TOTAL IONIZING DOSE TEST REPORT

No. 06T-RTAX2000S-D21PH1
June 19, 2006
J.J. Wang
(650) 318-4576
jih-jong.wang@actel.com

I. SUMMARY TABLE

Parameter	Tolerance
1. Gross Functionality	Passed 300 krad (Si)
2. Power Supply Current (I_{CCA}/I_{CCI})	Post 200 krad (Si) and after 6 days room temperature annealing: every DUT passed the spec Post 300 krad (Si) and after 6 days room temperature annealing: every DUT passed I_{CCA} spec; the worst case radiation induced I_{CCI}-delta is 59 mA (the spec is 28 mA)
3. Input Threshold (V_{TIL}/V_{IH})	Passed 300 krad (Si)
4. Output Drive (V_{OL}/V_{OH})	Passed 300 krad (Si)
5. Propagation Delay	Passed 300 krad (Si) for 10% degradation criterion
6. Transition Time	Passed 300 krad (Si)

II. TOTAL IONIZING DOSE (TID) TESTING

This testing is designed on the base of an extensive database (see, for example, TID data of antifuse-based FPGA in http://www.klabs.org/ and http://www.actel.com/) accumulated from the TID testing of many generations of antifuse-based FPGAs. One distinctive quality about this testing is the bench measurement of electrical parameters. Compared to an automatic-tester measurement, the bench measurement provides lower noise, better accuracy and flexibility. The bench measurement samples pins for the measurements. However, two factors make sampling appropriate: first, the tolerance is often determined by either the I_{CC} or propagation delay, both are global parameters; second, the total dose effect is uniformly distributed across the chip.

A. Device-Under-Test (DUT) and Irradiation Parameters

Table 1 lists the DUT and irradiation parameters. During irradiation each input and most of the output is grounded through a 100k-ohm resistor; during annealing each input or output is tied to the ground or V_{CCI} with a 2.7-k ohm resistor. Appendix A contains the schematics of the irradiation-bias circuit.

Table 1 DUT and Irradiation Parameters

Part Number	RTAX2000S
Package	CQFP352
Foundry	United Microelectronics Corp.
Technology	0.15 µm CMOS
DUT Design	TOP_RTAX2000S_TID_r1
Die Lot Number	D21PH1
Quantity Tested	5
Serial Number	200 krad: 3176, 3188 300 krad: 3169, 3170, 3175
Radiation Facility	Defense Microelectronics Activity
Radiation Source	Co-60
Dose Rate (±5%)	3 krad (Si)/min
Irradiation Temperature	Room
Irradiation and Measurement Bias (V_{CCI}/V_{CCA})	Static at 3.3 V/1.5 V
IO Configuration	Single ended: LVTTL Differential pair: LVPECL

B. Test Method

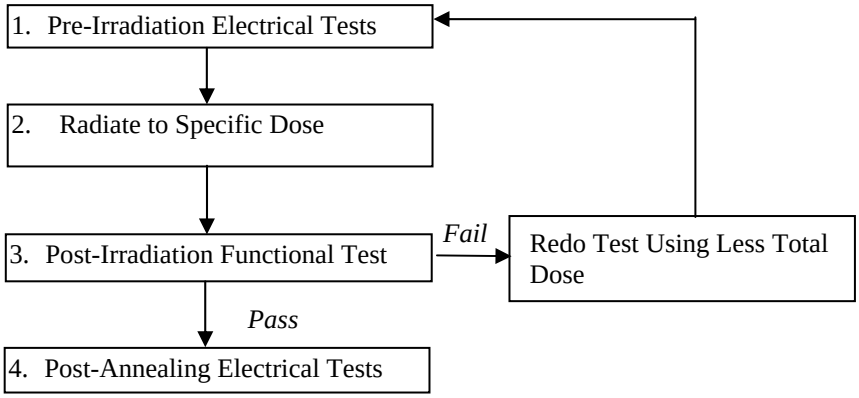


Figure 1 Parametric test flow chart

The test method generally follows the guidelines in the military standard TM1019. Figure 1 is the flow chart showing the steps for parametric tests, irradiation, and post-irradiation annealing.

The accelerated aging, or rebound test mentioned in TM1019 is unnecessary because there is no adverse time dependent effect (TDE) in products manufactured by sub-micron CMOS technology. Elevated temperature annealing reduces the effects originated from radiation-induced leakages. As indicated by testing data in the following sections, the predominant radiation effects in RTAX2000S are due to radiation-induced leakages.

Room temperature annealing is performed in this test; the duration is approximately 6 days.

C. Design and Parametric Measurements

DUTs use a high utilization generic design (RTAX2000S_TID) to test total dose effects in typical space applications. Appendix B contains the schematics and Verilog files for the logic design.

Table 2 lists each electrical parameter to be measured and the corresponding logic design. The functionality is measured on the output pin (O_BS) of a combinational buffer-string with 14,000 buffers, output pins (O_ANDP_CLKF, O_ORP_CLKF, O_FF_CLKF, O_ANDC_CLKF, O_ORC_CLKF, O_ANDP_CLKG, O_ORP_CLKG, O_FF_CLKG, O_ANDC_CLKG, O_ORC_CLKG, O_ANDP_CLKH, O_ORP_CLKH, O_FF_CLKH, O_ANDC_CLKH, O_ORC_CLKH, O_ANDP_HCLKA, O_ORP_HCLKA, O_FF_HCLKA, O_ANDC_HCLKA, and O_ORC_HCLKA) of four (4) shift registers with 10,728 bits total, and half of the output pins (OUTX0, OUTX1, OUTX2, OUTX3, OUTX4, OUTX5, OUTX6 and OUTX7) of the embedded RAM configured as 16k×16.

I_{CC} is measured on the power supply of the logic-array (I_{CCA}) and I/O (I_{CCI}) respectively. The input logic threshold (V_{IL}/V_{IH}) is measured on single-ended inputs EN8, DA, IO_I1, IO_I2, IO_I3, IO_I4, IO_I5 and IO_I6, and also on differential inputs DIO_I1P, DIO_I2P, DIO_I3P, DIO_I4P, DIO_I5P, DIO_I6P and DIO_I7P. The differential inputs are configured as LVPECL instead of LVDS. Because LVPECL uses 3.3 VDC, which is higher than 2.5 VDC used by LVDS, it is the worst case among the differential pairs. During the measurement on the differential inputs, the N (negative) side of the differential pair is biased at 1.8 V. The output-drive voltage (V_{OL}/V_{OH}) is measured on QA0. The propagation delay is measured on the output (O_BS) of the buffer string. The definition is the time delay from the triggering edge at the CLOCK input to the switching edge at the output O_BS. Both the delays of low-to-high and high-to-low output transitions are measured; the reported delay is the average of these two measurements. The transition characteristics, measured on the output O_BS, are shown as oscilloscope snapshots.

Table 2 Logic Design for Parametric Measurements

Parameters	Logic Design
1. Functionality	All key logic functions (O_BS, O_ANDP_CLKF, O_ORP_CLKF, O_FF_CLKF, O_ANDC_CLKF, O_ORC_CLKF, O_ANDP_CLKG, O_ORP_CLKG, O_FF_CLKG, O_ANDC_CLKG, O_ORC_CLKG, O_ANDP_CLKH, O_ORP_CLKH, O_FF_CLKH, O_ANDC_CLKH, O_ORC_CLKH, O_ANDP_HCLKA, O_ORP_HCLKA, O_FF_HCLKA, O_ANDC_HCLKA, and O_ORC_HCLKA), and outputs of embedded RAM (OUTX0, OUTX1, OUTX2, OUTX3, OUTX4, OUTX5, OUTX6 and OUTX7)
2. I_{CC} (I_{CCA}/I_{CCI})	DUT power supply
3. Input Threshold (V_{IL}/V_{IH})	Single ended inputs (EN8/YQ0, DA/QA0, IO_I1/IO_O1, IO_I2/IO_O2, IO_I3/IO_O3, IO_I4/IO_O4, IO_I5/IO_O5, IO_I6/IO_O6), and differential inputs (DIO_I1P/DIO_O1, DIO_I2P/DIO_O2, DIO_I3P/DIO_O3, DIO_I4P/DIO_O4, DIO_I5P/DIO_O5, DIO_I6P/DIO_O6, DIO_I7P/DIO_O7)
4. Output Drive (V_{OL}/V_{OH})	Output buffer (DA/QA0)
5. Propagation Delay	String of buffers (CLOCK to O_BS)
6. Transition Characteristic	D flip-flop output (O_BS)

III. TEST RESULTS

A. *Functionality*

Every DUT passed the pre-irradiation and post-annealing functional tests.

B. *Power Supply Current (I_{CCA} and I_{CCI})*

Figures 2 to 7 show the in-flux power supply current (I_{CC}). Table 3 summarizes the pre-irradiation, post-irradiation and post-annealing I_{CC}. The post-annealing I_{CC} for four different bit patterns in the RAM, all '0', all '1', checkerboard and inverted-checkerboard, are basically the same value.

These measurements are performed on bench using a bench board. Since there is certain amount of current due to the bench board alone, the radiation-induced current, which is the delta between the post-irradiation (or post-annealing) and pre-irradiation, is the real meaningful quantity. In spec, the allowable delta I_{CCA} is 450 mA, and the allowable delta I_{CCI} is 28 mA.

Table 3 Pre-irradiation, Post Irradiation and Post-Annealing I_{CC}

DUT	Total Dose (krad)	I_{CCA} (mA)				I_{CCI} (mA)			
		Pre-irrad	Post-irrad	Post-ann	Post-ann Δ	Pre-irrad	Post-irrad	Post-ann	Post-ann Δ
3169	300	6	173	33	27	23	152	71	48
3170	300	3	81	7	4	21	158	71	50
3175	300	1	65	5	4	21	160	80	59
3176	200	4	4	4	0	23	62	41	18
3188	200	5	5	3	-2	22	77	48	26

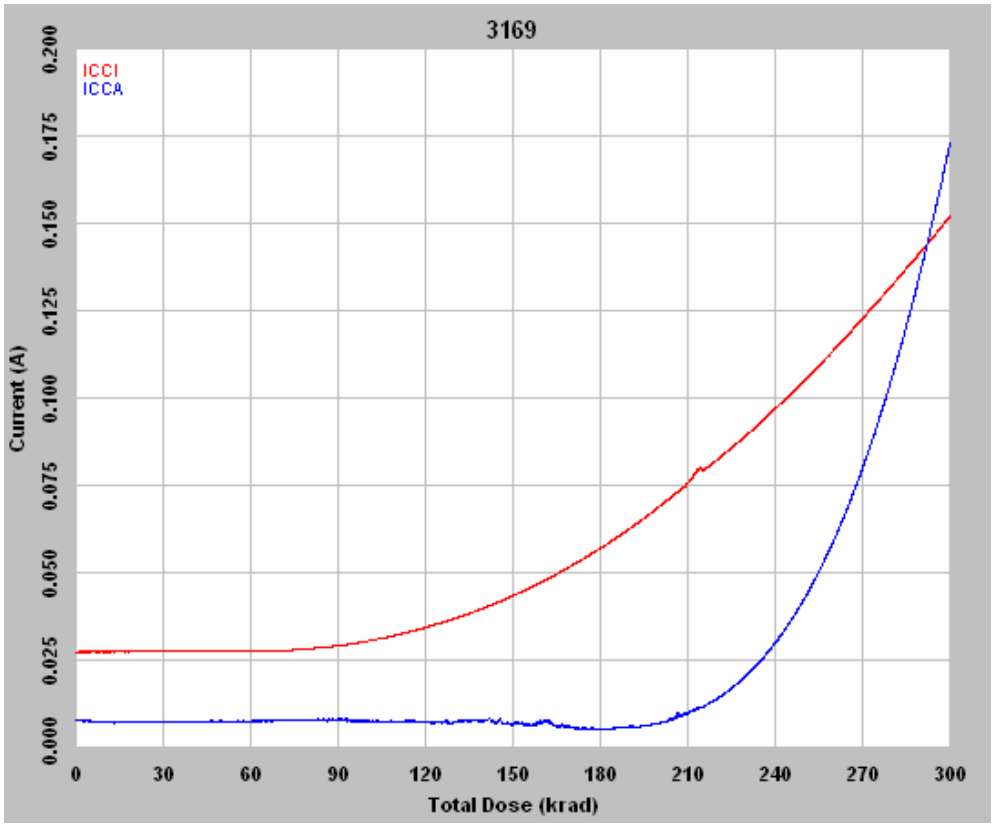


Figure 2 DUT 3169 in-flux I_{CCA} and I_{CCI}

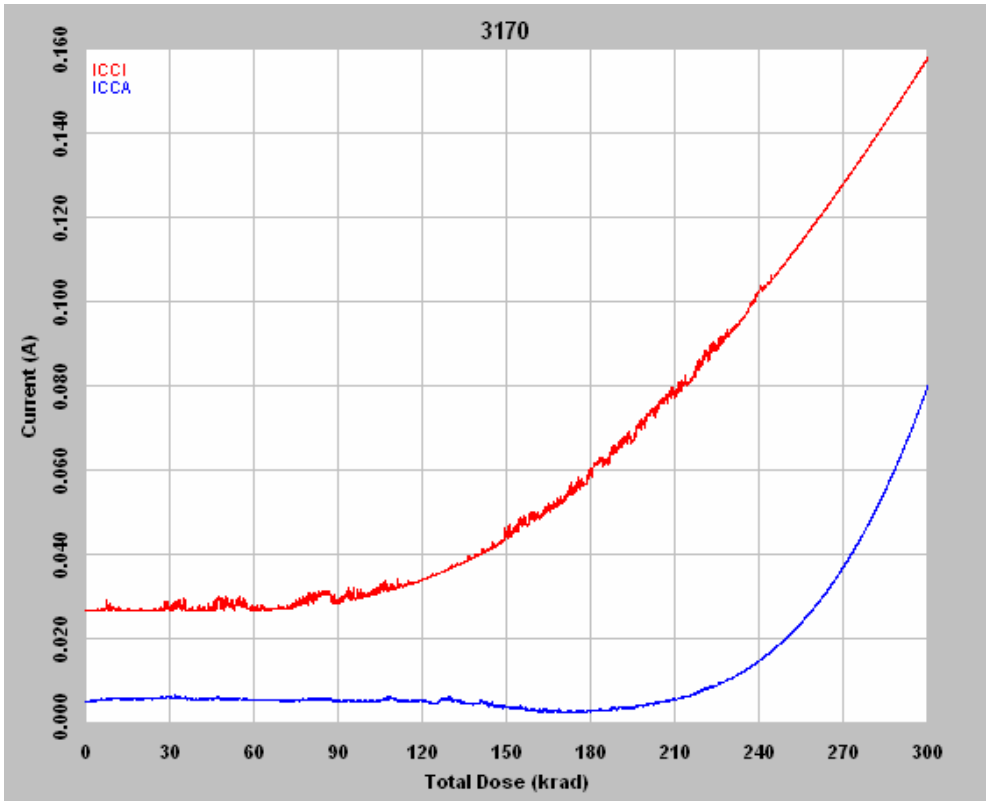


Figure 3 DUT 3170 in-flux I_{CCA} and I_{CCI}

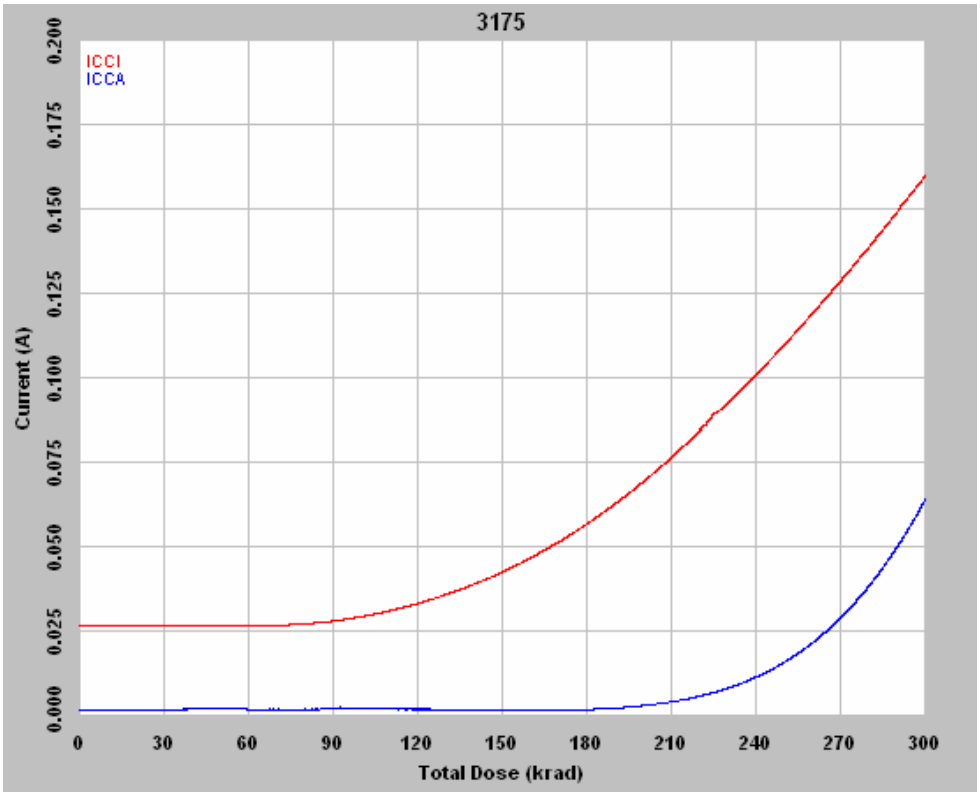


Figure 4 DUT 3175 in-flux I_{CCA} and I_{CCI}

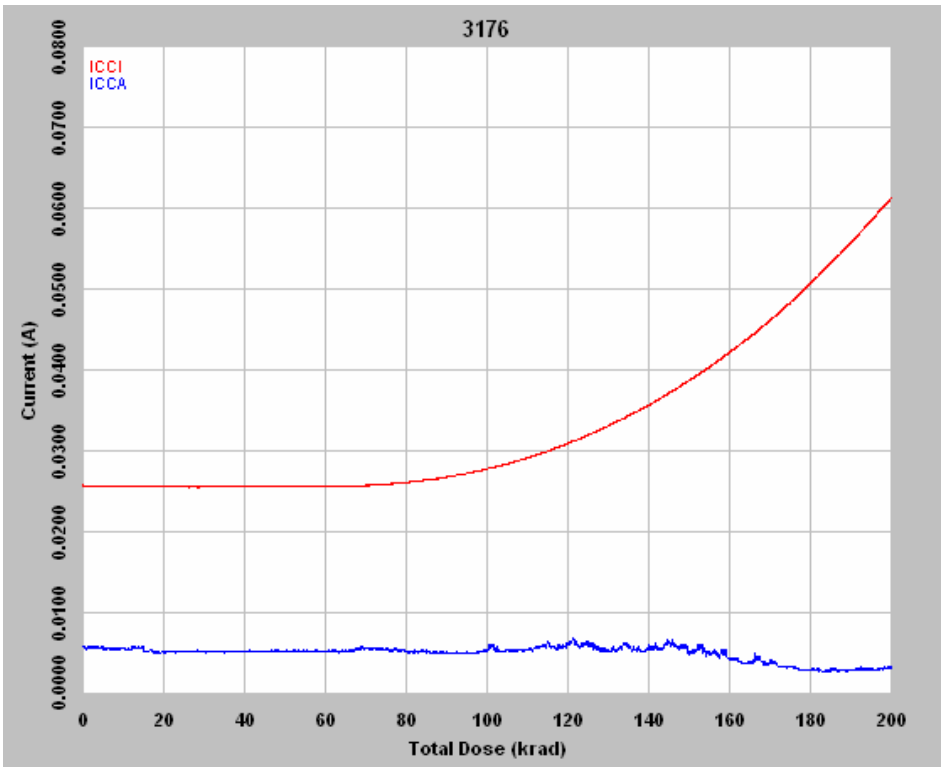


Figure 5 DUT 3176 in-flux I_{CCA} and I_{CCI}

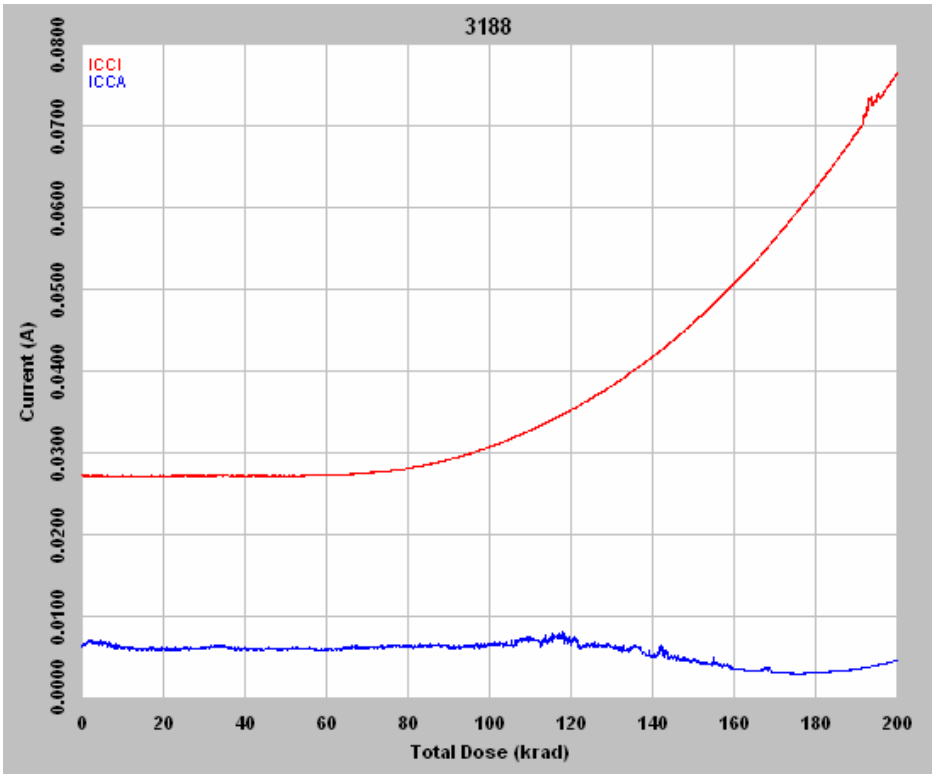


Figure 6 DUT 3188 in-flux I_{CCA} and I_{CCI}

C. *Single-Ended Input Logic Threshold (V_{IL}/V_{IH})*

Table 4 lists the pre-irradiation and post-annealing single-ended input logic threshold. All data are within the spec limits, and the post-annealing change in every case is less than 10%.

Table 4a Pre-Irradiation and Post-Annealing Input Thresholds

In/Out Pin:		IO_I1/IO_O1				IO_I2/IO_O2			
DUT	Total Dose	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
		V_{IL} (V)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
3169	300 krad	1.52	1.50	1.48	1.49	1.51	1.51	1.49	1.48
3170	300 krad	1.51	1.51	1.51	1.47	1.51	1.51	1.49	1.48
3175	300 krad	1.52	1.51	1.50	1.48	1.52	1.51	1.49	1.51
3176	200 krad	1.51	1.50	1.49	1.48	1.52	1.53	1.49	1.50
3188	200 krad	1.53	1.50	1.51	1.49	1.53	1.51	1.51	1.48

Table 4b Pre-Irradiation and Post-Annealing Input Thresholds

In/Out Pin:		IO_I3/IO_O3				IO_I4/IO_O4			
DUT	Total Dose	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
		V_{IL} (V)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
3169	300 krad	1.48	1.48	1.47	1.47	1.51	1.50	1.48	1.49
3170	300 krad	1.48	1.47	1.53	1.48	1.51	1.50	1.49	1.49
3175	300 krad	1.48	1.52	1.48	1.48	1.51	1.50	1.50	1.50
3176	200 krad	1.48	1.48	1.48	1.53	1.52	1.51	1.49	1.49
3188	200 krad	1.48	1.47	1.53	1.48	1.50	1.50	1.50	1.48

Table 4c Pre-Irradiation and Post-Annealing Input Thresholds

In/Out Pin:		IO_I5/IO_O5				IO_I6/IO_O6			
DUT	Total Dose	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
		V_{IL} (V)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
3169	300 krad	1.51	1.50	1.48	1.49	1.51	1.50	1.51	1.49
3170	300 krad	1.51	1.50	1.49	1.47	1.51	1.51	1.50	1.49
3175	300 krad	1.51	1.51	1.49	1.48	1.52	1.51	1.50	1.48
3176	200 krad	1.51	1.50	1.48	1.47	1.52	1.51	1.51	1.49
3188	200 krad	1.52	1.50	1.50	1.50	1.52	1.51	1.51	1.50

Table 4d Pre-Irradiation and Post-Annealing Input Thresholds

In/Out Pin:		DA/QA0				EN8/YQ0			
DUT	Total Dose	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
		V_{IL} (V)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
3169	300 krad	1.51	1.49	1.49	1.47	1.51	1.48	1.48	1.46
3170	300 krad	1.51	1.49	1.48	1.47	1.51	1.48	1.48	1.47
3175	300 krad	1.52	1.49	1.50	1.47	1.51	1.48	1.47	1.47
3176	200 krad	1.52	1.50	1.49	1.48	1.52	1.49	1.49	1.46
3188	200 krad	1.51	1.49	1.49	1.48	1.51	1.49	1.48	1.46

D. Differential Input (LVPECL) Threshold Voltage (V_{IL}/V_{IH})

Table 5 lists the pre-irradiation and post-annealing LVPECL differential input threshold voltages. These data show negligible changes of LVPECL input threshold voltages for the 200-krad or 300-krad irradiation.

Table 5a Pre-Irradiation and Post-Annealing Differential Input Thresholds

In/Out Pin:		DIO_I1P/DIO_O1				DIO_I2P/DIO_O2			
DUT	Total Dose	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
		V_{IL} (V)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
3169	300 krad	1.78	1.78	1.77	1.78	1.79	1.79	1.78	1.78
3170	300 krad	1.78	1.79	1.78	1.78	1.79	1.79	1.78	1.78
3175	300 krad	1.78	1.79	1.78	1.78	1.79	1.79	1.78	1.77
3176	200 krad	1.78	1.79	1.78	1.78	1.79	1.79	1.78	1.78
3188	200 krad	1.78	1.78	1.78	1.78	1.79	1.78	1.76	1.76

Table 5b Pre-Irradiation and Post-Annealing Differential Input Thresholds

In/Out Pin:		DIO_I3P/DIO_O3				DIO_I4P/DIO_O4			
DUT	Total Dose	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
		V_{IL} (V)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
3169	300 krad	1.78	1.78	1.78	1.78	1.79	1.80	1.78	1.78
3170	300 krad	1.79	1.79	1.78	1.78	1.80	1.80	1.78	1.79
3175	300 krad	1.79	1.79	1.78	1.78	1.80	1.79	1.78	1.78
3176	200 krad	1.78	1.78	1.78	1.78	1.79	1.79	1.78	1.78
3188	200 krad	1.79	1.79	1.78	1.78	1.78	1.79	1.78	1.78

Table 5c Pre-Irradiation and Post-Annealing Differential Input Thresholds

In/Out Pin:		DIO_I5P/DIO_O5				DIO_I6P/DIO_O6			
DUT	Total Dose	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
		V_{IL} (V)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
3169	300 krad	1.78	1.78	1.78	1.78	1.79	1.79	1.78	1.78
3170	300 krad	1.79	1.79	1.78	1.78	1.79	1.79	1.77	1.77
3175	300 krad	1.79	1.79	1.78	1.78	1.79	1.79	1.78	1.79
3176	200 krad	1.79	1.79	1.78	1.78	1.79	1.78	1.77	1.78
3188	200 krad	1.79	1.79	1.78	1.78	1.79	1.79	1.78	1.78

Table 5d Pre-Irradiation and Post-Annealing Differential Input Thresholds

In/Out Pin:		DIO_I7P/DIO_O7			
DUT	Total Dose	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
		V_{IL} (V)		V_{IH} (V)	
3169	300 krad	1.80	1.80	1.79	1.79
3170	300 krad	1.80	1.80	1.78	1.79
3175	300 krad	1.80	1.80	1.79	1.79
3176	200 krad	1.80	1.80	1.79	1.79
3188	200 krad	1.81	1.81	1.79	1.80

E. Output-Drive Voltage (V_{OL}/V_{OH})

The pre-irradiation and post-annealing V_{OL}/V_{OH} are listed in Tables 6 and 7. The post-annealing data are within the spec limits; in each case, the radiation-induced degradation is within 10%.

Table 6 Pre-Irradiation and Post-Annealing V_{OL} (V) at Various Sinking Current

DUT	Total Dose	1 mA		12 mA		20 mA		50 mA		100 mA	
		Pre-rad	Pos-an	Pre-rad	Pos-an	Pre-rad	Pos-an	Pre-rad	Pos-an	Pre-rad	Pos-an
3169	300 krad	0.007	0.007	0.088	0.086	0.147	0.144	0.373	0.365	0.783	0.767
3170	300 krad	0.007	0.008	0.088	0.086	0.148	0.144	0.375	0.365	0.787	0.768
3175	300 krad	0.007	0.007	0.088	0.086	0.148	0.144	0.374	0.364	0.786	0.766
3176	200 krad	0.007	0.007	0.088	0.085	0.146	0.143	0.372	0.363	0.780	0.763
3188	200 krad	0.007	0.007	0.088	0.087	0.148	0.145	0.376	0.369	0.790	0.776

Table 7 Pre-Irradiation and Post-Annealing V_{OH} (V) at Various Sourcing Current

DUT	Total Dose	1 mA		8 mA		20 mA		50 mA		100 mA	
		Pre-rad	Pos-an	Pre-rad	Pos-an	Pre-rad	Pos-an	Pre-rad	Pos-an	Pre-rad	Pos-an
3169	300 krad	3.29	3.28	3.22	3.22	3.10	3.10	2.79	2.79	2.19	2.18
3170	300 krad	3.29	3.29	3.22	3.22	3.10	3.10	2.79	2.79	2.18	2.18
3175	300 krad	3.29	3.29	3.22	3.22	3.10	3.10	2.79	2.79	2.18	2.18
3176	200 krad	3.29	3.29	3.22	3.22	3.10	3.10	2.79	2.79	2.18	2.18
3188	200 krad	3.29	3.29	3.22	3.22	3.10	3.10	2.79	2.79	2.18	2.17

F. Propagation Delay

Table 8 lists the post-annealing propagation delays. It shows the change due to irradiation in each case is not significant.

Table 8 Radiation-Induced Propagation Delay Degradations

DUT	Total Dose	Pre-Irradiation (µs)	Post-Annealing (µs)	Degradation
3169	300 krad	6.853	6.804	-0.72%
3170	300 krad	7.219	7.174	-0.62%
3175	300 krad	7.272	7.277	0.06%
3176	200 krad	7.308	7.247	-0.84%
3188	200 krad	6.934	6.912	-0.32%

G. *Transition Time*

Figures 7 to 16 show the pre-irradiation and post-annealing transition edges. In each case, the radiation-induced transition-time degradation is not observable.

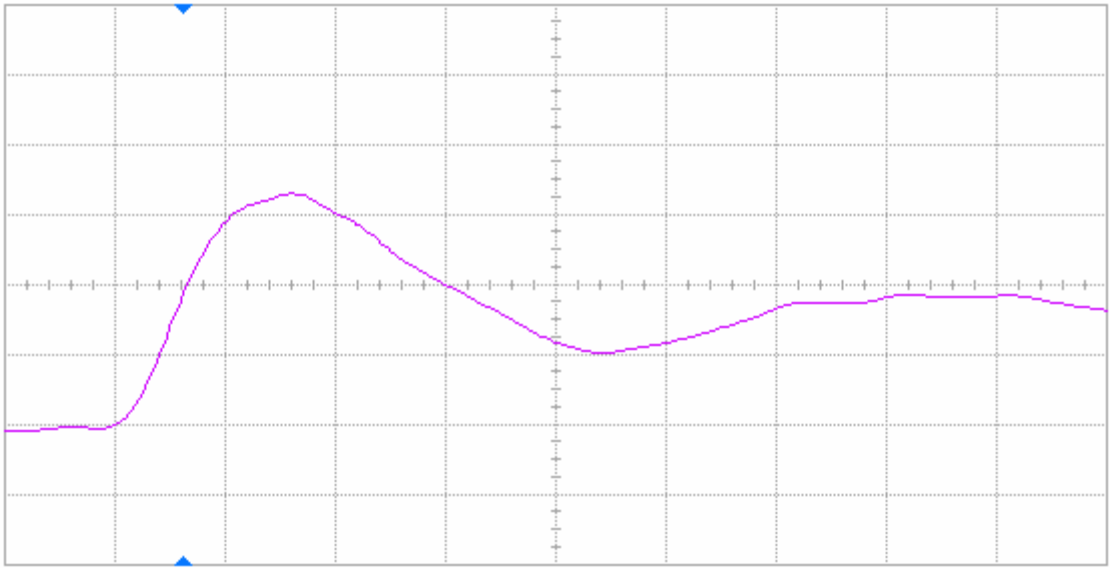


Figure 7(a) DUT 3169 pre-irradiation rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

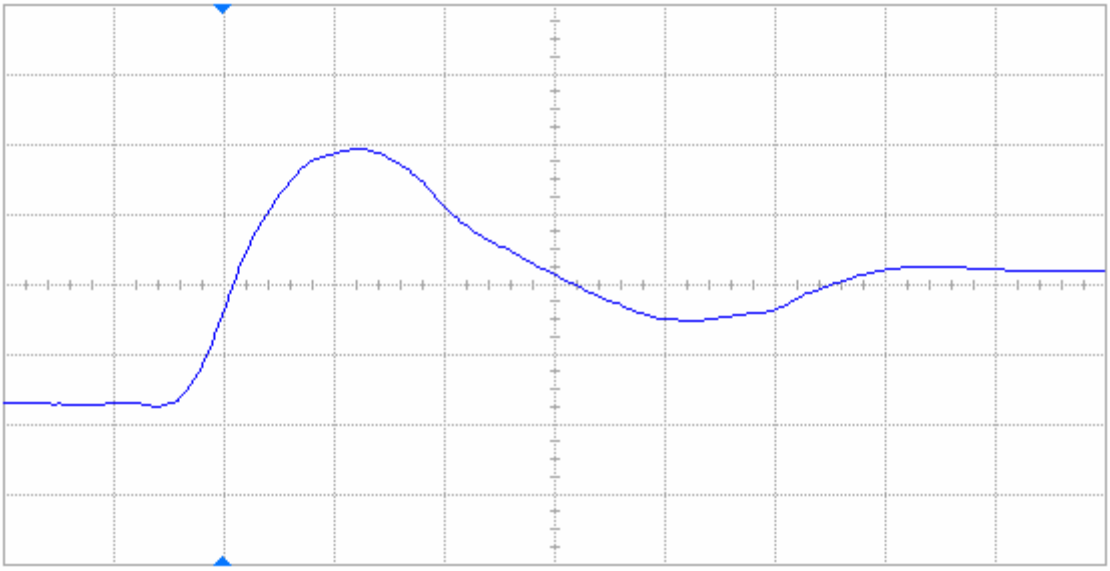


Figure 7(b) DUT 3169 post-annealing rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

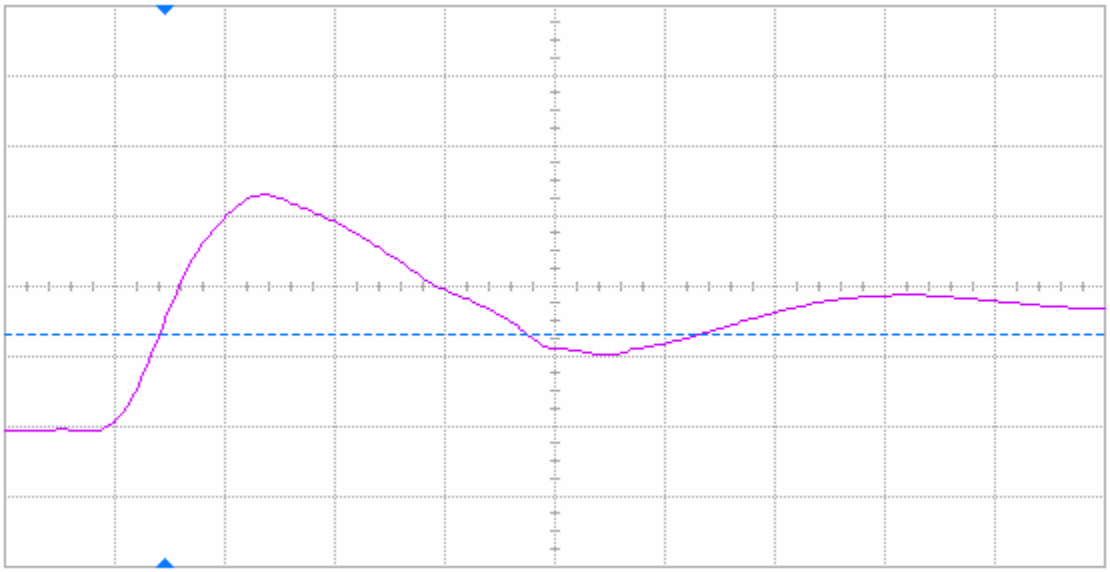


Figure 8(a) DUT 3170 pre-irradiation rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

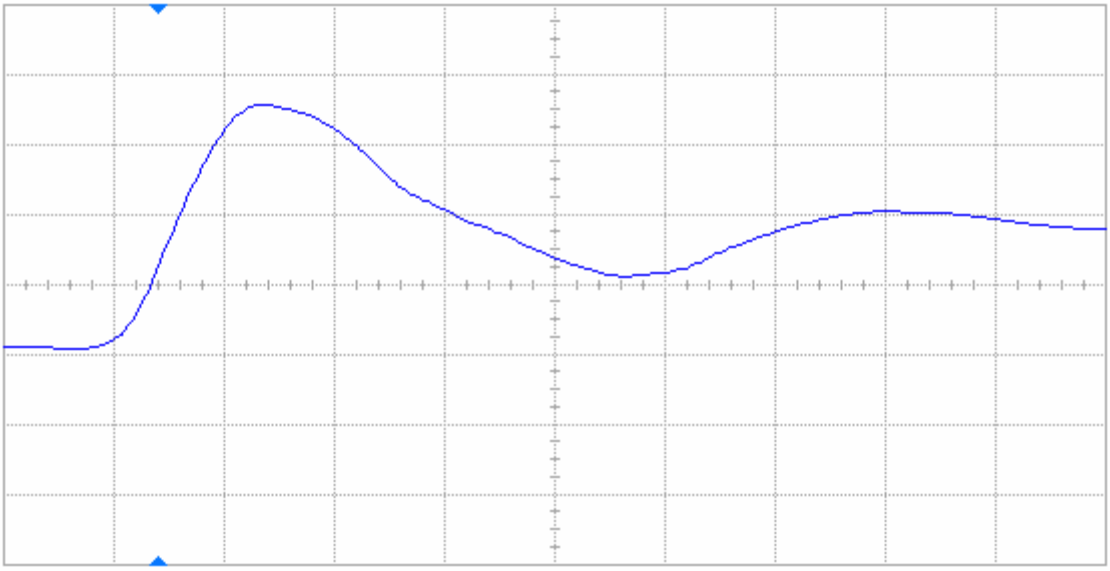


Figure 8(b) DUT 3170 post-annealing rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

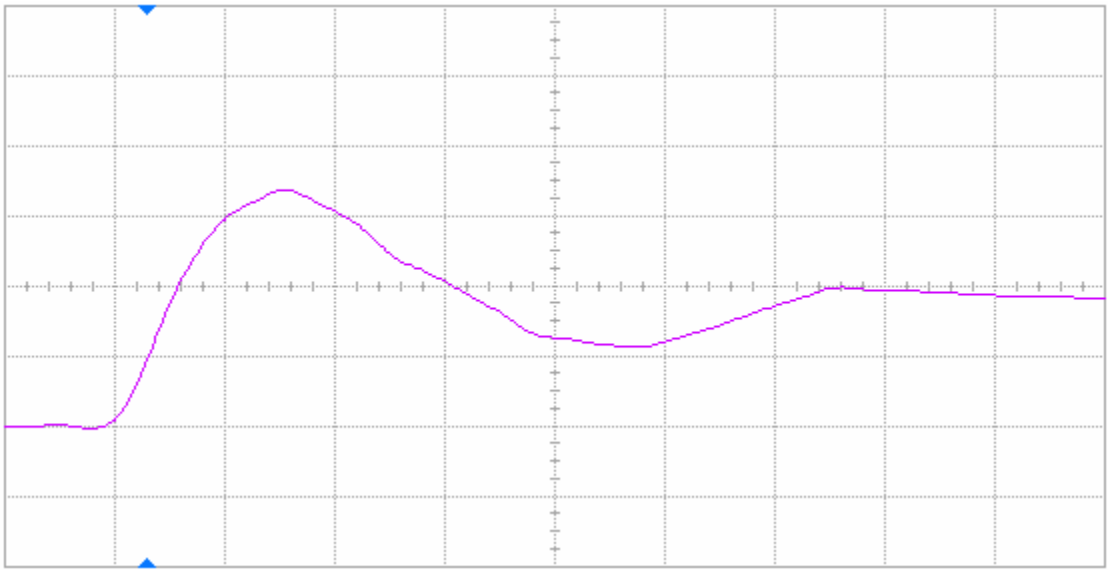


Figure 9(a) DUT 3175 pre-radiation rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

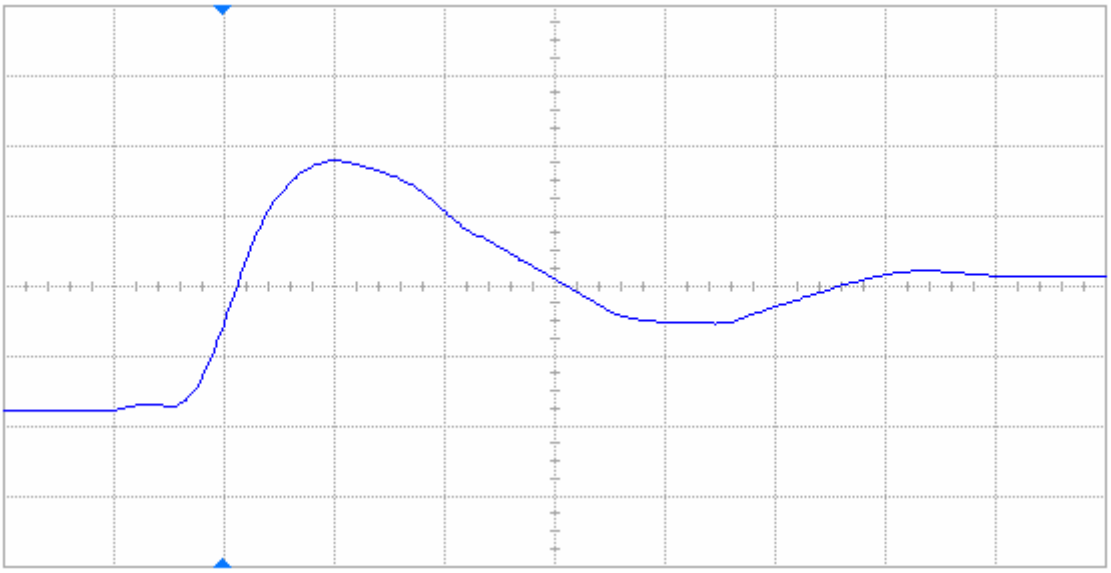


Figure 9(b) DUT 3175 post-annealing rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

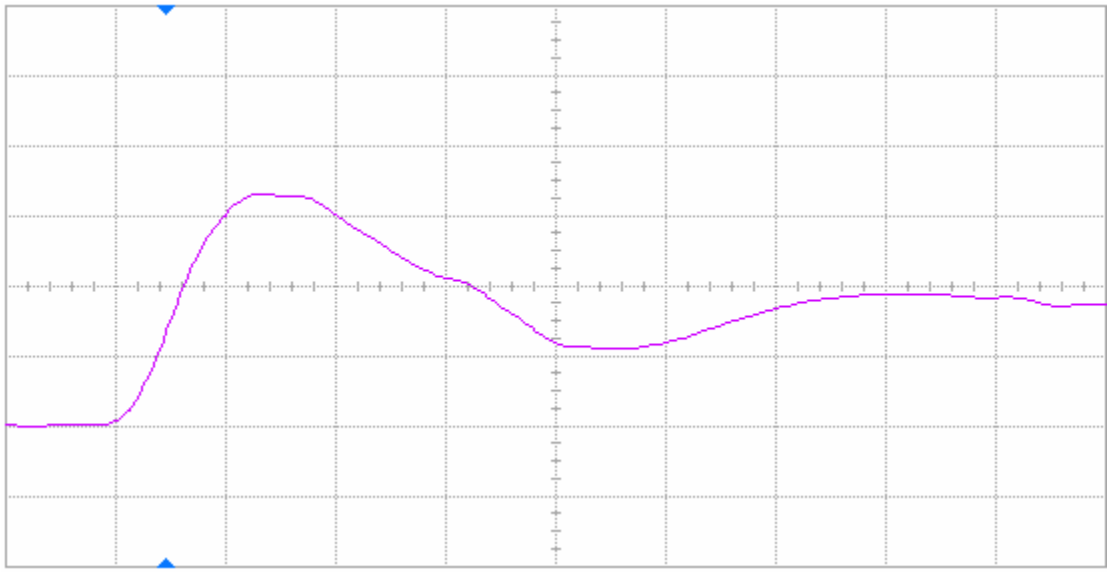


Figure 10(a) DUT 3176 pre-irradiation rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

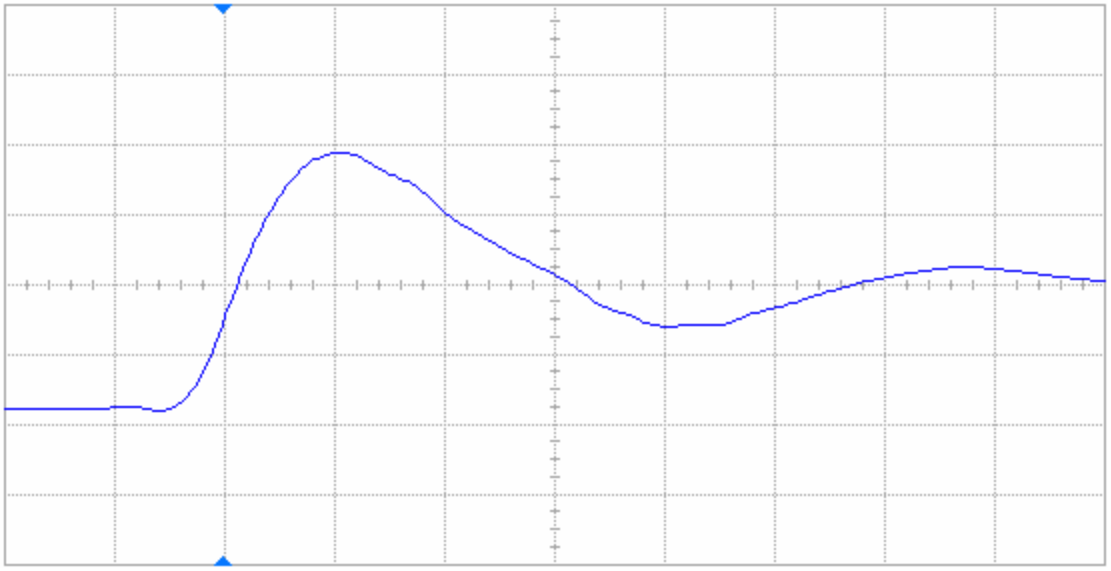


Figure 10(b) DUT 3176 post-annealing rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

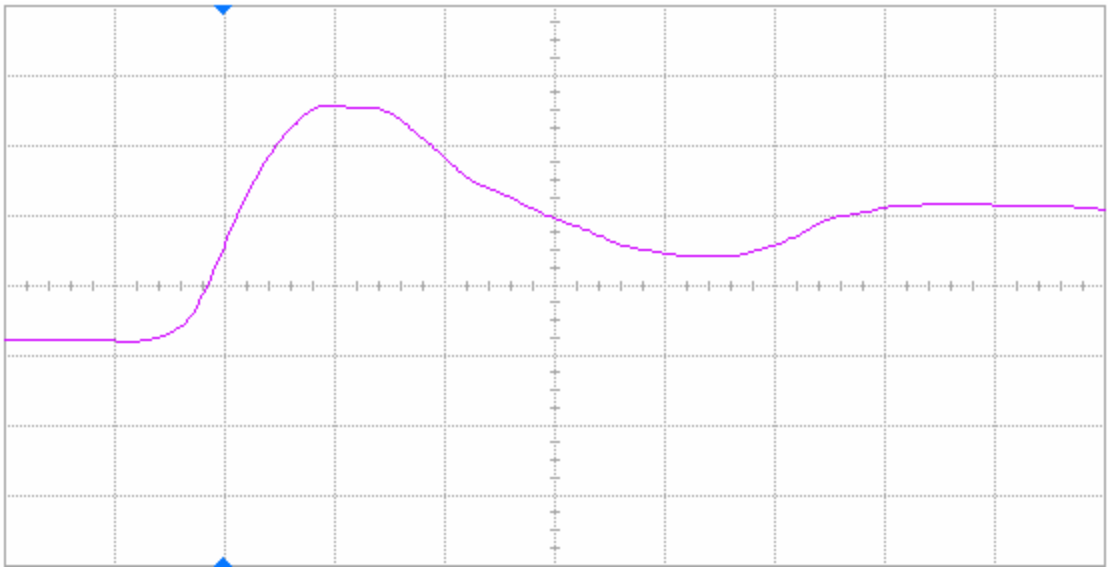


Figure 11(a) DUT 3188 pre-irradiation rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

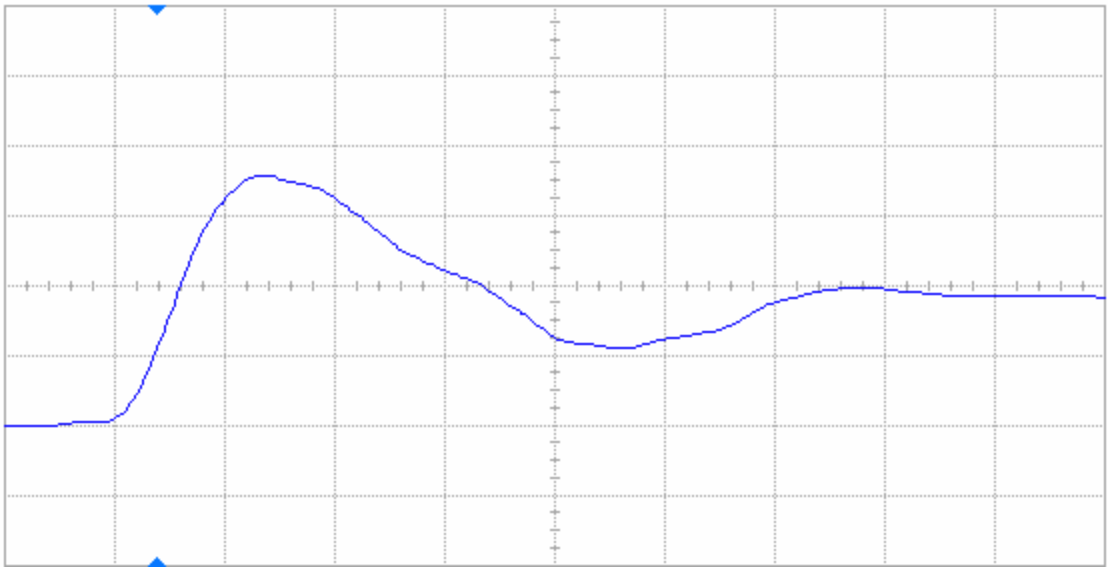


Figure 11(b) DUT 3188 post-annealing rising edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

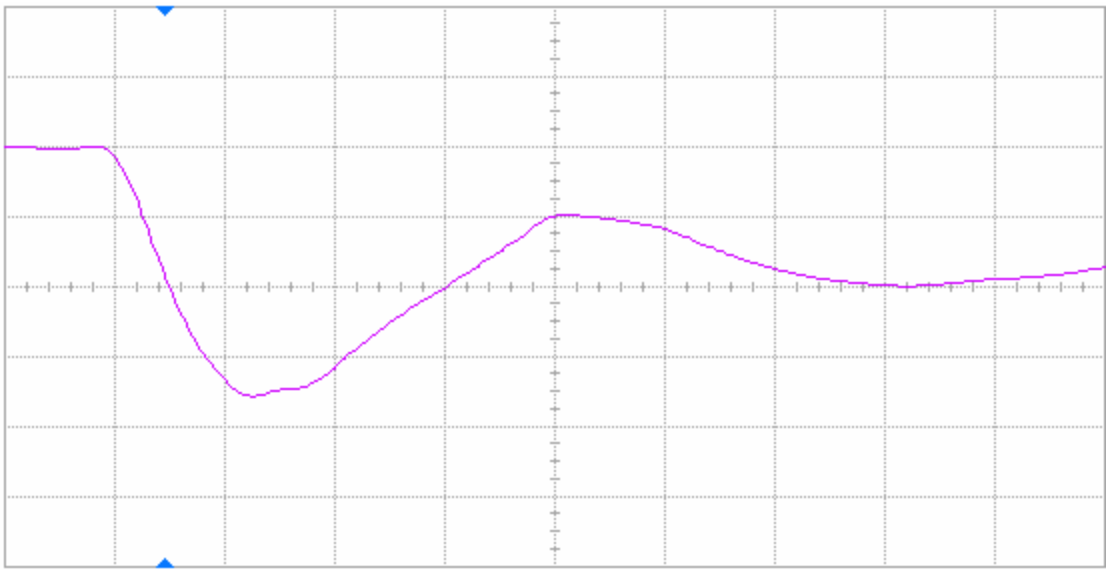


Figure 12(a) DUT 3169 pre-irradiation falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

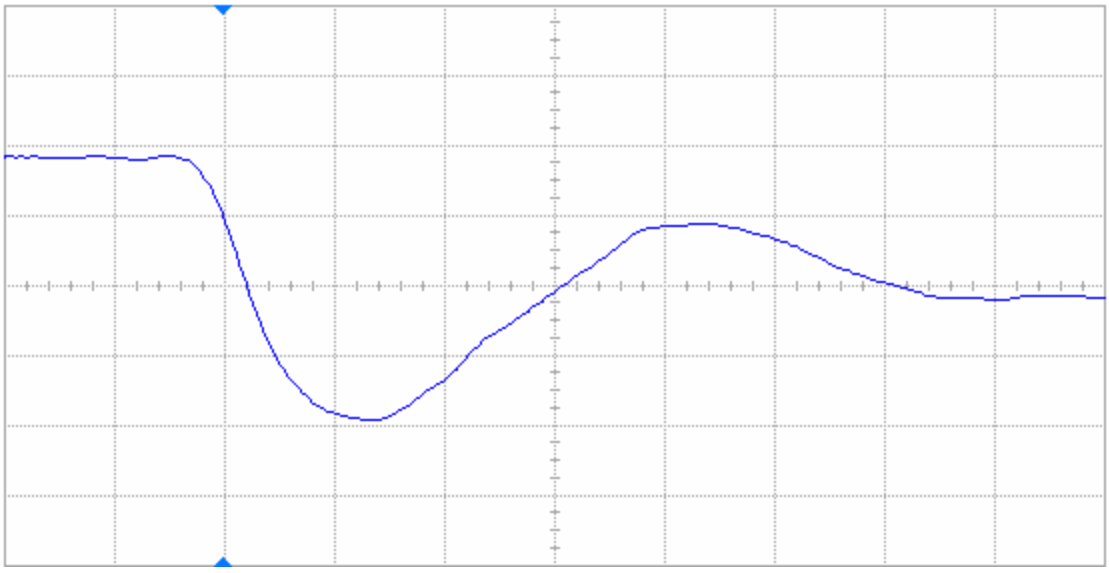


Figure 12(b) DUT 3169 post-annealing falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

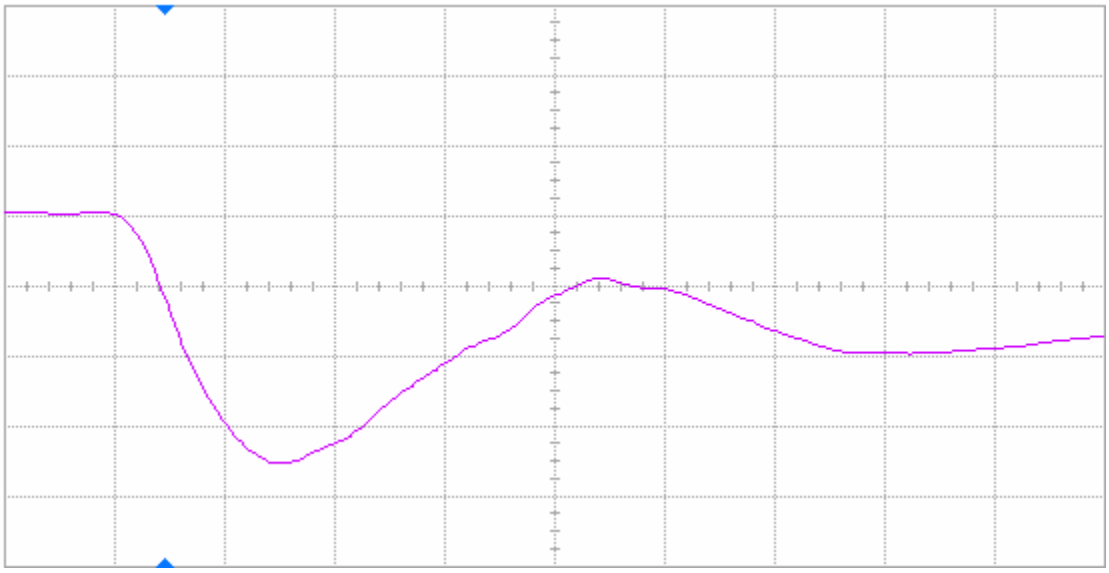


Figure 13(a) DUT 3170 pre-irradiation falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

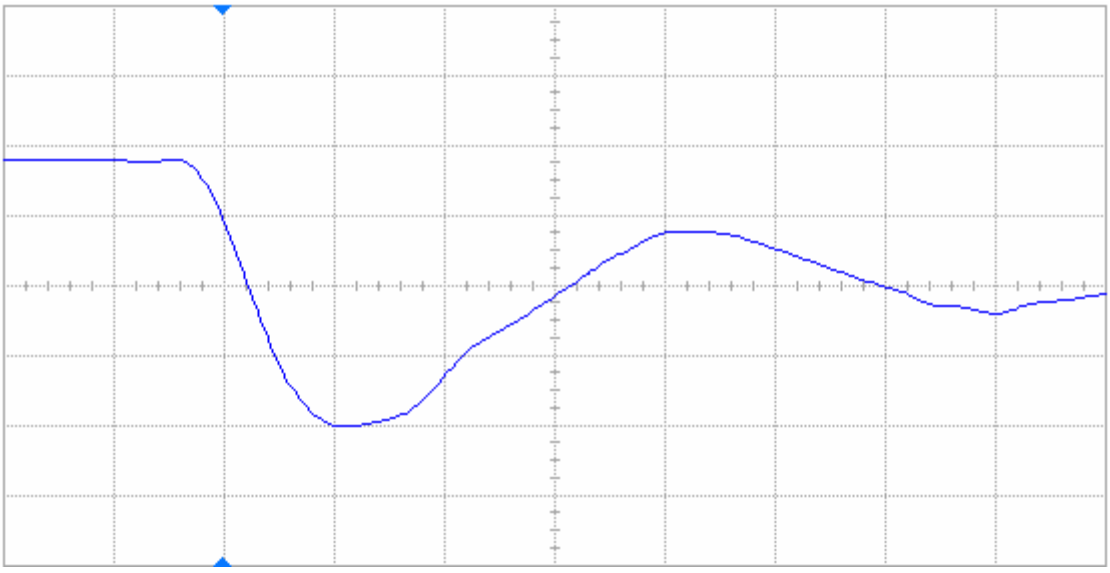


Figure 13(b) DUT 3170 post-annealing falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

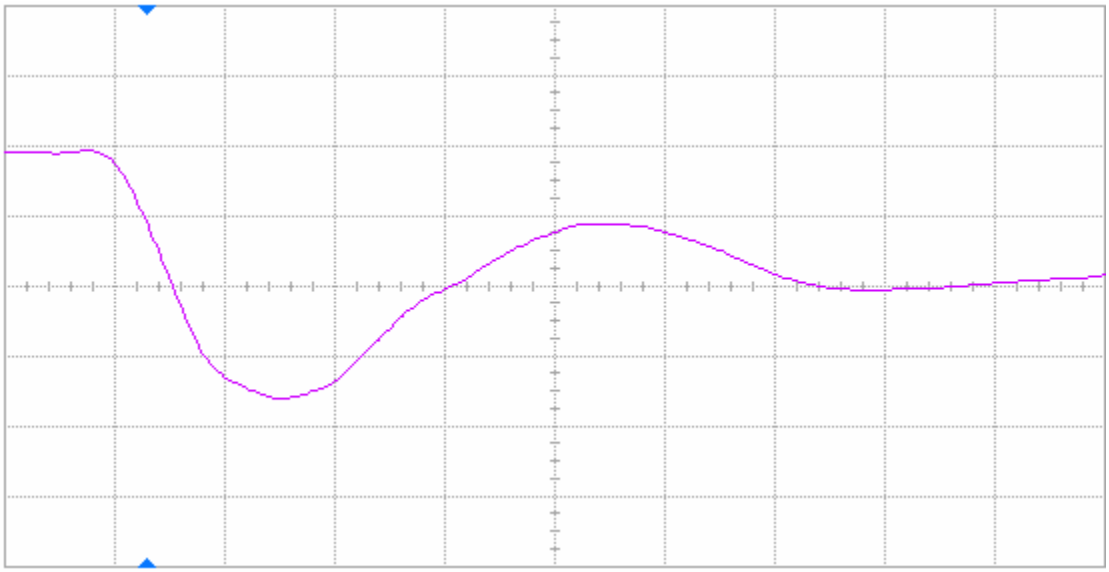


Figure 14(a) DUT 3175 pre-irradiation falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

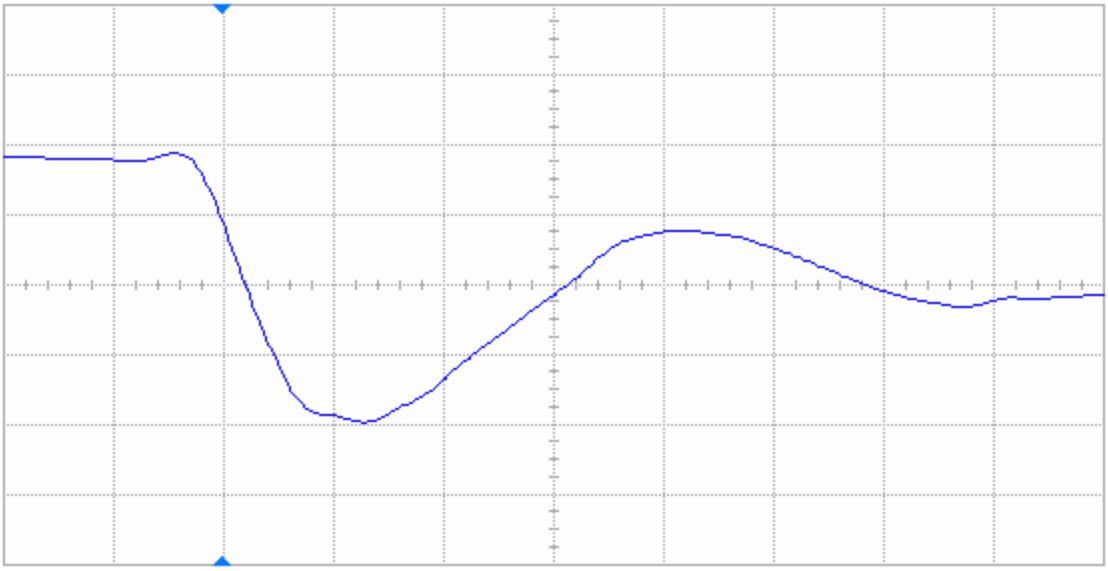


Figure 14(b) DUT 3175 post-annealing falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

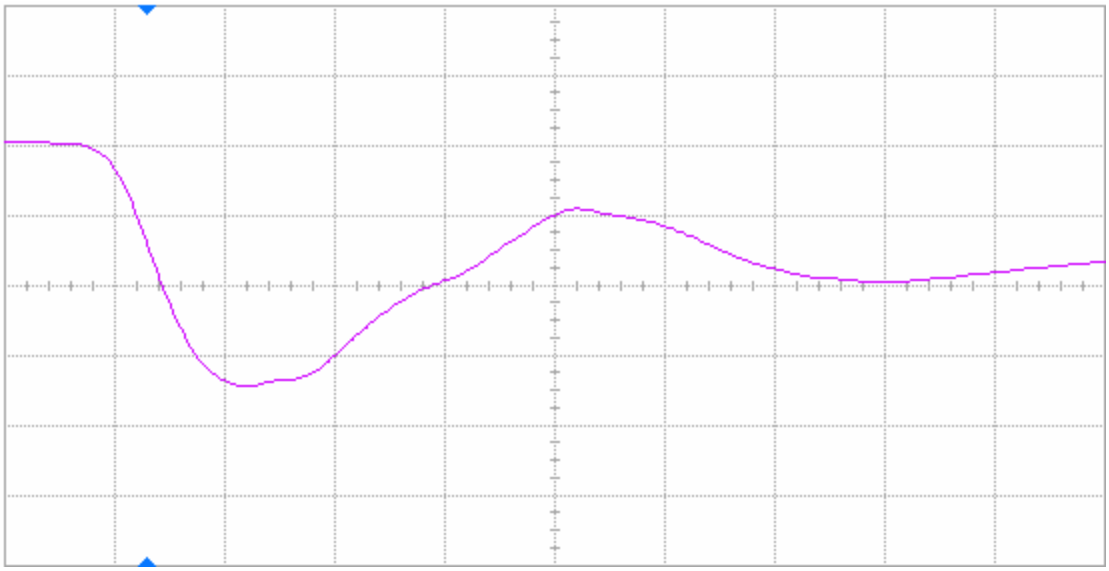


Figure 15(a) DUT 3176 pre-irradiation falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

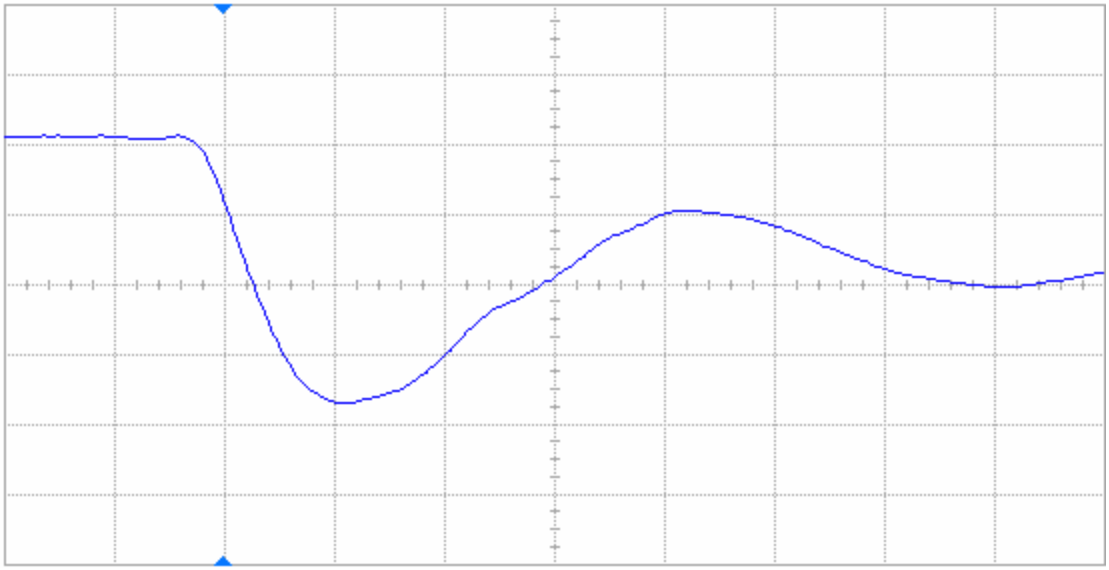


Figure 15(b) DUT 3176 post-annealing falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

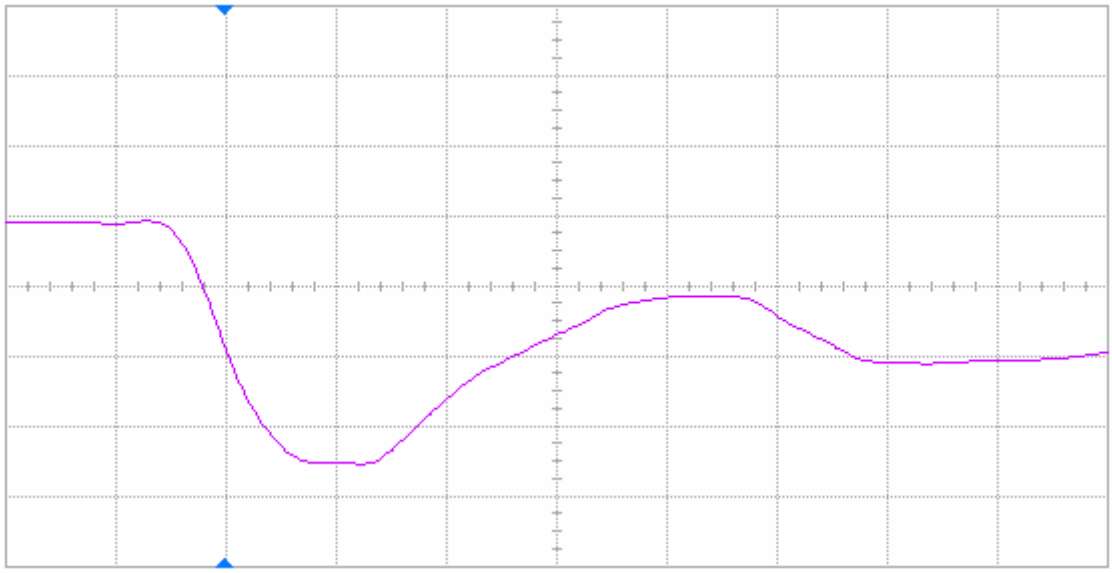


Figure 16(a) DUT 3188 pre-irradiation falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

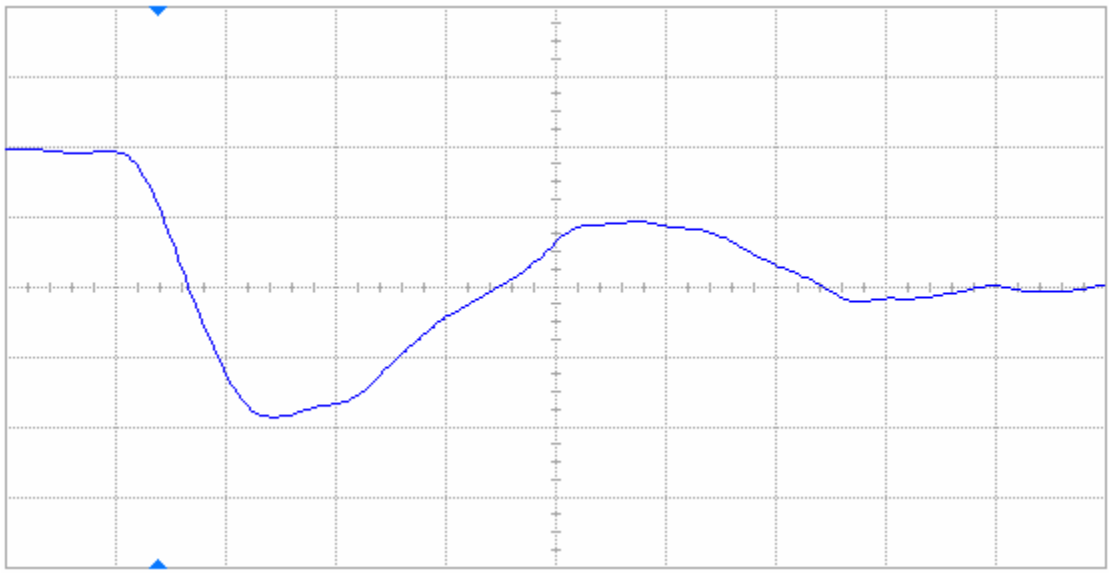


Figure 16(b) DUT 3188 post-annealing falling edge, abscissa scale is 2 V/div and ordinate scale is 2 ns/div.

APPENDIX A DUT BIAS

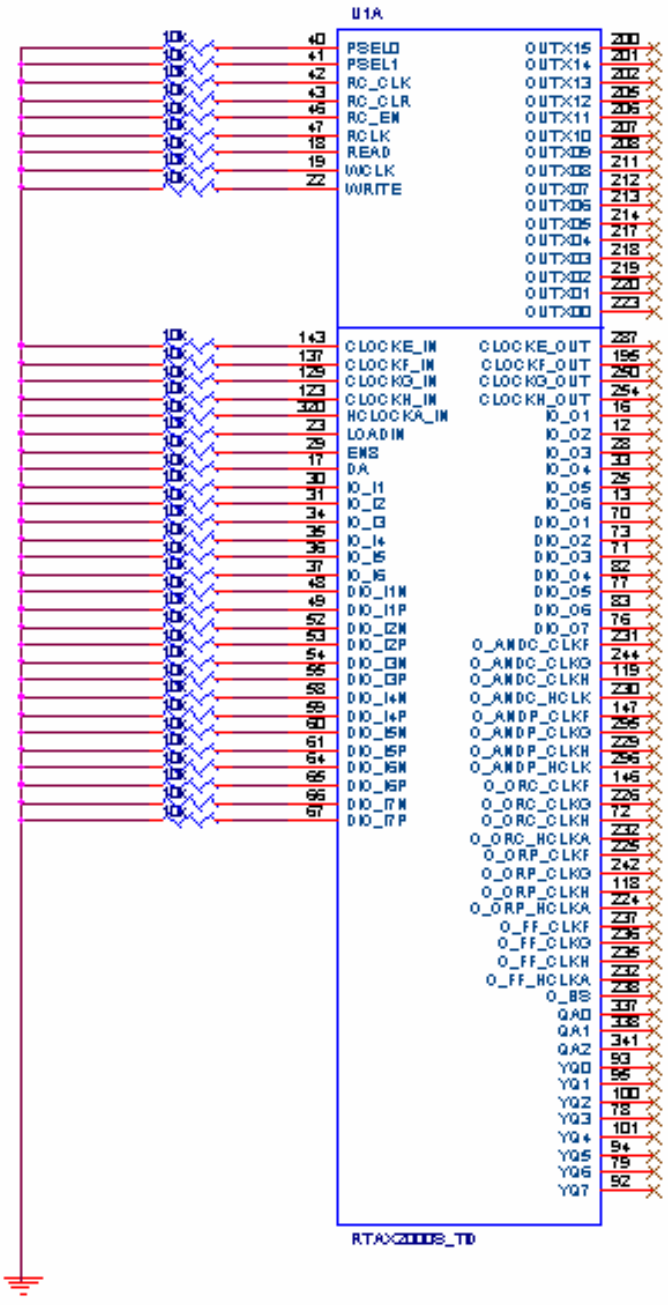


Figure A1 IO bias during irradiation

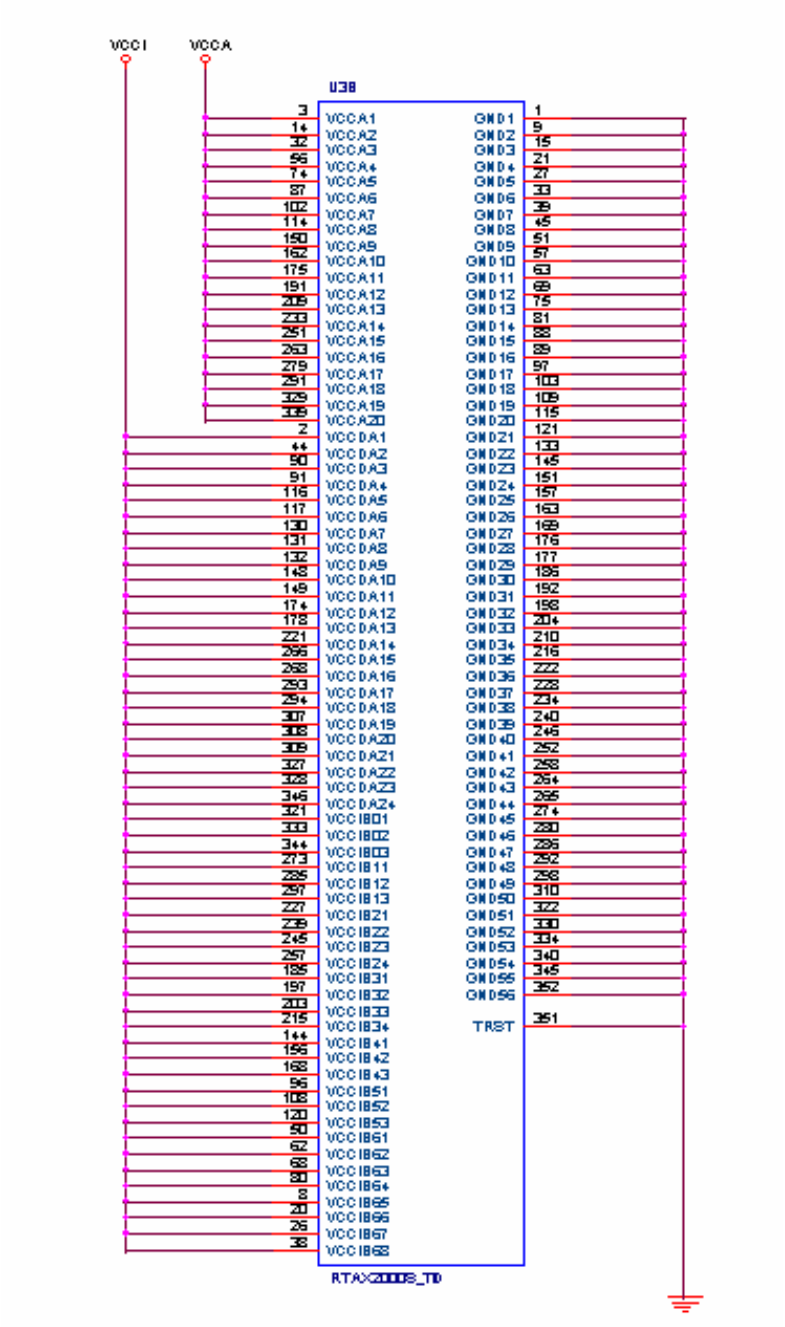
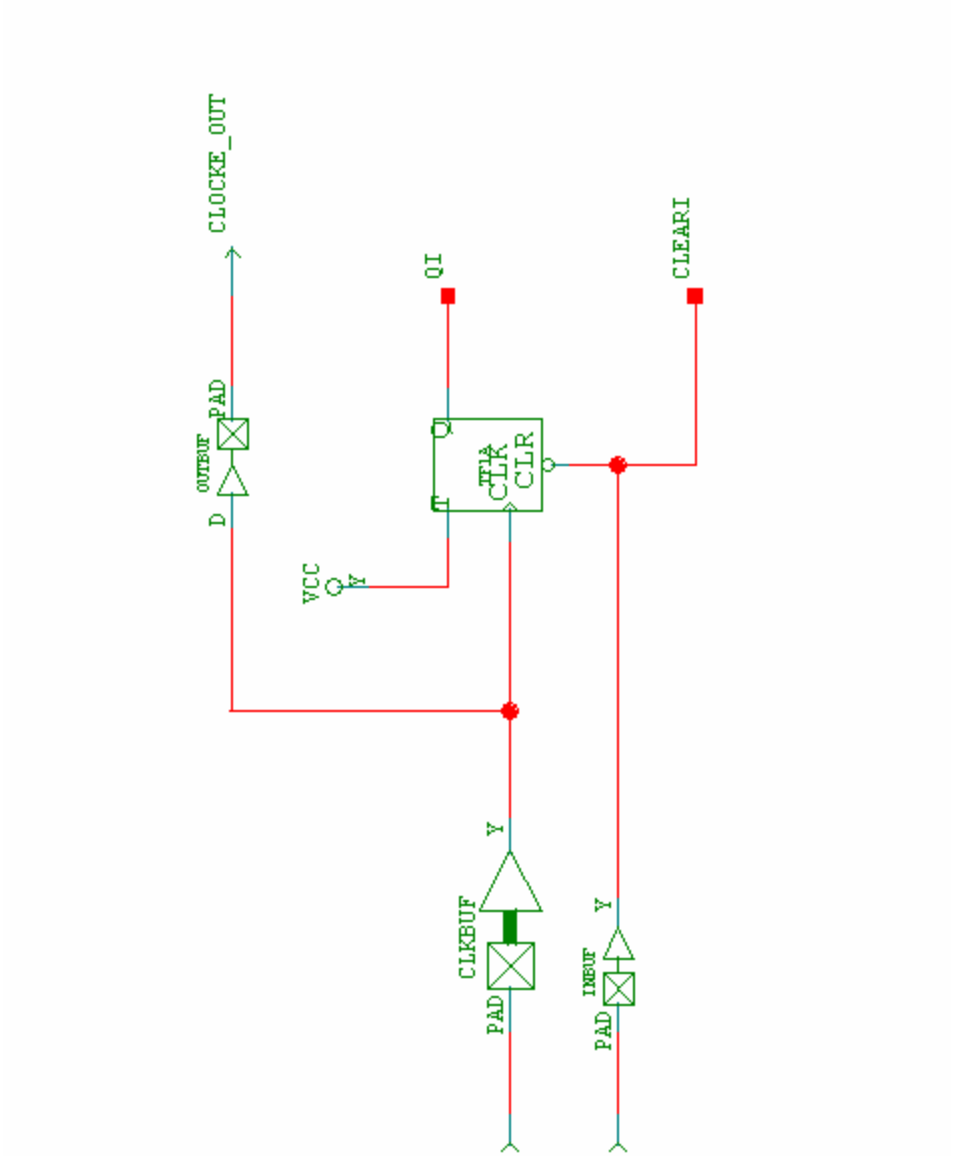
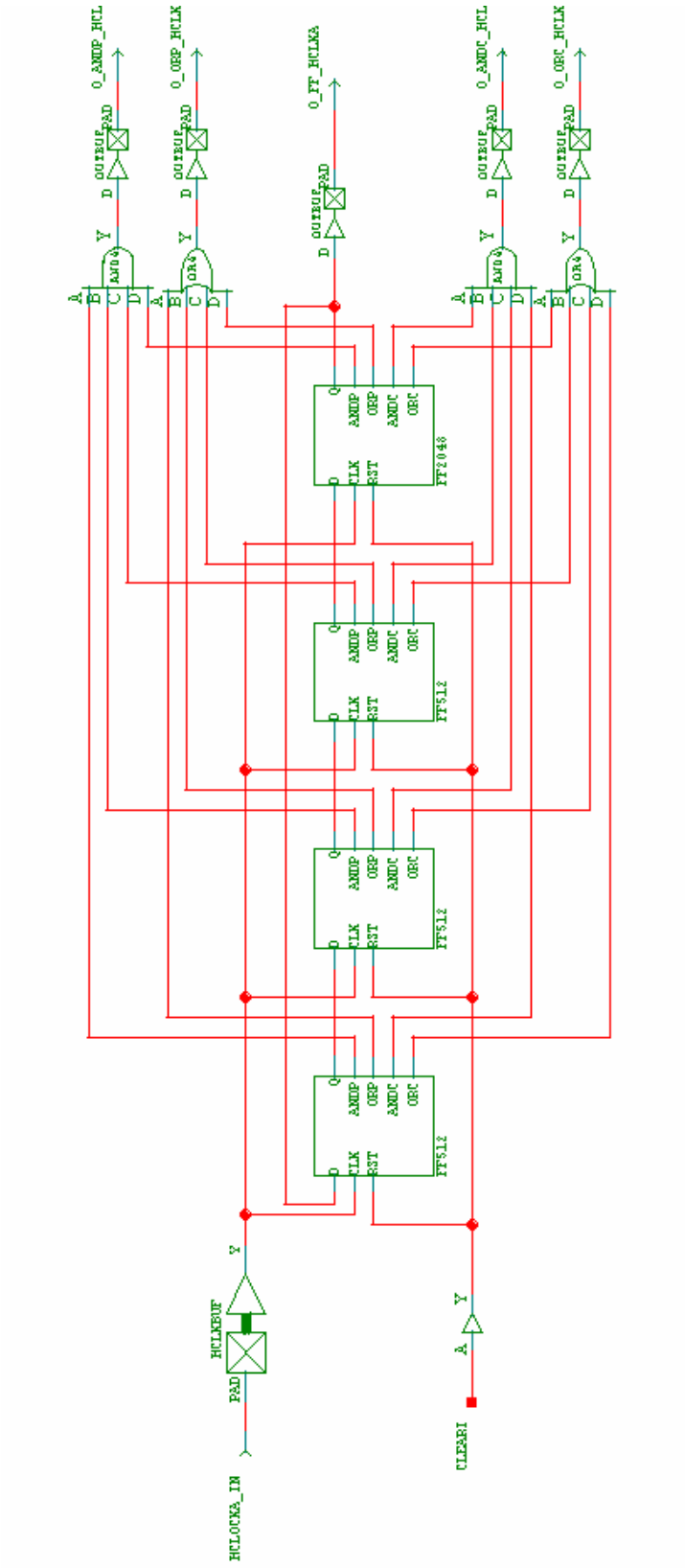


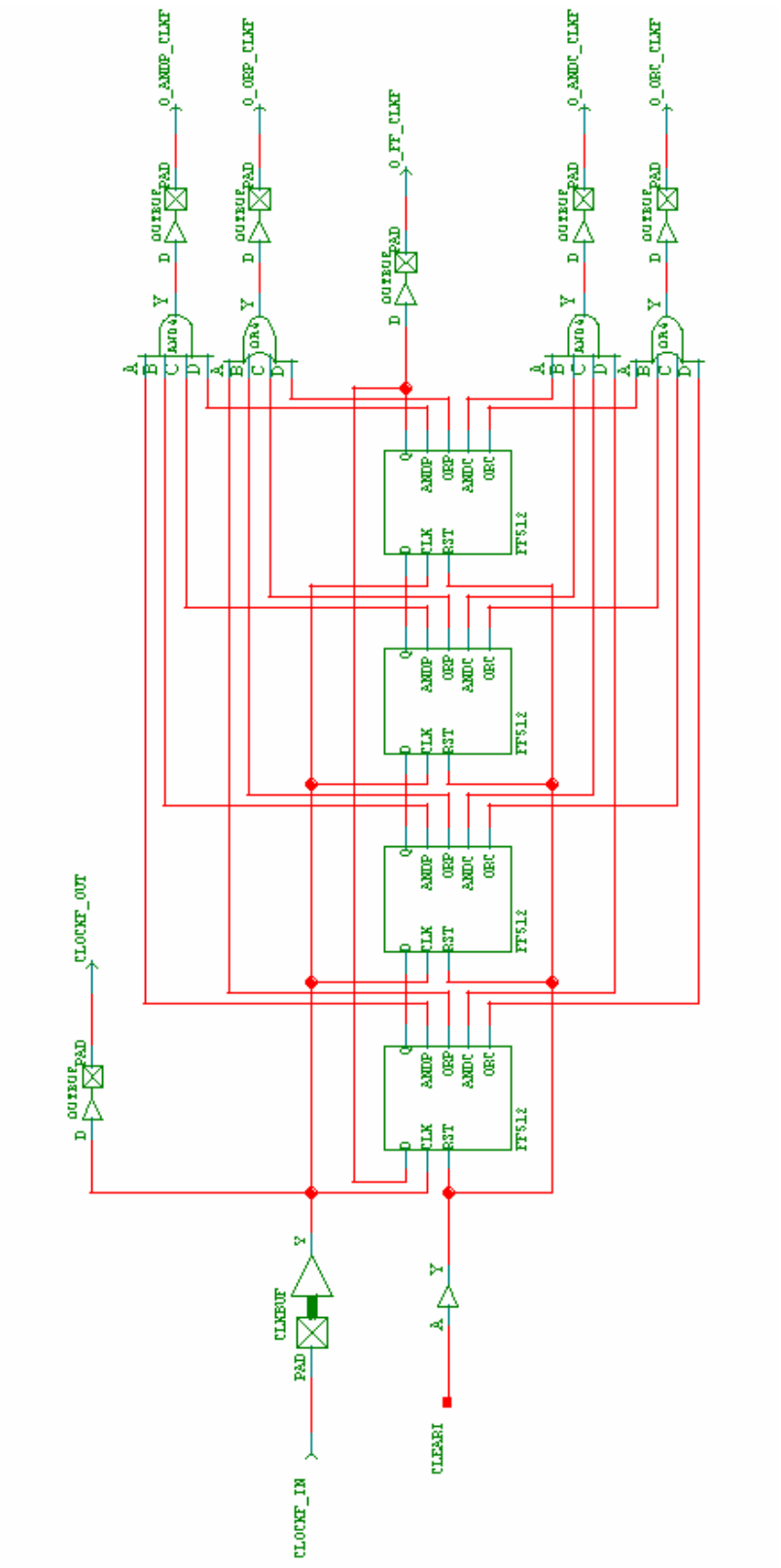
Figure A2 Power supply, ground and special pins bias during irradiation

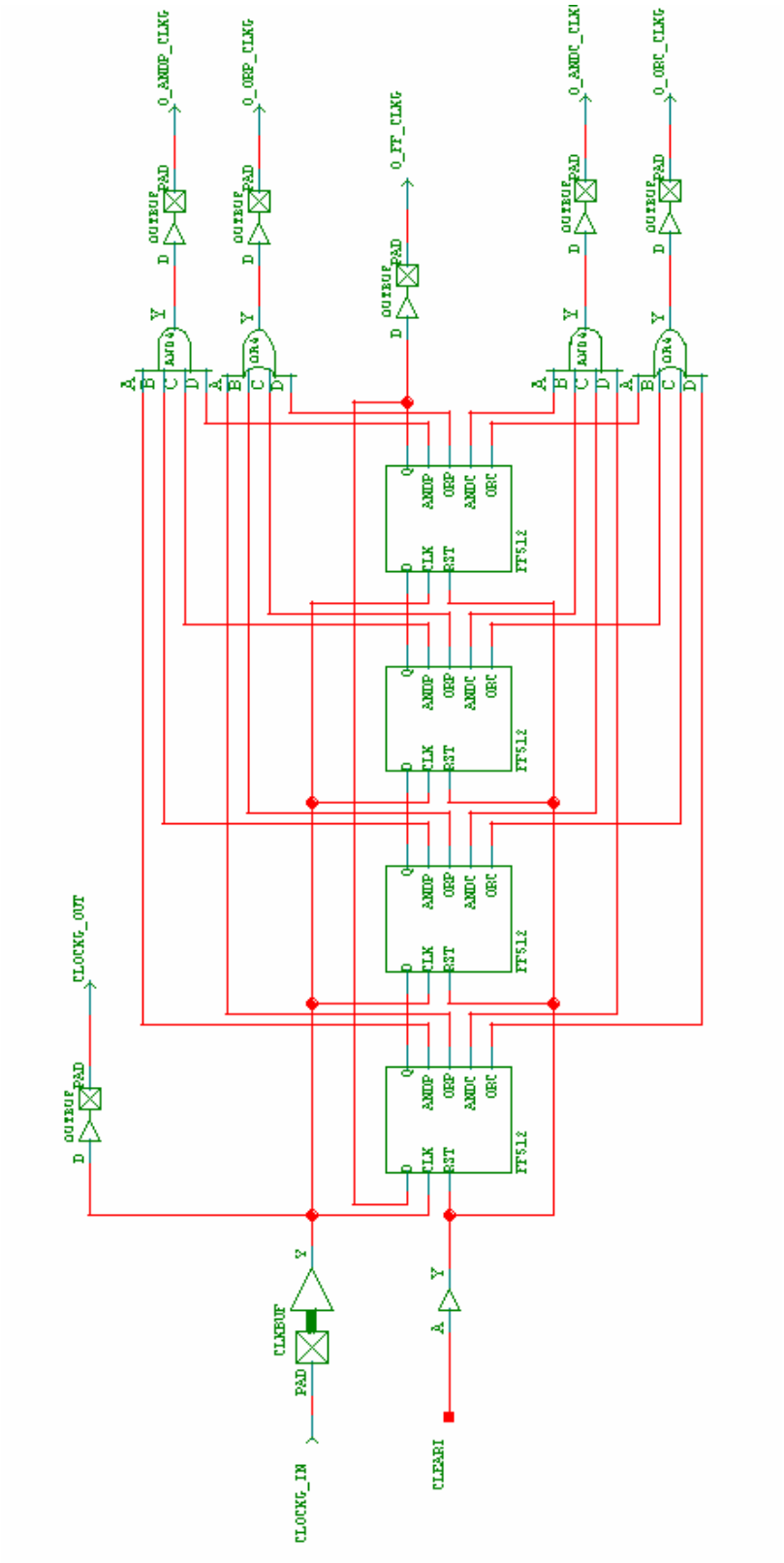
APPENDIX B DUT DESIGN SCHEMATICS AND VERILOG FILES

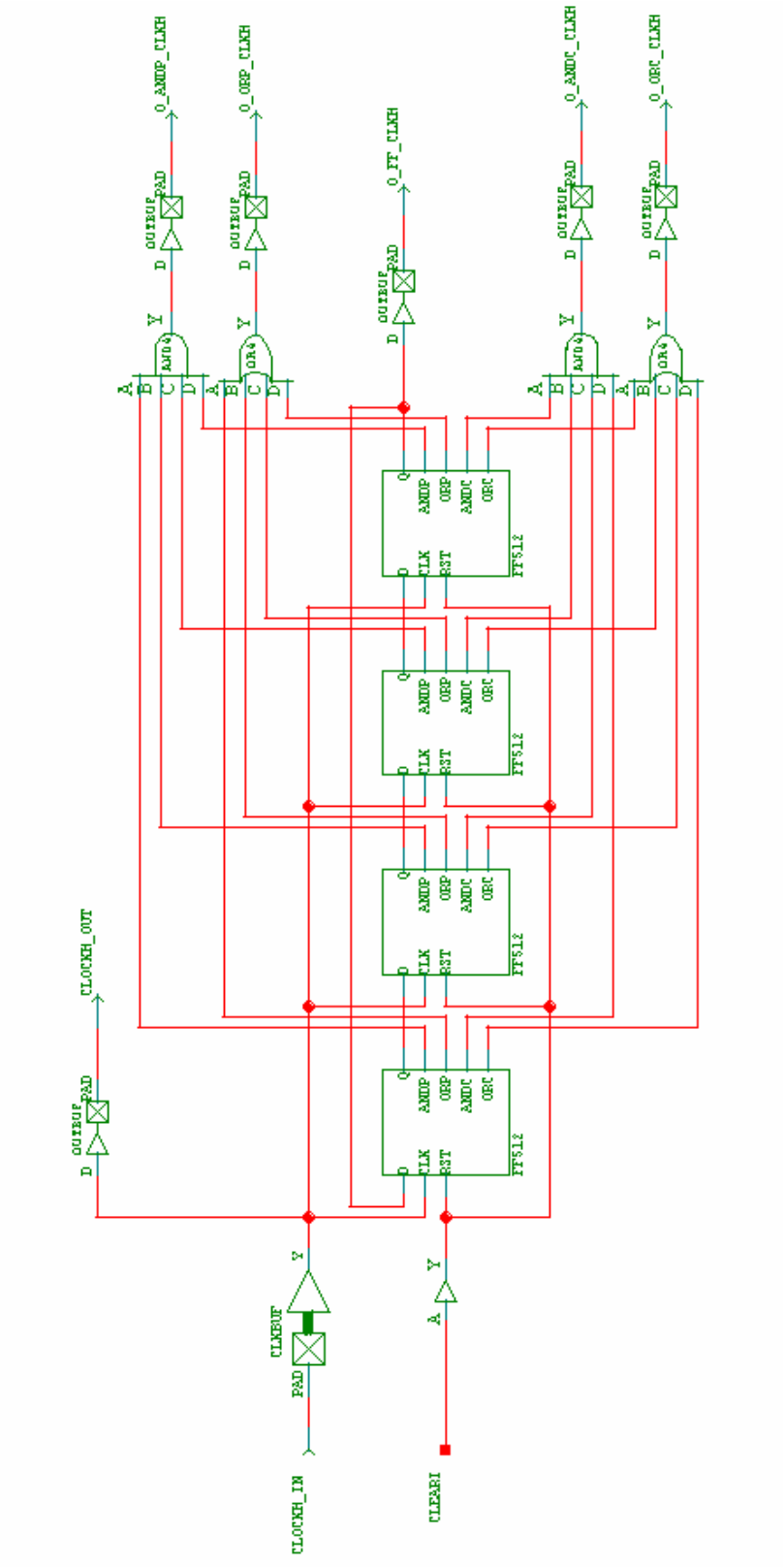


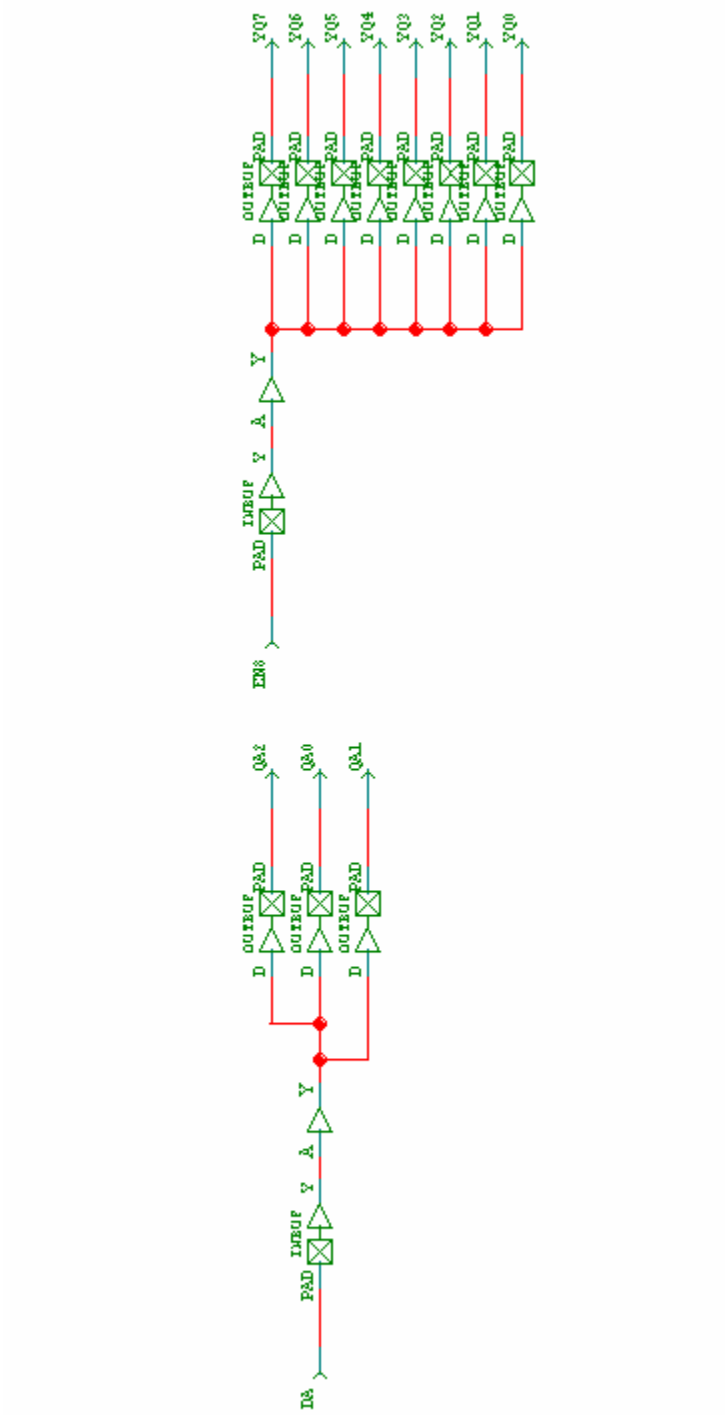

O_AMDP_HCL
O_ORP_HCLK
O_FF_HCLKA
O_AMDC_HCL
O_ORC_HCLK
OUTBUF
PAD
D
Y
AN04
OR4
A
B
C
D
Q
AMDP
ORP
AMDC
ORC
CLK
RST
FF2048
FF512
HCLKBUF
HCLOCKA_IN
CLEARI

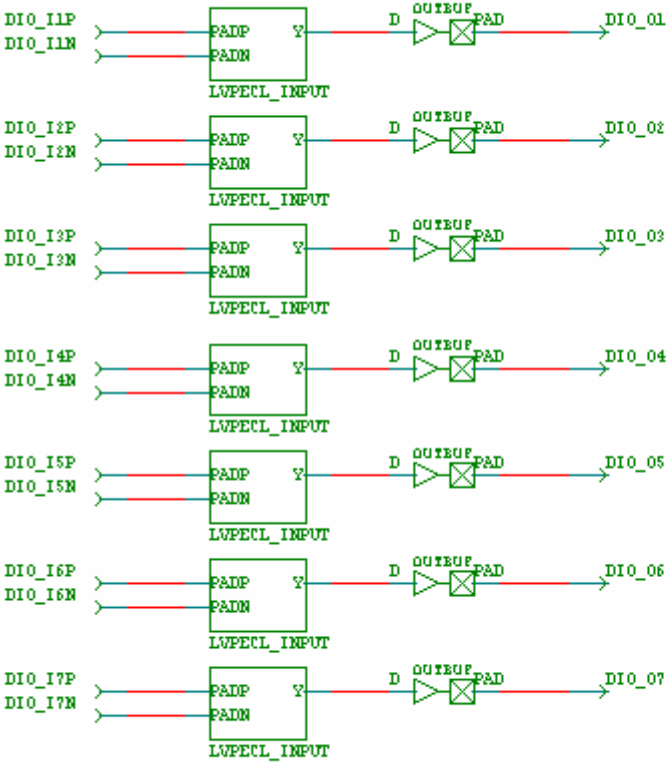

O_B3
OUTBUF_PAD
I
Out
In
BUFT1k
BUFT1k
BUFT1k
BUFT1k
BUFT10k
QI

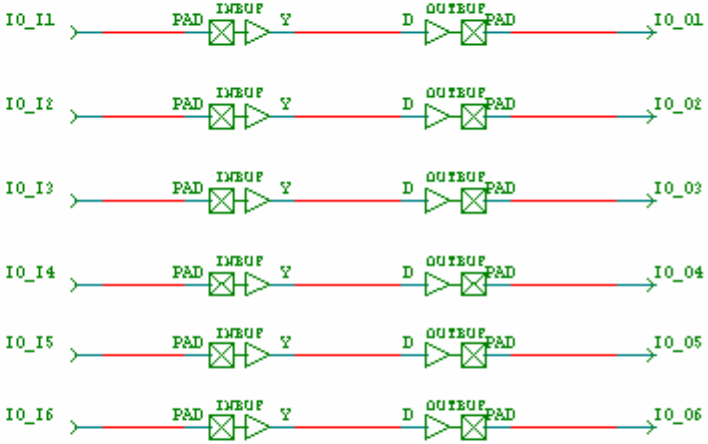

O_AMDP_CLKF
O_ORP_CLKF
O_FT_CLKF
O_AMDC_CLKF
O_ORC_CLKF
OUTBUF
PAD
AND4
OR4
FF512
CLOCKF_OUT
CLKBUF
CLOCKF_IN
CLEARI


O_AMDP_CLKG
O_ORP_CLKG
O_FT_CLKG
O_AMDC_CLKG
O_ORC_CLKG
OUTBUF
PAD
D
Y
AND4
OR4
A
B
C
D
Q
AMDP
ORP
AMDC
ORC
CLK
RST
FF512
CLOCKG_OUT
CLKBUF
CLOCKG_IN
CLEARI


O_AMDP_CLKH
O_ORP_CLKH
O_FF_CLKH
O_AMDC_CLKH
O_ORC_CLKH
OUTBUF
PAD
AND4
OR4
FF512
AMDP
ORP
AMDC
ORC
CLK
RST
CLOCKH_OUT
CLKBUF
CLOCKH_IN
CLEAR1


YQ7
YQ6
YQ5
YQ4
YQ3
YQ2
YQ1
YQ0
OUTBUF
PAD
D
Y
A
INBUF
EN6
Q42
Q40
Q41
Da


DIO_I1P
DIO_I1N
PADP
PADN
Y
LVPECL_INPUT
D
OUTBUF
PAD
DIO_01
DIO_I2P
DIO_I2N
DIO_02
DIO_I3P
DIO_I3N
DIO_03
DIO_I4P
DIO_I4N
DIO_04
DIO_I5P
DIO_I5N
DIO_05
DIO_I6P
DIO_I6N
DIO_06
DIO_I7P
DIO_I7N
DIO_07


IO_I1
PAD
IWBUF
Y
D
OUTBUF
IO_01
IO_I2
IO_02
IO_I3
IO_03
IO_I4
IO_04
IO_I5
IO_05
IO_I6
IO_06

```
//BUFF10k
`timescale 1 ns/100 ps

module BUFF10k (In, Out);

input In;

output Out;

wire x1/*synthesis syn_keep=1 alspreserve=1*/;
wire x2/*synthesis syn_keep=1 alspreserve=1*/;
wire x3/*synthesis syn_keep=1 alspreserve=1*/;
wire x4/*synthesis syn_keep=1 alspreserve=1*/;
wire x5/*synthesis syn_keep=1 alspreserve=1*/;
wire x6/*synthesis syn_keep=1 alspreserve=1*/;
wire x7/*synthesis syn_keep=1 alspreserve=1*/;
wire x8/*synthesis syn_keep=1 alspreserve=1*/;
wire x9/*synthesis syn_keep=1 alspreserve=1*/;

BUFF1k buff1 (.In(In), .Out(x1));
BUFF1k buff2 (.In(x1), .Out(x2));
BUFF1k buff3 (.In(x2), .Out(x3));
BUFF1k buff4 (.In(x3), .Out(x4));
BUFF1k buff5 (.In(x4), .Out(x5));
BUFF1k buff6 (.In(x5), .Out(x6));
BUFF1k buff7 (.In(x6), .Out(x7));
BUFF1k buff8 (.In(x7), .Out(x8));
BUFF1k buff9 (.In(x8), .Out(x9));
BUFF1k buff10 (.In(x9), .Out(Out));

Endmodule
```

```
//BUFF1k
`timescale 1 ns/100 ps

module BUFF1k (In, Out);

input In;

output Out;

wire x1/*synthesis syn_keep=1 alspreserve=1*/;
wire x2/*synthesis syn_keep=1 alspreserve=1*/;
wire x3/*synthesis syn_keep=1 alspreserve=1*/;
wire x4/*synthesis syn_keep=1 alspreserve=1*/;
wire x5/*synthesis syn_keep=1 alspreserve=1*/;
wire x6/*synthesis syn_keep=1 alspreserve=1*/;
wire x7/*synthesis syn_keep=1 alspreserve=1*/;
wire x8/*synthesis syn_keep=1 alspreserve=1*/;
wire x9/*synthesis syn_keep=1 alspreserve=1*/;
wire x10/*synthesis syn_keep=1 alspreserve=1*/;
wire x11/*synthesis syn_keep=1 alspreserve=1*/;
wire x12/*synthesis syn_keep=1 alspreserve=1*/;
wire x13/*synthesis syn_keep=1 alspreserve=1*/;
wire x14/*synthesis syn_keep=1 alspreserve=1*/;
wire x15/*synthesis syn_keep=1 alspreserve=1*/;
wire x16/*synthesis syn_keep=1 alspreserve=1*/;
wire x17/*synthesis syn_keep=1 alspreserve=1*/;
wire x18/*synthesis syn_keep=1 alspreserve=1*/;
wire x19/*synthesis syn_keep=1 alspreserve=1*/;

BUFF50 buff1 (.In(In), .Out(x1));
BUFF50 buff2 (.In(x1), .Out(x2));
BUFF50 buff3 (.In(x2), .Out(x3));
BUFF50 buff4 (.In(x3), .Out(x4));
BUFF50 buff5 (.In(x4), .Out(x5));
BUFF50 buff6 (.In(x5), .Out(x6));
BUFF50 buff7 (.In(x6), .Out(x7));
BUFF50 buff8 (.In(x7), .Out(x8));
BUFF50 buff9 (.In(x8), .Out(x9));
BUFF50 buff10 (.In(x9), .Out(x10));

BUFF50 buff11 (.In(x10), .Out(x11));
BUFF50 buff12 (.In(x11), .Out(x12));
BUFF50 buff13 (.In(x12), .Out(x13));
BUFF50 buff14 (.In(x13), .Out(x14));
BUFF50 buff15 (.In(x14), .Out(x15));
BUFF50 buff16 (.In(x15), .Out(x16));
BUFF50 buff17 (.In(x16), .Out(x17));
BUFF50 buff18 (.In(x17), .Out(x18));
BUFF50 buff19 (.In(x18), .Out(x19));
BUFF50 buff20 (.In(x19), .Out(Out));

endmodule
```

```
//BUFF50
`timescale 1 ns/100 ps

module BUFF50 (In, Out);

input In;

output Out;

wire x1 /*synthesis syn_keep=1 alspreserve=1*/;
wire x2 /*synthesis syn_keep=1 alspreserve=1*/;
wire x3 /*synthesis syn_keep=1 alspreserve=1*/;
wire x4 /*synthesis syn_keep=1 alspreserve=1*/;
wire x5 /*synthesis syn_keep=1 alspreserve=1*/;
wire x6/*synthesis syn_keep=1 alspreserve=1*/;
wire x7/*synthesis syn_keep=1 alspreserve=1*/;
wire x8/*synthesis syn_keep=1 alspreserve=1*/;
wire x9/*synthesis syn_keep=1 alspreserve=1*/;
wire x10/*synthesis syn_keep=1 alspreserve=1*/;
wire x11/*synthesis syn_keep=1 alspreserve=1*/;
wire x12/*synthesis syn_keep=1 alspreserve=1*/;
wire x13/*synthesis syn_keep=1 alspreserve=1*/;
wire x14/*synthesis syn_keep=1 alspreserve=1*/;
wire x15/*synthesis syn_keep=1 alspreserve=1*/;
wire x16/*synthesis syn_keep=1 alspreserve=1*/;
wire x17/*synthesis syn_keep=1 alspreserve=1*/;
wire x18/*synthesis syn_keep=1 alspreserve=1*/;
wire x19/*synthesis syn_keep=1 alspreserve=1*/;
wire x20/*synthesis syn_keep=1 alspreserve=1*/;
wire x21/*synthesis syn_keep=1 alspreserve=1*/;
wire x22/*synthesis syn_keep=1 alspreserve=1*/;
wire x23/*synthesis syn_keep=1 alspreserve=1*/;
wire x24/*synthesis syn_keep=1 alspreserve=1*/;
wire x25/*synthesis syn_keep=1 alspreserve=1*/;
wire x26/*synthesis syn_keep=1 alspreserve=1*/;
wire x27/*synthesis syn_keep=1 alspreserve=1*/;
wire x28/*synthesis syn_keep=1 alspreserve=1*/;
wire x29/*synthesis syn_keep=1 alspreserve=1*/;
wire x30/*synthesis syn_keep=1 alspreserve=1*/;
wire x31/*synthesis syn_keep=1 alspreserve=1*/;
wire x32/*synthesis syn_keep=1 alspreserve=1*/;
wire x33/*synthesis syn_keep=1 alspreserve=1*/;
wire x34/*synthesis syn_keep=1 alspreserve=1*/;
wire x35/*synthesis syn_keep=1 alspreserve=1*/;
wire x36/*synthesis syn_keep=1 alspreserve=1*/;
wire x37/*synthesis syn_keep=1 alspreserve=1*/;
wire x38/*synthesis syn_keep=1 alspreserve=1*/;
wire x39/*synthesis syn_keep=1 alspreserve=1*/;
wire x40/*synthesis syn_keep=1 alspreserve=1*/;
wire x41/*synthesis syn_keep=1 alspreserve=1*/;
wire x42/*synthesis syn_keep=1 alspreserve=1*/;
wire x43/*synthesis syn_keep=1 alspreserve=1*/;
wire x44/*synthesis syn_keep=1 alspreserve=1*/;
wire x45/*synthesis syn_keep=1 alspreserve=1*/;
wire x46/*synthesis syn_keep=1 alspreserve=1*/;
```

```
wire x47/*synthesis syn_keep=1 alspreserve=1*/;
wire x48/*synthesis syn_keep=1 alspreserve=1*/;
wire x49/*synthesis syn_keep=1 alspreserve=1*/;

BUFF buff1 (.A(In), .Y(x1));
BUFF buff2 (.A(x1), .Y(x2));
BUFF buff3 (.A(x2), .Y(x3));
BUFF buff4 (.A(x3), .Y(x4));
BUFF buff5 (.A(x4), .Y(x5));
BUFF buff6 (.A(x5), .Y(x6));
BUFF buff7 (.A(x6), .Y(x7));
BUFF buff8 (.A(x7), .Y(x8));
BUFF buff9 (.A(x8), .Y(x9));
BUFF buff10 (.A(x9), .Y(x10));

BUFF buff11 (.A(x10), .Y(x11));
BUFF buff12 (.A(x11), .Y(x12));
BUFF buff13 (.A(x12), .Y(x13));
BUFF buff14 (.A(x13), .Y(x14));
BUFF buff15 (.A(x14), .Y(x15));
BUFF buff16 (.A(x15), .Y(x16));
BUFF buff17 (.A(x16), .Y(x17));
BUFF buff18 (.A(x17), .Y(x18));
BUFF buff19 (.A(x18), .Y(x19));
BUFF buff20 (.A(x19), .Y(x20));

BUFF buff21 (.A(x20), .Y(x21));
BUFF buff22 (.A(x21), .Y(x22));
BUFF buff23 (.A(x22), .Y(x23));
BUFF buff24 (.A(x23), .Y(x24));
BUFF buff25 (.A(x24), .Y(x25));
BUFF buff26 (.A(x25), .Y(x26));
BUFF buff27 (.A(x26), .Y(x27));
BUFF buff28 (.A(x27), .Y(x28));
BUFF buff29 (.A(x28), .Y(x29));
BUFF buff30 (.A(x29), .Y(x30));

BUFF buff31 (.A(x30), .Y(x31));
BUFF buff32 (.A(x31), .Y(x32));
BUFF buff33 (.A(x32), .Y(x33));
BUFF buff34 (.A(x33), .Y(x34));
BUFF buff35 (.A(x34), .Y(x35));
BUFF buff36 (.A(x35), .Y(x36));
BUFF buff37 (.A(x36), .Y(x37));
BUFF buff38 (.A(x37), .Y(x38));
BUFF buff39 (.A(x38), .Y(x39));
BUFF buff40 (.A(x39), .Y(x40));

BUFF buff41 (.A(x40), .Y(x41));
BUFF buff42 (.A(x41), .Y(x42));
BUFF buff43 (.A(x42), .Y(x43));
BUFF buff44 (.A(x43), .Y(x44));
BUFF buff45 (.A(x44), .Y(x45));
BUFF buff46 (.A(x45), .Y(x46));
BUFF buff47 (.A(x46), .Y(x47));
```

```
BUFF buff48 (.A(x47), .Y(x48));
BUFF buff49 (.A(x48), .Y(x49));
BUFF buff50 (.A(x49), .Y(Out));

endmodule
```

```
//FF2048
`timescale 1 ns/100 ps
module FF2048 (D, Q, CLK, RST, ANDP, ORP, ANDC, ORC);

input D, CLK, RST;
output Q, ANDP, ORP, ANDC, ORC;

wire x1, x2, x3, Q;
wire andp_a, andp_b, andp_c, andp_d, orp_a, orp_b, orp_c, orp_d;
wire andc_a, andc_b, andc_c, andc_d, orc_a, orc_b, orc_c, orc_d;

FF512 dff_a (.D(D), .Q(x1), .CLK(CLK), .RST(RST), .ANDP(andp_a), .ORP(orp_a),
.ANDC(andc_a), .ORC(orc_a));

FF512 dff_b (.D(x1), .Q(x2), .CLK(CLK), .RST(RST), .ANDP(andp_b), .ORP(orp_b),
.ANDC(andc_b), .ORC(orc_b));

FF512 dff_c (.D(x2), .Q(x3), .CLK(CLK), .RST(RST), .ANDP(andp_c), .ORP(orp_c),
.ANDC(andc_c), .ORC(orc_c));

FF512 dff_d (.D(x3), .Q(Q), .CLK(CLK), .RST(RST), .ANDP(andp_d), .ORP(orp_d),
.ANDC(andc_d), .ORC(orc_d));

AND4 and4p (.A(andp_a), .B(andp_b), .C(andp_c), .D(andp_d), .Y(ANDP));
OR4 or4p (.A(orp_a), .B(orp_b), .C(orp_c), .D(orp_d), .Y(ORP));

AND4 and4c (.A(andc_a), .B(andc_b), .C(andc_c), .D(andc_d), .Y(ANDC));
OR4 or4c (.A(orc_a), .B(orc_b), .C(orc_c), .D(orc_d), .Y(ORC));

endmodule
```

```
//FF512
`timescale 1 ns/100 ps
module FF512 (D, Q, CLK, RST, ANDP, ORP, ANDC, ORC);

input D, CLK, RST;
output Q, ANDP, ORP, ANDC, ORC;

wire x1, x2, x3, Q;
wire andp_a, andp_b, andp_c, andp_d, orp_a, orp_b, orp_c, orp_d;
wire andc_a, andc_b, andc_c, andc_d, orc_a, orc_b, orc_c, orc_d;

FF128 dff_a (.D(D), .Q(x1), .CLK(CLK), .RST(RST), .ANDP(andp_a), .ORP(orp_a),
.ANDC(andc_a), .ORC(orc_a));

FF128 dff_b (.D(x1), .Q(x2), .CLK(CLK), .RST(RST), .ANDP(andp_b), .ORP(orp_b),
.ANDC(andc_b), .ORC(orc_b));

FF128 dff_c (.D(x2), .Q(x3), .CLK(CLK), .RST(RST), .ANDP(andp_c), .ORP(orp_c),
.ANDC(andc_c), .ORC(orc_c));

FF128 dff_d (.D(x3), .Q(Q), .CLK(CLK), .RST(RST), .ANDP(andp_d), .ORP(orp_d),
.ANDC(andc_d), .ORC(orc_d));

AND4 and4p (.A(andp_a), .B(andp_b), .C(andp_c), .D(andp_d), .Y(ANDP));
OR4 or4p (.A(orp_a), .B(orp_b), .C(orp_c), .D(orp_d), .Y(ORP));

AND4 and4c (.A(andc_a), .B(andc_b), .C(andc_c), .D(andc_d), .Y(ANDC));
OR4 or4c (.A(orc_a), .B(orc_b), .C(orc_c), .D(orc_d), .Y(ORC));

endmodule
```

```
//FF128
`timescale 1 ns/100 ps
module FF128 (D, Q, CLK, RST, ANDP, ORP, ANDC, ORC);

input D, CLK, RST;
output Q, ANDP, ORP, ANDC, ORC;

wire x1, x2, x3, Q;
wire andp_a, andp_b, andp_c, andp_d, orp_a, orp_b, orp_c, orp_d;
wire andc_a, andc_b, andc_c, andc_d, orc_a, orc_b, orc_c, orc_d;

FF32 dff_a (.D(D), .Q(x1), .CLK(CLK), .RST(RST), .ANDP(andp_a), .ORP(orp_a),
.ANDC(andc_a), .ORC(orc_a));

FF32 dff_b (.D(x1), .Q(x2), .CLK(CLK), .RST(RST), .ANDP(andp_b), .ORP(orp_b),
.ANDC(andc_b), .ORC(orc_b));

FF32 dff_c (.D(x2), .Q(x3), .CLK(CLK), .RST(RST), .ANDP(andp_c), .ORP(orp_c),
.ANDC(andc_c), .ORC(orc_c));

FF32 dff_d (.D(x3), .Q(Q), .CLK(CLK), .RST(RST), .ANDP(andp_d), .ORP(orp_d),
.ANDC(andc_d), .ORC(orc_d));

AND4 and4p (.A(andp_a), .B(andp_b), .C(andp_c), .D(andp_d), .Y(ANDP));
OR4 or4p (.A(orp_a), .B(orp_b), .C(orp_c), .D(orp_d), .Y(ORP));

AND4 and4c (.A(andc_a), .B(andc_b), .C(andc_c), .D(andc_d), .Y(ANDC));
OR4 or4c (.A(orc_a), .B(orc_b), .C(orc_c), .D(orc_d), .Y(ORC));

endmodule
```

```
//FF32
`timescale 1 ns/100 ps
module FF32 (D, Q, CLK, RST, ANDP, ORP, ANDC, ORC);

input D, CLK, RST;
output Q, ANDP, ORP, ANDC, ORC;

wire x1, x2, x3, Q;
wire andp_a, andp_b, andp_c, andp_d, orp_a, orp_b, orp_c, orp_d;
wire andc_a, andc_b, andc_c, andc_d, orc_a, orc_b, orc_c, orc_d;

FF8 dff_a (.D(D), .Q(x1), .CLK(CLK), .RST(RST), .ANDP(andp_a), .ORP(orp_a),
.ANDC(andc_a), .ORC(orc_a));

FF8 dff_b (.D(x1), .Q(x2), .CLK(CLK), .RST(RST), .ANDP(andp_b), .ORP(orp_b),
.ANDC(andc_b), .ORC(orc_b));

FF8 dff_c (.D(x2), .Q(x3), .CLK(CLK), .RST(RST), .ANDP(andp_c), .ORP(orp_c),
.ANDC(andc_c), .ORC(orc_c));

FF8 dff_d (.D(x3), .Q(Q), .CLK(CLK), .RST(RST), .ANDP(andp_d), .ORP(orp_d),
.ANDC(andc_d), .ORC(orc_d));

AND4 and4p (.A(andp_a), .B(andp_b), .C(andp_c), .D(andp_d), .Y(ANDP));
OR4 or4p (.A(orp_a), .B(orp_b), .C(orp_c), .D(orp_d), .Y(ORP));

AND4 and4c (.A(andc_a), .B(andc_b), .C(andc_c), .D(andc_d), .Y(ANDC));
OR4 or4c (.A(orc_a), .B(orc_b), .C(orc_c), .D(orc_d), .Y(ORC));

endmodule
```

```
//FF8
`timescale 1 ns/100 ps

module FF8 (D, Q, CLK, RST, ANDP, ORP, ANDC, ORC);

input D, CLK, RST;
output Q, ANDP, ORP, ANDC, ORC;

wire x1, x2, x3, x4, x5, x6, x7;

DFC1B dff1 (.D(D), .Q(x1), .CLK(CLK), .CLR(RST));
DFP1B dff2 (.D(x1), .Q(x2), .CLK(CLK), .PRE(RST));
DFC1B dff3 (.D(x2), .Q(x3), .CLK(CLK), .CLR(RST));
DFP1B dff4 (.D(x3), .Q(x4), .CLK(CLK), .PRE(RST));
DFC1B dff5 (.D(x4), .Q(x5), .CLK(CLK), .CLR(RST));
DFP1B dff6 (.D(x5), .Q(x6), .CLK(CLK), .PRE(RST));
DFC1B dff7 (.D(x6), .Q(x7), .CLK(CLK), .CLR(RST));
DFP1B dff8 (.D(x7), .Q(Q), .CLK(CLK), .PRE(RST));

AND4 and4p (.A(x2), .B(x4), .C(x6), .D(Q), .Y(ANDP));
OR4 or4p (.A(x2), .B(x4), .C(x6), .D(Q), .Y(ORP));

AND4 and4c (.A(x1), .B(x3), .C(x5), .D(x7), .Y(ANDC));
OR4 or4c (.A(x1), .B(x3), .C(x5), .D(x7), .Y(ORC));

endmodule
```

```
`timescale 1 ns/100 ps
//RAM_Top_Blk.v
module RAM_Top_Blk(Psel0, Psel1, RC_en, RC_clr, RC_clk, Write, Read, Wclk, Rclk,
                Q_RAM);
input Psel0, Psel1, RC_en, RC_clr, RC_clk, Write, Read, Wclk, Rclk;
output [17:0] Q_RAM;

wire Gnd, Vcc;
wire mx0, mx1;
wire [13:0] rc;
wire [7:0] dec;
wire y_0w, y_0r, y_1w, y_1r, y_2w, y_2r, y_3w, y_3r, y_4w, y_4r, y_5w, y_5r,
    y_6w, y_6r, y_7w, y_7r;
wire [17:0] DIN;
wire [17:0] Q_b0;
wire [17:0] Q_b1;
wire [17:0] Q_b2;
wire [17:0] Q_b3;
wire [17:0] Q_b4;
wire [17:0] Q_b5;
wire [17:0] Q_b6;
wire [17:0] Q_b7;

GND gnd_0(.Y(Gnd));
VCC vcc_0(.Y(Vcc));

mux_2x1 mux_0(.Data0_port(Gnd), .Data1_port(Vcc), .Sel0(Psel0), .Result(mx0));
mux_2x1 mux_1(.Data0_port(Gnd), .Data1_port(Vcc), .Sel0(Psel1), .Result(mx1));

counter_14 counter_0(.Enable(RC_en), .Aclr(RC_clr), .Clock(RC_clk), .Q(rc));

decoder_3x8 decoder_0(.Data0(rc[11]), .Data1(rc[12]), .Data2(rc[13]), .Eq(dec));

NAND2 nand_0w(.A(dec[0]), .B(Write), .Y(y_0w));
NAND2 nand_0r(.A(dec[0]), .B(Read), .Y(y_0r));

ram_2048x18 ram_blk0(.Data(DIN), .Q(Q_b0), .WAddress(rc[10:0]), .RAddress(rc[10:0]),
                .WE(y_0w), .RE(y_0r), .WClock(Wclk), .RClock(Rclk));

assign DIN[0]=mx0, DIN[1]=mx1, DIN[2]=mx0, DIN[3]=mx1, DIN[4]=mx0, DIN[5]=mx1,
     DIN[6]=mx0, DIN[7]=mx1, DIN[8]=mx0, DIN[9]=mx1, DIN[10]=mx0, DIN[11]=mx1,
     DIN[12]=mx0, DIN[13]=mx1, DIN[14]=mx0, DIN[15]=mx1, DIN[16]=mx0, DIN[17]=mx1;

NAND2 nand_1w(.A(dec[1]), .B(Write), .Y(y_1w));
NAND2 nand_1r(.A(dec[1]), .B(Read), .Y(y_1r));

ram_2048x18 ram_blk1(.Data(DIN), .Q(Q_b1), .WAddress(rc[10:0]), .RAddress(rc[10:0]),
                .WE(y_1w), .RE(y_1r), .WClock(Wclk), .RClock(Rclk));

NAND2 nand_2w(.A(dec[2]), .B(Write), .Y(y_2w));
NAND2 nand_2r(.A(dec[2]), .B(Read), .Y(y_2r));

ram_2048x18 ram_blk2(.Data(DIN),
                .Q(Q_b2), .WAddress(rc[10:0]), .RAddress(rc[10:0]),
```

```
                .WE(y_2w), .RE(y_2r), .WClock(Wclk), .RClock(Rclk));

NAND2 nand_3w(.A(dec[3]), .B(Write), .Y(y_3w));
NAND2 nand_3r(.A(dec[3]), .B(Read), .Y(y_3r));

ram_2048x18 ram_blk3(.Data(DIN),
                .Q(Q_b3), .WAddress(rc[10:0]), .RAddress(rc[10:0]),
                .WE(y_3w), .RE(y_3r), .WClock(Wclk), .RClock(Rclk));

NAND2 nand_4w(.A(dec[4]), .B(Write), .Y(y_4w));
NAND2 nand_4r(.A(dec[4]), .B(Read), .Y(y_4r));

ram_2048x18 ram_blk4(.Data(DIN),
                .Q(Q_b4), .WAddress(rc[10:0]), .RAddress(rc[10:0]),
                .WE(y_4w), .RE(y_4r), .WClock(Wclk), .RClock(Rclk));

NAND2 nand_5w(.A(dec[5]), .B(Write), .Y(y_5w));
NAND2 nand_5r(.A(dec[5]), .B(Read), .Y(y_5r));

ram_2048x18 ram_blk5(.Data(DIN),
                .Q(Q_b5), .WAddress(rc[10:0]), .RAddress(rc[10:0]),
                .WE(y_5w), .RE(y_5r), .WClock(Wclk), .RClock(Rclk));

NAND2 nand_6w(.A(dec[6]), .B(Write), .Y(y_6w));
NAND2 nand_6r(.A(dec[6]), .B(Read), .Y(y_6r));

ram_2048x18 ram_blk6(.Data(DIN),
                .Q(Q_b6), .WAddress(rc[10:0]), .RAddress(rc[10:0]),
                .WE(y_6w), .RE(y_6r), .WClock(Wclk), .RClock(Rclk));

NAND2 nand_7w(.A(dec[7]), .B(Write), .Y(y_7w));
NAND2 nand_7r(.A(dec[7]), .B(Read), .Y(y_7r));

ram_2048x18 ram_blk7(.Data(DIN),
                .Q(Q_b7), .WAddress(rc[10:0]), .RAddress(rc[10:0]),
                .WE(y_7w), .RE(y_7r), .WClock(Wclk), .RClock(Rclk));

mux_18x8 mux_18x8_0(.Data0_port(Q_b0), .Data1_port(Q_b1), .Data2_port(Q_b2),
                .Data3_port(Q_b3), .Data4_port(Q_b4), .Data5_port(Q_b5),
                .Data6_port(Q_b6), .Data7_port(Q_b7), .Sel0(rc[11]),
                .Sel1(rc[12]), .Sel2(rc[13]), .Result(Q_RAM));

endmodule
```

```
`timescale 1 ns/100 ps
//Version: 6.0 SP3 6.0.30.3

module mux_2x1(Data0_port,Data1_port,Sel0,Result);
input Data0_port, Data1_port, Sel0;
output  Result;


    MX2 MX2_Result(.A(Data0_port), .B(Data1_port), .S(Sel0), .Y(
        Result));

endmodule
```

```
`timescale 1 ns/100 ps
// Version: Designer v5.0 5.0.0.10

module counter_14(Enable,Aclr,Clock,Q);
input Enable, Aclr, Clock;
output [13:0] Q;

    wire ClrAux_0_net, ClrAux_7_net, MX2_1_Y, MX2_8_Y, MX2_5_Y,
       CM8_0_Y, MX2_11_Y, MX2_10_Y, MX2_4_Y, MX2_6_Y, MX2_7_Y,
       MX2_0_Y, MX2_3_Y, MX2_9_Y, MX2_2_Y, MX2_12_Y, VCC, GND;

    VCC VCC_1_net(.Y(VCC));
    GND GND_1_net(.Y(GND));
    DFC1D DFC1D_Q_7_inst(.D(MX2_1_Y), .CLK(Q[6]), .CLR(
       ClrAux_7_net), .Q(Q[7]));
    DFC1D DFC1D_Q_1_inst(.D(MX2_8_Y), .CLK(Q[0]), .CLR(
       ClrAux_0_net), .Q(Q[1]));
    BUFF BUFF_ClrAux_0_inst(.A(Aclr), .Y(ClrAux_0_net));
    MX2 MX2_9(.A(VCC), .B(GND), .S(Q[9]), .Y(MX2_9_Y));
    DFC1D DFC1D_Q_2_inst(.D(MX2_7_Y), .CLK(Q[1]), .CLR(
       ClrAux_0_net), .Q(Q[2]));
    MX2 MX2_0(.A(VCC), .B(GND), .S(Q[8]), .Y(MX2_0_Y));
    DFC1D DFC1D_Q_12_inst(.D(MX2_5_Y), .CLK(Q[11]), .CLR(
       ClrAux_7_net), .Q(Q[12]));
    DFC1D DFC1D_Q_3_inst(.D(MX2_12_Y), .CLK(Q[2]), .CLR(
       ClrAux_0_net), .Q(Q[3]));
    DFC1D DFC1D_Q_4_inst(.D(MX2_6_Y), .CLK(Q[3]), .CLR(
       ClrAux_0_net), .Q(Q[4]));
    CM8 CM8_0(.D0(GND), .D1(VCC), .D2(VCC), .D3(GND), .S00(Q[0]),
       .S01(VCC), .S10(Enable), .S11(GND), .Y(CM8_0_Y));
    MX2 MX2_11(.A(VCC), .B(GND), .S(Q[11]), .Y(MX2_11_Y));
    DFC1B DFC1B_Q_0_inst(.D(CM8_0_Y), .CLK(Clock), .CLR(
       ClrAux_0_net), .Q(Q[0]));
    MX2 MX2_6(.A(VCC), .B(GND), .S(Q[4]), .Y(MX2_6_Y));
    DFC1D DFC1D_Q_11_inst(.D(MX2_11_Y), .CLK(Q[10]), .CLR(
       ClrAux_7_net), .Q(Q[11]));
    MX2 MX2_3(.A(VCC), .B(GND), .S(Q[13]), .Y(MX2_3_Y));
    MX2 MX2_10(.A(VCC), .B(GND), .S(Q[5]), .Y(MX2_10_Y));
    BUFF BUFF_ClrAux_7_inst(.A(Aclr), .Y(ClrAux_7_net));
    MX2 MX2_4(.A(VCC), .B(GND), .S(Q[10]), .Y(MX2_4_Y));
    DFC1D DFC1D_Q_5_inst(.D(MX2_10_Y), .CLK(Q[4]), .CLR(
       ClrAux_0_net), .Q(Q[5]));
    DFC1D DFC1D_Q_9_inst(.D(MX2_9_Y), .CLK(Q[8]), .CLR(
       ClrAux_7_net), .Q(Q[9]));
    MX2 MX2_5(.A(VCC), .B(GND), .S(Q[12]), .Y(MX2_5_Y));
    MX2 MX2_8(.A(VCC), .B(GND), .S(Q[1]), .Y(MX2_8_Y));
    DFC1D DFC1D_Q_8_inst(.D(MX2_0_Y), .CLK(Q[7]), .CLR(
       ClrAux_7_net), .Q(Q[8]));
    MX2 MX2_2(.A(VCC), .B(GND), .S(Q[6]), .Y(MX2_2_Y));
    MX2 MX2_7(.A(VCC), .B(GND), .S(Q[2]), .Y(MX2_7_Y));
    MX2 MX2_1(.A(VCC), .B(GND), .S(Q[7]), .Y(MX2_1_Y));
    DFC1D DFC1D_Q_6_inst(.D(MX2_2_Y), .CLK(Q[5]), .CLR(
       ClrAux_0_net), .Q(Q[6]));
    DFC1D DFC1D_Q_13_inst(.D(MX2_3_Y), .CLK(Q[12]), .CLR(
       ClrAux_7_net), .Q(Q[13]));
```

```
    DFC1D DFC1D_Q_10_inst(.D(MX2_4_Y), .CLK(Q[9]), .CLR(
        ClrAux_7_net), .Q(Q[10]));
    MX2 MX2_12(.A(VCC), .B(GND), .S(Q[3]), .Y(MX2_12_Y));

endmodule
```

```
`timescale 1 ns/100 ps
// Version: 6.0 SP3 6.0.30.3

module decoder_3x8(Data0,Data1,Data2,Eq);
input Data0, Data1, Data2;
output [7:0] Eq;


    AND3A AND3A_Eq_3_inst(.A(Data2), .B(Data0), .C(Data1), .Y(
       Eq[3]));
    AND3 AND3_Eq_7_inst(.A(Data1), .B(Data0), .C(Data2), .Y(Eq[7])
       );
    AND3B AND3B_Eq_4_inst(.A(Data0), .B(Data1), .C(Data2), .Y(
       Eq[4]));
    AND3A AND3A_Eq_5_inst(.A(Data1), .B(Data0), .C(Data2), .Y(
       Eq[5]));
    AND3B AND3B_Eq_1_inst(.A(Data1), .B(Data2), .C(Data0), .Y(
       Eq[1]));
    AND3A AND3A_Eq_6_inst(.A(Data0), .B(Data1), .C(Data2), .Y(
       Eq[6]));
    AND3B AND3B_Eq_2_inst(.A(Data0), .B(Data2), .C(Data1), .Y(
       Eq[2]));
    AND3C AND3C_Eq_0_inst(.A(Data0), .B(Data1), .C(Data2), .Y(
       Eq[0]));

endmodule
```

```
`timescale 1 ns/100 ps
// Version: Designer v5.0 5.0.0.10

module ram_2048x18(Data,Q,WAddress,RAddress,WE,RE,WClock,RClock);
input [17:0] Data;
output [17:0] Q;
input [10:0] WAddress, RAddress;
input WE, RE, WClock, RClock;

    wire INV_1_Y, INV_0_Y, VCC, GND;

    VCC VCC_1_net(.Y(VCC));
    GND GND_1_net(.Y(GND));
    INV INV_0(.A(RE), .Y(INV_0_Y));
    RAM64K36P RAM64K36P_Q_9_inst(.WCLK(WClock), .RCLK(RClock),
        .DEPTH0(VCC), .DEPTH1(VCC), .DEPTH2(GND), .DEPTH3(GND),
        .WEN(INV_1_Y), .WW0(VCC), .WW1(VCC), .WW2(GND), .WRAD0(
        WAddress[0]), .WRAD1(WAddress[1]), .WRAD2(WAddress[2]),
        .WRAD3(WAddress[3]), .WRAD4(WAddress[4]), .WRAD5(
        WAddress[5]), .WRAD6(WAddress[6]), .WRAD7(WAddress[7]),
        .WRAD8(WAddress[8]), .WRAD9(WAddress[9]), .WRAD10(
        WAddress[10]), .WRAD11(GND), .WRAD12(GND), .WRAD13(GND),
        .WRAD14(GND), .WRAD15(GND), .WD0(Data[9]), .WD1(Data[10]),
        .WD2(Data[11]), .WD3(Data[12]), .WD4(Data[13]), .WD5(
        Data[14]), .WD6(Data[15]), .WD7(Data[16]), .WD8(Data[17]),
        .WD9(GND), .WD10(GND), .WD11(GND), .WD12(GND), .WD13(GND),
        .WD14(GND), .WD15(GND), .WD16(GND), .WD17(GND), .WD18(GND)
        , .WD19(GND), .WD20(GND), .WD21(GND), .WD22(GND), .WD23(
        GND), .WD24(GND), .WD25(GND), .WD26(GND), .WD27(GND),
        .WD28(GND), .WD29(GND), .WD30(GND), .WD31(GND), .WD32(GND)
        , .WD33(GND), .WD34(GND), .WD35(GND), .REN(INV_0_Y), .RW0(
        VCC), .RW1(VCC), .RW2(GND), .RDAD0(RAddress[0]), .RDAD1(
        RAddress[1]), .RDAD2(RAddress[2]), .RDAD3(RAddress[3]),
        .RDAD4(RAddress[4]), .RDAD5(RAddress[5]), .RDAD6(
        RAddress[6]), .RDAD7(RAddress[7]), .RDAD8(RAddress[8]),
        .RDAD9(RAddress[9]), .RDAD10(RAddress[10]), .RDAD11(GND),
        .RDAD12(GND), .RDAD13(GND), .RDAD14(GND), .RDAD15(GND),
        .RD0(Q[9]), .RD1(Q[10]), .RD2(Q[11]), .RD3(Q[12]), .RD4(
        Q[13]), .RD5(Q[14]), .RD6(Q[15]), .RD7(Q[16]), .RD8(Q[17])
        , .RD9(), .RD10(), .RD11(), .RD12(), .RD13(), .RD14(),
        .RD15(), .RD16(), .RD17(), .RD18(), .RD19(), .RD20(),
        .RD21(), .RD22(), .RD23(), .RD24(), .RD25(), .RD26(),
        .RD27(), .RD28(), .RD29(), .RD30(), .RD31(), .RD32(),
        .RD33(), .RD34(), .RD35());
    INV INV_1(.A(WE), .Y(INV_1_Y));
    RAM64K36P RAM64K36P_Q_0_inst(.WCLK(WClock), .RCLK(RClock),
        .DEPTH0(VCC), .DEPTH1(VCC), .DEPTH2(GND), .DEPTH3(GND),
        .WEN(INV_1_Y), .WW0(VCC), .WW1(VCC), .WW2(GND), .WRAD0(
        WAddress[0]), .WRAD1(WAddress[1]), .WRAD2(WAddress[2]),
        .WRAD3(WAddress[3]), .WRAD4(WAddress[4]), .WRAD5(
        WAddress[5]), .WRAD6(WAddress[6]), .WRAD7(WAddress[7]),
        .WRAD8(WAddress[8]), .WRAD9(WAddress[9]), .WRAD10(
        WAddress[10]), .WRAD11(GND), .WRAD12(GND), .WRAD13(GND),
        .WRAD14(GND), .WRAD15(GND), .WD0(Data[0]), .WD1(Data[1]),
        .WD2(Data[2]), .WD3(Data[3]), .WD4(Data[4]), .WD5(Data[5])
```

```
    , .WD6(Data[6]), .WD7(Data[7]), .WD8(Data[8]), .WD9(GND),
    .WD10(GND), .WD11(GND), .WD12(GND), .WD13(GND), .WD14(GND)
    , .WD15(GND), .WD16(GND), .WD17(GND), .WD18(GND), .WD19(
    GND), .WD20(GND), .WD21(GND), .WD22(GND), .WD23(GND),
    .WD24(GND), .WD25(GND), .WD26(GND), .WD27(GND), .WD28(GND)
    , .WD29(GND), .WD30(GND), .WD31(GND), .WD32(GND), .WD33(
    GND), .WD34(GND), .WD35(GND), .REN(INV_0_Y), .RW0(VCC),
    .RW1(VCC), .RW2(GND), .RDAD0(RAddress[0]), .RDAD1(
    RAddress[1]), .RDAD2(RAddress[2]), .RDAD3(RAddress[3]),
    .RDAD4(RAddress[4]), .RDAD5(RAddress[5]), .RDAD6(
    RAddress[6]), .RDAD7(RAddress[7]), .RDAD8(RAddress[8]),
    .RDAD9(RAddress[9]), .RDAD10(RAddress[10]), .RDAD11(GND),
    .RDAD12(GND), .RDAD13(GND), .RDAD14(GND), .RDAD15(GND),
    .RD0(Q[0]), .RD1(Q[1]), .RD2(Q[2]), .RD3(Q[3]), .RD4(Q[4])
    , .RD5(Q[5]), .RD6(Q[6]), .RD7(Q[7]), .RD8(Q[8]), .RD9(),
    .RD10(), .RD11(), .RD12(), .RD13(), .RD14(), .RD15(),
    .RD16(), .RD17(), .RD18(), .RD19(), .RD20(), .RD21(),
    .RD22(), .RD23(), .RD24(), .RD25(), .RD26(), .RD27(),
    .RD28(), .RD29(), .RD30(), .RD31(), .RD32(), .RD33(),
    .RD34(), .RD35());

endmodule
```

```
`timescale 1 ns/100 ps
// Version: Designer v5.0 5.0.0.10

module mux_18x8(Data0_port,Data1_port,Data2_port,Data3_port,
      Data4_port,Data5_port,Data6_port,Data7_port,Sel0,Sel1,
      Sel2,Result);
input [17:0] Data0_port, Data1_port, Data2_port, Data3_port,
      Data4_port, Data5_port, Data6_port, Data7_port;
input Sel0, Sel1, Sel2;
output [17:0] Result;

   wire SelAux_0_0_net, SelAux_0_6_net, SelAux_0_12_net,
      SelAux_0_18_net, SelAux_0_24_net, SelAux_0_30_net,
      SelAux_1_0_net, SelAux_1_6_net, SelAux_1_12_net,
      SelAux_1_18_net, SelAux_1_24_net, SelAux_1_30_net,
      SelAux_2_0_net, SelAux_2_6_net, SelAux_2_12_net, MX4_35_Y,
      MX4_8_Y, MX4_31_Y, MX4_15_Y, MX4_23_Y, MX4_0_Y, MX4_26_Y,
      MX4_16_Y, MX4_29_Y, MX4_1_Y, MX4_13_Y, MX4_32_Y, MX4_18_Y,
      MX4_27_Y, MX4_21_Y, MX4_28_Y, MX4_20_Y, MX4_10_Y,
      MX4_12_Y, MX4_4_Y, MX4_6_Y, MX4_7_Y, MX4_3_Y, MX4_34_Y,
      MX4_5_Y, MX4_33_Y, MX4_22_Y, MX4_2_Y, MX4_11_Y, MX4_30_Y,
      MX4_14_Y, MX4_24_Y, MX4_19_Y, MX4_25_Y, MX4_17_Y, MX4_9_Y;

   MX4 MX4_16(.D0(Data0_port[7]), .D1(Data1_port[7]), .D2(
      Data2_port[7]), .D3(Data3_port[7]), .S0(SelAux_0_12_net),
      .S1(SelAux_1_12_net), .Y(MX4_16_Y));
   BUFF BUFF_SelAux_1_12_inst(.A(Sel1), .Y(SelAux_1_12_net));
   MX2 MX2_Result_1_inst(.A(MX4_8_Y), .B(MX4_4_Y), .S(
      SelAux_2_0_net), .Y(Result[1]));
   BUFF BUFF_SelAux_0_12_inst(.A(Sel0), .Y(SelAux_0_12_net));
   MX2 MX2_Result_8_inst(.A(MX4_29_Y), .B(MX4_22_Y), .S(
      SelAux_2_6_net), .Y(Result[8]));
   MX4 MX4_1(.D0(Data0_port[9]), .D1(Data1_port[9]), .D2(
      Data2_port[9]), .D3(Data3_port[9]), .S0(SelAux_0_18_net),
      .S1(SelAux_1_18_net), .Y(MX4_1_Y));
   MX2 MX2_Result_2_inst(.A(MX4_31_Y), .B(MX4_6_Y), .S(
      SelAux_2_0_net), .Y(Result[2]));
   MX4 MX4_3(.D0(Data4_port[4]), .D1(Data5_port[4]), .D2(
      Data6_port[4]), .D3(Data7_port[4]), .S0(SelAux_0_6_net),
      .S1(SelAux_1_6_net), .Y(MX4_3_Y));
   MX2 MX2_Result_5_inst(.A(MX4_0_Y), .B(MX4_34_Y), .S(
      SelAux_2_0_net), .Y(Result[5]));
   MX2 MX2_Result_0_inst(.A(MX4_35_Y), .B(MX4_12_Y), .S(
      SelAux_2_0_net), .Y(Result[0]));
   MX2 MX2_Result_14_inst(.A(MX4_21_Y), .B(MX4_19_Y), .S(
      SelAux_2_12_net), .Y(Result[14]));
   MX2 MX2_Result_17_inst(.A(MX4_10_Y), .B(MX4_9_Y), .S(
      SelAux_2_12_net), .Y(Result[17]));
   MX4 MX4_7(.D0(Data4_port[3]), .D1(Data5_port[3]), .D2(
      Data6_port[3]), .D3(Data7_port[3]), .S0(SelAux_0_6_net),
      .S1(SelAux_1_6_net), .Y(MX4_7_Y));
   MX4 MX4_23(.D0(Data0_port[4]), .D1(Data1_port[4]), .D2(
      Data2_port[4]), .D3(Data3_port[4]), .S0(SelAux_0_6_net),
      .S1(SelAux_1_6_net), .Y(MX4_23_Y));
   MX4 MX4_4(.D0(Data4_port[1]), .D1(Data5_port[1]), .D2(
```

```
    Data6_port[1]), .D3(Data7_port[1]), .S0(SelAux_0_0_net),
    .S1(SelAux_1_0_net), .Y(MX4_4_Y));
MX4 MX4_27(.D0(Data0_port[13]), .D1(Data1_port[13]), .D2(
    Data2_port[13]), .D3(Data3_port[13]), .S0(SelAux_0_24_net)
    , .S1(SelAux_1_24_net), .Y(MX4_27_Y));
BUFF BUFF_SelAux_2_12_inst(.A(Sel2), .Y(SelAux_2_12_net));
MX4 MX4_14(.D0(Data4_port[12]), .D1(Data5_port[12]), .D2(
    Data6_port[12]), .D3(Data7_port[12]), .S0(SelAux_0_24_net)
    , .S1(SelAux_1_24_net), .Y(MX4_14_Y));
MX4 MX4_5(.D0(Data4_port[6]), .D1(Data5_port[6]), .D2(
    Data6_port[6]), .D3(Data7_port[6]), .S0(SelAux_0_12_net),
    .S1(SelAux_1_12_net), .Y(MX4_5_Y));
MX4 MX4_9(.D0(Data4_port[17]), .D1(Data5_port[17]), .D2(
    Data6_port[17]), .D3(Data7_port[17]), .S0(SelAux_0_30_net)
    , .S1(SelAux_1_30_net), .Y(MX4_9_Y));
MX4 MX4_34(.D0(Data4_port[5]), .D1(Data5_port[5]), .D2(
    Data6_port[5]), .D3(Data7_port[5]), .S0(SelAux_0_6_net),
    .S1(SelAux_1_6_net), .Y(MX4_34_Y));
MX4 MX4_29(.D0(Data0_port[8]), .D1(Data1_port[8]), .D2(
    Data2_port[8]), .D3(Data3_port[8]), .S0(SelAux_0_12_net),
    .S1(SelAux_1_12_net), .Y(MX4_29_Y));
MX2 MX2_Result_3_inst(.A(MX4_15_Y), .B(MX4_7_Y), .S(
    SelAux_2_0_net), .Y(Result[3]));
MX4 MX4_21(.D0(Data0_port[14]), .D1(Data1_port[14]), .D2(
    Data2_port[14]), .D3(Data3_port[14]), .S0(SelAux_0_24_net)
    , .S1(SelAux_1_24_net), .Y(MX4_21_Y));
MX4 MX4_28(.D0(Data0_port[15]), .D1(Data1_port[15]), .D2(
    Data2_port[15]), .D3(Data3_port[15]), .S0(SelAux_0_30_net)
    , .S1(SelAux_1_30_net), .Y(MX4_28_Y));
MX4 MX4_22(.D0(Data4_port[8]), .D1(Data5_port[8]), .D2(
    Data6_port[8]), .D3(Data7_port[8]), .S0(SelAux_0_12_net),
    .S1(SelAux_1_12_net), .Y(MX4_22_Y));
BUFF BUFF_SelAux_1_6_inst(.A(Sel1), .Y(SelAux_1_6_net));
MX4 MX4_20(.D0(Data0_port[16]), .D1(Data1_port[16]), .D2(
    Data2_port[16]), .D3(Data3_port[16]), .S0(SelAux_0_30_net)
    , .S1(SelAux_1_30_net), .Y(MX4_20_Y));
BUFF BUFF_SelAux_2_0_inst(.A(Sel2), .Y(SelAux_2_0_net));
MX4 MX4_25(.D0(Data4_port[15]), .D1(Data5_port[15]), .D2(
    Data6_port[15]), .D3(Data7_port[15]), .S0(SelAux_0_30_net)
    , .S1(SelAux_1_30_net), .Y(MX4_25_Y));
MX2 MX2_Result_16_inst(.A(MX4_20_Y), .B(MX4_17_Y), .S(
    SelAux_2_12_net), .Y(Result[16]));
BUFF BUFF_SelAux_1_18_inst(.A(Sel1), .Y(SelAux_1_18_net));
BUFF BUFF_SelAux_0_0_inst(.A(Sel0), .Y(SelAux_0_0_net));
MX2 MX2_Result_13_inst(.A(MX4_27_Y), .B(MX4_24_Y), .S(
    SelAux_2_12_net), .Y(Result[13]));
BUFF BUFF_SelAux_0_18_inst(.A(Sel0), .Y(SelAux_0_18_net));
MX4 MX4_8(.D0(Data0_port[1]), .D1(Data1_port[1]), .D2(
    Data2_port[1]), .D3(Data3_port[1]), .S0(SelAux_0_0_net),
    .S1(SelAux_1_0_net), .Y(MX4_8_Y));
BUFF BUFF_SelAux_1_30_inst(.A(Sel1), .Y(SelAux_1_30_net));
MX4 MX4_26(.D0(Data0_port[6]), .D1(Data1_port[6]), .D2(
    Data2_port[6]), .D3(Data3_port[6]), .S0(SelAux_0_12_net),
    .S1(SelAux_1_12_net), .Y(MX4_26_Y));
MX4 MX4_6(.D0(Data4_port[2]), .D1(Data5_port[2]), .D2(
```

```
    Data6_port[2]), .D3(Data7_port[2]), .S0(SelAux_0_0_net),
    .S1(SelAux_1_0_net), .Y(MX4_6_Y));
MX2 MX2_Result_12_inst(.A(MX4_18_Y), .B(MX4_14_Y), .S(
    SelAux_2_12_net), .Y(Result[12]));
MX4 MX4_13(.D0(Data0_port[10]), .D1(Data1_port[10]), .D2(
    Data2_port[10]), .D3(Data3_port[10]), .S0(SelAux_0_18_net)
    , .S1(SelAux_1_18_net), .Y(MX4_13_Y));
MX2 MX2_Result_11_inst(.A(MX4_32_Y), .B(MX4_30_Y), .S(
    SelAux_2_6_net), .Y(Result[11]));
BUFF BUFF_SelAux_0_6_inst(.A(Sel0), .Y(SelAux_0_6_net));
MX4 MX4_17(.D0(Data4_port[16]), .D1(Data5_port[16]), .D2(
    Data6_port[16]), .D3(Data7_port[16]), .S0(SelAux_0_30_net)
    , .S1(SelAux_1_30_net), .Y(MX4_17_Y));
BUFF BUFF_SelAux_1_0_inst(.A(Sel1), .Y(SelAux_1_0_net));
MX4 MX4_2(.D0(Data4_port[9]), .D1(Data5_port[9]), .D2(
    Data6_port[9]), .D3(Data7_port[9]), .S0(SelAux_0_18_net),
    .S1(SelAux_1_18_net), .Y(MX4_2_Y));
MX4 MX4_33(.D0(Data4_port[7]), .D1(Data5_port[7]), .D2(
    Data6_port[7]), .D3(Data7_port[7]), .S0(SelAux_0_12_net),
    .S1(SelAux_1_12_net), .Y(MX4_33_Y));
MX2 MX2_Result_15_inst(.A(MX4_28_Y), .B(MX4_25_Y), .S(
    SelAux_2_12_net), .Y(Result[15]));
MX4 MX4_19(.D0(Data4_port[14]), .D1(Data5_port[14]), .D2(
    Data6_port[14]), .D3(Data7_port[14]), .S0(SelAux_0_24_net)
    , .S1(SelAux_1_24_net), .Y(MX4_19_Y));
BUFF BUFF_SelAux_2_6_inst(.A(Sel2), .Y(SelAux_2_6_net));
MX4 MX4_11(.D0(Data4_port[10]), .D1(Data5_port[10]), .D2(
    Data6_port[10]), .D3(Data7_port[10]), .S0(SelAux_0_18_net)
    , .S1(SelAux_1_18_net), .Y(MX4_11_Y));
MX2 MX2_Result_9_inst(.A(MX4_1_Y), .B(MX4_2_Y), .S(
    SelAux_2_6_net), .Y(Result[9]));
MX4 MX4_18(.D0(Data0_port[12]), .D1(Data1_port[12]), .D2(
    Data2_port[12]), .D3(Data3_port[12]), .S0(SelAux_0_24_net)
    , .S1(SelAux_1_24_net), .Y(MX4_18_Y));
BUFF BUFF_SelAux_0_24_inst(.A(Sel0), .Y(SelAux_0_24_net));
BUFF BUFF_SelAux_1_24_inst(.A(Sel1), .Y(SelAux_1_24_net));
MX4 MX4_12(.D0(Data4_port[0]), .D1(Data5_port[0]), .D2(
    Data6_port[0]), .D3(Data7_port[0]), .S0(SelAux_0_0_net),
    .S1(SelAux_1_0_net), .Y(MX4_12_Y));
MX4 MX4_31(.D0(Data0_port[2]), .D1(Data1_port[2]), .D2(
    Data2_port[2]), .D3(Data3_port[2]), .S0(SelAux_0_0_net),
    .S1(SelAux_1_0_net), .Y(MX4_31_Y));
MX2 MX2_Result_7_inst(.A(MX4_16_Y), .B(MX4_33_Y), .S(
    SelAux_2_6_net), .Y(Result[7]));
MX4 MX4_0(.D0(Data0_port[5]), .D1(Data1_port[5]), .D2(
    Data2_port[5]), .D3(Data3_port[5]), .S0(SelAux_0_6_net),
    .S1(SelAux_1_6_net), .Y(MX4_0_Y));
MX4 MX4_10(.D0(Data0_port[17]), .D1(Data1_port[17]), .D2(
    Data2_port[17]), .D3(Data3_port[17]), .S0(SelAux_0_30_net)
    , .S1(SelAux_1_30_net), .Y(MX4_10_Y));
MX4 MX4_32(.D0(Data0_port[11]), .D1(Data1_port[11]), .D2(
    Data2_port[11]), .D3(Data3_port[11]), .S0(SelAux_0_18_net)
    , .S1(SelAux_1_18_net), .Y(MX4_32_Y));
MX4 MX4_30(.D0(Data4_port[11]), .D1(Data5_port[11]), .D2(
    Data6_port[11]), .D3(Data7_port[11]), .S0(SelAux_0_18_net)
```

```
    , .S1(SelAux_1_18_net), .Y(MX4_30_Y));
  MX4 MX4_15(.D0(Data0_port[3]), .D1(Data1_port[3]), .D2(
    Data2_port[3]), .D3(Data3_port[3]), .S0(SelAux_0_6_net),
    .S1(SelAux_1_6_net), .Y(MX4_15_Y));
  MX4 MX4_35(.D0(Data0_port[0]), .D1(Data1_port[0]), .D2(
    Data2_port[0]), .D3(Data3_port[0]), .S0(SelAux_0_0_net),
    .S1(SelAux_1_0_net), .Y(MX4_35_Y));
  MX2 MX2_Result_6_inst(.A(MX4_26_Y), .B(MX4_5_Y), .S(
    SelAux_2_6_net), .Y(Result[6]));
  MX4 MX4_24(.D0(Data4_port[13]), .D1(Data5_port[13]), .D2(
    Data6_port[13]), .D3(Data7_port[13]), .S0(SelAux_0_24_net)
    , .S1(SelAux_1_24_net), .Y(MX4_24_Y));
  BUFF BUFF_SelAux_0_30_inst(.A(Sel0), .Y(SelAux_0_30_net));
  MX2 MX2_Result_10_inst(.A(MX4_13_Y), .B(MX4_11_Y), .S(
    SelAux_2_6_net), .Y(Result[10]));
  MX2 MX2_Result_4_inst(.A(MX4_23_Y), .B(MX4_3_Y), .S(
    SelAux_2_0_net), .Y(Result[4]));

endmodule
```