What is TACS?

- Transient Analysis of Control Systems (TACS)
- Introduced to EMTP in 1976
- Developed to model controls for HVdc converters (Pacific Intertie)
- Model interactions between system transients and control circuits
- Several variants on parallel development paths:
  EPRI-DCG, EMTDC, ATP
- We'll hear about future developments in EMTP control modeling later

U I Department of Electrical Engineering
ECE 529: Utility Applications of Power Electronics
Session 5

What TACS Can Do?

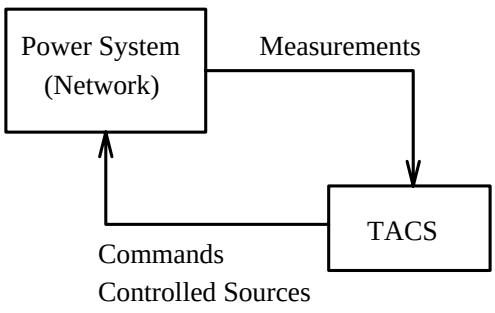
- Model control systems
  - Generator excitation and governor control
  - Control loops for power electronic converters
  - Firing circuits for power semiconductors
  - Relay algorithms
- Monitor and post-process network variables
  - Analog and digital filtering
  - Compute FFT's
  - Compute motor/generator torque
  - Perform reference frame transformations
  - Compute RMS values of voltage/current

What TACS Can Do?

- Simulate mechanical and electromechanical systems
- Non-linear response
- Create models for devices or behavior not represented by built-in EMTP models
- Multi-frequency or variable frequency sources

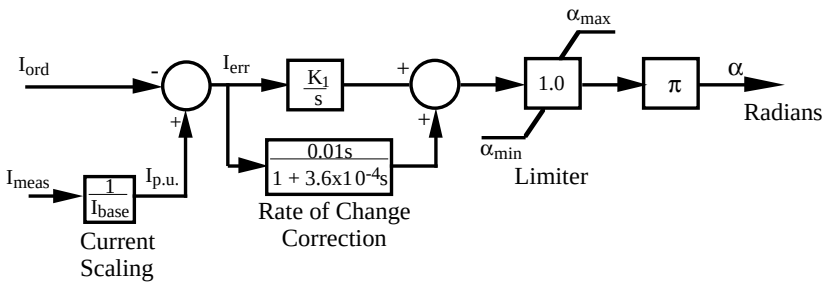
TACS Relationship to Network Solution

- Control system models solved separately from network
- Different equation formats
- Solved separately (network then TACS at each time step)
- Leads to one step time delay when info back to network from TACS



General Data Format

- TACS is designed for user to enter data as read off of block diagram
- Laplace domain representation
- Still have digital time domain solution
- Signal and variable names up to 6 characters
- Arbitrary connection of blocks



TACS and ATPDraw

- ATPDraw has an interface for TACS
- ATPDraw interface to TACS not as nice as circuit interface
- Connections between measurements and TACS optional
- Several of the sample cases sent with ATPDraw have TACS modeling in them

Types of TACS Cases

- Network only (TACS not included)
- TACS used in conjunction with network
  - TACS started with line “TACS HYBRID”
  - TACS ends with BLANK line
- TACS alone (no network)
  - TACS started with line “TACS STAND ALONE”
  - TACS ends with BLANK line
  - No BLANK lines ended network sections


```
BEGIN NEW DATA CASE
C DeltaT<---TMax<---XOpt<---COpt<-Epsiln<-TolMat<-TStart

C --IOut<--IPlot<-IDoubl<-KSSOut<-MaxOut<---IPun<-MemSav<---ICat<-NEnerg<-IPrSup

TACS HYBRID
C Enter all TACS components
C TACS output requests (to plot data file)
C
BLANK   end of TACS
C ..........................  Circuit data  ..................................
BLANK ends circuit data
C
C ..........................  Switch data  ...................................
C includes TACS controlled switches.
BLANK ends switch data
C
C ..........................Source data  ....................................
C include TACS controlled sources.
BLANK ends source data
C
C ..........................  Output Request Data  ...........................
C
C Node voltage requests for network. DO NOT repeat TACS output requests here!!!
C
BLANK ends output requests
C
C If used: request network and TACS variables for plotting
BLANK ends plot request
BLANK ends all cases
```

```
BEGIN NEW DATA CASE
C User can insert special requests cards here
C
C ..........................  Miscellaneous data ..............................
C DeltaT<---TMax<---XOpt<---COpt<-Epsiln<-TolMat<-TStart

C --IOut<--IPlot<-IDoubl<-KSSOut<-MaxOut<---IPun<-MemSav<---ICat<-NEnerg<-IPrSup

C
TACS STAND ALONE
C Enter all TACS components, including:
C               Transfer functions
C               Sources
C               Supplemental variables
C               Initial conditions
C               TACS output requests (to plot data file)
C
BLANK end of TACS
C
C Notice that there are no cards for the network, and no blank lines to end
C the sections.
C
C ............... Plot Request (most people no longer use this) ............
C
C If used: request network and TACS variables for plotting
C
BLANK ends plot request
BLANK ends all cases
```


TACS Pre-Defined Internal Sources

- TIMEX = simulation time in seconds (starting from 0.0)
- ISTEP = number of time steps since simulation began
- DELTAT = simulation time step (TIMEX = ISTEP *DELTAT)
- FREQHZ = system frequency in Hz.
  Defined by first sinusoidal source seen in simulation
- OMEGAR = 2 * π * FREQHZ (units rad/sec)
- ZERO = 0.0
- MINUS1 = -1.0
- PLUS1 = 1.0
- UNITY = 1.0 (use PLUS1)
- PI = 3.14159

User-Define Sources

- Used as inputs to other TACS blocks
- User specifies start and stop times for each
- Type 11: DC source (called Level Signal in the rule book)
- Type 14: Cosine (AC source).
- Type 23: Pulse (Alternates between 0.0 and defined amplitude)
- Type 24: Ramp (output rises from zero to amplitude at end of period)

TACS Measurements from the Network

- Variable name must match node from network
- Type 90: Node voltage measurement.
- Type 91: Switch current.
- Type 92: Internal variables from special EMTP components.
  Examples include variables from within the dynamic synchronous machine and the universal machine.
- Type 93: Switch position.
  Output = 0.0 if switch is open and 1.0 if switch is closed.

Template for TACS Sources

```
C TACS SOURCES
C
C   TYPE       A        B          C
C   11        AMPL     -----      -----      DC source
C   14        AMPL     FREQ       PHASE      AC source
C   23        AMPL     T(sec)     WIDTH      pulse
C   24        AMPL     T(sec)     -----      ramp
C   90        1.0 dc   FQ ac      -----      EMTP VOLTAGE
C   91        1.0 dc   FQ ac      -----      EMTP AMPS
C   92                                       Special
C   93                                       SW state
C
C TACS SOURCE
C <TYPE code in the first two columns
C OUTPUT  <-----A---<----B----<----C----                  <-T-START-<-T-STOP--
```


TACS Pre-defined Devices

```
C DEVICES
C
C  CODE                            CODE
C   50  frequency-sensor            58  integrator
C   51  relay switch                59  derivative
C   52  level switch                60  input IF
C   53  transport delay             61  signal selector
C   54  pulse delay                 62  sample and track
C   55  digitizer                   63  min/max function
C   56  point-by-point function     64  min/max tracking
C   57  time-sequenced switch       65  counter
C                                   66  RMS value
C
C    use 88 in the first two columns
C       ||<CODE                                    see rule book
C DEVICE||
C       ||
C OUTPUTvv+IN1--> +IN2--> +IN3--> +IN4--> +IN5--> <--A--<--B--<--C--<--D--<--E--
88.....>^^+.....^ +.....^ +.....^                  ^.....^.....^.....^.....^.....
```

Transfer Functions

- User can define general transfer functions
- Laplace domain polynomials in numerator and denominator

$$G(s) = gain * \frac{N_0 + N_1 s + ... + N_m s^m}{D_0 + D_1 s + ... + D_m s^m}$$

- Can have constant gain block (zero order transfer function)
- Transfer functions are converted to difference equations

Limiters

- User can specify limits for output of transfer functions
- Limits set by numbers or outputs of other TACS blocks
- Windup limiter: has static or soft limits
  - Zero order transfer function
  - TACS processes block and then looks at output
  - Output prior to limit can move far beyond the limit
  - Can take considerable time to back off of limit
- Non-windup limiter: has dynamic or hard limits
  - Backs off of limit quickly (doesn't saturate)
  - Implement with first order transfer function
  - Can only have a pole, no zeros
  - Any additional poles and all zeros are ignored

Transfer Function Template

```
C TRANSFER FUNCTION
C
C <- the order of the highest power of "s" is place in the first two columns
C    if greater than 0  two more lines follow for numerator and denominator
C
C  TRANSFER FUNCTION                                          LIMITS
C                                                           fixed     NAMED
C OUTPUT  +IN1--> +IN2--> +IN3--> +IN4--> +IN5--> <-gain<--low<-highLOW-->HIGH->

C
C Put the order of the transfer function in columns 1 and 2 of
C the above line. If the card is a z-block put a zero in column 2
C or leave it blank.
C
C<-N/D-0--<-N/D-1---<--N/D-2--<--N/D-3--<--N/D-4--<--N/D-5--<--N/D-6--<--N/D-7--

C
C There can be multiple lines for each depending on the order of the polynomial
C It is possible to have an order greater than 7.
```


Free Format FORTRAN Expressions

- Also called supplemental variables
- User supplied, free-format FORTRAN statements
- Three categories of supplemental devices
  - Type 99: Input group. Inputs variables defined by other Type 99 statements or TACS signal sources.
  - Type 98: Output group. Outputs of these statements used for other type 98 statements or be used as TACS outputs.
  - Type 88: Inside group. Cover all cases omitted in the above 2 groups, but can also serve in the same capacity as both type 98 and type 99. Many users only use type 88.

```
C FORTRAN EXPRESSION
C
C OUTPUT  = free-format Fortran expression.............................
88.....^  =
```

Available FORTRAN Expressions and Limitations

- Algebraic operators: + , - , * , / , **
- Relational operators
  1. .EQ. , .NE. , .LT. , .LE. , .GE. , .GT.
  2. The outputs of these logical operators are binary (0.0 and 1.0)
- FORTRAN functions
  1. SQRT, ABS, EXP, LOG, LOG10
  2. SIN, COS, TAN, COTAN (argument in radians)
  3. ASIN, ACOS, ATAN (answer in radians)
  4. SINH, COSH, TANH

Available FORTRAN Expressions and Limitations (cont.)

- Special functions
  - TRUNC, MINUS, INVRS
  - RAD, DEG, NOT
  - SIGN (-1 for neg. number, +1 for pos. number or zero)
  - SEQ6, RAN (random number generator)
- The user is allowed up to 20 levels of nesting.
- Can't do SQRT(A**2+B**2)
- Can combine functions: SQRT(ABS(INVRS(X)))
- Does not allow: GOTO, IF, DO, SUBROUTINE, or FUNCTION

Outputs from TACS

- TACS controlled sources in network portion of EMTP.
  - Source type 60 in the network
  - The variable name for the source must match a name in TACS
- TACS controlled switches: type 11, 12, or 13.
- Interface to type 59 dynamic synchronous machine
- Interface connections to type 19 dynamic machine.
- TACS generated modulating source, type 17 in the network.
  Creates a variable scaling factor that multiplies an existing source
- Also send output to plot data file

```
C ...........OUTPUT FROM TACS
C VAR1->VAR2->VAR3->VAR4->VAR5->VAR6->VAR7->VAR8->VAR9->VAR10>
33
```


Initial Conditions

- Need to set steady-state initial conditions for some TACS blocks
- Especially if correct outputs at time $t = 0^+$ are desired
- Network and TACS undergo separate steady-state initialization with the network computed first
- TACS outputs to Network are zero unless otherwise specified
- Set initial conditions for outputs of
  - First order and higher transfer functions
  - FORTRAN statements
  - Many of the built-in devices
- Set initial conditions with:

```
C VAR1->  <-----MAGN
77
```