

---

# **gen\_stm32 Documentation**

*Release*

*[https://github.com/vroncevic/gen\\_stm32/releases](https://github.com/vroncevic/gen_stm32/releases)*

**Vladimir Roncevic <elektron.ronca@gmail.com>**

**May 28, 2021**



---

## Contents:

---

<b>1</b>	<b>gen_stm32</b>	<b>3</b>
1.1	gen_stm32 package . . . . .	3
1.1.1	Subpackages . . . . .	3
1.1.1.1	gen_stm32.pro package . . . . .	3
1.1.2	Module contents . . . . .	9
<b>2</b>	<b>Installation</b>	<b>11</b>
<b>3</b>	<b>Dependencies</b>	<b>13</b>
<b>4</b>	<b>Generation flow of project setup</b>	<b>15</b>
<b>5</b>	<b>Tool structure</b>	<b>17</b>
<b>6</b>	<b>Copyright and licence</b>	<b>21</b>
<b>7</b>	<b>Indices and tables</b>	<b>23</b>
	<b>Python Module Index</b>	<b>25</b>
	<b>Index</b>	<b>27</b>



**gen\_stm32** is toolset for generation STM32 project skeleton for development of embedded applications.

Developed in `python` code: **100%**.

The README is used to introduce the tool and provide instructions on how to install the tool, any machine dependencies it may have and any other information that should be provided before the tool is installed.



## 1.1 gen\_stm32 package

### 1.1.1 Subpackages

#### 1.1.1.1 gen\_stm32.pro package

##### Subpackages

##### gen\_stm32.pro.config package

##### Submodules

##### gen\_stm32.pro.config.pro\_name module

**Module** pro\_name.py

**Copyright** Copyright (C) 2018 Vladimir Roncevic <elektron.ronca@gmail.com> gen\_stm8 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen\_stm8 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

**Info** Defined class ProName with attribute(s) and method(s). Defined API for project name with preparations for generation.

**class** gen\_stm32.pro.config.pro\_name.ProName (*verbose=False*)

Defined class ProName with attribute(s) and method(s). Defined API for project name with preparations for generation. It defines:

#### attributes

GEN\_VERBOSE - console text indicator for process-phase.  
\_\_verbose - enable/disable verbose option.  
\_\_pro\_name - project name.

#### methods

\_\_init\_\_ - initial constructor.  
pro\_name - property methods for set/get operations.  
is\_pro\_name\_ok - checking is project name ok.  
\_\_str\_\_ - dunder method for ProName.

```
GEN_VERBOSE = 'GEN_STM32::PRO::CONFIG::PRO_NAME'
```

```
is_pro_name_ok()
```

Checking is project name ok.

**Returns** boolean status, True (not None) | False.

**Return type** <bool>

**Exceptions** None

#### pro\_name

Property method for getting project name.

**Returns** formatted project name | None.

**Return type** <str> | <NoneType>

**Exceptions** None

## gen\_stm32.pro.config.template\_dir module

**Module** template\_dir.py

**Copyright** Copyright (C) 2018 Vladimir Roncevic <[elektron.ronca@gmail.com](mailto:elektron.ronca@gmail.com)> gen\_stm8 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen\_stm8 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

**Info** Defined class TemplateDir with attribute(s) and method(s). Defined API for template directory for generation.

```
class gen_stm32.pro.config.template_dir.TemplateDir(verbose=False)
```

Defined class TemplateDir with attribute(s) and method(s). Defined API for template directory for generation.  
It defines:

#### attributes

GEN\_VERBOSE - console text indicator for process-phase.  
\_\_verbose - enable/disable verbose option.  
\_\_template\_dir - project template dir.

#### methods

\_\_init\_\_ - initial constructor.  
template\_dir - property methods for set/get operations.



is\_template\_dir\_ok - checking is template dir ok.

\_\_str\_\_ - dunder method for TemplateDir.

```
GEN_VERBOSE = 'GEN_STM32::PRO::CONFIG::TEMPLATE_DIR'
```

```
is_template_dir_ok()
```

Checking is project template dir ok.

**Returns** boolean status, True (not None) | False.

**Return type** <bool>

**Exceptions** None

```
template_dir
```

Property method for getting template dir.

**Returns** formatted template dir | None.

**Return type** <str> | <NoneType>

**Exceptions** None

## Module contents

**Module** \_\_init\_\_.py

**Copyright** Copyright (C) 2018 Vladimir Roncevic <[elektron.ronca@gmail.com](mailto:elektron.ronca@gmail.com)> gen\_stm8 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen\_stm8 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

**Info** Defined class ProConfig with attribute(s) and method(s). Defined project configuration container.

```
class gen_stm32.pro.config.ProConfig(verbose=False)
```

Defined class ProConfig with attribute(s) and method(s). Defined project configuration container. It defines:

### attributes

GEN\_VERBOSE - console text indicator for process-phase.

TEMPLATES - templates key.

MODULES - modules key.

FORMAT - format for template file.

\_\_verbose - enable/disable verbose option.

\_\_config - configuration dictionary.

### methods

\_\_init\_\_ - initial constructor.

pro\_name - property methods for set/get operations.

is\_config\_ok - checking is project configuration ok.

\_\_str\_\_ - dunder method for ProConfig.

```
FORMAT = 'template'
```

```
GEN_VERBOSE = 'GEN_STM32::PRO::CONFIG::PRO_CONFIG'
```

```
MODULES = 'modules'
```

```
TEMPLATES = 'templates'
```

#### **config**

Property method for getting project configuration.

**Returns** formatted project configuration | None.

**Return type** <dict> | <NoneType>

**Exceptions** None

#### **is\_config\_ok()**

Checking is project configuration ok.

**Returns** boolean status, True (ok) | False.

**Return type** <bool>

**Exceptions** None

## Submodules

### gen\_stm32.pro.read\_template module

**Module** read\_template.py

**Copyright** Copyright (C) 2018 Vladimir Roncevic <[elektron.ronca@gmail.com](mailto:elektron.ronca@gmail.com)> gen\_stm32 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen\_stm32 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

**Info** Define class ReadTemplate with attribute(s) and method(s). Created API for read a templates and return a string representations.

**class** gen\_stm32.pro.read\_template.**ReadTemplate** (*verbose=False*)

Bases: *ats\_utilities.config\_io.base\_check.FileChecking*, *gen\_stm32.pro.config.template\_dir.TemplateDir*

Define class ReadTemplate with attribute(s) and method(s). Created API for read a templates and return a string representations. It defines:

#### **attributes**

GEN\_VERBOSE - console text indicator for process-phase.

TEMPLATE\_DIR - prefix path to templates.

#### **methods**

`__init__` - initial constructor.

`read` - read a templates and return a content.

`__str__` - dunder method for ReadTemplate.

```
GEN_VERBOSE = 'GEN_STM32::PRO::READ_TEMPLATE'
```

```
TEMPLATE_DIR = '/../conf/template/'
```

```
VERBOSE = 'ATS_UTILITIES'
```

**read** (*config*, *verbose=False*)

Read a templates and return a content.

**Parameters**

- **config** (*<dict>*) – parameter file name.
- **verbose** (*<bool>*) – enable/disable verbose option.

**Returns** template content list | empty list.

**Return type** *<list>*

**Exceptions** ATSTypeError | ATSBadCallError

## gen\_stm32.pro.write\_template module

**Module** write\_template.py

**Copyright** Copyright (C) 2018 Vladimir Roncevic <[elektron.ronca@gmail.com](mailto:elektron.ronca@gmail.com)> gen\_stm32 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen\_stm32 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

**Info** Define class WriteTemplate with attribute(s) and method(s). Created API for writing template content with parameters.

**class** gen\_stm32.pro.write\_template.**WriteTemplate** (*verbose=False*)

Bases: ats\_utilities.config\_io.base\_check.FileChecking

Define class WriteTemplate with attribute(s) and method(s). Created API for writing template content with parameters. It defines:

**attributes**

GEN\_VERBOSE - console text indicator for process-phase.

**methods**

`__init__` - initial constructor.

`write` - write a templates with parameters.

`__str__` - dunder method for WriteTemplate.

`GEN_VERBOSE` = 'GEN\_STM32::PRO::WRITE\_TEMPLATE'

`VERBOSE` = 'ATS\_UTILITIES'

**write** (*templates*, *project\_name*, *verbose=False*)

Write a templates with parameters.

**Parameters**

- **templates** (*<list>*) – parameter templates.
- **project\_name** (*<str>*) – parameter project name.
- **verbose** (*<bool>*) – enable/disable verbose option.

**Returns** boolean status, True (success) | False.

**Return type** *<bool>*

**Exceptions** ATSTypeError | ATSBadCallError

## Module contents

**Module** `__init__.py`

**Copyright** Copyright (C) 2018 Vladimir Roncevic <[elektron.ronca@gmail.com](mailto:elektron.ronca@gmail.com)> gen\_stm32 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen\_stm32 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

**Info** Defined class STM32Setup with attribute(s) and method(s). Generate project content with parameters from a project setup.

**class** `gen_stm32.pro.STM32Setup` (*verbose=False*)  
Bases: `ats_utilities.config_io.base_check.FileChecking`, `gen_stm32.pro.config.ProConfig`, `gen_stm32.pro.config.pro_name.ProName`

Defined class STM32Setup with attribute(s) and method(s). Generate project content with parameters from a project setup. It defines:

### attributes

`GEN_VERBOSE` - console text indicator for process-phase.

`PRO_STRUCTURE` - project structure.

`__reader` - reader API.

`__writer` - writer API.

### methods

`__init__` - initial constructor.

`get_reader` - getter for reader object.

`get_writer` - getter for writer object.

`gen_pro_setup` - generate STM32 project structure.

`__str__` - dunder method for STM32Setup.

`PRO_STRUCTURE = '/../conf/project.yaml'`

`VERBOSE = 'ATS_UTILITIES'`

`gen_pro_setup` (*project\_name*, *verbose=False*)

Generate project structure.

### Parameters

- **project\_name** (<str>) – parameter tool name.
- **verbose** (<bool>) – enable/disable verbose option.

**Returns** boolean status, True (success) | False.

**Return type** <bool>

**Exceptions** None

`get_reader` ()

Getter for reader object. :return: Read template object :rtype: <ReadTemplate> :exceptions: None

`get_writer()`

Getter for writer object. :return: Write template object :rtype: <WriteTemplate> :exceptions: None

## 1.1.2 Module contents

Module `__init__.py`

**Copyright** Copyright (C) 2018 Vladimir Roncevic <[elektron.ronca@gmail.com](mailto:elektron.ronca@gmail.com)> gen\_stm32 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen\_stm32 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <<http://www.gnu.org/licenses/>>.

**Info** Defined class GenSTM32 with attribute(s) and method(s). Load a base info, create an CLI interface and run operation(s).

**class** `gen_stm32.GenSTM32` (*verbose=False*)

Bases: `ats_utilities.cli.cfg_cli.CfgCLI`

Defined class GenSTM32 with attribute(s) and method(s). Load a base info, create an CLI interface and run operation(s). It defines:

### attributes

`__metaclass__` - setting cooperative metaclasses.  
`GEN_VERBOSE` - console text indicator for process-phase.  
`CONFIG` - configuration file path.  
`LOG` - tool log file path.  
`OPS` - list of tool options.  
`logger` - logger object API.

### methods

`__init__` - initial constructor.  
`process` - process and run tool option(s).  
`__str__` - dunder method for GenSTM32.

`CONFIG = '/conf/gen_stm32.cfg'`

`GEN_VERBOSE = 'GEN_STM32'`

`LOG = '/log/gen_stm32.log'`

`OPS = ['-g', '--gen', '-v', '--verbose', '--version']`

`VERBOSE = 'ATS_UTILITIES'`

**process** (*verbose=False*)

Process and run operation.

**Parameters** `verbose` (<bool>) – enable/disable verbose option.

**Returns** boolean status, True (success) | False.

**Return type** <bool>

**Exceptions** None



## CHAPTER 2

---

### Installation

---

Navigate to release [page](#) download and extract release archive.

To install **gen\_stm32** type the following:

```
tar xvzf gen_stm32-x.y.z.tar.gz
cd gen_stm32-x.y.z/
# python2
pip install -r requirements.txt
python setup.py install_lib
python setup.py install_data
python setup.py install_egg_info
# python3
pip3 install -r requirements.txt
python3 setup.py install_lib
python3 setup.py install_data
python3 setup.py install_egg_info
```

You can use Docker to create image/container, or You can use pip to install:

```
# python2
pip install gen-stm32
# python3
pip3 install gen-stm32
```





## CHAPTER 3

---

### Dependencies

---

**gen\_stm32** requires next modules and libraries:

- [ats-utilities](#) - Python App/Tool/Script Utilities



## CHAPTER 4

---

### Generation flow of project setup

---

Base flow of generation process:



**gen\_stm32** is based on Template mechanism:

Code structure:

```
gen_stm32/
├── conf/
│   ├── gen_stm32.cfg
│   ├── gen_stm32_util.cfg
│   ├── project.yaml
│   └── template/
│       ├── build/
│       │   ├── includes/
│       │   │   └── STM32F4xx_StdPeriph_Driver/
│       │   │       └── src/
│       │   │           └── subdir.template
│       │   ├── Makefile.template
│       │   ├── objects.template
│       │   ├── source/
│       │   │   └── subdir.template
│       │   └── sources.template
│       └── includes/
│           ├── CMSIS/
│           │   ├── arm_common_tables.template
│           │   ├── arm_math.template
│           │   ├── core_cm0.template
│           │   ├── core_cm3.template
│           │   ├── core_cm4_simd.template
│           │   ├── core_cm4.template
│           │   ├── core_cmFunc.template
│           │   └── core_cmInstr.template
│           └── STM32F4xx/
│               ├── stm32f4xx_conf.template
│               ├── stm32f4xx.template
│               └── system_stm32f4xx.template
│           └── STM32F4xx_StdPeriph_Driver/
```

(continues on next page)

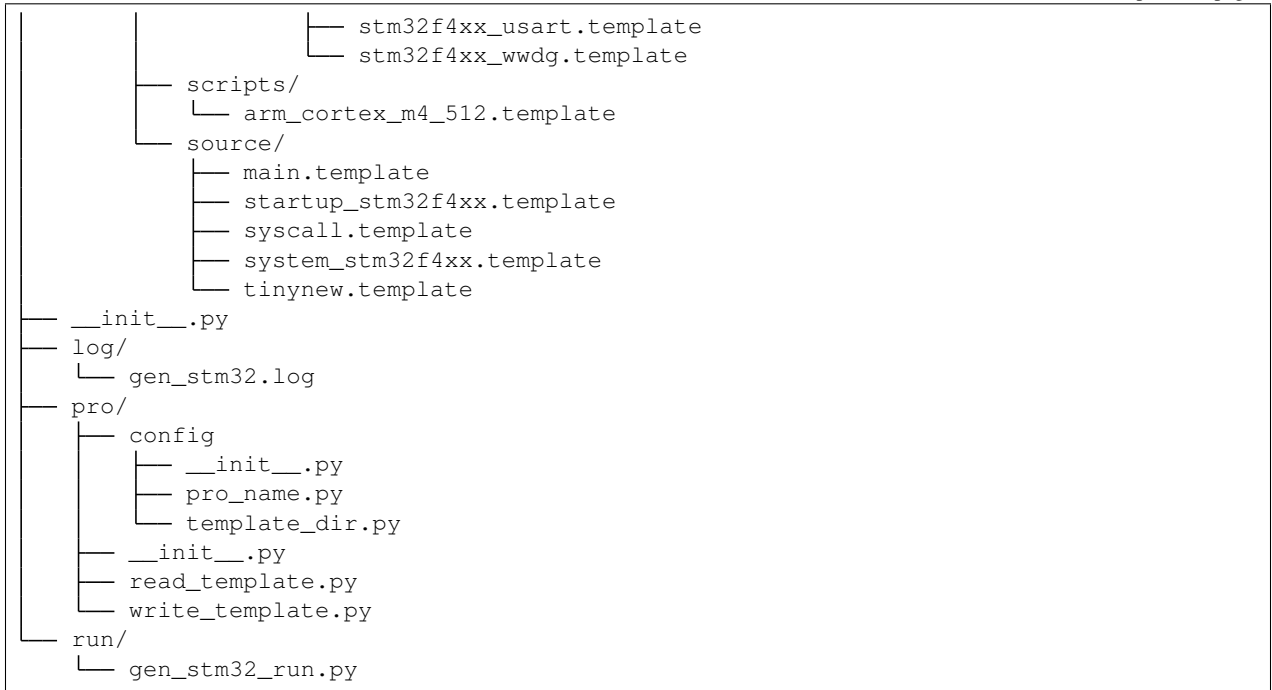
(continued from previous page)

```
inc/
├── misc.template
├── stm32f4xx_adc.template
├── stm32f4xx_can.template
├── stm32f4xx_crc.template
├── stm32f4xx_cryp.template
├── stm32f4xx_dac.template
├── stm32f4xx_dbgmcu.template
├── stm32f4xx_dcmi.template
├── stm32f4xx_dma.template
├── stm32f4xx_exti.template
├── stm32f4xx_flash.template
├── stm32f4xx_fsmc.template
├── stm32f4xx_gpio.template
├── stm32f4xx_hash.template
├── stm32f4xx_i2c.template
├── stm32f4xx_iwdg.template
├── stm32f4xx_pwr.template
├── stm32f4xx_rcc.template
├── stm32f4xx_rng.template
├── stm32f4xx_rtc.template
├── stm32f4xx_sdio.template
├── stm32f4xx_spi.template
├── stm32f4xx_syscfg.template
├── stm32f4xx_tim.template
├── stm32f4xx_usart.template
└── stm32f4xx_wwdg.template

src/
├── misc.template
├── stm32f4xx_adc.template
├── stm32f4xx_can.template
├── stm32f4xx_crc.template
├── stm32f4xx_cryp_aes.template
├── stm32f4xx_cryp_des.template
├── stm32f4xx_cryp_tdes.template
├── stm32f4xx_cryp.template
├── stm32f4xx_dac.template
├── stm32f4xx_dbgmcu.template
├── stm32f4xx_dcmi.template
├── stm32f4xx_dma.template
├── stm32f4xx_exti.template
├── stm32f4xx_flash.template
├── stm32f4xx_fsmc.template
├── stm32f4xx_gpio.template
├── stm32f4xx_hash_md5.template
├── stm32f4xx_hash_shal.template
├── stm32f4xx_hash.template
├── stm32f4xx_i2c.template
├── stm32f4xx_iwdg.template
├── stm32f4xx_pwr.template
├── stm32f4xx_rcc.template
├── stm32f4xx_rng.template
├── stm32f4xx_rtc.template
├── stm32f4xx_sdio.template
├── stm32f4xx_spi.template
├── stm32f4xx_syscfg.template
└── stm32f4xx_tim.template
```

(continues on next page)

(continued from previous page)







---

### Copyright and licence

---

Copyright (C) 2018 by [vrontcevic.github.io/gen\\_stm32](https://vrontcevic.github.io/gen_stm32)

**gen\_stm32** is free software; you can redistribute it and/or modify it under the same terms as Python itself, either Python version 2.x/3.x or, at your option, any later version of Python 3 you may have available.

Lets help and support PSF.





# CHAPTER 7

---

## Indices and tables

---

- `genindex`
- `modindex`
- `search`



**g**

gen\_stm32, 9  
gen\_stm32.pro, 8  
gen\_stm32.pro.config, 5  
gen\_stm32.pro.config.pro\_name, 3  
gen\_stm32.pro.config.template\_dir, 4  
gen\_stm32.pro.read\_template, 6  
gen\_stm32.pro.write\_template, 7



**C**

CONFIG (*gen\_stm32.GenSTM32* attribute), 9  
 config (*gen\_stm32.pro.config.ProConfig* attribute), 6

**F**

FORMAT (*gen\_stm32.pro.config.ProConfig* attribute), 5

**G**

gen\_pro\_setup() (*gen\_stm32.pro.STM32Setup* method), 8  
 gen\_stm32 (module), 9  
 gen\_stm32.pro (module), 8  
 gen\_stm32.pro.config (module), 5  
 gen\_stm32.pro.config.pro\_name (module), 3  
 gen\_stm32.pro.config.template\_dir (module), 4  
 gen\_stm32.pro.read\_template (module), 6  
 gen\_stm32.pro.write\_template (module), 7  
 GEN\_VERBOSE (*gen\_stm32.GenSTM32* attribute), 9  
 GEN\_VERBOSE (*gen\_stm32.pro.config.pro\_name.ProName* attribute), 4  
 GEN\_VERBOSE (*gen\_stm32.pro.config.ProConfig* attribute), 5  
 GEN\_VERBOSE (*gen\_stm32.pro.config.template\_dir.TemplateDir* attribute), 5  
 GEN\_VERBOSE (*gen\_stm32.pro.read\_template.ReadTemplate* attribute), 6  
 GEN\_VERBOSE (*gen\_stm32.pro.write\_template.WriteTemplate* attribute), 7  
 GenSTM32 (class in *gen\_stm32*), 9  
 get\_reader() (*gen\_stm32.pro.STM32Setup* method), 8  
 get\_writer() (*gen\_stm32.pro.STM32Setup* method), 8

**I**

is\_config\_ok() (*gen\_stm32.pro.config.ProConfig* method), 6

is\_pro\_name\_ok() (*gen\_stm32.pro.config.pro\_name.ProName* method), 4

is\_template\_dir\_ok() (*gen\_stm32.pro.config.template\_dir.TemplateDir* method), 5

**L**

LOG (*gen\_stm32.GenSTM32* attribute), 9

**M**

MODULES (*gen\_stm32.pro.config.ProConfig* attribute), 5

**O**

OPS (*gen\_stm32.GenSTM32* attribute), 9

**P**

pro\_name (*gen\_stm32.pro.config.pro\_name.ProName* attribute), 4

PRO\_STRUCTURE (*gen\_stm32.pro.STM32Setup* attribute), 8

process() (*gen\_stm32.GenSTM32* method), 9

ProConfig (class in *gen\_stm32.pro.config*), 5

ProName (class in *gen\_stm32.pro.config.pro\_name*), 3

**R**

read() (*gen\_stm32.pro.read\_template.ReadTemplate* method), 6

ReadTemplate (class in *gen\_stm32.pro.read\_template*), 6

**S**

STM32Setup (class in *gen\_stm32.pro*), 8

**T**

template\_dir (*gen\_stm32.pro.config.template\_dir.TemplateDir* attribute), 5

TEMPLATE\_DIR (*gen\_stm32.pro.read\_template.ReadTemplate* attribute), 6

TemplateDir (class in  
*gen\_stm32.pro.config.template\_dir*), 4  
TEMPLATES (*gen\_stm32.pro.config.ProConfig* at-  
tribute), 5

## V

VERBOSE (*gen\_stm32.GenSTM32* attribute), 9  
VERBOSE (*gen\_stm32.pro.read\_template.ReadTemplate*  
attribute), 6  
VERBOSE (*gen\_stm32.pro.STM32Setup* attribute), 8  
VERBOSE (*gen\_stm32.pro.write\_template.WriteTemplate*  
attribute), 7

## W

write() (*gen\_stm32.pro.write\_template.WriteTemplate*  
method), 7  
WriteTemplate (class in  
*gen\_stm32.pro.write\_template*), 7