1 Migrating from CycloneSSL 1.8.6 to 1.9.0

1.1 New features

CycloneSSL 1.9.0 adds support for TLS 1.3 protocol (RFC8446).

1.2 Project

In order to adapt to TLS 1.3, the architecture of the SSL/TLS library has changed.

When used in conjunction with TLS 1.0, 1.1 or 1.2, CycloneSSL 1.9.0 now depends on the following files:

- cyclone_ssl/tls.c
- cyclone_ssl/tls_cipher_suites.c
- cyclone_ssl/tls_handshake.c
- cyclone_ssl/tls_client.c
- cyclone_ssl/tls_client_extensions.c (new file)
- cyclone_ssl/tls_client_misc.c
- cyclone_ssl/tls_server.c
- cyclone_ssl/tls_server_extensions.c (new file)
- cyclone_ssl/tls_server_misc.c
- cyclone_ssl/tls_common.c
- cyclone_ssl/tls_extensions.c (new file)
- cyclone_ssl/tls_certificate.c
- cyclone_ssl/tls_signature.c
- cyclone_ssl/tls_key_material.c
- cyclone_ssl/tls_transcript_hash.c (new file)
- cyclone_ssl/tls_cache.c
- cyclone_ssl/tls_ticket.c (new file)
- cyclone_ssl/tls_ffdhe.c
- cyclone_ssl/tls_record.c
- cyclone_ssl/tls_misc.c

The following file are deprecated and must be removed from your project:

- cyclone_ssl\tls_handshake_hash.c (to be removed)
- cyclone_ssl\tls_handshake_hash.h (to be removed)
- cyclone_ssl\tls_handshake_misc.c (to be removed)
- cyclone_ssl\tls_handshake_misc.h (to be removed)

In order to make use of TLS 1.3 (CycloneSSL Ultimate licensees), you must must add the following files to your project:

- cyclone_ssl/tls13_client.c (new file)
- cyclone_ssl/tls13_client_extensions.c (new file)
- cyclone_ssl/tls13_client_misc.c (new file)
- cyclone_ssl/tls13_server.c (new file)
- cyclone_ssl/tls13_server_extensions.c (new file)
- cyclone_ssl/tls13_server_misc.c (new file)
- cyclone_ssl/tls13_common.c (new file)
- cyclone_ssl/tls13_key_material.c (new file)
- cyclone_ssl/tls13_misc.c (new file)

1.3 Configuration header

The compilation flags that are used to enable/disable TLS key exchange methods (RSA, DHE_RSA, ECDHE_RSA, ECDHE_ECDSA, etc) have been renamed to avoid potential name conflicts.

It is highly recommended to use the new parameter names in your tls_config.h configuration file, although CycloneSSL 1.9.0 continue to support old parameter names (for compatibility purpose):

Deprecated parameters	New parameters
TLS_RSA_SUPPORT	TLS_RSA_KE_SUPPORT
TLS_DHE_RSA_SUPPORT	TLS_DHE_RSA_KE_SUPPORT
TLS_DHE_DSS_SUPPORT	TLS_DHE_DSS_KE_SUPPORT
TLS_DH_ANON_SUPPORT	TLS_DH_ANON_KE_SUPPORT
TLS_ECDHE_RSA_SUPPORT	TLS_ECDHE_RSA_KE_SUPPORT
TLS_ECDHE_ECDSA_SUPPORT	TLS_ECDHE_ECDSA_KE_SUPPORT
TLS_ECDH_ANON_SUPPORT	TLS_ECDH_ANON_KE_SUPPORT
TLS_PSK_SUPPORT	TLS_PSK_KE_SUPPORT
TLS_RSA_PSK_SUPPORT	TLS_RSA_PSK_KE_SUPPORT
TLS_DHE_PSK_SUPPORT	TLS_DHE_PSK_KE_SUPPORT
TLS_ECDHE_PSK_SUPPORT	TLS_ECDHE_PSK_KE_SUPPORT

2 Migrating from CycloneSSL 1.8.2 to 1.8.6

No changes required. Version 1.8.6 can be used as drop-in replacement.

3 Migrating from CycloneSSL 1.8.0 to 1.8.2

No changes required. Version 1.8.2 can be used as drop-in replacement.

4 Migrating from CycloneSSL 1.7.8 to 1.8.0

4.1 New features

CycloneSSL 1.8.0 is a major update that adds support for DTLS (Datagram Transport Layer Security. DTLS version 1.0 and 1.2 are supported. This release also supports the following features:

- Extended Master Secret extension
- Max Fragment Length extension
- Fallback SCSV

4.2 Project

The SSL/TLS library now depends on the following files:

- cyclone_ssl/tls.c
- cyclone_ssl/tls_cipher_suites.c
- cyclone_ssl/tls_handshake.c (new file)
- cyclone_ssl/tls_handshake_hash. c (new file)
- cyclone_ssl/tls_handshake_misc. c (new file)
- cyclone_ssl/tls_common.c
- cyclone_ssl/tls_record.c
- cyclone_ssl/tls_signature. c (new file)
- cyclone_ssl/tls_certificate. c (new file)
- cyclone_ssl/tls_key_material. c (new file)
- cyclone_ssl/tls_misc.c
- cyclone_ssl/tls_cache.c
- cyclone_ssl/ssl_misc. c (new file)
- cyclone_ssl/dtls_record. c (new file, only for CycloneSSL Pro and Ultimate licensees)
- cyclone_ssl/dtls_misc. c (new file, only for CycloneSSL Pro and Ultimate licensees)

The following file are deprecated and must be removed from your project:

- cyclone_ssl\ssl_common.c (to be removed)
- cyclone_ssl\ssl_common.h (to be removed)

4.3 Crypto library

The “cyclone_crypto” folder has been reorganized. Cryptographic algorithms are now sorted by family (hash, cipher, MAC, etc). Your project or makefile must be modified in order to reflect the new file structure:

Old filenames (version 1.7.8)	New filenames (version 1.8.0)
cyclone_crypto/crypto.h	cyclone_crypto/core/crypto.h
cyclone_crypto/md5.c cyclone_crypto/md5.h	cyclone_crypto/hash/md5.c cyclone_crypto/hash/md5.h
cyclone_crypto/sha1.c cyclone_crypto/sha1.h	cyclone_crypto/hash/sha1.c cyclone_crypto/hash/sha1.h
cyclone_crypto/sha224.c cyclone_crypto/sha224.h	cyclone_crypto/hash/sha224.c cyclone_crypto/hash/sha224.h
cyclone_crypto/sha256.c cyclone_crypto/sha256.h	cyclone_crypto/hash/sha256.c cyclone_crypto/hash/sha256.h
cyclone_crypto/sha384.c cyclone_crypto/sha384.h	cyclone_crypto/hash/sha384.c cyclone_crypto/hash/sha384.h
cyclone_crypto/sha512.c cyclone_crypto/sha512.h	cyclone_crypto/hash/sha512.c cyclone_crypto/hash/sha512.h
cyclone_crypto/hmac.c cyclone_crypto/hmac.h	cyclone_crypto/mac/hmac.c cyclone_crypto/mac/hmac.h
cyclone_crypto/poly1305.c cyclone_crypto/poly1305.h	cyclone_crypto/mac/poly1305.c cyclone_crypto/mac/poly1305.h
cyclone_crypto/rc4.c cyclone_crypto/rc4.h	cyclone_crypto/cipher/rc4.c cyclone_crypto/cipher/rc4.h
cyclone_crypto/idea.c cyclone_crypto/idea.h	cyclone_crypto/cipher/idea.c cyclone_crypto/cipher/idea.h
cyclone_crypto/des.c cyclone_crypto/des.h	cyclone_crypto/cipher/des.c cyclone_crypto/cipher/des.h
cyclone_crypto/des3.c cyclone_crypto/des3.h	cyclone_crypto/cipher/des3.c cyclone_crypto/cipher/des3.h
cyclone_crypto/aes.c cyclone_crypto/aes.h	cyclone_crypto/cipher/aes.c cyclone_crypto/cipher/aes.h
cyclone_crypto/camellia.c cyclone_crypto/camellia.h	cyclone_crypto/cipher/camellia.c cyclone_crypto/cipher/camellia.h
cyclone_crypto/aria.c cyclone_crypto/aria.h	cyclone_crypto/cipher/aria.c cyclone_crypto/cipher/aria.h
cyclone_crypto/seed.c cyclone_crypto/seed.h	cyclone_crypto/cipher/seed.c cyclone_crypto/cipher/seed.h
cyclone_crypto/chacha.c cyclone_crypto/chacha.h	cyclone_crypto/cipher/chacha.c cyclone_crypto/cipher/chacha.h
cyclone_crypto/cipher_mode_cbc.c cyclone_crypto/cipher_mode_cbc.h	cyclone_crypto/cipher_mode/cbc.c cyclone_crypto/cipher_mode/cbc.h
cyclone_crypto/cipher_mode_ccm.c cyclone_crypto/cipher_mode_ccm.h	cyclone_crypto/aead/ccm.c cyclone_crypto/aead/ccm.h
cyclone_crypto/cipher_mode_gcm.c cyclone_crypto/cipher_mode_gcm.h	cyclone_crypto/aead/gcm.c cyclone_crypto/aead/gcm.h
cyclone_crypto/chacha20_poly1305.c cyclone_crypto/chacha20_poly1305.h	cyclone_crypto/aead/chacha20_poly1305.c cyclone_crypto/aead/chacha20_poly1305.h
cyclone_crypto/dh.c cyclone_crypto/dh.h	cyclone_crypto/pkc/dh.c cyclone_crypto/pkc/dh.h

cyclone_crypto/rsa.c cyclone_crypto/rsa.h	cyclone_crypto/pkc/rsa.c cyclone_crypto/pkc/rsa.h
cyclone_crypto/dsa.c cyclone_crypto/dsa.h	cyclone_crypto/pkc/dsa.c cyclone_crypto/pkc/dsa.h
cyclone_crypto/mpi.c cyclone_crypto/mpi.h	cyclone_crypto/mpi/mpi.c cyclone_crypto/mpi/mpi.h
cyclone_crypto/base64.c cyclone_crypto/base64.h	cyclone_crypto/encoding/base64.c cyclone_crypto/encoding/base64.h
cyclone_crypto/asn1.c cyclone_crypto/asn1.h	cyclone_crypto/encoding/asn1.c cyclone_crypto/encoding/asn1.h
cyclone_crypto/oid.c cyclone_crypto/oid.h	cyclone_crypto/encoding/oid.c cyclone_crypto/encoding/oid.h
cyclone_crypto/pem.c cyclone_crypto/pem.h	cyclone_crypto/certificate/pem_import.c cyclone_crypto/certificate/pem_import.h
cyclone_crypto/x509.c cyclone_crypto/x509.h	cyclone_crypto/certificate/x509_common.c cyclone_crypto/certificate/x509_common.h cyclone_crypto/certificate/x509_cert_import.c cyclone_crypto/certificate/x509_cert_import.h cyclone_crypto/certificate/x509_cert_validate.c cyclone_crypto/certificate/x509_cert_validate.h
cyclone_crypto/yarrow.c cyclone_crypto/yarrow.h	cyclone_crypto/rng/yarrow.c cyclone_crypto/rng/yarrow.h

Remark: The x509.c source file has been replaced by a set of 3 distinct files (x509_common.c, x509_cert_parse.c and x509_cert_validate.c)

A new naming convention is used for ASM files (mpi_architecture_compiler.extension). If assembler optimization is used in your project to accelerate MPI calculation, then your makefile must be modified as follows:

Old filenames (version 1.7.8)	New filenames (version 1.8.0)
cyclone_crypto/mpi_asm_keil_arm7.s	cyclone_crypto/mpi/mpi_arm_v4_keil.s
cyclone_crypto/mpi_asm_keil_cortex_m3.s	cyclone_crypto/mpi/mpi_arm_v7m_keil.s
cyclone_crypto/mpi_asm_iar_cortex_m3.s	cyclone_crypto/mpi/mpi_arm_v7m_iar.s
cyclone_crypto/mpi_asm_gcc_cortex_m3.S	cyclone_crypto/mpi/mpi_arm_v7m_gcc.S
cyclone_crypto/mpi_asm_gcc_cortex_a9.S	cyclone_crypto/mpi/mpi_arm_v7a_gcc.S
cyclone_crypto/mpi_asm_gcc_cortex_mips.S	cyclone_crypto/mpi/mpi_mips_gcc.S

5 Migrating from CycloneSSL 1.7.6 to 1.7.8

CycloneSSL 1.7.8 can be used as a drop-in replacement for version 1.7.6.

6 Migrating from CycloneSSL 1.7.4 to 1.7.6

6.1 New features

CycloneSSL 1.7.6 supports some new features:

- Non-blocking operation (bare metal environments)
- ALPN TLS extension
- PSK key exchange (Plain PSK as well as RSA_PSK, DHE_PSK, ECDHE_PSK cipher suites are supported)
- New I/O abstraction layer

6.2 File name conflict with some GCC toolchains

It has been reported that the header file endian.h conflicts with some GCC toolchains. The following files must be removed from the project:

- common/endian.c
- common/endian.h

The following files must be added to the project in replacement of endian.c and .h:

- common/cpu_endian.c
- common/cpu_endian.h

For big-endian MCUs (AVR32, Coldfire V2), the preprocessor macro _BIG_ENDIAN must be removed from C project properties. _CPU_BIG_ENDIAN must be used as replacement in C project properties. This modification does not concern little-endian MCUs (Cortex-M3/4/7, Cortex-A5/8/9, ARM7, ARM9, PIC32, RX600)

6.3 Project

The following files must be added to your project:

- cyclone_ssl/tls_client_misc.c <= New file
- cyclone_ssl/tls_client_misc.h <= New file
- cyclone_ssl/tls_server_misc.c <= New file
- cyclone_ssl/tls_server_misc.h <= New file

The following files must be removed from your project:

- cyclone_ssl/tls_io.c <= Removed file
- cyclone_ssl/tls_io.h <= Removed file

6.4 Non-blocking operation

CycloneSSL now supports RTOS-less operation. As a consequence the prototype of the tlsWrite() function has changed. In RTOS-less mode, a call to tlsWrite() may have written less data bytes that requested. As a result a new parameter "written" has been added and reflects the actual number of bytes that have been written. This parameter is optional and can be NULL when a RTOS is being used.

```
error = tlsWrite(tlsContext, buffer, length, NULL, 0)
```

6.5 I/O abstraction layer

The SSL does not depend anymore on CycloneTCP sockets or any other socket interface. Two callback functions can be registered to redirect low-level read/write operations (at socket level).

If CycloneTCP native socket are used, you may continue to call tlsSetSocket() in order to bind the SSL stack to a particular socket.

```
Socket *socket;

//Open a socket
socket = socketOpen(SOCKET_TYPE_STREAM, SOCKET_IP_PROTO_TCP);
...
//Bind SSL to the relevant socket
error = tlsSetSocket(tlsContext, socket);
```

If BSD sockets are used (or any other I/O layer), you must register I/O callback functions before calling tlsConnect():

```
SOCKET sock;

//Open a socket
sock = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
...
//Set send and receive callbacks (I/O abstraction layer)
error = tlsSetIoCallbacks(tlsContext, (TlsIoHandle) sock,
   tlsSendCallback, tlsReceiveCallback);
```

Here is an example of implementation for the send and receive callback functions:

```
/**
 * @brief TLS send callback (I/O abstraction layer)
 * @param[in] socket Handle that identifies a socket
 * @param[in] data Pointer to a buffer containing the data to be transmitted
 * @param[in] length Number of data bytes to send
 * @param[out] written Number of bytes that have been transmitted
 * @param[in] flags Unused parameter
 * @return Error code
 **/

error_t tlsSendCallback(TlsIoHandle socket, const void *data,
   size_t length, size_t *written, uint_t flags)
{
   int_t n;
   error_t error;

   //Send data
   n = send((SOCKET) socket, data, length, 0);

   //Check return value
   if(n > 0)
   {
      //Total number of data that have been written
      *written = n;
```

```
      //Successful write operation
      error = NO_ERROR;
   }
   else
   {
#if defined(_WIN32)
      //Timeout error?
      if(WSAGetLastError() == WSAEWOULDBLOCK)
         error = ERROR_TIMEOUT;
      else
         error = ERROR_WRITE_FAILED;
#else
      //Timeout error?
      if(errno == EAGAIN || errno == EWOULDBLOCK)
         error = ERROR_TIMEOUT;
      else
         error = ERROR_WRITE_FAILED;
#endif
   }

   //Return status code
   return error;
}


/**
 * @brief TLS receive callback (I/O abstraction layer)
 * @param[in] socket Handle that identifies a socket
 * @param[out] data Buffer where to store the incoming data
 * @param[in] size Maximum number of bytes that can be received
 * @param[out] received Number of bytes that have been received
 * @param[in] flags Unused parameter
 * @return Error code
 **/

error_t tlsReceiveCallback(TlsIoHandle socket, void *data,
   size_t size, size_t *received, uint_t flags)
{
   int_t n;
   error_t error;

   //Send data
   n = recv((SOCKET) socket, data, size, 0);

   //Check return value
   if(n > 0)
   {
      //Total number of data that have been received
      *received = n;
      //Successful write operation
      error = NO_ERROR;
   }
   else
   {
#if defined(_WIN32)
      //Timeout error?
      if(WSAGetLastError() == WSAEWOULDBLOCK)
```

```
         error = ERROR_TIMEOUT;
      else
         error = ERROR_READ_FAILED;
#else
      //Timeout error?
      if(errno == EAGAIN || errno == EWOULDBLOCK)
         error = ERROR_TIMEOUT;
      else
         error = ERROR_READ_FAILED;
#endif
   }

   //Return status code
   return error;
}
```

7 Migrating from CycloneSSL 1.7.2 to 1.7.4

In the 1.7.4 release, CycloneSSL Lite, Pro and Ultimate packages only includes the cryptographic algorithms that are part of their respective license. Non SSL-related cryptographic algorithms can be evaluated by downloading the open-source package.

7.1 CycloneSSL Lite

The following source files are not part the CycloneSSL Lite license and should be removed from the project:

- cyclone_crypto\idea.c
- cyclone_crypto\camellia.c
- cyclone_crypto\aria.c
- cyclone_crypto\seed.c
- cyclone_crypto\cipher_mode_ccm.c
- cyclone_crypto\cipher_mode_gcm.c
- cyclone_crypto\dh.c
- cyclone_crypto\dsa.c
- cyclone_crypto\ec.c
- cyclone_crypto\ec_curves.c
- cyclone_crypto\ecdh.c
- cyclone_crypto\ecdsa.c

To ensure proper building, the following flags must be disabled in your tls_config.h file:

```
//DHE_RSA key exchange support
#define TLS_DHE_RSA_SUPPORT DISABLED
//DHE_DSS key exchange support
#define TLS_DHE_DSS_SUPPORT DISABLED
//DH_anon key exchange support
#define TLS_DH_ANON_SUPPORT DISABLED
//ECDHE_RSA key exchange support
#define TLS_ECDHE_RSA_SUPPORT DISABLED
//ECDHE_ECDSA key exchange support
#define TLS_ECDHE_ECDSA_SUPPORT DISABLED
//ECDH_anon key exchange support
#define TLS_ECDH_ANON_SUPPORT DISABLED

//DSA signature capability
#define TLS_DSA_SIGN_SUPPORT DISABLED
//ECDSA signature capability
#define TLS_ECDSA_SIGN_SUPPORT DISABLED

//IDEA cipher support
#define TLS_IDEA_SUPPORT DISABLED
//Camellia cipher support
#define TLS_CAMELLIA_SUPPORT DISABLED
//SEED cipher support
#define TLS_SEED_SUPPORT DISABLED
//ARIA cipher support
#define TLS_ARIA_SUPPORT DISABLED

//CCM AEAD support
```

```
#define TLS_CCM_CIPHER_SUPPORT DISABLED
//GCM AEAD support
#define TLS_GCM_CIPHER_SUPPORT DISABLED
//ChaCha20Poly1305 AEAD support
#define TLS_CHACHA20_POLY1305_SUPPORT DISABLED
```

The following features flags be disabled in your crypto_config.h file:

```
//IDEA support
#define IDEA_SUPPORT DISABLED
//Camellia support
#define CAMELLIA_SUPPORT DISABLED
//SEED support
#define SEED_SUPPORT DISABLED
//ARIA support
#define ARIA_SUPPORT DISABLED

//CCM mode support
#define CCM_SUPPORT DISABLED
//GCM mode support
#define GCM_SUPPORT DISABLED

//Chacha support
#define CHACHA_SUPPORT DISABLED
//Poly1305 support
#define POLY1305_SUPPORT DISABLED
//Chacha20Poly1305 support
#define CHACHA20_POLY1305_SUPPORT DISABLED

//Diffie-Hellman support
#define DH_SUPPORT DISABLED
//DSA support
#define DSA_SUPPORT DISABLED

//Elliptic curve cryptography support
#define EC_SUPPORT DISABLED
//ECDH support
#define ECDH_SUPPORT DISABLED
//ECDSA support
#define ECDSA_SUPPORT DISABLED
```

When updating the project tree, it is recommended to clean up the cyclone_crypto subdirectory before copying the new files. Only the files that are required by the SSL/TLS library will be copied. In order to speed up the building process, the following source files should be removed from your project:

- cyclone_crypto\md2.c
- cyclone_crypto\md4.c
- cyclone_crypto\ripemd128.c
- cyclone_crypto\ripemd128.c
- cyclone_crypto\tiger.c
- cyclone_crypto\whirlpool.c
- cyclone_crypto\sha512_224.c
- cyclone_crypto\sha512_256.c
- cyclone_crypto\cipher_mode_cfb.c

- cyclone_crypto\cipher_mode_ctr.c
- cyclone_crypto\cipher_mode_ecb.c
- cyclone_crypto\cipher_mode_ofb.c
- cyclone_crypto\pkcs5.c
- cyclone_crypto\rc6.c

7.2 CycloneSSL Pro

The following source files are not part the CycloneSSL Pro license and should be removed from the project:

- cyclone_crypto\ec.c
- cyclone_crypto\ec_curves.c
- cyclone_crypto\ecdh.c
- cyclone_crypto\ecdsa.c

To ensure proper building, the following flags must be disabled in your tls_config.h file:

```
//ECDHE_RSA key exchange support
#define TLS_ECDHE_RSA_SUPPORT DISABLED
//ECDHE_ECDSA key exchange support
#define TLS_ECDHE_ECDSA_SUPPORT DISABLED
//ECDH_anon key exchange support
#define TLS_ECDH_ANON_SUPPORT DISABLED

//ECDSA signature capability
#define TLS_ECDSA_SIGN_SUPPORT DISABLED

//ChaCha20Poly1305 AEAD support
#define TLS_CHACHA20_POLY1305_SUPPORT DISABLED
```

The following flags must be disabled in your crypto_config.h file:

```
//Chacha support
#define CHACHA_SUPPORT DISABLED
//Poly1305 support
#define POLY1305_SUPPORT DISABLED
//Chacha20Poly1305 support
#define CHACHA20_POLY1305_SUPPORT DISABLED

//Elliptic curve cryptography support
#define EC_SUPPORT DISABLED
//ECDH support
#define ECDH_SUPPORT DISABLED
//ECDSA support
#define ECDSA_SUPPORT DISABLED
```

When updating the project tree, it is recommended to clean up the cyclone_crypto subdirectory before copying the new files. Only the files that are required by the SSL/TLS library will be copied. In order to speed up the building process, the following source files should be removed from your project:

- cyclone_crypto\md2.c
- cyclone_crypto\md4.c

- cyclone_crypto\ripemd128.c
- cyclone_crypto\ripemd128.c
- cyclone_crypto\tiger.c
- cyclone_crypto\whirlpool.c
- cyclone_crypto\sha512_224.c
- cyclone_crypto\sha512_256.c
- cyclone_crypto\cipher_mode_cfb.c
- cyclone_crypto\cipher_mode_ctr.c
- cyclone_crypto\cipher_mode_ecb.c
- cyclone_crypto\cipher_mode_ofb.c
- cyclone_crypto\pkcs5.c
- cyclone_crypto\rc6.c

7.3 CycloneSSL Ultimate

When updating the project tree, it is recommended to clean up the cyclone_crypto subdirectory before copying the new files. Only the files that are required by the SSL/TLS library will be copied. In order to speed up the building process, the following source files should be removed from your project:

- cyclone_crypto\md2.c
- cyclone_crypto\md4.c
- cyclone_crypto\ripemd128.c
- cyclone_crypto\ripemd128.c
- cyclone_crypto\tiger.c
- cyclone_crypto\whirlpool.c
- cyclone_crypto\sha512_224.c
- cyclone_crypto\sha512_256.c
- cyclone_crypto\cipher_mode_cfb.c
- cyclone_crypto\cipher_mode_ctr.c
- cyclone_crypto\cipher_mode_ecb.c
- cyclone_crypto\cipher_mode_ofb.c
- cyclone_crypto\pkcs5.c
- cyclone_crypto\rc6.c

