Automation of the analysis of Large Hadron Collider superconducting circuit families using PSPICE and XYCE in the STEAM framework at CERN

A THESIS

submitted in partial fulfillment of the requirements

for the award of the dual degree of

Bachelor of Science - Master of Science

in

Electrical Engineering and Computer Science

by

Vijay Chakravarty

(18309)

DEPARTMENT OF EECS
INDIAN INSTITUTE OF SCIENCE EDUCATION AND RESEARCH BHOPAL
BHOPAL – 462 066

April 2023

CERTIFICATE

This is to certify that Vijay Chakravarty, BS-MS (Dual Degree) student in Department of Electrical Engineering and Computer Science, has completed bona fide work on the thesis entitled **'Automation of the analysis of Large Hadron Collider superconducting circuit families using PSPICE and XYCE in the STEAM framework at CERN'** under my supervision and guidance.

April 2023
IISER Bhopal

<Signature>
Dr. Mitradip Bhattacharjee

Committee Member	Signature	Date

ACADEMIC INTEGRITY AND COPYRIGHT DISCLAIMER

I hereby declare that this project report is my own work and due acknowledgement has been made wherever the work described is based on the findings of other investigators. This report has not been accepted for the award of any other degree or diploma at IISER Bhopal or any other educational institution. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission.

I certify that all copyrighted material incorporated into this document is in compliance with the Indian Copyright (Amendment) Act (2012) and that I have received written permission from the copyright owners for my use of their work, which is beyond the scope of the law. I agree to indemnify and safeguard IISER Bhopal from any claims that may arise from any copyright violation.

April 2023 <Signature>
IISER Bhopal Vijay Chakravarty

ACKNOWLEDGEMENT

I would like to thank the following people, without whom I would not have been able to complete this research, and without whom I would not have made it through the first semester of my master's degree!

The STEAM team at CERN, especially to my supervisor Dr Emmanuele Ravaioli, whose insight and knowledge into the subject matter steered me through this research. And special thanks to my supervisor at IISERB, Dr Mitradip Bhattacharjee, whose support allowed my studies to go the extra mile.

Finally, I would like to thank my friends and family for supporting me during the compilation of this dissertation.

Vijay Chakravarty

ABSTRACT

The STEAM project at CERN aims to achieve specialized, trusted, consistent, repeatable and sustainable software tools and models for rapid Simulation of Transient Effects in Accelerator superconducting Magnet circuits. This thesis intends to develop an analysis function to simulate all circuit families of the LHC superconducting circuit library using the STEAM framework. We develop a four-step strategy for the same. Firstly, existing circuit models are validated. In the next step, models are created for the remaining circuits of every circuit family. Thirdly, the circuit specific models are verified. Finally, the models are improved.

This paper summarizes the simulation and analysis method used for LHC library circuits. PSPICE and XYCE software are used to run the simulations. The analysis function developed will help automate the simulations of vastly different circuit families with a single click and in a short time. Such a function helps perform periodic checks on the circuit simulations. It also helps to quickly identify and debug errors when a particular simulation of a transient in a chosen circuit fails to pass sanity checks. Clearly, this study would be useful to the Machine Protection and Electrical Integrity Group for performance assessment.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1. INTRODUCTION

1.1 STEAM project at CERN

Due to the simultaneous occurrence of non-linear, interdependent events that are frequently characterized by time scales and dimensions that differ by orders of magnitude, modeling transient effects in superconducting circuits is extremely difficult. Superconducting filaments, strands, cables, magnet structures, and other components of the magnet electrical circuit, such as power sources, protection mechanisms, and current leads, experience electromagnetic, thermal, and mechanical transients. The STEAM (Simulation of Transient Effects in Accelerator Magnets) [1] framework included a number of programs designed to model electrical, electromagnetic, and thermal transients in superconducting magnets and circuits. Software in the software package may simulate quenching in superconducting circuits and magnets that are either self-protected or protected by energy extraction, quench heaters, CLIQ, or a combination of these.

For canted cos theta (ProteCCT) and multipole and solenoid (LEDET, both 2D and 3D) magnets, several of the programs employ finite difference (FD) solvers. These programs contain physics-based techniques for eddy currents, coupling losses, and filament magnetization in metallic magnet components. Other software uses FE solvers and has functionality to create multipole, canted cos theta, or high-temperature pancake coils (FiQuS in 2D or 3D) or multipole magnet simulation models automatically (SIGMA in 2D). Other programs use FD (LEDET, PyBBQ) or FE (BBQ) solvers to simulate quench in busbars. To conveniently control, version, and maintain the model inputs, the STEAM models employ a standard, text-based (YAML) input file type.

Increasingly, every program makes use of the same basic functions that are written in C and are compatible with tool-specific programming languages. Last but not least, many applications for modeling particular issues can be connected to a cooperative simulation (COSIM). Most LHC and HL-LHC superconducting circuit models were methodically created, tested, and versioned. These examples of transients, which encompass a wide range of magnet types, conductor specifications, and quench protection systems, are shown using STEAM modeling tools. The community is provided with the STEAM framework as a versatile, useful, and computationally efficient software package.

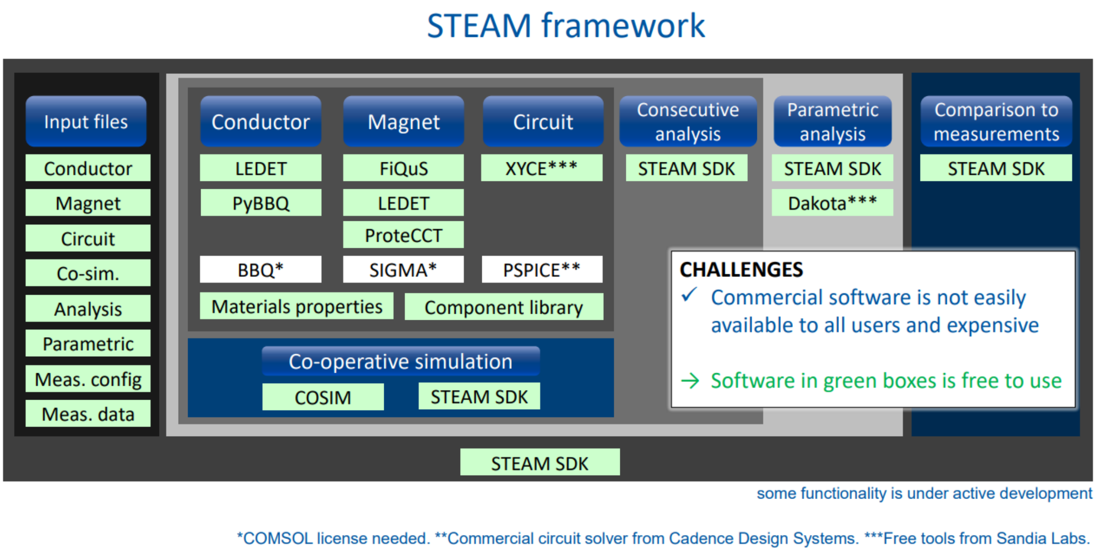


Figure 1: STEAM structure

1.2 PSPICE and XYCE

1.2.1 PSPICE

Cadence® PSpice® A/D [2] is a fully functional analog circuit simulator that supports digital components. It has a user-friendly graphical user interface and seamlessly connects with Cadence PCB schematic input solutions, giving users access to the whole design process to help with nearly any design challenge, from high-frequency systems to low-power IC designs. Due to its widespread popularity, several IC suppliers offer PSpice models with integrated mathematical functions and behavioural modelling methodologies.

Figure 2: PSPICE AD logo

The software has the following highlights:

• Enhances convergence, dependability, and simulation times for large designs

• Through the use of combined analog and event-driven digital simulations, speed is increased without sacrificing accuracy.

• Investigates the behaviour of circuits utilizing fundamental DC, AC, noise, and transient analysis

• Makes it possible to use the PSpice® Systems Option to integrate system-level interfaces with electrical designs.

• Enables the automatic application of A-to-D and D-to-A interfaces as well as the automatic identification of analog and digital signals

• Enables designers to investigate design linkages with "what if" scenarios before making a hardware decision.

• Allows for the identification and simulation of functional circuitry building blocks utilizing mathematical expressions, functions, and behavioural devices.

1.2.2 XYCE

An open source, SPICE-compatible, high-performance analog circuit simulator called Xyce [3] (pronounced "zīs", rhymes with "spice") is able to solve extremely large circuit issues by supporting large-scale parallel computing systems. Additionally, it enables small-scale parallel runs on Unix-like systems as well as serial execution on all popular desktop platforms. Xyce has been used to research more general network systems, such as neural networks and power grids, in addition to analog electronic simulation.

Figure 3: XYCE logo

It was developed by Sandia National Laboratory, one of the three National Nuclear Security R&D laboratories in the US.

The software has the following highlights:

- Modern Code base: Although Xyce aims to be SPICE-compatible, it is not a derivative of SPICE. Designed and written from scratch in C++, it has a modular, flexible design
- Support for large-scale parallel computing: It uses a message-passing implementation scheme to run on serial, shared and distributed memory parallel systems
- Differential-algebraic equation (DAE) formulation: This enhances encapsulation between the model and solver layers of the source code and enables the device models to be almost independent of the analysis type. For each sort of analysis, new device functions are built in a SPICE-based code. This is not necessary with the DAE implementation in Xyce.
- Analysis options and model support: Steady state (DCOP), transient (TRAN), small-signal frequency domain (AC), and small-signal noise response (NOISE) along with exotic analysis methods like Harmonic Balance, Multi-Time PDE and model-order reduction methods

1.3 LHC circuit families

1.3.1 IPQ:

The 78 Individually-Powered Quadrupole (IPQ) [4] superconducting circuits that make up the LHC matching sections and dispersion suppressor quadrupole operate in either 4.5 K liquid helium or 1.9 K superfluid helium, at a nominal current of 3610, 4310 or 5390 A. These circuits may be found in two different configurations with one quadrupole magnet, which has two apertures that are powered separately (see Fig. 4), and with two quadrupole magnets connected in series, which has four apertures that are powered separately (see Fig. 5). The circuit's primary parts are two power converters, 20 diodes connected in parallel to each power converter (PC), two Direct Current Current Transformers (DCCTs), which are used to accurately measure the current, three warm current leads, one earth resistance, and two or four magnet apertures. These magnets are protected against the effects of a quench by quench heaters. The MQM, MQML, MQMC and MQY types of superconducting Nb-Ti quadrupole magnets are used to construct these circuits. When these kinds of magnets are employed, the quantity of magnets and their specifications vary depending on the insertion. The following table provides an overview of the IPQ circuits in the LHC [5]

Circuit	Location	PC	Magnets	Temperature [K]	Nominal current [A]
RQ4	IR 2,8	RPHH	2xMQY	4.5	3610
RQ5	IR 2,8	RPHH	2xMQY	4.5	3610
RQ6	IR 2,8	RPHGB	MQM+MQML	4.5	4310
RQ7	IR 1,2,5,8	RPHGA	2xMQM	1.9	5390
RQ9	IR 1,2,4,5,6,8	RPHGA	MQM+MQMC	1.9	5390
RQ4	IR 1,5,6	RPHH	MQY	4.5	3610
RQ5	IR 4,6	RPHH	MQY	4.5	3610
RQ6	IR 4	RPHGB	MQY	4.5	3610
RQ5	IR 1,5	RPHGB	MQML	4.5	4310
RQ6	IR 1,5	RPHGB	MQML	4.5	4310
RQ7	IR 4	RPHGA	MQM	1.9	5390
RQ8	IR 1,2,4,5,6,8	RPHGA	MQML	1.9	5390
RQ10	IR 1,2,4,5,6,8	RPHGA	MQML	1.9	5390

Table 1: IPQ circuits summary

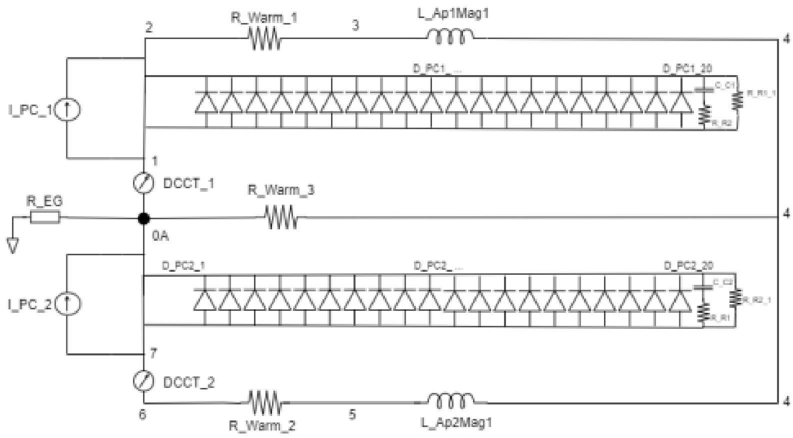


Figure 4: Schematic of the IPQ superconducting circuit, in one magnet configuration (two apertures).

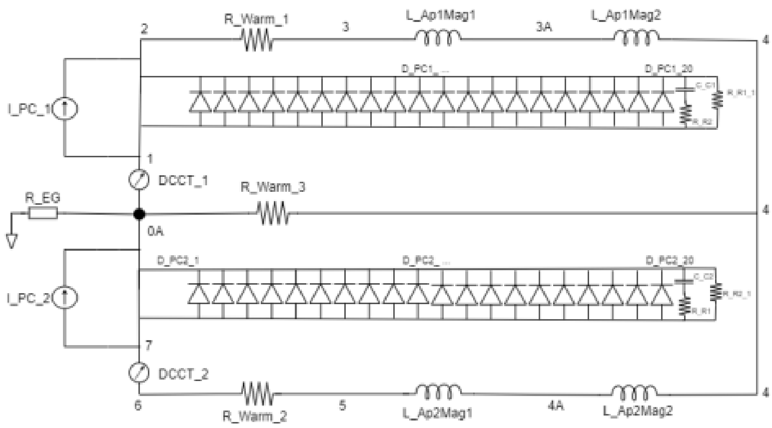


Figure 5: Schematic of the IPQ superconducting circuit, in two magnets configuration (four apertures).

1.3.2 IPD:

The LHC utilizes IPD superconducting circuits as separation magnets, which are located in the interaction regions of the LHC. These magnets are responsible for bringing the LHC's two beams into collision and then separating them again beyond the collision point. The IPD circuit family includes four circuit types: RD1-4, each of which contains a single magnet - MBX (RD1), MBRC (RD2), MBRS (RD3), and MBRB (RD4) [6]. Although these magnets are very similar, they differ in their nominal current, aperture number, separation distance, operating temperature, and bore diameter. Brookhaven National Laboratory (BNL) constructed and tested the magnets based on previous designs used in the RHIC. Since each circuit contains only one magnet, the transients that occur in the circuit and magnet can be simulated using STEAM-LEDET [7,8] as a standalone tool.

Nominal field strength	3.8 T
Peak Field in coil	4.2 T
Magnetic length	0.945 m
Aperture separation	188 mm (MBRC), 194 mm (MBRB); 414 mm (MBRS)
Outer cold bore diameter	78 mm (MBX); 69 mm (MBRS/B/C)
Nominal current	5750 A (MBX, MBRS); 6050 A (MBRB/C)
Operation temperature	1.9 K (MBX); 4.5 K (MBRS/C/B)
Superconductor	Nb-Ti
Inductance at nominal current	26 mH per aperture

Table 2: IPD circuit magnet parameters

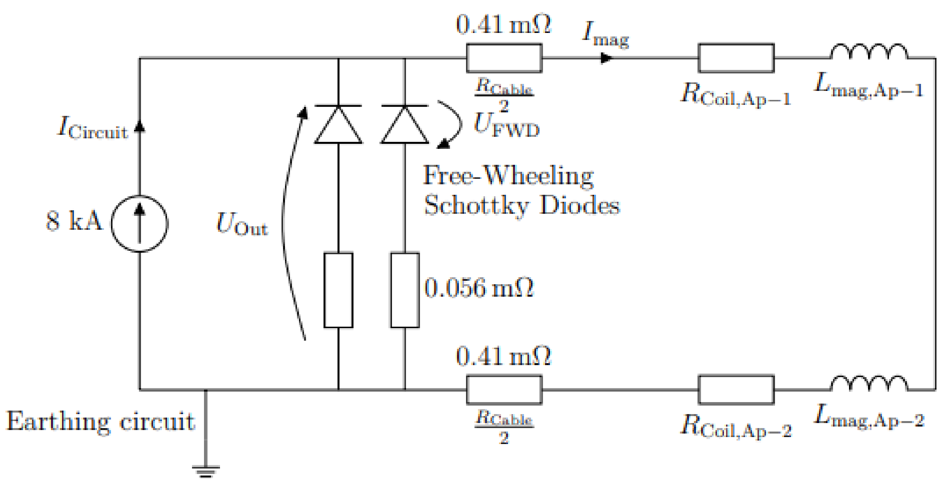


Figure 6: Electrical diagram of the LHC IPD circuits with two apertures

1.3.3 RQX:

The proton beams in the LHC machine are focused before collision at four locations, namely IRs 1 and 5 for high luminosity interaction regions (ATLAS and CMS) and IRs 2 and 8 for low luminosity interaction regions (ALICE and LHCb), using the inner triplet system. This system comprises three quadrupole optical elements and four magnets (Q1, Q2a, Q2b, and Q3). The Q1 and Q3 cold masses are designed, manufactured, and tested for acceptance by the KEK laboratory in Japan, while the Q2a and Q2b cold masses are designed, manufactured, and tested by FERMILAB. In addition, the inner triplet systems contain other corrector magnets and include one of the superconducting separation dipoles (D1) in the low luminosity interaction regions. The inner triplet quadrupole magnets MQXA and MQXB, having different design and parameters, are 'mixed' in the four insertions. Two nested high-current power converters: [8kA, 8V] and [6kA, 8V], implement a modular concept using [2kA,8V] current sources in parallel.[9] Across the Q1 magnet, a [±600A, ±10V] [10] trim power converter corrects differences in the integrated gradients of the Q1 and Q3 magnets.

	MQXA (Q1, Q3)	MQXB (Q2a, Q2b)
Nominal gradient	205 T/m	205 T/m
Nominal current	6450 A	11390 A
Ultimate current	6960 A	12290 A
Inductance	90.7 mH (=L1/2)	18.5 mH (=L2/2)
Stored energy at nominal current	1890 kJ	1200 kJ

Table 3: RQX magnet parameter

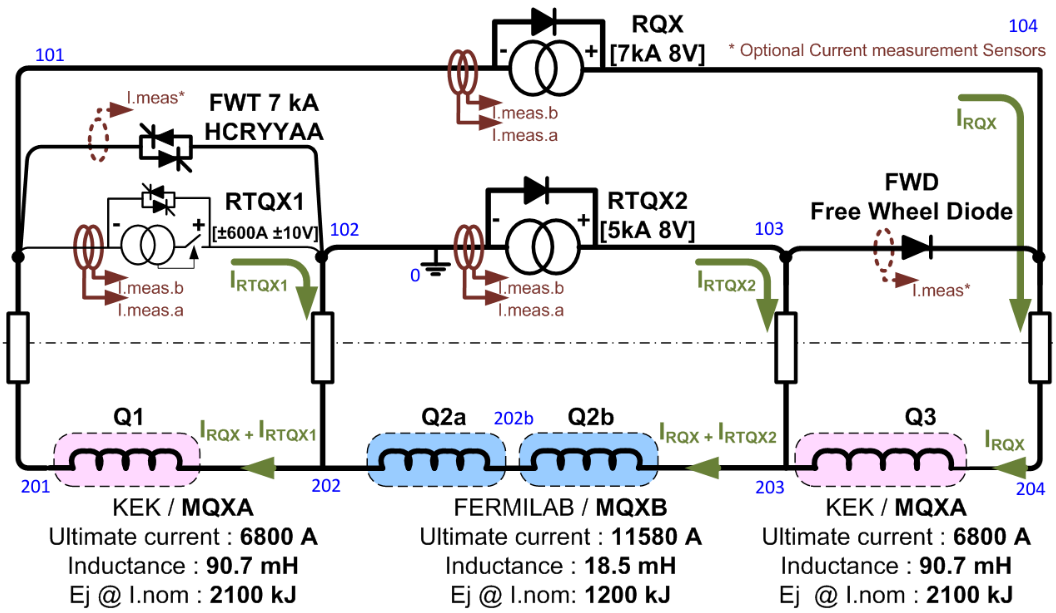


Figure 7: RQX circuit schematic

1.3.4 RQ:

The LHC consists of 16 main quadrupole circuits. Each quadrupole circuit is independently powered and consists of a varying number of quadrupole magnets, typically between 47 and 51 depending on the circuit. However, from an electrical perspective, the main quadrupole magnets in each section are divided into two separate circuits, designated by the abbreviations RQF and RQD. The final letter of each abbreviation indicates the intended function of the respective circuit: "F" denotes a circuit that focuses the particle beams in the horizontal plane, while "D" denotes a circuit that de-focuses the beams in the horizontal plane but focuses them in the vertical plane.[11]

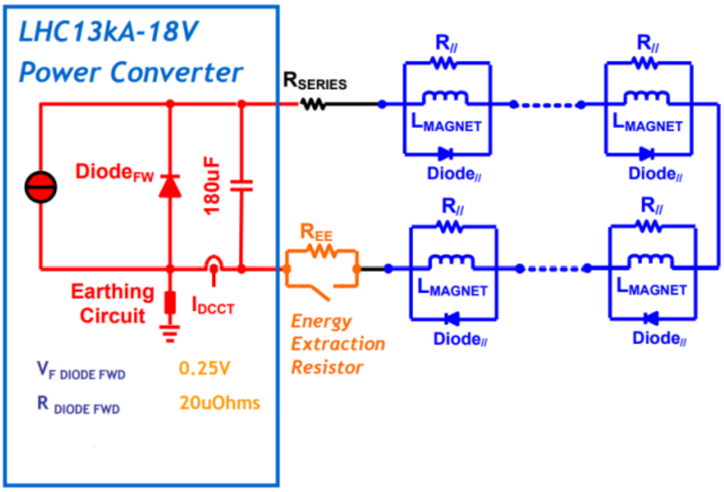


Figure 8: RQ circuit schematic [12

Each magnet has a resistor and bypass diode in parallel. For reliability reasons, the power converter contains several modules and filters, which are needed for providing the nominal current to the magnets [13]. The energy-extraction system is not a direct part of the protection system for the main quadrupole magnets, but reduces the energy stored in the circuit after detecting a quench [14].

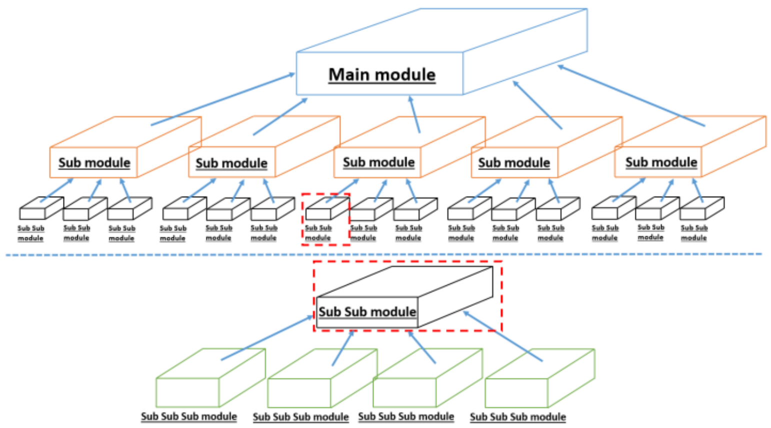


Figure 9: Modular structure of the LHC main quadrupole power converter

1.3.5 600A circuits:

There are a total of 394 600 A circuits powering 5339 superconducting magnets. Each of the circuits is protected actively, and the power supply is shut off once a specific quench voltage is detected by the quench detection units. In circuits without quench detection, the power supply is shut off when it reaches its voltage limit. The power supply crowbar, which runs parallel to the power supply, discharges the circuit when the power supply is deactivated. Additionally, half of the 600 A circuits have an energy extraction system that discharges the circuit quickly upon quench detection. The crowbar and the energy extraction system both add resistance to the circuit, but they have different purposes. The crowbar ensures that the maximum value over the power supply is limited and provides resistance to the circuit after power supply deactivation, while the energy extraction system is specifically designed to provide a minimum resistance upon quench detection for a faster discharge. However, in the case of a short in the power supply and the absence of a special 'DC contacts switch,' the crowbar does not add resistance to the circuit.For the purpose of this analysis, 600A circuits are dealt in two main categories as shown below:

1	Main		Sub-class	#magnets	Magnet type
26	600 A	Decapole spool pieces	RCD	77	MCD
27	600 A	Sextupole spool pieces	RCS	154	MCS
28	600 A	Defocusing octupole correctors	ROD	13	MO
29	600 A	Defocusing octupole correctors	ROD	8	MO
30	600 A	Focusing octupole correctors	ROF	13	MO
31	600 A	Focusing octupole correctors	ROF	8	MO
32	600 A	Matching section quadrupole	RQ6	6	MQTLH
33	600 A	Skew quadrupole corrector	RQS.AxxBx	4	MQS
34	600 A	(De)Focusing tune shift quadrupole	RQTD and RQTF	8	MQT
35	600 A	Dispersion suppressor quadrupole	RQTL9	2	MQTLI
36	600 A	Defocusing chromaticity sectupole	RSD	12	MS
37	600 A	Defocusing chromaticity sectupole	RSD	11	MS
38	600 A	Focusing chromaticity sectupole	RSF	10	MS
39	600 A	Focusing chromaticity sectupole	RSF	9	MS
40	600 A	Skew sextupole corrector	RSS	4	MSS

Table 4: Circuits with Energy Extraction

1	Main		Sub-class	#magnets	Magnet type
42	600 A	Skew quadrupole corrector	RQS.R/LxBx	2	MQS
43	600 A	Tuning quadrupole	RQT12	1	MQT
44	600 A	Tuning quadrupole	RQT13	1	MQT
45	600 A	Tuning quadrupole	RQTL7	1	MQTLI
46	600 A	Tuning quadrupole	RQTL8	1	MQTLI
47	600 A	Tuning quadrupole	RQTL10	1	MQTLI
48	600 A	Tuning quadrupole	RQTL11	1	MQTLI
49	600 A	Triplet corrector	RCBXH1/2	1	MCBXH
50	600 A	Triplet corrector	RCBXV1/2	1	MCBXV
51	600 A	Triplet corrector	RCBXH3	1	MCBXH
52	600 A	Triplet corrector	RCBXV3	1	MCBXV
53	600 A	Triplet corrector	RQSX3	1	MQSX

Table 5: Circuits without Energy Extraction

1.3.6 80-120 A circuits and 60 A circuits

80-120 A circuits consist of RCO with 77 magnets and RCBC, RCBY and triplet corrector circuits with 1 magnet each. RCB circuits with 1 magnet come under the 60 A circuits. All of these magnets are self-protected and their circuits do not possess any energy extraction system.

1	Main		Sub-class	#magnets	Magnet type	I_nom	I_ultim
54	80-120 A	Octupole spool pieces (Landau)	RCO	77	MCO	100	110
55	80-120 A		RCBCH	1	MCBCH	100	110
56	80-120 A		RCBCV	1	MCBCV	100	110
57	80-120 A		RCBCH	1	MCBCH	80	86
58	80-120 A		RCBCV	1	MCBCV	80	86
59	80-120 A	Triplet corrector	RCOSX3	1	MCOSX	100	110
60	80-120 A	Triplet corrector	RCOX3	1	MCOX	100	110
61	80-120 A	Triplet corrector	RCSSX3	1	MCSSX	100	110
62	80-120 A	Triplet corrector	RCSX3	1	MCSX	100	110
63	80-120 A	Triplet corrector	RCTX3	1	MCTX	100	110
64	80-120 A		RCBYH	1	MCBYH	72	77
65	80-120 A		RCBYV	1	MCBYV	72	77
66	60 A		RCBH	1	MCBH	55	60
67	60 A		RCBV	1	MCBV	55	60

Table 6: 80-120 A circuits and 60 A circuits

2. STRUCTURE

2.1 STEAM subgroups and modules

The following STEAM subgroups are used extensively for the simulations:

- steam-sdk: Source code for STEAM SDK and various APIs

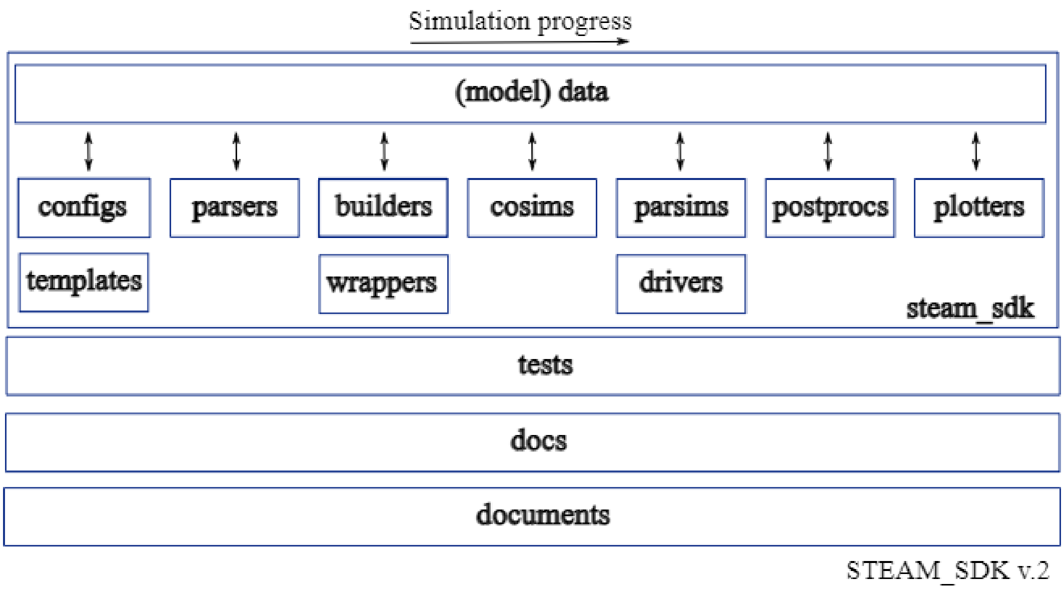


Figure 10: Anatomy of STEAM_SDK

From steam_sdk, different modules are used in the background to run the analysis successfully. Within one such module analyses, AnalysisSTEAM is the backbone of all analysis functions. The different steps highlighted in Figures 9 and 10 are run from AnalysisSTEAM. BuilderModel from builders module, DataAnalysis from data module, ParserPSPICE and ParserXYCE from parsers module and Viewer from viewers module form the rest of the skeleton of the analysis.

- steam_models: Repository of STEAM superconducting magnets and circuit models. Each circuit model is defined with a .yaml file, which is similar to a text file that preserves tree-structure (Publicly available without authentication)

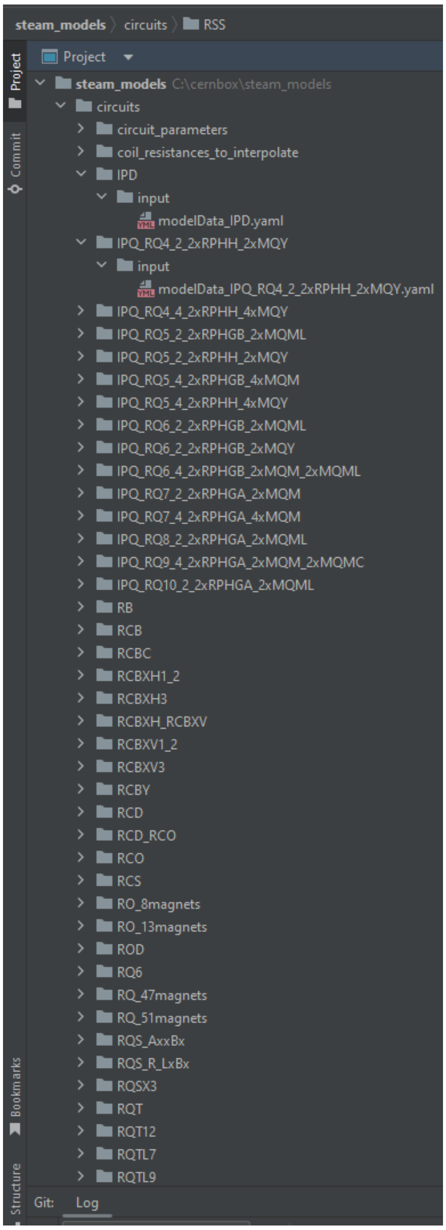


Figure 11: Different circuit models within steam_models

- steam-models-dev: Collection of STEAM superconducting magnet and circuit models and analyses, including those still in development mode. This repository relies on steam_models as a submodule

The following three images highlight a typical example of an analysis file, which is generated as a summary after running the simulation through the analysis function.

Circuit_Analysis_PSPICE.py × README.md × modelData_RQD_51magnets.yaml × interpolation_resistance_RQD_51magnets.csv ×

```yaml
GeneralParameters:
  analysis_name: all_circuits
  flag_permanent_settings: true
  model:
    name: all
    version: '1.0'
    case: null
    state: draft
PermanentSettings:
  comsolexe_path: C:\Program Files\COMSOL\COMSOL53a\Multiphysics\bin\win64\comsol.exe
  JAVA_jdk_path: C:\Program Files\Java\jdk1.8.0_281
  CFunLibPath: C:\Users\<user>\PycharmProjects\MaterialsLibrary
  COSIM_path: \\eosproject-smb\eos\project\s\steam\download\cosim\steam-cosim_v0.5.exe
  Dakota_path: null
  FiQuS_path: null
  LEDET_path: \\eosproject-smb\eos\project\s\steam\sw\snapshots\ledet\LEDET.exe
  ProteCCT_path: \\eosproject-smb\eos\project\s\steam\sw\snapshots\protecct\ProteCCT.exe
  PSPICE_path: C:\Cadence\SPB_17.4\tools\bin\psp_cmd.exe
  PyBBQ_path: null
  XYCE_path: null
  PSPICE_library_path: \\eosproject-smb\eos\project\s\steam\download\steam-pspice-library
  local_Dakota_path: null
  local_FiQuS_folder: null
  local_LEDET_folder: C:\tempLEDET\LEDET
  local_PyBBQ_folder: null
  local_PSPICE_folder: C:\tempPSPICE
  local_XYCE_folder: C:\tempXYCE
WorkingFolders:
  library_path: ../../steam_models
  output_path: output/PSPICE
  temp_path: temp
```

Figure 12: In the above snap, General Parameters, Permanent Settings and the Working Folders are specified.

```yaml
  local_PyBBQ_folder: null
  local_PSPICE_folder: C:\tempPSPICE
  local_XYCE_folder: C:\tempXYCE
WorkingFolders:
  library_path: ../../../steam_models
  output_path: output/PSPICE
  temp_path: temp
AnalysisStepDefinition:
  makeModel_IPD_1:
    type: MakeModel
    model_name: BM
    file_model_data: IPD
    case_model: circuit
    software: [PSPICE]
    simulation_name: null
    simulation_number: null
    flag_build: true
    flag_dump_all: false
    verbose: true
    flag_plot_all: false
    flag_json: false
  modifyModel1_IPD_1:
    type: ModifyModel
    model_name: BM
    variable_to_change: PostProcess.probe.probe_type
    variable_value: [CSDF]
    new_model_name: []
    simulation_numbers: []
    simulation_name: IPD
    software: [PSPICE]
  loadCircuitParameters_IPD_1:
    type: LoadCircuitParameters
    model_name: BM
    path_file_circuit_parameters: ../../steam_models/circuits/circuit_parameters/IPD_circuit_parameters.csv
    selected_circuit_name: RD2.L2
  modifyModel2_IPD_1:
    type: ModifyModel
    model_name: BM
    variable_to_change: PostProcess.probe.variables
    variable_value: [['V(2,1)', 'V(3,1)', 'V(2,3)', 'V(1,0)', I(R_ground), I(x_mag_1.L_mag), I(x_crowbar.R_crowbar), V(x_mag_1.0_coil_resistance_1)]]
    new_model_name: []
    simulation_numbers: [0]
    simulation_name: IPD
    software: [PSPICE]
  createStimulus_IPD_1:
    type: WriteStimulusFile
```

Figure 13: This snap shows few steps of the simulation: first the model is made, circuit parameters are loaded, desired modifications that have been made programatically to the simulation postprocessing in the analysis are shown.

```yaml
  createStimulus_IPD_1:
    type: WriteStimulusFile
    output_file: C:\cernbox\steam-models-dev\analyses\example_circuit_analysis\output\PSPICE\coil_resistances.stl
    path_interpolation_file: [C:\cernbox\steam_models\circuits\coil_resistances_to_interpolate\interpolation_resistance_MBRC.csv]
    n_total_magnets: 1
    n_apertures: 1
    current_level: [5800]
    magnets: [1]
    t_offset: [640.5]
    interpolation_type: Linear
    type_file_writing: w
    n_sampling: 1
    magnet_types: [1]
  runSimulation_IPD_1:
    type: RunSimulation
    software: PSPICE
    simulation_name: IPD
    simulation_numbers: [0]
    simFileType: null
AnalysisStepSequence:
- makeModel_IPD_1
- loadCircuitParameters_IPD_1
- modifyModel1_IPD_1
- modifyModel2_IPD_1
- createStimulus_IPD_1
- runSimulation_IPD_1
```

Figure 14: The stimulus file is generated and the simulation is run.

2.2 Pre-requisite Files

2.2.1 modelData YAML file:

YAML (short for "YAML Ain't Markup Language") is a human-readable data serialization language. It is often used for configuration files, but can also be used for data exchange between programming languages. YAML uses indentation to indicate nested structures, and has a simple syntax for defining key-value pairs, lists, and maps. Our modelData yaml files in steam_models have the following main sections:

- General Parameters: contains details about circuit name, model and additional files used
- Stimuli: contains the stimulus file “coil_resistances.stl”
- Libraries: contains component libraries needed for the simulation
- Netlist: provides a list of components used in the circuit, their nodes, values and parameters
- Analysis: provides analysis type and time schedule options
- PostProcess: contains option to change signal variables and probe type

```
GeneralParameters:
  circuit_name: IPD
  model:
    name: null
    version: null
    case: null
    state: null
  additional_files:
  - C:\gitlab\steam-pspice-library\misc\Items\misc_PowerSupplies.lib
  - C:\gitlab\steam-pspice-library\power_supply\Items\generic_power_converters.lib
  - C:\gitlab\steam-pspice-library\magnet\Items\magnets_generic.lib
AuxiliaryFiles:
  files_to_include: []
Stimuli:
  stimulus_files:
  - coil_resistances.stl
Libraries:
  component_libraries:
  - C:\gitlab\steam-pspice-library\misc\Items\misc_PowerSupplies.lib
  - C:\gitlab\steam-pspice-library\power_supply\Items\generic_power_converters.lib
  - C:\gitlab\steam-pspice-library\magnet\Items\magnets_generic.lib
GlobalParameters:
  global_parameters:
    L_magnet: '0.026'
    C_ground_magnet: '5.00E-08'
    R_parallel: '1.00E+03'
    R_circuit: '0.00045'
    R_crowbar: '5.60E-05'
    R_ground: '1E+09'
    I_start: '0'
    I_end_1: '1000'
    I_end_2: '5800'
    dI_dt: '10'
    dI_dt2: '1'
    t_start: '10'
    t_plateau: '20'
    t_PC_off: '640.5'
    t_PC_switch_off: '10E-03'
    L_parasitic_series: '1E-09'
    R_parasitic_series: '1E-09'
    R_parasitic_parallel: '1E+06'
InitialConditions:
  initial_conditions: {}
```

```yaml
  initial_conditions: {}
Netlist:
  x_PC:
    type: parametrized component
    nodes: ['1', '2']
    value: PC_controlled_equations
    parameters:
      I_start: I_start
      I_end_1: I_end_1
      I_end_2: I_end_2
      dI_dt: dI_dt
      dI_dt2: dI_dt2
      t_start: t_start
      t_plateau: t_plateau
      t_PC_off: t_PC_off
      t_PC_switch_off: t_PC_switch_off
      L_parasitic_series: L_parasitic_series
      R_parasitic_series: R_parasitic_series
      R_parasitic_parallel: R_parasitic_parallel
  x_crowbar:
    type: parametrized component
    nodes: ['1','2']
    value: generic_crowbar
    parameters:
      R_crowbar: R_crowbar
  R1:
    type: standard component
    nodes: ['2', '3']
    value: R_circuit
    parameters: null
  R_ground:
    type: standard component
    nodes: ['1', '0']
    value: R_ground
    parameters: null
  x_mag_1:
    type: parametrized component
    nodes: [ '3', '1', '0', '0' ]
    value: magnet_EQ_1_with_resistance_M1
    parameters:
      L_magnet: L_magnet
      C_ground_magnet: C_ground_magnet
      R_parallel: R_parallel
Options:
  options_simulation:
    all options: default
  options_autoconverge:
    all options: default
  flag_inCOSIM: null
Analysis:
  analysis_type: transient
  simulation_time:
    time_start: '0.0'
    time_end: '2000.0'
    min_time_step: '0.1'
    time_schedule:
      '0.0': '0.5'
      '630.0': '0.001'
      '645.0': '0.5'
PostProcess:
  probe:
    probe_type: standard
    variables: []
```


Figure 15: Example of a modelData yaml file

2.2.2 Configuration YAML file:

For each circuit type, we create a separate configuration YAML file. These files contain a list of signals to be saved for Viewer function that creates plots. Additionally, one configuration file can have multiple events of the same circuit type, if different signal plots are of interest. For example, in the figure below, IPD_1, IPD_2, et cetera can be different events for the same circuit type IPD.

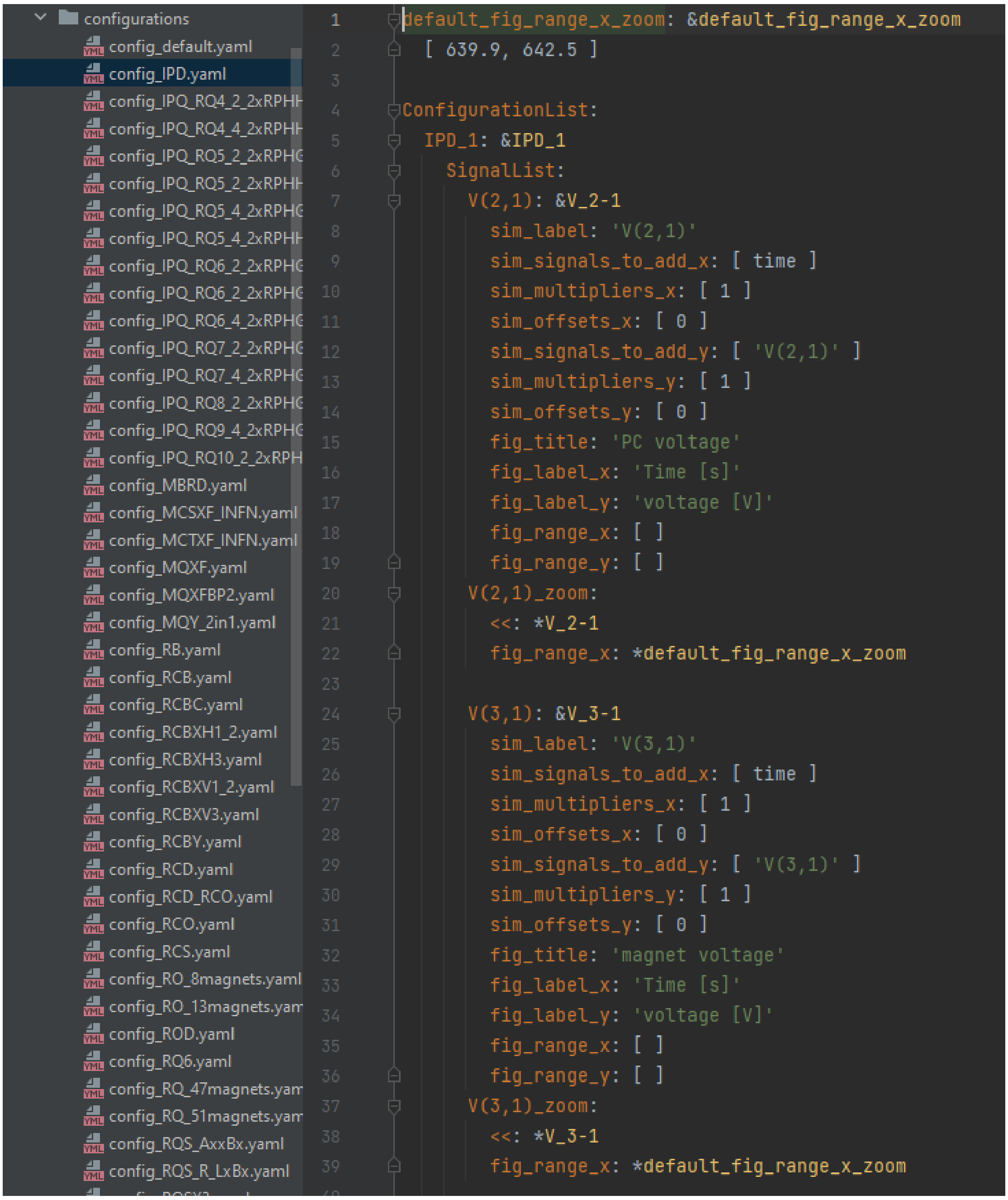


Figure 16: Example of a configuration file

2.2.3 Circuit YAML file:

This file contains a list of all simulations for testing. Each key represents a simulation with the option to change its input variables and parameters. In the figure below, one can see many such inputs. For example, magnet_name_list provides which magnets should be used in the superconducting circuit; variables_to_save corresponds to signals to plot (mentioned earlier in configuration files); selected_circuit lets us choose the specific circuit from a csv file of that circuit family (described in the next section). Moreover, keys_for_Viewer help to initialise the arguments for Viewer function. Setting up of keys like t_offset is indispensable for the generation of stimulus files. Finally, checks_against_targets helps us establish tests for each signal of every simulation at any specified time.

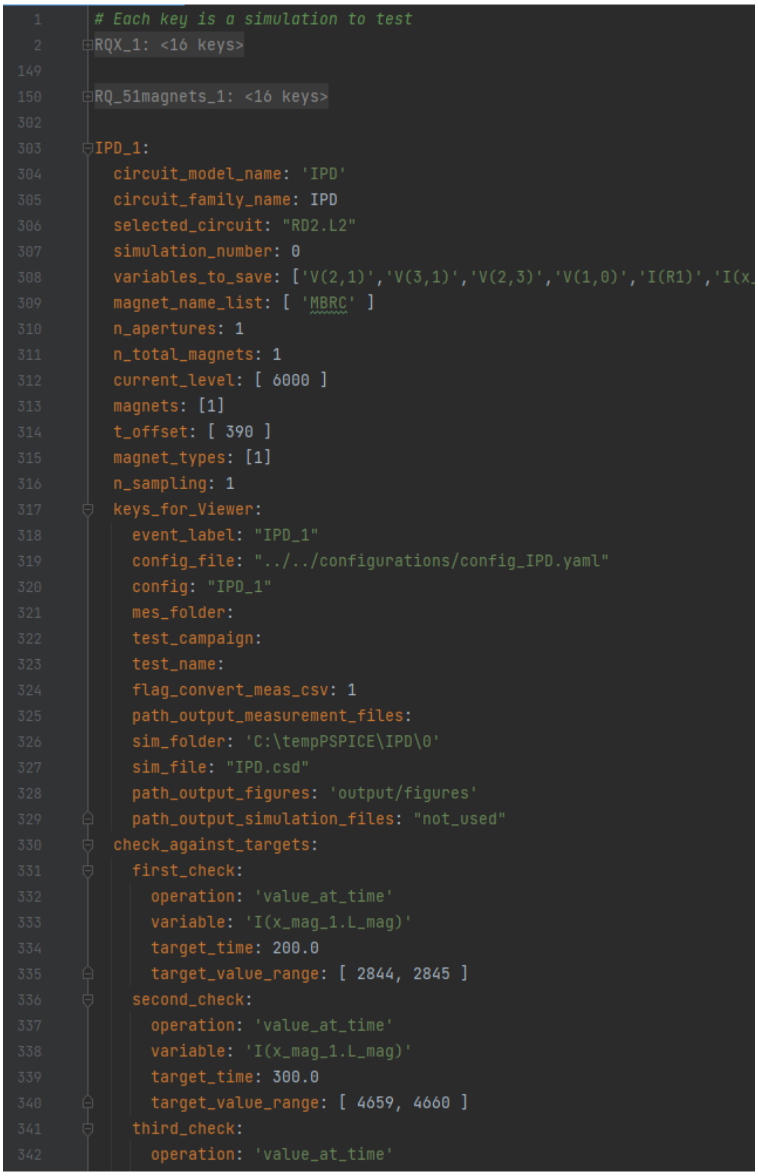


Figure 17: Circuit yaml file

2.2.4 Circuit parameters csv file:

For each of the circuit families mentioned in Section 1.3, we have a circuit_parameters csv file storing different circuits possible for various circuit types within that family. For example, in the figure below, IPD_circuit_paramters.csv file stores all the circuits possible from its four circuit types RD1-4. (Refer to section 1.3.2) For each circuit corresponding to each row of the file, there are all the Global Parameters needed to run the simulation for that specific circuit. These same Global Parameters need to be fed into the Global Parameters section of modelData yaml file for IPD. For that purpose, we use the loadCircuitParameters step in the analysis. (See Figures 9 and 10) Therefore, the parameters csv file stores the correct circuit parameters to be loaded for the simulation.

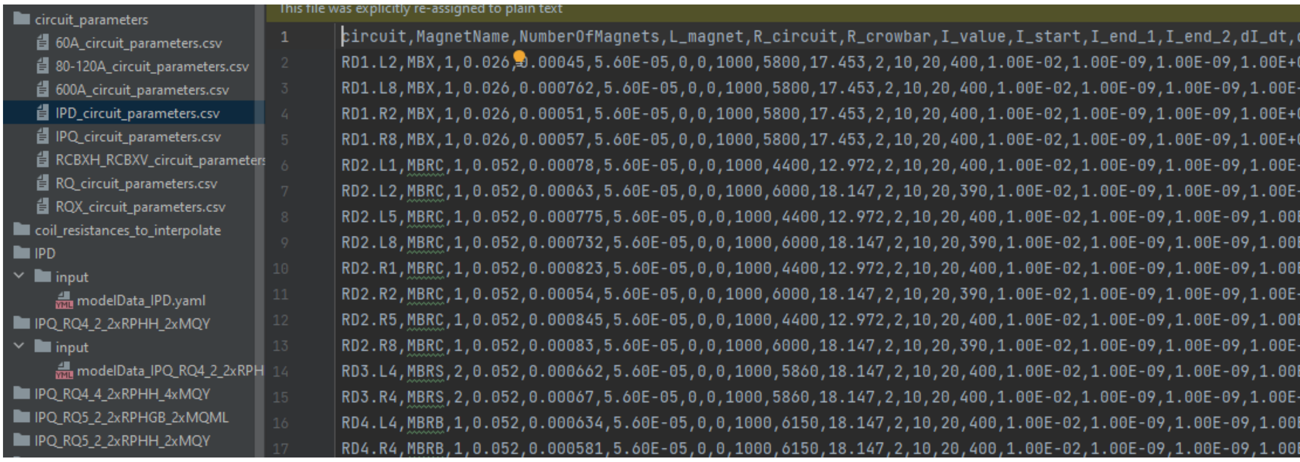


Figure 18: Example of a circuit parameters file

2.2.5 Interpolation resistance csv file:

Most of these files have been compiled using a MATLAB script to parse information from simulation results from earlier work of PSPICE-LEDET co-simulation by the STEAM team. (See Section 3.2) However, for the self-protected magnets of 600A, 80-120A and 60A families, the files were generated again after running 3D LEDET simulations in multiple locations.

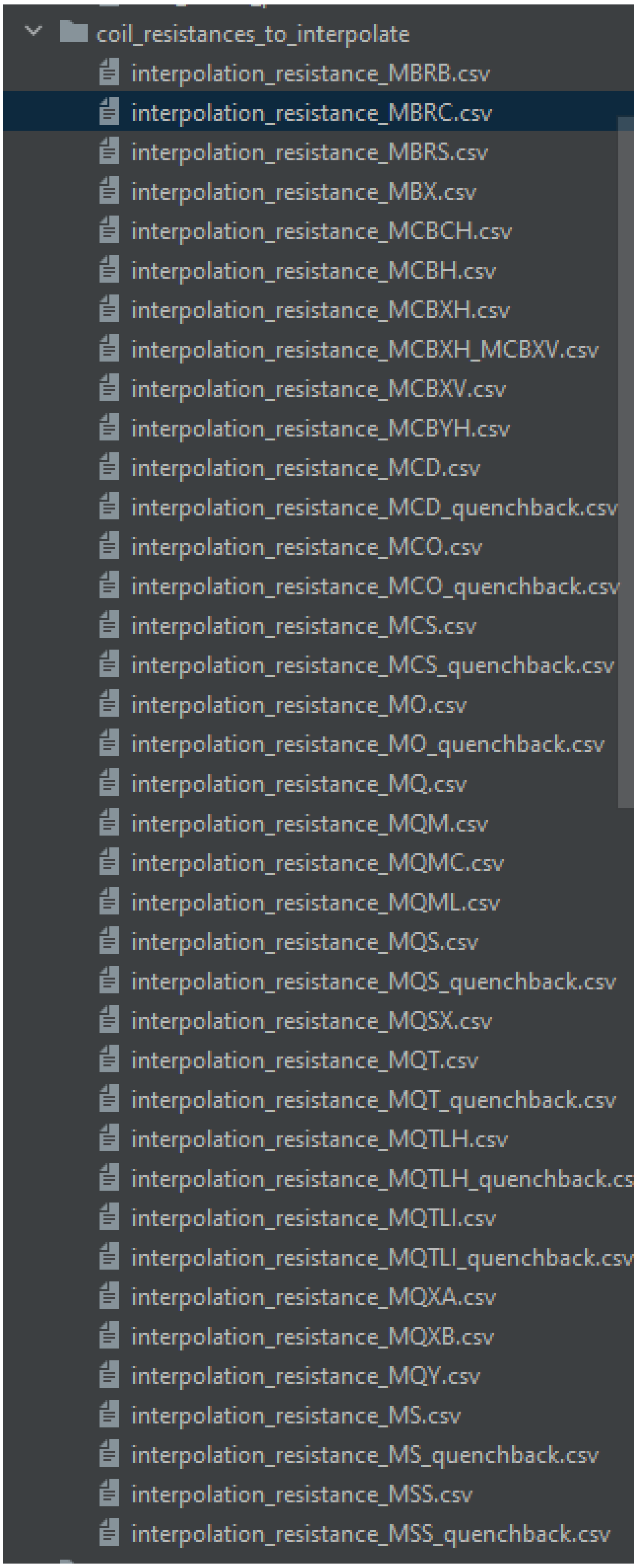


Figure 19: List of interpolation resistance files

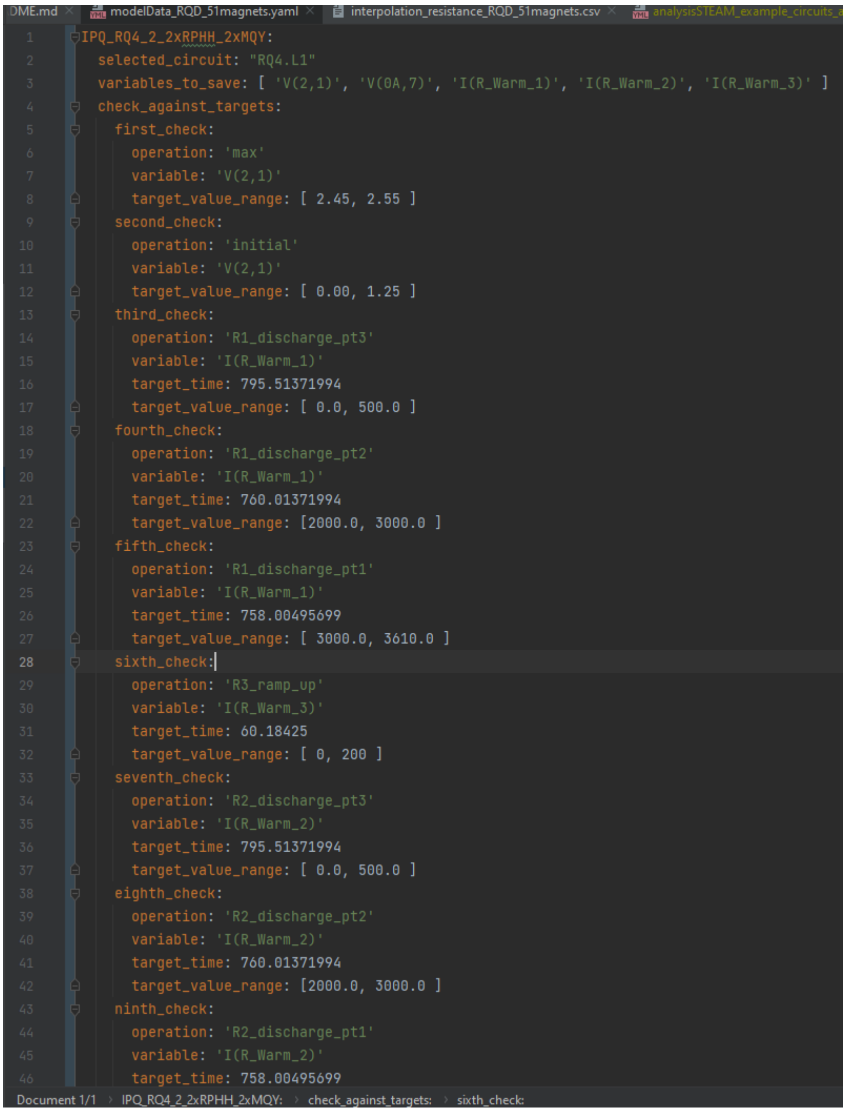
	A	B	C	D	E	F	G	H	I	J	K
1	6500			6000			5400			4800	
2	time_vect	R_CoilSections_1		time_vect	R_CoilSections_1		time_vect	R_CoilSections_1		time_vect	R_CoilSectio
3	-1.20E-01	0.00E+00		-1.20E-01	0.00E+00		-1.20E-01	0.00E+00		-1.20E-01	0.00E+00
4	-1.20E-01	0.00E+00		-1.20E-01	0.00E+00		-1.20E-01	0.00E+00		-1.20E-01	0.00E+00
5	-1.19E-01	0.00E+00		-1.19E-01	0.00E+00		-1.19E-01	0.00E+00		-1.19E-01	0.00E+00
6	-1.19E-01	0.00E+00		-1.19E-01	0.00E+00		-1.19E-01	0.00E+00		-1.19E-01	0.00E+00
7	-1.18E-01	0.00E+00		-1.18E-01	0.00E+00		-1.18E-01	0.00E+00		-1.18E-01	0.00E+00
8	-1.18E-01	0.00E+00		-1.18E-01	0.00E+00		-1.18E-01	0.00E+00		-1.18E-01	0.00E+00
9	-1.17E-01	0.00E+00		-1.17E-01	0.00E+00		-1.17E-01	0.00E+00		-1.17E-01	0.00E+00
10	-1.17E-01	0.00E+00		-1.17E-01	0.00E+00		-1.17E-01	0.00E+00		-1.17E-01	0.00E+00
11	-1.16E-01	0.00E+00		-1.16E-01	0.00E+00		-1.16E-01	0.00E+00		-1.16E-01	0.00E+00
12	-1.16E-01	0.00E+00		-1.16E-01	0.00E+00		-1.16E-01	0.00E+00		-1.16E-01	0.00E+00
13	-1.15E-01	0.00E+00		-1.15E-01	0.00E+00		-1.15E-01	0.00E+00		-1.15E-01	0.00E+00
14	-1.15E-01	0.00E+00		-1.15E-01	0.00E+00		-1.15E-01	0.00E+00		-1.15E-01	0.00E+00
15	-1.14E-01	0.00E+00		-1.14E-01	0.00E+00		-1.14E-01	0.00E+00		-1.14E-01	0.00E+00
16	-1.14E-01	0.00E+00		-1.14E-01	0.00E+00		-1.14E-01	0.00E+00		-1.14E-01	0.00E+00
17	-1.13E-01	0.00E+00		-1.13E-01	0.00E+00		-1.13E-01	0.00E+00		-1.13E-01	0.00E+00
18	-1.13E-01	0.00E+00		-1.13E-01	0.00E+00		-1.13E-01	0.00E+00		-1.13E-01	0.00E+00
19	-1.12E-01	0.00E+00		-1.12E-01	0.00E+00		-1.12E-01	0.00E+00		-1.12E-01	0.00E+00
20	-1.12E-01	0.00E+00		-1.12E-01	0.00E+00		-1.12E-01	0.00E+00		-1.12E-01	0.00E+00
21	-1.11E-01	0.00E+00		-1.11E-01	0.00E+00		-1.11E-01	0.00E+00		-1.11E-01	0.00E+00
22	-1.11E-01	0.00E+00		-1.11E-01	0.00E+00		-1.11E-01	0.00E+00		-1.11E-01	0.00E+00
23	-1.10E-01	0.00E+00		-1.10E-01	0.00E+00		-1.10E-01	0.00E+00		-1.10E-01	0.00E+00
24	-1.10E-01	0.00E+00		-1.10E-01	0.00E+00		-1.10E-01	0.00E+00		-1.10E-01	0.00E+00
25	-1.09E-01	0.00E+00		-1.09E-01	0.00E+00		-1.09E-01	0.00E+00		-1.09E-01	0.00E+00
26	-1.09E-01	0.00E+00		-1.09E-01	0.00E+00		-1.09E-01	0.00E+00		-1.09E-01	0.00E+00
27	-1.08E-01	0.00E+00		-1.08E-01	0.00E+00		-1.08E-01	0.00E+00		-1.08E-01	0.00E+00
28	-1.08E-01	0.00E+00		-1.08E-01	0.00E+00		-1.08E-01	0.00E+00		-1.08E-01	0.00E+00
29	-1.07E-01	0.00E+00		-1.07E-01	0.00E+00		-1.07E-01	0.00E+00		-1.07E-01	0.00E+00
30	-1.07E-01	0.00E+00		-1.07E-01	0.00E+00		-1.07E-01	0.00E+00		-1.07E-01	0.00E+00
31	-1.06E-01	0.00E+00		-1.06E-01	0.00E+00		-1.06E-01	0.00E+00		-1.06E-01	0.00E+00
32	-1.06E-01	0.00E+00		-1.06E-01	0.00E+00		-1.06E-01	0.00E+00		-1.06E-01	0.00E+00
33	-1.05E-01	0.00E+00		-1.05E-01	0.00E+00		-1.05E-01	0.00E+00		-1.05E-01	0.00E+00
34	-1.05E-01	0.00E+00		-1.05E-01	0.00E+00		-1.05E-01	0.00E+00		-1.05E-01	0.00E+00
35	-1.04E-01	0.00E+00		-1.04E-01	0.00E+00		-1.04E-01	0.00E+00		-1.04E-01	0.00E+00
36	-1.04E-01	0.00E+00		-1.04E-01	0.00E+00		-1.04E-01	0.00E+00		-1.04E-01	0.00E+00
37	-1.03E-01	0.00E+00		-1.03E-01	0.00E+00		-1.03E-01	0.00E+00		-1.03E-01	0.00E+00
38	-1.03E-01	0.00E+00		-1.03E-01	0.00E+00		-1.03E-01	0.00E+00		-1.03E-01	0.00E+00
39	-1.02E-01	0.00E+00		-1.02E-01	0.00E+00		-1.02E-01	0.00E+00		-1.02E-01	0.00E+00
40	-1.02E-01	0.00E+00		-1.02E-01	0.00E+00		-1.02E-01	0.00E+00		-1.02E-01	0.00E+00
41	-1.01E-01	0.00E+00		-1.01E-01	0.00E+00		-1.01E-01	0.00E+00		-1.01E-01	0.00E+00
42	-1.01E-01	0.00E+00		-1.01E-01	0.00E+00		-1.01E-01	0.00E+00		-1.01E-01	0.00E+00
43	-1.00E-01	0.00E+00		-1.00E-01	0.00E+00		-1.00E-01	0.00E+00		-1.00E-01	0.00E+00
44	-9.95E-02	0.00E+00		-9.95E-02	0.00E+00		-9.95E-02	0.00E+00		-9.95E-02	0.00E+00
45	-9.90E-02	0.00E+00		-9.90E-02	0.00E+00		-9.90E-02	0.00E+00		-9.90E-02	0.00E+00
46	-9.85E-02	0.00E+00		-9.85E-02	0.00E+00		-9.85E-02	0.00E+00		-9.85E-02	0.00E+00
47	-9.80E-02	0.00E+00		-9.80E-02	0.00E+00		-9.80E-02	0.00E+00		-9.80E-02	0.00E+00
48	-9.75E-02	0.00E+00		-9.75E-02	0.00E+00		-9.75E-02	0.00E+00		-9.75E-02	0.00E+00
49	-9.70E-02	0.00E+00		-9.70E-02	0.00E+00		-9.70E-02	0.00E+00		-9.70E-02	0.00E+00

interpolation_resistance_MBRC

Figure 20: Example of an interpolation csv file with current levels in the first row. For magnets with two apertures, there will be an extra R_CoilSections_2 column for each current level

2.3 Viewer Plots and Sanity Checks

Plots are automatically generated and saved for the desired signals of each circuit simulation. Zoomed-in plots around the time of PC switch off are also automatically generated for each plot. Additionally, sanity checks are exercised upon the signal values at points of interest. For instance, the following image shows the different checks performed on signals generated for the IPQ_RQ4_2_2xRPHH_2xMQY circuit. These include checking the maximum and initial values of voltage across the power converter, checking the currents in R1 and R2 resistors at three time points during the discharge and checking the current through R3 resistor during the ramp-up.



```
IPQ_RQ4_2_2xRPHH_2xMQY:
  selected_circuit: "RQ4.L1"
  variables_to_save: [ 'V(2,1)', 'V(0A,7)', 'I(R_Warm_1)', 'I(R_Warm_2)', 'I(R_Warm_3)' ]
  check_against_targets:
    first_check:
      operation: 'max'
      variable: 'V(2,1)'
      target_value_range: [ 2.45, 2.55 ]
    second_check:
      operation: 'initial'
      variable: 'V(2,1)'
      target_value_range: [ 0.00, 1.25 ]
    third_check:
      operation: 'R1_discharge_pt3'
      variable: 'I(R_Warm_1)'
      target_time: 795.51371994
      target_value_range: [ 0.0, 500.0 ]
    fourth_check:
      operation: 'R1_discharge_pt2'
      variable: 'I(R_Warm_1)'
      target_time: 760.01371994
      target_value_range: [2000.0, 3000.0 ]
    fifth_check:
      operation: 'R1_discharge_pt1'
      variable: 'I(R_Warm_1)'
      target_time: 758.00495699
      target_value_range: [ 3000.0, 3610.0 ]
    sixth_check:
      operation: 'R3_ramp_up'
      variable: 'I(R_Warm_3)'
      target_time: 60.18425
      target_value_range: [ 0, 200 ]
    seventh_check:
      operation: 'R2_discharge_pt3'
      variable: 'I(R_Warm_2)'
      target_time: 795.51371994
      target_value_range: [ 0.0, 500.0 ]
    eighth_check:
      operation: 'R2_discharge_pt2'
      variable: 'I(R_Warm_2)'
      target_time: 760.01371994
      target_value_range: [2000.0, 3000.0 ]
    ninth_check:
      operation: 'R2_discharge_pt1'
      variable: 'I(R_Warm_2)'
      target_time: 758.00495699
```

Figure 21: Sanity Checks

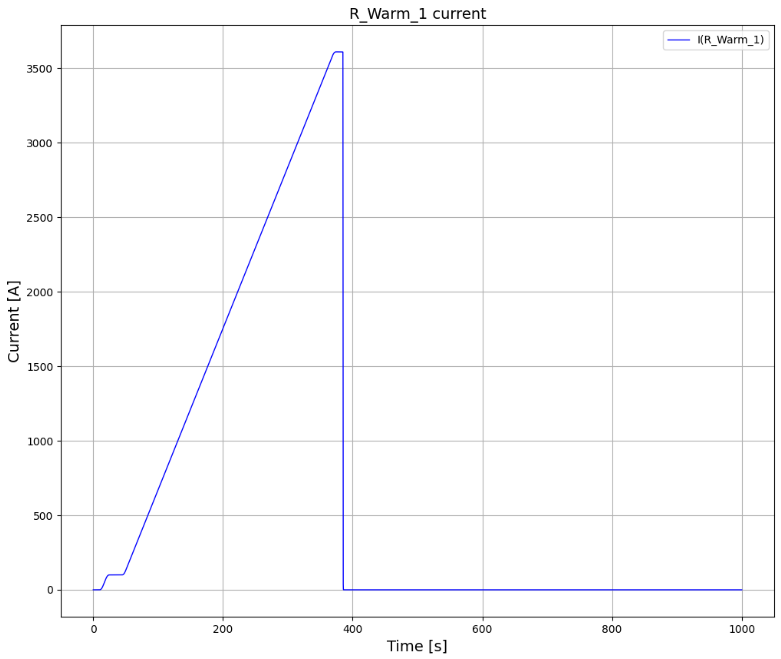


Figure 22: R_warm_1 current plot

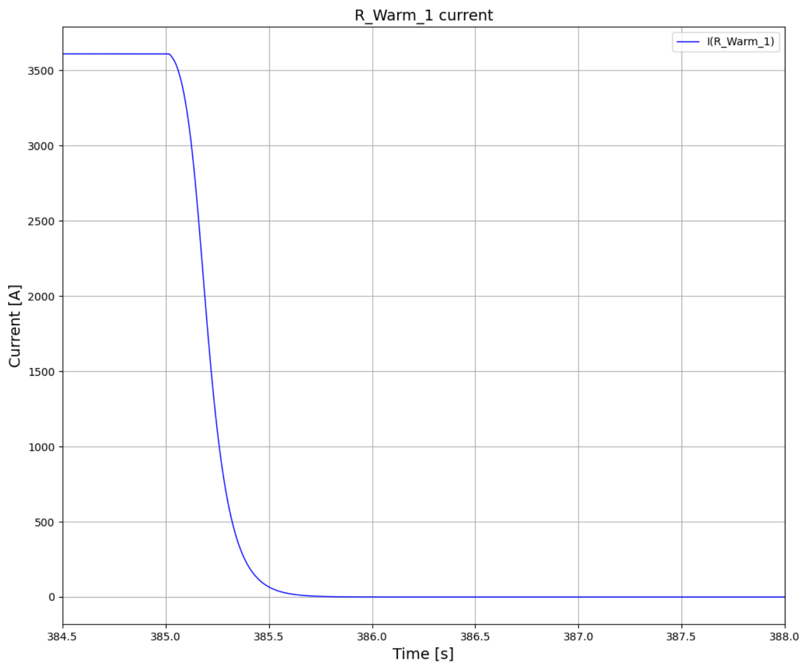


Figure 23: Zoomed in R_warm_1 plot around the time PC is switched off

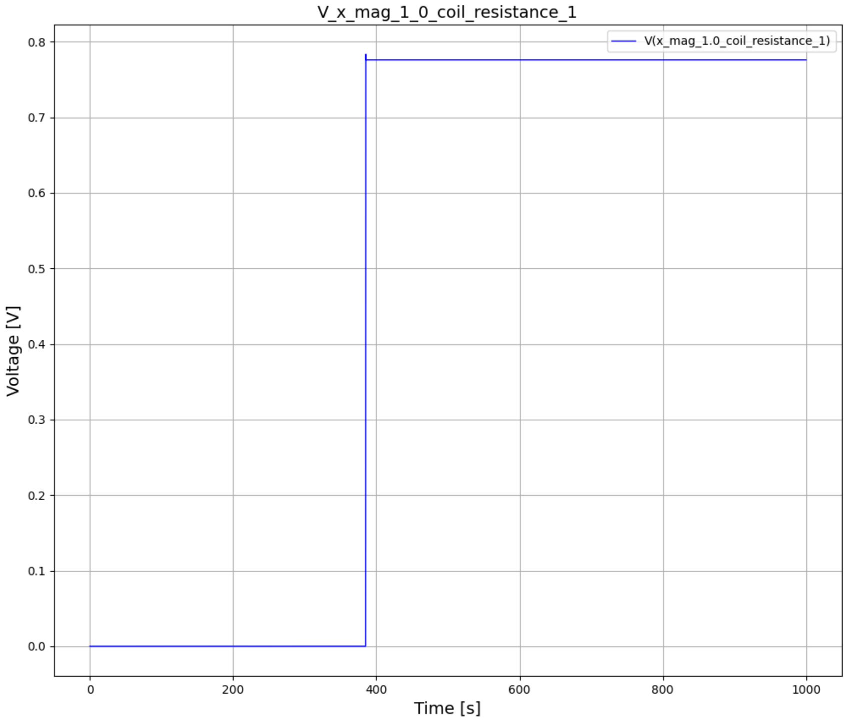


Figure 24: V(x_mag_1.0_coil_resistance_1) plot

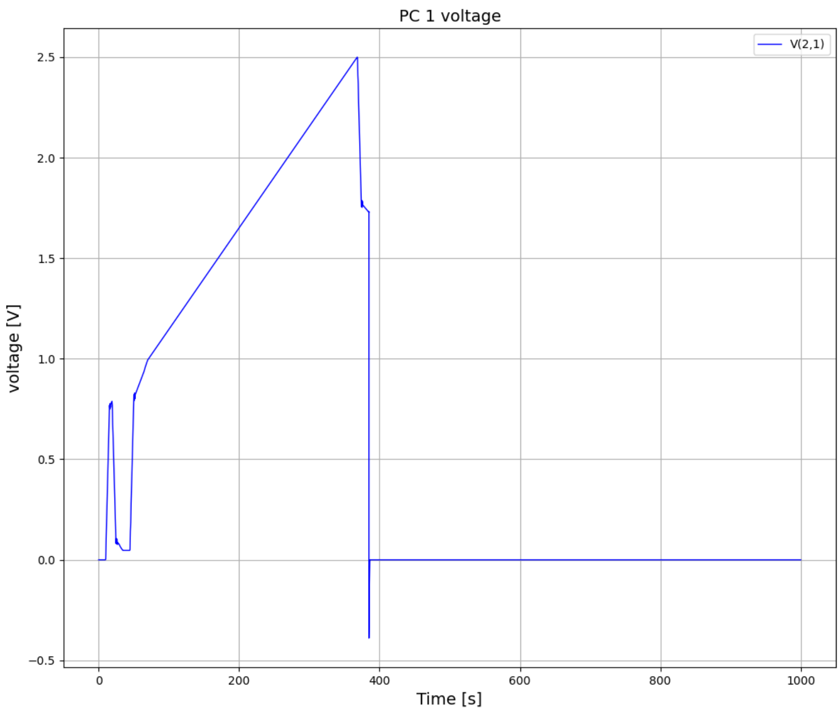


Figure 25: Voltage across PC1 plot

3. ANALYSIS STRATEGY

3.1 Steps A, B, and C

STEP A: Validation of existing circuits
Ensuring that the circuit models that are already present in the library run successfully as part of the new analysis. This step has been done for all circuit families and it has provided us a bird's eye view spanning through the breadth of different families that need to be incorporated under one analysis function.

STEP B: Making models of all remaining circuits
Every circuit family may have some circuit models that need to be created from scratch to include different parameters like magnet types or PC types. For example, 14 different circuit models had to be created for the IPQ circuit family in order to be able to simulate each of the 78 different circuit types.

STEP C: Verification of circuit specific models
In the next step, it has to be ensured that each of the circuits are able to modify their model parameters as per their own specifications. We use LoadCircuitParameters function to ensure that the correct parameters are input into the model before every run of the simulation.

3.2 Step D: Improving the Model

Once steps A, B, and C are done and pushed on gitlab, we embark on the final step to improve the model. This is done through three tasks:

- **Coil Resistance Interpolation:**
  Lumped-Element Dynamic Electro-Thermal (LEDET) [15] model includes non-linear dynamic effects such as the dependence of the magnet's differential self-inductance on the presence of inter-filament and inter-strand coupling currents in the conductor. Superconducting magnets are primarily operated in stationary conditions, so these effects are often neglected, however, they affect magnet performance, especially with high ramp rates. Using LEDET, the electro-magnetic and thermal domains can be modeled with three sub-networks of lumped-elements: reproducing the electrical transient in the main magnet circuit, the thermal transient in the coil cross-section, and the electro-magnetic transient of the inter-filament and inter-strand coupling currents in the superconductor. LEDET helps to simultaneously model macroscopic electrical transients and phenomena at the level of superconducting strands. The Coil resistances used for interpolation are derived from earlier work of PSPICE-LEDET co-simulation by the STEAM team. [16] Once the coil resistances are taken from the LEDET output, we can use them to interpolate coil resistances for current levels in the range. Other options can be fine tuned, like how many magnets and apertures are there, what time offset should be used, etc.

  $$R = \frac{(R2 - R1)}{(I2 - I1)} * (I - I1) + R1$$

  For interpolation, we use the good old formula shown above, where R is the interpolated resistance, I is the current level at which R is sought, I1 and I2 are lower and upper bounds of the range of current levels, R1 and R2 are the coil resistances at I1 and I2. This formula is used to find R at every time step.

- **Stimulus File Generation:**

  Once we have the interpolated coil resistances, we can feed that into our next step of stimulus (.stl) file generation. R_coil_1_M1 in the following image acts as an external stimulus to the parametrized magnet component in the magnets library file. This way of using interpolated coil resistances to determine exact stimulus for the parametrized magnet helps us with automation of simulations of very different magnet types. Thus, we replace the magnet component to include coil resistances.

```

.STIMULUS R_coil_1_M1 PWL
+ TIME_SCALE_FACTOR = 1
+ VALUE_SCALE_FACTOR = 1
+ ( 0s, 0.0 )
+ ( 757.70505s, 0.0 )
+ ( 757.71005s, 0.0 )
+ ( 757.71505s, 0.0 )
+ ( 757.72005s, 0.0 )
+ ( 757.72505s, 0.0 )
+ ( 757.73005s, 0.0 )
+ ( 757.73505s, 0.0 )
+ ( 757.74005s, 0.00024080550000000003 )
+ ( 757.74505s, 0.0014309271 )
+ ( 757.7500500000001s, 0.0016714044 )
+ ( 757.7550500000001s, 0.0017625471 )
+ ( 757.7600500000001s, 0.001863127274999998 )
+ ( 757.7650500000001s, 0.00197258625 )
+ ( 757.7700500000001s, 0.00219744255 )
+ ( 757.7750500000001s, 0.002380548675 )
+ ( 757.7800500000001s, 0.00253927455 )
+ ( 757.7850500000001s, 0.0027746855250000003 )
+ ( 757.7900500000001s, 0.0029600976000000003 )
```

Figure 26: Stimulus file example

- **Change of Power Converters:**

  Global parameters like ramp rate and acceleration of current, etc. are included in the IPQ circuits to control the power converter with an analytical equation as shown below:

```
* PSPICE Generic Power Converter components library
* Version 1.0: 2022/06/01, Emmanuele Ravaioli, STEAM, TE-MPE-PE, CERN, Geneva, CH
* This library includes models of power supplies, i.e. power converters (PC), that can be used for multiple circuits

*$********************************************************************************

* Subcircuit: PC_controlled_equations
* Power supply, i.e. power converter (PC), controlled with an analytical equation
* Notes:
* - The PC is modelede as two voltage-controlled current sources in parallel governed by analytical equations
* - One current source models the first phase (I_start-->I_end_1) and the other the second phase (I_end_1-->I_end_2)
* - The PC current is set to follow this pattern:
* --- stay at I_start until t>t_start
* --- change parabolically, then linearly, then parabolically from I_start to I_end_1
* --- wait at I_end_1 for a time t_plateau
* --- change parabolically, then linearly, then parabolically from I_end_1 to I_end_2
* --- if at any time the simulation time reaches t_PC_off, the PC is switched off and its current set to 0 A even if the ramp is not finished
* - The transition from parabolic to linear phases is set so that the dI_dt does not change abruptly
* - To simplify the lunderstanding of the logic, virtual voltage nodes are used to calculate relevant times
* - Parasitic components are included. The default settings improve numerical convergence in many practical cases.
```

Figure 27: Description of the analytical equation for the PC

4. CONCLUSIONS

4.1 Automation Flow

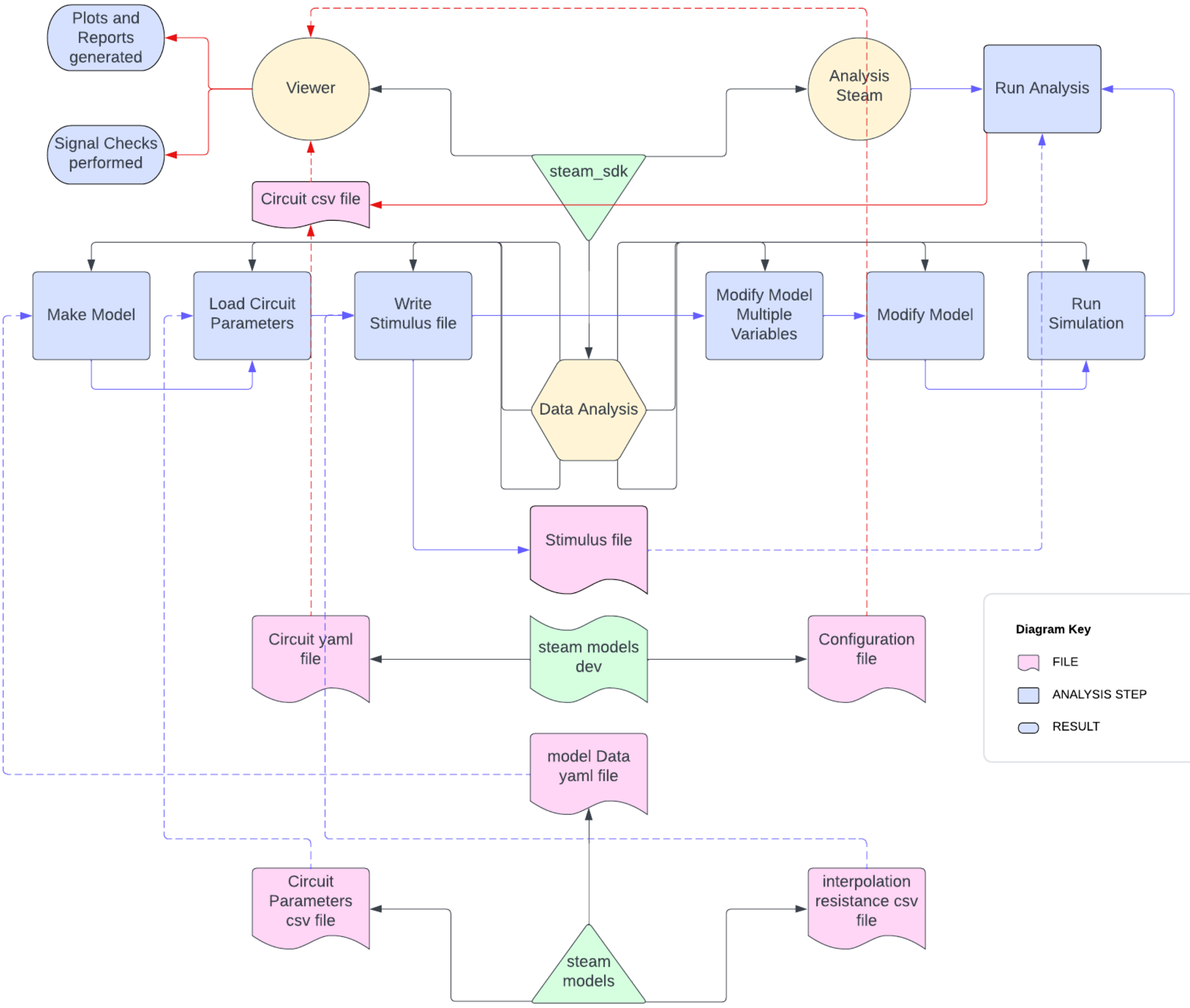


Figure 28: A pictorial representation of the Automation flow

In the above diagram, the solid black lines connect modules, submodules, and the files of the steam project. There are two types of coloured lines: blue and red, representing two steps. The blue lines show the first step, i.e., the flow during the analysis, and the red lines show the next step, i.e., the processing done after the analysis. The dotted blue and red lines represent the input of a file during these two steps, respectively. As discussed in Section 2.1, the three steam subgroups, namely, steam models, steam models dev, and steam_sdk, are represented in green. The five pre-requisite files discussed in Section 2.2 are contained within steam models dev and steam models. Steam_sdk has three main modules: Viewer, Analysis Steam, and Data Analysis, respectively. Data Analysis leads to six different submodules, which form the analysis steps. Analysis Steam provides for the final running of the analysis using software PSPICE or XYCE and feeds into the folder directories pointed to by the Circuit csv file. The circuit yaml file containing the keys of each simulation adds rows to the Circuit csv file during the analysis. This Circuit csv file is read by the Viewer to produce plots and reports, and finally, the automation flow ends with checks on specific signals of the simulation, as discussed in Section 2.3.

Therefore, the flow of the automation can be summarised as follows:

Step 1: Start from Make Model and follow the solid blue lines to go through the six steps of Data Analysis, which ends with Run Analysis.

Step 2: Follow the solid red lines from Run Analysis to go to the Viewer through the Circuit csv file and terminate with plots, reports, and signal checks.

4.2 The way ahead

This paper achieves two goals. Firstly, it develops an improved analysis strategy for all the circuits of different types within different circuit families of the LHC circuit library. Secondly, it provides for the automation of these analyses so that all of them could be run with a single click. An automated analysis and inbuilt subsequent signal check strategy for such a vast library of complicated circuits of the world's largest and most powerful particle accelerator will be a handy tool for the Machine Protection and Electrical Integrity Group at CERN for performance assessment. The way ahead would involve the usage of this automated analysis for improvements in superconductor circuit modeling as well as its integration with pre-existing frameworks at CERN. One such integration attempt has already commenced: the real-time Parsim events generated by the SigMon (Signal Monitoring) project within the Control and Beams Study for Magnet Protection section will be used as a starting point for this analysis so that simulations are representative of real events detected from the LHC. This work is expected to be completed by July 2023. This paper would be updated to include the same.

BIBLIOGRAPHY

[1] (2022). STEAM Software Framework for Simulation of Transient Effects in Accelerator Magnets and Circuits. Indico. https://indico.cern.ch/event/1161448/

[2] (2020). Cadence PSPICE. PSpice. https://www.pspice.com/

[3] (2022). XYCE. Sandia National Laboratories. https://xyce.sandia.gov/

[4] Murgia, F. (2020). Multiphysics modelling of the lhc individually powered quadrupole superconducting circuits [Master].

[5] Bruning, O. S. (2004). LHC Design Report. Geneva : CERN, 2004. - 548 p.

[6] STEAM Circuit Library Path: cern.ch/dfs/Projects/LHC/RD/Circuit Documentation

[7] S. Russenschuck, ROXIE: Routine for the Optimization of Magnet X-Sections, Inverse Field Calculation and Coil End Design, Ed. Geneva, Switzerland: CERN, April 1999.

[8] LHC ROXIE Magnets Repository
URL: http://roxie-lhc-magnets.web.cern.ch/roxie-LHC-magnets/MBRC/
http://roxie-lhc-magnets.web.cern.ch/roxie-LHC-magnets/MBRS/
http://roxie-lhc-magnets.web.cern.ch/roxie-LHC-magnets/MBX/
http://roxie-lhc-magnets.web.cern.ch/roxie-LHC-magnets/MBRB/

[9] F. Bordry, A. Dupaquier, “High Current, Low Voltage Power Converters for LHC", EPAC’96, Sitges, June 96.

[10] F. Bordry, A. Dupaquier, G. Kniegl, R. Weber, "FourQuadrant Converter [±600A, ±12V]. Prototype for LHC", PAC'99, New-York, Mars 99

[11] P. Schmüser K.H. Meß and S. Wolff. Superconducting Accelerator Magnets. Singapore: World Scientific, 1996.

[12] L. Charnay and V. Montabonnet. LHC13kA-18V LHC MQ Power Converter Guide. English. Version EDMS Doc. N973221 V.2. CERN. 2009, p. 22.

[13] O. S. Bruening et al. LHC Design Report, Volume I, The LHC Main Ring. CERN Yellow Reports: Monographs. Geneva: CERN, 2004. doi: 10.5170/CERN-2004-003- V-1. url: http://cds.cern.ch/record/782076.

[14] Dahlerup-Petersen, K and Rodriguez-Mateos, F and Schmidt, R and Sonnemann, F. “Energy Extraction for the LHC Superconducting Circuits”. In: LHC-Project-Report484. CERN-LHC-Project-Report-484 (Aug. 2001), p. 4. url: https://cds.cern. ch/record/514346.

[15] Ravaioli, E. (2022). LEDET Manual. STEAM.

[16] (2018). STEAM-COSIM. EDMS CERN. https://espace.cern.ch/steam/_layouts/15/start.aspx#/COSIM/Home.aspx