NeuroCheck Program Flow Notes - Course Handout

With user-configured software products, the goals of simplicity and power / flexibility often conflict with each other. One of the methods that NeuroCheck uses to offer *both* is to provide automatic default configurations suitable for basic applications, and then offer powerful tools for modifying these for more complex or unusual applications. One of the areas where NeuroCheck uses this concept is program flow. Program flow refers to the sequence of execution. NeuroCheck also has the capability (and similar method to simplify it) to let you select & control the passage of data and data objects from one function to another, but that is not the subject of this handout.

A recap of NeuroCheck program hierarchy terminology is useful at this point. NeuroCheck includes & supports an extra level of program hierarchy. What many other products would call an entire program is treated as a subprogram in NeuroCheck:

NeuroCheck Term	**Level**	**Description**
Check Function	Lowest	Individual tool
Check	Medium	Subprogram. On simple applications, one check is often the entire program
Check Routine	Highest	Overall program, stored as a file, consisting of one or more Checks plus other items such as “start actions” and ”end actions”

The default program flow is that Check Functions (tools) and Checks are executed sequentially. (see view in left pane, left tab in Manual or Test mode) Where there is a failure at a tool within a Check, the rest of the tools in the Check are not executed, and NeuroCheck determines that the result of that Check is “Not OK”. Examples of failure at a tool are:

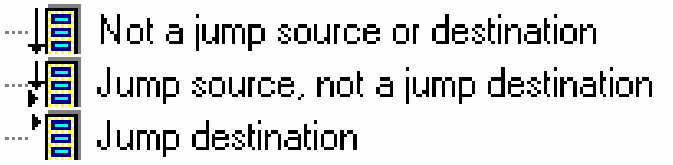
- A measurement is taken, and the measurement is determined to be out of tolerance.
- A measurement related tool failed to execute because the item to be measured was missing from the image. .

And when this does not occur, NeuroCheck determines the result of the Check to be “OK”. Then the default is that at the end of a check, execution moves to the next Check. But this behavior may be modified through the Properties->Flow Control dialog box for that Check. Using this, for each result case, (“Check OK” and “Check not OK) you may select a different check for execution to jump to, and NeuroCheck will follow this.

Any jumps programmed into the destination Check will be similarly followed. Otherwise, the “destination” check will (upon completion of itself) send execution back to the end of the Check that sent execution to it, and normal sequential execution will continue. So, then the next normal (“non-destination”) Check in the sequence after the “source” will be executed.

(The “ignore” term in the dialog box is not relevant to program flow. It refers to effect on the “result” of the overall Check Routine, this “result” being a property which you may or may not wish to use for things such as I/O control.)

Programming a Check to be a “destination” also converts it into a Check that can only be entered by a jump. This status is indicated by a change in the arrows to the left of the Check icon as follows



So the following facts arise from the above:

- Eventually execution returns to the initial source of the jump
- Eventually, all non-destination checks are executed.
- A check that is made to be a destination of a jump will never be exited in the simple routine sequential manner. It will either follow it’s own “jump”, or return execution to the Check that “called” it.