Base on Multi-Level software structure for Efuse Common Library

Steve Xie, Yanfen Fang, Nick Song
ADVANTEST

TestConX中国 China

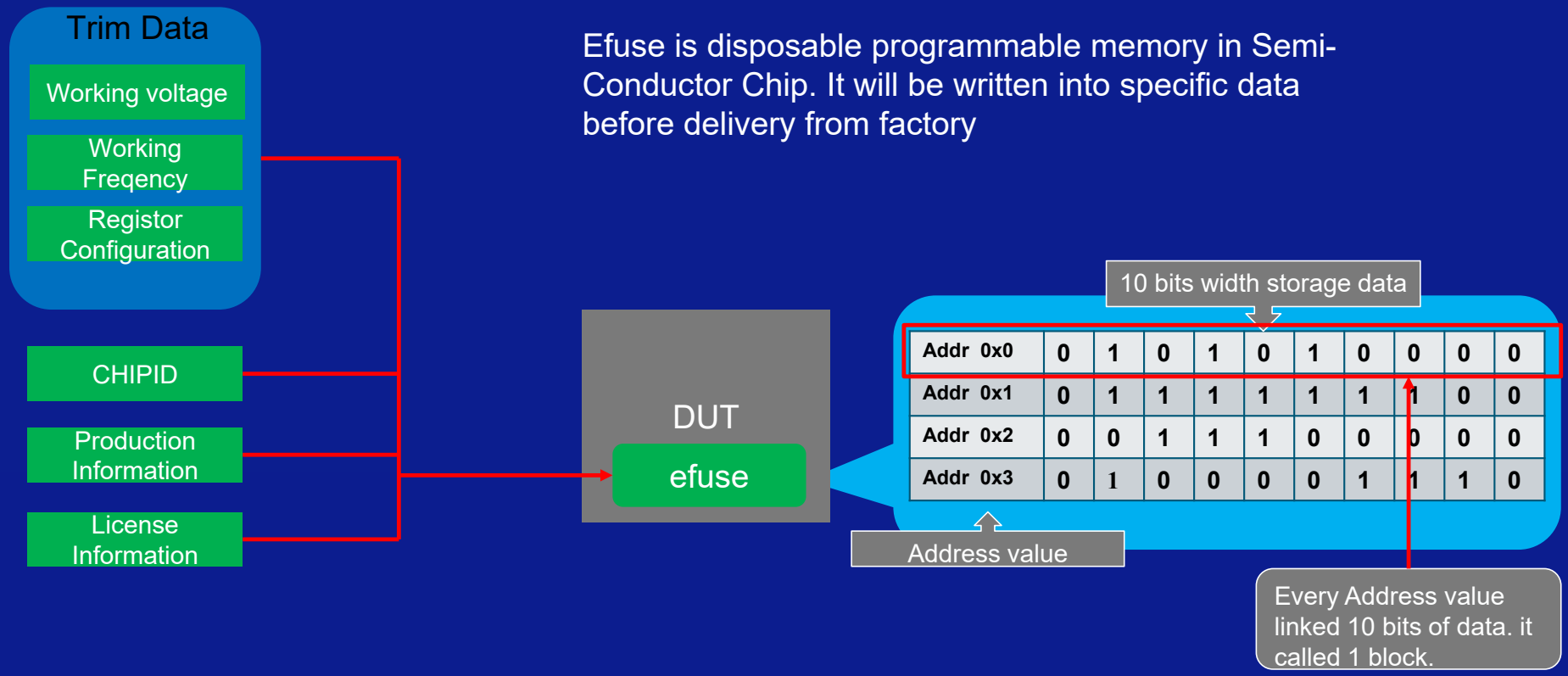
Virtual ▪ November 1-4, 2022

ADVANTEST

Agenda

- What is Efuse
- Normal Processing Flow For efuse
- New efuse Structure

What Is efuse



TestConX China

Base on Multi-Level software structure for Efuse Common Library

2022

3

Normal Processing Flow For efuse

Getting efuse map

Modify old lib /method for compatible with efuse data update

Create multiple write test suite for each adrress

Need Optimizing

- Data location change
- Data property change
- Data processing algorithm change
- Transfer protocol update

Originally efuse test will running as below:

Firstly, collecting data distribution information from chip introduction document(efuse map).

Secondly, developing test method for transfer real data into efuse data container.

Finally, download efuse data into chip using specific protocol.

Based on these steps, testing engineer will refactor test code of efuse data management.

Modifying test code frequently will influence stability and robust of test program, and it will increase the risk of error occurring of test code.

TestConX中国 China

Base on Multi-Level software structure for Efuse Common Library

2022

4

Data location change

Normal Processing Flow For efuse

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data1	Data2		Chip Version				Reserve Data		
0x1	Calibration data									
0x2	ChipID			Processing Log Data				Reserve Data		

Address 0x0 = (Chip Version >> 24) | (Data2 >> 8) | (Data1>>0)

Address 0x1 = Calibration data

Address 0x2 = (Processing Log Data >> 32) | (ChipID)

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data2		Data1	Reserve Data				Chip Version		
0x1					Processing Log Data			Reserve Data		
0x2	Calibration data									

Switch data position in same address

Switch data position in same address

Switch data position in different address

Address 0x0 = (Chip Version >> 48) | (Data2 >> 0) | (Data1>>16)

Address 0x1 = (Processing Log Data >> 32) | (ChipID)

Address 0x2 = Calibration data

Lot of code change

Data Shift

Data region reflect

Code

To remap Data location in efuse address structure, we need do lot of work for data shift and data region reflect. **This structure is unfriendly with program update if data location has change, and every change action will lead to big potential risk**

TestConX中国 China

Base on Multi-Level software structure for Efuse Common Library

2022

5

5

Data property change

Normal Processing Flow For efuse

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data1	Data2		Chip Version				Reserve Data		
0x1	Calibration data									
0x2	ChipID				Processing Log Data			Reserve Data		

Address 0x0 = (Chip Version >> 24) | (Data2 >> 8) | (Data1>>0)
Address 0x1 = Calibration data
Address 0x2 = (Processing Log Data >> 32) | (ChipID)

Same code statement in **different** efuse data. **If we send 32 bits data into Chip Version, byte6 will be written into invalid data**

Chip Version data length -1

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data 1	Data2		Chip Version				Reserve Data		
0x1	Calibration data									
0x2	ChipID				Processing Log Data			Reserve Data		

TestConX中国 China

Base on Multi-Level software structure for Efuse Common Library

2022

6

Data property change

Normal Processing Flow For efuse

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data1	Data2		Chip Version				Reserve Data		
0x1	Calibration data									
0x2	ChipID				Processing Log Data			Reserve Data		

Tough Point

Chip Version = Chip Version & 0xffffffff

Chip Version = Chip Version & 0xffffff

Addition code

Address 0x0 = (Chip Version >> 24) | (Data2 >> 8) | (Data1>>0)

Address 0x1 = Calibration data

Address 0x2 = (Processing Log Data >> 32) | (ChipID)

For prevent invalid data write into byte6, we need add bit mask for ensuring data length is correct before data shift.

But this changing need insert each position that data length is different

Chip Version data length -1

Data length change

Code

Lot of code working

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data 1	Data2		Chip Version				Reserve Data		
0x1	Calibration data									
0x2	ChipID				Processing Log Data			Reserve Data		

TestConX中国 China

Base on Multi-Level software structure for Efuse Common Library

2022

7

Data property change

Normal Processing Flow For efuse

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data 1	Data2		Chip Version			Reserve Data			

If Chip version data has 2 sub-data: data1 and data2

Each of these data is calculated in different links, so it cannot get value at same time

For data1, we need calculate address data as below:

Chip version = ((data1 & 0x3ff) | ((data1 & 0x7c00) >> 14)) & 0x07c3ff Address0x0 = Address0x0 | Chip version >> 24

For data2, we need calculate address data as below:

Chip version = (((data2 & 0xf) >> 10) | ((data2 & 0x1f0) >> 19)) & 0xf83c00 Address0x0 = Address0x0 | Chip version >> 24

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8	Bit 9	Bit 10	Bit 11	Bit 12	Bit 13	Bit 14	Bit 15	Bit 16	Bit 17	Bit 18	Bit 19	Bit 20	Bit 21	Bit 22	Bit 23
data1	data1	data1	data1	data1	data1	data1	data1	data1	data1	data2	data2	data2	data2	data1	data1	data1	data1	data1	data2	data2	data2	data2	data2

If data bit location has changed, the reference calculate formula will changed accordingly

For data1, we need calculate address data as below:

Chip version = ((data1 & 0xff) | ((data1 & 0x7f00) >> 14)) & 0x1fc0ff

For data2, we need calculate address data as below:

Chip version = (((data2 & 0x3f) >> 8) | ((data2 & 0x1c0) >> 21)) & 0xc03f00

Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8	Bit 9	Bit 10	Bit 11	Bit 12	Bit 13	Bit 14	Bit 15	Bit 16	Bit 17	Bit 18	Bit 19	Bit 20	Bit 21	Bit 22	Bit 23
data1	data1	data1	data1	data1	data1	data1	data1	data2	data2	data2	data2	data2	data2	data1	data1	data1	data1	data1	data1	data1	data2	data2	data2

Impact: Lot of complex data calculate formula need change, and counter-intuitive mask process will **greatly amplify probability of risk occurrence.**

Assign value format change

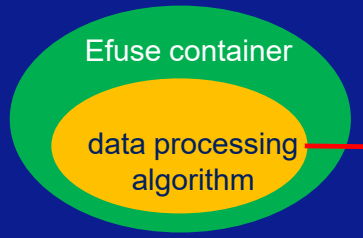
Code

Base on Multi-Level software structure for Efuse Common Library

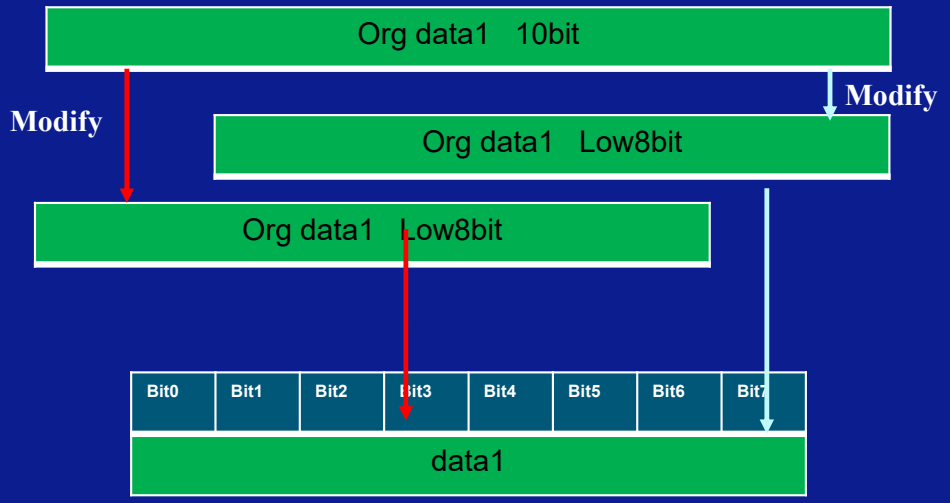
Data processing algorithm change

Normal Processing Flow For efuse

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data 1	Data2		Chip Version				Reserve Data		



Cannot modifying data processing algorithm skip Efuse container structure



Normal data processing algorithm code often mix into data container structure code, this structure has several problem below:

1. **it is hard to update algorithm without influence data structure code.**
2. **It is hard to find out update code position for new member**

TestConX中国 China

Base on Multi-Level software structure for Efuse Common Library

2022

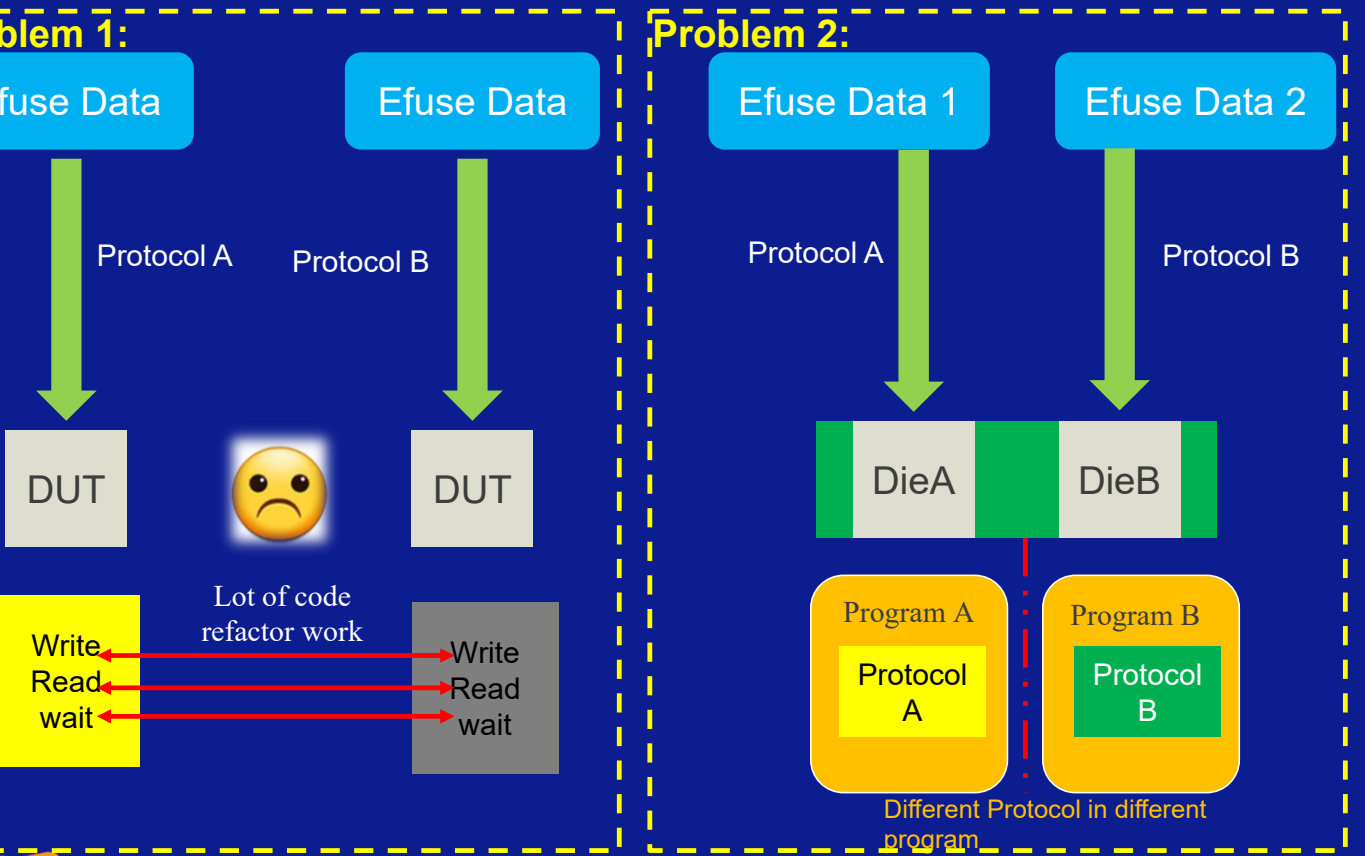
9

Transfer protocol update

Normal Processing Flow For efuse

Problem 1:
Protocol update always need modify protocol structure reference code like:
Write Action
Read Action
Waite Action

Problem 2:
we need develop different program for different die



TestConX China

Base on Multi-Level software structure for Efuse Common Library

2022

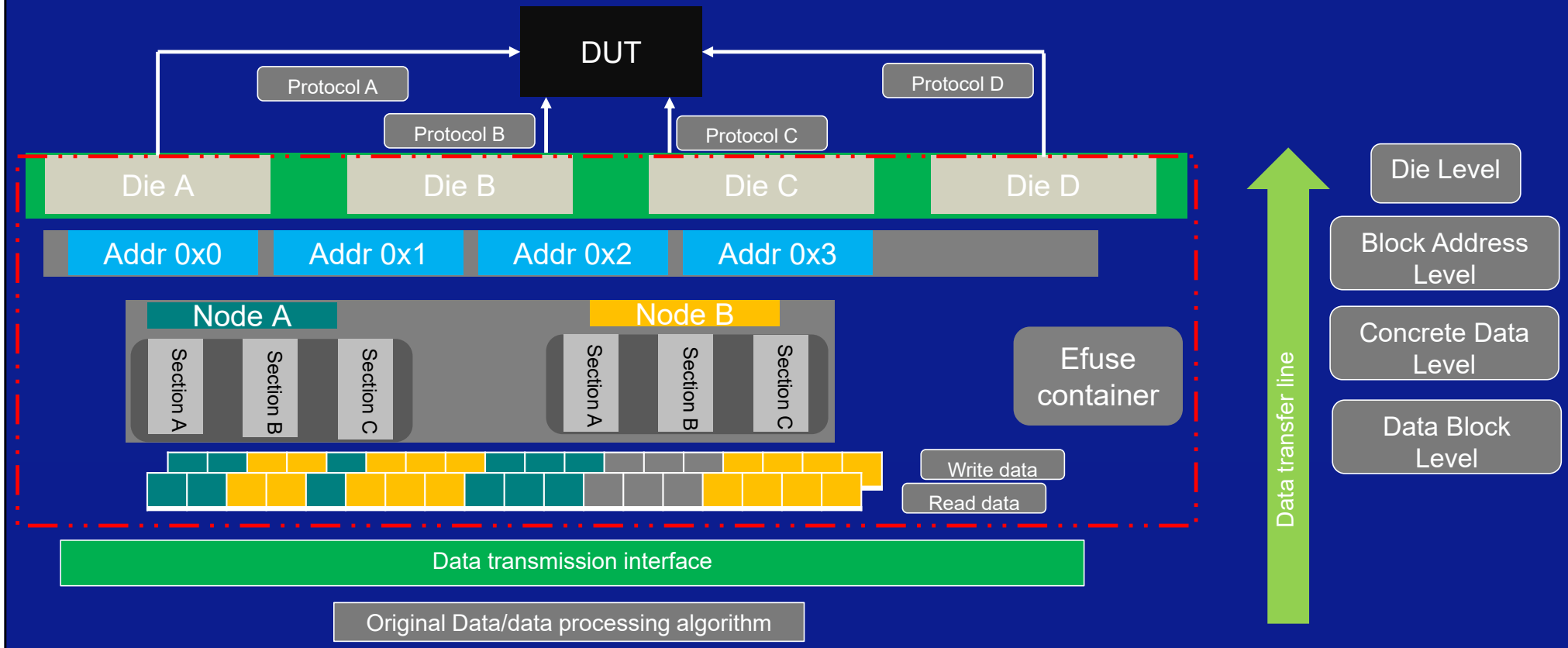
10

New efuse Structure

Normal Process	New Efuse Solution
Data location change	CSV Input Format
Data property change	CSV Input Format/ Data transmission interface
Data processing algorithm change	Data transmission interface
Transfer protocol update	Multi-Die Protocol Management/Protocol data filling

New efuse Structure

- Multi-Level Data Container



TestConX中国 China

Base on Multi-Level software structure for Efuse Common Library

2022

New efuse Structure

- Multi-Level Data Container

```
map<string, EFUSE_DIE> DIE_MAP;
```

We using 2 level map structure to manage different Die and different data

Manage

Addr 0x0 | Addr 0x1 | Addr 0x2 | Addr 0x3

```
map<string, EFUSE_NODE>NODE_MAP;
```

Manage

Manage

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data 1	Data2		Chip Version				Reserve Data		
0x1	Calibration data									
0x2	ChipID			Processing Log Data				Reserve Data		

Node A

```
class UNISOC_EFUSE_NODE
```

TestConX中国 China

Base on Multi-Level software structure for Efuse Common Library

2022

13

New efuse Structure

- CSV Input Format

address	Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0x0	Data 1	Data2		Chip Version				Reserve Data		
0x1	Calibration data									
0x2	ChipID				Processing Log Data			Reserve Data		

Address 0x0 = (Chip Version >> 24) | (Data2 >> 8) | (Data1>>0)
Address 0x1 = Calibration data
Address 0x2 = (Processing Log Data >> 32) | (ChipID)

Chip Version = Chip Version & 0xffffffff
Chip Version = Chip Version & 0xffffff

Coding work

Forms Input

	NodeName	HDMI_TRIM	CVBS_TRIM	dsv
1	signBit	Y	N	Y
2	BlockID	0	3	4
3	BlockLoc	0,5,7~10,15,20	0~23	0,7~8
4	Type	WR	WR	WR

All above coding work become simple forms input
Efficiency and safety improve greatly

CSV

- Data Shift
- Data region reflect
- Data length change

All these action only need modify CSV data

Base on Multi-Level software structure for Efuse Common Library

2022

14

New efuse Structure

- Separate Structure Design

Data Container structure

data processing algorithm

Optimizing

Efuse Container

Contain

Write Data

Read Data

Individual data container for write and read. This structure will avoid data confusion

Data transmission interface

Split data processing algorithm from data container structure. Add Data transmission interface Level for convenient transfer data

We provided various set and get function for developer send or extract data convenience in **Data transmission interface**

Data processing algorithm

New solution

Data processing algorithm

```
/*********************************set org data into node*******************/
EFUSE.getEFUSENode(efuse_node).update_TestResult(site, ethdata.trim_eth_amp[site]);
```

```
void update_TestResult(int site, int data);
void update_TestResult(int site, int data, int BitLoc);
void update_TestResult(int site, Unsigned128 data);
void update_TestResult(int site, Unsigned128 data, Unsigned128 BitLoc);
void getAllDUTsReadData(MultiSiteUnsignedLong128bit& data);
Unsigned128 getReadData(int site);
void getAllDUTsTestResultOnNode(MultiSiteUnsignedLong128bit& data);
void getAllDUTsTestResultOnNode(MultiSiteInt& data);
Unsigned128 getAllDUTsTestResultOnNode(int site);
```

We using set/get function at where original data is generate or calculate, this make program updating become easy

TestConX中国 China

Base on Multi-Level software structure for Efuse Common Library

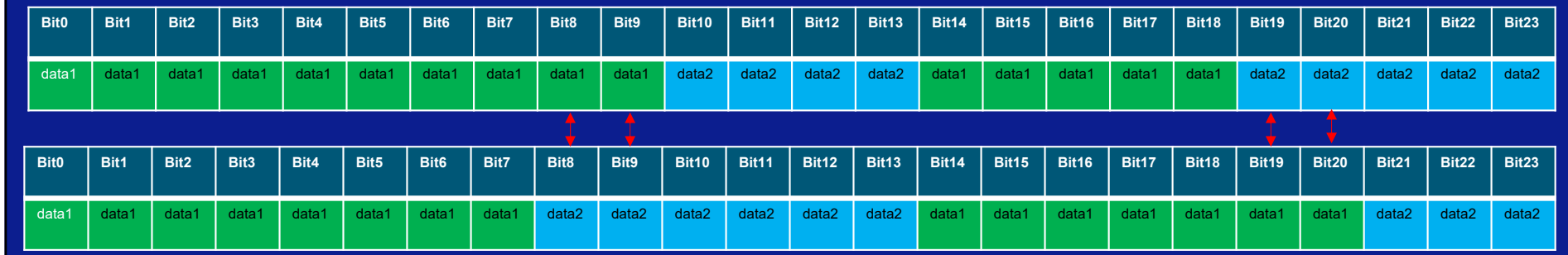
2022

15

15

New efuse Structure

- Simplify coding



If data bit location has changed, the reference calculate formula will changed accordingly

For data1, we need calculate address data as below:

Chip version = ((data1 & 0xff) | ((data1 & 0x7f00) >> 14)) & 0x1fc0ff

For data2, we need calculate address data as below:

Chip version = (((data2 & 0x3f) >> 8) | ((data2 & 0x1c0) >> 21)) & 0xe03f00

Complex coding

Address0x0 = Address0x0 | Chip version >> 24

Address0x0 = Address0x0 | Chip version >> 24

Optimizing

Complex coding work become simple **BitLoc value setting**. This BitLoc do not need to be concerned about node position in block level

```
void update_TestResult(int site, int data, int BitLoc);
```

data1/data2

Value

Data2:
BitLoc = 00000000001111000011111
Data1:
BitLoc = 11111111100001111100000

Data2:
BitLoc = 00000000111111000000111
Data1:
BitLoc = 11111111000000111111000

8 changes → 2 changes

Safety and usability

TestConX中国 China

Base on Multi-Level software structure for Efuse Common Library

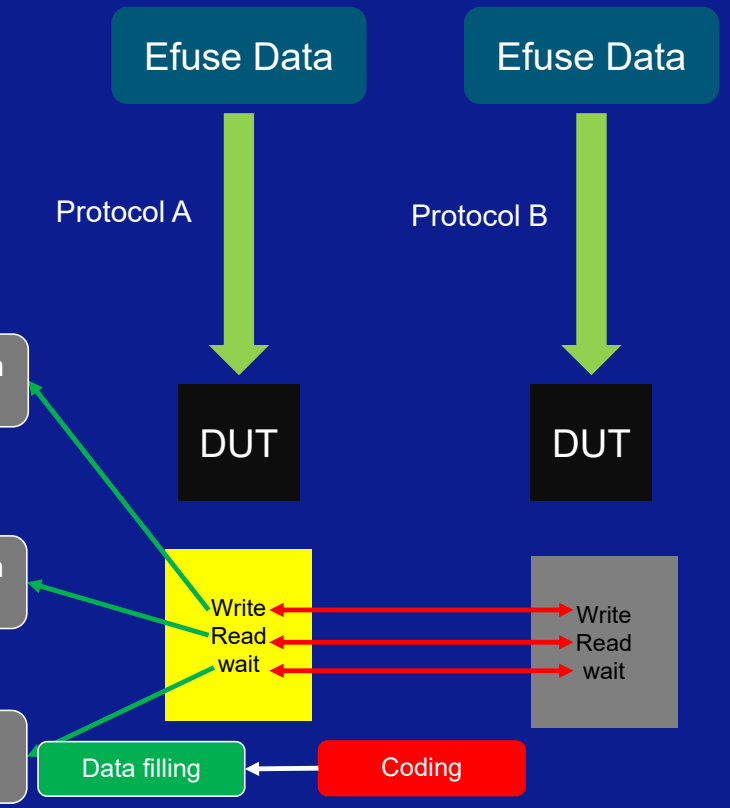
2022

16

New efuse Structure

- Protocol Description Format

Test Method	EFUSE_CBB_tml.EfuseInit
Parameters	...
d2sMode	LearningMode
Die	1
Die0	
BlockBitsWidth	32
PortName	protocol_transfer
NodeConfigPath	AllNodes.csv
Write	
ADDR_PIN	address_pin
LabelName	protocol-pattern-write
ADDRLSB	128
ADDRMSB	120
DATALSB	150
DATAMSB	142
Read	
ADDR_PIN	address_pin
DATA_PIN	data_pin
LabelName	protocol-pattern-read
ADDRLSB	128
ADDRMSB	120
DATALSB	150
DATAMSB	142
waitTemplate	
LabelName	protocol-pattern-wait



Based on convenient operation interface of **ADVANTEST SmarTest** environment, we just need fill reference description data in testsuite properties. Then we using **D2S technology** to dynamic generate protocol vector and transfer to chip.(**D2S technology** is protocol software solution of **ADVANTEST SmarTest**, it support various serial/parallel protocol pattern generate,)

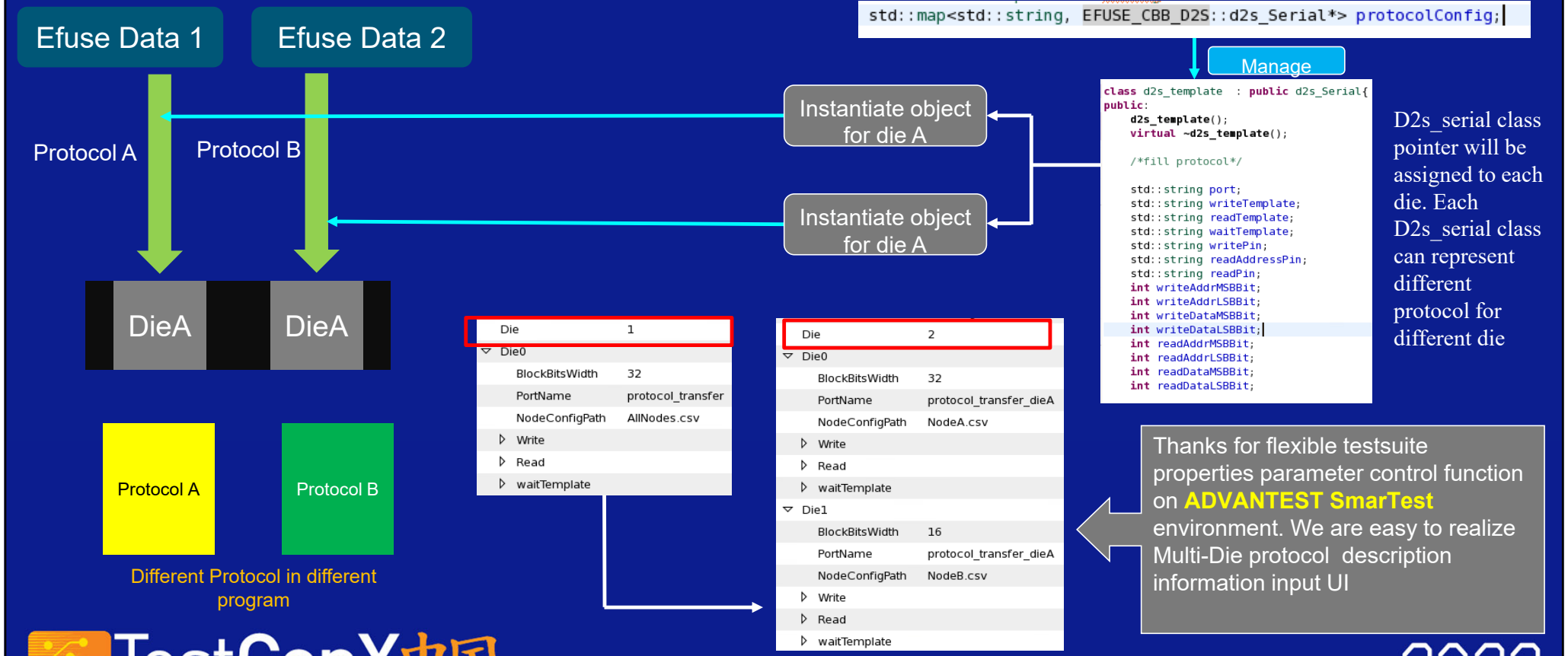
TestConX中国 China 2022

Base on Multi-Level software structure for Efuse Common Library

17

New efuse Structure

- MultiDie Protocol Management



D2s_serial class pointer will be assigned to each die. Each D2s_serial class can represent different protocol for different die

Thanks for flexible testsuite properties parameter control function on **ADVANTEST SmarTest** environment. We are easy to realize Multi-Die protocol description information input UI

TestConX中国 China

2022

Base on Multi-Level software structure for Efuse Common Library

18

Summary

Implement Environment:
ADVANTEST SmarTest on V93000 platform

Dgydqwdjh=

41 P dnh#surjudp #ghyhorsp hqw#z run#fkdqjh#wr#fvy#hglw#z klgh#ixvh#G dwd#Orfdwlrq# Fkdqjh#

51 Xsgdwh#G dwd#Surshuw|#z lwkrxw#hidfwru#edvlf#gdwd#vwuxfwxuh

61 Vlp sdi|#surwrfro#hdd}h#z lwkrxw#hvvqp hwkrg#p rgli|lqj#

71 Frp sdwlen#P xowlG lh#surwrfro#q#4#surjudp

With Thanks to Our Sponsors!

Honored

JCT TECHNOLOGY 金东唐科技

Distinguished

smiths interconnect

TERADYNE

FELDMAN ENGINEERING


金东唐科技
TECHNOLOGY
5G
自动化&智能化的综合方案供应商
Automation & intelligent integrated solution supplier
jdt-precision.com


TestConX中国™
China
is sponsored by
smiths interconnect
Your Global Partner for Innovative
Semiconductor Test Solutions

Enabling the Next Generation of Technology
Through Advanced Test Solutions

TERADYNE.COM

COPYRIGHT NOTICE