HLCD- homework5 报告

2000012741 钟作奇

Problem 1&2

有

$$Y(i,j)+=A(i,k)\times B(k,j)$$

$$A^A=\begin{bmatrix}1&0&0\\0&0&1\end{bmatrix},A^B=\begin{bmatrix}0&0&1\\0&1&0\end{bmatrix},A^Y=\begin{bmatrix}1&0&0\\0&1&0\end{bmatrix} \tag{1}$$

Problem 1

对于Problem1，有

$$T=\begin{bmatrix}0&1&0\\0&0&1\\1&1&1\end{bmatrix},\ T^{-1}=\begin{bmatrix}-1&-1&1\\1&0&0\\0&1&0\end{bmatrix} \tag{2}$$

可计算得对应的 A^XT^{-1} 与最简非零解(dx,dy,dt)为

$$A^AT^{-1}=\begin{bmatrix}-1&-1&1\\0&1&0\end{bmatrix},(dx,dy,dt)_A=(1,0,1)$$

$$A^BT^{-1}=\begin{bmatrix}0&1&0\\1&0&0\end{bmatrix},(dx,dy,dt)_B=(0,0,1) \tag{3}$$

$$A^YT^{-1}=\begin{bmatrix}-1&-1&1\\1&0&0\end{bmatrix},(dx,dy,dt)_Y=(0,1,1)$$

可知A与Y均为Systolic型，其中A沿方向$(1,0)$传播，B沿方向$(1,0)$传播；B为Stationary型。使用PPT中的代码模块构筑即得`PEArray1`程序。

Problem 2

对于Problem2，有

$$T=\begin{bmatrix}1&0&0\\0&0&1\\0&1&0\end{bmatrix},\ T^{-1}=\begin{bmatrix}1&0&0\\0&0&1\\0&1&0\end{bmatrix} \tag{4}$$

可计算得对应的 A^XT^{-1} 与最简非零解(dx,dy,dt)为

$$A^A T^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, (dx, dy, dt)_A = (0, 0, 1)$$

$$A^B T^{-1} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, (dx, dy, dt)_B = (1, 0, 0) \tag{5}$$

$$A^Y T^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, (dx, dy, dt)_Y = (0, 1, 0)$$

可知B与Y均为Multicast型，其中B在x方向上进行多播（即：对于相同的y不同的x，复用一个数据），Y在y方向上进行多播；A为Stationary型。使用PPT中的代码模块构筑即得`PEArray2`程序。另外，对于C的output，构筑了一个adder tree。且为了与testbench中的延时匹配，输出过程中在adder tree里增加了一个reg。

Problem 3

对于Problem3，使用buffer对输入输出进行缓冲，暂存一些数据。设计PE时，相较于前两题的PE，这里将C的输入输出分为两个通道共四个端口。对于A与B，使用`a_buf`，`b_buf`对数据进行暂存，使用`ab_flow`对A与B的数据是否继续输入进行控制。对于C，进行`Input`，`Compute`，`Update`，`Output`四个过程的处理。其中：

- `Input` 将C输入路`ci`的输入模块定义
- `Compute` 在输入路上进行，在a, b数据均valid、a, b的out端ready、可以开始计算的命令`start`发出且对于这一组数未进行计算`!calculated`时，进行$c' = a * b + c$的计算。计算后，这一组数被赋值为已经计算，且a, b可以继续flow
- `Update` 则是在满足更新条件时对输出通道`co`传递的`c_data`进行更新；同时，如果`ci`路前面PE的c可以送出来而且这个位置和PE的y相同时，`ci`路的C替换之(通过`replace`信号在`co`路实现)，并发出开始计算的信号；
- `Output` 路则是传递`c_data`，在`replace`信号为1后，其变为valid；同时，用`co_rec`的buffer链接前面`co`路的输出。如果这一级的数据被替换了那么输出之，否则则输出上一PE `co`路的数据（类似于水闸：要么是这个库区放水，否则就接收停留上一级的数据）

对于PE与PE之间的链接，与原先的主要差别在把赋值`:=`变成了物理连线`<>`，因为能否赋值、能否继续传输的一系列判断与操作都由`buffer`进行了。与整体的输入输出链接时，将输入`a_in`,`b_in`,`c_in`的位置的buffer置于ready状态，而数据C的输入处PE`co`路的`valid`先置为0.

代码通过了所给testbench。

```
PEArrayTest1:
PEArray1
- should handle 4x4x4 case
- should handle 8x4x4 case
PEArrayTest3:
PEArray3
- should handle continue stream of data, 4x4x4
- should handle continue stream of data, 4x4x8
- should handle discontinue input stream of data, 4x4x4
- should handle discontinue input stream of data, 4x4x8
- should handle discontinue output stream of data, 4x4x4
- should handle discontinue output stream of data, 4x4x8
- should handle discontinue I/O stream of data, 4x4x4
- should handle discontinue I/O stream of data, 4x4x8
PEArrayTest2:
PEArray2
- should handle 4x4x4 case
- should handle 4x8x4 case
```