# L'estensione di STIL 1450 per la descrizione del flusso del programma di test

David Dowding
Agilent Technologies
Ernie Wahl
Agere Systems
Don Organ
Inovys

***Con la recente adozione dello standard IEEE 1450 STIL[1] e delle sue estensioni, l'industria dei semiconduttori può disporre di strumenti che garantiscono da un lato livelli di automazione più spinti e dall'altro contribuiscono a ridurre il tempo di introduzione sul mercato di nuovi prodotti***

Nella progettazione, collaudo e fabbricazione di semiconduttori, non c'è alcun dettaglio che non venga attentamente analizzato in termini di impatto economico sul costo finale del prodotto. Gran parte del costo di fabbricazione dei semiconduttori è imputabile alle spese complessive per sviluppo, impiego e mantenimento di programmi di test. L'automazione della progettazione della generazione di programmi di test ha contribuito a migliorare gli aspetti economici delle attività legate al design dei circuiti integrati. Negli ultimi tempi si sono fatti significativi passi in avanti circa l'automazione della generazione di modelli di prova per test sia funzionali sia strutturali. L'automazione della generazione di programmi di test è stata condotta all'interno delle maggiori organizzazioni che sviluppano programmi di collaudo; tuttavia, questa capacità è stata percepita come un vantaggio di tipo proprietario, che non è stato reso disponibile all'intero settore. Fornire un insieme di strumenti per la generazione di programmi di test standard soddisferebbe solo una parte delle esigenze, poiché ciascuna piattaforma ATE richiederebbe un generatore, un analizzatore e un interprete propri. La capacità di fornire una descrizione standard del flusso del programma di test è il modo migliore per favorire l'automazione della generazione di programmi di test. Questo è lo scopo dell'estensione P1450.4 allo standard STIL.

Parte delle difficoltà legate allo sviluppo di uno standard per il flusso del programma di test risiede nel fatto che i programmi di test sono utilizzati nel settore da moltissimo tempo. Ogni costruttore di semiconduttori ha sviluppato le proprie procedure e metodologie standard per gestire la sempre maggiore complessità dei dispositivi e per soddisfare le esigenze del suo specifico flusso di produzione. Tradizionalmente, i costruttori di ATE hanno sviluppato propri ambienti e compilatori di programmi di test e fornito metodi per facilitare l'utilizzo di tecniche specifiche da parte dei propri clienti, per ottimizzare il caricamento dei programmi e i tempi di collaudo, e per gestire le nuove risorse di collaudo a disposizione. Alcuni di questi metodi e tecniche sviluppati dai costruttori di ATE e di componentistica si sono diffusi nel settore fino a essere ampiamente adottati dalla maggior parte di essi. L'impegno profuso nello sviluppo dello standard P1450.4 ha condotto alla definizione di specifiche più ampie, nel tentativo di includere la maggior parte di tali metodi e procedure. Uno degli obiettivi primari dell'estensione STIL P1450.4 Test Flow è supportare le metodologie nuove e già esistenti, le tecniche consolidate e gli standard emergenti. Questa specifica è destinata a fornire una descrizione uniforme del flusso del programma di test che possa essere stratificato secondo le definizioni dello standard 1450 STIL. Esso dovrebbe fornire un meccanismo d'interscambio e uno strumento per una maggiore automazione tra gli strumenti di progettazione e le piattaforme ATE.

## Caratteristiche dell'estensione del flusso del programma di test

Il campo di applicabilità dell'estensione P1450.4 è molto ampio. La descrizione riportata di seguito illustra un gran numero di funzionalità previste dallo

standard. Non si tratta di un elenco esaustivo ma di una sintesi delle metodologie, delle tecniche e delle strutture dati attualmente previste in questa estensione.

**1.** Garantire il completo allineamento con STIL 1450.0 e le sue estensioni: per tale motivo sono stati mantenuti lo stesso ordine e le stesse strutture dello standard 1450-1999 STIL. Questo consente di estendere la stessa metodologia di analisi ed elaborazione di STIL ai dati relativi al flusso del programma di test [2]. Con la proposta P1450.4, vengono aggiunte nuove strutture di dati e parole chiave per i flussi di programma, ma l'elaborazione dello STIL attuale non richiederà cambiamenti comportamentali. Proprio come con lo standard di base, le strutture dell'estensione P1450.4 di più alto livello vengono bloccate al livello più alto di STIL. Come nel dominio STIL, è possibile fare riferimento ai blocchi denominati: quelli non denominati sono considerati come "predefiniti" in ragione della loro tipologia e i dati in essi contenuti usati ogniqualvolta una struttura del loro tipo rientra nel suo campo di applicabilità. Ciò è identico al comportamento di risoluzione dei dati STIL per i blocchi denominati e non denominati [3]. Per esempio un blocco SignalGroups rientra nel campi di applicabilità quando vi rientra la sua struttura di riferimento. Si sta considerando di incorporare le strutture di Livello IEEE 1450.2-2001 DC nelle opportune strutture di flusso in modo trasparente [4]. Inoltre, l'estensione P1450.4 prevede ove possibile di usare e sfruttare le strutture P1450.1 e P1450.6 (EDA e Core Test Language).

**2.** Fornire l'organizzazione e il controllo della sequenza di programmi: gli odierni programmi di test spesso non sono soltanto flussi lineari di righe di testo. Alcuni hanno centinaia di sequenze. Uno degli obiettivi di questa estensione è fornire un metodo efficace per lo sviluppo e il mantenimento di programmi di test con numerose sequenze di testo. I test di raggruppamento gerarchico in sequenze di "sottoflussi" e la stratificazione di tali gruppi in ulteriori "sottoflussi" fa parte dell'approccio che è stato definito.

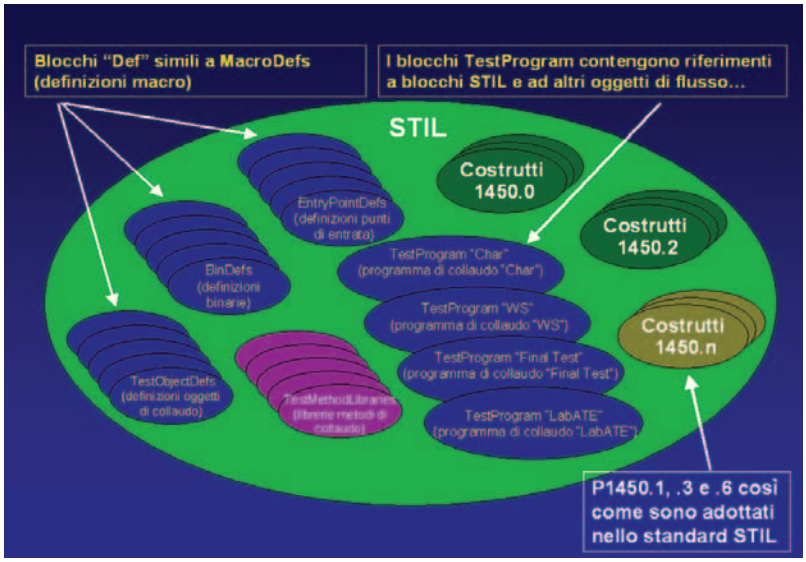


**Fig. 1 - Programma di collaudo basato su STIL**

Questo approccio gerarchico fornirà il supporto per le librerie di test riutilizzabili e per le sequenze di test. Questa estensione crea strutture che consentono il supporto di tale gerarchia. Un determinato punto nella descrizione del flusso può rappresentare una singola chiamata a un TestMethod o un completo sottoflusso di più chiamate a TestMethod con diramazioni e salti nel flusso in funzione dei risultati di un test.

Il punto di vista a livello più elevato del programma di test può essere rappresentato da un singolo "FlowNode", un test che può essere "forzato" al fine di rilevare livelli inferiori di sequenze di flusso che infine daranno una condizione di "Passa" (Passa) o "Non passa" (Fail) con assegnazioni a contenitori (bin) "hard" e "soft". La rappresentazione grafica dei dati di flusso sarà lasciata all'implementazione di strumenti operanti su questi dati. I dati stessi conterranno la struttura e il meccanismo di riferimento che permetterà a un tecnico di collaudo che è a conoscenza delle specifiche dell'estensione di avere una visione concettuale intuitiva del flusso del programma.

**3.** Promuovere il riutilizzo dei componenti dei programmi di test: come affermato in precedenza, è possibile denominare i blocchi del flusso di programma di prova. Questi blocchi di test possono essere definiti in modo da costituire entità di collaudo complete ed essere rintracciabili grazie al nome del loro dominio. Una volta collocato questo oggetto in un determinato flusso, esso necessita esclusivamente di risorse di connessione all'interno del flusso stabilito (per esempio nel punto della sequenza in cui si trova e nella direzione in cui procederà il controllo del flusso di programma in seguito al test). Anche le azioni del sistema di test in tempo reale e quelle pilotate dalle interruzioni dell'apparecchiatura ATE possono essere definite e denominate per il riutilizzo in vari programmi di test.

**4.** Modularità della sequenza di test e del test di supporto e organizzazione del programma gerarchico. Oltre alle sequenze di test più denominate e richiamate più volte, le sequenze di test possono essere incapsulate in un blocco di "sottoflusso" e denominate.

Le librerie di queste sequenze di test sono utilizzabili per contenere la metodologia standard di test per tipologie di dispositivi o famiglie di dispositivi.

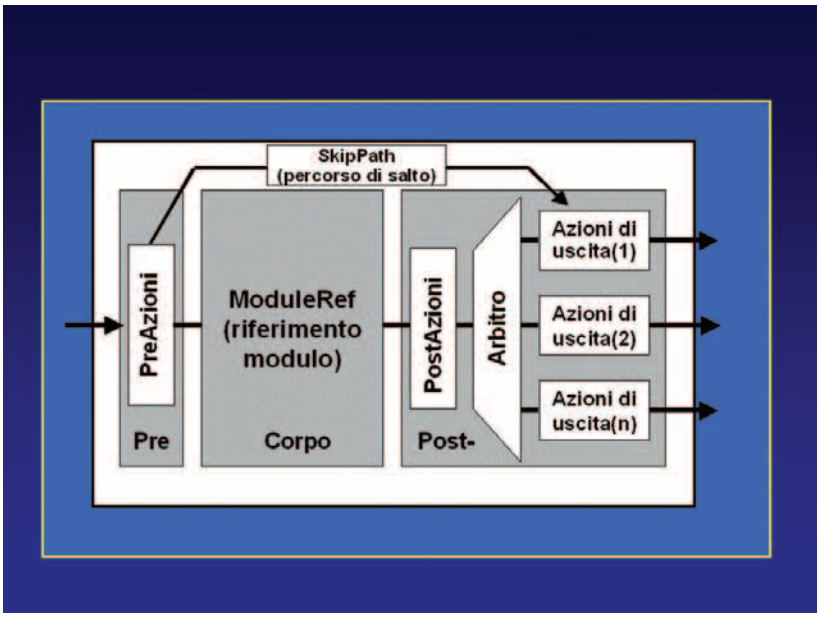


**Fig. 2 - Schema a blocchi dei nodi di flusso**

Queste possono poi essere riutilizzate per costituire un flusso di programma di test. I blocchi di sottoflusso sono richiamabili più volte nei programmi di test e possono essere riutilizzate in diversi punti di questi ultimi.
L'estensione supporterà diversi programmi di test completamente specificati nella sequenza di un determinato file STIL. Ogni programma sarà contenuto all'interno del proprio blocco TestProgram. Ciascuno di questi ultimi disporrà di proprie azioni di sistema e di interruzione del test (per esempio OnStart, OnReset, ecc.). Ciò consentirà alla descrizione di un singolo System-on-Chip (SoC) di distinguere le descrizioni dei programmi di test per Wafer-Sort, QA/PackageTest, ecc. all'interno di un singolo file STIL. I moduli all'interno di questi potranno essere incapsulamenti di una sequenza completa di test, con la sola differenziazione della risoluzione del segnale per l'hardware di prova e l'assemblaggio dei dispositivi.
**5.** Meccanismi di controllo Multi-Site: la metodologia Multi-Site assicura una migliore utilizzazione del tester e una migliore gestione dei costi associati al test. L'impatto sul flusso dei programmi di test può essere affrontato nel sistema operativo del tester con scarsi effetti sull'impatto del programma di test. Tuttavia, l'allocazione delle risorse del tester e le dipendenze dell'ordine di esecuzione hardware possono generare l'esigenza di passare attraverso più siti di prova eseguiti simultaneamente in modo da poter commutare a un'esecuzione seriale locale. Inoltre, la sequenza del flusso di test può richiedere speciali considerazioni inerenti al flusso. Tali "eccezioni" possono essere anticipate nel flusso del programma di test. Il flusso del programma di test deve fornire le istruzioni del tester per il corretto funzionamento del flusso in relazione alle condizioni di eccezione.
Questa estensione ha identificato molte di queste situazioni del flusso e fornisce meccanismi per specificare le direttive di flusso alternative. Si tratta di un funzionamento complesso e l'estensione può essere limitata nel coprire le situazioni basilari, e affidare le altre all'implementazione dei controlli di sistema di determinate piattaforme di tester.
**6.** Strategie di binning generico: la maggior parte degli ATE prevedono il concetto di contenitori (bin) "hardware" e "software". I contenitori hardware si riferiscono a sonde/handler o alla categorizzazione dei dispositivi in prova. Tipicamente esiste un numero finito di contenitori hardware disponibili sulle apparecchiature di prova ausiliarie. Potrebbero esistere diversi contenitori software associati a un contenitore hardware. I contenitori software vengono tipicamente usati per specifiche di risultati di prova superiori. Per esempio, una specifica classe di test può contenere diversi test all'interno del relativo programma per verificare tutti i dettagli e tutti gli aspetti dei parametri dei dati tecnici relativi a quel test. Una o più classi di test potrebbero usare lo stesso contenitore hardware, ma i vari test potrebbero avere ciascuno un proprio contenitore software per agevolare l'isolamento e la registrazione dei risultati delle prove sui dispositivi.
La mappatura dei test in relazione ai rispettivi contenitori software e hardware viene chiamata "mappatura dei contenitori" (bin map). Questa estensione definirà i meccanismi di contenimento e fornirà le strutture di supporto per l'uso di tali meccanismi. Inoltre, l'uso di test e sequenze modulari ne consentirà di conseguenza il riutilizzo con la corretta mappatura di contenimento.
**7.** Allineamento con l'ordine di esecuzione dei test a partire da strumenti EDA come definito nel linguaggio Core Test Language P1450.6: le strutture del CTL P1450.6 usano il blocco Environment P1450.1 per localizzare le proprie strutture CTL di base[5]. Unitamente ai meccanismi di rintracciabilità di settore P1450.6, è possibile incapsulare una determinata "metodologia di test" per ciascun nucleo supportato da CTL. Ciascun nucleo può avere diverse metodologie, come test di produzione, diagnostica, ecc.
Il programma di test combina e ordina i test e le metodologie di test dei vari nuclei con la sequenza di test e le informazioni sulla dipendenza fornite dal progettista SoC. Le strutture di flusso di P1450.4 possono essere utilizzate per allineare e disporre in sequenza i test di nucleo all'interno del programma di test complessivo. I provider di nuclei e i progettisti di dispositivi SoC utilizzeranno questa estensione per ordinare la sequenza di test e fornire maggiori informazioni sul contenuto dei test al gruppo di sviluppo di quest'ultimo. Si

intende che tale estensione, ove possibile, realizzerà l'allineamento con altre estensioni STIL per promuoverne l'utilizzo nonché le possibili automazioni.

**8.** Fornire conferme e interruzioni al sistema ATE: le conferme (assertion) e le interruzioni (interrupt) asincrone e in tempo reale di un sistema ATE possono variare da una piattaforma di tester all'altra. Per tale motivo potrebbe rivelarsi necessario realizzare specifiche interazioni e configurazioni di dispositivi con sequenze di applicazione di risorse del tester differenti. In passato, alcuni di questi test speciali sono stati inseriti nel programma di prova e altri sono stati inseriti all'esterno del programma, anche se associati ad esso. Questa estensione fornisce un insieme di compiti di tipologia generica relativie alle conferme e alle interruzioni, come OnLoad, OnStart, OnReset, ecc. Può trattarsi di compiti globali di una tipologia e/o possono essere nominati compiti associati a una tipologia. Il programma di test può designare sequenze di compiti speciali o usare descrizioni globali dell'utente. Inoltre, l'utente potrà definire compiti User-Defined e associarli alle asserzioni della piattaforma di test. Questo approccio fornisce al flusso del programma di test la completa descrizione di tutte le attività e le sequenze all'interno e all'esterno del tradizionale flusso di programma necessario per il controllo del tester.

**9.** Fornire un'interfaccia strutturata per le comunicazioni con TestMethods: questa estensione allo standard STIL è incentrata sul flusso di programma e non definisce l'effettiva sequenza di test o "TestMethod". Per fornire l'indispensabile coordinamento tra il programma di test e l'attivazione, la valutazione dei risultati di prova e la risposta del flusso, sarà descritta un'interfaccia tra il flusso e il test. La descrizione di questa interfaccia fornirà un protocollo di comunicazione dati chiaro e sufficiente per trasferire i parametri dal flusso a TestMethod e quindi ricevere in cambio i risultati dal test sul flusso per il corretto tracciamento della risposta di quest'ultimo. Una visione semplice di questa interazione è analoga ad un richiamo di funzione software di trasferimento dei parametri e per indicare il successo o il fallimento della funzione viene fornito in risposta un codice risultante. Oltre al risultato di Pass o Fail, alcuni TestMethods possono ritrasferire al flusso di programma di test un risultato che può essere un valore misurato o una serie di risultati.

Le interfacce da e verso i TestMethods sono accompagnate in questa estensione da azioni post-test che possono essere basate sul(i) risultato(i) di prova e da un "arbitro" che valuta le azioni opportune, come le assegnazioni di contenimento e le impostazioni dei dispositivi per concludere il test, o che prepara il dispositivo per la successiva sequenza di test del flusso, e concludendo con l'opportuno flusso di salto condizionale da tale sequenza di test.

**10.** Supporto di molteplici comportamenti di test in un singolo programma o di molteplici programmi di test in una singola sequenza di test STIL: come affermato in precedenza, questa estensione fornisce i meccanismi e le strutture che hanno natura modulare. Il tracciamento di specifiche strutture "denominate" può scomporre il dispositivo e le risorse del tester e la capacità di connessione per fornire test da eseguire su una varietà di apparecchiature ausiliarie, e per fornire sequenze di test riutilizzabili e configurate in base alle impostazioni del dispositivo o dell'hardware. Per esempio, le sequenze WaferSort e PackageTest possono variare sui segnali disponibili ai confini del dispositivo. Alcuni test possono essere eseguiti in entrambi i casi con l'unica preoccupazione della risoluzione dei segnali limite. In altri casi, il comportamento della sequenza di test varierà in base al tipo di condizione di quest'ultimo. In questo l'estensione fornisce allo sviluppatore del test la flessibilità per affrontarne il differente comportamento all'interno del TestMethod o delle strutture del suo flusso di programma.

Questo fornisce la flessibilità di avere più programmi di test in altrettante sequenze di file STIL, o i comportamenti di tutti i programmi di test in un singolo programma di test in una singola sequenza di file STIL, oppure alcune variazioni intermedie.

Altre caratteristiche non descritte nei paragrafi precedenti comprendono l'uso di variabili di configurazione di sistemi di test standard definite all'esterno dell'insieme equazione/specifica STIL (come gli indicatori di stato e di caricamento del programma), variabili runtime per il controllo dei programmi e del sistema e insiemi di variabili Multi-Site per supportare il comportamento delle varianti runtime. Esiste inoltre una specifica che descrive le sequenze di testo "interne" particolari per un flusso di test e non denominate o non riutilizzabili. Queste e altre capacità sono state sviluppate in questa estensione per garantire la massima flessibilità nella definizione del flusso del programma di test e per il suo controllo.

**RIFERIMENTI BIBLIOGRAFICI**

[1] IEEE Computer Society, IEEE Standard Test Interface Language (STIL) for Digital Test Vector Data Language Manual IEEE Std. 1450.1999. IEEE New York, September 1999

[2] Gregory A. Maston, "Structuring STIL for Incremental Test Development", Proceedings of the International Test Conference, 1997

[3] Tony Taylor, Gregory A. Maston, "Standard Test Interface Language (STIL) - A New Language for Pattern and Waveforms", Proceeding of the International Test Conference, 1996

[4] IEEE Computer Society, IEEE Standard for Extensions to Standard Test Interface Language (STIL) - (IEEE Std. 1450.1999) for DC Level Specification. IEEE New York, September 1999

[5] Tony Taylor, "Standard Test Interface Language (STIL), Extending the Standard", Proceeding of the International Test Conference, 1998

*Tratto dall'articolo originale: "Extending STIL 1450 Standard for Test Program Flow". Si ringrazia l'organizzazione IEEE per la collaborazione*

**readerservice.it**

| | |
|---|---|
| Agere Systems | n. 23 |
| Agilent Technologies | n. 24 |
| Inovys | www.inovys.com |