ICLAB LAB09 EXERCISE

311510125 CHENG-WEI TING

2023/05/17

OUTLINE

- Topic Review
- State Control
  - FSM
  - Counter
- Optimization
  - User unchanged
  - Return judgement
- Reference

OUTLINE

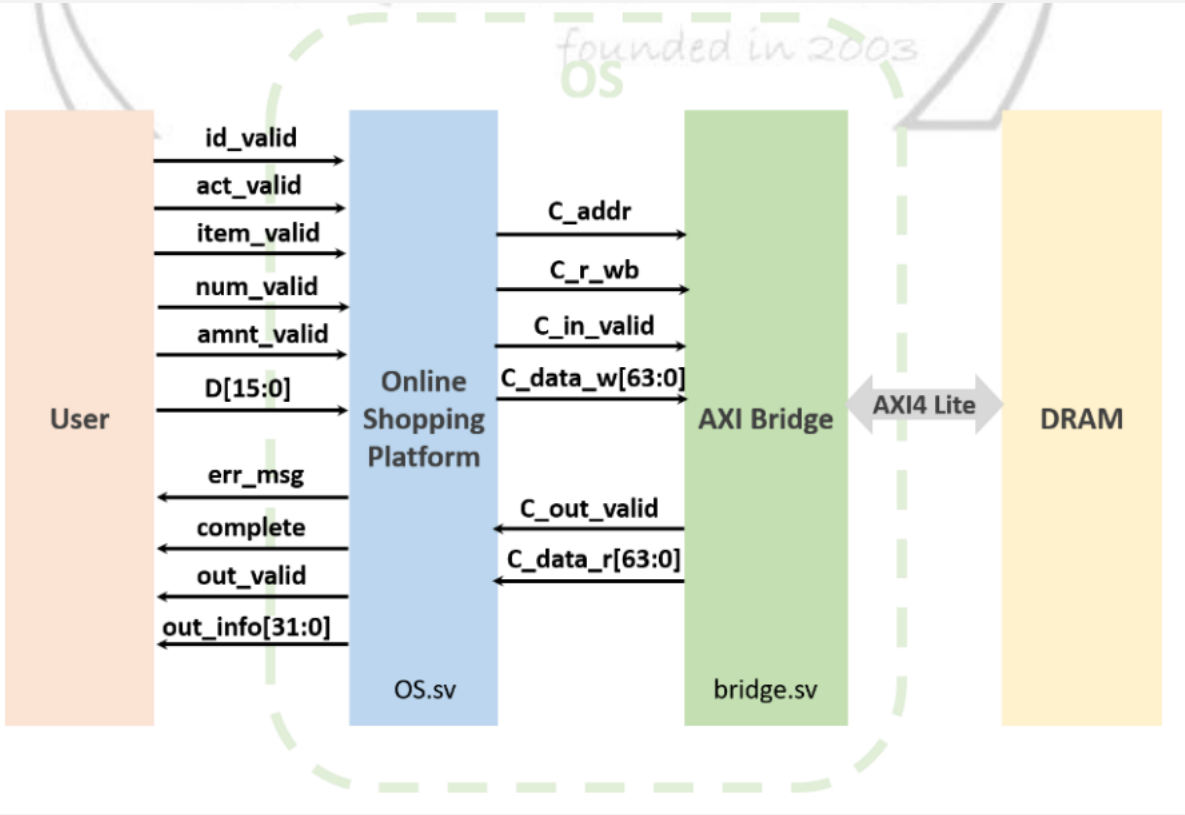
- Topic Review
- State Control
  - FSM
  - Counter
- Optimization
  - User unchanged
  - Return judgement
- Reference

TOPIC REVIEW (1)

ICLabee

Operations

- **Buy**
  Input: (User ID), {item ID}, {# of item}, {Seller ID}
  Output: {User info}
- **Check**
  Input: (User ID), (Seller ID, … if needed)
  Output: {16'd0, user's deposit} or {14'd0, seller's stocks}
- **Deposit**
  Input: (User ID), amount of money
  Output: {16'd0, user's deposit}
- **Return**
  Input: (User ID), {item ID}, {# of item}, {Seller ID}
  Output: {14'd0, user's stocks}



TOPIC REVIEW (2)

- Buy :
  - at most 63 items once
  - fee depends on user level
  - Exp gained after complete operation, upgrade if Exp reaches required Exp
  - 3 types of error :
    1. inventory is full
    2. inventory is not enough
    3. out of money
- Check :
  - Id_valid be valid for 1-5 cycles after act_valid → check seller inventory
  - If not → check user money
  - no error

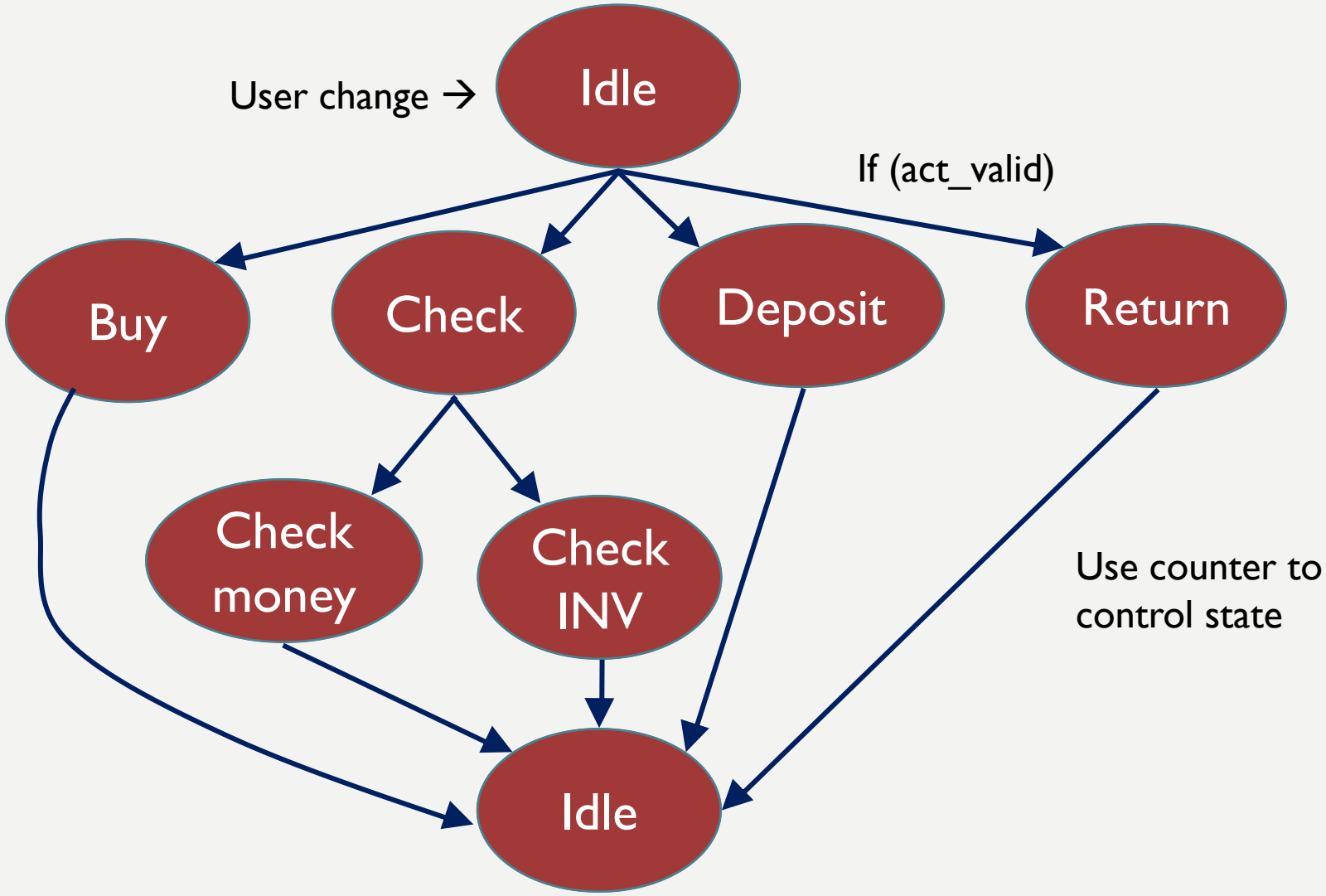
TOPIC REVIEW (3)

- Deposit :
  - 1 type of error : Wallet is full if the amount > 65535
- Return :
  - refund does not include fee, Exp would not change
  - 4 types of error :
    1. Wrong operation : return immediately after Buy & refund to the most recent buyer
    2. Wrong seller ID
    3. Wrong number
    4. Wrong item

OUTLINE

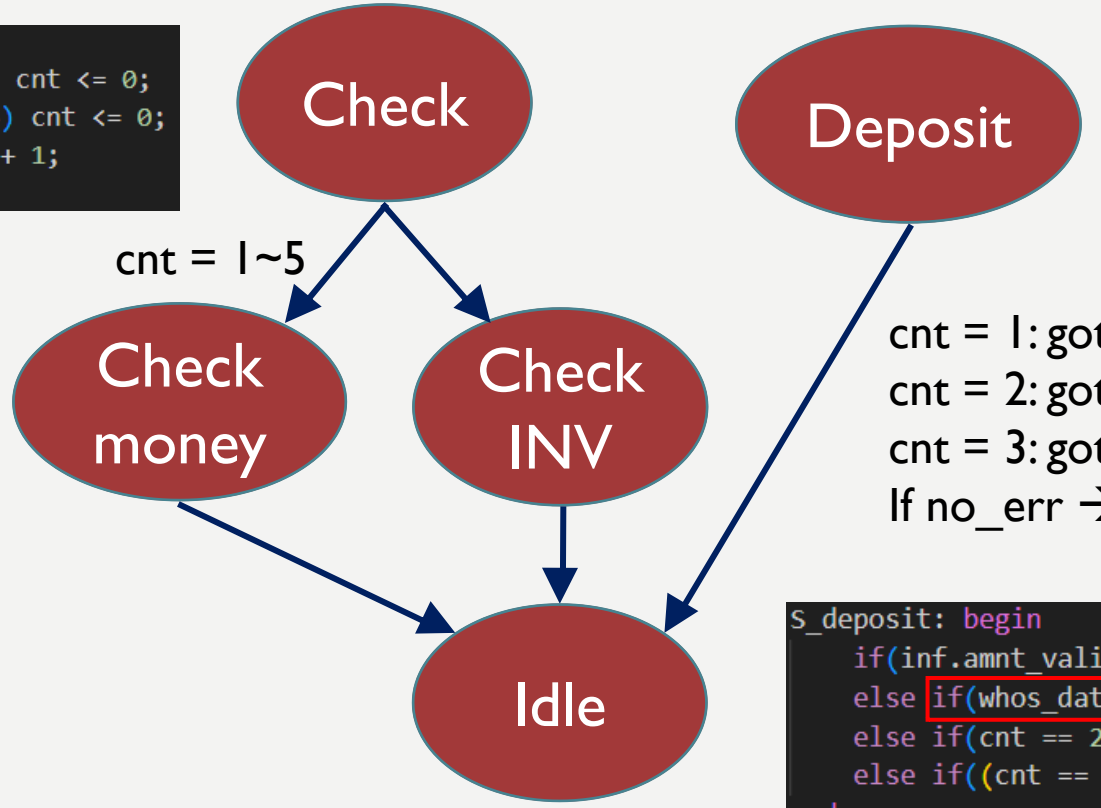
- Topic Review
- State Control
  - FSM
  - Counter
- Optimization
  - User unchanged
  - Return judgement
- Reference

STATE CONTROL - FSM



STATE CONTROL – COUNTER (1)

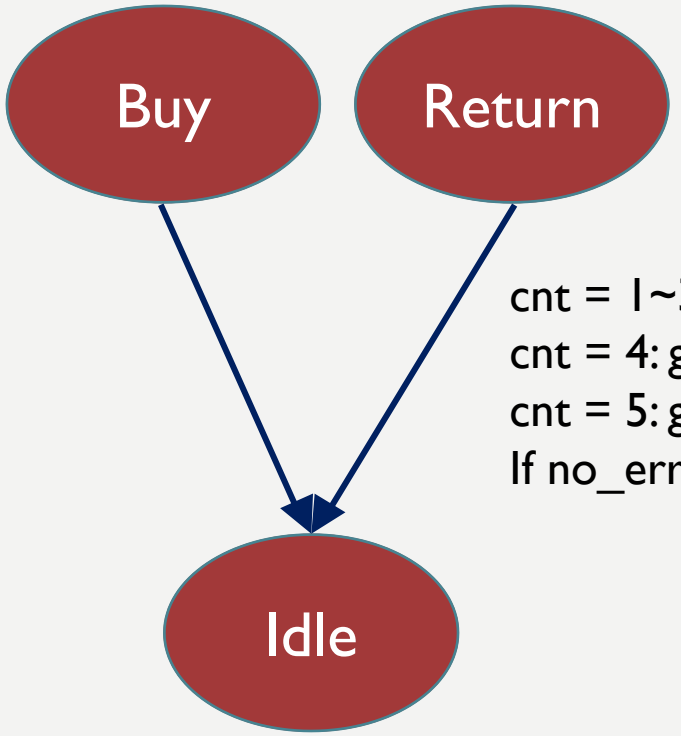
```
S_check: begin
    if(inf.id_valid) cnt <= 0;
    else if(cnt == 5) cnt <= 0;
    else cnt <= cnt + 1;
end
```



cnt = 1: got amnt
cnt = 2: got user info
cnt = 3: got errmsg
If no_err → cnt = 4 : write Dram

```
S_deposit: begin
    if(inf.amnt_valid) cnt <= cnt + 1; //1: get data
    else if(whos_data == 1) cnt <= cnt + 1; //2:start calculation
    else if(cnt == 2) cnt <= cnt + 1;//3:got errmsg
    else if((cnt == 3)&&(temp_errmsg == No_Err)) cnt <= cnt + 1;//4:write
end
```

STATE CONTROL – COUNTER (2)




```
S_buy: begin
    if(inf.item_valid || inf.num_valid || inf.id_valid) cnt <= cnt + 1; //0~3 get data
    else if((cnt == 3)&&(whos_data == 2)) cnt <= cnt + 1; //4:start calculation
    else if(cnt == 4) cnt <= cnt + 1;//5:got errmsg
    else if((cnt == 5)&&(temp_errmsg == No_Err)) cnt <= cnt + 1;//6:write
end
```

cnt = 1~3 : got item / num / id
cnt = 4: got user & seller info (buy: D.d_id[0], return: seller of user's shopping history)
cnt = 5: got errmsg
If no_err → cnt = 6 : write Dram

```
S_return: begin
    if(inf.item_valid || inf.num_valid || inf.id_valid) cnt <= cnt + 1;//0~3 get data
    else if((cnt == 3)&&(whos_data == 2)) cnt <= cnt + 1; //4:start calculation
    else if(cnt == 4) cnt <= cnt + 1;//5:got errmsg
    else if((cnt == 5)&&(temp_errmsg == No_Err)) cnt <= cnt + 1;//6:write
end
```

OUTLINE

- Topic Review
- State Control
  - FSM
  - Counter
- Optimization
  - User unchanged
  - Return judgement
- Reference

OPTIMIZATION - USER UNCHANGED

- whos_data:
  - if(whos_data == 1) → got user(buyer) info
  - if(whos_data == 2) → got seller info → ready to calculate

→ At state idle : if user didn't change → whos_data = 1

→ wouldn't read user again

```
S_idle: begin
    if(user_change) whos_data <= 0;
    else whos_data <= 1;
end
```

```
S_buy: begin
    if(inf.item_valid || inf.num_valid || inf.id_valid) cnt <= cnt + 1; //0~3 get data
    else if((cnt == 3)&&(whos_data == 2)) cnt <= cnt + 1; //4:start calculation
    else if(cnt == 4) cnt <= cnt + 1;//5:got errmsg
    else if((cnt == 5)&&(temp_errmsg == No_Err)) cnt <= cnt + 1;//6:write
end
```

OPTIMIZATION - RETURN JUDGEMENT

- We will not use shopping history of seller

  & pattern would not check Dram data

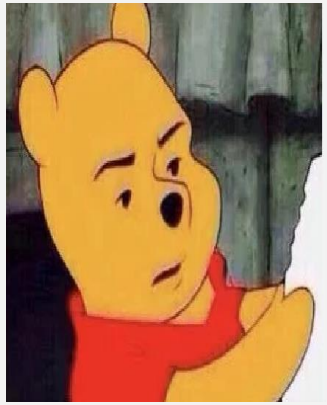
  → Use shopping history of seller to store the most recent buyer

- Return_valid (256 bits) → User's last operation is 'Buy' or not
- Be_returned_valid (256 bits) → Seller's last operation is 'Sell' or not

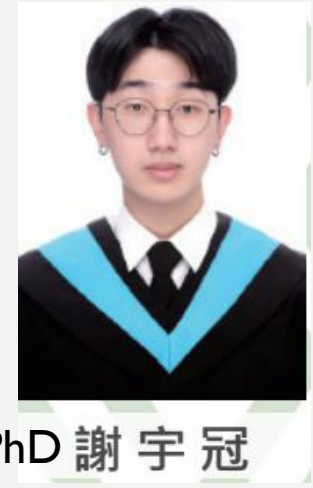
```
if(!return_valid[current_user]) temp_errmsg <= Wrong_act; //buyer do other operation
else if(!be_returned_valid[user_userinfo.shop_history.seller_ID]) temp_errmsg <= Wrong_act; //B do other operation
else if(interactor_userinfo.shop_history.seller_ID != current_user) temp_errmsg <= Wrong_act; //both OK but not recent buyer
else if(user_userinfo.shop_history.seller_ID != interactor) temp_errmsg <= Wrong_ID;
else if(user_userinfo.shop_history.item_num != item_num) temp_errmsg <= Wrong_Num;
else if(user_userinfo.shop_history.item_ID != item) temp_errmsg <= Wrong_Item;
else temp_errmsg <= No_Err;
```

OUTLINE

- Topic Review
- State Control
  - FSM
  - Counter
- Optimization
  - User unchanged
  - Return judgement
- Reference



維尼學長



THANKS FOR YOUR LISTENING