# Execution

**Notation** : `execution`

**Description** : The execution of a transaction defines the state transition function: `stf`. However, before any transaction can be executed it needs to go through the initial tests of intrinsic validity.

## 1 Intrinsic Validity

The criteria for intrinsic validity are as follows:

- The transaction follows the rules for *well-formed RLPs* (recursive length prefixes.)
- The *signature* on the transaction is valid.
- The *nonce* on the transaction is valid, i.e. it is equivalent to the sender account's current nonce.
- The `gas_limit` is greater than or equal to the `intrinsic_gas` used by the transaction.
- The sender's account balance contains the cost required in up-front payment.

Accordingly, the post-transactional state of Ethereum is expressed thus:

```
transaction(post.state) = stf(present.state, transaction)
```

While the amount of gas used in the execution is expressed: `stf(gas_used)` and the accrued log items belonging to the transaction are expressed: `stf(logsbloom, content)(logsbloom, set)` Information concering the result of a transaction's execution is stored in the transaction receipt `tx_receipt`. The set of log events which are created through the execution of the transaction, `logs_set` in addition to the bloom filter which contains the actual information from those log events `logs_bloom` are located in the transaction receipt. In addition, the post-transaction state `post_transaction(state)` and the amount of gas used in the block containing the transaction receipt post(gas_used) are stored in the transaction receipt. Thusly the transaction receipt is a record of any given `execution`.

The execution of a valid transaction begins with an irrevocable change made to the state: the nonce of the account of the sender is incremented by one and the balance is reduced by part of the up-front cost.

`Intrinsic gas` is the amount of gas this transaction requires to be paid prior to execution.

The execution of a valid transaction begins with an irrevocable change made to the state: the nonce of the account of the sender is incremented by one, and the balance is reduced by part of the up-front total cost paid by the sender `valid.transaction = increment.sender(nonce)`.

The `original_transactor` will differ from the `sender` if the `message_call` or `contract_creation` comes from a contract account executing code.

After a transaction is executed, there comes a provisional state[1], Gas used for the execution of individual EVM opcodes prior to their potential addition to the `world_state`[2], and an associated substate [3].

1. `post_execution(provisional.state)`
2. `productive_gas`
3. `substate_a`