Subject: ..... Year: ..... Month: ..... Date: .....

(از سمت راست خوانده شود)

بسم الله

جبر بول:

حیدبول چیست؟ هر گاه مجموعه‌ای با شد که از عمل $(O, \square)$ (۲ عمل) $(A, \square, O)$ تشکیل شده باشد و دارای ۱ و ۰ نیز باشد.

۱- بسته بودن: $\forall a,b \in A: \; a \square b \in A \;,\; a \circ b \in A$

۲- جابجایی: $\forall a,b \in A: \; a \circ b = b \circ a,\; a \square b = b \square a$

۳- شرکت پذیری: $\forall a,b,c \in A: \; a \circ (b \circ c) = (a \circ b) \circ c$

$a \square (b \square c) = (a \square b) \square c$

۴- توزیع پذیری: $\forall a,b,c \in A:$

$a \circ (b \square c) = (a \circ b) \square (a \circ c)$

$a \square (b \circ c) = (a \square b) \circ (a \square c)$

۵- عضو خنثی: شرط عضو خنثی برای عمل $\square$: $\exists e_{\square} \in A \;:\; a \square e_{\square} = e_{\square} \square a = a$

$\exists e_{\circ} \in A \;:\; a \circ e_{\circ} = e_{\circ} \circ a = a$

۶- عضو معکوس: هر عضو در A وجود دارد عضو معکوسی در A بطوری که اگر بین این دو عضو عمل $\square$ یا $\circ$ باشد عضو خنثی آن مجموعه (عمل شود).

Subject: ..... Year: ..... Month: ..... Date: .....

جبر بول:

$(R, +, \times)$ یک جبر بول است.

$(Q, \wedge, \vee)$ جبر بول است.

PM چیست؟

۱ متغیر چند تابعی شدنی است.

۱ متغیر تعداد Max و Min.

۱ متغیر هیچ DPT ی شدنی است.

Subject: Year: Month: Date:

آیا $(R, +, \times)$ یک میدان است؟

میدان زیرا خاصیت توزیع پذیری را ندارد.

$3\times(2+5) = (3\times2)+(3\times5)$

$2+(3\times4) = (2+3)\times(2+4)$ ع

P_M مجموعه همه مجموعه های دنیا است.

همه مجموعه های دنیا زیرمجموعه M هستند.

همه مجموعه های دنیا عضو P_M هستند.

آیا $(P_M, \cup, \cap)$ یک میدان است؟

همه خاصیت را دارد. (عضو خنثی $\cup$ ، $\emptyset$ و عضو خنثی $\cap$ ، M) است.

خاصیت عضو مکمل هم دارد.

$$\begin{cases} A \cap \bar{A} = \emptyset \\ A \cup \bar{A} = M \end{cases}$$

آیا $(Q, \vee, \wedge)$ یک میدان است؟

نه (عضو خنثی F، T است)

خاصیت عضو مکمل

$Q \wedge \bar{Q} = F = e_1$

$Q \vee \bar{Q} = T = e_2$

Subject: Year: Month: Date:

$\{\{1, 0\}, +, \cdot\}$ چرا میدان است، زیرا

$a \cdot b \Rightarrow$	a	b	a.b
	0	0	0
	0	1	0
	1	0	0
	1	1	1

a+b	a+b
00	0
01	1
10	1
11	1

زیرا خاصیتهای زیر را دارد.

۱- بسته بودن: همیشه جواب صفر و یک است.

۲- جابجایی: دارد $a+b=b+a$ / $a \cdot b = b \cdot a$

۳- شرکت پذیری:

میدان: نامحدود، میدانی که اعضای آن روی مجموعه نامحدود باشد.

محدود، میدانی که اعضای آن روی مجموعه محدود باشد.

برای اثبات حالتهای میدانی روی میدانهای محدود به کمک جدول حقیقت صورت میگیرد چون میدان محدود است برای همه حالتها را می توان در این جدول چک نمود.

Subject: ..... Year: ..... Month: ..... Date: .....

قضایای جبر بول،

۱) idempotency (خودتوانی)

$a = a \circ a \circ a \dots a$

$a = a \square a \square a \dots a$

۲) involution

$\overline{\overline{a}} = a$

۳) جذب

$a \circ (a \square b) = a$

$a \square (a \circ b) = a$

۴) جذب

$a \circ (\bar{a} \square b) = a \circ b$

$a \square (\bar{a} \circ b) = a \square b$

۵) دمورگان

$\overline{a \square b} = \bar{a} \circ \bar{b}$

$\overline{a \circ b} = \bar{a} \square \bar{b}$

۶) عضو خنثی (عضو همانی) Sink (null)

$a \square$

amitisweb.com

Subject: ..... Year: ..... Month: ..... Date: .....

اصول موضوعه چیست؟
قضایای جبر بول کدام هستند؟

شرکت پذیری:

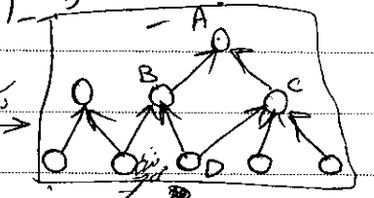
a	b	c	b+c	a+b	a+(b+c)	(a+b)+c
0	0	0	0	0	0	0
0	0	1	1	0	1	1
0	1	0	1	1	1	1
0	1	1	1	1	1	1
1	0	0	0	0	0	0
1	0	1	1	0	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

وقتی چک کنید می بینید که دو ستون آخر برابرند ⟸ شرکت پذیری داریم +

amitisweb.com

ادامه...

اصول موضوعه، گزاره هایی هستند که درستی آنها را بدون دلیل می پذیریم.



اصول موضوعه ⟸

(A چرا درست بدون دلیل درست بودن B، C حال چرا D درست زیرا D یک موضوع بدیهی است و بی دلیل پذیرفته ایم که به این اصول موضوعه گوییم.)

(اصول دیگری که بتوان این نسبت را به این معنی نسبت کرد این جبرها اشتباه است.)

Subject: Year: Month: Date:

* $a \circ e_\square = e_\square \circ a = e_\square$
$a \square e_\circ = e_\circ \square a = e_\circ$ (عضو خنثی)

اثبات: شروع از اصول و روابط و با درستی قضیه را اثبات گوییم.

$$a \circ (\bar{a} \square b) = (a \circ \bar{a}) \square (a \circ b)$$
$$= e_\square \square (a \circ b) \quad \text{اصل متمم}$$
$$= a \circ b \quad \text{اصل عضو خنثی}$$

دوگان Duality:
اگر دقت کنید تقریباً بین + و × رابطه‌ای وجود دارد.

دوگان ↓
$\square \to \circ$
$\circ \to \square$
$e_\circ \to e_\square$
$e_\square \to e_\circ$

$\bar{e}_\circ = e_\square$
$\bar{e}_\square = e_\circ$

اتحاد: تساوی‌ای که به ازای تمام مقادیر مجهول برقرار باشد را اتحاد گوییم.

نکته: دوگان فقط روی اتحادها انجام می‌شود (هنگام اثبات).

مثال: به روش دوگان اثبات کنید.

روی یک سوی کمان توان دوگان زد. فقط روی اتحادهای تک‌متغیره دوگان زد.

رابطه: $f(x,y,z) = \underbrace{xy^2 + z}_{\text{عبارت}}$ ← تابع سوئیچینگ

تعریف حیری یک تابع: با عبارت از این هم تابع بدست آید.

* به تابع‌ها در حیر سوئیچینگ = تابع سوئیچینگ Switching function گوییم.

* توابع را می‌توان به صورتی تعریف کرد که همهٔ حالتهای تابع را برای x,y,z دارد که به این تعریف با جدول حقیقت گوییم.

تعریف تابع: دادن رابطه / دادن کل مقادیر تابع به ازای همهٔ حالتهای x,y,z ← جدول حقیقت

$$f(x,y,z) = \begin{cases} x + \bar{y}z \\ \text{رسم جدول} \end{cases}$$

x y z	f(x,y,z)
...	...
⋮	⋮

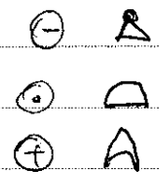
Subject: Year: Month: Date:

تعداد توابع با ورودی n متغیر برابر است با 2^{2^n} (در صورتی که توابع را متفاوت در نظر بگیریم)

دو بیت سوئیچ تعداد توابع با ورودی ۳ متغیر $\Leftarrow 2^{2^3} = 2^8 = 256$

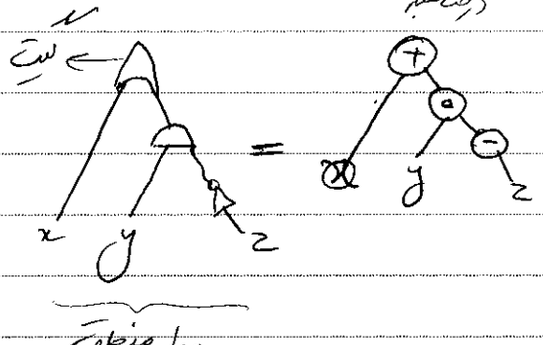
اما تعداد توابع دو بیت سوئیچ برای ۱ متغیر ∞ است.

$f(x,y,z) = x^2 + y$ ← این یک تابع ۳ متغیره است
(با اینکه فقط x و y دارد اما حداقل از ۳ متغیر است)



$f(x,y,z) = x + \bar{y}z$

gate ← سلف



مدار منطقی

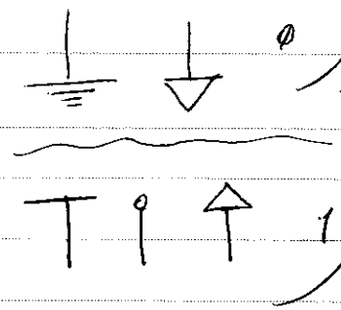
a b	f_0	f_1	f_2 - - - -	f_{14}	f_{15}
0 0	0	0	0	1	1
0 1	0	0	1	1	1

Subject: Year: Month: Date:

توابع ثابت

$f_0(a,b) = 0$

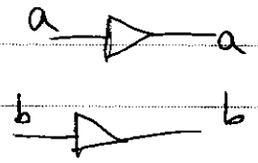
$f_{15}(a,b) = 1$



توابع همانی

$f_{12}(a,b) = a$

$f_{10}(a,b) = b$

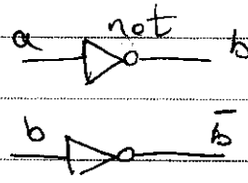


بافر

توابع نقیض

$f_3(a,b) = \bar{a}$

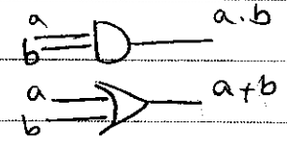
$f_5(a,b) = \bar{b}$



and/or

$f_8(a,b) = ab$

$f_{14}(a,b) = a + b$



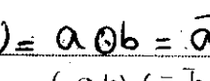
xor_xnor

a b	a⊙b
0 0	1
0 1	0
1 0	0
1 1	1

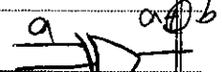
← xnor

xnor $f_9(a,b) = a \odot b = \overline{a \oplus b}$

$(ab) + (\bar{a}\bar{b})$



xor $f_6(a,b) = a \oplus b = \overline{a \odot b}$



Subject: Year: Month: Date:

$f(a,b) = \overline{ab} = \bar{a} + \bar{b}$ nand_nor

$f_1^7(a,b) = \overline{a+b} = \bar{a} \cdot \bar{b}$

(۳) nand:

(۴) nor:

توابع implication

$p \to q \equiv \sim p \vee q$ در نرم‌افزار

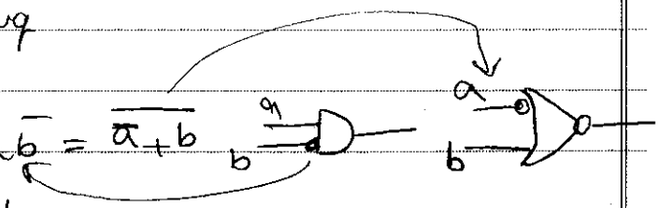
$f_{11}(a,b) = \bar{a} + b$

$f_{13}(a,b) = a + \bar{b}$

توابع inhibition

$p \to q \equiv \sim p \vee q$

$\sim(p \to q) = p \wedge \sim q$

$f_4(a,b) = a\bar{b} = \overline{\bar{a} + b}$ 

$f_2(a,b) = \bar{a}b$

چه مداری ساده‌تر است؟

Subject: Year: Month: Date:

کیفیت معماری: مداری که طراحی می‌کنید باید این ویژگی‌ها را داشته باشد.

۱- سرعت ↑

۲- توان مصرفی ↓

۳- سطح اشغال شده ↓

۴- قیمت ↓

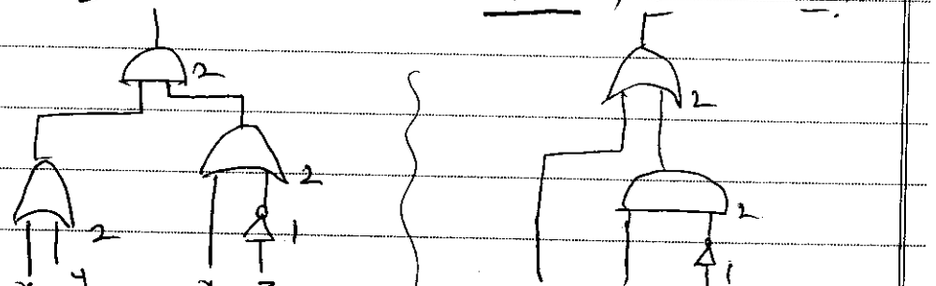
۵- قابلیت پذیرش ↓

مهم‌ترین این عوامل: حجم ↓ ، سرعت ↑ (تأخیر ↓)

تعداد ترانزیستورهای لازم بیشتر از یک است یعنی ترانزیستور نداریم که یک ترانزیستور داشته باشد.

تعداد ترانزیستورهای لازم برابر ۲ است (Gate inputs GI) است

(سادگی) بین دو مدار زیر کدام ساده‌تر است؟ (مداری که GI کمتر است)

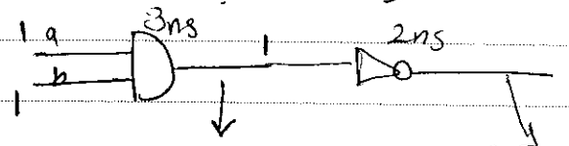


تأخیر در مدار سری / تأخیر در مدار موازی / مسیر بحرانی

Subject: Year: Month: Date:

سرعت و تأخیر

قبل از اینکه تأخیر گیت ها را حساب کنیم حالت خروجی گیت حالت خروجی مشخص می باشد.



اگر در لحظه t_0 $b=0$ آنگاه

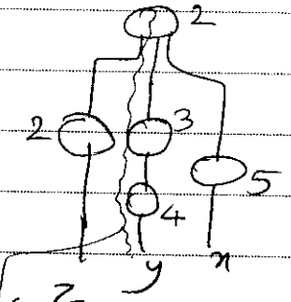
خروجی در زمان t_0+3_{ns} خروجی عوض می شود.

بعد از t_0+3+2_{ns} عوض می شود.

تأخیر

همیشه در مدار سری تأخیر جمع می شود جمع تأخیرها.

همیشه در مدار موازی تأخیر خروجی می شود Max تأخیرها.



تأخیر مدار $z=4$

$y=9$

$x=7$

Delay: $\Rightarrow$ Max = 9ns

مسیر بحرانی: مسیری که بیشترین تأخیر را دارد Critical Path

طول مسیر بحرانی / شاخص ها

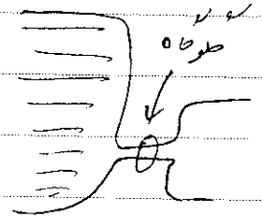
سرعت

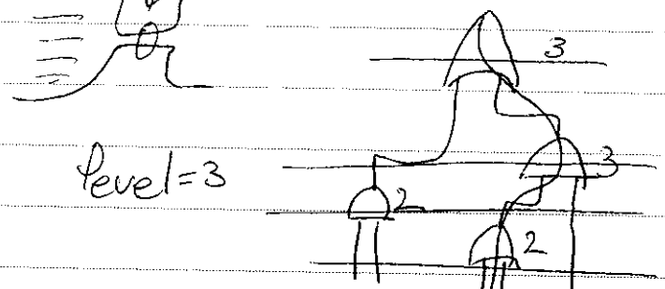
Subject: Year: Month: Date:

جلسه دوم مدار منطقی

مسیر بحرانی حالت خروجی گیت ها در مدار مهم

یکی از سیستم هایی که آنرا بهبود ندهیم هرجای دیگر را اگر بهبود دهیم هیچ فایده ای ندارد.





تأخیر این مدار

برای بهبود مدار باید مسیر بحرانی را بهبود دهیم.

برای شاخص بهبودی تعداد دروازه ها در نظر می گیریم.

شاخص سرعت را عوض می کنیم برای آنکه دقیقتر بتوانیم آنرا محاسبه کنیم

تعداد طبقات

می آییم level (طبقات) مدار را محاسبه کرده و برای بهبود می آییم

تعداد level ها یش را پایین کنیم. در مدار فوق level=3

طبقات را برای سرعت

/POS / SOP / PT / ST

Subject: Year: Month: Date:

(رمنطقی به این ساختارها DAG گفته نه درخت اما در رسمهای بعدی (نیز براحتی درخت ها افتاده البته در رسمهای بعدی نیز هم درخت نیستند حتماً DAG هستند)

هدف ما ساده‌سازی و افزایش سرعت.

ما در مثال ساده‌ترین مدار نیستیم، ما در مثال ساده‌ترین عبارت هم نیستیم بلکه هدف پیاده‌کردن ساده‌ترین مدار درین پیرسرعتها است. و برای رسیدن به این هدف روی عبارت کار می‌کنیم.

روش Quine.

روش جهت یافتن ساده‌ترین مدار درین پیرسرعتها.

(حرف، ST، PT، SOP، POS) ← انواع عبارتهایی که در این روش به کارش سروکار داریم.

حرف: عبارتی که فقط از یک متغیر تشکیل شده است مثل a، b، c

که می‌تواند با خودش یا با نقیضش (ø=0 یا 1) باشد.

حرف است ← x اما حرف نیست ← x+ø

ST: جمع چندین حرف با یکدیگر، البته یک حرف تنها هم می‌تواند ST باشد. (Sum Term)

a+b+c یا a یا b+ø

ST ST ST

چون ø حرف ← ST است

PT: ضرب چندین حرف با یک حرف تنها

Subject: Year: Month: Date:

PT چیست؟ / ST چیست؟ / DAG چیست؟

تعداد لیت input ↓ ⇐ ساده‌تر

تعداد طبقات ↓ ⇐ سرعت ↑ (منطقی اولویت سرعت است.)

سرعت تا آنجایی که می‌شه خوب شه پس سراغ پیچیدگی می‌رویم تا ساده‌ترش کنیم.

فرض کنید تابعی به صورت روبرو داریم

G(a, b, c, d, e, f) = abc + ade + cdf

(برای یک تابع مدارهای مختلفی می‌توانیم داشته باشیم)

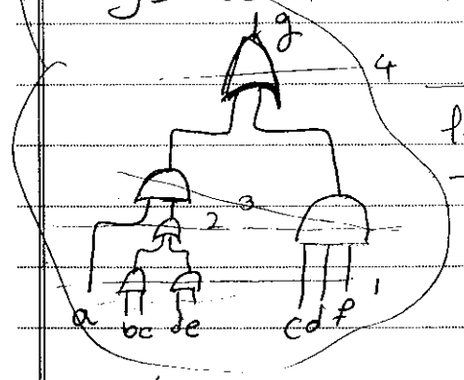
(برای ورودها و عبارتها تنها ø یک یا یک است)

⇐ برای ساده‌سازی مدار می‌رویم تابع را ساده می‌کنیم.

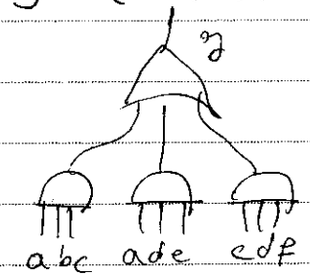
ساده‌سازی عبارت نتیجه ساده‌سازی مدار را می‌دهد زیرا.

DAG

g = abc + ade + cdf



g = a(bc+de)+cdf



level=2

تعداد لیت = 12

تعداد لیت = 7

این پنج تا برای عبارت هستند Max، Min برای تابع هستند

Subject: Year: Month: Date:

جمع چندین PT با یک PT تهی. SOP = Sum of

$\underbrace{a+b+c}_{SOP}$ ، $\underbrace{\bar{a}\bar{b}c + ac\bar{d} + b\bar{c}}_{SOP}$ ↓ $\underbrace{a}_{SOP}$ POS =

ضرب چندین ST با یک ST تهی.

$(\overbrace{a+b}^{ST})(\overbrace{\bar{c}+d}^{ST})(\bar{a}+b+\bar{c})$ با abc

حرفی یک PT است ← (abc) ، (xy)

ST است و POS است.

چون مرتب است. ← چون Sum است

ه $a.(b*c+de)+cdf$ → POS نه، SOP

نه ST، PT

چون باید فقط ۰ باشد ← چون باید فقط ۱ باشد.

⟸ پس همه عبارتی داشته باشیم که خروجی ۰ باشد، همه عبارتی هم داشته باشیم که جز هیچ یک از این پنج تا نباشد.

تابع ماکسترم ← Maxterm

تابع مینترم

Minterm تابعی که به ازای یک ترکیب خاص از ورودی ۱ است.

تعداد توابع min و max با n ورودی؟

Subject: Year: Month: Date:

مثال تابع دو ورودی

a	b	c	f(a,b,c)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

→ minterm است

چون فقط یک جا ۱ است.

تابع ماکسترم Maxterm

فقط به ازای یک ترکیب خاص از ورودی ۰ است و به ازای بقیه ۱ است.

با n ورودی به تعداد 2^{2^n} تابع داریم.

با n ورودی به تعداد 2^n تابع ماکسترم داریم.

با n ورودی به تعداد 2^n تابع minterm داریم.

$2^n - 2^{n+1}$

بقیه توابع نه Max و نه min.

Subject: Year: Month: Date:

(دو قضیه مهم)

قضیه اساسی اول صدر سوئیچها:

هر تابع قابل نمایش به صورت جمع مینترم‌ها است.

هر تابع قابل نمایش به صورت ضرب ماکسترم‌ها است.

فرض کنید تابعی به صورت روبرو داریم.

a b c	f(a,b,c)					
0 0 0	0	0	0	0	0	0
0 0 1	1	1	0	0	0	0
0 1 0	1	0	1	0	0	0
0 1 1	0	0	0	0	0	0
1 0 0	1	0	0	1	0	0
1 0 1	1	0	0	0	1	0
1 1 0	1	0	0	0	0	1
1 1 1	0	0	0	0	0	0

به تعداد ۱ها ستون

هر ستون یک عدد یک دارد

این هر یک minterm

به تعداد صفرها ستون

هر ستون فقط یک صفر دارد

این هر یک Maxterm

Subject: Year: Month: Date:

مثال قبل مینترم‌ها برابر است با:

$f(a,b,c) = m_1(a,bc) \dots$

فرم استاندارد یا نرمال (Canonical)

$\sum m(1,2,4,5,6) \equiv \prod M(0,3,7)$ $m \subset M$ تابع به صورت نمایش

است.

قضیه اساسی دوم:

هر مینترم قابل نمایش به شکل PT است که همه متغیرها در آن

است.

هر ماکسترم قابل نمایش به شکل ST است که همه متغیرها در آن

است.

$\rightarrow m_1(a,bc) = \bar{a}\bar{b}c$
a b c
0 0 1

واسطه ۱ / not صفر / min

$\rightarrow M_3(a,b,c) = a+\bar{b}+\bar{c}$
0 1 1

واسطه ۰ / not ۱ / Max

$\rightarrow M_7(a,b,c) = \bar{a}+\bar{b}+\bar{c}$
1 1 1

Subject: Year: Month: Date:

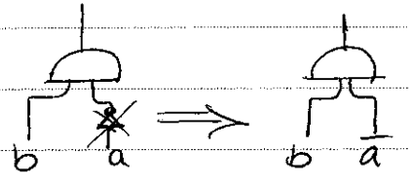
* ترکیب عبارت اول و دوم را نیز هم بنویسید.

هر تابعی قابل بیان به صورت SOP است.

هر تابعی قابل بیان به صورت POS است.

* مدارات منطقی را که گیت not در ورودی دارند می‌توانیم آن را حذف کنیم.

اما ورودی‌ها باید p باشد.



* خروجی مدار منطقی‌شان صفر طبقه است.

* PT و ST ، یک طبقه است.

* SOP, POS ، دو طبقه است.

در شبکه هر تابع حداکثر ۲ طبقه دارد.

ورودی‌ها مدارهای هستند که p طبقه باشند.

Canonical → SOP ⟹ ساده‌ترین SOP

→ POS ⟹ ساده‌ترین POS

$f(a,b,c) = ab\bar{c} + \bar{a}b\bar{c} + a\bar{b}\bar{c}$

$f(a,b,c) = (a+\bar{a})b\bar{c} + a\bar{b}\bar{c}$

$= b\bar{c} + a\bar{b}\bar{c}$

$= \bar{c}(b + a\bar{b})$

$= (b+a)\bar{c}$

$f(a,b,c) = b\bar{c} + a\bar{c}$ ← SOP

ولی در مدارات نمی‌توانیم از آی‌دی‌ها و اصول استفاده کنیم، برای همین Quine

استفاده کنیم.

قوانین مورد نیاز در این روش.

۱- تکرار $a = a + \cdots + a$

$a = a \cdot \cdots \cdot a$

۲- همجواری منطقی ۱- متغیرها یکی باشند.

۲- در not ها یک تفاوت باشد. از یکی خودش را نفی

همجوار $\bar{a}b\bar{c}d$ و $\bar{a}bcd$

غیرهمجوار چون متغیرهایش ... xyz و wyz

Subject: Year: Month: Date:

$x\bar{y}\bar{z} + x\bar{y}z \Rightarrow \boxed{x\bar{y}}$ → ساده شده

$ab\bar{c} + \bar{a}b\bar{c} + a\bar{b}\bar{c}$

روش اول → $\underline{ab\bar{c}} + \underline{\bar{a}b\bar{c}} + \underline{a\bar{b}\bar{c}}$

$\{a\bar{c} + ab\bar{c}\}$

روش دوم → $\underline{ab\bar{c}} + \underline{\bar{a}b\bar{c}} + \underline{ab\bar{c}} + \underline{a\bar{b}\bar{c}}$

تکرار — همجوار — همجوار

$= b\bar{c} + a\bar{c}$ ✓

روش Quine : با استفاده از قوانین تکرار و همجواری می‌توان به تابع ساده رسید

Subject: Year: Month: Date:

کارنوف Karnaugh-Map

روش ساده سازی کارنوف

روش‌های ساده‌سازی چیست؟

قوانین در روش Quien چیست؟

× در ساده سازی کارنوف هر متغیر مثبت دارای ـــــ همسایه مجاور است

مهم‌ترین هدف برای ساده‌سازی چیست؟

Cube ← ۲ درجه‌ای

۳ نکته ساده‌سازی

2^n خانه‌ای ـــــ Cube ـــــ متغیر حذف شده

prime Emplicant چیست؟

prime Emplicant ۲ درجه چیست؟

Subject: Year: Month: Date:

Karnaugh Map

روش ساده سازی کارنو

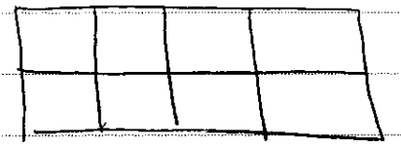
هر مینترم ۳ متغیره دارد ۳ همسایه همجوار است.

$a\bar{b}\bar{c}$ ↙ ↓ ↘

$\bar{a}\bar{b}c$ abc $a\bar{b}\bar{c}$

هر مینترم n متغیره دارای n همسایه همجوار است.

جدول کارنو جدولی است با ۸ خانه که هر خانه هم ۳ تا همجوار (همسایه) دارد



تعداد خانه ها ۸، تعداد P ها

که PT ، که ST

که PT ، به هر PT یک که نسبت می دهیم که از ۳ سلول ۰، ۱، - تشکیل شده است.

که اگر تابع آمده ۱ اگر نیامده ⊘ و اگر لقیشش بود ∅.

مثال: خروجی ۱

f(a,b,c) نتیجه ∅

a bc

$\bar{c}b$ ⟸ ۱∅ که

که ST ، به هر ST یک که نسبت می دهیم که از ۳ سلول ۰، ۱، - تشکیل شده است.

Subject: Year: Month: Date:

وزن همسایه هست ۵

Subject: Year: Month: Date:

مثال: همواری از روی کدها راحت محاسبه می‌شود.

$g(a,b,c,d,e,f)$

$ab\bar{c}d\bar{e}f$ ، $\bar{d}ca\bar{e}\bar{b}$

1 0 1 0 1 -
1 0 1 0 0 -

ساده شده ⟹ 1 0 1 0 - -

همواری چون در خط — حتماً باید یکی باشد و در صفر و یکی هم فقط یک اختلاف داشته باشند.

تعریف دیگر یک PT:

یک PT مقادیری از ورودی‌ها که باید به ورودی‌های آن PT را یک بکند.

$b\bar{c}$ $\xrightarrow{f}$ (abc) ⟹ -10

برای یک ST که not دارد صفر می‌دهیم چه مقادیری بدهیم تا تابع 0 شود.

$f(a,b,c,d,e)$

$(abcde)$ خروجی ← 0

$\bar{b}+a+\bar{e}$ ⟹ $0 1 - - 1$

مثال:

فرض کنید 3 متغیر داریم. چند حالت PT می‌توانیم ایجاد کنیم.

$\bar{3}^n$

3^n ⟸ کل حالت‌های PT

① a
② $\bar{a}$
③ -

اما اگر هم "—" باشند که صحیح نیست

کل P.T ها برای n متغیر ≡ $3^n - 1$ عدد

حل کارنو

$f(a,b,c)=ab\bar{c}+\bar{a}b\bar{c}+a\bar{b}\bar{c}$ ⟸

c \ ab	$\bar{a}\bar{b}$	$\bar{a}b$	ab	$a\bar{b}$
$\bar{c}$	0	1	1	1
c	0	0	0	0

$\bar{b}$ b b

وقتی SOP می‌خواهیم بزنیم خانه‌های 0 را یک می‌کنیم.

PT ⟹ -10 ، $1-0$

SOP جمله ⟹ $b\bar{c}$ ، $a\bar{c}$ ⟹ $b\bar{c}+a\bar{c}$

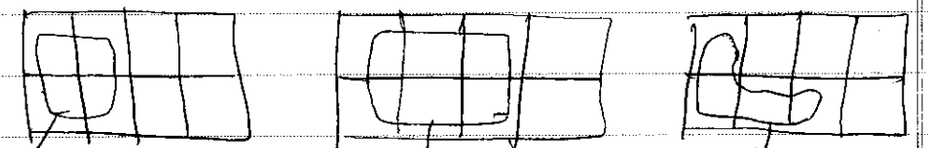
c \ ab	$\bar{a}\bar{b}$	$\bar{a}b$	ab	$a\bar{b}$
0	1	1		
1	1	1		

$0-0$ ⟸ $0-0$

$0-1$ ⟸ $0-1$

Subject: Year: Month: Date:

Cube ها:
۱- اولاً 2^n خانه باشند.
۲- شکل آن‌ها مستطیل یا مربع باشد.
↳ روی خانه‌هایی که می‌توانیم با هم بگیریم.



Cube ۴ تایی / Cube ۲ تایی / Cube هشتی

نکات ساده‌سازی:
۱- Cube ها تا حد امکان بزرگ باشند.
۲- Cube ها تا حد امکان تعدادشان کم باشد.
۳- خانه‌ای که حاوی 1 است حتماً (یک Cube ها همه آن‌ها را) پوشش بدهند.

یک جدول برای ۳ متغیر ۸ مینترم داریم و هر متغیر هم ۳ همسایه دارد.

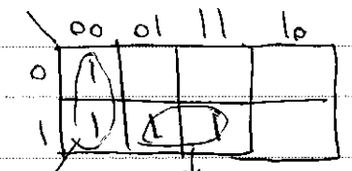
Cube ۲ خانه‌ای یک متغیرش حذف شده است.

Cube ۴ خانه‌ای ۲ متغیرش حذف شده است.

Cube 2^n تایی n متغیرش حذف شده است.

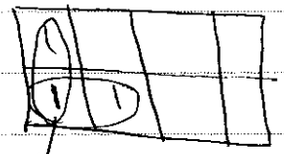
جلسه ششم

$f(a,b,c) = \sum m(0,1,3,7)$ این تابع را ساده کنید.



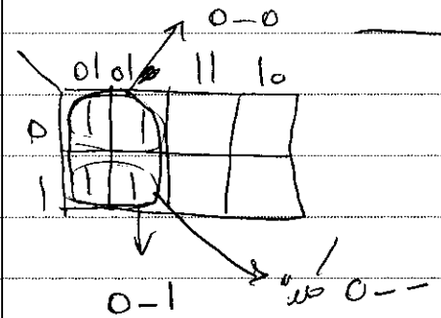
00- -11 $\Rightarrow \bar{a}\bar{b} + bc$

$\bar{a}\bar{b}$ bc



در این حالت Cube ۲ تایی داریم که همسایه ندارد پس می‌ماند.

در این حالت نشدنی است.



Subject: ..... Year: ..... Month: ..... Date: .....

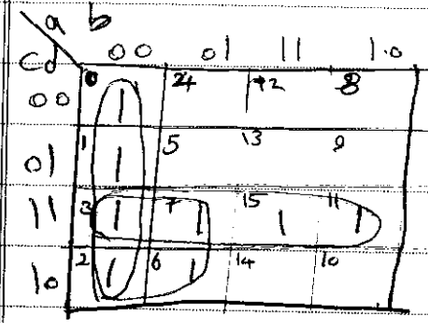
Prime Implicant

Cube ی که داخل cube های بزرگتر از خودش جا نشود. (یعنی نتوان آن را دیگر ساده شود)

یعنی Cube هایی که در ساده سازی می گیریم همان Prime Implicant ها هستند.

جدول ۴ متغیره

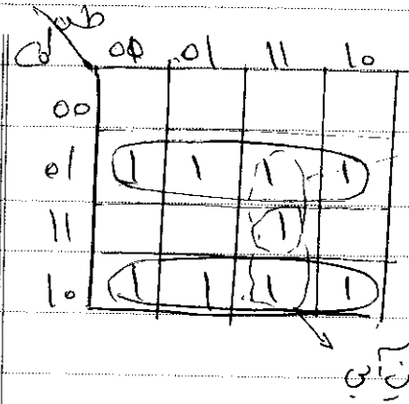
جدول ۴ متغیره دارای ۱۶ خانه است و هر خانه دارای ۴ همسایه است. راه هر مینترم هم ۴ همسایه دارد.



$$f(a,b,c,d) = \sum m(0,1,2,3,6,7,11,15)$$

$$= \bar{a}\bar{b} + \bar{a}c + cd$$

Subject: ..... Year: ..... Month: ..... Date: .....



در ساده سازی جواب مینیمم به فرم زیر نیست.

$$f(a,b,c,d) = \bar{c}d + c\bar{d} + abc$$

$$f(a,b,c,d) = \bar{c}d + c\bar{d} + abd$$

۴ دسته Prime Implicant داریم:

۱- PI های ضروری: که همشان EPI هم هستند.

۲- PI های غیرضروری: که همشان N.EPI هم هستند.

PI هایی هستند که هر چی ساده کنیم جدول را، در جدول دیده می شود.

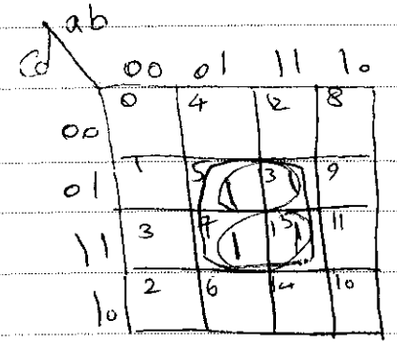
مثل مثال بالا که PI های ضروری $\bar{c}d + c\bar{d}$

PI های غیرضروری $\begin{matrix} abc \\ abd \end{matrix}$

PI های غیرضروری که PI هستند که در هیچ جواب ساده سازی ای ممکن نیست.

Subject: Year: Month: Date:

این جدول را ساده کنید.



$f(a,b,c,d) = \sum m(5,7,13,15)$

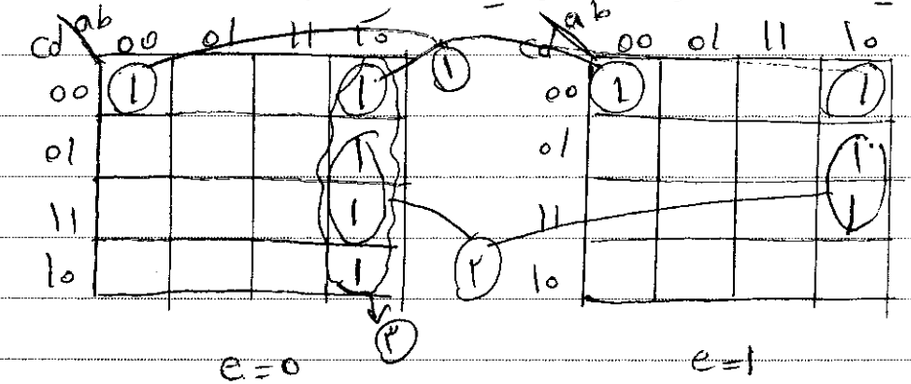
$f(a,b,c,d) = \bar{a}b\bar{c}d + \bar{a}bcd + ab\bar{c}d + abcd$

$= bd$ ✓

$f(a,b,c,d) = b\bar{c}d + bcd$ ← این هم ساده است اما ساده‌ترین نیست.

جدول کارنو 5 متغیره

دارای 32 خانه است و هر خانه دارای 5 همسایه است.



① $\_000\_ \Rightarrow \bar{b}\bar{c}\bar{d}$

Subject: Year: Month: Date:

همانست در 5 متغیر و بیش دیگر از جدول کارنو استفاده نشود.

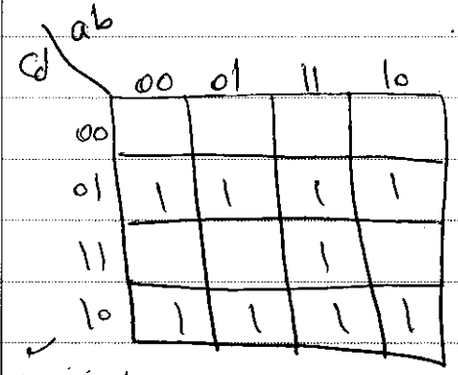
P_n هست آرایه PI هایی که n خط دارند.

P_n هست آرایه خانه‌های 2^n خانه‌ای است.

P_0 خانه‌هایی که PI هیشان خط ندارند.

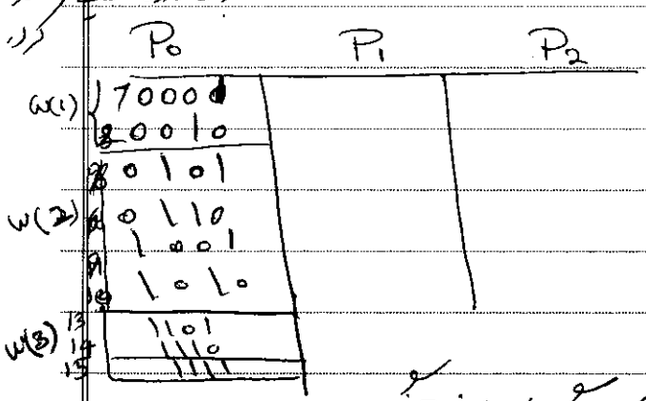
P_1 " " یک خط دارند.

P_2 " " دو خط دارند.



1, 2, 5, 6, 9, 10, 13, 14, 15

را بر وزن همینگ sort کرد.



وزن همینگ به تعداد یک‌ها در هر کد می‌گویند وزن همینگ.

Subject: Year: Month: Date:

گفته بودیم که وزنش h است و خواهیم با کدی که وزن $h+1$ ساره کند که

حصل که ساره شده است وزنش h است.

P_0

✓1	0001	
✓2	0010	
✓5	0101	
✓6	0110	
✓7	1001	1
✓10	1010	
✓13	1101	
✓14	1110	
✓15	1111	

P_1

		وزن
0_01	1,5 ✓	1
_001	1,9 ✓	
0_10	2,6 ✓	
_010	2,10 ✓	
_101	5,13 ✓	
_110	6,14 ✓	
1_01	9,13 ✓	2
1_10	10,14 ✓	
11_1	13,15	
111_	14,15	

P_2

		وزن
__01	1,5,9,13	1
__10	3,6,10,14	

b_3b_2 \ b_1b_0	00	01	11	10
00			d	
01		1	d	
11	1	1	d	d
10	1		d	d

تمام P_2 هایی که خانه هایشان D تاس است.

Don't care ها:

Don't care ها خانه هایی هستند که چه صفر باشند چه یک، دهیچگاه استفاده نکرده ایم. یعنی تراشیم صفر بدهیم که نفع ساده سازی باشد یا یک.

$f = 2,3,5,7,10,11,12,13,14,15$

آن d هایی که به نفع ساده سازی است یک در نظر می گیریم و آن d هایی که به نفع ساده سازی نیست صفر در نظر می گیریم.

تعداد PI ها: ۴ تا هیچگاه از جدول کارنو نمی رویم بگیرد از الگوریتم استفاده می کنیم.

در الگوریتم مک کلاسکی ۲ تا که همسایه هستند و چون فقط رنگ موقعیت تفاوت دارند می توانیم از ۲ تا آنی که صفر دارد یکی برداریم.

Subject: Year: Month: Date:

ورودی PI دیگر نشده پس آن PI ضروری می شود.

PI ضروری: PI هایی که فقط داخل یک PI است.

PI ضروری هم ضروری کننده، مهمتر این است که فقط داخل یک PI است.

Subject: Year: Month: Date:

الگوریتم مک‌کلوسکی تابع

$f(a,b,c,d) = 2,3,5,7,\underbrace{10,11,12,13,14,15}_{\text{dont care}}$

P_0	P_1	P_2	P_3
✓2 ورن ۱	✓2,3	2,3,10,11 ①	
✓3	✓2,10 ←		
✓5 ورن ۲	✓3,7	3,7,11,15 ②	
✓10	✓3,11 ←	3,11,7,15	
✓12	✓5,7	5,7,13,15 ④	
✓7	✓5,13	~~10,11,14,15~~	
✓11	✓10,11 ←	10,11,14,15 ⑤	
✓13	✓10,14	12,13,14,15 ⑥	
✓14	✓12,13		
✓15 ورن ۴	✓12,14		
	✓7,15		
	✓11,15		
	✓13,15		
	✓14,15		

حذف می‌شوند ← 12, 7, 11, 13

حذف نمی‌شوند

P_2 باید تمام مینترم‌ها اشتراک ○ است ○ پوشش داده شود.

⑤ تا PI داریم که در صورت کانونی حتماً باید باشد.

حال کدام از این PI ضروریند: P_2

(2,3,10,11) ← 2, 3

3,7,11,15 ← 7, 11

5,7,13,15 ← 5, 13

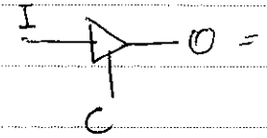
حداقل می‌توان گفت که در تابع ۴ مینترم ما ۴ PI داریم

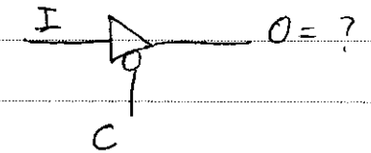
Subject: Year: Month: Date:

p-Switch (Active low) یا قطع یا اتصال است فقط 1 را عبور می‌دهد.

n-Switch (Active high) یا قطع است یا اتصال است فقط 0 را عبور می‌دهد.

منطق ۳ مقداره ← مانند:





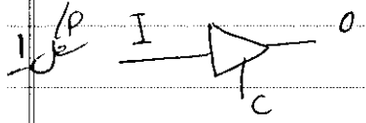
شکل یک not بافر.

شکل یک and بافر.

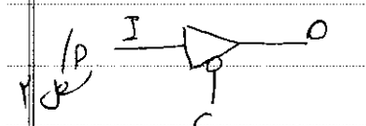
Subject: Year: Month: Date:

منطق‌های چند مقداره

منطق‌های سه مقداره روی [0, 1, Z] می‌باشند که معروف‌ترین نمونه‌ی این منطق بافرهای ۳ حالته هستند.



یا وصل است که عین یک است.



یا قطع است که Z است.

در این حالت هر چی در I بود وارد O می‌شود. $I \xrightarrow{\text{وصل}} O$

$$C=\begin{cases}1 & \text{وصل}\\ 0 & Z \text{ یعنی قطع شده}\end{cases}$$

در حالت شکل ۲ هم می‌شود not حالت اول کرد اگر C=0 بود، آنگاه بافر وصل است.

شکل ۲: $\begin{cases} C=1 & \text{قطع}\\ C=0 & \text{وصل}\end{cases}$

مجموعه اعمال منطقی کامل:

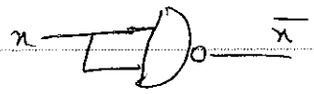
مجموعه‌ای که بتوان با آن تمام دنیا را ساخت.

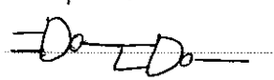
یا مجموعه {and, or, not} یک مجموعه منطقی کامل است.

و {and, not} و

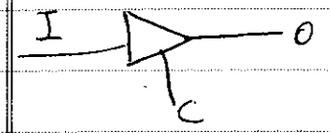
Subject: Year: Month: Date:

nand

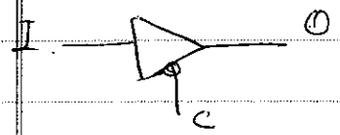




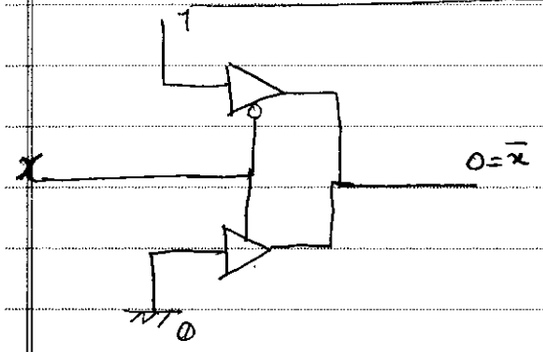
بافر سه حالته یک مدار کامل منطقی است و می‌توان تمام گیت‌ها را با آن ساخت.



C	
0	HZ
1	I



C	O
0	I
1	Z



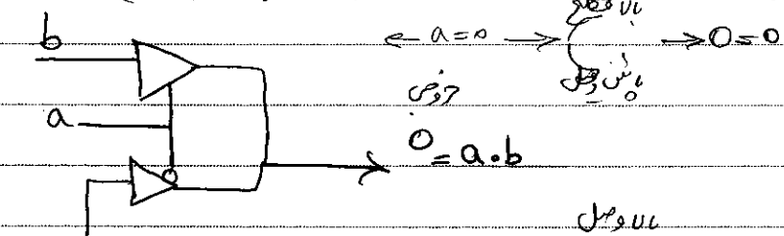
سمت بیت not از روی بافر

ورودی	خروجی
0	1
1	0

سمت بیت and از روی بافر

and ∞ a.b

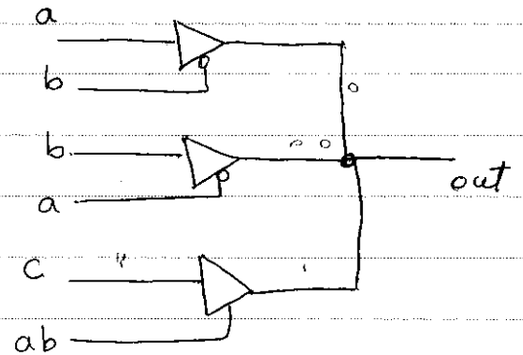
(اگر a=0 شد 0 و اگر a=1 بود شد b بدهد)



Subject: Year: Month: Date:

تمرین ۱۰

مدار زیر را داریم



خروجی تابع چیست؟

a	b	c	
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	a
1	0	1	a
1	1	0	0
1	1	1	1

$(a \oplus b) + c$

$$out \begin{cases} w & ab=0 \\ c & ab=1 \end{cases} \qquad w = \begin{cases} b & a=0 \\ 1 & a=1 \end{cases} \quad \begin{matrix} = \bar{a}b + a1 \\ = \bar{a}b + a = a + b \end{matrix}$$

منطق ۴ مقداره / جدول حقیقت بافر ۱ و منطق ۴ مقداره

Subject: Year: Month: Date:

حامد جهانگیر ۹۰،۰،۲۱ - منطقی - رایانه اصفهان

منطق چهار مقداره

در سیم می‌تواند ولتاژ مختلف باشد بدانید که در کل سیم، ولتاژ یکسان است.

U مقدار ولتاژ نامعلوم که الان حالش را نمی‌دانیم. ولتاژ یک سیم در کل و فقط یک مقدار است

مقدارهای ممکن:

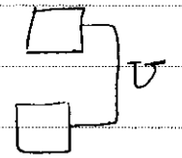
0, 1, Z, U

Z: قطع بودن

U ولتاژ نامعلوم

0 ولتاژ صفر

1 ولتاژ ۵ ولت

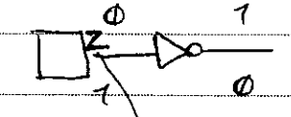


سیم B ولتاژ از سیم A می‌گیرد

نیت به A ولتاژ داده.

A به نیت ولتاژی (هر چیزی باشد) که به آن متصل می‌شود می‌دهد

اختلاف پتانسیل مقدارهایی که باید اینجا دهیم که واحدش را که کلن است را از نقطه‌ای به نقطه دیگر سیم.



Z قطع سیم در خروجی ولتاژ نامعلوم

TF داریم.

حاصل ضرب ۰ در هر چیزی ۰ است.

حاصل جمع ۱ با هر چیزی ۱ است.

منطق چهار مقداره Active high (در حالت)

جدول حقیقت ۴ تایی حالت منطق ۴ مقداره

b \ a	0	1	Z	U
0	Z	Z	Z	Z
1	0	1	U	U
Z	{0,Z}	{1,Z}	{U,Z}	{U,Z}
U	{0,Z}	{1,Z}	{U,Z}	U

وقتی b=0 حتماً قطع است. b=0 حتماً قطع است.

وقتی b=1 ولتاژ a به 0 متصل می‌شود.

b=Z پس بافر مشخص نیست خروجی چیست یا U

{U,Z} ≡ U

{هر چیزی با نامعلوم}

{U,Z} یعنی همان U یعنی نامعلوم.

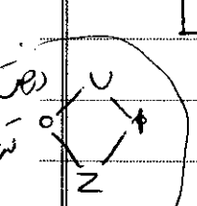
U = Z یا U

هر چیزی با U، یا شود می‌شود U.

یک سیم داریم که از یک طرف مقدار a می‌گیرد و از یک طرف دیگر b حال این سیم چه مقداری می‌گیرد؟

جدول حقیقت سیم با منطق ۴ مقداره

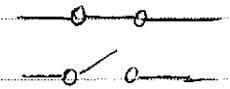
b \ a	0	1	Z	U
0	0	U	0	U
1	U	1	1	U
Z	0	1	Z	U
U	U	U	U	U



اگر یکی از سیم‌ها U باشد حتماً خروجی نامعلوم U است. وقتی a,b ورودی

Subject: Year: Month: Date:

عنصر سوئیچ، Switch



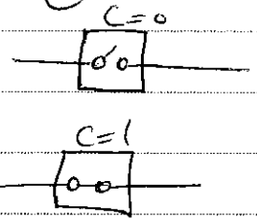
میخواهیم سوئیچ هایی داشته باشیم که الکتریسیته را روشن، خاموش شوند.

این سوئیچ‌ها دارای یک سری ورودی (کنترل) اگر c=0 سوئیچ قطع،

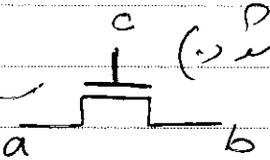
اگر c=1 سوئیچ وصل است. دقت داشته باشید این گفته به سوئیچ وصل

نیست (مثل آنکه سوئیچ را مکانیکی خاموش روشن می‌کنید اما به سوئیچ وصل

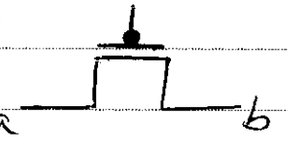
نمی‌شود.)



Active high سوئیچ



Aothive low سوئیچ

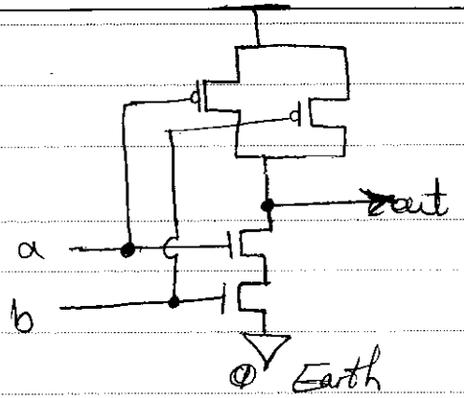


سوئیچ ها منطق کامل هستند.

برای اینکه اثبات کنیم می‌آییم با هر سری منطق کامل دیگر

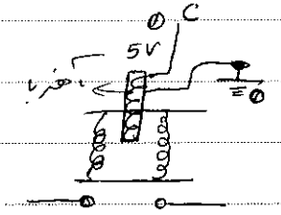
P-Switch / P-Switch / N-Switch

Subject: Year: Month: Date:



a	b	out
0	0	1
0	1	1
1	0	1
1	1	0

روشهای ساخت سوئیچ،

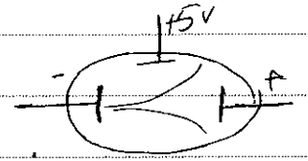


اول با رله ساختند، سپس با لامپ خلاء

c=0 → on

c=1 → off

Active low

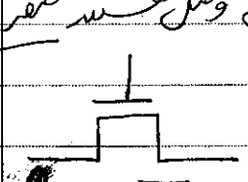


← لامپ خلاء

سپس با ترانزیستور (cmos) ساختند. و هم اکنون هم با ترانزیستور ساخته می‌شود.

N-Switch

وقتی که قطع هستند چیزی را وصل می‌کنند و وقتی وصل هستند صفر را وصل می‌کنند (یعنی 1 را وصل می‌کنند)



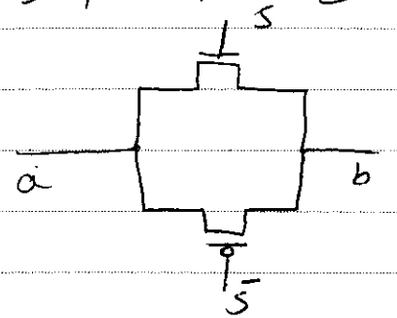
P-Switch

منطق ۷ مقداره چگونه است؟

Subject: Year: Month: Date:

این مدل سوئیچ ها ساخته شده با ترانزیستور می باشد. (مانند سوئیچی که با دست ما کار می کند)

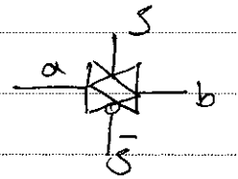
حال آنها می توان سوئیچی ساخت با ترانزیستور که هم صفر و هم یک را عبور دهد



C-Switch

T-gate

نحوه ساخت یک T-gate



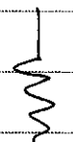
منطق ۷ مقداره

در این منطق مقاومت نسبت بیت است. (این منطق از سه منطق ۳ مقداره ساخته شده است.)

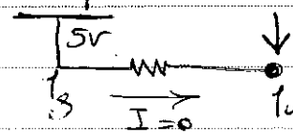
مقدارهای منطق ۷ مقداره.

قوی ← 0_S 1_S U_S

ضعیف ← 0_W 1_W U_W Z



Subject: Year: Month: Date:



همان در این مقاومت صفر ولتاژ دو سر سیم پیچ 1w است.

هر دو طرف 5 است و مقدار $\underline{1}$ می گیریم اما آنکه با $\underline{1}$ ، مقدار $\underline{1}$ ساده است

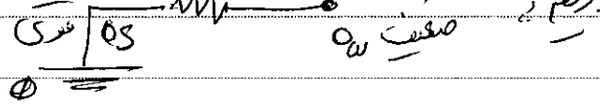
می شود 1 قوی، اما آنکه با مقاومت $\underline{1}$ ساده می شود $\underline{1}$ ضعیف

($\underline{5}$ ولت هر دو طرف $\underline{1}$ است آنکه با منبع ولتاژ $\underline{1}$ ساده (از طریق سیم)

منبع ولتاژ وصل است می شود 1 قوی) آنکه با مقاومت به منبع

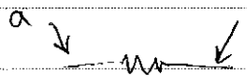
ولتاژ وصل شده است می شود 1 ضعیف)

برای صفر هم همین صورت داریم.

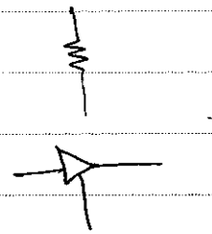


Subject: Year: Month: Date:

a	b
U_S	U_W
1_S	1_W
0_S	0_W
U_W	U_W
1_W	1
0_W	0
Z	Z

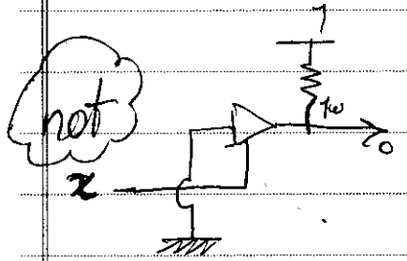


در منطق ۷ مقداره نیز مقاومت و ولتاژ حالت یک منطق کامل است



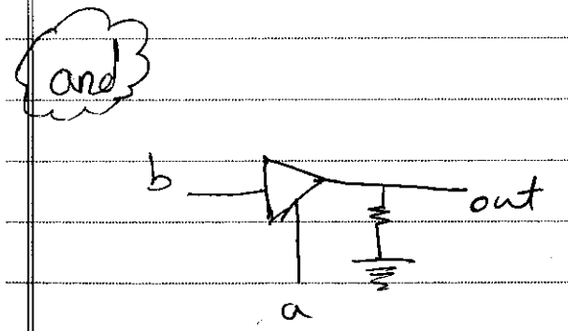
Resident + Buffer 3-State + Active low

هر دو با هم منطق کامل هستند



x	
0	5V
1	0V

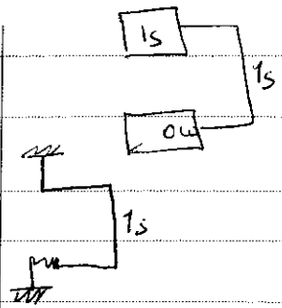
تمامی خروجی‌ها Strong می‌دهند.

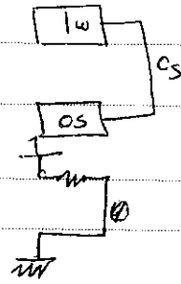


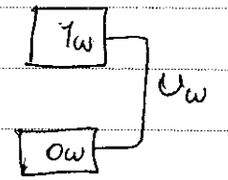
$y = ab$

Subject: Year: Month: Date:

در منطق ۷ مقداره، سیگنال را می‌توان مقاومت و ولتاژ در نظر گرفت



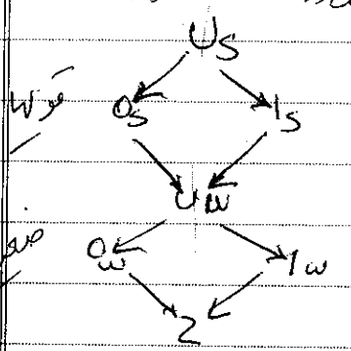




$$\frac{V-0}{R_2} = \frac{5}{R_1+R_2} \Rightarrow V = \left(\frac{R_2}{R_1+R_2}\right)5$$

$0 < \frac{R_2}{R_1+R_2} < 1$

در منطق ۷ مقداره



توان مقاومت ۷ مقداره

توان مقاومت ۴ مقداره

$Z, 0_W \rightarrow 0_W$

$Z, U_W \rightarrow U_W$

نمی‌توان از مقاومت برای تقویت سیگنال استفاده کرد.

در منطق ۷ مقداره به مقاومت می‌توان نیز تقویت کننده داد.

Subject: Year: Month: Date:

Combinational Components — مدارات ترکیبی

برای ساخت یک سیستم دیجیتالی، یک روش ۱- اتصال گیت‌ها به هم که به این طراحی سطح گیت گوییم.

۲- طراحی سطح مداری که در آن از المان‌های گیت صرف نظر شده و مدار را با چیزی که در هم نظر داریم تا مدار خود را پیاده‌سازی کنیم. اجزای روی طراحی سطح مداری است، مدار منطقی طراحی سطح گیت است.)

و چون بدون حافظه هستند می‌گوییم مدارات ترکیبی (یعنی Latch F.F ندارند).

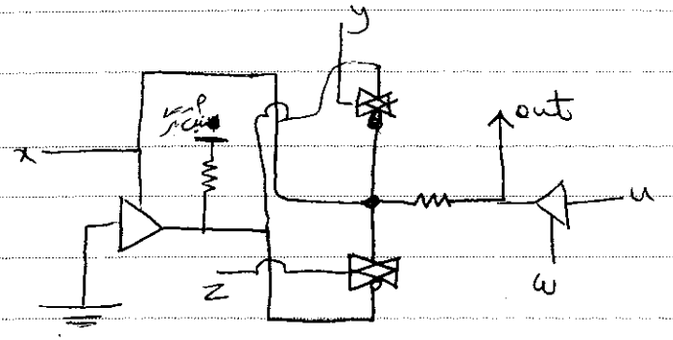
مدارات که دارای حافظه هستند می‌گوییم مدارات ترتیبی.

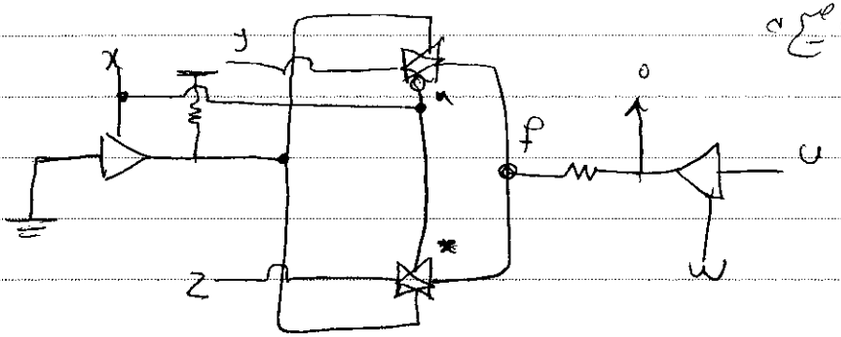
ترکیبی = بدون حافظه

ترتیبی = حافظه دارد.

Subject: Year: Month: Date:

خروجی تابع ورودی است.





$$f = \begin{cases} y & x=0 \\ z & x=1 \end{cases} \Rightarrow \bar{x}y + xz$$

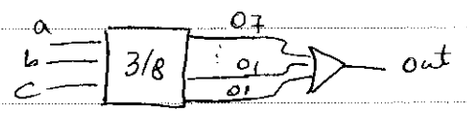
$$o = \begin{cases} v & w=1 \\ f & w=0 \end{cases}$$

$$o = wv + \bar{w}f$$

$$o = wv + \bar{w}\bar{x}y + \bar{w}xz$$

Subject: Year: Month: Date:

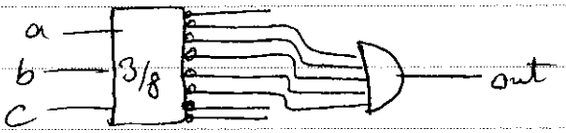
$f(a,b,c) = \sum m(0,1,7)$ Active high + OR



Min

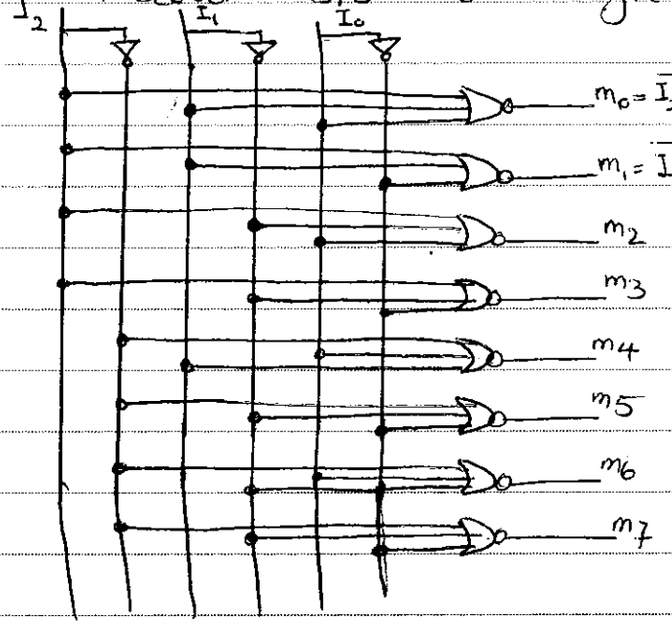
$f(a,bc) = \prod M(2,3,4,5,6)$ Active Low + And

Max



Decoder 3/8 Active high

nor بیت



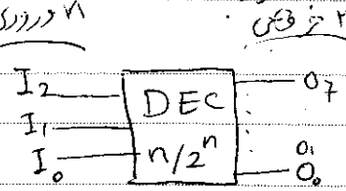
$m_0(I_2 I_1 I_0) = \bar{I}_2 \cdot \bar{I}_1 \bar{I}_0 = \overline{I_2+I_1+I_0}$

Subject: Year: Month: Date:

Dec چیست؟

(DEC) Decoder

به صورت یک Box فرض می‌شود که n ورودی و 2^n خروجی دارد. هر کدام از خروجی‌ها که سیگنال ورودی داده شده است Active می‌شود.



$I_2\ I_1\ I_0$
$1\ \ 0\ \ 1 \rightarrow o_5$ ✓

Active high

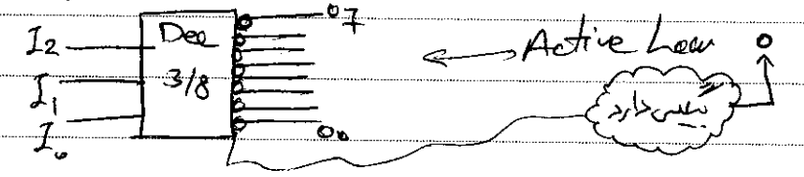
$110 \rightarrow o_6$ عدد صفر + بقیه یک

آن که فعال می‌شود ! می‌شود بقیه صفر

Active Low

$110 \rightarrow o_6$ عدد صفر + بقیه یک

آن که فعال می‌شود ۰ می‌شود بقیه نه



نام‌گذاری Dec ها

Dec از نوع Active high که ستیم‌ها را با OR می‌کند.

Component مولد ستیم

Dec از نوع Active low که ستیم‌ها را با And می‌کند.

پیچیدگی Dec چگونه است؟

Subject: Year: Month: Date:

#GI

شاخص پیچیدگی مدار برابر است با تعداد ورودی‌ها × تعداد خروجی‌ها

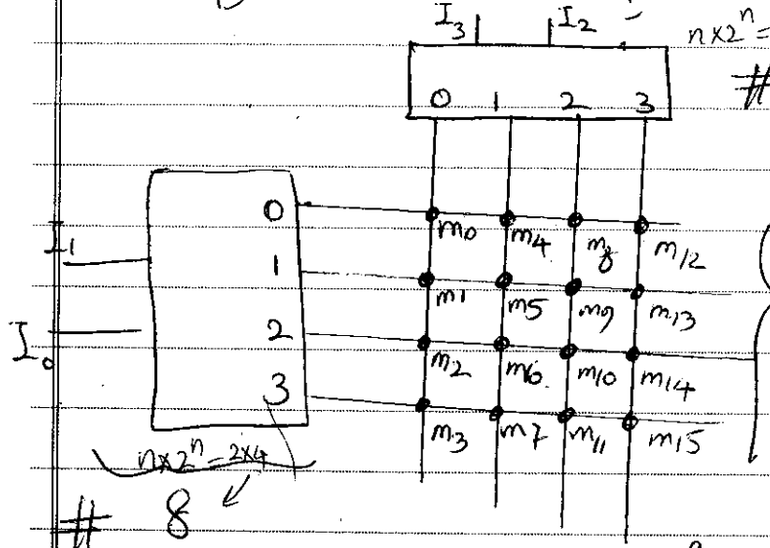
تعداد خروجی‌ها 2^n تعداد ورودی‌ها n ×

$\Rightarrow n2^n = GI$

روش ساده سازی Dec برحسب اینکه سرعت ↓ یا پیچیدگی ↓ (پیچیدگی کمتر می‌شود ساده‌تر می‌شود اما سرعت ↓)

روش ساده سازی دیکودر 4×16 طبقه ↑

برای ساخت Dec 4×16 بیایم از Dec 2×4 استفاده کنیم.



8 ✓

ورودی‌ها $\times$ تعداد خروجی‌ها

$\frac{16\times2}{32}$ → ورودی and گیت

Subject: Year: Month: Date:

از روش ساده سازی ماتریس، برای ساده‌سازی Dec استفاده می‌شود زیرا تعداد ترانزیستورها خیلی کم می‌شود با اینکه تعداد طبقات زیاد است اما برای Dec استفاده می‌شود.

بعضی از Dec ورودی Enable دارند.

اگر En=0 همه‌ی خروجی‌ها غیرفعال می‌شوند (در A.low همگی 0 و در A.High همگی 1 می‌شوند).

اگر En=1 همه کارهایش را انجام می‌دهد.

En + or بیاید ⟸ Active low

En • and بیاید ⟸ Active high

از نظر پیچیدگی آنکه En دارد پیچیده‌تر از آنکه En ندارد اما در نهایت پیچیدگی هر دو $n2^n$ است.

$n2^n + 2^n \times 2$ (2 ورودی)

(En and یا or بابر)

$O(n2^n)$

خروجی‌ها or می‌شوند با Active low در En

Subject: ..... Year: ..... Month: ..... Date: .....

جلسه پنجم

تست سال ۸۴ شماره ۶۴

درستی نرستی ایشگل های رویروی گذشته

Dec دارای ـــــ گیت است.

Mux دارای ـــــ گیت or ، ـــــ گیت and ،

ـــــ ورودی گیت or ، ـــــ ورودی گیت and دارد

Dec اول ـــــ طبقه بود، ویژگی ؟ GI#

رویکرد ماتریس ساده کاربرد شد ـــــ طبقه اما GI# ✓

Mux ـــــ طبقه است، GI# ؟

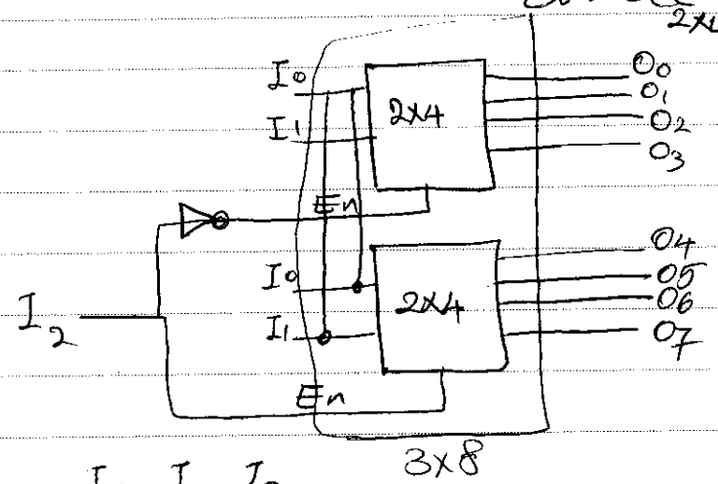
تعریف = ( )O

اما Mux خروجی ساده است.

Subject: ..... Year: ..... Month: ..... Date: .....

ترکیبی Dec2x4 بعنی Dec3x8

سازنده 2- Dec$_{2x4}$ است.



I_2	I_1	I_0	
0	0	0	O_0
0	0	1	O_1
0	1	0	O_2
0	1	1	O_3
1	0	0	O_4
1	0	1	O_5
1	1	0	O_6
1	1	1	O_7

Subject: Year: Month: Date:

برای Mux وصل-غیر وصل نداریم انتخاب داریم.

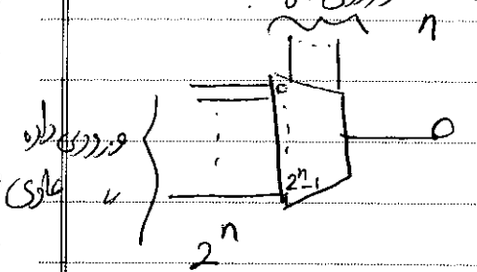
Mux

2 دسته ورودی دارد. 1- ورودی های داده (عادی)
2- ورودی های انتخاب

تعداد ورودی های انتخاب n تا باشد تعداد ورودی داده 2^n است.

انتخاب

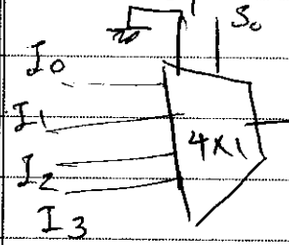
* با توجه به ورودی های انتخاب یکی از ورودی های داده وصل می شود و به خروجی وصل می شود.



نحوه Mux منطق کامل است.

هر Mux نزدیک Mux کوچکتر در دلش است.

Mux 4x1 ، Mux 2x1 است.



اگر $S_1=0$ باشد با توجه به S_0 یا I_0

Subject: Year: Month: Date:

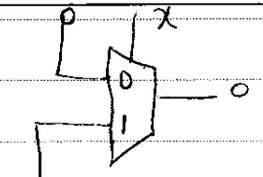
سیم بازی

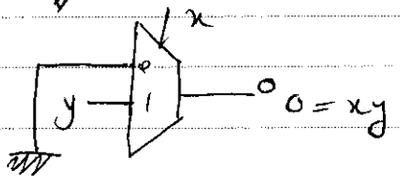
$x=0 \rightarrow o=1$

$x=1 \rightarrow o=0$

$o=\bar{x}$

زمین



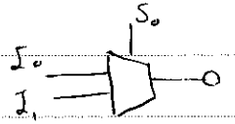


$o=xy$

x	y	o
0	-	0
1	y	y

دیود به صورت وصل

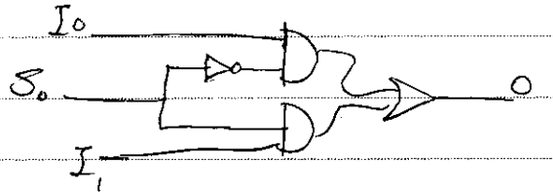
Mux منطق کامل است.



$o=\begin{cases} I_0 & S_0=0 \\ I_1 & S_0=1 \end{cases}$

$o=\bar{S_0}I_0+S_0I_1$

پیاده سازی با ~~NAND~~ گیت



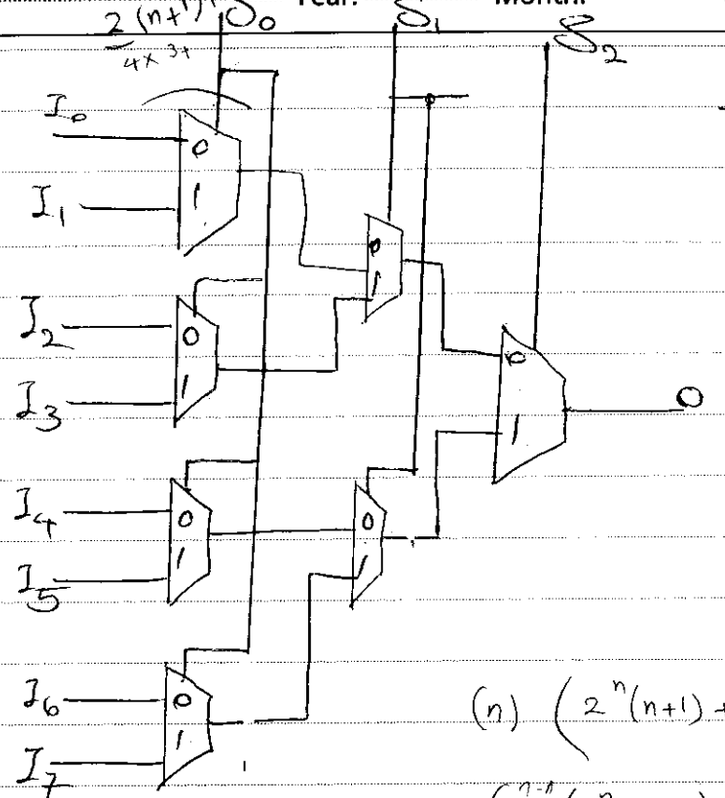
فرمول کلی Mux

SOP

$$o=\sum_{j=0}^{2^n-1} m_j(S_{n-1},\dots,S_0)\,I_j$$

کانونیکال SOP ~~Canonical~~ SOP است زیرا ساده ترین SOP است زیرا وجود هیچ گیتی نداریم ساده تر

Subject: ..... Year: ..... Month: ..... Date: .....



$2^n(n+1)+2^n$

$(n)\left(2^n(n+1)+2^n\right)$

$(2^{n-1})(2^n(n+1)+2^n)$

Mux

$8 \times 7 \quad 6\left(2^n-1\right)\left(2^1(2)^6+1\right)$

2^3

این مدار 3 طبقه Mux است، هر Mux خود از 2 طبقه

است ← Mux ی سود 7 طبقه.

$2^n(n+1)+2^n$

$1 \to 2(2)+2=6$ Mux 2x1 ← $6 \times (2^n-1)$ تعداد Mux (2x1)

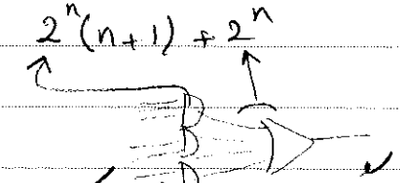
کل پیچیدگی می‌شود.

Subject: ..... Year: ..... Month: ..... Date: .....

پس Mux از این ساده‌تر نمی‌توان کرد و به این می‌گویند

$O=\sum_{j=0}^{2^n-1} m_j(S_{n-1},\dots,S_0)\, I_j$ کسری and و or

پیچیدگی $O(n2^n)$



یک بیت or دارد.

2^n تا or و 2^n تا and که هر یک $(n+1)$ ورودی

دارد.

پیچیدگی برای Mux ماتریس Dec نیز کم شده است. حال در

این Mux ساده کنیم در همان 2 طبقه هست و بیش از 2 طبقه است

و سرعت بالاتر است.

Mux ① بیت or که دارای 2^n ورودی است. 2^n تا بیت and که دارای $(n+1)$ ورودی است.

Dec خروجی 1 طبقه بود اما پیچیده یعنی GI زیاد بود آمدیم به روش

ماتریس ساده کردیم GI هایش کم شد اما 2 طبقه شد GI $(n2^n)$

Dec $O(n2^n)$

Mux ← 2 طبقه است. or ← 1

and ← 2^n

(حتماً یک ست MUX می‌آید)

Subject: ..... Year: ..... Month: ..... Date: .....

← روش MUX ضلبی ساده‌تری می‌شود حجم سخت افزار لغو می‌شود اما سرعت

ضلبی کم می‌شود.

فرمول شانن

$$f(a,b,c) = \begin{cases} f(a,0,c) & b=0 \\ f(a,1,c) & b=1 \end{cases} = \bar{b} f(a,0,c) + b f(a,1,c)$$

$$f(x_{n-1}, x_{n-1}, \dots, x_1, x_0) = \bar{x}_i \cdot f\big|_{x_i=0} + x_i \cdot f\big|_{x_i=1}$$

که برای یک متغیر شبیه هم می‌شود.

مثال:

$$f(a,b,c,d) = (a \oplus b) + \bar{c} d$$

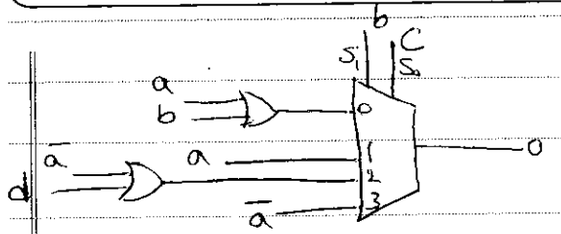
$$f(a,b,c,d) = \begin{cases} a \oplus b & d=0 \\ a \oplus b + \bar{c} & d=1 \end{cases} \Rightarrow \bar{d}(a \oplus b) + d[(a \oplus b) + \bar{c}]$$

$$f(a,b,c,d) = \bar{b}\bar{c} f(a,0,0,c) + \bar{b} c f(a,0,1,c) +$$

$$b\bar{c} f(a,1,0,c) + b\bar{c} f(a,1,1,c)$$

بلک ساخت تابع n متغیره نیز همین است Mult

Subject: ..... Year: ..... Month: ..... Date: .....



که تابع را خواهید n متغیره را بدون هیچ سخت افزار دیگری بسازید نیاز هست

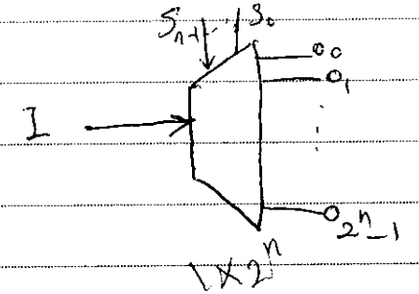
(2^{n-1} X 1) Mux کافیست.

PEMUX برعکس MUX می‌باشد

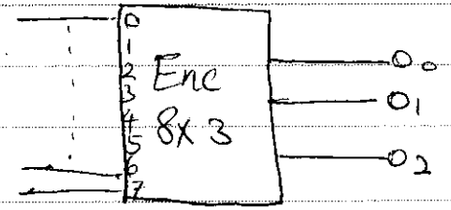
یک ورودی عادی (داده) دارد و n ورودی انتخاب دارد. 2^n خروجی دارد.

سیگنال‌های $S_{n-1} \dots S_1 S_0$ عددی که انتخاب شده و واقع O_n ها انتخاب می‌شود

که باعث می‌شود ورودی منتقل شود. و بقیه خروجی غیرفعال هستند.



Subject: Year: Month: Date:



8 تا

2^8 تا حالت ورودی داریم که فقط 8 تا مجاز و بقیه don't care.

جدول صحت:

I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0	O_0	O_1	O_2
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

$O_2 = I_4 + I_5 + I_6 + I_7$

$O_1 = I_2 + I_3 + I_6 + I_7$

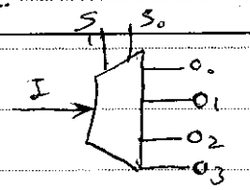
$O_0 = I_1 + I_3 + I_5 + I_7$

پیاده‌سازی (Dec $2^n \times n$)

$\left(\frac{n \times 2^n}{2}\right) + n2^{n-1}$ Enc

پیاده‌سازی $O(n2^n)$

Subject: Year: Month: Date:



Demux عکس Decoder است و در واقع

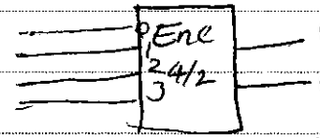
I همان (En) است.

هر کدام از Deco ها برای یک En می‌شود Demux یک‌بیتی (فقط یک En دارد)

Demux/Dec دارای En

Dec فاقد En

Encoder $2^n/n$



فقط یکی از ورودی‌ها می‌تواند یک باشد و بقیه نمی‌توانند یک باشند و در غیر اینصورت ورود غیرمجاز است که خروجی آن می‌شود don't care.

اگر دو تا ورودی 1 باشد خروجی don't care (یعنی چیزی نمی‌تواند باشد و در این حالت وظیفه‌ی خاصی ندارد و اعتراض نکنید)

Subject: Year: Month: Date:

Encoder اولویت دار چگونه است؟

Encoder اولویت دار:

Encoder $n2^n$ یک

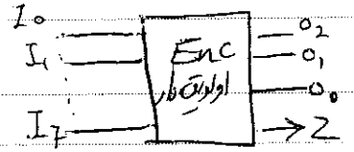
Encoder اولویت دار به مداری گفته می‌شود که اگر بیش از یک ورودی فعال باشد، ورودی با بالاترین اولویت را کد می‌کند.

اگر n ورودی داشته باشیم، n حالت اولویت مختلف داریم که ۲ تای آن را استفاده می‌کنیم.

Enc اولویت‌دار مانند Enc معمولی، فقط تفاوت اینکه در Enc معمولی فقط یک ورودی فعال است اما در Enc اولویت‌دار می‌توانند چند ورودی فعال باشند و هیچ کدام از ورودی‌ها اولویت ندارند.

فعال نباشد که در اینصورت هم مجاز است که در Enc اولویت

صفر و Z می‌شود یعنی هیچ یک از ورودی‌ها فعال نیست.



Dec $n \times 2^n$

Encoder $2^n \times n$

Cube کردن سطرها:

وقتی دو سطر حقیقت مشابه داشتند آنها را ادغام کرده و Cube می‌کنیم.

1	0	0	0
1	1	0	0

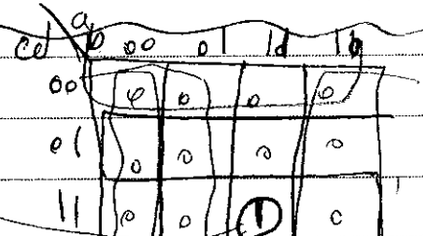
ادغام کردن Cube

1	X	0	0

a	b	c	d	O
0	X	X	X	0
X	0	X	X	0
X	X	0	X	0
X	X	X	0	0
1	1	1	1	1

گیت and

کمااینکه بخواهیم جدول حقیقت بنویسیم $16 = 2^4$ سطر می‌خواهیم اما با ادغام تبدیل به ۵ سطر شد.



این

Subject: ..... Year: ..... Month: ..... Date: .....

جلسه ششم

در اولویت دادن به Encoder ها دو اولویت صعودی، نزولی بهم.

PENC Encoder اولویت دارنده

فرض کنید بیشترین اولویت برای I_0 باشد.

صدول حقیقت آندر اولویت دارنده

I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	O_2	O_1	O_0	Z
1	x	x	x	x	x	x	x	0	0	0	128
0	1	x	x	x	x	x	x	0	0	1	64
0	0	1	x	x	x	x	x	0	1	0	32
0	0	0	1	x	x	x	x	0	1	1	16
0	0	0	0	1	x	x	x	1	0	0	8
0	0	0	0	0	1	x	x	1	0	1	4
0	0	0	0	0	0	1	x	1	1	0	2
0	0	0	0	0	0	0	1	1	1	1	1

این ۱ ها را زیاد کنید تا کمتر شود

256

Subject: ..... Year: ..... Month: ..... Date: .....

وقتی دوصدل حقیقت Cube می‌کشیم حواسمان Cube کردن دوصدل کاربرداست.

نکته: دوصدل حقیقت عبارت ساده شده را می‌بینیم اما دوصدل مرجع بهترین است اما دوصدل کارنو عبارت ساده شده درست آمده ساده‌ترین است.

حال عبارتهای درست آمده از صدل بل

خروجی O_2 این ... $O_2 = \bar{I}_0\bar{I}_1\bar{I}_2\bar{I}_3 I_4 + \bar{I}_0 I_1 \bar{I}_2\bar{I}_3 I_4 I_5$ ...

خروجی O_1 این ... $O_1 = \bar{I}_0\bar{I}_1 I_2 + \bar{I}_0 I_1 \bar{I}_2 I_3$ ...

حال مدار در سطح 5 عبارت خروجی ساده شده 3 تا منتهی می‌شود.

Component هایی ترتیبی که قبلیم بهم کنیم Comp. ترتیبی ایجاد می‌کند.

آشکار ساز حافظه دارد

Comp ترتیبی

شمارنده حافظه دارد / واحدهای حافظه

مدارات ترکیبی حافظه دارند / مدارات ترکیبی حافظه ندارند

Subject: Year: Month: Date:

ترتیبی سنکرون : داده‌های حافظه این مدارها باید تغییر کنند ، ایرخواست

تغییر لبه همه باهم تغییر میکنند . (هماهنگ زمان)

ترتیبی آسنکرون : هر داده حافظه در هر زمان که خواستند تغییر میکنند

و تغییر یکی به یکی دیگه ربطی نداره . (ناهماهنگ)

مدارات ترکیبی که داخل مدارات ترتیبی آسنکرون استفاده میشوند باید

یک سری ویژگی داشته باشند (که در این ویژگی را نداشته باشد آسنکرون ایراد

می‌گیرند) این مدارات ترکیبی نباید Hazard (خطر) داشته باشند.

Hazard

به مدار ترکیبی میگیم هزارد داره که glitch (اشکال پرو)

در سیگنال خروجی داشته باشه .

از این به بعد تو دل سنکرون استفاده میکنیم اشکالی نداره

glitch چیست ؟ زمانی که انتظار داریم یک سیگنال از یک

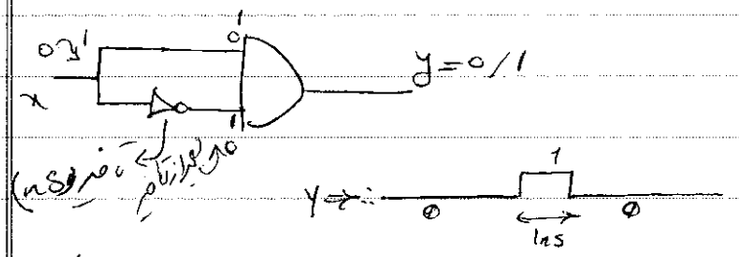
Subject: Year: Month: Date:

glitch چیست؟ Race و glitch علت هزارد است.

که مقدار درست دارد به این glitch میزند.

(glitch مقدار غلط ← موقتی)

مثال :



این مدار glitch دارد.

Hazard ← الی

[رفع اشکال Hazard : Hard Fresh میزنند]

روش تشخیص Hazard در مدار ترکیبی .

(سابقه)

Race چیست ؟ علت glitch ، Race است . هرگاه در یک مدار دو ورودی

در یک مدار ۲ یا چند سیگنال را تغییر دهیم آنگاه در آن مدار Race

ایجاد شده است . و می‌توانیم این کار را انجام دهیم تا بفهمیم زودتر از

کدام تغییر می‌کند آنکه زودتر تغییر می‌کند پدیده است . و آنکه

دیرتر تغییر می‌کند پاسخ است . (برای کیت سوسیت نداریم .)

Subject: Year: Month: Date:

از روش تشخیص Hazard در مدارات Sop و POS

یک استراتژی این است که ببینیم Race مجاز نیست زیرا Race میتواند

سبب glitch و هرزنت ← Hazard داشته باشیم.

به یک قانون می‌گوییم که اگر ما ترکیبی داریم حتی اگر هم ورودی همزمان تغییر

کنند یعنی از هر لحظه از زمان فقط می‌تواند یکی از آن‌ها را تغییر دهد.

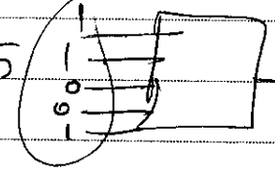
Hazard functional ← Race بین ورودی‌های مدار ترکیبی که قانون

خوب را بیان نماید انجام دهیم. (اگر که Race بین ورودی‌ها اجازه دهند)

Hazard Fonctional

که نباید بین ورودی‌ها یک سری ورودی‌ها را که ببینیم مشکل در مدار زیر عدد

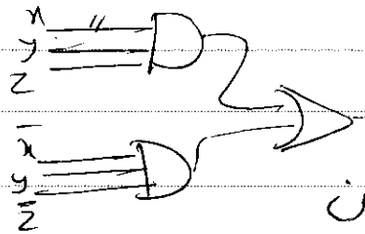
25 را می‌خواهیم ! کنیم باید بین آن یک عدد اکی 17



ببریم بعد ! .

11001 25

Subject: Year: Month: Date:



مشخص می‌کنیم مدار روبرو

x را تغییر می‌دهد که باید

ایجاد Race بین x و x می‌شود چون

۲ پایه ورودی می‌خواهند همزمان تغییر کنند. این است که این

مدار Functional ست بگ چون فقط ورودی x تغییر کرده

اما در کل مدار ۲ پایه ورودی عوض می‌شود این می‌شود Non Functional

آپن تری Functional قانون داریم طرفی آید قانون رعایت می‌کند اما اگر نمی‌شود Non Functional

روش تشخیص مدار Non-functional .

در روی یک Sop مدار ۲ طبقه طبقه اول and طبقه دوم or.

هر دکارنو هر Cube یک بیت and است.

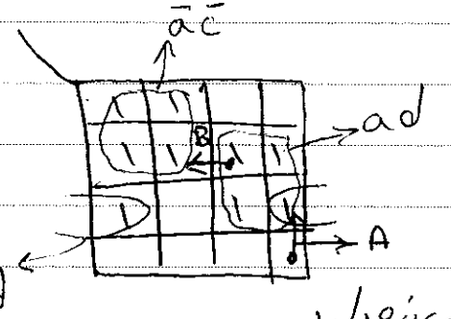
وقتی یکانه‌خانه‌های کارنو هیم آن بیت and ۱ است که داخل آن

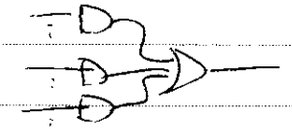
Cube ببریم

Subject: Year: Month: Date:

Race غیرحرانی

Race یک Race وجود دارد اما ایجاد glitch نمی‌کند. چون تأثیری در خروجی و ایجاد خطا نمی‌گذارد.





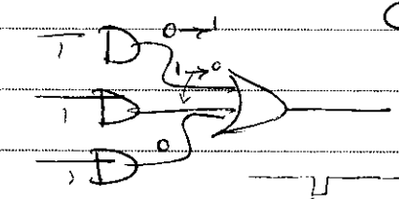
Race غیرحرانی مقدار مدار بستگی به پرنده دارد.

(A) → Race داریم → Race اما غیرحرانی (مانند سن روی Cube)

(چون هرکدام پرنده رفتند فرقی ندارد.)

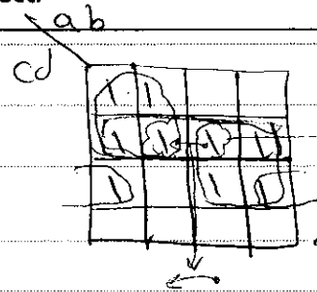
(B) Race داریم → Race حرانی → glitch ایجاد

از یک Cube می‌خواهیم یک تو خالی Cube

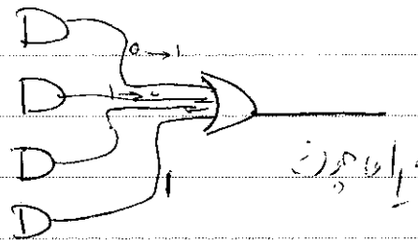


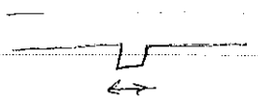
سه سر ورودی یک Cube می‌خواهد صفر

P سه یک و سه ۱ اما ضرورتی به ۱ باشد، اگر یک فقط ۲ تا صفر باشد



رفع هازارد با اضافه کردن Cube می‌شد که البته آن Cube بزرگتر شده هزینه است

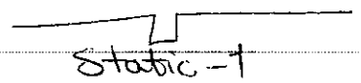




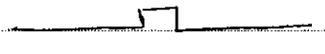
این سه ورودی Cube دارد Race ایجاد می‌کند

یکی از ورودی‌ها ۱ است

glitch (Sop) $\overset{Min}{\cdots}$



glitch (POS) $\overset{Max}{\cdots}$

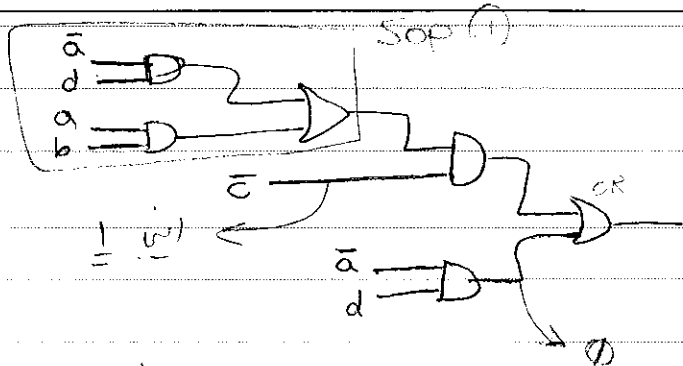


برخی Sop هازارد داشته باشند از نوع Static-1 است

برخی POS هازارد داشته باشند از نوع Static-0 است

هست هازارد سال ۸۶

Subject: .... Year: .... Month: .... Date: ....



abcd → abcd

1101 → 0101

① این یک sop است بررسی می‌کنیم که glitch داریم یا نه و در اینجا

and یعنی c̄ جایی glitch رخ می‌شه که 1 باشد یعنی c=0 و در گیت

OR یعنی (ād) وقتی 0 باشد رد می‌کند پس در نتیجه:

$c=0$, $\bar{a}d=0$ ⇒ 1101 → 0101

Subject: .... Year: .... Month: .... Date: ....

تست ۶۳ سال ۸۶

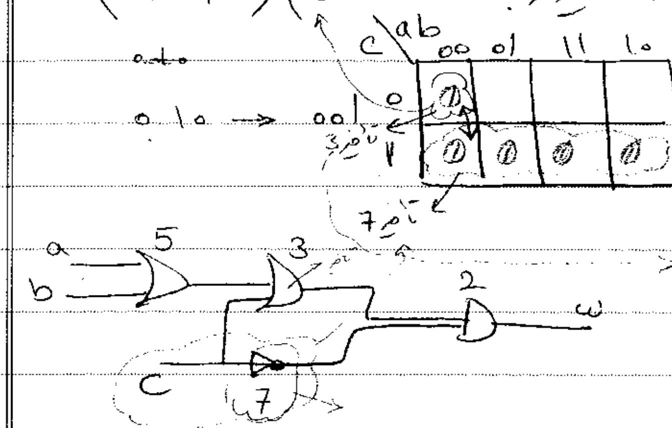
~~(a+b)+~~ همیشه روی کارنو باید هازاردها را تشخیص دهیم

در ضمن جایی مشخص دارد که Cube تا تأخیر بیشتر Cube = $(a+b+c)\cdot(\bar{c}$

000

010 → 001

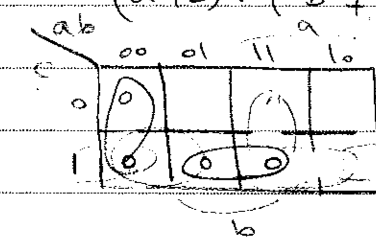
تأخیر بیشتر بوده



این فقط c تغییر می‌کند که تأخیرش

تست ۶۴ سال ۸۲

$(a+b)\cdot(\bar{b}+\bar{c})\ (a+\bar{c})$ ⇒



$(a+b)(\bar{b}+\bar{c})\ (a+\bar{c})$

↓

گزینه ۲ Nand$(\bar{a}\cdot c)$

$(a\cdot\bar{c})$

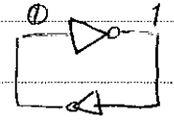
Subject: Year: Month: Date:

عناصر حافظه‌ای

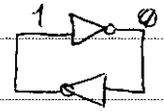
حافظه چیزی است که اطلاعات را بتوانه در خودش ذخیره کند. هر سیستمی که حافظه بتواند ۲ حالت پایدار داشته باشد می‌تواند حافظه باشد.

حالت پایدار: سیستم یک حالت دارد اگر عامل خارجی اثر نگذارد و سیستم به کار خود ادامه دهد می‌شود حالت پایدار.

حالت ناپایدار: اگر عامل خارجی آمد سیستم از حالت خود خارج شد تا وقتی که عامل خارجی باشد وقتی عامل خارجی رفت و سیستم دوباره به حالت قبل رفت می‌شود حالت ناپایدار.



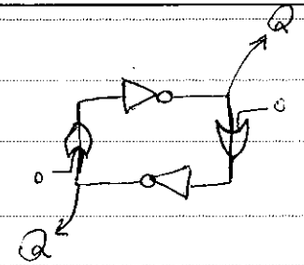
back-to-back not gates



یک مدار خیلی ساده که دو حالت پایدار دارد

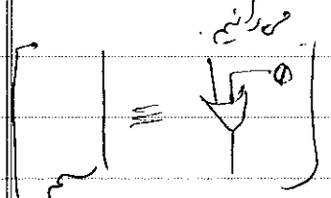
می‌تواند نقش حافظه را بازی کند.

اگر بخواهیم روشن نوشتن توی حافظه وجود داشته باشد باید همان نوشتن را بدون وجود داشته باشد می‌توانیم بنویسیم چون



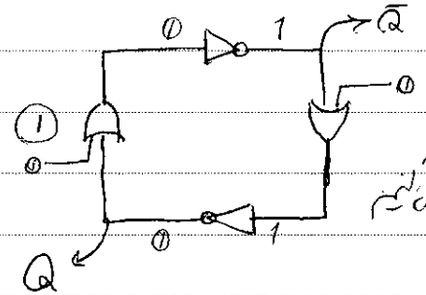
back to back نوع مستقل مدار نوشتن

سه تا گیت or داریم که برای یکی از ورودی‌ها صفر باشد



می‌دانیم

حال جهت تغییر کافیست یکی از ورودی‌های گیت or را تغییر دهیم به ۱ و آن را دوباره صفر کنیم همان حالت می‌ماند.



می‌خواهیم 01 → 10 شود پس ① می‌کنیم 1

اگر بخواهیم تو ... bit محتوای حافظه حفظ شود باید هر دو or صفر شوند. اگر بخواهیم ۱ بنویسیم باید or سمت چپ ۱ شود و اگر بخواهیم صفر شود باید or سمت راست ۱ شده.

پس ورودی or سمت راست می‌شود Reset. چون این شد تابع را صفر می‌کند.

پس ورودی or سمت چپ می‌شود Set. چون این شد

Subject: Year: Month: Date:

جلسه هفتم: فلیپ فلاپ Flip-Flop

فلیپ فلاپ چیست؟ چه فرقی با latch دارد؟

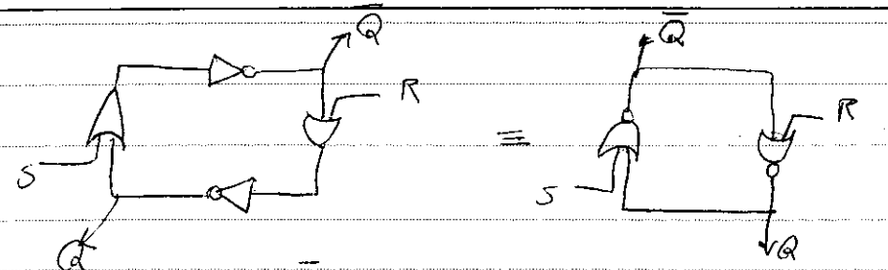
مدل ریاضی FSM:

در این مدل زمان گسسته است. (در صورتی در سیستم می‌توان از FSM استفاده کرد که زمان گسسته باشد.) روش کار زمان را پیوسته در نظر گرفته سپس کار می‌کنیم.

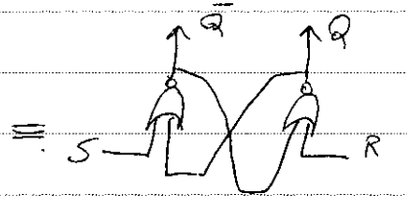
clock یک سیگنال پریودیک و منظم است و وصل کرده به latch تا با آن تغییر حالت به هنگام این حالت گسسته ایجاد کرده و حال می‌توانیم از FSM استفاده کنیم.

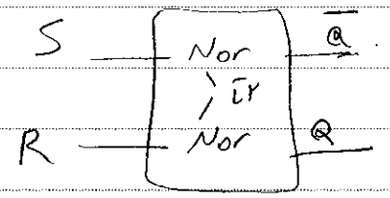
پس برای استفاده از مدل FSM باید حالت گسسته را توسط clock ایجاد کرد و توان مدار سریع را ایجاد نمود.

Subject: Year: Month: Date:



که این یک latch است.





latch

نوع SR دارای ۲ ورودی است.

FlipFlop چون است که امکان اعمال را دارد.

فلیپ فلاپ SR دارای ۲ ورودی است. (سطح ۱)

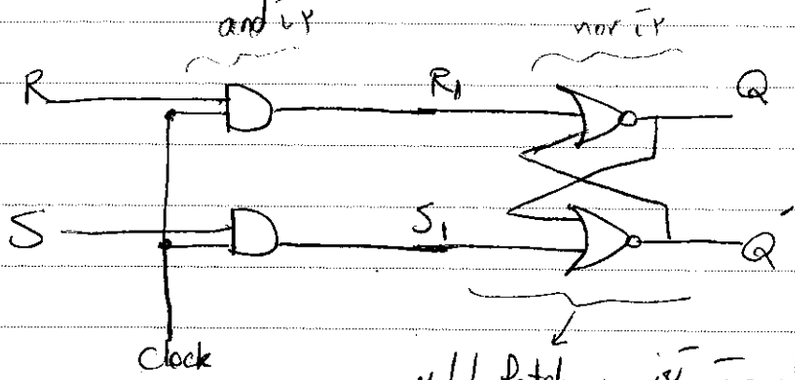
در حالت فلاپ SR با سطح بالا ورودی باید and انجام گیرد

Subject: Year: Month: Date:

Flip Flop چیست؟

clock همان latch است که اوکان اعمال کلک هم دارد.

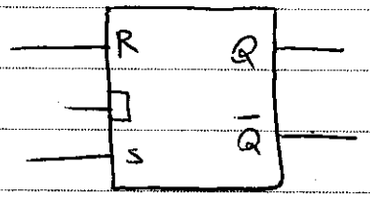
1 Flip Flop (حساس به سطح clock)



clock=1 شد حق تغییر در latch داریم.

clock=0 شد حق تغییر در latch نداریم و حفظ مقدار می‌شود در latch

به کل این ساختار Flip Flop گویند.

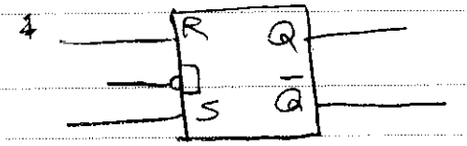


(1 حساس به سطح Flip-Flop)

خروجی DFF حساس به سطح چگونه است؟

Subject: Year: Month: Date:

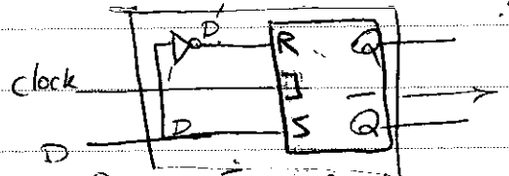
حساس به سطح 0 کافیست به and ورودی not اضافه کنیم.



(حساس به سطح 0 Flip_Flop)

D_Flip Flop

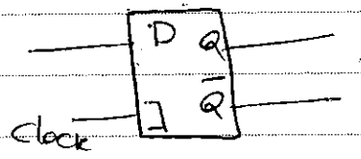
وقتی clock=1 هرچی در ورودی بوده به داخل FlipFlop منتقل شده است.

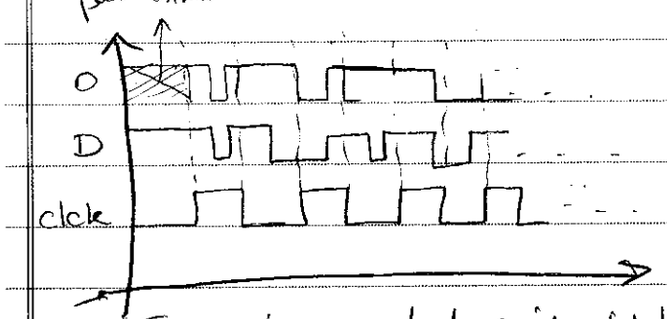


D به S و D' به R می‌شود.

DFF حساس به سطح 1

اگر clock=1 شد هرچی به D بدهیم به داخل FF منتقل می‌شود.





نکته اول Unknown چون clock=0 و حاضری از مقدار قبلی D نداریم و نمی‌دانیم می‌باید بیاوریم.

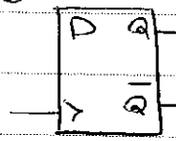
خروجی Dff حساس به لبه چگونه است؟

Subject: Year: Month: Date:

هنوز F.F حساس به سطح قبل استفاده در FSM نشدیم برای همین

برای f.f حساس به لبه می‌رویم.

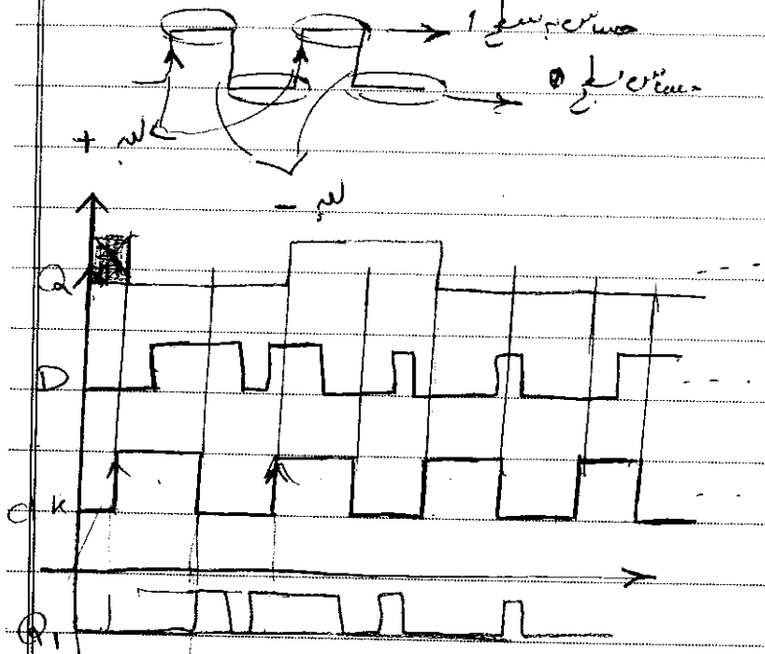
amultisweb.com



وقتی clock است هر چی در D است در خروجی می‌بینیم وقتی 0 است مقدار قبلی حفظ می‌شه

(D.FF حساس به لبه مثبت (بالارونده).

وقتی clock=0، clock=1 مقدار را حفظ می‌کند تنها وقتی که لبه داریم مقدار D (ورودی) به خروجی (Q) انتقال می‌یابد تا لبه بعد.

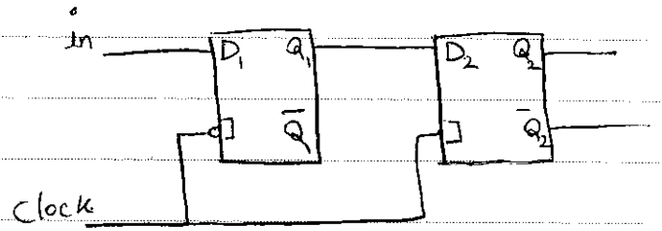


ساخت FF حساس به لبه چگونه است؟

Subject: Year: Month: Date:

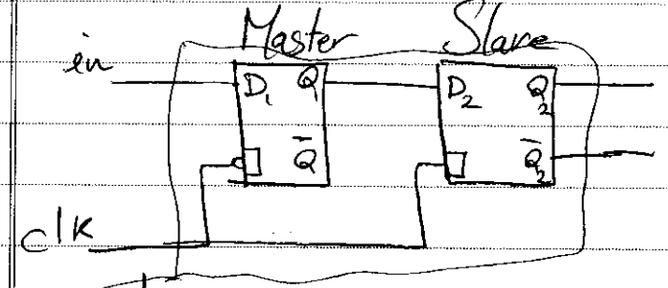
f_f حساس به لبه تغییرات آسان هاهنت است با تغییر clock و مراحل

می‌توانیم در مدل FSM کار کنیم.



آخرین ا

Q_1 از D می‌گیره، Q_2 از Q_1 می‌گیره (با توجه به این سیگنال‌ها



فرض می‌کنیم.

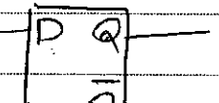
یک روش ساخت FF حساس به لبه این بود که دو ff حساس به سطح

و یک گیت Gasket

کنیم.

حالت Master_Slave پیش می‌آید در نتیجه این کار یک ff

حساس به لبه می‌کند.



Subject: Year: Month: Date:

Master کلمه

نکته: اگر Master حساس به سطح ۰ باشد ← D.Ff حساس به لبه ↓ می‌شود.

نکته: اگر Master حساس به سطح ۱ باشد ← D.Ff حساس به لبه ↑ می‌شود.

★ داخل دل Ff ← latch است. (and با ~ ، nor با ~)

★ داخل دل latch ، back-to-back (nor با ~) است.

کی Race داخل Flip Flop ، latch است.

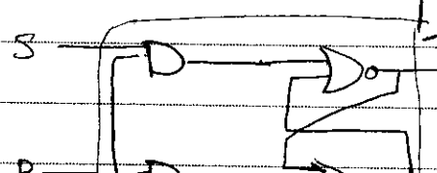
R,S روی Race هستند که یکی برنده، یکی بازنده است.

وچون خروجی وابسته است به برنده این یک Race کرانی است.

پس اگر RS=11 باشند هیچ clock بزنیم، latch داخل F.F

دچار Race کرانی می‌شوند و چون مقادیر نامعلوم می‌شوند پس

ما مقدار غیر مجاز RS=11 نمی‌دهیم.



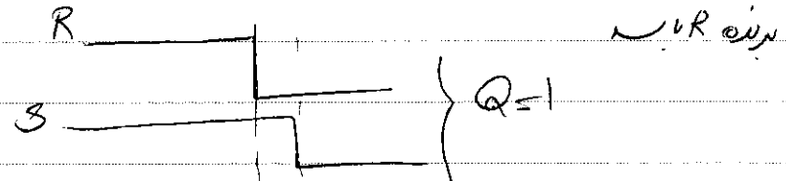
Subject: Year: Month: Date:

دچی معادل از Ff ، latch چیست Race کرانی پیش می‌آید؟

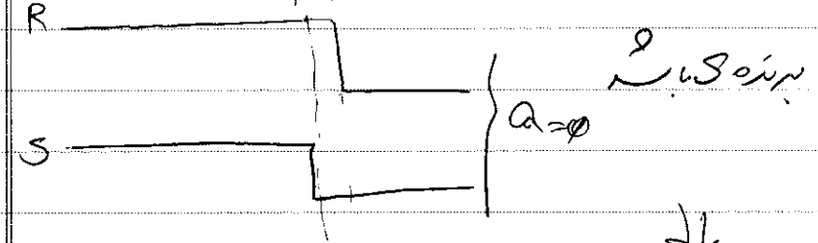
clock=1 باشد ، Race نداریم.

clock=0 باشد ، Race داریم و از نوع کرانی است زیرا:

برنده R باشد



برنده S باشد



Race کرانی داریم.

چون خروجی با برنده‌های مختلف تغییر می‌کند ← Race کرانی داریم.

متاسفانه در latch ، F.F مقدار ۱۱ نمی‌دهیم چون Race

کرانی پیش می‌آید.

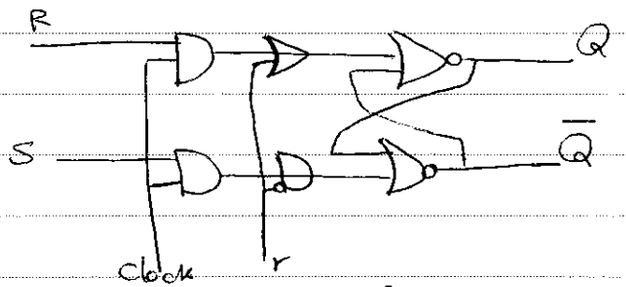
Subject: ..... Year: ..... Month: ..... Date: .....

ورودی R آسنکرون

اگر R=0 باشد FF کارهای خودرا انجام می دهد. اگر R=1 باشد

بدون توجه به clock, Rs, حتی داخل FF صفر می شود.

(یعنی Q=0 و $\overline{Q}=1$)



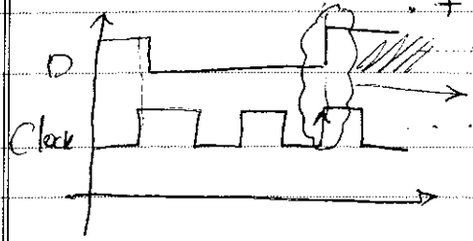
اگر r=0 باشد مدار قرائتی ندارد. اگر r=1 باشد مدار قرائت را انجام می دهد

سال	گزینه	فصل
۸۵	۶۷	F.F
۸۳	۶۷	H.
۸۰	۱۱	H.
۸۰	۱۴	F.F

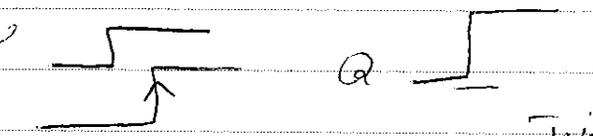
Subject: ..... Year: ..... Month: ..... Date: .....

برای جلوگیری از Race چکار باید کرد؟

Race ایجاد شده در F.F حساس به لبه +

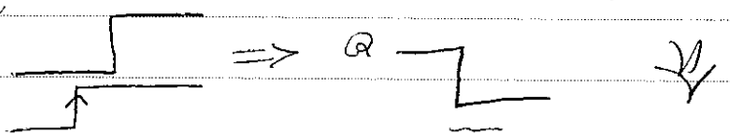


اگر D برنده لبه



پس دو ورودی متفاوت

اگر clock برنده لبه

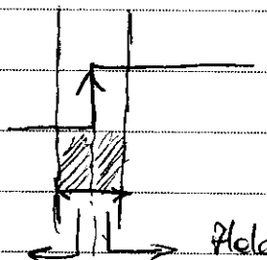


Race جبرانی

برای حل این مشکل نباید مقدار ورودی و لبه کلاک با هم تغییر کنند

باید از Setup time و Hold time کمک می گیریم.

یعنی ورودی FF باید Setup time و Hold time حق تغییر ندارند.



(ts) Setup ⟷ Hold time (th)

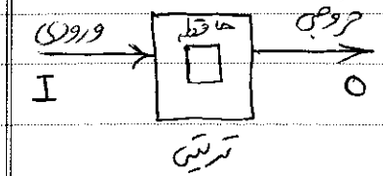
Subject: Year: Month: Date:

یک مدل FSM از چه بخش‌هایی تشکیل شده است؟

FSM = Finite State Machine مدل ریاضی FSM

(I, O, S, T, F_o)

مدل ریاضی FSM از پنج مجموعه است.



* I : ورودی‌ها
* O : خروجی‌ها
* محتویات داخل حافظه همیشه یک سمبل است از یک .
* T تابع T

$$\text{حفظ} \begin{cases} T: I \times S \to S \\ S[n] = T(i[n-1], S[n-1]) \end{cases}$$

(Mealy)

$S[n]$ محتویات حافظه در لحظه n تابعی است از ورودی $n-1$ لحظه قبل

و محتویات حافظه در لحظه $n-1$ و اسم است.

* F_o تابع F_o

$$\text{خروج} \begin{cases} O[n] = F_o(i[n], S[n]) \end{cases}$$

(Mealy)

خروجی در هر لحظه n تابع ورودی در همان لحظه و حالت همان لحظه

حافظه است.

Subject: Year: Month: Date:

$O[n]$ و $S[n]$ چیست؟

مثال:

$I = \{x, y\}$

حالت اولیه:

$O = \{0, 1, 2, 3\}$

$S[1] = B$

$S = \{A, B, C\}$

(پرسش سمت FSM)

$T = \{(x, A, A), (y, A, C), (x, B, B), (y, B, A), (x, C, B), (y, C, A)\}$

(حالت فعلی، ورودی، حالت بعدی)

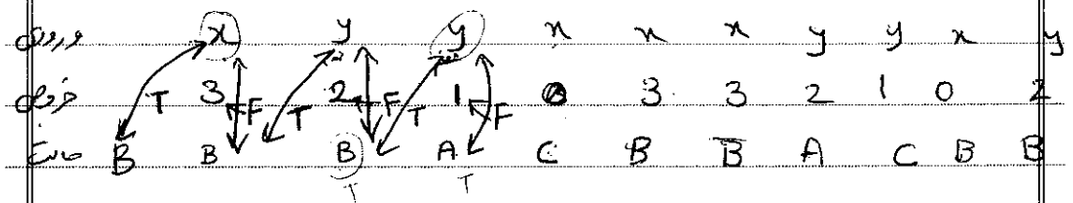
$F = \{(x, A, 2), (y, A, 1), (x, B, 3), (y, B, 2), (x, 2, 0), (y, C, 3)\}$

(حالت فعلی، ورودی، خروجی)

هرگاه یکی این 5 تا را در آنگاه FSM معین داده است.

(زمان در FSM گسسته است)

n=	1	2	3	4	5	6	7	8	9	10
ورودی	x	y	y	x	x	x	y	y	x	y
خروجی	3	2	1	0	3	3	2	1	0	2
حالت	B	B	B	A	C	B	B	A	C	B	B



Subject: Year: Month: Date:

جدول حالت ۲ ورودی = ورودی خروجی = R

سمت چپ ورودیهای فعلی و حالت فعلی، سمت راست خروجی فعلی و حالت بعدی

(جدول حالت)

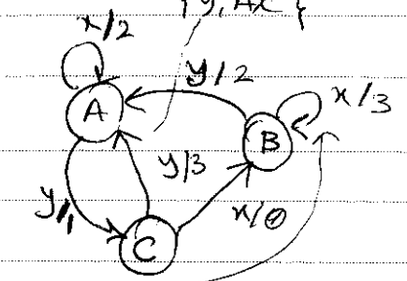
ورودی فعلی	حالت فعلی	خروجی فعلی	حالت بعدی
x	A	2	A
y	A	1	C
x	B	3	B
y	B	2	A
x	C	0	B
y	C	3	A

ورودی 2^2 × حالت 3 = 6 سطر مختلف داریم

Subject: Year: Month: Date:

(حالت فعلی A، ورودی y، حالت بعدی C)

حالت گذار نمایش FSM



F = {(x, B, 3) ...

F و 0 هم بازخوانی

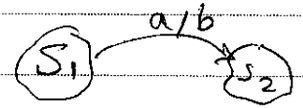
← این گراف یعنی این پنج مجموعه را داریم

یعنی FSM داریم. به این گراف گوییم

(نمودار حالت State diagram)

FSM ← Formal = مجموعه‌ای رسمی

State diagram = دیاگرامی گرافی



آمدن ورودی a در حالت S_1 و آمدن خروجی b

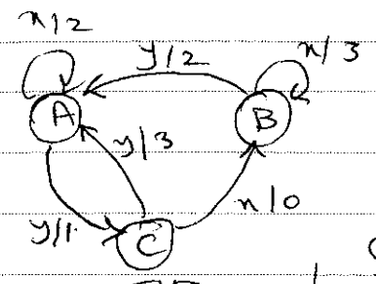
و رفتن به حالت S_2 خواهیم بود.

روش سوم State Table ،

Subject: Year: Month: Date:

ترسیم مدار منطقی یک FSM

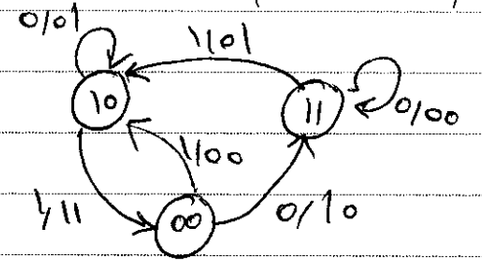
۱- تخصیص کد باینری به حالت‌ها



حالت	کد
A	10
B	11
C	00

ورودی	
x	0
y	1

خروجی	
Φ	10
1	11
2	01
3	00



۲- ساخت مدار

حالا وقتی FSM را داریم برای ساخت باید

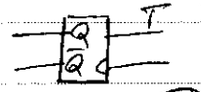
۱- حالت‌ها را مشخص کنیم (حالت چیست؟ مخزن‌هایی حافظه است)

۲- از فلیپ فلاپ استفاده کنیم زیرا می‌خواهیم حالت‌ها را ذخیره کنیم.

Subject: Year: Month: Date:

چون در این حالت ۲ بیت هستند نیاز به ۲ فلیپ فلاپ است.

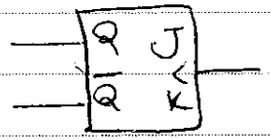
طرز کار فلیپ فلاپ T



$T=0$ ← محتویات خود را حفظ می‌کند

$T=1$ ← هر چه تولید شده نقض می‌کند

طرز کار فلیپ فلاپ Set and Reset / فلیپ فلاپ JK (چون شبیه RS)



$JK=00$ ← حفظ حالت قبلی

J K
Φ 1 ← Reset می‌شود

J K
1 0 ← Set می‌شود

J K
1 1 ← مخزن خودش را نقض می‌کند

Subject: Year: Month: Date:

حالت ورودی تابع

$O[n] = F_o(i[n], S[n])$ خروجی لحظه n

همان لحظه حالت لحظه

۲ـ ساعت F_o

خروجی لحظه n تابعی از ورودی n و حالت n است که با مدار ترکیبی F_o می‌توان ساخت.

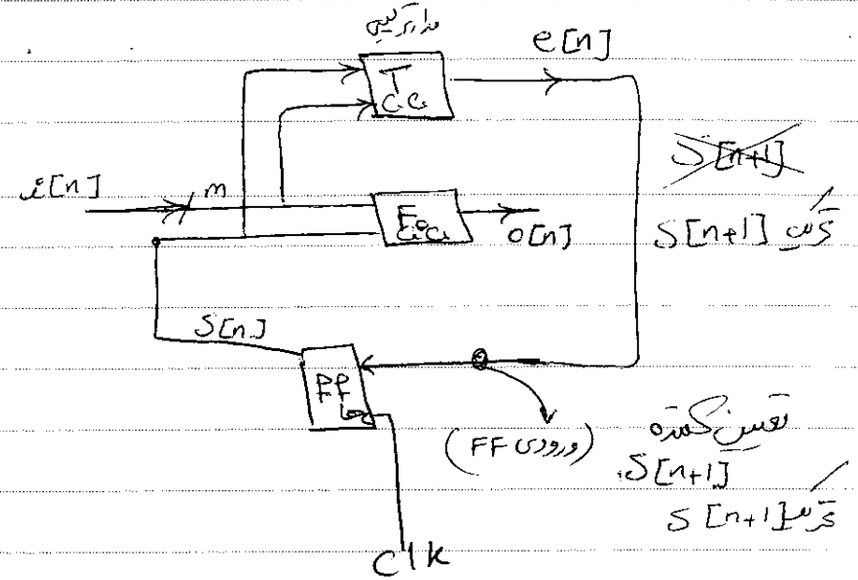
خروجی FlipFlop ها حالت لحظه (n) است و اما بدانید که ورودی فلیپ فلاپ تعیین کننده لحظه n+1 است. که به این دلیل که توسط ورودی F.F است از آنجایی می‌گوییم تحریک لحظه n+1 فلیپ فلاپ.

تحریک لحظه n همان تحریک در لحظه n+1 است.

۳ـ ساعت τ

$e[n] = S[n+1]$

$e(n)$ تحریک لحظه n است که با F.f می‌دهیم که از F.f حالت لحظه n+1 را بگیریم.



۴ـ ساعت مدار ترکیبی برای ساخت مدار ترکیبی نیز به جدول حقیقت نیاز است.

در جدول حقیقت F_o

(راست ، چپ)

(خروجی ، ورودی)

در جدول حالت

(راست ، چپ)

(e(n), S(n) , O(n))

پس از رسم جدول حالت

جدول حقیقت را رسم می‌کنیم.

Subject: Year: Month: Date:

R,S بیت سمت چپ $Q_1, \bar{Q}_1$

است.

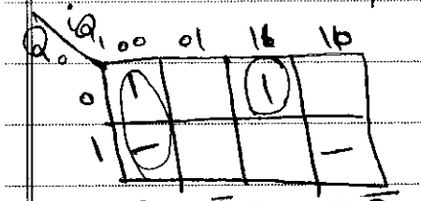
T بیت سمت راست $Q_0, \bar{Q}_0$

شو

ورودی

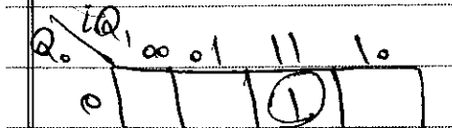
مدل حقیقت T

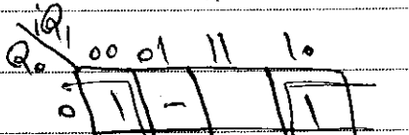
i[n]	S[n] $Q_1\ Q_0$	o[n]	S[n+1] $Q_1\ Q_0$	R S T
0	0 0	1 0	1 1	0 1 1
0	0 1	— —	— —	— — —
0	1 0	0 1	1 0	0 — 0
0	1 1	0 0	1 1	0 — 0
1	0 0	0 0	1 0	0 1 0
1	0 1	— —	— —	— — —
1	1 0	1 1	0 0	1 0 0
1	1 1	0 1	1 0	0 — 1



$Q_1 = i\bar{Q}_1 + iQ_1\bar{Q}_0$

$Q_0 = Q_1\bar{Q}_0 + iQ_1$





Subject: Year: Month: Date:

نمودن جدول حالت

i[n]	S[n]	o[n]	S[n+1]
0	0 0	1 0	1 1
0	0 1	— —	— —
0	1 0	0 1	1 0
0	1 1	0 0	1 1
1	0 0	0 0	1 0
1	0 1	— —	— —
1	1 0	1 1	0 0
1	1 1	0 1	1 0

ورودی‌های حالت — خروجی حالت

مدل حقیقت Fo

از روی دیاگرام حالت

سمت F0

سمت T چون خروجی T برابر $S(n+1)$ است یک کنترل $S(n+1)$ است که مشخص می‌شود به کدام F.F وضعیت $S(n+1)$ داده می‌شود.

روی قسمت مثلث‌ها دارید کنیم مادراین مثال ۲ قسمت

مثلاً طریقی RS و یک T.

۳ ویژگی FSM چیست؟

Subject: Year: Month: Date:

مدارهای ساخته شده توسط اشخاص مختلف متفاوت است فقط مدارشان

؟ آن همه از کدام استفاده می‌کنند همه مدارات ۲ طبقه است و .

چون تخصیص یک توسط افراد و اتاق ـ طبیعی فلیپ‌فلاپ‌ها متفاوت است

سری مدار متفاوت است .

★ FSM یک مدل ریاضی است برای مدل کردن حالت‌هایی که حافظه دارند.

زمان آن گسسته و مقادیرشان گسسته است .

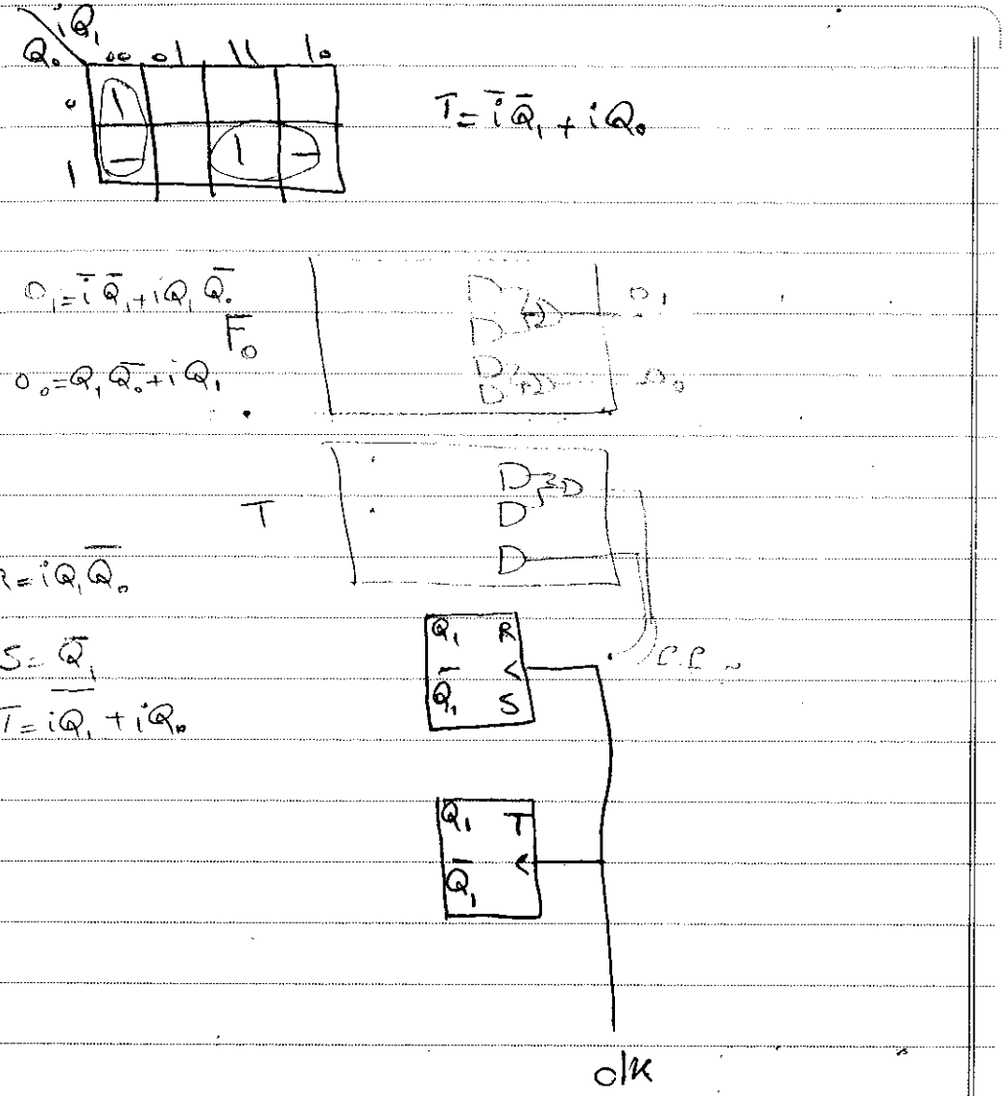
★ FSM لزوما به معنی مدار منطقی نیست امکان دیگر ساختش این

که با مدار منطقی بسازیم .

چه‌کار کنیم در F.F پیش نیاید حتما پیش بیاید Race . از قانون

Setup Time, Hold up استفاده کنیم. حال چه کار کنیم

که این مدارها این قانون را برای ورودی‌های F.F مان



نوشتیم FSM ← مدار تولید کردیم

این کار چگونه انجام شد؟

Subject: Year: Month: Date:

dff چیست؟ / dmin و dmax چیست؟ / رابطه‌ی بین ts و th چیست؟

سینال clock سنکرون‌ها

آن سینال تغییر نمی‌کند مگر اینکه clock بخورد. (امیدوارند که این سینال تغییراتش کمی عقب‌تر از clock است (اول clk میکند تا ضربه‌ی سینال).

باید ورودی ff با clk حداقل سنکرون باشد تا Hold Time و Setup time طبق فلیپ‌ها رعایت شود.

R,S,T تابع از نباشند چون نمی‌کند T, S, R

هم تغییر می‌کند پس باید هم با clk سنکرون باشد که T و S و R

در هر موقع خواستند عوض شوند.

تأخیر فلیپ = فاصله زمانی بین خوردن clk و تغییر خروجی d_{FF}

dmin و dmax

T یک مدار ترکیبی است که وارد می‌شود عواملی است که طولانی‌ترین

دیگر تاهترین مسیر هم دارد. که تاخیر در کوتاهترین مسیر در مدار ترکیبی را

d_{min} گوئیم.

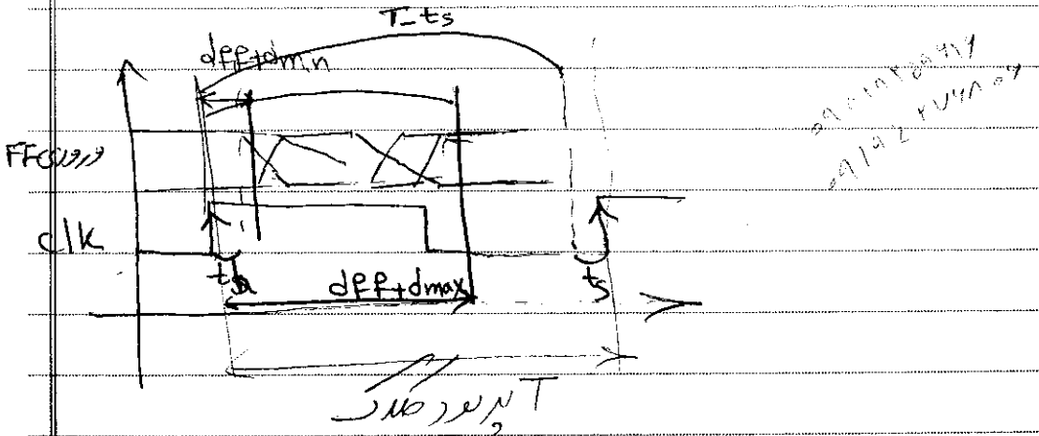
از وقتی clk می‌خورد کمترین زمانی که ff شروع به تاخیر می‌کند برابر

تاخیر فلیپ فلاپ + کوتاهترین مسیری که طی کند d_{min} یعنی $d_{FF}+d_{min}$

از وقتی clk می‌خورد بیشترین زمانی که ff شروع به تاخیر می‌کند برابر تاخیر

فلیپ فلاپ + تاخیر بزرگترین مسیری که طی کند d_{max} یعنی $d_{FF}+d_{max}$

پس تغییرات فقط بین فاصله زمانی $d_{FF}+d_{max}$ و $d_{FF}+d_{min}$ است.



$d_{FF}+d_{min} > t_h$