

QATzip

1.1.2

Generated by Doxygen 1.9.1



<b>1 Module Index</b>	<b>1</b>
1.1 Modules	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Module Documentation</b>	<b>7</b>
4.1 Data Compression API	7
4.1.1 Detailed Description	9
4.1.2 Macro Definition Documentation	9
4.1.2.1 QATZIP_API_VERSION_NUM_MAJOR	9
4.1.2.2 QATZIP_API_VERSION_NUM_MINOR	9
4.1.2.3 QZ_MAX_STRING_LENGTH	9
4.1.2.4 QZ_OK	9
4.1.2.5 QZ_SKID_PAD_SZ	9
4.1.2.6 QZ_SW_BACKUP_BIT_POSITION	10
4.1.2.7 QZ_SW_EXECUTION_BIT	10
4.1.3 Typedef Documentation	11
4.1.3.1 PinMem_T	11
4.1.3.2 QzCrc64Config_T	11
4.1.3.3 QzCrcType_T	11
4.1.3.4 QzDataFormat_T	11
4.1.3.5 QzDirection_T	11
4.1.3.6 QzHuffmanHdr_T	12
4.1.3.7 qzLZ4SCallbackFn	13
4.1.3.8 QzMetadataBlob_T	14
4.1.3.9 QzPollingMode_T	14
4.1.3.10 QzSession_T	14
4.1.3.11 QzSessionParams_T	14
4.1.3.12 QzSoftwareComponentType_T	14
4.1.3.13 QzStatus_T	15
4.1.3.14 QzStream_T	15
4.1.4 Enumeration Type Documentation	15
4.1.4.1 PinMem_E	15
4.1.4.2 QzCrcType_E	15
4.1.4.3 QzDataFormat_E	16
4.1.4.4 QzDirection_E	16
4.1.4.5 QzHuffmanHdr_E	16
4.1.4.6 QzPollingMode_E	17
4.1.4.7 QzSoftwareComponentType_E	18

4.1.5 Function Documentation	18
4.1.5.1 qzAllocateMetadata()	18
4.1.5.2 qzClose()	19
4.1.5.3 qzCompress()	20
4.1.5.4 qzCompressCrc()	21
4.1.5.5 qzCompressStream()	22
4.1.5.6 qzCompressWithMetadataExt()	24
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 qzInit()	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	45
<b>5 Class Documentation</b>	<b>47</b>
5.1 QatThread_S Struct Reference	47
5.2 QzCrc64Config_S Struct Reference	47
5.2.1 Detailed Description	47
5.2.2 Member Data Documentation	48
5.2.2.1 initial_value	48
5.2.2.2 polynomial	48
5.2.2.3 reflect_in	48
5.2.2.4 reflect_out	48
5.2.2.5 xor_out	48
5.3 QzSession_S Struct Reference	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 . . . . .	50
5.4 QzSessionParams_S Struct Reference . . . . .	50
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 . . . . .	51
5.4.2.3 data_fmt . . . . .	51
5.4.2.4 direction . . . . .	51
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 . . . . .	52
5.4.2.10 strm_buff_sz . . . . .	52
5.4.2.11 sw_backup . . . . .	52
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 . . . . .	53
5.5.1.2 comp_lvl . . . . .	53
5.5.1.3 direction . . . . .	53
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 . . . . .	54
5.5.1.9 req_cnt_thrshold . . . . .	54
5.5.1.10 strm_buff_sz . . . . .	54
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 . . . . .	55
5.6.1.2 huffman_hdr . . . . .	55
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 lz4s_mini_match . . . . .	55
5.8.1.2 qzCallback . . . . .	56
5.8.1.3 qzCallback_external . . . . .	56
5.9 QzSoftwareVersionInfo_S Struct Reference . . . . .	56
5.10 QzStatus_S Struct Reference . . . . .	56
5.10.1 Detailed Description . . . . .	57
5.10.2 Member Data Documentation . . . . .	57
5.10.2.1 algo_hw . . . . .	57
5.10.2.2 algo_sw . . . . .	57
5.10.2.3 hw_session_status . . . . .	57
5.10.2.4 memory_allocated . . . . .	57
5.10.2.5 qat_hw_count . . . . .	57
5.10.2.6 qat_instance_attach . . . . .	57
5.10.2.7 qat_mem_drvr . . . . .	58
5.10.2.8 qat_service_init . . . . .	58
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 . . . . .	59
5.11.2.1 crc_32 . . . . .	59
5.11.2.2 crc_type . . . . .	59
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 . . . . .	60
5.11.2.9 pending_out . . . . .	60
5.11.2.10 reserved . . . . .	60
5.12 ThreadList_S Struct Reference . . . . .	60
<b>6 File Documentation</b>	<b>61</b>
6.1 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

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 include/qz_utils.h File Reference . . . . .	69

## Index

71





# Chapter 1

## Module Index

### 1.1 Modules

Here is a list of all modules:

Data Compression API . . . . .	<a href="#">7</a>
debug API . . . . .	<a href="#">45</a>



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">QatThread_S</a>	47
<a href="#">QzCrc64Config_S</a>	47
<a href="#">QzSession_S</a>	48
<a href="#">QzSessionParams_S</a>	50
<a href="#">QzSessionParamsCommon_S</a>	52
<a href="#">QzSessionParamsDeflate_S</a>	54
<a href="#">QzSessionParamsLZ4_S</a>	55
<a href="#">QzSessionParamsLZ4S_S</a>	55
<a href="#">QzSoftwareVersionInfo_S</a>	56
<a href="#">QzStatus_S</a>	56
<a href="#">QzStream_S</a>	58
<a href="#">ThreadList_S</a>	60



## Chapter 3

# File Index

### 3.1 File List

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

include/ <a href="#">qatzip.h</a> . . . . .	61
include/ <a href="#">qz_utils.h</a> . . . . .	69



## 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 [qzTeardownSession](#) ([QzSession\\_T](#) \*sess)
- [QATZIP\\_API](#) int [qzClose](#) ([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 [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 ↵  
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 ↵  
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**

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

**Return values**

<code>dest_sz</code>	Max compressed data output length in bytes. When <code>src_sz</code> is equal to 0, the return value is <code>QZ_COMPRESSED_SZ_OF_EMPTY_FILE(34)</code> . 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, &params); qzCompress(&sess, src, &src_len, dest, &dest_len, 1); qzDecompress(&sess, src, &src_len, dest, &dest_len); qzTeardownSession(&sess); qzClose(&sess);
```

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

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

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

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

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

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

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

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

1. Addition of LZ4 and LZ4s
  2. Addition of post processing functions for out of LZ4s
  3. Compression level up to 12 for LZ4 and LZ4s
  4. Support for 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_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	<i>external</i>	User context provided through the 'qzCallback_external' pointer in the QzSessionParams_T structure.
in	<i>src</i>	Point to source buffer
in, out	<i>src_len</i>	Length of source buffer. Modified to number of bytes consumed
in	<i>dest</i>	Point to destination buffer
in, out	<i>dest_len</i>	Length of destination buffer. Modified to length of compressed data when function returns
in, out	<i>ExtStatus</i>	'qzCallback' customized error code.

##### Return values

<i>QZ_OK</i>	Function executed successfully
<i>QZ_FAIL</i>	Function did not succeed
<i>QZ_PARAMS</i>	params are invalid
<i>QZ_POST_PROCESS_ERROR</i>	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, &params); qzCompress(&sess, src, &src_len, dest, &dest_len, 1); qzDecompress(&sess, src, &src_len, dest, &dest_len); qzTeardownSession(&sess); qzClose(&sess);
```

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

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

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

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

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

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

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

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

1. Addition of LZ4 and LZ4s
  2. Addition of post processing functions for out of LZ4s
  3. Compression level up to 12 for LZ4 and LZ4s
  4. Support for 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

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()

```
QATZIP_API int qzAllocateMetadata (
    QzMetadataBlob_T * metadata,
    size_t data_size,
    uint32_t hw_buff_sz )
```

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	<i>metadata</i>	Pointer to opaque metadata.
in	<i>data_size</i>	Size of uncompressed buffer.
in	<i>hw_buff_sz</i>	Data size at which individual block will be compressed.

Return values

QZ_OK	Function executed successfully
QZ_FAIL	Function did not succeed

## Return values

<code>QZ_PARAMS</code>	*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()

```
QATZIP_API int qzClose (
    QzSession_T * sess )
```

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

<code>QZ_OK</code>	Function executed successfully
<code>QZ_FAIL</code>	Function did not succeed
<code>QZ_PARAMS</code>	*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()**

```
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 )
```

**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	<i>sess</i>	Session handle (pointer to opaque instance and session data)
in	<i>src</i>	Point to source buffer
in, out	<i>src_len</i>	Length of source buffer. Modified to number of bytes consumed
in	<i>dest</i>	Point to destination buffer
in, out	<i>dest_len</i>	Length of destination buffer. Modified to length of compressed data when function returns
in	<i>last</i>	1 for 'No more data to be compressed' 0 for 'More data to be compressed'
in, out	<i>ext_rc</i>	qzCompressExt only. If not NULL, <i>ext_rc</i> 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

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

## Precondition

None

## Postcondition

None

## Note

Only a synchronous version of this function is provided.

## See also

None

## 4.1.5.4 qzCompressCrc()

```
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 )
```

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	<i>sess</i>	Session handle (pointer to opaque instance and session data)
in	<i>src</i>	Point to source buffer
in, out	<i>src_len</i>	Length of source buffer. Modified to number of bytes consumed
in	<i>dest</i>	Point to destination buffer
in, out	<i>dest_len</i>	Length of destination buffer. Modified to length of compressed data when function returns
in	<i>last</i>	1 for 'No more data to be compressed' 0 for 'More data to be compressed'
in, out	<i>crc</i>	Pointer to CRC32 or CRC64 checksum buffer
in, out	<i>ext_rc</i>	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**

<i>QZ_OK</i>	Function executed successfully
<i>QZ_FAIL</i>	Function did not succeed
<i>QZ_PARAMS</i>	*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()**

```
QATZIP_API int qzCompressStream (
    QzSession_T * sess,
    QzStream_T * strm,
    unsigned int last )
```

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

<code>in</code>	<code>sess</code>	Session handle (pointer to opaque instance and session data)
<code>in, out</code>	<code>strm</code>	Stream handle
<code>in</code>	<code>last</code>	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

<code>QZ_OK</code>	Function executed successfully
<code>QZ_FAIL</code>	Function did not succeed
<code>QZ_PARAMS</code>	*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 or 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. <span style="float: right;">Generated by Doxygen</span>
in	comp_thrshold	Compressed block threshold.



## Return values

<i>QZ_OK</i>	Function executed successfully
<i>QZ_FAIL</i>	Function did not succeed
<i>QZ_PARAMS</i>	*sess or *metadata is NULL or Member of params is invalid, hw_buff_sz_override is invalid data size.
<i>QZ_METADATA_OVERFLOW</i>	Unable to populate metadata due to insufficient memory allocated.
<i>QZ_NOT_SUPPORTED</i>	Compression with metadata is not supported with given algorithm or format.

## Precondition

None

## Postcondition

None

## Note

Only a synchronous version of this function is provided.

## See also

None

## 4.1.5.7 qzDecompress()

```
QATZIP_API int qzDecompress (
    QzSession_T * sess,
    const unsigned char * src,
    unsigned int * src_len,
    unsigned char * dest,
    unsigned int * dest_len )
```

## 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	<i>sess</i>	Session handle (pointer to opaque instance and session data)
in	<i>src</i>	Point to source buffer
in	<i>src_len</i>	Length of source buffer. Modified to length of processed compressed data when function returns
Generated by Doxygen		
in	<i>dest</i>	Point to destination buffer
in, out	<i>dest_len</i>	Length of destination buffer. Modified to length of decompressed data when function returns

## Return values

<i>QZ_OK</i>	Function executed successfully
<i>QZ_FAIL</i>	Function did not succeed
<i>QZ_PARAMS</i>	*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()

```
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 )
```

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	<i>sess</i>	Session handle (pointer to opaque instance and session data)
in	<i>src</i>	Point to source buffer
in	<i>src_len</i>	Length of source buffer. Modified to length of processed compressed data when function returns
in	<i>dest</i>	Point to destination buffer
in, out	<i>dest_len</i>	Length of destination buffer. Modified to length of decompressed data when function returns
in, out	<i>crc</i>	Pointer to CRC32 or CRC64 checksum buffer

## Return values

<code>QZ_OK</code>	Function executed successfully
<code>QZ_FAIL</code>	Function did not succeed
<code>QZ_PARAMS</code>	*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()

```
QATZIP_API int qzDecompressStream (
    QzSession_T * sess,
    QzStream_T * strm,
    unsigned int last )
```

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**

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

**Return values**

<code>QZ_OK</code>	Function executed successfully
<code>QZ_FAIL</code>	Function did not succeed
<code>QZ_PARAMS</code>	*sess is NULL or member of params is invalid
<code>QZ_NEED_MORE</code>	*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 qzInit and qzSetupSession.

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` or 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	<i>sess</i>	Session handle (pointer to opaque instance and session data)
in	<i>src</i>	Point to source buffer
in	<i>src_len</i>	Length of source buffer. Modified to length of processed compressed data when function returns
in	<i>dest</i>	Point to destination buffer
in, out	<i>dest_len</i>	Length of destination buffer. Modified to length of decompressed data when function returns
in, out	<i>ext_rc</i>	If not NULL, <i>ext_rc</i> 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	<i>metadata</i>	Pointer to opaque metadata.
in	<i>hw_buff_sz_override</i>	Expected size of decompressed block.

## Return values

<i>QZ_OK</i>	Function executed successfully.
<i>QZ_FAIL</i>	Function did not succeed.
<i>QZ_PARAMS</i>	* <i>sess</i> or * <i>metadata</i> is NULL or Member of params is invalid, <i>hw_buff_sz_override</i> is invalid data size.
<i>QZ_METADATA_OVERFLOW</i>	Unable to populate metadata due to insufficient memory allocated.
<i>QZ_NOT_SUPPORTED</i>	Decompression with metadata is not supported with given algorithm or format.

## Precondition

None

## Postcondition

None

## Note

Only a synchronous version of this function is provided.

## See also

None

## 4.1.5.11 qzEndStream()

```
QATZIP_API int qzEndStream (
    QzSession_T * sess,
    QzStream_T * strm )
```

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**

<i>QZ_OK</i>	Function executed successfully
<i>QZ_FAIL</i>	Function did not succeed
<i>QZ_PARAMS</i>	*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()**

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

Free allocated memory

@description Free allocated memory.

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

**Parameters**

in	<i>m</i>	Memory address to be freed
----	----------	----------------------------

**Precondition**

None

**Postcondition**

None

**Note**

Only a synchronous version of this function is provided.

**See also**

None

**4.1.5.13 qzFreeMetadata()**

```
QATZIP_API int qzFreeMetadata (
    QzMetadataBlob_T * metadata )
```

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	<i>metadata</i>	Pointer to opaque metadata.
----	-----------------	-----------------------------

**Return values**

<i>QZ_OK</i>	Function executed successfully.
<i>QZ_FAIL</i>	Function did not succeed.
<i>QZ_PARAMS</i>	*metadata is NULL.

**Precondition**

None

**Postcondition**

None

**Note**

Only a synchronous version of this function is provided.

**See also**

None

#### 4.1.5.14 qzGetDefaults()

```
QATZIP_API int qzGetDefaults (
    QzSessionParams_T * defaults )
```

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()

```
QATZIP_API int qzGetSessionCrc64Config (
    QzSession_T * sess,
    QzCrc64Config_T * crc64_config )
```

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

## Return values

<i>QZ_OK</i>	Function executed successfully
<i>QZ_FAIL</i>	Session was not setup
<i>QZ_PARAMS</i>	*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()**

```
QATZIP_API int qzGetSoftwareComponentCount (
    unsigned int * num_elem )
```

Requests the number of Software components used by the QATZip library

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

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

## Parameters

in, out	<i>num_elem</i>	pointer to an unsigned int to populate how many software componets are associated with QATZip
---------	-----------------	---

**Return values**

<i>QZ_OK</i>	Function executed successfully
<i>QZ_FAIL</i>	Function did not succeed
<i>QZ_NO_SW_AVAIL</i>	Function did not find a software provider for fallback
<i>QZ_NO_HW</i>	Function did not find an installed kernel driver
<i>QZ_NOSW_NO_HW</i>	Functions did not find an installed kernel driver or software provider
<i>QZ_PARAMS</i>	*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()**

```
QATZIP_API int qzGetSoftwareComponentVersionList (
    QzSoftwareVersionInfo_T * api_info,
    unsigned int * num_elem )
```

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

## Return values

<code>QZ_OK</code>	Function executed successfully
<code>QZ_FAIL</code>	Function did not succeed
<code>QZ_NO_SW_AVAIL</code>	Function did not find a software provider for fallback
<code>QZ_NO_HW</code>	Function did not find an installed kernel driver
<code>QZ_NOSW_NO_HW</code>	Functions did not find an installed kernel driver or software provider
<code>QZ_PARAMS</code>	*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()

```
QATZIP_API int qzGetStatus (
    QzSession_T * sess,
    QzStatus_T * status )
```

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\_allocated 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	Y	0	QZ_FAIL
N	Y	1	QZ_NO_HW (1)
Y	N	0	QZ_OK
Y	N	1	QZ_NO_SW_AVAIL (2)
Y	Y	0	QZ_OK
Y	Y	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 be 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	<i>sess</i>	Session handle (pointer to opaque instance and session data)
in	<i>status</i>	Pointer to QATzip status structure

#### Return values

<i>QZ_OK</i>	Function executed successfully. The hardware based compression session has been created
<i>QZ_PARAMS</i>	*status is NULL

#### Precondition

None

#### Postcondition

None

#### Note

Only a synchronous version of this function is provided.

#### See also

None

## 4.1.5.19 qzInit()

```
QATZIP_API int qzInit (
    QzSession_T * sess,
    unsigned char sw_backup )
```

Initialize QAT hardware

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

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

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

## Parameters

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

## Return values

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

## Precondition

None

## Postcondition

None

**Note**

Only a synchronous version of this function is provided.

**See also**

None

**4.1.5.20 qzMalloc()**

```
QATZIP_API void* qzMalloc (
    size_t sz,
    int numa,
    int force_pinned )
```

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	<i>sz</i>	Memory size to be allocated
in	<i>numa</i>	NUMA node from which to allocate memory
in	<i>force_pinned</i>	PINNED_MEM allocate contiguous memory COMMON_MEM allocate non-contiguous memory

**Return values**

<i>NULL</i>	Fail to allocate memory
<i>address</i>	The address of allocated memory

**Precondition**

None

**Postcondition**

None

**Note**

Only a synchronous version of this function is provided.

**See also**

None

**4.1.5.21 qzMemFindAddr()**

```
QATZIP_API int qzMemFindAddr (
    unsigned char * a )
```

Check whether the address is available

@description Check whether the address is available.

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

**Parameters**

in	<i>a</i>	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()**

```
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 )
```

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

#### Return values

<i>QZ_OK</i>	Function executed successfully.
<i>QZ_FAIL</i>	Function did not succeed.
<i>QZ_PARAMS</i>	Metadata is NULL.
<i>QZ_OUT_OF_RANGE</i>	<code>block_num</code> specified is out of range.

#### Precondition

None

#### Postcondition

None

#### Note

Only a synchronous version of this function is provided.

#### See also

None



## 4.1.5.23 qzMetadataBlockWrite()

```
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 )
```

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

## Return values

<i>QZ_OK</i>	Function executed successfully.
<i>QZ_FAIL</i>	Function did not succeed.
<i>QZ_PARAMS</i>	Metadata is NULL.
<i>QZ_OUT_OF_RANGE</i>	<code>block_num</code> specified is out of range.

## Precondition

None

## Postcondition

None

**Note**

Only a synchronous version of this function is provided.

**See also**

None

**4.1.5.24 qzSetDefaults()**

```
QATZIP_API int qzSetDefaults (
    QzSessionParams_T * defaults )
```

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	<i>defaults</i>	The pointer to value to be set as default
----	-----------------	---

**Return values**

<i>QZ_OK</i>	Success on setting default value
<i>QZ_PARAM</i>	Fail to set default value

**Precondition**

None

**Postcondition**

None

**Note**

Only a synchronous version of this function is provided.

**See also**

None

**4.1.5.25 qzSetSessionCrc64Config()**

```
QATZIP_API int qzSetSessionCrc64Config (
    QzSession_T * sess,
    QzCrc64Config_T * crc64_config )
```

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	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 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()**

```
QATZIP_API int qzSetupSession (
    QzSession_T * sess,
    QzSessionParams_T * params )
```

Initialize a QATzip session

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

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

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

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

#### Parameters

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

#### Return values

<i>QZ_OK</i>	Function executed successfully. A hardware or software based compression session has been created
<i>QZ_DUPLICATE</i>	*sess includes an existing hardware or software session
<i>QZ_PARAMS</i>	*sess is NULL or member of params is invalid
<i>QZ_NOSW_NO_HW</i>	No hardware and no sw session being established
<i>QZ_NOSW_NO_MDRV</i>	No memory driver. No software session established
<i>QZ_NOSW_NO_INST_ATTACH</i>	No instance available No software session established
<i>QZ_NO_LOW_MEM</i>	Not enough pinned memory available No software session established
<i>QZ_UNSUPPORTED_FMT</i>	No support for requested algorithm; using software
<i>QZ_NOSW_UNSUPPORTED_FMT</i>	No support for requested algorithm; No software session established
<i>QZ_NO_SW_AVAIL</i>	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()

```
QATZIP_API int qzTeardownSession (
    QzSession_T * sess )
```

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:

- [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.

## 5.2.2 Member Data Documentation

### 5.2.2.1 initial\_value

`uint64_t QzCrc64Config_S::initial_value`

Defaults to 0x0000000000000000

### 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:

- [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:

- [include/qatzip.h](#)

## 5.4 QzSessionParams\_S Struct Reference

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

### Public Attributes

- [QzHuffmanHdr\\_T](#) `huffman_hdr`
- [QzDirection\\_T](#) `direction`
- [QzDataFormat\\_T](#) `data_fmt`
- unsigned int `comp_lvl`
- unsigned char `comp_algorithm`
- unsigned int `max_forks`
- unsigned char `sw_backup`
- unsigned int `hw_buff_sz`
- unsigned int `strm_buff_sz`
- unsigned int `input_sz_thrshold`
- unsigned int `req_cnt_thrshold`
- unsigned int `wait_cnt_thrshold`

### 5.4.1 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

```
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 qatzip\_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)

#### 5.4.2.12 wait\_cnt\_thrshold

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

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

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

- [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

```
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

```
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 qatzip\_internal.h

### 5.5.1.10 strm\_buff\_sz

`unsigned int QzSessionParamsCommon_S::strm_buff_sz`

Stream buffer size between [1K .. 2M - 5K] Default strm\_buf\_sz equals to hw\_buff\_sz

### 5.5.1.11 sw\_backup

`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:

- [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:

- [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:

- [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 lz4s 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:

- [include/qatzip.h](#)

## 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:

- [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 [qat\\_mem\\_drvr](#)
- unsigned char [qat\\_instance\\_attach](#)
- unsigned long int [memory\\_allocated](#)
- 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\_allocated

```
unsigned long int QzStatus_S::memory_allocated
```

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

### 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:

- [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

#### 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:

- [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:

- [include/qz\\_utils.h](#)

## Chapter 6

# File Documentation

### 6.1 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)

- `#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_POSITION))`
- `#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_POSITION))`

- `#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 }`

## 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\_thrsld)
- [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 )    (_BackupVariable &= ~(1 << QZ_SW_BACKUP_BIT_POSITION))
```

SW backup/fallback disabled

#### 6.1.1.7 QZ\_DISABLE\_SOFTWARE\_ONLY\_EXECUTION

```
#define QZ_DISABLE_SOFTWARE_ONLY_EXECUTION(  
    _BackupVariable )    (_BackupVariable &= ~(1 << QZ_SW_FORCESW_BIT_POSITION))
```

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

```
#define QZ_ENABLE_SOFTWARE_BACKUP(  
    _BackupVariable )    (_BackupVariable |= (1 << QZ_SW_BACKUP_BIT_POSITION))
```

SW backup/fallback enabled

#### 6.1.1.10 QZ\_ENABLE\_SOFTWARE\_ONLY\_EXECUTION

```
#define QZ_ENABLE_SOFTWARE_ONLY_EXECUTION(  
    _BackupVariable )    (_BackupVariable |= (1 << QZ_SW_FORCESW_BIT_POSITION))
```

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)
```

Insufficient 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

### 6.1.1.29 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 include/qz\_utils.h File Reference

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

### Classes

- struct [ThreadList\\_S](#)
- struct [QatThread\\_S](#)

### Macros

- `#define QZ_DEBUG(...)`

### Typedefs

- typedef enum `SERV_E` `Serv_T`
- typedef enum `ENGINE_E` `Engine_T`
- typedef struct [ThreadList\\_S](#) `ThreadList_T`
- typedef struct [QatThread\\_S](#) `QatThread_T`

### Enumerations

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

### Functions

- void `initDebugLock` (void)
- void `dumpThreadInfo` (void)
- void `insertThread` (unsigned int th\_id, `Serv_T` serv\_type, `Engine_T` engine\_type)



# Index

- algo\_hw
  - QzStatus\_S, [57](#)
- algo\_sw
  - QzStatus\_S, [57](#)
- COMMON\_MEM
  - Data Compression API, [15](#)
- comp\_algorithm
  - QzSessionParams\_S, [50](#)
  - QzSessionParamsCommon\_S, [52](#)
- comp\_lvl
  - QzSessionParams\_S, [50](#)
  - QzSessionParamsCommon\_S, [53](#)
- crc\_32
  - QzStream\_S, [59](#)
- crc\_type
  - QzStream\_S, [59](#)
- Data Compression API, [7](#)
  - COMMON\_MEM, [15](#)
  - NONE, [15](#)
  - PinMem\_E, [15](#)
  - PinMem\_T, [11](#)
  - PINNED\_MEM, [15](#)
  - QATZIP\_API\_VERSION\_NUM\_MAJOR, [9](#)
  - QATZIP\_API\_VERSION\_NUM\_MINOR, [9](#)
  - QZ\_ADLER, [15](#)
  - QZ\_BUSY\_POLLING, [18](#)
  - QZ\_CRC32, [15](#)
  - QZ\_DEFLATE\_4B, [16](#)
  - QZ\_DEFLATE\_GZIP, [16](#)
  - QZ\_DEFLATE\_GZIP\_EXT, [16](#)
  - QZ\_DEFLATE\_RAW, [16](#)
  - QZ\_DIR\_BOTH, [16](#)
  - QZ\_DIR\_COMPRESS, [16](#)
  - QZ\_DIR\_DECOMPRESS, [16](#)
  - QZ\_DYNAMIC\_HDR, [17](#)
  - QZ\_MAX\_STRING\_LENGTH, [9](#)
  - QZ\_OK, [9](#)
  - QZ\_PERIODICAL\_POLLING, [18](#)
  - QZ\_SKID\_PAD\_SZ, [9](#)
  - QZ\_STATIC\_HDR, [17](#)
  - QZ\_SW\_BACKUP\_BIT\_POSITION, [10](#)
  - QZ\_SW\_EXECUTION\_BIT, [10](#)
  - qzAllocateMetadata, [18](#)
  - qzClose, [19](#)
  - qzCompress, [20](#)
  - qzCompressCrc, [21](#)
  - qzCompressStream, [22](#)
  - qzCompressWithMetadataExt, [23](#)
  - QzCrc64Config\_T, [11](#)
  - QzCrcType\_E, [15](#)
  - QzCrcType\_T, [11](#)
  - QzDataFormat\_E, [16](#)
  - QzDataFormat\_T, [11](#)
  - qzDecompress, [25](#)
  - qzDecompressCrc, [26](#)
  - qzDecompressStream, [27](#)
  - qzDecompressWithMetadataExt, [28](#)
  - QzDirection\_E, [16](#)
  - QzDirection\_T, [11](#)
  - qzEndStream, [29](#)
  - qzFree, [30](#)
  - qzFreeMetadata, [31](#)
  - qzGetDefaults, [31](#)
  - qzGetSessionCrc64Config, [32](#)
  - qzGetSoftwareComponentCount, [33](#)
  - qzGetSoftwareComponentVersionList, [34](#)
  - qzGetStatus, [35](#)
  - QzHuffmanHdr\_E, [16](#)
  - QzHuffmanHdr\_T, [11](#)
  - qzInit, [36](#)
  - qzLZ4SCallbackFn, [12](#)
  - qzMalloc, [38](#)
  - qzMemFindAddr, [38](#)
  - QzMetadataBlob\_T, [14](#)
  - qzMetadataBlockRead, [39](#)
  - qzMetadataBlockWrite, [40](#)
  - QzPollingMode\_E, [17](#)
  - QzPollingMode\_T, [14](#)
  - QzSession\_T, [14](#)
  - QzSessionParams\_T, [14](#)
  - qzSetDefaults, [42](#)
  - qzSetSessionCrc64Config, [42](#)
  - qzSetupSession, [43](#)
  - QzSoftwareComponentType\_E, [18](#)
  - QzSoftwareComponentType\_T, [14](#)
  - QzStatus\_T, [14](#)
  - QzStream\_T, [15](#)
  - qzTeardownSession, [44](#)
- data\_fmt
  - QzSessionParams\_S, [51](#)
  - QzSessionParamsDeflate\_S, [54](#)
- debug API, [45](#)
- direction
  - QzSessionParams\_S, [51](#)
  - QzSessionParamsCommon\_S, [53](#)
- huffman\_hdr
  - QzSessionParams\_S, [51](#)

- QzSessionParamsDeflate\_S, 55
- hw\_buff\_sz
  - QzSessionParams\_S, 51
  - QzSessionParamsCommon\_S, 53
- hw\_session\_stat
  - QzSession\_S, 49
- hw\_session\_status
  - QzStatus\_S, 57
- in
  - QzStream\_S, 59
- in\_sz
  - QzStream\_S, 59
- include/qatzip.h, 61
- include/qz\_utils.h, 69
- initial\_value
  - QzCrc64Config\_S, 48
- input\_sz\_thrshold
  - QzSessionParams\_S, 51
  - QzSessionParamsCommon\_S, 53
- internal
  - QzSession\_S, 49
- is\_sensitive\_mode
  - QzSessionParamsCommon\_S, 53
- lz4s\_mini\_match
  - QzSessionParamsLZ4S\_S, 55
- max\_forks
  - QzSessionParams\_S, 51
  - QzSessionParamsCommon\_S, 53
- memory\_allocated
  - QzStatus\_S, 57
- NONE
  - Data Compression API, 15
- opaque
  - QzStream\_S, 59
- out
  - QzStream\_S, 59
- out\_sz
  - QzStream\_S, 59
- pending\_in
  - QzStream\_S, 59
- pending\_out
  - QzStream\_S, 60
- PinMem\_E
  - Data Compression API, 15
- PinMem\_T
  - Data Compression API, 11
- PINNED\_MEM
  - Data Compression API, 15
- polling\_mode
  - QzSessionParamsCommon\_S, 53
- polynomial
  - QzCrc64Config\_S, 48
- qat\_hw\_count
  - QzStatus\_S, 57
- qat\_instance\_attach
  - QzStatus\_S, 57
- qat\_mem\_drvr
  - QzStatus\_S, 57
- qat\_service\_init
  - QzStatus\_S, 58
- QatThread\_S, 47
- qatzip.h
  - QATZIP\_API, 65
  - QATZIP\_API\_VERSION, 65
  - QZ\_BUF\_ERROR, 65
  - QZ\_DATA\_ERROR, 65
  - QZ\_DEFLATE, 65
  - QZ\_DISABLE\_SOFTWARE\_BACKUP, 65
  - QZ\_DISABLE\_SOFTWARE\_ONLY\_EXECUTION, 66
  - QZ\_DUPLICATE, 66
  - QZ\_ENABLE\_SOFTWARE\_BACKUP, 66
  - QZ\_ENABLE\_SOFTWARE\_ONLY\_EXECUTION, 66
  - QZ\_FAIL, 66
  - QZ\_FORCE\_SW, 66
  - QZ\_INTEG, 66
  - QZ\_LOW\_DEST\_MEM, 67
  - QZ\_LOW\_MEM, 67
  - QZ\_METADATA\_OVERFLOW, 67
  - QZ\_NO\_HW, 67
  - QZ\_NO\_INST\_ATTACH, 67
  - QZ\_NO\_MDRV, 67
  - QZ\_NO\_SW\_AVAIL, 67
  - QZ\_NONE, 67
  - QZ\_NOSW\_LOW\_MEM, 68
  - QZ\_NOSW\_NO\_HW, 68
  - QZ\_NOSW\_NO\_INST\_ATTACH, 68
  - QZ\_NOSW\_NO\_MDRV, 68
  - QZ\_NOSW\_UNSUPPORTED\_FMT, 68
  - QZ\_NOT\_SUPPORTED, 68
  - QZ\_OUT\_OF\_RANGE, 68
  - QZ\_PARAMS, 68
  - QZ\_POST\_PROCESS\_ERROR, 69
  - QZ\_TIMEOUT, 69
  - QZ\_UNSUPPORTED\_FMT, 69
- QATZIP\_API
  - qatzip.h, 65
- QATZIP\_API\_VERSION
  - qatzip.h, 65
- QATZIP\_API\_VERSION\_NUM\_MAJOR
  - Data Compression API, 9
- QATZIP\_API\_VERSION\_NUM\_MINOR
  - Data Compression API, 9
- QZ\_ADLER
  - Data Compression API, 15
- QZ\_BUF\_ERROR
  - qatzip.h, 65
- QZ\_BUSY\_POLLING
  - Data Compression API, 18
- QZ\_CRC32



- Data Compression API, 15
- QZ\_DATA\_ERROR
  - qatzip.h, 65
- QZ\_DEFLATE
  - qatzip.h, 65
- QZ\_DEFLATE\_4B
  - Data Compression API, 16
- QZ\_DEFLATE\_GZIP
  - Data Compression API, 16
- QZ\_DEFLATE\_GZIP\_EXT
  - Data Compression API, 16
- QZ\_DEFLATE\_RAW
  - Data Compression API, 16
- QZ\_DIR\_BOTH
  - Data Compression API, 16
- QZ\_DIR\_COMPRESS
  - Data Compression API, 16
- QZ\_DIR\_DECOMPRESS
  - Data Compression API, 16
- QZ\_DISABLE\_SOFTWARE\_BACKUP
  - qatzip.h, 65
- QZ\_DISABLE\_SOFTWARE\_ONLY\_EXECUTION
  - qatzip.h, 66
- QZ\_DUPLICATE
  - qatzip.h, 66
- QZ\_DYNAMIC\_HDR
  - Data Compression API, 17
- QZ\_ENABLE\_SOFTWARE\_BACKUP
  - qatzip.h, 66
- QZ\_ENABLE\_SOFTWARE\_ONLY\_EXECUTION
  - qatzip.h, 66
- QZ\_FAIL
  - qatzip.h, 66
- QZ\_FORCE\_SW
  - qatzip.h, 66
- QZ\_INTEG
  - qatzip.h, 66
- QZ\_LOW\_DEST\_MEM
  - qatzip.h, 67
- QZ\_LOW\_MEM
  - qatzip.h, 67
- QZ\_MAX\_STRING\_LENGTH
  - Data Compression API, 9
- QZ\_METADATA\_OVERFLOW
  - qatzip.h, 67
- QZ\_NO\_HW
  - qatzip.h, 67
- QZ\_NO\_INST\_ATTACH
  - qatzip.h, 67
- QZ\_NO\_MDRV
  - qatzip.h, 67
- QZ\_NO\_SW\_AVAIL
  - qatzip.h, 67
- QZ\_NONE
  - qatzip.h, 67
- QZ\_NOSW\_LOW\_MEM
  - qatzip.h, 68
- QZ\_NOSW\_NO\_HW
  - qatzip.h, 68
- QZ\_NOSW\_NO\_INST\_ATTACH
  - qatzip.h, 68
- QZ\_NOSW\_NO\_MDRV
  - qatzip.h, 68
- QZ\_NOSW\_UNSUPPORTED\_FMT
  - qatzip.h, 68
- QZ\_NOT\_SUPPORTED
  - qatzip.h, 68
- QZ\_OK
  - Data Compression API, 9
- QZ\_OUT\_OF\_RANGE
  - qatzip.h, 68
- QZ\_PARAMS
  - qatzip.h, 68
- QZ\_PERIODICAL\_POLLING
  - Data Compression API, 18
- QZ\_POST\_PROCESS\_ERROR
  - qatzip.h, 69
- QZ\_SKID\_PAD\_SZ
  - Data Compression API, 9
- QZ\_STATIC\_HDR
  - Data Compression API, 17
- QZ\_SW\_BACKUP\_BIT\_POSITION
  - Data Compression API, 10
- QZ\_SW\_EXECUTION\_BIT
  - Data Compression API, 10
- QZ\_TIMEOUT
  - qatzip.h, 69
- QZ\_UNSUPPORTED\_FMT
  - qatzip.h, 69
- qzAllocateMetadata
  - Data Compression API, 18
- qzCallback
  - QzSessionParamsLZ4S\_S, 55
- qzCallback\_external
  - QzSessionParamsLZ4S\_S, 56
- qzClose
  - Data Compression API, 19
- qzCompress
  - Data Compression API, 20
- qzCompressCrc
  - Data Compression API, 21
- qzCompressStream
  - Data Compression API, 22
- qzCompressWithMetadataExt
  - Data Compression API, 23
- QzCrc64Config\_S, 47
  - initial\_value, 48
  - polynomial, 48
  - reflect\_in, 48
  - reflect\_out, 48
  - xor\_out, 48
- QzCrc64Config\_T
  - Data Compression API, 11
- QzCrcType\_E
  - Data Compression API, 15
- QzCrcType\_T

- Data Compression API, 11
- QzDataFormat\_E
  - Data Compression API, 16
- QzDataFormat\_T
  - Data Compression API, 11
- qzDecompress
  - Data Compression API, 25
- qzDecompressCrc
  - Data Compression API, 26
- qzDecompressStream
  - Data Compression API, 27
- qzDecompressWithMetadataExt
  - Data Compression API, 28
- QzDirection\_E
  - Data Compression API, 16
- QzDirection\_T
  - Data Compression API, 11
- qzEndStream
  - Data Compression API, 29
- qzFree
  - Data Compression API, 30
- qzFreeMetadata
  - Data Compression API, 31
- qzGetDefaults
  - Data Compression API, 31
- qzGetSessionCrc64Config
  - Data Compression API, 32
- qzGetSoftwareComponentCount
  - Data Compression API, 33
- qzGetSoftwareComponentVersionList
  - Data Compression API, 34
- qzGetStatus
  - Data Compression API, 35
- QzHuffmanHdr\_E
  - Data Compression API, 16
- QzHuffmanHdr\_T
  - Data Compression API, 11
- qzInit
  - Data Compression API, 36
- qzLZ4SCallbackFn
  - Data Compression API, 12
- qzMalloc
  - Data Compression API, 38
- qzMemFindAddr
  - Data Compression API, 38
- 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, 14
- QzSession\_S, 48
  - hw\_session\_stat, 49
  - internal, 49
  - thd\_sess\_stat, 49
  - total\_in, 49
  - total\_out, 49
- QzSession\_T
  - Data Compression API, 14
- QzSessionParams\_S, 50
  - comp\_algorithm, 50
  - comp\_lvl, 50
  - data\_fmt, 51
  - direction, 51
  - huffman\_hdr, 51
  - hw\_buff\_sz, 51
  - input\_sz\_thrshold, 51
  - max\_forks, 51
  - req\_cnt\_thrshold, 51
  - strm\_buff\_sz, 52
  - sw\_backup, 52
  - wait\_cnt\_thrshold, 52
- QzSessionParams\_T
  - Data Compression API, 14
- QzSessionParamsCommon\_S, 52
  - comp\_algorithm, 52
  - comp\_lvl, 53
  - direction, 53
  - hw\_buff\_sz, 53
  - input\_sz\_thrshold, 53
  - is\_sensitive\_mode, 53
  - max\_forks, 53
  - polling\_mode, 53
  - req\_cnt\_thrshold, 54
  - strm\_buff\_sz, 54
  - sw\_backup, 54
  - wait\_cnt\_thrshold, 54
- QzSessionParamsDeflate\_S, 54
  - data\_fmt, 54
  - huffman\_hdr, 55
- QzSessionParamsLZ4\_S, 55
- QzSessionParamsLZ4S\_S, 55
  - lz4s\_mini\_match, 55
  - qzCallback, 55
  - qzCallback\_external, 56
- qzSetDefaults
  - 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, 57
  - algo\_sw, 57
  - hw\_session\_status, 57
  - memory\_allocated, 57
  - qat\_hw\_count, 57

- qat\_instance\_attach, [57](#)
- qat\_mem\_drvr, [57](#)
- qat\_service\_init, [58](#)
- using\_huge\_pages, [58](#)
- QzStatus\_T
  - Data Compression API, [14](#)
- QzStream\_S, [58](#)
  - crc\_32, [59](#)
  - crc\_type, [59](#)
  - in, [59](#)
  - in\_sz, [59](#)
  - opaque, [59](#)
  - out, [59](#)
  - out\_sz, [59](#)
  - pending\_in, [59](#)
  - pending\_out, [60](#)
  - reserved, [60](#)
- QzStream\_T
  - Data Compression API, [15](#)
- qzTeardownSession
  - Data Compression API, [44](#)
- reflect\_in
  - QzCrc64Config\_S, [48](#)
- reflect\_out
  - QzCrc64Config\_S, [48](#)
- req\_cnt\_thrshold
  - QzSessionParams\_S, [51](#)
  - QzSessionParamsCommon\_S, [54](#)
- reserved
  - QzStream\_S, [60](#)
- strm\_buff\_sz
  - QzSessionParams\_S, [52](#)
  - QzSessionParamsCommon\_S, [54](#)
- sw\_backup
  - QzSessionParams\_S, [52](#)
  - QzSessionParamsCommon\_S, [54](#)
- thd\_sess\_stat
  - QzSession\_S, [49](#)
- ThreadList\_S, [60](#)
- total\_in
  - QzSession\_S, [49](#)
- total\_out
  - QzSession\_S, [49](#)
- using\_huge\_pages
  - QzStatus\_S, [58](#)
- wait\_cnt\_thrshold
  - QzSessionParams\_S, [52](#)
  - QzSessionParamsCommon\_S, [54](#)
- xor\_out
  - QzCrc64Config\_S, [48](#)