Paper Title*

Biruk B. Seyoum, Alessandro Biondi and Giorgio Butazzo
Scuola Superiore Sant'Anna,
Pisa, Italy
email: {b.seyoum, alessandro.biondi, giorgio.butazzo}@santannapisa.it

Abstract*—*to be completed before sept 09

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Floorplanning is one of the major challenges in the field of dynamic parital reconfiguration. The placement of the static and reconfigurable regions (RR) on the FPGA fabric must satisfy the application requirements set by the application designer while also respecting the technological constraints set by the manufacturer. The conventional approach for an automated generation of FPGA floorplans usually involves two steps. First for each RR, all the possible rectangular RRs that satisfy the resources requirement of the RR are enumerated. This is done by starting a scan on the fpga fabric from the bottom left corner and lisitng all the rectangles that contain all the necessary resources for the respective RRs. Then some sort of heuristics/optimization is applied to choose the optimal ones from the set of possible RRs. This approach has many problems { *to be listed later*}

Our approach instead focuses on applying the optimization process on a lower level of abstraction of the fpga fabric i.e., rather than applying optimization to select the most optimal one from a set of pre-scanned slots, we modeled the different types of resources on the fpga and their distribution along with forbidden regions as a set of constraints and added these constraints to the predefined constraints related to dpr.

Let us consider a floorplanning example where we have to make a floorplan for two RR: RR$_i$ and RR$_k$ on the FPGA fabric. Each RR has resource requirements denoted as {D$_i$, B$_i$, C$_i$} and {D$_k$, B$_k$, C$_k$} where D, B and C represent DSP, BRAM and CLB respectively.
Our proposed system takes as an input in the resource requirement of each RR and a description of the resource distribution of the FPGA fabric and it returns the placement coordinates of the RRs on the fpga fabric.
A RR is represented using 4 parameters i.e. the two bottom left coordinates and the width and the height of the region. In our considered example RR$_i$ and RR$_k$ are represented as (x$_i$, y$_i$, w$_i$, h$_i$) and (x$_k$, y$_k$, w$_k$, h$_k$). A forbidden region is also represented as a RR hence a forbidden region F$_i$ can be represented as fx$_i$, fy$_k$, fw$_i$, fy$_i$

Identify applicable funding agency here. If none, delete this.

II. FPGA Floorplanning Problem Description

CLBs, BRAMs and DSPs are the fundamental building blocks of FPGAs. These resources which have a heterogenous distribution are organized as grids of columns. A single column contains a single type of resource. The entire grid of columns are further organized into quadrants named clock regions. Heterogeneous distribution of resources implies that the layout of resources differ from clock region to clock region. Resources in the same clock region share the same clock. FPGAs also contain hardware elements such as hard processors, I/O blocks etc... which should not be included inside reconfigurable regions. In this work, these regions are referred to as forbidden regions. A single column inside a clock region is named a tile. The smallest addressable segments of the device a Xilinx configuration memory is called a frame. A single frame inside the configuration memory is mapped to a single tile on the FPGA fabric hence a tile is the minimum reconfigurable unit.

Floorplanning is a step in the fpga design flow where the placement of a region (or sub circuit) on the fpga fabric is set by designer instead of the placer in the place and route tool. In FPGA design flows that are not related to dynamic partial reconfiguration, floorplanning is optional and it is usually done to assist the place and route tool but in dynamic partial reconfiguration it is a mandatory step in the design process. A typical floorplanning problem related to DPR consists of finding placemenets for **N** reconfigurable regions (RR) which have unique resource requirements. A reconfigurable region must satisfy all of the following DPR requirements

- there must be enough resources within each RR
- A frame can not be shared between two RRs (no interference between RRs)
- static regions and forbidden regions on the FPGA must not be included in the RR
- Left and right edges of the RR must be placed in proper positions

Besides respecting the DPR requirements a good floorplan must minimize the wasted resources in a RR and also the distance between RRs (intra RR wirelength) and the distance between a RR and a static resource it connects to such as an I/O block.

III. Proposed Approach

This work consists of modelling the FPGA with a x-y coordinate system. A single unit on the x axis equals one

column wide and one unit on the y axis (a row) equals to the height of one bram. BRAMs were chosen as the unit for a single row on the y axis since for virtex and 7 series FPGAs, which were utilized in this work, the number of BRAMs per tile is less than the number of CLBs per tile or DSPs per tile. For example, in one of the FPGAs that was used in this work, xc7z045fbv676, a single tile consisted of 50 CLBs, 10 Brams and 20 DSPs.
The origin of the coordinate system is located at the bottom left corner. There are **W** columns on the x axis and **N** rows on the y axis. A rectangular reconfigurable region, $\mathbf{R}_i$, can be described with four parameters i.e., the two bottom left coordinates $\mathbf{x}_i$, $\mathbf{y}_i$ and the width $\mathbf{w}_i$ and height $\mathbf{h}_i$. An FPGA is composed of **C** clock regions. If there are C_h number of clock regions in the horizontal direction and C_v number of clock regions in the vertical direction then the total number of clock regions **C** is defined as

$$C = C_h \cdot C_v \tag{1}$$

Between two C_h lies the central clock column. This column is considered a forbidden region and should not be included inside a RR. In our model, we merged all the C_h clock regions in to a single clock region and hence our model consists of only C_v number of clock regions. To avoid breaking the constraint of not including the central clock column in a RR, the position inside the horizontally merged clock regions, where the central clock column used to be, is modeled as a forbidden region. { *Insert a picture of original and merged clock regions to clarify* }

A merged clock region consists of **r** rows hence

$$H = C_v \cdot r \tag{2}$$

A. transformation to binary

To use linear optimization the x or y axis can also be represented with a set of binary variables which denote each row or column respectively. Choosing which axis to change in to binary is an important design decision. In all the FPGA families that were chosen to be studied for this project, the Y axis was observed to contain less number of resources than the x axis. For example in kintex xc7z045fbv676 contains 100 columns on the x axis and 70 rows on the y axis. Perhaps representing the y axis with binar of variables was a better option as it results in a less number of binary variables. The reduction on the number of binary variables to represent each row on the y axis was made possible by assigning a row per bram instead of rows per clb or per dsp on each column as a bram spans at least 5 or more clb high and 2 or more dsp high. This is actually a neccessary abstraction as brams are indivisble and a finegrained assignment of rows (that is based on clbs or dsps) ends up generating a floorplan that violates rules in the implemetnation tool.

B. FPGA resource finger-printing

The FPGA is now abstracted using binary variables on the y axis and integers on the x axis and C_v clock regions. The distribution of each resource on the x axis of the FPGA in a single row can be represented using a piece-wise linear function. For example in zynq xc7z015 the number of clbs on the x axis between the bottom left corner i.e., (0, 0) and a point x, on the x axis is represented using F(x) as

$$F(x) = \begin{cases} x, & \mathbf{0 \le x < 4}, \\ (x-1), & \mathbf{4 \le x < 7}, \\ (x-2), & \mathbf{7 \le x < 10}, \\ (x-3), & \mathbf{10 \le x < 15}, \\ (x-4), & \mathbf{15 \le x < 18}, \\ (x-5), & \mathbf{18 \le x < 22}, \\ (x-6), & \mathbf{22 \le x < 25}, \\ (x-7), & \mathbf{25 \le x < W}, \end{cases} \tag{3}$$

The number of clb, in a height of a single row, between x_i and x_k where $x_i \ge x_k$ can then be represented as clb(x_i, x_k) such that

$$clb(x_i, x_k) = F(x_k) - F(x_i) \tag{4}$$

if β_{ijk} represents row k in clock region j for a RR_i on the fpga then C(x_i, y_i, w_i, h_i) which is the total number of clbs in a RR S_i can be calculated as

$$C(x_i, y_i, w_i, h_i) = \sum_{j=0}^{C_v} \sum_{k=1}^{r} \beta_{ijk} \cdot (F(x_i + w_i) - F(x_i)) \tag{5}$$

where h_i which is the height of S_i and can be expressed as

$$h_i = \sum_{j=0}^{C_v} \sum_{k=1}^{r} \beta_{ijk} \tag{6}$$

The same resource finger-printing using piecewise linear functions can be done to the bram and dsp on the fpga and this can then be used to determine the amount of a specific type of resource within a RR.

IV. Design

A. definition of optimization variables

To encode the MILP formulation the following binary and real variables are defined.

Variables related to the size, location and number of resources in RR_i.
For each RR_i

- x_i, y_i w_i, h_i $\in \mathbb{Z}$ represent the bottom left coordinates, the width and the height of S_i respectively
- clb_i, $bram_i$ and dsp_i $\in \mathbb{Z}$ represent the number of clb, bram and dsp between x_i and $x_i + w_i$ in a single row respectively.
- clb_req$_i$, bram_req$_i$ and dsp_req$_i$ $\in \mathbb{Z}$ represent the required number of clb, bram and dsp in S_i.
- $\beta_{ijk} \in$ 0,1 represents row k in clock region j for RR_i.

Variables denoting the relationship between two RRs

For two RRs RR$_i$ and RR$_k$

- γ_{ik} ∈ [0,1] is a binary variable used to identify whether S$_i$ is found on the left or on the right of S$_k$
  γ_{ik} = 1 if x$_i$ ≤ x$_k$ [i.e. S$_i$ is on the left of S$_k$]
- θ_{ik} ∈ [0,1] is a binary variable used to identify whether S$_i$ is found on the top or bottom of of S$_k$
  θ_{ik} = 1 if y$_i$ ≤ y$_k$ [i.e. S$_i$ is found below S$_k$]
- Γ_{ik} ∈ [0,1] is used to denote if bottom right x coordinate of S$_i$ is found to the right of the bottom left coordinate of S$_k$
  Γ = 1 if x$_i$ + w$_i$ ≥ x$_k$
- η_{ik} ∈ [0,1] is used to denote if bottom right x coordinate of S$_k$ is found to the right of the bottom left coordinate of S$_i$
  η = 1 if x$_k$ + w$_k$ ≥ x$_i$
- Ω_{ik} ∈ [0,1] is used to denote if the top y coordinate of S$_i$ is found above the lower y coordinate of S$_k$
  Ω = 1 if y$_i$ + h$_i$ ≥ y$_k$
- Ψ_{ik} ∈ [0,1] is used to denote if top y coordinate of S$_k$ is found above the lower y coordinate of S$_i$
  Ψ = 1 if y$_k$ + h$_k$ ≥ y$_i$
- Δ_{ik} ∈ [0,1] is a binary variable which indicates interfernce between RR$_i$ and RR$_k$.
  Δ_{ik} = 0 if there is no interference between the RRs [i.e. not a single tile is shared between regions]

B. Constraint definition

*1) **Semantics constraints**:* The following constraints ensure the soundness of some of the variables.

Constraint 1: ∀ S$_1$ ∈ N, ∀ i = 1...,N$^{\text{max}}$, ∀ x$_i$ = 0..., W, ∀ y$_i$ = 0..., H ∀ w$_i$, ∀ h$_i$

$$\begin{aligned} x_i + w_i \leq W \\ y_i + h_i \leq H \end{aligned} \tag{7}$$

meaning: the right most x coordinate and the top y coordinates of S$_i$ must not exceed the boundaries of the fabric

Constraint 2: ∀ i = 1...,N$^{\text{max}}$, ∀ j = 1...,clk_reg$^{\text{max}}$, ∀ k = 1...,r ∀ h$_i$

$$h_i = \sum_{j=1}^{C_v} \sum_{k=1}^{r} \beta_{ijk} \tag{8}$$

meaning: The height of S$_i$ must be the sum of binary rows in each clock region which are set to 1

Constraint 3: ∀ i = 1...,N$^{\text{max}}$, ∀ j = 1...,clk_reg$^{\text{max}}$, ∀ k = 1...,r ∀ h$_i$

$$y_i \leq H - \beta_{ijk} \cdot (H - (k + (r-1) \cdot j)) \tag{9}$$

meaning: y$_i$ must be constrained not to be greater than the lowest chosen row

Constraint 4: ∀ i = 1...,N$^{\text{max}}$, ∀ j = 1...,clk_reg$^{\text{max}}$, ∀ k = 1...,r

$$\beta_{ij(k+1)} \geq \beta_{ijk} + \beta_{ij(k+2)} - 1 \tag{10}$$

meaning: rows in the same clock region in S$_i$ must be contigious i.e., if β_{ij0} = 1 & β_{ij2} = 1 thenβ_{ij1} must also be equal to 1.

*2) **Resource constraints**:* These set of constraints ensure that each a slot for a RR satisfies the resource requirements of the application. The total number of clbs in a RR$_i$ is expressed as

$$CLB(x_i, y_i, w_i, h_i) = \sum_{j=1}^{C_v} \sum_{k=1}^{r} \beta_{ijk} \cdot clb_i \tag{11}$$

In the same way the amount of bram and dsp can also be expressed as

$$BRAM(x_i, y_i, w_i, h_i) = \sum_{j=1}^{C_v} \sum_{k=1}^{r} \beta_{ijk} \cdot bram_i \tag{12}$$

$$DSP(x_i, y_i, w_i, h_i) = \sum_{j=1}^{C_v} \sum_{k=1}^{r} \beta_{ijk} \cdot dsp_i \tag{13}$$

The resource constraint can simply be stated as the required number of clbs, brams and dsps must be greater than or equal to CLB(x$_i$,y$_i$,w$_i$,h$_i$), BRAM(x$_i$,y$_i$,w$_i$,h$_i$) and DSP(x$_i$,y$_i$,w$_i$,h$_i$) respectively. But the above functions are non linear and can not be used directly to formulate linear constraints.

linearization: In order to employ 11, 12 and 13 as linear constraint, they must first be linearized. To linearize these functions we define three auxilary real variables $\tau1_{ijk}$, $\tau2_{ijk}$ and $\tau3_{ijk}$.

$\tau1_{ijk} \in \mathbb{R}$ | $\tau1_{ijk}$ = β_{ijk} · clb$_i$

$\tau2_{ijk} \in \mathbb{R}$ | $\tau2_{ijk}$ = β_{ijk} · bram$_i$

$\tau3_{ijk} \in \mathbb{R}$ | $\tau3_{ijk}$ = β_{ijk} · dsp$_i$

Hence CLB(x$_i$,y$_i$,w$_i$,h$_i$), BRAM(x$_i$,y$_i$,w$_i$,h$_i$) and DSP(x$_i$,y$_i$,w$_i$,h$_i$) can be restated as

$$CLB(x_i, y_i, w_i, h_i) = \sum_{j=0}^{C_v} \sum_{k=0}^{r} \tau 1_{ijk} \tag{14}$$

$$BRAM(x_i, y_i, w_i, h_i) = \sum_{j=0}^{C_v} \sum_{k=0}^{r} \tau 2_{ijk} \tag{15}$$

$$DSP(x_i, y_i, w_i, h_i) = \sum_{j=0}^{C_v} \sum_{k=0}^{r} \tau 3_{ijk} \tag{16}$$

Now the non linear expression is replaced by a linear one and to complete the linearlization a few constraints must be set. The following are constraints related to $\tau 1_{ijk}$ but similar constraints can be set for $\tau 2_{ijk}$ and $\tau 3_{ijk}$.

Constraint 5: $\forall$ i = 1...,N$^{\text{max}}$, $\forall$ j = 1...,clk_reg$^{\text{max}}$, $\forall$ k = 1...,r

$$\begin{aligned} &\tau 1_{ijk} \geq 0 \\ &\tau 1_{ijk} \leq BIG\_M \cdot \beta_{ijk} \\ &\tau 1_{ijk} \leq clb_i \\ &\tau 1_{ijk} \geq clb_i - (1 - \beta_{ijk}) \\ clb\_req_i &\geq \sum_{j=1}^{C_v} \sum_{k=1}^{r} \tau 1_{ijk} \end{aligned} \tag{17}$$

*3) **Non-interference constraints:***

clock region aligned boundary: A frame (tile) is the smallest reconfigurable physical region and it spans one clock region high and one resource type wide. A reconfigurable frame can not contain logic from more than one reconfigurable partition hence the boundaries of a RR$_i$ must be forced to fit in to clock region boundaries. Such a constraint is enforced by forcing all the rows in a clock region to also be included in the RR if atleast one is included i.e., if one row in a clock region j is part of a RR then the remaining rows within the same clock region j must also be forced to be part of the same RR to satisfy this constraint.
To help us set this constraint we define an intermediate variable l$_j$ $\in \mathbb{Z}$ | $\forall$ j = 1...,clk_reg$^{\text{max}}$, $\forall$ k = 1...,r

$$l_j = \sum_{k=1}^{r} \beta_{ijk} \tag{18}$$

then the following constraint will force the RR boundaries to be aligned with the clock region boundaries by forcing β_{ijk} to become part of the RR if at least one row with in the same clock region becomes part of RR$_i$

Constraint 6: $\forall$ i = 1...,N$^{\text{max}}$, $\forall$ j = 1...,clk_reg$^{\text{max}}$, $\forall$ k = 1...,r

$$\beta_{ijk} \geq \sum_{k=0}^{r} (l_j - beta_{ijk})/(r-1) \tag{19}$$

Interference between two RRs: Two R$_i$ and R$_k$ are said to be non interfering under the following conditions

```
**if** $x_i \leq x_k$ and $y_i \leq y_k$ **then**
   $x_i + w_i < x_k$ or $y_i + h_i < y_k$
**else if** $x_i \geq x_k$ and $y_i \geq y_k$ **then**
   $x_i + w_k < x_i$ or $y_k + h_k < y_i$
**else if** $x_1 < x_k$ and $y_i > y_k$ **then**
   $x_i + w_i < x_k$ or $y_k + h_k < y_i$
**else**
   $x_k + w_k < x_k$ or $y_i + h_i < y_k$
**end if**
```

This above condition can be encoded into a set of MILP constraints as follows

Constraint 7: S$_i$ $\in$ N and S$_k$ $\in$ N

$$\begin{aligned} &\delta_{ik} \geq \gamma_{ik} + \theta_{ik} + \Gamma_{ik} + \Omega_{ik} - 3 \\ &\delta_{ik} \geq (1 - \gamma_{ik}) + \theta_{ik} + \eta_{ik} + \Omega_{ik} - 3 \\ &\delta_{ik} \geq \gamma_{ik} + (1 - \theta_{ik}) + \Gamma_{ik} + \Psi_{ik} - 3 \\ &\delta_{ik} \geq (1 - \gamma_{ik}) + (1 - \theta_{ik}) + \eta_{ik} + \Psi_{ik} - 3 \\ &\delta_{ik} = 0 \end{aligned} \tag{20}$$

4) Interference with Forbidden regions: As stated before forbidden regions are also modeled as a normal RR hence the constraint for non intereference between RR$_i$ and a forbidden region F$_k$ can be set in the same way as done in the previous constraint formulation between two RRs

V. Experiemtnal Results

A. Experimental Setup

How was the experiment implmented i.e, with what kind of prog. langaguge, on what optimization tool, on what platforms... What is the system being tested for ? What is the compositon of the synthethic task suite used for testing. What type of FPGAs are modeled ? What are the challenges when switching between models.

The system is tested for Exec time Vs Num of Reconfigurable regions and Exec time Vs %of resources used by the system on both zynq and virtex

The system is also tested for % of wasted resources (coeficients used to set allowed percentage of resources to be wasted %)VS Exec time on both virtex and zynq. The average wasted resources are also reported when not imposing these constraints and what effect it has on exec time and feasibility in general. my guess is that upto a certain point (upto a certain % of utilization of resources) not imposing these constraints improves the exec time but after the utilization increases (i.e., more applications require more resources) not imposing these coeffecients leads to infeasible constrants. This has to be tested with a carefully designed test suit which will test the limist of the platform.

Finally a case study on a real application on Zynq.