Thanks for the credits!

Tutorial for ***SimuQ***

a DSL for Quantum Simulation with Analog Compilation

Presenter: Yuxiang Peng, Pengyu Liu, and Jiaqi Leng

09/20/2023

A joint work with Jacob Young and Xiaodi Wu

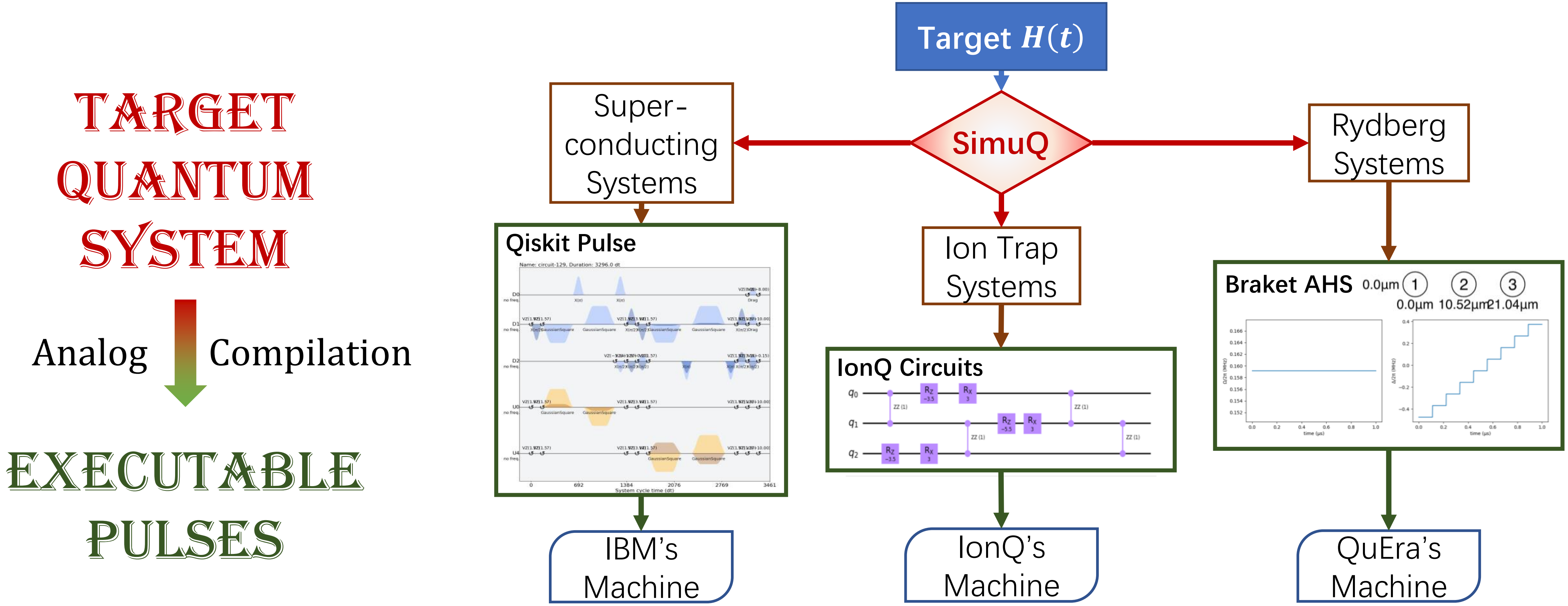
Thanks to contributors: Cedric Lin, Cody Wang, Joseph Li

GitHub repo:
https://github.com/PicksPeng/SimuQ

arXiv: 2303.02775

Project website:
https://pickspeng.github.io/SimuQ/

SimuQ: SIMULation language for Quantum



A software tool for

Hamiltonian-oriented quantum computing!

Tutorial Outline

- **First session**
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- **Break**
- **Second session**
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min)
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

Tutorial Outline

- **First session**
  - Motivation & introduction (~40 min) Presenter: Yuxiang Peng
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- **Break**
- **Second session**
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min)
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

Quantum Hamiltonian Simulation

Quantum systems

$|\psi(0)\rangle$ → **Schrödinger equation** $|\dot{\psi}\rangle = -iH(t)|\psi\rangle$ → $|\psi(T)\rangle$

$$\frac{1}{\sqrt{2}}\begin{pmatrix} i \\ 0 \\ 0 \\ -1 \end{pmatrix}$$

$$H(t) = Z_1 Z_2$$

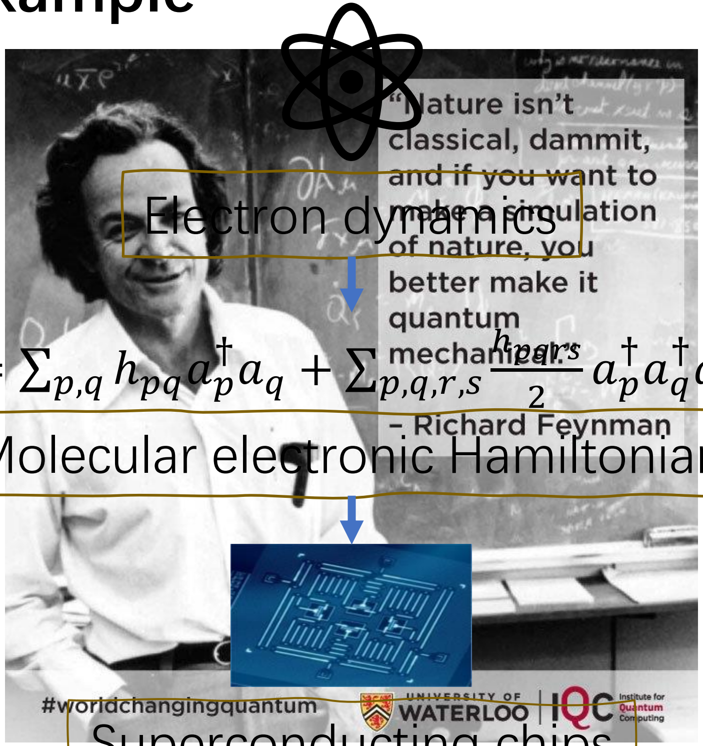
$$\begin{pmatrix} 1 & & & \\ & -1 & & \\ & & -1 & \\ & & & 1 \end{pmatrix}$$

?

$Z_j = I \otimes \cdots \otimes I \otimes Z \otimes I \otimes \cdots \otimes I$: Pauli Z operator on j-th site

Example

Hard to control



Electron dynamics

↓

$$H = \sum_{p,q} h_{pq} a_p^\dagger a_q + \sum_{p,q,r,s} \frac{h_{pqrs}}{2} a_p^\dagger a_q^\dagger a_r a_s$$

Molecular electronic Hamiltonian

↓

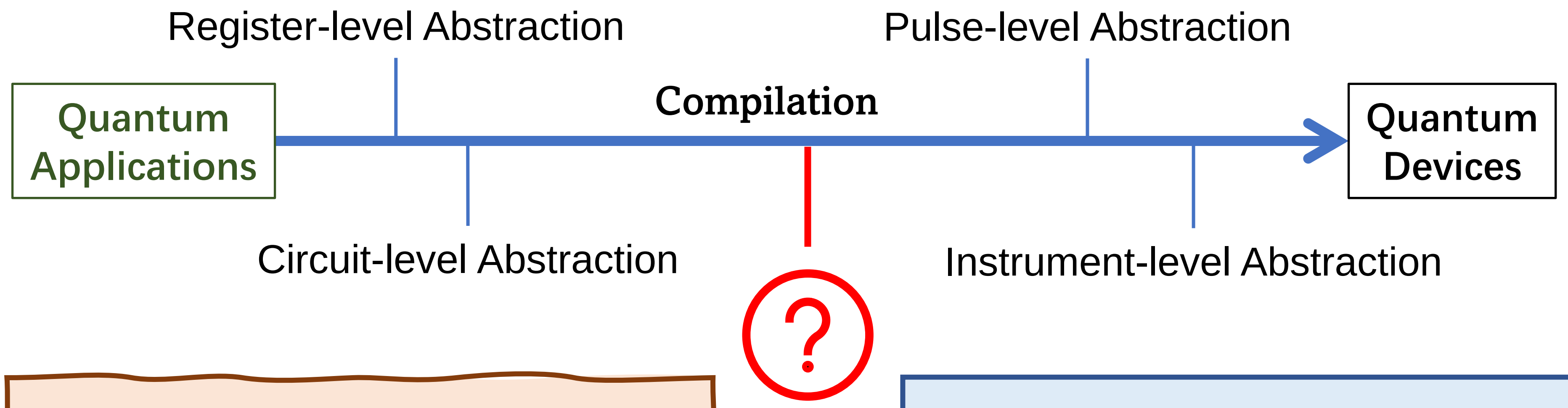
Superconducting chips

Easy to control

Classically hard to compute: Exponential dimension

Quantumly "easy" to simulate: Map to a controllable quantum system

Towards Quantum Applications



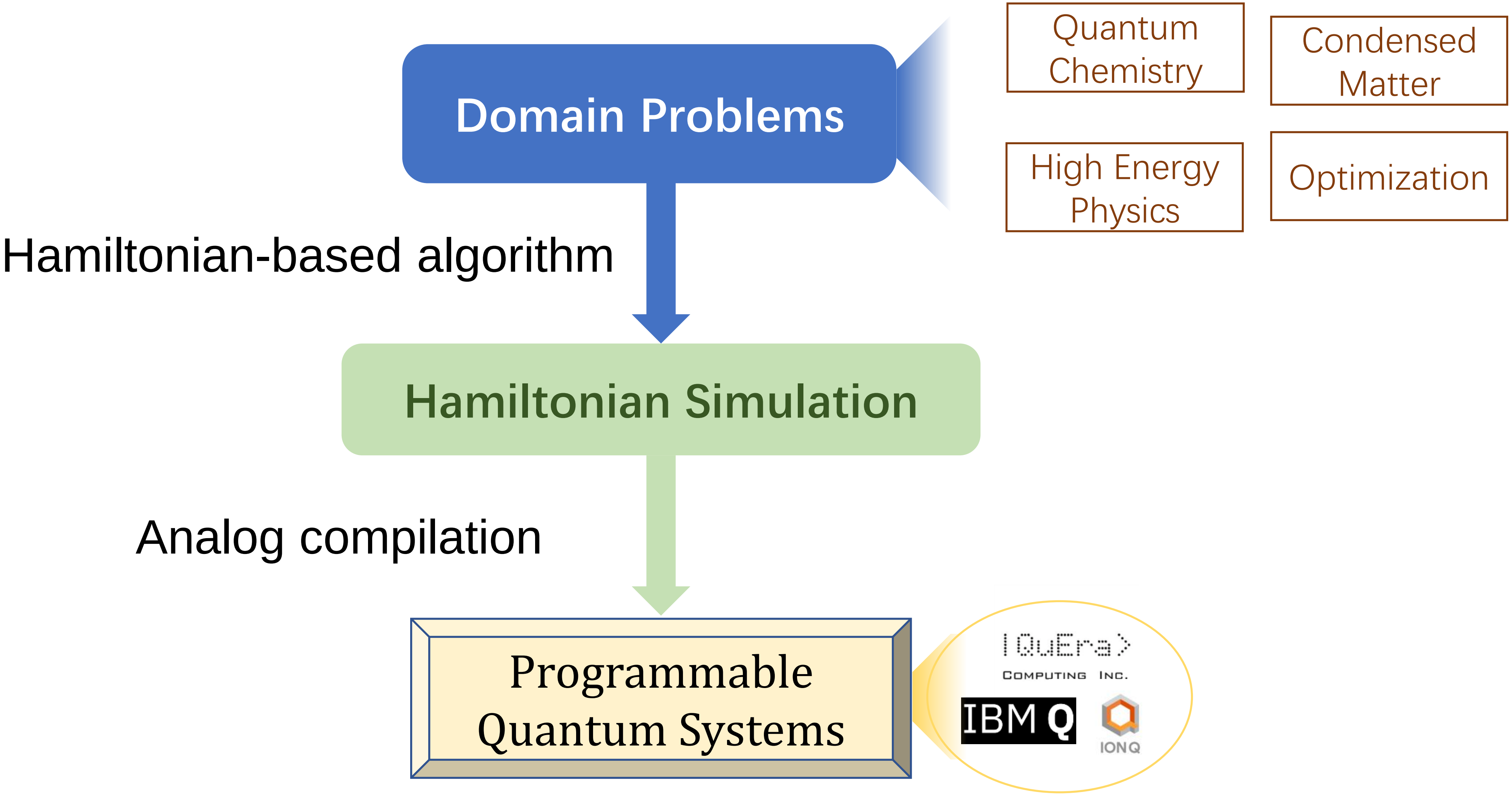
Large overheads
Digital information hard to control

Intermediate Abstraction
for Physics-Related Problems

Goal: Deliver quantum applications on modern quantum devices

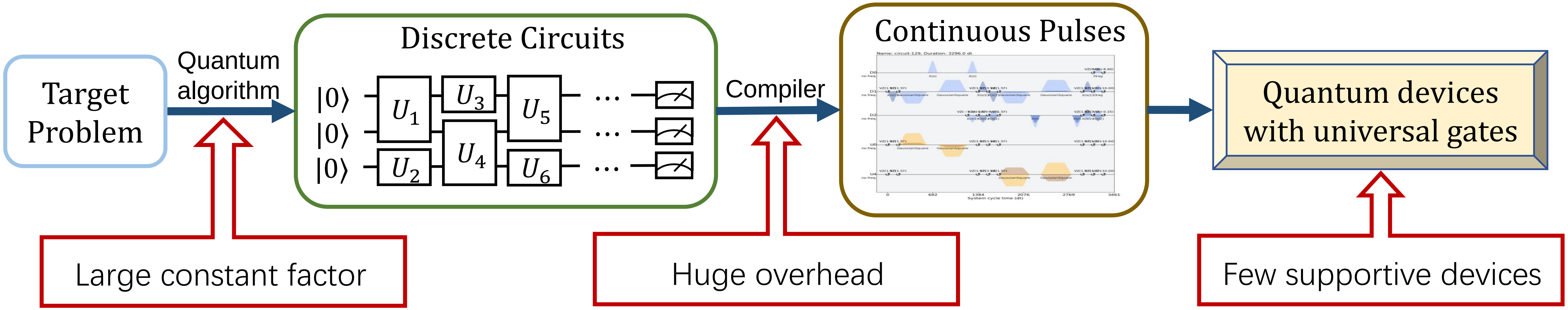
↳ **Hamiltonian-oriented** algorithm designs and programming!

Hamiltonian-Oriented Quantum Computing

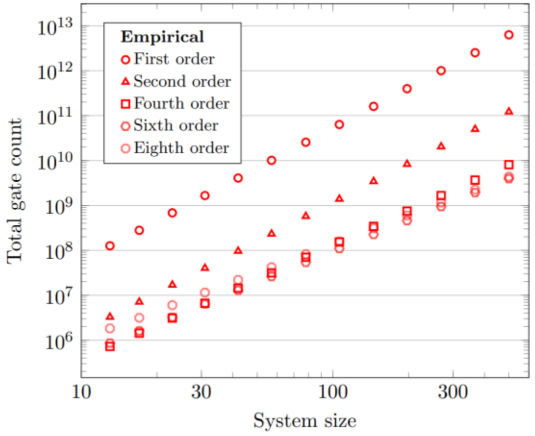


Why Hamiltonian-Oriented?

Digital Quantum Computing Paradigm



[1]:



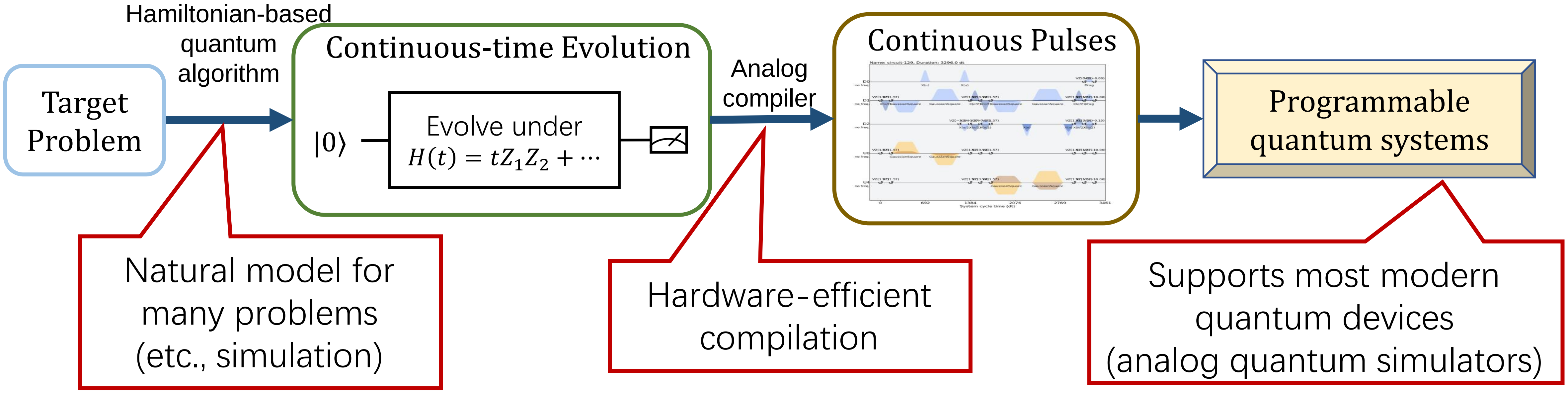
On IBM devices:
$10 \cdot T_{\text{CNOT}} \approx T_1 \approx T_2$
$T_{R_{XX}(\theta)} \approx 2 \cdot T_{\text{CNOT}}$

Available:
Superconducting
Ion Trap

[1]: Childs et al., Toward the first quantum simulation with quantum speedup, PNAS, 2018.

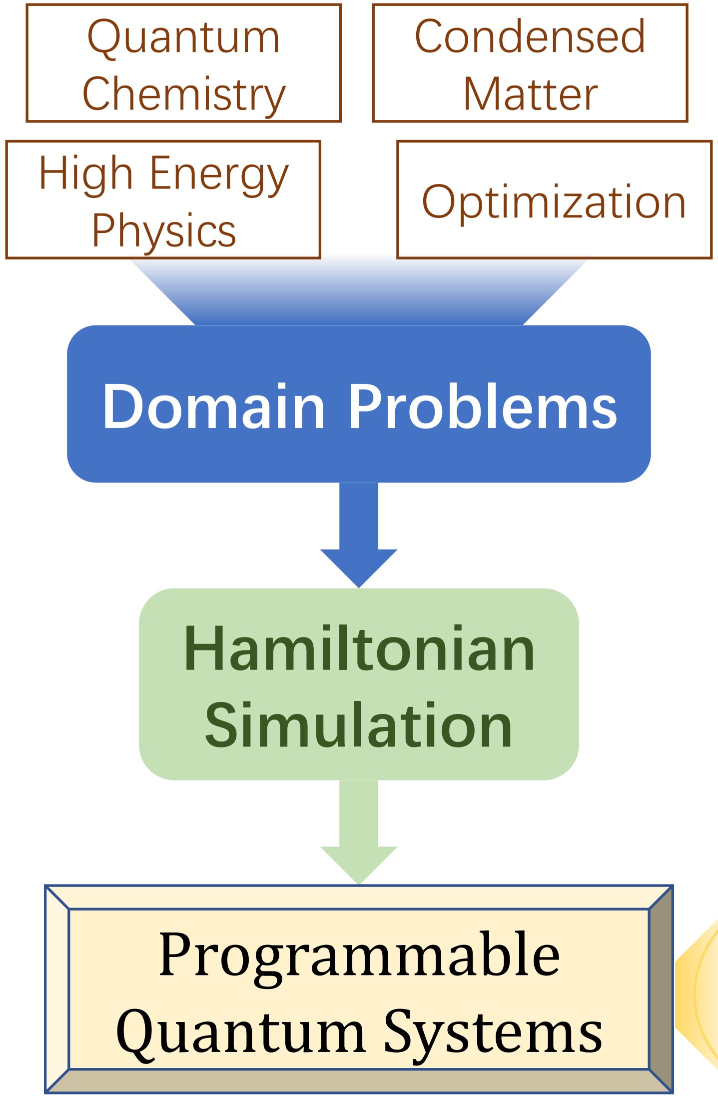
Why Hamiltonian-Oriented?

Analog (Hamiltonian-oriented) Quantum Computing Paradigm

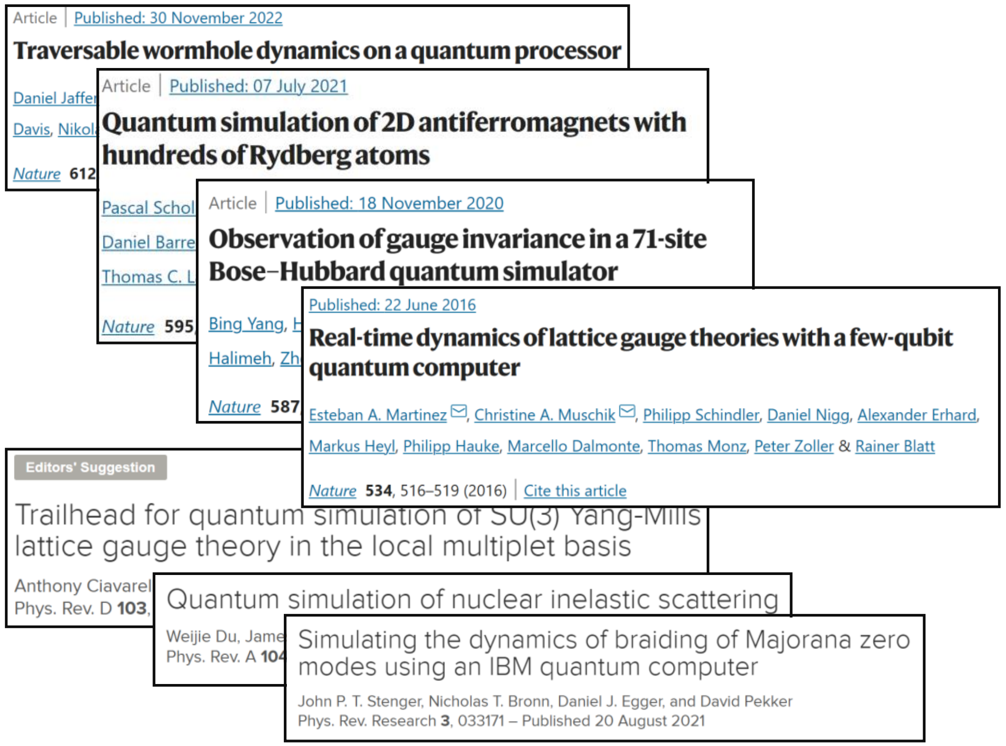


Ready for modern quantum devices!

A New Trend in Scientific Researches



>20 papers published since 2020 in Physics Review & Nature series



...

Quantum Simulation for Everyone

Quantum mechanics

Undergrad

I want to simulate

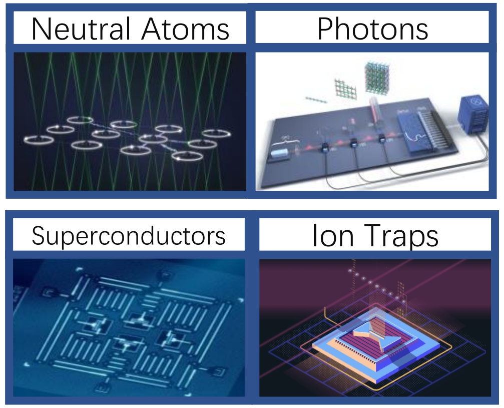
$$X_1X_2 + \cdots + X_{99}X_{100}$$
$$+Z_1 + \cdots + Z_{100}$$

Reality:

Lack of software supports

SimuQ *breaks barriers!*

Quantum Devices



Domain-specific knowledges

- Analog or digital devices?
- Device physics and specifications?
- Quantum algorithms?
- Programming languages?
- ……

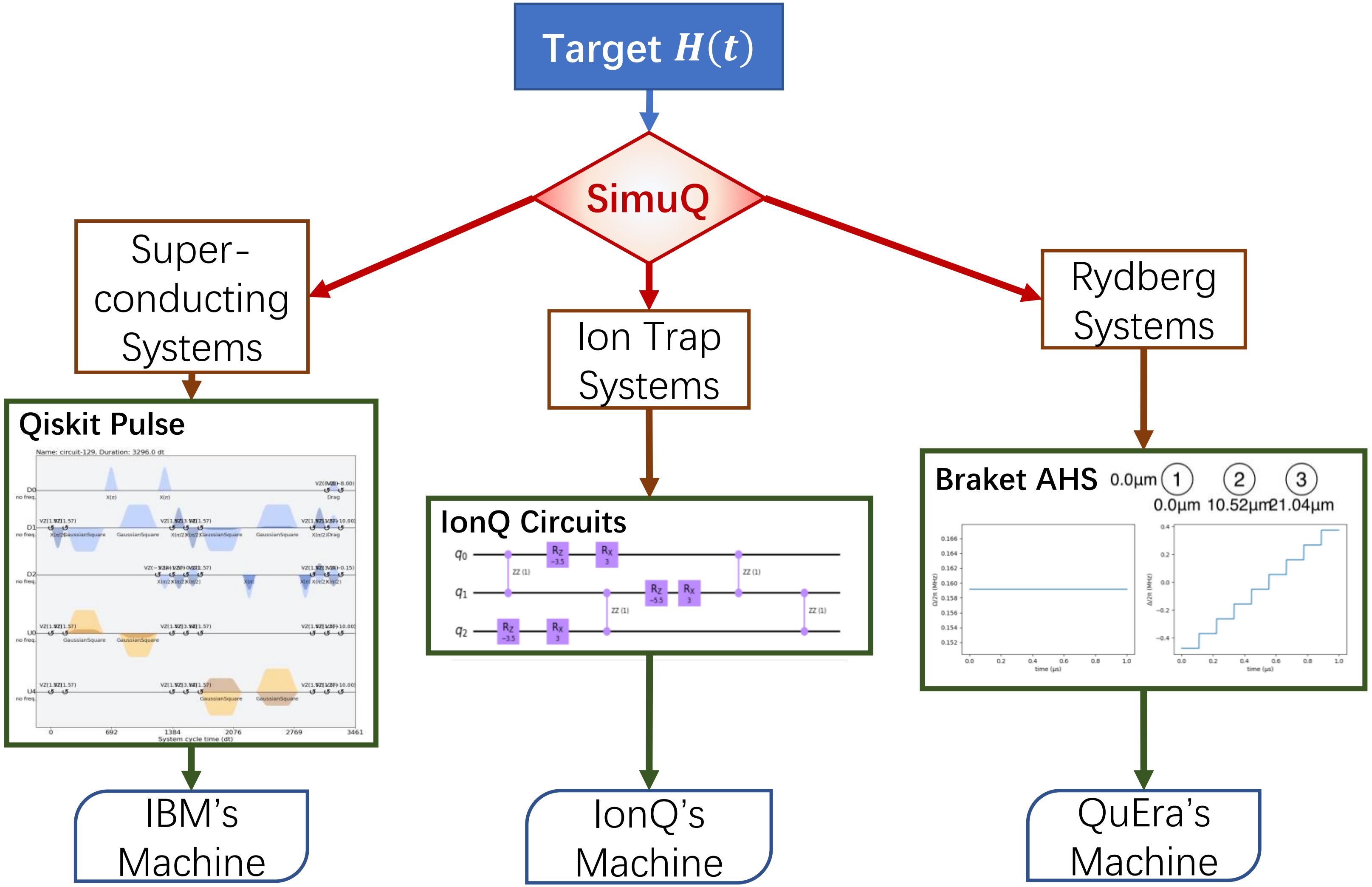
SimuQ: SIMUlation language for Quantum

TARGET QUANTUM SYSTEM

Analog Compilation

EXECUTABLE PULSES

- Analog instruction sets
- Hardware-efficient compilation
- Supports analog simulators



Example: Quantum Walk

$$L = \begin{bmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{bmatrix}$$

Simulate $U(t) = e^{-iLt}$

Circuit-based implementation

1. Decompose L into Pauli tensors
2. Apply Trotterization
3. Decompose into CNOT-based circuits

$\Rightarrow$ ~50 CNOT gates per step

Close to the limits of modern devices

Hamiltonian-Oriented Algorithm Design

Hamiltonian embedding [1] $(n = N - 1)$:

$$H = g\left(\sum_{j=1}^{n-1} Z_j Z_{j+1} + Z_1 + (-1)^n Z_n\right) \quad \text{Penalty}$$

$$+\hat{n}_1 + (-1)^n \hat{n}_n - \sum_{j=1}^{n} X_j \quad \text{Perturbation}$$

Perturbation theory: $e^{-iHt}|_{\mathcal{H}} \approx e^{-iLt}$

$\mathcal{H} = \mathrm{span}(\{|5\rangle, |13\rangle, |9\rangle, |11\rangle, |10\rangle\})$:

Can be deployed on Rydberg devices (up to 256 atoms)

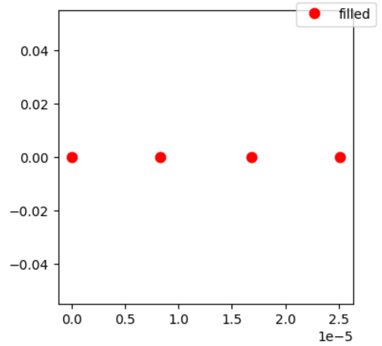
[1]: Leng et al., Expanding Hardware-Efficiently Manipulable Hilbert Space via Hamiltonian Embedding, in preparation

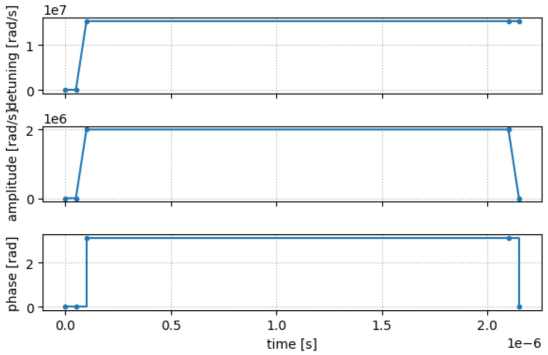
Example: Quantum Walk

```
walk = QSystem()

n = N - 1
q = [qubit(walk) for _ in range (n)]
nhat = [(q[j].I - q[j].Z) / 2 for j in range(n)]
sgnn = -1 if n % 2 == 1 else 1

Hpen = q[0].Z + sgnn * q[n - 1].Z
for j in range(n - 1) :
    Hpen += q[j].Z * q[j + 1].Z

Q = nhat[0] + sgnn * nhat[n - 1]
for j in range(n) :
    Q -= q[j].X

H = g * Hpen + Q
walk.add_evolution(H, T)
```

Programming

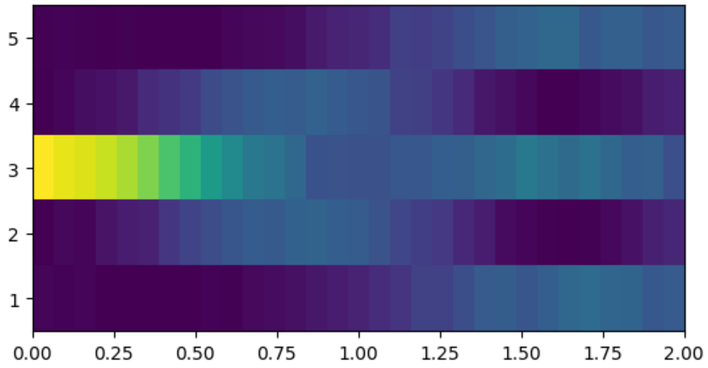
Compile





Visualization

Execution



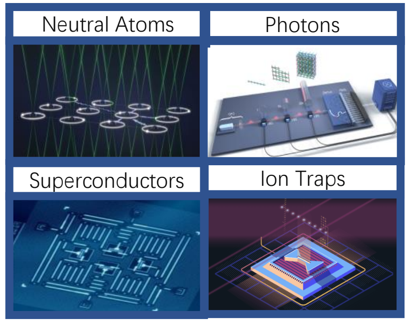
initial state $|00100\rangle$

Results

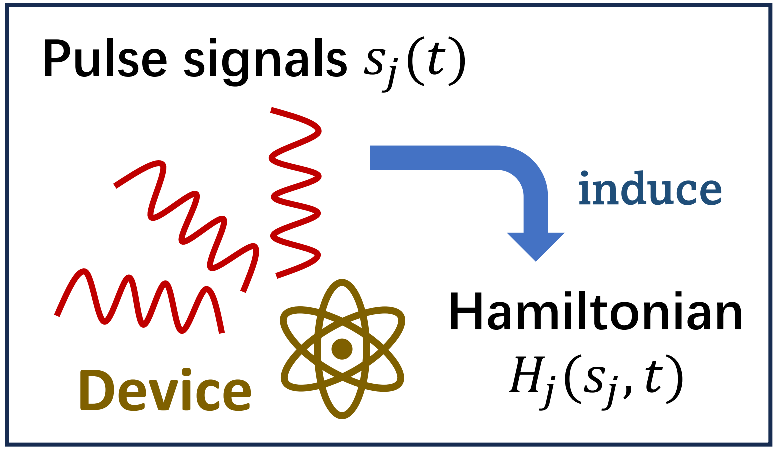
Automated by SimuQ

More details in Notebook session 2

Abstraction for Analog Quantum Simulators



Different devices are vastly different → A unifying description



Device Hamiltonian:

$$H(t) = H_0(t) + \sum_j H_j(s_j, t)$$

Device evolution:

$$U = \mathcal{T}\exp(-i \int H(t)\mathrm{d}t)$$

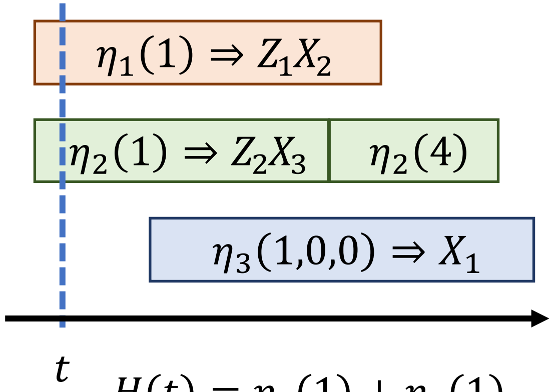
ABSTRACT

Abstract Analog Instruction Set

- $\eta_1(a) \Rightarrow aZ_1X_2$
- $\eta_2(a) \Rightarrow aZ_2X_3$
- $\eta_3(a, b, c) \Rightarrow aX_1 + bY_1 + cZ_1$
- ⋮

Parameterized Hamiltonians

Schedule



$$H(t) = \eta_1(1) + \eta_2(1) = Z_1X_2 + Z_2X_3$$

Device evolution:

$$U = \mathcal{T}\exp(-i \int H(t)\mathrm{d}t)$$

Example: Rydberg AAIS

Configurable Parameters:

1. Atom position $\{\tilde{x}_j\}_{j=1}^{N}$.
2. Local laser configurations $\left\{\left(\tilde{\Delta}_j(t), \tilde{\Omega}_j(t), \tilde{\phi}_j(t)\right)\right\}_{j=1}^{N}$

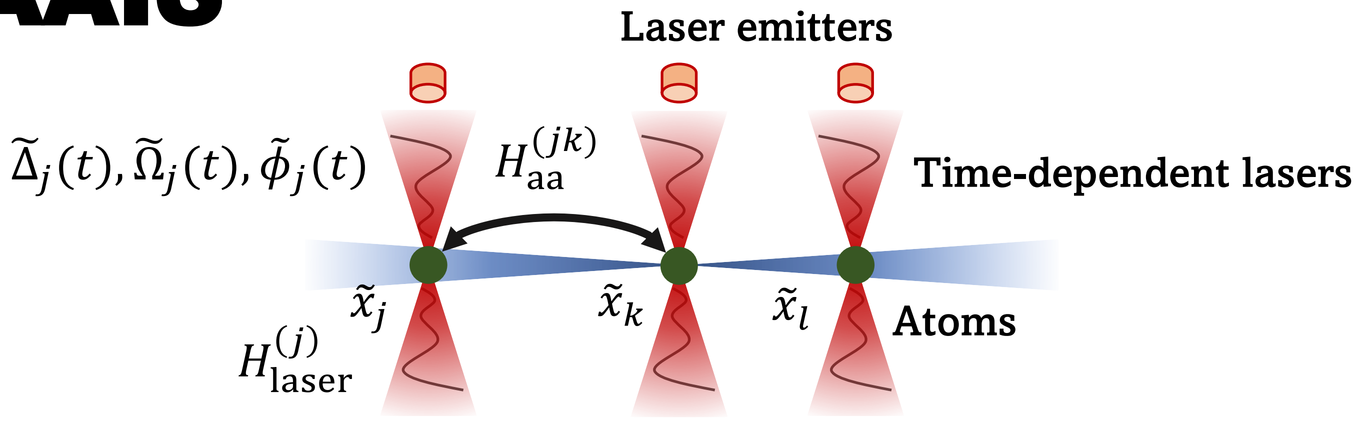
Device Hamiltonian:

$$H(t) = \sum_{j<k} H_{\text{aa}}^{(jk)} + \sum_j H_{\text{laser}}^{(j)}(t)$$

$$H_{\text{aa}}^{(jk)} = \frac{C_6}{|\tilde{x}_j - \tilde{x}_k|^6} \hat{n}_j \hat{n}_k$$

$$H_{\text{laser}}^{(j)}(t) = \tilde{\Delta}_j(t) \hat{n}_j + \frac{\tilde{\Omega}_j(t)}{2} H_{\text{phase}}^{(j)}\left(\tilde{\phi}_j(t)\right)$$

$$H_{\text{phase}}^{(j)}(\varphi) = \cos(\varphi) X_j - \sin(\varphi) Y_j$$



ABSTRACT

AAIS

Global variables

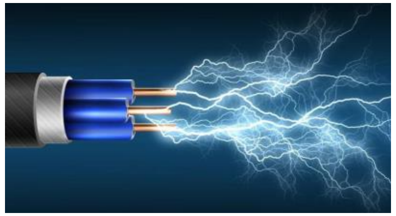
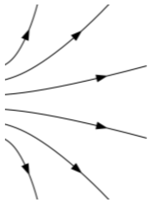
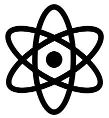
$$\eta_{\text{system}}(\vec{x}) \Rightarrow \sum_{j<k} H_{\text{aa}}^{(jk)}$$

$$\forall j, \quad \eta_{\text{laser}}^{(j)}(\Delta, \Omega, \phi) \Rightarrow \Delta \hat{n}_j + \frac{\Omega}{2} H_{\text{phase}}^{(j)}(\phi)$$

Local variables

More details in Advanced discussion

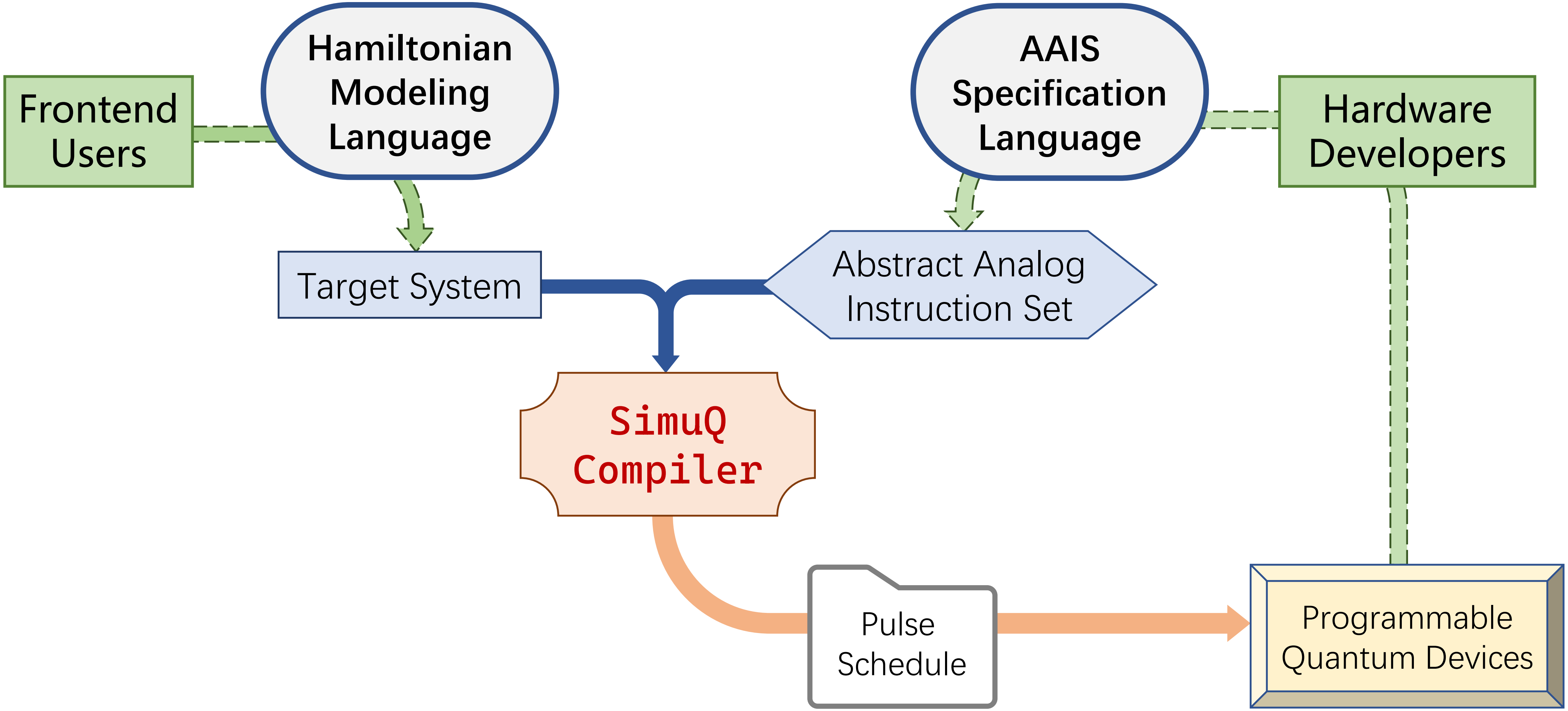
Abstract Analog Instruction Set (AAIS)

	Carriers	Engineered Signals	Signal Effects	Device Evolution
Physics				
Abstraction	**Signal Lines**	**Instructions**	**Instruction Ham.**	**Total Ham.**

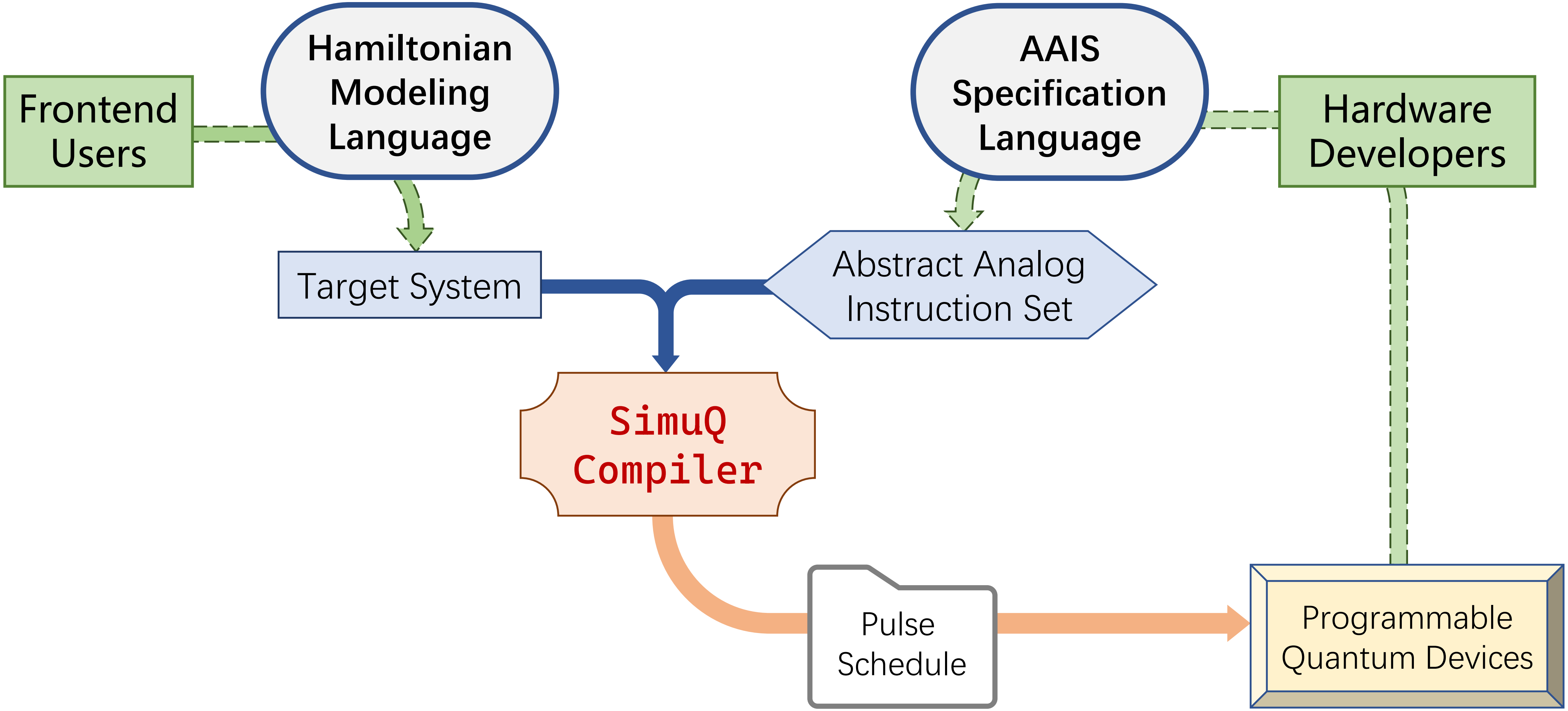
- Exposes programmability of analog quantum simulators.
- Can be programmed via AAIS-SL.
- Enables automatic analog compilation.
- A new computational model.

Generalize circuit-based quantum devices

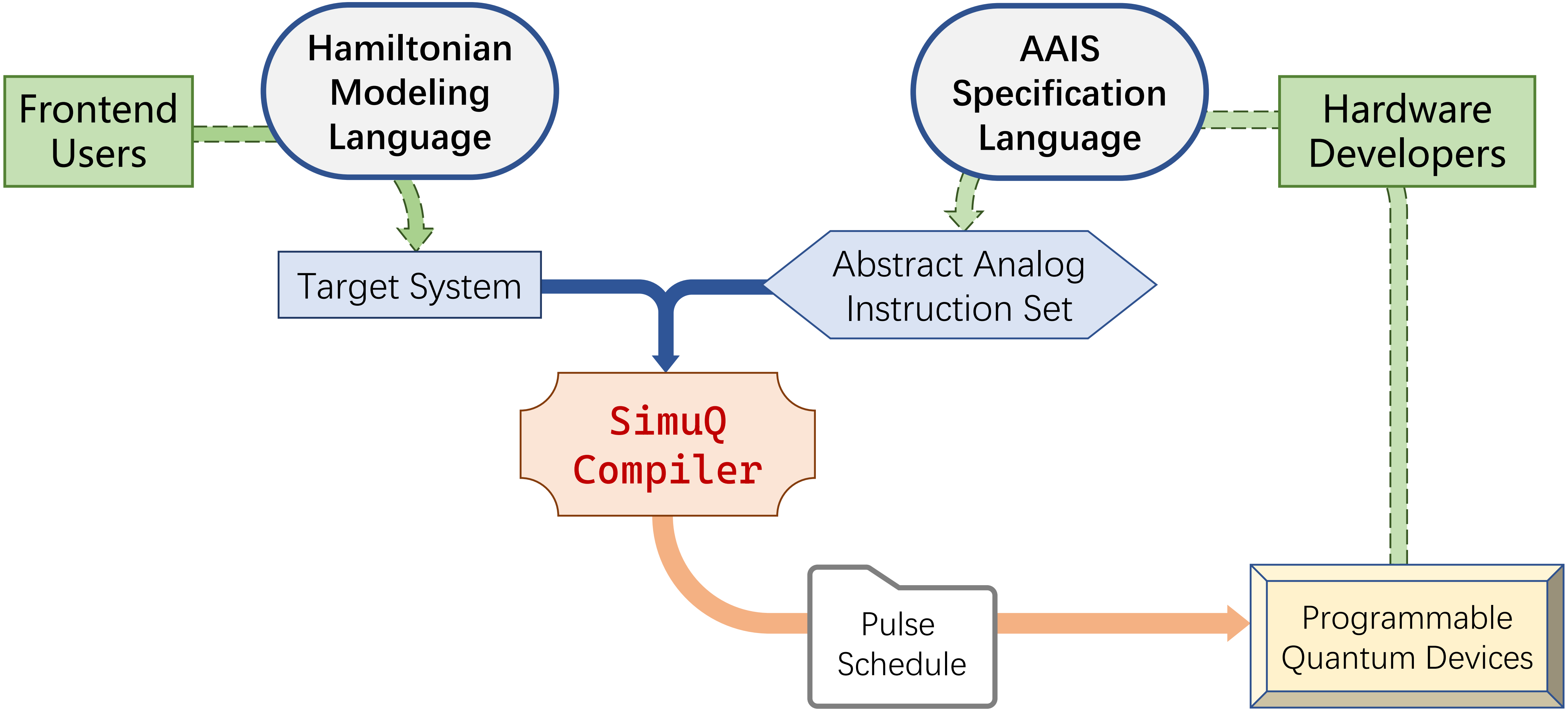
SimuQ Framework



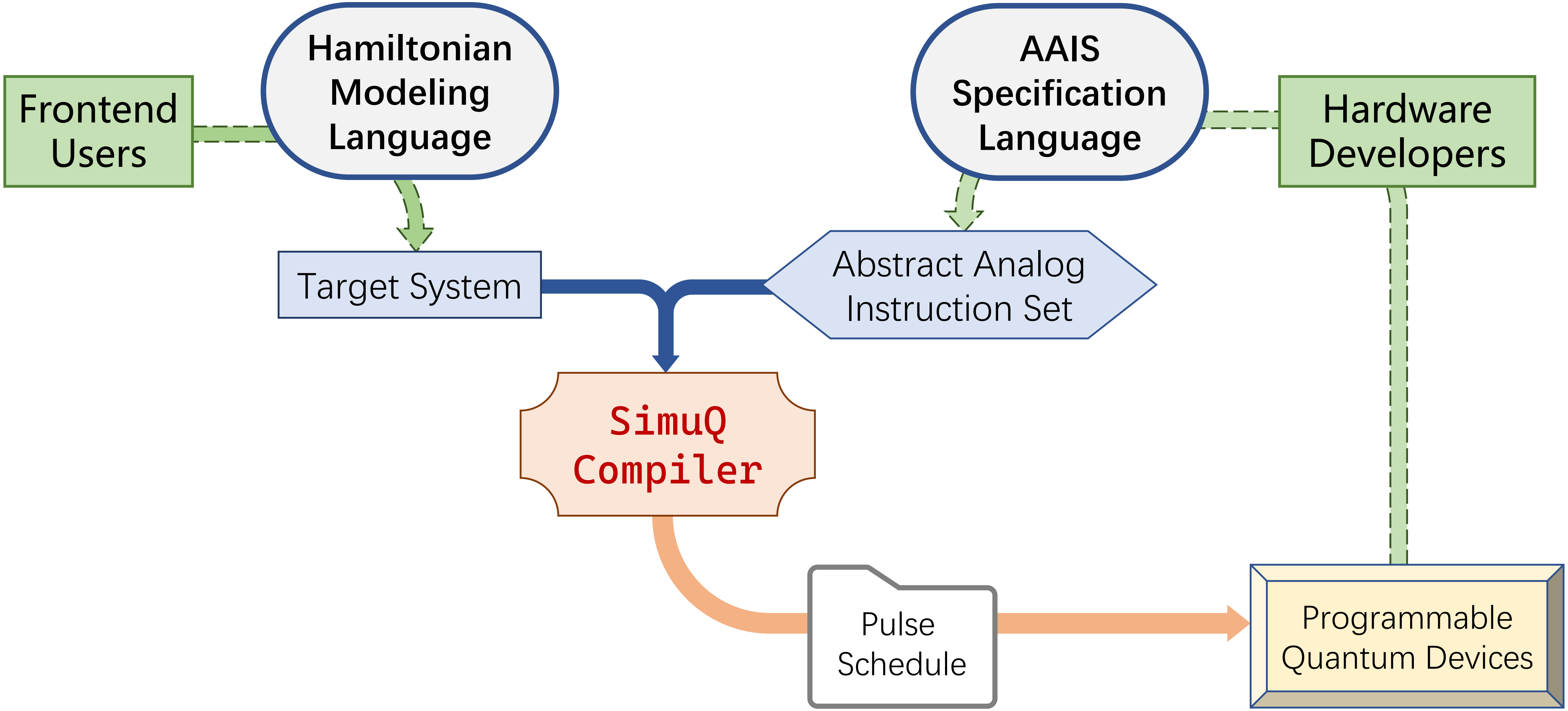
SimuQ Framework



SimuQ Framework



SimuQ Framework



More details of the compiler in Advanced Discussion

Tutorial Outline

- First session
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min) Presenter: Yuxiang Peng
  - Notebook session 1: basics (~40 min)
- Break
- Second session
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min)
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

Installation Guide

As easy as: `pip install simuq`

Providers	Backends	Cloud Pricing
QuTiP [qutip]	Ideal simulators	0
Amazon Braket [braket]	Ideal (& noisy) simulators QuEra's Rydberg arrays (Aquila) IonQ's ion traps (Hamorny) IonQ's ion traps (Aria-1/2)	0 $(0.3+0.01*s) per job $(0.3+0.01*s) per job $(0.3+0.03*s) per job
IonQ Quantum Cloud [ionq]	IonQ's ion traps (Hamorny, Aria-1/2) Ideal & noisy simulators	Consult IonQ 0
IBM Q-Experience [ibm]	IBM's transmons Ideal & noisy simulators	0/$1.6 per second 0

Claim your AWS credits for SimuQ trials:

Send requests to aws-qce23-credits@amazon.com **with name & affiliation!**

Tutorial Outline

Find notebooks at QR or in SimuQ repo
`notebooks/tutorials/`

- First session
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min) Presenter: Pengyu Liu
- Break
- Second session
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min)
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

Tutorial Outline

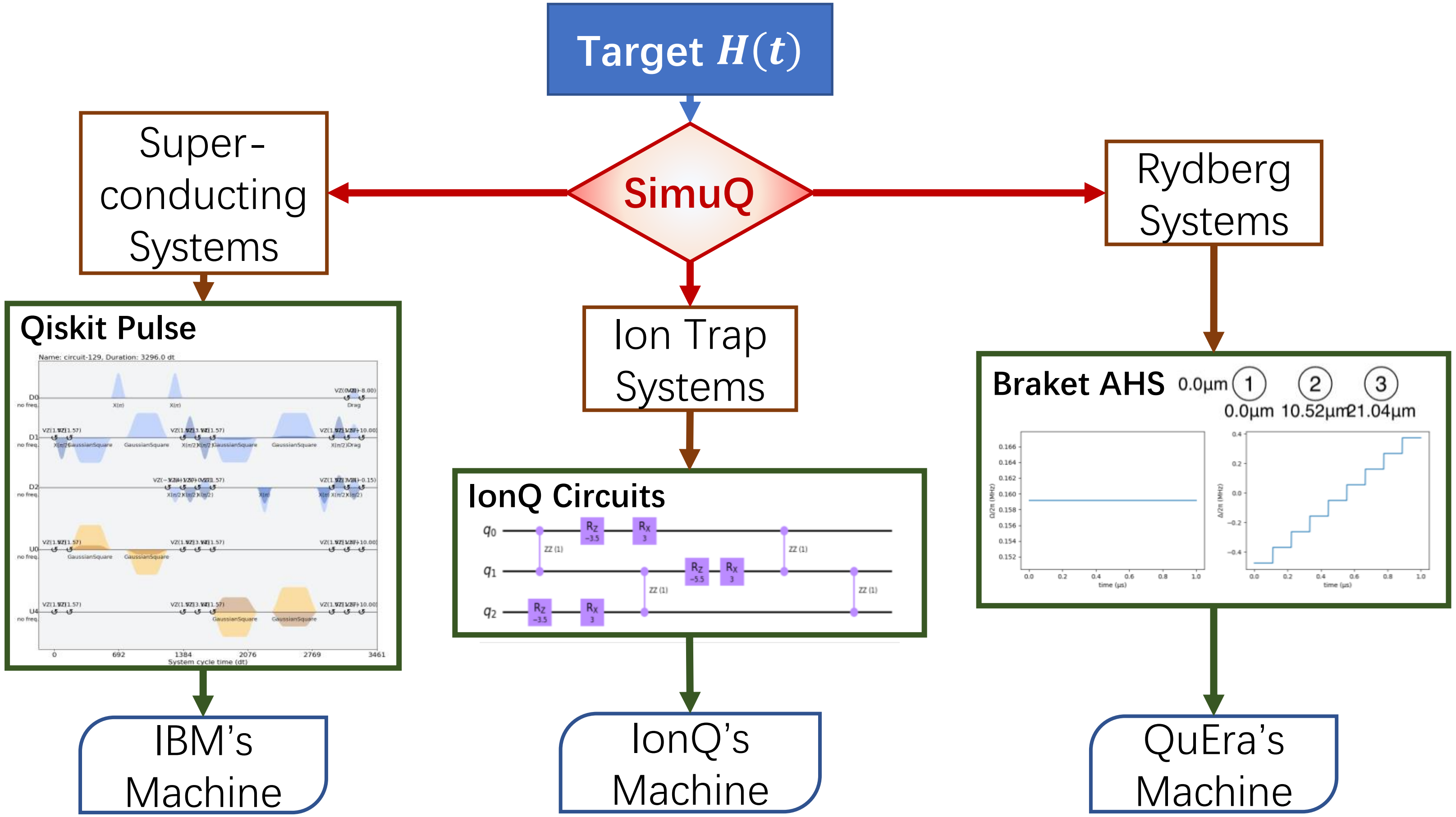
- First session
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- Break
- Second session
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min)
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

SimuQ: SIMUlation language for Quantum

TARGET
QUANTUM
SYSTEM

Analog Compilation

EXECUTABLE
PULSES



Claim your AWS credits for SimuQ trials:

Send requests to aws-qce23-credits@amazon.com with name & affiliation!

Tutorial Outline

Find notebooks at QR or in SimuQ repo

`notebooks/tutorials/`

- First session
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- Break
- Second session
  - Notebook session 2: applications (~50 min) Presenter 1: Jiaqi Leng
  - Advanced discussion (~30 min)
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

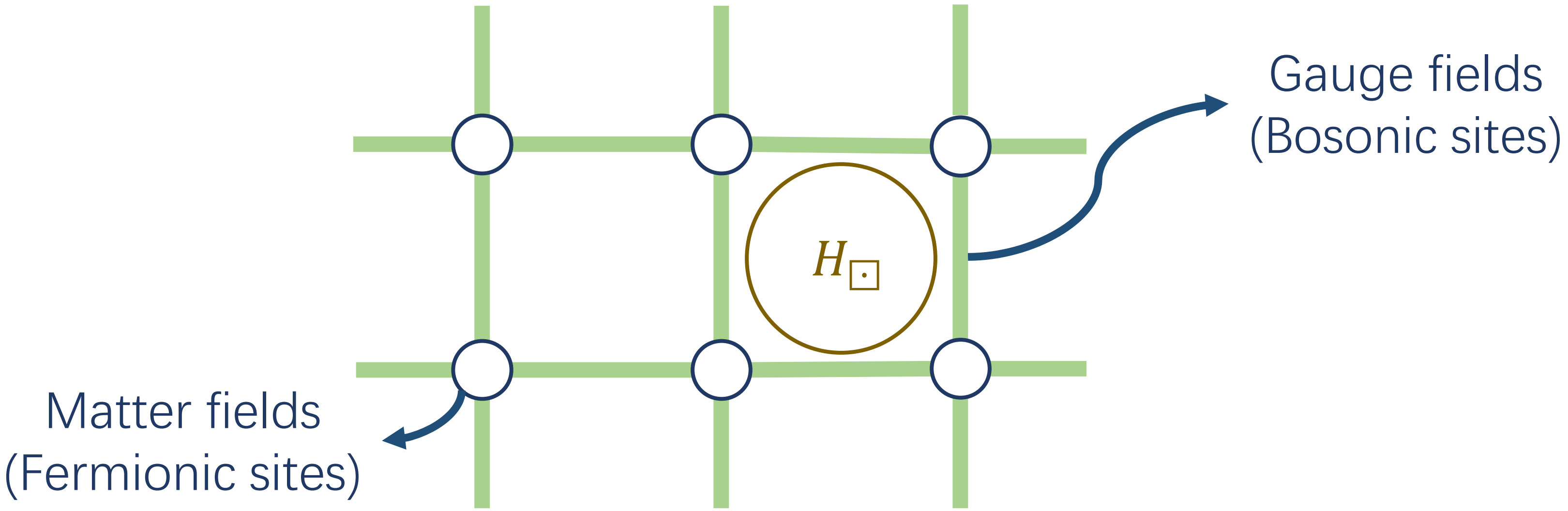
Tutorial Outline

Find notebooks at QR or in SimuQ repo
`notebooks/tutorials/`

- First session
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- Break
- **Second session**
  - **Notebook session 2: applications (~50 min)** Presenter 2: Yuxiang Peng
  - Advanced discussion (~30 min)
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

Quantum Simulation in High-Energy Physics

Lattice gauge theory (LGT)

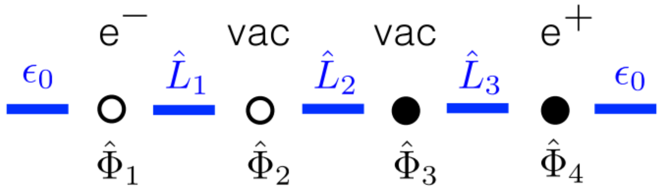


Simulate LGT with quantum simulation?

SimuQ lightens your efforts!

Reproduce demonstration of HEP simulation

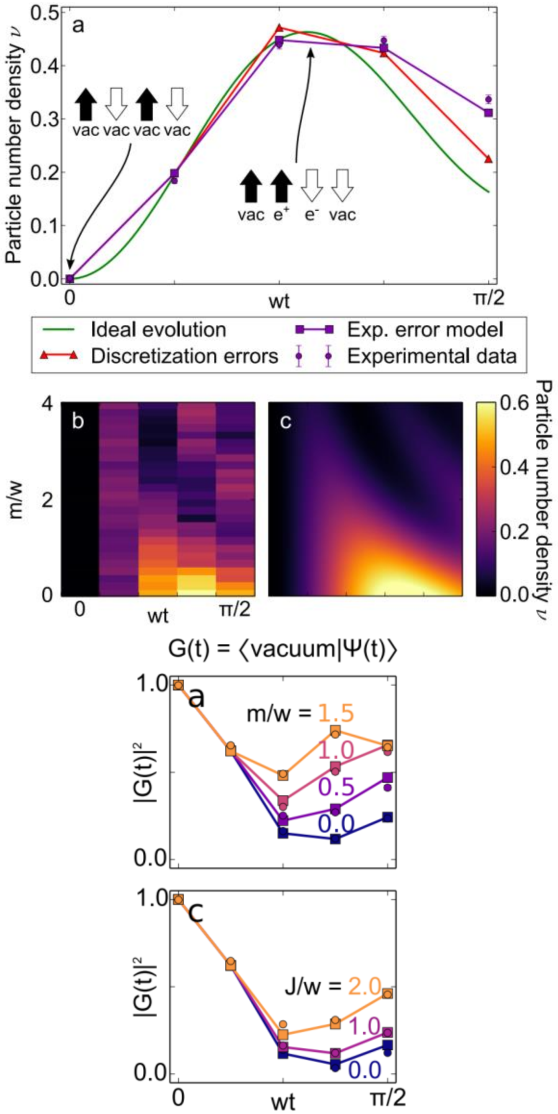
- Dynamics of quantum fluctuation [2]



$$\hat{H}_{\text{lat}} = -iw \sum_{n=1}^{N-1} \left[\hat{\Phi}_n^\dagger e^{i\hat{\theta}_n} \hat{\Phi}_{n+1} - h.c. \right] + J \sum_{n=1}^{N-1} \hat{L}_n^2 + m \sum_{n=1}^{N} (-1)^n \hat{\Phi}_n^\dagger \hat{\Phi}_n.$$

- Encoding of electrons and positrons
  - Odd-even occupation:
  - $|0101\rangle = |\text{vac}, \text{vac}, \text{vac}, \text{vac}\rangle$
  - $|1010\rangle = |\text{e}^-,\ \text{e}^+,\ \text{e}^-,\ \text{e}^+\rangle$

- Easy experiment reproduction in SimuQ!



Source of figures: [2]

[2]: Martinez, E., Muschik, C., Schindler, P. et al. Real-time dynamics of lattice gauge theories with a few-qubit quantum computer. Nature 534, 516–519 (2016).

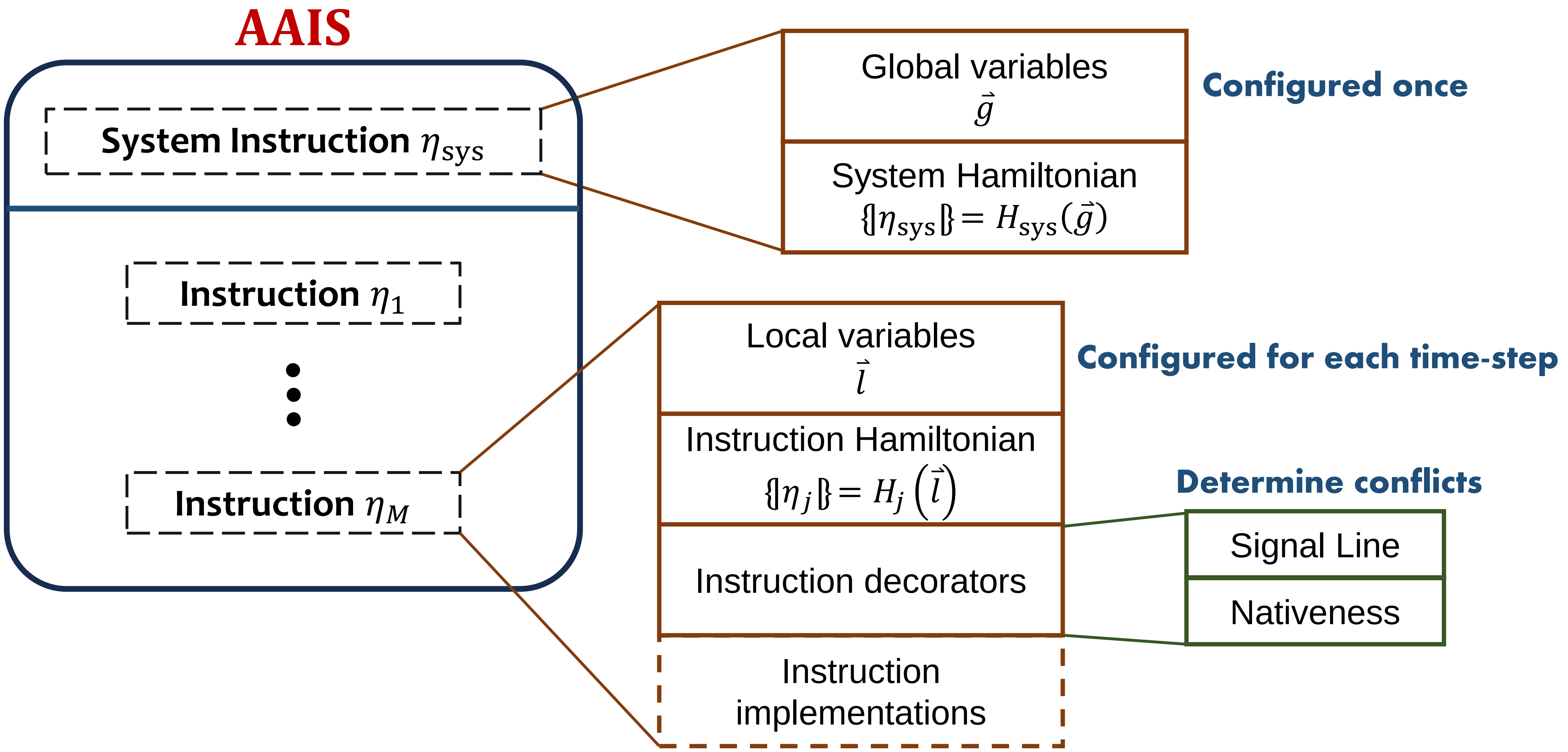
Tutorial Outline

- First session
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- Break
- Second session
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min) Presenter: Yuxiang Peng
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

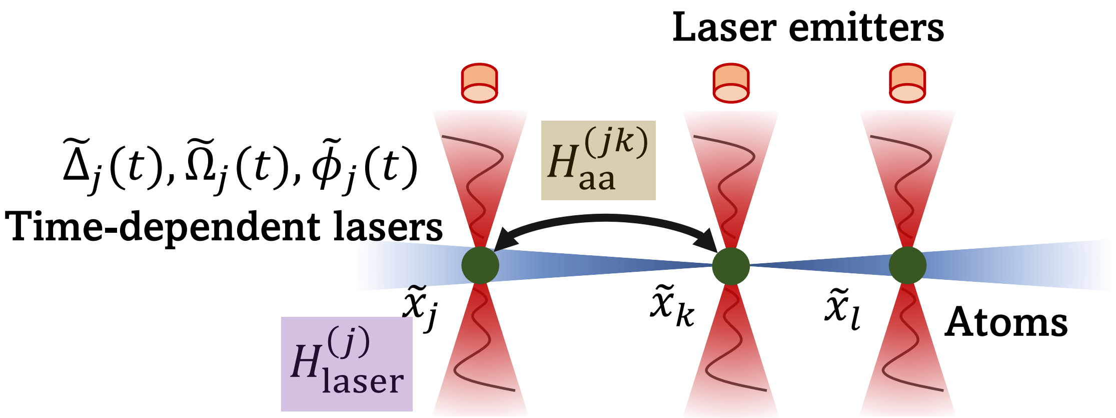
Tutorial Outline

- First session
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- Break
- Second session
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min) Presenter: Yuxiang Peng
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

AAIS Design Details



AAIS Specification Language



AAIS

$$\eta_{system}(\vec{x}) \Rightarrow \sum_{j<k} H_{aa}^{(jk)}$$

$$\eta_{laser}^{(j)}(\Delta, \Omega, \phi) \Rightarrow \Delta \hat{n}_j + \frac{\Omega}{2} H_{phase}^{(j)}(\phi)$$

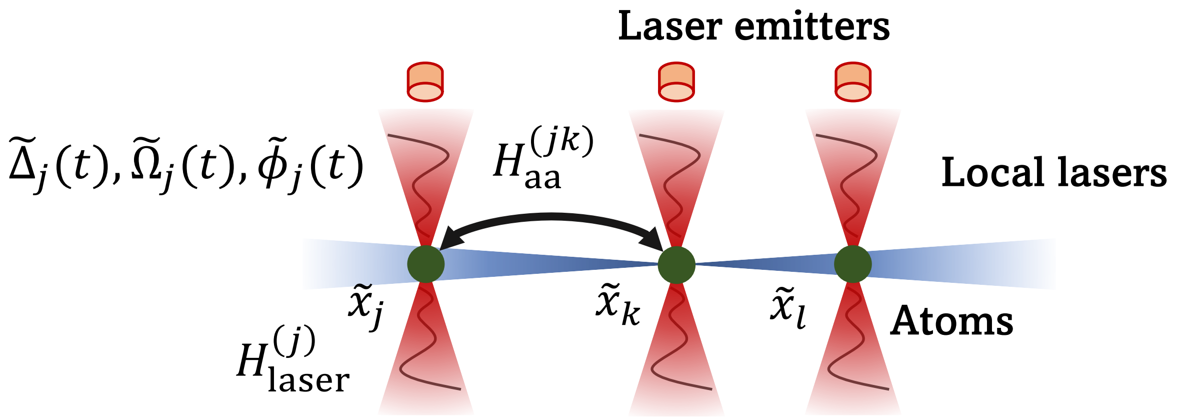
AAIS-SL Program

```
rydberg = QMachine()
q = [Qubit(rydberg) for _ in range(n)]
x = [0] + [rydberg.add_global_variable() for _ in range(n - 1)]
noper = [(q[i].I - q[i].Z) / 2 for i in range(n)]
h = 0
for i in range(n):
    for j in range(i):
        h += (C_6 / (x[i] - x[j]) ** 6) * noper[i] * noper[j]
rydberg.set_sys_ham(h)
for i in range(n):
    L = rydberg.add_signal_line()
    ins = L.add_instruction()
    d = ins.add_local_variable()
    o = ins.add_local_variable()
    p = ins.add_local_variable()
    XY = cos(p) * q[i].X - sin(p) * q[i].Y
    ins.set_ham(-d * noper[i] + o / 2 * XY)
```

QuEra Devices

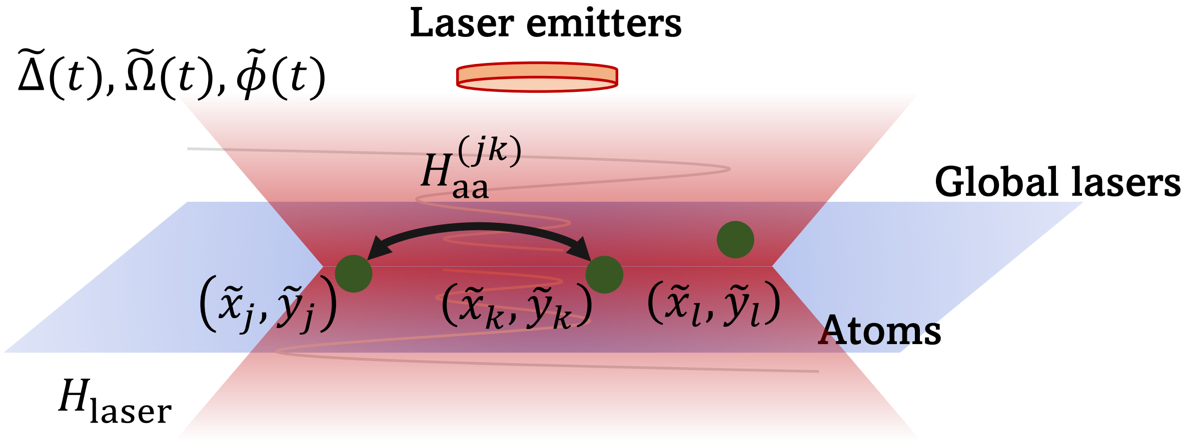
Ideal Rydberg machines

Configurable Parameters:
1. 1D Atom position $\{\tilde{x}_j\}_{j=1}^{N}$.
2. Local laser configurations $\{(\tilde{\Delta}_j(t), \tilde{\Omega}_j(t), \tilde{\phi}_j(t))\}_{j=1}^{N}$



QuEra machines

Configurable Parameters:
1. 2D atom position $\{(\tilde{x}_j, \tilde{y}_j)\}_{j=1}^{N}$
2. Global laser configurations $\tilde{\Delta}(t), \tilde{\Omega}(t), \tilde{\phi}(t)$



Rydberg AAIS for QuEra

Capable of simulating Ising-type systems

AAIS

$\eta_{\text{system}}(\vec{x}) \Rightarrow \sum_{j<k} H_{\text{aa}}^{(jk)}$

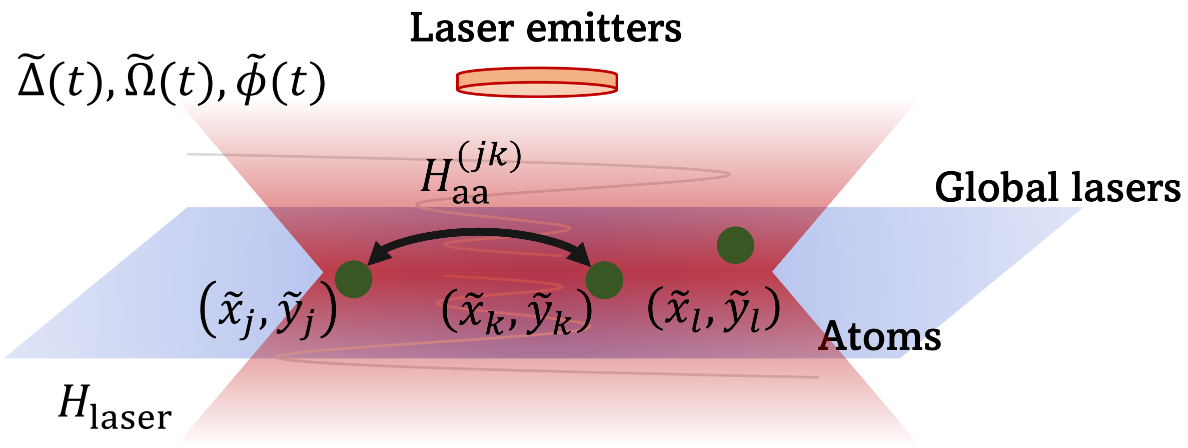
$\eta_{\text{laser}}(\Delta,\Omega,\phi) \Rightarrow \Delta \sum_j \hat{n}_j + \frac{\Omega}{2}\sum_j H_{\text{phase}}^{(j)}(\phi)$

- 1D-variant
  $\vec{x} = \{x_j\}_{j=1}^N, H_{\text{aa}}^{(jk)} = \frac{C_6}{|x_j-x_k|^6}\hat{n}_j\hat{n}_k$
- 2D-variant
  $\vec{x} = \{(x_j,y_j)\}_{j=1}^N, H_{\text{aa}}^{(jk)} = \frac{C_6}{\left((x_j-x_k)^2+(y_j-y_k)^2\right)^3}\hat{n}_j\hat{n}_k$

QuEra machines

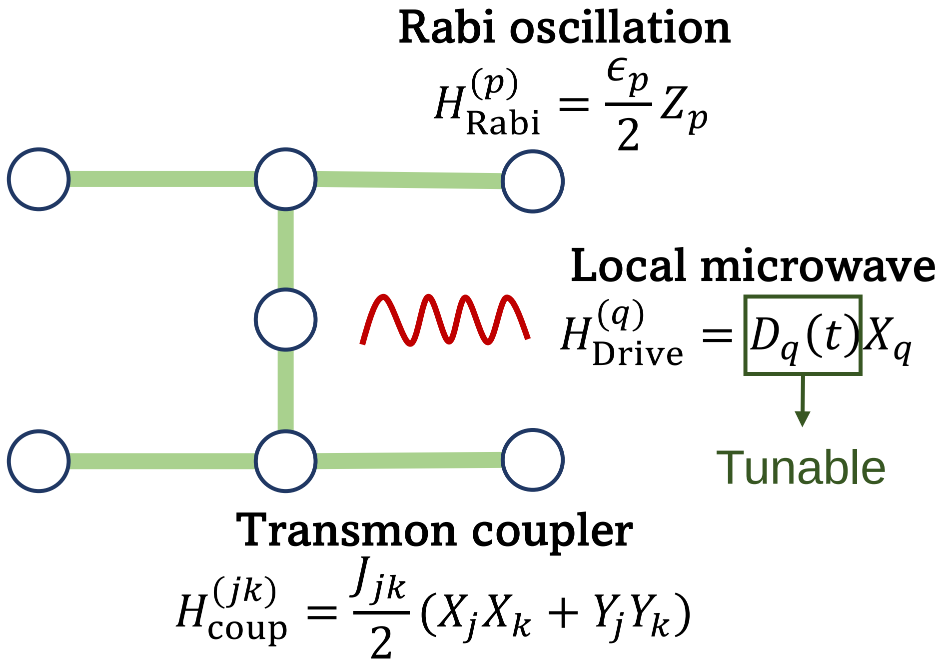
Configurable Parameters:

1. 2D atom position $\{(\tilde{x}_j,\tilde{y}_j)\}_{j=1}^N$
2. Global laser configurations $\tilde{\Delta}(t),\tilde{\Omega}(t),\tilde{\phi}(t)$

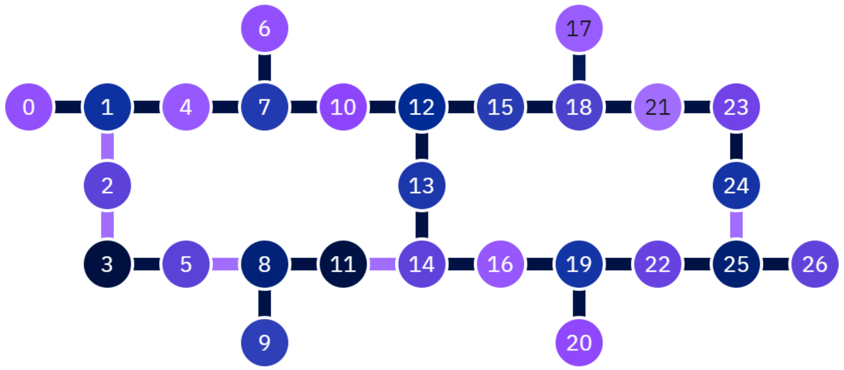


IBM Devices

- Fixed-frequency transmon systems
- Rotating frame:
  - $H_{\mathrm{coup}}^{(jk)} \Rightarrow \omega_{ZX} Z_j X_k + \omega_{ZZ} Z_j Z_k + \cdots$
- Native Ham. is not so useful!
  - Few systems can be mapped natively
  - It requires frequent calibration



Typical topology



Heisenberg AAIS

- An AAIS for Heisenberg-type systems.

- Instructions:
  - 1-q: $\{\!|\eta_{j,P}|\!\}(a) = aP_j$
  - 2-q: $\{\!|\eta_{jk,PP}|\!\}(a) = aP_jP_k$
  - $P \in \{X, Y, Z\}$
  - Implementation: pulse-efficient gates
    - Pulse length ∝ evolution time
    - Shorter duration compared to CNOT-based implementation.

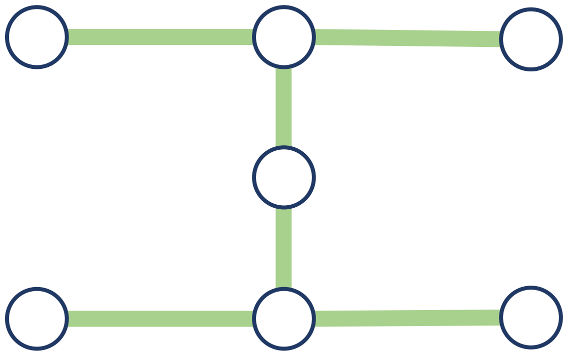
- Variants: 2-Pauli AAIS

1-q instructions

$\{\!|\eta_{p,X}|\!\}(a) = aX_p$

$\{\!|\eta_{p,Y}|\!\}(a) = aY_p$

$\{\!|\eta_{p,Z}|\!\}(a) = aZ_p$



2-q instructions

$\{\!|\eta_{jk,XX}|\!\}(a) = aX_jX_k$

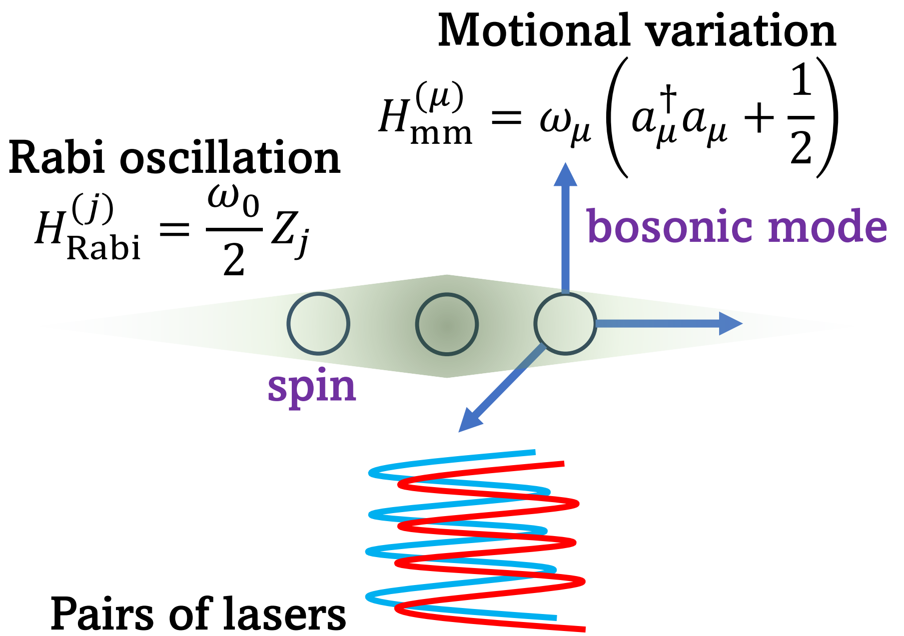
$\{\!|\eta_{jk,YY}|\!\}(a) = aY_jY_k$

$\{\!|\eta_{jk,ZZ}|\!\}(a) = aZ_jZ_k$

On the edges

IonQ Devices

- Trapped ion arrays
- Native Hamiltonian is complicated
- Employ Heisenberg AAIS
  - Full connectivity
  - Implementation:
    - Arbitrary angle Mølmer-Sørensen gates
    - Pulse length ∝ evolution time

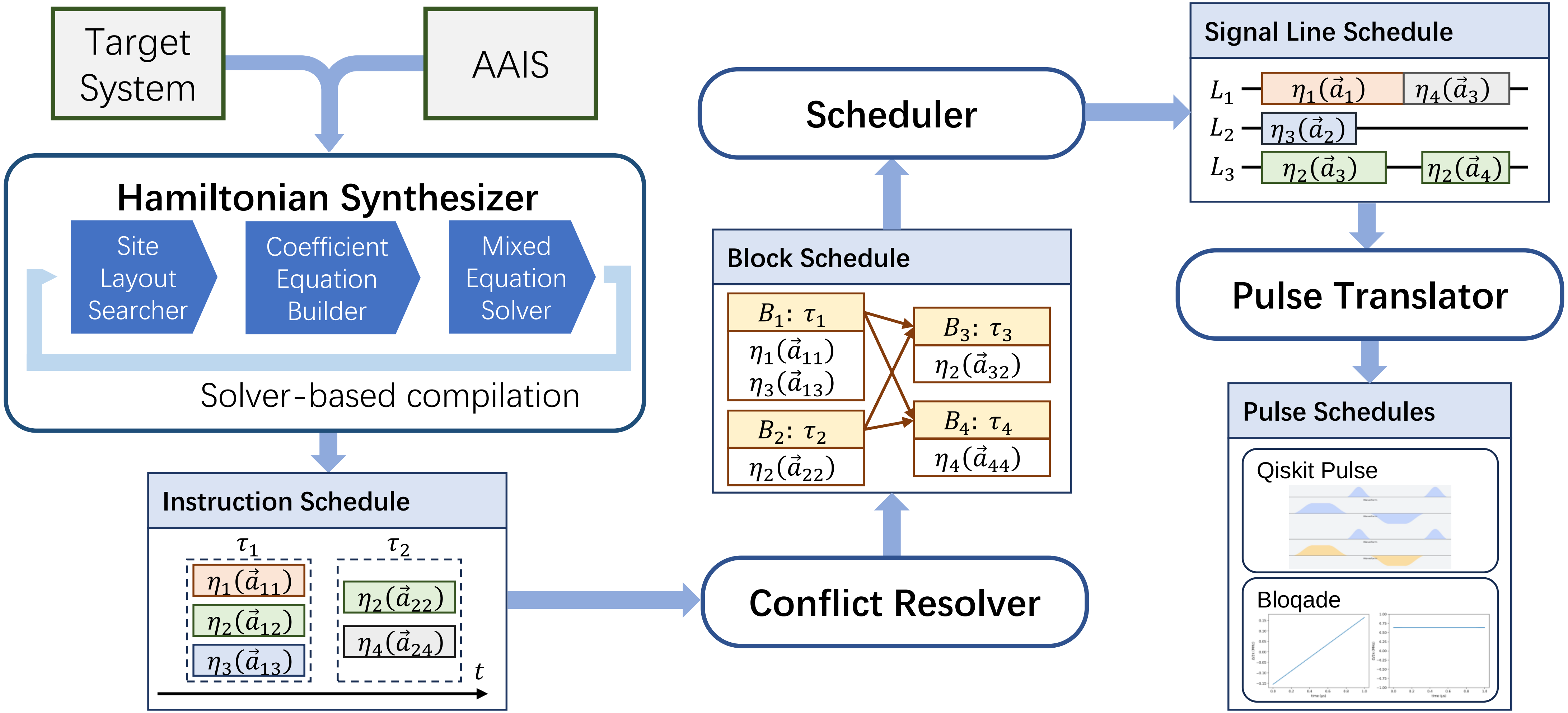


$$H_{\text{int.}}^{(j)}(t) = \Omega_L \left(\sum_P \alpha_P P_j \right) e^{-i\Delta\omega_L t + i\Delta\varphi_L + i\sum_\mu \eta_{j\mu}\left(a_\mu^\dagger + a_\mu\right)} + h.c.$$

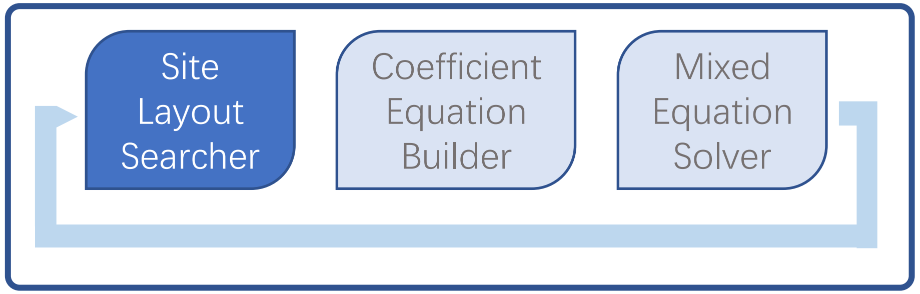
Tutorial Outline

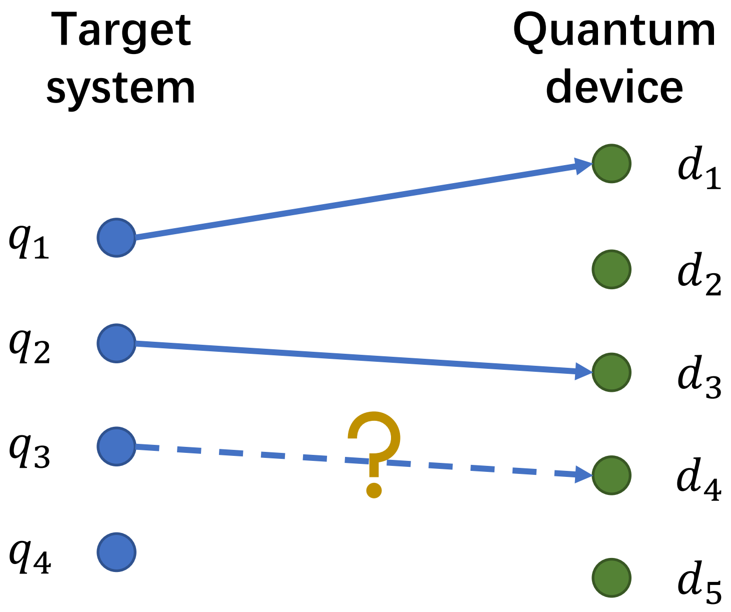
- First session
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- Break
- Second session
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min) Presenter: Yuxiang Peng
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

SimuQ Compiler



Hamiltonian Synthesizer





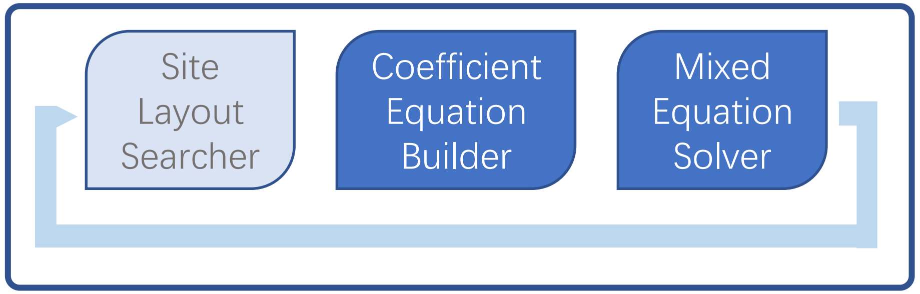
Difference to circuit model:
No SWAP gates for many devices

Brute-force search with heuristics

Rydberg, ion trap: Easy (perfect symmetry)
Superconducting: Hard (specific topology)

May be overloaded by manual assignments

Hamiltonian Synthesizer



Synthesize $\boxed{H_{\text{tar}} = Z_1 X_2}$ from

$\eta_1(\alpha): \alpha Z_1 X_2 + Z_1$ $\eta_2(\beta): \beta Z_1$

$$\begin{cases} Z_1 X_2: & \alpha + 0 = 1 \\ Z_1: & 1 + \beta = 0 \end{cases}$$

(SMT) Solver

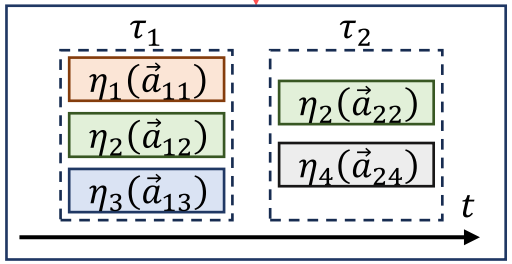
- Lst-sqr-based
- dReal4

Activated Instructions

$\eta_1(1): \quad Z_1 X_2 + Z_1$

$\eta_2(-1): -Z_1$

$\eta_1(1) + \eta_2(-1) = H_{\text{tar}}$



Build equation for each product-Hamiltonian *P*

$$\tau \cdot \sum_{j=1}^{m} \{\!|\eta_j|\!\}[P](\mathbf{v}) \cdot s_j = t \cdot \mathcal{M}(H_{\text{tar}})[P]$$

- Coefficient of P in instruction ι_j
- Valuation of variables
- Indicator in $\{0,1\}$
- Coefficient of P in H_{tar} in layout $\mathcal{M}$

Conflict Resolver

Activated Instructions

$\eta_1(1)$: $Z_1X_2 + Z_1$

$\eta_2(-1)$: $-Z_1$

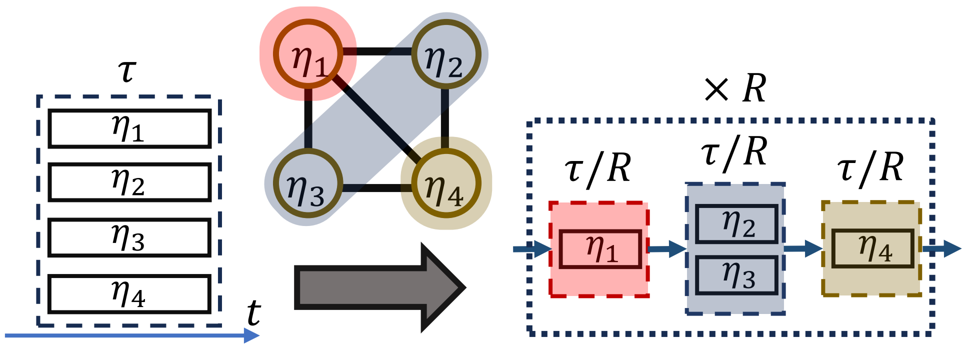
$\eta_1(1) + \eta_2(-1) = H_{\text{tar}}$

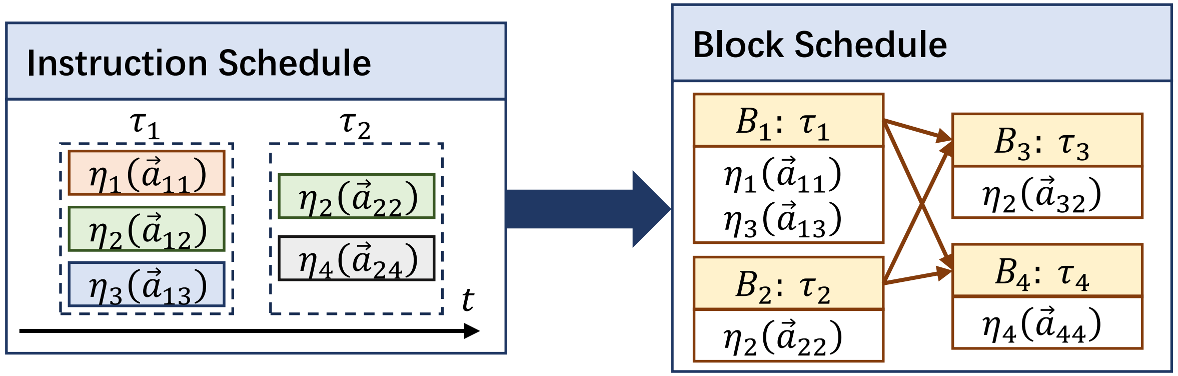
! **Conflicting Instructions:**

$\eta_1(\alpha)$ and $\eta_2(\beta)$
cannot be executed
simultaneously on IBM

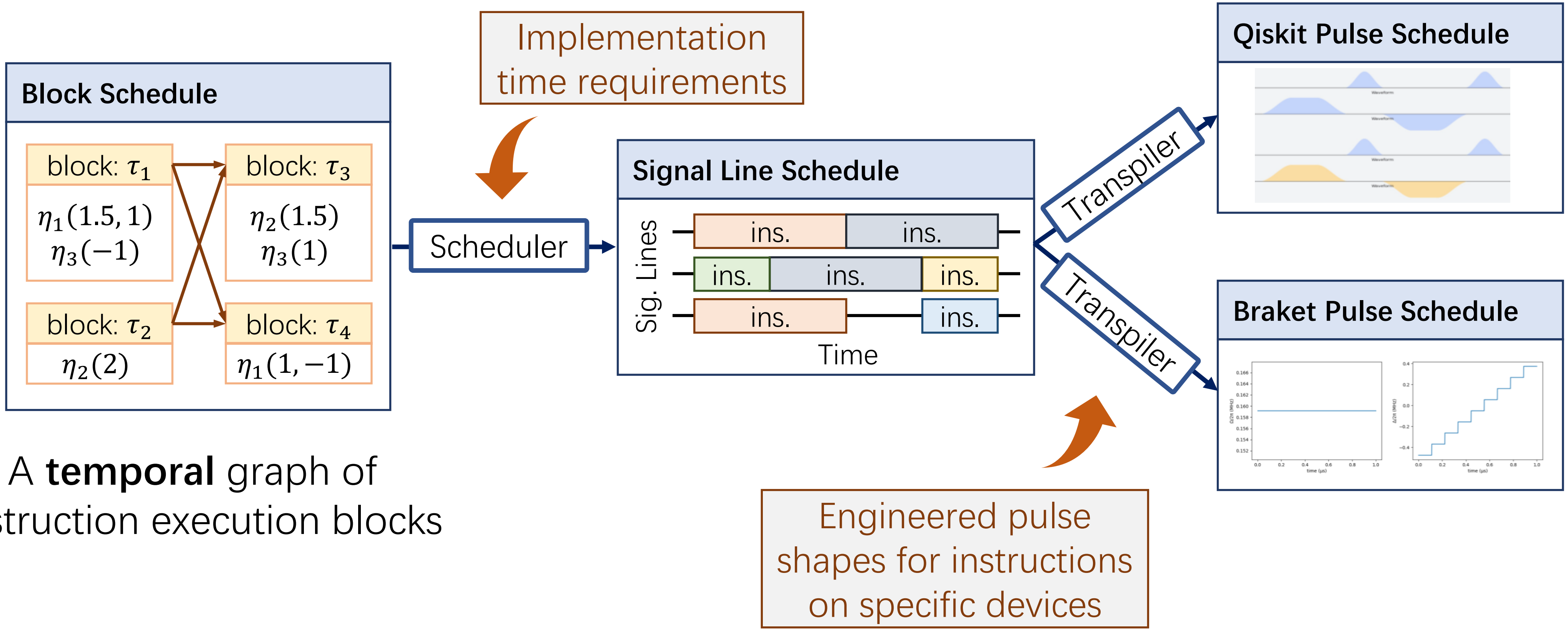
Resolve conflicts by Trotterization

Conflict relation graph





Scheduler and Transpiler



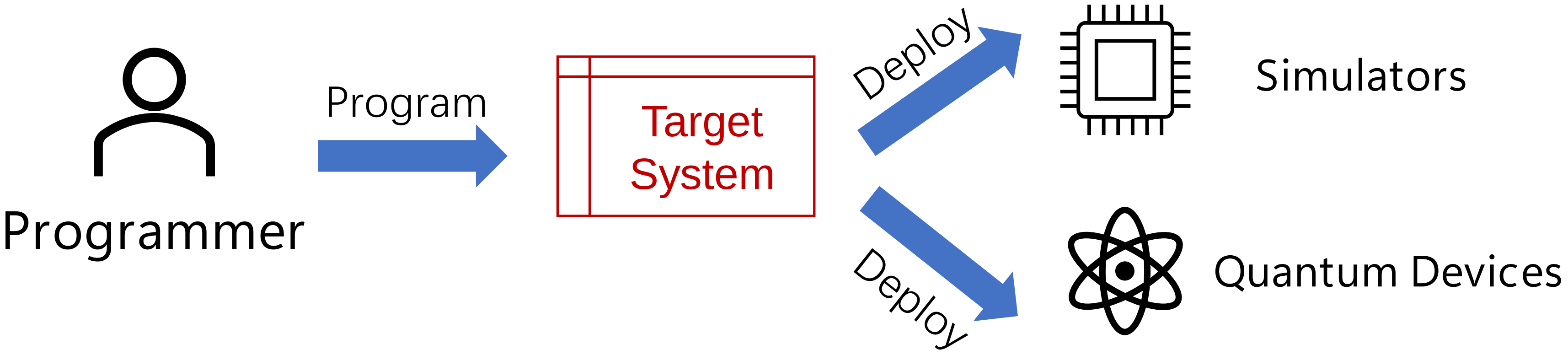
A **temporal** graph of
Instruction execution blocks

Tutorial Outline

- First session
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- Break
- Second session
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min) Presenter: Yuxiang Peng
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

As a user:
What to explore now

Front-end User Perspective



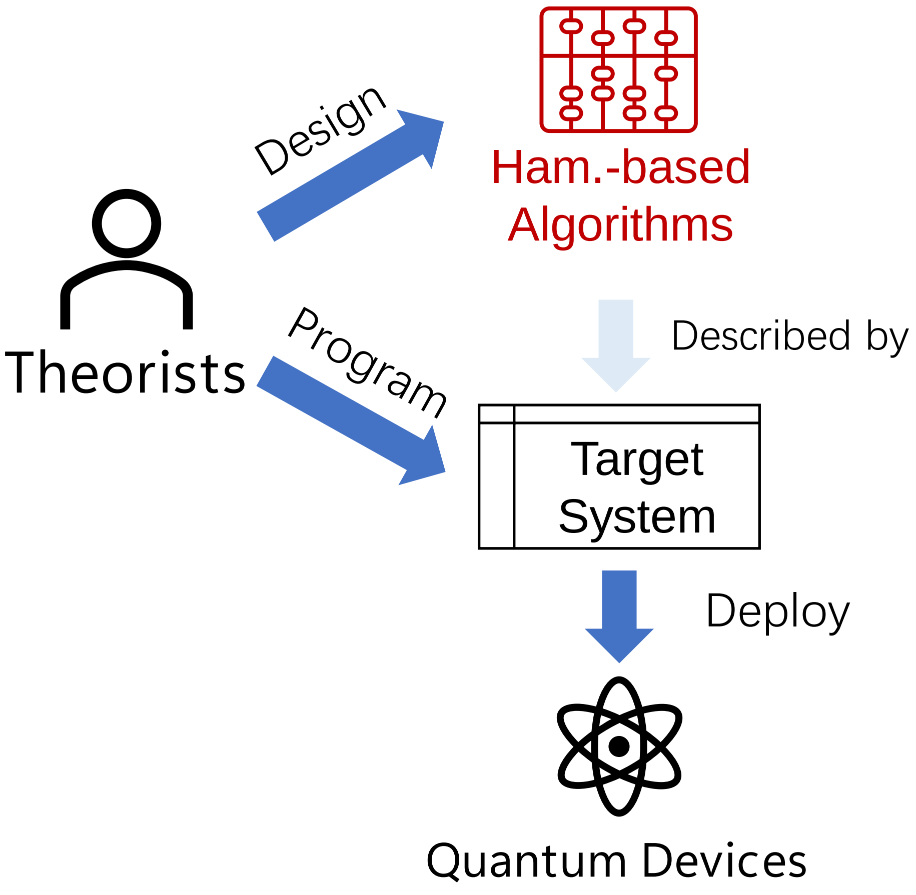
- Quantum mechanics researchers
  - Simulate interesting systems
- Quantum hardware investigators
  - Evaluate devices in applications
  - Cross-platform performance evaluation
- A quantum simulation benchmark

System name	**LoC**	**# of sites**	**Compilation time (s)**			**IBM Duration (ms)**		**IonQ #2q-gate**
			QuEra	IBM	IonQ	SimuQ	Qiskit	
ising_chain	13	6	0.177	0.224	0.155	2.06	8.69	20
		32	39.3	54.6	47.2	3.24	39.2	124
		64	663	257	N.S.	3.15	81.2	
		96	2298	1086	N.S.	3.26	450	
ising_cycle	13	6	0.585	N.S.	0.13			24
		12	3.47	1.49	1.37	2.05	37.8	48
		32	114	483	53.8	3.35	144	128
		64	3454	T.O.	N.S.			
heis_chain	15	32	N.S.	143	138	10.1	119	372
qaoa_cycle	19	12	N.S.	0.503	1.50	0.83	37.6	36
qhd	16	16	N.S.	N.S.	66.3			480
mis_chain	22	12	5.45	19.1	25.2	18.9	94.0	440
		24	53.1	328	278	18.9	162	920
mis_grid	29	16	28.4	N.S.	85.4			960
		25	141	N.S.	489			1600
kitaev	13	18	4.67	15.6	8.74	2.12	21.2	68
schwinger	18	10	N.S.	N.S.	1.09			28
o3nlσm	19	30	N.S.	N.S.	77.7			588

Algorithm Designer Perspective

- Domain experts
  1. Investigate domain problems
     - I.e., non-convex optimization [3].
  2. Design Hamiltonian-based algorithms
  3. Deploy on devices for testing

- Potential new directions
  - Hybrid digital-analog algorithms
  - Error reduction algorithms
  - Characterization of analog simulators

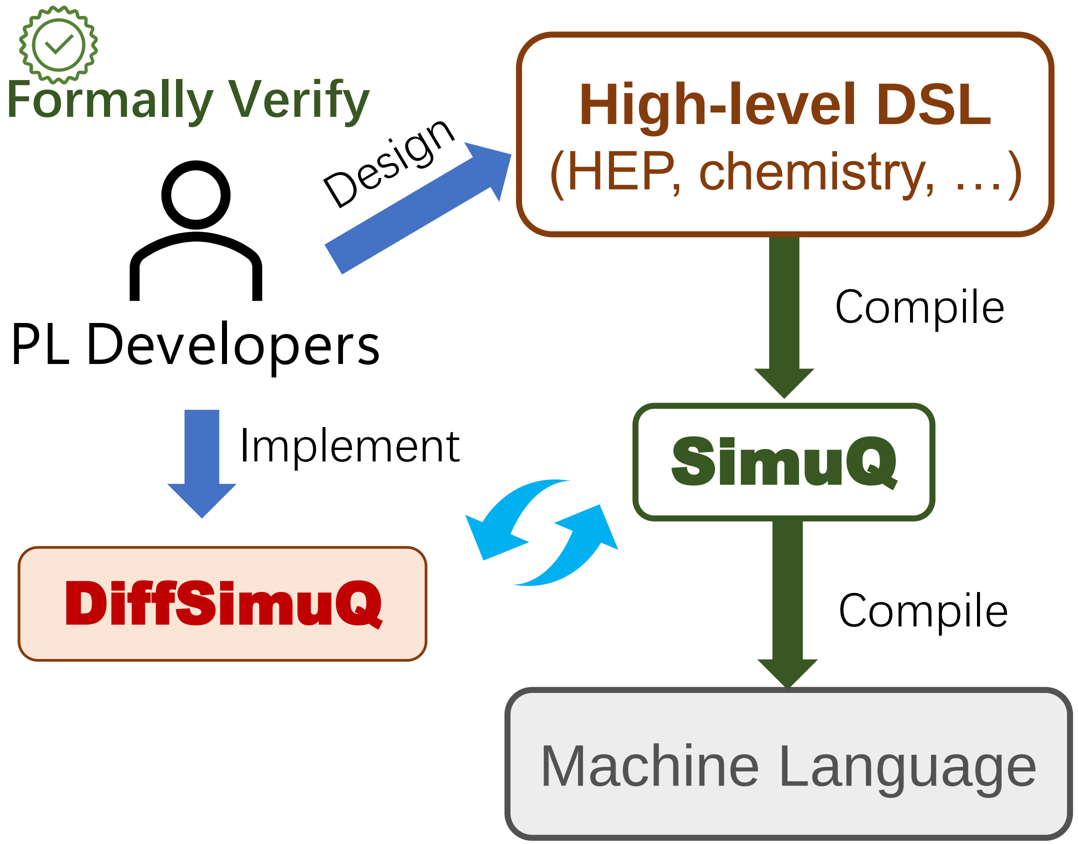


[3]: Leng et al., Quantum Hamiltonian Descent, arXiv:2303.01471

As a developer:
What to investigate in the future

PL Developer Perspective

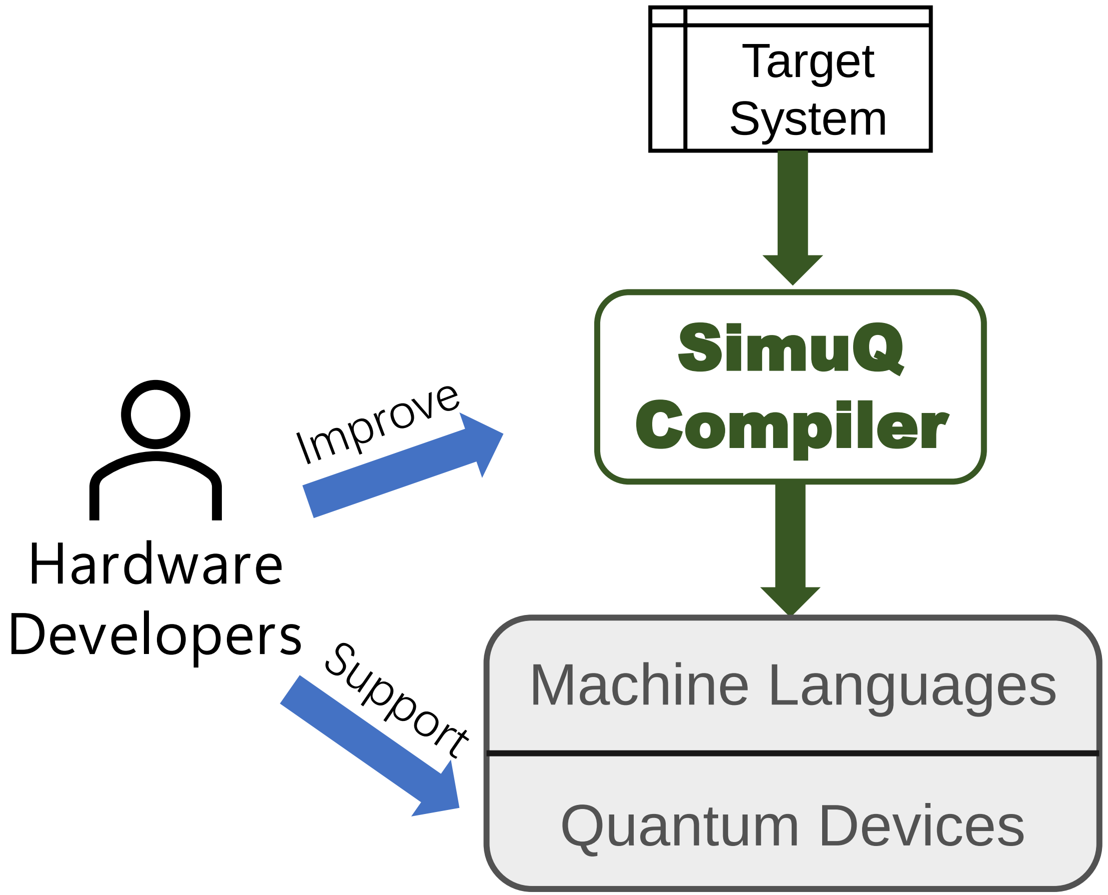
- Domain PL experts
  - Design higher-level DSL
  - Interfaces for common domain users
  - Compilation to SimuQ
- Pulse/VQE PL developers
  - A meta language with variables
  - Auto-differentiation [3]
- Formally verify the suite of SimuQ



[3]: Leng et al., Differentiable Analog Quantum Computing for Optimization and Control, NeurIPS, 2022

Hardware Developer Perspective

- Quantum architecture researchers
  - Hamiltonian-based layout synthesis
  - Pulse-aware compilations
  - Better Trotterization strategies
- Quantum technology developers
  - Integrate error mitigation techniques
  - Robustness analysis of AHS
- Hardware providers
  - Design and implement AAIS for existing devices
  - Develop novel devices realizing powerful AAIS



Tutorial Outline

- **First session**
  - Motivation & introduction (~40 min)
  - Installation guide (~10 min)
  - Notebook session 1: basics (~40 min)
- **Break**
- **Second session**
  - Notebook session 2: applications (~50 min)
  - Advanced discussion (~30 min)
    - AAIS design for multiple devices
    - SimuQ compilation
    - Potential usages of SimuQ
  - Open discussion (~10 min)

Recap of SimuQ

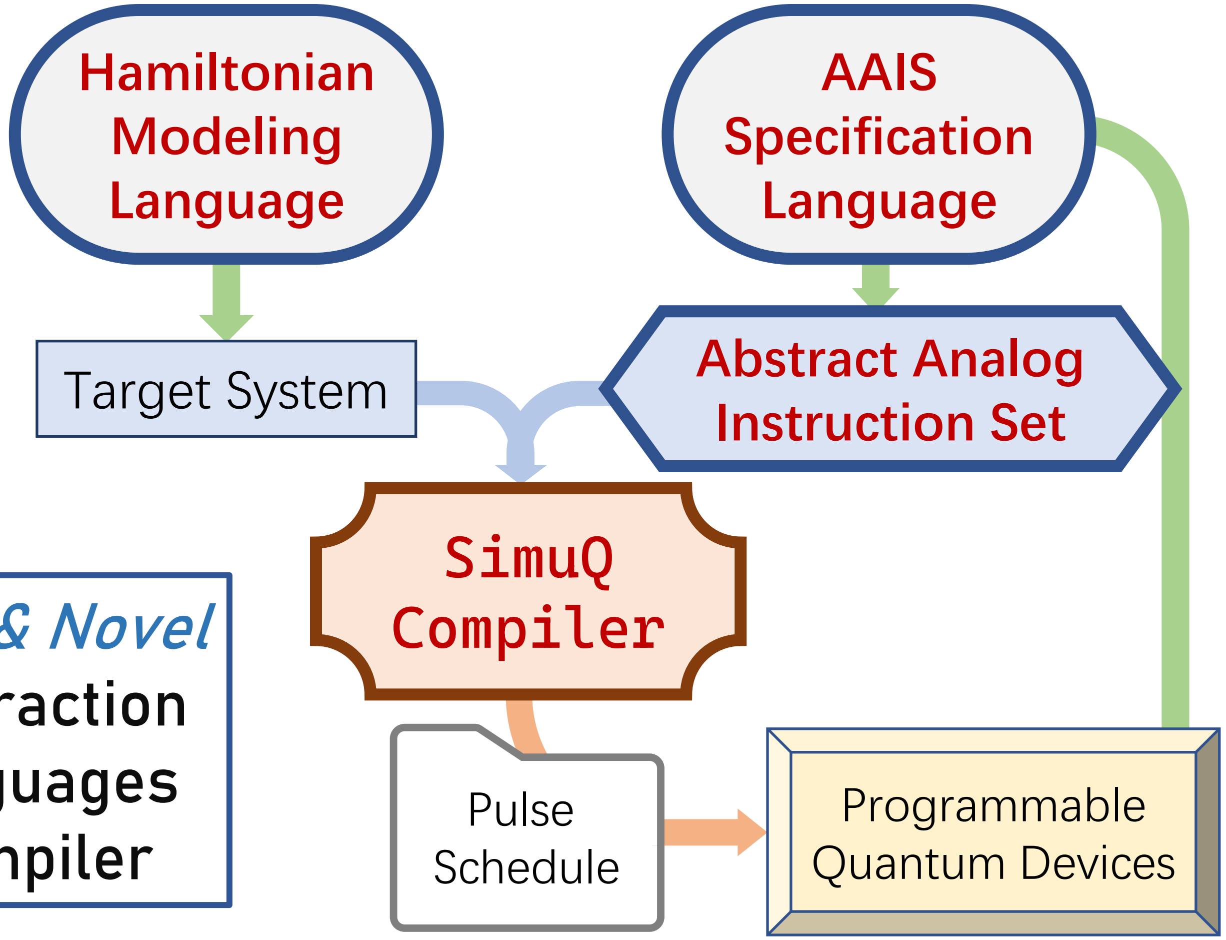
End-to-end Framework

TARGET QUANTUM SYSTEM

Analog ↓ Compilation

EXECUTABLE PULSES

New & Novel **Abstraction Languages Compiler**



Enhance your capability of harnessing the power of quantum devices

GitHub repo:
https://github.com/PicksPeng/SimuQ

arXiv: 2303.02775

Project website:
https://pickspeng.github.io/SimuQ/

Try SimuQ!

Contributions welcome!

Claim your AWS credits for trials!

Send requests to aws-qce23-credits@amazon.com
with name & affiliation!