# Beyond-CMOS computing:
## non-von Neumann architectures for ML

MAS General Exam

Camron Blackburn

4 | 12 | 24

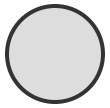
# Exam Content

Dissertation (broad) Objective:

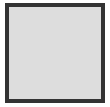
*to propose a new computing framework for ultra-low power computing with devices which have the capability of reaching fundamental physical limits for information processing and an architecture that can optimally map software applications to efficient hardware*

Required understanding:

Non-von Neumann Architectures for Energy Efficient Computing

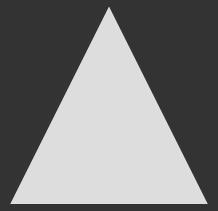


Superconducting Device Physics and Electronics Design



Fundamental Physical Limits to Information Processing

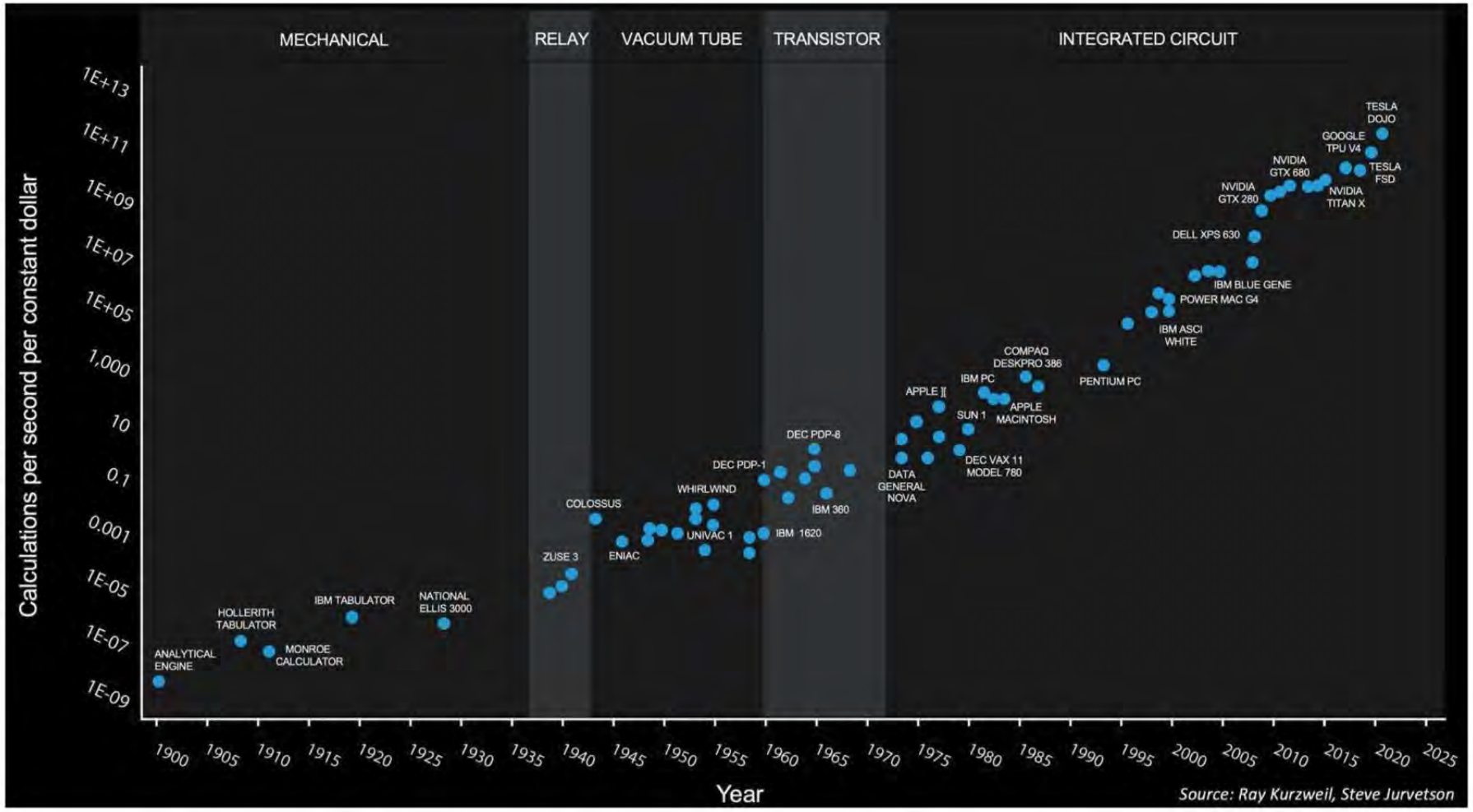
# Non-von Neumann Architectures for Energy Efficient Computing



# CONTENTS

- Current state of High-Performance Computing
- Computing Architecture Categorizations
- Metrics for evaluating computing systems
- Coarse-grain Reconfigurable Architectures (CGRAs)
- Globally Asynchronous, Locally Synchronous (GALS) pipelining methods
- DNN Accelerator Design
- Tools for modeling DNN Accelerator architectures
  - AQFP implementation

# 1. The State of High-Performance Computing

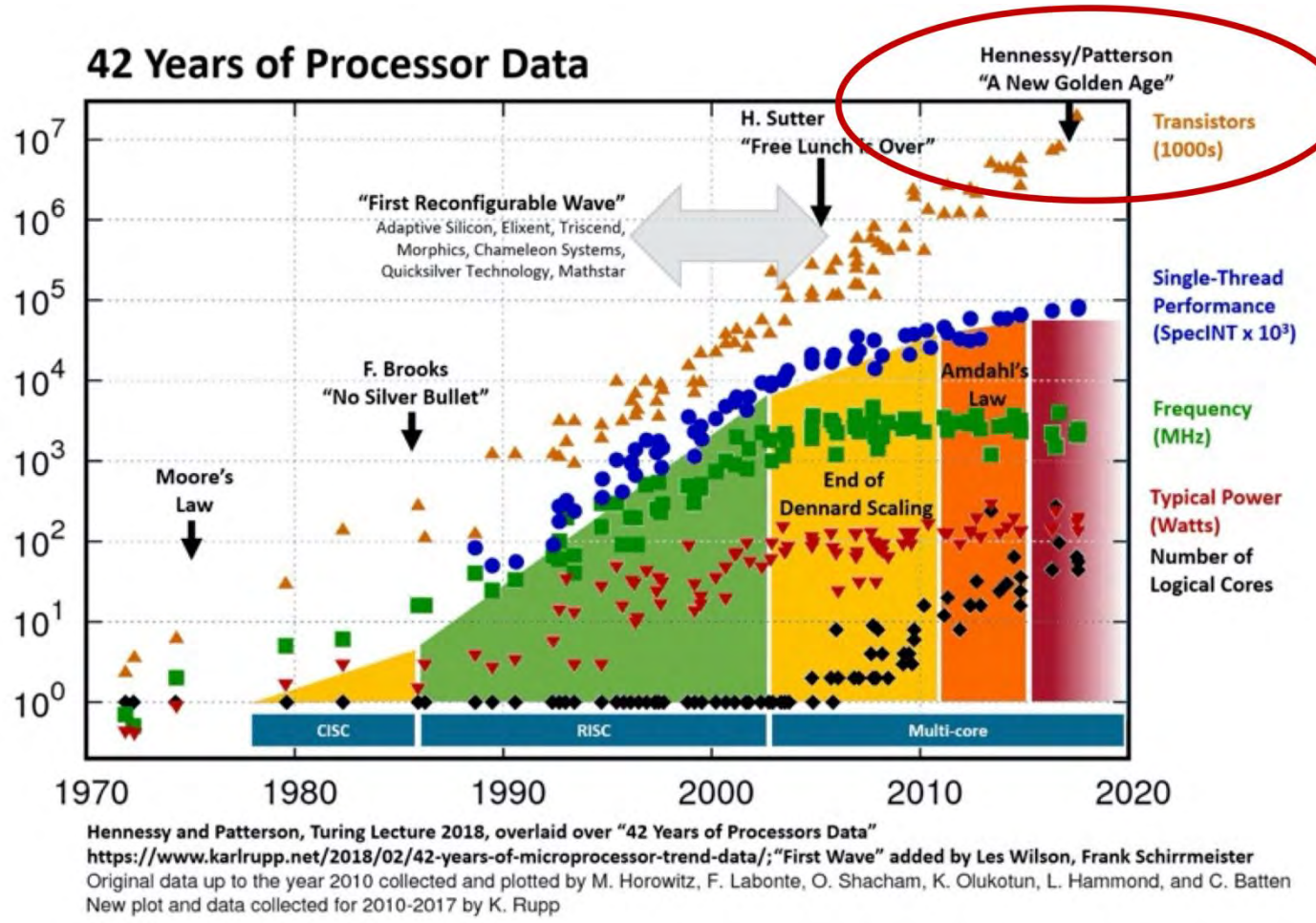



C. Mead, “Neuromorphic Engineering: In Memory of Misha Mahowald” *Neural Computation* (2023)

# 1. Scaling laws slowing down

Scaling laws that enabled the boom in IC development over the last 50 years are slowing down

- Moore's Law
- Dennard Scaling
- The power wall




Hennessy & Patterson predict a new golden age is approaching: one which relies on hardware/software co-design and open-source cores and architectures [2]

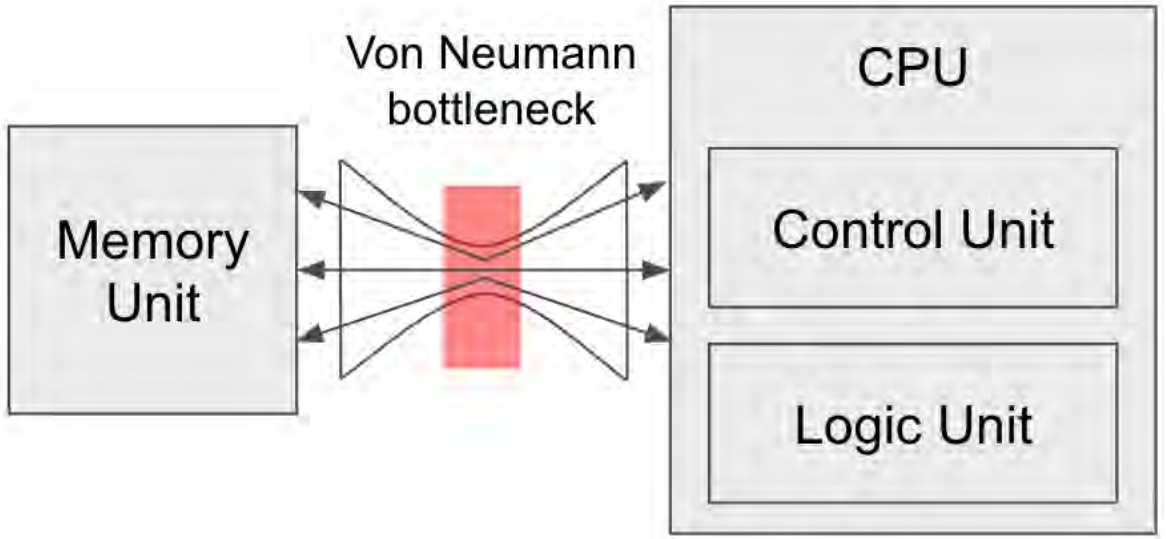
[fig] Vivienne Sze, 6.6010 Analysis and Design of Digital Integrated Circuits, 2022

[2] J. Hennessy and D. Patterson, "A new golden age for computer architecture," *ACM/IEEE 45th ISCA,* (2018)

# Von Neumann Bottleneck

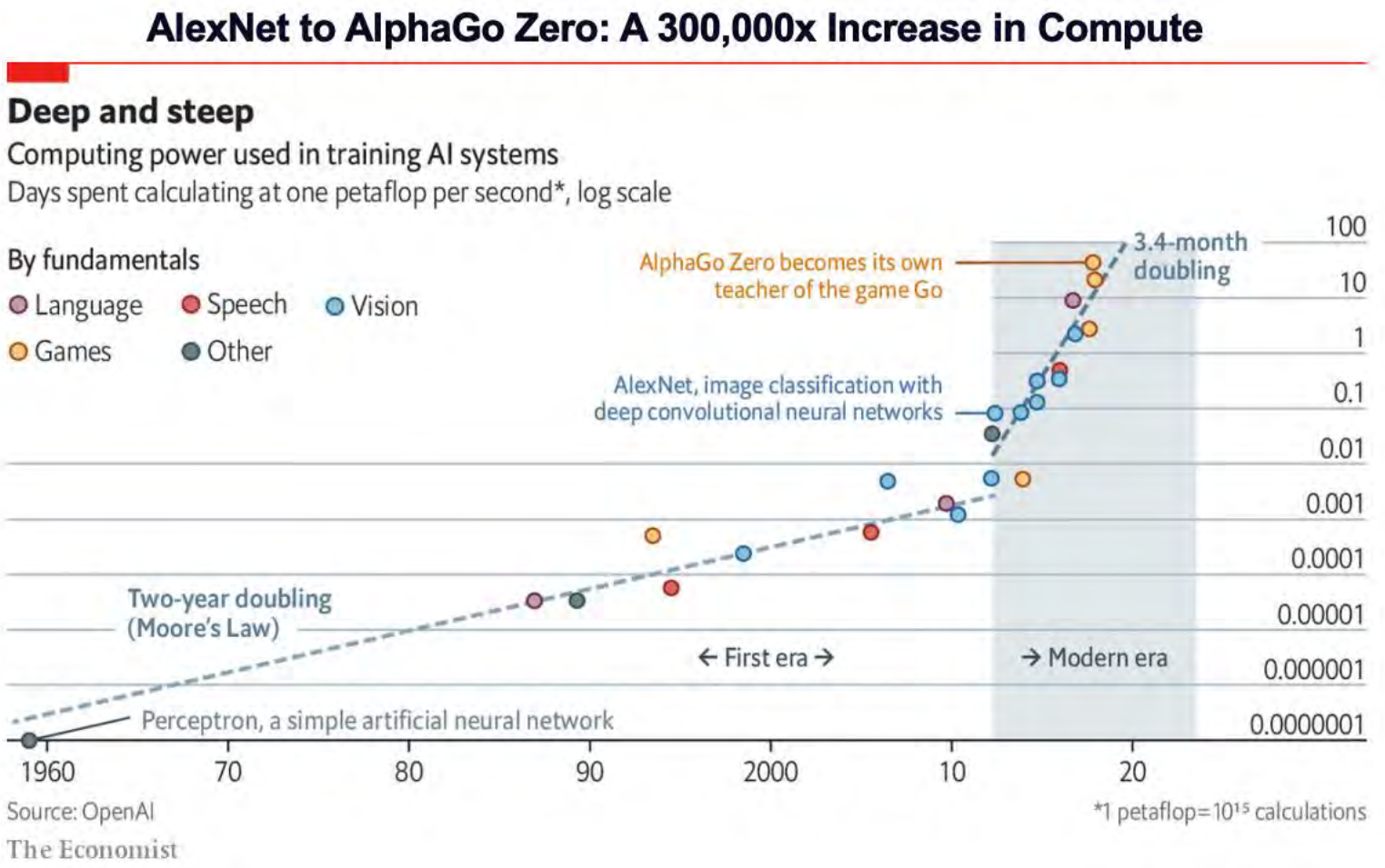
Find plot to demonstrate the interconnect bottleneck – back throughput and very energy expensive
Energy consumption of data movement

| Operation: | Energy (pJ) |
|---|---|
| 8b Add | 0.03 |
| 16b Add | 0.05 |
| 32b Add | 0.1 |
| 16b FP Add | 0.4 |
| 32b FP Add | 0.9 |
| 8b Multiply | 0.2 |
| 32b Multiply | 3.1 |
| 16b FP Multiply | 1.1 |
| 32b FP Multiply | 3.7 |
| 32b SRAM Read (8KB) | 5 |
| 32b DRAM Read | 640 |


V. Sze, Y. Chen, T. Yang and J. S. Emer, "Efficient Processing of Deep neural Networks", Springer Cham, (May 2022)

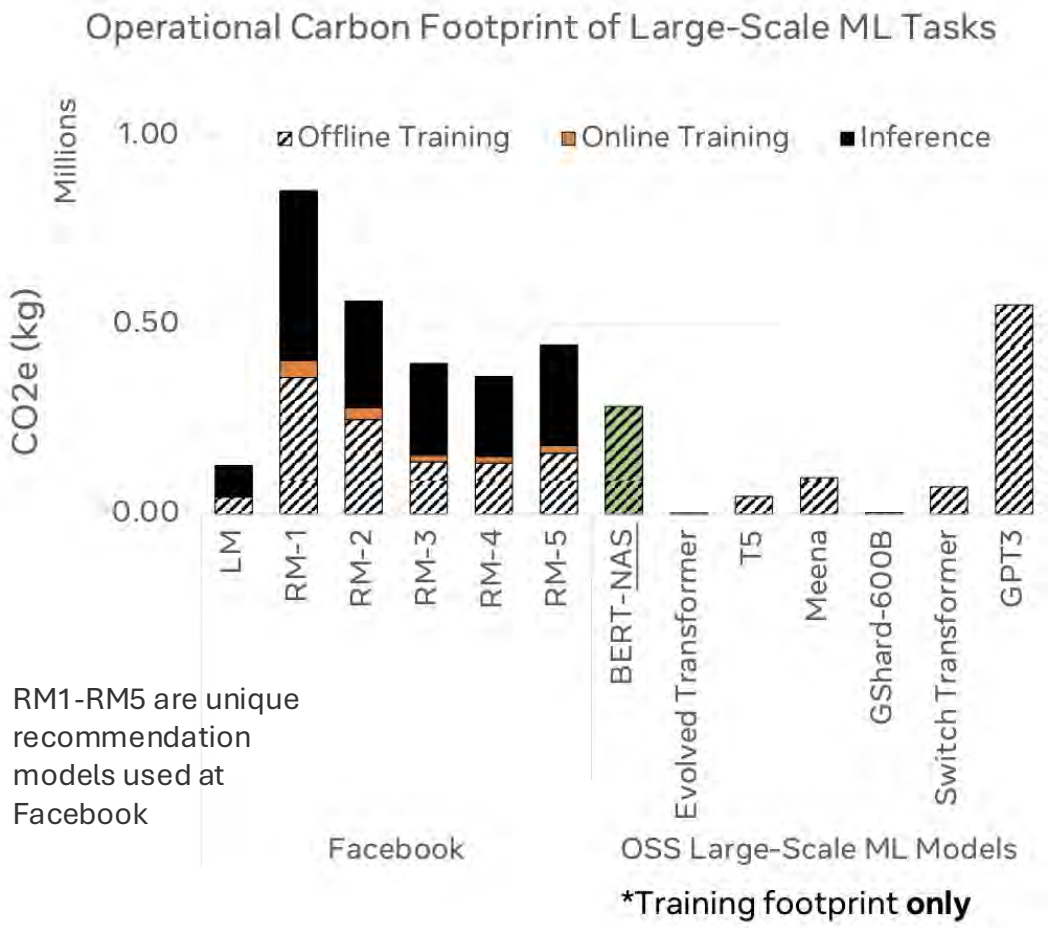
# AI applications Demand More Compute




"The cost of training machines is becoming a problem," *The Economist*. (Jun 2020)

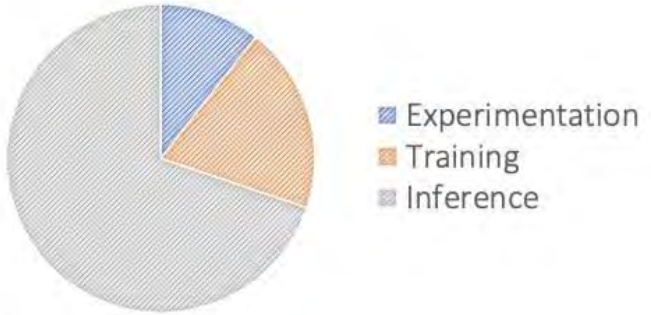
[1]

# AI Training vs. Inference: Energy Impact



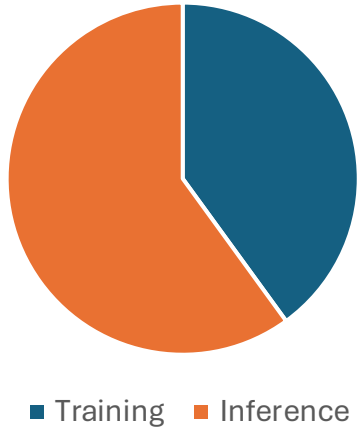

RM1-RM5 are unique recommendation models used at Facebook

**Facebook AI Compute Energy**




10:20:70 for Experimentation, Training, Inference [2]

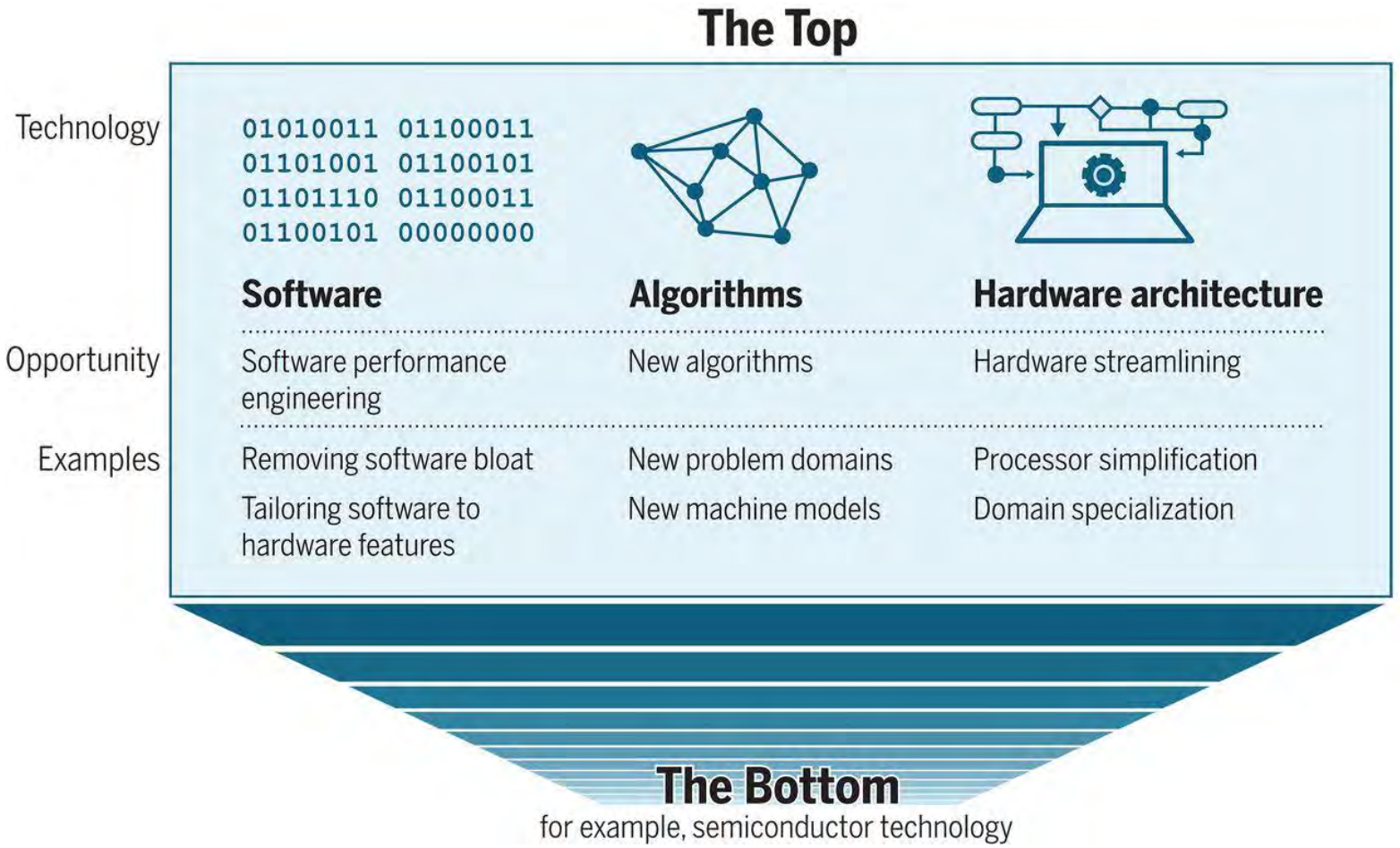
**Google AI Compute Energy**




"Across all three years [2019 – 2021], about **three-fifths** of the ML energy use was for **inference**, and **two-fifths** were for **training**." [1]

[1] D. Patterson et al., "The Carbon Footprint of Machine Learning Training Will Plateau, Then Shrink," in Computer (July 2022 )
[2] C.-J. Wu *et al.*, "Sustainable AI: Environmental Implications, Challenges and Opportunities." arXiv, (Jan 2022)

# Entering a New Computing Era




"In the post-Moore era, performance improvements from software, algorithms, and hardware architecture will increasingly require concurrent changes across other levels of the stack"

→ this will be easiest by relying on MODULAR DESIGN across the stack

C. E. Leiserson, *et. al.*, "There's plenty of room at the Top: What will drive computer performance after Moore's law?", *Science*, (2020)

# Computing Architecture Landscape

The most important trade-off in the post-Moore era is flexibility/programmability vs. efficiency (energy, area, and performance)

Increasing efficiency

**CPU**
Central Processing Unit

**GPU**
Graphics Processing Unit

**FPGA**
Field Programmable Gate Array

**CGRA**
Coarse-grained Reconfigurable Array

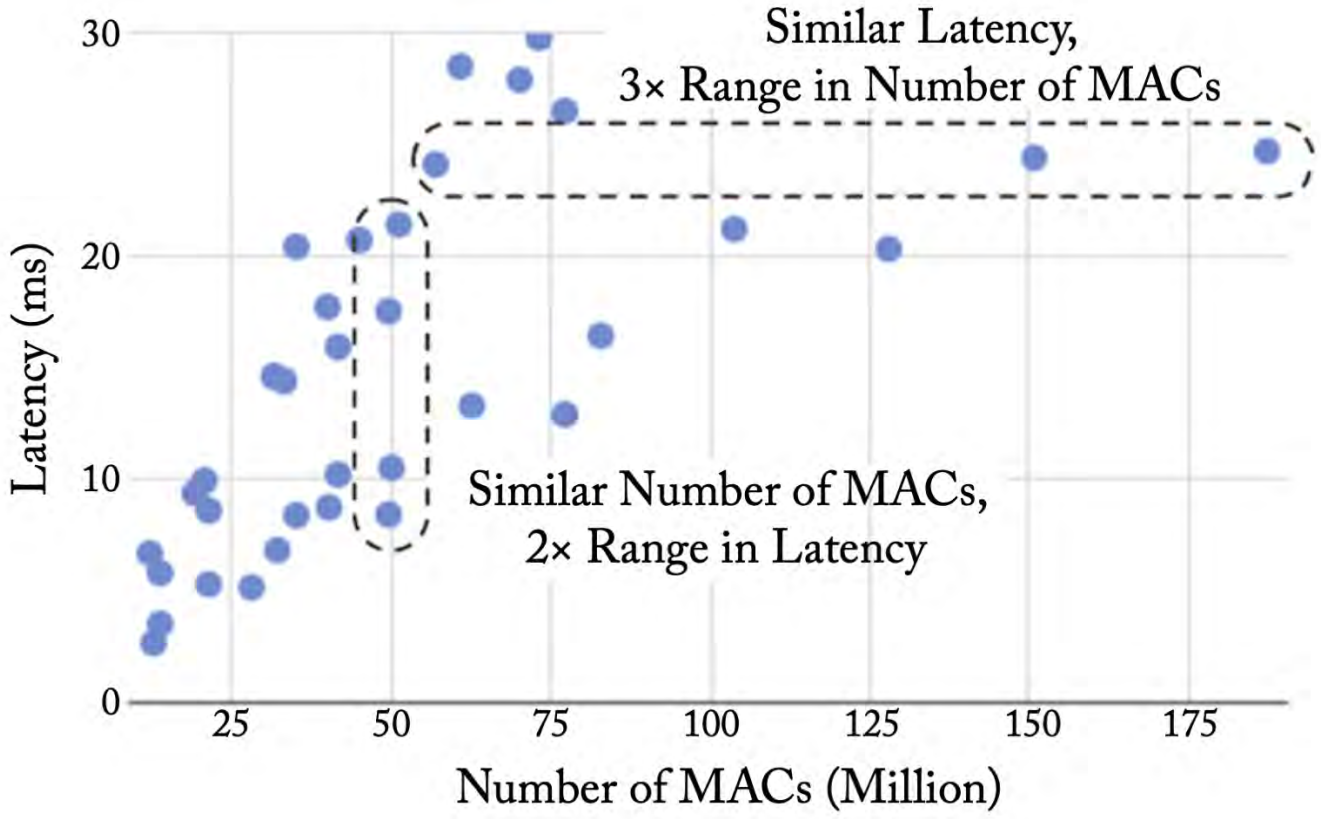
**ASIC**
Application-specific Integrated Circuit

Increasing flexibility

# Metrics for Evaluating Computing

- Accuracy
- Throughput and Latency
- Energy and Power
- Cost
- Flexibility
- Scalability

All with interdependent tradeoffs, that can be very application dependent




V. Sze, Y. -H. Chen, T. -J. Yang and J. S. Emer, "Efficient Processing of Deep neural Networks", Springer Cham, (May 2022)

# Coarse-Grain Reconfigurable Architecture

The most important trade-off in the post-Moore era is flexibility/programmability vs. efficiency (energy, area, and performance)

Increasing efficiency

**CPU**
Central Processing Unit

**GPU**
Graphics Processing Unit

**FPGA**
Field Programmable Gate Array

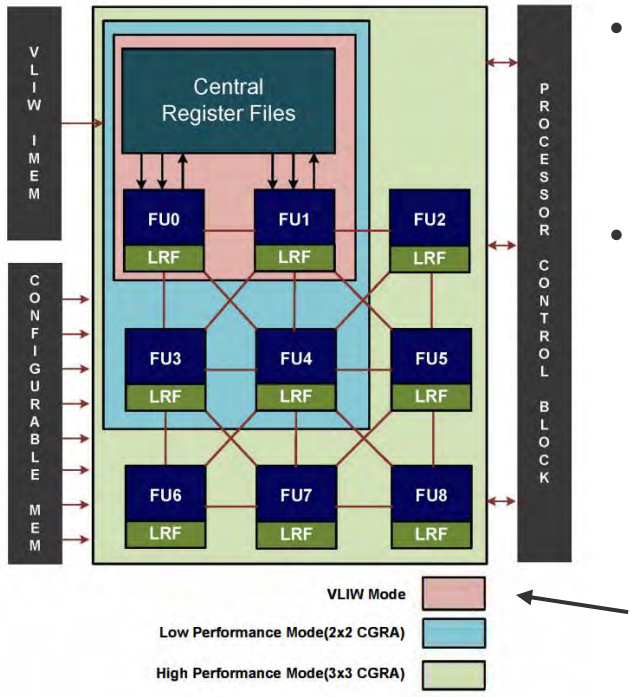
**CGRA**
Coarse-grained Reconfigurable Array

**ASIC**
Application-specific Integrated Circuit

Increasing flexibility

# Coarse-Grain Reconfigurable Architectures

"CGRAs are defined as domain-specific flexible hardware, on which the computation is performed both spatially and temporally and the execution is driven by configuration flow or data flow." [1]




Samsung Reconfigurable Processor (SRP) [2] can select how many processing elements are used

**Domain-specific Flexibility:**

- Some element of post-fabrication flexibility, but it's confined to specific uses or applications
- Not as general as FPGA

**Both Spatial and Temporal Computation:**

- Computational modules are tiles to form spatial architecture that can take advantage of parallel computing resources and dataflow communication
- Unlike FPGAs, The modules still have control logic and local register/scratchpad cache allowing for time multiplexing

**Configuration- or Data-driven Execution:**

- At the configuration or data level, operations are defined by propagation, as opposed to conventional CPU pipelining
- Configuration-driven CGRAs can exploit interconnects for efficient execution order
- Data-driven CGRAs are full dataflow machines, i.e. execution is only set by producer-consumer relationship

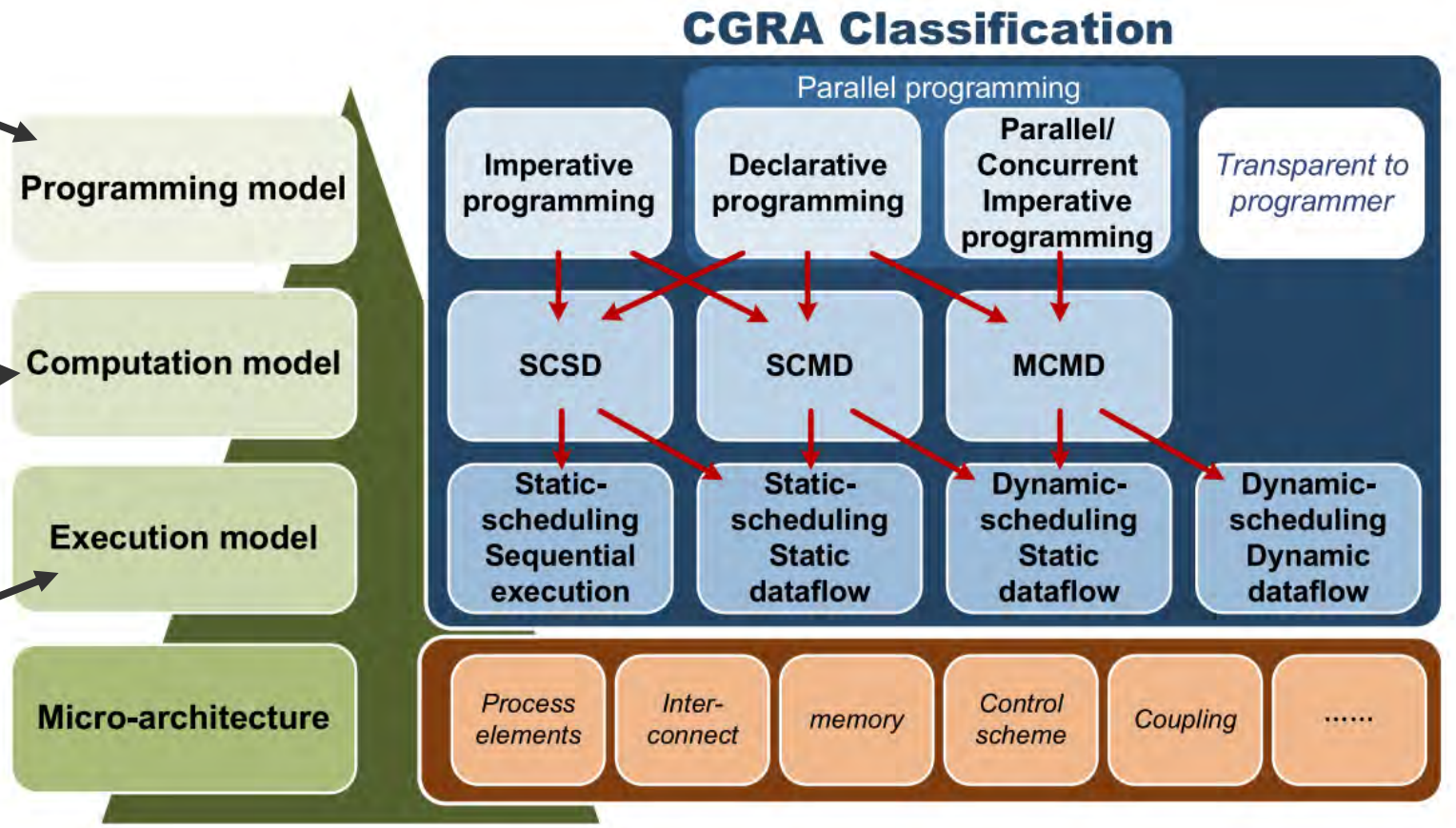

[1] L. Liu, *et. al.*, "A Survey of Coarse-Grained Reconfigurable Architecture and Design: Taxonomy, Challenges, and Applications", *ACM Computing Surveys*, (Nov 2020)
[2] C. Kim, *et. al.*, "ULP-SRP: Ultra low power Samsung Reconfigurable Processor for biomedical applications," in *2012 Int'l Conf. FPT*, Seoul, Korea (South): IEEE, (Dec. 2012)

# Coarse-Grain Reconfigurable Architectures

Abstracts away all the underlying computing system to describe algorithms and data structures

Represents the computational semantics that produces solutions for executing applications described by the programming model and reflects the physical characteristics of the system

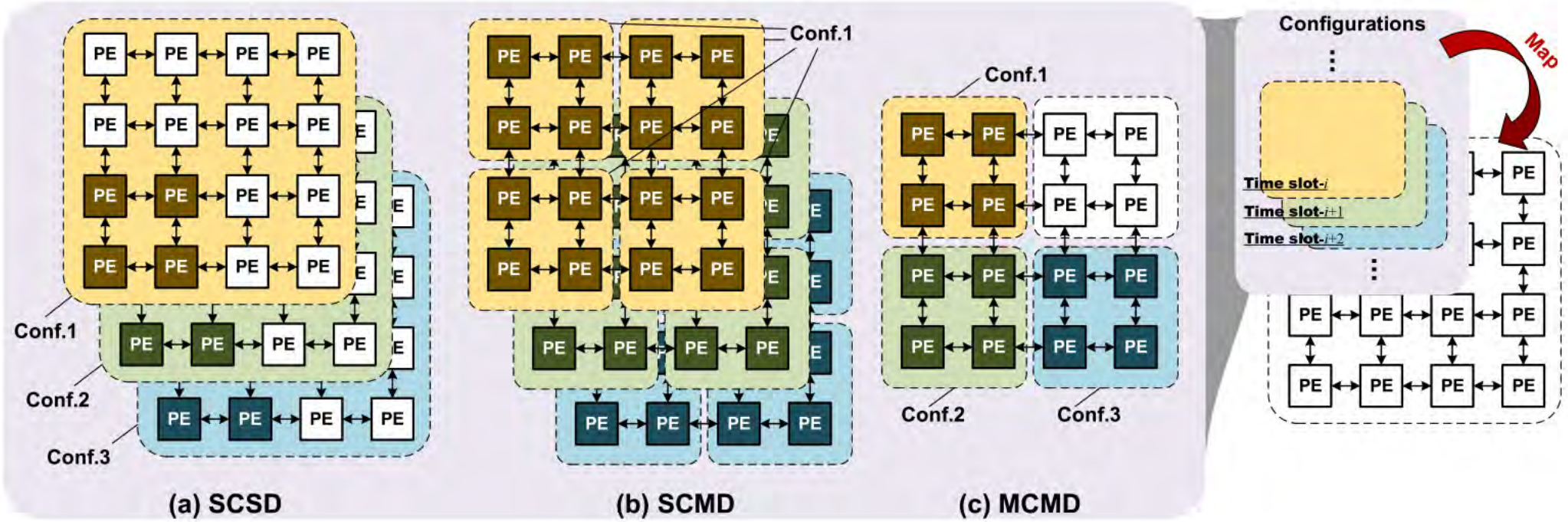
The abstract representation of the micro-architecture handling operation scheduling scheme




L. Liu, *et. al.*, "A Survey of Coarse-Grained Reconfigurable Architecture and Design: Taxonomy, Challenges, and Applications", *ACM Computing Surveys*, (Nov 2020)

# Coarse-Grain Reconfigurable Architectures

All CGRAs are MIMD (multiple instruction, multiple data) according to Flynn's taxonomy, so we can further categorize according to configuration




**Single configuration, single data (SCSD)**
Strength: instruction level parallelism, simple implementation

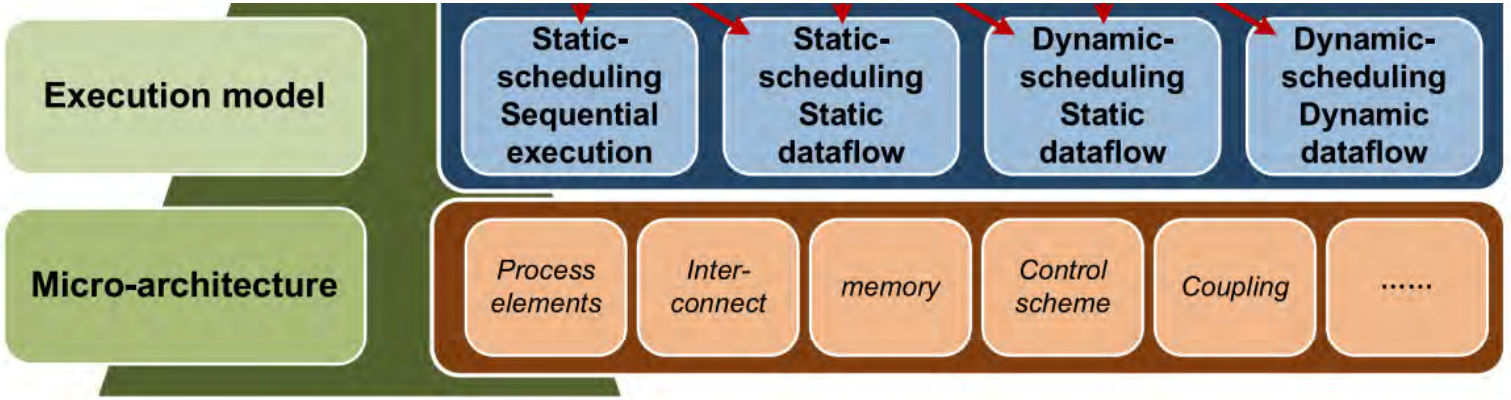
**Single configuration, multiple data (SCMD)**
Strength: data-level parallelism, like a spatially tiled SIMD

**Multiple configuration, multiple data (MCMD)**
Strength: thread-level parallelism, simultaneous multithreading and temporal multithreading

L. Liu, *et. al.*, "A Survey of Coarse-Grained Reconfigurable Architecture and Design: Taxonomy, Challenges, and Applications", *ACM Computing Surveys*, (Nov 2020)

# Coarse-Grain Reconfigurable Architectures

Execution model is like the lower-level instruction set determining how and when operations are performed




**Scheduling**

Refers to mechanism for fetching configurations and mapping them to hardware memory

- Static: configurations are mapped by the compiler and not updated at runtime. Lower hardware overhead at expense of performance
- Dynamic: configuration mapping is done on the fly determined by resource allocation. Higher performance at the expense of circuit overhead

**Execution**

Refers to the mechanism for executing computation within a single configuration

- Sequential: operations are executed in the order that the compiler defines before runtime
- Dataflow: operations are executed based on operand availability
  - Static: operand arrival is signaled by tokens, only once all are present is operation executed
  - Dynamic: operand arrival is signaled by tokens and tagged, only once all with matching tags are present is the operation executed (to allow for multiple instances of the same routine)

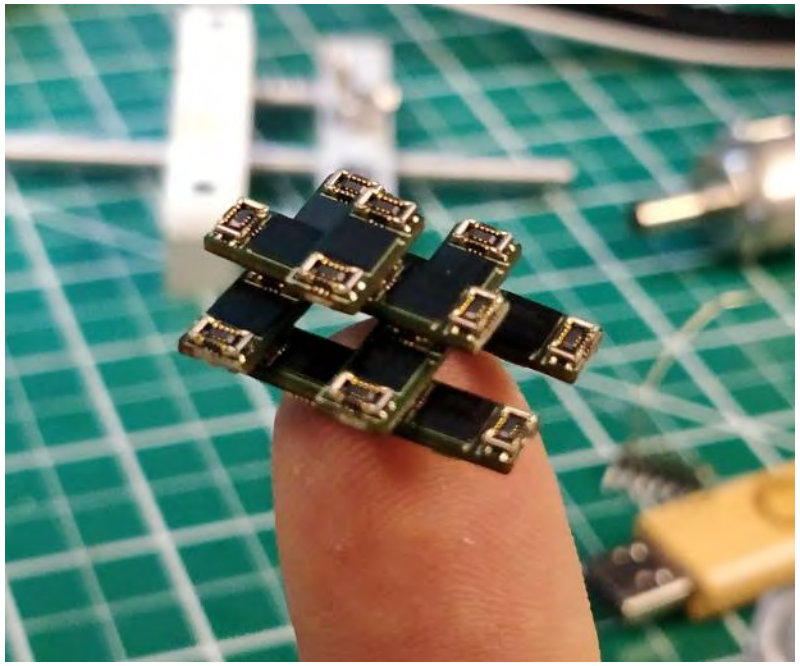
L. Liu, *et. al.*, "A Survey of Coarse-Grained Reconfigurable Architecture and Design: Taxonomy, Challenges, and Applications", *ACM Computing Surveys*, (Nov 2020)

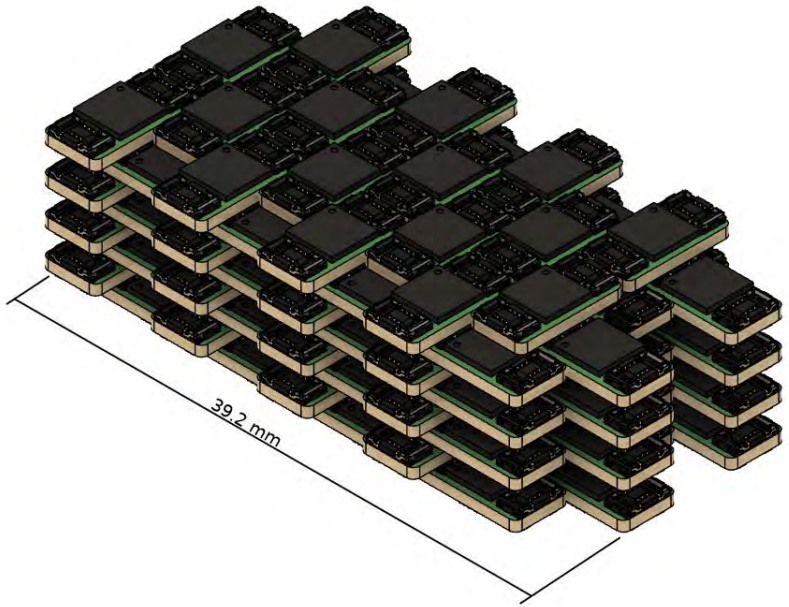
# Coarse-Grain Reconfigurable Architectures

DICE: Discrete Integrated Circuit Elements

computation: MCMD

execution: dynamic scheduling, dynamic dataflow



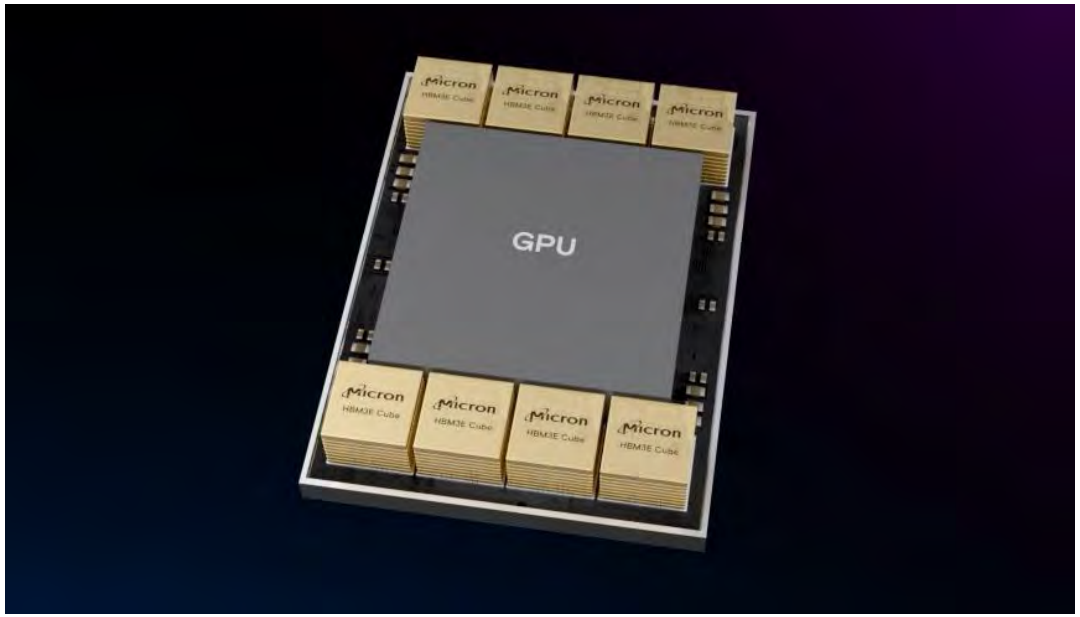



Citations

# Coarse-Grain Reconfigurable Architectures

CGRAs are not saved from the memory wall problem!!

Depending on the shared memory allocation, they could be more sensitive to the memory wall problem than vector machines (full dataflow architectures) because multiple PEs running multiple threads could run into scheduling conflicts accessing shared memory, and therefore increase latency




State of the art High Bandwidth Memory solution → HBM3E from Micron capable of 1.2 TB/s with 24GB of capacity [1]

To be integrated into upcoming Nvidia H200 GPU with 141 GB at 4.8 TB/s

[vid] Micron HBM3E, https://www.micron.com/products/memory/hbm/hbm3e

[1] “Micron Commences Volume Production of Industry-Leading HBM3E Solution to Accelerate the Growth of AI | Micron Technology” Press release (Feb 2024)

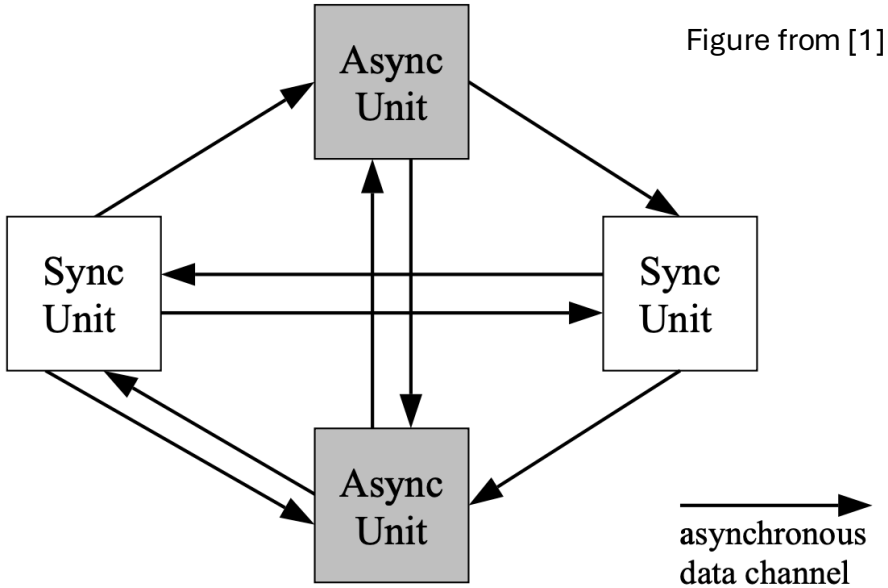
# Globally Asynchronous, Locally Synchronous

GALS: An asynchronous method for sending data signals across clock domains

Metastability: THE major issue in designing GALS systems

- At any boundary between independent clock domains the system can enter a metastable point which requires significant power, area, and timing costs to overcome
- Main strategies for reliable data transfer:
  - Pausible-clock generators
  - FIFO buffers
  - Boundary synchronization

Figure from [1]

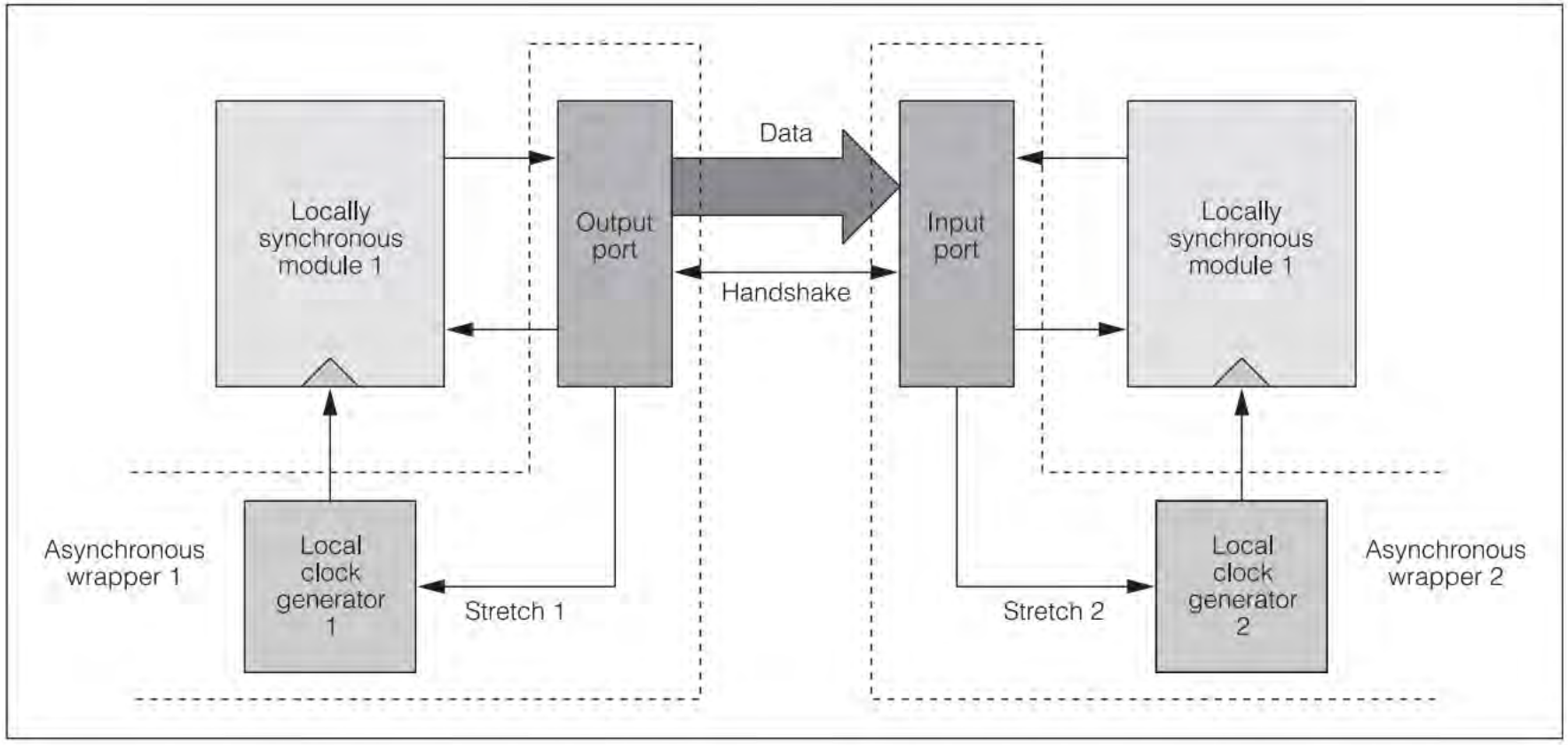



- Locally synchronous operations are required at minimum at combinational logic level
- Asynchronous buffers introduce some amount of overhead so the tradeoff becomes area and energy versus flexibility

M. Krstic, E. Grass, F. K. Gürkaynak and P. Vivet, "Globally Asynchronous, Locally Synchronous Circuits: Overview and Outlook," *IEEE Design & Test of Computers, (*2007)
[1] J. Muttersbach, *et. al.* , “Practical design of globally-asynchronous locally-synchronous systems,” in *(ASYNC 2000)*

# Globally Asynchronous, Locally Synchronous

Pausible clocking scheme:



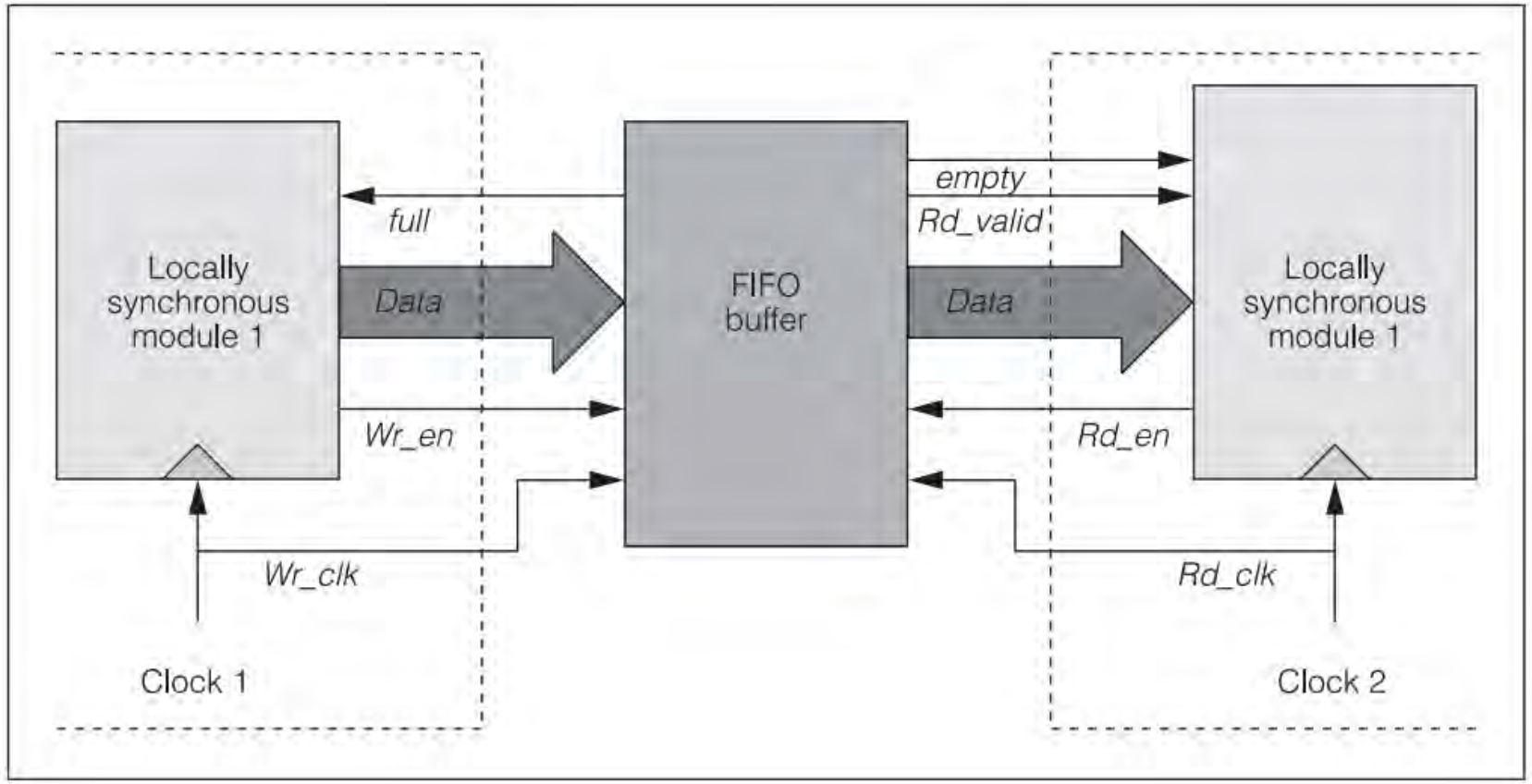

Local clock generators within the asynchronous wrappers drive the locally synchronous modules so that data transmission/receiver signals are guaranteed to be OFF when data is transferred across clock domains

*Perhaps similar to a module-scale version of the Variable Activation QFP clocking scheme I've discussed in the past*

M. Krstic, E. Grass, F. K. Gürkaynak and P. Vivet, "Globally Asynchronous, Locally Synchronous Circuits: Overview and Outlook," *IEEE Design & Test of Computers, (*2007)

# Globally Asynchronous, Locally Synchronous

FIFO buffer scheme:

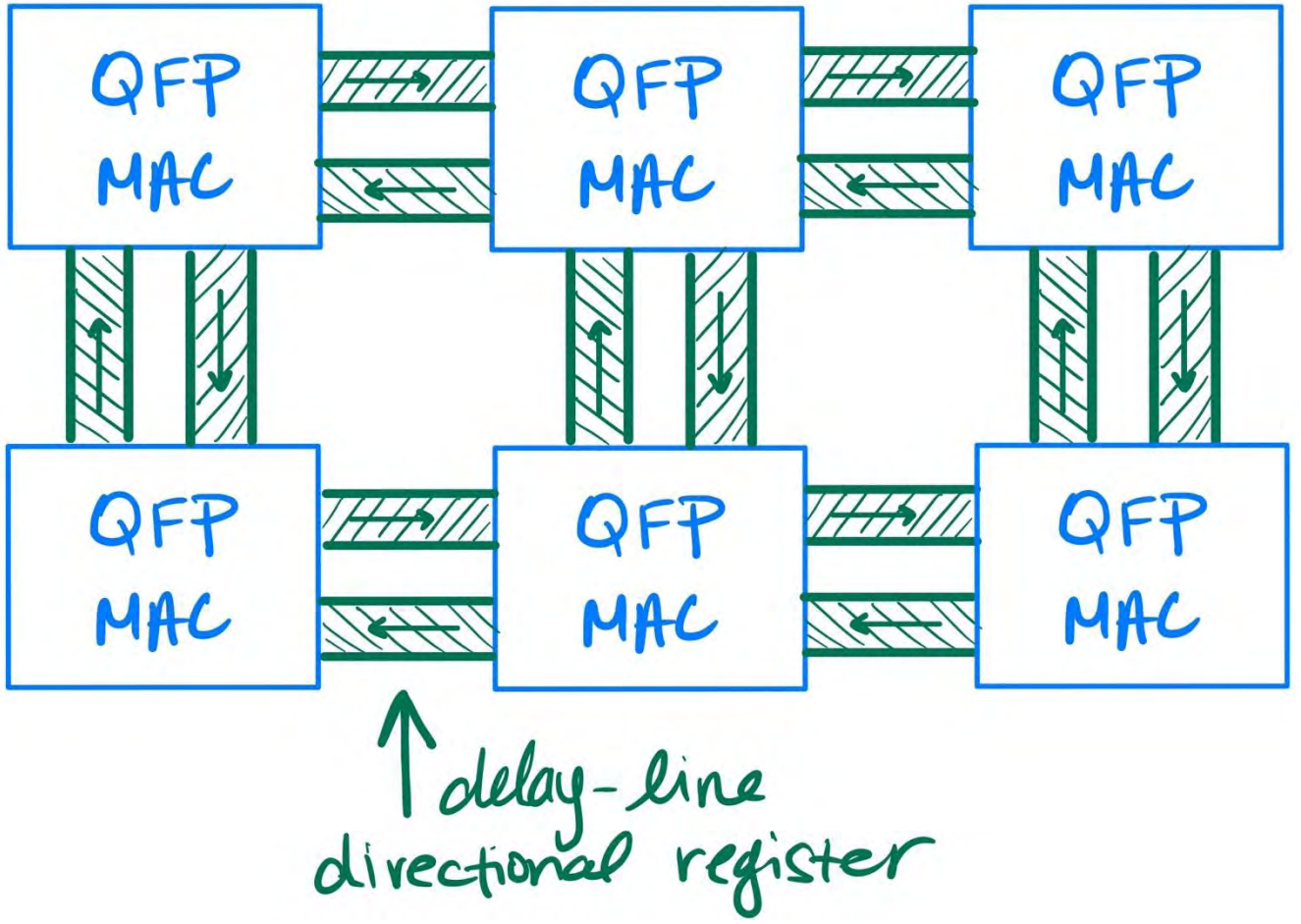



- FIFO buffers are added between adjacent synchronous modules, moving data in one direction. Additional buffers can be added to allow multi-directional dataflow.
- For large SoC applications, this can be best because it hides long interconnect delays.
- Most common implementation in GALS systems

M. Krstic, E. Grass, F. K. Gürkaynak and P. Vivet, "Globally Asynchronous, Locally Synchronous Circuits: Overview and Outlook," *IEEE Design & Test of Computers, (*2007)

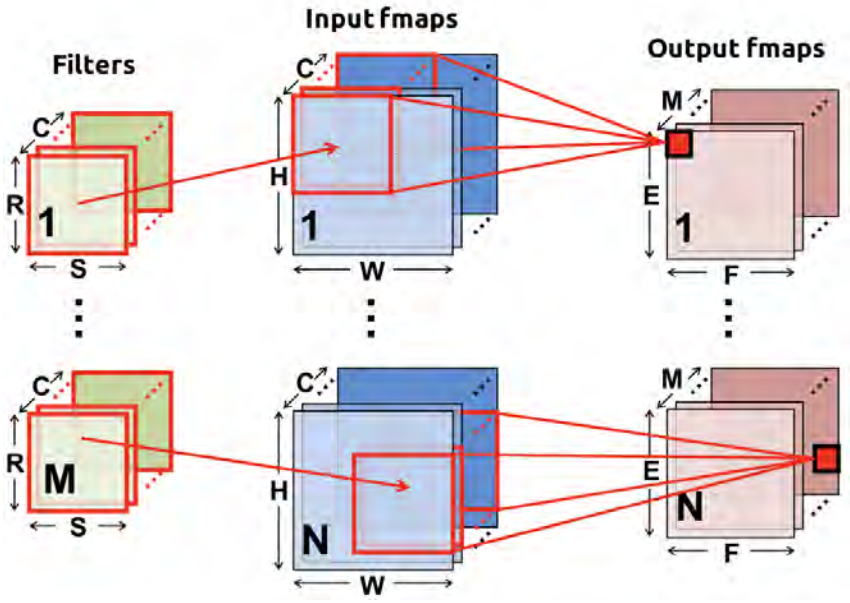
# Globally Asynchronous, Locally Synchronous

Delay Line Memory behaving as FIFO buffers

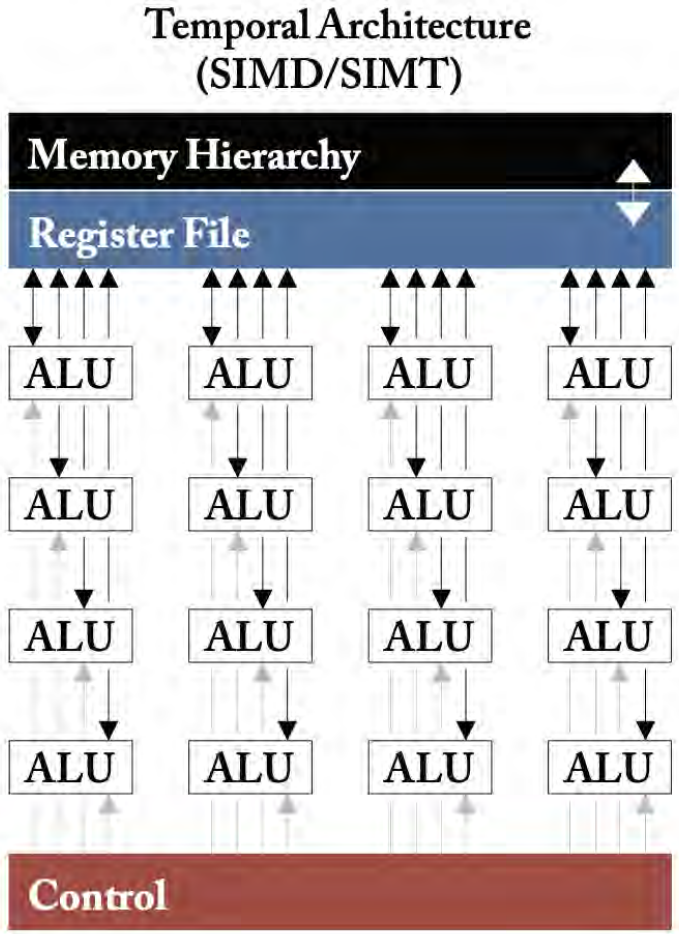


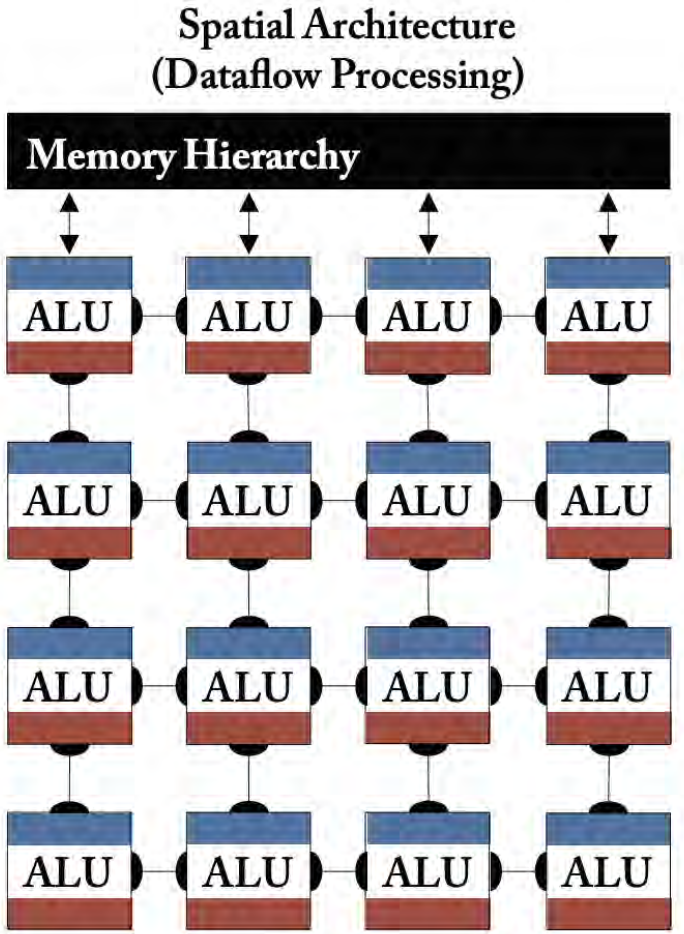
# DNN Accelerator Design

Deep Neural Networks (DNN) can exploit highly parallel computation




Convolutional Neural Network (CNN) layer



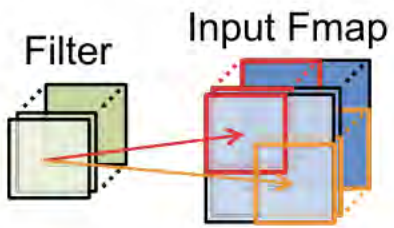

V. Sze, Y. -H. Chen, T. -J. Yang and J. S. Emer, "Efficient Processing of Deep neural Networks", Springer Cham, (May 2022)

# DNN Accelerator Design

Hardware accelerators can exploit data reuse to minimize data movement

**Convolutional Reuse**

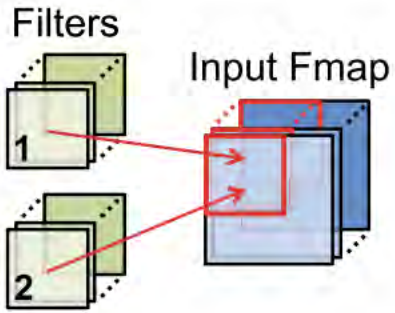
CONV layers only
(sliding window)




Reuse: Activations
Filter weights

**Fmap Reuse**

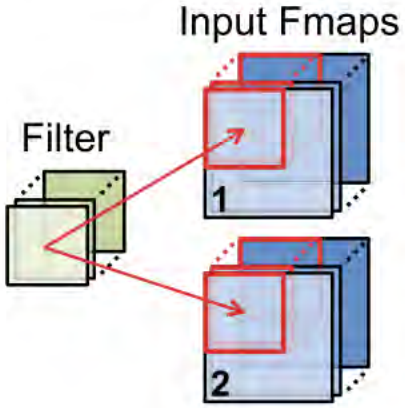
CONV and FC layers



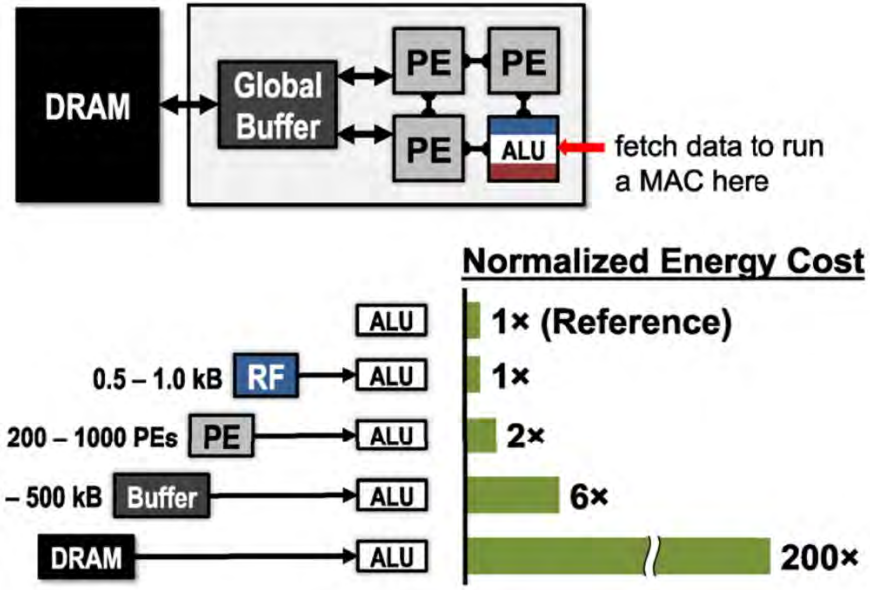

Reuse: Activations

**Filter Reuse**

CONV and FC layers
(batch size > 1)




Reuse: Filter weights

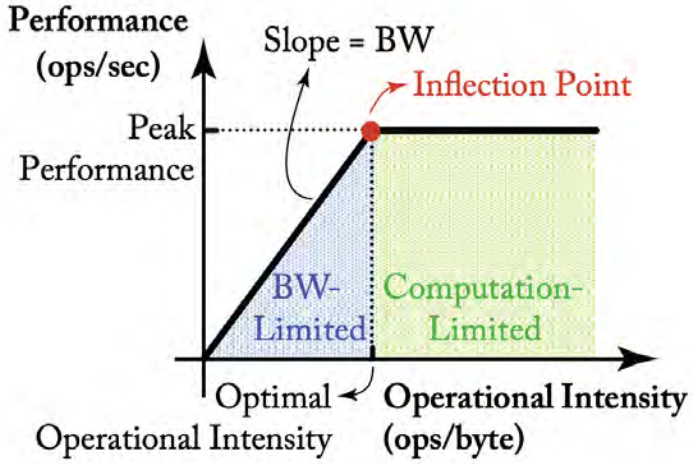



V. Sze, Y. -H. Chen, T. -J. Yang and J. S. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey", *Proceedings of the IEEE,* (Dec 2017)

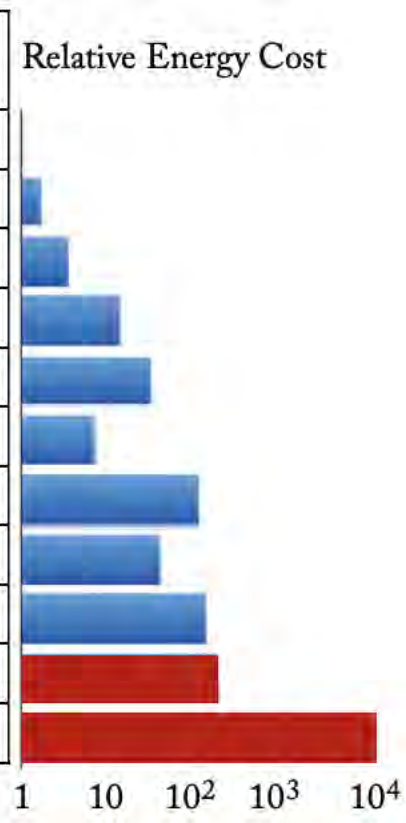
# DNN Accelerator Design

Can improve energy performance by exploiting sparsity:

$$\frac{\text{operations}}{\text{joule}} = \frac{\text{effectual operations} + \text{unexploited ineffectual operations}}{\text{joule}} \times \frac{\text{effectual operations}}{\text{effectual operations} + \text{unexploited ineffectual operations}} \times \frac{1}{\frac{\text{effectual operations}}{\text{operations}}}.$$

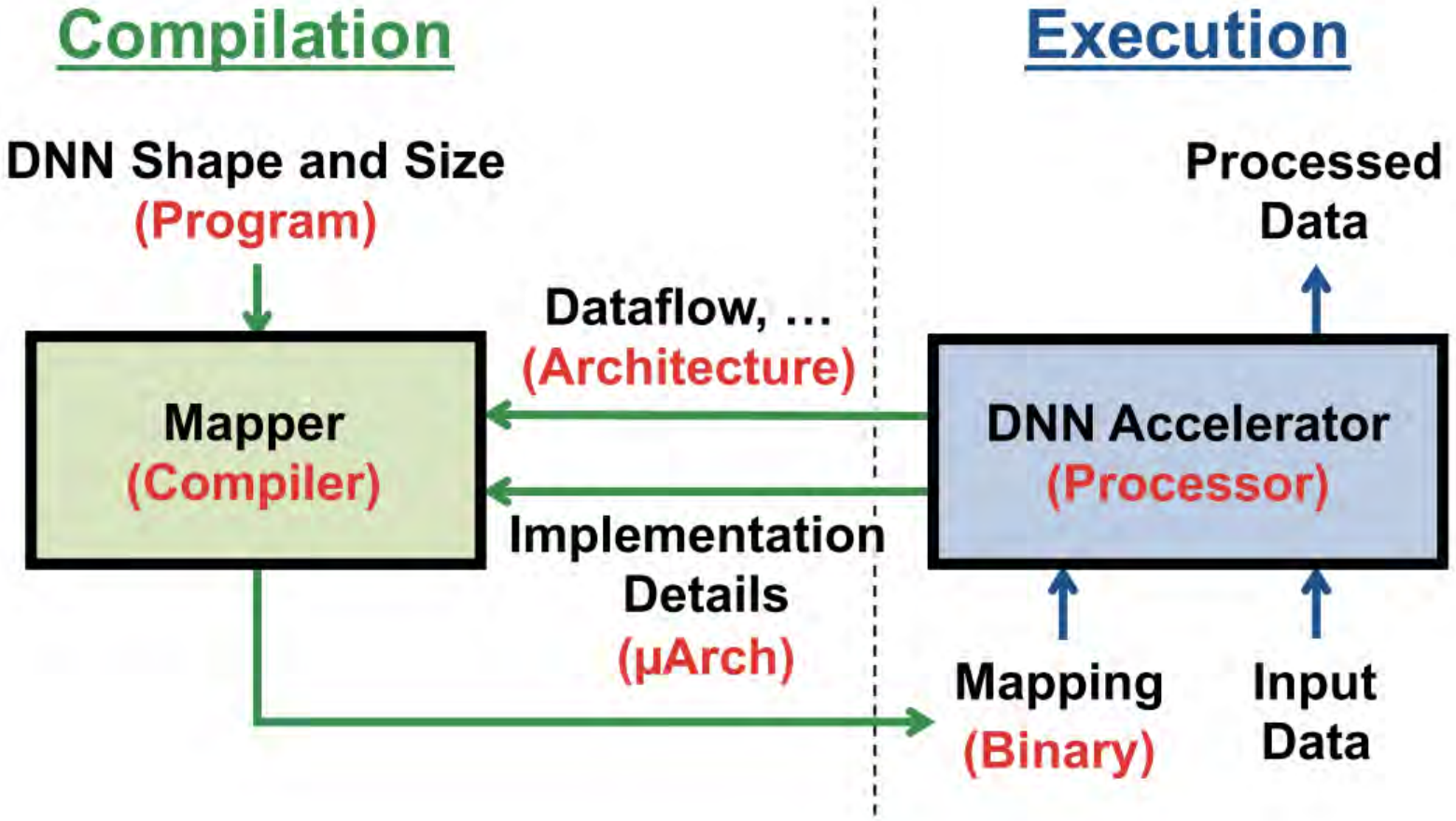



| Operation: | Energy (pJ) |
|---|---|
| 8b Add | 0.03 |
| 16b Add | 0.05 |
| 32b Add | 0.1 |
| 16b FP Add | 0.4 |
| 32b FP Add | 0.9 |
| 8b Multiply | 0.2 |
| 32b Multiply | 3.1 |
| 16b FP Multiply | 1.1 |
| 32b FP Multiply | 3.7 |
| 32b SRAM Read (8KB) | 5 |
| 32b DRAM Read | 640 |




V. Sze, Y. -H. Chen, T. -J. Yang and J. S. Emer, "Efficient Processing of Deep neural Networks", Springer Cham, (May 2022)

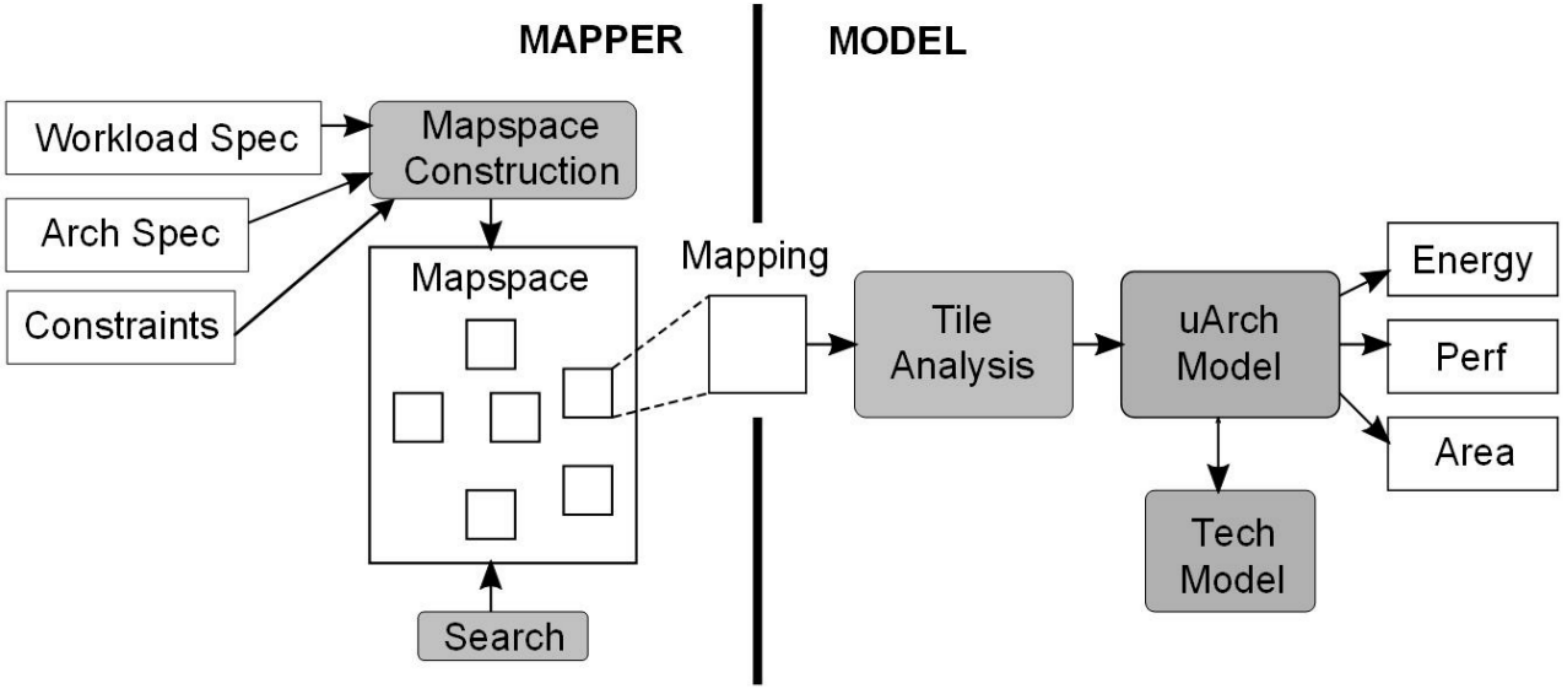
# Accelerator Operation Workflow



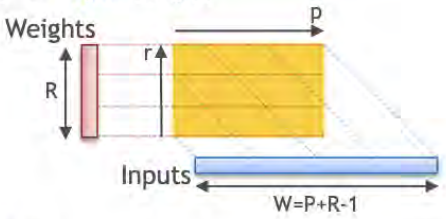

Accelerator workflow (black text) in comparison to typical CPU workflow (red text)

V. Sze, Y. -H. Chen, T. -J. Yang and J. S. Emer, "Efficient Processing of Deep Neural Networks: A Tutorial and Survey", *Proceedings of the IEEE,* (Dec 2017)

# Timeloop: Architecture Modeling Tool

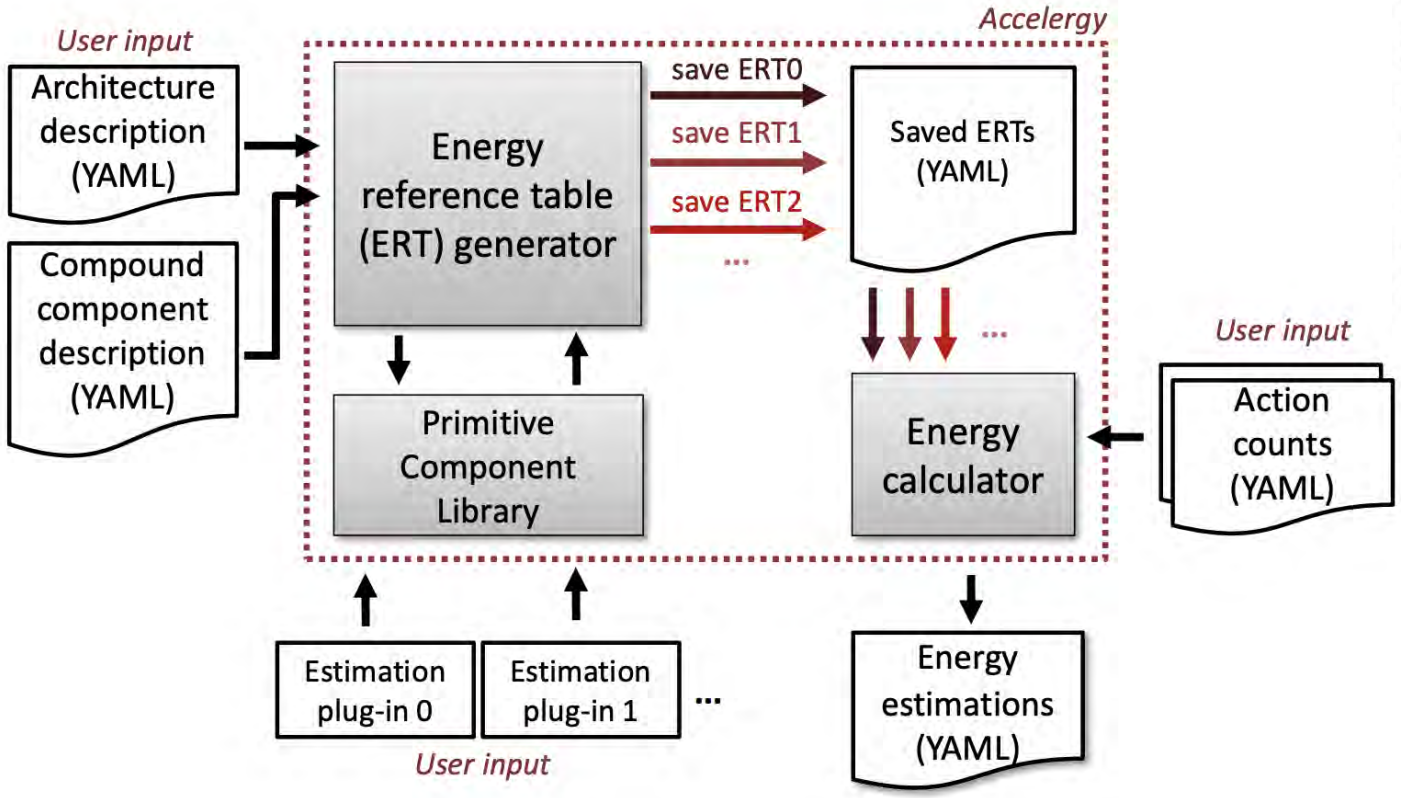



```
1  // === 1D Convolution Workload ===
2  for r=[0:R):
3   for p=[0:P):
4    Output[p] += Weight[r] * Input[r+p];
5
6  // === Mapping ===
7  // DRAM
8  for r3=[0:R3):
9   for p3=[0:P3):
10   // GBuf
11   for r2=[0:R2):
12    for p2=[0:P2):
13     // Spatial: GBuf->RFile
14     parallel_for r1=[0:R1):
15      parallel_for p1=[0:P1):
16       // RFile
17       for r0=[0:R0):
18        for p0=[0:P0):
19         p = p3*P2*P1*P0 + p2*P1*P0 + p1*P0 + p0;
20         r = r3*R2*R1*R0 + r2*R1*R0 + r1*R0 + r0;
21         Output[p] += Weight[r] * Input[r+p];
```




A. Parashar *et al.*, "Timeloop: A Systematic Approach to DNN Accelerator Evaluation," in *2019 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)* (Mar. 2019)

# Accelergy: Energy Estimator for Timeloop




**Example 1: a counter primitive component class**

```
1 name: counter #class name
2 attributes:   #default attributes
3   technology: 65nm
4   datawidth: 16
5 actions: #list of actions
6   - name: count
7   - name: idle
```

**Table 1: Selected components, hardware attributes, and action types in DNN primitive component library**

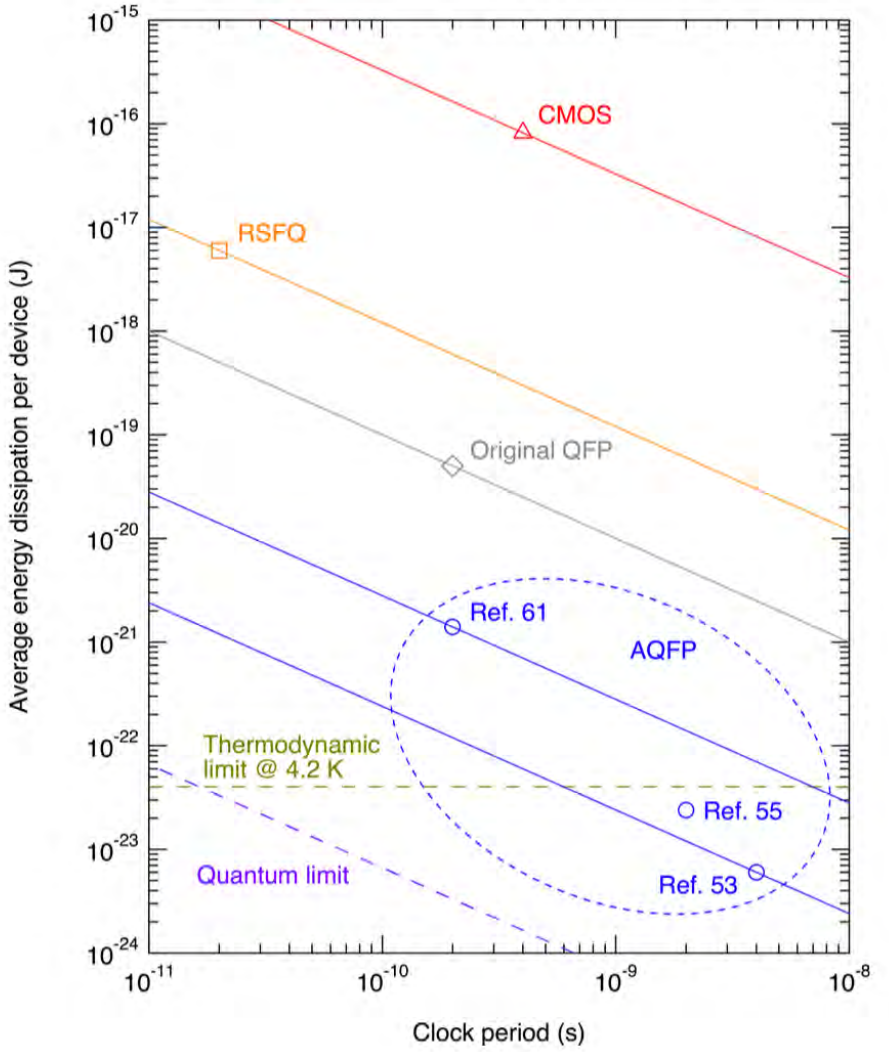
| Primitive Components | Hardware Attributes | Action Types |
|---|---|---|
| SRAM | width<br>depth<br># of read ports<br># of write ports<br># of banks | random_read<br>random_write<br>repeated_read<br>repeated_data_write<br>bypassed_read |
| MAC | bitwidth<br># of pipeline stages | random_MAC<br>constant_MAC<br>gated_MAC<br>zero_gated_MAC |
| counter | count limit | count |
| router | FIFO width<br>crossbar input ports<br>crossbar output ports | random_transfer<br>repeated_transfer<br>gated_transfer |
| wire | data width<br>wire length | random_transfer<br>repeated_ transfer |

Y. N. Wu, J. S. Emer, and V. Sze, "Accelergy: An Architecture-Level Energy Estimation Methodology for Accelerator Designs," in *2019 IEEE/ACM ICCAD*, Nov. 2019

# Extending the Tools for AQFP

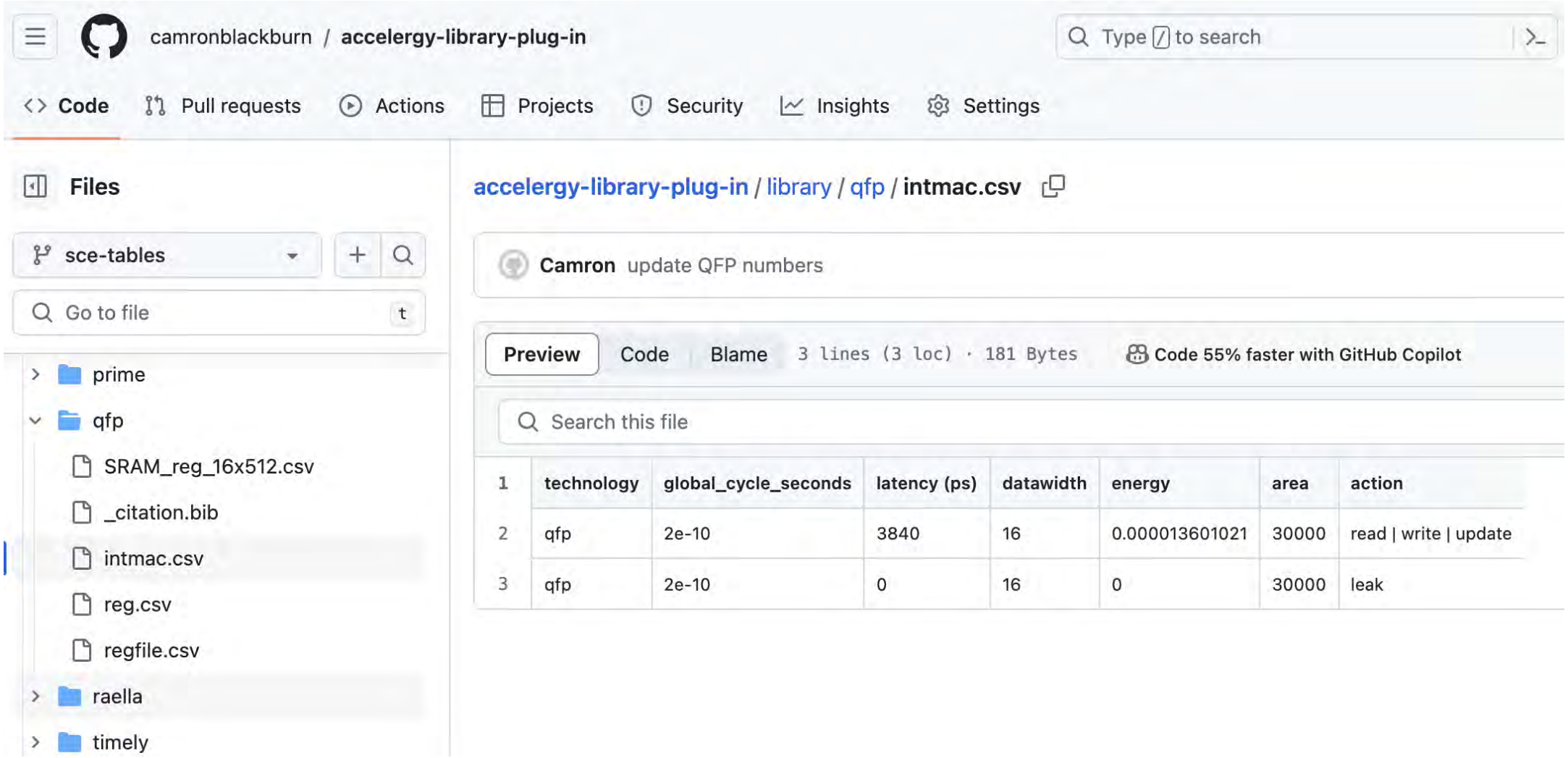
**A**diabatic **Q**uantum **F**lux **P**arametron

- Ultra-low switching energy, $\gg I_c\Phi_0$ because of adiabatic operation
  - **1.4zJ** ($10^{-21}$J) experimentally recorded energy dissipation for a single QFP switch [1]
- Clock speed comparable to CMOS at **5-10 GHz**
- High gain device
  - **50x – 100x gain** from a few uAs of current
- Decently robust to issues of flux trapping (when compared to SFQ)




[1] N. Takeuchi et. al., Appl. Phys. Lett., vol. 144, no. 4, p. 042602 (2019)

# Extending the Tools for AQFP

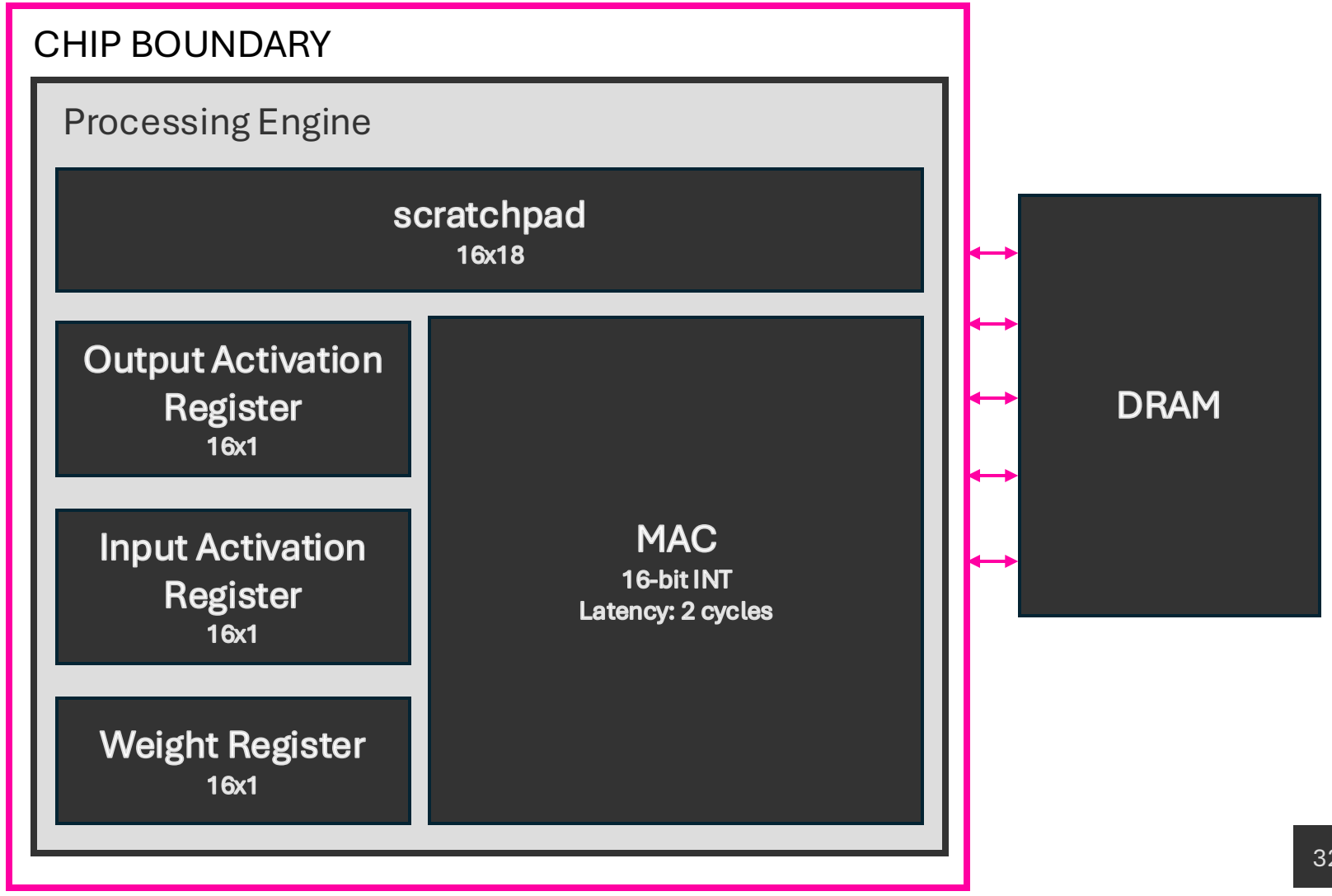



| 1 | technology | global_cycle_seconds | latency (ps) | datawidth | energy | area | action |
|---|---|---|---|---|---|---|---|
| 2 | qfp | 2e-10 | 3840 | 16 | 0.000013601021 | 30000 | read \| write \| update |
| 3 | qfp | 2e-10 | 0 | 16 | 0 | 30000 | leak |

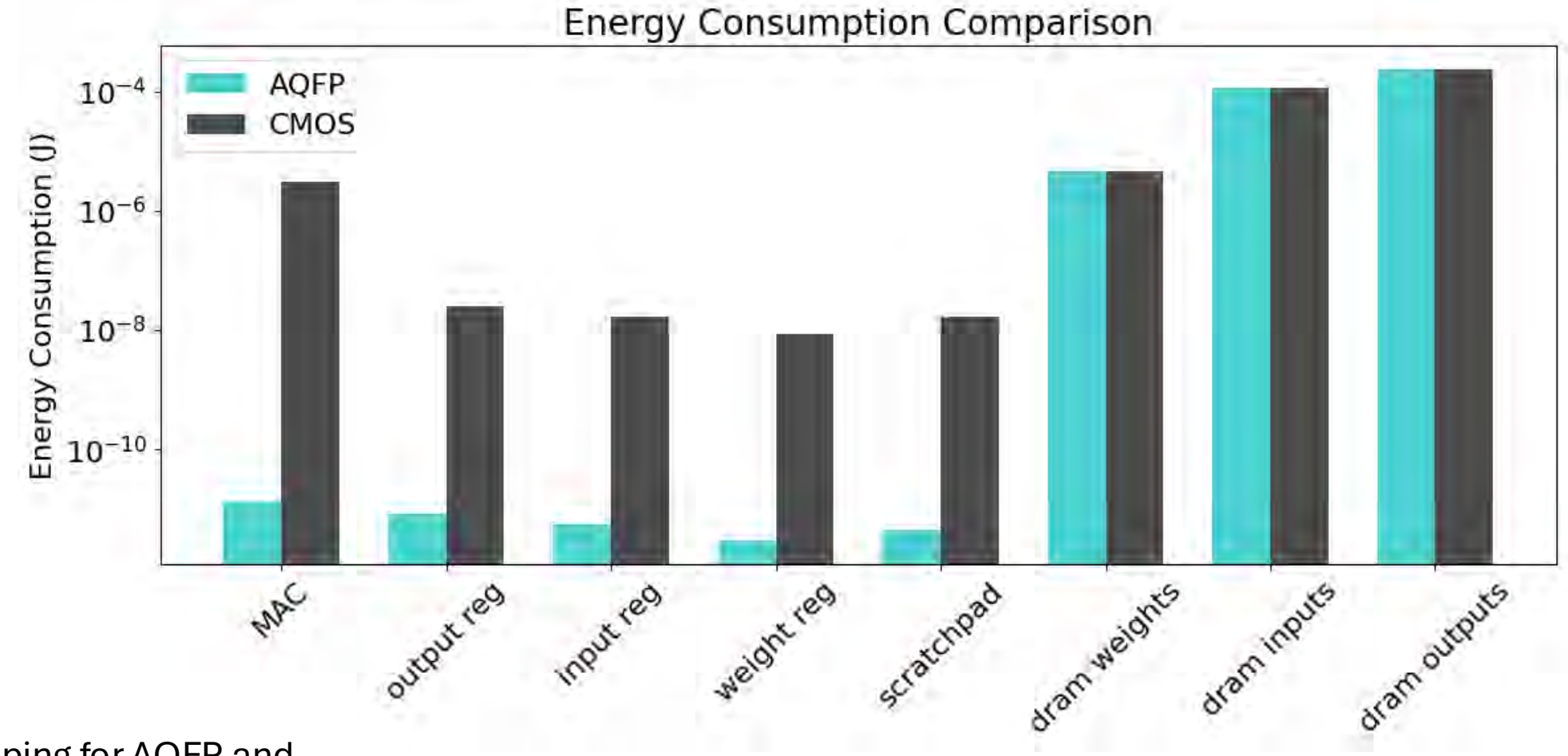
# Single PE example: CMOS vs AQFP

Example single PE architecture run for 45nm CMOS default tool values and QFP projected performance

```
// Single PE loop nest

int i[N][C][W][H];      # Input activations
int w[M][C][R][S];      # Filter Weights
int o[N][M][P][Q];      # Output activations

// Level 1 - DRAM temporal
for n1 = [0, N/N0) {
  for m1 = [0, M/M0) {
    for c1 = [0, C/C0) {

    // Level 0 - PE temporal
      for r0 = [0, R) {
        for s0 = [0, S){
          for m0 = [0, M0) {
            for c0 = [0, C0) {
              for n0 = [0, N0) {
                for p0 = [0, P) {
                  for q0 = [0, Q) {
                    n = n1*N0 + n0
                    m = m1*M0 + m0
                    c = c1*C0 + c0
                    r = r0
                    s = s0
                    p = p0
                    q = q0

                    o[n][m][p][q] += i[n][c][p+r][q+s]
                                   *  w[m][c][r][s]
}. . . }
```



# Single PE example: CMOS vs AQFP




*same mapping for AQFP and CMOS. And no change in DRAM

# Next steps for AQFP Architecture Modeling

- Model different types of memory
  - How to account for interconnect length in memory modeling?
  - Compare the tradeoff of different types of superconducting memory
- More accurate primitive models
  - Extend into floating point operations
- Include area estimates
- Explore mapping optimizations on QFP logic versus CMOS
  - Requires capturing the unique tradeoffs in superconductivity vs semiconductor IC

# Superconducting Electronics Design

# CONTENTS

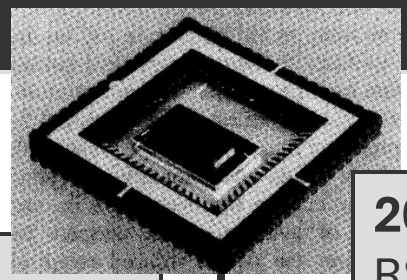
- Superconducting digital logic timeline
- Review of Superconductivity Basics
  - Macroscopic Quantum Model
  - Flux Quantization
  - Josephson Junctions
- Adiabatic Quantum Flux Parametron
  - AQFP analysis: Node Method
  - AQFP Device Characteristics
    - Energy Dissipation
  - AQFP analysis: Circuit Quantization

# Superconducting Digital Logic

**1956** Dudley Buck invents cryotron switching device [1]

**1960** Parametron-based tunnel diode pair logic (QFP inspiration) [2]

**1970s** IBM "Josephson Computer" with voltage state logic [3]



**1977** Likharev invents parametric quantron (PQ) logic (rf-SQUID) [4]

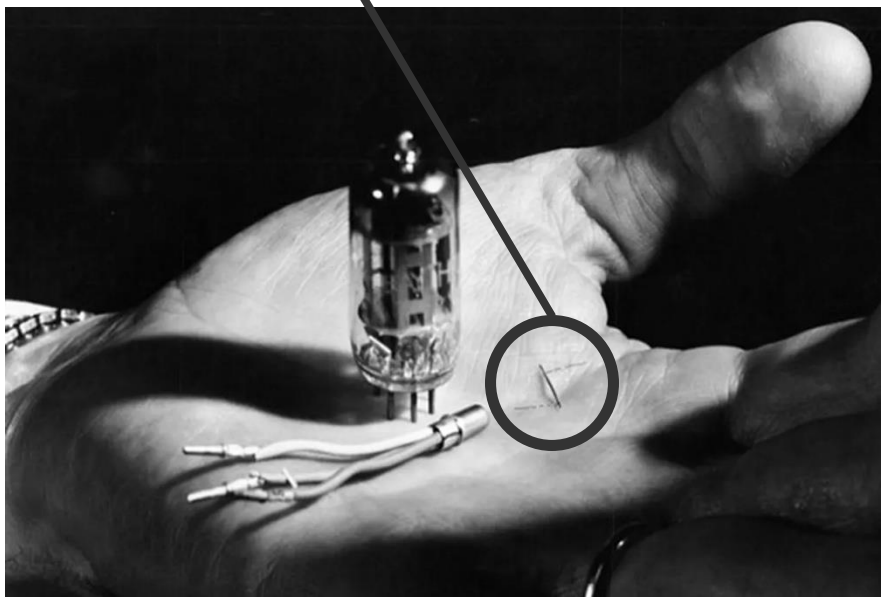
**1986** DC Flux parametron, renamed to QFP [5]

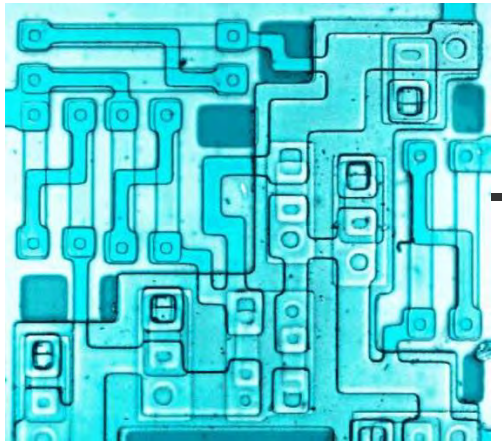
**1991** RSFQ take off for ultrafast operating speed [6]

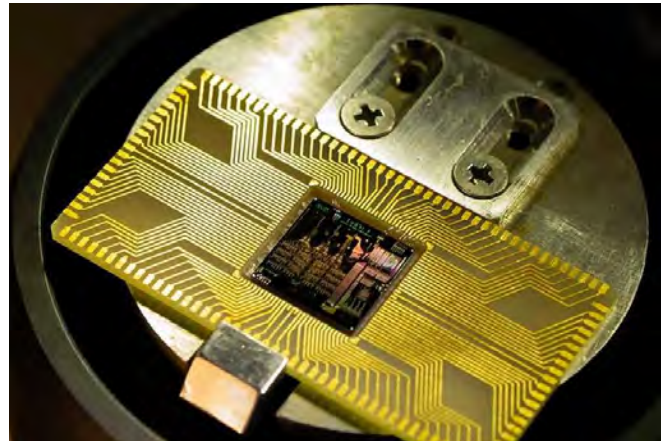
**2010s** energy efficient RSFQ modifications are explored – ERSFQ, eeSFQ, DSFQ, . . .

**2011** Adiabatic operation of QFP is revived [7]

**2020** MANA, first adiabatic superconducting microprocessor [8]





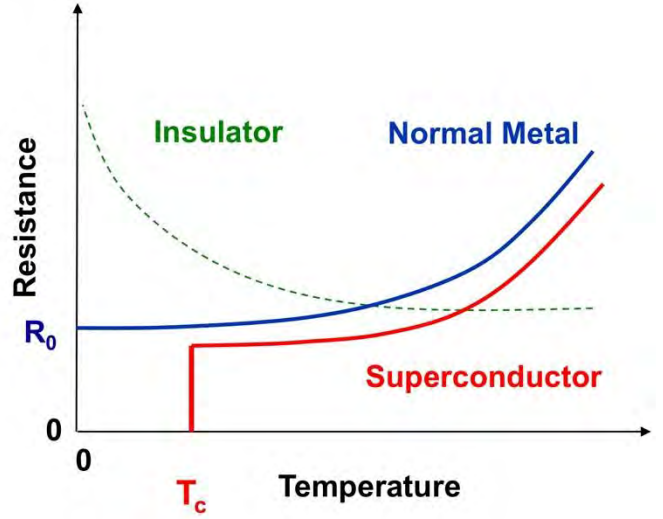


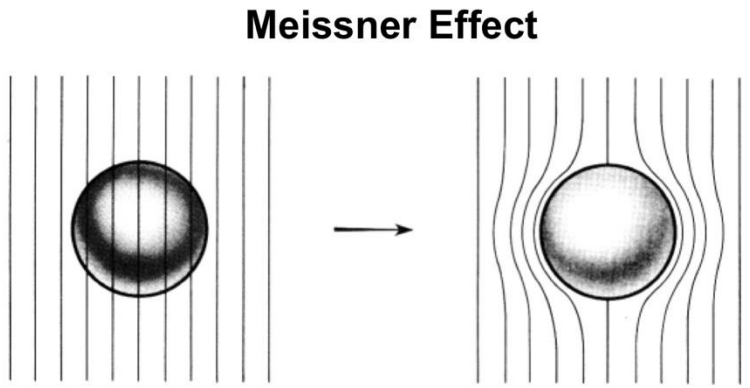
# Superconductivity Review

Requirements for a superconductor:

(1) ZERO electrical resistance below its transition temperature, $T_c$
- If injected with DC current, a superconducting loop will have a persistent electric current *forever* → there is no power loss

(2) Perfect diamagnetism
- The superconductor expels all magnetic fields (Meissner Effect)
- More than simple a "perfect conductor" which would preserve trapped fields with screening currents



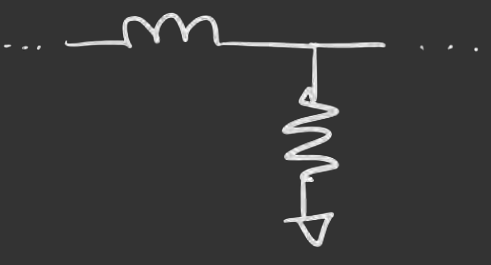



# Superconductor Constitutive Laws

## London's First Equation:

$$\vec{E} = \frac{d}{dt}(\Lambda \vec{J})$$

replaces Ohm's law $\hookrightarrow \Lambda \equiv \frac{m^*}{n^* q^{*2}}$



normal wire

Superconductor

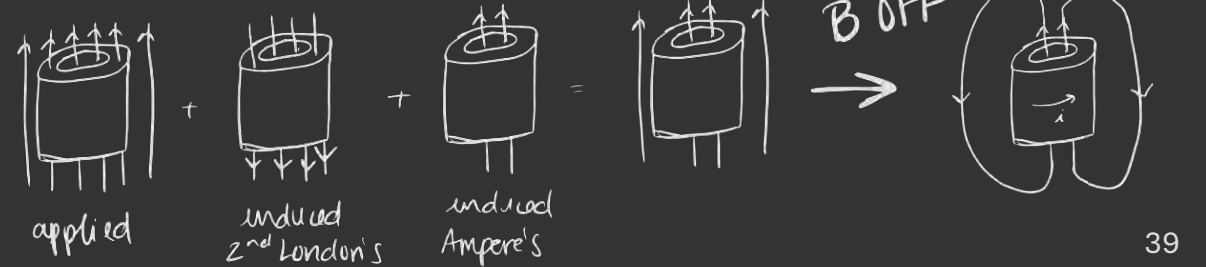
$\left(V = L\frac{di}{dt}\right)$

## London's Second Equation:

$$\nabla \times (\Lambda \vec{J}) = -\vec{B}$$

enforces Meisner effect by demanding initial conditions on Faraday's Law $\left(\nabla \times \vec{E} = -\frac{\partial B}{\partial t}\right)$

$\vec{B}$ ON, $T \rightarrow < T_c$



# Macroscopic Quantum Model

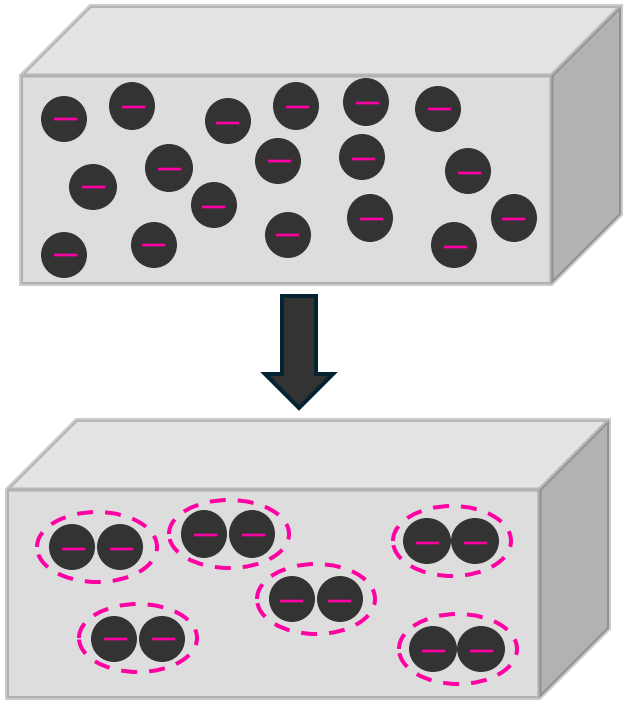
The whole superconducting material can be described by a macroscopic quantum wave function comprised of the superposition of cooper pairs

→ Phase coherence at a macroscopic scale!

$$\Psi(\mathbf{r},t) = \sqrt{n^\star(\mathbf{r},t)}\, e^{i\theta(\mathbf{r},t)}$$

$$i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t) = \frac{1}{2m^\star}\left(\frac{\hbar}{i}\nabla - q^\star\mathbf{A}(\mathbf{r},t)\right)^2\Psi(\mathbf{r},t) + q^\star\phi(\mathbf{r},t)\,\Psi(\mathbf{r},t)$$

$n^*$ -- local density of cooper pairs
$q^*$ -- charge of cooper pair
$m^*$ -- mass of cooper pair



$T > T_c$
Normal electrons

$T < T_c$
Superconducting cooper pairs

Figure adapted from W. Oliver, "6.645: Physics and Engineering of Superconducting Qubits" Course, Lecture 2 (Feb 2023)

# Flux Quantization

the quantum wavefunction can derive superconducting current density:

$$\vec{J}_s = -\frac{1}{\Lambda}\left(\frac{\hbar}{2e}\nabla\theta + \vec{A}\right)$$

then the flux is

$$\Phi = \int \vec{B}\cdot d\hat{s} = \oint_c \vec{A}\cdot d\vec{l}$$

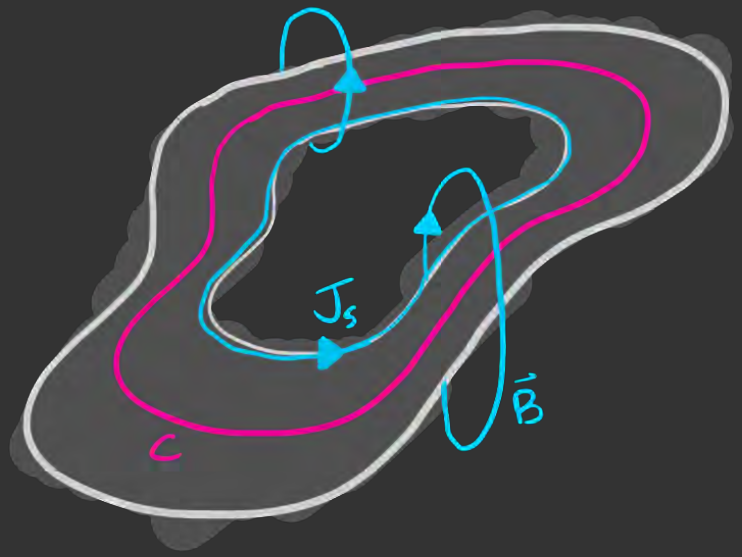
$$\Phi = \oint_c \left(\Lambda \vec{J}_s + \frac{\hbar}{2e}\nabla\theta\right)\cdot d\vec{l} = \frac{\hbar}{2e}\Delta\theta = \frac{h}{2e}n \longrightarrow \Phi = n\Phi_0$$

$\Delta\theta \rightarrow \frac{n}{2\pi}$

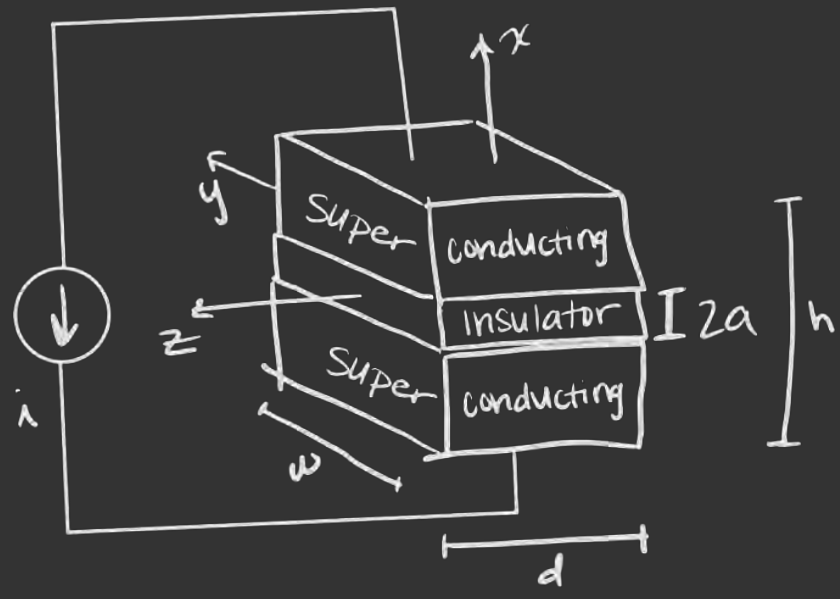
$\vec{J}_s = 0$ for contour deep below surface

$\Phi_0 = 2.07 \times 10^{-15}$ Wb

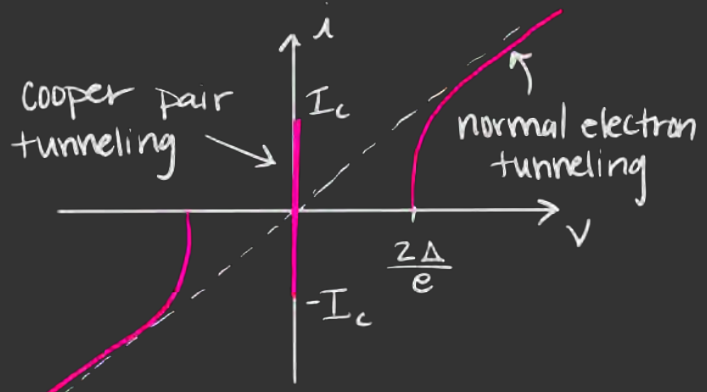
single flux quantum



# Josephson Tunneling




$$\vec{J}_s = \frac{q^*}{m^*} \mathrm{Re}\left\{ \psi^* \frac{\hbar}{i} \nabla \psi \right\}$$

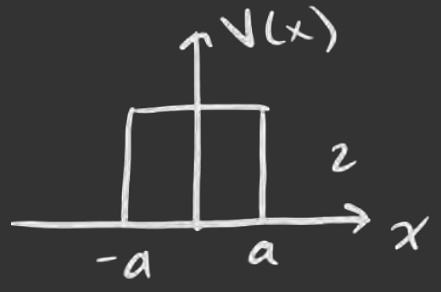


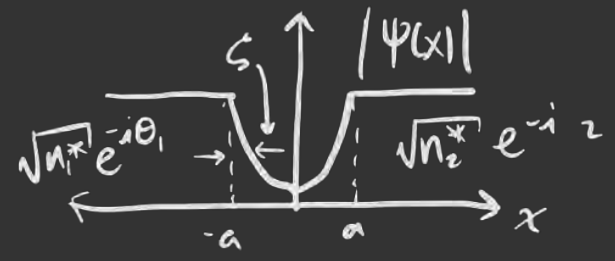

current-phase

$$J_s(y,z,t) = J_c(y,z) \sin\left(\theta_1(y,z,t) - \theta_2(y,z,t)\right)$$

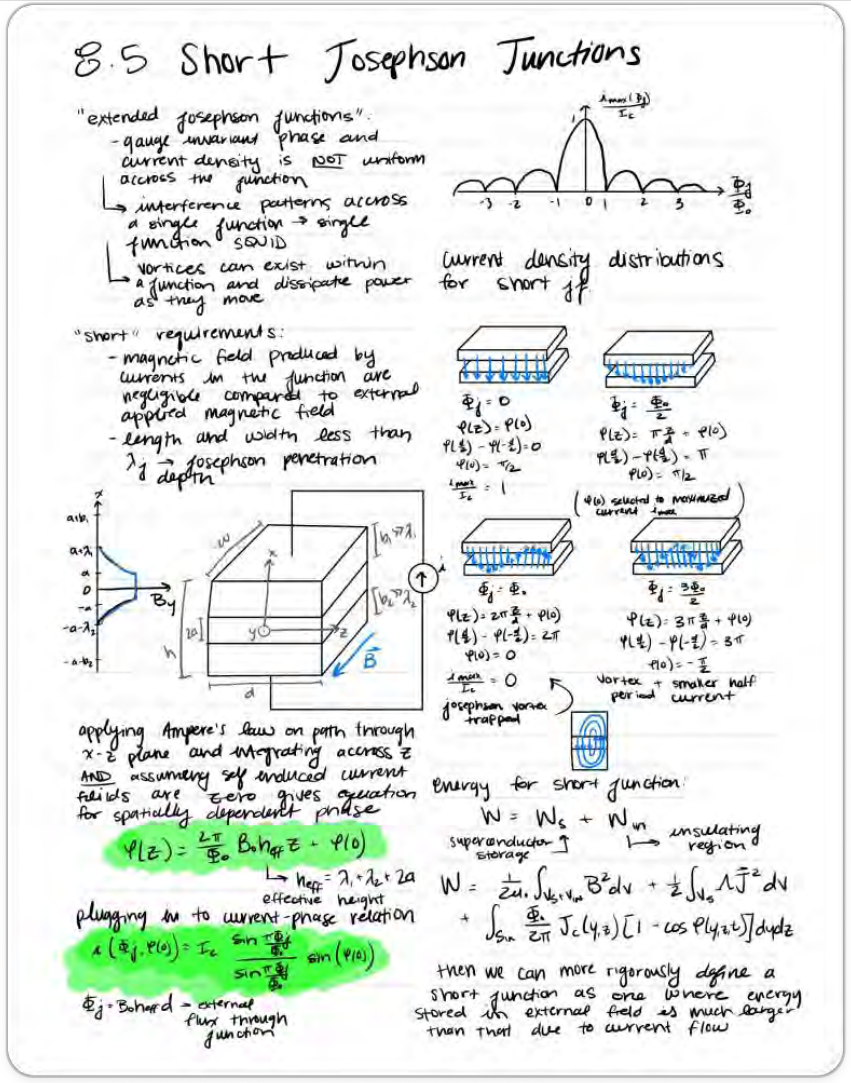
voltage-phase

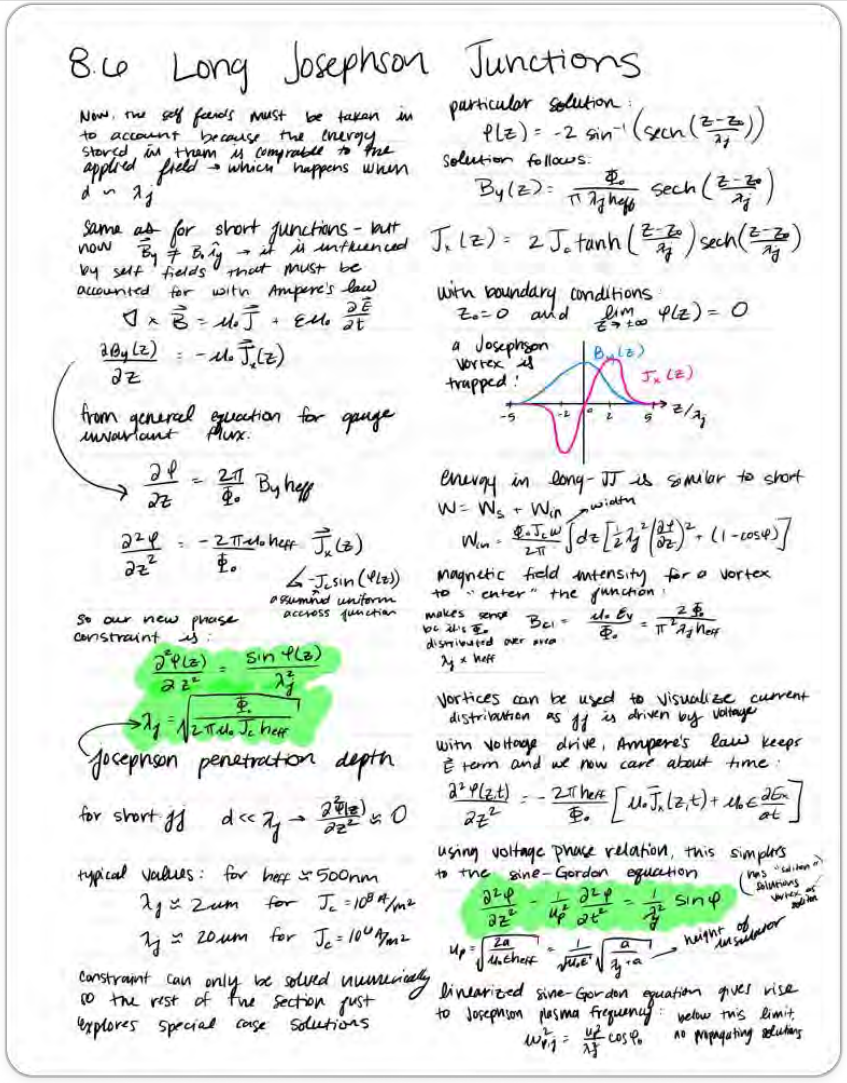
$$\frac{\partial \varphi(y,z,t)}{\partial t} = \frac{2\pi}{\Phi_0} \int_1^2 \vec{E}(\vec{r},t) \cdot d\vec{l}$$



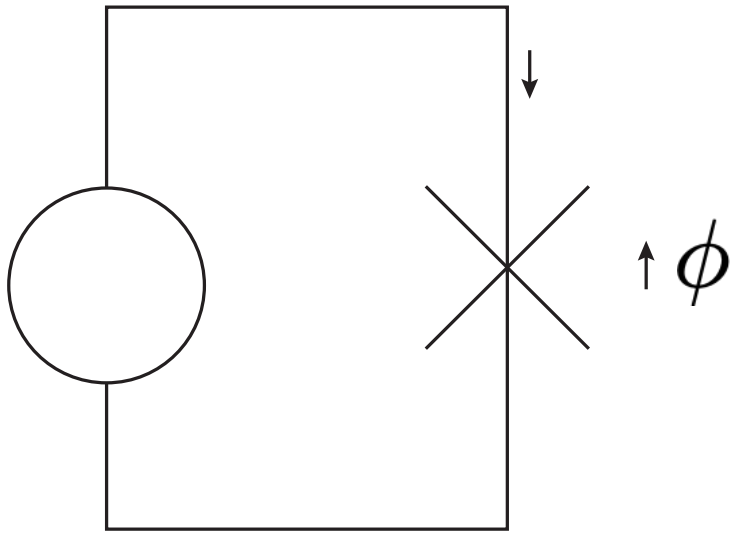



# Extended Josephson junction model



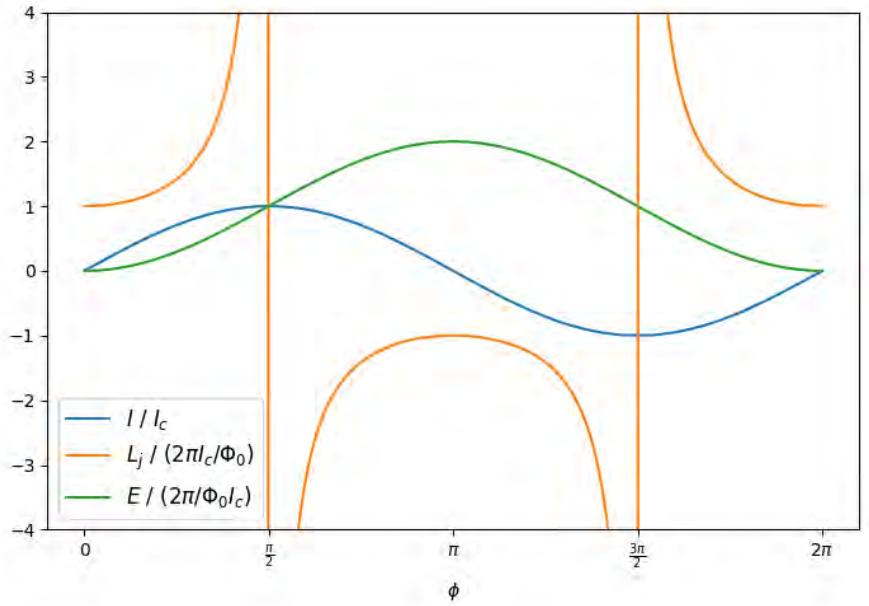
# Lumped Element JJ Model




JJ current and voltage relations:

$$i = I_c \sin\phi$$

$$v(t) = \frac{\Phi_0}{2\pi}\frac{d\phi}{dt}$$




Ideal junction can be thought of as a non-linear phase dependent inductor:

$$L_j = \frac{\Phi_0}{2\pi I_c \cos\phi}$$

Ideal energy dissipation:

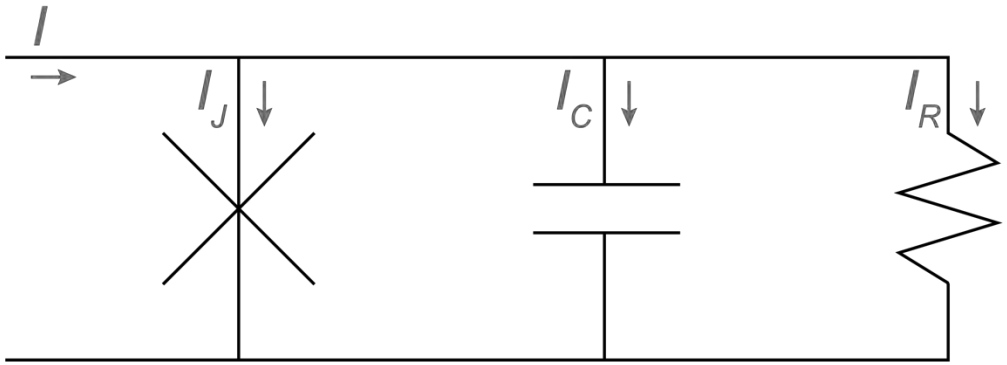
$$E_{JJ} = \int IV\,dt = \int (I_c \sin\phi)\left(\frac{\hbar}{2e}\frac{\partial\phi}{\partial t}\right)dt$$

$$= \frac{\Phi_0}{2\pi} I_c \int_0^{\phi} \sin\phi'\,d\phi'$$

$$= \frac{\Phi_0}{2\pi} I_c (1 - \cos\phi)$$

# JJ as a Circuit Element

## Resistively and Capacitively Shunted Junction (RCSJ) model




$$T_a = mgl\sin\phi + \eta\dot{\phi} + \mu\ddot{\phi}$$

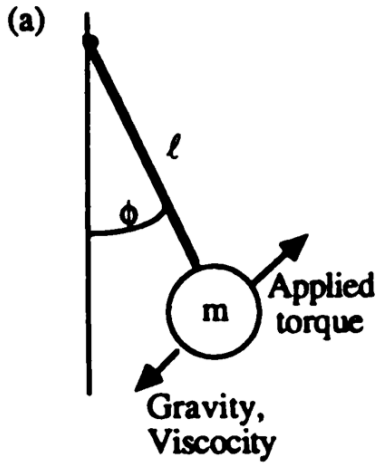
- $T_a$: Applied torque
- $\eta\dot{\phi}$: viscosity
- $\mu\ddot{\phi}$: Moment of inertia

Analogous to a driven pendulum, with a washboard potential

$$I = I_J + I_R + I_C$$

$$I = I_c\sin\phi + \frac{1}{R}\frac{\Phi_0}{2\pi}\dot{\phi} + C\frac{\Phi_0}{2\pi}\ddot{\phi}$$

$$\rightarrow U = \frac{\Phi_0}{2\pi}I_c\left(1 - \frac{I}{I_c}\phi - \cos\phi\right)$$





Kadin, A. M., 1999, Introduction to Superconducting Circuits (Wiley, New York)

# Superconducting Circuit Basics

flux: $\phi = \int_{-\infty}^{t} v\, dt$

charge: $Q = \int_{-\infty}^{t} i\, dt$

because of flux quantization:

phase: $\varphi = \frac{2\pi\phi}{\Phi_0}$

all branch elements can be described by constitutive relations:

capacitive:

$v_b = f(Q_b)$

inductive:

$i_b = g(\phi_b)$

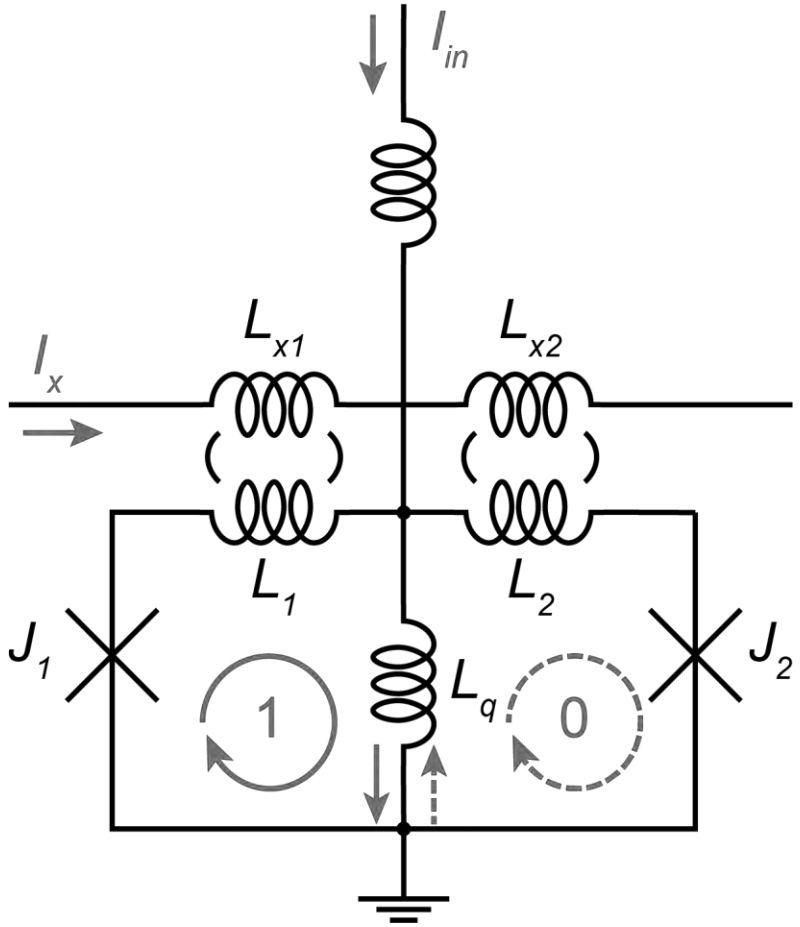
e.g. for inductor:

$i_b = \frac{\phi_b}{L}$

or JJ:

$i_b = I_c \sin\left(\frac{2\pi}{\Phi_0}\phi_b\right)$

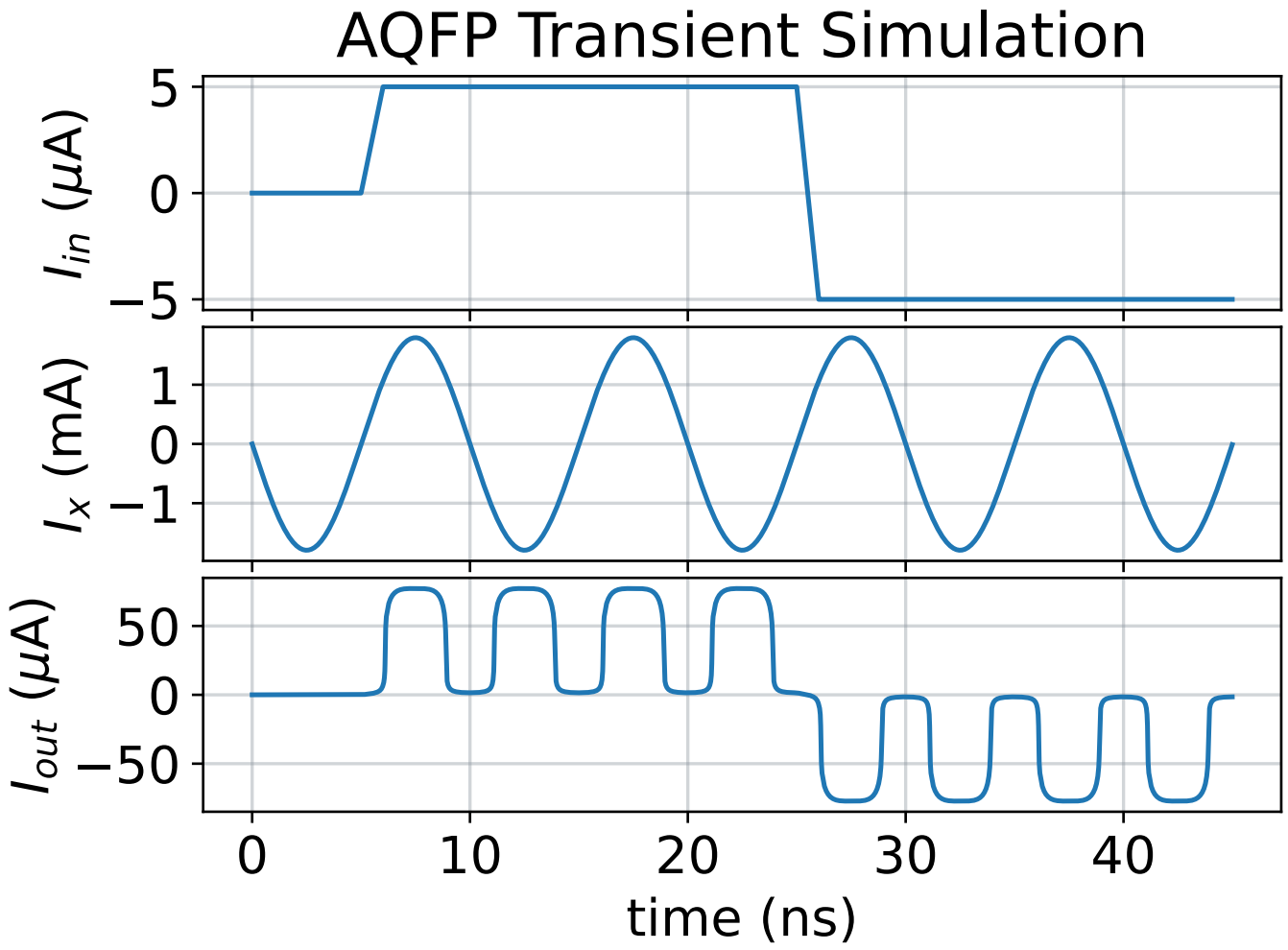
# The Adiabatic Quantum Flux Parametron




Depending on the polarity of a small input current, $I_{\mathrm{in}}$, a single magnetic flux quantum will be trapped in either the left or right loop, corresponding to a digital 1 or 0 value

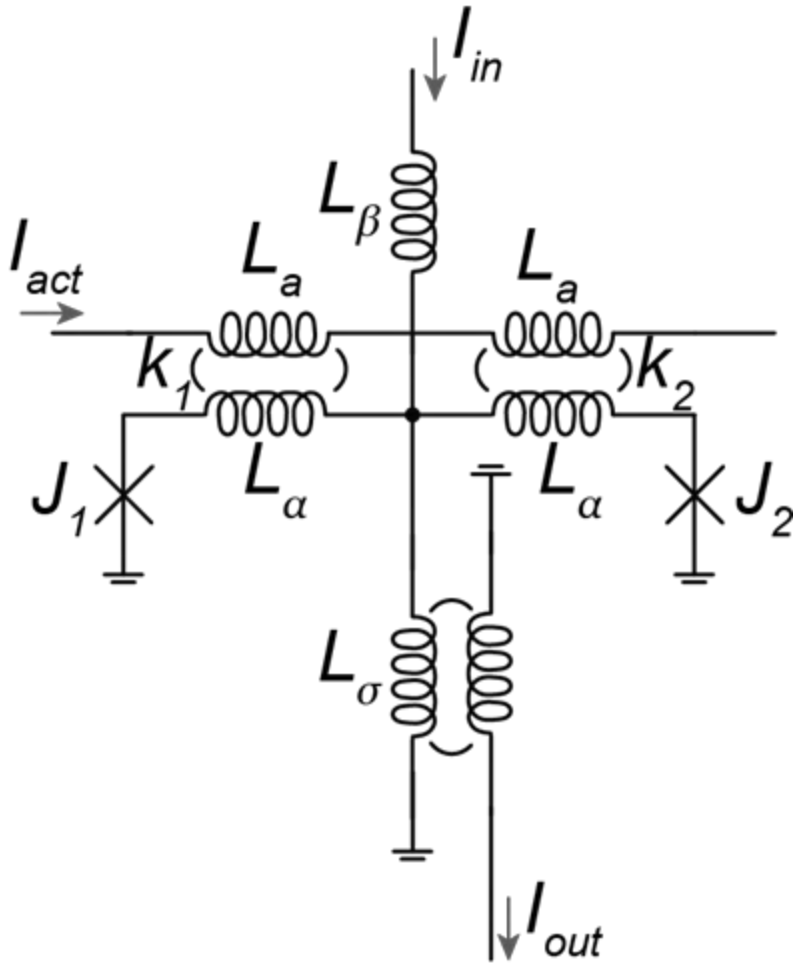
The fluxon will only be trapped if the AQFP is activated with an excitation signal, $I_{\mathrm{x}}$, which forces the Josephson junctions into their low-impedance state

# The Adiabatic Quantum Flux Parametron




Citations

# AQFP Circuit Analysis: Node Method



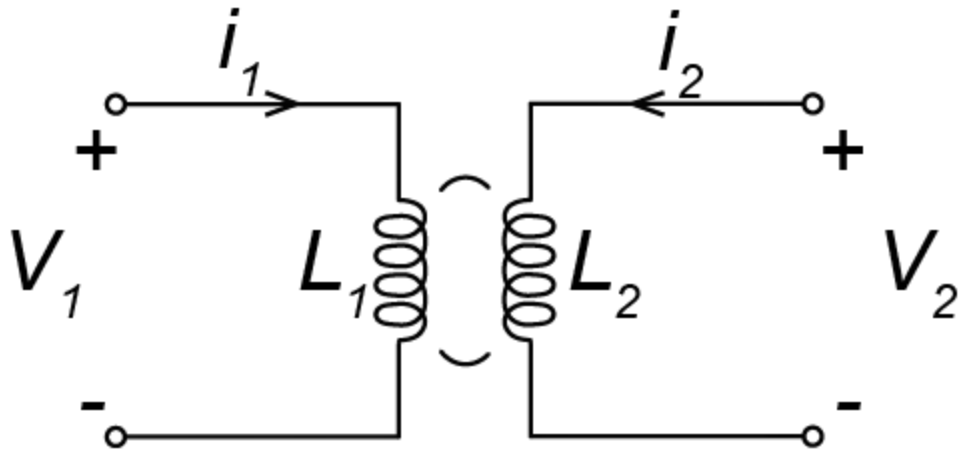

GOAL: There are two dynamical degrees of freedom within the AQFP circuit (one for each JJ) and we want to find the operating point of the circuit given some input and activation signal, so we are solving for the output signal.

1. Define circuit topology with Kirchoff's current laws
2. Then express KCLs in terms of branch variable dynamics to constrain the system to a particular operating point

But before starting, we need to understand how to handle the inductive coupling transformers.

Apologies for the change up in variable names . . .

# AQFP Circuit Analysis: Transformers




*M*: mutual inductance

$$emf_1 = M\frac{di_2}{dt} \rightarrow \Phi_1 = Mi_2$$

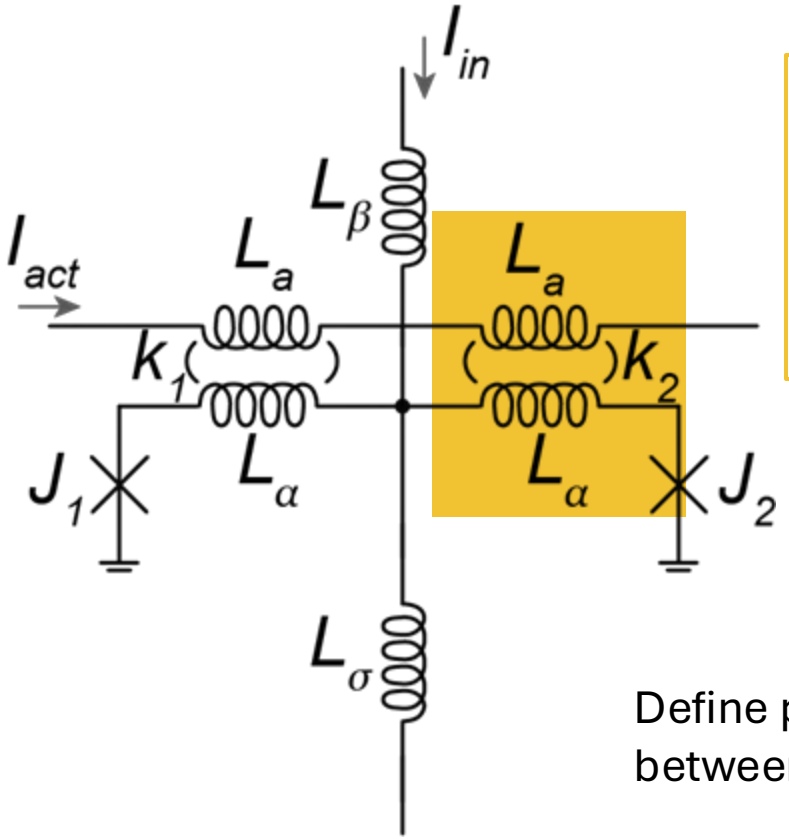
$$\begin{bmatrix}\Phi_1\\ \Phi_2\end{bmatrix} = \begin{pmatrix}L_1 & M\\ M & L_2\end{pmatrix}\begin{bmatrix}i_1\\ i_2\end{bmatrix}$$

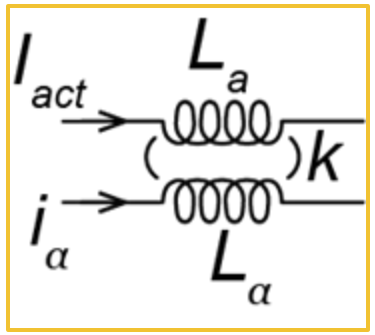
*k*: coupling constant $M = k\sqrt{L_1 L_2}$

*n*: inductor ratio $n^2 = L_2/L_1$

$$\rightarrow \begin{bmatrix}\Phi_1\\ \Phi_2\end{bmatrix} = L_1\begin{pmatrix}1 & nk\\ nk & n^2\end{pmatrix}\begin{bmatrix}i_1\\ i_2\end{bmatrix}$$

# AQFP Circuit Analysis: Transformers




$$\begin{bmatrix}\Phi_a \\ \Phi_\alpha\end{bmatrix} = L_a \begin{pmatrix}1 & nk \\ nk & n^2\end{pmatrix} \begin{bmatrix}i_a \\ i_\alpha\end{bmatrix}$$

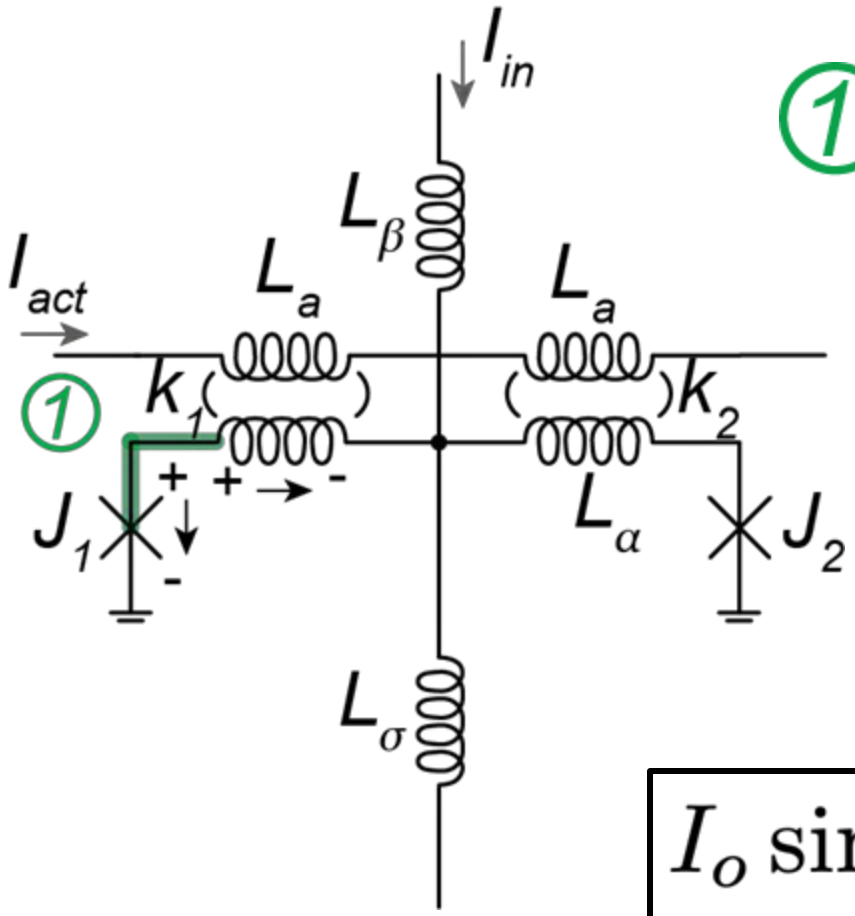
$$i_\alpha = \frac{1}{L_\alpha(1-k^2)}(\phi_\alpha - \phi_a nk)\bar{\Phi}_o$$

Flux has been replaced by phase: $\Phi = \frac{\Phi_o}{2\pi}\phi = \bar{\Phi}_o\phi$

Define phase angle that characterizes the relationship between activation inductor and loop inductor: $\alpha \equiv \frac{1}{1-k^2}(\phi_a - \frac{k}{n}\phi_\alpha)$

$$\boxed{i_\alpha = \frac{\bar{\Phi}_o}{L_\alpha}(\phi_\alpha - nk\alpha)}$$

# QFP Circuit Analysis: Node Method




1 $-i_{j1} - i_{\alpha 1} = 0$

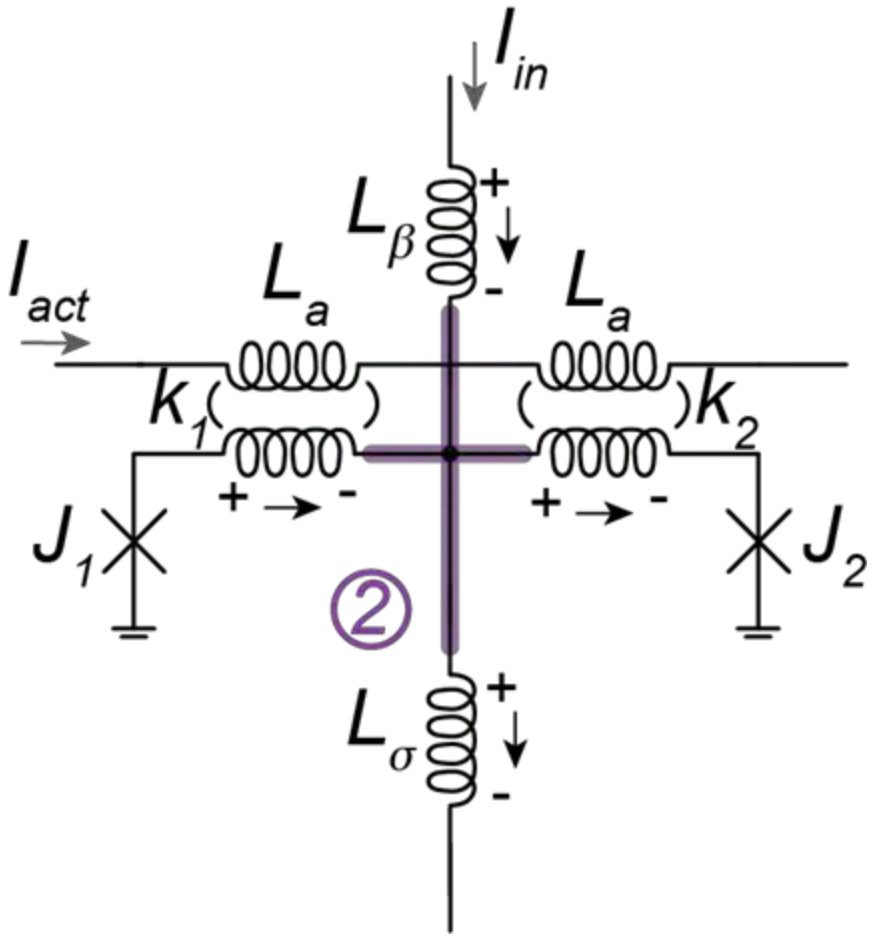
$$i_{\alpha 1} = \frac{\bar{\Phi}_o}{L_\alpha}(\phi_{\alpha 1} - nk_1\alpha)$$

$$\phi_{\alpha 1} = \phi_{j1} - \phi_\sigma$$

Define inverse inductance: $\ell = \frac{\bar{\Phi}_o}{L}$

$$I_o \sin\phi_{j1} + \ell_{\alpha 1}(\phi_{j1} - \phi_\sigma - nk_1\alpha) = 0$$

# QFP Circuit Analysis: Node Method




② $i_\beta + i_{\alpha 1} - i_{\alpha 2} - i_\sigma = 0$

$$i_{\alpha 2} = \frac{\bar{\Phi}_o}{L_\alpha}(\phi_{\alpha 2} - nk_2\alpha)$$

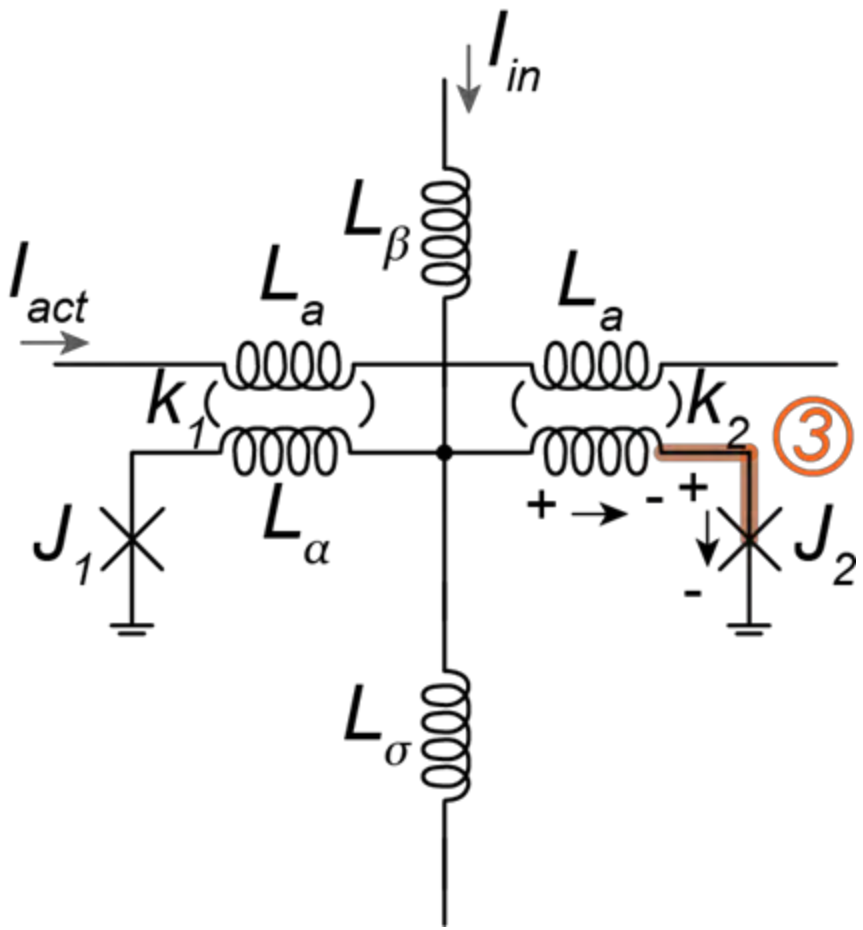
$$\phi_{\alpha 2} = \phi_\sigma - \phi_{j2}$$

$$\ell_\beta(\phi_\beta - \phi_\sigma) + \ell_{\alpha 1}(\phi_{j1} - \phi_\sigma - nk_1\alpha) - \ell_{\alpha 2}(\phi_\sigma - \phi_{j2} - nk_2\alpha) - \ell_\sigma\phi_\sigma = 0$$

assume $\ell_{\alpha 1} = \ell_{\alpha 2}$

$$\ell_\alpha(\phi_{j1} + \phi_{j2} - 2\phi_\sigma + n(k_2 - k_1)\alpha) + \ell_\beta(\phi_\beta - \phi_\sigma) - \ell_\sigma\phi_\sigma = 0$$

# QFP Circuit Analysis: Node Method

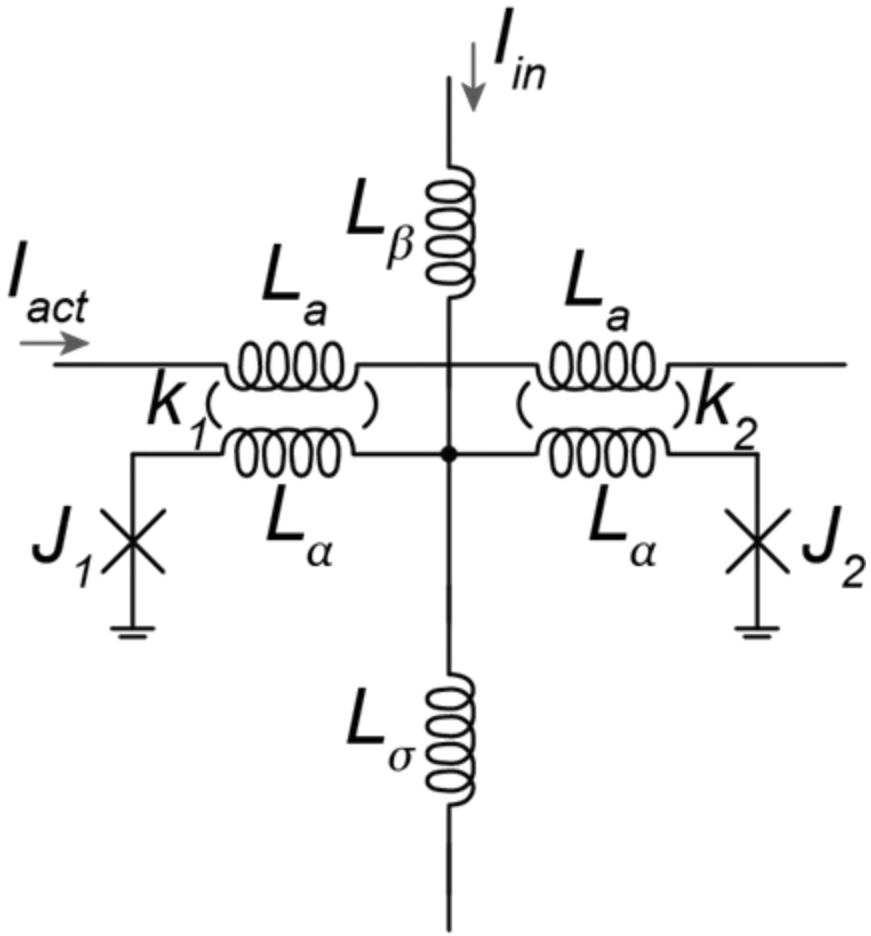



③ $i_{\alpha 2} - i_{j2} = 0$

$$\ell_{\alpha 2}(\phi_\sigma - \phi_{j2} - nk_2\alpha) - I_o \sin\phi_{j2} = 0$$

$$I_o \sin\phi_{j2} + \ell_{\alpha 2}(\phi_{j2} - \phi_\sigma + nk_2\alpha) = 0$$

# QFP Circuit Analysis: Circuit Constraints




KCL summary :

① $I_o \sin\phi_{j1} + \ell_{\alpha 1}(\phi_{j1} - \phi_\sigma - nk_1\alpha) = 0$

② $\ell_\alpha(\phi_{j1}+\phi_{j2} - 2\phi_\sigma + n(k_2 - k_1)\alpha) + \ell_\beta(\phi_\beta - \phi_\sigma) - \ell_\sigma\phi_\sigma = 0$

③ $I_o \sin\phi_{j2} + \ell_{\alpha 2}(\phi_{j2} - \phi_\sigma + nk_2\alpha) = 0$

# QFP Circuit Analysis: Circuit Constraints

We have an extra equation for only two degrees of freedom, so we can express equations in terms of sum and difference across both JJs:

$$\phi_+ = \frac{1}{2}(\phi_{j1} + \phi_{j2}) \qquad \phi_- = \frac{1}{2}(\phi_{j1} - \phi_{j2})$$

① + ③

$$2I_o \sin\phi_+ \cos\phi_- + \ell_\alpha(2\phi_+ - 2\phi_\sigma + n(k_2 - k_1)\alpha) = 0$$

① - ③

$$2I_o \cos\phi_+ \sin\phi_- + \ell_\alpha(2\phi_- + n(k_2 + k_1)\alpha) = 0$$

# QFP Circuit Analysis: Circuit Constraints

Use (2) to solve for $\phi_\sigma$:

$$\phi_\sigma = \frac{\ell_\beta \phi_\beta + \ell_\alpha(2\phi_+ + n(k_2 - k_1)\alpha)}{2\ell_\alpha + \ell_\sigma + \ell_\beta}$$

① + ③

$$2I_o \sin\phi_+ \cos\phi_- + \ell_+ \left(2\phi_+ + n(k_2 - k_1)\alpha + \frac{2\ell_\beta}{\ell_\sigma + \ell_\beta}\phi_\beta\right) = 0$$

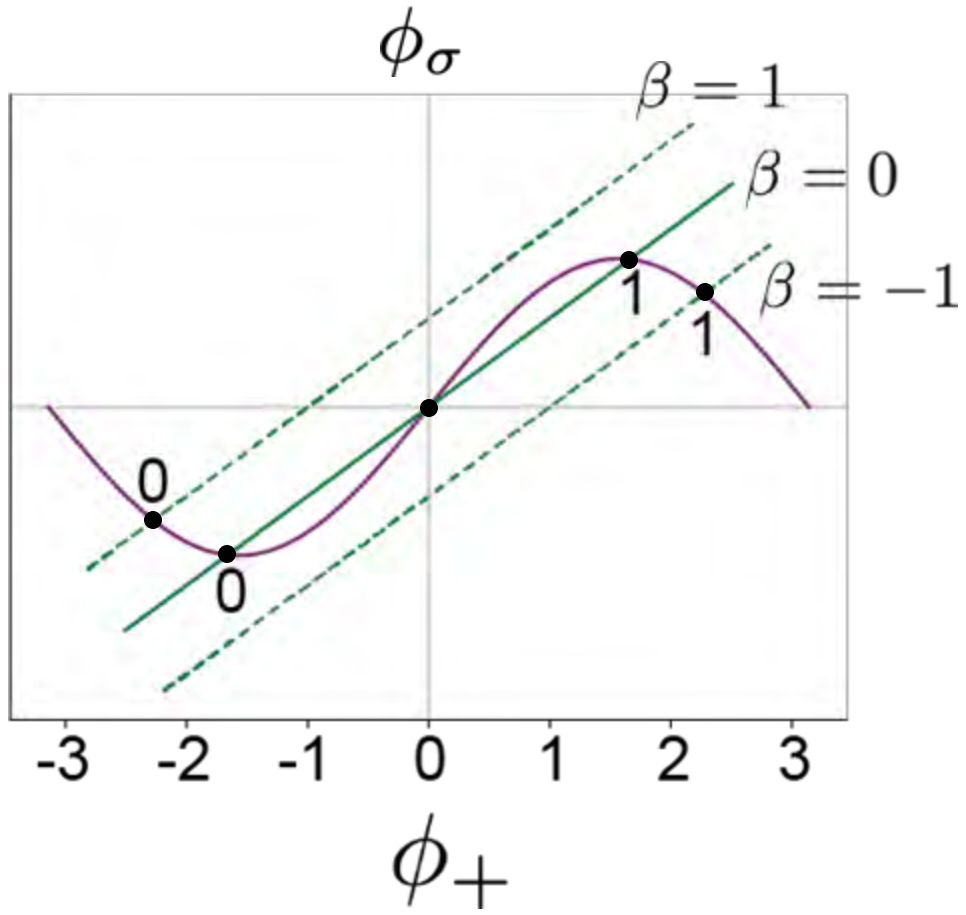
① - ③

$$\ell_+ = \frac{\ell_\alpha(\ell_\sigma + \ell_\beta)}{2\ell_\alpha + \ell_\sigma + \ell_\beta}$$

$$2I_o \cos\phi_+ \sin\phi_- + \ell_\alpha(2\phi_- + n(k_2 + k_1)\alpha) = 0$$

Solved when $\alpha = \pm\frac{2\pi}{n(k_2 + k_1)}$ and $\phi_- = \mp\pi$ → which corresponds to double well potential

# QFP Circuit Analysis: Load Line



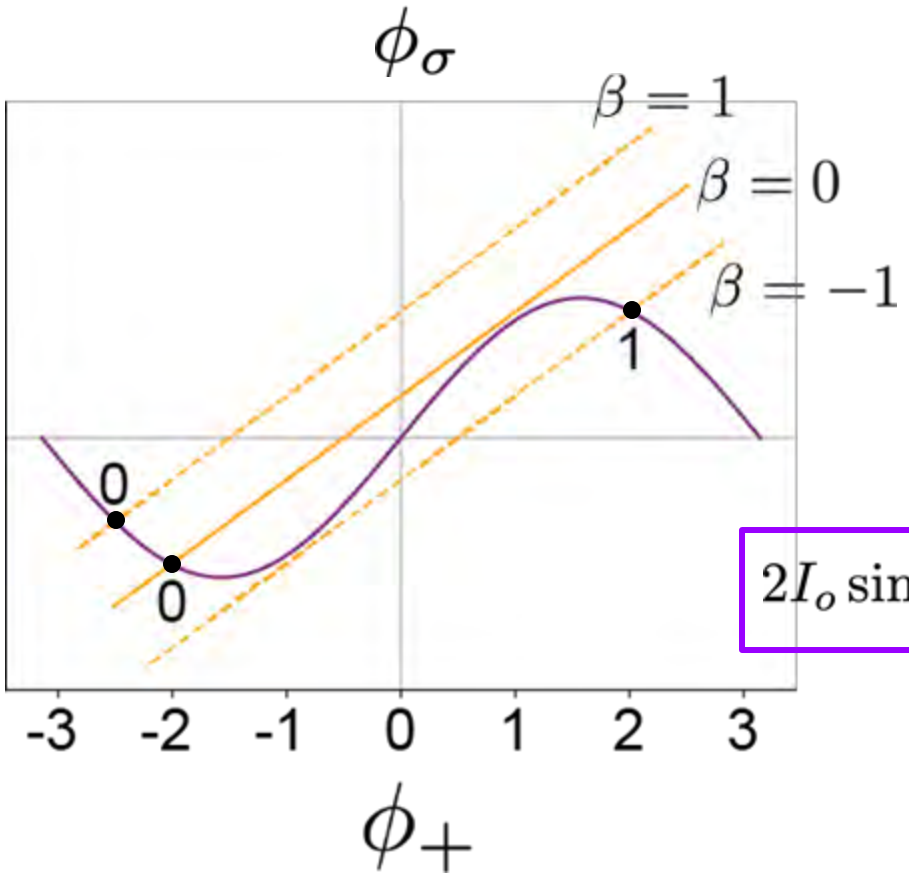

$$k_1 = k_2 \rightarrow \mathbf{BUF}$$

No $\alpha$ dependence on logic state

$$2I_o \sin\phi_+ + \ell_+ \left(2\phi_+ + \frac{2\ell_\beta}{\ell_\sigma + \ell_\beta}\phi_\beta\right) = 0$$

Citations

# QFP Circuit Analysis: Load Line




$$k_1 > k_2$$

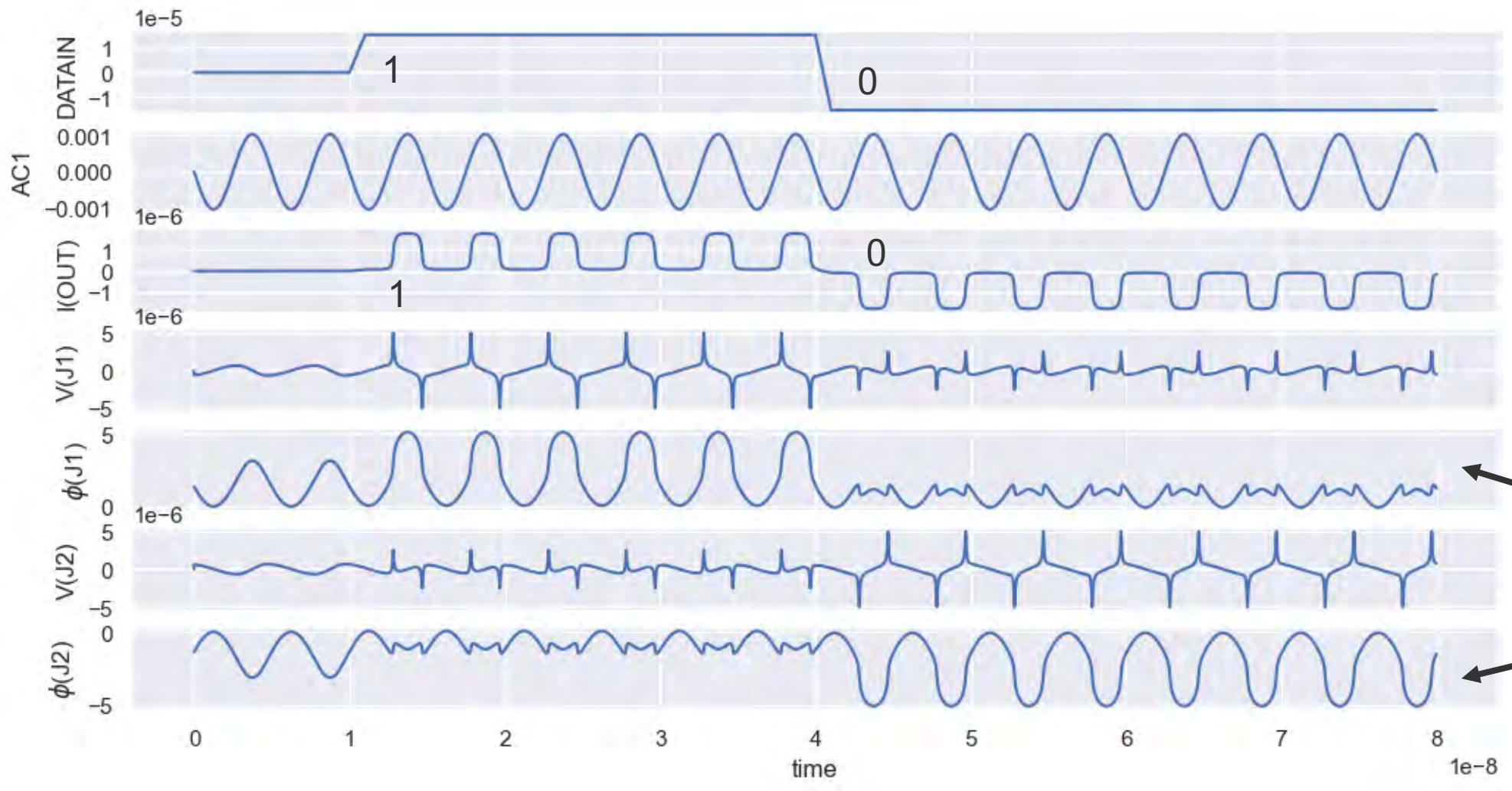
$\alpha$ dependence on logic state behaves as a current source in one loop, causing the null input to favor one side

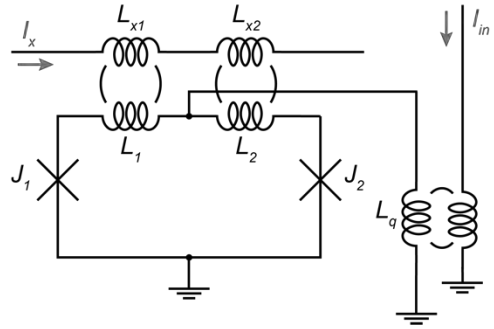
$$2I_o \sin\phi_+ + \ell_+ \left(2\phi_+ + n(k_2 - k_1)\alpha + \frac{2\ell_\beta}{\ell_\sigma + \ell_\beta}\phi_\beta\right) = 0$$

Add a guide on variables

Citations

# Simulated Transient Behavior of AQFP

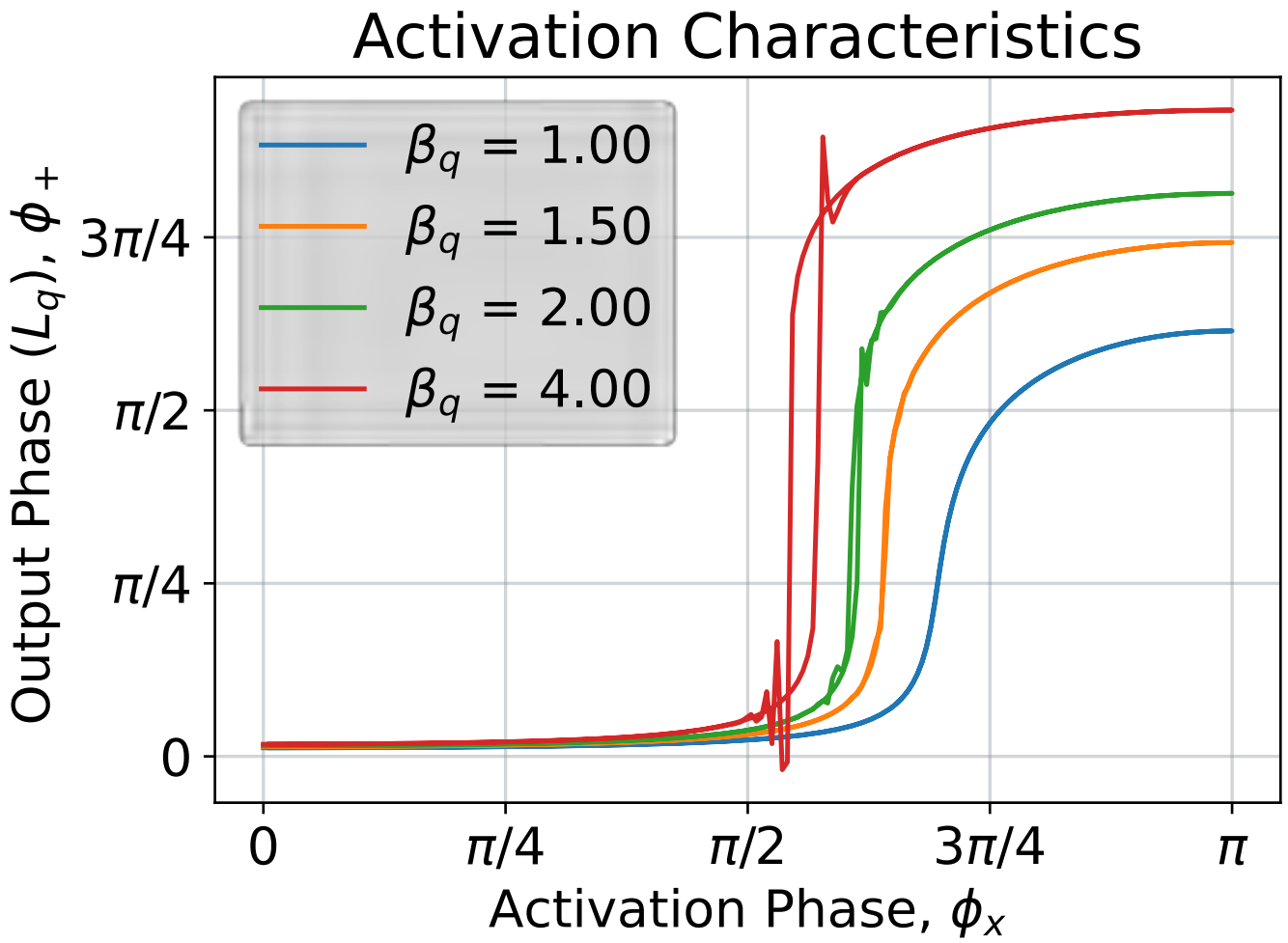



200MHz output signal

$2\pi$ phase shift between both junctions for each data pulse

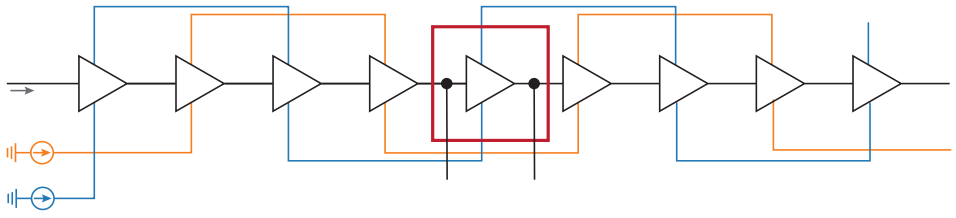
# Activation Characteristics

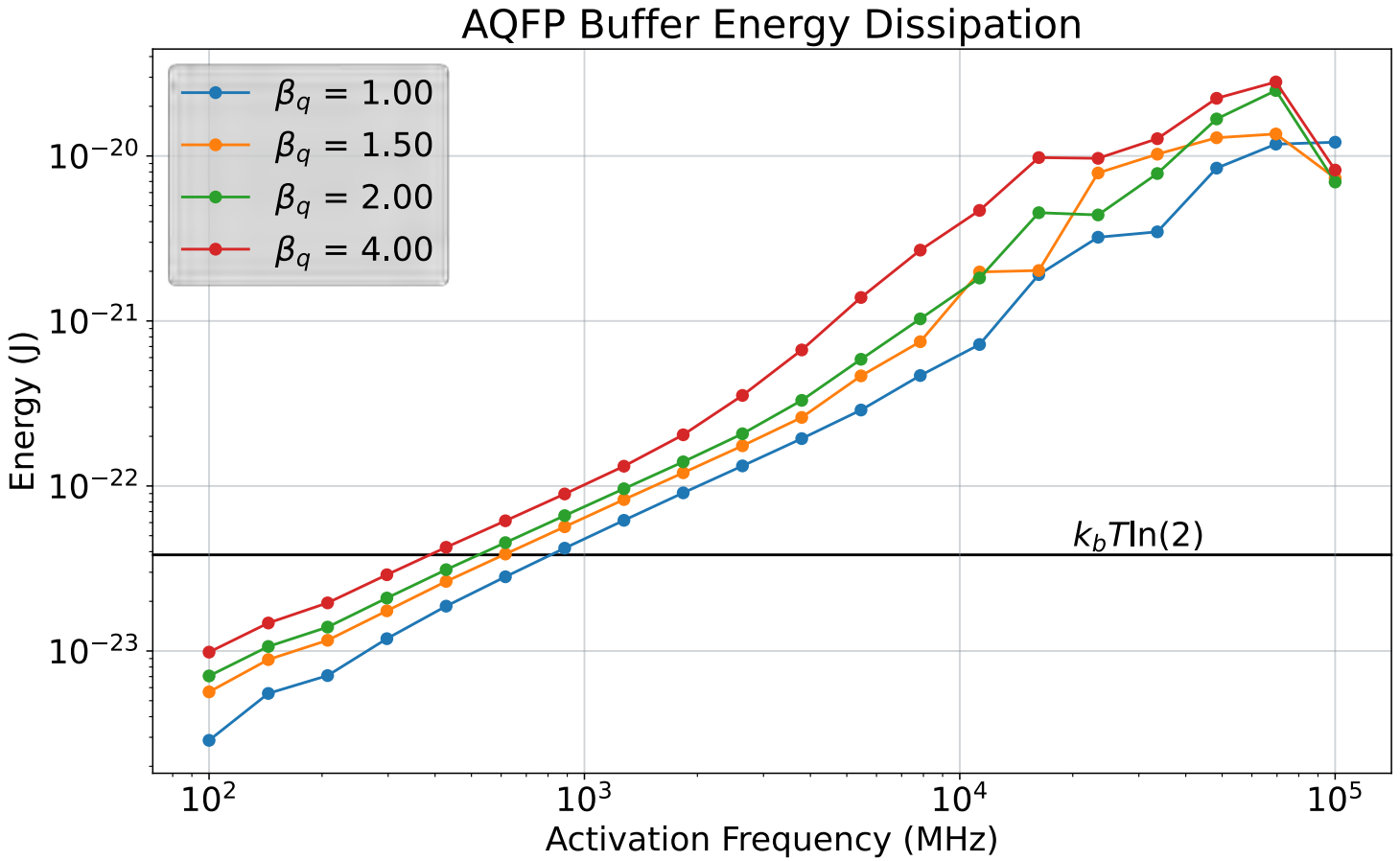



$\beta_q \to \ell_\sigma$

# SPICE Extracted Switching Energy

$$E_{\text{diss}} = \int_0^{1/f} I_{x1} \cdot (V_{\text{out}} - V_{\text{in}}) dx$$





# AQFP Energy Dissipation

Energy required for a single QFP switching, phase across the junction, $\phi$, increases by $2\pi$ and the energy dissipated can be described with the voltage across the JJ resistor

$$E_x = \int_0^{\tau_x} \frac{{V_1}^2}{R} dt = \frac{1}{R}\left(\frac{\Phi_0}{2\pi}\right)^2 \int_0^{\tau_x} \left(\frac{d\phi_1}{dt}\right)^2 dt$$

$$\approx \frac{1}{R}\left(\frac{\Phi_0}{\tau_x}\right)^2 \int_0^{\tau_x} dt = \frac{{\Phi_0}^2}{R\tau_x}$$

Then for excitation and reset to single well, the total switching energy is $2E_x$

$$E_{sw} = E_x + E_r \approx \frac{2{\Phi_0}^2}{R\tau_x}$$

Alternative phase-evolution derivation:

$$E_{sw} \approx 2I_c\Phi_0\frac{\tau_j}{\tau_x}$$

- $\tau_j$: JJ characteristic time constant, i.e. $L/R$ time constant
- 2: 2 switches of the JJ for full cycle (OFF > ON > OFF)
- $I_c\Phi_0$: JJ energy ~$E_J$
- $\tau_x$: Excitation time constant, $\frac{1}{2}f$ for sinusoidal excitation frequency, $f$

$$\tau_j \approx 2\pi\frac{L_j}{R} = \frac{\Phi_0}{I_cR} = \sqrt{\frac{2\pi\Phi_0 C_s}{\beta_c J_c}}$$

- $C_s$: Junction capacitance per area
- $J_c$: Critical current density

McCumber parameter:

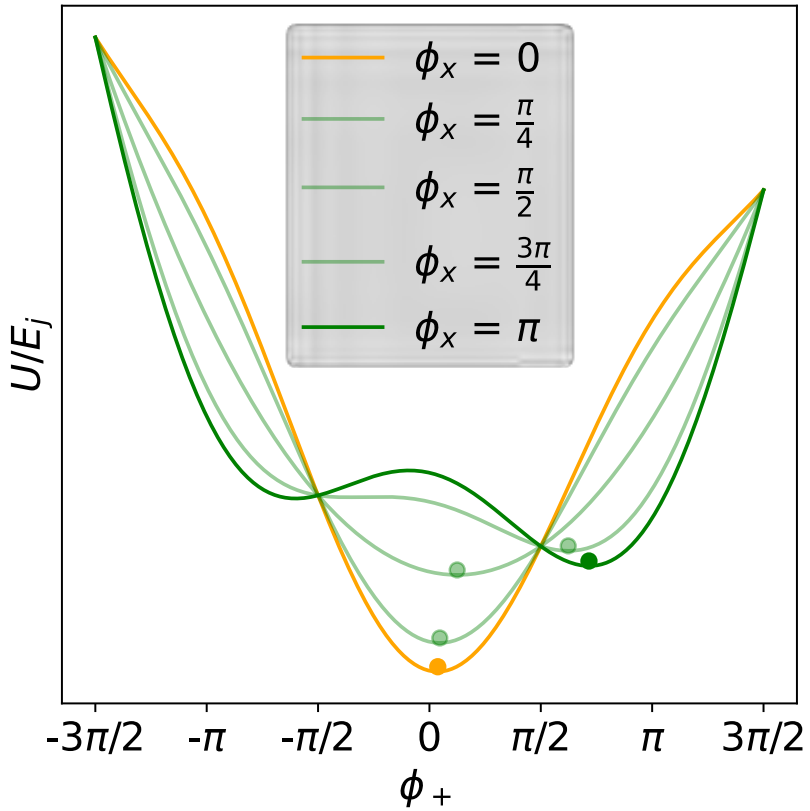
$$\beta_c = Q^2 = \frac{2\pi R^2 C I_c}{\Phi_0}$$

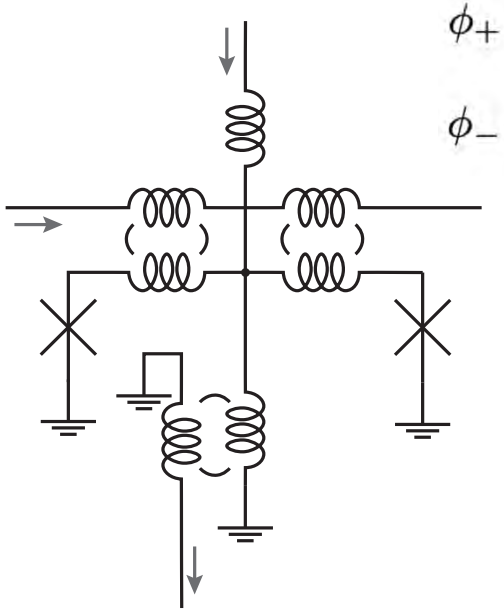
# More Fundamental Energy Analysis?

- How accurate is the classical SPICE $\int (I \cdot V) dt$ method for determining the energy dissipation limits of the AQFP ?
- Could it actually operate in a reversible adiabatic regime?
- Circuit quantization, a method for linking a physical circuit to its corresponding quantum Hamiltonian, could provide a path to a more rigorous definition for the energy dissipated when an AQFP undergoes a classical bit-flip
- Additionally, circuit quantization provides a formalism for deriving the potential energy for the circuit, providing another perspective on AQFP circuit behavior

# AQFP Potential Energy

$$\frac{U_{2D}}{E_j} = \frac{(\phi_+ - \phi_{in})^2}{2\beta_q} - 2\cos\phi_x \cos\phi_+$$



$$E_j = \frac{I_c \Phi_0}{2\pi} \rightarrow \text{energy normalization factor}$$

$$\beta_q = \frac{2\pi L_q I_c}{\Phi_0} \rightarrow \text{normalized inductance value for } L_q$$

$$\beta_L = \frac{2\pi L_L I_c}{\Phi_0} \rightarrow \text{normalized inductance value for } L_L$$

$$\phi_+ = \frac{1}{2}(\phi_1 + \phi_2) \rightarrow \text{sum mode of JJ phases}$$

$$\phi_- = \frac{1}{2}(\phi_1 - \phi_2) \rightarrow \text{difference mode of JJ phases}$$

# QFP Analysis: Circuit Quantization

Most circuit quantization methods assume that all external flux relevant to the circuit is not time dependent → NOT true for the AQFP, which relies on time-dependent external flux for its state change, so we will follow method from [1]

Circuit quantization workflow:

1. Write down Lagrangian in terms of branch fluxes ($\Phi_{b1}, \Phi_{b2}, \ldots, \Phi_{bn}$) and charges ($\dot{\Phi}_{b1}, \dot{\Phi}_{b2}, \ldots \dot{\Phi}_{bn}$), with $\boldsymbol{C}$ = diag($C_{b1}, C_{b2}, \ldots, C_{bn}$) and $\boldsymbol{L^{-1}}$= diag($L_{b1}^{-1}, L_{b2}^{-1}, \ldots, L_{bn}^{-1}$)

$$\mathcal{L} = \mathcal{L}_T - \mathcal{L}_V$$

Kinetic energy stored in capacitive components

Potential energy stored in inductive components

$$\mathcal{L} = \frac{1}{2}\dot{\Phi}_b^T C \dot{\Phi}_b - \frac{1}{2}\Phi_{b_L}^T L^{-1} \Phi_{b_L} + \sum_{i=1}^{n} \frac{\Phi_0}{2\pi} I_c \cos\left(\frac{2\pi}{\Phi_0}\Phi_{b_{Ji}}\right)$$

[1] X. You, J. A. Sauls, and J. Koch, "Circuit quantization in the presence of time-dependent external flux," *Phys. Rev. B*, (May 2019)

# QFP Analysis: Circuit Quantization

Circuit quantization workflow:

2. Next, we want to find the optimal coordinates to express the Lagrangian, i.e. irrotational degrees of freedom, $\tilde{\boldsymbol{\Phi}}$, and external flux factors, $\boldsymbol{\Phi_{ex}}$. So we impose two constraints through two matrix operations

Flux quantization:

$$\boldsymbol{\Phi_{ex}} = \boldsymbol{R\Phi_b}$$

$$R_{ij} = \begin{cases} 1 & \text{if } \Phi_{b,j} \text{ is enclosed in } \Phi_{ex,i} \text{ loop with same orientation} \\ -1 & \text{if } \Phi_{b,j} \text{ is enclosed in } \Phi_{ex,i} \text{ loop with opposite orientation} \\ 0 & \text{if } \Phi_{b,j} \text{ is not enclosed } \Phi_{ex,i} \text{ loop} \end{cases}$$

Kirchoff's voltage law:

$$\tilde{\boldsymbol{\Phi}} = \boldsymbol{M\Phi_b}$$

And $\boldsymbol{M}$ is further set by the irrotational constraint to ensure that result does not depend on $\dot{\Phi}_{ex}$, to match time-independent solutions $\longrightarrow$ $\boldsymbol{RC^{-1}M^T = 0}$

[1] X. You, J. A. Sauls, and J. Koch, "Circuit quantization in the presence of time-dependent external flux," *Phys. Rev. B*, (May 2019)

# QFP Analysis: Circuit Quantization

Circuit quantization workflow:

2. Then you can express the Lagrangian in terms of dynamical degrees of freedom with simple substitution $\mathbf{\Phi_b} = \mathbf{M_+^{-1}}\tilde{\mathbf{\Phi}}_+$ where

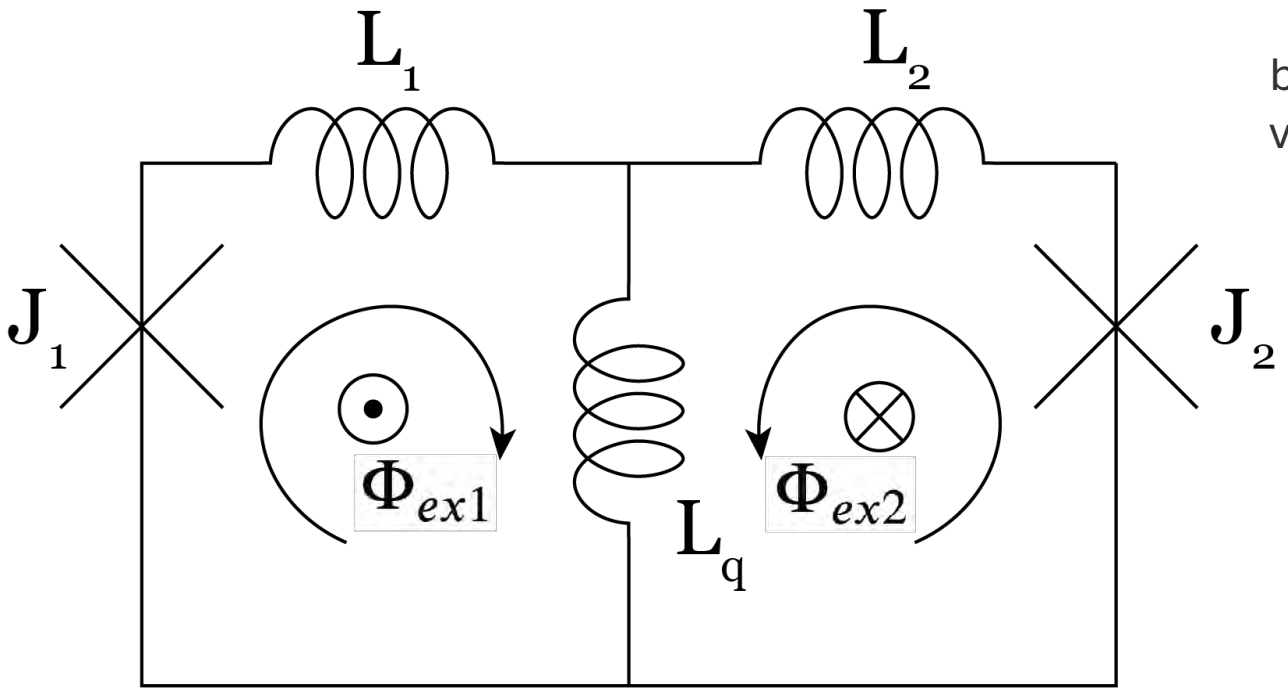
$$\tilde{\mathbf{\Phi}}_+ := \begin{pmatrix} \tilde{\mathbf{\Phi}} \\ \mathbf{\Phi}_x \end{pmatrix}$$

$$\boldsymbol{M}_+ := \begin{pmatrix} \boldsymbol{M} \\ \boldsymbol{R} \end{pmatrix}$$

Then it's just a Legendre transformation to derive the Hamiltonian from the Lagrangian

[1] X. You, J. A. Sauls, and J. Koch, "Circuit quantization in the presence of time-dependent external flux," *Phys. Rev. B*, (May 2019)

# QFP Analysis: Circuit Quantization




branch variables

$$\Phi_b = \begin{pmatrix} \Phi_{J_1} \\ \Phi_{J_2} \\ \Phi_{L_1} \\ \Phi_{L_2} \\ \Phi_{L_q} \end{pmatrix} \longrightarrow \quad \Phi_{ex} = \begin{pmatrix} \Phi_{ex1} \\ \Phi_{ex2} \end{pmatrix}$$

optimized variables

$$\tilde{\Phi} = \begin{pmatrix} \tilde{\Phi}_1 \\ \tilde{\Phi}_2 \\ \tilde{\Phi}_3 \end{pmatrix}$$

Flux quantization constraint

$$R = \begin{bmatrix} 1 & 0 & -1 & 1 & 0 \\ 0 & 1 & 1 & 0 & -1 \end{bmatrix}$$

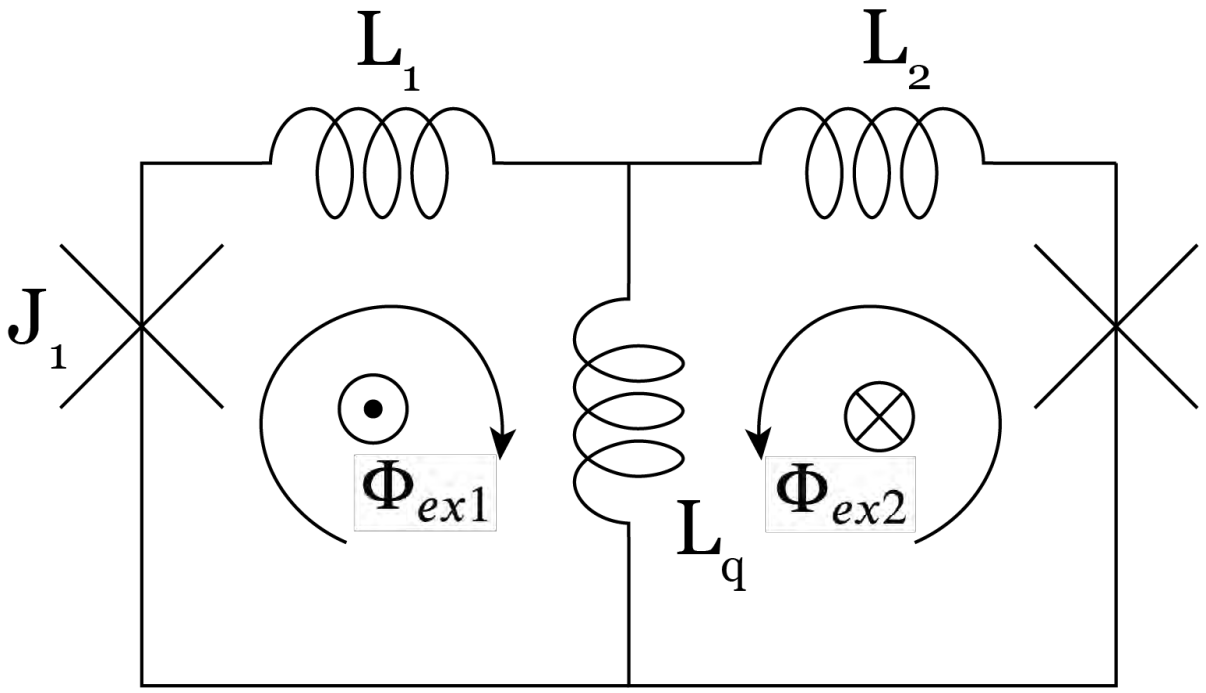
Irrotational constraint

$$CM_{i1} + M_{i3} - M_{i5} = 0$$
$$CM_{i2} - M_{i4} + M_{i5} = 0$$

$$\longrightarrow \quad M = \begin{bmatrix} 1/2 & 1/2 & \frac{1}{2}C & \frac{3}{2}C & C \\ -1/2 & 1/2 & \frac{3}{2}C & \frac{3}{2}C & C \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} \xrightarrow[C \to 0]{\lim} M_+ = \begin{bmatrix} 1/2 & 1/2 & 0 & 0 & 0 \\ -1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & -1 \\ 0 & 1 & 0 & -1 & 1 \end{bmatrix}$$

Adapted from C. Z. Pratt, K. J. Ray, and J. P. Crutchfield, "Extracting Equations of Motion from Superconducting Circuits." arXiv, (Jan 2024)

# QFP Analysis: Circuit Quantization




$$\begin{pmatrix} \Phi_{J_1} \\ \Phi_{J_2} \\ \Phi_{L_1} \\ \Phi_{L_2} \\ \Phi_{L_q} \end{pmatrix} = \begin{pmatrix} \tilde{\Phi}_1 - \tilde{\Phi}_2 \\ \tilde{\Phi}_1 + \tilde{\Phi}_2 \\ \frac{1}{3}(-\tilde{\Phi}_1 + \tilde{\Phi}_2 + \tilde{\Phi}_3 + 2\Phi_{ex1} + \Phi_{ex2}) \\ \frac{1}{3}(3\tilde{\Phi}_1 + \tilde{\Phi}_2 + \tilde{\Phi}_3 - \Phi_{ex1} - 2\Phi_{ex2}) \\ \frac{1}{3}(-2\tilde{\Phi}_2 + \tilde{\Phi}_3 - \Phi_{ex1} + \Phi_{ex2}) \end{pmatrix}$$

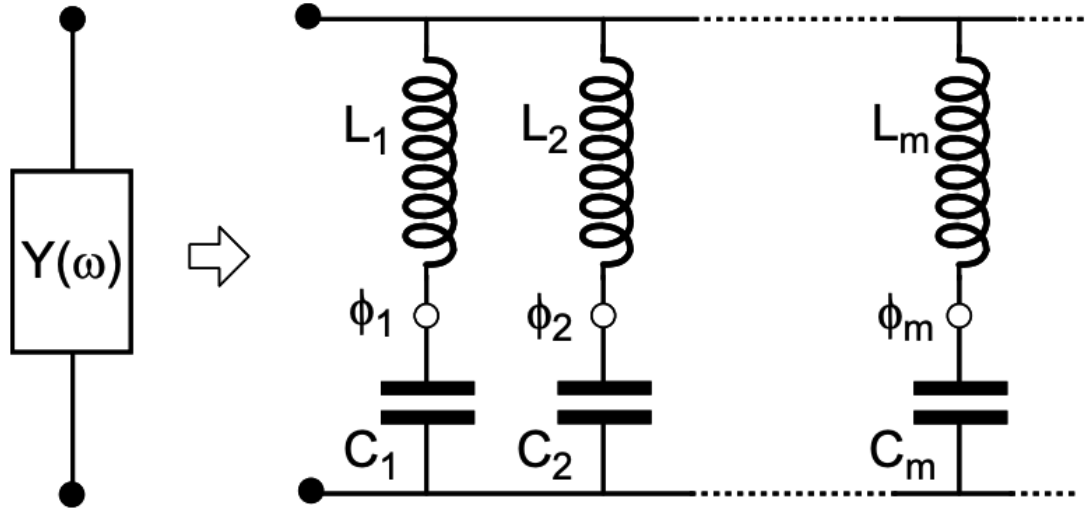
Usually $\Phi_{ex1} = \Phi_{ex2}$

$$\begin{aligned} \mathcal{L} = & \frac{1}{2} C_j (\dot{\tilde{\Phi}}_1^2 + \dot{\tilde{\Phi}}_2^2) \\ & - \frac{1}{9L_1}(-\tilde{\Phi}_1 + \tilde{\Phi}_2 + \tilde{\Phi}_3 + 3\Phi_{ex})^2 \\ & - \frac{1}{9L_2}(3\tilde{\Phi}_1 + \tilde{\Phi}_2 + \tilde{\Phi}_3 - 3\Phi_{ex})^2 \\ & - \frac{1}{9L_q}(-2\tilde{\Phi}_2 + \tilde{\Phi}_3)^2 \\ & + E_J \cos(\tilde{\varphi}_1 - \tilde{\varphi}_2) + E_J \cos(\tilde{\varphi}_1 + \tilde{\varphi}_2) \end{aligned}$$

Citations

# Next Steps

- Add dissipation to the circuit: extend the circuit Hamiltonian with the Caldeira-Leggett model for admittance to represent a dampening resistor in parallel with the JJ




Citations

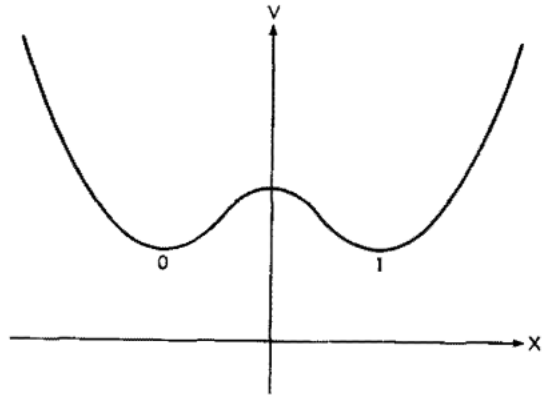
# Fundamental Physical Limits of Computation

# CONTENTS

# Landauer Limit

The minimum energy requires to erase one bit of information is set by

$$E \geq k_b \mathrm{T} \ln(2)$$




- Note that this only applies to computational information that is correlated with other available information, as opposed to be random and independent

R. Landauer, "Irreversibility and Heat Generation in the Computing Process," *IBM Journal of Research and Development* (1961), doi: 10.1147/rd.53.0183

# Reversible Computing

- Logically Reversible :
  - Known information is not discarded from the digital state of the machine through ejection into the thermal environment
  - Every desired irreversible computation can always be embedded into a functionally equivalent reversible one [1]

- Thermodynamically Reversible :
  - The total increase in entropy from the machine's operation per useful computational operation should be extremely small, and asymptotically approach zero as the technology improves

- Bypass Laundauer's limit by never erasing anything

[1] C. H. Bennett, "Logical Reversibility of Computation," *IBM J. Res. & Dev.*, vol. 17, no. 6, pp. 525–532, (Nov. 1973)

# Reversible Computing

“the ideal of physically reversible computing is by no means an easy path forward, it is the only way that general digital computing can continue to move forward indefinitely, with no clear limits in the foreseeable future—in contrast with the conventional, non-reversible computing paradigm, which is necessarily limited by Landauer’s principle” [1]

[1] M. P. Frank, “Physical Foundations of Landauer’s Principle,” vol. 11106, 2018, pp. 3–33. doi: 10.1007/978-3-319-99498-7_1.

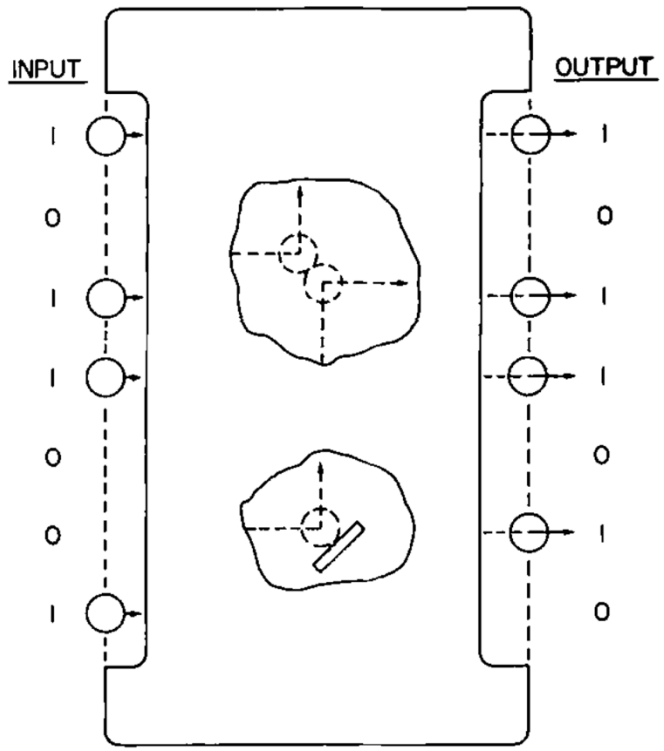
# Reversible Computing

- In modern day adiabatic devices, there's typically a tradeoff between the energy dissipated and the delay interval required to carry out a primitive digital operation
- dissipation-delay product (DdP):

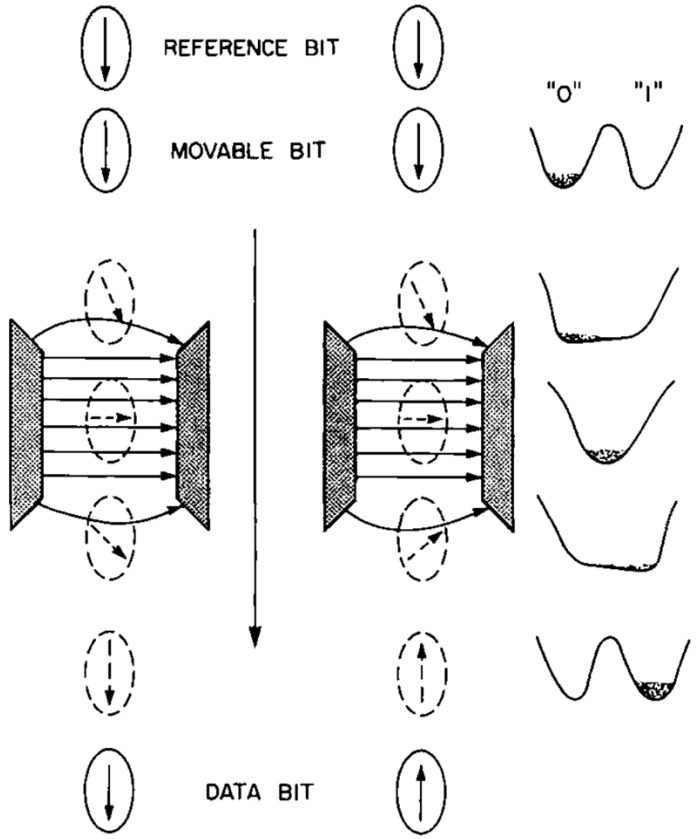
$$E_{diss} * t_{delay} = c_e$$

  where $c_e$ is the energy coefficient of the technology, or the dissipation-delay constant
  - there are no known technology-independent boundaries on the dissipation-delay constant

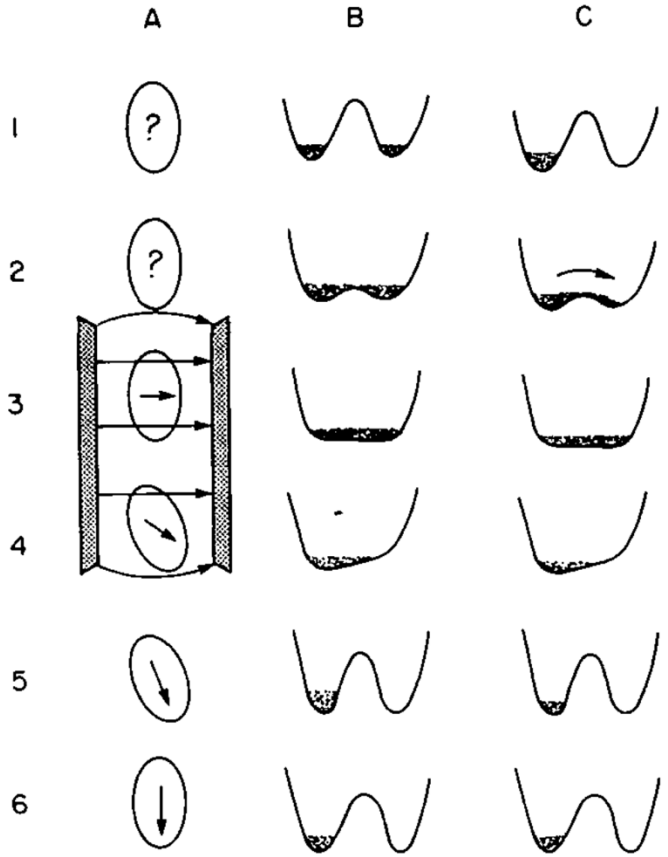
# Reversible Computing: Toy Examples




Logically reversible ballistic computer
The input is always conserved in the output



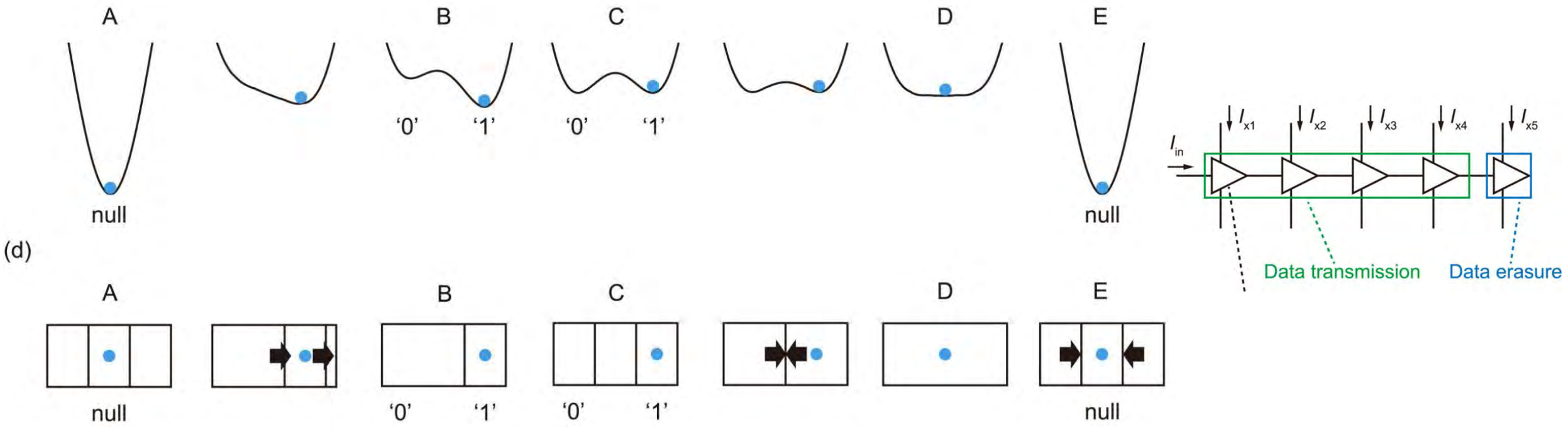

Reversible copying using a one-domain ferromagnet




Erasure of bistable one-domain ferromagnet

C. H. Bennett, "The thermodynamics of computation—a review", *Int J Theor Phys* (1982)

# Reversible Computing in AQFP

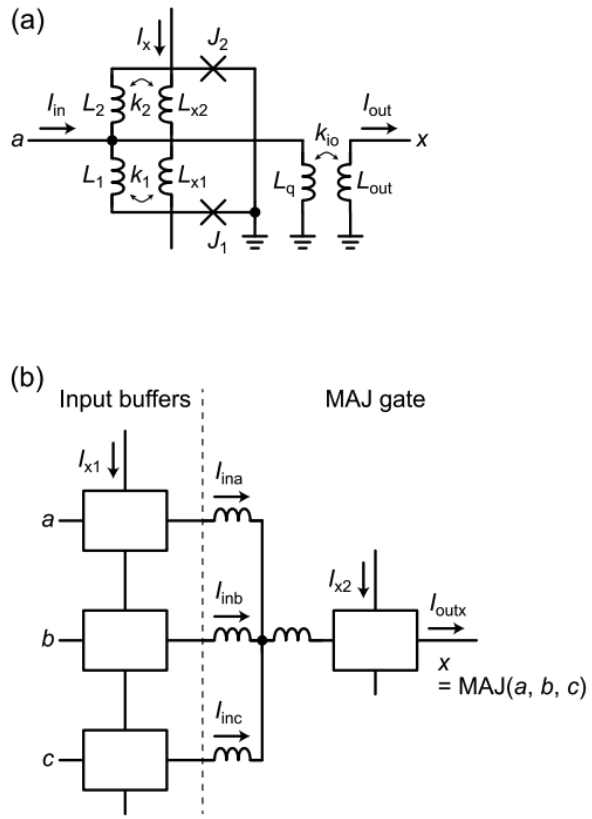



The erasure comes whenever the QFP is not balance on both input and output, i.e. where it appears in the chain

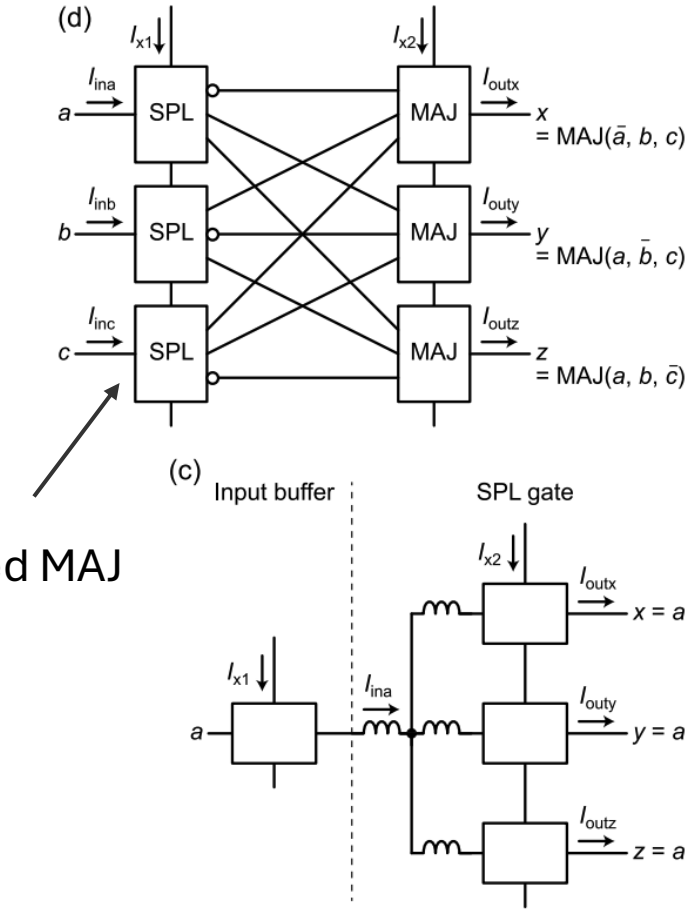
T. Yamae, N. Takeuchi, and N. Yoshikawa, "Minimum energy dissipation required for information processing using adiabatic quantum-flux-parametron circuits," *Journal of Applied Physics* (Feb. 2024,)

# AQFP Reversible Computing

AQFP irreversible logic gates:



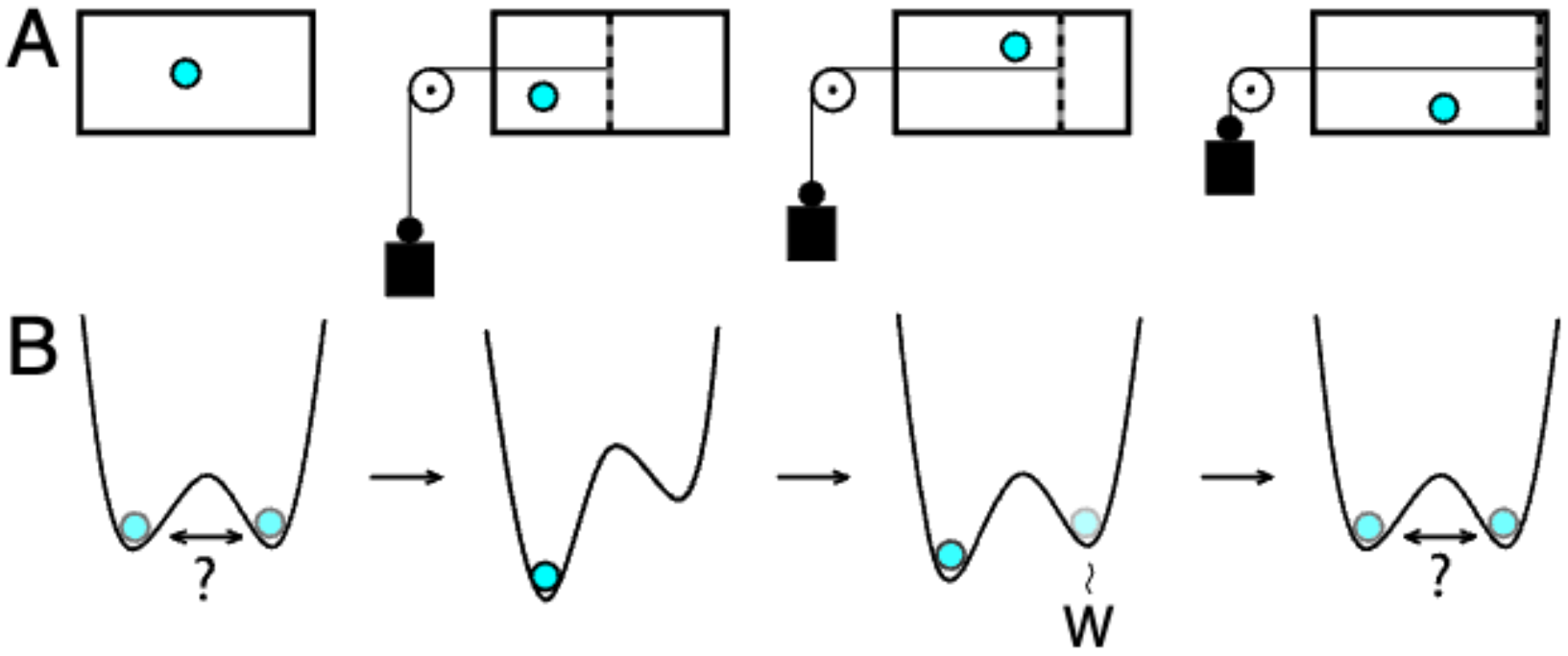

RQFP reversible logic gates:




Splitter, inverted MAJ

N. Takeuchi, Y. Yamanashi, and N. Yoshikawa, "Reversible logic gate using adiabatic superconducting devices," *Sci Rep*, vol. 4, no. 1, p. 6354, Sep. 2014, doi: 10.1038/srep06354

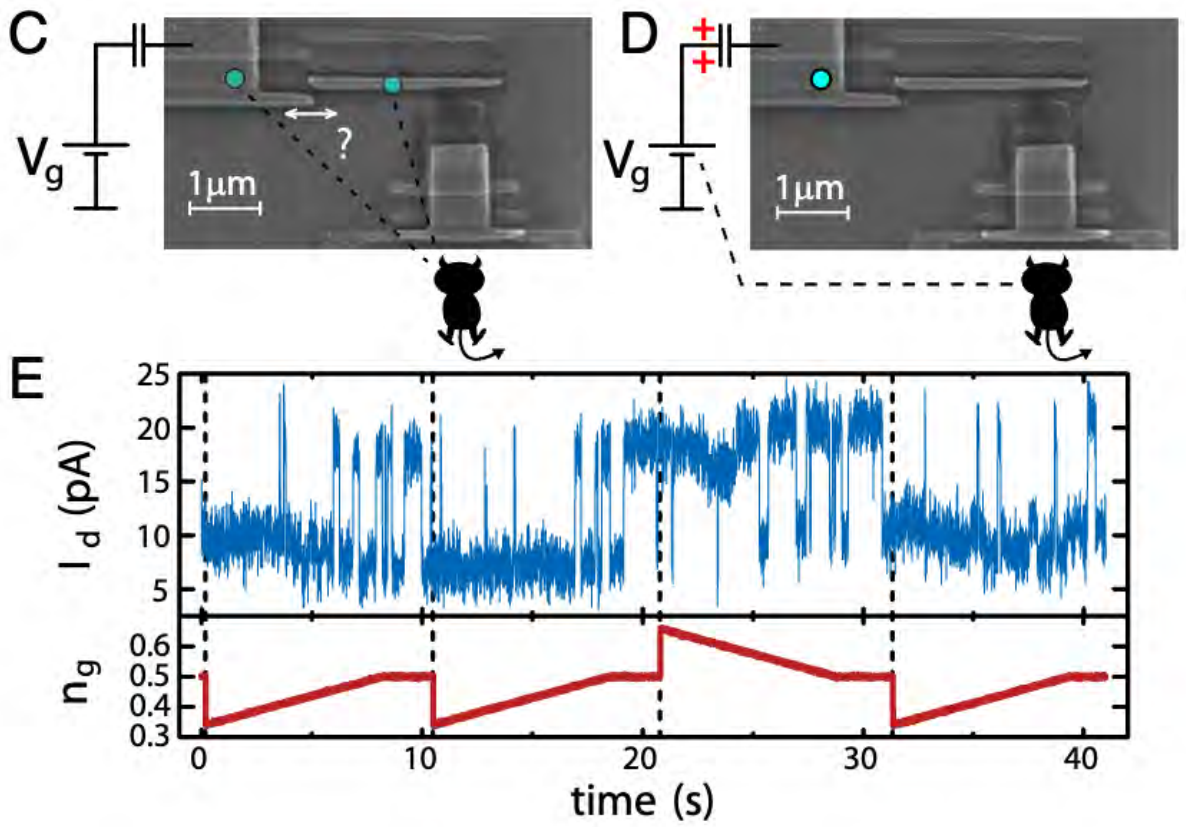
# Szilard Engine – Thought Experiment

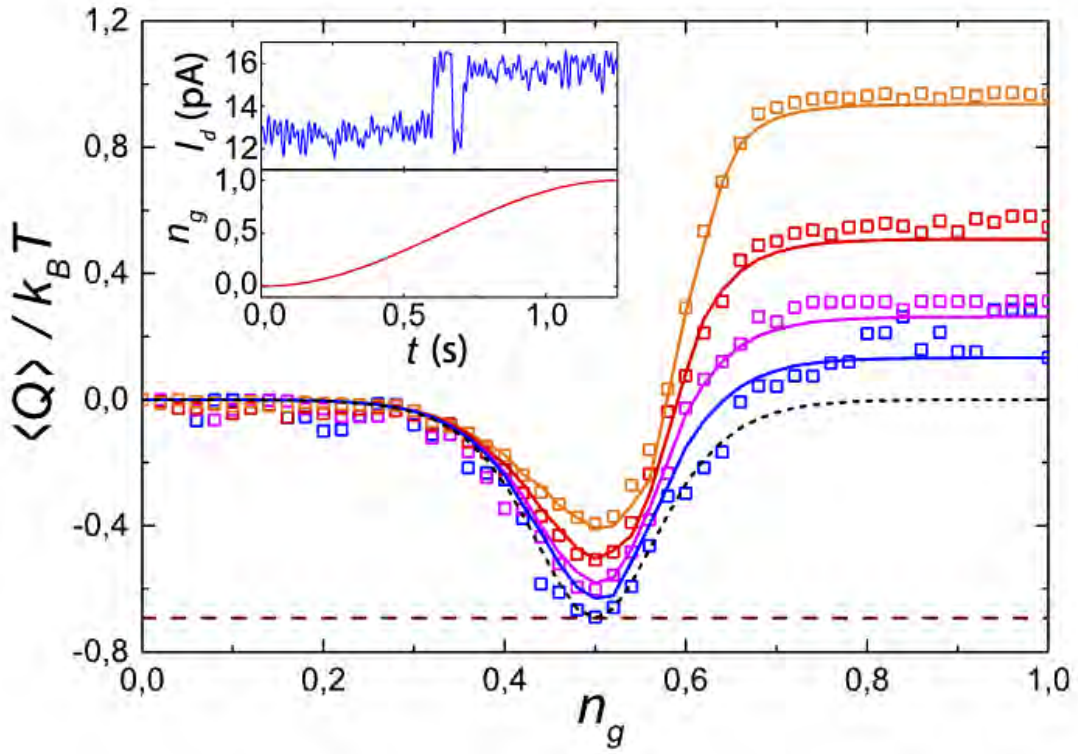



A Szilard engine uses the information about the location of a bit in a box to extract $k_b T \ln(2)$ of work

J. Koski, V. Maisi, J. Pekola, and D. Averin, "Experimental realization of a Szilard engine with a single electron", PNAS (2014)

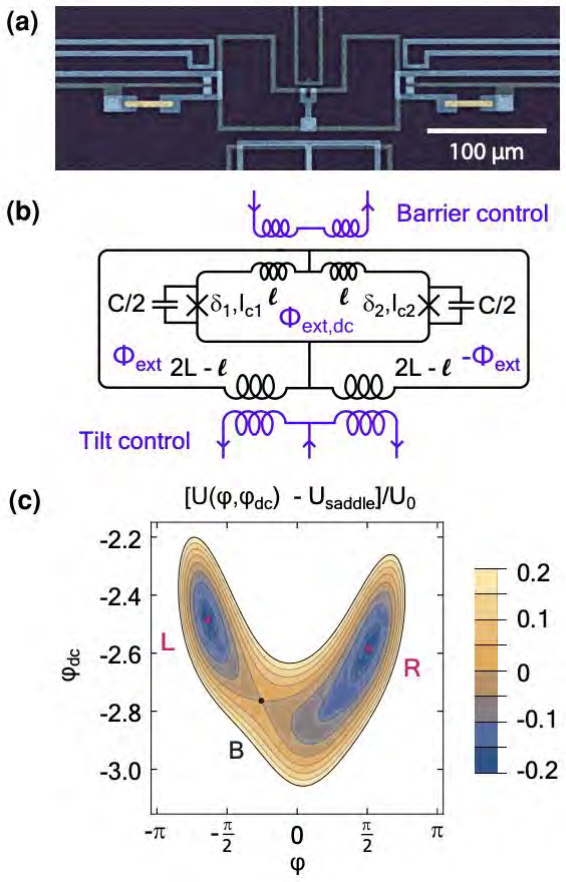
# Single electron Szilard Engine



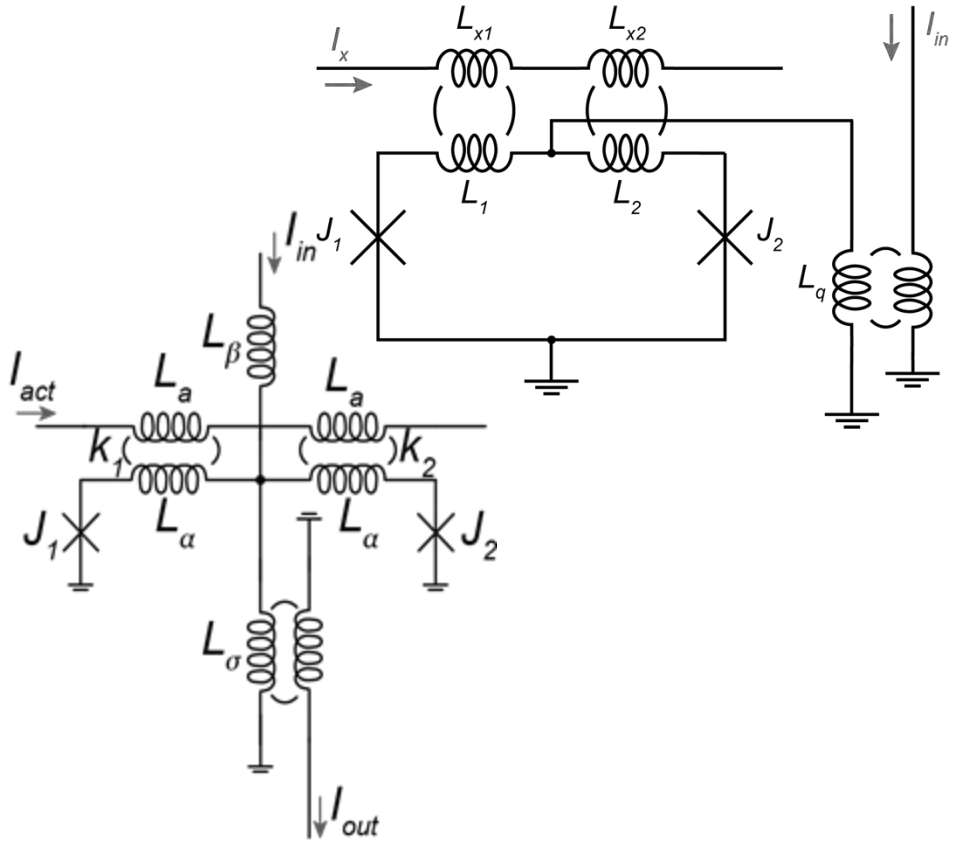




J. Koski, V. Maisi, J. Pekola, and D. Averin, "Experimental realization of a Szilard engine with a single electron", PNAS (2014)

# “AQFP” Szilard Engine




QFP!




$$H = \frac{Q^2}{2C} + \frac{Q_{dc}^2}{C/2} + U_0 f(\varphi, \varphi_{dc}),$$

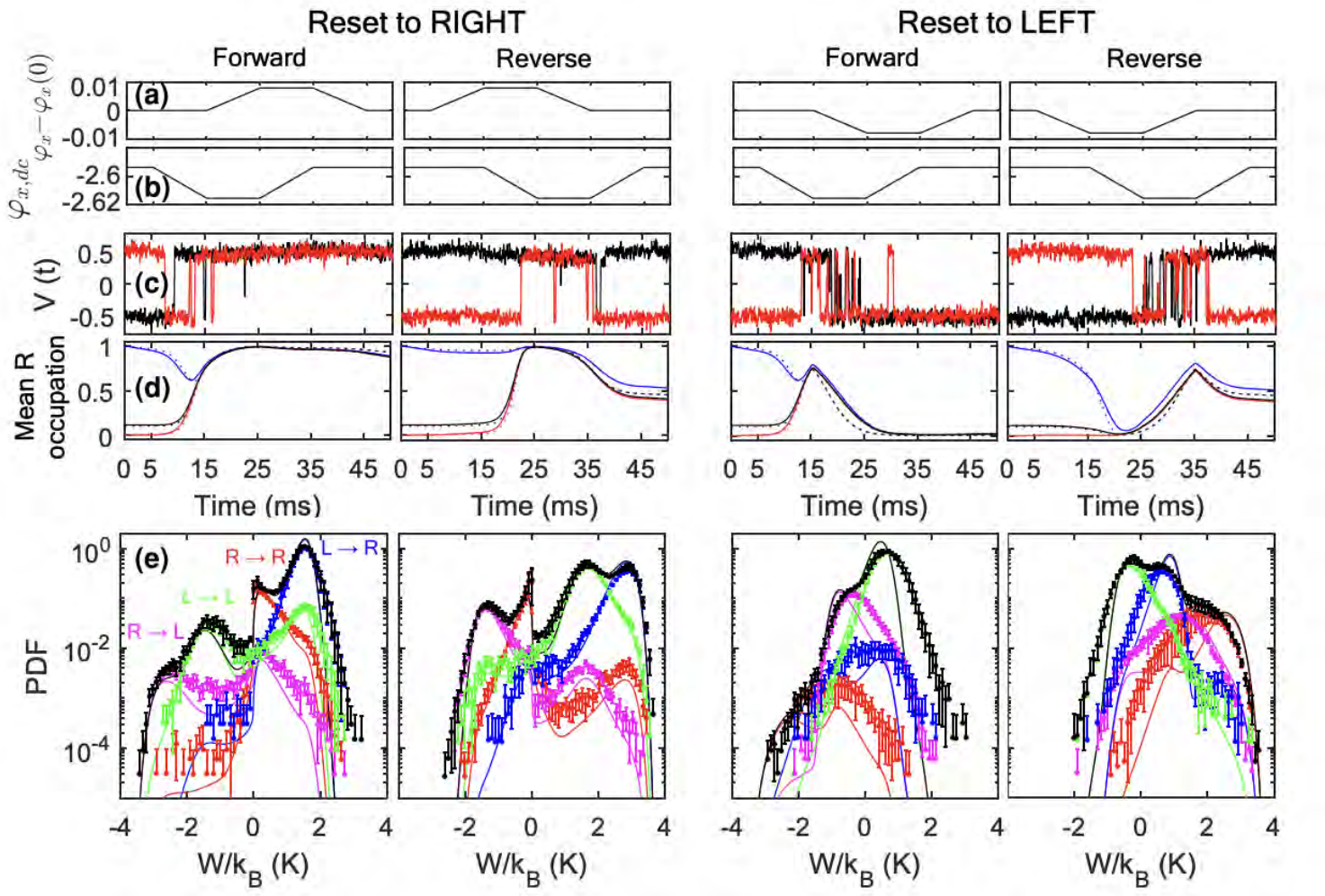
$$f(\varphi, \varphi_{dc}) = \frac{1}{2}(\varphi - \varphi_x)^2 + \frac{\gamma}{2}(\varphi_{dc} - \varphi_{x,dc})^2 + \beta_L \cos\frac{\varphi_{dc}}{2}\cos\varphi + \delta\beta \sin\frac{\varphi_{dc}}{2}\sin\varphi.$$

[1] O.-P. Saira *et al.*, “Nonequilibrium thermodynamics of erasure with superconducting flux logic,” *Phys. Rev. Research*, vol. 2, no. 1, p. 013249, (Mar. 2020)

# “AQFP” Szilard Engine

$\varphi_x - \varphi_x(0)$ → data signal
$\varphi_{x,dc}$ → activation signal

Red trace → left well state
Blue trace → right well state

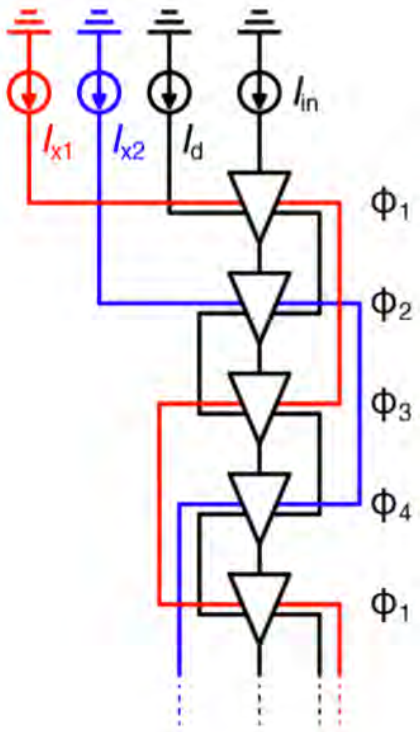


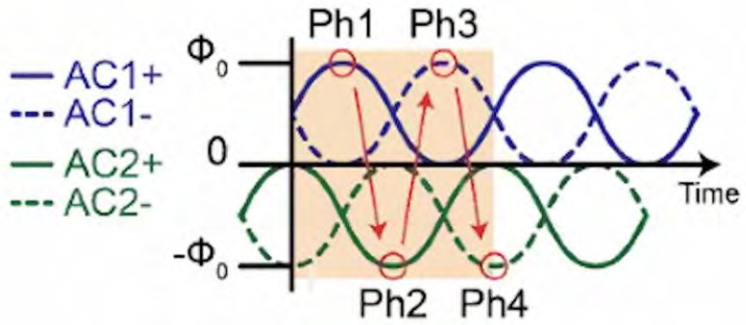

[1] O.-P. Saira *et al.*, “Nonequilibrium thermodynamics of erasure with superconducting flux logic,” *Phys. Rev. Research*, vol. 2, no. 1, p. 013249, (Mar. 2020)

# Additional Slides

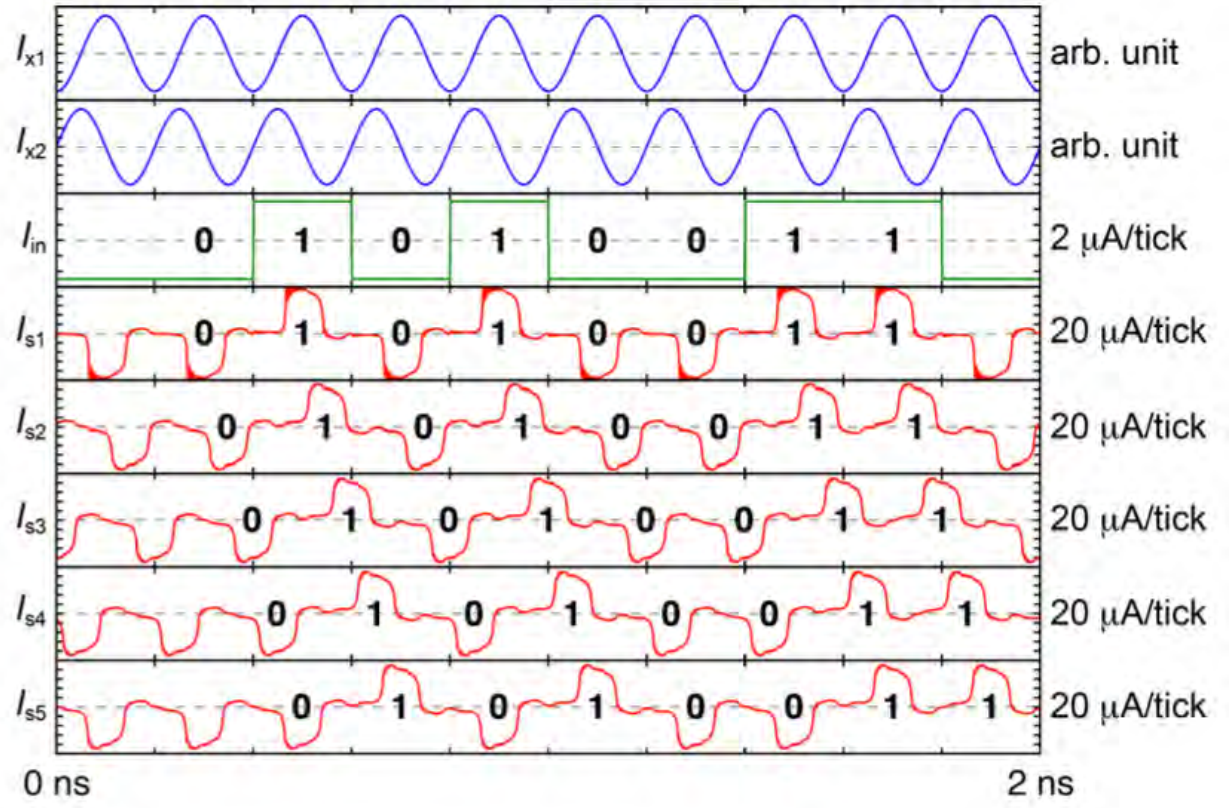
# AQFP Four Phase Clocking

Logic states are determined by the polarity of a current pulse that is baton-passed between chains of AQFP devices, therefore a multiphase clocking scheme is needed for logic propagation.



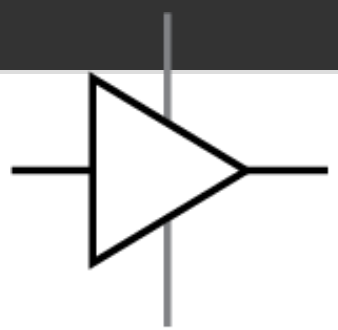




Therefore, some notion of synchronous clock is required for AQFP logic

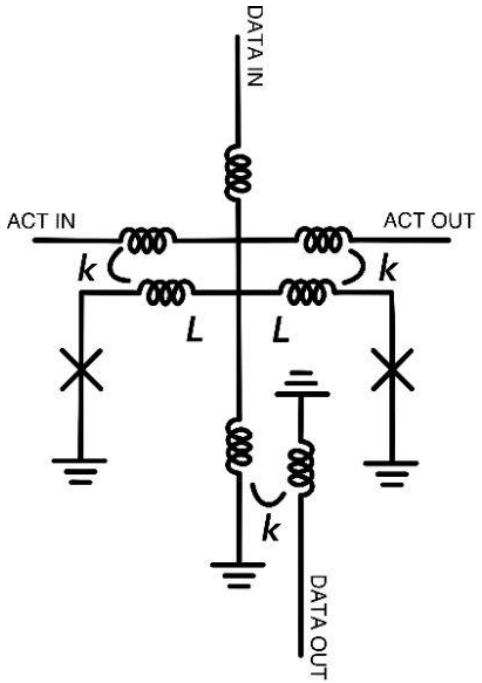


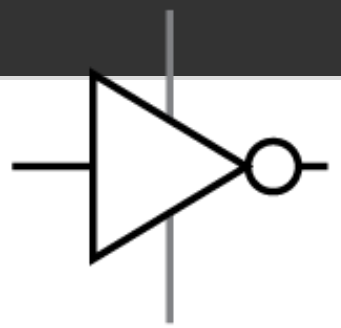

N. Takeuchi, *et. al.*, "Adiabatic Quantum-Flux-Parametron: A Tutorial Review", *IEICE Trans. Electron.* (2022)

# AQFP Logic Library

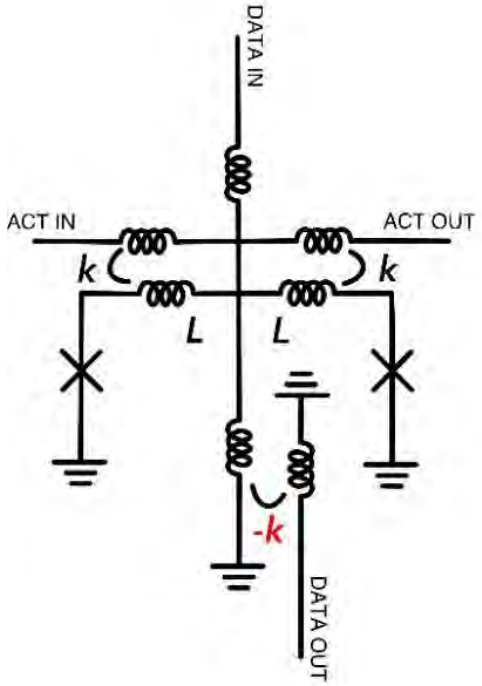


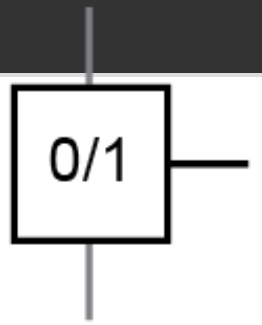
BUF



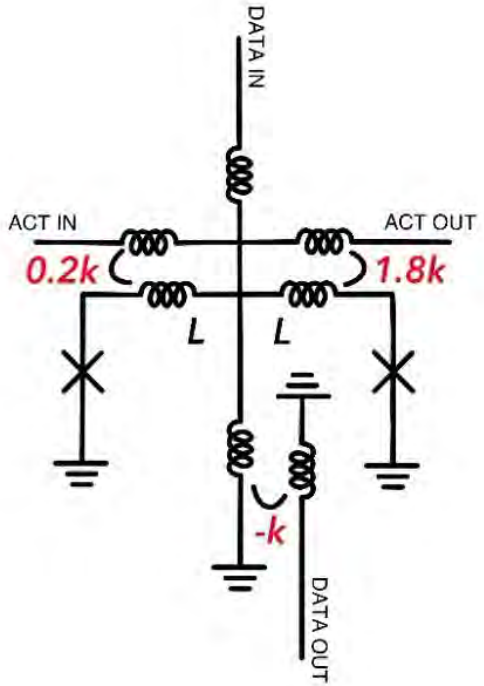



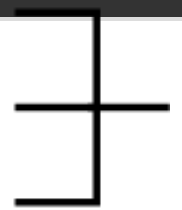
INV



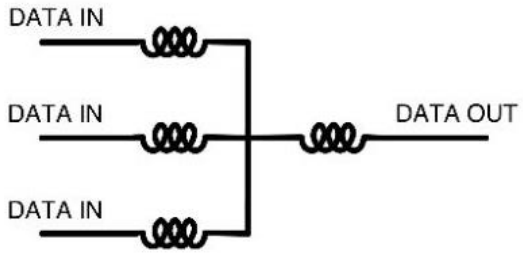




CONST 0/1



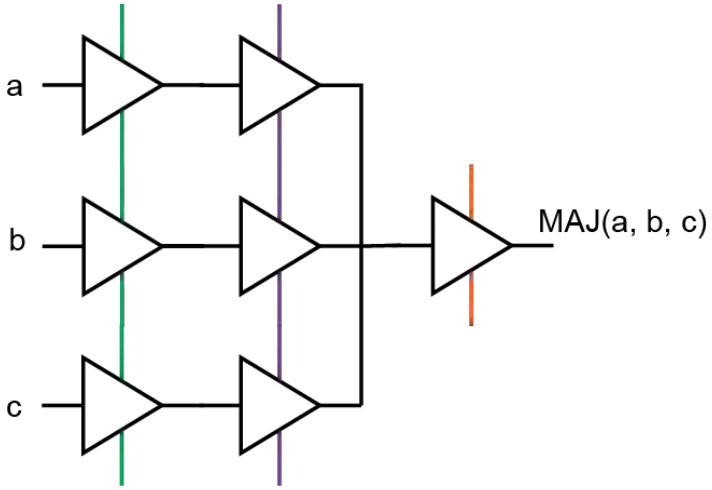
MAJ

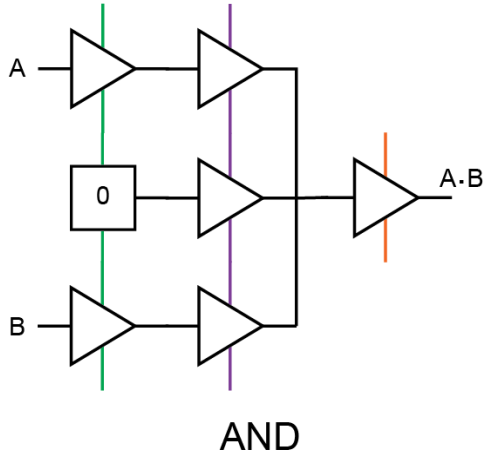


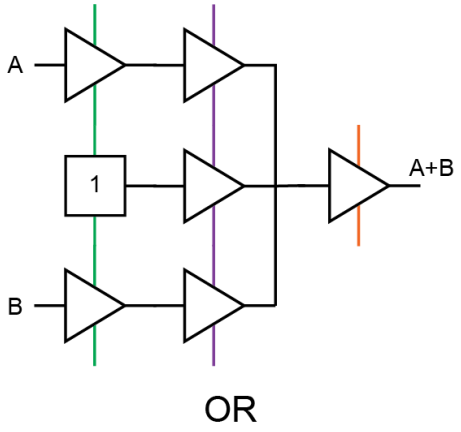

Citations

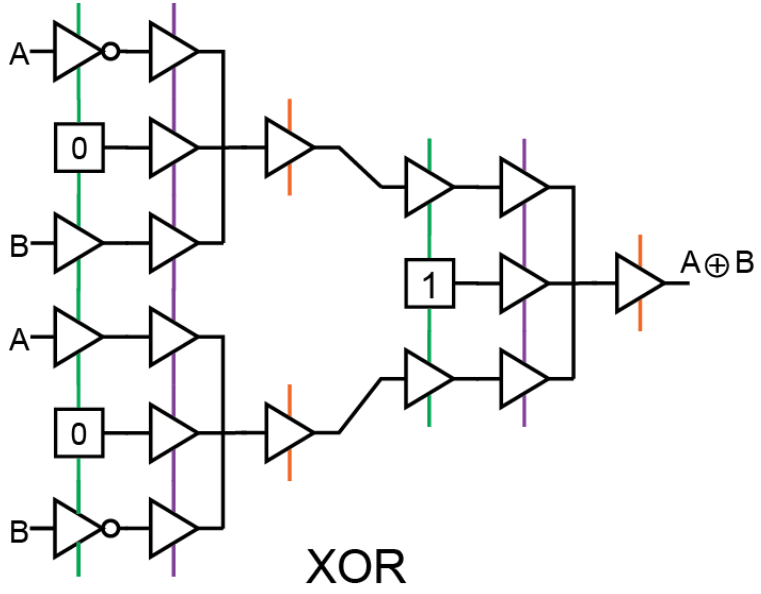
# AQFP Combinational Logic

Majority gate logic:



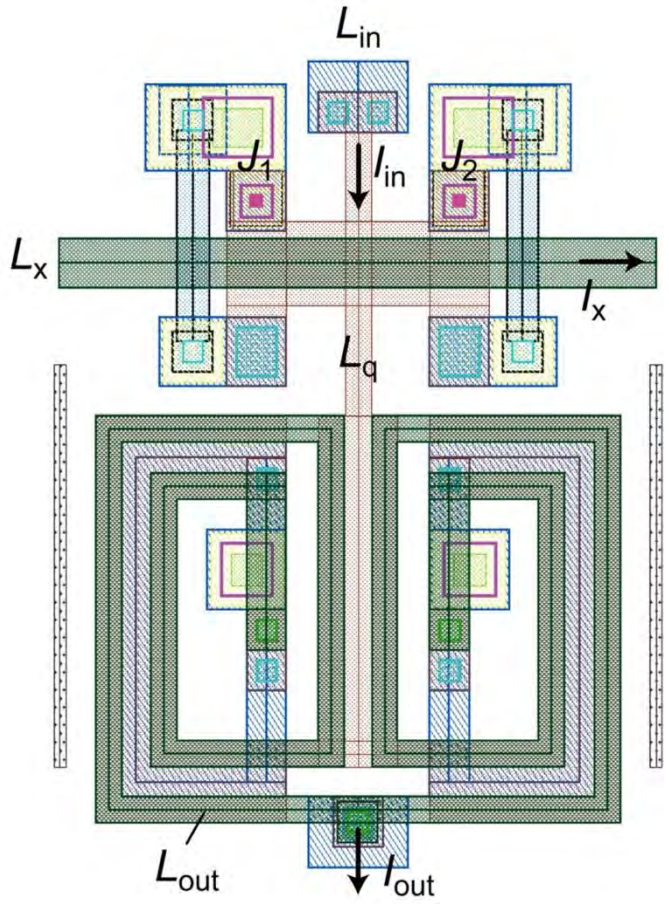




Citations

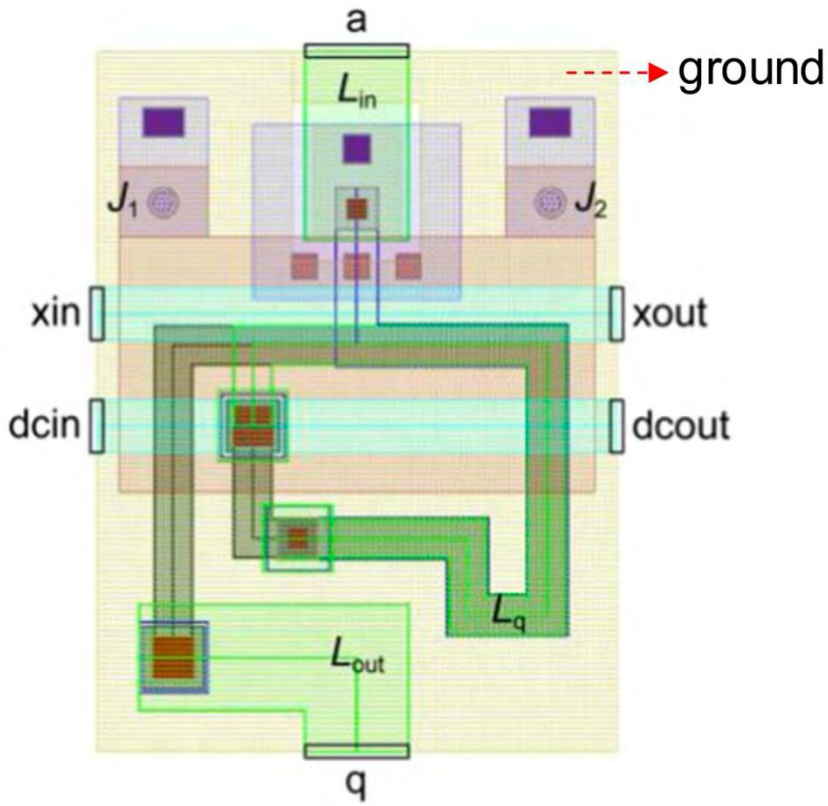
# AQFP Layouts

30$\mu$m x 40$\mu$m, AIST-STP2 process



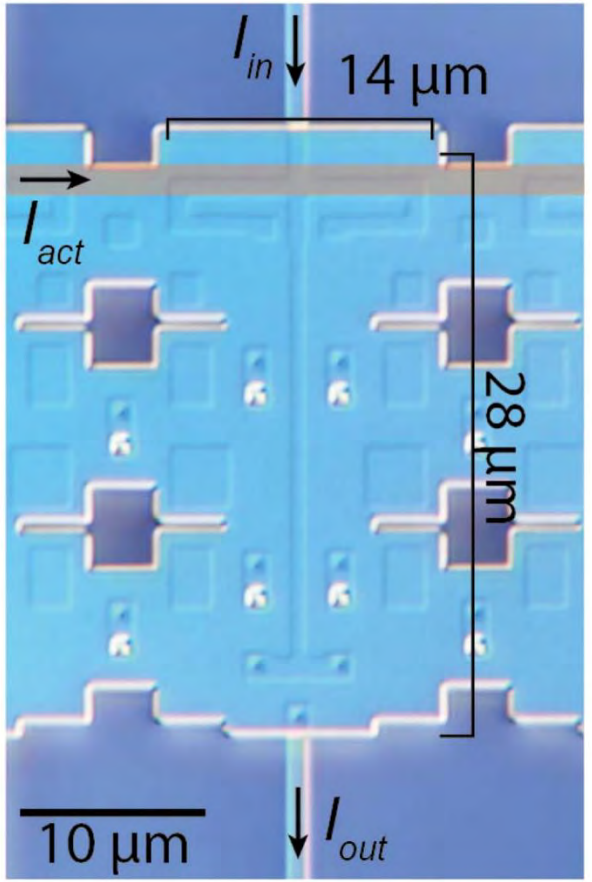

N. Takeuchi et. al., J. Appl. Phys. 117, 173912 (2015)

15$\mu$m x 20$\mu$m, MITLL SFQ5ee process




Y. He et. al., Supercond. Sci. Technol. 33 (2020)

MITLL SFQ5ee process



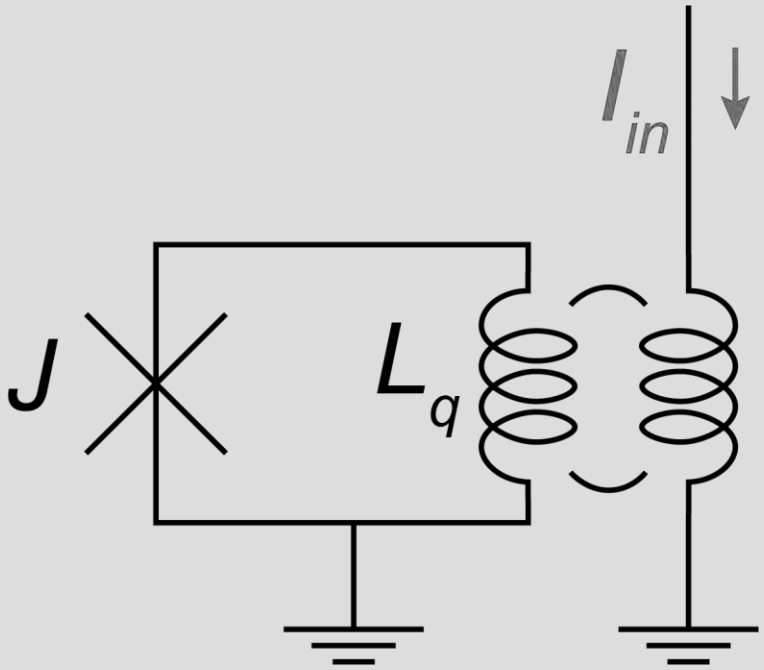

L.C. Blackburn et. al., IEEE TAS, vol. 33, 5, (2023)

Citations

AQFP Potential Energy derivation (from circuit elements instead of Lagrangian)

# rf-SQUID TODO: update variable names

Superconducting Quantum Interference Device (SQUID)
rf-SQUID: single junction in a loop




$$U_{\mathrm{rf}} = U_J + U_m$$

$$U_J = -E_j \cos\phi$$

$$U_m = \frac{E_j \phi_L^2}{2\beta_L} = \frac{1}{2} L i^2$$

$$E_j = \frac{I_c \Phi_0}{2\pi}$$

$$\beta_L = \frac{2\pi L I_c}{\Phi_0}$$

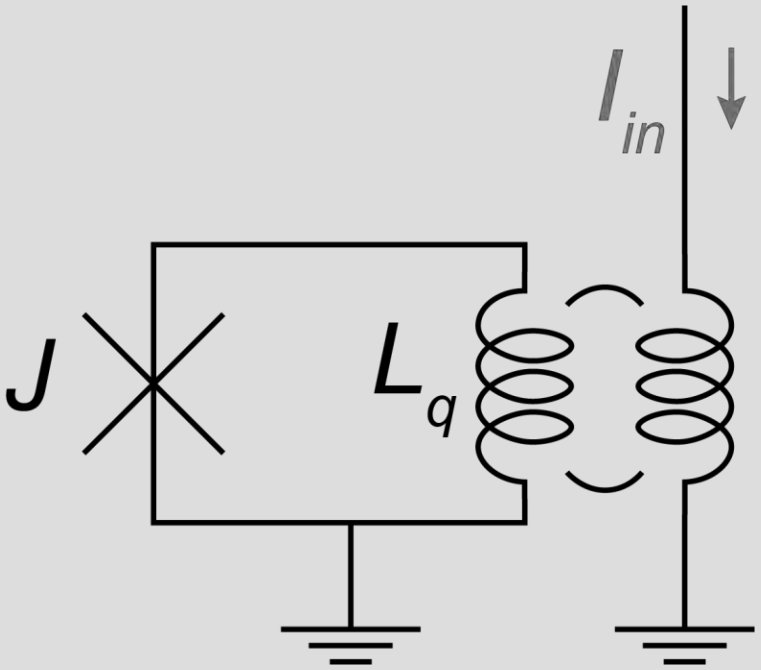
Conservation of flux / current:

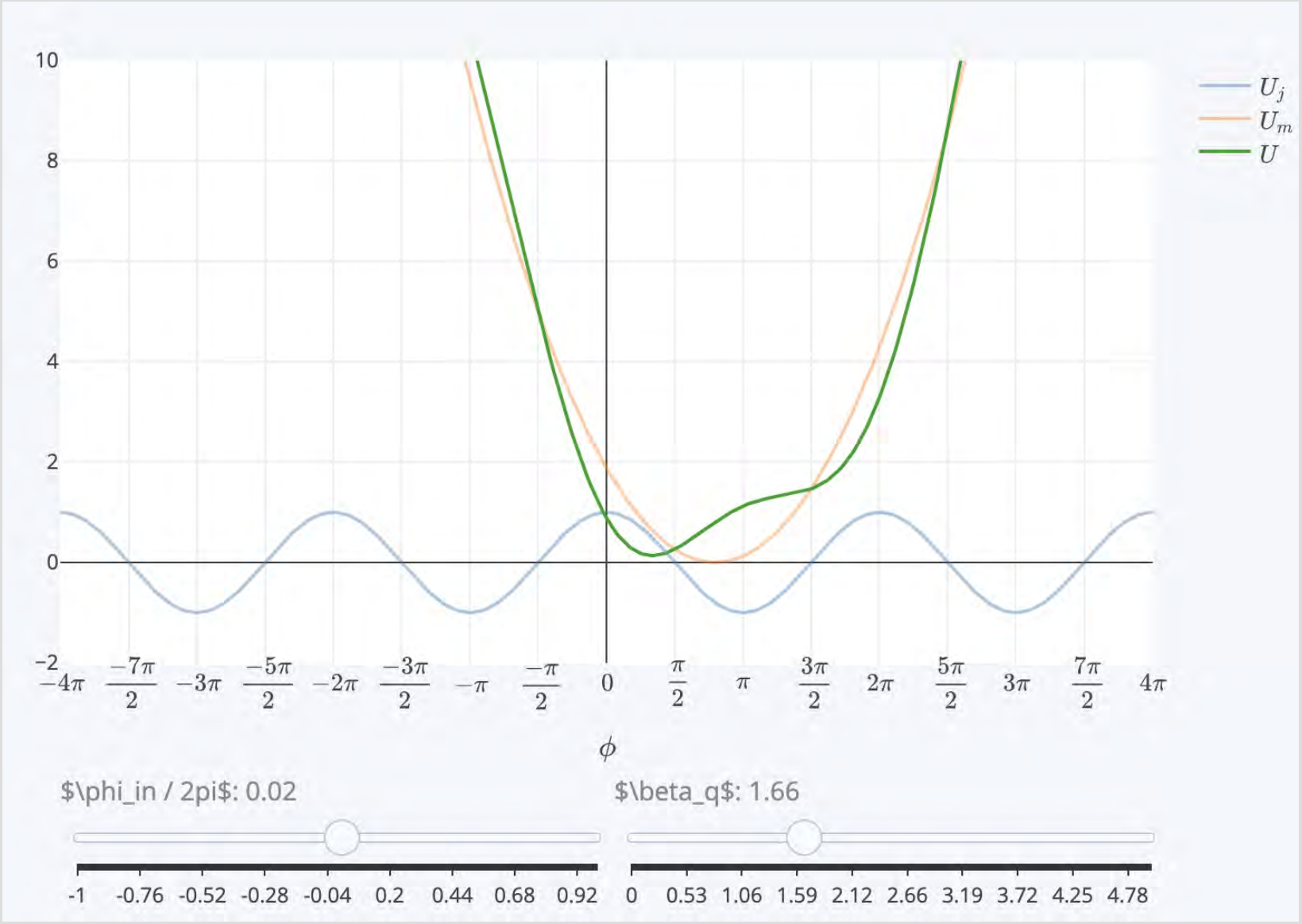
$$\phi - \phi_L = \phi_b$$

$$\frac{U_{\mathrm{rf}}}{E_j} = \frac{(\phi - \phi_b)^2}{2\beta_L} - \cos\phi$$

# rf-SQUID

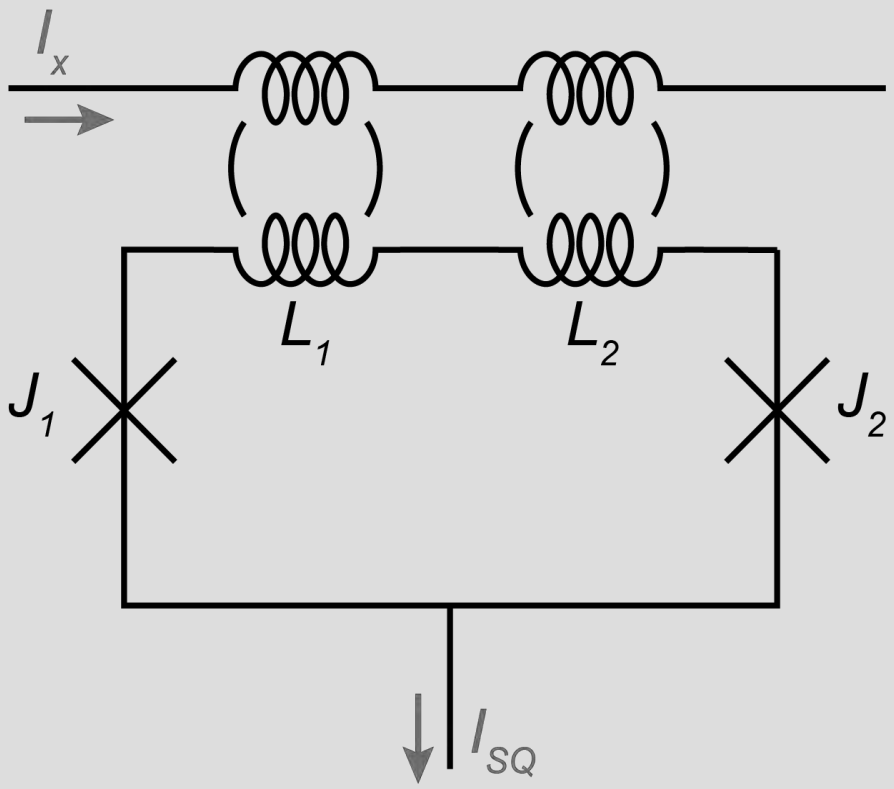
$$\frac{U_{\mathrm{rf}}}{E_j} = \frac{(\phi - \phi_b)^2}{2\beta_L} - \cos\phi$$



# dc-SQUID

dc-SQUID: two junctions inside a loop

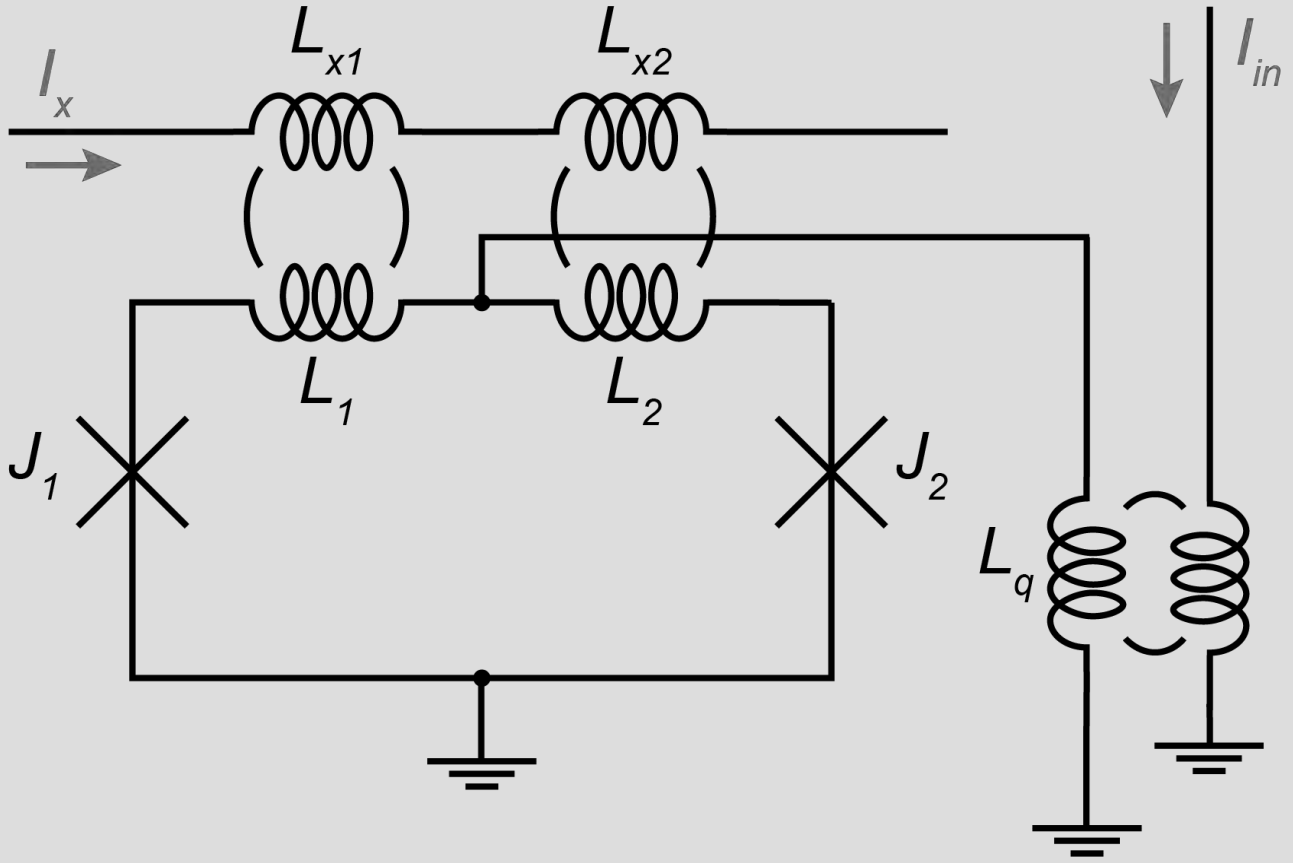



Behaves like a JJ with a tunable $I_c$, dependent on an externally applied flux, $\Phi_{ext}$ (in this case, $I_x$ coupled through the transformer)

If we ignore the loop inductance and assume that both junctions are identical, the new critical current of the effective JJ is

$$I_{c,SQ} = 2I_c \cos\frac{\pi\Phi_{ext}}{\Phi_0}$$

# QFP Potential

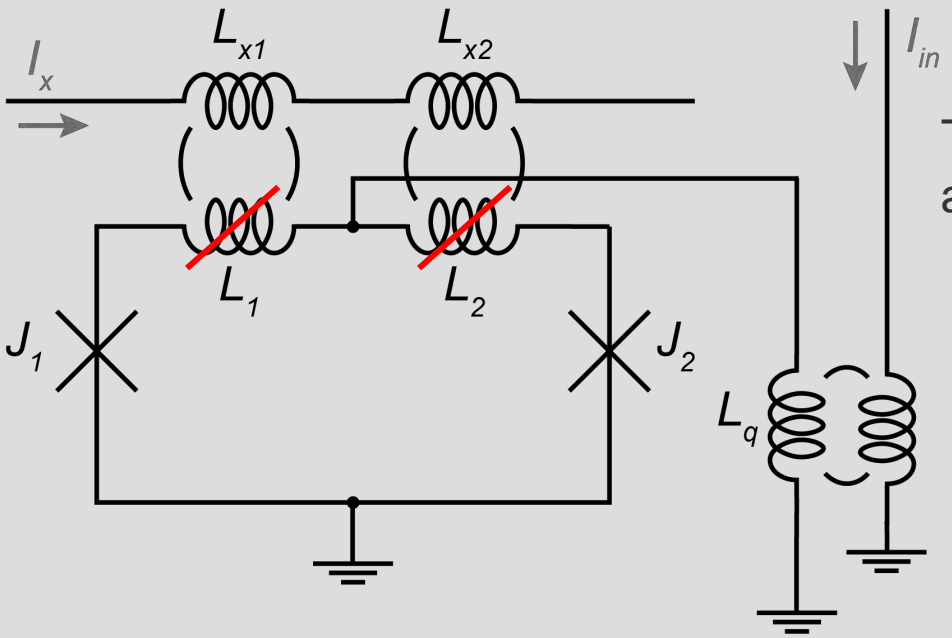



Quantum Flux Parametron (QFP) can be thought of as an rf-SQUID with the single JJ replaced by a tunable dc-SQUID

$I_x$: activation signal

$I_{in}$: data signal (input/output shares a net)

# QFP Potential




$$U_{\text{qfp}} = U_J + U_m$$

To start, let's assume that $L_1$ and $L_2$ are small and their energy contributions can be ignored

$$\begin{aligned} U_J &= U_{J_1} + U_{J_2} \\ &= E_J(-\cos\phi_1 - \cos\phi_2) \\ &= -2E_J \cos\phi_+ \cos\phi_- \end{aligned}$$

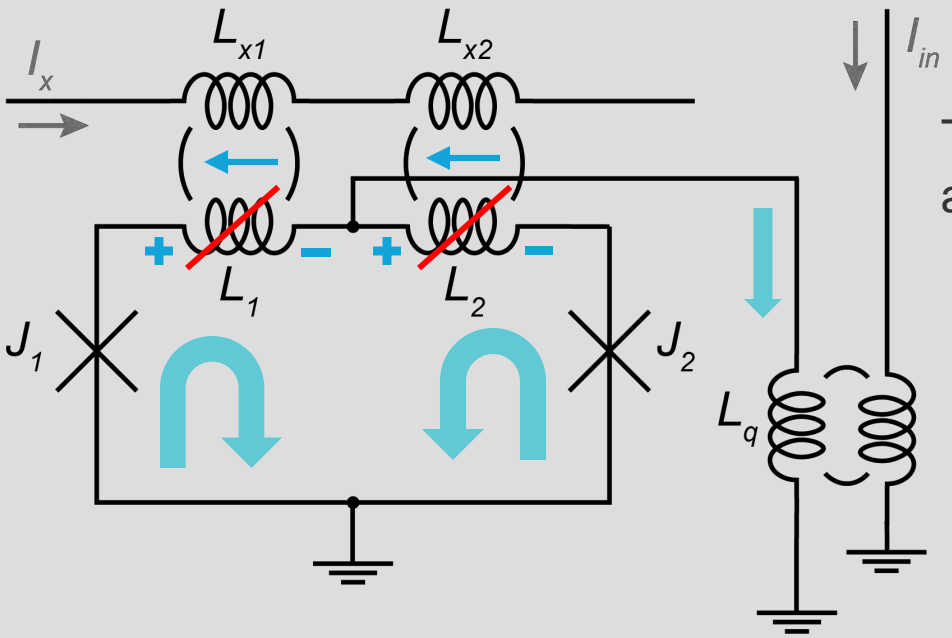
$$U_m = \frac{E_j \phi_q^2}{2\beta_q}$$

$$\beta_q = \frac{2\pi L_q I_c}{\Phi_0}$$

$$\phi_- = \frac{1}{2}(\phi_1 - \phi_2)$$

$$\phi_+ = \frac{1}{2}(\phi_1 + \phi_2)$$

# QFP Potential




$$U_{\text{qfp}} = U_J + U_m$$

To start, let's assume that $L_1$ and $L_2$ are small and their energy contributions can be ignored

$$\begin{aligned} U_J &= U_{J_1} + U_{J_2} \\ &= E_J(-\cos\phi_1 - \cos\phi_2) \\ &= -2E_J \cos\phi_+ \cos\phi_- \end{aligned}$$

$$U_m = \frac{E_j \phi_q^2}{2\beta_q}$$

$$\beta_q = \frac{2\pi L_q I_c}{\Phi_0}$$

$$\phi_- = \frac{1}{2}(\phi_1 - \phi_2)$$

$$\phi_+ = \frac{1}{2}(\phi_1 + \phi_2)$$

Conservation of flux / current:

$$\phi_1 - \phi_x = \phi_2 + \phi_x = \phi_q + \phi_{\text{in}}$$

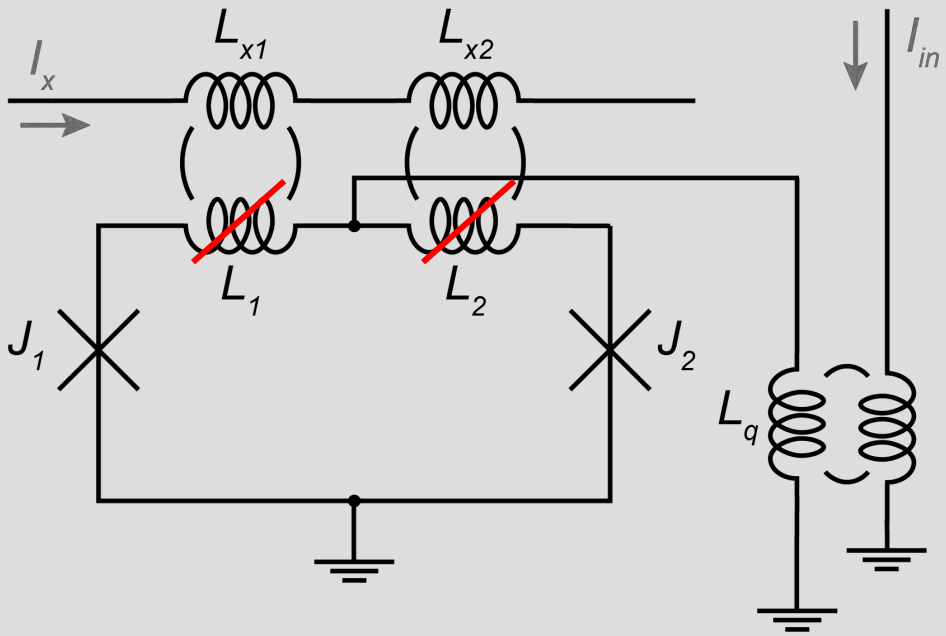
$$\rightarrow \phi_q = \phi_+ - \phi_{\text{in}}$$

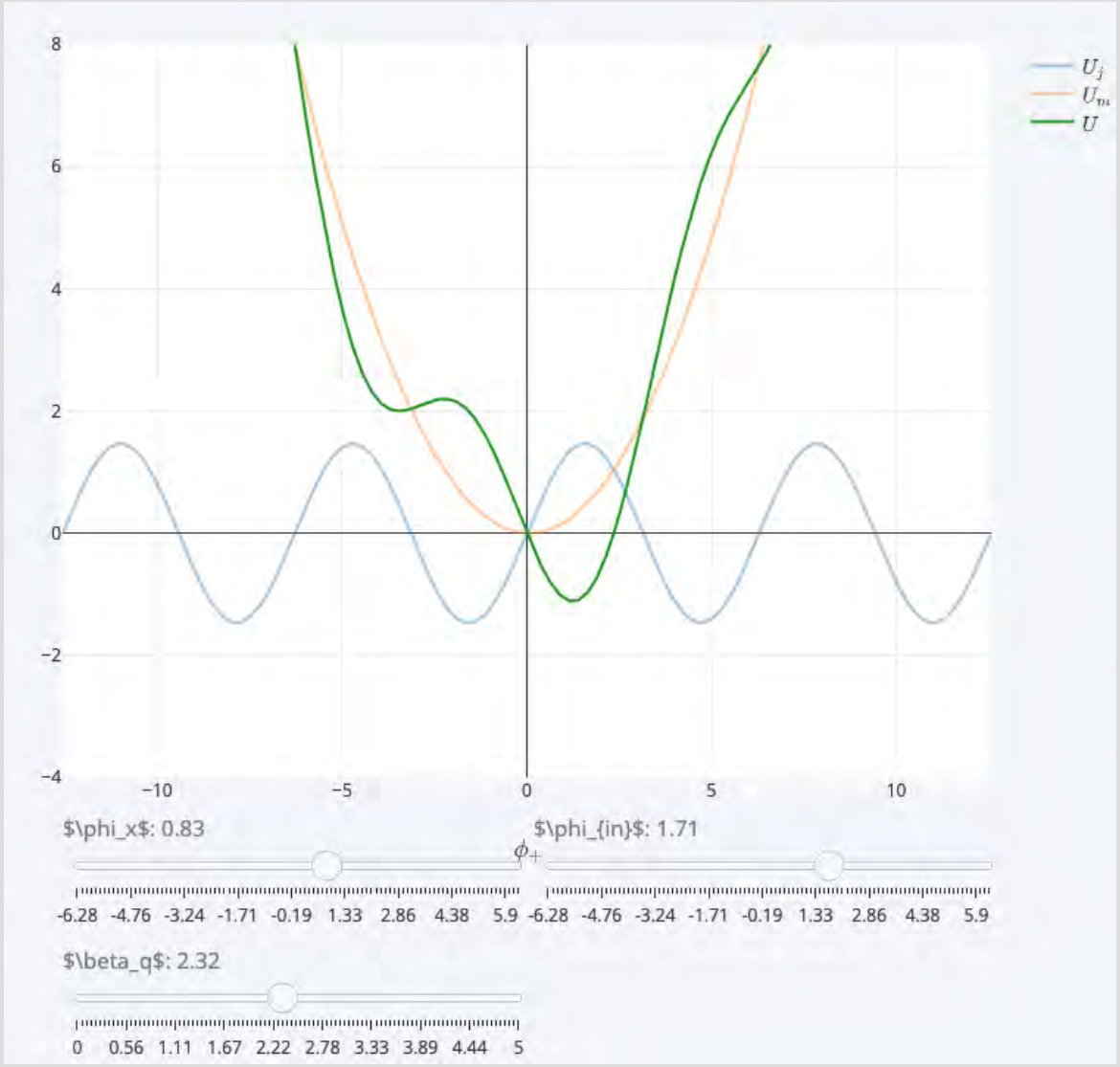
$$\rightarrow \phi_- = \phi_x$$

$$\frac{U_{\text{qfp}}}{E_j} = \frac{(\phi_+ - \phi_{\text{in}})^2}{2\beta_q} - 2\cos\phi_x \cos\phi_+$$

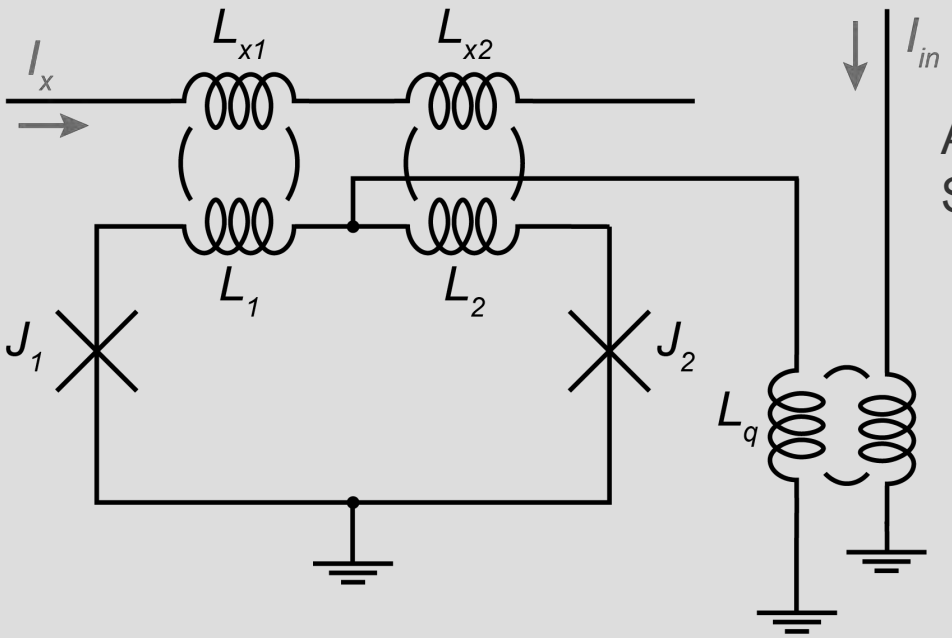
# QFP Potential

$$\frac{U_{\mathrm{qfp}}}{E_j} = \frac{(\phi_+ - \phi_{\mathrm{in}})^2}{2\beta_q} - 2\cos\phi_x \cos\phi_+$$



# QFP Potential




$$U_{\mathrm{qfp}} = U_J + U_m$$

A more accurate QFP potential includes dc-SQUID inductors

$$U_J = -2E_J \cos\phi_+ \cos\phi_-$$

$$U_m = \frac{E_j(\phi_{L_1}^2 + \phi_{L_2}^2)}{2\beta_L} + \frac{E_j\phi_q^2}{2\beta_q}$$

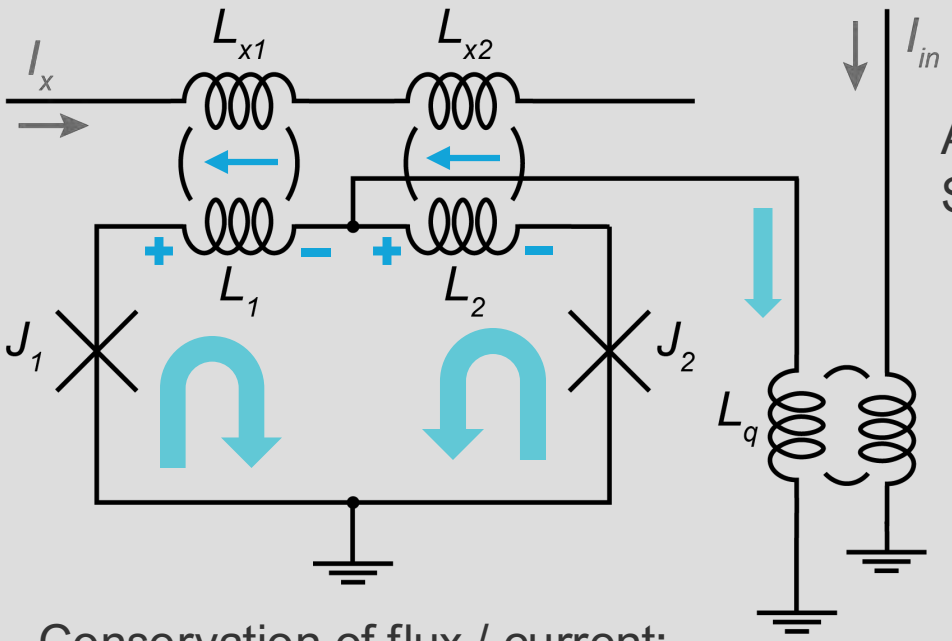
$$\beta_q = \frac{2\pi L_q I_c}{\Phi_0}$$

$$\phi_- = \frac{1}{2}(\phi_1 - \phi_2)$$

$$\phi_+ = \frac{1}{2}(\phi_1 + \phi_2)$$

$$\beta_L = \frac{2\pi L I_c}{\Phi_0}$$

# QFP Potential




$$U_{\mathrm{qfp}} = U_J + U_m$$

A more accurate QFP potential includes dc-SQUID inductors

$$U_J = -2E_J \cos\phi_+ \cos\phi_-$$

$$U_m = \frac{E_j(\phi_{L_1}^2 + \phi_{L_2}^2)}{2\beta_L} + \frac{E_j\phi_q^2}{2\beta_q}$$

$$\beta_q = \frac{2\pi L_q I_c}{\Phi_0}$$

$$\phi_- = \frac{1}{2}(\phi_1 - \phi_2)$$

$$\phi_+ = \frac{1}{2}(\phi_1 + \phi_2)$$

$$\beta_L = \frac{2\pi L I_c}{\Phi_0}$$

Conservation of flux / current:

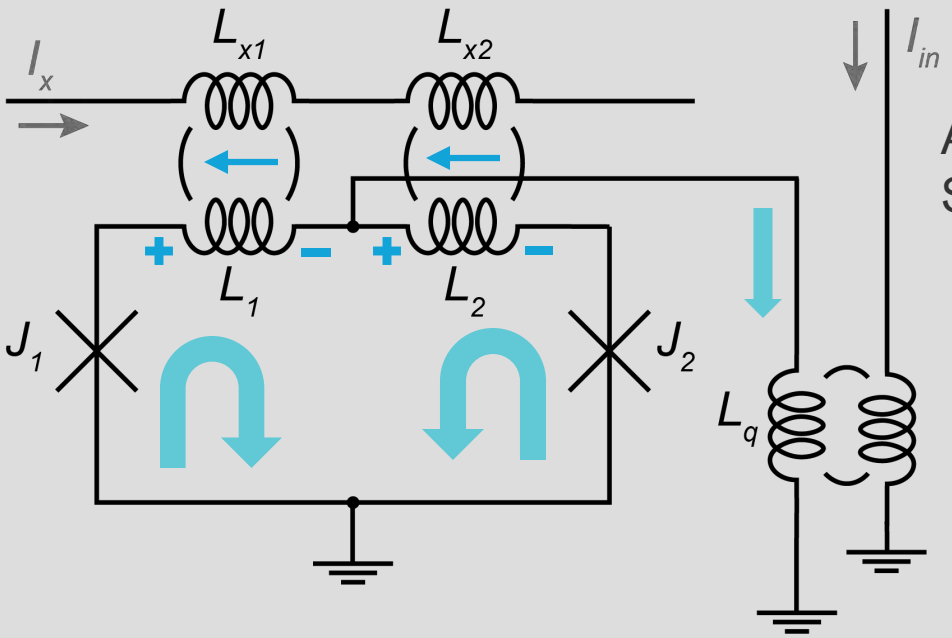
$$\phi_1 + \phi_{L_1} - \phi_x = \phi_2 + \phi_{L_2} + \phi_x = \phi_q + \phi_{\mathrm{in}}$$

$$\rightarrow \phi_{L_1} = -[(\phi_+ - \phi_q - \phi_{\mathrm{in}}) + (\phi_- - \phi_x)]$$

$$\rightarrow \phi_{L_2} = -[(\phi_+ - \phi_q - \phi_{\mathrm{in}}) - (\phi_- - \phi_x)]$$

$$\rightarrow \phi_{L_1}^2 + \phi_{L_2}^2 = 2[(\phi_+ - \phi_q - \phi_{\mathrm{in}})^2 + (\phi_- - \phi_x)^2]$$

# QFP Potential




$$U_{\mathrm{qfp}} = U_J + U_m$$

A more accurate QFP potential includes dc-SQUID inductors

$$U_J = -2E_J \cos\phi_+ \cos\phi_-$$

$$U_m = \frac{E_j(\phi_{L_1}^2 + \phi_{L_2}^2)}{2\beta_L} + \frac{E_j\phi_q^2}{2\beta_q}$$

$$\beta_q = \frac{2\pi L_q I_c}{\Phi_0}$$

$$\phi_- = \frac{1}{2}(\phi_1 - \phi_2)$$

$$\phi_+ = \frac{1}{2}(\phi_1 + \phi_2)$$

$$\beta_L = \frac{2\pi L I_c}{\Phi_0}$$

. . . after some approximations to remove $\phi_q$ and a but of algebra:

$$\frac{U_{\mathrm{qfp}}}{E_j} \approx \frac{(\phi_+ - \phi_{\mathrm{in}})^2}{2\beta_q + \beta_L} + \frac{(\phi_- - \phi_x)^2}{\beta_L} - 2\cos\phi_- \cos\phi_+$$

# QFP Potential

$$\frac{U_{\mathrm{qfp}}}{E_j} \approx \frac{(\phi_+ - \phi_{\mathrm{in}})^2}{2\beta_q + \beta_L} + \frac{(\phi_- - \phi_x)^2}{\beta_L} - 2\cos\phi_- \cos\phi_+$$