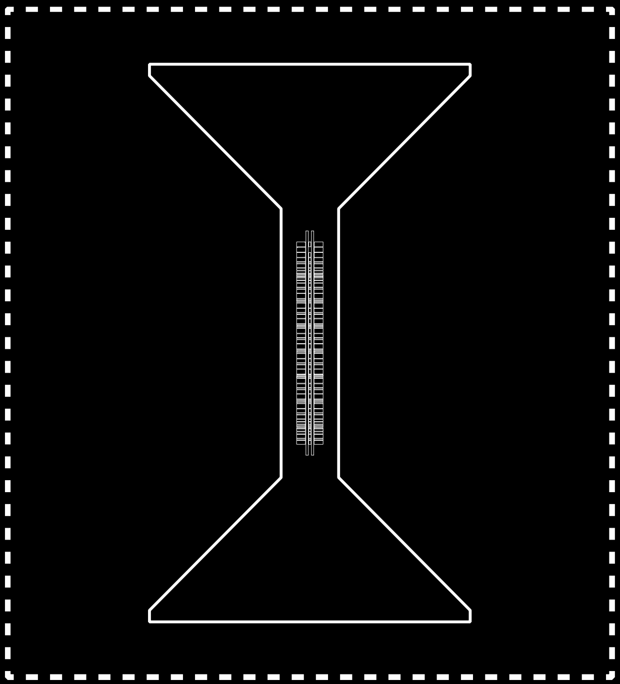

Quantinuum System Model H1

Product Data Sheet

Version 6.8.3 July 18th, 2024

TABLE OF CONTENTS

INTRODUCTION

This Product Data Sheet covers all features and characteristics of the **Quantinuum System Model H1 Emulator**.

FEATURES

- High fidelity noise models and parameters closely mimicking System Model H1 hardware performance. Each emulator uses the same physical noise model, but noise parameters reflect the performance of the device being emulated.
- Uses identical API for job submission as System Model H1, enabling seamless translation from emulator to hardware
- Uses identical compiler as System Model H1, containing all the native gates, transport operations and classical operations used in System Model H1
- Provides identical output format as System Model H1
- Allows usage of unique System Model H1 attributes: all-to-all connectivity and qubit reuse after mid-circuit measurement
- Available even while System Model H1 is offline to enable maximized productivity and development time
- TKET supported in the stack provides circuit optimization to all submitted circuits. Additional details on TKET options can be found in the Quantinuum Application Programming Interface (API) Specification.

USE CASES

The System Model H1 emulator provides a high-fidelity emulation of System Model H1. Use cases include:

- Debugging of quantum code before running on physical hardware
- Optimization of quantum code in the presence of noise mechanisms
- Exploring new algorithms and techniques for quantum error correction
- Introduction to System Model H1 and its unique differentiating capabilities such as qubit reuse after mid-circuit measurement, all-to-all connectivity, and high-fidelity gates

FUNCTIONAL REQUIREMENTS

The System Model H1 emulator is meant to be a functional emulation of System Model H1 and therefore supports the same functional operations as H1. Specifically, the System Model H1 emulator supports:

- OPENQASM 2.0 circuits
- Quantinuum QASM enhancements, including classical logic, math, and program flow control
- Quantinuum native gate set[1]
- Common compound gates from OPENQASM library, e.g., CX, H
- User-defined compound gates
- User option of noiseless simulation or inclusion of System Model H1 noise models

---

[1] For definition of native gates, please request a copy of the *Quantinuum System Model H1 Product Data Sheet*

- Large quantum circuits with a limit of 10,000 on the number of shots
- Identical queuing prioritization as System Model H1

EMULATOR ACCESS AND OUTPUT

Communication with the System Model H1 emulator occurs through an API endpoint based on the OpenQASM 2.0 standard (Cross, Lev, John, & Jay, 2017). Interface details are given in the *Quantinuum Application Programming Interface (API) Specification.*

Users can select a System Model H1 emulator in the machine list API, designated with the “E” suffix machine name. The output of H1 Emulator is a JSON-formatted array, identical to the output format of System Model H1. Through the Job Submission API, users may select the type of emulator used and turning on or off the application of the error model.

PERFORMANCE

The performance of the System Model H1 emulator is measured in the fidelity to hardware. With inclusion of accurate and up-to-date noise models, the System H1 Emulator can provide a high-fidelity representation of System Model H1 output. Fidelity is verified at Quantinuum by comparison between the emulator and hardware outputs. However, noise models cannot fully capture the behavior of System Model H1; users should expect some variance. In the case of exceptional or unexplained variance, users should contact Quantinuum technical support at QCsupport@quantinuum.com to discuss the circuit and results.

EMULATION METHOD

The System Model H1 emulator, accessible via the API, receives instructions directly from the same compilers used by the System Model H1 physical quantum hardware. These compilers translate the submitted quantum program into a set of instructions comprising of the native gate operations and the transport operations necessary to reconfigure the ion chain at each step of the program.

Users can choose between either a state vector or stabilizer emulation method; in both cases results are performed shot-by-shot. The state vector emulation method can run any general quantum circuits, while the stabilizer emulation method is restricted to circuits involving only quantum unitary gates that are Clifford operations.

The error model for the emulation can be turned on or off, allowing noisy or noise-free emulations, respectively. The emulated error model includes:

- asymmetric depolarizing gate noise
- leakage errors
- crosstalk noise
- dephasing noise due to transport and qubit idling

Except for dephasing, errors on physical qubits are modeled as stochastic processes. For the state vector emulation, dephasing is handled as a coherent Z rotation according to a dephasing rate and the duration the qubit spends in transport or while idling while other qubits are being gated. For the stabilizer emulation, the dephasing noise is treated as a stochastic Z error where the probability of a Z error is equal to the Pauli twirled approximation of the coherent dephasing channel, which is proportional to the square of the dephasing rate multiplied by the duration.

NOISE MODEL

Users who have direct access to the Quantinuum API have the option of experimenting with the physical noise parameters of the emulator. When deviating from the default emulation model, users should not assume that performance predicted with modified error parameters will match hardware performance.

All parameters listed in Table 1 are the default settings of the System Model H1 emulators. As updates to the System Model H1 quantum computers are made, the emulator noise parameters and the underlying error model are subject to change to accommodate performance improvements, updates in the methodology for measuring devices parameter and research into the noise sources themselves

All the errors are applied even when only certain parameters are specified. Only the parameters specified are overridden. To turn off certain error parameters, explicitly set them to 0.

For more information on the errors observed, see the following publications: Realization of Real-Time Fault-Tolerant Quantum Error Correction, Implementing Fault-tolerant Entangling Gates on the Five-qubit Code and the Color Code.

Table 1 Default Settings of the System Model H1 Emulators

Default Settings	H1-1	H1-2
General		
Qubits	20	20
Connectivity	All-to-all	All-to-all
Parallel two-qubit operations	5	5
Physical Noise		
Single-Qubit Fault Probability (p1)	2.1×10^{-5}	5.45×10^{-5}
Two-Qubit Fault Probability (p2)	8.8×10^{-4}	2.97×10^{-3}
Bit Flip Measurement Probability (0 outcome) (p_meas)	1.0×10^{-3}	1.17×10^{-3}
Bit Flip Measurement Probability (1 outcome) (p_meas)	4.0×10^{-3}	5.26×10^{-3}
Crosstalk Measurement Fault Probability (p_crosstalk_meas)	1.45×10^{-5}	3.4×10^{-5}
Initialization Fault Probability (p_init)	3.62×10^{-5}	3.62×10^{-5}
Crosstalk Initialization Probability (p_crosstalk_init)	5.020×10^{-6}	5.020×10^{-6}
Ratio of Single-Qubit Spontaneous Emission to p1 (p1_emission_ratio)	0.54	0.488

Ratio of Single-Qubit Spontaneous Emission in Two-Qubit Gate to p2 (p2_emission_ratio)	0.43	0.206
Dephasing Noise		
Quadratic Dephasing Rate (quadratic_dephasing_rate)	0.122	0.122
Linear Dephasing Rate (linear_dephasing_rate)	0.0	0.0
Coherent to Incoherent Factor (coherent_to_incoherent_factor)	2.5	2.5
Arbitrary Angle Noise Scaling		
Fit Parameter 1 (przz_a)	1.651	1.651
Fit Parameter 2 (przz_b)	0.175	0.175
Fit Parameter 3 (przz_c)	1.651	1.651
Fit Parameter 4 (przz_d)	0.175	0.175
Polynomial (przz_power)	1.0	1.0

PHYSICAL NOISE

The emulator runs with default error parameters that represent a noise environment that closely resembles the respective hardware. These error parameters can be set and used to override the default error parameters and do finer-grain tweaks of the error model. Modification of the error parameters away from default values is an advanced option and not recommended as a starting point for emulations of hardware performance.

- **Single-Qubit Fault Probability (p1):** probability of a fault occurring during a single-qubit gate
- **Two-Qubit Fault Probability (p2):** probability of a fault occurring during a two-qubit gate
- **Bit Flip Measurement Probability (p_meas):** probability of a bit flip being applied to a measurement. Either a float or a tuple of 2 floats. If it is a single float then that error rate is used to bitflip both 0 and 1 measurement results. If a tuple is supplied, the first element is the probability a bit flip is applied if a 0 result occurs during measurement while the second error rate if a 1 is measured.
- **Crosstalk Measurement Fault Probability (p_crosstalk_meas):** probability of a crosstalk measurement fault occurring
- **Initialization Fault Probability (p_init):** probability of a fault occurring during initialization of a qubit
- **Crosstalk Initialization Fault Probability (p_crosstalk_init):** probability of a cross-talk fault occurring during initialization of a qubit
- **Ratio of Single-Qubit Spontaneous Emission to p1 (p1_emission ratio):** fraction of p1 that is spontaneous emission for a single qubit instead of asymmetric depolarizing noise
- **Ratio of Single-Qubit Spontaneous Emission in Two-Qubit Gate to p2 (p2_emission_ratio):** fraction of p2 that is spontaneous emission for a single qubit in a two-qubit gate instead of asymmetric depolarizing noise

The single and two-qubit fault probabilities are largely modeled using asymmetric depolarizing channels; however, there is smaller probability that a spontaneous emission event happens. The probability is about an order of magnitude lower than the corresponding asymmetric depolarizing error rate. The spontaneous emission error rates can be scaled using the scaling parameters given in the Scaling section. If a spontaneous emission event happens then ¼ the time X is applied, ¼ the time Y is applied, and ½ the time leakage is applied. For more details see: Realization of Real-Time Fault-Tolerant Quantum Error Correction.

The two-qubit fault probability corresponds to the asymmetric depolarizing probability of the System Model H2 fully entangling two-qubit gate, $ZZ()$. The probability of asymmetric depolarizing error for the arbitrary angle two-qubit gate, $RZZ(\theta)$, depends on the angle θ. The spontaneous emission error channel is the same for both $ZZ()$ and $RZZ(\theta)$.

DEPHASING NOISE

The noise model includes a memory error for which Z is applied. This is often called "dephasing" or "memory" noise and depends on the duration for which the qubits are idling or transporting in the trap. We potentially model two types of dephasing noise: one where the probability of applying Z is quadratically dependent on the duration and another where the probability is linearly dependent on the duration. Note, we apply both sorts of noise simultaneously. For state vector simulations, the quadratic noise is modeled in the emulator by default as coherent noise. For this coherent quadratic dephasing noise, the RZ gate is applied with an angle proportional to quadratic dephasing rate multiplied by the duration. The resulting probability of the RZ gate applying a Z operation on a plus state is $\sin(\text{rate x duration}/2)^2$, which is why we call this a form of quadratic dephasing

For the stabilizer simulator, by default this quadratic noise is modeled incoherently by applying Pauli Z with probability, $\sin(\text{frequency x duration}/2)^2$, to model more closely the quadratic dependency with frequency and time, as seen in the coherent model. Note, stabilizer simulations can only simulate Clifford and measurement-like gates, so the RZ gate cannot be applied directly.

For both state vector and stabilizer simulations, linear dephasing is modeled with Z applied using a probability equal to the linear dephasing rate multiplied by the duration.

Switching between the coherent and incoherent quadratic dephasing model can be accomplished by setting *coherent_dephasing* either True or False. As mentioned, *coherent_dephasing* is True by default for the state vector simulations and False by default for stabilizer simulations. If *coherent_dephasing* is set to False then the frequency for the quadratic error model (*quadratic_dephasing_rate*) is multiplied by *coherent_to_incoherent_factor* to attempt to make up for increased noise due to coherent effects; however, how sensitive circuits are to coherent effects depends on the circuit. Therefore, users may want to adjust this factor appropriately.

In addition, a transport dephasing parameter (*transport_dephasing*) and an idle dephasing parameter (*idle_dephasing*) are both turned on by default. Both can be toggled off.

- **Coherent Dephasing (*coherent_dephasing*):** A boolean value determining whether quadratic dephasing is applied (default: True).
- **Coherent Quadratic Dephasing Model:** the gate RZ (frequency x duration) is applied during transport and qubit idling where frequency is equal to *quadratic_dephasing_rate* (units of 2π radians per second). This model is used if *coherent_dephasing* is True. Applied by default for the state vector simulator.

- **Quadratic Dephasing Rate (quadratic_dephasing_rate):** The frequency, f, in applying $RZ\,(\text{frequency x duration})$ during transport and idling.
- **Incoherent Quadratic Dephasing Model:** Pauli Z is applied during transport and qubit idling according to the probability $\sin(\text{frequency x duration}/2)^2$ where frequency is equal to *quadratic_dephasing_rate* multiplied by the *coherent_to_incoherent_factor* (all in units of 2π radians per second). This model is used if *coherent_dephasing* is False. This model is mostly used to mimic coherent dephasing noise for stabilizer simulations and is applied by default for the stabilizer simulator.
    - ***Incoherence Multiplier (coherent_to_incoherent_factor):*** A multiplier on the quadratic term when running stabilizer simulations to attempt to account for increases in error due to coherent effects in the circuit.
- **Linear Dephasing Model:** Pauli Z is applied during transport and qubit idling according to the probability of linear_dephasing_rate x duration where *linear_dephasing_rate* is per second (s^{-1}), and duration is in units of seconds. This model is used in conjunction with either the coherent or incoherent quadratic dephasing model.
- **Linear Dephasing Rate (*linear_dephasing_rate*)**: The probability of applying Z with $p = rd$ where r is rate and d is duration. This models the memory error. Note both the quadratic and linear term can be applied in the same simulation.
- **Transport Dephasing (*transport_dephasing*):** A boolean affecting whether memory noise is applied during transport.
- **Idle Dephasing (*idle_dephasing*):** A boolean affecting if memory noise is applied due to qubit idling.

ARBITRARY ANGLE NOISE SCALING

The System Model H2 systems have a native arbitrary-angle ZZ gate, $RZZ(\theta)$. For implementation of this gate in the System Model H2 emulator, certain parameters relate to the strength of the asymmetric depolarizing noise. These parameters depend on the angle θ. This is normalized so that $\theta = \frac{\pi}{2}$ gives the two-qubit fault probability (p2).

The parameters for asymmetric depolarizing noise are fit parameters that fit the noise estimated as the angle θ changes per this equation:

$$\left(przz_a * (|\theta|/\pi)^{przz_{power}} + przz_b\right) * p2 \quad \theta < 0$$

$$\left(przz_c * (|\theta|/\pi)^{przz_{power}} + przz_d\right) * p2 \quad \theta > 0$$

$$(przz_b + przz_d) * 0.5 \quad \theta = 0$$

- Fit Parameter 1 (przz_a)
- Fit Parameter 2 (przz_b)
- Fit Parameter 3 (przz_c)
- Fit Parameter 4 (przz_d)
- Polynomial (przz_power)

SCALING

A scaling factor can be applied that multiplies all the default or supplied error parameters by the scaling rate. In this case, a 1 does not change the error rates while 0 makes all the errors have a probability of 0. Other aspects of the noise model can scale specific error rates in the error model, which include:

- **Scaling (scale):** scale all error rates in the model linearly
- **P1 Scaling (p1_scale):** scale the probability of single-qubit gates having a fault
- **P2 Scaling (p2_scale):** scale the probability of two-qubit gates having a fault
- **Measurement Scaling (meas_scale):** scale the probability of measurement having a fault
- **Initialization Scaling (init_scale):** scale the probability of initialization having a fault
- **Memory Scaling (memory_scale):** linearly scale the probability of dephasing causing a fault
- **Emission Scaling (emission_scale):** scale the probability that a spontaneous emission event happens during a single or two-qubit gate
- **Cross-talk Scaling (crosstalk_scale):** scale the probability that measurement or initialization crosstalk events get applied to qubits, during mid-circuit measurement and reset (initialization), "crosstalk" noise can occur that effectively measures other qubits in the trap or cause them to leak.
- **Leakage Scaling (leakage_scale):** scale the probability that a leakage even happens during single or two-qubit gates as well as during initialization or crosstalk; on the device half the time, spontaneous emission leads to a leakage event

APPENDIX

A H-System Quantum Credit (HQC) is defined as:

$$HQC = 5 + \frac{N_{1q} + 10\,N_{2q} + 5\,N_m}{5000} C$$

where N_{1q} is the number of single-qubit gates, N_{2q} is the number of native two-qubit gates, N_m is the number of state preparation and measurement operations in a circuit, including the initial implicit state preparation and any intermediate and final measurements and state resets, and C is the shot count. When a circuit is submitted, whether to a quantum computer, syntax checker, or emulator, the cost in HQCs is returned with the results.

REFERENCES

Cross, A. W., Lev, B. S., John, S. A., & Jay, G. M. (2017). Open Quantum Assembly Language. *arXiv:1707.03429v2*.