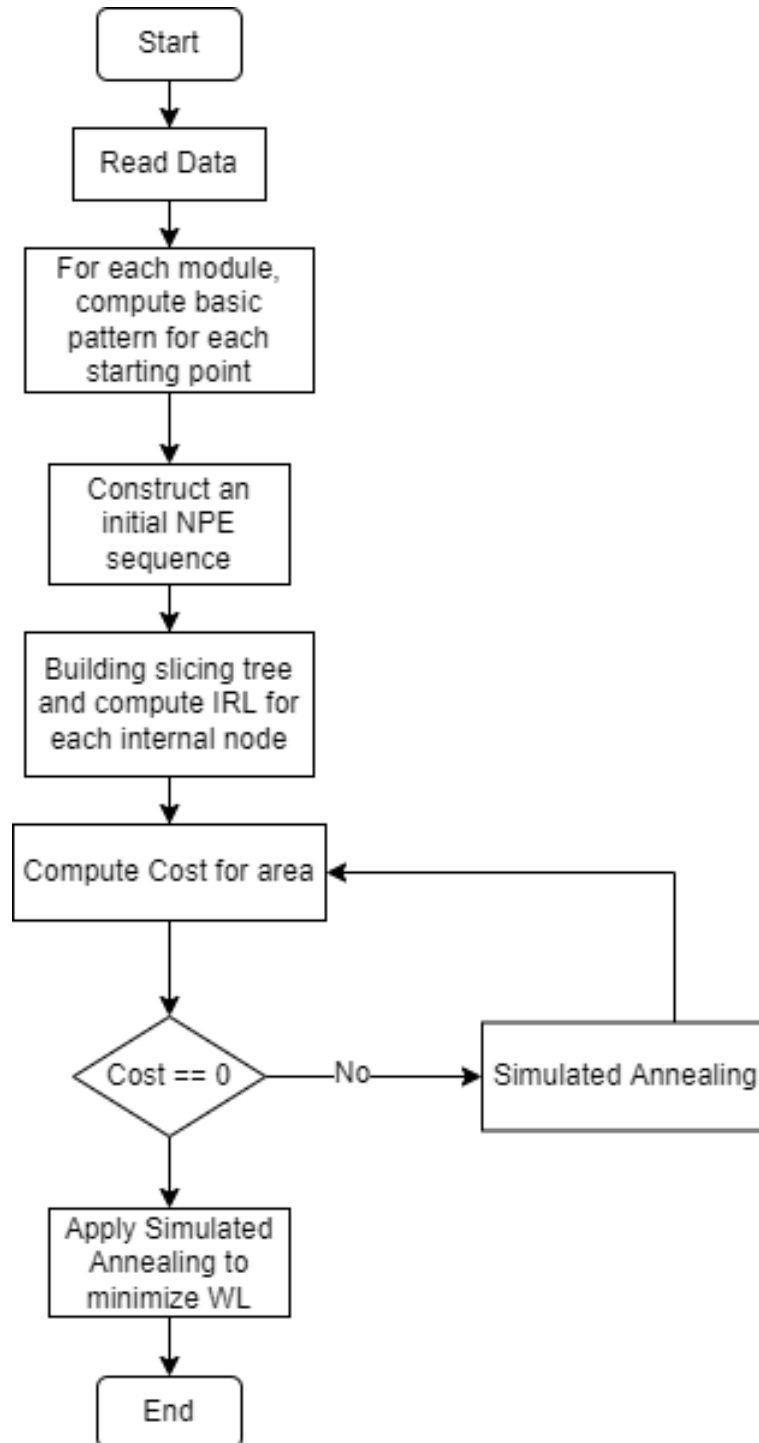


# CS51600 FPGA Final Project

李至弘, 109062610

## 1. Flow Chart



- (1) Compute IRL for each node: 參考 “Floorplan Design for Multi-Million Gate FPGAs “ 中計算每一個 internal node IRL 的演算法

<pre> Evaluate_Node(u)   if u is leaf return   Evaluate_Node(u.left)   Evaluate_Node(u.right)   for every point (x,y) on the pattern do     if u is vertically sliced       Get_Realization_list_V(u,x,y)     else       Get_Realization_list_H(u,x,y) </pre>	<pre> Get_Realization_list_V(u,x,y) Begin:   L(u,x,y) ← ∅ /*initially empty*/   l_v ← L(v,x,y)   len ←  l_v  /*length of l_v*/   for i:=len to 1     x_q = (x + w(l_v[i])) mod w_p     Let l_q be L(q,x_q,y)     if i= 1       upperheight ← α * H + 1     else       upperheight ← h(l_v[i-1])     find j, satisfying h(l_q[j]) ≤ h(l_v[i])       and h(l_q[j-1]) &gt; h(l_v[i])<sup>3</sup>     while (j ≥ 1 and h(l_q[j]) &lt; upperheight) do       h_new ← max(h(l_q[j]), h(l_v[i]))       w_new ← w(l_q[j]) + w(l_v[i])       if (L(u,x,y) is empty and w_new &lt; α * W)         or w_new &lt; width of the first element in           L(u,x,y)         r_new = (x,y,w_new,h_new)         insert r_new as the first element to L(u,x,y)       j ← j-1   End </pre>
---	---

- (2) 首先透過 SA 找到可以擺進去整個 layout 的結果，在透過 SA 在可接受的時間內最小化 Wirelength.
- (3) 解決 S>D 情況，所有 module 在存入 module list 時，不需要 multiplier 的 module 都放入到 list 的最前面，在建構最一開始的 normal polish expression, NPE 時，從 layout 最左下角開始往上擺，一直擺到超過最大的 row number 即跳到下一個 column，來讓一開始不需要 multiplier 的 module 放置在僅有 CLB 的位置上，降低整體執行時間，提供較好的 initial solution。
- (4) 限制每個產生的可行解長寬比不能超過 15，降低執行時間。

## 2. How to compile:

In /Project/src enter the following command:

```
$make
```

How to execute:

In /Project/bin enter the following command:

```
./project <arch file> <module file> <net file> <output file>
```

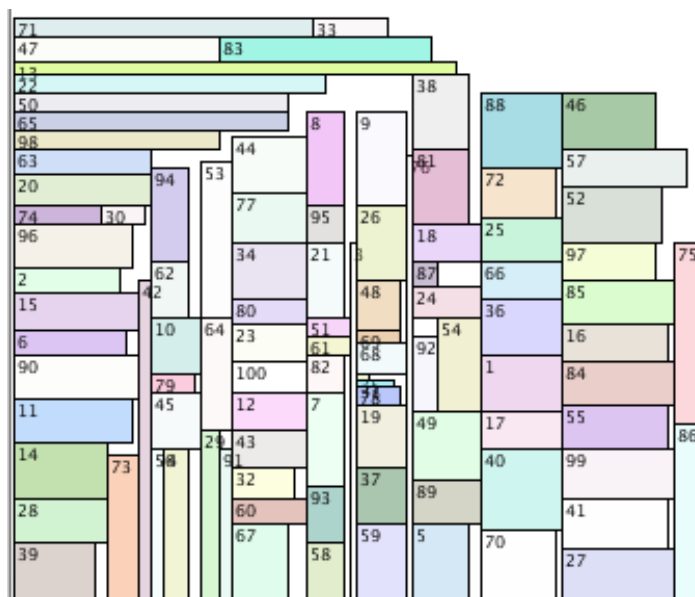
E.g.

```
$ ./project ../benchmarks/case1.arch ../benchmarks/case1.module ../benchmarks/case1.net ../outputs/case1.floorplan
```

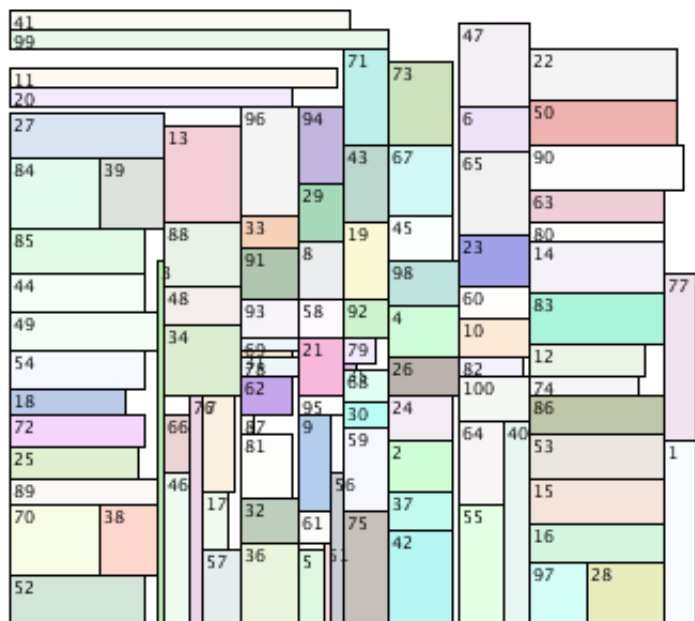
### 3. Results

	HPWL	Run Time
Case1	65510	7 m 44.075 s
Case2	65729.5	8 m 10.422 s
Case3	409102	9 m 55.109 s
Case4	302532.5	9 m 55.038 s
Case5	546647	9 m 50.508 s
Case6	512821	9 m 55.05 s

Case1.



Case2.



The treemap visualization displays a hierarchical structure of data. The root node is labeled '144' and branches into numerous sub-nodes, each represented by a colored rectangle. The nodes are labeled with numerical values, and the colors represent different categories or clusters. The treemap is organized into several main groups, with the largest group on the left and smaller groups on the right. The colors used include shades of green, yellow, orange, red, purple, blue, and grey.

This treemap visualization represents the 2010 U.S. Census data, showing the population distribution across various states. The treemap is color-coded by state and labeled with population values. The states included are:

- Alabama
- Alaska
- Arizona
- Arkansas
- California
- Colorado
- Connecticut
- Delaware
- District of Columbia
- Florida
- Georgia
- Hawaii
- Idaho
- Illinois
- Indiana
- Iowa
- Kansas
- Kentucky
- Louisiana
- Maine
- Maryland
- Massachusetts
- Michigan
- Minnesota
- Mississippi
- Missouri
- Montana
- Nebraska
- Nevada
- New Hampshire
- New Jersey
- New Mexico
- New York
- North Carolina
- North Dakota
- Ohio
- Oklahoma
- Oregon
- Pennsylvania
- Rhode Island
- South Carolina
- South Dakota
- Tennessee
- Texas
- Utah
- Vermont
- Virginia
- Washington
- West Virginia
- Wisconsin
- Wyoming