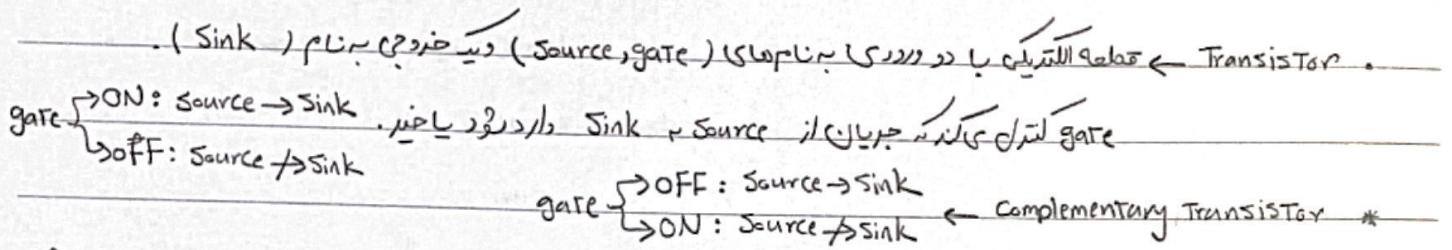


Transistors پ.ا.1



فلزاتی، سیلیکونی

راههای مختلفی برای پیاده‌سازی منطقی با اینها وجود دارد: ۱) استفاده از تراک و نترآک → Transistor موجود اداره: ۲) جریان الگولی

(2)

(3) لوله‌ای خلا (Vacuum tubes)

در لوله‌ای، الگولن عالی تواند از بیم (Source) به بیم صفحه (Sink) جریان پیدا کند. درین این دیگر grid به عنوان

grid مذکور شده تغییر داشته اند. می‌توان جریان الگولن عالی را مسدود کرد.

قانون مور (moore) در سال ۱۹۴۵، دوری بین مردم تقدیر ترانزیستورها در مدارهای مجتمع هر سال دو برابر خواهد شد.

Logical gates from transistors پ.ا.2

می‌توان تابع بولی میان NOT, OR, AND میان آنها را با استفاده از ترانزیستورها پیاده‌سازی کرد.

برای هرگز دو دروری میان $\{0,1\}^2 \rightarrow \{0,1\}^2$: G، این پیاده‌سازی شامل دیگر دو میان $\{0,1\}^m \rightarrow \{0,1\}^m$ خواهد بود که حاصل $= 2^m$ می‌شود اما دسته‌ای اعمال G را در این برابر ۱ نمود.

\Rightarrow اگر مداری مstellen از NOT, OR, AND وجود داشت باشد تابع $\{0,1\}^n \rightarrow \{0,1\}^n$ را محاسبه کنند می‌توان آن را در دنیا فیزیکی ساخت.

DNA مولکولی سی

Biological Computing پ.ا.3

محاسبه می‌تواند براساس سیم سیای biological یا شیمیایی باشد. \Rightarrow مساقه‌ی دن ترانزیستورها در تیجه لیستهای منطقی براساس

برای کوکردن عمليات قابل اجرا در سیم سیای بالاترایی \Rightarrow تبدیل مدارهای بولی به DNA sequences \Rightarrow Cello programming language.

cellular automata and the game of life پ.ا.4

چیز خالی نداشت

شبکه عصبی \rightarrow مدل ریاضی‌دانی برای اعماق است.

می‌توان شبکه عصبی را بعنوان مدار بولی‌ای در نظر نهاد و درسته باشد که با جایگزینی AND/OR/NOT آن را با استفاده از آستانه (threshold gates) می‌توان ایجاد کرد.

تعریف لاتک: به ازای هر بُردار (w_0, w_1, \dots, w_k) از اعداد صحیح و عدد صحیح t تابع متناهی $w_i x_i \geq t$ است که $x \in \{0, 1\}^k$ ، خروجی ای سر، اگر و تنها اگر $t \leq \sum_{i=0}^k w_i x_i$ برای درری

(همیلی خودمنزد)، زمانی که عنوان خروجی امیده که مجموع وزن داریت‌های درری از مقدار آستانه در نظر نهاده شود یا صادق شود

$$x = (1, 1, 0, 1, 0), w = (1, 1, 0, 0), t = 2$$

$$\sum_{i=0}^k x_i w_i = 1 + 1 + 0 + 0 + 0 = 2 \geq t \Rightarrow \text{خواهد بود}$$

$$x \in \{0, 1\}^5, w = (1, 1, 1, 1, 1), t = 3$$

در این مثال w تابع threshold function است و x با w ورودی است. (خروجی

هم میل مثال بالا بوده‌است).

شبکه عصبی مهندسی \rightarrow قرار است سیلوزی و تقلید کن و هر چند قراره یکدی معاملات را انجام بده برای همین

دارند و زمانی که این میل مثالها ازید آستانه مذکور عبوری کند، روند می‌شوند.

شبکه عصبی مهندسی \rightarrow قرار است سیلوزی و تقلید کن و هر چند قراره یکدی معاملات را انجام بده برای همین

حددد ب y یا $y = f(x)$ با x ای داشته باشد که f زیست‌پذیر باشد.

در این مثال y خروجی هر لایه با اعمال $\sum x_i w_i$ activation function می‌شود.

$f: R \rightarrow R$ (sigmoid, ReLU, hyperbolic tangent)

که می‌توان این تابع را با اعداد حقیقی کاری کن و آن را توانم اعداد حقیقی را به باینری تبدیل کنن که با این روشی بزرگی قابلیت پیاده‌سازی داشته باشد.

a computer made from marbles & pipes ۱.۲.۴

جیز فایلی نراثت و هنرین روزنامه دار از ایرانی کتاب نوایت می‌کند.

The NAND Function ۱.۴

$$\text{NAND}(a, b) = \begin{cases} 0 & a=b=1 \\ 1 & \text{otherwise} \end{cases}$$

میتوان کیت سای NOT, AND, OR, میتوان کیت سای

۱.۱۰ مینی

- * $\cdot \text{NOT}(a) = \text{NOT}(\text{AND}(a, a)) = \text{NAND}(a, a)$
- $\cdot \text{AND}(a, b) = \text{NOT}(\text{NOT}(\text{AND}(a, b))) = \text{NAND}(\text{NAND}(a, b), \text{NAND}(a, b))$
- $\cdot \text{OR}(a, b) = \text{NOT}(\text{AND}(\text{NOT}(a), \text{NOT}(b))) = \text{NAND}(\text{NAND}(a, a), \text{NAND}(b, b))$

$x = x_0, x_1, x_2$ مثال \leftarrow محاسبه با استفاده از NAND majority

$$\text{Maj} : \{0, 1\}^3 \rightarrow \{0, 1\}, \text{Maj}(x) = \begin{cases} 1 & x_0 + x_1 + x_2 \geq 2 \\ 0 & \text{otherwise} \end{cases}$$

می توانم تابع Maj را این‌طوری تعریف کنم: خروجی ایست که در دو جفت اول آن ارز وجود داشته باشند بعلاوه، ن

$$\text{Maj}(x_0, x_1, x_2) = \text{OR}(\text{AND}(x_0, x_1), \text{OR}(\text{AND}(x_0, x_2), \text{AND}(x_1, x_2))) = ((x_0 \wedge x_1) \vee ((x_1 \wedge x_2) \vee (x_0 \wedge x_2)))$$

حال از * استفاده می‌کنم و تیکت سای NAND, AND, OR, NOT خواهیم داشت:

$$\text{Maj}(x_0, x_1, x_2) = \text{NAND}(\text{NAND}[\text{NAND}(\text{NAND}(x_0, x_1), \text{NAND}(x_0, x_2)), \text{NAND}(\text{NAND}(x_0, x_1), \text{NAND}(x_1, x_2))], \text{NAND}(x_1, x_2))$$

از عربی این تابع می‌توان مثل موارد زیر را در نظر گرفت:

NAND Circuits ۱.۴.۱

XOR میتوان NAND را \leftarrow جایز

$$\text{XOR}(a, b) = \text{AND}(\text{NOT}(\text{AND}(a, b)), \text{OR}(a, b))$$

$$\text{NOT}, \text{OR}, \text{AND} \rightarrow \text{XOR}(a, b) = \text{NAND}[\text{NAND}(a, \text{NAND}(a, b)), \text{NAND}(b, \text{NAND}(a, b))]$$

NAND

a	b	$a \oplus b$	$\text{XOR}(a, b)$
0	0	0	0
1	0	1	1
0	1	1	1
1	1	0	0

برای

قفسیه ۳۱۲ NAND پر عملیات universal operation است. هر مدار بولی C با دلیت \neg مدار

C باشد اگر دلیت وجود دارد که میان تابعی که C محاسبه کند را محاسبه کند.

کلیت NOT را میتوان با دلیت NAND را بازی سایی AND OR را میتوان با دلیت NAND درست کرد. بنابراین حالت اول ۳ برابر تعداد دلیت های پر مدار بولی که \neg AND, OR, ساخته شده دلیت NAND نیاز است.

* اثبات قفسیه ۳۱۲ است تا در مدار بولی ساخته شده بازی سایی NOT, AND, OR, هر دو دلیت دلخواه باشند.

* دو مدل محاسباتی هم قدرست اگر مجدد مسابقه از توابع بتواتر محاسبه شوند.

مثال ۱ مدار NAND برای محاسبه تابع increment

$$INC_n: \{0,1\}^n \rightarrow \{0,1\}^{n+1} \Rightarrow INC_n(x) = y, \quad y = \sum_{i=0}^{n-1} y_i 2^i = \left(\sum_{i=0}^{n-1} x_i 2^i \right) + 1$$

مثال ۲

(حذلی خودرسانی) تابع INC پر مدار بولی تابعی از عددهای n رقمی (ربه عنوان وروری دریافت کنند و حاصل $x_0 + 1$ را ببرند) می توانیم تابع increment و لینجوری تغییر کنیم: "اگر بگذر ارزش ترین بیت افغان کن و Carry را انتقال بده" و برای این کار لافنه از کم ارزش ترین بیت نموده کنیم، همان‌طورهای قبیل کم تازه‌ترانی که بهترین هم‌زبریم، در این امر را با برابر انتقال می‌کنیم و مترقبه می‌رومیم. (بدای ساده‌تر از خاکیت LSB-first استاد می‌نموده این x_0, x_1, \dots)

Input: x_0, x_1, \dots, x_{n-1} representing the number $\sum_{i=0}^{n-1} x_i 2^i$

Output: $y \in \{0,1\}^{n+1}$ such that $\sum_{i=0}^{n-1} y_i 2^i = \sum_{i=0}^{n-1} x_i 2^i + 1$

Let $c_0 \leftarrow 1$

for $i = 0, \dots, n-1$ do

 Let $y_i \leftarrow XOR(x_i, c_i)$

 if $c_i = x_i = 1$ then

$c_{i+1} = 1$

 else

$c_{i+1} = 0$

 endif

end for

Let $y_n \leftarrow c_n$

NAND پر مدار

let $c_0 \leftarrow NAND(x_0, NAND(x_0, x_0))$

for $i = 0, \dots, n-1$ do

 let $y_i \leftarrow XOR_NAND(x_i, c_i)$ □

$c_{i+1} \leftarrow NAND(NAND(c_i, x_i), NAND(c_i, x_i))$

let $y_n \leftarrow NAND(NAND(c_n, c_n), NAND(c_n, c_n))$

XOR_NAND پر مدار تعریف می‌شود. □

مثال \leftarrow مدار NAND بایی جمع دو عدد.

• مدار تابعی با عیوب تدریجی این که و انجام بسیم یا هی توئین از grade-school استفاده نمی کند که درست بینهایت است.

$ADD_n : \{0,1\}^n \rightarrow \{0,1\}^{n+1}$, x_0, \dots, x_{n-1}
تابع اسلامی: $x \in \{0,1\}^n$ مدار ADD_n باشد.

البرهی برداشت.

($x = x_0, x_1, \dots, x_{n-1}$) استفاده LSB-first است.

Input: $u \in \{0,1\}^n$, $v \in \{0,1\}^n$

Output: $x+y$

Let $c_0 \leftarrow 0$

for $i=0, \dots, n-1$ do

Let $y_i \leftarrow u_i + v_i \bmod 2$
if $u_i + v_i + c_i > 2$ then
 $c_{i+1} \leftarrow 1$
else
 $c_{i+1} \leftarrow 0$

end if

end for

Let $y_n \leftarrow c_n$

NAND [NAND(x_0 , NAND(x_0, x_0)),

let $c_0 \leftarrow$ NAND(x_0 , NAND(x_0, x_0))]

for $i=0, \dots, n-1$ do

Let $y_i \leftarrow \text{XOR}_{\text{NAND}}(u_i, u_i)$
 $c_{i+1} \leftarrow \text{MAJ}_{\text{3NAND}}(u_i, v_i, c_i)$

Let $y_n \leftarrow c_n$

تابع \rightarrow تابع NAND بازیابی تواند در مثال سایر قبل تعریف $\text{MAJ}_{\text{3NAND}}, \text{XOR}_{\text{NAND}}$ *

شده اند جایزه ارسان شوند.

ساختار متفقین

$P = \text{NAND}(a, b)$ زبان برنامه نویسی ای است که مفهومی آن (بجز خطاها) مربوط به ورودی و خروجی (به کل) است.

The NAND-Circ programming language ۹.۴.۴

$U = \text{NAND}(A, B)$

$V = \text{NAND}(A, U)$

$W = \text{NAND}(B, U)$

$Y[0] = \text{NAND}(U, W)$

مثال \leftarrow برنامه ای که XOR را محاسبه می کند.

* است اگر از ورودی یک خروجی را بخواهیم $X[n]$ می تذکریم که در آن

X نیز متفق نورودی یا خروجی است.

. است

تعریف ۳.۱۴. Computing by a NAND-CIRC program

$f: \{0,1\}^n \rightarrow \{0,1\}^m$ باشد، دویم P محاسبه می‌کند f را این:

۱) P یک n -متغیره ووری و m -متغیره خروجی داشته باشد.

۲) برای هر درایه x در $\{0,1\}^n$ دمیم و خروجی y را دریافت می‌کنیم، آن وروری x ، آن P بدین خروجی y دریافت می‌کند.

equivalence

NAND circuits and straight-line programs are

قلم ۳.۱۷.

برای در تابع f ، s, m ، $f: \{0,1\}^n \rightarrow \{0,1\}^m$ باشد خود قبل محاسبه است NC program

آن دسته‌ای f با استفاده از مدار NAND قابل محاسبه است. (این مدل قلم ۳.۹)

این AON-program با ۵ خط قابل تبدیل به f باشد. کافی است تا NAND-Circ program

هر دسته از NOT, OR, AND، "این بوسیله" هارا با تعاریف متناهی‌ترین استفاده از NAND جاییز ارسانیم.

توابع نامتناهی را حساب کنیم. چون هر فرآیند متناهی را محاسبه می‌کند و من توانم "Turing Complete" c NC programming language

نمایم.

Equivalence between models of finite computation

قلم ۳.۱۹.

(or most 3s line) (with s-line)
NAND-circ program \leftrightarrow AON-program

NAND-circuit (or most 3s gates) \leftrightarrow AON-circuit (with s-gates)

هم مدل سایی کنیم این معرفی نهاده روش تنویر

برای مدل سایی کنیم این مدار خود را برای مدل سایی برنامه می‌کنیم

مدار کم رزیار می‌شود.

اگر مدار سایی بولی را بایسای عیناً $NAND \leq AON$ هم درست کنیم، باز هم در قدرت بامارسای امن‌کردن برابر خواهد بود

function \rightarrow specification of task, circuit / program \rightarrow implementation of task

Circuits with other gate sets ۳.۱۸

محاسبه می‌کنیم با $\{g_1, g_2, \dots, g_k\}$ از توابع $\{f_1, f_2, \dots, f_m\}$ صاریح تعریف نیم داش عنصر و به عنوان دسته استفاده

کنیم. همچنین می‌توان کد زبان برنامه‌نویی g تعریف کرد که هر خود ادن G شامل معادله‌ی g باشد

مثل $f_{i,j}$ با اعمال کردن تابع g_i ($g_i \in g$) روی متغیر سایی i از قاعده تعریف شده اند یا وردی از جاییست
 $f_{i,j} = g_i(\dots)$

General Straight-line programs $\leftarrow ۳.۲۰$ ترتیب.

تعريف داشته باشند.

IF, ZERO, ONE circuits $\leftarrow \text{JL3}$

$$\text{ONE} : \{0,1\} \rightarrow \{1\}, \text{ZERO} : \{0,1\} \rightarrow \{0\}, \text{IF} : \{0,1\}^3 \rightarrow \{0,1\}$$

↓

ا) عنوان دریافت، a مقدار است

ب) خروجی درست برقرار بودن را رد

c) خروجی درست عدم برقراری را رد

ابنی \hookrightarrow Universal circuit $\{ \text{IF, ZERO, ONE} \}^*$

$$\text{NAND}(a,b) = \text{IF}(a, \text{IF}(b, \text{ZERO}, \text{ONE}), \text{ONE})$$

Specification Vs Implementation

۳.۱.۱

Computation: input \rightarrow [] \rightarrow output (process that maps an input to output)

function \rightarrow specification of a computational task. (What output should be produced for every particular input)

Program (or circuit or any other way to specify algorithms) \rightarrow implementation
of How to compute the desired output from the input.



مجموعه F را به عنوان مجموعه ای از تابع بولین به صورت $F = \{f_0, \dots, f_{n-1}\}$ تعریف می کنیم به صورت

$$\cdot k_i \in N, f_i : \{0, 1\}^{k_i} \rightarrow \{0, 1\}^m$$

خروجی

F program، دنباله ای از خطهای است به درجه ای از آن ما مقادیر از اعمال یکی از تابع بولین در مجموعه F است از متغیرهای $x[0], x[1], \dots, x[m-1]$ در دسترس است. \leftarrow

$$U_p = f_m(x[0], x[1], v_1)$$

و در

$$(Y[0]) = f_p(v_1, U_p)$$

بیت در خروجی

(۱) F program است، اگر برای برنامه ای با F نوشت (universal set of operations)، F مجموعه ای از تابع $NAND$ را می توان محاسبه کند.

حالات فصل

الدریم دستورالعمل برای انجام یک محاسبه به عنوان دنباله عملیات سای "ساده" یا "ابتداي" است.

یک مجموعه کاربردی برای عملیات "ابتداي" مجموعه $\{AND, OR, NOT\}$ است.

یک مجموعه کاربردی برای عملیات "ابتداي" مجموعه $\{NAND, ZERO, ONE\}$ است. \leftarrow

از $NAND$ میتوان برای پیاده سازی خیلی از تابع دیگر استفاده کرد.

میتوان به صورتی، مفهومی تابع $\{0, 1\}^m \rightarrow \{0, 1\}^m$ را به عنوان یک تابع محاسبه پذیر با استفاده از زبان برنامه نویسی $NAND-Circ$ تعریف کنیم.

برای یک مجموعه از عملیات های ساده، مفهوم محاسبه پذیر بودن بوسیله یک مدار و مفهوم محاسبه پذیر بودن بوسیله یک straight-line program برابرند.