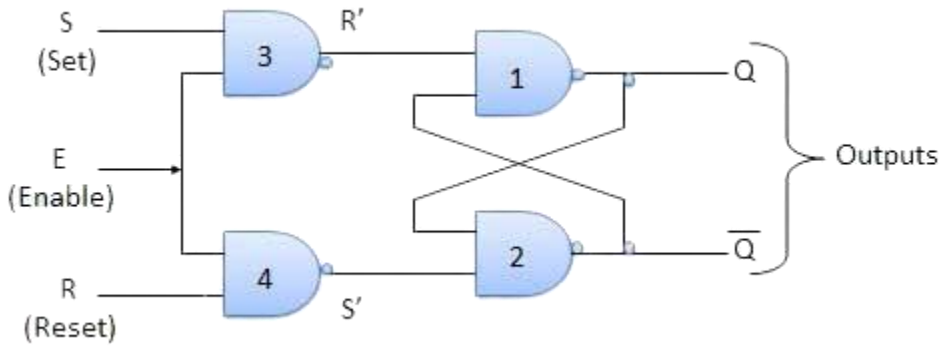
بسم الله الرحمن الرحيم

رحلتي من تغميس الشاي الى البروسيسور

Sequential Logic Design



احنا شرحنا عن الـ combinational logic الاخراج الخاص فيو بعتمد على الـ inputs الحالية.

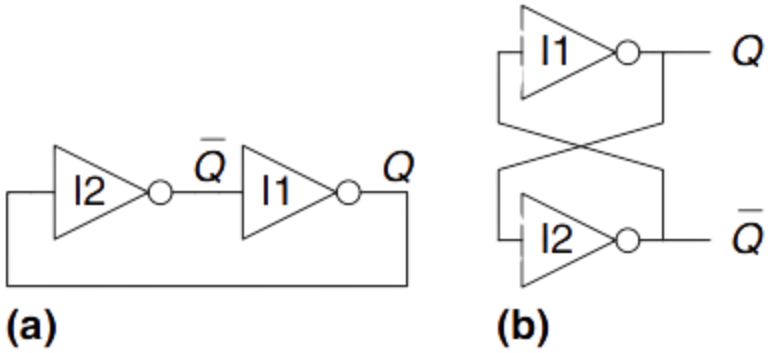
في هذا الفصل راح نشرح تحليل وتصميم الـ sequential logic الـ outputs الخاص فيها بتعتمد على الـ input الحالية والسابقة وكمان عندو memory .

عندو ذاكرة عشان للحفاظ على المعلومات السابقة .

state of a digital sequential circuit : هي مجموعة من البتات تسمى state variables والتي تحتوي على كافة المعلومات عن الماضي او السابق المطلوبه عشان تفسر سلوك المستقبلي لدائرة الحالة هاي بتشمل معلومات عن الحالة السابقة للدائرة والتي تؤثر على مستقبل الدائرة .

راح نبدأ بشرح الـ flip flop وهي عبارة عن دوائر بسيطة تخزن بتا واحده من الـ state .

LATCHES AND FLIP-FLOPS



الاساس عشان تبني الـ memory هو الـ bistable element وهو عنصر ذو حالتين مستقرتين.

الـ (a) بوضح لك bistable element بسيط بتكون من 2 inverters متصلة في الـ loop .

الـ (b) بوضح لك نفس الصورة الي موجودة في a لكن تم اعادة رسمها للتاكيد العاكسات مترابطة بمعنى Input I1 هو الـ output الخاص في I2 والعكس صحيح.

الدائرة هاي ما بتحتوي على مدخلات لكن بتحتوي على مخرجين Q & Q` .

تحليل هذه الدائرة بختلف عن تحليل الدائرة combinational لانها هاي دائرية الـ Q بعتمد على الـ Q` والـ Q` بعتمد على Q .

نفترض حالتين الـ Q = 0 او الـ Q = 1 فا في الحالة الاولى بكون عنا :

- **CASE 1 :**

اذا الـ Q = 0

لو تلاحظ عندك في هاي الصورة في (a) الـ I2 مبدئيا بكون FALSE 0 لكن بنتج اخراج TRUE 1 وفي الـ I1 بكون الادخال TRUE 1 والاخراج بكون FALSE 0 .

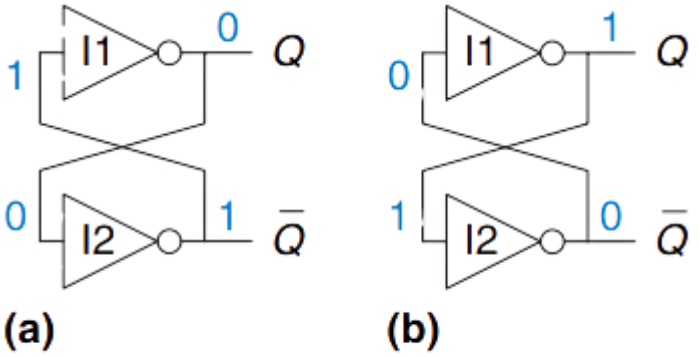
وهذا الافتراض بتفق مع الافتراض الاصلي Q = 0 فا بهاي الحالة بكون بتكون Stable مستقرة.

- **CASE 2 :**

اذا الـ Q = 1

لو تلاحظ عندك في هاي الصورة في (b) مدخل I2 يكون 1 TRUE والاخراج ينتج عنه 0 FALSE و I1 الادخال 0 FALSE والاخراج ينتج عنه 1 TRUE .

وهذا يعني انه Stable مستقر.

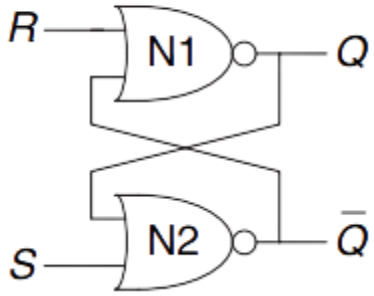


العاكسات عندها بس حالتين مستقرتان الـ 0 و الـ 1 وتمسى bistable بس في نقطة لازم تعرفها هو ان الدائرة عندها حالة ثالثه بين الـ 0 و الـ 1 بتكون محتمله وبكون اسمها metastable وراح نناقشها بعدين.

العنصر مع N Stable State و log2N of information لذالك يخزن الـ bistable element بتا واحد.
والعاكسات في Q binary state variable قيمة الـ Q بتخبرنا بكلشئ عن الماضي وهو ضروري لشرح سلوك المستقبلي لدائرة.
اذا كانت Q = 0 راح تضلها للابد 0 واذا كانت 1 راح تضلها 1 للابد.

SR Latch

وحدة من ابسط الدوائر الـ sequential هي الـ SR Latch بتكون من بوابتين cross-coupled NOR gates .



عندو مدخلين S & R ومخرجان Q & Q` هو بشبه الـ cross-coupled inverters الي شرحناه فوق ولكن الفرق هو بنقدر نتحكم في حالتو من خلال الـ R & S فوق ما كنا بنقدر نتحكم في حالتو ما عنا مدخلات مدخلاتو لحالها وبتتبدل لحالها هون بنقدر نتحكم في الـ Q بشكل عادي.

احدى الطرق الجيده لفهم مثل هاي الدوائر والدوائر الغير مالوفه هو استخدام الـ Truth Table والـ nor gate شرحناها تنتج 1 عندما تكون 00 وتنتج صفر للكل .

الحالات :

- **الحالة الاولى :**

R = 1 , S = 0

الـ N1 راح يكون FALSE 0 والـ N2 راح تكون TRUE .

- **الحالة الثانية :**

R = 0 , S = 1

في N1 بكون الادخال 0 وراح يكون الاخراج 1 يعني TRUE اما عند N2 الادخال راح يكون 1 يعني TRUE والاخراج راح يكون العكس بسبب العاكس يعني راح ينتج 0 FALSE .

- **الحالة الثالثة :**

تعتبر حاله غير مستقره

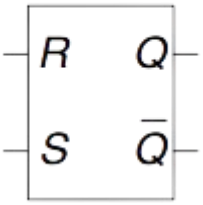
وتستمر الحالات هكذا الى النهايه

Case	S	R	Q	$\overline{Q}$
IV	0	0	Q_{prev}	$\overline{Q}_{prev}$
I	0	1	0	1
II	1	0	1	0
III	1	1	0	0

متغيرات الـ S & R تسمى Set & Reset

الـ reset bit يدل على جعله FALSE .

غالبا ما تكون الـ Q & Q` تكون complementary .



الصورة هاي بتمثل كـ symbol

استخدام الرموز عشان نسوي تجريد ونشيل التفاصيل.

في طرق مختلفه عشان نسوي SR Latch منها استخدام بوابات منطقيه او ترانزيستورات.

مثل الـ cross-coupled inverters يعتبر الـ bistable element SR Latch

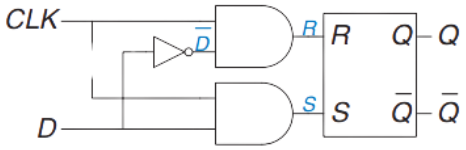
عند تأكيد R، تتم إعادة تعيين الحالة إلى 0.

عندما يتم التأكيد على S، يتم تعيين الحالة على 1.

وعندما لا يتم التاكيد اي من القيم تحتفظ في حالة القيمة القديمه.

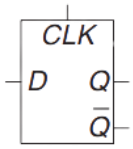
D Latch

الـ SR Latch يعتبر غريب خاصه لما الـ S & R يصيرو 1 في نفس الوقت.



CLK	D	$\overline{D}$	S	R	Q	$\overline{Q}$
0	X	$\overline{X}$	0	0	Q_{prev}	$\overline{Q}_{prev}$
1	0	1	0	1	0	1
1	1	0	1	0	1	0

(b)



Q = 0 Reset

Q = 1 Set

هاي صورة على بتفرجيك الـ D Latch عندها 2 inputs الـ D هو الـ Data Input هو الي بتحكم شو الـ State راح تكون.

الـ CLK هو الـ Clock Input بتحكم في الوقت الذي يجب ان تتغير به الـ state .

وهاي الصورة فيها الـ Truth Table الخاص في الـ D Latch .

مبدئيا العقد الداخلية : D , R & S .

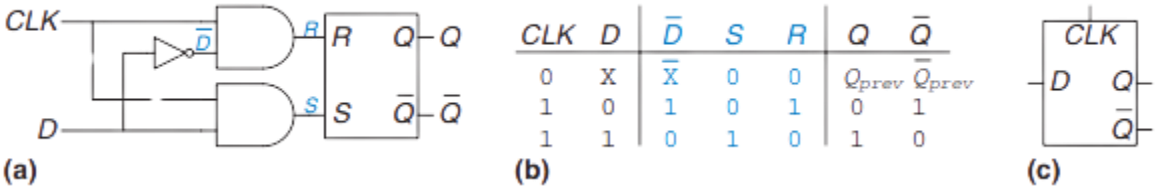
اذا كانت الـ CLK = 0 الـ S & R بكونو FALSE بغض النظر عن قيمة D .

اذا كانت الـ CLK = 1 بوابة الـ AND راح تنتج TRUE والاخرى FALSE اعتمادا على قيمة الـ D .

لما الـ CLK = 0 لو تلاحظ الـ Q & Q` بتساوي قيمتها القديمه .

على كل حال منطقيا في جميع الاحوال الـ Q هو الـ complement او المكمل لـ Q` والعكس صحيح.

D Flip-Flop & Latches



الصورة بتوضح اذا كانت الـ CLK = 0 الموضوع راح يكون كلو FALSE و ممكن تجرب لحالك وبغض النظر عن قيمة D .

اذا كانت الـ CLK = 1 بوابة AND راح تنتج وحدة TRUE والاخرى FALSE اعتمادا على قيمة D .

في جميع الحالات Q بتكون مكمل لـ `Q .

التفصيل

في (b) لو تلاحظ عندك الـ Truth Table في عندك عقد داخلية D , S & R اذا كانت الـ CLK = 0 الـ R & S راح تكون FALSE اما اذا كانت CLK = 1 الـ R & S وحده منهم راح تكون TRUE والاخرى FALSE اعتماد على قيمة الـ D وعندما تكون CLK = 0 الـ Q بتذكر قيمتو القديمة Qprev واعتماد على قيمة .

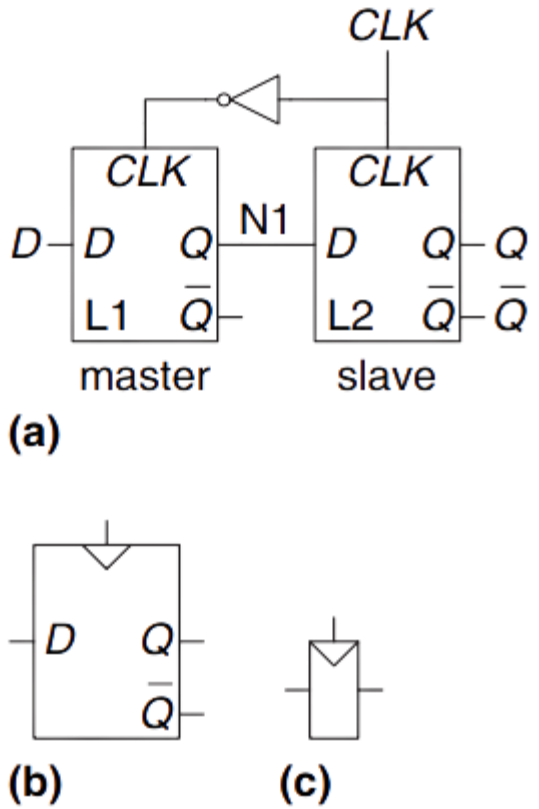
وترى ان الـ CLK تتحكم في وقت التدفق البيانات من خلال الـ Latch .

عندما يكون CLK = 1، يكون transparent بمعنى يسمح بتمرير البيانات .

عندما يكون CLK = 0 يكون opaque بمعنى لا يسمح بتمرير البيانات تكون معلقة ويحتفظ بالقيمة القديمة.

الـ symbol او رمز المزلاج هذا يظهر في صورة (c).

الـ D flip-flop



الـ D فليب فلوب يمكن بنائه من خلال two back-to-back D latches ويتم التحكم فيهما بواسطة الـ CLK complementary clocks مثل ما هو موضح في صورة (a) .

الـ Latch الاول L1 يسمى master .

الـ Latch الثاني L2 يسمى slave .

والـ nodes الي بينهم N1.

والـ Symbol الرمز لـ D Flip Flop موجود في صوره (b).

عندما لا يكون بحاجة لمخرج Q` غالبا بتكون مثل صورة (c). هل ممكن ان لا يكون بحاجة الى مخرج Q ؟ نعم من الممكن احد هاذي الاسباب لتشفير لحفظ خصوصية الـ output او لتبسيط او اصلا اذا لم تكن مهتم في العاكس للـ output .

لما تكون الـ CLK = 0 يكون الـ Master Latch راح يصير transparent والـ slave Latch راح يصير opaque .

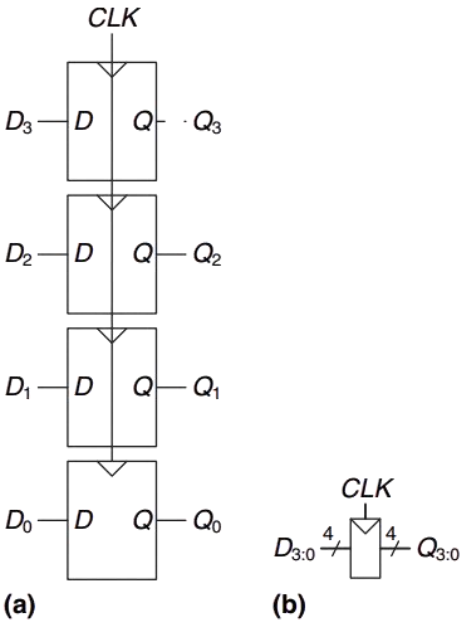
بمعنى اي قيمة عند الـ D راح تنتشر من خلال الـ N1 .

اما عندما تكون الـ CLK = 1 الموضوع بصير العكس الـ Master بصير opaque والـ slave بصير transparent والقيمة تنتشر في N1 الى الـ Q وعند الـ D يتم قطعها .

عندما تتغير الـ CLK من 0 الى 1 في هذه اللحظة يتم نسخ القيمة الموجودة في D الى Q .

Register

الـ N-bit Register عبارة عن N Flip-Flop بتشترك في مدخل الـ CLK بحيث كل البتات يتم تحديثها بنفس الوقت .



الصورة هاي بتوضح الرسم التخطيطي والرمز (symbol) لـ 4-bit register مع ادخالات inputs الي هي من D0:D3 ومخرجات من Q0:Q3 .

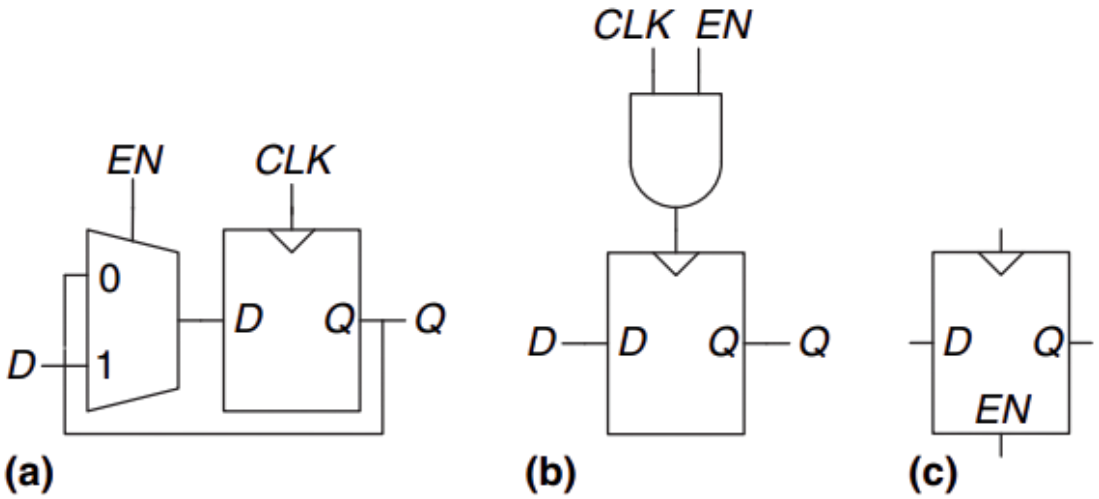
Enabled Flip-Flop

هذا النوع من الـ flip flop يضيف ادخال اخر يسمى EN OR ENABLE لتحديد ما اذا كان سيتم تحميل البيانات على الـ Clock Edge .

عندما تكون EN = TRUE يتصرف مثل ordinary D flip-flop .

عندما تكون EN = FALSE الـ Flip Flop يتجاهل Clock ويقوم بحفظ حالته .

الـ Enabled flip-flops بتكون مفيدة عندما نرغب في تحميل قيمة جديدة في الـ flip flop في وقت معين وليس على كل clock edge .

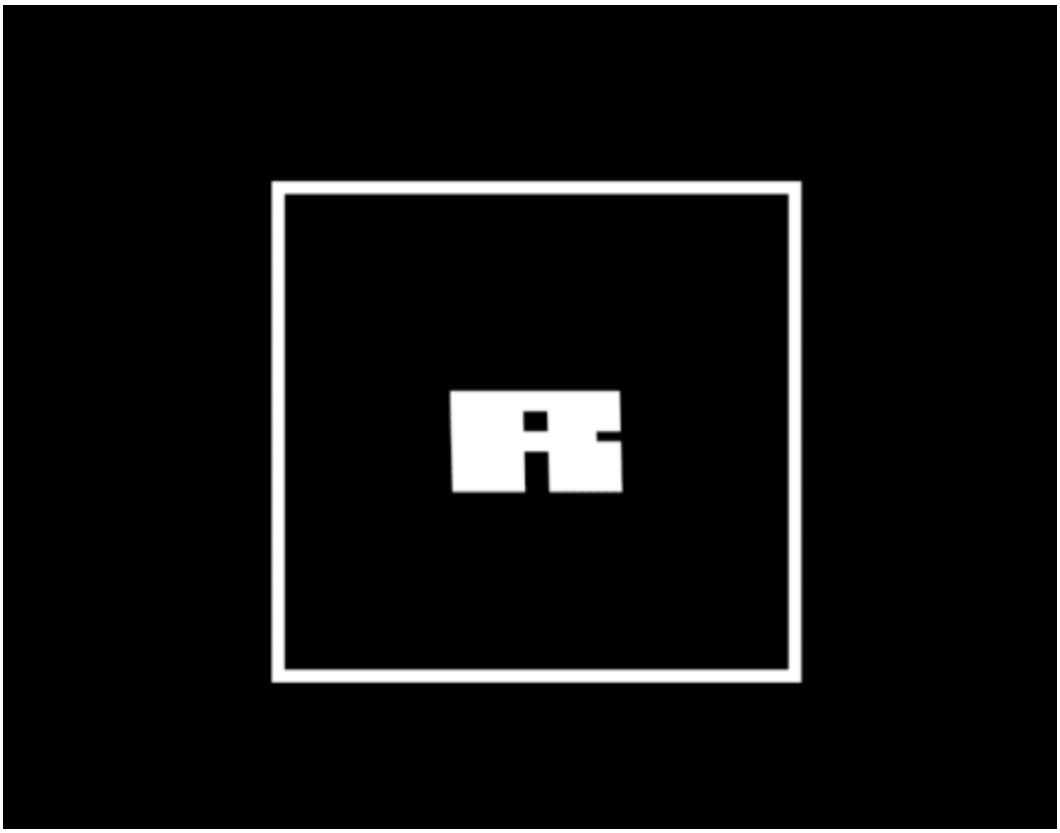


الصورة هاي بتوضح طريقتين لانشاء enabled flip-flop من خلال الـ D Flip Flop و بوابة اضافية.

في (a) يختار الـ MUX القيمة الي ستمر عند الـ D Input اذا كانت الـ EN = TRUE ستعمل بشكل طبيعي.

اما

EN = FALSE والـ CLK = FALSE سيحتفظ في قيمته القديمة.



AhmadAlFareed

Twitter : https://twitter.com/dr_retkit

YouTube : https://www.youtube.com/@retkit1823