A Fully Homomorphic Encryption Scheme for Real-Time Safe Control

Stobbe, P.J.; Keijzer, T.; Ferrari, Riccardo M.G.

DOI
10.1109/CDC51059.2022.9993055

Publication date
2022

Document Version
Final published version

Published in
Proceedings of the IEEE 61st Conference on Decision and Control (CDC 2022)

2022 IEEE 61st Conference on Decision and Control (CDC)
December 6-9, 2022. Cancún, Mexico

A Fully Homomorphic Encryption Scheme for Real-Time Safe Control

Pieter Stobbe[1], Twan Keijzer[1] and Riccardo M.G. Ferrari[1]

2022 IEEE 61st Conference on Decision and Control (CDC) | 978-1-6654-6761-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/CDC51059.2022.9993055

***Abstract*— Fully Homomorphic Encryption (FHE) has made it possible to perform addition and multiplication operations on encrypted data. Using FHE in control thus has the advantage that control effort for a plant can be calculated remotely without ever decrypting the exchanged information. FHE in its current form is however not practically applicable for real-time control as its computational load is very high compared to traditional encryption methods. In this paper a reformulation of the Gentry FHE scheme is proposed and applied on an FPGA to solve this problem. It is shown that the resulting FHE scheme can be implemented for real-time stabilization of an inverted double pendulum using discrete time control.**

I. Introduction

Cryptography has allowed for the development of control-systems, such as hydroelectric dams or energy grids at a regional level or higher, that must be securely monitored and controlled over long distances. Such spatially distributed systems require a remote connection between the plant actuators, sensors and the controller which can only be feasibly secured from intrusion via encryption.

Currently, industrial control systems are secured by end-to-end encryption, utilising a mix of symmetric-key and public-key encryption schemes [1], [2], [3]. These methods are successful in securing low rate communication within large scale control systems. However, they are unsuitable for high sampling frequency feedback-control, as they require multiple encryption and decryption steps. These operations introduce time overhead, reducing the stability margin and possibly de-stabilizing the plant. Furthermore the decryption of data at the controller level means that these methods do not provide security if the controller itself is compromised.

Homomorphic encryption (HME) schemes present a solution to these problems. These schemes allow for multiplication and/or addition of encrypted numbers, thus removing the need for decryption and encryption at the controller level. There are two main types of HME: Partially (PHE) and Fully Homomorphic Encryption (FHE). PHE schemes support only multiplication or addition, whereas FHE schemes support both. The first HME scheme was RSA [2], followed by PHE schemes such as EL Gamal [4] and Paillier [5].

More recently lattice-based FHE schemes have been introduced in [6], [7], [8]. For encrypted control this means these schemes allow for implementation of a broad range of feedback control. However, the high computational complexity of these lattice based schemes prevents them from being used in real-time on conventional hardware.

[1]Delft Center for Systems and Control, Delft University of Technology, The Netherlands. pj.stobbe@hotmail.com, {t.keijzer, r.Ferrari}@tudelft.nl.

PHE schemes have also been proposed for control, such as in [9] which proposes a combination of the El Gamal [4] and RSA [2] schemes. This control scheme, however, requires the controller state to be sent to the plant for decryption and re-encryption at each time step, adding additional overhead. More recently, [10] has demonstrated direct feedback control with the PHE scheme from [5]. Due to the limited homomorphic properties of PHE, the controller used un-encrypted controller gains, posing a security risk.

However, recently more attention has been directed to FHE schemes for control, such as in [11], [12], [13], [14], [15]. These schemes however still suffer from two problems. First of all, the representation of encrypted signals requires orders of magnitude more storage than the original plaintext. This means that, due to limited computation and bandwidth resources, real-time control with FHE is limited in complexity and update rate. In [12], a two-state LTI controller is implemented with an update rate of 2 Hz while in [16] a direct feedback controller for high-level control of a drone reaches an update rate of 10 Hz.

Secondly, PHE and FHE only allow for encryption of unsigned integers, whereas control requires the use of real numbers. For PHE and FHE the real numbers can be represented as unsigned integers through the Q format. One limitation that remains is that multiplications will shift the location of the decimal point, eventually leading to overflow. Under normal conditions, the decimal point can be shifted back with right hand bit-shifts. However, no FHE schemes currently support homomorphic right hand bit-shifts without excessive penalties on *multiplicative depth*, which is defined as the maximum allowed number of consecutive multiplications. Lattice based encryption schemes support only a relatively shallow multiplicative depth, after which ciphers can no longer be correctly decrypted. Alternative solutions, such as periodic reset [11] and scaling of the state space matrices [13] have been proposed to solve this problem. These methods however affect stability and performances, limiting applicability.

The problems of computational complexity and fixed precision have hindered the acceptance of FHE for real-time control. In this paper we propose an FHE scheme for real-time secure control implemented on an Field Programmable Gate Array (FPGA) which address this issue. The contributions of the paper are:

- The Gentry's FHE scheme [6] has been reformulated with analytical operations that allows for more intuitive manipulation of the scheme.
- A so-called *reduced cipher* is introduced via a change of representation of the original cipher to reduce the

computational complexity of the FHE scheme.
- The FHE scheme is implemented on an FPGA for real-time control of an unstable plant to demonstrate the benefits of the novel *reduced cipher*.

The resulting FHE scheme can be used in combination with existing control schemes based on additions and multiplications and, while an FPGA was chosen here, can also be implemented on any conventional hardware. Note that the encryption properties of Gentry's scheme [6] are unchanged by using the novel *reduced cipher*.

In the following Section II introduces the considered control setup and Gentry's FHE scheme. In Section III the *reduced cipher* is introduced and its equivalence is proven. Section IV shows results of implementing FHE on an FPGA for control of a inverted double pendulum.

A. Notation

For a positive scalar x, we denote individual digits of its binary representation as $x^{[i]}$. That is, $x = \sum_{i=0}^{\infty} 2^i x^{[i]}$. For any $x \in \mathbb{N}$ we define $(x)^\ell = \sum_{i=0}^{\ell-1} 2^i x^{[i]}$ which are the ℓ least significant binary digits of x, such that if $x \leq q$ where $q = 2^\ell - 1$, then $(x)^\ell = x$ and if $x > q$, then $(x)^\ell \neq x$. We denote $[x]^\ell = [x^{[0]}, \ldots, x^{[\ell]}]$ as a vector whose elements are the binary digits of $(x)^\ell$; $g = [2^0, \ldots, 2^{\ell-1}]^\top$ and the set $\mathbb{Z}_q = \{0, \ldots, q-1\}$, where $q \in \mathbb{N}$. We denote bit-shifts of a x by i bits as $x \ll i = 2^i x$ and $x \gg i = 2^{-i} x$. These concepts can be extended to matrices $X \in \mathbb{N}^{n_1 \times n_2}$, where $(X)^\ell$, $[X]^\ell$, and bitshifts are applied element-wise. $G_n = I_n \otimes g$, while the encrypted version of a variable x is denoted as $\mathbf{E}(x)$. Finally, for a discrete time signal $x(k)$, where k is the current time step, the short-hand notation $x^+ = x(k+1)$ is used.

II. Problem Statement

This section will cover the control scenario in Section II-A, followed by section II-B, which introduces Gentry's FHE scheme using the proposed novel, simplified notation.

A. Control Scenario

In this paper we consider a nonlinear plant of the form

$$\begin{cases} \dot{x} = f(x,u) + \xi, \\ y = h(x,u) + \eta, \end{cases} \tag{1}$$

and a discrete time, dynamical linear controller of the form

$$\begin{cases} \hat{x}^+ = g(\hat{x}, u, y, L), \\ u^+ = v(\hat{x}^+, K), \end{cases} \tag{2}$$

where $x \in \mathbb{R}^\rho$ is the state, $\hat{x} \in \mathbb{R}^\rho$ the state estimate, $u \in \mathbb{R}^\gamma$ the input and $y \in \mathbb{R}^\nu$ the output. $f(\cdot)$ and $h(\cdot)$ are the known state transition and output functions, and ξ and η represent external disturbances or model uncertainty. The controller consists of two parts: $g(\cdot)$ to obtain the next $\hat{x}$; and $v(\cdot)$ to obtain the new control input. In this paper we consider the plant is controlled by an encrypted version of the controller, which using notation from Section I-A is denoted by

$$\begin{cases} \mathbf{E}(\hat{x}^+) = \tilde{g}(\mathbf{E}(\hat{x}), \mathbf{E}(u), \mathbf{E}(y), \mathbf{E}(L)), \\ \mathbf{E}(u^+) = \tilde{v}(\mathbf{E}(\hat{x}^+), \mathbf{E}(K)). \end{cases} \tag{3}$$

The entire encrypted control loop is shown in Figure 1, which will be discussed in more detail in Section 1. To limit the scope of this paper to its focus of encryption, we make two assumptions on the unencrypted control.

Assumption 1: The control law (2) can be constructed with addition, subtraction and multiplication operations only. This holds true for all linear control methods such as PID, state-feedback and LQR control [17].

Assumption 2: The plant in Equation (1) is stabilised by the unencrypted controller (2).

B. Fully Homomorphic Encryption Scheme by Gentry et al.

In this paper, Gentry's FHE scheme [7] is adapted to become more computationally efficient. Gentry's FHE scheme consists of four procedures: Key generation, encryption, homomorphic operations, and decryption. Gentry introduced four functions to perform these procedures. These functions are defined using notation from Section I-A as follows:

Definition 1: For any matrix $a \in \mathbb{N}^{N \times (n+1)}$, $b \in \mathbb{N}^{N \times N}$, and $c \in \mathbb{Z}_q^{n+1 \times 1}$

$$\textbf{BitDecomp}(a) = [a]^\ell \tag{4}$$

$$\textbf{BitDecomp}^{-1}(b) = b \cdot G_{n+1} \tag{5}$$

$$\textbf{Flatten}(b) = [b \cdot G_{n+1}]^\ell \tag{6}$$

$$\textbf{PowersOf2}(c) = c \cdot G_{n+1}^\top \tag{7}$$

We are now ready to define the procedures used in [7].

Key Generation: A public-private key pair is generated as follows: Pick parameters $m \in \mathbb{N}$, $n \in \mathbb{N}$ and $q \in \mathbb{N}$ based on the required security and precision respectively. The private key is $s = [1, -t]^\top$ where $t \in \mathbb{Z}_q^{1 \times n}$ is sampled uniformly on the interval $[0, q-1]$. The public key is $A = [b, B]$ where $b = B \cdot t^\top + e$, each element of $B \in \mathbb{Z}_q^{m \times n}$ is sampled uniformly on the interval $[0, q-1]$, and each element of $e \in \mathbb{Z}_q^m$ is sampled from the χ_q distribution [18].

Encryption: A message $\mu \in \mathbb{Z}_q$ can be encrypted as a cipher $C \in \mathbb{Z}_2^{N \times N}$ via the following relation

$$\begin{aligned} C = &\textbf{Enc}(\mu) = \textbf{Flatten}(\mu \cdot I_N + \textbf{BitDecomp}(R \cdot A)) = \\ &\left[(\mu \cdot I_N + [R \cdot A]^\ell) \cdot G_{n+1}\right]^\ell, \end{aligned} \tag{8}$$

where $N = \ell(n+1)$ depends on the message size through $\ell = \lfloor \log_2(q) \rfloor + 1$ and each element of $R \in \mathbb{Z}_2^{N \times m}$ is sampled uniformly on the interval $[0, 1]$.

Decryption: Ciphers are decrypted using the **MPDec** algorithm as proposed in [19]:

$$\mu = \textbf{MPDec}((C\textbf{PowersOf2}(s))^\ell) \tag{9}$$

The **MPDec** algorithm [19] uses the first ℓ elements of its input to retrieve μ. Proof that the correct message is retrieved in this way can be found in [7].

Homomorphic Operations: The homomorphic operations for ciphers $C_1 = \mathbf{Enc}(\mu_1)$, $C_2 = \mathbf{Enc}(\mu_2)$ and scalar α are

$$\begin{aligned}
\text{Sum: } C_3 =& \mathbf{Flatten}(C_1 + C_2) = \\
& [(C_1 + C_2) \cdot G_{n+1}]^{\ell}, \\
\text{Product: } C_4 =& \mathbf{Flatten}(C_1 \cdot C_2) = \\
& [(C_1 \cdot C_2) \cdot G_{n+1}]^{\ell}, \\
\text{Scalar product: } C_5 =& \mathbf{Flatten}(\mathbf{Flatten}(\alpha I_N) \cdot C_2) = \\
& [([(\alpha I_N) \cdot G_{n+1}]^{\ell} C_2) \cdot G_{n+1}]^{\ell}, \\
\text{Scalar sum: } C_6 =& \mathbf{Flatten}(\alpha I_N + C_2) = \\
& [(\alpha I_N + C_2) \cdot G_{n+1}]^{\ell}.
\end{aligned} \tag{10}$$

For these homomorphic operations it is proven that

$$\begin{aligned}
\mu_3 =& \mu_1 + \mu_2 \quad \Longleftrightarrow \mu_3 = \mathbf{MPDec}(C_3)\,, \\
\mu_4 =& \mu_1 \mu_2 \quad \Longleftrightarrow \mu_4 = \mathbf{MPDec}(C_4)\,, \\
\mu_5 =& \alpha \mu_2 \quad \Longleftrightarrow \mu_5 = \mathbf{MPDec}(C_5)\,, \\
\mu_6 =& \alpha + \mu_2 \quad \Longleftrightarrow \mu_6 = \mathbf{MPDec}(C_6)\,.
\end{aligned} \tag{11}$$

C. FHE in Control

The Gentry FHE scheme [7] has excellent theoretical properties, but there are two obstacles which, until now, have prevented implementation of the scheme in control. Firstly, any message $\mu \in \mathbb{Z}_q$ containing ℓ bits of information, when encrypted, becomes a cipher $C \in \mathbb{Z}_2^{N \times N}$ containing $N^2 = (n+1)^2 \ell^2$ bits of information. Therefore storage and transfer of ciphers requires more memory than unencrypted equivalents. The problem of size becomes even more pronounced when performing homomorphic operations. Direct implementation of homomorphic operations requires multiple steps in which intermediate ciphers can become as large as $\mathbb{Z}_N^{N \times N}$ containing $N^2(\lfloor \log_2(N) \rfloor + 1)$ bits of information.

Even more important than the strain on storage and communication, is the strain on the computational resources. For direct implementation of homomorphic addition, $N^2(n+2) - N(n+1)$ addition operations and $N^2(n+1)$ multiplication operations are needed, whereas its unencrypted equivalent requires only a single addition. In this paper a so-called *reduced cipher* is introduced to reduce the computational load of FHE, allowing for faster update rates of control laws.

The second obstacle is the representation of real numbers with unsigned integers. To this end we employ the commonly used fixed precision representation called Q format [10][1]. Alternatives using floating point numbers are currently being researched [20] but are not yet sufficiently mature. Q format allows for representing a fixed accuracy number β with an integer message $\mu \in \mathbb{Z}_p$ where $\lfloor \log_2(p) \rfloor + 1 = m_q + n_q$ as

$$\begin{aligned}
\beta &= -2^{m_q - 1} \mu^{[m_q + n_q - 1]} + \sum_{i=0}^{m_q + n_q - 2} 2^{i - n_q} \mu^{[i]} \\
\mu &= \begin{cases} 2^{n_q} \beta & \text{if } \beta \geq 0 \\ -2^{m_q + n_q} + |\beta| 2^{n_q} & \text{if } \beta < 0 \end{cases}
\end{aligned} \tag{12}$$

such that β can be any value in $[-2^{m_q - 1}, 2^{m_q - 1})$ rounded to the nearest 2^{-n_q}. When performing multiplication of two messages $\mu_3 = \mu_1 \cdot \mu_2$, where μ_1 and μ_2 are obtained from Equation (12), the result has to fit a $m_q + 2n_q$ sized register to yield an exact result. The available storage for each message is limited and so after a certain number of consecutive multiplications overflow would occur.

Therefore, conventionally, a right-bitshift by n_q bits is performed after each multiplication such that the $m_q + n_q$ least significant bits of μ_3 can be used to retrieve $\beta_1 \beta_2$[2]. However, no HME scheme supports such operation on ciphers without penalty on multiplicative depth. Thus, consecutive multiplications have formed a great obstacle in HME. This problem is important for controllers, which often have internal states that are updated at each timestep without being decrypted. Until now this obstacle has been dealt with using a periodic reset [10] or by transforming the state space variables [13]. These methods, however, affect the stability and performance of the controller such that direct implementation of FHE with existing control schemes is not possible.

III. Reduced Ciphers for Fast FHE Implementation

In this section the so-called *reduced cipher* will be presented for computationally efficient implementation of FHE for discrete control. It will be shown that, with the *reduced cipher*, encryption, homomorphic operations, and decryption can be made orders of magnitude more computationally efficient, enabling real-time implementation of FHE for control.

Given a cipher $C \in \mathbb{Z}_2^{N \times N}$, the so-called *reduced cipher* will be denoted as $\tilde{C} \in \mathbb{Z}_q^{N \times (n+1)}$ and is defined as

$$\tilde{C} = \mathbf{BitDecomp}^{-1}(C) = C G_{n+1}\,.$$

Here Definition 1 is used to rewrite the relation between cipher and *reduced cipher*. Note that the reduced *cipher contains* exactly the same information as the original cipher. In Theorem 1 it will be shown that using the *reduced cipher* reduces the total number of computer operations needed, and completely eliminates the need for doing hardware multiplications when performing homomorphic multiplication.

Lemma 1: For any matrix $\Lambda \in \mathbb{N}^{n_1 \times n_2}$, we have $[\Lambda]^{\ell} G_{n_2} = (\Lambda)^{\ell}$.

Proof: First consider $\alpha \in \mathbb{N}$. for any α it holds

$$(\alpha)^{\ell} = \sum_{i=0}^{\ell - 1} 2^i \alpha^{[i]} = \left[\alpha^{[0]}, \ldots, \alpha^{[\ell - 1]}\right] \cdot g = [\alpha]^{\ell} \cdot g\,.$$

Then apply this relation on each element of Λ, giving $(\Lambda)^{\ell} = [\Lambda]^{\ell} \cdot I_{n_2} \otimes g = [\Lambda]^{\ell} \cdot G_{n_2}$ ∎

Theorem 1: Given ciphers $C_1, C_2 \in \mathbb{Z}_2^{N \times N}$ and scalar $\alpha \in \mathbb{Z}^q$ the existing homomorphic operations can equiva-

[1]We will be using the Q-notation as introduced by Texas-Instruments, which is used in code libraries such as the TMS320C64x+ IQmath.

[2]rounded down to the nearest 2^{-n_q}, due to truncation during the right hand bitshift.

lently be written using the *reduced cipher* as

$$C_3 = [(C_1 + C_2)G_{n+1}]^\ell \leftrightarrow \tilde{C}_3 = \left(\tilde{C}_1 + \tilde{C}_2\right)^\ell \quad (13)$$

$$C_4 = [(C_1 \cdot C_2)G_{n+1}]^\ell \leftrightarrow \tilde{C}_4 = \left(C_1 \cdot \tilde{C}_2\right)^\ell \quad (14)$$

$$C_5 = [[\alpha G_{n+1}]^\ell \cdot C_1 G_{n+1}]^\ell \leftrightarrow \tilde{C}_5 = ([\alpha G_{n+1}]^\ell \tilde{C}_1)^\ell \quad (15)$$

$$C_6 = [(\alpha I_N + C_1)G_{n+1}]^\ell \leftrightarrow \tilde{C}_6 = (\alpha G_{n+1} + \tilde{C}_1)^\ell \quad (16)$$

Proof: Each equivalence is proven separately below.

$$\begin{aligned}
\tilde{C}_3 &= [(C_1 + C_2)G_{n+1}]^\ell G_{n+1} \\
&= (C_1 G_{n+1} + C_2 G_{n+1})^\ell = \left(\tilde{C}_1 + \tilde{C}_2\right)^\ell \\
\tilde{C}_4 &= [(C_1 \cdot C_2)G_{n+1}]^\ell G_{n+1} = \left(C_1 \cdot \tilde{C}_2\right)^\ell \\
\tilde{C}_5 &= \left[[\alpha I_N G_{n+1}]^\ell \cdot C_1 G_{n+1}\right]^\ell G_{n+1} \\
&= \left([\alpha G_{n+1}]^\ell \cdot \tilde{C}_1\right)^\ell \\
\tilde{C}_6 &= [(\alpha I_N + C_1)G_{n+1}]^\ell G_{n+1} = \\
&\quad (\alpha G_{n+1} + C_1 G_{n+1})^\ell = \left(\alpha G_{n+1} + \tilde{C}_1\right)^\ell
\end{aligned}$$

Here Definition 1 and Lemma 1 were used. ∎

Theorem 1 has shown equivalences between homomorphic operations on ciphers and on *reduced ciphers*. In the following corollaries it will be shown how these equivalences are used in encryption and decryption for the FHE scheme.

Corollary 1: The term αG_{n+1} from Theorem 1 can be generated using only bitshifts. Due to the structure of αG_{n+1} the number of operations needed to obtain $\tilde{C}_5$ and $\tilde{C}_6$ can be reduced to respectively $\mathcal{O}(n^2\ell^2)$ and $\mathcal{O}(n\ell)$.

Proof: Denote $\alpha G_{n+1} = I_{n+1} \otimes \alpha g$. Note that αG_{n+1} contains $(n+1)$ instances of αg, such that only $N = (n+1)\ell$ entries are non-zero. Therefore, by skipping the structural zeros, we require only $\mathcal{O}(N^2)$, $\mathcal{O}(N)$ operations respectively to obtain $\tilde{C}_5$ and $\tilde{C}_6$. Furthermore, αg can be generated using $\mathcal{O}(\ell)$ bitshifts as $\alpha g = [\alpha, \alpha \ll 1, \ldots, \alpha \ll \ell - 1]$. ∎

Corollary 2: Encryption and decryption can be rewritten in terms of *reduced ciphers* using theorem 1.

Proof: Encryption is performed using Equation (8), where $\textbf{BitDecomp}(RA) = [RA]^\ell \in \mathbb{Z}_2^{N \times N}$ is of the same form as a cipher. Encryption is thus a special case of the homomorphic scalar sum as defined in equation (11). Applying Equation (16) and Lemma 1 to encryption yields

$$\tilde{C} = (\mu G_{n+1} + [R \cdot A]^\ell G_{n+1})^\ell = (\mu G_{n+1} + R \cdot A)^\ell \quad (17)$$

Decryption can be rewritten using the novel notation as $\mu = \textbf{MPDec}((CG_{n+1}s)^\ell) = \textbf{MPDec}((\tilde{C}s)^\ell)$ ∎

The actual reduction of computational complexity obtained by using the *reduced ciphers* for the homomorphic operations is summarized in Table I. The table shows the computational complexity and memory utilisation of the operations that are involved in evaluating the homomorphic operations from equation (10), both with an without the use of *reduced ciphers*. The number of operations is reduced and homomorphic multiplication no longer requires multiplication of cipher elements. Furthermore, Table I shows the reduction in required memory for performing the operations without requiring intermediate reading and writing of memory. Note however, that the reduced cipher only increases computational efficiency, but does not affect the encryption properties of Gentry's scheme. Furthermore, *Reduced ciphers* contain the same amount of data as regular ciphers and so the communicational bandwidth required to transfer the ciphers is unchanged.

TABLE I

NUMBER OF OPERATIONS REQUIRED FOR HOMOMORPHIC OPERATIONS.

	sum		product	
	Cipher	Red. Cipher	Cipher	Red. Cipher
Bit Operation	0	0	$\mathcal{O}(n^3\ell^3)$	$\mathcal{O}(n^3\ell^2)$
Addition	$\mathcal{O}(n^3\ell^2)$	$\mathcal{O}(n^2\ell)$	$\mathcal{O}(n^3\ell^3)$	$\mathcal{O}(n^3\ell^2)$
Multiplication	$\mathcal{O}(n^3\ell^2)$	0	$\mathcal{O}(n^3\ell^2)$	0
Memory	$\mathcal{O}(n^2\ell^2)$	$\mathcal{O}(n^2\ell^2)$	$\mathcal{O}(n^2\ell^2\log(n\ell))$	$\mathcal{O}(n^2\ell^2)$
	scalar sum		scalar product	
Bit Operation	$\mathcal{O}(n^3\ell^2)$	$\mathcal{O}(\ell)$	$\mathcal{O}(n^2\ell^3)$	$\mathcal{O}(n^2\ell^2)$
Addition	$\mathcal{O}(n^3\ell^2)$	$\mathcal{O}(n\ell)$	$\mathcal{O}(n^3\ell^3)$	$\mathcal{O}(n^2\ell^2)$
Multiplication	$\mathcal{O}(n^2\ell)$	0	$\mathcal{O}(n^3\ell^2)$	0
Memory	$\mathcal{O}(n^2\ell^2)$	$\mathcal{O}(n^2\ell^2)$	$\mathcal{O}(n^2\ell^2\log(\ell))$	$\mathcal{O}(n^2\ell^2)$

IV. RESULTS ON A SIMULATED PLANT

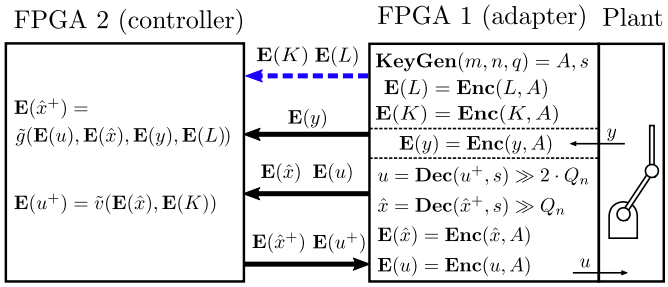

Fig. 1. The experimental setup where FPGA 1 performs encryption and decryption at the plant and FPGA 2 contains the remote controller. The key exchange, as indicated with the dashed blue arrow, is only required at initialization.

In this section we will apply the novel FHE scheme to the control of an inverted double pendulum. To achieve a realistic setup, the encrypted control is implemented on two FPGAs, as shown in Figure 1. It will be shown that it is possible to perform stabilising control of the unstable plant in real-time with the encrypted controller. Below, first the properties of an FPGA and the used setup will be discussed. Next the double pendulum model and control law will be introduced. Lastly, the obtained results will be shown.

A. Hardware Resources of an FPGA

An FPGA contains generic logic cells and memory components of differing sizes and configurability. The most common type of logic cell is called Adaptive Logic Modules (ALM), these can be configured to perform any operation. Though ALM's can be configured to perform multiplication, this would be very inefficient and so FPGA's are generally equipped with Digital Signal Processing (DSP) slices which are specifically made to perform multiplication. Unfortunately, due to the die space requirements, there are fewer

available. To illustrate, on any particular FPGA, ALM's are usually available in the order of tens of thousands, whereas there are usually only DSP's available in the order of tens. If a design's speed relies on multiplication, the limited number of DSP-slices could bottle-neck the computational speed.

The *reduced cipher* implementation as presented in Section III reduces the computational load of the scheme on any platform, however one aspect is particularly beneficial to FPGA design. As shown in Table I, the total number of operations is reduced by an order of magnitude when using the *reduced cipher*. More importantly however, is that all multiplication operations are replaced by bit-operations and additions. Replacing all multiplications with bit-operations ensures the FPGA design will not be bottle-necked by the availability of DSP-slices.

B. Experimental Setup

The encrypted control scheme has been implemented in VHDL for use on two Nexys 4 FPGA's in the configuration as shown in Figure 1. The results are obtained from a hardware simulation of the FPGA coupled with a high resolution simulation of the double pendulum. In the following first the choice for FPGA's as hardware platform is argued. Then, the simulated plant and the corresponding controller is described.

FPGA's can be programmed to operate without the need for a software layer and so is the platform chosen for implementation. Furthermore, it can be seen from table I that multiplication operations, which are the most computationally expensive on an FPGA, are completely eliminated by using the *reduced cipher*. With this implementation on an FPGA a new control input can be generated every $0.8\ ms$, which would not have been possible using the original ciphers or on conventional hardware.

The chosen plant is the inverted double pendulum depicted in Figure 2. The dynamics of the double pendulum's state $\theta = [\theta_1\ \theta_2]^\top$ is modeled as

$$\begin{aligned}
&\begin{cases} M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + G(\theta) = T\,, \\ T + \tau_e \dot{T} = k_m u \end{cases} \\
&M(\theta) = \begin{bmatrix} P_1 + P_2 + 2P_3\cos\theta_2 & P_2 + P_3\cos\theta_2 \\ P_2 + P_3\cos\theta_2 & P_2 \end{bmatrix} \\
&C(\theta,\dot{\theta}) = \begin{bmatrix} b_1 - P_3\dot{\theta}_2\sin\theta_2 & -P_3(\dot{\theta}_1+\dot{\theta}_2)\sin\theta_2 \\ P_3\dot{\theta}_1\sin\theta_2 & b_2 \end{bmatrix} \\
&G(\theta) = \begin{bmatrix} -g_1\sin\theta_1 - g_2\sin(\theta_1+\theta_2) \\ -g_2\sin(\theta_1+\theta_2) \end{bmatrix} \\
&P_1 = m_1c_1^2 + m_2l_1^2 + I_1, \quad P_2 = m_2c_2^2 + I_2 \\
&P_3 = m_2l_1c_2, \quad g_1 = (m_1c_1 + m_2l_1)g, \quad g_2 = m_2c_2g
\end{aligned} \tag{18}$$

where θ_1 and θ_2 denote the angles of the pendulum links as shown in Figure 2. The system has the same form as (1). Furthermore, m_1,m_2 are the masses of the links; l_1, l_2 are their lengths; c_1, c_2 are the centers of mass; I_1, I_2 are the mass moments of inertia; b_1,b_2 are the damping coefficients of the joints; k_m,τ_e are the electrical motor gain and time constant, and g is the gravitational acceleration.

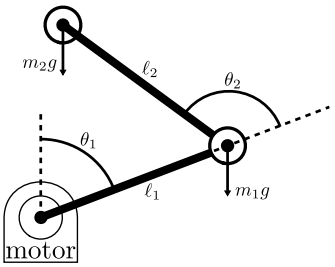

Fig. 2. Double pendulum as modeled by Equation (18).

The double pendulum is initialized at an initial state $\theta = \theta_0^\top = [0.0289, 0.1156]^\top$, $\dot{\theta} = \dot{\theta_0}^\top = [0.0669, 0.0049]^\top$ and $T = T_0 = 0$, and is controlled such that both pendulums point upwards, i.e. $\theta = [0\ 0]^\top$. A discrete time linearization of Model (18) can be made around $\theta = \dot{\theta} = [0\ 0]^\top$ as

$$\begin{cases} x(k+1) = A_d x(k) + B_d u(k)\,, \\ y(k) = C_d x(k)\,, \end{cases} \tag{19}$$

where $x(k) = [\theta_1(k)\ \dot{\theta}_1(k)\ \theta_2(k)\ \dot{\theta}_2(k)\ T]^\top$, $y(k) = [\theta_1(k)\ \theta_2(k)]^\top$, and A_d, B_d, and C_d are matrices of appropriate size. This linearized model is used to implement an observer and state feedback controller as

$$\begin{cases} \hat{x}(k+1) = A_d\hat{x}(k) + B_d u(k) + L(y(k) - C_d\hat{x}(k))\,, \\ u(k+1) = K\hat{x}(k+1)\,, \end{cases} \tag{20}$$

where L is the observer gain and K is the state feedback gain. The controller takes the same form as (2). The controller is updated at a rate of $f = 100\ Hz$. L has been obtained by placing the observer poles at $[0.7\ 0.5\ 0.8\ 0.6\ 0.85]$ and $K = [-12.6\ \ -1.8\ \ -9.8\ \ -0.95\ \ 0.015]$. The input and the state estimate are initialized at $u(1) = 0$ and $\hat{x}(0) = 0$. The controller (20) is encrypted to obtain the equivalent controller $\tilde{g}(\cdot)$, $\tilde{v}(\cdot)$ of the form (3). The model and encryption parameters used can be found in Table II.

TABLE II
MODEL AND ENCRYPTION PARAMETERS

Parameter	Value	Parameter	Value
m_1	$0.125\ kg$	m_2	$0.05\ kg$
l_1	$0.1\ m$	l_2	$0.1\ m$
c_1	$-0.04\ m$	c_2	$0.06\ m$
I_1	$0.074\ kgm^2$	I_2	$0.00012\ kgm^2$
b_1	$4.8\ kgs^{-1}$	b_2	$0.0002\ kgs^{-1}$
k_m	$50\ Nm$	τ_e	$0.03\ s$
g	$9.81ms^{-2}$	n	7
ℓ	64	m	7
m_q	10	n_q	22
f	$100\ Hz$		

The final control loop as shown in Figure 1 works as follows: At boot-up FPGA 1, the adapter, generates an encryption key pair. FPGA 1 then encrypts the state space matrices and sends them to FPGA 2, the controller. Next the control loop starts. First, the adapter encrypts measurement vector y and sends it to the controller (3) which computes $\mathbf{E}(u^+)$ and $\mathbf{E}(\hat{x}^+)$. To extend multiplicative depth and to prevent overflow, these signals are then sent to the adapter for decryption. This is a solution similar to that of [9]. The control effort is applied to the plant, after which u^+ and

$\hat{x}^+$ are bit-shifted, encrypted and sent back to the controller along with the new measurements $\mathbf{E}(y)$.

C. Performance

Figure 3 shows the results of system (18) being controlled according to $\tilde{g}(\cdot), \tilde{v}(\cdot)$. The encrypted observer estimates the states correctly and the plant is stabilized by the encrypted controller. Controlling the plant without encryption, i.e. according to (20), yields identical results. This illustrates that the encrypted controller and unencrypted controller are indeed equivalent.

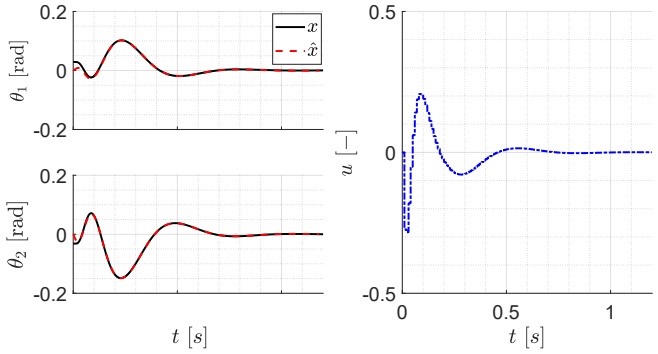

Fig. 3. Simulation results, θ_1, θ_2 and control effort u

The experimental setup serves to highlight the contributions made to FHE. One can see that the plant is controlled towards an unstable equilibrium which requires a fast update rate of the encrypted controller. Due to the use of the *reduced cipher*, this has become possible on the chosen hardware (Nexys 4 FPGA).

V. Conclusion

The use of large scale systems such as hydroelectric dams or energy grids, has led to the need for secure monitoring and control over large distances. Securing such control systems from cyber-attacks is important to the safe operation. One of the ways to achieve this is through encryption.

Using traditional encryption schemes, only the communication links can be secured, but signals have to be decrypted at the controller to calculate the control action. Fully Homomoprhic Encryption (FHE) has been developed such that operations can be performed on encrypted signals. Therefore, it has the potential to close the loop of encryption for secure control. The main obstacle to widespread implementation of FHE in control is the high computational complexity. In this paper, the so-called *reduced cipher* has been introduced, which allows for reducing the the computational complexity significantly. Specifically, the total number of operations performed is reduced by an order of magnitude. The *reduced cipher* and analytical description of the encryption scheme are meant to enable more intuitive implementation and manipulation of the Gentry scheme for control purposes and extension of the capabilities of the scheme.

The presented FHE scheme is the first, to the best of the authors knowledge, that has been implemented for real-time control of an unstable plant. In future work we would like to extend the principle to more complex plants to show the full capability of the scheme. Furthermore, we will explore how to perform right hand bit shifts and other operations on encrypted data. This would be an elegant solution to the problem of shifting decimal points when multiplying fixed precision numbers and could enable to implement more complex control techniques.

References

[1] M. Dworkin, E. Barker, J. Nechvatal, J. Foti, L. Bassham, E. Roback, and J. Dray, "Advanced encryption standard (aes)," 2001-11-26 2001.

[2] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, p. 120–126, 1978.

[3] R. Smith, "Cryptography concepts and effects on control system communications," 2018.

[4] T. El Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Advances in Cryptology*, G. R. Blakley and D. Chaum, Eds. Berlin, Heidelberg: Springer, 1985, pp. 10–18.

[5] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," *EUROCRYPT*, p. 223–238, 1999.

[6] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Stanford University, 2009.

[7] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," *Advances in Cryptology*, p. 75–92, 2013.

[8] J. H. Cheon and D. Stehlé, "Fully homomophic encryption over the integers revisited," in *Advances in Cryptology – EUROCRYPT 2015*, E. Oswald and M. Fischlin, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 513–536.

[9] K. Kogiso and T. Fujita, "Cyber-security enhancement of networked control systems using homomorphic encryption," in *CDC*, 12 2015.

[10] J. Tran, F. Farokhi, M. Cantoni, and I. Shames, "Implementing homomorphic encryption based secure feedback control," *Control Engineering Practice*, vol. 97, p. 104350, 2020.

[11] C. Murguia, F. Farokhi, and I. Shames, "Secure and private implementation of dynamic controllers using semi-homomorphic encryption," 2019.

[12] J. Kim, C. Lee, H. Shim, J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Encrypting controller using fully homomorphic encryption for security of cyber-physical systems," *IFAC-PapersOnLine*, vol. 49, no. 22, pp. 175–180, 2016, 6th IFAC Workshop on Distributed Estimation and Control in Networked Systems NECSYS 2016.

[13] J. Kim, H. Shim, and K. Han, "Dynamic controller that operates over homomorphically encrypted data for infinite time horizon," 2019.

[14] J. Kim, H. Shim, H. Sandberg, and K. H. Johansson, "Method for Running Dynamic Systems over Encrypted Data for Infinite Time Horizon without Bootstrapping and Re-encryption," in *60th IEEE Conference on Decision and Control*, 2021, pp. 5614–5619.

[15] M. P. Chaher, B. Jayawardhana, and J. Kim, "Homomorphic Encryption-Enabled Distance-Based Distributed Formation Control with Distance Mismatch Estimators," in *60th IEEE Conference on Decision and Control*, 2021, pp. 4915–4922.

[16] J. Cheon, K. Han, S.-M. Hong, H. Kim, J. Kim, S. Kim, H. Seo, H. Shim, and Y. Song, "Toward a secure drone system: Flying with real-time homomorphic authenticated encryption," *IEEE Access*, vol. 6, pp. 24 325–24 339, 2018.

[17] K. J. Åström and R. M. Murray, *Feedback systems: An introduction for scientists and Engineers*. Princeton University Press, 2021.

[18] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *J. ACM*, vol. 56, no. 6, pp. 34:1–34:40, 2009.

[19] D. Micciancio and C. Peikert, "Trapdoors for lattices: Simpler, tighter, faster, smaller," in *Advances in Cryptology – EUROCRYPT 2012*, D. Pointcheval and T. Johansson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 700–718.

[20] S. Moon and Y. Lee, "An efficient encrypted floating-point representation using HEAAN and TFHE," *Security and Communication Networks*, vol. 2020, pp. 1–18, 03 2020.