```
        Collapsed Stuck Fault Summary Report
-----------------------------------------------
Fault class                      code   #faults
-------------------------------  ----  ---------
Detected                          DT      25228
Possibly detected                 PT          0
Undetectable                      UD          6
ATPG untestable                   AU          1
Not detected                      ND          0
-----------------------------------------------
Total faults                              25235
Test coverage                            100.00%
-----------------------------------------------
```

Phase 3: Standard boundary scan design

We need to write the top module in order to generate the frame as shown and the script **phase3.tcl** will insert the TAP controller with standard instructions EXTEST, SAMPLE, PRELOAD & BYPASS.

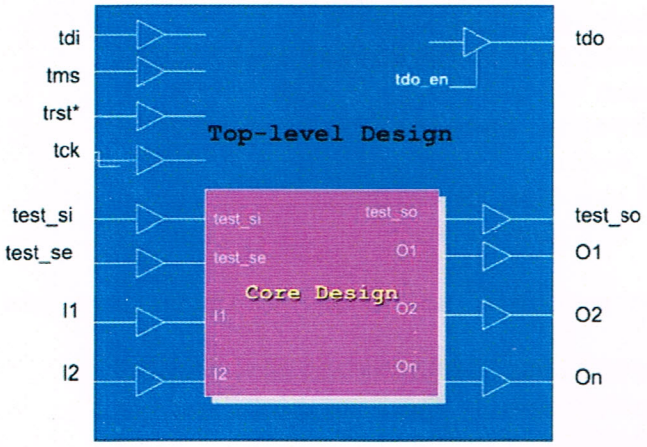


Figure 1 - BSD Ready Design

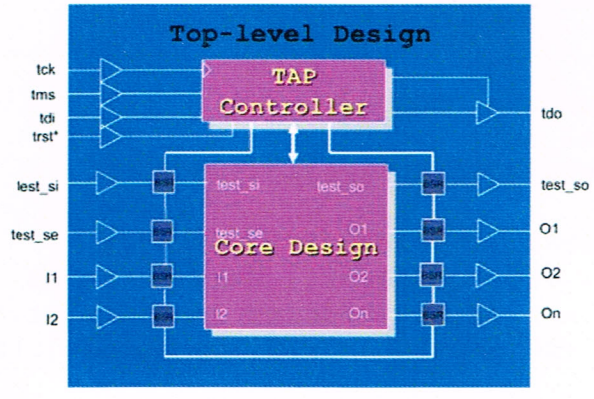


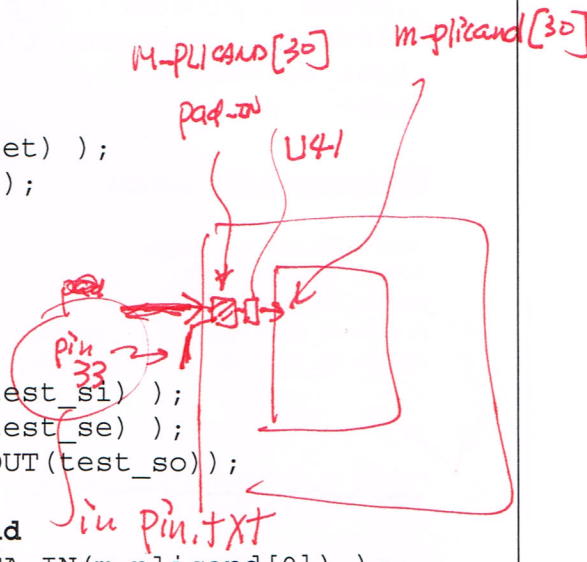
Figure 2 TAP Inserted Design

Top.v

```
module TOP (CLK,RESET,M_PLICAND,M_PLIER,PRODUCT,TEST_SI,TEST_SO,
          TEST_SE,TCK,TRSTN,TDI,TMS,TDO);
output [63:0] PRODUCT;
input [31:0] M_PLICAND;
input [31:0] M_PLIER;
input CLK, RESET, TEST_SI, TEST_SE;
output TEST_SO;
//boundry scan ports
input TCK, TDI, TRSTN, TMS;
output TDO;
wire [63:0] product;
wire [31:0] m_plicand;
wire [31:0] m_plier;
wire clk, reset, test_si, test_se;
wire test_so;
GTECH_INBUF U1 ( .PAD_IN(RESET), .DATA_IN(reset) );
GTECH_INBUF U2 ( .PAD_IN(CLK), .DATA_IN(clk) );
GTECH_INBUF U3 ( .PAD_IN(TCK));
GTECH_INBUF U4 ( .PAD_IN(TRSTN));
GTECH_INBUF U5 ( .PAD_IN(TDI));
GTECH_INBUF U6 ( .PAD_IN(TMS));
GTECH_INOUTBUF  U7 ( .PAD_INOUT(TDO));
GTECH_INBUF U8 ( .PAD_IN(TEST_SI), .DATA_IN(test_si) );
GTECH_INBUF U9 ( .PAD_IN(TEST_SE), .DATA_IN(test_se) );
GTECH_OUTBUF U10 ( .PAD_OUT(TEST_SO), .DATA_OUT(test_so));

// inserting buffer for 32-bit input m_plicand
GTECH_INBUF U11 ( .PAD_IN(M_PLICAND[0]), .DATA_IN(m_plicand[0]) );
GTECH_INBUF U12 ( .PAD_IN(M_PLICAND[1]), .DATA_IN(m_plicand[1]) );
GTECH_INBUF U13 ( .PAD_IN(M_PLICAND[2]), .DATA_IN(m_plicand[2]) );
.            .                          .                        .
.            .                          .                        .
.            .                          .                        .
GTECH_INBUF U41 ( .PAD_IN(M_PLICAND[30]), .DATA_IN(m_plicand[30]) );
GTECH_INBUF U42 ( .PAD_IN(M_PLICAND[31]), .DATA_IN(m_plicand[31]) );

// inserting input buffer for 32-bit input m_plier
GTECH_INBUF U43 ( .PAD_IN(M_PLIER[0]), .DATA_IN(m_plier[0]) );
GTECH_INBUF U44 ( .PAD_IN(M_PLIER[1]), .DATA_IN(m_plier[1]) );
GTECH_INBUF U45 ( .PAD_IN(M_PLIER[2]), .DATA_IN(m_plier[2]) );
.            .                          .                        .
.            .                          .                        .
.            .                          .                        .
GTECH_INBUF U73 ( .PAD_IN(M_PLIER[30]), .DATA_IN(m_plier[30]) );
GTECH_INBUF U74 ( .PAD_IN(M_PLIER[31]), .DATA_IN(m_plier[31]) );
```



```
// inserting output buffer for 64-bit input m_plicand
GTECH_OUTBUF U75(.PAD_OUT(PRODUCT[63]), .DATA_OUT(product[63]));
GTECH_OUTBUF U76(.PAD_OUT(PRODUCT[62]), .DATA_OUT(product[62]));
GTECH_OUTBUF U77(.PAD_OUT(PRODUCT[61]), .DATA_OUT(product[61]));
.              .                        .                      .
.              .                        .                      .
.              .                        .                      .
GTECH_OUTBUF U136(.PAD_OUT(PRODUCT[2]), .DATA_OUT(product[2]));
GTECH_OUTBUF U137(.PAD_OUT(PRODUCT[1]), .DATA_OUT(product[1]));
GTECH_OUTBUF U138(.PAD_OUT(PRODUCT[0]), .DATA_OUT(product[0]));


booth4_mult multiplier_core
(.product(product),.m_plicand(m_plicand),.m_plier(m_plier),
.clk(clk),.reset(reset),.test_si(test_si),.test_se(test_se),
.test_so(test_so));
endmodule
```

Pin mapping file: pin.txt

```
PACKAGE=my_package;
PORT= CLK, PIN=P1;
PORT= RESET, PIN=P2;
PORT= M_PLICAND[0], PIN=P3;
PORT= M_PLICAND[1], PIN=P4;
.       .       .       .
.       .       .       .
.       .       .       .
PORT= M_PLICAND[30], PIN=P33;
PORT= M_PLICAND[31], PIN=P34;

PORT= M_PLIER[0], PIN=P35;
PORT= M_PLIER[1], PIN=P36;
.       .       .       .
.       .       .       .
.       .       .       .
PORT= M_PLIER[30], PIN=P65;
PORT= M_PLIER[31], PIN=P66;

PORT= PRODUCT[0], PIN=P67;
PORT= PRODUCT[1], PIN=P68;
.       .       .       .
.       .       .       .
.       .       .       .
PORT= PRODUCT[61], PIN=P128;
PORT= PRODUCT[62], PIN=P129;
PORT= PRODUCT[63], PIN=P130;

PORT= TEST_SI, PIN=P131;
PORT= TEST_SO, PIN=P132;
PORT= TEST_SE, PIN=P133;
PORT= TRSTN, PIN=P134;
PORT= TDO, PIN=P135;
```

```
PORT= TDI, PIN=P136;
PORT= TMS, PIN=P137;
PORT= TCK, PIN=P138;
```

Script for inserting Boundary Scan cells: phase3.tcl

```
#----------- SET VARIABLES FOR BOUNDARY SCAN INSERTION ---------

set link_library /home/kethinpr/VTV-1/saed90nm_typ.db
set target_library [list /home/kethinpr/VTV-1/saed90nm_typ.db]
set synthetic_library
/opt/CAD/Synopsys/Current/Synthesis/libraries/syn/class.sdb
set link_library [concat $link_library $synthetic_library
$target_library]
set synthetic_library
/opt/CAD/Synopsys/Current/Synthesis/libraries/syn/dw_foundation.sldb
set link_library [concat $link_library $synthetic_library
$target_library]
set synthetic_library
/opt/CAD/Synopsys/Current/Synthesis/libraries/syn/dft_jtag.sldb
set link_library [concat $link_library $synthetic_library
$target_library]


# Read in top-level design

read_verilog /home/kethinpr/VTV-1/multiplier_scan.v
read_verilog /home/kethinpr/VTV-1/top.v
set current_design TOP
link

set_fix_multiple_port_nets -all -buffer_constants
set_dft_signal -view existing_dft -type ScanClock -port CLK -timing
[list 45 55]

#----- dont_touch CORE and pads
set_dont_touch [get_designs  booth4_mult*]
set_dont_touch [list GTECH_INBUF GTECH_INOUTBUF GTECH_OUTBUF]

#----------------- SET VARIABLES FOR TAP CONTROLLER -----------
#---read pin map for BSR cell order

read_pin_map /home/kethinpr/VTV-1/pin.txt
set_dft_signal -view spec -type TCK -port TCK
set_dft_signal -view spec -type TDI -port TDI
set_dft_signal -view spec -type TDO -port TDO
set_dft_signal -view spec -type TMS -port TMS
set_dft_signal -view spec -type TRST -port TRSTN
```