Block and Input/Output Buffer Placement for Skew/Delay Minimization in Flip-chip Design

Introduction

Flip-chip bonding was developed by IBM in 1960's. It gives the highest chip density of any packaging method to support the pad-limited ASIC design. One of the most important characteristics of flip chip designs is that the input and output buffers could be placed anywhere inside a chip, like core cells. We use the top metal or an extra metal layer, called Re-Distributed Layer (RDL), to connect input or output buffers to bump balls. Figure 1 shows the cross section of RDL. Bump balls are placed on RDL and use RDL to connect to IO buffers.

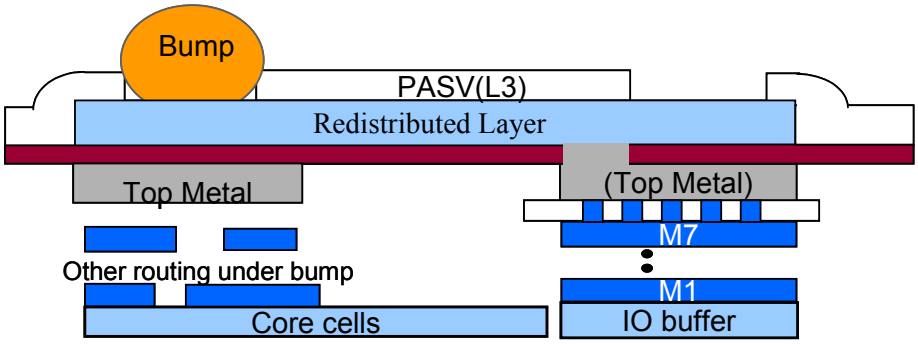

Figure 1. Cross section of RDL

In some memory-related designs, like memory controllers, there are a large number of input/output pins being used as data bus. For such designs, we have to control the timing of the input/output signals. In other words, we have to make sure that the input signals arrive at the core simultaneously. In the same way, it also needs to make sure that the output signals arrive at bump balls simultaneously. This can be achieved through controlling the positions of bump balls, input/output buffers and first-stage/last-stage cells.

In this problem, we assume the design is a block-based design. All the input/output signals are connected to block ports through the input/output buffers. But to simplify the complexity, we assume all the bump balls are placed at pre-defined locations and their signals are determined. So, the program needs to place input buffers, output buffers and blocks to make sure that all input signals from bump balls via input buffers to blocks arrive simultaneously and output signals from blocks via output buffers to bump balls arrive simultaneously. To approximate the goal, this problem asks to minimize the following objective function:

$$WeightA \times \left(\sum_{j=1}^{n1} \sum_{k=j+1}^{n1} \left| d_j^i - d_k^i \right| + \sum_{j=1}^{n2} \sum_{k=j+1}^{n2} \left| d_j^o - d_k^o \right| \right) + WeightB \times \left(\sum_{j=1}^{n1} d_j^i + \sum_{j=1}^{n2} d_j^o \right)$$

n1 and *n2* are the numbers of input and output signals, respectively. d^i_j and d^o_j are the path delays of the *j*th input signal and the *j*th output signal, respectively. The path delay of an input signal is the delay of a path from a bump ball via an input buffer to a block port of a block; the path delay of an output signal is the delay of a path from a block port of a block via an output buffer to a bump ball. *WeightA* and *WeightB* are the weighting factors, and they are specified as input parameters.

The blocks ports of a block are the ports used to connect the block and input/output buffers. Minimizing the above objective function means that it needs to minimize the skew of the path delays of all input signals, output signals and the total path delay. To simplify the calculation of a path delay, the distance between two objects (bump balls, buffers, or block ports) are used to estimate the path delay between these two objects.

Figure 2 is an example of this problem. This is a block-based design, where each block has block ports that will be connected to input/output buffers. Bump balls are placed on the RDL; therefore, bump balls can overlap with input/output buffers and blocks.

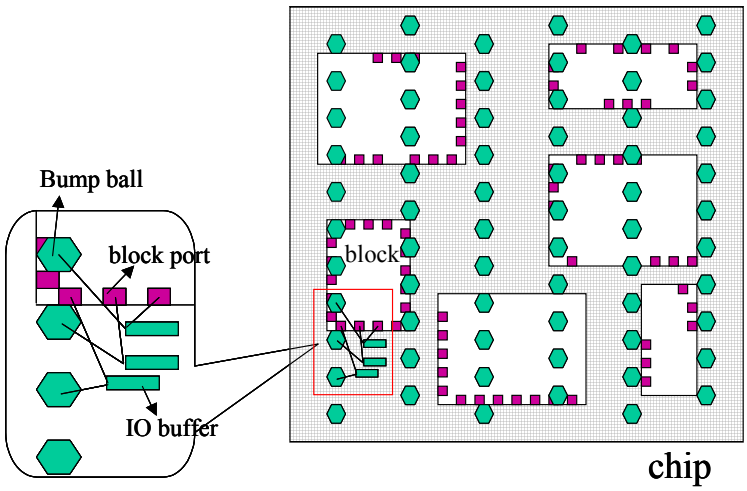

Figure 2. Example of this problem

Problem:

Please determine the locations and rotation degrees (to be defined later) of input buffers, output buffers and blocks to minimize the above objective function. Use Manhattan distance as the path delay of two objects. The Manhattan distance of two objects can be determined by

$$|x1 - x2| + |y1 - y2|$$

where ($x1$, $y1$) and ($x2$, $y2$) are the coordinates of the two objects. For a bump ball, its center is used as its coordinate.

Problem Assumptions:

1. The design is a block-based design, all core cells are partitioned or grouped into blocks and there are block ports to connect to input and output buffers.
2. The coordinate of the chip is set to the bottom left corner of the chip, i.e., (0, 0). See Figure 3.
3. The coordinate of a bump ball is relative to the coordinate of the chip. See Figure 3.
4. The coordinate of a block is set to the bottom left corner of the block. The coordinate of a block port of a block is relative to the bottom left corner of the block. See Figure 3.
5. The coordinate of an input/output buffer is set to the bottom left corner of the buffer. The coordinate of an input/output (IO) port of an input/output buffer is relative to the coordinate of the buffer. See Figure 3.

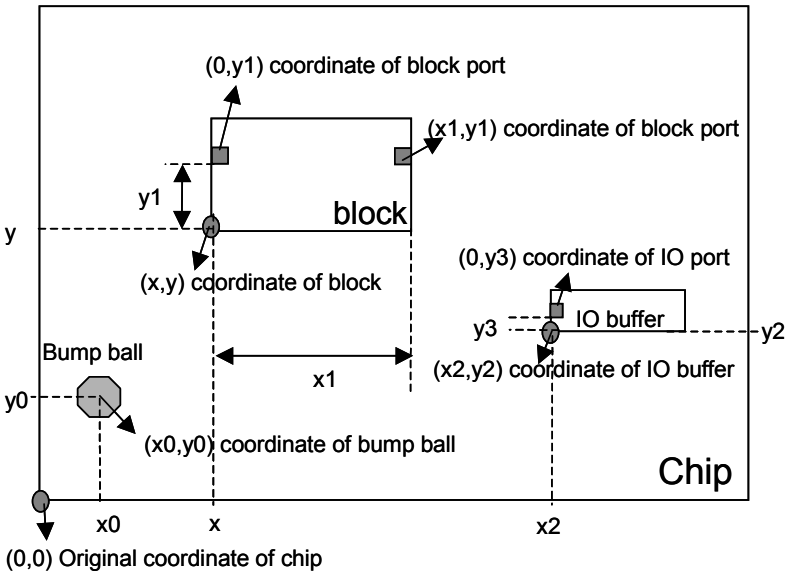

Figure 3. Illustration of coordinate definitions.

6. All input and output buffers are zero delay. Cell delay is neglected.
7. Input buffers, output buffers and blocks can be rotated under four different degrees as shown in Figure 4.

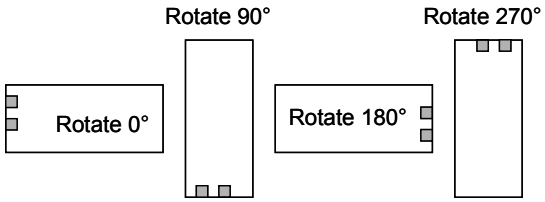

Figure 4. Example of rotation

8. Input buffers, output buffers and blocks can not overlap with each other.
9. Bump balls are placed in RDL, so they can overlap with input buffers, output buffers and blocks.
10. The locations of input buffers, output buffers and blocks must snap to grid. A grid is the minimal x-y resolution to place objects, as Figure 5 shows. All the input/output buffers and blocks have multiple-grid width and height.

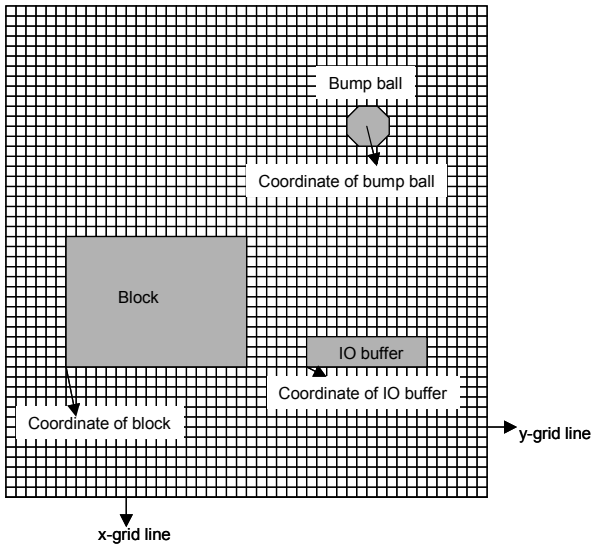

Figure 5. Illustration of "snap to grid".

The coordinate of a block, IO (input or output) buffer, or bump ball must be set at the cross point of x-y grid lines. The x-y grid sizes are input parameters.

11. The shape of a bump ball is regular octagon and its height and width are the same. See Figure 6.

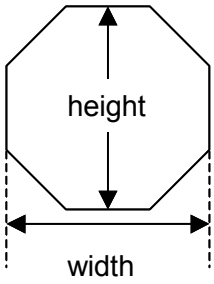

Figure 6. Shape of bump ball

12. The definitions of width (W) and height (H) of a chip, block or input/output buffer are shown in Figure 7.

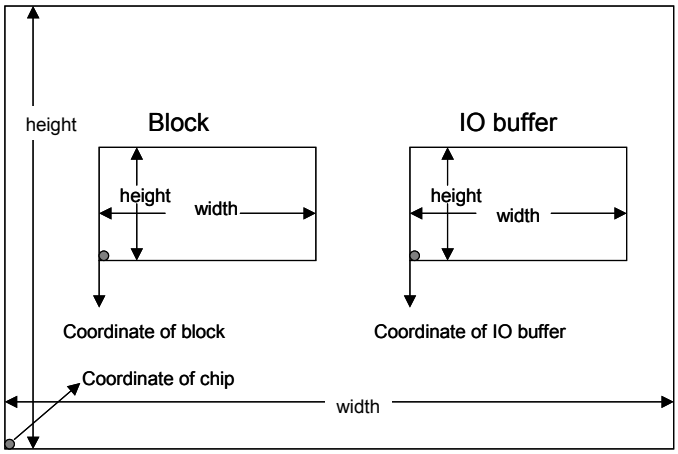

Figure 7. Definitions of width and height of chip, block and IO buffer.

Example of delay calculation.

In Figure 8, the coordinate of bump ball, BA1, is (10, 10). The coordinate of input buffer, I1, is (40, 50) and its input and output port coordinates are (0, 10) and (0, 30), respectively. The coordinate of block, BL1, is (200, 300) and its block port is (0, 30). The path delay from BA1 via I1 to the block port of BL1 is the summation of the delay from BA1 to the input port of I1 and the delay from the output port of I1 to the block port of BL1.

The delay from BA1 to the input port of I1 is equal to:

$$|10-(40+0)|+|10-(50+10)|=80$$

The delay from the output port of I1 to the block port of BL1 is equal to:

$$|(50+30)-(300+30)|+|(40+0)-(200+0)|=410$$

So that path delay from the bump ball to the block is equal to 490.

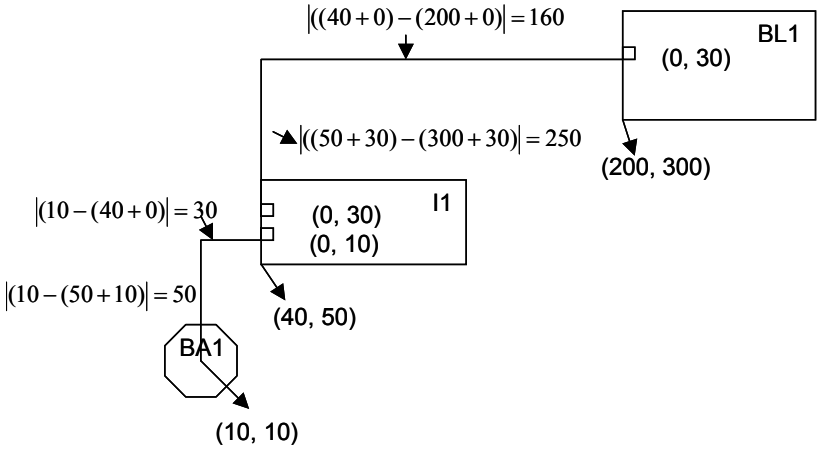

Figure 8. Example of delay calculation

Input parameters:

1. Chip size (WxH).
2. Sizes (WxH) of input/output buffers and blocks under 0-degree rotation.
3. X-Y grid sizes.
4. *WeightA* and *WeightB*.
5. Bump ball size.
6. Netlist among bump balls, input buffers, output buffers, and block ports of blocks.
7. The coordinates of bump balls.
8. The coordinates of block ports of blocks under 0-degree rotation.
9. The coordinate of the input/output port of an input/output buffer under 0-degree rotation.

Input file format:

There are three sections in an input file. The first part specifies the program parameters, including chip size, *WeightA*, *WeightB*, grid size and the height of a bump ball. The second part is the library section that specifies the basic properties of input/output buffers and blocks. The third part, called design section, defines the location and name of a bump ball and the netlist of the design.

Syntax:

#

'#' is the leading character of a comment which is ended by a newline.

CHIP_SIZE: W x H

This line specifies the size of the chip, where ***W*** is the width of the chip and ***H*** is the height of the chip.

GRID_SIZE: x y

This line specifies the grid size, where ***x*** is the distance between two consecutive x-grid lines and ***y*** is the distance between two consecutive y-grid lines.

WEIGHT_A=n

This line specifies the weighting factor *WeightA* whose value is set to ***n***.

WEIGHT_B=m

This line specifies the weighting factor *WeightB* whose value is set to ***m***.

BUMP_HEIGHT=h

This line specifies the height of a bump ball whose value is set to ***h***.

```
[LIBRARY]
    [IO_PROPERTY]
        INPUT CELL_NAME WxH I(x1,y1) O(x2,y2)
        …
        OUTPUT CELL_NAME WxH I(x1, y1) O(x2,y2)
        …
    [END IO_PROPERTY]
    [BLOCKS]
        [BLOCK BLOCK_NAME WxH]
            PORT_NAME (x, y)
            …
        [END BLOCK]
        …
    [END BLOCKS]
[END LIBRARY]
[DESIGN]
    [BUMP_BALL]
        BUMP_NAME (x, y)
        …
    [END BUMP_BALL]
```

[NET_LIST]

CELL_NAME INST_NAME I O

…

[END NET_LIST]

[END DESIGN]

where

[LIBRARY] :

Start of library section.

[IO_PROPERTY] :

Start of input/output buffer properties section.

INPUT CELL_NAME WxH I(x1, y1) O(x2, y2) :

It specifies the properties of an input buffer.

CELL_NAME is the cell name of this buffer. ***W*** and ***H*** are the width and height of this buffer under 0-degree rotation. ***I(x1, y1)*** and ***O(x2, y2)*** specify the coordinates of the input and output ports of this buffer under 0-degree rotation, respectively.

OUTPUT CELL_NAME I(x1, y1) O(x2, y2) :

It specifies the properties of an output buffer.

CELL_NAME is the cell name of this buffer. ***W*** and ***H*** are the width and height of this buffer under 0-degree rotation. ***I(x1, y1)*** and ***O(x2, y2)*** specify the coordinates of the input and output ports of this buffer under 0-degree rotation, respectively.

[END IO_PROPERTY] :

End of input/output buffer properties section.

[BLOCKS] :

Start of blocks properties section.

[BLOCK BLOCK_NAME WxH] :

Start of a block properties section.

BLOCK is the keyword of this line. ***BLOCK_NAME*** is the name of this block, ***W*** and ***H*** are the width and height of this block under 0-degree rotation.

PORT_NAME (x, y) :

It specifies the coordinate of a block port.

PORT_NAME is the name of this block port and ***(x, y)*** is the coordinate of this block port under 0-degree rotation.

[END BLOCK] :

End of a block properties section.

[END BLOCKS] :

End of blocks properties section.

[END LIBRARY] :

End of library section.

[DESIGN] :

Start of design section.

[BUMP_BALL] :

Start of bump ball section

BUMP_NAME (x, y)

It specifies the coordinate of a bump ball.

BUMP_NAME is the name of this ball. ***(x, y)*** is the coordinate of this ball.

[END BUMP_BALL] :

End of bump ball section.

[NET_LIST] :

Start of netlist section.

CELL_NAME INST_NAME I O

It specifies the netlist of an input/output buffer.

CELL_NAME specifies which buffer defined in the library section is used. ***INST_NAME*** is the instance name of this buffer. ***I*** and ***O*** are the names of block port and bump ball.

[END NET_LIST]

End of netlist section.

[END DESIGN] :

End of design section.

An example is used to illustrate the input file format. It is shown in the following:

 # comment is started by '#' chars.
 # Specify the size of the chip
CHIP_SIZE : 3000 x 3000
#specify the X -Y grid size
GRID_SIZE : 5 x 10
 # specify the WeightA and WeightB of the objective function
WEIGHT_A=50
WEIGHT_B=50
#specify the height of a bump ball

```
BUMP_HEIGHT = 190
[LIBRARY]
    [IO_PROPERTY]
        INPUT XAA 200 x 40 I (0, 10) O (0, 30)
        OUTPUT YAA 370 x 65 I (0, 10) O (0, 40)
    [END IO_PROPERTY]
    [BLOCKS]
        [BLOCK BL1 500 x 400]
            BL1_1 (0, 0)
            BL1_2 (10, 0)
            BL1_3 (0, 50)
        [END BLOCK]
        [BLOCK BL2 300 x 400]
            BL2_1 (0, 200)
        [END BLOCK]
    [END BLOCKS]
[END LIBRARY]
[DESIGN]
    [BUMP_BALL]
        BA1 (10, 10)
        BA2 (10, 210)
        BA3 (10, 410)
        BA4 (10, 610)
    [END BUMP_BALL]
    [NET_LIST]
        XAA I1 BA1 BL1_1
        XAA I2 BA2 BL2_1
        YAA I3 BL1_2 BA3
        YAA I4 BL1_3 BA4
    [END NET_LIST]
[END DESIGN]
```

In this example, the size of this chip is 3000 x 3000, X-Y grid size is 5 and 10. There are two types of input/output buffers in the library. One is XAA and the other is YAA. There are two input buffers, two output buffers, two blocks and four bump balls in this example. Block1 has three block ports and Block2 has one block port. The netlist from a bump ball to a block is easy to understand; for

example, the input buffer, “I1”, connects to bump ball, “BA1”, and block port ,“BL1_1”, therefore, we can find that the bump ball “BA1” connects to “BL1_1” of BL1 via “I1”.

Output file format:

The output file must contain:

1. The coordinates and rotation degrees of all input buffers.
2. The coordinates and rotation degrees of all output buffers.
3. The coordinates and rotation degrees of all blocks.
4. Values of *WeightA* and *WeightB*.
5. Value of the objective function.
6. Execution time.

Note: Once the location and rotation degree of a block or an input/output buffer are determined, the coordinate of the block or input/output buffer is the position at which the bottom left corner of the block or input/output buffer is.

Syntax of the output file:

[INPUT_BUFFER]

INPUT_NAME (X,Y) ROTATION

….

[END INPUT_BUFFER]

The coordinates of input buffers are put in this section, where ***INPUT_NAME*** is the name of an input buffer and ***(X,Y)*** is its coordinate. ***ROTATION*** is the rotation degree of this buffer. It would be one of 0, 90, 180, or 270.

[OUTPUT_BUFFER]

OUTPUT_NAME (X,Y) ROTATION

….

[END OUTPUT_BUFFER]

The coordinates of output buffers are put in this section, where ***OUTPUT_NAME*** is the name of an output buffer and ***(X,Y)*** is its coordinate. ***ROTATION*** is the rotation degree of this buffer. It would be one of 0, 90, 180, or 270.

[BLOCK]

BLOCK_NAME (X,Y) ROTATION

….

[END BLOCK]

The coordinates of blocks are put in this section, where ***BLOCK_NAME*** is the name of a block and ***(X,Y)*** is its coordinate. ***ROTATION*** is the rotation degree of this block. It would be one of 0, 90, 180, or 270.

WEIGHT_A=n

This line reports the value of *WeightA* that is specified in the input file.

WEIGHT_B=m

This line reports the value of *WeightB* that is specified in the input file.

RESULT = n

The result, where ***n*** is the value of the objective function.

EXECUTION_TIME=m sec.

The execution time of the program.

Example of the output file format:

[INPUT_BUFFER]
Input1 (470, 200) 0
Input2 (470, 600) 180
II1 (470, 1000) 90
[END INPUT_BUFFER]
[OUTPUT_BUFFER]
Output1 (400, 500) 0
Output2 (400, 900) 90
[END OUTPUT_BUFFER]
[BLOCK]
Block1 (300, 30) 270
Block2 (600, 30) 0
[END BLOCK]
WEIGHT_A=50
WEIGHT_B=50
RESULT= 25000
EXECUTION_TIME=3000 sec.

In this example, it reports the coordinates and rotation degrees of blocks, input buffers and output buffers. The value of the objective function and the execution time of the program also are reported. Please note that this example is used to illustrate the format of the output file format, and is not the real case result.

Language and Platform:

Programming language should be either C or C++ and program must be developed under UNIX or LINUX based platform.

Grading Strategy for the program efficiency and results

1. Total score of a test case is calculated as follows:
   (a). Value of the objective function : 60%
   (b). Execution time : 30%
   (c). Graphical interface for displaying the placement of input/output buffers, blocks and bump balls : 10%
2. If core dump occurs or the run-time is more than 8 hours while executing a test case, the score for this test case will be 0.