Micromega Corporation

Release Notes

uM-FPU64 IDE Release 406

Changes for Release 406 (c)

- fixed code generation for `%` (mod), `&` (and), | (or), `^` (xor) operators in certain 64-bit calculations
- added IDE symbols `IDE1`, `IDE2`, `IDE3` for `SEROUT` debug mode data channels
- fixed parsing of `SERIAL` function `WRITE_FLOAT`, `WRITE_LONG`, `WRITE_CHAR`, and `READ_CHAR` operations to support `ASYNC`, `IDE1`, `IDE2`, `IDE3` devices
- fixed LTOA to generate expression code before opcode

Changes for Release 406 (b)

- fixed code generation for 32-bit constant comparisons preceded by a 64-bit expression
- fixed error message for invalid `MOP` operations
- fixed code generation for `STRSEL` and `STRFIELD` functions with indirect values
- added `SLOW` as IDE symbol for `DEVIO,I2C` speed of 100 kHz
- added `START_READ` as new action for `DEVIO,I2C` (available in next firmware release)

Changes for Release 406 (a)

- fixed problem that caused an exception if a file in the recent file menu could not be found.

Changes for Release 406

uM-FPU64 IDE Release 406 adds several new features and fixes some known problems.

Firmware Upgrade

To use uM-FPU64 IDE r406 software, the uM-FPU64 chip must be running firmware release 404 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 V404.dat*
44-pin chip: *uMFPU64 64K44 Firmware V404.dat*

New Features

- support has been added for the new LU decomposition operations added to the `MOP` instruction in uM-FPU64 firmware release V404.
- support has been added for the new Cholesky decomposition operations added to the `MOP` instruction in uM-FPU64 firmware release V404.
- tab processing has been added to the source window. Tabs are automatically converted to spaces.

- string can now contain the following special characters:

\r	carriage return
\n	linefeed
\t	horizontal tab
\v	vertical tab

- expanded assembler data directives to include the following:

#byte	8-bit data values
#word	16-bit integer values
#long, #long32	32-bit integer values
#long64	64-bit integer values
#float, #float32	32-bit floating point values
#double, #float64	64-bit floating point values

- assembler data directives now accept multiple comma-separated values
- auto-indent has been added to the source window.
- most recent file is now opened automatically at start-up.
- a *Program* button has been added to the source window.
- the *Program* button, *Program Functions* button, and *Program Functions* menu item now compile the source code before programming Flash memory.
- Matrix Display Window has been added.

Changes

- changed *RAM Window*, *Number Converter, SEROUT* and *SERIN* window properties, to allow other windows to be brought to the front
- Improved the update speed for RAM Window
- the status bar in the main window now shows a programming status message
- the *Program Status* dialog now exits automatically if programming is successful
- changed clear screen character for SEROUT windows to vertical tab (0x0B, \v)
- added total function size in bytes to Function window and removed 'bytes' suffix on each row
- allow pointer for address register in `DEVIO,WRITE_MEMR` and `DEVIO,READ_MEMR` instructions
- improved code optimization for register arrays
- built-in target definitions replaced by target definition files

Bug Fixes

- `IF` statements that used expressions could branch to the wrong case. This was caused by the status byte being changed by the `RIGHT` instruction. The uM-FPU64 r404 firmware fixes this problem. The `RIGHT` instruction now leaves the status byte unchanged.
- Fixed problems with displaying the cursor and breakpoints that could occasionally occur after expanding and collapsing assembler code

- Fixed a problem with the debug window that caused the original `FCALL` trace to be repeated when single stepping into a function with source level debugging enabled.
- fixed problem with cursor disappearing from the Source-level Debug Display if the cursor was on a breakpoint inside an expanded source line when the source line was collapsed.
- reimplemented the *Program Status* dialog to improve progress display and avoid a problem causing the dialog to sometimes be hidden behind the main window on completion of Flash programming
- fixed problems with code generation for 64-bit arrays
- fixed problem with code optimizer if same matrix element is on left and right side of equation
- fixed problem with 64-bit constant expressions
- fixed update of function list after programming or clearing Flash memory

Tab Processing and Auto-Indent

Tab processing has been added to the source window to make entering code easier and to improve the layout of source files. All tab characters are now replaced by one or more spaces.

Automatic Tab Replacement

When a source file is opened by the IDE, or text is pasted into the source window, all tab characters are replaced by spaces to approximate the old tab settings. Saved files will no longer contain tab characters.

Tab Processing

When a *tab* key is pressed in the source window, the following actions now occur:

Tab with No Selection

If the line immediately above the current line has a space in the same position, spaces will be inserted into the current line until the first non-space character in the line above. This makes it easy to line up the columns text such as definitions or comments. If the line immediately above the current line has a non-space character in the same position, then spaces will be added until the next tab stop. The tab stop for the first 20 characters of a line is two, and the tab stop after 20 characters is four. This makes it easy to indent code, but saves typing later in the line when tabbing to a particular column.

Tab with Text Selection

An indent is inserted by adding two spaces to the start of all lines covered by the text selection. The text selection remains in place.

Shift-Tab

An indent is removed by deleting up to two spaces from the start of the current line, and if text is selected, from all lines covered by the text selection. The text selection remains in place.

Delete

If a delete character is entered immediately after a tab or auto-indent, the last tab stop will be deleted.

Auto-Indent

If a *return* key or *shift-return* key is entered at the end of a source code line, the following actions occur:

Return

If a directive or control statements is detected on the current line, the next line will be indented by two additional spaces, otherwise the next line will have the same indent as the current line. The recognized directives or control statements are as follows:

```
#function
#asm
do
while
for
```

```
if...then
else
elseif
select
case
```

Shift-Return

A *shift-return* key causes the same action as the *return* key, but also appends the matching end statement. The cursor is positioned on the next line. The matching end statement are as follows:

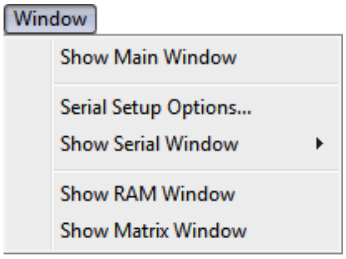
```
#function
#end

#asm
#endasm
```

```
do
loop

while
loop

for
next
```

```
if...then
endif

select
case
endselect
```

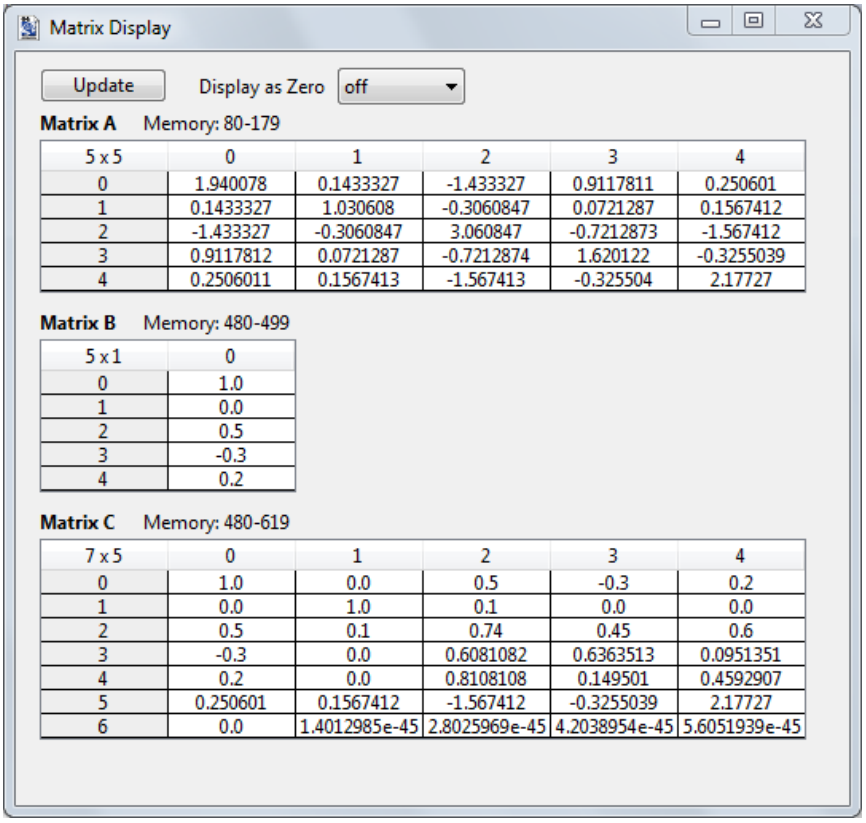
Matrix Display Window

Window Menu



Show Matrix Window

The **Matrix Display** window is brought to the front. This window is used to view the contents of matrix A, matrix B, and matrix C. The matrix values are not updated automatically, they must be updated manually using the *Update* button.



Matrix A — Memory: 80-179

5 x 5	0	1	2	3	4
0	1.940078	0.1433327	-1.433327	0.9117811	0.250601
1	0.1433327	1.030608	-0.3060847	0.0721287	0.1567412
2	-1.433327	-0.3060847	3.060847	-0.7212873	-1.567412
3	0.9117812	0.0721287	-0.7212874	1.620122	-0.3255039
4	0.2506011	0.1567413	-1.567413	-0.325504	2.17727

Matrix B — Memory: 480-499

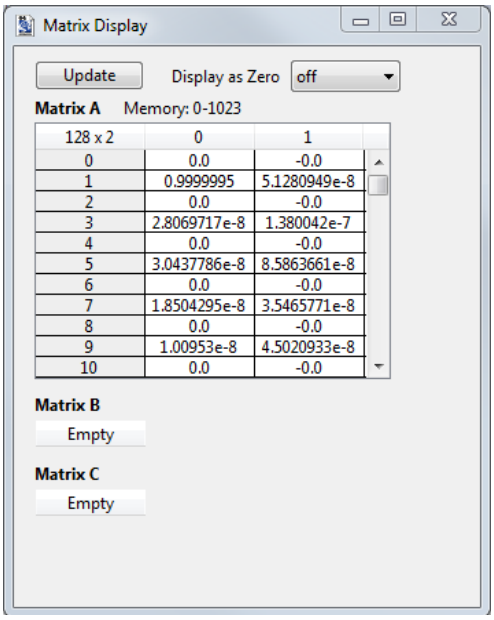
5 x 1	0
0	1.0
1	0.0
2	0.5
3	-0.3
4	0.2

Matrix C — Memory: 480-619

7 x 5	0	1	2	3	4
0	1.0	0.0	0.5	-0.3	0.2
1	0.0	1.0	0.1	0.0	0.0
2	0.5	0.1	0.74	0.45	0.6
3	-0.3	0.0	0.6081082	0.6363513	0.0951351
4	0.2	0.0	0.8108108	0.149501	0.4592907
5	0.250601	0.1567412	-1.567412	-0.3255039	2.17727
6	0.0	1.4012985e-45	2.8025969e-45	4.2038954e-45	5.6051939e-45

The *Display as Zero* option can be used to display values that are close to zero as ~0.0. A zero comparison value from 1e-1 to 1e-15 can be selected from the pop-up menu. If the absolute value of the matrix element is less than the zero comparison value, the value is displayed as ~0.0.

Display as Zero: off



Display as Zero: < 1e-6