Total Ionizing Dose Test Report

No. 10T-RTAX2000S-D4CYG1

December 10, 2010

Table of Contents

I. Summary Table 3

II. Total Ionizing Dose (TID) Testing 3

A. Device-Under-Test (DUT) and Irradiation Parameters 4
B. Test Method 5
C. Design and Parametric Measurements 6

III. Test Results 7

A. Functionality 7
B. Power Supply Current (I_{CCA} and I_{CCI}) 7
C. Single-Ended Input Logic Threshold (V_{IL}/V_{IH}) 11
D. Differential Input (LVPECL) Threshold Voltage (V_{IL}/V_{IH}) 13
E. Output-Drive Voltage (V_{OL}/V_{OH}) 15
F. Propagation Delay 16
G. Transition Characteristics 18

Appendix A DUT Bias 30

Appendix B DUT Design Schematics and Verilog Files 32

TOTAL IONIZING DOSE TEST REPORT

No. 10T-RTAX2000S-D4CYG1

Dec. 10, 2010

CK Huang, J.J. Wang

(650) 318-4209, (650) 318-4576

chang-kai.huang@microsemi.com*, *jih-jong.wang@microsemi.com

I. Summary Table

Parameter	*Tolerance*
1. Gross Functionality	Passed 300 krad(SiO_2)
2. Power Supply Current (I_{CCA}/I_{CCI})	Passed 200 krad(SiO_2)
3. Input Threshold (V_{TIL}/V_{IH})	Passed 300 krad(SiO_2)
4. Output Drive (V_{OL}/V_{OH})	Passed 300 krad(SiO_2)
5. Propagation Delay	Passed 300 krad(SiO_2) for 10% degradation criterion
6. Transition Charateristics	Passed 300 krad(SiO_2)

II. Total Ionizing Dose (TID) Testing

This testing is designed on the base of an extensive database (see 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.

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 2.7k-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_AX2000S_TID
Die Lot Number	D4CYG1
Quantity Tested	6
Serial Number	200 krad(SiO_2): 4678, 4729, 4822 300 krad(SiO_2): 4575, 4638, 4648
Radiation Facility	Defense Microelectronics Activity
Radiation Source	Co-60
Dose Rate (±5%)	5 krad(SiO_2)/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

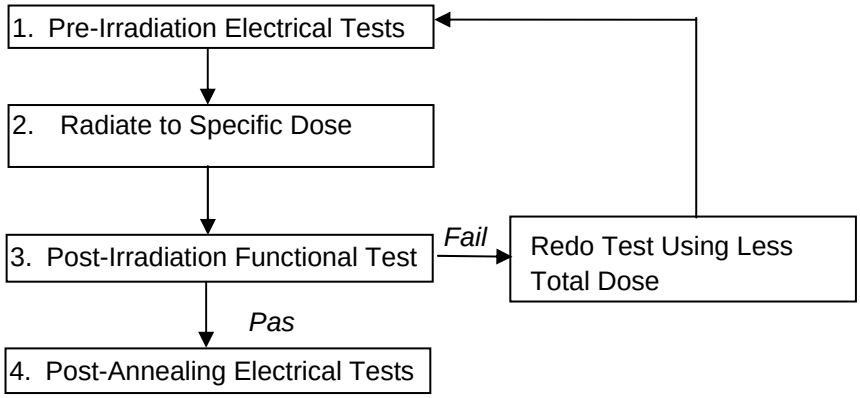


Figure 1 Parametric test flow chart

The test method generally follows the guidelines in the military standard TM1019.8. Figure 1 is the flow chart describing the steps for functional and parametric tests, irradiation, and post-irradiation annealing.

The accelerated aging, or rebound test mentioned in TM1019.8 is unnecessary because there is no adverse time-dependent effect (TDE) in Microsemi SoC Products Group products manufactured by deep sub-micron CMOS technologies. Elevated temperature annealing basically reduces the effects originating from radiation-induced leakage currents. As indicated by test data in the following sections, the predominant radiation effects in RTAX2000S are due to radiation-induced leakage currents.

Room temperature annealing is performed in this test; the duration is approximately 7 days.

C. Design and Parametric Measurements

The DUT uses a high utilization, generic design (TOP_AX2000S_TID) to evaluate total dose effects for typical space applications. Appendix B contains the schematics and Verilog files of this design.

Table 2 lists measured electrical parameters 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, using 3.3 VDC, is worse than LVDS, which uses 2.5 VDC. 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 and YQ0. 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 captures.

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 (EN8/YQ0, DA/QA0)
5. Propagation Delay	String of buffers (CLOCK to O_BS)
6. Transition Characteristic	String of buffers output (O_BS)

III. Test Results

A. Functionality

Every DUT passed the pre-irradiation and post-annealing functional tests. The as-irradiated DUT is functionally tested on the output (O_FF_HCLKA) of the largest shift register.

#

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

Figures 2-6 plots the in-flux standby I_{CCA} and I_{CCI} versus total dose for each DUT. Table 3 summarizes the pre-irradiation, post-irradiation and post-annealing I_{CC}. The post-annealing I_{CC} for four different bit patterns, all '0', all '1', checkerboard and inverted-checkerboard, in the RAM are basically the same.

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

DUT	Total Dose krad(SiO_2)	I_{CCA} (mA)			I_{CCI} (mA)		
		Pre-irrad	Post-irrad	Post-ann	Pre-irrad	Post-irrad	Post-ann
4575	300krad	18	145	26	27	111	92
4638	300krad	13	76	20	28	157	63
4648	300krad	11	119	13	28	119	57
4678	200krad	16	28	18	28	84	62
4729	200krad	16	23	14	28	64	43
4822	200krad	14	22	16	30	61	52

In compliance with TM1019.8 subsection 3.11.2.c, the post-irradiation-parametric limit (PIPL) for the post-annealing I_{CCI} in this test is defined as the addition of highest I_{CCI}, I_{CCDA} and $I_{CCDIFFA}$ values in Table 2-4 of the RTAXS spec sheet:

http://www.actel.com/documents/RTAXS_DS.pdf

For I_{CCA}, the PIPL is 500 mA; the PIPL of I_{CCI} equals to 35+10+3.13×7 = 66.91(mA). Note that there are 7 pairs of differential LVPECL inputs in each DUT. Based on these PIPL, post-annealed DUT passes both the I_{CCA} and I_{CCI} spec for 200 krad(SiO_2).

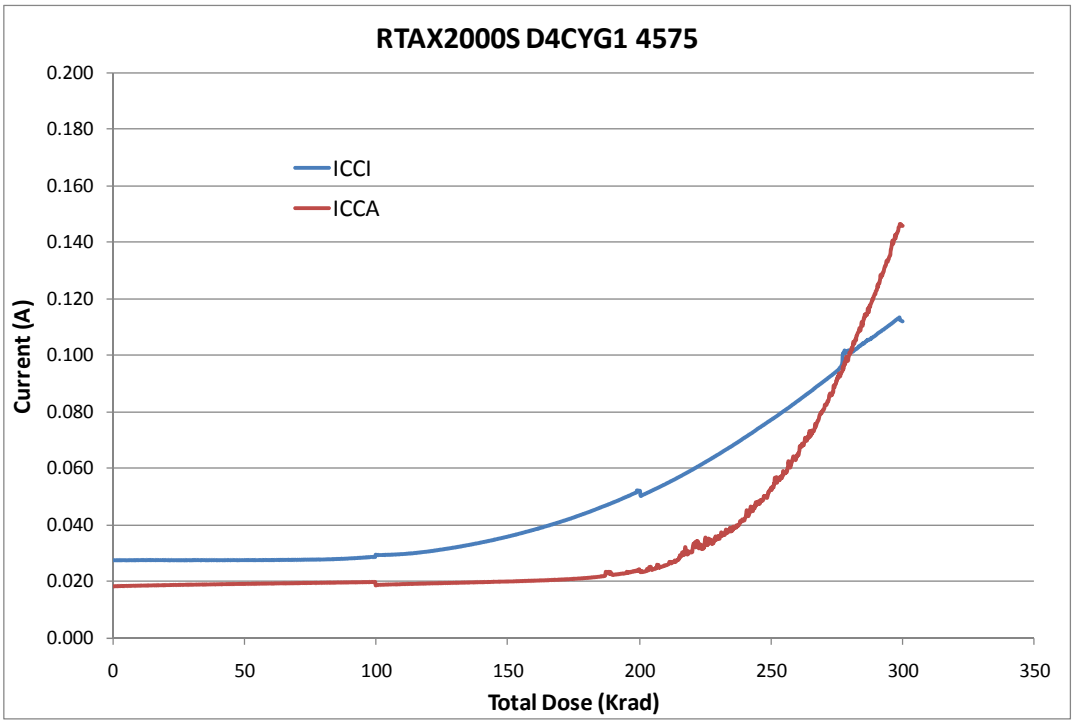


Figure 2 DUT 4575 in-flux I_{CCA} and I_{CCI}.

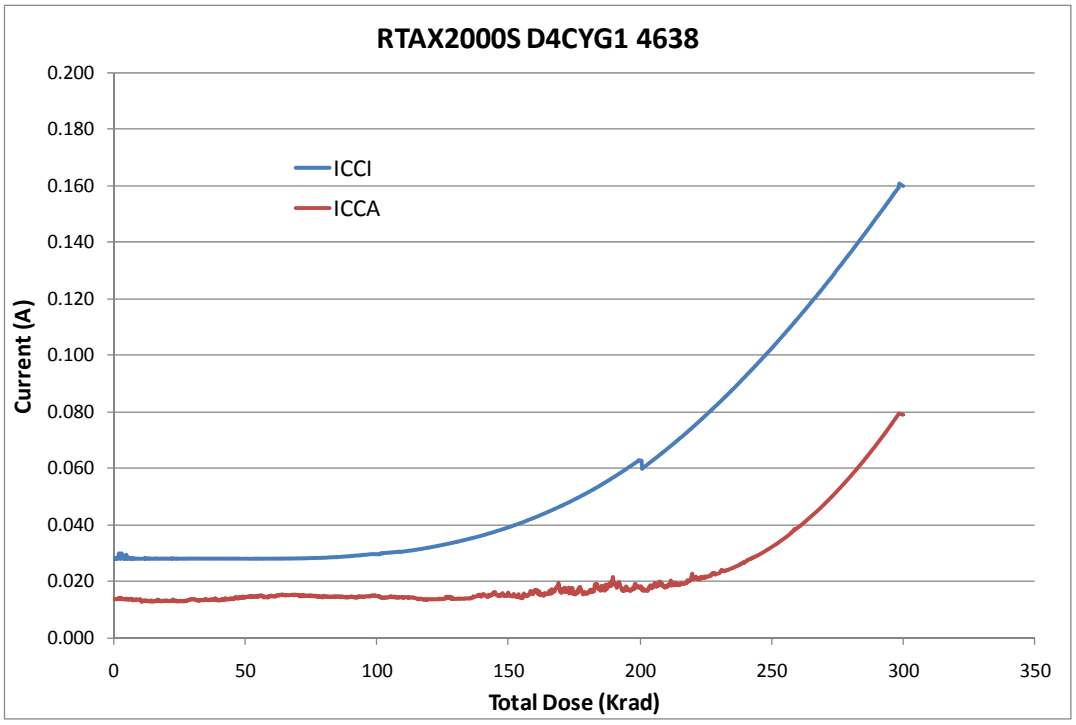


Figure 3 DUT 4638 in-flux I_{CCA} and I_{CCI}.

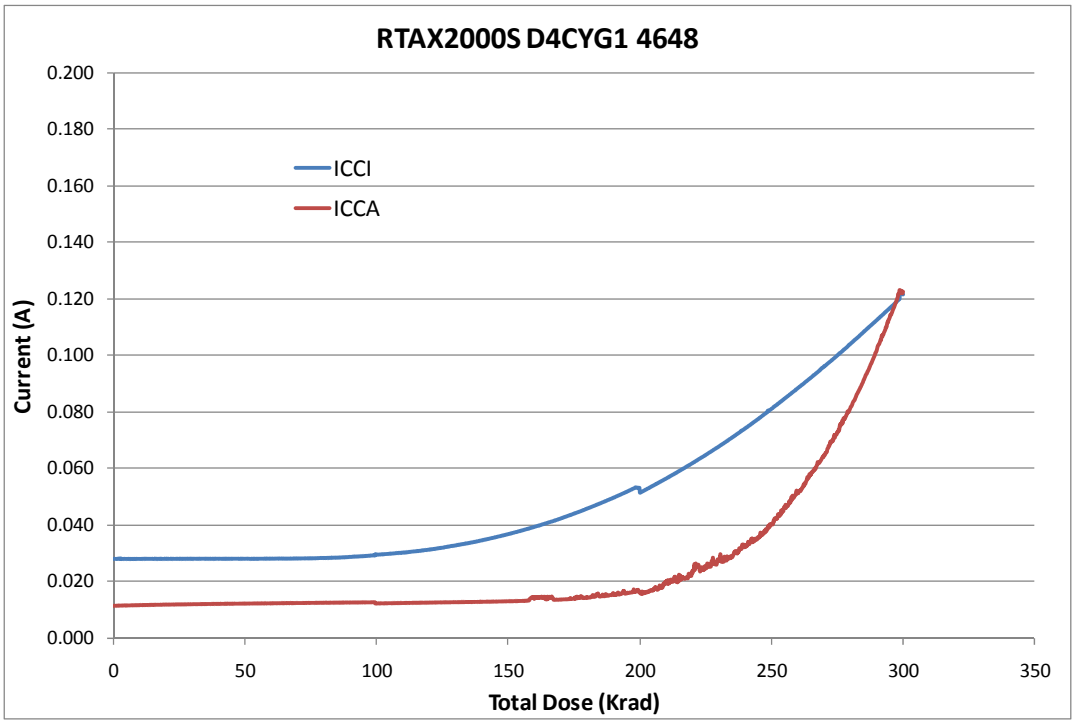


Figure 4 DUT 4648 in-flux I_{CCA} and I_{CCI}.

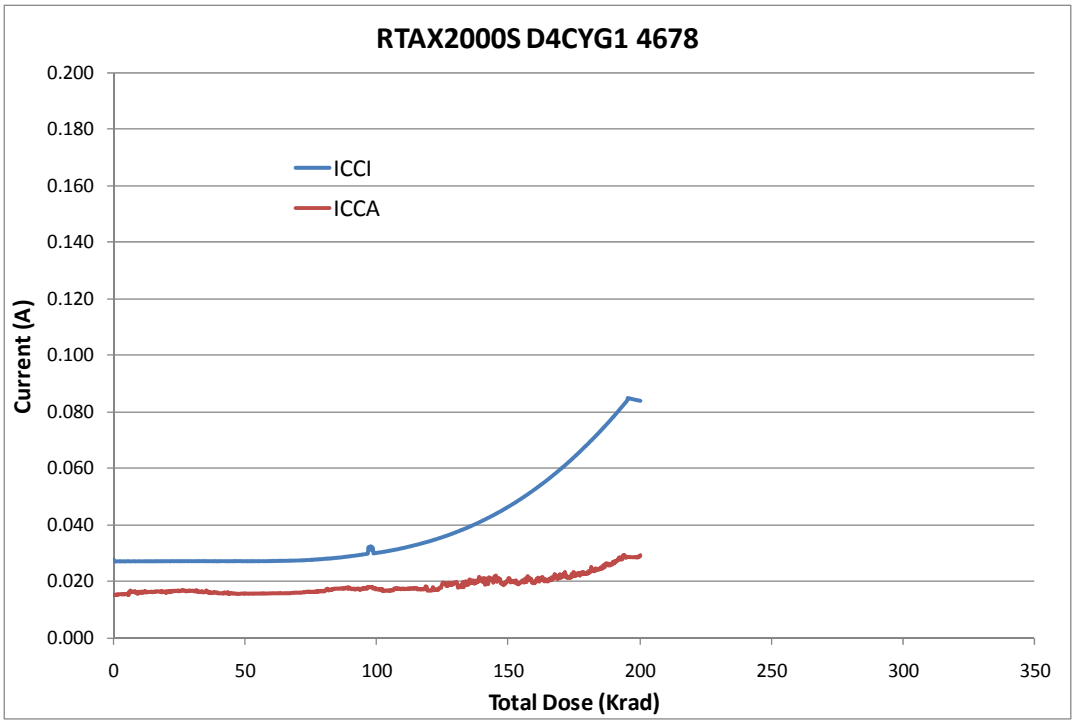


Figure 5 DUT 4678 in-flux I_{CCA} and I_{CCI}.

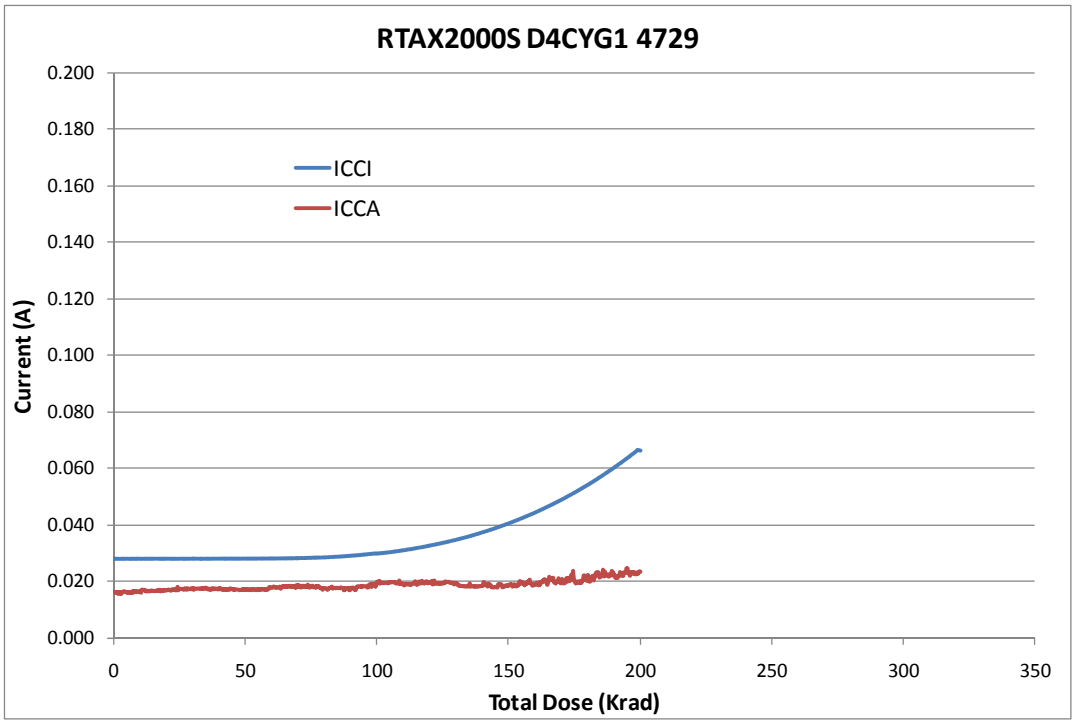


Figure 6 DUT 4729 in-flux I_{CCA} and I_{CCI}.

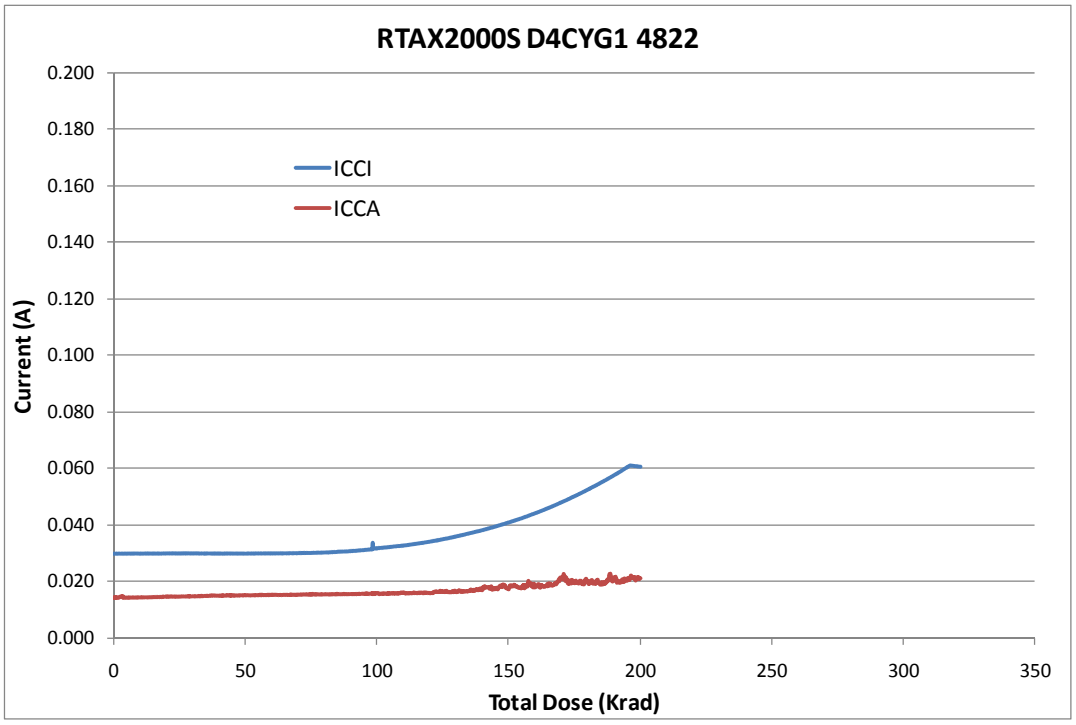


Figure 7 DUT 4822 in-flux I_{CCA} and I_{CCI}.

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. The post-annealing shift in every case is very small.

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

DUT	4575 (300krad)				4638 (300krad)			
Input Pin	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
	V_{IL} (mV)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
DA	1535	1525	1515	1505	1530	1520	1515	1505
EN8	1540	1505	1495	1495	1510	1535	1500	1510
IO_I6	1535	1520	1500	1495	1530	1525	1510	1495
IO_I5	1525	1525	1500	1500	1530	1515	1510	1510
IO_I4	1545	1515	1530	1525	1530	1515	1535	1505
IO_I3	1505	1480	1565	1510	1490	1495	1570	1505
IO_I2	1530	1530	1525	1540	1515	1510	1545	1520
IO_I1	1525	1530	1545	1530	1535	1525	1505	1535

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

DUT	4648 (300krad)				4678 (200krad)			
Input Pin	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
	V_{IL} (mV)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
DA	1535	1520	1515	1500	1535	1520	1515	1500
EN8	1510	1535	1515	1510	1540	1535	1495	1510
IO_I6	1530	1525	1500	1495	1535	1520	1500	1495
IO_I5	1515	1515	1510	1495	1525	1515	1500	1495
IO_I4	1525	1515	1535	1500	1515	1515	1505	1505
IO_I3	1490	1480	1565	1500	1490	1490	1565	1500
IO_I2	1530	1510	1545	1510	1515	1510	1520	1510
IO_I1	1530	1520	1505	1530	1535	1530	1545	1530

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

DUT	4729 (200krad)				4822 (200krad)			
Input Pin	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
	V_{IL} (mV)		V_{IH} (V)		V_{IL} (V)		V_{IH} (V)	
DA	1530	1520	1510	1505	1530	1520	1505	1505
EN8	1515	1535	1495	1490	1540	1530	1495	1490
IO_I6	1535	1525	1500	1495	1535	1525	1500	1495
IO_I5	1525	1525	1490	1485	1525	1510	1500	1495
IO_I4	1520	1515	1530	1525	1540	1515	1530	1500
IO_I3	1490	1490	1565	1510	1490	1480	1565	1500
IO_I2	1515	1510	1535	1535	1515	1510	1525	1495
IO_I1	1520	1530	1535	1535	1530	1515	1545	1530

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

Table 5 lists the LVPECL differential input threshold voltage changes due to irradiations. All pins show negligible changes, and all the data are within the specification.

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

DUT	4575 (300krad)				4638 (300krad)			
Input Pin	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
	V_{IL} (mV)		V_{IH} (mV)		V_{IL} (mV)		V_{IH} (mV)	
DIO_I7P	1795	1795	1785	1785	1800	1795	1795	1790
DIO_I6P	1780	1775	1780	1795	1780	1780	1795	1780
DIO_I5P	1785	1790	1770	1775	1790	1780	1770	1760
DIO_I4P	1765	1780	1770	1780	1800	1790	1770	1785
DIO_I3P	1785	1800	1780	1785	1785	1790	1780	1775
DIO_I2P	1795	1790	1785	1780	1790	1790	1780	1785
DIO_I1P	1770	1790	1770	1770	1775	1790	1765	1770

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

DUT	4648 (300krad)				4678 (200krad)			
Input Pin	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
	V_{IL} (mV)		V_{IH} (mV)		V_{IL} (mV)		V_{IH} (mV)	
DIO_I7P	1795	1795	1780	1785	1785	1795	1780	1780
DIO_I6P	1780	1780	1790	1780	1790	1775	1770	1770
DIO_I5P	1785	1795	1775	1780	1790	1785	1775	1775
DIO_I4P	1790	1785	1770	1785	1785	1770	1770	1785
DIO_I3P	1800	1785	1785	1775	1785	1790	1775	1775
DIO_I2P	1790	1785	1780	1785	1775	1785	1780	1780
DIO_I1P	1775	1780	1765	1780	1785	1790	1780	1780

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

DUT	4729 (200krad)				4822 (200krad)			
Input Pin	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann	Pre-Irrad	Post-Ann
	V_{IL} (mV)		V_{IH} (mV)		V_{IL} (mV)		V_{IH} (mV)	
DIO_I7P	1795	1805	1785	1790	1795	1795	1785	1785
DIO_I6P	1810	1780	1795	1795	1780	1805	1780	1780
DIO_I5P	1785	1785	1775	1770	1800	1790	1775	1775
DIO_I4P	1785	1790	1775	1780	1765	1770	1770	1785
DIO_I3P	1775	1790	1770	1775	1785	1785	1780	1770
DIO_I2P	1775	1785	1770	1780	1785	1785	1780	1780
DIO_I1P	1780	1785	1770	1770	1790	1790	1770	1765

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.

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

Sourcing Current	Pin\DUT	4575 (300krad)		4638 (300krad)		4648 (300krad)		4678 (200krad)		4729 (200krad)		4822 (200krad)	
		Pre-rad	Post-an	Pre-rad	Post-an	Pre-rad	Post-an	Pre-rad	Post-an	Pre-rad	Post-an	Pre-rad	Post-an
1mA	QA0	6	6	7	6	7	7	7	6	6	6	6	6
	YQ0	7	6	7	6	7	5	7	6	7	6	7	5
8mA	QA0	84	80	84	81	85	82	86	82	84	80	85	81
	YQ0	94	89	94	90	92	88	95	90	93	88	93	88
20mA	QA0	141	134	140	136	142	138	144	137	141	134	142	136
	YQ0	157	149	158	152	154	148	159	151	156	149	156	148
50mA	QA0	357	338	356	343	361	347	367	348	357	339	361	344
	YQ0	398	379	401	386	390	376	402	385	395	378	394	376
100mA	QA0	750	710	750	719	759	727	770	731	750	711	758	720
	YQ0	831	791	839	807	816	784	840	805	825	789	823	787

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

Sourcing Current	Pin\DUT	4575 (300krad)		4638 (300krad)		4648 (300krad)		4678 (200krad)		4729 (200krad)		4822 (200krad)	
		Pre-rad	Post-an	Pre-rad	Post-an	Pre-rad	Post-an	Pre-rad	Post-an	Pre-rad	Post-an	Pre-rad	Post-an
1mA	QA0	3289	3283	3289	3283	3289	3283	3289	3283	3289	3283	3289	3283
	YQ0	3288	3280	3288	3280	3288	3280	3288	3280	3288	3285	3288	3285
8mA	QA0	3224	3218	3224	3218	3224	3218	3224	3223	3224	3223	3224	3223
	YQ0	3218	3210	3218	3210	3218	3215	3218	3215	3218	3215	3218	3215
20mA	QA0	3114	3108	3114	3113	3109	3103	3109	3108	3114	3113	3114	3108
	YQ0	3098	3090	3098	3090	3098	3095	3103	3095	3103	3095	3103	3095
50mA	QA0	2818	2818	2823	2823	2813	2818	2818	2818	2823	2823	2818	2818
	YQ0	2783	2780	2788	2780	2788	2785	2793	2785	2793	2785	2793	2790
100mA	QA0	2253	2252	2258	2257	2243	2242	2248	2252	2258	2262	2248	2252
	YQ0	2184	2175	2189	2180	2194	2190	2194	2190	2204	2200	2209	2200

F. Propagation Delay

The propagation delay was measured in-situ, post-irradiation, and post-annealing. The irradiation was temporarily stopped at each total-dose increment of 100 krad for the measurement. Each measurement has a 2-minutes wait after a DUT is removed from the chamber. The results are plotted in Figure 8, and listed in Table 8. As shown in Figure 8, the propagation delay initially decreases with the total dose, but the change is small throughout the irradiation. Referring to in-flux static current plots (Figure 2-7), a device probably heats up as the dose increases. The rising temperature could be the root cause of the increasing trend at high doses.

The post-annealing data, on the other hand, show decreased delay in every case.

The radiation delta in every case is well within the 10% degradation criterion. User can take the worst case for the design-margin consideration.

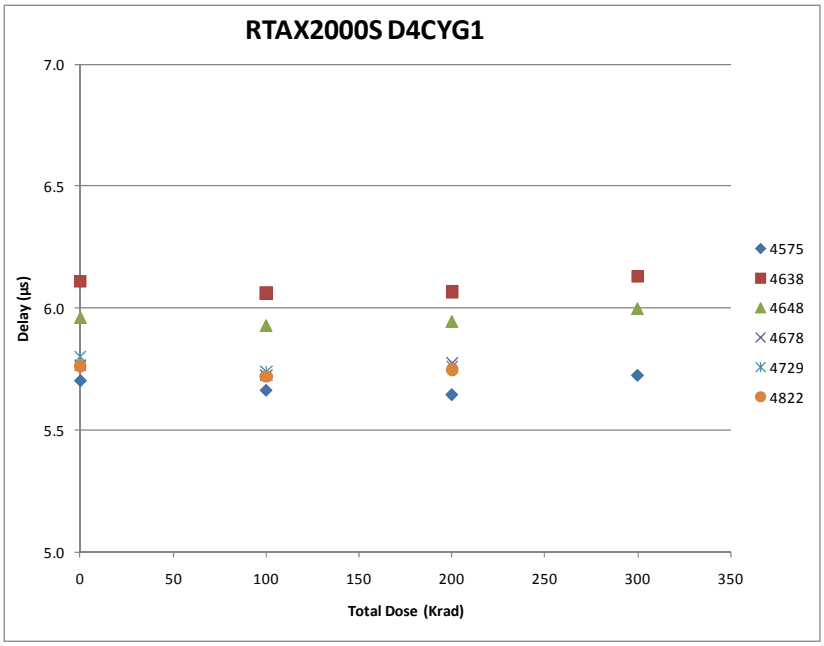


Figure 8 In-situ propagation delay versus total dose. The measurement is performed outside the irradiation chamber.

Table 8 Radiation-Induced Propagation-Delay Degradations

RTAX2000S D4CYG1						
Delay (µs)						
	DUT	Pre-rad	100 krad	200 krad	300 krad	Post-ann
	4575	5.7029	5.6636	5.6458	5.7246	5.6526
	4638	6.1087	6.0618	6.0670	6.1308	6.0520
	4648	5.9600	5.9272	5.9434	5.9966	5.8877
	4678	5.7663	5.7263	5.7725	-	5.7190
	4729	5.7994	5.7414	5.7620	-	5.7433
	4822	5.7636	5.7202	5.7479	-	5.7198
Radiation Δ (%)						
	DUT	Pre-rad	100 krad	200 krad	300 krad	Post-ann
	4575	-	-0.69%	-1.00%	0.38%	-0.88%
	4638	-	-0.77%	-0.68%	0.36%	-0.93%
	4648	-	-0.55%	-0.28%	0.61%	-1.21%
	4678	-	-0.69%	0.11%	-	-0.82%
	4729	-	-1.00%	-0.64%	-	-0.97%
	4822	-	-0.75%	-0.27%	-	-0.76%

G. Transition Characteristics

Figures 9 to 20 show the pre-irradiation and post-annealing transition edges. In each case, the radiation-induced transition-time degradation is insignificant.

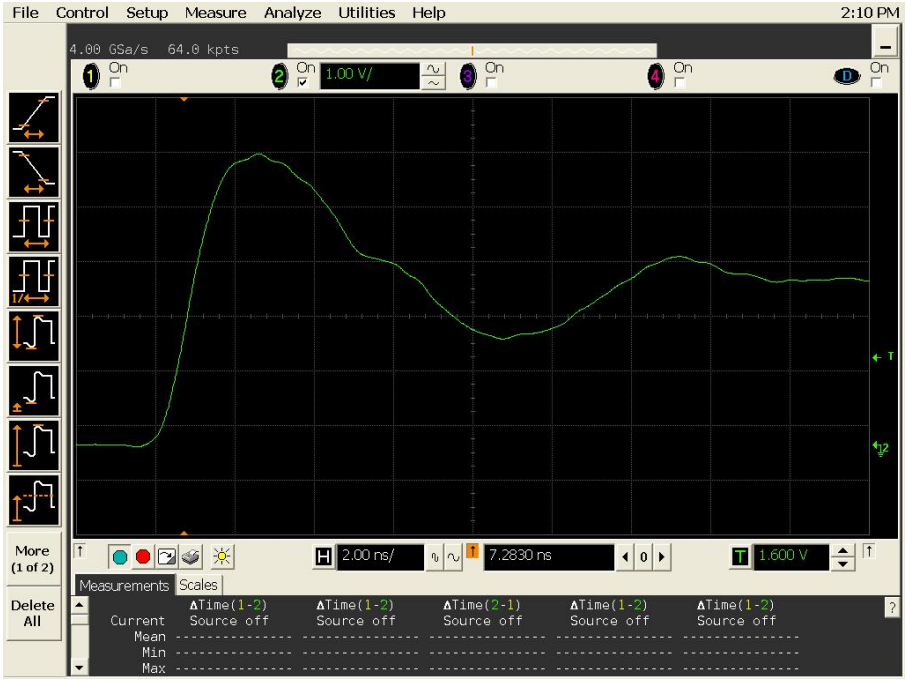


Figure 9(a) DUT 4575 pre-irradiation rising edge.

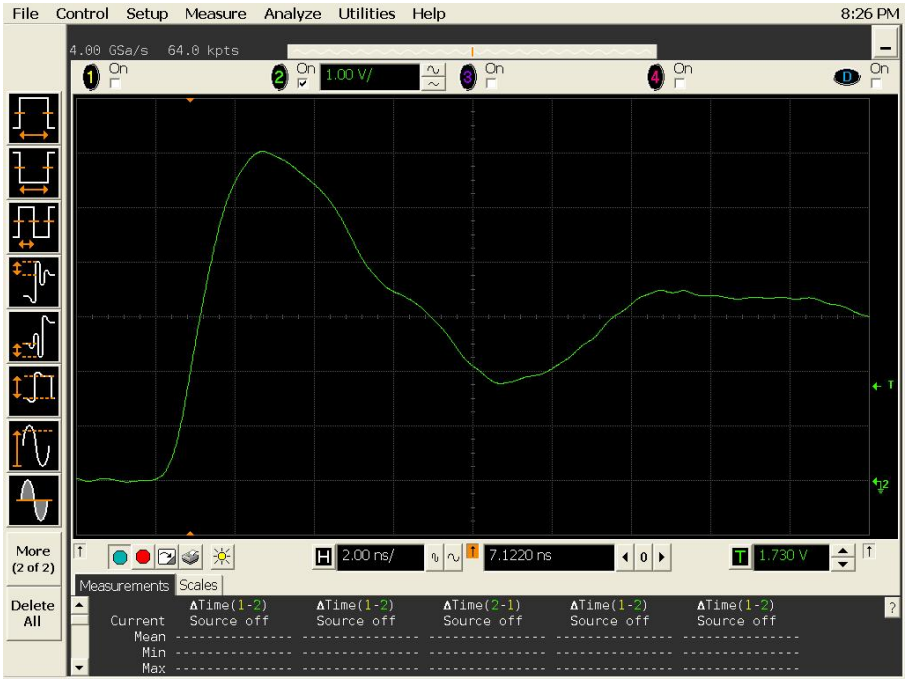


Figure 9(b) DUT 4575 post-annealing rising edge. Notice the time scale is different from pre-irradiation.

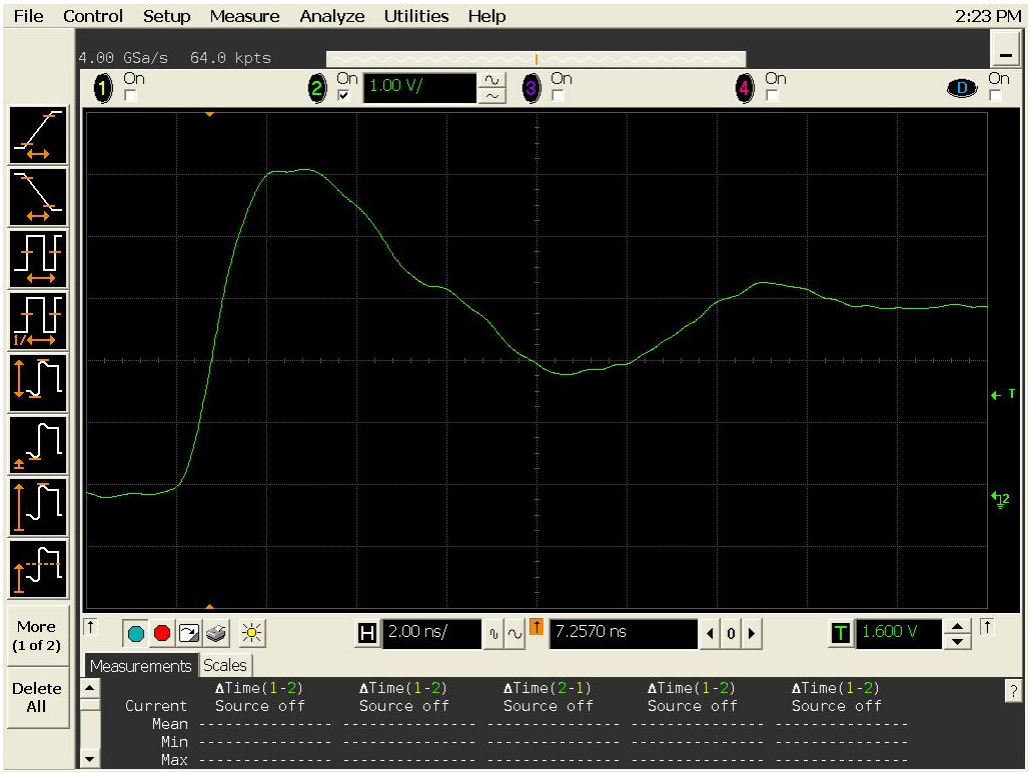


Figure 10(a) DUT 4638 pre-irradiation rising edge.

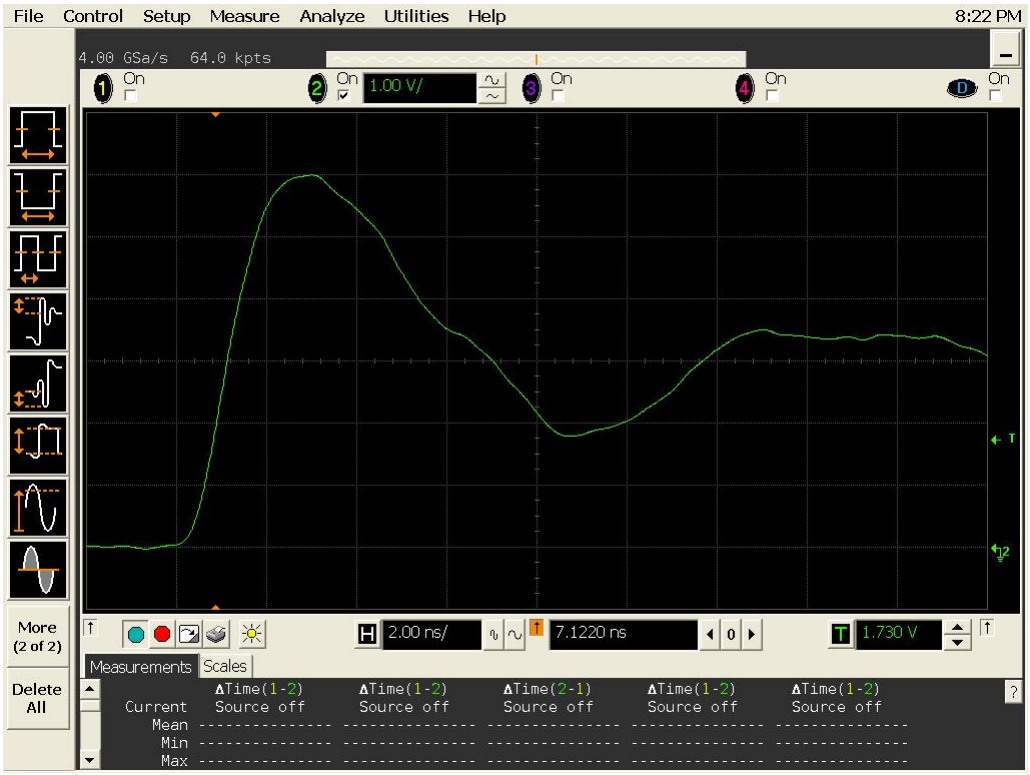


Figure 10(b) DUT 4638 post-annealing rising edge.

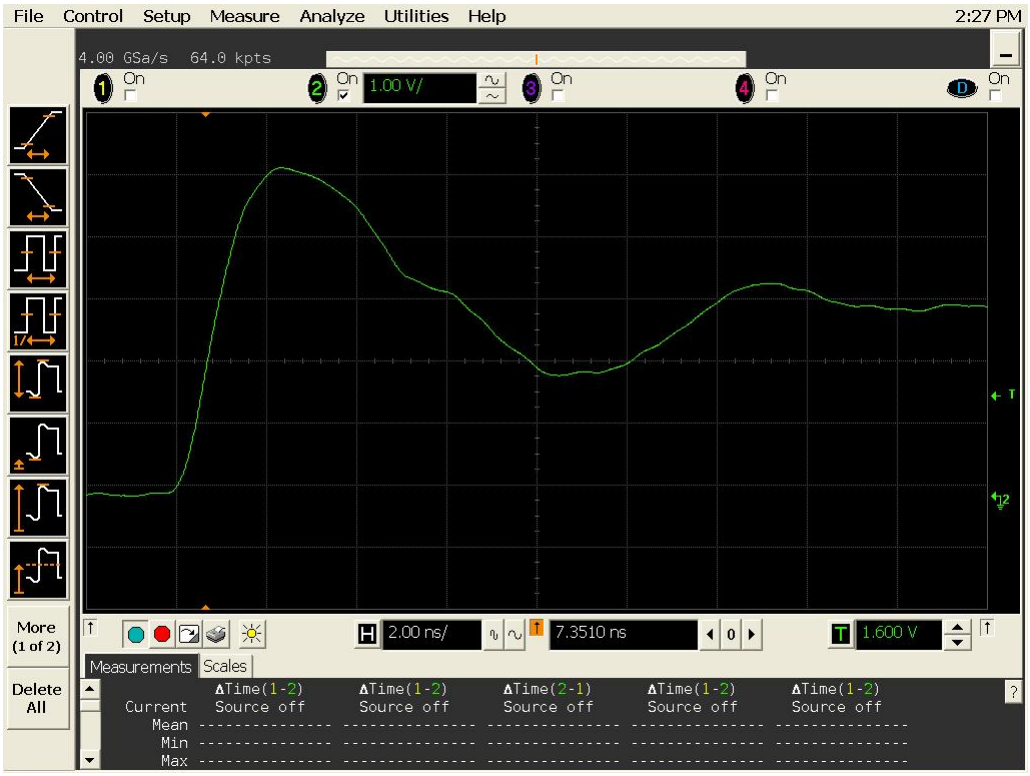


Figure 11(a) DUT 4648 pre-radiation rising edge.

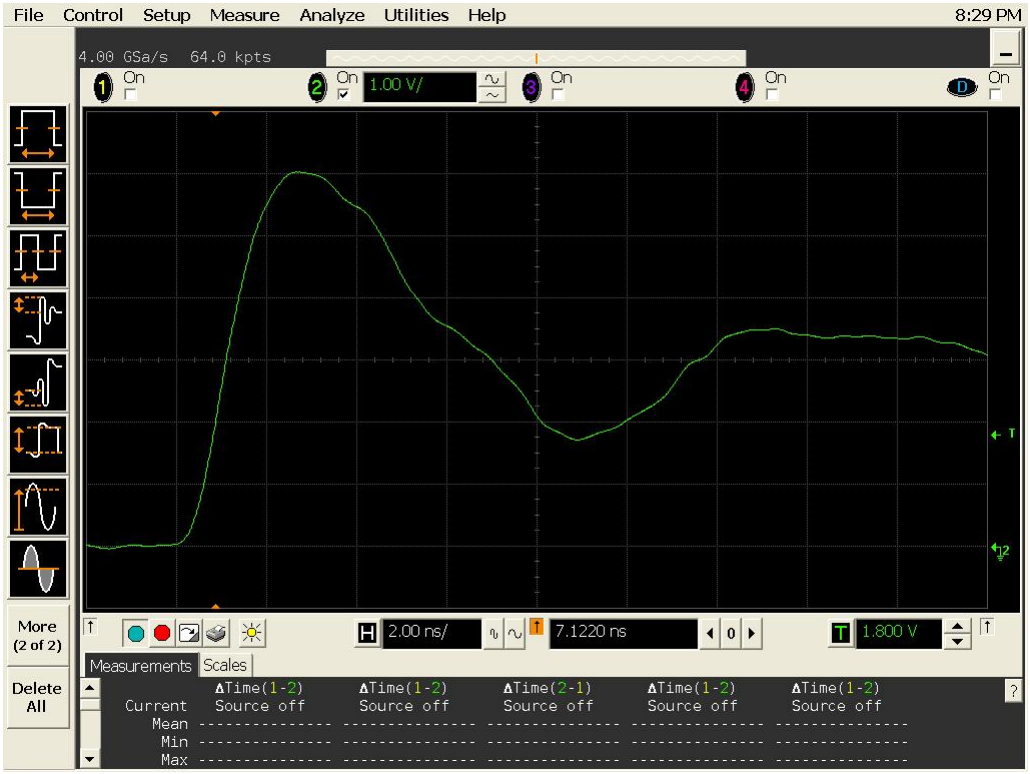


Figure 11(b) DUT 4648 post-annealing rising edge.

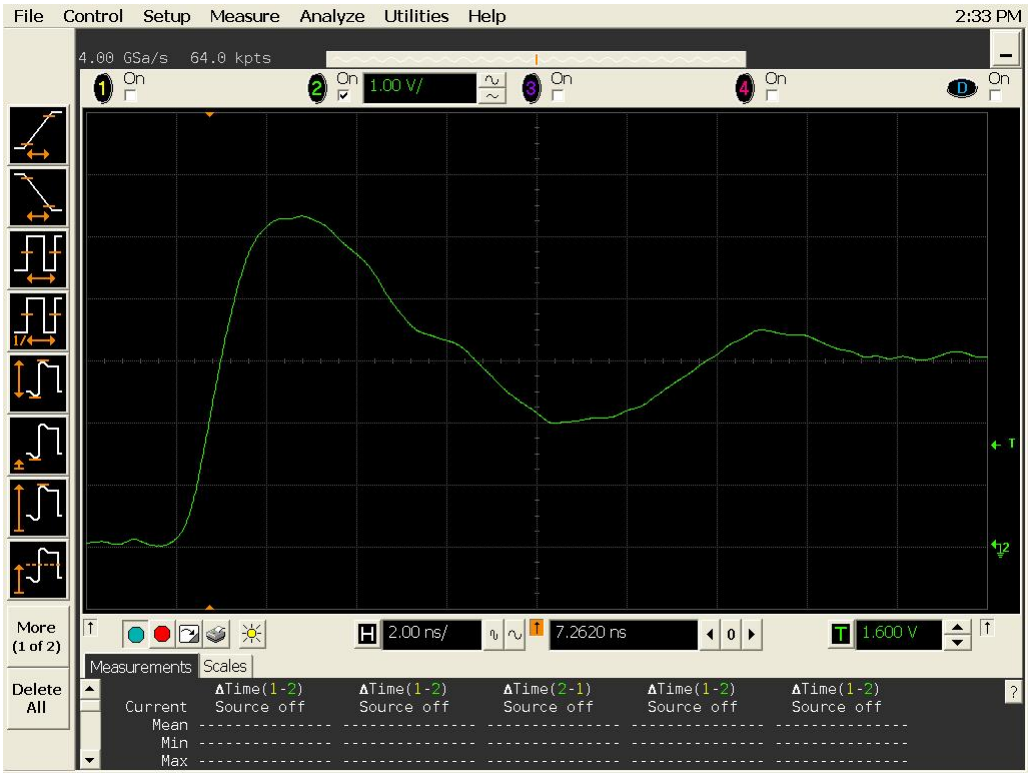


Figure 12(a) DUT 4678 pre-irradiation rising edge.

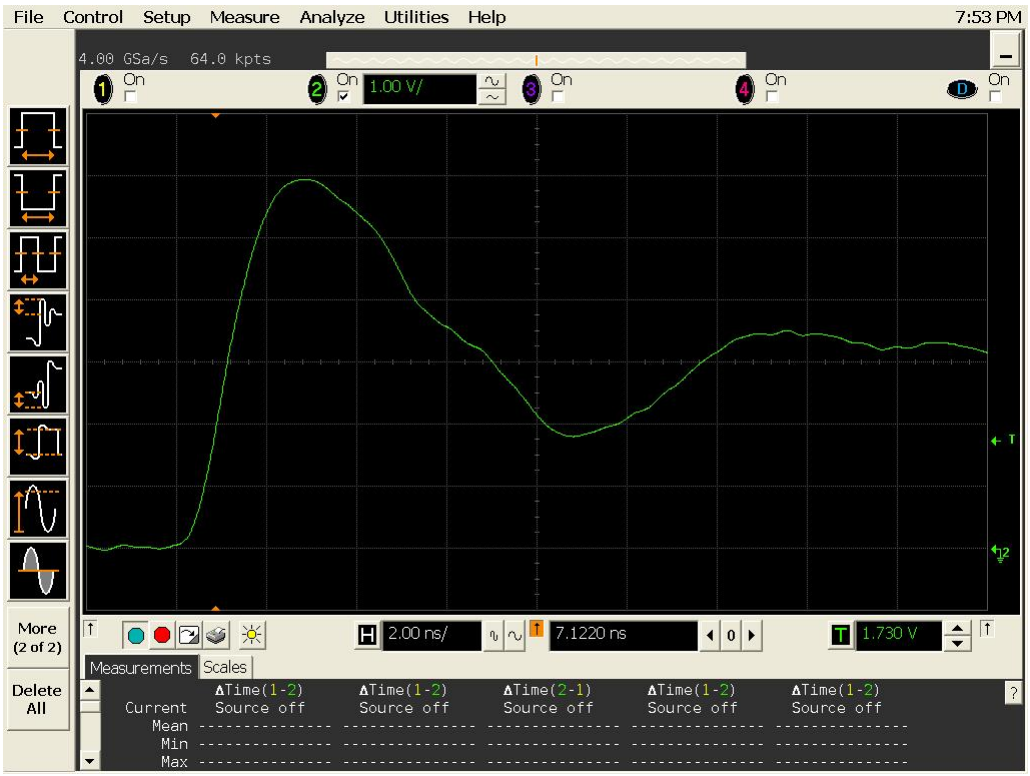


Figure 12(b) DUT 4678 post-annealing rising edge.

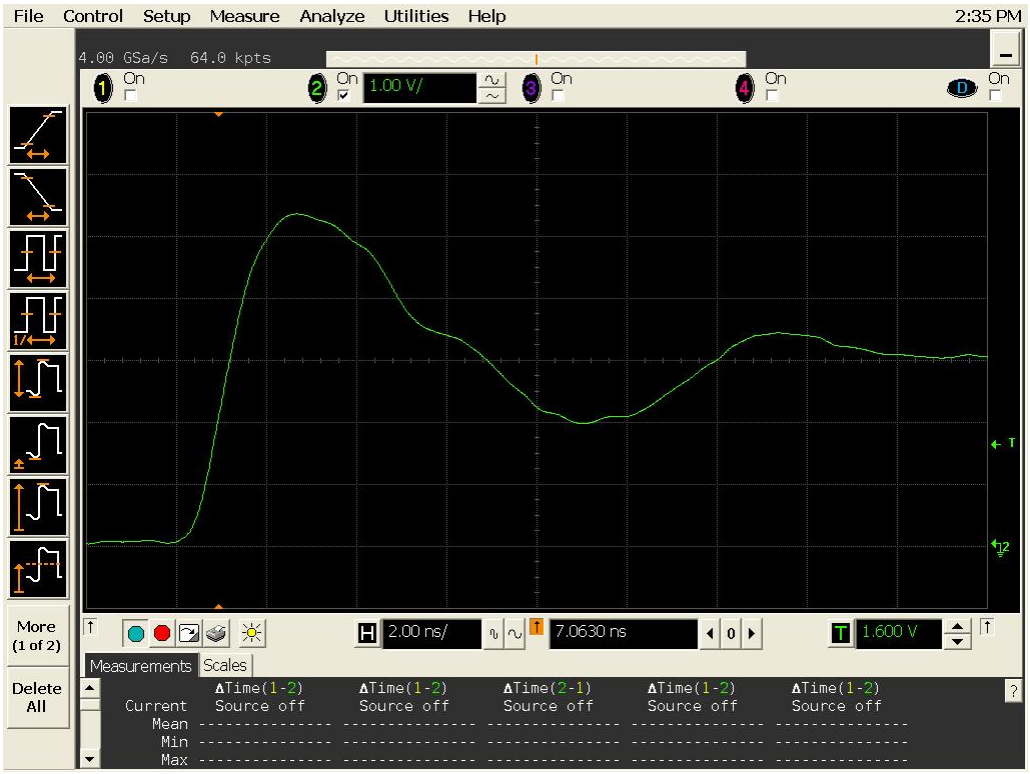


Figure 13(a) DUT 4729 pre-irradiation rising edge.

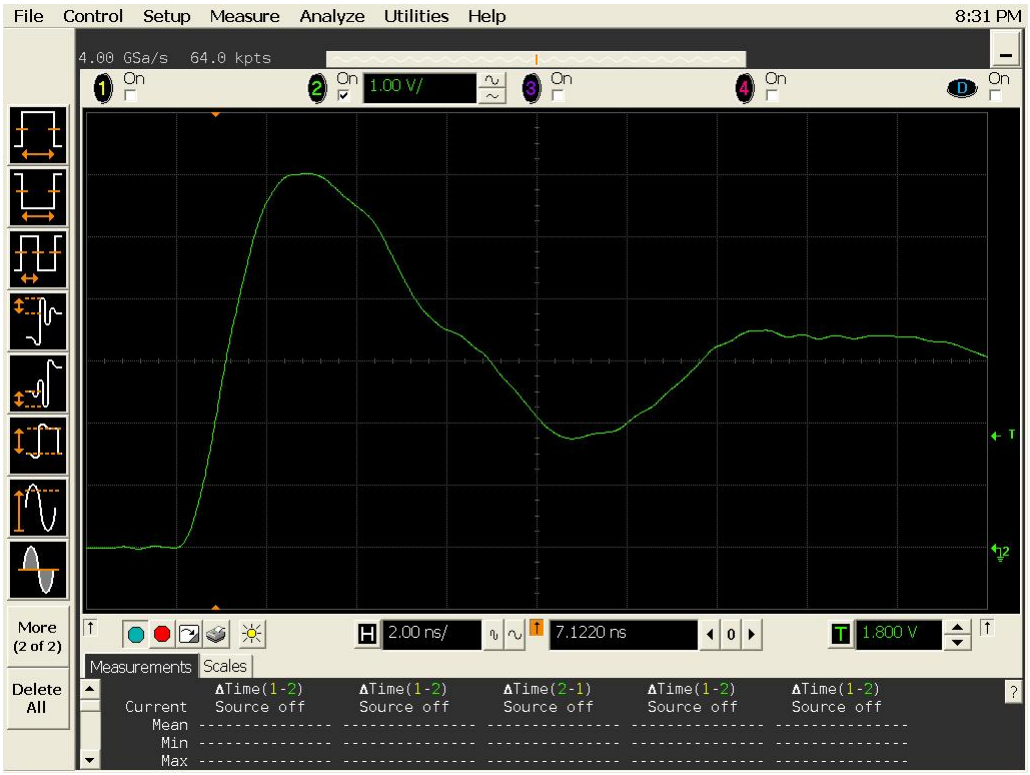


Figure 13(b) DUT 4729 post-annealing rising edge.

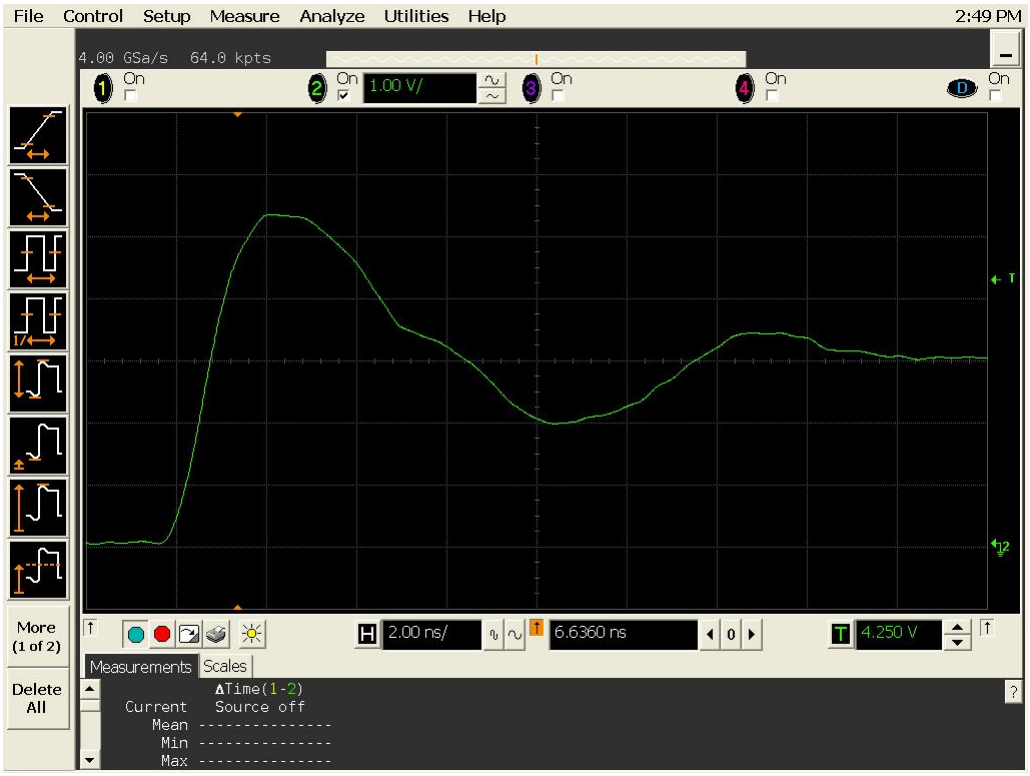


Figure 14(a) DUT 4822 pre-irradiation rising edge.

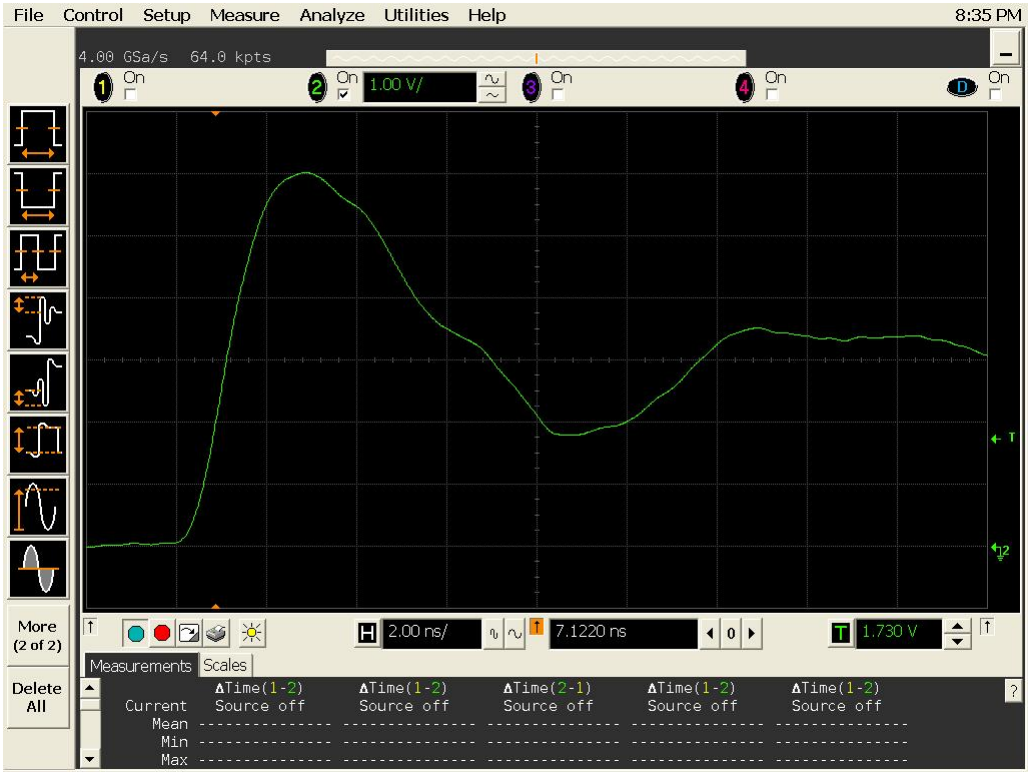


Figure 14(b) DUT 4822 post-annealing rising edge.

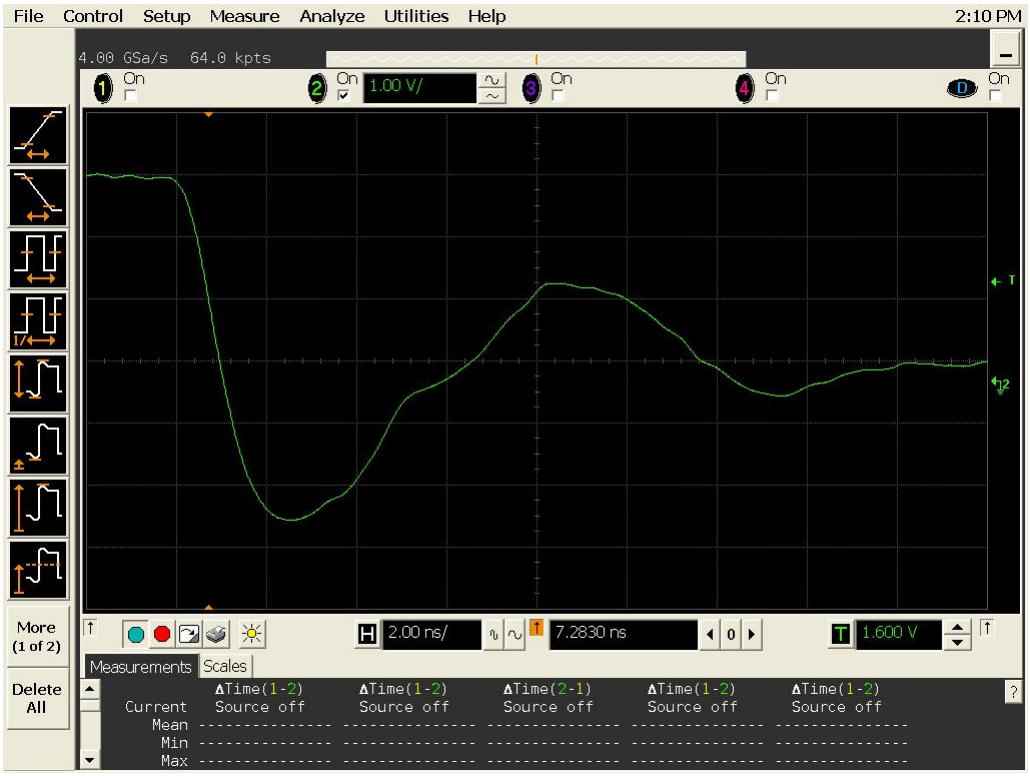


Figure 15(a) DUT 4575 pre-radiation falling edge.

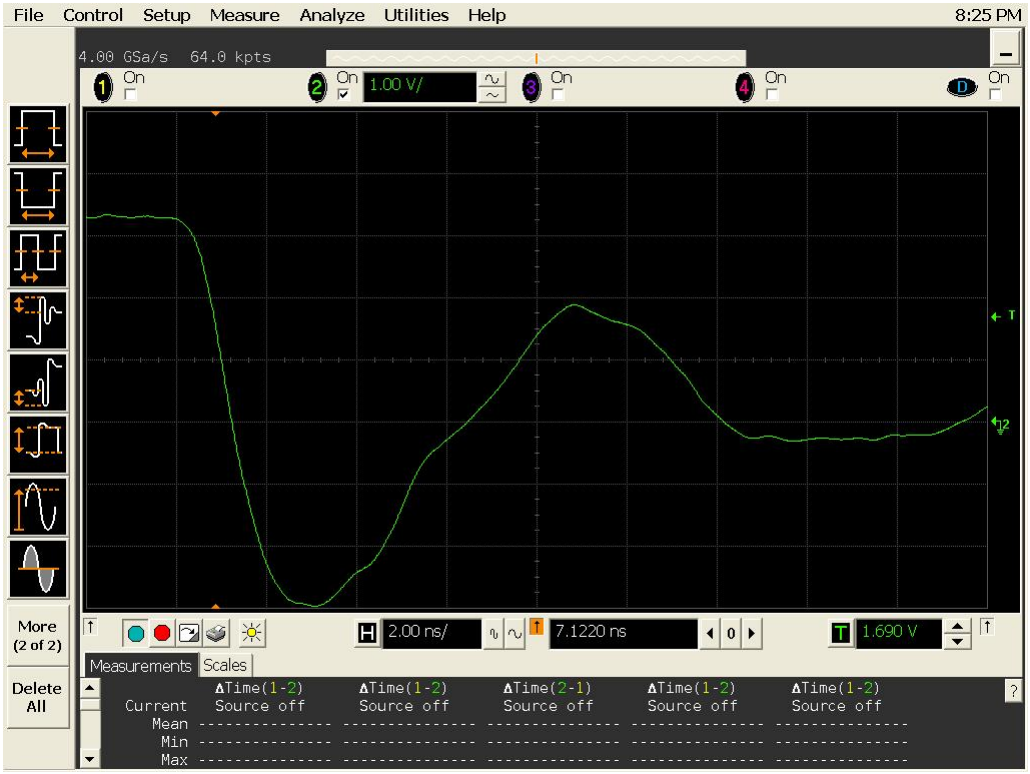


Figure 15(b) DUT 4575 post-annealing falling edge.



Figure 16(a) DUT 4638 pre-irradiation falling edge.

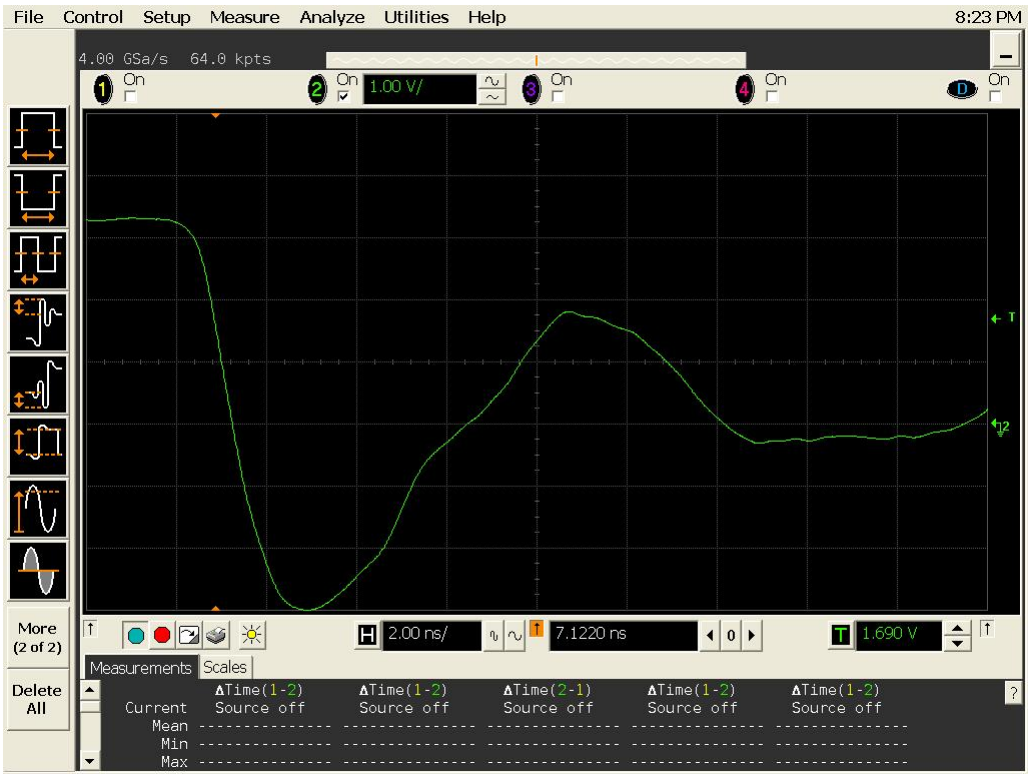


Figure 16(b) DUT 4638 post-annealing falling edge.

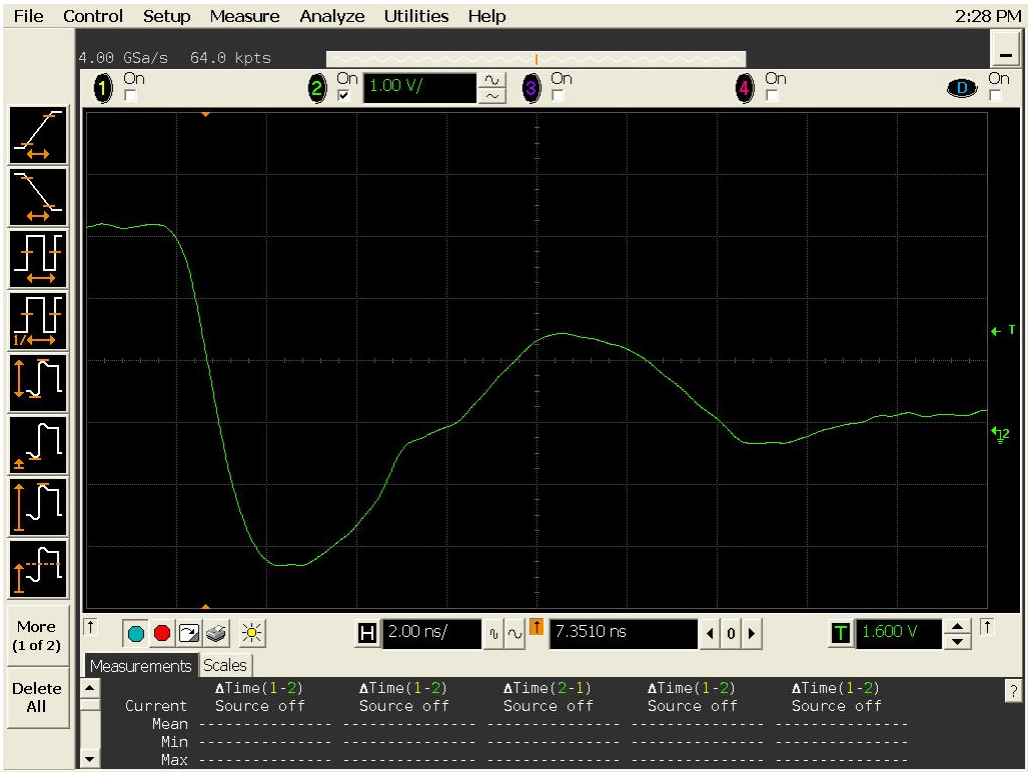


Figure 17(a) DUT 4648 pre-irradiation falling edge.



Figure 17(b) DUT 4648 post-annealing falling edge.

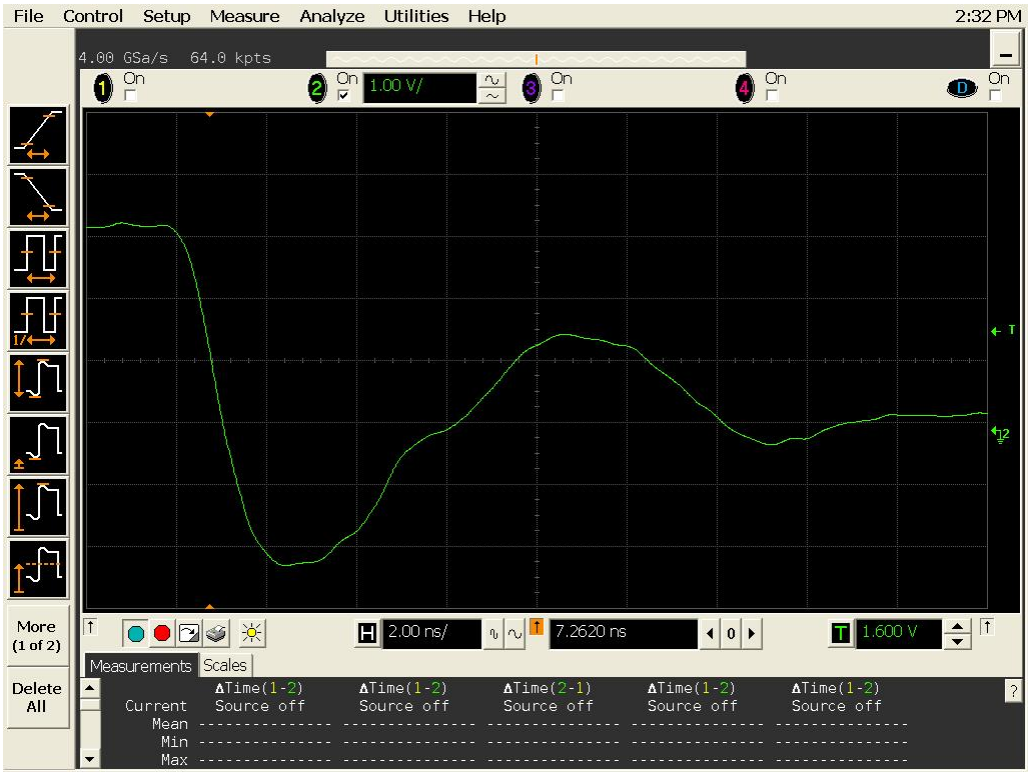


Figure 18(a) DUT 4678 pre-irradiation falling edge.



Figure 18(b) DUT 4678 post-annealing falling edge.

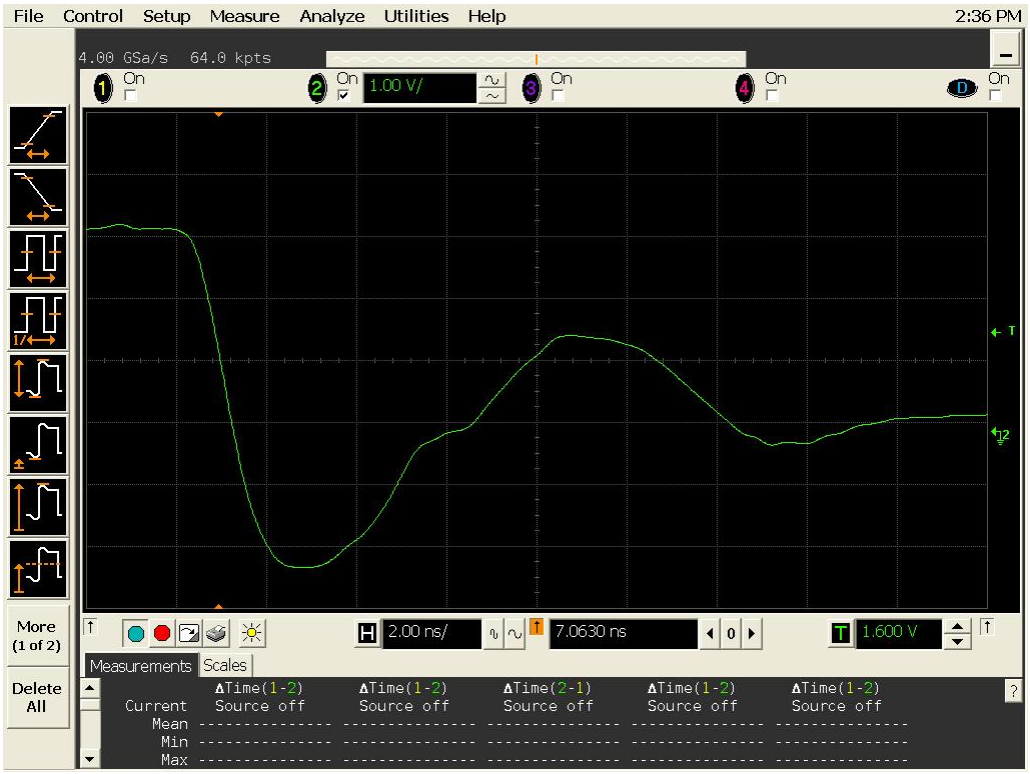


Figure 19(a) DUT 4729 pre-irradiation falling edge.



Figure 19(b) DUT 4729 post-annealing falling edge.

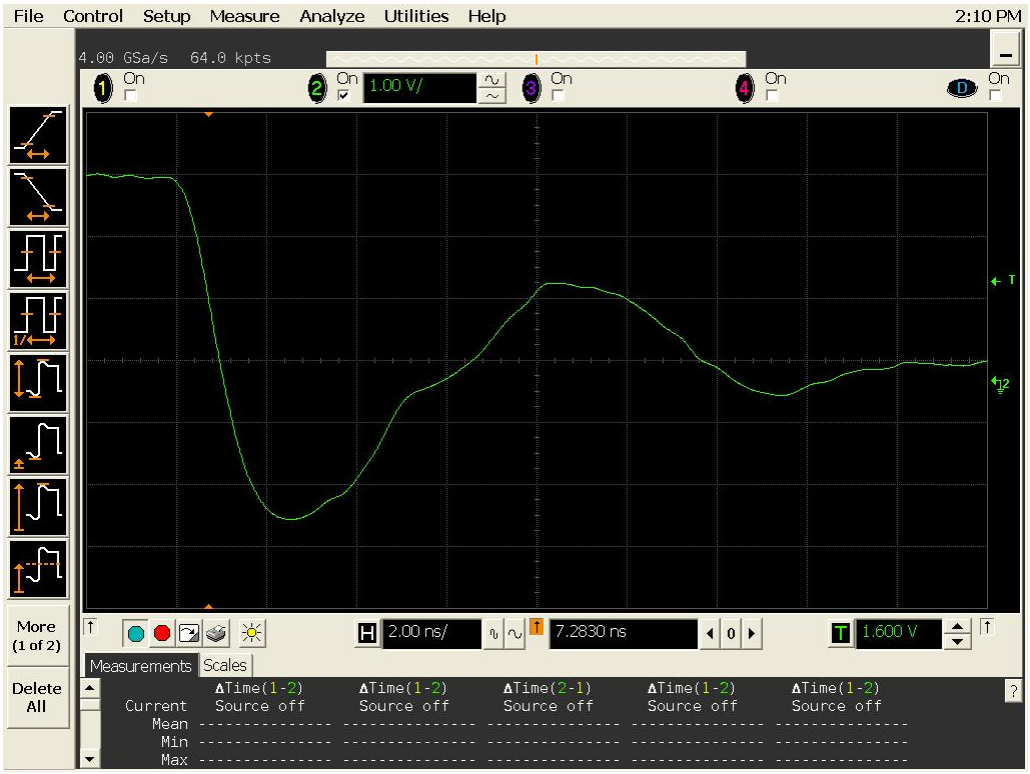


Figure 20(a) DUT 4822 pre-irradiation falling edge.

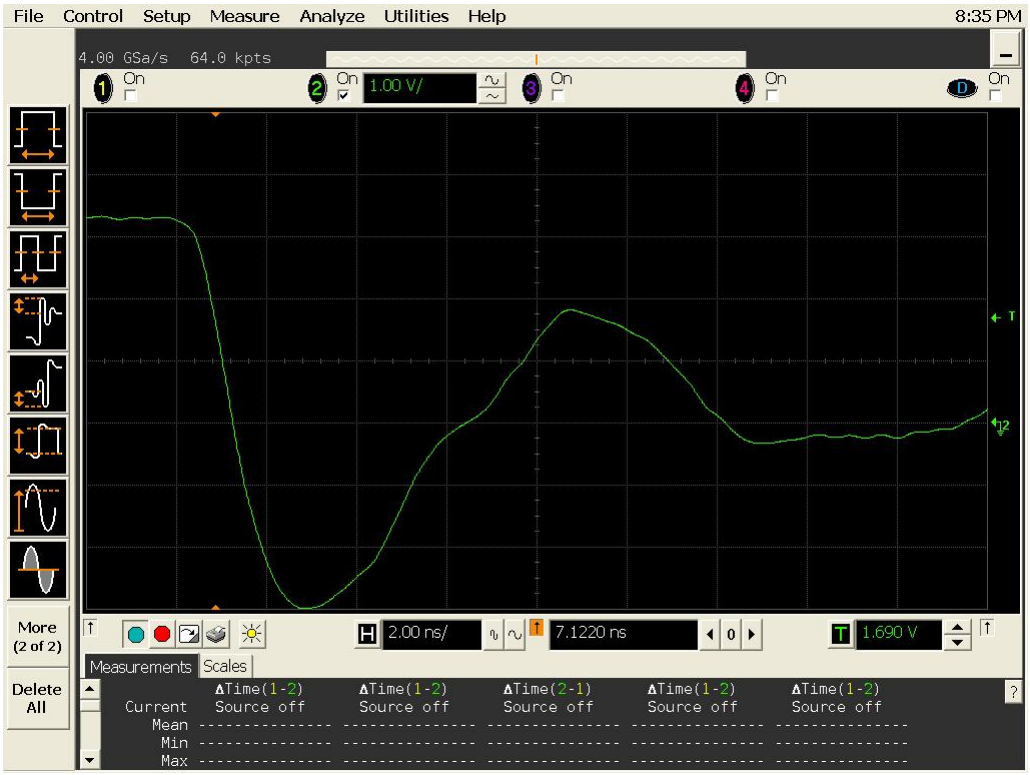


Figure 20(b) DUT 4822 post-annealing falling edge.

Appendix A DUT Bias

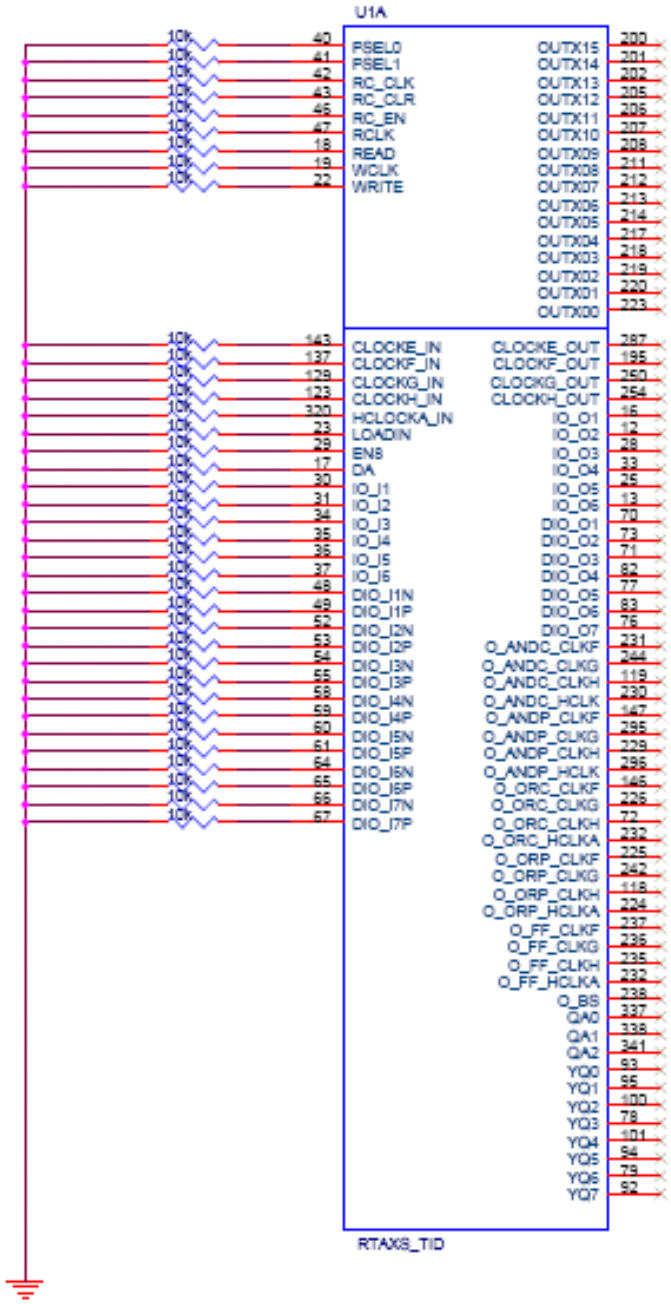


Figure A1 IO bias during irradiation

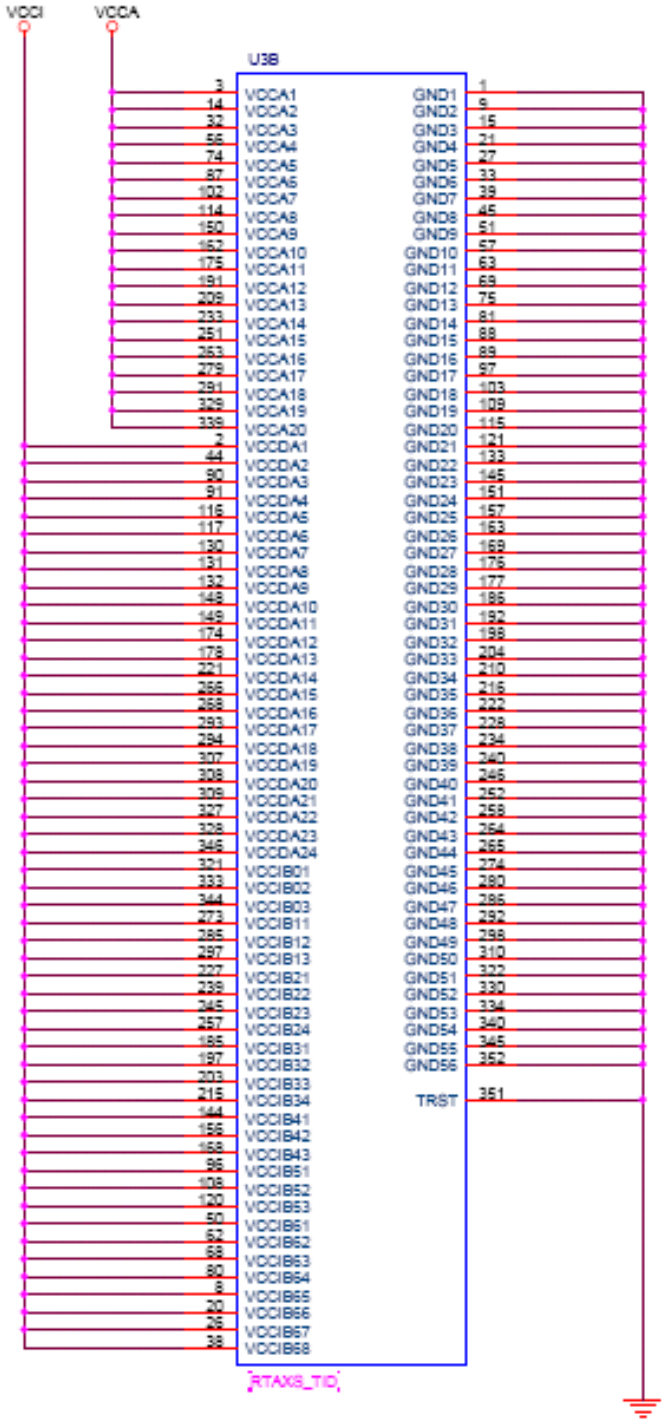
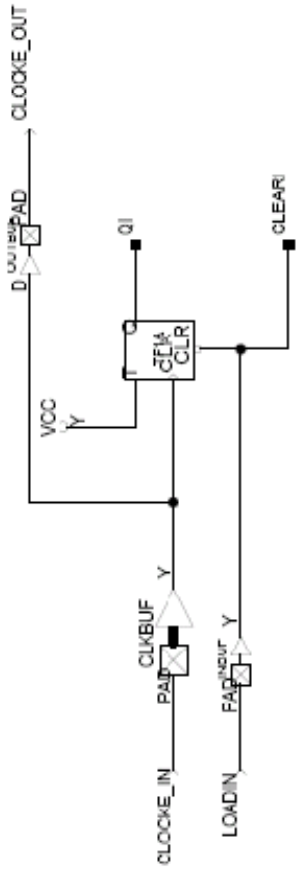
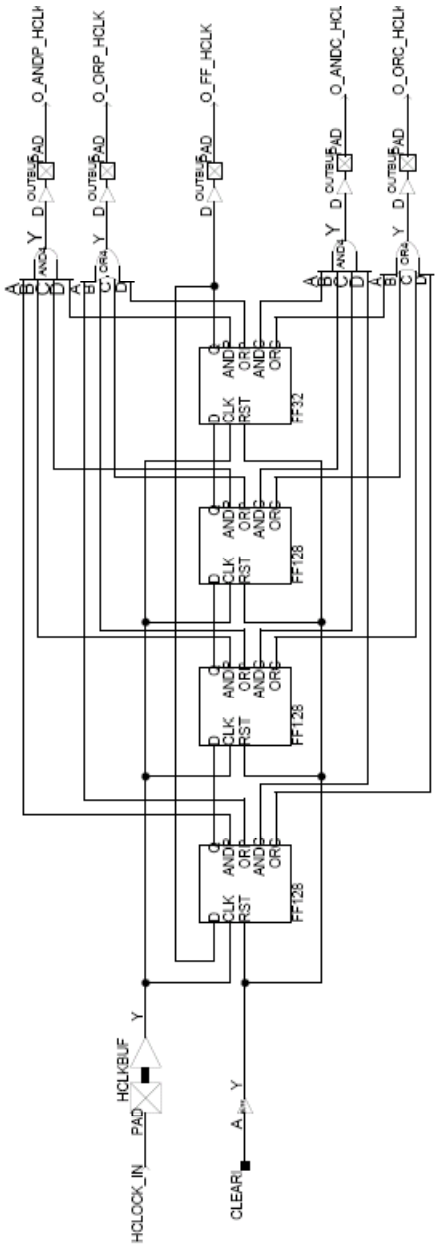
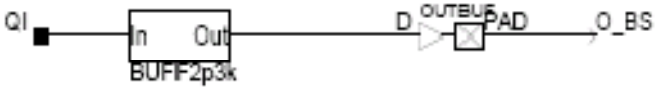


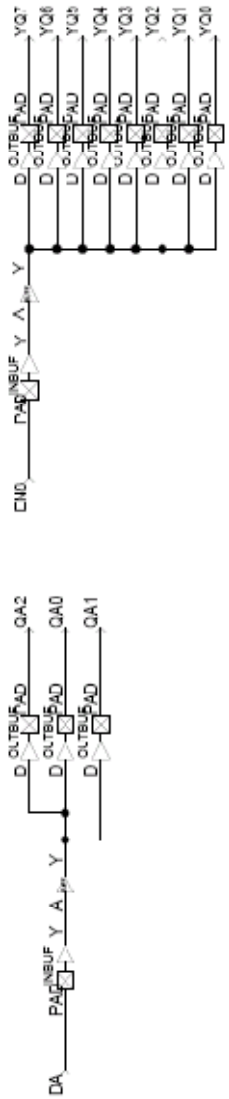
Figure A2 Power supply, ground and special pins bias during irradiation

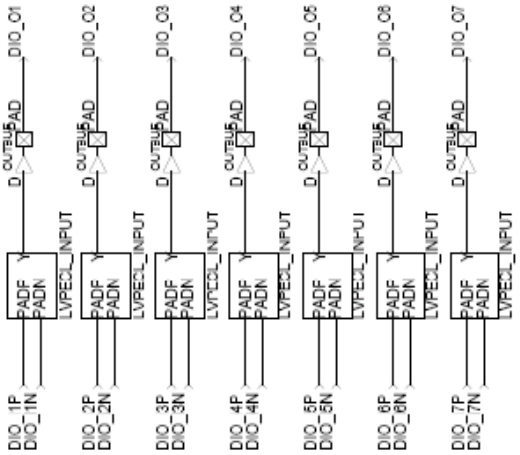
Appendix B DUT Design Schematics and Verilog Files

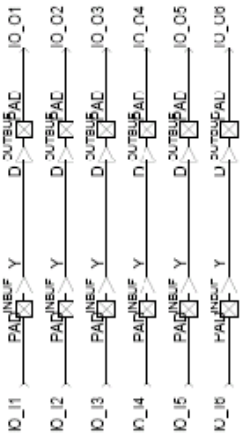


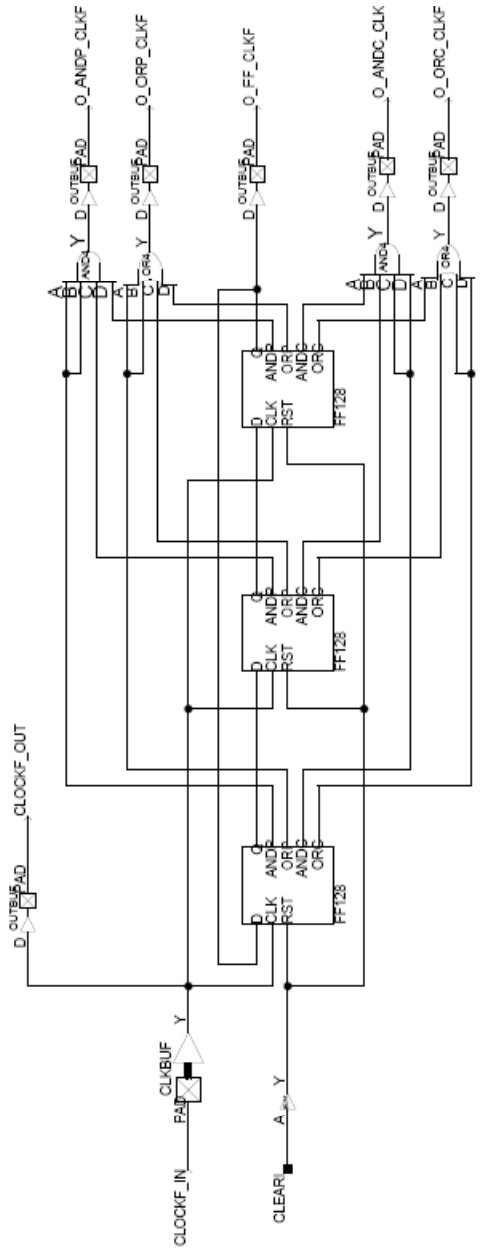

O_ANDP_HCLK
O_ORP_HCLK
O_FF_HCLK
O_ANDC_HCL
O_ORC_HCLK
OUTBUF
PAD
D
Y
AND4
OR4
A
B
C
D
G
AND
OR
CLK
RST
FF32
FF128
HCLKBUF
HCLOCK_IN
CLEAR

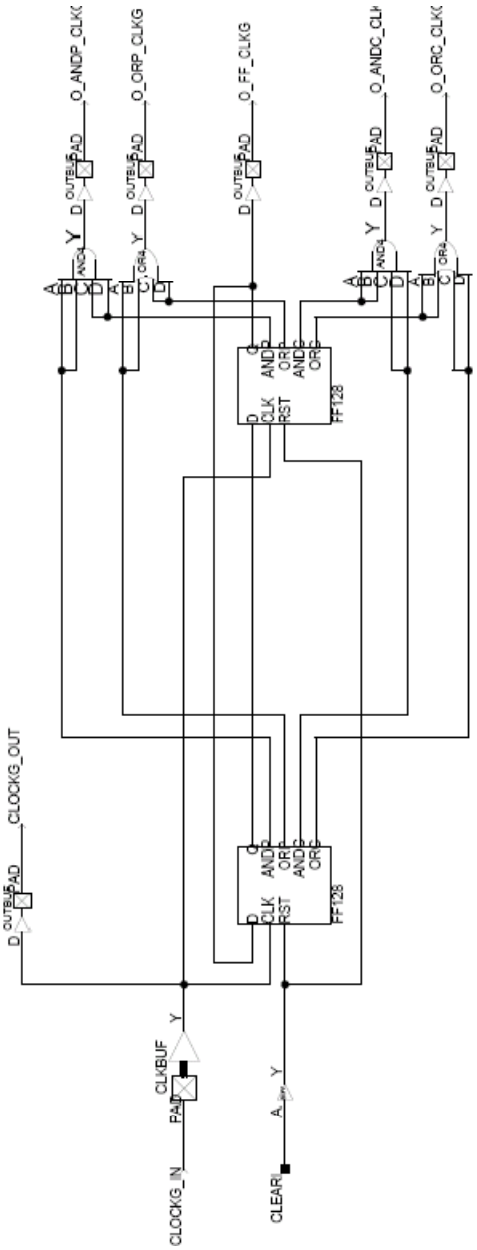

QI
In
Out
BUFF2p3k
D
OUTBUF
PAD
O_BS

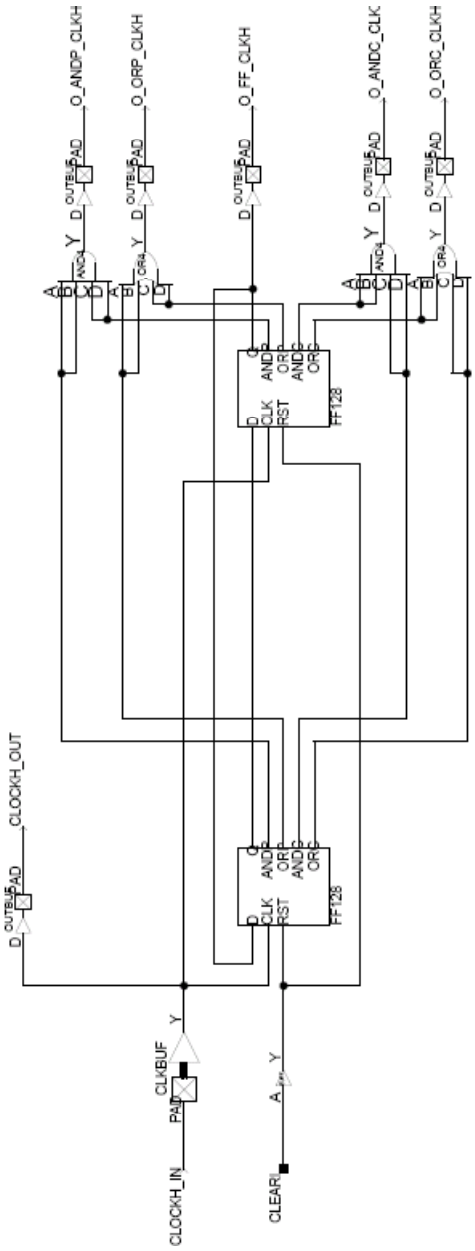

YQ7
YQ6
YQ5
YQ4
YQ3
YQ2
YQ1
YQ0
OUTBUF_AD
D
Y
A
INBUF
PAD
CLK0
QA2
QA0
QA1
DA


DIO_1P
DIO_1N
DIO_2P
DIO_2N
DIO_3P
DIO_3N
DIO_4P
DIO_4N
DIO_5P
DIO_5N
DIO_6P
DIO_6N
DIO_7P
DIO_7N
PADP
PADN
LVPECL_INPUT
D
Y
OUTBUF
PAD
DIO_O1
DIO_O2
DIO_O3
DIO_O4
DIO_O5
DIO_O6
DIO_O7


IO_I1
IO_I2
IO_I3
IO_I4
IO_I5
IO_I6
IO_O1
IO_O2
IO_O3
IO_O4
IO_O5
IO_O6
INBUF
OUTBUF
PAD
Y
D


O_ANDP_CLKF
O_ORP_CLKF
O_FF_CLKF
O_ANDC_CLK
O_ORC_CLKF
CLOCKF_OUT
CLKBUF
CLOCKF_IN
CLEAR
FF128


O_ANDP_CLKG
O_ORP_CLKG
O_FF_CLKG
O_ANDC_CLKG
O_ORC_CLKG
CLOCKG_OUT
CLKBUF
CLOCKG_IN
CLEAR
FF128


O_ANDP_CLKH
O_ORP_CLKH
O_FF_CLKH
O_ANDC_CLK
O_ORC_CLKH
OUTBUF
PAD
AND4
OR4
FF128
CLKBUF
CLOCKH_OUT
CLOCKH_IN
CLEARI

```
// BUFF2p3k.v
`timescale 1 ns/100 ps

module BUFF2p3k (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*/;

BUFF1k buff1k_1 (.In(In), .Out(x1));
BUFF1k buff1k_2 (.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(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

// 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

// Top_RAM_Module.v
`timescale 1 ns/100 ps

module Top_RAM_Module(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 [5:0] Q_RAM;

wire Gnd, Vcc;
wire mx0, mx1;
wire [12:0] rc;
wire [3:0] 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 [5:0] DIN;
wire [5:0] Q_b0;
wire [5:0] Q_b1;
wire [5:0] Q_b2;
wire [5:0] Q_b3;
//wire [5:0] Q_b4;
//wire [5:0] Q_b5;
//wire [5:0] Q_b6;
//wire [5:0] 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_13 counter_0(.Enable(RC_en), .Aclr(RC_clr), .Clock(RC_clk), .Q(rc));

decoder_2to4 decoder_0(.Data0(rc[11]), .Data1(rc[12]), .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_2048x6 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;

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

ram_2048x6 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_2048x6 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_2048x6 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_2048x3 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_2048x3 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_2048x3 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_2048x3 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_6x4 mux_6x4_0(.Data0_port(Q_b0), .Data1_port(Q_b1), .Data2_port(Q_b2),
               .Data3_port(Q_b3), .Sel0(rc[11]),
```

```
        .Sel1(rc[12]), .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: 6.2 SP2 6.2.52.7


module counter_13(Enable,Aclr,Clock,Q);
input Enable, Aclr, Clock;
output [12:0] Q;

    wire ClrAux_0_net, ClrAux_7_net, MX2_1_Y, MX2_7_Y, MX2_4_Y,
        CM8_0_Y, MX2_10_Y, MX2_9_Y, MX2_3_Y, MX2_5_Y, MX2_6_Y,
        MX2_0_Y, MX2_8_Y, MX2_2_Y, MX2_11_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_7_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[5]), .Y(MX2_9_Y));
    DFC1D DFC1D_Q_2_inst(.D(MX2_6_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_4_Y), .CLK(Q[11]), .CLR(
        ClrAux_7_net), .Q(Q[12]));
    DFC1D DFC1D_Q_3_inst(.D(MX2_11_Y), .CLK(Q[2]), .CLR(
        ClrAux_0_net), .Q(Q[3]));
    DFC1D DFC1D_Q_4_inst(.D(MX2_5_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[3]), .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[2]), .Y(MX2_6_Y));
    MX2 MX2_3(.A(VCC), .B(GND), .S(Q[10]), .Y(MX2_3_Y));
    DFC1D DFC1D_Q_11_inst(.D(MX2_10_Y), .CLK(Q[10]), .CLR(
        ClrAux_7_net), .Q(Q[11]));
    MX2 MX2_10(.A(VCC), .B(GND), .S(Q[11]), .Y(MX2_10_Y));
    BUFF BUFF_ClrAux_7_inst(.A(Aclr), .Y(ClrAux_7_net));
```

```
    MX2 MX2_4(.A(VCC), .B(GND), .S(Q[12]), .Y(MX2_4_Y));
    DFC1D DFC1D_Q_5_inst(.D(MX2_9_Y), .CLK(Q[4]), .CLR(
        ClrAux_0_net), .Q(Q[5]));
    DFC1D DFC1D_Q_9_inst(.D(MX2_8_Y), .CLK(Q[8]), .CLR(
        ClrAux_7_net), .Q(Q[9]));
    MX2 MX2_5(.A(VCC), .B(GND), .S(Q[4]), .Y(MX2_5_Y));
    MX2 MX2_8(.A(VCC), .B(GND), .S(Q[9]), .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[1]), .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_10_inst(.D(MX2_3_Y), .CLK(Q[9]), .CLR(
        ClrAux_7_net), .Q(Q[10]));

endmodule

`timescale 1 ns/100 ps
// Version: 6.2 SP2 6.2.52.7


module decoder_2to4(Data0,Data1,Eq);
input Data0, Data1;
output [3:0] Eq;


    AND2A AND2A_Eq_1_inst(.A(Data1), .B(Data0), .Y(Eq[1]));
    AND2 AND2_Eq_3_inst(.A(Data0), .B(Data1), .Y(Eq[3]));
    AND2A AND2A_Eq_2_inst(.A(Data0), .B(Data1), .Y(Eq[2]));
    AND2B AND2B_Eq_0_inst(.A(Data0), .B(Data1), .Y(Eq[0]));

endmodule

`timescale 1 ns/100 ps
// Version: 6.2 SP2 6.2.52.7


module ram_2048x6(Data,Q,WAddress,RAddress,WE,RE,WClock,RClock);
input [5:0] Data;
output [5:0] Q;
input [10:0] WAddress, RAddress;
input WE, RE, WClock, RClock;

    wire WEP, REP, VCC, GND;

    VCC VCC_1_net(.Y(VCC));
    GND GND_1_net(.Y(GND));
    RAM64K36P ram_2048x6_R0C2(.WCLK(WClock), .RCLK(RClock),
        .DEPTH0(GND), .DEPTH1(GND), .DEPTH2(GND), .DEPTH3(GND),
        .WEN(WEP), .WW0(VCC), .WW1(GND), .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[4]), .WD1(Data[5]),
    .WD2(GND), .WD3(GND), .WD4(GND), .WD5(GND), .WD6(GND),
    .WD7(GND), .WD8(GND), .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(REP), .RW0(VCC), .RW1(GND), .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[4]), .RD1(Q[5]), .RD2()
    , .RD3(), .RD4(), .RD5(), .RD6(), .RD7(), .RD8(), .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 REBUBBLE(.A(RE), .Y(REP));
  INV WEBUBBLE(.A(WE), .Y(WEP));
  RAM64K36P ram_2048x6_R0C1(.WCLK(WClock), .RCLK(RClock),
    .DEPTH0(GND), .DEPTH1(GND), .DEPTH2(GND), .DEPTH3(GND),
    .WEN(WEP), .WW0(VCC), .WW1(GND), .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[2]), .WD1(Data[3]),
    .WD2(GND), .WD3(GND), .WD4(GND), .WD5(GND), .WD6(GND),
    .WD7(GND), .WD8(GND), .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(REP), .RW0(VCC), .RW1(GND), .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[2]), .RD1(Q[3]), .RD2()
    , .RD3(), .RD4(), .RD5(), .RD6(), .RD7(), .RD8(), .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());
   RAM64K36P ram_2048x6_R0C0(.WCLK(WClock), .RCLK(RClock),
      .DEPTH0(GND), .DEPTH1(GND), .DEPTH2(GND), .DEPTH3(GND),
      .WEN(WEP), .WW0(VCC), .WW1(GND), .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(GND), .WD3(GND), .WD4(GND), .WD5(GND), .WD6(GND),
      .WD7(GND), .WD8(GND), .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(REP), .RW0(VCC), .RW1(GND), .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()
      , .RD3(), .RD4(), .RD5(), .RD6(), .RD7(), .RD8(), .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: 6.2 SP2 6.2.52.7

module mux_6x4(Data0_port,Data1_port,Data2_port,Data3_port,Sel0,
      Sel1,Result);
input [5:0] Data0_port, Data1_port, Data2_port, Data3_port;
input Sel0, Sel1;
output [5:0] Result;

   MX4 MX4_Result_0_inst(.D0(Data0_port[0]), .D1(Data1_port[0]),
      .D2(Data2_port[0]), .D3(Data3_port[0]), .S0(Sel0), .S1(
      Sel1), .Y(Result[0]));
   MX4 MX4_Result_2_inst(.D0(Data0_port[2]), .D1(Data1_port[2]),
      .D2(Data2_port[2]), .D3(Data3_port[2]), .S0(Sel0), .S1(
      Sel1), .Y(Result[2]));
   MX4 MX4_Result_5_inst(.D0(Data0_port[5]), .D1(Data1_port[5]),
      .D2(Data2_port[5]), .D3(Data3_port[5]), .S0(Sel0), .S1(
      Sel1), .Y(Result[5]));
   MX4 MX4_Result_1_inst(.D0(Data0_port[1]), .D1(Data1_port[1]),
      .D2(Data2_port[1]), .D3(Data3_port[1]), .S0(Sel0), .S1(
      Sel1), .Y(Result[1]));
```

```
    MX4 MX4_Result_4_inst(.D0(Data0_port[4]), .D1(Data1_port[4]),
        .D2(Data2_port[4]), .D3(Data3_port[4]), .S0(Sel0), .S1(
        Sel1), .Y(Result[4]));
    MX4 MX4_Result_3_inst(.D0(Data0_port[3]), .D1(Data1_port[3]),
        .D2(Data2_port[3]), .D3(Data3_port[3]), .S0(Sel0), .S1(
        Sel1), .Y(Result[3]));

endmodule
```



Microsemi Corporate Headquarters
2381 Morse Avenue, Irvine, CA 92614
Phone; 949.221.7100 · Fax: 949.756.0308
www.microsemi.com

Microsemi Corporation (NASDAQ: MSCC) offers the industry's most comprehensive portfolio of semiconductor technology. Committed to solving the most critical system challenges, Microsemi's products include high-performance, high-reliability analog and RF devices, mixed signal integrated circuits, FPGAs and customizable SoCs, and complete subsystems. Microsemi serves leading system manufacturers around the world in the defense, security, aerospace, enterprise, commercial, and industrial markets. Learn more at **www.microsemi.com** .

© 2010 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.