IARPA SuperTools Deliverable

ColdFlux Logic Cell Library for MIT-LL SFQ Process:

Part 1 - AQFP

Submitted by

ColdFlux Team
Yokohama National University

Version 3.0.3

Version History

Version 3.0.3

This is the updated version of the cell library for Phase 3 after feedback from the Test & Evaluation teams. The key changes for the AQFP part of the cell library are as follows:

- Updated cell documentation to include missing subcell documentation.
- Added `boost2f4` cell to library.

Version 3.0.2

This is the updated version of the cell library for Phase 3 after feedback from the Test & Evaluation teams. The key changes for the AQFP part of the cell library are as follows:

- An additional border around each cell had been added to ensure DRC clean layouts.
  - This does not include the track block subcells.
- Changed JJ layouts to comply with latest DRC rules.
- Maximum width on M4 is still exceeded, but this DRC error can be waived as M4 is a ground plane.
- General cell library cleanup.

Version 3.0

This is the version of the cell library for Phase 3 release to the Test & Evaluation teams. The key changes for the AQFP part of the cell library are as follows:

- The AQFP cell library is now divided into subcells and main cells.
  - Subcells are meant to be used when designing larger circuits by hand. PTL connections to the subcells are limited to either the input or output ports, or non included at all.
  - Main cells are used for placement-and-routing routines and always have PTL connections at all input and output ports.
- Updated the physical layouts of the AQFP cells to implement the track routing architecture presented in [1].

- Updated circuit netlist and testbench netlist files to implement JoSIM's latest features and standards.
- All combinations of the MAJ5 cell have been developed and included in the library.

Version 2.1.1

This is an updated version of the cell library for Phase 2B after taking into account feedback from the Test & Evaluation teams. The key changes for the AQFP part of the cell library are as follows:

- Replaced the Josephson junction designs of the AQFP cells to that of a grid-snapped design to adhere with MITLL's design grid resolutions.

The key changes for the RSFQ part of the cell library are as follows:

- Minor updates to cell layouts to adhere to latest DRC rules.
- LEF files, extracted using qPALACE, are included in the library.

Version 2.1

This version is the ColdFlux cell library deliverable for SuperTools Phase 2B. The key changes for the AQFP part of the cell library are as follows:

- Completely migrated and re-done schematic/symbol/netlist data from previous Xic formats to the gEDA gschem formats (same format as RSFQ).
- Introduction of new booster cells for driving long distance interconnects.
- Addition of refined MAJ5, AND3, and OR3 logic cells.
- Removal of dummy JJs in schematics/netlists previously used to avoid simulation issues in WRspice/jsim_n. JoSIM is the main simulator now which has no issues with the removal of these dummy JJs.
- LVS-clean cell data verified using InductEx-LVS.

The key changes for the RSFQ part of the cell library are as follows:

- RSFQ cells were redesigned from first principles using phase-based equations.
- Cell versions both with and without integrated PTL transmitters and receivers are included in version 2.1.
- Base circuit netlists are included to show how the base cell is designed. An optimized circuit netlist represents the optimized circuit, as extracted from the layout using InductEx.
- A testbench for each cell is included for easy user verification of cell functionality.
- The mesh file is also included in version 2.1 to allow the user to view the 3D layout as generated by TTH/InductEx.
- An XNOR and XNORT cell was developed.
- Additional Always0 (both synchronous and asynchronous) and Always0T cells were developed.
- Parameterized cell layouts are in development, exposing parameters such as inductor widths and track pitch. This allows scaling and regeneration of layouts while maintaining inductance.

Version 2.0

Updates the AQFP and RSFQ logic cell libraries for SuperTools Phase 2A, and combines both libraries in a single document.

Version 1.5

Previous release of AQFP Logic cell library document.

Version 1.1

Previous release of RSFQ Logic cell library document.

Acknowledgment

The research is based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the U.S. Army Research Office grant W911NF-17-1-0120. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation herein.

Contents

List of Figures

List of Tables

1. Introduction and Setup

1.1 Introduction

With our proposed minimalist design methodology, standard cell libraries have been built for MIT-LL 10 kA/cm^2 Nb/Al−AlO_x/Nb tri-layer process. This cell library contains the basic logic cells, on-chip interfaces and off-chip interfaces as listed below. All cells are driven by 4-phase clock generated by 2 AC sources and a DC source [2].

In the previously released cell libraries (versions 1.4, 1.5, 1.6, and 1.7/2.0), all views of the cell library were developed using Xic from the XicTools suite [3]. From version 2.1 onward, the cell's symbol view and schematic view were developed using gschem [4], whereas the GDS files can be viewed and edited by using Klayout [5]. A team decision amongst the cell library designers in ColdFlux was to drop Xic support due to undesirable user experience and difficulty in integrating ColdFlux tools into the Xic development branch. Further, while Xic had very good support for importing OpenAccess cell libraries, it has extremely limited and buggy capabilities for writing OpenAccess format data. To our knowledge, there are no other open-source tools that can use the OpenAccess API to read/write data, so we will no longer pursue this approach.

Section 1.3 details our naming conventions used for this cell library and the full list of cells is shown in Section 1.4. The library files for circuit design are GDS files that can be opened in Klayout or any other tool that can open standard GDS files.

The components of each cell are listed as follows:

- **Symbol**: These files end with `.sym`. This is the symbol view of the cell. It is designed in gschem and provides a hierarchical way to build digital logic circuits. It is a simple symbol with terminal (I/O) definitions and a file attribute which points to the appropriate sub-circuit definition (`.cir` file).

- **Schematic**: These files end with `.sch`. This is the schematic view of the cell designed in gschem. The circuit parameters have been extracted from the physical layout using InductEx [6]. We define the parameters using the `.param` directive that JoSIM supports. This allows easier manipulation of circuit parameters by other tools such as JoSIM tools.

- **Layout**: These files end with `.gds`. This is the physical layout of the cell designed in KLayout. The GDS files are now standard format unlike in previous versions (pre-v2.1) which also embed schematic and symbol views readable only by Xic. As mentioned above, the symbols and schematics are in separate files now. InductEx extractions can be performed on all layout files of the base cells (bfr, inv, branch3, etc).

- **Base netlist**: The base netlists (appended with `_base.cir` provides the base design of the cell, similar to the RSFQ cell library. The base netlist is also used by InductEx to back-annotate the extracted parameter values from the physical layout. The back-annotated values are used to construct the standard netlist of the cell.
- **Standard netlist**: The `.cir` files have been updated from the previous library versions to include the latest functionalities and standards of JoSIM [7]. Instead of including all the subcircuits within the larger netlist, the `.include` function is used to reference the specified netlist. This not only compacts the netlists of larger cells, but also ensures that the latest version of the subcircuit is referenced.
- **InductEx netlist**: These files end with `.ix.cir`. This is the netlist used for extracting inductances and is a slightly different format compared to the standard netlist.
- **LVS file** : These files end with `.json`. This the LVS configuration file already prepared to run LVS via `$ inductex-lvs (cellname).json`.
- **Analog waveform**: Using the analog circuit testbenches (prefixed with `test_*`) included in the cell library, functionality was confirmed using JoSIM [7].
- **Verilog model**: Behavior-level model of a cell with built-in timing specification block. It is written in hardware-description-language (HDL) Verilog and can be found as a separate file named 'CellName.v' under the `verilog` sub-directory. Each cell model has an embedded module named 'biasDir_b.v' as an I/O interface to produce a normalized clock based on the relative directions the AC and DC current.
- **SDF**: Standard delay format file containing propagation delays and timing constraints for each cell. Generated by the AQFP timing extraction tool AQFPTX [8]. This file is the `verilog` subdirectory.
- **Digital waveform**: Generated by the open source Verilog simulator Icarus-Verilog [9] and viewed using the open source wave viewer GTKWave [10].
- **Switching energy**: Switching energies of each logic cell are calculated based on the methodology described in [11]. The energy information is provided as an input-frequency dependent look-up table (.csv) and can be found as a separate file named 'energy_cellName.csv' under the `energy` sub-directory. This lookup table summaries the energy evolution based on different operating frequency and input Boolean logic patterns, which is compatible with the probabilistic power analysis tool AQFP-QPA.

1.2 Cell library structure

```
coldflux-aqfp-cell-library
|    README.md: readme file
|
|--- lib_jjmit: cell data (schematic, symbol, layout, test schematics)
|    |--- doc: PDF of current documentation of the library
```

```
|    |--- energy: Switching energy tables for each cell
|    |--- figures: Various screenshots for documentation
|    |--- lef: Contains LEF file for the library
|    |--- lvs: LVS results of the cells produced by inductex-lvs
|    |--- tech: tech and ldf files for inductex, lvs
|    |--- verilog: Verilog models including global SDF
|
|--- legacy: old files and documentation for reference purposes.
     |   cell-lib-doc-resources: contains .docx files for PDFs.
```

1.3 Conventions

The pin naming conventions for the logic cells in the AQFP library is listed in Table 1.1. The logic cells themselves are named using the following convention:

```
<gate><inputs>_<polarities>_<type>
```

Table 1.2 describes in detail the different naming components and Table 1.3 shows some examples.

The AQFP library also includes some track blocks for the construction of larger circuits. The track blocks are named using the following convention:

```
tr_<type>_<layer>_<shape>_<connection>
```

Table 1.4 describes in detail the different naming components and Table 1.5 shows some examples. An `x` indicates that there are wires crossing. For this version of the AQFP cell library we also assume that a PTL on M1 will always run horizontally and a PTL on M3 always runs vertically. We have, however, included additional interconnect slices which allows wires on M1 and M3 in all directions, including various corner interconnect slices.

Table 1.1: Pin naming conventions.

Name	**Convention**
Input	a, b, c, d...
Output (single)	q
Output (multi)	$q0$, $q1$, $q2$...
AC excitation (single)	xin / $xout$
AC excitation (multi)	$xin1$, $xin2$, $xin3$... / $xout1$, $xout2$, $xout3$...
DC offset (single)	$dcin$ / $dcout$
DC offset (multi)	$dcin1$, $dcin2$, $dcin3$... / $dcout1$, $dcout2$, $dcout3$...

Table 1.2: Naming convention of AQFP logic cells.

Name Component	Description
<gate>	This is the shortened name of the gate. In the case where the gate has one or zero (e.g., constant cells) inputs, we simply use the <gate>.
<inputs>	This is the number of logic inputs, i.e., the number of inputs that directly produces the logic function. It doesn't include clocking inputs. In the case of fan-out circuits (spl2, spl3), the number indicates number of outputs/fan-out.
<polarities>	For each logic input i (a, b, c, $d\ldots$), we designate its polarity as positive (p) or negative (n). The polarity mainly applies to Boolean logic gates like MAJ, OR, and AND. It is not used for splitter (spl2 or spl3), for example. The ordering explicitly specifies exactly which input is which polarity. For example: ppn designates that input a is positive, input b is positive, and input c is negative.
<type>	The AQFP cell library is divided into subcells and main cells. Subcells are meant to be used when designing larger circuits by hand. PTL connections to the subcells are limited to either the input or output ports, or non included at all. Subcells have the sub suffix and the i or o suffix indicates a PTL connection at the input or output port respectively. Main cells are used for placement-and-routing routines and always have PTL connections at all input and output ports. Main cells do not have the <type> suffix.

Table 1.3: Naming examples and explanations.

Name	Explanation
maj3_ppn	A 3-input majority gate. Input a is positive, input b is positive, and input c is negative.
or2_pp_sub	A 2-input OR gate subcell. Both input a and b are positive inputs. There are no PTL connections.
spl3	A 1-input to 3-output splitter/fan-out.
branch3_sub_o	A 1-input to 3-output branch subcell with a PTL connection at the output.

Table 1.4: Naming convention of AQFP track blocks.

Name Component	Description
tr	This indicates that the cell is a track block.
<type>	This indicates the type of track block, for example `wire` indicates that it is a track block with a wire, `bias_pair` indicates that the track block contains the AC and DC bias pair lines and `conn` indicates a connection between metal layers.
<layer>	This indicates the metal layer on which connections are found within the track block.
<shape>	For track blocks which contain wires, the shape of the wire is described. For example `c` indicates a corner and `T` indicates a t-shaped wire.
<connection>	This indicates the position of the connections for the track blocks. For example `t` indicates 'top', `b` indicates 'bottom', `l` indicates 'left' and `r` refers to 'right'.

Table 1.5: Naming examples and explanations for the track block cells.

Name	Explanation
tr_bias_pair_10um	A track block with the AC and DC bias pair. The width of the cell is 10um.
tr_bias_DCxAC_tl	A track block with the DC bias line crossing (indicated by the `x`) the AC bias line. There is a connection to the DC line at the top and left side of the cell. In this case, the AC line runs from the left to right side of the cell, uninterrupted.
tr_wire_M1M3_c_br	A track block with a wire on both layers M1 and M3. The wire has a corner shape with connections to the bottom and right side of the cell. M1 always runs horizontally and M3 always runs vertically. So in this case the bottom connection will be on M3 and the connection on the right side will be on M1.

1.4 Full List of AQFP Cells

Table 1.6 shows a full listing of our AQFP cells. The cell name, type of cell, number of Josephson junctions (JJs), height and width in µm, and the Boolean function for each cell is shown. Type is defined as follows: 'S' refers to sub-cells which are used only for building logic cells and are not normally used standalone; 'C' refers to normal logic cells which are used to build digital circuits; 'B' refers to structures that can be both 'S' and 'C'; 'M' refers to macro-type cells which are composed of 'B' or 'C' type cells. 'M' type cells do not have a fixed area as its own placement is dependent on how other neighboring cells are also arranged. This also implies that we do not provide a fixed GDS layout as the place-and-route is responsible for generating the macro layout. 'I' type cells are interface cells for communicating off-chip or on-chip between AQFP-SFQ logic families. 'W' type cells are interconnect slices for both bias and data signal propagation.

Table 1.6: Summary listing of all AQFP cells to date.

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
bfr_sub	B	2	40	30	$q = a$
bfr_sub_i	S	2	40	30	$q = a$
bfr_sub_o	S	2	40	30	$q = a$
bfr	C	2	40	30	$q = a$
inv_sub	B	2	40	30	$q = \bar{a}$
inv_sub_i	S	2	40	30	$q = \bar{a}$
inv_sub_o	S	2	40	30	$q = \bar{a}$
inv	C	2	40	30	$q = \bar{a}$
const0_sub	B	2	40	30	$q = 0$
const0	C	2	40	30	$q = 0$
const1_sub	B	2	40	30	$q = 1$
const1	C	2	40	30	$q = 1$
boost1_sub	B	4	40	50	$q = a$
boost1_sub_i	S	4	40	50	$q = a$
boost1_sub_o	S	4	40	50	$q = a$
boost1	C	4	40	50	$q = a$
boost2f2_sub	B	6	40	90	$q_0, q_1 = a$
boost2f2_sub_i	S	6	40	90	$q_0, q_1 = a$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
boost2f2_sub_o	S	6	40	90	$q_0, q_1 = a$
boost2f2	C	6	40	90	$q_0, q_1 = a$
boost2f4_sub	B	10	40	130	$q_0, q_1, q_2, q_3 = a$
boost2f4	C	10	40	130	$q_0, q_1, q_2, q_3 = a$
bfrL_sub	B	2	40	30	$q = a$
bfrL_sub_i	S	2	40	30	$q = a$
bfrL_sub_o	S	2	40	30	$q = a$
bfrL	C	2	40	30	$q = a$
spl2L_sub	B	2	60	50	$q_0, q_1 = a$
spl2L	C	2	60	50	$q_0, q_1 = a$
spl3L_sub	B	2	60	70	$q_0, q_1, q_2 = a$
spl3L	C	2	60	70	$q_0, q_1, q_2 = a$
spl2_sub	B	2	60	50	$q_0, q_1 = a$
spl2	C	2	60	50	$q_0, q_1 = a$
spl3_sub	B	2	60	70	$q_0, q_1, q_2 = a$
spl3	C	2	60	70	$q_0, q_1, q_2 = a$
maj3_ppp_sub	B	6	60	70	$q = \mathrm{MAJ}(a, b, c) = ab + ac + bc$
maj3_ppp	C	6	60	70	$q = \mathrm{MAJ}(a, b, c) = ab + ac + bc$
maj3_ppn_sub	B	6	60	70	$q = \mathrm{MAJ}(a, b, \bar{c}) = ab + a\bar{c} + b\bar{c}$
maj3_ppn	C	6	60	70	$q = \mathrm{MAJ}(a, b, \bar{c}) = ab + a\bar{c} + b\bar{c}$
maj3_pnp_sub	B	6	60	70	$q = \mathrm{MAJ}(a, \bar{b}, c) = a\bar{b} + ac + \bar{b}c$
maj3_pnp	C	6	60	70	$q = \mathrm{MAJ}(a, \bar{b}, c) = a\bar{b} + ac + \bar{b}c$
maj3_pnn_sub	B	6	6	70	$q = \mathrm{MAJ}(a, \bar{b}, \bar{c}) = a\bar{b} + a\bar{c} + \bar{b}\bar{c}$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
maj3_pnn	C	6	6	70	$q = \text{MAJ}(a, \bar{b}, \bar{c}) = a\bar{b} + a\bar{c} + \bar{b}\bar{c}$
maj3_npp_sub	B	6	60	70	$q = \text{MAJ}(\bar{a}, b, c) = \bar{a}b + \bar{a}c + bc$
maj3_npp	C	6	60	70	$q = \text{MAJ}(\bar{a}, b, c) = \bar{a}b + \bar{a}c + bc$
maj3_npn_sub	B	6	60	70	$q = \text{MAJ}(\bar{a}, b, \bar{c}) = \bar{a}b + \bar{a}\bar{c} + b\bar{c}$
maj3_npn	C	6	60	70	$q = \text{MAJ}(\bar{a}, b, \bar{c}) = \bar{a}b + \bar{a}\bar{c} + b\bar{c}$
maj3_nnp_sub	B	6	60	70	$q = \text{MAJ}(\bar{a}, \bar{b}, c) = \bar{a}\bar{b} + \bar{a}c + \bar{b}c$
maj3_nnp	C	6	60	70	$q = \text{MAJ}(\bar{a}, \bar{b}, c) = \bar{a}\bar{b} + \bar{a}c + \bar{b}c$
maj3_nnn_sub	B	6	60	70	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}) = \bar{a}\bar{b} + \bar{a}\bar{c} + \bar{b}\bar{c} = \text{MIN}(a, b, c)$
maj3_nnn	C	6	60	70	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}) = \bar{a}\bar{b} + \bar{a}\bar{c} + \bar{b}\bar{c} = \text{MIN}(a, b, c)$
maj5_ppppp_sub	B	10	70	110	$q = \text{MAJ}(a, b, c, d, e) = abc+abd+abe+acd+ace+ade+bcd+bce+bde+cde$
maj5_ppppp	C	10	70	110	$q = \text{MAJ}(a, b, c, d, e) = abc+abd+abe+acd+ace+ade+bcd+bce+bde+cde$
maj5_ppppn_sub	B	10	70	110	$q = \text{MAJ}(a, b, c, d, \bar{e}) = abc+abd+ab\bar{e}+acd+ac\bar{e}+ad\bar{e}+bcd+bc\bar{e}+bd\bar{e}+cd\bar{e}$
maj5_ppppn	C	10	70	110	$q = \text{MAJ}(a, b, c, d, \bar{e}) = abc+abd+ab\bar{e}+acd+ac\bar{e}+ad\bar{e}+bcd+bc\bar{e}+bd\bar{e}+cd\bar{e}$
maj5_pppnp_sub	B	10	70	110	$q = \text{MAJ}(a, b, c, \bar{d}, e) = abc+ab\bar{d}+abe+ac\bar{d}+ace+a\bar{d}e+bc\bar{d}+bce+b\bar{d}e+c\bar{d}e$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
maj5_pppnp	C	10	70	110	$q = \text{MAJ}(a,b,c,\bar{d},e) = abc+ab\bar{d}+abe+ac\bar{d}+ace+a\bar{d}e+bc\bar{d}+bce+b\bar{d}e+c\bar{d}e$
maj5_ppnpp_sub	B	10	70	110	$q = \text{MAJ}(a,b,\bar{c},d,e) = ab\bar{c}+abd+abe+a\bar{c}d+a\bar{c}e+ade+b\bar{c}d+b\bar{c}e+bde+\bar{c}de$
maj5_ppnpp	C	10	70	110	$q = \text{MAJ}(a,b,\bar{c},d,e) = ab\bar{c}+abd+abe+a\bar{c}d+a\bar{c}e+ade+b\bar{c}d+b\bar{c}e+bde+\bar{c}de$
maj5_pnppp_sub	B	10	70	110	$q = \text{MAJ}(a,\bar{b},c,d,e) = a\bar{b}c + a\bar{b}d + a\bar{b}e + acd + ace+ade+\bar{b}cd+b = \bar{b}ce+\bar{b}de + cde$
maj5_pnppp	C	10	70	110	$q = \text{MAJ}(a,\bar{b},c,d,e) = a\bar{b}c + a\bar{b}d + a\bar{b}e + acd + ace+ade+\bar{b}cd+b = \bar{b}ce+\bar{b}de + cde$
maj5_npppp_sub	B	10	70	110	$q = \text{MAJ}(\bar{a},b,c,d,e) = \bar{a}bc+\bar{a}bd+\bar{a}be+\bar{a}cd+\bar{a}ce+\bar{a}de+bcd+bce+bde+cde$
maj5_npppp	C	10	70	110	$q = \text{MAJ}(\bar{a},b,c,d,e) = \bar{a}bc+\bar{a}bd+\bar{a}be+\bar{a}cd+\bar{a}ce+\bar{a}de+bcd+bce+bde+cde$
maj5_pppnn_sub	B	10	70	110	$q = \text{MAJ}(a,b,c,\bar{d},\bar{e}) = abc+ab\bar{d}+ab\bar{e}+ac\bar{d}+ac\bar{e}+a\bar{d}\bar{e}+bc\bar{d}+bc\bar{e}+b\bar{d}\bar{e}+c\bar{d}\bar{e}$
maj5_pppnn	C	10	70	110	$q = \text{MAJ}(a,b,c,\bar{d},\bar{e}) = abc+ab\bar{d}+ab\bar{e}+ac\bar{d}+ac\bar{e}+a\bar{d}\bar{e}+bc\bar{d}+bc\bar{e}+b\bar{d}\bar{e}+c\bar{d}\bar{e}$
maj5_ppnpn_sub	B	10	70	110	$q = \text{MAJ}(a,b,\bar{c},d,\bar{e}) = ab\bar{c}+abd+ab\bar{e}+a\bar{c}d+a\bar{c}\bar{e}+ad\bar{e}+b\bar{c}d+b\bar{c}\bar{e}+bd\bar{e}+\bar{c}d\bar{e}$
maj5_ppnpn	C	10	70	110	$q = \text{MAJ}(a,b,\bar{c},d,\bar{e}) = ab\bar{c}+abd+ab\bar{e}+a\bar{c}d+a\bar{c}\bar{e}+ad\bar{e}+b\bar{c}d+b\bar{c}\bar{e}+bd\bar{e}+\bar{c}d\bar{e}$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
maj5_pnppn_sub	B	10	70	110	$q = \text{MAJ}(a, \bar{b}, c, d, \bar{e}) = a\bar{b}c + a\bar{b}d + a\bar{b}\bar{e} + acd + ac\bar{e} + ad\bar{e} + \bar{b}cd + \bar{b}c\bar{e} + \bar{b}d\bar{e} + cd\bar{e}$
maj5_pnppn	C	10	70	110	$q = \text{MAJ}(a, \bar{b}, c, d, \bar{e}) = a\bar{b}c + a\bar{b}d + a\bar{b}\bar{e} + acd + ac\bar{e} + ad\bar{e} + \bar{b}cd + \bar{b}c\bar{e} + \bar{b}d\bar{e} + cd\bar{e}$
maj5_npppn_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, b, c, d, \bar{e}) = \bar{a}bc + \bar{a}bd + \bar{a}b\bar{e} + \bar{a}cd + \bar{a}c\bar{e} + \bar{a}d\bar{e} + bcd + bc\bar{e} + bd\bar{e} + cd\bar{e}$
maj5_npppn	C	10	70	110	$q = \text{MAJ}(\bar{a}, b, c, d, \bar{e}) = \bar{a}bc + \bar{a}bd + \bar{a}b\bar{e} + \bar{a}cd + \bar{a}c\bar{e} + \bar{a}d\bar{e} + bcd + bc\bar{e} + bd\bar{e} + cd\bar{e}$
maj5_ppnnp_sub	B	10	70	110	$q = \text{MAJ}(a, b, \bar{c}, \bar{d}, e) = ab\bar{c} + ab\bar{d} + abe + a\bar{c}\bar{d} + a\bar{c}e + a\bar{d}e + b\bar{c}\bar{d} + b\bar{c}e + b\bar{d}e + \bar{c}\bar{d}e$
maj5_ppnnp	C	10	70	110	$q = \text{MAJ}(a, b, \bar{c}, \bar{d}, e) = ab\bar{c} + ab\bar{d} + abe + a\bar{c}\bar{d} + a\bar{c}e + a\bar{d}e + b\bar{c}\bar{d} + b\bar{c}e + b\bar{d}e + \bar{c}\bar{d}e$
maj5_pnpnp_sub	B	10	70	110	$q = \text{MAJ}(a, \bar{b}, c, \bar{d}, e) = a\bar{b}c + a\bar{b}\bar{d} + a\bar{b}e + ac\bar{d} + ace + a\bar{d}e + \bar{b}c\bar{d} + \bar{b}ce + \bar{b}\bar{d}e + c\bar{d}e$
maj5_pnpnp	C	10	70	110	$q = \text{MAJ}(a, \bar{b}, c, \bar{d}, e) = a\bar{b}c + a\bar{b}\bar{d} + a\bar{b}e + ac\bar{d} + ace + a\bar{d}e + \bar{b}c\bar{d} + \bar{b}ce + \bar{b}\bar{d}e + c\bar{d}e$
maj5_nppnp_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, b, c, \bar{d}, e) = \bar{a}bc + \bar{a}b\bar{d} + \bar{a}be + \bar{a}c\bar{d} + \bar{a}ce + \bar{a}\bar{d}e + bc\bar{d} + bce + b\bar{d}e + c\bar{d}e$
maj5_nppnp	C	10	70	110	$q = \text{MAJ}(\bar{a}, b, c, \bar{d}, e) = \bar{a}bc + \bar{a}b\bar{d} + \bar{a}be + \bar{a}c\bar{d} + \bar{a}ce + \bar{a}\bar{d}e + bc\bar{d} + bce + b\bar{d}e + c\bar{d}e$
maj5_pnnpp_sub	B	10	70	110	$q = \text{MAJ}(a, \bar{b}, \bar{c}, d, e) = a\bar{b}\bar{c} + a\bar{b}d + a\bar{b}e + a\bar{c}d + a\bar{c}e + ade + \bar{b}\bar{c}d + \bar{b}\bar{c}e + \bar{b}de + \bar{c}de$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
maj5_pnnpp	C	10	70	110	$q = \text{MAJ}(a, \bar{b}, \bar{c}, d, e) = a\bar{b}\bar{c}+a\bar{b}d+a\bar{b}e+a\bar{c}d+a\bar{c}e+ade+\bar{b}\bar{c}d+\bar{b}\bar{c}e+\bar{b}de+\bar{c}de$
maj5_npnpp_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, b, \bar{c}, d, e) = \bar{a}b\bar{c}+\bar{a}bd+\bar{a}be+\bar{a}\bar{c}d+\bar{a}\bar{c}e+\bar{a}de+b\bar{c}d+b\bar{c}e+bde+\bar{c}de$
maj5_npnpp	C	10	70	110	$q = \text{MAJ}(\bar{a}, b, \bar{c}, d, e) = \bar{a}b\bar{c}+\bar{a}bd+\bar{a}be+\bar{a}\bar{c}d+\bar{a}\bar{c}e+\bar{a}de+b\bar{c}d+b\bar{c}e+bde+\bar{c}de$
maj5_nnppp_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, c, d, e) = \bar{a}\bar{b}c+\bar{a}\bar{b}d+\bar{a}\bar{b}e+\bar{a}cd+\bar{a}ce+\bar{a}de+\bar{b}cd+\bar{b}ce+\bar{b}de+cde$
maj5_nnppp	C	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, c, d, e) = \bar{a}\bar{b}c+\bar{a}\bar{b}d+\bar{a}\bar{b}e+\bar{a}cd+\bar{a}ce+\bar{a}de+\bar{b}cd+\bar{b}ce+\bar{b}de+cde$
maj5_ppnnn_sub	B	10	70	110	$q = \text{MAJ}(a, b, \bar{c}, \bar{d}, e\bar{e}) = ab\bar{c}+ab\bar{d}+ab\bar{e}+a\bar{c}\bar{d}+a\bar{c}\bar{e}+a\bar{d}\bar{e}+b\bar{c}\bar{d}+b\bar{c}\bar{e}+b\bar{d}\bar{e}+\bar{c}\bar{d}\bar{e}$
maj5_ppnnn	C	10	70	110	$q = \text{MAJ}(a, b, \bar{c}, \bar{d}, \bar{e}) = ab\bar{c}+ab\bar{d}+ab\bar{e}+a\bar{c}\bar{d}+a\bar{c}\bar{e}+a\bar{d}\bar{e}+b\bar{c}\bar{d}+b\bar{c}\bar{e}+b\bar{d}\bar{e}+\bar{c}\bar{d}\bar{e}$
maj5_pnpnn_sub	B	10	70	110	$q = \text{MAJ}(a, \bar{b}, c, \bar{d}, \bar{e}) = a\bar{b}c+a\bar{b}\bar{d}+a\bar{b}\bar{e}+ac\bar{d}+ac\bar{e}+a\bar{d}\bar{e}+\bar{b}c\bar{d}+\bar{b}c\bar{e}+\bar{b}\bar{d}\bar{e}+c\bar{d}\bar{e}$
maj5_pnpnn	C	10	70	110	$q = \text{MAJ}(a, \bar{b}, c, \bar{d}, \bar{e}) = a\bar{b}c+a\bar{b}\bar{d}+a\bar{b}\bar{e}+ac\bar{d}+ac\bar{e}+a\bar{d}\bar{e}+\bar{b}c\bar{d}+\bar{b}c\bar{e}+\bar{b}\bar{d}\bar{e}+c\bar{d}\bar{e}$
maj5_nppnn_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, b, c, \bar{d}, \bar{e}) = \bar{a}bc+\bar{a}b\bar{d}+\bar{a}b\bar{e}+\bar{a}c\bar{d}+\bar{a}c\bar{e}+\bar{a}\bar{d}\bar{e}+bc\bar{d}+bc\bar{e}+b\bar{d}\bar{e}+c\bar{d}\bar{e}$
maj5_nppnn	C	10	70	110	$q = \text{MAJ}(\bar{a}, b, c, \bar{d}, \bar{e}) = \bar{a}bc+\bar{a}b\bar{d}+\bar{a}b\bar{e}+\bar{a}c\bar{d}+\bar{a}c\bar{e}+\bar{a}\bar{d}\bar{e}+bc\bar{d}+bc\bar{e}+b\bar{d}\bar{e}+c\bar{d}\bar{e}$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
maj5_pnnnp_sub	B	10	70	110	$q = \text{MAJ}(a, \bar{b}, \bar{c}, \bar{d}, e) = a\bar{b}\bar{c}+a\bar{b}\bar{d}+a\bar{b}e+a\bar{c}\bar{d}+a\bar{c}e+a\bar{d}e+\bar{b}\bar{c}\bar{d}+\bar{b}\bar{c}e+\bar{b}\bar{d}e+\bar{c}\bar{d}e$
maj5_pnnnp	C	10	70	110	$q = \text{MAJ}(a, \bar{b}, \bar{c}, \bar{d}, e) = a\bar{b}\bar{c}+a\bar{b}\bar{d}+a\bar{b}e+a\bar{c}\bar{d}+a\bar{c}e+a\bar{d}e+\bar{b}\bar{c}\bar{d}+\bar{b}\bar{c}e+\bar{b}\bar{d}e+\bar{c}\bar{d}e$
maj5_npnnp_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, b, \bar{c}, \bar{d}, e) = \bar{a}b\bar{c}+\bar{a}b\bar{d}+\bar{a}be+\bar{a}\bar{c}\bar{d}+\bar{a}\bar{c}e+\bar{a}\bar{d}e+b\bar{c}\bar{d}+b\bar{c}e+b\bar{d}e+\bar{c}\bar{d}e$
maj5_npnnp	C	10	70	110	$q = \text{MAJ}(\bar{a}, b, \bar{c}, \bar{d}, e) = \bar{a}b\bar{c}+\bar{a}b\bar{d}+\bar{a}be+\bar{a}\bar{c}\bar{d}+\bar{a}\bar{c}e+\bar{a}\bar{d}e+b\bar{c}\bar{d}+b\bar{c}e+b\bar{d}e+\bar{c}\bar{d}e$
maj5_nnnpp_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}, d, e) = \bar{a}\bar{b}\bar{c}+\bar{a}\bar{b}d+\bar{a}\bar{b}e+\bar{a}\bar{c}d+\bar{a}\bar{c}e+\bar{a}de+\bar{b}\bar{c}d+\bar{b}\bar{c}e+\bar{b}de+\bar{c}de$
maj5_nnnpp	C	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}, d, e) = \bar{a}\bar{b}\bar{c}+\bar{a}\bar{b}d+\bar{a}\bar{b}e+\bar{a}\bar{c}d+\bar{a}\bar{c}e+\bar{a}de+\bar{b}\bar{c}d+\bar{b}\bar{c}e+\bar{b}de+\bar{c}de$
maj5_nnpnp_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, c, \bar{d}, e) = \bar{a}\bar{b}c+\bar{a}\bar{b}\bar{d}+\bar{a}\bar{b}e+\bar{a}c\bar{d}+\bar{a}ce+\bar{a}\bar{d}e+\bar{b}c\bar{d}+\bar{b}ce+\bar{b}\bar{d}e+c\bar{d}e$
maj5_nnpnp	C	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, c, \bar{d}, e) = \bar{a}\bar{b}c+\bar{a}\bar{b}\bar{d}+\bar{a}\bar{b}e+\bar{a}c\bar{d}+\bar{a}ce+\bar{a}\bar{d}e+\bar{b}c\bar{d}+\bar{b}ce+\bar{b}\bar{d}e+c\bar{d}e$
maj5_pnnnn_sub	B	10	70	110	$q = \text{MAJ}(a, \bar{b}, \bar{c}, \bar{d}, \bar{e}) = a\bar{b}\bar{c}+a\bar{b}\bar{d}+a\bar{b}\bar{e}+a\bar{c}\bar{d}+a\bar{c}\bar{e}+a\bar{d}\bar{e}+\bar{b}\bar{c}\bar{d}+\bar{b}\bar{c}\bar{e}+\bar{b}\bar{d}\bar{e}+\bar{c}\bar{d}\bar{e}$
maj5_pnnnn	C	10	70	110	$q = \text{MAJ}(a, \bar{b}, \bar{c}, \bar{d}, \bar{e}) = a\bar{b}\bar{c}+a\bar{b}\bar{d}+a\bar{b}\bar{e}+a\bar{c}\bar{d}+a\bar{c}\bar{e}+a\bar{d}\bar{e}+\bar{b}\bar{c}\bar{d}+\bar{b}\bar{c}\bar{e}+\bar{b}\bar{d}\bar{e}+\bar{c}\bar{d}\bar{e}$
maj5_npnnn_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, b, \bar{c}, \bar{d}, \bar{e}) = \bar{a}b\bar{c}+\bar{a}b\bar{d}+\bar{a}b\bar{e}+\bar{a}\bar{c}\bar{d}+\bar{a}\bar{c}\bar{e}+\bar{a}\bar{d}\bar{e}+b\bar{c}\bar{d}+b\bar{c}\bar{e}+b\bar{d}\bar{e}+\bar{c}\bar{d}\bar{e}$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
maj5_npnnn	C	10	70	110	$q = \text{MAJ}(\bar{a}, b, \bar{c}, \bar{d}, \bar{e}) = \bar{a}b\bar{c}+\bar{a}b\bar{d}+\bar{a}b\bar{e}+\bar{a}\bar{c}\bar{d}+\bar{a}\bar{c}\bar{e}+\bar{a}\bar{d}\bar{e}+b\bar{c}\bar{d}+b\bar{c}\bar{e}+b\bar{d}\bar{e}+\bar{c}\bar{d}\bar{e}$
maj5_nnpnn_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, c, \bar{d}, \bar{e}) = \bar{a}\bar{b}c+\bar{a}\bar{b}\bar{d}+\bar{a}\bar{b}\bar{e}+\bar{a}c\bar{d}+\bar{a}c\bar{e}+\bar{a}\bar{d}\bar{e}+\bar{b}c\bar{d}+\bar{b}c\bar{e}+\bar{b}\bar{d}\bar{e}+c\bar{d}\bar{e}$
maj5_nnpnn	C	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, c, \bar{d}, \bar{e}) = \bar{a}\bar{b}c+\bar{a}\bar{b}\bar{d}+\bar{a}\bar{b}\bar{e}+\bar{a}c\bar{d}+\bar{a}c\bar{e}+\bar{a}\bar{d}\bar{e}+\bar{b}c\bar{d}+\bar{b}c\bar{e}+\bar{b}\bar{d}\bar{e}+c\bar{d}\bar{e}$
maj5_nnnpn_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}, d, \bar{e}) = \bar{a}\bar{b}\bar{c}+\bar{a}\bar{b}d+\bar{a}\bar{b}\bar{e}+\bar{a}\bar{c}d+\bar{a}\bar{c}\bar{e}+\bar{a}d\bar{e}+\bar{b}\bar{c}d+\bar{b}\bar{c}\bar{e}+\bar{b}d\bar{e}+\bar{c}d\bar{e}$
maj5_nnnpn	C	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}, d, \bar{e}) = \bar{a}\bar{b}\bar{c}+\bar{a}\bar{b}d+\bar{a}\bar{b}\bar{e}+\bar{a}\bar{c}d+\bar{a}\bar{c}\bar{e}+\bar{a}d\bar{e}+\bar{b}\bar{c}d+\bar{b}\bar{c}\bar{e}+\bar{b}d\bar{e}+\bar{c}d\bar{e}$
maj5_nnnnp_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}, \bar{d}, e) = \bar{a}\bar{b}\bar{c}+\bar{a}\bar{b}\bar{d}+\bar{a}\bar{b}e+\bar{a}\bar{c}\bar{d}+\bar{a}\bar{c}e+\bar{a}\bar{d}e+\bar{b}\bar{c}\bar{d}+\bar{b}\bar{c}e+\bar{b}\bar{d}e+\bar{c}\bar{d}e$
maj5_nnnnp	C	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}, \bar{d}, e) = \bar{a}\bar{b}\bar{c}+\bar{a}\bar{b}\bar{d}+\bar{a}\bar{b}e+\bar{a}\bar{c}\bar{d}+\bar{a}\bar{c}e+\bar{a}\bar{d}e+\bar{b}\bar{c}\bar{d}+\bar{b}\bar{c}e+\bar{b}\bar{d}e+\bar{c}\bar{d}e$
maj5_nnnnn_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}, \bar{d}, \bar{e}) = \bar{a}\bar{b}\bar{c}+\bar{a}\bar{b}\bar{d}+\bar{a}\bar{b}\bar{e}+\bar{a}\bar{c}\bar{d}+\bar{a}\bar{c}\bar{e}+\bar{a}\bar{d}\bar{e}+\bar{b}\bar{c}\bar{d}+\bar{b}\bar{c}\bar{e}+\bar{b}\bar{d}\bar{e}+\bar{c}\bar{d}\bar{e}$
maj5_nnnnn	C	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, \bar{c}, \bar{d}, \bar{e}) = \bar{a}\bar{b}\bar{c}+\bar{a}\bar{b}\bar{d}+\bar{a}\bar{b}\bar{e}+\bar{a}\bar{c}\bar{d}+\bar{a}\bar{c}\bar{e}+\bar{a}\bar{d}\bar{e}+\bar{b}\bar{c}\bar{d}+\bar{b}\bar{c}\bar{e}+\bar{b}\bar{d}\bar{e}+\bar{c}\bar{d}\bar{e}$
maj5_nnppn_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, c, d, \bar{e}) = \bar{a}\bar{b}c+\bar{a}\bar{b}d+\bar{a}\bar{b}\bar{e}+\bar{a}cd+\bar{a}c\bar{e}+\bar{a}d\bar{e}+\bar{b}cd+\bar{b}c\bar{e}+\bar{b}d\bar{e}+cd\bar{e}$
maj5_nnppn	C	10	70	110	$q = \text{MAJ}(\bar{a}, \bar{b}, c, d, \bar{e}) = \bar{a}\bar{b}c+\bar{a}\bar{b}d+\bar{a}\bar{b}\bar{e}+\bar{a}cd+\bar{a}c\bar{e}+\bar{a}d\bar{e}+\bar{b}cd+\bar{b}c\bar{e}+\bar{b}d\bar{e}+cd\bar{e}$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
maj5_npnpn_sub	B	10	70	110	$q = \text{MAJ}(\bar{a}, b, \bar{c}, d, \bar{e}) = \bar{a}b\bar{c}+\bar{a}bd+\bar{a}b\bar{e}+\bar{a}\bar{c}d+\bar{a}\bar{c}\bar{e}+\bar{a}d\bar{e}+b\bar{c}d+b\bar{c}\bar{e}+bd\bar{e}+\bar{c}d\bar{e}$
maj5_npnpn	C	10	70	110	$q = \text{MAJ}(\bar{a}, b, \bar{c}, d, \bar{e}) = \bar{a}b\bar{c}+\bar{a}bd+\bar{a}b\bar{e}+\bar{a}\bar{c}d+\bar{a}\bar{c}\bar{e}+\bar{a}d\bar{e}+b\bar{c}d+b\bar{c}\bar{e}+bd\bar{e}+\bar{c}d\bar{e}$
maj5_pnnpn_sub	B	10	70	110	$q = \text{MAJ}(a, \bar{b}, \bar{c}, d, \bar{e}) = a\bar{b}\bar{c}+a\bar{b}d+a\bar{b}\bar{e}+a\bar{c}d+a\bar{c}\bar{e}+ad\bar{e}+\bar{b}\bar{c}d+\bar{b}\bar{c}\bar{e}+\bar{b}d\bar{e}+\bar{c}d\bar{e}$
maj5_pnnpn	C	10	70	110	$q = \text{MAJ}(a, \bar{b}, \bar{c}, d, \bar{e}) = a\bar{b}\bar{c}+a\bar{b}d+a\bar{b}\bar{e}+a\bar{c}d+a\bar{c}\bar{e}+ad\bar{e}+\bar{b}\bar{c}d+\bar{b}\bar{c}\bar{e}+\bar{b}d\bar{e}+\bar{c}d\bar{e}$
and2_pp_sub	B	6	60	70	$q = ab$
and2_pp	C	6	60	70	$q = ab$
and2_pn_sub	B	6	60	70	$q = a\bar{b}$
and2_pn	C	6	60	70	$q = a\bar{b}$
and2_np_sub	B	6	60	70	$q = \bar{a}b$
and2_np	C	6	60	70	$q = \bar{a}b$
and2_nn_sub	B	6	60	70	$q = \bar{a}\bar{b} = \overline{a+b}$
and2_nn	C	6	60	70	$q = \bar{a}\bar{b} = \overline{a+b}$
and3_ppp_sub	B	10	60	110	$q = abc$
and3_ppp	C	10	70	110	$q = abc$
and3_ppn_sub	B	10	60	110	$q = ab\bar{c}$
and3_ppn	C	10	70	110	$q = ab\bar{c}$
and3_pnp_sub	B	10	60	110	$q = a\bar{b}c$
and3_pnp	C	10	70	110	$q = a\bar{b}c$
and3_npp_sub	B	10	60	110	$q = \bar{a}bc$
and3_npp	C	10	70	110	$q = \bar{a}bc$
and3_pnn_sub	B	10	60	110	$q = a\bar{b}\bar{c}$
and3_pnn	C	10	70	110	$q = a\bar{b}\bar{c}$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
and3_npn_sub	B	10	60	110	$q = \bar{a}b\bar{c}$
and3_npn	C	10	70	110	$q = \bar{a}b\bar{c}$
and3_nnp_sub	B	10	60	110	$q = \bar{a}\bar{b}c$
and3_nnp	C	10	70	110	$q = \bar{a}\bar{b}c$
and3_nnn_sub	B	10	60	110	$q = \bar{a}\bar{b}\bar{c} = \overline{a+b+c}$
and3_nnn	C	10	70	110	$q = \bar{a}\bar{b}\bar{c} = \overline{a+b+c}$
or2_pp_sub	B	6	60	70	$q = a + b$
or2_pp	C	6	60	70	$q = a + b$
or2_pn_sub	B	6	60	70	$q = a + \bar{b}$
or2_pn	C	6	60	70	$q = a + \bar{b}$
or2_np_sub	B	6	60	70	$q = \bar{a} + b$
or2_np	C	6	60	70	$q = \bar{a} + b$
or2_nn_sub	B	6	60	70	$q = \bar{a} + \bar{b} = \overline{ab}$
or2_nn	C	6	60	70	$q = \bar{a} + \bar{b} = \overline{ab}$
or3_ppp_sub	B	10	60	110	$q = a + b + c$
or3_ppp	C	10	70	110	$q = a + b + c$
or3_ppn_sub	B	10	60	110	$q = a + b + \bar{c}$
or3_ppn	C	10	70	110	$q = a + b + \bar{c}$
or3_pnp_sub	B	10	60	110	$q = a + \bar{b} + c$
or3_pnp	C	10	70	110	$q = a + \bar{b} + c$
or3_npp_sub	B	10	60	110	$q = \bar{a} + b + c$
or3_npp	C	10	70	110	$q = \bar{a} + b + c$
or3_pnn_sub	B	10	60	110	$q = a + \bar{b} + \bar{c}$
or3_pnn	C	10	70	110	$q = a + \bar{b} + \bar{c}$
or3_npn_sub	B	10	60	110	$q = \bar{a} + b + \bar{c}$
or3_npn	C	10	70	110	$q = \bar{a} + b + \bar{c}$
or3_nnp_sub	B	10	60	110	$q = \bar{a} + \bar{b} + c$

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
or3_nnp	C	10	70	110	$q = \bar{a} + \bar{b} + c$
or3_nnn_sub	B	10	60	110	$q = \bar{a} + \bar{b} + \bar{c} = \overline{abc}$
or3_nnn	C	10	70	110	$q = \bar{a} + \bar{b} + \bar{c} = \overline{abc}$
qfpl_sub	B	6	100	50	$q' = a(\overline{a \oplus b}) + q(a \oplus b)$
qfpl	C	6	100	50	$q' = a(\overline{a \oplus b}) + q(a \oplus b)$
ndro_qfpl	M	18	N/A	N/A	$q' = de + q\bar{e}$
ndro_fb	M	26	N/A	N/A	$q' = de + q\bar{e}$
branch2_sub	S	0	20	50	N/A
branch2_sub_i	S	0	20	50	N/A
branch2_sub_o	S	0	20	50	N/A
branch3_sub	S	0	20	70	N/A
branch3_sub_i	S	0	20	70	N/A
branch3_sub_o	S	0	20	70	N/A
branch5_sub	S	0	20	110	N/A
branch5_sub_i	S	0	30	110	N/A
branch5_sub_o	S	0	30	110	N/A
merge2_sub	S	0	20	50	N/A
merge2_sub_i	S	0	20	50	N/A
merge2_sub_o	S	0	20	50	N/A
merge3_sub	S	0	20	70	N/A
merge3_sub_i	S	0	20	70	N/A
merge3_sub_o	S	0	20	70	N/A
merge5_sub	S	0	20	110	N/A
merge5_sub_i	S	0	30	110	N/A
merge5_sub_o	S	0	30	110	N/A
storage_gate	C	2	40	30	N/A
storage_gate_sub	S	2	40	30	N/A

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
storage_gate_sub_o	S	2	40	30	N/A
pre_bsquid_sub	S	2	30	30	N/A
bfr_squid_sub	S	4	40	30	N/A
qdc_sub	I	8	110	30	N/A
qdc	I	8	110	30	N/A
aqfp2rsfq	I	4	40	30	N/A
rsfq2aqfp	I	7	80	60	N/A
tr_blank	W	0	10	10	N/A
tr_conn_M1M3	W	0	10	10	N/A
tr_conn_M1M6	W	0	10	10	N/A
tr_bias_AC_c_tl	W	0	20	10	N/A
tr_bias_AC_c_tr	W	0	20	10	N/A
tr_bias_DC_c_bl	W	0	20	10	N/A
tr_bias_DC_c_br	W	0	20	10	N/A
tr_bias_ACxDC_bl	W	0	20	10	N/A
tr_bias_ACxDC_br	W	0	20	10	N/A
tr_bias_DCxAC_tl	W	0	20	10	N/A
tr_bias_DCxAC_tr	W	0	20	10	N/A
tr_bias_xAC_tb	W	0	20	10	N/A
tr_bias_xDC_tb	W	0	20	10	N/A
tr_bias_pair_10um	W	0	20	10	N/A
tr_bias_pair_20um	W	0	20	20	N/A
tr_wire_M1_halfPTL	W	0	10	10	N/A
tr_wire_M3_halfPTL	W	0	10	10	N/A
tr_wire_M1_h	W	0	10	10	N/A
tr_wire_M1_v	W	0	10	10	N/A
tr_wire_M3_h	W	0	10	10	N/A

Continued on the next page...

Table 1.6: (Continued from previous page.)

Cell Name	Type	JJs	Height (µm)	Width (µm)	Boolean Function
tr_wire_M3_v	W	0	10	10	N/A
tr_wire_M1_c_bl	W	0	10	10	N/A
tr_wire_M1_c_br	W	0	10	10	N/A
tr_wire_M1_c_tl	W	0	10	10	N/A
tr_wire_M1_c_tr	W	0	10	10	N/A
tr_wire_M3_c_bl	W	0	10	10	N/A
tr_wire_M3_c_br	W	0	10	10	N/A
tr_wire_M3_c_tl	W	0	10	10	N/A
tr_wire_M3_c_tr	W	0	10	10	N/A
tr_wire_M1M3_c_bl	W	0	10	10	N/A
tr_wire_M1M3_c_br	W	0	10	10	N/A
tr_wire_M1M3_c_tl	W	0	10	10	N/A
tr_wire_M1M3_c_tr	W	0	10	10	N/A
tr_wire_M5_c_bl	W	0	10	10	N/A
tr_wire_M5_c_br	W	0	10	10	N/A
tr_wire_M5_c_tl	W	0	10	10	N/A
tr_wire_M5_c_tr	W	0	10	10	N/A
tr_wire_M5_T_lbr	W	0	10	10	N/A
tr_wire_M5_T_ltr	W	0	10	10	N/A
tr_wire_M5_T_tlb	W	0	10	10	N/A
tr_wire_M5_T_trb	W	0	10	10	N/A
tr_wire_M5_v	W	0	10	10	N/A
tr_wire_M5_h	W	0	10	10	N/A
tr_wire_xM5_v	W	0	10	10	N/A
tr_wire_xM5_h	W	0	10	10	N/A

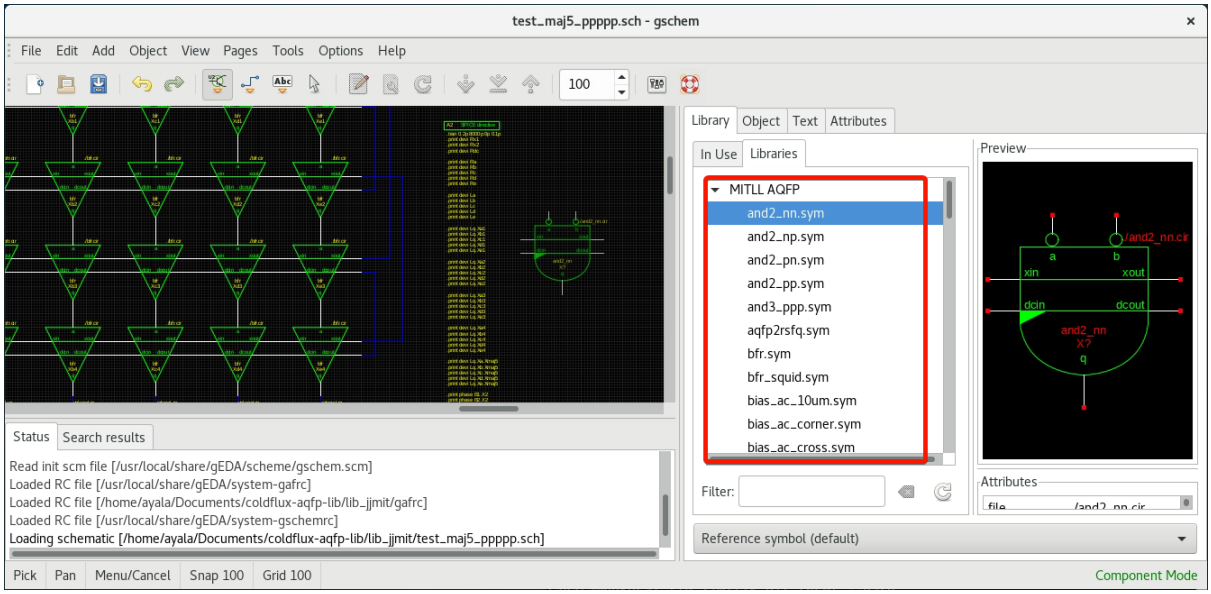


Figure 1.1: gschem interface with the AQFP library (MIT LL AQFP) loaded.

1.5 Library setup for schematic capture in gSchem

The library is built using the gEDA environment. It can be loaded into the schematic capture tool `gschem` so that a circuit designer may instantiate AQFP logic cells into their own schematic design. This can be done through a `gafrc` initialization file in the working/design directory that `gschem` is invoked in.

1.5.1 Recommended: Working inside `lib_jjmit`

The easiest way to make sure all your designs netlist correctly is by working in the same folder as `lib_jjmit`. The folder already includes an appropriate `gafrc` file so the AQFP library symbols will load as seen in Figure 1.1.

1.5.2 Using your own workspace

This approach is not fully tested. We tried alternative ways to prepare the cells so that either the sub-circuit netlists or source schematics can be linked to the symbols regardless of where the files are located, but we ran into netlisting issues. The following is one way to get your schematics to netlist when working in a directory other than `lib_jjmit`.

1. Create a workspace directory for your designs. This can simply be a new folder in any location that you have full access permissions to. For example: `/home/user/gschem-designs`

2. Copy `lib_jjmit` to any accessible location.

3. In your workspace directory, create a file called `gafrc` whose contents are:

```
(component-library "/path/to/lib_jjmit" "MITLL AQFP")
(source-library "/path/to/lib_jjmit")
```

Where `/path/to/lib_jjmit` is the location where you copied `lib_jjmit`.

4. When you invoke `gschem` in your workspace directory, the AQFP library should be loaded under `MITLL AQFP`.

5. When you instantiate an AQFP cell, be sure to prepend the `file` attribute of each instance with your chosen `/path/to/lib_jjmit`. By default, the `file` attribute is simply `./cell-name.cir`. They should be changed to `/path/to/lib_jjmit/cell-name.cir` in order for `gnetlist` to properly find the sub-circuit netlists.

1.6 Simulation examples

Along with the logic cells themselves, we included several simulation examples for analog and digital simulation. For analog simulation, you will find several `test_*.cir` netlist file. These netlist files can be simulated with the following command:

```
$ josim test_{cell name}.cir -o {path/to/results/results.csv}
```

The above will run JoSIM on the desired `test_*.cir` file and store the simulation results as a CSV file in your desired location. The CSV file can be plotted using any CSV-compatible plotting tool. A python plotting script is available on the JoSIM website [12].

For digital simulation, the `verilog/_celltestbench` contains a complete set of cell testbenches to perform digital simulation. They can be run as follows:

```
$ iverilog *.v -o {cell name} //compile .v source file
$ ./{cell name} or vvp {cell name} // run simulation
$ gtkwave tb_{cell name}_example.vcd & // open the simulation result in waveform viewer
```

1.7 Useful commands and helper scripts

1.7.1 Useful commands

In this section, we list a few useful commands assuming you have the following environment:

- CentOS 7
- Python 3.8
- gEDA environment, `http://wiki.geda-project.org/geda:download`
- InductEx
- InductEx-LVS
- KLayout, `https://www.klayout.de/build.html`

Opening a schematic or symbol

gEDA's gschem tool is primarily used for schematic and symbol creation. You can use `gschem` to open the schematic files (`.sch`) or symbol files (`.sym`).

```
$ gschem {desired-schematic-or-symbol}.{sch-or-sym}
```

Opening a layout for viewing or editing

KLayout is the recommended tool for opening the GDS files. You can open them just for viewing via:

```
$ klayout {desired-cell-to-view}.gds
```

To open the GDS file for editing, you need to include `-e` as shown below:

```
$ klayout -e {desired-cell-to-view}.gds
```

Netlisting a schematic

`gschem` schematics are netlisted using gEDA's `gnetlist` utility with the `spice-sdb` format:

```
$ gnetlist -g spice-sdb -o {desired-netlist-filename}.cir {schematic-to-netlist}.sch
```

Note that AQFP cells are generally built hierarchically meaning internal sub-circuits may also contain sub-circuits within. At this moment, JoSIM cannot parse multiple levels of sub-circuit definitions in the netlist, but a workaround is to define all sub-circuits at the top-level. See Section 1.7.2 for `lib_jjmit/auto-netlist.sh` that will automatically netlist the desired schematic and then "flatten" the resulting netlist for JoSIM compatibility.

Simulating a netlist

```
$ josim -o {desired-output-filename}.csv {netlist-to-simulate}.cir
```

Note that the netlists of the logic cells by themselves cannot be simulated as they are simply sub-circuits. A fully complete netlist with appropriate input sources, bias sources, output loads, simulation directives, and printing directives are needed. Our included collection of `test_*.sch` files and their corresponding `test_*.cir` netlists are all ready for simulation as explained in Section 1.6.

Running LVS

```
$ inductex-lvs {desired-cell}.json
```

Note that our `.json` files are configured to use `tech/mit_sfq5ee_v1.7.aqfp.tech` technology file and to store results into the `lvs` subdirectory.

Extracting inductances

```
$ inductex {desired-cell}.gds -n {desired-cell}.ix.cir -l tech/mitll_sfq5ee.ldf
```

The `tech` subdirectory has a number of LDFs for InductEx. Typically `tech/mitll_sfq5ee.ldf` is used for the AQFP library, but interface cells such as `aqfp2rsfq` may require `tech/mitll_sfq5ee_res.ldf`. The InductEx netlists support the back annotation feature from v3.0 of the AQFP library.

1.7.2 Helper scripts

Several helper scripts are bundled primarily to help maintain the cell library. But others may find the scripts useful too.

lib_jjmit/flatcir.py

Processes `*.cir` netlists so that all sub-circuits including nested ones are moved to the top-level of the netlist. Duplicate sub-circuits are removed and netlist comments are stripped. Original netlist will be overwritten but a backup of it will be saved as `*.gnetlist.cir`. The script is expected to work with netlists generated by `gnetlist -g spice-sdb`.

```
$ ./flatcir.py {desired-netlist-to-flatten}.cir
```

lib_jjmit/josim-waveforms/simres_formatter.py

Formats JSIM/JoSIM simulation results into Cadence VCSV (Virutoso Visualization & Analysis, ViVA format) or normal CSV format. By default (no optional arguments), it will convert JSIM results to VCSV format. In the context of the SuperTools project, we primarily use this script to convert the JoSIM results to VCSV format so that we can carefully inspect the waveforms in ViVA. Note: this script is not necessary if you use an alternative plotting tool that can read CSV files.

```
$ ./simres_formatter.py -p josim {josim-result}.csv
```

The above command will parse a JoSIM output result (`.csv` or `.dat`) and generate a VCSV file named `josim-result.csv.vcsv`. This VCSV file can be opened in Cadence Virtuoso Visualization & Analysis (ViVA) tool. More options can be found when running the script with `--help`.

lib_jjmit/auto-netlist.sh

This is a bash script that can accept multiple `*.sch` schematic files (wildcard globbing also possible) for netlisting. It will generate the normal netlist and then flatten it with `flatcir.py`.

```
$ ./auto-netlist.sh {file-1}.sch {file-2}.sch ... {file-n}.sch
```

lib_jjmit/auto-sim.sh

This is a bash script that can accept multiple `*.cir` netlist files (wildcard globbing also possible) for JoSIM simulation. It will run JoSIM sequentially for each netlist, save each result into the `josim-waveforms` subdirectory, and also generate Cadence VCSV format of each result.

```
$ ./auto-sim.sh {file-1}.cir {file-2}.cir ... {file-n}.cir
```

lib_jjmit/auto-netsim.sh

This is a bash script that can accept multiple `*.sch` schematic files (wildcard globbing also possible) for netlisting and immediate JoSIM simulation. It effectively combines the actions of `auto-netlist.sh` and `auto-sim.sh` into one script.

```
$ ./auto-netsim.sh {file-1}.sch {file-2}.sch ... {file-n}.sch
```

lib_jjmit/auto-lvs.sh

This is a bash script that can accept multiple `*.json` LVS files (wildcard globbing also possible) for running LVS via inductex-lvs.

```
$ ./auto-lvs.sh {file-1}.json {file-2}.json ... {file-n}.json
```

2. AQFP Cell Library

This section presents the schematic, netlist, physical layout, digital model, analog simulation, digital simulation, and switching energy of ***representative*** AQFP cells. Only main cells (with PTL connections) are discussed. The schematic, symbol, base netlist and switching energy are the same for both the subcell and main cell versions of each cell. We do not fully describe all variations because it is rather redundant. Instead, we show representative majority, AND, and OR cells. In the respective text, we list the variations of those cells that are also available. Table 1.6 is a complete listing of all of our AQFP cells.

2.1 Combinational Cells

2.1.1 BFR

The buffer (`bfr`) is the basic logic structure of AQFP circuits consisting of a double-Josephson-junction SQUID. The library includes the following variations of the buffer cell: `bfr_sub`, `bfr_sub_i`, `bfr_sub_o` and `bfr`. The symbol, schematic, digital model and switching energy is the same for all variations.

Symbol

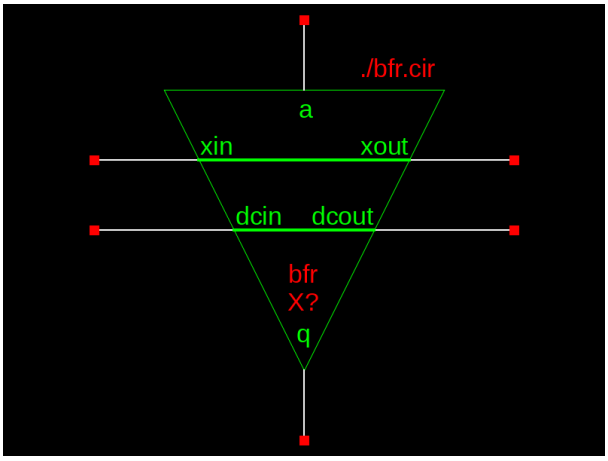


Figure 2.1: `bfr` symbol.

Table 2.1: `bfr` pin list.

Pin	Description
A	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

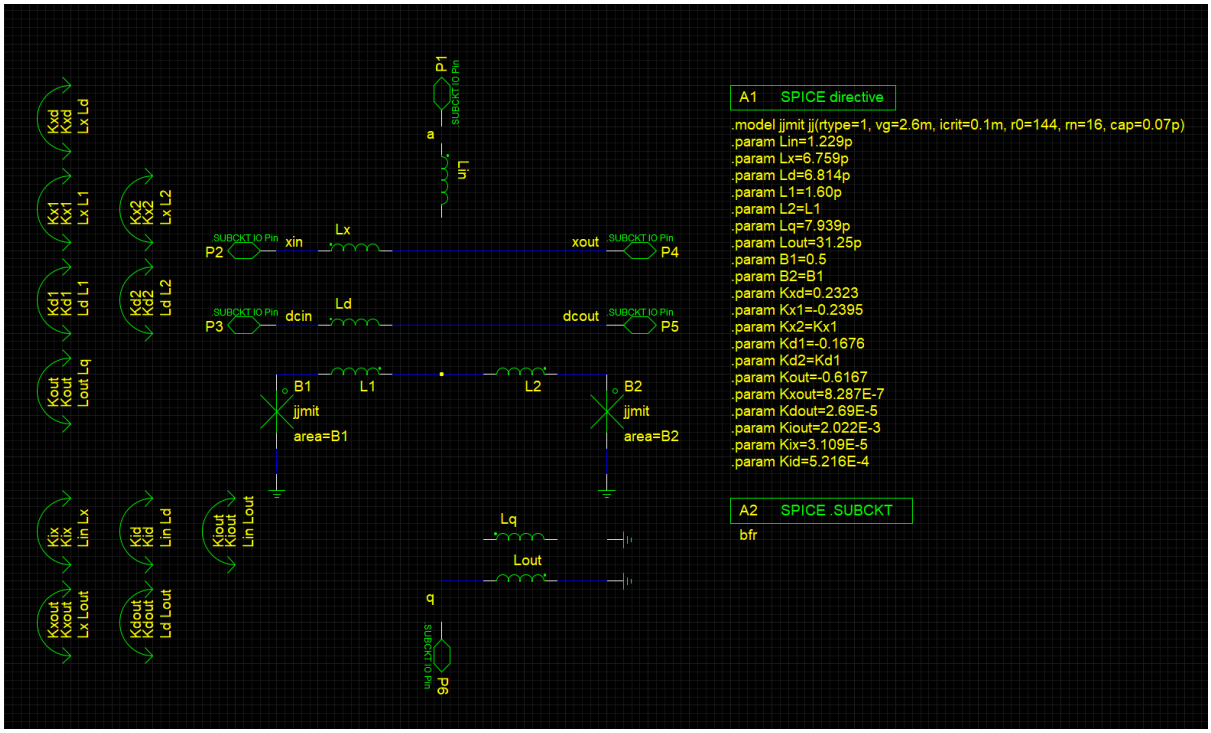


Figure 2.2: `bfr` schematic.

Layout

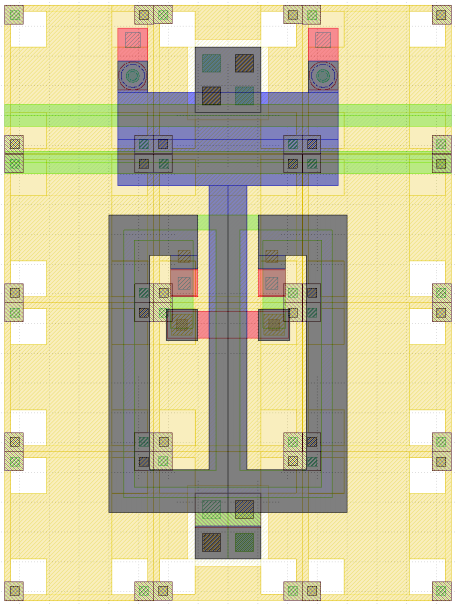


Figure 2.3: `bfr` layout.

Analog model

```
.SUBCKT bfr a xin dcin xout dcout q
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lx=7.4p
.param Ld=7.45p
.param Lin=1.23p
.param L1=1.59p
.param L2=L1
.param Lq=7.92p
.param Lout=31.2p
.param B1=0.5
.param B2=B1
.param Kxd=0.2322
.param Kx1=-0.186
.param Kx2=Kx1
.param Kd1=-0.133
.param Kd2=Kd1
.param Kout=-0.495
.param Kxout=3.68E-5
.param Kdout=3.27E-5
.param Kiout=2.201E-3
.param Kix=-1.145E-4
.param Kid=-1.063E-5
B1 1 0 jjmit area=B1
B2 3 0 jjmit area=B2
Lx xin xout Lx
Ld dcin dcout Ld
Lin a 2 Lin
L1 2 1 L1
L2 3 2 L2
Lq 2 0 Lq
Lout 0 q Lout
Kxd Lx Ld Kxd
Kx1 Lx L1 Kx1
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
```

```
36 Kd2 Ld L2 Kd2
37 Kid Lin Ld Kid
38 Kiout Lin Lout Kiout
39 Kix Lin Lx Kix
40 Kxout Lx Lout Kxout
41 Kdout Ld Lout Kdout
42 Kout Lout Lq Kout
43 .ends bfr
```

Listing 2.1: `bfr` netlist (.cir).

Simulation result

Simulation waveform of an BUFFER (bfr) DUT (design under test) with 4-stage buffer before the input (a). Another 3-stage buffer is placed after the inverter's q output. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (input of the first buffer) as 010101010111, AC source 1 (generates phase 1 and 3), AC source 2 (generates phase 2 and 4), and the output of the final out of the buffer chain as 10010101010111 with an random initial outputs 10. This is because of the meander structure of AQFP circuits. In an 8-stage AQFP circuit, it takes two clock cycles to propagate the data to the output. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

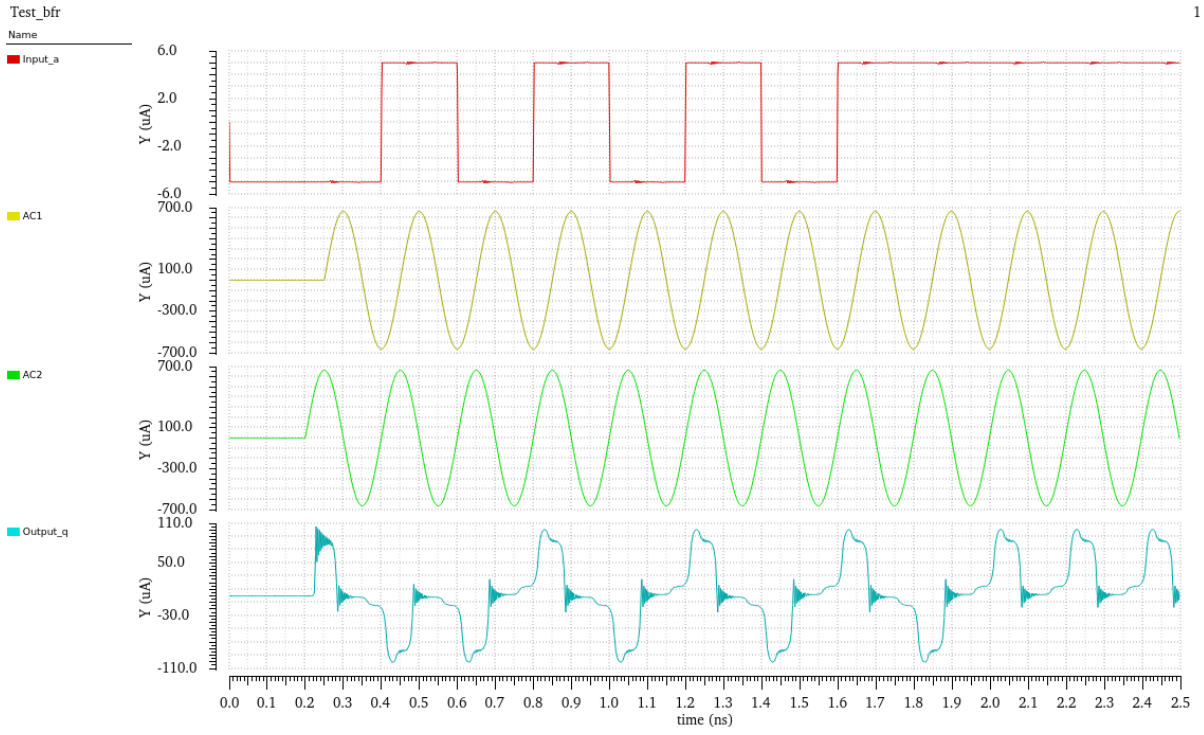


Figure 2.4: `bfr` analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2  module bfr(a, q, xin, xout, dcin, dcout);
3    input a;
4    output q;
5    inout xin, xout, dcin, dcout;
6    reg q;
7    parameter pul_wid = 100;
8    wire not_a;
9
10   assign not_a = !a;
11
12   biasDir_b I0(xin, xout, dcin, dcout, gatex);
13
14   initial begin
15
16   $timeformat(-12, 1, "␣ps", 8); // time format
17
18   // output regsiter initialization
19   q = 1'bz;
20   end // initialization
21
22   specify
23     specparam d_clk   = 5;
24     specparam clk_d  = 50;
25
26     $setup(posedge a &&& a, posedge gatex, d_clk);
27     $setup(negedge a &&& not_a, posedge gatex, d_clk);
28     $hold(posedge gatex, negedge a &&& a, clk_d);
29     $hold(posedge gatex, posedge a &&& not_a, clk_d);
30   endspecify
31
32   always @(posedge gatex)
33     begin
34       if (a == 1 | a == 0)
35         begin
36           q <= a;
37           q <= #pul_wid 1'bz;
38         end
39       else
40         begin
41           q <= 1'bx;
42           q <= #pul_wid 1'bz;
43         end
44     end
45
46 endmodule
```

Listing 2.2: `bfr` Verilog model code.

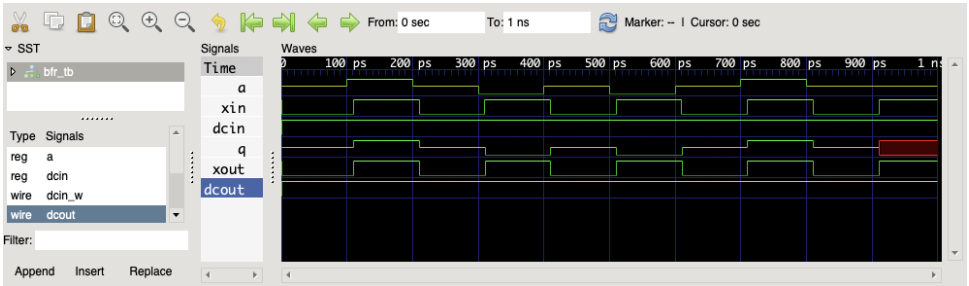


Figure 2.5: `bfr` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($a = 0$ and $a = 1$) and clock frequencies (from 0.1GHz to 10GHz).

Table 2.2: `bfr` switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'0' (J)	1.14E-23	2.32E-23	5.86E-23	1.18E-22	2.38E-22	7.18E-22	2.40E-21
'1' (J)	1.14E-23	2.32E-23	5.86E-23	1.18E-22	2.38E-22	7.18E-22	2.40E-21

2.1.2 INV

The AQFP inverter (`inv`) is based on the buffer cell. The operational principle is to generate the opposite direction of the output current by changing the sign of the coupling coefficient of the output transformer (e.g. *Kout* of the schematic in 2.7). The library includes the following variations of the buffer cell: `inv_sub`, `inv_sub_i`, `inv_sub_o` and `inv`. The symbol, schematic, digital model and switching energy is the same for all variations.

Symbol

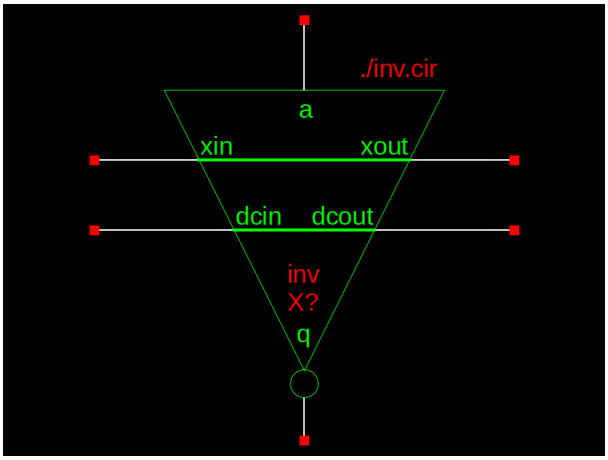


Figure 2.6: `inv` symbol.

Table 2.3: `inv` pin list.

Pin	Description
A	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

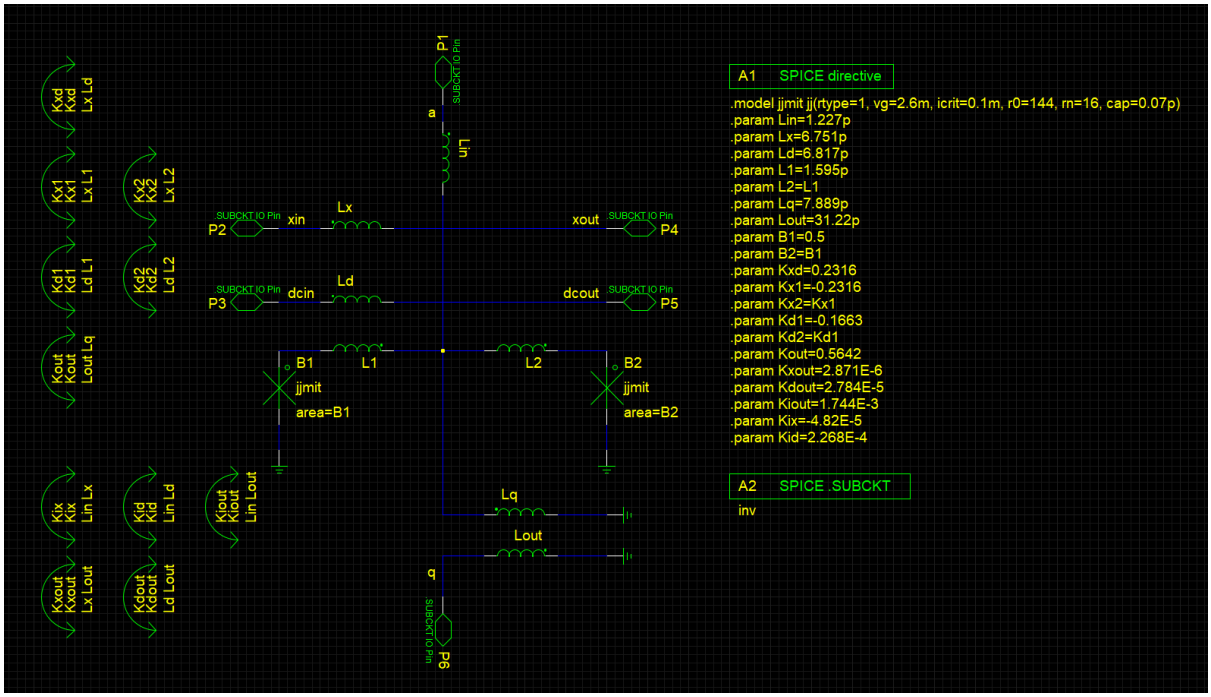


Figure 2.7: `inv` schematic.

Layout

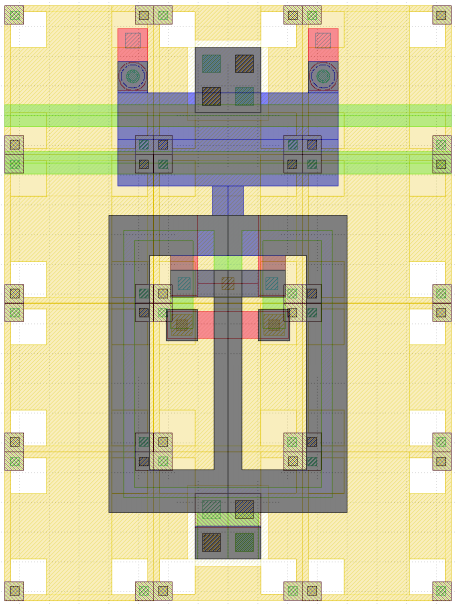


Figure 2.8: `inv` layout.

Analog model

```
.SUBCKT inv a xin dcin xout dcout q
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.23p
.param Lx=7.4p
.param Ld=7.45p
.param L1=1.59p
.param L2=L1
.param Lq=7.92p
.param Lout=31.2p
.param B1=0.5
.param B2=B1
.param Kxd=0.2322
.param Kx1=-0.186
.param Kx2=Kx1
.param Kd1=-0.133
.param Kd2=Kd1
.param Kout=0.495
.param Kxout=3.68E-5
.param Kdout=3.27E-5
.param Kiout=2.201E-3
.param Kix=-1.145E-4
.param Kid=-1.063E-5
B1 1 0 jjmit area=B1
B2 3 0 jjmit area=B2
Lin a 2 Lin
L1 2 1 L1
L2 3 2 L2
Lq 2 0 Lq
Lout 0 q Lout
Lx xin xout Lx
Ld dcin dcout Ld
Kid Lin Ld Kid
Kiout Lin Lout Kiout
Kix Lin Lx Kix
Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
.ends inv
```

Listing 2.3: `inv` netlist (.cir).

Simulation result

Simulation waveform of an INVERTER (inv) DUT (design under test) with 4-stage buffer before the input (a). Another 3-stage buffer is placed after the inverter's q output. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input a (input of the first buffer) as 01010101, AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the output (q) generated from the final stage of buffer as 1110101010 with two random initial outputs 11. This is because of the meander structure of AQFP circuits. In an 8-stage AQFP circuit, it takes two clock cycles to propagate the data to the output. Input peak-to-peak amplitude is $\pm 5\,\mu A$, AC amplitude is $658\,\mu A$, and DC is set to $843\,\mu A$.

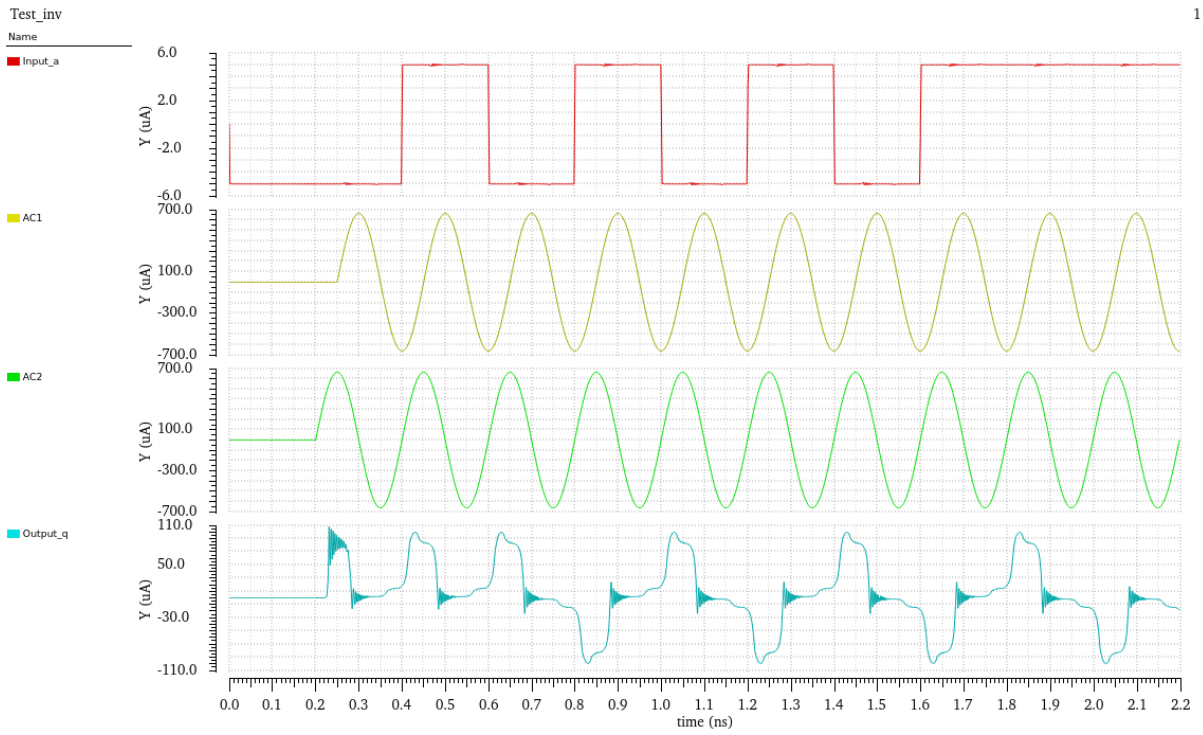


Figure 2.9: inv analog waveform.

Digital model

```
1
2  `timescale 1ps/10fs
3  module inv(a, q, xin, xout, dcin, dcout);
4    input a;
5    output q;
6    inout xin, xout, dcin, dcout;
7    reg q;
8    parameter pul_wid = 100;
9    wire not_a;
10
11   assign not_a = !a;
12
13   biasDir_b I0(xin, xout, dcin, dcout, gatex);
14
15   initial begin
16
17   $timeformat(-12, 1, "␣ps", 8); // time format
18
19   // output regsiter initialization
20   q = 1'bz;
21   end // initialization
22
23   specify
24     specparam d_clk   = 5;
25     specparam clk_d  = 50;
26
27     $setup(posedge a &&& a, posedge gatex, d_clk);
28     $setup(negedge a &&& not_a, posedge gatex, d_clk);
29     $hold(posedge gatex, negedge a &&& a, clk_d);
30     $hold(posedge gatex, posedge a &&& not_a, clk_d);
31   endspecify
32
33   always @(posedge gatex)
34     begin
```

```
35        if (a == 1 | a == 0)
36          begin
37            q <= ~a;
38            q <= #pul_wid 1'bz;
39          end
40        else
41          begin
42            q <= 1'bx;
43            q <= #pul_wid 1'bz;
44          end
45      end
46
47  endmodule
```

Listing 2.4: `inv` Verilog model code.

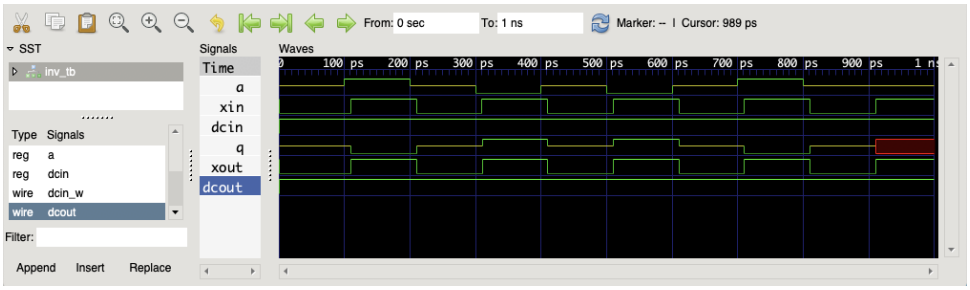


Figure 2.10: `inv` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($a = 0$ and $a = 1$) and clock frequencies.

Table 2.4: `inv` switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'0' (J)	1.78E-23	3.57E-23	8.99E-23	1.85E-22	4.50E-22	1.78E-21	5.18E-21
'1' (J)	1.78E-23	3.57E-23	8.97E-23	1.84E-22	4.45E-22	1.75E-21	5.06E-21

2.1.3 AND2

The 2-input AQFP AND cell is built from buffer and constant 0 cells. The operational principle is to merge the output of two buffers and a constant 0 cell through a 3-to-1 branch. There are 4 types of AND2 gate with different inputs (positive and negative): and2_pp, and2_pn, and2_np, and2_nn. Each type of AND2 gate has both a subcell and main cell version. Here we present the AND gate main cell with two positive inputs named and2_pp.

Symbol

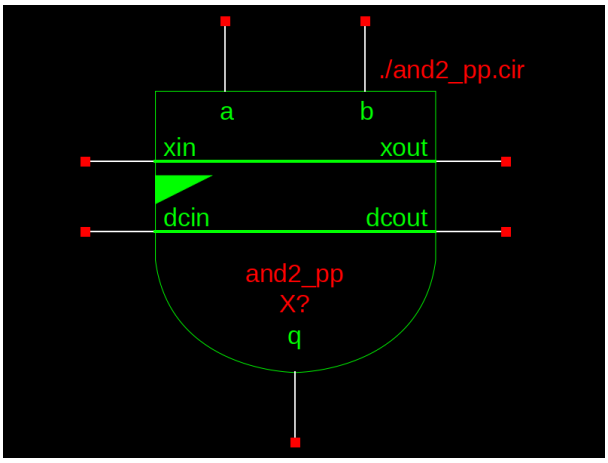


Figure 2.11: and2_pp symbol.

Table 2.5: and2_pp pin list.

Pin	Description
A	data input
B	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

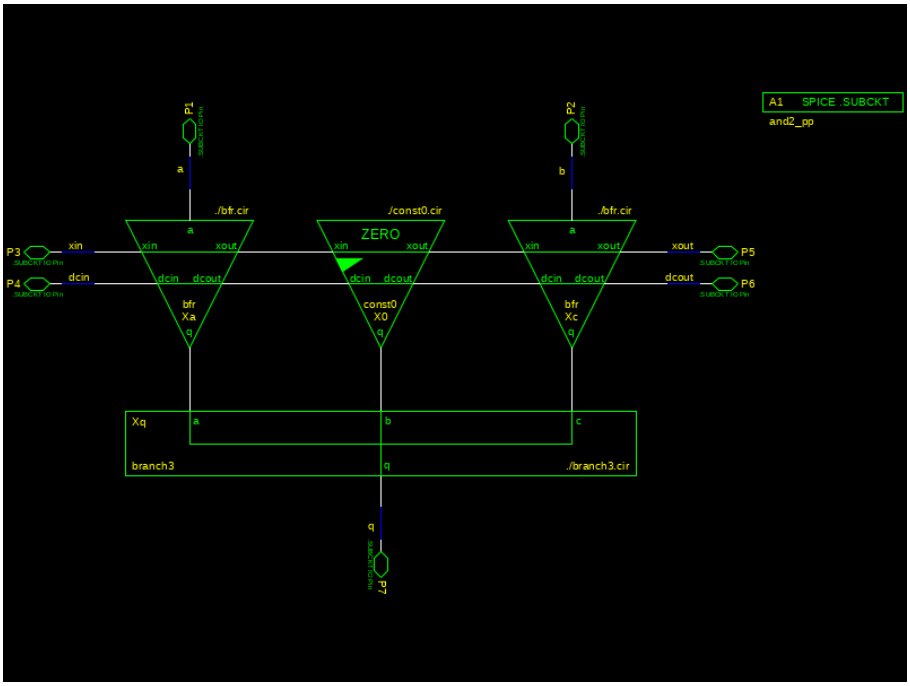


Figure 2.12: and2_pp schematic.

Layout

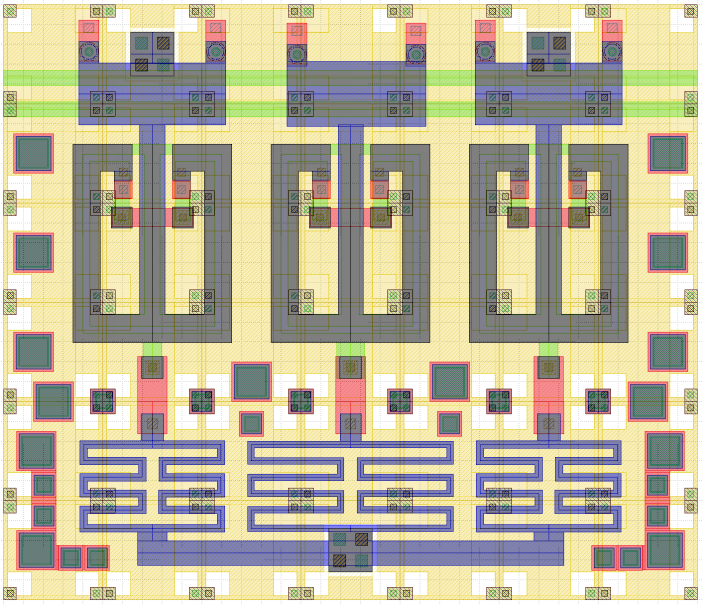


Figure 2.13: and2_pp layout.

Analog model

```
1  .include bfr_sub_i.cir
2  .include const0_sub.cir
3  .include merge3_sub_o.cir
4  .include bias_pair_05um.cir
5  .SUBCKT AND2_pp a b xin dcin xout dcout q
6  Xbias_in xin dcin 1 2 bias_pair_05um
7  Xa a 1 2 3 4 5 bfr_sub_i
8  Xb b 3 4 6 7 8 bfr_sub_i
9  Xc 6 7 9 10 11 const0_sub
10 Xq 5 8 11 q merge3_sub_o
11 Xbias_out 9 10 xout dcout bias_pair_05um
12 .ends AND2_pp
```

Listing 2.5: and2_pp netlist (.cir).

Simulation result

Simulation waveform of an AND (and2_pp) DUT (design under test) with two 4-stage buffers before the input 'a' and 'b'. Another 3-stage buffer is placed after the AND's output 'q'. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (a) as 01010101 and (b) as 00110011 (inputs of the first stage buffers), AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the output generated from the final stage of buffer (q) as 1100010001 with two random initial outputs 11. This is because of the meander structure of AQFP circuits. In an 8-stage AQFP circuit, it takes two clock cycles to propagate the data to the output. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

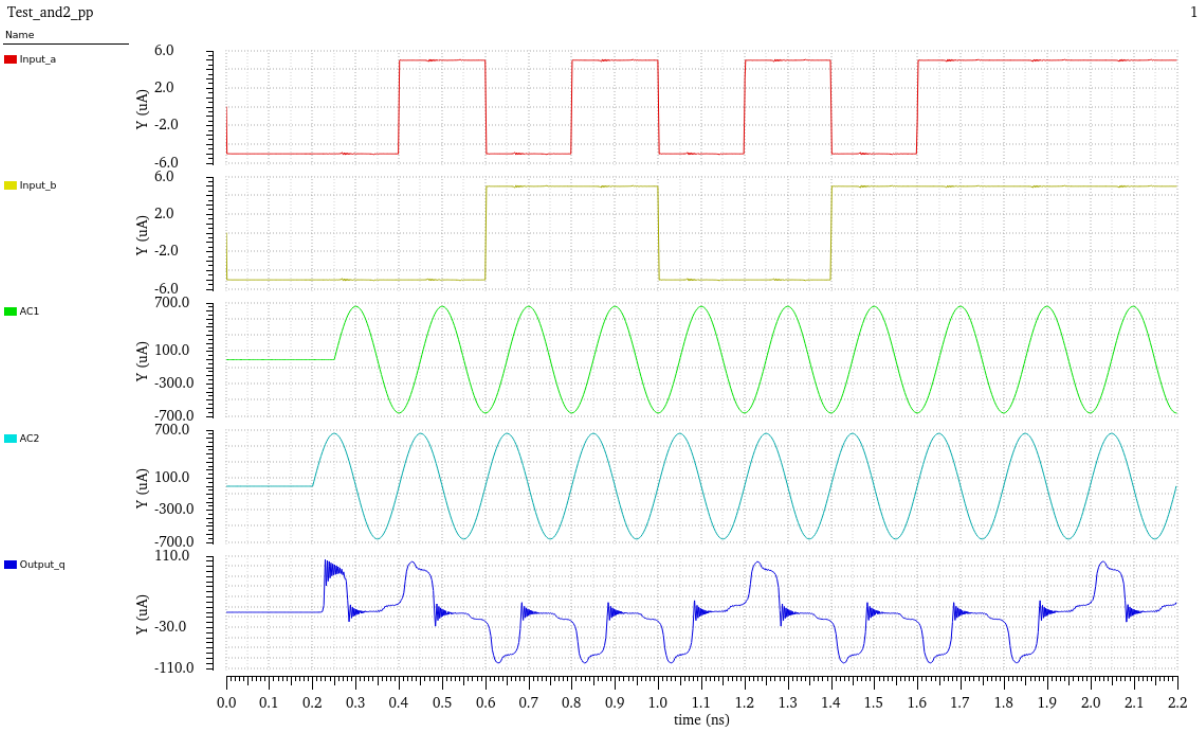


Figure 2.14: and2_pp analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2  module and2_pp(a, b, q, xin, xout, dcin, dcout);
3    input a, b;
4    output q;
5    inout xin, xout, dcin, dcout;
6    reg q;
7    parameter pul_wid = 100;
8    wire not_a, not_b;
9
10   assign not_a = !a;
11   assign not_b = !b;
12
13
14   biasDir_b I0(xin, xout, dcin, dcout, gatex);
15
16   initial begin
17
18   $timeformat(-12, 1, "␣ps", 8); // time format
19
20   // output regsiter initialization
21   q = 1'bz;
22   end // initialization
23
24   specify
25     specparam d_clk   = 5;
26     specparam clk_d  = 50;
27
28     $setup(posedge a &&& a, posedge gatex, d_clk);
29     $setup(negedge a &&& not_a, posedge gatex, d_clk);
30     $hold(posedge gatex, negedge a &&& a, clk_d);
31     $hold(posedge gatex, posedge a &&& not_a, clk_d);
32
33     $setup(posedge b &&& b, posedge gatex, d_clk);
34     $setup(negedge b &&& not_b, posedge gatex, d_clk);
35     $hold(posedge gatex, negedge b &&& b, clk_d);
36     $hold(posedge gatex, posedge b &&& not_b, clk_d);
37   endspecify
38
39   always @(posedge gatex)
40     begin
41       if ((a!==1'bz) & (b!==1'bz) & (a!==1'bx) & (b!==1'bx))
42         begin
43           q <= a & b;
44           q <= #pul_wid 1'bz;
45         end
46       else
47         begin
48           q <= 1'bx;
49           q <= #pul_wid 1'bz;
50         end
51     end
52
53 endmodule
```

Listing 2.6: and2_pp Verilog model code.

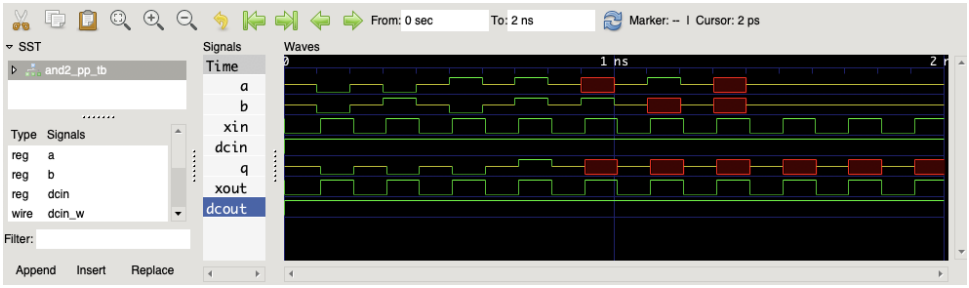


Figure 2.15: and2_pp digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($ab = 00$, $ab = 01$, $ab = 10$ and $ab = 11$) and clock frequencies.

Table 2.6: and2_pp switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'00' (J)	2.85E-23	6.31E-23	1.69E-22	3.68E-22	9.28E-22	3.93E-21	1.31E-20
'01' (J)	5.89E-21	5.76E-21	5.38E-21	4.95E-21	5.45E-21	1.07E-20	1.70E-20
'10' (J)	5.92E-21	5.79E-21	5.41E-21	4.98E-21	5.47E-21	1.07E-20	1.71E-20
'11' (J)	1.84E-21	1.67E-21	1.65E-21	2.24E-21	3.51E-21	7.10E-21	1.38E-20

2.1.4 AND3

AQFP AND cells are built from buffer cells and constant cells. The operational principle of a 3-input AND gate is to merge the output of three buffers and 2 constant 0 cells through a 5-to-1 branch. There are 8 types of 3-input AND gate with different input combinations (positive and negative): and3_ppp, and3_ppn, and3_pnp, and3_pnn, and3_npp, and3_npn, and3_nnp, and3_nnn. Each type of AND3 gate has both a subcell and main cell version. Here we only present the 3-input AND gate main cell with three positive inputs named and3_ppp.

Symbol

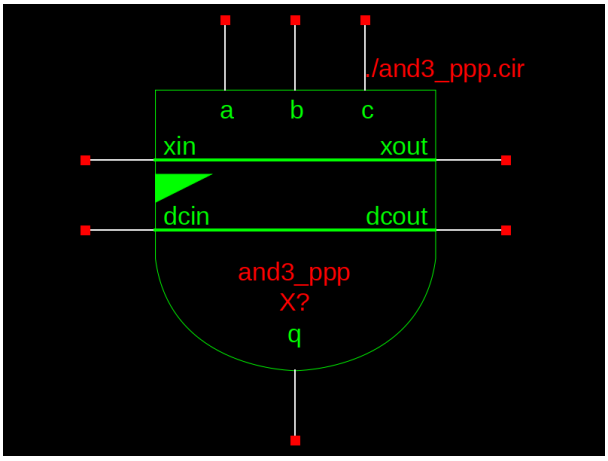


Figure 2.16: and3_ppp symbol.

Table 2.7: and3_ppp pin list.

Pin	Description
A	data input
B	data input
C	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

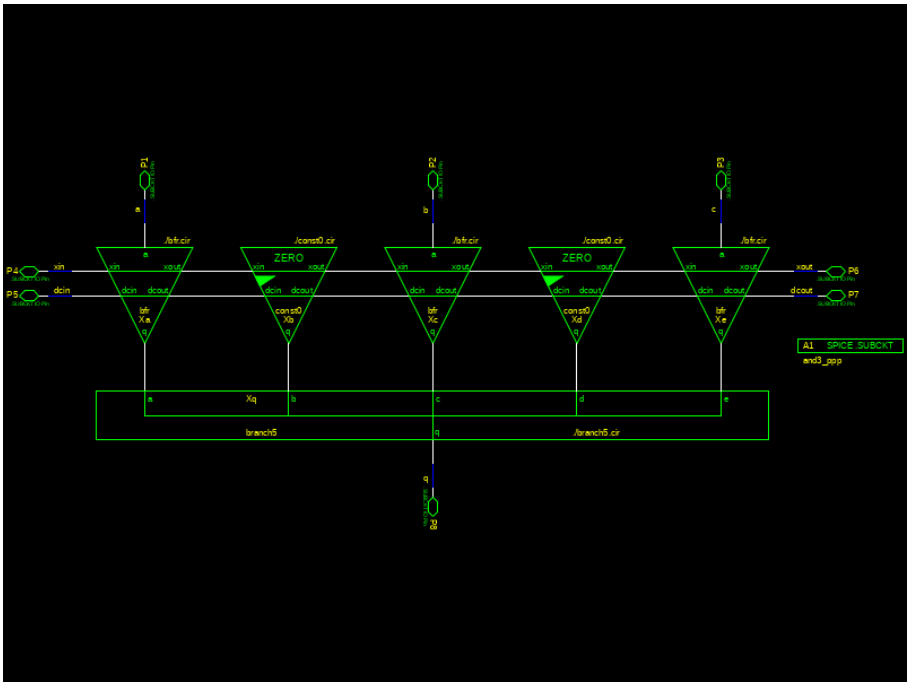


Figure 2.17: and3_ppp schematic.

Layout

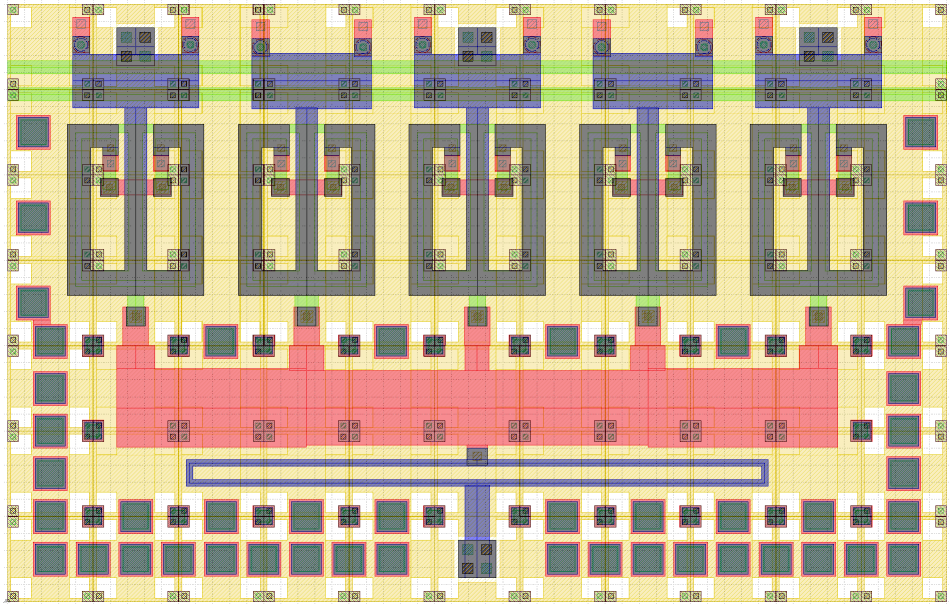


Figure 2.18: and3_ppp layout.

Analog model

```
.include bfr_sub_i.cir
.include const0_sub.cir
.include bias_pair_05um.cir
.include merge5_sub_o.cir
.SUBCKT AND3_ppp a b c xin dcin xout dcout q
Xbias_in xin dcin 1 2 bias_pair_05um
Xa a 1 2 3 4 5 bfr_sub_i
Xconst0_1 3 4 6 7 8 const0_sub
Xb b 6 7 9 10 11 bfr_sub_i
Xconst0_2 9 10 12 13 14 const0_sub
Xc c 12 13 15 16 17 bfr_sub_i
Xq 5 8 11 14 17 q merge5_sub_o
Xbias_out 15 16 xout dcout bias_pair_05um
.ends AND3_ppp
```

Listing 2.7: and3_ppp netlist (.cir).

Simulation result

Simulation waveform of a 3-input AND (and3_ppp) DUT (design under test) with three 4-stage buffers before the input 'a', 'b' and 'c'. Another 3-stage buffer is placed after the AND's output 'q'. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (a) as 01010101, (b) as 00110011, (c) as 00001111 (inputs of the first stage buffers), AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the output generated from the final stage of buffer (q) as 1100000001 with two random initial outputs 11. This is because of the meander structure of AQFP circuits. In an 8-stage AQFP circuit, it takes two clock cycles to propagate the data to the output. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

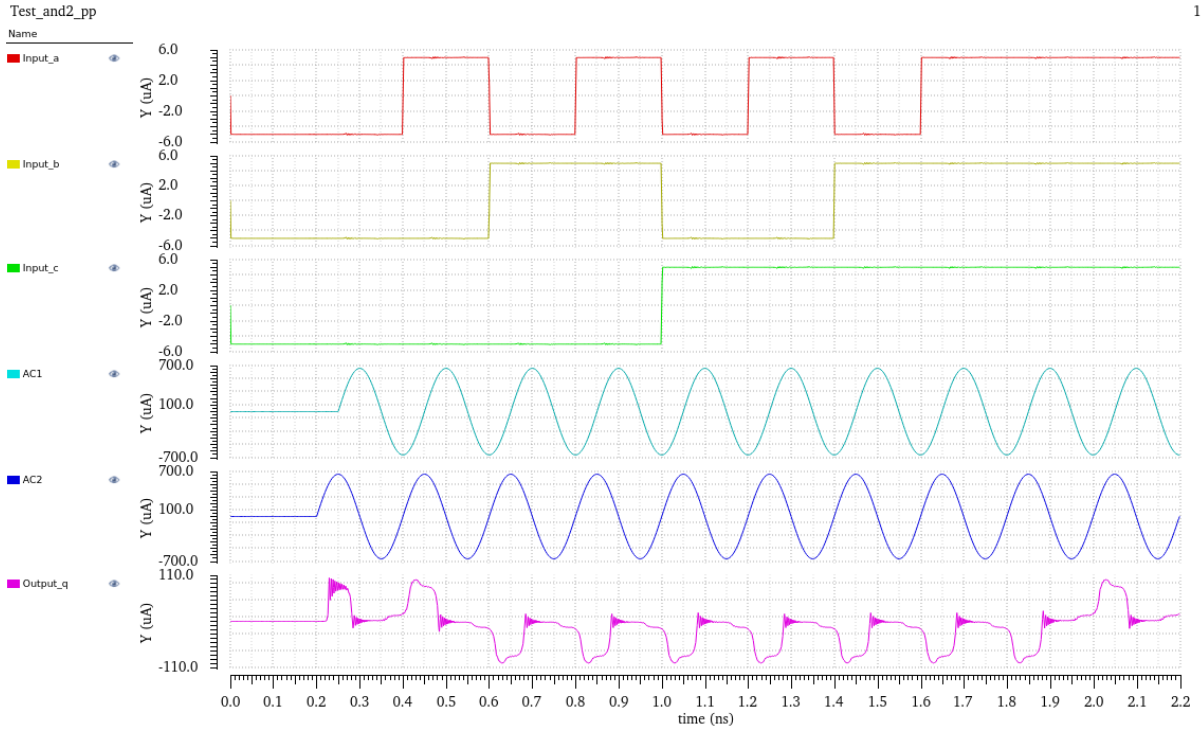


Figure 2.19: and3_ppp analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2  module and3_ppp(a, b, c, q, xin, xout, dcin, dcout);
3    input a, b, c;
4    output q;
5    inout xin, xout, dcin, dcout;
6    reg q;
7    parameter pul_wid = 100;
8    wire [2:0] in_and3, neg_in_and3;
9
10   assign in_and3 = {a, b, c};
11   assign neg_in_and3 = ~in_and3;
12
13   biasDir_b I0(xin, xout, dcin, dcout, gatex);
14
15   initial begin
16
17   $timeformat(-12, 1, "␣ps", 8); // time format
18
19   // output regsiter initialization
20   q = 1'bz;
21   end // initialization
22
23   specify
24     specparam d_clk   = 5;
25     specparam clk_d  = 50;
26
27     $setup(posedge in_and3[2] &&& in_and3[2], posedge gatex, d_clk);
28     $setup(negedge in_and3[2] &&& neg_in_and3[2], posedge gatex, d_clk);
29     $hold(posedge gatex, negedge in_and3[2] &&& in_and3[2], clk_d);
30     $hold(posedge gatex, posedge in_and3[2] &&& neg_in_and3[2], clk_d);
31
32     $setup(posedge in_and3[1] &&& in_and3[1], posedge gatex, d_clk);
33     $setup(negedge in_and3[1] &&& neg_in_and3[1], posedge gatex, d_clk);
34     $hold(posedge gatex, negedge in_and3[1] &&& in_and3[1], clk_d);
35     $hold(posedge gatex, posedge in_and3[1] &&& neg_in_and3[1], clk_d);
36
37     $setup(posedge in_and3[0] &&& in_and3[0], posedge gatex, d_clk);
38     $setup(negedge in_and3[0] &&& neg_in_and3[0], posedge gatex, d_clk);
39     $hold(posedge gatex, negedge in_and3[0] &&& in_and3[0], clk_d);
40     $hold(posedge gatex, posedge in_and3[0] &&& neg_in_and3[0], clk_d);
41
42   endspecify
43
44   always @(posedge gatex)
45     begin
46       if ((&in_and3 !== 1'bx) & (&in_and3 !== 1'bz))
47         begin
48           q <= &in_and3;
49           q <= #pul_wid 1'bz;
50         end
51       else
52         begin
53           q <= 1'bx;
54           q <= #pul_wid 1'bz;
55         end
56     end
57 endmodule
```

Listing 2.8: and3_ppp Verilog model code.

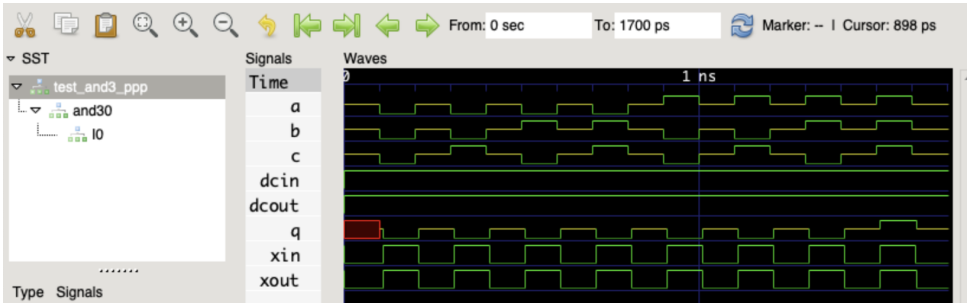


Figure 2.20: and3_ppp digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($abc = 000$, $abc = 001$, $abc = 010$, $abc = 011$, $abc = 100$, $abc = 101$, $abc = 110$ and $abc = 111$) and clock frequencies (0.1GHz to 10GHz).

Table 2.8: and3_ppp switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'000' (J)	4.98E-23	1.05E-22	2.73E-22	5.77E-22	1.46E-21	6.40E-21	1.83E-20
'001' (J)	8.15E-21	8.06E-21	7.81E-21	7.39E-21	7.54E-21	1.37E-20	2.84E-20
'010' (J)	8.63E-21	8.55E-21	8.31E-21	7.89E-21	7.70E-21	1.36E-20	2.87E-20
'011' (J)	1.34E-20	1.31E-20	1.24E-20	1.13E-20	1.10E-20	2.15E-20	3.64E-20
'100' (J)	8.15E-21	8.06E-21	7.81E-21	7.39E-21	7.54E-21	1.37E-20	2.84E-20
'101' (J)	1.32E-20	1.29E-20	1.22E-20	1.11E-20	1.08E-20	2.15E-20	3.61E-20
'110' (J)	1.34E-20	1.31E-20	1.24E-20	1.13E-20	1.10E-20	2.15E-20	3.64E-20
'111' (J)	1.34E-20	1.20E-21	1.31E-21	2.35E-21	4.21E-21	1.27E-20	2.88E-20

2.1.5 OR2

The 2-input AQFP OR cell is built from buffer and constant 1 cells. The operational principle is to merge the output of two buffers and a constant 1 cell through a 3-to-1 branch. There are 4 types of OR2 gate with different inputs (positive and negative): `or_pp`, `or_pn`, `or_np`, `or_nn`. Each type of OR2 gate has both a subcell and main cell version. Here we only present the OR2 gate with two positive inputs named `or_pp` as an example.

Symbol

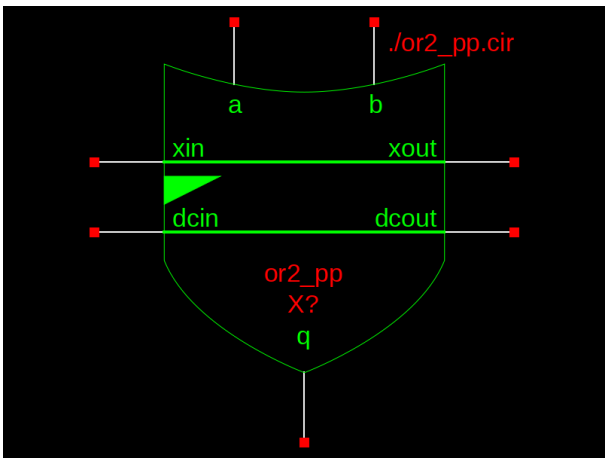


Figure 2.21: `or2_pp` symbol.

Table 2.9: `or2_pp` pin list.

Pin	Description
A	data input
B	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

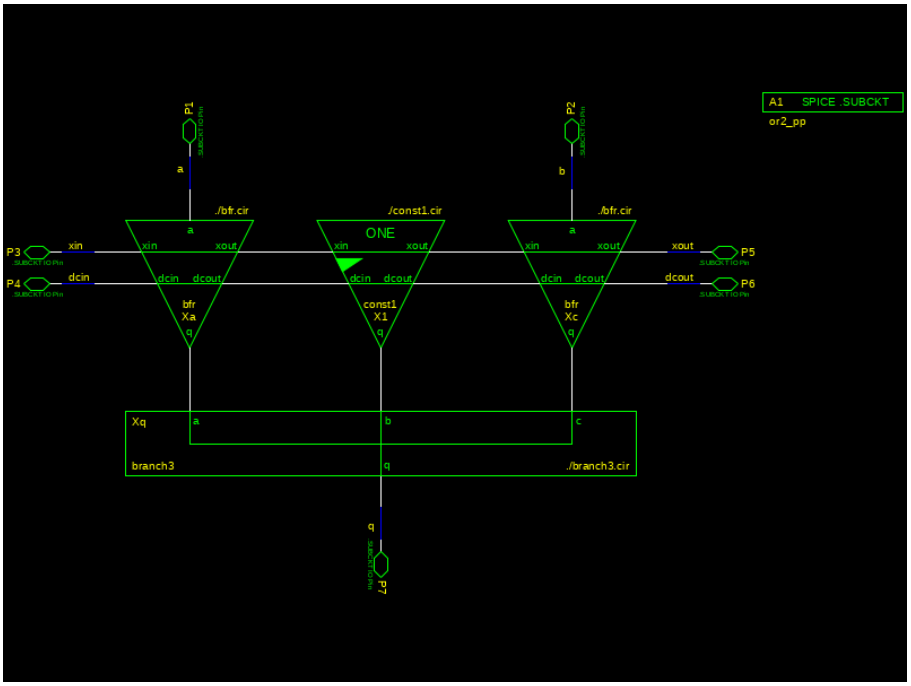


Figure 2.22: `or2_pp` schematic.

Layout

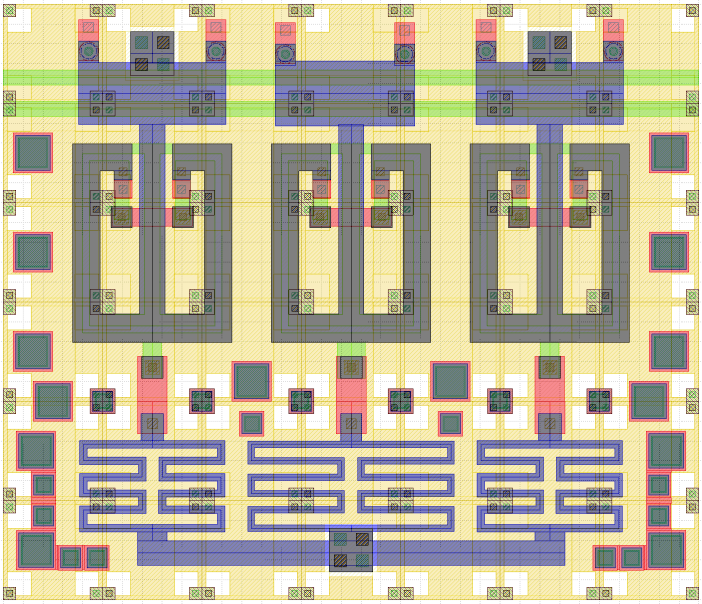


Figure 2.23: `or2_pp` layout.

Analog model

```
.include bfr_sub_i.cir
.include const1_sub.cir
.include merge3_sub_o.cir
.include bias_pair_05um.cir
.SUBCKT OR2_pp a b xin dcin xout dcout q
Xbias_in xin dcin 1 2 bias_pair_05um
Xa a 1 2 3 4 5 bfr_sub_i
Xb b 3 4 6 7 8 bfr_sub_i
Xc 6 7 9 10 11 const1_sub
Xq 5 8 11 q merge3_sub_o
Xbias_out 9 10 xout dcout bias_pair_05um
.ends OR2_pp
```

Listing 2.9: `or2_pp` netlist (.cir).

Simulation result

Simulation waveform of an 2-input OR (or2_pp) DUT (design under test) with two 4-stage buffers before the input 'a' and 'b'. Another 3-stage buffer is placed after the AND's output 'q'. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (a) as 01010101 and (b) as 00110011 (inputs of the first stage buffers), AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the output generated from the final stage of buffer (q) as 0001110111 with two random initial outputs 00. This is because of the meander structure of AQFP circuits. In an 8-stage AQFP circuit, it takes two clock cycles to propagate the data to the output. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

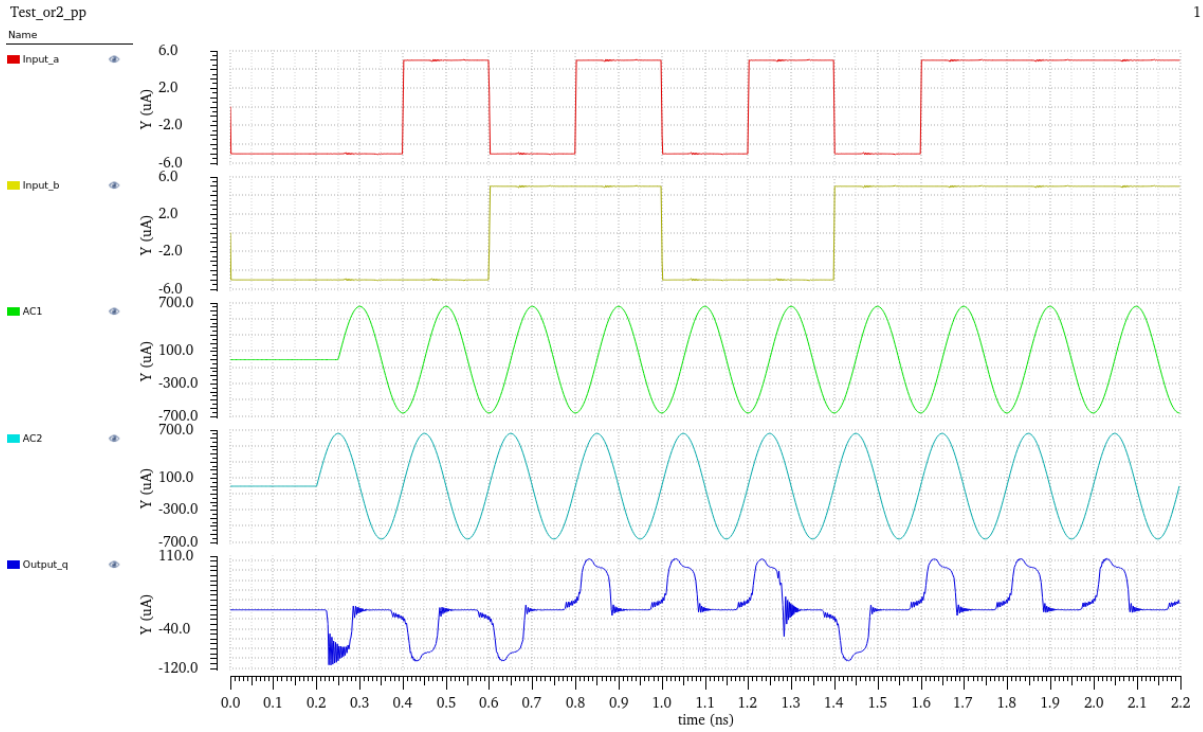


Figure 2.24: `or2_pp` analog waveform.

Digital model

```
1
2  `timescale 1ps/10fs
3  module or2_pp(a, b, q, xin, xout, dcin, dcout);
4    input a, b;
5    output q;
6    inout xin, xout, dcin, dcout;
7    reg q;
8    parameter pul_wid = 100;
9    wire not_a, not_b;
10
11   assign not_a = ~a;
12   assign not_b = ~b;
13
14   biasDir_b I0(xin, xout, dcin, dcout, gatex);
15
16   initial begin
17
18   $timeformat(-12, 1, "␣ps", 8); // time format
19
20   // output regsiter initialization
21   q = 1'bz;
22   end // initialization
23
24   specify
25     specparam d_clk   = 5;
26     specparam clk_d  = 50;
27
28     $setup(posedge a &&& a, posedge gatex, d_clk);
29     $setup(negedge a &&& not_a, posedge gatex, d_clk);
30     $hold(posedge gatex, negedge a &&& a, clk_d);
31     $hold(posedge gatex, posedge a &&& not_a, clk_d);
32
33     $setup(posedge b &&& b, posedge gatex, d_clk);
34     $setup(negedge b &&& not_b, posedge gatex, d_clk);
35     $hold(posedge gatex, negedge b &&& b, clk_d);
36     $hold(posedge gatex, posedge b &&& not_b, clk_d);
37   endspecify
38
39   always @(posedge gatex)
40     begin
41       if ((a!==1'bz) & (b!==1'bz) & (a!==1'bx) & (b!==1'bx))
42         begin
43           q <= a | b;
44           q <= #pul_wid 1'bz;
45         end
46       else
47         begin
48           q <= 1'bx;
49           q <= #pul_wid 1'bz;
50         end
51     end
52
53 endmodule
```

Listing 2.10: or2_pp Verilog model code.

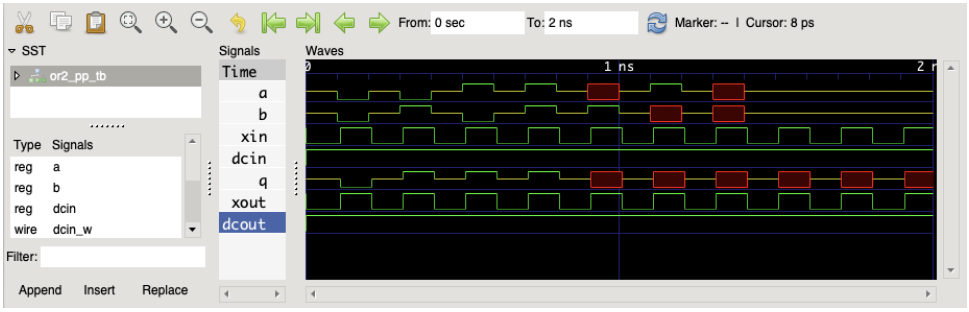


Figure 2.25: or2_pp digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($ab = 00$, $ab = 01$, $ab = 10$ and $ab = 11$) and clock frequencies.

Table 2.10: or2_pp switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'00' (J)	1.84E-21	1.67E-21	1.65E-21	2.24E-21	3.51E-21	7.10E-21	1.38E-20
'01' (J)	5.92E-21	5.79E-21	5.41E-21	4.98E-21	5.47E-21	1.07E-20	1.71E-20
'10' (J)	5.89E-21	5.76E-21	5.38E-21	4.95E-21	5.45E-21	1.07E-20	1.70E-20
'11' (J)	2.87E-23	6.33E-23	1.69E-22	3.68E-22	9.29E-22	3.93E-21	1.31E-20

2.1.6 OR3

AQFP OR cells are built from buffer cells and constant cells. The operational principle of a 3-input OR gate is to merge the output of three buffers and 2 constant '1' cells through a 5-to-1 branch. There are 8 possible types of 3-input OR gate with different inputs (positive and negative):`or3_ppp`, `or3_ppn`, `or3_pnp`, `or3_pnn`, `or3_npp`, `or3_npn`, `or3_nnp`, `or3_nnn`. Each type of OR3 gate has both a subcell and main cell version. Here we only present the 3-input OR gate main cell with three positive inputs named `or3_ppp`.

Symbol

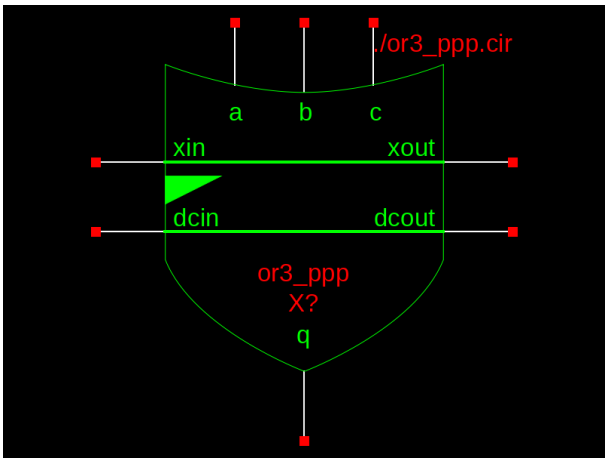


Figure 2.26: `or3_ppp` symbol.

Table 2.11: `or3_ppp` pin list.

Pin	Description
A	data input
B	data input
C	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

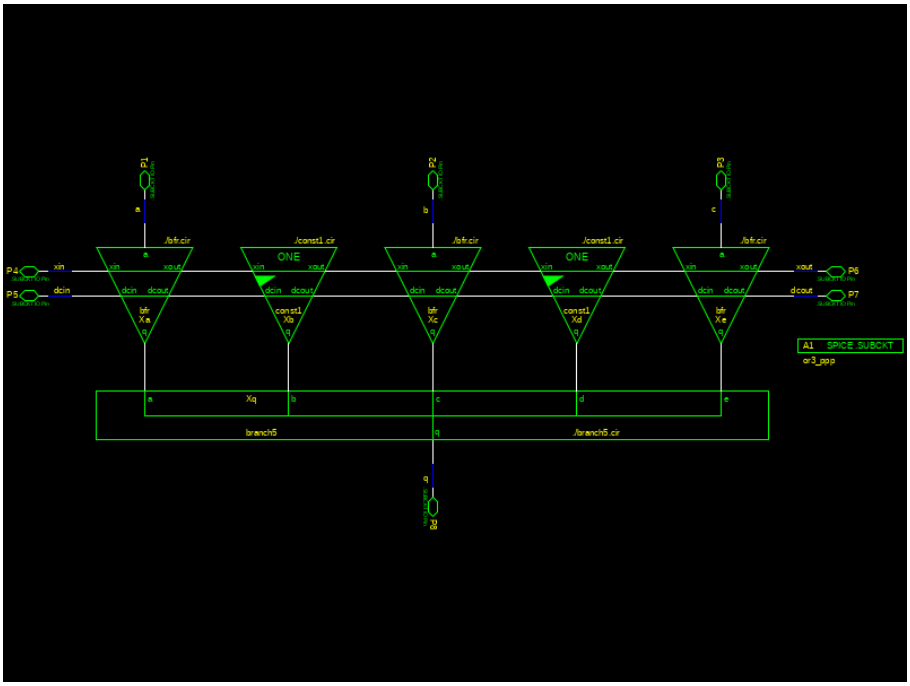


Figure 2.27: `or3_ppp` schematic.

Layout

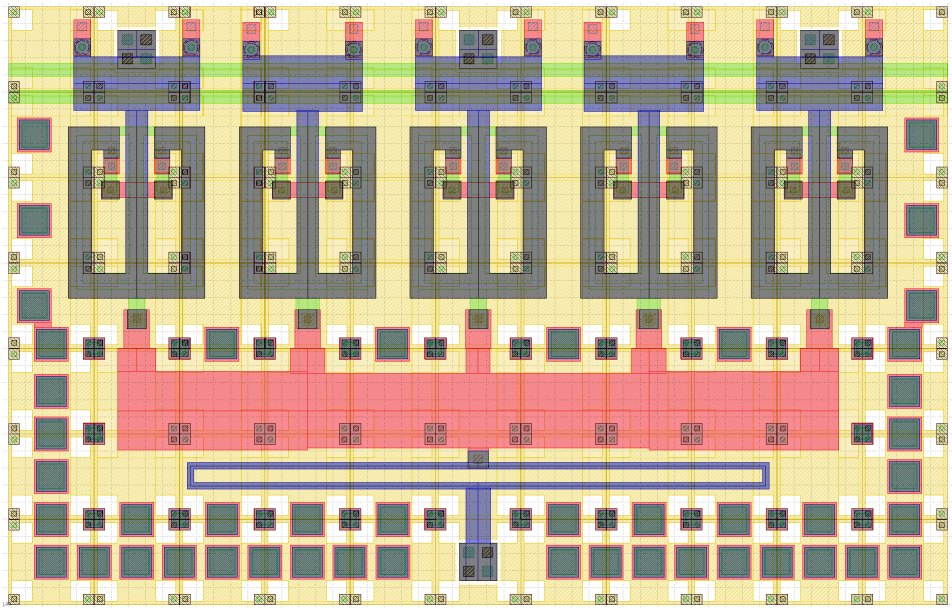


Figure 2.28: `or3_ppppp` layout.

Analog model

```
.include bfr_sub_i.cir
.include const1_sub.cir
.include bias_pair_05um.cir
.include merge5_sub_o.cir
.SUBCKT OR3_ppp a b c xin dcin xout dcout q
Xbias_in xin dcin 1 2 bias_pair_05um
Xa a 1 2 3 4 5 bfr_sub_i
Xconst1_1 3 4 6 7 8 const1_sub
Xb b 6 7 9 10 11 bfr_sub_i
Xconst1_2 9 10 12 13 14 const1_sub
Xc c 12 13 15 16 17 bfr_sub_i
Xq 5 8 11 14 17 q merge5_sub_o
Xbias_out 15 16 xout dcout bias_pair_05um
.ends OR3_ppp
```

Listing 2.11: `or3_ppp` netlist (.cir).

Simulation result

Simulation waveform of a 3-input OR (or3_ppp) DUT (design under test) with three 4-stage buffers before the input 'a', 'b' and 'c'. Another 3-stage buffer is placed after the OR's output 'q'. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (a) as 01010101, (b) as 00110011, (c) as 00001111 (inputs of the first stage buffers), AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the output generated from the final stage of buffer (q) as 1001111111 with two random initial outputs 10. This is because of the meander structure of AQFP circuits. In an 8-stage AQFP circuit, it takes two clock cycles to propagate the data to the output. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

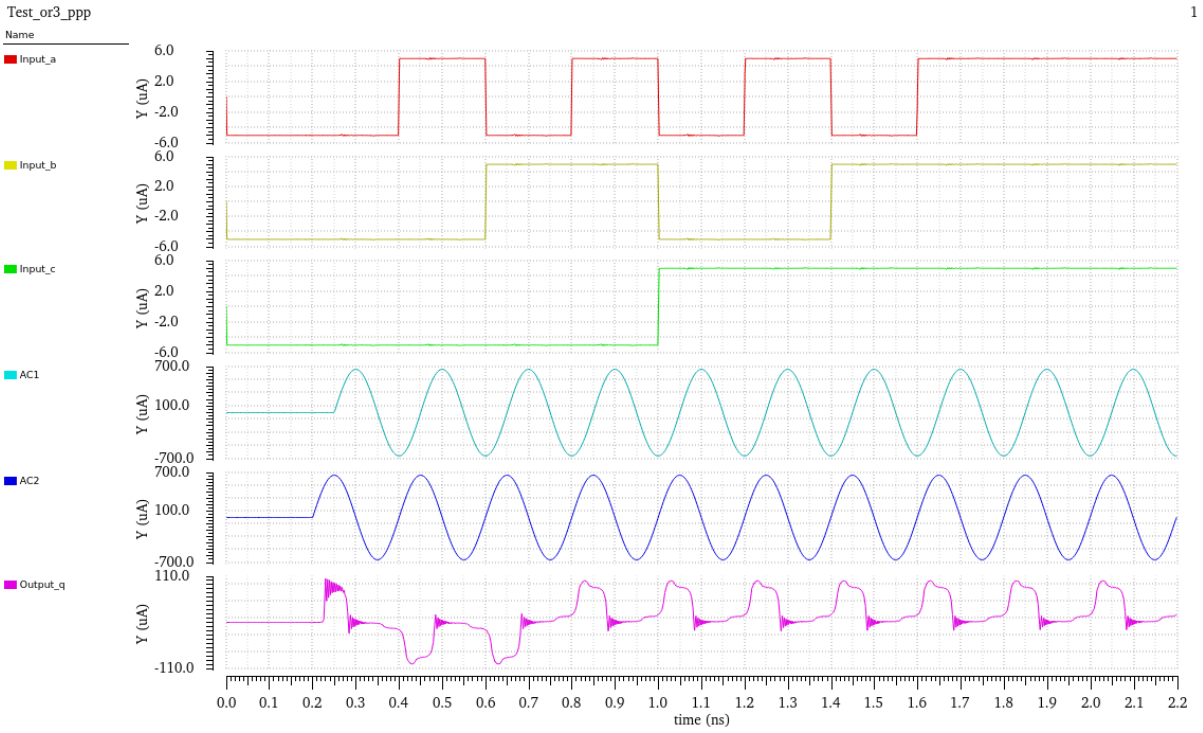


Figure 2.29: `or3_ppp` analog waveform.

Digital model

```
1
2  `timescale 1ps/10fs
3  module or3_ppp(a, b, c, q, xin, xout, dcin, dcout);
4    input a, b, c;
5    output q;
6    inout xin, xout, dcin, dcout;
7    reg q;
8    parameter pul_wid = 100;
9    wire [2:0] in_or3, neg_in_or3;
10
11   assign in_or3 = {a, b, c};
12   assign neg_in_or3 = ~in_or3;
13
14   biasDir_b I0(xin, xout, dcin, dcout, gatex);
15
16   initial begin
17
18   $timeformat(-12, 1, "␣ps", 8); // time format
19
20   // output regsiter initialization
21   q = 1'bz;
22   end // initialization
23
24   specify
25     specparam d_clk   = 5;
26     specparam clk_d  = 50;
27
28     $setup(posedge in_or3[2] &&& in_or3[2], posedge gatex, d_clk);
29     $setup(negedge in_or3[2] &&& neg_in_or3[2], posedge gatex, d_clk);
30     $hold(posedge gatex, negedge in_or3[2] &&& in_or3[2], clk_d);
31     $hold(posedge gatex, posedge in_or3[2] &&& neg_in_or3[2], clk_d);
32
33     $setup(posedge in_or3[1] &&& in_or3[1], posedge gatex, d_clk);
34     $setup(negedge in_or3[1] &&& neg_in_or3[1], posedge gatex, d_clk);
35     $hold(posedge gatex, negedge in_or3[1] &&& in_or3[1], clk_d);
36     $hold(posedge gatex, posedge in_or3[1] &&& neg_in_or3[1], clk_d);
37
38     $setup(posedge in_or3[0] &&& in_or3[0], posedge gatex, d_clk);
39     $setup(negedge in_or3[0] &&& neg_in_or3[0], posedge gatex, d_clk);
40     $hold(posedge gatex, negedge in_or3[0] &&& in_or3[0], clk_d);
41     $hold(posedge gatex, posedge in_or3[0] &&& neg_in_or3[0], clk_d);
42
43   endspecify
44
45   always @(posedge gatex)
46     begin
47       if ((&in_or3 !== 1'bx) & (&in_or3 !== 1'bz))
48         begin
49           q <= |in_or3;
50           q <= #pul_wid 1'bz;
51         end
52       else
53         begin
54           q <= 1'bx;
55           q <= #pul_wid 1'bz;
56         end
57     end
58 endmodule
```

Listing 2.12: or3_ppp Verilog model code.

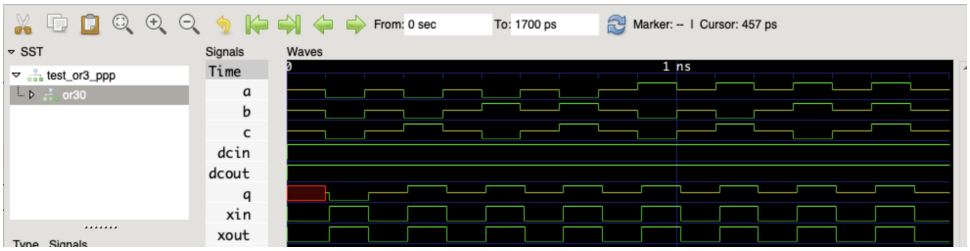


Figure 2.30: `or3_ppp` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($abc = 000$, $abc = 001$, $abc = 010$, $abc = 011$, $abc = 100$, $abc = 101$, $abc = 110$ and $abc = 111$) and clock frequencies (0.1GHz to 10GHz).

Table 2.12: `or3_ppp` switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'000' (J)	1.34E-20	1.20E-21	1.31E-21	2.35E-21	4.21E-21	1.27E-20	2.88E-20
'001' (J)	1.34E-20	1.31E-20	1.24E-20	1.13E-20	1.11E-20	2.15E-20	3.64E-20
'010' (J)	1.32E-20	1.29E-20	1.22E-20	1.11E-20	1.08E-20	2.15E-20	3.61E-20
'011' (J)	8.15E-21	8.06E-21	7.81E-21	7.39E-21	7.54E-21	1.37E-20	2.84E-20
'100' (J)	1.34E-20	1.31E-20	1.24E-20	1.13E-20	1.11E-20	2.15E-20	3.64E-20
'101' (J)	9.69E-21	1.22E-20	9.38E-21	8.95E-21	8.77E-21	1.46E-20	2.98E-20
'110' (J)	8.15E-21	8.06E-21	7.81E-21	7.39E-21	7.54E-21	1.37E-20	2.84E-20
'111' (J)	5.00E-23	1.06E-22	2.73E-22	5.77E-22	1.46E-21	6.40E-21	1.83E-20

2.1.7 MAJ3

AQFP majority cells are built from buffer cells. The operational principle of a 3-input majority gate is to merge the output of three buffers through a 3-to-1 branch. There are 8 types of 3-input majority gate with different inputs (positive and negative): `maj3_ppp`, `maj3_ppn`, `maj3_pnp`, `maj3_pnn`, `maj3_npp`, `maj3_npn`, `maj3_nnp`, `maj3_nnn`. Each type of MAJ3 gate has both a subcell and main cell version. Here we only present the 3-input majority gate main cell with three positive inputs named `maj3_ppp`.

Symbol

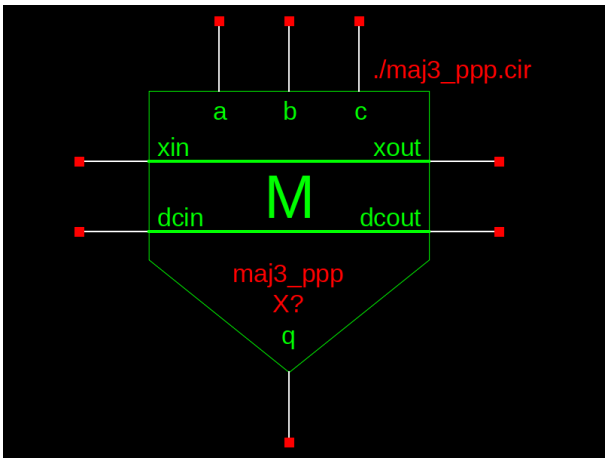


Figure 2.31: `maj3_ppp` symbol.

Table 2.13: `maj3_ppp` pin list.

Pin	Description
A	data input
B	data input
C	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

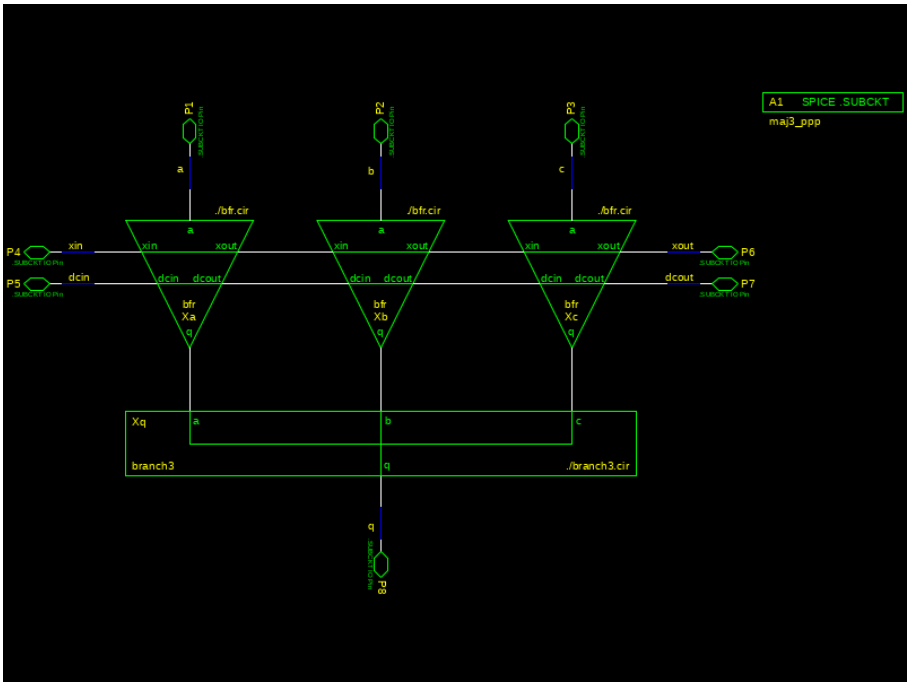


Figure 2.32: `maj3_ppp` schematic.

Layout

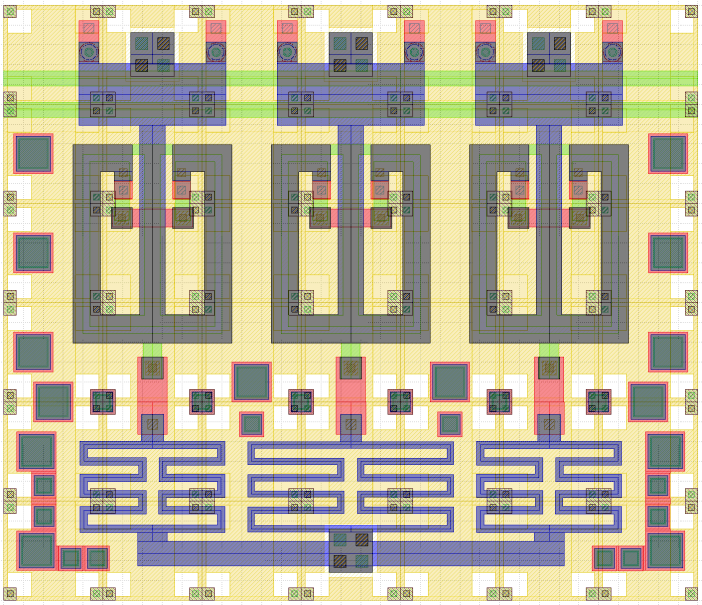


Figure 2.33: `maj3_ppp` layout.

Analog model

```
.include bfr_sub_i.cir
.include merge3_sub_o.cir
.include bias_pair_05um.cir
.SUBCKT MAJ3_ppp a b c xin dcin xout dcout q
Xbias_in xin dcin 1 2 bias_pair_05um
Xa a 1 2 3 4 5 bfr_sub_i
Xb b 3 4 6 7 8 bfr_sub_i
Xc c 6 7 9 10 11 bfr_sub_i
Xq 5 8 11 q merge3_sub_o
Xbias_out 9 10 xout dcout bias_pair_05um
.ends MAJ3_ppp
```

Listing 2.13: `maj3_ppp` netlist (.cir).

Simulation result

Simulation waveform of a 3-input MAJORITY (maj_ppp) DUT (design under test) with three 4-stage buffers before the input 'a', 'b' and 'c'. Another 3-stage buffer is placed after the MAJORITY's output 'q'. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (a) as 01010101, (b) as 00110011 and (c) as 00001111 (inputs of the first stage buffers), AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the output generated from the final stage of buffer (d) as 1000010111 with two random initial outputs 10. This is because of the meander structure of AQFP circuits. In an 8-stage AQFP circuit, it takes two clock cycles to propagate the data to the output. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

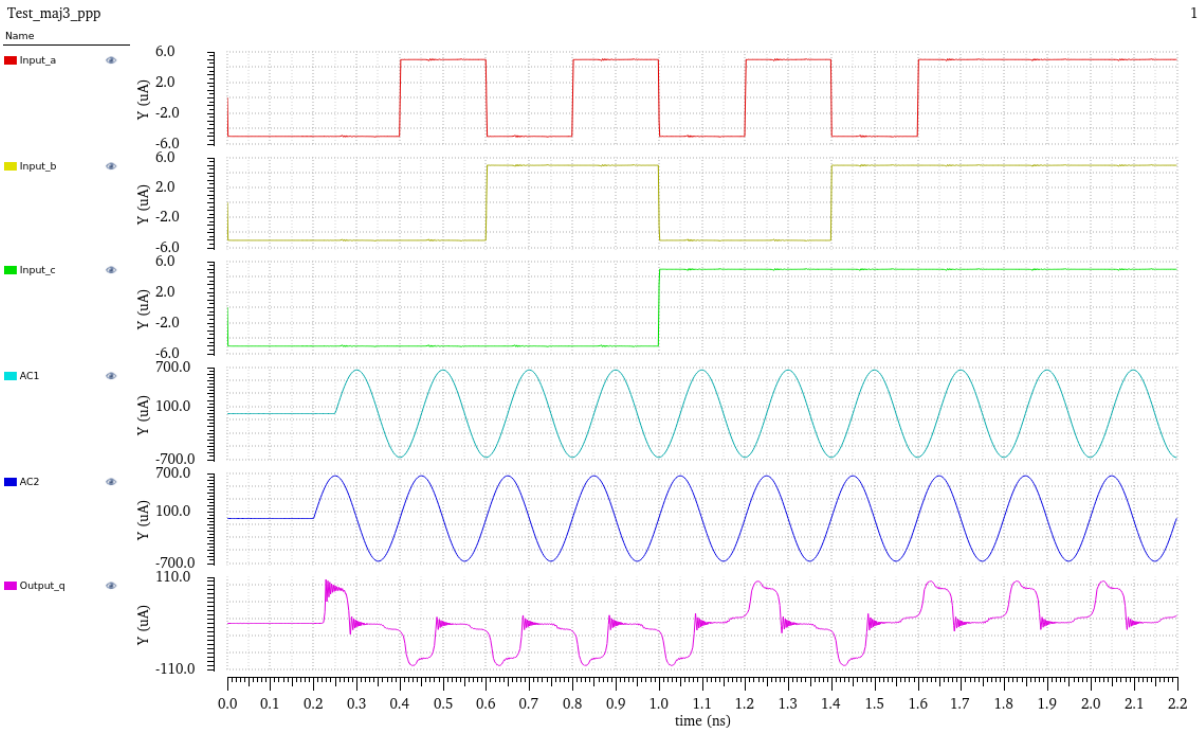


Figure 2.34: `maj3_ppp` analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2  module maj3_ppp(a, b, c, q, xin, xout, dcin, dcout);
3  input a, b, c;
4    output q;
5    inout xin, xout, dcin, dcout;
6    reg q;
7    parameter pul_wid = 100;
8    wire [2:0] in_maj3, neg_in_maj3;
9
10   assign in_maj3 = {a, b, c};
11   assign neg_in_maj3 = ~in_maj3;
12
13   biasDir_b I0(xin, xout, dcin, dcout, gatex);
14
15   initial begin
16
17   $timeformat(-12, 1, "␣ps", 8); // time format
18
19   // output regsiter initialization
20   q = 1'bz;
21   end // initialization
22
23   specify
24     specparam d_clk   = 5;
25     specparam clk_d  = 50;
26
27     $setup(posedge in_maj3[2] &&& in_maj3[2], posedge gatex, d_clk);
28     $setup(negedge in_maj3[2] &&& neg_in_maj3[2], posedge gatex, d_clk);
29     $hold(posedge gatex, negedge in_maj3[2] &&& in_maj3[2], clk_d);
30     $hold(posedge gatex, posedge in_maj3[2] &&& neg_in_maj3[2], clk_d);
31
32     $setup(posedge in_maj3[1] &&& in_maj3[1], posedge gatex, d_clk);
33     $setup(negedge in_maj3[1] &&& neg_in_maj3[1], posedge gatex, d_clk);
34     $hold(posedge gatex, negedge in_maj3[1] &&& in_maj3[1], clk_d);
35     $hold(posedge gatex, posedge in_maj3[1] &&& neg_in_maj3[1], clk_d);
36
37     $setup(posedge in_maj3[0] &&& in_maj3[0], posedge gatex, d_clk);
38     $setup(negedge in_maj3[0] &&& neg_in_maj3[0], posedge gatex, d_clk);
39     $hold(posedge gatex, negedge in_maj3[0] &&& in_maj3[0], clk_d);
40     $hold(posedge gatex, posedge in_maj3[0] &&& neg_in_maj3[0], clk_d);
41
42   endspecify
43
44   always @(posedge gatex)
45     begin
46       if ((&in_maj3 !== 1'bx) & (&in_maj3 !== 1'bz))
47         begin
48           q <= (a & b)|(b & c)|(a&&c);
49           q <= #pul_wid 1'bz;
50         end
51       else
52         begin
53           q <= 1'bx;
54           q <= #pul_wid 1'bz;
55         end
56     end
57
58 endmodule
```

Listing 2.14: maj3_ppp Verilog model code.

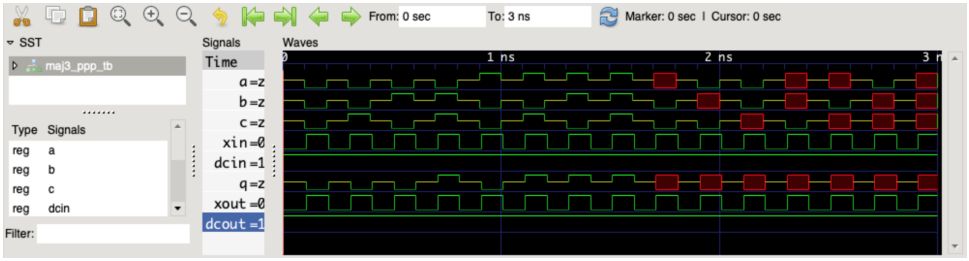


Figure 2.35: `maj3_ppp` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($abc = 000$, $abc = 001$, $abc = 010$, $abc = 011$, $abc = 100$, $abc = 101$, $abc = 110$ and $abc = 111$) and clock frequencies.

Table 2.14: `maj3_ppp` switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'000' (J)	2.36E-23	5.40E-23	1.42E-22	2.92E-22	6.28E-22	2.49E-21	8.02E-21
'001' (J)	5.29E-21	5.15E-21	4.76E-21	4.31E-21	4.63E-21	8.39E-21	1.34E-20
'010' (J)	5.82E-21	5.68E-21	5.29E-21	4.72E-21	5.01E-21	8.73E-21	1.48E-20
'011' (J)	5.30E-21	5.17E-21	4.77E-21	4.33E-21	4.64E-21	8.39E-21	1.35E-20
'100' (J)	5.30E-21	5.17E-21	4.77E-21	4.33E-21	4.64E-21	8.39E-21	1.35E-20
'101' (J)	5.82E-21	5.68E-21	5.29E-21	4.79E-21	5.01E-21	8.73E-21	1.48E-20
'110' (J)	5.29E-21	5.15E-21	4.76E-21	4.31E-21	4.63E-21	8.39E-21	1.34E-20
'111' (J)	2.36E-23	5.40E-23	1.42E-22	2.92E-22	6.28E-22	2.49E-21	8.02E-21

2.1.8 MAJ5

AQFP majority cells are built from buffer cells. The operational principle of a 5-input majority gate is to merge the output of five buffers through a 5-to-1 branch. There are 32 types of 5-input majority gate with different inputs (positive and negative). Each type of MAJ5 gate has both a subcell and main cell version. Here we only present the 5-input majority gate main cell with five positive inputs named `maj5_ppppp`.

Symbol

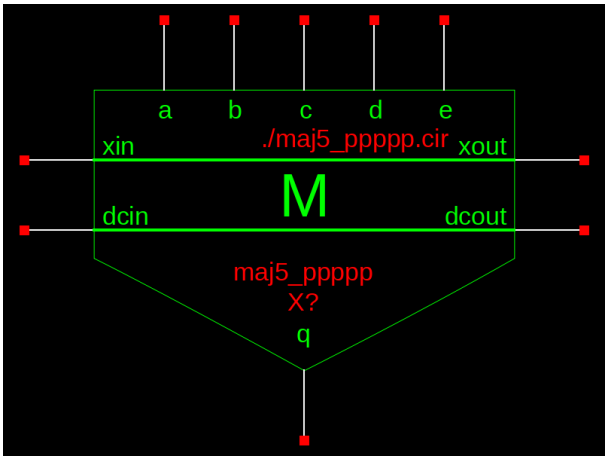


Figure 2.36: `maj5_ppppp` symbol.

Table 2.15: `maj5_ppppp` pin list.

Pin	Description
A	data input
B	data input
C	data input
D	data input
E	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

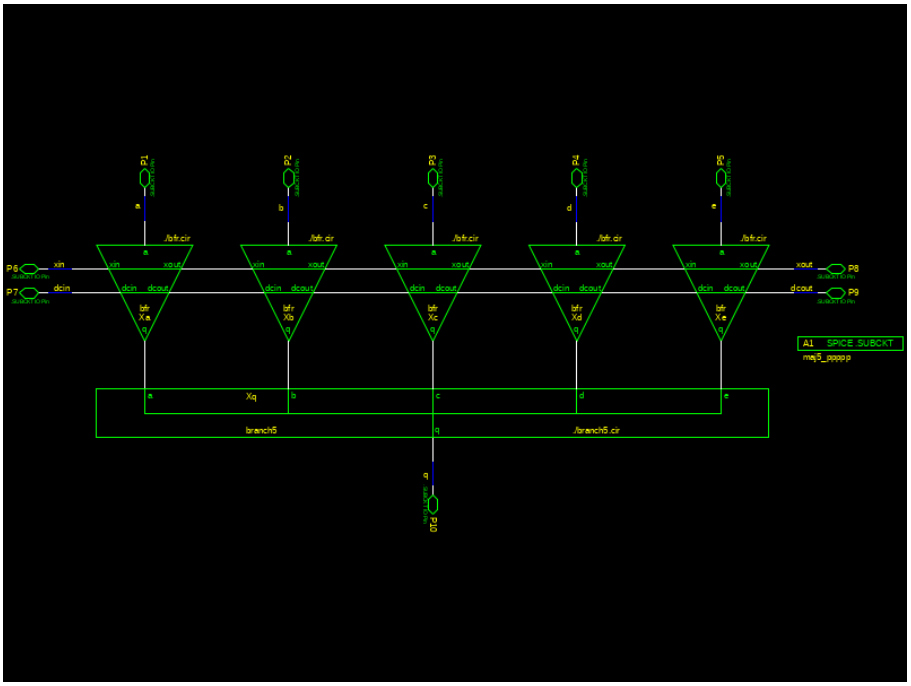


Figure 2.37: `maj5_ppppp` schematic.

Layout

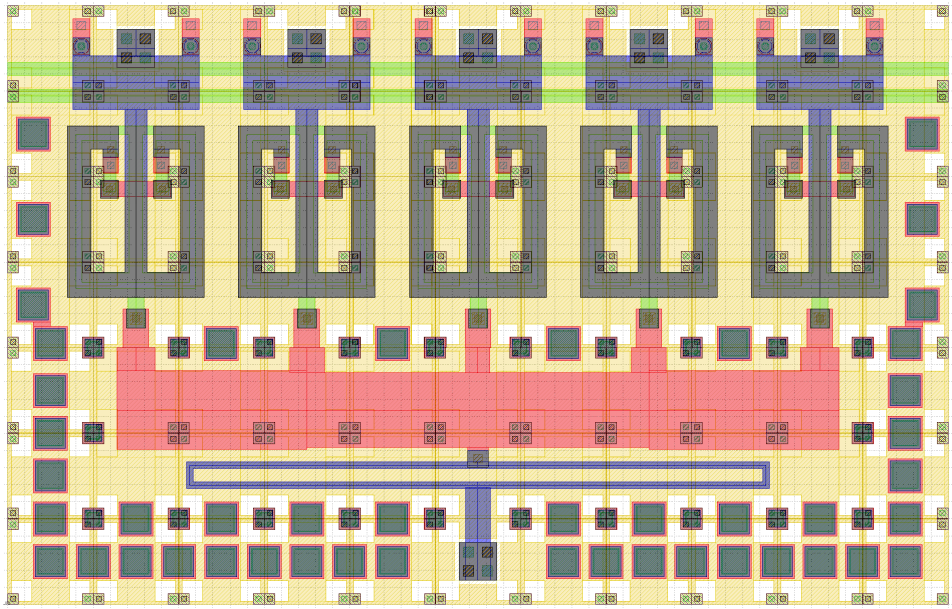


Figure 2.38: `maj5_ppppp` layout.

Analog model

```
.include bfr_sub_i.cir
.include bias_pair_05um.cir
.include merge5_sub_o.cir
.SUBCKT MAJ5_ppppp a b c d e xin dcin xout dcout q
Xbias_in xin dcin 1 2 bias_pair_05um
Xa a 1 2 3 4 5 bfr_sub_i
Xb b 3 4 6 7 8 bfr_sub_i
Xc c 6 7 9 10 11 bfr_sub_i
Xd d 9 10 12 13 14 bfr_sub_i
Xe e 12 13 15 16 17 bfr_sub_i
Xq 5 8 11 14 17 q merge5merge5_sub_o
Xbias_out 15 16 xout dcout bias_pair_05um
.ends MAJ5_ppppp
```

Listing 2.15: `maj5_ppppp` netlist (.cir).

Simulation result

Simulation waveform of a 5-input MAJORITY (maj5_ppppp) DUT (design under test) with three 4-stage buffers before the input 'a', 'b', 'c', 'd' and 'e'. Another 3-stage buffer is placed after the MAJORITY's output 'q'. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (a) as 01010100101101010101010101, (b) as 00110011001100110011001100110011, (c) as 00001111000011110000111100001111, (d) as 00000001111111100000000011111111, and (e) as 00000000000000000111111111111111 (inputs of the first stage buffers), AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the output generated from the final stage of buffer (q) as 11000000010001011100010111011111111 with two random initial outputs 11. This is because of the meander structure of AQFP circuits. In an 8-stage AQFP circuit, it takes two clock cycles to propagate the data to the output. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

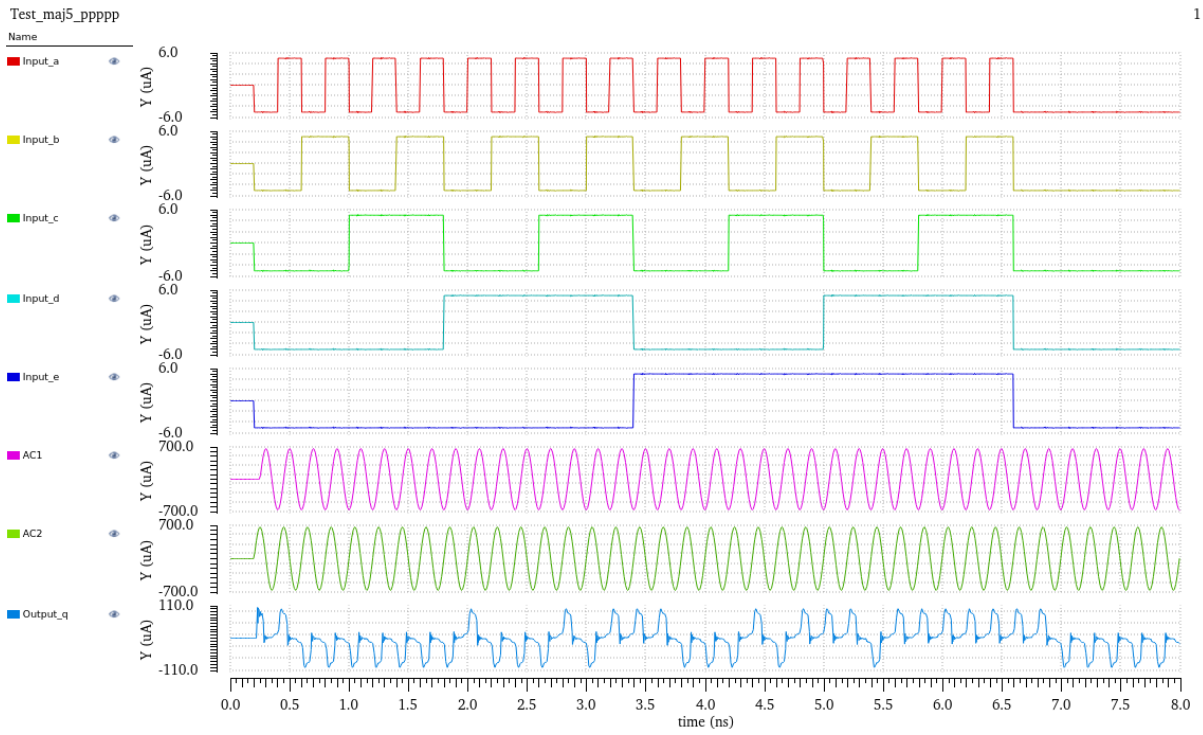


Figure 2.39: `maj5_ppppp` analog waveform.

Digital model

```
1
2  `timescale 1ps/10fs
3  module maj5_ppppp(a, b, c, d, e, q, xin, xout, dcin, dcout);
4    input a, b, c, d, e;
5    output q;
6    inout xin, xout, dcin, dcout;
7    wire [4:0] in_maj5;
8    reg q;
9    parameter pul_wid = 100;
10   wire [4:0] neg_in_maj5;
11
12   assign in_maj5 = {a, b, c, d, e};
13   assign neg_in_maj5 = ~in_maj5;
14
15   biasDir_b I0(xin, xout, dcin, dcout, gatex);
16
17   initial begin
18
19   $timeformat(-12, 1, " ps", 8); // time format
20
21   // output regsiter initialization
22   q = 1'bz;
23   end // initialization
24
25   specify
26     specparam d_clk   = 5;
27     specparam clk_d  = 50;
28
29     $setup(posedge in_maj5[4] &&& in_maj5[4], posedge gatex, d_clk);
30     $setup(negedge in_maj5[4] &&& neg_in_maj5[4], posedge gatex, d_clk);
31     $hold(posedge gatex, negedge in_maj5[4] &&& in_maj5[4], clk_d);
32     $hold(posedge gatex, posedge in_maj5[4] &&& neg_in_maj5[4], clk_d);
33
34     $setup(posedge in_maj5[3] &&& in_maj5[3], posedge gatex, d_clk);
```

```
35      $setup(negedge in_maj5[3] &&& neg_in_maj5[3], posedge gatex, d_clk);
36      $hold(posedge gatex, negedge in_maj5[3] &&& in_maj5[3], clk_d);
37      $hold(posedge gatex, posedge in_maj5[3] &&& neg_in_maj5[3], clk_d);
38
39      $setup(posedge in_maj5[2] &&& in_maj5[2], posedge gatex, d_clk);
40      $setup(negedge in_maj5[2] &&& neg_in_maj5[2], posedge gatex, d_clk);
41      $hold(posedge gatex, negedge in_maj5[2] &&& in_maj5[2], clk_d);
42      $hold(posedge gatex, posedge in_maj5[2] &&& neg_in_maj5[2], clk_d);
43
44      $setup(posedge in_maj5[1] &&& in_maj5[1], posedge gatex, d_clk);
45      $setup(negedge in_maj5[1] &&& neg_in_maj5[1], posedge gatex, d_clk);
46      $hold(posedge gatex, negedge in_maj5[1] &&& in_maj5[1], clk_d);
47      $hold(posedge gatex, posedge in_maj5[1] &&& neg_in_maj5[1], clk_d);
48
49      $setup(posedge in_maj5[0] &&& in_maj5[0], posedge gatex, d_clk);
50      $setup(negedge in_maj5[0] &&& neg_in_maj5[0], posedge gatex, d_clk);
51      $hold(posedge gatex, negedge in_maj5[0] &&& in_maj5[0], clk_d);
52      $hold(posedge gatex, posedge in_maj5[0] &&& neg_in_maj5[0], clk_d);
53    endspecify
54
55    always @(posedge gatex)
56      begin
57        if (&in_maj5!== 1'bx & &in_maj5!== 1'bz)
58          begin
59            q <= (a & b & c)|(a & b &d)|(a &b &e)|(a & c & d)|(a & c & e)|(a & d & e)|(b & c
                  ↪  & d)|(b & c & e)|(b & d & e)|(c & d & e);
60            q <= #pul_wid 1'bz;
61          end
62        else
63          begin
64            q <= 1'bx;
65            q <= #pul_wid 1'bz;
66          end
67      end
68
69  endmodule
```

Listing 2.16: `maj5_ppppp` Verilog model code.

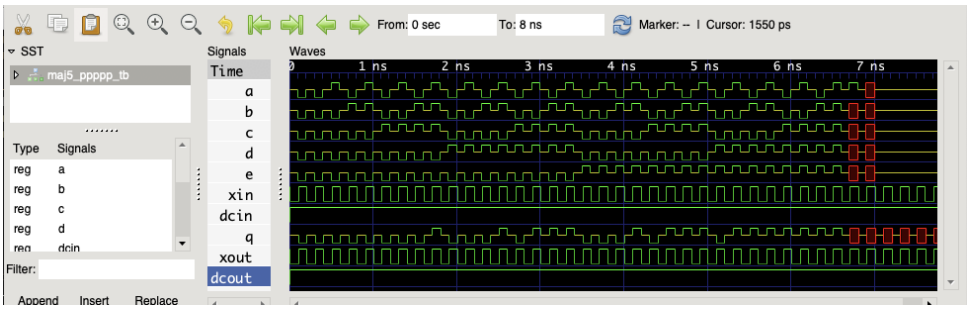


Figure 2.40: `maj5_ppppp` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns (from $abcde = 00000$ to $abcde = 11111$) and clock frequencies (from 0.1GHz to 10GHz).

Table 2.16: `maj5_ppppp` switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'00000' (J)	4.55E-23	9.68E-23	2.52E-22	5.38E-22	1.35E-21	5.62E-21	1.54E-20
'00001' (J)	6.07E-21	5.96E-21	5.63E-21	5.07E-21	5.96E-21	1.25E-20	1.78E-20
'00010' (J)	6.82E-21	6.71E-21	6.40E-21	5.95E-21	5.69E-21	1.31E-20	1.87E-20
'00011' (J)	8.60E-21	8.31E-21	7.47E-21	6.72E-21	6.76E-21	1.42E-20	1.59E-20
'00100' (J)	7.29E-21	7.18E-21	6.88E-21	6.32E-21	5.69E-21	1.33E-20	1.99E-20
'00101' (J)	1.03E-20	1.00E-20	9.18E-21	8.14E-21	7.96E-21	1.59E-20	1.96E-20
'00110' (J)	1.05E-20	1.02E-20	9.37E-21	8.26E-21	7.91E-21	1.61E-20	2.07E-20
'00111' (J)	8.60E-21	8.31E-21	7.47E-21	6.72E-21	6.76E-21	1.42E-20	1.59E-20
'01000' (J)	6.82E-21	6.71E-21	6.40E-21	5.95E-21	5.69E-21	1.31E-20	1.87E-20
'01001' (J)	1.03E-20	9.95E-21	9.08E-21	8.05E-21	7.56E-21	1.64E-20	1.89E-20
'01010' (J)	1.05E-20	1.02E-20	9.37E-21	8.09E-21	7.68E-21	1.68E-20	1.97E-20
'01011' (J)	1.03E-20	1.00E-20	9.18E-21	8.14E-21	7.96E-21	1.59E-20	1.96E-20
'01100' (J)	1.05E-20	1.02E-20	9.37E-21	8.26E-21	7.91E-21	1.61E-20	2.07E-20
'01101' (J)	1.03E-20	9.95E-21	9.08E-21	8.05E-21	7.56E-21	1.64E-20	1.89E-20
'01110' (J)	9.64E-21	9.35E-21	8.49E-21	7.39E-21	6.80E-21	1.59E-20	1.69E-20
'01111' (J)	6.07E-21	5.96E-21	5.63E-21	5.07E-21	5.96E-21	1.25E-20	1.78E-20
'10000' (J)	6.07E-21	5.96E-21	5.63E-21	5.07E-21	5.96E-21	1.25E-20	1.78E-20
'10001' (J)	9.64E-21	9.35E-21	8.49E-21	7.39E-21	6.80E-21	1.59E-20	1.69E-20
'10010' (J)	1.03E-20	9.95E-21	9.08E-21	8.05E-21	7.56E-21	1.64E-20	1.89E-20
'10011' (J)	1.05E-20	1.02E-20	9.37E-21	8.26E-21	7.91E-21	1.61E-20	2.07E-20
'10100' (J)	1.03E-20	1.00E-20	9.18E-21	8.14E-21	7.96E-21	1.59E-20	1.96E-20
'10101' (J)	1.05E-20	1.02E-20	9.37E-21	8.09E-21	7.68E-21	1.68E-20	1.97E-20
'10110' (J)	1.03E-20	9.95E-21	9.08E-21	8.05E-21	7.56E-21	1.64E-20	1.89E-20
'10111' (J)	6.82E-21	6.71E-21	6.40E-21	5.95E-21	5.69E-21	1.31E-20	1.87E-20
'11000' (J)	8.60E-21	8.31E-21	7.47E-21	6.72E-21	6.76E-21	1.42E-20	1.59E-20
'11001' (J)	1.05E-20	1.02E-20	9.37E-21	8.26E-21	7.91E-21	1.61E-20	2.07E-20
'11010' (J)	1.03E-20	1.00E-20	9.18E-21	8.14E-21	7.96E-21	1.59E-20	1.96E-20
'11011' (J)	7.29E-21	7.18E-21	6.88E-21	6.32E-21	5.69E-21	1.33E-20	1.99E-20
'11100' (J)	8.60E-21	8.31E-21	7.47E-21	6.72E-21	6.76E-21	1.42E-20	1.59E-20
'11101' (J)	6.82E-21	6.71E-21	6.40E-21	5.95E-21	5.69E-21	1.31E-20	1.87E-20
'11110' (J)	6.07E-21	5.96E-21	5.63E-21	5.07E-21	5.96E-21	1.25E-20	1.78E-20
'11111' (J)	4.55E-23	9.68E-23	2.52E-22	5.38E-22	1.35E-21	5.62E-21	1.54E-20

2.1.9 BOOST1

Boosters are the buffering elements in AQFP logic with large output current, which are used to increase the maximum interconnect wirelength during data propagation with low bit-error rate. They are also used to design splitting elements to increase the fanout during data propagation. There are two types of booster cells: `boost1` and `boost2`.

`boost1` is designed to amplify the output current of a single buffer to achieve higher drivability and longer maximum interconnect wirelength. The operational principle is to first split the input of a buffer and send the split signal to an additional buffer, then merge the outputs of two buffers by coupling the two outputs with one single inductor.

Symbol

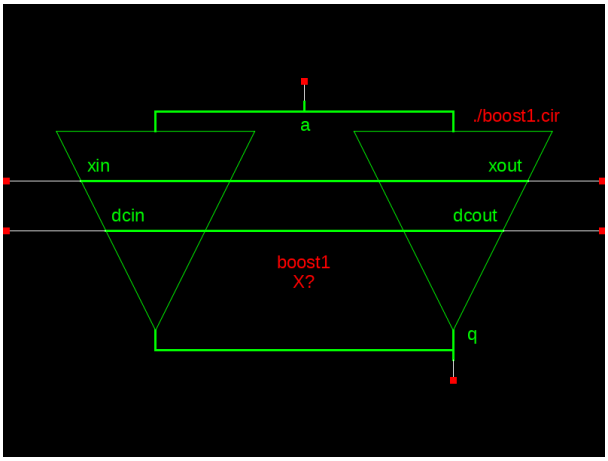


Figure 2.41: `boost1` symbol.

Table 2.17: `boost1` pin list.

Pin	Description
A	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

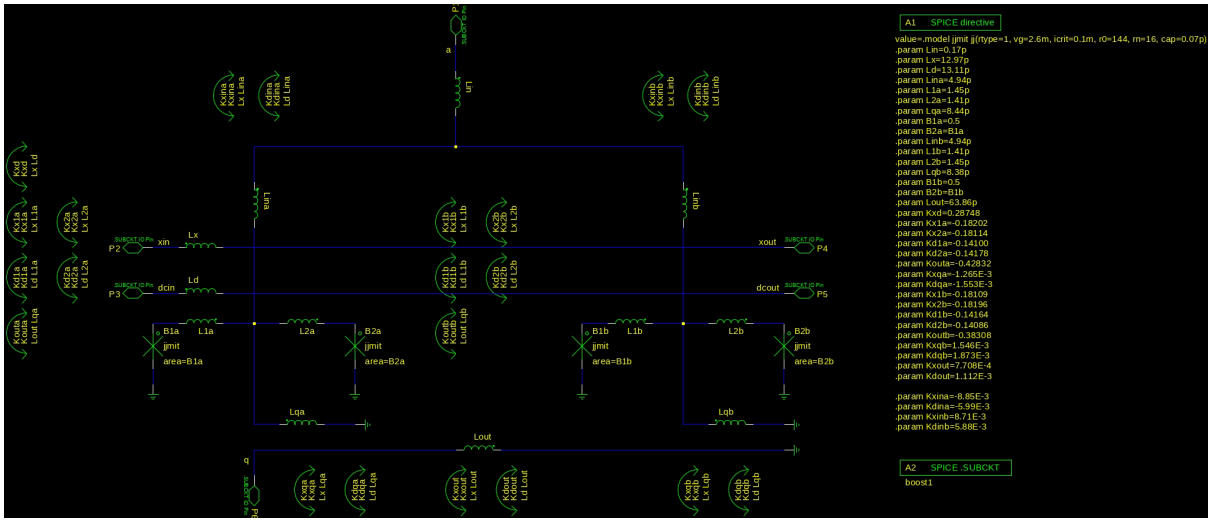


Figure 2.42: `boost1` schematic.

Layout

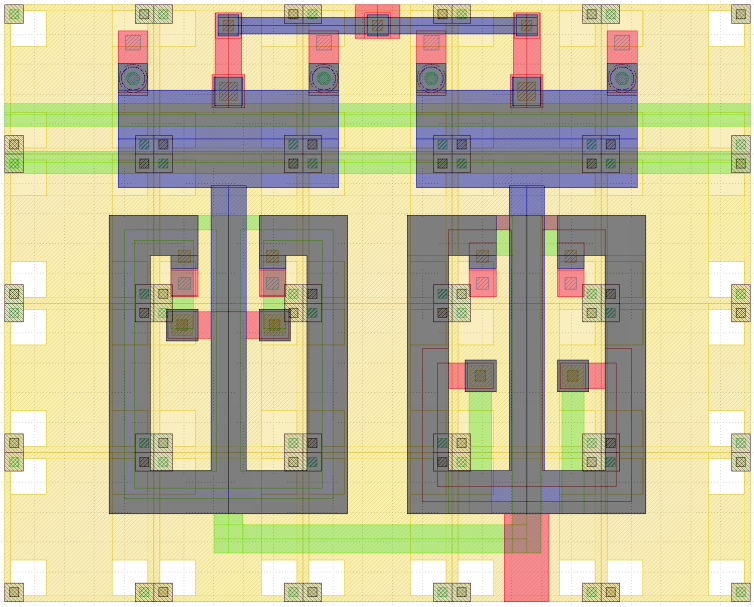


Figure 2.43: `boost1` layout.

Analog model

```
.SUBCKT boost1 a xin dcin xout dcout q
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param B1a=0.5
.param B2a=B1a
.param B1b=0.5
.param B2b=B1b
.param Lin=0.17p
```

```
.param Lina=4.94p
.param Linb=Lina
.param L1a=1.45p
.param L2a=1.41p
.param Lqa=8.44p
.param L1b=L1a
.param L2b=L2a
.param Lqb=Lqa
.param Lout=63.86p
.param Lx=12.97p
.param Ld=13.11p
.param Kxd=0.28748
.param Kx1a=-0.18202
.param Kx2a=-0.18114
.param Kd1a=-0.14100
.param Kd2a=-0.14178
.param Kouta=-0.42832
.param Kxqa=-1.265E-3
.param Kdqa=-1.553E-3
.param Kx1b=-0.18109
.param Kx2b=-0.18196
.param Kd1b=-0.14164
.param Kd2b=-0.14086
.param Koutb=-0.38308
.param Kxqb=1.546E-3
.param Kdqb=1.873E-3
.param Kxout=7.708E-4
.param Kdout=1.112E-3
.param Kxina=-8.85E-3
.param Kdina=-5.99E-3
.param Kxinb=8.71E-3
.param Kdinb=5.88E-3
B1a 1 0 jjmit area=B1a
B2a 3 0 jjmit area=B2a
B1b 5 0 jjmit area=B1b
B2b 7 0 jjmit area=B2b
Lin a 4 Lin
Lina 4 2 Lina
L1a 2 1 L1a
L2a 3 2 L2a
Lqa 2 0 Lqa
Linb 4 6 Linb
L1b 6 5 L1b
L2b 7 6 L2b
Lqb 6 0 Lqb
Lout 0 q Lout
Lx xin xout Lx
Ld dcin dcout Ld
Kdinb Ld Linb Kdinb
Kxinb Lx Linb Kxinb
Kdina Ld Lina Kdina
Kxina Lx Lina Kxina
Kxqa Lx Lqa Kxqa
Kdqa Ld Lqa Kdqa
Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kouta Lout Lqa Kouta
Kd2a Ld L2a Kd2a
Kx2a Lx L2a Kx2a
Kd1a Ld L1a Kd1a
Kx1a Lx L1a Kx1a
Kxd Lx Ld Kxd
Kxqb Lx Lqb Kxqb
Kdqb Ld Lqb Kdqb
Koutb Lout Lqb Koutb
Kd2b Ld L2b Kd2b
Kx2b Lx L2b Kx2b
Kd1b Ld L1b Kd1b
Kx1b Lx L1b Kx1b
```

```
77 | .ends boost1
```

Listing 2.17: `boost1` netlist (.cir).

Simulation result

Simulation waveform of a buffer type1 (`boost1`) DUT (design under test) with a 4-stage buffer before the input 'a'. Another two 3-stage buffers are placed after the `boost1`'s output 'q'. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (a) (input of the first stage buffers) as 01010101, AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the outputs generated from the output generated from the final stage of buffer (q) as 0001010101 with two random initial outputs 00. This is because of the meander structure of AQFP circuits. In an 8-stage AQFP circuit, it takes two clock cycles to propagate the data to the output. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

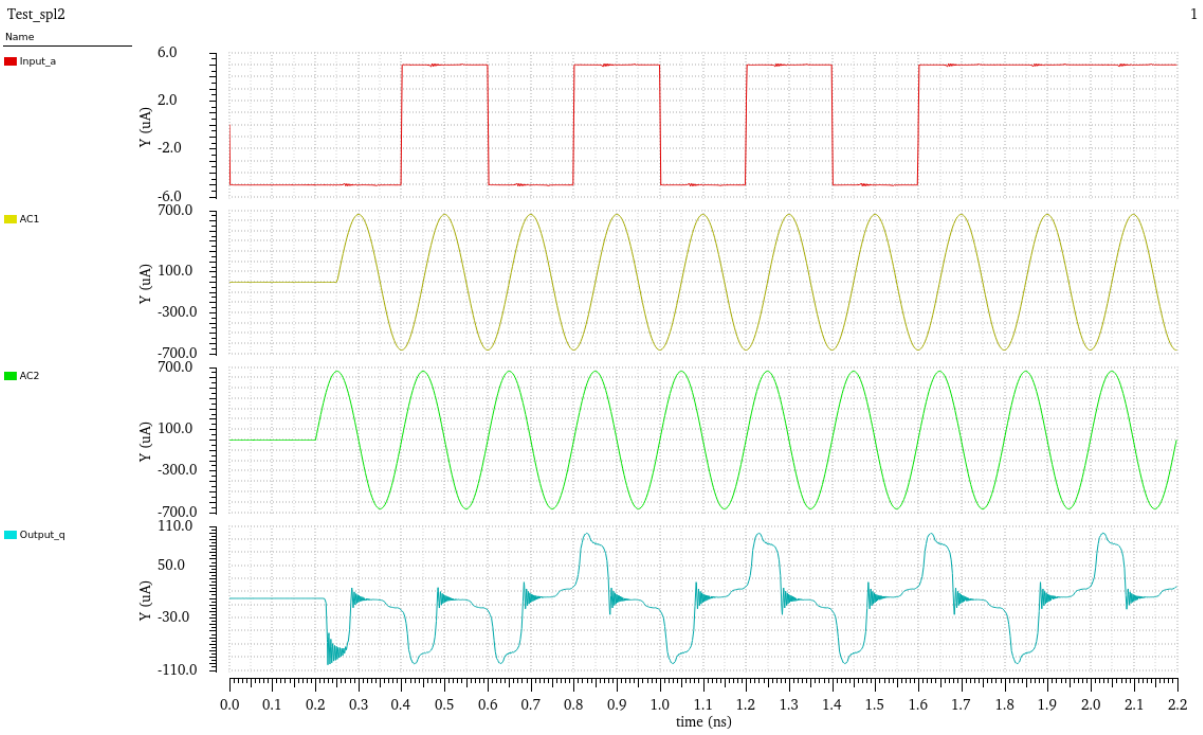


Figure 2.44: `boost1` analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2  module boost1(a, q, xin, xout, dcin, dcout);
3    input a;
4    output q;
5    inout xin, xout, dcin, dcout;
6    reg q;
7    parameter pul_wid = 100;
8    wire not_a;
9
10   assign not_a = !a;
11
12   biasDir_b I0(xin, xout, dcin, dcout, gatex);
```

```
13
14    initial begin
15
16    $timeformat(-12, 1, "␣ps", 8); // time format
17
18    // output regsiter initialization
19    q = 1'bz;
20    end // initialization
21
22    specify
23      specparam d_clk   = 5;
24      specparam clk_d  = 50;
25
26      $setup(posedge a &&& a, posedge gatex, d_clk);
27      $setup(negedge a &&& not_a, posedge gatex, d_clk);
28      $hold(posedge gatex, negedge a &&& a, clk_d);
29      $hold(posedge gatex, posedge a &&& not_a, clk_d);
30    endspecify
31
32    always @(posedge gatex)
33      begin
34        if (a == 1 | a == 0)
35          begin
36            q <= a;
37            q <= #pul_wid 1'bz;
38          end
39        else
40          begin
41            q <= 1'bx;
42            q <= #pul_wid 1'bz;
43          end
44      end
45
46  endmodule
```

Listing 2.18: `boost1` Verilog model code.

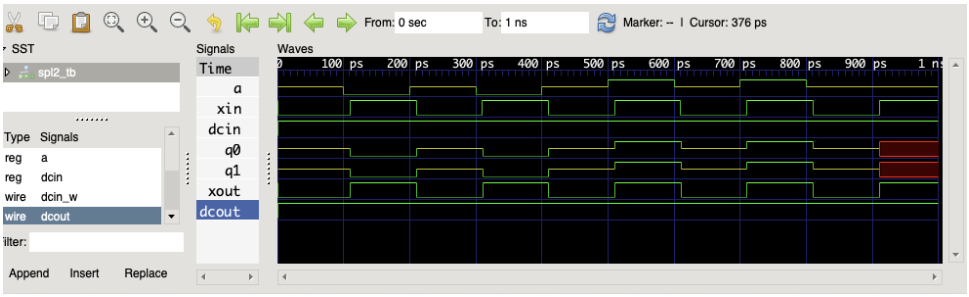


Figure 2.45: `spl2` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($a = 0$ and $a = 1$) and clock frequencies.

Table 2.18: `spl2` switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'0' (J)	2.72E-23	7.43E-23	2.16E-22	4.65E-22	1.09E-21	4.12E-21	1.05E-20
'1' (J)	2.71E-23	7.41E-23	2.16E-22	4.64E-22	1.09E-21	4.12E-21	1.05E-20

2.1.10 BOOST2

Boosters are the buffering elements in AQFP logic with large output current, which are used to increase the maximum interconnect wirelength during data propagation with low bit-error rate. They are also used to design splitting elements to increase the fanout during data propagation. There are two types of booster cells: `boost1` and `boost2`.

`boost2` uses another approach to first amplify the input current use a single buffer, then split the output and connect them to multiple buffers to obtain multiple fanouts. All the buffers in `boost2` share the same ac (clock) interconnect, therefore there is no extra clock phase consumed in this design. There are 2 types of boost2 cells with different fanouts: boost2 with 2 fanouts (`boost2f2`) and with 4 fanouts (`boost2f4`). Here we only present the `boost2f2` cell as an examples.

Symbol

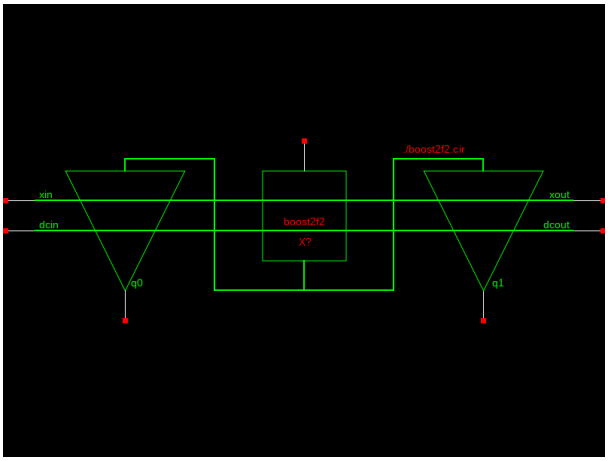


Figure 2.46: `boost2f2` symbol.

Table 2.19: `boost2f2` pin list.

Pin	Description
A	data input
XIN	serial clock input
DCIN	dc offset input
Q0	data output
Q1	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

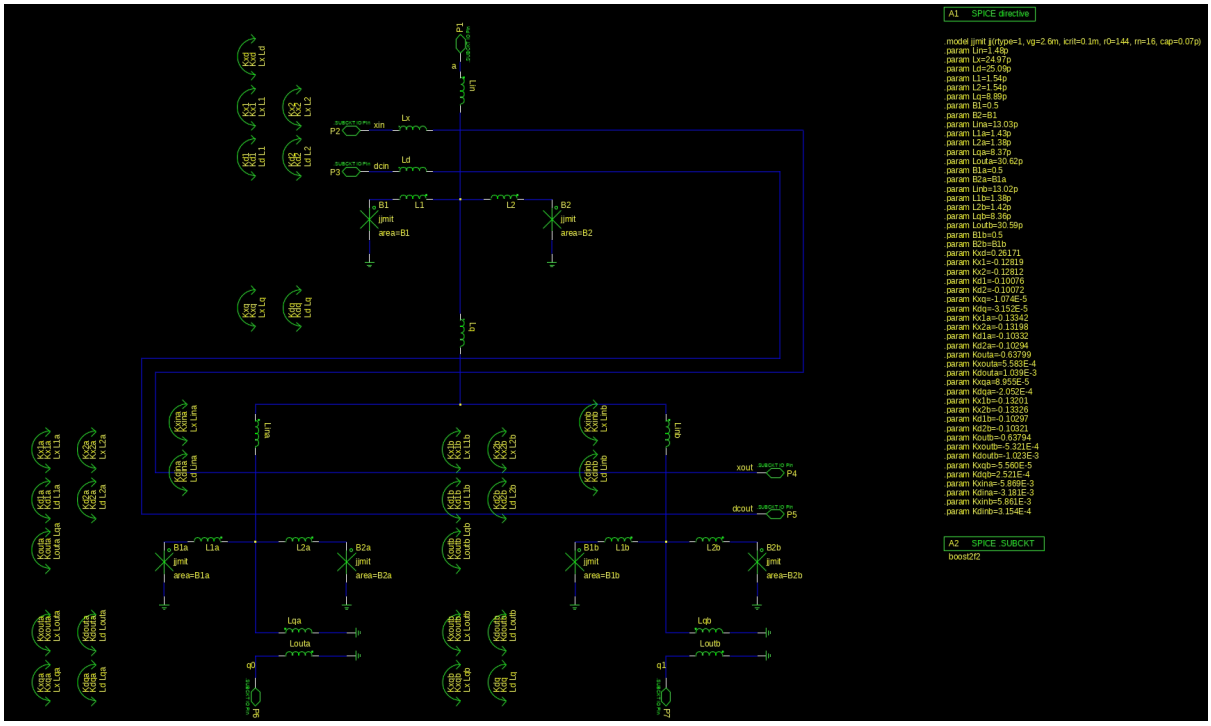


Figure 2.47: `boost2f2` schematic.

Layout

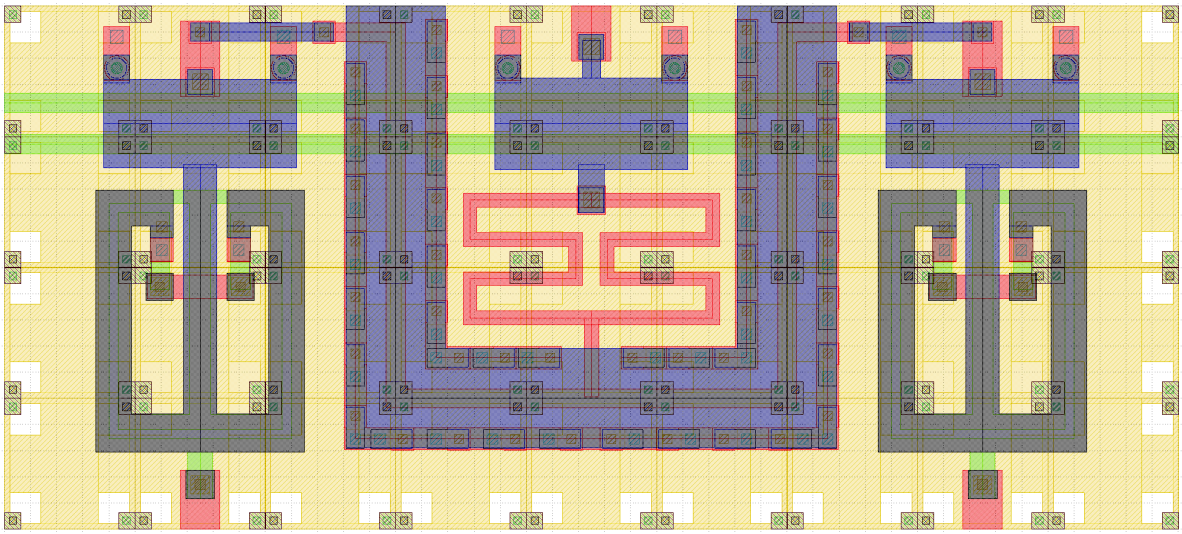


Figure 2.48: `boost2f2` layout.

Analog model

```
1 .SUBCKT boost2f2 a xin dcin xout dcout q0 q1
2 .model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
```

```
3  .param Lin=1.48p
4  .param Lx=24.97p
5  .param Ld=25.09p
6  .param L1=1.54p
7  .param L2=1.54p
8  .param Lq=8.89p
9  .param B1=0.5
10 .param B2=B1
11 .param Lina=13.03p
12 .param L1a=1.43p
13 .param L2a=1.38p
14 .param Lqa=8.37p
15 .param Louta=30.62p
16 .param B1a=0.5
17 .param B2a=B1a
18 .param Linb=13.02p
19 .param L1b=1.38p
20 .param L2b=1.42p
21 .param Lqb=8.36p
22 .param Loutb=30.59p
23 .param B1b=0.5
24 .param B2b=B1b
25 .param Kxd=0.26171
26 .param Kx1=-0.12819
27 .param Kx2=-0.12812
28 .param Kd1=-0.10076
29 .param Kd2=-0.10072
30 .param Kxq=-1.074E-5
31 .param Kdq=-3.152E-5
32 .param Kx1a=-0.13342
33 .param Kx2a=-0.13198
34 .param Kd1a=-0.10332
35 .param Kd2a=-0.10294
36 .param Kouta=-0.63799
37 .param Kxouta=5.583E-4
38 .param Kdouta=1.039E-3
39 .param Kxqa=8.955E-5
40 .param Kdqa=-2.052E-4
41 .param Kx1b=-0.13201
42 .param Kx2b=-0.13326
43 .param Kd1b=-0.10297
44 .param Kd2b=-0.10321
45 .param Koutb=-0.63794
46 .param Kxoutb=-5.321E-4
47 .param Kdoutb=-1.023E-3
48 .param Kxqb=-5.560E-5
49 .param Kdqb=2.521E-4
50 .param Kxina=-5.869E-3
51 .param Kdina=-3.181E-3
52 .param Kxinb=5.861E-3
53 .param Kdinb=3.154E-4
54 B1 1 0 jjmit area=B1
55 B2 3 0 jjmit area=B2
56 B1a 5 0 jjmit area=B1a
57 B2a 7 0 jjmit area=B2a
58 B1b 8 0 jjmit area=B1b
59 B2b 10 0 jjmit area=B2b
60 Lin a 2 Lin
61 L1 2 1 L1
62 L2 3 2 L2
63 Lq 2 4 Lq
64 Lina 4 6 Lina
65 Linb 4 9 Linb
66 L1a 6 5 L1a
67 L2a 7 6 L2a
68 Lqa 6 0 Lqa
69 Louta 0 q0 Louta
70 L1b 9 8 L1b
71 L2b 10 9 L2b
72 Lqb 9 0 Lqb
```

```
Loutb 0 q1 Loutb
Ld dcin dcout Ld
Lx xin xout Lx
Kxq Lx Lq Kxq
Kdq Ld Lq Kdq
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Kxqa Lx Lqa Kxqa
Kdqa Ld Lqa Kdqa
Kxouta Lx Louta Kxouta
Kdouta Ld Louta Kdouta
Kouta Louta Lqa Kouta
Kd2a Ld L2a Kd2a
Kx2a Lx L2a Kx2a
Kd1a Ld L1a Kd1a
Kx1a Lx L1a Kx1a
Kdinb Ld Linb Kdinb
Kxinb Lx Linb Kxinb
Kdina Ld Lina Kdina
Kxina Lx Lina Kxina
Kxqb Lx Lqb Kxqb
Kxoutb Lx Loutb Kxoutb
Kdoutb Ld Loutb Kdoutb
Koutb Loutb Lqb Koutb
Kd2b Ld L2b Kd2b
Kx2b Lx L2b Kx2b
Kd1b Ld L1b Kd1b
Kx1b Lx L1b Kx1b
.ends boost2f2
```

Listing 2.19: `boost2f2` netlist (.cir).

Simulation result

Simulation waveform of a buffer type2 with 2 fan-outs (`boost2f2`) DUT (design under test) with a 4-stage buffer before the input 'a'. Another two 3-stage buffers are placed after the `boost2f2`'s output 'q0' and 'q1'. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (a) (input of the first stage buffers) as 010101011, AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the outputs generated from the DUT `boost2f2` as 010101011 with a random initial output 0. This is because of the meander structure of AQFP circuits. Since the DUT `boost2f2` is placed after a 4-stage buffer, it takes one clock cycle to propagate the data to the output. Input peak-to-peak amplitude is $\pm 5\,\mu A$, AC amplitude is $658\,\mu A$, and DC is set to $843\,\mu A$.

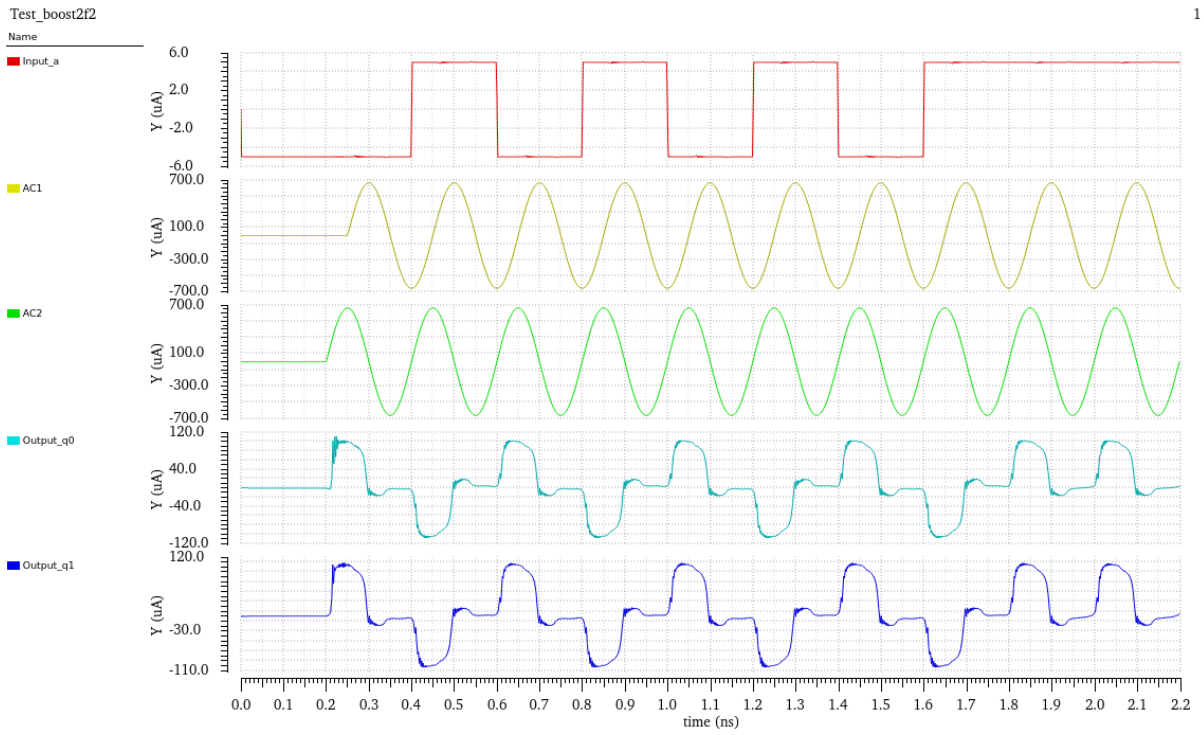


Figure 2.49: `boost2f2` analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2  module boost2f2(a, q0, q1, xin, xout, dcin, dcout);
3    input a;
4    output q0, q1;
5    inout xin, xout, dcin, dcout;
6    reg q0, q1;
7    parameter pul_wid = 100;
8    wire not_a;
9
10   assign not_a = ~a;
11
12   biasDir_b I0(xin, xout, dcin, dcout, gatex);
13
14   initial begin
15
16   $timeformat(-12, 1, "␣ps", 8); // time format
17
18   // output regsiter initialization
19   q0 = 1'bz;
20   q1 = 1'bz;
21   end // initialization
22
23   specify
24     specparam d_clk   = 5;
25     specparam clk_d  = 50;
26
27     $setup(posedge a &&& a, posedge gatex, d_clk);
28     $setup(negedge a &&& not_a, posedge gatex, d_clk);
29     $hold(posedge gatex, negedge a &&& a, clk_d);
30     $hold(posedge gatex, posedge a &&& not_a, clk_d);
31
32   endspecify
33
34   always @(posedge gatex)
```

```
35      begin
36        if (a == 1 | a == 0)
37          begin
38            q0 <= a;
39            q0 <= #pul_wid 1'bz;
40            q1 <= a;
41            q1 <= #pul_wid 1'bz;
42          end
43        else
44          begin
45            q0 <= 1'bx;
46            q0 <= #pul_wid 1'bz;
47            q1 <= 1'bx;
48            q1 <= #pul_wid 1'bz;
49          end
50      end
51
52  endmodule
```

Listing 2.20: `boost2f2` Verilog model code.

Figure 2.50: `spl2` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($a = 0$ and $a = 1$) and clock frequencies.

Table 2.20: `boost2f2` switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'0' (J)	2.22E-22	4.42E-22	1.10E-21	2.17E-21	4.32E-21	1.13E-20	2.94E-20
'1' (J)	2.51E-22	4.66E-22	1.11E-21	2.17E-21	4.38E-21	1.13E-20	2.63E-20

2.1.11 SPL

Splitters are the splitting elements in AQFP logic to increase the fanout during data propagation. An AQFP spliiter is built from buffer and 1-to-n branch cells. The operational principle is to split the output of a buffer through a 1-to-n branch. There are 3 types of splitter cells with different fanouts: splitter 1-to-2 (`spl2`), 1-to-3 (`spl3`) and 1-to-4 (`spl4`). One should notice that the (`spl4`) is designed using a different approach, which is more close to (`boost1`). Instead of reading out the outputs of the buffers in (`boost1`), a (`spl4`) splits the output current from the 2 buffers into four to achieve 4 fan-outs. Here we only present the `spl2` cell as an example. The other one can be found in the deliverables. The branch is introduced in sub-cells.

Symbol

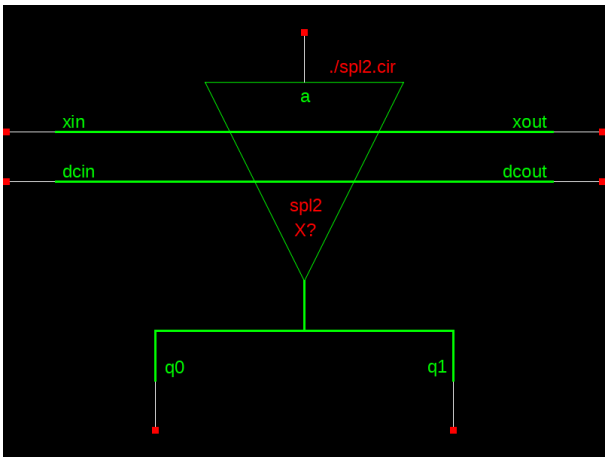


Figure 2.51: `spl2` symbol.

Table 2.21: `spl2` pin list.

Pin	Description
A	data input
XIN	serial clock input
DCIN	dc offset input
Q0	data output
Q1	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

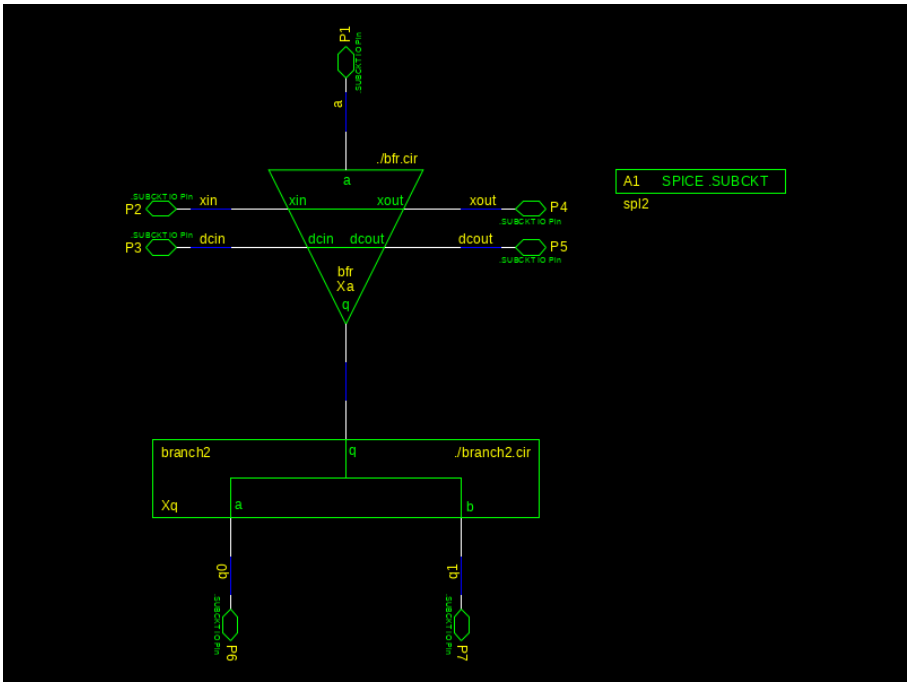


Figure 2.52: `spl2` schematic.

Layout

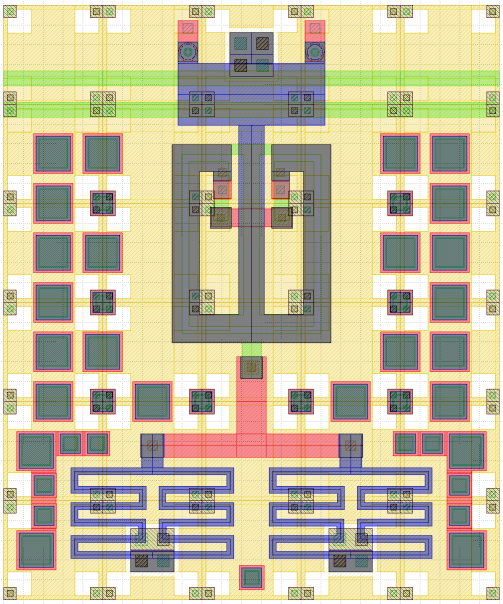


Figure 2.53: `spl2` layout.

Analog model

```
.SUBCKT bfr a xin dcin xout dcout q
Kid Lin Ld Kid
Kiout Lin Lout Kiout
Kix Lin Lx Kix
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.56p
.param Lx=6.53p
.param Ld=6.56p
.param L1=1.56p
.param L2=L1
.param Lq=8.25p
.param Lout=30.55p
.param B1=0.5
.param B2=B1
.param Kxd=0.2916
.param Kx1=-0.2461
.param Kx2=Kx1
.param Kd1=-0.1916
.param Kd2=Kd1
.param Kout=-0.6400
.param Kxout=-8.116E-6
.param Kdout=-2.873E-5
.param Kiout=2.201E-3
.param Kix=-1.145E-4
.param Kid=-1.063E-5

Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends bfr
.SUBCKT bias_pair_11um a b c d
.param L0=3.60p
.param L1=3.60p

L1 c d L1
L0 a b L0
.ends bias_pair_11um
.SUBCKT branch2 a b q
.param La=12.4p
.param Lb=12.4p
.param Lq=0.17p
Lq 1 q Lq
Lb b 1 Lb
La a 1 La
.ends branch2
.SUBCKT spl2 a xin dcin xout dcout q0 q1
Xq q0 q1 1 branch2
X1 xin 4 dcin 5 bias_pair_11um
X2 2 xout 3 dcout bias_pair_11um
Xa a 4 5 2 3 1 bfr
.ends spl2
```

Listing 2.21: `spl2` netlist (.cir).

Simulation result

Simulation waveform of a splitter 1-to-2 (spl2) DUT (design under test) with a 4-stage buffer before the input 'a'. Another two 3-stage buffers are placed after the spl2's output 'q0' and 'q1'. Clock frequency is 5 GHz. Signals from top to bottom: buffer chain input (a) (input of the first stage buffers) as 010101011, AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), and the outputs generated from the DUT spl2 as 010101011 with a random initial output 0. This is because of the meander structure of AQFP circuits. Since the DUT spl2 is placed after a 4-stage buffer, it takes one clock cycle to propagate the data to the output. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

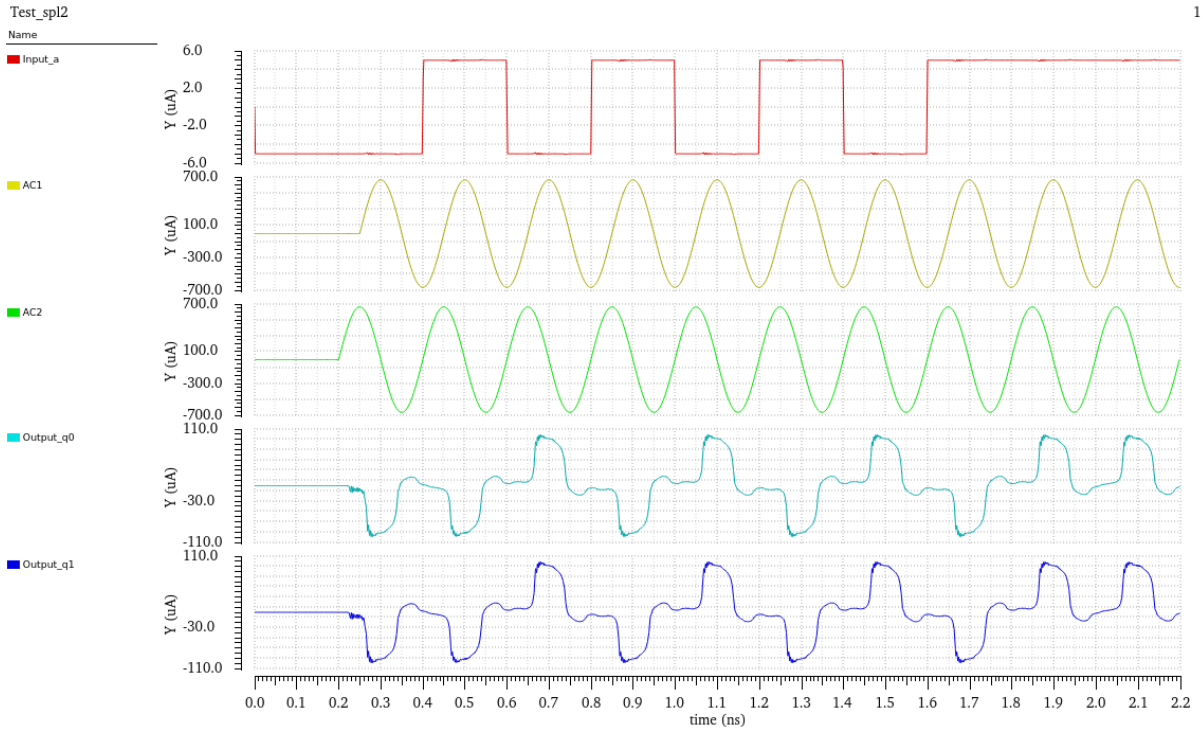


Figure 2.54: spl2 analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2  module spl2(a, q0, q1, xin, xout, dcin, dcout);
3    input a;
4    output q0, q1;
5    inout xin, xout, dcin, dcout;
6    reg q0, q1;
7    parameter pul_wid = 100;
8    wire not_a;
9
10   assign not_a = ~a;
11
12   biasDir_b I0(xin, xout, dcin, dcout, gatex);
13
14   initial begin
15
16   $timeformat(-12, 1, "␣ps", 8); // time format
```

```
17
18    // output regsiter initialization
19    q0 = 1'bz;
20    q1 = 1'bz;
21    end // initialization
22
23    specify
24      specparam d_clk   = 5;
25      specparam clk_d  = 50;
26
27      $setup(posedge a &&& a, posedge gatex, d_clk);
28      $setup(negedge a &&& not_a, posedge gatex, d_clk);
29      $hold(posedge gatex, negedge a &&& a, clk_d);
30      $hold(posedge gatex, posedge a &&& not_a, clk_d);
31
32    endspecify
33
34    always @(posedge gatex)
35      begin
36        if (a == 1 | a == 0)
37          begin
38            q0 <= a;
39            q0 <= #pul_wid 1'bz;
40            q1 <= a;
41            q1 <= #pul_wid 1'bz;
42          end
43        else
44          begin
45            q0 <= 1'bx;
46            q0 <= #pul_wid 1'bz;
47            q1 <= 1'bx;
48            q1 <= #pul_wid 1'bz;
49          end
50      end
51
52  endmodule
```

Listing 2.22: `spl2` Verilog model code.

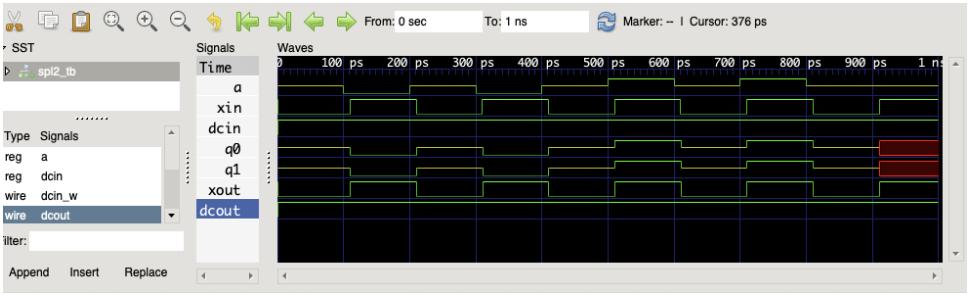


Figure 2.55: `spl2` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($a = 0$ and $a = 1$) and clock frequencies.

Table 2.22: `spl2` switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'0' (J)	3.37E-24	1.13E-23	3.51E-23	7.48E-23	1.55E-22	4.81E-22	2.06E-21
'1' (J)	3.37E-24	1.13E-23	3.03E-23	7.00E-23	1.50E-22	4.76E-22	2.05E-21

2.2 Sequential Cells

2.2.1 QFPL

The quantum-flux-parametron latch (`qfpl`) is a native memory element in the AQFP library. It consists of two `bfr` cells that serve as write gates and a single bi-stable `storage_gate` that is based on the `bfr` core design with no ac excitation line but rather a single dc bias to indefinitely store the state of the `qfpl`. `storage_gate` is introduced in the sub-cells section.

Symbol

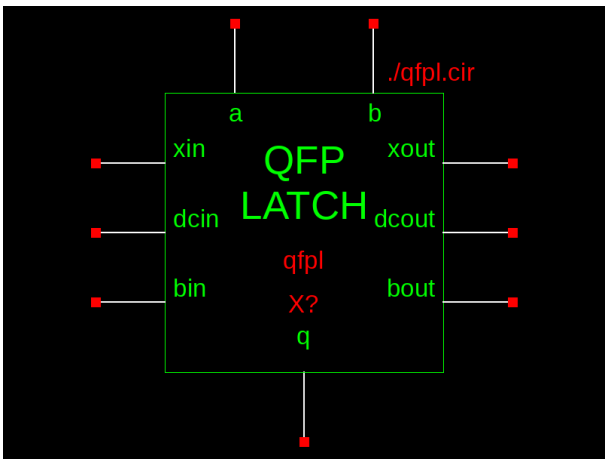


Figure 2.56: `qfpl` symbol.

Table 2.23: `qfpl` pin list.

Pin	Description
A	data input
B	data input
XIN	serial clock input
DCIN	dc offset input
BIN	dc bias input for storage gate
Q	data output
XOUT	serial clock output
DCOUT	dc offset output
BOUT	dc bias input for storage gate

Table 2.24: `qfpl` truth table.

A	B	Q
0	0	Write '0'
0	1	No change
1	0	No change
1	1	Write '1'

Schematic

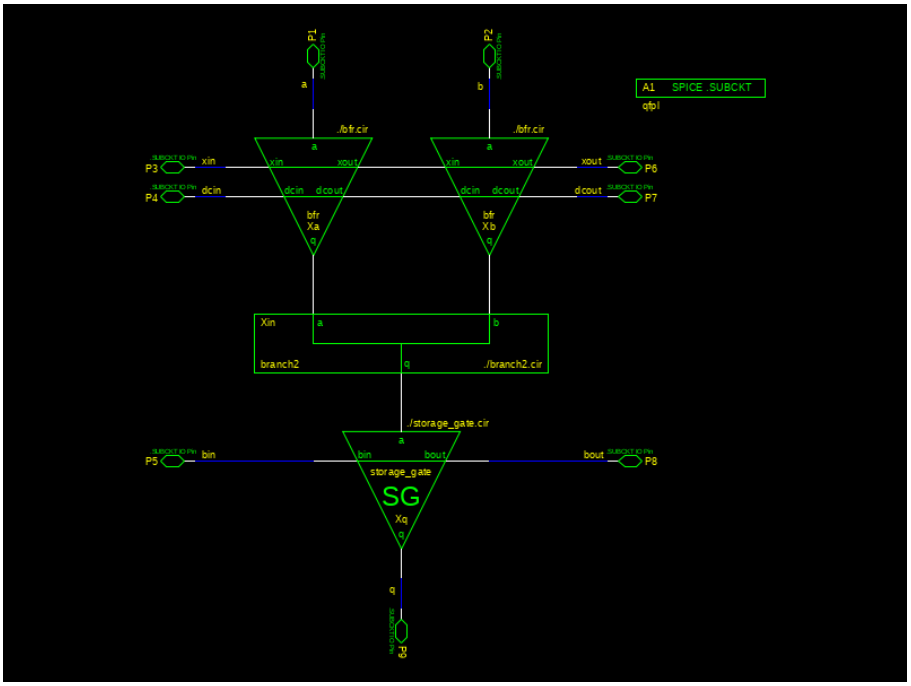


Figure 2.57: `qfpl` schematic.

Layout

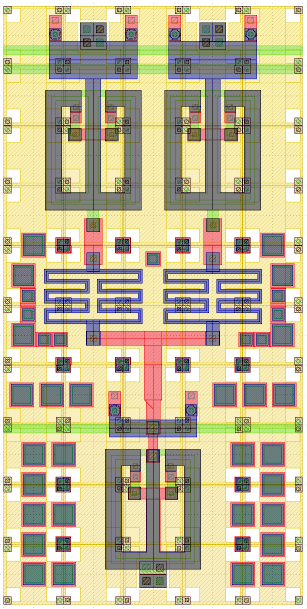


Figure 2.58: `qfpl` layout.

Analog model

```
.SUBCKT storage_gate a bin bout q
Kiout Lin Lout Kiout
Kbout Lb Lout Kbout
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.63p
.param Lb=9.41p
.param L1=2.21p
.param L2=L1
.param Lq=8.31p
.param Lout=30.46p
.param B1=0.5
.param B2=B1
.param Kb1=-0.2842
.param Kb2=Kb1
.param Kout=-0.6421
.param Kbout=-1.861E-6
.param Kiout=5.199E-3
.param Kib=-1.197E-4

Kib Lin Lb Kib
Kout Lout Lq Kout
Kb2 Lb L2 Kb2
Kb1 Lb L1 Kb1
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Lb bin bout Lb
.ends storage_gate
.SUBCKT branch2 a b q
.param La=12.4p
.param Lb=12.4p
```

```
.param Lq=0.17p
Lq 1 q Lq
Lb b 1 Lb
La a 1 La
.ends branch2
.SUBCKT bfr a xin dcin xout dcout q
Kid Lin Ld Kid
Kiout Lin Lout Kiout
Kix Lin Lx Kix
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.56p
.param Lx=6.53p
.param Ld=6.56p
.param L1=1.56p
.param L2=L1
.param Lq=8.25p
.param Lout=30.55p
.param B1=0.5
.param B2=B1
.param Kxd=0.2916
.param Kx1=-0.2461
.param Kx2=Kx1
.param Kd1=-0.1916
.param Kd2=Kd1
.param Kout=-0.6400
.param Kxout=-8.116E-6
.param Kdout=-2.873E-5
.param Kiout=2.201E-3
.param Kix=-1.145E-4
.param Kid=-1.063E-5

Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends bfr
.SUBCKT qfpl a b xin dcin bin xout dcout bout q
Xb b 4 5 xout dcout 3 bfr
Xa a xin dcin 4 5 2 bfr
Xin 2 3 1 branch2
Xq 1 bin bout q storage_gate
.ends qfpl
```

Listing 2.23: `qfpl` netlist (.cir).

Simulation result

Simulation waveform of the `qfpl`. A 4-stage buffer chain is connected both input A and B. Each input is applied a test pattern with a peak-to-peak amplitude is ±5 µA. AC amplitude is 664 µA at 5 GHz, and DC is set to 862 µA. Input A has a pattern of 010110001100 and B has a pattern of 001101001100. This results in an operation pattern of write '0', hold, hold, write '1', hold, hold, write '0', write '0', write '1', write '1', write '0', write '0'. The resulting data output is 000111001100 as shown in the simulation waveform.

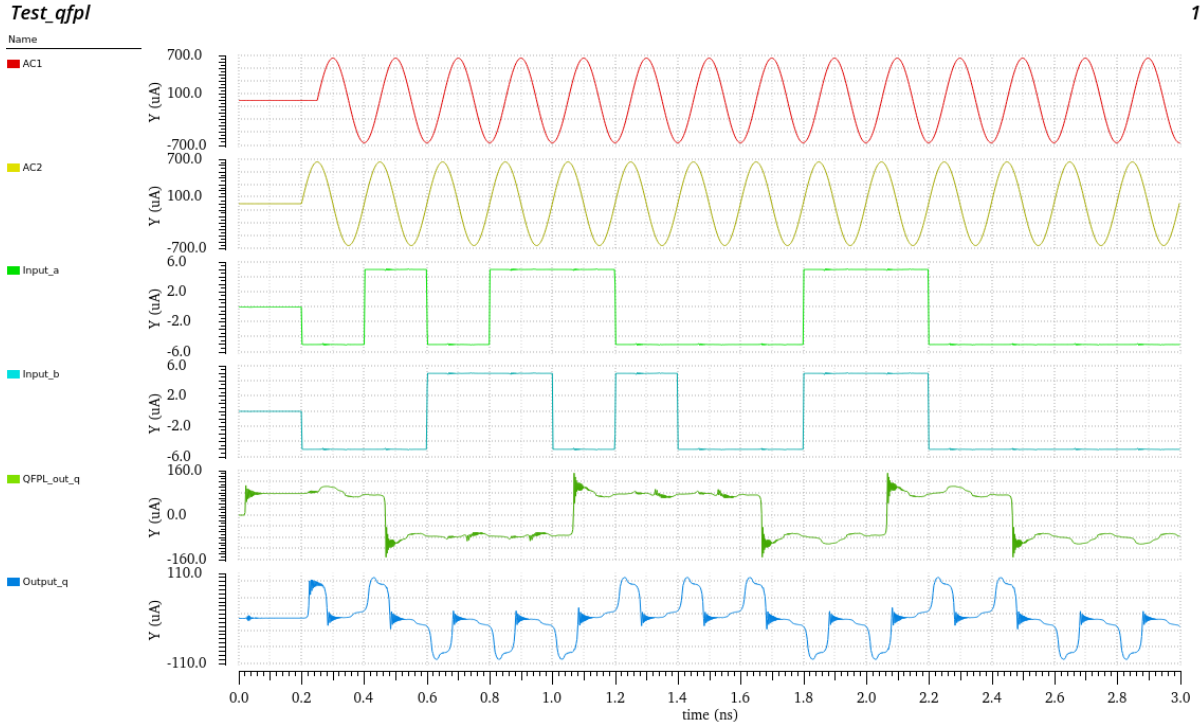


Figure 2.59: `qfpl` analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2
3  module qfpl(a, b, q, xin, xout, dcin, dcout, bin, bout);
4
5    input a, b;
6    inout xin, xout, dcin, dcout, bin, bout;
7    output q;
8
9    wire not_a, not_b;
10   reg q;
11
12   //use DC bias bidirection model for bin/bout
13   biasDC #(8) I1(bin, bout);
14   biasDir_b I2(xin, xout, dcin, dcout, gatex);
15
16   assign not_a = !a;
17   assign not_b = !b;
18
19   initial begin
20     $timeformat(-12, 1, "␣ps", 8); // time format
21     q = 1'bz;
22   end
23
24   specify
25     specparam d_clk   = 5;
26     specparam clk_d   = 50;
27
28     $setup(posedge a &&& a, posedge gatex, d_clk);
29     $setup(negedge a &&& not_a, posedge gatex, d_clk);
30     $hold(posedge gatex, negedge a &&& a, clk_d);
31     $hold(posedge gatex, posedge a &&& not_a, clk_d);
32
33     $setup(posedge b &&& b, posedge gatex, d_clk);
34     $setup(negedge b &&& not_b, posedge gatex, d_clk);
```

```
35      $hold(posedge gatex, negedge b &&& b, clk_d);
36      $hold(posedge gatex, posedge b &&& not_b, clk_d);
37    endspecify
38
39    always @(posedge gatex) begin
40      if (a==b && bin==1'b1) begin
41        //write case (using logical compare == so only valid logic values pass)
42        q <= a;
43
44        //no reset in QFPL, state is held indefinitely unlike other AQFP cells
45
46      end else if (a!=b && bin==1'b1) begin
47        //keep state
48      end else begin
49        //Possible errors:
50        //If QFPL bias is not initialized to '1' before operation
51        //then the QFPL should not work. (Detects misconnect of bias).
52
53        //If data are not valid (still Z or X), then a timing error occurred
54        //because of logical equality even 'z'=='z' or 'x'=='x' will not pass)
55
56        q <= 1'bx;
57      end
58    end
59  endmodule
```

Listing 2.24: `qfpl` Verilog model code.

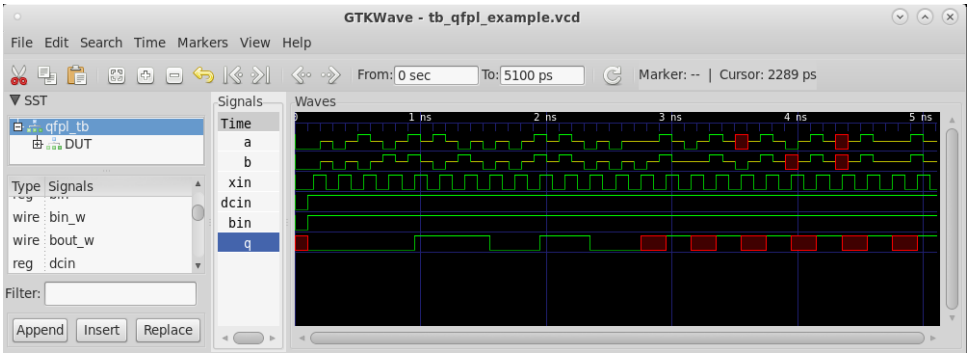


Figure 2.60: `qfpl` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

Switching energy

Different energy consumption results based on different data input patterns ($ab = 00$, $ab = 01$, $ab = 10$ and $ab = 11$) and clock frequencies.

Table 2.25: `qfpl` switching energy table.

Input Logic	Clock Rate (GHz)						
	0.1	0.2	0.5	1	2	5	10
'00' (J)	1.00E-20	1.00E-20	1.01E-20	1.02E-20	1.05E-20	1.14E-20	1.48E-20
'01' (J)	2.28E-23	4.64E-23	1.17E-22	2.36E-22	4.76E-22	1.44E-21	4.80E-21
'10' (J)	2.28E-23	4.64E-23	1.17E-22	2.36E-22	4.76E-22	1.44E-21	4.80E-21
'11' (J)	1.00E-20	1.00E-20	1.01E-20	1.02E-20	1.05E-20	1.14E-20	1.48E-20

2.2.2 NDRO_QFPL

There are two design approaches to achieve an AQFP non-destructive read-out (ndro): rewriting the boolean logic function of a `qfpl` by adding proper combinational logic (`ndro_qfpl`), or using pure combination logic gates with a feedback loop (`ndro_fb`)

Symbol

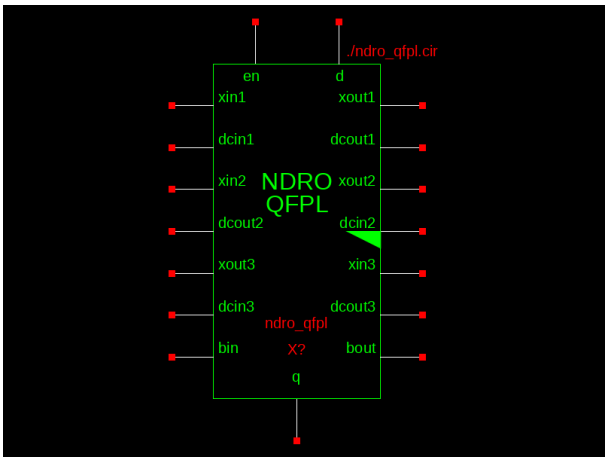


Figure 2.61: `ndro_qfpl` symbol.

Table 2.26: `ndro_qfpl` pin list.

Pin	Description
EN	data input
D	data input
XIN1	serial clock input
XIN2	serial clock input
XIN3	serial clock input
DCIN1	dc offset input
DCIN2	dc offset input
DCIN3	dc offset input
BIN	dc bias input for storage gate
Q	data output
XOUT1	serial clock output
XOUT2	serial clock output
XOUT3	serial clock output
DCOUT1	dc offset output
DCOUT2	dc offset output
DCOUT3	dc offset output
BOUT	dc bias input for storage gate

Table 2.27: `ndro_qfpl` truth table.

E	D	Q
0	0	No Change
0	1	No change
1	0	Write '0'
1	1	Write '1'

Schematic

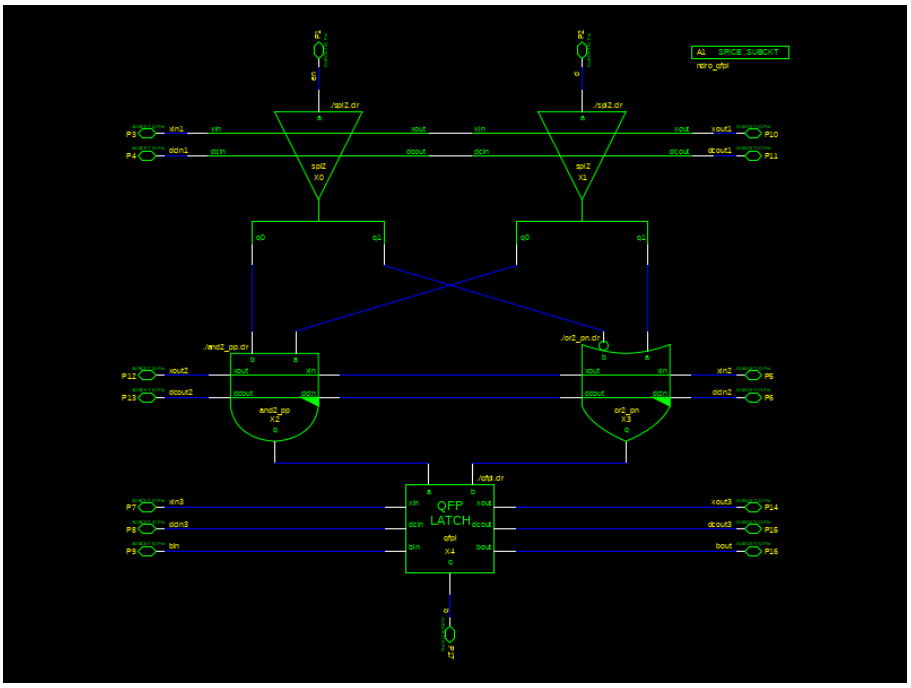


Figure 2.62: `ndro_qfpl` schematic.

Layout

This is a multi-phase macro cell so we cannot fix the location of the excitation lines as it depends on how the `ndro_qfpl` fits in a larger circuit design. Thus there is no fixed layout, as it should be placed dynamically on a case-by-case basis.

Analog model

```
.SUBCKT storage_gate a bin bout q
Kiout Lin Lout Kiout
Kbout Lb Lout Kbout
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.63p
.param Lb=9.41p
.param L1=2.21p
.param L2=L1
.param Lq=8.31p
.param Lout=30.46p
.param B1=0.5
.param B2=B1
```

```
.param Kb1=-0.2842
.param Kb2=Kb1
.param Kout=-0.6421
.param Kbout=-1.861E-6
.param Kiout=5.199E-3
.param Kib=-1.197E-4

Kib Lin Lb Kib
Kout Lout Lq Kout
Kb2 Lb L2 Kb2
Kb1 Lb L1 Kb1
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Lb bin bout Lb
.ends storage_gate
.SUBCKT branch2 a b q
.param La=12.4p
.param Lb=12.4p
.param Lq=0.17p
Lq 1 q Lq
Lb b 1 Lb
La a 1 La
.ends branch2
.SUBCKT bfr a xin dcin xout dcout q
Kid Lin Ld Kid
Kiout Lin Lout Kiout
Kix Lin Lx Kix
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.56p
.param Lx=6.53p
.param Ld=6.56p
.param L1=1.56p
.param L2=L1
.param Lq=8.25p
.param Lout=30.55p
.param B1=0.5
.param B2=B1
.param Kxd=0.2916
.param Kx1=-0.2461
.param Kx2=Kx1
.param Kd1=-0.1916
.param Kd2=Kd1
.param Kout=-0.6400
.param Kxout=-8.116E-6
.param Kdout=-2.873E-5
.param Kiout=2.201E-3
.param Kix=-1.145E-4
.param Kid=-1.063E-5

Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
```

```
Lx xin xout Lx
.ends bfr
.SUBCKT qfpl a b xin dcin bin xout dcout bout q
Xb b 4 5 xout dcout 3 bfr
Xa a xin dcin 4 5 2 bfr
Xin 2 3 1 branch2
Xq 1 bin bout q storage_gate
.ends qfpl
.SUBCKT bias_pair_11um a c b d
.param L0=3.60p
.param L1=3.60p

L1 c d L1
L0 a b L0
.ends bias_pair_11um
.SUBCKT spl2 a xin dcin xout dcout q0 q1
Xq q0 q1 1 branch2
X1 xin dcin 4 5 bias_pair_11um
X2 2 3 xout dcout bias_pair_11um
Xa a 4 5 2 3 1 bfr
.ends spl2
.SUBCKT branch3 a b c q
.param La=13.60p
.param Lb=10.30p
.param Lc=13.60p
.param Lq=0.28p
Lq 1 q Lq
Lc c 1 Lc
Lb b 1 Lb
La a 1 La
.ends branch3
.SUBCKT const0 xin dcin xout dcout q
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lx=6.62p
.param Ld=6.60p
.param L1=1.61p
.param L2=1.76p
.param Lq=8.19p
.param Lout=30.65p
.param B1=0.5
.param B2=B1
.param Kxd=0.2972
.param Kx1=-0.2265
.param Kx2=-0.2602
.param Kd1=-0.1688
.param Kd2=-0.2088
.param Kout=-0.6452
.param Kxout=-2.545E-4
.param Kdout=-5.447E-4
Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends const0
.SUBCKT and2_pp a b xin dcin xout dcout q
X0 2 3 5 6 7 const0
Xq 1 7 4 q branch3
Xc b 5 6 xout dcout 4 bfr
```

```
153   Xa a xin dcin 2 3 1 bfr
154   .ends and2_pp
155   .SUBCKT const1 xin dcin xout dcout q
156   X0 xout dcout xin dcin q const0
157   .ends const1
158   .SUBCKT inv a xin dcin xout dcout q
159   Kid Lin Ld Kid
160   Kiout Lin Lout Kiout
161   Kix Lin Lx Kix
162   .model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
163   .param Lin=1.56p
164   .param Lx=6.54p
165   .param Ld=6.57p
166   .param L1=1.56p
167   .param L2=L1
168   .param Lq=7.14p
169   .param Lout=30.24p
170   .param B1=0.5
171   .param B2=B1
172   .param Kxd=0.2912
173   .param Kx1=-0.2460
174   .param Kx2=Kx1
175   .param Kd1=-0.1909
176   .param Kd2=Kd1
177   .param Kout=0.5917
178   .param Kxout=1.335E-5
179   .param Kdout=2.443E-5
180   .param Kiout=1.172E-3
181   .param Kix=-2.712E-4
182   .param Kid=2.605E-4
183
184   Kxout Lx Lout Kxout
185   Kdout Ld Lout Kdout
186   Kout Lout Lq Kout
187   Kd2 Ld L2 Kd2
188   Kx2 Lx L2 Kx2
189   Kd1 Ld L1 Kd1
190   Kx1 Lx L1 Kx1
191   Kxd Lx Ld Kxd
192   Lin a 2 Lin
193   B2 3 0 B2 jjmit  area=B2
194   B1 1 0 B1 jjmit  area=B1
195   Lout 0 q Lout
196   Lq 2 0 Lq
197   L2 3 2 L2
198   L1 2 1 L1
199   Ld dcin dcout Ld
200   Lx xin xout Lx
201   .ends inv
202   .SUBCKT or2_pn a b xin dcin xout dcout q
203   Xa a xin dcin 6 7 1 bfr
204   Xc b 4 5 xout dcout 3 inv
205   X1 6 7 4 5 2 const1
206   Xq 1 2 3 q branch3
207   .ends or2_pn
208   .SUBCKT ndro_qfpl en d xin1 dcin1 xin2 dcin2 xin3 dcin3 bin xout1 dcout1 xout2 dcout2
          ↪ xout3 dcout3 bout q
209   X3 8 6 xin2 dcin2 9 10 2 or2_pn
210   X2 7 5 9 10 xout2 dcout2 1 and2_pp
211   X1 d 3 4 xout1 dcout1 7 8 spl2
212   X0 en xin1 dcin1 3 4 5 6 spl2
213   X4 1 2 xin3 dcin3 bin xout3 dcout3 bout q qfpl
214   .ends ndro_qfpl
```

Listing 2.25: `ndro_qfpl` netlist (.cir).

Simulation result

Simulation waveform of the ndro_qfpl. A 4-stage buffer chain is connected both input EN and D. Each input is applied a test pattern with a peak-to-peak amplitude is ±5 µA. AC amplitude is 664 µA at 5 GHz, and DC is set to 862 µA. Input EN has a pattern of 10001000100 and D has a pattern of 01011010010. This results in an operation pattern of write '0', hold, hold, hold, write '1', hold, hold, hold, write '0', hold, hold. The resulting data output is 00001111000 as shown in the simulation waveform.

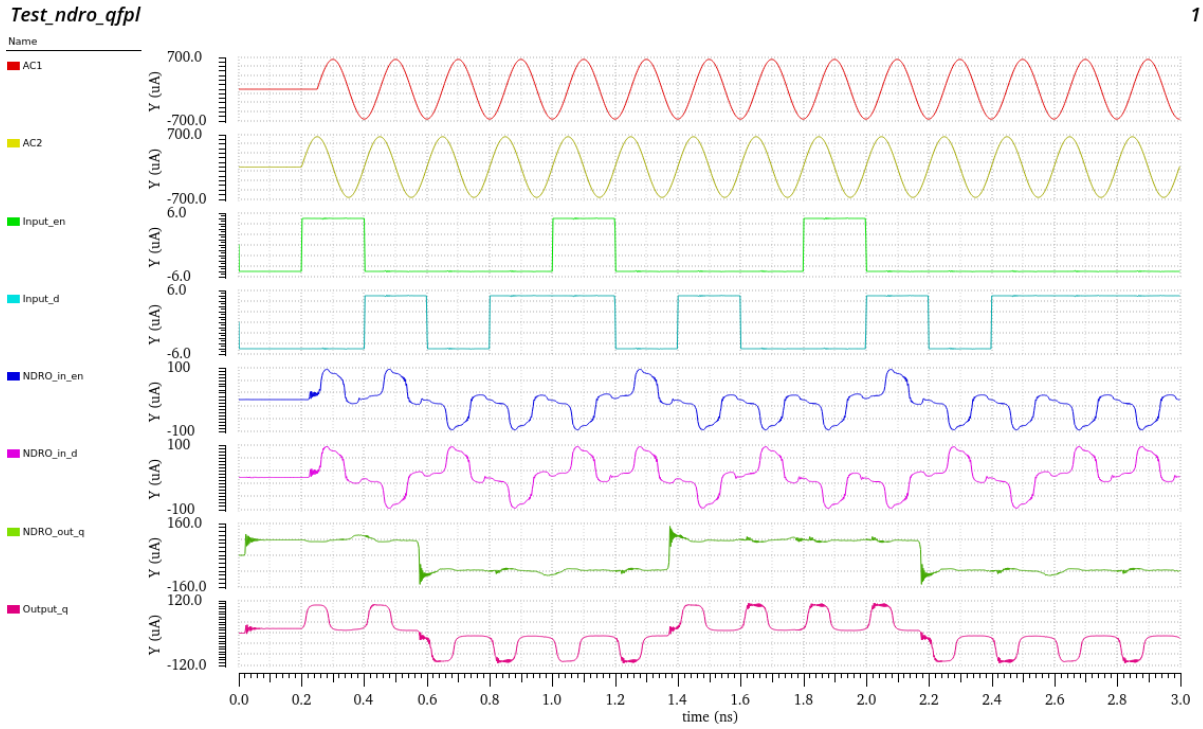


Figure 2.63: ndro_qfpl analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2
3  module ndro_qfpl(en, d, xin1, xin2, xin3, dcin1, dcin2, dcin3, bin,
4                   q, xout1, xout2, xout3, dcout1, dcout2, dcout3, bout);
5
6    input en, d;
7    inout xin1, xin2, xin3, dcin1, dcin2, dcin3, bin;
8    inout xout1, xout2, xout3, dcout1, dcout2, dcout3, bout;
9    output q;
10   wire s0_q0, s0_q1, s1_q0, s1_q1, and_q, or_q;
11   wire xph1, xph2, dcph1, dcph2;
12
13   //call submodule
14   spl2    s0(.a(en), .xin(xin1), .dcin(dcin1), .q0(s0_q0), .q1(s0_q1), .xout(xph1), .dcout
         ↪ (dcph1));
15   spl2    s1(.a(d), .xin(xph1), .dcin(dcph1), .q0(s1_q0), .q1(s1_q1), .xout(xout1), .dcout
         ↪ (dcout1));
16   and2_pp a0(.a(s1_q0), .b(s0_q0), .xin(xph2), .dcin(dcph2), .q(and_q), .xout(xout2), .
         ↪ dcout(dcout2));
17   or2_pn  o0(.a(s1_q1), .b(s0_q1), .xin(xin2), .dcin(dcin2), .q(or_q), .xout(xph2), .dcout
         ↪ (dcph2));
18   qfpl    q0(.a(and_q), .b(or_q), .xin(xin3), .dcin(dcin3), .bin(bin), .q(q) .xout(xout3),
         ↪  .dcout(dcout3), .bout(bout));
19
20 endmodule
```

Listing 2.26: `ndro_qfpl` Verilog model code.

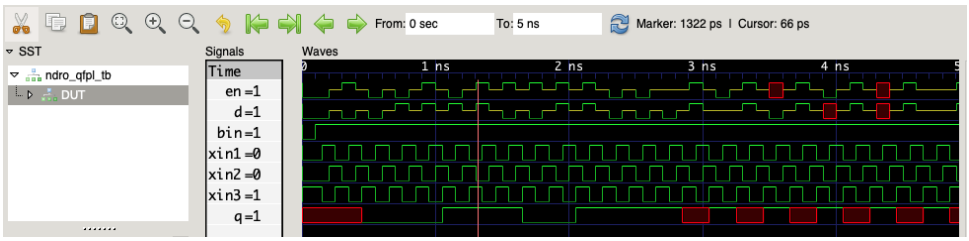


Figure 2.64: `ndro_qfpl` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

2.2.3 NDRO_FB

Non-destructive read-out with feedback (ndro_bf), previously named as 'D-Flip-Flop (DFF)', is considered as a sequential logic element built from the developed AQFP combinational logic cells. Here we present the symbol view, block diagram (schematic), netlist and simulation results from a test circuit.

Symbol

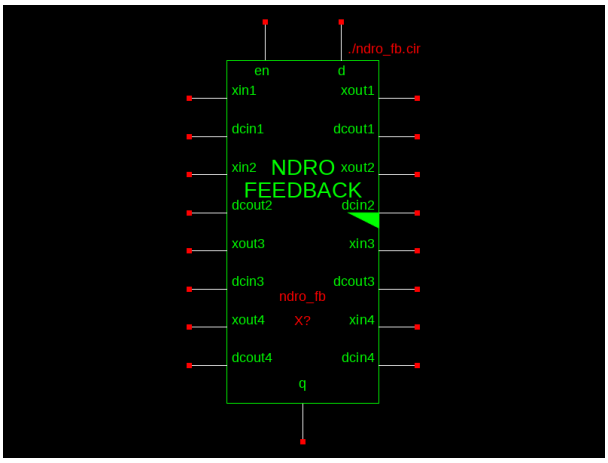


Figure 2.65: ndro_fb symbol.

Table 2.28: ndro_fb pin list.

Pin	Description
E	data input
D	data input
XIN1	serial clock input
XIN2	serial clock input
XIN3	serial clock input
XIN4	serial clock input
DCIN1	dc offset input
DCIN2	dc offset input
DCIN3	dc offset input
DCIN4	dc offset input
Q	data output
XOUT1	serial clock output
XOUT2	serial clock output
XOUT3	serial clock output
XOUT4	serial clock output
DCOUT1	dc offset output
DCOUT2	dc offset output
DCOUT3	dc offset output
DCOUT4	dc offset output

Schematic

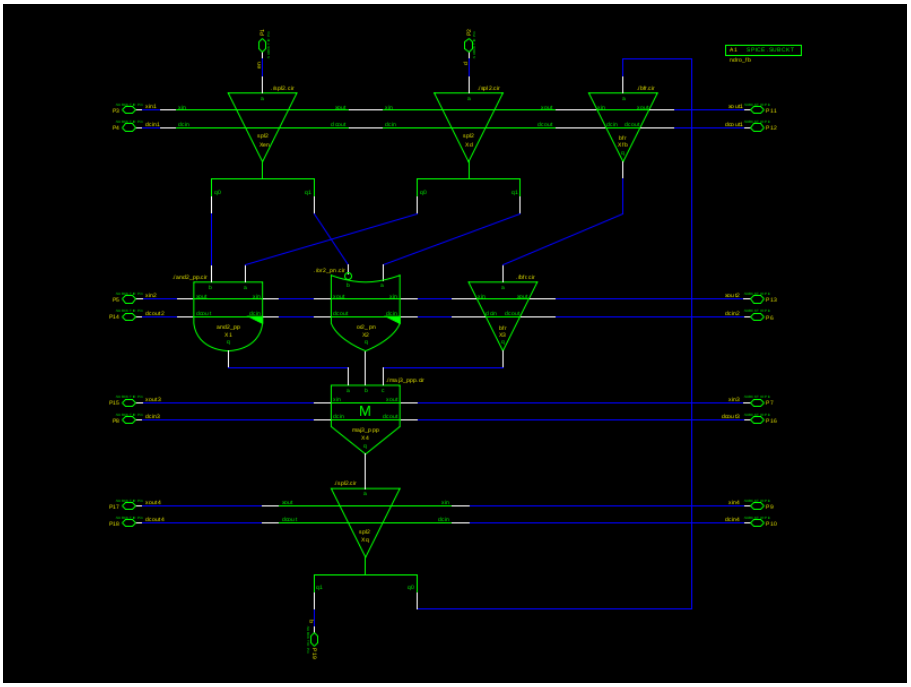


Figure 2.66: `ndro_fb` schematic.

Layout

This is a multi-phase macro cell so we cannot fix the location of the excitation lines as it depends on how the `ndro_fb` fits in a larger circuit design. Thus there is no fixed layout, as it should be placed dynamically on a case-by-case basis.

Analog model

```
.SUBCKT bfr a xin dcin xout dcout q
Kid Lin Ld Kid
Kiout Lin Lout Kiout
Kix Lin Lx Kix
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.56p
.param Lx=6.53p
.param Ld=6.56p
.param L1=1.56p
.param L2=L1
.param Lq=8.25p
.param Lout=30.55p
.param B1=0.5
.param B2=B1
.param Kxd=0.2916
.param Kx1=-0.2461
.param Kx2=Kx1
.param Kd1=-0.1916
.param Kd2=Kd1
.param Kout=-0.6400
.param Kxout=-8.116E-6
.param Kdout=-2.873E-5
.param Kiout=2.201E-3
.param Kix=-1.145E-4
```

```
.param Kid=-1.063E-5

Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends bfr
.SUBCKT branch3 a b c q
.param La=13.60p
.param Lb=10.30p
.param Lc=13.60p
.param Lq=0.28p
Lq 1 q Lq
Lc c 1 Lc
Lb b 1 Lb
La a 1 La
.ends branch3
.SUBCKT const0 xin dcin xout dcout q
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lx=6.62p
.param Ld=6.60p
.param L1=1.61p
.param L2=1.76p
.param Lq=8.19p
.param Lout=30.65p
.param B1=0.5
.param B2=B1
.param Kxd=0.2972
.param Kx1=-0.2265
.param Kx2=-0.2602
.param Kd1=-0.1688
.param Kd2=-0.2088
.param Kout=-0.6452
.param Kxout=-2.545E-4
.param Kdout=-5.447E-4
Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends const0
.SUBCKT and2_pp a b xin dcin xout dcout q
X0 2 3 5 6 7 const0
Xq 1 7 4 q branch3
Xc b 5 6 xout dcout 4 bfr
Xa a xin dcin 2 3 1 bfr
```

```
.ends and2_pp
.SUBCKT bias_pair_11um a c b d
.param L0=3.60p
.param L1=3.60p

L1 c d L1
L0 a b L0
.ends bias_pair_11um
.SUBCKT branch2 a b q
.param La=12.4p
.param Lb=12.4p
.param Lq=0.17p
Lq 1 q Lq
Lb b 1 Lb
La a 1 La
.ends branch2
.SUBCKT spl2 a xin dcin xout dcout q0 q1
Xq q0 q1 1 branch2
X1 xin dcin 4 5 bias_pair_11um
X2 2 3 xout dcout bias_pair_11um
Xa a 4 5 2 3 1 bfr
.ends spl2
.SUBCKT maj3_ppp a b c xin dcin xout dcout q
Xq 1 4 7 q branch3
Xc c 5 6 xout dcout 7 bfr
Xb b 2 3 5 6 4 bfr
Xa a xin dcin 2 3 1 bfr
.ends maj3_ppp
.SUBCKT const1 xin dcin xout dcout q
X0 xout dcout xin dcin q const0
.ends const1
.SUBCKT inv a xin dcin xout dcout q
Kid Lin Ld Kid
Kiout Lin Lout Kiout
Kix Lin Lx Kix
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.56p
.param Lx=6.54p
.param Ld=6.57p
.param L1=1.56p
.param L2=L1
.param Lq=7.14p
.param Lout=30.24p
.param B1=0.5
.param B2=B1
.param Kxd=0.2912
.param Kx1=-0.2460
.param Kx2=Kx1
.param Kd1=-0.1909
.param Kd2=Kd1
.param Kout=0.5917
.param Kxout=1.335E-5
.param Kdout=2.443E-5
.param Kiout=1.172E-3
.param Kix=-2.712E-4
.param Kid=2.605E-4

Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
```

```
165  L2 3 2 L2
166  L1 2 1 L1
167  Ld dcin dcout Ld
168  Lx xin xout Lx
169  .ends inv
170  .SUBCKT or2_pn a b xin dcin xout dcout q
171  Xa a xin dcin 6 7 1 bfr
172  Xc b 4 5 xout dcout 3 inv
173  X1 6 7 4 5 2 const1
174  Xq 1 2 3 q branch3
175  .ends or2_pn
176  .SUBCKT ndro_fb en d xin1 dcin1 xin2 dcin2 xin3 dcin3 xin4 dcin4 xout1 dcout1 xout2
         ↪ dcout2 xout3 dcout3 xout4 dcout4 q
177  X2 8 4 14 15 9 10 18 or2_pn
178  X4 11 18 13 xout3 dcin3 xin3 dcout3 17 maj3_ppp
179  Xq 17 xin4 dcin4 xout4 dcout4 16 q spl2
180  Xfb 16 5 6 xout1 dcout1 12 bfr
181  X3 12 14 15 xout2 dcin2 13 bfr
182  X1 7 3 9 10 xin2 dcout2 11 and2_pp
183  Xd d 1 2 5 6 7 8 spl2
184  Xen en xin1 dcin1 1 2 3 4 spl2
185  .ends ndro_fb
```

Listing 2.27: `ndro_fb` netlist (.cir).

Simulation result

Simulation waveform of a `ndro_fb` DUT (design under test) with 4-stage buffer before the D input and before the EN input. Another 4-stage buffer is placed after the `ndro_fb`'s Q output. Clock frequency is 5 GHz. Signals from top to bottom: AC source 1 (generates phase 1 and 3), AC source 2 (generates phase 2 and 4), the buffer chain input D and EN, output EN from buffer just before `ndro_fb`, output D from buffer just before `ndro_fb` and the output Q from buffer just after `ndro_fb`. Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

When textttndro_fb is set to 0 (D =1), data is toggled while E=0 to show data remains as 0, whereas when textttndro_fb is set to 1 (D=0), data is toggled while E=0 to show data remains as 1, when textttndro_fb is set to 0 again, data is toggled while E=0 to show data remains as 0.

As shown in the waveform, where D = NDRO_in_en = (1)10001001000 and E = NDRO_in_d = (0)010110100101, the output Q is (1)0000011110, correspondingly. The numbers marked in brackets represent the initial random outputs given by the meander structure of AQFP circuit, it takes one clock cycle to propagate data from the input buffer to the last-stage buffer.

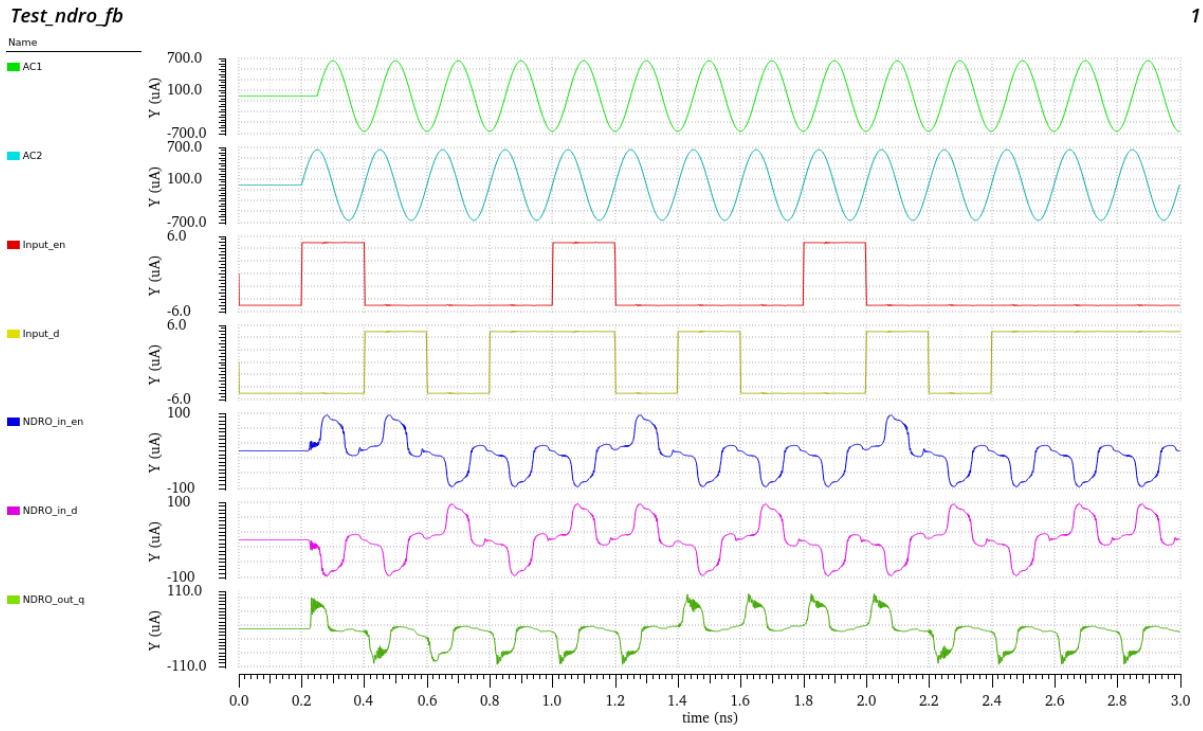


Figure 2.67: `ndro_fb` analog waveform.

Digital model

```
1  `timescale 1ps/10fs
2
3  module ndro_fb(en, d, xin1, xin2, xin3, xin4, dcin1, dcin2, dcin3, dcin4,
4                 q, xout1, xout2, xout3, dcout1, dcout2, dcout3, dcout4);
5
6    input en, d;
7    inout xin1, xin2, xin3, xin4, dcin1, dcin2, dcin3, dcin4;
8    inout xout1, xout2, xout3, xout4, dcout1, dcout2, dcout3, dcout4;
9    output q;
10   wire s0_q0, s0_q1, s1_q0, s1_q1;
11   wire b0_a, b0_q, and_q, or_q, maj_q;
12   wire xph1_0, xph1_1, xph2_0, xph2_1, dcph1_0, dcph1_1, dcph2_0, dcph2_1;
13
14   //call submodule
15   spl2     s0(.a(en), .xin(xin1),   .dcin(dcin1),   .q0(s0_q0), .q1(s0_q1), .xout(xph1_0),
         ↪  .dcout(dcph1_0));
16   spl2     s1(.a(d),  .xin(xph1_0), .dcin(dcph1_0), .q0(s1_q0), .q1(s1_q1), .xout(xph1_1),
         ↪  .dcout(dcph1_1));
17   bfr      b0(.a(b0_a), .xin(xph1_1), .dcin(dcph1_1), .q(b0_q), .xout(xout1), .dcout(
         ↪ dcout1));
18   bfr      b1(.a(b0_q), .xin(xph2_1), .dcin(dcph2_1), .q(b1_q), .xout(xout2), .dcout(dcin2
         ↪ ));
19   and2_pp  a0(.a(s1_q0), .b(s0_q0), .xin(xph2_0), .dcin(dcph2_0), .q(and_q), .xout(xin2),
         ↪ .dcout(dcout2));
20   or2_pn   o0(.a(s1_q1), .b(s0_q1), .xin(xph2_1), .dcin(dcph2_1), .q(or_q), .xout(xph2_0),
         ↪  .dcout(dcph2_0));
21   maj3_ppp m0(.a(and_q), .b(or_q), .c(b1_q), .xin(xin3), .dcin(dcin3), .q(maj_q), .xout(
         ↪ xout3), .dcout(dcout3));
22   spl2     s2(.a(maj_q), .xin(xin4), .dcin(dcin4), .q0(q), .q1(b0_a), .xout(xout4), .dcout
         ↪ (dcout4));
23
24 endmodule
```

Listing 2.28: `ndro_fb` Verilog model code.

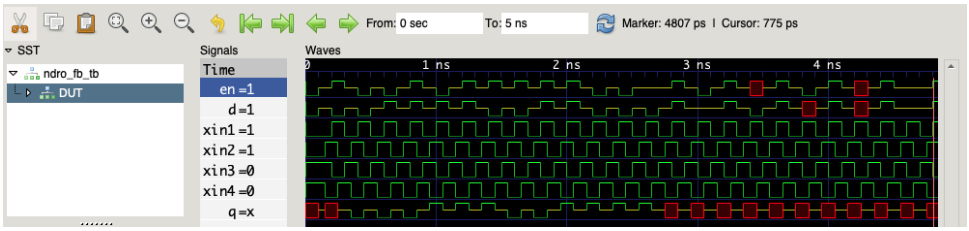


Figure 2.68: `ndro_fb` digital waveform. HDL '1': AQFP '1'; HDL '0': AQFP '0'; HDL 'z': inactive; HDL 'x': error.

2.3 Interconnect slices

The interconnect of AQFP logic is composed of passive transmission lines (PTLs). PTLs are stripline structures where a signal line has a dedicated ground plane above and below it. RSFQ logic uses striplines to ballistically transport an SFQ pulse from RSFQ cell to another through active transmitter-receiver pairs. AQFP logic uses PTLs to send positive or negative current pulses and the logic cells can connect directly to PTLs without using transmitter-receiver circuits.

The active logic area is dedicated to layers above the M4 ground plane whereas the routing interconnecting exists below the M4 ground plane. The AQFP interconnect slices are 10 µm × 10 µm, except for the bias pair slices which are 10 µm × 20 µm. Table 2.29 summarizes the different types of interconnect cells that are available and Fig. 2.70 shows their corresponding layouts. Interconnect slices for wires on M5 are included in Table 2.29, but are only included in the AQFP cell library as additional interconnect slices for manual circuit design with signal lines on M5.

Table 2.29: Summary of interconnect slices. Each slice is designated with bidirectional ports a-to-b and/or c-to-d. The signal layers for corresponding to those ports are listed. The shielding ground layers are also listed.

Name	Signal		Ground	Purpose
	a-b	c-d		
tr_bias_AC_c_tl	M7	M7	M4	Bias pair with AC corner with top and left ports.
tr_bias_AC_c_tr	M7	M7	M4	Bias pair with AC corner with top and right ports.
tr_bias_ACxDC_bl	M7	M7	M4	Bias pair with AC crossing DC with bottom and left ports.
tr_bias_ACxDC_br	M7	M7	M4	Bias pair with AC crossing DC with bottom and right ports.
tr_bias_DC_c_bl	M7	M7	M4	Bias pair with DC corner with bottom and left ports.
tr_bias_DC_c_br	M7	M7	M4	Bias pair with DC corner with bottom and right ports.
tr_bias_DCxAC_bl	M7	M7	M4	Bias pair with DC crossing AC with top and left ports.
tr_bias_DCxAC_br	M7	M7	M4	Bias pair with DC crossing AC with top and right ports.
tr_bias_pair_10um	M7	M7	M4	Bias AC/DC parallel pair with length of 10um.
tr_bias_xAC_tb	M7	M7	M4	Bias AC crossing with top and bottom ports.
tr_bias_xDC_tb	M7	M7	M4	Bias DC crossing with top and bottom ports.
tr_blank	N/A	N/A	M4	Blank track block template.
tr_conn_M1M3	M1 M3	N/A	M0 M2 M4	Connection between M1 and M3.
tr_conn_M1M6	M1 M6	N/A	M0 M2 M4	Connection between M1 and M6.

Continued on the next page...

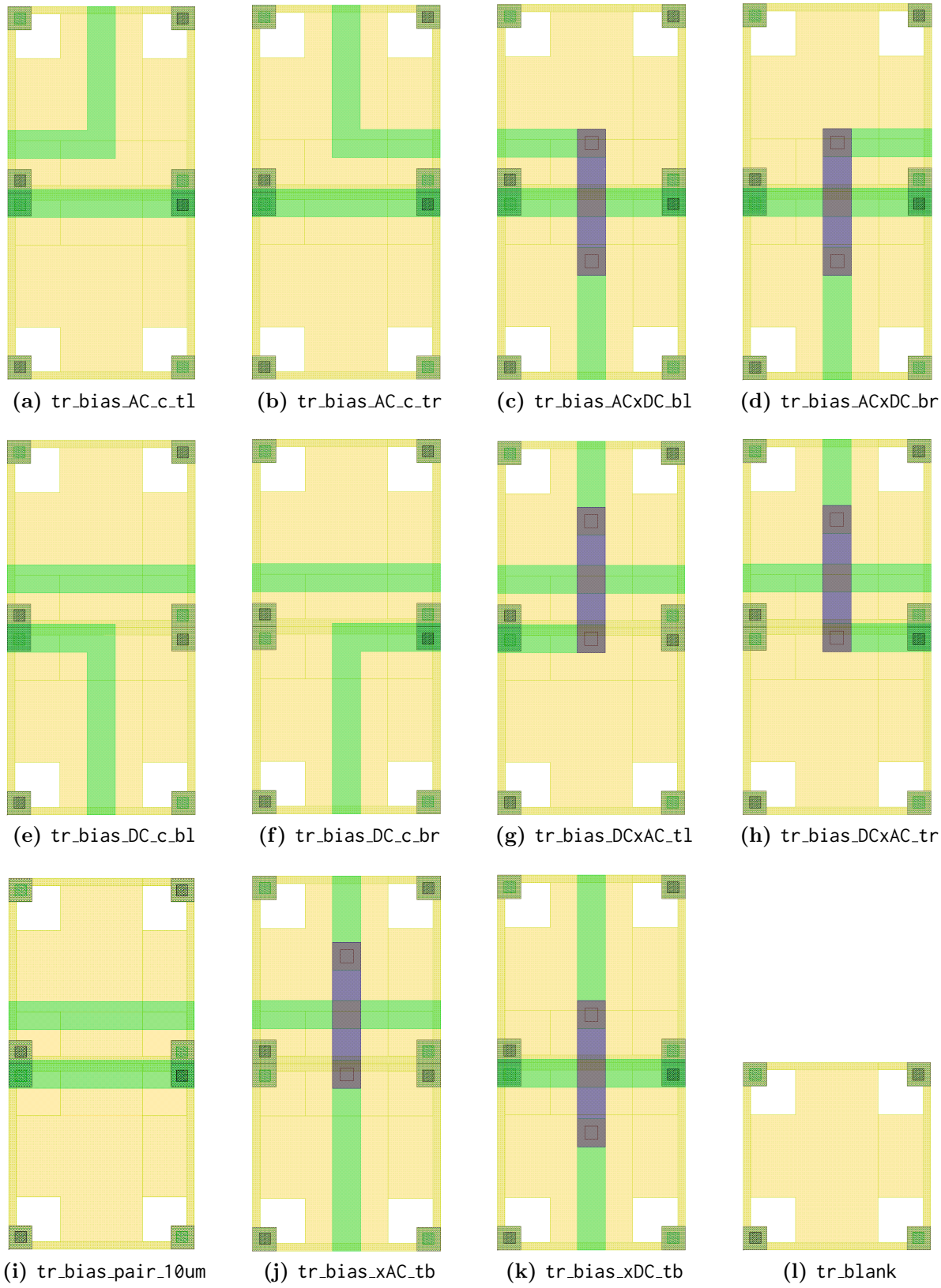
Table 2.29: (Continued from previous page.)

Name	Signal		Ground	Purpose
	a-b	c-d		
tr_wire_M1_halfPTL	M1	N/A	M0 M2	Half PTL on M1 to stack with tr_conn_M1M6.
tr_wire_M3_halfPTL	M3	N/A	M2 M4	Half PTL on M3 to stack with tr_conn_M1M6.
tr_wire_M1_h	M1	N/A	M0 M2	Horizontal wire on M1.
tr_wire_M1_v	M1	N/A	M0 M2	Vertical wire on M1.
tr_wire_M3_v	M3	N/A	M2 M4	Vertical wire on M3.
tr_wire_M3_h	M3	N/A	M2 M4	Horizontal wire on M3.
tr_wire_M1_c_bl	M1	N/A	M0 M2	Corner M1 wire connection with bottom and left ports.
tr_wire_M1_c_br	M1	N/A	M0 M2	Corner M1 wire connection with bottom and right ports.
tr_wire_M1_c_tl	M1	N/A	M0 M2	Corner M1 wire connection with top and left ports.
tr_wire_M1_c_tr	M1	N/A	M0 M2	Corner M1 wire connection with top and right ports.
tr_wire_M3_c_bl	M3	N/A	M2 M4	Corner M3 wire connection with bottom and left ports.
tr_wire_M3_c_br	M3	N/A	M2 M4	Corner M3 wire connection with bottom and right ports.
tr_wire_M3_c_tl	M3	N/A	M2 M4	Corner M3 wire connection with top and left ports.
tr_wire_M3_c_tr	M3	N/A	M2 M4	Corner M3 wire connection with top and right ports.
tr_wire_M1M3_c_bl	M1 M3	N/A	M0 M2 M4	Corner M1/M3 wire connection with bottom and left ports.
tr_wire_M1M3_c_br	M1 M3	N/A	M0 M2 M4	Corner M1/M3 wire connection with bottom and right ports.
tr_wire_M1M3_c_tl	M1 M3	N/A	M0 M2 M4	Corner M1/M3 wire connection with top and left ports.
tr_wire_M1M3_c_tr	M1 M3	N/A	M0 M2 M4	Corner M1/M3 wire connection with top and right ports.
tr_wire_M5_c_bl	M5	N/A	M4 M6	Corner M5 wire with bottom and left ports.
tr_wire_M5_c_br	M5	N/A	M4 M6	Corner M5 wire with bottom and right ports.
tr_wire_M5_c_tl	M5	N/A	M4 M6	Corner M5 wire with top and left ports.
tr_wire_M5_c_tr	M5	N/A	M4 M6	Corner M5 wire with top and right ports.
tr_wire_M5_h	M5	N/A	M4 M6	Horizontal wire on M5.
tr_wire_M5_v	M5	N/A	M4 M6	Vertical wire on M5.
tr_wire_M5_T_lbr	M5	N/A	M4 M6	T-shaped M5 wire with left, bottom and right ports.
tr_wire_M5_T_ltr	M5	N/A	M4 M6	T-shaped M5 wire with left, top and right ports.
tr_wire_M5_T_tlb	M5	N/A	M4 M6	T-shaped M5 wire with top, left and bottom ports.

Continued on the next page...

Table 2.29: (Continued from previous page.)

Name	Signal		Ground	Purpose
	a-b	c-d		
tr_wire_M5_T_trb	M5	N/A	M4 M6	T-shaped M5 wire with top, right and bottom ports.
tr_wire_xM5_h	M5	N/A	M4 M7	Horizontal wire M5 cross.
tr_wire_xM5_v	M5	N/A	M4 M7	Vertical wire M5 cross.

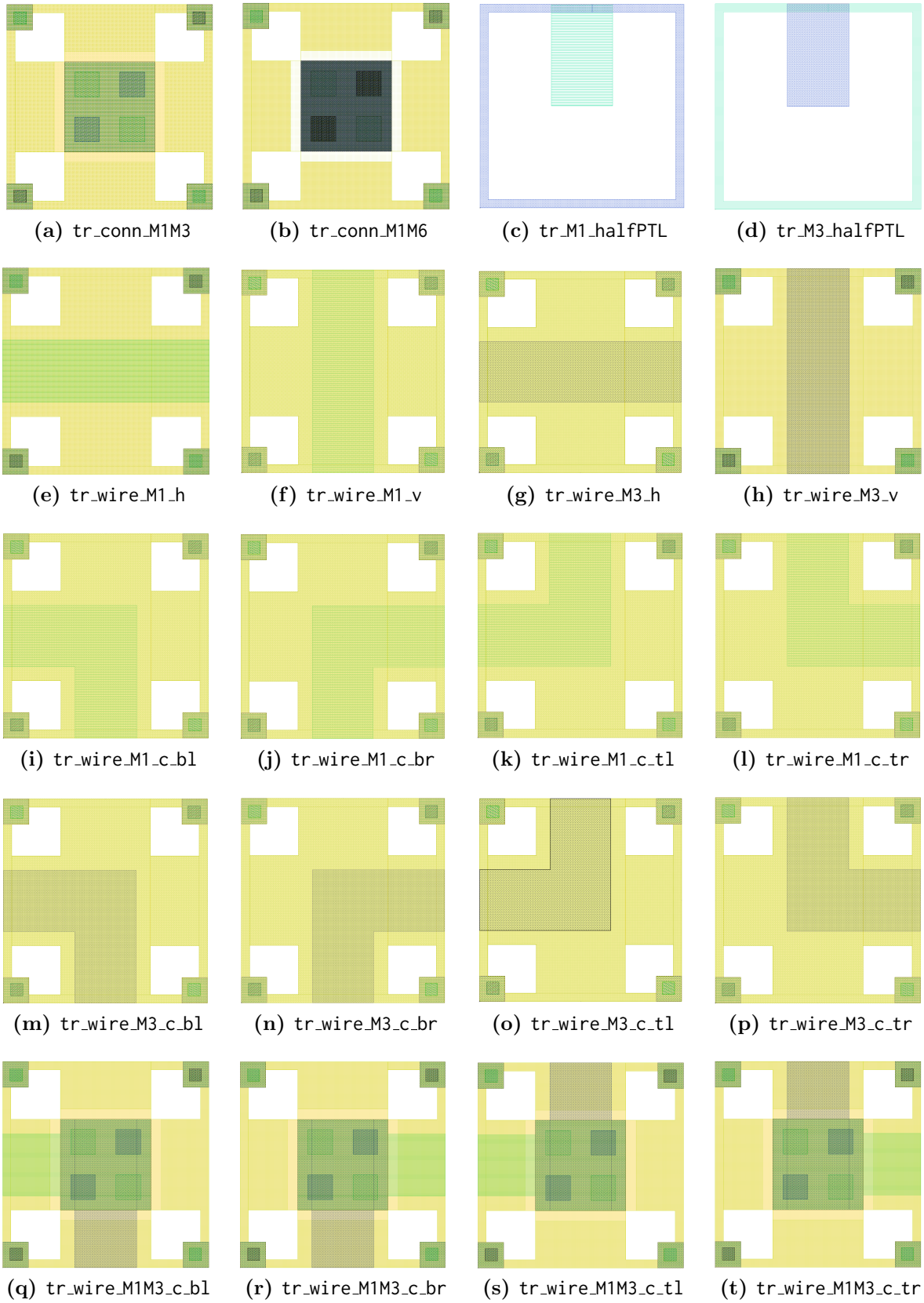


(a) tr_bias_AC_c_tl **(b)** tr_bias_AC_c_tr **(c)** tr_bias_ACxDC_bl **(d)** tr_bias_ACxDC_br

(e) tr_bias_DC_c_bl **(f)** tr_bias_DC_c_br **(g)** tr_bias_DCxAC_tl **(h)** tr_bias_DCxAC_tr

(i) tr_bias_pair_10um **(j)** tr_bias_xAC_tb **(k)** tr_bias_xDC_tb **(l)** tr_blank

Figure 2.69: Bias line interconnect slices. Each slice is 10 µm × 20 µm. A blank 10 µm × 10 µm track block, which is used as a template for all interconnect slices, is also shown in (l).



(a) tr_conn_M1M3 (b) tr_conn_M1M6 (c) tr_M1_halfPTL (d) tr_M3_halfPTL

(e) tr_wire_M1_h (f) tr_wire_M1_v (g) tr_wire_M3_h (h) tr_wire_M3_v

(i) tr_wire_M1_c_bl (j) tr_wire_M1_c_br (k) tr_wire_M1_c_tl (l) tr_wire_M1_c_tr

(m) tr_wire_M3_c_bl (n) tr_wire_M3_c_br (o) tr_wire_M3_c_tl (p) tr_wire_M3_c_tr

(q) tr_wire_M1M3_c_bl (r) tr_wire_M1M3_c_br (s) tr_wire_M1M3_c_tl (t) tr_wire_M1M3_c_tr

Figure 2.70: Cell-to-cell wire interconnect slices made of stripline-type passive transmission lines. Each slice is 10 µm × 10 µm.

2.4 Off-chip Interface

QDC (QFP-DC-Converter) is an off-chip interface circuit consisting of a dc-SQUID that converts an AQFP signal to an amplified unipolar return-to-zero (RZ) voltage signal for external read out. The qdc test circuit consists of a single buffer before the qdc input and the qdc itself. The qdc has 3 stages: inverter (**inv**), pre-buffer-to-squid (`pre_bsquid`), and buffer-to-squid (**bfr_squid**). **bfr_squid** (previous name: stack) is the dc-SQUID itself coupled to the `pre_bsquid`. The **bfr_squid** creates an output voltage only when it senses negative current at its input. Thus, the qdc includes an inverter so that a positive input results in a high output voltage level. The `pre_bsquid` is an AQFP buffer with the output transformer removed and uses Josephson junctions with large critical current I_c. The `pre_bsquid` is used to amplify the output current level. Here we only introduce the `pre_bsquid` and **bfr_squid**, together with simulation results of a qdc test circuit.

2.4.1 PRE_BSQUID

Symbol

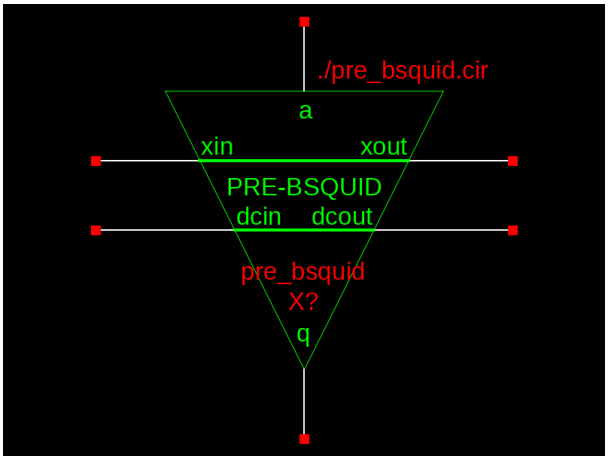


Figure 2.71: `pre_bsquid` symbol.

Table 2.30: `pre_bsquid` pin list.

Pin	Description
A	data input
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

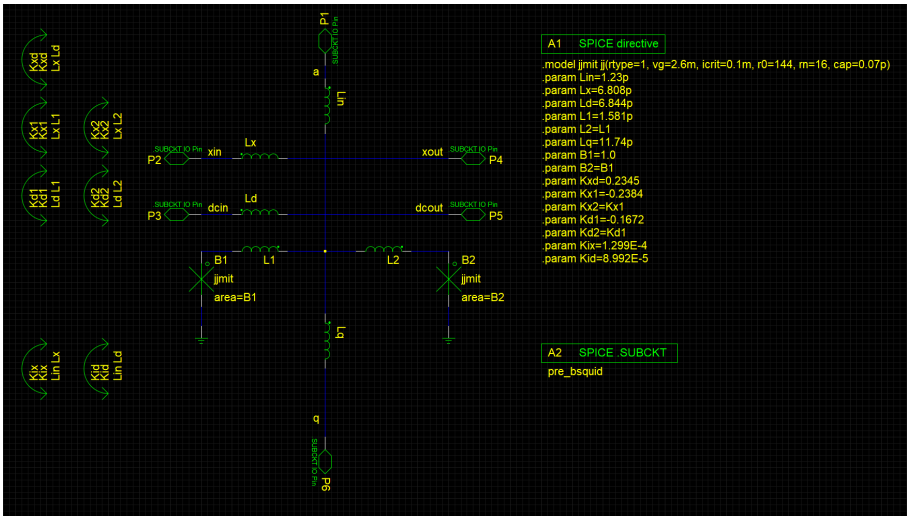


Figure 2.72: `pre_bsquid` schematic.

Layout

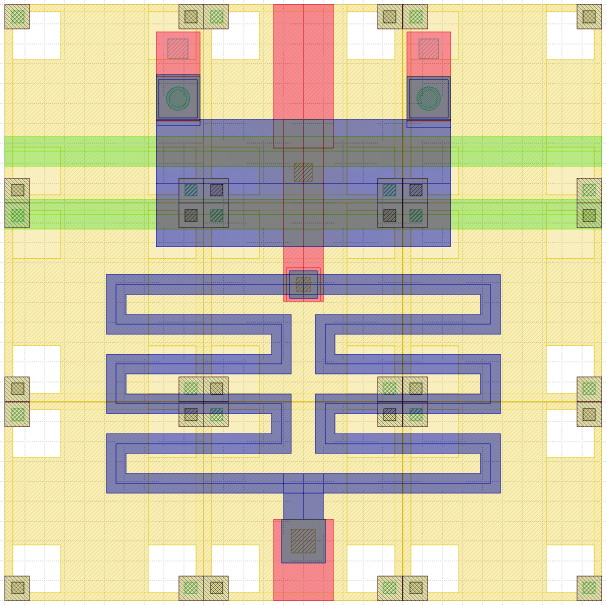


Figure 2.73: `pre_bsquid` layout.

Analog model

```
.SUBCKT pre_bsquid a xin dcin xout dcout q
Kid Lin Ld Kid
Kix Lin Lx Kix
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=0.90p
.param Lx=6.37p
.param Ld=6.37p
.param L1=1.64p
.param L2=L1
.param Lq=11.92p
.param B1=1.0
.param B2=B1
.param Kxd=0.2529
.param Kx1=-0.1950
.param Kx2=Kx1
.param Kd1=-0.1300
.param Kd2=Kd1
.param Kix=4.423E-3
.param Kid=4.041E-3

Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lq 2 q Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends pre_bsquid
```

Listing 2.29: `pre_bsquid` netlist (.cir).

2.4.2 BFR_SQUID

Symbol

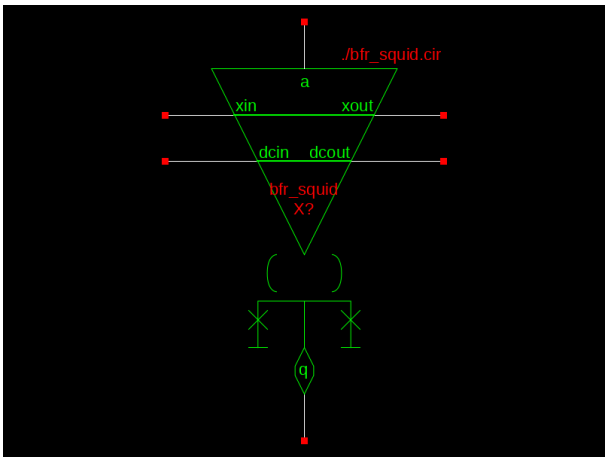


Figure 2.74: `bfr_squid` symbol.

Table 2.31: `stack` pin list.

Pin	Description
A	data input
XIN	serial clock input
DCIN	dc offset input
BIN	bias input for SQUID
XOUT	serial clock output
DCOUT	dc offset output
BOUT	bias output for SQUID

Schematic

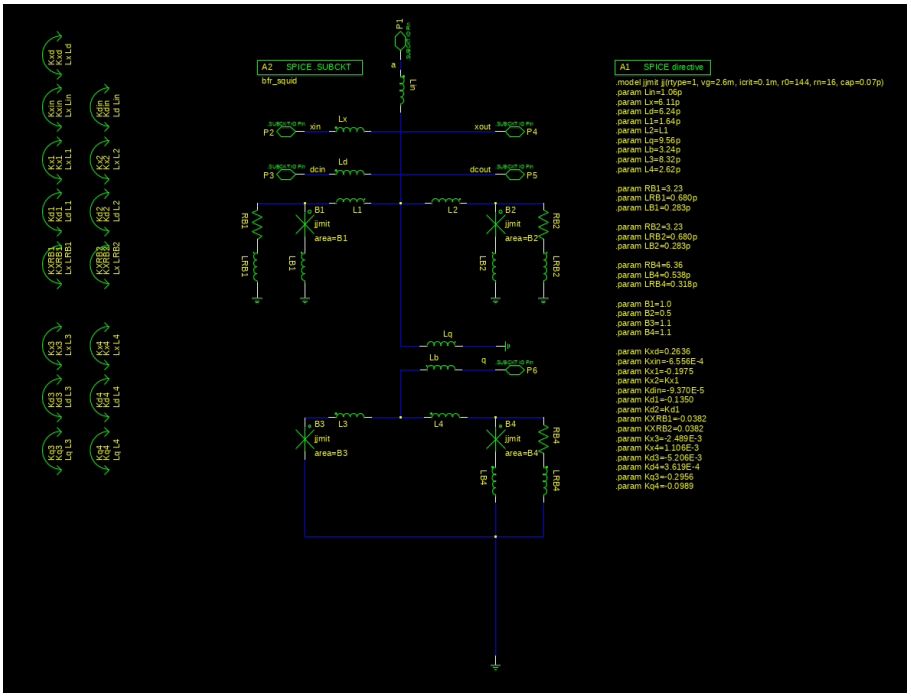


Figure 2.75: `bfr_squid` schematic.

Layout

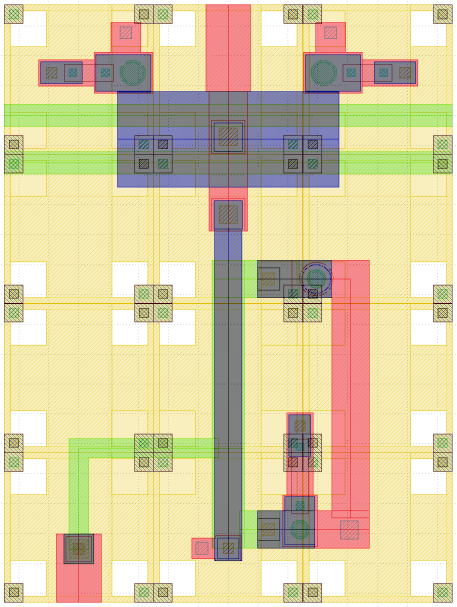


Figure 2.76: `bfr_squid` layout.

Analog model

```
.SUBCKT bfr_squid a xin dcin xout dcout q
Kdin Ld Lin Kdin
Kxin Lx Lin Kxin
Kq4 Lq L4 Kq4
Kq3 Lq L3 Kq3
Kd4 Ld L4 Kd4
Kd3 Ld L3 Kd3
Kx4 Lx L4 Kx4
Kx3 Lx L3 Kx3
KXRB2 Lx LRB2 KXRB2
L3 9 8 L3
L4 8 10 L4
LRB4 12 0 LRB4
RB4 10 12 RB4
LB4 11 0 LB4
B4 10 11 B4 jjmit  area=B4
B3 9 0 B3 jjmit  area=B3
Lb 8 q Lb
LRB2 7 0 LRB2
RB2 3 7 RB2
LRB1 6 0 LRB1
RB1 1 6 RB1
LB2 5 0 LB2
LB1 4 0 LB1
Lq 0 2 Lq
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.06p
.param Lx=6.11p
.param Ld=6.24p
.param L1=1.64p
.param L2=L1
.param Lq=9.56p
.param Lb=3.24p
.param L3=8.32p
.param L4=2.62p

.param RB1=3.23
.param LRB1=0.680p
.param LB1=0.283p

.param RB2=3.23
.param LRB2=0.680p
.param LB2=0.283p

.param RB4=6.36
.param LB4=0.538p
.param LRB4=0.318p

.param B1=1.0
.param B2=0.5
.param B3=1.1
.param B4=1.1

.param Kxd=0.2636
.param Kxin=-6.556E-4
.param Kx1=-0.1975
.param Kx2=Kx1
.param Kdin=-9.370E-5
.param Kd1=-0.1350
.param Kd2=Kd1
.param KXRB1=-0.0382
.param KXRB2=0.0382
.param Kx3=-2.489E-3
.param Kx4=1.106E-3
.param Kd3=-5.206E-3
.param Kd4=3.619E-4
.param Kq3=-0.2956
```

```
68  .param Kq4=-0.0989
69  KXRB1 Lx LRB1 KXRB1
70  Kd2 Ld L2 Kd2
71  Kx2 Lx L2 Kx2
72  Kd1 Ld L1 Kd1
73  Kx1 Lx L1 Kx1
74  Kxd Lx Ld Kxd
75  Lin a 2 Lin
76  B2 3 5 B2 jjmit  area=B2
77  B1 1 4 B1 jjmit  area=B1
78  L2 3 2 L2
79  L1 2 1 L1
80  Ld dcin dcout Ld
81  Lx xin xout Lx
82  .ends bfr_squid
```

Listing 2.30: `bfr_squid` netlist (.cir).

2.4.3 QDC

Symbol

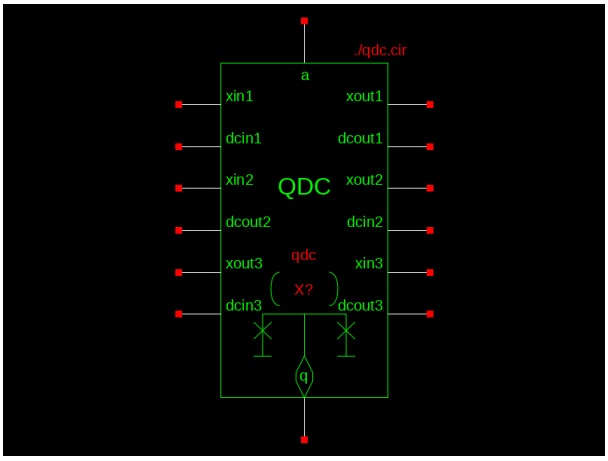


Figure 2.77: `qdc` symbol.

Table 2.32: qdc pin list.

Pin	Description
A	data input
XIN1	serial AC clock input (inv)
DCIN1	dc offset input (inv)
XIN2	serial AC clock input (pre_bsquid)
DCIN2	dc offset input (pre_bsquid)
XIN3	serial AC clock input (stack)
DCIN3	dc offset input (stack)
XOUT1	serial AC clock output (inv)
DCOUT1	dc offset output (inv)
XOUT2	serial AC clock output (pre_bsquid)
DCOUT2	dc offset output (pre_bsquid)
XOUT3	serial AC clock output (stack)
DCOUT3	dc offset output (stack)
Q	data output (observed through bias input of SQUID)

Schematic

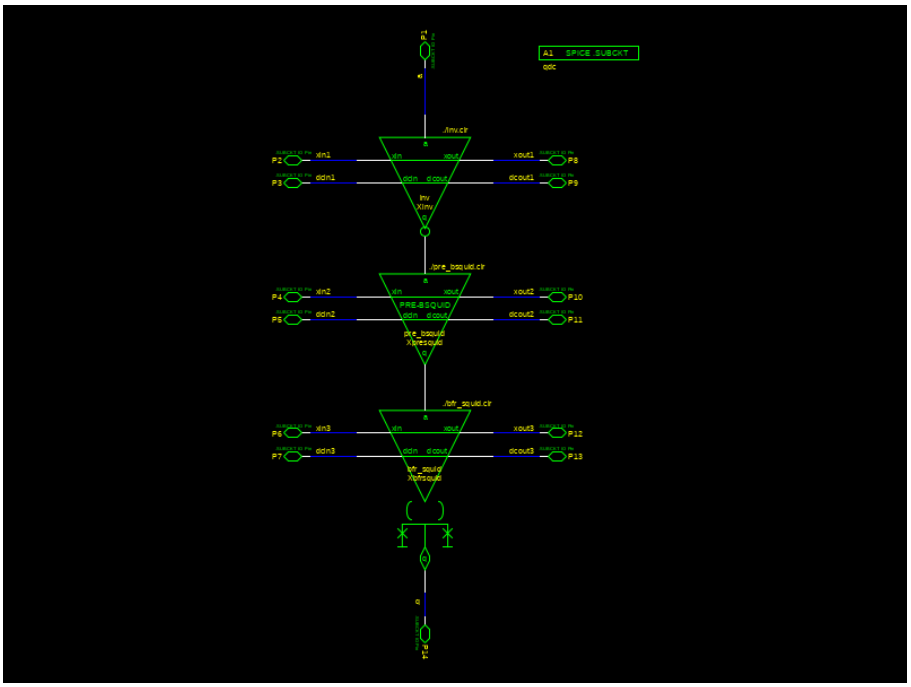


Figure 2.78: qdc schematic.

Layout

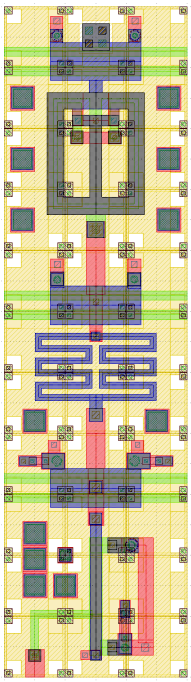


Figure 2.79: qdc layout.

Analog model

```
.SUBCKT inv a xin dcin xout dcout q
Kid Lin Ld Kid
Kiout Lin Lout Kiout
Kix Lin Lx Kix
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.56p
.param Lx=6.54p
.param Ld=6.57p
.param L1=1.56p
.param L2=L1
.param Lq=7.14p
.param Lout=30.24p
.param B1=0.5
.param B2=B1
.param Kxd=0.2912
.param Kx1=-0.2460
.param Kx2=Kx1
.param Kd1=-0.1909
.param Kd2=Kd1
.param Kout=0.5917
.param Kxout=1.335E-5
.param Kdout=2.443E-5
.param Kiout=1.172E-3
.param Kix=-2.712E-4
.param Kid=2.605E-4

Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
```

```
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends inv
.SUBCKT pre_bsquid a xin dcin xout dcout q
Kid Lin Ld Kid
Kix Lin Lx Kix
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=0.90p
.param Lx=6.37p
.param Ld=6.37p
.param L1=1.64p
.param L2=L1
.param Lq=11.92p
.param B1=1.0
.param B2=B1
.param Kxd=0.2529
.param Kx1=-0.1950
.param Kx2=Kx1
.param Kd1=-0.1300
.param Kd2=Kd1
.param Kix=4.423E-3
.param Kid=4.041E-3

Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lq 2 q Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends pre_bsquid
.SUBCKT bfr_squid a xin dcin xout dcout q
Kdin Ld Lin Kdin
Kxin Lx Lin Kxin
Kq4 Lq L4 Kq4
Kq3 Lq L3 Kq3
Kd4 Ld L4 Kd4
Kd3 Ld L3 Kd3
Kx4 Lx L4 Kx4
Kx3 Lx L3 Kx3
KXRB2 Lx LRB2 KXRB2
L3 9 8 L3
L4 8 10 L4
LRB4 12 0 LRB4
RB4 10 12 RB4
LB4 11 0 LB4
B4 10 11 B4 jjmit  area=B4
B3 9 0 B3 jjmit  area=B3
Lb 8 q Lb
LRB2 7 0 LRB2
RB2 3 7 RB2
LRB1 6 0 LRB1
RB1 1 6 RB1
LB2 5 0 LB2
LB1 4 0 LB1
```

```
Lq 0 2 Lq
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.06p
.param Lx=6.11p
.param Ld=6.24p
.param L1=1.64p
.param L2=L1
.param Lq=9.56p
.param Lb=3.24p
.param L3=8.32p
.param L4=2.62p

.param RB1=3.23
.param LRB1=0.680p
.param LB1=0.283p

.param RB2=3.23
.param LRB2=0.680p
.param LB2=0.283p

.param RB4=6.36
.param LB4=0.538p
.param LRB4=0.318p

.param B1=1.0
.param B2=0.5
.param B3=1.1
.param B4=1.1

.param Kxd=0.2636
.param Kxin=-6.556E-4
.param Kx1=-0.1975
.param Kx2=Kx1
.param Kdin=-9.370E-5
.param Kd1=-0.1350
.param Kd2=Kd1
.param KXRB1=-0.0382
.param KXRB2=0.0382
.param Kx3=-2.489E-3
.param Kx4=1.106E-3
.param Kd3=-5.206E-3
.param Kd4=3.619E-4
.param Kq3=-0.2956
.param Kq4=-0.0989
KXRB1 Lx LRB1 KXRB1
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
Lin a 2 Lin
B2 3 5 B2 jjmit  area=B2
B1 1 4 B1 jjmit  area=B1
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends bfr_squid
.SUBCKT qdc a xin1 dcin1 xin2 dcin2 xin3 dcin3 xout1 dcout1 xout2 dcout2 xout3 dcout3 q
Xbfrsquid 2 xin3 dcin3 xout3 dcout3 q bfr_squid
Xpresquid 1 xin2 dcin2 xout2 dcout2 2 pre_bsquid
Xinv a xin1 dcin1 xout1 dcout1 1 inv
.ends qdc
```

Listing 2.31: qdc netlist (.cir).

Simulation result

An input of 0 1 0 1 0 1 1 0 is sent to the input of the qdc which corresponds to the output shown on the waveform. Signals from top to bottom: AC source 1 (AC1) (generates phase 1 and 3), AC source 2 (AC2) (generates phase 2 and 4), Input at the buffer before the qdc and the output of qdc (measured across a resistor connected to qdc's output). Input peak-to-peak amplitude is ±5 µA, AC amplitude is 658 µA, and DC is set to 843 µA.

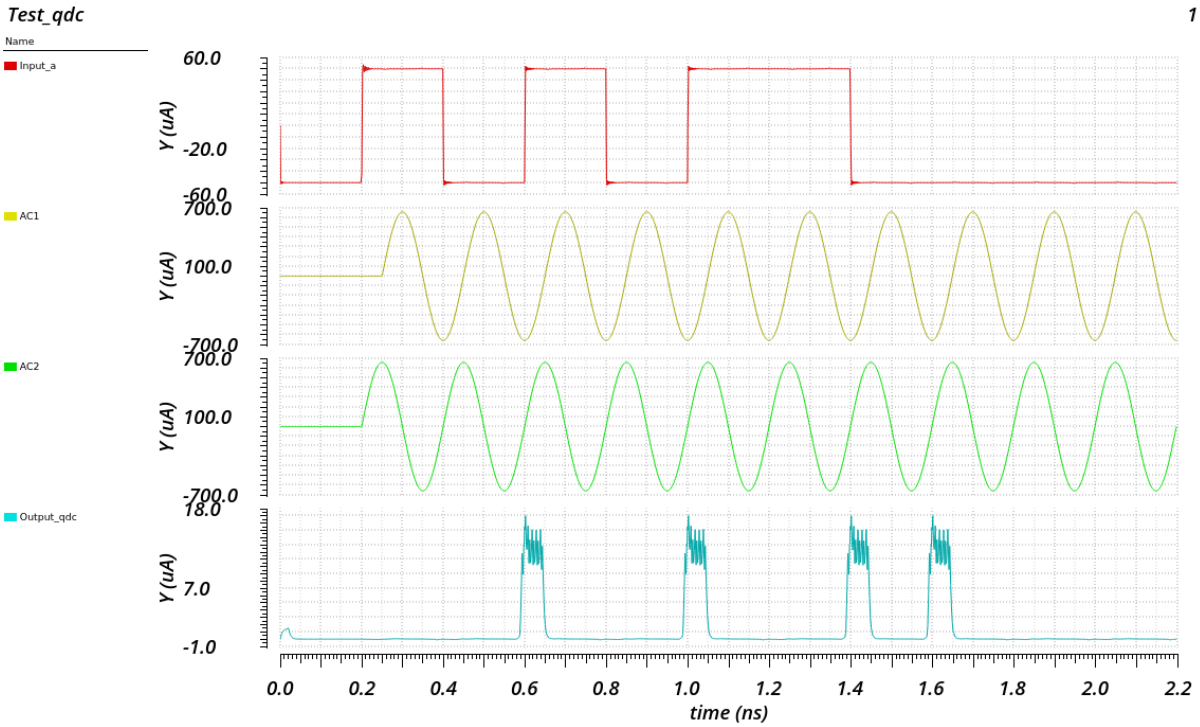


Figure 2.80: qdc analog waveform.

2.5 On-chip Interfaces

2.5.1 AQFP2RSFQ

`aqfp2rsfq` is an interface that allows AQFP logic to transfer its current-based data to SFQ logic which uses SFQ-encoded data. It is based on an AQFP `bfr` cell and a magnetically coupled dc/SFQ converter. Both components are synchronized to the AC excitation current such when it rises, the AQFP buffer sets its output while an SFQ clock is generated through a Josephson junction switching event caused by magnetic flux applied from the AC excitation current via magnetic coupling. The output of AQFP buffer is magnetically coupled to the input branch of the Josephson comparator in the dc/SFQ converter. In the previous design, the polarity of this coupling was negative so the circuit effectively behaves like an inverter since the negative coupling is more natural in physical layout. In this version (ver2.1), we improved the physical layout design and were able to make the coupling positive, which means the circuit is now non-inverting. The comparator will produce an appropriate output when the SFQ clock arrives at the comparator, and the circuit behaves like a buffer. When the AQFP has a logic '1' state during excitation, the dc/SFQ converter will produce an output SFQ pulse. Otherwise, no SFQ pulse is generated.

Symbol

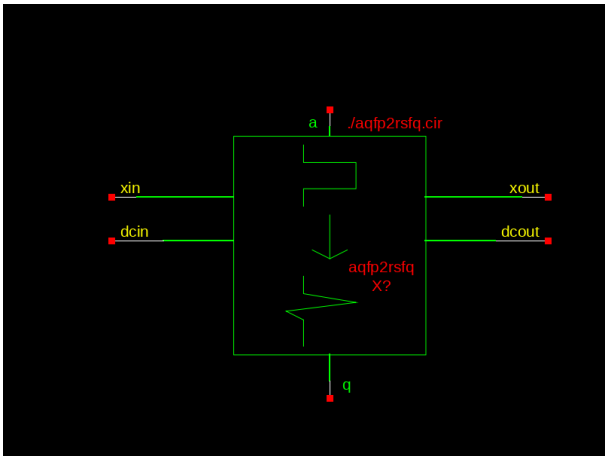


Figure 2.81: `aqfp2rsfq` symbol.

Table 2.33: `aqfp2rsfq` pin list.

Pin	Description
A	data input (AQFP)
XIN	serial clock input
DCIN	dc offset input
Q	data output (RSFQ)
XOUT	serial clock output
DCOUT	dc offset output

Schematic

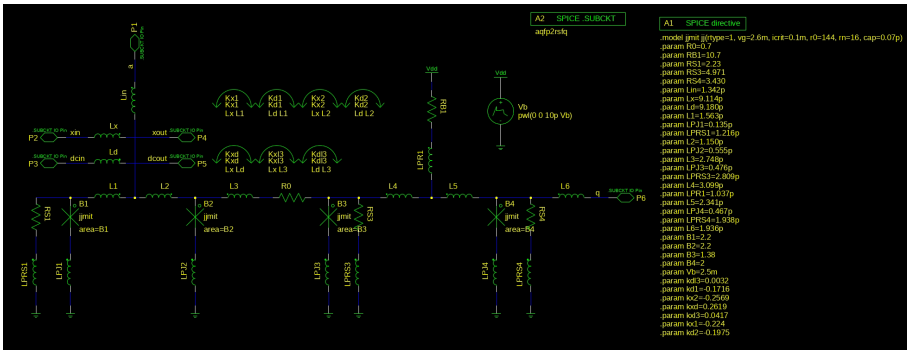


Figure 2.82: `aqfp2rsfq` schematic.

Layout

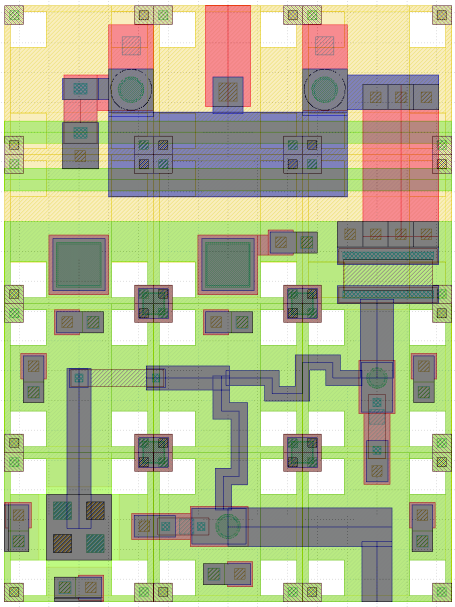


Figure 2.83: `aqfp2rsfq` layout.

Analog model

```
.SUBCKT aqfp2rsfq a xin dcin xout dcout q
Vb Vdd 0 pwl(0 0 10p Vb)
RS1 1 15 RS1
LPRS1 15 0 LPRS1
LPJ1 2 0 LPJ1
LPRS4 13 0 LPRS4
LPRS3 14 0 LPRS3
RS3 7 14 RS3
RS4 10 13 RS4
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param R0=0.7
.param RB1=10.7
.param RS1=2.23
```

```
14  .param RS3=4.971
15  .param RS4=3.430
16  .param Lin=1.342p
17  .param Lx=9.114p
18  .param Ld=9.180p
19  .param L1=1.563p
20  .param LPJ1=0.135p
21  .param LPRS1=1.216p
22  .param L2=1.150p
23  .param LPJ2=0.555p
24  .param L3=2.748p
25  .param LPJ3=0.476p
26  .param LPRS3=2.809p
27  .param L4=3.099p
28  .param LPR1=1.037p
29  .param L5=2.341p
30  .param LPJ4=0.467p
31  .param LPRS4=1.938p
32  .param L6=1.936p
33  .param B1=2.2
34  .param B2=2.2
35  .param B3=1.38
36  .param B4=2
37  .param Vb=2.5m
38  .param kdl3=0.0032
39  .param kd1=-0.1716
40  .param kx2=-0.2569
41  .param kxd=0.2619
42  .param kxl3=0.0417
43  .param kx1=-0.224
44  .param kd2=-0.1975
45  B4 10 12 B4 jjmit  area=B4
46  Kdl3 Ld L3 Kdl3
47  Kxl3 Lx L3 Kxl3
48  Kxd Lx Ld Kxd
49  Kd2 Ld L2 Kd2
50  Kx2 Lx L2 Kx2
51  Kd1 Ld L1 Kd1
52  Kx1 Lx L1 Kx1
53  RB1 Vdd 11 RB1
54  Lin a 5 Lin
55  Ld dcout dcin Ld
56  Lx xout xin Lx
57  LPJ4 12 0 LPJ4
58  LPR1 11 9 LPR1
59  L6 10 q L6
60  L5 9 10 L5
61  L4 7 9 L4
62  LPJ3 8 0 LPJ3
63  B3 7 8 B3 jjmit  area=B3
64  R0 6 7 R0
65  L3 3 6 L3
66  LPJ2 4 0 LPJ2
67  L2 3 5 L2
68  L1 5 1 L1
69  B2 3 4 B2 jjmit  area=B2
70  B1 1 2 B1 jjmit  area=B1
71  .ends aqfp2rsfq
```

Listing 2.32: `aqfp2rsfq` netlist (.cir).

Simulation result

Simulation waveform of the `aqfp2rsfq`. The data input peak-to-peak amplitude is ±5 µA, AC amplitude is 800 µA, and DC is set to 1.2 mA. The dc/SFQ component uses a bias voltage of 2.5 mV. The AQFP data pattern is 10010110 which resulted in the SFQ comparator producing a same data pattern of 10010110 and the circuit operates as a buffer.

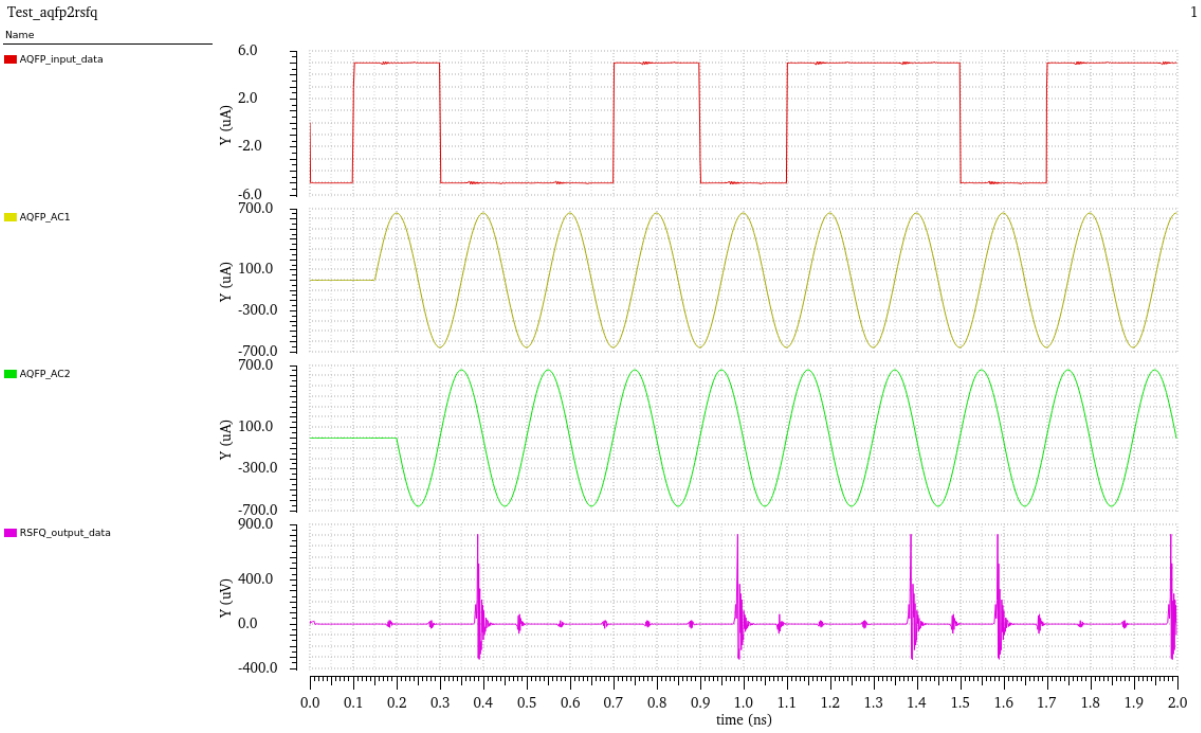


Figure 2.84: `aqfp2rsfq` analog waveform.

Switching energy

To be investigated in the future.

2.5.2 RSFQ2AQFP

`rsfq2aqfp` is an interface that allows RSFQ logic to transfer its SFQ-based data to AQFP logic which operates on bidirectional current-encoded data. It is based on an RSFQ DFF whose storage loop is magnetically coupled to an output inductor. An inductor with about −400 µA offset current is coupled to the output inductor to shift the current levels induced into the output inductor from unipolar to bipolar. This bipolar current is then suitable as data input for AQFP. When the DFF is in logic state '1', a positive output current is produced. Otherwise, the output current will remain negative. An SFQ clock is used to reset the `rsfq2aqfp` interface back to state '0'.

Symbol

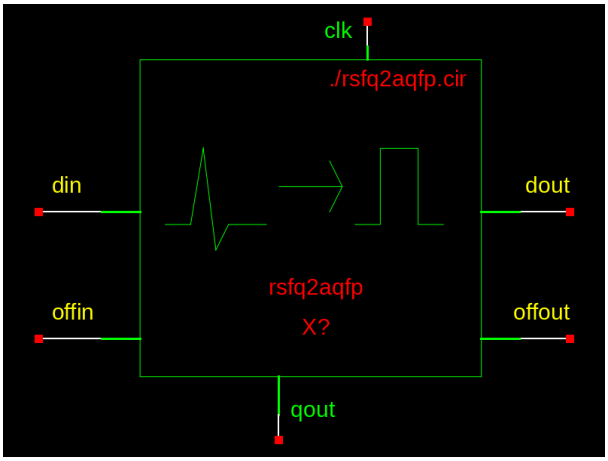


Figure 2.85: `rsfq2aqfp` symbol.

Table 2.34: `rsfq2aqfp` pin list.

Pin	Description
A	data input (RSFQ)
CLK	clock (RSFQ)
OFF	dc offset to shift the level of Q
Q	data output (AQFP)
SQ	data output (SFQ) to eject SFQ pulse

Schematic

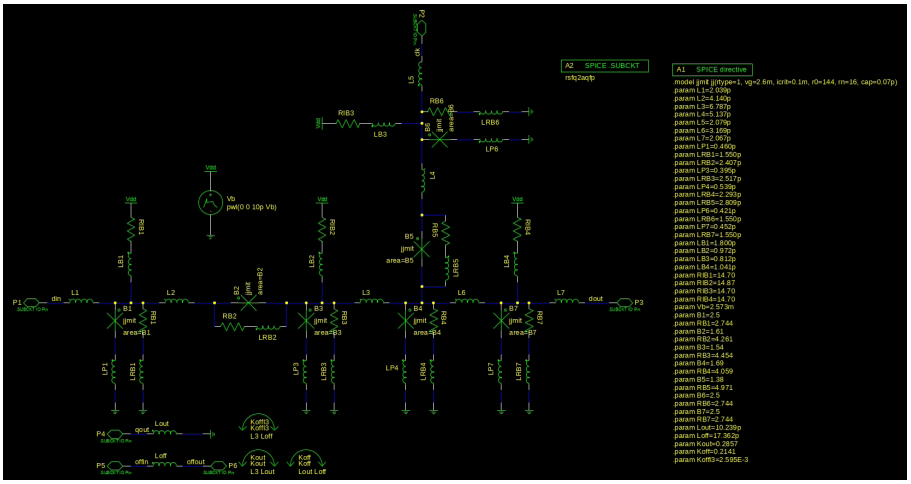


Figure 2.86: `rsfq2aqfp` schematic.

Layout

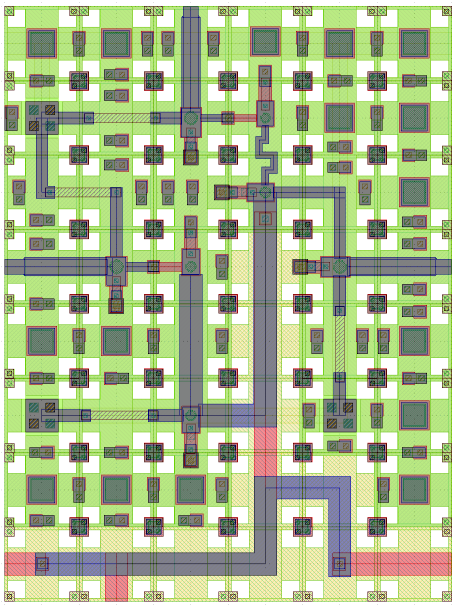


Figure 2.87: `rsfq2aqfp` layout.

Analog model

```
.SUBCKT rsfq2aqfp din clk dout qout offin offout
Vb Vdd 0 pwl(0 0 10p Vb)
RIB3 Vdd 22 RIB3
RIB4 Vdd 16 RIB4
RIB2 Vdd 9 RIB2
RIB1 Vdd 4 RIB1
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param L1=2.039p
```

```
.param L2=4.140p
.param L3=6.787p
.param L4=5.137p
.param L5=2.079p
.param L6=3.169p
.param L7=2.067p
.param LP1=0.460p
.param LRB1=1.550p
.param LRB2=2.407p
.param LP3=0.395p
.param LRB3=2.517p
.param LP4=0.539p
.param LRB4=2.293p
.param LRB5=2.809p
.param LP6=0.421p
.param LRB6=1.550p
.param LP7=0.452p
.param LRB7=1.550p
.param LB1=1.800p
.param LB2=0.972p
.param LB3=0.812p
.param LB4=1.041p
.param RIB1=14.70
.param RIB2=14.87
.param RIB3=14.70
.param RIB4=14.70
.param Vb=2.573m
.param B1=2.5
.param RB1=2.744
.param B2=1.61
.param RB2=4.261
.param B3=1.54
.param RB3=4.454
.param B4=1.69
.param RB4=4.059
.param B5=1.38
.param RB5=4.971
.param B6=2.5
.param RB6=2.744
.param B7=2.5
.param RB7=2.744
.param Lout=10.239p
.param Loff=17.362p
.param Kout=0.2857
.param Koff=0.2141
.param Koffl3=2.595E-3
Koffl3 L3 Loff Koffl3
Koff Lout Loff Koff
Kout L3 Lout Kout
Loff offin offout Loff
Lout qout 0 Lout
L5 20 clk L5
LRB6 23 0 LRB6
RB6 20 23 RB6
LB3 22 20 LB3
LP6 21 0 LP6
B6 20 21 B6 jjmit  area=B6
L4 20 18 L4
LRB5 19 11 LRB5
RB5 18 19 RB5
B5 18 11 B5 jjmit  area=B5
L7 14 dout L7
LRB7 17 0 LRB7
RB7 14 17 RB7
LB4 16 14 LB4
LP7 15 0 LP7
B7 14 15 B7 jjmit  area=B7
L6 11 14 L6
LRB4 13 0 LRB4
RB4 11 13 RB4
```

```
79   LP4 12 0 LP4
80   B4 11 12 B4 jjmit  area=B4
81   L3 5 11 L3
82   LRB3 10 0 LRB3
83   RB3 5 10 RB3
84   LB2 9 5 LB2
85   LP3 8 0 LP3
86   B3 5 8 B3 jjmit  area=B3
87   L1              din 1 L1
88   LRB2   6 5 LRB2
89   LRB1 7 0 LRB1
90   RB1 1 7 RB1
91   RB2 3 6 RB2
92   B2 3 5 B2 jjmit  area=B2
93   LB1 4 1 LB1
94   L2 1 3 L2
95   LP1 2 0 LP1
96   B1 1 2 B1 jjmit  area=B1
97   .ends rsfq2aqfp
```

Listing 2.33: `rsfq2aqfp` netlist (.cir).

Simulation result

Simulation waveform of the `rsfq2aqfp`. The data input is provided through JTLs connected to the RSFQ-based DFF with an input pattern of 01001011. An SFQ clock pulse is provided at the same frequency as the 5 GHz AQFP AC excitation clock. The AC amplitude is 800 µA, and DC is set to 1.2 mA. The bipolar AQFP output current matches the same data as the SFQ-based encoding of 01001011.

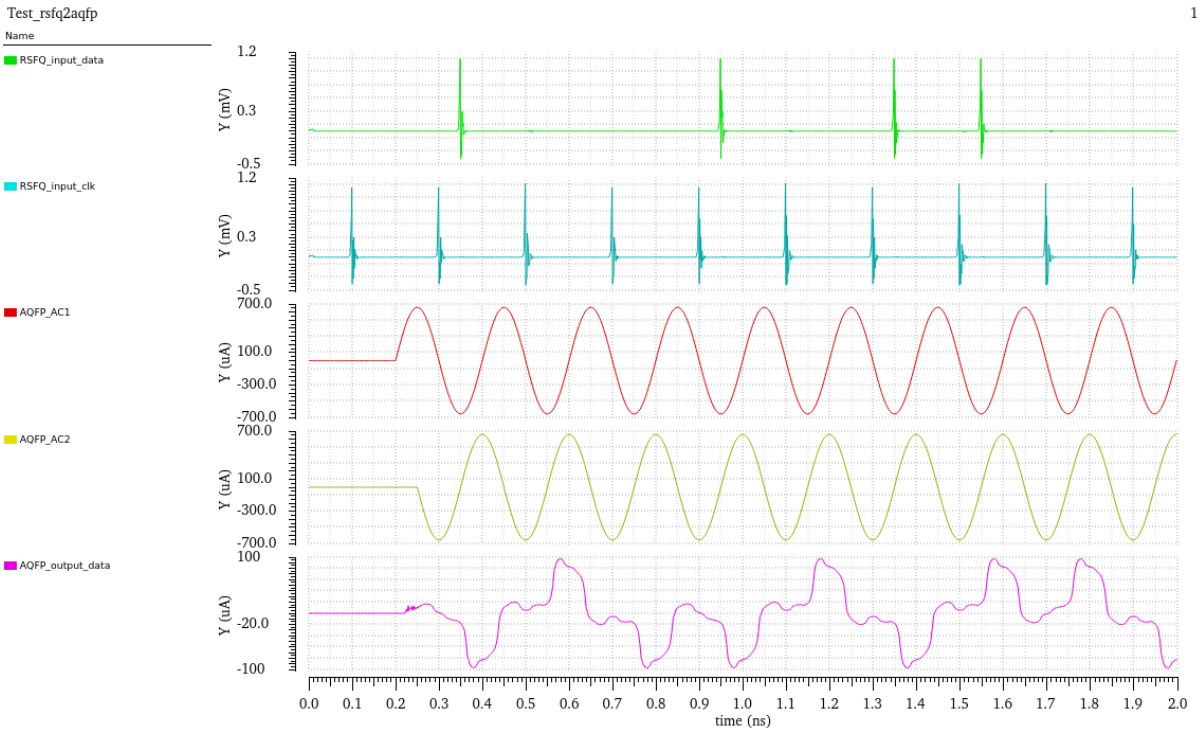


Figure 2.88: `rsfq2aqfp` analog waveform.

Switching energy

To be investigated in the future.

2.6 Sub-Cells

2.6.1 Constant

Constant cells contain constant 0 (const0) and constant 1 (const1) that are building blocks to develop AQFP logic AND and OR cells. Both const0 and const1 are designed by changing the symmetric architecture of a SQUID into asymmetric to generate the constant positive current (as logic '1') or negative current (as logic '0'). Here we only introduce the constant 0.

Symbol

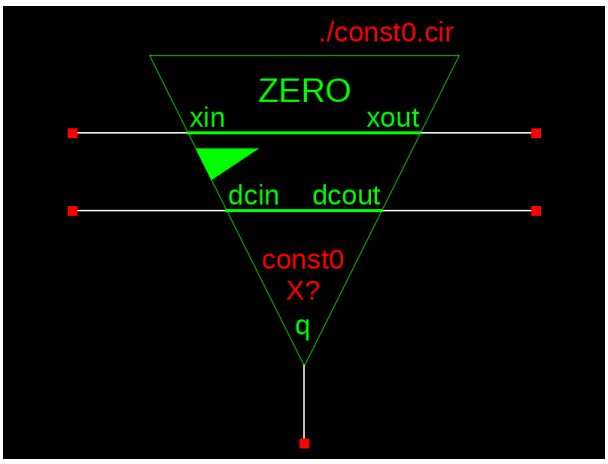


Figure 2.89: `const0` symbol.

Table 2.35: `const0` pin list.

Pin	Description
XIN	serial clock input
DCIN	dc offset input
Q	data output
XOUT	serial clock output
DCOUT	dc offset output

Schematic

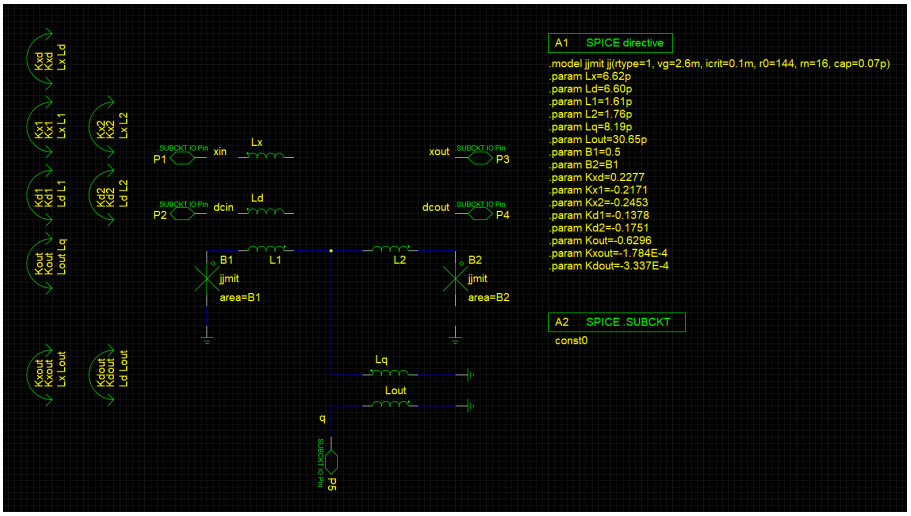


Figure 2.90: `const0` schematic.

Layout

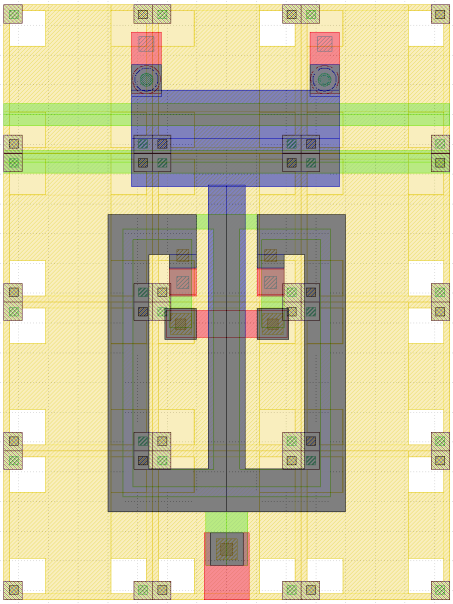


Figure 2.91: `const0` layout.

Analog model

```
.SUBCKT const0 xin dcin xout dcout q
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lx=6.62p
.param Ld=6.60p
.param L1=1.61p
.param L2=1.76p
.param Lq=8.19p
.param Lout=30.65p
.param B1=0.5
.param B2=B1
.param Kxd=0.2972
.param Kx1=-0.2265
.param Kx2=-0.2602
.param Kd1=-0.1688
.param Kd2=-0.2088
.param Kout=-0.6452
.param Kxout=-2.545E-4
.param Kdout=-5.447E-4
Kxout Lx Lout Kxout
Kdout Ld Lout Kdout
Kout Lout Lq Kout
Kd2 Ld L2 Kd2
Kx2 Lx L2 Kx2
Kd1 Ld L1 Kd1
Kx1 Lx L1 Kx1
Kxd Lx Ld Kxd
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Ld dcin dcout Ld
Lx xin xout Lx
.ends const0
```

Listing 2.34: `const0` netlist (.cir).

2.6.2 Branch

A 1-to-n (or n-to-1) branch, which consists of (n+1) superconducting inductors, is used to split or merge AQFP current. It is a basic building block to make AQFP logic cells such as AND, OR, Majority and Splitter introduced in section II. Here we only introduce 1-to-2 branch, others can be found as separate files in the deliverables.

Symbol

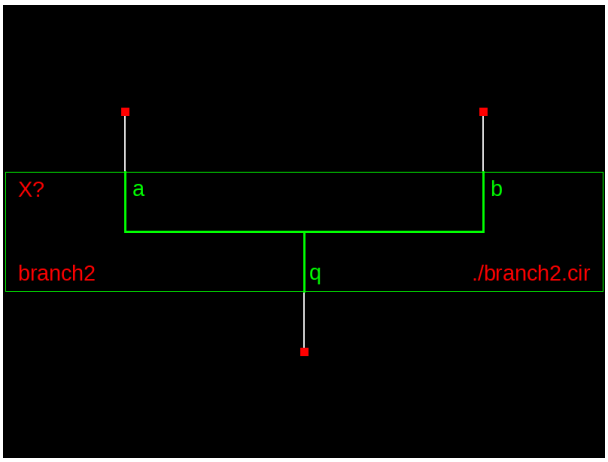


Figure 2.92: `branch2` symbol.

Table 2.36: `branch2` pin list.

Pin	Description
A	data inout
B	data inout
C	data inout

Schematic

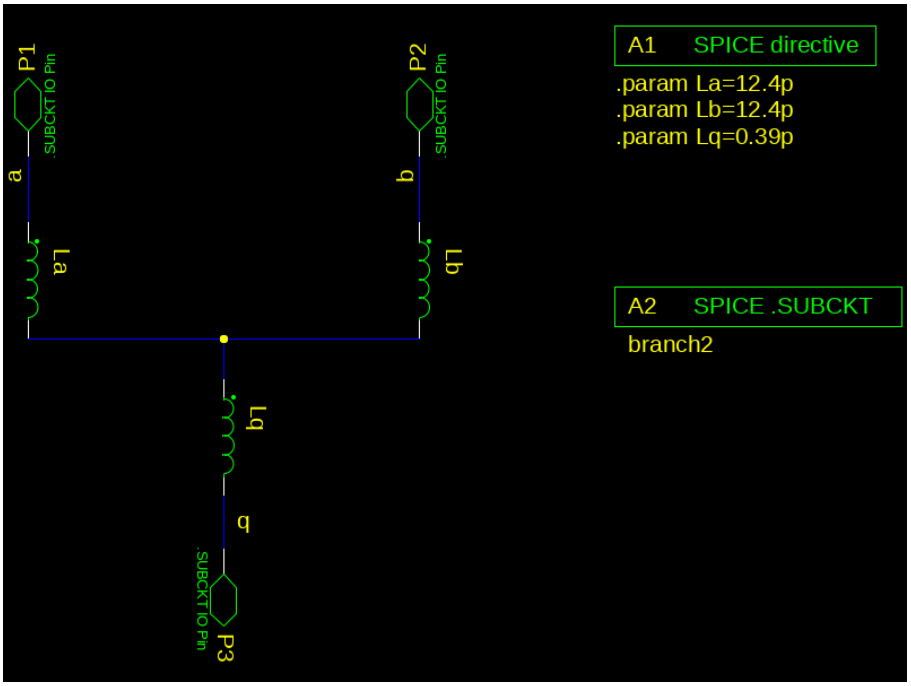


Figure 2.93: `branch2` schematic.

Layout

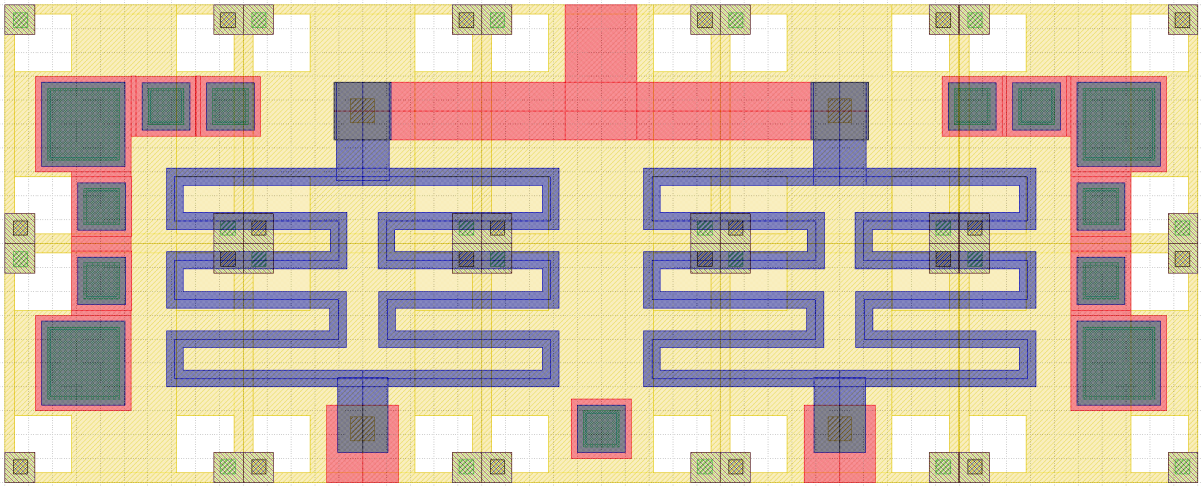


Figure 2.94: `branch2` layout.

Analog model

```
1  ******************************
2  * Begin .SUBCKT model         *
3  * spice-sdb ver 4.28.2007     *
4  ******************************
5  .SUBCKT branch2 a b q
6  *==============  Begin SPICE netlist of main design ============
7  .param La=12.4p
8  .param Lb=12.4p
9  .param Lq=0.39p
10 Lq 1 q Lq
11 Lb b 1 Lb
12 La a 1 La
13 .ends branch2
14 ******************************
```

Listing 2.35: branch2 netlist (.cir).

2.6.3 Storage gate

An AQFP storage gate is a building block to develop AQFP sequential logic cells such as latch (`qfpl`) and non-destructive read-out (`ndro_qfpl`).

Symbol

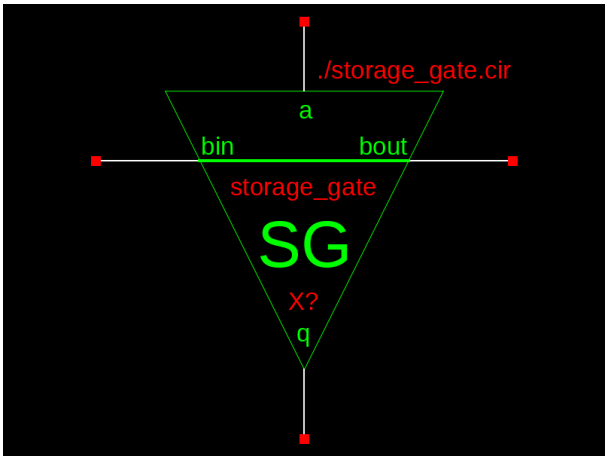


Figure 2.95: `storage_gate` symbol.

Table 2.37: `storage_gate` pin list.

Pin	Description
A	data inout
B	data inout
C	data inout

Schematic

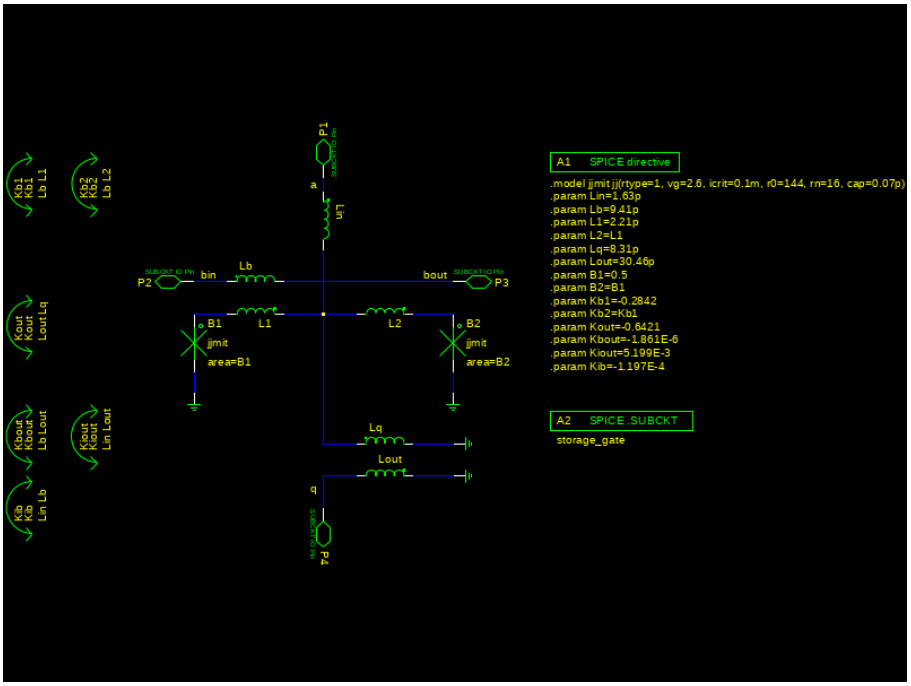


Figure 2.96: `storage_gate` schematic.

Layout

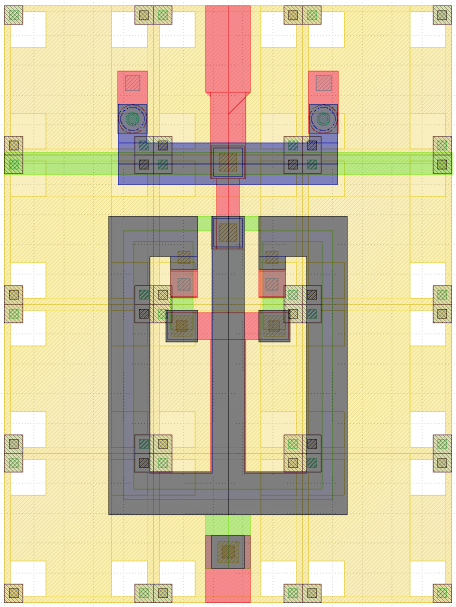


Figure 2.97: `storage_gate` layout.

Analog model

```
.SUBCKT storage_gate a bin bout q
Kiout Lin Lout Kiout
Kbout Lb Lout Kbout
.model jjmit jj(rtype=1, vg=2.6m, icrit=0.1m, r0=144, rn=16, cap=0.07p)
.param Lin=1.63p
.param Lb=9.41p
.param L1=2.21p
.param L2=L1
.param Lq=8.31p
.param Lout=30.46p
.param B1=0.5
.param B2=B1
.param Kb1=-0.2842
.param Kb2=Kb1
.param Kout=-0.6421
.param Kbout=-1.663E-5
.param Kiout=5.199E-3
.param Kib=-4.608E-5

Kib Lin Lb Kib
Kout Lout Lq Kout
Kb2 Lb L2 Kb2
Kb1 Lb L1 Kb1
Lin a 2 Lin
B2 3 0 B2 jjmit  area=B2
B1 1 0 B1 jjmit  area=B1
Lout 0 q Lout
Lq 2 0 Lq
L2 3 2 L2
L1 2 1 L1
Lb bin bout Lb
.ends storage_gate
```

Listing 2.36: `storage_gate` netlist (.cir).

2.6.4 HDL ac/dc interface

Verilog module 'biasDir_b.v' is an internal module for each AQFP logic gate for handling the excitation ac clock (AC/xout) and dc bias (dcin/dcout). Because some cells such as 'bfr' can be placed in a normal or flipped orientation, it was necessary to develop a bi-directional interface to handle the ac/dc bias. For gates such as AND or OR which can only be placed in a single direction relative to the bias direction, we enforce a uni-directional version of this interface. This interface also determines the relative directions of ac and dc bias applied to the cell. If their directions are the same, the ac clock (modeled as a digital square wave) will be applied to the logic cell as is. If their directions are different, the gate will receive an inverted version of the ac clock. This allows us to model 4-phase clocking at the HDL level using two ac clocks (ac1 with a 0 degree shift, and ac2 with a -90 degree shift) and a dc bias (modeled as a 0 to 1 transition at the beginning of simulation).

Digital model

```
1  `timescale 1ps/10fs
2
3  module biasDir_b(xin, xout, dcin, dcout, gatex);
4
5    parameter propagationDelay = 0.198;
6
7    inout xin, xout;
8    inout dcin, dcout;
9    output gatex;
10   reg xiFrst, xoFrst, dciFrst, dcoFrst, xiReg, xoReg, dciReg, dcoReg;
11   reg fndDirx, fndDirdc, gatex, sameDir, clock;
12
13   initial begin
14     xiFrst = 1'b0;
15     xoFrst = 1'b0;
16     dciFrst = 1'b0;
17     dcoFrst = 1'b0;
18     xiReg  = 1'b0;
19     xoReg  = 1'b0;
20     dciReg = 1'b0;
21     dcoReg = 1'b0;
22     fndDirx = 1'b0;
23     fndDirdc = 1'b0;
24     gatex  = 1'b0;
25     sameDir = 1'b0;
26     clock = 1'b0;
27   end
28
29   always@(xin or xout or dcin or dcout) begin
30     if((xin==1'b1 || xin==1'b0) && !fndDirx) begin
31       xiFrst = 1'b1;
32       fndDirx = 1'b1;
33     end
34
35     if((xout==1'b1 || xout==1'b0) && !fndDirx) begin
36       xoFrst = 1'b1;
37       fndDirx = 1'b1;
38     end
39
40     if((dcin==1'b1) && !fndDirdc) begin
41       dciFrst = 1'b1;
42       fndDirdc = 1'b1;
43     end
44
45     if((dcout==1'b1) && !fndDirdc) begin
```

```
46        dcoFrst = 1'b1;
47        fndDirdc = 1'b1;
48      end
49
50      xiReg <= #propagationDelay xin;
51      xoReg <= #propagationDelay xout;
52      dciReg <= dcin;
53      dcoReg <= dcout;
54      clock <= xiFrst ? xin : xout;
55      sameDir <= dciFrst ^~ xiFrst; //bug: was previously xor; should be xnor
56      gatex <= sameDir? clock : !clock;
57    end
58
59    assign xin  = xoFrst ? xoReg : 1'bz;
60    assign xout = xiFrst ? xiReg : 1'bz;
61    assign dcin = dcoFrst ? dcoReg : 1'bz;
62    assign dcout = dciFrst ? dciReg : 1'bz;
63
64  endmodule
```

Listing 2.37: HDL ac/dc interface Verilog model code.

2.6.5 HDL dc interface

Verilog module 'biasDC.v' is an internal module for special cells such as the `storage_gate` which only have a dc bias line. This module does not apply any special normalization but it is necessary for resolving the bi-directional ports of the dc bias in HDL gate-level modeling.

Digital model

```
1  `timescale 1ps/10fs
2
3  module biasDC(a, b);
4
5  parameter propagationDelay = 0.033;
6  parameter length = 1;
7
8  inout a, b;
9
10 reg aFrst, bFrst, aReg, bReg, fndDir;
11
12 initial begin
13   aFrst = 1'b0;
14   bFrst = 1'b0;
15   aReg  = 1'b0;
16   bReg  = 1'b0;
17   fndDir = 1'b0;
18 end
19
20 always@(a or b) begin
21   if((a==1'b1 || a==1'b0) && !fndDir) begin
22     aFrst = 1'b1;
23     fndDir = 1'b1;
24   end
25
26   if((b==1'b1 || b==1'b0) && !fndDir) begin
27     bFrst = 1'b1;
28     fndDir = 1'b1;
29   end
30
31   aReg <= #(propagationDelay*length) a;
32   bReg <= #(propagationDelay*length) b;
33 end
34
35 assign a  = bFrst ? bReg : 1'bz;
36 assign b  = aFrst ? aReg : 1'bz;
37
38 endmodule
```

Listing 2.38: HDL dc interface Verilog model code.

2.7 Standard Delay Format (SDF)

Included in the cell library under the `verilog` sub-directory, we include a global Standard Delay Format (SDF) file which includes timing parameters extracted from AQFPTX [8]. This SDF file is for the standard 5 GHz clock rate under nominal bias levels. The listing is shown in Listing 2.39.

```
1
2  (DELAYFILE
3    (SDFVERSION "4.0")
4    (DESIGN "top")
5    (DATE "Thursday September 29 GMT 2022")
6    (VENDOR "")
7    (PROGRAM "AQFPTX")
8    (VERSION "1.4")
9    (DIVIDER /)
10   (PROCESS "mit")
11   (TEMPERATURE 1:2:4)
12   (TIMESCALE 1ps)
13   (CELL
14       (CELLTYPE "inv")
15       (INSTANCE *)
16         (DELAY
17             (ABSOLUTE
18                 (COND xin
19                     (IOPATH xin q (13.33))
20                 )
21             )
22         )
23       (TIMINGCHECK
24           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
25           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
26           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
27           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
28
29       )
30   )
31
32   (CELL
33       (CELLTYPE "and2_nn")
34       (INSTANCE *)
35         (DELAY
36             (ABSOLUTE
37                 (COND xin
38                     (IOPATH xin q (13.33))
39                 )
40             )
41         )
42       (TIMINGCHECK
43           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
44           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
45           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
46           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
47           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
48           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
49           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
50           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
51
52       )
53   )
54
55   (CELL
56       (CELLTYPE "and2_np")
57       (INSTANCE *)
58         (DELAY
59             (ABSOLUTE
60                 (COND xin
61                     (IOPATH xin q (13.33))
```

```
62                )
63            )
64          )
65        (TIMINGCHECK
66            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
67            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
68            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
69            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
70            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
71            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
72            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
73            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
74
75        )
76    )
77
78    (CELL
79        (CELLTYPE "and2_pn")
80        (INSTANCE *)
81          (DELAY
82              (ABSOLUTE
83                  (COND xin
84                      (IOPATH xin q (13.33))
85                  )
86              )
87          )
88        (TIMINGCHECK
89            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
90            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
91            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
92            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
93            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
94            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
95            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
96            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
97
98        )
99    )
100
101   (CELL
102       (CELLTYPE "and2_pp")
103       (INSTANCE *)
104         (DELAY
105             (ABSOLUTE
106                 (COND xin
107                     (IOPATH xin q (13.33))
108                 )
109             )
110         )
111       (TIMINGCHECK
112           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
113           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
114           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
115           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
116           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
117           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
118           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
119           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
120
121       )
122   )
123
124   (CELL
125       (CELLTYPE "and3_nnn")
126       (INSTANCE *)
127         (DELAY
128             (ABSOLUTE
129                 (COND xin
130                     (IOPATH xin q (13.33))
131                 )
```

```
            )
        )
      (TIMINGCHECK
          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
          (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
          (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
          (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
          (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
      )
  )

  (CELL
      (CELLTYPE "and3_nnp")
      (INSTANCE *)
        (DELAY
            (ABSOLUTE
                (COND xin
                    (IOPATH xin q (13.33))
                )
            )
        )
      (TIMINGCHECK
          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
          (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
          (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
          (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
          (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
      )
  )

  (CELL
      (CELLTYPE "and3_npn")
      (INSTANCE *)
        (DELAY
            (ABSOLUTE
                (COND xin
                    (IOPATH xin q (13.33))
                )
            )
        )
      (TIMINGCHECK
          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
          (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
          (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
          (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
          (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
      )
  )

```

```
(CELL
    (CELLTYPE "and3_npp")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
        (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
        (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
        (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
        (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
    )
)

(CELL
    (CELLTYPE "and3_pnn")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
        (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
        (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
        (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
        (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
    )
)

(CELL
    (CELLTYPE "and3_pnp")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
```

```
272           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
273           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
274           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
275           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
276           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
277       )
278   )
279
280   (CELL
281       (CELLTYPE "and3_ppn")
282       (INSTANCE *)
283         (DELAY
284             (ABSOLUTE
285                 (COND xin
286                     (IOPATH xin q (13.33))
287                 )
288             )
289         )
290       (TIMINGCHECK
291           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
292           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
293           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
294           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
295           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
296           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
297           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
298           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
299           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
300           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
301           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
302           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
303       )
304   )
305
306   (CELL
307       (CELLTYPE "and3_ppp")
308       (INSTANCE *)
309         (DELAY
310             (ABSOLUTE
311                 (COND xin
312                     (IOPATH xin q (13.33))
313                 )
314             )
315         )
316       (TIMINGCHECK
317           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
318           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
319           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
320           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
321           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
322           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
323           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
324           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
325           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
326           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
327           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
328           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
329       )
330   )
331
332   (CELL
333       (CELLTYPE "bfr")
334       (INSTANCE *)
335         (DELAY
336             (ABSOLUTE
337                 (COND xin
338                     (IOPATH xin q (13.33))
339                 )
340             )
341         )
```

```
342         (TIMINGCHECK
343             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
344             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
345             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
346             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
347 
348         )
349     )
350 
351     (CELL
352         (CELLTYPE "const0")
353         (INSTANCE *)
354         (DELAY
355             (ABSOLUTE
356                 (COND xin
357                     (IOPATH xin q (13.33))
358                 )
359             )
360         )
361     )
362 
363     (CELL
364         (CELLTYPE "const1")
365         (INSTANCE *)
366         (DELAY
367             (ABSOLUTE
368                 (COND xin
369                     (IOPATH xin q (13.33))
370                 )
371             )
372         )
373     )
374 
375     (CELL
376         (CELLTYPE "maj3_nnn")
377         (INSTANCE *)
378           (DELAY
379               (ABSOLUTE
380                   (COND xin
381                       (IOPATH xin q (13.33))
382                   )
383               )
384           )
385         (TIMINGCHECK
386             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
387             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
388             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
389             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
390             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
391             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
392             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
393             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
394             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
395             (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
396             (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
397             (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
398         )
399     )
400 
401     (CELL
402         (CELLTYPE "maj3_nnp")
403         (INSTANCE *)
404           (DELAY
405               (ABSOLUTE
406                   (COND xin
407                       (IOPATH xin q (13.33))
408                   )
409               )
410           )
411         (TIMINGCHECK
```

```
412           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
413           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
414           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
415           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
416           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
417           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
418           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
419           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
420           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
421           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
422           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
423           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
424       )
425   )
426
427   (CELL
428       (CELLTYPE "maj3_npn")
429       (INSTANCE *)
430         (DELAY
431             (ABSOLUTE
432                 (COND xin
433                     (IOPATH xin q (13.33))
434                 )
435             )
436         )
437       (TIMINGCHECK
438           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
439           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
440           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
441           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
442           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
443           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
444           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
445           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
446           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
447           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
448           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
449           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
450       )
451   )
452
453   (CELL
454       (CELLTYPE "maj3_npp")
455       (INSTANCE *)
456         (DELAY
457             (ABSOLUTE
458                 (COND xin
459                     (IOPATH xin q (13.33))
460                 )
461             )
462         )
463       (TIMINGCHECK
464           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
465           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
466           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
467           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
468           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
469           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
470           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
471           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
472           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
473           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
474           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
475           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
476       )
477   )
478
479
480   (CELL
481       (CELLTYPE "maj3_pnn")
```

```
482         (INSTANCE *)
483           (DELAY
484               (ABSOLUTE
485                   (COND xin
486                       (IOPATH xin q (13.33))
487                   )
488               )
489           )
490         (TIMINGCHECK
491             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
492             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
493             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
494             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
495             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
496             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
497             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
498             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
499             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
500             (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
501             (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
502             (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
503         )
504     )
505
506     (CELL
507         (CELLTYPE "maj3_pnp")
508         (INSTANCE *)
509           (DELAY
510               (ABSOLUTE
511                   (COND xin
512                       (IOPATH xin q (13.33))
513                   )
514               )
515           )
516         (TIMINGCHECK
517             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
518             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
519             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
520             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
521             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
522             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
523             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
524             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
525             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
526             (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
527             (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
528             (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
529         )
530     )
531
532     (CELL
533         (CELLTYPE "maj3_ppn")
534         (INSTANCE *)
535           (DELAY
536               (ABSOLUTE
537                   (COND xin
538                       (IOPATH xin q (13.33))
539                   )
540               )
541           )
542         (TIMINGCHECK
543             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
544             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
545             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
546             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
547             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
548             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
549             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
550             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
551             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
```

```
552           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
553           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
554           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
555       )
556   )
557
558   (CELL
559       (CELLTYPE "maj3_ppp")
560       (INSTANCE *)
561         (DELAY
562             (ABSOLUTE
563                 (COND xin
564                     (IOPATH xin q (13.33))
565                 )
566             )
567         )
568       (TIMINGCHECK
569           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
570           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
571           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
572           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
573           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
574           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
575           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
576           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
577           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
578           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
579           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
580           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
581       )
582   )
583
584   (CELL
585       (CELLTYPE "maj5_nnnnn")
586       (INSTANCE *)
587         (DELAY
588             (ABSOLUTE
589                 (COND xin
590                     (IOPATH xin q (13.33))
591                 )
592             )
593         )
594       (TIMINGCHECK
595           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
596           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
597           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
598           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
599           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
600           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
601           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
602           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
603           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
604           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
605           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
606           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
607           (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
608           (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
609           (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
610           (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
611           (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
612           (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
613           (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
614           (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
615       )
616   )
617
618   (CELL
619       (CELLTYPE "maj5_nnnnp")
620       (INSTANCE *)
621         (DELAY
```

```
622             (ABSOLUTE
623                 (COND xin
624                     (IOPATH xin q (13.33))
625                 )
626             )
627         )
628         (TIMINGCHECK
629             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
630             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
631             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
632             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
633             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
634             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
635             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
636             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
637             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
638             (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
639             (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
640             (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
641             (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
642             (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
643             (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
644             (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
645             (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
646             (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
647             (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
648             (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
649         )
650     )
651
652     (CELL
653         (CELLTYPE "maj5_nnnpn")
654         (INSTANCE *)
655           (DELAY
656             (ABSOLUTE
657                 (COND xin
658                     (IOPATH xin q (13.33))
659                 )
660             )
661           )
662         (TIMINGCHECK
663             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
664             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
665             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
666             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
667             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
668             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
669             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
670             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
671             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
672             (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
673             (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
674             (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
675             (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
676             (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
677             (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
678             (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
679             (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
680             (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
681             (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
682             (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
683         )
684     )
685
686     (CELL
687         (CELLTYPE "maj5_nnnpp")
688         (INSTANCE *)
689           (DELAY
690             (ABSOLUTE
691                 (COND xin
```

```
                    (IOPATH xin q (13.33))
                )
            )
        )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
        (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
        (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
        (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
        (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
        (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
        (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
        (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
        (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
        (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
        (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
    )
)

(CELL
    (CELLTYPE "maj5_nnpnn")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
        (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
        (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
        (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
        (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
        (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
        (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
        (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
        (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
        (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
        (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
    )
)

(CELL
    (CELLTYPE "maj5_nnpnp")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
```

```
762             )
763         )
764         (TIMINGCHECK
765             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
766             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
767             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
768             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
769             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
770             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
771             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
772             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
773             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
774             (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
775             (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
776             (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
777             (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
778             (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
779             (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
780             (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
781             (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
782             (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
783             (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
784             (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
785         )
786     )
787
788     (CELL
789         (CELLTYPE "maj5_nnppn")
790         (INSTANCE *)
791           (DELAY
792               (ABSOLUTE
793                   (COND xin
794                       (IOPATH xin q (13.33))
795                   )
796               )
797           )
798         (TIMINGCHECK
799             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
800             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
801             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
802             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
803             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
804             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
805             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
806             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
807             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
808             (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
809             (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
810             (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
811             (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
812             (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
813             (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
814             (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
815             (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
816             (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
817             (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
818             (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
819         )
820     )
821
822     (CELL
823         (CELLTYPE "maj5_nnppp")
824         (INSTANCE *)
825           (DELAY
826               (ABSOLUTE
827                   (COND xin
828                       (IOPATH xin q (13.33))
829                   )
830               )
831           )
```

```
832         (TIMINGCHECK
833             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
834             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
835             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
836             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
837             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
838             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
839             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
840             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
841             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
842             (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
843             (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
844             (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
845             (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
846             (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
847             (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
848             (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
849             (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
850             (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
851             (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
852             (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
853         )
854     )
855 
856     (CELL
857         (CELLTYPE "maj5_npnnn")
858         (INSTANCE *)
859           (DELAY
860               (ABSOLUTE
861                   (COND xin
862                       (IOPATH xin q (13.33))
863                   )
864               )
865           )
866         (TIMINGCHECK
867             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
868             (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
869             (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
870             (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
871             (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
872             (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
873             (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
874             (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
875             (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
876             (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
877             (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
878             (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
879             (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
880             (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
881             (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
882             (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
883             (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
884             (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
885             (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
886             (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
887         )
888     )
889 
890     (CELL
891         (CELLTYPE "maj5_npnnp")
892         (INSTANCE *)
893           (DELAY
894               (ABSOLUTE
895                   (COND xin
896                       (IOPATH xin q (13.33))
897                   )
898               )
899           )
900         (TIMINGCHECK
901             (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
```

```
902           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
903           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
904           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
905           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
906           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
907           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
908           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
909           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
910           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
911           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
912           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
913           (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
914           (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
915           (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
916           (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
917           (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
918           (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
919           (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
920           (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
921       )
922   )
923
924   (CELL
925       (CELLTYPE "maj5_npnpn")
926       (INSTANCE *)
927         (DELAY
928             (ABSOLUTE
929                 (COND xin
930                     (IOPATH xin q (13.33))
931                 )
932             )
933         )
934       (TIMINGCHECK
935           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
936           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
937           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
938           (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
939           (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
940           (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
941           (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
942           (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
943           (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
944           (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
945           (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
946           (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
947           (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
948           (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
949           (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
950           (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
951           (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
952           (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
953           (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
954           (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
955       )
956   )
957
958   (CELL
959       (CELLTYPE "maj5_npnpp")
960       (INSTANCE *)
961         (DELAY
962             (ABSOLUTE
963                 (COND xin
964                     (IOPATH xin q (13.33))
965                 )
966             )
967         )
968       (TIMINGCHECK
969           (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
970           (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
971           (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
```

```
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
          (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
          (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
          (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
          (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
          (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
          (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
          (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
          (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
          (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
          (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
      )
  )

  (CELL
      (CELLTYPE "maj5_nppnn")
      (INSTANCE *)
        (DELAY
            (ABSOLUTE
                (COND xin
                    (IOPATH xin q (13.33))
                )
            )
        )
      (TIMINGCHECK
          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
          (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
          (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
          (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
          (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
          (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
          (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
          (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
          (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
          (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
          (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
      )
  )

  (CELL
      (CELLTYPE "maj5_nppnp")
      (INSTANCE *)
        (DELAY
            (ABSOLUTE
                (COND xin
                    (IOPATH xin q (13.33))
                )
            )
        )
      (TIMINGCHECK
          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
```

```
1042            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1043            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1044            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1045            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1046            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1047            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1048            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1049            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1050            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1051            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1052            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1053            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1054            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1055            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1056            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1057        )
1058    )
1059
1060    (CELL
1061        (CELLTYPE "maj5_npppn")
1062        (INSTANCE *)
1063          (DELAY
1064              (ABSOLUTE
1065                  (COND xin
1066                      (IOPATH xin q (13.33))
1067                  )
1068              )
1069          )
1070        (TIMINGCHECK
1071            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1072            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1073            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1074            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1075            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1076            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1077            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1078            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1079            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1080            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1081            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1082            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1083            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1084            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1085            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1086            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1087            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1088            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1089            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1090            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1091        )
1092    )
1093
1094    (CELL
1095        (CELLTYPE "maj5_npppp")
1096        (INSTANCE *)
1097          (DELAY
1098              (ABSOLUTE
1099                  (COND xin
1100                      (IOPATH xin q (13.33))
1101                  )
1102              )
1103          )
1104        (TIMINGCHECK
1105            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1106            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1107            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1108            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1109            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1110            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1111            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
```

```
1112            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1113            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1114            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1115            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1116            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1117            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1118            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1119            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1120            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1121            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1122            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1123            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1124            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1125        )
1126    )
1127
1128    (CELL
1129        (CELLTYPE "maj5_pnnnn")
1130        (INSTANCE *)
1131          (DELAY
1132              (ABSOLUTE
1133                  (COND xin
1134                      (IOPATH xin q (13.33))
1135                  )
1136              )
1137          )
1138        (TIMINGCHECK
1139            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1140            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1141            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1142            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1143            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1144            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1145            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1146            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1147            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1148            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1149            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1150            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1151            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1152            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1153            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1154            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1155            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1156            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1157            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1158            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1159        )
1160    )
1161
1162    (CELL
1163        (CELLTYPE "maj5_pnnnp")
1164        (INSTANCE *)
1165          (DELAY
1166              (ABSOLUTE
1167                  (COND xin
1168                      (IOPATH xin q (13.33))
1169                  )
1170              )
1171          )
1172        (TIMINGCHECK
1173            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1174            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1175            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1176            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1177            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1178            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1179            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1180            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1181            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
```

```
1182            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1183            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1184            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1185            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1186            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1187            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1188            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1189            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1190            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1191            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1192            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1193        )
1194    )
1195
1196    (CELL
1197        (CELLTYPE "maj5_pnnpn")
1198        (INSTANCE *)
1199          (DELAY
1200              (ABSOLUTE
1201                  (COND xin
1202                      (IOPATH xin q (13.33))
1203                  )
1204              )
1205          )
1206        (TIMINGCHECK
1207            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1208            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1209            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1210            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1211            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1212            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1213            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1214            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1215            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1216            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1217            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1218            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1219            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1220            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1221            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1222            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1223            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1224            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1225            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1226            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1227        )
1228    )
1229
1230    (CELL
1231        (CELLTYPE "maj5_pnnpp")
1232        (INSTANCE *)
1233          (DELAY
1234              (ABSOLUTE
1235                  (COND xin
1236                      (IOPATH xin q (13.33))
1237                  )
1238              )
1239          )
1240        (TIMINGCHECK
1241            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1242            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1243            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1244            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1245            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1246            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1247            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1248            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1249            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1250            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1251            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
```

```
          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
          (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
          (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
          (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
          (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
          (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
          (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
      )
  )

  (CELL
      (CELLTYPE "maj5_pnpnn")
      (INSTANCE *)
        (DELAY
            (ABSOLUTE
                (COND xin
                    (IOPATH xin q (13.33))
                )
            )
        )
      (TIMINGCHECK
          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
          (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
          (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
          (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
          (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
          (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
          (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
          (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
          (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
          (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
          (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
      )
  )

  (CELL
      (CELLTYPE "maj5_pnpnp")
      (INSTANCE *)
        (DELAY
            (ABSOLUTE
                (COND xin
                    (IOPATH xin q (13.33))
                )
            )
        )
      (TIMINGCHECK
          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
          (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
          (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
          (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
          (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
          (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
```

```
1322            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1323            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1324            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1325            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1326            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1327            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1328            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1329        )
1330    )
1331
1332    (CELL
1333        (CELLTYPE "maj5_pnppn")
1334        (INSTANCE *)
1335          (DELAY
1336              (ABSOLUTE
1337                  (COND xin
1338                      (IOPATH xin q (13.33))
1339                  )
1340              )
1341          )
1342        (TIMINGCHECK
1343            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1344            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1345            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1346            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1347            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1348            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1349            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1350            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1351            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1352            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1353            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1354            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1355            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1356            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1357            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1358            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1359            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1360            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1361            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1362            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1363        )
1364    )
1365
1366    (CELL
1367        (CELLTYPE "maj5_pnppp")
1368        (INSTANCE *)
1369          (DELAY
1370              (ABSOLUTE
1371                  (COND xin
1372                      (IOPATH xin q (13.33))
1373                  )
1374              )
1375          )
1376        (TIMINGCHECK
1377            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1378            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1379            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1380            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1381            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1382            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1383            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1384            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1385            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1386            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1387            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1388            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1389            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1390            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1391            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
```

```
1392            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1393            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1394            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1395            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1396            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1397        )
1398    )
1399
1400    (CELL
1401        (CELLTYPE "maj5_ppnnn")
1402        (INSTANCE *)
1403          (DELAY
1404              (ABSOLUTE
1405                  (COND xin
1406                      (IOPATH xin q (13.33))
1407                  )
1408              )
1409          )
1410        (TIMINGCHECK
1411            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1412            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1413            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1414            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1415            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1416            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1417            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1418            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1419            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1420            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1421            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1422            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1423            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1424            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1425            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1426            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1427            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1428            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1429            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1430            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1431        )
1432    )
1433
1434    (CELL
1435        (CELLTYPE "maj5_ppnnp")
1436        (INSTANCE *)
1437          (DELAY
1438              (ABSOLUTE
1439                  (COND xin
1440                      (IOPATH xin q (13.33))
1441                  )
1442              )
1443          )
1444        (TIMINGCHECK
1445            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1446            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1447            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1448            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1449            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1450            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1451            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1452            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1453            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1454            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1455            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1456            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1457            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1458            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1459            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1460            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1461            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
```

```
1462            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1463            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1464            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1465        )
1466    )
1467
1468    (CELL
1469        (CELLTYPE "maj5_ppnpn")
1470        (INSTANCE *)
1471          (DELAY
1472              (ABSOLUTE
1473                  (COND xin
1474                      (IOPATH xin q (13.33))
1475                  )
1476              )
1477          )
1478        (TIMINGCHECK
1479            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1480            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1481            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1482            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1483            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1484            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1485            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1486            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1487            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1488            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1489            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1490            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1491            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1492            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1493            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1494            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1495            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1496            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1497            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1498            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1499        )
1500    )
1501
1502    (CELL
1503        (CELLTYPE "maj5_ppnpp")
1504        (INSTANCE *)
1505          (DELAY
1506              (ABSOLUTE
1507                  (COND xin
1508                      (IOPATH xin q (13.33))
1509                  )
1510              )
1511          )
1512        (TIMINGCHECK
1513            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1514            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1515            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1516            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1517            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1518            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1519            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1520            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1521            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1522            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1523            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1524            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1525            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1526            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1527            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1528            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1529            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1530            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1531            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
```

```
1532            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1533        )
1534    )
1535
1536    (CELL
1537        (CELLTYPE "maj5_pppnn")
1538        (INSTANCE *)
1539          (DELAY
1540              (ABSOLUTE
1541                  (COND xin
1542                      (IOPATH xin q (13.33))
1543                  )
1544              )
1545          )
1546        (TIMINGCHECK
1547            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1548            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1549            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1550            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1551            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1552            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1553            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1554            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1555            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1556            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1557            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1558            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1559            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1560            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1561            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1562            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1563            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1564            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1565            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1566            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1567        )
1568    )
1569
1570    (CELL
1571        (CELLTYPE "maj5_pppnp")
1572        (INSTANCE *)
1573          (DELAY
1574              (ABSOLUTE
1575                  (COND xin
1576                      (IOPATH xin q (13.33))
1577                  )
1578              )
1579          )
1580        (TIMINGCHECK
1581            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1582            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1583            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1584            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1585            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1586            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1587            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1588            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1589            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1590            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1591            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1592            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1593            (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1594            (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1595            (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1596            (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1597            (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1598            (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1599            (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1600            (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1601        )
```

```
1602  )
1603
1604  (CELL
1605      (CELLTYPE "maj5_ppppn")
1606      (INSTANCE *)
1607        (DELAY
1608            (ABSOLUTE
1609                (COND xin
1610                    (IOPATH xin q (13.33))
1611                )
1612            )
1613        )
1614      (TIMINGCHECK
1615          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1616          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1617          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1618          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1619          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1620          (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1621          (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1622          (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1623          (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1624          (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1625          (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1626          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1627          (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1628          (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1629          (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1630          (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1631          (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1632          (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1633          (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1634          (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1635      )
1636  )
1637
1638  (CELL
1639      (CELLTYPE "maj5_ppppp")
1640      (INSTANCE *)
1641        (DELAY
1642            (ABSOLUTE
1643                (COND xin
1644                    (IOPATH xin q (13.33))
1645                )
1646            )
1647        )
1648      (TIMINGCHECK
1649          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1650          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1651          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1652          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1653          (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1654          (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1655          (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1656          (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1657          (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1658          (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1659          (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1660          (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1661          (SETUP (COND d (posedge d)) (posedge gatex) (-8.10))
1662          (SETUP (COND not_d (negedge d)) (posedge gatex) (-8.05))
1663          (HOLD (posedge gatex) (COND d (negedge d)) (10.90))
1664          (HOLD (posedge gatex) (COND not_d (posedge d)) (10.95))
1665          (SETUP (COND e (posedge e)) (posedge gatex) (-8.10))
1666          (SETUP (COND not_e (negedge e)) (posedge gatex) (-8.05))
1667          (HOLD (posedge gatex) (COND e (negedge e)) (10.90))
1668          (HOLD (posedge gatex) (COND not_e (posedge e)) (10.95))
1669      )
1670  )
1671
```

```
(CELL
    (CELLTYPE "qfpl")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
        (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
    )
)

(CELL
    (CELLTYPE "or2_nn")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
        (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))

    )
)

(CELL
    (CELLTYPE "or2_np")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
        (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))

    )
)

(CELL
    (CELLTYPE "or2_pn")
```

```
1742        (INSTANCE *)
1743          (DELAY
1744              (ABSOLUTE
1745                  (COND xin
1746                      (IOPATH xin q (13.33))
1747                  )
1748              )
1749          )
1750        (TIMINGCHECK
1751            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1752            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1753            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1754            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1755            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1756            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1757            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1758            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1759
1760        )
1761    )
1762
1763    (CELL
1764        (CELLTYPE "or2_pp")
1765        (INSTANCE *)
1766          (DELAY
1767              (ABSOLUTE
1768                  (COND xin
1769                      (IOPATH xin q (13.33))
1770                  )
1771              )
1772          )
1773        (TIMINGCHECK
1774            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1775            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1776            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1777            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1778            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1779            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1780            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1781            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1782
1783        )
1784    )
1785
1786    (CELL
1787        (CELLTYPE "or3_nnn")
1788        (INSTANCE *)
1789          (DELAY
1790              (ABSOLUTE
1791                  (COND xin
1792                      (IOPATH xin q (13.33))
1793                  )
1794              )
1795          )
1796        (TIMINGCHECK
1797            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1798            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1799            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1800            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1801            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1802            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1803            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1804            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1805            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1806            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1807            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1808            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1809        )
1810    )
1811
```

```
(CELL
    (CELLTYPE "or3_nnp")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
        (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
        (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
        (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
        (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
    )
)

(CELL
    (CELLTYPE "or3_npn")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
        (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
        (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
        (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
        (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
    )
)

(CELL
    (CELLTYPE "or3_npp")
    (INSTANCE *)
      (DELAY
          (ABSOLUTE
              (COND xin
                  (IOPATH xin q (13.33))
              )
          )
      )
    (TIMINGCHECK
        (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
        (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
        (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
        (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
        (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
```

```
1882            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1883            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1884            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1885            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1886            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1887        )
1888    )
1889
1890    (CELL
1891        (CELLTYPE "or3_pnn")
1892        (INSTANCE *)
1893          (DELAY
1894              (ABSOLUTE
1895                  (COND xin
1896                      (IOPATH xin q (13.33))
1897                  )
1898              )
1899          )
1900        (TIMINGCHECK
1901            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1902            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1903            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1904            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1905            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1906            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1907            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1908            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1909            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1910            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1911            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1912            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1913        )
1914    )
1915
1916    (CELL
1917        (CELLTYPE "or3_pnp")
1918        (INSTANCE *)
1919          (DELAY
1920              (ABSOLUTE
1921                  (COND xin
1922                      (IOPATH xin q (13.33))
1923                  )
1924              )
1925          )
1926        (TIMINGCHECK
1927            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1928            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1929            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1930            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1931            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1932            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1933            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1934            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1935            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1936            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1937            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1938            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1939        )
1940    )
1941
1942    (CELL
1943        (CELLTYPE "or3_ppn")
1944        (INSTANCE *)
1945          (DELAY
1946              (ABSOLUTE
1947                  (COND xin
1948                      (IOPATH xin q (13.33))
1949                  )
1950              )
1951          )
```

```
1952        (TIMINGCHECK
1953            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1954            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1955            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1956            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1957            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1958            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1959            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1960            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1961            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1962            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1963            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1964            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1965        )
1966    )
1967
1968    (CELL
1969        (CELLTYPE "or3_ppp")
1970        (INSTANCE *)
1971          (DELAY
1972              (ABSOLUTE
1973                  (COND xin
1974                      (IOPATH xin q (13.33))
1975                  )
1976              )
1977          )
1978        (TIMINGCHECK
1979            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
1980            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
1981            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
1982            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
1983            (SETUP (COND b (posedge b)) (posedge gatex) (-8.10))
1984            (SETUP (COND not_b (negedge b)) (posedge gatex) (-8.05))
1985            (HOLD (posedge gatex) (COND b (negedge b)) (10.90))
1986            (HOLD (posedge gatex) (COND not_b (posedge b)) (10.95))
1987            (SETUP (COND c (posedge c)) (posedge gatex) (-8.10))
1988            (SETUP (COND not_c (negedge c)) (posedge gatex) (-8.05))
1989            (HOLD (posedge gatex) (COND c (negedge c)) (10.90))
1990            (HOLD (posedge gatex) (COND not_c (posedge c)) (10.95))
1991        )
1992    )
1993
1994    (CELL
1995        (CELLTYPE "spl2")
1996        (INSTANCE *)
1997          (DELAY
1998              (ABSOLUTE
1999                  (COND xin
2000                      (IOPATH xin q0 (13.33))
2001                  )
2002                  (COND xin
2003                      (IOPATH xin q1 (13.33))
2004                  )
2005              )
2006          )
2007        (TIMINGCHECK
2008            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
2009            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
2010            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
2011            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
2012
2013        )
2014    )
2015
2016    (CELL
2017        (CELLTYPE "spl3")
2018        (INSTANCE *)
2019          (DELAY
2020              (ABSOLUTE
2021                  (COND xin
```

```
                    (IOPATH xin q0 (13.33))
                )
                (COND xin
                    (IOPATH xin q1 (13.33))
                )
                (COND xin
                    (IOPATH xin q2 (13.33))
                )
            )
        )
        (TIMINGCHECK
            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))

        )
    )

    (CELL
        (CELLTYPE "spl4")
        (INSTANCE *)
          (DELAY
              (ABSOLUTE
                  (COND xin
                      (IOPATH xin q0 (13.33))
                  )
                  (COND xin
                      (IOPATH xin q1 (13.33))
                  )
                  (COND xin
                      (IOPATH xin q2 (13.33))
                  )

                  (COND xin
                      (IOPATH xin q3 (13.33))
                  )
              )
          )
        (TIMINGCHECK
            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
        )
    )

    (CELL
        (CELLTYPE "boost1")
        (INSTANCE *)
          (DELAY
              (ABSOLUTE
                  (COND xin
                      (IOPATH xin q (13.33))
                  )
              )
          )
        (TIMINGCHECK
            (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
            (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
            (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
            (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))

        )
    )

    (CELL
        (CELLTYPE "boost2f2")
        (INSTANCE *)
          (DELAY
```

```
            (ABSOLUTE
                (COND xin
                    (IOPATH xin q0 (13.33))
                )
                (COND xin
                    (IOPATH xin q1 (13.33))
                )
            )
        )
      (TIMINGCHECK
          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))

      )
  )

  (CELL
      (CELLTYPE "boost2f4")
      (INSTANCE *)
        (DELAY
            (ABSOLUTE
                (COND xin
                    (IOPATH xin q0 (13.33))
                )
                (COND xin
                    (IOPATH xin q1 (13.33))
                )
                (COND xin
                    (IOPATH xin q2 (13.33))
                )

                (COND xin
                    (IOPATH xin q3 (13.33))
                )
            )
        )
      (TIMINGCHECK
          (SETUP (COND a (posedge a)) (posedge gatex) (-8.10))
          (SETUP (COND not_a (negedge a)) (posedge gatex) (-8.05))
          (HOLD (posedge gatex) (COND a (negedge a)) (10.90))
          (HOLD (posedge gatex) (COND not_a (posedge a)) (10.95))
      )
  )
)
```

Listing 2.39: SDF file included in the library.

Bibliography

[1] C. J. Fourie, C. L. Ayala, L. Schindler, T. Tanaka, and N. Yoshikawa, "Design and characterization of track routing architecture for RSFQ and AQFP circuits in a multilayer process," *IEEE Trans. Appl. Supercond.*, vol. 30, no. 6, pp. 1–9, Sep. 2020.

[2] N. Takeuchi, Y. Yamanashi, and N. Yoshikawa, "Adiabatic quantum-flux-parametron cell library adopting minimalist design," *J. Appl. Phys.*, vol. 117, no. 17, p. 173 912, May 2015. DOI: 10.1063/1.4919838.

[3] (2017). XicTools suite, [Online]. Available: http://www.wrcad.com/xictools/index.html.

[4] gEDA Project. (2021), [Online]. Available: http://wiki.geda-project.org/geda:gaf.

[5] M. Köfferlein. (2020). KLayout, [Online]. Available: https://www.klayout.de/.

[6] C. J. Fourie, "Full-Gate Verification of Superconducting Integrated Circuit Layouts With InductEx," *IEEE Trans. Appl. Supercond.*, vol. 25, no. 1, Feb. 2015.

[7] J. A. Delport, K. Jackman, P. Le Roux, and C. J. Fourie, "JoSIM—Superconductor SPICE Simulator," *IEEE Trans. Appl. Supercond.*, vol. 29, no. 5, pp. 1–5, 2019.

[8] C. L. Ayala, O. Chen, and N. Yoshikawa, "AQFPTX: Adiabatic Quantum-Flux-Parametron timing extraction tool," in *2019 IEEE International Superconductive Electronics Conference (ISEC)*, Jul. 2019, pp. 1–3.

[9] S. Williams. (2018). Icarus verilog, [Online]. Available: http://iverilog.icarus.com/.

[10] (2018). Gtkwave, [Online]. Available: http://gtkwave.sourceforge.net/.

[11] T. Yamae, N. Takeuchi, and N. Yoshikawa, "Systematic method to evaluate energy dissipation in adiabatic quantum-flux-parametron logic," *J. Appl. Phys.*, vol. 126, no. 17, p. 173 903, Nov. 2019.

[12] J. A. Delport. (2020). JoSIM, [Online]. Available: https://github.com/JoeyDelp/JoSIM/.