mLSI Design with MINT

Radhakrishna Sanka[1], Joshua Lippai[1] and Douglas Densmore[1]

[1]Department of Electrical & Computer Engineering, Boston University, Boston, MA

{sanka,jlippai,dougd}@bu.edu

1. INTRODUCTION

Despite these many applications for microfluidics in augmenting the synthetic biology workflow, adoption of microfluidics as a potential experimental and test platform has been slow in the synthetic biology community at large. The majority of synthetic biologists lack both the expertise knowledge in fluid dynamics and microfluidic design and the expensive capital equipment for microfluidic fabrication. Layout of microfluidic designs by hand is both time consuming and error prone. In addition, experiments involving mLSI also require a complex control platform including both software and hardware for valve control and fluid manipulation. Such setups are frequently customized for the application at hand and may be difficult to claim for future reuse. Improvements in automation for microfluidic device design and readily available open source software and hardware for control platforms could increase the speed of adoption of microfluidic technologies by synthetic biologists.

In [3] the authors undertook an herculean effort to design and fabricate the MLSI device with only the aid of tools such as AutoCAD and Solidworks most researchers who use mircofluidics today spend a large portion of their time in manually drawing out every physical feature that constitutes the physical device. In this work we leverage MINT [5] to easily define large designs and Fluigi to automatically generate the physical layout of these designs and report the updates made to the MINT specification standard and the architectural updates made to Fluigi.

2. MINT

Previous versions of MINT [5], required the user to specify each of the components and lay them out the connections between each of the component. At that version, generating grids of reaction chambers required the was a tedious task which required the user to write scripts that could generate MINT that would be processed for generating a physical design. This process of defining individual components and connections was arduous and hindered the user's ability to specify geometric constraints on the generated layout.

Various geometric constraints such as "GRID", "BANK" and "TREE" would require the relative component positions and orientations to be fixed. The implication of statements such as "GRID" are that it also requires the place and route tool to process and satisfy the geometric constraints given by the designer. In order to satisfy these constraints, an additional stage in the process that would generate macro cells and algorithms that will optimize the placement within these macro cells will also be implemented.

In addition to being able to add geometric constraints. The updates to the Fluigi and MINT architecture now abstract the layers for manufacturing and the component connectivity, hence the layers can be defined as either "FLOW", "CONTROL", "INTEGRATION" without any worry about how many feature depths are present in the device. The components in the "FLOW" layer include everything that make the device functional. The components in the "CONTROL" layer are typically control the components in the "FLOW" layer. Finally the components in the "INTEGRATION" layer will include all the components that will require external integrations and imply hard constraints onto placement algorithms to ensure that external integrations are not blocked by other features. This redesigned architecture can accommodate multiple fabrication protocols for the devices.

3. FLUIGI

Fluigi is the Place and Route tool that is used for automatically generating the physical layout of the microfluidic chip. Figure 1 show a high level description of the various stages where a MINT description of the device is converted into a physical device. The gray box consists of a parser that was generated using ANTLR [4], the purple box consists of the bulk of algorithms that will generate the physical design, the red box consists of a routines that check if the generated design is valid or not and finally the orange box consist of plug-ins that will generate the design outputs.

The new extensions to MINT will require changes to how the device is modeled in Fluigi. The updated process is described in Algorithm 2. In order to accommodate the new constraints that will be introduced in this work, the Microfluidic Device model as described in [2] will be completely restructured.

4. CONCLUSION

Physical design automation remains to be an actively researched problem [1] in the microfluidics space but the control of these MLSI devices remains yet to be integrated with the physical design automation flow. Using the current virtual microfluidic device model Fluigi[2] currently generates control sequences for a single architecture of microfluidic devices. The ability to group components and the facility to export the device to to a standard format will further allow the device descriptions to be imported into future tools that generate control sequences for arbitrary biology protocols in addition to designing large scale microfluidic designs.

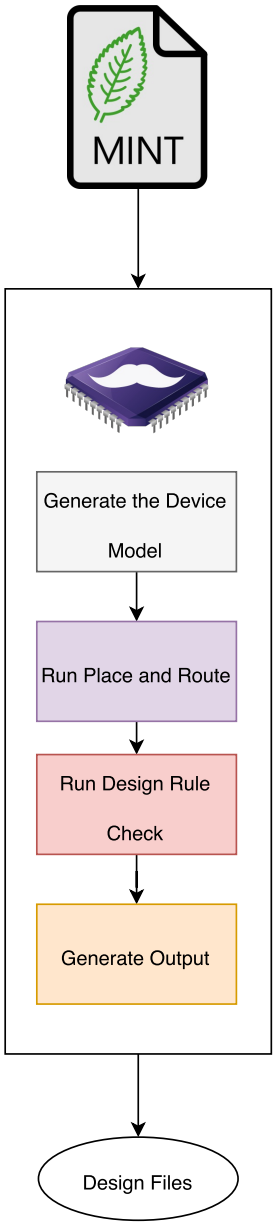

Figure 1: Fluigi Flow - This diagram describes the various stages of the Fluigi Place and Route tool that generates the design files that will be sent for manufacturing.

Require: $N :=$ MINT Description
Ensure: N is a valid description
$D := generate\_device(N)$
Require:
$LB := (\{FLOW, CONTROL, INTEGRATIOM\}\ in\ D$
for all LB **do**
$G := extract\_groups(FLOW)$
$C := generate\_placement\_cells(D, G)$
$FP := generate\_flow\_placements(D, C)$
for $Pin\ FP$ **do**
$place\_flow(P)$
$CP := place\_control(CONTROL, P)$

$IP := place\_integration(INTEGRATION, CP, FP)$
end for
$route(P, CP, FP)$
$D := import\_place(D, P, CP, IP)$
$RESULT = design\_rule\_check(D)$
if $RESULT$ **then**
$generate\_output()$
else
$redo()$
end if
end for

Figure 2: Place and Route: The above algorithm describes the place and route process and highlights the dependencies between the layers within each individual LAYER BLOCK that consists of a FLOW, CONTROL and INTEGRATION. These dependencies determine the order in which they are placed and routed. The bulk of the placement is determined during the $place\_flow()$.

5. REFERENCES

[1] I. E. Araci and P. Brisk. Recent developments in microfluidic large scale integration. *Current opinion in biotechnology*, 25:60–68, 2014.

[2] H. Huang. *Fluigi: An end-to-end Software Workflow for Microfluidic Design*. PhD thesis, Boston University, 2015.

[3] J. Melin and S. R. Quake. Microfluidic Large-Scale Integration: The Evolution of Design Rules for Biological Automation. *Annual Review of Biophysics and Biomolecular Structure*, 36(1):213–231, 2007.

[4] T. Parr and K. Fisher. LL(*): The Foundation of the ANTLR Parser Generator. In *Proceedings of the 32Nd ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '11, pages 425–436, New York, NY, USA, 2011. ACM.

[5] R. Sanka, H. Huang, R. Silva, and D. Densmore. Mint - microfluidic netlist. poster presented at IWBDA 2016, Aug. 2016.