Application Report
SLAA891B–April 2019–Revised February 2020

Automating Capacitive Touch Sensor PCB Design Using OpenSCAD Scripts

Dennis Lehman

ABSTRACT

Products with capacitive touch sensor interfaces are more common than ever and typically feature a few button sensors or maybe a collection of buttons, such as a keypad. The button sensors provide a simple "touch" or "no touch" user input and their typical round or square shape can be created easily with a PCB CAD design tool. As product interfaces evolve, interfaces are beginning to include more complex sensor types, such as a slider, wheel, or touchpad. Designing these types of capacitive touch sensors shapes with a PCB CAD design tool can require a considerable amount of time, even for the best engineers and PCB designers. And even after the design has been completed, changes in product requirements often force the designer to make modifications to the design or completely start the design over. Automating the design steps can all but eliminate this time consuming process.

This application report describes how to create scalable self-capacitive slider, wheel, and touchpad sensors with OpenSCAD, a free Solid 3D CAD modeling tool, and custom scripts created by Texas Instruments. Modifying the script parameters and running the script generates a new sensor pattern that can be exported to DXF format then easily imported into any PCB CAD tool. This flexibility and speed makes it now possible to create complex sensor shapes effortlessly.

The scripts described in this document can be downloaded from http://www.ti.com/lit/zip/slaa891.

Contents

List of Figures

Trademarks

1 Introduction

OpenSCAD is a free software application for creating solid 3D CAD objects. It is a script-only based modeler that uses its own description language; parts can be previewed but cannot be interactively selected or modified by the mouse in the 3D view. OpenSCAD is free and released under the General Public License version 2. OpenSCAD is available from http://www.openscad.org/. It is also available for macOS® and Linux® operating systems.

The version at the time of this writing is 2019-05. Although the primary strength of OpenSCAD is 3D rendering, its 2D capabilities are perfect for creating 2D capacitive electrode shapes that can be quickly manipulated by simple modifications to the script parameters that control the various attributes of the design.

The sensor design scripts are provided by Texas Instruments under a BSD license so you are free to modify them if needed. These scripts create and manipulate primitive objects that are assembled into complex slider and wheel sensor designs, using only these seven OpenSCAD commands:

square()
circle()
translate()
rotate()
hull()
difference()
intersection()

For information about these commands, refer to the OpenSCAD user guide.

This document does not cover the theory of capacitive touch or provide design guidelines for construction and layout of capacitive sensors on a PCB, but it does illustrate common slider, wheel and, touchpad sensor designs and their geometric features that relate to the corresponding script parameters.

It is assumed the reader has some familiarity with capacitive touch technology, slider or wheel sensors and sensor design principles. If new to capacitive touch, visit http://www.ti.com/captivate for information about MSP430™ capacitive touch sensing microcontrollers and http://www.ti.com/captivatetechguide for sensor design guidelines.

NOTE: The PCB examples provided in this document are demonstrated using Altium Designer 19. Refer to your PCB CAD tool documentation to perform the equivalent steps.

2 Getting Started

OpenSCAD.exe and the OpenSCAD scripts are supported in Windows®, macOS®, and Linux® operating systems. The optional batch files provided with this application report were written and tested on Windows 10. If using macOS or Linux, it is up to the user to create a similar shell script for that platform.

2.1 Prerequisites

The following steps assume that the Windows user has admin rights and can do the following on the PC:

- Download and install OpenSCAD (may require administrative rights).
- If using the optional batch files, modify the path environment variable.

Preparations:

1. Download and install OpenSCAD from http://www.openscad.org/.

   On Windows OS, the installer may need to be executed with administrator rights by right mouse clicking on the .exe and selecting "Run as Administrator."

   The default installation directory is `C:\Program Files\OpenSCAD`.
2. Create a project directory **C:\OpenSCAD\Projects** and download the sensor script zip file into this directory. Unzip the contents.
3. If using the optional batch files, modify the path environment variable by appending the path `C:\Program Files\OpenSCAD` (see the following steps). Modifying the path environment variable is required to run the batch files without providing a path to OpenSCAD.exe on the command line.

To modify the environment path in Windows 10 (optional):

1. Open System Properties (right click Computer in the start menu, or use the keyboard shortcut Winkey+Pause)
2. Click Advanced system settings in the sidebar.
3. Click Environmental Variables.
4. From the system variables, select "PATH" and click Edit.
5. Add the OpenSCAD installation directory path **C:\Program Files\OpenSCAD** to the end of the path list.

2.2 Workflow

Slider, wheel, and touchpad sensors have geometric features of length, width, radii, angles, and clearances that must be considered when constructing a sensor. The reader is responsible to understand their sensor specifications before using these scripts, or the resulting output may not provide the optimal performance for an application. If you are not familiar with the construction of these various sensors, illustrations of the script parameters are shown starting in Section 3.

The process or workflow for creating a sensor requires three steps (see Figure 1):

1. Modify the script and render (F6) the sensor.
2. Export to DXF.
3. Import into a PCB CAD tool and convert to a copper region.

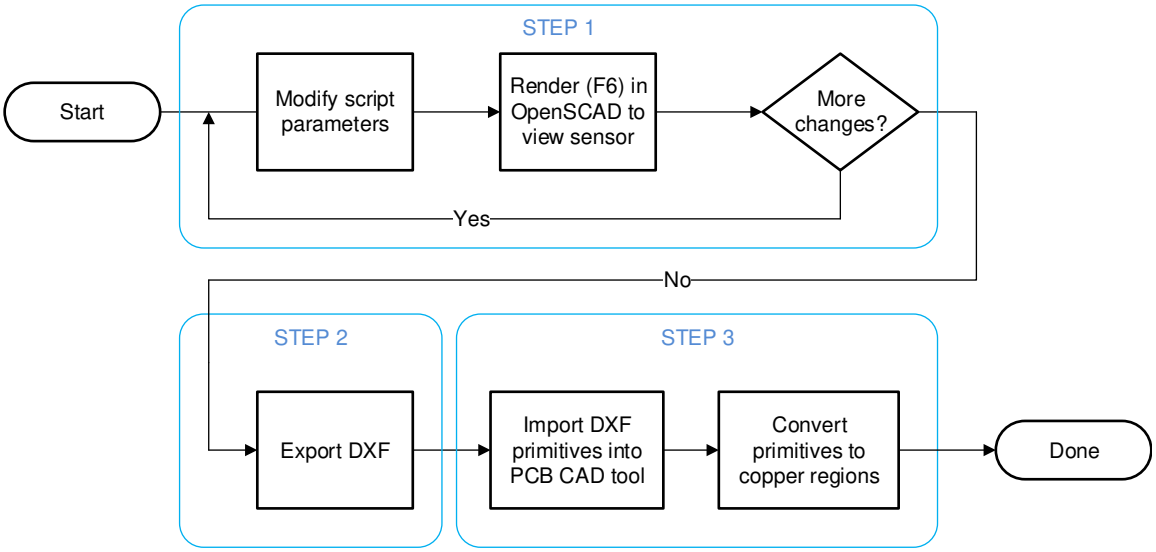


Figure 1. Workflow

2.3 Creating a Sensor

Launch the OpenSCAD program, select the menu command File>Open, and navigate to the directory where the scripts were placed. For this example, select the wheel script. After the script opens notice the menu that appears in the viewing area or canvas (see Figure 2). There are three controls that are needed to render and view the wheel sensor design. First, click the "Render" icon or F6 to render the wheel design, then click the "View All" to show the entire wheel, then "Top View" to look straight down on the wheel.

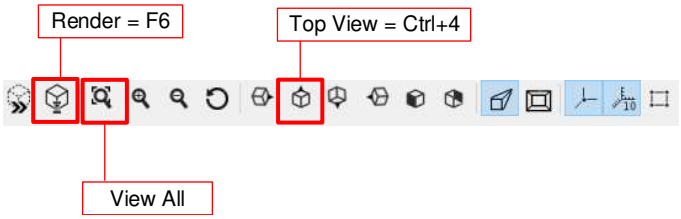


Figure 2. OpenSCAD Canvas View Menu

Figure 3 shows a wheel sensor. To zoom in or zoom out use the scroll wheel on your mouse. With the mouse cursor on the canvas, pan the view by holding down the right mouse button while moving the mouse cursor. To rotate around any of the 3 axes, hold down the left mouse button while moving the mouse cursor.

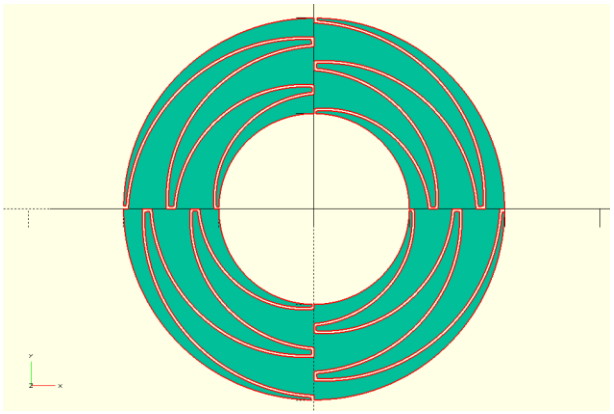


Figure 3. Default Wheel Sensor

Before modifying the script parameters, it is important to understand that OpenSCAD is unit-less and does not apply specific dimensions to the sensor design. The values used for the dimension parameters can represent whatever units you choose to work in, such as mm, mils, or inches. Just remember that when importing the DXF file into a PCB CAD tool, you must select the same scale or units used in your design.

NOTE: The accuracy of an object dimensions created by these script cannot be ensured for all possible combinations of user inputs. A reasonable effort has been made to ensure the accuracy; however, the user must verify the accuracy of any design created from these scripts.

NOTE: Depending on the size and complexity of the object that is being rendered, OpenSCAD can take several seconds to render larger and more complex designs.

The following discussion uses the example_wheel.scad script to demonstrate the steps to modify parameters, render the design, and generate the DXF outputs using batch files. Start by experimenting with a few of the parameters and render the modified wheel design. For detailed information about the wheel parameters, see Section 4.

```
// USER DEFINED INPUTS:
// STEP #1
// USER DEFINED NUMBER OF ELEMENTS IN THE WHEEL (MIN IS 3, TYPICAL IS 4)
total_elements = 4;

// STEP #2
// USER DEFINED NUMBER OF FINGERS (TINES) (TYPICAL = 5)
tines = 5;

// STEP #3 // START ANGLE (DEGREES)
start_angle = 0;

// STEP #4
// USER DEFINED OUTER RADIUS (mm IN THIS EXAMPLE)
r_outer = 20;

// STEP #5
// USER DEFINED INNER RADIUS (mm IN THIS EXAMPLE)
r_inner = 10;


// STEP #6
// USER DEFINED CLEARANCES AND TIP WIDTH (mm IN THIS EXAMPLE)
clearance = 0.25;
tip = .5;
```

2.4 Output DXF

After the sensor has been successfully rendered, it must be exported into a DXF format file that can be imported by the PCB CAD tool. The recommended method to generate the DXF file output is to select File > Export > Export DXF from the OpenSCAD menu. After export, the last step is to import the DXF into the PCB CAD tool and convert to copper regions. Section 7 describes this process.

(Optional) Creating DXF output from command line:

DXF outputs can also be created using the provided batch files from a command line interface. Although not required, these batch files automate the output process and also allow the user to generate individual DXF output files for each sensor element if needed when importing into some PCB CAD design tools.

The following example uses the command line interface to create a single sensor element and generate a DXF output.

```
openscad.exe -o wheel_element_1.dxf -D "element=1" wheel.scad
```

The syntax for the command line is:

- DXF output filename = -o wheel_element_1.dxf
- String constant parameter specifies which element = -D "element=1"
- Script to execute = wheel.scad

Generate a DXF output file for the entire sensor, change the -D parameter to "element=99".

```
openscad.exe -o wheel_all_elements.dxf -D "element=99" wheel.scad
```

For more information about executing OpenSCAD and its options from the command line, refer to the OpenSCAD documentation.

2.5 Batch Files

Batch files automate the process of generating one or more DXF output files, one for each sensor element. Using the example wheel.bat batch file, all four elements are rendered and output into separate DXF files. The contents of the wheel.bat file are:

```
openscad.exe -o wheel_element_0.dxf -D "element=0" wheel.scad
openscad.exe -o wheel_element_1.dxf -D "element=1" wheel.scad
openscad.exe -o wheel_element_2.dxf -D "element=2" wheel.scad
openscad.exe -o wheel_element_3.dxf -D "element=3" wheel.scad
```

When creating a slider or wheel sensor that has more or fewer than four elements, it is suggested to create a copy of the slider or wheel batch file and modify it to match the number of elements specified in the new script. For example, if you create a 3-element slider, copy slider.bat and give it a new name, such as slider3.bat. Edit this new file so it renders and outputs 3 elements, as shown in this example:

```
openscad.exe -o slider_element_0.dxf -D "element=0" slider.scad
openscad.exe -o slider_element_1.dxf -D "element=1" slider.scad
openscad.exe -o slider_element_2.dxf -D "element=2" slider.scad
```

To execute a batch file, open a command window and navigate to the directory where the script files are located. Type the name of the batch file including the .bat filename extension (for example, "slider3.bat").

3 Slider

Slider sensors are constructed from overlapping or inter-digitated linear sections of copper referred to as sensor "elements". The number of elements is defined by the script parameter **total_elements** (see Figure 4). The inter-digitation is created by "tines" that are defined by the script parameter **tines**. Element tines allow better coupling between electrodes and a better linear response as a finger moves up and down the slider. The left and right ends include a small region referred to as "padding" that is defined by the script width parameter **padding**. Sliders need at least 3 elements to work properly and rarely need more than 4 elements to provide an adequate linear active area of input. The overall length and width of the slider are defined by the script parameters **slider_length** and **slider_width**, respectively.

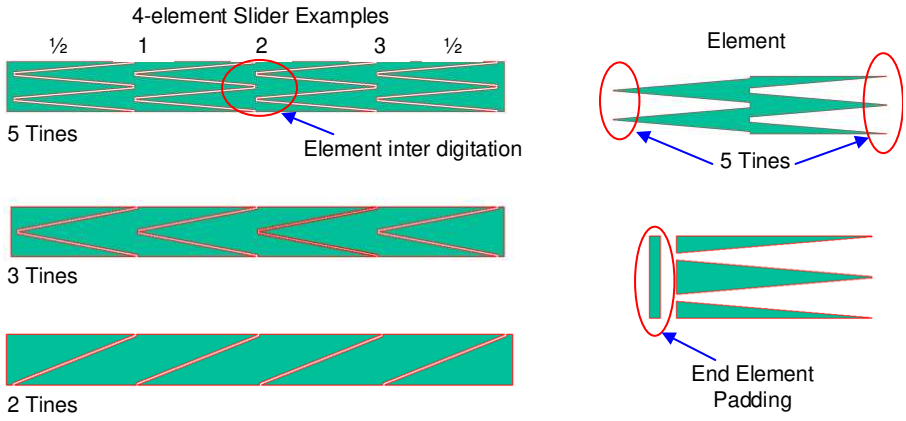


Figure 4. Slider Sensor Elements

Two additional sensor attributes are spacing and tip width. The spacing between neighboring electrodes is defined by the script parameter **clearance**. The width of the tip of each tine is defined by the script parameter **tip_width** (see Figure 5).

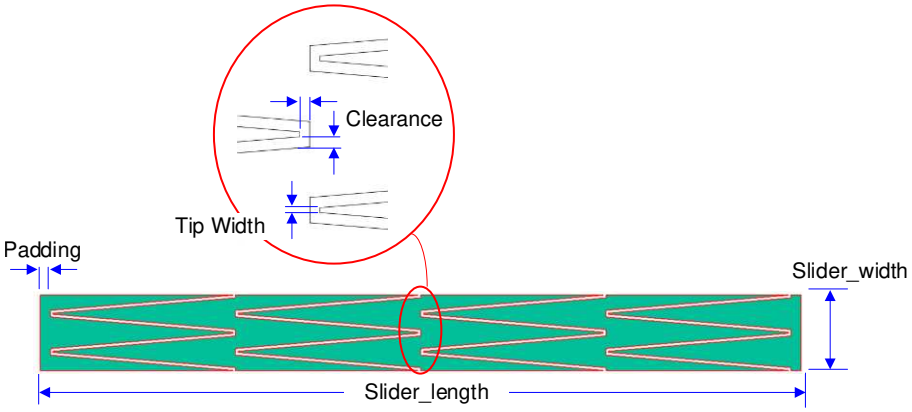


Figure 5. Slider Sensor Clearance

4 Wheel

Wheel sensors are constructed from overlapping or inter-digitated curved sections of copper referred to as sensor "elements". The number of elements is defined by the script parameter **total_elements**. The inter-digitation is created by "tines" that are defined by the script parameter **tines** (see Figure 6). Element tines allow better coupling between electrodes and a better linear response as a finger moves up and down the slider. Like the slider, the wheel needs a minimum of 3 elements and rarely needs more than 4 elements to provide an good linear input response.

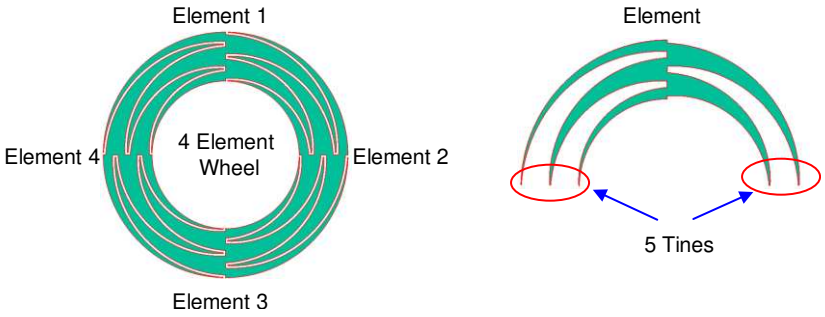


Figure 6. Wheel Elements

The outside perimeter of the wheel is defined by the script parameter **r_outside**. The inside perimeter of the wheel is defined by the parameter **r_inside**. The difference between these two parameters determines the width of the wheel. The spacing between neighboring electrodes is defined by the parameter **clearance**. The width of the tip of each tine is defined by the script parameter **tip_width**. The **clearance** and **tip_width** parameters determine the level of coupling between any two electrodes (see Figure 7).

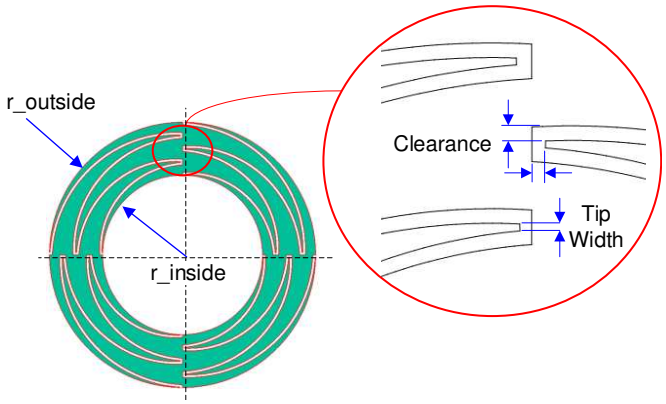


Figure 7. Wheel Radius and Clearance

A wheel is typically designed with element 0 located at the top position on the wheel. This position is defined by the parameter **start_angle**. It is possible to select other orientations if needed (see Figure 8).

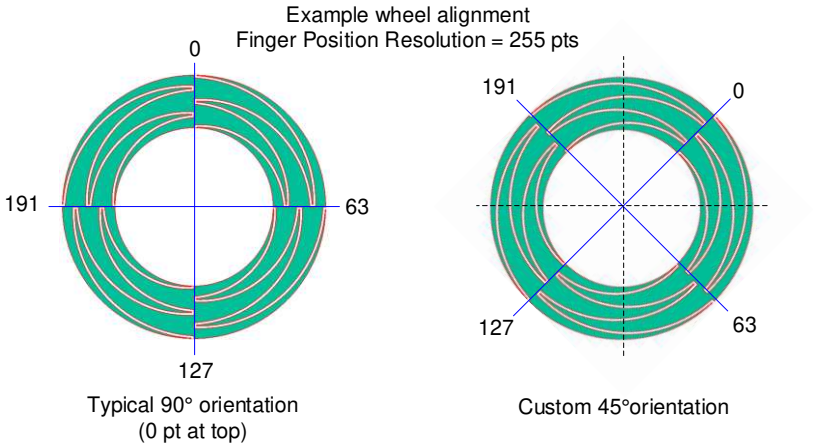


Figure 8. Wheel Orientation

5 Curved Slider

Curved slider sensors are essentially a hybrid between a slider and a wheel. They may be useful in some product applications where a full wheel HMI input is not needed and a linear slider does not fit the product's mechanical design, or the designer is looking for product differentiation. The arc of a curved is controlled by the script parameters **r_outside**, **r_inside**, **start_angle**, and **stop_angle**. Figure 9 shows an example. For a description of these and the other sensor parameters, see Section 3 and Section 4.

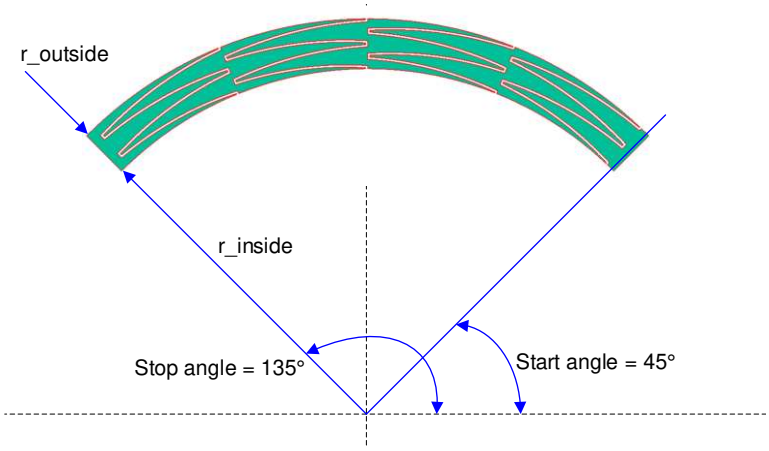


Figure 9. Curved Slider Dimensions

NOTE: The curved slider script can also generate a wheel sensor by setting start_angle = 0, stop_angle = 360, and padding = 0.

6 Touchpad

Touchpad sensors are constructed using a matrix of diamond shaped patterns with an outline that can be square (rectangular), circular, or custom as defined by the script parameter **shape**. The number of diamonds to create is defined by the script parameters **rows** and **columns**. The size of the touchpad area is defined by the script parameters **touchpad_width** and **touchpad_height**. The last dimension is the diamond spacing and is defined by the script parameter **diamond_spacing**. The touchpad script uses these parameters to calculate the diamond size and generates a 2-dimensional matrix of diamonds fitted to the specified shape (see Figure 10).

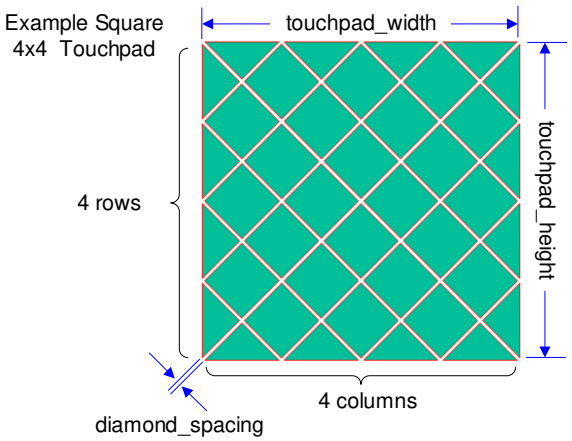


Figure 10. Touchpad Dimensions

As mentioned, there are four touchpad shapes that can be created, with a few examples shown in Figure 11.

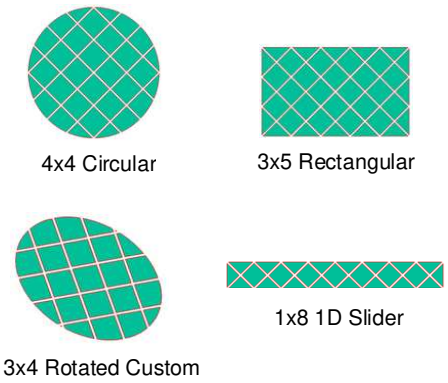
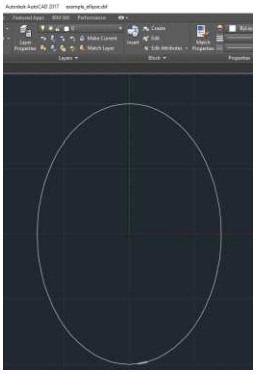


Figure 11. Touchpad Shapes

The two example rectangular shapes were created with the **shape** parameter defined as a square, but using unequal width and height dimensions to create the rectangular shape. To maintain a symmetrical diamond shape, the width and height dimensions should be proportional to the number of rows and columns. For example, the 3x5 (3 rows, 5 cols) rectangle has a height of 30 mm and width of 50 mm. If not proportional, the diamonds become elongated, which is acceptable but can cause a different response to the motion of a finger in the X direction compared to the Y direction.

The custom touchpad in Figure 12 was created by setting the **shape** parameter to custom. This parameter does not actually create the custom shape, rather it imports any shape outline that can be created by a CAD tool that can create a DXF output, such as AutoCAD. In the touchpad script, an example DXF file, example_ellipse.dxf is specified by default and is provided in the zip file download to let you experiment with this option. You can create custom shapes and export to a DXF file, then substitute the example with dxf = "user_defined_shape.dxf". In addition, there is a parameter that rotates the shape and the diamond matrix together if needed before importing into the PCB CAD tool.

Create ellipse
H = 20 mm, W = 14 mm



```
//================ USER INPUTS ================
// #1 - SPECIFY IF ROUND OR SQUARE OR CUSTOM
shape = $custom;

// #2 - SPECIFY THE NUMBER OF RX AND TX
rows = 4;
columns = 4;

// #3 - SPECIFY THE WIDTH AND HEIGHT OF THE TOUCHPAD IN UNITS
//      IF CIRCLE, WIDTH AND HEIGHT SHOULD BE SET EQUAL
//      IF CUSTOM, REFER TO DXF OUTPUT
touchpad_width = 14;
touchpad_height = 20;

// #4 - SPECIFY THE EDGE TO EDGE SPACING BETWEEN DIAMONDS
diamond_spacing = .2;

// #5 - SPECIFY DXF FILE TO IMPORT
dxf = "example_ellipse.dxf"; // example file (20mm x 14mm ellipse)

// #6 - (OPTIONAL FOR CUSTOM) SPECIFY OBJECT ROTATION (DEGREES)
sensor_rotation = 60;
```

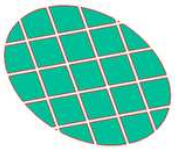


Figure 12. Custom Touchpad Example

7 PCB CAD Tool

This section describes the steps to import and convert the DXF primitives generated by the sensor scripts to their equivalent copper regions on the PCB using Altium Designer. For other PCB CAD tools, refer to the PCB CAD tool documentation to perform the equivalent steps.

This example uses the DXF file "example_wheel.dxf" that is included with the downloaded scripts. This description assumes that you have an Altium Designer program open and a project with a schematic and PCB docs. To import the DXF primitive, switch to the PCB doc and select the File > Import > DXF/DWG menu, then navigate to the **C:\OpenSCAD\Projects** directory where the zip file was extracted and select example_wheel.dxf. When the Import File dialog box appears, choose the units that were selected in the original design (see Section 2) (for this example, select Scale = mm) (see Figure 13). Next, set the lines to very narrow (virtually zero width) and select a copper layer, such as the Top Layer, to place the imported primitive. The last step is to position the mouse where the origin of the wheel should be and click the SELECT button, then click the OK button. Figure 14 shows the result of a successful import.

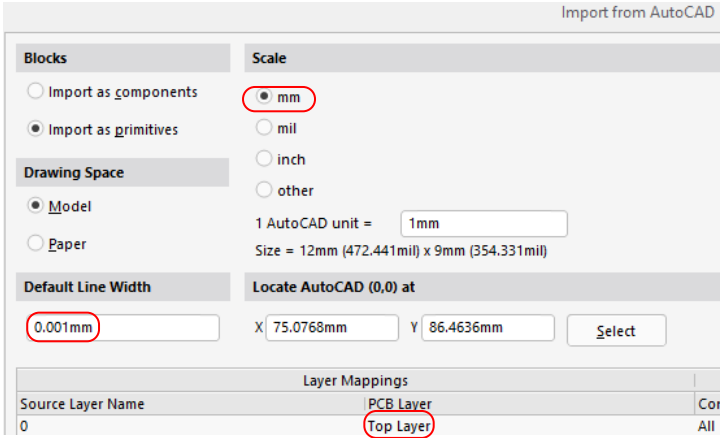


Figure 13. Altium Import Dialog

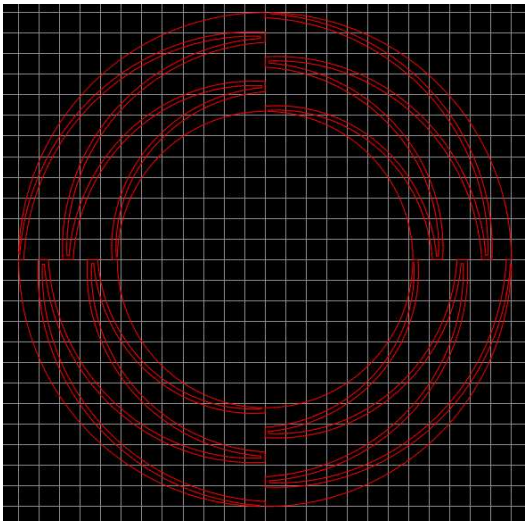


Figure 14. Imported DXF Primitives

The next steps convert the DXF primitives into individual copper regions.

1. Place a polygon pour over the whole sensor and modify its properties (see Figure 15).
2. Adjust the design rules so the copper pour has nearly no clearance in relation to all objects.
3. Repour the polygon.
4. From the menu select Tools > Convert > Explode Polygon to Free Primitives.
5. Select and delete the large region that encompasses the smaller regions.

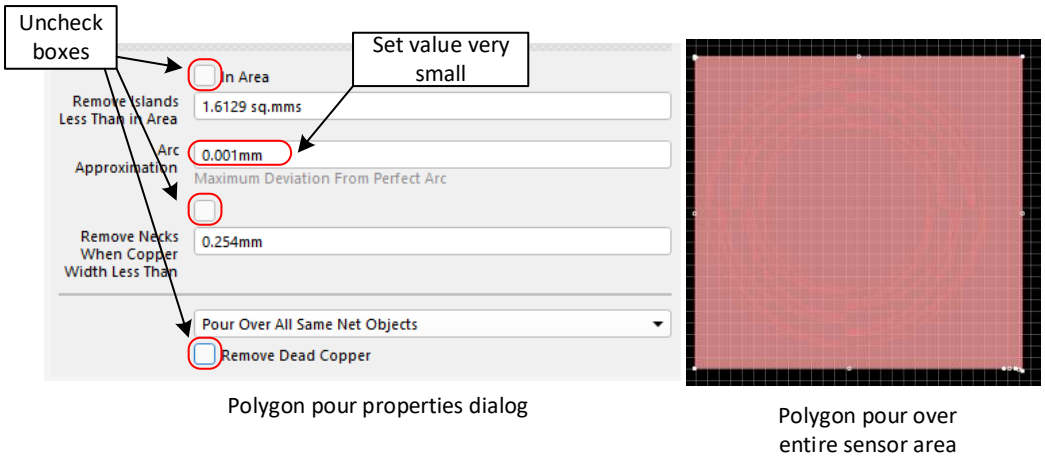


Figure 15. Polygon Pour and Modified Properties

When complete, the design should look like Figure 16.

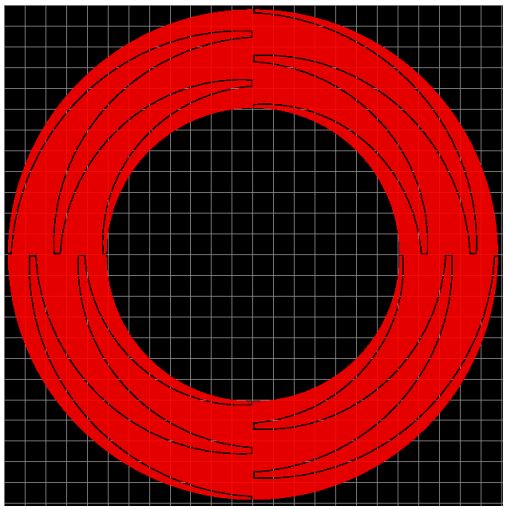


Figure 16. Finished Wheel

Congratulations! You have completed your first capacitive touch wheel design.

8 Summary

The barrier to designing complex capacitive touch sliders and wheels has now been eliminated. Creating scalable capacitive touch slider, wheel, and touchpad PCB sensors can now be quick and simple using OpenSCAD and custom scripts provided by Texas Instruments. This document demonstrated the three steps required to create a wheel sensor on a PCB and describes the anatomy of these sensor types. For your next capacitive touch design, consider using one of these sensor types and leveraging the power of these scripts.

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from July 26, 2019 to February 26, 2020 **Page**

- Changed the document title 1
- Updated the abstract 1
- Throughout document, changed description of sensors to include touchpads 1
- Changed Figure 1 *Workflow* 4
- Deleted the sentence that began "Notice that element 0 is..." in the first paragraph of Section 3 *Slider* 7
- Changed Figure 4 *Slider Sensor Elements* 7
- Changed Figure 5 *Slider Sensor Clearance* 7
- Changed Figure 6 *Wheel Elements* 8
- Changed Figure 7 *Wheel Radius and Clearance* 8
- Changed Figure 8 *Wheel Orientation* 9
- Added Section 6 *Touchpad* 10

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265