```c
// Common U2F HID transport header - Review Draft
// 2014-10-08
// Editor: Jakob Ehrensvard, Yubico, jakob@yubico.com

#ifndef __U2FHID_H_INCLUDED__
#define __U2FHID_H_INCLUDED__

#ifdef _MSC_VER  // Windows
typedef unsigned char     uint8_t;
typedef unsigned short    uint16_t;
typedef unsigned int      uint32_t;
typedef unsigned long int uint64_t;
#else
#include <stdint.h>
#endif

#ifdef __cplusplus
extern "C" {
#endif

// Size of HID reports

#define HID_RPT_SIZE            64      // Default size of raw HID report

// Frame layout - command- and continuation frames

#define CID_BROADCAST           0xffffffff // Broadcast channel id

#define TYPE_MASK               0x80    // Frame type mask
#define TYPE_INIT               0x80    // Initial frame identifier
#define TYPE_CONT               0x00    // Continuation frame identifier

typedef struct {
  uint32_t cid;                        // Channel identifier
  union {
    uint8_t type;                      // Frame type - b7 defines type
    struct {
      uint8_t cmd;                     // Command - b7 set
      uint8_t bcnth;                   // Message byte count - high part
      uint8_t bcntl;                   // Message byte count - low part
      uint8_t data[HID_RPT_SIZE - 7];  // Data payload
    } init;
    struct {
      uint8_t seq;                     // Sequence number - b7 cleared
      uint8_t data[HID_RPT_SIZE - 5];  // Data payload
    } cont;
  };
} U2FHID_FRAME;

#define FRAME_TYPE(f) ((f).type & TYPE_MASK)
#define FRAME_CMD(f)  ((f).init.cmd & ~TYPE_MASK)
#define MSG_LEN(f)    ((f).init.bcnth*256 + (f).init.bcntl)
#define FRAME_SEQ(f)  ((f).cont.seq & ~TYPE_MASK)

// HID usage- and usage-page definitions

#define FIDO_USAGE_PAGE         0xf1d0  // FIDO alliance HID usage page
#define FIDO_USAGE_U2FHID       0x01    // U2FHID usage for top-level collection
#define FIDO_USAGE_DATA_IN      0x20    // Raw IN data report
#define FIDO_USAGE_DATA_OUT     0x21    // Raw OUT data report
```

```
// General constants

#define U2FHID_IF_VERSION       2       // Current interface implementation version
#define U2FHID_TRANS_TIMEOUT    3000    // Default message timeout in ms

// U2FHID native commands

#define U2FHID_PING          (TYPE_INIT | 0x01)  // Echo data through local processor
only
#define U2FHID_MSG           (TYPE_INIT | 0x03)  // Send U2F message frame
#define U2FHID_LOCK          (TYPE_INIT | 0x04)  // Send lock channel command
#define U2FHID_INIT          (TYPE_INIT | 0x06)  // Channel initialization
#define U2FHID_WINK          (TYPE_INIT | 0x08)  // Send device identification wink
#define U2FHID_SYNC          (TYPE_INIT | 0x3c)  // Protocol resync command
#define U2FHID_ERROR         (TYPE_INIT | 0x3f)  // Error response

#define U2FHID_VENDOR_FIRST (TYPE_INIT | 0x40)  // First vendor defined command
#define U2FHID_VENDOR_LAST  (TYPE_INIT | 0x7f)  // Last vendor defined command

// U2FHID_INIT command defines

#define INIT_NONCE_SIZE         8       // Size of channel initialization challenge
#define CAPFLAG_WINK            0x01    // Device supports WINK command

typedef struct {
  uint8_t nonce[INIT_NONCE_SIZE];       // Client application nonce
} U2FHID_INIT_REQ;

typedef struct {
  uint8_t nonce[INIT_NONCE_SIZE];       // Client application nonce
  uint32_t cid;                         // Channel identifier
  uint8_t versionInterface;             // Interface version
  uint8_t versionMajor;                 // Major version number
  uint8_t versionMinor;                 // Minor version number
  uint8_t versionBuild;                 // Build version number
  uint8_t capFlags;                     // Capabilities flags
} U2FHID_INIT_RESP;

// U2FHID_SYNC command defines

typedef struct {
  uint8_t nonce;                        // Client application nonce
} U2FHID_SYNC_REQ;

typedef struct {
  uint8_t nonce;                        // Client application nonce
} U2FHID_SYNC_RESP;

// Low-level error codes. Return as negatives.

#define ERR_NONE                0x00    // No error
#define ERR_INVALID_CMD         0x01    // Invalid command
#define ERR_INVALID_PAR         0x02    // Invalid parameter
#define ERR_INVALID_LEN         0x03    // Invalid message length
#define ERR_INVALID_SEQ         0x04    // Invalid message sequencing
#define ERR_MSG_TIMEOUT         0x05    // Message has timed out
#define ERR_CHANNEL_BUSY        0x06    // Channel busy
#define ERR_LOCK_REQUIRED       0x0a    // Command requires channel lock
#define ERR_SYNC_FAIL           0x0b    // SYNC command failed
#define ERR_OTHER               0x7f    // Other unspecified error
```

```
#ifdef __cplusplus
}
#endif

#endif  // __U2FHID_H_INCLUDED__
```