# OpenCalc

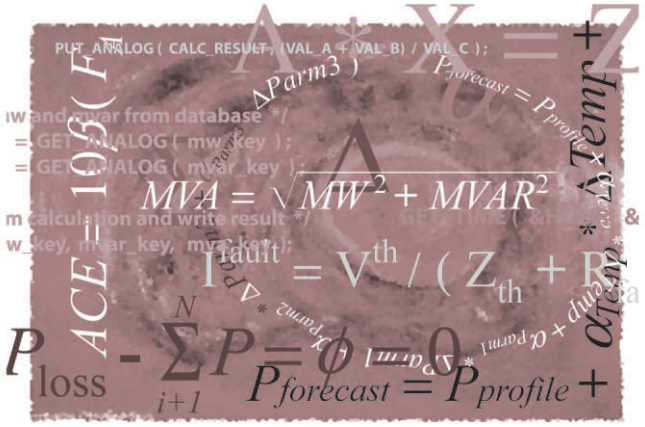
## Calculation Product

The ability for power system maintenance staff to respond to the ever changing needs of the utility business with the capability to create a variety of new applications functions or supplement existing functionality is essential. It is important to possess an intuitive calculation and development product with easy-to-use features and commands that allow efficient design of business logic, system monitoring, and control functions. OSI's **OpenCalc™** Calculation subsystem is an integrated development environment that allows users of the **monarch™** platform to design and create custom calculations logic or sophisticated control and monitoring applications.

The simplicity and user-friendliness of **OpenCalc** allows a development staff with limited programming knowledge to create simple calculations using a diverse library of easy to use, pre-built functions. The more knowledgeable developer is empowered to write sophisticated calculations that execute on a periodic, demand or event basis. Experienced staff can create their own data models within the **monarch** platform, use **OpenCalc** as the logic and calculation engine to perform business logic calculations, and display the solution results via OSI's **OpenView™** Graphical User Interface.

**OpenCalc** calculations can be used to:

- Model business rules
- Create customized data processing to meet unique operational needs
- Perform supplementary calculations
- Perform conditional calculations
- Perform aggregated calculations
- Perform open loop or closed loop control
- Monitor process data and trigger alarms or other applications
- Perform periodic data processing or data maintenance tasks
- Perform data storage or archival tasks
- Trigger other applications to execute on any node in **monarch**
- Exercise any other system functions on a periodic or on-demand basis, conditionally or based on time events




A rich set of libraries with built-in functions simplify creation of straightforward or complex calculations. Data access to the system databases are abstracted and made very uncomplicated via simple function calls. The majority of the **monarch** API function calls are made available via wrappers to allow novice programmers to deploy such functions in their calculations.

Standard function libraries are included for:

- Full mathematical functions
- Integration, Max, Min, Average calculation functions
- Full logical operators
- Power Engineering functions (e.g. Power Factor, MVA, etc.)
- Real-time Database Access (single item, single record)
- Real-time Database Access (entire object, multiple records)
- Full access to data quality codes, tags and other attributes
- Historical data access
- Date/Time manipulation functions
- Serial Communications functions
- Alarm annunciation functions
- Creation of system events
- Control functions (single point or multiple points)
- Raw file I/O

*Continued…*

In addition to the libraries of functions, more advanced users can tap the full power of 'C' programming libraries and functions to perform sophisticated computations. Users can also build their own or use 3rd party libraries to expand functionality and meet their changing business needs.

Resultant calculation scripts can be executed periodically based on a specified periodicity, and order of hierarchy, or can be executed on demand or can be triggered via a system event.

Resultant calculations scripts are impervious to failover and data backups. Such tasks are automatically handled by the **monarch** platform without requiring the calculations developer to consider such complexities.

**OpenCalc** has a combination of simplicity and power that will support any business model, from creation of simple business/engineering calculations to a dynamic extension of **monarch** functionality that customizes the standard system operations to meet specific business needs. This includes incorporation of unique or proprietary business rules which can provide the competitive edge needed in a highly competitive utility environment.