Didapro 9, Le Mans

Un simulateur logique pensé pour l'enseignement

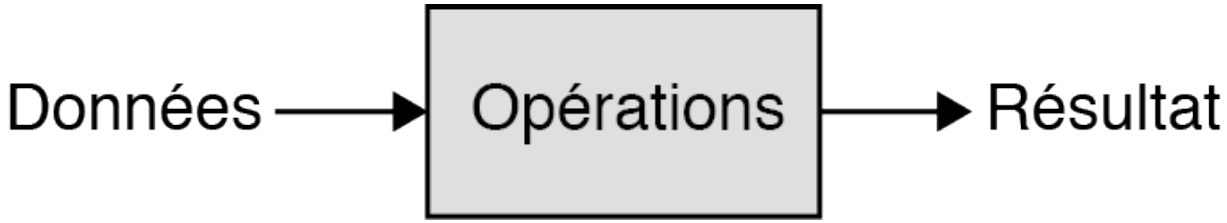
Jean-Philippe Pellet & Gabriel Parriaux

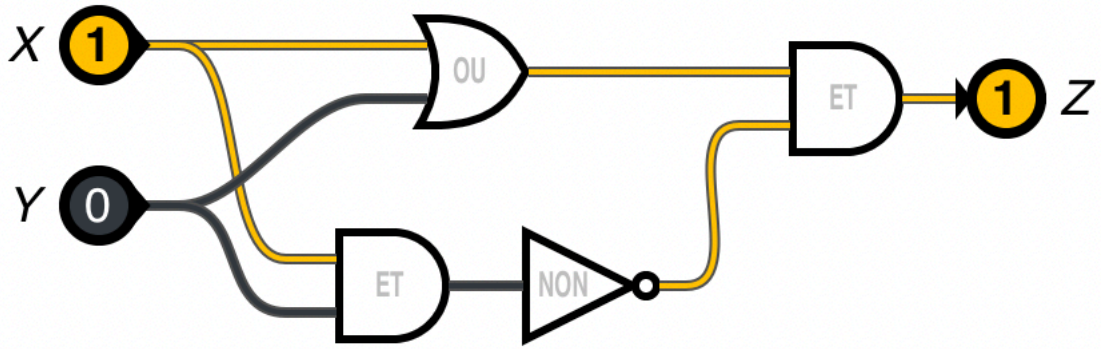
HEP Vaud, Lausanne, Suisse

18 mai 2022

Pourquoi les systèmes logiques?

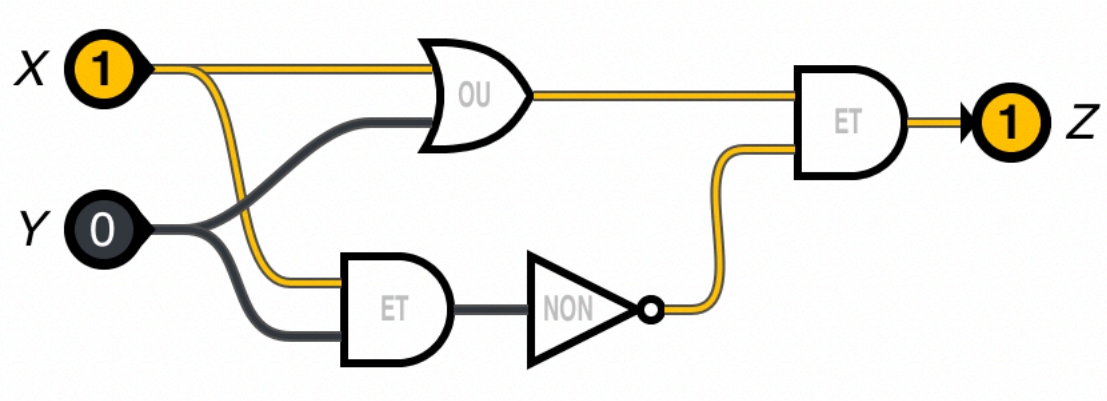
- **Un système logique simple** (combinatoire) **représente extrêmement bien le modèle typique d'un système de traitement de l'information**

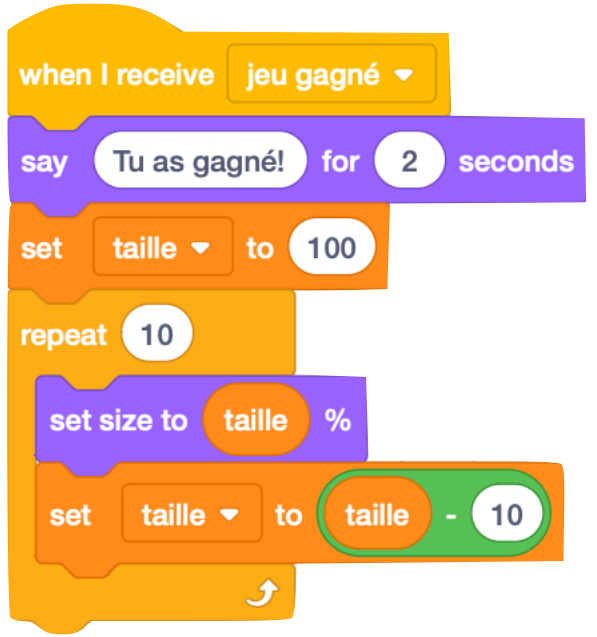




Pourquoi les systèmes logiques?

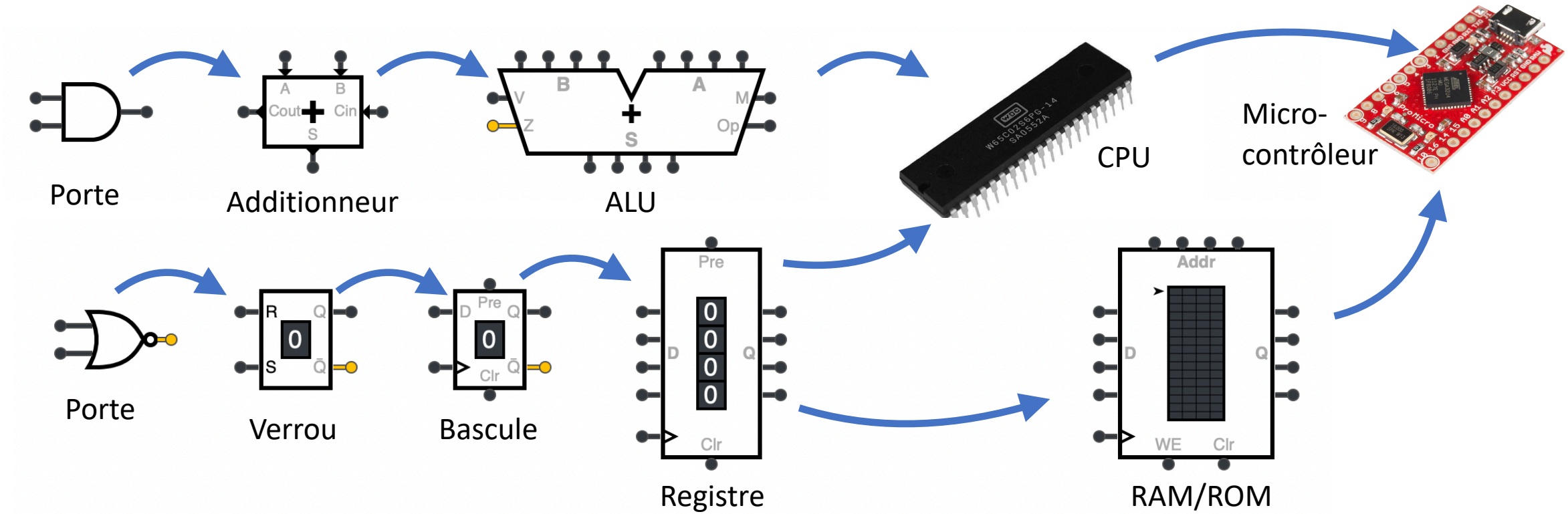
- **Un système logique est un langage graphique formel exécutable directement «en interne»** (dans le langage lui-même)
  - Scratch, pour éviter les erreurs de syntaxe
    - Les erreurs sont de «vraies» erreurs de raisonnement
  - Ici, idem: pas non plus d'erreur de syntaxe dans le langage des circuits logiques





Pourquoi les systèmes logiques?

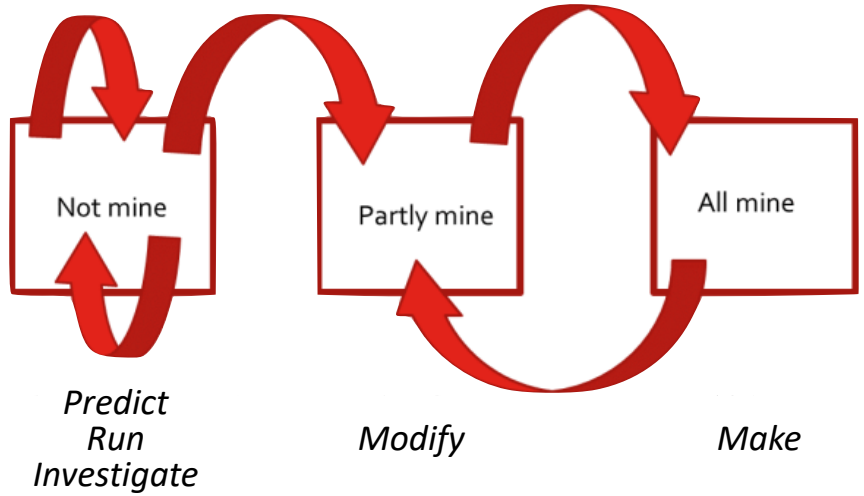
- **Travailler avec des composants logiques de plus en plus capables illustre très bien les principes d'abstraction, de modularisation, de décomposition**
  - ✦ Plus facilement qu'en faisant de la programmation, parce qu'on commence par le bas



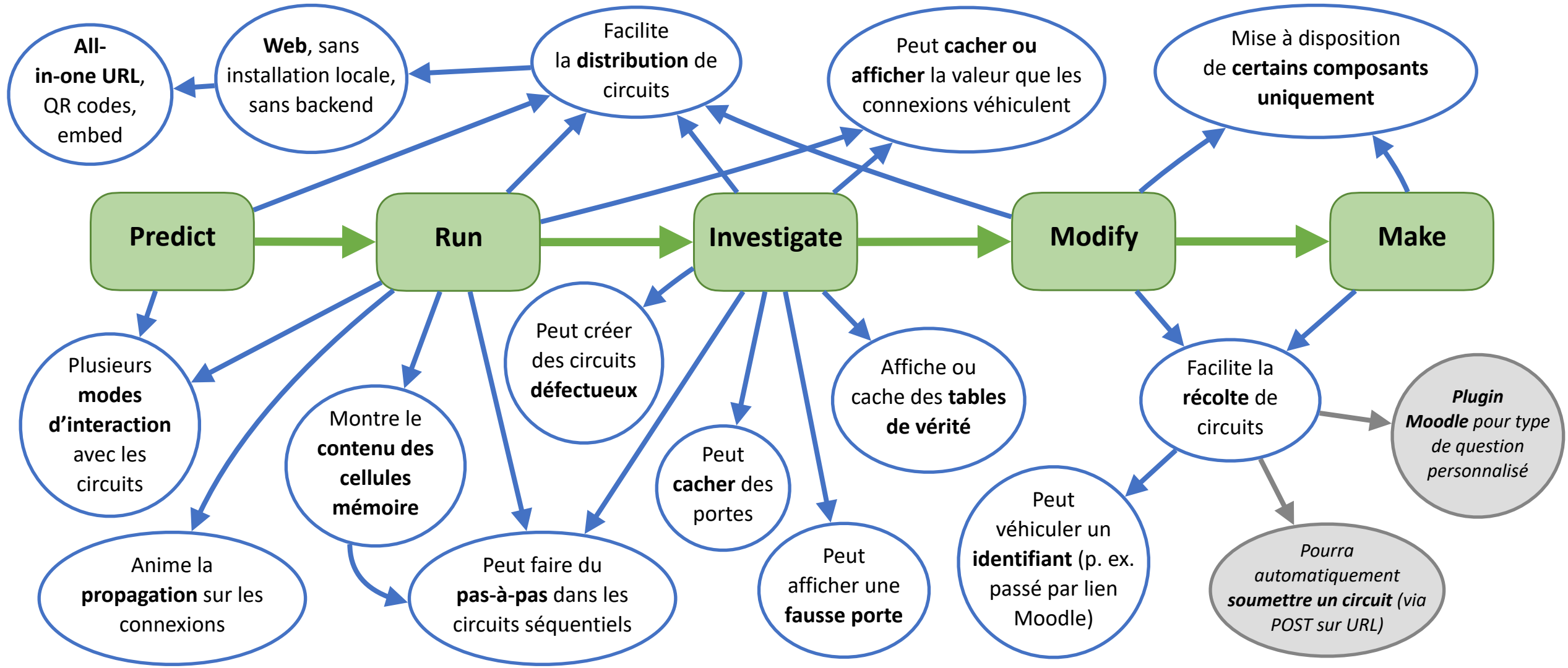
Pourquoi les systèmes logiques?

- **Se prête bien** (avec la bonne plateforme) **à de nombreuses «techniques pédagogiques»**
- **PRIMM**
  - *Predict:* sans jouer avec le système, dire ce qu'il fait
  - *Run:* faire tourner un système et vérifier les prédictions
  - *Investigate:* annoter, tracer, débugger un peu, nommer les variables, etc.
  - *Modify:* changer/compléter le code avec de petits puis moyens changements
  - *Make:* création d'un petit système complet
- **Démos!** ⇒ **https://bit.ly/logic-didapro**

https://primmportal.com

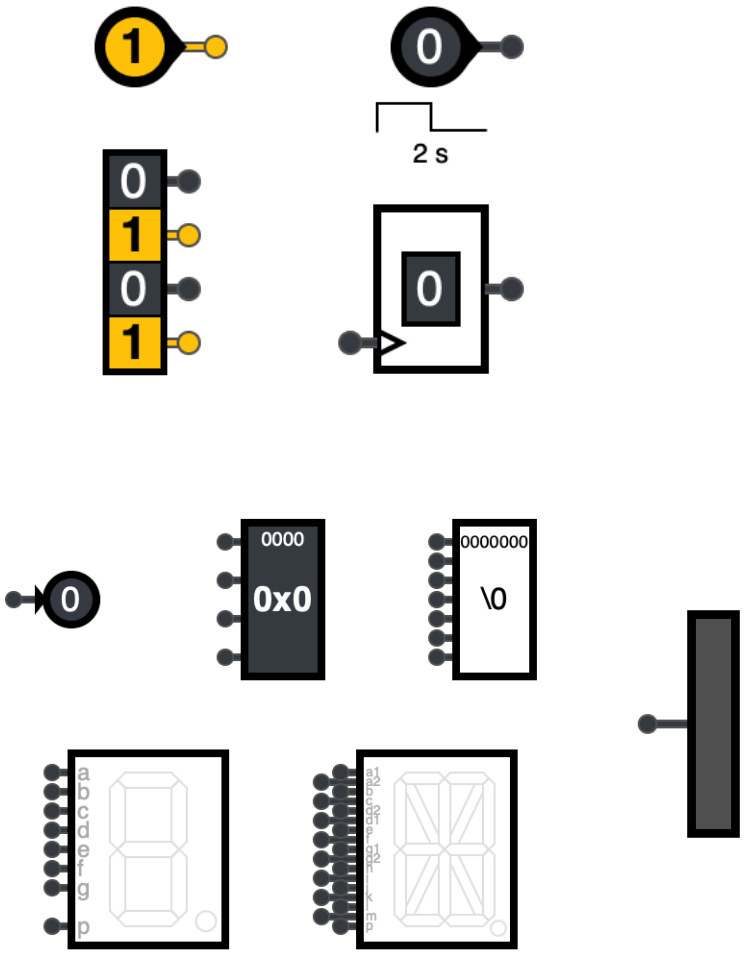


PRIMM sur un simulateur logique — *démos*



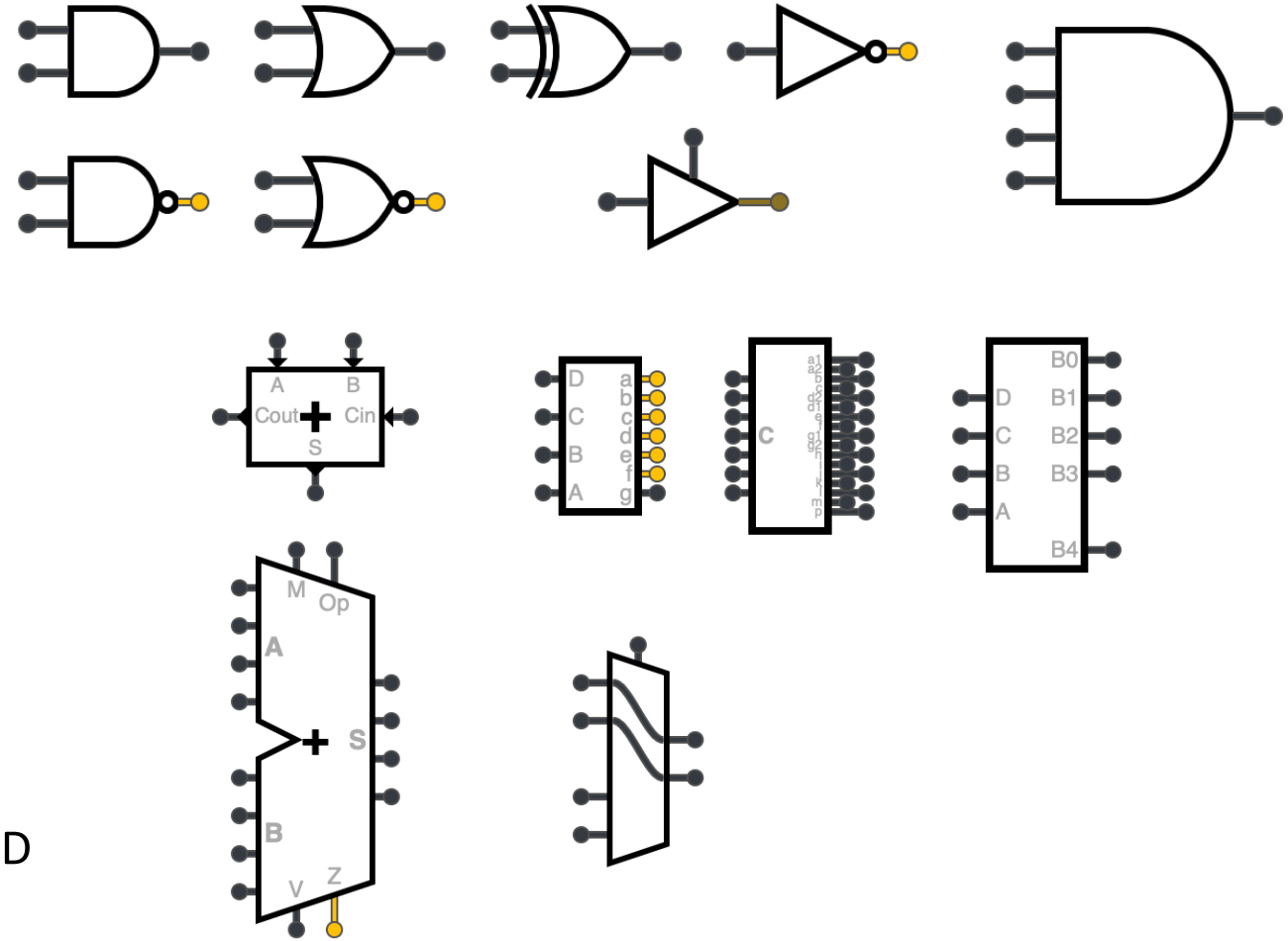
Composants: entrées/sorties

- **Entrées**:
  - 1 bit (poussoir, commutateur),
  - 4 bits (dip switches)
  - Horloge
  - Aléatoire (sur coup d'horloge)
- **Sorties:**
  - 1 bit (standard ou «bande lumineuse»)
  - 4 bits (hexa, décimal signé ou non)
  - 7 bits (ASCII)
  - Afficheur 7 segments, 16 segments



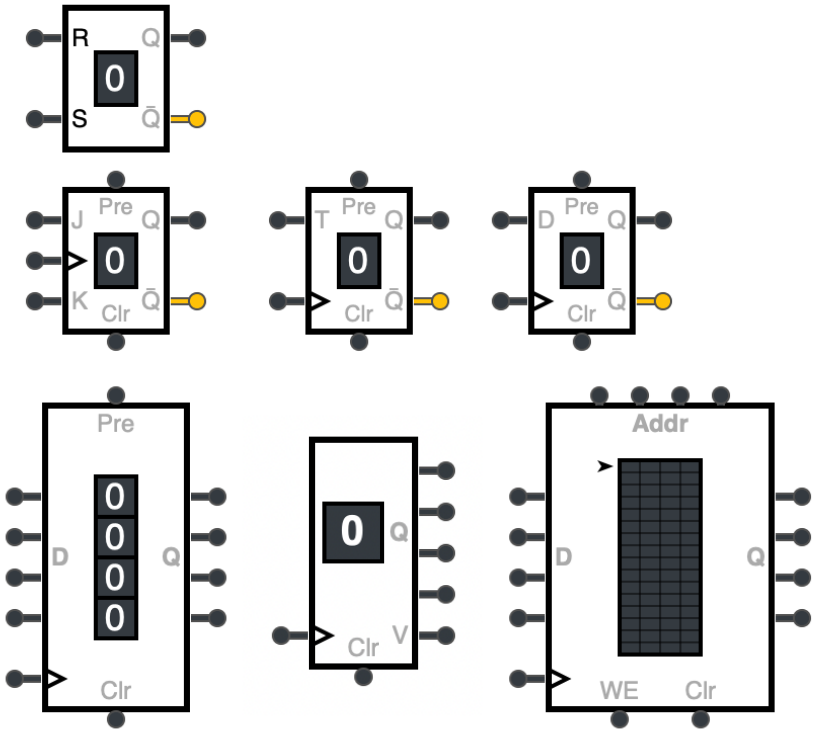
Composants: logique combinatoire

- **Portes**:
  - AND, OR, XOR, NOT
  - NAND, NOR
  - IMPLY, NIMPLY
  - Buffer à trois états (haute impédance)
  - 2, 3 ou 4 entrées
- **Composants combinatoires**
  - Additionneur
  - ALU (addition, soustraction, AND, OR)
  - Multiplexeurs (8/4/2 vers 4/2/1)
  - Décodeurs 7 segments, 16 segments, BCD



Composants: logique séquentielles

- **Composants séquentiels**
  - Verrou SR
  - Bascules: JK, T, D
  - Registre 4 bits
  - Compteur
  - RAM de 16 × 4 bits



- **Pas encore disponible**
  - Registre à décalage, bus, ROM, composants personnalisés

Merci!

logic.modulo-info.ch

https://github.com/jppellet/Logic-Circuit-Simulator

Activités en réserve

1. Décodeur 2 bits

Depuis deux entrées représentant un nombre entre 0 et 3, créer quatre sorties dont toujours exactement une est active: celle dont l'index correspond au nombre représenté par l'entrée.

3. Mini-ALU

Construire une mini-ALU de 4 bits avec un bit de contrôle pour choisir entre addition et soustraction à l'aide de 4 additionneurs et de portes logiques supplémentaires.

2. Code de Hamming(7, 4)

À partir de 4 entrées de données D1 à D4, générer les 3 bits de parité P1 à P3 d'un code de Hamming(7, 4).

Bit #	1	2	3	4	5	6	7
Transmitted bit	p_1	p_2	d_1	p_3	d_2	d_3	d_4
p_1	Yes	No	Yes	No	Yes	No	Yes
p_2	No	Yes	Yes	No	No	Yes	Yes
p_3	No	No	No	Yes	Yes	Yes	Yes

4. Compteur

À l'aide de 4 bascules D, construire un compteur qui compte de 0 à 15 par incrément de 1 à chaque coup d'horloge.