COMPAQT: Compressed Waveform Memory Architecture for Scalable Qubit Control

Satvik Maurya
University of Wisconsin-Madison
Madison, USA
smaurya@wisc.edu

Swamit Tannu
University of Wisconsin-Madison
Madison, USA
swamit@cs.wisc.edu

Abstract—On superconducting architectures, the state of a qubit is manipulated by using microwave pulses. Typically, the pulses are stored in the waveform memory and then streamed to the Digital-to-Analog Converter (DAC) to synthesize the gate operations. The waveform memory requires tens of Gigabytes per second of bandwidth to manipulate the qubit. Unfortunately, the required memory bandwidth grows linearly with the number of qubits. As a result, the bandwidth demand limits the number of qubits we can control concurrently. For example, on current RFSoCs-based qubit control platforms, we can control less than 40 qubits. In addition, the high memory bandwidth for cryogenic ASIC controllers designed to operate within a tight power budget translates to significant power dissipation, thus limiting scalability.

In this paper, we show that waveforms are highly compressible, and we leverage this property to enable a scalable and efficient microarchitecture COMPAQT - *Compressed Waveform Memory Architecture for Qubit Control*. Waveform memory is read-only and COMPAQT leverages this to compress waveforms at compile time and store the compressed waveform in the on-chip memory. To generate the pulse, COMPAQT decompresses the waveform at runtime and then streams the decompressed waveform to the DACs. Using the hardware-efficient discrete cosine transform, COMPAQT can achieve, on average, 5x increase in the waveform memory bandwidth, which can enable 5x increase in the total number of qubits controlled in an RFSoC setup. Moreover, COMPAQT microarchitecture for cryogenic CMOS ASIC controllers can result in a 2.5x power reduction over uncompressed baseline. We also propose an adaptive compression scheme to further reduce the power consumed by the decompression engine, enabling up to 4x power reduction.

Qubits are sensitive, and even a slight change in the control waveform can increase the gate error rate. We evaluate the impact of COMPAQT on the gate and circuit fidelity using IBM quantum computers. We see less than 0.1% degradation in fidelity when using COMPAQT.

Keywords-Qubit Control; Quantum Computer Architecture; Quantum Control Hardware;

I. INTRODUCTION

The true potential of quantum computers can be unlocked by building a quantum computer that can run error correction on thousands of qubits to enable fault tolerance. Several industry and academic labs have built machines with more than fifty qubits [6], [28], with plans to increase the number of qubits to thousands of qubits to demonstrate fault-tolerance [62]. However, building a large-scale quantum computer is challenging. Qubit devices are susceptible to
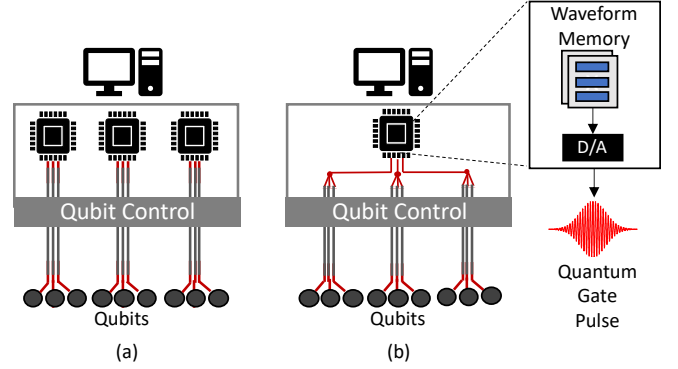

Figure 1. (a) Distributed Qubit Control with FPGAs (b) Integrated Qubit Control with RFSoCs.

noise and require precise control. Superconducting quantum computers use microwave pulses to perform operations on qubits. A typical quantum program that uses tens of qubits requires hundreds of such pulses. The control hardware sends these pulses through coaxial cables to qubits at cryogenic temperatures, while the control hardware works at room temperature. Today, all quantum computers employ brute-force approaches to scaling the control hardware where discrete, off-the-shelf components like Field Programmable Gate Arrays (FPGAs), Digital to Analog Converters (DACs), and Analog to Digital Converters (ADCs) are replicated depending on the number of qubits in the target quantum processor. For example, the control hardware of the Sycamore chip [6] required 200+ DACs, 9 ADCs, and 30+ FPGAs for control and readout of 53 qubits. As the number of qubits increases, this brute-force scaling of control hardware will become infeasible in terms of both cost and complexity.

Increasing integration – i.e., decreasing the number of discrete hardware components used for quantum control will make qubit control more scalable. At present, there are two broad approaches for increasing integration: (1) Using off-the-shelf RFSoCs (2) Building custom Qubit Control ASICs. RFSoCs integrate an FPGA, a processor, DACs, and ADCs on the same chip and are typically used for telecommunication applications. RFSoCs can synthesize signals with tens of gigahertz bandwidth. Recently proposed "QICK" and "ICARUS-Q" platforms leverage RFSoCs to
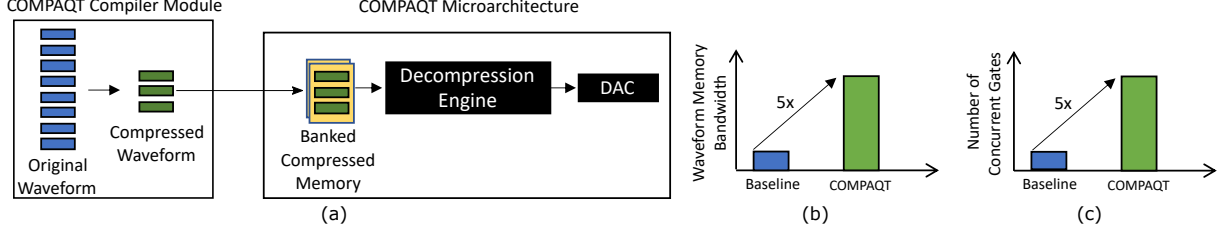
Figure 2. (a) Organization of COMPAQT (b) Waveform memory bandwidth boost with COMPAQT (c) Improvement in the number of concurrent gates that a controller can support.

control qubits [55], [70]. Figure 1 shows how Qubit Control with RFSoCs would differ from a distributed approach. In theory, RFSocs are capable of supporting 100+ qubits per board (with the help of frequency-division multiplexing (FDM)) [70]. Compared to designing ASICs that can integrate a large number of DACs and ADCs on a chip, RFSoC based solutions offer flexibility that is crucial for near term qubit control as quantum hardware is rapidly evolving. As discussed in [70], using FDM with high-bandwidth DACs and ADCs available on these RFSoCs can increase the number of qubits controlled by one board, thus reducing the total cost and complexity of the setup. However, controlling more than thousands of qubits would require custom control hardware and new quantum-classical interfaces. We envision the following transition for qubit control as we scale the number of qubits.

FPGA controllers → RFSoC controllers → ASIC controllers

To scale beyond thousands of qubits, we will need qubit controller that can operate at cryogenic temperatures [7], [17], [21], [30], [74]. For cryogenic control chips, the cooling capacity of the dilution refrigerator results in the "Power Wall" – the average power dissipated by these chips should not exceed the rated cooling power of the dilution refrigerator. To this end, several architectural and design proposals envision scalable qubit control using ultra-low power CMOS and SFQ technologies [7], [17], [21], [30], [47], [73].

With increasing integration, potentially more qubits can be supported per control chip, reducing both complexity and cost. However, the memory bandwidth and capacity required to generate the pulses will scale linearly with the number of qubits as each qubit device requires a unique pulse shape (waveform) to enable high fidelity gates [18], [63], [69]. Furthermore, for performing multi-qubit gates, most architectures use a coupler between qubits, and for reading the qubit, the readout resonators are used, and even these non-qubit components use device-specific pulses to enable high-fidelity operations. As a result, the number of unique pulses grows super-linearly with the increasing number of qubits. For example, we estimate, on IBM quantum computers, each qubit device requires about 18KB to store single-qubit, two-qubit, and readout pulses. To support more gates, we require

additional memory capacity. We estimate that a hundred-qubit quantum computer would require up to 5MB of memory for pulse shapes of basic gates. Moreover, we will require significantly more memory capacity to support complex gates. For example, we estimate that a single Toffoli gate can require >20KB to store a pulse applied on three qubits [34].

Memory bandwidth is the primary bottleneck in scaling the number of qubits. For example, on IBM quantum computers, two DACs per qubit (with a sampling rate of 4.54 Gigasamples/sec each) are required to perform a single-qubit gate. We need a waveform memory to stream samples with more than 16 GB/s bandwidth to such DACs. Furthermore, the peak bandwidth required scales linearly with the number of qubits. To perform concurrent gates on a hundred-qubit machines, we need more than 2 TB/s. Current qubit control setups such as the one used for Google's Sycamore chip solve this problem using distributed waveform memory across 30+ FPGAs. However, distributed control does not scale practically with increasing number of qubits.

This paper focuses on reducing bandwidth and capacity overhead to scale qubit control. Waveforms are designed to have a tight frequency spectrum to limit crosstalk and leakage errors. As a result, waveforms are smooth and highly compressible. We evaluate the compressibility of waveforms used on three IBM machines and see that on average, we can reduce the capacity overhead by 8x using Discrete Cosine Transform (DCT) based compression. We use DCT based compression due to its high energy compaction properties. Moreover, DCT can be efficiently implemented in hardware. The waveform memory is loaded before the execution starts and is not updated during the execution; current compilers update waveform libraries only during calibration.

In this paper, we propose COMPAQT: *Compressed Waveform Memory Architecture for Scalable Qubit Control*. COMPAQT mitigates the waveform bandwidth bottleneck by compressing waveform libraries at compile time and storing compressed waveforms in the waveform memory. To execute a gate, COMPAQT decompresses the waveforms and then streams it to the DAC as shown in Figure 2(a). COMPAQT boosts the memory bandwidth as the hardware decompression engine expands the memory contents, increasing the data supplied to the DAC per unit time, as shown in Figure 2(b).
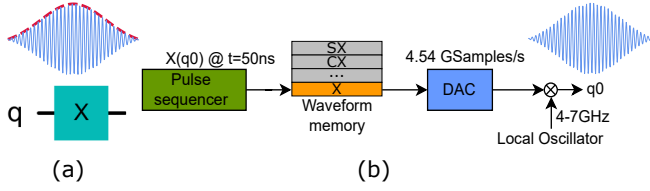
Figure 3. (a) An X-gate and its corresponding microwave pulse (in blue) The envelope of the pulse is shown in red (b) A typical pipeline for generating control pulses.

Figure 4. π-pulse shapes of all (a) 27 qubits on IBM Toronto (b) 65 qubits on IBM Brooklyn (c) 127 qubits on IBM Washington machines.

This bandwidth gain directly translates to a gain in the number of concurrent operations that a controller can support, as shown in Figure 2(c). COMPAQT trades memory bandwidth with computational bandwidth. To enable fast and efficient decompression, we use a windowed DCT algorithm during compression which is decompressed with a corresponding inverse DCT (IDCT) engine in hardware. Windowing reduces hardware complexity, but it also reduces net compressibility. We pick window sizes that maximize the net bandwidth benefits. Fixed/floating-point multipliers in the IDCT algorithm are a major challenge in building a high-speed and efficient decompression engine. We use an integer IDCT algorithm [76] that replaces multipliers with shifters and adders to significantly improve the performance and efficiency of COMPAQT.

The key contributions of this paper are summarized below:

- **Waveform memory bandwidth is a bottleneck**: We show how the bandwidth demand reduces the scalability of RFSoC based controllers (Section III).
- **Gate Pulse Waveforms are highly compressible**: We show that waveforms can be effectively compressed using the Discrete Cosine Transform without degrading the gate fidelity (Section IV).
- **COMPAQT Microarchitecture**: We present a compressed waveform memory architecture that can increase the number of qubits supported by RFSoC based controllers by 5x (Section V).

II. BACKGROUND

This section gives a brief description of how qubits are controlled, the organization of a typical control computer, and the challenge of scaling waveform memory.

A. Manipulating Qubits via Gate Pulses

Quantum algorithms can be expressed as quantum circuits, a sequence of gate operations applied on qubit variables. On superconducting architectures, the gates are implemented using microwave pulses as qubit devices (such as transmons) are essentially oscillators that respond to signals in the frequency range of 4-10 GHz. Quantum control refers to the synthesis of these microwave pulses to implement various quantum gates. These microwave pulses can have different amplitudes, s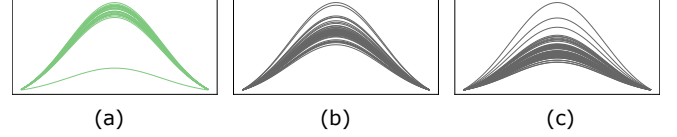hapes, and duration depending on the type of gate and the qubit architecture. The state of a qubit is represented by a vector on a Bloch sphere [53]. When a pulse is applied, the state rotates about the X/Y/Z axis of the Bloch sphere. Since Z rotations can be implemented in software [49], microwave pulses are composed of two components that control the qubit state evolution about the X and Y-axis of the Bloch sphere. Consider the circuit shown in Figure 3(a), with X-gate applied on qubit q1, which inverts the qubit state. Physically, this gate is implemented as a pulse with a frequency equal to the qubit resonant frequency (shown in blue). The envelope of the pulse refers to its shape (shown as dotted line), and we will refer to it as a waveform henceforth.

The physical implementation of two-qubit (2Q) gates such as CNOT/CZ depends on qubit architecture [35]. For example, IBM machines use fixed-frequency qubits, wherein 2Q gates are implemented using Cross-Resonance (CR) gates [14], [61] whereas the frequency-tunable qubits used by Google implement a 2Q gate called the iSWAP gate [6], [35]. Both implementations may require additional pulses for the coupler that enables interactions between neighboring qubits. In addition to standard gates, custom pulses can significantly boost application fidelity. For example, Toffoli gates have been demonstrated on superconducting hardware [34], [71]. Furthermore, Shi et al. proposed to use custom pulses to enable multi-qubit operators to reduce circuit depth and improve fidelity [67], and Xie et al. showed that carefully re-shaping the waveforms can reduce ZZ-crosstalk [78].

B. Control Computer Organization

Figure 3(b) shows a simplified control pipeline used to generate microwave gate pulses to manipulate the state of the qubit. A pulse sequencer is used to trigger the execution of gates (an X gate for qubit q0 in this case). The pulse envelopes are stored in the waveform memory, and they consist of the Inphase (I) and Quadrature (Q) components for rotating the qubit about the X- and Y-axis of the Bloch sphere. Since DACs have a fixed sampling rate (sampling rate of IBM machines annotated in the figure), the waveform memory interface should supply data with a bandwidth that matches the sampling rate to ensure signal integrity. Higher DAC sampling rates are desirable for high-fidelity gates. The waveforms are converted to pulses at an intermediate frequency by a DAC. The resultant pulse is then mixed with a microwave carrier generated by a local oscillator (LO) to produce the final microwave pulse.

State-of-the-art quantum control hardware uses discrete FPGAs, DACs, and ADCs placed at room temperature. Controllers can be made more scalable with the use of RFSoC platforms that integrate a CPU, FPGA, DACs, and ADCs on the same chip. The CPU is interfaced with the control hardware to calibrate qubits and load experiments, while the qubits are actively controlled using an FPGA. As suggested in [70], a single RFSoC can control more than 100 qubits with the help of frequency division multiplexing (FDM), making this approach more scalable in terms of cost and complexity. The high-bandwidth DACs/ADCs on-chip eliminate the need for external analog mixers; pulses can be directly synthesized at the desired frequency. Other ways of implementing the control computer include ASICs that can integrate large numbers of discrete components but are expensive to design and manufacture.

C. Challenge: Scaling Waveform Memory

Unfortunately, every qubit has a unique waveform for every physical gate that the system supports. As shown in Figure 4, every qubit on IBM quantum computers has a different π pulse, which is tuned to maximize gate fidelity. These waveforms are periodically updated via calibration cycles. Furthermore, the connections between adjacent qubits (couplers) are unique, and this results in different waveforms used for controlling the qubit and the coupler for multi-qubit gates like the `CNOT` or `Toffoli`. Finally, pulses used for readout have waveforms that depend on the qubit.

With the increasing number of qubits, the total number of unique waveforms that have to be stored in the waveform memory increase linearly. Furthermore, this increase in capacity requirements will be accompanied by an increase in memory bandwidth requirements, since more qubits would need to be driven concurrently, especially for running quantum error correction circuits. While this increase in memory capacity and bandwidth requirements is not a bottleneck with brute-force scaling of control, the next section will show how the memory architecture in integrated RFSoC controllers can limit scalability.

III. WAVEFORM MEMORY BOTTLENECK

In this section, we will show the impact of linearly scaling the bandwidth demand of qubit control on FPGA-based systems. We use parameters described in the Table I to model capacity and bandwidth requirements, which are derived from IBM [3], [29] and Google [6], [26], [48] systems.

We estimate memory capacity (MC) and bandwidth (BW) required for controlling one qubit -

$$MC = \sum_{i=0}^{n_{sq}} f_s.N_s.\tau_i + \sum_{j=0}^{d*n_{tq}} f_s.N_s.\tau_j + f_s.N_s.\tau_{readout}$$

The product of sampling frequency of DAC (f_s), sample size (N_s), and gate latency(τ) is the memory required for storing one gate waveform. We can perform a of total n_{sq}

Vendor	Sampling Rate(f_s)	Sample Size(N_s)	Gate Set	Gate Latencies(τ)	Connectivity	Memory (per qubit)
IBM	4.54GS/s	32-bits	X, SX, CX	1Q:30ns; 2Q:300ns; Readout:300ns	Heavy hexagonal	18KB
Google	1GS/s	28-bits	fsim, iSWAP, phased XZ	1Q:25ns; 2Q:30ns; Readout:500ns	Grid	3KB

types of single qubit gates and n_{tq} types of two qubit gates. Furthermore, two qubit waveforms are unique for every pair of qubits, and thus for a qubit with d neighboring qubits, we will need $d*n_{tq}$ waveforms. In addition to gates, each qubit will need a readout pulse with latency $\tau_{readout}$. As shown in Table I, IBM uses two single-qubit gates and one type of two-qubit gate, which require a total of 18KB of waveform memory for every qubit device. Note that the sample size s accounts for both I and Q channels. The required memory bandwidth of waveform buffer is determined by the sampling frequency and bit-width of DAC as shown below.

$$BW = f_s.s$$

A. Scaling Capacity and Bandwidth Demand

Capacity scaling. Every qubit device requires a unique waveform, which is determined during the calibration process. It is essential to build control engine that supports the device specific waveforms to enable high gate fidelity. As a result, required memory capacity scales linearly with number of qubits. In addition to storing waveforms for qubits themselves, some architectures require waveforms for couplers that connect qubits to improve fidelity of two qubit (2Q) gates. The capacity required by the coupler waveforms depends on the connectivity, and this must be considered when determining the total capacity. Coupler waveforms are unique for every pair of qubits. As shown in Figure 5(a), the required waveform memory capacity scales linearly with the number of qubits in the system (some approximations made to account for coupler waveforms). The RFSoC capacity includes both Block-RAM (BRAM) and UltraRAM (URAM) blocks [79] and has been added as a reference for understanding how the capacity requirements scale (both BRAMs and URAMs will be collectively referred as BRAMs from here on).

Bandwidth scaling. Figure 5(b) shows the linear scaling of the required memory bandwidth to drive all qubits concurrently with the 6GS/s DACs found on RFSoCs. Using such high-bandwidth DACs eliminates the need for external analog mixing, requiring frequent re-calibration. Google/IBM systems will show a similar, albeit lesser bandwidth scaling since those systems employ brute-force scaling with discrete
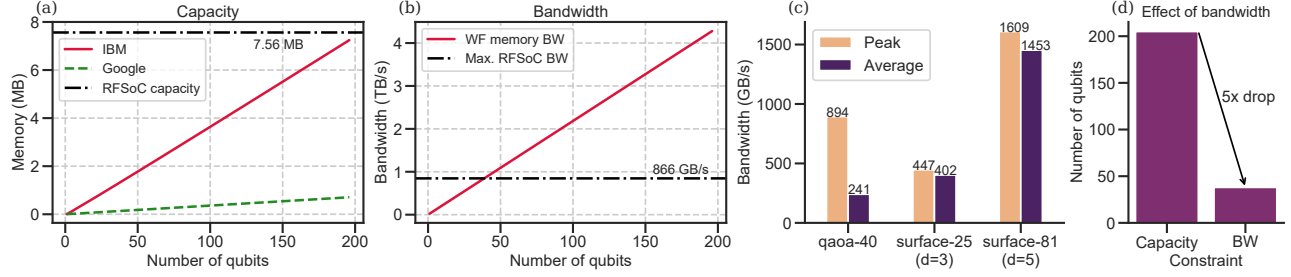
Figure 5. Waveform (WF) memory (a) capacity and (b) bandwidth (BW) requirements (c) Peak and average bandwidth for select benchmarks (d) For an RFSoC platform, if only capacity is considered a constraint, the number of qubits that can be supported is >200. If bandwidth is taken as a constraint, this number drops by 5x to less than 40.

DACs and FPGAs. The reference RFSoC bandwidth[1] shows that a single RFSoC can concurrently control less than 40 qubits, as FPGA clocks are significantly slower than DAC sampling rates. For example, On the QICK platform, the DAC is 16x slower compared to waveform memory.

Circuit Scalability. To understand the impact of waveform memory bandwidth, we estimate the peak and average bandwidth required for representative circuits that users would want to run in the near future. Average bandwidth is determined by how many gates are performed concurrently on an average, whereas peak bandwidth is determined by the maximum concurrency needed for executing the circuit. Shown in Figure 5(c), which plots the peak and average bandwidth for three benchmarks – Quantum Approximate Optimization Algorithm (QAOA) with 40 qubits, a distance-3 surface code with 25 qubits, and a distance-5 surface code with 81 qubits. For surface codes, the difference between peak and average bandwidth is small as quantum error correction codes such as the surface code [20] are designed to run concurrent gates as any delay in generating and measuring quantum error syndromes can significantly degrade the protection offered by the error correction code. NISQ applications such as QAOA are not bandwidth intensive on average. However, the last step of all NISQ circuits involves the concurrent measurement of all qubits, requiring the maximum possible waveform memory bandwidth. Unfortunately, serializing measurements is not an option as it can significantly degrade readout fidelity.

In summary, for RFSoCs, bandwidth becomes a bottleneck as they can support the capacity but not the bandwidth to run surface code circuits with highly concurrent gates.

B. Meeting Capacity and Bandwidth Demand

The demand for capacity and bandwidth of waveform memory scales linearly with the number of qubits. This presents a challenge for scaling quantum control computers, irrespective of whether it is implemented on an RFSoCs or

[1]Determined by assuming a 16x difference in clock speeds of the FPGA and DAC and considering 1260 BRAMs [79].

an ASIC. On RFSoCs, the internal memory bandwidth and on-chip memory capacity can limit how many qubits we can control. We can boost the memory bandwidth by interleaving (or banking) the memory but due to rigid constraints on Block RAMs, the peak internal bandwidth is limited.

On cryogenic ASICs, power is the primary constraint and thus providing the necessary capacity and bandwidth within a limited power budget is challenging. Furthermore, techniques to improve scalability of qubit control, such as Frequency Division Multiplexing (FDM), cannot be supported if the waveform memory is too small or if it cannot provide the necessary bandwidth. For example, the QICK framework can potentially control 100+ qubits per RFSoC board with FDM. However, before waveforms are mixed onto a single channel for FDM, the waveforms for all the multiplexed qubits must be stored and then individually generated, which means that the waveform memory must have sufficient capacity and bandwidth for all qubits. As shown in Figure 5(d), the bandwidth on RFSoC platforms is insufficient to control more than 40 qubits.

IV. WAVEFORM COMPRESSION

In this section, first we give an overview of COMPAQT, and then discuss different methods that can be used to enable compression and their impact on the gate fidelity.

A. COMPAQT: Insight and Overview

Insight. We observe that waveforms are highly compressible as they are designed to have a tight bandwidth and low spectral leakage, which means they are smooth and change slowly in time-domain. We leverage this property to alleviate the high waveform memory bandwidth. We propose COMPAQT: *Compressed Waveform Memory Architecture for Scalable Qubit Control*. By compressing the waveforms, we reduce the required capacity, and by using a low-latency decompression engine, we can quickly expand the compressed waveform to meet the desired memory bandwidth demand.

Design Overview. Figure 6 shows overview of COMPAQT, where waveforms are compressed in the software and then transferred to the controller waveform memory. The pulse sequencer can then play waveforms by decompressing each
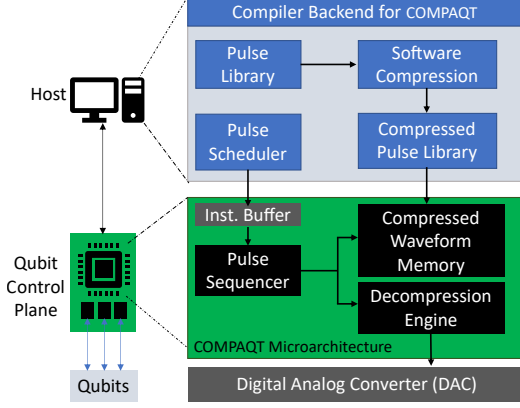
Figure 6.   Architectural overview of COMPAQT

sample and then passing it to the DAC. COMPAQT leverages the read-only nature of the Waveform Memory, which is loaded with waveforms at the end of a calibration cycle. To reduce capacity and bandwidth overhead, COMPAQT compresses the waveforms such that only a few samples are needed for representing the entire waveform. Since waveform memory is a ROM, compressing the waveforms has no hardware overhead and can be done in purely software while decompressing the waveforms must be done entirely on hardware. Note that there could be other (and better) compression techniques, the techniques used in this paper are purely to demonstrate the compressibility of waveforms and the advantages compression offers for improving scalability.

B. Selecting the Compression Algorithm

Base-delta and dictionary-based compression are widely studied algorithms to maximize on-chip memory capacity [2], [57]. However, traditional memory compression algorithms are not effective in compressing waveform data. Delta compression reduces the number of bits by storing only the difference between adjacent samples. However, waveforms can have zero crossings or sign changes. If samples are represented by signed data types (fixed or floating point), the zero crossings will result in a difference that occupies the entire bit-field of the original samples. Furthermore, any non-uniform deltas will add complexity to decompression. Our evaluations on IBM hardware, show that the size of waveforms with no zero-crossings can be reduced by up to 2x as deltas generally occupy half the original bit-width of the sample. However, there is no reduction for waveforms with one or more zero crossings, as shown in Figure 7(a). Moreover, in base-delta compression there is a sequential dependence between samples, which limits its utility for improving bandwidth. Similarly, other widely used dictionary-based compression schemes are not effective, as waveform sample values can have arbitrary values, which rarely repeat.

We propose to use **transform-based compression**, which is widely used for audio and video codecs. These compression schemes leverage the Discrete Cosine Transform (DCT) that transforms the input data from the spatial domain to the frequency domain [1], where data can be represented with significantly less memory [60], [72]. Specifically, the DCT of the input signal results in high-energy components in the first few samples after which the magnitude of the transformed samples becomes significantly smaller. The original signal can be recovered by computing the inverse (IDCT) of the transformed sequence.

C. DCT-based Waveform Compression

Discrete Cosine Transform (DCT). The N-point DCT of a signal $x[n]$ and its inverse (IDCT) can be computed using Equations 1 and 2 respectively.

$$y[k] = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x[n] cos \left[\frac{\pi(2n+1)k}{2N} \right] \quad (1)$$

$$x[k] = \frac{y[0]}{\sqrt{N}} + \sqrt{\frac{2}{N}} \sum_{n=1}^{N-1} y[n] cos \left[\frac{\pi(2k+1)n}{2N} \right] \quad (2)$$

In COMPAQT, decompression is performed on hardware, and since it is part of the critical path during waveform synthesis, it must have low latency and a minimal hardware footprint. If a waveform with N samples is compressed with an N-point DCT, decompressing it would be complex as 1) N can vary for different waveforms, and 2) N can be very large – a two-qubit CR waveform on IBM systems can have 1000+ samples. For this reason, we use a windowed DCT scheme in which the input waveform is broken into windows of a fixed size called a *window size (WS)*. The windowed DCT scheme shall hereon be abbreviated as DCT-W while the N-point scheme shall be abbreviated as DCT-N.

The compression algorithm using the DCT is as follows: The DCT of a window of the input waveform is computed (for DCT-N, the window would be the entire waveform) after which a thresholding operation is performed, which sets samples with low magnitudes equal to zero. Finally, Run-Length Encoding (RLE) [9] is performed on the transformed window. RLE replaces all the zeros with a single codeword that contains two fields: 1) a signature identifying it as the RLE codeword and 2) the number of zeros that have been encoded to yield the compressed waveform.

Table II
IMPLEMENTED DCT VARIANTS.

Variant	Description	Hardware Complexity
DCT-N	N-point DCT, WS = length of waveform	High
DCT-W	Windowed DCT, WS = 8, 16	Moderate
int-DCT-W	Windowed integer-DCT, WS = 8, 16	Low

The compression using DCT-N has been visualized in Figure 8 for single-qubit gates. We use typical Derivative Removal by Adiabatic Gate (DRAG) waveforms used in most superconducting systems to implement single qubit
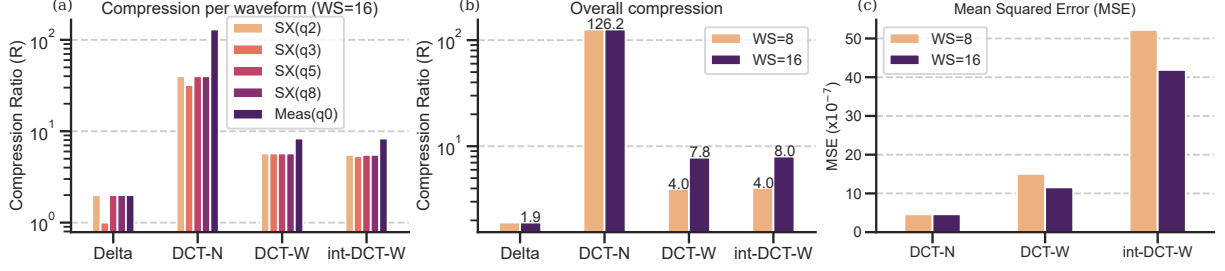
Figure 7.   (a) Compression Ratio (old size / new size) for five waveforms on IBM Guadalupe (b) Overall Compression Ratio for qft-4 (c) Average Mean Squared Error (MSE) for all waveforms of qft-4.

gates. RLE is started only when the transformed waveform after thresholding is consistently zero. This process is done separately for I and Q channels, and to simplify the decompression, the number of samples per window after compression are kept the same for both channels.

Integer-DCT. The DCT/IDCT in its original form requires a minimum number of fixed/floating-point multipliers that make hardware implementations very resource intensive. The 8-point DCT/IDCT (WS=8) requires a minimum of 11 multipliers [42]. Larger transforms need resources, with the 16-point DCT/IDCT requiring a minimum of 26 multipliers.

To make waveform compression using DCT scalable for quantum control applications, we also implemented an integer DCT consistent with the HEVC standard [72]. Since this variant of the DCT was aimed at making hardware implementations as efficient as possible, we do not use the arbitrary N-point transform. The windowed integer DCT will henceforth be abbreviated as int-DCT-W. Since the DCT coefficients are scaled by a constant factor, the input waveform is scaled by the same factor in software (this can be done in hardware too). The constant scaling factor S for an N-point DCT is given by $S = 2^{6 + \frac{log_2 N}{2}}$. All DCT variants are summarized in Table II. We used two different window sizes to evaluate the compression of waveforms using DCT.

Fidelity-Aware Thresholding. Each gate pulse is unique and a uniform threshold during compression can cause fidelity loss for some qubits. To enable high fidelity gates, we find optimal threshold for every gate pulse. We use the Algorithm 1, to tune the threshold to obtain the target gate fidelity. We observe that MSE between decompressed and

Figure 8.   Input waveform and its DCT, the vertical green line shows the start of RLE where |sample| < threshold.

uncompressed pulses are highly co-related to the gate fidelity, and we can use MSE for tuning the threshold at compile time. We can take a step further and integrate the Fidelity-Aware compression within the gate calibration loop.

Algorithm 1: Fidelity-Aware Compression

Input: W_{in} (Gate Pulse), ε (Target Error)
Output: W_{out} (Compressed Pulse)

1 **Function** Get_Compressed_Pulse(W_{in}, ε):
2   **while** $mse \leq \varepsilon$ **do**
3   Y = int-DCT(W_{in}, W_{out}) // Compute DCT
4   **for** s **in** Y **do**
5   **if** $s < threshold$ **then** s = 0
6   **end**
7   W_{out} = int-iDCT (Y)
8   mse = get_MSE (W_{in}, W_{out})
9   threshold = threshold/2
10   **if** $threshold < 10^{-6}$ **then return** -1
11   // No solution found
12   **end**
13 **return** W_{out}

D. Efficacy of DCT Compression

The DCT-based compression scheme is lossy and its effectiveness depends on two factors: 1) Compressibility of the waveforms. 2) Fidelity of decompressed waveforms. To evaluate compressibility, we measure the reduction in memory size to store all waveforms used for a benchmark circuit. For evaluating the effect on fidelity, we ran two-qubit Randomized Benchmarking [44] on various IBM machines.

Compressibility. Figure 7(a) shows the reduction ($R = old\ size/new\ size$) in required memory for five representative waveforms of qft-4 benchmark derived from [39] implemented on the IBM Guadalupe machine. Figure 7(b) shows the overall reduction in memory for qft-4. Note that measurement and 2Q gates are longer and more compressible than 1Q gates which affects the total compression ratio for qft-4. A window size of 8 has the least reduction because RLE is limited to a maximum of 8 samples at a
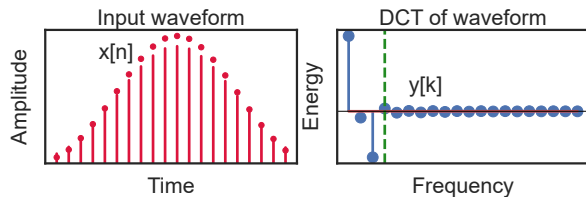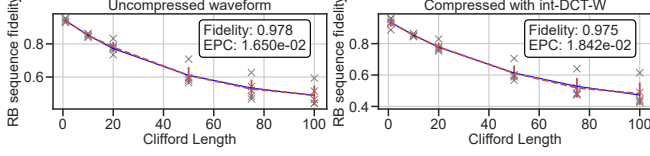
Figure 9. Randomized Benchmarking of two qubit system with the uncompressed baseline gate pulses and compressed gate pulses with int-DCT-W on IBM Guadalupe.

Figure 10. Decompression pipeline for waveforms compressed with DCT-W/int-DCT-W (WS=8).

time. Increasing the window size makes the IDCT design complex. Thus, more compression can be achieved at the cost of hardware complexity. Delta compression shows an overall reduction in size for the `qft-4` circuit but not all waveforms are compressible due to the overhead of supporting zero crossing as shown in Figure 7(a).

Impact on gate fidelity. Distortion in the waveforms can reduce gate fidelity. To understand whether compression distorts the waveforms, we compute Mean Squared error (MSE) between the original waveforms and the decompressed waveforms. Figure 7(c) shows the average MSE over all waveforms used for the `qft_n4` benchmark for three variants of DCT-based compression. The MSE lies between 10^{-5} and 10^{-7}, and it is highest (but still negligible) for the `int-DCT-W` variant since it incorporates integer approximations before the thresholding step in the algorithm.

Moreover, to test the gate fidelity, first we compress and then decompress waveforms by emulating the decompression engine design, and then we use these decompressed waveforms to drive qubits on IBM quantum computers. Figure 9 shows the RB fidelity with increasing number of Clifford gates and average Error Per Clifford (EPC) which averages the total error over the number of gates. Table III summarizes the fidelity of two-qubit Randomized Benchmarking (RB) on three IBM machines. The results show that even though DCT based compression is lossy, the decompressed waveform is similar to the original, because of which there is minimal effect on the fidelity of the gates.

Table III
RB FIDELITY FOR UNCOMPRESSED BASELINE PULSE AND COMPRESSED PULSES WITH WINDOW SIZE OF 16.

Design	2Q RB Fidelity IBM Bogota	2Q RB Fidelity IBM Guadalupe	2Q RB Fidelity IBM Hanoi
Baseline	0.980	0.978	0.987
DCT-N	0.982	0.977	0.989
DCT-W	0.983	0.976	0.986
int-DCT-W	0.983	0.975	0.988

V. COMPAQT MICROARCHITECTURE

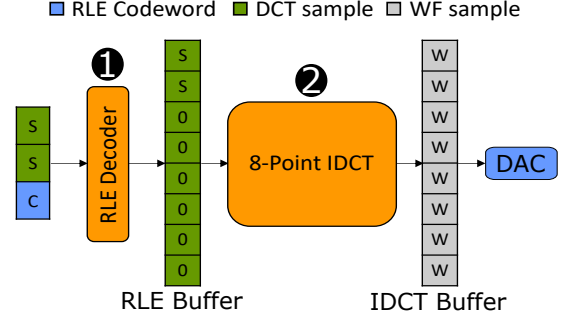In this section, we discuss the architecture of the decompression pipeline, the decompression engine, and the waveform memory. Furthermore, we will discuss adaptive decompression strategies to make COMPAQT scalable with cryogenic ASICs.

A. Decompression Pipeline Architecture

Decompression is a two-step process, as shown in Figure 10. In ❶, the codeword generated by RLE is decoded to yield all the encoded zeros and then in ❷, the IDCT of the resultant vector produces the decompressed waveform. To enable hardware efficient design, we use the windowed IDCT algorithm in COMPAQT. If compression was done using a window size of sixteen, the samples generated by the RLE decoder and IDCT would scale accordingly.

Compression reduces the waveform capacity, but, more importantly, by storing a small number of samples and expanding them in the RLE buffer, DCT based compression enables higher bandwidth. As shown in Figure 10, the three compressed waveform samples yield eight waveform samples after decompression, thereby boosting the waveform memory bandwidth. In our design, for simplicity and efficient hardware synthesis, we design compressed waveform memory with uniform width, which is determined by the size of the worst-case compressed window. For example, the width of the compressed DCT samples (green+blue squares) in Figure 10 is three samples, including the RLE codeword. Figure 11 shows the histogram of the number of samples in every window for `int-DCT-W` showing that waveforms are highly compressible. Based on an empirical analysis, the worst-case number of samples the design must cater to is thus three as shown by the histogram. The design of the decompression
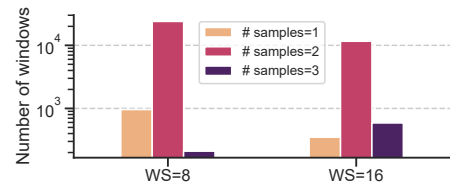

Figure 11. Histogram of the number of samples per window in the compressed waveform (including the RLE codeword) for 132 waveforms from IBM Guadalupe.
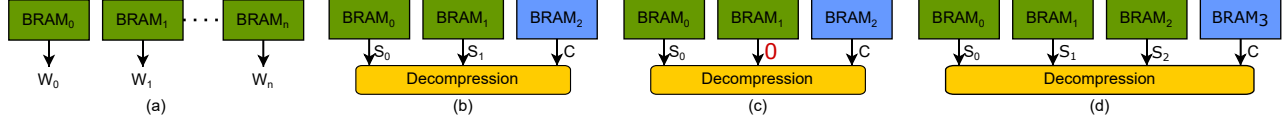
Figure 12. (a) Interleaved waveform samples in different BRAMs to boost bandwidth (b) Reduction in number of BRAMs for a waveform with maximum of three samples per window (including RLE code word) (c) Compressed window with two non-zero samples interleaved in three BRAMs (d) Interleaved memory organization for a pulse that requires four samples for a worst-case window.

pipeline will be complex if we enable each window to have a variable number of samples. In COMPAQT, we choose a uniform input buffer size which sacrifices compressibility to enable a significant performance boost.

B. Decompression Engine Design

In the DCT decompression pipeline, the RLE decoder depends on the RLE codeword to decompress all zeros and feed them to the IDCT stage. The signature in the RLE codeword can be used to identify the codeword, and if the RLE codeword specifies that c_n zeros have been encoded, the last c_n inputs of the IDCT stage can be set to 0.

A major advantage of using int-DCT-W over DCT-W is that the multipliers used for computing the inverse transform can be replaced with shift-and-add operations, drastically reducing hardware complexity and improving performance. We encourage readers to peruse existing designs for DCT-W [10], [19], [66] and int-DCT-W [12], [68], [83] (IDCT circuits are simply the reverse of DCT circuits).

Table IV
HARDWARE RESOURCES NEEDED FOR THE IDCT ENGINE OF DCT-W AND INT-DCT-W.

Variant	Window size(WS)	Multipliers	Adders	Shifters
DCT-W	8	11	29	0
int-DCT-W		0	50	26
DCT-W	16	26	81	0
int-DCT-W		0	186	128

Table IV summarizes the differences in requirements for implementing the IDCT in hardware for both DCT-W and int-DCT-W with a window size of eight. The DCT-W design is based on Loeffler's algorithm and is the least expensive implementation [42]. For int-DCT-W, the multiplications are converted to shift-and-add operations [68]. This optimization enables the IDCT engine to have a constant latency of one clock cycle.

C. Banked Compressed Waveform Memory

Typically, the RFSoC FPGA clock is slower ($\approx$0.3 GHz) than the DAC sampling frequency ($\approx$6 GHz). As a result, even in the baseline uncompressed systems, feeding data to the DACs at the peak sampling rate can become challenging. To solve this problem, waveform samples for a single

waveform are interleaved in multiple BRAMs. By banking waveform memory, we can compensate for the difference in frequencies between the FPGA and the DAC as shown in Figure 12(a). However, this approach is not scalable and eventually hits a BRAM bandwidth wall due to the finite number of BRAMs available. In COMPAQT, we use memory interleaving with compressed waveforms, as fewer BRAMs are needed for a waveform with the decompression pipeline.

Figure 11 showed that regardless of the window size, the number of samples in the compressed waveform is ≤ 3 for int-DCT-W. Figure 12(b) is an example showing how if the waveform samples were originally interleaved in multiple BRAMs, the number of BRAMs needed after compression with WS=16 would be just three (if WS=8 was used, the number of BRAMs would have been six, since two 8-point IDCT modules would be needed). The decompression block in the figure is the pipeline shown in Figure 10 and the results after decompression can be stored on distributed memory in the RFSoC until they are used. Figure 12(c) shows how the input to the decompression engine is zero if the number of samples in the window is less than the maximum number and Figure 12(d) show a case where the maximum number of samples per window is four.

Table V
NUMBER OF QUBITS THAT CAN BE SUPPORTED BY AN FPGA BASED DESIGN WITH UNCOMPRESSED WAVEFORM MEMORY AND A DESIGN WITH COMPRESSED MEMORY USING INT-DCT-W FOR DIFFERENT WINDOW SIZES.

	Uncompressed	WS=8	WS=16
Number of qubits (normalized)	1	2.66	5.33

We reduce the number of concurrent reads from BRAMs required to control one qubit. As a result, RFSoCs can drive significantly more qubits. Table V shows how different window sizes increase the number of qubits that can be controlled by an RFSoC based controller. This gain is independent of the ratio of FPGA clock frequency and DAC sampling rate as long as the clock frequency ratio is a multiple of the window size. If the ratio is not a multiple of the window, then the gain will be slightly lower. As an example, consider the case where the ratio is 6x. For such a system, uncompressed waveforms would require interleaving of samples between six BRAMs per waveform. If compressed
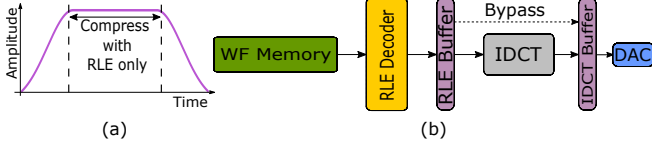
Figure 13. (a) A flat-top waveform – the constant part of the waveform can be compressed effectively with RLE only; (b) Adaptive decompression – the IDCT engine can be bypassed for such flat-top waveforms to save power.

Benchmark	Description	# qubits	Purpose	# CNOTs
swap	Swap gate	2	Fidelity	3
toffoli	Toffoli gate	3	Fidelity	12
qft-4	Quantum Fourier Transform	4	Fidelity	27
adder-4	4-bit Full Adder	4	Fidelity	33
bv-5	Bernstein Vazirani	6	Fidelity	2
qaoa-6	Optimization Algorithm	6	Fidelity	142
qaoa-8a	Optimization Algorithm	8	Fidelity	76
qaoa-8b	Optimization Algorithm	8	Fidelity	113
qaoa-10	Optimization Algorithm	10	Fidelity	138
surface-17	Surface Code Patch (d=3)	17	Scalability	-
surface-25	Surface Code Patch (d=3)	25	Scalability	-

waveforms with WS=8 were used, the number of BRAMs needed per waveform would be three (assuming the same distribution as shown in Figure 11). The gain in the number of qubits for such a system would be 2x, which is slightly less than the case where the ratio is 8x. The ratio between the DAC and FPGA was 16x in QICK due to which it can theoretically support about 36 qubits. Using COMPAQT with WS=8, number of qubits can be increased to about 95 qubits, and for WS=16, we can drive 191 qubits concurrently. Although the availability of DACs and the limit of Frequency Division Multiplexing (FDM) may result in lesser qubits, the theoretical bounds of the control computer will no longer bottleneck the scalability of the FPGA platforms.

D. Adaptive Decompression with COMPAQT

A diverse set of waveforms are used to manipulate qubits. Some gate waveforms have more structure than others. For example, the flat-top waveform as shown in Figure 13(a) is commonly used to perform multi-qubit gates [8], [14], [61], [64]. Optimizing the decompression pipeline for such waveforms can help reduce power dissipation for ASIC designs even further. The constant period of the waveform just repeats the same value for a long duration, which can be represented by a single RLE word that can be decoded and fed directly into the buffer preceding the DAC as shown in Figure 13(b). This saves power since both the memory and IDCT engine will not be used during this period. Once the flat period is over, the IDCT-driven decompression can resume. The algorithm for adaptive decompression will treat the constant period of the waveform as a single window rather than dividing it into multiple windows (depending on the window size).

VI. METHODOLOGY

Benchmarks. The benchmarks used for evaluating compressed waveforms are listed in Table VI. The surface code benchmarks [75] were selected to evaluate the effect on scalability when compressed waveforms are used, and the fidelity benchmarks were selected to complement the RB experiments summarized in Section IV. Benchmarks such as qft-4, adder-4, and qaoa-6 include sequences of non-Clifford gates that are not used in RB experiments. Larger benchmarks using more qubits were not used since they are prone to higher variability in the results.

Software System. The compression module was developed using Python and integrated with Qiskit and Qiskit Pulse [3], [5]. DCT-N and DCT-W were adapted from SciPy [65] while int-DCT-W were developed based on the HEVC transform [72]. The standard Qiskit transpiler was used to transpile the fidelity benchmarks for the target system.

Control Hardware. The hardware overhead of using DCT based compression on RFSoCs was evaluated using Xilinx's Vivado Design Suite. The QICK [70][2] was synthesized as a baseline design on the zc7u7ev SoC and the int-DCT-W IDCT engine for different window sizes was implemented using Verilog and integrated with QICK. We validated the decompression pipeline integrated with QICK to ensure functional correctness. Furthermore, we measure the impact of integrated IDCT engine on clock frequency and resource utilization of overall control processor. To evaluate our ASIC design, we use Synopsys Design Compiler with the 40nm TSMC CLN40G cell library to estimate the power dissipation of IDCT engine, along with the Destiny cache model [58] which integrates CACTI [13] to estimate power dissipation of SRAM based waveform memory.

Quantum Hardware. RB and benchmark fidelity experiments were performed on real quantum hardware. RB experiments were performed on the 5-qubit IBM Bogota, 16-qubit IBM Guadalupe, and 27-qubit IBM Hanoi [29]. Benchmark fidelities with compressed and uncompressed waveforms were evaluated on IBM Guadalupe and Toronto hardware. We evaluate fidelity (F) by computing Total Variational Distance (TVD) between ideal (P) and experimental (Q) output distributions as shown in Equation 3.

$$F(P,Q) = 1 - TVD(P,Q) \qquad (3)$$

VII. EVALUATIONS

In this section, we evaluate 1) the impact on benchmark fidelity when compressed waveforms are used, 2) the scalability of RFSoC and ASIC controllers for important applications
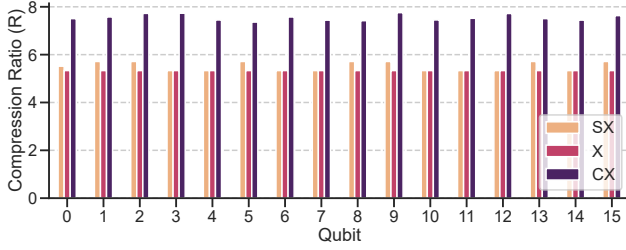
[2]https://github.com/openquantumhardware/qick

Figure 14. Compression ratios of the basis gates for all 16 qubits of IBM Guadalupe using int-DCT-W with WS=16.

Figure 15. COMPAQT fidelity normalized to baseline fidelity on IBM Guadalupe. The baseline fidelities are annotated beneath the benchmark names.

like Quantum Error Correction (QEC), and 3) the execution time for compressing waveforms at compile time.

A. Gate Compressibility with COMPAQT

We evaluated the compressibility of the gate pulses using COMPAQT for five IBM quantum computers. Table VII reports the minimum, average, and maximum compressibility of a control pulse used on IBM hardware. Despite the diversity in the pulse shapes, we observe that all control pulses are highly compressible. For example, as shown in Figure 14, on average, we can compress the control pulses by more than 5x for each qubit device on the IBM-Guadalupe. Note that in Figure 14, the compression ratios for CNOT gates are averaged over all CNOT gates that can be performed using a qubit device, whereas Table VII reports the minimum and maximum compression ratio for an individual gate pulse on the IBM hardware. We observe that on all platforms the SX-gate has the lowest compression ratio of 5.33 with our design.

Table VII
MINIMUM, MAXIMUM, AND AVERAGE COMPRESSION RATIOS WITH
INT-DCT-W (WS=16).

Machine	Compression Ratio (R)		
	Min.	Max.	Avg.
IBM Toronto	5.33	8.11	6.49
IBM Montreal	5.33	8.02	6.45
IBM Mumbai	5.33	8.05	6.47
IBM Guadalupe	5.33	8.02	6.48
IBM Lima	5.33	7.92	6.33

B. Gate Fidelity with COMPAQT

To evaluate whether compressed waveforms degrade the fidelity of quantum circuits, we run circuits for 80K shots with and without waveform compression and compute the normalized fidelity by taking the ratio of the fidelities of COMPAQT and the uncompressed baseline.

Figure 15 shows the normalized benchmark fidelities for WS=8 and WS=16 with int-DCT-W on IBM Guadalupe. Norm. fidelity close to one means both the baseline and COMPAQT have identical fidelities, indicating no degradation
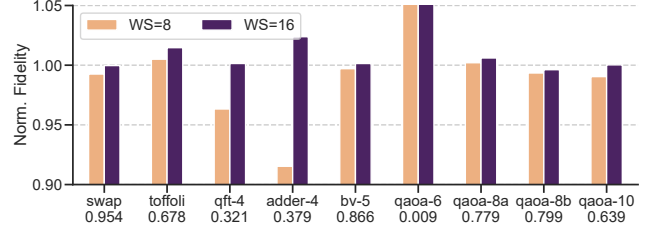
due to lossy compression. For window size of WS=16 we see no fidelity degradation but for the WS=8, there are significant fidelity losses for some benchmarks due to distortions introduced at the boundaries of consecutive windows. These distortions can be reduced by using overlapping windows to compress the waveform. Note that in some cases normalized fidelity is slightly greater than one, indicating COMPAQT is more reliable. This can be attributed to the variability in the experiments. Note that the fidelity for the QAOA benchmarks was determined by computing the normalized fidelity [27], [43]. We observe that the fidelity degradation is less than 0.5% for the largest QAOA benchmark using 10 qubits with a window size of 16.

C. Scalability of COMPAQT with RFSoC

To evaluate the scalability of COMPAQT on RFSoC platforms, we evaluate the maximum clock frequency, resource utilization and maximum number of concurrent gates that can be executed for key benchmark circuits.

Clock Frequency. We use QICK [70] as the baseline design with a maximum frequency of 294 MHz. For COMPAQT, the maximum achievable frequency drops by more than 33% to 195 MHz for the pipelined DCT-W engine, as shown in Figure 16. This degradation results from complex design and long critical path due to multipliers. In comparison, the unpipelined int-DCT-W design introduced a worst-case degradation of 10%, which can be pipelined to enable a design with no clock frequency degradation. These results can be scaled to arbitrary number of qubits as to control multiple qubits the individual qubit control block will be instantiated in parallel having a little to no effect on the critical path.

Resource Utilization. Table VIII shows the usage of critical FPGA resources like Look-Up Tables (LUTs) and Flip-Flops (FFs) of the baseline design (QICK) and a single IDCT engine for int-DCT-W with different window sizes. The baseline design can control a single qubit, but it includes components like the AXI interface that will not scale linearly with the number of qubits. The resources are not enough to accommodate 100+ qubits. The overall resource usage per module will be lower than 1% for RFSoCs, as they have

Design	LUT Utilization	FF Utilization
Baseline	3386 (1.4%)	6448 (1.4%)
`int-DCT-W` (WS=8)	601 (0.26%)	266 (0.05%)
`int-DCT-W` (WS=16)	1954 (0.85%)	671 (0.15%)
`int-DCT-W` (WS=32)	9063 (3.93%)	1197 (0.26%)

Figure 17. Increasing the number of logical qubits that can be supported; (a) The peak concurrent operations for a d=3 code; (b) The maximum logical qubits that can be supported by different Xilinx RFSoC based designs.

significantly more resources than the device used for this evaluation (the RFSoC used by QICK has 400K+ LUTs and 800K+ FFs). Compared to BRAMs, LUTs and FFs are more readily available on FPGA and so using a compressed memory architecture trades memory units (BRAMs) with system logic (compute) per qubit in FPGAs. The design of the IDCT hardware used for these evaluations can be further optimized by pipelining.

Although increasing window size can improve compression efficiency, using a larger window size such as WS=32 with `int-DCT-W` uses significantly more resources than WS=16, which makes it a sub-optimal design as a single iDCT engine uses more than 4% of the total LUTs. In summary, int-DCT with WS=32 reduces clock frequency and adds complexity, limiting the scalability.

Scalability of QEC Experiments. Demonstrating a small set of logical qubits that run quantum error correction is a crucial milestone in achieving fault-tolerant quantum computation. Surface code is a leading candidate for implementing QEC [20] and its effectiveness have been demonstrated experimentally for low-distance codes [4], [36], [45], [82]. As systems scale and add more qubits to move towards fault-tolerant systems, the number of logical qubits that a single controller can support will thus be an important metric.

Figure 17(a) shows the maximum number of concurrent operations at any given time during the syndrome generation cycle in two variants of a distance three surface code patch. More than 80% of the physical qubits to create one logical
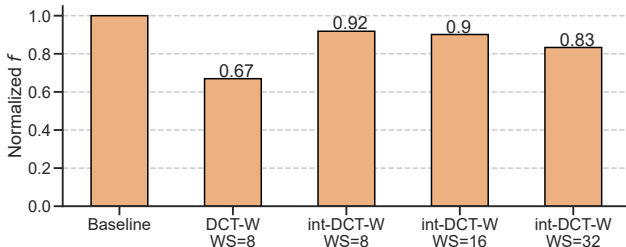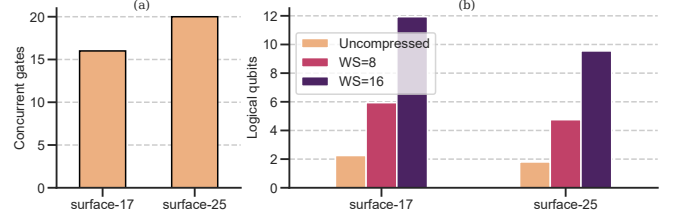
qubit are driven concurrently, which makes it important for a controller hardware to be able to support concurrency in as many qubits as possible. Figure 17(b) show that COMPAQT can control 5x more logical qubits compared to an uncompressed baseline. Similarly, for ASIC controllers, using compressed waveform memory can reduce memory power dissipation by more than 2x which can allow more logical qubits to be controlled.

D. Scalability of COMPAQT with ASICs

The intrinsic BRAM bandwidth on FPGA-fabric limits their scalability. Compressing waveform memory can help improve such systems. However, unlike BRAMs, SRAMs used in ASICs can operate at much higher frequencies and deliver significantly more bandwidth. However, for ASICs designed to work at cryogenic temperatures, it is crucial for the overall architecture to be as power efficient as possible due to tiny power budget at 4K. While the cooling capacity of dilution refrigerators will probably have to be expanded in the future to accommodate more qubits, the power budget will still be limited and minimizing the power dissipated by digital components like waveform memory can help optimize other components of qubit control that are critical for high-fidelity gates. For example, 50% of the total power (23mW) in the cryogenic chip presented in [21] was dissipated by digital components (including the digital frontend of the DACs).

In general, the energy dissipated per access by memory increases with the size of the memory. Since SRAMs can operate at much higher frequencies than FPGA BRAMs, interleaving memory samples between multiple banks is not

Figure 16. Degradation of clock frequency compared to baseline for different decompressed waveform memory architectures.
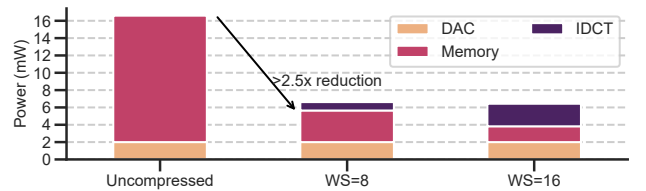

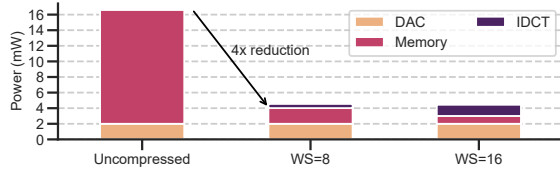Figure 18. Power dissipated by the cryogenic controller with the uncompressed and compressed memory.

Figure 19. Power consumption for flat-top waveform with uncompressed baseline and COMPAQT using adaptive decompression with WS=8 and WS=16.

Figure 20. Average time to compress a gate waveform using int-DCT-W compression.

necessary and so every window can have variable number of compressed samples since every sample will be fetched sequentially and decompression will start every time an RLE codeword is detected. By compressing waveforms, SRAMs can be made smaller and fewer accesses are required to read the same waveform when compared to an uncompressed waveform memory. Figure 18 shows how memory power is reduced by more than 2.5x when compressed waveforms are used for a CMOS based cryogenic controller. Memory power was estimated with the Destiny model [58]. The DAC power of 2mW was added as a reference and the power dissipated by the `int-DCT-W` IDCT engine for corresponding window sizes was estimated via Synopsys Design Compiler with a TSMC 40nm cell library – actual power dissipated by the IDCT engine can be even lower, the data shown here motivates the use of compression as the overhead of using the IDCT engine does not overshadow the decrease in memory power. Using the compressed waveform architecture can thus reduce the overall power dissipated by the waveform memory by at least 3x.

Moreover, we can further reduce power consumption using adaptive decompression as shown in Figure 19. Since memory is accessed only during the rise and fall ramps of the waveform, memory power is reduced over the entire period along with the power dissipated by the IDCT engine, yielding an additional 1.5x power savings. This gain will vary with the duration of the flat-top waveform, a 100ns waveform was used for this evaluation.

E. Software Overhead of COMPAQT

To evaluate the overhead of decompression, we measure the compression latency of the COMPAQT compiler module. Figure 20 shows the average time to compress a waveform using `int-DCT-W` for three IBM machines. Since waveforms need only be compressed at the end of every calibration cycle, the overhead of compressing waveforms is negligible compared to the time taken to calibrate all qubits – regular calibration of the Sycamore chip lasts four hours [6]. As a result, compression has little or no overhead on the system as a whole and does not affect system performance metrics such as repetition rate and Circuit Level Operations Per Second (CLOPS) [15].
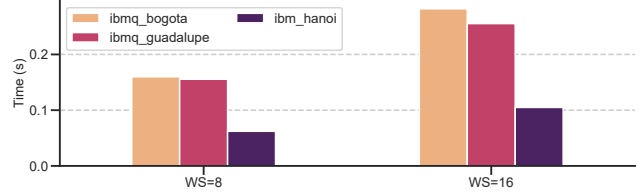
VIII. RELATED WORK

Efficient qubit control is crucial to scale quantum computers. While most control platforms use discrete FPGAs and analog ICs, we expect RFSoC-based and custom ASIC-based solutions to grow in the future.

RFSoC controllers. We use QICK as the baseline, which is an open source RFSoC-based qubit control framework [70], that reduces discrete analog components by using direct digital synthesis. Similarly, recent works from a different group of researchers also demonstrate the effectiveness of RFSoCs in controlling superconducting qubits [25], [55].

ASIC controllers. Studies on ASIC controllers for qubit control have mainly been focused on developing cryogenic controllers. Fully/semi-autonomous cryogenic chips were shown by [17], [21], [32] that integrated both digital and analog blocks on a single ASIC. These prototype chips are designed to control a small number of qubits. Other CMOS based ASICs were presented in [7], [31], [33], [54], [56], [77], [80]. A complete system-level study for a control processor using SFQ pulses was presented in [30]. Other controller architectures that use SFQ pulses were described in [38], [40], [41], [46], [47].

Control architectures. Many prior works are focused on FPGA/RFSoC/ASIC controllers. Fu et al. presented an FPGA-based microarchitecture for controlling a transmon qubit [22], [24]. Scalable architectures for QECC applications were presented in [16], [74], whereas ISA specific works for qubit control include [23] which introduced an executable Quantum ISA, [11] which compared their proposed scalar and vector instruction sets with other ISAs, and [37] which recommended using a small number of expressive two-qubit gates. At the same time, several works focused on microarchitectural challenges in building a quantum computer with hundreds of thousands of qubits [50], [51], [52].

IX. DISCUSSION

Qubit control pulses have a tight footprint in the frequency domain. Any spurious frequencies in the control pulse can introduce control error, crosstalk, and leakage errors. As a result, due to the tiny footprint in the frequency domain, control pulses can be compressed and stored efficiently. To the best of our knowledge, COMPAQT is the first to leverage

this fundamental property to build a scalable and efficient control using compressed waveform memories and compiler-guided compression that ensures high fidelity. Moreover, these insights can be used for designing single flux quantum (SFQ) based qubit control, in which on-chip memory is limited to tens of kilobytes [30].

Enabling Novel Design Tradeoffs. We expose new tradeoffs to enable scalable qubit control. A key novelty of the proposed microarchitecture is the hardware software co-design of IDCT engine that offers high compressibility without reducing the gate fidelity. To achieve this, we designed a discrete cosine transform-based compression scheme that uses only integer arithmetic and eliminates the use of complex multiplier circuits. However, naive approximations in the DCT engine can result in distortions in certain gate pulses. To avoid such corner-cases, we propose compiler-guided compression to navigate the trade-off between compressibility and fidelity. Our design enables the lightweight design of the IDCT engine to ensure efficient synthesis on current RFSoC-based control computers. Moreover, we empirically show that COMPAQT can support 5x more qubits compared to the uncompressed baseline with minimal degradation in gate and circuit fidelity.

Impact on Future Designs. To understand the applicability of our insights on future qubit architectures, we evaluate the compressibility of control pulses for current and emerging qubit technologies. Table IX shows that control pulses are compressible not only for mainstream transmon qubits using single and two qubit gates but also complex control pulses along with the control pulses for emerging fluxonium qubits. To enable fault-tolerant quantum computers, we must focus on microarchitectural challenges in building control hardware that can support thousands of qubits. To that end, we propose COMPAQT, which can help mitigate the bottlenecks in the qubit control pipeline by leveraging compressibility of control pulses.

Table IX
COMPRESSION RATIOS (R) FOR COMPLEX GATE PULSES FOR TRANSMON AND SINGLE QUBIT GATES FOR EMERGING FLUXONIUM USING INT-DCT-W WITH WS=16.

Device Type	Gate	Description	R
Transmon	`iToffoli`	Three Qubit Gate Pulse [34]	8.32
Transmon	`Toffoli`	Three Qubit Gate Pulse [81]	5.31
Transmon	`CCZ`	Three Qubit Gate Pulse [81]	5.59
Fluxonium	$X, \frac{X}{2}, \frac{Z}{2}, \frac{Y}{2}$	Single Qubit Gate Pulse [59]	7.2

X. CONCLUSIONS

As qubits increase beyond the 100-qubit mark, scaling the control hardware with brute-force methods will be infeasible. Integration using RFSoCs and cryogenic ASICs will be required to enable scale qubit control hardware. However, in this paper, we show that the internal memory bandwidth used for playing waveforms in RFSoC-based systems becomes a bottleneck. Furthermore, for controller ASICs, the power dissipated by the waveform memory is a significant portion of the total power and optimizing the memory architecture will enable better scalability in such systems. To solve these problems with waveform memory in different types of controllers, we present COMPAQT, a compressed waveform memory architecture that compresses waveforms in software before they are loaded on the controller. In hardware, waveforms are decompressed just before they are needed. Our results show that for RFSoC-based systems, the number of qubits that can be supported increases by more than 5x compared to the baseline. For CMOS ASICs, the reduced number of accesses required per waveform reduces power dissipation by more than 2.5x. Compressed waveform memory has minimal effect on the gate fidelity, as shown on real quantum hardware, while enabling more scalable controllers.

REFERENCES

[1] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[2] A. R. Alameldeen and D. A. Wood, "Adaptive cache compression for high-performance processors," in *Proceedings. 31st Annual International Symposium on Computer Architecture, 2004.* IEEE, 2004, pp. 212–223.

[3] T. Alexander, N. Kanazawa, D. J. Egger, L. Capelluto, C. J. Wood, A. Javadi-Abhari, and D. C. McKay, "Qiskit pulse: programming quantum computers through the cloud with pulses," *Quantum Science and Technology*, vol. 5, no. 4, p. 044006, Aug. 2020. [Online]. Available: https://doi.org/10.1088/2058-9565/aba404

[4] and Zijun Chen, K. J. Satzinger, J. Atalaya, A. N. Korotkov, A. Dunsworth, D. Sank, C. Quintana, M. McEwen, R. Barends, P. V. Klimov, S. Hong, C. Jones, A. Petukhov, D. Kafri, S. Demura, B. Burkett, C. Gidney, A. G. Fowler, A. Paler, H. Putterman, I. Aleiner, F. Arute, K. Arya, R. Babbush, J. C. Bardin, A. Bengtsson, A. Bourassa, M. Broughton, B. B. Buckley, D. A. Buell, N. Bushnell, B. Chiaro, R. Collins, W. Courtney, A. R. Derk, D. Eppens, C. Erickson, E. Farhi, B. Foxen, M. Giustina, A. Greene, J. A. Gross, M. P. Harrigan, S. D. Harrington, J. Hilton, A. Ho, T. Huang, W. J. Huggins, L. B. Ioffe, S. V. Isakov, E. Jeffrey, Z. Jiang, K. Kechedzhi, S. Kim, A. Kitaev, F. Kostritsa, D. Landhuis, P. Laptev, E. Lucero, O. Martin, J. R. McClean, T. McCourt, X. Mi, K. C. Miao, M. Mohseni, S. Montazeri, W. Mruczkiewicz, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Newman, M. Y.

Niu, T. E. O'Brien, A. Opremcak, E. Ostby, B. Pató, N. Redd, P. Roushan, N. C. Rubin, V. Shvarts, D. Strain, M. Szalay, M. D. Trevithick, B. Villalonga, T. White, Z. J. Yao, P. Yeh, J. Yoo, A. Zalcman, H. Neven, S. Boixo, V. Smelyanskiy, Y. Chen, A. Megrant, and J. Kelly, "Exponential suppression of bit or phase errors with cyclic error correction," *Nature*, vol. 595, no. 7867, pp. 383–387, Jul. 2021. [Online]. Available: https://doi.org/10.1038/s41586-021-03588-y

[5] M. S. ANIS, H. Abraham, AduOffei, R. Agarwal, G. Agliardi, M. Aharoni, I. Y. Akhalwaya, G. Aleksandrowicz, T. Alexander, M. Amy, S. Anagolum, E. Arbel, A. Asfaw, A. Athalye, A. Avkhadiev, C. Azaustre, A. Banerjee, S. Banerjee, W. Bang, A. Bansal, P. Barkoutsos, A. Barnawal, G. Barron, G. S. Barron, L. Bello, Y. Ben-Haim, D. Bevenius, D. Bhatnagar, A. Bhobe, P. Bianchini, L. S. Bishop, C. Blank, S. Bolos, S. Bopardikar, S. Bosch, S. Brandhofer, Brandon, S. Bravyi, N. Bronn, Bryce-Fuller, D. Bucher, A. Burov, F. Cabrera, P. Calpin, L. Capelluto, J. Carballo, G. Carrascal, A. Carriker, I. Carvalho, A. Chen, C.-F. Chen, E. Chen, J. C. Chen, R. Chen, F. Chevallier, R. Cholarajan, J. M. Chow, S. Churchill, C. Claus, C. Clauss, A. Clothier, R. Cocking, R. Cocuzzo, J. Connor, F. Correa, A. J. Cross, A. W. Cross, S. Cross, J. Cruz-Benito, C. Culver, A. D. Córcoles-Gonzales, N. D, S. Dague, T. E. Dandachi, A. N. Dangwal, J. Daniel, M. Daniels, M. Dartiailh, A. R. Davila, F. Debouni, A. Dekusar, A. Deshmukh, M. Deshpande, D. Ding, J. Doi, E. M. Dow, E. Drechsler, E. Dumitrescu, K. Dumon, I. Duran, K. EL-Safty, E. Eastman, G. Eberle, A. Ebrahimi, P. Eendebak, D. Egger, A. Espiricueta, M. Everitt, D. Facoetti, Farida, P. M. Fernández, S. Ferracin, D. Ferrari, A. H. Ferrera, R. Fouilland, A. Frisch, A. Fuhrer, B. Fuller, M. GEORGE, J. Gacon, B. G. Gago, C. Gambella, J. M. Gambetta, A. Gammanpila, L. Garcia, T. Garg, S. Garion, T. Gates, L. Gil, A. Gilliam, A. Giridharan, J. Gomez-Mosquera, Gonzalo, S. de la Puente González, J. Gorzinski, I. Gould, D. Greenberg, D. Grinko, W. Guan, J. A. Gunnels, N. Gupta, J. M. Günther, M. Haglund, I. Haide, I. Hamamura, O. C. Hamido, F. Harkins, A. Hasan, V. Havlicek, J. Hellmers, Ł. Herok, S. Hillmich, H. Horii, C. Howington, S. Hu, W. Hu, J. Huang, R. Huisman, H. Imai, T. Imamichi, K. Ishizaki, Ishwor, R. Iten, T. Itoko, A. Javadi, A. Javadi-Abhari, W. Javed, M. Jivrajani, K. Johns, S. Johnstun, Jonathan-Shoemaker, JosDenmark, JoshDumo, J. Judge, T. Kachmann, A. Kale, N. Kanazawa, J. Kane, Kang-Bae, A. Kapila, A. Karazeev, P. Kassebaum, J. Kelso, S. Kelso, V. Khanderao, S. King, Y. Kobayashi, A. Kovyrshin, R. Krishnakumar, V. Krishnan, K. Krsulich, P. Kumkar, G. Kus, R. LaRose, E. Lacal, R. Lambert, J. Lapeyre, J. Latone, S. Lawrence, C. Lee, G. Li, J. Lishman, D. Liu, P. Liu, Y. Maeng, S. Maheshkar, K. Majmudar, A. Malyshev, M. E. Mandouh, J. Manela, Manjula, J. Marecek, M. Marques, K. Marwaha, D. Maslov, P. Maszota, D. Mathews, A. Matsuo, F. Mazhandu, D. McClure, M. McElaney, C. McGarry, D. McKay, D. McPherson, S. Meesala, D. Meirom, C. Mendell, T. Metcalfe, M. Mevissen, A. Meyer, A. Mezzacapo, R. Midha, Z. Minev, A. Mitchell, N. Moll, A. Montanez, G. Monteiro, M. D. Mooring, R. Morales, N. Moran, D. Morcuende, S. Mostafa, M. Motta, R. Moyard, P. Murali, J. Müggenburg, D. Nadlinger, K. Nakanishi, G. Nannicini, P. Nation, E. Navarro, Y. Naveh, S. W. Neagle, P. Neuweiler, A. Ngoueya, J. Nicander, Nick-Singstock, P. Niroula, H. Norlen, NuoWenLei, L. J. O'Riordan, O. Ogunbayo, P. Ollitrault, T. Onodera, R. Otaolea, S. Oud, D. Padilha, H. Paik, S. Pal, Y. Pang, A. Panigrahi, V. R.

Pascuzzi, S. Perriello, E. Peterson, A. Phan, F. Piro, M. Pistoia, C. Piveteau, J. Plewa, P. Pocreau, A. Pozas-Kerstjens, R. Pracht, M. Prokop, V. Prutyanov, S. Puri, D. Puzzuoli, J. Pérez, Quintiii, R. I. Rahman, A. Raja, R. Rajeev, N. Ramagiri, A. Rao, R. Raymond, O. Reardon-Smith, R. M.-C. Redondo, M. Reuter, J. Rice, M. Riedemann, D. Risinger, M. L. Rocca, D. M. Rodríguez, RohithKarur, B. Rosand, M. Rossmannek, M. Ryu, T. SAPV, A. Saha, A. Ash-Saki, M. Sandberg, H. Sandesara, R. Sapra, H. Sargsyan, A. Sarkar, N. Sathaye, B. Schmitt, C. Schnabel, Z. Schoenfeld, T. L. Scholten, E. Schoute, M. Schulterbrandt, J. Schwarm, J. Seaward, Sergi, I. F. Sertage, K. Setia, F. Shah, N. Shammah, R. Sharma, Y. Shi, J. Shoemaker, A. Silva, A. Simonetto, D. Singh, P. Singh, P. Singkanipa, Y. Siraichi, Siri, J. Sistos, I. Sitdikov, S. Sivarajah, M. B. Sletfjerding, J. A. Smolin, M. Soeken, I. O. Sokolov, I. Sokolov, SooluThomas, Starfish, D. Steenken, M. Stypulkoski, A. Suau, S. Sun, K. J. Sung, M. Suwama, O. Słowik, H. Takahashi, T. Takawale, I. Tavernelli, C. Taylor, P. Taylour, S. Thomas, M. Tillet, M. Tod, M. Tomasik, E. de la Torre, J. L. S. Toural, K. Trabing, M. Treinish, D. Trenev, TrishaPe, F. Truger, G. Tsilimigkounakis, D. Tulsi, W. Turner, Y. Vaknin, C. R. Valcarce, F. Varchon, A. Vartak, A. C. Vazquez, P. Vijaywargiya, V. Villar, B. Vishnu, D. Vogt-Lee, C. Vuillot, J. Weaver, J. Weidenfeller, R. Wieczorek, J. A. Wildstrom, J. Wilson, E. Winston, WinterSoldier, J. J. Woehr, S. Woerner, R. Woo, C. J. Wood, R. Wood, S. Wood, J. Wootton, M. Wright, B. Yang, D. Yeralin, R. Yonekura, D. Yonge-Mallo, R. Young, J. Yu, L. Yu, C. Zachow, L. Zdanski, H. Zhang, C. Zoufal, aeddins ibm, alexzhang13, b63, bartek bartlomiej, bcamorrison, brandhsn, catornow, charmerDark, deeplokhande, dekel.meirom, dime10, ehchen, fanizzamarco, fs1132429, gadial, galeinston, georgezhou20, georgios ts, gruu, hhorii, hykavitha, itoko, jliu45, jscott2, klinvill, krutik2966, ma5x, michelle4654, msuwama, ntgiwsvp, ordmoj, sagar pahwa, pritamsinha2304, ryancocuzzo, saswati qiskit, septembrr, sethmerkel, shaashwat, sternparky, strickroman, tigerjack, tsura crisaldo, welien, willhbang, yang.luh, and M. Čepulkovskis, "Qiskit: An open-source framework for quantum computing," 2021.

[6] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven, and J. M. Martinis, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, Oct. 2019. [Online]. Available: https://doi.org/10.1038/s41586-019-1666-5

[7] J. C. Bardin, T. White, M. Giustina, K. J. Satzinger, K. Arya, P. Roushan, B. Chiaro, J. Kelly, Z. Chen, B. Burkett, Y. Chen, E. Jeffrey, A. Dunsworth, A. Fowler, B. Foxen, C. Gidney, R. Graff, P. Klimov, J. Mutus, M. J.

McEwen, M. Neeley, C. J. Neill, E. Lucero, C. Quintana, A. Vainsencher, H. Neven, J. Martinis, T. Huang, S. Das, D. T. Sank, O. Naaman, A. E. Megrant, and R. Barends, "Design and Characterization of a 28-nm Bulk-CMOS Cryogenic Quantum Controller Dissipating Less Than 2 mW at 3 K," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 11, pp. 3043–3060, Nov. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8865458/

[8] M. Boissonneault, J. M. Gambetta, and A. Blais, "Dispersive regime of circuit qed: Photon-dependent qubit dephasing and relaxation rates," *Physical Review A*, vol. 79, no. 1, p. 013819, 2009.

[9] S. D. Bradley, "Optimizing a scheme for run length encoding," *Proceedings of the IEEE*, vol. 57, no. 1, pp. 108–109, 1969.

[10] K. Bukhari, G. Kuzmanov, and S. Vassiliadis, "Dct and idct implementations on different fpga technologies," in *Proceedings of ProRISC*. Citeseer, 2002, pp. 232–235.

[11] A. Butko, G. Michelogiannakis, S. Williams, C. Iancu, D. Donofrio, J. Shalf, J. Carter, and I. Siddiqi, "Understanding quantum control processor capabilities and limitations through circuit characterization," in *2020 International Conference on Rebooting Computing (ICRC)*. IEEE, Dec. 2020. [Online]. Available: https://doi.org/10.1109/icrc2020.2020.00011

[12] S. Chatterjee and K. Sarawadekar, "An optimized architecture of hevc core transform using real-valued dct coefficients," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 12, pp. 2052–2056, 2018.

[13] K. Chen, S. Li, N. Muralimanohar, J. H. Ahn, J. B. Brockman, and N. P. Jouppi, "Cacti-3dd: Architecture-level modeling for 3d die-stacked dram main memory," in *2012 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2012, pp. 33–38.

[14] J. M. Chow, A. D. Córcoles, J. M. Gambetta, C. Rigetti, B. R. Johnson, J. A. Smolin, J. R. Rozen, G. A. Keefe, M. B. Rothwell, M. B. Ketchen, and M. Steffen, "Simple all-microwave entangling gate for fixed-frequency superconducting qubits," *Physical Review Letters*, vol. 107, no. 8, Aug. 2011. [Online]. Available: https://doi.org/10.1103/physrevlett.107.080502

[15] "Driving Quantum Performance," https://research.ibm.com/blog/circuit-layer-operations-per-second.

[16] P. Das, C. A. Pattison, S. Manne, D. Carmean, K. Svore, M. Qureshi, and N. Delfosse, "A scalable decoder microarchitecture for fault-tolerant quantum computing," 2020.

[17] J. P. G. V. Dijk, B. Patra, S. Subramanian, X. Xue, N. Samkharadze, A. Corna, C. Jeon, F. Sheikh, E. Juarez-Hernandez, B. P. Esparza, H. Rampurawala, B. R. Carlton, S. Ravikumar, C. Nieva, S. Kim, H.-J. Lee, A. Sammak, G. Scappucci, M. Veldhorst, L. M. K. Vandersypen, E. Charbon, S. Pellerano, M. Babaie, and F. Sebastiano, "A scalable cryo-CMOS controller for the wideband frequency-multiplexed control of spin qubits and transmons," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 11, pp. 2930–2946, Nov. 2020. [Online]. Available: https://doi.org/10.1109/jssc.2020.3024678

[18] D. J. Egger and F. K. Wilhelm, "Adaptive hybrid optimal quantum control for imprecisely characterized systems," *Phys. Rev. Lett.*, vol. 112, p. 240503, Jun 2014. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.112.240503

[19] F. Fang, T. Chen, and R. A. Rutenbar, "Lightweight floating-point arithmetic: Case study of inverse discrete cosine transform," *EURASIP Journal on Advances in Signal Processing*, vol. 2002, no. 9, pp. 1–14, 2002.

[20] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Phys. Rev. A*, vol. 86, p. 032324, Sep 2012. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.86.032324

[21] D. J. Frank, S. Chakraborty, K. Tien, P. Rosno, T. Fox, M. Yeck, J. A. Glick, R. Robertazzi, R. Richetta, J. F. Bulzacchelli, D. Ramirez, D. Yilma, A. Davies, R. V. Joshi, S. D. Chambers, S. Lekuch, K. Inoue, D. Underwood, D. Wisnieff, C. Baks, D. Bethune, J. Timmerwilke, B. R. Johnson, B. P. Gaucher, and D. J. Friedman, "A cryo-CMOS low-power semi-autonomous qubit state controller in 14nm FinFET technology," in *2022 IEEE International Solid- State Circuits Conference (ISSCC)*. IEEE, Feb. 2022. [Online]. Available: https://doi.org/10.1109/isscc42614.2022.9731538

[22] X. Fu, L. Riesebos, L. Lao, C. G. Almudever, F. Sebastiano, R. Versluis, E. Charbon, and K. Bertels, "A heterogeneous quantum computer architecture," in *Proceedings of the ACM International Conference on Computing Frontiers*. Como Italy: ACM, May 2016, pp. 323–330. [Online]. Available: https://dl.acm.org/doi/10.1145/2903150.2906827

[23] X. Fu, L. Riesebos, M. A. Rol, J. van Straten, J. van Someren, N. Khammassi, I. Ashraf, R. F. L. Vermeulen, V. Newsum, K. K. L. Loh, J. C. de Sterke, W. J. Vlothuizen, R. N. Schouten, C. G. Almudever, L. DiCarlo, and K. Bertels, "eqasm: An executable quantum instruction set architecture," in *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, 2019, pp. 224–237.

[24] X. Fu, M. A. Rol, C. C. Bultink, J. van Someren, N. Khammassi, I. Ashraf, R. F. L. Vermeulen, J. C. de Sterke, W. J. Vlothuizen, R. N. Schouten, C. G. Almudever, L. DiCarlo, and K. Bertels, "An experimental microarchitecture for a superconducting quantum processor," in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO-50 '17. Cambridge, Massachusetts: Association for Computing Machinery, Oct. 2017, pp. 813–825. [Online]. Available: https://doi.org/10.1145/3123939.3123952

[25] R. Gebauer, "A flexible fpga-based control platform for superconducting multi-qubit experiments."

[26] "Quantum Computer Datasheet," https://quantumai.google/hardware/datasheet/weber.pdf, [Online; accessed 06-15-2021].

[27] A. Hashim, R. K. Naik, A. Morvan, J.-L. Ville, B. Mitchell, J. M. Kreikebaum, M. Davis, E. Smith, C. Iancu, K. P. O'Brien, I. Hincks, J. J. Wallman, J. Emerson, and I. Siddiqi, "Randomized compiling for scalable quantum computing on a noisy superconducting quantum processor," 2021.

[28] "IBM Quantum breaks the 100-qubit processor barrier," https://research.ibm.com/blog/127-qubit-quantum-processor-eagle.

[29] "IBM Quantum Services," https://quantum-computing.ibm.com/services?services=systems.

[30] M. R. Jokar, R. Rines, G. Pasandi, H. Cong, A. Holmes, Y. Shi, M. Pedram, and F. T. Chong, "Digiq: A scalable digital controller for quantum computers using sfq logic," 2022. [Online]. Available: https://arxiv.org/abs/2202.01407

[31] K. Kang, B. Kim, G. Choi, S.-K. Lee, J. Choi, J. Lee, S. Kang, M. Lee, H.-J. Song, Y. Chong, and J.-Y. Sim, "A 5.5mw/channel 2-to-7 ghz frequency synthesizable qubit-controlling cryogenic pulse modulator for scalable quantum computers," in *2021 Symposium on VLSI Circuits*, 2021, pp. 1–2.

[32] K. Kang, D. Minn, S. Bae, J. Lee, S. Bae, G. Jung, S. Kang, M. Lee, H.-J. Song, and J.-Y. Sim, "A cryo-CMOS controller IC with fully integrated frequency generators for superconducting qubits," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*. IEEE, Feb. 2022. [Online]. Available: https://doi.org/10.1109/isscc42614.2022.9731574

[33] G. Kiene, A. Catania, R. Overwater, P. Bruschi, E. Charbon, M. Babaie, and F. Sebastiano, "13.4 a 1gs/s 6-to-8b 0.5mw/qubit cryo-CMOS SAR ADC for quantum computing in 40nm CMOS," in *2021 IEEE International Solid- State Circuits Conference (ISSCC)*. IEEE, Feb. 2021. [Online]. Available: https://doi.org/10.1109/isscc42613.2021.9365927

[34] Y. Kim, A. Morvan, L. B. Nguyen, R. K. Naik, C. Jünger, L. Chen, J. M. Kreikebaum, D. I. Santiago, and I. Siddiqi, "High-fidelity three-qubit itoffoli gate for fixed-frequency superconducting qubits," *Nature Physics*, pp. 1–6, 2022.

[35] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, "A quantum engineer's guide to superconducting qubits," *Applied Physics Reviews*, vol. 6, no. 2, p. 021318, Jun. 2019. [Online]. Available: https://doi.org/10.1063/1.5089550

[36] S. Krinner, N. Lacroix, A. Remm, A. Di Paolo, E. Genois, C. Leroux, C. Hellings, S. Lazar, F. Swiadek, J. Herrmann *et al.*, "Realizing repeated quantum error correction in a distance-three surface code," *arXiv preprint arXiv:2112.03708*, 2021.

[37] L. Lao, P. Murali, M. Martonosi, and D. Browne, "Designing calibration and expressivity-efficient instruction sets for quantum computing," in *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, 2021, pp. 846–859.

[38] E. Leonard, M. A. Beck, J. Nelson, B. Christensen, T. Thorbeck, C. Howington, A. Opremcak, I. Pechenezhskiy, K. Dodge, N. Dupuis, M. Hutchings, J. Ku, F. Schlenker, J. Suttle, C. Wilen, S. Zhu, M. Vavilov, B. Plourde, and R. McDermott, "Digital coherent control of a superconducting qubit," *Phys. Rev. Applied*, vol. 11, p. 014009, Jan 2019. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevApplied.11.014009

[39] A. Li, S. Stein, S. Krishnamoorthy, and J. Ang, "Qasmbench: A low-level qasm benchmark suite for nisq evaluation and simulation," 2021.

[40] K. Li, R. McDermott, and M. G. Vavilov, "Hardware-efficient qubit control with single-flux-quantum pulse sequences," *Physical Review Applied*, vol. 12, no. 1, Jul. 2019. [Online]. Available: https://doi.org/10.1103/physrevapplied.12.014044

[41] P. J. Liebermann and F. K. Wilhelm, "Optimal qubit control using single-flux quantum pulses," *Phys. Rev. Applied*, vol. 6, p. 024022, Aug 2016. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevApplied.6.024022

[42] C. Loeffler, A. Ligtenberg, and G. S. Moschytz, "Practical fast 1-d dct algorithms with 11 multiplications," in *International Conference on Acoustics, Speech, and Signal Processing,*. IEEE, 1989, pp. 988–991.

[43] T. Lubinski, S. Johri, P. Varosy, J. Coleman, L. Zhao, J. Necaise, C. H. Baldwin, K. Mayer, and T. Proctor, "Application-oriented performance benchmarks for quantum computing," *arXiv preprint arXiv:2110.03137*, 2021.

[44] E. Magesan, J. M. Gambetta, and J. Emerson, "Scalable and robust randomized benchmarking of quantum processes," *Physical Review Letters*, vol. 106, no. 18, May 2011. [Online]. Available: http://dx.doi.org/10.1103/PhysRevLett.106.180504

[45] J. Marques, B. Varbanov, M. Moreira, H. Ali, N. Muthusubramanian, C. Zachariadis, F. Battistel, M. Beekman, N. Haider, W. Vlothuizen *et al.*, "Logical-qubit operations in an error-detecting surface code," *Nature Physics*, vol. 18, no. 1, pp. 80–86, 2022.

[46] R. McDermott and M. G. Vavilov, "Accurate qubit control with single flux quantum pulses," *Phys. Rev. Applied*, vol. 2, p. 014007, Jul 2014. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevApplied.2.014007

[47] R. McDermott, M. G. Vavilov, B. L. T. Plourde, F. K. Wilhelm, P. J. Liebermann, O. A. Mukhanov, and T. A. Ohki, "Quantum–classical interface based on single flux quantum digital logic," *Quantum Science and Technology*, vol. 3, no. 2, p. 024004, Jan. 2018. [Online]. Available: https://doi.org/10.1088/2058-9565/aaa3a0

[48] M. McEwen, D. Kafri, Z. Chen, J. Atalaya, K. J. Satzinger, C. Quintana, P. V. Klimov, D. Sank, C. Gidney, A. G. Fowler, F. Arute, K. Arya, B. Buckley, B. Burkett, N. Bushnell, B. Chiaro, R. Collins, S. Demura, A. Dunsworth, C. Erickson, B. Foxen, M. Giustina, T. Huang, S. Hong, E. Jeffrey, S. Kim, K. Kechedzhi, F. Kostritsa, P. Laptev, A. Megrant, X. Mi, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Niu, A. Paler, N. Redd, P. Roushan, T. C. White, J. Yao, P. Yeh, A. Zalcman, Y. Chen, V. N. Smelyanskiy, J. M. Martinis, H. Neven, J. Kelly, A. N. Korotkov, A. G. Petukhov, and R. Barends, "Removing leakage-induced correlated errors in superconducting quantum error correction," *Nature Communications*, vol. 12, no. 1, Mar. 2021. [Online]. Available: https://doi.org/10.1038/s41467-021-21982-y

[49] D. C. McKay, C. J. Wood, S. Sheldon, J. M. Chow, and J. M. Gambetta, "Efficient Z-Gates for Quantum Computing," *Physical Review A*, vol. 96, no. 2, p. 022330, Aug. 2017, arXiv: 1612.00858 version: 2. [Online]. Available: http://arxiv.org/abs/1612.00858

[50] R. V. Meter and C. Horsman, "A blueprint for building a quantum computer," *Communications of the ACM*, vol. 56, no. 10, pp. 84–93, Oct. 2013. [Online]. Available: https://doi.org/10.1145/2494568

[51] R. V. Meter and M. Oskin, "Architectural implications of quantum computing technologies," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 2, no. 1, pp. 31–63, Jan. 2006. [Online]. Available: https://doi.org/10.1145/1126257.1126259

[52] T. S. Metodi, A. I. Faruque, and F. T. Chong, "Quantum computing for computer architects, second edition," *Synthesis Lectures on Computer Architecture*, vol. 6, no. 1, pp. 1–203, Mar. 2011. [Online]. Available: https://doi.org/10.2200/s00331ed1v01y201101cac013

[53] M. A. Nielsen and I. L. Chuang, *Cambridge series on information and the natural sciences: Quantum computation and quantum information*. Cambridge, England: Cambridge University Press, Oct. 2000.

[54] J.-S. Park, S. Subramanian, L. Lampert, T. Mladenov, I. Klotchkov, D. J. Kurian, E. Juarez-Hernandez, B. Perez-Esparza, S. R. Kale, K. T. Asma Beevi, S. Premaratne, T. Watson, S. Suzuki, M. Rahman, J. B. Timbadiya, S. Soni, and S. Pellerano, "13.1 a fully integrated cryo-cmos soc for qubit control in quantum computers capable of state manipulation, readout and high-speed gate pulsing of spin qubits in intel 22nm ffl finfet technology," in *2021 IEEE International Solid- State Circuits Conference (ISSCC)*, vol. 64, 2021, pp. 208–210.

[55] K. H. Park, Y. S. Yap, Y. P. Tan, C. Hufnagel, L. H. Nguyen, K. H. Lau, S. Efthymiou, S. Carrazza, R. P. Budoyo, and R. Dumke, "Icarus-q: A scalable rfsoc-based control system for superconducting quantum computers," 2021. [Online]. Available: https://arxiv.org/abs/2112.02933

[56] S. J. Pauka, K. Das, R. Kalra, A. Moini, Y. Yang, M. Trainer, A. Bousquet, C. Cantaloube, N. Dick, G. C. Gardner, M. J. Manfra, and D. J. Reilly, "A cryogenic CMOS chip for generating control signals for multiple qubits," *Nature Electronics*, vol. 4, no. 1, pp. 64–70, Jan. 2021. [Online]. Available: http://www.nature.com/articles/s41928-020-00528-y

[57] G. Pekhimenko, V. Seshadri, O. Mutlu, M. A. Kozuch, P. B. Gibbons, and T. C. Mowry, "Base-delta-immediate compression: Practical data compression for on-chip caches," in *2012 21st international conference on parallel architectures and compilation techniques (PACT)*. IEEE, 2012, pp. 377–388.

[58] M. Poremba, S. Mittal, D. Li, J. S. Vetter, and Y. Xie, "Destiny: A tool for modeling emerging 3d nvm and edram caches," in *2015 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2015, pp. 1543–1546.

[59] T. Propson, B. E. Jackson, J. Koch, Z. Manchester, and D. I. Schuster, "Robust quantum optimal control with trajectory optimization," *Physical Review Applied*, vol. 17, no. 1, p. 014036, 2022.

[60] T. Recommendation, "81: Digital compression and coding of continuous-tone still images—requirements and guidelines," Technical report, CCITT, Tech. Rep., 1992.

[61] C. Rigetti and M. Devoret, "Fully microwave-tunable universal gates in superconducting qubits with linear couplings and fixed transition frequencies," *Physical Review B*, vol. 81, no. 13, Apr. 2010. [Online]. Available: https://doi.org/10.1103/physrevb.81.134507

[62] "IBM Quantum Roadmap," https://www.ibm.com/blogs/research/2020/09/ibm-quantum-roadmap/.

[63] K. Rojan, D. M. Reich, I. Dotsenko, J.-M. Raimond, C. P. Koch, and G. Morigi, "Arbitrary-quantum-state preparation of a harmonic oscillator via optimal control," *Phys. Rev. A*, vol. 90, p. 023824, Aug 2014. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.90.023824

[64] M. Rol, F. Battistel, F. Malinowski, C. Bultink, B. Tarasinski, R. Vollmer, N. Haider, N. Muthusubramanian, A. Bruno, B. Terhal *et al.*, "Fast, high-fidelity conditional-phase gate exploiting leakage interference in weakly anharmonic super-conducting qubits," *Physical review letters*, vol. 123, no. 12, p. 120502, 2019.

[65] "Scipy fftpack," https://docs.scipy.org/doc/scipy/reference/generated/scipy.fftpack.dct.html.

[66] S. Shen, W. Shen, Y. Fan, and X. Zeng, "A unified 4/8/16/32-point integer idct architecture for multiple video coding standards," in *2012 IEEE International Conference on Multimedia and Expo*. IEEE, 2012, pp. 788–793.

[67] Y. Shi, N. Leung, P. Gokhale, Z. Rossi, D. I. Schuster, H. Hoffmann, and F. T. Chong, "Optimized compilation of aggregated instructions for realistic quantum computers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM, Apr. 2019. [Online]. Available: https://doi.org/10.1145/3297858.3304018

[68] A. Singhadia, M. Mamillapalli, and I. Chakrabarti, "Hardware-efficient 2d-dct/idct architecture for portable hevc-compliant devices," *IEEE Transactions on Consumer Electronics*, vol. 66, no. 3, pp. 203–212, 2020.

[69] A. Spörl, T. Schulte-Herbrüggen, S. J. Glaser, V. Bergholm, M. J. Storcz, J. Ferber, and F. K. Wilhelm, "Optimal control of coupled josephson qubits," *Phys. Rev. A*, vol. 75, p. 012302, Jan 2007. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.75.012302

[70] L. Stefanazzi, K. Treptow, N. Wilcer, C. Stoughton, S. Montella, C. Bradford, G. Cancelo, S. Saxena, H. Arnaldi, S. Sussman, A. Houck, A. Agrawal, H. Zhang, C. Ding, and D. I. Schuster, "The qick (quantum instrumentation control kit): Readout and control for qubits and detectors," 2021. [Online]. Available: https://arxiv.org/abs/2110.00557

[71] V. M. Stojanović, A. Fedorov, A. Wallraff, and C. Bruder, "Quantum-control approach to realizing a toffoli gate in circuit QED," *Physical Review B*, vol. 85, no. 5, Feb. 2012. [Online]. Available: https://doi.org/10.1103/physrevb.85.054504

[72] V. Sze, M. Budagavi, and G. J. Sullivan, "High efficiency video coding (hevc)," in *Integrated circuit and systems, algorithms and architectures*. Springer, 2014, vol. 39, p. 40.

[73] S. S. Tannu, D. M. Carmean, and M. K. Qureshi, "Cryogenic-dram based memory system for scalable quantum computers: A feasibility study," in *Proceedings of the International Symposium on Memory Systems*, 2017, pp. 189–195.

[74] S. S. Tannu, Z. A. Myers, P. J. Nair, D. M. Carmean, and M. K. Qureshi, "Taming the instruction bandwidth of quantum computers via hardware-managed error correction," in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture*. ACM, Oct. 2017. [Online]. Available: https://doi.org/10.1145/3123939.3123940

[75] Y. Tomita and K. M. Svore, "Low-distance surface codes under realistic quantum noise," *Phys. Rev. A*, vol. 90, p. 062320, Dec 2014. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.90.062320

[76] T. D. Tran, "Fast multiplierless approximation of the dct," *Synthesis*, vol. 10, p. 1, 1999.

[77] J. P. G. van Dijk, B. Patra, S. Pellerano, E. Charbon, F. Sebastiano, and M. Babaie, "Designing a DDS-based SoC for high-fidelity multi-qubit control," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 12, pp. 5380–5393, Dec. 2020. [Online]. Available: https://doi.org/10.1109/tcsi.2020.3019413

[78] L. Xie, J. Zhai, Z. Zhang, J. Allcock, S. Zhang, and Y.-C. Zheng, "Suppressing ZZ crosstalk of quantum computers through pulse and scheduling co-optimization," in *Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM, Feb. 2022. [Online]. Available: https://doi.org/10.1145/3503222.3507761

[79] "Xilinx Ultrascale," https://docs.xilinx.com/v/u/en-US/ds890-ultrascale-overview.

[80] X. Xue, B. Patra, J. P. G. van Dijk, N. Samkharadze, S. Subramanian, A. Corna, B. P. Wuetz, C. Jeon, F. Sheikh, E. Juarez-Hernandez, B. P. Esparza, H. Rampurawala, B. Carlton, S. Ravikumar, C. Nieva, S. Kim, H.-J. Lee, A. Sammak, G. Scappucci, M. Veldhorst, F. Sebastiano, M. Babaie, S. Pellerano, E. Charbon, and L. M. K. Vandersypen, "CMOS-based cryogenic control of silicon quantum circuits," *Nature*, vol. 593, no. 7858, pp. 205–210, May 2021. [Online]. Available: https://doi.org/10.1038/s41586-021-03469-4

[81] E. Zahedinejad, J. Ghosh, and B. C. Sanders, "Designing high-fidelity single-shot three-qubit gates: a machine-learning approach," *Physical Review Applied*, vol. 6, no. 5, p. 054005, 2016.

[82] Y. Zhao, Y. Ye, H.-L. Huang, Y. Zhang, D. Wu, H. Guan, Q. Zhu, Z. Wei, T. He, S. Cao *et al.*, "Realizing an error-correcting surface code with superconducting qubits," *arXiv preprint arXiv:2112.13505*, 2021.

[83] Z. Zhou and Z. Pan, "Effective hardware accelerator for 2d dct/idct using improved loeffler architecture," *IEEE Access*, vol. 10, pp. 11 011–11 020, 2022.