Department of Electrical and Electronic Engineering
Shahjalal University of Science and Technology

EEE 126: Electrical Circuit Simulation Laboratory

Experiment # 06: Sub-Circuits using Net listing and Schematics and Three Phase Circuits

Objective:

The objective of this experiment is to learn how to write and invoke a sub-circuit and implement the concept in three phase circuits.

Simple Subcircuits in PSpice

One of the more useful concepts in PSpice is the use of *subcircuits* to group elements into clusters in order to replicate the clusters without having to re-enter all the elements each time. This is very useful for several reasons. First is the labor savings of replacing many lines of circuit data with a single subcircuit call. This also reduces the chance of making a typo. Second, the use of a subcircuit usually improves clarity by removing confusing clutter. The user can suppress printing unwanted details internal to a subcircuit, thus making the output easier to understand. If desired, the user can place often-used subcircuits into an *include* file so that the main source file for the problem is kept simple. Then the definition of the subcircuit is out of sight entirely.

Coding a Subcircuit

Each subcircuit used in a study must have a unique name. This is true of any other circuit element. Also, there must be a list of at least two nodes that can be connected to elements external to the subcircuit. A subcircuit can have many external node connections, if needed. Later, we will find that parameters can be passed to a subcircuit in order to allow unique behavior and responses from an instance of a subcircuit.

The initial line of a subcircuit section must begin with ".SUBCKT," followed by the name and then the external node list. After that, optional features (not to be discussed yet) can be added. The best method of understanding the use of a subcircuit is by example. Below, we find a cluster of components that can be combined into a subcircuit.

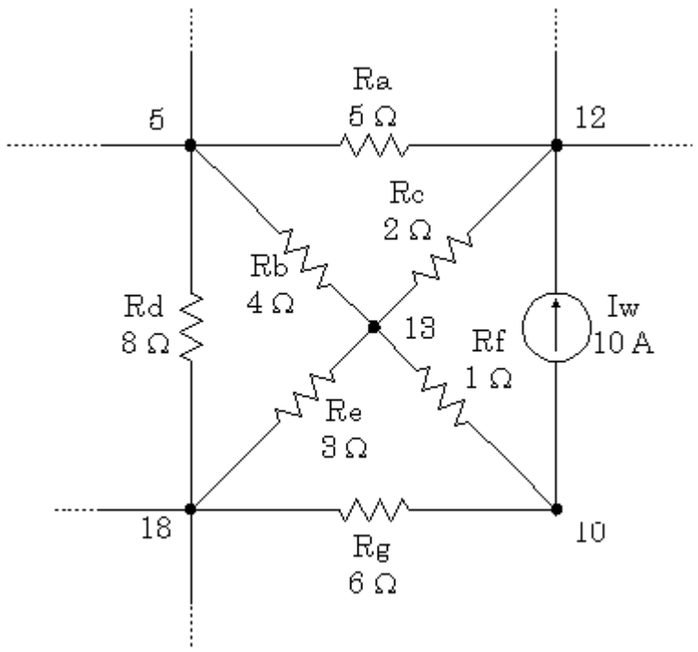

Fig. 1 : Subcircuit of Example_1

Note that nodes 5, 12 and 18 have external connections. Therefore, they must be included in the node list in the subcircuit definition. Nodes 10 and 13 do not have external connections and need not be (indeed *should* not be) included in this node list. They are internal nodes and will be used to help define the subcircuit. Now, we can code the above subcircuit as follows. Note that the code could be embedded into the rest of the code for the main circuit or could be placed in a separate *include* file.

```
*        name           nodelist
.SUBCKT Example_1    5    12    18
Iw    10    12    DC    10A
Ra     5    12    5.0
Rb     5    13    4.0
Rc    12    13    2.0
Rd     5    18    8.0
Re    13    18    3.0
Rf    10    13    1.0
Rg    10    18    6.0
.ENDS
```

Note that the subcircuit section must be terminated with a ".ENDS" command.

Invoking a Subcircuit

All subcircuit calls are made by declaring a part with a unique name beginning with "X," followed by the node list and then the subcircuit name. The node list in the calling statement must have the same number of nodes as the node list in the subcircuit definition. To demonstrate the use of the calling statement, we present the following main circuit which contains two instances of the above subcircuit. X1 and X2 are the two instances of the subcircuit "Example_1." For added clarity, the subcircuit's defined external nodes are shown in parentheses. Note that these nodes are mapped into the main circuit by *different numbers*.

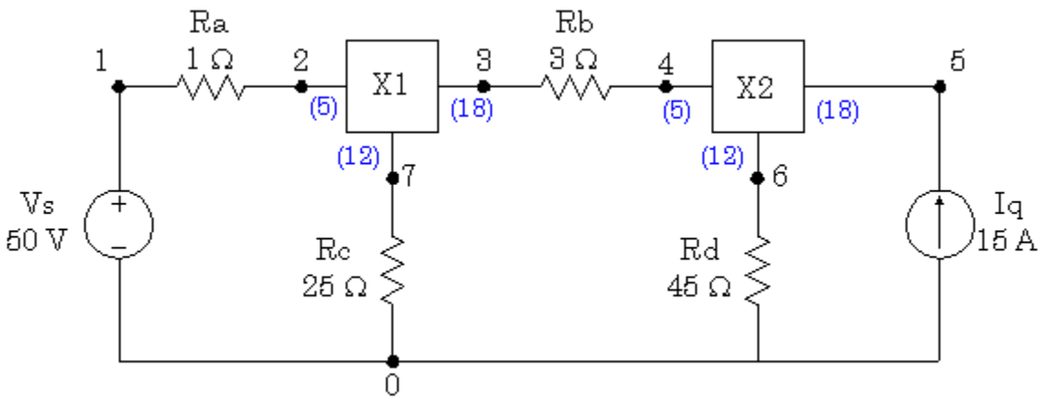

Fig. 2: Circuit using the subcircuit Example_1

The code for the above circuit with the subcircuit included follows:

```
Subcircuit Example No. 1
*        name           nodelist
.SUBCKT Example_1    5    12    18
Iw    10    12    DC    10A
Ra     5    12    5.0
Rb     5    13    4.0
Rc    12    13    2.0
Rd     5    18    8.0
Re    13    18    3.0
Rf    10    13    1.0
Rg    10    18    6.0
.ENDS
```

```
Vs     1     0    DC    50V
Ra     1     2    1.0   ; different from Ra above
Rb     3     4    3.0   ; different from Rb above
Rc     7     0   25.0   ; different from Rc above
Rd     6     0   45.0   ; different from Rd above

Iq     0     5    DC    15A
*      nodelist        name
X1     2     7     3    Example_1
X2     4     6     5    Example_1
.END
```

What are the voltages of node 10 and node 13 of the subcircuit X1 and X2?

Scope of Element Names and Nodes in a Subcircuit

Scope of names and nodes is local to a subcircuit. In the main circuit of which the above subcircuit is a part, there is a node 5 and there are resistors with the names of "Ra," "Rb," "Rc," and "Rd," and PSpice can keep these apparent duplications separated. If the subcircuit were invoked as "X1," for example, PSpice would consider the subcircuit parts as "X1.Ra," "X1.Rb" and so on. Additionally, the internal node numbers would be treated as "X1.10," "X2.10," "X2.13" and so forth. Thus PSpice maintains uniqueness of element names and node numbers.

Nesting of Subcircuits

Subcircuit *calls* may be nested as long as they are not circular. In other words, we can have a part name starting with "X" within a .SUBCKT/.ENDS block provided that the "X" part definition does not call on that block for its own definition.

However, subcircuit *definitions* may *not* be nested. i.e., we can't have one .SUBCKT/.ENDS block nested within another.

The advantage of using the subcircuit in PSpice is now apparent. We could have replicated the subcircuit many times, using only one line of code per replication.

Subcircuits using Schematics

DCLICKL=Double CLICK Left, CLICKL=CLICK Left

Step 1: Create the MODEL

1. Draw the subcircuit of Fig. 1.
2. Replace the input (nodes 5 and 12) and output (node 18) lines with the *interface port symbols* IF_IN and IF_OUT (from *port.slb*). To label each port symbol, **DCLICKL** on each symbol and fill in.
3. Save the schematic to file *Ex_1.sch*.
4. Perform **Tools, Create Subcircuit** to generate the subcircuit MODEL definition. This model definition is automatically stored as *Ex_1.sub* in the same directory as *Ex_1.sch*.
5. To make the MODEL definition (*Ex_1.sub*) available locally (valid only to *Ex_1.sch*): **Analysis, Library and Include Files**, enter *Ex_1.sub* in the File Name Field, **Add Library** (no asterisk), **OK**. (Add Library* would make the model global and available to all schematics).

Step 2: Create the PART

6. To create a PART, we first create a symbol. Assign the symbol a PART name (such as EX_1), and store the PART in a symbol library (such as *userlib.slb*). The procedure to do this is as follows:

File, Symbolize, type PART name *Ex_1* in *Save As* dialog box, **OK**, and note the *Choose Library for Schematic Symbol dialog box*. To save part *Ex_1* in library *userlib.slb* in a specific directory **CLICKL** to select *userlib*, **OPEN**. (If *userlib.slb* does not exist, enter the whole path in file name box, **Open**.)

7. The next step is to edit the new PART symbol by bringing up the Symbol Editor: **File, Edit Library, File, Open, DCLICKL** on *userlib.slb*, **Part, Get, DCLICKL** on *Ex_1* to bring up the initial symbol.
8. Modify the initial symbol by moving the pins and attributes by **select, DRAG**. Also, **DCLICKL** on any attributeO or pin to place pin numbers and reorient pin names. You may wish to remain with the initial symbol as well.
9. To save the edited PART symbol again in the symbol library (*userlib.slb*): **File, Save** (and if necessary, **Yes** add to list of configured libraries).

Step 3:Associate the MDEL with the PART

10. To associate the MODEL definition (*Ex_1.sub*) with the PART symbol (in *userlib.slb*): **Part, Attributes, CLICKL** on PART=, enter *Ex_1* in Value field, **Save Attr, CLICKL** on MODEL=, enter *Ex_1* in Value field, **Save Attr, OK**. (**DRAG** part name *Ex_1* to desired position.
11. To return to Schematics, **File, Close,** (and if necessary, **Yes** save changes to Part, **Yes** save changes to library).
12. **File, Close** to exit the *Ex_1.sch* window (and if necessary, **Yes** save all changes).

Circuit design using the subcircuit

13. Create a new schematic screen.
14. Fetch subcircuit PART *Ex_1* from *userlib.slb*.
15. Add additional components to create the circuit in Fig. 2.
16. Store the circuit in a file of your choosing, run PSpice (transient mode) and test the circuit.

Exercise of Three Phase Supply using the load as a subcircuit:

Define a RLC series circuit with R=100Ω, L=0.31H and C=31.83μF as a subcircuit. Use the subcircuit in three different phases of a balanced three phase supply with f=50Hz and amplitude=100V. Connect the sources and the loads in YY, YΔ, ΔY and/or ΔΔ and check the relationships between line current and phase current and line voltage and phase voltage. Check whether they match with your calculations.