CIFSYM -- CIF Symbol Handler

by

Stephen Trimberger

Technical Report 3253

December 1979

Computer Science

California Institute of Technology

Pasadena, California 91125

Silicon Structures Project

Sponsored by

Burroughs Corporation, Digital Equipment Corporation,

Hewlett-Packard Company, Honeywell Incorporated,

International Business Machines Corporation,

Intel Corporation, Xerox Corporation,

and the National Science Foundation

The FORTRAN CIF2.0 Symbol Handler

The symbol handler is a module that uses the FORTRAN CIF parser (SSP memo #5350) and intercepts all calls from that parser. CIFSYM handles all CIF symbols, definition deletion and calls, and generates calls to its client program's subroutines to output the four graphic primitives supported by CIF2.0, boxes, wires, round flashes, and polygons. For a description of CIF2.0, see SSP file #2686, "The Caltech Intermediate Form for LSI Layout Description."

CIFSYM handles several possibly large semi-random access disk files for intermediate storage. Since a whole symbol need not fit into main memory, very large symbols can be easily handled by CIFSYM.

To use CIFSYM, you must load PARSER and CIFSYM with your CLIENT:

```
@load PARSER, CIFSYM, CLIENT
Link: Loading
EXIT
@save CLIENT
```

CIFSYM Entry Points

Processing of a CIF file is invoked by calling the subroutine:

CALL CIFSYM(doits, cifnam, dskin, errout)

CIFSYM is the entry point for parsing CIF files. Doits is an array of eight booleans. The booleans tell CIFSYM whether or not to process that layer. Cifnam is an array of eight CIF layer names. The layer names of those which have a true "doit" are the only ones that must be defined. CIFSYM will only recognize the layers listed in Cifnam, all others will be flagged as errors. Dskin is the FORTRAN unit number for the input file, errout is the unit number for the error messages file.

CIFSYM proceeds until a CIF graphic command is reached, then the subroutine for that graphic command is called. When the user's subroutine for a given graphic command ends, control automatically passes back to CIFSYM and parsing continues from that spot. CIFSYM returns to the client when the CIF file is finished.

CIFSYM provides four procedures for the client to use to put data into CIFSYM's representation of a symbol. These procedures may only be called when CIFSYM has a symbol open for definition, which the client can easily determine because when CIFSYM has a symbol open for definition, it calls the client's subroutines SYMxxx with the data. See below and see the file CHUCK.FOR for an example of CIFSYM use. All numbers in the procedures must be integers. Positions must be in CIF units of 1/100 micron, angles are in units of 1/100 degree. Npts is the number of points contained in the array pts. Pts is an array dimensioned at least 2*npts, containing x-y pairs of points in consecutive locations. Layers correspond to the positions of the CIF layers given in Cifnam in the call to CIFSYM.

```
PUTBOX(xsize, ysize, angle, xcen, ycen, layer)
PUTWIR(width, npts, pts, layer)
PUTROU(diam, xcen, ycen, layer)
PUTPOL(npts, pts, layer)
```

Common Blocks

Because FORTRAN has no scoping, the client program is free to use these common blocks. However, clients are warned to not change any of the data because that might cause the CIF processing to fail. These are the common blocks in CIFSYM. PARSER has its own common blocks and the reader is directed to #3350 for a description of them.

```
      common/zconst/ rndpad, box, poly, wire, eof
      common/zcntrl/ debug, doit(8)
      common/zcustf/curcel, curt11, curt12, curt21,
     *      curt22, curtx, curty, curunt, currec,
     *      curwrd, curlay, curlin
      common/zerdat/erary(205)
      common/zlystf/cifnam(8), units(8), recnum(8),
     *      recpos(8), recbuf(510,8), oldlay, insym,
     *      nlayer, reclen
      common/zscstf/ num, den
      common/zstack/celstk(50), t11stk(50), t12stk(50),
     *      t21stk(50), t22stk(50), txstk(50), tystk(50),
     *      recstk(50), wrdstk(50),stktop, stkmax
      common/zsystf/symtbl(300), deftbl(300), instbl(300),
     *      filtbl(300,8),symlen, symmax
```

/zconst/ contains the constants used to flag different entries in CIFSYM's intermediate files.

```
integer rndpad, box, poly, wire, eof
```

/zcntrl/ contains information for controlling data movement. If debug is true, CIFSYM debugging is on and intermediate files will be saved after execution. Doit is the array of booleans which tells whether or not to process the given CIF layer.

```
boolean debug, doit(8)
```

/zcustf/ contains the data on the current symbol(s) being expanded, if any. Curcel is the CIFSYM internal symbol number (0 if no symbol being expanded), curunt is the current input unit for data from the symbol temp file. Currec and curwrd are the current record and position in that record being read and curlay is the current layer being read. Curlin is the source line on which the current graphic element was found. The KLALs are the six numbers in the transformation matrix from the cell's coordinates to output coordinates.

```
integer curcel, curunt, currec, curwrd, curlay,
curlin

real curt11, curt12, curt21, curt22, curtx, curty
```

/zerdat/ contains the current data item input. It is used sometimes when an error occurs for additional debugging data.

```
        integer erary(205)
```

/zlystf/ contains data on each layer. Cifnam is the name of the layer, units is the unit number on which the data for that layer is kept, recnum and recpos are the current positions for writing data to those files. Recbuf is the buffer for writing data to the files. Oldlay is the old layer saved over a symbol call. Nlayer is the number of layers (8) and reclen is the length of a record buffer (510). Insym is a boolean telling if the current incoming CIF data is in a symbol.

```
        integer cifnam(8), units(8), recnum(8), recpos(8)

        integer rechuf(510,8), oldlay, nlayer, reclen

        boolean insym
```

/zscstf/ contains the two numbers on a CIF symbol for scaling the numbers inside.

```
        integer num, den
```

/zststf/ contains the information for the symbol call stack. Celstk is the stack of cell names. Recstk and wrdstk are the stacks for the saved positions of reading symbol data. Stktop is the current top-of-stack position. Stkmax is the maximum stack depth (50). The ktAls are the stack of transformation matrices to get to the current symbol.

```
        integer celstk(50), recstk(50), wrdstk(50),stktop,
stkmax
```

```
real t11stk(50), t12stk(50), t21stk(50)

real t22stk(50), txstk(50), tystk(50)
```

/zsystf/ contains the symbol table information. Symtbl is the table of CIF numbers. Filtbl is the table of starting records for the data for each layer of the symbol (each symbol starts at the beginning of a record in the file). Symlen is the largest table slot used so far. Symmax is the symbol table size (300). Deftbl is the table of booleans which tells whether or not the current symbol has been defined. Instbl is the table which tells whether or not the current symbol is being instantiated. Both booleans are used primarily for error checking.

```
integer symtbl(300), filtbl(300,8), symlen, symmax

boolean deftbl(300), instbl(300)
```

The only entry of any interest to a client of CIFSYM might be /zcustf/ curcel, which tells if the current data are coming from a symbol or are they first-time data. This would be useful if the client modifies data (such as application of bloating) as it sees it in symbol defs. The client would not want to bloat the data the second time when it is read out of the symbol.

Graphic Routines

CIFSYM calls eight subroutines which must be supplied by the client. The first four subroutines are used when a

graphic element is encountered in a symbol definition. The numbers passed to the SYMxxx subroutines are the numbers in the symbol coordinate space. All numbers are integers. Positions are in the CIF units of 1/100 micron, angles are 1/100 degree. Layer numbers correspond to the layers that were passed into CIFSYM when it was called. Pts is the number of points in the path. At this point, duplicate points have been removed.

The user may carry out any operation desired on the data, then call the PUTxxx subroutines defined in CIFSYM to add data to the saved representation of the symbol. This is helpful when the client wishes to fracture wires and polygons into rectangles, for example, for pattern generator conversion. That can be done and the resulting rectangles put into the symbol with repeated calls to PUTBOX.

```
SYMBOX(xsize, ysize, angle, xcen, ycen, layer)
SYMWIR(width, npts, pts, layer)
SYMROU(diam, xcen, ycen, layer)
SYMPOL(npts, pts, layer)
```

The DOxxx subroutines are called when there is data to be plotted. They are called when data is found outside any symbols in the CIF code and also when a symbol call is being expanded. The graphic features described are given in absolute coordinates. All transformations have been applied to the data. The parameters are as described in the paragraph above.

```
DOBOX(xsize, ysize, angle, xcen, ycen, layer)
DOWIR(width, npts, pts, layer)
```

DOROU(diam, xcen, ycen, layer)
DOPOL(npts, pts, layer)

CIFSYM Internal Names

CIFSYM has a number of internal subroutines and common blocks which will interfere with user's names for of subroutines and common blocks. For that reason, ALL CIFSYM INTERNAL PROCEDURES AND COMMON BLOCKS START WITH THE LETTER 'Z' except those used for interfacing to PARSER. The user is directed to memo #3350 for a list of the parser internal names and parser external names, ALL OF WHICH ARE USED BY CIFSYM. The client program must not attempt to use any of these names.

Subroutines

ZBEFOR	ZAFTER	ZFNDSY
ZLYINI	ZLYFIN	ZINST
ZPUSH	ZPOP	ZSETIN
ZSETMI	ZMMULT	ZTRREC
ZTRROU	ZTRPOL	ZTRWIR
ZTRPT	ZSCALE	ZINNUM
ZPUTI	ZFLUSH	ZERR2

Common Blocks

ZCONST	ZCNTRL	ZCUSTF
ZERDAT	ZLYSTF	ZSCSTF
ZSTACK	ZSYSTF	

Error Messages

Error messages are preceded by the text of the current line of the CIF file that is being processed and the current line number and character position on that line. The position on the line should point to the end of the command. Many of the messages are followed by additional information for debugging such as the current input item from a symbol file or the current symbol call stack.

Warning -- Symbol previously defined. Re-defining symbol.

The symbol currently being started was previously defined. The symbol is re defined, which is equivalent to deleting the old symbol definition at that spot. Any use of the symbol previous to this point will use the old definition. Any use after this point will use the new definition.

Error -- Layer specification not recognized. Layer Unchanged.

CIFSYM only recognizes the layers passed to it. For a list of the NMOS layer specifications, see Display File #2686, "The Caltech Intermediate Form for LSI Layout Description". All other layer designations are flagged by this message.

Warning -- Unrecognized user command.

At present, no user commands are defined, so all are flagged with this message. There are some proposals for user commands which will be gathered together in an upcoming Display File.

Error -- Too many symbols defined. Implementation Limitation. Dying.

CIFSYM is currently limited to 300 symbols total. In order to increase this number, the size of the arrays in common block /zsystf/ in file CIFSYM.FOR must be increased, and the initialization of variable SYMMAX in subroutine ZBEFOR in CIFSYM must be changed to the new table size.

Error -- Embedded symbol definition. Parser Error. Terminating current symbol.

The parser passed an embedded symbol definition. There is something wrong with the parser. CIFSYM assumes that the user wanted to terminate the current symbol before starting the new one.

Error -- No start for symbol being finished. Parser Error. Command skipped.

The parser passed a Def Finish command without a Def Start command. There is something wrong with the parser. CIFSYM ignores the command.

Error -- No layer specified before graphic command. Command Skipped.

The user must specify a layer before specifying any graphics. If there is no layer specified, the command is skipped.

Error -- Recursive symbol call. Call was skipped.

A symbol tried to call itself either directly or through another symbol call perhaps many levels deep. If the recursive call was done, an infinite amount of data would be generated. The call was not carried out.

Error -- Bad data type in ZINST. Program Error. Execution Terminated.

Error -- Call stack underflow. Program Error. Execution Terminated.

Error -- SKIINP on bottom-level cell. Program Error. Execution Terminated.

These three errors arise from impossible conditions in CIFSYM. They should be reported as bugs.

Error -- call stack overflow. Implementation Limitation. Execution Terminated.

Currently, CIFSYM allows Calls to nest no more than 50 deep. To increase this, the size of the arrays in common block /zstack/ must be increased and variable STKMAX should be changed to the new value in subroutine ZBEFOK in file CIFSYM.FOK.

Error -- Symbol in Call not defined. Call was skipped.

A call was made to a symbol that was not defined in the CIF file. The call, of course, could not be carried out.

Error -- Symbol numbers must be >= 0. Command skipped.

An attempt was made to define a symbol with an illegal symbol number. The symbol definition could not be carried out.

Error -- No symbol open on PUTxxx. Graphics skipped.

The client may only call a PUTxxx routine when a symbol is open for definition. CIFSYM will ignore the calls otherwise.

Warning -- Duplicate points found. Extra one deleted.

A wire or polygon was found with doubled points. The extra point is deleted before the wire is sent to the client program.

Error -- A or B is zero in symbol def. Set to 1.

One of the parameters on a Define Symbol command was zero. If A=0, then all the data in the symbol will be reduced to a singularity. If B=0, there will be an attempted division by zero. Neither could be correct.

Wildcard Error. xxxxx...Can't help you further.

This should never happen. If it does, write down xxxxx, the number of the error and report a program bug.