Norges Bank

3 | 2026

Norges Bank Papers

Central bank digital currency – experimental testing in project phase 5

Summary

According to Strategy 25, Norges Bank should "[…] *prepare the ground for the issue, if appropriate, of a central bank digital currency (CBDC). A CBDC may be necessary to ensure that NOK will continue to be a safe, efficient and attractive means of payment in the future too*." (Norges Bank, 2022)

As a follow-up to this strategic objective, Norges Bank established a project to explore the need for introducing a central bank digital currency (CBDC[1]) and, if so, how a CBDC solution should be designed[2]. The project assumed that the solution would be based on blockchain technology[3] so that the benefits of this technology could be utilised. As part of the investigation, the project conducted technical and functional testing in two technological sandboxes. The aim of the testing was to gain insight into how blockchain technology can be used to promote a safe, efficient and attractive payment system, and to highlight the benefits, challenges, and drawbacks of different technical alternatives.

The two sandboxes are based on two different ledger[4] platforms, connected via a bridge that enables the transfer of tokens[5] (CBDC, tokenised bank deposits (TBD), and tokenised securities) between them. The test programme in this investigation phase includes various forms of payments with retail CBDC between individuals, the use of wholesale CBDC for settlement of payments between banks, payments with TBD with central bank settlement in tokenised reserves, and the use of ledger technology to automate and streamline government debt management.

The test cases were modelled on key processes in the payment and financial system, but carried out in a simplified and adapted manner to test the characteristics, benefits, and challenges of various alternative technological designs. The test programme did not include capacity tests, stress tests, or the technology's ability to handle abnormal events and attacks from malicious actors.

The main findings can be summarised in three points. Firstly, a single-ledger[6] architecture, where CBDC, TBD, and tokenized securities are registered on the same ledger, offers

[1] Central bank digital currency (CBDC) *is a digital form of central bank money issued by the central bank and represents a direct claim on the central bank*.

[2] See Norges Bank (2026).

[3] Blockchain technology: A type of distributed ledger (DLT) in which transactions are stored in 'blocks' that are cryptographically linked together in a chain and updated/synchronised between multiple participants without a single shared central database. 'A blockchain is a decentralised and distributed digital "ledger" that makes it possible to record, track, and visualise all digital transactions' (Store norske leksikon, 2026).

[4] The term "ledger" is used in the report as a collective term for a shared, distributed, and cryptographically secured register (DLT), and includes both blockchains and other variants of distributed ledgers/registers.

[5] Token/tokenisation: Tokenisation involves representing money, securities, or other rights as digital tokens on a *ledger*, so that ownership and transfers can be handled in a programmable way.

[6] Single-ledger: An architecture where all digital assets are tokenised and managed on a single DLT/ledger. Transfers and settlements take place within the same ledger, without the need for multiple DLTs or bridges between them.

advantages. Specifically, the architecture enables atomic[7] settlement across assets, eliminates counterparty risk, provides a high degree of traceability, and simplifies technical compliance with certain regulatory requirements. Secondly, multi-ledger[8] architectures — where different assets such as CBDC, TBD, and securities are registered on different ledgers and moved between them using bridges[9] — increase technical and operational complexity and present a larger attack surface. Delivery versus Payment (DvP)[10] can be achieved, but when values are settled across different ledgers, this does not occur simultaneously. This means that atomic settlement cannot be achieved with such a design. This increases the need for governance, monitoring, and standardisation. Thirdly, the most significant benefits of CBDC and ledger technology will only be realised if actors other than the central bank itself can register their own assets on the ledger and develop solutions that use CBDC as a means of settlement.

This report documents the technical design, implementation, and findings from the testing in the sandboxes. The report is a technical elaboration and supporting report to the project's final report[11], and does not contain recommendations on the introduction of CBDC.

The source code that forms the basis for the test systems in the sandboxes will be published as *open source* and can be found in our github repository[12].

[7] Atomicity means that a compound transaction is either executed in its entirety or not executed at all. All relevant state changes, such as the transfer of money and securities, are treated as a single indivisible operation. There are no partial or intermediate states where only parts of the transaction are completed. Atomicity thus eliminates the risk of one party delivering without the consideration being fulfilled simultaneously.

[8] Multi-ledger: A multi-ledger architecture means that different assets or actors use separate ledgers, and that settlement between them requires coordination or bridges.

[9] Bridges: A technical mechanism that connects two (or more) separate DLTs so that tokens/values and relevant messages can be transferred between them, typically by locking/burning on one ledger and issuance/disbursement on another.

[10] Delivery versus Payment (DvP) is a settlement principle whereby the delivery of an asset and payment occur simultaneously and are conditional on each other, so that neither party is exposed to counterparty risk in the settlement. In traditional systems, DvP is achieved through coordination between separate securities and payment systems. In DLT-based solutions, DvP can either be implemented as a single technical atomic transaction when money and assets are represented on the same ledger (single-ledger), or as an orchestrated process across multiple ledgers, where simultaneity and finality are not achieved through a single technical operation.

[11] See Norges Bank (2026).

[12] See Github (2026)

Table of Contents

Introduction

According to Strategy 25, Norges Bank should "[...] *prepare the ground for the issue, if appropriate, of a central bank digital currency (CBDC). A CBDC may be necessary to ensure that NOK will continue to be a safe, efficient and attractive means of payment in the future too*." (Norges Bank, 2022, p.10)

As a follow-up to this strategic objective, Norges Bank established a project to explore the need for introducing central bank digital currency (CBDC) and, if so, how such a CBDC solution should be designed.[13] The project was based on the premise that a CBDC solution should be based on blockchain technology so that the benefits of this technology could be utilised. As part of the exploration, the project carried out technical and functional testing of the technology for payment and other financial services in two technological sandboxes[14].

This report describes the technological choices underlying the establishment of the sandboxes. It also explains how the sandboxes were established and how the technical and functional testing was carried out. Finally, the main findings from the testing are summarised.

The overall objective of the technical testing has been to obtain information relevant to assessing whether the introduction of CBDC based on blockchain technology will contribute to an efficient and secure payment system. The testing was intended to provide insight into the benefits, challenges, and disadvantages of various technical choices, including how money, securities, and other claims can be represented in tokenised form on a ledger, how values can be transferred between ledgers, how settlements can be automated using smart contracts[15], and how roles and responsibilities can be technically enforced on a common technological platform.

The technological sandboxes are established as isolated test environments without connection to production systems. The sandboxes were developed in collaboration with two external IT vendors. They are based on two different ledger platforms and are connected by a technological bridge enabling the 'movement' of tokens between the

[13] See Norges Bank (2026)
[14] A technological sandbox is a confined test environment where technology and functionality can be tried out safely, without affecting real systems or data.
[15] Smart contract: *programme code that runs on a ledger and automatically enforces rules, rights, and transactions when predefined conditions are met*.

ledgers by locking value on one side and issuing a representation of the value on the other ledger[16].

DLT, blockchain, and ledger

In this report, the term 'ledger' is used to refer to a blockchain for recording ownership and transactions based on technology that falls under DLT (Distributed Ledger Technology). DLT is a shared, distributed, and cryptographically secured register, and also includes other variants of distributed ledger besides blockchains. The ledger solutions tested in the project have not necessarily been shared or distributed between independent actors. The blockchain may just as well be centralised or partially distributed within a controlled and permissioned setup.

The sandboxes were used to experiment with different architectures and mechanisms, setting aside the full complexity of today's infrastructure. Payments, settlement and securities trading, and government securities lifecycle management are modelled to resemble key elements of current systems and processes, but have been deliberately simplified and generalised. Consequently, the tests do not cover all details of existing processes, regulatory requirements, or interfaces with external systems.

The testing did not include load and stress tests or the technology's ability to handle error events or attacks from malicious actors. The focus has been on expected use and operation under normal conditions ('happy path'), architecture choices, rights and role frameworks, and the interaction between ledgers, smart contracts, and adjacent system components, such as bridges.

The testing approach has been twofold. Firstly, we wanted to explore the applicability of the technology in realistic use cases based on practical needs and existing processes in the payment and financial system. Secondly, we wanted to use different solutions and architectures in each scenario to identify the strengths and weaknesses of different approaches and different applications of the technology[17].

The testing explored the characteristics of two types of CBDC, each serving a different purpose[18]:

[16] In practice, the money never moves, but is locked on one ledger in order to be represented and used for payments on another ledger. Therefore, descriptions such as 'moving' and 'bridges' are only metaphors for what actually happens in a DLT context.
[17] The testing was limited to a selection of ledger technologies, including Hyperledger Besu and Hyperledger Fabric, and is intended to highlight architectural and design choices rather than provide a comprehensive technology comparison.
[18] For a more detailed description of the purpose of introducing CBDC, see Norges Bank (2026)

Retail CBDC are central bank digital currency available to the public for use in payments at points of sale, between individuals, in online commerce, and for receipts and disbursements to and from the public sector. The introduction of retail CBDC will entail a significant restructuring of the payment system and, if widely adopted, may affect the roles and business models of banks and other private actors in the payment system.

Wholesale CBDC is comparable to traditional central bank reserves, but on a new technological platform based on ledger technology. Wholesale CBDC is only available to banks and other financial actors with accounts at the central bank, and is used for settlement between banks, including in connection with trading in tokenised securities and TBD.

The majority of the testing has focused on wholesale CBDC between banks. With the exception of test case 1, which focuses on retail CBDC, the tests deal with the use of CBDC as a means of settlement in interbank settlements and in interaction with tokenised securities and TBD.

A total of six test cases were carried out during the project period:

- Test case 1: Establishment of a ledger for retail CBDC and exploration of the characteristics of such a ledger.
- Test case 2: Establishment of a ledger for wholesale CBDC and exploration of the characteristics of such a ledger.
- Test case 3: Establishment of a bridge for transfers of tokens/values between ledgers and exploration of the characteristics of such a bridge.
- Test case 4: Characteristics of tokenised bank deposits in a multi-ledger architecture.
- Test case 5: Trading and settlement of tokenised securities on a single shared ledger.
- Test case 6: Issuance and life cycle of government bonds on a single shared ledger.

Some of the tests were conducted in collaboration with DNB as a participating bank in the sandboxes. DNB Carnegie has also contributed to the design of certain test scenarios involving securities trading. This has made it possible to test technical solutions against relevant needs and transactions among actors in the financial sector.

Each test case is described in separate sections with the following structure: first, the purpose of the test case is described, followed by technical design. Finally, the results,

insights, and findings of each test case are summarised. At the very end of the report, lessons learned from all test cases are summarised.

Test case 1 — Establishment of a ledger for retail CBDC and exploration of the characteristics of such a ledger

The purpose of the test case

The purpose of the test case was to explore how retail CBDC can be designed in a two-tier distribution model, where the central bank has technically enforceable control[19] over frameworks, rules, and the issuance and redemption of central bank money, while banks and regulated third parties have responsibility for customer relationships and customer data. The test was based on the premise that the central bank must ensure that no other actors can issue and destroy CBDC, while at the same time the central bank has no access to information about individual customers' payments with or holdings of CBDC.

A key objective was to explore how roles, rights, and governance capabilities can be technically enforced in a DLT-based solution. Ledger technology does not include embedded access control, and the test case was therefore designed to illustrate how the necessary mechanisms for access control and division of responsibilities can be defined explicitly, both on the ledger and in adjacent systems. This included exploring how the solution can support the principle that banks have direct customer contact, in line with the current division of responsibilities in the payment system.

Furthermore, the aim was to explore how the central bank can obtain real-time, aggregated overviews of circulation, and holdings in the system without access to detailed customer data. The test case thus provided insight into how access control at the ledger level can be used to balance the central bank's need for governance and control of issuance and redemption of central bank money with consideration for privacy, division of responsibilities, and the role of banks in a retail CBDC solution.

[19] In this report, the term 'the central bank's control' is used to describe the central bank's technically enforceable rights and responsibilities in the tested architecture, particularly in relation to the issuance and destruction of CBDC, the setting of overall rules, and access to aggregated overview data.
The term does not imply full operational control over all components, ongoing operation of all ledgers, or access to detailed information about transactions or customer relationships at banks and other actors. Operational responsibility, data access, and technical operation may be distributed among several actors in line with the architecture and role framework.

Technical design

The sandbox solution in this test case is based on Hyperledger Besu, which is a permissioned ledger compatible with the Ethereum Virtual Machine (EVM)[20].

The solution follows a two-tier distribution model as illustrated in Figure 1. The central bank constitutes the top layer and is the only actor with the right to issue and redeem retail CBDC. Banks and other approved, regulated third parties constitute the distribution layer and act as intermediaries between the central bank and the end customers. End customers do not participate directly in the ledger's governance structure, but use services provided by banks and other third-party actors.

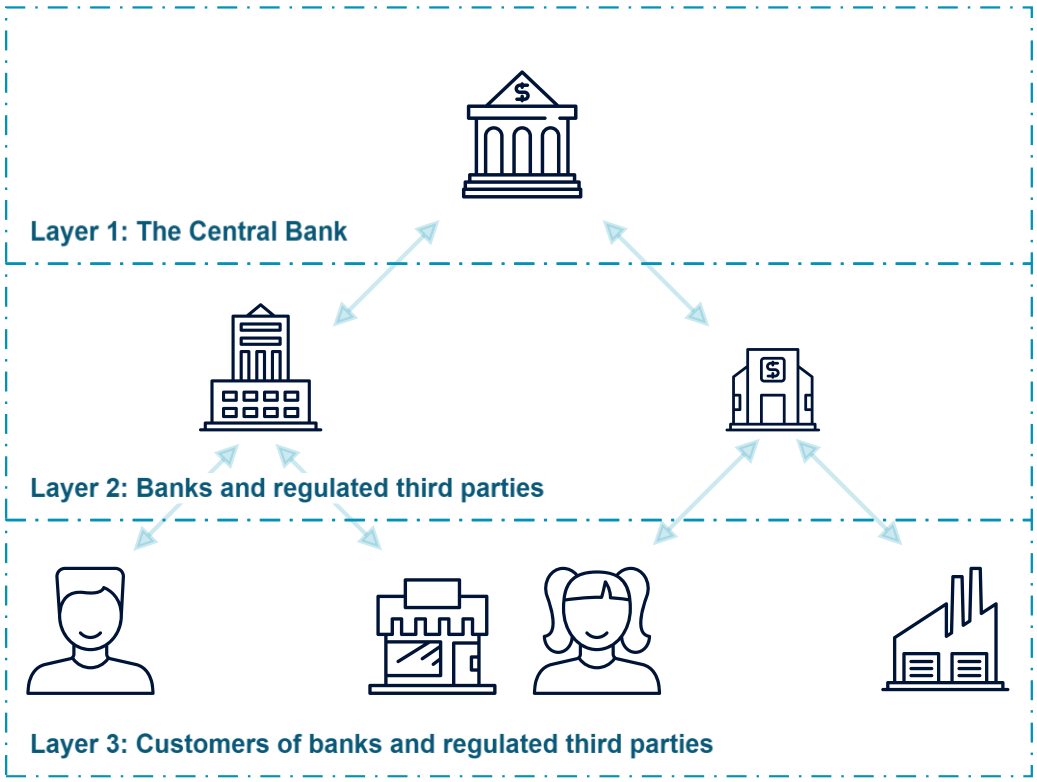

Figure 1: Two-tier distribution model

The distribution of responsibilities and access is technically enforced through role-based smart contracts on the ledger. Banks are responsible for customer onboarding and for linking customers to ledger addresses that are registered in the banks' own account registers. Only addresses approved by banks can hold and use retail CBDC. End customers can manage their own holdings and make payments, but cannot create addresses or

[20] Ethereum Virtual Machine (EVM) is a standardised execution model for smart contracts used by Ethereum and a number of compatible ledger platforms, including Hyperledger Besu. EVM defines how contract code is executed deterministically across nodes, how accounts and contracts store state, and how transactions are validated and recorded in a shared ledger.

change the rights themselves. The central bank does not have access to customer data and does not have responsibility for AML/KYC/CTF, but through its own roles it has access to aggregated overviews of circulation, holdings, and transaction activity on the ledger.

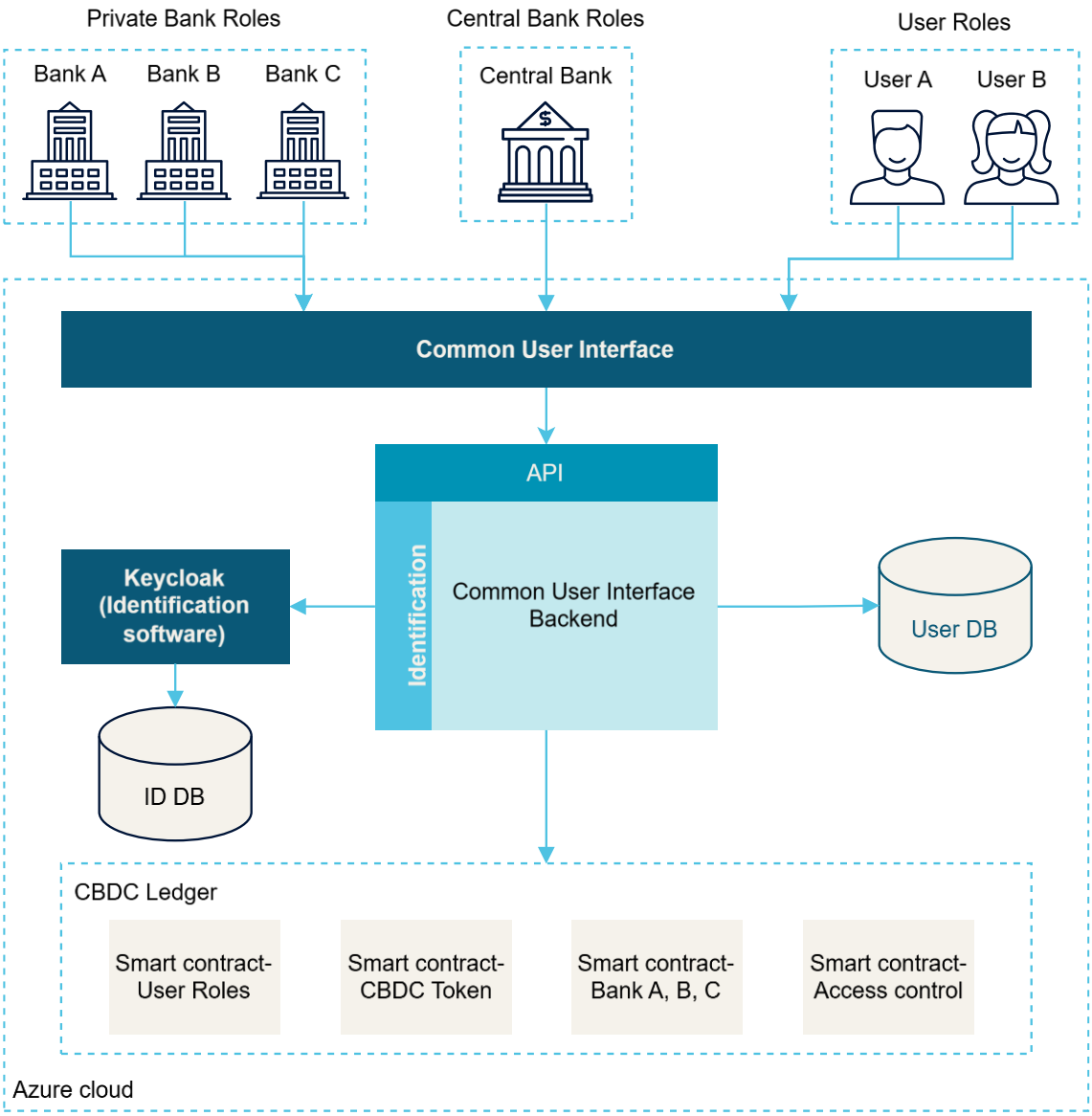

Figure 2: High-level architecture sketch of the retail CBDC sandbox

Figure 2 shows the solution design at a high level. A common application layer with user interfaces and API[21]s handles authentication, authorisation, and interaction with the ledger.

Various login mechanisms have been tested, including traditional login via external ID solutions and the use of digital wallets. The ledger itself contains only what is necessary to represent ownership and transfers of retail CBDC, while customer information and identity data remain at the banks and outside the ledger. The design has been deliberately chosen to test how technical mechanisms can ensure appropriate division of responsibilities and control over issuance and redemption of central bank money while limiting access to customer data.

Results and findings

Testing in the retail sandbox shows that role-based access control can be technically enforced in a DLT-based solution, allowing the distribution of responsibilities and rights between the central bank, banks, and end customers to be implemented directly on the ledger using smart contracts. Various login mechanisms were tested. Digital wallets provide a high degree of user control but entail a risk of key loss. Logging in via external ID solutions offers better user-friendliness and the possibility of recovery, but involves an additional intermediary.

The testing was not intended to provide a basis for a comprehensive assessment of privacy in a legal sense, but to highlight technical features that are relevant to privacy matters. The solution is designed so that customer information remains with the banks, while the ledger only contains what is necessary to carry out transfers of retail CBDC. The solution is based on pseudonymity[22], which reduces the exposure of personal data during normal operation. However, this creates a structural privacy risk if ledger addresses can be linked to identity, as the full transaction history will then be visible. The choice of ledger model and address structure therefore has significant implications for how privacy requirements can be met. Access control at the application level is not sufficient to handle traceability at the ledger level.

[21] An Application Programming Interface (API) is a set of rules and protocols that enables different software systems to communicate with each other and exchange data.

[22] Pseudonymity is a feature of a system where actions and data are linked to an identifier (a pseudonym) rather than directly to a real identity. The pseudonym acts as a substitute for the person or actor, making it possible to track activity over time without necessarily knowing who is behind it. However, if the link between the pseudonym and the identity is established, past and future actions can be linked to the same identity.

The test also showed that updating and further developing smart contracts is demanding, as contracts are immutable after publication and changes require new contracts and managing dependencies.

Test case 2 — Establishment of a ledger for wholesale CBDC and exploration of the characteristics of such a ledger

The purpose of the test case

The purpose of the test case was to establish a technical sandbox for CBDC used for settlement between banks (wholesale CBDC) and to explore whether Hyperledger Fabric[23] can support such a solution in a secure and controllable manner. The test case was intended to provide practical experience with the basic functions of an interbank settlement system, where CBDC is used as a means of settlement between banks on a shared ledger.

A key objective was to explore how the rights, access to information, and confidentiality can be technically enforced in such a system. This requires an unambiguous definition of what actions different actors are entitled to perform on the ledger, and that the rights can be technically enforced. Simultaneously, the solution must ensure confidentiality between banks, so that banks can settle payments with each other without individual banks having access to confidential information about other banks' payments and so forth.

Furthermore, the aim was to explore how the central bank can have access to sufficient information to ensure that the settlement system functions as intended. This includes, among other things, an overview of the total amount and use of CBDC in circulation, without the central bank having access to detailed data on transactions between banks.

Technical design

In test case 2, Hyperledger Fabric is used as a DLT platform to explore the properties of wholesale CBDC in an interbank settlement environment. The solution is established as a joint settlement platform where the central bank and all participating banks are part of the same network, but with decentralised operation and differentiated access to information and information sharing.

Decentralised operation means that each bank operates and hosts its own node on the ledger and is thus responsible for its own technical infrastructure. Figure 3 visualises how the 'bank peer node' is handled by each bank and connects to the ledger.

[23] Hyperledger Fabric is an open source platform for shared ledgers (DLT) developed under the Linux Foundation, designed for businesses that need controlled access and the ability to segregate data between participants.

In the tested solution, governance of access to information and information sharing are implemented using *private data collections* (PDC). PDC enables settlements to be carried out on a single shared ledger, while selected parts of the transaction data are only shared between authorised parties.

The test scenario has been further expanded to explore a UTXO-based model for representing wholesale CBDC, in contrast to the retail sandbox which was designed as an account-based model.

Token models

The UTXO model can be understood as digital 'banknotes' or 'value bits' with a specific value. When a payment is made, one or more 'value bits' must be used to cover the total amount of the payment. For example, if NOK 100 is paid using a digital 'NOK 200 note', the original value bit is consumed and two new ones are created: one that is transferred to the recipient (NOK 100) and one that is returned to the payer as 'change' (NOK 100). The remaining value is an example of an *unspent transaction output* (UTXO).

In the UTXO model, ownership is not linked to an account with a balance, but to each individual value bit. Ownership is defined by who is authorised to use the value bit as input in a new transaction. Each value bit has its own history from issuance to destruction, which makes it technically easier to verify that the same value bit cannot be used twice and to track values through subsequent transactions. This provides high traceability at the token level and may be well suited for parallel processing of transactions.

The account-based model is similar to how banks currently manage bank deposits. Each participant has an account that shows a total balance at any given time. When a payment of NOK 100 is made, the payer's balance is reduced by NOK 100 and the recipient's balance increases accordingly. In this model, the focus is on the balance per account, not on the specific 'value bits' that make up the balance. Traceability is more closely linked to the transaction log and changes in account balances over time: it is possible to identify who has sent and received values and how balances change, but the value does not have its own 'banknote history'. The account-based model often provides simpler balance management and more intuitive access control for data visibility at the account level.

Both models can be used for digital settlements, but they represent different technical approaches with different strengths and weaknesses in terms of traceability, complexity, and access management.

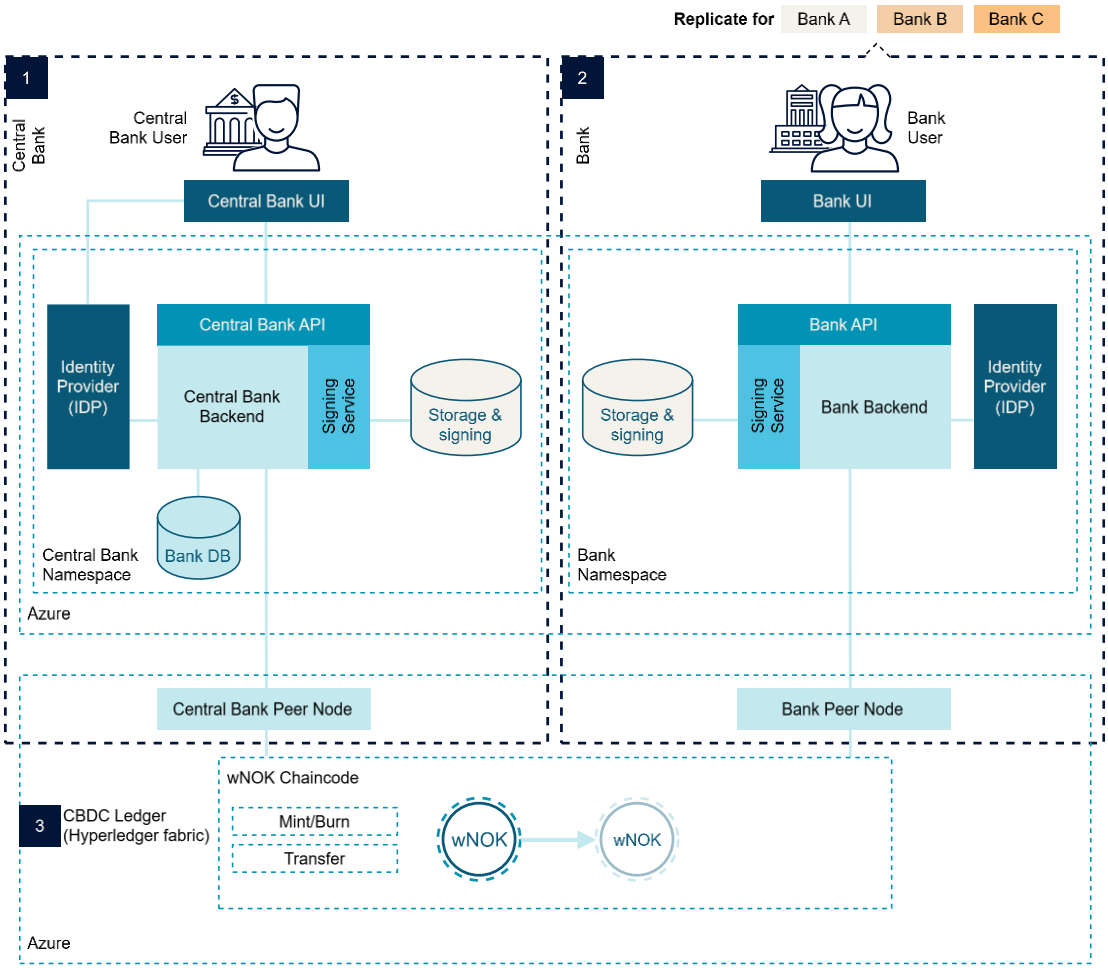

*Figure 3: Architectural sketch of the CBDC sandbox for settlement. **1)** Shows the components used by the central bank that are part of the central bank's infrastructure and operational responsibility. **2)** shows the components used by each participating bank and which are part of each bank's infrastructure and operational responsibility. **3)** shows the Hyperledger Fabric network, ie the ledger itself. Note that each peer node is located on the ledger. This is also where the wNOK (wholesale CBDC) token is located.*

The solution has two separate user interfaces, one for the central bank and one for banks, both of which are tailored to their respective roles. The central bank's view provides an overview of CBDC in circulation, the central bank's total holdings of CBDC, transaction activity over selected time periods, and total amounts per bank. The banks' user interface can be used for payment orders and transfers of CBDC to other banks or back to the central bank, as well as to obtain information about the bank's own balance and transactions.

The test case includes three participating banks (here referred to as Bank A, Bank B, and Bank C in Figure 3). This allowed us to test features such as access to the ledger, access to information, and shielding of information. With more participating banks, it becomes easier to assess how the solution behaves when more participants are included in the same infrastructure.

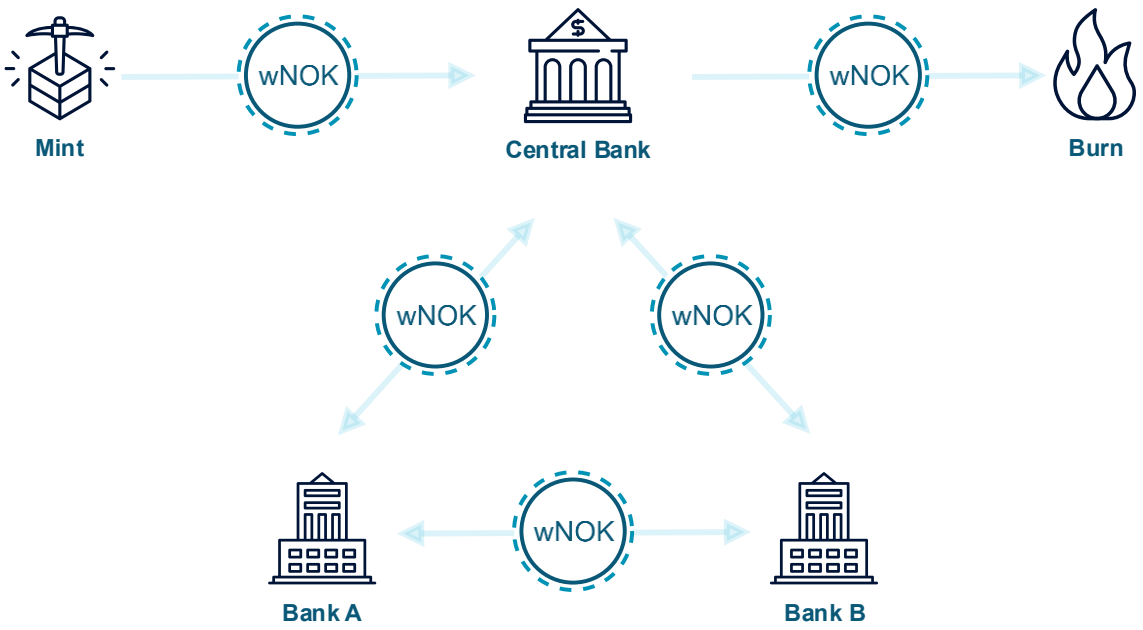

Figure 4: Overview of how wholesale CBDC is issued (mint), destroyed (burn), and moved between participants in the system. In the figure, wNOK = wholesale CBDC.

The central bank issues wholesale CBDC and the newly created tokens are registered at the central bank's address on the ledger. From there, wholesale CBDC is distributed to participating banks (illustrated here as Bank A and Bank B). The banks can then use wholesale CBDC freely to settle payments between themselves through transfers of tokens on the ledger. When a bank wishes to convert all or part of its holdings of wholesale CBDC into traditional reserves, the wholesale CBDC is sent back to the central bank, which destroys the tokens and hence takes them out of circulation. As an integrated part of the process, the holdings of traditional reserves will be increased accordingly. However, this part of the transaction has not been included in the test.

Results and findings

This test case demonstrates that Hyperledger Fabric may be a relevant technology choice for CBDC used for settlement between banks, particularly in situations where there are different needs for access to information and where the information must be shielded. The tests show that this flexibility entails costs and challenges, including more demanding setup, operation, and management than solutions based on a single fully shared ledger with equal access to information for all.

The test also shows that Hyperledger Fabric makes it technically possible to handle wholesale CBDC in both an account-based model and a UTXO-based model. Both models support the basic functions required in an interbank setting, including issuance, transfers between banks, and redemption. The UTXO-model also provides traceability at the token level, which may be relevant for verification, troubleshooting, and reconstruction of events.

The test shows that the use of PDC enables control over which transaction data different actors have technical read access to, even if the settlements are carried out on a common ledger. In the tested solution, the central bank can monitor the interbank settlement, including amounts, counterparties, and timing, as well as access aggregated overviews of circulation. Detailed bilateral transaction data and information about underlying customer relationships are shielded and therefore not available to the central bank. PDC is used to ensure that banks only have access to information about the settlements in which they themselves are a part of, and not about other banks' transactions.

The tests indicate that Hyperledger Fabric can be used as the technology for a CBDC solution for settlement with technical feasibility for multiple token models, precise access control for data visibility, and access to aggregated overview data. The primary problem identified by the test is that the structure of this solution becomes relatively complex. As more banks connect and more information boundaries need to be enforced, the requirements for standardisation, governance, and operational robustness increase compared to solutions based on a fully shared single-ledger.

Test case 3 — Establishment of a bridge for transfers of tokens/values between ledgers and exploration of the characteristics of such a bridge

The purpose of the test case

The purpose of the test case was to explore the consequences of using wholesale CBDC as a means of settlement in an infrastructure where CBDC and other values are registered on different ledgers.

The test case was based on the central bank needing to control the use of CBDC on a separate, closed ledger, while other actors wanted to develop payment solutions using smart contracts on their own external ledgers. Through the test, we sought to clarify how wholesale CBDC could be used as a means of settlement across systems, how governance and role-based responsibility could be ensured, and what characteristics such a structure would have.

In the test case, wholesale CBDC was registered on a central bank-controlled ledger, while securities and application logic were located on an external ledger outside of the central bank's control. A bridge was used as a technical mechanism to enable interaction between the two ledgers in such a multi-ledger structure.

Technical design

The bridge in test case 3 is built as a centralised link between two separate blockchain environments: a core ledger for wholesale CBDC based on Hyperledger Fabric, and an external ledger based on Hyperledger Besu. The purpose is to enable the use of CBDC in applications on the external ledger without actually moving CBDC out of the core ledger and without relinquishing control over central bank money in circulation.

Figure 5 shows the main components and the money flow between them. On the left is the core ledger with the wholesale CBDC contract. On the right-hand side is the external Besu ledger, which contains, among other things, a wrapped[24] wholesale CBDC contract (wNOK*) and a simple zero-coupon bond contract. Between those ledgers is the bridge component, which has system access to both ledgers.

Wholesale CBDC can only be issued and destroyed by the central bank. In the bridge setup, a custody account on the core ledger is used for temporary locking of CBDC when the funds are made available on an external ledger. This ensures that the same CBDC value cannot be used in parallel on multiple platforms/systems.

On the external ledger, wholesale CBDC is represented by a smart contract where only the bridge has the rights to issue and destroy tokens. The bridge is implemented as an external component connected to both ledgers and is responsible for coordinating transfers of CBDC.

The specific transfer flow is illustrated in the figure below.

[24] A wrapped token is a representation of a value that is locked on another ledger and can be used as a means of payment or a means of settlement on the target ledger.

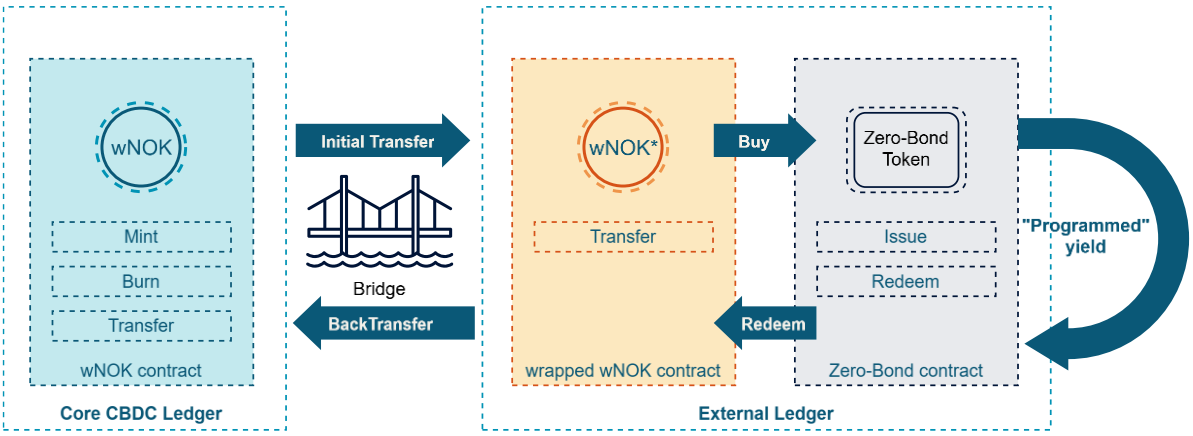

Figure 5: Overview of how money is moved between ledgers. On the core ledger, wholesale CBDC is listed as 'wNOK'. On the external ledger, the wrapped wholesale CBDC is listed as 'wNOK'*

When a bank wishes to use CBDC in an application on the external ledger, a bridge operation is initiated. In a typical scenario, a bank will log into a web interface connected to the core ledger, select the amount and destination address on the external ledger, and send a request to the bridge to 'move' the CBDC to the specified address. The bridge then starts a process that broadly follows these steps:

1. The bridge first creates an *asset reference* on the core ledger and locks the relevant wholesale CBDC amount in its custody account. After this operation, the amount remains on the core ledger but is no longer available for normal transactions.
2. Once the locking is confirmed on the core ledger, the bridge sends a transaction to the *wrapped wholesale CBDC* contract on the external ledger, which issues a corresponding *wrapped wholesale CBDC* amount to the recipient's address on the external ledger.
3. The recipient can now use *wrapped wholesale CBDC* in applications on the external ledger, for example to purchase zero-coupon bond tokens. The zero-coupon contract receives *wrapped wholesale CBDC* from the buyer, issues bond tokens, and upon maturity, the contract repays the face value of the bonds to the bondholder's address in *wrapped wholesale CBDC*.

When the actor on the external ledger wants to transfer the funds back to the core ledger, a reverse transaction flow is executed. The user (or an application on its behalf) sends *wrapped wholesale CBDC* back to the bridge's address on Besu (a so-called 'bridge back' operation). The bridge validates that the amount received is correct and then performs two steps:

1. The bridge calls the *wrapped CBDC* contract and destroys the received *wrapped CBDC* amount. After this, *the wrapped wholesale CBDC* no longer exists on the external ledger.

2. The bridge sends a transaction to the core ledger that unlocks the corresponding wholesale CBDC in the custody account and transfers it back to the recipient's account on the core ledger.

The process is built as a coordinated and sequential flow, where the locking of CBDC on the core ledger must be confirmed before wrapped wholesale CBDC can be issued on the external ledger. If locking on the core ledger is unsuccessful, the process is interrupted and no wrapped CBDC is issued. Upon reversal, the wrapped CBDC is destroyed on the external ledger before the funds are unlocked on the core ledger. If destruction is successful but the unlock operation on the core ledger fails, the value remains locked and unavailable until the bridge either succeeds on a retry or reconstructs a consistent state based on previous confirmations. Wrapped CBDC cannot remain in circulation without a confirmed lock on the core ledger.

As long as the wholesale CBDC amount remains locked on the core ledger, the total amount of CBDC remains under central bank control, even if the funds are temporarily made available for use on an external ledger in the form of *wrapped wholesale CBDC*. The locking and issuance mechanism ensures that there is always an unambiguous allocation of the amount: either as available CBDC on the core ledger or as locked CBDC on the core ledger with a corresponding *wrapped CBDC* on the external ledger. There is therefore no double-spending or deviation in the total amount of CBDC.

Since the bridge is a central component with rights to lock funds on the core ledger and issue/burn tokens on the external ledger, the technical design is based on a clear division of responsibilities and limited rights. On the core ledger, the bridge only has the rights needed to move funds in and out of its own custody account, not to be an issuer of new CBDC. On the external ledger, the bridge's account is set as the owner/administrator of the wrapped CBDC contract, with exclusive access to the issuance and destruction functions. It is not possible, except for the bridge, to change the total amount of *wrapped wholesale CBDC* .

The bridge's function means that the central bank does not need to operate or govern the infrastructure on the external ledger. It is sufficient for the bridge to have an account on the external ledger and access to a node endpoint to send and read transactions. This provides flexibility to use CBDC for payments on external ledgers without taking operational responsibility for the entire network of ledgers. This architecture also requires that the external ledger has a governance and control regime that prevents the contract representing *wrapped wholesale CBDC* from being changed, replaced, or circumvented in a way that weakens the bridge's function. Consequently, the technical and operational

control of the external infrastructure must be assessed before CBDC can be used as a means of settlement on such a platform.

The technical design also entails certain limitations. In this test, the bridge can only transfer wholesale CBDC between the ledgers, not other types of tokens or data. Nor does it allow smart contracts to be called directly across the ledgers.

The zero-coupon bond contract is an example of a pure application that occurs on the external ledger and only uses *wrapped wholesale CBDC* as a means of payment; the logic of this contract does not directly affect the core ledger. This is an important distinction between a bridge used solely for liquidity transfer and more advanced interoperability solutions where contracts on one ledger can trigger contracts on another.

The design used in test case 3 is based on the bridge being understandable and controllable from a business and regulatory perspective. Each step in the flow between the ledgers can be explained as simple operations: *lock, mint (issue), transfer, burn(destroy), and unlock*. Figure 5 visualises how these steps are connected, making it possible to trace the flow of tokens throughout the entire process, both for technical and non-technical readers. This provides a framework in which the use of CBDC in external applications can be gradually expanded without changing the basic principles of control of the issuance and redemption of central bank money.

Results and findings

The tests show that it is possible to maintain unambiguous control over one and the same CBDC value even if the funds are made available on an external ledger via a bridge. The solution ensures that the same money is never available in two places at the same time, by locking the CBDC value on the core ledger while issuing a representation of the value on the external ledger, as long as the bridge is functioning correctly.

The settlement process in the bridge-based setup worked as intended within the test conditions, but did not have the same degree of simultaneity and finality as settlement within a single shared ledger. When conducting a settlement via a bridge, the process consists of several coordinated steps across ledgers, which can result in temporary intermediate states in the event of delays or technical interruptions. In the test, this was handled in a controlled technical environment. Nevertheless, in a real operating environment, such settlement processes would require continuous operational monitoring and active error handling.

The bridge solution requires one actor to have privileged access to both systems in order to unlock, issue, and destroy CBDC. In this test, this role was assigned to the central bank. Although the technical operation of the bridge can in theory be carried out by a third party, the solution requires that the central bank has control over and responsibility for the functions that affect the total amount of central bank money in circulation at all times. The issuance and destruction of CBDC must therefore be technically and organisationally subject to the central bank, regardless of who operates the underlying infrastructure. Such centralisation provides clear and technically enforceable control over central bank money in circulation. However, it also entails increased responsibility and stricter requirements for the operation, security, and monitoring of the entity managing the bridge.

In architectures with multiple independent ledgers, interoperability will require some form of bridge or equivalent interoperability protocol between the ledgers. When CBDC is mainly used as a means of settlement for processes carried out on external ledgers, few of the key benefits of DLT are realised, such as close integration between money and business logic, atomic interaction, and reduced need for external orchestration. The tests indicate that the full benefits of a blockchain-based CBDC solution require external actors to be able to use CBDC directly in their own solutions, so that money, logic, and processes can interact on the same technological platform.

Test case 4 — Characteristics of tokenised bank deposits in a multi-ledger architecture

The purpose of the test case

The purpose of this test case was to explore the characteristics, advantages, and challenges of a blockchain-based infrastructure where TBD[25] are used as a means of payment, and where wholesale CBDC was used as the final means of settlement between the banks involved. The test was intended to demonstrate how banks could issue and manage TBD on their own ledger technology, to explore whether TBD could be converted securely and seamlessly to retail CBDC issued by the central bank, and to highlight the advantages and disadvantages of a structure in which the actors operate on multiple ledgers. The test case builds on the experiences from test case 3, but with a slightly different focus: here, TBD are the means of payment, while wholesale CBDC is used exclusively for the interbank settlement.

[25] TBD (tokenised bank deposits): A digital bank deposit issued by a commercial bank as a token on a DLT/ledger, where the token represents a claim on the bank with the same value as the deposit (typically 1:1)

A key motivation was to explore characteristics in a scenario where the payment system is not built on a single shared ledger for all banks and participants, but where each bank has greater freedom to choose its own technology platform and pace of innovation. This is consistent both with recent international descriptions of 'unified ledgers'[26] and 'regulated liability networks'[27], and with a market in which national and international participants must interact across system and jurisdictional boundaries. The test also had an innovation objective: to explore whether bank deposits in tokenised form could enable new processes or use cases, including whether banks could offer customers fast real-time conversion between TBD and retail CBDC across technologies in a secure and controlled manner, and how interbank settlements could be handled when both the central bank's CBDC and the banks' TBD systems were part of the same testing environment.

Technical design

The test setup includes mechanisms for issuance and management of TBD at individual banks, conversion between TBD and retail CBDC, and settlement between banks in wholesale CBDC. The solution builds on the architecture from test case 3, where the core ledger for wholesale CBDC and the retail CBDC ledger are connected via a bridge controlled by the central bank. In test case 4, separate ledgers have also been established for TBD at each commercial bank. The overall architecture is referred to as a 'Unified Ledger Sandbox' and is illustrated in Figure 6.

[26] BIS 'Unified Ledger': Concept for a common, programmable platform where tokenised money (central bank money and bank deposits) and tokenised assets can interact, allowing transactions to be automated and settled more seamlessly — preferably with final settlement in central bank money.

[27] Regulated Liability Network (RLN): Concept for a regulated (often permissioned) financial infrastructure with a shared ledger that records, transfers, and settles tokenised liabilities from regulated issuers — typically central bank money, TBD, and e-money — with the aim of settlement at par and better interaction between institutions.

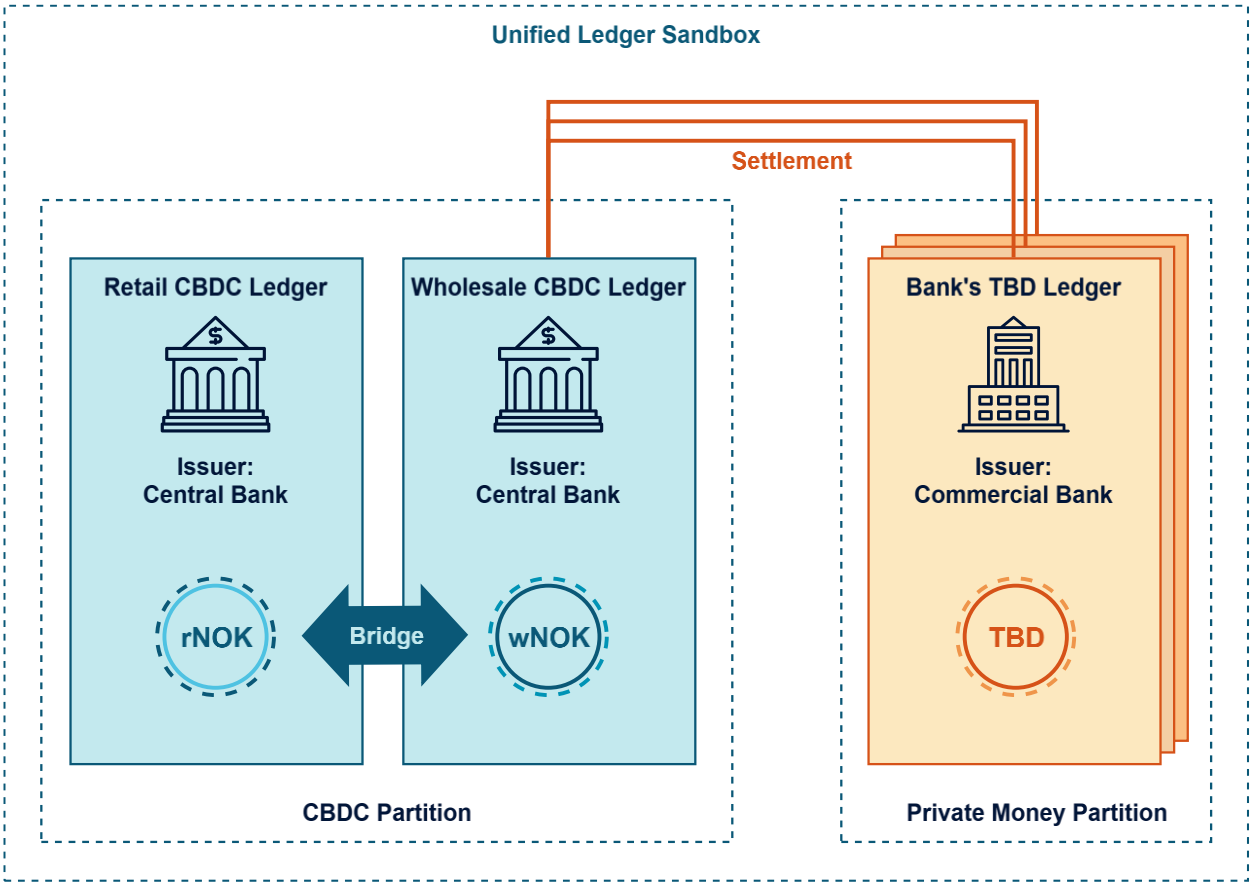

Figure 6: High-level overview of the implemented solution. Retail CBDC and wholesale CBDC on two different ledgers are connected via a bridge controlled by the central bank. TBD are found on ledgers at commercial banks. Transactions with TBD are settled in wholesale CBDC.

Figure 6 shows the core elements of the solution: The wholesale CBDC and retail CBDC are located on separate ledgers, connected by a bridge that can lock the wholesale CBDC and issue the retail CBDC, and vice versa. In addition, each bank has its own TBD ledger where the bank's customers' deposits are represented as TBD tokens. Transactions in TBD between banks are settled in CBDC on the wholesale CBDC ledger. The TBD ledgers are implemented as separate channels in the same distributed network used for wholesale CBDC, but each channel is logically separate and controlled by the individual bank; these channels could therefore be considered separate ledgers, see textbox below.

Channels in Hyperledger Fabric

In Hyperledger Fabric, a channel is a logically delimited subnetwork that functions as a separate ledger, with its own data, smart contracts, and access rules. Only actors who are members of a channel have technical access to information about transactions and holdings on the channel.

An organisation can be a member of several channels at the same time. This allows several logically separate ledgers to run on a single shared technology platform and to segregate sensitive information between actors.

The actor who creates and owns a channel is also responsible for the operational management of the channel, including access control, membership, smart contracts, and the operation of the associated infrastructure.

In this solution, each bank's TBD ledger is implemented as a separate channel, controlled and operated by the individual bank. The channels are therefore treated architecturally as separate ledgers, even though they are technically part of the same Fabric network.

each customer has an identifiable account with a balance in TBD. Only the bank has the right to issue and destroy TBD on its own ledger, while transfers between customer accounts are made according to rules defined in the smart contract. The central bank is a member at channel level in the sandbox in order to be able to read aggregated data (e.g., total outstanding TBD), but has no control rights over the TBD contracts.

The goal of test case 4 has been to use this setup to explore the characteristics of DLT-based versions of two financial processes:

- *Conversion from TBD to retail CBDC* (comparable to a cash withdrawal from a bank account), and
- *The interbank settlement for TBD* (the settlement in *wholesale CBDC* between banks, as a result of transfers between customers in different banks).

Both processes have been implemented so that, technically speaking, they can function independently of which ledger the individual bank has chosen for its TBD ledger. In the sandbox, TBD is built on Hyperledger Fabric channels, but the logic for settlement and conversion is placed on the wholesale CBDC ledger and in a separate orchestration component, so that, in principle, other types of TBD ledgers could be used as long as they can be connected to these components.

Banks therefore have complete freedom to choose the ledger for their TBD. In doing so, we have deliberately chosen one of the most complex approaches to issuing TBD.

Orchestration of processes in a multi-ledger setup

In the tested multi-ledger setup, composite processes such as TBD conversion and interbank settlement cannot be executed as a single atomic transaction on a single ledger. The connection between necessary operations is therefore enforced through explicit orchestration in the application layer, outside the ledgers. The orchestration is implemented as state-based process logic (*finite state machines*) and controls the order of operations, locking of funds to prevent double spending, bridge calls where applicable, and handling of errors and interruptions.

It is important to distinguish the orchestration from the bridge itself.

A **bridge** is a dedicated technical component that enables the transfer or representation of values between ledgers, typically by locking a value on one ledger and issuing a corresponding representation on another. The bridge has no business logic and no comprehensive understanding of the process it is part of.

Orchestration, on the other hand, is responsible for the overall process and determines *when* and *whether* irreversible steps should be performed. Settlement in a multi-ledger setup is therefore not achieved through the bridge alone, but through a controlled orchestrated process in which irreversible operations are only performed when the necessary prior confirmations are in place. If errors occur before a defined point in the process, the entire process can be interrupted and reversed.

In principle, bridges can be used as part of an orchestration logic. *In the tested solution, however, the bridge is only used as a value transfer mechanism*, while the actual orchestration — including sequence governance, locking, and error handling — is explicitly located in the application layer outside the bridge.

Conversion from TBD to retail CBDC

Figure 7 illustrates the conversion from TBD to retail CBDC. The process can be seen as the DLT-based parallel to a customer making a cash withdrawal from their bank account.

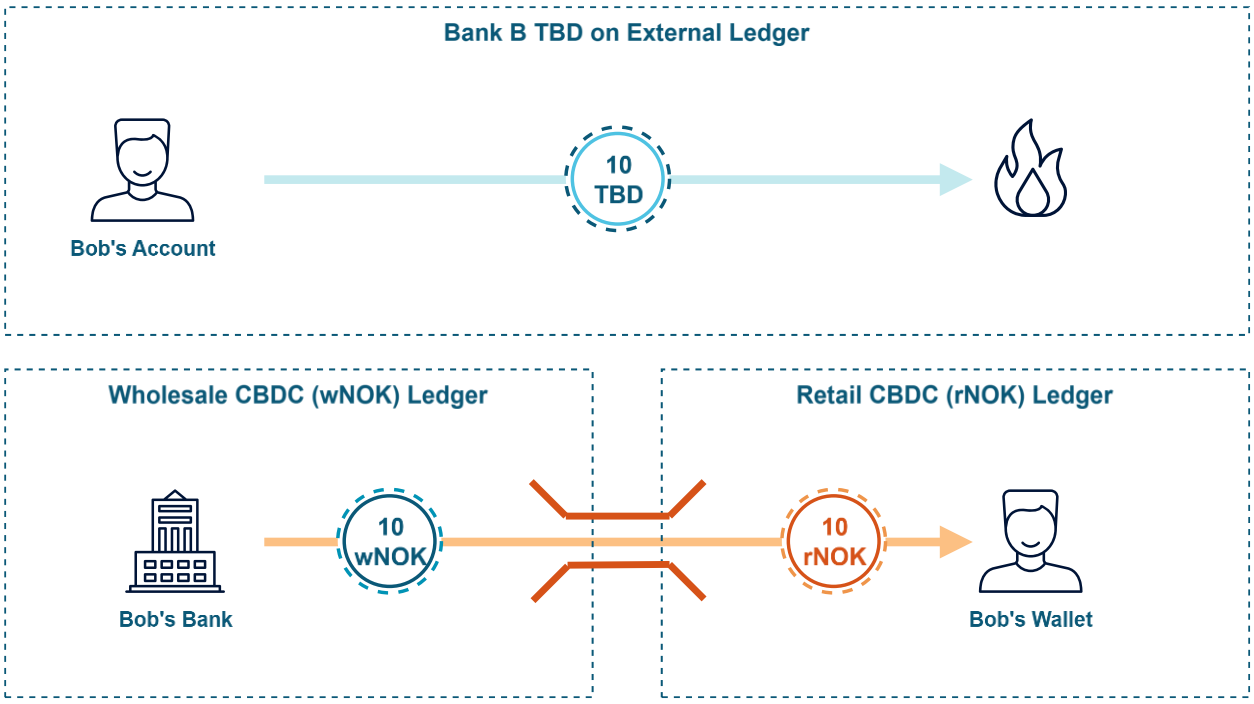

Figure 7: Conversion from TBD to retail CBDC

Technically, the following occurs:

1. The customer initiates a withdrawal by asking the bank to convert a given amount of TBD to retail CBDC.
2. The bank checks that the customer has a sufficient TBD balance and then locks the relevant TBD amount on the bank's TBD ledger. This is done using the chaincode on the TBD channel, which changes the customer's balance so that the amount cannot be used in other transactions while the withdrawal process is ongoing.
3. The bank then initiates a settlement transaction on the wholesale CBDC ledger. The bank reduces its reserves in the wholesale CBDC by the same amount that the customer is to receive in the retail CBDC. This reduction is made through a transaction from the bank's account to a bridge-controlled account, which is part of the bridge mechanism of the retail CBDC ledger.
4. The bridge between the wholesale CBDC and retail CBDC ledger locks the wholesale CBDC (wNOK) and issues CBDC on the retail CBDC ledger. The retail CBDC is first credited to the bank's account on the retail CBDC ledger.
5. The bank then transfers the relevant amount in retail CBDC to the customer's address on the retail CBDC ledger. After this, the customer has a direct claim on the central bank in the form of retail CBDC, and the corresponding TBD amount is permanently reduced on the bank's TBD ledger.

At the balance sheet level, this means, as stated in Figure 7, that the bank's deposits (TBD) are reduced and the bank's reserves (wholesale CBDC) are reduced accordingly. On the central bank's balance sheet, the bank's reserves are reduced, while the holdings of retail CBDC increase. This solution ensures that there is always a one-to-one relationship between the reduction in TBD and the increase in retail CBDC, and that no new money is created in the process.

Interbank settlement for TBD

Figure 8 shows how the interbank settlement for TBD is modelled in the sandbox. The scenario is that the bank customers transfer TBD between one another across banks, for example from a customer in Bank B to a customer in Bank A. Today, a corresponding settlement takes place by updating the banks' deposits (customer accounts) in their respective internal ledgers, while the banks' mutual positions are settled in central bank reserves. In test case 4, the same transfers are made as an integrated process.

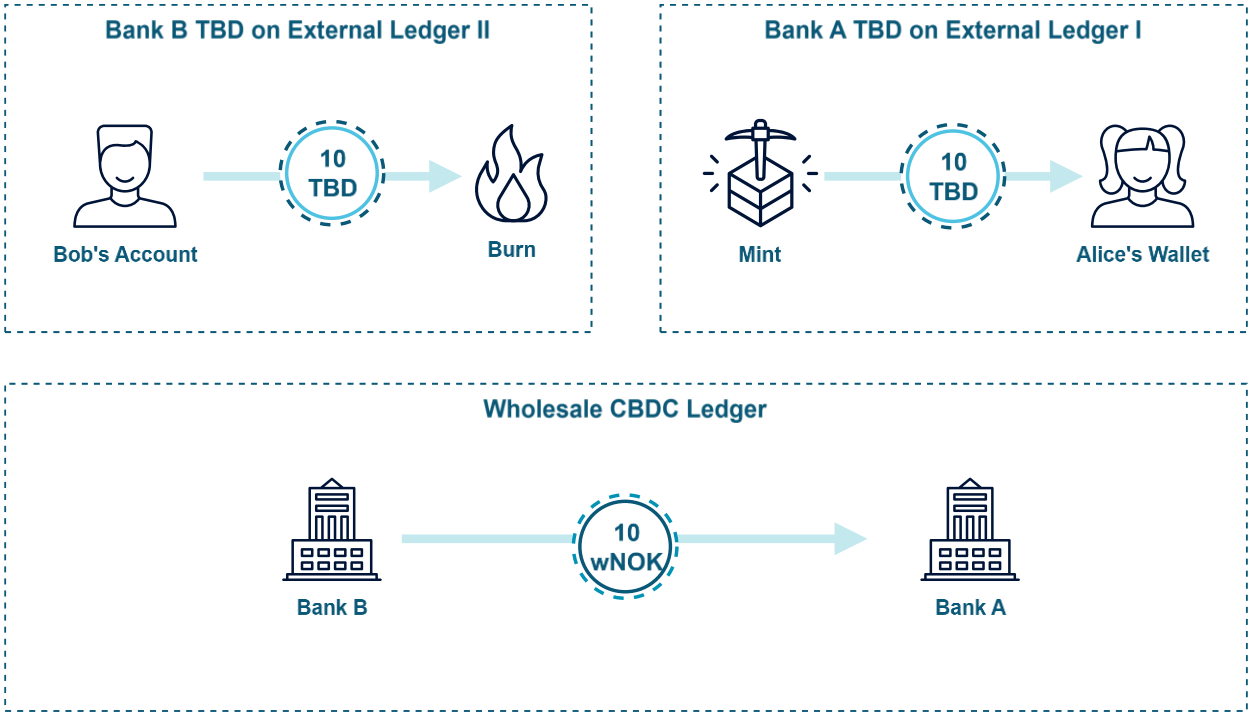

Figure 8: The interbank settlement for TBD

Technically, such a credit transfer consists of three main operations on three different ledgers:

1. **On Bank B's TBD ledger:** Bank B reduces the customer's TBD balance by the amount to be sent. In the sandbox implementation, this is done by destroying or locking TBD tokens from the customer's account, with a reference to a unique settlement ID.

2. **On the TBD ledger of Bank A:** Bank A increases the recipient's TBD balance by the corresponding amount. This can be implemented by issuing new TBD tokens to the recipient's account, linked to the same settlement ID.
3. **On the wholesale CBDC ledger**: A settlement is carried out in the wholesale CBDC between Bank A and Bank B. Bank B's reserves in wholesale CBDC are reduced, while Bank A's reserves are increased by the same amount in an integrated operation.

These three operations are linked through the settlement process, or in the form of an orchestrated protocol on the wholesale CBDC ledger. The protocol ensures that there is a single source of truth[28] for the settlement itself in the wholesale CBDC, by storing a settlement transaction with references to the two banks' TBD transactions (for example, via unique transaction IDs). If something goes wrong before the actual CBDC transaction on the wholesale CBDC ledger is completed, the process is not executed. If the CBDC transaction has been completed, the settlement is considered final, and banks must ensure that their TBD ledgers reflect this.

In Figure 8, this is also expressed in balance sheet form: Bank B's deposits (TBD) are reduced, with a corresponding reduction in its wholesale CBDC reserves. Bank A's deposits (TBD) are increased, with a corresponding increase in its wholesale CBDC reserves. On the central bank's balance sheet, reserves are transferred from Bank B to Bank A, while the total amount of wholesale CBDC and TBD in the system as a whole remains unchanged.

The solution is deliberately designed so that the logic for the interbank settlement lies on the wholesale CBDC ledger, not in each individual TBD ledger. This means that the settlement process itself can be standardised and controlled centrally, while banks can have a high degree of freedom to choose the technology and structure for their own TBD ledgers, as long as they can refer to a unique settlement ID and perform issuance/destruction or similar operations in accordance with the settlement result.

Results and findings

In a multi-ledger setup, the settlement takes place across several independent systems. This means that transactions must be coordinated outside the ledgers themselves, which can result in intermediate states and a lower degree of simultaneity than atomic settlement in a single shared ledger. Such solutions therefore require additional mechanisms for coordination and error handling. When involved actors use separate ledgers, the smart

[28] Single source of truth: In an IT context, this is known as *the single source of truth (SSOT)* and refers to a single authoritative source that determines the final truth of information. In this setup, the wholesale CBDC settlement on the wholesale CBDC ledger serves as the single source of truth.

contracts cannot directly control cross-ledger operations. Coordination must take place through external mechanisms, such as orchestration or bridges, which increase technical and operational complexity compared to a single-ledger architecture.

The test shows that when each actor has its own ledger, several of the key benefits of blockchain technology are reduced. This applies in particular to the possibilities for coherent automation, atomic settlement, and close integration between money and business logic, as such features largely require that assets and logic are handled on the same ledger. In multi-ledger setups, these connections must be established through external orchestration and messaging. This can be compared to how such processes are handled in today's payment systems, which raises the question of whether improvements to existing systems may be a better alternative than implementing a CBDC based on ledger technology.

Test case 5 — Settlement of tokenised securities on a single shared ledger

The purpose of the test case

The purpose of the test case was to explore whether trading and settlement of tokenised securities provided significant benefits when all relevant assets and actors operate on a single shared ledger, compared with the multi-ledger solutions tested. The test was based on experience from previous test cases, which showed that cross-ledger transactions reduce the ability to reap the full advantages of ledger technology and CBDC, particularly in terms of atomicity, automation, and effective risk reduction.

The test case is relevant because the simultaneous transfer of ownership of multiple assets on the same ledger both provides atomicity and reduces settlement and counterparty risk. Unlike bridge and multi-ledger solutions, where the settlements take place as coordinated sequential processes, a single shared ledger allows all changes to take place in a single transaction. This enables real-time settlement, high traceability, and a single source of truth for both money and securities.

The test also aimed to highlight the new opportunities for automation and efficiency that arise when financial processes, including order processing, settlement, role validation, and compliance are implemented directly as smart contracts on a single ledger. This includes the ability to mimic netting-like effects in a gross settlement system, reduce liquidity requirements and eliminate the need for bridges and external orchestration between separate systems.

At the same time, the test case was intended to highlight the new challenges posed by a single-ledger model. When all actors were required to use a single shared ledger, this could be perceived as more intrusive than solutions where each actor had their own ledger. Issues related to governance, change management, and division of responsibilities were therefore an important part of the basis for assessment.

Technical design

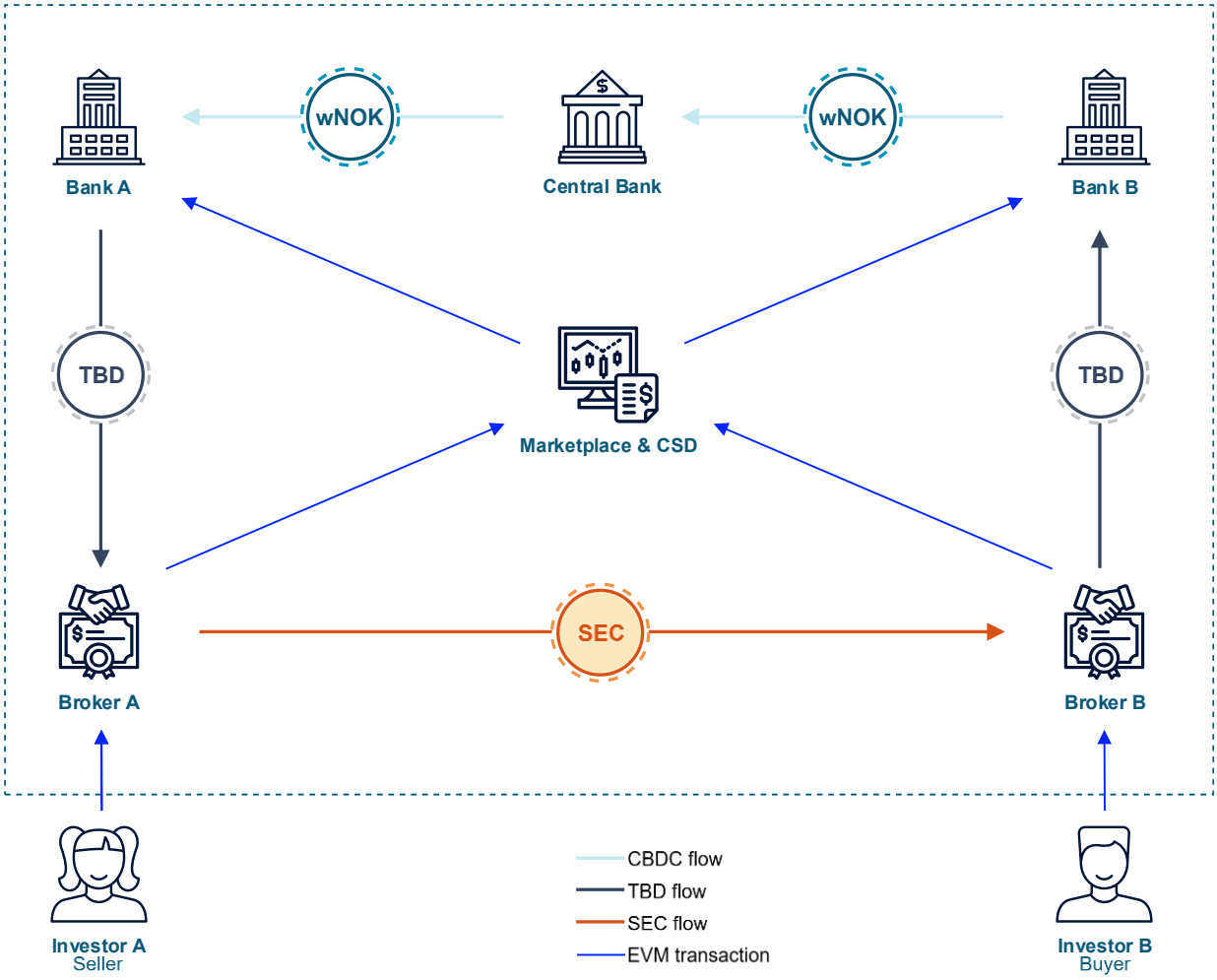

Figure 9: Overview of actors and transaction flow on the single ledger. Wholesale CBDC is listed as wNOK, TBD are listed as TBD, and securities are listed as SEC.

In this test, the central bank, banks, brokers, the marketplace, and the investors operate on a single shared digital ledger. The central bank issues wholesale CBDC, which is used as a means of settlement between banks, while each bank issues TBD to its own customers. When two investors trade a security through their brokers, orders are recorded in a shared order book on the ledger.

When a buy and sell order are matched in the shared order book, the settlement is initiated automatically. The buyer's bank debits the customer's TBD and simultaneously settles the trade in wholesale CBDC with the seller's bank, which issues new TBD to the seller. At the same time, the ownership of the security is transferred from the seller to the buyer. The customer does not have direct access to the wholesale CBDC; the settlement in central bank money takes place exclusively between the banks. The settlement is implemented as an atomic DvP process, where the cash and securities legs are either executed together with final effect, or not executed at all if one step in the process fails. The security therefore only changes ownership if the payment is completed at the same time.

About atomicity

Atomicity exists when all relevant state changes in a settlement, such as the transfer of money and securities, are executed as a single, technically indivisible, simultaneous operation, typically on a single ledger or within the same system. The transaction has only one final state: completed or not completed. There are no intermediate states where only parts of the settlement have been completed.

In coordinated settlements, such as in multi-ledger architectures or when using bridges, the settlement is executed as a sequence of dependent operations across systems. Such solutions may be functionally interconnected, but do not achieve the same technical simultaneity as in atomicity, and require explicit orchestration and error handling.

Access to the system is restricted to approved actors, and addresses and rights are validated against a common register. The architecture thus facilitates trading, payment, and record-keeping within a single technical platform, with consistent traceability and without the need for external reconciliation at the moment of settlement.

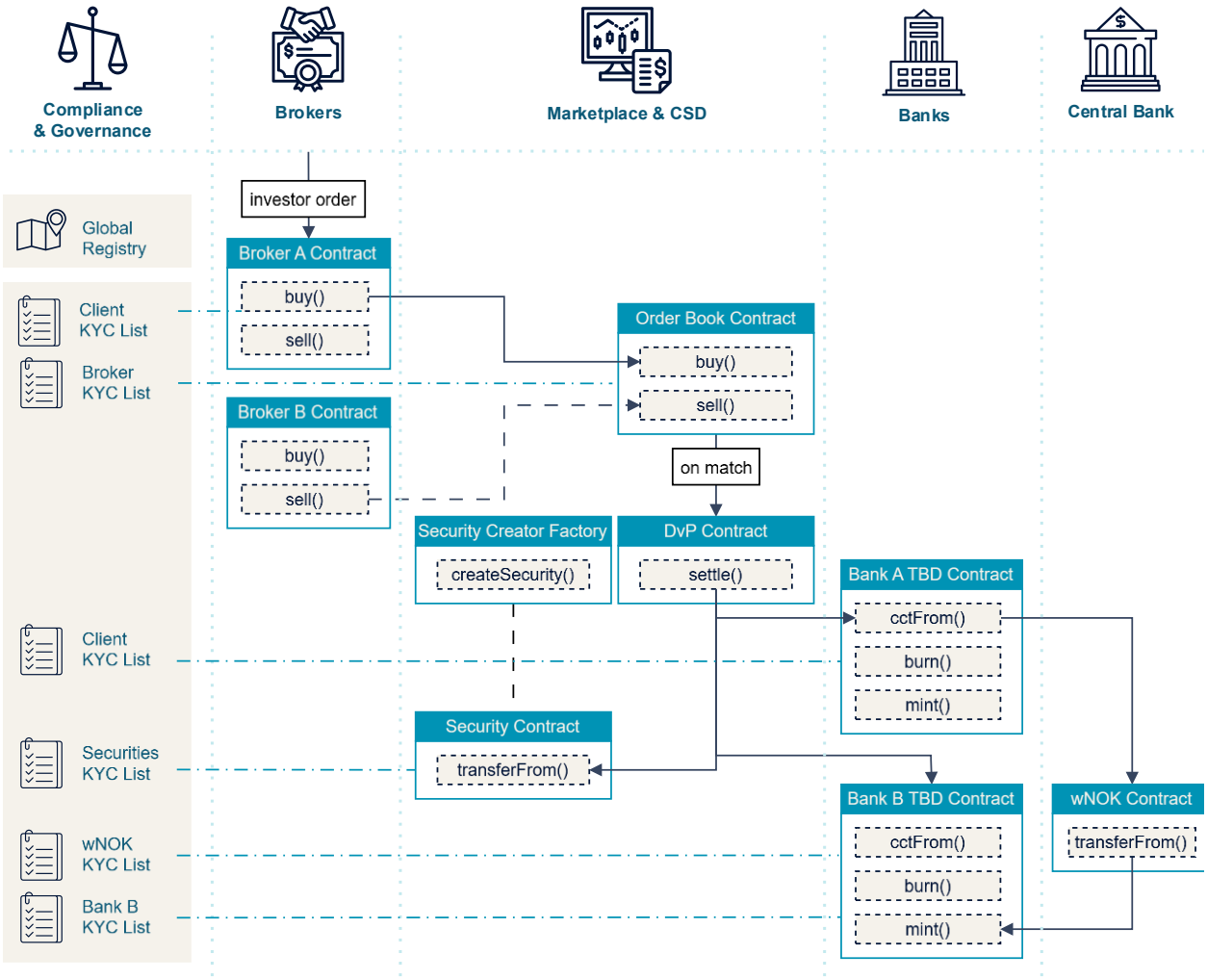

Figure 10: Overview of how the smart contracts that constitute the system for trading tokenised securities interact.

The test setup includes a *Global Registry* as a single source of truth for contract addresses and whitelists (approved banks, brokers, and customers). An investor places an order via the *Broker contract*, which first checks that the customer is approved and that the broker is authorised. The order is forwarded to the *Order Book contract*, which matches purchases and sales and, if there is a match, initiates the settlement by calling the *DvP contract*.

The *DvP contract* is used to enforce an atomic settlement in which both legs are executed simultaneously:

- **The security leg:** The security (issued by *the Security Creator Factory* and represented in *the Security contract*) is locked with the seller and prepared for a transfer.

- **The cash leg:** The buyer's bank settles the trade in wholesale CBDC with the seller's bank. The buyer's bank's outlay is covered by reducing the customer's TBD balance, so that the customer's claim on the bank is cancelled and the corresponding TBD is issued at the receiving bank. The customer never receives wholesale CBDC; the settlement in central bank money takes place exclusively between the banks.

The settlement in both legs is completed atomically in the same transaction; if a check fails (e.g. due to lack of authorisation, insufficient balance, an invalid counterparty, or incorrect version in *the Registry*), everything is stopped, so that there is no transfer of ownership of securities and no payment. If all checks pass, the *DvP contract* calls *transferFrom()* in the *Security* contract so that the ownership is registered to the buyer, and the settlement is completed.

The *'Factory'* pattern is used to issue new securities in a controlled manner (deterministic addressing via the Registry and built-in whitelists), mirroring what a production process might look like. Although full AML/KYC/CTF is outside the scope, the test enforces a simple variant: unknown customers or unauthorised brokers are stopped on-chain before orders are accepted.

Netting-like effects in a smart contract-based settlement model

In traditional securities and payment systems, netting is achieved by collecting multiple trades over a period of time in batch settlements and settling only net positions between the participants at fixed times. This reduces liquidity needs, but can also involve intermediate counterparty risk, operational complexity, and the need for clearing and guarantee mechanisms.

In the smart contract-based model implemented here, there is no such deferred settlement step. All trades are settled immediately and atomically through the *DvP contract*. At the same time, the model can provide ***netting-like effects***, particularly in the form of reduced liquidity needs, through the way in which multiple transactions and operations can be linked together and executed collectively in a single atomic process.

This is made possible by the fact that the DvP settlement can be surrounded by pre- and post-smart contract calls (pre- and post-hooks), which are executed as part of the same transaction. These surrounding calls are shown below in Figure 11. Such calls can, for example, be used to obtain the necessary liquidity, execute related trades, or clean up and consolidate positions after the settlement. Multiple trades, payments, and any liquidity steps can thus be chained together so that only the total net effect materialises on the participant's balance, without funds having to be provided gross in advance. The result is that economic effects traditionally associated with netting, particularly reduced liquidity

commitment, can be achieved without deferred settlement, batch processing, or counterparty risk. It is therefore more accurate to refer to this as netting-like effects rather than netting in the traditional sense. The settlement remains immediate and final, but the settlement process itself may internally consist of several coordinated steps that either succeed collectively or are not completed at all.

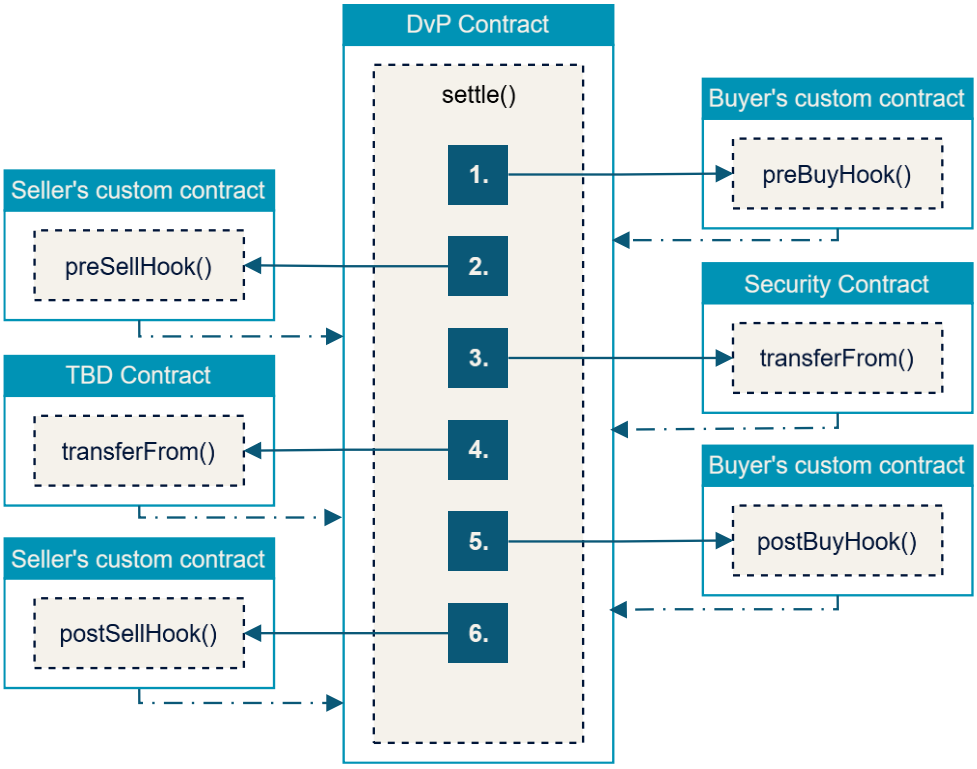

Figure 11: Enhanced DvP contract, where four external smart contract functions — preBuyHook(), preSellHook(), postBuyHook(), and postSellHook() — are called as part of the same transaction. The function calls surround the actual DvP process, including the transfer of securities and TBD (including CBDC)

Results and findings

The testing shows that when tokenized money and securities are handled on a single shared ledger, the settlement can be carried out atomically: either both the money and securities legs are executed simultaneously and with final effect, or the entire transaction is cancelled if one part fails. The tested single-ledger architecture thus meets key requirements for coherent and secure settlement, technical enforcement of roles and access, as well as traceability and compliance. Compared to multi-ledger solutions, the need for intermediaries such as bridges and messaging mechanisms is reduced, and the technical complexity is lower. However, experience from the test shows that several issues need to be clarified. Before such a solution can be implemented, further testing is required

in terms of robustness under real load conditions, cybersecurity, contingency planning and recovery in the event of failure, as well as interoperability with foreign and multinational systems. The requirement that all actors must use a single shared ledger also raises questions about flexibility, incentives, and willingness to adopt, especially for actors with specific regulatory or international needs.

The test also shows that the use of smart contracts on a shared ledger enables the enforcement of settlement rules, roles, and access directly in the infrastructure, without the need for external orchestration or manual control steps. Responsibility for the infrastructure points to a shared model, where the central bank is responsible for the CBDC and principles for secure settlement, while banks and other actors manage tokens representing bank deposits, securities, and other assets. The actual operation of the ledger and the core infrastructure may be suitable for joint organisation, ie through a joint venture.

Test case 5 has not had the objective of investigating how privacy considerations are taken into account. However, the test has highlighted structural limitations in single-ledger architectures based on EVM-compatible platforms, such as Hyperledger Besu. Privacy is mainly based on pseudonymity, which does not provide adequate protection if unauthorised parties can link addresses to identities. Role and access control in smart contracts limits the actions that actors can perform, but does not protect against analysis of transaction patterns at the ledger level. Experience from other test cases indicates that alternative DLT platforms, such as Hyperledger Fabric, have a better technical basis for confidentiality through more granular access control for data visibility. Additional privacy measures, such as anonymisation layers or layered architectures, have not been explored in this test.

Test case 6 — Issuance of government bonds on a single shared ledger

The purpose of the test case

The purpose of this test was to determine whether the entire life cycle of government bonds could be tokenised and recorded on a single shared ledger and executed using smart contracts. The entire transaction chain, from issuance via an auction, trading on the secondary market, coupon payments, buybacks, and redemption at maturity, was handled through smart contracts. Government bonds, TBD, and wholesale CBDC were represented as separate tokens. The settlement was to be carried out atomically — so that either both the settlement of money and securities was carried out simultaneously, or none of the transfers of ownership are carried out.

Tokenised securities are often highlighted as a use case with great potential. The idea behind the test was to explore:

1. Whether it would be possible to easily expand the ledger to handle the issuance and life cycle of government bonds.
2. What advantages and drawbacks did tokenised securities offer issuers and traders.

Technical design

The solution is designed as a hybrid model where parts of the process are handled on ledgers (on-chain) using smart contracts, while certain operations are carried out off-chain for practical and regulatory reasons. This applies in particular to bid opening and allocation of securities in the auction process, which is not fully automated on the ledger.

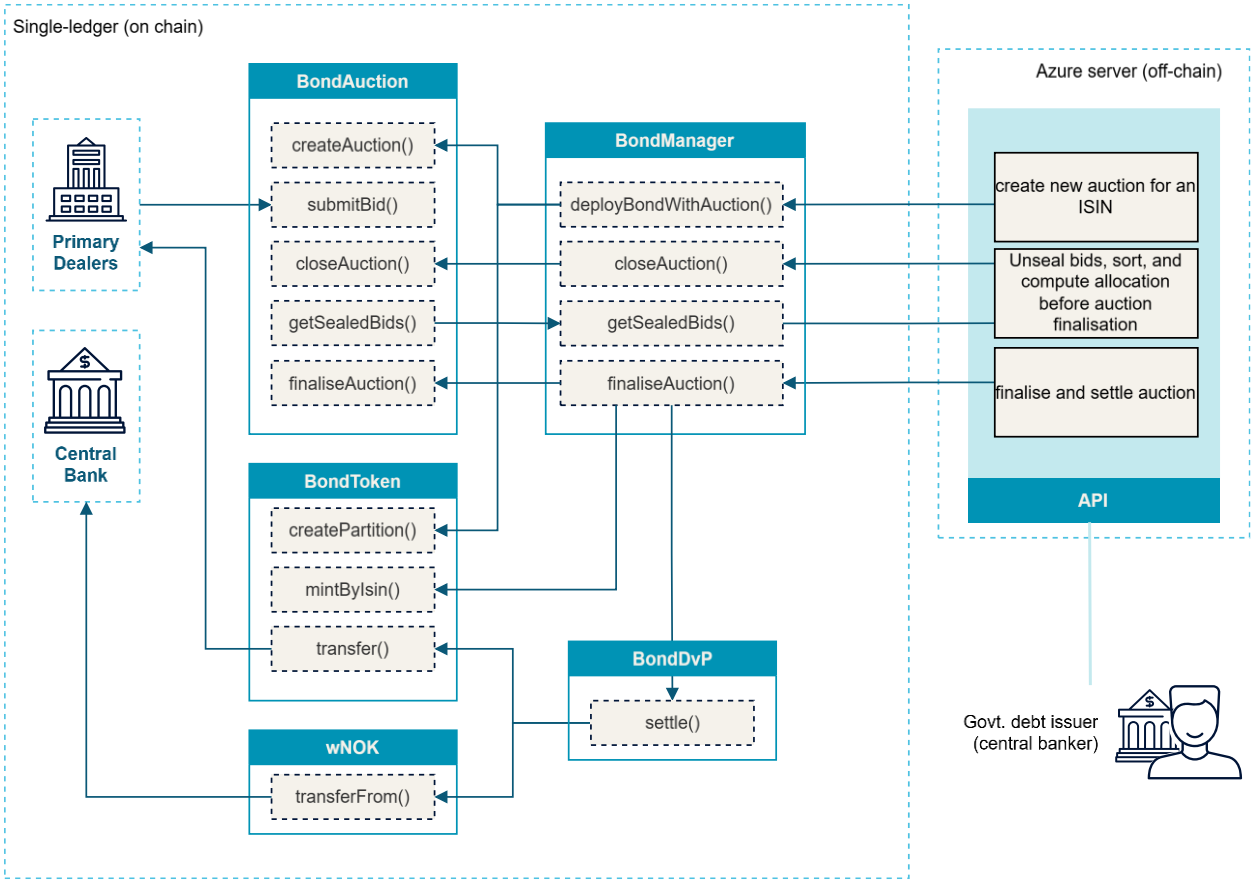

Figure 12: Overview of smart contract architecture for tokenised government bonds

Figure 12 shows an overview of how the smart contract system for tokenised government bonds is implemented in the sandbox. The entire system is controlled by a central

component, '*BondManager*', which handles the initiation of new auctions, final deadline for submitting bids, allocation of securities, and finally the settlement of the money and security leg. This contract calls on other, more specialised contracts, which handle settlement, for example.

The overview in Figure 12 shows four main processes in the auction cycle for government bonds. These include the establishment of the auction, the submission and registration of bids, the closing and resolution of the auction (including the decision on which bids are accepted), and the execution of allocation, issuance of bond tokens, and the associated settlement. In addition, there is a mechanism for extending an existing loan (ISIN), but this is not shown in the figure.

The auction process begins with the central bank creating a new auction through the API interface. This part is initiated outside the ledger. The API interface then activates the relevant smart contracts, which create both new bond tokens and the auction itself on the ledger. Once this is in place, the auction is opened for bids from the primary dealers.

During the bidding phase, the primary dealers seal their bids using both the central bank's and their own encryption keys. This takes place outside the ledger to ensure confidentiality, while allowing the bids to be verified later. The sealed bids are then submitted to the auction on the ledger.

When the bid deadline has been met, the central bank closes the auction through the API interface. The API then triggers the '*BondAuction*' smart contract, which closes the auction on the ledger and prevents new bids from being received. After the auction is closed, the central bank opens the sealed bids through the API interface and verifies their content. This part of the process also takes place outside the ledger.

In the final phase, the central bank assesses whether the bids should be accepted. Via the API interface, the auction can either be completed or cancelled if the bids are not to be accepted. The result of the auction is published as part of this process. When the auction is completed, the API interface activates the *BondManager* smart contract, which initiates the settlement and allocation of the bond tokens through the underlying smart contracts on the ledger.

Finally, the API interface creates a time-triggered workflow outside the ledger. This workflow is automatically activated when it is time for coupon payment on the bond.

Extension of an existing bond:

If the central bank wishes to add volume to an existing loan (ISIN), almost the same routine as when creating an auction is followed. The extension is posted as an auction and follows the auction process mentioned above. An important difference is that the ISIN is retained, and the coupon rate is unchanged.

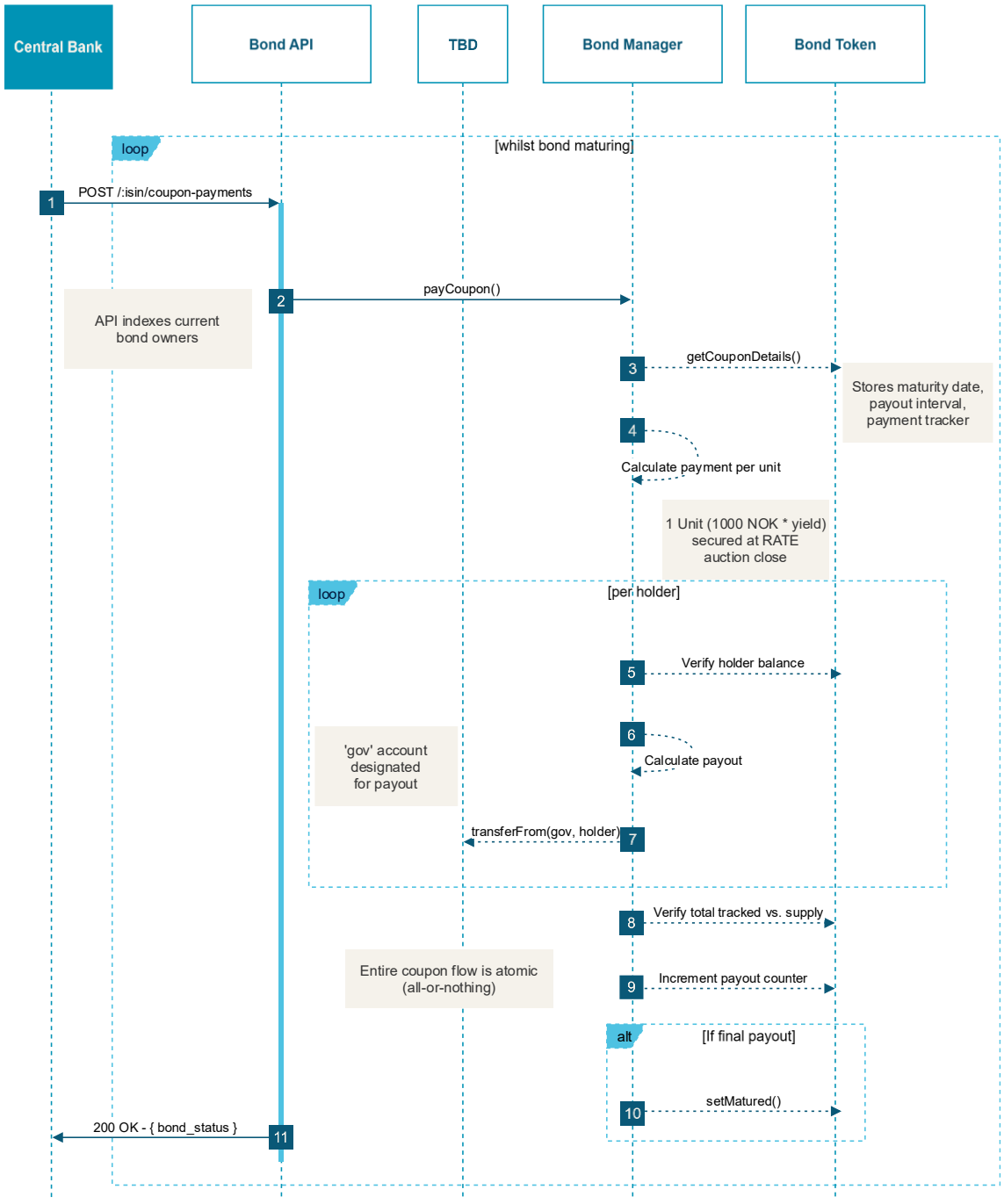

Figure 13 Process diagram for the issuance and auction of tokenised bonds

Coupon payments

In order to execute coupon payments and redemptions at maturity, the smart contracts must have access to externally generated overviews of the bondholders. This is because the contracts themselves do not identify all the addresses that are entitled to disbursement at any given time. These ownership lists are therefore produced outside the ledger and used as the basis for the operations that are then performed on the ledger.

The model in Figure 14 is a simplified representation of the current bond cycle. Coupon payments are repeated at fixed intervals until the bond reaches its maturity date.

When the time for a coupon payment arrives, the process begins with the central bank calling the BondAPI for the relevant bond, identified by its ISIN. The API then activates the smart contract function *payCoupon()* in *BondManager*. At the same time, the API ensures that there is an up-to-date overview of who owns the bond at this point in time, so that the disbursement goes to the correct recipients.

The *BondToken* contains key information about the bond, including the maturity date, the interval between coupon payments, and a counter showing how many coupons have already been paid. *BondManager* retrieves this information and then calculates how large the coupon should be per unit. A unit is linked to a nominal amount, for example NOK 1,000, and the coupon rate is the one set at the original interest rate auction.

BondManager then reviews all registered bondholders. For each holder, the holdings in *BondToken* are checked and the individual coupon payment is calculated based on how many units the holder owns. Once the amount has been determined, a transfer is instructed from a dedicated 'gov' account (represented by TBD) to the relevant bondholder. In this way, each owner receives the coupon payment corresponding to their own holdings.

The entire coupon payment is defined as atomic. This means that either all disbursements are made collectively, or the entire operation is reversed if something goes wrong. This reduces the risk of partial or inconsistent disbursements.

Once all owners have received their coupon payments, the *BondManager* checks against the *BondToken* to ensure that the total coupon paid out matches the total bond holdings in the system. The counter in *BondToken* is then updated so that the system registers that the current coupon period has been completed. If this was the last coupon payment according to the bond structure, *BondManager* calls a function in *BondToken* that marks the bond as matured.

Finally, the API sends a status message back to the central bank. This confirms where the bond is in its term and whether it has now reached maturity.

Redemption

When the bond has reached maturity and all coupons have been paid, the central bank starts the redemption phase, cf. Figure 14. The central bank calls the *BondAPI* against a separate endpoint for redemption, which in turn activates the *BondManager* function *redeem()*. BondToken maintains a status (*isMature*) that shows that the bond has reached maturity, and this status is used by *BondManager* as the basis for redemption.

The *BondManager* then retrieves relevant information from the *BondToken*, including what is needed to determine what the bondholders will receive upon redemption. In theory, the amount can be calculated based on the remaining payments on the bond. When the bond has matured, the principal shall be repaid, possibly together with the last coupon payment.

Furthermore, the *BondManager* reviews all remaining bondholders and verifies each individual holding in *BondToken*. The redemption settlement itself takes place via the *BondDvP* component, so that payment and cancellation of the bond tokens take place simultaneously. The *BondManager* instructs *BondDvP* to carry out the settlement, whereupon *BondDvP* calls *BondToken* to redeem the owner's holdings through *redeemFor(holder, balance)*. In practice, this means that the bond tokens are removed from the owner's address. At the same time, the settlement amount is transferred from the 'gov' account (TBD) to the same owner.

The redemption flow is atomic. Once the redemption flow is completed, the *BondManager* verifies that the bond tokens are destroyed. The bond is then fully redeemed and closed in the system. The *BondAPI* then sends a status report to the central bank confirming that the redemption has been completed.

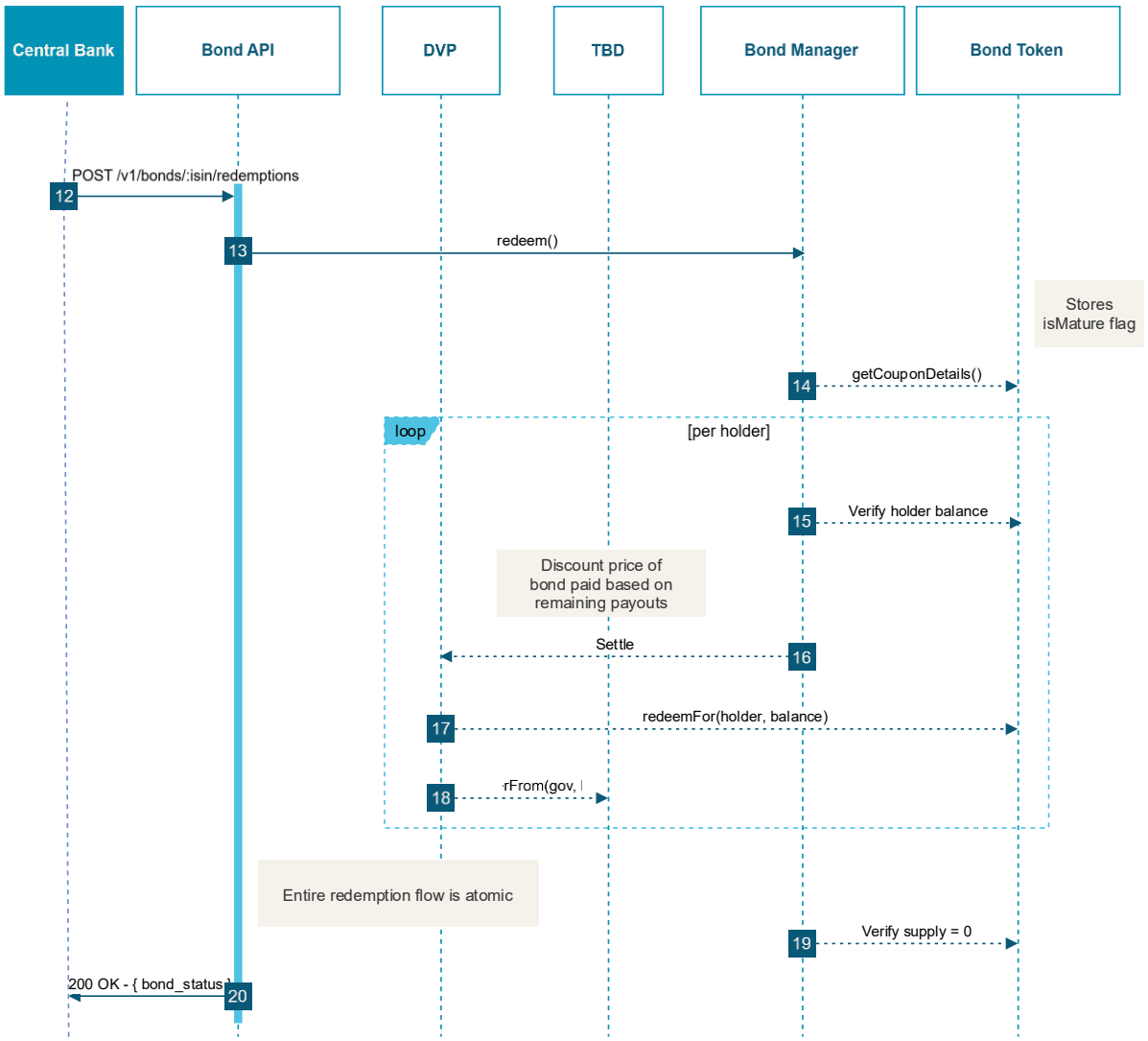

Figure 14: Process diagram for coupon payments to owners of tokenised government bonds

Results and findings

The test shows that a ledger can be used to manage large parts of the life cycle of government bonds, from issuance and trading to coupon payments, buybacks, and redemption at maturity. Important state changes and settlements can be automated through smart contracts, which provides good traceability, control, and efficiency. At the same time, the test shows that certain key processes still need to be handled outside the ledger, in particular the opening and allocation of bids in auctions, as well as the identification of current bondholders prior to coupon payments and redemption.

Atomic DvP has been implemented in line with its purpose. This means that ownership of securities and payments are settled simultaneously, and that the entire transaction is

cancelled if one part fails. This reduces the counterparty risk and appears to be a strength of the solution. The auction process is conducted with encrypted bids ('sealed bids'), where bids can only be read outside the ledger, while the allocation decision is recorded on the ledger with a signed cryptographic proof. This provides good verifiability, but does not provide complete assurance that the allocation is automatically correct in relation to all bids.

The solution could strengthen the integrity, control, and rights management in auction implementation, but some processes in the test are not fully automated and need to be supported by external routines (off-chain). Confidentiality and privacy on the ledger have been improved through encrypted bids and strict access control, but certain information, such as activity patterns, timing, and number of bids, can still be observed on the ledger. The findings thus illustrate the need for a balance between what should be handled on the ledger and what should still be resolved outside it.

Summary of findings

The technical testing has provided practical insight into how CBDC based on ledger technology can be used in payment and settlement processes, and how architecture choices largely determine the benefits, limitations, and role of CBDC. Across the test cases, architecture emerges as the most important factor determining which capabilities can be realised with this technology.

A key finding concerns the distinct characteristics of single-ledger and multi-ledger architectures. When money, assets, and business logic are handled on a single shared ledger, settlements can be executed simultaneously and atomically. This enables integrated DvP, reduces counterparty risk, and makes it possible to enforce rules, roles, and rights directly in the infrastructure using smart contracts. In such setups, CBDC serves as a functionally integrated part of the processes.

In multi-ledger architectures, where assets and applications are distributed across separate ledgers, the link between transactions must instead be established through bridges, messaging, or external orchestration. The tests show that this leads to greater technical and operational complexity, reduced simultaneity, and increased settlement risk. At the same time, the tests indicate that several key benefits of ledger technology, particularly those related to close integration between money and business logic, are not realised to the same extent as in single-ledger architectures.

The testing also shows that when settlement and associated logic are consolidated on a single ledger, multiple operations can form part of a single atomic process. This can, for example, yield efficiency and liquidity gains, including netting-like effects, without deferred settlement or intermediate counterparty risk. At the same time, the tests indicate that certain functions and processes are not suitable for execution directly on the ledger, and that an appropriate architecture will therefore involve a deliberate combination of on-chain and off-chain components.

The tests have mainly focused on technical feasibility and conceptual properties. Issues such as performance under load, handling of errors and deviations outside the 'happy path', operational management over time, security, contingency, and interaction among a larger number of actors have received limited attention. Further work should also address how requirements for anonymity and privacy can be better met, for example through layered architectures or the use of secondary layers (layer 2) above a core ledger. These conditions should therefore be a focus for further testing, as well as specific applications where CBDC and ledger technology can provide added value beyond replicating current structures. This could include testing interaction in broader ecosystems, assessing incentives and governance models, and exploring architectures that enable genuine integration of money, assets, and processes.

Publishing source code

Availability of Source Code and Use of the Sandbox Solutions

The source code underlying the sandbox solutions and test cases described in this report will be made publicly available as open source. The purpose is to promote transparency, reproducibility, and knowledge sharing, and to enable other stakeholders to use and explore the source code for their own testing.

The code has been developed to support the specific test cases and experiments conducted in the project and is tailored to simplified processes and controlled sandbox environments. The repositories can be forked and run locally in dedicated development environments, allowing interested parties to set up sandbox solutions and test functionality and architectural choices on their own machines. The code may also serve as a starting point for further experimentation or for comparison with alternative technical solutions. The source code will be made available in Norges Banks Github repository[29].

[29] See Github (2026).

Limitations and disclaimers

The code is **not intended for use in production environments** and must not be used as the basis for real-world payment, settlement, or securities systems. Comprehensive assessments of security, robustness, performance, privacy, or regulatory compliance have not been carried out, and deliberate simplifications have been made in both architecture and implementation to enable efficient experimental testing.

All use of the code is at the user's own risk. Any further use or development requires independent assessments of technical, security, legal, and regulatory aspects.

References

BIS (2025) *Annual Economic Report 2025*, Chapter III. The next-generation monetary and financial system. (link)

BIS (2023) *Annual Economic Report 2023*, Chapter III. Blueprint for the future monetary system: improving the old, enabling the new. (link)

Github (2026) Official CBDC lab repo of Norges Bank. (link)

Linux Foundation (2026) Linux Foundation Projects (link)

Norges Bank (2026) Central bank digital currency – final report for project phase 5. Norges Bank Papers 1/2026, 23.03.2026. (link)

Norges Bank (2022) "The central bank. *Strategy 25*". (link)

Store norske leksikon (2026) Definition of 'blockchain'. (in Norwegian) (link)

Norges Bank
Norges Bank Papers

Oslo 2026

Visiting address: Bankplassen 2
Mailing address: P.O. Box 1179
Sentrum, 0107 Oslo, Norway
Telephone: +47 22 31 60 00
E-mail: post@norges-bank.no
www.norges-bank.no

ISSN 1894-0277 (online)
ISBN 978-82-8379-399-4 (online)