COUNT TRIANGULATION: SYSTEM THAT SUBSTANTIATES TEST COUNT FROM TESTER, HANDLER, AND PHYSICAL UNIT VIA DATALOG, API COMMAND AND OTSR, TO TRAP UNIT MIXING IN SEMICONDUCTOR FINAL TEST MANUFACTURING

Neilvin S. Remo
Ralp Lyndon M. Manalo
Reandro M. Lim

Product and Process Engineering
Microchip Technology Operation (Phils) Corporation
102 Accuracy Drive, cor. Excellence Avenue, Carmelray Industrial Park 1,
Canlubang, Calamba City 4028 Philippines
Neilvin.Remo@microchip.com, RalpLyndon.Manalo@microchip.com, Reandro.Lim@microchip.com

ABSTRACT

In Semiconductor Final Test manufacturing, the integrity of test counts is a critical measure of an effective testing process. Failure to maintain strict oversight can lead to issues, including the inadvertent mixing of units, which may go undetected. Such oversights can result in customer complaints, rescreen of materials, product scrappage, and ultimately, the risk of jeopardizing the business.

In response to the growing need for quality, efficiency, and reliability, it has become essential to enhance automation within the testing and manufacturing processes. Beyond the conventional method of using tester-derived test counts and physical unit counts in the test count summary structure, this paper provides the framework for the adoption of Count Triangulation under the test manufacturing environment, which effectively utilizes handler test count in the whole validation chain. Using Application Programming Interface (API) commands, handler count can be harvested automatically to allow a three-point validation system that would further enhance the anti-mixing controls. The Count Triangulation system will aggregate data from three independent sources, published in Online Test Summary Report (OTSR), to comprehensively assess the accuracy of counts and detect any instances of potential mixing, mis-binning, or mis-handling events. OTSR webservice will provide an automatic assessment of count integrity and link with Manufacturing Execution System (MES) to manage the movement or holding of lots. This system will provide a safeguard to detect possible incidents of mixing.

It is noteworthy that although certain handlers are equipped with built in API commands, the knowledge of its presence is not widely known which consequently leads to its underutilization. This paper, as highlighted by the authors, provides one of the ways to utilize API commands through presenting a method of automated retrieval of handler counts.

1. 0 INTRODUCTION

Count Integrity is one of the considerations for device handling in the final test semiconductor manufacturing. The testing process must be able to show, with evidence, that the resulting good units are supported with objective data to guarantee the accuracy of the entire process. If neglected, detection of misprocessed and potentially mixed units might escape the process. It would lead to serious consequences such as customer complaints, rescreen of material, and loss of business trust. In the shifting quality standards, the direction to reinforce quality controls from manual to automated has become a necessity, simply because it is more reliable, trustworthy, and sustainable. On the current process, we are relying on tester summary and physical count to assess if there's a mixing occurrence. However, in case there is an issue with the handler (i.e. sorting, jams), the current process may not be enough to easily capture if there is a mixing event. Through Count Triangulation, as shown in Fig. 1, the handler count is added through API. The use of API will allow the automated harvesting of handler count. Together, these values are compared to the tester count, and physical count which are all published in OTSR.

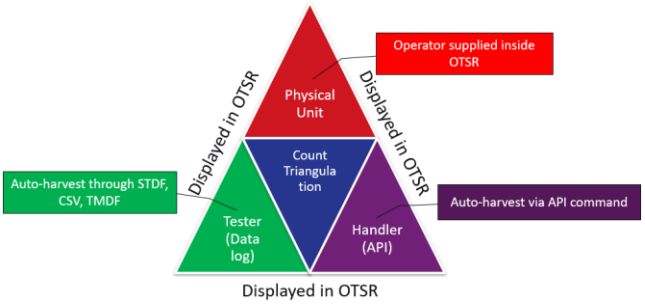

Fig. 1. Count Triangulation Illustration

Afterwards, the OTSR webservice will compare these data to make an objective analysis to see if the good units that will pass on onto the next step are safe from any mixing, mis-

binning, or mis-handling events. The OTSR webservice will link with MES to hold or to move a lot. The whole Count Triangulation process is shown in Fig. 2.

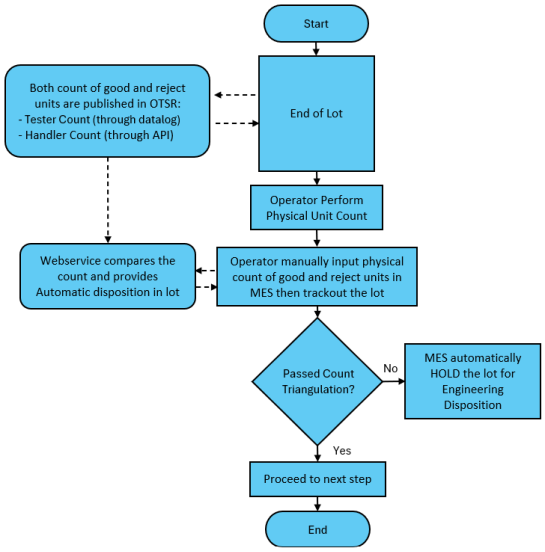

Fig. 2. Count Triangulation Process Flow

This study used final test gravity fed handlers with API commands accessed via GPIB communication to harvest handler count. It includes checking that the handler count fetched from the API is aligned with the actual handler count sorted. Different handler events were also observed and recorded to verify the effectiveness of Count Triangulation.

1.1 Application Programming Interface (API)

An API is a set of rules that define how a software program can request and receive information from other software. API used in this paper is based on IEEE 488.2 commands and their syntax which enables communication between Final Test handler and tester as shown in Fig. 3.

Fig. 3. API Communication between handler and tester

2. 0 REVIEW OF RELATED WORK

As stated in a case study on Application Programming Interface (S. Pawar, 2023), API allows communication between different software or computers. In this way, information, and functionalities of business application from internal departments, business partners and third parties can be acquired. Modern equipment used at final test semiconductor today already has existing API commands which can be used to collect information to their advantage. One of the API applications is automated collection of handler count. In the paper entitled "Application of 8D Methodology for Minimizing Test Mixing Event in Semiconductor Test Manufacturing" (Y. T. Prasetyo, 2021), it was found out that mixing event was due to 2 excess units present on the lot. Manual review of handler logs has been crucial for understanding process mapping of the lot as part of applying 8D methodology. Supplementary to this paper, the Count Triangulation would have made the process mapping easier through automatic fetching of handler count and comparing it to the tester count and physical count.

3.0 METHODOLOGY

In a basic validation process in Test manufacturing setting, the tester count is compared with physical count. To assess if there is no mixing event in a particular lot, the rule used is that the tester count must be greater than or equal to the physical count (*Tester count >/= Physical Count*). Otherwise, there is a mixing occurrence (See Table 1). In this case, manual checking of handler logs is done to further investigate the mixing occurrence.

Table 1. Rule in Comparing Tester and Physical Good Units Count

Scenario	Arguments	Quantitative Illustration		Judgment
		Tester	Physical	
1	T=P	10	10	PASSED
2	T>P	11	10	PASSED
3	T<P	9	10	MIXING OCCURRENCE

[a] T = Tester. [b] P = Physical

There are some instances where mixing occurs even though physical count is tallied with tester count. One scenario is when one good unit was lost in the plunger area after testing, but at the same time, there was one leftover unit (from different lot) in good output track prior testing the lot, which compensates the loss quantity in count summary. This scenario will give a passing judgment based on the above-mentioned rule since tester count will be equal to the physical count. From this scenario, the only factors that were considered are the event the unit was plunged and tested, and after the unit is sorted in the tube. The events in between those factors are not seen. This event in between can be further defined through additional handler count matching between tester and physical count. Handler count will give additional count information aside from the tester count and the sorted physical count which will further enhance the counting rule. However, collecting handler count might involve analysis of logs, or manual collection of data through documentation in log sheets. It is essential that information coming from the handler can be extracted by automated means to make the

data reliable and up to date at the same time. The API commands make automatic collection of handler count information possible. Below is a sample API code to get and reset handler count:

GetCount.API – get handler count after end of lot.
ResetCount.API – reset handler count before start and after end of lot.

These commands can be sent from tester to handler via Test program command or Tester Operator Interface (OI) instructions through GPIB communication lines or via RS232. The handler response can be stored as part of test summary. See Fig. 4 for sample of tester to handler communication using API command.

No.	Tester (via Test Program)		Handler	Remarks
1	GetHandlerModel.API	→		Inquire handler model
2		←	Handler send its model type	Handler respond to tester inquiry
3	Program identifies which API Command to use for the specific handler model			Test Program execution
4	ResetCount.API (Before Lot start)	→		Reset the count of handler prior testing the lot
5		←	Handler reset its count and send acknowledgement to Tester	Handler respond to tester command
6	GetCount.API (After Lot End)	→		Inquire count from the handler
7		←	Handler send the current handler count	Handler respond to tester inquiry
8	ResetCount.API (After Lot End)	→		Reset the count of handler after testing the lot
9		←	Handler reset its count and send acknowledgement to Tester	Handler respond to tester command
10	Operator Interface reflects the handler count in OTSR			Test Program execution

Fig. 4. Sample of Tester to Handler Communication

Through API, sorted units from handler output track will be counted accordingly to separate count of good and reject units. As for example as shown in Fig. 5, count of multiple output track (Track 1 and 2) will need to sum up to get the total sorted good units from the handler. Moreover, count of the output track from the rejected units (Track 3, 4, 5 and 6) can be separated according to their respective handler bin assignments.

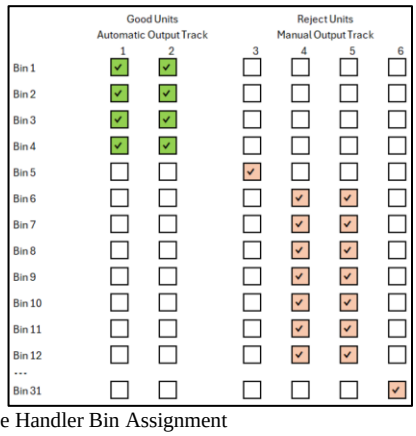

Fig. 5. Sample Handler Bin Assignment

To generate count summary matching rule which will consider tester, handler and physical count, a combination of two permutations were calculated. These were permutation of tester, handler and physical count, and permutation of mathematical symbols which are "equal" (=) and "greater than" (>). For permutation of tester, handler and physical count, formula for permutation with no repetition but with order consideration was used as below:

$$P = n! / (n - r)!$$
$$n = 3 \text{ and } r = 3$$

where n is the number of elements in the set (tester, handler, and physical count), while r is the number of elements to choose. This resulted in six arrangements.

For permutation of the mathematical symbols, formula of permutation with repetition and order consideration was used as below:

$$P = n^r$$
$$n = 2 \text{ and } r = 2$$

where n is the number of elements in the set ("=", and ">") while r is the number of elements to choose. This resulted in four arrangements. See Table 2 for combinations of each permutation.

Table 2. Combinations of Each Permutation

1	T	H	P
2	T	P	H
3	P	T	H
4	P	H	T
5	H	T	P
6	H	P	T

1	=	=
2	=	>
3	>	=
4	>	=

[a] T = Tester. [b] P = Physical. [c] H = Handler.

These two permutations were combined which arrived at 24 (6 x 4) different arguments (see Table 3). However, only 13 unique scenarios have been identified due to duplication of some arguments.

Table 3. Different Combinations of Tester Count, Handler Count and Physical Count.

Scenario	Arguments					Remarks
1	T	=	H	=	P	
2	T	=	H	>	P	
3	T	>	H	=	P	
4	T	>	H	>	P	
5	T	=	P	=	H	Same with Scenario 1
6	T	=	P	>	H	
7	T	>	P	=	H	Same with Scenario 3
8	T	>	P	>	H	
9	P	=	T	=	H	Same with Scenario 1
10	P	=	T	>	H	Same with Scenario 6
11	P	>	T	=	H	
12	P	>	T	>	H	
13	P	=	H	=	T	Same with Scenario 1
14	P	=	H	>	T	
15	P	>	H	=	T	Same with Scenario 11
16	P	>	H	>	T	
17	H	=	T	=	P	Same with Scenario 1
18	H	=	T	>	P	Same with Scenario 2
19	H	>	T	=	P	
20	H	>	T	>	P	
21	H	=	P	=	T	Same with Scenario 1
22	H	=	P	>	T	Same with Scenario 14
23	H	>	P	=	T	Same with Scenario 19
24	H	>	P	>	T	

[a] T = Tester. [b] P = Physical. [c] H = Handler.

The final Count Triangulation rule comparing the tester, handler and physical good units count is shown in Table 4. To assess if there is no mixing event in a particular lot, the Count Triangulation rule is that the tester count must be greater than or equal to the handler count while the handler count must be greater than or equal to the physical count (*Tester count >/= Handler count >/= Physical Count*). Otherwise, there is a mixing occurrence.

Table 4. Count Triangulation Rule Comparing Tester, Handler and Physical Good Units Count

Scenario	Arguments	Quantitative Illustration			Judgment
		Tester	Handler	Physical	
1	T=H=P	10	10	10	PASSED COUNT TRIANGULATION
2	T>H=P	11	10	10	PASSED COUNT TRIANGULATION
3	T=H>P	10	10	9	PASSED COUNT TRIANGULATION
4	T>H>P	12	11	10	PASSED COUNT TRIANGULATION
5	H=P>T	9	10	10	FAILED COUNT TRIANGULATION
6	P>H=T	10	10	11	FAILED COUNT TRIANGULATION
7	H>P>T	9	11	10	FAILED COUNT TRIANGULATION
8	P>H>T	8	9	10	FAILED COUNT TRIANGULATION
9	P>T>H	9	8	10	FAILED COUNT TRIANGULATION
10	H>T>P	11	12	10	FAILED COUNT TRIANGULATION
11	H>T=P	10	11	10	FAILED COUNT TRIANGULATION
12	P=T>H	10	9	10	FAILED COUNT TRIANGULATION
13	T>P>H	11	9	10	FAILED COUNT TRIANGULATION

[a] T = Tester. [b] P = Physical. [c] H = Handler.

Scenarios from 1 to 4 indicate passing the Count Triangulation rule with no mixing event. Scenarios 5 to 9 indicate failing the Count Triangulation rule with mixing event which can be captured using the rule mentioned in Table 1 (*Tester count >/= Physical Count*). Moreover, scenarios 10 to 13 indicate failing the Count Triangulation rule with mixing event which cannot be captured using the rule mentioned in Table 1. This is an additional feature of Count Triangulation.

Webservice will then provide automatic disposition while MES will do the execution whether to hold or release the lot to the next step.

To simulate the identified arguments, one pilot setup with enabled Count Triangulation was used to observe different potential scenarios that may happen during testing in a duration of one quarter. More than 100 lots (>3,000,000 units) were processed in this duration from the pilot setup.

4.0 RESULTS AND DISCUSSION

In this section, the researchers validated the implementation of the Count Triangulation in semiconductor final test manufacturing using one pilot setup. Initial validation run detailed in Table 5, provides a sample OTSR comparing tester, handler and physical count. Based on the initial simulation, tester count, handler count and physical count of good units were tallied. Thus, according to the Count Triangulation rule, this is validated passing (*Tester count >/= Handler count >/= Physical Count*).

Table 5. Sample OTSR

OTSR	Tester Count		Handler Count		Physical Count	Count Triangulation Results
Lot number	Tested Qty In	Good Qty Out	Tested Qty In	Good Qty Out	Good Qty Out	
AAAA.000	12910	12872	12910	12872	12872	PASSED
BBBB.000	15121	15108	15121	15108	15108	PASSED

Through series of validations done in one setup for a duration of one quarter, different potential scenarios were observed that might or might not cause count variance and verified against the predefined Count Triangulation rule (refer to Table 6). Event A showed a smooth-running setup where there was no issue encountered when processing the lot. This is an ideal setup where tester, handler and physical count of good unit are all equal after lot processing.

In Event B, a discrepancy was noted where the tester count of good units exceeded that of the handler count. This discrepancy typically arises when a good unit is inadvertently dropped inside the handler or is incorrectly sorted into the service/default bin due to a jam.

Event C involved an issue with the good output track, where a tube with missing stopper was placed. Additionally, in a separate instance of this event, manual intervention by an operator in transferring or handling good units led to a missing physical good unit. Consequently, the handler count of good units becomes more than the physical count of good units.

For Event D, mis-binning was one of the possible causes where some of rejects or untested units were incidentally sorted into the good unit output track. This event resulted in the handler count of good units to be more than the tester count of good units.

Moreover, for Event E, mishandling occurred in which there was a leftover unit in the tube loaded at the good unit output track. For this event, the physical count of good unit is more than the handler count of good unit.

Table 6. Potential Scenarios that might or might not cause Count Variance

Event	Potential Scenario	Argument
A	All count matches.	T=H=P
B	Passed unit loss/dropped after testing due to the ff: - binned as service/default bin due to jam. Mis-sorting. - unit/s loss/dropped inside the handler after testing. Mis-sorting.	T>H
C	Passed unit loss/dropped after sorting probably due to the ff: - missing stopper at good output track tube. Mis-handling. - transferring/handling of good units. Mis-handling.	H>P
D	Reject/untested unit binned on good tube. Mis-binning. Unexpected unit left from sorter dropped to good output track. Mis-sorting.	H>T
E	There's a left over unit on good output track tube prior testing. Mis-handling.	P>H

[a] T = Tester. [b] P = Physical. [c] H = Handler.

From these potential scenarios, different combinations of events were sorted out depending on the arguments identified in the Count Triangulation rule. By comparing the combination of events and its respective arguments, the judgment whether it is pass or fail in the Count Triangulation rule was further justified (See Table 7).

Table 7. Count Triangulation Judgment from Different Potential Scenarios that might or might not cause Count Variance

Scenario	Arguments	Quantitative Illustration			Judgment	Event
		Tester	Handler	Physical		
1	T=H=P	10	10	10	PASSED COUNT TRIANGULATION	A
2	T>H=P	11	10	10	PASSED COUNT TRIANGULATION	B
3	T=H>P	10	10	9	PASSED COUNT TRIANGULATION	C
4	T>H>P	12	11	10	PASSED COUNT TRIANGULATION	B and C
5	H=P>T	9	10	10	FAILED COUNT TRIANGULATION	D
6	P>H=T	10	10	11	FAILED COUNT TRIANGULATION	E
7	H>P>T	9	11	10	FAILED COUNT TRIANGULATION	C and D
8	P>H>T	8	9	10	FAILED COUNT TRIANGULATION	D and E
9	P>T>H	9	8	10	FAILED COUNT TRIANGULATION	B and E
10	H>T>P	11	12	10	FAILED COUNT TRIANGULATION	C and D
11	H>T=P	10	11	10	FAILED COUNT TRIANGULATION	C and D
12	P=T>H	10	9	10	FAILED COUNT TRIANGULATION	B and E
13	T>P>H	11	9	10	FAILED COUNT TRIANGULATION	B and E

[a] T = Tester. [b] P = Physical. [c] H = Handler.

Through identifying the series of events and its corresponding arguments, the mixing categories in Count Triangulation were observed. Discrepancy of quantity in tester and handler count occurs when there is mis-binning or mis-sorting. Moreover, discrepancy in handler and physical count occurs when there is mis-handling. Finally, the discrepancy in tester and physical count is dependent on the combinations of different scenarios. This can be represented in Fig. 6.

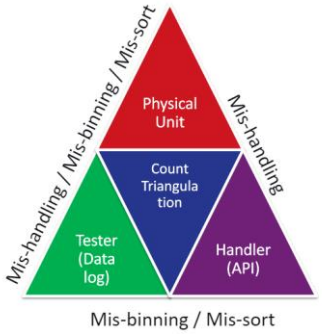

Fig. 6. Count Triangulation Mixing Categories

However, there are some other combinations of events observed that may lead to wrong judgment of mixing event. This serves as the limitation of Count Triangulation implementation (See Table 8). For Limitation 1, a combination of event B and D (See Table 6) both occurred during testing of the lot. This combination of events causes a potential compensation of tester and handler count which makes the arguments satisfy the T=H=P rule. This event also applies with Limitation 2 which compensates handler and physical count.

Table 8. Other Potential Scenarios that might not cause Count Variance

Limitation	Arguments	Remarks	Event
1	T=H=P	PASSED COUNT TRIANGULATION	B and D
2	T=H=P	PASSED COUNT TRIANGULATION	C and E

[a] T = Tester. [b] P = Physical. [c] H = Handler.

5.0 CONCLUSION

In conclusion, the implementation of Count Triangulation in final test gravity fed handlers effectively enhance the count validation process covering 86.67% of different combination of mixing possibilities (based on the arguments and scenarios presented) to trap unit mixing through providing additional handler count (through API) in the test count validation structure, to substantiate a judgement and compare tester, handler, and physical count after testing of the lot. This additional feature strengthens the controls for mixing, mis-binning, or mis-handling events in the semiconductor final test manufacturing as compared with the rule where only tester count and physical count is compared, in which it only captures 60% of mixing possibilities. Within the one quarter implementation in one pilot setup, the Count Triangulation was able to detect potential mixing events and provides correct tester, handler and physical count based on the given arguments and scenarios presented.

6.0 RECOMMENDATIONS

There are some other readily available API commands in the handler that can be used to monitor and control different parameters needed for Semiconductor final test manufacturing process optimization aside from automatic fetching of handler count through API, which is done in this paper. API commands have a vast range of functions, including temperature control, soak time configuration, site mapping setting and managing setup files, among others. The API can also be utilized in monitoring jam and error rate of handlers which can provide valuable information to analyze yield and setup issues. This information helps determine if proactive preventive maintenance of the equipment is recommended.

Moreover, from the given limitations of Count Triangulation, mixing occurrence can be further avoided through additional controls that can be implemented in the production environment such as testing of dummy lot after every lot processed to clean up the handler track and its tube, and adding an automated Final Test Data Checker which will check the unique ID (Wafer ID, X and Y Coordinates) of a specific good unit from previous step to ensure that good units received in the latest step are good units from the previous step.

7.0 ACKNOWLEDGMENT

The authors would like to thank Imelda Cruz, Donald Cruz, Luther Mark Camson, Gilbert Laurel and the rest of Microchip Technology, Philippines for giving their utmost support and guidance in the completion of this study.

8.0 REFERENCES

1. "IEEE Standard for Application Programming Interfaces (APIs) for Deep Learning (DL) Inference Engines," in IEEE Std 2941.2-2023, vol., no., pp.1-31, 22 Nov. 2023, doi: 10.1109/IEEESTD.2023.10326144.

2. S. Pawar, S. K and G. Laxmi, "Application Programming Interface with a Case Study of SOA," 2023 International Conference on Integrated Intelligence and Communication Systems (ICIICS), Kalaburagi, India, 2023, pp. 1-5, doi: 10.1109/ICIICS59993.2023.10421593.

3. Y. T. Prasetyo, A. M. A. Cagubcob, S. F. Persada and A. A. N. P. Redi, "Application of 8D Methodology for Minimizing Test Mixing Event in Semiconductor Test Manufacturing," 2021 IEEE 8th International Conference on Industrial Engineering and Applications (ICIEA), Chengdu, China, 2021, pp. 360-367, doi: 10.1109/ICIEA52957.2021.9436692.

9.0 ABOUT THE AUTHORS

Neilvin Remo has 13 years of experience in Final Test with specialization in Process Engineering, Preventive Maintenance and Line Sustenance. He became part of Microchip Technology Operations (Phils.) Corporation last 2022 and currently working as Process Engineer. He graduated at Manuel S. Enverga University Foundation finishing BS Electronics Engineering degree.

Ralp Lyndon Manalo is a graduate of BS in Electronics Engineering at First Asia Institute of Technology and Humanities. He is currently taking ME Major in Electronics Engineering at Mapua University. He started working at Microchip Technology Operations (Phils.) Corporation since 2018 and currently contributing as Senior Test Product Engineer.

Reandro Lim is a graduate of BS in Electronics Engineering at University of Perpetual Help System DALTA. He has 17 years of experience in Final Test and Wafer Probe as Test Product Engineer. He joined Microchip Technology Operations (Phils.) December 2020 as a Senior Test Product Engineer II.