Calculators today have a very straight forward interface, where the arithmetic is all done the way it is though it school, and the *pemdas* order of operations is observed. However the first handheld calculators did not use this notation, instead they used what is called *Reverse Polish Notation*. In this format, numbers and operators are typed in the order they need to be operated on, and the order of operations is implied. For example:

Normal Notation:

(3 + 5) \* 2 + 8

Reverse Polish Notation (RPN):

3 5 + 2 \*8 +

Both of these will evaluate to 24.

In normal notation following the order of operations, the 3 + 5 is evaluated first, multiplied by the 2 and finally the 8 is added.

In RPN the 3 and 5 and put on the ‘stack’. The addition looks at the top two items on the stack and evaluated and 8 is put back on the stack. The 2 is then put on the stack and the 8 and 2 are multiplied, and the 16 is put on the stack. Finally the same happens with the 8. In this way not parentheses are needed.

Tasks

1. Open the **Reverse Polish Notation.vi** and try to do some simple math.
2. Open the code for this and look at how the event structure is working (you do not need to implement this)
3. Use the **highlight execution** and try a few more equations. Observe how the data is handled as you press different keys
4. Probe the **Display Test** wire (see code comment) to understand what the format looks like for the code
5. Replace the **Reverse Polish Notation Solution.vi** with your own solution

Notes:

* If there are not enough numbers on the stack, use 0 (so 7 \* evaluates to 0)
* If there are too many numbers on the stack, display the top number ( so 2 3 4 5 + evaluates to 9)

New Topics Covered:

* Stack
* Shift Registers
* String manipulation

Other topics viewed but not implemented:

* Event Structures
* Control References