HLS（High Level Synthesis ）是Xilinx公司推出的最新一代FPGA设计工具。众所周知，FPGA全硬件开发的方案中，直接用硬件语言进行设计，效率会比较低，且不利于调试和仿真。故如果在开发时先通过C/C++/System C对信号或数字处理直接建模，然后通过 Vivado HLS工具将C/C++/System C所建立的模型直接转换为RTL级模型。这样将大大提高FPGA设计的效率，加速了FPGA在高性能信号和数据处理邻域的应用和推广。

好了，废话不那么多了，直接开始今天的正题吧。

实话说，本人在用Vivado HLS之前曾用过类似的高级工具将C/C++转为HDL，故在接触到HLS时并没有显得那么陌生，感觉这类工具差不多。当然Vivado HLS的优势就是整合在Vivado这款软件中，有利于代码的调试且界面更加友好些，当然这款软件和我之前用过的高级综合工具Catapult C在性能上还是有些差距的。

论坛中已经有人将HLS建工程流程完整走了一遍，这里就不再赘述，本次将的是关于HLS中代码一些设计问题。

众所周知，C/C++和HDL虽然在外观上看起来很像，但是从思想上有很大的区别，C语言的串行执行让其能更好让人了解，我们只需从main函数一步一步往下读就行，而HDL的并行代码就很复杂了，所有代码都是同时从触发条件开始执行，所以在C转HDL时就需要注意其语言风格和对应的硬件电路了。举个例子，来看一下C代码中的一些关键属性：

还是最经典的

（1）函数：C语言中函数可以看作是一个一个功能块，然后通过主函数进行调用。函数用于表示设计的层次，当然对于硬件逻辑来说也一样，这不难理解，在HLS中如果没使用内联，一个函数会生成一个.v文件（Verilog开发下）。

（2）参数：也就是函数与函数之间的接口，顶层函数的参数决定了硬件RTL接口的端口，除此之外，还会生成表示逻辑的start，done，idle，ready等端口。

（3）类型：所有变量都需要有对应的类型。在C中主要有整型int，浮点型数double，float等，但是在HDL中都是最基础的二进制数0/1，所以此处设计特别需要注意，不同类型的C对面积和性能能有不同的影响。

（4）循环：C中的循环在HLS中有不同的处理方法，其是影响生成的HDL代码的性能和面积的重要因素。

（5）数组：数组是一种直观且有用的软件结构，能帮助我们更容易的理解C算法，但是在硬件语言中，对数组的访问经常变成性能的瓶颈。默认情况下，C中的数组在硬件语言中对应着RAM。我们在设计的时候可以更改数组的存储方式来提高系统的性能。

（6）操作符：在C代码中的操作符，比如加减乘除，可能要求共享，而且不同的操作符对程序有着性能和面积上的影响。

好了，接下来会说说这些关键属性。

类型：前面说了C语言中那一套在HDL是行不通的，所以这个时候我们就要做出转换了。

在HLS中数据类型主要包括标准C整型类型、任意精度类型和浮点类型三种。

1、标准的C整型类型分为长整型（64位）、短整型（16位）、无符号类型、整型（32位）、字符型（8位）

2、任意精度类型，这是我们在HLS设计最常用的数据类型，也是面积和速度最优的一种数据类型，在此着重讲一下。

（1）任意精度的整数

在HLS中有专门的库函数表示任意整数(1~1024位)“ap\_int.h”，其用法如下：

#include “ap\_int.h”

void int\_arb(…)

{

ap\_int<9> aa;//9位整数位,aa

ap\_uint<10> bb//10位无符号整数bb

}

(2)任意精度的小数

类似，在HLS有将浮点小数转为任意精度的小数也就是定点小数。其使用格式为ap\_[u]fixed<W,I,Q,O,N>该库函数为“ap\_fixed.h”需要注意的是这个库函数只能用于C++，不能用于C。其中W代表数据的位宽，I代表整数位的位宽，Q代表量化模式，O代表溢出模式，N代表在回卷模式下的饱和位数，这里需要注意的就是W，I这两位。其用法如下：

#include “ap\_fixed.h”

void fixed\_arb(…)

{

ap\_fixed<32,16,AP\_RND\_CONV,AP\_SAT> cc;//32位，16位整数位

ap\_ufixed<32,16,AP\_RND\_CONV,AP\_SAT> dd;//32位无符号，16个整数位

}

其中AP\_RND\_CONV为量化模式，表示的是当产生更大的精度范围时，将所得的值四舍五入到最近的值；AP\_SAT为溢出模式，当要求更多的位时，所实现的位饱和。当然还有很多关于量化模式和溢出模式不同的标志位，如果我们在定点设置的位数恰当的情况下，这两种模式的选择对结果影响不大，我们在设计的时候选择这两种就行。

3、浮点类型

当然在一些特殊情况下，需要选用浮点来进行设计，HLS也支持浮点的设计，由于本人没做多少浮点的设计，在此就举个简单例子配上自己的理解下。