QisMLib core APIs

July 9, 2018

- Dynamic library : **qismlib64.dll** (Windows 64), **libqism64.so** (Linux 64)
- All APIs are classic (C++ 98) object-oriented C++ designed to be simple, cross platform, extensible and always backwards compatible (source and binary) with a simple and consistent error handling scheme (error codes and error strings)
- Refer to the individual header files for detailed API reference
- APIs are individually licensed to suit the client's needs
- Only the notable API features are listed here
- Web Page: http://www.artwork.com/gdsii/qislib_mt/index.htm

QisMLib (qismlib.h)

Gateway to the QisMLib system. Use this to load one or more GDSII/OASIS/DBLOAD files and get access to other APIs in the system

Key operations	Objective
:: QisMLib_initialize_once	• Initialize the QisMLib once at the start of the application and get a handle to the only instance of the QisMLib object • Requires ONE license of **11003** • All extensions specified in the config file (default: qismlib.cfg) will be loaded during this operation
:: QisMLib_close_once	• Close the QisMLib once at the end of the application • Releases the held license 11003 • Any extensions loaded into the system will be unloaded during this operation
:: QisMLib_get_lib_info	• Get library version information
QisMLib::Load_file	• Load a GDSII/OASIS or DBLOAD file to create a new instance of the qismfile object (file database)
QisMLib::Unload_file	• Unload an existing qismfile database and free up the resources
QisMLib:: Get_extension_api	• Get access to any of the available APIs in the QisMLib system. Some of these are implemented in the core DL while others could be implemented in extension DLs loaded during initialization
QisMLib:: Get_extension_report	• Get a report (human readable string) of all extensions loaded and info/error/warning messages encountered during the loading process

QisMFile (qismfile.h)

API to work with a single GDSII/OASIS/DBLOAD file database loaded into QisMLib

Key operations	Objective
QisMFileLoadCtrl::Set_layer_map QisMFileLoadCtrl::Set_file_data_on_disk QisMFileLoadCtrl::Set_ignore_texts	• Specify various options that will be applied during file load - A map of layers to be loaded, keep the database on disk to reduce memory footprint, ignore TEXT items etc.
QisMFile::Get_default_top_cell	• Get the name of deepest root cell. This becomes the view cell when the file is loaded
QisMFile::Get_cell_list QisMFile::Get_top_cell_list QisMFile::Get_non_top_cell_list	• Get list of cells in the database
QisMFile:: Get_cell_children_list QisMFile:: Get_cell_extents QisMFile:: Get_cell_index	• Get information about a particular cell
QisMFile:: Get_layer_list	• Get a list of layers and corresponding datatypes available
QisMFile::Grid QisMFile::Units	• Get file units and resolution
QisMFile::Get_file_report	• Get a compact summary of the database as a human readable string
QisMFile::Create_exploder	• Create an instance of the exploder (spatial query object) to get data vectors (boundaries, paths, texts, cell references) from a specific view (cell, layers, window, nesting level) • Requires ONE license of **11027** per call to this function
QisMFile::Destroy_exploder	• Destroy an instance of the exploder object • Releases ONE license of 11027 held by this exploder object
QisMFileV2:: SaveAs_memory_maps	• Create a snapshot of this database on disk for faster loading next time • ONLY available for GDSII databases • If the database is loaded to memory, a DBLOAD cache is created. Otherwise, a LOAD cache is created • Both DBLOAD and LOAD caches are accompanied by a SCAN file. This file MUST always be present alongside the DBLOAD/LOAD caches • For LOAD cache, the original GDSII file MUST also be present alongside • It's best to create only DBLOAD maps since they are self contained and independent of the original GDSII data

QisMExploder (qismexploder.h)

API to use a single exploder object to perform spatial queries on a view and get data vectors crossing it

Key operations	Objective
QisMView:: Set_view_cell QisMView:: Set_layers_on QisMView:: Set_layers_off QisMView:: Set_text_on QisMView:: Set_exact_window QisMView:: Set_nesting_level	• Set the view for the spatial query
QisMExploder:: Set_convert_paths_to_boundaries QisMExploder:: Set_vector_fully_inside QisMExploder:: Set_array_mode	• Various settings to control what data vectors are returned
QisMExploder:: Get_vector_data	• Get data vectors in the specified view based on the specified settings • All data vectors are returned to the client in form of callbacks to the specified handler
QisMExploder:: Get_boundaries	• Get ONLY boundaries in the specified view in form of a QisMBStore (boundary storage) object
QisMExploder:: Get_cell_references	• Get ONLY references to the specified cell in the current view • All data vectors are returned to the client in form of callbacks to the specified handler
QisMExploder:: Get_cell_tree	• Get ONLY references belonging to the specified cell's tree (children cells) for the specified or all nesting levels • All data vectors are returned to the client in form of callbacks to the specified handler

QisMBool (qismbool.h)

API to perform multi-threaded boolean operations (union, difference, xor, intersection etc.) on large sets of polygons

Key operations	Objective
QisMBool::Create_instance	• Create an instance of the boolean object • Requires ONE license of **11047** per call to this function
QisMBool:: Destroy_instance	• Destroy an instance of the boolean object • Release ONE license of 11047 held by the boolean object
QisMBoolInst::UnionMT	• Unionize a set of polygons using multiple threads to create a new polygon set
QisMBoolInst:: BinaryMT	• Perform boolean operation (union, difference, xor,

	intersection) between two sets of polygons using multiple threads to create a new polygon set
QisMBoolInst:: BooleanST	- Perform boolean operation between two sets of polygons using ONE thread and optimized for small polygons sets (<10K vertices)
QisMBoolX:: Find_connected_sets	- Re-group polygons from an input set into sets of connected (touching/overlapping) polygons

QisMClipper (qismclipper.h)

API to clip a single boundary/path to a set of polygonal or rectangular windows

Key operations	Objective
QisMClipper:: Create_poly_instance	- Create an instance of the clipper object from a set of polygonal windows - Requires ONE license of **11047** per call to this function (same as QisMBool::Create_instance)
QisMClipper:: Create_box_instance	- Create an instance of the clipper object from a set of box (orthogonal rectangle) windows - Requires ONE license of 11047 per call to this function (same as QisMBool::Create_instance)
QisMClipper::Destroy_instance	- Destroy an instance of the clipper object - Release ONE license of 11047 held by the boolean object
QisMClipperObj::Clip_boundary QisMClipperObj:: Clip_boundaries	- Clip one or a set of boundaries to the windows assigned to this clipper
QisMClipperObj::Clip_path	- Clip one path to the windows assigned to this clipper resulting in a set of clipped boundaries
QisMClipperObj:: Boundary_interaction QisMClipperObj::Path_interaction	- Test the interaction of a boundary or a path with the windows assigned to this clipper object - Result could be Crossing/Fully inside/Enclosing/Disjoint
QisMClipperObj::Point_interaction	- Test the interaction of a point with the windows assigned to this clipper object - Result could be Outside/Inside/On an edge/Is a vertex

QisMDraw (qismdraw.h)

Render a view directly to the client screen or to an image buffer formatted as GIF/XPM/Windows Bitmap

Key operations	Objective
QisMDraw::Open_draw	- Create an instance of the draw object (only one per qismfile database)

	- Requires ONE license of **11057** per call to this function
QisMDraw::Close_draw	- Destroy an instance of the draw object - Releases ONE license of 11057 held by the draw object
QisMDrawObject::Set_display_filter_size QisMDrawObject::Set_fill QisMDrawObject::Set_backgound_color QisMDrawObject::Set_reference_marker QisMDrawObject::Set_text_marker QisMDrawObject::Set_geometry_marker QisMDrawObject::Set_marker_shape QisMDrawObject::Set_scale_bar QisMDrawObject::Set_cell_labels QisMDrawObject::Set_cell_outline QisMDrawObject::Set_layers_outline_color QisMDrawObject::Set_layers_fill_color	- Specify various rendering options to draw a view
QisMView:: Set_view_cell QisMView:: Set_layers_on QisMView:: Set_layers_off QisMView:: Set_text_on QisMView:: Set_nesting_level QisMDrawObject:: Set_ window	- Set the view for drawing
QisMDrawObject::Set_image_size	- Set the image size and aspect ratio for the drawing
QisMDrawObject::Redraw_direct	- Draw directly to a client screen (CWnd* - Windows, Pixmap - Linux)
QisMDrawObject::Redraw_image	- Draw to an image buffer formatted as GIF/XPM (Linux)/Windows Bitmap (Windows)
QisMDrawObject::Get_display_vector_data	- Get the data in the specified view as vectors (boundary, path, text, cell reference) and flash (data smaller than the display filter size) to enable the client to do their own drawing