**mrvlFlashBCH Library**

MARVELL CONFIDENTIAL

Copyright(c) 2014 Marvell Semiconductor Inc., All Rights Reserved

Shashi Kiran Chilappagari and Phong Sy Nguyen

[shashic@marvell.com](mailto:shashic@marvell.com) and [psnguyen@marvell.com](mailto:psnguyen@marvell.com)

This is the documentation for the library for Flash BCH ENDEC.

This library version is 1.0.0.

Please refer to driver\_mrvlFlashBCH.cpp for an illustration on the usage of different functions.

The library supports the following functions.

1. **mrvlCreateFlashBCH**

* This function is used to create the mrvlFlashBCH library
* Arguments: void
* Return type: int
  + 0: successful creation of library
  + 1: failure of creation of library

1. **mrvlReleaseFlashBCH**

* This function is used to release the mrvlFlashBCH library
* Arguments: void
* Return type: int
  + 0: successful release of library
  + 1: failure of release of library

1. **mrvlInitializeFlashBCH**

* This function is used to initialize the mrvlFlashBCH library.
* Arguments:
  + int \_num\_codewords: This is the number of BCH codewords in the data stream.
  + int\* \_codeword\_lengths\_bch: This is an array of integers that specifies the user lengths of BCH codewords.
  + int \_t\_bch: This is the correction power of the desired BCH code.
* Return type: int
  + 0: success
  + 1: failure

1. **mrvlGetParityLength**

* This function is to get the parity length of the current BCH code.
* Arguments: void
* Return type:
  + int \_parity\_length: This integer indicates the parity length.

1. **mrvlEncodeBCH**

* This function is used to encode user bits into BCH coded bits.
* Arguments:
  + int\* \_input: This is an array of user bits. This needs to be passed on as an input to the function.
  + int\* \_output: This is an array of coded bits. This array is filled in by the function.
* Return type: int
  + 0: success
  + 1: failure

1. **mrvlDecodeBCH**

* This function is to run the BCH decoder.
* Arguments:
  + int\* \_input: This is an array of integers that contains the input to the primary decoder. This needs to be passed on as an input to the function. For one-bit input mode, this is a binary array. For two-bit input mode, this is an array of values 0, 1, 2, or 3.
  + int\* \_output: This is an array of integers that contains the decoded bits. This array is filled in by the function.
* Return type: int
  + 0: the decoder can correct all errors
  + 1: the decoder fails to correct all errors

1. **mrvlGenerateRandomBits**

* This function is used to generate an array of pseudo-random bits.
* Arguments:
  + int\* \_random\_array: This is an array of pseudo-random binary bits. This array is filled in by the function.
  + int \_length: This is the length of the array. This needs to be passed on as an input to the function.
  + int \_seed: This is the seed used by the random generator. This needs to be passed on as an input to the function.
* Return type: void

1. **mrvlAddGaussianNoise**

* This function is used to add Additive White Gaussian Noise to an array of binary values, assuming that the BPSK modulation is used.
* Arguments:
  + int\* \_user\_data: This is an array of noiseless binary values. This needs to be passed on as an input to the function.
  + double\* \_noisy\_data: This is an array of noisy data. This array is filled in by the function.
  + int \_length: This is the length of the array. This needs to be passed on as an input to the function.
  + double \_snr: This is the signal-to-noise-ratio (SNR) at which the noisy data is generated. This needs to be passed on as an input to the function.
  + int \_noise\_seed: This is the seed used by the noise generator. This needs to be passed on as an input to the function.
* Return type: void

1. **mrvlSwapData**

* This function serves as a utility to swap bits in a byte and swap bytes in a word to provide the desired endianess.
* Arguments:
  + int\* \_input: This is an array of binary data. This needs to be passed on as an input to the function.
  + int \_bit\_length: This is the length of the input data stream. This needs to be passed on as an input to the function.
  + int\* \_output: This is an array that contains the swapped binary data. This array is filled in by the function.
  + int \_word\_size: This is the number of bits in a word. For example, if there are 4 bytes in a word and 8 bits in a byte then this is set as 8\*4 = 32. This needs to be passed on as an input to the function.
  + int \_byte\_size: This is the number of bits in a byte (typically 8). This needs to be passed on as an input to the function.
  + int \_byte\_swap\_flag: Set this to 1 (or 0) to enable (or disable) swapping bytes in a word.
  + int \_bit\_swap\_flag: Set this to 1 (or 0) to enable (or disable) swapping bits in a bytes.
* Return type: void

1. **mrvlFlashBCHVersion**

* This function is to get the version of the mrvlFlashBCH library
* Arguments
  + char version[100]: This is a char array in which the version of the mrvlFlashBCH is filled in by the function.
* Return type: void