In [2]:
```
import numpy as np
import scipy.sparse as sp
import scipy.sparse.linalg as spla
# import cupy as cp
# import cupyx.scipy.sparse as cpx_sp
# import cupyx.scipy.sparse.linalg as cpx_spla
import matplotlib.pyplot as plt
```

In [3]:
```
PI = 3.14159265358979323846264338327590288419716 9
wavelen = 1.55e-6
k = 2 * PI / wavelen
```

In [4]:
```
core_index = 2
cladding_index = 1.44
width, height = 2e-6, .15e-6  #waveguide dimensions
dx = 0.01e-6  #grid resolution

x = np.arange(-3e-6, 3e-6+dx, dx)
y = np.arange(-3e-6, 3e-6+dx, dx)
X, Y = np.meshgrid(x, y)
n = np.full(X.shape, cladding_index)
mask = (np.abs(X) <= width / 2) & (np.abs(Y) <= height / 2)
n[mask] = core_index
```

In [82]:
```
# Create laplacian matrix
Nx, Ny = len(x), len(y)
Nxy = Nx * Ny
diag = -4 * np.ones(Nxy)
off_diag = np.ones(Nxy - 1)
off_diag[np.arange(1, Nxy) % Nx == 0] = 0
laplacian = sp.diags([diag, off_diag, off_diag, np.ones(Nxy - Nx), np.ones(Nxy - Nx
n_squared = n.ravel() ** 2  #ravel and unravel mean the same thing. That is why irr
mass_matrix = sp.diags(k ** 2 * n_squared)

# Get eigenvector (TE? Electric field is perp to propegation - the |E| field matrix
A = laplacian / dx**2 + mass_matrix
```

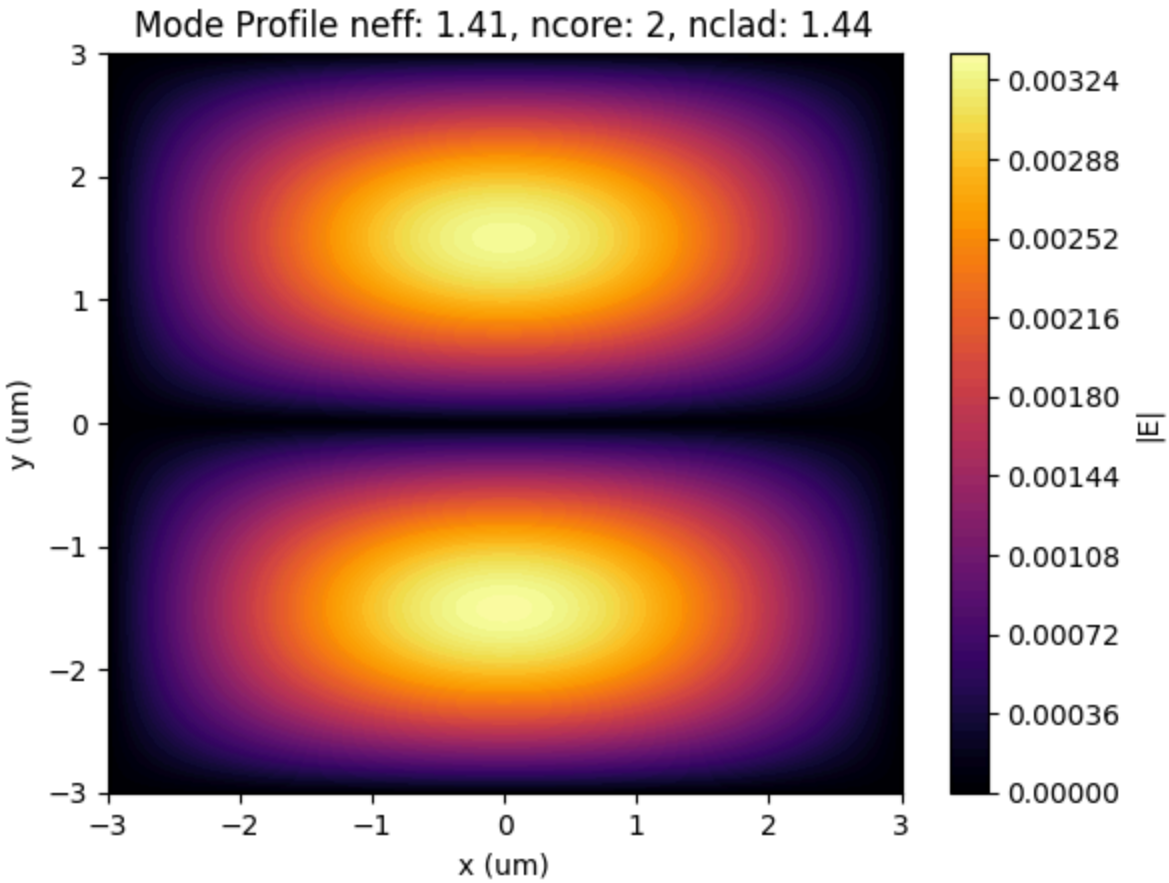
In [83]:
```
A_gpu = cpx_sp.csr_matrix(A)
eigvals_gpu, eigvecs_gpu = cpx_spla.eigsh(A_gpu, k=3, which="LA")
eigvals = cp.asnumpy(eigvals_gpu)
eigvecs = cp.asnumpy(eigvecs_gpu)
```

In [84]:
```
# eigvals, eigvecs = spla.eigs(A, k=3, which="LR")  # Solve for smallest eigenvalue
```

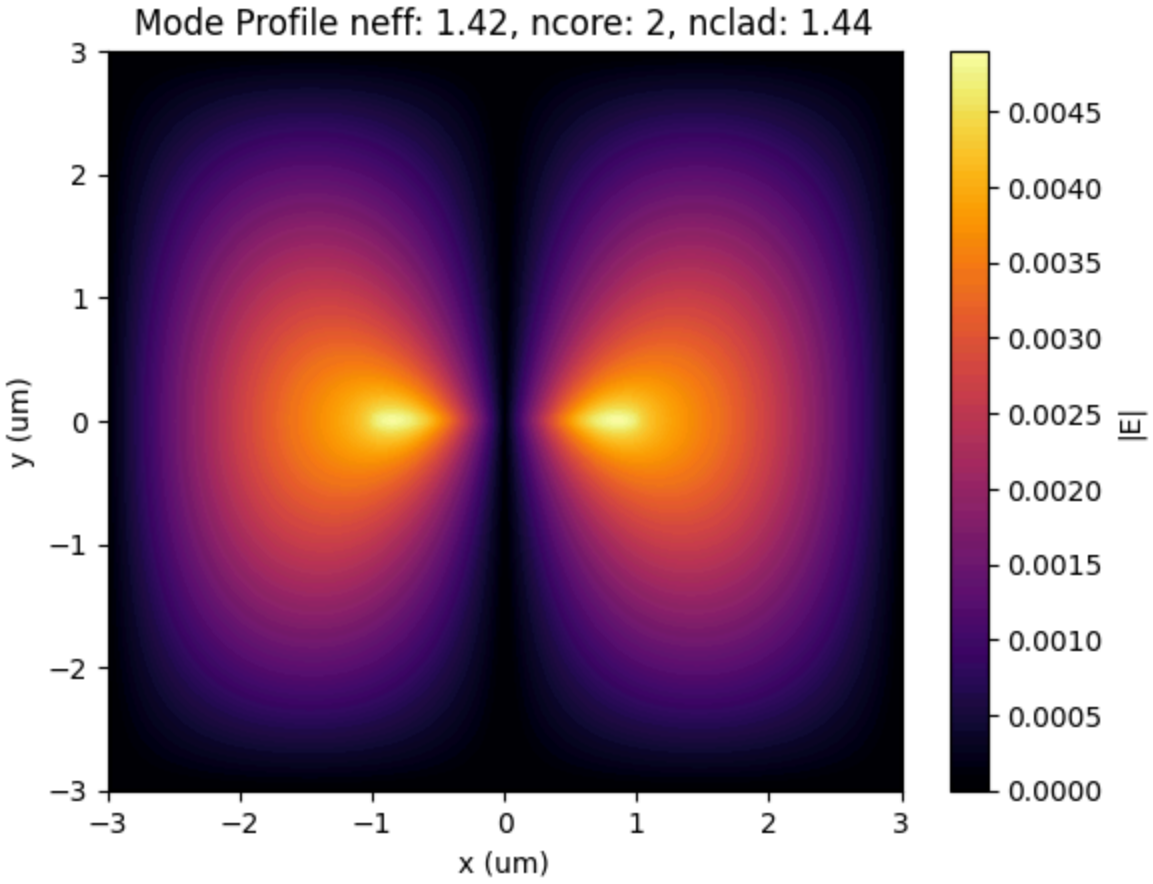
In [ ]:
```
print(eigvals)


for i in range(len(eigvals.real)):
    beta = np.sqrt(eigvals[i])
    print(f"neff: {beta/k}")
    print(f"ncore : {(core_index):e}")
    print(f"ncladding: {(cladding_index):e}")
```

```
    mode_profile = eigvecs[:, i].reshape(X.shape)
    np.savetxt(f"mode{i}.csv", mode_profile, delimiter=",")
    # np.savetxt(f"mode{i}_width{width}.csv", mode_profile, delimiter=",")
    plt.contourf(X * 1e6, Y * 1e6, np.abs(mode_profile), levels=100, cmap='inferno'
    plt.colorbar(label="|E|")
    plt.xlabel("x (um)")
    plt.ylabel("y (um)")
    plt.title(f"Mode Profile neff: {beta/k:.3g}, ncore: {(core_index):.3g}, nclad:
    plt.show()
    if cladding_index < beta/k < core_index:
        print("good solution")
    else:
        print("leaky or spurious mode")
```

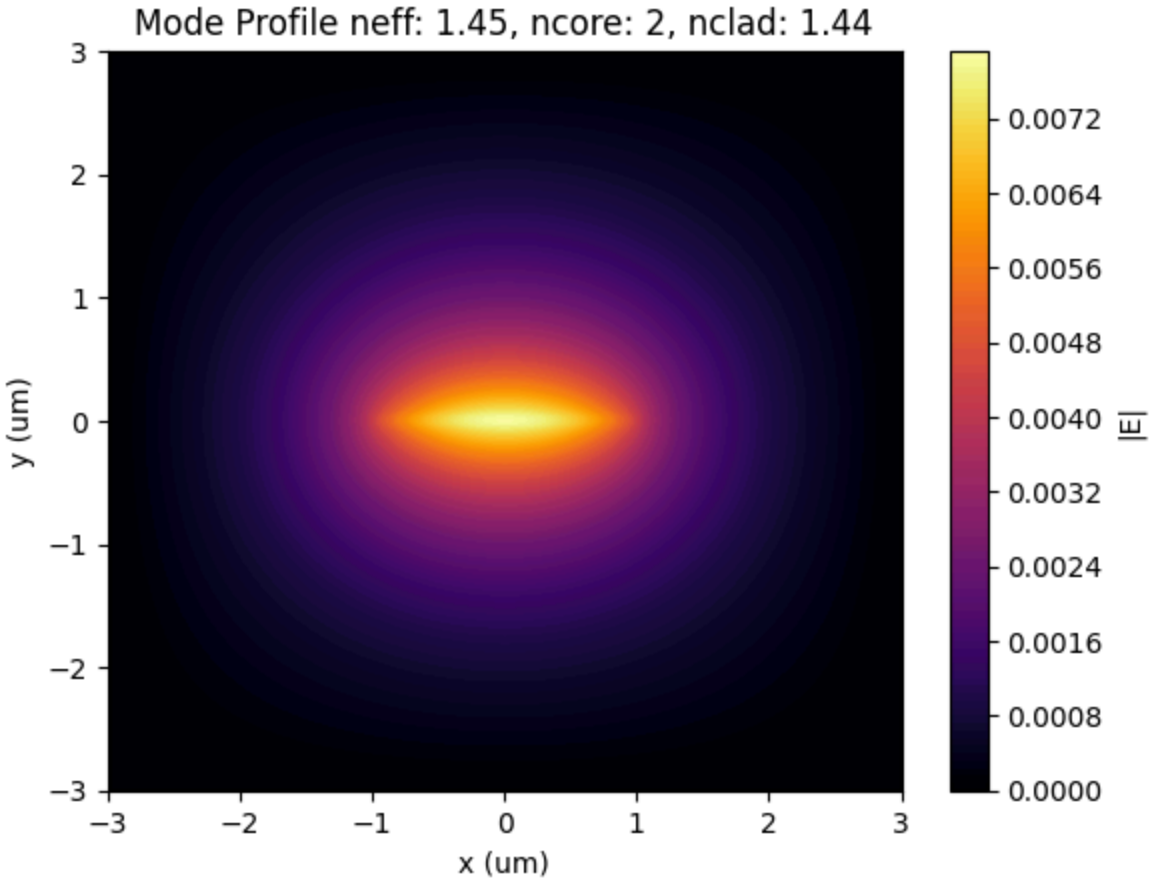
```
[3.27125045e+13 3.30458827e+13 3.47362491e+13]
neff: 1.4109405906361572
ncore : 2.000000e+00
ncladding: 1.440000e+00
```




```
leaky or spurious mode
neff: 1.4181119221401188
ncore : 2.000000e+00
ncladding: 1.440000e+00
```



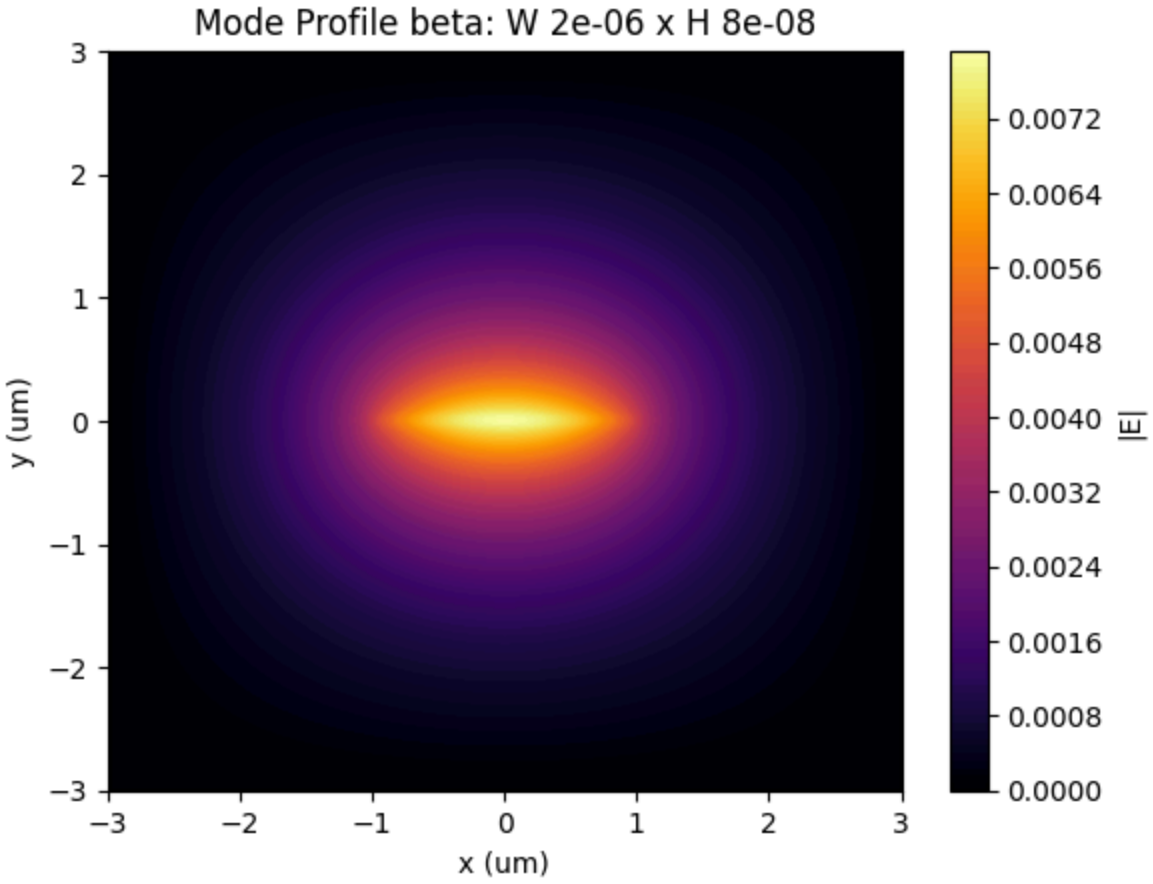

```
leaky or spurious mode
neff: 1.453929305454598
ncore : 2.000000e+00
ncladding: 1.440000e+00
```



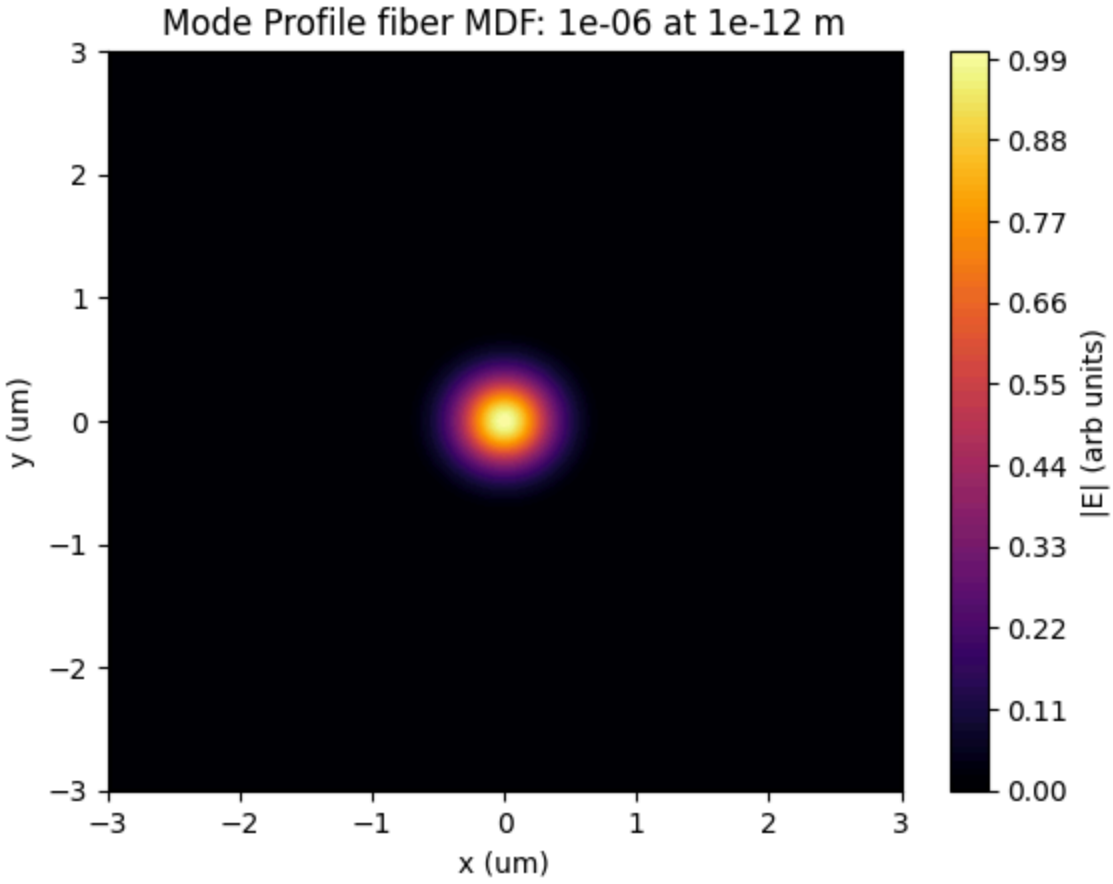
good solution

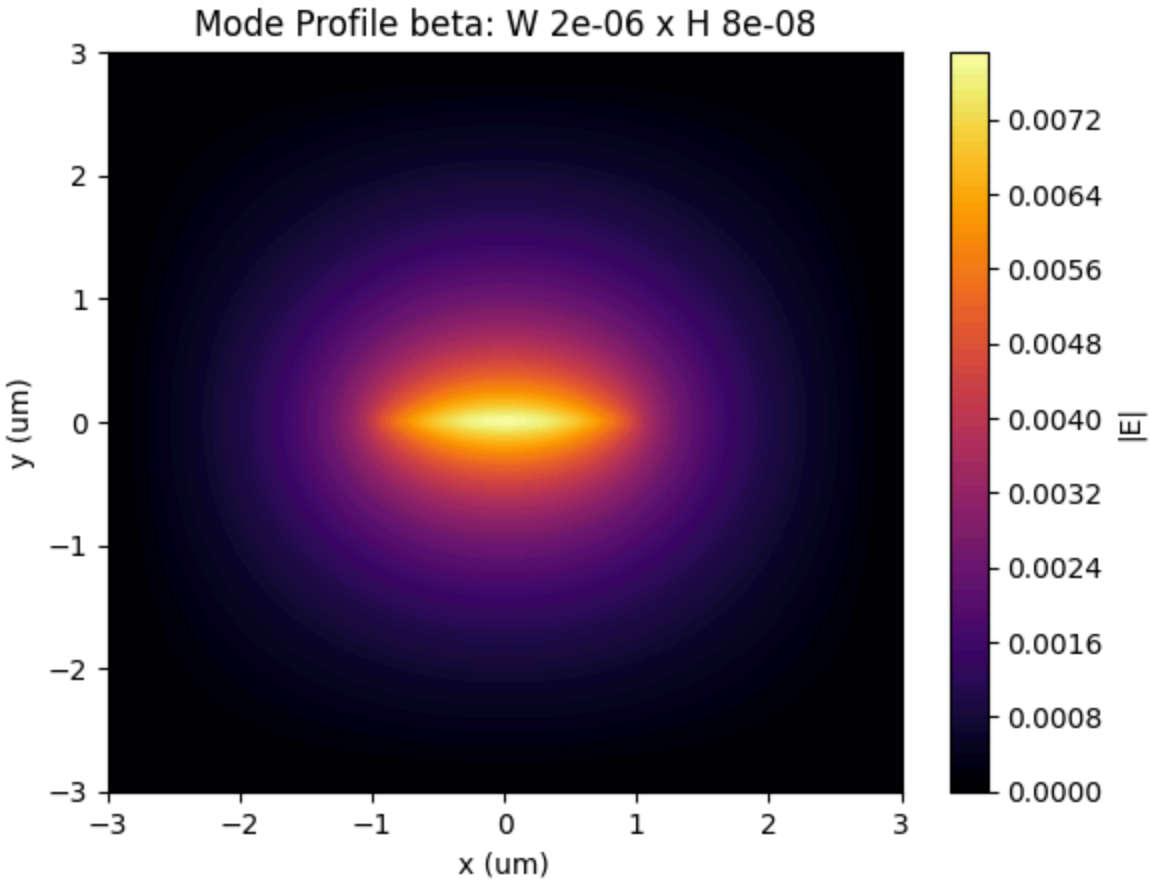
In [73]:
```
wg_mode = np.loadtxt("mode2_width2e-06.csv", delimiter=',')
plt.contourf(X * 1e6, Y * 1e6, np.abs(wg_mode), levels=100, cmap='inferno')
plt.colorbar(label="|E|")
plt.xlabel("x (um)")
plt.ylabel("y (um)")
plt.title(f"Mode Profile beta: W {width} x H {height}")
```

Out[73]: Text(0.5, 1.0, 'Mode Profile beta: W 2e-06 x H 8e-08')



In [74]:
```
x = np.arange(-3e-6, 3e-6+dx, dx)
y = np.arange(-3e-6, 3e-6+dx, dx)
X, Y = np.meshgrid(x, y)
wg_mode = np.pad(wg_mode, [(len(y) - wg_mode.shape[0]) // 2, (len(x) - wg_mode.shap

#Fiber MDF (hypothetical)
MDF = 1e-6

z = 1e-12
zR = PI/wavelen * (MDF / 2)**2
wz = MDF / 2 * np.sqrt(1 + (z / zR)**2)
fiber_mode = np.exp(-2*(X**2 + Y**2) / wz**2)
plt.contourf(X * 1e6, Y * 1e6, fiber_mode, levels=100, cmap='inferno')
plt.colorbar(label="|E| (arb units)")
plt.xlabel("x (um)")
plt.ylabel("y (um)")
plt.title(f"Mode Profile fiber MDF: {MDF} at {z} m")
plt.show()

plt.contourf(X * 1e6, Y * 1e6, np.abs(wg_mode), levels=100, cmap='inferno')
plt.colorbar(label="|E|")
plt.xlabel("x (um)")
plt.ylabel("y (um)")
plt.title(f"Mode Profile beta: W {width} x H {height}")
plt.show()
```

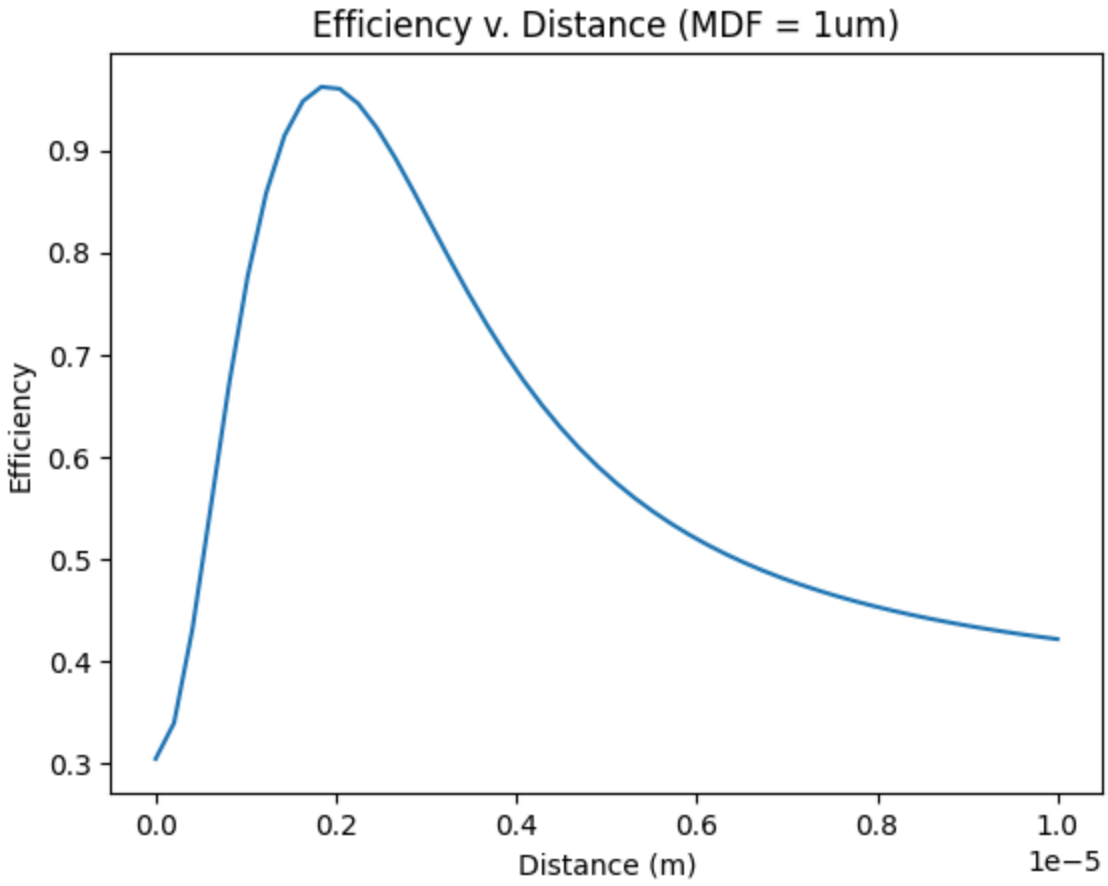

Mode Profile fiber MDF: 1e-06 at 1e-12 m
y (um)
x (um)
|E| (arb units)
3
2
1
0
−1
−2
−3
0.99
0.88
0.77
0.66
0.55
0.44
0.33
0.22
0.11
0.00


Mode Profile beta: W 2e-06 x H 8e-08
y (um)
x (um)
|E|
3
2
1
0
−1
−2
−3
0.0072
0.0064
0.0056
0.0048
0.0040
0.0032
0.0024
0.0016
0.0008
0.0000

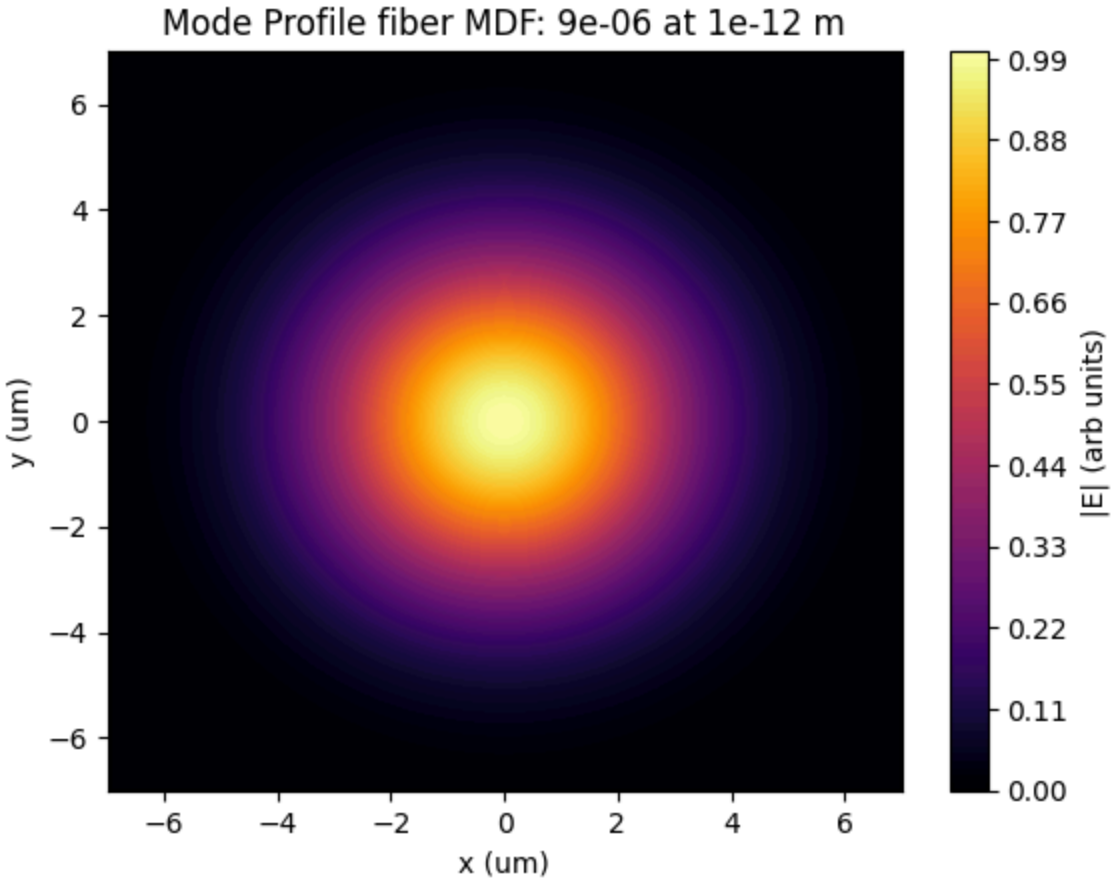
In [12]:
```
def cal_eff(psy1, psy2):
    overlap = np.sum(psy1 * psy2)
    norm1 = np.sum(np.abs(psy1)**2)
    norm2 = np.sum(np.abs(psy2)**2)
    return np.abs(overlap)**2 / (norm1 * norm2)
```

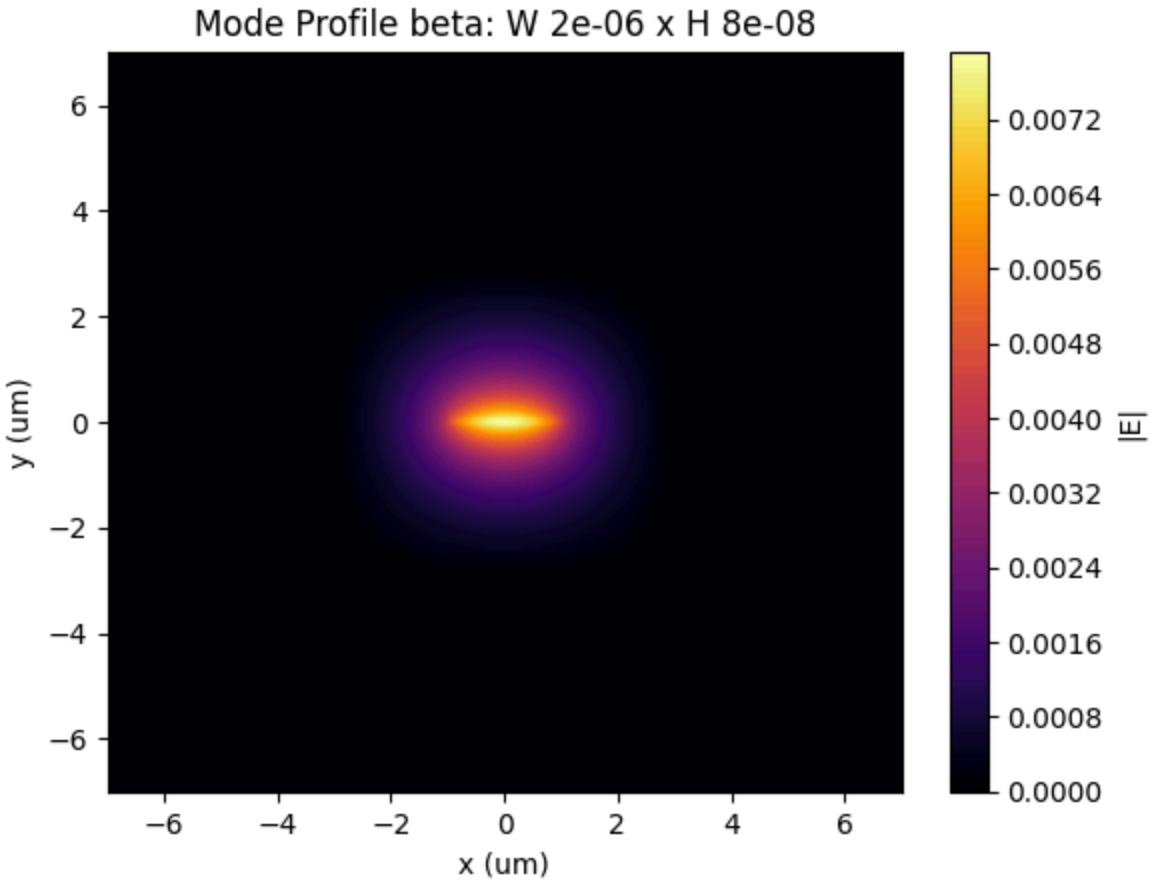
In [76]:
```
zs = np.linspace(0, 1e-5, 50)
efficieincies = []
for z in zs:
    zR = PI/wavelen * (MDF / 2)**2
    wz = MDF / 2 * np.sqrt(1 + (z / zR)**2)
    fiber_mode = np.exp(-2*(X**2 + Y**2) / wz**2)
    couple_eff = cal_eff(wg_mode, fiber_mode)
    efficieincies.append(couple_eff)

print(efficieincies)
plt.title("Efficiency v. Distance (MDF = 1um)")
plt.xlabel("Distance (m)")
plt.ylabel("Efficiency")
plt.plot(zs, efficieincies)
plt.show()
```

```
[0.30490536935501616, 0.33999219726914115, 0.43183857886554156, 0.5514989963333088,
0.6720869179650486, 0.7766584937330429, 0.8577896001489391, 0.9142286391251203, 0.94
78765482610294, 0.9619351766990862, 0.9599945487538352, 0.9456420467743963, 0.922274
911447815, 0.8929546250098529, 0.8602781825039209, 0.826300424488246, 0.792529451377
2134, 0.7599845952747213, 0.7292887500902754, 0.7007678898412175, 0.674540388325739
7, 0.6505886947255476, 0.6288125668649281, 0.6090663523446738, 0.5911838376031636,
0.5749940381947904, 0.5603306983321463, 0.5470375851450698, 0.5349710657673804, 0.52
4000989459553, 0.514010556709253, 0.5048956190149693, 0.496563691101106, 0.488932849
64143374, 0.48193062230563743, 0.4754929258864335, 0.4695630839047127, 0.46409093663
875345, 0.45903204610918324, 0.4543469926173913, 0.45000075631049036, 0.445962175848
6615, 0.44220347588659387, 0.4386998553090372, 0.4354291287052325, 0.43237141425816
7, 0.42950886196036936, 0.42682541678947766, 0.4243066121522395, 0.4219393895199337]
```



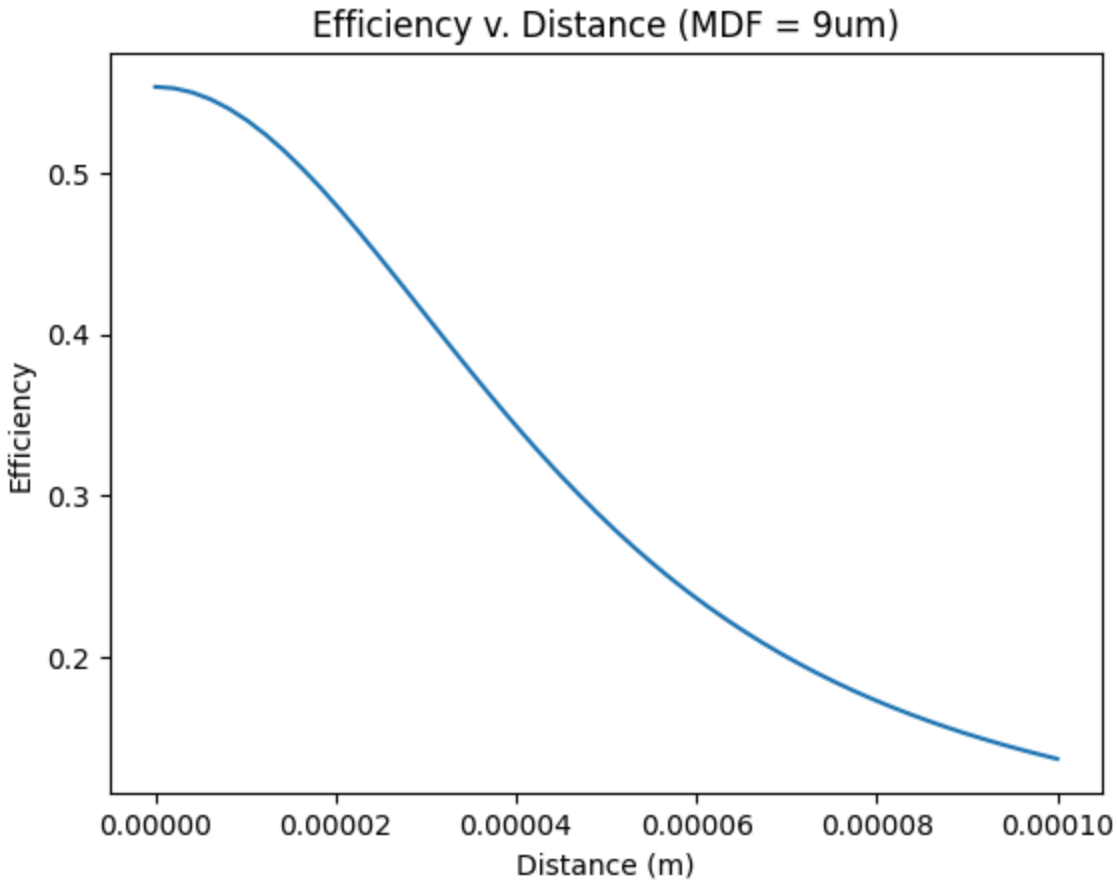
In [77]:
```
x = np.arange(-7e-6, 7e-6+dx, dx)
y = np.arange(-7e-6, 7e-6+dx, dx)
X, Y = np.meshgrid(x, y)
wg_mode = np.pad(wg_mode, [(len(y) - wg_mode.shape[0]) // 2, (len(x) - wg_mode.shap

#Fiber MDF (SM1550P)
MDF = 9e-6

z = 1e-12
zR = PI/wavelen * (MDF / 2)**2
wz = MDF / 2 * np.sqrt(1 + (z / zR)**2)
fiber_mode = np.exp(-2*(X**2 + Y**2) / wz**2)
plt.contourf(X * 1e6, Y * 1e6, fiber_mode, levels=100, cmap='inferno')
plt.colorbar(label="|E| (arb units)")
plt.xlabel("x (um)")
plt.ylabel("y (um)")
plt.title(f"Mode Profile fiber MDF: {MDF} at {z} m")
plt.show()

plt.contourf(X * 1e6, Y * 1e6, np.abs(wg_mode), levels=100, cmap='inferno')
plt.colorbar(label="|E|")
plt.xlabel("x (um)")
plt.ylabel("y (um)")
plt.title(f"Mode Profile beta: W {width} x H {height}")
plt.show()
```


Mode Profile fiber MDF: 9e-06 at 1e-12 m
y (um)
x (um)
|E| (arb units)
6
4
2
0
−2
−4
−6
0.99
0.88
0.77
0.66
0.55
0.44
0.33
0.22
0.11
0.00


Mode Profile beta: W 2e-06 x H 8e-08
y (um)
x (um)
|E|
6
4
2
0
−2
−4
−6
0.0072
0.0064
0.0056
0.0048
0.0040
0.0032
0.0024
0.0016
0.0008
0.0000

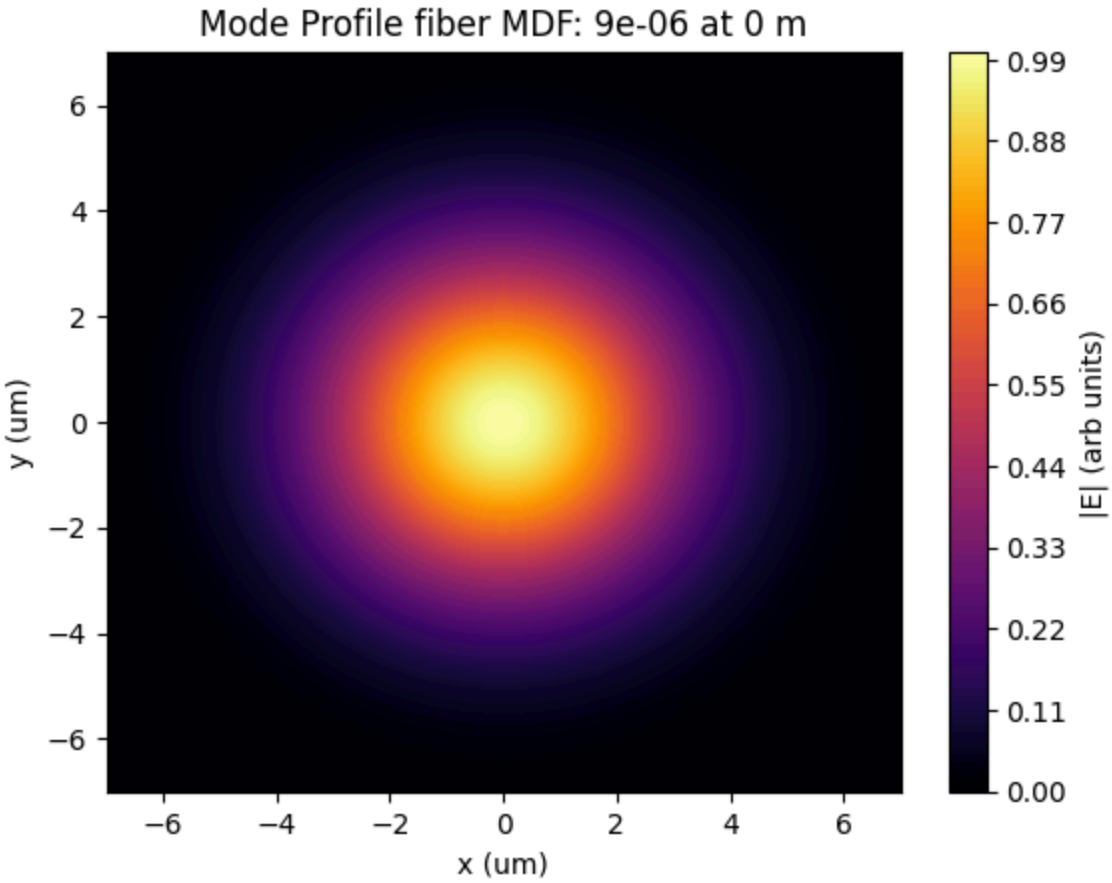
In [78]:
```
zs = np.linspace(0, 1e-4, 50)
efficieincies = []
for z in zs:
    zR = PI/wavelen * (MDF / 2)**2
    wz = MDF / 2 * np.sqrt(1 + (z / zR)**2)
    fiber_mode = np.exp(-2*(X**2 + Y**2) / wz**2)
    couple_eff = cal_eff(wg_mode, fiber_mode)
    efficieincies.append(couple_eff)

print(efficieincies)
plt.title("Efficiency v. Distance (MDF = 9um)")
plt.xlabel("Distance (m)")
plt.ylabel("Efficiency")
plt.plot(zs, efficieincies)
plt.show()
```

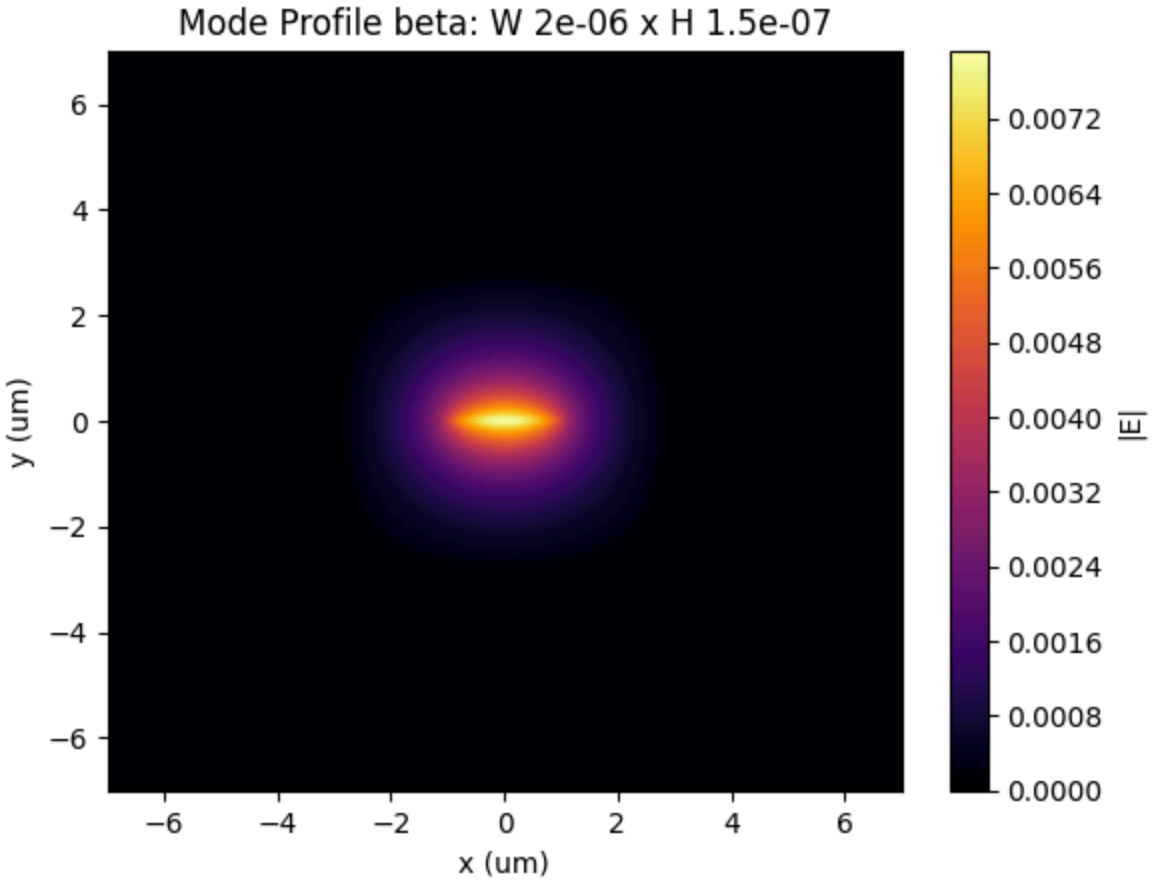
```
[0.55313253707858, 0.5522660041127793, 0.5496814780204325, 0.5454236182675077, 0.539
5650213512909, 0.5322035630836062, 0.5234588719897728, 0.5134681273662077, 0.5023814
203621431, 0.4903569430973261, 0.47755627260155364, 0.46413999127942795, 0.450263837
33479525, 0.43607551538753636, 0.42171222979846634, 0.40729894085126767, 0.392947294
14294693, 0.37875513989207854, 0.36480654146647257, 0.35117216874246177, 0.337909978
06281786, 0.32506609248443946, 0.3126758102435883, 0.30076468347693375, 0.2893496218
3700613, 0.27843998620373245, 0.26803864625474044, 0.2581429825317936, 0.24874581921
39531, 0.2398362784113768, 0.2314005506578538, 0.22342257953843478, 0.21588466110405
438, 0.20876796092429353, 0.20205295332909287, 0.19571978861423414, 0.18974859477228
61, 0.18411972071304356, 0.17881392801822166, 0.1738125381011734, 0.169097541279574
5, 0.16465167377743178, 0.16045846810441083, 0.1565022816576546, 0.1527683077867205,
0.14924257297980642, 0.14591192328515673, 0.14276400258513708, 0.1397872248967447,
0.1369707424823356]
```

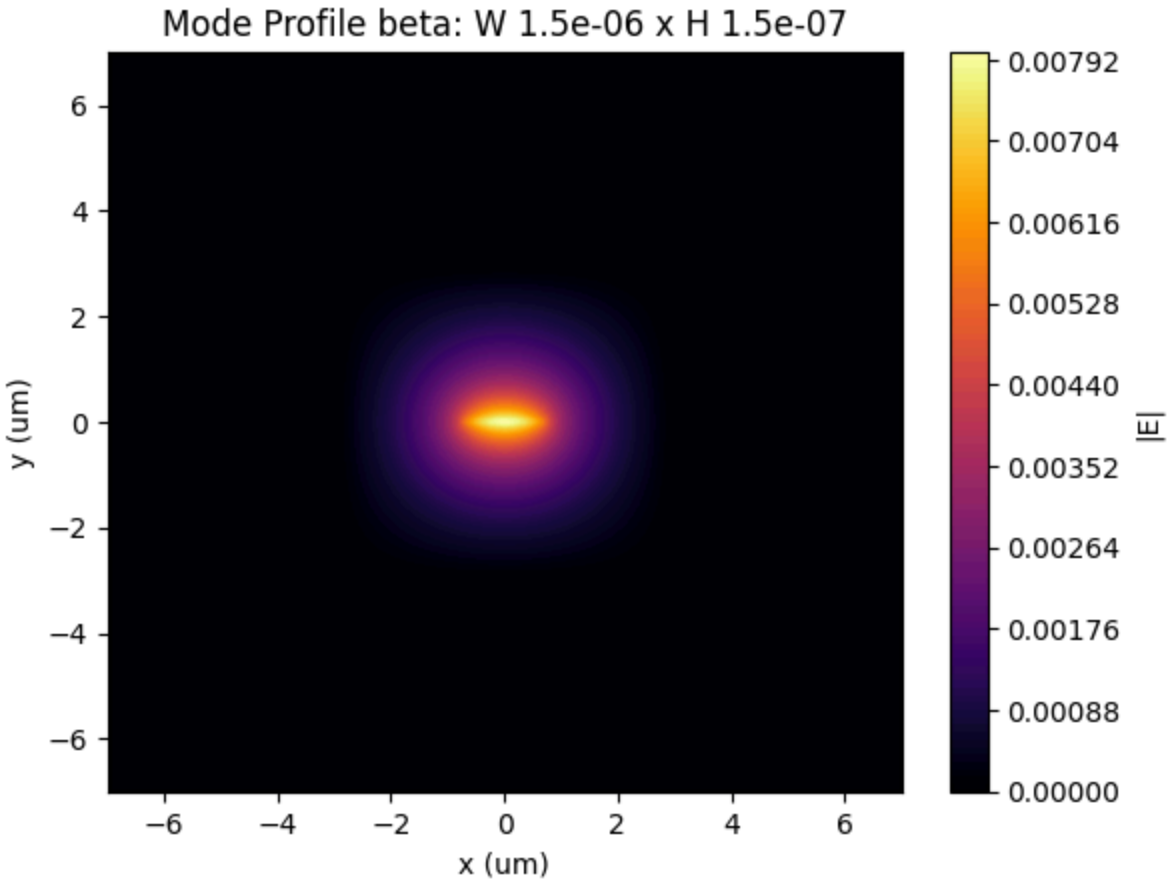


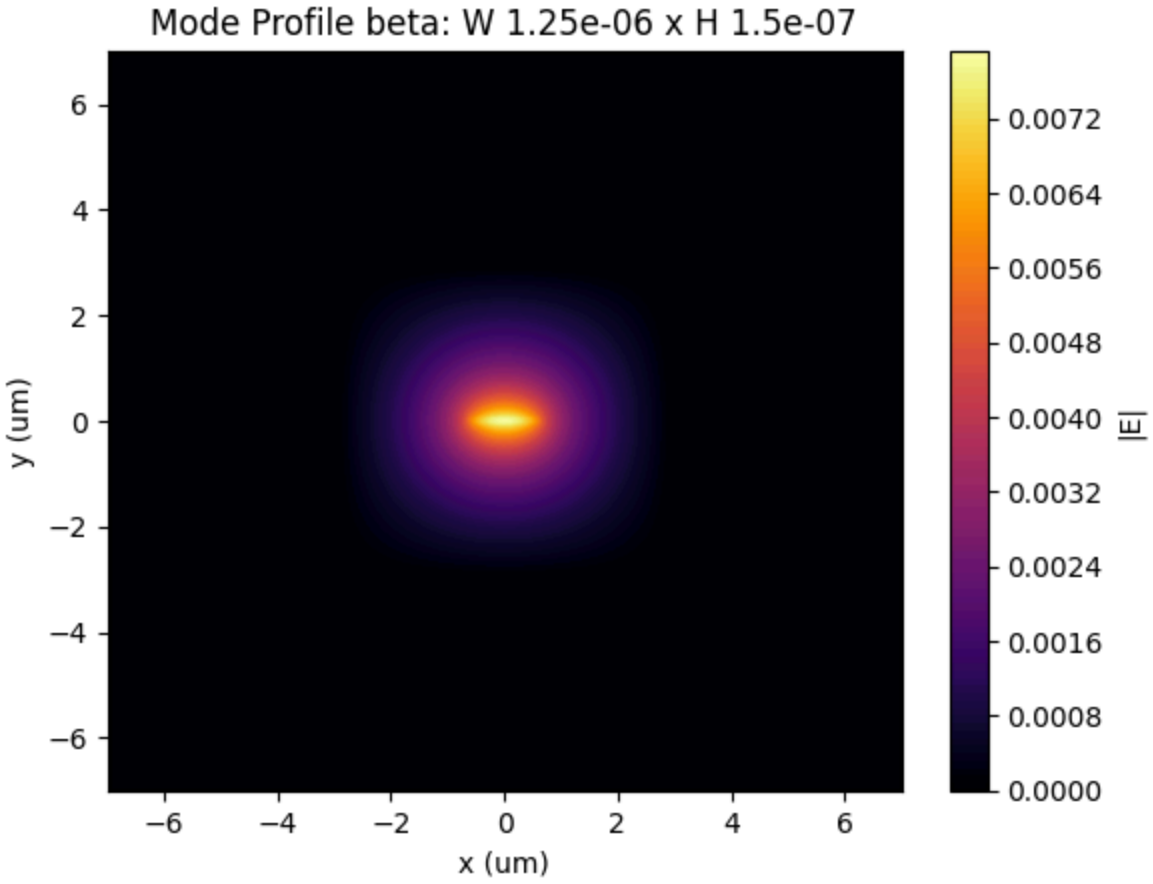
In [10]:
```
widths = [2e-6, 1.5e-6, 1.25e-6, 1e-6, 7.5e-7]
wg_modes = []
x = np.arange(-7e-6, 7e-6+dx, dx)
y = np.arange(-7e-6, 7e-6+dx, dx)
X, Y = np.meshgrid(x, y)
#Fiber MDF (SM1550P)
MDF = 9e-6

z = 0
zR = PI/wavelen * (MDF / 2)**2
wz = MDF / 2 * np.sqrt(1 + (z / zR)**2)
fiber_mode = np.exp(-2*(X**2 + Y**2) / wz**2)
plt.contourf(X * 1e6, Y * 1e6, fiber_mode, levels=100, cmap='inferno')
plt.colorbar(label="|E| (arb units)")
plt.xlabel("x (um)")
plt.ylabel("y (um)")
plt.title(f"Mode Profile fiber MDF: {MDF} at {z} m")
plt.show()
for w in widths:
    filename = f"mode0_width{w}.csv"
    wg_mode = np.loadtxt(filename, delimiter=',')
    wg_mode = np.pad(wg_mode, [(len(y) - wg_mode.shape[0]) // 2, (len(x) - wg_mode.
    wg_modes.append(wg_mode)
    plt.contourf(X * 1e6, Y * 1e6, np.abs(wg_mode), levels=100, cmap='inferno')
    plt.colorbar(label="|E|")
    plt.xlabel("x (um)")
    plt.ylabel("y (um)")
```

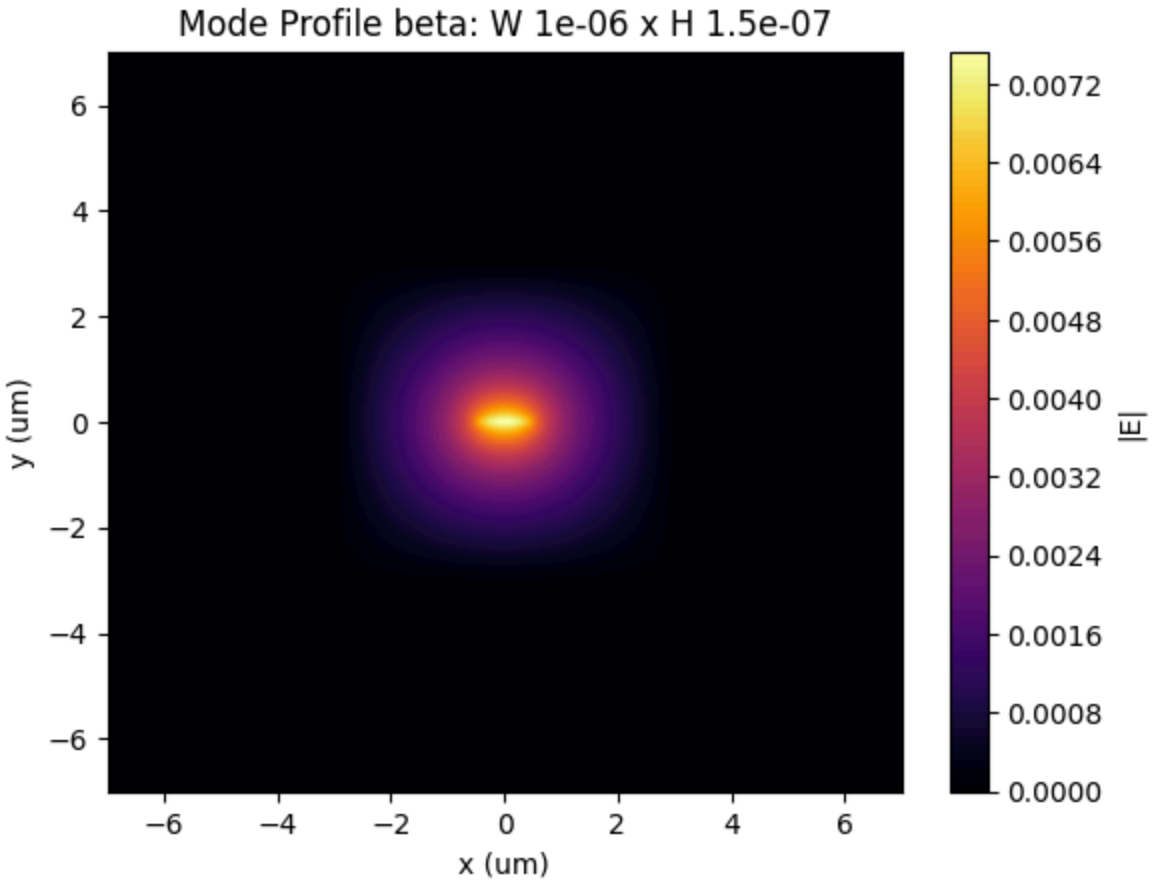
```
    plt.title(f"Mode Profile beta: W {w} x H {height}")
    plt.show()
```

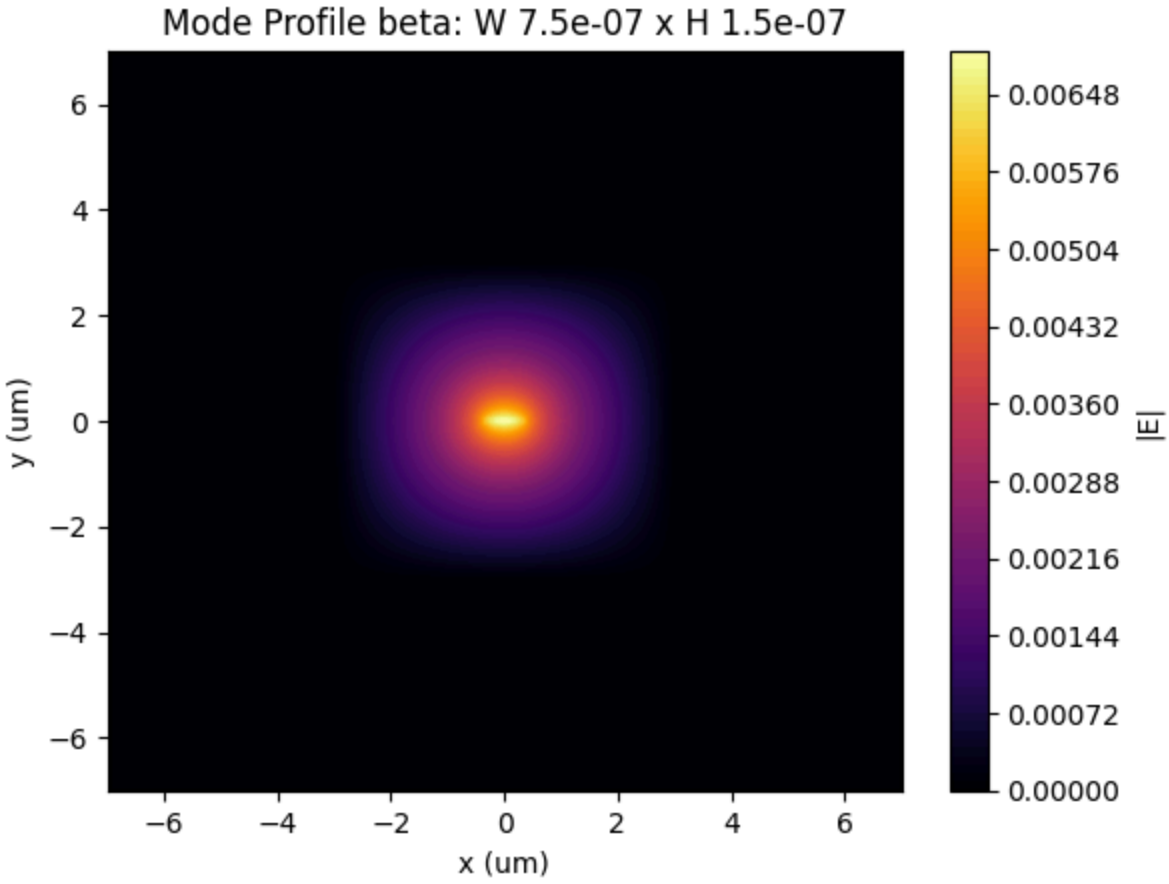



Mode Profile beta: W 2e-06 x H 1.5e-07
y (um)
x (um)
|E|
6
4
2
0
−2
−4
−6
0.0072
0.0064
0.0056
0.0048
0.0040
0.0032
0.0024
0.0016
0.0008
0.0000

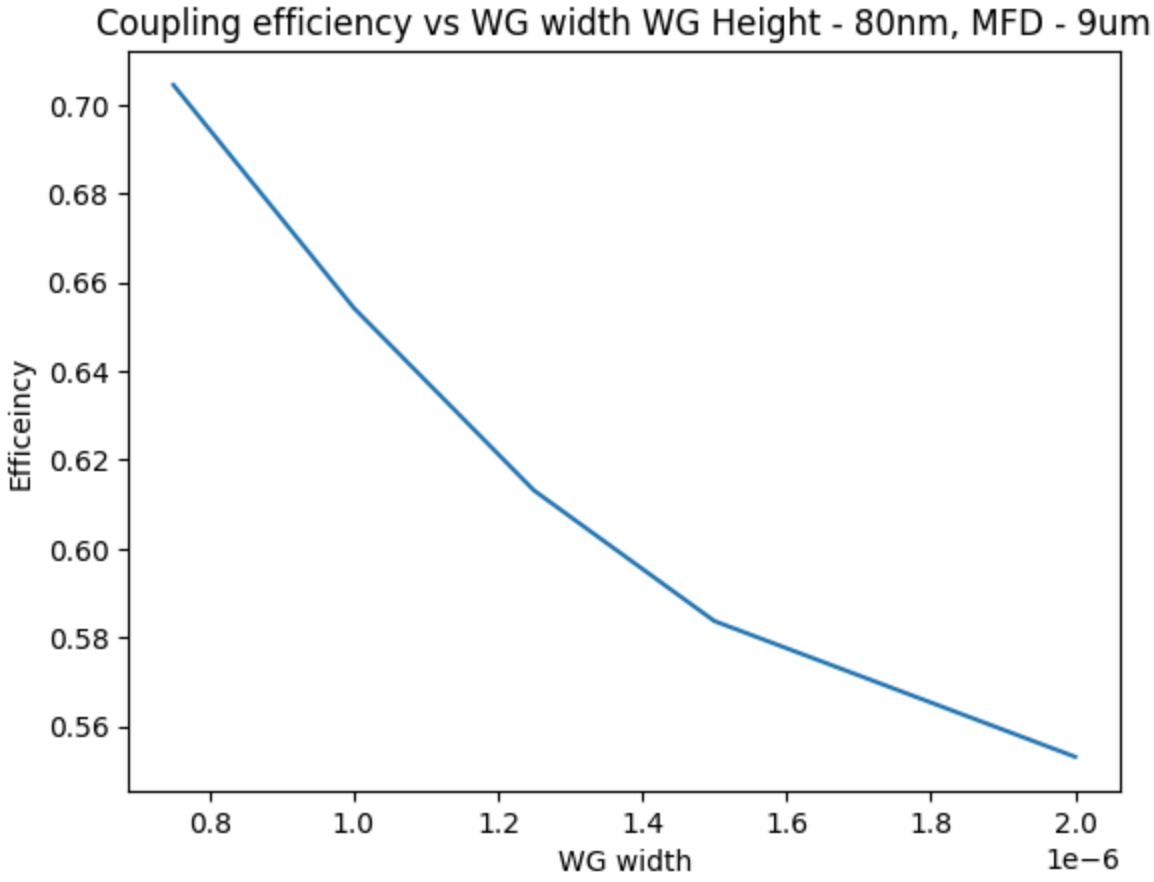

Mode Profile beta: W 1.5e-06 x H 1.5e-07
y (um)
x (um)
|E|
6
4
2
0
−2
−4
−6
0.00792
0.00704
0.00616
0.00528
0.00440
0.00352
0.00264
0.00176
0.00088
0.00000


Mode Profile beta: W 1.25e-06 x H 1.5e-07
y (um)
x (um)
|E|
6
4
2
0
−2
−4
−6
0.0072
0.0064
0.0056
0.0048
0.0040
0.0032
0.0024
0.0016
0.0008
0.0000


Mode Profile beta: W 1e-06 x H 1.5e-07
y (um)
x (um)
|E|
6
4
2
0
−2
−4
−6
0.0072
0.0064
0.0056
0.0048
0.0040
0.0032
0.0024
0.0016
0.0008
0.0000



In [14]:
```
effs = []
for mode in wg_modes:
    effs.append(cal_eff(mode, fiber_mode))

plt.title("Coupling efficiency vs WG width WG Height - 80nm, MFD - 9um")
plt.plot(widths, effs)
plt.ylabel("Efficeincy")
plt.xlabel("WG width")
plt.show()
```


Coupling efficiency vs WG width WG Height - 80nm, MFD - 9um
Efficeincy
0.70
0.68
0.66
0.64
0.62
0.60
0.58
0.56
0.8
1.0
1.2
1.4
1.6
1.8
2.0
WG width
1e−6