

- TRANSP and Simulink [with Dan Boyer]
 - with a brief preview of the framework discussion scheduled for tomorrow at 2:45pm
- TRANSP and the Iter Integrated Modeling & Analysis Suite (IMAS) [with Marina Gorelenkova]

- TRANSP and Simulink [with Dan Boyer]
 - with a brief preview of the framework discussion scheduled for tomorrow at 2:45pm
- TRANSP and the Iter Integrated Modeling & Analysis Suite (IMAS) [with Marina Gorelenkova]

Simulink and TRANSP componentization

- Like virtually every fusion code, TRANSP was developed to be in complete control of execution and data flow
- For TRANSP to be a component of larger, integrated modeling it will need the ability to hand off some control to an external driver (for example a framework)
- We have a user request for making TRANSP available from Simulink simulations
 - Integrated modeling environment running on top of Matlab
 - Predominant for Plasma Control Systems (PCS), including for Iter
- This is immediately useful AND an intermediate step toward componentization
- We will use the model of co-simulation: TRANSP and Simulink halt between time steps to exchange data

TRANSP/Simulink co-simulation approach

- Co-simulation is a symmetric approach: each code hands off the bare minimum amount of control
- TRANSP and Simulink are each in sole control, except when they pause at predetermined points in time to exchange data
- Each runs as a separate process (or multiple processes for parallel TRANSP)
- At the end of a time step, they wait for a data-exchange file to update
- The initial implementation is under active development
- We do not expect to invite beta testers until at least the end of the year

TRANSP/Simulink co-simulation next steps

- Once the initial implementation is working, we plan to migrate to:
 - Proper (but old school) Inter-Process Communication (IPC) for better robustness and performance
 - POSIX MQ or UNIX domain sockets (the latter used by Chrome browser with great success)
 - Memory-based data exchange, when possible
 - mmap() for PlasmaState and POSIX shmem for trcom, both with semaphores
- Three main criteria for choosing IPC: is it part of the common Unices?, will it still be in 20 years?, is it easy to use?
- We have looked at newer IPC (ZeroMQ, nanomsg), but concluded that turn-of-the-millennium IPC meets our needs, without unnecessary complexity
- Final choices will be based on testing for actual use case

- We are enabling TRANSP as synthetic plasma in Simulink PCS in piecemeal and general approach (useful also for other frameworks)
- The Simulink building blocks (models and Level-2 C S-functions) to enable co-simulation with TRANSP have been implemented
- The Simulink model runtransp.slx passes user specified input to an S-function that:
 - creates an input file
 - launches an executable (not yet real TRANSP) and waits for it to finish
 - reads an output file
 - passes the data back to the Simulink model
- Another Simulink model, wait4file.slx, does time stepping and waits for a data-exchange file to be updated at the end of each time step

Screenshot of TRANSP in Simulink

- When TRANSP is ready, we will put all these pieces together and perform co-simulation
- Screenshot of the runtransp model in use:

The screenshot shows the Simulink environment for a model named 'runtransp'. The main workspace displays an S-Function block labeled 'sfuntransp' connected to a 'Display' block. The Display block shows three numerical values: $3.0000e+00$, $4.0000e+00$, and $1.8000e+01$.

The Command Window on the right shows a warning message: "Warning: MATLAB has... For more informati...".

The Diagnostic Viewer at the bottom shows simulation results for two runs:

- Simulation 1: 08:53 AM, Elapsed: 0.149 sec
- Simulation 2: 08:54 AM, Elapsed: 0.656 sec

The Diagnostic Viewer also displays a warning: "Output port 1 of 'runtransp/constant' is not connected. [2 similar]". Below this, the following MATLAB code is visible:

```
mdlStart(); s = 2.000000, A = [1.500000, 2.000000, 9.000000]
mdlStart(); Input data specified in Simulink was written to the TRANSP input file 'transp_input_file.txt'...
mdlOutputs(): Output data from the TRANSP output file 'transp_output_file.txt' was read into Simulink...
mdlUpdate(): Open a shell, with all necessary environment variables defined, to run TRANSP...
mdlUpdate(): We exited TRANSP and the shell, and made it safely back to Simulink...
```

Co-simulation with TRANSP

- We plan to enable co-simulation with TRANSP by providing users with a CoSim module and library that handle IPC
- Callable from expert file
- Screenshot of log file from test of initial, file-based implementation:

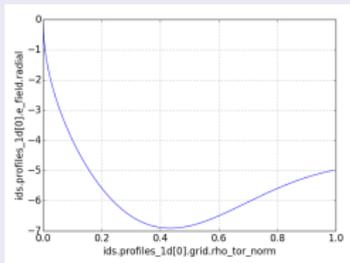
```
*** FULL NEUTRAL SOURCE CALCULATION PERFORMED ***          NSTEP=    12
TA= 4.00000E-02 CPU TIME= 9.31931E-02 SECONDS.  DT= 7.50000E-03
%check_save_state: nbflag          = T
%check_save_state: iwrite_now      = F
%check_save_state: check at wall_hours = 8.71499999999951433E-002
--> plasma_hash("gframe"): TA= 4.0000000E-02 NSTEP=    12 Hash code: 102013058
->PRGCHK: bdy curvature ratio at t= 4.5000E-02 seconds is: 5.4779E-02
% MHDEQ: TG1= 0.040000 ; TG2= 0.045000 ; DTG= 5.000E-03
*** Isolver ***
Avg. GS error:      8.827E-03
Plasma Current:    9.999E+05, target: 1.000E+06, error: 0.007%
Edge Q:            23.182, target: 23.728, error: 2.304%

* MHD EQUILIBRIUM CALCULATED, CPU TIME = 2.4058E+00 SECONDS
DATA R*BT AT EDGE: 9.0591E+01
->EQBDY_CHECK: last flux surface curvature ratio is: 6.1199E-02
teped= 3.694967E+01 neped= 8.114725E+13 width= 2.367699E+00
% GFRAME - GEOMETRY TIMESTEP TG1= 0.040000 TO TG2= 0.045000 @ NSTEP    12
GFRAME TG2 MOMENTS CHECKSUM: 1.1100893944771D+04
CoSim, waiting for file update: current time is Fri Apr 28 14:32:40 2017, data-exchange file was modified on Fri Apr 28 14:32:16 2017
CoSim, waiting for file update: current time is Fri Apr 28 14:32:50 2017, data-exchange file was modified on Fri Apr 28 14:32:49 2017
CoSim, data-exchange file was updated
%MFRCCHK - LABEL "BALE0_SGF", # 1= -1.30321E-42 RESET TO ZERO
```

- We are developing data translators between TRANSP and IMAS
- Source code is available at
`https://github.com/transp/transp-imas-translator`
- Data translation in each direction is done by a standalone executable
- Implemented in Fortran90 and making calls to TRANSP and IMAS libraries
- The transp2imas translator is complete enough to allow NUBEAM simulations using input data from IMAS
 - Is being tested by comparing two NUBEAM simulations, one directly using TRANSP output data, the other with the same data first translated to IMAS format
 - See Marina's presentation

Problems with differences between IMAS versions

- Plot of `core_profiles%profiles_1d(nfinal)%e_field%radial`:



- Data written on my desktop by `transp2imas` linked with IMAS from git repo August 29, 2016
- Error message when trying to build `transp2imas` with IMAS version 3.2.2ua13.3.2 (the latest version on PPPL cluster):

```
transp2imas.f90(1365): error #6460: This is not a field name that is defined in
the encompassing structure. [E_FIELD]
      allocate(cp%profiles_1d(it)%e_field%radial(offset))
      ^
transp2imas.f90(1365): error #6460: This is not a field name that is defined in
the encompassing structure. [RADIAL]
      allocate(cp%profiles_1d(it)%e_field%radial(offset))
      ^
```

- There are other similar changes that break code linked with IMAS

Preview of tomorrow's framework discussion

- The goal will be to solicit input from the user community
 - Do you use TRANSP coupled to other codes, using OMFIT, Simulink or similar?
 - If not, do you see a need for it?
 - Anything else that is relevant we should know about?
- Some previous framework efforts had mixed success
- We plan to take pragmatic approach, driven by user needs
- Discussion tomorrow at 2:45pm, bring your wish list and we will mull it over

- We are developing a TRANSP-Simulink co-simulation capability
 - Directly useful for Simulink PCS with TRANSP as synthetic plasma
 - Also a pilot project for using TRANSP in framework
- IMAS data can currently be tentatively used with TRANSP and NUBEAM
- We can not recommend it for general use until IMAS updates stop breaking our translators

Extra slide: Screenshot of TRANSP in Simulink

- Specifying input:

The screenshot shows the Simulink environment with a block diagram containing an 'sfuntransp' S-Function block connected to a 'Display' block. The 'Block Parameters: S-Function' dialog box is open, showing the following configuration:

Block Parameters: S-Function

S-Function

User-definable block. Blocks can be written in C, MATLAB (Level-1), and Fortran and must conform to S-function standards. The variables t , x , u , and $flag$ are automatically passed to the S-function by Simulink. You can specify additional parameters in the 'S-function parameters' field. If the S-function block requires additional source files for building generated code, specify the filenames in the 'S-function modules' field. Enter the filenames only; do not use extensions or full pathnames, e.g., enter 'src src1', not 'src.c src1.c'.

Parameters

S-function name: Edit

S-function parameters:

S-function modules:

Buttons: OK, Cancel, Help, Apply