Micromega Corporation

# Release Notes

# uM-FPU64 IDE Release 407

## Changes for Release 407

uM-FPU64 IDE Release 407 adds several new features and fixes some known problems.

## Firmware Upgrade

To use uM-FPU64 IDE r407 software, the uM-FPU64 chip must be running firmware release 405 or higher. Firmware files are supplied with the IDE and installed in the *Firmware* folder of the IDE installation directory. The firmware can be upgraded using the *Tools>Firmware Update…* menu item. Select the appropriate firmware file as follows:

| | |
|---|---|
| 28-pin chip: | *uMFPU64 64K28 Firmware V405.dat* |
| 44-pin chip: | *uMFPU64 64K44 Firmware V405.dat* |

## Changes to IDE Interface

- *Interactive Compiler* window added to the *Tools* menu. Used for writing compiled instructions to the FPU instruction buffer.
- *Debug> Reset* menu item added to allow the FPU to be reset from the IDE

## Changes to Compiler

- `rtc(NUM_TO_DATE,` *register*`)` procedure converts date/time number to values in consecutive registers
- `rtc(DATE_TO_NUM,` *register*`,...)` function procedure converts values in consecutive registers to date/time number
- `rtc(NUM_TO_STR,...)` and `rtc(STR_TO_NUM,...)` functions now has options to convert date, time, or date/time
- `rtc` functions and procedures updated to use new date/time number format (see `RTC` instruction for more information)
- `timelong()` function now returns milliseconds if a 32-bit register is selected, or microseconds if a 64-bit register is selected
- `devio(I2C, START_WRITE,...)` function now allows a long expression for the device ID
- `devio(I2C, START_READ,...)` function added to support multiple reads as a single I²C transaction

- `devio(LCD,...)` functions always set the `RS` pin prior to use. This allows the pin to be shared.

- added `FLOAT32`, `LONG32`, `ULONG32` as valid parameter types for functions
- added `#ide_required` directive to specify the IDE software version required by the source code
- added `#firmware_required` directive to specify the uM-FPU firmware version required by the source code
- `version` procedure stores uM-FPU64 version string in string buffer, and version number in register 0

# Bug Fixes

- fixed problem with pointer array code generation.
- fixed problem in RAM Display Window with certain signed values.
- fixed problem causing hang if function parameter type is unknown.

# Interactive Compiler

The interactive compiler window takes source code, compiles the code and sends it to the uM-FPU64 instruction buffer. This can be used for a variety of testing applications. User-defined functions can be called, devices can be accessed using FPU instructions, etc. The instruction buffer is cleared before the compiled code is sent. If the FPU is running, the code will be executed immediately. If the FPU is currently at a breakpoint, the instructions will be executed when the next *Go* or *Step* command is issued.

Only equations, procedures and assembly code are supported by the interactive compiler, but all of the symbol definitions from the last source code compile can be used in the interactive compiler window. For example:

Call procedure main:

```
main
```

Initialize the LCD on pin 0 and write a test string:

```
devio(LCD, ENABLE, 0, ROWS_4+COLS_20)
devio(LCD, WRITE_STR, "test")
```

A control-click inside the interactive compiler window displays a context-sensitive pop-up menu of all registers, constants, procedures, functions, and operators that are currently defined. Selecting an item from the pop-up will insert that item into the interactive compiler window. This is a useful way to test user-defined functions. Once the functions are programmed into Flash, the interactive compiler window can be used to call the functions for testing. Equations and procedure calls use the same syntax as the source code compiler.