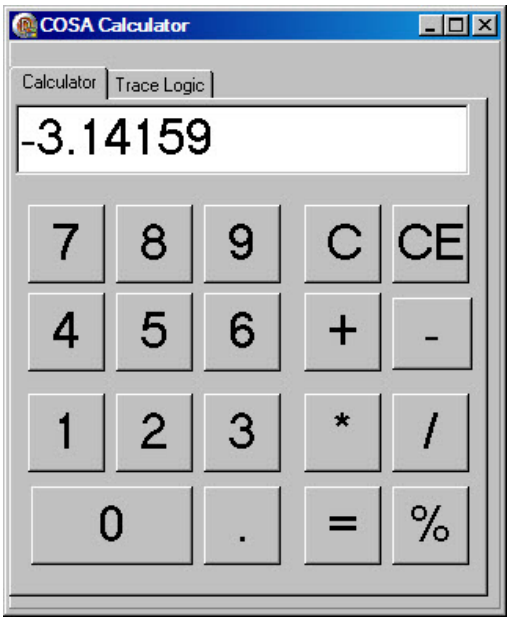
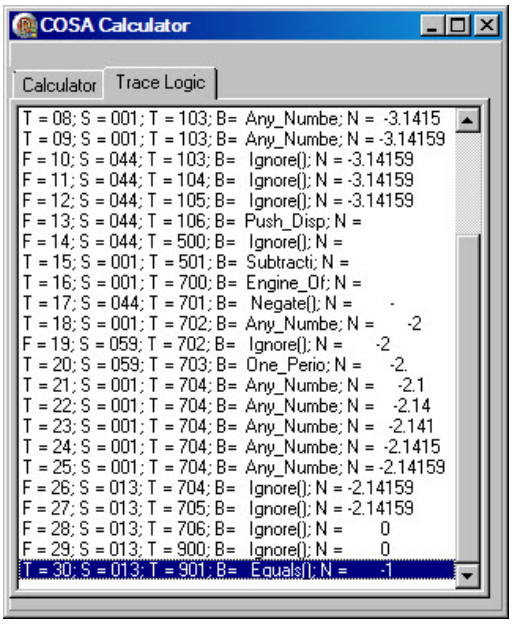
Software Calculator Challenge





Try to Design and Develop a Better Calculator

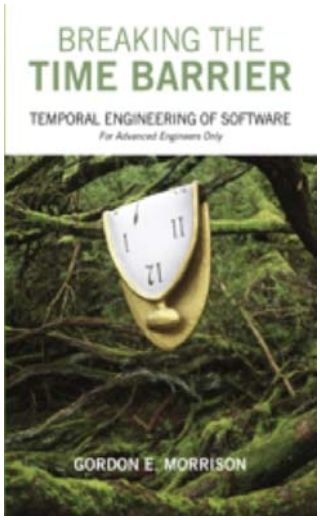
Send any questions or results to:

calculator@VSMerlot.com

Or logon to:
www.SpruceCommunity.org

Breaking the Time Barrier – The Temporal Engineering of Software

By Gordon E. Morrison



Specification

Subject to change. Changes will be posted.

The calculator must be implemented in a point and click Windows GUI using C, C++, or Delphi. You can't copy the calculator design, statechart, and implementation from my book "*Breaking the Time Barrier"*. And you can't use my temporal strategy.

The calculator GUI design spec is at the top of this document. The calculator has nineteen buttons. The buttons respond to events. The buttons are: "add", "subtract", "multiply", "divide", "percent", and "equal sign" buttons. The calculator also has buttons numbered one through nine, a zero button, and a decimal point button.

To operate the calculator a user clicks on a 'number' button to display the number. Clicking on the 'decimal' will allow the user to enter a fraction portion. Clicking on the 'subtract' button allows the user to enter a 'negative' number before the whole portion of the number is entered.

The calculator is capable of doing chain operations, i.e. "A+B-C+D=" or "A*D/F=". The calculator doesn't have a change sign key.

The calculator must provide a trace file counting every 'if-then-else-case-switch-default' (ITE) logic transition. The same trace information must also be provided in the 'listbox' under the 'Trace' tab of the calculator GUI.

Efficiency will be determined by the amount of work being done for each given problem. The trace file should show every test made by the application. The trace file should show the following:

Column 1: ITE Count – the number of True/False tests being made.
Column 2: Dynamic State Value
Column 3: Function-Procedure-State (FPS) Name
Column 4: Unique Trace Number for each FPS name of Column 3
Column 5: The result of each operation in Column 1. (each click entry)

The trace file must be in text format and be named: "<company>_calc_challenge.txt". See the "ITE Trace" or the "COSA Trace" links at www.vsmerlot.com for examples of these formats.

The efficiency will be determined by the number of click entries for each specific problem entered into the calculator divided by the number of logic tests represented by the last number in column 1 for each problem. Each test for efficiency will start with a fresh calculator except the tests involving chain operations.

Efficiency = (number of click entries) / (number of logic tests);

A sample test matrix is included with this document.

Calculator test matrix		
Input	**Output**	Notes
1 + 1 =	2	Spaces for readability – not actually input
-1 + 1 =	0	
1 - - 1	2	
100 % 10=	Operational Error!	Operational Error
100 + 10 %	110	
3 / 0 =	Divide by Zero.	
1 / 3 =	.33333333333333333	15 digits displayed
2 / 3 =	.6666666666666666667	Round off last digit
-2 / 3 =	-.6666666666666666667	Round off last digit
C C C C C C	0	Repeated Clear or CE
-1 + -1 =	-2	
+ + 1 =	Unknown Operation	Operational Error
1.1.1	Operational Error!	
1.1 <clear>	0	
1.1 <clearentry> 1.2=	Operational Error!	Operational Error
9999999999999999*9999999999999999 =	1E32	Limits on display
9.1111111111111111*9.1111111111111111=	83.0123456790123	
1 - - - 1 =	2-Unknown Operation	'Clear' creates Operational Error
100 - -10 %	90	Negative percentages
-3.14159 - - 2.14159 =	-1	**Final Test Steps Count**

All the challenges, data artifacts, collaborations, candidate solutions, and experiments covered in this presentation are accessible from SPRUCE (Systems and Software Producibility Collaboration and Experimentation Environment). SPRUCE is an open web portal designed to bring together DoD software developers, users, and software engineering researchers by collaborating on specifying and solving software producibility challenge problems. SPRUCE emphasizes collaboration around well-defined challenge problems with project-specific artifacts representative of DoD projects, and experimentation for reproducing the stated problems, establishing benchmarks and evaluating solutions. SPRUCE also hosts an online-accessible experimentation facility that stores and replicates experiments easily. SPRUCE is funded by the Office of the Secretary of Defense (OSD) and supported and managed by the Air Force Research Laboratory (AFRL).