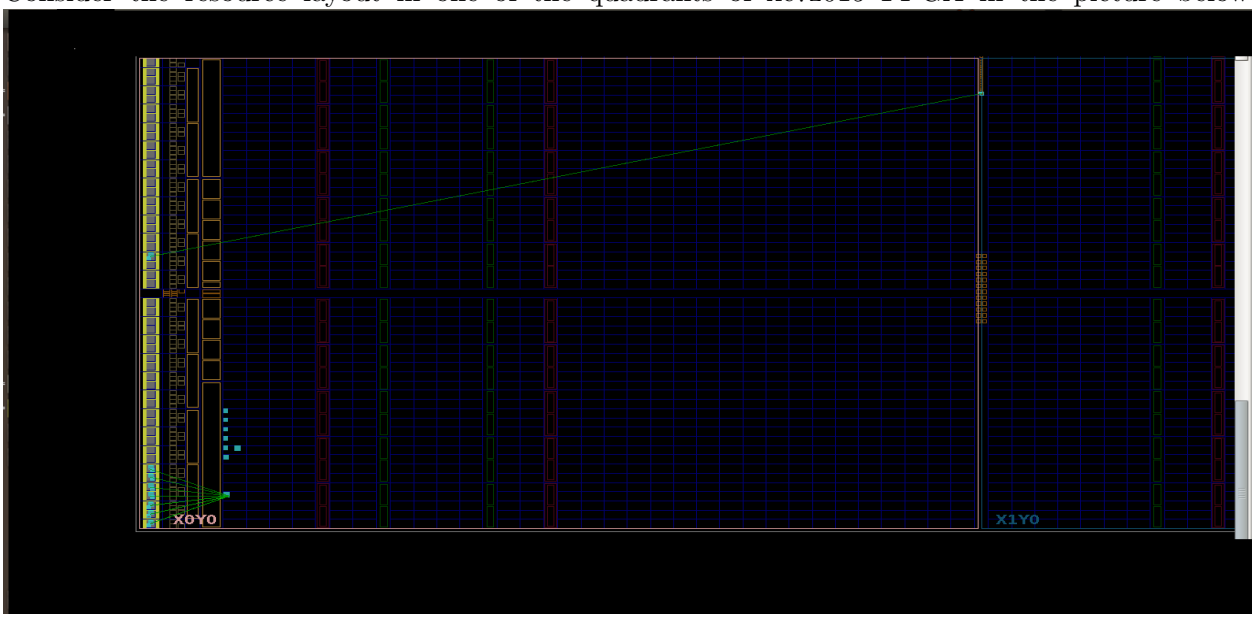
Notes for the FloorPlanning Paper

Let us consider a floorplanning example where we have to make a floorplan for two slots S_1 and S_2 on the FPGA fabric. Each slot has resource requirements denoted as $\{D_1, B_1, C_1\}$ and $\{D_2, B_2, C_2\}$ where D, B and C represent DSP, BRAM and CLB respectively.
Our proposed system takes as an input, the resource requirement of each slot and a description of the resource distribution of the FPGA fabric and it returns the placement coordinates of the slots. A slot is represented using 4 parameters i.e. the two bottom left coordinates and the width and the height of the slot. In our considered example the slots S_1 and S_2 are represented as (x_1, y_1, w_1, h_1) and (x_2, y_2, w_2, h_2).

Consider the resource layout in one of the quadrants of xc7z015 FPGA in the picture below



The resource distribution on the x and y coordinates can be described as

#BRAM	h.0	0<x<6
	h.1	7<x<11
	h.2	7<x<W
#DSP	h.0	0<x<4
	h.1	5<x<14
	h.2	7<x<W

#CLB	h.x	0<x<4
	h.(x-1)	5<x<7
	h.(x-2)	8<x<12
	h.(x-3)	13<x<15
	h.(x-4)	17<x<29

where

$$h \in \{0, H-1\} \quad and \quad x \in \{0, W-1\}$$

(**to be included**: the position of the forbidden regions which are going to be used to formulate constraints)

The PR constraints used to generate the slot positions are

- there must be enough resources within the slots
- Only one reconfigurable partition within a frame
- there should be no common tiles between slots (no interference)
- forbidden regions must not be included in the slots
- Left and right edges must be placed in proper positions
- the amount of wasted resources should be minimized (Wasting DSPs is more expensive than BRAMs which in turn is more expensive than CLBs)
- Other optimizations such as lower wire length between slots or lower length to I/O etc... can be added as constraints

constraint 1: This is fairly straight forward and needs no further elaboration
constraint 2: A frame is the smallest reconfigurable physical region and it is aligned with the boundaries of the clock regions. A Reconfigurable Frame cannot contain logic from more than one Reconfigurable Partition.
Constraint 3: This is also fairly straight forward and can be stated as two slots S_1 {x_1, y_1, w_1, h_1} and S_2{ x_2, y_2, w_2, h_2 } do not interfere if

$x_1 + w_1 < x_2$ or $y_1 + h_1 < y_2 : (x_1 < x_2$ and $y_1 < y_2)$

constraint 4: Global resources, clock resources (BUFG, BUFR, MMCM etc...), static components (BSCAN, XADC etc...) must not be included in the rectangles. The FPGA resource description must also include these components to avoid including them in the rectangles

constraint 5: The left and right edges of the rectangles must be placed between CLB-CLB, CLB-BRAM or CLB-DSP and not between two interconnect columns (INT-INT).

constraint 6: This constraint has two parts. The first part states that the slots must be the smallest possible rectangles which must also contain the required resources. The second part of the constraint states that the number of wasted resources should be kept to the minimum. Resources are considered wasted if they are included in a slot while not being used. All FPGA resources are not equally valuable. This means it is better to waste CLBs than BRAMs than DSPs. Hence there should also be some kind of constraint to state the maximum amount of resources that can be wasted in search of an optimal slot. This constraint must also force the optimizer to start searching for slots from the columns of the most expensive resources.

constraint 7: Other constraints such as minimal wire length between slots (to reduce routing delay and power consumption) can be used as constraints to choose the optimal slots

Previous approaches of solving the floorplanning problem by other authors mostly involve two steps. First, all the possible slots are enumerated. This is done by starting a scan on the fpga fabric from the bottom left corner and lisitng all the possible rectangles which will contain all the necessary resources for the respective slots. Then some sort of heuristics/optimization is applied to choose the best ones from the set of possible slots.
Our approach instead focuses on using a mathemathical techniques to generate the slots first hand (without enumerating all possible slots) by using the proper constraints.

The system will finally return the parameters of S_1 $\{x_1, y_1, w_1, h_1\}$ and $S_2\{ x_2, y_2, w_2, h_2 \}$