Update IAutomation Interface PCB-Investigator 5.0

IAction:

new ID_ActionItems

`Undo last action if stack is not empty.`

ID_UNDO_LAST_ACTION = 131,

The current job will be added to a selected library from small dialog menu.

ID_ADD_CURRENT_JOB_TO_LIBRARY = 132,

Go through all layer and remove each double element who have a complete identically object (e.g. round pad with same diameter and location)

ID_REMOVE_DOUBLE_ELEMENTS = 133,

Generate netlist for all layers with connections through drills.

ID_GENERATE_NETLIST = 134,

Select nets like in shape mode and check connections through drills.

ID_NET_MODE_SHAPE_OVER_LAYERS = 135,

Change job name for misc info file.

ID_SET_JOB_NAME

CompareWorker:

Compare to layers graphically, internal it will be create two images of the layers and compare.

<param name="OutputWindow">The job window, for the output.</param>

<param name="layer1">The layer from the first job.</param>

<param name="Parent1">The parent window from the first job.</param>

<param name="layer2">The layer from the second job.</param>

<param name="Parent2">The parent window from the second job.</param>

<param name="CreateLayerIfSame"></param>

<param name="screenRect">The relevant area.</param>

<param name="Precisioned">It will be more accurate and slower if the value is bigger (1 is 5 m).</param>

<param name="ResultList">All results in one list.</param>

<param name="connectResultAreas">Add connected errorareas to one bigger area.</param>

<returns>true if the layers have the same visual nature.</returns>

public bool CompareLayers(IPCBIWindow OutputWindow, ILayer layer1, IPCBIWindow Parent1, ILayer layer2, IPCBIWindow Parent2, bool CreateLayerIfSame, RectangleF screenRect, int Precisioned, out List<ResultDifferencesCompareClass> ResultList, bool connectResultAreas)

and some other params for different methods of CompareLayers.

IAutmation:

Get the internal working directory for temp-data or unzip jobs etc.

<returns>directory path</returns>

public static string GetPCBIWorkingDirectory()

The first pin is drawn with transparent color, if you want to turn off this set the value to false.

This is saved in options and PCB-Investigator remembers last setting.

public static bool HighlightFirstPinOfCMPs

The unit factor for calculating the package insert point.

The insert point is most in the middle of the package. You see the insert point marked with a small yellow circle inside of the package outline.

e.g. 39.37f = mm; 1 = mils

public static float ComponentInsertPointUnitFactor

Special Library Mode use extra license.

<param name="LicenseKey">Key to check the license is ok.</param>

public static void SetLibraryMode(string LicenseKey)

Surfaces in transparent mode are hidden by default; you can change this by using ShowTransparentSurfaces or ask for the value if it is changed.

public static bool ShowTransparentSurfaces

Drill Tool Mixed Color setting, change for easy finding of drills.

public static bool DrillMixedColor

ICMPObject

Creates outline as IPolyClass.

<returns>Polygon of Object.</returns>

public override PCBI.MathUtils.IPolyClass GetPolygonOutline()

IEnum

New Types from ODB++ 8.0 format

public enum MatrixLayerAddType

Drill types for drill layers, new in ODB++ 8.0.

public enum IDrillType

Set color groups in attribute histogram

public class ColoredAttributeHistogram

Save groups of colorsettings.

public class ColoredAttributeGroup

Save Name of setting and RGBValue.

public class ColorPair

IMath

The middle pf two points.

<param name="FromP">The frist point.</param>

<param name="ToP">The second point.</param>

<returns>Middle of the two points.</returns>

static public PointF GetMidPoint(PointF FromP, PointF ToP)

Calculate the distance between two edges.

<param name="Edge1">first edge, can be arc or line</param>

<param name="Edge2">second edge, can be arc or line</param>

<param name="From">the start point of shortest way between the two edges</param>

<param name="To">end point of shortest way</param>

<returns>distance between from and to</returns>

static public double DistanceEdgeToEdge(IEdge Edge1, IEdge Edge2, ref PointD From, ref PointD To)

IMatrix

Copy the layer

<param name="LayerName">old layer name</param>

<param name="newLayerName">name of the second layer</param>

<returns>name of layer, it's possible that it is different to newLayerName</returns>

public string CopyLayer(string LayerName, string newLayerName)

Works only from different thread, to break calculation like MakePositive or GetShortsOnNegativLayer.

public void BreakCalculation()

The matrix layer add_type returns unknown if the LayerName or the matrix are unknown.

This property is new since ODB++ 8.0.

<param name="LayerName">Name of relevant layer</param>

<returns>MatrixLayerAddType (enum)</returns>

public MatrixLayerAddType GetMatrixLayerAddType(string LayerName)

Set the layer add type.

<param name="LayerName">the relevant layer name</param>

<param name="AddType">AddType of layer</param>

public void SetMatrixLayerAddType(string LayerName, MatrixLayerAddType AddType)

Get the matrix layer color for the relevant layer.

<param name="LayerName">relevant layer name</param>

<returns>Color of the layer</returns>

public System.Drawing.Color GetMatrixLayerColor(string LayerName)

Set the matrix layer color.

<param name="LayerName">name of relevant layer</param>

<param name="Color">color which should be added to the matrix for this layer</param>

public void SetMatrixLayerColor(string LayerName, System.Drawing.Color Color)

Set context for one layer.

<param name="LayerName">the relevant layer</param>

<param name="Context">new value of context</param>

public void SetMatrixLayerContext(string LayerName, MatrixLayerContext Context)

Set polarity for one layer.

<param name="LayerName">the relevant layer</param>

<param name="Polarity">new value of Polarity</param>

public void SetMatrixLayerPolarity(string LayerName, MatrixLayerPolarity Polarity)

Set type for one layer.

<param name="LayerName">the relevant layer</param>

<param name="LayerType">new value of type</param>

public void SetMatrixLayerType(string LayerName, MatrixLayerType LayerType)

Set the drill start for one drill layer.

<param name="DrillLayerName">The relevant Drill layer</param>

<param name="StartLayer">Name of start layer or empty string for all layers.</param>

public void SetDrillStartLayerFor(string DrillLayerName, string StartLayer)

Set drill end layer.

<param name="DrillLayerName">relevant drill layer name</param>

<param name="EndLayer">Name of the end layer or empty string for all layers</param>

public void SetDrillEndLayerFor(string DrillLayerName, string EndLayer)

Remove step information from matrix (do not delete files on HDD).

<param name="StepName">Name of the step to remove.</param>

public void RemoveStep(string StepName)

New Order for Matrix by names beginning with first index.

<param name="LayernamesInCorrectOrder"></param>

public void SetMatrixOrder(List<string> LayernamesInCorrectOrder)

IObject

Creates outline as IPolyClass.

<returns>Polygon of Object.</returns>

public virtual PCBI.MathUtils.IPolyClass GetPolygonOutline()

IArcSpecifics

Width of the pen (thickness of arc).

public double PenWidth;

ITextSpecifics

For example "1-1-1-1".

public string Occurrence;

Factor to stretch the text.

public float WidthFactor;

IODBLayer

Create polygonized objects for all layerobjects

<param name="flattenValue">0 if you don't whant to fallten, else bigger value.</param>

<param name="OnlySelected">If you only need some features, select them and set OnlySelected</param>

<returns>list of objects, moste are Surfaces. Some others are possible (only connected elments are replaced).</returns>

public List<IODBObject> GetPolygonizeLayerObjects(double flattenValue, bool OnlySelected)

IODBObject

Calculate distance between this object and an point.

<param name="Point">The relevant point</param>

<returns>distance between point and object</returns>

public double DistanceTo(MathUtils.PointD Point)

Returns a Dictionary with FeatureAttributeEnum key and value as string.

<returns>List of attributes as dictionary.</returns>

public Dictionary<FeatureAttributeEnum, string> GetAttributesDictionary()

Set the specific values of the object, uses the last ShapeIndex!

<param name="Specifics">The new specifics for the object</param>

public override void SetSpecifics(IObjectSpecificsD Specifics)

Is this element positive?

public bool Positive

Creates outline as IPolyClass.

<returns>Polygon of Object.</returns>

public override PCBI.MathUtils.IPolyClass GetPolygonOutline()

IPCBIWindow

Get or Set the real job path for imports in different formats e.g. IPC2581 or GenCad 1.4.

public string OriginalJobPath

Get the name of the job, if it is loaded from zip/tgz the real name is given back not the temporary unpacked name.

<returns>Jobname</returns>

public string GetJobName()

Set name of job for misc file.

<param name="JobName">Jobname</param>

public void SetJobName(string JobName)

Result of LoadData to handle how to go on.

public enum LoadInformation

```
{
            Unknown,
            Successful,
            PathUnknown,
            PlugInNotFound,
            Error
}
```

Load data from all available formats (depending on your license).

<param name="FullPath">Path of file or directory of data structure (ODB++)</param>

<param name="Information">Returnvalue of success or error (check errorlog for details).</param>

<returns>True if data is loaded.</returns>

public bool LoadData(string FullPath, out LoadInformation Information)

Save Job with all opened layers, attributes, nets and packages.

public void SaveJobImperative()

Calculate world to client coordinates.

<param name="WorldPoint">the world coordinate point</param>

<returns>the client coordinate point</returns>

public Point WorldToClient(PCBI.MathUtils.PointD WorldPoint)

ZoomRect set the view in PCB-Investigator to the rectangle (if the rectangle side ratio does not apply to the window size PCB-Investigator make the view bigger).

<param name="Rect">chosen view rectangle</param>

public void ZoomRect(PCBI.MathUtils.RectangleD Rect)

Adds an PicuteLayer to the current ODB++ Job.

<param name="FilePath">location of the image</param>

<param name="LayerName">the new layer name</param>

<param name="FitIn">try to fit the image in profile</param>

<param name="OpenPictureLayerDialog">Would you transform the image?</param>

public IPictureLayer AddPictureLayer(string FilePath, string LayerName, bool OpenPictureLayerDialog)

Occurs if user change view e.g. from 2D to 3D view.

public event EventHandler PCBIMainViewChanged;

Show notes dialog for edit and switch between notes.

This does not close the open dialog!

public void ShowNotesDialog()

Show notes dialog for edit and switch between notes and initialize a new note.

This does not close the open dialog!

<param name="NoteText">Text for note.</param>

<param name="RelevantLayerName">The layer on which the note should be added.</param>

public void ShowNotesDialog(string NoteText, string RelevantLayerName)

Update layer list, step control and zoom home.

Important after working in other threads and add extra controls like tab pages or menus.

public void UpdateControlsAndResetView()

Check for the type, if the type is loaded and a instance is created the plugin will be returned.

<param name="t">type of the plugin</param>

<returns>the plugin or null</returns>

public Plugin.Interfaces.IPlugin GetPlugInInstance(Type t)

Send a message to other plugin with information what to do

<param name="ReceiverName">The fullName of plugin.Type</param>

<param name="SenderName">the sender name</param>

<param name="MessageID">What to do? 1=add items, 2 = save,...</param>

<param name="Params">The relevant parameters</param>

<returns>true if it works</returns>

public bool SendMessage(string ReceiverName, string SenderName, int MessageID, List<object> Params)

Create a list of all loaded plugins with name of plugins as keys.

<returns>All loaded plugins.</returns>

public Dictionary<string, Plugin.Interfaces.IPlugin> GetAllLoadedPlugIns()

Preview image is created from outline with drills. Simple without special pads and only one drill layer.

The image will be created of the loaded job, if no job is loaded the return bitmap is black.

It will use the current step for pcb outline and drills.

<param name="ImageSize">Size of the result image for preview.</param>

<returns>Image for preview.</returns>

public Bitmap CreatePreviewImage(Size ImageSize)

In options you can add colors for layers in there activation order, here you can use the index to set color of activation.

<param name="IndexOfActivation">order of layer activation</param>

<param name="ColorOfLayer">color of layer</param>

public void SetSpecialColorForLayer(int IndexOfActivation, Color ColorOfLayer)

Show pin information on components or not.

public bool ShowPinNumbers

Limit count of active layers in PCB-Investigator.

public int MaxCountActiveLayers

IPin

Create a list of lines and arc who build the pin outline.

<param name="ParentCMP">The component which contains the pin</param>

<returns>outline of the pin contour</returns>

public List<IObjectSpecifics> GetOutline(ICMPObject ParentCMP)

Creates outline as IPolyClass.

<param name="ParentCMP">The component which contains the pin</param>

<returns>Polygon of Object.</returns>

public PCBI.MathUtils.IPolyClass GetPolygonOutline(ICMPObject ParentCMP)

Pin Type Ellipse, Rectangle or Polygon.

If not possible to get the type it is unknown.

public PinType Type

Types of pins and unknown if not possible to get the type.

```
public enum PinType
{
        Unknown,
        PinEllipse,
        PinRectangle,
        PinPolygon
}
```

IPolygon

Add complete polygon to current polygon.

<param name="SecondPolygon">Other Polygon to add.</param>

public void AddPolygon(IPolyClass SecondPolygon)

Check all edges and calculate bound new.

public void UpdateBounds()

Rotate the polygon.

<param name="Angle">Angle in degrees.</param>

public void Rotate(double Angle)

Clone the object to get a second polygon with same values.

<returns>A identically copy.</returns>

public IPolyClass Clone()

Calculate distance from this polygon to second polygon.

<param name="SecondPoly">Polygon to calculate distance.</param>

<param name="From">Point from this polygon.</param>

<param name="To">Point to second polygon.</param>

<returns>Distance in mils.</returns>

public double DistanceTo(IPolyClass SecondPoly, ref PointD From, ref PointD To)

IArcSpecificsD

Width of the pen (thickness of arc).

public double PenWidth;

ITextSpecificsD

For example "1-1-1-1".

public string Occurrence;

Factor to stretch the text.

public double WidthFactor;

IStep

Gets a bitmap for the chosen rectangle area, expands the chosen area if it is too small.

<param name="Layers">All layer to draw.</param>

<param name="DetailRectangle">The choosen area in mils.</param>

<param name="Width">The destination width.</param>

<param name="Height">The destination height.</param>

<param name="DrawPCBOutline">Outline of PCB contour in the image</param>

<param name="BoardColor">Color to fille the PCB contour.</param>

<param name="ShowComponentDetails">Show the Component Info from Component View setup.</param>

<returns>Returns a picture form the coosed layers.</returns>

public Bitmap GetBitmap(List<ILayer> Layers, RectangleF DetailRectangle, int Width, int Height, bool DrawPCBOutline, Color BoardColor, bool ShowComponentDetails)

Create image from clipping rectangle, the background is white for this image.

<param name="ClippingRect">The relevant rectangle.</param>

<param name="PictureWidth">Width of image.</param>

<param name="PictureHeight">Height of image.</param>

<returns>A Image of all active layers with visible area of clipping rectangle.</returns>

public Bitmap DrawColoredImage(RectangleF ClippingRect, int PictureWidth, int PictureHeight)

Reads Gerber or Excellon data, errors are reported in the ErrorLog.

<param name="FullPath">The filename with full path.</param>

<param name="Type">The type of file.</param>

<param name="LeadingNumbers">Leading number count e.g. 3 or 0 if PCB-Investigator should check for it.</param>

<param name="TrailingNumbers">Trailing number count e.g. 5 or 0 if PCB-Investigator should check for it.</param>

<param name="unitInch">inch or mm or unknown</param>

<param name="OverwriteIfExists">If true overriedes existing layer, if false change the name of the layer.</param>

<param name="TryAutoRecognition">For Excellon1/2 the size of file will be adjusted on the other layers of this step.</param>

<returns>Returns name of the layer or an empty if an error occurred.</returns>

public string AddGerberLayer(string FullPath, bool OverwriteIfExists, FormatTypes Type, int LeadingNumbers = 0, int TrailingNumbers = 0, bool? unitInch = null, bool TryAutoRecognition = true)

All layers will be turned off.

<param name="UnloadLayers">Unload all layers to save RAM.</param>

public void TurnOffAllLayer(bool UnloadLayers)

The step will be flatted, all repeated objects from other steps are placed on the new step.

<param name="newStepName">This must be a new stepname!</param>

<param name="saveRAM">Save RAM takes much more time but needs less RAM.</param>

<returns>false if it is falled</returns>

public bool FlattenStep(string newStepName, bool saveRAM)

Step attributes as strings.

<returns>List of all included step attributes.</returns>

public List<string> GetStepAttributes()

Add attribute to current step.

<param name="attribute">name of attribute starts with "."</param>

<param name="value">value of relevant attribute</param>

public void AddStepAttribute(string attribute, string value)

Load and get back layer attributes.

<param name="LayerName">Relevant layer name for the attributes.</param>

<returns>dictionary of attributes and values of the layer</returns>

public Dictionary<string, string> GetLayerAttributes(string LayerName)

Search for attribute of the relevant layer and return it or give an empty string if it dosn't contain the attribute name.

<param name="LayerName">relevant layer</param>

<param name="AttributeName">attribute name of relevant attribut (e.g. .comment)</param>

<returns>value or empty string</returns>

public string GetLayerAttribute(string LayerName, string AttributeName)

Set an layer attribute value, in ODB++ all attributes start with "." and use lower cases.

<param name="LayerName">Relevant layer name. </param>

<param name="Attribute">The name of odb attribtue starts with .(point) and use lower cases.</param>

<param name="Value">value of odb attribte</param>

public void SetLayerAttribute(string LayerName, string Attribute, string Value)

Go throug all relevant layers and round each element on digits depending on the repair value.

<param name="repairValue">Base calculation value for digit of the repaired values.</param>

<param name="RelevantLayers">All layernames who are of interest.</param>

public void RepairElementsForAOI(double repairValue,List<string> RelevantLayers)

PCBIColors

Return the color of PCB-Investigator layer list background and matrix.

<param name="MType">The type of relevant color</param>

<param name="MContext">The content of relevant color</param>

<returns>Color of internal using</returns>

public static Color GetLayerTypeColor(MatrixLayerType MType, MatrixLayerContext MContext)

IPCBIStyleLibrary

many new icons and Bitmaps

PlugInForm (new)

helper for plugins to comunicate with PCB-I