集中講義令和8年2月24日～26日

コンピュータシステム入門

Day 2: デジタル回路の基礎 (2)

講師　陳　オリビア　（大学院システム情報科学研究院）
TA　GPT-5 Thinking (OPEN AI)
　　Gemini 2.5 pro (Google)

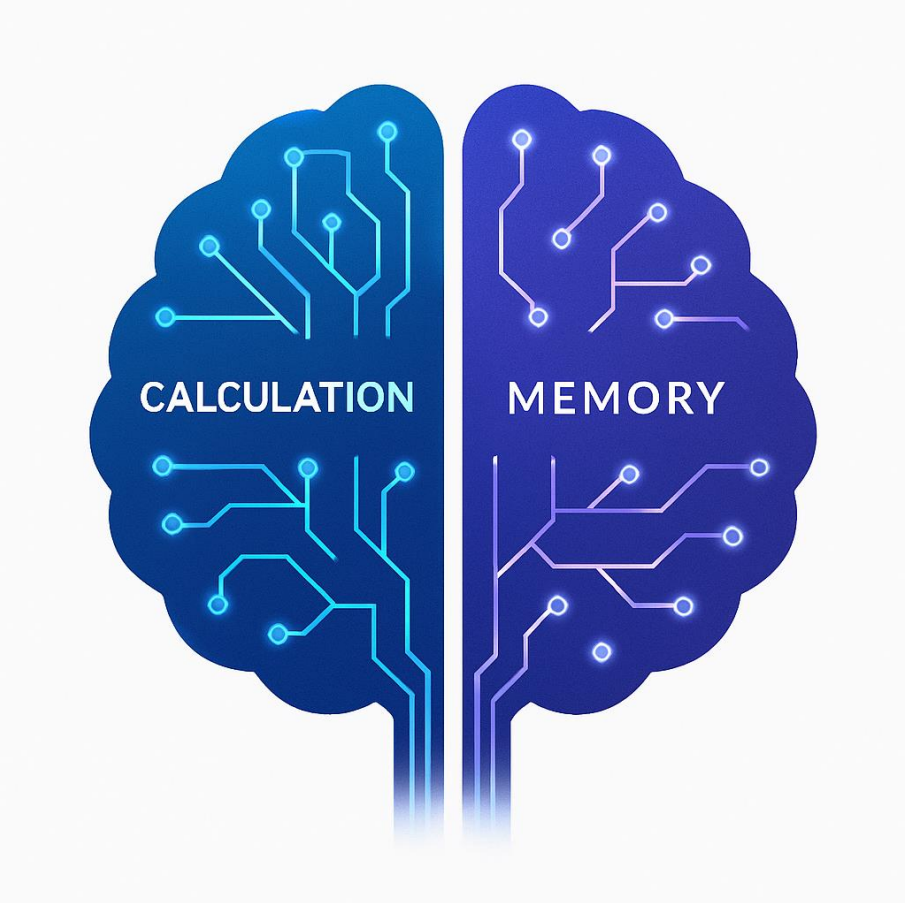
CONTENTS

今日のスケジュール

	時間帯	モジュール	タイプ
午前中	10:00 ~ 11:00	デジタル回路の基礎（２）	講義
	11:00 ~ 11:15	Coffee Break	
	11:15 ~ 12:30	コンピューターアーキテクチャ	講義
	12:30 ~ 13:30	Lunch Break	
午後	13:30 ~ 14:30	性能評価とチップ設計	講義
	14:30 ~ 15:00	Verilogで回路を記述してみよう	演習
	15:00 ~ 15:15	Coffee Break	
	15:15 ~ 17:00	Verilogで回路を記述してみよう（継続）	演習

CONTENTS

デジタル回路の基礎 (2)



コンピュータはどうやって物事を
「記憶」するのか？

CONTENTS

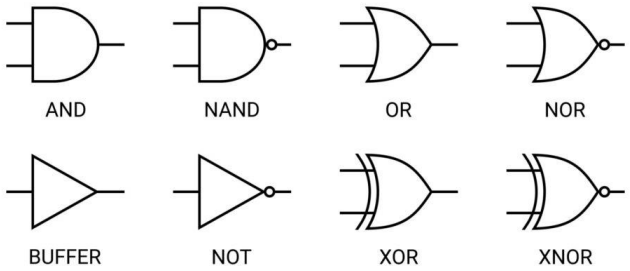
モジュール４：デジタル回路の基礎（２）

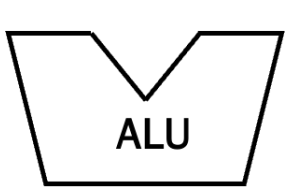
CONTENTS

モジュール４：デジタル回路の基礎（２）

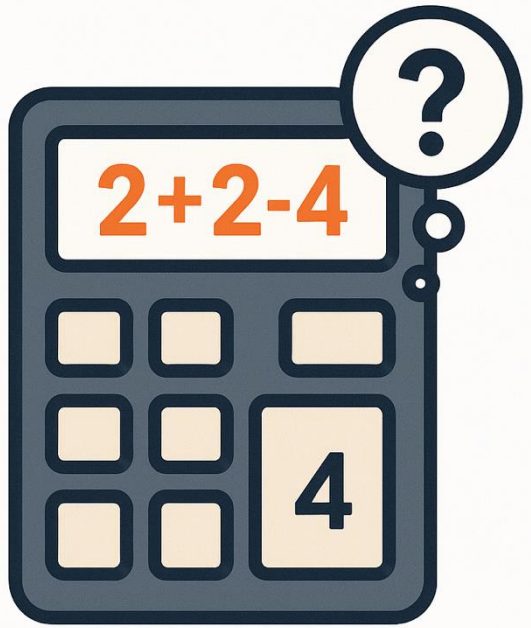
前回の復習：組み合わせ回路

前回のまとめ：組み合わせ回路





FA



- 入力が決まれば、出力は常に一意に決まる。
- 例：加算器（計算はできる）
- **しかし…過去の入力や結果を覚えておくことができない！**

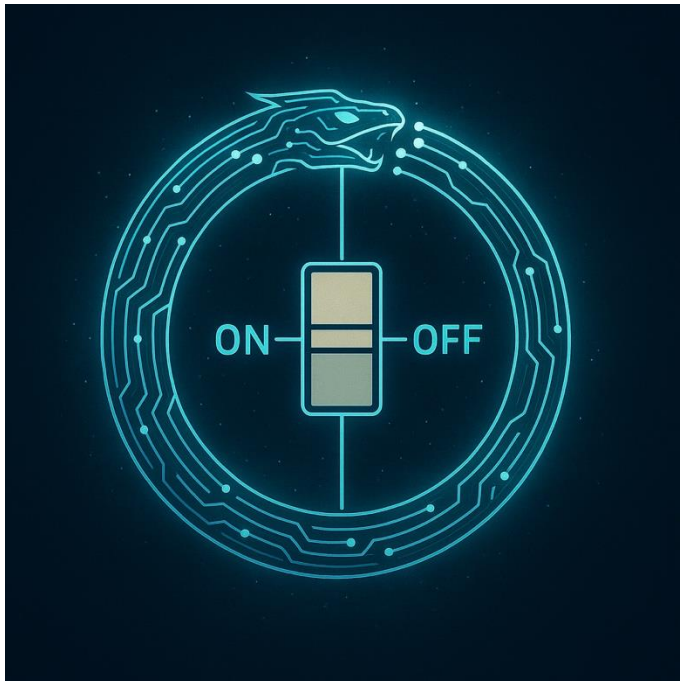
今日の核心的な問い

計算結果の保存

現在の状態の維持　？　次の動作の準備

どうすれば、回路に「状態」を記憶させられるのか？

基本アイデア：フィードバック

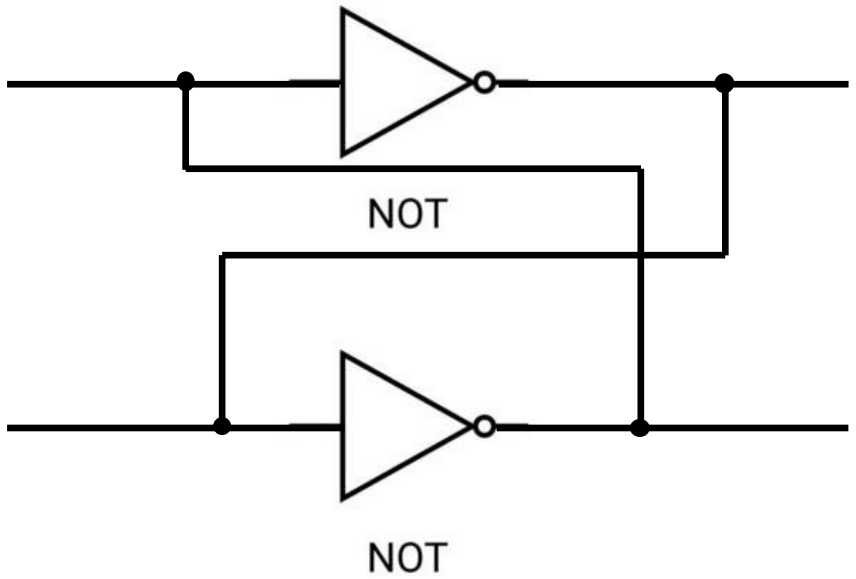


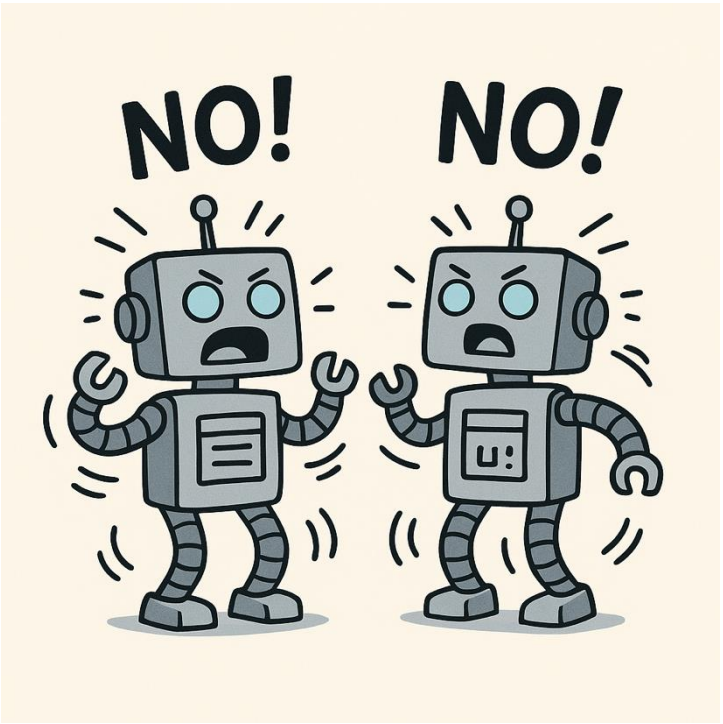
もし、回路の「出力」を、再び「入力」に戻したら…？

CONTENTS

モジュール４：デジタル回路の基礎（２）

不安定な記憶回路：NOT×2のフィードバック

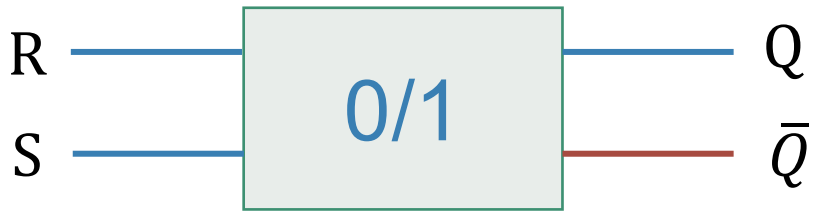




- **不安定な記憶回路**
- ONとOFFを高速で繰り返し、状態が安定しない。
- これでは記憶装置として使えない…

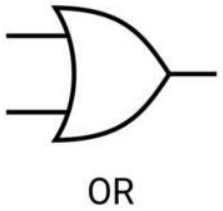
安定した記憶回路：SRラッチ

安定した記憶回路：SRラッチ

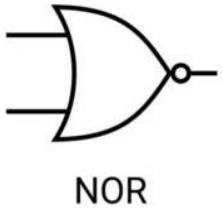


S	R	Q	$\bar{Q}$
0	0	保持	保持
0	1	0	1
1	0	1	0
1	1	0	0

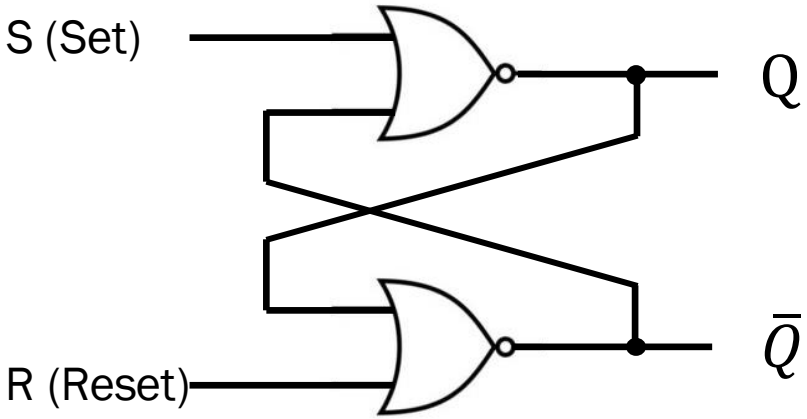
安定した記憶回路：SRラッチ



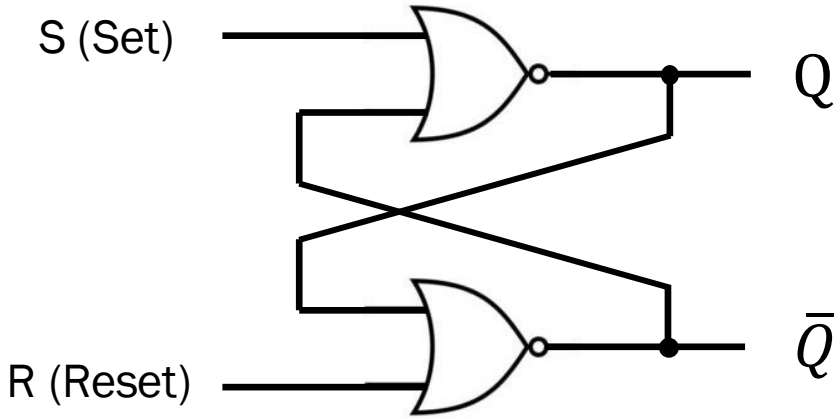
A	B	A+B
0	0	0
1	0	1
0	1	1
1	1	1



A	B	$\overline{A+B}$
0	0	1
1	0	0
0	1	0
1	1	0



SRラッチの動作①：セット



参考：NORの真理値表

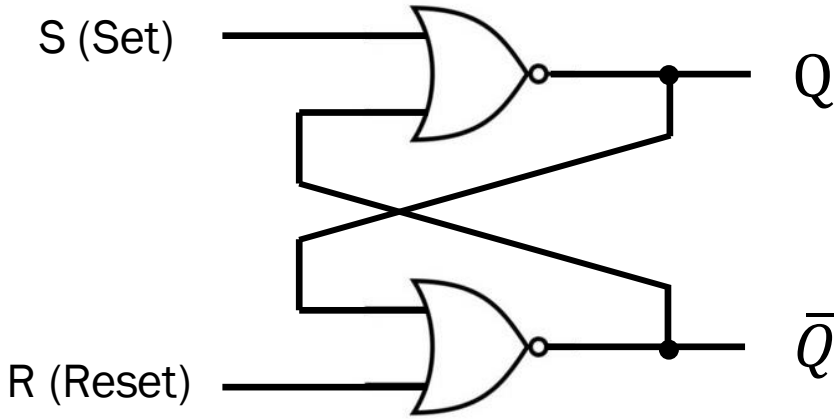
A	B	$\overline{A+B}$
0	0	1
1	0	0
0	1	0
1	1	0

- **Set (S) = 1：記憶を「1」にする**
- Sに「1」を入力すると、出力Qが「1」になる

SRラッチの真理値表

S	R	Q	$\bar{Q}$
1	0	1	0

SRラッチの動作②：リセット



参考：NORの真理値表

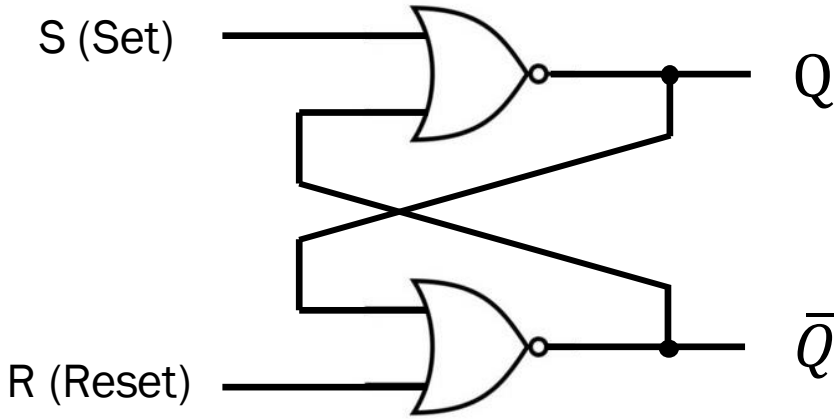
A	B	$\overline{A+B}$
0	0	1
1	0	0
0	1	0
1	1	0

- **Reset (R) = 1：記憶を「0」にする**
- Rに「1」を入力すると、出力Qが「0」になる

SRラッチの真理値表

S	R	Q	$\bar{Q}$
0	1	0	1
1	0	1	0

SRラッチの動作③：記憶の保持



参考：NORの真理値表

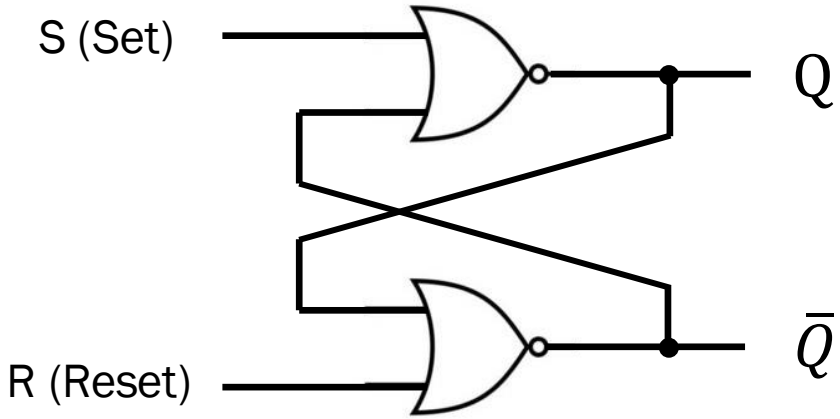
A	B	$\overline{A+B}$
0	0	1
1	0	0
0	1	0
1	1	0

SRラッチの真理値表

S	R	Q	$\bar{Q}$
0	0	保持	保持
0	1	0	1
1	0	1	0

- **S=0, R=0：記憶を「保持」する**
- 両方の入力が0のとき、回路は直前の状態を保ち続ける。
  これが「記憶」の正体！

SRラッチの問題点



参考：NORの真理値表

A	B	$\overline{A+B}$
0	0	1
1	0	0
0	1	0
1	1	0

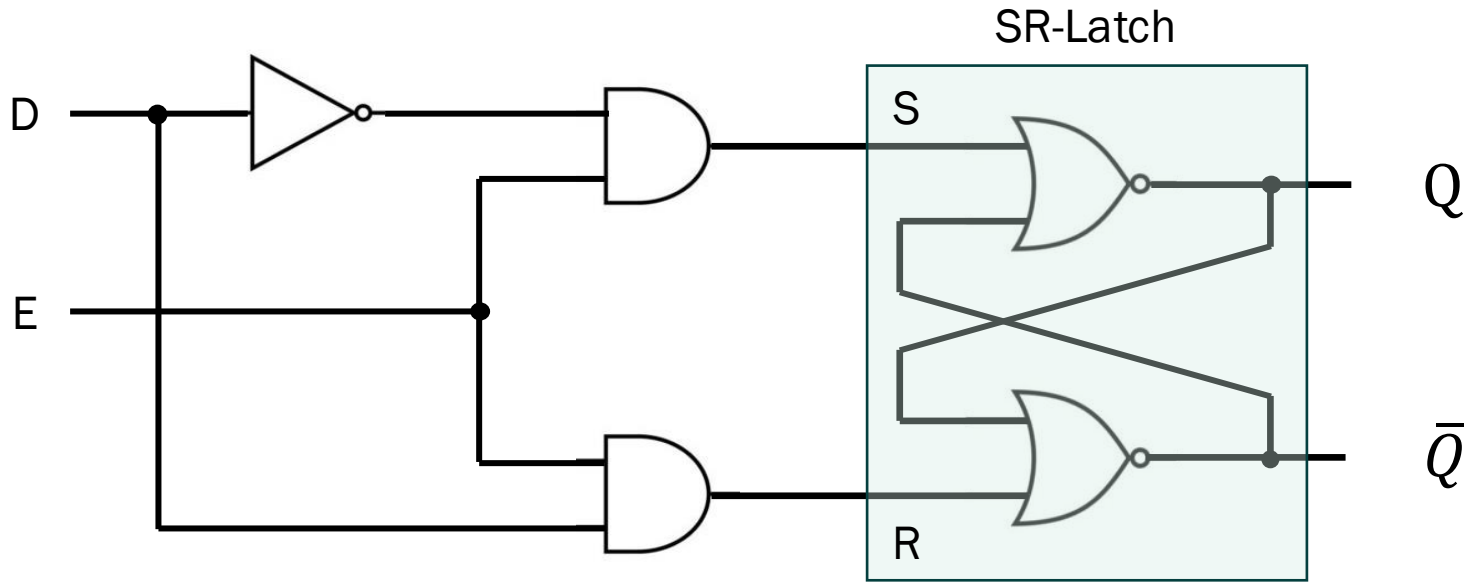
SRラッチの真理値表

S	R	Q	$\bar{Q}$
0	0	保持	保持
0	1	0	1
1	0	1	0
1	1	0	0

- **問題点：S=1, R=1の入力**
- 出力が矛盾した状態になり、次にS, Rが0に戻ったときに状態が不安定になる。

Dラッチ：より安全な記憶素子

- **Dラッチ：入力を1つにして問題を解決**
- データ入力(D)と、書き込み許可(E)の2つの入力を持つ

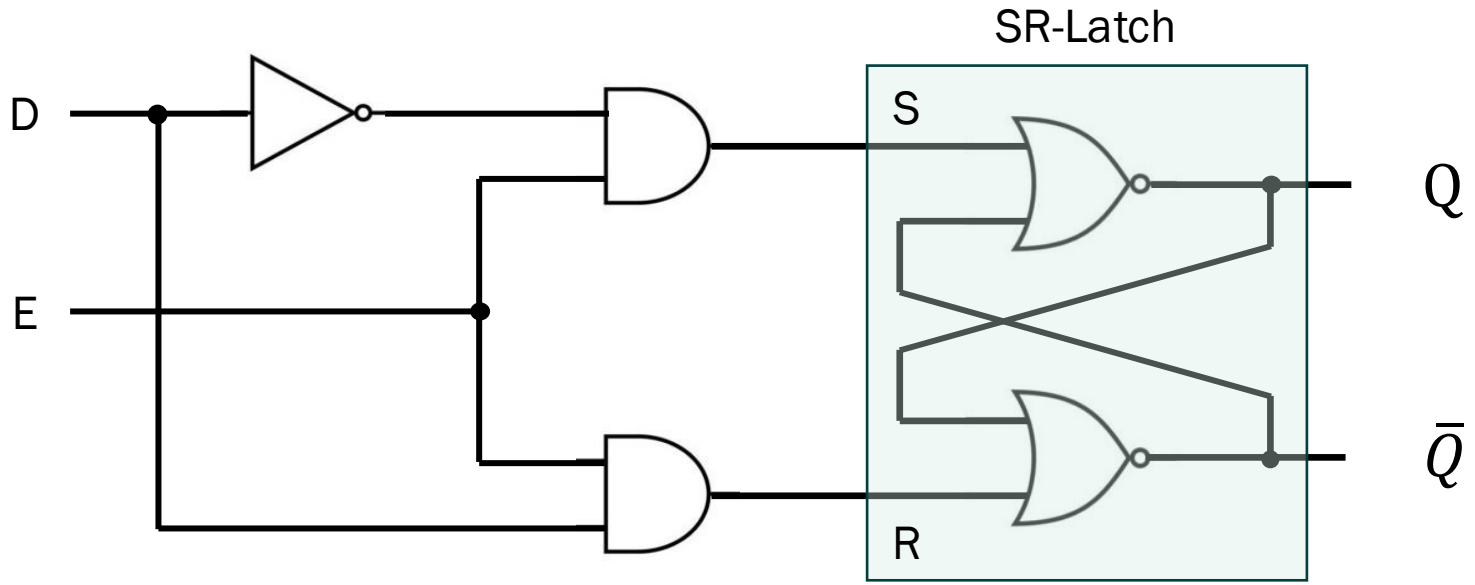


Dラッチの真理値表

D	E	Q	$\bar{Q}$

Dラッチ：より安全な記憶素子

- Enable = 1 のとき：データ(D)を書き込む（QはDと同じになる）
- Enable = 0 のとき：データを保持する（Qは変化しない）



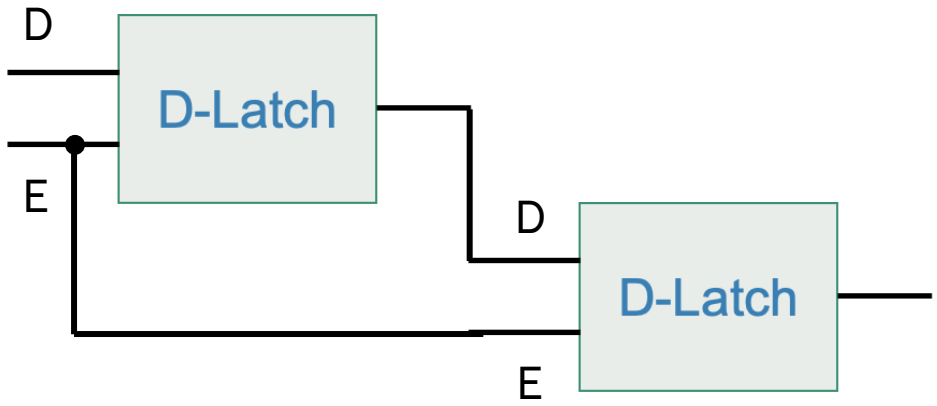
Dラッチの真理値表

E	D	Q	$\bar{Q}$
0	0	保持	保持
0	1	保持	保持
1	0	0	1
1	1	1	0

CONTENTS

モジュール４：デジタル回路の基礎（２）

タイミングの問題

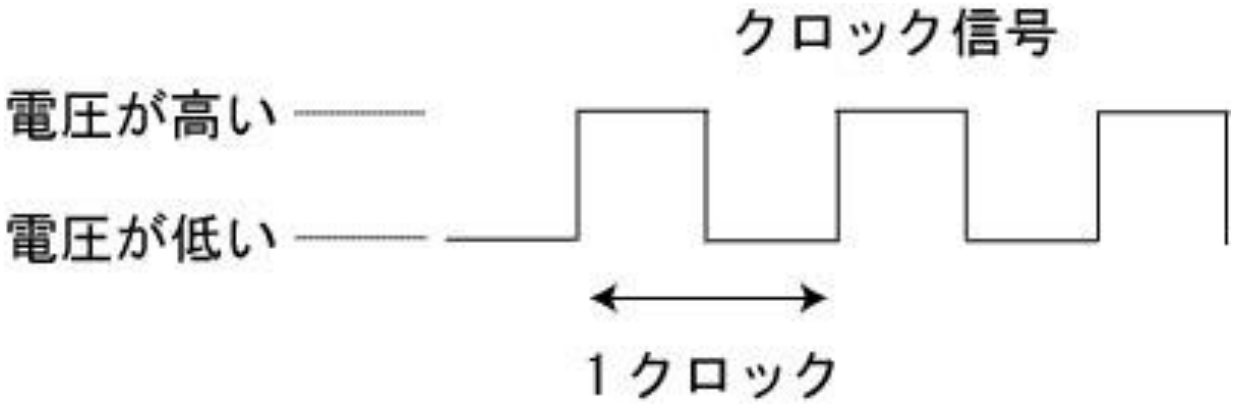


複数の記憶素子を連携させるには、どうすればいい？

解決策：クロック信号

- **クロック信号：コンピュータの「指揮者」**
- 全ての記憶素子に、同じタイミングで「今だ！」という信号を送る。

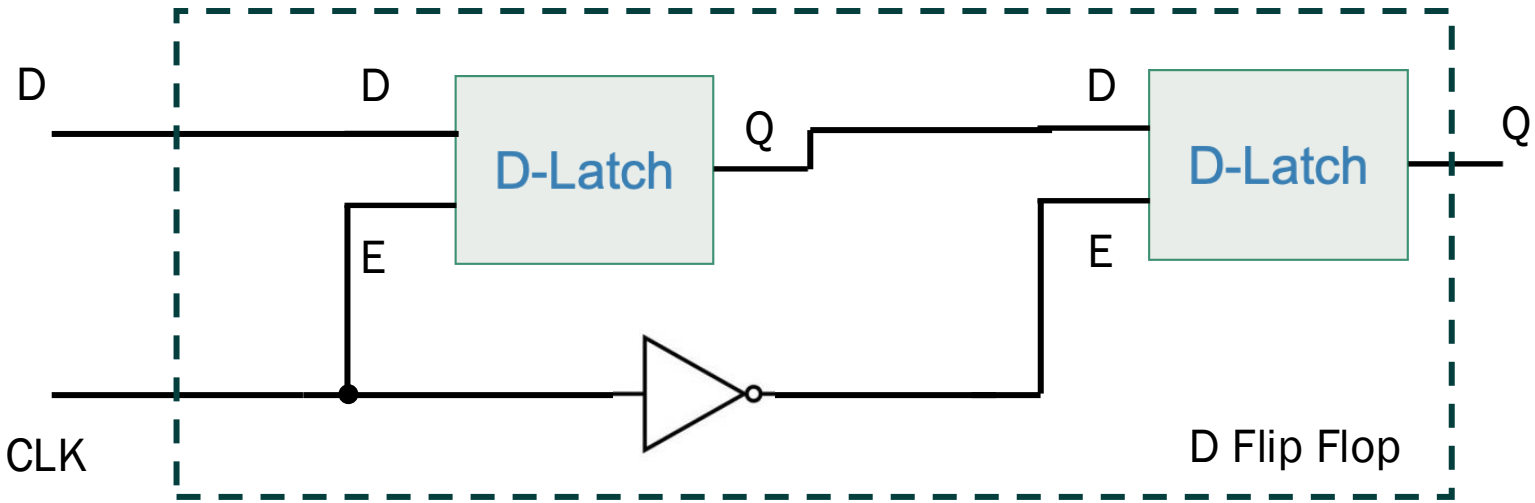
クロックの波形

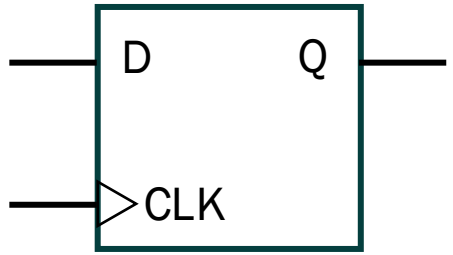


クロック信号は、ONとOFFを正確なリズムで繰り返す電気信号。

Dフリップフロップ：記憶の完成形

- **Dフリップフロップ (DFF)：現代の記憶素子の基本**
- クロックがONになる**瞬間**（立ち上がりエッジ）だけ、データを更新する



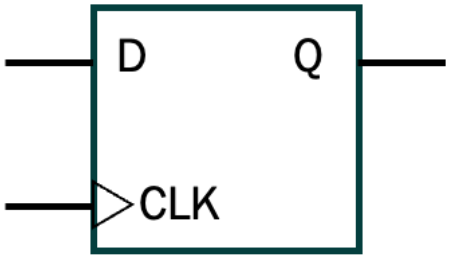


CLK	D	Q	$\bar{Q}$
LOW	0	Latch	Latch
LOW	1	Latch	Latch
HIGH	0	0	1
HIGH	1	1	0

CONTENTS

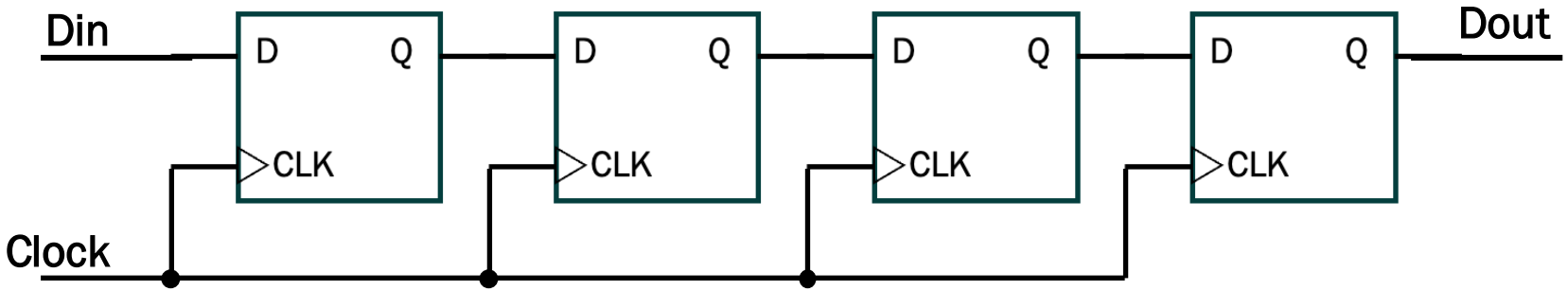
モジュール４：デジタル回路の基礎（２）

1ビットから複数ビットへ：レジスタ



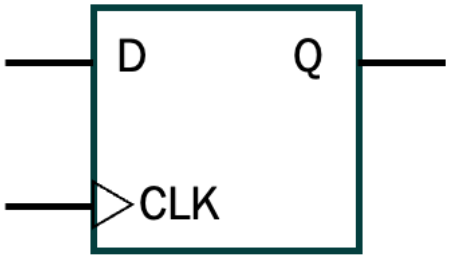
DFF: 1ビットメモリ

- **レジスタ：複数のビットをまとめて記憶する装置**
- CPU内部で、計算結果などを一時的に保持するために使われる。



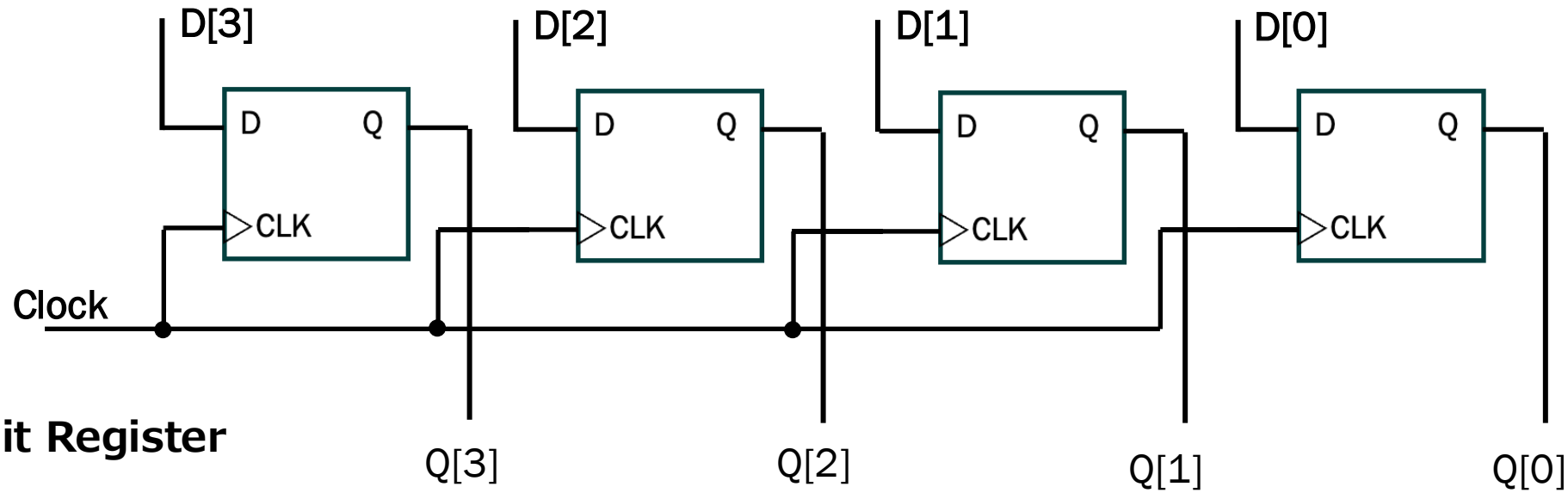
4-bit Shift Register

1ビットから複数ビットへ：レジスタ

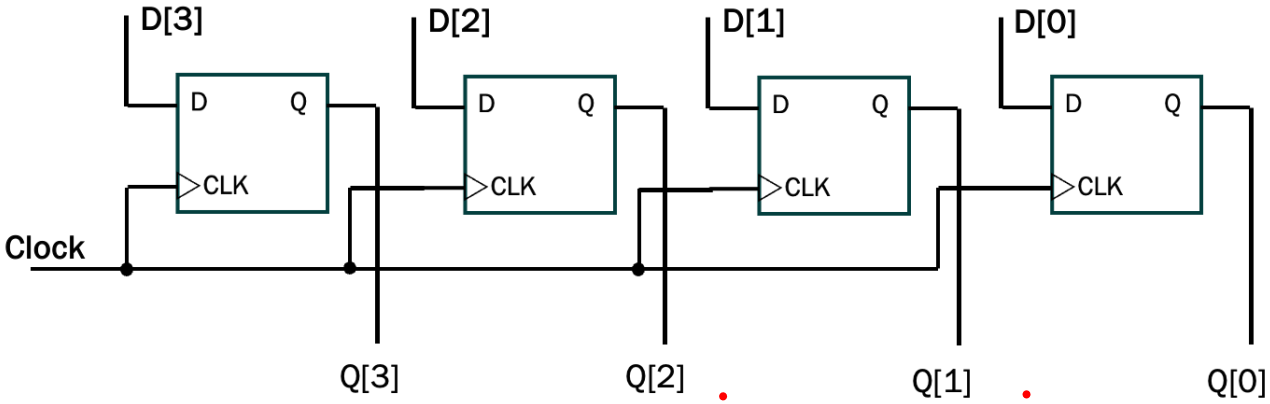


- **レジスタ：複数のビットをまとめて記憶する装置**
- CPU内部で、計算結果などを一時的に保持するために使われる。

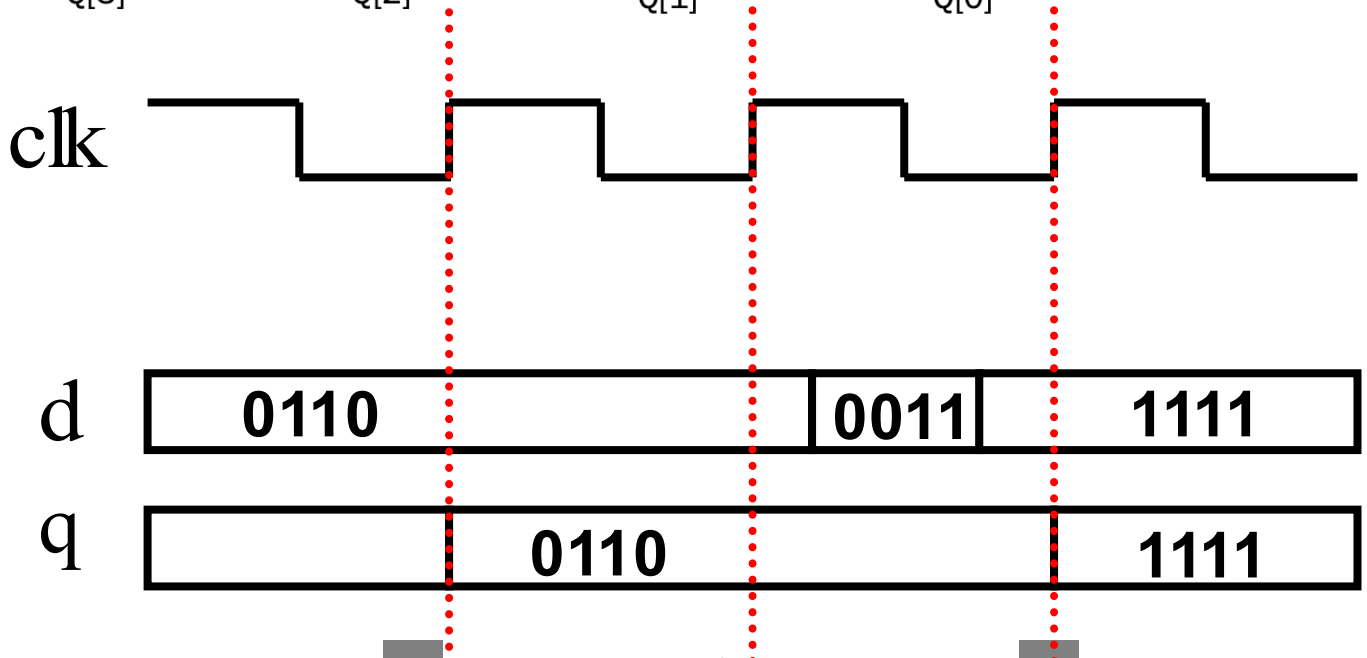
DFF: 1 ビットメモリ



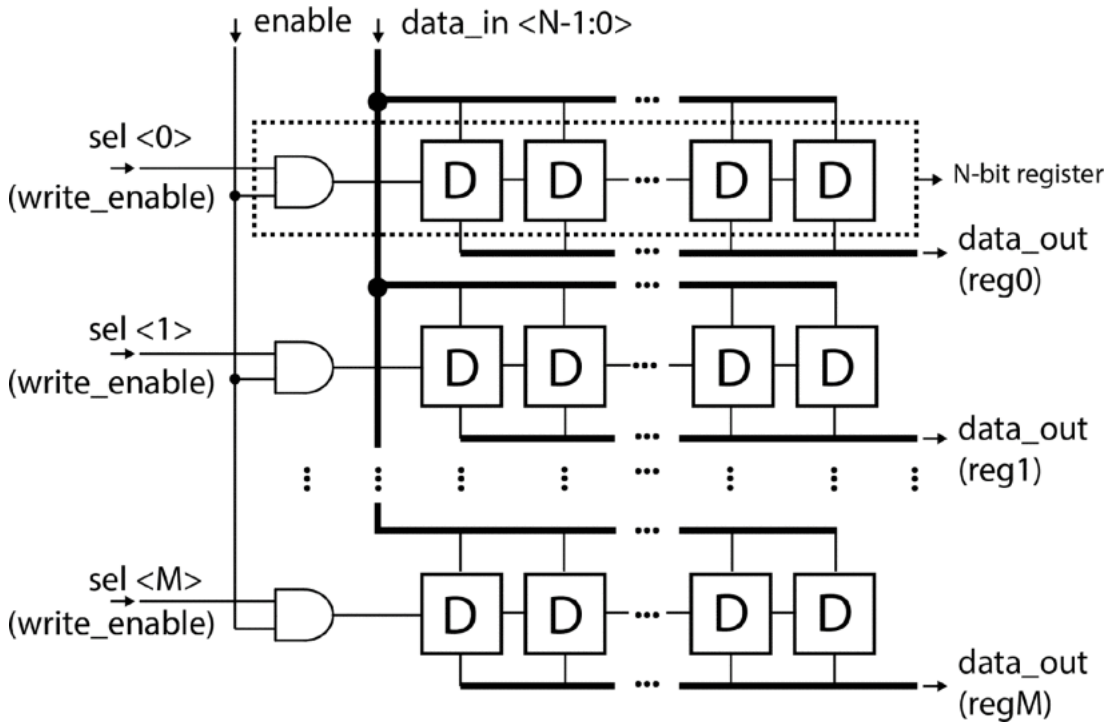
レジスタの動作



クロックが一回「カチッ」と鳴るたびに、
4ビットのデータがまとめて記憶される

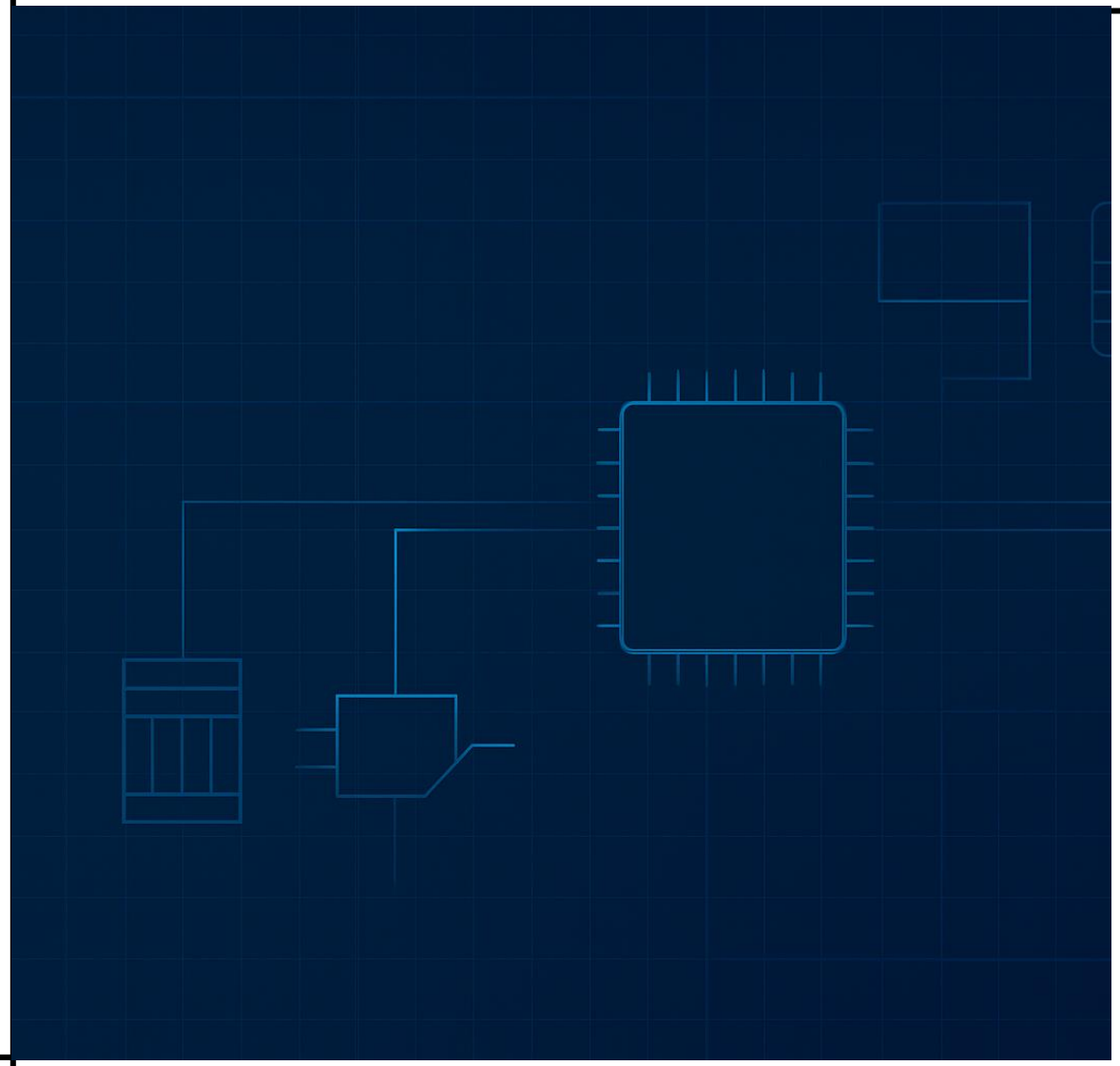


より大きな記憶装置：メモリ(RAM)



- **メモリ(RAM)：膨大な数のレジスタの集合体**
- 数百万、数億の記憶場所に、それぞれ「住所（アドレス）」が割り振られている

次回予告



次回：コンピューターアーキテクチャ