CrcEngine Documentation

Release 0.3

GardenTools

Apr 09, 2021

CONTENTS:

CHAPTER

ONE

CRCENGINE

A python library for CRC calculation providing table-based as well as bit-bashing implementations (for reference).

- Free software: GNU General Public License v3
- Documentation: https://crcengine.readthedocs.io.

1.1 Installing

CrcEngine can be installing using pip with

```
pip install crcengine
```

1.2 Usage

Pre-defined algorithms such as CRC32 are available. Tailored algorithms can be created by calling CrcEngine.create() and other related methods.

A calculation engine for a specific named algorithm can be obtained using CrcEngine.new(). Algorithms which are not pre-defined can be created using CrcEngine.create()

A list of pre-defined algorithms can be obtained using crcengine.algorithms_available()

```
>>> list(crcengine.algorithms_available())
['crc8', 'crc8-autosar', 'crc8-bluetooth', 'crc8-ccitt', 'crc8-gsm-b', 'crc8-sae-j1850
↪', 'crc15-can', 'crc16-kermit', 'crc16-ccitt-true', 'crc16-xmodem', 'crc16-autosar',
↪ 'crc16-ccitt-false', 'crc16-cdma2000', 'crc16-ibm', 'crc16-modbus', 'crc16-profibus
↪', 'crc24-flexray16-a', 'crc24-flexray16-b', 'crc32', 'crc32-bzip2', 'crc32-c',
↪'crc64-ecma']
```

1.2.1 Built-in algorithms

crc8, crc8-autosar, crc8-bluetooth, crc8-ccitt, crc8-gsm-b, crc8-sae-j1850, crc15-can, crc16-kermit, crc16-ccitt-true, crc16-xmodem, crc16-autosar, crc16-ccitt-false, crc16-cdma2000, crc16-ibm, crc16-modbus, crc16-profibus, crc24-flexray16-a, crc24-flexray16-b, crc32, crc32-bzip2, crc32-c, crc64-ecma

1.3 Examples

Using a pre-defined algorithm

```
import crcengine
crc_algorithm = crcengine.new('crc32-bzip2')
result = crc_algorithm(b'123456789')
print('CRC=0x{:08x}'.format(result))
```

Output: > CRC=0xfc891918

Defining an algorithm

```
import crcengine
crc_openpgp = crcengine.create(0x864cfb, 24, 0xb704ce, ref_in=False,
                               ref_out=False, xor_out=0,
                               name='crc-24-openpgp')
```

1.4 Code Generation

The library can generate C code for a given table-algorithm. The code produced is intended to be a reasonable compromise between size, complexity and speed without requiring allocation of memory for table generation at runtime.

Faster implementations of specific algorithms can be achieved in software which unroll loops and pipeline the operations different bytes to introduce parallelism in the calculation see intel_soft_src for example. Some processors also include instructions specifically for crc calculation.

1.4.1 Code Generation Example usage:

Generating code into a directory named "out" by passing CRC parameters

```
params = crcengine.get_algorithm_params('crc32')
crcengine.generate_code(params, 'out/')
```

or referencing the algorithm by name

```
crcengine.generate_code('crc16-xmodem', 'out/')
```

1.5 Downloading

- The source is available on github
- Git clone crcengine.git
- On pypi.org

1.6 Running the tests

Tests can be performed directly by executing pytest in the "tests" directory

1.7 Running the Codegen tests

The codegen tests make use of ceedling which is expected to be installed as a ruby gem. The unit tests are configured to compile with gcc.

With thanks to Greg Cook for providing such a thoroughly collated list of CRC definitions

CHAPTER

TWO

HISTORY

2.1 0.3.1 (2021-04-09)

- Correcting metadata for python version in setup.cfg

2.2 0.3.0 (2021-04-05)

- Fixed code generation for algorithms whose result doesn't wholly fill a data-type
- Added unit tests for generated C, run using Ceedling
- Added command-line entry point
- Added support for invoking as a module via python -m
- Switched over to using setup.cfg rather than setup.py
- Python 3.9 support added

2.3 0.2.0 (2020-01-30)

Added Sphinx documentation

2.4 0.1.1 (2019-11-19)

Addressing dependency issues when installing package in some environments

2.5 0.1.0 (2019-11-18)

- First experimental release on PyPI. Code generation support is incomplete and API is prone to change

CHAPTER

THREE

CRCENGINE

3.1 crcengine package

3.1.1 Submodules

3.1.2 crcengine.algorithms module

3.1.3 crcengine.calc module

3.1.4 crcengine.codegen module

3.1.5 crcengine.version module

3.1.6 Module contents

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search