svoi
Development
Labs

https://SVOI.dev

Smart contract timer based on FreeTON blockchain messages

SVOI.dev Labs 2020

No Tick-Tock messages aka Timer smart contract.

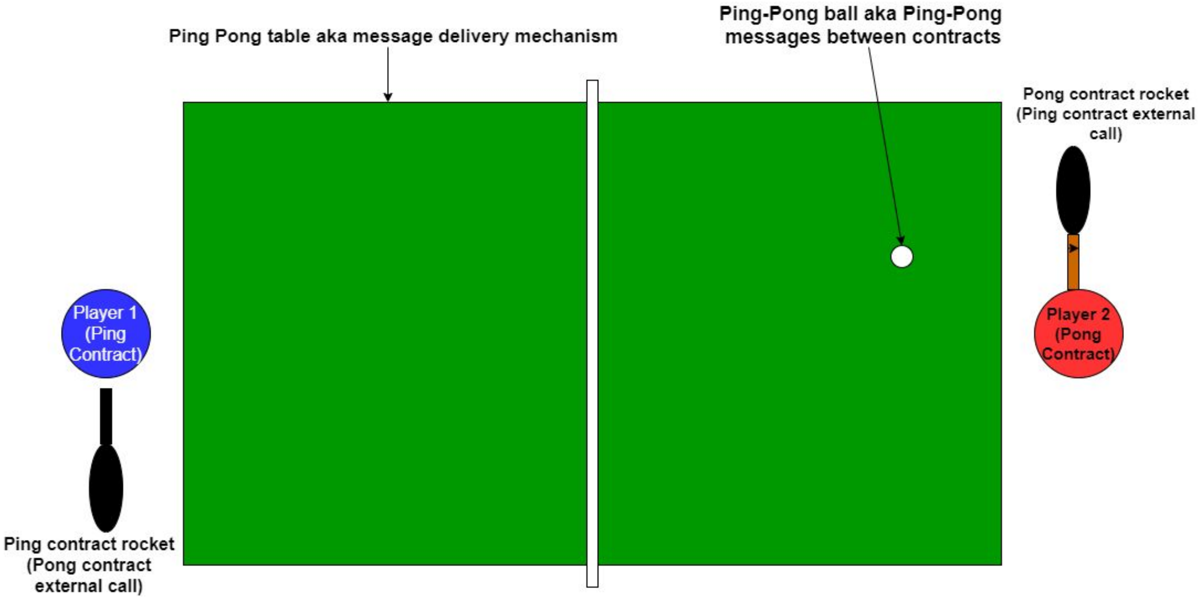
This contest purpose is to create Timer smart contracts, and the most obvious way to do this is by using Tick-Tock messages provided by FreeTon network, but there is a way to create flexible Timer smart contracts that do not need such messages. Instead, it generates similar messages by itself. This mechanism will be described below.

Idea behind smart-contract.

The mechanism of Tick-Tock messages provides an easy way to create Timer contracts, but this mechanism has limitations and most severe is that such messages are sent only to specified smart contracts. So the main goal of this smart contract is to achieve similar behavior without usage of Tick-Tock messages, so everyone could deploy and use timer smart contract for its own needs, i.g. for De-Fi applications. Or maybe the price for waking up by the Timer smart contract that uses Tick-Tock messages will be too high. Who knows?

How does it work?

Have you ever heard about the Ping-Pong? I bet you have. Created smart contract uses it's idea for Timer smart contract realization.
Ping-Pong timer works as illustrated below.


Ping Pong table aka message delivery mechanism
Ping-Pong ball aka Ping-Pong messages between contracts
Pong contract rocket (Ping contract external call)
Player 1 (Ping Contract)
Player 2 (Pong Contract)
Ping contract rocket (Pong contract external call)

So what do we have here?

- There are two smart contracts - Ping contract and Pong contract;
- One contract calls another contract using a known interface. For example, Ping contract calls Pong contract pong() method and Pong contract calls Ping contract ping() method;
- This allows contracts to interchange messages infinitely between contracts (of course until they have enough TONs :)).
- One of the contracts is a helper for another contract, in this realization - Pong is the Leader contract (it stores all the information needed for contract call) and Ping is just responding to Pong messages.

Ping Pong smart contract realization.

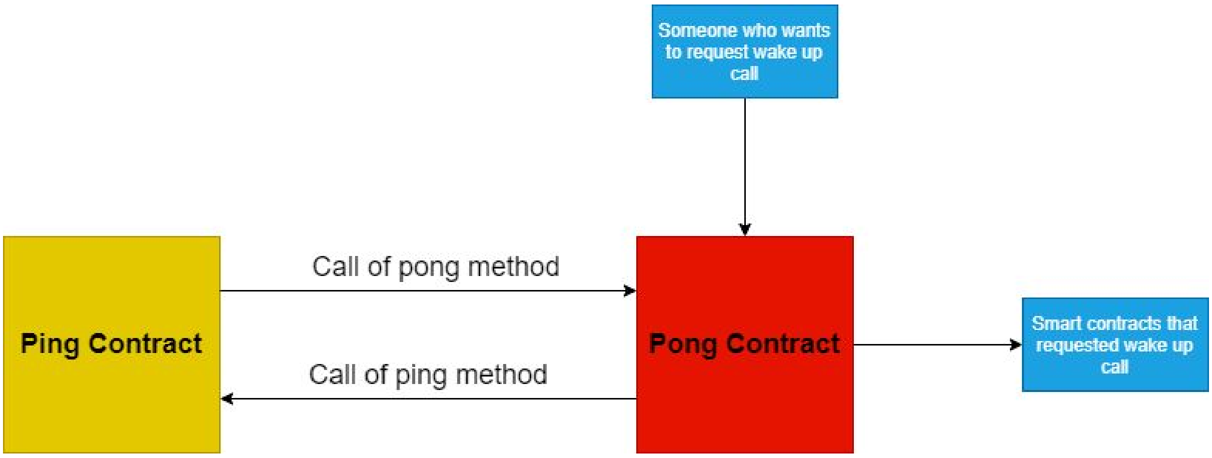
Both of the contracts can be any timer you can imagine (because of it, this realizations uses the simplest of all just to show the mechanism realization). Timer is just a contract that uses a known interface to wake up contracts that requested such action. It stores them in mapping, one contract per one time delta. So if a smart contract implements a given interface it can be called by Ping-Pong timer.

Now let's describe Ping and Pong contracts.

Ping and Pong contracts mechanism.

Both of contracts must store each other's addresses and public keys, because ping() and pong() methods have modifiers onlyPing/onlyPong, i.e. those methods can only be called by Ping or Pong respectively (or by contract deployer if something gets broken).

Ping contract ping() method does the only thing: calls Pong contract pong() method which receives no arguments and returns no value.
Pong contract pong() method is a little bit more complicated: it calls smart contracts that requested wake up call and calls the Ping contract ping() method so the loop can be continued.
This mechanism is illustrated below.


Someone who wants to request wake up call
Call of pong method
Ping Contract
Pong Contract
Call of ping method
Smart contracts that requested wake up call

It implements a kind of endless loop, which can be stopped by setting **pingPong** variable to **false** using **setPing(bool)** method of the Ping contract and **setPong(bool)** method of the Pong contract.
To continue the loop after it is stopped, the deployer of the contract must set **pingPong** variable to **true** and then call **ping()** method of the Ping contract or **pong()** method of the Pong contract.

Wake up interface.

Wake up interface is pretty simple in realization: the developer of a smart contract must implement a single function called **call()** which receives no arguments and returns nothing. Usually smart contracts are woken up to do some routine stuff, i.e. update info, so there is no need for a complicated wake up interface.

Interesting possibilities

(for money making)

If you have two contracts constantly interacting with each other, why don't you make money from this? Just implement a ping-pong interface for them, become a timer and take a commission for it! Sounds interesting, doesn't it?

Conclusion.

This submission purpose is to demonstrate a possible solution for timer realization which does not require Tick Tock messages provided by FreeTON network and thus can be useful for small applications.

There are plans for future development if it seems interesting for juri and community (you can improve chances of it happening by voting for this proposal).

Have a nice day! :)

Links

FreeTon forum login: @ntn

Smart contract realization:
https://github.com/SVOIcom/PingPongTimerContract