

通用文档格式

- [1 文档约定](#)
- [2 项目打包结构](#)
 - [2.1 概述](#)
 - [2.2 .zip 文件组织方式](#)
 - [2.3 project.json 信息文件](#)
- [3 通用文件头格式](#)
- [4 二进制数据 BLOB](#)
- [5 修订日志](#)
 - [5.1 2022030201](#)
 - [5.2 2020102801](#)
 - [5.3 2020101201](#)
 - [5.4 2020100901](#)
 - [5.5 2020092801](#)

1 文档约定

- 以行为单位，每一行都是一个合法的 JSON 数组，此数组称为 图元结构数组
- 如格式无特殊说明，图元结构数组中，每一个元素的位置都是固定的，同一个元素 不会出现 按情况划分到不同位置的场景（除了跨版本兼容）
- 旋转角度 以逆时针方向为正，统一使用角度制
- 若无特殊说明，所有坐标、长度、大小统一使用 0.01 inch 为单位
- 几乎所有图元都要带上文件内的 唯一编号
- 几乎所有图元都带有 锁定 参数，对于已经锁定的图元，编辑器里应表现出如下行为特征
 1. 不能拖拽
 2. 不能删除
 3. 不能通过鼠标键盘调整大小
 4. 不能通过鼠标键盘调整形状
- 所有颜色都使用 "#RRGGBB" 的方式表达，如果需要表示无颜色（完全透明），则用 ""
- 所有具有范围特性的图元，比如 PART 等，以及文件本身，都称作 块级图元
- 所有使用 是否XXXX 描述的属性，都使用 1 是 0 否 的编码方式
- [xxx, xxx, xxx, xxx] 表示一个序列
- {"KEY": "VALUE", "_key": "@vAlUe"} 表示键值对，除格式明确指明 预留 的键值对，其余可自由设定，留给工具一些特殊场景自由发挥的空间
- 本 约定 中未明确描述的部分（如转义等），全部依据 [RFC 7195 The JavaScript Object Notation \(JSON\) Data Interchange Format](#)

2 项目打包结构

2.1 概述

项目打包结构选用 zip 压缩格式，主要因为可采用以下几点优势

1. 良好的通用性
2. 借助其所支持的目录结构，实现信息的分类
3. 借助其信息压缩算法，实现项目信息的高效存储和交换

但是因为 zip 压缩格式有一定的限制，例如文件名非法字符的限制等，有时不方便直接以文件原始的名称直接命名则需要采用 无实际含义的编号 为其命名，只要和 project.json 的信息能对应上即可，例如

1. SYMBOL 目录下的文件
2. SHEET 目录下的文件夹

2.2 .zip 文件组织方式

当需要进行整个项目打包时，则依照如下文件夹组织规则，生成 .zip 压缩文件

将使用如下目录结构打包

```

ROOT
| - project.json .....// 项目信息文件，具有整个项目的结构性信息
| - SYMBOL .....// 符号模板 及 Block Symbol 数据格式文件夹
| - | - symbol-uuid-1 .....// SYMBOL 原理图库类型文档 数据，文件名用于和 project.json 内 symbols 一节对应，无实际含义
| - | - symbol-uuid-2
| - | - symbol-uuid-3
| - FOOTPRINT .....// 符号模板 数据格式文件夹
| - | - footprint-uuid-1 .....// FOOTPRINT PCB库类型文档 数据，文件名用于和 project.json 内 footprints 一节对应，无实际含义
| - | - footprint-uuid-2
| - | - footprint-uuid-3
| - INSTANCE
| - | - instance-part-1
| - | - instance-part-2
| - BLOB .....// 二进制数据文件夹
| - | - blob-hash1 .....// BLOB 类型文档内容
| - | - blob-hash2
| - SHEET .....// 原理图信息文件夹
| - | - schematic-uuid-1 .....// 用于和 project.json 内 schematics 一节对应的原理图编号，无实际含义
| - | - | - 1 .....// SCH 原理图类型文档 数据，文件名为 Sheet 编号，例如 DX 里 $8I5 中的 8
| - | - | - 3
| - | - | - 8
| - | - schematic-uuid-2
| - | - | - 1
| - | - | - 2
| - | - schematic-uuid-3
| - | - | - 1
| - PCB
| - | - pcb-uuid-1 .....// PCB 类型文档数据
| - | - pcb-uuid-2
| - POUR
| - | - pcb-uuid-1_eid1 .....// PCB 覆铜结果类型文档数据
| - | - pcb-uuid-2_eid2

```

INSTANCE 内是依照 INSTANCE 实例属性类型文档 撰写的 Instance Value 信息，推荐按照层次图底层图的 Sheet 进行分组，但其它分组方式亦可，文件名是自由的，不假设其具有任何关键逻辑含义

2.3 project.json 信息文件

```

{
  "devices": {
    "device-uuid-1": {
      "title": "DV2005",
      "description": "asdfasdfasdf",
      "tags": [],
      "images": [],
      "attributes": {
        "symbol_uuid": "symbol-uuid-1",
        "footprint_uuid": "footprint-uuid-2",
        "manufacture": "LCSC",
        "value": "10uF"
      }
    },
    "device-uuid-2": {
      "title": "DSSD122",
      "description": "asdfasdfasdf",
      "tags": [],
      "images": [],
      "attributes": {
        "symbol_uuid": "symbol-uuid-2",
        "footprint_uuid": "footprint-uuid-3",
        "manufacture": "LCSC",
        "value": "30uF"
      }
    },
    "device-uuid-3": {
      "title": "S8022",
      "description": "asdfasdfasdf",
      "tags": [],
      "images": [],
      "attributes": {
        "symbol_uuid": "symbol-uuid-3",
        "footprint_uuid": "footprint-uuid-1",
        "manufacture": "LCSC",
        "value": "20uF"
      }
    }
  },
  "symbols": {

```

```

"symbol-uuid-1": {
    // 与 SYMBOLS 目录里对应的文件名称一致
    "title": "symbol1", // Symbol 名称
    "source": "", // Symbol 来源: 如果是工程库则固定留空
    "version": "", // Symbol 来源版本号: 如果是工程库则固定留空
    "type": 17, // Symbol 类型编号, 详见【Symbol 类型】一节
    "desc": "TI Memory", // Symbol 说明, 可留空, 如果不是工程库则固定留空
    "tags": ["Memory"] // Symbol 分类标记, 可留空, 如果不是工程库则固定留空
},
"symbol-uuid-2": {
    "title": "resister:ABcDEfg1",
    "source": "resister:abcdeFG1",
    "version": "4567",
    "type": 22,
    "desc": "",
    "tags": []
},
"symbol-uuid-3": {
    "title": "hw_builtin:SHEET(A3)",
    "source": "hw_builtin:sheet(A3)",
    "version": "5678",
    "type": 20,
    "desc": "",
    "tags": []
}
},
"footprints": {
    "footprint-uuid-1": {
        // 与 SYMBOLS 目录里对应的文件名称一致
        "title": "0805", // Footprint 名称
        "source": "", // Footprint 来源: 如果是工程库则固定留空
        "version": "", // Footprint 来源版本号: 如果是工程库则固定留空
        "type": 17, // Footprint 类型编号, 详见【Footprint 类型】一节
        "desc": "TI Memory", // Footprint 说明, 可留空, 如果不是工程库则固定留空
        "tags": ["Memory"] // Footprint 分类标记, 可留空, 如果不是工程库则固定留空
    },
    "footprint-uuid-2": {
        "title": "asdf",
        "source": "asdf",
        "version": "4567",
        "type": 22,
        "desc": "",
        "tags": []
    },
    "footprint-uuid-3": {
        "title": "fdsa",
        "source": "fdsa",
        "version": "5678",
        "type": 20,
        "desc": "",
        "tags": []
    }
}
},
"schematics": {
    "schematic-uuid-1": {
        // 与 SHEET 目录下的原理图名称一致
        "name": "Schematic1", // 原理图名称
        "sheets": [
            // 原理图下所有 Sheet 的信息, Sheet 出现的顺序应与左侧树显示的顺序一致
            {
                "id": 1, // Sheet 编号, 与 SHEET 目录下/对应原理图下/对应的文件名一致, 例如 DX 里 $8I5 中的 8
                "name": "1" // Sheet 显示的名称
            },
            {
                "id": 3,
                "name": "A"
            },
            {
                "id": 8,
                "name": "3"
            }
        ]
    },
    "schematic-uuid-2": {
        "name": "Schematic2",
        "sheets": [
            {
                "id": 1,
                "name": "1"
            }
        ]
    }
}

```

```
{
  {
    "id": 2,
    "name": "2"
  }
}

},
"pcbs": {
  "pcb-uuid-1": "PCB Title 1",
  "pcb-uuid-2": "AAbbCCd"
},
"boards": {
  "Board1": {
    "schematic": "schematic-uuid-1",
    "pcb": "pcb-uuid-1"
  },
  "Board2": {
    "schematic": "schematic-uuid-2"
  }
},
"config": {
  "title": "Project3", // 工程名称
  "defaultSheet": "device-uuid-3", // 默认图框配置
  "cbbProject": false // 是否 CBB 工程, 如果是 CBB 工程, 则导入时需要检查 CBB 工程的基本要求
}
}
```

3 通用文件头格式

["DOCTYPE", "SCH", "1.0"]

1. 文档类型标识: DOCTYPE
2. 文档类型: SCH SYMBOL INSTANCE PCB FOOTPRINT PANEL
3. 文档格式版本号, 每次发布设计格式变动的版本前, 都要调整版本号
 - o 文件格式带上版本号, 方便通过版本号做前向兼容, 不需要每一个图元都要做前向兼容
 - o 例如将 ARC
 - o ["ARC", "e5", 340, 210, ...]
 - o 调整成
 - o ["ARC", "e5", [340, 210], ...]
 - o 只需要调整版本号就好, 不需要在格式上预留以前的位置

["HEAD", { "editorVersion": "4.7.8", "importFlag": 0 }]

1. 文档头标识: HEAD
2. 内部参数: Key-Value: 为编辑器附加信息, 用于数据分析等功能, 可选

4 二进制数据 BLOB

["DOCTYPE", "BLOB", "1.0"]

1. 文档类型标识: DOCTYPE
2. 文档类型: BLOB
3. 文档格式版本号, 每次发布设计格式变动的版本前, 都要调整版本号

["BLOB", "blob-hash-id1", "aaa.png", ""]

1. 二进制数据标识: BLOB
2. 二进制哈希码: 计算方式
 1. 最终二进制数据字符串, 使用 UTF-8 编码
 2. 使用 SHA-256 计算哈希值
 3. 将哈希值使用 HEX 编码成十六进制字符串, 全小写
3. 文件名: 仅用于参考, 可留空
4. 二进制数据: 使用类似 Data URLs 的规范, 但有区别
 1. 一般格式, 与 Data Urls 完全兼容 data:[<mediatype>][;base64],<data>
 - 如 
 - 如 data:text/html,<html></html>
 - 一般情况下, 应尽可能使用一般格式, 方便直接使用 Web API 加载, 不需要算法转换
 2. 扩展格式, 扩展了其功能性, 加上了如 gzip/deflate 等编码转换功能 data:<mediatype>[pipeline],<data>
 - 如 data:text/html;gzip;base64,asdfasdf
 1. 先将 asdfasdf 进行 base64 解码
 2. 再用 gzip 解压缩
 3. 最后才以 text/html 去加载
 - 如 data:text/css;deflate;aes128;base64,aaaaaaa
 1. 先将 aaaaaaa 进行 base64 解码
 2. 再用 aes128 解密 (具体加解密逻辑待定)
 3. 再用 deflate 解压缩
 4. 最后才以 text/css 去加载此数据

["BLOB", "blob-hash-id2", "test.xls", "data:application/vnd.ms-excel;base64,xxsasdfawerwerqwer"]

5 修订日志

5.1 2022030201

添加 BLOB 类型文档

5.2 2020102801

添加导入工程结构 POUR 文件夹

5.3 2020101201

project.json 添加 pcbs 一节

5.4 2020100901

根据专业版业务调整导入结构

5.5 2020092801

合并 PCB 和原理图公共信息