A new digital currency system

Kees van Hee[1] and Jacob Wijngaard[2]

1 Introduction

In this paper we give a motivation for Central Bank Digital Currency and we describe a model for a Central Bank Digital Currency-system (CBDC-system). The main function of such a system is facilitating payments between *economic actors*, such as households, businesses, banks and the government. The currency in this system is only so-called *base money* created by the Central Bank (CB). Although we are in favor of a cashless society with only one digital currency, the CBDC-system can also function when these wishes are not fulfilled. There are many papers about the pros and cons of such a system, but very little attention is paid to the technical and organizational feasibility of such a system. Our view on CBDC is very close to that of Bordo and Levin (2017), although we dive deeper in the technical and organizational issues. Most authors, including Bordo and Levin, seem to think that it is possible to make a digital look-a-like of a coin or bank note that can be transferred between two parties in isolation. That is not the case and we show how to deal with this.

In this paper we present a model of a CBDC-system which can be seen as a *proof-of-concept*, i.e. a proof that it is feasible to build such a system. So the model is not a *blue print* for such a system. The model is useful in understanding what is possible, impossible and what is difficult or easy to realize. For example one could use the model to evaluate the requirements formulated by monetary authorities, e.g. the ECB (report on the digital euro, ECB (2020)) . Sometimes system requirements are such that it is impossible to build a system obeying the requirements, or it is extremely complicated to build it.

The system we propose differs essentially from the existing banking system. For instance the digital currency is not stored at a bank, but with the actor and banks play no role in money transfer. But many features are similar to the existing monetary system which makes migration and public acceptance easy. The system we propose is meant for base money only and therefore we call it CBDC-system, but it can be used for other monetary systems as well.

In section 2 we describe the drawbacks of the existing system and we give arguments for a CBDC-system. In the sections 3 until 5 we describe the CBDC payment model. Section 6 shows

[1] Professor emeritus, TU/e, ICT
[2] Professor emeritus, RuG, Operations

how this payment model facilitates additional financial functionality. In section 7 we sketch the new role of the banks. In section 8 we sketch the broader potential of the payment system: how it can help in preventing tax evasion, how it can be used to improve the payment of VAT and income tax. It helps to create the possibilities for new rule-based monetary policies. In section 9 we discuss the implementation and migration process. Performance issues are included here. Conclusions are formulated in section 10.

The CBDC-system heavily leans on modern cryptography. In the paper we left out as much as possible of these techniques, but in the appendix we give in a nutshell the relevant notions of cryptography.

2 Motivation

The present monetary system has two important drawbacks and both are caused by the commercial banks. The first drawback is that the vast majority of our money is created and stored by banks. Today we have two forms of money, *cash* (coins and bank notes) and *demand deposits* (balances on current bank accounts). Cash is part of the so called *base money*[3]. The rest of the base money is invisible for normal *economic actors*. It consists of the *reserves* of (commercial) banks and the government, at the Central Bank (CB). A demand deposit is (only) a *claim* on base money. Such claims are generally accepted and form the main part of the available money. Today ca 95 % of our money is claims on base money and if all economic actors would cashing in their claims, this would be a disaster because banks don't have the base money. This is one of the drawbacks of the existing system. The most fundamental element of the debate about the role of the banks is the question whether banks should be allowed to *create* money in the form of demand deposits. Bank credits are a strange form of money. Nevertheless, from 1971, after the abolition of the Bretton Woods agreement, it has functioned well for a while. By adapting the interest rate for reserves, the availability of credit was controlled, and through this the whole economy. And especially during the period 1985 – 2005, the system appeared to be really "under control". That is why that period is called "the great moderation". In between, however, there are serious doubts. It is clear that the banks have played an important role in the emergence of the financial crisis. The American mortgage market was the biggest culprit. But the lack of transparency and the sale of too complex financial products contributed as well[4]. The structural freedom of banks due to the current monetary system is often seen as the root cause. There are different proposals for

[3] See e.g. Ryan-Collins, Greenham, Werner and Jackson (2011)
[4] See e.g. Roubini and Mihm (2010)

improvement. Sharper restrictions with respect to liquidity and solvability[5], better monitoring and control, narrow banking (banks concentrate on payment and saving and give credit only insofar as that can be guaranteed absolutely)[6].
There are general rules with respect to reserves and liquidity (Basel I, II and III). But the position of a bank is judged afterwards and the judgement of the different categories of assets and the validity of the rules are not always clear[7]. This implies that the banks have in fact a large freedom with respect to the creation of money because the claims are rarely cashed.

The second drawback of the present monetary system is the cumbersome way the money transfer is performed by commercial banks, often with help of card schemes like Master card or Visa. It is very complicated and therefore inefficient. The payment role of banks emerged for practical reasons: if an actor X has to transfer an amount A to actor Y then X has to go to the bank and withdraw the money then move to Y and then Y should deposit the money at his bank. If X and Y happen to have the same the bank then it is easier to subtract the amount A of account X and add it to account Y. So then the bank performs the payment. In case they have different banks a similar procedure could be done where the banks do the money transfer for a batch of actors. It is done in two steps *clearing*, the update of the accounts and *settlement* the transfer of money between the banks, which is less than the sum of the individual transfers because transactions in a batch can cancel out each other.

We propose to give all economic actors access to the base money and use that for all payments, through a payment system that functions independently of the banks. The option to give all economic actors access to base money has been and is being explored rather widely already. For instance in Sweden and in the Eurozone[8]. But in these explorations the current banking model is not disputed. CBDC is seen as a form of cash, digital cash, next to the still existing physical cash. Such a hybrid system is very ambiguous. On one hand actors get access to base money, but on the other hand this access has to be restricted to protect the current banking model. In our opinion such a system is doomed to fail. Our proposals go much further and are in the same spirit as the proposals by the Positive Money movement in the UK[9]. These proposals have not received serious attention in the main stream literature[10], but the suggestions of Bordo and Levin (2017) come close. They propose also a more complete switch to CBDC although they leave the role of the banks more or less unchanged.

[5] See e.g. Admati and Hellwig (2013)
[6] See Kay (2009)
[7] See e.g. Admati and Hellwig (2013)
[8] Sweden was rather early (see Sveriges Riksbank (2017)). The Bank for International Settlements gave it also serious attention (see BIS (2018)) and now the ECB is exploring the possibilities (see ECB (2020))
[9] See Jackson and Dyson (2012)
[10] This is clear from the special issue in the Cambridge Journal of Economics (see Ingham, Coutts and Konzelmann (2016)

In the CBDC-system we describe here, only the CB can create and destroy money. Commercial banks have as main function savings and lending. Probably they will also develop new financial services on top of the CBDC-system. Of course there is a need for borrowing money in the market. The CB will only lend money to banks.

The CBDC-system has the following advantages:

- There is only digital currency created by the CB
- Money is stored by the actors themselves
- Money transfer is very easy: only the essential functions are performed by the CBDC-system in a very efficient and secure way
- The system is an ideal platform for additional financial services (see section 6)
- The system has good features to avoid money laundry and tax evasion (see section 8)
- The system has good features for rule based monetary policies (see section 8 and Wijngaard and Van Hee (2021) for an elaboration)

Many people are afraid of a centralized system where a Big Brother could control everything. For that reason crypto currencies were invented. The first one in 1983 by David Chaum (Chaum (1983)) and later in 2008 the Bitcoin by Satoshi (Satoshi (2008). From the latter system there are many new variants available. In our CBDC-system the information stored is extremely limited. The system is only facilitating transactions and does not store these data and so the system does not know the balances of the actors. So the Big Brother problem is solved in this CBDC-system.
A big problem of crypto currency is the *double spending* problem: how to avoid that the same money is spend twice or more times. That is why systems like the bitcoin system have introduced the so-called block chain which is extremely time and energy consuming. In our system double spending is solved by the CBDC-system in a trivial way. So that is also an advantage .

3 Essential features of the CBDC-system

In principle it is possible to construct a digital look-a-like of a physical currency, i.e. a coin or bank note. However, that is unnecessarily complicated. This is also noted in Bordo and Levin (2017), but here it will be explained in more detail. If we pay with cash we always have to look for the right coins and then we often receive change in return because we did not have the exact change. It is much easier to have only a e-wallet with one amount, the *balance* of an account and then one can pay any amount less or equal to the balance. So it is more efficient to store one amount: the balance of the account. This is the same as in the well-known banking system that we are using today, and that feature is worth to keep in a CBDC-system. Even in the *bitcoin system* there are also no “digital coins” or “digital bank notes”, although the term “bitcoin” suggests this. However there the *transactions* are stored, which means that if actor X pays an amount A to actor Y then Y records this transaction and he may spend this A, i.e. transfer it to some other actor, at a later stage. So in order to pay a large sum the payer has to find enough transactions to do the payment and if there is not an exact match, he has to transfer the change to himself as a new transaction. This is even more laborious than a cash equivalent. Coins and bank notes have complex marks on it and bank notes have a unique identifier, a string of characters . One of the main characteristics of physical currency is that it can be transferred between two actors in *isolation*, i.e. without any contact with a third party. Of course a forger can try to ‘copy’ a physical currency. In order to verify a coin or bank note one inspects the marks on the coin or bank note. It is practically impossible to verify if the currency is unique, i.e. if its identifier does not occur twice or more times in the monetary system.

Suppose a digital equivalent of a physical currency unit is just a single sequences of bits. Then we could encode this bitstring such that it can be verified that it satisfies all the official characteristics of the currency (see appendix). But it is always very easy to copy it! And then we have a new currency unit that satisfies all the criteria. So we cannot conclude that it is a copy. This is the big difference between physical currency and digital currency: it should be impossible to spend the currency twice. This *no double spending* is one of the most important requirements of the system.

It is impossible to guarantee that a bitstring is not copied and so to prevent that digital currency in the form of just a bitstring is spend twice or more times.

In the *bitcoin* system there is a *public database*, called a *blockchain* that keeps track of all bitcoin transactions. The owner of bitcoins possess a *reference* to this database. And in this database data can never be changed, only data can be added. So if there was a payment with some currency unit then that can be traced and so it cannot be repeated.

Also in our system, with accounts and balances, there is a public database besides the information people store in their own database, i.e. the *digital wallet* on a smartphone. The information in the public database is essential in preventing actors to spend their currency unit twice. This implies that it is not possible to perform so-called *off-line transactions*, because it is always necessary to verify that the client is not spending the same currency twice[11]. However it is possible to have an off-line payment system on top of the CBDC-system. This off-line payment system can be provided by banks or other financial service providers (cf. section 6).

[11] The ECB, in her report on the digital euro, (see ECB (2020)) stresses the importance of offline functionality. It is questionable whether they are aware of the full consequences

An important requirement is that the CB does not become “Big Brother” so it must facilitate payments but it should not keep track of the money the actors possess. In the next section we will show that it is sufficient to store a “fingerprint” of the transactions.

4 Building blocks

We consider one currency zone, like the dollar zone or the euro zone. All currency is CBDC and only the CB can create and destroy currency. Money transfer is done through the system. So banks don’t play a role in payment processing, but they will still play an essential role in the economy (see section 7).
In the following description we use as little as possible notions from cryptography. In the appendix these notions are elucidated. What we need is a *fingerprinting* mechanism and a *secure message passing* system.
The system has the following building blocks:

- Actors have *accounts,* as many as they like. An account is an abstract object that is “owned” by the actor and that is stored on the hardware of the actor or in a *cloud* of a service provider operating on behalf of the actor. The account has several properties such as a *unique account identity* and the identity of the actor. But the most essential property of the account is the *balance*, i.e. the amount of currency on the account. This balance is at least zero, so never negative. An exception is made for banks. If they have a license of the CB, they are allowed to borrow from the central bank. In order to do this, banks have, besides the normal accounts at the CB also *C-accounts* (credit accounts), with a balance <= 0. All other actors may borrow money from other actors, in particular from banks, but not from the CB. It is also possible that the government has C-accounts, so that the government can borrow directly from CB.
- There is a *public database* as part of the CBDC-system and in that database there is some data of the accounts, but neither the balance itself nor the transactions themselves. What is stored is a *digital fingerprint* of the balance account. One way to realize this is by a so-called *hash function*, also called a *one way function*. A one way function is a function H such that it is easy to calculate H(X) for some character string X, but if H(X)=Y and only Y and H are known, then it is practically impossible to calculate X. This fingerprint is necessary to verify if the account information of the actor is not manipulated. The public database is called “public” because all actors having access to it. We avoid the term “central” database because it will be a distributed database, i.e. a network of databases, but all under control of the CB.
- There is a *clearing house* function in the CBDC-system. Here the most basic payment action is performed: the increase of the balance of the *acquiring* actor and the decrease of the

balance of the *paying* actor. The well-known *settlement* function is not necessary because it is just the two actors and the public database.

- A *secure transmission* system to send messages between actors and the CB. This transmission system could be independent of the CB and it should satisfy the following properties:
    - *Integrity*: messages should reach their destination without changes made by others.
    - *Confidentiality:* nobody is able to read the content of the message except the addressee. This property can be guaranteed by proper encryption.
    - *Authenticity*: the addressee can verify the senders identity. This is done by a *digital signature*.

These transmission requirements are realizable by todays encryption technology (see the appendix). It is possible that these systems will improve in the future, specifically if quantum computers can be used, but the functions of the CBDC-system will be the same (see section 6). With these building blocks we will construct the CBDC-system. The CB can be seen as a *trusted third party* for payments between two actors. But this does not mean that the CB keeps record of the values of balances of the actors. We only let the CB keep a fingerprint of the balance and maybe also of the transactions. The last is not necessary for the payments but can be useful for other type of verifications (see section 8).

5 Payment processing

Here we will sketch the main function of the system: the transfer of money from one account to another. Each actor has one or more accounts in the CBDC-system. And because it is a very large system, for a whole payment zone, we assume that the public database of this system is distributed, which means that it is a *network of connected database servers*. We may call it a *dedicated computing cloud*. Each account is assigned to a specific database of the network. We call them *CB-server* or just *server*. Each of these servers stores many account objects. Remember that from each account only the fingerprint of the current balance and maybe the fingerprints of transactions are stored here. Further all actors store their current balance record on a private server. This can be a PC at home, a smartphone or a server in the cloud from some service provider. It is possible and advisable to store the balance record in more than one place. We call such a private server an *e-wallet,* which term is usually restricted to storage on a smartphone. The data stored in these databases is as follows.
The e-wallet stores *account records* of the form:

[accountid: X, date: T, balance: B, nonce: N]

Where X is the identity of the account, T the date and time of last update, i.e. of the last money transfer, B is the balance at that time and N is just a random sequence of bits called *nonce*, generated by the CBDC-system at the last transfer. The nonce is a technicality used for the fingerprint. We call this record the *e-wallet record*.

The database of the CB-servers stores records (referring to these wallet records) of the form:

[accountid: X, date: T, nonce: N, fingerprint: H (X.T.B.N)]

Where the function H is a one-way function for character strings and X.T.B.N is the character string formed by concatenation of the strings X, T, B and N. We call this the *CB-record*.

Of course the CB-server stores also the identity data of the actors and which accounts they possess. But we do not consider them here.

An actor might want to verify that his account record is the current one, for instance to prove his credibility. He then has to send it, in a secure way, to the CBDC-system and the system reads X.T.B.N from it, computes H(X.T.B.N) and compares it with the fingerprint in its own account record of the actor. If they are identical, then the CBDC-system can confirm the correctness of the e-wallet account.

For other purposes than pure payment, it is useful to add another attribute in the account records: a *sequence number* of the balance update. With such a number it is possible to see if a set of updates is *complete*, i.e. not missing an element.

Note that if the actor loses his e-wallet record he loses the money as well! Therefore it is recommended to make copies of these records in a secure way. If it is stolen by some intruder that person can only read the balance but he can't withdraw money from it, because the intruder does not have the digital signature. The intruder can also change the account record in the e-wallet, but that makes it invalid. This is as serious as losing the record, but no intruder can steal the money or put money on the account, since the CB-server has the fingerprint of the current e-wallet record and only the actor can send a payment instruction with his digital signature.

Next we consider the *kernel protocol* to transfer an amount A from actor X to actor Y

First of all X and Y should reach an *agreement* to transfer the money. This will be done outside the scope of the CBDC-system. But at some point in time they will both produce a record that shows their intention and is sent with their digital signatures to the CBDC-system.

These *common transaction records* will look like:

[from: X, to: Y, amount: B, label: L]

The label L could be empty, but there are good reasons to label a transaction. Examples are an invoice identity or a VAT classification. Note that the CBDC-system does not do anything with this label, except that it is part of the fingerprint.

The *kernel protocol* has the following steps:

1 X and Y agree on a money transfer of amount B from X to Y. They produce both the same (common) transaction record. This first step is performed outside of the CBDC-system.

2 X and Y both send each their account record and the common transaction record in a secure message, signed with their digital signature, to the CBDC-system, in fact to their own CB-server.

3 The CB-servers check the authenticity of the messages by the digital signatures.

4 They check the account records (by checking the fingerprints).

5 If the balance of X is insufficient for the transaction, then the CB-server of X sends a return message with a disproval and also a similar message to the database of the acquirer, who returns a similar message to the acquirer.

6 If the balance of X is sufficient, his CB-server sends the transaction record to the CB-server of Y.

7 The CB-server of Y compares the (common) transaction records of both actors and if identical, which proves that they both agree upon the transaction, he updates the balance of the account of Y by adding B and confirming it to the CB-of X.

8 Then the CB-server of X updates his account by subtracting B from the balance, and informs the CB-server of Y.

9 Then both CB-servers send to their actors a secure message with digital signature and the updated e-wallet record with a fresh nonce.

10 Both CB-servers delete the transaction information after updating their CB-record with the fingerprint as described above.

As mentioned before, it is possible, and recommended, to extend the storage of the CB-servers with fingerprints of the approved transaction records besides the account records This creates the possibility to verify the *transaction history* of an account. We emphasize that it only facilitates the *verification* of a sequence of transactions that is presented by the actor or service provider on behave of the actor. The reason is that the system stores only the fingerprints of the transaction records and not the transaction records themselves.

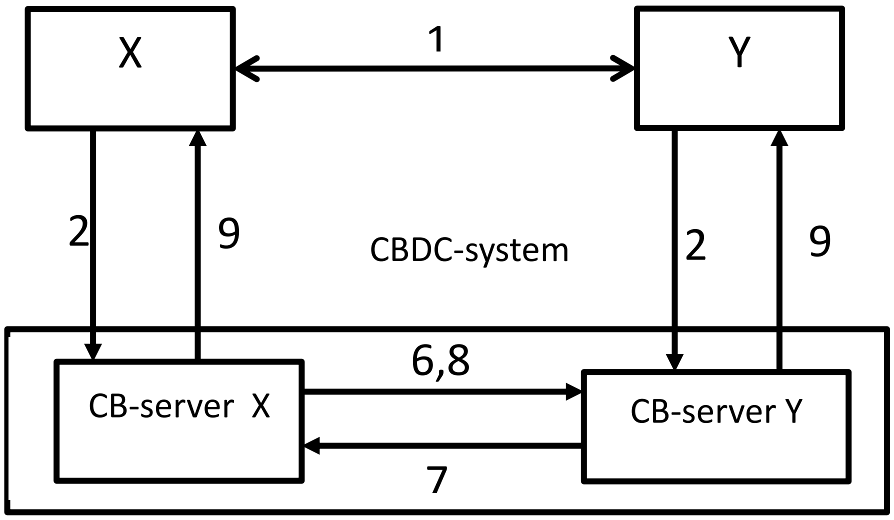

Fig.1

In the figure we display the different components of the CBDC -system and the numbers refer to the messages produced in each of the 10 steps of the protocol. It is possible that X and Y share the same CB-server, i.e. the CB-server X equals CB-server Y. The kernel protocol is very simple and is the essence of money transfer. So it never has to change in the future. Maybe there will come better ways of secure message passing or better one-way functions, but that does not change this protocol. The protocol is so simple that it must be possible to give mathematical proofs for the correctness of an implementation, which makes verification by edp-auditors superfluous.

The protocol looks very similar to the way the banks operate today. The essential differences are: (1) the use of CBDC (i.e. base money) instead of demand deposits (claims on base money), (2) a clear split in the currency: the money of an actor is his account balance and that balance is stored by the actor in his e-wallet and not at the CB and (3) commercial banks don’t play a role in this process.

The CBDC-system does not store the balance but is able to verify the content of the e-wallet. So the CBDC-system can’t do anything with the money of the actors.

Note that the communication within the CB-server can be secured in the same way as between these databases and the e-wallets. But since it is one system it can be done with less security.

6 Additional functionality and financial services

When two persons want to transfer money from one to the other, they can use their smartphones. There should be an app (application) such that one actor creates the (common) transaction record by asking first the account identity of the other and sends the transaction record to the other. Both send it, with their e-wallet record that they store on their smartphone, to their CB-server. The rest is as the kernel protocol. From the user point of view this will not differ much from the payment apps that are available today. Households will have several accounts and they will have applications to balance their e-wallet records. A similar procedure can be applied for payments with PC's or tablets. The e-wallet can be stored locally or somewhere in the cloud.

When a consumer wants to pay for products in a shop a similar procedure can be followed. The cash register of the shop determines the amount the consumer has to pay and the consumer provides his account identity either by a card or with his smartphone. Then the shop and the consumer follow the steps of the kernel protocol.

Note that these functions are in fact implementations of the first step of the kernel protocol and are outside of the CBDC-system.

In practice there are all kinds of special transactions, such as a *deferred payment* and the *direct debit.* The first occurs when the exact amount of a transaction is not known at the moment the transaction starts, e.g. in case of parking or refuel of gasoline. The protocol only differs in the first step of the protocol. First X and Y have to agree on a maximum amount that X will pay to Y in exchange of the delivered product or service. That has to be verified by the CB-server of X. Then the real (common) transaction records are determined and the transaction is handled by the kernel protocol.

The *direct debit* occurs when there is a contract between actors X and Y such that Y may collect automatically, which is usual for utility businesses. In this case the protocol starts with an authorization message from X without an amount and Y adds the amount in the answer message later. But here Y can send this kind of transaction messages unlimited until X sends a message to retract the authorization. This means that the first step can be organized as a preliminary process outside of the kernel protocol.

A typical service would be to offer *off-line payments* with a smartphone app. These payments could work in the following way:

- Actors create a special account for off-line payments and they put enough money on it from their normal payments account, usually small amounts for shopping or leisure like activities. This amount is stored by the smartphone app in a so-called *off-line wallet*. This account is blocked for payments not made by the smartphone app.

- When actor X wants to pay an amount B to actor Y, they have to agree on such a transaction with help of the smartphone app and then the common transaction record is stored on both smartphones. The total amount of the transactions generated in this way can't be larger than the amount coming from the special account of the payer.
- As soon as they make contact with the CBDC-system the transactions are processed in the order of occurrence and the content of the wallet is adapted. Only when both actors of a transaction had contact with the CBDC-system, the transaction can executed. This because both actors must agree upon a transaction.

In the practice of payment services there are more payment variants. But they can be implemented on top of the proposed CBDC-system, because front-end systems can preprocess more complicated transactions and derive the kernel transactions form it and send it to the CBDC-system So the system should provide sufficient public interfaces (API's) such that financial service providers, including banks, can create their own functions on top of the CBDC-system. This is precisely in accordance with the European PSD2 rules[12].

Besides the transaction-related functions the CBDC-system also has *management functions*, such as creation, combining, splitting and deleting accounts. But this is all quite standard functionality. It is important to keep the kernel protocol so small as possible and to have additional functions in the *financial service layers* on top of the CBDC-system. They can be delivered by fintech companies or banks could move into this direction. (see section 7).

A more comprehensive financial service is a bookkeeping service that adds information to transactions, such that they can be recorded directly in the general ledger. Existing services as factoring and credit loan are other examples. A new service could be automatic VAT computation and payment (see section 8).

Probably there will emerge service providers offering more complex transactions. Like we have seen in the former section they split these transactions up into kernel transactions. while they administer all kind of additional information for the actors This will be the playing field of new *fintech* businesses, but also old services of existing financial institutions will migrate to his layer.

Of course there will be trade with other payment zones. This can be facilitated by actors in the role of *valuta trader*. Such a trader has an account in each zone, e.g. one in the euro zone and one in the dollar zone. The trader receives from an euro actor an amount of euro's on its account and sends this actor, who also has a dollar account, a corresponding amount of dollars

[12] See the website of the ECB

on its dollar account in the other zone. He should have enough buffer in euros and dollars to play this game successfully. Often banks will play the role of valuta trader. In fact there do not exist "cross border" payments, but only payments between payment zones.

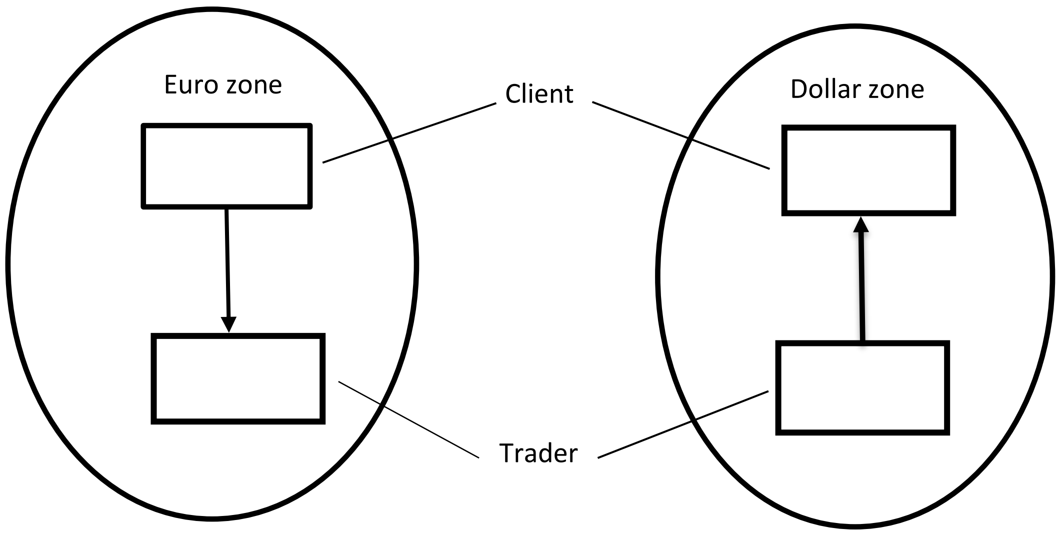

Fig.2

Of course there should be oversight on the services in the financial services layer. But the advantage of the CBCD-system is that it becomes much easier for providers of new services to enter this layer. Comparable with the app stores for smart phones and tablets.

7 New role of banks

Banks play today an essential role in payments and in storage of money. Both functions will be taken over by the CBDC-system. So banks will have to focus on mediation between actors with savings and actors that want to borrow money and their main task will be the analysis of credit risks. Actors who have superfluous CBDC can put it on a CBDC savings account, i.e. an account of the actor for which the bank has access rights. The bank is allowed to put this money on one of their own accounts (the *investment switch*) as soon as they have found borrowers. So the savings will be only a very short time on the investment switch of the bank. Of course the bank will pay less interest to the lender than it receives from the borrower. The difference is a reward for mediation and for the risk that the borrower is not able to refund. To guarantee that there are sufficient credit possibilities, banks have the exclusive right to borrow money from the CB.

The approach here is different from the PM approach (Jackson and Dyson (2012)). They propose that actors transfer their superfluous CBDC to the *investment pool* of a bank. The bank has to keep track of the amounts contributed by the different actors. It functions for these actors as a kind of savings account at the bank. It would be tempting then for the banks to develop a service where they perform payments between clients with such savings accounts. That would mean that they keep their existing payment system! Today banks periodically compute what they have to transfer for their clients to another bank and they also compute what they will receive for their clients from another bank (called *clearing*) and the difference of these amounts is 'transferred', called *settlement,* which means that the banks update their own accounts. Keeping this payment system should not be possible because it bypasses the CBDC-system and so the traceability of payments.
It is clear that this process requires several transactions involving the accounts of the CBDC-system. And therefore it will be a natural task of the banks to provide such transaction services for their clients. So banks stand foremost to provide additional financial services as mentioned in section 6.

8 Money laundering, taxation and monetary policy

Money laundering
In order to detect and to prevent money laundering the tax office needs the traceability of transactions. A CBDC-system *without cash* makes this easy. The CBDC-system stores fingerprints of balances and transactions. So the CBDC-system has no direct information of the financial data of the actors. But it can be used for verification purposes. So if the tax office wants to verify a certain financial transaction sequence it asks the actor to provide the transaction data. In order to verify that these data are the correct ones and that no transactions are missing, the tax office can compute the finger prints of these data and ask the CBDC-system for the sequence of fingerprints. If they are the same then the tax office knows the actor is honest. Of course this requires a retention obligation similar as what we have to day. In order to facilitate this process, a financial service provider could perform this transaction recording process for actors. This means that these actors do not use the CBDC-system themselves, but via their provider, for instance a bank!

Taxation
Transactions between businesses and households and also between businesses and businesses involve the payment of VAT. Instead of bookkeeping of these taxes and periodical settlement, an interesting possibility is to do these payments real-time. This means that each transaction should have some additional attributes identifying the type of transaction (using this label L),

and then the tax payment can be done automatically. Maybe this is also a good option for wage related taxes. Note that these attributes can also be useful for detection of money laundering. So this is typically a very useful additional service for businesses that can be combined with the transaction recording mentioned before.

Monetary policy

The focus of monetary policy is to keep the buying power stable and to have sufficient money in the system to facilitate the credits that are necessary for economic development. The main tools of monetary policy are the interest rate on base money and more direct interventions in the money quantity by buying or selling government bonds. Stability in general is important. That is why there has been a quest for general rules. The Taylor rule (Taylor (1993)) is one of the more serious proposals. The main stream conclusion is that a rule based policy is insufficient, that discretion is always necessary[13]. Bordo and Levin (2017) stress that a complete CBDC-system (and without cash!) gives new opportunities. The zero lower bound on the interest rate on base money is removed and they propose an adapted Taylor rule. See Buiter (2009) for a more complete analysis of the removal of the zero lower bound.
But a complete CBDC-system gives more opportunities than only breaking the zero lower bound. Instead of stabilizing the price level it is also possible to index the account balances, monitor the price development and compensate the account balance for possible increases. This option is also mentioned by Bordo and Levin (2017), but not selected. It can nevertheless be a useful option. A further option in this spirit is to link the account balances to some easy to monitor proxy of the GDP, e.g. the total sum of payments from households to businesses in the past year. Then the money is a kind of share in the economy and so it is not only fiduciary money but it has some real value. Being able to label the transactions is essential here again[14].

9 Implementation, migration and performance

It is very important that the migration of the existing monetary system to the new one proceeds smoothly. An incremental change strategy, in which a big change is realized in small steps, is preferable to a big-bang strategy where the new system has to be used at once. The introduction of the euro was a comparable operation with also a more or less incremental migration process.

[13] Taylor (1993) also stresses that it is always a combination, but that it is important nevertheless to have such a rule.

[14] In a separate paper we are going to come back to this issue of rule based monetary policies (Wijngaard and Van Hee (2021)

The implementation of the CBDC-system mainly concerns the organization of the accounts and the software to realize the transactions. We should start with the implementation of the kernel protocol. The banks can keep a big part of their existing functions so that they have the time to prepare for competition with other providers of payment services in the financial service layer such as payment service providers, ict-businesses, providers of cloud services, telecom operators and accounting firms.
Although the real operation requires a comprehensive planning, we sketch here only the most important steps:

1. The first step is to convert the money that banks have created into CBDC. This is just an administrative step: all banks obtain a C-account with a (negative) balance that equals exactly the money they have created, i.e. the demand deposits. It is a matter of monetary policy if the CB charges them with interest, either positive or negative.
2. For all actors CBDC-accounts will be created at the same time. Almost all actors will have already accounts at a bank and these accounts have a unique number. So it is obvious to use these numbers also for the accounts in the CBDC-system. In fact we copy the bank accounts into the CBDC-system. Note that negative values in the CBDC-system are not allowed, so if a negative bank account is "copied" , the bank should provide a loan in order to make the CB-account zero.
3. In the beginning the protocol is as in figure 1, but X and Y are the "bank of X" and the "bank of Y". So the digital payments will be performed via the banks: they will store the balance records for their clients and if an actor gives a payment order, the bank will produce the transaction record and retrieve the two balance records, send this all to the CBDC-system and receives the updated balance records for the client. If it is money transfer between two clients of the same bank, then it is easy, if the transfer is between accounts at different banks then the bank of the payer will request the balance record of the receiver at his bank. By the encryption there is no risk of fraud, but the banks have to protect the privacy of the clients, both the payer and the receiver. This step requires a software effort of the banks, but the clients do not notice the difference with the existing system. The banks can use the already available TIPS developments[15]. This gives banks an advantage as financial service provider. Once this step is realized, other actors may provide these functions as well. In principle everybody can do it himself and most likely big businesses will do that.
4. For consumer transactions, mainly in shops and hospitality and catering services, we must make a distinction between cash and card transactions. The cash transactions will die out in time. So an actor can pay with cash in shops for some time, but the change is added to his account from the shop account. Also at banks actors can convert cash to digital, i.e. the amount of cash is added to the balance record of the actor. But nobody can obtain cash anymore.

[15] TIPS stands for Target Instant Payment Settlement, the development launched by the ECB to realize real time payments. See the website of the ECB

5. Credit card transitions are similar as today: the credit card company is paying the bill and later he will invoice the consumer.
6. For debit card payments the existing systems have to be adapted. Today one uses in the euro zone mostly the C-TAP protocol, where the transactions are processed via the debit card organizations (e.g. Master Card, with Maestro, Visa with V-pay) via a payment terminal in the shop. This will change completely as described in section 6. The role of the debit card will change: it is only a way to communicate an account identity.

An important question is what happens if there is a major power outage. Of course all balance records should be stored in several places, for instance in the cloud. The transactions continue when the power is up again. Maybe a few transactions are lost and have to be renewed. During the outage there is still some payment possible between mobile devices and systems with power back up if the CBDC-system is still up. Of course the CBDC-system should have the best possible power back up. This is not different from the existing systems.

An important question is of course if this system is technically feasible.
To make a rough estimate of the computational efforts and the required storage capacity we focus on the euro zone. There are about 350 million people living in this zone[16] and if we add the businesses and other organizations we estimate a need for one billion accounts. With an average storage of ca 1 MB per account the CBDC-system needs a storage capacity of ca 1 petabytes.
There are about 2.10^{11} payment transactions per year in the Euro zone[17]. This means on average ca 6000 transaction per second and 10.000 in the peaks hours. The computing time per transaction per processor is small (ca 100 milliseconds). The computations consist of decryption and encryption of the messages and in between the trivial updates of the account records and the fingerprinting of these records. So we need 1000 processors to handle this payment workload. With 4 processors per server that means 250 servers each with 4 terabytes of storage capacity each, spread geographically over the Euro zone. With a proper *load balancing* algorithm the accounts can be distributed in such a way that each server in the CBDC-system has about the same workload. In this way the system is *scalable* and the performance is controllable. For safety and security reasons it is wise to back up copies of each account on several other servers while only one server is managing (hosting) the account. So the probability of the loss of an account can be made as small as the probability that a meteorite destroys the earth.

[16] see https://en.wikipedia.org/wiki/Eurozone
[17] see. www.ecb.europa.eu/stats/payment_statistics/payment_services/html/index.en.html

10 Conclusion

This paper is meant to contribute to the debate on the role of banks in the financial and monetary system. Specifically the creation of money by banks is considered as a mistake. The most natural way to remove the creation of money from the banks, is to make the base money available for all economic actors. Base money today consists of coins and bank notes and of the reserves of banks and government at the CB. In the form of coins and bank notes it is already available for all actors. But that is not scalable to large amounts and large number of transactions. A CBDC-system as sketched in this paper is a much better solution. It makes payments between two actors much easier and real time. It is feasible to create such a system and to migrate from the existing monetary system.
The CBDC-system is owned by the CB or the government. So it is in a sense a monopolist. But it is only a low level infrastructure for payments. Competing parties offering these basic functions would make it less efficient and less transparent. However the fintech layers on top of the CBDC-system offer plenty of space for entrepreneurship an competition. In fact the CBDC-system is a great enabler of competition in the financial industry. Also the CBDC-system can be a great enabler for monetary and tax policies[18].
We do not advocate a crypto currency like the Bitcoin. These systems arise from the fear of a government controlled system. And the price they pay to solve the double spending problem, is exorbitant computer power and energy consumption in the block chain. And such a currency gives no stability whatsoever. We believe that the CB and government are the right party to provide the monetary system and to determine the monetary policy, of course with separated authorities.
The existing monetary system is not designed by engineers but grown by some organic process. We think "it is time to govern the further growth by a good design".

References

Admati and Hellwig (2013), "The Bankers New Clothes", Princeton University Press, 2013

BIS (2018), Central bank digital currencies, BIS, 2018

Bordo and Levin (2017), "Central Bank Digital Currency and the Future of Monetary Policy", Working paper 23711, NBER Working Paper Series, 2017

[18] See the forthcoming paper of Wijngaard and Van Hee (2021)

Buiter (2009), “Negative nominal interest rates; Three ways to overcome the zero lower bound”, *The North American Journal of Economics and Finance*, **20**, 213-238, 2009

Chaum, David (1983). "Blind signatures for untraceable payments". *Advances in Cryptology Proceedings of Crypto*. **82**. pp. 199–203.

Diffie and Hellman (1976). "New directions in cryptography". *IEEE Transactions on Information Theory*. **22** (6): 644

ECB (2020), “Report on a Digital Euro”, ECB, 2020

Gilbert and Handschuh (2003), “Security analysis of SHA-256 and sisters”, *Selected areas in cryptography,* 2003, pp175-193

Ingham, Coutts and Konzelmann (2016), “Introduction: ‘cranks’ and ‘brave heretics’: rethinking money and banking after the Great Financial Crisis”, *Cambridge Journal of Economics*, **40**, 1247-1257, 2016

Jackson and Dyson (2012), “Modernising Money”, Positive Money, 2012

Koblitz, N. (1987). "Elliptic curve cryptosystems". Mathematics of Computation. 48 (177): 203–209. doi:10.2307/2007884. JSTOR 2007884.

Kay (2009), “Narrow Banking; The Reform of Banking Regulation”, Centre for the Study of Financial Innovation, 2009

Rivest, Shamir and Adleman (1978), "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems". *Communications of the ACM*. **21** (2): 120–126

Roubini and Mihm (2010), “Crisis Economics”, Penguin Books

Ryan-Collins, Greenham, Werner and Jackson (2011), “Where does Money come from”, NEF

Satoshi (2008). A Peer-to-Peer Electronic Cash System. Archive www.bitcoin.org

Sveriges Riksbank (2017): The Riksbank’s e-krona project – Report 1, 2017

Taylor (1993). “Discretion versus Policy Rules in Practice.” *Carnegie-Rochester Conference Series on Public Policy*, 39:195-214.

Wijngaard and Van Hee (2021, forthcoming), “Possibilities for rule based monetary policy in a CBDC based monetary system”, https://www.robuustgeld.nl/2021-2/

Appendix 1: Cryptography in a nutshell

We use cryptography for three purposes: fingerprinting, digital signatures and confidential message passing.
For fingerprinting we use *hash functions*. An hash function H makes from a big character string a shorter one. So H applied to a string A gives as shorter string B: H(A)=B. There are well-known examples of hash functions that we used already before computers exist. The *checksum* is such an example. There we add the digits of a large number and use that sum as a hash of the large number. The goal is often to have a shorter representation of the first string, to *identify* the original string. In theory this can go wrong because there exist several long strings that are mapped by H to the same short string. The checksum is a very bad example, since if we switch two digits the hash remains the same. In practice such a collision seldom happens, because the number of big strings that we use is 'neglectable' small compared to the total number of strings on which we could apply the hash function. So hash functions are not *injective* and also not *invertible* such as the key pairs of an asymmetrical encryption system. But it costs very much computing time to find a string X if you only know Y and that H(X)=Y. A frequently used hash function is called SHA-256 (Secret Hash Algorithm 256)[19] which is developed by NSA.

That is why hash functions are used to protect the *authenticity* of data. This is why a hash code of data can be used as *fingerprint*. For instance if one wants to send a message M and one wants to guarantee that the message will not be altered during sending, then one sends M and separately H(M). If the receiver reads L instead of M he can see that the message is manipulated because when he computes H(L) he sees that it differs from H(M). The same system can be applied to files. At the moment a file M is stored the value H(M) is computed and stored separately. As soon as the file M is retrieved one also retrieves H(M) to check if file M has been altered. The probability that such a manipulation will not be detected is neglectable.

The basis of most cryptography techniques is the time consumption of the computation of the *discrete logarithm* of a very large number with 100 to 1000 digits. All computations take place with numbers smaller than a very large number N. This is realized by an operation called *modulo* or *mod* for short, which is defined as: X *mod* N is the remainder of X by division of N, which means subtracting of N from X so many times that a number between 0 and N remains. In this system we can have the classical multiplication mod N, for instance X.Y *mod* N where X.Y

[19] Gilbert and Handschuh (2003)

is the classical multiplication and then then remainder by division by N. And so we can compute the X-power of a number g: g^X *mod* N where g is multiplied by itself X-1 times. Note that it is easy to compute g^X *mod* N even for very large X, since we compute $g.g=g^2$, $g^2.g^2=g^4$,$g^4.g^4=g^8$, etc. (all computations are *mod* N). So g^X *mod* N requires ca ^{2}log X multiplications.
A *discrete logarithm* is the computation of X if Y, g and N are given and $Y=g^X$ mod N. We can denote this solution by $X={}^g\log_N(Y)$, just like the well-known classical logarithm, but the subscript N indicates that we use the *mod* operation. For large numbers these computations are taking many years on a very fast computer and therefore they are considered as practically impossible.
Instead of this classical multiplications with binary numbers there is another computational system, consisting of the set of all points with coordinates (x,y) on an *elliptic curve*. i.e. they satisfy the equation $y^2=x^3+a.x+b$ for some numbers a and b. (cf Koblitz(1987)). On this set there is a special *multiplication operation*, namely if we draw a straight line through two arbitrary points on the curve then there is exactly one other intersection of the curve. If we mirror this point on the x-axis we obtain again a point on the curve and we define that point as the result of the multiplication. The set with this multiplication operation behaves like the number system *mod* N. (It is a mathematical group) Here the same discrete logarithm can be defined and the computation effort of it is orders of magnitude larger than for the binary numbers given above. Elliptic curve cryptography is considered currently as the most safe system.

One of the major tricks, due to Diffie and Hellman (1976) is to create a *secret key* for encryption of messages. All actors share some information: the numbers g and N. Each actor A has a *secret key*, i.e. a large binary number that we also denote with A. Further there is a *public register* where all the *public keys* are registered. The public key of A is g^A.
If A and B want to exchange messages they can look up in the public register each other's public key, i.e. g^A and g^B respectively. Then A computes $(g^B)^A$ and B computes $(g^A)^B$ and these are the same. Nobody else can do this. So A and B have the same secret key without exchanging information! With this secret key they can encrypt and decrypt their messages. Using the same key is called a *symmetric encryption system* and there are many good ones. The problem is that both actors have to agree upon the key, but the Diffie-Helman trick makes it a safe procedure. In case one party starts the communication, say A, he can also choose a random binary number Y and send this to B as g^Y. Then A can compute $(g^B)^Y$ and B can compute $(g^Y)^B$ which is also the same key and it can be a different one for each communication, which is obviously more safe. The only drawback is that B can't verify that A is the sender. Therefore A sends in the first encrypted message $(g^B)^A$ as his *digital signature* to B. Note that the digital signature of A is different for each receiver B.

Another famous encryption system is called *RSA-encryption* after the inventors Rivest, Shamir and Adleman (1978). This is an *asymmetric encryption system* where everybody has a secret and a public key, like in the system above, but these key pairs have an extra property, namely that the secret key can be used to decrypt a message encoded with the public key and vice versa, a message encoded with the public key can be decoded by the secret key. In this way there is no exchange of keys necessary in a communication. The computation barrier to crack the code is also the computation effort of the discrete logarithm. It is also easy to create a digital signature with this system. Suppose A wants to send a message to B and he wants to prove his identity. Then A sends a message with his name and his name also encrypted with his secret key. The result is encrypted with the public key of B. Then B can decode the message with his secret key and he sees the name of A and looks up the public key of A and he decrypts the message further and sees that it is indeed A.
The efficiency of the elliptic curve cryptography with Diffie and Hellman is more efficient.