A UNIVERSAL AUTO TEST PROGRAM GENERATION ON ADVANTEST V93000 ATE PLATFORM

Xin Song[1*], *Yefang Wang* [1], *Hanyan Chen*[1]
[1]Advantest (China) Co. Ltd, Shanghai 201203, China
*Corresponding Author's Email: Nick.Song@advantest.com

2023 China Semiconductor Technology International Conference (CSTIC) | 979-8-3503-1100-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/CSTIC58779.2023.10219237

ABSTRACT

The semiconductor industry is in a period of rapid growth. 5G, IOT and HPC/AI need to keep up with the changing request of the consumer market. As a core component, IC accelerates the speed of upgrading and iteration.

ATE test is an important part of production. The test program development is the core of the whole process. How to improve the efficiency of test program development is the concern of many Fables at present. Therefore, ATPG (Auto Test Program Generation) has been urgent request of many febless.

Based on the principle of ATPG, this paper introduces the relevant requirements analysis, framework design and result. Based on the V93000 platform of Advantest, the standard specifications of ATPG are shown.

Keywords—ATPG, ATE, V93000

INTRODUCTION

ATPG Overall

ATPG (Auto Test Program Generation) is a way to improve test program development efficiency. Test program is divided into "Test Flow", "Pin Config", "Level Set" and "Timing Set" and "Test Method" on V93000 platform.

TABLE I. TEST PROGRAM STRUCTURE

Program Part	*Description*
Test Flow	Test suite execution sequence
Pin Config	Test resource assignment on ATE
Level Set	DC set of Power and I/O
Timing Set	AC set of I/O
Test Method	Test resource control

Actually, "Test Method" is code base and too complex and flexible. That means it is difficult to auto generate. The other four part could implement with ATPG solution.

The ATPG solution is a set of python scripts that correspond to four V93000 program part. Each script has input file and output file interface. User obtains the generated V93000 program module by configuring the input file and executing the script.

ATPG Flow

The ATPG test flow solution could generate ATE test flow on Smartest 7 environment on V93000 platform. User needs a test flow configuration file which could describe test flow information. The information could be divided 3 parts.

- Flow Information: Define flow name, global variables, and various files (pin, level, timing and pattern) to be loaded.
- Testsuite Information: Define information about testsuite in flow. Such as suite name, suite sequence, level and timing used by suite.
- Testmethod Information: Define relevant parameters of test suites.

Based on this information, we can complete the information collection of ATE test flow.

ATPG Flow -- Framework

This ATPG solution design a framework to achieve input file reading and storage, flow information process and flow file generation. Input file is flow configuration file and could be red by ATPG solution. Then the storage information will be used to sperate some items as V93000 flow file structure (Table II). After reassembly, the generated test flow can become recognized by V93000.

TABLE II. V93000 FLOW STRUCTURE

Flow Part	*Description*
Flags	Global variable definition
Testmethodparameters	Parameter for test
Testmethods	Test method file to be called
Testsuites	Level/Timing and local flag
Testflow	Flow sequence
Binning	Hardware Bin setting
Context	Other files associated with Flow

The specific calling relationship is shown in figure 1.

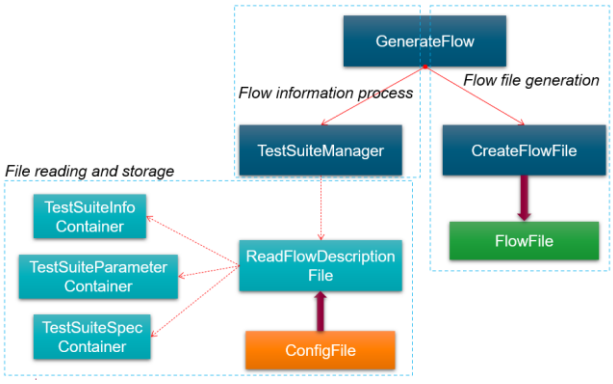

Figure 1: ATE Flow Framework

ATPG Flow – Interface

The ATPG solution's interface is also very simple. User just specify the config file path and click "run" button then get the generated flow file.

Figure 2: ATPG Flow solution interface

ATPG Pin Config

ATE Pin Config generation costs at least one day manually, especially for AI/HPC device, the process will cost more time due to massive channels. ATPG solution can improve setup efficiency and has some advantages as below.

- ATPG Pin Config solution can help generate pin setup files which is suitable for Smartest 7 of Advantest [1].
- The solution processes digital channels, analog channels, DPS channels, utility channels, groups definitions.
- This solution processes multisite pin files generation and site deleting.

ATE Pin Config -- Framework

Framework works like building blocks for users to get V93000 pin config files.

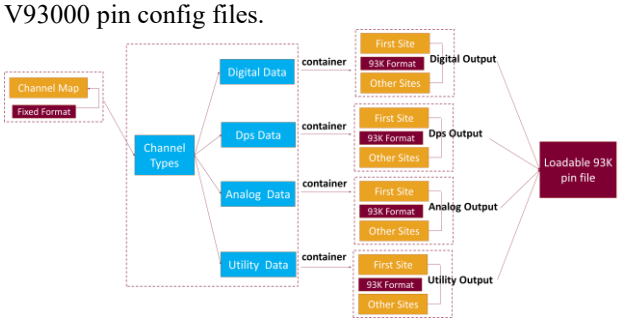

Figure 3: ATE Pattern Iteration Process

As expected, we get LB vendor channel map firstly. ATPG solution analyzes channel map and process key information and stores it in different data structures. Combined with the V93000 unique string, preserved channel information in data structures will be translated into V93000 pin file by ATPG solution.

ATE Pin Config -- Channel Map Description Table

LB vendors always provide channel map file for resource definitions of tester channel. But the format is not uniform. So, we design a common simple template which could be accepted by different PCB vendors. (Figure 4)

Pin Num	Name	Net Name	SITE1	SITE2	SITE3	SITE4	PIN_GROUP1	PIN_GROUP2	PIN_GROUP3
A1	trig_awg1	231_S2_CT1	D11215	D11215	D11215	D11215	g_digital		
A2	trig_dgtz1	431_S2_CT1&432_S2_CT1	D11216	D11216	D11213	D11213	g_digital		
A3	MCB1_ABCD+	xxx	431_C-	431_C-	432_C-	432_C-			
A4	MCB2_ABCD-	xxx	431_C+	431_C+	432_C+	432_C+			
A5	MCB3_EFGH+	xxx	231_E+	231_G+	231_F+	231_H+			
A6	MCB4_EFGH-	xxx	231_E-	231_G-	231_F-	231_H-			
A7	GPIOA	xxx	D22208	D22206	D11701	D10716	g_digital		
A8	GPIOB	xxx	D42302	D22016		D10808	g_digital		
A9	GPIOC	xxx	D22309	D22109		D11107	g_digital		
A10	DPS1	xxx	430_P9	429_P6	429_P9	225_P9			
A11	DPS2								
A12	DPS2								
A13	DPS2								
A14	DPS2								
A15	DPS2	xxx	430_P11-P14	429_P12-P15	429_P01-P04	425_P01-P04			
A16	DPS2								
A17	DPS2								
A18	DPS2								
A19	DPS2								
A20	DPS3	xxx	430_P5	230_P11	229_P11	225_P16			
A21	K1	xxx	4URW15	4URW1	4URW2	4UW7			
A22	K2	xxx	4UW13	4UW13	4URW3	4URW3			
A23	K3	xxx	5UW0	5UW0	8URW15	8URW15			

Figure 4: Channel Map File Template

User must follow some filling rules which could be realized by ATPG solution. Such as Digital pin should start with "D", Power pin should include "P" and so on.

ATE Pin Config – Data Storage & Generation

The specific V93000 format of each channel in detail is shown in Figure 5. There are specified channel information like Dps, analog, digital, trigger pins in Smartest 7.

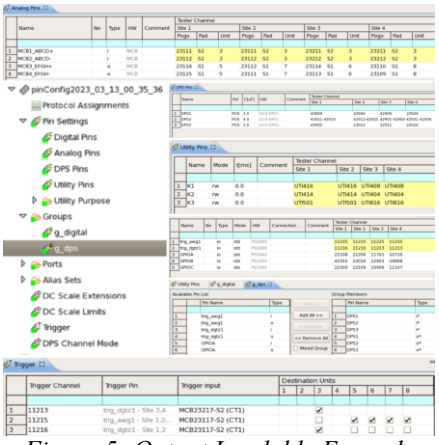

Figure 5: Output Loadable Example

The above information in Figure 5 is actually stored in Smartest as text information shown below in Figure 6, which can be generated by the ATPG solution to achieve rapid offline generation and modified by the ATPG solution to match the ATE resources.

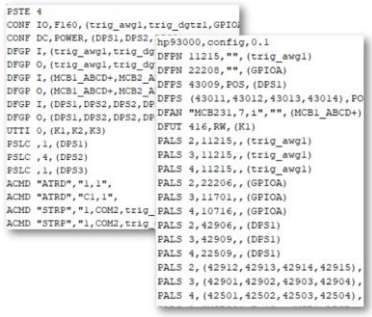

Figure 6: Output Text View Example

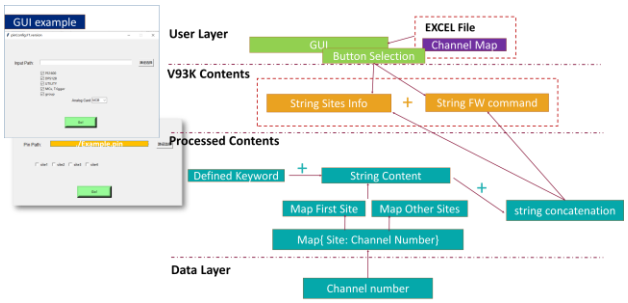

Figure 7: Data Container Design

As shown above in figure 7, a single common data container includes information like channel number and keywords. For different channel types, independent data containers will be added to be more compatible.

When the test program pin config does not match that on OSAT, ATPG solution can automatically modify the pin config file to achieve matching.

ATPG Level – Input File

For V93000 program, the level setup assigns values for digital I/O pins and power supply voltages to DUT. It can be organized as one single file, or its individual level equation sets and test data blocks can be stored in separate files [2]. In the latter case, "master file" is a file to manage all level files. For this solution, we use "master file" to organize different level files as figure 8 shown. Therefore, the input file is in excel format and it's designed that one sheet corresponds to one equation set, which requires three modules: DPS module, level sets module and specs module.

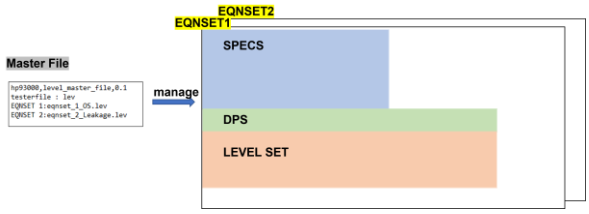

Figure 8: Master File Manages Level Files

DPS module uses expressions(variables) to specific the DPS pins levels and parameters of the DPS pins resources to define the behavior of DPS pins. Normal resource parameters can be grouped by functions as voltage, current, connections and timings, voltage settling time and voltage slew rate.

Level sets module assigns values to the level resources of I/O pins using expressions. It includes two parts: pin assignment and its feature such as vih/l, voh/l and ioh/l.

Specs module links the spec variables and its units used in the DPS module and the level sets module. We also merge the values of the spec variables which are defined in spec set in this module. The relationship between these three modules is shown in figure 9.

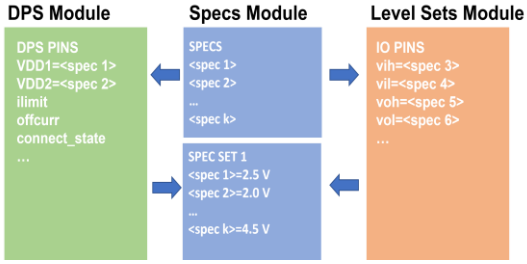

Figure 9: Three Modules Relationship

ATPG Level -- Data and Storage

After all the level related information is filled in excel, this solution programmed in python will process the information and generate the corresponding level files.

First, the entire excel will be analyzed by "ReadLevelDescriptionFileTotal" class, and "ReadLevelDescriptionFileBysheet" class will process the content of each sheet page separately and create a corresponding "sheet container". And one "sheet container" includes three types of containers, which are "specsetcontainer", "DPSsettingcontainer" and "levelsetcontainerDic". "levelsetcontainerDic" manages multiple "levelsetcontainer", because in V93000 program, one equation set may contain multiple level sets. After processing all the sheets, we will get a "sheetContainerDic".

Then "ReadLevelDescriptionFileBysheet" class splits and stores the settings of the spec set, DPS set and level set into the corresponding containers through various fill functions, as shown in figure 10.

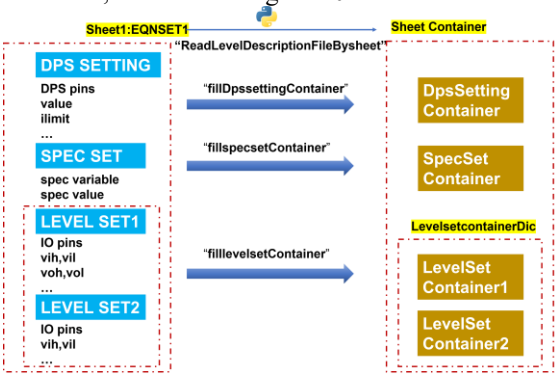

Figure 10: Data Classification and Storage

AFPG Level --File Generation

After storing all parameters to corresponding containers, we need to retrieve and print them based on the format requirements of V93000 level file. The structure of an equation-based level setup includes three key parts: Specs module, DPS module and level sets module. Figure 11 below illustrates the structure of an equation-based level setup.

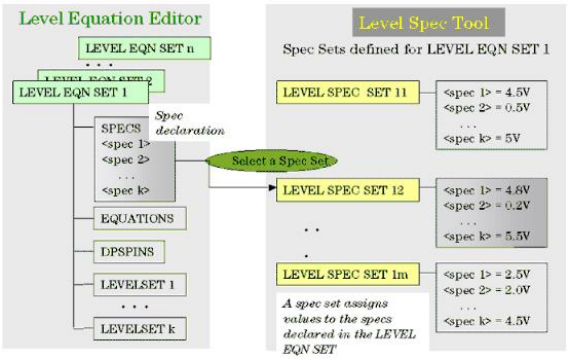

Figure 11: Structure of Level Setup

As mentioned above, each container provides a filling function to store the corresponding level settings. It also provides access function as an interface to get the settings of each container. "GenerateLevelBysheet" class iterates through the contents of each container in sequence, first setting DPS set, then level set and then spec set. Next the contents are printed to level file in turn, as shown in figure 12. After all level files are generated, "GenerateLevelTotal" will generate a master file to manage all of them.

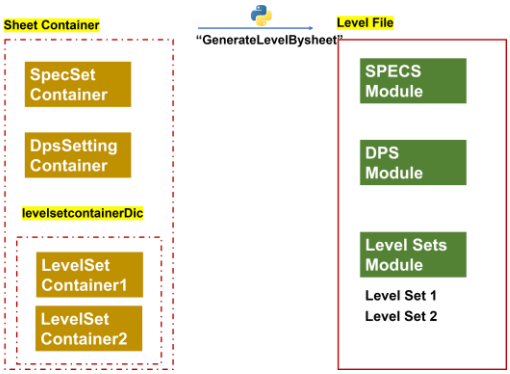

Figure 12: Level File Generation

APTG Level -- TMS

For SoC (System on Chip), different power supply domains have different power up and down sequence requirements. ATPG should consider how to define parameters to represent different power up/down sequences.

In V93000, to specify setup times and delays for DPS pins, it provides two options: DPS connect setup time or voltage ramp. But it can't set within the same set in one DPS set of one channel [3]. If only power up/down sequence is considered, rather than the power up/down slope, we can only use the "t_ms "parameter to generate a graph showing the DPS pin directly power up/down after "t_ms" time, as shown at the top of figure 13. Alternatively, the voltage ramp parameter is used to specify setup times and slew rates separately for rising and falling DPS voltage transitions. As the bottom of figure 13 shown, to facilitate user understanding, we convert original voltage ramp parameters (vout_rise_settling_t_ms, vout_rise_t_ms_per_volt) of V93000 into "up_starttime" and "up_waittime". "up_starttime" represents total time of pin voltage rising and waiting. And the same applies to power off. With this function, the user does not have to understand V93000 up and down settings but can double check by simple graphical comparison.

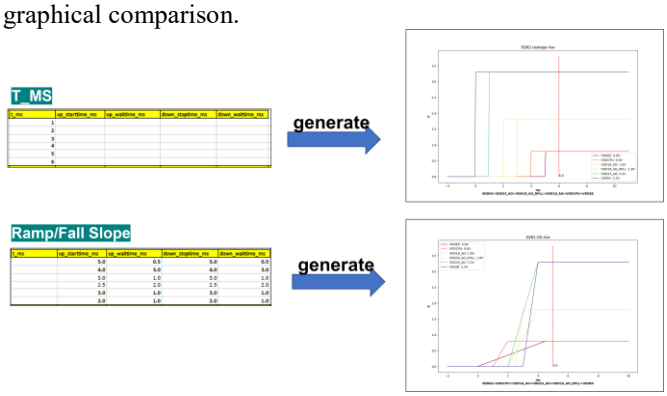

Figure 13: Power Up/Down Diagram

APTG Level – Interface

The final interface of ATPG level is shown in figure 14. User only need to fill in the input excel and load it; the level file can be automatically generated.

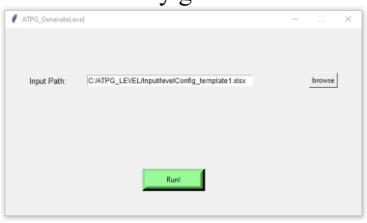

Figure 14: ATPG Level GUI

APTG Timing

The automatic generation of timing is relatively less complex. Because timing is bound to pattern. Using various pattern tools, when generating patterns, the corresponding timing will also be generated.

However, the number of timings generated can be very large, ranging from tens to hundreds, so it is necessary to contact the "master file" feature of the V93000 to organize them [4]. The master file is shown in figure 15.

```
hp93000,timing_master_file,0.1
testerfile: demo_project.tim

EQNSET 1 :DC/test500.tim
WAVETABLE "test500" :DC/test500.tim

EQNSET 1 [Remap 2]:DC/test501.tim
WAVETABLE "test501" :DC/test501.tim

EQNSET 1 [Remap 3]:DC/test502.tim
WAVETABLE "test502" :DC/test502.tim
```

Figure 15: V93000 Timing Master File

In this figure, there are three timing files, each file has

"equation set" and "wavetable set". A set of "equation set" and "wavetable set" correspond at least one pattern.

Each "equation set" has unique number so some "equation set" number will be remapped to other number. Usually, the timing file is classified and placed under different folders.

Based on these features, ATPG Timing solution achieve three core features.

- Auto generate timing master file
- Auto remap equation set number
- Auto find available timing file and record path

Besides this, ATPG solution also consider user will continue to add new timing file that means timing master file could be auto refreshed.

APTG Timing -- Framework

ATPG Timing solution has two key features: create new master file and refresh exist master file. "GenerateMasterFile" is main class. It controls "CreateMasterFile" class and "RefreshExistFile" class.

"TimingFileParser" class is used to parse all timing files and record file informations. "TimingFileManager" record file name to file list.

"RefreshMasterFile" class is used to refresh exist timing master file. All information in master file will be reserverd in "MasterFileInfo" class. The unused timing files will be checked and move to backup folder by "MoveTimingFile2Backup" class.

The details framework of ATPG timing solution is shown in figure 16.

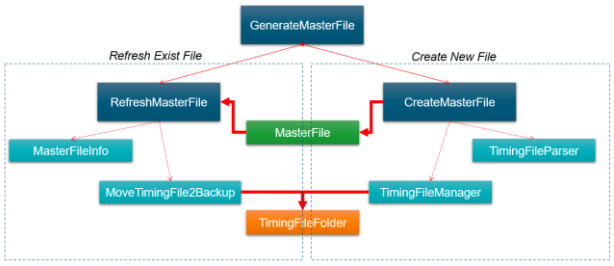

Figure 16: V93000 Timing Master File

ATPG Timing – Interface

The ATPG Timing solution's interface is as below. It could achieve "create new file" and "refresh exist file" automatically.

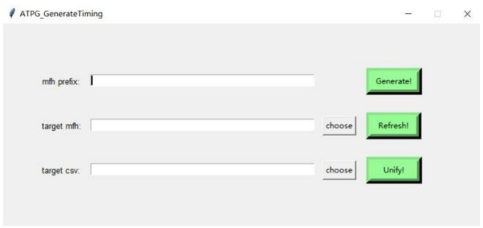

Figure 17: ATPG Timing solution interface

CONCLUSION

This ATPG solution summarize a universal input information and shows high efficiency. It is based on V93000 platform and could be used for major ICs.

ACKNOWLEDGEMENTS

We would like to express our gratitude to Beryl Xu, whose brilliant ideas and perceptive observations have proved immensely constructive.

REFERENCES

[1] ADVANTEST. *Test Documentation Center: Pin configuration file*, Rev7.5.4, topic. 94263.

[2] ADVANTEST. *Test Documentation Center: Level setup master file*, Rev7.6.4, topic. 91598.

[3] ADVANTEST. *Test Documentation Center: DPS connect setup time and voltage ramps*, Rev7.6.4, topic. 125205.

[4] ADVANTEST. *Test Documentation Center: Master file format for timing setup*, Rev7.5.5, topic. 91498.