|  |  |  |
| --- | --- | --- |
| **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 *flag*s )**  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\* clglmalloc(** 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 clglmsync(**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.

|  |  |  |
| --- | --- | --- |
|  | **STDCL 1.4 C/C++ Quick Reference Card** | |
| 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. |