Safe autonomy

Lecture-2

Logic & Formal Spec.

Logic

- Systematic method for inference.
- Boolean Algebra.

0 and 1, $\wedge$, $\vee$, $\neg$, combination $\otimes$, $\oplus$,

$0 \wedge 1 = 1 \wedge 0 = 0 = 0 \wedge 0$; $1 \wedge 1 = 1$

$0 \vee 1 = 1 \vee 0 = 1 = 1 \vee 1$; $0 \vee 0 = 0$.

$\neg 0 = 1$; $\neg 1 = 0$

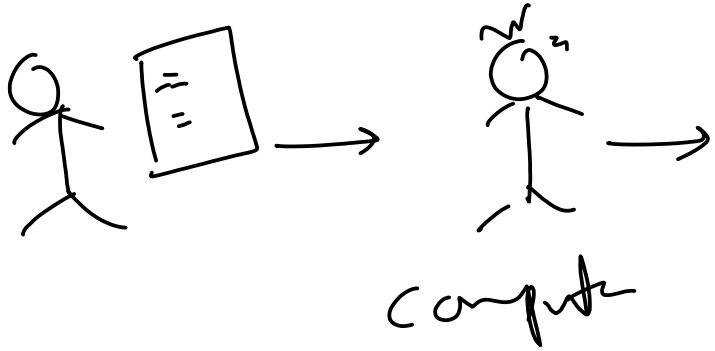
Transistors : Storing & Comm. Boolean Valu.

Vaccme Tubes

'2' things. → 0 & 1

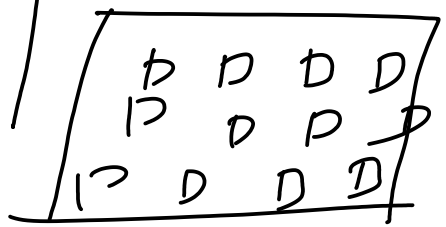
⊗ Inputs → Circuits → Output.

Baker, Carpenter, Plumber, Computer.



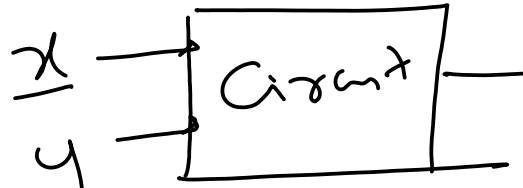
1900s

Los Alamos



Richad Feyn

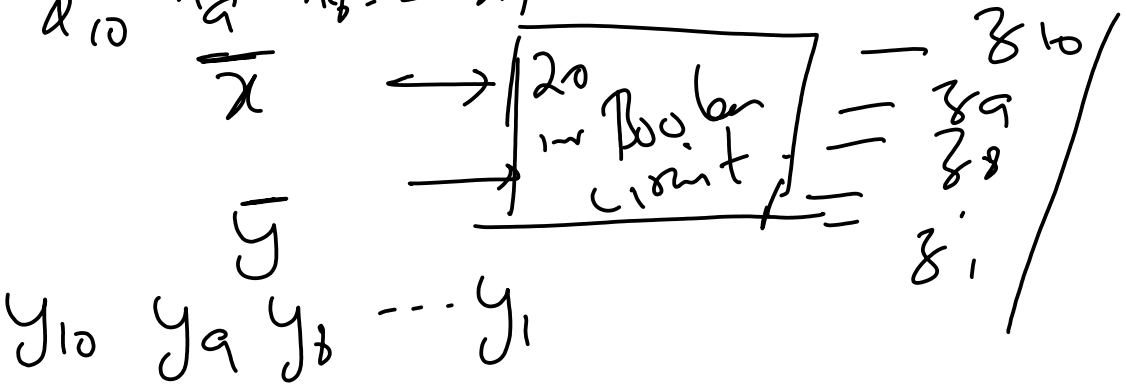
Automatic Computer:



$z = x/y$

x, y, z are +ve integers

all of them 10 bits → 1024



"Function Proper"

$x = N_{m_1}$ / N_{m_1}, 0, ..., 1024

$y = N_{m_2}$ / N_{m_2}, 0, ..., 1024

$z = \text{Output}(N_{m_1}, N_{m_2})$, $z = ?$ N_{m_1} / N_{m_2}

$(1024)^2$ operations

$z = ?$ N_{m_1} / N_{m_2}

$k = N_{m_1}, N_{m_2}$

$k = N_{m_1} / k < N_{m_2}$

"Testing": Running circuit for a few instances

"Proof" of correctness.

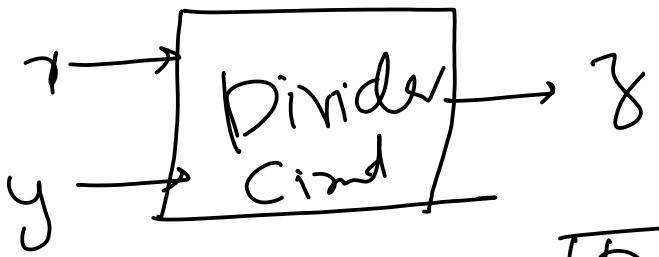
Automatic Computation Engine: Effort to "define"/"check" correctness is much more than designing the circuit/Engine.

"Rigorous Specification":

→ Expressing requirements of a "circuit/Engine" in mathematically precise manner.

Partial Specification $(1024)^2$

$z = DC(x, y)$ | Boolean formula

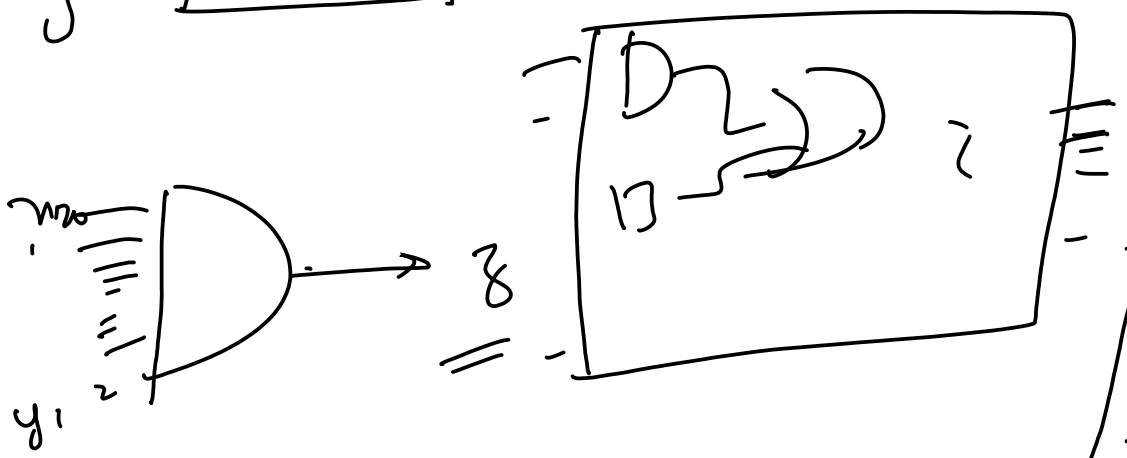


$z_1 \leftarrow (x_1 \wedge \cdots \wedge y_1)$

$z_2 \leftarrow (\cdots)$

$z_3 \leftarrow \cdots$

$z_{10^2} \leftarrow (\cdots)$



If $x \geq y$ then $z \geq 1$

$z \leq x$

$x = 1024$

$y = 1$

$z = 1024$

$z_2 = x_0 \wedge \cdots \wedge y_1$

Wrong

$x = 100, y = 62$

$z = 1$

$x \% 4 = 0 \wedge y \% 2 = 0 \wedge y \% 4 \neq 0 \Rightarrow z \% 2 = 0$

$(y \neq 0) \Rightarrow x \geq y \times z \;,\; x \leq y \times z + y^{-1}$

Rigorous Spec $(z = DC(x,y) \wedge y \neq 0) \Rightarrow (z \leq x)$

$(\phi(x,y,z) \wedge y \neq 0) \Rightarrow (z \leq x)$

$(\phi(x,y,z) \wedge y \neq 0) \Rightarrow ((x \geq y) \Rightarrow (z \geq 1))$

Machine readable.

Boolean formula

"Formal" Specification

→ Mathematically precise statements that express a relationship between Input & Output.

⇒ Spec(Input, Output)

Reactive Systems

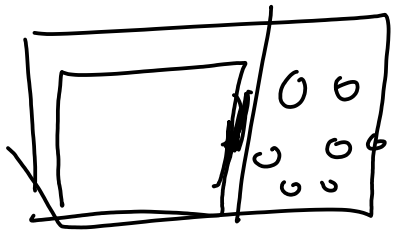
→ Never stop functioning: Operating System.

Specification is not about the goal, but the process.

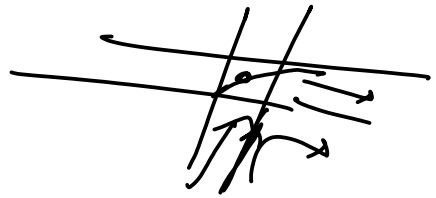
Resources ← Pgms

Traffic light

Segments of road ← vehicles

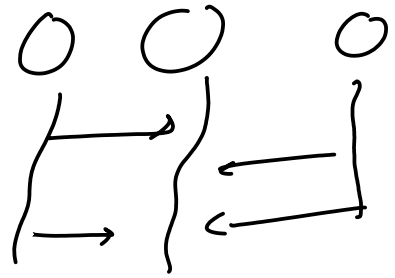


Reactive System



1977: Amir Pnueli

Linear Temporal Logic ← Philosophy: Temporal Logics



Foundation: For a reactive System, "truth" changes with time.

Time: Discrete time

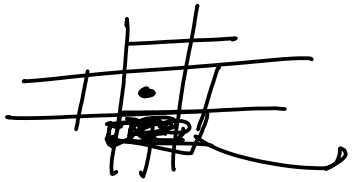
Boolean : $\mathbb{N} \longrightarrow \mathbb{B} \{0,1\}$

a	a	a	a	a	a	...
=	0	0	1	1	0	...

Prop / Boolean Logic

a = 0
b = 1
c = 0

| Static

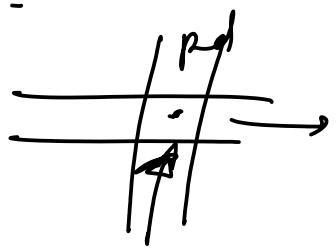


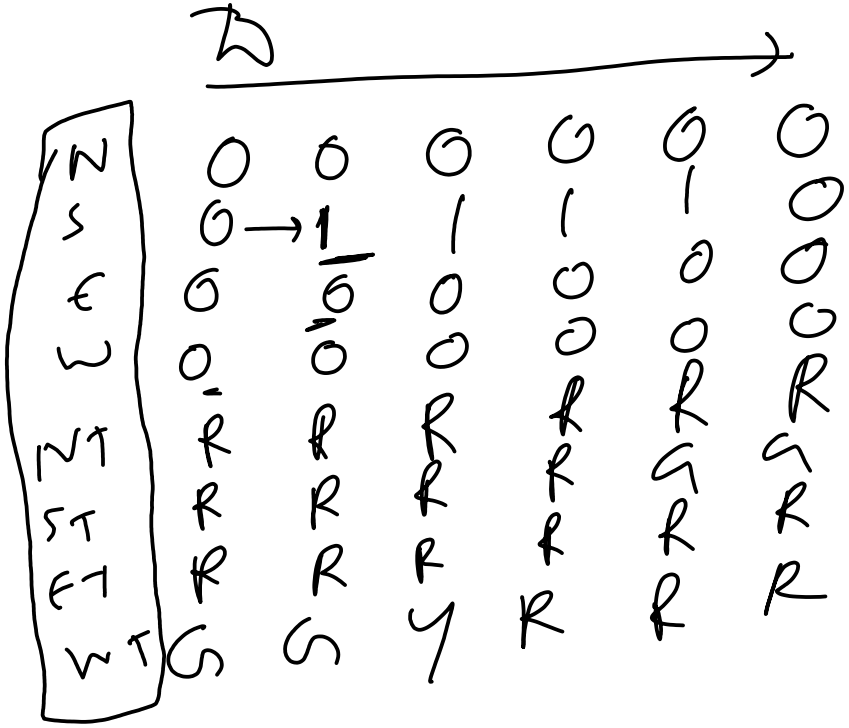
0 → I

Discrete time Reactive System.

"Properties" of behavior

behavior → Recording of the state "in" time.





$\mathcal{B} \models X[S=1]$

$X[W=G]$

$X[NT=G]$

$X[NT=R]$

$X[E=1]^X$

$X\,X[NT=G]$

$X\,X\,X[S=1]$

Linear Temporal Logic

→ Mathematical rep of behaviors in time.

→ Atomic Prop: $\mathbb{B}^n$ (n boolean vars)

→ Behavior in time
$\mathcal{D} : \mathbb{N} \to \mathbb{B}^n$ $\quad \mathcal{B}(1) = \text{state 1}$
$\mathcal{B}(2) = \text{state 2}$
$\vdots$

→ Boolean conn $\wedge, \neg, \vee$.

→ Temporal Const: X, G, F, U

What do X, G, F, U mean

$\mathcal{B} \models X(\phi)$ iff $\mathcal{B}[1] \models \phi$

$\mathcal{C}, T_{ex}$

$B_1 = a\ 0\ 0\ 0\ 0\ 0\ 0\ 0 \cdots$ $\quad B_1 \models X[a=0]$

$B_2 = 1\ 1\ 0\ 0\ 0 \cdots$ $\quad B_2 \models X[a<1]$

$B_3 = 0\ 1\ 0\ 0\ 0 \cdots$ $\quad B_3 \models X[a=1]$

$B_4 = 1\ 0\ 1\ 1\ 1 \cdots$ $\quad B_4 \models X[a>0]$