




Code Explanation: Speaker with Crickit (Door Open Sound)


```
import audiocore
```

 Imports the audiocore module.


 Used to handle .wav audio files and prepare them for playback through the Crickit speaker.


```
import audioio
```

 Imports the audioio module.


 Controls the audio output system in CircuitPython, allowing playback of sound files through Crickit's built-in amplifier.


```
import board
```

 Imports the board module.

 Allows CircuitPython to recognize and communicate with the **Crickit speaker pins** connected to the CPX board.


```
speaker = audioio.AudioOut(board.A0)
```

 Creates a **speaker output object** using **A0**, the audio connection between CPX and Crickit.

 The Crickit board uses this pin to receive audio signals from the CPX and amplify them to the speaker terminals (**SPEAKER +** and **SPEAKER -**).


```
with open("door_open.wav", "rb") as f:
```

 Opens the file **door_open.wav** in **read-binary mode (rb)**.

 The .wav sound file must be stored in the **root folder of the CPX drive** before running this code.

```
wave = audiocore.WaveFile(f)
```

 Converts the opened audio file into a **WaveFile** format that can be played by the Crickit's audio system.

 This prepares the sound data for playback.

```
speaker.play(wave)
```

👉 Starts playing the .wav file through the **Crickit speaker terminals**.

🔊 The Crickit's internal amplifier boosts the sound, so it plays louder and clearer.

```
while speaker.playing:
```

```
    pass
```

👉 Keeps the program running **until the audio playback is finished**.

🕒 Prevents the program from ending before the sound completes.