QATzip

1.0.9

Generated by Doxygen 1.9.1

1 Module Index	1
1.1 Modules	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Module Documentation	7
4.1 Data Compression API	7
4.1.1 Detailed Description	8
4.1.2 Macro Definition Documentation	8
4.1.2.1 QATZIP_API_VERSION_NUM_MAJOR	9
4.1.2.2 QATZIP_API_VERSION_NUM_MINOR	9
4.1.2.3 QZ_MAX_STRING_LENGTH	9
4.1.2.4 QZ_OK	9
4.1.2.5 QZ_SKID_PAD_SZ	9
4.1.2.6 QZ_SW_BACKUP_BIT_POSITION	10
4.1.2.7 QZ_SW_EXECUTION_BIT	10
4.1.3 Typedef Documentation	11
4.1.3.1 PinMem_T	11
4.1.3.2 QzCrcPolynomial_T	11
4.1.3.3 QzCrcType_T	11
4.1.3.4 QzDataFormat T	11
4.1.3.5 QzDirection_T	11
4.1.3.6 QzHuffmanHdr T	12
4.1.3.7 qzLZ4SCallbackFn	
4.1.3.8 QzPollingMode_T	
4.1.3.9 QzSession_T	
4.1.3.10 QzSessionParams_T	
4.1.3.11 QzSoftwareComponentType T	
4.1.3.12 QzStatus_T	
4.1.3.13 QzStream_T	
4.1.4 Enumeration Type Documentation	
4.1.4.1 PinMem_E	
4.1.4.2 QzCrcPolynomial_E	
4.1.4.3 QzCrcType_E	
4.1.4.4 QzDataFormat E	
4.1.4.5 QzDirection_E	
4.1.4.6 QzHuffmanHdr_E	
4.1.4.7 QzPollingMode_E	
4.1.4.8 QzSoftwareComponentType_E	
=====:::::::::::::::::::::::::::	

	4.1.5 Function Documentation	18
	4.1.5.1 qzClose()	18
	4.1.5.2 qzCompress()	19
	4.1.5.3 qzCompressCrc()	20
	4.1.5.4 qzCompressStream()	22
	4.1.5.5 qzDecompress()	23
	4.1.5.6 qzDecompressStream()	24
	4.1.5.7 qzEndStream()	25
	4.1.5.8 qzFree()	26
	4.1.5.9 qzGetDefaults()	27
	4.1.5.10 qzGetSoftwareComponentCount()	28
	4.1.5.11 qzGetSoftwareComponentVersionList()	28
	4.1.5.12 qzGetStatus()	29
	4.1.5.13 qzlnit()	31
	4.1.5.14 qzMalloc()	32
	4.1.5.15 qzMemFindAddr()	33
	4.1.5.16 qzSetDefaults()	33
	4.1.5.17 qzSetupSession()	34
	4.1.5.18 qzTeardownSession()	35
	4.2 debug API	36
5	Class Documentation	37
•	5.1 QatThread_S Struct Reference	37
	5.2 QzSession_S Struct Reference	37
	5.2.1 Detailed Description	37
	5.2.2 Member Data Documentation	38
	5.2.2.1 hw session stat	
	5,2,2,2 internal	38
	5.2.2.3 thd_sess_stat	38
	5.2.2.4 total_in	38
	5.2.2.5 total_out	38
	5.3 QzSessionParams_S Struct Reference	38
	5.3.1 Detailed Description	39
	5.3.2 Member Data Documentation	39
	5.3.2.1 comp_algorithm	39
	5.3.2.2 comp_lvl	39
	5.3.2.3 data_fmt	39
	5.3.2.4 direction	39
	5.3.2.5 huffman_hdr	40
	5.3.2.6 hw_buff_sz	40
	5.3.2.7 input_sz_thrshold	40
	5.3.2.8 max_forks	40

5.3.2.9 req_cnt_thrshold	. 40
5.3.2.10 strm_buff_sz	. 40
5.3.2.11 sw_backup	. 40
5.3.2.12 wait_cnt_thrshold	. 41
5.4 QzSessionParamsGen3_S Struct Reference	. 41
5.4.1 Member Data Documentation	. 41
5.4.1.1 comp_algorithm	. 41
5.4.1.2 comp_lvl	. 41
5.4.1.3 crc_polynomial	. 42
5.4.1.4 data_fmt	. 42
5.4.1.5 direction	. 42
5.4.1.6 huffman_hdr	. 42
5.4.1.7 hw_buff_sz	. 42
5.4.1.8 input_sz_thrshold	. 42
5.4.1.9 is_sensitive_mode	. 42
5.4.1.10 lz4s_mini_match	. 43
5.4.1.11 max_forks	. 43
5.4.1.12 mem_type	. 43
5.4.1.13 polling_mode	. 43
5.4.1.14 qzCallback	. 43
5.4.1.15 qzCallback_external	. 43
5.4.1.16 req_cnt_thrshold	. 43
5.4.1.17 strm_buff_sz	. 43
5.4.1.18 sw_backup	. 44
5.4.1.19 wait_cnt_thrshold	. 44
5.5 QzSoftwareVersionInfo_S Struct Reference	. 44
5.6 QzStatus_S Struct Reference	. 44
5.6.1 Detailed Description	. 45
5.6.2 Member Data Documentation	. 45
5.6.2.1 algo_hw	. 45
5.6.2.2 algo_sw	. 45
5.6.2.3 hw_session_status	. 45
5.6.2.4 memory_alloced	. 45
5.6.2.5 qat_hw_count	. 45
5.6.2.6 qat_instance_attach	. 45
5.6.2.7 qat_mem_drvr	. 46
5.6.2.8 qat_service_init	. 46
5.6.2.9 using_huge_pages	. 46
5.7 QzStream_S Struct Reference	. 46
5.7.1 Detailed Description	. 46
5.7.2 Member Data Documentation	. 47
5.7.2.1 crc_32	. 47

	5.7.2.2 crc_type	47
	5.7.2.3 in	47
	5.7.2.4 in_sz	47
	5.7.2.5 opaque	47
	5.7.2.6 out	47
	5.7.2.7 out_sz	47
	5.7.2.8 pending_in	48
	5.7.2.9 pending_out	48
	5.7.2.10 reserved	48
5.8 Thread	List_S Struct Reference	48
6 File Docume	entation	49
6.1 applicat	tions.qat.shims.qatzip.qatzip/include/qatzip.h File Reference	49
6.1.1	Macro Definition Documentation	52
	6.1.1.1 QATZIP_API	52
	6.1.1.2 QATZIP_API_VERSION	53
	6.1.1.3 QZ_BUF_ERROR	53
	6.1.1.4 QZ_DATA_ERROR	53
	6.1.1.5 QZ_DEFLATE	53
	6.1.1.6 QZ_DISABLE_SOFTWARE_BACKUP	53
	6.1.1.7 QZ_DISABLE_SOFTWARE_ONLY_EXECUTION	53
	6.1.1.8 QZ_DUPLICATE	53
	6.1.1.9 QZ_ENABLE_SOFTWARE_BACKUP	54
	6.1.1.10 QZ_ENABLE_SOFTWARE_ONLY_EXECUTION	54
	6.1.1.11 QZ_FAIL	54
	6.1.1.12 QZ_FORCE_SW	54
	6.1.1.13 QZ_INTEG	54
	6.1.1.14 QZ_LOW_DEST_MEM	54
	6.1.1.15 QZ_LOW_MEM	54
	6.1.1.16 QZ_NO_HW	55
	6.1.1.17 QZ_NO_INST_ATTACH	55
	6.1.1.18 QZ_NO_MDRV	55
	6.1.1.19 QZ_NO_SW_AVAIL	55
	6.1.1.20 QZ_NONE	55
	6.1.1.21 QZ_NOSW_LOW_MEM	55
	6.1.1.22 QZ_NOSW_NO_HW	55
	6.1.1.23 QZ_NOSW_NO_INST_ATTACH	55
	6.1.1.24 QZ_NOSW_NO_MDRV	56
	6.1.1.25 QZ_NOSW_UNSUPPORTED_FMT	56
	6.1.1.26 QZ_PARAMS	56
	6.1.1.27 QZ_POST_PROCESS_ERROR	56
	6.1.1.28 QZ_TIMEOUT	56

Index	50
6.2 applications.qat.shims.qatzip.qatzip/include/qz_utils.h File Reference	. 5
6.1.2.1 QzDataFormatGen3_E	50
6.1.2 Enumeration Type Documentation	50
6.1.1.29 QZ_UNSUPPORTED_FMT	. 50

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Data Compression API													 							- 7	
debug API										 			 							36	

2 Module Index

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

tThread_S	37
Session_S	37
SessionParams_S	38
SessionParamsGen3_S	41
SoftwareVersionInfo_S	44
Status_S	44
Stream_S	46
readList S	48

4 Class Index

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

applications.qat.shims.qatzip.qatzip/include/qatzip.h	49
applications.qat.shims.qatzip.qatzip/include/qz_utils.h	57

6 File Index

Chapter 4

Module Documentation

4.1 Data Compression API

Classes

- struct QzSessionParams_S
- struct QzSession S
- struct QzStatus S
- struct QzStream_S

Macros

- #define QATZIP_API_VERSION_NUM_MAJOR (2)
- #define QATZIP_API_VERSION_NUM_MINOR (3)
- #define QZ_OK (0)
- #define QZ_SW_BACKUP_BIT_POSITION (0)
- #define QZ SW EXECUTION BIT (4)
- #define QZ_MAX_STRING_LENGTH 64
- #define QZ_SKID_PAD_SZ 48

Typedefs

- typedef enum QzHuffmanHdr_E QzHuffmanHdr_T
- typedef enum PinMem_E PinMem_T
- typedef enum QzDirection_E QzDirection_T
- typedef enum QzDataFormat E QzDataFormat T
- typedef enum QzPollingMode E QzPollingMode T
- typedef enum QzCrcType_E QzCrcType_T
- typedef enum QzCrcPolynomial_E QzCrcPolynomial_T
- typedef enum QzSoftwareComponentType_E QzSoftwareComponentType_T
- typedef int(* qzLZ4SCallbackFn) (void *external, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest_len, int *ExtStatus)
- typedef struct QzSessionParams_S QzSessionParams_T
- typedef struct QzSession_S QzSession_T
- typedef struct QzStatus_S QzStatus_T
- typedef struct QzStream_S QzStream_T

Enumerations

```
    enum QzHuffmanHdr E { QZ DYNAMIC HDR = 0 , QZ STATIC HDR }

• enum PinMem_E { COMMON_MEM = 0 , PINNED_MEM }

    enum QzDirection_E { QZ_DIR_COMPRESS = 0 , QZ_DIR_DECOMPRESS , QZ_DIR_BOTH }

enum QzDataFormat E {
 QZ DEFLATE 4B = 0, QZ DEFLATE GZIP, QZ DEFLATE GZIP EXT, QZ DEFLATE RAW,
 QZ FMT NUM }

    enum QzPollingMode E { QZ PERIODICAL POLLING = 0 , QZ BUSY POLLING }
```

- enum QzCrcType_E { QZ_CRC32 = 0 , QZ_ADLER , NONE }
- enum QzCrcPolynomial_E { QZ_CRC_POLYNOMIAL_DEFAULT = 0 }
- enum QzSoftwareComponentType E {

QZ COMPONENT FIRMWARE = 0, QZ COMPONENT KERNEL DRIVER, QZ COMPONENT USER ← DRIVER, QZ COMPONENT QATZIP API, QZ COMPONENT SOFTWARE PROVIDER }

Functions

- QATZIP API int gzInit (QzSession T *sess, unsigned char sw backup)
- QATZIP API int qzSetupSession (QzSession T *sess, QzSessionParams T *params)
- QATZIP API int gzCompress (QzSession T *sess, const unsigned char *src, unsigned int *src len, unsigned char *dest, unsigned int *dest_len, unsigned int last)
- QATZIP API int gzCompressCrc (QzSession T *sess, const unsigned char *src, unsigned int *src len, unsigned char *dest, unsigned int *dest len, unsigned int last, unsigned long *crc)
- QATZIP_API int qzDecompress (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest len)
- QATZIP API int gzTeardownSession (QzSession T *sess)
- QATZIP API int gzClose (QzSession T*sess)
- QATZIP_API int qzGetStatus (QzSession_T *sess, QzStatus_T *status)
- QATZIP API int gzSetDefaults (QzSessionParams T *defaults)
- QATZIP_API int qzGetDefaults (QzSessionParams_T *defaults)
- QATZIP_API void * qzMalloc (size t sz, int numa, int force pinned)
- QATZIP API void qzFree (void *m)
- QATZIP API int gzMemFindAddr (unsigned char *a)
- QATZIP_API int qzCompressStream (QzSession_T *sess, QzStream_T *strm, unsigned int last)
- QATZIP_API int qzDecompressStream (QzSession_T *sess, QzStream_T *strm, unsigned int last)
- QATZIP_API int qzEndStream (QzSession_T *sess, QzStream_T *strm)
- QATZIP API int qzGetSoftwareComponentVersionList (QzSoftwareVersionInfo T *api info, unsigned int *num_elem)
- QATZIP API int qzGetSoftwareComponentCount (unsigned int *num elem)

4.1.1 Detailed Description

@description These functions specify the API for data compression operations.

Remarks

4.1.2 Macro Definition Documentation

4.1.2.1 QATZIP_API_VERSION_NUM_MAJOR

```
#define QATZIP_API_VERSION_NUM_MAJOR (2)
```

QATzip Major Version Number @description The QATzip API major version number. This number will be incremented when significant changes to the API have occurred. The combination of the major and minor number definitions represent the complete version number for this interface.

4.1.2.2 QATZIP_API_VERSION_NUM_MINOR

```
#define QATZIP_API_VERSION_NUM_MINOR (3)
```

QATzip Minor Version Number @description The QATzip API minor version number. This number will be incremented when minor changes to the API have occurred. The combination of the major and minor number definitions represent the complete version number for this interface.

4.1.2.3 QZ_MAX_STRING_LENGTH

```
#define QZ_MAX_STRING_LENGTH 64
```

QATzip software version structure

@description This structure contains data relating to the versions of a QATZip or a subcomponent of this library platform.

4.1.2.4 QZ_OK

```
#define QZ_OK (0)
```

QATzip Session Status definitions and function return codes

@description This list identifies valid values for session status and function return codes. Success

4.1.2.5 QZ_SKID_PAD_SZ

```
#define QZ_SKID_PAD_SZ 48
```

Get the maximum compressed output length

@description Get the maximum compressed output length.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	src_sz	Input data length in bytes sess Session handle (pointer to opaque instance and session data)	
----	--------	--	--

Return values

dest_sz	Max compressed data output length in bytes. When src_sz is equal to 0, the return value is
	QZ_COMPRESSED_SZ_OF_EMPTY_FILE(34). When integer overflow happens, the return value is 0

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.2.6 QZ_SW_BACKUP_BIT_POSITION

```
#define QZ_SW_BACKUP_BIT_POSITION (0)
```

QATzip Session software configuration settings

@description The following definitions can be used with the sw_backup variable in structs and functions to configure the session

QZ_ENABLE_SOFTWARE_BACKUP Congifure session with software fallback

QZ ENABLE SOFTWARE ONLY EXECUTION Configure session to only use software

4.1.2.7 QZ_SW_EXECUTION_BIT

```
#define QZ_SW_EXECUTION_BIT (4)
```

QATzip Extended return information

@description The following definitions can be used with the extended return values.

QZ_SW_EXECUTION indicates if a request for services was performed in software.

QZ_HW_TIMEOUT indicates if a request to hardware was timed out.

If set in the extended return value, QZ_POST_PROCESS_FAIL indicates post processing of the LZ4s compressed data has failed.

4.1.3 Typedef Documentation

4.1.3.1 PinMem_T

typedef enum PinMem_E PinMem_T

Supported memory types

@description This enumerated list identifies memory types supported by QATzip.

4.1.3.2 QzCrcPolynomial T

typedef enum QzCrcPolynomial_E QzCrcPolynomial_T

Supported polynomial for CRC64 compression

@description This enumerated list identifies the polynomials available for use when a CRC or CRC64 is generated for a buffer.

@Defalut Polynomial: CRC-32 checksum is described in RFC 1952 Implementing RFC 1952 CRC: $http \leftarrow : //www.ietf.org/rfc/rfc1952.txt$

4.1.3.3 QzCrcType_T

typedef enum QzCrcType_E QzCrcType_T

Supported checksum type

@description This enumerated list identifies the checksum type for input/output data. The format can be CRC32, Adler or none.

4.1.3.4 QzDataFormat_T

 ${\tt typedef \ enum \ QzDataFormat_E \ QzDataFormat_T}$

Streaming API input and output format

@description This enumerated list identifies the data format supported by QATzip streaming API. A format can be raw deflate data block, deflate block wrapped by GZip header and footer, or deflate data block wrapped by GZip extension header and footer.

4.1.3.5 QzDirection_T

typedef enum QzDirection_E QzDirection_T

Compress or decompress setting

@description This enumerated list identifies the session directions supported by QATzip. A session can be compress, decompress or both.

4.1.3.6 QzHuffmanHdr_T

typedef enum QzHuffmanHdr_E QzHuffmanHdr_T

This API provides access to underlying compression functions in QAT hardware. The API supports an implementation that provides compression service in software if all of the required resources are not available to execute the compression service in hardware.

The API supports threaded applications. Applications can create threads and each of these threads can invoke the API defined herein.

For simplicity, initializations and setup function calls are not required to obtain compression services. If the initialization and setup functions are not called before compression or decompression requests, then they will be called with default arguments from within the compression or decompression functions. This results in several legal calling scenarios, described below.

Scenario 1 - All functions explicitly invoked by caller, with all arguments provided.

qzInit(&sess, sw_backup); qzSetupSession(&sess, ¶ms); qzCompress(&sess, src, &src_len, dest, &dest_len, 1); qzDecompress(&sess, src, &src_len, dest, &dest_len); qzTeardownSession(&sess); qzClose(&sess);

Scenario 2 - Initialization function called, setup function not invoked by caller. This scenario can be used to specify the sw_backup argument to qzInit.

qzInit(&sess, sw_backup); qzCompress(&sess, src, &src_len, dest, &dest_len, 1); calls qzSetupSession(sess, NULL); qzTeardownSession(&sess); qzClose(&sess);

Scenario 3 - Calling application simply invokes the actual qzCompress functions.

qzCompress(&sess, src, &src_len, dest, &dest_len, 0); calls qzInit(sess, 1); calls qzSetupSession(sess, NULL); qzCompress(&sess, src, &src_len, dest, &dest_len, 1);

Notes: Invoking qzSetupSession with NULL for params sets up a session with default session attributed, detailed in the function description below.

If an application terminates without invoking tear down and close functions, process termination will invoke memory and hardware instance cleanup.

If a thread terminates without invoking tear down and close functions, memory and hardware are not cleaned up until the application exits.

Additions for QAT 2.0 and beyond platforms though Extending QzSessionParamsGen3_T, QzDataFormatGen3_T and Using qzSetupSessionGen3 to setup session.

- 1. Addition of LZ4 and LZ4s
- 2. Addition of post processing functions for out of LZ4s
- 3. Compression level up to 12 for LZ4 and LZ4s
- 4. Support for Shared Virtual Memory
- Support for gzip header with additional compression algorithms
 Supported Huffman Headers

@description This enumerated list identifies the Huffman header types supported by QATzip.

4.1.3.7 qzLZ4SCallbackFn

typedef int(* qzLZ4SCallbackFn) (void *external, const unsigned char *src, unsigned int *src_ \leftarrow len, unsigned char *dest, unsigned int *dest_len, int *ExtStatus)

Post processing callback after LZ4s compression

@description This function will be called in qzCompressCrc for post processing of lz4s payloads. Function implementation should be provided by user and comply with this prototype's rules. The input paramter 'dest' will contain the compressed lz4s format data.

The user callback function should be provided through the QzSessionParams_T. And set data format of compression to 'QZ_LZ4S_FH', then post-processing will be trigger.

qzCallback's first parameter 'external' can be a customized compression context which can be setup before QAT qzSetupSession. It can be provided to QATZip though the 'qzCallback_external' variable in the QzSessionParams← T structure.

ExtStatus will be embedded into extended return codes when qzLZ4SCallbackFn return QZ_POST_PROCESS_ ERROR. See extended return code section and *Ext API for details.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	external	User context provided through the 'qzCallback_external' pointer in the
		QzSessionParams_T structure.
in	src	Point to source buffer
in,out	src_len	Length of source buffer. Modified to number of bytes consumed
in	dest	Point to destination buffer
in,out	dest_len	Length of destination buffer. Modified to length of compressed data when function
		returns
in,out	ExtStatus	'qzCallback' customized error code.

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	params are invalid
QZ_POST_PROCESS_ERROR	post processing error

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.3.8 QzPollingMode_T

```
typedef enum QzPollingMode_E QzPollingMode_T
```

Supported polling mode

@description Specifies whether the instance must be busy polling, or be periodical polling.

4.1.3.9 QzSession_T

```
typedef struct QzSession_S QzSession_T
```

QATzip Session opaque data storage

@description This structure contains a pointer to a structure with session state.

4.1.3.10 QzSessionParams_T

```
typedef struct QzSessionParams_S QzSessionParams_T
```

QATzip Session Initialization parameters

@description This structure contains data for initializing a session.

4.1.3.11 QzSoftwareComponentType_T

```
{\tt typedef\ enum\ QzSoftwareComponentType\_E\ QzSoftwareComponentType\_T}
```

Software Component type

@description This enumerated list specifies the type of software that is being described.

4.1.3.12 QzStatus_T

```
typedef struct QzStatus_S QzStatus_T
```

QATzip status structure

@description This structure contains data relating to the status of QAT on the platform.

4.1.3.13 QzStream_T

typedef struct QzStream_S QzStream_T

QATzip Stream data storage

@description This structure contains metadata needed for stream operation.

4.1.4 Enumeration Type Documentation

4.1.4.1 PinMem_E

enum PinMem_E

Supported memory types

@description This enumerated list identifies memory types supported by QATzip.

Enumerator

COMMON_MEM	Allocate non-contiguous memory
PINNED_MEM	Allocate contiguous memory

4.1.4.2 QzCrcPolynomial_E

enum QzCrcPolynomial_E

Supported polynomial for CRC64 compression

@description This enumerated list identifies the polynomials available for use when a CRC or CRC64 is generated for a buffer.

@Defalut Polynomial: CRC-32 checksum is described in RFC 1952 Implementing RFC 1952 CRC: $http \leftarrow : //www.ietf.org/rfc/rfc1952.txt$

Enumerator

4.1.4.3 QzCrcType_E

enum QzCrcType_E

Supported checksum type

@description This enumerated list identifies the checksum type for input/output data. The format can be CRC32, Adler or none.

Enumerator

QZ_CRC32	CRC32 checksum
QZ_ADLER	Adler checksum
NONE	No checksum

4.1.4.4 QzDataFormat_E

enum QzDataFormat_E

Streaming API input and output format

@description This enumerated list identifies the data format supported by QATzip streaming API. A format can be raw deflate data block, deflate block wrapped by GZip header and footer, or deflate data block wrapped by GZip extension header and footer.

Enumerator

QZ_DEFLATE_4B	Data is in raw deflate format with 4 byte header
QZ_DEFLATE_GZIP	Data is in deflate wrapped by GZip header and footer
QZ_DEFLATE_GZIP_EXT	Data is in deflate wrapped by GZip extended header and footer
QZ_DEFLATE_RAW	Data is in raw deflate format

4.1.4.5 QzDirection E

enum QzDirection_E

Compress or decompress setting

@description This enumerated list identifies the session directions supported by QATzip. A session can be compress, decompress or both.

Enumerator

QZ_DIR_COMPRESS	Session will be used for compression
QZ_DIR_DECOMPRESS	Session will be used for decompression
QZ_DIR_BOTH	Session will be used for both compression and decompression

4.1.4.6 QzHuffmanHdr_E

enum QzHuffmanHdr_E

This API provides access to underlying compression functions in QAT hardware. The API supports an implementation that provides compression service in software if all of the required resources are not available to execute the compression service in hardware.

The API supports threaded applications. Applications can create threads and each of these threads can invoke the API defined herein.

For simplicity, initializations and setup function calls are not required to obtain compression services. If the initialization and setup functions are not called before compression or decompression requests, then they will be called with default arguments from within the compression or decompression functions. This results in several legal calling scenarios, described below.

Scenario 1 - All functions explicitly invoked by caller, with all arguments provided.

qzInit(&sess, sw_backup); qzSetupSession(&sess, ¶ms); qzCompress(&sess, src, &src_len, dest, &dest_len, 1); qzDecompress(&sess, src, &src_len, dest, &dest_len); qzTeardownSession(&sess); qzClose(&sess);

Scenario 2 - Initialization function called, setup function not invoked by caller. This scenario can be used to specify the sw_backup argument to qzInit.

qzInit(&sess, sw_backup); qzCompress(&sess, src, &src_len, dest, &dest_len, 1); calls qzSetupSession(sess, NULL); qzTeardownSession(&sess); qzClose(&sess);

Scenario 3 - Calling application simply invokes the actual gzCompress functions.

qzCompress(&sess, src, &src_len, dest, &dest_len, 0); calls qzInit(sess, 1); calls qzSetupSession(sess, NULL); qzCompress(&sess, src, &src_len, dest, &dest_len, 1);

Notes: Invoking qzSetupSession with NULL for params sets up a session with default session attributed, detailed in the function description below.

If an application terminates without invoking tear down and close functions, process termination will invoke memory and hardware instance cleanup.

If a thread terminates without invoking tear down and close functions, memory and hardware are not cleaned up until the application exits.

Additions for QAT 2.0 and beyond platforms though Extending QzSessionParamsGen3_T, QzDataFormatGen3_T and Using qzSetupSessionGen3 to setup session.

- 1. Addition of LZ4 and LZ4s
- 2. Addition of post processing functions for out of LZ4s
- 3. Compression level up to 12 for LZ4 and LZ4s
- 4. Support for Shared Virtual Memory
- Support for gzip header with additional compression algorithms Supported Huffman Headers

@description This enumerated list identifies the Huffman header types supported by QATzip.

Enumerator

QZ_DYNAMIC_HDR	Full Dynamic Huffman Trees
QZ_STATIC_HDR	Static Huffman Trees

4.1.4.7 QzPollingMode_E

```
enum QzPollingMode_E
```

Supported polling mode

@description Specifies whether the instance must be busy polling, or be periodical polling.

Enumerator

QZ_PERIODICAL_POLLING	No busy polling
QZ_BUSY_POLLING	busy polling

4.1.4.8 QzSoftwareComponentType_E

```
enum QzSoftwareComponentType_E
```

Software Component type

@description This enumerated list specifies the type of software that is being described.

4.1.5 Function Documentation

4.1.5.1 qzClose()

Terminates a QATzip session

@description This function closes the connection with QAT.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)
----	------	--

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*sess is NULL or member of params is invalid

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.2 qzCompress()

Compress a buffer

@description This function will compress a buffer if either a hardware based session or a software based session is available. If no session has been established - as indicated by the contents of *sess - then this function will attempt to set up a session using qzInit and qzSetupSession.

The resulting compressed block of data will be composed of one or more gzip blocks, as per RFC 1952.

This function will place completed compression blocks in the output buffer.

The caller must check the updated src_len. This value will be the number of consumed bytes on exit. The calling API may have to process the destination buffer and call again.

The parameter dest_len will be set to the number of bytes produced in the destination buffer. This value may be zero if no data was produced which may occur if the consumed data is retained internally. A possible reason for this may be small amounts of data in the src buffer.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)
in	src	Point to source buffer
in,out	src_len	Length of source buffer. Modified to number of bytes consumed
in	dest	Point to destination buffer
in,out	dest_len	Length of destination buffer. Modified to length of compressed data when function
		returns
in	last	1 for 'No more data to be compressed' 0 for 'More data to be compressed'
in,out	ext_rc	qzCompressExt only. If not NULL, ext_rc point to a location where extended return codes may be returned. See extended return code section for details. if NULL, no extended information will be provided.

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*sess is NULL or member of params is invalid

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.3 qzCompressCrc()

Compress a buffer and return the CRC checksum

@description This function will compress a buffer if either a hardware based session or a software based session is available. If no session has been established - as indicated by the contents of *sess - then this function will attempt to set up a session using qzInit and qzSetupSession.

The resulting compressed block of data will be composed of one or more gzip blocks, as per RFC 1952.

This function will place completed compression blocks in the output buffer and put either a CRC32 or CRC64 checksum for the compressed input data in the user provided buffer *crc.

The caller must check the updated src_len. This value will be the number of consumed bytes on exit. The calling API may have to process the destination buffer and call again.

The parameter dest_len will be set to the number of bytes produced in the destination buffer. This value may be zero if no data was produced which may occur if the consumed data is retained internally. A possible reason for this may be small amounts of data in the src buffer.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)	
in	src	Point to source buffer	
in,out	src_len	Length of source buffer. Modified to number of bytes consumed	
in	dest	Point to destination buffer	
in,out	dest_len	Length of destination buffer. Modified to length of compressed data when function	
		returns	
in	last	1 for 'No more data to be compressed' 0 for 'More data to be compressed'	
in,out	crc	Pointer to CRC32 or CRC64 checksum buffer	
in,out	ext_rc	qzCompressCrcExt only. If not NULL, ext_rc point to a location where extended return	
		codes may be returned. See extended return code section for details. if NULL, no	
		extended information will be provided.	

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*sess is NULL or member of params is invalid

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.4 qzCompressStream()

Compress data in stream and return checksum

@description This function will compress data in stream buffer if either a hardware based session or a software based session is available. If no session has been established - as indicated by the contents of *sess - then this function will attempt to set up a session using qzInit and qzSetupSession. The function will start to compress the data when receiving sufficient number of bytes - as defined by hw_buff_sz in QzSessionParams_T - or reaching the end of input data - as indicated by last parameter.

The resulting compressed block of data will be composed of one or more gzip blocks, per RFC 1952, or deflate blocks, per RFC 1951.

This function will place completed compression blocks in the *out of QzStream_T structure and put checksum for compressed input data in crc32 of QzStream_T structure.

The caller must check the updated in_sz of QzStream_T. This value will be the number of consumed bytes on exit. The calling API may have to process the destination buffer and call again.

The parameter out_sz in QzStream_T will be set to the number of bytes produced in the destination buffer. This value may be zero if no data was produced which may occur if the consumed data is retained internally. A possible reason for this may be small amounts of data in the src buffer.

The caller must check the updated pending_in of QzStream_T. This value will be the number of unprocessed bytes held in QATzip. The calling API may have to feed more input data or indicate reaching the end of input and call again.

The caller must check the updated pending_out of QzStream_T. This value will be the number of processed bytes held in QATzip. The calling API may have to process the destination buffer and call again.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)	
in,out	strm	Stream handle	
in	last	1 for 'No more data to be compressed' 0 for 'More data to be compressed' (always set to 1 in the Microsoft(R) Windows(TM) QATzip implementation)	

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*sess is NULL or member of params is invalid

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.5 qzDecompress()

Decompress a buffer

@description This function will decompress a buffer if either a hardware based session or a software based session is available. If no session has been established - as indicated by the contents of *sess - then this function will attempt to set up a session using qzInit and qzSetupSession.

The input compressed block of data will be composed of one or more gzip blocks, as per RFC 1952.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)	
in	src	Point to source buffer	
in	src_len	Length of source buffer. Modified to length of processed compressed data when	
		function returns	
in	dest	Point to destination buffer	
in,out	dest_len	Length of destination buffer. Modified to length of decompressed data when function	
		returns	
in,out	ext_rc	qzDecompressExt only. If not NULL, ext_rc point to a location where extended return	
		codes may be returned. See extended return code section for details. if NULL, no	
		extended information will be provided.	

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*sess is NULL or member of params is invalid

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.6 qzDecompressStream()

Decompress data in stream and return checksum

@description This function will decompress data in stream buffer if either a hardware based session or a software based session is available. If no session has been established - as indicated by the contents of *sess - then this function will attempt to set up a session using qzInit and qzSetupSession. The function will start to decompress the data when receiving sufficient number of bytes - as defined by hw_buff_sz in QzSessionParams_T - or reaching the end of input data - as indicated by last parameter.

The input compressed block of data will be composed of one or more gzip blocks, per RFC 1952, or deflate blocks, per RFC 1951.

This function will place completed decompression blocks in the *out of QzStream_T structure and put checksum for decompressed data in crc32 of QzStream_T structure.

The caller must check the updated in_sz of QzStream_T. This value will be the number of consumed bytes on exit. The calling API may have to process the destination buffer and call again.

The parameter out_sz in QzStream_T will be set to the number of bytes produced in the destination buffer. This value may be zero if no data was produced which may occur if the consumed data is retained internally. A possible reason for this may be small amounts of data in the src buffer.

The caller must check the updated pending_in of QzStream_T. This value will be the number of unprocessed bytes held in QATzip. The calling API may have to feed more input data or indicate reaching the end of input and call again.

The caller must check the updated pending_out of QzStream_T. This value will be the number of processed bytes held in QATzip. The calling API may have to process the destination buffer and call again.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)	
in,out	strm	Stream handle	
in	last	1 for 'No more data to be compressed' 0 for 'More data to be compressed'	

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*sess is NULL or member of params is invalid
QZ NEED MORE	*last is set but end of block is absent

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.7 qzEndStream()

Terminates a QATzip stream

@description This function disconnects stream handle from session handle then reset stream flag and release stream memory.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)
----	------	--

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*sess is NULL or member of params is invalid

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.8 qzFree()

```
QATZIP_API void qzFree ( void * m )
```

Free allocated memory

@description Free allocated memory.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	т	Memory address to be freed
----	---	----------------------------

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.9 qzGetDefaults()

Get default QzSessionParams_T value

@description Get default QzSessionParams_T value.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

	in	defaults	The pointer to default value
--	----	----------	------------------------------

Return values

QZ_OK	Success on getting default value
QZ_PARAM	Fail to get default value

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.10 qzGetSoftwareComponentCount()

```
QATZIP_API int qzGetSoftwareComponentCount ( unsigned\ int\ *\ num\_elem\ )
```

Requests the number of Software components used by the QATZip library

@description This function populates num_elem variable with the number of software components available to the library.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant Yes @threadSafe Yes

Parameters

ſ	in,out	num_elem	pointer to an unsigned int to populate how many software componets are	1
			associated with QATZip	

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_NO_SW_AVAIL	Function did not find a software provider for fallback
QZ_NO_HW	Function did not find an installed kernel driver
QZ_NOSW_NO_HW	Functions did not find an installed kernel driver or software provider
QZ_PARAMS	*num_elem is NULL

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.11 qzGetSoftwareComponentVersionList()

Requests the release versions of the QATZip Library sub components.

@description Populate an array of pre-allocated QzSoftwareVersionInfo_T structs with the names and versions of QATzip sub components.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant Yes @threadSafe Yes

Parameters

in,out	api_info	pointer to a QzSoftwareVersionInfo_T structure to populate.
in,out	num_elem	pointer to an unsigned int expressing how many elements are in the array provided
		in api_info

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_NO_SW_AVAIL	Function did not find a software provider for fallback
QZ_NO_HW	Function did not find an installed kernel driver
QZ_NOSW_NO_HW	Functions did not find an installed kernel driver or software provider
QZ_PARAMS	*api_info or num_elem is NULL or not large enough to store all
	QzSoftwareVersionInfo_T structures

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.12 qzGetStatus()

Get current QAT status

@description This function retrieves the status of QAT in the platform. The status structure will be filled in as follows: qat_hw_count Number of discovered QAT devices on PCU bus qat_service_init 1 if qzInit has been successfully run, 0 otherwise qat_mem_drvr 1 if the QAT memory driver is installed, 0 otherwise qat_instance_attach 1 if session has attached to a hardware instance, 0 otherwise memory_alloced Amount of memory, in kilobytes, from kernel or huge pages allocated by this process/thread. using_huge_pages 1 if memory is being allocated from huge pages, 0 if memory is being allocated from standard kernel memory hw_session_status Hw session status: one of: QZ_OK QZ_FAIL QZ_NO_HW QZ_NO_MDRV QZ_NO_INST_ATTACH QZ_LOW_MEM QZ_NOSW_NO_HW QZ_NOSW_NO_MDRV QZ_NOSW_NO_INST_ATTACH QZ_NOSW_LOW_MEM QZ_NO_SW_AVAIL

Applications should verify the elements of the status structure are correct for the required operations. It should be noted that some information will be available only after qzInit has been called, either implicitly or explicitly. The qat_service_init element of the status structure will indicate if initialization has taken place.

The hw_session_status will depend on the availability of hardware based compression and software based compression. The following table indicates what hw_session_status based on the availability of compression engines and the sw_backup flag.

```
| HW | SW Engine | sw backup | hw session stat |
```

30 Module Documentation

avail	avail	setting	
N	N	0	QZ_NOSW_NO_HW
N	N	1	QZ_NOSW_NO_HW
N	Υ	0	QZ_FAIL
N	Υ	1	QZ_NO_HW (1)
Υ	N	0	QZ_OK
Υ	N	1	QZ_NO_SW_AVAIL (2)
Υ	Υ	0	QZ_OK
Υ	Υ	1	QZ_OK

Note 1: If an application indicates software backup is required by setting sw_backup=1, and a software engine is available and if no hardware based compression engine is available then the hw_session_status will be set to QZ_NO_HW. All compression and decompression will use the software engine. Note 2: If an application indicates software backup is required by setting sw_backup=1, and if no software based compression engine is available then the hw_session_status will be set to QZ_NO_SW_AVAIL. In this case, QAT based compression may be used however no software backup will available. If the application relies on software backup being avialable, then this return code can be treated as an error. @context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)
in	status	Pointer to QATzip status structure

Return values

QZ_OK	Function executed successfully. The hardware based compression session has been created
QZ PARAMS	*status is NULL

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.13 qzlnit()

Initialize QAT hardware

@description This function initializes the QAT hardware. This function is optional in the function calling sequence. If desired, this call can be made to avoid latency impact during the first call to qzDecompress or qzCompress, or to set the sw_backup parameter explicitly. The input parameter sw_backup specifies the behavior of the function and that of the functions called with the same session in the event there are insufficient resources to establish a QAT based compression or decompression session.

The required resources include access to the QAT hardware, contiguous pinned memory for mapping the hardware rings, and contiguous pinned memory for buffers.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects This function will: 1) start the user space driver if necessary 2) allocate all hardware instances available @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data.)	
in	sw_backup	see QZ_SW_* definitions for expected behavior	

Return values

QZ_OK	Function executed successfully. A hardware or software instance has been allocated to the calling process/thread
QZ_DUPLICATE	This process/thread already has a hardware instance
QZ_PARAMS	*sess is NULL
QZ_NOSW_NO_HW	No hardware and no software session being established
QZ_NOSW_NO_MDRV	No memory driver. No software session established
QZ_NOSW_NO_INST_ATTACH	No instance available No software session established
QZ_NOSW_LOW_MEM	Not enough pinned memory available No software session established
QZ_UNSUPPORTED_FMT	No support for requested algorithm; using software
QZ_NOSW_UNSUPPORTED_FMT	No support for requested algorithm; No software session established
QZ_NO_SW_AVAIL	No software is available. This will be returned when sw_backup is set but the session does not support software operations or software fallback is unavailable to the application.

Precondition

None

Postcondition

None

32 Module Documentation

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.14 qzMalloc()

Allocate different types of memory

@description Allocate different types of memory.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	SZ	Memory size to be allocated
in	numa	NUMA node from which to allocate memory
in	force_pinned	PINNED_MEM allocate contiguous memory COMMON_MEM allocate non-contiguous memory

Return values

NULL	Fail to allocate memory
address	The address of allocated memory

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.15 qzMemFindAddr()

Check whether the address is available

@description Check whether the address is available.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in a Address to be checked

Return values

1	The address is available
0	The address is not available

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.16 qzSetDefaults()

Set default QzSessionParams_T value

@description Set default QzSessionParams_T value.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

34 Module Documentation

Parameters

in	defaults	The pointer to value to be set as default]
----	----------	---	---

Return values

QZ_OK	Success on setting default value
QZ_PARAM	Fail to set default value

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.17 qzSetupSession()

Initialize a QATzip session

@description This function establishes a QAT session. This involves associating a hardware instance to the session, allocating buffers. If all of these activities can not be completed successfully, then this function will set up a software based session of param->sw_backup that is set to 1.

Before this function is called, the hardware must have been successfully started via qzInit.

If *sess includes an existing hardware or software session, then QZ_DUPLICATE will be returned without modifying the existing session.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)
in	params	Parameters for session

Return values

QZ_OK	Function executed successfully. A hardware or software based compression session has been created
QZ_DUPLICATE	*sess includes an existing hardware or software session
QZ_PARAMS	*sess is NULL or member of params is invalid
QZ_NOSW_NO_HW	No hardware and no sw session being established
QZ_NOSW_NO_MDRV	No memory driver. No software session established
QZ_NOSW_NO_INST_ATTACH	No instance available No software session established
QZ_NO_LOW_MEM	Not enough pinned memory available No software session established
QZ_UNSUPPORTED_FMT	No support for requested algorithm; using software
QZ_NOSW_UNSUPPORTED_FMT	No support for requested algorithm; No software session established
QZ_NO_SW_AVAIL	No software is available. This may returned when sw_backup is set to 1 but the session does not support software backup or software backup is unavailable to the application.

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.18 qzTeardownSession()

Uninitialize a QATzip session

@description This function disconnects a session from a hardware instance and deallocates buffers. If no session has been initialized, then no action will take place.

@context This function shall not be called in an interrupt context. @assumptions None @sideEffects None @blocking Yes @reentrant No @threadSafe Yes

Parameters

in	sess	Session handle (pointer to opaque instance and session data)

36 Module Documentation

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*sess is NULL or member of params is invalid

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.2 debug API

@description These functions specify the API for debug operations.

Remarks

Chapter 5

Class Documentation

5.1 QatThread_S Struct Reference

Public Attributes

- ThreadList_T * comp_th_list
- unsigned int num_comp_th
- pthread_mutex_t comp_lock
- ThreadList_T * decomp_th_list
- unsigned int num_decomp_th
- pthread_mutex_t decomp_lock

The documentation for this struct was generated from the following file:

• applications.qat.shims.qatzip.qatzip/include/qz_utils.h

5.2 QzSession_S Struct Reference

```
#include <qatzip.h>
```

Public Attributes

- signed long int hw_session_stat
- int thd_sess_stat
- void * internal
- unsigned long total_in
- unsigned long total_out

5.2.1 Detailed Description

QATzip Session opaque data storage

@description This structure contains a pointer to a structure with session state.

38 Class Documentation

5.2.2 Member Data Documentation

5.2.2.1 hw_session_stat

```
signed long int QzSession_S::hw_session_stat
```

Filled in during initialization, session startup and decompression

5.2.2.2 internal

```
void* QzSession_S::internal
```

Session data is opaque to outside world

5.2.2.3 thd_sess_stat

```
int QzSession_S::thd_sess_stat
```

Note process compression and decompression thread state

5.2.2.4 total_in

```
unsigned long QzSession_S::total_in
```

Total processed input data length in this session

5.2.2.5 total_out

```
unsigned long QzSession_S::total_out
```

Total output data length in this session

The documentation for this struct was generated from the following file:

· applications.qat.shims.qatzip.qatzip/include/qatzip.h

5.3 QzSessionParams_S Struct Reference

```
#include <qatzip.h>
```

Public Attributes

- QzHuffmanHdr_T huffman_hdr
- QzDirection_T direction
- QzDataFormat_T data_fmt
- unsigned int comp_lvl
- unsigned char comp_algorithm
- unsigned int max_forks
- unsigned char sw_backup
- unsigned int hw_buff_sz
- unsigned int strm_buff_sz
- · unsigned int input sz thrshold
- unsigned int req_cnt_thrshold
- unsigned int wait_cnt_thrshold

5.3.1 Detailed Description

QATzip Session Initialization parameters

@description This structure contains data for initializing a session.

5.3.2 Member Data Documentation

5.3.2.1 comp_algorithm

unsigned char QzSessionParams_S::comp_algorithm

Compress/decompression algorithms

5.3.2.2 comp_lvl

unsigned int QzSessionParams_S::comp_lvl

Compression level 1 to 9

5.3.2.3 data_fmt

QzDataFormat_T QzSessionParams_S::data_fmt

Deflate, deflate with GZip or deflate with GZip ext

5.3.2.4 direction

QzDirection_T QzSessionParams_S::direction

Compress or decompress

40 Class Documentation

5.3.2.5 huffman_hdr

```
QzHuffmanHdr_T QzSessionParams_S::huffman_hdr
```

Dynamic or Static Huffman headers

5.3.2.6 hw_buff_sz

```
unsigned int QzSessionParams_S::hw_buff_sz
```

Default buffer size, must be a power of 2k 4K,8K,16K,32K,64K,128K

5.3.2.7 input_sz_thrshold

```
unsigned int QzSessionParams_S::input_sz_thrshold
```

Default threshold of compression service's input size for sw failover, if the size of input request is less than the threshold, QATzip will route the request to software

5.3.2.8 max_forks

```
unsigned int QzSessionParams_S::max_forks
```

Maximum forks permitted in the current thread 0 means no forking permitted

5.3.2.9 req_cnt_thrshold

```
unsigned int QzSessionParams_S::req_cnt_thrshold
```

Set between 1 and NUM BUFF, default NUM BUFF NUM BUFF is defined in gatzip internal.h

5.3.2.10 strm_buff_sz

```
unsigned int QzSessionParams_S::strm_buff_sz
```

Stream buffer size between [1K .. 2M - 5K] Default strm_buf_sz equals to hw_buff_sz

5.3.2.11 sw_backup

```
unsigned char QzSessionParams_S::sw_backup
```

bit field defining SW configuration (see QZ_SW_* definitions)

5.3.2.12 wait_cnt_thrshold

```
unsigned int QzSessionParams_S::wait_cnt_thrshold
```

When previous try failed, wait for specific number of calls before retrying to open device. Default threshold is 8

The documentation for this struct was generated from the following file:

· applications.qat.shims.qatzip.qatzip/include/qatzip.h

5.4 QzSessionParamsGen3 S Struct Reference

Public Attributes

- QzHuffmanHdr_T huffman_hdr
- · QzDirection T direction
- QzDataFormatGen3_T data_fmt
- · unsigned int comp IvI
- unsigned char comp_algorithm
- unsigned int max_forks
- unsigned char sw_backup
- unsigned int hw_buff_sz
- unsigned int strm_buff_sz
- unsigned int input_sz_thrshold
- unsigned int req_cnt_thrshold
- unsigned int wait_cnt_thrshold
- PinMem_T mem_type
- qzLZ4SCallbackFn qzCallback
- void * qzCallback_external
- QzPollingMode_T polling_mode
- unsigned int is_sensitive_mode
- unsigned int lz4s_mini_match
- QzCrcPolynomial_T crc_polynomial

5.4.1 Member Data Documentation

5.4.1.1 comp_algorithm

unsigned char QzSessionParamsGen3_S::comp_algorithm

Compress/decompression algorithms

5.4.1.2 comp_lvl

unsigned int QzSessionParamsGen3_S::comp_lvl

Compression level 1 to 12 for QAT CPM2.0. If the comp_algorithm is deflate, values > max will be set to max

42 Class Documentation

5.4.1.3 crc_polynomial

```
QzCrcPolynomial_T QzSessionParamsGen3_S::crc_polynomial
```

When generating a CRC or CRC64 determines the polynomial used Default set to QZ_CRC_POLYNOMIAL_← DEFAULT

5.4.1.4 data_fmt

```
QzDataFormatGen3_T QzSessionParamsGen3_S::data_fmt
```

Deflate, deflate with GZip or deflate with GZip ext LZ4 or LZ4S and zstd

5.4.1.5 direction

QzDirection_T QzSessionParamsGen3_S::direction

Compress or decompress

5.4.1.6 huffman_hdr

QzHuffmanHdr_T QzSessionParamsGen3_S::huffman_hdr

Dynamic or Static Huffman headers

5.4.1.7 hw_buff_sz

unsigned int QzSessionParamsGen3_S::hw_buff_sz

Default buffer size, must be a power of 2k 4K,8K,16K,32K,64K,128K

5.4.1.8 input_sz_thrshold

```
unsigned int QzSessionParamsGen3_S::input_sz_thrshold
```

Default threshold of compression service's input size for sw failover, if the size of input request is less than the threshold, QATzip will route the request to software

5.4.1.9 is_sensitive_mode

 $\verb"unsigned" int QzSessionParamsGen3_S:: is_sensitive_mode"$

0 means disable sensitive mode, 1 means enable sensitive mode

5.4.1.10 lz4s_mini_match

unsigned int QzSessionParamsGen3_S::lz4s_mini_match

Set Iz4s dictionary mini match, which would be 3 or 4

5.4.1.11 max_forks

unsigned int QzSessionParamsGen3_S::max_forks

Maximum forks permitted in the current thread 0 means no forking permitted

5.4.1.12 mem_type

PinMem_T QzSessionParamsGen3_S::mem_type

If not specified, default will be Pinned for qat 1.x and common for QAT 2.0

5.4.1.13 polling_mode

QzPollingMode_T QzSessionParamsGen3_S::polling_mode

0 means no busy polling, 1 means busy polling

5.4.1.14 qzCallback

 $\verb|qzLZ4SCallbackFn|| QzSessionParamsGen3_S:: \verb|qzCallback||$

post processing callback for zstd compression

5.4.1.15 qzCallback_external

void* QzSessionParamsGen3_S::qzCallback_external

An opaque pointer provided by the user to be passed into qzCallback during post processing

5.4.1.16 req_cnt_thrshold

unsigned int QzSessionParamsGen3_S::req_cnt_thrshold

Set between 1 and NUM_BUFF, default NUM_BUFF NUM_BUFF is defined in qatzip_internal.h

5.4.1.17 strm_buff_sz

 ${\tt unsigned\ int\ QzSessionParamsGen3_S::strm_buff_sz}$

Stream buffer size between [1K .. 2M - 5K] Default strm_buf_sz equals to hw_buff_sz

44 Class Documentation

5.4.1.18 sw_backup

```
unsigned char QzSessionParamsGen3_S::sw_backup
```

bit field defining SW configuration (see QZ_SW_* definitions)

5.4.1.19 wait_cnt_thrshold

```
unsigned int QzSessionParamsGen3_S::wait_cnt_thrshold
```

When previous try failed, wait for specific number of calls before retrying to open device. Default threshold is 8

The documentation for this struct was generated from the following file:

· applications.qat.shims.qatzip.qatzip/include/qatzip.h

5.5 QzSoftwareVersionInfo_S Struct Reference

Public Attributes

- QzSoftwareComponentType_T component_type
- unsigned char component_name [QZ_MAX_STRING_LENGTH]
- · unsigned int major_version
- · unsigned int minor_version
- unsigned int patch_version
- unsigned int build_number
- unsigned char reserved [52]

The documentation for this struct was generated from the following file:

• applications.qat.shims.qatzip.qatzip/include/qatzip.h

5.6 QzStatus_S Struct Reference

```
#include <qatzip.h>
```

Public Attributes

- unsigned short int qat_hw_count
- · unsigned char qat_service_init
- unsigned char qat_mem_drvr
- unsigned char qat_instance_attach
- · unsigned long int memory alloced
- unsigned char using huge pages
- signed long int hw_session_status
- unsigned char algo_sw [QZ_MAX_ALGORITHMS]
- unsigned char algo_hw [QZ_MAX_ALGORITHMS]

5.6.1 Detailed Description

QATzip status structure

@description This structure contains data relating to the status of QAT on the platform.

5.6.2 Member Data Documentation

5.6.2.1 algo_hw

unsigned char QzStatus_S::algo_hw[QZ_MAX_ALGORITHMS]

Count of hardware devices supporting algorithms

5.6.2.2 algo_sw

unsigned char QzStatus_S::algo_sw[QZ_MAX_ALGORITHMS]

Support software algorithms

5.6.2.3 hw_session_status

signed long int QzStatus_S::hw_session_status

One of QATzip Session Status

5.6.2.4 memory_alloced

unsigned long int QzStatus_S::memory_alloced

Amount of memory allocated by this thread/process

5.6.2.5 qat_hw_count

unsigned short int QzStatus_S::qat_hw_count

From PCI scan

5.6.2.6 qat_instance_attach

unsigned char QzStatus_S::qat_instance_attach

Is this thread/g_process properly attached to an Instance?

46 Class Documentation

5.6.2.7 qat_mem_drvr

```
unsigned char QzStatus_S::qat_mem_drvr
```

1 if /dev/qat_mem exists 2 if /dev/qat_mem has been opened 0 otherwise

5.6.2.8 qat_service_init

```
unsigned char QzStatus_S::qat_service_init
```

Check if the available services have been initialized

5.6.2.9 using_huge_pages

```
unsigned char QzStatus_S::using_huge_pages
```

Are memory slabs coming from huge pages?

The documentation for this struct was generated from the following file:

· applications.qat.shims.qatzip.qatzip/include/qatzip.h

5.7 QzStream_S Struct Reference

```
#include <qatzip.h>
```

Public Attributes

- unsigned int in_sz
- unsigned int out_sz
- unsigned char * in
- $\bullet \ \ unsigned \ char * {\color{red} out}$
- unsigned int pending_in
- unsigned int pending_out
- QzCrcType_T crc_type
- unsigned int crc_32
- unsigned long long reserved
- void * opaque

5.7.1 Detailed Description

QATzip Stream data storage

@description This structure contains metadata needed for stream operation.

5.7.2 Member Data Documentation

5.7.2.1 crc_32

unsigned int QzStream_S::crc_32

Checksum value

5.7.2.2 crc_type

QzCrcType_T QzStream_S::crc_type

Checksum type in Adler, CRC32 or none

5.7.2.3 in

unsigned char* QzStream_S::in

Input data pointer set by application

5.7.2.4 in_sz

unsigned int QzStream_S::in_sz

Set by application, reset by QATzip to indicate consumed data

5.7.2.5 opaque

void* QzStream_S::opaque

Internal storage managed by QATzip

5.7.2.6 out

unsigned char* QzStream_S::out

Output data pointer set by application

5.7.2.7 out_sz

unsigned int QzStream_S::out_sz

Set by application, reset by QATzip to indicate processed data

48 Class Documentation

5.7.2.8 pending_in

```
unsigned int QzStream_S::pending_in
```

Unprocessed bytes held in QATzip

5.7.2.9 pending_out

```
unsigned int QzStream_S::pending_out
```

Processed bytes held in QATzip

5.7.2.10 reserved

```
unsigned long long QzStream_S::reserved
```

Reserved for future use

The documentation for this struct was generated from the following file:

• applications.qat.shims.qatzip.qatzip/include/qatzip.h

5.8 ThreadList_S Struct Reference

Public Attributes

- unsigned int thread_id
- unsigned int comp_hw_count
- unsigned int comp_sw_count
- unsigned int decomp_hw_count
- unsigned int decomp_sw_count
- struct ThreadList_S * next

The documentation for this struct was generated from the following file:

• applications.qat.shims.qatzip.qatzip/include/qz_utils.h

Chapter 6

File Documentation

6.1 applications.qat.shims.qatzip.qatzip/include/qatzip.h File Reference

```
#include <string.h>
#include <stdint.h>
```

Classes

- struct QzSessionParams_S
- struct QzSessionParamsGen3 S
- struct QzSession S
- struct QzStatus_S
- struct QzSoftwareVersionInfo_S
- struct QzStream_S

Macros

- #define QATZIP_API_VERSION_NUM_MAJOR (2)
- #define QATZIP_API_VERSION_NUM_MINOR (3)
- #define QATZIP_API_VERSION
- #define QATZIP_API
- #define QZ_OK (0)
- #define QZ_DUPLICATE (1)
- #define QZ_FORCE_SW (2)
- #define QZ_PARAMS (-1)
- #define QZ_FAIL (-2)
- #define QZ BUF ERROR (-3)
- #define QZ_DATA_ERROR (-4)
- #define QZ_TIMEOUT (-5)
- #define QZ_INTEG (-100)
- #define QZ_NO_HW (11)
- #define QZ NO MDRV (12)
- #define QZ_NO_INST_ATTACH (13)
- #define QZ LOW MEM (14)
- #define QZ_LOW_DEST_MEM (15)

50 File Documentation

```
    #define QZ_UNSUPPORTED_FMT (16)
```

- #define QZ_NONE (100)
- #define QZ_NOSW_NO_HW (-101)
- #define QZ_NOSW_NO_MDRV (-102)
- #define QZ_NOSW_NO_INST_ATTACH (-103)
- #define QZ NOSW LOW MEM (-104)
- #define QZ_NO_SW_AVAIL (-105)
- #define QZ_NOSW_UNSUPPORTED_FMT (-116)
- #define QZ_POST_PROCESS_ERROR (-117)
- #define QZ MAX ALGORITHMS ((int)255)
- #define QZ_DEFLATE ((unsigned char)8)
- #define QZ LZ4 ((unsigned char)'4')
- #define QZ_LZ4s ((unsigned char)'s')
- #define QZ_ZSTD ((unsigned char)'Z')
- #define **MIN**(a, b) (((a)<(b))?(a):(b))
- #define QZ_HUFF_HDR_DEFAULT QZ_DYNAMIC_HDR
- #define QZ DIRECTION DEFAULT QZ DIR BOTH
- #define QZ_DATA_FORMAT_DEFAULT QZ_DEFLATE_GZIP_EXT
- #define QZ COMP LEVEL DEFAULT 1
- #define QZ_COMP_ALGOL_DEFAULT QZ_DEFLATE
- #define QZ POLL SLEEP DEFAULT 10
- #define QZ MAX FORK DEFAULT 3
- #define QZ SW BACKUP DEFAULT 1
- #define QZ_HW_BUFF_SZ (64*1024)
- #define QZ_HW_BUFF_SZ_Gen3 (1*1024*1024)
- #define QZ_HW_BUFF_MIN_SZ (1*1024)
- #define QZ_HW_BUFF_MAX_SZ (512*1024)
- #define QZ_HW_BUFF_MAX_SZ_Gen3 (2*1024*1024*1024U)
- #define QZ_STRM_BUFF_SZ_DEFAULT QZ_HW_BUFF_SZ
- #define QZ_STRM_BUFF_MIN_SZ (1*1024)
- #define QZ STRM BUFF MAX SZ (2*1024*1024 5*1024)
- #define QZ_COMP_THRESHOLD_DEFAULT 1024
- #define QZ COMP THRESHOLD MINIMUM 128
- #define QZ_REQ_THRESHOLD_MINIMUM 1
- #define QZ REQ THRESHOLD MAXIMUM NUM BUFF
- #define QZ_REQ_THRESHOLD_DEFAULT QZ_REQ_THRESHOLD_MAXIMUM
- #define QZ_WAIT_CNT_THRESHOLD_DEFAULT 8
- #define QZ_DEFLATE_COMP_LVL_MINIMUM (1)
- #define QZ_DEFLATE_COMP_LVL_MAXIMUM (9)
- #define QZ_LZS_COMP_LVL_MINIMUM (1)
- #define QZ_LZS_COMP_LVL_MAXIMUM (12)
- #define QZ_SW_BACKUP_BIT_POSITION (0)
- #define QZ_SW_FORCESW_BIT_POSITION (1)
- #define QZ_ENABLE_SOFTWARE_BACKUP(_BackupVariable) (_BackupVariable |= (1 << QZ_SW_BACKUP_BIT_POSITIC
- #define QZ_ENABLE_SOFTWARE_ONLY_EXECUTION(_BackupVariable) (_BackupVariable |= (1 << QZ_SW_FORCESW_BIT_POSITION))
- #define QZ_DISABLE_SOFTWARE_BACKUP(_BackupVariable) (_BackupVariable &= \sim (1 << QZ_SW_BACKUP_BIT_POSI
- #define QZ_DISABLE_SOFTWARE_ONLY_EXECUTION(_BackupVariable) (_BackupVariable &= \sim (1 << QZ_SW_FORCESW_BIT_POSITION))
- #define QZ_SW_EXECUTION_BIT (4)
- #define QZ_SW_EXECUTION_MASK (1 << QZ_SW_EXECUTION_BIT)
- #define QZ_SW_EXECUTION(ret, ext_rc) (!ret && (ext_rc & QZ_SW_EXECUTION_MASK))
- #define QZ_TIMEOUT_BIT (8)
- #define QZ_TIMEOUT_MASK (1 << QZ_TIMEOUT_BIT)
- #define QZ_HW_TIMEOUT(ret, ext_rc) (!ret && (ext_rc & QZ_TIMEOUT_MASK))

- #define QZ POST PROCESS FAIL BIT (10)
- #define QZ_POST_PROCESS_FAIL_MASK (1 << QZ_POST_PROCESS_FAIL_BIT)
- #define QZ POST PROCESS FAIL (ret, ext rc) (ret && (ext rc & QZ POST PROCESS FAIL MASK))
- #define QZ MAX STRING LENGTH 64
- #define QZ SKID PAD SZ 48
- #define QZ_COMPRESSED_SZ_OF_EMPTY_FILE 34

Typedefs

- typedef enum QzHuffmanHdr E QzHuffmanHdr T
- typedef enum PinMem E PinMem T
- typedef enum QzDirection_E QzDirection_T
- typedef enum QzDataFormat_E QzDataFormat_T
- typedef enum QzDataFormatGen3_E QzDataFormatGen3_T
- typedef enum QzPollingMode_E QzPollingMode_T
- typedef enum QzCrcType_E QzCrcType_T
- typedef enum QzCrcPolynomial_E QzCrcPolynomial_T
- typedef enum QzSoftwareComponentType E QzSoftwareComponentType T
- typedef int(* qzLZ4SCallbackFn) (void *external, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest_len, int *ExtStatus)
- typedef struct QzSessionParams S QzSessionParams T
- typedef struct QzSessionParamsGen3 S QzSessionParamsGen3_T
- typedef struct QzSession S QzSession T
- typedef struct QzStatus S QzStatus T
- typedef struct QzSoftwareVersionInfo_S QzSoftwareVersionInfo_T

enum QzHuffmanHdr E { QZ DYNAMIC HDR = 0 , QZ STATIC HDR }

typedef struct QzStream_S QzStream_T

Enumerations

```
enum PinMem_E { COMMON_MEM = 0 , PINNED_MEM }
enum QzDirection_E { QZ_DIR_COMPRESS = 0 , QZ_DIR_DECOMPRESS , QZ_DIR_BOTH }
enum QzDataFormat_E {
QZ_DEFLATE_4B = 0 , QZ_DEFLATE_GZIP , QZ_DEFLATE_GZIP_EXT , QZ_DEFLATE_RAW ,
QZ_FMT_NUM }
enum QzDataFormatGen3_E {
QZ_DEFLATE_4B_Gen3 = 0 , QZ_DEFLATE_GZIP_Gen3 , QZ_DEFLATE_GZIP_EXT_Gen3 , QZ_DEFLATE_RAW_Gen3 ,
QZ_LZ4_FH , QZ_LZ4S_FH , QZ_LZ4S_PP , QZ_ZSTD_RAW }
enum QzPollingMode_E { QZ_PERIODICAL_POLLING = 0 , QZ_BUSY_POLLING }
```

- enum QzCrcType E { QZ CRC32 = 0 , QZ ADLER , NONE }
- enum QzCrcPolynomial_E { QZ_CRC_POLYNOMIAL_DEFAULT = 0 }
- enum QzSoftwareComponentType E {

```
QZ_COMPONENT_FIRMWARE = 0 , QZ_COMPONENT_KERNEL_DRIVER , QZ_COMPONENT_USER ← DRIVER , QZ_COMPONENT_QATZIP_API , QZ_COMPONENT_SOFTWARE_PROVIDER }
```

52 File Documentation

Functions

- QATZIP_API int qzInit (QzSession_T *sess, unsigned char sw_backup)
- QATZIP_API int qzSetupSession (QzSession_T *sess, QzSessionParams_T *params)
- QATZIP_API int qzSetupSessionGen3 (QzSession_T *sess, QzSessionParamsGen3_T *params)
- QATZIP_API int qzCompress (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest_len, unsigned int last)
- QATZIP_API int qzCompressExt (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest_len, unsigned int last, uint64_t *ext_rc)
- QATZIP_API int qzCompressCrc (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest_len, unsigned int last, unsigned long *crc)
- QATZIP_API int qzCompressCrcExt (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest len, unsigned int last, unsigned long *crc, uint64 t *ext rc)
- QATZIP_API int qzCompressCrc64 (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest len, unsigned int last, uint64 t *crc)
- QATZIP_API int qzCompressCrc64Ext (QzSession_T *sess, const unsigned char *src, unsigned int *src
 —len, unsigned char *dest, unsigned int *dest_len, unsigned int last, uint64_t *crc, uint64_t *ext_rc)
- QATZIP_API int qzDecompress (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest len)
- QATZIP_API int qzDecompressExt (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest len, uint64 t *ext rc)
- QATZIP_API int qzTeardownSession (QzSession_T *sess)
- QATZIP_API int qzClose (QzSession_T *sess)
- QATZIP_API int qzGetStatus (QzSession_T *sess, QzStatus_T *status)
- QATZIP API unsigned int qzMaxCompressedLength (unsigned int src sz, QzSession T *sess)
- QATZIP API int qzSetDefaults (QzSessionParams T *defaults)
- QATZIP_API int qzSetDefaultsGen3 (QzSessionParamsGen3_T *defaults)
- QATZIP API int gzGetDefaults (QzSessionParams T*defaults)
- QATZIP_API int qzGetDefaultsGen3 (QzSessionParamsGen3_T *defaults)
- QATZIP API void * qzMalloc (size t sz, int numa, int force pinned)
- QATZIP API void qzFree (void *m)
- QATZIP API int qzMemFindAddr (unsigned char *a)
- QATZIP_API int qzCompressStream (QzSession_T *sess, QzStream_T *strm, unsigned int last)
- QATZIP_API int qzDecompressStream (QzSession_T *sess, QzStream_T *strm, unsigned int last)
- QATZIP_API int qzEndStream (QzSession_T *sess, QzStream_T *strm)
- QATZIP_API int qzGetSoftwareComponentVersionList (QzSoftwareVersionInfo_T *api_info, unsigned int *num elem)
- QATZIP API int qzGetSoftwareComponentCount (unsigned int *num elem)

6.1.1 Macro Definition Documentation

6.1.1.1 QATZIP_API

#define QATZIP_API

These macros define how the project will be built QATZIP_LINK_DLL must be defined if linking the DLL QATZIP ← _BUILD_DLL must be defined when building a DLL No definition required if building the project as static library

6.1.1.2 QATZIP_API_VERSION

```
#define QATZIP_API_VERSION
```

Value:

```
(QATZIP_API_VERSION_NUM_MAJOR * 10000 + QATZIP_API_VERSION_NUM_MINOR * 100)
```

6.1.1.3 QZ_BUF_ERROR

```
#define QZ_BUF_ERROR (-3)
```

Insufficient buffer error

6.1.1.4 QZ_DATA_ERROR

```
#define QZ_DATA_ERROR (-4)
```

Input data was corrupted

6.1.1.5 QZ_DEFLATE

```
#define QZ_DEFLATE ((unsigned char)8)
```

used in gzip header to indicate deflate blocks and in session params

6.1.1.6 QZ_DISABLE_SOFTWARE_BACKUP

SW backup/fallback disabled

6.1.1.7 QZ_DISABLE_SOFTWARE_ONLY_EXECUTION

Disable SW only compression/decompression operations

6.1.1.8 QZ_DUPLICATE

```
#define QZ_DUPLICATE (1)
```

Can not process function again. No failure

54 File Documentation

6.1.1.9 QZ_ENABLE_SOFTWARE_BACKUP

SW backup/fallback enabled

6.1.1.10 QZ_ENABLE_SOFTWARE_ONLY_EXECUTION

Force SW to perform all compression/decompression operations

6.1.1.11 QZ_FAIL

```
\#define QZ\_FAIL (-2)
```

Unspecified error

6.1.1.12 QZ_FORCE_SW

```
#define QZ_FORCE_SW (2)
```

Using SW: Switch to software because of previous block

6.1.1.13 **QZ_INTEG**

```
#define QZ_INTEG (-100)
```

Integrity checked failed

6.1.1.14 QZ_LOW_DEST_MEM

```
#define QZ_LOW_DEST_MEM (15)
```

Using SW: Not enough pinned memory for dest buffer

6.1.1.15 QZ_LOW_MEM

```
#define QZ_LOW_MEM (14)
```

Using SW: Not enough pinned memory

6.1.1.16 QZ_NO_HW

```
#define QZ_NO_HW (11)
```

Using SW: No QAT HW detected

6.1.1.17 QZ_NO_INST_ATTACH

```
#define QZ_NO_INST_ATTACH (13)
```

Using SW: Could not attach to an instance

6.1.1.18 QZ_NO_MDRV

```
#define QZ_NO_MDRV (12)
```

Using SW: No memory driver detected

6.1.1.19 QZ_NO_SW_AVAIL

```
#define QZ_NO_SW_AVAIL (-105)
```

Session may require software, but no software is available

6.1.1.20 QZ_NONE

```
#define QZ_NONE (100)
```

Device uninitialized

6.1.1.21 QZ_NOSW_LOW_MEM

```
#define QZ_NOSW_LOW_MEM (-104)
```

Not using SW: not enough pinned memory

6.1.1.22 QZ_NOSW_NO_HW

```
#define QZ_NOSW_NO_HW (-101)
```

Not using SW: No QAT HW detected

6.1.1.23 QZ_NOSW_NO_INST_ATTACH

```
#define QZ_NOSW_NO_INST_ATTACH (-103)
```

Not using SW: Could not attach to instance

56 File Documentation

6.1.1.24 QZ_NOSW_NO_MDRV

```
#define QZ_NOSW_NO_MDRV (-102)
```

Not using SW: No memory driver detected

6.1.1.25 QZ_NOSW_UNSUPPORTED_FMT

```
#define QZ_NOSW_UNSUPPORTED_FMT (-116)
```

Not using SW: QAT device does not support data format

6.1.1.26 QZ PARAMS

```
#define QZ_PARAMS (-1)
```

Invalid parameter in function call

6.1.1.27 QZ_POST_PROCESS_ERROR

```
#define QZ_POST_PROCESS_ERROR (-117)
```

Using post process: post process callback encountered an error

6.1.1.28 **QZ_TIMEOUT**

```
#define QZ_TIMEOUT (-5)
```

Operation timed out

6.1.1.29 QZ UNSUPPORTED FMT

```
#define QZ_UNSUPPORTED_FMT (16)
```

Using SW: QAT device does not support data format

6.1.2 Enumeration Type Documentation

6.1.2.1 QzDataFormatGen3_E

enum QzDataFormatGen3_E

Enumerator

QZ_DEFLATE_4B_Gen3	Data is in raw deflate format with 4 byte header
QZ_DEFLATE_GZIP_Gen3	Data is in deflate wrapped by GZip header and footer
QZ_DEFLATE_GZIP_EXT_Gen3	Data is in deflate wrapped by GZip extended header and footer
QZ_DEFLATE_RAW_Gen3	Data is in raw deflate format
QZ_LZ4_FH	Data is in LZ4 format with frame headers
QZ_LZ4S_FH	Data is in LZ4s format with frame headers
QZ_LZ4S_PP	Data is in LZ4s format and has been post processed
QZ_ZSTD_RAW	Data is in raw zStandard format

6.2 applications.qat.shims.qatzip.qatzip/include/qz_utils.h File Reference

```
#include <stdarg.h>
#include <pthread.h>
#include <stdio.h>
```

Classes

- struct ThreadList_S
- struct QatThread_S

Macros

• #define QZ_DEBUG(...)

Typedefs

- typedef enum SERV_E Serv_T
- typedef enum ENGINE_E Engine_T
- typedef struct ThreadList_S ThreadList_T
- typedef struct QatThread_S QatThread_T

Enumerations

- enum SERV_E { COMPRESSION = 0 , DECOMPRESSION }
- enum **ENGINE_E** { HW = 0, SW }

Functions

- void initDebugLock (void)
- void dumpThreadInfo (void)
- void **insertThread** (unsigned int th_id, Serv_T serv_type, Engine_T engine_type)

58 File Documentation

Index

algo_hw	qzClose, 18
QzStatus_S, 45	qzCompress, 19
algo_sw	qzCompressCrc, 20
QzStatus_S, 45	qzCompressStream, 21
applications.qat.shims.qatzip.qatzip/include/qatzip.h, 49	QzCrcPolynomial_E, 15
applications.qat.shims.qatzip.qatzip/include/qz_utils.h,	QzCrcPolynomial_T, 11
57	QzCrcType_E, 15
	QzCrcType_T, 11
COMMON_MEM	QzDataFormat_E, 16
Data Compression API, 15	QzDataFormat_T, 11
comp_algorithm	qzDecompress, 23
QzSessionParams_S, 39	qzDecompressStream, 24
QzSessionParamsGen3_S, 41	QzDirection_E, 16
comp_lvl	QzDirection_T, 11
QzSessionParams_S, 39	qzEndStream, 25
QzSessionParamsGen3_S, 41	qzFree, 26
crc_32	qzGetDefaults, 27
QzStream_S, 47	qzGetSoftwareComponentCount, 27
crc_polynomial	qzGetSoftwareComponentVersionList, 28
QzSessionParamsGen3_S, 41	qzGetStatus, 29
crc_type	QzHuffmanHdr_E, 16
QzStream_S, 47	QzHuffmanHdr_T, 11
	qzInit, 30
Data Compression API, 7	qzLZ4SCallbackFn, 12
COMMON_MEM, 15	qzMalloc, 32
NONE, 16	qzMemFindAddr, 32
PinMem_E, 15	QzPollingMode_E, 18
PinMem_T, 11	QzPollingMode_T, 14
PINNED_MEM, 15	QzSession_T, 14
QATZIP_API_VERSION_NUM_MAJOR, 8	QzSessionParams_T, 14
QATZIP_API_VERSION_NUM_MINOR, 9	qzSetDefaults, 33
QZ_ADLER, 16	qzSetupSession, 34
QZ_BUSY_POLLING, 18	QzSoftwareComponentType E, 18
QZ_CRC32, 16	QzSoftwareComponentType_T, 14
QZ_CRC_POLYNOMIAL_DEFAULT, 15	QzStatus_T, 14
QZ_DEFLATE_4B, 16	QzStream_T, 14
QZ_DEFLATE_GZIP, 16	qzTeardownSession, 35
QZ_DEFLATE_GZIP_EXT, 16	data_fmt
QZ_DEFLATE_RAW, 16	QzSessionParams_S, 39
QZ_DIR_BOTH, 16	QzSessionParamsGen3 S, 42
QZ_DIR_COMPRESS, 16	debug API, 36
QZ_DIR_DECOMPRESS, 16	direction
QZ_DYNAMIC_HDR, 18	QzSessionParams_S, 39
QZ_MAX_STRING_LENGTH, 9	QzSessionParamsGen3_S, 42
QZ_OK, 9	<u> </u>
QZ_PERIODICAL_POLLING, 18	huffman_hdr
QZ_SKID_PAD_SZ, 9	QzSessionParams_S, 39
QZ_STATIC_HDR, 18	QzSessionParamsGen3_S, 42
QZ_SW_BACKUP_BIT_POSITION, 10	hw_buff_sz
QZ SW EXECUTION BIT. 10	OzsaccionParame S 40

QzSessionParamsGen3_S, 42	QatThread_S, 37
hw_session_stat	qatzip.h
QzSession_S, 38	QATZIP_API, 52
hw_session_status	QATZIP_API_VERSION, 52
QzStatus_S, 45	QZ_BUF_ERROR, 53
	QZ_DATA_ERROR, 53
in	QZ_DEFLATE, 53
QzStream_S, 47	QZ_DEFLATE_4B_Gen3, 57
in_sz	QZ_DEFLATE_GZIP_EXT_Gen3, 57
QzStream_S, 47	QZ DEFLATE GZIP Gen3, 57
input_sz_thrshold	QZ DEFLATE RAW Gen3, 57
QzSessionParams_S, 40	QZ_DISABLE_SOFTWARE_BACKUP, 53
QzSessionParamsGen3_S, 42	QZ_DISABLE_SOFTWARE_ONLY_EXECUTION,
internal	53
QzSession_S, 38	QZ DUPLICATE, 53
is_sensitive_mode	QZ_ENABLE_SOFTWARE_BACKUP, 53
QzSessionParamsGen3_S, 42	QZ_ENABLE_SOFTWARE_ONLY_EXECUTION,
<u> </u>	54
lz4s_mini_match	QZ FAIL, 54
QzSessionParamsGen3 S, 42	QZ_FORCE_SW, 54
	QZ_INTEG, 54
max_forks	_ · · · ·
QzSessionParams_S, 40	QZ_LOW_DEST_MEM, 54
QzSessionParamsGen3_S, 43	QZ_LOW_MEM, 54
mem_type	QZ_LZ4_FH, 57
QzSessionParamsGen3_S, 43	QZ_LZ4S_FH, 57
memory_alloced	QZ_LZ4S_PP, 57
QzStatus_S, 45	QZ_NO_HW, 54
-	QZ_NO_INST_ATTACH, 55
NONE	QZ_NO_MDRV, 55
Data Compression API, 16	QZ_NO_SW_AVAIL, 55
	QZ_NONE, 55
opaque	QZ_NOSW_LOW_MEM, 55
QzStream_S, 47	QZ_NOSW_NO_HW, 55
out	QZ_NOSW_NO_INST_ATTACH, 55
QzStream_S, 47	QZ_NOSW_NO_MDRV, 55
out_sz	QZ_NOSW_UNSUPPORTED_FMT, 56
QzStream_S, 47	QZ_PARAMS, 56
	QZ_POST_PROCESS_ERROR, 56
pending_in	QZ_TIMEOUT, 56
QzStream_S, 47	QZ_UNSUPPORTED_FMT, 56
pending_out	QZ_ZSTD_RAW, 57
QzStream_S, 48	QzDataFormatGen3_E, 56
PinMem_E	QATZIP_API
Data Compression API, 15	qatzip.h, 52
PinMem_T	QATZIP_API_VERSION
Data Compression API, 11	qatzip.h, 52
PINNED_MEM	QATZIP_API_VERSION_NUM_MAJOR
Data Compression API, 15	Data Compression API, 8
polling_mode	QATZIP_API_VERSION_NUM_MINOR
QzSessionParamsGen3_S, 43	Data Compression API, 9
	QZ_ADLER
qat_hw_count	Data Compression API, 16
QzStatus_S, 45	QZ_BUF_ERROR
qat_instance_attach	qatzip.h, 53
QzStatus_S, 45	QZ_BUSY_POLLING
qat_mem_drvr	Data Compression API, 18
QzStatus_S, 45	QZ_CRC32
qat_service_init	Data Compression API, 16
QzStatus_S, 46	- ata 6011p100010117111, 10

QZ_CRC_POLYNOMIAL_DEFAULT	QZ_NO_HW
Data Compression API, 15	qatzip.h, 54
QZ_DATA_ERROR	QZ_NO_INST_ATTACH
qatzip.h, 53	qatzip.h, 55
QZ DEFLATE	QZ_NO_MDRV
qatzip.h, 53	qatzip.h, <mark>55</mark>
QZ DEFLATE 4B	QZ_NO_SW_AVAIL
Data Compression API, 16	qatzip.h, 55
QZ_DEFLATE_4B_Gen3	QZ NONE
gatzip.h, 57	qatzip.h, 55
QZ DEFLATE GZIP	QZ NOSW LOW MEM
Data Compression API, 16	qatzip.h, 55
QZ_DEFLATE_GZIP_EXT	QZ_NOSW_NO_HW
Data Compression API, 16	qatzip.h, 55
•	
QZ_DEFLATE_GZIP_EXT_Gen3	QZ_NOSW_NO_INST_ATTACH
qatzip.h, 57	qatzip.h, 55
QZ_DEFLATE_GZIP_Gen3	QZ_NOSW_NO_MDRV
qatzip.h, 57	qatzip.h, 55
QZ_DEFLATE_RAW	QZ_NOSW_UNSUPPORTED_FMT
Data Compression API, 16	qatzip.h, 56
QZ_DEFLATE_RAW_Gen3	QZ_OK
qatzip.h, 57	Data Compression API, 9
QZ_DIR_BOTH	QZ_PARAMS
Data Compression API, 16	qatzip.h, 56
QZ_DIR_COMPRESS	QZ_PERIODICAL_POLLING
Data Compression API, 16	Data Compression API, 18
QZ_DIR_DECOMPRESS	QZ_POST_PROCESS_ERROR
Data Compression API, 16	qatzip.h, <mark>56</mark>
QZ_DISABLE_SOFTWARE_BACKUP	QZ_SKID_PAD_SZ
qatzip.h, 53	Data Compression API, 9
QZ_DISABLE_SOFTWARE_ONLY_EXECUTION	QZ_STATIC_HDR
qatzip.h, 53	Data Compression API, 18
QZ_DUPLICATE	QZ_SW_BACKUP_BIT_POSITION
qatzip.h, 53	Data Compression API, 10
QZ_DYNAMIC_HDR	QZ_SW_EXECUTION_BIT
Data Compression API, 18	Data Compression API, 10
QZ_ENABLE_SOFTWARE_BACKUP	QZ_TIMEOUT
qatzip.h, 53	qatzip.h, <mark>56</mark>
QZ_ENABLE_SOFTWARE_ONLY_EXECUTION	QZ_UNSUPPORTED_FMT
qatzip.h, 54	qatzip.h, 56
QZ_FAIL	QZ_ZSTD_RAW
qatzip.h, 54	qatzip.h, 57
QZ_FORCE_SW	qzCallback
qatzip.h, 54	QzSessionParamsGen3_S, 43
QZ_INTEG	qzCallback_external
qatzip.h, 54	QzSessionParamsGen3_S, 43
QZ_LOW_DEST_MEM	qzClose
gatzip.h, 54	Data Compression API, 18
QZ_LOW_MEM	qzCompress
gatzip.h, 54	Data Compression API, 19
QZ_LZ4_FH	qzCompressCrc
qatzip.h, 57	Data Compression API, 20
QZ_LZ4S_FH	qzCompressStream
qatzip.h, 57	Data Compression API, 21
QZ_LZ4S_PP	QzCrcPolynomial_E
qatzip.h, 57	Data Compression API, 15
QZ_MAX_STRING_LENGTH	QzCrcPolynomial_T
Data Compression API, 9	Data Compression API, 11
= a.ap	- a.a. comprocolon / ii i, i i

QzCrcType_E	direction, 39
Data Compression API, 15	huffman_hdr, 39
QzCrcType_T	hw_buff_sz, 40
Data Compression API, 11	input_sz_thrshold, 40
QzDataFormat_E	max_forks, 40
Data Compression API, 16	req_cnt_thrshold, 40
QzDataFormat_T	strm_buff_sz, 40
Data Compression API, 11	sw_backup, 40
QzDataFormatGen3_E	wait_cnt_thrshold, 40
qatzip.h, 56	QzSessionParams_T
qzDecompress	Data Compression API, 14
Data Compression API, 23	QzSessionParamsGen3_S, 41
qzDecompressStream	comp_algorithm, 41
Data Compression API, 24	comp_lvl, 41
QzDirection_E	crc_polynomial, 41
Data Compression API, 16	data_fmt, 42
QzDirection_T	direction, 42
Data Compression API, 11	huffman hdr, 42
qzEndStream	hw_buff_sz, 42
Data Compression API, 25	input_sz_thrshold, 42
gzFree	is_sensitive_mode, 42
Data Compression API, 26	Iz4s_mini_match, 42
qzGetDefaults	max_forks, 43
Data Compression API, 27	mem_type, 43
qzGetSoftwareComponentCount	polling_mode, 43
Data Compression API, 27	qzCallback, 43
qzGetSoftwareComponentVersionList	qzCallback_external, 43
Data Compression API, 28	req_cnt_thrshold, 43
qzGetStatus	strm_buff_sz, 43
Data Compression API, 29	sw backup, 43
QzHuffmanHdr_E	wait_cnt_thrshold, 44
Data Compression API, 16	qzSetDefaults
QzHuffmanHdr T	Data Compression API, 33
Data Compression API, 11	qzSetupSession
qzInit	Data Compression API, 34
Data Compression API, 30	QzSoftwareComponentType_E
qzLZ4SCallbackFn	Data Compression API, 18
Data Compression API, 12	QzSoftwareComponentType_T
qzMalloc	Data Compression API, 14
Data Compression API, 32	QzSoftwareVersionInfo S, 44
qzMemFindAddr	QzStatus_S, 44
Data Compression API, 32	algo_hw, 45
QzPollingMode_E	algo_sw, 45
Data Compression API, 18	hw_session_status, 45
QzPollingMode_T	memory_alloced, 45
Data Compression API, 14	qat_hw_count, 45
QzSession_S, 37	qat_instance_attach, 45
hw_session_stat, 38	qat_mem_drvr, 45
internal, 38	qat_service_init, 46
thd_sess_stat, 38	using_huge_pages, 46
total_in, 38	QzStatus_T
total_out, 38	Data Compression API, 14
QzSession_T	QzStream_S, 46
Data Compression API, 14	crc_32, 47
QzSessionParams_S, 38	crc_type, 47
comp_algorithm, 39	in, 47
comp_lvl, 39	in_sz, 47
data_fmt, 39	opaque, 47
uaia_iiii, oo	opaque, 47

```
out, 47
    out_sz, 47
    pending_in, 47
    pending_out, 48
    reserved, 48
QzStream T
    Data Compression API, 14
qzTeardownSession
    Data Compression API, 35
req_cnt_thrshold
    QzSessionParams S, 40
    QzSessionParamsGen3_S, 43
reserved
    QzStream_S, 48
strm_buff_sz
    QzSessionParams_S, 40
    QzSessionParamsGen3_S, 43
sw_backup
    QzSessionParams_S, 40
    QzSessionParamsGen3_S, 43
thd_sess_stat
    QzSession_S, 38
ThreadList_S, 48
total_in
    QzSession_S, 38
total_out
    QzSession_S, 38
using_huge_pages
    QzStatus_S, 46
wait_cnt_thrshold
    QzSessionParams_S, 40
    QzSessionParamsGen3_S, 44
```