An Optimized DFT and Test Pattern Generation Strategy for an Intel High Performance Microprocessor

David M. Wu, Mike Lin, Madhukar Reddy, Talal Jaber, Anil Sabbavarapu, Larry Thatcher

Intel Corporation

Contact Author: david.m.wu@intel.com

Abstract

This paper describes an optimized DFT architecture and its implementation strategy for an Intel high performance (>3 GHz) microprocessor. Major DFT features and ATPG techniques implemented are described and key results are presented to show the return-on-investments (ROI) in the high volume manufacturing (HVM) test environments.

1. Introduction

This paper describes the design for testability and debug, and ATPG strategies for a high performance (>3 GHz), high density microprocessor (>150M Transistors) with super rich system features for Intel high end platforms. Our DFX team had committed to develop and implement the most advanced and cost effective DFX strategy that would meet the quality goal predicted by an Intel Internal prediction tool developed by the Intel CQN (Corporate Quality Network) group. To meet this challenge, a highly skilled team was formed with key DFX experts in the various DFX areas. In the technology readiness phase, a DFX portfolio was developed, feasibility analyzed, gaps, mitigation plan and backup plans were identified. At the end of the technology readiness phase, we decided to optimize industrial style scan and DFT methodologies [1-5] with some innovative DFT features to reduce overall test cost. In order to handle test data volume limitation and the potential Vcc droop problems during the HVM test process, we developed a partition ATPG strategy [6-9] and implemented required scan DFT controls to isolate the targeted logical units or clusters. In addition, some innovative methodologies for other DFT/DFD (Design for Debug) areas such as array DFT, I/O DFT, Design for Burn-In and Design for Debug were developed.

Section 2 describes a Partition ATPG (PATPG) architecture and the required scan DFT features to overcome the limitation of the commercial ATPG capacities and to reduce the potential di/dt impact during manufacture tests. To meet the overhead constraint and performance goal, we developed a 'Skip-Scan' technique and established a set of skip scan design rules and a very strict scan waiver process to ensure meeting the test coverage requirements even if we minimize the silicon area overheads.

Section 3 depicts the array DFT design and validation strategies. Since the allowed DPM budget for entire arrays on the chip is extremely low, it is necessary to have a very comprehensive array DFT test strategy. We use Programmable Array BIST (PBIST) [10] to test the largest arrays and an optimized array BIST technique to test smaller and medium arrays. Direct Access Testing (DAT) [11] for array access and diagnosis and Programmable Weak Write Test Mode (PWWTM) [13] for memory cell stability test to reduce the test time.

Section 4 describes an enhanced TAP controller called 'Integrated Test Controller (ITC)' that has special hooks to control the additional DFT and DFD features we have implemented in the silicon. The key DFD and IO DFT features inherited design from the previous Intel microprocessor [14] are addressed in this section.

Section 5 describes burn-in DFT techniques. A built-in self test feature is used to generate burn-in toggling test patterns for the logic circuits. Two BIST schemes were used to provide high toggle coverage for burn-in of the arrays.

Section 6 highlights the key difference of design-for debug features mostly described by the previous paper on the Intel microprocessors [15]. A scanout system is developed and validated. A system clock freeze feature during scan test is implemented to help at-speed debug.

Key learnings from the DFX implementation processes and results of ATPG data for key logical partition units and clusters are depicted in the last section.

2. Logic DFT and ATPG

Generating scan ATPG vectors for the targeted microprocessor 'PX' poses many formidable challenges due to the sheer complexity of the tasks. One of the key challenges is imposed by the extra large number of logic

gates and faults presented in the full chip netlist that created problems for ATPG tools capabilities to effectively generate test patterns. In this section, we describe a Partition ATPG (PATPG) methodology adopted by us to break down the problem into a set of smaller and manageable sub-problems. We partition the design into smaller blocks and generate ATPG vectors at full-chip level for these blocks so that complex transformation of ATPG vectors is not needed to apply them at full chip level. The required Scan DFT and PATPG methodologies are described in this section.

The PX microprocessor design is highly complicated with multiple clock domains, multi-cycle paths, domino circuit including OTB (Over-Time Borrowing) and static circuits, and extraordinary high number of transistor circuits. Analysis shows that a cluster size of logic can be handled by the commercial ATPG tools used by PX. In addition, to overcome the potential di/dt problem, a lower level of partition becomes necessary. To accommodate these requirements, PX scan DFT adopted a Hierarchical Scan Architecture (HSA). The HSA divided the full chip into a group of "Clusters". A cluster is usually a top level functional logical entity within the microprocessor such as the floating-point execution cluster. A scan control logic block was designed to enable the testing of a single cluster or a combination of multiple clusters. Test patterns can be generated accordingly for either a single cluster or for a combination of multiple clusters.

Similarly, a cluster can be divided into a group of 'Units'. A sub-level test controller was designed to enable the testing of either a single unit or a combination of multiple units. In the ATPG process, we expect that there should be no issue for the commercial ATPG tool to handle size and complexity of a cluster. However, we have prepared the 'unit' level ATPG test generation capability in the case of intolerable di/dt problem during the HVM testing process.

There are 36 scan chains distributed throughout the full chip hierarchically. Each "Cluster Partition" is bounded by scannable scan chains. Most "Unit Partitions" are also bounded by scan chains although there are exceptions. There could combinational logic at the boundaries of partitions. Our overall methodology benefits from a scan architecture with partitioning as one of the primary goals and has the following related features:

- Partitions targeted for test are flexible and configurable. For example, partition under test (PUT) can be a cluster, a unit, or combinations of units/clusters. This enables us to scale the size of PUT according to the complexity presented to the ATPG tool or desired test application, and also target ATPG for faults in the "exposed" logic between partitions.
- Scan chains in partition not under test (PNUT) can be bypassed and one can configure scan chains for any PUT selected to connect to the chip pins at full-chip level. This optimizes test time as only scan chains in PUT are configured between chip pins
- Our methodology requires neither separate scan chains nor separate functional clock control for scan instances at the boundaries of ATPG partitions. This reduces scan design complexity and overhead significantly and leads to "design friendly" scan design.

Note that partitioning netlists is an old topic in EDA literature. There are several published papers on strategies for partitioning netlists, faults and patterns for parallel and/or distributed ATPG/fault simulation [6-9]. The objective of this paper is not to propose another such algorithm. The main contribution of this paper is to describe a practical method illustrating the implementation challenges for such a "divide and conquer" strategy in complex and high performance processors and using a commercial ATPG and simulation validation tools.

2.1 SCAN DFT ARCHITECTURE TO SUPPORT PATPG

PX has seven logical clusters that make up the full chip. Each one of those clusters has one cluster test controller (CTC) module and at least one unit test controller (UTC) module. These CTCs and UTCs are used to configure the scan chains and control the clocking of PUTs and PNUTs in ATPG mode. Every CTC module buffers 36 scan chains. It also contains control and staging logic for scan control signals. The outputs of the CTC drive the UTC modules. Each UTC contains a control register for PATPG control. These register bits are connected into a signals and clock control special scan chain that connects all UTC control registers into one global chain. This register consists of control bits for scan shift clock enable, controlling various functional clocks, and the "functional clock gating test override" signal. An example of scan chain routing and partitioning for ATPG is shown in Figure1.

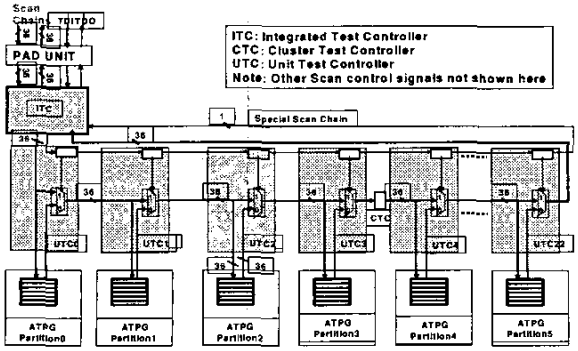

Figure 1: Scan Chain Routing & Partitioning for ATPG

The scan shift clock enable controls scan chain selection and scan shifting. This bit controls a bypass mux in the UTC for the 36 scan chains. If the scan shift clock enable is active then the 36 scan chains that are connected to the UTC are part of the active chains for ATPG, or if it is not active the scan chains that are connected to the UTC are bypassed with a mux like structure.

2.2 Skip Scan Methodologies

The ATPG process for PX has been a challenging task. Not only we have to deal with highly demanded test coverage numbers, we also have to deal with the limitation of budgets in terms of the silicon area, leakage power, and scan performance impact. To meet our challenge, a set of well documented design-for-test rules are communicated to all PX design teams and a well developed scan waiver process was 'enforced' throughout the development phase of the project. As a result, a very cost effective scan implementation was very successfully implemented in PX. One of the very valuable techniques is the 'Skip Scan' Technique. It is also called Datapath Interleaved Scan (DI-Scan) Technique since it is the most effective to be applied in the datapath logic following some Skip Scan design rules. The Skip Scan technique helped us significantly in meeting an aggressive scan area budget and still maintain performance target and test coverage goal. Figure 2 shows simple examples of DI-scan datapath pipelines in which scan are skipped in one sequential stage between two scanned sequential stages. Data flow direction is from left to right as indicated. The top portion in this figure shows a flop based datapath design and the bottom portion shows a latch based design. The scan type employed LSSD-Like Design with edge triggered flip-flop. In Figure 2, CLK and CLK# are ph1 and ph2 clocks respectively. The skipped functional logic parts are shown in shaded boxes, the scan logic parts are shown in white boxes and the "clouds" represent combinational logic. Latches driven by phi2 clock, CLK#, are transparent when clock is inactive and hence are not scanned even in full-scan design. In a full-scan design, the non-scan stages would also have been converted to scan stages. Note that, both the datapath pipelines start with scan and end with scan. Skipping scan in at most one sequential stage between two scanned sequential stages is the predominant DI-scan technique employed in datapath pipelines. This minimized the risk of lowered test coverage from scan based ATPG tools. However, for some data paths, we relaxed this restriction and allowed for skipping scan in two consecutive sequential stages for improved area and timing after completing careful analysis and based on ATPG test coverage results. In a pipelined microprocessor designs, often the micro-architecture calls for delaying certain signals by several clocks in order to synchronize them with the rest of the logic. This is accomplished by using several back-to-back register stages. In other cases, often FIFO structures are used to capture data that is transmitted at a high burst rate for consumption later. These FIFOs also have back-to-back register stages with no logic in between. In this paper, we call such stages as "staging pipelines". We further extended the skip-scan technique more aggressively into staging pipelines by taking advantage of absence of combinational logic (except for buffers and inverters) in staging pipelines and skipping scan in more sequential stages. From ATPG studies we have done, we concluded that we can skip scan in up to a maximum of three consecutive stages and with out significant degradation in ATPG test coverage and with some increased ATPG execution times.

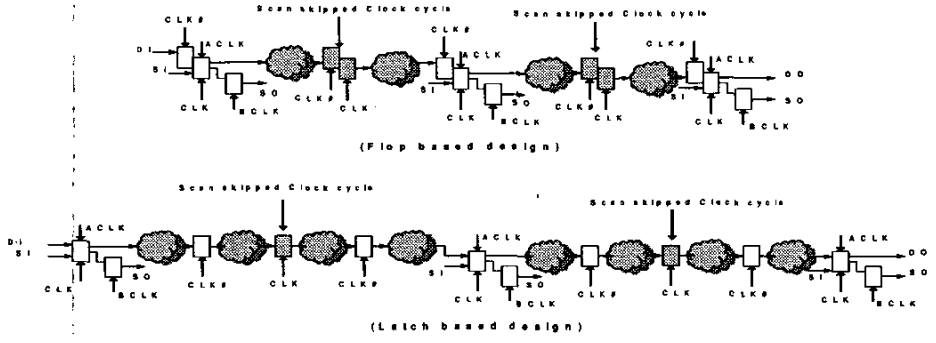

FIGURE 2: DATAPATH INTERLEAVED SCAN IN FLOP BASED DESIGNS AND LATCH BASED DESIGNS

In PX we architected scan with partitioning for ATPG as one of the primary goals. The DI-scan technique is employed in PX in such a way that we can generate ATPG tests for DI-scan pipelines in an integrated manner with rest of the logic in Partition Under Test. Since our goal is to generate ATPG patterns using the scan based ATPG tools, we have restricted the number of sequential stages where we are skipping scan and additionally defined required attributes of the data paths where DI-scan could be employed. In section 2.3, we first define DI-scan rules for ideal datapath pipelines from ATPG point of view and illustrate how a scan ATPG tool with limited sequential capabilities would be able to generate tests for such pipelines. As some of these rules are violated in some real datapath pipelines, we will explain what complexity such violations add to ATPG and how they may impact the test coverage.

2.3 Skip Scan Design Rules in the Datapath: DI-Scan Rules

To make the DI-Scan work, it is important to establish some DFT rules. Key DFT rules for DI-Scan rules including the following:

1. DI-scan is used only in datapath pipelines.

2. Control logic must be full-scan.

3. DI-scan pipelines are "uniform" and the dataflow direction is only forward. That is, no feed forward or feed backward through sequential stages. This prevents logic of uneven sequential depth to converge and enables known states from scan chains to propagate forward through datapath pipeline in a "wave" like manner when functional clocks are exercised. In case of pipelines with latch based design, both P-latches and N-latches must be considered as sequential stages. N-latches are at clock cycle boundaries. P-latches are at half-cycle boundaries and are transparent when clocks are inactive. P-latches are not scanned. (Note: No feedback loops in combinational logic that create sequential state are allowed. No feedback loops made of a latch and combinational logic are allowed. These rules are part of basic scan DFT rules).

4. DI-scan pipeline begins with scan at first clock cycle and ends with scan at last clock cycle in the pipeline. Scan is skipped only at sequential stages at alternate clock cycles when traced forward from the first clock cycle. It is OK if scan is not skipped at some clock cycles. In case of pipelines with latch based design, skip-scan pipelines should look like this: Scanned N-latch -> P-latch -> unscanned N-latch (scan skipped clock cycle) -> P-latch -> Scanned N-latch -> P-latch and so on. Note that only a section of pipeline is described here.

5. Non-scan state elements in skip-scan must hold states during scan shifts (excludes P-latches which are transparent). This is trivially satisfied in PX scan architecture as functional clocks are inactive during scan shifting.

6. It is imperative that all functional clock gating be implemented within the Local Clock Enable (LCE) block. This ensures DFT overrides for functional clock gating are implemented. No further clock gating is used after LCE.

7. It must be possible to control the functional clock gating logic combinationally from scan flops. This ensures patterns can be generated to test clock gating logic with DFT overrides set to inactive state.

8. For flops with enables on data input, it must be possible to control the enable logic combinationally from scan flops. This includes self-loop situations for flops.

9. In DI-scan datapath pipelines, selects for multiple tri-state driven nodes are (needed for contention-free ATPG patterns): Fully decoded within combinational logic, or Fully decoded within a DI-scan stage (i.e., within the two clock cycles logic of a DI-scan stage), or If coming from control logic, fully decoded within combinational logic, or If coming from control logic and not fully-decoded within combinational logic Hold Scan is used.

2.4 ATPG Results for Key Units and Representative Clusters

During the process of establishing the DFT strategy, we faced some challenges in making decision on the ROI (Return-ON-Investment) tradeoff, with no negotiation of performance and quality goals. For clusters or units that do not have any performance sensitivities, the full scan/skip scan methodologies were very successfully implemented and the DFT rules were strongly 'enforced'. However, for some clusters or units that have little room for area overhead that could cause performance issues, we exercised very extensive test coverage analysis to isolate certain logic blocks to be 'non-scannabel' blocks (NSBs). These NSBs are distributes in relatively small amounts of areas across the whole chips and the design engineers were responsible to generate Functional Test Patterns either using structural based functional test (SBFT) or traditional functional test methods to coverage the gaps. The following ATPG data are test coverages results (before the functional tests) that reflect the test coverage gaps due to NSB blocks.

In Table 1, the ATPG data of 5 Units that has implemented very aggressive Skip Scan methods are depicted. The number of logic gates in these units ranged from ~64K to ~460K. The percentage of scan sequentials of these Units ranged form ~31% to ~ 76%. Even though

the actual implementation is 'Skip Scan', we also simulated the 'full scan' (excluding the NSBs) implementation to compare the test coverage results. In Unit 1, the Skip Scan case has a lower percentage (~73%) of scannable sequential but it obtained a better test coverage (~93%) comparing to the case of 'full scan' (~85% scan percentage, exclude NSBs, and ~79% test coverage). One of the key reasons of this result is due to the decoder logic spreading across multiple stages of sequential elements that actually makes the ATPG harder for full-scan circuit. The Unit 2,3,4,5 demonstrated some impressive results of Skip Scan techniques. With a relatively low scan percentage (from ~31% to ~ 77%); the Skip Scan test coverage loss for Unit 5 (~50% scan) is nearly 0%. For Units 2, 3, 4, the Skip Scan coverage gaps comparing to 'full scan' is ranged from ~1% to 6%. As previously stated, additional functional test patterns will be applied to coverage these gaps in order to ensure the targeted test coverages both in stuck-at faults and delay faults are satisfied.

Table 1: Unit level ATPG results, Skip Scan vs. Full Scan

Unit	# Gates	S@ Faults (Uncollapsed)	Total # of flops	UNIT with DI-Scan Datapath		UNIT with Full-Scan Datapath		Comments (ATPG untargeted logic not completely nofaulted. Other T-scan exceptions present. PIs controlled, Pos observed)
				% scan	Test Covg.	% scan	Test Covg.	
Unit 1	140382	294226	4263	73.73%	93.32%	85.78%	79.4%	4345 buses. Sequential decoding. Contention prevention. Untestable faults in select logic.
Unit 2	85747	153896	3626	31.05%	91.24%	98.79%	98.71%	1576 buses. Sequential decoding. Contention prevention. Untestable faults in select logic.
Unit 3	125252	234598	5571	76.84%	97.94%	92.44%	98.19%	2652 buses. Sequential decoding. Contention prevention. Untestable faults in select logic. 2 stage deep non-scan EBB embedded.
Unit 4	464093	361538	17157	40.06%	85.60%	45.38%	86.95%	5930 buses. Sequential decoding. Contention prevention. Untestable faults in select logic. 4 stage deep non-scan EBB embedded. Abort limit of 500 not sufficient
Unit 5	64105	125806	4845	50.28%	99.97%	N.A.	N.A.	EBB level run. 128 pass-gate MUXes, combinationally fully decoded. Unit level coverage for this EBB is 99.13%

Two representative Clusters ATPG results are depicted in Table 2. Cluster 1 has ~900K stuck-at faults with an approximately ~14K sequential elements. It is mostly synthesizable random logic and the test coverage reached 98.2% without taking credits for the ignorable faults and possibly detected faults. Cluster 2 has an approximately 3.45 M faults with 146K sequential elements. This Cluster is a very performance critical and area budget limited cluster, we have implemented many aggressive scan waiver techniques in addition to the Skip Scan methodology and the NSBs isolation to optimize the cost factor but still preserve the defined coverage target. We used only ~48% of scan sequential in this cluster and obtained ~91% test coverage for the entire cluster.

Table 2: Cluster ATPG results

Name	# Faults	Skip Scan	Total # of seqs	% scan	RLS/CBD/EBB	Test Cov	Comments
Cluster 1	**900K**	No	**13.9K**	**99.56%**	RLS	**98.2%**	Nearly Full Scan
Cluster 2	**3.45M**	Yes	**146K**	**48.65%**	CBD/EBB	**91.5%**	Aggressive Scan

3. Array DFT

Our Array DFT test strategy is to use PBIST (Programmable Built-In Self Test) [10] to test the second level cache and use DAT [11] to test the remaining arrays. In addition, we use PWWTM (Programmable Weak Write Test Mode) [13] to test cell-stability defects of 7 large small-signal memories. The following are the detail for each of the 3 modes.

3.1 Programmable BIST (PBIST):

We uses PBIST to test 4 Level 1 (UL1) arrays (Data, Tag, LRU and State arrays). PBIST is a widely used technique within Intel. It not only supports application of all kitchen sink patterns (> 50 Algorithms). It also supports testing of 7 different configurations of UL1 sizes. Figure 3 depicts the diagram of PBIST structure. PBIST can be thought of as a micro controller connected to fairly sophisticated address generation logic and distributed data generation/comparison logic. Access to all portions of the PBIST is available through the JTAG TAP controller. PBIST is used for all at-speed production testing of the UL1. Because the testing is done at operational speed, it can detect many delay related subtle defects. It can also raster memory data at the operational speed for memory repair, fault diagnosis, and yield improvement. PBIST can also be used during POST (Power-On Self-Test) to check the health of UL1. POST runs as part of the chip microcode reset sequence and has the responsibility of reporting to microcode whether the UL1 cache is operable. The maximum size of PX UL1 is 2 M bytes. However, there are 7 possible configurations with 3 possible sizes: 2MB, 1MB, and 0.5MB. Since a PX chip may be sold at any one of the 7 configuration when either some part of UL1 is defective or market demands for smaller cache size version, PBIST is enhanced to test all 7 configurations. Note that such testing is done on top of the possible reconfigurations to substitute the defective column with the spare column.

3.2 Direct Access Testing (DAT): The DAT (Direct Access Test) mode in PX as shown in Figure 4 is very similar to that presented in [11]. The DAT mode allows direct, parallel access from FSB pins to the I/O of the target array through CRB (Control Register Bus) and the local DAT circuit adjacent to the array. This parallel connection provides a much higher bandwidth to transfer array test information back and forth between the tester and the targeted array allowing 100 times faster production test than accessing though the TAP's serial control Register Bus. PX DAT mode covers 99.9% of all arrays including UL1 arrays. In addition, it supports back-to-back test of all arrays except UL1 arrays, which can be tested back-to-back by PBIST (covered above). The key reason to support simple read-write to UL1 arrays is to enable SBFT (Scan Based Functional Test) and FRIT (Functional Random Instruction Test) [12]. In addition, it can be used for dumping 4 UL1 arrays for debug. While PX has more arrays (>110) and more multi-ports arrays with more ports in such arrays in any earlier Pentium 4 CPUs, the percentage of arrays and the percentage of ports in multi-port arrays covered by DAT mode is higher in PX than any previous CPU. As such, the array test coverage (note) of PX is 99.3% - the highest in Pentium 4 family.

3.3 Programmable Weak-Write Test Mode (PWWTM):

In [13] WWTM (Weak-Write Test Mode) was used to detect many stability types of defects prevalent in memory cells. In this mode, a value is opposite to the cell value is "weakly" written into a memory cells such that if in the presence of defect, the value of the memory cell will be flipped. On the other hand, if the cell is stable – free of major defects, the cell value will remain unchanged. Production data, however, show that it is very hard to choose the optimal design point of the weak right circuit such that the stress caused by the weak-write is just strong enough to push the unstable cell to flip the stable without cause significant over-kills or under-kills. To overcome the issue, the WWTM was enhanced to make design point of the weak-write circuit become programmable. This way design point can be programmed to the optimal condition after the enough data is collected from silicon. This technical has been shown to drastically reduce both over-

kills and under-kills in detecting stability defects of small signal memories.

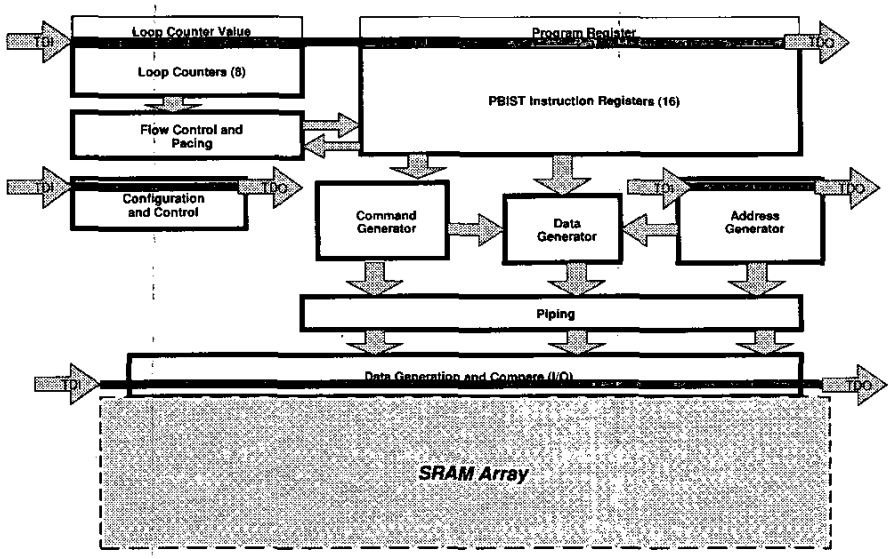

Figure 3: Programmable Built-in Self Test (PBIST) Architecture

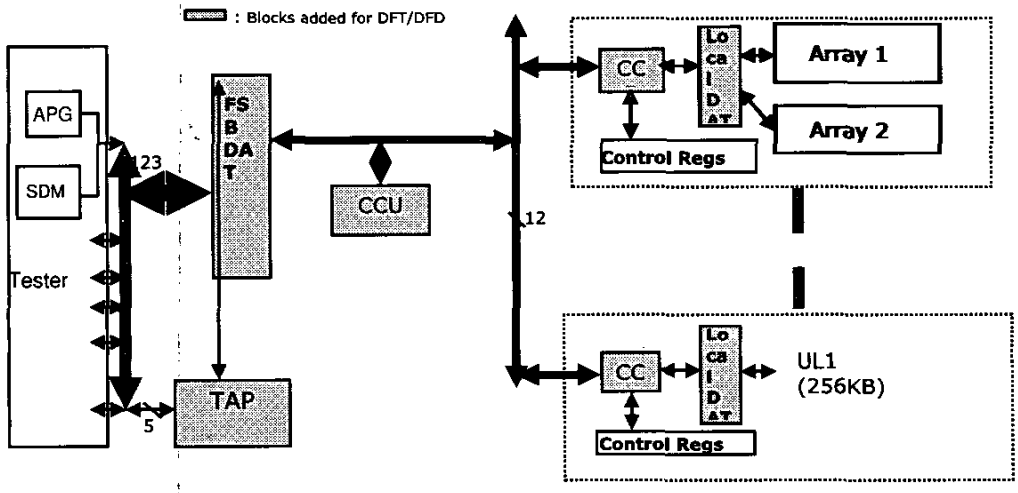

Figure 4: Direct Access Testing (DAT) architecture

4. Integrated Test Controller and IO DFT

The Integrated Test Controller (ITC) includes the Test Access Port (TAP) logic and all the controller logic that controls various debug and test features. The TAP complies with the IEEE 1149.1 ("JTAG") test architecture standard, and additionally provides access to most of the testability and debug features including Micro-breakpoints, Control register bus access, LBIST, Scan, Scanout, Signature mode, thermal sensor control, I/O self-testing, fuse programming and DAT mode . The TAP supports an instruction set of many functions. It provides not only a rich set customer-visible features, but also crucial proprietary functionalities for silicon debug, system validation and production testing .

The TAP logic consists of a finite state machine controller, a serially-accessible instruction register, instruction decode logic and several data registers. The set of data registers includes those described in the 1149.1 standard (the bypass register, device ID register, BIST result register and boundary scan register), as well as several product sepecific registers. These pre-defined registers, together with control logic implemented in the TAP, provide access to all of the testability features.

The TAP logic and all test data registers are accessed serially through 5 dedicated pins on the chip package:

TCK (and thus the TAP itself) is designed to operate at any frequency between 0 Hz and a maximum frequency that will match the bus clock frequency (currently 200 Mhz). But remember that the tap clock frequency has no relation to any other clock on chip (i.e., there aren't any ratios or phase relationships). The tap clock is completely asynchronous to bus clock and the core clocks (although it is possible to purposely run the tap clock at exactly the same frequency as the bus clock, for testing reasons). The 0 Hz frequency means it is possible to stop the tap clock in the middle of a test, and then later on start tap clock back up and continue the test (although care must be taken with certain features to ensure they stay in sync).

TMS, TDI and **TDO** operate synchronously with **TCK** . **TRST#** is an asynchronous input signal. This 5-pin interface operates as defined in the 1149.1 specification.

4.1 The TAP Feature List

The testability features accessed through the TAP are summarized in the table . The second column lists, for each feature, whether it is defined as part of the 1149.1 standard, or product specific. The third column lists the TAP instructions which have been implemented to access each feature. The following table shows the testability features accessed through the TAP

Testability Feature	Feature. defined by	Supported TAP Instructions
Boundary Scan	1149.1	EXTEST, SAMPLE/PRELOAD
Bypass Register	1149.1	BYPASS, HIGHZ, CLAMP
Device Identification Register	1149.1	IDCODE
BIST	1149.1	RUNBIST
Serial Control Register bus access	Product Spec	CRBUSGO, CRBUSNOGO, CRPRELOAD, CRBUSPOLL, CRCANCEL
Array Freeze	Product Spec	ARRAYFRZ
Stalls (Allocator and Front End)	Product Spec	STALLREQ
Thermal sensor control	Product Spec	TSENCAT, TSENTHROT
DAT mode (Parallel Control Register bus access)	Product Spec	DATSERIAL, TESTMODE, ISCANDATMODE
StopClk	Product Spec	STOPCLK, STARTCLK, ALLCLKEN, ALLCLKDIS
Scanout chain, snapshot mode	Product Spec	SCANOUTLOAD, SCANOUTSHIFT
Scanout chain, signature mode	Product Spec	TESTMODE
Tscan	Product Spec	ISCANDATMODE, TSCANTIMER, TSCAN, TSCANSETUP, TSCANCREG, TSCANFNCNTR,
I/O Self-Test	Product Spec	IOTESTLOAD,IOTESTMODE, BSCANREAD
Micro-breakpoint mechanism	Product Spec	BRKPTCTL[A,B]
Probe Mode	Product Spec	PMENTER, PMEXIT, PMNOW, WRSUBPIR, READPDR0, READPDR1,PMSETTHID, PMCLRTHID
Fuse Programming	Product Spec	FUSECTL, FUSESHIFT, FUSECSR, FUSESSR,
Clock Comp Programming	Product Spec	PHASEDSHIFT
Secure/Unsecure modes	Product Spec	LOCK, UNLOCK
Status Reporting	Product Spec	TAPSTATUS, TAPSTATUSPRVT
ViewPLL	Product Spec	VIEWPLL

4.2 IO DFT:

This microprocessor inherits its bus protocol (including IO frequency characteristics - maximum of 800 MTS for Data IOs) from the previous microprocessor design. Key features including AC I/O Loopback DFT that is essential for screening out both functional and timing defects at the pads in order to reduce DPM. The detail design description and specifications were described in [14].

As we have described in the last year's ITC paper, a hybrid DFT architecture including test compression features has been implemented in this microprocessor. The detail description was presented in [17].

5. Burn-in (BI) DFT and methodology

PX implements a distinct feature WRPBIST-BI mode to increase toggle coverage of random logic, which is simple to program and easy to achieve target toggle coverage as shown from simulation data.

Pentuim4 CPUs use traditional ucode-based BIST patterns to toggle CPU circuit including both array and random logic. While BIST can achieve decent array toggle coverage, it doesn't achieve enough toggle coverages on random logic. Thus, SBFT and FRITS patterns were later added to achieve reasonable toggle coverage for random logic. For a subsequent processor, even all of the above methods were not enough to achieve the desirable logic toggle coverage. Therefore, scan patterns were used to achieve the required coverage. In addition, to avoid the significant efforts of generating ATPG patterns for every step of silicon, random patterns, rather than ATPG patterns, were applied to random logic. This was done after thorough analysis and some experiments to demonstrate that the contention during BI is not detrimental for circuit following certain design-for-BI rules.

In contrast, PX uses an on-die WRPBIST logic to generate pseudo-random pattern and shift the patterns thru scan chains to achieve very high toggle coverage. Throughout WRPBIST-BI mode (Figure 5), no functional clock of targeted logic is triggered, thus, the potentially detrimental contention is totally avoided, which eliminate the circuit design rules to tolerate contention and avoid any lingering doubt on whether contention can actually impact long term reliability of circuit. It is also easy to generate WRPBIST BI patterns because the patterns only need to set up the essential control and are independent of logic content of the block to be toggled. SBFT and FRITs are reserved as the backup methods and they will be used only if WRPBIST-BI has unexpected issue or even higher toggle coverage for random logic is desirable.

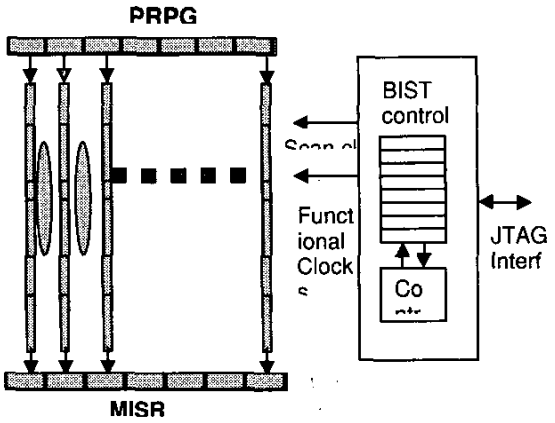

Figure 5 WRPBIST –BI Architecture

6. Design –for –Debug (DFD)

In the technology readiness phase, we had very active debate about either to treat Debug Scanout chains as part of the scan system. After many analyses, we decided to adopt the traditional design approach that completely separate scanout system from the ATPG scan chains. We allocated 10% of the full chip FFs as the signature cells and 1% scanout snapshot with an ability of performing system clock freeze and scan dump. The primary reason of this approach is to prevent the generation of excessive power due to scan at the system speed. The other key reason is to keep ATPG chains unpolluted by the scanout system. The detail of description of Intel micro-processor DFD features were described in [15].

7. Conclusion

The key contributions of the PX Microprocessor DFX team can be summarized as follows:

1. The PX DFX team has developed a highly competitive DFT and test strategy that enabled the first Intel full chip ATPG methodology with a very low scan overhead. By applying Skip Scan techniques and some innovative scan reduction methods, we were able to use ~52% of the full chip sequential and still remain nearly full chip ATPG (except NSBs) without scarifying the targeted performance, power and silicon budget. The total overhead estimated for scan DFT overhead is <3% of the chip area.

2. To overcome the issues of HVM power or Vcc Droop problems and the potential test data volume explosion, we developed a full chip partition ATPG structure and

methodology with a flexible scan DFT structure to enable the required clocking control logic. As a result, we have overcome the limitation of the commercial tools and manufacturing di/dt issues.

3. Novel Array DFT schemes were implemented to reduce approximately 50% of the total HVM test time for all arrays compared to earlier DAT scheme described in [11].

4. This is also the first Intel microprocessor that has a on-chip weighted random patterns BIST structure with a Hierarchical Structure including a test compression structure such s Illinois scan and X-Compact [17, 18, 19] The logic built-in self test structure also enabled the burn-in that saved significant efforts/resource in pattern generation and transformation.

5. In the DFD front, we addressed the HVM power issues by separating the scanout system and the traditional scan system. Scan for Debug techniques were explored by using system clock freeze features to take advantage of the large amount of scan sequential on chip.

In summary, the PX DFX team has developed a highly competitive DFT and test strategy that will serve as a BKM (best known method) for future Intel microprocessors to establish better microprocessor test methods that will significantly reduce product test cost and still achieving the highest product quality.

8. Acknowledgement

The authors would like to recognize the significant contributions provided by the PX microprocessor DFX team and the PX architecture and design teams throughout the entire project.

References

[1] Nandu Tendolkar, et. al. , "Novel Technique for Achieving High At-Speed Transition Fault Test Coverage for Motrola's Microprocessors Based on PowerPC Instruction Set Architecture" Pro of the 20th VTS'02

[2] C. Pyron et. al., "DFT Advances in the Motorola's MPC7400, a PowerPC Microprocessor",Proc IEEE ITC, 1999, pp 137-146

[3] R. Scott Fetherston et' al. "Testability Features of AMD-K6 Microprocessor", Proc. ITC 1997, pp 424-432

[4] M. Kruko et. al., "Microprocessor Test and Test Tool Methodology for the 500 MHz IBM S/390 G5 Chip", Proc. ITC 1998, pp717-726

[5] T. J. Wood , "The Test and Debug Features of the AMD-K7 Microprocessor", Proc. ITC 1999, pp130-136.

[6] L. Lai et. al. "Logic BIST Using Constrained Scan Cells", Proc. VTS 2004..

[7] Klenke, R.H.; Williams, R.D.; Aylor, J.H.; Parallelization Methods for Circuit Partitioning Based Parallel ATPG; Proc. of IEEE VLSI Test Symposium, 1993.

[8] Aguado, M.J.; Miranda, M.A.; de la Torre, E.; Lopez-Barrio, C.; A Dynamic Communication Strategy for the Distributed ATPG System DPLATON; Proc. of Design Automation Conference, 1993.

[9] Wolf, J.M.; Kaufman, L.M.; Klenke, R.H.; Aylor, J.H.; Waxman, R.; An Analysis of Fault Partitioned Parallel Test Generation; IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume 15, Issue 5, May 1996.

[10] K. Zarrineh, et. al, "Self-Test Architecture for Testing Complex Memory Structures," Pro. Int. Test Conf. pp. 547-556, Oct, 2000.

[11] A. Sivaram, et. al. "Efficient Embedded Memory Testing with APG," Pro. Int. Test Conf. p 47-54, Oct.,2002.

[12] P. Parvathala, K. Maneparambil and W. Lindsay, "FRITS – a microprocessor Functional BIST Method," Pro. Int'l. Test Conf., Pp. 590~598, Oct., 2002.

[13] A. Meixner and J. Banik "Weak Write Test Mode: an SRAM Cell Stability Design for Test Technique," Pro. Int'l. Test Conf. pp. 1043~1052, Nov. 1997.

[14] M. Tripp, T.M. Mak, and A. Meixner "Elimination of traditional Functional Testing of interface Timings at Intel," Pro. Int'l. Test Conf., Sept, 2003.

[15] A. Carbine and D. Feltham, "Pentium Pro Processor Design for Test and Debug," pp. 294~303, Int' Test Conf. Nov. 1997.

[16] J. Waicukaiski at. Al. "Transition Fault Simulation by Parallel Pattern Single Fault Propagation," Proc. IEEE ITC, 1986, pp542-549

[17] D. Wu; M. Lin, et. al. "H-DFT: A Hybrid DFT Architecture for Low-Cost High Quality Structural Testing," Pro. Int'l Test Conf. pp. 1229~1238, Sept. 2003.

[18] Mitra, S., and K.S. Kim, "X-Compact: An Efficient Response Compaction Technique for Test Cost Reduction," Proc. Intl. Test Conf., pp. 311-320, 2002

[19] Hamzaoglu, I., and J.H. Patel, "Reducing Test Application Time for Full Scan Embedded Cores," Proc. Intl. Symp. Fault-Tolerant Computing, pp. 260-267, 1999.