DEXHUNE

A GOLD PEGGED DX TOKEN ON AVALANCHE C-CHAIN

BY
ELIX EXO

Abstract

The following document briefly analyzes the design of existing stablecoins and proposes a new way to create price pegged assets. This system is called "Dexhune", and aims to bridge the divide between blockchains and real world assets in an easily replicable and affordable format.

Introduction

A "Stablecoin" is a type of asset on "cryptocurrency" networks that aims to have a specific price, the methods of pegging such prices vary, in the following segment explore two of the major types of stablecoins; algorithmic stablecoins and backed stablecoins (Rawal, 2020)

An Algorithmic stablecoin is one where the token can be minted or burnt rather than redeemed, this allows that asset-A can be burnt to mint more of asset-B and vice versa (Qureshi, 2021). In this situation if asset-A is the stablecoin trading on a CF/CP liquidity pool and its price goes down by 0.05%, a party can buy the stablecoin at the discounted price and burn it to mint asset-B. The conversion system uses a blockchain oracle to determine how much can be minted or burnt, a blockchain oracle is a mechanism for introducing external data into a blockchain, such as prices (Cedro Labs, 2023). Ideally so long as the oracle presents the correct price then all conversions will be accurate.

On the other end of the spectrum are backed stablecoins, the assumption behind these systems is that every token in circulation is fully backed and can be redeemed for the real asset or an asset of equal value, this requires that the issuer have full liquidity for redemptions (The Serenity Research, 2021).

Problem Statement

Stablecoins at their core are unstable, the vast majority of stablecoins use Constant function/product (CF/CP) liquidity pools (LPs) in order to trade on decentralized exchanges that make use of that trading structure, this come with a flaw, CF/CP LPs pair assets based on relative value of the two assets in the pool, if there were two assets ; Token-A and Token-B, with 1000/500 tokens each respectively, it can then be said that Token-B is twice as valuable as Token-A, each purchase of Token-B reduces its presence in the pool, thereby making it more scarce and therefore more valuable, each sale of Token-B into the pool makes it less scarce and less valuable, this applies both ways for each token (Berezon, 2020).

Regardless of if a token is backed or algorithmic they will find themselves at the mercy of such Liquidity pools.

On the other end of the spectrum are centralized exchanges which uses speculative orderbooks, a stablecoin on these exchanges is subject to the whims of "makers" and "takers", who make and take orders to buy or sell the token, but do so at free market rates, thereby allowing the token to be subject to market forces (Coinscapture, 2023).

The third matter of consideration are frontend applications used to interact with smart contracts, smart contracts are Turing complete or Turing incomplete instruction sets or apps that exist on either an EVM or a locking script or some similar setup (sCrypt, 2020 ; Oh, 2020). The frontend or "dapp" allows users to easily interact with the contract (Cardano Foundation, 2020). But this comes with the pitfall of convenience over decentralization, most dapps are not decentralized, they depend on privately controlled domain names which can change what is routed to, along with often centrally hosted software and lastly; API keys.

In order for a Dapp to interact with live data on the blockchain it needs to query it, but to do so requires an RPC connection (Kovacs, 2021). if one does not own the infrastructure needed to run an RPC then they have to use an RPC service and purchase API key subscriptions, if their subscription ends and they are unable to repay then any functionality that their dapp depends on API keys for; becomes unavailable.

Lastly are oracles; because blockchains are sandboxed environments; data cannot get in without a transaction and cannot get out without an API call, oracles use transaction data to push price updates required for redemptions or conversions of a stablecoin (Cedro Labs, 2023). However, if the oracle is compromised or is unable to push the transaction due to an insufficient balance, then the issuer's price will differ from the open market price, thereby opening an arbitrage opportunity that pushes the market price towards the incorrect issuer price, but the panic from such a scenario results in widespread sale of the token coupled with sudden liquidation of leveraged assets.

Proposal

Dexhune proposes a new type of pegged asset that negates the need for speculative orderbooks and CF/CP Liquidity pools. Dexhune proposes a new form of decentralized oracle that scales in difficulty to exploit.

The core of which is an exchange that allows anyone to create their own pegged asset using parity mechanisms and incentives that will be discussed in detail within this document.

Price DAO

The price DAO is the first and most vital piece of Dexhune, it is a decentralized oracle making use of NFTs in a DAO format where holders propose price updates then vote to accept or reject them.

The price DAO presents exchange rates of DXH to AVAX and vice versa. The 'Price DAO' accepts "proposals" from NFT holders, these proposals are numerical values concerning the "Price" data on the smart contract, proposals need "votes" from NFT holders to either pass or be rejected, these votes decide to either accept or reject a price update.

Additionally, each vote is equally counted and yields 20DXH per vote to the proposer from the contract's balance, this reward is only paid if the proposal passes. All voting sessions can be finalized in 5 minutes but are invalid after 10 minutes. The NFT collection used for this price DAO is 'Peng'.

Rewards per proposal are dynamic, this means for every vote cast (both up or down), the contract will pay 20DXH, if 20 addresses vote and the proposal passes, then the proposer gets 400DXH. So the maximum payout should be 20,000DXH if all 1000 Pengs vote.

When considering security it must be noted that if someone were to gain majority vote and decided to set the price to something incorrect, such as; 1000AVAX to 1DXH, that would deteriorate trust in the exchange and drain the contract's liquidity. But the initial mint price for 'Peng' is 14000 AVAX. The cost to exploit determines how much the attacker spends versus how much they can derive from the exchange, because the exchange starts at such a low balance it is not worth the mint price to exploit, rather the only minters would be "True Believers". ideally by the time it becomes profitable to exploit the exchange; all Pengs should already be minted. Once there is a robustness of 'Price DAO' members then it becomes vastly harder to exploit.

The equation for the price is; XAU / 10000 = 0.0001XAU to ##USD.

Then

##USD / Price-AVAX = AVAX to DXH

Price-AVAX / ##USD = DXH to AVAX

So if 1XAU is $1980, 1DXH would be worth $0.198, if AVAX price is $12.710, then;

AVAX to DXH would be : 0.0155AVAX to 1DXH

DXH to AVAX would be : 64DXH to 1AVAX.

Ideally price updates should fill two fields;

AVAX to DXH and DXH to AVAX.

The voter submits the values as;

0.0140 : 1 ; 64 : 1

Observe that the price field can accept arbitrary data such as "three dollars and fifty cents" which is a string completely unusable by any contract that depends on the DAO.

Nonetheless, because votes happen so often the 'Price DAO' would need to be used alongside an automated system, a prototype of which will be created alongside the DAO.

Once fully developed it can be projected that the ideal price for a 'Peng' is; (Estimated value from exploitation / 1000). And this scales as the project grows because to capture a Peng is to capture 1/1000th the potential to drain the Liquidity.

Moreover, much like Byzantine fault tolerance, the system incentivizes participants to behave in the best interests of the network. If a party could make comfy profits from voting in addition to dividends for supporting the Price, whereas attempting to exploit the system could result in major losses if the cost of Peng is far greater than the amount gotten from the exploit; they would be more inclined to behave.

The rarity of Pengs is integral to their security, and over time as 'Pengs' get lost or confiscated they will become even more expensive due to scarcity.

Lastly; all value from mint is used as liquidity in the Dexhune exchange contract, thus serving a dual function.

DXH Token

The second smart contract is the DXH token, a time based dividend token.

Parameters

Name : Dexhune

Ticker : DXH

Price : 0.0001XAU (Gold Troy Ounces)

Decimals : 0

The DXH token is an ERC-20 which at certain intervals will mint a fixed amount of tokens distributed to holders based on how much they own, the amount distributed is always 0.12% of existing supply every 4 days.

For this to happen the 'Mint' function has to be called. Dividends cease after 40 years (after 315360000 blocks or being called 3650 times). Each mint interval in blocks is; 86,400 blocks.

PerpetualMintToExchange

In addition to dividends the contract mints rewards to the 'Exchange Contract', minting 300,000DXH each time, this is available every day or every 21600 blocks, and can be called in perpetuity.

PerpetualMintToDAO

The third mint function specifically targets the 'Price DAO', minting 300,000DXH every 21600 blocks, and can be called in perpetuity.

The dividend rate is; 10.95% APY with compounding value per mint.

It is to be noted that the token has no decimals, this makes it impossible to peg against other assets on CF/CP LPs or orderbooks.

Notwithstanding, in 40 years time, even after dividends cease, rewards to the oracle and Exchange continue at a fixed rate of 300,000DXH each. This amounts to 21,900,000,000 in 100 years from launch and 219,000,000,000 in 1000 years.

The estimated total supply increase from dividends is; 796,286,621. And with that all matters concerning 'Mint' are covered.

Finally, When the DXH contract is created a total of 10,000,000DXH is minted to the deployer, 90% of this will be deposited into the 'Price DAO' at launch to make up its functional balance, whereas 10% will be sent to the 'Exchange Contract'.

Dexhune Exchange

The third smart contract is the Dexhune Exchange, which allows (3) modes; 'peer trading', 'mass settlement' and 'Mass RE-settlement'. Using oracle-based price mechanisms.

Peer Trading

Peer trading allows that users put up their tokens for a predetermined price and takers accept these offers in order to gain rewards in a minted token, this ensures low slippage and no fees.

Trades occur as follows; users deposit their token into the 'Exchange contract' pending a trade, this stores the trade details in the contract, within 1 of 100,000 "cache slots".

takers can then accept certain trades, thereby sending their token for swap into the 'Exchange', this causes the contract to settle both parties and reset the cache slot for that particular trade.

Each taker is rewarded 2DXH per trade up to a maximum limit of the amount of DXH in the 'Exchange' Balance.

Trades are completed once the taker's end is fulfilled, the settlement and reward are done in the same transaction.

Trades can equally be cancelled before they are executed, or cancelled if they last longer than 6 blocks.

Clearing abandoned trades has to be done by calling the 'Check Trades' function. Each 'cache slot' has its block height attached, if the height is older than 6 blocks then it is erased and the funds returned. The function can only be called once every hour. The fee for returning all funds is paid by the caller, this could be quite large depending on the number of pending trades.

Listed tokens and DXH can only be traded for AVAX and vice versa. Any cross-swap will have to be manually done.

Order Taker rewards are only issued for trades of DXH.

Taker Rewards

The reward rate begins for "takes" of 2,000DXH, that is; if a taker posts 2,000DXH upwards for a DXH order. Any order less than 2,000DXH is not eligible for rewards. Orders of 20,000DXH yield 4DXH in rewards, whereas orders of 200,000DXH yield 8DXH.

This also applies for 'takes' of AVAX to DXH.

Cache slots

Cache slots are pieces of data stored within the contract with a limited number of entries of no more than 100,000, each slot stores the details of the particular token contract for trade, along with the amount for trade, maker address and the block height. Once the trade is complete the data is erased.

If all slots are full then the contract rejects the order.

Pricing

Price for each trade is gotten by checking the 'Price DAO', this shows the exchange rate of AVAX to DXH and vice versa.

Each DXH is valued at 0.0001 XAU (Gold Troy Ounces).

The price for DXH is enforced on all trades, all tokens listed are either at a price relative to DXH or at parity. Parity means the contract checks how much DXH there is in the token's stated parity address and determines the token's price relative to DXH and enforces that for trades of the target token within the 'Exchange contract'. Tokens can be traded for AVAX.

Nonetheless, the value received from each trade is determined when the trade is initiated, this means the exact amount for trade will not change regardless of what happens to the price of either token.

Listing

The default token is AVAX and [DXH], the [DXH] token is set with a special "OwnerOnly" function called 'SetOtherToken' which adds the token's contract address.

New tokens can be added by the creator of those tokens. Adding a token creates a separate "cache slot" specifically for it, stating its contract address, parity address (if applicable). There will be a total of 1,000,000 listing slots, making them somewhat scarce.

listing will cost an increasing amount of DXH, each time a new token is listed the cost increases by 0.5%, with a starting price of 1000DXH, and a maximum price of 1,000,000DXH. All DXH from listing fees are held within the contract and are eventually used to reward traders or issue 'mass settlement'.

Parity

Checking parity just means the listed token has a stated "parity address" that holds DXH and [TOKEN].

Each time a transaction is initiated for a parity token the exchange calls 'balanceOf' at the DXH token contract in regards to the stated token's parity address, then does the same at [TOKEN]'s contract address, thereby getting the balance of DXH and [TOKEN] held by the parity address.

Once the 'exchange contract' checks the balance of both DXH and the listed token, it then calculates their relative value based on how much of either token is present.

This is done by dividing the amount of each token, if there is 1540DXH to 240TOKEN, then the price is; 6.416 to 1. But because there are no decimals the value is rounded down to 6DXH to 1TOKEN. To get the inverse it divides TOKEN amount by DXH amount, producing an output of 0.155 to 1. Rounded down this is 1 to 1.

The complete price data is; "6 : 1 ; 1 : 1".

Granted this is an inexact representation of value.

Mass settlement

Mass settlement is a trading structure exclusive to DXH trades, this allows that funds for all trades *from* or *to* DXH are pooled and settled upon initiation of a new DXH maker transaction.

This works as follows; if user-A wants to BUY 1000DXH, and enough DXH exists in the contract to settle the trade then it is automatically settled. If they want to SELL 10AVAX worth of DXH and enough AVAX exists within the contract then it automatically settles.

Mass settlement is only triggered by new order 'takes', this means each time someone tries to 'take' an order to buy or sell DXH to or from AVAX they also have to pay for the computation of checking pending mass settlements and settling them (this cost is specific only to taking orders for DXH and does not affect making DXH orders).

In conjuncture with mass settlement, takers can still voluntarily complete trades to gain rewards.

However, because the exchange contract holds a great deal of DXH (from daily mints), this means most trades *to* DXH will be instantly settled with no counterparty.

This accumulates AVAX in the contract, all acclimated AVAX is used to settle DXH trades *to* AVAX, irrespective of counterparty.

Any AVAX sent to the contract for whatever reason becomes part of this liquidity pool.

Partial Settlement

This allows takers to chip off smaller amounts of a maker's size in the absence of 'mass settlement'.

Mass Settlement and Mass RE-Settlement only deal with whole orders and cannot "chip" orders, if a maker puts an order to sell 1,000,00DXH while only 50,000DXH worth of AVAX exists for 'mass settlement', 'Mass settlement' will not trigger for that order.

In this scenario order takers are taking orders of 2000DXH sells only (due to their profit motive). Each 2000DXH "take" will update the 1,000,000DXH order and "chip" away at it. The update reduces the amount for trade from 1,000,000 to 998,000, then credits the maker with the respective AVAX and the taker with 2000DXH.

This will happen each time a taker sends a lower amount to the contract under the 'takeOrder' function, the contract will seek outstanding orders and chip them, thereby updating the order details and settling the chipped amounts to the respective addresses.

In conclusion to this segment; 'Partial Settlement' is available to all tokens.

Mass RE-settlement

Mass Settlement as previously discussed greatly enhances DXH capital efficiency, however the same cannot be said of other tokens, Mass RE-Settlement brings this benefit to other listed tokens but in a more controlled manner.

Mass Re-settlement makes use of AVAX or [TOKEN] from pending maker orders for a particular token.

For example; if Maker-Group-A attempts to BUY a sum total of 200AVAX worth of [TOKEN] at the same time as Maker-Group-B attempts to SELL a sum total of 100AVAX worth of [TOKEN], the contract will automatically connect the maker buy/sell orders and RE-Settle them. In this scenario 100AVAX worth of [TOKEN] is settled, any orders which cannot be wholly settled are ignored.

Mass RE-Settlement is triggered for a particular token each time a new taker transaction is made, this means if there are groups of makers attempting to buy/sell, their orders will not be automatically connected until a taker joins the pool. This also means the taker pays the necessary fee for the compute of Mass RE-settlement.

The contract will prioritize a user's order 'take' in this process, meaning if the taker was attempting to BUY 10AVAX worth of a 200AVAX sum total of SELLS, the contract first settles 10AVAX worth of [TOKEN], updating partial or whole orders before attempting to mass RE-settle. All this happens under one transaction. This distinction should also be noted for 'Mass Settlement'.

Although 'Mass RE-Settlement' mainly targets maker orders, it can also act on pooled tokens within the exchange contract.

Example; User-A wants to buy 1000 units of [TOKEN] and 2000 units exist within the 'Exchange Contract' from unknown deposit(s), the contract will consider this a valid source to settle buy orders. Essentially, all units of a listed token within the 'Exchange Contract' are valid for 'Mass RE-Settlement'. As opposed to AVAX which is limited only to AVAX deposited during a maker transaction for that particular listed token.

This concludes the specifications on the 'Exchange Contract'.

Frontend

The Dexhune frontend will be somewhat opaque compared to contemporary Dapps, this is because it will make use of a "Blind" design, this means it will not query the blockchain for address information or smart contract states, rather each transaction is built using preset bytecode and user input, then presented to the user's wallet to sign and broadcast. This means if the user inputs incorrect data; the transaction can fail.

This "Blind" design is considered to eliminate a critical security hole in purchasing, renewing and owning API keys.

The second unique feature of the frontend is the hosting, the frontend will be immutably hosted using IPFS, however, this document proposes 'SSHS' or "Star Side Hosting Service', in simple terms; it allows the disclosure of IP addresses associated with IPFS hosts of a certain item, this IP in conjuncture with a wallet public key are stored with the SSHS client once installed, the public key is provided by the hosting party, thereby they can be identified and tipped for their service.

The third matter of consideration is the domain name renewal contract, which is discussed as follows;

Renewal Contract

This system will use Avvy Domains. The Avvy name for Dexhune will be held by a 'Renewal Contract', which is an NFT collection called 'Marker Fragments' with 100,000 items, they have an initial cost of ~$5 worth of AVAX. But each time 'Fragments' are minted; the cost of mint increases 10%, with a max price of 100AVAX, all AVAX is stored within the contract.

The contract uses the stored AVAX to renew the domain name. Approximately every 6 months any user will be eligible to call the 'renew' function, which prompts the contract to renew the Avvy name for 6 years, as a reward fo calling the 'renew' function, the contract grants 1% of any AVAX it holds to the address that called the function.

Conclusion

In the above document, issues regarding existing stablecoins were discussed, and a potential solution was proposed. Dexhune aims to open up the pathway for innovators to create any sort of price pegged asset and create a more harmonious de-fi environment.

References

Cardano Foundation, (2020) "An Introduction to Decentralized Applications" https://medium.com/cardanorss/an-introduction-to-decentralized-applications-d0fb4f961647

Cedro Labs, (2023) "How do Oracles Work?" https://medium.com/cedro-finance/how-do-oracles-work-12ddff3d41bc

Coinscapture, (2023) "Crypto Orderbooks: How do they work?" https://medium.com/@coinscapture/crypto-order-books-how-do-they-work-7f72d27d3104

Dmitriy Berezon, (2020) "Constant Function Market Makers: DeFi's “Zero to One” Innovation" https://medium.com/bollinger-investment-group/constant-function-market-makers-defis-zero-to-one-innovation-968f77022159

Franciska Kovacs, (2021) "Why Are RPCs So Important In Blockchain Development? 5 Use Cases" https://medium.com/ankr-network/why-are-rpcs-so-important-in-blockchain-development-5-use-cases-60a16a02c143

Haseeb Qureshi, (2021) "A Visual Explanation of Algorithmic Stablecoins" https://medium.com/dragonfly-research/a-visual-explanation-of-algorithmic-stablecoins-9a0c1f0f51a0

SCrypt, 2020 "Introduction to Bitcoin Smart Contracts" https://medium.com/@xiaohuiliu/introduction-to-bitcoin-smart-contracts-9c0ea37dc757

Se Jin Oh, 2020 "What is a Smart Contract? A Beginner's Guide to Blockchain" https://medium.com/haechi-audit/what-is-a-smart-contract-5fa0d32939af

The Serenity Research, 2021 "Overview of Asset Backed Stablecoins" https://medium.com/coinmonks/market-info-overview-of-asset-backed-stablecoin-7e111488e4af

Yogesh Rawal, 2020 "Complete guide to Stablecoins" https://medium.com/akeo-tech/complete-guide-to-stablecoins-in-2020-1f37b7e11d9d