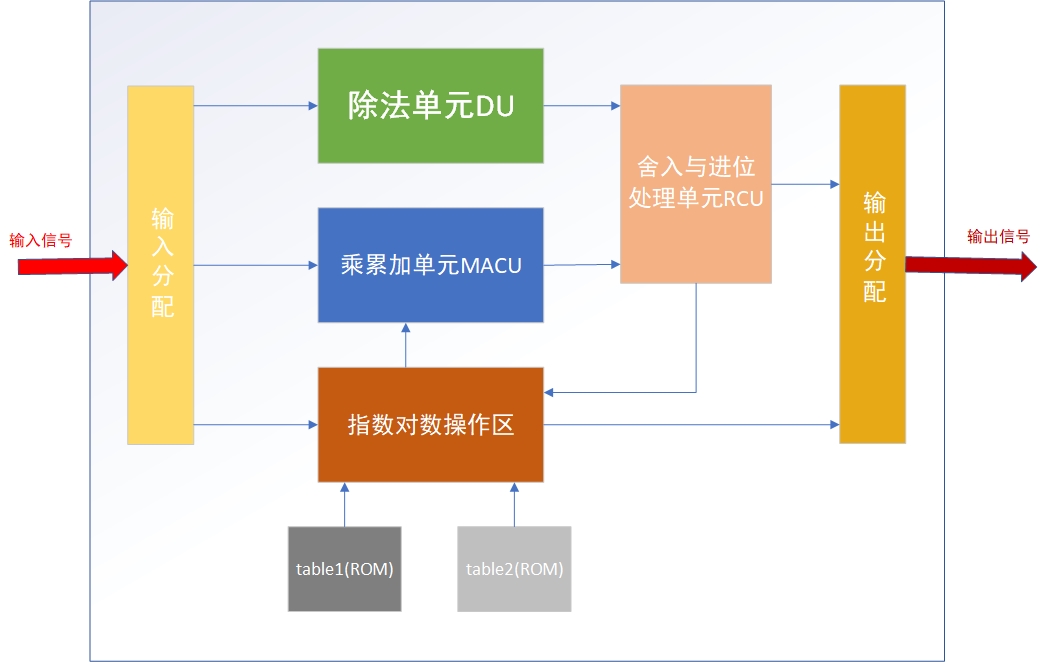
RTL集成文档

结构图如下：



1. 单元作用

乘累加单元MACU实现kx+b的运算，借此可以实现三种运算，置x=1可以做加法；置b=0可以做乘法；不做处理可以做MAC运算

除法单元DU是除法的专用单元

舍入与进位处理单元RCU是本属于除法和乘累加算法中的一部分，专门独立以复用资源

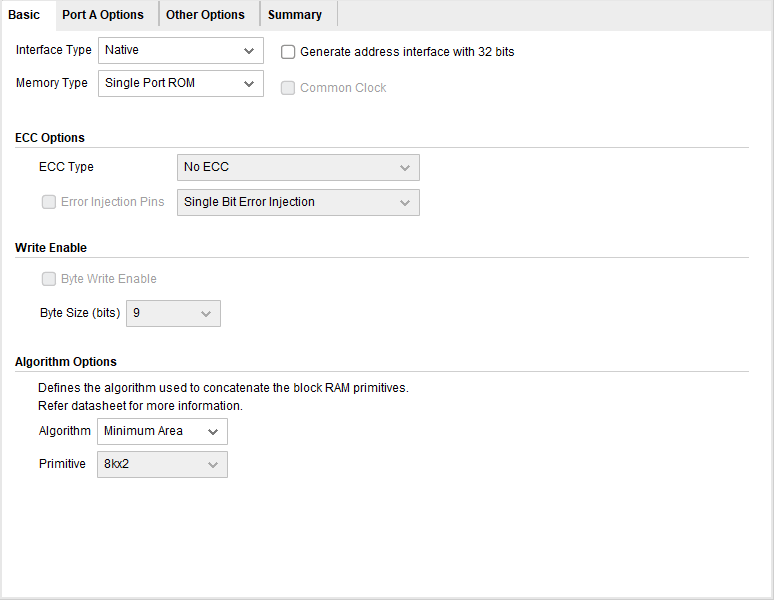
除法和乘累加支持非规格数运算

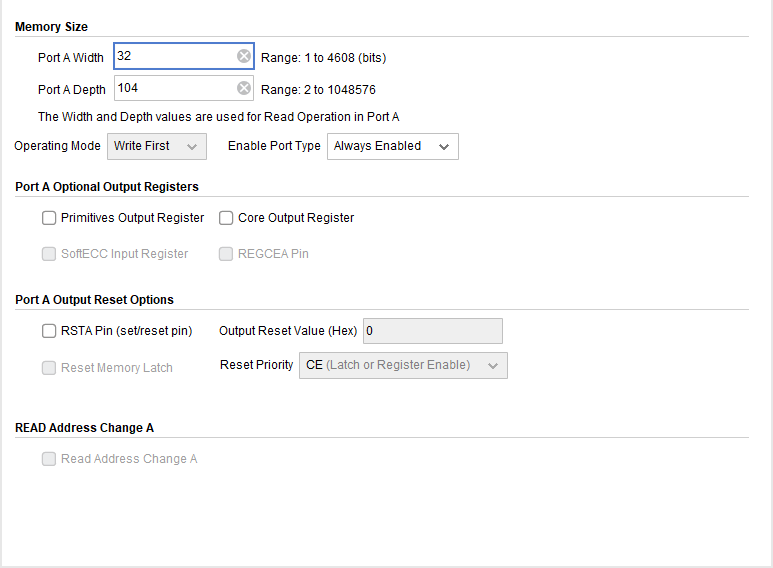
指数对数操作会调用乘累加运算，所以没有单独集成成一个单元，需要在MACU的更上一层以调用MACU，所以指数对数操作区会与MACU和RCU互动。同时指数对数操作区会调用专门的两个表table1和table2

输入分配和输出分配就是根据操作码分配数据流向

1. RTL实现

直接导入.v文件即可（\_tb.v为测试文件），另外两个ROM使用的是IP核，其配置如下：





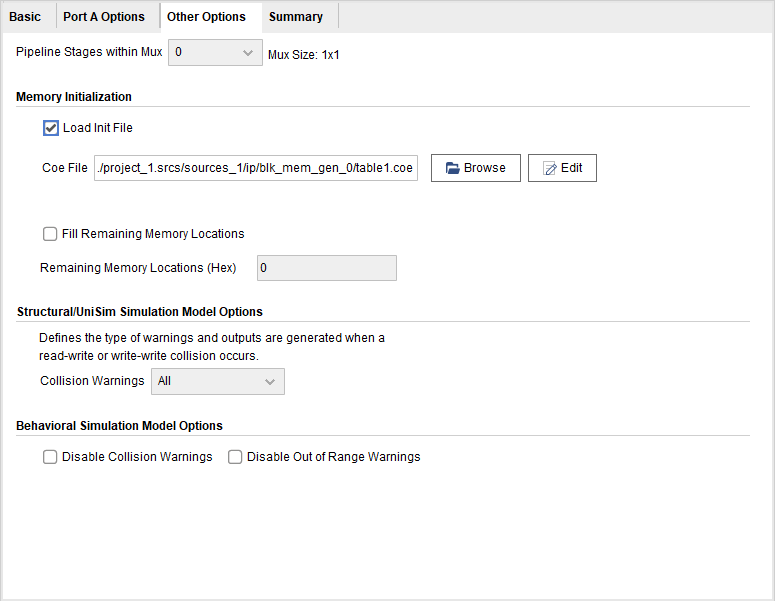


Table1和table2只有数据位宽、深度、导入数据、例化信号不一样。其他注意三个改动点：单端口rom、不要输出寄存器、总是使能

|  |  |  |  |
| --- | --- | --- | --- |
|  | 位宽width | 深度depth | 初始化文件 |
| Table1 | 32 | 104 | table1.coe |
| Table2 | 16 | 256 | table.coe |

**输入端口含义如下**

Clk时钟，上升沿除法

Rst复位，高电平有效

Input\_valid输入有效，高电平有效

Data1、data2、data3 三个操作数

Opcode 操作码

其中，当opcode=0时执行加法，data1、data2作为操作数，流水运行大概12周期；

当opcode=1时，执行乘法，data1、data2作为操作数，流水运行大概12周期；

当opcode=2时，执行除法，data1作为被除数、data2作为除数，状态运行大概39周期；

当opcode=3时，执行乘累加，data1作为k，data2作为x，data3作为b，流水运行大概12周期；

当opcode=4时，执行指数，data1作为操作数，状态运行大概26周期；

当opcode=5时，执行对数，data1作为操作数，状态运行大概25周期；

加乘除支持非规格化数；对数指数只支持合法的规格化数，若不是则会记最接近的合法规格化数计算

**输出端口如下**

data\_o输出数

idle空闲信号，高电平有效

output\_up输出更新信号，高电平有效

1. 测试说明

仿真文件\_tb.v和仿真数据test\_xxx.txt导入即可，要更改仿真的目标算法只需要更改initial块中的opcode初始化值即可，opcode意义与输入意义相同。如遇到错误信息会输出并暂停，仿真结束finish推出仿真

仿真覆盖如下：

加乘除采取边界测试，两个操作数都从[0000-00ff] （被除数为[0001-00ff]），[7b00-7bff]，正负都测

指数对数采取合法输入的全覆盖，即指数为[c8da,8400]，[0400,498c]；对数为[0400,7bff]