CARACTERIZAÇÃO

In [29]:
```
import seaborn as sns, matplotlib.pyplot as plt, pandas as pd, locale, numpy as np
from matplotlib import ticker as mtick
from matplotlib.ticker import FormatStrFormatter
from math import ceil
```

In [2]:
```
locale.setlocale(locale.LC_ALL, 'pt_BR.UTF-8')
sns.set()
sns.set_style('whitegrid')
#plt.rcParams['figure.figsize'] = (8,5)
plt.rcParams['figure.figsize'] = (12,7)
plt.rcParams['font.family'] = 'Calibri'
plt.rc('font', size=14)
plt.rc('axes', labelsize=15)
plt.rc('xtick', labelsize=15)
plt.rc('ytick', labelsize=15)
plt.rc('legend', fontsize=15)
```

In [3]:
```
df = pd.read_excel('Gabriel Tkacz - P4-2.xlsx')
df.drop('Unnamed: 2', axis=1, inplace=True)
df.insert(2, 'Corrente (mA)', (1000 * df['Corrente (A)']))
df.insert(3, 'Potência (W)', (df['Voltagem (V)'] * df['Corrente (A)']))
df.insert(4, 'Potência (mW)', (1000 * df['Potência (W)']))
df
```

Out[3]:

	Voltagem (V)	Corrente (A)	Corrente (mA)	Potência (W)	Potência (mW)	Potencia Luminosa	Dimensão da placa
0	0.0	8.140000	8140.000000	0.000000	0.000000	1000 W/m2	2m x 1m
1	0.5	8.140000	8139.999985	4.070000	4069.999992	NaN	NaN
2	1.0	8.140000	8139.999967	8.140000	8139.999967	NaN	NaN
3	1.5	8.140000	8139.999945	12.210000	12209.999917	NaN	NaN
4	2.0	8.140000	8139.999918	16.280000	16279.999835	NaN	NaN
...	...	...	...	...	...	...	...
96	48.0	-0.810267	-810.267242	-38.892828	-38892.827638	NaN	NaN
97	48.5	-2.731293	-2731.292745	-132.467698	-132467.698120	NaN	NaN
98	49.0	-5.064634	-5064.634310	-248.167081	-248167.081188	NaN	NaN
99	49.5	-7.898789	-7898.788700	-390.990041	-390990.040630	NaN	NaN
100	50.0	-11.341247	-11341.247028	-567.062351	-567062.351405	NaN	NaN

101 rows × 7 columns

In [4]:
```
Plight = df['Potencia Luminosa'].values[0]
Plight = int(Plight.split()[0])

dimensao = df['Dimensão da placa'].values[0]
Acell = int(dimensao[0]) * int(dimensao[-2])

a = df['Corrente (A)'].values
idx = min(range(len(a)), key=lambda i: abs(a[i]))

Isc = round(df['Corrente (A)'].values[0], 1)
Voc = round(df['Voltagem (V)'].values[idx], 1)

df_max = df.loc[df['Potência (W)'].idxmax()]
Pmax = round((df_max['Potência (W)']), 1)
Imp = round((df_max['Corrente (A)']), 1)
Vmp = round((df_max['Voltagem (V)']), 1)
```

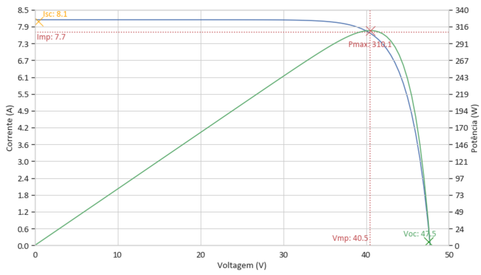
In [5]:
```
fig, ax1 = plt.subplots()
ax2 = ax1.twinx()

ax1.plot(df['Voltagem (V)'], df['Corrente (A)'])
ax2.plot(df['Voltagem (V)'], df['Potência (W)'], c='g')

ax1.plot(0.35, Isc, marker="x", markersize=15, markeredgecolor="orange")
ax1.annotate(f'Isc: {Isc}', (1, Isc+0.1), ha='left', va='bottom', c='orange')
ax1.plot(Voc, 0.15, marker="x", markersize=15, markeredgecolor="green")
ax1.annotate(f'Voc: {Voc}', (Voc-1, 0.4), ha='center', va='center', c='g')

ax2.plot(Vmp, Pmax, marker="x", markersize=15, markeredgecolor="r")
ax2.annotate(f'Pmax: {Pmax}', (Vmp, Pmax-20), ha='center', va='center', c='r')
ax1.axhline(y=Imp, color='r', linestyle=':')
ax1.annotate(f'Imp: {Imp}', (2, Imp-0.2), ha='center', va='center', c='r')
ax2.axvline(x=Vmp, color='r', linestyle=':')
ax2.annotate(f'Vmp: {Vmp}', (Vmp-0.25, 10), ha='right', va='center', c='r')

ax1.set_ylabel('Corrente (A)')
ax2.set_ylabel('Potência (W)')
ax1.set_xlabel('Voltagem (V)')
ax1.set_ylim([0, 8.5])
ax2.set_ylim([0, 340])
ax1.set_xlim([0, 50])
ax2.set_xlim([0, 50])
ax1.yaxis.set_major_locator(mtick.LinearLocator(15))
ax2.yaxis.set_major_locator(mtick.LinearLocator(15))
ax1.yaxis.set_major_formatter(FormatStrFormatter('%.1f'))
ax2.yaxis.set_major_formatter(FormatStrFormatter('%.0f'))
#ax1.autoscale(enable=True, axis='y')
#ax2.autoscale(enable=True, axis='y')
plt.savefig('CxVxW.png')
plt.show()
```



In [6]:
```
FF = round((Imp*Vmp)/(Isc*Voc) * 100, 1)
n = round((Pmax/Acell)/Plight * 100, 1)
```

In [14]:
```
print(f'Isc: {Isc}A. O Isc ou "Short-Circuit Current" é a corrente de curto-circuito, ou seja, a corrente quando a tensão é 0.')

print('\n')

print(f'Voc: {Voc}V. O Voc ou "Open-Circuit Voltage" é a tensão de circuito aberto, ou seja, a tensão quando a corrente é 0.')

print('\n')

print(f'Pmax: {Pmax}W. O Pmax é a potência máxima atingida pela célula solar.')

print('\n')

print(f'Imp: {Imp}A. O Imp é a corrente atingida em Pmax.')

print('\n')

print(f'Vmp: {Vmp}V. O Vmp é a tensão atingida em Pmax.')

print('\n')

print(f'FF: {FF}%. O FF ou Fator de Preenchimento é quanto uma célula solar se aproxima da idealidade.')

print('\n')

print(f'Eficiência: {n}%. A eficiência (η) é quantos porcento de luz solar é convertido em energia elétrica.')
```

```
Isc: 8.1A. O Isc ou "Short-Circuit Current" é a corrente de curto-circuito, ou seja, a corrente quando a tensão é 0.


Voc: 47.5V. O Voc ou "Open-Circuit Voltage" é a tensão de circuito aberto, ou seja, a tensão quando a corrente é 0.


Pmax: 310.1W. O Pmax é a potência máxima atingida pela célula solar.


Imp: 7.7A. O Imp é a corrente atingida em Pmax.


Vmp: 40.5V. O Vmp é a tensão atingida em Pmax.


FF: 81.1%. O FF ou Fator de Preenchimento é quanto uma célula solar se aproxima da idealidade.


Eficiência: 15.5%. A eficiência (η) é quantos porcento de luz solar é convertido em energia elétrica.
```

DIMENSIONAMENTO

In [68]:
```
coordenadas = (-9.725615619593865, -67.70347637374564)
irr_media = 4.56e3 #https://i.imgur.com/1PszqvJ.jpeg

casa = {
    'chuveiro':  {'quantidade': 1, 'consumo': 5000, 'uso': 0.5},
    'lampada':   {'quantidade': 2, 'consumo': 10,   'uso': 4},
    'TV':        {'quantidade': 1, 'consumo': 100,  'uso': 5},
    'geladeira': {'quantidade': 1, 'consumo': 70,   'uso': 24}
}
```

In [69]:
```
energia_1placa = (n/100) * irr_media * Acell

energia_casa = 0
for v in casa.values():
    valor = v['quantidade'] * v['consumo'] * v['uso']
    energia_casa += valor

qtd_placas_p = (energia_casa/energia_1placa)
qtd_placas = ceil(qtd_placas_p)
sobra = (qtd_placas - qtd_placas_p) * energia_1placa
sobra = round(sobra, 1)
```

In [70]:
```
capacidade_1bateria = 100 * 12
autonomia = 2 * energia_casa

qtd_baterias = ceil(autonomia/capacidade_1bateria)
```

In [87]:
```
print(f'A célula solar tem eficiência de {n}%, e cada placa solar gera {energia_1placa}kWh por dia.')
print(f'A casa consome {energia_casa}kWh por dia.')
print(f'Sendo assim, são necessárias {qtd_placas} placas para sustentar a casa.')

print('\n')

print(f'Para dois dias de autonomia, são necessários {autonomia}kWh. Cada bateria tem capacidade de {capacidade_1bateria}kWh.')
print(f'Sendo assim, são necessárias {qtd_baterias} baterias para dois dias de autonomia.')
print(f'Com as baterias carregando a {sobra}kWh (excesso de energia) por dia, são necessários cerca de {ceil(autonomia/(sobra*qtd_baterias))} dias de carregamento para ter 2 dias de autonomia.')
```

```
A célula solar tem eficiência de 15.5%, e cada placa solar gera 1413.6kWh por dia.
A casa consome 4760.0kWh por dia.
Sendo assim, são necessárias 4 placas para sustentar a casa.


Para dois dias de autonomia, são necessários 9520.0kWh. Cada bateria tem capacidade de 1200kWh.
Sendo assim, são necessárias 8 baterias para dois dias de autonomia.
Com as baterias carregando a 894.4kWh (excesso de energia) por dia, são necessários cerca de 2 dias de carregamento para ter 2 dias de autonomia.
```