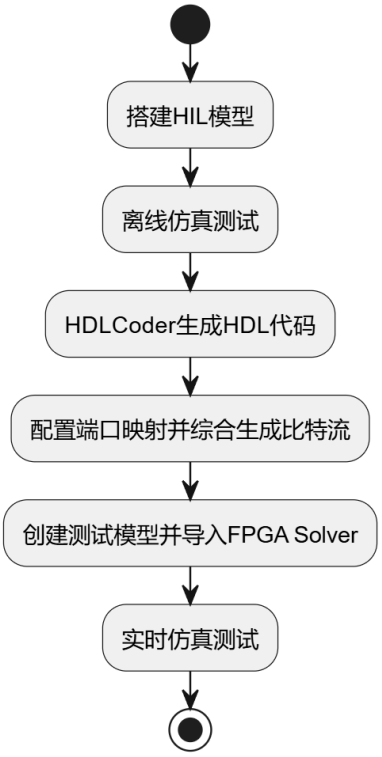
HDLCoder使用方法

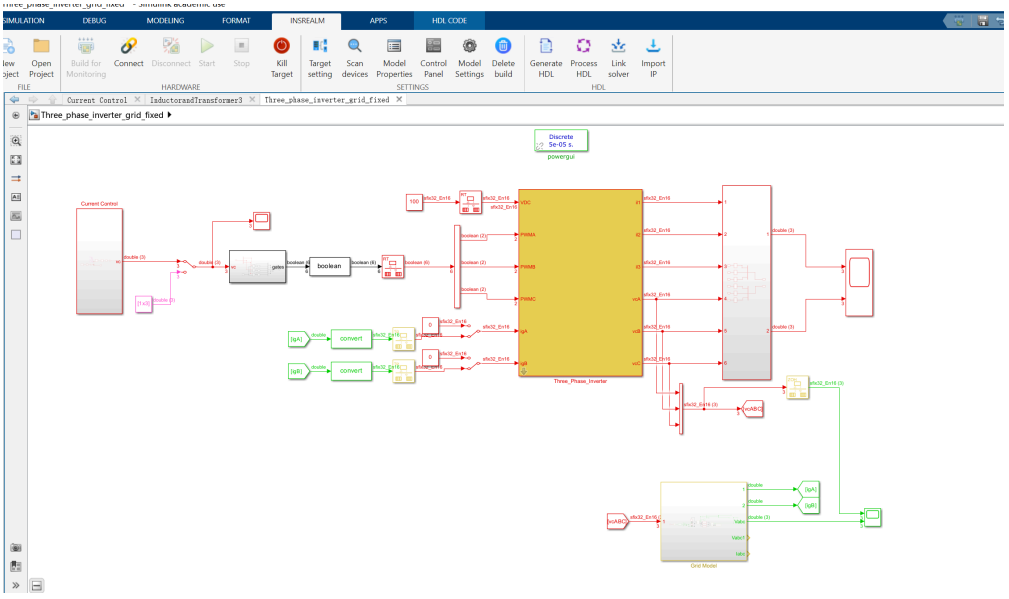
完整的使用流程如下：



1. HIL模型

1.1 模型搭建

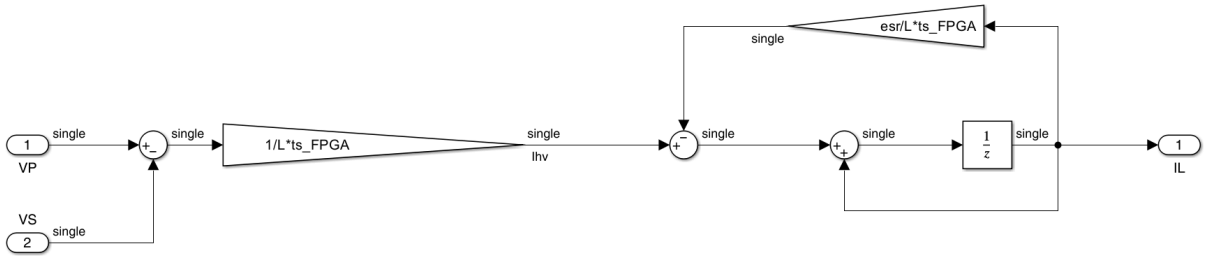
首先需要根据数学方程搭建simulink的模型，比如下面搭建了一个三相逆变器的模型。黄色部分为逆变器本体模型，外部的为控制算法模型。如下所示



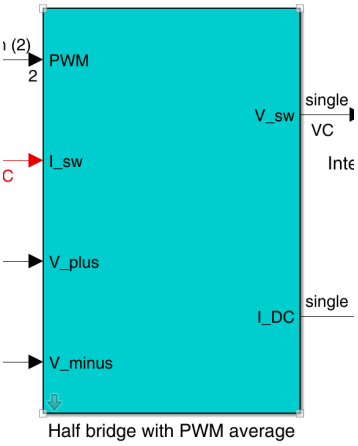
模型内部都是基于电路的数学原理来进行建模。比如RL网络建模，需要基于方程

$$v_L = L\frac{di}{dt} \Rightarrow i_L(t+\Delta t) = i_L(t) + \frac{v_L}{L} * \Delta t$$

上述方程是基于电感的前向欧拉离散方法进行建模。再结合电阻的方程，即可建模，模型则如下图所示



其中 V_P 和 V_N 是RL网络左侧和右侧的电压，ts_FPGA为模型的步长。
其他RC网络也用类似的方法进行建模。开关桥臂则使用提供的模型库进行搭建。 PWM为送入桥臂的上下管PWM信号，I_sw为流出桥臂中点的电流，V_plus和V_minus为桥臂直流侧的正负极电压，V_sw为桥臂中点电压输出，I_DC为直流侧电流。

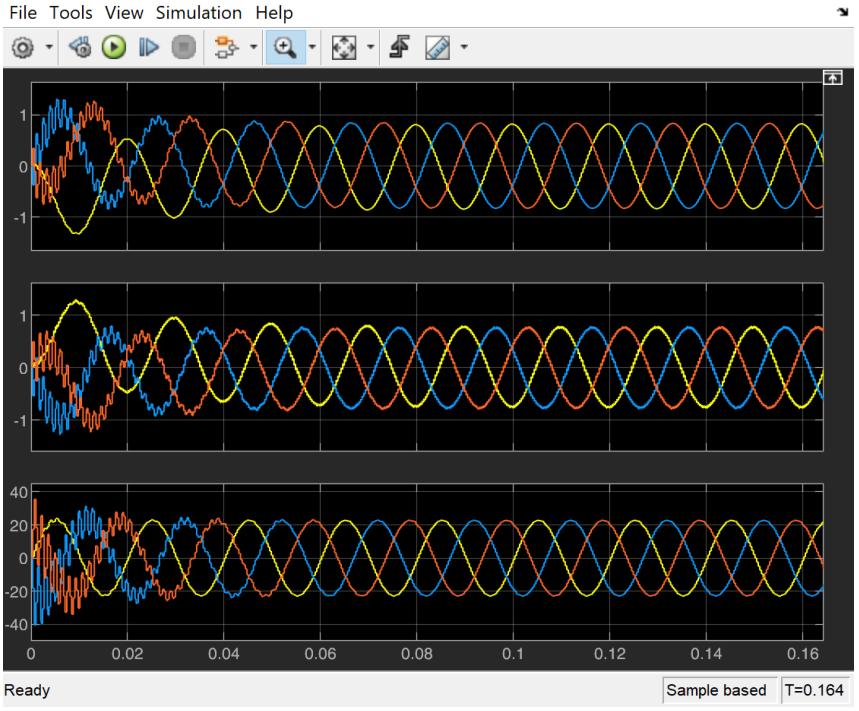


！！！！有几点需要注意！！！！：

1. 电路模型本身需要被放置到一个subsystem内部，所有需要修改的参数作为输入端口，需要查看的波形（包含DAC波形）添加到输出端口。
2. 数据类型配置。
    1. 模型中所有的数据类型需要设置为single类型
    2. 数据端口的数据类型位宽**目前支持最大为32位**，比如可以配置为`fixdt(1,32,16)`，表示32位的有符号定点数，小数部分为16位，或者配置为`single`，即单精度浮点数
    3. 端口的数字IO信号需要配置为`boolean`类型。
    4. 需要连接到dac的需要数据配置为`uint16`类型。
3. 外部输入的数字io信号（比如PWM输入信号）需要添加一个delay模块，以避免亚稳态问题
4. 模型保存的路径不能太长，且不能含有中文路径。否则后续FPGA的比特流生成会出错。推荐路径长度（含模型名字）应该少于40个字符。

1.2 离线仿真验证

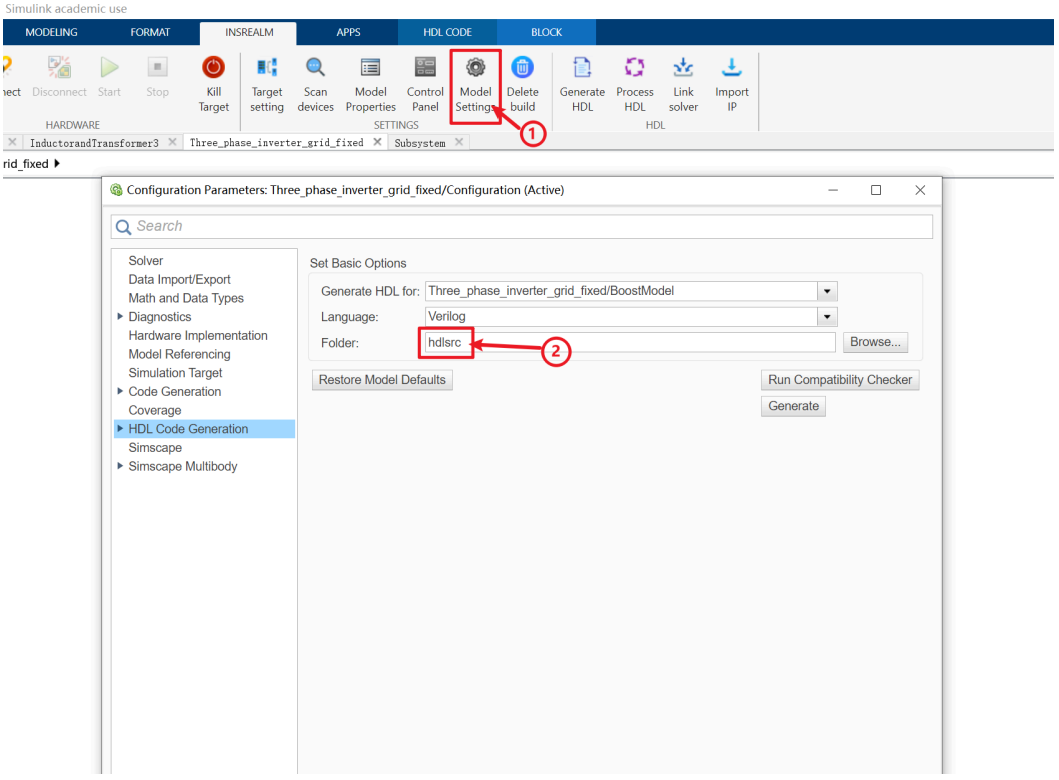
模型搭建好了之后，按照离线仿真的方式，点击Run看一下波形是否正确。如下图所示为上述搭建的三相逆变器的网侧电流波形、逆变器侧电流波形以及电容电压波形。



1.3 软件设置

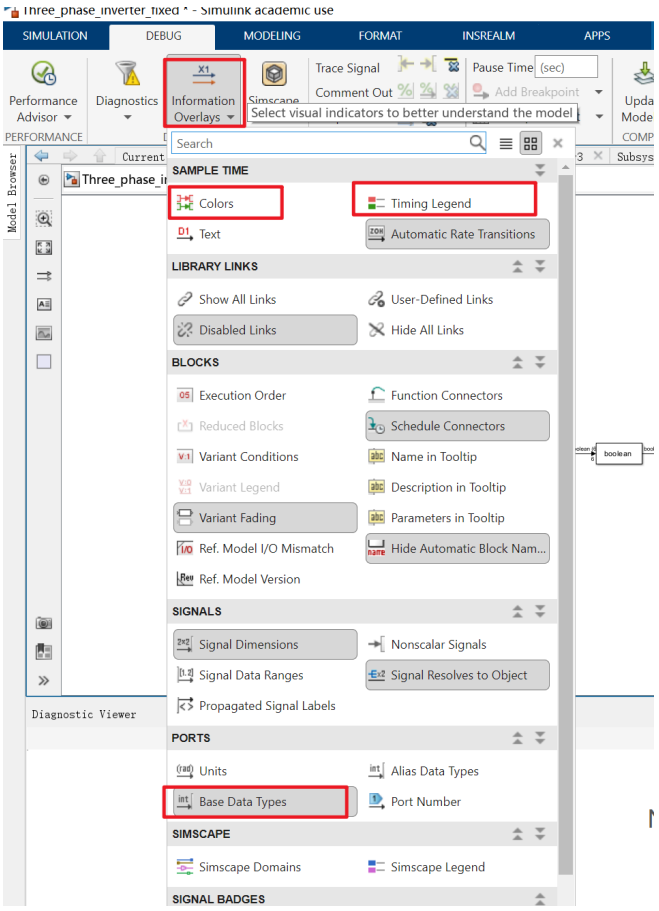
代码生成路径设置

如下图所示，进行生成代码路径的设置，这里Folder路径填入相对路径。默认使用hdlsrc即可。这样simulink生成的代码将会存放在slx模型文件所在的hdlsrc文件夹下。需要注意的是，这里的路径只能用相对路径，否则会报错。

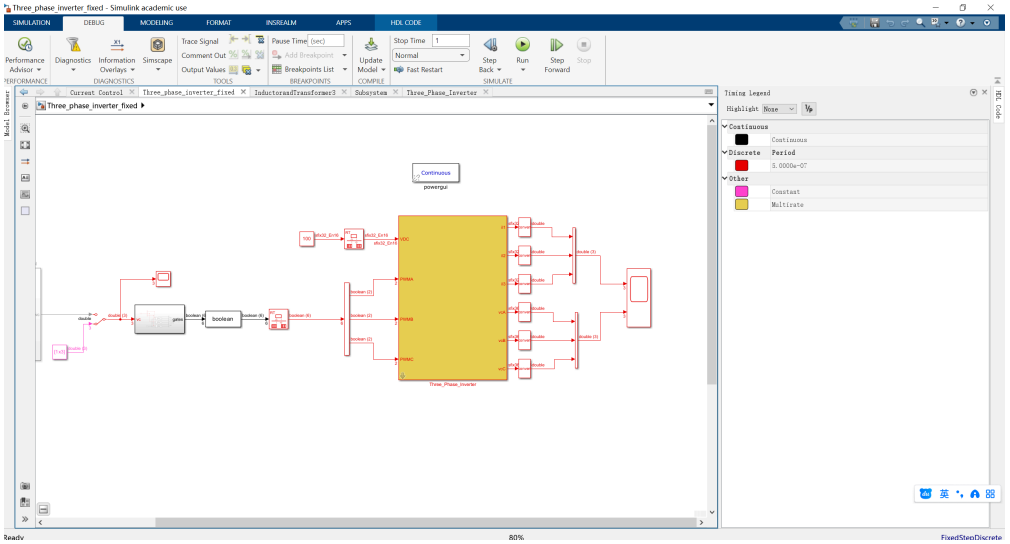


辅助信息显示设置

如下图所示，勾选DEBUG/Information Overlays下面的Colors, Timing Legend, Base Data Types。

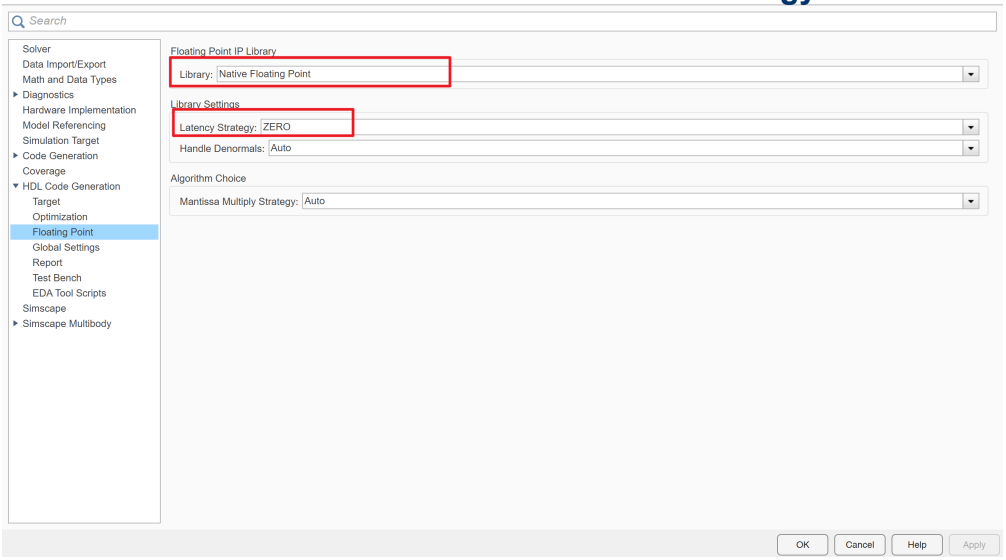


这样可以在模型上看到步长、数据类型等信息。如下图所示，右侧的Timing legend就写明了每种步长对应的颜色。左侧模型中连线上面则写了数据类型。方便用户debug。



浮点设置

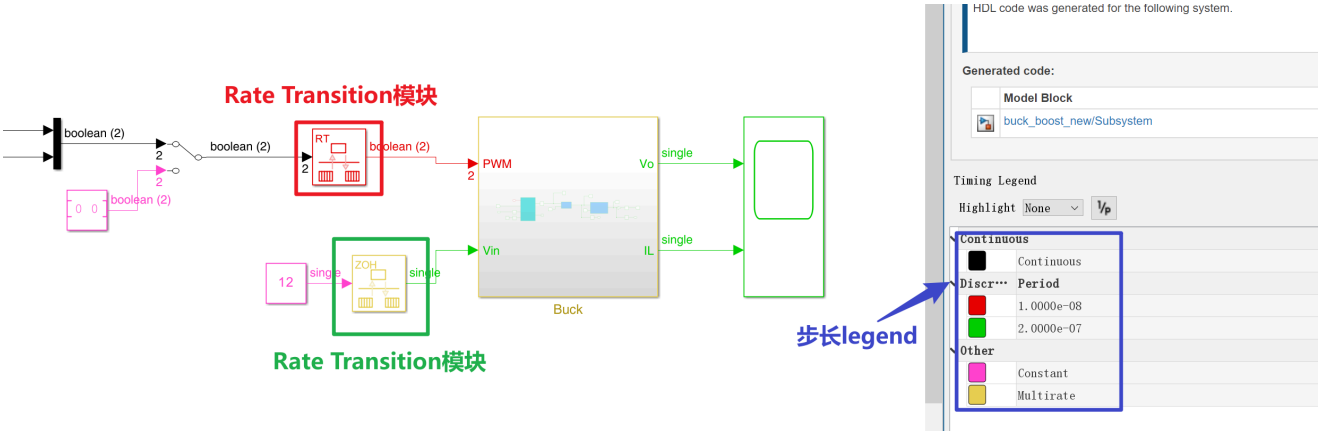
如下图所示为HDL Coder的浮点设置，需要将Library选择为Native Floating Point



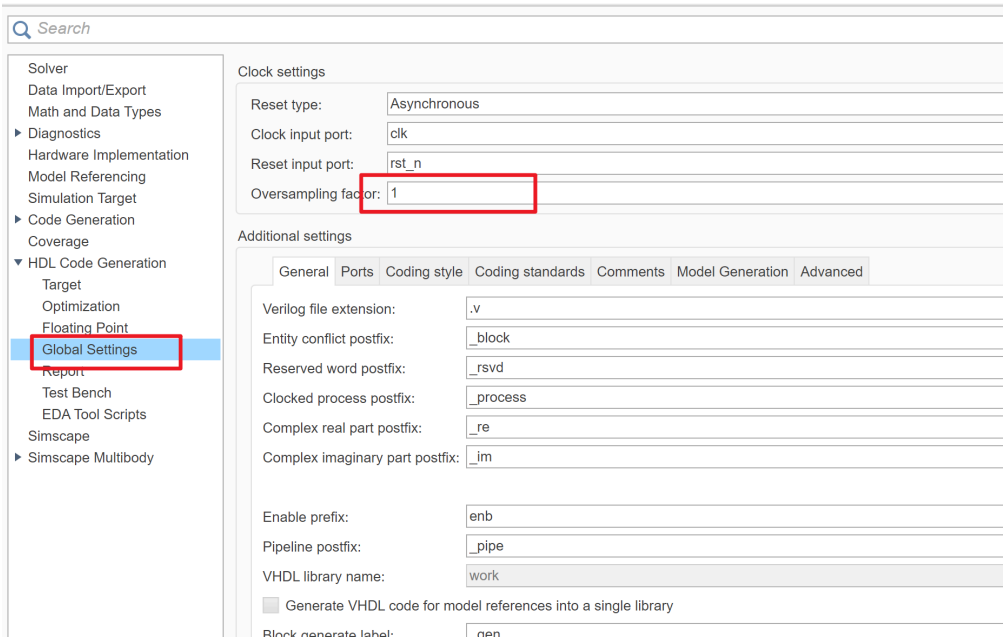
由于浮点的计算一般比较消耗时间，比如一个浮点数的乘法或者加减法，视FPGA的主频或者性能不同，需要若干个时钟才能计算出结果。这里simulink为了处理而Latency Strategy有三个选项可选，ZERO，MIN，MAX。如果选择MIN，则代码生成的时候，会自动在浮点运算上增加流水线（6拍）延迟，以满足时序要求。选择MAX，自动增加8拍流水延迟，选择zero，则不加延迟。这里概念相对比较复杂，刚开始入门可以选择MIN。
如需要把模型步长进一步压缩到极致，则需要选择zero，并使用多周期约束。这部分内容在高阶内容中进行介绍。

步长设置

由于此模型将运行在FPGA上，因此这里设置的步长为FPGA的仿真步长。与普通的模型仿真略有不同，FPGA的模型通常通过rate transition模块，手动指定模型路径的步长。比如如下图所示，此时绿色部分就是转换过后的步长1e-7，红色的模型为1e-8。内部也将通过自动推断来确定步长。



通常如果模型是浮点运算，往往需要设置步长为100ns~1us，具体要视模型中的延迟长度来决定。如果是定点运算，则直接设置为最短步长1e-8即可。此外还需要设置过采样系数，设置位置如下图所示。

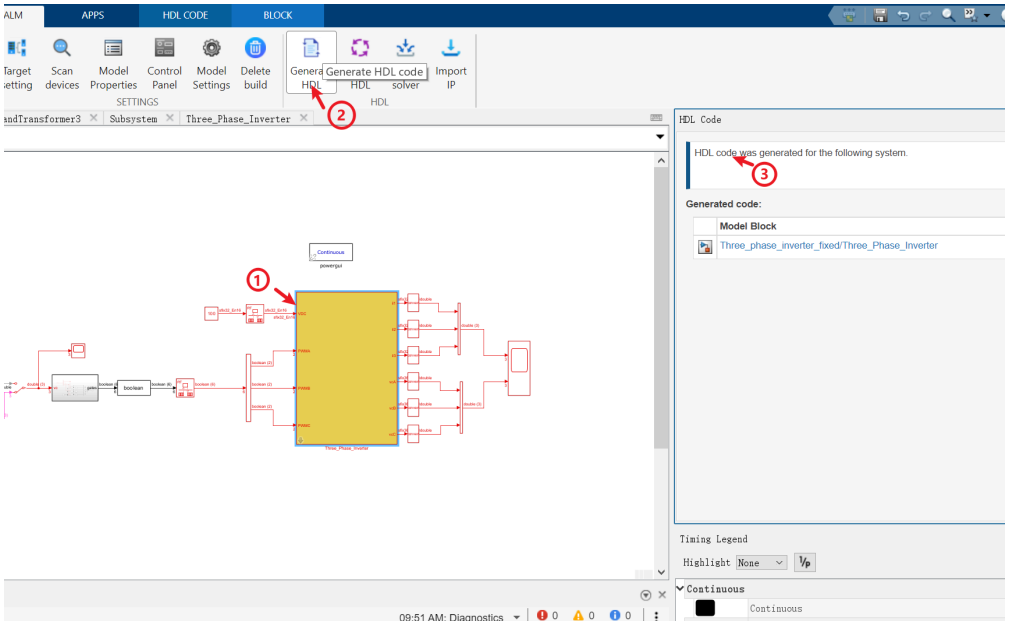


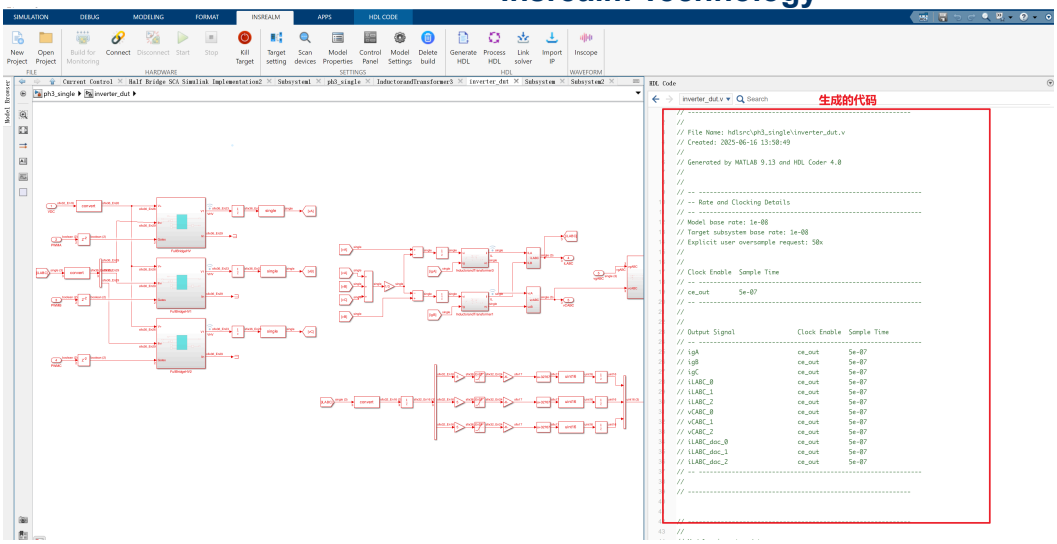
Oversample factor设置：

- 单一步长模型：如果整个模型只使用一种步长，则设置 `Oversample factor = 步长 / FPGA主频周期`
  - 例如：FPGA主频100M，步长为5e-7s时，Oversample factor = 5e-7 / 10e-9 = 50
  - 例如：FPGA主频100M，步长为1e-7s时，Oversample factor = 1e-7/ 10e-9 = 10
- 多步长模型：如果模型中同时存在多种步长，则设置 `Oversample factor = 最小步长 / FPGA主频周期`。例如：FPGA主频100M，模型中同时有1e-7s和1e-8s，则选择最小的1e-8s，Oversample factor = 1e-8 / 10e-9 = 1。

2. RTL代码生成

首先选中需要转换为HDL代码的模块，如下图所示。然后点击Generate HDL，等待HDLCoder的代码生成。当提示HDL code was generated for the following system，表示代码生成成功。此时可以点击下面的模型名字，查看生成的HDL代码



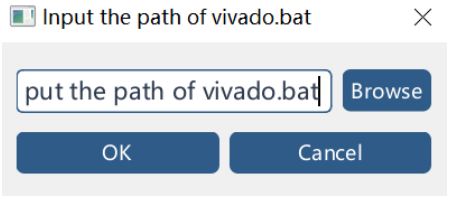


插入寄存器

对于所有的输入，输出端口，需要插入一个unit delay。

3. Bitstream生成

当HDL代码生成完毕之后，即可点击Process HDL。第一次使用会弹窗提示输入vivado.bat的路径



需要安装Vivado2023.2版本，安装好后，填入vivado.bat的路径。比如：`C:\Xilinx\Vivado\2023.2\bin\vivado.bat`

点击OK之后会弹出下图所示的界面。对所有的输入输出端口进行一定的映射。一般情况下，使用默认配置即可。表格中各列说明如下

- `variable_name`：端口名字
- `interface_type`：端口类型，比如cpu, digital, dac等
- `data_type`：端口的数据类型
- `direction`：端口方向，输入或者输出
- `offset_address/channel`：端口地址或者硬件通道序号
- `is_param`：是否配置为模型的参数。当interface_type为axilite，且端口方向为输入时，可以配置此选项。如果勾选，则配置为模型参数，将在后续步骤中block的mask上配置此参数，且在模型运行之前进行配置了；如果不勾选，则配置为模型的输入端口，可以在模型运行过程中动态修改。

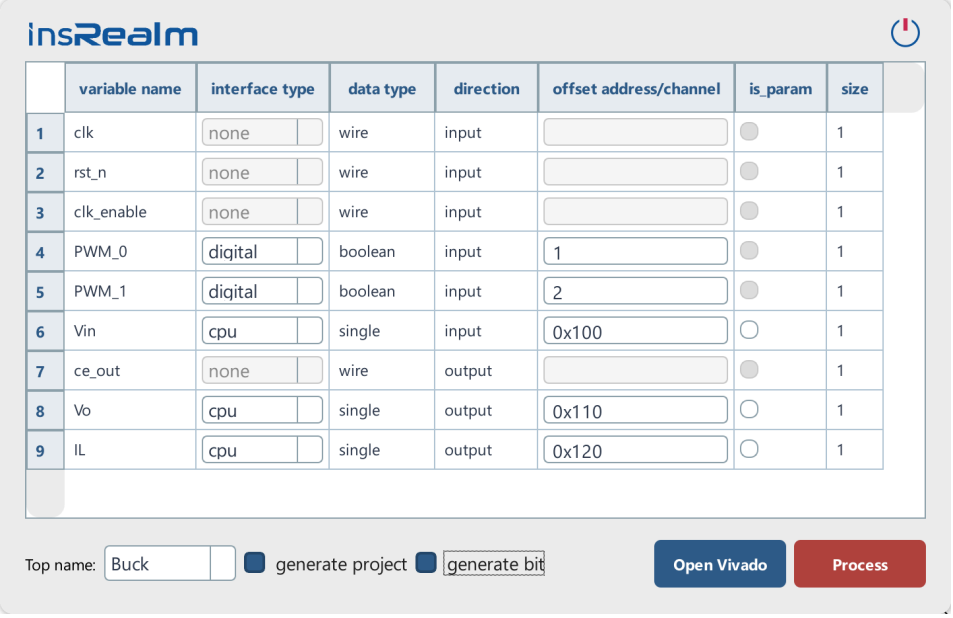
> **注意**

> 如果端口名字以 `param_`开头，那么将默认勾选上is_param，并且前缀相同，序号不同的变量将被合并为同一个向量参数。比如param_A_0、param_A_1、param_A_2将被合并为一个A参数，这样便于后续用户填写参数。

端口类型说明

- `interface type`是端口的类型选择，需要对每一个输入输出端口设置，可选选项为
  - `cpu`，即将此端口配置为内存，通过自动生成的驱动代码即可配置改参数。如果数据类型为single/sfix/ufix，默认配置为此类型。选择此类型时，offset address/channel内填写内存的地址（程序会自动分配），且表格最后一栏
  - `digital`，数字io。选择此类型时，端口将被映射到数字IO引脚上。offset address/channel内填写通道映射到引脚。如果数据类型为boolean，默认配置为此类型。且只有类型为boolean时才能
  - `dac`，模拟输出通道，填写后面的offset address/channel，配置其引脚，目前开放的通道为1~24，其中 **1~8对应转接板上F1~F8（10MHz超高速DAC），9~24对应转接板上DAC1~DAC16（1MHz高速DAC）**。如果数据类型为uint16，默认配置为此类型。
  - `none`，不配置映射。
- `offset address/channel`，即地址或者通道编号。比如接口类型是CPU，此参数代表CPU访问的数据地址，一般不需要修改。如果接口类型是DAC，此参数表示数据映射到的DAC通道序号，如果是digital input，则表示映射到的GPIO的引脚编号。

一般情况下，用户可以直接使用默认的配置，也可以根据自己的需要再修改通道配置。



配置完了之后点击右下角Process，即可开始bit生成。此过程视电脑性能，大概需要十几分钟到一个小时。综合过程中，会弹出一个命令行窗口，将打印综合过程中vivado软件的输出。

> **综合过程注意事项**

- 如果发现命令行窗口长时间没有新的打印，比如超过十分钟，可能是vivado软件遇到了BUG，建议尝试关掉之后重新编译
- 综合结束之后，命令行中可以查看综合之后的WNS（worst negative slack），如果这个值是负数，说明FPGA模型时序不满足，则需要调整模型来满足时序。一般造成这个的原因就是计算的链路太长了，可以考虑在适当的地方增加unit delay（即增加锁存器）来满足时序要求。

4. 模型运行

创建新的RTScale模型文件，添加`FPGA Solver`模块，然后点击`Link solver`，选择刚刚搭建的simulink模型，程序会自动完成链接操作，并创建对应的输入和输出端口（如果路径发生了改变，比如拷贝模型到其他位置，或者其他电脑，需要重新link模型）。
有几点需要注意：

1. 输出和输出的端口数量在block上进行了显示，且每个线设置了对应的变量的名字，请确保后续接线按照此顺序和数量。
2. 为了方便用户使用，所有的端口的类型都使用double，并且在模型内部自动完成了类型与对应端口的转换，无需用户手动转换类型。

最后，用户可以根据仿真需要，对模块的输入和输出端口进行相应的连接，比如下图中搭建了三个PWM模块，并搭建了控制框图。搭建完毕之后，点击Build for Monitoring，然后connect之后点运行即可实时运行仿真模型。