ngx_buf.h Documentation :

Macros Defined :

```
#define NGX_CHAIN_ERROR     (ngx_chain_t *) NGX_ERROR
#define ngx_buf_in_memory(b)        (b->temporary || b->memory || b->mmap)
#define ngx_buf_in_memory_only(b)   (ngx_buf_in_memory(b) && !b->in_file)
#define ngx_alloc_buf(pool)  ngx_palloc(pool, sizeof(ngx_buf_t))
#define ngx_calloc_buf(pool) ngx_pcalloc(pool, sizeof(ngx_buf_t))

#define ngx_buf_special(b)                                                   \
    ((b->flush || b->last_buf || b->sync)                                    \
     && !ngx_buf_in_memory(b) && !b->in_file)

#define ngx_buf_sync_only(b)                                                 \
    (b->sync                                                \
     && !ngx_buf_in_memory(b) && !b->in_file && !b->flush && !b->last_buf)

#define ngx_buf_size(b)                                                      \
    (ngx_buf_in_memory(b) ? (off_t) (b->last - b->pos):                      \
                            (b->file_last - b->file_pos))

#define ngx_free_chain(pool, cl)                                             \
    cl->next = pool->chain;                                                  \
    pool->chain = cl
```

Type Definitions :

```
typedef void *                    ngx_buf_tag_t;
typedef struct ngx_buf_s          ngx_buf_t;
typedef struct ngx_output_chain_ctx_s  ngx_output_chain_ctx_t;
typedef ngx_int_t (*ngx_output_chain_filter_pt)(void *ctx, ngx_chain_t *in);
```

Data Structures :

```
struct ngx_buf_s {
    u_char          *pos;
    u_char          *last;
    off_t           file_pos;
    off_t           file_last;
    u_char          *start;         ->Pointer to start of the buffer
    u_char          *end;           ->Pointer to end of the buffer
    ngx_buf_tag_t    tag;
    ngx_file_t      *file;
    ngx_buf_t       *shadow;

    unsigned        temporary:1;  -> indicates that buffer content could be changed
```

```
    unsigned         memory:1;     ->indicates  buffer content is in a memory cache or in a read only memory
                                      and must not be changed
    unsigned         mmap:1;        -> indicates buffer content is mmap()ed and must not be changed
    unsigned         recycled:1;
    unsigned         in_file:1;
    unsigned         flush:1;
    unsigned         sync:1;
    unsigned         last_buf:1;
    unsigned         last_in_chain:1;
    unsigned         last_shadow:1;
    unsigned         temp_file:1;
    int   num;
}

struct ngx_chain_s {
    ngx_buf_t      *buf;
    ngx_chain_t   *next;
}

typedef struct {
    ngx_int_t    num;
    size_t         size;
} ngx_bufs_t;

struct ngx_output_chain_ctx_s {
    ngx_buf_t                   *buf;
    ngx_chain_t                 *in;
    ngx_chain_t                 *free;
    ngx_chain_t                 *busy;

    unsigned                     sendfile:1;
    unsigned                    directio:1;
#if (NGX_HAVE_ALIGNED_DIRECTIO)
    unsigned                    unaligned:1;
#endif
    unsigned                    need_in_memory:1;
    unsigned                    need_in_temp:1;
#if (NGX_HAVE_FILE_AIO)
    unsigned                    aio:1;
    ngx_output_chain_aio_pt      aio_handler;
#endif
    off_t                       alignment;
    ngx_pool_t                  *pool;
    ngx_int_t                   allocated;
    ngx_bufs_t                  bufs;
    ngx_buf_tag_t               tag;
```

```
    ngx_output_chain_filter_pt   output_filter;
    void                        *filter_ctx;
}

typedef struct {
    ngx_chain_t                 *out;
    ngx_chain_t                 **last;
    ngx_connection_t            *connection;
    ngx_pool_t                  *pool;
    off_t                       limit;
} ngx_chain_writer_ctx_t;
```

Functions Defined :

```
ngx_buf_t *ngx_create_temp_buf( ngx_pool_t *pool,
                                size_t          size
                              )

ngx_chain_t *ngx_create_chain_of_bufs( ngx_pool_t *pool,
                                       ngx_bufs_t  *bufs
                                     )

ngx_chain_t *ngx_alloc_chain_link( ngx_pool_t *pool)

ngx_int_t ngx_output_chain( ngx_output_chain_ctx_t   *ctx,
                            ngx_chain_t              *in
                          )
ngx_int_t ngx_chain_writer( void            *ctx,
                            ngx_chain_t  *in
                          )

ngx_int_t ngx_chain_add_copy( ngx_pool_t   *pool,
                              ngx_chain_t  **chain,
                              ngx_chain_t  *in
                            )

ngx_chain_t *ngx_chain_get_free_buf( ngx_pool_t  *p,
                                     ngx_chain_t **free
                                   )

void ngx_chain_update_chains( ngx_pool_t     *p,
                              ngx_chain_t    **free,
                              ngx_chain_t    **busy,
                              ngx_chain_t    **out,
                              ngx_buf_tag_t tag
                            )
```

Include Dependency Graph :