A scalable control system for a superconducting adiabatic quantum optimization processor

M W Johnson¹, P Bunyk¹, F Maibaum², E Tolkacheva¹, A J Berkley¹, E M Chapple¹, R Harris¹, J Johansson¹, T Lanting¹, I Perminov¹, E Ladizinsky¹, T Oh¹ and G Rose¹

¹ D-Wave Systems Incorporated, 100-4401 Still Creek Drive, Burnaby, BC, V5C 6G9, Canada
² Physikalisch Technische Bundesanstalt, Bundesallee 100, 38116 Braunschweig, Germany

E-mail: mwjohnson@dwavesys.com

Received 7 January 2010, in final form 15 March 2010
Published 21 April 2010
Online at stacks.iop.org/SUST/23/065004

Abstract
We have designed, fabricated and operated a scalable system for applying independently programmable time-independent, and limited time-dependent flux biases to control superconducting devices in an integrated circuit. Here we report on the operation of a system designed to supply 64 flux biases to devices in a circuit designed to be a unit cell for a superconducting adiabatic quantum optimization system. The system requires six digital address lines, two power lines, and a handful of global analog lines.

(Some figures in this article are in colour only in the electronic version)

1. Introduction

Several proposals for how one might implement a quantum computer now exist. One of these is based on enabling adiabatic quantum optimization algorithms in networks of superconducting flux qubits connected via tunable coupling devices [1]. Flux qubits can be manipulated by applying magnetic flux via currents along inductively coupled control lines. This can be accomplished with one analog control line per device driven by room temperature current sources and routed, through appropriate filtering, down to the target device on chip. Beyond the scale of a few dozens of such qubits the one-analog-line-per-device approach becomes impractical. Hundreds of qubits could require thousands of wires, each subject to filtering, cross-talk, and thermal requirements so as to minimize disturbance of the thermal and electromagnetic environment of the targeted qubits, which are operated at millikelvin temperatures. We require an approach that does not use so many wires.

One advantage of using superconductor based qubits is the existence of a compatible classical digital and mixed signal electronics technology based on the manipulation of single flux quanta (SFQ) [2, 3]. The ability to manufacture classical control circuitry [4–6] on the same chip, with the same fabrication technology as is used in construction of the qubits, addresses many of the thermal and electromagnetic compatibility requirements faced in integrating control circuitry with such a processor. The idea of using SFQ circuitry to control flux qubits is not new, and has investigated by a number of researchers [7–13].

We present here a description of a functioning system of on-chip programmable magnetic memory (PMM) designed to manipulate the parameters and state of superconducting flux qubits and tunable couplers, in such a way as to overcome the scalability limitations of the one-analog-line-per-device paradigm. This system comprises three key parts.

The first of these is a SFQ demultiplexer used as an addressing system. It is constructed as a binary tree of $2^N - 1$:2 SFQ demultiplexer gates as shown in figure 1. For the specific design discussed here, the number of address lines $N$ is 6. This demultiplexer allows many devices to be addressed using only a few address lines.

The second part is a set of digital-to-analog converters (DACs), located at the leaves of the address tree. These DACs comprise storage inductors that can hold an integer number of single magnetic flux quanta ($\phi_0 = \hbar/2e$). Their digital input are single flux quanta, and their analog output are the stored flux, which can be coupled into a target device. The magnitude of this output flux is proportional to the number of stored flux quanta. Each DAC has two such storage inductors,
Thus, this circuit required $8 \times 15$ as described in [15]. Though not discussed here, the state of equivalent to the tunable coupler described elsewhere [14]. An coupling the output of the DAC into a variable gain element, of a DAC to a time-dependent signal. This is achieved by device. In our architecture, the output of these DACs is static. a COARSE stage, and a FINE stage, named for the relative strength with which their output flux is coupled to the target device. In our architecture, the output of these DACs is static. The third part is a method for converting the static output of a DAC to a time-dependent signal. This is achieved by coupling the output of the DAC into a variable gain element, equivalent to the tunable coupler described elsewhere [14]. An analog line carrying a time-dependent current is coupled to the target device via the variable gain element. This approach is useful in the types of circuits of interest here because single analog lines can be shared among large numbers of devices that need the same functional dependence on time, but may require individual tunability of the gain and offset.

We designed, fabricated, and operated integrated circuits comprising this type of control system architecture. One such circuit includes eight superconducting rf-SQUID flux qubits as described in [15]. Though not discussed here, the state of each qubit was read out via a quantum flux parametron (QFP) latch or buffer which was in turn read by an x-y addressable dc-SQUID array as discussed in detail in [16]. Each rf-SQUID flux qubit has inductive ports coupled to five different DACs, and 24 compound Josephson junction (CJJ) rf-SQUID couplers [14], each of which is coupled to a single DAC. Thus, this circuit required $8 \times 5 + 24 = 64$ DACs. The particular control circuit described here comprises two 1:32 demultiplexers with six shared address lines and two separate power lines. This circuit included 1538 junctions ranging in size from a minimum of 0.6 $\mu$m diameter (32 of them) to a maximum of 4 $\mu$m in diameter.

The paper is organized as follows: requirements on control circuitry derived from the devices, architecture and operating procedures in superconducting adiabatic quantum optimization systems are discussed in section 2. The specific control circuitry architecture is discussed in section 3. Data demonstrating the performance relative to requirements is presented in section 4. Conclusions are presented in section 5.

The measurements reported in section 4 were performed on chips fabricated in a four Nb layer superconducting process employing a standard Nb/AlO$_x$/Nb trilayer, a TiPt resistor layer, and planarized PECVD SiO$_2$ dielectric layers. Design rules included 0.25 $\mu$m lines and spaces for wiring layers and a minimum junction diameter of 0.6 $\mu$m. A sample process cross-section is shown in figure 2.

2. Magnetic memory requirements

Our intent is to embody a specific quantum algorithm in hardware. This algorithm, known as adiabatic quantum optimization (AQO), is a novel approach for solving combinatorial optimization problems [17, 18]. Unlike the incumbent techniques for such problems, such as simulated annealing or genetic algorithms, AQO algorithms include procedures that are explicitly quantum mechanical. The requirement to provide the quantum mechanical resources necessary for running this algorithm places unusual constraints on processor systems and components.

Algorithm 1: An adiabatic quantum optimization algorithm.

**Input:** A run-time $t_f$; a repeat count $R$; an allowed edge set $E$; an $N$ dimensional vector $\hat{h}$ and an upper diagonal $N \times N$ matrix $\hat{K}$ with $h_j, K_{ij} \in \mathbb{R}$ and $K_{ij} \in E$.

**Output:** An $R$ element array Outcell, for which each element is a set $\{s^*_j\}$, $s_j \in \{-1, +1\}$, which represents a potential minimizer of $\mathcal{E}(s_1, \ldots, s_N) = \sum_{j=1}^{N} h_j s_j + \sum_{i,j \in E} K_{ij} s_i s_j$.

```
set $t_f = 100 \, \mu s$, $R = 128$; 
load $\hat{h}$ and $\hat{K}$ values into hardware; 
wait 1 ms for hardware to cool down; 
for $j = 1$ to $R$ do
    | Run annealing algorithm; 
    | Read out qubits to generate trial solution $\{s^*_j\}$; 
    | Set Outcell(j)={$s^*_j$}; 
end
```

Quantum computation intimately ties the physics of the underlying hardware to its intended algorithmic use. Both...
The necessity for explicitly defining the set \( E \) arises because ultimately we will connect this term in the objective function to physical couplings between pairs of qubits, and for a variety of reasons the number of elements in \( E \) will generally be much less than the total number of possible pairs \( N(N - 1)/2 \). A design constraint on processor architecture is that qubits must be connected in such a way so that finding the minimum of equation (1) is NP-hard. Even with this constraint, it is straightforward to find realizable sets \( E \) for which this holds, and we will focus exclusively on these cases.

An AQQO algorithm exists for solving this problem. The approach is outlined in algorithms 1 and 2. The control system reported on here enters into these in the load step of algorithm 1.

### 2.1. Processor interconnect architecture

There are many possibilities for how one might try to build a hardware system capable of running algorithms 1 and 2. Here we focus on a unit cell consisting of eight qubits and 24 couplers. See figure 3 for a schematic and photograph showing the interconnect pattern. Copies of this unit cell can be connected together, as indicated in the top of figure 3, and this is how we design larger systems. This choice of unit cell fixes the allowed edge set \( E \), and satisfies the constraint that minimizing equation (1) be NP-hard.

### 2.2. Number of DACs

The total number of DACs required for circuits of increasing complexity is shown in table 1. Here we provide a brief overview of how these numbers arise, and refer the reader to [14, 15] for further details.

| Unit cells | Qubits | Couplers | DACS | JJs |
|------------|--------|----------|------|-----|
| 1          | 8      | 16       | 56   | 1500|
| 4          | 32     | 72       | 232  | 6000|
| 16         | 128    | 328      | 968  | 24000|
| 64         | 512    | 1416     | 3976 | 96000|
| 256        | 2048   | 5896     | 16136| 384000|

Consider the following discrete optimization problem. Given a vector \( \hat{h} \) and upper diagonal matrix \( \hat{K} \), where the elements of both are real numbers, find the set \( \{ s_i^1 \} \) that minimizes the objective function

\[
\mathcal{E}(s_1, \ldots, s_N) = \sum_{j=1}^{N} h_j s_j + \sum_{i,j \in E} K_{ij} s_is_j \tag{1}
\]

where \( s_j = \{-1, +1\} \), and \( E \) is an set of \((i, j)\) pairs where \( K_{ij} \) is allowed to be non-zero. We call \( E \) the allowed edge set. The necessity for explicitly defining the set \( E \) arises because ultimately we will connect this term in the objective function to physical couplings between pairs of qubits, and for a variety of reasons the number of elements in \( E \) will generally be much less than the total number of possible pairs \( N(N - 1)/2 \). A design constraint on processor architecture is that qubits must be connected in such a way so that finding the minimum of equation (1) is NP-hard. Even with this constraint, it is straightforward to find realizable sets \( E \) for which this holds, and we will focus exclusively on these cases.

An AQQO algorithm exists for solving this problem. The approach is outlined in algorithms 1 and 2. The control system reported on here enters into these in the load step of algorithm 1.
required. For this design, the flux bias is provided by the DAC shown in red in figure 4.

2.2.2. Five DACs per qubit. The potential energy of an ideal compound Josephson junction rf-SQUID qubit is [19]

\[
U = \frac{\Phi_0 I_c}{2\pi} \cos \left( \frac{2\pi \Phi_q}{\Phi_0} \right) \cos \left( \frac{\pi \Phi_{CJJ}}{\Phi_0} \right) + \frac{(\Phi_q - \Phi_q^*)^2}{2L_q} + \frac{(\Phi_{CJJ} - \Phi_{CJJ}^*)^2}{2L_{CJJ}} \tag{2}
\]

where \( I_c \) is the sum of Josephson critical currents in the compound-junction, \( L_q \) and \( L_{CJJ} \) are the inductance in the qubit and compound-junction loop, respectively. Likewise \( \Phi_q \), \( \Phi_q^* \), and \( \Phi_{CJJ} \), \( \Phi_{CJJ}^* \) are the internal and applied flux for the qubit and CJJ loop respectively.

Equation (2) is only applicable when the two junctions making up the compound-junction are identical. Junction critical current \( I_{S} \) of identically drawn Josephson junctions in superconductor fabrication processes are reported to have a normal distribution with a standard deviation of anywhere from 1% to 5% [20, 21]. Thus, we expect real compound-junctions to be naturally imbalanced. This causes difficulties in running the annealing algorithm, algorithm 2 [15]. To overcome the junction imbalance problem we use a more complex structure which we call a compound–compound-Josephson junction (CCJJ) which is described in detail in [15]. This provides two additional degrees of control freedom per qubit, which can be used to correct for reasonable junction imbalance (~5% \( I_c \) difference). We access these structures via the blue CCJJ minor DACs in figure 4.

As inter-qubit coupling strength is adjusted, the susceptibility of the coupler, and the extent to which it inductively loads the qubit, will change [14]. This causes the qubit inductance \( L_q \) in equation (2) to be dependent upon the choice of \( \{K_{ij}\} \). To overcome the resulting problem-dependent inter-qubit imbalance, we add an additional compound-junction, comprising much larger junctions, in series with the qubit inductance. We call this structure an \( L \)-tuner [15]. The Josephson inductance of this compound-junction is modified with application of a flux bias applied through an on-chip flux DAC, shown in green in figure 4.

As discussed in [22], care must be taken during annealing to ensure that the final Hamiltonian \( H_F \), the one encoding the problem we wish to solve, is that which was intended. Using a compound-junction to modify the relative weights of \( H_J \) and \( H_F \) causes \( h \) and \( K \) to change during annealing, both in an absolute sense, and relative to each other. This arises because although energy scales \( h \) and \( K \) are both functions of the persistent currents in the qubits \( (I_0) \), they have different functional dependencies. Qubit \( I_p \) changes during annealing, distorting \( H_F \).

To keep the relative scale constant, the value of the applied flux used to implement \( h \) must change during annealing, but \( \Phi_q(t) \) will be different for each qubit, depending on the intended value of \( h \) for that problem. This is accomplished by giving each qubit another tunable coupler, coupled to both the qubit and a shared external analog flux bias line. We call this an \( I_p \)-compensator. Each such coupler is used as a variable gain element, programmed with its own DAC (the pink DAC in figure 4), and used to scale a global controlled signal to the locally required \( h \).

Finally, each qubit has a DAC that can apply a small dc flux bias to its main loop (the black DAC in figure 4).

2.3. Precision and range requirements

Requirements on precision and range of flux from the DACs ultimately depend on the precision to which the elements of \( h \) and \( K \) are to be specified. The system described here was designed to be able to attain four effective bits of precision on parameters \( h_j \) and \( K_{ij} \); in other words, the elements \( h_j \) and \( K_{ij} \) can be specified with a relative precision of about 5%. This does not mean that the DACs need only four bits of precision. The DAC requirements are derived from those on Hamiltonian parameters \( h \) and \( K \), based on which aspect of a qubit or coupler is being controlled. In our case, requiring four bits of precision in \( h \) and \( K \) typically translates into a requirement of about eight bits of precision in each of the DACs.

The primary design parameters for each DAC is its dynamic range: how much flux is it necessary for the DAC to provide, and how FINE a control of that flux is needed. All of the DACs were designed to cover their respective ranges in subdivisions of either 300 or 400 steps. However, they differ in the total amount of flux coupled at maximum range from around 25 m\( \Phi_0 \) for the qubit flux bias DAC to as much as 0.9 \( \Phi_0 \) from the coupler DAC. A summary of desired flux ranges and minimal flux steps is shown in table 2.

2.4. Programming constraints

In the design discussed here, there are five DACs per qubit and one per coupler that need to be programmed to implement a specific problem instance. While the DACs are being programmed, power is applied to the SFQ circuits in the
address tree, and the chip will heat. The amount of time we must wait for it to cool afterward (step 3 in algorithm 1) depends on the peak temperatures reached by the various portions of the circuitry, and the relaxation mechanisms enabling their return to equilibrium [23]. Minimizing overall programming time, including that required to cool, is an important design constraint, and must be considered when comparing control circuitry architectures.

Given the block architecture described above, the number of DACs we must program increases with processor size as shown in Table 1. While not every DAC will need to be programmed for each unique configuration of $h_j$ and $K_{ij}$ in practice, in what follows the assumption will be that all are. In our multiple flux quantum based encoding scheme, the programming time will depend on the value programmed. To estimate the basic scaling with number of devices, it is probably reasonable to assume that each time the processor is programmed for a new problem, each DAC stage must receive on average half of its designed capacity in the processor is programmed for a new problem, each DAC

| DAC type     | Span | $\Delta \Phi$ | Min $\Phi_0$ | Max $\Phi_0$ | COARSE/FINE ratio |
|--------------|------|---------------|--------------|--------------|------------------|
| Qubit flux   | 25.5 m$\Phi_0$ | 0.1 m$\Phi_0$ | 17           | 17           | 14.1             |
| CJ balance   | 66.1 m$\Phi_0$ | 0.4 m$\Phi_0$ | 17           | 17           | 14.1             |
| L-tuner      | 0.465 $\Phi_0$ | 1.1 m$\Phi_0$ | 40           | 10           | 10.7             |
| Coupler      | 0.968 $\Phi_0$ | 2.2 m$\Phi_0$ | 40           | 10           | 10.6             |

**Table 2.** Designed flux ranges and minimal flux steps by DAC type.

Programming speed can then be bought at the expense of additional input lines and more parallelization—using more, shallower address trees each with its own separate input. However, with or without parallelization, one must be able to load all the pulses without errors. The more DACs being programmed, the more pulses there are that must be routed through the address tree with fidelity, and the smaller the acceptable error probability per pulse. For example, per Table 1, with 2048 qubits, we require 16 136 DACs. An average problem would require loading $\sim 3.2 \times 10^3$ flux quanta onto the chip. If we want 95% confidence that we can program problems correctly 99 times out of 100, the probability that any flux quanta makes an error should not be greater than $10^{-9}$. With 128 qubits, $10^{-8}$ is sufficient.

Bit error probability in SFQ circuits has been extensively studied [26–28]. Satisfying these and even more demanding error rate requirements at sub-kelvin temperatures is straightforward, but needs to be confirmed for any particular implementation.

### 3. Magnetic memory architecture

#### 3.1. Two-stage multiple flux quantum DAC with reset

The DACs each have two stages. Each stage comprises a large storage inductor in series with a two-junction reset SQUID, and an input junction, as shown in Figure 5(a). Each stage was designed with $\beta = 2\pi L I_s / \Phi_0$ in the range 75–300, depending on their function, and thus able to hold in the range of 10–40 flux quanta of either polarity. Here $L$ is the stage inductance, and $I_s$ is the effective critical current of the two-junction reset SQUID.

The two-junction reset SQUID is used to empty the DAC stage of stored flux. This is accomplished by applying $\Phi_0/2$ flux to the reset loop, so that its effective critical current $2L_s \cos (\pi \Phi_0 / \Phi_0)$, and thus DAC stage $\beta$, is diminished to below the level required to store flux. For this reset function to be effective, it must be possible to suppress the effective critical current of the reset SQUID to less than that required to store one $\Phi_0$ in the storage inductor. This requirement places an upper bound on the DAC stage $\beta$. It also places a requirement on how closely matched the $I_s$s of the two junctions in the reset SQUID must be to each other. This is because the minimum effective critical current will not be less than the difference in the $L_s$s of the two reset junctions. Thus given a particular fabrication process, with its feature size, penetration depths, and characteristic junction $I_s$ spread, there will be some maximum number of $\Phi_0$ that can be stored in a DAC that can be reliably reset to an empty state. This limits the dynamic range of an individual DAC stage.

We can achieve a dynamic range greater than that of an individual DAC stage, and shorten programming times, by connecting two or more stages together, as indicated in Figure 5(a). The intervening transformer couples the different stages into the target circuit with different weights. The flux transformer can be thought of as playing the role of an R–2R ladder, such as is frequently used in construction of semiconductor DACs. One important difference is that successive stages of this DAC differ from each other not by factors of two, but more typically by a factor of ten, depending on the requirements set by the target device. The other difference is that we are transforming and dividing flux rather than voltage.
shown in figure 5(b) was extracted from the layout using FastHenry 3.0wr [24]. The three-port inductance matrix for the flux transformer [24].

To see this more clearly, we consider a simple lumped equivalent circuit of figure 6(a). The resulting $Y_{EQ}$ is plotted versus frequency in figure 6(b). The magnitude of $R_{EQ}$ will be frequency dependent, and can be determined as $R_{EQ}(f) = (\pi Y_Q(f))^{-1}$ where $Y_Q$ is the admittance of the circuit across the terminals shown in figure 6(a). The resulting $R_{EQ}$ is plotted versus frequency in figure 6(c) for circuit parameters described in the caption. At low frequency $R_{EQ}$ is around 10 MΩ and grows at frequencies above a few gigahertz.

Even at low frequency, most current fluctuations from $R_{SH}$ are shunted by the DAC input junction, whose small-signal inductance is between 100 and 150 times smaller than the DAC storage inductor $L_{DAC}$, depending on the state of the DAC. Noise current that does flow into $L_{DAC}$ can in turn couple into the qubit, though it is further reduced by a factor of $M/(L_Q + L_{DJ})$, or around 1/40 for the circuits discussed here. At higher frequencies fluctuations are further shunted by the junction capacitances.

Table 3. The three-port inductance matrix for the flux transformer shown in figure 5(b) was calculated by modeling the layout using FastHenry 3.0wr [24].

|        | Fine   | Coarse | Target |
|--------|--------|--------|--------|
| Fine   | 3.50 nH| 9.9 pH | 0.7 pH |
| Coarse | 3.90 nH| 10.2 pH|        |
| Target | 7.8 pH |        |        |

The flux transformer can be modeled with the equivalent circuit shown in figure 5(b), where we consider only two storage inductors $L_{COARSE}$, $L_{FINE}$, and the target inductor $L_{TARGET}$. The inductance matrix for the equivalent circuit shown in figure 5(b) was calculated by modeling the layout using FastHenry 3.0wr [24], and is shown in table 3.

The flux coupled out of this DAC can be summarized by the expression:

$$\Phi_{OUT} = k \left( N_{COARSE} \Phi_0 + \frac{1}{\gamma} N_{FINE} \Phi_0 \right)$$  (3)

where $N_{COARSE,FINE}$ represent the integer number of flux quanta that are stored within the respective DAC stages, $k$ the coupling constant describing the amount of flux from the COARSE stage into the output device, and $\gamma$, the division ratio between COARSE and FINE, which is typically 10 for the devices discussed here.

### 3.2. DAC noise

One concern with using SFQ circuitry in this way arises from the fact that its Josephson junctions are usually critically damped with external shunt resistors. These resistors are a source of fluctuations which may ultimately decrease the precision with which Hamiltonian parameters $h$ and $K$ can be specified. In the design presented in figure 5(a), a number of factors serve to isolate the resistors in the SFQ circuitry from the junctions in the qubit, so that their impact on the qubit can be quite small.

To see this more clearly, we consider a simple lumped element circuit model of the DAC and qubit. A description of our system using lumped elements is reasonable at low enough frequencies, but will eventually fail at higher frequencies, for example when the 1 mm long coils in the DAC spirals approach $\lambda$/4 at around 100 GHz. Fortunately, we are most concerned with fluctuations occurring at an energy scale comparable to, or less than, the tunnel splitting of our qubits during the annealing algorithm, typically a few gigahertz or less [15].

There are many shunted junctions in the SFQ circuits discussed here, but none couples more strongly into the qubit than the input junctions of the various COARSE DAC stages, labeled $J_2$ in figure 5(a). In what follows, the effect of this junction’s shunt resistor on a qubit is considered by analyzing an equivalent circuit shown in figure 6(a) for the case of the qubit flux bias DAC. Here the DAC input junction has been linearized, and is represented by its Josephson inductance $L_{DJ}$ and junction capacitance $C_{DJ}$. Two different operating points of the DAC input junction, corresponding to the DAC being empty or full of flux, are used to determine the small-signal inductance $L_{DJ}$. The qubit’s four CCJJ junctions are represented as a single linearized junction described by $L_{QJ}$ and $C_{QJ}$.

One way to characterize the extent to which fluctuations in $R_{SH}$ couple into qubit junctions is by comparing its effect to that of an equivalent shunt resistor $R_{SH}$ connected directly across the qubit junctions, as indicated in figure 6(b). The magnitude of $R_{EQ}$ will be frequency dependent, and can be determined as $R_{EQ}(f) = (\pi Y_Q(f))^{-1}$ where $Y_Q$ is the admittance of the circuit across the terminals shown in figure 6(a). The resulting $R_{EQ}$ is plotted versus frequency in figure 6(c) for circuit parameters described in the caption. At low frequency $R_{EQ}$ is around 10 MΩ and grows at frequencies above a few gigahertz.
The other DACs couple into the qubit in a less straightforward fashion, but we can still make a reasonable estimate of their impact. An example is the case of the L-tuner DAC. The L-tuner is a dc-SQUID connected in series with the qubit inductor \([15]\). It is flux biased in such a manner that it would not apply any flux into the qubit if its two 8 \(\mu\)A, 1.85 \(\mu\)m diameter junctions were identical, but does so when they differ. A 1% mismatch between junction critical currents is typical for junctions of this size in our process. A typical operating point for the L-tuner corresponds to a flux bias of \(\Phi_0/4\). For this case, following equation 4(c) from \([15]\), about 1% of the flux applied to the L-tuner will get applied into the qubit body. This is comparable but smaller than the corresponding factor of \(M/(L_Q + L_Q)\) of 1/40 discussed for the qubit flux bias DAC. For this reason, we expect the corresponding \(R_{EQ}\) for the case of the L-tuner DAC to be larger than that of the qubit flux bias DAC. It should then have a relatively smaller effect on the qubit. Similar arguments apply to the other DAC types.

An \(R_{EQ}\) of 10 M\(\Omega\) will contribute approximately \(L_Q\sqrt{4\Phi_0}/R\) \(\approx 50\) p\(\Phi_0\) Hz\(^{-1/2}\) flux noise into the qubit, considerably less than the 1.3 \(\mu\Phi_0\) Hz\(^{-1/2}\) at 1 Hz observed in 1/f noise in our qubits \([15, 25]\). Thus we do not expect the shunt resistors in our control circuitry will add a significant amount of flux noise to our qubits.

### 3.3. Programmable time-dependent signals

As discussed in section 2.2.2, we require the ability to supply time-dependent signals to each of the qubits to compensate for the fact that the qubit persistent current changes during the annealing process. These time-dependent signals need to have the same temporal shape but with different magnitudes. The DACs discussed above can hold static flux, and are not suited to provide real-time signals. This is because they do not include a sample-and-hold stage to protect the output from transients during programming. Moreover, real-time updating of the DACs would raise the temperature of the chip to an unacceptable level.

Rather, time-dependent signals can be customized using the tunable coupler discussed in \([14]\) as a variable or programmable gain element. A global analog bias line holding a master copy of the desired time-dependent signal is coupled to each qubit on the chip through its own programmable gain element, as indicated in figure 7. Each programmable gain element is controlled by its own DAC. In conjunction with an additional DAC (not shown) to provide a flux offset to each target device, the master copy of the signal can be uniquely transformed for each target in the following manner:

\[
\Phi_1(t) = a_i + g_i \Phi_{global}(t)
\]

where \(a_i\) and \(g_i\) are programmable on a per device basis. This is not as flexible as having independent arbitrary waveform generators for each device, but it is flexible enough to satisfy the requirements of the \(I_F\) compensator.

### 3.4. Demultiplexer tree

The DACs discussed above were loaded with SFQ pulses routed through a binary tree demultiplexer circuit shown in figure 1. Each address tree is fed SFQ pulses originating in an SFQ generator circuit, namely a flux biased dc-SQUID. Each node of the tree is made of a 1:2 SFQ demultiplexer circuit, as shown in figure 8. The 1:2 demultiplexer circuit is addressed with a magnetically coupled flux bias line which steers an incoming SFQ pulse to one of its outputs based on the sign and magnitude of current on that address line.

Reversing the polarity of the bias current allows flux quanta of opposite polarity to be routed, though the sign of the address current must also be reversed to get this negative flux to the same output port. This makes use of a symmetry not commonly exploited in RSFQ circuits. Here it allows the DAC...
those reported here is shown in figure 9. The image was of one of the DACs and demultiplexer cells equivalent to service the circuit block.

Address lines are shared for all demultiplexer nodes at a particular depth of the tree. The final address line in the tree allows DAC programming to be performed incrementally—starting from the previously programmed state without first resetting to the empty (no stored flux) state.

Address lines are shared externally accessible analog line. The qubit degeneracy point is easily measured [16]. If the DAC is then programmed with, for example, +5 $\Phi_0$ in its COARSE stage, one can determine the change in current on that analog qubit flux bias line required to compensate for the shift in degeneracy point. This allows us to find the ratio of mutual inductance between each DAC stage to store flux of either sign, and allows the state of each stage to be both incremented and decremented. This in turn allows DAC programming to be performed incrementally—starting from the previously programmed state without first resetting to the empty (no stored flux) state.

In the architecture described above, many of the DACs are embedded deeply within the circuit, with no convenient or direct method to determine how much flux they actually apply to the target per $\Phi_0$ in the FINE and COARSE stages. This inconvenience is addressed in two ways. Each variant of DAC is implemented in a separate standalone or break-out circuit in which it applies a flux bias directly to a two-junction dc-SQUID. The dc-SQUID $I_c$ versus $\Phi$ modulation curve is then measured versus DAC state, and a precise calibration of FINE and COARSE weights ($k$ and $\gamma$) can be extracted. Data from parameters extracted in this way is presented in table 4. Second, within the body of the circuit block, wherever a DAC is used to apply a flux bias, an analog line is also used to flux bias that target device in parallel with that DAC. This combination is indicated in the inset of figure 10. This single analog line is shared amongst all like control nodes for all qubits, so that only a handful of such lines are required to service the entire chip. Of course, the shared analog line cannot be used for independent control of all devices simultaneously, but it is nevertheless useful for testing individual devices. For example, each qubit is flux biased by its own DAC and a single shared externally accessible analog line. The qubit degeneracy point is easily measured [16]. If the DAC is then programmed with, for example, +5 $\Phi_0$ in its COARSE stage, one can determine the change in current on that analog qubit flux bias line required to compensate for the shift in degeneracy point. This allows us to find the ratio of mutual inductance between the analog line and the qubit to that between each DAC stage (COARSE and FINE) and the qubit. We can independently measure the mutual inductance of the analog line into the qubit by noting the $\Phi_0$ periodicity in its response. We can then

### Table 4. Designed versus achieved DAC parameters.

| DAC type   | COARSE step (m$\Phi_0$) Design | Achievement | FINE step (m$\Phi_0$) Design | Achievement |
|------------|-------------------------------|-------------|-------------------------------|-------------|
| Qubit flux | 3.0                           | 3.506(3)    | 0.21                          | 0.268(3)    |
| CCJJ       | 5.6                           | 3.899(2)    | 0.40                          | 0.296(3)    |
| L-tuner    | 11.3                          | 8.481(1)    | 1.1                           | 1.061(1)    |
| Coupler    | 23.6                          | 19.022(1)   | 2.2                           | 1.788(1)    |

| DAC type   | MAX COARSE $\Phi_0$ Design | Achievement | Max coupled flux ($\Phi_0$) Design | Achievement |
|------------|---------------------------|-------------|----------------------------------|-------------|
| Qubit flux | 17                        | 22          | 0.050                            | 0.077       |
| CCJJ       | 17                        | 22          | 0.093                            | 0.086       |
| L-tuner    | 41                        | 40          | 0.460                            | 0.339       |
| Coupler    | 41                        | 35          | 0.960                            | 0.665       |

4. Demonstration of control circuit functionality

For the circuit block discussed in this paper, the control circuitry in its entirety represents moderate complexity—certainly not the most complex or heterogeneous SFQ circuit demonstrated to date, nor the one with the most junctions. The eight-qubit circuit block reported here, including the analog line into the qubit to that between each DAC stage to store flux of either sign, and allows the state of each stage to be both incremented and decremented. This in turn allows DAC programming to be performed incrementally—starting from the previously programmed state without first resetting to the empty (no stored flux) state.

In the architecture described above, many of the DACs are embedded deeply within the circuit, with no convenient or direct method to determine how much flux they actually apply to the target per $\Phi_0$ in the FINE and COARSE stages. This inconvenience is addressed in two ways. Each variant of DAC is implemented in a separate standalone or break-out circuit in which it applies a flux bias directly to a two-junction dc-SQUID. The dc-SQUID $I_c$ versus $\Phi$ modulation curve is then measured versus DAC state, and a precise calibration of FINE and COARSE weights ($k$ and $\gamma$) can be extracted. Data from parameters extracted in this way is presented in table 4. Second, within the body of the circuit block, wherever a DAC is used to apply a flux bias, an analog line is also used to flux bias that target device in parallel with that DAC. This combination is indicated in the inset of figure 10. This single analog line is shared amongst all like control nodes for all qubits, so that only a handful of such lines are required to service the entire chip. Of course, the shared analog line cannot be used for independent control of all devices simultaneously, but it is nevertheless useful for testing individual devices. For example, each qubit is flux biased by its own DAC and a single shared externally accessible analog line. The qubit degeneracy point is easily measured [16]. If the DAC is then programmed with, for example, +5 $\Phi_0$ in its COARSE stage, one can determine the change in current on that analog qubit flux bias line required to compensate for the shift in degeneracy point. This allows us to find the ratio of mutual inductance between the analog line and the qubit to that between each DAC stage (COARSE and FINE) and the qubit. We can independently measure the mutual inductance of the analog line into the qubit by noting the $\Phi_0$ periodicity in its response. We can then

![Figure 9. SEM image of a portion of the DAC and demultiplexer circuitry after deposition and patterning of the resistor layer and the trilayer steps, but prior to applying the upper three dielectric and metal layers.](image-url)
determine \( k \) and \( \gamma \) for that DAC, which are the parameters we need to determine how much flux the DAC applies to its target.

This feedback measurement is applicable to determining \( k \) and \( \gamma \) for DACs used for various types of control, not just qubit flux bias. The only thing that differs is the nature of the measured quantity. For the qubit flux bias DAC, the qubit degeneracy point is used. For the L-tuner DAC, a measure of the qubit’s inductance, ultimately a measure of its circulating current, can be used. For the CCJJ DACs, a measure that quantifies the imbalance in the qubit’s compound-junction is used. In all cases we use the measured quantity to determine what analog signal is necessary to compensate for a change in the programmed DAC state. In this way we can determine how much flux each DAC on chip applies to its target.

4.1. DAC biasing a dc-SQUID

The couplers shown in red in figure 3 are designed to couple qubits between different unit cells. For the eight-qubit unit cell under study here, these were not connected to qubits on both ends. Instead, the inter-unit cell couplers were wired up in such a way that their compound-junction, still biased by its own DAC, could be operated as a hysteretic dc-SQUID. This coupler’s DAC could thus be used to apply flux to a dc-SQUID, and so we traced out the \( I_c \) versus \( \Phi \) threshold characteristics for this dc-SQUID as discussed in [16]. The \( I_c \) versus \( \Phi \) curve shown in figure 10 was taken both with an analog flux bias controlled directly from room temperature electronics ((a), blue dots) or using the DAC ((a), black circles). To make such a plot, we have to know the coupling constant \( k \) between the COARSE DAC stage and the dc-SQUID it biases, as well as the mutual inductance between the analog bias line and the dc-SQUID. As mentioned earlier, the mutual inductance between analog line and dc-SQUID is easily determined by observing the periodicity of the modulation curve. Also discussed earlier, the coupling constant \( k \) and division ratio \( \gamma \) are measured separately with a feedback procedure.

The two threshold curves shown in figure 10(a) begin to deviate from each other past about \( \pm 0.65 \Phi_0 \). This corresponds to where the COARSE stage of this DAC has reached its capacity, and fails to store additional flux. To be clear, the black circles are plotted versus flux programmed into the DAC COARSE stage, not flux actually applied by that DAC stage to the dc-SQUID. A plot of the latter would fall on top of the blue dots.

Figure 10(b) shows the same threshold curve versus flux programmed into the COARSE stage (black dots), but in addition, at COARSE values of \( +14 \Phi_0 \), \( +15 \Phi_0 \), and \( +16 \Phi_0 \), flux ranging from \( -6 \Phi_0 \) to \( +6 \Phi_0 \) is programmed into the FINE DAC stage as well. This is shown in the boxed region in figure 10(b), which in turn is expanded in figure 10(c). In figure 10(c), it is clear that there is sufficient range in the FINE DAC stage to bridge the COARSE DAC steps.

It is worth noticing that the range of the FINE DAC achieved from adjacent COARSE settings overlap each other. It is essential that they do not overlap, as this would result in holes in the range of flux that could be provided by the DAC. This means that there can be more than one way to obtain a particular output flux. But while the ranges overlap, the specific levels achieved from adjacent COARSE settings do not, in general, line up. Of course there is no need for them to.

To use the DAC, it is necessary for the reset function to operate. A reset pulse, a current pulse on the reset line, is made...
by first increasing the flux bias on the reset SQUID (shown in figure 5) to a predetermined level, and then lowering it back to zero. The predetermined level is chosen to be just beyond the current corresponding to a flux bias of $\Phi_0/2$ on the reset SQUID. This reset pulse was usually adequate to reset the DAC. There were cases when the $I_c$ spread of junctions for a particular DAC stage differed from each other by more than about 5%, where a single reset pulse was not sufficient, and the DAC would occasionally retain one or two $\Phi_0$ after the reset. In these cases, the reset pulse had to be repeated several times to consistently empty the DAC. While the correlation between junction spread, DAC stage $\beta$, and reset function is not quantitatively understood by us, we expect the problem to worsen with increased junction $I_c$ spread, and with increased DAC stage storage capacity ($\beta$). We found that for the chips reported upon here, it was always possible to reset all of the DAC stages by applying multiple reset pulses.

4.2. DAC applying flux bias to a qubit

Each qubit has a DAC that can apply a flux bias to its body. Figure 11 shows the flux response relative to the COARSE and FINE stages for one of these DACs. The limits of the COARSE stage capacity are just visible at the extremia of the plot. To adequately cover the range, the maximum span achievable with the FINE stage must be enough to cover one step of the COARSE stage. This coverage is clearly attained in figure 11.

4.3. DAC control of inter-qubit coupling

One of the most challenging cases to treat in the design of these DACs was the situation in which a large flux span needed to be applied to a low inductance SQUID. This situation is most extreme in the case of the coupler DAC and the $L$-tuner DAC. In fact, the dc-SQUID threshold curve presented in figure 10 above is an example of such a case—these dc-SQUIDs are patterned identically to the compound-junction used in the coupler and $I_p$-compensator.

We can also observe a DAC controlling inter-qubit coupling. Figure 12 shows a plot of the effective coupling between two qubits, via a tunable coupler, as a function of flux applied to control the coupler by an analog control line (red points). Inter-qubit coupling using the DAC to control the same coupler is shown as blue circles in the same plot. More details about the type of measurement used to obtain this plot are discussed in [14].

While presented as an inter-qubit coupler, a similar device is employed in customizing shared time-dependent signals, as discussed in section 3.3. The key difference is that in figure 7, one of the qubit ports on the coupler is connected to a global analog bias line. We found it straightforward to operating this programmable coupler in the fashion described in section 3.3.

4.4. Summary of DAC performance

Table 4(a) summarizes design targets versus achieved COARSE and FINE step sizes for the various types of DAC implemented on the eight-qubit block. Data from table 4(a) are extracted from separate break-out versions of the circuit. Uncertainties reported derive from the measurement uncertainty of that parameter for that individual device. Measuring these parameters in situ using the feedback technique described above shows device-to-device variation with a standard deviation of typically 2%. For example, the distribution of $k$ values measured for qubit flux bias DACs on a chip containing a $4 \times 4$ array of tiled unit cells, shown in figure 13, exhibits a relative standard deviation of 1.2%. This is consistent with observed variation in mutual inductance of simple microstrip transformers used in this circuit, such as might occur with variations in dielectric thickness of the same scale. Table 4(b) summarizes the maximum number of SFQ and maximum coupled flux by each DAC type.

It is worth observing that we were able to confirm that all 64 of the two-stage DACs on the single-unit-cell eight-qubit chips discussed here yielded. By yielded we mean that they behaved as expected, per tables 4(a) and (b), and that variations in coupling between identically designed DACs were of the order of 2%. Moreover, there were no significant differences in maximum storage capacity between identically designed
Figure 13. Distribution of DAC \( k \) values for 121 identically designed qubit flux bias DACs on one of the chips tested. The relative standard deviation is 1.2%.

Copies. This strongly suggests that the DAC storage coils yielded. An inter- or intra-layer short in one of the coils would likely have changed that coil’s storage capacity significantly, and this would have been observable.

Deviations between design targets and achieved parameters for this chip are as large as 30% for some of the couplings. This is primarily due to the challenge of performing sufficiently accurate 3D electromagnetic modeling of the superconducting inductors and transformers used in creating the two-stage DACs.

4.5. Demonstration of demultiplexer functionality

Delivering pulses to the DAC requires that bias current and address signals be applied to the demultiplexer tree. Bias current is shared for all demultiplexer circuits in a particular tree, and address is common for all demultiplexers at a particular level in the tree. The design and fabrication of the chip must be sufficiently uniform such that all cells work with common levels. It is also necessary that the operating margins are wide enough that a robust, low error rate operating point can be obtained.

As discussed in section 4.1, several DACs that are attached to unused boundary couplers are wired up for use as dc-SQUIDs. Operating margins required to address the FINE and COARSE DAC stages of each of these were obtained with respect to global bias current and level of address signal. These operating regions are shown in figure 14. Routing an SFQ pulse to a particular DAC requires the successful navigation of six demultiplexer gates, each with its own address current. While the signs of these various address levels may differ, their magnitude in flux was held to a common value, and this common magnitude of address flux is the address axis in figure 14.

As far as addressing these six DACs, there is clearly adequate uniformity in this demultiplexer tree that they can all be operated at a common bias current and address level. We have determined that all 64 DACs on the chip discussed here, as well as those on another subsequently tested, were addressable with chip-wide common bias current and address levels.

Figure 14 shows the boundary outside of which the probability of failing to increment a DAC stage is of order 0.1 or higher. As mentioned in section 2, we require the error rate to be considerably less than this. The dependence of error probability in SFQ circuits has been studied in some detail by a few different groups [26–28]. However, we are interested in the aggregate error probability of the entire demultiplexer tree. The probability that a pulse fails to be loaded into a DAC was measured as a function of demultiplexer bias current at nominal address level, and is shown in figure 15.

As expected, the margins decrease as the error probability requirement decreases. There is a significant bias current range with \( P_{\text{error}} < 10^{-6} \). At the chosen operating point, over 15 000 000 operations were performed, and no errors were observed. This places an upper bound on the probability of error \( P_{\text{err}} < 2.5 \times 10^{-7} \) with 95% confidence.

It is not sufficient that the address tree route pulses to the addressed DAC. It must do so exclusively, and not route pulses to any other DAC. Confirming that pulses arrived at the intended location, a requirement to attain the data shown above, does not demonstrate exclusivity.
Indeed if significantly overbiased, with no address applied, the demultiplexer tree is capable of operating in a broadcast mode, where pulses are duplicated rather than routed at each 1:2 demultiplexer node. While this should not happen under normal circumstances, failure to test for this would be an oversight.

Testing exclusivity was performed for most of the DACs on one of the chips tested at nominal bias current and address levels by (1) confirming that a DAC D was addressable and could be programmed, and then (2) sending pulses to every other DAC on that tree and confirming afterward that the state of D was unchanged. No cases of pulse misdirection were observed.

5. Conclusion

We have presented a description of a functioning system of on-chip programmable magnetic memory designed to manipulate the parameters and state of rf-SQUID superconducting qubits for use in implementing AQO algorithms. The system is inherently scalable, and in turn allows specialized AQO for use in implementing AQO algorithms. The system was designed—programmable flux DAC, programmable variable gain element, SFQ demultiplexer tree—can be used to control the devices described. The two-stage DACs used to manipulate the various controls on the qubits, and the demultiplexer address tree used to address those DACs were shown to work as intended. The address tree is shown to pass SFQ pulses with very low error rate, and to address the DACs exclusively. The two-stage DAC design was shown to be effective at providing a flux bias with a dynamic range in excess of eight bits of precision (at dc). The design targets on several variants of DAC were presented, and while the variations between designed and achieved flux coupled into target sometimes reached 30%, this is close enough to satisfy our current requirements.

While the control system described here was designed to operate an AQO processor, it is probable that the devices described—programmable flux DAC, programmable variable gain element, SFQ demultiplexer tree—can be used to control other types of quantum information processors implemented with superconductors.

References

[1] Kaminsky W M, Lloyd S and Orlando T P 2004 Scalable superconducting architecture for adiabatic quantum computation arXiv:quant-ph/0403090v2
[2] Likharev K K and Semenov V K 1991 RSFQ logic/memory family: a new Josephson junction technology for sub-terahertz clock frequency digital systems IEEE Trans. Appl. Supercond. 13–28
[3] Harrick J P, Pridmore Brown D C and Silver A 1980 Analog-to-digital conversion with unatched SQUID’s IEEE Trans. Electron. Devices 27 1887–96
[4] Abelson L A and Kerber G L 2004 Superconductor integrated circuit fabrication technology Proc. IEEE 92 1517–33
[5] Silver A, Kleinassser A, Kerber G, Herr Q, Dorojevets M, Bunyk P and Abelson L 2003 Development of superconductor electronics technology for high-end computing Supercond. Sci. Technol. 16 1368–74
[6] Yohannes D, Sarwana S, Tolpygo S K, Sahu A, Polyakov Y A and Semenov V K 2005 Characterization of HYPRES’ 4.5 ka/cm² & 8 ka/cm² Nb/Cu/Nb fabrication processes IEEE Trans. Appl. Supercond. 15 90–3
[7] Castellano M G, Chiarello F, Lenni R, Torrioli G, Carelli P, Cosmelli C, Khabipov M, Zorin A B and Balashov D 2007 Rapid single-flux quantum control of the energy potential in a double SQUID qubit circuit Supercond. Sci. Technol. 20 500–5
[8] Xingxiang Z, Habif J L, Herr A M, Feldman M J and Bocko M F 2001 A tipping pulse scheme for a rf-SQUID qubit IEEE Trans. Appl. Supercond. 11 1018–21
[9] de Castro R C R, Bocko M F, Herr A M, Mancini C A and Feldman M J 2001 Design of an RSFQ control circuit to observe MQC on an rf-SQUID IEEE Trans. Appl. Supercond. 11 1014–7
[10] Orlando T P, Lloyd S, Levitov L S, Berggren K K, Feldman M J, Bocko M F, Mooji J E, Harmans C J P and van der Wal C H 2002 Flux-based superconducting qubits for quantum computation Physica C 372–376 (Part 1) 194–200
[11] Semenov V K and Averin D V 2003 SFQ control circuits for Josephson junction qubits IEEE Trans. Appl. Supercond. 13 960–5
[12] Berggren K K 2004 Quantum computing with superconductors Proc. IEEE 92 1630–8
[13] Kaplunenko V K and Ustinov A V 2004 Experimental test of a superconducting digital interface for vortex qubits Eur. Phys. J. B 38 3–8
[14] Harris R, Lanting T, Berkley A J, Johansson J, Johnson M W, Bunyk P, Ladizinsky E, Ladizinsky N, Oh T and Han S 2009 Compound Josephson-junction coupler for flux qubits with minimal crosstalk Phys. Rev. B 80 125206
[15] Harris R et al 2010 Experimental demonstration of a robust and scalable flux qubit Phys. Rev. B 81 134510
[16] Berkley A J, Johnson M W, Bunyk P, Harris R, Johansson J, Lanting T, Ladizinsky E, Tolkacheva E, Amin M H S and Rose G 2009 A scalable readout system for a superconducting adiabatic quantum optimization system arXiv:0905.0891v1
[17] Farhi E, Goldstone J, Gutmann S, Lapan J, Lundgren A and Preda D 2001 A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem Science 292 472–5
[18] Steffen M, van Dam W, Hogg T, Bryga Y and Chuang I 2003 Experimental implementation of an adiabatic quantum optimization algorithm Phys. Rev. Lett. 90 067903
[19] Han S, Lapointe J and Lukes J E 1989 Thermal activation in a two-dimensional potential Phys. Rev. Lett. 63 1712–15
[20] Abelson L A, Daly K, Martinez N and Smith A D 1995 LTS Josephson junction critical current uniformities for FINE applications IEEE Trans. Appl. Supercond. 5 2727–30
[21] Nakada D, Berggren K K, Macedo E, Liberman V and Orlando T P 2003 Improved critical-current-density uniformity by using inductive IEEE Trans. Appl. Supercond. 13 111–4
[22] Harris R et al 2009 Implementation of a quantum annealing algorithm using a superconducting circuit arXiv:0903.3906v1
[23] Savin A M, Pekola J P, Averin D V and Semenov V K 2006 Thermal budget of superconducting digital circuits at sub-kelvin temperatures J. Appl. Phys. 99 084501
[24] http://www.wrcad.com/ftp/pub/README.FASTHENRY
[25] Lanting T et al 2009 Geometrical dependence of the low-frequency noise in superconducting flux qubits Phys. Rev. B 79 084501(R)
[26] Herr Q P and Feldman M J 1996 Error rate of a superconducting circuit Appl. Phys. Lett. 69 694–6
[27] Herr Q P, Johnson M W and Feldman M J 1999 Temperature-dependent bit-error rate of a clocked superconducting digital circuit IEEE Trans. Appl. Supercond. 9 3594–7
[28] Satchell J 1999 Limitations on HTS single flux quantum logic IEEE Trans. Appl. Supercond. 9 3841–4