

Contents

1	Module Aio : libaio-ocaml Linux async I/O interface for ocaml	1
2		1

1 Module Aio : libaio-ocaml Linux async I/O interface for ocaml

This module implements the libaio bindings that interface with the Linux system calls.

Version 0.0.0 - goswin-v-b@web.de

2

```
module Buffer :
sig
  type t = (int, Bigarray.int8_unsigned_elt, Bigarray.c_layout) Bigarray.Array1.t
    Page aligned buffer for use with Aio.

  exception Unaligned
    Exception when attempting unaligned access.

  val page_size : unit -> int
    Systems page size. Buffers must be multiples of it.

  val create : int -> t
    Allocate an uninitialized buffer.

  val clear : t -> unit
    zero fill a buffer and rewind

  val length : t -> int
    Length of the buffer.

  val get_int8 : t -> int -> int
  val get_uint8 : t -> int -> int
  val get_int16 : t -> int -> int
  val get_uint16 : t -> int -> int
  val get_int31 : t -> int -> int
  val get_int32 : t -> int -> int32
  val get_int64 : t -> int -> int64
```

Access with alignment and range checking

```
val unsafe_get_uint8 : t -> int -> int
val unsafe_get_int8 : t -> int -> int
val unsafe_get_uint16 : t -> int -> int
val unsafe_get_int16 : t -> int -> int
val unsafe_get_int31 : t -> int -> int
val unsafe_get_int32 : t -> int -> int32
val unsafe_get_int64 : t -> int -> int64
```

Unsafe access without checks.

```
val set_int8 : t -> int -> int -> unit
val set_uint8 : t -> int -> int -> unit
val set_int16 : t -> int -> int -> unit
val set_uint16 : t -> int -> int -> unit
val set_int31 : t -> int -> int -> unit
val set_int32 : t -> int -> int32 -> unit
val set_int64 : t -> int -> int64 -> unit
```

Access with alignment and range checking

```
val unsafe_set_uint8 : t -> int -> int -> unit
val unsafe_set_int8 : t -> int -> int -> unit
val unsafe_set_uint16 : t -> int -> int -> unit
val unsafe_set_int16 : t -> int -> int -> unit
val unsafe_set_uint16 : t -> int -> int -> unit
val unsafe_set_int16 : t -> int -> int -> unit
val unsafe_set_int31 : t -> int -> int -> unit
val unsafe_set_int32 : t -> int -> int32 -> unit
val unsafe_set_int64 : t -> int -> int64 -> unit
```

Unsafe access without checks.

```
val unsafe_get_substr : t -> int -> int -> string
```

unsafe extract string from buffer

```
val get_substr : t -> int -> int -> string
```

extract string from buffer

```
val get_str : t -> string
```

convert buffer to string

```
val unsafe_set_substr : t -> int -> string -> unit
    unsafe import string from buffer
```

```
val set_substr : t -> int -> string -> unit
    import string to buffer
```

```
val set_str : string -> t
    convert string to buffer
```

```
val get_be_int16 : t -> int -> int
val get_be_uint16 : t -> int -> int
val get_be_int31 : t -> int -> int
val get_be_int32 : t -> int -> int32
val get_be_int64 : t -> int -> int64
```

Access with alignment and range checking

```
val unsafe_get_be_int16 : t -> int -> int
val unsafe_get_be_uint16 : t -> int -> int
val unsafe_get_be_int31 : t -> int -> int
val unsafe_get_be_int32 : t -> int -> int32
val unsafe_get_be_int64 : t -> int -> int64
```

Unsafe access without checks.

```
val set_be_int16 : t -> int -> int -> unit
val set_be_uint16 : t -> int -> int -> unit
val set_be_int31 : t -> int -> int -> unit
val set_be_int32 : t -> int -> int32 -> unit
val set_be_int64 : t -> int -> int64 -> unit
```

Access with alignment and range checking

```
val unsafe_set_be_int16 : t -> int -> int -> unit
val unsafe_set_be_uint16 : t -> int -> int -> unit
val unsafe_set_be_int31 : t -> int -> int -> unit
val unsafe_set_be_int32 : t -> int -> int32 -> unit
val unsafe_set_be_int64 : t -> int -> int64 -> unit
```

Unsafe access without checks.

```

val get_le_int16 : t -> int -> int
val get_le_uint16 : t -> int -> int
val get_le_int31 : t -> int -> int
val get_le_int32 : t -> int -> int32
val get_le_int64 : t -> int -> int64

```

Access with alignment and range checking

```

val unsafe_get_le_int16 : t -> int -> int
val unsafe_get_le_uint16 : t -> int -> int
val unsafe_get_le_int31 : t -> int -> int
val unsafe_get_le_int32 : t -> int -> int32
val unsafe_get_le_int64 : t -> int -> int64

```

Unsafe access without checks.

```

val set_le_int16 : t -> int -> int -> unit
val set_le_uint16 : t -> int -> int -> unit
val set_le_int31 : t -> int -> int -> unit
val set_le_int32 : t -> int -> int32 -> unit
val set_le_int64 : t -> int -> int64 -> unit

```

Access with alignment and range checking

```

val unsafe_set_le_int16 : t -> int -> int -> unit
val unsafe_set_le_uint16 : t -> int -> int -> unit
val unsafe_set_le_int31 : t -> int -> int -> unit
val unsafe_set_le_int32 : t -> int -> int32 -> unit
val unsafe_set_le_int64 : t -> int -> int64 -> unit

```

Unsafe access without checks.

```

val get_net_int16 : t -> int -> int
val get_net_uint16 : t -> int -> int
val get_net_int31 : t -> int -> int
val get_net_int32 : t -> int -> int32
val get_net_int64 : t -> int -> int64

```

Access with alignment and range checking

```

val unsafe_get_net_int16 : t -> int -> int
val unsafe_get_net_uint16 : t -> int -> int
val unsafe_get_net_int31 : t -> int -> int
val unsafe_get_net_int32 : t -> int -> int32
val unsafe_get_net_int64 : t -> int -> int64

```

Unsafe access without checks.

```
val set_net_int16 : t -> int -> int -> unit
val set_net_uint16 : t -> int -> int -> unit
val set_net_int31 : t -> int -> int -> unit
val set_net_int32 : t -> int -> int32 -> unit
val set_net_int64 : t -> int -> int64 -> unit
```

Access with alignment and range checking

```
val unsafe_set_net_int16 : t -> int -> int -> unit
val unsafe_set_net_uint16 : t -> int -> int -> unit
val unsafe_set_net_int31 : t -> int -> int -> unit
val unsafe_set_net_int32 : t -> int -> int32 -> unit
val unsafe_set_net_int64 : t -> int -> int64 -> unit
```

Unsafe access without checks.

end

type result =

- | Result of Buffer.t
- | Errno of int
- | Partial of Buffer.t * int

The type for a result of a completed I/O request

exception Error of int

An error has occurred during a request.

exception Incomplete of Buffer.t * int

A request was only partially completed.

val result : result -> Buffer.t

Extract the Buffer.t from a result or throw the proper exception

type context

The type for a libaio Context.

val context : int -> context

Create a new context for n simultaneous requests.

val read :

context ->

Unix.file_descr -> int64 -> Buffer.t -> (result -> unit) -> unit

fill buffer from file at given offset and call continuation

```

val write :
  context ->
  Unix.file_descr -> int64 -> Buffer.t -> (result -> unit) -> unit
  write buffer to file at given offset and call continuation

val poll :
  context -> Unix.file_descr -> int -> (Unix.file_descr -> unit) -> unit
  poll file descriptor and call continuation

val run : context -> unit
  run the context till there are no more pending requests

val process : context -> unit
  process finished events and return

val fd : context -> Unix.file_descr
  return eventfd associated with the context

val sync_read : Unix.file_descr -> int64 -> Buffer.t -> unit
  fill buffer from file at given offset, blocking

val sync_write : Unix.file_descr -> int64 -> Buffer.t -> unit
  write buffer to file at given offset, blocking

```