

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

```
CLCONTEXT* stddev
CLCONTEXT* stdcpu
CLCONTEXT* stdgpu
CLCONTEXT* stdrpu
```

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

Platform

```
int clgetndev( CLCONTEXT* clcontext )
Returns number of devices in context.
```

```
int clgetdevinfo( CLCONTEXT* clcontext,
  struct cldev_info* info )
Get information about each device in context.
```

Dynamic CL Program Loader

```
void* clopen( CLCONTEXT* clcontext,
  char* filename, int flags )
  flags: CLLD_NOW, CLLD_NOBUILD
Build the OpenCL device program and return a handle to the program.
```

```
void* clsopen( CLCONTEXT* clcontext,
  char* srcstr, int flags )
  flags: CLLD_NOW, CLLD_NOBUILD
Build the OpenCL device program and return a handle to the program.
```

```
cl_kernel clsym( CLCONTEXT* clcontext,
  void* handle, char* symbol, int flags )
  flags: CLLD_NOW
Returns the kernel object identified by name from the compiled OpenCL device program.
```

```
int clclose( CLCONTEXT* clcontext, void* handle )
Close the OpenCL device program and release associated resources.
```

```
void* clbuild( CLCONTEXT* clcontext,
  void* handle, char* options, int flags )
Build the OpenCL device program and return the handle to the program.
```

Memory Management

```
void* clmalloc( CLCONTEXT* clcontext,
  size_t size, int flags )
  flags: CL_MEM_DETACHED
```

Allocate memory that can be shared across OpenCL devices.

```
void* clmrealloc( CLCONTEXT* clcontext,
  void* ptr, size_t size, int flags )
  flags: CL_MEM_DETACHED
Re-allocate (re-size) memory that can be shared across OpenCL devices.
```

```
int clfree( void* ptr )
Free device-shareable memory allocated with clmalloc() or an equivalent call.
```

```
int clmctl( void* ptr, int op, ... )
int clmctl_va( void* ptr, int op, va_list )
  op: CL_MCTL_SET_IMAGE2D,
  CL_MCTL_SET_USERFLAGS,
  CL_MCTL_CLR_USERFLAGS
Perform general operations on device-shareable memory allocations.
```

```
cl_event clmsync( CLCONTEXT* clcontext,
  unsigned int devnum, void* ptr, 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.
```

```
cl_event clmcopy( CLCONTEXT* clcontext,
  unsigned int devnum, void* src, void* dst,
  int flags )
  flags: CL_EVENT_WAIT | CL_EVENT_NOWAIT,
  CL_EVENT_NORELEASE
Copy memory on an OpenCL device.
```

```
int clmattach( CLCONTEXT* clcontext, void* ptr )
Attach device-shareable memory to context.
```

```
int clmdetach( void* ptr )
Detach device-shareable memory from context.
```

```
size_t clsizeofmem( void* ptr )
Return the size of device-shareable memory allocated with clmalloc() or an equivalent call.
```

```
void* clgmalloc( CLCONTEXT* clcontext,
  cl_GLuint glbufobj, cl_GLenum target,
  cl_GLint miplevel, int flags )
  flags: CL_MEM_DETACHED,
  CL_MEM_GLBUF | CL_MEM_GLTEX2D
| CL_MEM_GLTEX3D | CL_MEM_GLRBUF
Allocate CL/GL interoperable memory that can be shared across devices.
```

```
cl_event clgmsync( CLCONTEXT* clcontext,
  unsigned int devnum, void* ptr, int flags )
  flags: CL_MEM_CLBUF | CL_MEM_GLBUF,
  CL_EVENT_WAIT | CL_EVENT_NOWAIT,
  CL_EVENT_NORELEASE
Synchronize CL/GL interoperable memory on device.
```

Kernel Management

```
clndrange_t clndrange_init[1|2|3] d(
  int gtoff0, int gtsz0, int ltsz0
  [, int gtoff1, int gtsz1, int ltsz1,
  [, int gtoff2, int gtsz2, int ltsz2 ] ] )
Initialize N-dimensional range.
```

```
void clarg_set( CLCONTEXT* clcontext, cl_kernel krn,
  unsigned int argnum, Tn arg )
Set intrinsic argument of kernel.
```

```
void clarg_set_global( CLCONTEXT* clcontext,
  cl_kernel krn, unsigned int argnum, void* ptr )
Set pointer argument of kernel.
```

```
cl_event clfork( CLCONTEXT* clcontext, unsigned
  int devnum, cl_kernel krn, clndrange_t* ndr_ptr,
  int flags )
  flags: CL_EVENT_WAIT | CL_EVENT_NOWAIT,
  CL_EVENT_NORELEASE
Fork kernel for execution on device.
```

```
cl_event clforka( CLCONTEXT* clcontext, unsigned
  int devnum, cl_kernel krn, clndrange_t* ndr_ptr,
  int flags [, arg0, ..., argn ] )
  flags: CL_EVENT_WAIT | CL_EVENT_NOWAIT,
  CL_EVENT_NORELEASE
Fork kernel for execution on device, setting kernel arguments as necessary.
```

Synchronization

```
cl_event clflush( CLCONTEXT* clcontext,
  unsigned int devnum, int flags )
  flags: CL_KERNEL_EVENT, CL_MEM_EVENT
  CL_ALL_EVENT, CL_EVENT_NORELEASE
Flush all enqueued operations (non-blocking).
```

```
cl_event clwait( CLCONTEXT* clcontext,
  unsigned int devnum, int flags )
  flags: CL_KERNEL_EVENT, CL_MEM_EVENT
  CL_ALL_EVENT, CL_EVENT_NORELEASE
Block on all enqueued operations.
```

Environment Variables

STDDEV, STDCPU, STDGPU, STDRPU
Enable/disable (1/0) default context.

STD[DEV|CPU|GPU|RPU]_PLATFORM_NAME
Select platform by name for default context.

STD[DEV|CPU|GPU|RPU]_MAX_NDEV
Limit number of devices in context.

STD[DEV|CPU|GPU|RPU]_LOCK
Set exclusive lock key for context.

Notation:

[a | b | ...] indicates a choice between several alternatives and is not part of the syntax.