2021 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION CHINA
SHANGHAI | MAY 26, 2021

Completely Release the Power of Verification IP – A Step-by-Step Guidance for In-House IP Development

Bin Liu, Nan Ni, Chongyu Zhang, RockerIC, Xi’an
Xiaoming Ma, Chipsemicorp, Shanghai
Kailong Wang, Qualcomm, Shanghai
Qiaochu Zhu, GOODiX, Shanghai

Introduction
介绍

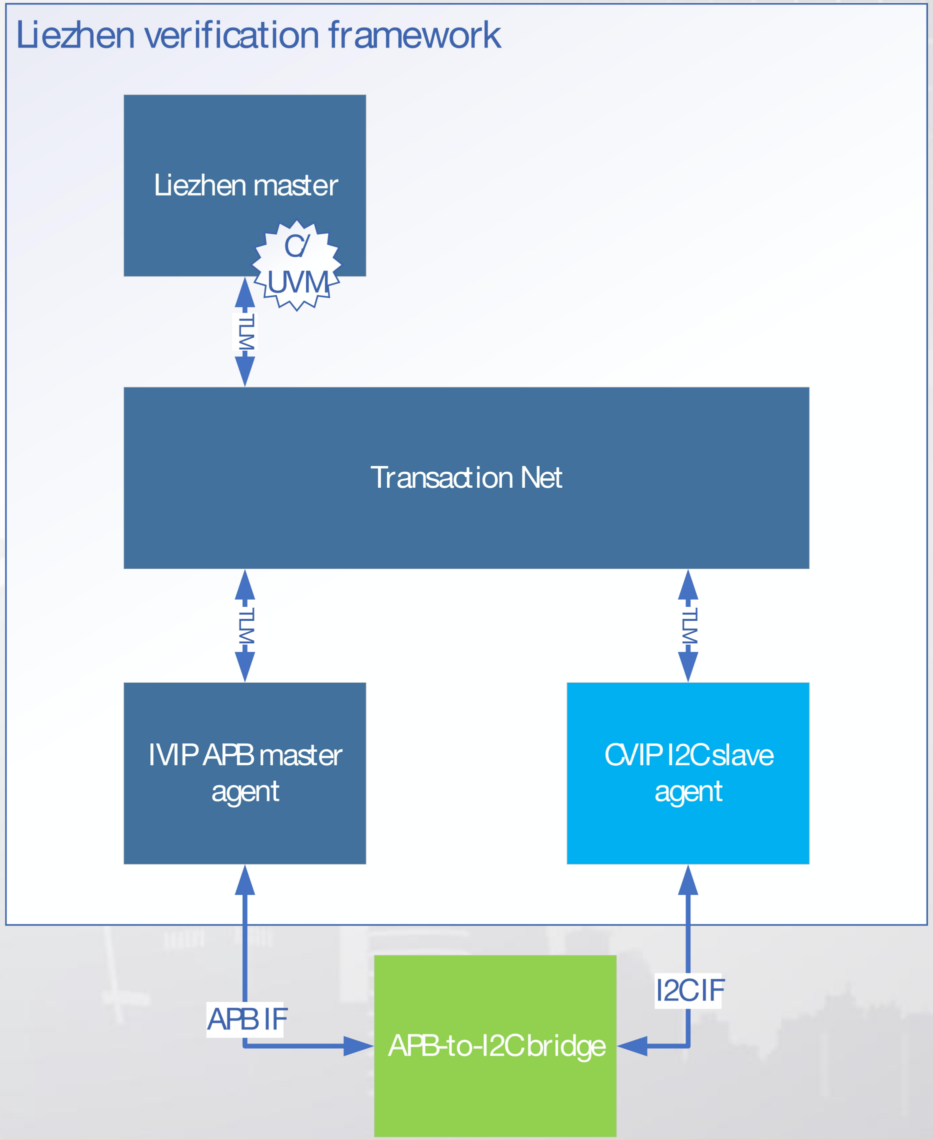
- 大型IC设计公司在储备IP，以此在未来不确定的商业中提高竞争力
- IP的稳定性和易用性在促进产业繁荣，但又使得我们缺失重要IP的研发能力
- 自研（in-house）验证IP（VIP）与设计IP（DIP）一同储备，共同完成交付
- 自研验证IP（in-house VIP, IVIP）可以减少商业验证IP（commercial VIP, CVIP）的使用率，降低成本，完全开放的代码可以进一步加速IDIP验证
- IP储备将是一家优秀设计公司需要重新考虑的地方，VIP与DIP同等重要
- 我们将以I2C接口IP的设计和验证过程为例对IVIP的开发和应用进行详细说明

Characteristics of CVIP
CVIP的各项特性

- 完整的协议特性和可配置性
- 随机约束验证参数和可控制性
- 各项覆盖率包含其中（接口跳转、协议特性）
- 可快速部署以及拓展的验证组件
- 完备的协议特性测试计划和覆盖率映射关系
- 必要的底层单元测试序列
- 可支持错误插入和协议时序检查
- 详尽的用户文档和测试环境集成示范用例

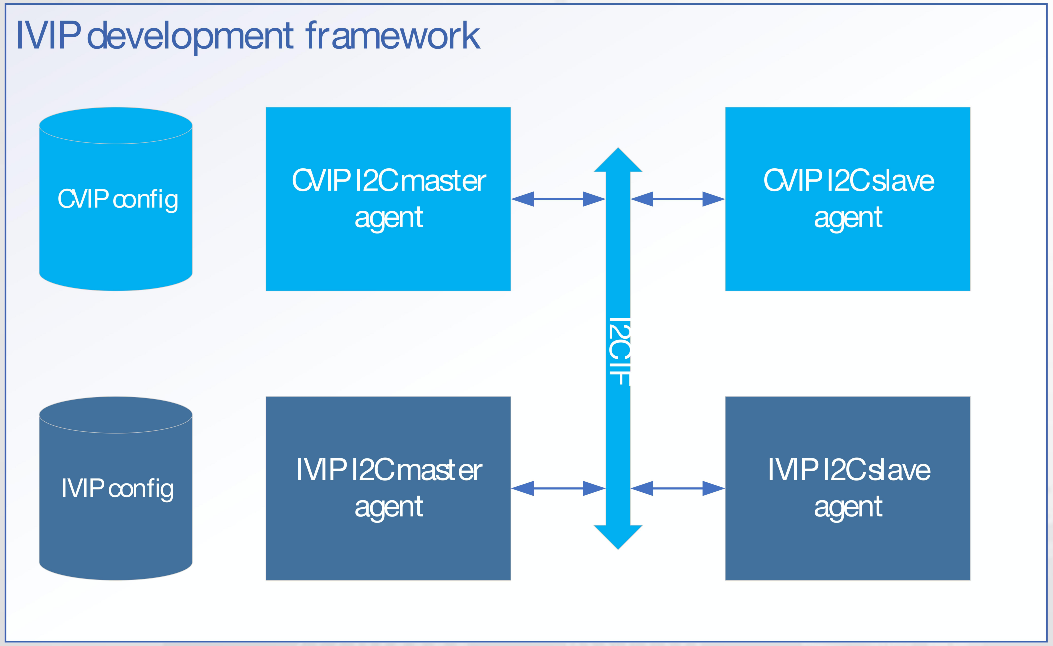
IDIP Development by CVIP
利用CVIP开发IDIP

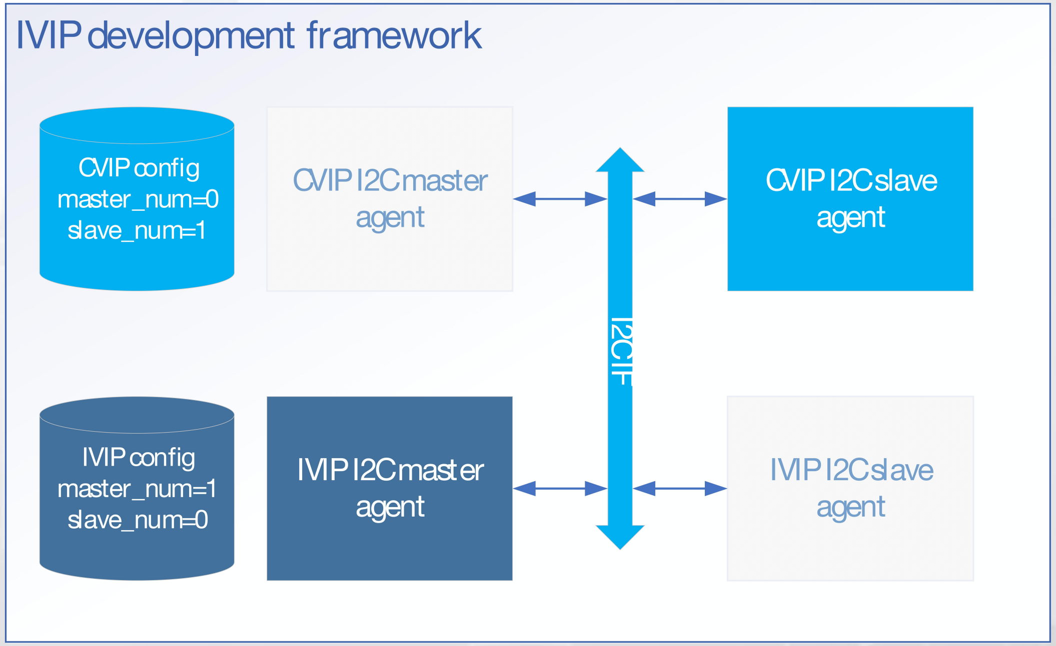
- 在验证IDIP前期，引入CVIP开展验证工作，重点关注设计接口协议和内部功能
- IVIP与CVIP混合方式进行验证环境搭建，将其融合在Liezhen自动化验证平台中
- Liezhen验证平台可以支持SV/UVM/C等多种测试形式和标准测试指令
- Liezhen平台重要组件是自主研发，终端的接口IP在逐步由IVIP取代CVIP



IVIP Development by CVIP
利用CVIP开发IVIP

- IVIP的文件命名和类型命名规范，为以后的组件取代做基础
- 采取统一的验证环境，支持独立开发master, slave, system组件





IVIP Development by CVIP
利用CVIP开发IVIP

- 复用CVIP的验证计划和内置的覆盖率，来完成前期IVIP的各项协议特性开发
- 基于CVIP的测试用例，开发IVIP的参考用例

```
svt_i2c_master_dut_top_level_plan
⊟ Master0
  ⊟ svt_i2c_master_sub_plan
    ⊟ Master_To_Slave
      ⊞ Command
      ⊞ Addressing_mode
      ⊞ Data_size
      ⊞ Start_Byte
      ⊞ Repeated_start_Stop
      ⊞ Second_byte_of_general_call
      ⊞ Slave_addresses_send_by_master__
      ⊞ Cross_of_command__slave_address__start_byte__Repeated_start__Stop
```

```
- tests/
  |- basic_system_tests/
  |- env/
  |- intermediate_system_tests/
  |- master_agent_tests/
  |- sim/
  |- slave_agent_tests/
  |- svt_reference_tests/
  `- tb/

master_agent_tests/
|- seq_lib/
|- tests/
|  |- rkv_master_10bit_nack_addr_test.sv
|  |- rkv_master_7bit_nack_addr_test.sv
|  |- rkv_master_address_slave1_test.sv
|  |- rkv_master_address_test.sv
|  |- rkv_master_address_test_misc.sv
|  |- rkv_master_base_test.sv
|  |- rkv_master_data_size_test.sv
```

Code Refactoring in IVIP Development
在IVIP开发过程中的代码重构

- 一个健康为维护的VIP，在开发中需要分别进行添加新功能和代码重构
- 在开发IVIP高级功能前，我们先对之前的基础功能代码进行重构工作
- 这些可用于IVIP代码重构的经验包括
  - 变量改名
  - 函数改名
  - 添加参数
  - 引入解释性变量
  - 搬移类成员

Code Refactoring in IVIP Development
在IVIP开发过程中的代码重构

- 变量改名

```
semaphore       lock;              //
bit             nack_flag=0;       //
bit is10bits_again=0;
bit same_slv_addr_in10bit=0;
bit [9:0] pre_addr=0;
bit start_flag = 0; // start flag
bit end_flag = 0; // stop flag
bit rs_flag = 0; // repeated start

semaphore restart_cmd_lock;

bit addr_10bit_again = 0; /
bit same_addr_10bit = 0; //
bit [9:0] last_trans_addr =
bit trans_start_flag = 0; /
bit trans_stop_flag = 0; //
bit trans_restart_flag = 0;
bit[2:0]  num_of_retry=0;
int retry_num=0;
```

- 函数改名

```
task start_gen();
task stop_gen();
task rw_slave_7bit_addr(bit[6:0] a
task rw_slave_10bit_addr(bit [9:0
task send_byte(bit[7:0] send_byte


task drive_stop();
task drive_7bit_addr(bit[6:0] add
task drive_10bit_addr(bit [9:0] a
task drive_write_byte(bit[7:0] se
                  ,logic ack_bit = 1'bz
                  ,int hold_data_time =
                  ,int scl_low_time = t
                  ,int scl_high_time =
                  );
```

- 添加参数

```
task drive_highspeed_setup(bit[7:0] send_byte,
                           fm_plus_mode = 0);
  int scl_low_time = fm_plus_mode == 0 ?
  cfg.scl_low_time_fs : cfg.scl_low_time_fm_plus;
  int scl_high_time = fm_plus_mode == 0 ?
  cfg.scl_high_time_fs : cfg.scl_high_time_fm_plus;
  int hold_data_time = fm_plus_mode == 0 ?
  cfg.min_hd_dat_time_fs : cfg.min_hd_dat_time_fm_plus;
  // Highspeed byte code
  drive_write_byte(send_byte,
                  ,.hold_data_time(hold_data_time)
                  ,.scl_low_time(scl_low_time)
                  ,.scl_high_time(scl_high_time));
endtask
```

Implementation of IVIP Advanced Features
IVIP高级特性的实现和测试

2021 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION CHINA
SHANGHAI | MAY 26, 2021
accellera SYSTEMS INITIATIVE

- 在组件中预留事件，便于用户监听，做出自定义响应

```
uvm_event TX_XACT_CONSUMED;
uvm_event EVENT_START_GENERATED;
uvm_event EVENT_STOP_GENERATED;
uvm_event EVENT_ACK_GENERATED;
uvm_event EVENT_NACK_GENERATED;
uvm_event EVENT_ACK_RECEIVED;
uvm_event EVENT_NACK_RECEIVED;
uvm_event EVENT_REPEATED_START_GENERATED;
uvm_event EVENT_START_BYTE_TRANSMITED;
uvm_event EVENT_GENERAL_CALL_ADDR_SENT;
uvm_event EVENT_GENERAL_CALL_SEC_BYTE_SENT;
uvm_event EVENT_ARBITRATION_LOSS_DETECTED;
```

- 预置回调函数便于用户后期做小规模手术处理，免于通过类的覆盖

```
`uvm_register_cb(lvc_i2c_master_driver, lvc_i2c_master_driver_callback)
```

```
task lvc_i2c_master_driver::consume_from_seq_item_port();
  lvc_i2c_master_transaction trans;
  lvc_i2c_master_transaction trans_out;
  bit success = 0;
  bit drop = 0;
  forever begin
    trans = new();
    `uvm_info("consume_from_seq_item_port", "Get request item from sequencer...", UVM_DEBUG)
    seq_item_port.get_next_item(trans);

    `uvm_info("consume_from_seq_item_port", "Got request item from sequencer...", UVM_DEBUG)
    // TODO:: if trans.exception_list or randomized_transaction_exception_list ??
    // branches open here
    drop = 0;
    // callback called for loading exceptions from callbacks
    post_seq_item_get_cb_exec(trans, drop);
```

```
task lvc_i2c_master_driver::post_seq_item_get_cb_exec(lvc_i2c_master_transaction xact, ref bit drop);
  post_seq_item_get(xact, drop);
  `uvm_do_callbacks(lvc_i2c_master_driver, lvc_i2c_master_driver_callback, post_seq_item_get(this, xact, drop))
endtask
```

Implementation of IVIP Advanced Features
IVIP高级特性的实现和测试

- 协议覆盖率可以结合回调函数与monitor进行独立绑定

```
class lvc_i2c_master_monitor_def_cov_callback
  extends lvc_i2c_master_monitor_def_cov_data_callbacks;
  lvc_i2c_master_transaction  xact;
  event xact_event;
  covergroup master_xacts @(xact_event);
    command : coverpoint xact.cmd
    {
      bins write_cmd     = {0};
      bins read_cmd      = {1};
      bins gen_call_cmd  = {2};
      bins device_id_cmd = {3};
    }
  endgroup
  virtual function void master_xact_observed_cov(
     lvc_i2c_master_monitor monitor,
     lvc_i2c_master_transaction xact);
    this.xact = xact;
    -> xact_event;
  endfunction
endclass
```

- 对安全性和稳定性测试有要求的IP，需要支持错误插入

```
typedef enum
{
   DATA_SIZE_LESS_THAN_8_BIT_ERROR
  ,NO_OP_ERROR
  ,START_STOP_ERROR
  ,STOP_IN_BYTE
  ,FALSE_STOP_BEFORE_START
  ,START_FOLLOWED_BY_START
  ,START_IN_BYTE
} master_error_kind_enum;
```

```
task drive_stop_after_start_error(
  output loop_branch_enum loop_branch = LOOP_NO_JUMP);
task drive_stop_before_start_error(
  output loop_branch_enum loop_branch = LOOP_NO_JUMP);
task drive_start_error();
task drive_stop_error();
function lvc_i2c_master_transaction_exception get_exception(
  master_error_kind_enum error_kind);
function void update_exceptions();
task plant_exception_event(
  master_error_kind_enum error_kind);
```

Implementation of IVIP Advanced Features
IVIP高级特性的实现和测试

- 将协议检查方法置于monitor，将检查状态置于error_check_stats

```
class lvc_err_check_stats extends uvm_object;
  typedef enum {
    IGNORE
    , VERBOSE
    , DEBUG
    , NOTE
    , WARNING
    , ERROR
    , EXPECTED
    , DEFAULT
  } fail_effect_enum;
```

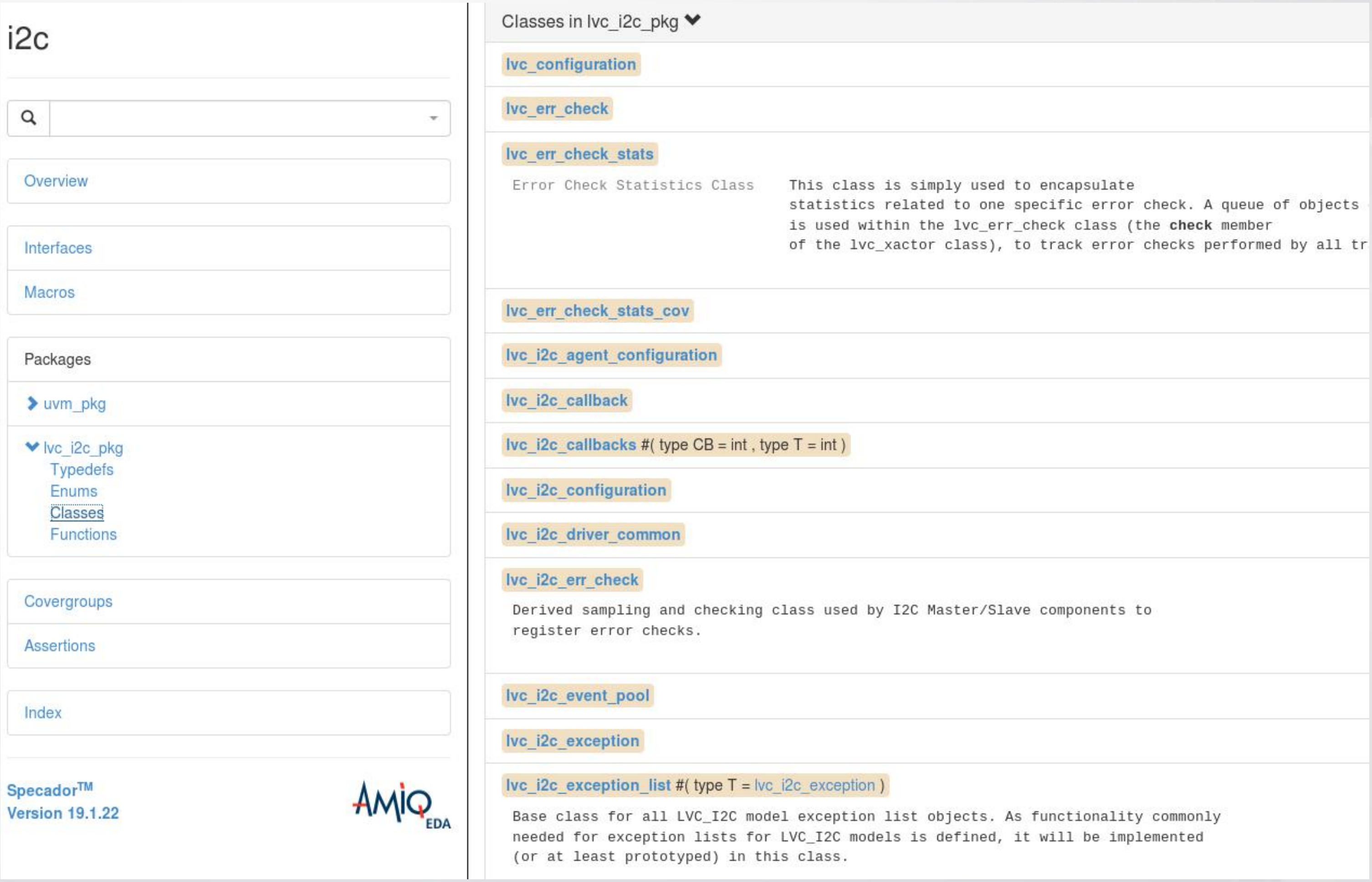
- 通过err_check类来管理多个独立的协议时序检查状态，进行过滤和报告

```
class lvc_i2c_err_check extends lvc_err_check;

  lvc_err_check_stats i2c_dsize_size_8_bit                               ;
  lvc_err_check_stats i2c_not_gen_repeated_start_or_stop_after_nack;
  lvc_err_check_stats i2c_invalid_reserved_slave_id                      ;
  lvc_err_check_stats i2c_ack_received_for_invalid_slave_id              ;
  lvc_err_check_stats i2c_start_byte_followed_by_ack                     ;
  lvc_err_check_stats i2c_master_id_followed_by_ack                      ;
  lvc_err_check_stats i2c_general_call_followed_by_0000_0000             ;
  lvc_err_check_stats i2c_invalid_master_id_transmit_speed               ;
  lvc_err_check_stats i2c_10b_slv_addr_fst_rd_cmd                        ;
```

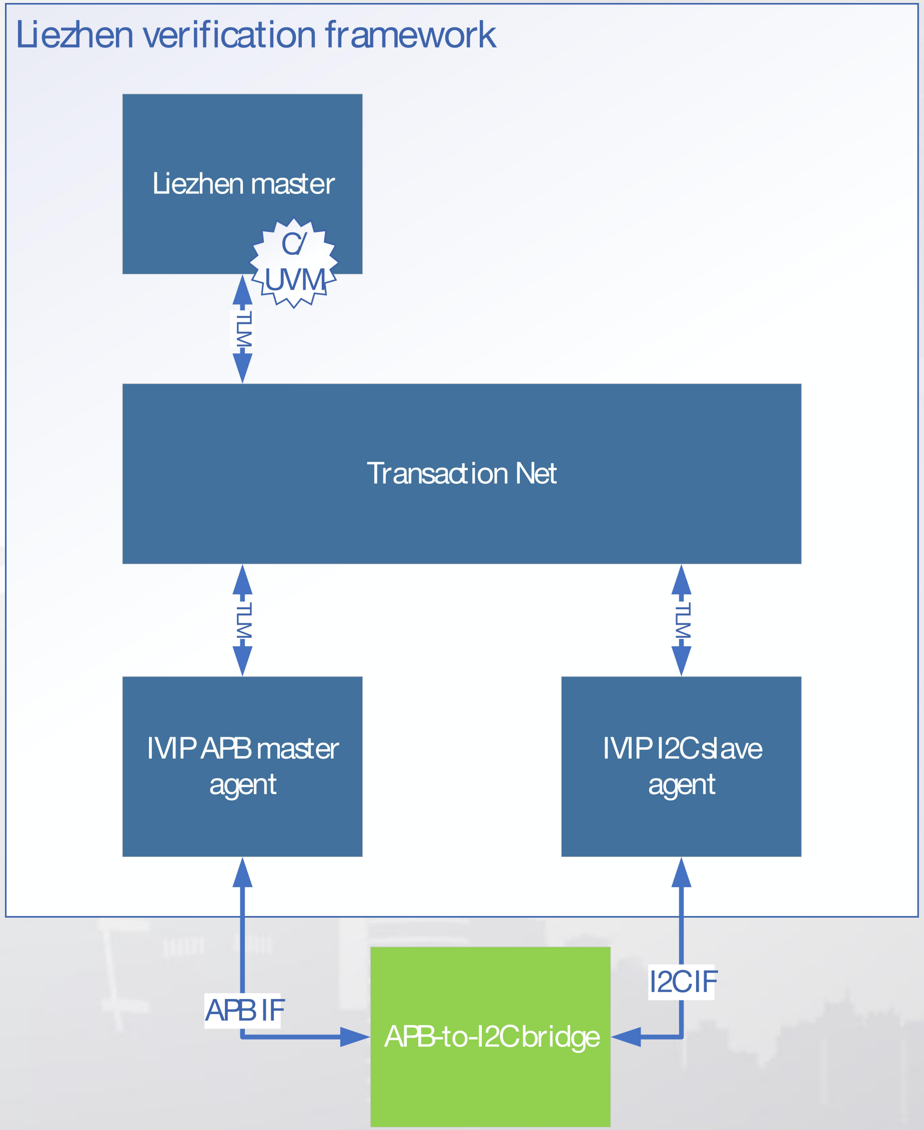
Implementation of IVIP Advanced Features
IVIP高级特性的实现和测试

- 通过特定的代码注释格式（JavaDoc或者NaturalDoc），可以在每次IVIP发布的时候，利用DVT Eclipse工具快速生成相应的文档，有利于客户的阅读使用



CVIP Replacement by IVIP
由IVIP来取代CVIP

- 在完成IVIP开发后，便可将IDIP+CVIP的验证环境由IDIP+IVIP来取代
- 基于更透明的底层验证VIP组件，使得Liezhen验证平台更易于调试和拓展
- 最终将IDIP与IVIP作为一个完整的包进行交付，各自采用独立版本组合发布



Summary
总结

- 通过一个I2C接口设计验证周期来对自研IP开发进行方法学层面说明
- 从验证IDIP、开发IVIP、以及最终交付IDIP + IVIP的过程中看到，完善的开发步骤既可以保证DIP如期验证，同时完成VIP开发，最终实现自主IP交付标准
- IVIP可作为独立验证组件，被集成到其它验证环境，取代CVIP
- IVIP的高级特性使其具备100%替代原有CVIP的能力，实现技术自主化
- 完备的VIP开发流程可以延长我们的验证能力半径，同时也提高了VIP发布的质量和可靠性

DVCon Slide Guidelines

- Use Arial or Calibri font for English slide text
- Use Microsoft YaHei font for Chinese slide text
- First-order bullets should be 24 to 32 point
  - Second-order bullets should be 24 to 26 point
    - Third-order bullets should be 22 to 24 point
- Your presentation will be shown in a very large room
  - These font guidelines will help ensure everyone can read your slides!

~~No Company Logo except on title slide!~~