

STDCL provides a simplified interface to OpenCL designed in a style familiar to conventional UNIX/C programmers.

The STDCL interface provides support for default contexts, a dynamic CL program loader, memory management, kernel management, and asynchronous operations.

Default CL Contexts

```
type(C_PTR) ~ CLCONTEXT* stddev
type(C_PTR) ~ CLCONTEXT* stdcpu
type(C_PTR) ~ CLCONTEXT* stdgpu
type(C_PTR) ~ CLCONTEXT* stdrpu

Default context for [ all | CPU | GPU | RPU ]
OpenCL supported devices.
```

Platform

```
integer(C_INT)
function clgetndev( clcontext )
    type(C_PTR) clcontext

Returns number of devices in context.
```

Dynamic CL Program Loader

```
type(C_PTR)
function clopen( clcontext, filename, flags )
    type(C_PTR) clcontext
    character(kind=C_CHAR) filename
    integer(C_INT) flags
    flags: CLLD_NOW, CLLD_BUILD

Build the OpenCL device program and return a
handle to the program.
```

```
type(C_PTR)
function clsopen( clcontext, srcstr, flags )
    type(C_PTR) clcontext
    character(kind=C_CHAR) srcstr
    integer(C_INT) flags
    flags: CLLD_NOW, CLLD_NOBUILD

Build the OpenCL device program and return a
handle to the program.
```

```
type(C_PTR) ~ cl_kernel
function clsym( clcontext, handle, symbol, flags )
    type(C_PTR) clcontext
    type(C_PTR) handle
    character(kind=C_CHAR) symbol
    integer(C_INT) flags
    flags: CLLD_NOW

Returns the kernel object identified by name
from the compiled OpenCL device program.
```

```
integer(C_INT)
function clclose( clcontext, handle )
    type(C_PTR) clcontext
    type(C_PTR) handle

Close the OpenCL device program and release
associated resources.
```

```
type(C_PTR)
function clbuild( clcontext, handle, options, flag )
    type(C_PTR) clcontext
    type(C_PTR) handle
    character(kind=C_CHAR) options
    integer(C_INT) flags

Build the OpenCL device program and return the
handle to the program.
```

Memory Management

```
type(C_PTR)
function clmalloc( clcontext, size, flags )
    type(C_PTR) clcontext
    integer(C_SIZE_T) size
    integer(C_INT) flag
    flags: CL_MEM_DETACHED

Allocate memory that can be shared across OpenCL
devices.
```

```
type(C_PTR)
function clmrealloc( clcontext, ptr, size, flags )
    type(C_PTR) clcontext
    type(C_PTR) ptr
    integer(C_SIZE_T) size
    integer(C_INT) flags
    flags: CL_MEM_DETACHED

Re-allocate (re-size) memory that can be shared
across OpenCL devices.
```

```
integer(C_INT)
function clfree( ptr )
    type(C_PTR) ptr

Free device-shareable memory allocated with
clmalloc() or an equivalent call.
```

```
type(C_PTR) ~ cl_event
function clmsync( clcontext, devnum, ptr, flags )
    type(C_PTR) clcontext
    integer(C_INT) devnum
    type(C_PTR) ptr
    integer(C_INT) flags
    flags: CL_MEM_HOST | CL_MEM_DEVICE,
           CL_EVENT_WAIT | CL_EVENT_NOWAIT,
           CL_EVENT_NORELEASE

Synchronize memory on host or OpenCL device,
performing a memory copy as necessary.
```

```
type(C_PTR) ~ cl_event
function clmcopy( clcontext, devnum, src, dst,
                  flags )
    type(C_PTR) clcontext
    integer(C_INT) devnum
    Type(C_PTR) src
    Type(C_PTR) dst
    integer(C_INT) flags
    flags: CL_EVENT_WAIT | CL_EVENT_NOWAIT,
           CL_EVENT_NORELEASE

Copy memory on an OpenCL device.
```

```
integer(C_INT)
function clmattach( clcontext, ptr )
    type(C_PTR) clcontext
    type(C_PTR) ptr

Attach device-shareable memory to context.

integer(C_INT)
function clmdetach( ptr )
    type(C_PTR) ptr

Detach device-shareable memory from context.
```

```
integer(C_SIZE_T)
function clsizeofmem( ptr )
    type(C_PTR) ptr

Return the size of device-shareable memory allocated
with clmalloc() or an equivalent call.
```

Kernel Management

```
type(clndrange_struct)
function clndrange_init[1|2|3] d( gtoff0, gtsz0, ltsz0
    [, gtoff1, gtsz1, ltsz1, [, gtoff2, gtsz2, ltsz2 ] ] )
    integer(C_INT) gtoff0 [, gtoff1 [, gtoff2 ] ]
    integer(C_INT) gtsz0 [, gtsz1 [, gtsz2 ] ]
    integer(C_INT) ltsz0 [, ltsz1 [, ltsz2 ] ]

Initialize N-dimensional range.
```

```
integer(C_INT)
function clarg_set( clcontext, krn, argnum, arg )
    type(C_PTR) clcontext
    type(C_PTR) krn
    integer(C_INT) argnum
    Tn arg

Set intrinsic argument of kernel.
```

```
integer(C_INT)
function clarg_set_global( clcontext, krn, argnum, ptr )
    type(C_PTR) clcontext
    type(C_PTR) krn
    integer(C_INT) argnum
    type(C_PTR) ptr

Set pointer argument of kernel.
```

```
type(C_PTR) ~ cl_event
function clfork( clcontext, devnum, krn, ndr_ptr, flags )
    type(C_PTR) clcontext
    integer(C_INT) devnum
    type(C_PTR) krn
    type(C_PTR) ndr_ptr ~ C_LOC(clndrange_struct)
    integer(C_INT) flags
    flags: CL_EVENT_WAIT | CL_EVENT_NOWAIT,
           CL_EVENT_NORELEASE

Fork kernel for execution on OpenCL device.
```

Synchronization

```
type(C_PTR)
function clflush( clcontext, devnum, flags )
    type(C_PTR) clcontext
    integer(C_INT) devnum
    integer(C_INT) flags
    flags: CL_KERNEL_EVENT, CL_MEM_EVENT,
           CL_ALL_EVENT, CL_EVENT_NORELEASE

Flush all enqueued operations (non-blocking).
```

```
type(C_PTR)
function clwait( clcontext, devnum, flags )
    type(C_PTR) clcontext
    integer(C_INT) devnum
    integer(C_INT) flags
    flags: CL_KERNEL_EVENT, CL_MEM_EVENT,
           CL_ALL_EVENT, CL_EVENT_NORELEASE

Block on all enqueued operations.
```

Notation:

~ type indicates the opaque type for which the C_PTR is used as a proxy since Fortran does not support type aliasing.
 [a | b | ...] indicates a choice between several alternatives and is not part of the syntax.