1　rtl as blackbox 学习

1.1 学习资料

1）参照 ug902 学习（05/22/2019 Version 2019.1　及以后版本）。

2）在 2019.1 及以上版本 vivado_hls 中有 rtl_as_blackbox 的 Example project。

1.2 如何使用 rtl blackbox

1）需要 c++的完整代码，包括 rtl 模块功能的 c++函数。

2）在运行 c++仿真时，只运行 c++代码；在综合时，hls 会利用 json 文件在 hls 生成的代码中实例化一个我们自己编写的 rtl 模块。

1.3 json 文件

1）json 文件格式见 ug902 表 42，具体写法见 ug902 json 文件示例与 Example project。

2）如果 json 文件错误，比如少‘]’，IDE 不会报错，但综合会不成功。

3）json 文件中的 rtl_resource_usage 的数值在 hls 的综合报告中属于 instance。

Instance	0	1	2	1	0

图 1-1 hls 综合报告-utilization

4）json 文件中"c_port_direction"："out"中的输入输出属性是 c++代码中的输入输出属性。

5）Q：rtl 黑盒数组、FIFO 的传入传出怎么设计 Verilog 代码？

A：设计简单 HLS 代码综合后查看产生的 Verilog 代码学习。

比如数组的使用：

C++函数：

RAM_test(ap_uint<10>in_arr[10], ap_uint<10>out_arr[10])

综合成的 verilog 代码如图 1-2；

数组的输入输出与 ug902 中一致；

不同点：缺少 ap_ctrl_chain 协议中的 module_clock_enable 信号和 ap_ctrl_chain_protocol_continue 信号。

```
module RAM_test (
        ap_clk,
        ap_rst,
        ap_start,
        ap_done,
        ap_idle,
        ap_ready,
        in_arr_V_address0,
        in_arr_V_ce0,
        in_arr_V_q0,
        out_arr_V_address0,
        out_arr_V_ce0,
        out_arr_V_we0,
        out_arr_V_d0
);

parameter    ap_ST_fsm_state1 = 3'd1;
parameter    ap_ST_fsm_state2 = 3'd2;
parameter    ap_ST_fsm_state3 = 3'd4;

input   ap_clk;
input   ap_rst;
input   ap_start;
output   ap_done;
output   ap_idle;
output   ap_ready;
output  [3:0] in_arr_V_address0;
output   in_arr_V_ce0;
input  [9:0] in_arr_V_q0;
output  [3:0] out_arr_V_address0;
output   out_arr_V_ce0;
output   out_arr_V_we0;
output  [9:0] out_arr_V_d0;
```

图 1-1 hls 综合出的数组代码

2　RS BRIEF 的 rtl blackbox

1）在原 c++代码中使用的 linebuffer 窗为 25×25 的 ap_uint<8>数组类型，综合后为 RAM，一个时钟上升沿输入地址，下一个上升沿出数据，计算描述子需要 256×2 个时钟周期。

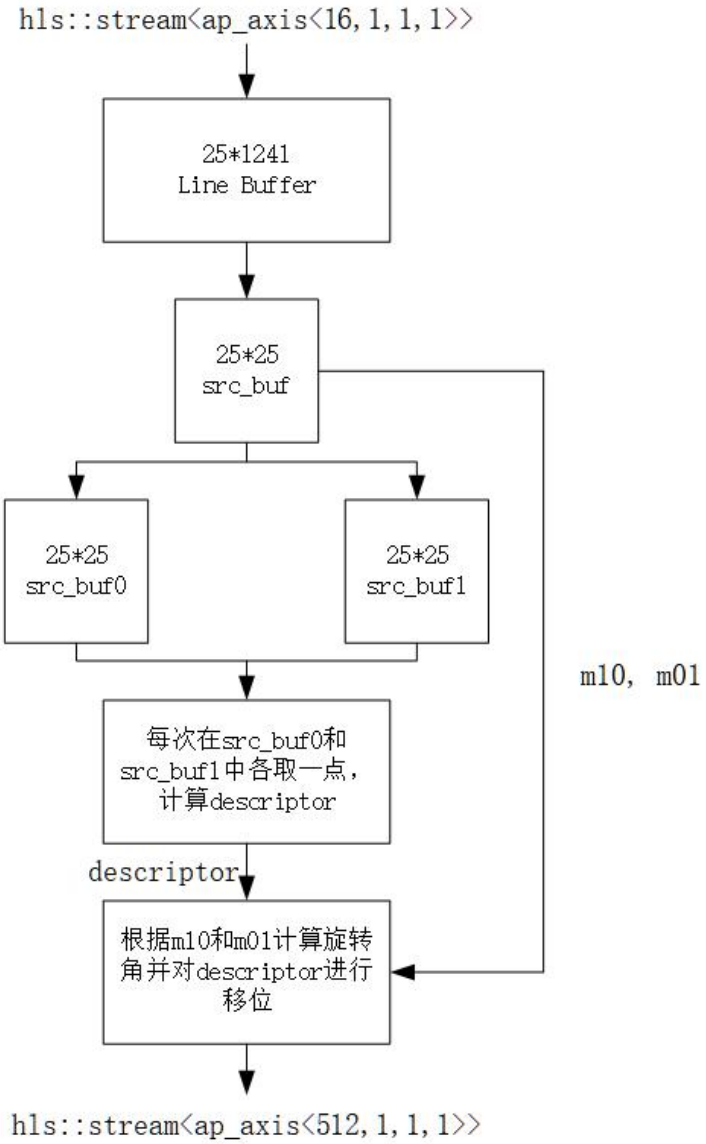

图 2-1 RS_BRIEF 模块图

2）在 c++ 中把 25×25 的 ap_uint<8> 数组修改为一个 ap_uint<5000> data，使用 rtl blackbox 传进 rtl，由于 data 被综合成 reg [4999:0]，可以一个周期出结果。

3 rtl blackbox 可以对于现有功能的 rtl 模块做快速 AXI 接口封装