Project Update

Team #302: Asynchronous Ballistic Reversible Superconducting (ABRS) Computing

Frank Allen, Oscar Lopez, James Hardy, and Fadi Matloob

Agenda

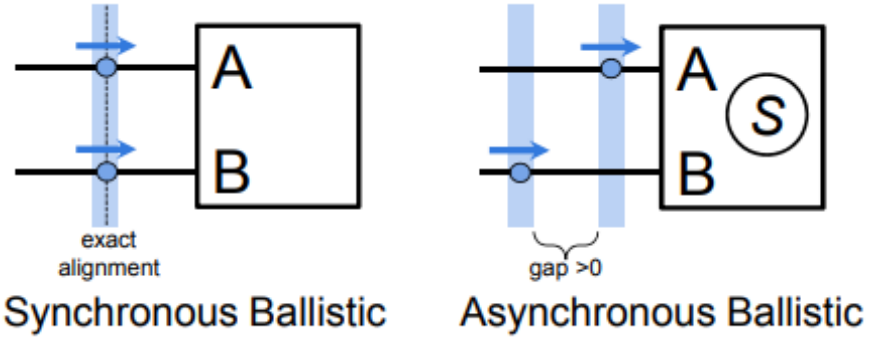
- Summary
  - Project Scope
  - Function Decomposition
- Progress Breakdown
  - Walking Through Components
- Challenges
  - Bugs & Difficulties
- Next Steps
  - Plan For Progress

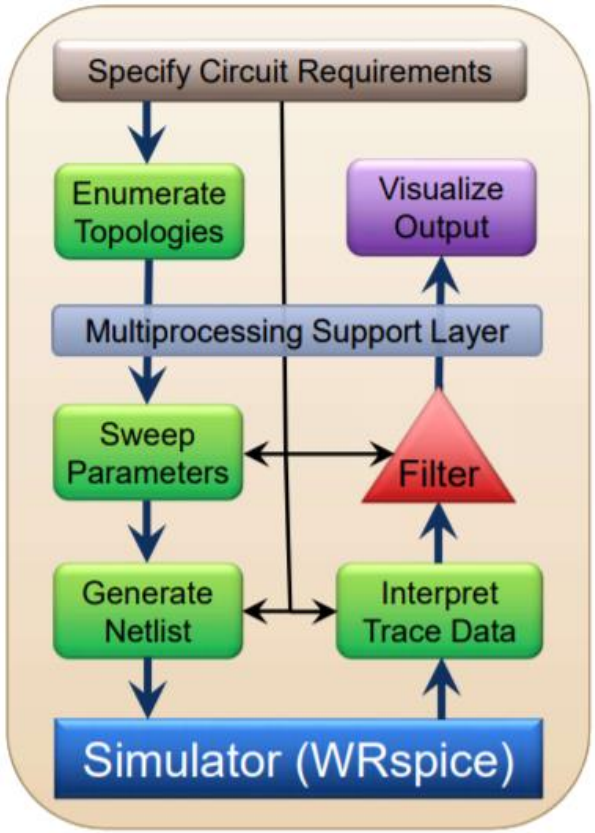
Outline

- Summary
  - Project Scope
  - Function Decomposition
- Progress Breakdown
  - Walking Through Components
- Challenges
  - Bugs & Difficulties
- Next Steps
  - Plan For Progress

Project Scope

- Project Goal:
  - Create a software tool to find functioning asynchronous reversible superconducting circuits
- Key Requirements:
  - Generate Circuit Topologies
  - Sweep Possible Device Values
  - Interpret and Filter the Results
  - Visualize Functional Circuits



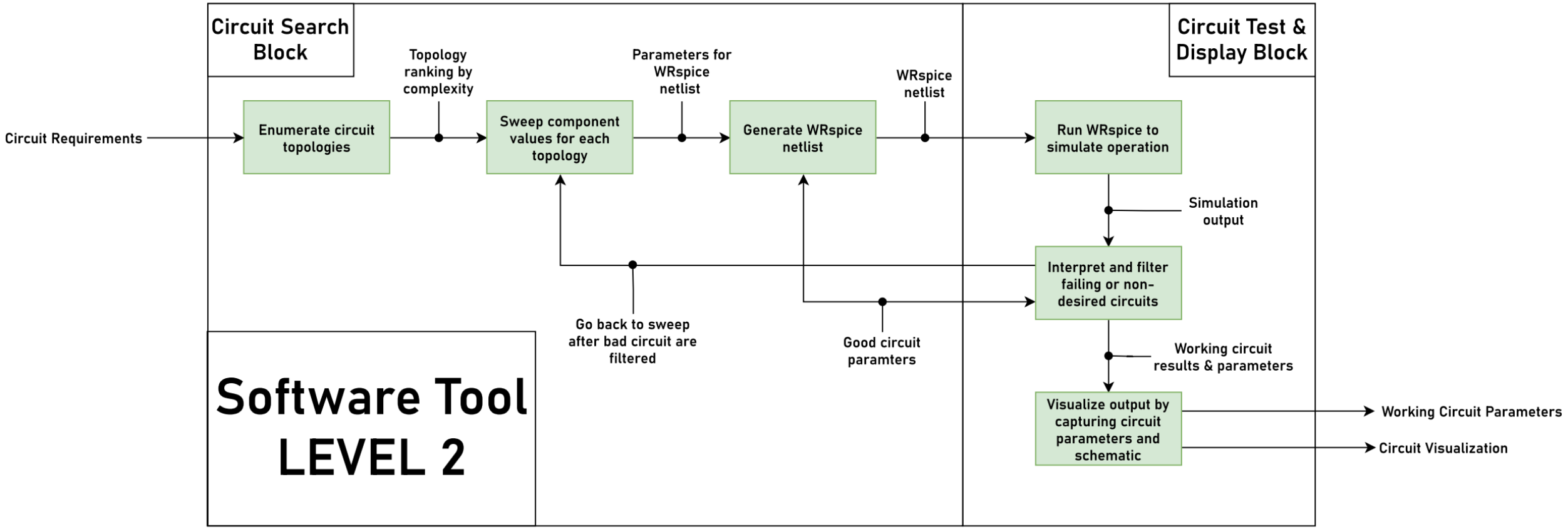


Fadi Matloob

Outline

- Summary
  - Project Scope
  - Function Decomposition
- Progress Breakdown
  - Walking Through Components
- Challenges
  - Bugs & Difficulties
- Next Steps
  - Plan For Progress

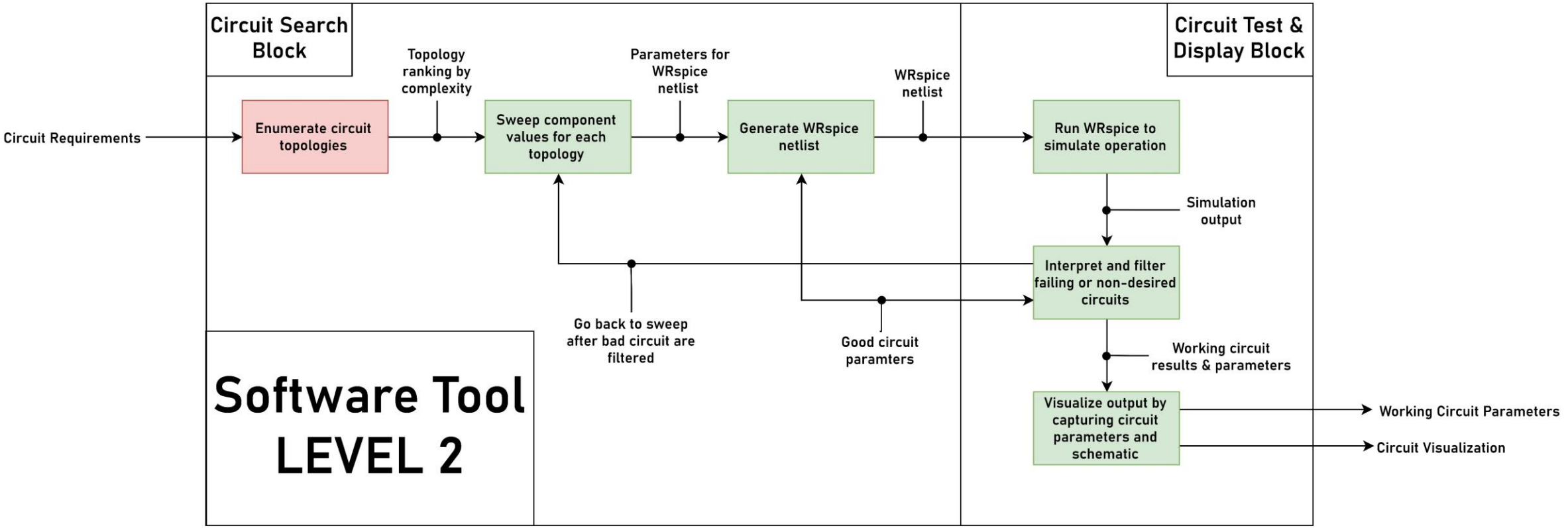
Functional Decomposition Cont.



Outline

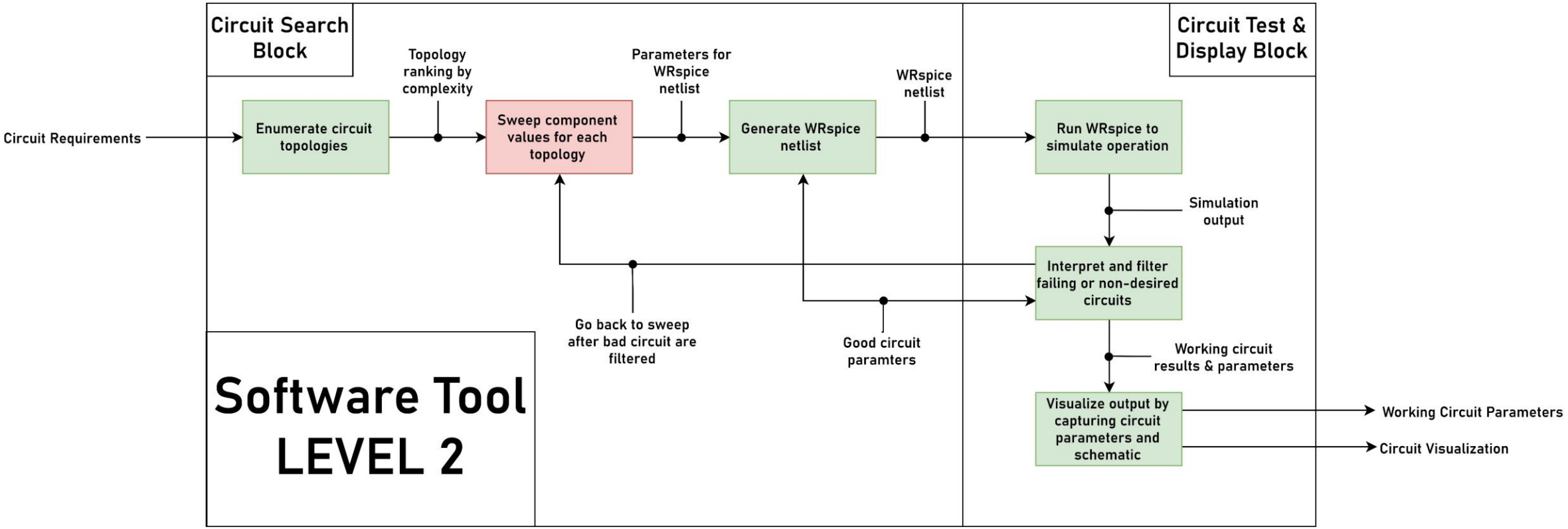
- Summary
  - Project Scope
  - Function Decomposition
- Progress Breakdown
  - Walking Through Components
- Challenges
  - Bugs & Difficulties
- Next Steps
  - Plan For Progress

Walking Through Components – Clickable



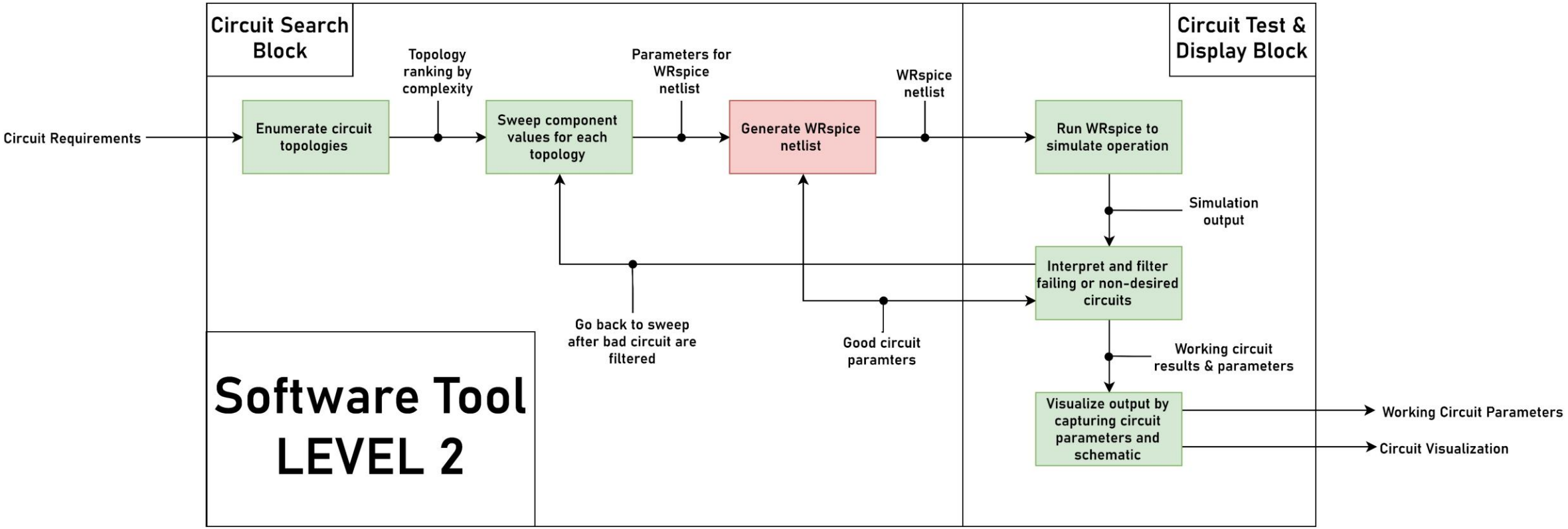
James Hardy

Walking Through Components Cont.



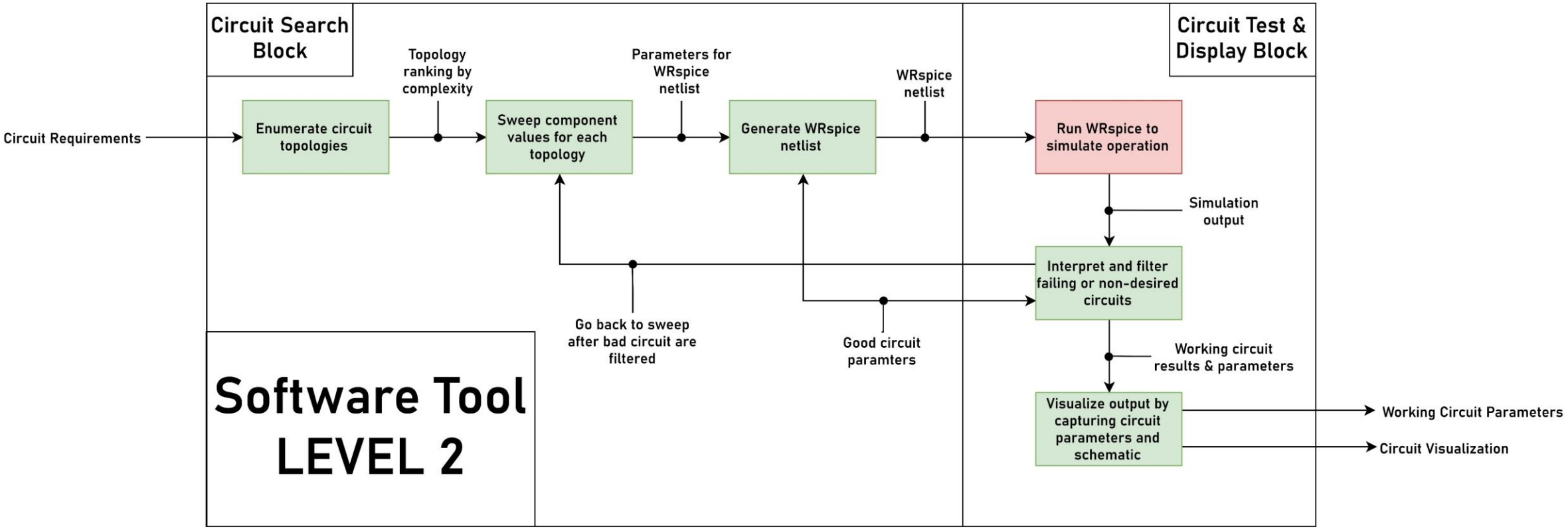
James Hardy

Walking Through Components Cont.



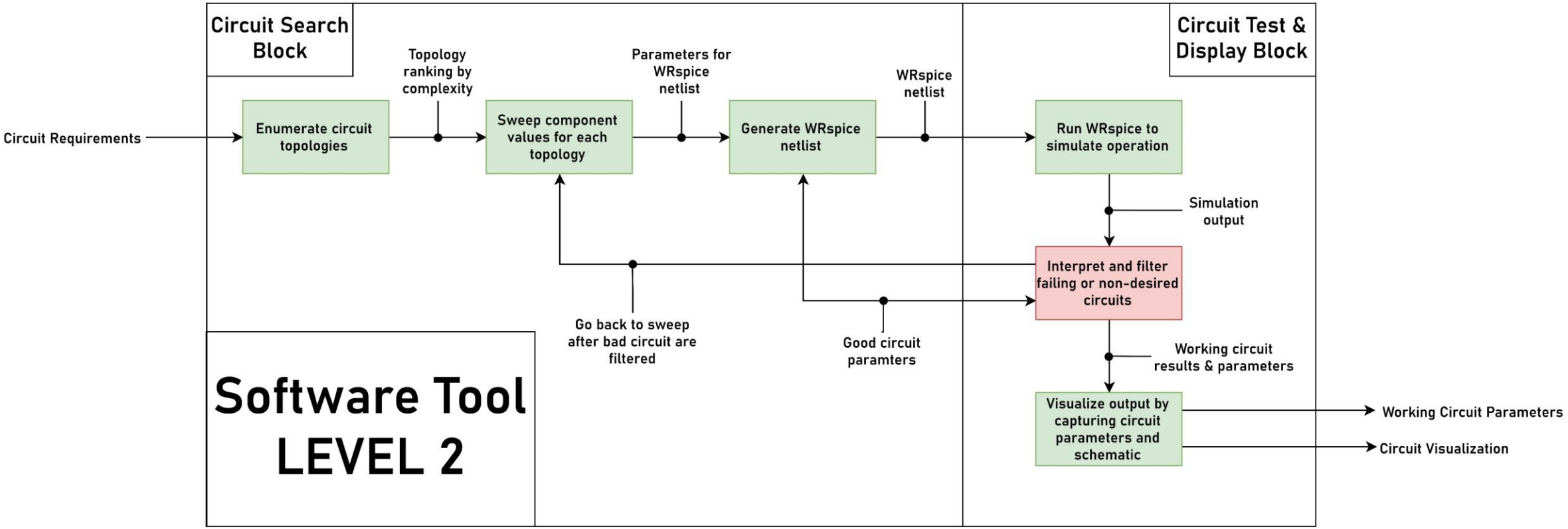
Frank Allen

Walking Through Components Cont.

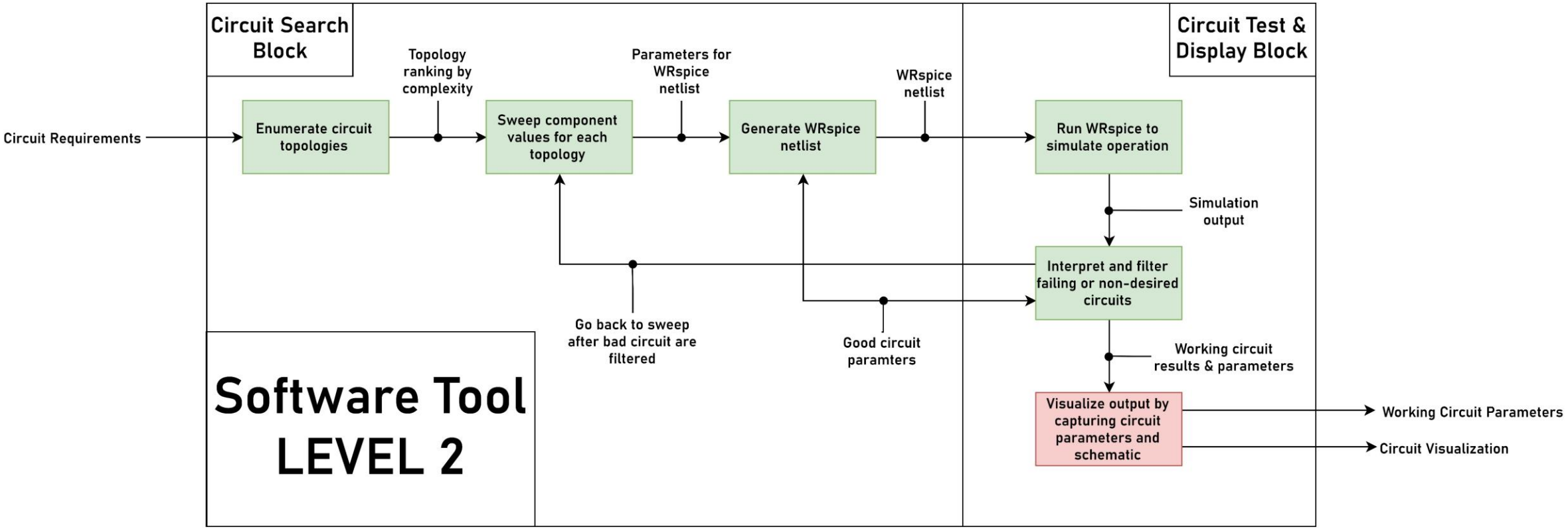


Frank Allen

Walking Through Components Cont.



Walking Through Components Cont.

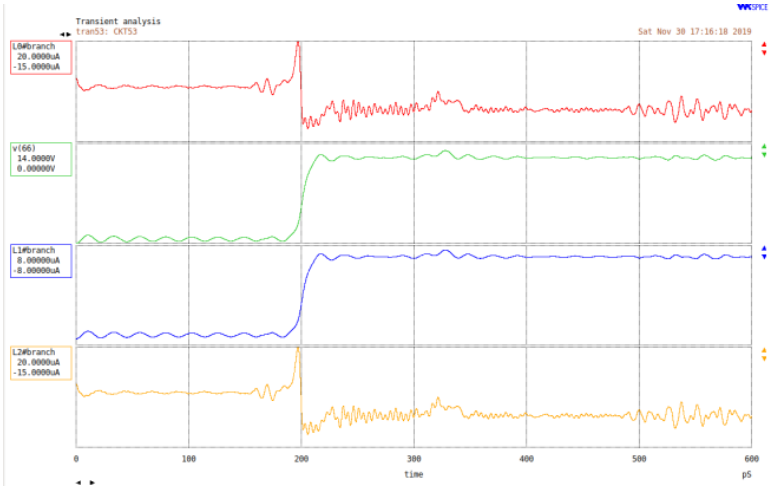


Outline

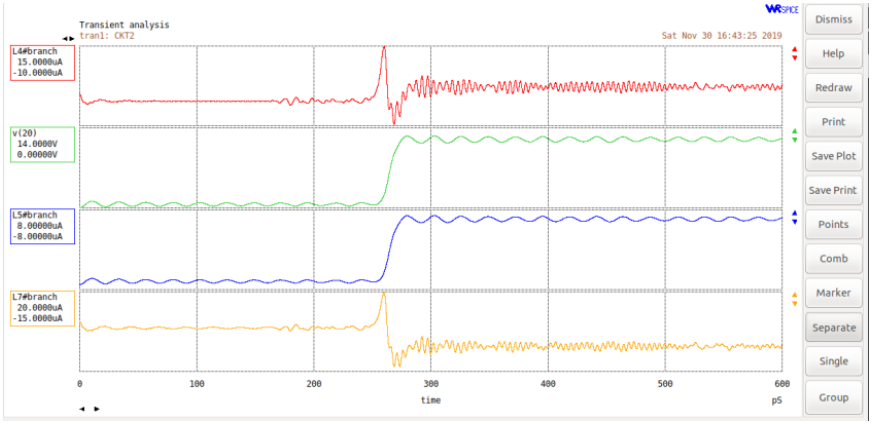
- Summary
  - Project Scope
  - Function Decomposition
- Progress Breakdown
  - Walking Through Components
- Challenges
  - Bugs & Difficulties
- Next Steps
  - Plan For Progress

Bugs & Difficulties

- Combinational circuits
  - Series & Parallel
- Sweeping values
  - Algorithm
- Simulation
  - Matching output
  - Testing automation
- Filtering results
  - Fluxon existence only



Generated Circuit



Existing Circuit

Frank Allen

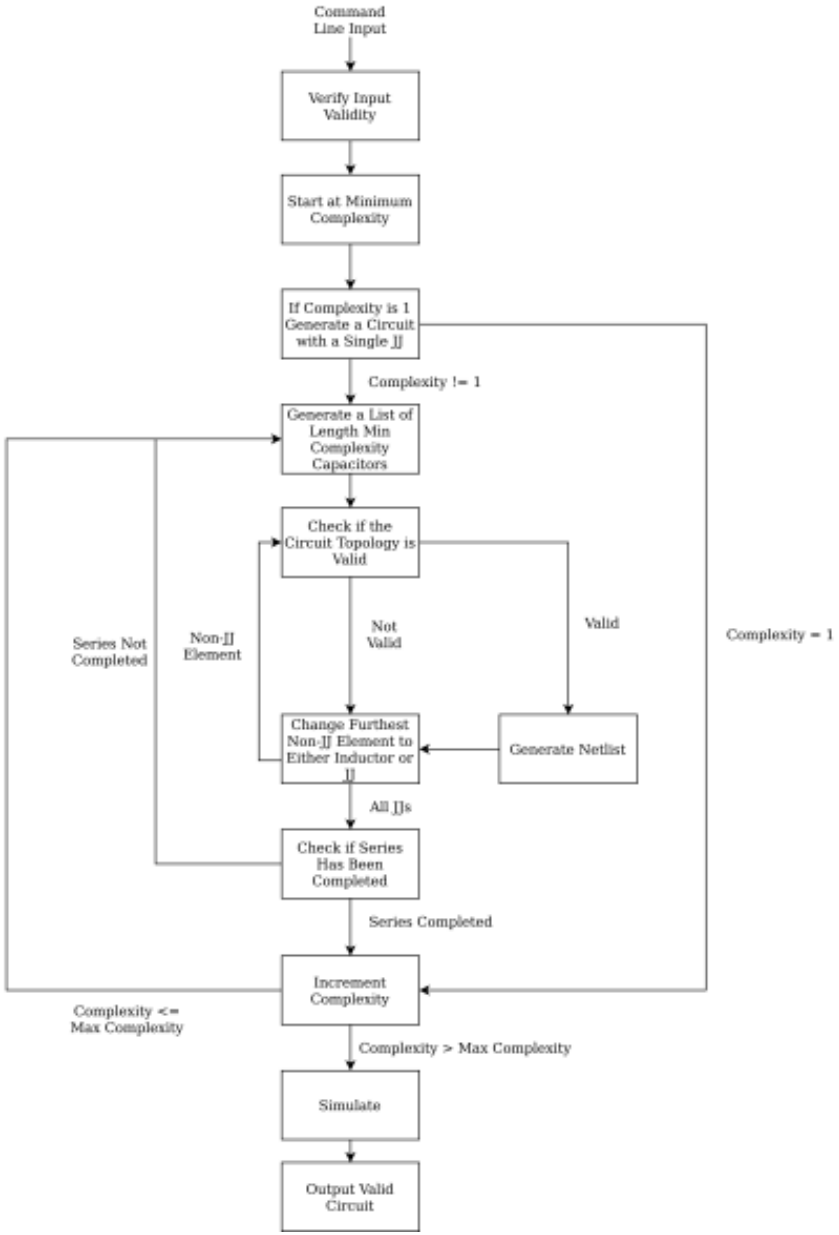
Outline

- Summary
  - Project Scope
  - Function Decomposition
- Progress Breakdown
  - Walking Through Components
- Challenges
  - Bugs & Difficulties
- Next Steps
  - Plan For Progress

Plan For Progress

- Discussion with Michael Frank
  - Matching output
- Testing and validation
  - Parallel processing
- Visual output

Code


Command Line Input
Verify Input Validity
Start at Minimum Complexity
If Complexity is 1 Generate a Circuit with a Single JJ
Complexity != 1
Generate a List of Length Min Complexity Capacitors
Check if the Circuit Topology is Valid
Not Valid
Valid
Complexity = 1
Series Not Completed
Non-JJ Element
Change Furthest Non-JJ Element to Either Inductor or JJ
Generate Netlist
All JJs
Check if Series Has Been Completed
Series Completed
Increment Complexity
Complexity <= Max Complexity
Complexity > Max Complexity
Simulate
Output Valid Circuit


testfile0 testfile1 testfile2 testfile3 testfile4 testfile5 testfile6 testfile7 testfile8 testfile9 testfile10 testfile11 testfile12
testfile13 testfile14 testfile15 testfile16 testfile17 testfile18 testfile19 testfile20 testfile21 testfile22 testfile23 testfile24 testfile25
testfile26 testfile27 testfile28 testfile29 testfile30 testfile31 testfile32 testfile33 testfile34 testfile35 testfile36 testfile37 testfile38
testfile39 testfile40 testfile41 testfile42 testfile43 testfile44 testfile45 testfile46 testfile47 testfile48 testfile49 testfile50 testfile51
testfile52 testfile53 testfile54 testfile55 testfile56 testfile57 testfile58 testfile59 testfile60 testfile61 testfile62 testfile63 testfile64
testfile65 testfile66 testfile67 testfile68 testfile69 testfile70 testfile71 testfile72 testfile73 testfile74 testfile75 testfile76 testfile77
testfile78 testfile79 testfile80 testfile81 testfile82 testfile83 testfile84 testfile85 testfile86 testfile87 testfile88 testfile89 testfile90
testfile91 testfile92 testfile93 testfile94 testfile95 testfile96 testfile97 testfile98 testfile99 testfile100 testfile101 testfile102 testfile103
testfile104 testfile105 testfile106 testfile107 testfile108 testfile109 testfile110 testfile111 testfile112 testfile113 testfile114 testfile115 testfile116
testfile117 testfile118 testfile119 testfile120 testfile121 testfile122 testfile123 testfile124 testfile125 testfile126 testfile127 testfile128 testfile129

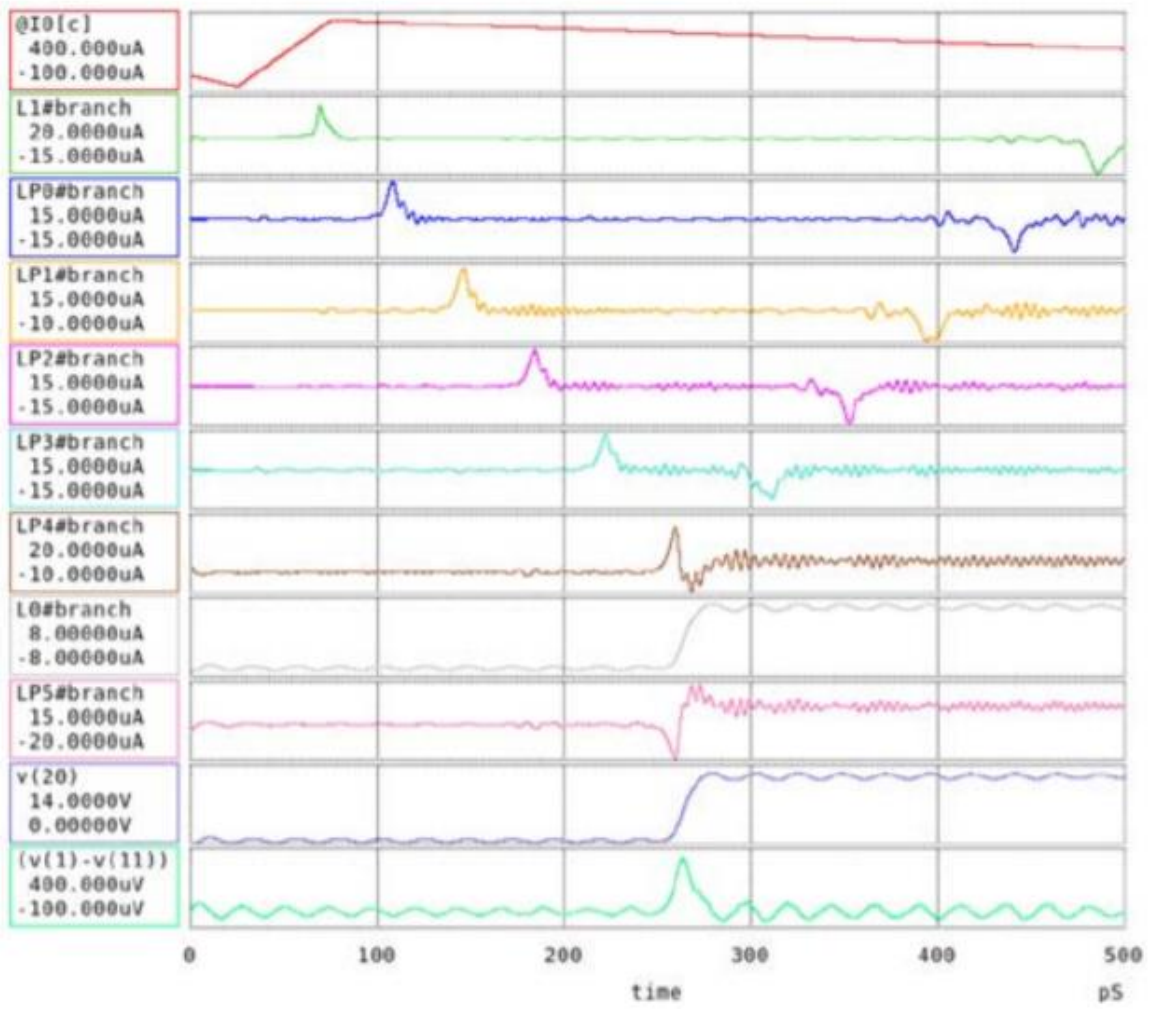
Frank Allen

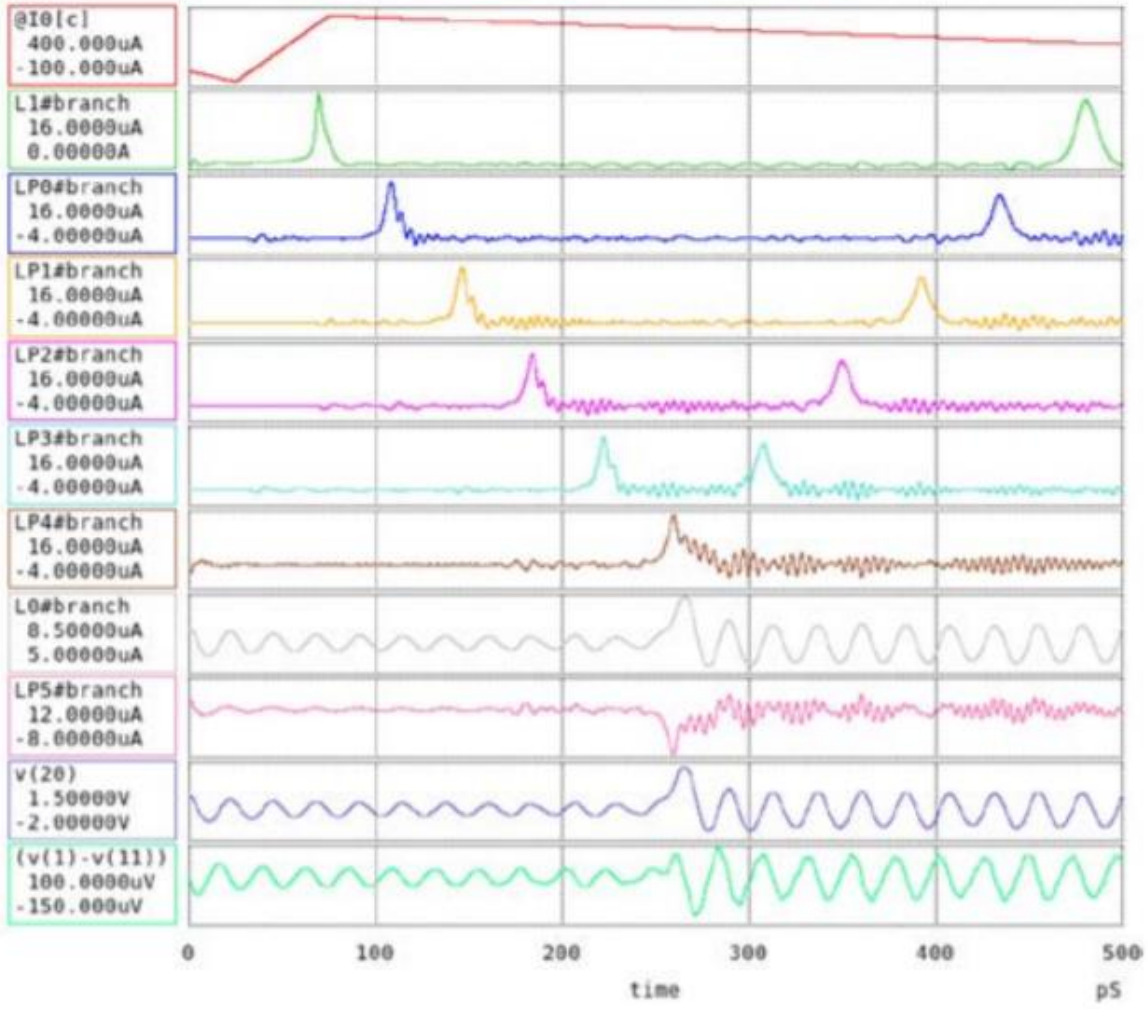
Netlist Example - Clickable

```
.lib C:\Users\FaDiMaT\Downloads\XIC\main.lib master
.model jjk2 jj(rtype=0, cct=1, vg=2.8m, icrit=1.5u, cap=60f)

.tran 0.02ps 0.6ns uic

x0 aa zz master

L0  aa    a    0H
L1  aa    zz   300pH     ic=-7uA
L2  b    zz   0H        ic=-7uA
B0  a    b    66   jjk2  ics=7.5uA


.save tran L0#branch v(66) L1#branch L2#branch



.control
run
plot L0#branch v(66) L1#branch L2#branch
write outtest2 L0#branch v(66) L1#branch L2#branch
.endc
```

```
*=====================================================================*
      * TEAM 382                                                      *
      * Lib file                                                      *
      * Contents:                                                     *
      *          - DCSFQ                                              *
      *            - Purpose:                                         *
      *              - Convert DC current to Single Fluxon Quantum    *
      *                Pluse                                          *
      *            - Schematic provided by SUNY                       *
      *              - physics.sunysb.edu/Physics/RSFQ/Lib/AR/dcsfq.html *
      *          - LJJ                                                *
      *            - Purpose:                                         *
      *              - Transmittion line, forces delay                *
      *            - Schematic provided by Sandia Labs                *
      *          - 20LJJ                                              *
      *            - Purpose:                                         *
      *              - 20 LJJS                                        *
      *              - spaced so proding can be done                  *
      *              - makes it easier to do a complete 100           *
      *          - Piecewise Current source                           *
      *              - Needed for the DCSFQ input                     *
*=====================================================================*


.lib master
.subckt master top_out bottom_out

.model jjk jj(rtype=0, cct=1, vg=2.8m, icrit=1.5u, cap=60f)
I0 0    top1 pwl(0 0 25p -0.07m 75p 0.35m 900p 0 r)
X0 top1 top2 20uAdcsfq
L0 top2 top3 150pH ic=13.7855589uA
X1 top3 0       top4 bottom1 dljj20
X2 top4 bottom1 top5 bottom2 dljj20
X3 top5 bottom2 top6 bottom3 dljj20
X4 top6 bottom3 top7 bottom4 dljj20
X5 top7 bottom4 top_out bottom_out dljj20

.subckt dljj-seg LT LB RT RB
.model jjk jj(rtype=0, cct=1, vg=2.8m, icrit=1.5u, cap=60f)
B0 5  6  7 jjk ics=1.5uA
L0 LT 5    7.845pH
L1 5  RT   7.845pH
L2 LB 6    7.845pH
L3 6  RB   7.845pH
.ends dljj-seg

.subckt dljj20 LT LB RT RB
X0  LT LB 5  25 dljj-seg
X1  5  25 6  26 dljj-seg
X2  6  26 7  27 dljj-seg
X3  7  27 8  28 dljj-seg
X4  8  28 9  29 dljj-seg
X5  9  29 10 30 dljj-seg
X6  10 30 11 31 dljj-seg
X7  11 31 12 32 dljj-seg
X8  12 32 13 33 dljj-seg
X9  13 33 14 34 dljj-seg
X10 14 34 15 35 dljj-seg
X11 15 35 16 36 dljj-seg
X12 16 36 17 37 dljj-seg
X13 17 37 18 38 dljj-seg
X14 18 38 19 39 dljj-seg
X15 19 39 20 40 dljj-seg
X16 20 40 21 41 dljj-seg
X17 21 41 22 42 dljj-seg
X18 22 42 23 43 dljj-seg
X19 23 43 RT RB dljj-seg
```

WRspice Example





Visual Example