

# 1 Usage of PicoBlaze memory update scripts

Script `'mem_update.bash'` can be use for update both program and data memory in bitstream file. Usage:

```
.\'mem_update.bash' -b BIT_NAME [-m MEM_NAME -r RAM_MODULE_NAME -i IMPACT_SCRIPT -p -d]
```

Options:

- `-b` (required) bitstream file name (with extension and full or relative path)
- `-m` File containing translated picoBlaze code or data mem file
- `-r` Ram module name (default=`'ram_1024_x_18'`)
- `-d` Replace data memory (use ram module name=`'ram_2048_x_9'` if option `-r` is not used)
- `-i` Source impact script for different board (default Spartan-3AN Design Kit)
- `-p` Program FPGA
- `-h` Help message

Typical usage:

- Replacing program memory in bitstream file with default ram module name  
`.\'mem_update.bash' -b BIT_NAME -m MEM_NAME`
- Replacing program memory in bitstream file with default ram module name and writting bitstream into FPGA  
`.\'mem_update.bash' -b BIT_NAME -m MEM_NAME -p`
- Writting bitstream file into FPGA without any changes  
`.\'mem_update.bash' -b BIT_NAME -p`
- Replacing program memory in bitstream file with custom ram module name  
`.\'mem_update.bash' -b BIT_NAME -m MEM_NAME -r RAM_MODULE_NAME`
- Writting bitstream file into FPGA without any changes with custom impact script  
`.\'mem_update.bash' -b BIT_NAME -i IMPACT_SCRIPT -p`
- Replacing data memory in bitstream file with default ram module name  
`.\'mem_update.bash' -b BIT_NAME -m MEM_NAME -d`
- Replacing data memory in bitstream file with default ram module name and writting bitstream into FPGA  
`.\'mem_update.bash' -b BIT_NAME -m MEM_NAME -d -p`

Note: Both data and program memory cannot be replaced in one step. This operation have to be split into two: replacing data memory without programming FPGA and then replacing program memory with programming option:

```
.\'mem_update.bash' -b BIT_NAME -m MEM_NAME -d  
    where MEM_NAME is name of mem file containing data memory  
.\'mem_update.bash' -b BIT_NAME -m MEM_NAME -p  
    where MEM_NAME is name of mem file containing program memory
```

Script `'text_mem.py'` translate data source file into mem file required by `'mem_update.bash'` script and psm file containing adress declaration for picoBlaze assembler. Usage:

```
.\'text_mem.py' -n SOURCE_FILE [-b -h]
```

Options:

- `-n` (required) source file name
- `-b` Big Endian byte order for words (default Little Endian)
- `-h` Help message

Source file format:

```
var_name:x ram_content
```

where `var_name` will be used as identifier for adress declaration in psm file and data type declared by `:x` as follows:

- `:s` string - each char from `ram_content` will be convert to ascii code. Extra zero byte will be added at the end of the string.
- `:b` byte - `ram_content` should contain decimal, hexadecimal or binary values separated by spaces. Leftmost data will be written at lowest address. Data entry length will be calculated and declared into psm file with `var_name_length` name.
- `:w` word - `ram_content` should contain decimal, hexadecimal or binary values separated by spaces. Leftmost data will be written at two lowest addresses depending on choosed byte order (Little or Big Endian). Data entry length will be calculated and declared into psm file with `var_name_length` name.

Data in `ram_content` format for byte or word data type:

- digits without any prefix for decimal value (e.g. 1000)
- `$` prefix for hexadecimal value (e.g. `$F2`)
- `%` prefix for binary value (e.g. `%0010`)

Values in `ram_content` should be in range of declared data type: 0-255 for byte and 0-65535 for word.

Lines started by semicolon `';'` will be treated as comment and written into psm file. Any empty line or line without colon will be ommited.

Example source file:

```
;this is string (ABCDE to 5 bytes ascii + 00 byte as eos)
string:s ABCDE
;this values will be storage as byte
byte:b 01 $F %001
;this values will be storage as word
word:w 02 $1FF %100000000
```