QATzip 1.2.0

Generated by Doxygen 1.9.7

1 Module Index 1
1.1 Modules
2 Class Index
2.1 Class List
3 File Index 5
3.1 File List
4 Module Documentation 7
4.1 Data Compression API
4.1.1 Detailed Description
4.1.2 Macro Definition Documentation
4.1.2.1 QATZIP_API_VERSION_NUM_MAJOR
4.1.2.2 QATZIP_API_VERSION_NUM_MINOR
4.1.2.3 QZ_MAX_STRING_LENGTH
4.1.2.4 QZ_OK
4.1.2.5 QZ_SKID_PAD_SZ
4.1.2.6 QZ_SW_BACKUP_BIT_POSITION
4.1.2.7 QZ_SW_EXECUTION_BIT
4.1.3 Typedef Documentation
4.1.3.1 PinMem_T
4.1.3.2 QzCrc64Config T
4.1.3.3 QzCrcType_T
4.1.3.4 QzDataFormat_T
4.1.3.5 QzDirection_T
4.1.3.6 QzHuffmanHdr_T
4.1.3.7 gzLZ4SCallbackFn
4.1.3.8 QzMetadataBlob_T
4.1.3.9 QzPollingMode_T
4.1.3.10 QzSession_T
4.1.3.11 QzSessionParams T
4.1.3.12 QzSoftwareComponentType_T
4.1.3.13 QzStatus_T
4.1.3.14 QzStream_T
4.1.4 Enumeration Type Documentation
4.1.4.1 PinMem E
4.1.4.3 QzDataFormat_E
4.1.4.4 QzDirection_E
4.1.4.5 QzHuffmanHdr_E
4.1.4.6 QzPollingMode_E
4.1.4.7 QzSoftwareComponentType_E

4.1.5 Function Documentation	ıc
4.1.5.1 qzAllocateMetadata()	18
4.1.5.2 qzClose()	19
4.1.5.3 qzCompress()	19
4.1.5.4 qzCompressCrc()	21
4.1.5.5 qzCompressStream()	22
4.1.5.6 qzCompressWithMetadataExt()	23
4.1.5.7 qzDecompress()	25
4.1.5.8 qzDecompressCrc()	26
4.1.5.9 qzDecompressStream()	27
4.1.5.10 qzDecompressWithMetadataExt()	28
4.1.5.11 qzEndStream()	29
4.1.5.12 qzFree()	30
4.1.5.13 qzFreeMetadata()	31
4.1.5.14 qzGetDefaults()	32
4.1.5.15 qzGetSessionCrc64Config()	32
4.1.5.16 qzGetSoftwareComponentCount()	33
4.1.5.17 qzGetSoftwareComponentVersionList()	34
4.1.5.18 qzGetStatus()	35
4.1.5.19 qzlnit()	37
4.1.5.20 qzMalloc()	38
4.1.5.21 qzMemFindAddr()	39
4.1.5.22 qzMetadataBlockRead()	39
4.1.5.23 qzMetadataBlockWrite()	41
4.1.5.24 qzSetDefaults()	42
4.1.5.25 qzSetSessionCrc64Config()	43
4.1.5.26 qzSetupSession()	43
4.1.5.27 qzTeardownSession()	45
4.2 debug API	46
	47
	47
- to the second 2 to the secon	47
The state of the s	47
	48
	48
F-7	48
	48
	48
	48
-	48
5.3.1 Detailed Description	49

5.3.2 Member Data Documentation	. 49
5.3.2.1 hw_session_stat	. 49
5.3.2.2 internal	. 49
5.3.2.3 thd_sess_stat	. 49
5.3.2.4 total_in	. 49
5.3.2.5 total_out	. 49
5.4 QzSessionParams_S Struct Reference	. 49
5.4.1 Detailed Description	. 50
5.4.2 Member Data Documentation	. 50
5.4.2.1 comp_algorithm	. 50
5.4.2.2 comp_lvl	. 50
5.4.2.3 data_fmt	. 50
5.4.2.4 direction	. 50
5.4.2.5 huffman_hdr	. 51
5.4.2.6 hw_buff_sz	. 51
5.4.2.7 input_sz_thrshold	. 51
5.4.2.8 max_forks	. 51
5.4.2.9 req_cnt_thrshold	. 51
5.4.2.10 strm_buff_sz	. 51
5.4.2.11 sw_backup	. 51
5.4.2.12 wait_cnt_thrshold	. 52
5.5 QzSessionParamsCommon_S Struct Reference	. 52
5.5.1 Member Data Documentation	. 52
5.5.1.1 comp_algorithm	. 52
5.5.1.2 comp_lvl	. 52
5.5.1.3 direction	. 52
5.5.1.4 hw_buff_sz	. 53
5.5.1.5 input_sz_thrshold	. 53
5.5.1.6 is_sensitive_mode	. 53
5.5.1.7 max_forks	. 53
5.5.1.8 polling_mode	. 53
5.5.1.9 req_cnt_thrshold	. 53
5.5.1.10 strm_buff_sz	. 53
5.5.1.11 sw_backup	. 54
5.5.1.12 wait_cnt_thrshold	. 54
5.6 QzSessionParamsDeflate_S Struct Reference	. 54
5.6.1 Member Data Documentation	. 54
5.6.1.1 data_fmt	. 54
5.6.1.2 huffman_hdr	. 54
5.7 QzSessionParamsLZ4_S Struct Reference	. 55
5.8 QzSessionParamsLZ4S_S Struct Reference	. 55
5.8.1 Member Data Documentation	. 55

	5.8.1.1 IZ4s_mini_match	55
	5.8.1.2 qzCallback	55
	5.8.1.3 qzCallback_external	55
	5.9 QzSoftwareVersionInfo_S Struct Reference	56
	5.10 QzStatus_S Struct Reference	56
	5.10.1 Detailed Description	56
	5.10.2 Member Data Documentation	56
	5.10.2.1 algo_hw	56
	5.10.2.2 algo_sw	57
	5.10.2.3 hw_session_status	57
	5.10.2.4 memory_alloced	57
	5.10.2.5 qat_hw_count	57
	5.10.2.6 qat_instance_attach	57
	5.10.2.7 qat_mem_drvr	57
	5.10.2.8 qat_service_init	57
	5.10.2.9 using_huge_pages	58
	5.11 QzStream_S Struct Reference	58
	5.11.1 Detailed Description	58
	5.11.2 Member Data Documentation	58
	5.11.2.1 crc_32	58
	5.11.2.2 crc_type	58
	5.11.2.3 in	59
	5.11.2.4 in_sz	59
	5.11.2.5 opaque	59
	5.11.2.6 out	59
	5.11.2.7 out_sz	59
	5.11.2.8 pending_in	59
	5.11.2.9 pending_out	59
	5.11.2.10 reserved	60
	5.12 ThreadList_S Struct Reference	60
6 I	File Documentation	61
	6.1 applications.qat.shims.qatzip.qatzip/include/qatzip.h File Reference	61
	6.1.1 Macro Definition Documentation	65
	6.1.1.1 QATZIP API	65
	6.1.1.2 QATZIP_API_VERSION	65
	6.1.1.3 QZ_BUF_ERROR	65
	6.1.1.4 QZ_DATA_ERROR	65
	6.1.1.5 QZ_DEFLATE	65
	6.1.1.6 QZ_DISABLE_SOFTWARE_BACKUP	66
	6.1.1.7 QZ_DISABLE_SOFTWARE_ONLY_EXECUTION	66
	6.1.1.8 QZ_DUPLICATE	66
		-

79

	6.1.1.9 QZ_ENABLE_SOFTWARE_BACKUP	66
	6.1.1.10 QZ_ENABLE_SOFTWARE_ONLY_EXECUTION	66
	6.1.1.11 QZ_FAIL	66
	6.1.1.12 QZ_FORCE_SW	66
	6.1.1.13 QZ_INTEG	67
	6.1.1.14 QZ_LOW_DEST_MEM	67
	6.1.1.15 QZ_LOW_MEM	67
	6.1.1.16 QZ_METADATA_OVERFLOW	67
	6.1.1.17 QZ_NO_HW	67
	6.1.1.18 QZ_NO_INST_ATTACH	67
	6.1.1.19 QZ_NO_MDRV	67
	6.1.1.20 QZ_NO_SW_AVAIL	67
	6.1.1.21 QZ_NONE	68
	6.1.1.22 QZ_NOSW_LOW_MEM	68
	6.1.1.23 QZ_NOSW_NO_HW	68
	6.1.1.24 QZ_NOSW_NO_INST_ATTACH	68
	6.1.1.25 QZ_NOSW_NO_MDRV	68
	6.1.1.26 QZ_NOSW_UNSUPPORTED_FMT	68
	6.1.1.27 QZ_NOT_SUPPORTED	68
	6.1.1.28 QZ_OUT_OF_RANGE	68
	6.1.1.29 QZ_PARAMS	69
	6.1.1.30 QZ_POST_PROCESS_ERROR	69
	6.1.1.31 QZ_TIMEOUT	69
	6.1.1.32 QZ_UNSUPPORTED_FMT	69
6.2 qatzip.h		69
6.3 applicatio	ns.qat.shims.qatzip.qatzip/include/qz_utils.h File Reference	75
6.4 qz_utils.h		76

Index

Chapter 1

Module Index

1.1 Modules

Here		11: - 4	- 4 - 11		
Here	าร ล	IIST	ot all	moc	HILLES

Data Compression API												 				 				7
debug API												 				 				46

2 Module Index

Chapter 2

Class Index

2.1 Class List

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

QatThread_S	47
QzCrc64Config_S	47
QzSession_S	48
QzSessionParams_S	49
QzSessionParamsCommon_S	52
QzSessionParamsDeflate_S	54
QzSessionParamsLZ4_S	55
QzSessionParamsLZ4S_S	55
QzSoftwareVersionInfo_S	56
QzStatus_S	56
QzStream_S	58
ThreadList S	60

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 .											61
applications.qat.shims.qatzip.qatzip/include/qz_utils.h									 		75

6 File Index

Chapter 4

Module Documentation

4.1 Data Compression API

Classes

- struct QzSessionParams S
- struct QzSession S
- struct QzStatus_S
- struct QzCrc64Config_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 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 QzCrc64Config_S QzCrc64Config_T
- typedef void * QzMetadataBlob_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 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 qzInit (QzSession_T *sess, unsigned char sw_backup)
- QATZIP API int qzSetupSession (QzSession T *sess, QzSessionParams 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 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 qzCompressWithMetadataExt (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, QzMetadataBlob_T *metadata, uint32_t hw_buff_sz_override, uint32_t comp_thrshold)
- 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 qzDecompressCrc (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest len, unsigned long *crc)
- QATZIP_API int qzDecompressWithMetadataExt (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest_len, uint64_t *ext_rc, QzMetadataBlob_T *metadata, uint32_t hw_buff_sz_override)
- 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 qzSetDefaults (QzSessionParams T *defaults)
- QATZIP_API int qzGetDefaults (QzSessionParams_T *defaults)
- QATZIP API void * qzMalloc (size t sz, int numa, int force pinned)
- QATZIP API int gzAllocateMetadata (QzMetadataBlob T *metadata, size t data size, uint32 t hw buff sz)
- QATZIP_API void qzFree (void *m)
- QATZIP API int gzFreeMetadata (QzMetadataBlob T metadata)
- 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)
- QATZIP_API int qzGetSessionCrc64Config (QzSession_T *sess, QzCrc64Config_T *crc64_config)
- QATZIP API int azSetSessionCrc64Config (QzSession T *sess, QzCrc64Config T *crc64 config)
- QATZIP_API int qzMetadataBlockRead (uint32_t block_num, QzMetadataBlob_T metadata, uint32_←
 t *block offset, uint32 t *block size, uint32 t *block flags, uint32 t *block hash)
- QATZIP_API int qzMetadataBlockWrite (uint32_t block_num, QzMetadataBlob_T metadata, uint32_
 t *block_offset, uint32_t *block_size, uint32_t *block_flags, uint32_t *block_hash)

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 QzCrc64Config_T

typedef struct QzCrc64Config_S QzCrc64Config_T

QATzip CRC64 configuration structure

@description This structure contains data relating to configuration of the sessions CRC64 functionality. Session defaults to using ECMA-182 Normal on creation.

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

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 gzlnit.

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
- 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 QzMetadataBlob_T

```
typedef void* QzMetadataBlob_T
```

QATzip pointer to opaque metadata.

@description The opaque pointer to metadata.

4.1.3.9 QzPollingMode_T

```
typedef enum QzPollingMode_E QzPollingMode_T
```

Supported polling mode

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

4.1.3.10 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.

4.1.3.11 QzSessionParams_T

```
typedef struct QzSessionParams_S QzSessionParams_T
```

QATzip Session Initialization parameters

@description This structure contains data for initializing a session.

4.1.3.12 QzSoftwareComponentType_T

```
typedef enum QzSoftwareComponentType_E QzSoftwareComponentType_T
```

Software Component type

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

4.1.3.13 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.

4.1.3.14 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 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

4.1.4.3 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

4.1.4.4 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

4.1.4.5 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 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
- 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

4.1.4.6 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

4.1.4.7 QzSoftwareComponentType_E

```
\verb"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 qzAllocateMetadata()

Allocate memory for metadata.

@description Allocate memory for metadata. The function takes the size of entire input buffer and the data size at which individual block will be compressed. These parameters will be used to calculate and allocate required memory for metadata.

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

Parameters

in,	out	metadata	Pointer to opaque metadata.
in		data_size	Size of uncompressed buffer.
in		hw_buff_sz	Data size at which individual block will be compressed.

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*metadata is NULL, or data_size is 0, or data_size is greater than 1GB, or incorrect hw_buff_sz.

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.2 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

4.1.5.3 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

ec			

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.4 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 CRC32 or CRC64 checksum for 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 or qzCompressCrc64Ext 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

4.1.5.5 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

4.1.5.6 qzCompressWithMetadataExt()

```
QATZIP_API int qzCompressWithMetadataExt (
    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,
    QzMetadataBlob_T * metadata,
    uint32_t hw_buff_sz_override,
    uint32_t comp_thrshold )
```

Compress a buffer and write metadata for each compressed block into the opaque metadata structure.

@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.

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.

The metadata for each compressed block will be written into the opaque metadata structure specified as function param metadata.

comp_thrshold specifies compression threshold of a block. If compressed size of the block is > comp_thrshold, the compression function shall copy the uncompressed data to the output buffer and set the size of the block in the metadata to the size of the uncompressed block. If the compressed size of the block is <= comp_thrshold, the compressed data will be copied to the output buffer and the compressed size will be set in the metadata.

hw_buff_sz_override specifies the data size to be used for the each compression operation. It overrides the hw—buff_sz parameter specified at session creation. If 0 is provided for this parameter, then the hw_buff_sz specified at session creation will be used. Memory for the opaque metadata structure should be allocated based on the hw_buff_sz_override that is used for the compression operation.

@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	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.
in,out	metadata	Pointer to opaque metadata.
in	hw_buff_sz_override	Data size to be used for compression.
in	comp_thrshold	Compressed block threshold.

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed
QZ_PARAMS	*sess or metadata is NULL or Member of params is invalid,
	hw_buff_sz_override is invalid data size.
QZ_METADATA_OVERFLOW	Unable to populate metadata due to insufficient memory allocated.
QZ_NOT_SUPPORTED	Compression with metadata is not supported with given algorithm or format.
QZ_NOSW_NO_HW	Function did not find an installed kernel driver or software provider.
QZ_NOSW_NO_INST_ATTACH	No instance available.

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.7 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

4.1.5.8 qzDecompressCrc()

Decompress a buffer and return the CRC checksum

@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.

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

@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	crc	Pointer to CRC32 or CRC64 checksum buffer
in,out	ext_rc	qzDecompressCrcExt or qzDecompressCrc64Ext 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

4.1.5.9 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)	
ſ	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

4.1.5.10 qzDecompressWithMetadataExt()

```
QATZIP_API int qzDecompressWithMetadataExt (
    QzSession_T * sess,
    const unsigned char * src,
    unsigned int * src_len,
    unsigned char * dest,
    unsigned int * dest_len,
    uint64_t * ext_rc,
    QzMetadataBlob_T * metadata,
    uint32_t hw_buff_sz_override )
```

Decompress a buffer with metadata.

@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 content of *sess - then this function will attempt to set up a session using gzInit and gzSetupSession.

The metadata function parameter specifies metadata of compressed file which can be used for regular or parallel decompression.

hw_buff_sz_override specifies the data size to be used for the each decompression operation. It overrides the hw buff_sz parameter specified at session creation. If 0 is provided for this parameter, then the hw_buff_sz specified at session creation will be used. Memory for the opaque metadata structure should be allocated based on the hw_buff_sz_override that is used for the compression operation.

@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	If not NULL, ext_rc points 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.
in	metadata	Pointer to opaque metadata.
in	hw_buff_sz_override	Expected size of decompressed block.

Return values

QZ_OK	Function executed successfully.
QZ_FAIL	Function did not succeed.
QZ_PARAMS	*sess or metadata is NULL or Member of params is invalid,
	hw_buff_sz_override is invalid data size.
QZ_METADATA_OVERFLOW	Unable to populate metadata due to insufficient memory allocated.
QZ_NOT_SUPPORTED	Decompression with metadata is not supported with given algorithm or
	format.
QZ_NOSW_NO_HW	Function did not find an installed kernel driver or software provider.
QZ_NOSW_NO_INST_ATTACH	No instance available.

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.11 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

4.1.5.12 qzFree()

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	m	Memory address to be freed

Precondition

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.13 qzFreeMetadata()

Free memory allocated for metadata.

@description Free memory allocated for metadata.

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

Parameters

in	metadata	Pointer to opaque metadata.
----	----------	-----------------------------

Return values

QZ_OK	Function executed successfully.
QZ_FAIL	Function did not succeed.
QZ_PARAMS	metadata is NULL.

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

4.1.5.14 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

4.1.5.15 qzGetSessionCrc64Config()

Requests the CRC64 configuration of the provided session

@description This function populates crc64_config with the CRC64 configuration details of sess. This function has a dependency on invoking a setup session function first.

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

Parameters

in	sess	Session handle (pointer to opaque instance and session data)
out	crc64_config	Configuration for CRC 64 generation.

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Session was not setup
QZ_PARAMS	*sess or *crc64_config is NULL

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.16 qzGetSoftwareComponentCount()

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

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

Return values

QZ_OK	Function executed successfully

Return values

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

4.1.5.17 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

Return values

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

4.1.5.18 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 |
```

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 available, 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 be	
QZ_PARAMS	*status is NULL

Precond	ition
---------	-------

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

4.1.5.19 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	ss Session handle (pointer to opaque instance and session data.)	
Ī	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

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.20 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

4.1.5.21 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

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

4.1.5.22 qzMetadataBlockRead()

Read metadata parameters.

@description This function reads metadata information for the block specified by the function param block_num.

block_offset returns offset value in bytes from the previous compressed block of the compressed data.

block_size returns the block size in bytes of the compressed block. Some blocks may be uncompressed if size > threshold as specified during compression and the size returned will reflect the same.

block_flags returns the value 1 if the data is compressed and 0 if the data is not compressed.

block_hash returns the xxHash value of the plain text of the hw_buff_sz payload sent for compression operation.

If NULL is specified for any of the metadata parameters (block_offset, block_size, block_flags, block_hash) reading the parameter value will be ignored.

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

Parameters

in	block_num	Block number of which metadata information should be read.
in	metadata	Pointer to opaque metadata.
in,out	block_offset	Pointer to the block offset value.
in,out	block_size	Pointer to the block size value.
in,out	block_flags	Pointer to the block flags value.
in,out	block_hash	Pointer to the block xxHash value.

Return values

QZ_OK	Function executed successfully.
QZ_FAIL	Function did not succeed.
QZ_PARAMS	Metadata is NULL.
QZ_OUT_OF_RANGE	block_num specified is out of range.

Uro	non	MITI	An.
FIC	con	ulli	UII

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

4.1.5.23 qzMetadataBlockWrite()

Write metadata parameters.

@description This function writes metadata information for the block specified by the function param block_num.

block_offset writes offset value in bytes from the previous compressed block of the compressed data.

block_size writes the block size in bytes of the compressed block.

block_flags causes the metadata to indicate the data is compressed if passed a value of 1 and indicates uncompressed if value passed is zero (0).

block_hash writes the xxHash value of the plain text of the hw_buff_sz payload sent for compression operation.

If NULL is specified for any of the metadata parameters (block_offset, block_size, block_flags, block_hash) writing the parameter value into metadata will be ignored.

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

Parameters

in	block_num	Block number into which metadata information should be written.
in,out	metadata	Pointer to opaque metadata.
in	block_offset	Pointer to the block offset value.
in	block_size	Pointer to the block size value.
in	block_flags	Pointer to the block flags value.
in	block_hash	Pointer to the block xxHash value.

Return values

QZ_OK	Function executed successfully.
QZ_FAIL	Function did not succeed.
QZ_PARAMS	Metadata is NULL.
QZ_OUT_OF_RANGE	block_num specified is out of range.

Precondition

None

Postcondition

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.24 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

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

4.1.5.25 qzSetSessionCrc64Config()

Sets the CRC64 configuration of the provided session with a user defined set of parameters.

@description This function populates the CRC64 configuration details of sess using the paramaters provided in crc64_config. This function has a dependency on invoking a setup session function first.

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

Parameters

in	sess	sess Session handle (pointer to opaque instance and session dat	
out	crc64_config	Configuration for CRC 64 generation.	

Return values

QZ_OK	Function executed successfully	
QZ_FAIL	Session was not setup	
QZ_PARAMS	*sess or *crc64_config is NULL or contains invalid paramters.	

Precondition

None

Postcondition

None

Note

Only a synchronous version of this function is provided.

See also

None

4.1.5.26 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

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

4.1.5.27 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)]
----	------	--	---

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 QzCrc64Config_S Struct Reference

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

Public Attributes

- uint64_t polynomial
- uint64_t initial_value
- uint32_t reflect_in
- · uint32_t reflect_out
- uint64_t xor_out

5.2.1 Detailed Description

QATzip CRC64 configuration structure

@description This structure contains data relating to configuration of the sessions CRC64 functionality. Session defaults to using ECMA-182 Normal on creation.

48 Class Documentation

5.2.2 Member Data Documentation

5.2.2.1 initial_value

```
uint64_t QzCrc64Config_S::initial_value
```

5.2.2.2 polynomial

```
uint64_t QzCrc64Config_S::polynomial
```

Polynomial used for CRC64 calculation. Default 0x42F0E1EBA9EA3693

5.2.2.3 reflect_in

```
uint32_t QzCrc64Config_S::reflect_in
```

Reflect bit order before CRC calculation. Default 0

5.2.2.4 reflect out

```
uint32_t QzCrc64Config_S::reflect_out
```

Reflect bit order after CRC calculation. Default 0

5.2.2.5 xor_out

```
uint64_t QzCrc64Config_S::xor_out
```

Defaults to 0x0000000000000000

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

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

5.3 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

5.3.1 Detailed Description

QATzip Session opaque data storage

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

5.3.2 Member Data Documentation

5.3.2.1 hw_session_stat

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

Filled in during initialization, session startup and decompression

5.3.2.2 internal

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

Session data is opaque to outside world

5.3.2.3 thd_sess_stat

```
int QzSession_S::thd_sess_stat
```

Note process compression and decompression thread state

5.3.2.4 total_in

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

Total processed input data length in this session

5.3.2.5 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

5.4 QzSessionParams_S Struct Reference

#include <qatzip.h>

50 Class Documentation

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

5.4.1 Detailed Description

QATzip Session Initialization parameters

@description This structure contains data for initializing a session.

5.4.2 Member Data Documentation

5.4.2.1 comp_algorithm

 ${\tt unsigned\ char\ QzSessionParams_S::comp_algorithm}$

Compress/decompression algorithms

5.4.2.2 comp_lvl

unsigned int QzSessionParams_S::comp_lvl

Compression level 1 to 9

5.4.2.3 data_fmt

QzDataFormat_T QzSessionParams_S::data_fmt

Deflate, deflate with GZip or deflate with GZip ext

5.4.2.4 direction

QzDirection_T QzSessionParams_S::direction

Compress or decompress

5.4.2.5 huffman_hdr

QzHuffmanHdr_T QzSessionParams_S::huffman_hdr

Dynamic or Static Huffman headers

5.4.2.6 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

5.4.2.7 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

5.4.2.8 max_forks

unsigned int QzSessionParams_S::max_forks

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

5.4.2.9 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

5.4.2.10 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

5.4.2.11 sw_backup

unsigned char QzSessionParams_S::sw_backup

bit field defining SW configuration (see QZ_SW_* definitions)

52 Class Documentation

5.4.2.12 wait_cnt_thrshold

```
\verb"unsigned" int QzSessionParams\_S:: \verb"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 QzSessionParamsCommon S Struct Reference

Public Attributes

- QzDirection_T direction
- 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
- QzPollingMode_T polling_mode
- unsigned int is_sensitive_mode

5.5.1 Member Data Documentation

5.5.1.1 comp_algorithm

```
\verb"unsigned" char QzSessionParamsCommon\_S::comp\_algorithm"
```

Compress/decompression algorithms

5.5.1.2 comp_lvl

```
unsigned int QzSessionParamsCommon_S::comp_lvl
```

Compression level 1 to 9

5.5.1.3 direction

QzDirection_T QzSessionParamsCommon_S::direction

Compress or decompress

5.5.1.4 hw_buff_sz

 $\verb"unsigned" int QzSessionParamsCommon_S::hw_buff_sz"$

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

5.5.1.5 input_sz_thrshold

unsigned int QzSessionParamsCommon_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

5.5.1.6 is_sensitive_mode

unsigned int QzSessionParamsCommon_S::is_sensitive_mode

0 means disable sensitive mode, 1 means enable sensitive mode

5.5.1.7 max_forks

unsigned int QzSessionParamsCommon_S::max_forks

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

5.5.1.8 polling_mode

QzPollingMode_T QzSessionParamsCommon_S::polling_mode

0 means no busy polling, 1 means busy polling

5.5.1.9 req_cnt_thrshold

unsigned int QzSessionParamsCommon_S::req_cnt_thrshold

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

5.5.1.10 strm_buff_sz

 $\verb"unsigned" int QzSessionParamsCommon_S::strm_buff_sz"$

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

54 Class Documentation

5.5.1.11 sw_backup

```
\verb"unsigned" char QzSessionParamsCommon\_S::sw\_backup"
```

bit field defining SW configuration (see QZ_SW_* definitions)

5.5.1.12 wait_cnt_thrshold

```
unsigned int QzSessionParamsCommon_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

5.6 QzSessionParamsDeflate_S Struct Reference

Public Attributes

- QzSessionParamsCommon_T common_params
- QzHuffmanHdr_T huffman_hdr
- QzDataFormat_T data_fmt

5.6.1 Member Data Documentation

5.6.1.1 data_fmt

```
QzDataFormat_T QzSessionParamsDeflate_S::data_fmt
```

Deflate, deflate with GZip or deflate with GZip ext

5.6.1.2 huffman_hdr

```
QzHuffmanHdr_T QzSessionParamsDeflate_S::huffman_hdr
```

Dynamic or Static Huffman headers

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

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

5.7 QzSessionParamsLZ4_S Struct Reference

Public Attributes

• QzSessionParamsCommon_T common_params

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

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

5.8 QzSessionParamsLZ4S_S Struct Reference

Public Attributes

- QzSessionParamsCommon_T common_params
- qzLZ4SCallbackFn qzCallback
- void * qzCallback_external
- unsigned int lz4s_mini_match

5.8.1 Member Data Documentation

5.8.1.1 lz4s_mini_match

unsigned int QzSessionParamsLZ4S_S::lz4s_mini_match

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

5.8.1.2 qzCallback

qzLZ4SCallbackFn QzSessionParamsLZ4S_S::qzCallback

post processing callback for zstd compression

5.8.1.3 qzCallback external

void* QzSessionParamsLZ4S_S::qzCallback_external

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

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

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

56 Class Documentation

5.9 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

5.10 QzStatus S Struct Reference

#include <qatzip.h>

Public Attributes

- unsigned short int qat_hw_count
- unsigned char qat_service_init
- · unsigned char gat 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]

5.10.1 Detailed Description

QATzip status structure

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

5.10.2 Member Data Documentation

5.10.2.1 algo hw

unsigned char QzStatus_S::algo_hw[QZ_MAX_ALGORITHMS]

Count of hardware devices supporting algorithms

5.10.2.2 algo_sw

unsigned char QzStatus_S::algo_sw[QZ_MAX_ALGORITHMS]

Support software algorithms

5.10.2.3 hw_session_status

signed long int QzStatus_S::hw_session_status

One of QATzip Session Status

5.10.2.4 memory_alloced

unsigned long int QzStatus_S::memory_alloced

Amount of memory allocated by this thread/process

5.10.2.5 qat_hw_count

unsigned short int QzStatus_S::qat_hw_count

From PCI scan

5.10.2.6 qat_instance_attach

unsigned char QzStatus_S::qat_instance_attach

Is this thread/g_process properly attached to an Instance?

5.10.2.7 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

5.10.2.8 qat_service_init

unsigned char QzStatus_S::qat_service_init

Check if the available services have been initialized

58 Class Documentation

5.10.2.9 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

5.11 QzStream S Struct Reference

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

Public Attributes

- unsigned int in_sz
- unsigned int out_sz
- unsigned char * in
- unsigned char * out
- unsigned int pending_in
- unsigned int pending_out
- QzCrcType_T crc_type
- unsigned int crc_32
- · unsigned long long reserved
- void * opaque

5.11.1 Detailed Description

QATzip Stream data storage

@description This structure contains metadata needed for stream operation.

5.11.2 Member Data Documentation

```
5.11.2.1 crc_32
```

```
unsigned int QzStream_S::crc_32
```

Checksum value

5.11.2.2 crc_type

```
QzCrcType_T QzStream_S::crc_type
```

Checksum type in Adler, CRC32 or none

5.11.2.3 in

unsigned char* QzStream_S::in

Input data pointer set by application

5.11.2.4 in_sz

```
unsigned int QzStream_S::in_sz
```

Set by application, reset by QATzip to indicate consumed data

5.11.2.5 opaque

void* QzStream_S::opaque

Internal storage managed by QATzip

5.11.2.6 out

unsigned char* QzStream_S::out

Output data pointer set by application

5.11.2.7 out_sz

unsigned int QzStream_S::out_sz

Set by application, reset by QATzip to indicate processed data

5.11.2.8 pending_in

unsigned int QzStream_S::pending_in

Unprocessed bytes held in QATzip

5.11.2.9 pending_out

unsigned int QzStream_S::pending_out

Processed bytes held in QATzip

60 Class Documentation

5.11.2.10 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

5.12 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 QzSessionParamsCommon_S
- struct QzSessionParamsDeflate S
- struct QzSessionParamsLZ4_S
- struct QzSessionParamsLZ4S S
- struct QzSession_S
- struct QzStatus_S
- struct QzSoftwareVersionInfo S
- struct QzCrc64Config_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)

62 File Documentation

- #define QZ_NO_MDRV (12)
- #define QZ_NO_INST_ATTACH (13)
- #define QZ_LOW_MEM (14)
- #define QZ_LOW_DEST_MEM (15)
- #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 METADATA OVERFLOW (-118)
- #define QZ_OUT_OF_RANGE (-119)
- #define QZ_NOT_SUPPORTED (-200)
- #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_DEFLATE_COMP_LVL_MAXIMUM_Gen3 (12)
- #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 &= ~(1 << QZ_SW_BACKUP_BIT_POSI

- #define QZ_DISABLE_SOFTWARE_ONLY_EXECUTION(_BackupVariable) (_BackupVariable &= ~(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 QzPollingMode_E QzPollingMode_T
- typedef enum QzCrcType_E QzCrcType_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 QzSessionParamsCommon_S QzSessionParamsCommon_T
- typedef struct QzSessionParamsDeflate_S QzSessionParamsDeflate_T
- typedef struct QzSessionParamsLZ4_S QzSessionParamsLZ4_T
- typedef struct QzSessionParamsLZ4S_S QzSessionParamsLZ4S_T
- typedef struct QzSession S QzSession T
- typedef struct QzStatus S QzStatus T
- typedef struct QzSoftwareVersionInfo_S QzSoftwareVersionInfo_T
- typedef struct QzCrc64Config_S QzCrc64Config_T
- typedef void * QzMetadataBlob_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 QzSoftwareComponentType E {
 - QZ_COMPONENT_FIRMWARE = 0 , QZ_COMPONENT_KERNEL_DRIVER , QZ_COMPONENT_USER ← DRIVER , QZ_COMPONENT_QATZIP_API , QZ_COMPONENT_SOFTWARE_PROVIDER }

64 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 qzSetupSessionDeflate (QzSession T *sess, QzSessionParamsDeflate T *params)
- QATZIP_API int qzSetupSessionLZ4 (QzSession_T *sess, QzSessionParamsLZ4_T *params)
- QATZIP API int qzSetupSessionLZ4S (QzSession_T *sess, QzSessionParamsLZ4S_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 qzCompressWithMetadataExt (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, QzMetadataBlob_T *metadata, uint32 t hw buff sz override, uint32 t comp thrshold)
- 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 qzDecompressCrc (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest_len, unsigned long *crc)
- QATZIP_API int qzDecompressCrcExt (QzSession_T *sess, const unsigned char *src, unsigned int *src
 — len, unsigned char *dest, unsigned int *dest_len, unsigned long *crc, uint64_t *ext_rc)
- QATZIP_API int qzDecompressCrc64 (QzSession_T *sess, const unsigned char *src, unsigned int *src_ ← len, unsigned char *dest, unsigned int *dest_len, uint64_t *crc)
- QATZIP_API int qzDecompressCrc64Ext (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest_len, uint64_t *crc, uint64_t *ext_rc)
- QATZIP_API int qzDecompressWithMetadataExt (QzSession_T *sess, const unsigned char *src, unsigned int *src_len, unsigned char *dest, unsigned int *dest_len, uint64_t *ext_rc, QzMetadataBlob_T *metadata, uint32_t hw_buff_sz_override)
- 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 qzSetDefaultsDeflate (QzSessionParamsDeflate_T *defaults)
- QATZIP_API int qzSetDefaultsLZ4 (QzSessionParamsLZ4_T *defaults)
- QATZIP_API int qzSetDefaultsLZ4S (QzSessionParamsLZ4S_T *defaults)
- QATZIP_API int qzGetDefaults (QzSessionParams_T *defaults)
- QATZIP API int qzGetDefaultsDeflate (QzSessionParamsDeflate T *defaults)
- QATZIP API int qzGetDefaultsLZ4 (QzSessionParamsLZ4 T *defaults)
- QATZIP API int qzGetDefaultsLZ4S (QzSessionParamsLZ4S T *defaults)
- QATZIP API void * qzMalloc (size t sz, int numa, int force pinned)
- QATZIP_API int qzAllocateMetadata (QzMetadataBlob_T *metadata, size_t data_size, uint32_t hw_buff_sz)
- QATZIP_API void qzFree (void *m)
- QATZIP_API int qzFreeMetadata (QzMetadataBlob_T metadata)
- 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)
- QATZIP_API int qzGetSessionCrc64Config (QzSession_T *sess, QzCrc64Config_T *crc64_config)
- QATZIP_API int qzSetSessionCrc64Config (QzSession_T *sess, QzCrc64Config_T *crc64_config)
- QATZIP_API int qzMetadataBlockRead (uint32_t block_num, QzMetadataBlob_T metadata, uint32_
 t *block_offset, uint32_t *block_size, uint32_t *block_flags, uint32_t *block_hash)
- QATZIP_API int qzMetadataBlockWrite (uint32_t block_num, QzMetadataBlob_T metadata, uint32_
 t *block_offset, uint32_t *block_size, uint32_t *block_flags, uint32_t *block_hash)

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

```
#define QZ_DISABLE_SOFTWARE_BACKUP(  \_BackupVariable \ ) \quad (\_BackupVariable \ \&= \ \sim (1 << QZ\_SW\_BACKUP\_BIT\_POSITION))
```

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

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_METADATA_OVERFLOW

```
#define QZ_METADATA_OVERFLOW (-118)
```

Insufficent memory allocated for metadata

6.1.1.17 QZ_NO_HW

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

Using SW: No QAT HW detected

6.1.1.18 QZ_NO_INST_ATTACH

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

Using SW: Could not attach to an instance

6.1.1.19 QZ_NO_MDRV

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

Using SW: No memory driver detected

6.1.1.20 QZ_NO_SW_AVAIL

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

Session may require software, but no software is available

6.1.1.21 QZ_NONE

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

Device uninitialized

6.1.1.22 QZ_NOSW_LOW_MEM

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

Not using SW: not enough pinned memory

6.1.1.23 QZ_NOSW_NO_HW

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

Not using SW: No QAT HW detected

6.1.1.24 QZ_NOSW_NO_INST_ATTACH

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

Not using SW: Could not attach to instance

6.1.1.25 QZ_NOSW_NO_MDRV

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

Not using SW: No memory driver detected

6.1.1.26 QZ_NOSW_UNSUPPORTED_FMT

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

Not using SW: QAT device does not support data format

6.1.1.27 QZ_NOT_SUPPORTED

```
#define QZ_NOT_SUPPORTED (-200)
```

Request not supported

6.1.1.28 QZ_OUT_OF_RANGE

```
#define QZ_OUT_OF_RANGE (-119)
```

Metadata block_num specified is out of range

6.2 qatzip.h

6.1.1.29 QZ_PARAMS

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

Invalid parameter in function call

6.1.1.30 QZ_POST_PROCESS_ERROR

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

Using post process: post process callback encountered an error

6.1.1.31 **QZ_TIMEOUT**

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

Operation timed out

6.1.1.32 QZ_UNSUPPORTED_FMT

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

Using SW: QAT device does not support data format

6.2 qatzip.h

```
Go to the documentation of this file.
```

```
00001 /
00002
00003
           BSD LICENSE
00004
00005
           Copyright(c) 2007-2023 Intel Corporation. All rights reserved.
00006
           All rights reserved.
00007
00008
           Redistribution and use in source and binary forms, with or without
00009
           modification, are permitted provided that the following conditions
00010
00011
00012 *
             \star Redistributions of source code must retain the above copyright
00013
               notice, this list of conditions and the following disclaimer.
00014
             * Redistributions in binary form must reproduce the above copyright
00015
              notice, this list of conditions and the following disclaimer in
00016
               the documentation and/or other materials provided with the
00017
00018
             \star Neither the name of Intel Corporation nor the names of its
00019
               contributors may be used to endorse or promote products derived
00020
               from this software without specific prior written permission.
00021
00022
           THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
00023
           "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
00024
           LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
00025
           A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
           OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
00026
           SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
00027
00028
00029
           DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
00030
           THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
           (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
00031
00032
           OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
00033
00034
```

```
00050 #ifndef _QATZIP_H
00051 #define _QATZIP_H
00052
00053 #ifdef __cplusplus
00054 extern"C" {
00055 #endif
00057 #include <string.h>
00058 #include <stdint.h>
00059
00071 #define OATZIP API VERSION NUM MAJOR (2)
00072
00083 #define QATZIP_API_VERSION_NUM_MINOR (3)
00084
00085 /\star Define a macro as an integer to test \star/
00086 #define QATZIP_API_VERSION (QATZIP_API_VERSION_NUM_MAJOR * 10000 + QATZIP_API_VERSION_NUM_MINOR * 100)
00088
00095 #if defined QATZIP_LINK_DLL
00096 #define QATZIP_API __declspec(dllimport)
00097 #elif defined QATZIP_BUILD_DLL
00098 #define QATZIP_API __declspec(dllexport)
00099 #else
00100 #define QATZIP_API
00101 #endif
00102
00179 typedef enum QzHuffmanHdr_E {
00180 QZ_DYNAMIC_HDR = 0,
00182
          OZ STATIC HDR
00184 } QzHuffmanHdr_T;
00185
00196 typedef enum PinMem_E {
00197 COMMON_MEM = 0,
00199 PINNED_MEM
00201 } PinMem_T;
00202
00214 typedef enum QzDirection_E {
        QZ_DIR_COMPRESS = 0,
QZ_DIR_DECOMPRESS,
00217
00219
          QZ_DIR_BOTH
00221 } QzDirection_T;
00222
00235 typedef enum QzDataFormat_E {
00236 QZ_DEFLATE_4B = 0,
00238 QZ_DEFLATE_GZIP,
00240
          QZ_DEFLATE_GZIP_EXT,
        QZ_DEFLATE_RAW,
00242
00244
          OZ FMT NUM
00245 } QzDataFormat_T;
00246
00257 typedef enum QzPollingMode_E {
      QZ_PERIODICAL_POLLING = 0,
00258
00260
          QZ_BUSY_POLLING,
00262 } QzPollingMode_T;
00263
00274 typedef enum QzCrcType_E {
        QZ\_CRC32 = 0,
00277
          QZ_ADLER,
00279
          NONE
00281 } QzCrcType_T;
00282
00293 typedef enum QzSoftwareComponentType_E {
00294 QZ_COMPONENT_FIRMWARE = 0,
00295
          QZ_COMPONENT_KERNEL_DRIVER,
        QZ_COMPONENT_USER_DRIVER,
QZ_COMPONENT_QATZIP_API,
00296
00297
00298
          OZ COMPONENT SOFTWARE PROVIDER
00299 } QzSoftwareComponentType_T;
00300
00311 #define QZ_OK
00313 #define QZ_DUPLICATE
                                           (1)
00315 #define QZ_FORCE_SW
                                           (2)
00317 #define QZ_PARAMS
                                          (-1)
00319 #define QZ_FAIL
                                          (-2)
00321 #define QZ_BUF_ERROR
                                           (-3)
                                           (-4)
00323 #define QZ_DATA_ERROR
00325 #define QZ_TIMEOUT
                                           (-5)
00327 #define QZ_INTEG
                                           (-100)
00329 #define QZ_NO_HW
00331 #define QZ_NO_MDRV
                                           (11)
                                           (12)
00333 #define QZ_NO_INST_ATTACH
                                           (13)
00335 #define QZ_LOW_MEM
00337 #define QZ_LOW_DEST_MEM
                                           (14)
                                           (15)
00339 #define QZ_UNSUPPORTED_FMT
                                           (16)
00341 #define QZ_NONE
                                           (100)
00343 #define QZ_NOSW_NO_HW
00345 #define QZ_NOSW_NO_MDRV
                                           (-101)
                                           (-102)
```

6.2 qatzip.h

```
00347 #define QZ_NOSW_NO_INST_ATTACH (-103)
00349 #define QZ_NOSW_LOW_MEM
00351 #define QZ_NO_SW_AVAIL
                                        (-105)
00353 #define QZ_NOSW_UNSUPPORTED_FMT (-116)
00355 #define QZ_POST_PROCESS_ERROR (-117)
00357 #define QZ_METADATA_OVERFLOW (-118)
00359 #define QZ_OUT_OF_RANGE
00361 #define QZ_NOT_SUPPORTED
00364 #define QZ_MAX_ALGORITHMS ((int)255)
00365 #define QZ_DEFLATE
                                   ((unsigned char)8)
                                  ((unsigned char)'4')
00368 #define QZ_LZ4
                                  ((unsigned char)'s')
00369 #define OZ LZ4s
00370 #define QZ_ZSTD
                                  ((unsigned char)'Z')
00371
00372 #ifndef MIN
00373 #define MIN(a,b) (((a)<(b))?(a):(b))
00374 #endif
00375 #ifdef __linux__
00376 #define QZ_MEMCPY(dest, src, dest_sz, src_sz) \
              memcpy((void *) (dest), (void *) (src), (size_t)MIN(dest_sz, src_sz))
00377
00378 #endif
00379 #ifdef _WIN64
00380 #define QZ_MEMCPY(dest, src, dest_sz, src_sz)
              memcpy_s((void *)(dest), dest_sz, (void *) (src), MIN(dest_sz, src_sz))
00381
00382 #endif
00447 typedef int (*qzLZ4SCallbackFn) (void *external, const unsigned char *src,
00448
                                        unsigned int *src_len, unsigned char *dest,
00449
                                        unsigned int *dest_len, int *ExtStatus);
00450
00460 typedef struct QzSessionParams_S {
00461
          QzHuffmanHdr_T huffman_hdr;
00463
          QzDirection_T direction;
00465
          QzDataFormat_T data_fmt;
00467
          unsigned int comp_lvl;
00469
          unsigned char comp_algorithm;
00471
          unsigned int max forks;
          unsigned char sw_backup;
00476
          unsigned int hw_buff_sz;
00479
          unsigned int strm_buff_sz;
00482
          unsigned int input_sz_thrshold;
00487
          unsigned int req_cnt_thrshold;
00490
          unsigned int wait cnt thrshold;
00493 #ifdef ERR_INJECTION
          void *fbError;
00494
00495
          void *fbErrorCurr;
00496
          /\star Linked list for simulated errors from HW \star/
00497 #endif
00498 } OzSessionParams T:
00499
00500 typedef struct QzSessionParamsCommon_S {
00501
          QzDirection_T direction;
00503
          unsigned int comp_lvl;
00505
          unsigned char comp_algorithm;
00507
          unsigned int max_forks;
00510
          unsigned char sw backup;
          unsigned int hw_buff_sz;
          unsigned int strm_buff_sz;
00515
00518
          unsigned int input_sz_thrshold;
00523
          unsigned int req_cnt_thrshold;
00526
          unsigned int wait cnt thrshold;
00529
          QzPollingMode_T polling_mode;
00531 unsigned int is_sensitive_mode;
00533 #ifdef ERR_INJECTION
00534
          void *fbError;
00535
          void *fbErrorCurr;
00536
          /\star Linked list for simulated errors from HW \star/
00537 #endif
00538 } QzSessionParamsCommon_T;
00540 typedef struct QzSessionParamsDeflate_S {
00541
         QzSessionParamsCommon_T common_params;
          QzHuffmanHdr_T huffman_hdr;
QzDataFormat_T data_fmt;
00542
00544
00546 } OzSessionParamsDeflate T;
00547
00548 typedef struct QzSessionParamsLZ4_S {
00549
          QzSessionParamsCommon_T common_params;
00550 } QzSessionParamsLZ4_T;
00551
00552 typedef struct QzSessionParamsLZ4S_S {
          QzSessionParamsCommon_T common_params;
00554
          qzLZ4SCallbackFn qzCallback;
00556
          void *qzCallback_external;
00559
          unsigned int lz4s_mini_match;
00561 } QzSessionParamsLZ4S_T;
00562
```

```
00563 #define QZ_HUFF_HDR_DEFAULT
                                                 OZ DYNAMIC HDR
00564 #define QZ_DIRECTION_DEFAULT
                                                 QZ_DIR_BOTH
00565 #define QZ_DATA_FORMAT_DEFAULT
                                                 QZ_DEFLATE_GZIP_EXT
00566 #define QZ_COMP_LEVEL_DEFAULT
00567 #define QZ_COMP_ALGOL_DEFAULT
                                                 OZ DEFLATE
00568 #define QZ_POLL_SLEEP_DEFAULT
                                                 10
00569 #define QZ_MAX_FORK_DEFAULT
00570 #define QZ_SW_BACKUP_DEFAULT
00571 #define QZ_HW_BUFF_SZ
                                                  (64 * 1024)
00572 #define QZ_HW_BUFF_SZ_Gen3
                                                  (1*1024*1024)
00573 #define QZ_HW_BUFF_MIN_SZ
                                                  (1 * 1024)
00574 #define QZ_HW_BUFF_MAX_SZ
                                                  (512 * 1024)
00575 #define QZ_HW_BUFF_MAX_SZ_Gen3
                                                  (2*1024*1024*1024U)
                                                 QZ_HW_BUFF_SZ
00576 #define QZ_STRM_BUFF_SZ_DEFAULT
00577 #define QZ_STRM_BUFF_MIN_SZ
                                                  (1 * 1024)
00578 #define QZ_STRM_BUFF_MAX_SZ
                                                  (2*1024*1024 - 5*1024)
00579 #define QZ_COMP_THRESHOLD_DEFAULT 00580 #define QZ_COMP_THRESHOLD_MINIMUM
                                                 1024
                                                 128
00581 #define QZ_REQ_THRESHOLD_MINIMUM
00582 #define QZ_REO_THRESHOLD_MAXIMUM
                                                 NUM BUFF
00583 #define QZ_REQ_THRESHOLD_DEFAULT
                                                 QZ_REQ_THRESHOLD_MAXIMUM
00584 #define QZ_WAIT_CNT_THRESHOLD_DEFAULT 8
00585 #define QZ_DEFLATE_COMP_LVL_MINIMUM (1)
00586 #define QZ_DEFLATE_COMP_LVL_MAXIMUM (9)
00587 #define QZ_DEFLATE_COMP_LVL_MAXIMUM_Gen3 (12)
00588 #define QZ_LZS_COMP_LVL_MINIMUM
                                                      (1)
00589 #define QZ_LZS_COMP_LVL_MAXIMUM
00590
00606 #define QZ_SW_BACKUP_BIT_POSITION (0) 00607 #define QZ_SW_FORCESW_BIT_POSITION (1)
00608
00609 #define QZ_ENABLE_SOFTWARE_BACKUP(_BackupVariable)
                (_BackupVariable |= (1 « QZ_SW_BACKUP_BIT_POSITION))
00610
00612 #define QZ_ENABLE_SOFTWARE_ONLY_EXECUTION(_BackupVariable)
00613 (_BackupVariable |= (1 « QZ_SW_FORCESW_BIT_POSITION))
00616 #define QZ_DISABLE_SOFTWARE_BACKUP (_BackupVariable) \
00617 (_BackupVariable &= ~(1 « QZ_SW_BACKUP_BIT_POSITION))
00619 #define QZ_DISABLE_SOFTWARE_ONLY_EXECUTION (_BackupVariable) \
               (_BackupVariable &= ~(1 « QZ_SW_FORCESW_BIT_POSITION))
00640 #define QZ_SW_EXECUTION_BIT (4)
00641 #define QZ_SW_EXECUTION_MASK (1 « QZ_SW_EXECUTION_BIT)
00642 #define QZ_SW_EXECUTION(ret, ext_rc) \ 00643 (!ret && (ext_rc & QZ_SW_EXECUTION_MASK))
00644
00645 #define QZ_TIMEOUT_BIT
00646 #define QZ_TIMEOUT_MASK
                                                  (1 « QZ_TIMEOUT_BIT)
00647 #define QZ_HW_TIMEOUT(ret, ext_rc)
            (!ret && (ext_rc & QZ_TIMEOUT_MASK))
00648
00649
00650 #define QZ_POST_PROCESS_FAIL_BIT
00650 #define QZ_POST_PROCESS_FAIL_MASK (1 « QZ_POST_PROCESS_FAIL_BIT)
00652 #define QZ_POST_PROCESS_FAIL(ret, ext_rc)
00653
            (ret && (ext_rc & QZ_POST_PROCESS_FAIL_MASK))
00654
00665 typedef struct QzSession_S {
       signed long int hw_session_stat;
00666
           int thd_sess_stat;
00670
           void *internal;
00672
          unsigned long total_in;
00674
          unsigned long total_out;
00676 } QzSession_T;
00677
00688 typedef struct QzStatus_S {
       unsigned short int qat_hw_count;
00691
           unsigned char qat_service_init;
00693
           unsigned char qat_mem_drvr;
00697
           unsigned char qat_instance_attach;
00699
           unsigned long int memory_alloced;
00701
           unsigned char using_huge_pages;
           signed long int hw_session_status;
00703
00705
           unsigned char algo_sw[QZ_MAX_ALGORITHMS];
00707
           unsigned char algo_hw[QZ_MAX_ALGORITHMS];
00709 } QzStatus_T;
00710
00721 #define QZ_MAX_STRING_LENGTH 64
00722 typedef struct QzSoftwareVersionInfo_S {
           QzSoftwareComponentType_T component_type;
00723
00724
           unsigned char component_name[QZ_MAX_STRING_LENGTH];
00725
           unsigned int major_version;
00726
           unsigned int minor_version;
00727
           unsigned int patch_version;
00728
           unsigned int build_number;
00729
           unsigned char reserved[52];
00730 } QzSoftwareVersionInfo_T;
00731
00742 typedef struct QzCrc64Config_S {
00743
          uint64_t polynomial;
```

6.2 qatzip.h 73

```
00745
          uint64_t initial_value;
00747
          uint32_t reflect_in;
00749
          uint32_t reflect_out;
00751
          uint64_t xor_out;
00753 } QzCrc64Config_T;
00754
00764 typedef void *QzMetadataBlob_T;
00765
00842 QATZIP_API int qzInit(QzSession_T *sess, unsigned char sw_backup);
00843
00913 QATZIP_API int qzSetupSession(QzSession_T *sess, QzSessionParams_T *params);
00914
00915 QATZIP_API int qzSetupSessionDeflate(QzSession_T *sess,
00916
                                             QzSessionParamsDeflate_T *params);
00917
00918 QATZIP_API int qzSetupSessionLZ4(QzSession_T *sess,
00919
                                         OzSessionParamsLZ4 T *params);
00920
00921 QATZIP_API int qzSetupSessionLZ4S(QzSession_T *sess,
00922
                                          QzSessionParamsLZ4S_T *params);
00923
00997 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);
00998
00999
01000
01001 QATZIP_API int qzCompressExt(QzSession_T *sess, const unsigned char *src,
01002
                                     unsigned int *src_len, unsigned char *dest,
01003
                                     unsigned int *dest_len, unsigned int last,
01004
                                     uint64_t *ext_rc);
01005
01006
01081 QATZIP_API int qzCompressCrc(QzSession_T *sess,
01082
                                     const unsigned char *src,
01083
                                     unsigned int *src_len,
01084
                                     unsigned char *dest,
01085
                                     unsigned int *dest_len,
01086
                                     unsigned int last,
                                     unsigned long *crc);
01088
01089 QATZIP_API int qzCompressCrcExt(QzSession_T *sess,
01090
                                        const unsigned char *src,
01091
                                        unsigned int *src len,
01092
                                        unsigned char *dest.
01093
                                        unsigned int *dest_len,
                                        unsigned int last,
01094
01095
                                        unsigned long *crc,
01096
                                        uint64_t *ext_rc);
01097
01098 OATZIP API int gzCompressCrc64(OzSession T *sess.
01099
                                       const unsigned char *src.
01100
                                       unsigned int *src_len,
01101
                                       unsigned char *dest,
01102
                                       unsigned int *dest_len,
01103
                                       unsigned int last,
01104
                                       uint64_t *crc);
01105
01106 QATZIP_API int qzCompressCrc64Ext(QzSession_T *sess,
01107
                                          const unsigned char *src,
01108
                                          unsigned int *src_len,
01109
                                          unsigned char *dest,
01110
                                          unsigned int *dest len,
                                          unsigned int last,
01111
01112
                                          uint64_t *crc,
                                          uint64_t *ext_rc);
01113
01114
01218 QATZIP_API int qzCompressWithMetadataExt(QzSession_T *sess,
          const unsigned char *src,
01219
01220
              unsigned int *src_len,
unsigned char *dest,
01221
              unsigned int *dest_len,
01223
              unsigned int last,
01224
              uint64_t *ext_rc,
              QzMetadataBlob_T *metadata,
uint32_t hw_buff_sz_override,
01225
01226
01227
              uint32 t comp thrshold);
01228
01287 QATZIP_API int qzDecompress(QzSession_T *sess, const unsigned char *src,
01288
                                    unsigned int *src_len, unsigned char *dest,
                                    unsigned int *dest_len);
01289
01290
01291 QATZIP_API int qzDecompressExt(QzSession_T *sess, const unsigned char *src,
                                       unsigned int *src_len, unsigned char *dest,
01292
                                       unsigned int *dest_len, uint64_t *ext_rc);
01293
01294
01355 QATZIP_API int qzDecompressCrc(QzSession_T *sess,
01356
                                       const unsigned char *src,
01357
                                       unsigned int *src len.
```

```
01358
                                      unsigned char *dest,
                                      unsigned int *dest_len,
01359
01360
                                      unsigned long *crc);
01361
01362 QATZIP_API int qzDecompressCrcExt(QzSession_T *sess,
01363
                                         const unsigned char *src.
01364
                                         unsigned int *src_len,
01365
                                         unsigned char *dest,
01366
                                         unsigned int *dest_len,
01367
                                         unsigned long *crc,
01368
                                        uint64_t *ext_rc);
01369
01370 QATZIP_API int qzDecompressCrc64(QzSession_T *sess,
01371
                                        const unsigned char *src,
01372
                                        unsigned int *src_len,
01373
                                        unsigned char *dest,
01374
                                        unsigned int *dest len.
01375
                                        uint64 t *crc);
01376
01377 QATZIP_API int qzDecompressCrc64Ext(QzSession_T *sess,
01378
                                           const unsigned char *src,
01379
                                           unsigned int *src_len,
01380
                                           unsigned char *dest,
01381
                                           unsigned int *dest len.
01382
                                           uint64_t *crc,
01383
                                           uint64_t *ext_rc);
01384
01465 QATZIP_API int qzDecompressWithMetadataExt(QzSession_T *sess,
01466
             const unsigned char *src,
01467
              unsigned int *src_len,
01468
              unsigned char *dest.
01469
              unsigned int *dest_len,
01470
              uint64_t *ext_rc,
01471
              QzMetadataBlob_T *metadata,
01472
              uint32_t hw_buff_sz_override);
01473
01515 OATZIP API int gzTeardownSession(OzSession T *sess);
01516
01556 QATZIP_API int qzClose(QzSession_T *sess);
01557
01656 QATZIP_API int qzGetStatus(QzSession_T *sess, QzStatus_T *status);
01657
01699 #define QZ_SKID_PAD_SZ 48
01700 #define QZ_COMPRESSED_SZ_OF_EMPTY_FILE 34
01701 QATZIP_API
01702 unsigned int qzMaxCompressedLength(unsigned int src_sz, QzSession_T *sess);
01703
01742 QATZIP_API int qzSetDefaults(QzSessionParams_T *defaults);
01743
01744 QATZIP_API int qzSetDefaultsDeflate(QzSessionParamsDeflate_T *defaults);
01746 QATZIP_API int qzSetDefaultsLZ4(QzSessionParamsLZ4_T *defaults);
01747
01748 QATZIP_API int qzSetDefaultsLZ4S(QzSessionParamsLZ4S_T *defaults);
01749
01787 OATZIP API int gzGetDefaults(OzSessionParams T *defaults);
01788
01789 QATZIP_API int qzGetDefaultsDeflate(QzSessionParamsDeflate_T *defaults);
01790
01791 QATZIP_API int qzGetDefaultsLZ4(QzSessionParamsLZ4_T *defaults);
01792
01793 QATZIP_API int qzGetDefaultsLZ4S(QzSessionParamsLZ4S_T *defaults);
01794
01835 QATZIP_API void *qzMalloc(size_t sz, int numa, int force_pinned);
01836
01882 QATZIP_API int qzAllocateMetadata(QzMetadataBlob_T *metadata,
01883
                                         size_t data_size,
                                         uint32_t hw_buff_sz);
01884
01885
01920 QATZIP_API void qzFree(void *m);
01921
01960 QATZIP_API int qzFreeMetadata(QzMetadataBlob_T metadata);
01961
02000 QATZIP_API int qzMemFindAddr(unsigned char *a);
02001
02011 typedef struct QzStream_S {
02012
          unsigned int in_sz;
02014
          unsigned int out_sz;
02016
          unsigned char *in ;
          unsigned char *out :
02018
02020
          unsigned int pending_in;
02022
          unsigned int pending_out;
02024
          QzCrcType_T crc_type;
02026
          unsigned int crc_32;
02028
          unsigned long long reserved;
02030
          void *opaque;
02032 } QzStream_T;
```

```
02033
02109 QATZIP_API
02110 int qzCompressStream(QzSession_T *sess, QzStream_T *strm, unsigned int last);
02111
02186 OATZIP API
02187 int gzDecompressStream(QzSession_T *sess, QzStream_T *strm, unsigned int last);
02229 QATZIP_API int qzEndStream(QzSession_T *sess, QzStream_T *strm);
02230
02281 OATZIP API
02282 int qzGetSoftwareComponentVersionList(QzSoftwareVersionInfo_T *api_info,
02283
                                            unsigned int *num_elem);
02284
02331 QATZIP_API int qzGetSoftwareComponentCount(unsigned int *num_elem);
02332
02375 QATZIP_API int qzGetSessionCrc64Config(QzSession_T *sess,
02376
                                              QzCrc64Config_T *crc64_config);
02377
02422 QATZIP_API int qzSetSessionCrc64Config(QzSession_T *sess,
                                             QzCrc64Config_T *crc64_config);
02424
02488 QATZIP_API int qzMetadataBlockRead(uint32_t block_num,
02489
                                          OzMetadataBlob T metadata,
02490
                                         uint32_t *block_offset,
02491
                                         uint32_t *block_size,
02492
                                         uint32_t *block_flags,
                                         uint32_t *block_hash);
02493
02494
02557 QATZIP_API int qzMetadataBlockWrite(uint32_t block_num,
02558
                                          QzMetadataBlob_T metadata,
02559
                                          uint32_t *block_offset,
02560
                                          uint32_t *block_size,
02561
                                          uint32_t *block_flags,
02562
                                          uint32_t *block_hash);
02563
02564 #ifdef __cplusplus
02565 }
02566 #endif
02568 #endif
```

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

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

6.4 qz utils.h

Go to the documentation of this file.

```
00001 /***
00002
00003
           BSD LICENSE
00004
00005
           Copyright(c) 2007-2023 Intel Corporation. All rights reserved.
00006
           All rights reserved.
00007
80000
           Redistribution and use in source and binary forms, with or without
00009
           modification, are permitted provided that the following conditions
00010
           are met:
00011
00012 *
             * Redistributions of source code must retain the above copyright
00013 *
              notice, this list of conditions and the following disclaimer.
00014 *
            \star Redistributions in binary form must reproduce the above copyright
00015 *
              notice, this list of conditions and the following disclaimer in
00016 *
              the documentation and/or other materials provided with the
00017 *
               distribution.
00018
            \star Neither the name of Intel Corporation nor the names of its
00019 *
               contributors may be used to endorse or promote products derived
00020 *
               from this software without specific prior written permission.
00021 *
           THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
00022 *
           "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
00024
           LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
00025
           A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
           OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
00026 *
00027
00028
           LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
00029
           DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
00030
           THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
00031
           (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
00032
           OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
00033
00034
                ************************
00035
00050 #ifndef _QZ_UTILS_H_
00051 #define _QZ_UTILS_H_
00052
00053 #include <stdarg.h>
00054 #include <pthread.h>
00055 #include <stdio.h>
00056 #include <time.h>
00057
00058 typedef enum SERV_E {
          COMPRESSION = 0,
00059
00060
         DECOMPRESSION
00061 } Serv_T;
00062
00063 typedef enum ENGINE_E {
00064
         HW = 0,
00065
         SW
00066 } Engine_T;
00067
00068 typedef struct ThreadList_S {
         unsigned int thread_id;
00069
00070
          unsigned int comp_hw_count;
00071
          unsigned int comp_sw_count;
00072
         unsigned int decomp_hw_count;
00073
         unsigned int decomp_sw_count;
          struct ThreadList_S *next;
00075 } ThreadList_T;
```

6.4 qz_utils.h 77

```
00077 typedef struct QatThread_S {
00078
           ThreadList_T *comp_th_list;
           unsigned int num_comp_th;
00079
          pthread_mutex_t comp_lock;
   ThreadList_T *decomp_th_list;
   unsigned int num_decomp_th;
00080
00081
00083
           pthread_mutex_t decomp_lock;
00084 } QatThread_T;
00085
00086 extern void initDebugLock(void);
00087 extern void dumpThreadInfo(void);
00088 extern void insertThread(unsigned int th_id,
00089
                                   Serv_T serv_type,
00090
                                   Engine_T engine_type);
00091
00092 #ifdef QATZIP_DEBUG
00093 static inline void QZ_DEBUG(const char *format, ...)
00094 {
00095
           va_list args;
00096
           va_start(args, format);
00097
           vfprintf(stdout, format, args);
00098
          va_end(args);
00099 }
00100 #else
00101 #define QZ_DEBUG(...)
00102 #endif
00103
00104 #ifdef ENABLE_TESTLOG
00105 #define QZ_TESTLOG(debuglevel, Readable, tag, ...) { \ 00106 FILE *fd = debuglevel > 1 ? stdout : stderr; \
00107
           fprintf(fd, "Tag: %s; ", tag); \
           if (Readable) { \
    fprintf(fd, "Time: %s %s; Location: %s->%s->%d; ", \
00108
00109
                       __DATE__, __TIME__, __FILE__, __func__, __LINE__); \
00110
           } else { \
00111
              struct timespec ts = { 0 }; \
clock_gettime(CLOCK_MONOTONIC_RAW, &ts); \
00112
00114
               fprintf(fd, "Time: %ld.%06lds; ", ts.tv_sec, ts.tv_nsec / 1000); \
00115
           fprintf(fd, "%s", "Info: "); \
00116
           fprintf(fd, __VA_ARGS__); \
fprintf(fd, " \n"); \
00117
00118
00119 }
00120 #endif
00121
00122 static inline void QZ_PRINT(const char *format, ...)
00123 {
00124
           va list args:
00125
           va_start(args, format);
00126
           vfprintf(stdout, format, args);
00127
           va_end(args);
00128 }
00129
00130 static inline void QZ_ERROR(const char *format, ...)
00131 {
           va_list args;
00133
           va_start(args, format);
00134
           vfprintf(stderr, format, args);
00135
           va_end(args);
00136 }
00137
00138 #endif
```

Index

```
algo hw
                                                      qzCompress, 19
    QzStatus_S, 56
                                                      qzCompressCrc, 20
                                                      qzCompressStream, 22
algo sw
                                                      qzCompressWithMetadataExt, 23
    QzStatus S, 56
                                                      QzCrc64Config_T, 11
applications.qat.shims.qatzip.qatzip/include/qatzip.h,
                                                      QzCrcType_E, 15
        61,69
applications.gat.shims.gatzip.gatzip/include/gz utils.h,
                                                      QzCrcType T, 11
                                                      QzDataFormat_E, 15
        75, 76
                                                      QzDataFormat_T, 11
COMMON MEM
                                                      qzDecompress, 25
    Data Compression API, 15
                                                      gzDecompressCrc, 26
comp algorithm
                                                      gzDecompressStream, 27
    QzSessionParams S, 50
                                                      qzDecompressWithMetadataExt, 28
    QzSessionParamsCommon S, 52
                                                      QzDirection E, 16
comp IvI
                                                      QzDirection T, 11
    QzSessionParams S, 50
                                                      qzEndStream, 29
    QzSessionParamsCommon S, 52
                                                      qzFree, 30
crc 32
                                                      qzFreeMetadata, 31
    QzStream_S, 58
                                                      qzGetDefaults, 31
crc_type
                                                      qzGetSessionCrc64Config, 32
    QzStream_S, 58
                                                      gzGetSoftwareComponentCount, 33
                                                      qzGetSoftwareComponentVersionList, 34
Data Compression API, 7
                                                      gzGetStatus, 35
    COMMON MEM, 15
                                                      QzHuffmanHdr E, 16
    NONE, 15
                                                      QzHuffmanHdr T, 11
    PinMem E, 15
                                                      qzInit, 36
    PinMem_T, 11
                                                      qzLZ4SCallbackFn, 12
    PINNED MEM, 15
                                                      qzMalloc, 38
    QATZIP API VERSION NUM MAJOR, 9
                                                      qzMemFindAddr, 38
    QATZIP_API_VERSION_NUM_MINOR, 9
                                                      QzMetadataBlob T, 14
    QZ_ADLER, 15
                                                      qzMetadataBlockRead, 39
    QZ_BUSY_POLLING, 17
                                                      qzMetadataBlockWrite, 40
    QZ_CRC32, 15
                                                      QzPollingMode_E, 17
    QZ_DEFLATE_4B, 16
                                                      QzPollingMode T, 14
    QZ DEFLATE GZIP, 16
                                                      QzSession T, 14
    QZ DEFLATE GZIP EXT, 16
                                                      QzSessionParams_T, 14
    QZ_DEFLATE_RAW, 16
                                                      qzSetDefaults, 42
    QZ DIR BOTH, 16
                                                      qzSetSessionCrc64Config, 42
    QZ DIR COMPRESS, 16
                                                      qzSetupSession, 43
    QZ DIR DECOMPRESS, 16
                                                      QzSoftwareComponentType_E, 18
    QZ_DYNAMIC_HDR, 17
                                                      QzSoftwareComponentType_T, 14
    QZ_MAX_STRING_LENGTH, 9
                                                      QzStatus T, 14
    QZ OK, 9
                                                      QzStream T, 15
    QZ PERIODICAL_POLLING, 17
                                                      qzTeardownSession, 45
    QZ_SKID_PAD_SZ, 9
                                                  data fmt
    QZ STATIC HDR, 17
                                                      QzSessionParams S, 50
    QZ SW BACKUP BIT POSITION, 10
                                                      QzSessionParamsDeflate_S, 54
    QZ SW EXECUTION BIT, 10
                                                  debug API, 46
    qzAllocateMetadata, 18
                                                  direction
    qzClose, 18
```

QzSessionParams_S, 50	QzCrc64Config_S, 48
QzSessionParamsCommon_S, 52	<u> </u>
42500010111	qat_hw_count
huffman_hdr	QzStatus_S, 57
QzSessionParams_S, 50	qat_instance_attach
QzSessionParamsDeflate_S, 54	QzStatus_S, 57
hw_buff_sz	qat_mem_drvr
QzSessionParams_S, 51	QzStatus_S, 57
	qat service init
QzSessionParamsCommon_S, 52	QzStatus S, 57
hw_session_stat	- :
QzSession_S, 49	QatThread_S, 47
hw_session_status	qatzip.h
QzStatus_S, 57	QATZIP_API, 65
	QATZIP_API_VERSION, 65
in	QZ_BUF_ERROR, 65
QzStream_S, 58	QZ_DATA_ERROR, 65
in_sz	QZ_DEFLATE, 65
QzStream_S, 59	QZ_DISABLE_SOFTWARE_BACKUP, 65
initial value	QZ_DISABLE_SOFTWARE_ONLY_EXECUTION,
- QzCrc64Config_S, 48	66
input_sz_thrshold	QZ_DUPLICATE, 66
QzSessionParams_S, 51	QZ_ENABLE_SOFTWARE_BACKUP, 66
QzSessionParamsCommon S, 53	
	QZ_ENABLE_SOFTWARE_ONLY_EXECUTION,
internal Consider S. 40	66
QzSession_S, 49	QZ_FAIL, 66
is_sensitive_mode	QZ_FORCE_SW, 66
QzSessionParamsCommon_S, 53	QZ_INTEG, 66
	QZ_LOW_DEST_MEM, 67
lz4s_mini_match	QZ_LOW_MEM, 67
QzSessionParamsLZ4S_S, 55	QZ_METADATA_OVERFLOW, 67
	QZ_NO_HW, 67
max_forks	QZ_NO_INST_ATTACH, 67
QzSessionParams_S, 51	QZ_NO_MDRV, 67
QzSessionParamsCommon_S, 53	QZ_NO_SW_AVAIL, 67
memory_alloced	QZ_NONE, 67
QzStatus_S, 57	QZ NOSW LOW MEM, 68
NONE	QZ_NOSW_NO_HW, 68
Data Compression API, 15	QZ_NOSW_NO_INST_ATTACH, 68
	QZ_NOSW_NO_MDRV, 68
opaque	QZ_NOSW_UNSUPPORTED_FMT, 68
QzStream_S, 59	QZ_NOT_SUPPORTED, 68
out	QZ_OUT_OF_RANGE, 68
QzStream_S, 59	QZ_PARAMS, 68
out sz	QZ_POST_PROCESS_ERROR, 69
QzStream_S, 59	QZ_TIMEOUT, 69
Q201104111_0, 00	QZ_UNSUPPORTED_FMT, 69
pending_in	QATZIP_API
QzStream S, 59	qatzip.h, 65
pending_out	QATZIP_API_VERSION
•	
QzStream_S, 59	qatzip.h, 65
PinMem_E	QATZIP_API_VERSION_NUM_MAJOR
Data Compression API, 15	Data Compression API, 9
PinMem_T	QATZIP_API_VERSION_NUM_MINOR
Data Compression API, 11	Data Compression API, 9
PINNED_MEM	QZ_ADLER
Data Compression API, 15	Data Compression API, 15
polling_mode	QZ_BUF_ERROR
QzSessionParamsCommon_S, 53	qatzip.h, 65
polynomial	QZ_BUSY_POLLING
1 2	

	Data Compression API, 17	qatzip.h, 68
$QZ_{}$	_CRC32	QZ_NOSW_NO_HW
	Data Compression API, 15	qatzip.h, 68
$QZ_{}$	DATA_ERROR	QZ_NOSW_NO_INST_ATTACH
	qatzip.h, 65	qatzip.h, 68
QΖ	DEFLATE	QZ_NOSW_NO_MDRV
_	qatzip.h, 65	qatzip.h, 68
QΖ	DEFLATE 4B	QZ_NOSW_UNSUPPORTED_FMT
~	Data Compression API, 16	qatzip.h, 68
07	DEFLATE GZIP	QZ_NOT_SUPPORTED
\	Data Compression API, 16	qatzip.h, 68
07	DEFLATE_GZIP_EXT	QZ OK
QΖ_	Data Compression API, 16	Data Compression API, 9
07	DEFLATE RAW	QZ_OUT_OF_RANGE
QZ_	-	
07	Data Compression API, 16	qatzip.h, 68
QZ_	_DIR_BOTH	QZ_PARAMS
07	Data Compression API, 16	qatzip.h, 68
QZ_	_DIR_COMPRESS	QZ_PERIODICAL_POLLING
	Data Compression API, 16	Data Compression API, 17
QZ_	_DIR_DECOMPRESS	QZ_POST_PROCESS_ERROR
	Data Compression API, 16	qatzip.h, 69
QZ_	_DISABLE_SOFTWARE_BACKUP	QZ_SKID_PAD_SZ
	qatzip.h, 65	Data Compression API, 9
$QZ_{}$	_DISABLE_SOFTWARE_ONLY_EXECUTION	QZ_STATIC_HDR
	qatzip.h, 66	Data Compression API, 17
$QZ_{}$	DUPLICATE	QZ_SW_BACKUP_BIT_POSITION
	qatzip.h, 66	Data Compression API, 10
$QZ_{}$	_DYNAMIC_HDR	QZ_SW_EXECUTION_BIT
	Data Compression API, 17	Data Compression API, 10
$QZ_{}$	_ENABLE_SOFTWARE_BACKUP	QZ_TIMEOUT
	qatzip.h, 66	qatzip.h, 69
$QZ_{}$	_ENABLE_SOFTWARE_ONLY_EXECUTION	QZ_UNSUPPORTED_FMT
	qatzip.h, 66	qatzip.h, 69
$QZ_{}$	_FAIL	qzAllocateMetadata
	qatzip.h, 66	Data Compression API, 18
$QZ_{\underline{}}$	FORCE_SW	qzCallback
	qatzip.h, 66	QzSessionParamsLZ4S_S, 55
$QZ_{}$	INTEG	qzCallback_external
	qatzip.h, 66	QzSessionParamsLZ4S_S, 55
QΖ	LOW DEST MEM	qzClose
	qatzip.h, 67	Data Compression API, 18
QΖ	LOW_MEM	qzCompress
	gatzip.h, 67	Data Compression API, 19
QΖ	MAX_STRING_LENGTH	qzCompressCrc
	Data Compression API, 9	Data Compression API, 20
07	METADATA_OVERFLOW	qzCompressStream
	gatzip.h, 67	Data Compression API, 22
07	NO_HW	qzCompressWithMetadataExt
~- _	gatzip.h, 67	Data Compression API, 23
07	NO_INST_ATTACH	QzCrc64Config_S, 47
α	qatzip.h, 67	initial_value, 48
07	NO_MDRV	polynomial, 48
UZ_		• •
07	qatzip.h, 67	reflect_in, 48
UZ_	NO_SW_AVAIL	reflect_out, 48
~~	qatzip.h, 67	xor_out, 48
QZ_	NONE	QzCrc64Config_T
~~	qatzip.h, 67	Data Compression API, 11
QZ_	NOSW_LOW_MEM	QzCrcType_E

Data Compression API, 15	hw_session_stat, 49
QzCrcType_T	internal, 49
Data Compression API, 11	
	thd_sess_stat, 49
QzDataFormat_E	total_in, 49
Data Compression API, 15	total_out, 49
QzDataFormat_T	QzSession_T
Data Compression API, 11	Data Compression API, 14
qzDecompress	QzSessionParams_S, 49
Data Compression API, 25	comp_algorithm, 50
qzDecompressCrc	comp_lvl, 50
Data Compression API, 26	data_fmt, 50
qzDecompressStream	direction, 50
Data Compression API, 27	huffman_hdr, 50
qzDecompressWithMetadataExt	hw_buff_sz, 51
Data Compression API, 28	input_sz_thrshold, 51
QzDirection E	max_forks, 51
Data Compression API, 16	req_cnt_thrshold, 51
QzDirection T	strm_buff_sz, 51
Data Compression API, 11	sw_backup, 51
qzEndStream	wait cnt thrshold, 51
Data Compression API, 29	QzSessionParams_T
qzFree	Data Compression API, 14
•	QzSessionParamsCommon_S, 52
Data Compression API, 30	
qzFreeMetadata	comp_algorithm, 52
Data Compression API, 31	comp_lvl, 52
qzGetDefaults	direction, 52
Data Compression API, 31	hw_buff_sz, 52
qzGetSessionCrc64Config	input_sz_thrshold, 53
Data Compression API, 32	is_sensitive_mode, 53
qzGetSoftwareComponentCount	max_forks, 53
Data Compression API, 33	polling_mode, 53
qzGetSoftwareComponentVersionList	req_cnt_thrshold, 53
Data Compression API, 34	strm_buff_sz, 53
qzGetStatus	sw_backup, 53
Data Compression API, 35	wait_cnt_thrshold, 54
QzHuffmanHdr_E	QzSessionParamsDeflate_S, 54
Data Compression API, 16	data_fmt, 54
QzHuffmanHdr_T	huffman_hdr, 54
Data Compression API, 11	QzSessionParamsLZ4 S, 55
qzInit	QzSessionParamsLZ4S_S, 55
Data Compression API, 36	Iz4s_mini_match, 55
qzLZ4SCallbackFn	gzCallback, 55
Data Compression API, 12	qzCallback_external, 55
qzMalloc	qzSetDefaults
Data Compression API, 38	
qzMemFindAddr	•
•	Data Compression API, 42
	Data Compression API, 42 qzSetSessionCrc64Config
Data Compression API, 38	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42
QzMetadataBlob_T	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession
QzMetadataBlob_T Data Compression API, 14	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession Data Compression API, 43
QzMetadataBlob_T Data Compression API, 14 qzMetadataBlockRead	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession Data Compression API, 43 QzSoftwareComponentType_E
QzMetadataBlob_T Data Compression API, 14 qzMetadataBlockRead Data Compression API, 39	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession Data Compression API, 43 QzSoftwareComponentType_E Data Compression API, 18
QzMetadataBlob_T Data Compression API, 14 qzMetadataBlockRead Data Compression API, 39 qzMetadataBlockWrite	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession Data Compression API, 43 QzSoftwareComponentType_E Data Compression API, 18 QzSoftwareComponentType_T
QzMetadataBlob_T Data Compression API, 14 qzMetadataBlockRead Data Compression API, 39 qzMetadataBlockWrite Data Compression API, 40	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession Data Compression API, 43 QzSoftwareComponentType_E Data Compression API, 18 QzSoftwareComponentType_T Data Compression API, 14
QzMetadataBlob_T Data Compression API, 14 qzMetadataBlockRead Data Compression API, 39 qzMetadataBlockWrite Data Compression API, 40 QzPollingMode_E	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession Data Compression API, 43 QzSoftwareComponentType_E Data Compression API, 18 QzSoftwareComponentType_T Data Compression API, 14 QzSoftwareVersionInfo_S, 56
QzMetadataBlob_T Data Compression API, 14 qzMetadataBlockRead Data Compression API, 39 qzMetadataBlockWrite Data Compression API, 40 QzPollingMode_E Data Compression API, 17	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession Data Compression API, 43 QzSoftwareComponentType_E Data Compression API, 18 QzSoftwareComponentType_T Data Compression API, 14 QzSoftwareVersionInfo_S, 56 QzStatus_S, 56
QzMetadataBlob_T Data Compression API, 14 qzMetadataBlockRead Data Compression API, 39 qzMetadataBlockWrite Data Compression API, 40 QzPollingMode_E Data Compression API, 17 QzPollingMode_T	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession Data Compression API, 43 QzSoftwareComponentType_E Data Compression API, 18 QzSoftwareComponentType_T Data Compression API, 14 QzSoftwareVersionInfo_S, 56 QzStatus_S, 56 algo_hw, 56
QzMetadataBlob_T Data Compression API, 14 qzMetadataBlockRead Data Compression API, 39 qzMetadataBlockWrite Data Compression API, 40 QzPollingMode_E Data Compression API, 17	Data Compression API, 42 qzSetSessionCrc64Config Data Compression API, 42 qzSetupSession Data Compression API, 43 QzSoftwareComponentType_E Data Compression API, 18 QzSoftwareComponentType_T Data Compression API, 14 QzSoftwareVersionInfo_S, 56 QzStatus_S, 56

```
memory_alloced, 57
    qat_hw_count, 57
    qat_instance_attach, 57
    qat_mem_drvr, 57
    qat_service_init, 57
    using_huge_pages, 57
QzStatus T
    Data Compression API, 14
QzStream S, 58
    crc_32, 58
    crc_type, 58
    in, 58
    in_sz, 59
    opaque, 59
    out, 59
    out_sz, 59
    pending_in, 59
    pending_out, 59
    reserved, 59
QzStream_T
    Data Compression API, 15
qzTeardownSession
    Data Compression API, 45
reflect_in
    QzCrc64Config_S, 48
reflect_out
    QzCrc64Config_S, 48
req_cnt_thrshold
    QzSessionParams_S, 51
    QzSessionParamsCommon S, 53
reserved
    QzStream_S, 59
strm buff sz
    QzSessionParams_S, 51
    QzSessionParamsCommon_S, 53
sw_backup
    QzSessionParams_S, 51
    QzSessionParamsCommon_S, 53
thd_sess_stat
    QzSession_S, 49
ThreadList S, 60
total in
    QzSession_S, 49
total_out
    QzSession_S, 49
using_huge_pages
    QzStatus_S, 57
wait_cnt_thrshold
    QzSessionParams S, 51
    QzSessionParamsCommon_S, 54
xor_out
    QzCrc64Config_S, 48
```