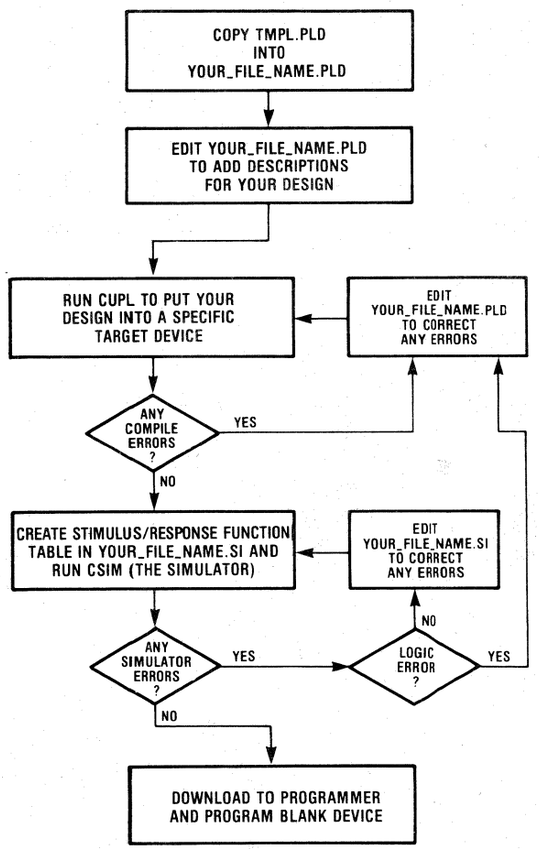
CUPL™ (2.0)

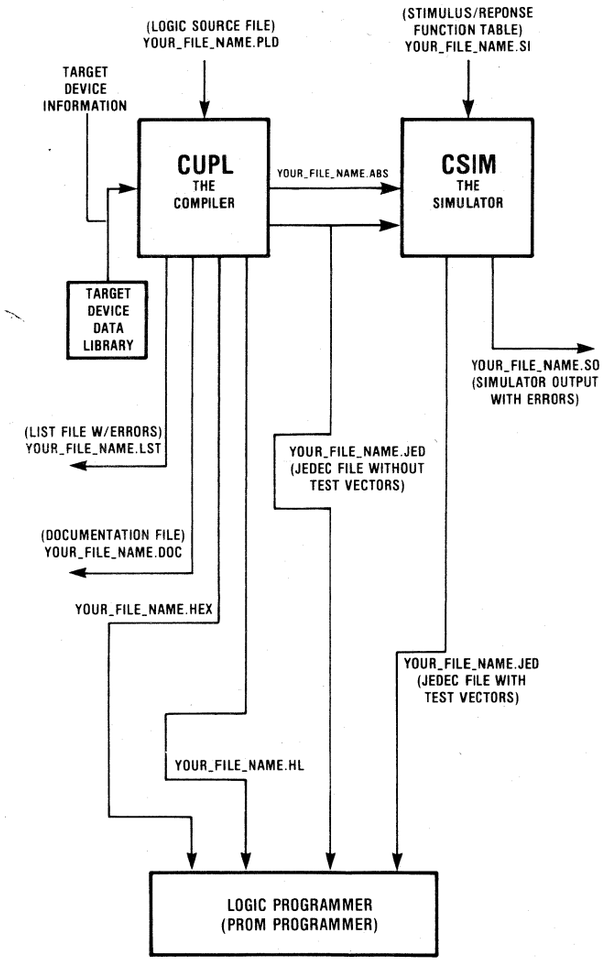
The Universal Compiler For Programmable Logic

QUICK REFERENCE GUIDE



CUPL and CSIM

INPUT and OUTPUT FILES



THE TEMPLATE FILE

Supplied with the **CUPL** files is a file named **TMPL.PLD.** This is a blank template file which you should use to build your logic source file. Its structure is as follows:

```
PARTNO      <FOR_THIS_FUNCTION>;
NAME        <YOUR_FILE_NAME>;
DATE        <DATE_OF_LAST_CHANGE>;
REV         <CURRENT_REV_NO>;
DESIGNER    <YOUR_NAME>;
COMPANY     <YOUR COMPANY>;
ASSEMBLY    <WHERE_PLD_USED>;
LOCATION    <ON_PC_BOARD>;

/*****************************/
/*      <FILL IN TITLE BLOCK WITH>      */
/* <EXPLANATION OF LOGIC FUNCTION> */
/*****************************/

/*  ALLOWABLE TARGET DEVICE TYPES:  */
/*       <LIST MENU OF DEVICES>       */

/** INPUTS **/
  PIN <NUMBER> = <NAME> ;
                        /**<DESCRIPTION>**/

/** OUTPUTS **/
  PIN <NUMBER> = <NAME> ;
                        /**<DESCRIPTION>**/

/** DECLARATIONS AND INTERMEDIATE
                    VARIABLE DEFINITIONS **/
  <INTERMEDIATE_VARIABLE = EXPRESSION>;
  <FIELD BIT_FIELD_VAR = [VARIABLE LIST]>;

/** LOGIC EQUATIONS **/
  <OUTPUT = EXPRESSION>;
```


RUNNING CUPL

In order to compile a specific logic source file **(YOUR_FILE_NAME.PLD)** for a specific target device, type the following:

CUPL [FLAGS] TARGET_DEVICE_CODE YOUR_FILE_NAME

For example:

CUPL -J -A P16L8 RAMCNTRL

which would compile the logic source file **RAMCNTRL.PLD** for a generic PAL16L8 Target Device, while producing a JEDEC File (YOUR_FILE_NAME.JED) and an absolute format file (YOUR_FILE_NAME.ABS) which must be present if the simulator (CSIM) is to be run later.

This list of Target Device types and their corresponding device codes can be found in an Appendix of the manual.

A list of **CUPL** option flags is presented below:

- **— J →** Produce YOUR_FILE_NAME.JED, the JEDEC format downloadable file.
- **— A →** Produce YOUR_FILE_NAME.ABS for later use by CSIM.
- **— S →** Automatically run CSIM after running CUPL.
- **— L →** Produce YOUR_FILE_NAME.LST with line numbers and error messages.
- **— I →** Produce YOUR_FILE_NAME.HL downloadable HL format file for IFL..
- **— H →** Produce YOUR_FILE_NAME.HEX ''ASCII Hex Space'' format file.
- **— F →** Produce YOUR_FILE_NAME.DOC with fuse map file.
- **— X →** Produce YOUR_FILE_NAME.DOC with fully expanded product-terms.
- **— G →** Program security fuse.
- **— R →** Disable global product-term merging. (FPLA devices).
- **— M0 →** Perform no logic minimization.
- **— M1 →** Perform local logic minimization (default).
- **— M2 →** Perform logic minimization until equations fit in target device.
- **— M3 →** Perform full logic minimization.
- **— D →** Deactivate unused OR-terms (Increases speed in FLPAs).
- **— U →** Use specified library for compilation.

*PAL is a Registered Trademark of Monolithic Memories, Inc.

CUPL SYNTAX

- **LOGICAL OPERATORS**

  & = LOGICAL AND
  # = LOGICAL OR
  $ = LOGICAL XOR
  ! = LOGICAL NEGATION

- **FREE FORM COMMENT STRUCTURE:**

  /* = START COMMENT
  */ = END COMMENT

- **VARIABLE EXTENSIONS**

```
VAR.D  = EXP; /* D of D FLIP-FLOP */
VAR.J  = EXP; /* J of J-K FLIP-FLOP */
VAR.K  = EXP; /* K of J-K FLIP-FLOP */
VAR.R  = EXP; /*R of R-S FLIP-FLOP*/
VAR.S  = EXP; /*S of R-S FLIP-FLOP*/
VAR.AR = EXP; /*ASYNCHRONOUS FF RESET*/
VAR.AP = EXP; /*ASYNCHRONOUS FF PRESET*/
VAR.SR = EXP; /*SYNCHRONOUS FF RESET*/
VAR.SP = EXP; /*SYNCHRONOUS FF PRESET*/
VAR.OE = EXP; /* THREE-STATE ENABLE */
```

- **NODE DECLARATIONS:**

```
NODE VARIABLE_NAME; /*SINGLE NODE, AS WHEN*/
                    /*USED FOR ''COMPLEMENT_ARRAY''*/
                                  /*IN IFL DEVICES.*/
NODE [VARIABLE_LIST];
                          /*MULTIPLE NODES, AS WHEN*/
                            /*USED FOR BURIED STATE*/
                                          /*BITS.*/
```

- **THE DISTRIBUTIVE PROPERTY:** (From BOOLEAN Algebra)

A & (B # C) is replaced by A&B # A&C
(where & operations are performed before # operations)

deMorgan's Theorem: (From BOOLEAN Algebra)
!(A#B) is replaced by !A & !B
also
!(A&B) is replaced by !A # !B

NOTE: This symbology tends to create large numbers of Product-Terms.

CUPL SYNTAX

- **MACRO SUBSTITUTION:**

A symbolic name may be arbitrarily created and defined as in:

MEMREQ = MEMW # MEMR;

where ''MEMREQ'' does not appear as a Pin variable name.

''MEMREQ'' may then be used in expressions for other variables. Whenever ''MEMREQ'' is used, the value ''MEMW # MEMR'' will be substituted.

- **THE LIST NOTATION:**

Groups of variables may be represented in a shorthand list notation in any of the following formats:

[A,B,C] as in [MEMR, MEMW, IOR, IOW]

or [X N ..Z] as in [ADR7..0]
which replaces
[ADR7, ADR6, ADR5, ADR4, ADR3, ADR2, ADR1, ADR0]

- **BIT FIELDS**

A group of bits may be declared to be equal to a single symbolic name as in:

FIELD IOADR = [A7..0];

where afterward, ''IOADR'' may be used in expressions.

- **EQUALITY AND ADDRESS RANGE:**

The '':'' operator compares a bit field with a hex constant value or list of constant values as in:

IOADR:C3
or
IOADR: [10..3F]

which will be true for addresses in the range of 10 hex through 3F hex inclusively.

NOTE: Hex constant values must contain the proper number of nibbles to include the most significant bit of the Bit-Field variable list.

Also, the '':''operator may also be used to operate on a Bit-Field variable list, as in:

IOADR:& — A7 & A6 & A5 & A4 & A3 & A2 & A1 & A0
IOADR:# — A7 # A6 # A5 # A4 # A3 # A2 # A1 # A0

PALASM-TO-CUPL LANGUAGE TRANSLATOR

This program will convert logic source files in the **PALASM*** Format to the **CUPL** source file format.

To convert one or more **PALASM** source files to **CUPL** format type:

PTOC FILE_NAME1 FILE_NAME2 ...(RET)

For example:

PTOC BUS_CNTL.ASM (RET)

which would produce the following **CUPL** format files:

BUS_CNTL.PLD.

and, if the original PALASM file had the ''Function Table'' information,

BUS_CNTL.SI

THE CUPL PREPROCESSOR

This program operates on the **CUPL** source file before compiler operations actually begin. Capabilities include:

String Subsystem:

$DEFINE ARG1 ARG2

where ARG1 is replaced with ARG2 until $UNDEF ARG1 is encountered.

File Inclusion:

$INCLUDE FILENAME

where the referenced file becomes part of the specification at Compile time.

Conditional Compilation:

Portions of the source specification may be compiled or not depending on whether or not the Argument (ARG) has been defined using the **$DEFINE** command. The formats are:

```
$IFDEF  ARG
...STATEMENTS...
   $ELSE
...STATEMENTS...
   $ENDIF
```

or, if not defined:

```
$IFNDEF  ARG
...STATEMENTS...
   $ELSE
...STATEMENTS...
   $ENDIF
```

*PALASM is a Registered Trademark of Monolithic Memories, Inc.

CSIM: THE SIMULATOR

CSIM is a Stimulus/Response Function Table oriented simulator which will compare each expected response with that which the logic in the associated .PLD file would produce given the specified stimulus.

The simulator input file (YOUR_FILE_NAME.SI) must contain the same header information as the associated logic source file (YOUR_FILE_NAME.PLD). Also, **CUPL** must have been previously run for the .PLD file with the —A option flag to produce an absolute file (YOUR_FILE_NAME.ABS), and also with the —J flag if you would like CSIM to append the function table Test-Vector information to your .JED file in order to produce a .JED file with both fuse and testing information.

The general format for the .SI file is:

''HEADER INFORMATION''

ORDER:

VAR1, VAR2, ..., VARN ;

VECTORS:

STIMULUS PATTERN 1	RESPONSE PATTERN 1
STIMULUS PATTERN 2	RESPONSE PATTERN 2
•	•
•	•
•	•
STIMULUS PATTERN N	RESPONSE PATTERN N

Within the Vector Table, inputs are defined with ''1'' (+5V), ''0'' (GND) while outputs are defined with ''H'' (+5V), ''L'' (GND), and ''Z'' (High Impedance). Don't cares are represented by X. A ''*'' in the response field causes the simulator to determine the output according to the logic definition contained in the .ABS file which was derived from your .PLD file.

- **DIRECTIVES**

(Place on any row of vector table)

```
$MSG ''YOUR_MESSAGE'' ;
$REPEAT n ;
$TRACE n ;
$EXIT ;
```

To run CSIM, type:

CSIM [FLAGS] TARGET_CODE YOUR_FILE_NAME

- **CSIM FLAGS**

—J → Produce a .JED file with test vectors.
—L → Produce a .SO file (simulator output).
—V → Display simulator output vectors.
—U → Use specified library for compilation.