RPU Designer使用手册

目录

[1. 引言 4](#_Toc69979480)

[1.1. 术语解释 4](#_Toc69979481)

[1.2. 图例 5](#_Toc69979482)

[2. 软件概述 6](#_Toc69979483)

[2.1. 软件简介 6](#_Toc69979484)

[2.2. 运行环境 6](#_Toc69979485)

[2.3. 软件目录结构 6](#_Toc69979486)

[3. 开始使用 7](#_Toc69979487)

[3.1. 软件安装 7](#_Toc69979488)

[3.2. 启动软件 7](#_Toc69979489)

[3.3. 主界面介绍 7](#_Toc69979490)

[3.4. 首页界面 8](#_Toc69979491)

[3.5. 解决方案资源管理器 9](#_Toc69979492)

[3.6. 算法编辑 9](#_Toc69979493)

[3.7. 资源编辑 10](#_Toc69979494)

[3.8. 运行算法 11](#_Toc69979495)

[3.9. 调试算法 12](#_Toc69979496)

[3.10. 编译生成配置码 13](#_Toc69979497)

[4. 算法配置示例 13](#_Toc69979498)

[4.1. 步骤1：新建项目和算法配置图 13](#_Toc69979499)

[4.2. 步骤2：新建项目 13](#_Toc69979500)

[4.3. 步骤3：新建算法文件并打开 14](#_Toc69979501)

[4.4. 步骤4：添加功能元素 14](#_Toc69979502)

[4.5. 步骤4：添加连线 15](#_Toc69979503)

[4.6. 步骤5：设置属性 16](#_Toc69979504)

[4.7. 步骤6：添加资源文件 17](#_Toc69979505)

[5. 使用技巧 17](#_Toc69979506)

**文档管理信息表**

|  |  |
| --- | --- |
| **主题** | RPU Designer使用手册 |
| **版本** |  |
| **内容** |  |
| **关键字** | 使用手册 |
| **创建时间** | 2021年04月16日 |
| **创建人** | mc |
| **最新发布日期** | 20--年0-月--日 |

**文档变更纪录**

|  |  |  |
| --- | --- | --- |
| **更改人** | **日期** | **更改内容** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**文档主要评审意见**

|  |  |  |
| --- | --- | --- |
| **评审人员** | **日期** | **意见** |
|  |  |  |
|  |  |  |
|  |  |  |

# 引言

本说明书是《RPU集成开发设计工具使用说明书》（以下简称RPU Designer使用说明书），主要用于指导用户使用RPU集成开发设计工具（以下简称RPU Designer）的算法配置、算法统计分析、仿真模拟、硬件调试、生成固件、OTP信息配置等，以使用户了解工具链的适用范围和使用方式，也为维护人员日后的维护工作提供参考。

## 术语解释

RPU Designer使用说明书中用到的所有术语如表 1‑1所示：

表 1‑1 术语解释表

|  |  |
| --- | --- |
| **简称/术语/程序** | **解释** |
| 微纳实验室 | 清华大学无锡应用技术研究院微电子与纳电子实验室 |
| IDE | 集成开发环境 |
| RPU | 可重构处理器 |
| RCA | 可重构单元阵列 |
| RCU-CFG | 行运算单元 |
| RC | 可重构单元，通常称为算子 |
| BFU | RC算子的一种，具有AU、LU、SU三种模式 |
| SBOX | RC算子的一种，具有查表置换功能 |
| BENES | RC算子的一种，具有128bit-128bit置换功能 |
| MEM | 寄存器 |
| RCH | 移位寄存器 |
| IMD | 立即数寄存器 |
| FIFO | 先入先出寄存器 |
| IFIFO/OFIFO | 输入/输出先入先出寄存器 |
| SIM | 仿真器 |
| EMU | 模拟器 |
| 解决方案 | RPU算法解决方案，通常是一种加密算法不同模式的集合 |
| 项目 | 解决方案的子项，通常是某种加密算法的一种模式 |
| 算法文件 | RPU芯片内部阵列的描述文件，通常一个项目内有多有算法文件 |
| rpudesigner | 可执行应用程序，RPU Designer的主程序 |
| maker | 可执行应用程序，固件签名程序 |
| cmdgen | 可执行应用程序，配置码生成程序 |
| rcs | 可执行应用程序，软件仿真程序 |

## 图例

RPU Designer使用说明书中用到的图例如下所示：

类型 实例

系统程序 主界面

按钮 确定

菜单 【文件→打印】

对话框 『用户登录』

窗口 ［索引窗口］

输入信息 “rpudesigner”

选项/参数名 用户组

选项/参数值 系统

目录 根目录

节点/终目录 统计图

工具栏 〖当前模式〗

图标 「保存」se_save_mw

页签 〈图形列表〉

# 软件概述

## 软件简介

RPU集成开发设计工具（以下简称RPU Designer）是清华大学微电子与纳电子实验室（以下简称微纳实验室）开发的具有自主知识产权的可重构加解密芯片集成开发设计工具。RPU Designer具有内置的、充分融合RPU的各种应用功能，为可重构芯片算法配置、分析和配置码编译提供成熟有效的解决方案。

用户使用本软件，可以方便对RPU进行重构配置，同时支持软件仿真调试和硬件联调，还具备烧写固件等功能。

## 运行环境

RPU Designer对主流的操作系统编程接口进行了高性能封装，屏蔽了底层硬件和操作系统的具体差异，从而保证上层应用获得更好的灵活性、可靠性和可移植性。

RPU Designer可单独运行于大多数版本的Unix、Windows、Linux系统，也支持Unix、Windows、Linux操作系统的异构、混合模式运行。目前已经测试通过的操作系统包括：Sun Solaris 8以上、HP-UNIX、IBM AIX5.0以上、Windows 2000、Windows XP、Windows 2003、Linux等。

## 软件目录结构

RPUDesigner

├─bin 可执行程序目录

├─doc 文档目录，存放软件相关文档

├─etc 配置目录，存放软件配置信息

├─image 图片资源目录，存放软件所需的图片资源

├─lib 库目录（unix like）

├─plugins 插件目录

├─script 脚本目录

├─log 日志输出目录，存放软件打印的日志文件（windows平台在用户目录下）

├─tmp 临时文件目录（windows平台在用户目录下）

└─var 参数变量目录（windows平台在用户目录下）

# 开始使用

## 软件安装

在各个平台下提供一致的软件安装方式，运行安装包开始安装：

1. Step1：选择安装时使用语言，下一步
2. Step2：阅读许可协议，选择“我接收协议”，下一步
3. Step3：选择安装位置，下一步
4. Step4：选择开始菜单文件夹，下一步
5. Step5：选择附加任务，下一步
6. Step6：确认安装，完成

## 启动软件

软件的可执行程序位于安装目录/bin下，分别有主应用程序rpudesigner、配置码编译器cmdgen、仿真程序rcs。

主应用程序rpudesigner可以带参数启动，用法如下：

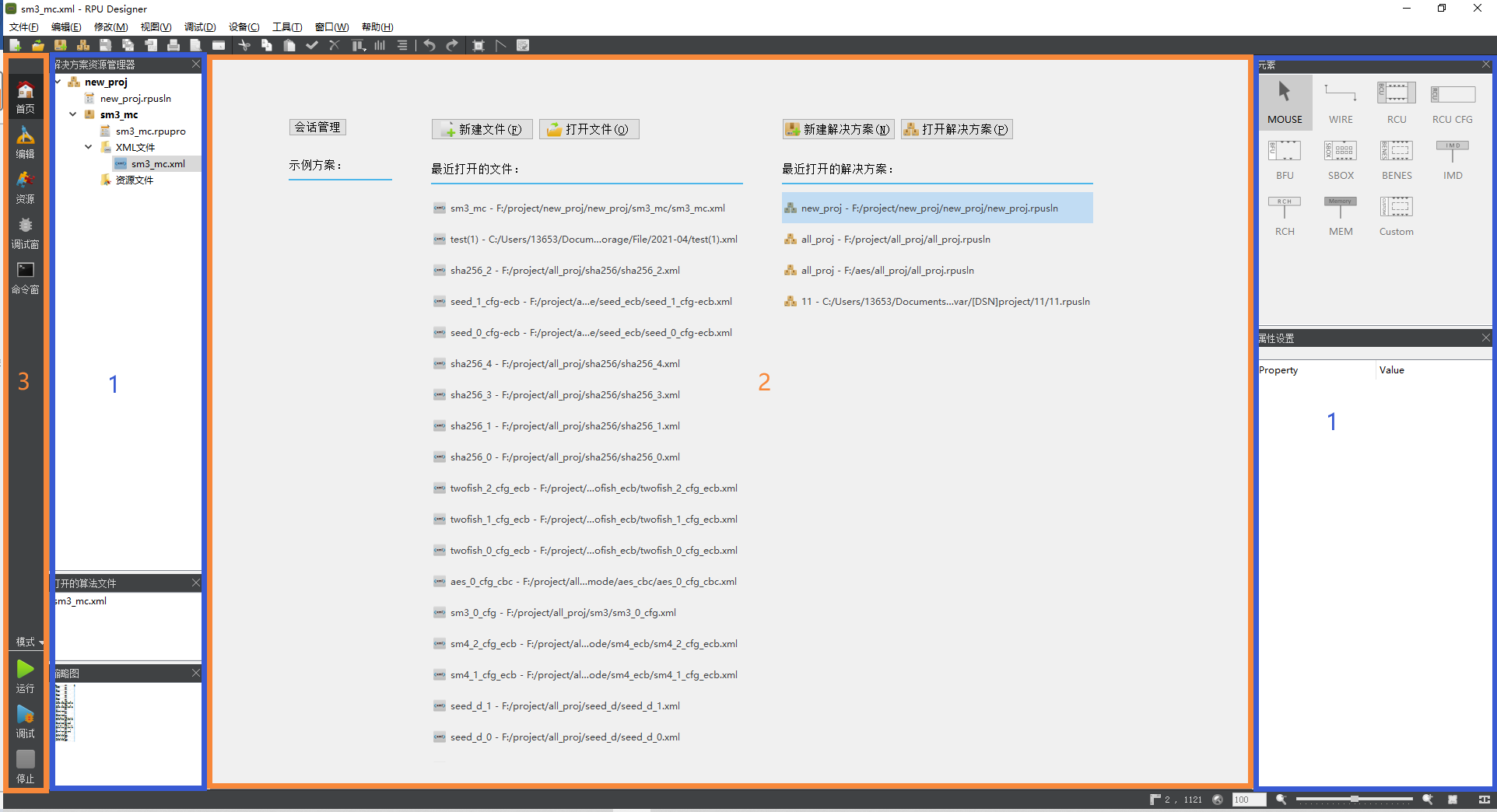
|  |  |
| --- | --- |
| Usage: ./rpudesigner [options] | |
| -h, --help | Displays this help. |
| -v, --version | Displays version information. |
| --client | Attempt to start a new program without check already running first instance. |
| --hidpi <enable/disable> | Enable high DPI device,default is enabled. |
| --admin | Run as admin user. |

## 主界面介绍

RPU Designer主界面如下图所示，可以分为3部分功能区。

如下图中3区是功能导航条，包括5个功能区：欢迎界面、算法编辑、资源编辑、调试、运行功能区的导航。这些功能也是用户配置、调试算法过程中主要操作的功能。

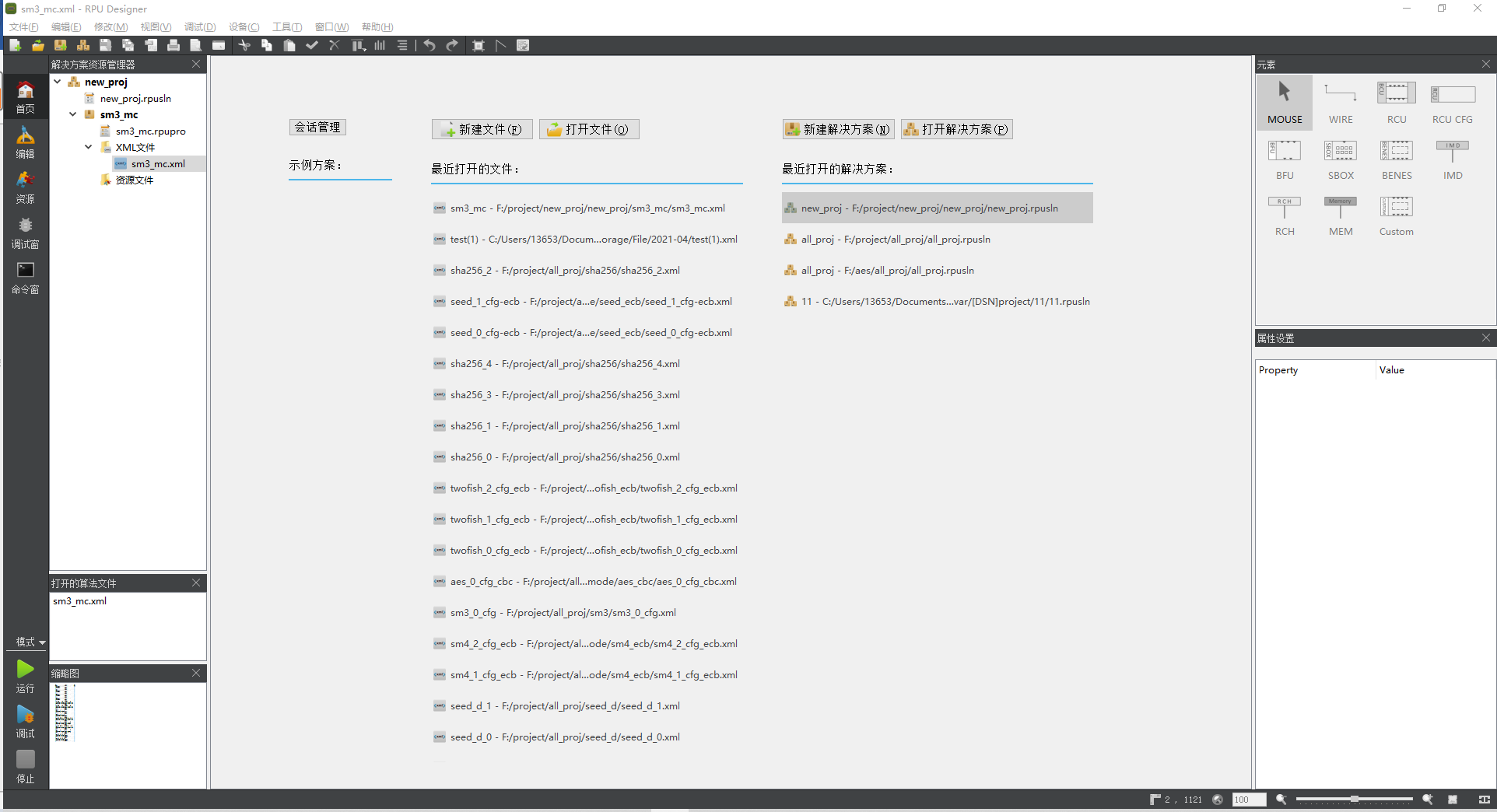
2区是5大功能区的主要的显示编辑界面，用户通过功能导航条来切换不同的显示编辑界面，如下图中2区是欢迎界面的快捷按钮。

1区是算法编辑、资源编辑的功能停靠窗，使用者通过操作此界面来管理算法解决方案、编辑算法文件。

在使用RPU Designer配置算法过程中，使用者主要围绕主界面、算法编辑界面、资源编辑界面、项目配置、调试界面、运行结果界面操作。

## 首页界面

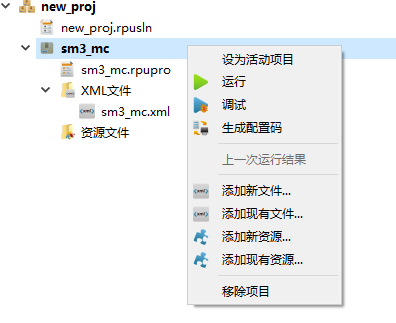
打开方式：〖导航工具条：首页〗

欢迎界面上有新建文件、打开文件、新建解决方案、打开解决方案快捷操作，并列出最近打开文件的文件和解决方案，如下图。

## 解决方案资源管理器

打开方式：【视图→停靠窗→显示解决方案资源管理器】

解决方案资源管理器是一个树状图形界面，陈列用户已打开的解决方案，解决方案资源管理器负责管理所有打开的解决方案下的算法文件、资源文件和项目配置信息，包括解决方案下项目管理，项目下算法文件的打开、增加、删除，资源文件的打开，增加，删除等。

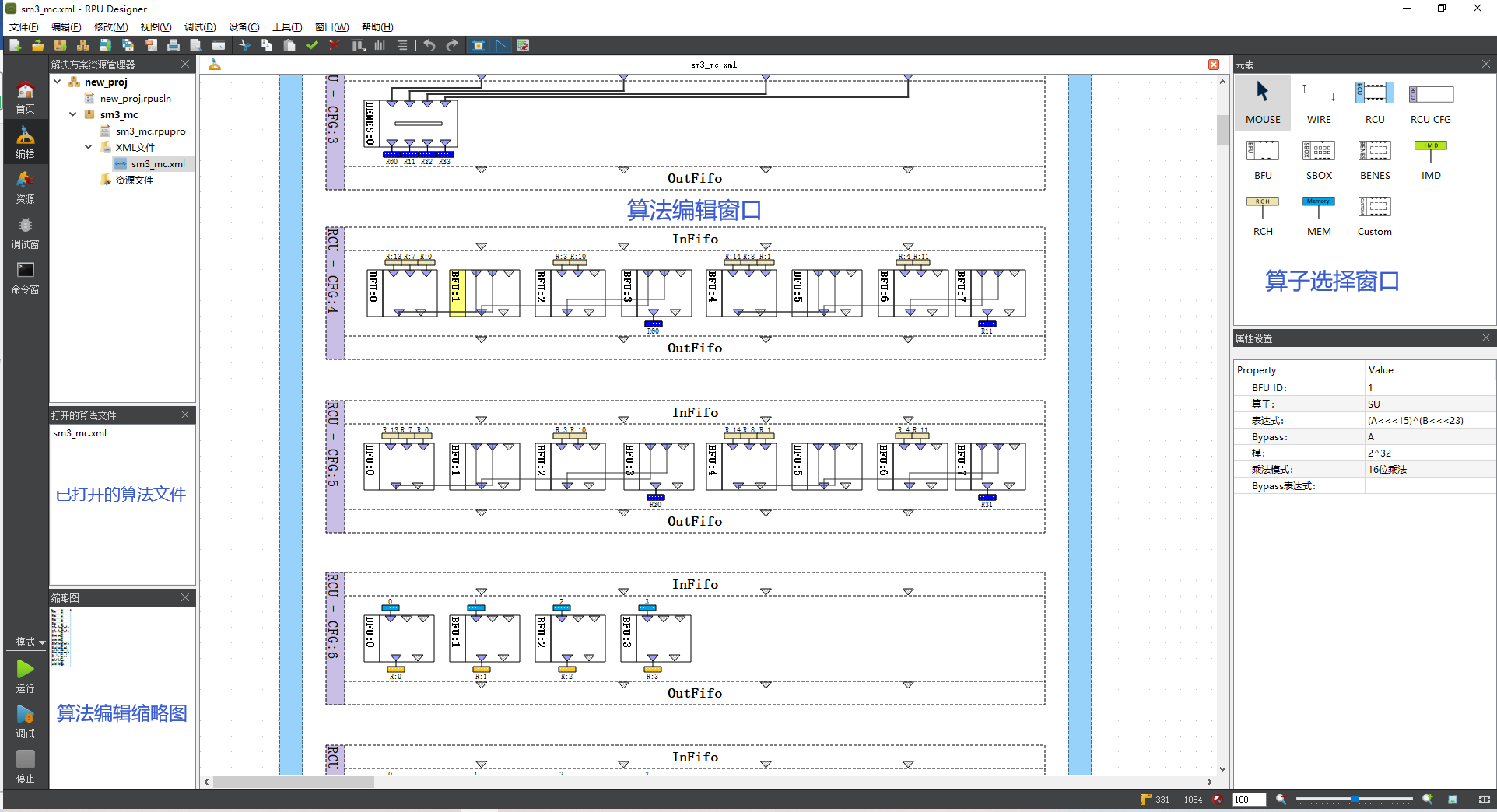
解决方案资源管理器下有且只有一个当前活动的项目，项目的配置、调试与运行都围绕当前活动的项目进行，可以通过上下文菜单\双击项目文件切换当前活动的项目。

## 算法编辑

打开方式：

* 〖导航工具条：编辑〗
* 『欢迎窗口→打开文件→*浏览打开文件*』
* 『欢迎窗口→打开文件→最近打开的文件→*选择文件*』
* 『打开的算法文件→*选择文件*』
* 『解决方案资源管理器→*双击/右击打开文件*』
* 【窗口→*选择打开的文件*】
* 【文件→打开文件】
* 【文件→最近访问的文件→*选择文件*】

算法编辑界面是RPU Designer的核心界面，在此完成RPU算法的详细配置，包括算子的连接信息、算子内部配置。

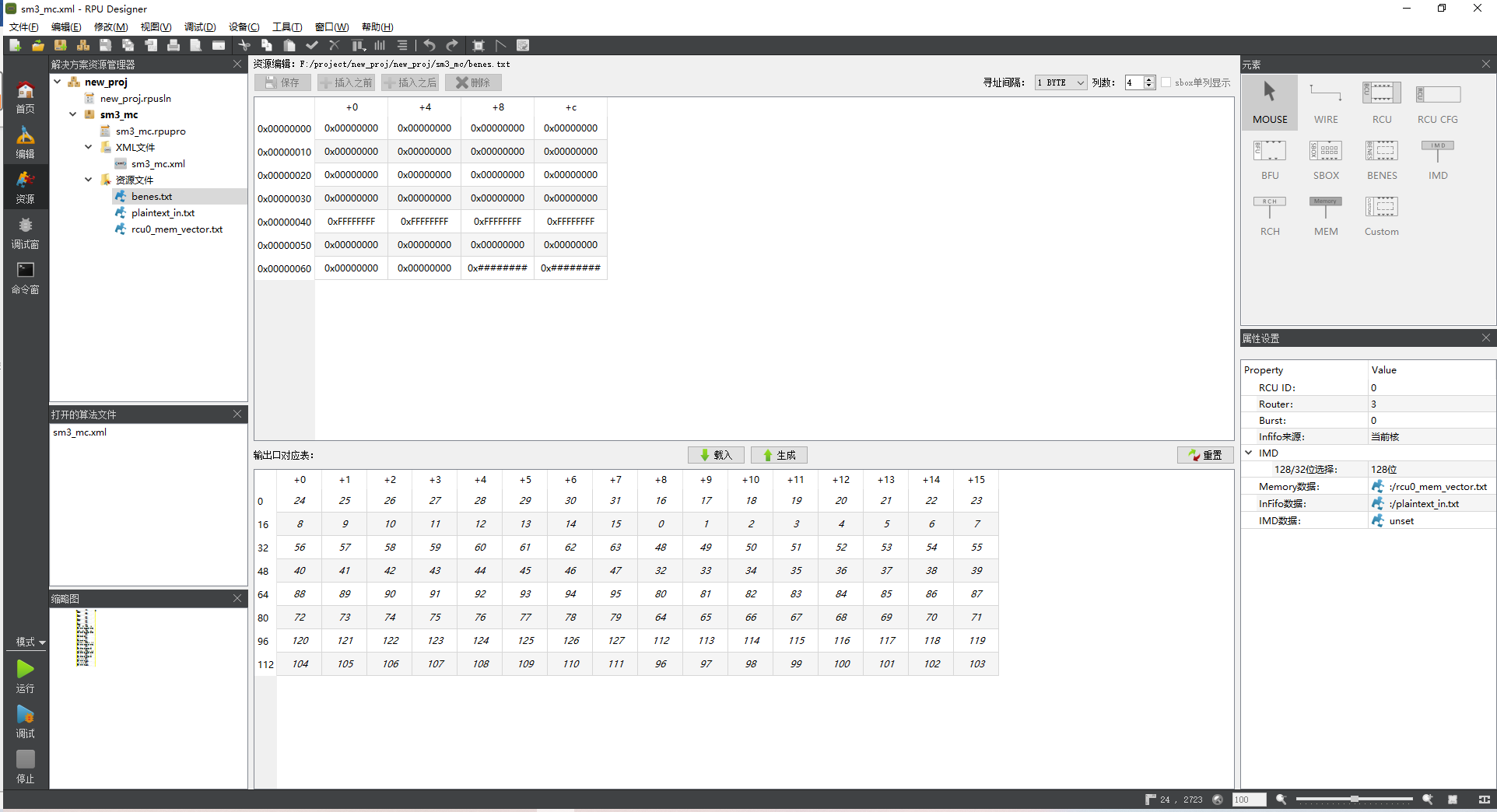
通过选择『元素』窗口中的不同功能来操作编辑『算法编辑』窗口算法配置，如在『元素』窗口中选择「WIRE」元素，此时「WIRE」元素图标处于选中状态，点击算子输出口即可开始连接算子连线。

## 资源编辑

打开方式：

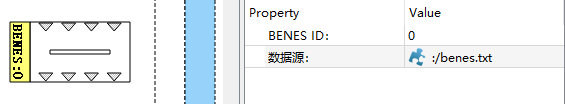
* 〖导航工具条：资源〗
* 『解决方案资源管理器→*双击/右击打开资源文件*』

RPU Designer资源种类有：SBOX、BENES、FIFO、Memory四种，分别对应RPU内部使用的数据。

如下图是BENES资源编辑界面，可通过功能区改变显示方式，如寻址间隔、列数等。

项目资源可在算法编辑、项目配置时选择使用。

* 算法编辑使用资源：

能使用资源的算子有SBOX、BENES和IMD，对应使用SBOX资源、BENES资源和IMD资源。选择使用资源：『算法编辑→单击SBOX』，此时属性设置会显示算子配置窗口（如下图），『算子配置→数据源』选择对应的资源文件。

## 运行算法（暂不支持）

打开方式：

* 〖导航工具条：运行〗
* 『解决方案资源管理器』项目文件上下文菜单「运行」
* 『解决方案资源管理器』项目文件上下文菜单「上一次运行结果」
* 【调试→运行】
* F5

算法配置完成后，可运行此算法来验证算法的正确性，运行结束后会打开运行结果界面，如下图是aes\_cbc加密算法的运行结果：

## 调试算法（暂不支持）

打开方式：

* 〖导航工具条：调试〗
* 『解决方案资源管理器』项目文件上下文菜单「调试」
* 【调试→调试】

可以通过调试的方式来debug算法配置，调试功能包含抓取算子数据、单步调试、断点、导出波形等功能，如下图正在调试aes\_cbc算法：

## 编译生成配置码

打开方式：

* 『解决方案资源管理器』项目文件上下文菜单「生成配置码」
* 【调试→生成配置码】

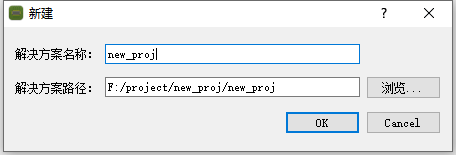
编译生成配置码并没有界面，软件右下角会有当前编译进度，当完成时会打开编译生成文件所在的目录。

编译生成配置码会把『解决方案资源管理器』下所有文件编译打包成配置码。

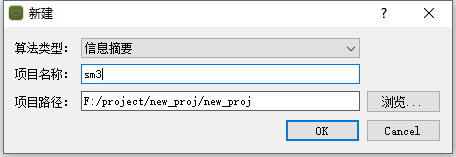
# 算法配置示例

从新建算法开始演示算法配置过程：

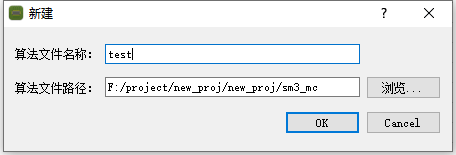
## 步骤1：新建项目和算法配置图

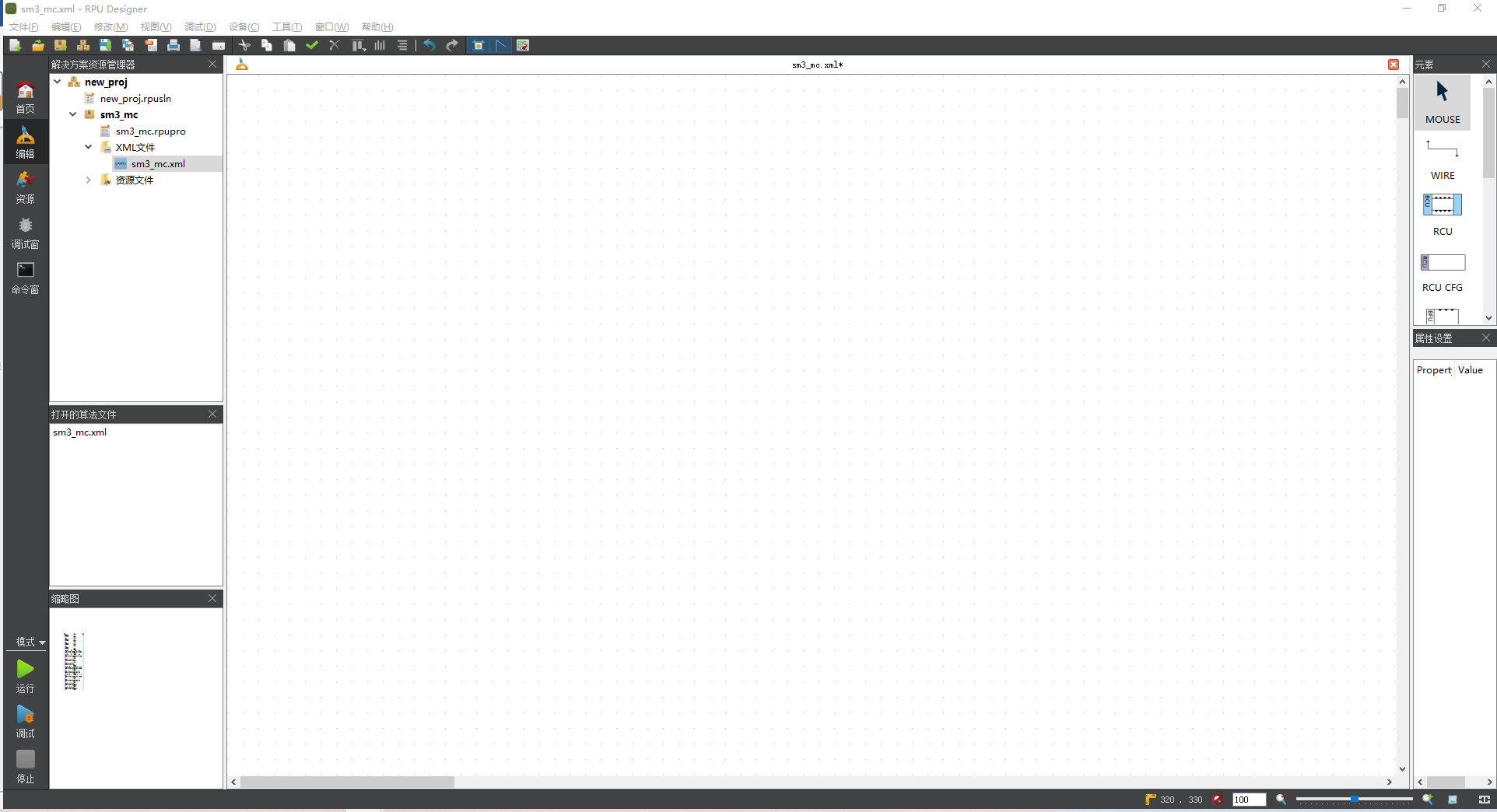
点击新建解决方案→写入解决方案名称→ok

## 步骤2：新建项目

『解决方案资源管理器』解决方案上下文菜单→新建项目→选择算法类型&命名项目名称→ok

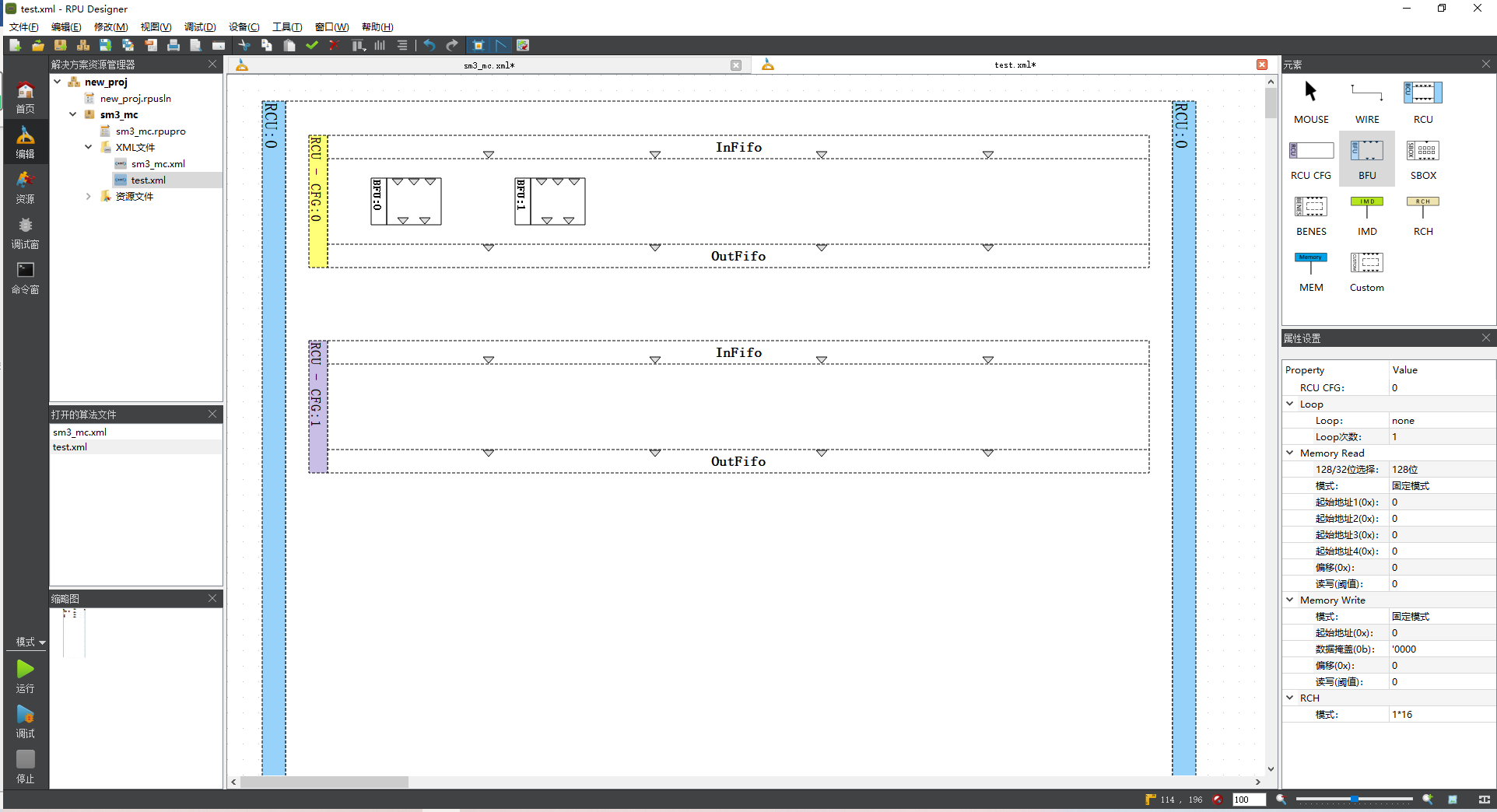
## 步骤3：新建算法文件并打开

『解决方案资源管理器』解决方案上下文菜单→添加新文件→命名算法文件名称→ok

『解决方案资源管理器』双击算法文件，即打开『算法编辑』窗口：

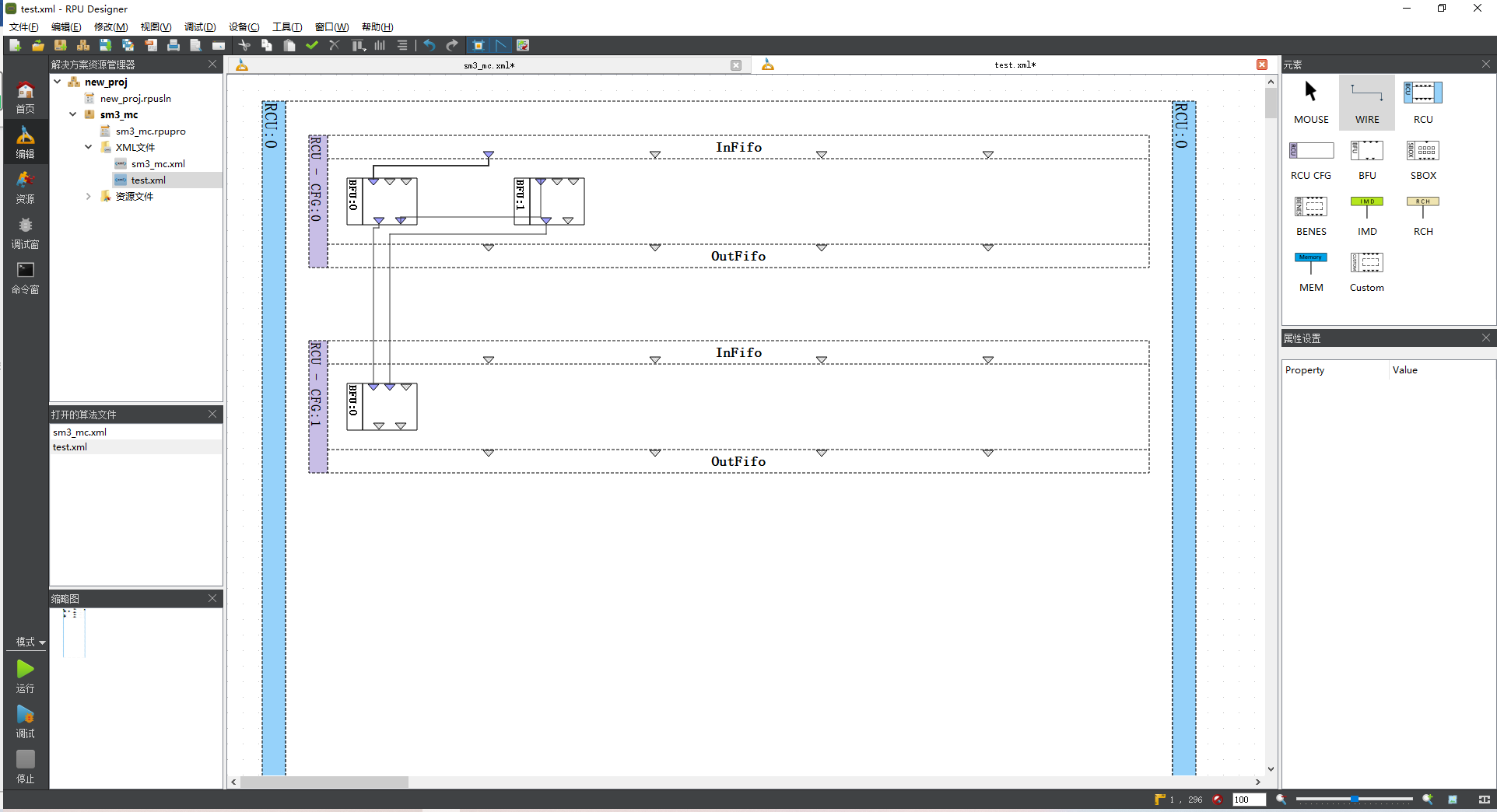
## 步骤4：添加功能元素

『元素』窗口中选择元素，然后在工作区中想要放置的位置点击，将元素添加到算法图中。

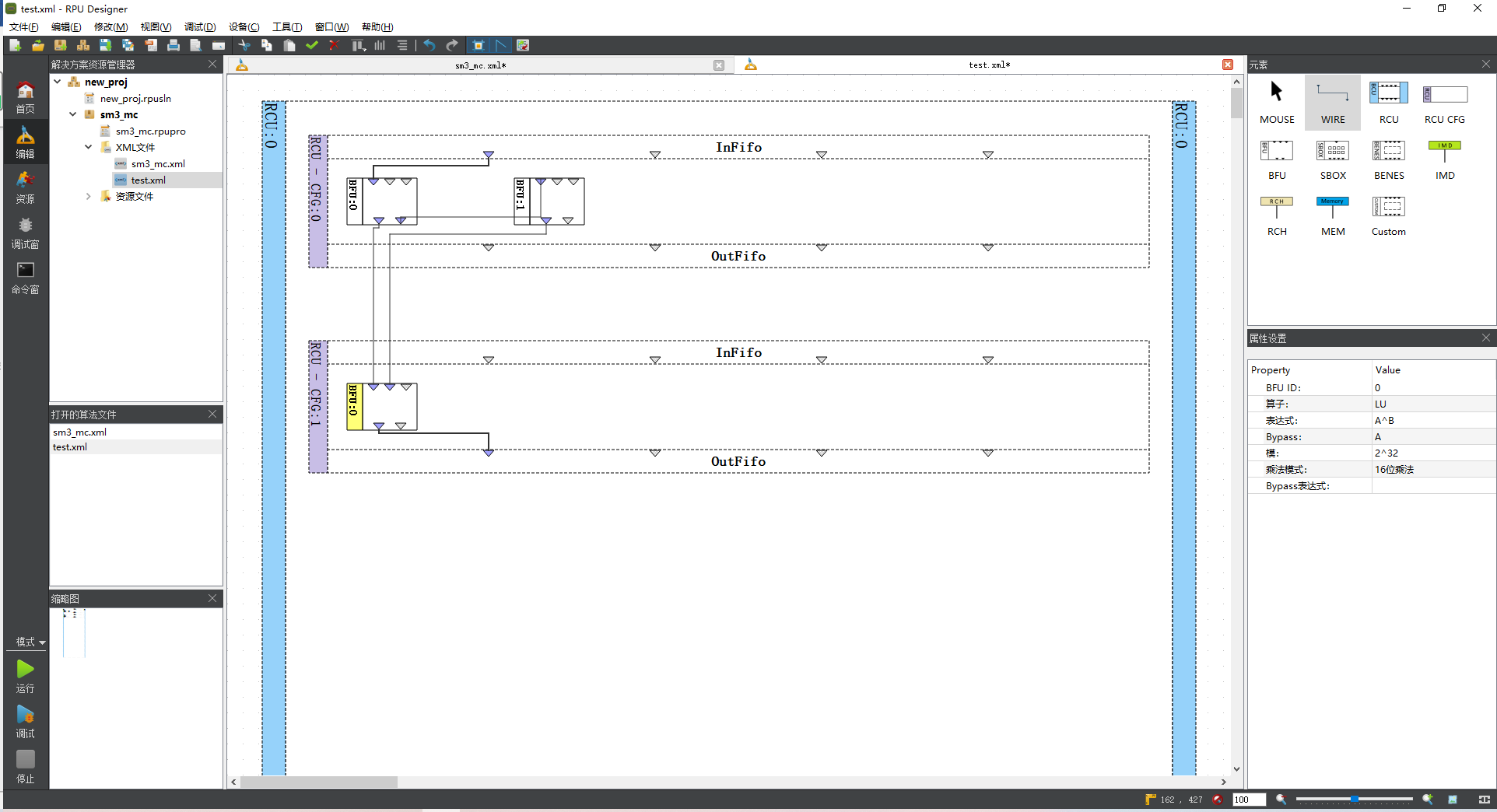
要添加BFU、SBOX、BENES、MEM、IMD、RCH、WIRE等具体算子和元素，必须先将RCU和RCU-CFG添加到子图中。注：算子配置方法与使用限制见rpu特性文档。

## 步骤4：添加连线

『元素』窗口中选择WIRE元素，点击编辑窗口某个算子的输出端口，再点击某个算子的输入端口，如果这两个端口是允许连接的，工作区上会显示出连线

注：在工作区中，算子的输出端口为算子模型下侧的端口，输入端口为算子模型上侧的端口；另外infifo的4个端口为输出端口，outfifo的4个端口为输入端口。算子配置方法与使用限制见rpu特性文档。

## 步骤5：设置属性

『算法编辑』单击某个算子，弹出『算子配置』窗口→设置属性→ok

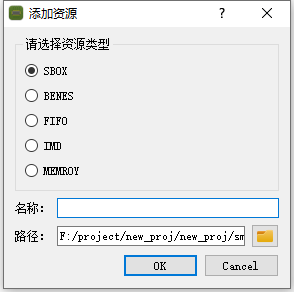
注：

单击BFU、SBOX、BENES算子模型分别设置各自的输入来源和属性；

单击RCU-CFG模型设置该行的memory读写端口和模式，设置该行的循环起止和次数信息；

单击RCU设置当前RCU的Burst和核信息，以及Memory、InFifo和IMD的数据来源文件。

## 步骤6：添加资源文件

『解决方案资源管理器』解决方案上下文菜单→添加新资源→选择资源文件类型&命名资源文件名称→ok

# 使用技巧

* 获取社区共享的解决方案：

【工具→共享社区】打开『共享社区窗口』，下载需要的算法解决方案。

* 共享解决方案到社区：

『解决方案资源管理器』解决方案上下文菜单→「打包并共享」。

* 保存应用程序关闭时状态：

【文件→会话→会话管理】打开『会话管理窗口』，勾选「启动是恢复上次会话」复选框。

* 保存硬件调试状态：

【调试→保存硬件调试状态】。

* 资源快速编辑：『资源编辑窗口』选中要编辑的单元格，ctrl+x剪切，ctrl+c复制，ctrl+v粘贴。
* 打开算法文件、资源文件校验错误

【工具→选项】打开『首选项窗口』→打开与保存→勾选「打开XML时不进行校验」、「打开资源文件时不进行校验」复选框。

* 算法编辑时带参数复制：

选中要选择的算子，【编辑→带连线复制】，快捷键Ctrl+Shift+c

* 算法编辑对齐：

【视图→自动布局】。

* 仿真运行时受最大周期限制：

【工具→选项】打开『首选项窗口』→调试与运行→修改最大周期数。

* 生成配置码后自动编译生成的文件：

【工具→选项】打开『首选项窗口』→生成配置码→完成后执行外部命令。

* 修改默认解决方案路径：

【工具→选项】打开『首选项窗口』→默认解决方案路径→修改默认解决方案路径。