In this example, we will present a sample code for plotting the pole-zero plot of a discrete-time system. The system function is given in Ornek-10, i.e.

$$H(z) = \frac{1+z^{-1}}{(1+j\frac{1}{2}z^{-1})(1-j\frac{1}{2}z^{-1})(1+\frac{1}{4}z^{-1})}$$

First, we need to define the system. Since it is given in its pole-zero form, we must convert it into the transfer function form. The reason is that the function that plots the pole-zero plot **only** accepts the transfer function coefficients as its arguments. Python has no built-in "zplane" function, therefore a custom *zplane()* function was given to you as part of the prelab files.

Note that if the system function is given in other forms, i.e. pole-zero form, **you should first convert it to the transfer function form**.

In [2]:
```
# import the necessary libraries
import numpy as np             # for using basic array functions
import matplotlib.pyplot as plt # for this example, it may not be necessary

# the main package for signal processing is called "scipy" and we will use "signal"
import scipy.signal as sgnl
# alternative syntax: from scipy import signal as sgnl
%matplotlib notebook

# WE NEED TO IMPORT THE CUSTOM (USER DEFINED) FUNCTION AS WELL, IN ORDER TO USE IT!!
import zplane
```

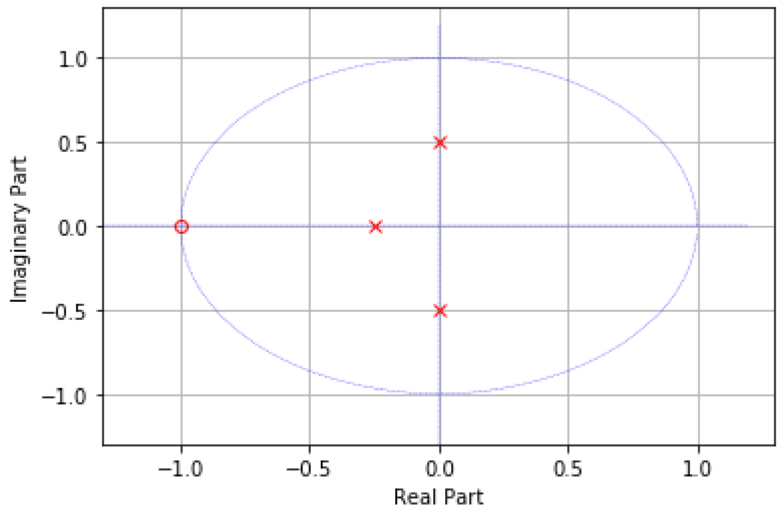
The above cell contains usual libraries we use in this course.

Note that in the last line, we have imported an additional function, which is not part of any of the built-in python libraries. Without importing this function **we will not be able to call the function**

In [3]:
```
zeross = np.array([-1])                                    # the system has a single ze
poless = np.array([-1j/2, 1j/2, -1/4])                     # the system has three poles
k = 1                                                      # the system has unity gain

b, a = sgnl.zpk2tf(zeross,poless,k)                        # call the function that con

zplane.zplane(b,a)                                         # call the function with coe
```



Note how we call the user-defined function as *zplane.zplane()*

Note that the zplane() function gives no information on the *ROC* of the system, i.e. does not show ROC on the plot, therefore we must manually specify the ROC, if necessary.

Z-Transform -> Frequency Response

Now, in this part of the example we will discuss the Z-transform and frequency response of the LTI systems.

Recall that, the existence of a Fourier transform indicates the stability of the system, i.e. the impulse response of the system must be absolutely summable in order for the F.T. to exist.

Under this constraint, the F.T. of a system can be obtained directly from its Z-transform by inserting $z \to re^{j\omega}$, with r=1.

Suppose that we are given a system with

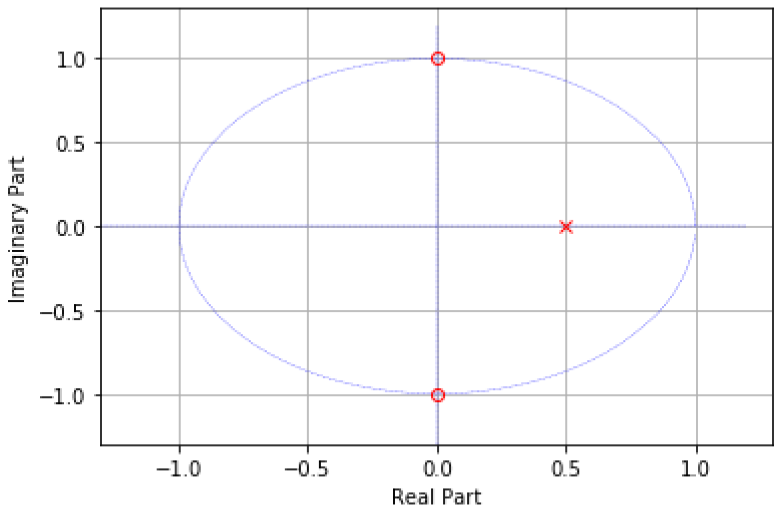
$$H(z) = \frac{(1+jz^{-1})(1-jz^{-1})}{1-\frac{1}{2}z^{-1}} = \frac{1+z^{-2}}{1-\frac{1}{2}z^{-1}}, ROC: |z| > \frac{1}{2}$$

We will plot the pole-zero diagram for this system, then find its F.T. by substituting $z \to e^{j\omega}$. Then we investigate the effect of poles and zeros on the frequency response (magnitude response, specifically).

```
In [4]:
num = np.array([1, 0, 1])          # note that the coeff of z^(-1) term is zero.
denum = np.array([1, -1.0/2])


plt.figure()
zplane(num, denum)
```

```
<Figure size 432x288 with 0 Axes>
```

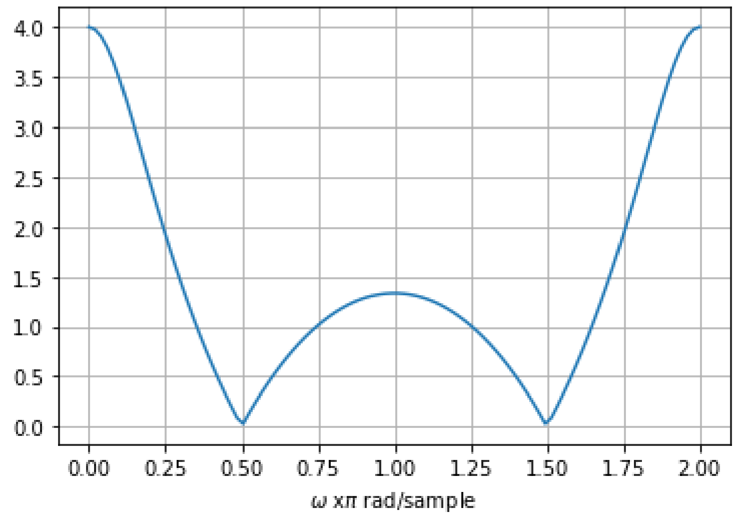


```
In [5]:
# Now, we define the frequency response as:
w = np.linspace(0, 2*np.pi, 100)

Hw = (1+np.exp(-2*1j*w))/(1-(1/2)*np.exp(-1j*w))
# alternatively, we could use the sgnl.freqz_zpk() function to calculate the Frequen
# or sgnl.freqz() function to calculate from the coefficients.

Hw_abs = abs(Hw)
plt.figure()
plt.plot(w/np.pi, Hw_abs)
plt.grid()
plt.xlabel('$\omega$ x$\pi$ rad/sample')
```

Out[5]:
```
Text(0.5, 0, '$\\omega$ x$\\pi$ rad/sample')
```



Now, there is a correspondence between the poles/zeros of the system and its frequency response. Recall that, by definition, "a pole" of the system is a root of its denominator term. Therefore, a pole makes the magnitude of the system go higher (notice the dividing by zero). Similarly, "a zero" is the root of the numerator term, thus makes the magnitude go to zero.

We will make inference by establishing correspondence betweem the pole-zero graph and the frequency response.

On the pole-zero graph, we trace the unit circle starting from $z = 0$ counter-clockwise. At the starting point, $z = 0$, the magnitude is 4. As we travel on the unit circle towards the zero at $z = j$, the magnitude decreases. At the system zero, $z = j$, the frequency response is zero, as expected. In the polar notation, $z = j = e^{j\pi/2}$, therefore, the angle of the zero from the positive real axis is the cut-off frequency of the system. Then, as we pass the system zero and move along, the magnitude increases again. However, this increase in the magnitude is not as much as the initial magnitude. This is because it is related to the distance total distance of the poles and zeros to a point on the unit circle. **Imagine the zeros trying to pull the magnitude lower, at the same time the poles trying to push it upwards.** As we travel rom $\omega = \pi/2$ to $\omega = 3\pi/2$, the effect of zeros surpasses the pole's. Then, the magnitude drops to zero again at $z = -j = e^{j3\pi/2}$. As we complete the circle towards the starting point, once again the pole prevails over the zeros and pushes the magnitude to 4.

To summarize, the zeros try to supress the magnitude response, while the poles try to amplify it.

In [ ]: