Increasing Productivity at Wafer Test Using Probe Data Analysis

Topic: Retest Analysis

Author: Akiko Balchiunas
IBM Microelectronics

Date : June 8, 2004

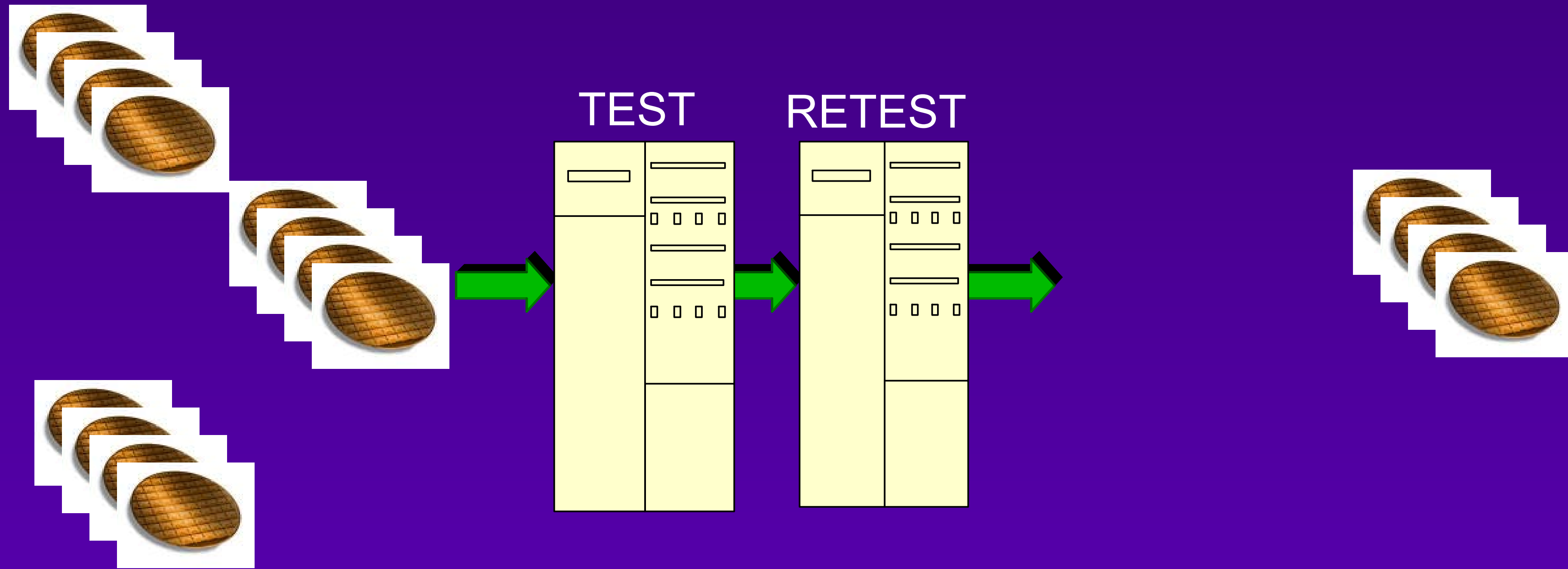
Agenda:

- Wafer Test Productivity Challenge
- Optimizing Retest Using Data
- 5 Step Retest Optimization Algorithm
- “Optimized Retest” Savings Example
- Actual Results
- Additional Information

Wafer Test Productivity Challenge:

CHALLENGE: “ Test Capacity “

Every microelectronics wafer test facility deals with the problem of not having enough tools to handle capacity.

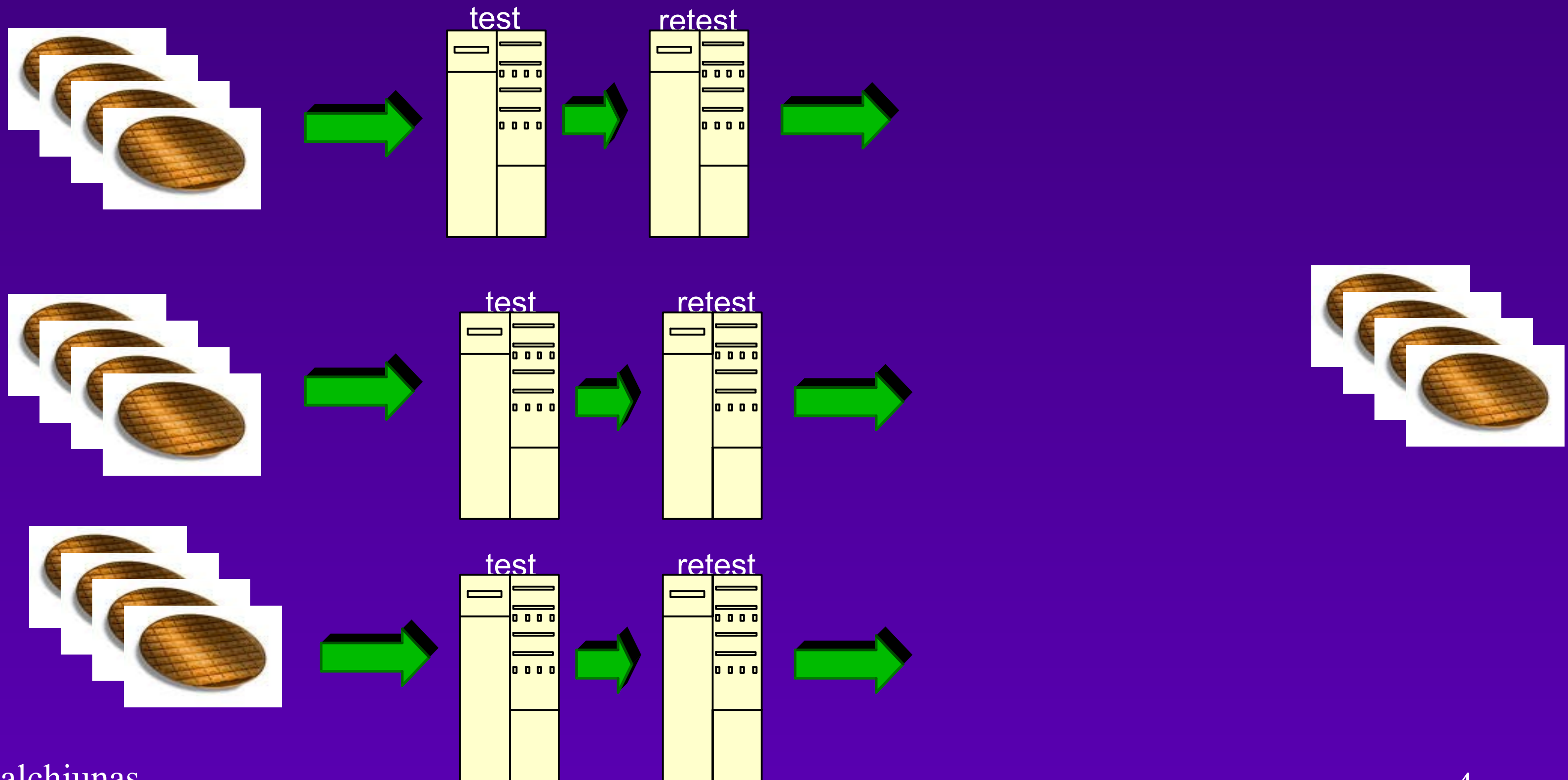


Wafer Test Productivity Challenge

ONE SOLUTION:

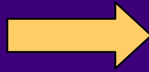
BUY MORE TESTERS →

1. Expensive ($1M & Up)
2. Long Lead Times (months)



Wafer Test Productivity Challenge

BETTER SOLUTION:

INCREASE PRODUCTIVITY 

1. Less Expensive (Eng. Costs)
2. Shorter Lead Times (weeks)

TEST

RETEST

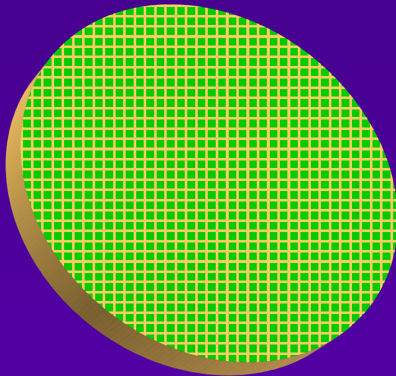
Wafer Test Productivity Challenge:

ONE WAY TO INCREASE PRODUCTIVY is by…..

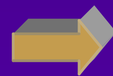
Reducing Test Cycle Time

Typical Wafer Test Cycle :

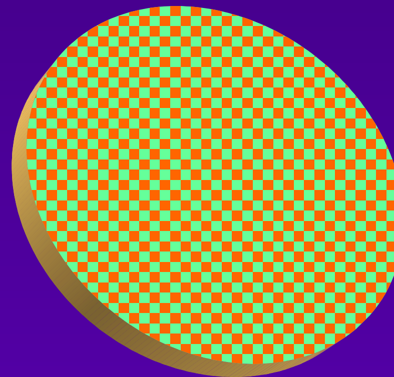
1st PASS TEST:



Rate: 100% coverage



TYPICAL RETEST PASS:



Rate: Retest all 1st Pass Fails

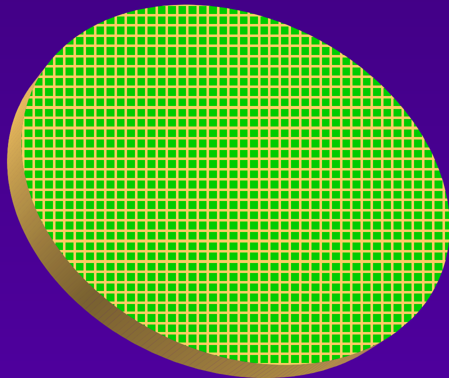
= Total Wafer Test Cycle Time

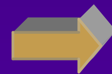
Wafer Test Productivity Challenge:

Reduce Test Cycle Time by… ***Optimizing Retest !***

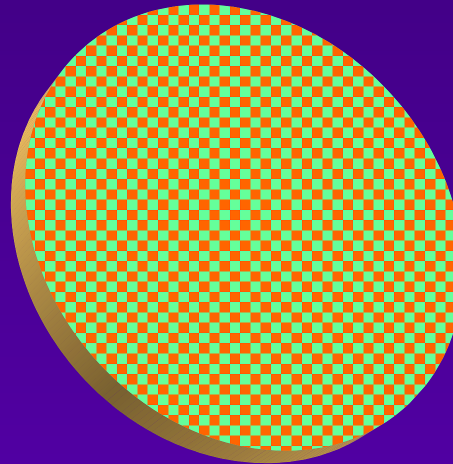
Reduced Wafer Test Cycle :

1st PASS TEST (same)





OPTIMIZED
RETEST PASS:



= Reduced Wafer Test Cycle Time

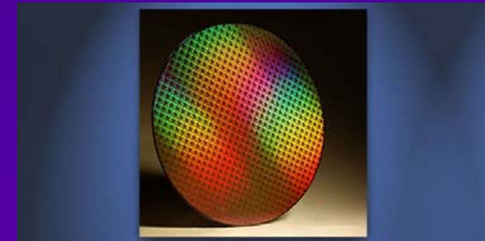
Rate: 100% coverage

Rate: Retest 1st Pass Fail sorts that recover statistically

Optimizing Retest Using Data

Retest Optimization Goals:

- **Minimize the number of chips retested**
- **Maximize the number of customer shippable parts**
- **Minimize yield loss**



Optimizing Retest Using Data

Achieve Goals by using…

5 STEP RETEST OPTIMATION ALGORITHM:

- “COMMON SENSE” METHODOLOGY
- Uses wafer sort data
- Costs = Engineering Analysis Time
- Savings = Hundreds of hours of retest time.

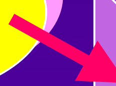
  = Millions of $$ of manfacturing test time

5 Step Retest Optimization Algorithm

STEP 1: **COLLECT 1st & 2nd PASS SORT DATA**

STEP 2: CREATE ***"RETEST ANALYSIS DATASET"***

STEP 3: **RUN *"2nd PASS STATUS FREQ. ANALYSIS"***



STEP 4: **CREATE *"SORT RECOVERY DATASET"***

STEP 5: **CREATE *"OPTIMIZED RETEST TABLE"***

SECTION A
Statistically determine which fail sorts recover on retest
(Steps 1->3)

SECTION B
Add Test Generated Yield Loss (TGYL) Sorts

SECTION C
Determine optimized retest sorts

Algorithm Step 1:

COLLECT 1st & 2nd PASS SORT DATA

1st PASS TEST:

Initial Test pass.

The data collected must contain:

- lot id / wafer id
- chip X/Y
- 1st Pass test suffix (RWJ1)
- Sort

2nd PASS TEST :

Retest 1st pass fails.

Again the data must contain

- lot id / wafer id,
- chip X/ Y
- 2nd Pass test suffix (RWJ2)
- Sort.

"1ST PASS TEST"

Lot ID	Wafr ID	X	Y	Suffix	sort	Desc.
Lot001	Wafer A	1	1	RWJ1	2	Contact Opens
Lot001	Wafer A	1	2	RWJ1	1	GOOD
Lot001	Wafer A	1	3	RWJ1	3	Contact Shorts
Lot001	Wafer A	1	4	RWJ1	6	Leakage
Lot001	Wafer A	1	5	RWJ1	9	Func 2

"2nd PASS TEST"

Lot ID	Wafr ID	X	Y	Suffix	sort	Desc.
Lot001	WaferA	1	1	RWJ2	11	Func4
Lot001	WaferA	1	2	---	---	----
Lot001	WaferA	1	3	RWJ2	1	GOOD
Lot001	WaferA	1	4	RWJ2	6	Leakage
Lot001	WaferA	1	5	RWJ2	1	GOOD

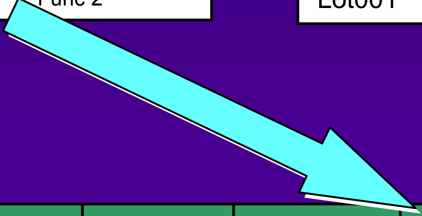
Algorithm Step 2: CREATE the "*RETEST ANALYSIS DATASET*"

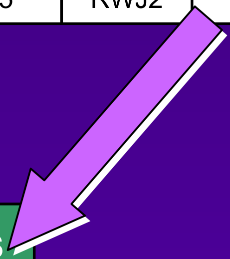
"from STEP1: *1ST PASS TEST SAMPLE*"

Lot ID	Wafer ID	CHIP X	CHIP Y	Suffix	sort	Description
Lot001	WaferA	1	1	RWJ1	2	Contact Opens
Lot001	WaferA	1	2	RWJ1	1	GOOD
Lot001	WaferA	1	3	RWJ1	3	Contact Shorts
Lot001	WaferA	1	4	RWJ1	6	Leakage
Lot001	WaferA	1	5	RWJ1	19	Func 2

"from STEP1: *2nd PASS TEST SAMPLE*

Lot ID	Wafer ID	CHIP X	CHIP Y	Suffix	sort	Description
Lot001	WaferA	1	1	RWJ2	11	Func4
Lot001	WaferA	1	2	---	---	----
Lot001	WaferA	1	3	RWJ2	1	GOOD
Lot001	WaferA	1	4	RWJ2	6	Leakage
Lot001	WaferA	1	5	RWJ2	1	GOOD





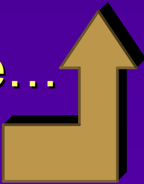
"*RETEST ANALYSIS DATASET sample*"

Lot ID	Wafer ID	X	Y	1st PASS SORT	2nd PASS SORT
Lot001	WaferA	1	1	2	11
Lot001	WaferA	1	2	1	*
Lot001	WaferA	1	3	3	1
Lot001	WaferA	1	4	6	6
Lot001	WaferA	1	5	19	1

Algorithm Step 2: CREATE the *"RETEST ANALYSIS DATASET"* Continued….

"RETEST ANALYSIS DATASET "

Lot ID	Wafer ID	CHIP X	CHIP Y	1st PASS SORT	2nd PASS SORT	2nd PASS SORT STATUS
Lot001	WaferA	1	1	2	11	*BAD*
Lot001	WaferA	1	2	1	*	*
Lot001	WaferA	1	3	3	1	*GOOD*
Lot001	WaferA	1	4	6	6	*BAD*
Lot001	WaferA	1	5	20	1	*GOOD*



Create "2nd PASS SORT STATUS" column where…

- ***BAD*** 2nd PASS SORT = FAILING sort
- ***GOOD*** 2nd PASS SORT = PASSING sort
- * 1st Pass Good (not used in analysis)

Algorithm Step 3:

RUN "2nd PASS STATUS" FREQ. ANALYSIS

"from STEP2:
RETEST ANALYSIS DATASET "

Lot ID	Wafer ID	X	Y	1st PASS SORT	2nd PASS SORT	2nd PASS SORT STATUS
Lot001	WaferA	1	1	2	11	BAD
Lot001	WaferA	1	2	1	*	*
Lot001	WaferA	1	3	3	1	GOOD
Lot001	WaferA	1	4	6	6	BAD
Lot001	WaferA	1	5	20	1	GOOD

Run data analysis that counts the *FREQUENCY* of "2nd Pass Status" (GOOD, BAD) per "1st Pass Sort"

Results of Freq. Analysis:

1st Pass Sort	Desc.	TOTAL Retested	BAD	GOOD
2	Contact Opens	750	625	125
3	Contact Shorts	160	155	5
4	Contact Power	60	60	0
6	Leak	20	18	2
10	Func 1	280	268	2
11	Func 2	9	6	3
14	Probe Melt	8	8	0
19	Func 3	330	328	2
20	Func 4	32	31	1
21	Func 5	200	160	40
22	Func 6	5	5	0

Akiko Balchiunas

Algorithm Step 3: RUN "2nd PASS STATUS" FREQ. ANALYSIS Continued….

Results of Freq. Analysis:

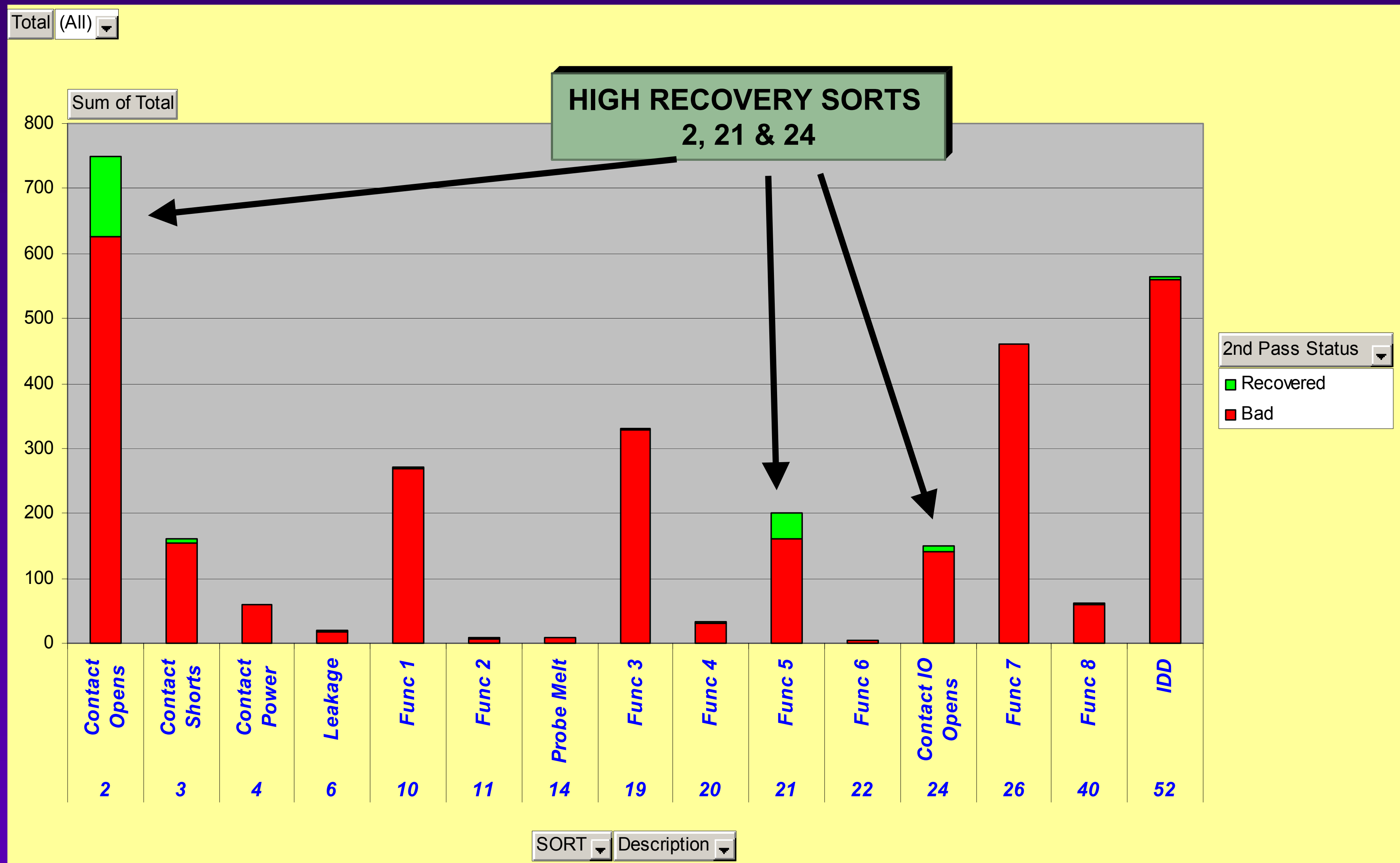
1st Pass Sort	Descrip	TOTAL Retested	BAD	GOOD Recov'd	Recovery Rate
2	Contact Opens	750	625	125	16.7%
3	Contact Shorts	160	155	5	3.1%
4	Contact Power	60	60	0	0.0%
6	Leakage	20	18	2	10.0%
10	Func 1	280	268	2	0.7%
11	Func 2	9	6	3	33.3%
14	Probe Melt	8	8	0	0.0%
19	Func 3	330	328	2	0.6%
20	Func 4	32	31	1	3.1%
21	Func 5	200	160	40	20.0%
22	Func 6	5	5	0	0.0%

Calculate the *Recovery Rate* for each "1st Pass Sort" where….

$$\text{Recovery Rate (\%)} = \frac{\text{Good (aka Recovered)}}{\text{Total Retested}}$$

RECOVERED SORTS graph

from STEP3: “2nd PASS STATUS” FREQUENCY ANALYSIS

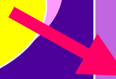


5 Step Retest Optimization Algorithm

STEP 1: **COLLECT 1st & 2nd PASS SORT DATA**

STEP 2: CREATE ***"RETEST ANALYSIS DATASET"***

STEP 3: **RUN *"2nd PASS STATUS FREQ. ANALYSIS"***



TGYL SORTS

STEP 4: **CREATE *"SORT RECOVERY DATASET"***

SECTION A

Statistically determine which fail sorts recover on retest

(Steps 1->3)

SECTION B

Add Test Generated Yield Loss (TGYL) Sorts

Algorithm STEP 4: Obtain TGLY Fail Sorts

Test Generated Yield Loss (TGLY)

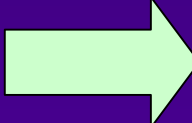
TGLY SORTS ARE TEST INDUCED FAILS

Historically test induced sorts include:

1. Contact
2. Leakage
3. Probe melts

Causes of TGYL include:

- Operator setup problems
- Bad front end hardware (probes)
- Testers drifting out of calibration
- PNP (Part Number Program) errors



TGYL Fail Sort Table

Sort	Description	TGYL Fail ?
2	**Contact Opens**	**Y**
3	**Contact Shorts**	**Y**
4	**Contacts Power**	**Y**
6	**Leakage**	**Y**
14	**Probe Melt**	**Y**
24	**Contact IO Opens**	**Y**

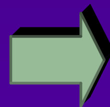
Algorithm Step 4:

CREATE the "SORT RECOVERY DATASET"

Create
SORT RECOVERY DATASET
by adding TGYL FAIL indicator.

TGYL Fail Sorts

Sort	Description	TGYL Fail ?
2	Contact Opens	Y
3	Contact Shorts	Y
4	Contacts Power	Y
6	Leakage	Y
14	Probe Melt	Y
24	Contact IO Opens	Y



SORT RECOVERY DATASET

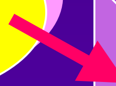
Sort	Desc.	TGYL Fail ?	TOTAL Retested	Bad	Reco v'd	Recovery Rate
2	Contact Opens	Y	750	625	125	**16.7%**
3	Contact Shorts	Y	160	155	5	3.1%
4	Contact Power	Y	60	60	0	0.0%
6	Leakage	Y	20	18	2	**10.0%**
10	Func 1		280	268	2	0.7%
11	Func 2		9	6	3	**33.3%**
14	Probe Melt	Y	8	8	0	0.0%
19	Func 3		330	328	2	0.6%
22	Func 6		5	5	0	0.0%
24	Contact IO Opens	Y	150	140	10	**6.7%**

5 Step Retest Optimization Algorithm

STEP 1: **COLLECT 1st & 2nd PASS SORT DATA**

STEP 2: CREATE ***"RETEST ANALYSIS DATASET"***

STEP 3: **RUN *"2nd PASS STATUS FREQ. ANALYSIS"***



STEP 4: **CREATE *"SORT RECOVERY DATASET"***

STEP 5: **CREATE *"OPTIMIZED RETEST TABLE"***

SECTION A

Statistically determine which fail sorts recover on retest

(Steps 1->3)

SECTION B

Add Test Generated Yield Loss (TGYL) Sorts

SECTION C

Determine optimized retest sorts

Algorithm STEP 5: Create *"Optimized Retest Table"*

Sort	Descrip.	TGYL Fail ?	Recovery Rate
2	Contact Opens	Y	**16.7%**
3	Contact Shorts	Y	**3.1%**
4	Contact Power	Y	0.0%
6	Leakage	Y	**10.0%**
10	Func 1		0.7%
11	Func 2		**33.3%**
14	Probe Melt	Y	0.0%
19	Func 3		0.6%
20	Func 4		**3.1%**
21	Func 5		**20.0%**
22	Func 6		0.0%
24	Contact IO Opens	Y	**6.7%**
26	Func 7		0.0%

WHICH SORTS TO RETEST ?

Determined by:

- TGLY Fail
- Recovery Rate

Recovery Rate Cut Point Determination vs. Supply/Demand

- Recovery Rate Cut Point = 0%
  Products that require every chip to meet demand
  - Low Yielding
  - High Demand / High Profit Margin

- Recovery Rate Cut Point = ~1->2%
  Products where test capacity is limited
  - High volume product / High-Med Profit Margin
  - Supply meeting customer demands

- Recovery Rate Cut Point < 3+%
  Low Profit Products that meet/exceed demands
  - Supply meeting / exceeding demands
  - Low Profit Margin

Algorithm STEP 5: Create *"Optimized Retest Table"*

Sort	Desc	TGYL Fail ?	Recovery Rate
2	Contact Opens	Y	**16.7%**
3	Contact Shorts	Y	**3.1%**
4	Contact Power	Y	0.0%
6	Leakage	Y	**10.0%**
10	Func 1		0.7%
11	Func 2		**33.3%**
14	Probe Melt	Y	0.0%
19	Func 3		0.6%
20	Func 4		**3.1%**
21	Func 5		**20.0%**
22	Func 6		0.0%
24	Contact IO Opens	Y	**6.7%**
26	Func 7		0.0%

RETEST SORT ? = YES

- TGLY Fail = Y
- Recovery Rate > 1.0 %

Algorithm STEP 5: Create *"Optimized Retest Table"*

Sort	Desc	TGYL Fail ?	Recovery Rate	Retest Sort ?
2	Contact Opens	Y	16.7%	YES
3	Contact Shorts	Y	3.1%	YES
4	Contact Power	Y	0.0%	YES
6	Leakage	Y	10.0%	YES
10	Func 1		0.7%	
11	Func 2		33.3%	YES
14	Probe Melt	Y	0.0%	YES
19	Func 3		0.6%	
20	Func 4		3.1%	YES
21	Func 5		20.0%	YES
22	Func 6		0.0%	
24	Contact IO Opens	Y	6.7%	YES
26	Func 7		0.0%	

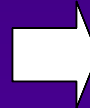
RETEST SORT ? = YES

- TGLY Fail = Y
- Recovery Rate > 1.0 %

“Optimized Retest ” Savings Example:

BEFORE OPTIMIZATION:

Sort	TOTAL Retest	Recovery Rate	Retest Sort ?
2	750	16.7%	YES
3	160	3.1%	YES
4	60	0.0%	YES
6	20	10.0%	YES
10	280	0.7%	
11	9	33.3%	YES
14	8	0.0%	YES
19	330	0.6%	
20	32	3.1%	YES
21	200	20.0%	YES
22	5	0.0%	
24	150	6.7%	YES
26	460	0.0%	
52	565	0.9%	
TOT	**3090**	**6.4%**	



AFTER OPTIMIZATION:

Sort	TOTAL Retest	Recovery Rate	Retest Sort ?
2	750	16.7%	YES
3	160	3.1%	YES
4	60	0.0%	YES
6	20	10.0%	YES
11	9	33.3%	YES
14	8	0.0%	YES
20	32	3.1%	YES
21	200	20.0%	YES
24	150	6.7%	YES
TOT	**1450**	**13.8%**	

"Optimized Retest " Savings Example:

SAVINGS SUMMARY:

- Chip Retest reduced *53%*
  - Before: 3090 chips Retested
  - After : 1450 chips Retested
- More Efficient Retest
  - Before: 6.4% Recovery Rate
  - After : 13.8% Recovery Rate

AFTER OPTIMIZATION:

Sort	TOTAL Retest	Recovery Rate	Retest Sort ?
2	750	16.7%	YES
3	160	3.1%	YES
4	60	0.0%	YES
6	20	10.0%	YES
11	9	33.3%	YES
14	8	0.0%	YES
20	32	3.1%	YES
21	200	20.0%	YES
24	150	6.7%	YES
TOT	**1450**	**13.8%**	

Actual Results

- Reduced the chip retest rate by ~80% & ~50% on IBM Microelectronics' 2 highest volume tool constrained products.
- Performed without spending millions on new testers and without impacting customer deliverables. (no yield loss)

- Implementation took less than a month.

Additional Information:

- Retest reduction has benefits in the longevity of probes. (reduction of touchdowns)
- Retest analysis has aided in the learning of which tests are susceptible to test generated errors.
  (Bad probing, operator setups, defective HW, etc)
- Partial Goods and Supply/Demand can also effect which sorts to retest. (see me for details)

Conclusion:

- Efficient way to solve capacity issues is to reduce test time.
- One way is to optimize retest
- A simple 5 step algorithm can be used which takes into account the following:
  - Sorts that statistically recover
  - Known Test Generated Yield Loss sorts
- As a result capacity is increased savings hundreds of hours of test time.

Thank you !:

- Thanks for the opportunity to present.
- Special thanks to Fred Taber who really encouraged me to submit a paper to this workshop.

QUESTIONS AND ANSWERS