TXLWizard: Reference Guide

Esteban Marín

2016-05-13

Contents

1 Introduction **3**
1.1 What does it do? . . . 3
1.2 Technical Information . . . 3

2 TXLWizard Example **4**

3 Class Reference **8**
3.1 TXLWizard.TXLWriter . . . 8
3.1.1 Description . . . 8
3.1.2 Methods . . . 8
3.1.3 Properties . . . 8
3.1.4 Example Usage . . . 8
3.2 Strukturen . . . 8
3.2.1 DataStructure . . . 8
3.2.2 FieldStructure . . . 11
3.2.3 PageStructure . . . 11
3.2.4 FormFieldStructure . . . 11

4 TXLConverter **12**

Appendices **13**

A TXLWizard: Advanced Example **13**

1 Introduction

This document describes the usage and technical reference of the python program "TXL-Wizard" written by Esteban Marin (estebanmarin@gmx.ch).

1.1 What does it do?

The "TXLWizard" provides routines for generating TXL files (.txl) for the preparation of E-Beam lithography masks using python code. The TXL files can be processed with BEAMER. See the following links:

- `http://genisys-gmbh.com/web/products/beamer.html`
- `http://cad035.psi.ch/LB_index.html`
- `http://cad035.psi.ch/LBDoc/BEAMER_Manual.pdf`

The generated TXL files are also converted to HTML / SVG for presentation in any modern browser or vector graphics application.
Moreover, a command line interface "TXLConverter" provides conversion of existing TXL files to HTML / SVG (See Section 4).

1.2 Technical Information

The "TXLWizard" is written in python and will run in Python version 2.7+ and 3.1+.
In order to use the "TXLWizard", package must be available as a python package, i.e. either it must be copied to `Path_to_my_python_installation/site-packages/` or to the path where your script is located.
Alternatively, you can also prepend the following command to your python script:

- Windows: `sys.path.append('path to the TXLWizard')`
- Linux: `sys.path.append('path to the TXLWizard')`

2 TXLWizard Example

The following code demonstrates a simple example usage of the “TXLWizard” for generating TXL files with python code.
The code can be found in the file `Content/Example_Simple.py`.
The resulting image is shown in Figure 1. A more advanced example is shown in Section A

```python
######################
# Import Libraries #
######################

# Import TXLWriter, the main class for generating TXL Output
import TXLWizard.TXLWriter

# Import Pre-Defined Shapes / Structures wrapped in functions
import TXLWizard.ShapeLibrary.EndpointDetectionWindows
import TXLWizard.ShapeLibrary.Label

# Import math module for calculations
import math


####################################
# Sample / Structure Parameters #
####################################

# Define all sample parameters
SampleParameters = {
    'Width': 8e3,
    'Height': 8e3,
    'Label': 'Simple Demo',
}

# Define all structure parameters
StructureParameters = {
    'Circle': {
        'Radius': 50,
        'Layer': 3
    },
    'CircleArray': {
        'Columns': 6,
        'Rows': 5,
        'ArrayXOffset': 500,
        'ArrayYOffset': -500,
        'ArrayOrigin': [0.75e3, 3e3],
        'Label': 'R{:d}C{:d}',
    }
}
```

```
42
43
44 ##########################
45 # Initialize TXLWriter #
46 ##########################
47 TXLWriter = TXLWizard.TXLWriter.TXLWriter(
48     Width=SampleParameters['Width'],
49     Height=SampleParameters['Height']
50 )
51
52 ######################
53 # Define Structures #
54 ######################
55
56 ## Sample Label ##
57
58 # Give the sample a nice label
59 SampleLabelObject = TXLWizard.ShapeLibrary.Label.GetLabel(
60     TXLWriter,
61     SampleParameters['Label'],
62     OriginPoint=[
63         0.5e3, 1. * SampleParameters['Height'] / 2. - 500
64     ],
65     FontSize=150,
66     StrokeWidth=20,
67     RoundCaps=True, # Set to False to improve e-Beam performance
68     Layer=1
69 )
70
71
72 ## Endpoint Detection ##
73
74 # Use Pre-Defined Endpoint Detection Windows
75 TXLWizard.ShapeLibrary.EndpointDetectionWindows.
       GetEndpointDetectionWindows(
76     TXLWriter, Layer=1)
77
78 ## User Structure: Circle ##
79
80 # Create Definition Structure for Circle that will be reused
81 CircleStructure = TXLWriter.AddDefinitionStructure('Circle')
82 CircleStructure.AddPattern('Circle',
83     Center=[0, 0],
84     Radius=StructureParameters['Circle']['Radius'],
85     Layer=StructureParameters['Circle']['Layer']
86
87 )
88
```

```
89  # Create array of the definition structure above
90  CircleArray = TXLWriter.AddContentStructure('CircleArray')
91  CircleArray.AddPattern('Array',
92      ReferencedStructureID=CircleStructure.ID,
93      OriginPoint=StructureParameters['CircleArray']['ArrayOrigin'
        ],
94      PositionDelta1=[
95          StructureParameters['CircleArray']['ArrayXOffset'], 0
96      ],
97      PositionDelta2=[
98          0, StructureParameters['CircleArray']['ArrayYOffset']
99      ],
100     Repetitions1=StructureParameters['CircleArray']['Columns'],
101     Repetitions2=StructureParameters['CircleArray']['Rows']
102 )
103
104
105
106 ############################
107 # Generate Output Files #
108 ############################
109
110 # Note: The suffix (.txl, .html, .svg) will be appended
    automatically
111 TXLWriter.GenerateFiles('Masks/Example_Simple')
```

Content/Example_Simple.py

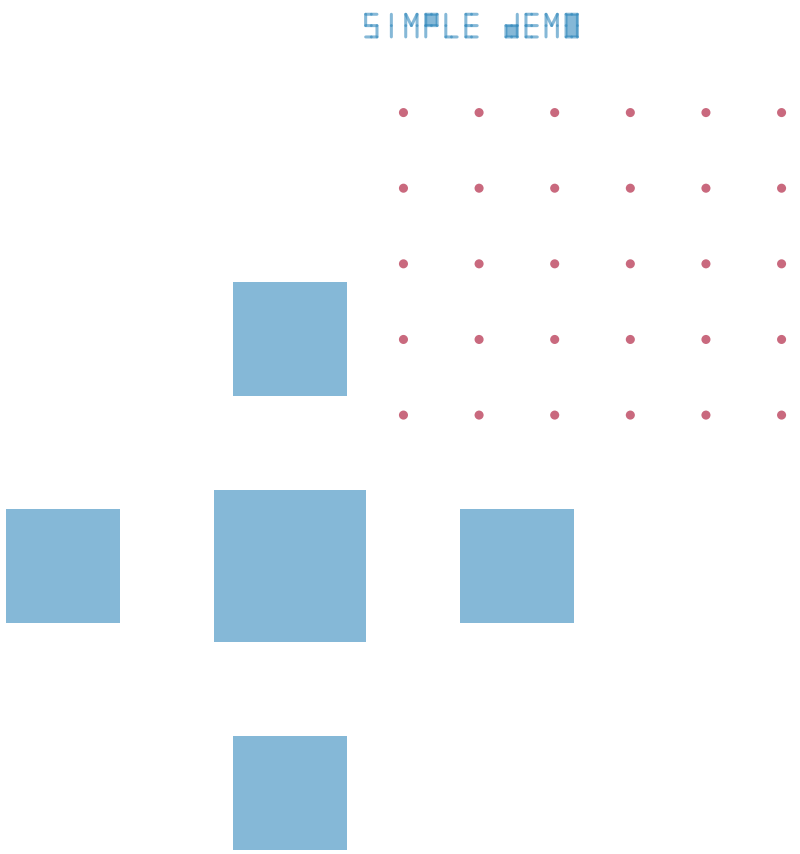

Figure 1: Simple Example: Generated Mask

3 Class Reference

This section provides information about the public API of “TXLWizard”. For a full class reference, see the documentation in the corresponding source code.

3.1 TXLWizard.TXLWriter

3.1.1 Description

Controller class for generating TXL / SVG / HTML output. Here we can add structures (definitions and content) which will be rendered in the output.

3.1.2 Methods

`__init__`
Description Constructor method
Parameters
Keyword Arguments (optional)

- **Width** `int` Width of the sample in um. Used to draw coordinate system.
- **Height** `int` Height of the sample in um. Used to draw coordinate system.
- **ShowCoordinateSystem** `bool` Show the coordinate system or not
- **GridDistance** `int` Coordinate Sytem Grid Spacing in um.
- **SubGridDistance** `int` Coordinate System Sub-Grid Spacing in um

3.1.3 Properties

3.1.4 Example Usage

```
1 import TXLWizard.TXLWriter
2
3 TXLWriter = TXLWizard.TXLWriter.TXLWriter(
4     Width=500,
5     Height=400
6 )
```

3.2 Strukturen

3.2.1 DataStructure

Die DataStructure ist wie folgt definiert und gilt jeweils für eine Datenbank-Tabelle:

- **Control** `Array` mit Informationen zur Datentabellen-Kontrolle.
  - **Table*** `String` Tabellen-Name
  - **IndexColumn*** `String` Name der Index-Spalte. Standardwert: “ID”

  - **ModifiedTSColumn*** `String` Name der Spalte mit Änderungs-Timestamp. Standardwert: “ModifiedTS”
  - **CategoryColumn*** `String` Name der Kategorien-Spalte. Falls angegeben (muss vom **Type Module** mit **DataCardinality** “1:n” sein), kann nach dieser Spalte kategorisiert werden. (z.B. select-Feld, etc.) Standardwert: “ID”
  - **NoUserInput*** `Bool` Ob kein User-Input in die Daten gelangen darf. Standardwert: `false`
  - **ReadOnly*** `Bool` Ob Datenbank-Tabelle schreibgeschützt ist. Standardwert: `false`
  - **NoPurge*** `Bool` Ob gelöschte Einträge in Datenbank-Tabelle nicht gelöscht werden sollen. Standardwert: `false`
  - **DBConnectionID*** `String` Name der Datenbankverbindung. Standardwert: “ClusterNetAppDB”

- **Columns** `Array` mit Definitionen der einzelnen Daten-Spalten. `Key` ist der Spalten-Name und `Value` ein `Array` mit der Definition:
  - **Type** `String` Typ der Spalte. Kann folgende Werte annehmen:
    * **Integer** Ganzzahl
    * **String** Text
    * **Float** Gleitkommazahl
    * **Checkbox** Checkbox
    * **Date** Datum
    * **Time** Zeit
    * **DateTime** Datum und Uhrzeit
    * **Array** `Array`, wird als JSON gespeichert.
    * **Module** Fremdschlüssel (ID, Referenz) zu einem Eintrag in einem anderen Modul.
    * **FileResource** Dateien
    * **Option** Mehrfachauswahl

    Wird **Type** nicht angegeben, so wird in `$GLOBALS['DefaultFieldStructures']` nach einem Eintrag gesucht. In diesem Fall kann lediglich der **DefaultValue** erhalten bleiben.
  - **Validation*** `Array` mit Validierungs-Optionen.
    * **Required*** `Bool` Ob Pflichtfeld. Standardwert: `false`
  - **LabelKey*** `String` Key für Language Label. Standardwert: “”
  - **TypeSpecificConfiguration*** `Array` spezifische Konfiguration (abhängig von **Type**)
    Type *“Module”*
    * **ModulePath** `String` Pfad des Moduls
    * **MModulePath*** `String` Pfad des M-Moduls der n:m-Relation (benötigt für **DataCardinality** “n:m”)
    * **DataCardinality*** `String` Kardinalität. Erlaubt: “1:n”, “n:m”, Standardwert: “1:n”

- **NToMTable**[*] `String` Tabelle für n:m-Relation. (benötigt für **DataCardinality** "n:m")
- **NForeignKeyColumn**[*] `String` Foreign Key Spalte n:m-Relation (n-Tabelle). (benötigt für **DataCardinality** "n:m")
- **MForeignKeyColumn**[*] `String` Foreign Key Spalte für n:m-Relation. (m-Tabelle) (benötigt für **DataCardinality** "n:m")
- **ModuleInstance**[*] `Reference` Referenz zu einer Modul-Instanz. Falls leer, wird diese anhand des **ModulePath** erzeugt.
- **MModuleInstance**[*] `Reference` Referenz zu einer M-Modul-Instanz. Falls leer, wird diese anhand des **ModulePath** erzeugt. (für **DataCardinality** "n:m")
- **ReplicateToJS**[*] `Bool` ob Javascript-Modul-Instanz erstellt werden soll. Standardwert: `false`
- **OverrideInstanceProperties**[*] `Array` Eigenschaften, die bei Instanziierung des Objekts gesetzt werden sollen. Standardwert: leeres `Array`
- **OverrideForeignInstanceProperties**[*] `Array` Eigenschaften, die bei Instanziierung des M-Objekts gesetzt werden sollen. (für **DataCardinality** "n:m") Standardwert: leeres `Array`
- **DetailViewType**[*] `String` Anzeigemodus in der Detailansicht. (für **DataCardinality** "n:m"). Erlaubte Werte: "Full", "Minimal" Standardwert: "Minimal"
- **OptionListAdditionalColumns**[*] `Array` Spalten, die beim Select-option zusätzlich im Frontend vorhanden sein sollen (für **DataCardinality** "1:n"). Standardwert: leeres `Array`
- **RequirePermission** `Bool` ob relationaler Eintrag gelesen werden können muss, um Datensatz zu lesen. Standardwert: `false`

Type *"Date"*

- **AddCurrentTimeToInputDate**[*] `Bool` ob beim Parsen von User-Input die aktuelle Uhrzeit mitgespeichert werden soll. Standardwert: `false`

Type *"String"*

- **EncryptDBContent**[*] `Bool` Ob Wert in der Datenbank verschlüsselt werden soll. Standardwert: `false`
- **RichText**[*] `Bool` Ob Rich Text editing & anzeige aktiviert ist. Standardwert: `false`
- **AdditionalAllowedHTMLTags**[*] `String` Weitere erlaubte Tags. Standardwert: ""

Type *"FileResource"*

- **IsCollection**[*] `Bool` Ob mehrere Dateien. Standardwert: `false`

Type *"Integer"*

- **RoundingFactor**[*] `Float` Rundungsfaktor. Standardwert: 1

Type *"Float"*

- **RoundingFactor**[*] `Float` Rundungsfaktor. Standardwert: 1

Type *"Checkbox"*

    - **YesNoLabelKeys*** `Array` spezielle Yes/No Label Keys. Standardwert: leeres `Array`

  Type *"Option"*

    - **Options** `Array` Optionen, jede es Element ein `Array` (ID =¿ 'ID', LabelKey =¿ 'LabelKey')
- **Visible*** `Bool` sichtbar? Standardwert: `true`
- **DefaultValue*** `String` Initialwert. Standardwert: **Type**-abhängigen Standardwert gesetzt.
- **Update*** `Bool` ob Feld bei Update in DB geschrieben wird. Standardwert: `true`
- **Insert*** `Bool` ob Feld bei Insert in DB geschrieben wird. Standardwert: `true`
- **NoUserInput*** `Bool` Ob Feld keine Benutzer-Eingaben erhalten darf. Standardwert: `false`
- **DisplayOptions*** Anzeigeoptionen. Wird an Form übergeben. Standardwert: leeres `Array`
    - **IsPassword*** `Bool` ob Textfeld als Password dargestellt werden soll. Standardwert: `false`
    - **IsCurrency*** `Bool` ob Textfeld als Währung dargestellt werden soll. Bsp: `2'700.35` Standardwert: `false`
    - **IsLanguageLabelKey*** `Bool` ob es ein Language Label Key ist. (für **type String**) Standardwert: `false`
    - **Multiline*** `Bool` ob mehrzeiliges Textfeld (nur für **String**) Standardwert: `false`
    - **FormRowCSSClasses*** `String` zusätzliche Klassen für Formular-Zeile. Standardwert: ""
    - **CSSClasses*** `String` zusätzliche Klassen für Formular-Feld. Standardwert: ""
    - **LinkURLs*** `Bool` ob URLs im Inhalt gesucht und verlinkt werden sollen (nur für **String**) Standardwert: `false`
    - **DateFormat*** `String` Datumsformat für "Date"-Feld. Standardwert: "d.m.Y"
- **FormFieldParameters*** `Array` Parameter, die an das Formular-Objekt übergeben werden. Standardwert: leeres `Array`
    - **IsTabFocusField*** `Bool` ob Textfeld im aktuellen Tab standardmässig fokussiert sein soll. Standardwert: `false`
- **SessionPersistent*** `Bool` Ob Wert der Variable Session-persistent gespeichert werden soll. Standardwert: `false`
- **UserPersistent*** `Bool` Ob Wert der Variable User-persistent gespeichert werden soll. Standardwert: `false`

3.2.2 FieldStructure

3.2.3 PageStructure

3.2.4 FormFieldStructure

4 TXLConverter

Appendices

A TXLWizard: Advanced Example

The following code demonstrates the usage of "TXLWizard" in a more advanved way, including labelling of array objects, nested referencing, etc. The generated mask is show in Figure 2.

```
1  ######################
2  # Import Libraries #
3  ######################
4
5  # Import TXLWriter, the main class for generating TXL Output
6  import TXLWizard.TXLWriter
7
8  # Import Pre-Defined Shapes / Structures wrapped in functions
9  import TXLWizard.ShapeLibrary.EndpointDetectionWindows
10 import TXLWizard.ShapeLibrary.Markers
11 import TXLWizard.ShapeLibrary.Label
12 import TXLWizard.ShapeLibrary.CornerCube
13
14 # Import math module for calculations
15 import math
16
17
18 ###################################
19 # Sample / Structure Parameters #
20 ###################################
21
22 # Define all sample parameters
23 SampleParameters = {
24     'Width': 8e3,
25     'Height': 8e3,
26     'Label': 'GOI Demo CornerCube',
27 }
28
29 # Define all structure parameters
30 StructureParameters = {
31     'CornerCube': {
32         'BridgeLength':8,
33         'ParabolaFocus': 9,
34         'XCutoff': 9,
35         'AirGapX': 3,
36         'AirGapY': 1,
37         'LabelXOffset': 0,
38         'LabelYOffset': 50,
39         'Label': 'R{:d}C{:d}',
40         'Layer': 2
```

```
41      },
42      'Circle': {
43          'Radius': 5,
44          'Layer': 3
45      },
46      'CornerCubeArray': {
47          'Columns': 6,
48          'Rows': 5,
49          'ArrayXOffset': 500,
50          'ArrayYOffset': -500,
51          'ArrayOrigin': [0.75e3, 3e3]
52      }
53  }
54
55
56  ############################
57  # Initialize TXLWriter #
58  ############################
59  TXLWriter = TXLWizard.TXLWriter.TXLWriter(
60      Width=SampleParameters['Width'],
61      Height=SampleParameters['Height']
62  )
63
64  ######################
65  # Define Structures #
66  ######################
67
68  ## Sample Label ##
69
70  # Give the sample a nice label...
71  SampleLabelObject = TXLWizard.ShapeLibrary.Label.GetLabel(
72      TXLWriter,
73      SampleParameters['Label'],
74      OriginPoint=[
75          0.5e3, 1. * SampleParameters['Height'] / 2. - 500
76      ],
77      FontSize=150,
78      StrokeWidth=20,
79      RoundCaps=True,# Set to False to improve e-Beam performance
80      Layer=1
81  )
82  # ...and some other information
83  Alphabet = TXLWizard.ShapeLibrary.Label.GetLabel(
84      TXLWriter,
85      'abcdefghijklmnopqrstuvwxyz0123456789 megamega ggg ah
          extraaaa rischaaaar',
86      OriginPoint=[
87          0.5e3, 1. * SampleParameters['Height'] / 2. - 600
```

```
88      ],
89      FontSize=50,
90      StrokeWidth=3,
91      RoundCaps=True, # Set to False to improve e-Beam performance
92      Layer=1
93  )
94
95  ## Endpoint Detection ##
96
97  # Use Pre-Defined Endpoint Detection Windows
98  TXLWizard.ShapeLibrary.EndpointDetectionWindows.
        GetEndpointDetectionWindows(
99      TXLWriter, Layer=1)
100
101 ## Alignment Markers ##
102
103 # Use Pre-Defined Alignment Markers
104 TXLWizard.ShapeLibrary.Markers.GetMarkers(
105     TXLWriter, Layer=1)
106
107
108 ## User Structure: Corner Cube ##
109
110 # Create Definition Structure for Corner Cube that will be
        reused
111 CornerCubeDefinition = TXLWizard.ShapeLibrary.CornerCube.
        GetCornerCube(
112     TXLWriter,
113     ParabolaFocus=StructureParameters['CornerCube']['
            ParabolaFocus'],
114     XCutoff=StructureParameters['CornerCube']['XCutoff'],
115     AirGapX=StructureParameters['CornerCube']['AirGapX'],
116     AirGapY=StructureParameters['CornerCube']['AirGapY'],
117     Layer=StructureParameters['CornerCube']['Layer']
118 )
119
120 # Create Definition Structure for combination of cornercube and
        additional circle
121 FullCornerCubeNoRotation = TXLWriter.AddDefinitionStructure('
        FullCornerCubeNoRotation')
122 FullCornerCubeNoRotation.AddPattern('Reference',
123     ReferencedStructureID=CornerCubeDefinition.ID,
124     OriginPoint=[1. * StructureParameters['CornerCube']['
            BridgeLength'] / 2., 0]
125 )
126 FullCornerCubeNoRotation.AddPattern('Circle',
127     Center=[0, 0],
128     Radius=StructureParameters['Circle']['Radius'],
```

```
129     Layer=StructureParameters['Circle']['Layer']
130 )
131
132 # Create definition structure with rotation of entire referenced
        structure
133 FullCornerCube = TXLWriter.AddDefinitionStructure('
    FullCornerCube',
134                                                    RotationAngle
                                                           =45)
135 FullCornerCube.AddPattern('Reference',
136     ReferencedStructureID=FullCornerCubeNoRotation.ID,
137     OriginPoint=[0, 0]
138 )
139
140 # Create array of the definition structure above
141 CornerCubeArrayFine = TXLWriter.AddContentStructure('
    CornerCubeArrayFine')
142 CornerCubeArrayFine.AddPattern('Array',
143     ReferencedStructureID=FullCornerCube.ID,
144     OriginPoint=StructureParameters['CornerCubeArray']['
          ArrayOrigin'],
145     PositionDelta1=[
146         StructureParameters['CornerCubeArray']['ArrayXOffset'],
              0
147     ],
148     PositionDelta2=[
149         0, StructureParameters['CornerCubeArray']['ArrayYOffset'
              ]
150     ],
151     Repetitions1=StructureParameters['CornerCubeArray']['Columns
          '],
152     Repetitions2=StructureParameters['CornerCubeArray']['Rows']
153 )
154
155
156 # Add Labels to each array element
157 for Row in range(1, StructureParameters['CornerCubeArray']['Rows
    '] + 1):
158     for Column in range(1, StructureParameters['CornerCubeArray'
          ]['Columns'] + 1):
159         RowColumnCountLabel = TXLWizard.ShapeLibrary.Label.
              GetLabel(
160             TXLWriter,
161             StructureParameters['CornerCube']['Label'].format(
                  Row, Column),
162             OriginPoint=[
163                 StructureParameters['CornerCubeArray']['
                      ArrayOrigin'][0]
```

```
164                     + StructureParameters['CornerCubeArray']['
                            ArrayXOffset']
165                     * (Column - 1) + StructureParameters['CornerCube
                            ']['LabelXOffset'],
166                     StructureParameters['CornerCubeArray']['
                            ArrayOrigin'][1]
167                     + StructureParameters['CornerCubeArray']['
                            ArrayYOffset']
168                     * (Row - 1) + StructureParameters['CornerCube']['
                            LabelYOffset']],
169                 FontSize=16,
170                 StrokeWidth=3,
171                 RoundCaps=True,# Set to False to improve e-Beam
                        performance
172                 Layer=1,
173                 RotationAngle=45
174             )
175
176
177 ############################
178 # Generate Output Files #
179 ############################
180
181 # Note: The suffix (.txl, .html, .svg) will be appended
        automatically
182 TXLWriter.GenerateFiles('Masks/Example_Advanced')
```

Content/Example_Advanced.py

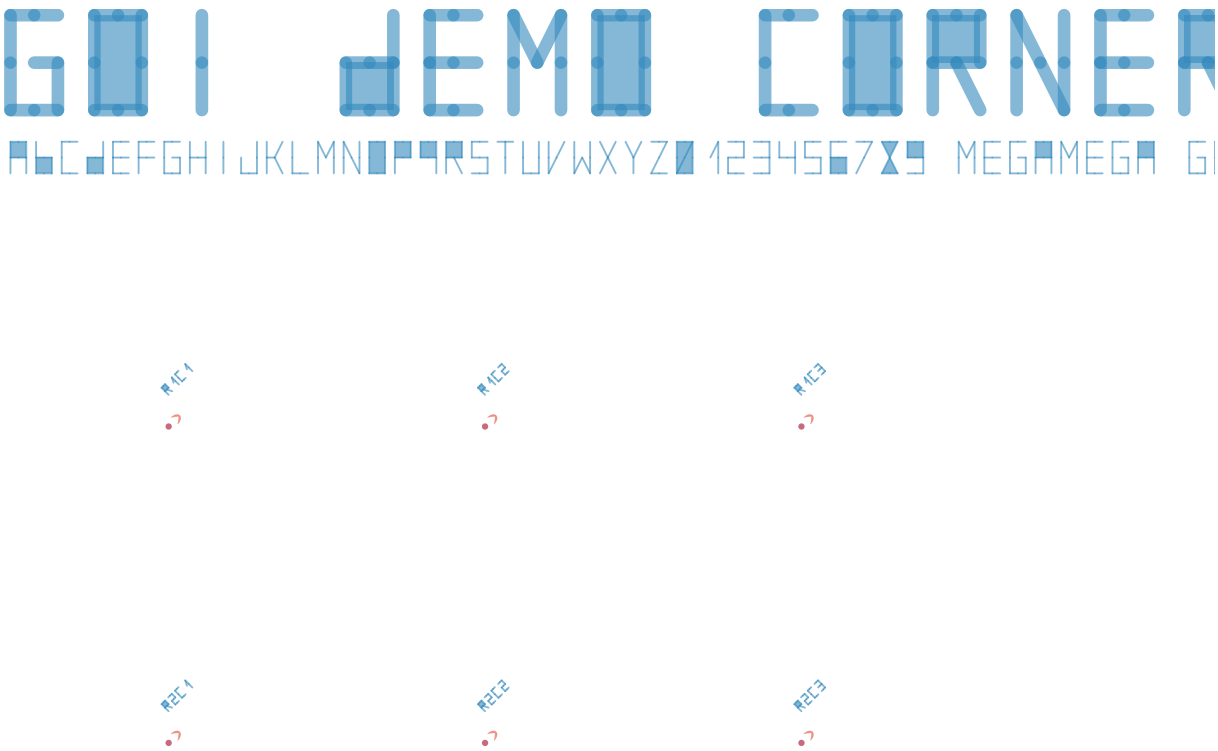

Figure 2: Advanced Example: Part of the Generated Mask