Low-Power Design Environment for MOSIS

David Garrett, Joshua Garrett, Mircea Stan
garrett@virginia.edu, http://csissun13.ee.virginia.edu/~dcg3w/home.html
University of Virginia - Department of Electrical Engineering
Charlottesville, VA 29903

Abstract

The release of a CMOS low power library by the Infopad group at UC Berkeley gave engineers the ability to develop low power circuits with standard cells in Magic with minimum geometry layouts [1]. We describe a process that translates this low power standard cell library from Magic into the Mentor Graphics (MGC) environment. Using the Design Kit Example (DKE), we describe the steps to build a cohesive library structure that allows low power ASIC design; schematic capture with Design Architect (DA) or synthesis with Autologic II, digital simulation with Quicksim II, and standard cell layout in IC station.

1. Low-Power Design

A large part of power dissipation in CMOS circuits is due the dynamic switching of capacitive nodes. For a given node in a CMOS circuit, the power dissipation is proportional to the square of the supply voltage, the load capacitance, and the switching activity [2], shown in equation 1. Out of

$$P \propto \alpha C_L (V_{DD})^2 \qquad (1)$$

the three factors, the supply voltage and the load capacitances can be directly controlled at the library level. The rationale behind providing the Low-Power (LP) library as a complement to the MOSIS Design Kit (MDK) [3] is to have a standard cell library that has accurate timing information at lower voltage supply levels, and that maintains minimum capacitances per transistor. Lower supply voltages adversely affect the propagation delays of logic gates, and at voltages near the transistor threshold voltage, the increase in delay is quite large. When designing CMOS circuits at low voltages it is important to have accurate timing in order to verify that timing constraints are still met. Minimum capacitances are achieved by using minimum geometry transistors. This again comes at a cost in performance, and for larger fanouts, driver cells are required. By using minimum geometry layouts, and providing low voltage timing, the LP library will help reduce the power consumption for standard cell designs. Power reduction by lower switching activity will largely depend on circuit implements.

The LP library provides a reduction in capacitances to help with the power consumption, and also arms the designer with accurate timing information for low-voltage operation. With careful simulations to understand the timing impacts, standard cell designs can be fabricating in the MOSIS process with significant power savings. With the use of QuickPower, the LP library can become a useful tool not only designing, but understanding low-power designs in CMOS.

2. Library Structure

As a starting point of the library, all of the various design information need to be partitioned into a library structure. The LP library is based on both DKE [4] and the MDK libraries, and is structured as seen in Figure 1.

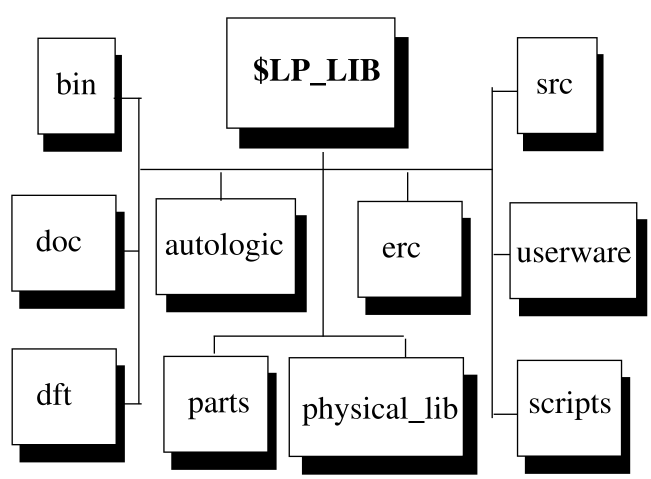

Figure 1: LP Library Directory

The major subdivision are explained as follows:

- **bin** - contains programs
- **doc** - documentation for the library
- **dft** - design for test models
- **autologic** - synthesis libraries
- **parts** - symbols, schematics, and timing
- **physical_lib** - cell layouts
- **src** - source code for timing and models
- **userware** - utilities for MGC environment

3. Symbol Generation

The first step in implementing the library is to generate symbols for all the logic cells in the library. The drawings for the symbols can be copied from the MGC generic library, or generated from scratch in DA. Figure 2 shows an example symbol for an inverter.

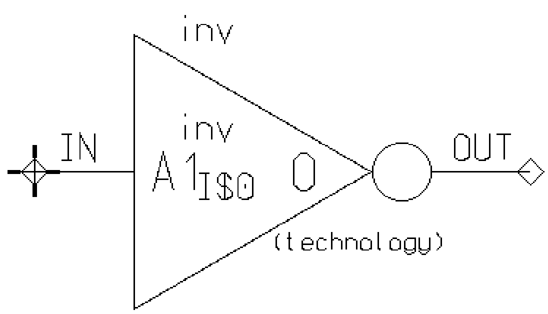

Figure 2: Inverter Symbol

Since many of the technology specific properties are added later in the process, the symbols must be stripped of all but the essential properties. Figure 3 shows the minimum assigned properties for an inverter as reported with *CIB.VIEW* utility from DKE.

4. Design Translation

Almost all of the remaining steps in translating the LP library to the MGC environment depend on having accurate layouts and extracted parameters.

4.1 Importing the Layouts

To make the layouts readable in ICStation, the standard cell layouts must be exported from Magic into a GDS II stream file with the command,

```
COMPONENT  inv  DEFAULT INTEFACE IS: inv

INTERFACE:  inv
   PINS:
              Compiled     User
        Id #   Pin Name    Pin Name  Properties
         1       A1             A1     (pin, A1)
                                       (pintype, IN)
         2       O              O      (pintype, OUT)
                                       (pin, O)
   BODY PROPERTIES:(model, (technology))
                   (inst, I$0)
                   (cell_name, inv)
   NTERFACE MODEL ENTRIES
   Model Entry   Type      Model Info
       0      mgc_symbolPath  : inv/inv
                   Labels: 'default_sym'
                   Status: Valid for inter-
                     face; Valid for
                     property
```

Figure 3: Symbol Properties

calma. These files are then read into IC Station with the GDS II to ICGraph conversion routine, and stored in the *$LP_LIB/physical_lib* directory. The SCMOS process from MDK needs to be linked to all of the cells at conversion time. Figure 4 shows the example inverter layout.

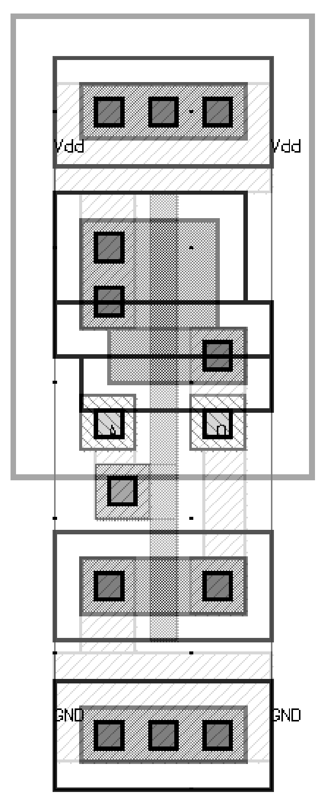

Figure 4: Inverter Layout

Several steps in addition to the conversion process are required to bind the layouts to the symbols for schematic-driven layout (SDL). First, all of the inputs, outputs, and power rails must be copied into the *METAL1.port* and *METAL2.port* layers. Using the *MAKE PORT* command, these layers are designated as ports to represented the input and output pads. The signal names must match exactly to the previously defined symbols. Also, a border around each standard cell shall be defined using the *FP1* layer. The *FP1* layer will completely encompass all of the metal routing within the layouts, but not completely cover the well definitions and select layers. This allows the cells to be placed side-by-side in rows and still share their wells with minimum separation.

Next, each of the physical layouts must be added to an IC Station *LIBRARY.* Using the built-in tools, the *METAL1* and *METAL2* layers are copied to the *METAL1_BLKG* and *METAL2_BLKG* layers and set to an external aspect. This allows the routing tools to avoid shorting metal layers in hierarchial cells. Figure 5 shows the ports, blockages and *FP1* layer in the inverter layout. The ports and the *FP1* layer are designated. The *BLKG* layers are represented by all the layers present in the figure.

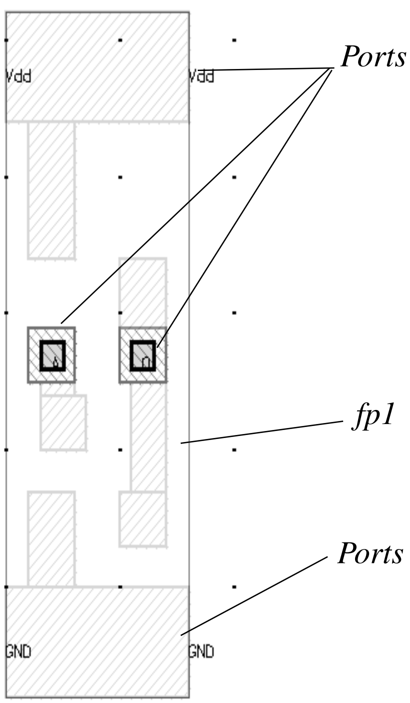

Figure 5: Ports and Blockages

Finally, each of the IC layouts must be set to a "standard cell" with a site type of [1] using the *$set_cell_type()* function. These property tells the automatic placement tools that all of the cells in the library have a standard height.

4.2. Model Extraction

As a starting point for timing development, HSPICE netlists can be extracted from the cell layouts. Because the LP library targets several technologies, a separated netlist must be extracted for each technology due to the differences in minimum feature sizes and parasitics. Initially, the LP library will target three technologies:

- Orbit 2.0um
- HP 1.2um
- HP 0.8um.

The distributed extraction function under the *EXTRACT(M)* menu in ICStation can be used to export an HPSICE netlist for each cell. The netlists are then saved in the *$LP_LIB/src/cellb/<tech>/netlists* directories, with each set of technology netlists in their respective (*<tech>*) directories. Figure 6 shows a netlist extracted from the inverter layout for the 2.0um Orbit process. Notice that the node names match the port definitions from the layout import step.

```
* File: scn20orbit_lib/netlists/invf101.sp.
      Creation time: Mon Aug 11 14:53:53 1997

.include scn20orbit_lib/netlists/invf101.sp.pex
.subckt invf101 A1 GND O VDD
* distributed parasitics:
x_1 A1 n1_1 n1_2 s_invf101_1
x_2 O n2_1 n2_2 s_invf101_2
x_4 VDD n4_1 n4_2 s_invf101_4
x_3 GND n3_1 n3_2 s_invf101_3
* devices:
m0 n2_1 n1_2 n4_2 n4_1 p l=2u w=12u
m1 n2_2 n1_1 n3_1 n3_2 n l=2u w=4u
.ends invf101
```

Figure 6: Inverter Extracted Netlists

5. Technology File Development

The timing models for the LP library must be based on simulations of various voltage, input slew, and output load vectors. These models can be developed for the LP library with the help of CellBuilder, an MGC tool that performs cell characterizations. The three steps involved in generating timing accurate symbols for the LP library are:

- collecting the raw data
- determining derating functions
- linking the data to the original symbols.

5.1 CellBuilder

CellBuilder can be used to generate the boolean Quickpart modeling tables, as well as the technology specific timing, capacitance, and constraint models. Because an integral part of low-power ASIC design is using a lower supply voltage, the LP library will be characterized at several discrete supply voltages, 5.0V, 3.3V, 2.5V, 1.8V, and 1.5V.

When using CellBuilder, each of the MOSIS fabrication technologies must be separated into different libraries each with their own technology specific netlists. Under each technology library, simulation 'corners' shall be set for all the supply voltages. The characterizations are performed at each 'corner' with the appropriate output load and inputs slew ranges. In addition to outputting the *.ts* file for the MGC timing compiler, CellBuilder also outputs an HTML datasheet for each part. The HTML description aids in the development of an on-line database for the LP library.

6.2 Temperature and Voltage Derating

To reduce the overhead of factoring in temperature dependence into the LP library, all of the characterizations are performed at a single, typical temperature. Then, within each technology, an inverter cell is simulated across a wide range of temperature values. A best-fit linear curve is fitted onto the inverter propagation delays, and this derating can be imposed on the all the cells for a particular technology.

In contrast, the voltage timing dependence cannot be modeled with a linear process. Previous work in low-power digital gates [3], showed an approximation of the propagation delay in terms of the supply voltage as shown in equation 2. This

$$T_D = \frac{C_L \times V_{DD}}{uC_{OX}(W/L)(V_{DD} - V_T)^{\alpha}} \qquad (2)$$

relationship can be used to generate a voltage dependent derating. An individual cell in the library will maintain a constant output load, oxide thickness, width, and length. For small voltage excursions the ratio between two propagation delays as defined by equation 2 can serve as a derating factor. By using one of the characterized voltages as a timing reference adjusted by equation 3, any voltage can be simulated in the LP library. For voltages close to 5 Volts, the derating factor will be close to 1, but as the reference supply voltage lowers, the derating factor grows exponentially, even for small voltage variations.

$$T_{D1}/T_{D2} = \frac{V_{DD1}(V_{DD2} - V_T)^{\alpha}}{V_{DD2}(V_{DD1} - V_T)^{\alpha}} \qquad (3)$$

Both the temperature and voltage deratings can be added to a DVE simulation setup script. Before simulating with the LP library, a DVE menu lets the user select the particular technology, temperature, and voltage settings.

6.3. Linking to the Part

When all of the characterization runs are completed, the *.ts* timing files and the *.qpt* Quickpart modeling files must be copied in the *$LP_LIB/src/techfiles/<tech>* and *$LP_LIB/src/tables* directories respectively. Using modified scripts from DKE, the Quickpart descriptions are compiled, and linked to the previous defined symbols in the *$LP_LIB/parts* directory. The timing files are also compiled using modified DKE scripts, but each is linked to the symbol using a different technology label. For example, for the Orbit 2.0um technology, technology files shall be linked to the symbols specifying the supply voltage, *scn20orbit_50, scn20orbit_33*, etc. Each symbol contains the timing information for all the technologies and supply voltages, and the different models are accessed by changing the *technology* parameter in design viewpoint.

7. Autologic II and DFT Generation

In order to provide synthesis capability, an Autologic II library must have detailed timing information as well as cell replacement rules. The method for Autologic II model development used for the LP library is to compile the *.ts* files directly. Once the technology files have been created, they can be converted into linearized synthesis libraries using MGC's *TRAMP* program. The three main steps in creating the autologic synthesis libraries are as follows:

- run TRAMP
  -create Autologic II cell definitions
  -NP cell definitions
- MAKETECHINIT
  -create default global data
- GENRULES
  -generate replacement rules

In addition, the design for test (DFT) models shall be generated by calling *TRAMP* with the switch '-*atpg*'. The Autologic II and DFT models shall be stored in the *$LP_LIB/autologic* and *$LP_LIB/dft* directories respectively.

8. Schematics

In order to facilitate accurate transistor-level simulation, schematics can be generated for each library cell using the MDK SDL toolkit. With this format, the same gate-level schematic can be run in Quicksim for digital simulation or in Accusim for analog simulation just by changing the primitive element in the design viewpoint. In order to support the variable supply voltages, the VDD component must be modified to include a globally set supply voltage. A script routine must be written similar to *sdl_prep* from MDK to prepare the analog simulation environment with the correct technologies and supply voltages. Figure 7 shows the inverter schematic with the V_{DD} component set at a voltage of 'voltage'.

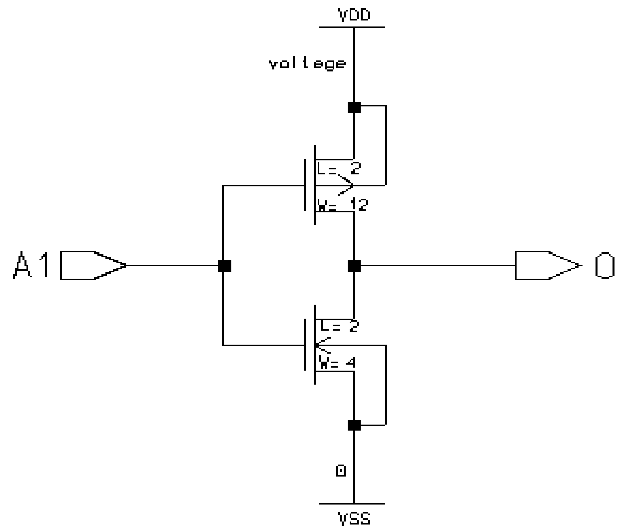

Figure 7: Inverter Schematic

9. Power Models

An area of further development for the LP library is the inclusion of QuickPower models for all the standard cells. The next release of CellBuilder will generate these models, and as soon as it is available, the models will be linked to the library. The models will be a great addition to the library, and they will allow early power analysis in designs with the LP library.

10. Databook Generation

Comprehensive documentation is important for any standard cell library. Two methods of documentation can be used for the LP library. First, a manual can be generated for the BOLD BROWSER system of hyperlinked documents. Using DKE's *gen_db* script, the design models, symbols, and timing description are bound into a single manual. Within this framework, the user can find specific timing information from the BOLD server on any of the cells.

In addition, because CellBuilder provides HTML and Postscript formats of its characterization data, the library information can also be compiled into an on-line server. Eventual support will be a library download site, as well as access to on-line manuals and the cells' information.

11. Conclusions

This paper has presented the design methodology for implementing a standard cell library in the MGC environment. This methodology was developed while converting the layouts from a Magic standard cell library into the MGC environment, but can also be applied to new library development.

The LP library was chosen for this project in order to provide a low-power design environment with the integrated Falcon Framework of MGC for processes available through MOSIS. The LP library has reduced node capacitances for low dynamic power dissipation per transition. In addition, the library has timing data for low-voltage supply, giving designers the ability to design at these levels, and further reducing the system power requirements.

Most of this work has been an investigation into complexity of developing the LP library. To date, the layouts have been imported, and a few of the timing characterizations have been completed. Much work remains to be done, not only to complete the timing models, but also to provide power models, Autologic models, as well as to verify the accuracy of the finished library.

Acknowledgments

This work was partially supported by NSF Career Grant MIP-9703440.

References

[1] Burd, T., *Low-Power CMOS Library Methodology*, Master of Science Thesis, University of California, Berkley.

[2] Chandrakasan, A. et al, *Design Considerations and Tools for Low-Voltage Digital System Design*, 33rd Design Automation Conference Proceedings, 1996, p. 502.

[3] *Mentor Design Kit for MOSIS*, Mentor Graphics Corporation, Software version V1.0_2, 1995.

[4] *Developing ASIC Libraries and Kits Using DKE*, Mentor Graphics Corporation, Soft-version V8.5_1.1, 1995.

Low-Power Design Environment for MOSIS

David Garrett, Joshua Garrett, Mircea Stan
garrett@virginia.edu, http://csissun13.ee.virginia.edu/~dcg3w/home.html
University of Virginia - Department of Electrical Engineering
Charlottesville, VA 29903

Abstract

The release of a CMOS low power library by the Infopad group at UC Berkeley gave engineers the ability to develop low power circuits with standard cells in Magic with minimum geometry layouts [1]. We describe a process that translates this low power standard cell library from Magic into the Mentor Graphics (MGC) environment. Using the Design Kit Example (DKE), we describe the steps to build a cohesive library structure that allows low power ASIC design; schematic capture with Design Architect (DA) or synthesis with Autologic II, digital simulation with Quicksim II, and standard cell layout in IC station.

1. Low-Power Design

A large part of power dissipation in CMOS circuits is due the dynamic switching of capacitive nodes. For a given node in a CMOS circuit, the power dissipation is proportional to the square of the supply voltage, the load capacitance, and the switching activity [2], shown in equation 1. Out of

$$P \propto \alpha C_L (V_{DD})^2 \qquad (1)$$

the three factors, the supply voltage and the load capacitances can be directly controlled at the library level. The rationale behind providing the Low-Power (LP) library as a complement to the MOSIS Design Kit (MDK) [3] is to have a standard cell library that has accurate timing information at lower voltage supply levels, and that maintains minimum capacitances per transistor. Lower supply voltages adversely affect the propagation delays of logic gates, and at voltages near the transistor threshold voltage, the increase in delay is quite large. When designing CMOS circuits at low voltages it is important to have accurate timing in order to verify that timing constraints are still met. Minimum capacitances are achieved by using minimum geometry transistors. This again comes at a cost in performance, and for larger fanouts, driver cells are required. By using minimum geometry layouts, and providing low voltage timing, the LP library will help reduce the power consumption for standard cell designs. Power reduction by lower switching activity will largely depend on circuit implements.

The LP library provides a reduction in capacitances to help with the power consumption, and also arms the designer with accurate timing information for low-voltage operation. With careful simulations to understand the timing impacts, standard cell designs can be fabricating in the MOSIS process with significant power savings. With the use of QuickPower, the LP library can become a useful tool not only designing, but understanding low-power designs in CMOS.

2. Library Structure

As a starting point of the library, all of the various design information need to be partitioned into a library structure. The LP library is based on both DKE [4] and the MDK libraries, and is structured as seen in Figure 1.

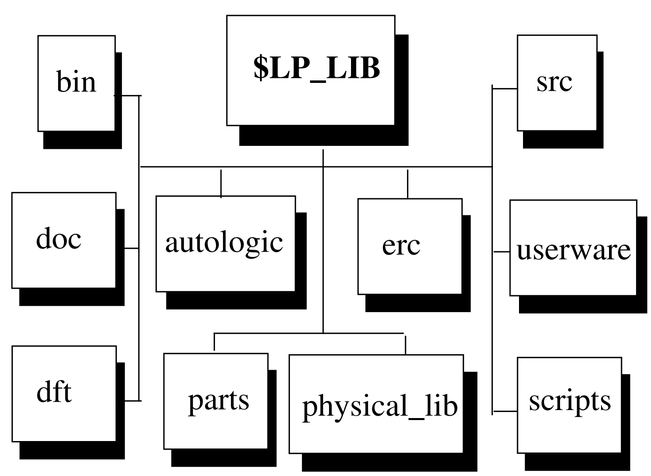

Figure 1: LP Library Directory

The major subdivision are explained as follows:

- **bin** - contains programs
- **doc** - documentation for the library
- **dft** - design for test models
- **autologic** - synthesis libraries
- **parts** - symbols, schematics, and timing
- **physical_lib** - cell layouts
- **src** - source code for timing and models
- **userware** - utilities for MGC environment

3. Symbol Generation

The first step in implementing the library is to generate symbols for all the logic cells in the library. The drawings for the symbols can be copied from the MGC generic library, or generated from scratch in DA. Figure 2 shows an example symbol for an inverter.

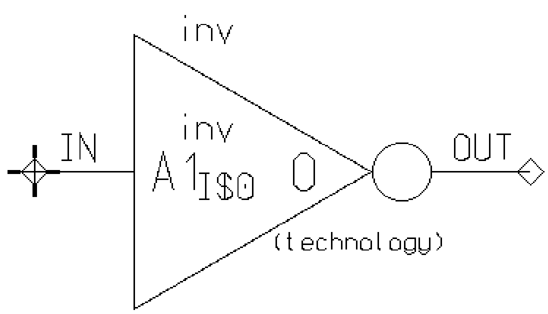

Figure 2: Inverter Symbol

Since many of the technology specific properties are added later in the process, the symbols must be stripped of all but the essential properties. Figure 3 shows the minimum assigned properties for an inverter as reported with *CIB.VIEW* utility from DKE.

4. Design Translation

Almost all of the remaining steps in translating the LP library to the MGC environment depend on having accurate layouts and extracted parameters.

4.1 Importing the Layouts

To make the layouts readable in ICStation, the standard cell layouts must be exported from Magic into a GDS II stream file with the command,

```
COMPONENT  inv  DEFAULT INTEFACE IS: inv

INTERFACE:  inv
   PINS:
              Compiled     User
        Id #   Pin Name   Pin Name  Properties
         1       A1          A1     (pin, A1)
                                    (pintype, IN)
         2       O           O      (pintype, OUT)
                                    (pin, O)
   BODY PROPERTIES:(model, (technology))
                   (inst, I$0)
                   (cell_name, inv)
    NTERFACE MODEL ENTRIES
   Model Entry   Type      Model Info
        0      mgc_symbolPath  : inv/inv
                           Labels: 'default_sym'
                           Status: Valid for inter-
                             face; Valid for
                             property
```

Figure 3: Symbol Properties

calma. These files are then read into IC Station with the GDS II to ICGraph conversion routine, and stored in the *$LP_LIB/physical_lib* directory. The SCMOS process from MDK needs to be linked to all of the cells at conversion time. Figure 4 shows the example inverter layout.

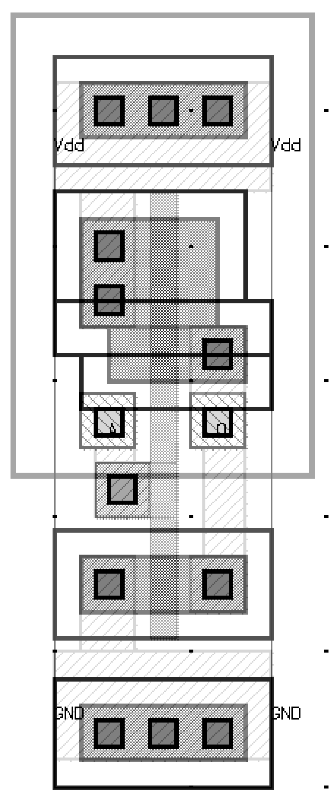

Figure 4: Inverter Layout

Several steps in addition to the conversion process are required to bind the layouts to the symbols for schematic-driven layout (SDL). First, all of the inputs, outputs, and power rails must be copied into the *METAL1.port* and *METAL2.port* layers. Using the *MAKE PORT* command, these layers are designated as ports to represented the input and output pads. The signal names must match exactly to the previously defined symbols. Also, a border around each standard cell shall be defined using the *FP1* layer. The *FP1* layer will completely encompass all of the metal routing within the layouts, but not completely cover the well definitions and select layers. This allows the cells to be placed side-by-side in rows and still share their wells with minimum separation.

Next, each of the physical layouts must be added to an IC Station *LIBRARY.* Using the built-in tools, the *METAL1* and *METAL2* layers are copied to the *METAL1_BLKG* and *METAL2_BLKG* layers and set to an external aspect. This allows the routing tools to avoid shorting metal layers in hierarchial cells. Figure 5 shows the ports, blockages and *FP1* layer in the inverter layout. The ports and the *FP1* layer are designated. The *BLKG* layers are represented by all the layers present in the figure.

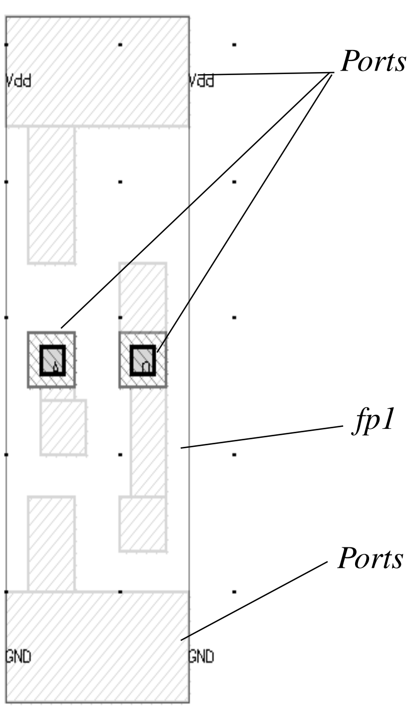

Figure 5: Ports and Blockages

Finally, each of the IC layouts must be set to a "standard cell" with a site type of [1] using the *$set_cell_type()* function. These property tells the automatic placement tools that all of the cells in the library have a standard height.

4.2. Model Extraction

As a starting point for timing development, HSPICE netlists can be extracted from the cell layouts. Because the LP library targets several technologies, a separated netlist must be extracted for each technology due to the differences in minimum feature sizes and parasitics. Initially, the LP library will target three technologies:

- Orbit 2.0um
- HP 1.2um
- HP 0.8um.

The distributed extraction function under the *EXTRACT(M)* menu in ICStation can be used to export an HPSICE netlist for each cell. The netlists are then saved in the *$LP_LIB/src/cellb/ <tech>/netlists* directories, with each set of technology netlists in their respective (*<tech>*) directories. Figure 6 shows a netlist extracted from the inverter layout for the 2.0um Orbit process. Notice that the node names match the port definitions from the layout import step.

```
* File: scn20orbit_lib/netlists/invf101.sp.
      Creation time: Mon Aug 11 14:53:53 1997

.include scn20orbit_lib/netlists/invf101.sp.pex
.subckt invf101 A1 GND O VDD
* distributed parasitics:
x_1 A1 n1_1 n1_2 s_invf101_1
x_2 O n2_1 n2_2 s_invf101_2
x_4 VDD n4_1 n4_2 s_invf101_4
x_3 GND n3_1 n3_2 s_invf101_3
* devices:
m0 n2_1 n1_2 n4_2 n4_1 p l=2u w=12u
m1 n2_2 n1_1 n3_1 n3_2 n l=2u w=4u
.ends invf101
```

Figure 6: Inverter Extracted Netlists

5. Technology File Development

The timing models for the LP library must be based on simulations of various voltage, input slew, and output load vectors. These models can be developed for the LP library with the help of CellBuilder, an MGC tool that performs cell characterizations. The three steps involved in generating timing accurate symbols for the LP library are:

- collecting the raw data
- determining derating functions
- linking the data to the original symbols.

5.1 CellBuilder

CellBuilder can be used to generate the boolean Quickpart modeling tables, as well as the technology specific timing, capacitance, and constraint models. Because an integral part of low-power ASIC design is using a lower supply voltage, the LP library will be characterized at several discrete supply voltages, 5.0V, 3.3V, 2.5V, 1.8V, and 1.5V.

When using CellBuilder, each of the MOSIS fabrication technologies must be separated into different libraries each with their own technology specific netlists. Under each technology library, simulation 'corners' shall be set for all the supply voltages. The characterizations are performed at each 'corner' with the appropriate output load and inputs slew ranges. In addition to outputting the *.ts* file for the MGC timing compiler, CellBuilder also outputs an HTML datasheet for each part. The HTML description aids in the development of an on-line database for the LP library.

6.2 Temperature and Voltage Derating

To reduce the overhead of factoring in temperature dependence into the LP library, all of the characterizations are performed at a single, typical temperature. Then, within each technology, an inverter cell is simulated across a wide range of temperature values. A best-fit linear curve is fitted onto the inverter propagation delays, and this derating can be imposed on the all the cells for a particular technology.

In contrast, the voltage timing dependence cannot be modeled with a linear process. Previous work in low-power digital gates [3], showed an approximation of the propagation delay in terms of the supply voltage as shown in equation 2. This

$$T_D = \frac{C_L \times V_{DD}}{uC_{OX}(W/L)(V_{DD} - V_T)^{\alpha}} \qquad (2)$$

relationship can be used to generate a voltage dependent derating. An individual cell in the library will maintain a constant output load, oxide thickness, width, and length. For small voltage excursions the ratio between two propagation delays as defined by equation 2 can serve as a derating factor. By using one of the characterized voltages as a timing reference adjusted by equation 3, any voltage can be simulated in the LP library. For voltages close to 5 Volts, the derating factor will be close to 1, but as the reference supply voltage lowers, the derating factor grows exponentially, even for small voltage variations.

$$T_{D1}/T_{D2} = \frac{V_{DD1}(V_{DD2} - V_T)^{\alpha}}{V_{DD2}(V_{DD1} - V_T)^{\alpha}} \qquad (3)$$

Both the temperature and voltage deratings can be added to a DVE simulation setup script. Before simulating with the LP library, a DVE menu lets the user select the particular technology, temperature, and voltage settings.

6.3. Linking to the Part

When all of the characterization runs are completed, the *.ts* timing files and the *.qpt* Quickpart modeling files must be copied in the *$LP_LIB/src/techfiles/<tech>* and *$LP_LIB/src/tables* directories respectively. Using modified scripts from DKE, the Quickpart descriptions are compiled, and linked to the previous defined symbols in the *$LP_LIB/parts* directory. The timing files are also compiled using modified DKE scripts, but each is linked to the symbol using a different technology label. For example, for the Orbit 2.0um technology, technology files shall be linked to the symbols specifying the supply voltage, *scn20orbit_50, scn20orbit_33*, etc. Each symbol contains the timing information for all the technologies and supply voltages, and the different models are accessed by changing the *technology* parameter in design viewpoint.

7. Autologic II and DFT Generation

In order to provide synthesis capability, an Autologic II library must have detailed timing information as well as cell replacement rules. The method for Autologic II model development used for the LP library is to compile the *.ts* files directly. Once the technology files have been created, they can be converted into linearized synthesis libraries using MGC's *TRAMP* program. The three main steps in creating the autologic synthesis libraries are as follows:

- run TRAMP
  -create Autologic II cell definitions
  -NP cell definitions
- MAKETECHINIT
  -create default global data
- GENRULES
  -generate replacement rules

In addition, the design for test (DFT) models shall be generated by calling *TRAMP* with the switch '-*atpg*'. The Autologic II and DFT models shall be stored in the *$LP_LIB/autologic* and *$LP_LIB/dft* directories respectively.

8. Schematics

In order to facilitate accurate transistor-level simulation, schematics can be generated for each library cell using the MDK SDL toolkit. With this format, the same gate-level schematic can be run in Quicksim for digital simulation or in Accusim for analog simulation just by changing the primitive element in the design viewpoint. In order to support the variable supply voltages, the VDD component must be modified to include a globally set supply voltage. A script routine must be written similar to *sdl_prep* from MDK to prepare the analog simulation environment with the correct technologies and supply voltages. Figure 7 shows the inverter schematic with the V_{DD} component set at a voltage of 'voltage'.

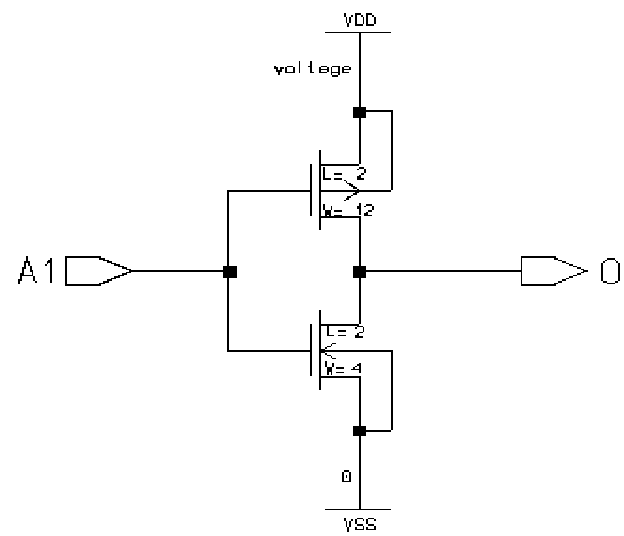

Figure 7: Inverter Schematic

9. Power Models

An area of further development for the LP library is the inclusion of QuickPower models for all the standard cells. The next release of CellBuilder will generate these models, and as soon as it is available, the models will be linked to the library. The models will be a great addition to the library, and they will allow early power analysis in designs with the LP library.

10. Databook Generation

Comprehensive documentation is important for any standard cell library. Two methods of documentation can be used for the LP library. First, a manual can be generated for the BOLD BROWSER system of hyperlinked documents. Using DKE's *gen_db* script, the design models, symbols, and timing description are bound into a single manual. Within this framework, the user can find specific timing information from the BOLD server on any of the cells.

In addition, because CellBuilder provides HTML and Postscript formats of its characterization data, the library information can also be compiled into an on-line server. Eventual support will be a library download site, as well as access to on-line manuals and the cells' information.

11. Conclusions

This paper has presented the design methodology for implementing a standard cell library in the MGC environment. This methodology was developed while converting the layouts from a Magic standard cell library into the MGC environment, but can also be applied to new library development.

The LP library was chosen for this project in order to provide a low-power design environment with the integrated Falcon Framework of MGC for processes available through MOSIS. The LP library has reduced node capacitances for low dynamic power dissipation per transition. In addition, the library has timing data for low-voltage supply, giving designers the ability to design at these levels, and further reducing the system power requirements.

Most of this work has been an investigation into complexity of developing the LP library. To date, the layouts have been imported, and a few of the timing characterizations have been completed. Much work remains to be done, not only to complete the timing models, but also to provide power models, Autologic models, as well as to verify the accuracy of the finished library.

Acknowledgments

This work was partially supported by NSF Career Grant MIP-9703440.

References

[1] Burd, T., *Low-Power CMOS Library Methodology*, Master of Science Thesis, University of California, Berkley.

[2] Chandrakasan, A. et al, *Design Considerations and Tools for Low-Voltage Digital System Design*, 33rd Design Automation Conference Proceedings, 1996, p. 502.

[3] *Mentor Design Kit for MOSIS*, Mentor Graphics Corporation, Software version V1.0_2, 1995.

[4] *Developing ASIC Libraries and Kits Using DKE*, Mentor Graphics Corporation, Soft-version V8.5_1.1, 1995.