Hindawi
International Journal of Reconfigurable Computing
Volume 2019, Article ID 1949121, 14 pages
https://doi.org/10.1155/2019/1949121

Research Article

Dimension Reduction Using Quantum Wavelet Transform on a High-Performance Reconfigurable Computer

Naveed Mahmud and Esam El-Araby

Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS 66045, USA

Correspondence should be addressed to Naveed Mahmud; naveed_923@ku.edu

Received 4 May 2019; Revised 16 August 2019; Accepted 1 September 2019; Published 11 November 2019

Academic Editor: Wim Vanderbauwhede

The high resolution of multidimensional space-time measurements and enormity of data readout counts in applications such as particle tracking in high-energy physics (HEP) is becoming nowadays a major challenge. In this work, we propose combining dimension reduction techniques with quantum information processing for application in domains that generate large volumes of data such as HEP. More specifically, we propose using quantum wavelet transform (QWT) to reduce the dimensionality of high spatial resolution data. The quantum wavelet transform takes advantage of the principles of quantum mechanics to achieve reductions in computation time while processing exponentially larger amount of information. We develop simpler and optimized emulation architectures than what has been previously reported, to perform quantum wavelet transform on high-resolution data. We also implement the inverse quantum wavelet transform (IQWT) to accurately reconstruct the data without any losses. The algorithms are prototyped on an FPGA-based quantum emulator that supports double-precision floating-point computations. Experimental work has been performed using high-resolution image data on a state-of-the-art multinode high-performance reconfigurable computer. The experimental results show that the proposed concepts represent a feasible approach to reducing dimensionality of high spatial resolution data generated by applications such as particle tracking in high-energy physics.

1. Introduction

High-energy physics deal with advanced instruments such as particle accelerators and detectors. These machines use electromagnetic fields to accelerate charged particles to high speeds and create collisions. By studying particle collisions and tracking collision trajectories, physicists can test the predictions of many theories of particle physics such as properties of the Higgs boson [1], discovering new particle families [2] as well as many high-energy physics problems [3]. There are a number of high-energy physics (HEP) research centers [4]. The largest particle accelerator is the Large Hadron Collider (LHC) in Geneva, Switzerland. Large-scale general-purpose particle detectors have been developed at the LHC. The ATLAS [5] and Compact Muon Solenoid (CMS) [6] are two examples which are used for studying the properties of the Higgs boson and investigating new physics. The ATLAS has an inner detector that has been used to observe the decay products of collisions. The pixel detector [7] is one of the main components of the inner detector, having over 80 million readout channels [8] (pixels), which contribute to half the total readout channels of the entire experiment. Reconstruction of high-energy particles from the pixel detector is considered a critical design and engineering challenge [9], due to its large readout count, high spatial resolution, and 3D space-time measurements. There have been efforts to improve the tracking performance of the ATLAS Inner Detector [9, 10], which involved insertion of additional pixel detector layer (Insertable B-Layer). Another approach that has been considered in the ATLAS FTK (Fast Track Trigger) upgrade [11] is using variable resolution patterns, where the data from the detector is compared to generated pattern banks of particle tracks and non-intersecting data is filtered. In high-dimensional datasets, e.g., the pixel detector readout data, not all the measured data variables are relevant in understanding the underlying regions of interest (RoI). Generally, statistical predictive models are applied to multidimensional datasets for detection and pattern matching, which is a computationally expensive process. Thus, an effective method is needed to reduce the dimensionality [12] of the data in such high-dimensional spatial sets, for faster detection and matching.

As a feasible solution to this problem, we here propose combining wavelet-based dimension reduction techniques [13–15] with quantum information processing (QIP) [16] for applications in domains that generate high-dimensional data volumes such as high-energy physics (HEP). More specifically, we propose using quantum wavelet transform (QWT) to reduce the dimensionality and high spatial resolution of data in HEP particle tracking. Wavelet-based dimension reduction has been shown to be an effective technique in image preprocessing, reducing computation time, reducing interprocessor overhead, and improving classification accuracy [13–15]. Even so, the large volume of data from domains such as high-energy particle physics, present a challenge for a classical wavelet-based method. The QWT has been demonstrated in previous works to be very useful in quantum image processing and quantum data compression [16–19]. Quantum information processing uses qubits as the basic units of information storage, compared to classical binary forms, and can exploit quantum mechanical properties such as entanglement and superposition [20]. Therefore, applying QIP techniques such as QWT for dimension reduction of HEP data will bring substantial improvements in storage and computation compared to classical signal processing techniques. To the best of our knowledge, this work is the first to investigate QWT-based dimension reduction for HEP applications. We develop simple and effective algorithms for QWT and inverse-QWT (IQWT) that are best suited for dimension reduction and present corresponding emulation hardware architectures for QWT and IQWT.

The objectives and focus of our work are to demonstrate the feasibility of QWT for dimension reduction, through emulation, and to evaluate the performance of the emulation architectures. Our proposed algorithms are prototyped on an FPGA-based quantum emulator that has been developed based on our previous works [21, 22] and has been shown to emulate full quantum algorithms such as quantum Fourier transform (QFT) [23] and Grover's search algorithm [24]. An FPGA platform was chosen because of its reconfigurability and flexibility in emulating multiple quantum algorithms. The emulator is based on the hardware system of DirectStream [25], which is a state-of-the-art reconfigurable computing platform. This emulation platform can be conveniently used to verify and benchmark future implementations of the proposed system in HEP applications. In the next section, we discuss fundamental concepts of quantum computing, QWT, and the related work done on QWT. In Section 3, we elaborate our proposed methods and emulation architectures. In Section 4, the experimental results and analysis are presented. Section 5 is our conclusion and future directions of this work.

2. Background and Related Work

In this section, we discuss background concepts of quantum computing and the quantum wavelet transform. We also discuss current and related work on QWT and high-energy particle detection.

2.1. Qubits, Superposition, and Entanglement. The qubit is the smallest unit of quantum information that describes a two-level quantum mechanical system. Physical implementations of the qubit can be electron/atomic/nuclear spin, where spin directions of the particle represent the two qubit levels. Other physical representations of the qubit can be photon polarization, superconducting Josephson junction, etc. [26]. The qubit is represented theoretically using the Bloch sphere [20], as shown in Figure 1. The basis states of the qubit, $|0\rangle$ and $|1\rangle$ are denoted by poles of the sphere. The property that distinguishes the qubit from the classical bit is superposition. The qubit can exist in a mixed or superposition state that is any other point on the surface of the sphere other than the poles. The overall state of the qubit can be defined using a linear superposition equation $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β are complex numbers determined from φ and θ as shown in Figure 1 and satisfying $|\alpha|^2 + |\beta|^2 = 1$. Another distinguishing property of qubits is entanglement [20]. Two or more qubits can be entangled together, which means each entangled qubit becomes strongly correlated to the other along all possible combinations of the qubits. Outcome of measurement of one qubit is dependent on the other measurement, but individually they exhibit completely random behavior. In quantum computing, most algorithms assume that the qubits are fully entangled [21]. A system of n entangled qubits can be represented in vector space as $N = 2^n$ complex basis state coefficients.

2.2. Quantum Wavelet Transform. The wavelet transform, similar to other transforms like Fourier transform, decomposes input signals into their components. The principal difference is that Fourier transform decomposes input signals into their sinusoidal orthogonal temporal-only bases, while wavelet transform uses a set of non-sinusoidal functions, usually called mother wavelets, that are both spatially and temporally localized [15]. This results in a very important feature unique to wavelet transform which is the preservation of spatial locality of data. In other words, wavelet transform gives information about both time and frequency of input data. Wavelet transform also has better computation speeds compared to other transforms [14]. Therefore, they are effective and widely used in many image processing applications [16]. The wavelet transform can be effectively implemented in the quantum information processing (QIP) domain as quantum wavelet transform (QWT) [16, 18, 19]. However, the related work on QWT is rare or preliminary. This is because quantum computing and QIP are fields that are gradually developing and have not yet reached full potential. Although many large-scale quantum hardware is being developed [27], their useful applications are still yet to be decided. We discuss the classical wavelet transform first and then apply it in QIP domain, to establish a model for the QWT. The general wavelet transform can be expressed by

$$F(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} f(t)\Psi_{a,b}^{*}\left(\frac{t-b}{a}\right)\mathrm{d}t, \tag{1}$$

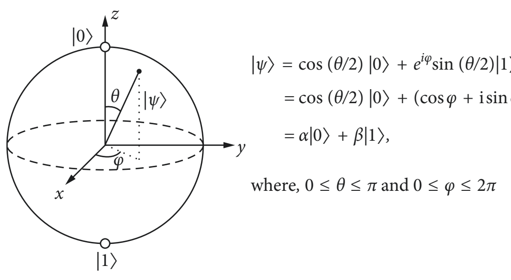

z
$|0\rangle$
θ
$|\psi\rangle$
y
x
φ
$|1\rangle$
$|\psi\rangle = \cos(\theta/2)\,|0\rangle + e^{i\varphi}\sin(\theta/2)|1\rangle$
$= \cos(\theta/2)\,|0\rangle + (\cos\varphi + i\sin\varphi)\sin(\theta/2)|1\rangle$
$= \alpha|0\rangle + \beta|1\rangle,$
where, $0 \le \theta \le \pi$ and $0 \le \varphi \le 2\pi$

Figure 1: Bloch sphere representation of a single qubit.

where Ψ is called the mother wavelet function in complex conjugate form, and a, b are the time dilation and displacement factors, respectively. Wavelet transforms can be classified as *discrete* or *continuous* depending on the use of orthogonal or non-orthogonal wavelets, respectively. For the purposes of this paper, we will discuss the discrete wavelet transform (DWT). The DWT is a decomposition of input signals into a set of wavelet functions that are orthogonal to its translations and scale. The first and simplest DWT was introduced by mathematician Alfred Haar [15] and is thus named the Haar wavelet transform. The Haar mother wavelet function can be constructed using a unit step function, $u(t)$, as shown in (2). The discretized version of the Haar wavelet function is defined as (3), where $t = q \cdot \Delta t$, $b = j \cdot \Delta t$, and $a = K \cdot \Delta t$, Δt is the sampling period, and K is the Haar window size in samples. Applying (3) in (1), the expression for the discrete Haar wavelet transform can be derived to be (4):

$$\Psi\left(\frac{t-b}{a}\right) = u\left(\frac{t-b}{a}\right) - 2u\left(\frac{t-b}{a} - \frac{1}{2}\right) + u\left(\frac{t-b}{a} - 1\right), \tag{2}$$

$$\Psi_D\left(\frac{q-j}{K}\right) = \begin{cases} +1, & 0 \le (q-j) < \dfrac{K}{2}, \\ -1, & \dfrac{K}{2} \le (q-j) < K, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

$$F_D(j,K) = \sum_{q=0}^{N-1} f_D(q \cdot \Delta t)\Psi_D\left(\frac{q-j}{K}\right), \tag{4}$$

where N is the number of data samples. When doing computation in the quantum domain, there are efficient methods of classical-to-quantum encoding [28–30]. Classical signal samples can be encoded as the coefficients of a quantum state, which is in superposition of its constituent basis states [28, 31]. The signal samples are transformed to a normalized sequence of amplitudes as shown in (5), where n is the number of qubits, $N = 2^n$ is the number of basis states of the quantum system, and $|\psi\rangle$ is the input quantum state. By applying the wavelet transform on the input quantum state, we can formulate the equivalent expression for the quantum Haar wavelet transform (QHT) as (6), where $|\psi\rangle_{\text{QHT}}$ is the output quantum state:

$$|\psi\rangle = \sum_{q=0}^{N-1} f(q \cdot \Delta t)|q\rangle, \text{ where } \sum_{q=0}^{N-1} |f(q \cdot \Delta t)|^2 = 1, \tag{5}$$

$$|\psi\rangle_{\text{QHT}} = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1}\sum_{q=0}^{N-1} f(q \cdot \Delta t)\Psi_D\left(\frac{q-j}{K}\right)|j\rangle. \tag{6}$$

There are many notable works on wavelets and applications of wavelet transforms [32–34]. We focused our survey on works of wavelet transform applied in the field of quantum information processing, i.e., quantum wavelet transforms (QWT). Early work on the QWT was reported in [16], where the authors present gate-level circuits for the quantum Haar wavelet and Daubechies $D^{(4)}$ wavelet. They propose techniques for efficient quantum implementation of permutation matrices, which are required for factorization of the unitary operations of the wavelet transforms. In [35], the authors present quantum algorithms for Haar wavelet transforms and demonstrate applications in analyzing the multiscale structure of a dynamical system by logistic mapping. They show the derivation of the quantum wavelet transform by factoring the classical operators into direct sums, direct products, and dot products, which is the same approach in [16]. The work in [36] also demonstrates similar quantum circuits for QWT based on the well-known pyramid and packet algorithms which are used in classical DWT. The work in [37] presents an analytical study of effects of imperfections in quantum computation of a QWT-based dynamical model. They propose a QWT-based algorithm for the Daubechies wavelets. The works in [38, 39] demonstrate applications of QWT in image watermarking. A more recent, novel watermarking method is proposed in [18], where they demonstrate improvement in invisibility and robustness of the watermarked image. The most recent work on QWT is presented in [19], where the authors provide quantum circuit derivations for the Haar and Daubechies wavelet transforms. The authors propose QHT circuits which contain k levels of permutations, where k is the kernel size.

The previous work on the QWT has mostly presented circuits and software simulations, and no hardware implementations were reported. In comparison, our focus is on efficient hardware implementation of the QWT, and we propose an optimized, low resource-intensive approach for emulation on classical hardware. To the best of our knowledge, our work is the first to (1) propose using QHT for reducing data dimensionality and (2) provide hardware emulation architectures for QHT. Our approach is simpler and optimized for emulation because it uses a single Haar kernel model and a pair of permutation models, where the permutation models are implemented as classical circuits. We propose classical circuits for permutation because (1) quantum permutation circuits implemented using multiple levels of swap operations [16, 19, 35, 36] have large quantum cost, and (2) classical permutation techniques such as index scheduling are space and time efficient for hardware implementation.

Moreover, among the previous work there have been no experimental demonstrations of QWTs on actual quantum hardware or on any quantum emulators. In our work, we present simplified architectures for implementing multilevel, multidimensional QHT operations on classical hardware and propose application of these methods in dimensionality reduction of particle tracking data in high-energy physics applications. Our proposed algorithms and architectures are easily generalizable, compared to previous works. In addition, our proposed architectures are more effective in utilizing minimal quantum and classical hardware resources which is more suited for dimension reduction. We experimentally evaluate the architectures on a high-throughput and high-accuracy FPGA quantum emulator. To the best of our knowledge, this work is the first to present experimental demonstrations of quantum wavelets used for dimension reduction in large-scale applications, e.g., LHC.

2.3. High-Energy Particle Detectors. The ATLAS Fast Tracker (FTK) is a hardware processor upgrade [9] for the Large Hadron Collider (LHC) which has been developed for faster reconstruction of tracks at 100 kHz. Details of the operating principle, hardware components, and performance can be found in [40]. The reconstruction is done by matching detector data with predefined track patterns that are stored in associative memory on ASICs. The data processing and pattern matching are done using FPGA hardware. The FTK receives data from the ATLAS pixel detector and stores them as clusters to reduce data size. The clusters are arranged into regions for parallel processing. In the processing units (PU), the tracks are stored with full resolution on input FPGAs, while other FPGA processors are responsible for converting the stored data into coarser resolution segments. This is followed by comparison of the course-grained segments with pre-stored Monte Carlo track patterns. The coarse granularity of the tracks can cause problems in identification and pattern matching and lead to slower tracking performance of the FTK. In this work, we propose QHT techniques to reduce dimensionality of full resolution data such as FTK particle tracks. We also demonstrate an FPGA-based hardware prototype that can be easily integrated into the current FTK ATLAS architecture.

3. Methodology and Emulation Architectures

In this section, we elaborate our methodology that uses QHT to achieve dimension reduction. We also detail the corresponding emulation hardware architectures that were implemented [41].

3.1. Dimension Reduction. The classical wavelet transform has been shown to achieve dimension reduction efficiently and can be used in various applications that use hyperspectral data, for example, remote sensing, mineralogy, and surveillance. Depending on the type of data and the application in which these data are being used, both 1D wavelet transform (1D-WT) and 2D wavelet transform (2D-WT) techniques can be used for dimension reduction. For example, while the data in remote sensing hyperspectral imagery is in the form of large 3D data cubes, 1D wavelet transform (1D-WT) was previously proposed [13, 14] for efficient dimensionality reduction of such data cubes. In the experimental work in [14], five levels of wavelet decomposition were used on images of size 217×512 pixels by 192 bands to achieve $\times 32$ reductions in data volume. In current and future large-scale applications, the volume of data can be overwhelming. For example, hyperspectral image cubes are typically hundreds of pixels in width and height [13], with 220–240 frequency bands [14]. The ATLAS pixel detector contains 1700 detection modules corresponding to 8×10^7 pixels [8] and has bandwidth capacity of 48 Gb/s [11]. Hence, it is necessary to investigate and apply newer paradigms of information processing and storage for supporting future applications at full bandwidths. In quantum information processing, exponentially greater amount of information can be held in the state of quantum system compared to a classical binary system. Thus, we propose using quantum information processing techniques such as the QWT for the processing of high volumes of data in large-scale applications. For example, a $64\text{K} \times 64\text{K}$ image can be reduced to a smaller resolution of 32×32 using a 32-qubit, 12-level QWT decomposition. The pixels are encoded as N basis states of a quantum state, where $N = 2^n$ and n is the number of qubits, i.e., 32.

Our proposed methodology for dimension reduction using quantum wavelet transforms is shown in Figure 2 [41]. In our proposed approach, each pixel of the input image is encoded as a basis coefficient of a quantum state. Input image data first undergoes a multidimensional quantum Haar transform, e.g., one-dimensional QHT (1D-QHT) or two-dimensional QHT (2D-QHT) operation. The operations can have multiple decomposition levels and separate the input image into a number of low frequency and high frequency replications, depending on the number of decomposition levels. The lowest frequency image replication retains the principal components of the input data without significant data loss. More importantly, the mirror images have reduced dimensionality and thus can be used for reducing preprocessing overhead or communication bandwidth congestion. Multidimensional inverse quantum Haar transform (1D-IQHT or 2D-IQHT) is then applied to reconstruct the original data. The 2D operations can be achieved by cascading 1D operations and multiple permutation sets.

The proposed kernel-based algorithms for multilevel 1D-QHT and 2D-QHT are elaborated in Algorithms 1 and 2, respectively. The algorithms perform multilevel decompositions of 1D-QHT or 2D-QHT operations based on a d-dimensional Haar wavelet kernel. The kernel functionality can be represented by a set of operations applied to some input states/pixels and is preceded and followed by perfect shuffle permutation operations [16] on the input and output states/pixels. The permutation operations are performed by means of index calculations and scheduling. Algorithm 1 performs 1D-QHT on a set of input pixels, X, to produce an output pixel set, Y. The input pixels first undergo input permutations, followed by 1D Haar kernel operations on 2 pixels every cycle, and output permutations. Algorithm 2 performs 2D-QHT on a set of input pixels, X, to produce an output pixel set, Y. The input pixels first undergo input permutations, followed by 2D Haar kernel operations on 4 pixels every cycle, and output permutations.

To efficiently extract output state data, quantum-to-classical readout techniques [28] such as quantum Fourier transform (QFT) can be employed. However, this was not

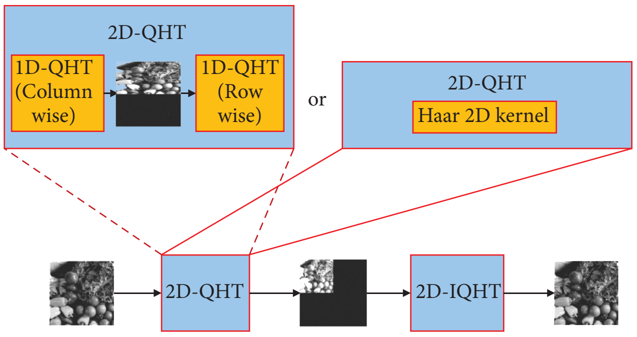


FIGURE 2: Dimension reduction using 2D-QHT and 2D-IQHT.

```
Input: $X = [x_0, x_1, \ldots x_{N-1}]$, $n_{\text{rows}}$, $n_{\text{cols}}$, $n_{\text{levels}}$
Output: $Y = [y_0, y_1, \ldots, y_{N-1}]$
$n_{\text{states}} = n_{\text{rows}} \times n_{\text{cols}} = N$
for $i_{\text{level}} = 1$; $i_{\text{level}} \le n_{\text{levels}}$; $i_{\text{level}}++$ do
    for $i_{\text{group}} = 0$; $i_{\text{group}} < (n_{\text{states}}/2)$; $i_{\text{group}}++$ do
        //Initial scheduler setup
        $i_{\text{colGroup}} = \lfloor i_{\text{group}}/(n_{\text{rows}}/2) \rfloor$
        $i_{\text{rowGroup}} = i_{\text{group}} - i_{\text{colGroup}} \times (n_{\text{rows}}/2)$
        $i_{\text{col}} = i_{\text{colGroup}}$
        $i_{\text{row}} = 2 \times i_{\text{rowGroup}}$
        //Input Permutations/Scheduler
        $i_{X_{00}} = i_{\text{row}} + (i_{\text{col}} \times n_{\text{rows}})$
        $i_{X_{10}} = i_{X_{00}} + 1$
        $X_{00} \longleftarrow X[i_{X_{00}}]$
        $X_{10} \longleftarrow X[i_{X_{10}}]$
        //1D-QHT kernel
        $Y_{00} = (X_{00} + X_{10})/\sqrt{2}$
        $Y_{10} = (X_{00} - X_{10})/\sqrt{2}$
        //Output Permutations/Scheduler
        $i_{Y_{00}} = (i_{\text{row}} + (i_{\text{col}} \times n_{\text{rows}}))/2$
        $i_{Y_{10}} = i_{Y_{00}} + (n_{\text{rows}}/2)$
        $Y[i_{Y_{00}}] \longleftarrow Y_{00}$
        $Y[i_{Y_{10}}] \longleftarrow Y_{10}$
    end for
end for
```

ALGORITHM 1: Multilevel 1D quantum Haar transform.

```
Input: $X = [x_0, x_1, \ldots, x_{N-1}]$, $n_{\text{rows}}$, $n_{\text{cols}}$, $n_{\text{levels}}$
Output: $Y = [y_0, y_1, \ldots, y_{N-1}]$
$n_{\text{states}} = n_{\text{rows}} \times n_{\text{cols}} = N$
for $i_{\text{level}} = 1$; $i_{\text{level}} \le n_{\text{levels}}$; $i_{\text{level}}++$ do
    for $i_{\text{group}} = 0$; $i_{\text{group}} < (n_{\text{states}}/4)$; $i_{\text{group}}++$ do
        //Initial scheduler setup
        $i_{\text{colGroup}} = \lfloor i_{\text{group}}/(n_{\text{rows}}/2) \rfloor$
        $i_{\text{rowGroup}} = i_{\text{group}} - i_{\text{colGroup}} \times (n_{\text{rows}}/2)$
        $i_{\text{col}} = 2 \times i_{\text{colGroup}}$
        $i_{\text{row}} = 2 \times i_{\text{rowGroup}}$
        //Input Permutations/Scheduler
        $i_{X_{00}} = i_{\text{row}} + (i_{\text{col}} \times n_{\text{rows}})$
        $i_{X_{10}} = i_{X_{00}} + 1$
        $i_{X_{01}} = i_{X_{00}} + n_{\text{rows}}$
        $i_{X_{11}} = i_{X_{01}} + 1$
        $X_{00} \longleftarrow X[i_{X_{00}}]$
        $X_{10} \longleftarrow X[i_{X_{10}}]$
        $X_{01} \longleftarrow X[i_{X_{01}}]$
        $X_{11} \longleftarrow X[i_{X_{11}}]$
        //2D-QHT kernel
        $Y_{00} = (X_{00} + X_{10} + X_{01} + X_{11})/2$
        $Y_{10} = (X_{00} - X_{10} + X_{01} - X_{11})/2$
        $Y_{01} = (X_{00} + X_{10} - X_{01} - X_{11})/2$
        $Y_{11} = (X_{00} - X_{10} - X_{01} + X_{11})/2$
        //Output Permutations/Scheduler
        $i_{Y_{00}} = (i_{\text{row}} + (i_{\text{col}} \times n_{\text{rows}}))/2$
        $i_{Y_{10}} = i_{Y_{00}} + (n_{\text{rows}}/2)$
        $i_{Y_{01}} = i_{Y_{00}} + (n_{\text{states}}/2)$
        $i_{Y_{11}} = i_{Y_{01}} + (n_{\text{rows}}/2)$
        $Y[i_{Y_{00}}] \longleftarrow Y_{00}$
        $Y[i_{Y_{10}}] \longleftarrow Y_{10}$
        $Y[i_{Y_{01}}] \longleftarrow Y_{01}$
        $Y[i_{Y_{11}}] \longleftarrow Y_{11}$
    end for
end for
```

ALGORITHM 2: Multilevel 2D quantum Haar transform.

required to be implemented in this work as full emulation of quantum computation was performed on classical hardware and the output of the emulator is in classical representation. For emulation, we develop circuit models based on these algorithms and integrate them into reconfigurable hardware architectures for multilevel, multidimensional (1D and 2D) QHT and IQHT. These models and emulation architectures are elaborated in the next section.

3.2. Quantum Haar Transform Kernel. The Haar wavelet kernel can be generalized by quantum operations using n qubits and a d-dimension kernel as shown in (7), where $\otimes$ is the Kronecker product [42], H is the Hadamard transform [20], and I is an identity matrix. Here, a group of entangled gates is denoted by the gate symbol with the size of the equivalent operation matrix as subscript, for example, H_{2^d}. The quantum Haar function can be implemented using d entangled H gates and $n-d$ entangled I gates as shown in (7). For example, the transformation matrix for 2D-QHT with $d = 2$ can be derived as shown in (9):

$$U_{\text{QHT}} = I_{2^{(n-d)}} \otimes H_{2^d}, \tag{7}$$

where

$$H_{2^d} = \underbrace{H \otimes H \otimes \cdots \otimes H}_{d},$$

$$I_{2^{(n-d)}} = \underbrace{I \otimes I \otimes \cdots \otimes I}_{(n-d)},$$

$$H = H_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \tag{8}$$

$$I = I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

where $n \Longrightarrow$ number of qubits and $d \Longrightarrow$ kernel dimension:

$$U_{\mathrm{QHT}}^{2\mathrm{D}} = I_{2^{(n-2)}} \otimes H_{2^2} = I_{2^n/4} \otimes H_4 = I_{N/4} \otimes H_4, \tag{9}$$

where

$$H_4 = H \otimes H = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}. \tag{10}$$

3.3. Permutation Operations. Perfect shuffle permutation on a given vector is described as partitioning the vector in half and shuffling the top and bottom portions of the halves [16]. In our algorithms for QHT and IQHT, we apply similar input and output permutation operations before and after applying the QHT kernel, respectively. The QHT kernel is performed on a set of k points, where $k = 2^d$. An input permutation operation involves dividing the input vector of size N, into k groups, and selecting a state (pixel) from every group(s), to be applied to the kernel operation. For 1D-QHT and 2D-QHT operations, the input permutations, $P_{\mathrm{in}}^{1\mathrm{D}}$ and $P_{\mathrm{in}}^{2\mathrm{D}}$, are shown in (11) and (12), respectively. An output permutation involves arranging the pixels from k groups into a single output state sequence. The output permutation for 1D-QHT and 2D-QHT operations, $P_{\mathrm{out}}^{1\mathrm{D}}$ and $P_{\mathrm{out}}^{2\mathrm{D}}$, are shown in (13) and (14), respectively:

$$P_{\mathrm{in}}^{1\mathrm{D}} : \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{(n_{\mathrm{rows}})} \\ x_{(n_{\mathrm{rows}}+1)} \\ \vdots \\ x_{(N-1)} \end{bmatrix} \longmapsto \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{(n_{\mathrm{rows}})} \\ x_{(n_{\mathrm{rows}}+1)} \\ \vdots \\ x_{(N-1)} \end{bmatrix}, \tag{11}$$

$$P_{\mathrm{in}}^{2\mathrm{D}} : \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{(n_{\mathrm{rows}})} \\ x_{(n_{\mathrm{rows}}+1)} \\ x_{(n_{\mathrm{rows}}+2)} \\ x_{(n_{\mathrm{rows}}+3)} \\ \vdots \\ \vdots \\ x_{(N-1)} \end{bmatrix} \longmapsto \begin{bmatrix} x_0 \\ x_1 \\ x_{(n_{\mathrm{rows}})} \\ x_{(n_{\mathrm{rows}}+1)} \\ x_2 \\ x_3 \\ x_{(n_{\mathrm{rows}}+2)} \\ x_{(n_{\mathrm{rows}}+3)} \\ \vdots \\ \vdots \\ \vdots \\ x_{(N-1)} \end{bmatrix}, \tag{12}$$

$$P_{\mathrm{out}}^{1\mathrm{D}} : \begin{bmatrix} y_0 \\ y_{(n_{\mathrm{rows}}/2)} \\ y_1 \\ y_{((n_{\mathrm{rows}}/2)+1)} \\ \vdots \\ \vdots \\ y_{(N-1)} \end{bmatrix} \longmapsto \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_{(N-1)} \end{bmatrix}, \tag{13}$$

$$P_{\mathrm{out}}^{2\mathrm{D}} : \begin{bmatrix} y_0 \\ y_{(n_{\mathrm{rows}}/2)} \\ y_{(N/2)} \\ y_{((n_{\mathrm{rows}}/2)+(N/2))} \\ \vdots \\ \vdots \\ y_{(N-1)} \end{bmatrix} \longmapsto \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ \vdots \\ \vdots \\ y_{(N-1)} \end{bmatrix}. \tag{14}$$

3.4. Emulation Architectures. While developing the emulation architectures for the proposed system, as an intermediate step, we design circuit models, illustrated in Figures 3 and 4 for 1D- and 2D-QHT/IQHT, respectively. These models are derived from the sequence of operations in Algorithms 1 and 2 and can contain quantum and/or classical circuits. The 1D- and 2D-QHT models in Figures 3(a) and 4(a), respectively, consist of input permutation models (P_{in}), followed by Haar kernel models (U_{QHT}) and then output permutation models (P_{out}). The 1D- and 2D-IQHT models in Figures 3(b) and 4(b), respectively, consist of inverse output permutation models $(P_{\mathrm{out}})^{-1}$, followed by Haar kernel models (U_{QHT}) and then inverse input permutation models $(P_{\mathrm{in}})^{-1}$. The inverse models are equivalent to the direct models, as the permutation operations are reversible. To achieve multilevel decompositions, multiple iterations of the Haar kernel models are applied. The QHT and IQHT operations for 1D and 2D are summarized as unitary transformations in (15) and (16), respectively. The emulation architectures of the 1D-QHT/IQHT and 2D-QHT/IQHT are shown in Figures 5 and 6, respectively. Since the hardware implementations of the 1D and 2D are similar, we focus our following discussions on the implementation of the 2D-QHT emulation architectures:

$$\begin{aligned} &1\mathrm{D}-\mathrm{QHT} : P_{\mathrm{out}}^{1\mathrm{D}} \cdot U_{\mathrm{QHT}}^{1\mathrm{D}} \cdot P_{\mathrm{in}}^{1\mathrm{D}}, \\ &1\mathrm{D}-\mathrm{IQHT} : \left(P_{\mathrm{in}}^{1\mathrm{D}}\right)^{-1} \cdot U_{\mathrm{QHT}}^{1\mathrm{D}} \cdot \left(P_{\mathrm{out}}^{1\mathrm{D}}\right)^{-1}, \end{aligned} \tag{15}$$

$$\begin{aligned} &2\mathrm{D}-\mathrm{QHT} : P_{\mathrm{out}}^{2\mathrm{D}} \cdot U_{\mathrm{QHT}}^{2\mathrm{D}} \cdot P_{\mathrm{in}}^{2\mathrm{D}}, \\ &2\mathrm{D}-\mathrm{IQHT} : \left(P_{\mathrm{in}}^{2\mathrm{D}}\right)^{-1} \cdot U_{\mathrm{QHT}}^{2\mathrm{D}} \cdot \left(P_{\mathrm{out}}^{2\mathrm{D}}\right)^{-1}. \end{aligned} \tag{16}$$

As shown in Figures 4(a) and (16), the first step in the 2D-QHT operation is the input permutation $P_{\mathrm{in}}^{2\mathrm{D}}$, which is described by (12). The permutations can be modeled as quantum circuits with multiple swap gates, but that would incur high resource utilization in the corresponding emulation architecture. For this reason, we use classical models

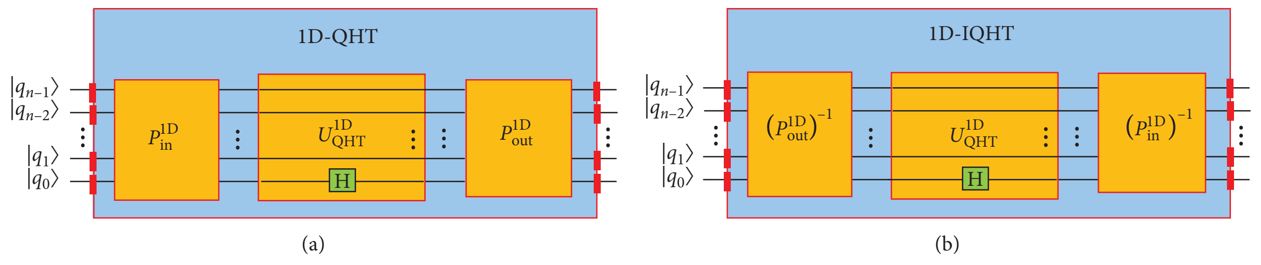


FIGURE 3: (a) 1D-quantum Haar transform circuit. (b) 1D-inverse quantum Haar transform circuit.

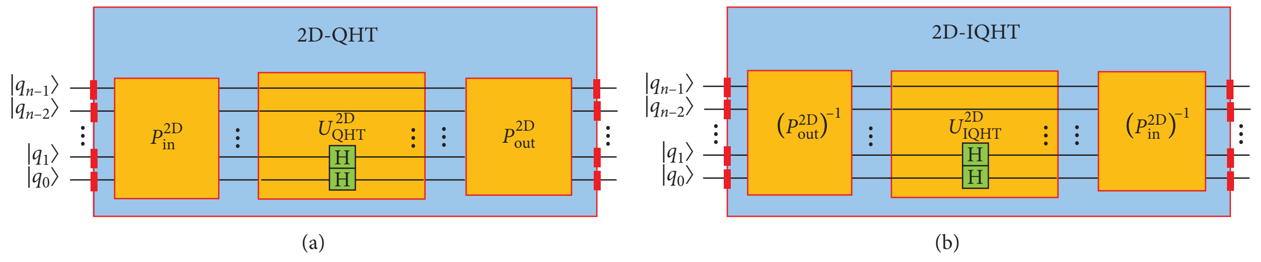


FIGURE 4: (a) 2D-quantum Haar transform circuit. (b) 2D-inverse quantum Haar transform circuit.

that involve simple index scheduling, and the corresponding emulation architecture is shown in Figure 6(a). The input is a vector of quantum state coefficients which are written to a memory array in the index order 0 to $N-1$. Four coefficient values are then read out each clock cycle, with the scheduler generating the read indices $i_{X_{00}}$, $i_{X_{01}}$, $i_{X_{10}}$, and $i_{X_{11}}$ according to the input permutation, see Algorithm 2 and (12). The scheduler maintains a counter, row index i_{row}, and a column index i_{col} to calculate the output indices. Multiplications and divisions by powers of two are replaced by logical shifts for optimizing area and speed. The scheduler also requires a floor operation unit.

As shown in Figures 4(b) and (9), the 2D-Haar transformation, U^{2D}_{QHT}, is modeled using a pair of Hadamard gates. The Hadamard pair operation reduces to kernel operations on a set of four coefficients as we described in Algorithm 2. The emulation architecture for the 2D-Haar kernel is shown in Figure 6(b). The design takes in four input coefficients, applies the kernel operations which involve addition and division, and outputs four coefficients per clock cycle. Conventional operator sharing techniques and logical shifts are applied to optimize for speed and area.

The final step in the 2D-QHT operation is the output permutation, P^{2D}_{out}, described by (14). The corresponding emulation architecture is shown in Figure 6(c) and works similar to the input permutation scheduler. The input vector of coefficients are written to a memory array, four values per clock cycle, with the scheduler generating the write indices $i_{Y_{00}}$, $i_{Y_{01}}$, $i_{Y_{10}}$, and $i_{Y_{11}}$ according to the output permutation described in Algorithm 2. The permuted coefficients are then read out from memory 4 values per clock cycle.

The emulation hardware architectures, i.e., input/output schedulers and 1D/2D Haar kernels, were integrated into a reconfigurable quantum emulator design based on our previous works [21, 22], whose high-level architecture is shown in Figure 7. The emulator stores input and output quantum states as vectors of the state coefficients and core kernel operations are extracted from the input quantum algorithm. The input state vector goes through the input permutations (input schedulers) before the kernel operation is applied iteratively across each state. To get the correct final quantum state that represents the transformed data, the output permutation (output schedulers) is applied. The architecture uses a fully pipelined dataflow architecture and supports single and double-precision floating-point arithmetic. For example, each quantum state coefficient is complex and is modeled in 32 bit floating-point precision for the real and imaginary components, respectively. The emulator also supports features such as fully-entangled input quantum state preparation from a set of input qubits and output quantum state measurement as a classical bit string. The emulator is generic and can efficiently run a given quantum algorithm that can be reduced to its corresponding unitary transformation.

4. Experimental Work

The experimental work was performed on DS8, a state-of-the-art high-performance reconfigurable computing (HPRC) system developed by DirectStream [25]. On the DS8 platform, developers can build applications on hardware systems ranging from single-node compute instances to multinode structures, see Figure 8. A single C2 compute node of the DS8 system is equipped with a high-end Intel-Altera Arria 10AX115N4F45E3SG FPGA, with on-chip resources such as adaptive logic modules (ALMs), block RAMs (BRAMs), digital signal processors (DSPs), and on-board resources such as two 32 GB SDRAM memory banks and four 8 MB SRAM memory banks, as shown in Figure 8. A user-friendly programming environment, previously known as Carte-C [43], is integrated into the DS hardware

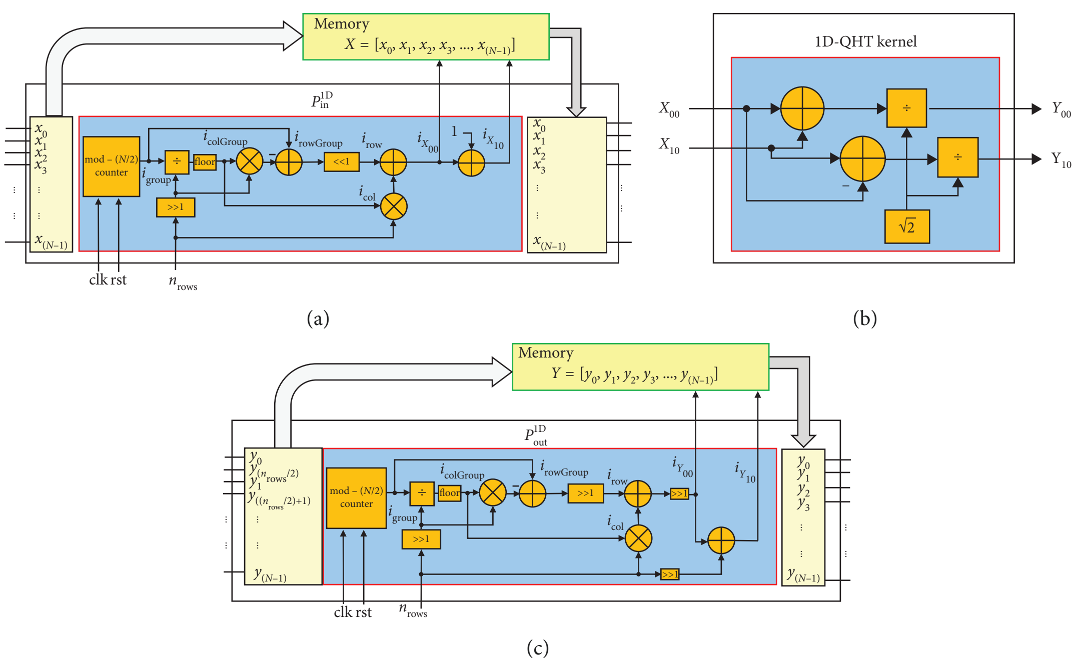


Figure 5: Emulation architectures for the (a) 1D input permutation, (b) 1D-Haar kernel, and (c) 1D output permutation.

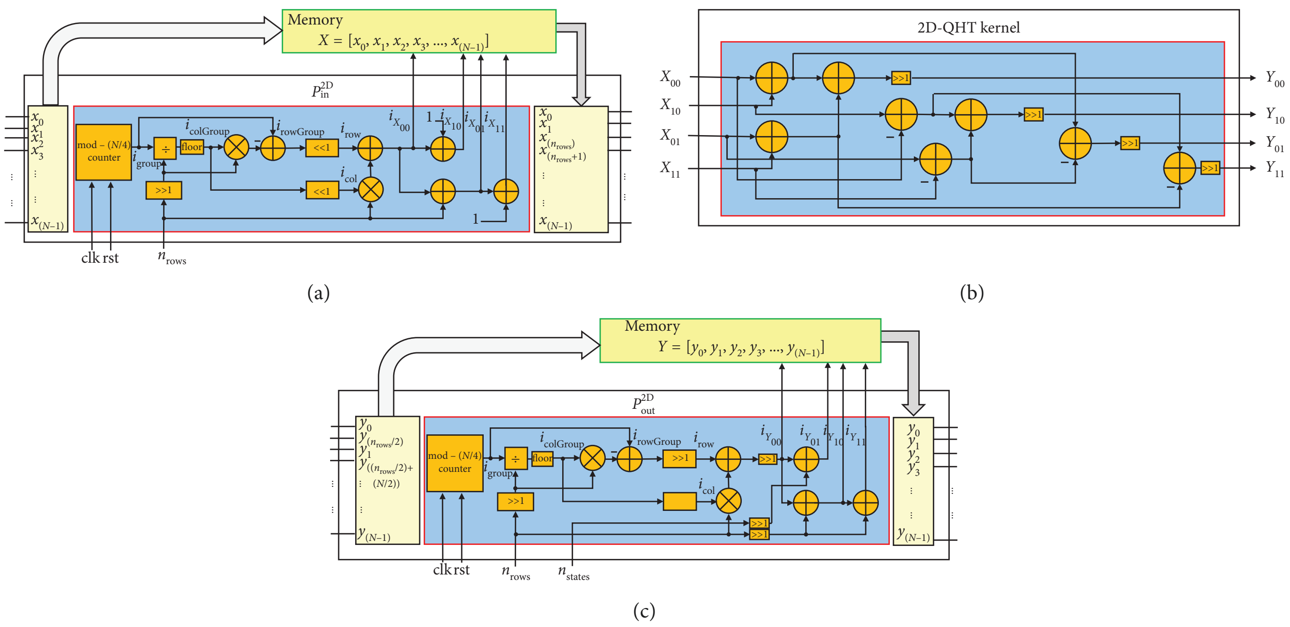


Figure 6: Emulation architectures for the (a) 2D input permutation, (b) 2D-Haar kernel, and (c) 2D output permutation.

systems. A high-level language (HLL) facilitates the development of complex, parallel, and reconfigurable codes in an efficient manner. The study in [44] showed that Carte-C has a highly productive environment, short acquisition time, and short learning time as well as a short development time. The DS8 architecture provides a combination of high performance, high scalability, runtime reconfiguration, and ease of use.

The QHT and IQHT architectures were implemented using C++ on the DS8 programming environment. Input images with a resolution of up to 1024×1024, and 256 shades of grayscale pixels, were used to test the designs. MATLAB was used to convert the images into greyscale, generate the input vectors for DS8, and reconstruct images from the output vectors. Synthesis and hardware builds were performed using Quartus Prime Version 17.02 on the DS8 environment. Figure 9(a) shows one of the input images converted to greyscale, and Figure 9(b) is the output after a 1D-QHT operation with 1 level of decomposition. Figure 9(c) is the output after a 1D-QHT operation with 2

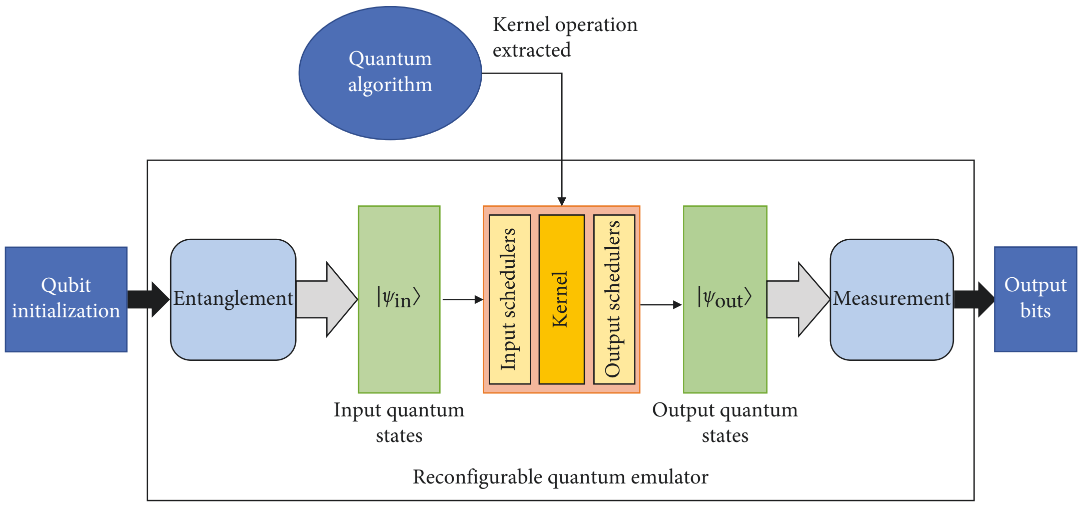


Figure 7: Reconfigurable quantum emulator architecture.

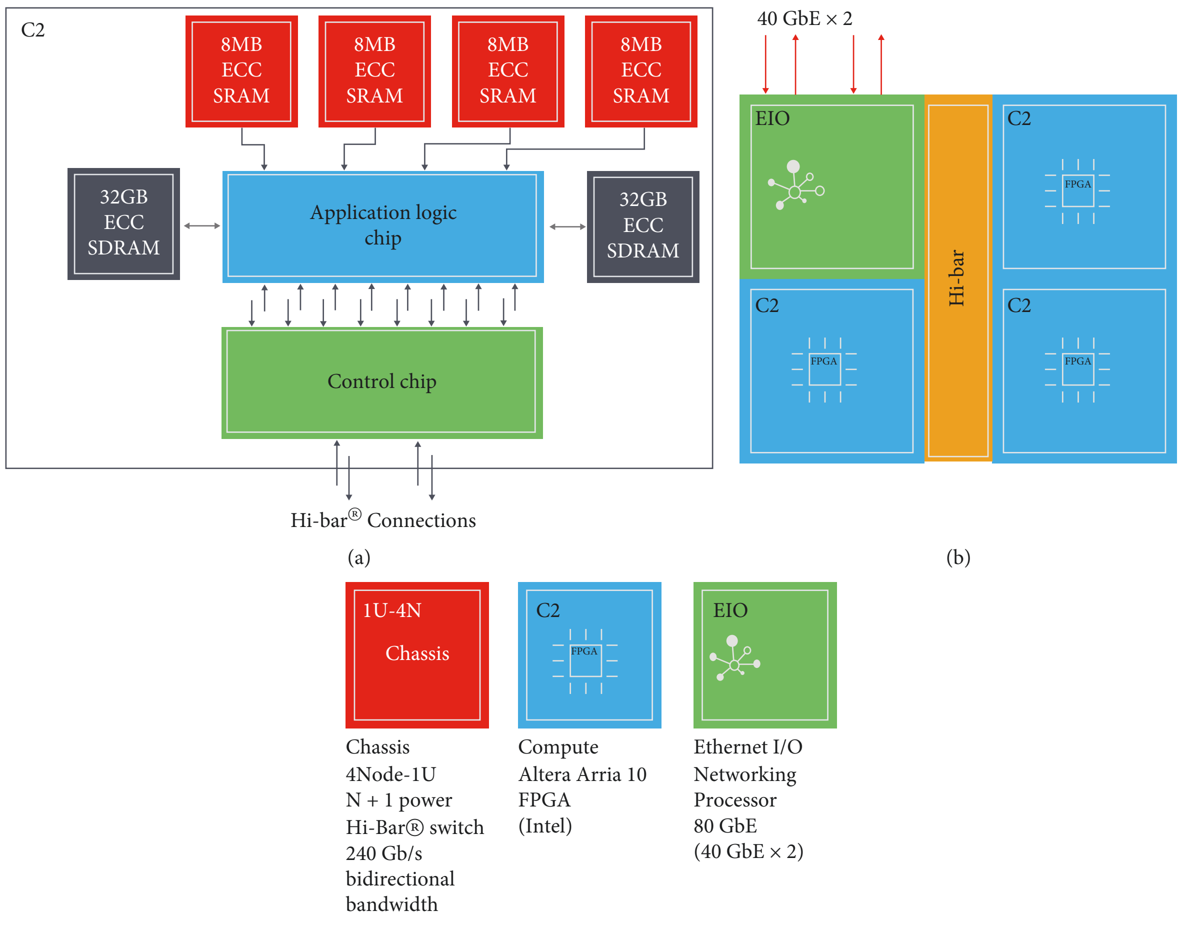


Figure 8: DS8 platform architectures. (a) Single compute node. (b) Multinode instance. (c) Node types.

levels of decomposition, and Figure 9(d) shows the reconstructed images after a 1D-IQHT operation was applied. Figures 10(a)–10(d) show the results from repeating the experiment using the 2D-QHT and 2D-IQHT architectures.

Resource utilizations from the hardware implementations are summarized in Tables 1 and 2 for 1D and 2D, respectively. The on-chip resources (ALMs, BRAMs, DSPs) are used up in implementing the static components of the design such as counters, adders, and shift operators and

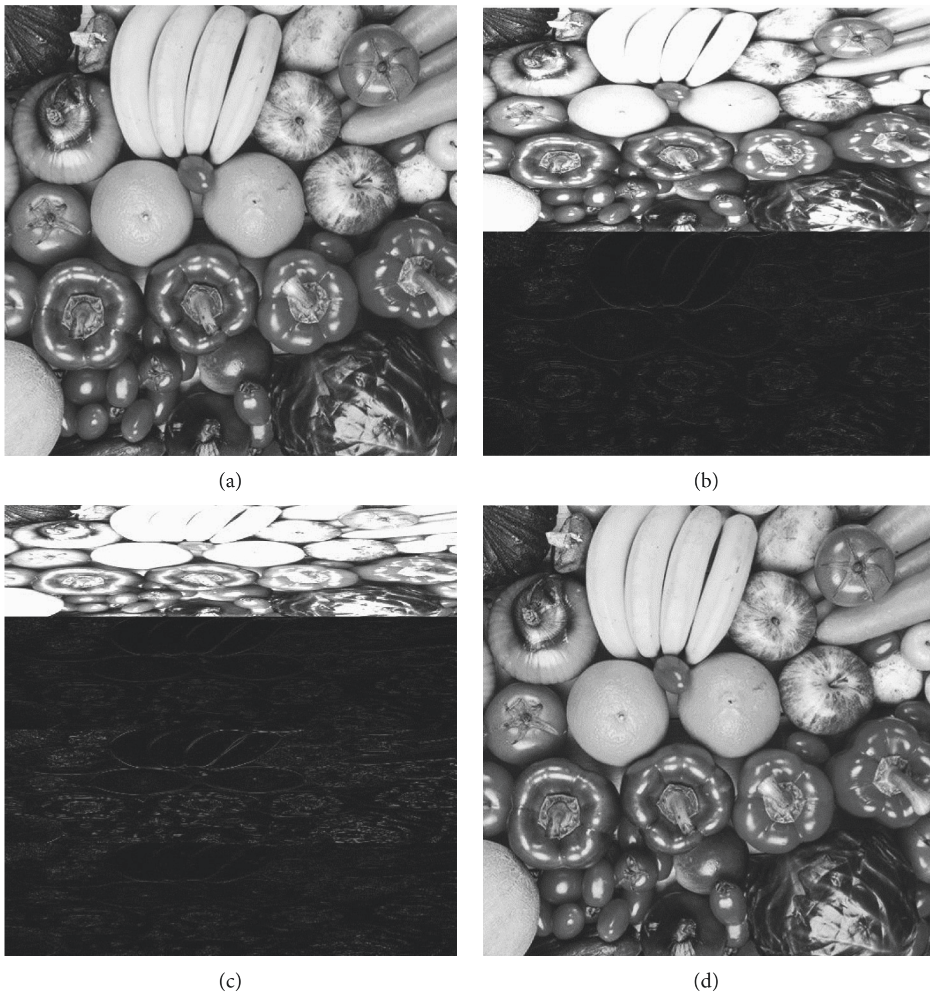


Figure 9: Experimental results of multilevel decomposition and reconstruction with 1D-QHT and 1D-IQHT. (a) Original image. (b) 1-level 1D-QHT. (c) 2-level 1D-QHT. (d) Reconstructed image using 1D-IQHT.

hence are constant as the emulated circuit size (number of qubits) increases. The low on-chip resource utilizations indicate that our proposed approach and emulation architecture designs are highly space-efficient. The 1D-QHT architecture consumes lower on-chip resources than 2D-QHT due to its less complex kernel operations. The low resource utilizations also indicate the flexibility of the QHT and IQHT designs for integrating with larger algorithms.

The SDRAM memory requirements for storage of the input and output images as quantum state vectors are also reported in Tables 1 and 2. For the highest resolution image of size 1024×1024, the pixels occupy 25% of the total on-board SDRAM memory (64 GB) available on a single DS node. The pixels of the input images are encoded as basis coefficients of a quantum state. For example, to store 16×16 or 256 pixels, we need 256 complex coefficients each of which have a real and imaginary component occupying total $2 \times 4 = 8$ bytes in 32 bit floating-point representation. Therefore, for storing both input and output images, $2 \times 256 \times 8 = 4096$ bytes of memory was required. The obtained memory usages for larger QHT circuits are consistent with expected values.

The hardware designs on the FPGA were pipelined to ensure a constant and high operating frequency of 233 MHz. The obtained emulation times for high resolution images are also feasible. For a 1024×1024 image, 20 qubits were sufficient for achieving dimension reduction using 1D-QHT and 2D-QHT. From our experimental results, we observe that the emulation time increases linearly with increase in the number of image pixels (states), as illustrated in Figure 11. This is because a large portion of the emulation time is dedicated to writing in and reading out the input/output state vectors of size N (number of pixels); hence, the emulation time complexity is $O(N)$. This indicates the benefit of using quantum encoding of data, i.e., encoding each image pixel as a basis state coefficient in the quantum state space. Finally, the emulation times for 1D-QHT are higher than 2D-QHT because of the higher number of iterations $N/2$ in the 1D algorithm, compared to $N/4$ iterations in the 2D algorithm, see Algorithms 1 and 2.

In general, on a classical emulation platform, the emulation execution time increases with both the spatial and temporal complexities of the quantum circuit. In other words, the emulation time of a quantum circuit on a classical

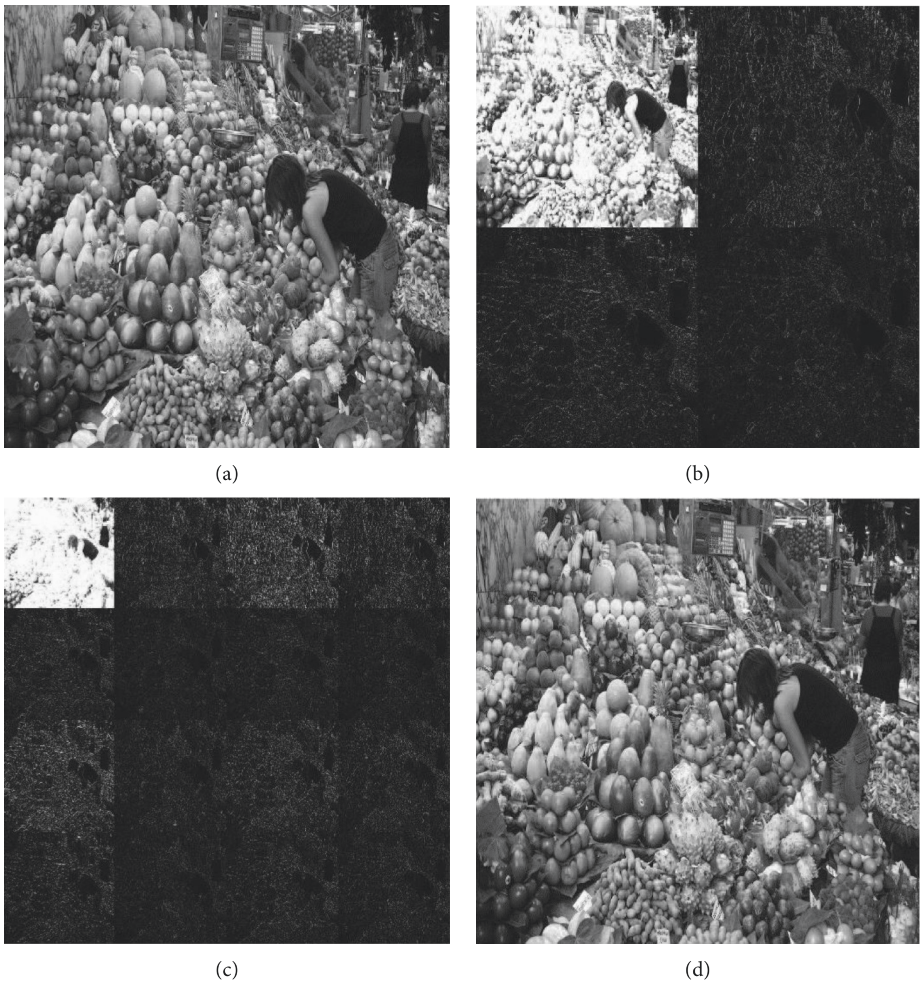


Figure 10: Experimental results of multilevel decomposition and reconstruction with 2D-QHT and 2D-IQHT. (a) Original image. (b) 1-level 2D-QHT. (c) 2-level 2D-QHT. (d) Reconstructed image using 2D-IQHT.

Table 1: 1D-QHT implementation results on Arria 10AX115N4F45E3SG FPGA.

Number of pixels	Number of qubits	Resource utilization* (%)			SDRAM** (bytes)	Emulation time (sec)
		ALMs	BRAMs	DSPs		
16 × 16	8	11	8	1	4 K	0.00018
32 × 32	10	11	8	1	16 K	0.00071
64 × 64	12	11	8	1	64 K	0.00285
128 × 128	14	11	8	1	256 K	0.01139
256 × 256	16	11	8	1	1 M	0.04557
512 × 512	18	11	8	1	4 M	0.18226
1024 × 1024	20	11	8	1	16 M	0.72905

*Total chip resources: $N_{\mathrm{ALM}} = 427{,}200$; $N_{\mathrm{BRAM}} = 2{,}713$; $N_{\mathrm{DSP}} = 1{,}518$. **Total on-board SDRAM memory: 2 parallel banks of 32 GB each.

Table 2: 2D-QHT implementation results on Arria 10AX115N4F45E3SG FPGA.

Number of pixels	Number of qubits	Resource utilization* (%)			SDRAM** (bytes)	Emulation time (sec)
		ALMs	BRAMs	DSPs		
16 × 16	8	14	9	2	4 K	0.00012
32 × 32	10	14	9	2	16 K	0.00047
64 × 64	12	14	9	2	64 K	0.00187
128 × 128	14	14	9	2	256 K	0.00746
256 × 256	16	14	9	2	1 M	0.02982
512 × 512	18	14	9	2	4 M	0.11926
1024 × 1024	20	14	9	2	16 M	0.47704

*Total chip resources: $N_{\mathrm{ALM}} = 427{,}200$; $N_{\mathrm{BRAM}} = 2{,}713$; $N_{\mathrm{DSP}} = 1{,}518$. **Total on-board SDRAM memory: 2 parallel banks of 32 GB each.

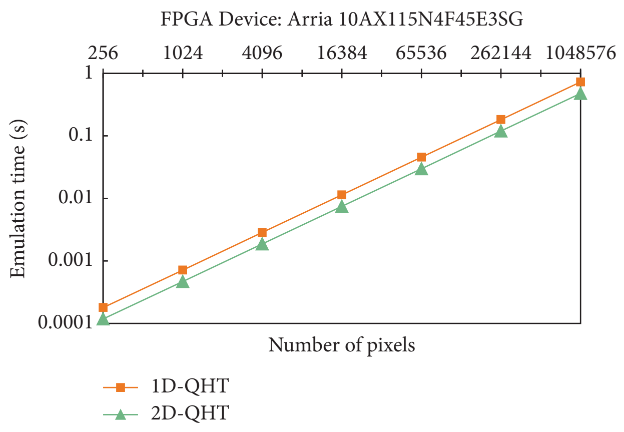


Figure 11: Emulation time as a function of data size (number of pixels).

Table 3: Comparison of the proposed work against previous works of FPGA emulation.

Reported work	Algorithm	Number of qubits	Precision	Frequency (MHz)	Emulation time (sec)
Fujishima [48]	Shor's factoring	—	—	80	10
Khalid et al. [49]	QFT	3	16 bit fixed pt.	82.1	$61E-9$
	Grover's search	3	16 bit fixed pt.	82.1	$84E-9$
Aminian et al. [50]	QFT	3	16 bit fixed pt	131.3	$46E-9$
Lee et al. [51]	QFT	5	24 bit fixed pt.	90	$219E-9$
	Grover's search	7	24 bit fixed pt.	85	$96.8E-9$
Silva et al. [52]	QFT	4	32 bit floating pt.	—	$4E-6$
Pilch et al. [53]	Deutsch	2	—	—	—
Mahmud et al. [22]	QFT	5	32 bit floating pt.	233	$4.63E-4^{\dagger}$
	Grover's search	5	32 bit floating pt.	233	$4.38E-7^{\dagger}$
Proposed work	QFT	20	32 bit floating pt.	233	18.4
	QHT	20	32 bit floating pt.	233	0.477
	Grover's search	22	32 bit floating pt.	233	$7.5E04$

†Results obtained at a later time to publication.

platform is generally a function of both the circuit width (number of qubits) and depth (number of gate levels). Due to optimizations and encoding techniques we used, the emulation time of our proposed emulation architectures is a function of only the quantum circuit width (number of qubits), as shown by our experimental results. On state-of-the-art superconducting NISQ devices [45, 46], the execution time is a function of only the depth (number of gate levels) of the circuit [47]. For our proposed 1D-QHT and 2D-QHT circuits, which are simple quantum circuits of depth 1, we estimate an execution time of 0.01 ms on a typical NISQ device processing a 7×7 qubit array with sampling frequency of 100 kHz [47]. The estimated execution time is constant for a fixed circuit depth and variable number of qubits in the quantum processing unit (QPU) array; i.e., the time complexity is theoretically $O(1)$. In comparison, the time complexity of our emulation is $O(N)$.

Our emulation experiments and implementations help in validating the functionality and feasibility of the proposed QHT-based methodology in achieving dimension reduction of high-resolution images. The emulation provides implications for the proposed system's application in fast, efficient processing of particle tracking data in the large-scale, high-energy physics domain. The emulation is memory-bound by the resources on a single DS FPGA node. For larger-scale emulation, the on-board memory has to be increased, or multi-node, and/or multichassis architectures of the DS system can be utilized in conjunction with efficient scheduling techniques and high-bandwidth networks [22].

We further quantitatively compare our obtained experimental results with the existing FPGA-based emulation work [48–53] as shown in Table 3. Among the related work on FPGA emulation of quantum circuits, our emulator has the capability of emulating the largest quantum circuits (QFT, QHT, and Grover's search), with highest operating frequency (233 MHz) and high precision (32 bit floating-point). Current FPGA hardware-emulators have many discrepancies (missing resource utilization, operating frequency, and emulation time) in the reporting of their results which makes a comprehensive comparison difficult. In our comparison, we included only hardware emulators, as most parallel-software-simulators are based on large-scale supercomputers such as Summit [47] and Sunway [54], which are extremely costly, power-hungry, and resource-hungry and are not comparable with FPGA-emulators. Also, they provide simulations of random quantum circuits and not full quantum algorithms.

5. Conclusions

Quantum information processing and quantum computing will have significant implications in the future of computing technology. As current quantum technology continues to improve, there is a great need to investigate useful applications in quantum information theory. In this work, we presented a first effort, to the best of our knowledge, to efficiently reduce data dimensionality using quantum processing methods such as quantum wavelet transform. We propose to apply these techniques in physics applications that investigate high-energy particle detection and tracking, where dimension reduction helps to reduce communication bandwidth and speedup preprocessing computations. Our proposed architectures are simpler and optimized for hardware implementation than previously reported works. We demonstrated the minimal resource utilization, high performance/throughout, and high precision of the proposed architectures. We prototyped our designs on a quantum emulator and demonstrated the feasibility of proposed techniques by conducting experiments using high-resolution test image data.

Due to limitations of the current state of quantum technology, e.g., cost, availability, and current scale (size) of quantum processors, it is beyond the scope of this work to actually implement the system and measure performance. Although not yet integrated with the ATLAS FTK project, the proposed approach and emulation hardware architectures are feasible for future implementations, with the maturing of current quantum technology. For future integration into the ATLAS FTK project, data conversion techniques such as quantum-to-classical and classical-to-quantum, which are heavily-researched current topics, must be perfected first, and we plan to conduct investigations of these techniques in our future work. Our future plans also include application of the proposed methods using real HEP data and combining QHT with Grover's search algorithm as a complete solution to HEP FTK problems. We will also investigate 3D-QHT, Daubechies wavelet transforms, and their application for real-time data streaming.

Data Availability

The test data used to support the findings of this study are available from the corresponding author upon request and approval from Direcstream.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

We would like to thank Prof. Alice Bean and Prof. Christopher Rogan from the department of Physics and Astronomy at the University of Kansas for their valuable insights and help in this work.

References

[1] CERN Accelerating science, "The Higgs boson," 2019, https://home.cern/science/physics/higgs-boson.

[2] CERN Accelerating science, "Unified forces," 2019, https://home.cern/science/physics/unified-forces.

[3] V. L. Ginzburg, *The Physics of a Lifetime: Reflections on the Problems and Personalities of 20th Century Physics*, Springer Science & Business Media, Berlin, Germany, 2013.

[4] I. Kisel, *Track Reconstruction and Pattern Recognition in High-Energy Physics*, 2019, https://www.physik.uni-heidelberg.de/c/image/exp/f/highrr/Kisel_HD_12.04.2016.pdf.

[5] G. Aad and ATLAS Collaboration, "The ATLAS experiment at the CERN large Hadron collider," *Journal of Instrumentation*, vol. 3, no. 8, 2008.

[6] C. O'Luanaigh, *New Results Indicate that New Particle is a Higgs Boson*, CERN, Geneva, Switzerland, 2019, https://home.cern/news/news/physics/new-results-indicate-new-particle-higgs-boson.

[7] CERN Accelerating science, "The inner detector," 2019, https://atlas.cern/discover/detector/inner-detector.

[8] F. Hugging, "The ATLAS pixel detector," in *Proceedings of the IEEE Symposium Conference Record Nuclear Science*, vol. 2, pp. 1077–1081, Rome, Italy, October 2004.

[9] M. Backhaus, "The upgraded pixel detector of the ATLAS experiment for run 2 at the large Hadron collider," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 831, pp. 65–70, 2016.

[10] H. Pernegger, *The Pixel Detector of the ATLAS Experiment for LHC Run-2*, 2015.

[11] S. Amerio, A. Andreani, A. Andreazza et al., *ATLAS FTK: Fast Track Trigger*, 2013.

[12] I. K. Fodor, *A Survey of Dimension Reduction Techniques*, pp. 1–18, Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA, USA, 2002.

[13] S. Kaewpijit, J. Le Moigne, and T. El-Ghazawi, "Automatic reduction of hyperspectral imagery using wavelet spectral analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 4, 2003.

[14] E. El-Araby, T. El-Ghazawi, J. Le Moigne, and K. Gaj, "Wavelet spectral dimension reduction of hyperspectral imagery on a reconfigurable computer," in *Proceedings of the IEEE International Conference on Field-Programmable Technology (FPT)*, pp. 399–402, Brisbane, Australia, December 2004.

[15] J. Wickmann, "A wavelet approach to dimension reduction and classification of hyperspectral data," Masters Thesis, Faculty of Mathematics and Natural Sciences, University of Oslo, Oslo, Norway, 2007.

[16] A. Fijany and C. P. Williams, *Quantum Wavelet Transforms: Fast Algorithms and Complete Circuits*, 1998, http://arxiv.org/abs/9809004v1.

[17] J. Stajic, "The future of quantum information processing," *Science*, vol. 339, no. 6124, p. 1163, 2013.

[18] S. Heidari, M. Naseri, R. Gheibi, M. Baghfalaki, M. R. Pourarian, and A. Farouk, "A new quantum watermarking based on quantum wavelet transforms," *Communications in Theoretical Physics*, vol. 67, no. 6, p. 732, 2017.

[19] L. Hai-Sheng, P. Fan, H. Xia, S. Song, and X. He, "The multilevel and multi-dimensional quantum wavelet packet transforms," *Scientific Reports*, vol. 8, no. 1, 2018.

[20] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, UK, 2010.

[21] N. Mahmud and E. El-Araby, "A scalable high-precision and high-throughput architecture for emulation of quantum algorithms," in *Proceedings of the 31st IEEE International System-on-Chip Conference (SOCC 2018)*, Washington, DC, USA, September 2018.

[22] N. Mahmud and E. El-Araby, "Towards higher scalability of quantum hardware emulation using efficient resource scheduling," in *Proceedings of the 3rd IEEE International Conference on Rebooting Computing (ICRC 2018)*, Washington, DC, USA, November 2018.

[23] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings of the 35th IEEE Annual Symposium on Foundations of Computer Science (SFCS '94)*, pp. 124–134, Santa Fe, NM, USA, November 1994.

[24] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of computing (STOC '96)*, pp. 212–219, Philadelphia, PA, USA, May 1996.

[25] DirectStream LLC, 2019 , https://directstream.com.

[26] Quantum Computing Report, *Qubit Technology*, 2019, https://quantumcomputingreport.com/scorecards/qubit-technology/.

[27] L. Gomes, "Quantum computing: both here and not here," *IEEE Spectrum*, April 2019, https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=83-22045.

[28] C. P. Williams, *Explorations in Quantum Computing*, Springer Science & Business Media, Berlin, Germany, 2010.

[29] L. Grover and T. Rudolph, *Creating Superpositions that Correspond to Efficiently Integrable Probability Distributions*, 2002, http://arxiv.org/abs/0208112.

[30] P. Kaye and M. Mosca, *Quantum Networks for Generating ArbitraryQuantum States*, pp. 1–3, 2004, http://arxiv.org/abs/0407102.

[31] X. Yao, H. Wang, Z. Liao et al., "Quantum image processing and its application to edge detection: theory and experiment," *Physical Review X*, vol. 7, no. 031041, 2017.

[32] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.

[33] B. Toufik and N. Mokhtar, "The wavelet transform for image processing applications," in *Advances in Wavelet Theory and Their Applications in Engineering, Physics and Technology*, pp. 395–422, InTech, Rijeka, Croatia, 2012.

[34] P. S. Addison, *The Illustrated Wavelet Transform Handbook: Introductory Theory and Applications in Science, Engineering, Medicine and Finance*, CRC Press, Boca Raton, FL, USA, 2017.

[35] D. Gosal and W. Lawton, *Quantum Haar Wavelet Transforms and Their Applications*, 2001.

[36] H. Ohnishi, H. Matsueda, and L. Zheng, "Quantum wavelet transform and matrix factorization," in *Proceedings of the IEEE International Quantum Electronics Conference*, pp. 1327-1328, Antwerp, Belgium, 2005.

[37] M. Terraneo and D. L. Shepelyansky, "Imperfection effects for multiple applications of the quantum wavelet transform," *Physical Review Letters*, vol. 90, no. 25, 2003.

[38] X.-H. Song, S. Wang, S. Liu, A. A. Abd El-Latif, and X.-M. Niu, "A dynamic watermarking scheme for quantum images using quantum wavelet transform," *Quantum Information Processing*, vol. 12, no. 12, pp. 3689–3706, 2013.

[39] Y.-G. Yang, P. Xu, J. Tian, and H. Zhang, "Analysis and improvement of the dynamic watermarking scheme for quantum images using quantum wavelet transform," *Quantum Information Processing*, vol. 13, no. 9, pp. 1931–1936, 2014.

[40] N. Ilic, "The ATLAS fast tracker and tracking at the high-luminosity LHC," *Journal of Instrumentation*, vol. 12, no. 2, 2017.

[41] N. Mahmud, E. El-Araby, and D. Caliga, "Scaling reconfigurable emulation of quantum algorithms at high precision and high throughput," *Quantum Engineering*, vol. 1, no. 2, 2019.

[42] R. K. Brylinski and G. Chen, *Mathematics of Quantum Computation*, CRC Press, Boca Raton, FL, USA, 2002.

[43] E. El-Araby, T. El-Ghazawi, J. Le Moigne, and R. Irish, "Reconfigurable processing for satellite on-board automatic cloud cover assessment," *Journal of Real-Time Image Processing*, vol. 4, no. 3, pp. 245–259, 2009.

[44] E. El-Araby, S. G. Merchant, and T. El-Ghazawi, "Assessing productivity of high-level design methodologies for high-performance reconfigurable computers," in *High-Performance Computing Using FPGAs*, W. Vanderbauwhede and K. Benkrid, Eds., pp. 719–745, Springer, New York, NY, USA, 2013.

[45] A Preview of Bristlecone, Googles New Quantum Processor, Google AI Blog, March 2018.

[46] IBM Announces Advances to IBM Q Systems & Ecosystem, IBM Press Release, November 2017.

[47] B. Villalonga, D. Lyakh, S. Boixo et al., "Establishing the quantum supremacy frontier with a 281 Pflop/s simulation," 2019, http://arxiv.org/abs/1905.00444v1.

[48] M. Fujishima, "FPGA-based high-speed emulator of quantum computing," in *Proceedings of the IEEE International Conference on Field-Programmable Technology (FPT 2003)*, Tokyo, Japan, December 2003.

[49] A. U. Khalid, Z. Zilic, and K. Radecka, "FPGA emulation of quantum circuits," in *Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD 04)*, pp. 310–315, San Jose, CA, USA, October 2004.

[50] M. Aminian, M. Saeedi, M. S. Zamani, and M. Sedighi, "FPGA-based circuit model emulation of quantum algorithms," in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI '08)*, pp. 399–404, Montpellier, France, April 2008.

[51] Y. H. Lee, M. Khalil-Hani, and M. N. Marsono, "An FPGA-based quantum computing emulation framework based on serial-parallel architecture," *International Journal of Reconfigurable Computing*, vol. 2016, Article ID 5718124, 18 pages, 2016.

[52] A. Silva and O. G. Zabaleta, "FPGA quantum computing emulator using high level design tools," in *Proceedings of the Eight Argentine Symposium and Conference on Embedded Systems (CASE'17)*, pp. 1–6, Buenos Aires, Argentina, August 2017.

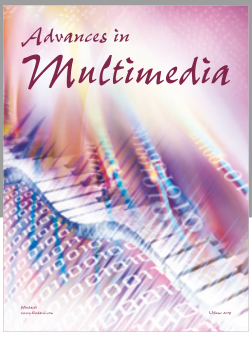
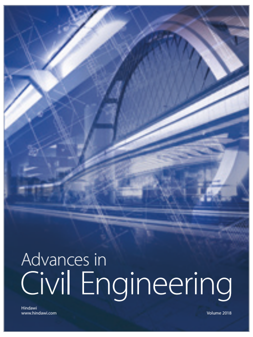
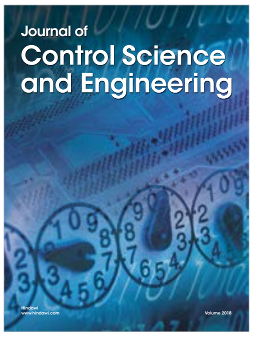
[53] J. Pilch and J. Długopolski, "An FPGA-based real quantum computer emulator," *Journal of Computational Electronics*, pp. 1–14, 2018.

[54] R. Li, B. Wu, M. Ying, X. Sun, and G. Yang, "Quantum supremacy circuit simulation on Sunway TaihuLight," 2018, http://arxiv.org/abs/1804.04797.

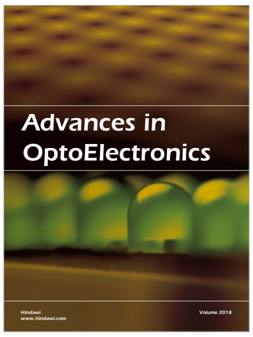
Journal of
Engineering

The Scientific World Journal

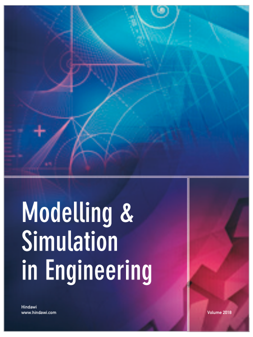
International Journal of
Rotating Machinery

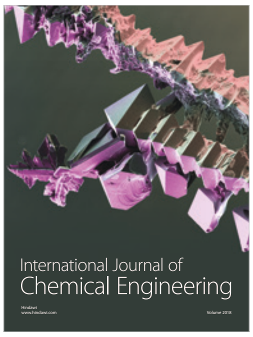
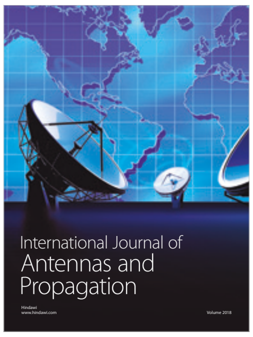
Journal of
Sensors

Advances in
Multimedia

Advances in
Civil Engineering

Journal of
Control Science and Engineering

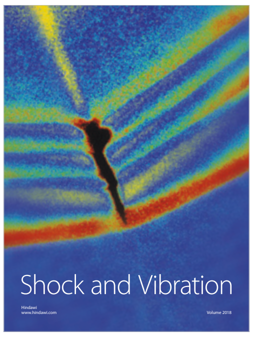
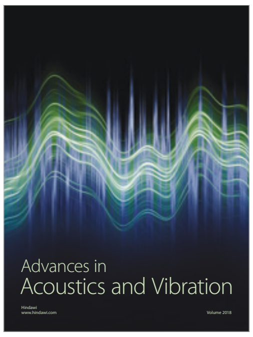
Journal of
Robotics

Journal of
Electrical and Computer Engineering

Advances in
OptoElectronics

VLSI Design

International Journal of
Navigation and Observation

Modelling & Simulation in Engineering

International Journal of
Aerospace Engineering

International Journal of
Chemical Engineering

International Journal of
Antennas and Propagation

Active and Passive
Electronic Components

Shock and Vibration

Advances in
Acoustics and Vibration