Final Year Project

High-level Synthesis
Design Space Exploration

Supervisor: Dr. Benjamin Carrion Schafer

Student: LU Shanqi 13116151D

Content

- Part1: Introduction
  - High-level Synthesis
- Part2: Methodology
  - Tool CyberWorkBench
  - Algorithm
- Part3: Graphical User Interface
  - Functions
  - Plotting widget
- Part4: Results
  - Comparison between brute force and simulated annealing

High-level Synthesis

- A process of converting behavior descriptions to hardware implementation
- From SystemC to hardware
- Tools: CyberWorkBench, Xilinx Vivado….

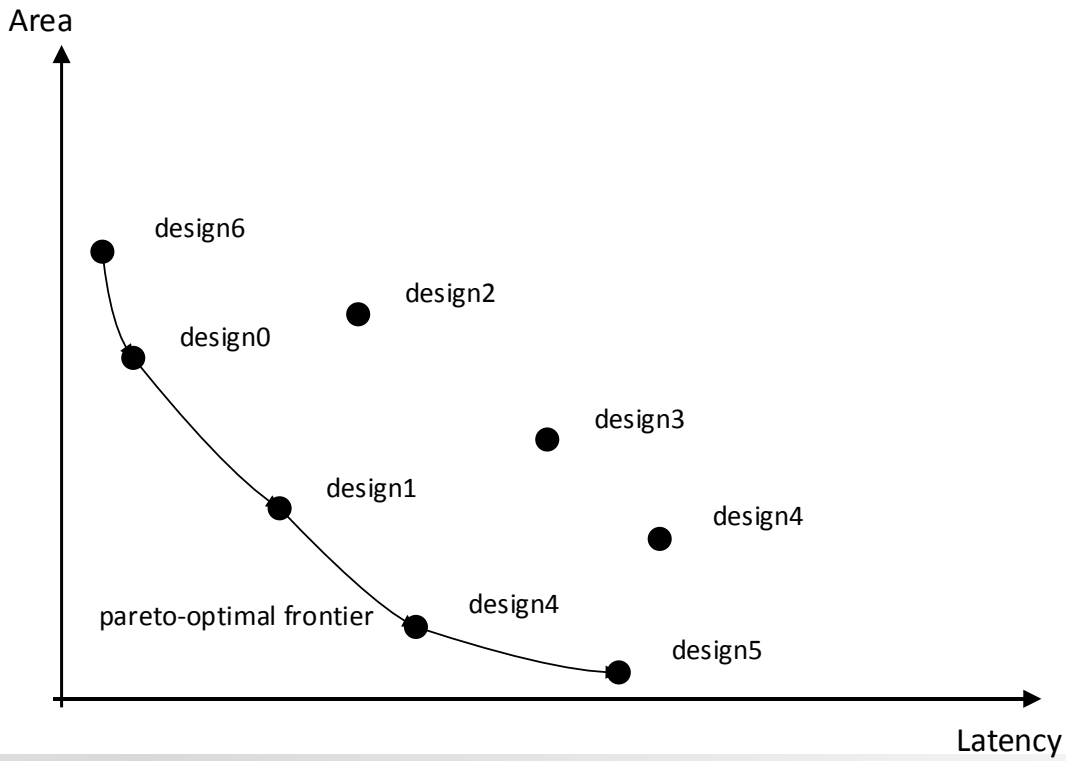
Higher Abstract Level

- Advantages
  - shorter marketing cycle
  - Increasing reusability of programming codes
- Disadvantages
  - Numerous synthesis options are available, hence hard to find optimal designs

Design Space Exploration

- An exploration process aiming to find optimal designs among uncountable candidates in high level synthesis
- Multi-objective optimization problem

- One design pareto dominate another.
- One design is not inferior to another design in all objectives, additionally, there is at least one better objective.



Objectives of this project

- To automate the process of design space exploration
- To develop a heuristic to accelerate design space exploration
- To develop a graphical user interface(GUI) to plot results dynamically

Tool: CyberWorkBench

- Parse description language(e.g. SystemC)
  - Special pragmas can be recognized by parser
- Generate constraint files(e.g. Functional Units constraint file)
- Synthesize
  - Results are store in a *.CSV file

Tool: CyberWorkBench

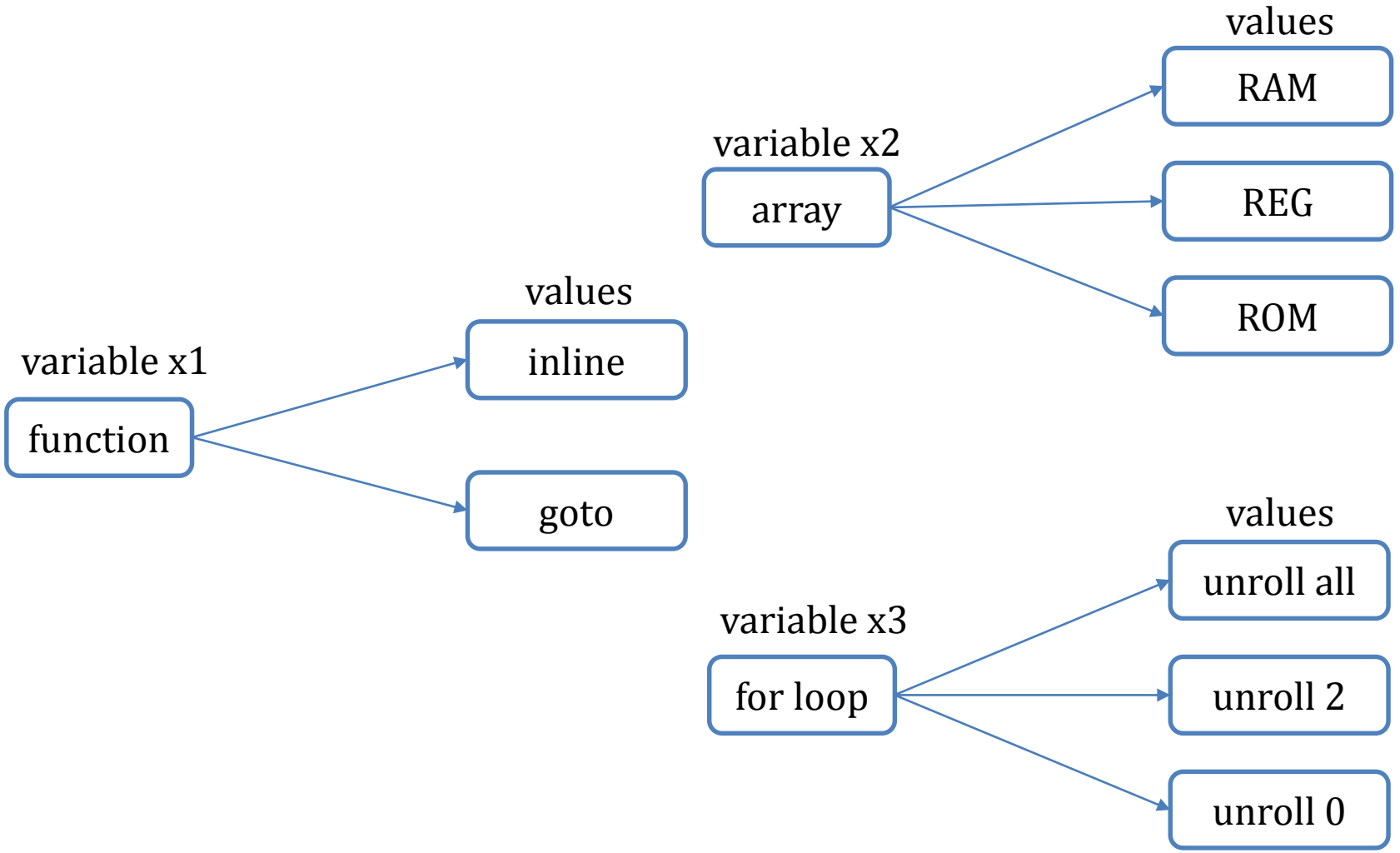
- Pragma insertion
- Key word: Cyber

```
/*Cyber unroll_times = all*/
for(int i=0;i<3;i++)
{
   arrray[i] = i;
}
```

```
array[0] = 0;
array[1] = 1;
array[2] = 2;
```

- Change high-level synthesis options(also called attributes) by inserting such kind of pragmas

Tool: CyberWorkBench



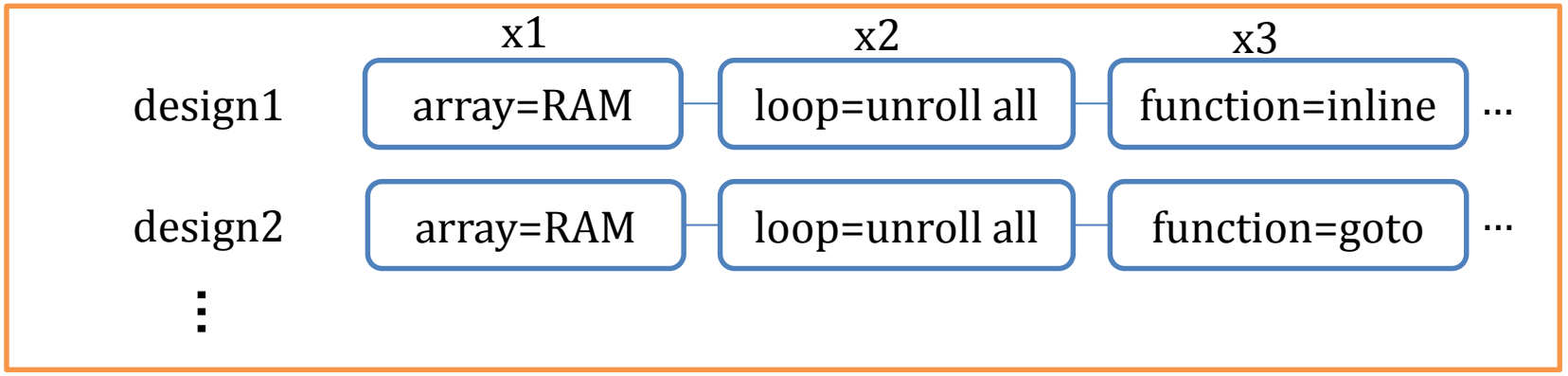
Problem clarification

- Treat high-level synthesis process as a black box function

$$(Area, latency) = f(x1, x2, x3\ldots..)$$

- Multiple input variables: x1,x2,x3…..
- Two objectives: area, latency
- Find more pareto optimal designs in shorter time

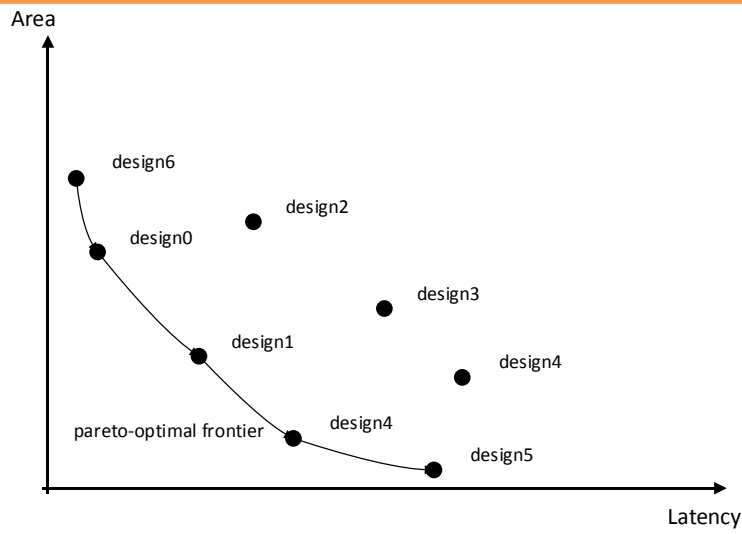
Problem clarification



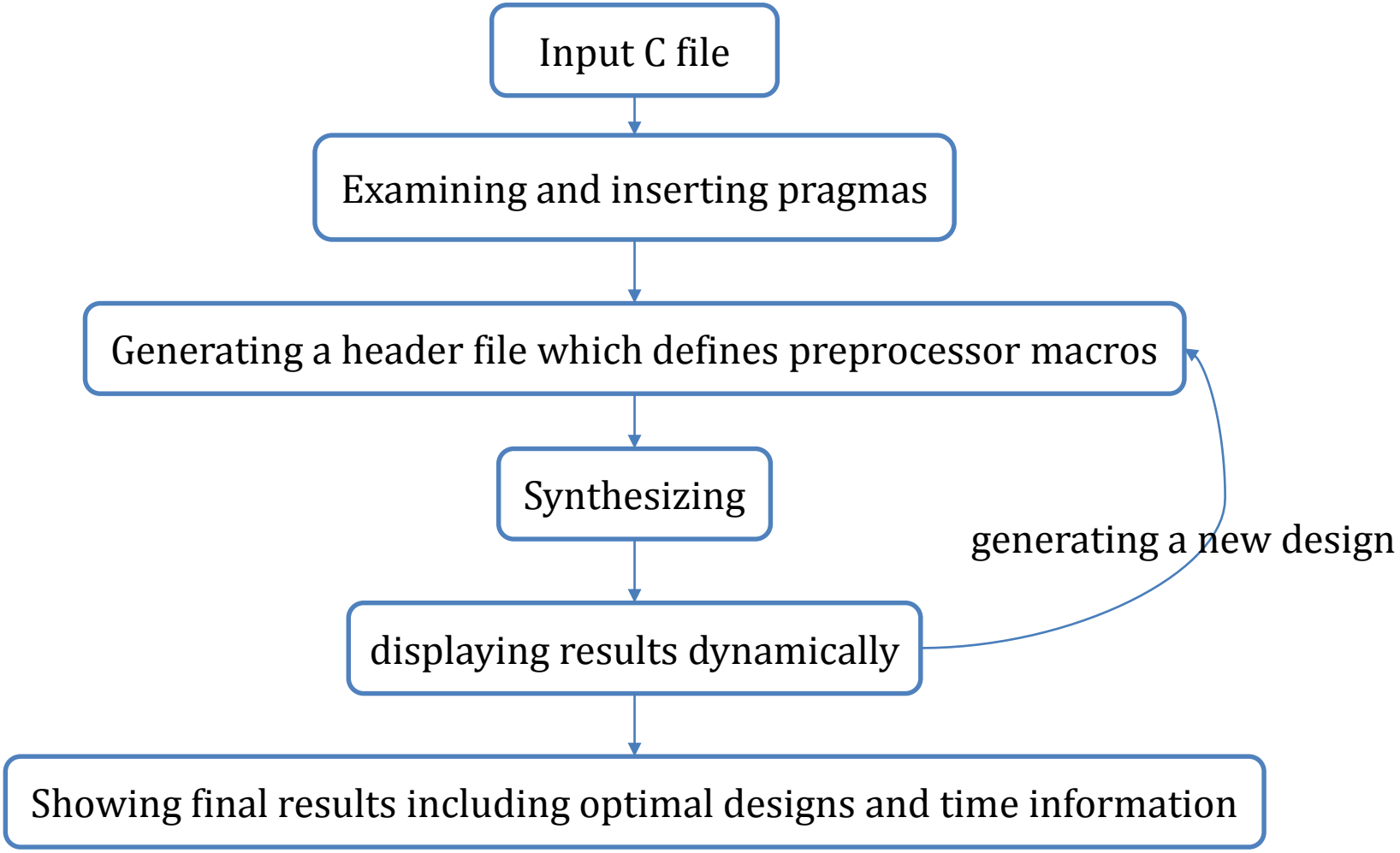
high-level synthesis process

$$(Area, latency) = f(x1, x2, x3 \dots)$$

synthesis results
design curve



Program structure

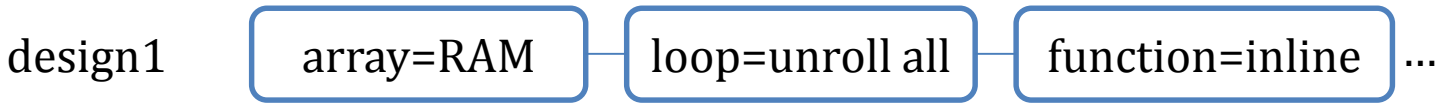


Algorithms

- Brute Force(exhaustion method)
  - a generate-and-test algorithm to check all possible candidates that satisfy specification of a problem
- Simulated Annealing
  - probabilistic heuristic
  - other heuristics(e.g. genetic algorithm)

Simulated Annealing

- Step1: Generate a initial design randomly as the base design and synthesize it. Synthesized result is used to calculate first GCF as base state of system. Set an initial temperature for the system.

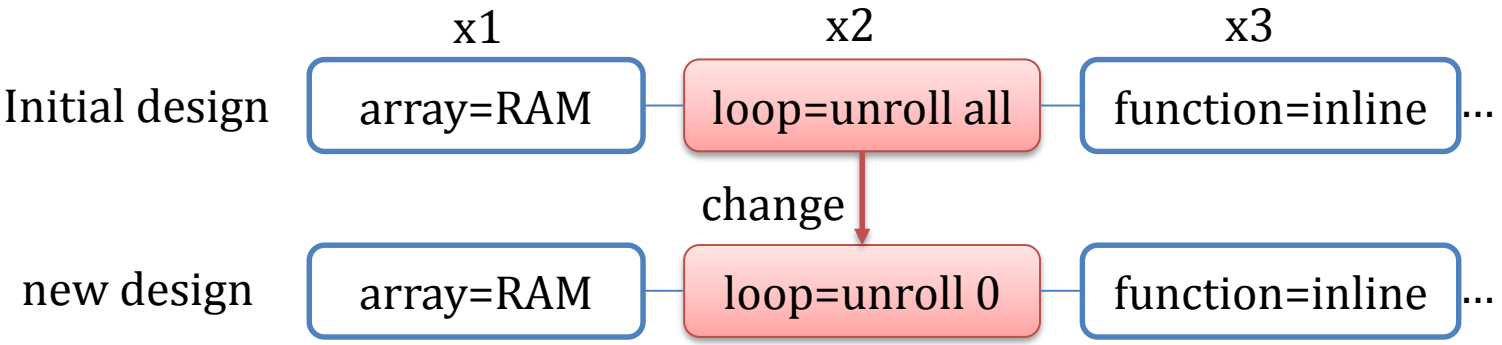


- Global cost function is defined as:

$$GCF = \partial A + \beta L$$

Simulated Annealing

- Step 2: Generate a new design from base design by randomly modifying one attribute's value.



- Step3: Compare new GCF and previous GCF. Then, determine whether to accept the new design. Probability to accept a worse design:

$$P = e^{-\frac{\Delta GCF}{k}}$$

Simulated Annealing

- Step4: If 5 better designs are consecutively generated, reduce current temperature by 10%. Change parameters in GCF for every 8 designs.

- If no exit condition is met, iterate from step2 to step 4. Exit iteration if one exit condition is met.

Simulated Annealing

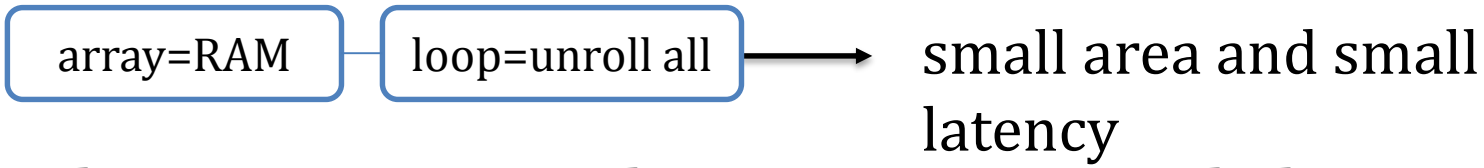
- Exit conditions:
  i. Current temperature is less than threshold.
  ii. Consecutively, more than 5 new designs are worse than previous design.
  iii. Cannot generate new designs by changing one of the attributes.
  iv. Synthesized designs are more than 70% of all designs.

Simulated Annealing

- Why SA?

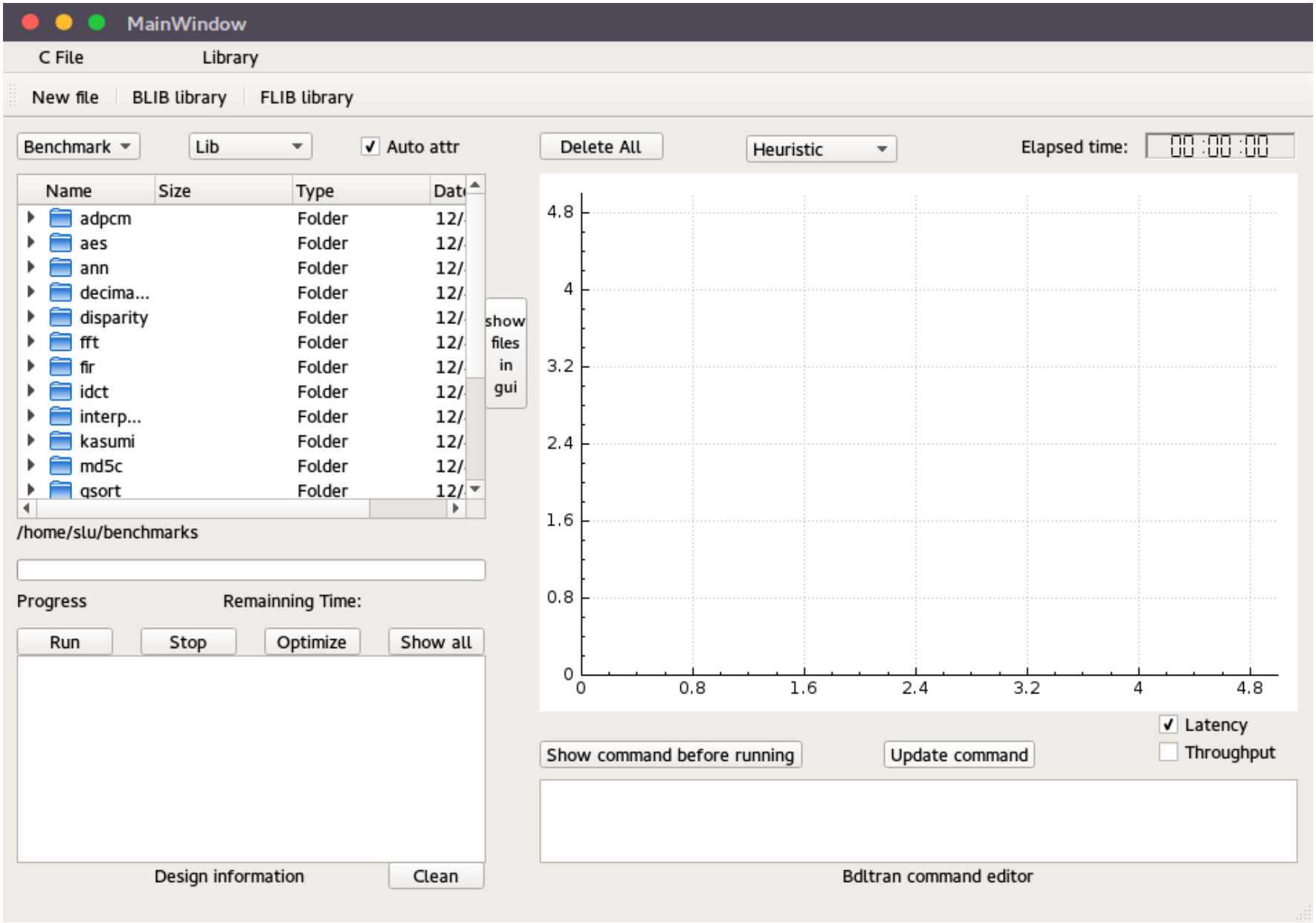
$$(Area, latency) = f(x1, x2, x3\ldots\ldots)$$

- Some input sequence might lead to a good result. e.g.



- Change one attribute at a time while maintaining other attribute combinations

Graphical User Interface for Design Space Exploration

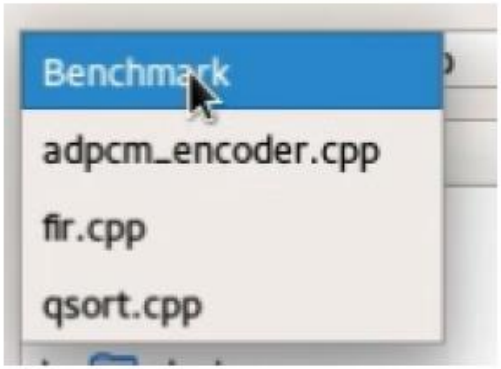

MainWindow
C File
Library
New file
BLIB library
FLIB library
Benchmark
Lib
Auto attr
Delete All
Heuristic
Elapsed time:
00 :00 :00
Name
Size
Type
Dat
adpcm
aes
ann
decima...
disparity
fft
fir
idct
interp...
kasumi
md5c
qsort
Folder
12/
show files in gui
/home/slu/benchmarks
Progress
Remainning Time:
Run
Stop
Optimize
Show all
Design information
Clean
4.8
4
3.2
2.4
1.6
0.8
0
0
0.8
1.6
2.4
3.2
4
4.8
Latency
Throughput
Show command before running
Update command
Bdltran command editor

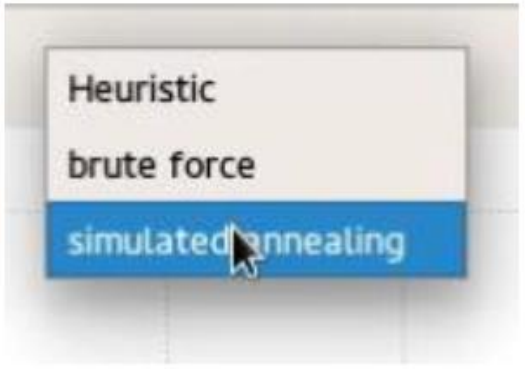
Qt Framework

- A cross-platform framework for developing applications
- Signals and Slots: Communication mechanism between different parts of the program
- QCustomPlot: An online open source widget for plotting
- Multithread programming: To prevent GUI freezing
- Model/View Programming: To modify data outside current program(used in file list)

Functions of the developed GUI

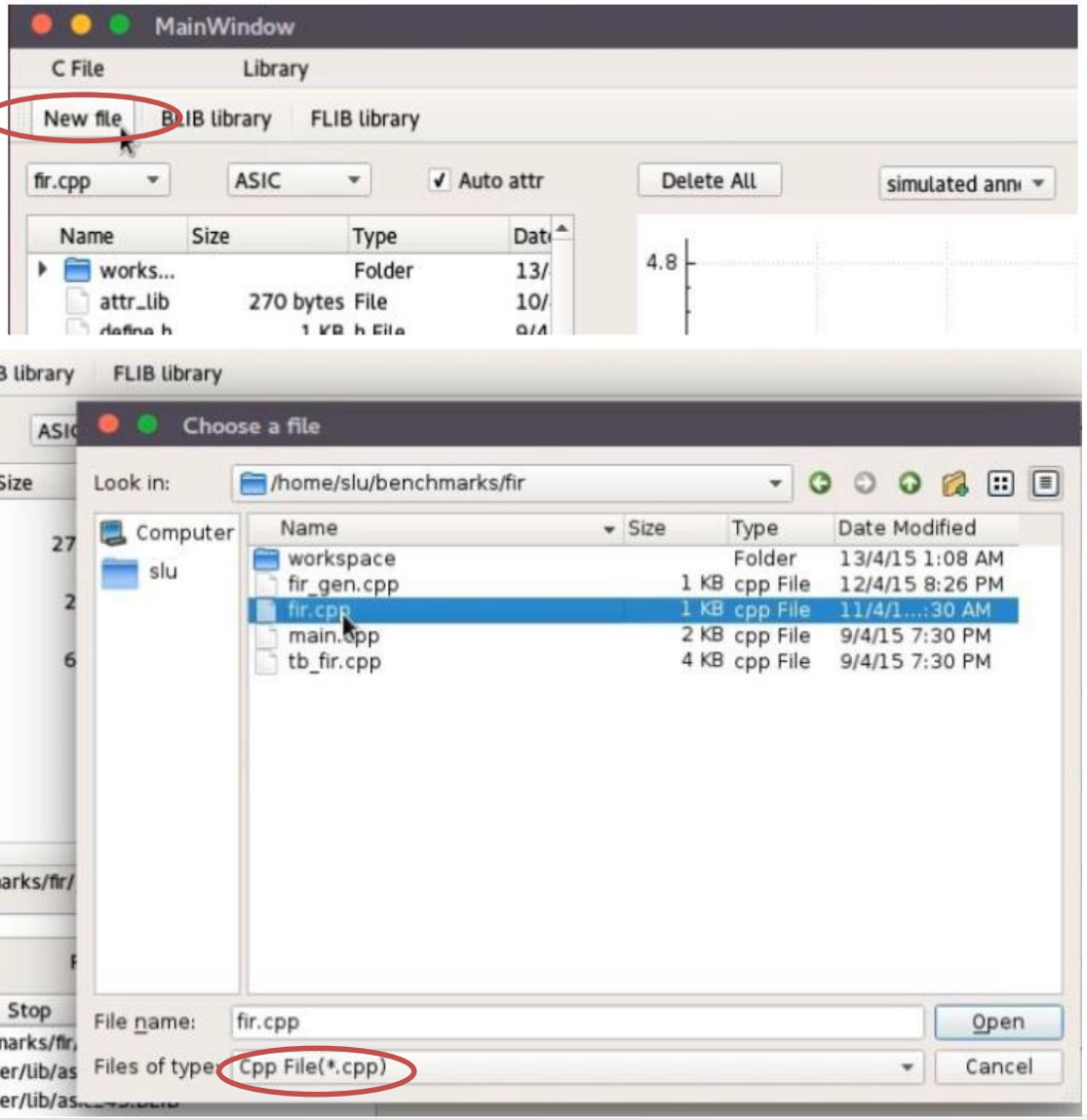
- ComboBox
- Selecting files or options





Functions of the developed GUI

- Selecting other benchmarks
- Selecting other libraries
- Filtering out files with wrong extensions



Functions of the developed GUI

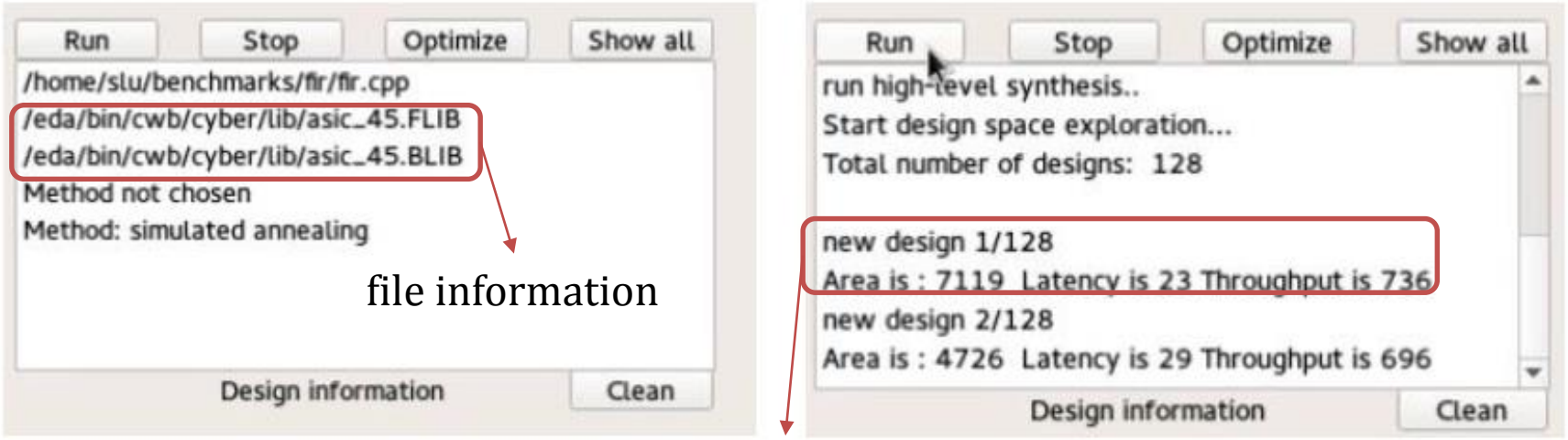
- Automatic attribute insertion

- To examine input SystemC file and insert pragma automatically based on syntax.

```
sc_uint<8> in_data_read[9]/*attr1*/;
sc_uint<16> coeff_read[9]/*attr2*/;
```

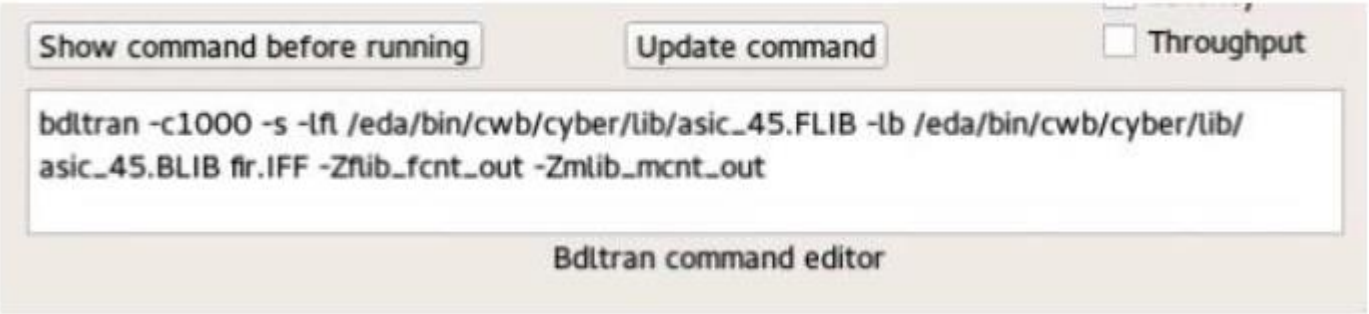
Functions of the developed GUI

- To show selection information such as which technology library has been selected.
- To Show design information dynamically while running, like that total number of designs, current number of designs and synthesis results.

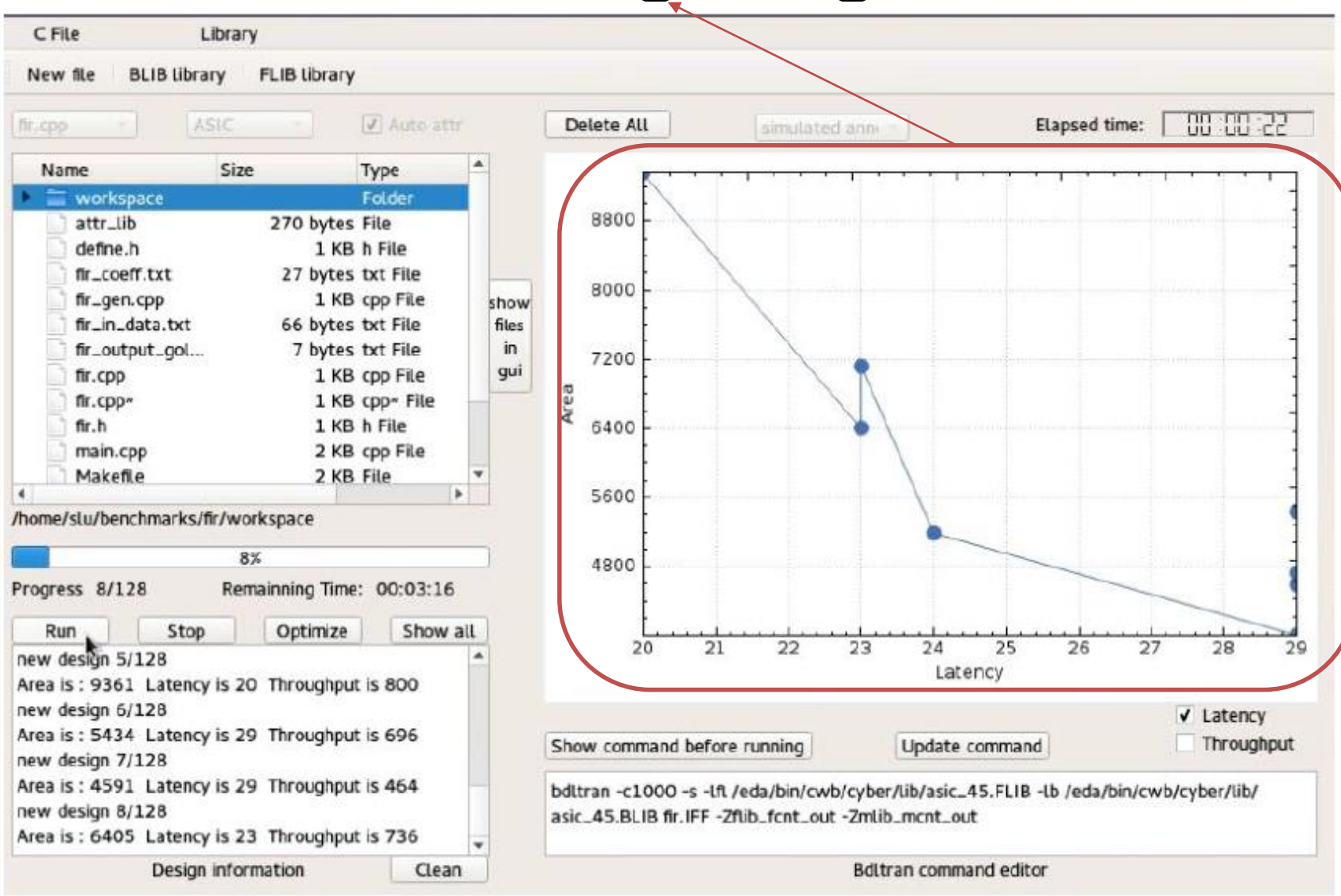


Functions of the developed GUI

- Embedded editor
- To edit synthesis command
- Standard shortcuts for editor like copy, cut and paste can work

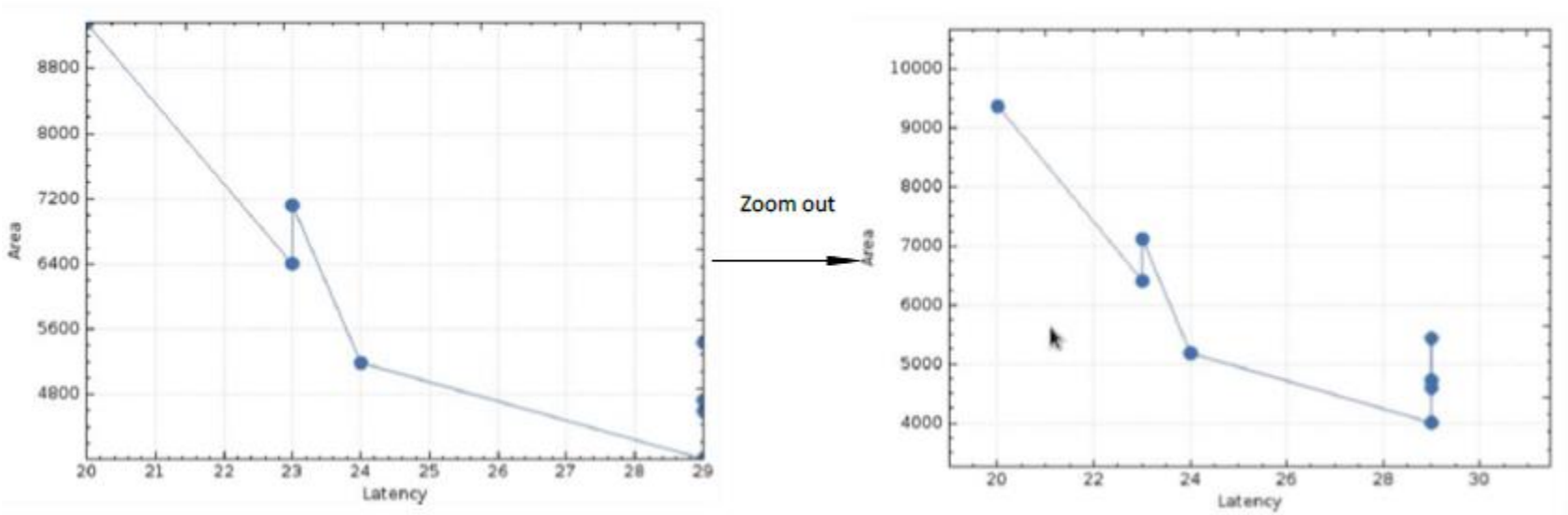


Plotting widget



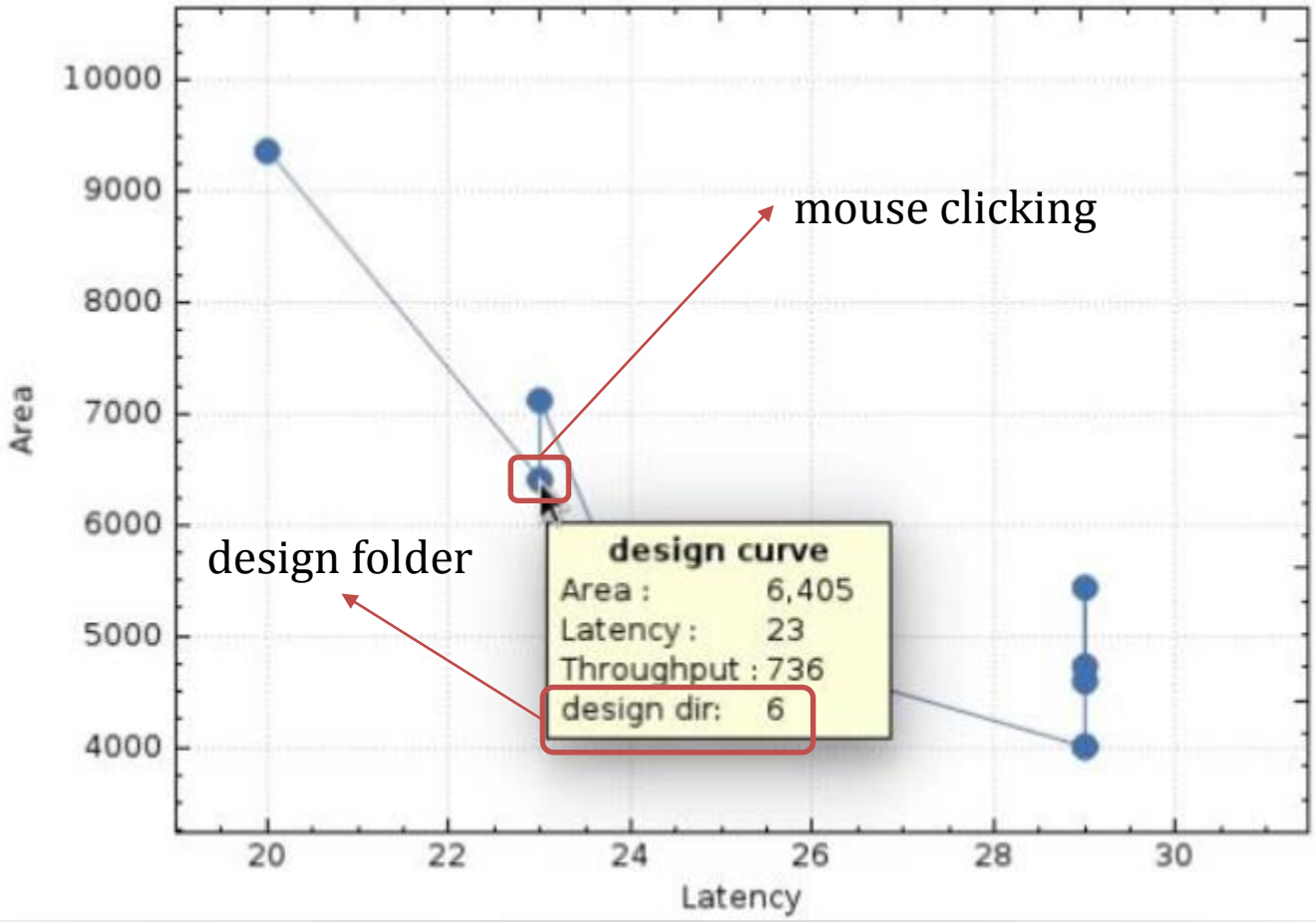
Plotting widget

- Zoom out and zoom in using mouse wheel



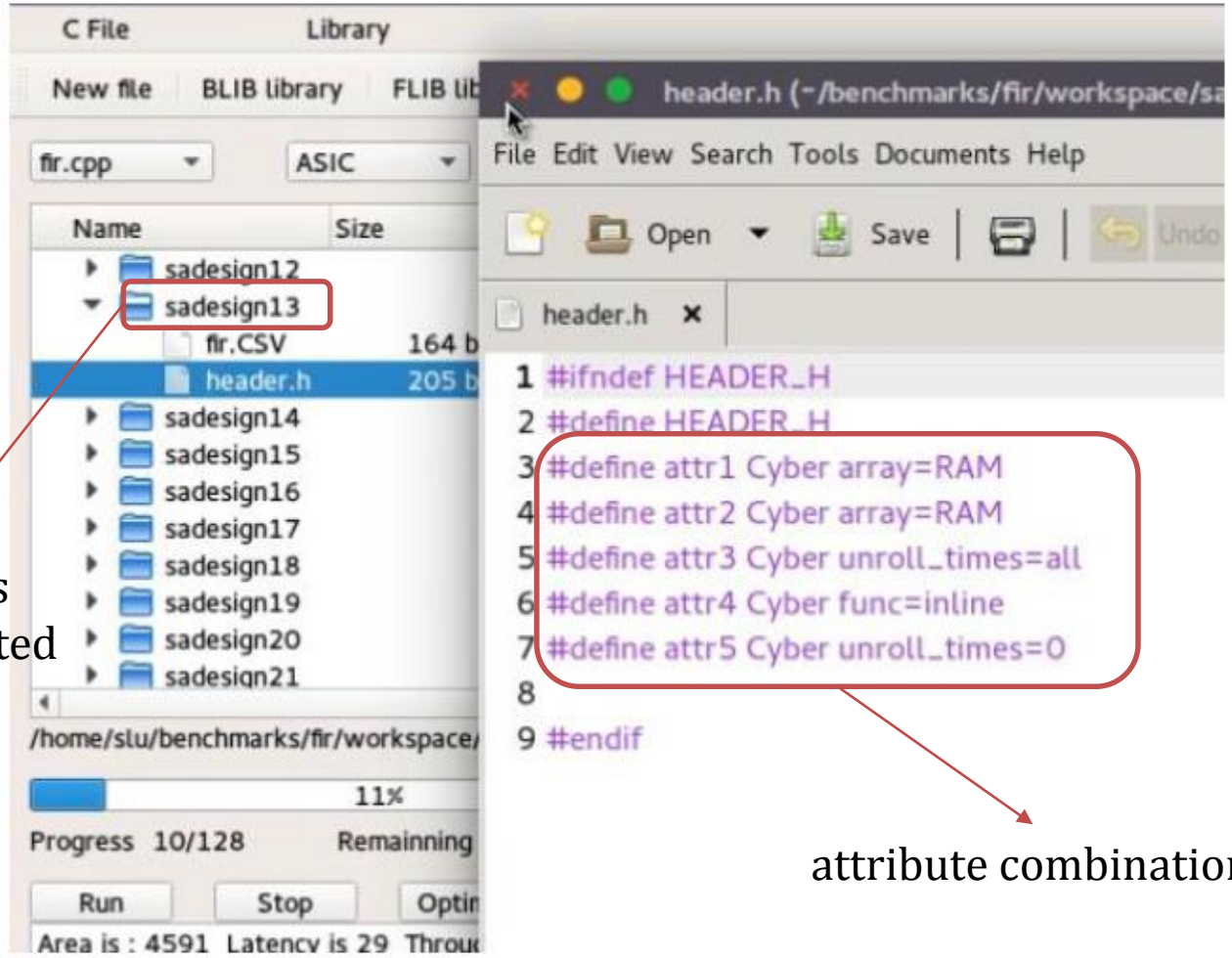
Plotting widget functions

- Interactivity
- Mouse clicking



Plotting widget functions

- File list



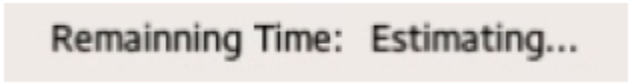
"sa" stands for simulated annealing

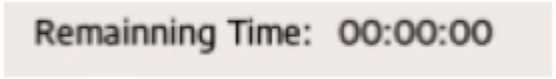
attribute combination

Timer

- Elapsed time

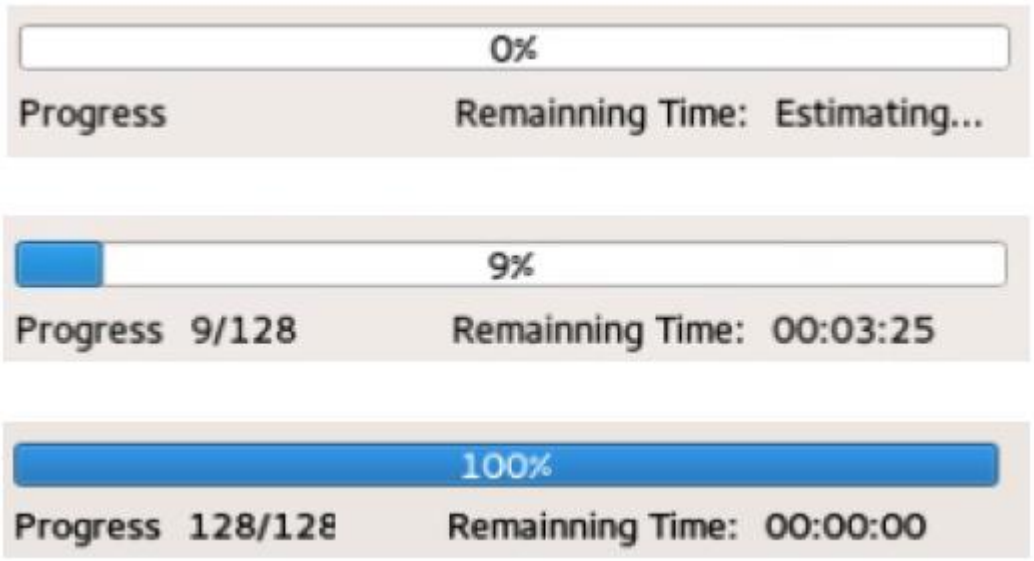
- Remaining time





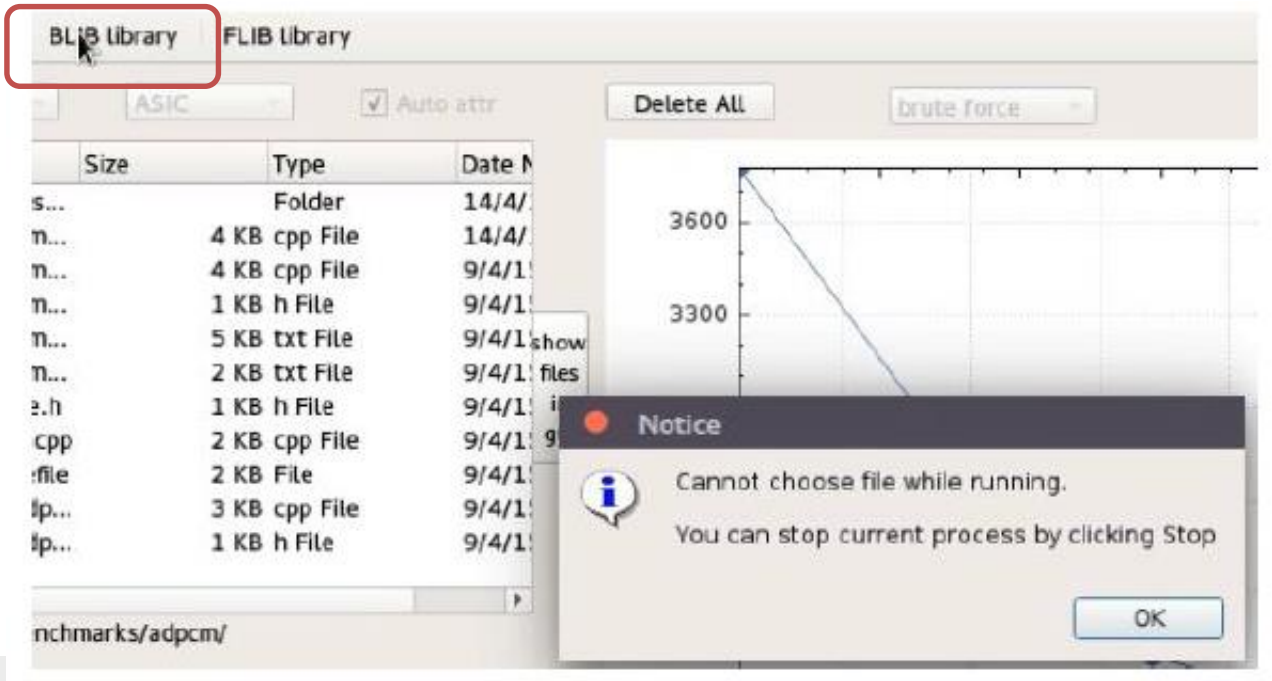
Progress Bar

- To show progress
- Relatively accurate for brute force
- Not useful if simulated annealing method is used



Conflict handling

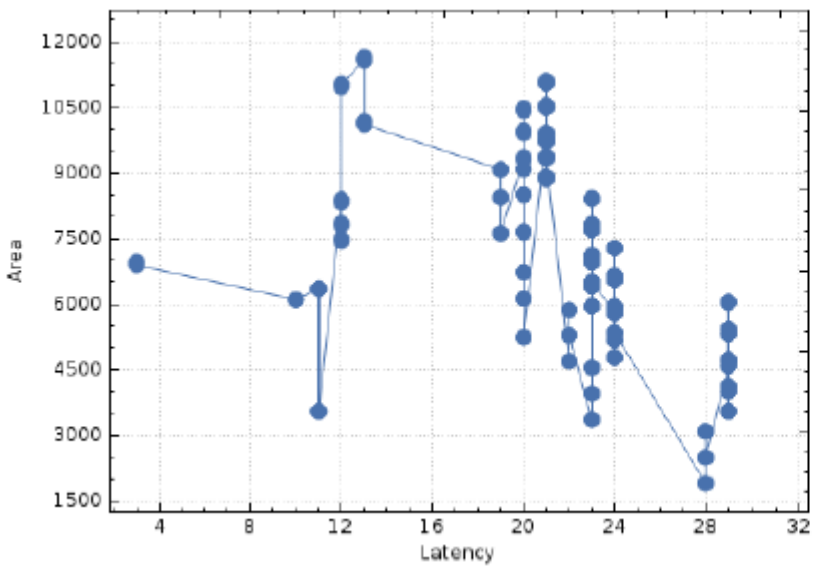
- Conflicts between widgets in program sometimes appear and render program's crashing
- Mechanisms to check conflicts and pop up warning messages

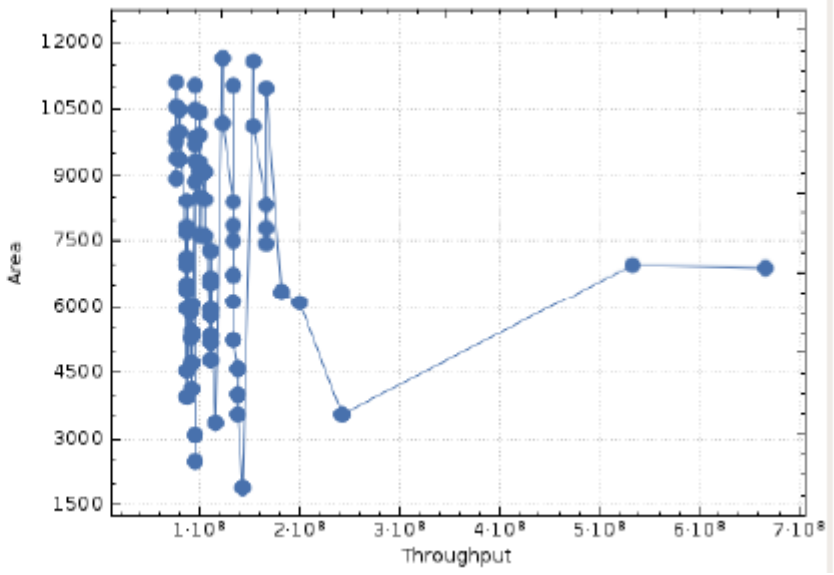


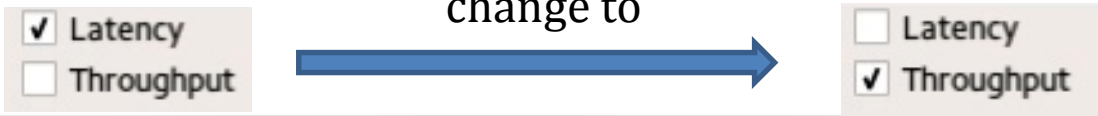
Change Coordinate

- Change to area versus throughput

$$Throughput = Output\_port\_number \times \frac{1}{CP\_delay \times Latency}$$

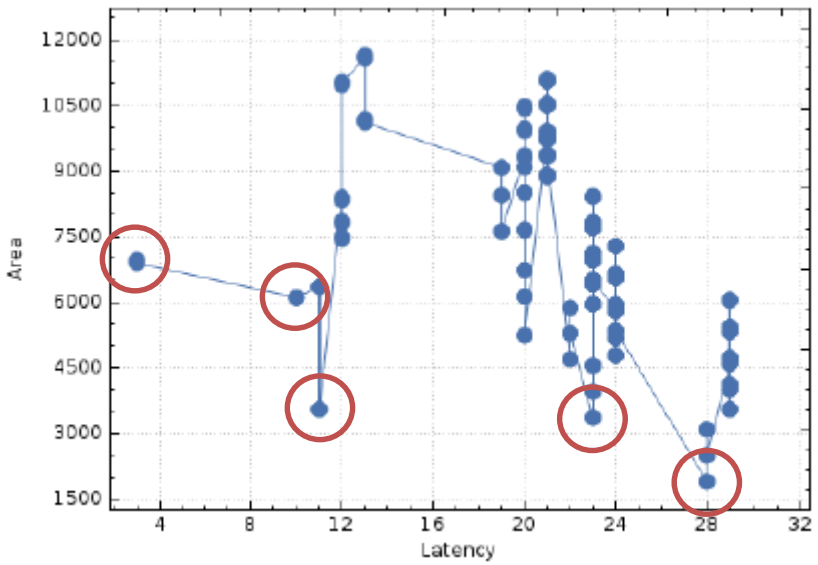


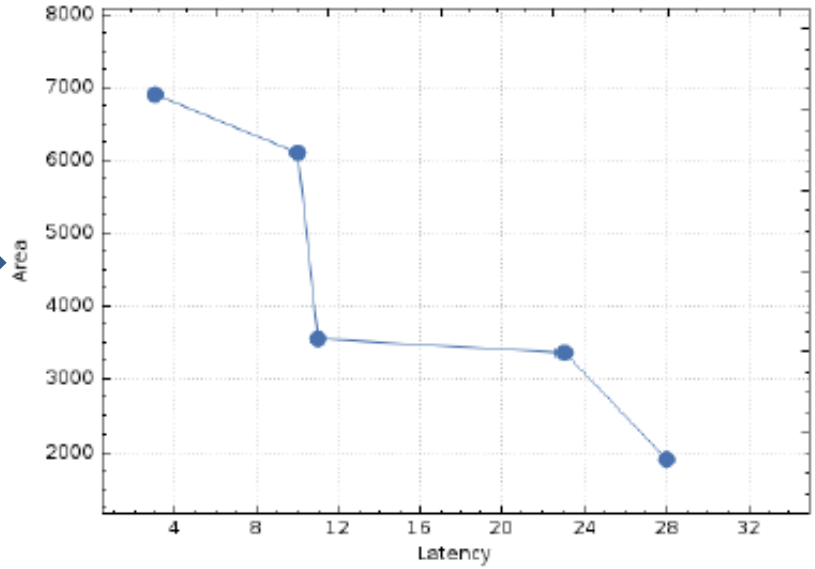




Show optimal designs

- Click button 



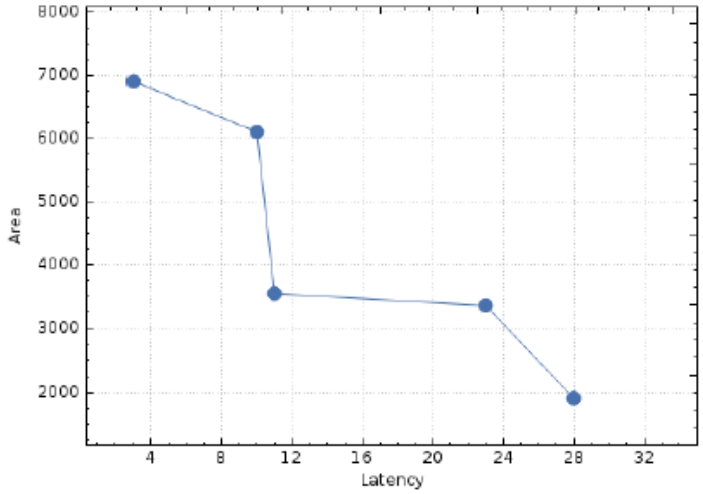


Synthesis Results

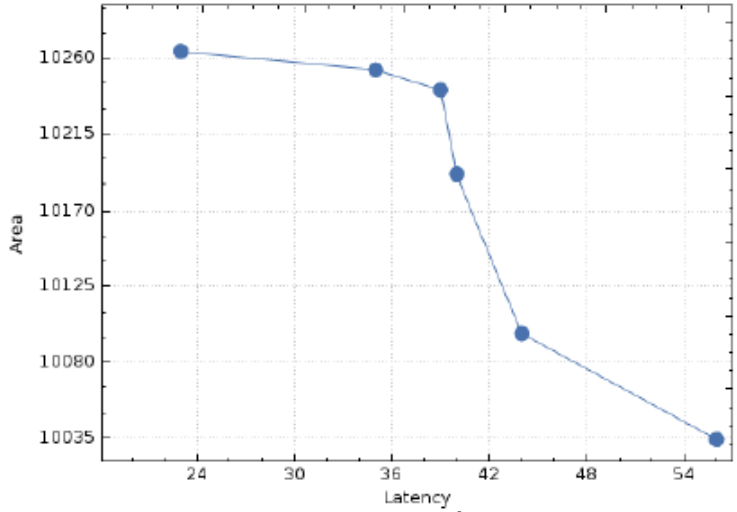
- Three benchmarks

Bench	Type	#lines	Explorable operations	Brute	SA
fir	C	86	array(2), loop(2), function(1)	340s	55s
qsort	C	119	array(1), loop(2),function(3)	843s	64s
adpcm_encoder	C	179	array(1),for(1),function(2)	130s	54s

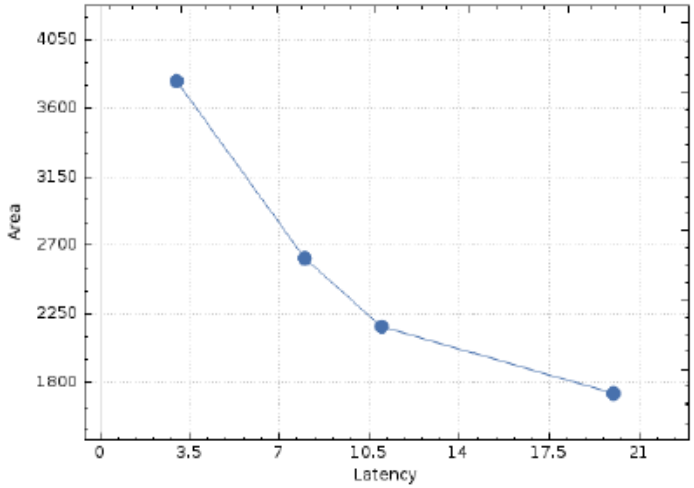
Brute Force Results



fir

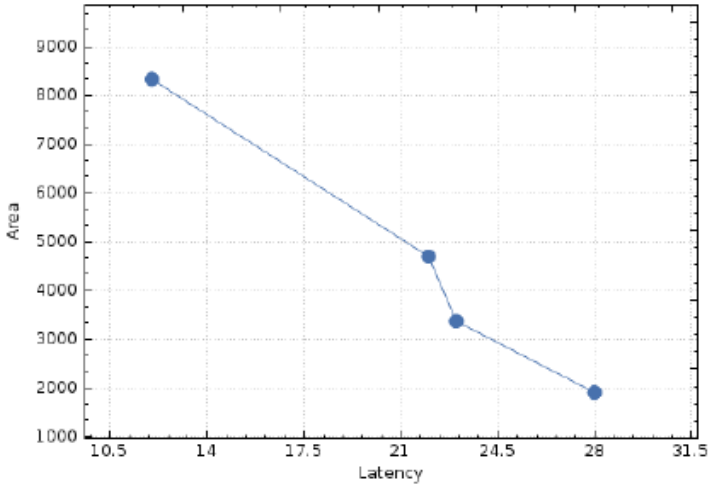


qsort

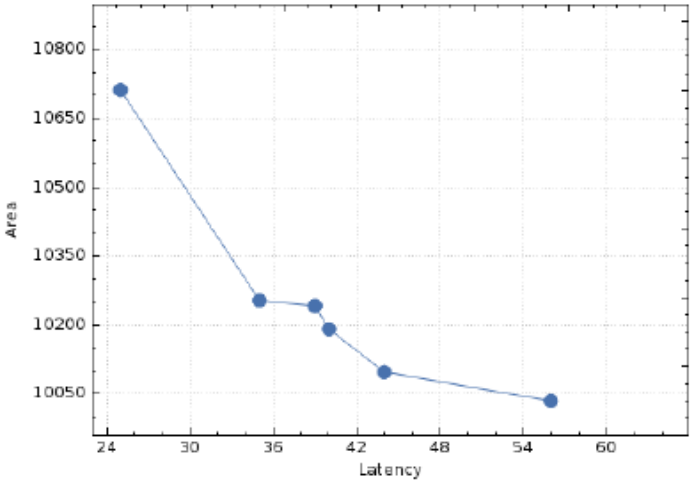


adpcm_encoder

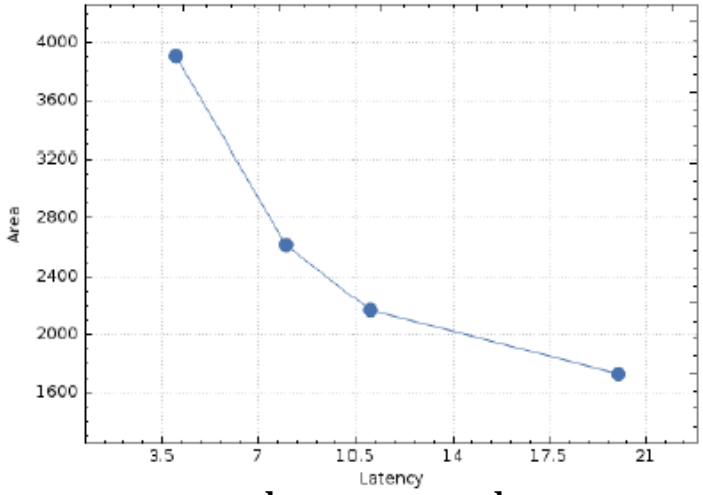
Simulated Annealing Results



fir



qsort



adpcm_encoder

Comparison

- Time Comparison
- Averagely, running time of simulated annealing algorithm for these three benchmarks is 21.8 percent of brute force algorithm, which means SA algorithm's speed is 4.59 times of BF algorithm's speed.

Bench	Brute force	Simulated annealing	SA versus BF
fir	340s	55s	16.2%
qsort	843s	64s	7.6%
adpcm_enconder	130s	54s	41.5%

$$average = \frac{16.2 + 7.6 + 41.5}{3} = 21.8\%$$

$$\frac{1}{21.8\%} = 4.59$$

Comparison

- Qualitative comparison
- In these experiments, brute force method has gone through all designs. Therefore, pareto dominance for brute force is 100%
- It has been found that simulated annealing could find 66% pareto dominated points.

Bench	Brute force	Simulated annealing	SA versus BF
fir	5	2	40%
qsort	6	5	83%
adpcm_enconder	4	3	75%

$$average = \frac{40 + 83 + 75}{3} = 66\%$$

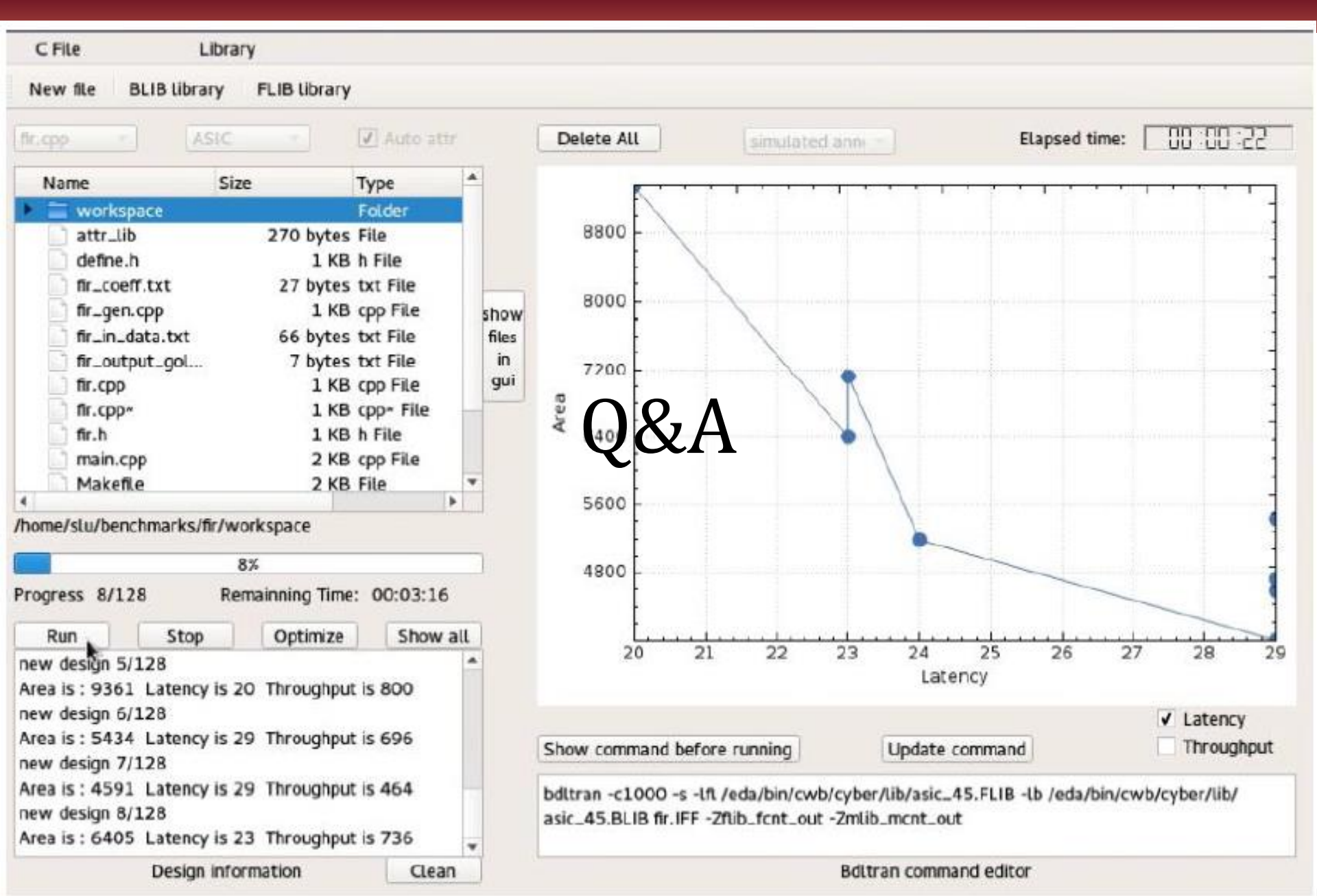
Conclusion

Achievements

- This project has completed the main goals. It has developed one heuristic (simulated annealing) for design space exploration that achieved around four times faster speed than brute force method.
- A graphical user interface that could display synthesis results dynamically was developed .

Limitations and future works

- Other heuristics can be developed and compared to simulated annealing
- Number of tested benchmarks is not enough
- Automatic attribute insertion could only support special language formats


C File
Library
New file
BLIB library
FLIB library
Delete All
Elapsed time:
Name
Size
Type
workspace
Folder
attr_lib 270 bytes File
define.h 1 KB h File
fir_coeff.txt 27 bytes txt File
fir_gen.cpp 1 KB cpp File
fir_in_data.txt 66 bytes txt File
fir_output_gol... 7 bytes txt File
fir.cpp 1 KB cpp File
fir.h 1 KB h File
main.cpp 2 KB cpp File
Makefile 2 KB File
show files in gui
/home/slu/benchmarks/fir/workspace
8%
Progress 8/128
Remainning Time: 00:03:16
Run
Stop
Optimize
Show all
new design 5/128
Area is : 9361 Latency is 20 Throughput is 800
new design 6/128
Area is : 5434 Latency is 29 Throughput is 696
new design 7/128
Area is : 4591 Latency is 29 Throughput is 464
new design 8/128
Area is : 6405 Latency is 23 Throughput is 736
Design information
Clean
8800
8000
5600
4800
Area
20
21
22
23
24
25
26
27
28
29
Latency
Q&A
Latency
Throughput
Show command before running
Update command
bdltran -c1000 -s -lfl /eda/bin/cwb/cyber/lib/asic_45.FLIB -lb /eda/bin/cwb/cyber/lib/asic_45.BLIB fir.IFF -Zflib_fcnt_out -Zmlib_mcnt_out
Bdltran command editor