

# What's New In HiTOP 5.30?

## 1. Windows Vista Support

HiTOP 5.30 now offers full support for Microsoft Windows Vista.

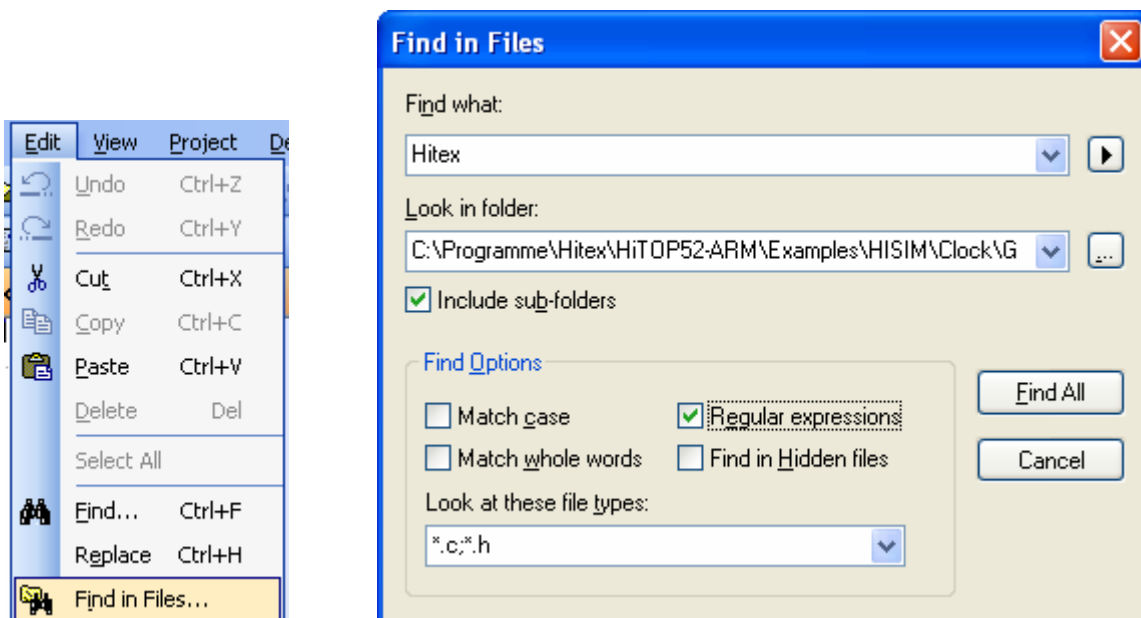
## 2. HiTOPOpen

With HiTOPOpen a standardized interface is offered that allows 3rd parties and customers to develop and offer SW components that directly interact with HiTOP. This interface is used for example for the graphical trace viewer in the Serial Wire Viewer extension for Cortex.

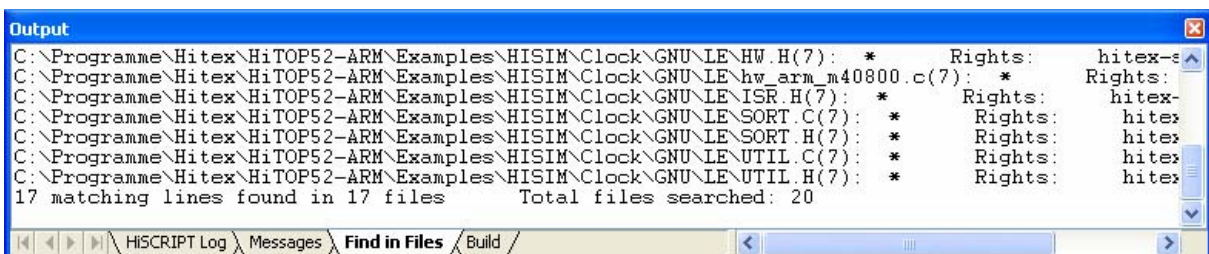
## 3. Find in Files

A new powerful search feature (Edit > Find in Files) that allows to search for data through folders or text is now part of the HiTOP IDE.

Toolbar Button: 



The search result is visualized in the output window:



A double-click activates the corresponding file in the source/editor window and positions on the found text.

## 4. New Editor

The editor on the HiTOP IDE has been replaced by a new and powerful one. In the debug mode, the display only shows minor changes, e.g. breakpoint and code markers look slightly different:

```

static void date_cpy (void)
{
    date.mem_date[0u] = '0' + date.day / 10u;
    date.mem_date[1u] = '0' + date.day % 10u;
    date.mem_date[2u] = ':';
    date.mem_date[3u] = '0' + date.month / 10u;
    date.mem_date[4u] = '0' + date.month % 10u;
    date.mem_date[5u] = ':';
    date.mem_date[6u] = '0' + date.year / 1000u;
    date.mem_date[7u] = '0' + (date.year / 100u) % 10u;
}
    
```

The new marker shows a line that has been selected via the „Find in Files“ or „Build“-Tab function.

### Collapse / Expand of statements blocks or multi-line comments:

```

while (lu != 0u)
{
    ValInt32 = func(GlobalCounter);

    if (nv != 0u)
    {
        never_exec();           /* function will be never execute */
    }
    if (GlobalCounter > 0u)
    {
        inc_date(GlobalCounter); /* function which counts days, months */
                                /* and years. This increment is not */
                                /* related to the Clock increment */
        lnk_lst();              /* linked lists */
    }
}
    
```

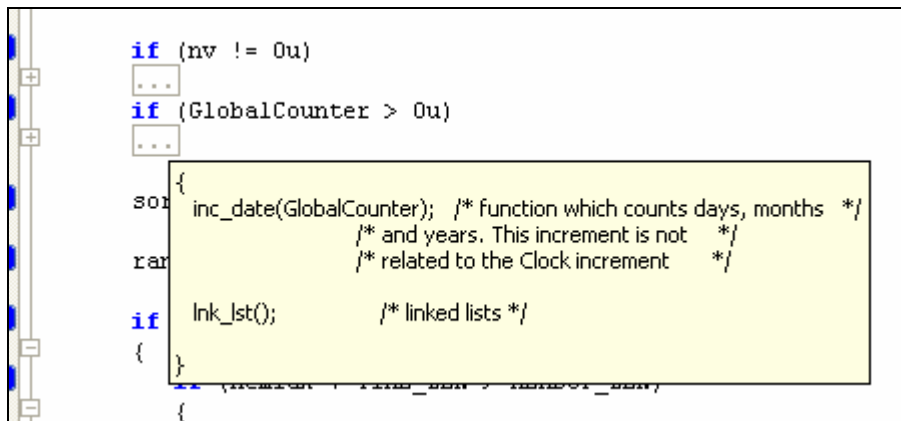
With click on the [-] mark the block can be collapsed:

```

while (lu != 0u)
{
    ValInt32 = func(GlobalCounter);

    if (nv != 0u)
    ...
    if (GlobalCounter > 0u)
    ...
}
    
```

Moving the mouse of a [...] block, the hidden content is displayed in the tool-tip:

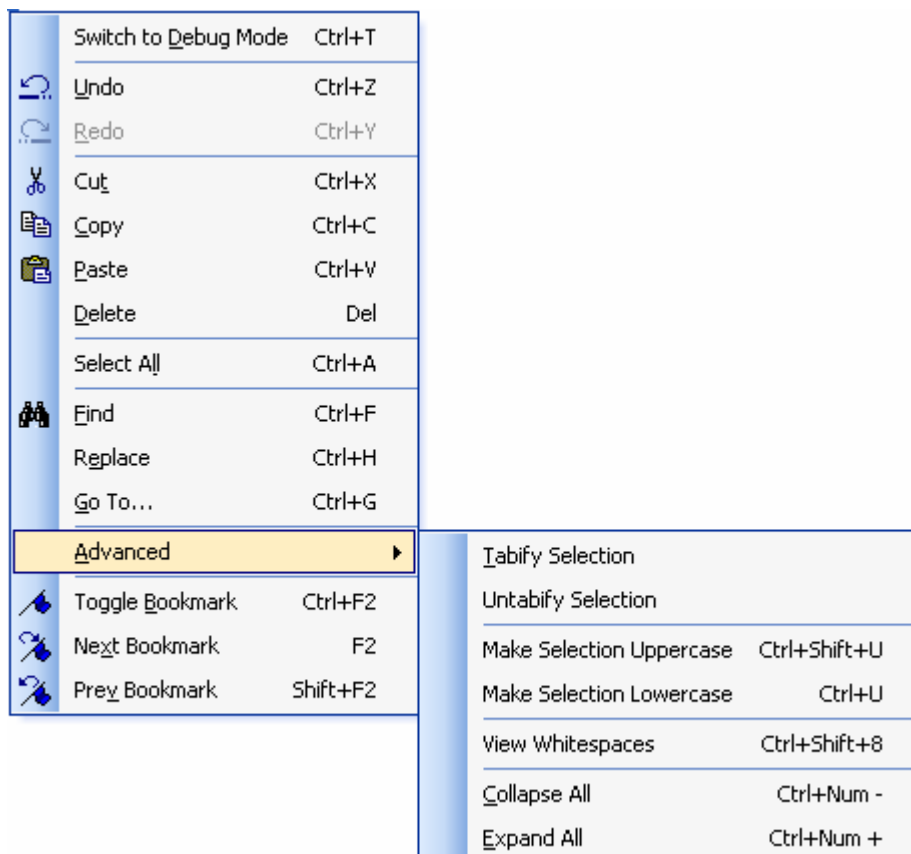


In addition, the context menu has two new commands: Collapse/Expand All

Switching into edit mode, a wealth of new features is available. In edit mode the debug-specific markings in the left columns are no longer visible and in addition, the edit mode is

shown in the status line: Debug Ln 50, Col 2 Edit Ln 50, Col 2

The new context menu for the edit mode offers:



Following are details on some of the features:

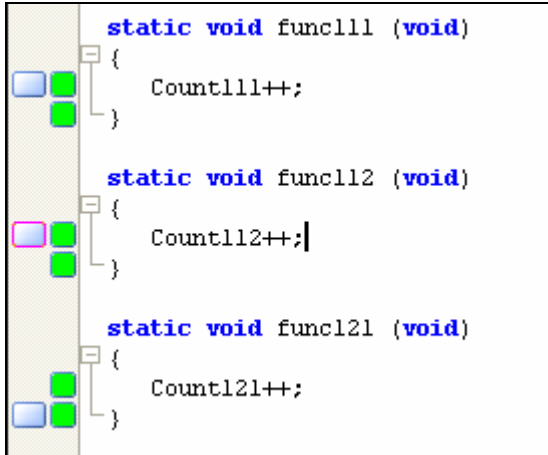
### Toggle/Next/Previous Bookmark

Bookmarks can be set, deleted and you can move between the bookmarks in the active file.

These features are also available from the toolbar:



The bookmarks are visible as shown below (also in debug mode):



### Tabify/Untabify Selection

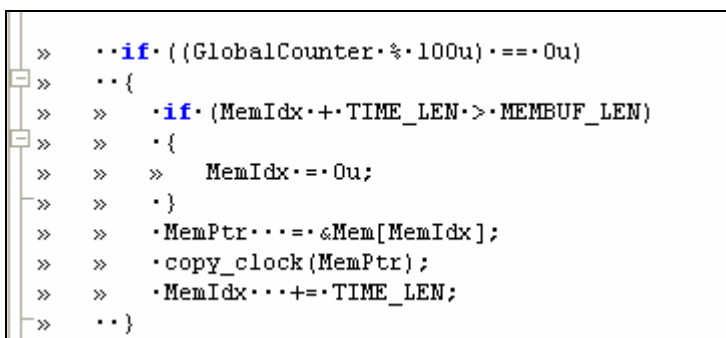
The blanks in a selected text are changed into tabs or vice versa.

### Make Selection Upper/Lower Case

A selected text will be switched from lower to upper case or vice versa.

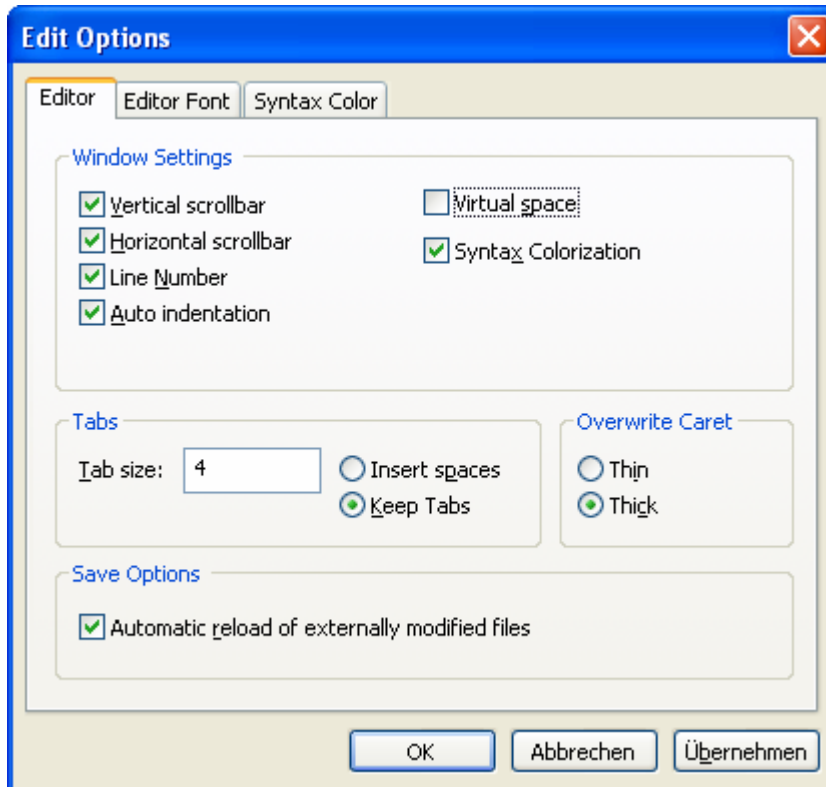
### View White Spaces

If this feature is turned on blanks and tabs are made visible:



## Editor Options

The editor now features a lot more control settings:



**Vertical/Horizontal Scrollbar** (Activate/deactivate the scroll bar(s))

**Line Number** (A display of the line numbers can be turned on: ):

```

73 void main (void)
74 {
75     static uint8_t Mem[MEMBUF_LEN];
76     static uint8_t *MemPtr;
77     static uint8_t MemIdx;
78
79     static uint16_t GlobalCounter;
80
81     static uint8_t ValInt8;
82     static uint16_t ValInt16;
83     static uint32_t ValInt32;
84
85     MemIdx = 0u;
86     GlobalCounter = 0u;
87     nv = 0u;
88     ran = 12345u;
89
90     init_hardware();
91     init_isr();
92     init_clock();
93     int_enable();

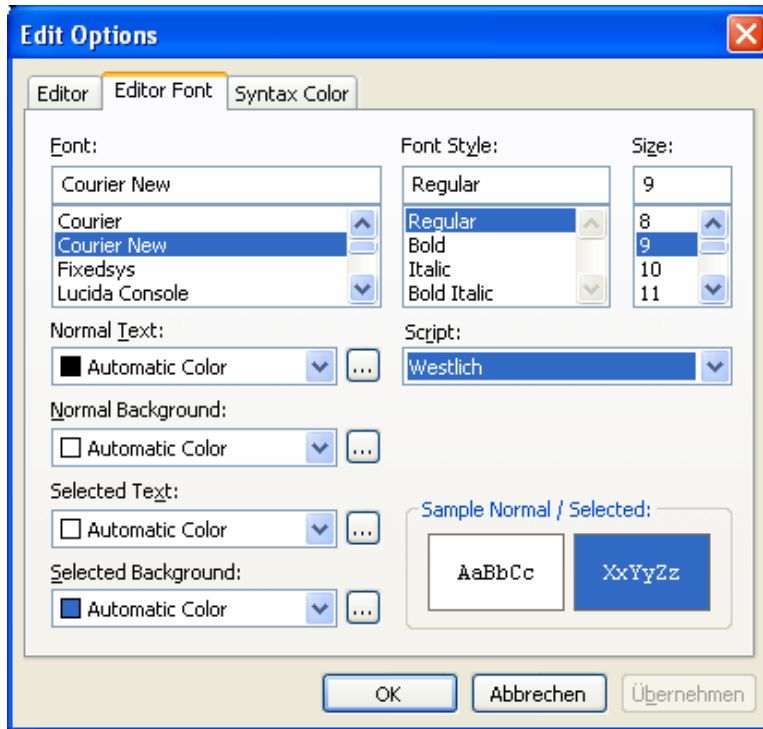
```



And many more helpful editor settings such as:

- Auto Indentation
- Virtual space (allows the cursor to be positioned anywhere, even behind the end of the line),
- Insert spaces (use tabs or replace with blanks)
- Overwrite caret (define the cursor display in overwrite mode)
- Syntax colorization (global enabling of the syntax highlighting)

The editor font tab offers the possibility to define the font, font size and front and background colors:



And the syntax color tab allows specifying the details on the syntax highlighting.

