User Guide for MIRCians to translate pulse sequences to WGE, Bangalore

Author:

Sneha Potdar, M.Tech.

Research Assistant

MIRC, DSI

Email: Sneha-mirc@dayanandasagar.edu

Table of Contents

1. Introduction

This document is intended for use once a pulse sequence program has been coded in either MATLAB, python or pulseq-gpi (1).

A simple, modular framework called TOPPE (2) can enable rapid prototyping of pulse sequences on GE scanner. Implementation of research pulse sequences on GE scanner requires knowledge EPIC programming that is typically time consuming. However, TOPPE pulse sequences allow execution of the sequences on the scanner with a set of external files that are generated in Pulseq (MATLAB/Python/GPI). This makes it convenient to play different types of RF pulses and gradient waveforms without EPIC programming.

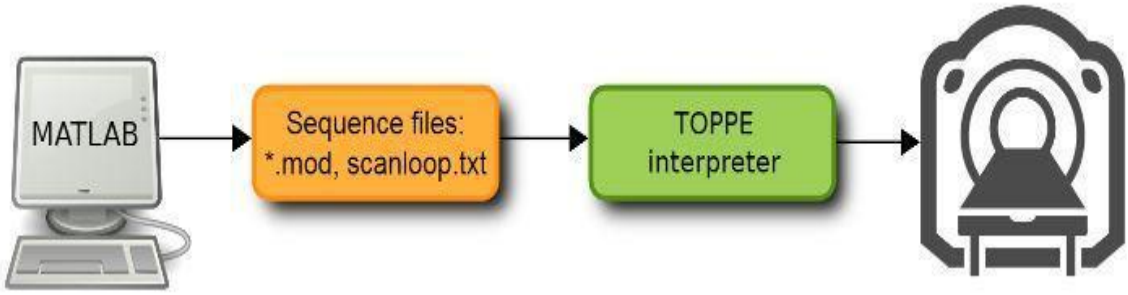


Figure 1: Overview of pulse sequence programming using TOPPE; Image reproduced from source: https://toppemri.github.io/[2]

Pulseq generated *.seq* file is given as an input to seq2ge function, where TOPPE sequence files specify all details of MR acquisition like timing, gradient and RF waveforms. These files are fed to TOPPE interpreter as binary executables. This can then be executed on a scanner.

1.1 TOPPE in 3 steps

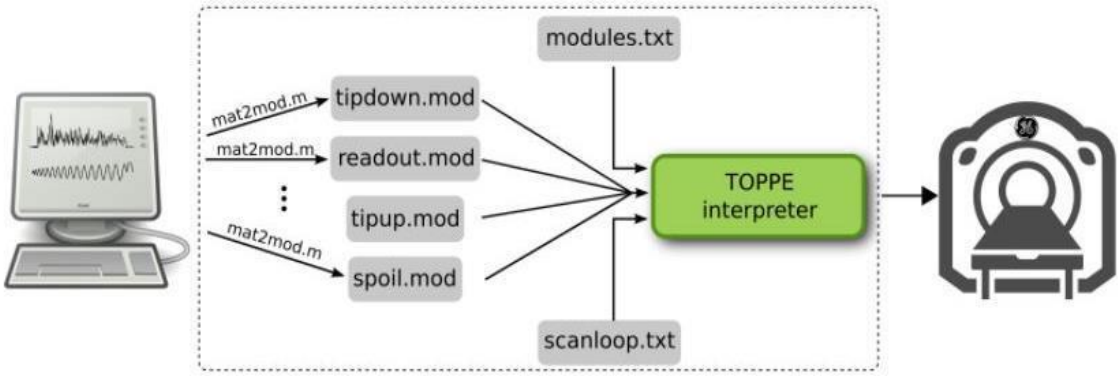


Figure 1: Steps involved in prototyping a sequence; Image reproduced from source: https://toppemri.github.io/[2]

1. Designing an RF pulse and gradient waveforms using any method using Pulseq (MATLAB/Python/GPI).
2. Create TOPPE sequence files tipdown.mod, readout.mod, modules.txt, scanloop.txt and depending on the sequences few core*.mod.
3. Copy the generated files to /usr/g/bin/ on the scanner and run the executable.

1.2 Essential Files

In addition to TOPPE and TOPPE.psd.o executable, the following files are needed to run a particular scan

1. *.mod
2. modules.txt
3. scanloop.txt
4. legacy files

1.2.1 Mod files

Several unique "cores/modules" are generated by TOPPEV1.e with each core/module associated with one or more mod files. For example:

RF excitation module is defined by tipdown.mod, which describes about

- RF amplitude
- Phase waveforms
- 3 gradients

Cartesian (spin-wrap) gradient echo is defined by readout.mod which describes about

- Readout
- Phase-encode gradient waveforms

Note that each .mod file give rise to *createseq( )* call in TOPPEv1.e

Mandatory points:

1. Make sure one of the readout (acquisition) .mod files is named readout.mod
2. Make sure one of the RF excitation .mod files is named tipdown.mod

1.2.2 Modules.txt

Module files (*.mod) needed to define a scan are listed in a small text file called modules.txt, which contains a line for each .mod file specifying the file name, core duration and whether it is an RF excitation module, readout module or gradients. For example: module.txt for 2D GRE looks like as shown below, generated at MIRC.

Make sure there are different RF mod files depending on your sequence. (For example for SE)

```
Total number of unique cores
10
wavfile_name    duration (us)    has_RF?    has_ADC?
tipdown.mod     0       1       0
core2.mod       0       0       0
core3.mod       0       0       0
readout.mod     0       0       1
core5.mod       0       0       0
core6.mod       0       0       0
core7.mod       0       0       0
core8.mod       0       0       0
core9.mod       0       0       0
core10.mod      0       0       0
```

1.2.3 Scanloop.txt

Scanloop.txt contains MR scan loop, where *nt* refers to number of times the *startseq( )* method called. It specifies the module to be played with RF and Gradient. Below is the corresponding data shown for 2D GRE sequence example (Generated at MIRC).

nt	maxslice	maxecho	maxview											
1280	0	0	256											
module	ia_rf	ia_th	ia_gx	ia_gy	ia_gz	dabslice	dabecho	dabview	dabon	phi	rfphase	recphase	extra_T	RF_offest
1	32766	32766	0	0	32766	0	0	0	0	0	0	0	0	0
2	32766	32766	-32766	-32766	-32766	0	0	0	0	0	0	0	0	0
3	32766	32766	0	0	0	0	0	0	0	0	0	0	0	0
4	32766	32766	32766	0	0	0	0	1	1	0	0	0	0	0
5	32766	32766	0	0	0	0	0	1	0	0	0	0	0	0
1	32766	32766	0	0	32766	0	0	1	0	0	0	0	0	0
2	32766	32766	-32766	-32510	-32766	0	0	1	0	0	0	0	0	0
3	32766	32766	0	0	0	0	0	1	0	0	0	0	0	0

1.2.4 Legacy files

The files legacy1.rho, legacy1.theta and legacy2.rho must be copied to /usr/g/bin n the scanner from subfolders of the examples directory.

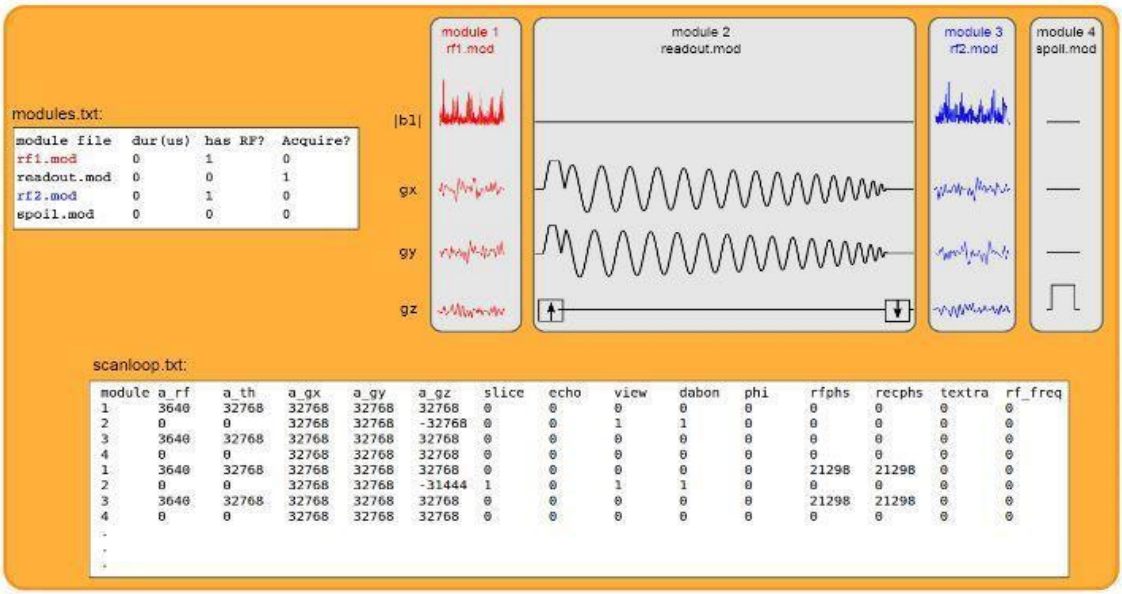


Figure 2: Overview of the structure of TOPPE file structure; Image reproduced from source: https://toppemri.github.io/[2]

2. Creating mod files – conditions to be met before sharing with JFWTC

MATLAB has a script named mat2mod which generates .mod files for RF, Gx, Gy and Gz waveforms. Following are the few points to be remembered during the process:

SL.NO	Mandatory Points	Status
1	Waveforms length should be same, else need to be zero padded	☐
2	Mod files of gradient waveforms should begin and end with 0	☐
3	If RF module is not present need to create a dummy module	☐
4	Readout.mod, data will be acquired for 4μs	☐

3. Steps to follow to run TOPPE sequence

SL.NO	Files	Destination
1	Copy TOPPEv1, TOPPEv1.psd.o and legacy files	/usr/g/bin/
2	Copy modules.txt, scanloop.txt and all mod files	/usr/g/bin/
3	Fill the prescribed details of the sequence – Scan prescription	
4	Save and run the autoprescan	
5	Click the scan button, pfile will be created	/usr/g/mrraw

4. Checklist of things to run/ files to send to JFWTC

☐ I have tipdown.mod, readout.mod, scanloop.txt, modules.txt

☐ I have multiple (core*.mod) module files

☐ I have different rf.mod files (if sequence has more than 1 RF)

☐ I have checked for ISNAN, ISINF and value greater than 32767 in Scanloop

☐ I have checked for 15 columns in Scanlooop

☐ I have checked(Traced) the kspace trajectory for the sequence

☐ I have legacy files – rho1, rho2 from the repository of the TOPPE folder

☐ I understand that TOPPEv1.e does not support cardiac/respiratory gating

☐ I understand that TOPPEv1.e does not perform monitoring on SAR, PNS and gradient heating

☐ I understand that I need to check if the data saved is in proper order

☐ I understand that B1 scaling across multiple RF pulses has not been taken into account

5. Worked carried out at MIRC

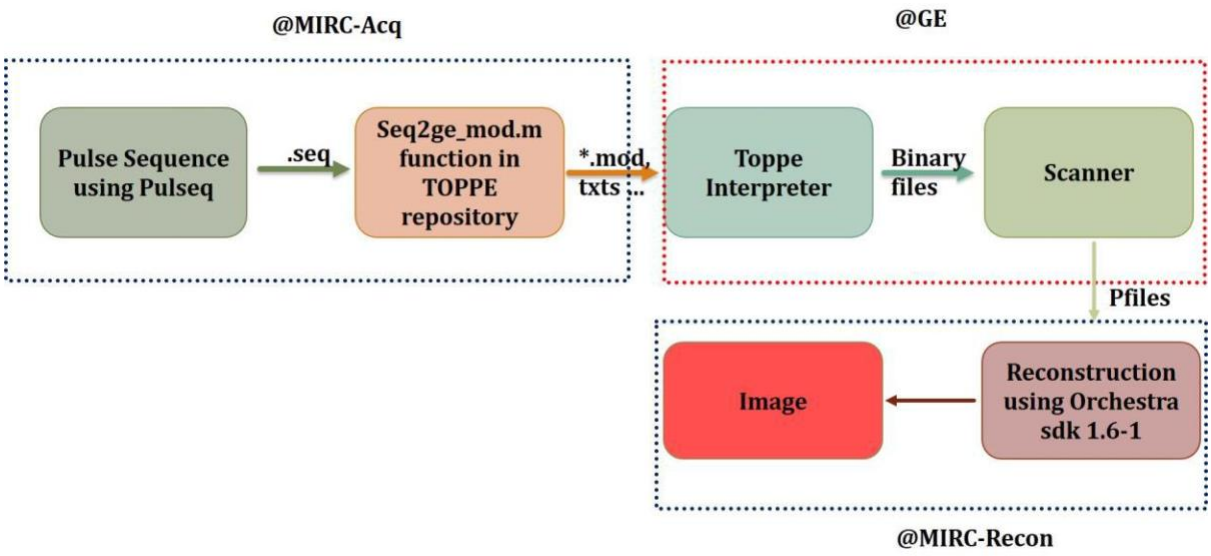


Figure 3: Workflow from MIRC to GE scanner and MIRC; Generated at MIRC.

For example:

1. Coded GRE sequence in MATLAB using Pulseq, compatible with TOPPE structure.
2. seq file of the GRE sequence was giving as input to seq2ge.m file.
3. Used mat2mod.m file to generate Readout.mod, core*.mod and tipdown.mod for the sequence.
4. Validated the waveforms through scansim.m file.
5. Generated 12 unique cores along with scanloop.txt, cores/module.txt.

5.1 Modules.txt

Total number of unique cores
12

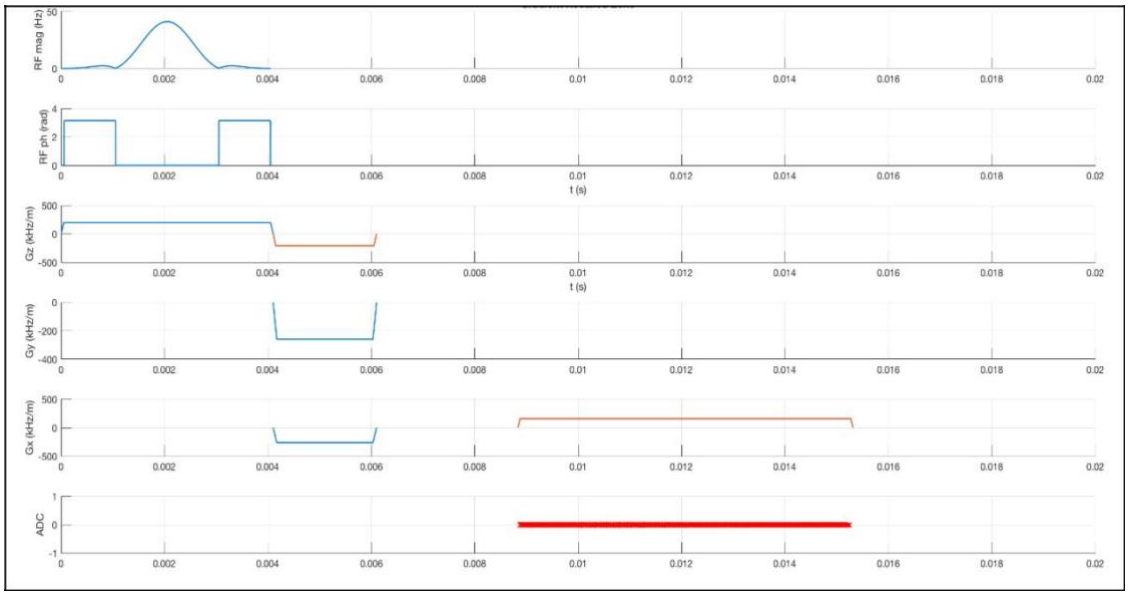
wavfile_name	duration (us)	has_RF?	has_ADC?
tipdown.mod	2768	1	0
core2.mod	2016	0	0
core3.mod	3444	0	0
readout.mod	6480	0	1
core5.mod	5444	0	0
core6.mod	2016	0	0
core7.mod	2016	0	0
core8.mod	2016	0	0
core9.mod	2016	0	0
core10.mod	2016	0	0
core11.mod	2016	0	0
core12.mod	2016	0	0

5.2 Tipdown.mod

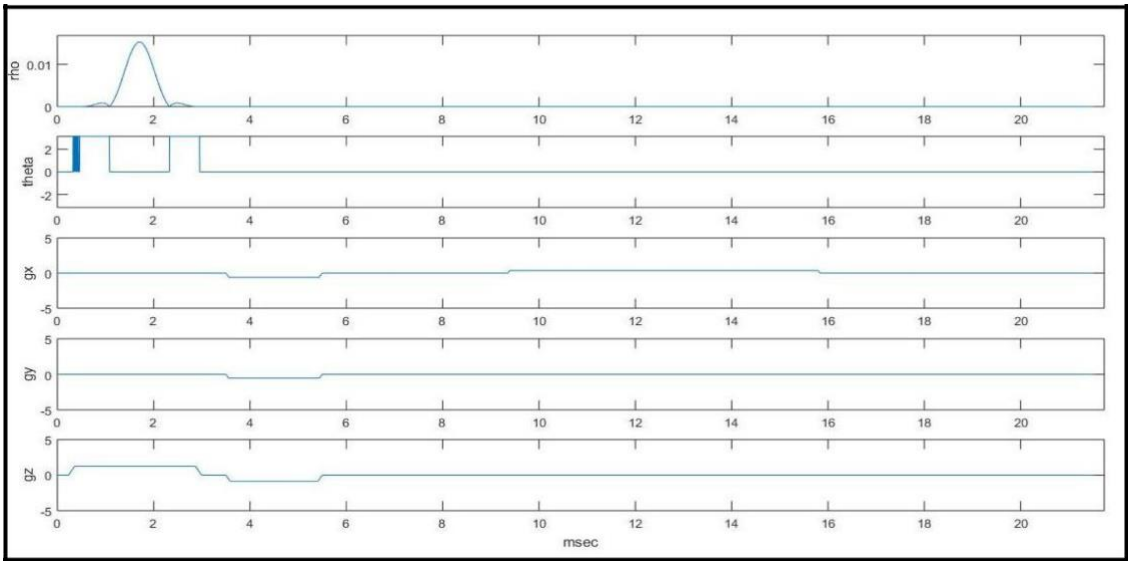
```
RF waveform file for ssfpbanding project.
Created by C:\Users\arush\Desktop\Tools\SEQ2GE\matlab\mat2mod.m on 24-Jul-
2017 16:17:21.
called by (dbstack(3)): seq2ge_mod.m
nchannels = 1, res = 696
Filename: tipdown.mod
Created with seq2ge_mod. Type: excitation.
 ¸b1max: 0.125000
gmax: 5.000000
   ´                                        2.784000
0.250220
0.178755
0.250220
0.330460
0.330460
1.000000
0.125000
0.000116
0.006462
90.000000
2784.000000
2000.000000
1.000000
0.008493
0.000000
      .
      .
0.000000
```

5.3 Waveforms

Pulseq output for 1 TR; FOV=256, Nx=Ny=256, TR/TE=20/10 ms, flip angle =15^0



Scansim.m output for 1 TR, same acquisition parameters as above



5.4 Workflow overview including image reconstruction of GE raw data (P files)

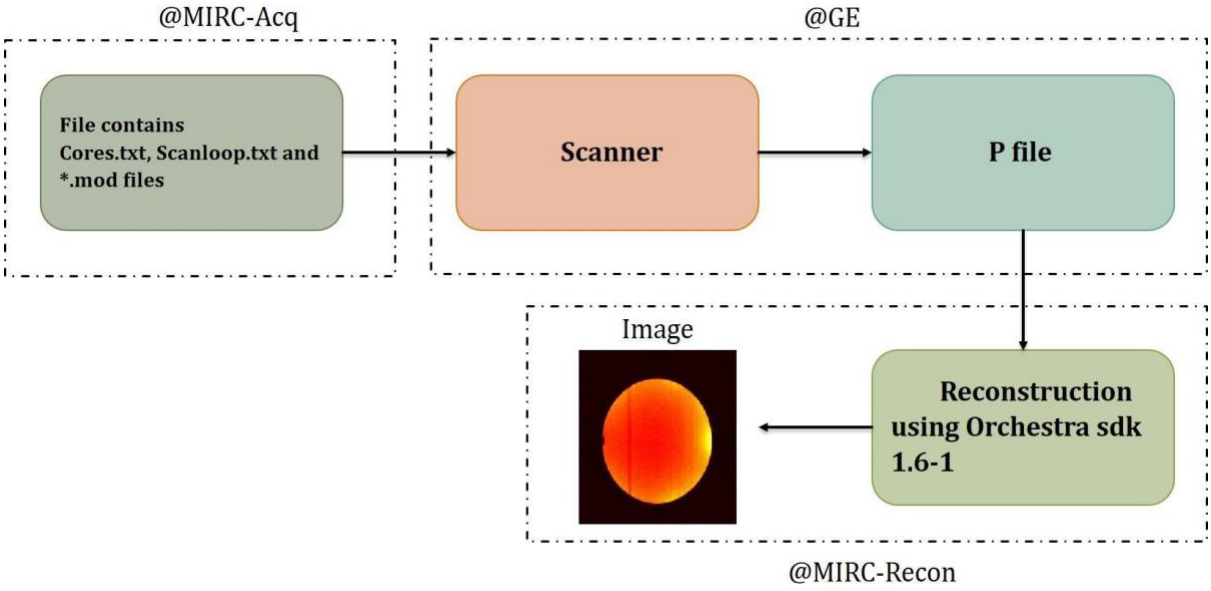


Figure 4: Detailed flow of work from MIRC to GE and MIRC; Generated at MIRC

Mod files generated at MIRC are sent to GE to obtain the P files, once the P files are obtained they are reconstructed using *orchestra sdk 1.6-1.MATLAB* at MIRC to get images.

Check for the following before proceeding to reconstruct data:

1. MATLAB R2015a or later. ☐
2. Orchestra sdk 1.6-1. MATLAB. ☐
3. Toppe_recon_v1_use.m file to reconstruct image. ☐
4. Necessary P files to reconstruct image. ☐
5. Format used to send mod files and get P files should be as
   Sequencename_Pulseq/Python/GPI_Date
   for example: GRE_Pulseq_22082017 ☐

Folder Structure:

MIRC-GE (Folder Name)

1. GPI
    a. Seq files
    b. Sequences
    c. Readme

2. MATLAB
    a. Seq files
    b. Sequences
        i. Pulseq master
        ii. .m files
    c. TOPPE
        i. Seq2ge
    d. Readme

3. PYTHON
    a. Seq files
    b. Sequences
    c. mr_gpi folder
    d. mr_nodes folder
    e. pulseq2jemris folder
    f. Readme

4. ReconGE
    a. Orchestra sdk 1.6-1.matlab
    b. Toppe_recon_v1_use
    c. P files
    d. GERecon.mexa64
    e. GERecon.mexmaci64
    f. GERecon.mexw64

Reference

[1]. Implementation_of_Pulseq_GPI_3814.pdf. Sravan KR et.al

[2]. https://github.com/toppeMRI/UserGuide/blob/master/TOPPE_UserGuide.pdf