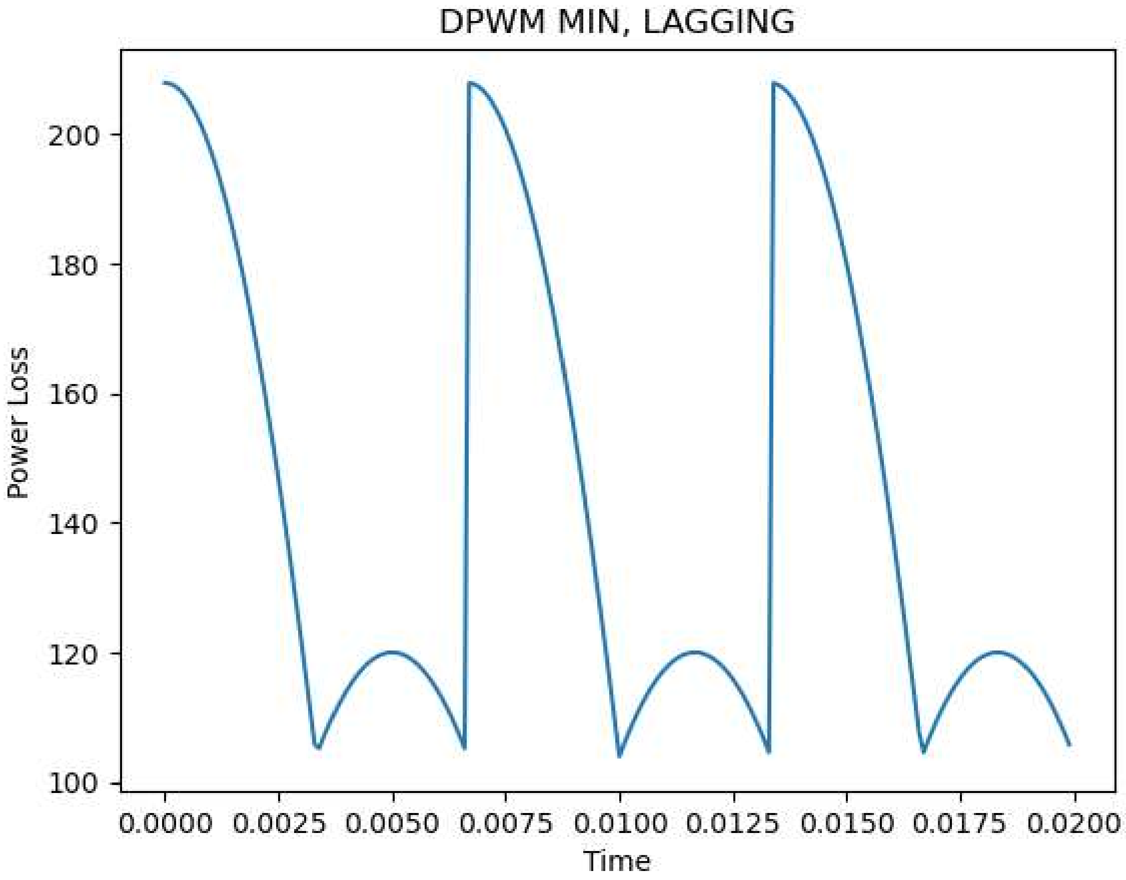
```
y15 = Power
plt.show()
```



In [16]:

```
#for DPWM MIN
Time = []
Power = []
for n in range(0,200):
    t = n/(50*200)
    ia = ma *(Vdc/2)*cos(w*t-(pi/6))
    ib = ma *(Vdc/2)*cos(w*t-2*pi/3-(pi/6))
    ic = ma *(Vdc/2)*cos(w*t+(2*pi/3)-(pi/6))
    if t<1/(6*f):
        na,nb,nc = switching(0,1,2)
    elif t<2/(6*f):
        na,nb,nc = switching(0,3,2)
    elif t<3/(6*f):
        na,nb,nc = switching(0,3,4)
    elif t<4/(6*f):
        na,nb,nc = switching(0,5,4)
    elif t<5/(6*f):
        na,nb,nc = switching(0,5,6)
    else:
        na,nb,nc = switching(0,1,6)
    loss = (na*abs(ia)+nb*abs(ib)+nc*abs(ic))*(fs)/(Im*2*Tsw)
    #print(na,nb,nc)
    Time.append(t)
    Power.append(loss)
plt.plot(Time, Power)
plt.xlabel("Time")
plt.ylabel("Power Loss")
plt.title("DPWM MIN, LAGGING")
```

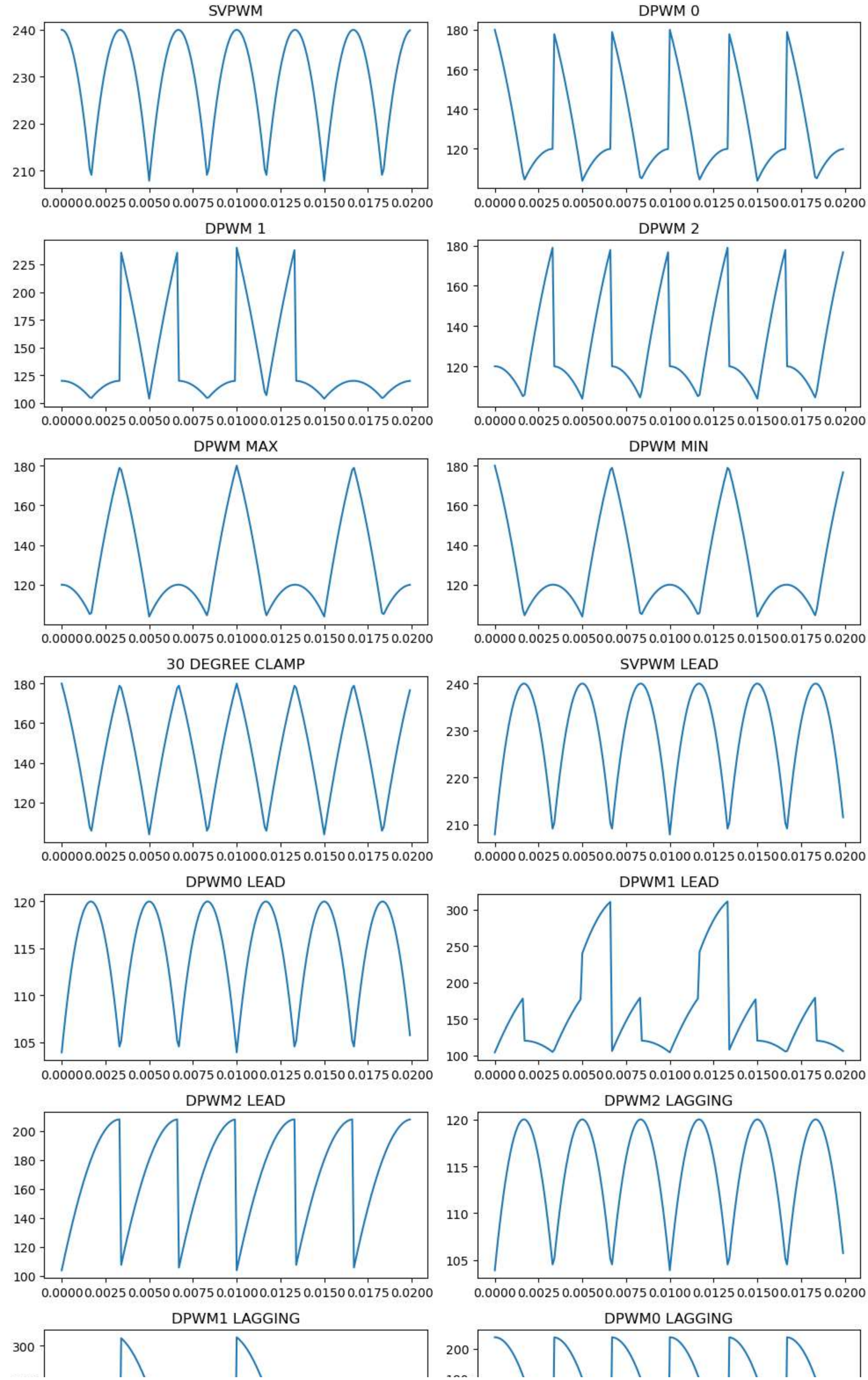
```
x16 = Time
y16 = Power
plt.show()
```



In [17]:

```
fig, axs = plt.subplots(8, 2,figsize=(10,20))



# Add plots to each of the subplots
axs[0, 0].plot(x1, y1)
axs[0, 1].plot(x2, y2)
axs[1, 0].plot(x3, y3)
axs[1, 1].plot(x4, y4)
axs[2, 0].plot(x5, y5)
axs[2, 1].plot(x6, y6)
axs[3, 0].plot(x7, y7)
axs[3, 1].plot(x8, y8)
axs[4, 0].plot(x9, y9)
axs[4, 1].plot(x10, y10)
axs[5, 0].plot(x11, y11)
axs[5, 1].plot(x12, y12)
axs[6, 0].plot(x13, y13)
axs[6, 1].plot(x14, y14)
axs[7, 0].plot(x15, y15)
axs[7, 1].plot(x16, y16)


# Set titles for each of the subplots
axs[0, 0].set_title('SVPWM')
axs[0, 1].set_title('DPWM 0')
axs[1, 0].set_title('DPWM 1')
```

```
axs[1, 1].set_title('DPWM 2')
axs[2, 0].set_title('DPWM MAX')
axs[2, 1].set_title('DPWM MIN')
axs[3, 0].set_title('30 DEGREE CLAMP')
axs[3, 1].set_title('SVPWM LEAD')
axs[4, 0].set_title('DPWM0 LEAD')
axs[4, 1].set_title('DPWM1 LEAD')
axs[5, 0].set_title('DPWM2 LEAD')
axs[5, 1].set_title('DPWM2 LAGGING')
axs[6, 0].set_title('DPWM1 LAGGING')
axs[6, 1].set_title('DPWM0 LAGGING')
axs[7, 0].set_title('DPWM MAX LAGGING')
axs[7, 1].set_title('DPWM MIN LAGGING')

# Set spacing between the subplots
fig.tight_layout()

# Show the figure
plt.show()
```



SVPWM

DPWM 0

DPWM 1

DPWM 2

DPWM MAX

DPWM MIN

30 DEGREE CLAMP

SVPWM LEAD

DPWM0 LEAD

DPWM1 LEAD

DPWM2 LEAD

DPWM2 LAGGING

DPWM1 LAGGING

DPWM0 LAGGING

In [21]:

```
# get handles of all subplots in the figure
subplots = fig.get_children()

# iterate over each subplot and calculate average and RMS values
results = []
for subplot in subplots:
    # check if subplot is an Axes object (i.e., a subplot)
    if isinstance(subplot, plt.Axes):
        # extract data from subplot
        data = subplot.lines[0].get_ydata()

        # calculate average and RMS values
        avg = np.mean(data)
        rms = np.sqrt(np.mean(np.square(data)))

        # store results in a dictionary
        results.append({'subplot': subplot, 'average': avg, 'rms': rms})

# print the results
for res in results:
    print(f" {res['subplot'].get_title()}: average = {res['average']:.2f}, RMS = {res[
```

```
 SVPWM: average = 229.18, RMS = 229.38
 DPWM 0: average = 130.24, RMS = 132.16
 DPWM 1: average = 135.26, RMS = 140.31
 DPWM 2: average = 129.64, RMS = 131.48
 DPWM MAX: average = 129.94, RMS = 131.82
 DPWM MIN: average = 129.94, RMS = 131.82
 30 DEGREE CLAMP: average = 145.29, RMS = 146.97
 SVPWM LEAD: average = 229.18, RMS = 229.38
 DPWM0 LEAD: average = 114.59, RMS = 114.69
 DPWM1 LEAD: average = 158.00, RMS = 169.11
 DPWM2 LEAD: average = 171.37, RMS = 174.27
 DPWM2 LAGGING: average = 114.59, RMS = 114.69
 DPWM1 LAGGING: average = 158.15, RMS = 169.22
 DPWM0 LAGGING: average = 172.41, RMS = 175.19
 DPWM MAX LAGGING: average = 143.50, RMS = 148.07
 DPWM MIN LAGGING: average = 143.50, RMS = 148.07
```

In [ ]: