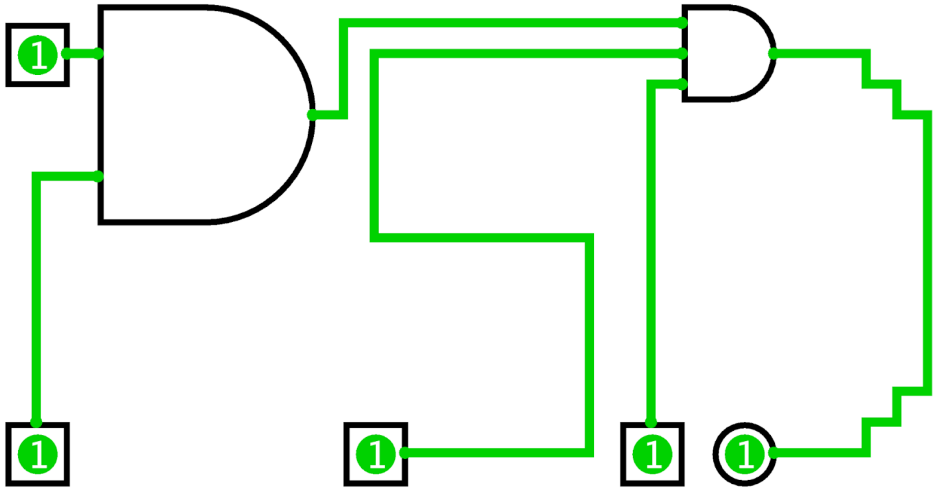
#PSDisFun❤️



LAB 8

Pengantar Sistem Digital

2023-2024 Gasal

GIK

#PSDisFun❤️

Petunjuk Pengerjaan

- Kerjakan semua soal sesuai dengan spesifikasi tiap soal.
- Mohon baca setiap spesifikasi soal dengan saksama sebelum bertanya kepada asisten dosen.
- Jika ada soal yang membingungkan atau kesalahan pada soal, silakan bertanya kepada salah satu asisten dosen yang sudah *stand-by.*
- Lakukan submisi semua file (sesuai spesifikasi yang ada pada soal) di SCELE sebelum **Kamis, 30 November 2023 pukul 14:50** **(Toleransi keterlambatan pengumpulan yaitu 5 menit)**.
- **Penalti sebesar 2 poin** akan dikenakan untuk keterlambatan **setiap menit**. Contoh: telat **15 menit**, maka dikenakan **penalti sebesar 20 poin** karena 2 * (15 - 5) = 20 dimana 5 menit pertama adalah waktu toleransi keterlambatan. **Jika terlambat selama > 55 jam**, lab **tidak akan dinilai**.

Submisi

Kumpulkan satu *file* jawaban dengan format penamaan *file*:

`LAB8_[Kode Asdos]_[NPM]_[Nama].zip`

Sirkuit Logisim diberi nama dengan format penamaan file:

`LAB8_[Kode Asdos]_[NPM]_[Nama].circ`

Jawaban penjelasan diberi nama dengan format penamaan file:

`LAB8_[Kode Asdos]_[NPM]_[Nama].pdf`

Catatan: Tanda '[' dan ']' tidak perlu ditulis!

Contoh:

- LAB8_ABC_2306123456_PakEsde.zip
- LAB8_ABC_2306123456_PakEsde.circ
- LAB8_ABC_2306123456_PakEsde.pdf

#PSDisFun❤️

Merancang Modified Parallel Load Counter

(16 Poin)

Rilis - 29/11/2023

Pak Esde sangat senang dengan kinerja kalian di semester ini dan ingin memberikan tantangan terakhir untuk menguji kemampuanmu.

Ia ingin anda membuat ***modified 4 bit parallel load counter*** dengan ***4 bit binary load input X*** dan *control input* berikut: **Load, Count, Inc2, Inc4, dan Inc8**. Present State dari counter tersebut akan direpresentasikan oleh **4 D flip-flop** dengan label A-D, dengan **A sebagai MSB** dan **D sebagai LSB**. Counter tersebut dapat melakukan operasi-operasi berikut berdasarkan control bitsnya:

Load	Count	Inc8	Inc4	Inc2	Outcome (on next clock cycle)
1	X	X	X	X	Load 4 bit binary load input (tanpa inkrementasi)
0	1	1	X	X	Counter di-*increment 8*
0	1	0	1	X	Counter di-*increment 4*
0	1	0	0	1	Counter di-*increment 2*
0	1	0	0	0	Counter di-*increment 1*
0	0	X	X	X	Do nothing

Note: seperti banyak counter lainnya, jika overflow (>=16), counter akan loop balik, jadi apabila 8 di-*increment* 8 maka hasilnya adalah 16-16=0, 15 di-*increment* 4 hasilnya 19-16=3, 13 di-*increment* 8 akan menghasilkan 21-16 =5, dst.

Contoh:

Load 4 bit binary load input (tanpa increment)

Jika Load = 1, Count = 1, Inc8 = 1, Inc4 = 0, Inc2 = 1 dan I = 1010 maka:

1010 akan di-*load* ke Counter (pada clock cycle selanjutnya) .

Note: jika Load = 1, maka control bit lain tidak berpengaruh

Counter di-*increment* 8

Jika Load = 0, Count = 1, Inc8 = 1, Inc4 = 1, Inc2 = 0 dan current state = 0110 counter akan berjalan dari:

0110 ⇒ 1110 ⇒ 0110 ⇒ 1110 ⇒ 0110 ⇒ **Ulang**

Note: jika Load = 0, Count = 1 dan Inc8 = 1, maka control input Inc4 and Inc2 tidak berpengaruh

Counter di-*increment* 4

Jika Load = 0, Count = 1, Inc8 = 0, Inc4 = 1, Inc2 = 1 dan current state = 0011 counter akan berjalan dari:

0011 ⇒ 0111 ⇒ 1011 ⇒ 1111 ⇒ 0011 ⇒ **Ulang**

Note: jika Load = 0, Count = 1, Inc8 = 0 dan Inc4 = 1, maka control input Inc2 tidak berpengaruh

Counter di-*increment* 2

Jika Load = 0, Count = 1, Inc8 = 0, Inc4 = 0, Inc2 = 1 dan current state = 0000 maka counter akan berjalan dari:

0000 ⇒ 0010 ⇒ 0100 ⇒ 0110 ⇒ 1000 ⇒ 1010 ⇒
1100 ⇒ 1110 ⇒ 0000 ⇒ **Ulang**

Counter di-*increment* 1

Jika Load = 0, Count = 1, Inc8 = 0, Inc4 = 0, Inc2 = 0 dan current state = 1011, counter akan berjalan dari:

1011 ⇒ 1100 ⇒ 1101 ⇒ 1110 ⇒ 1111 ⇒ 0000 ⇒ 0001 ⇒ 0010 ⇒ 0011 ⇒ 0100 ⇒
0101 ⇒ 0110 ⇒ 0111 ⇒ 1000 ⇒ 1001 ⇒ 1010 ⇒ 1011 ⇒ **Ulang**

Do Nothing

Jika Load = 0, Count = 0, Inc8 = 1, Inc4 = 1, Inc2 = 1 dan current state = 1111 counter akan berjalan dari:

1111 ⇒ 1111 ⇒ 1111 ⇒ **Ulang**

Note: jika Load = 0 and Count = 0, maka counter tidak akan berubah. Control input Inc8, Inc4 and Inc2 tidak akan berpengaruh

Contoh yang lebih detil dapat ditemukan di bagian 'Test cases to test Modified4BitCounter' di akhir dokumen soal lab

#PSDisFun❤️

Implementasi:

Anda akan membuat sirkuit dengan memodifikasi sirkuit ***Counter with Parallel Load*** yang dapat dilihat di **Chap 6: Register and Counter halaman 46**

Sebelum memulai, Pak Esde harus memastikan bahwa anda sudah mengerti apa yang anda harus lakukan untuk mengimplementasi spesifikasi di atas.

Ia memberitahumu apabila anda tidak melakukan load, maka bit C (second least significant bit) akan selalu dibalik jika kondisi-kondisi berikut bernilai True/=1:

- (Load = 0 DAN Count = 1 DAN Inc2 = 1 DAN Inc4 = 0 DAN Inc8 = 0) **ATAU**
- (bit D (LSB) akan dibalik dari 1 menjadi 0).

Hal ini karena kita membalik nilai bit C apabila counter diincrement 2 ATAU bit D dibalik dari 1 menjadi 0.

Tolong jawab pertanyaan berikut di PDF file.

1. Apa kondisi yang harus bernilai True untuk bit D (LSB) untuk dibalik (apabila diasumsikan bahwa anda tidak melakukan load)?
2. Apa kondisi yang harus bernilai True apabila bit B ingin dibalik (asumsi bahwa anda tidak melakukan load)?
3. Apa kondisi yang harus dipenuhi (bernilai True) apabila kita ingin membalik nilai bit A (asumsi bahwa anda tidak melakukan load)?
4. Apa modifikasi yang harus dibuat pada sirkuit ***Counter with Parallel Load*** untuk mengimplementasikan kondisi-kondisi/setting baru di atas?

Hint1: Remember that A is the MSB and D is the LSB. Also, be reminded that some operations have some of the control inputs as don't care. Ingat bahwa A adalah MSB dan D adalah LSB. Perlu diingat juga bahwa beberapa operasi di tabel di atas mempunyai nilai control input don't care

Hint2: Tentukan bagian mana dari sirkuit ***Counter with Parallel Load*** yang digunakan untuk membalik bit bit dari counter secara kondisional saat tidak melakukan load. Bagaimana caranya memodifikasi bagian tersebut untuk mengubah kondisi yang harus dipenuhi untuk bit tertentu untuk dibalik?

Hint3: Pada versi yang tidak dimodifikasi dari sirkuit ***Counter with Parallel Load,*** satu-satunya kondisi untuk membalik nilai bit C adalah pada saat bit D akan dibalik dari 1 menjadi 0. Anda tahu bahwa bit D akan dibalik dari 1 menjadi 0 jika dan hanya jika (kondisi untuk membalik nilai bit D terpenuhi) DAN (current state dari bit D adalah 1). Cara untuk menentukan hal tersebut pada sirkuit ***Counter with Parallel Load*** yang normal akan berguna dalam menjawab section ini dan dalam mendesain sirkuit.

#PSDisFun❤️

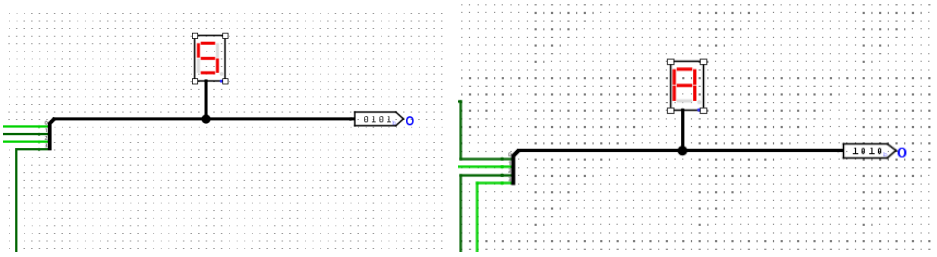
4 Bit Modified Parallel Load Counter

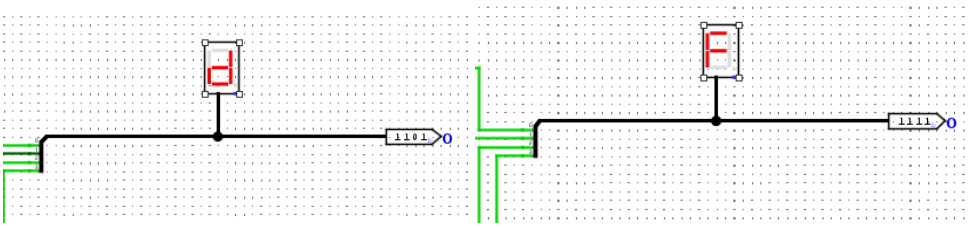
(60 Poin)

Setelah menjawab pertanyaan-pertanyaan di atas, buatlah sirkuit 4 bit parallel load dengan spesifikasi berikut:

- Circuit Name : `Modified4BitCounter`
- Input : `Clock, I, Load, Count, Inc2, Inc4, Inc8`
- Flip-flop : `A, B, C, D`
- Output : `O, CarryOut`
- Where:
  - I adalah 4 bit load input
  - Load, Count, Inc2, Inc4, dan Inc8 adalah 1 bit control input
  - Clock adalah 1 bit input yang akan digunakan sebagai clock
  - A, B, C, D adalah bit yang merepresentasikan counter (A adalah MSB, D adalah LSB)
  - O adalah 4 bit output yang menampilkan current state dari counter
  - CarryOut adalah 1 bit output yang memberi sinyal bahwa pada clock cycle berikutnya bit A akan berubah dari 1 menjadi 0
- Instruksi:
  - **Connect output O dengan sebuah hex digit display (ditemukan di Input/Output folder).**
- Info tambahan:
  - Tidak diperbolehkan untuk memakai multiplexer dan decoder dari logisim
  - **Anda dapat mendapatkan test case/contoh dari circuit yang berkerja dengan baik di bagian 'Test cases to test Modified4BitCounter' di akhir dokumen lab.**

contoh dari hex digit display:





#PSDisFun❤️

16 Bit Parallel Load Counter

(24 Poin)

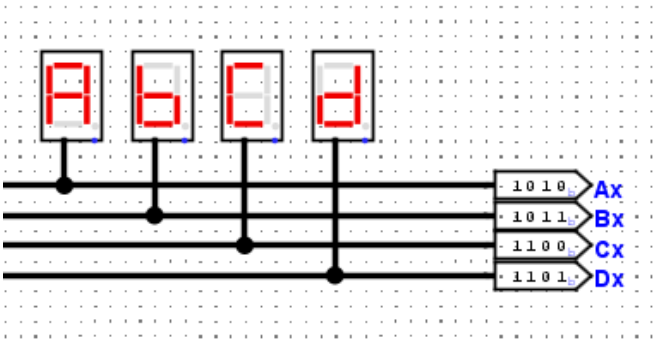
Pak Esde puas dengan 4 bit counter yang anda telah buat, namun ia sekarang ingin berhitung sampai dengan angka yang lebih besar dari 15, Ia ingin anda untuk menggunakan ***modified 4 bit parallel load counter*** untuk membuat ***modified 16 bit parallel load counter*** yang menghitung mulai dari **$(0000)_{16}$ atau (0000 0000 0000 0000)$_2$ sampai dengan $(FFFF)_{16}$ atau (1111 1111 1111 1111)$_2$** , dengan fungsionalitas yang sama dengan versi 4 bit.

Buatlah 16 bit parallel load counter dengan spesifikasi berikut:

- Circuit Name : `Modified16BitCounter`
- Input : `Wx, Xx, Yx, Zx, Load, Count, Inc2, Inc4, Inc8`
- Output : `Ax, Bx, Cx, Dx`
- Di mana:
  - Wx - Zx adalah 4 bit input yang masing-masing merepresentasikan satu hexadecimal digit dari 16 bit load input (Wx adalah the Most Significant hexadecimal digit dan Zx is adalah Least Significant hexadecimal digit)
  - Load, Count, Inc2, Inc4 dan Inc 8 adalah 1 bit control input
  - Ax - Dx adalah 4 bit output yang masing-masing merepresentasikan satu hexadecimal digit dari current state dari 16 bit counter (Ax adalah Most Significant hexadecimal digit dan Dx adalah Least Significant hexadecimal digit)
- Instruksi:
  - Connect setiap 4 bit output dengan sebuah hex digit display. Pastikan bahwa setiap hex digit display diletakkan sehingga leftmost hex digit display adalah untuk Most Significant hexadecimal digit [Ax] dan rightmost hex digit display untuk Least Significant hexadecimal digit [Dx]
  - Ingat bahwa ini hanya sebuah counter dengan kapasitas lebih besar sehingga semua increments/increases harus dimulai dari Least significant digit.
- Additional Info:
  - Anda boleh menggunakan Constant (ditemukan di Wiring folder) jika ada inputs/nilai yang kamu butuhkan untuk tetap konstan
  - Gunakan clock normal yang dapat ditemukan di Wiring folder untuk sirkuit ini,
  - **Anda dapat mendapatkan test case/contoh dari circuit yang berkerja dengan baik di bagian 'Test cases to test Modified16BitCounter' di akhir dokumen lab.**

#PSDisFun❤️

Contoh Hex digit display:



count terbaru sesuai dengan gambar di atas adalah angka hexadecimal bernilai 0xABCD

Wajib diingat bahwa plagiarisme adalah sebuah pelanggaran yang serius dan anda akan diberi nilai 0

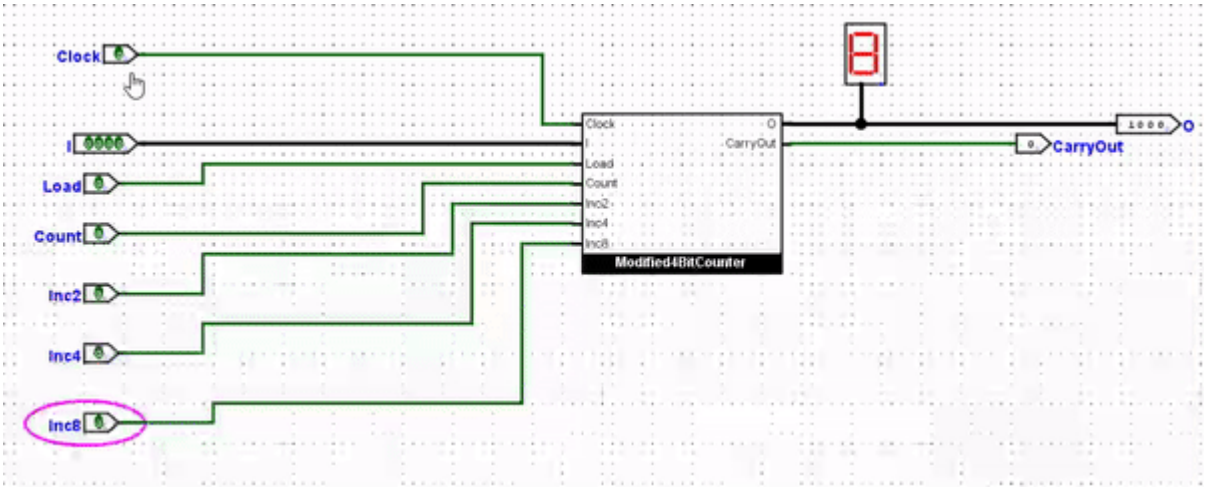
#PSDisFun❤️

Test cases to test Modified4BitCounter:

Note: jika gif tidak termuat atau anda ingin *pause* video contohnya, anda bisa cek folder ini untuk videonya.

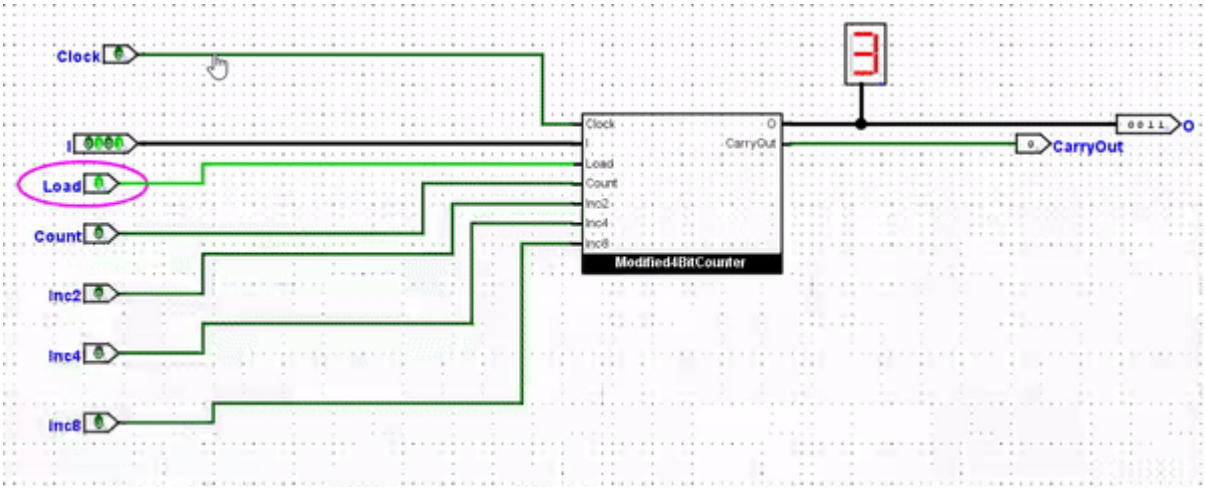
Testing Load:

- Pastikan sirkuitmu melakukan load I jika dan hanya jika Load = 1, terlepas dari control input lainnya
- Example 1: I = 1101, Load =1 , Count = 0, Inc2 =0, Inc4 = 0, Inc8 = 0: O = 1101(D)
- Example 2: I = 1001, Load =1 , Count = 1, Inc2 =1, Inc4 = 1, Inc8 = 1: O = 1001(9)
- Example 3: I = 1110, Load =1 , Count = 0, Inc2 =0, Inc4 = 1, Inc8 = 0: O = 1110(E)



Testing Increment Counter by 8:

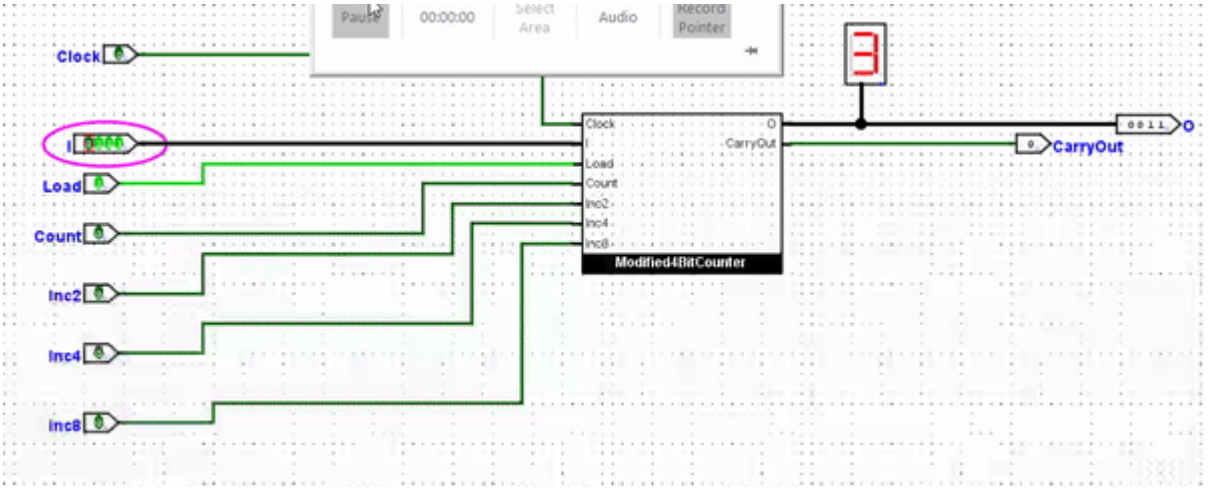
- Pastikan counter anda diincrement 8 jika dan hanya jika Load = 0, Count = 1 dan Inc8 = 1, terlepas dari control input lainnya.
- Example 1: Start at 0101(5), Load =0 , Count = 1, Inc2 =0, Inc4 = 0, Inc8 = 1: O = 0101(5) ⇒ 1101(D) ⇒ 0101(5) ⇒ Ulang
- Example 2: Start at 0011(3), Load =0 , Count = 1, Inc2 =1, Inc4 = 1, Inc8 = 1: O = 0011(3) ⇒ 1011(B) ⇒ 0011(3) ⇒ Ulang



#PSDisFun❤️

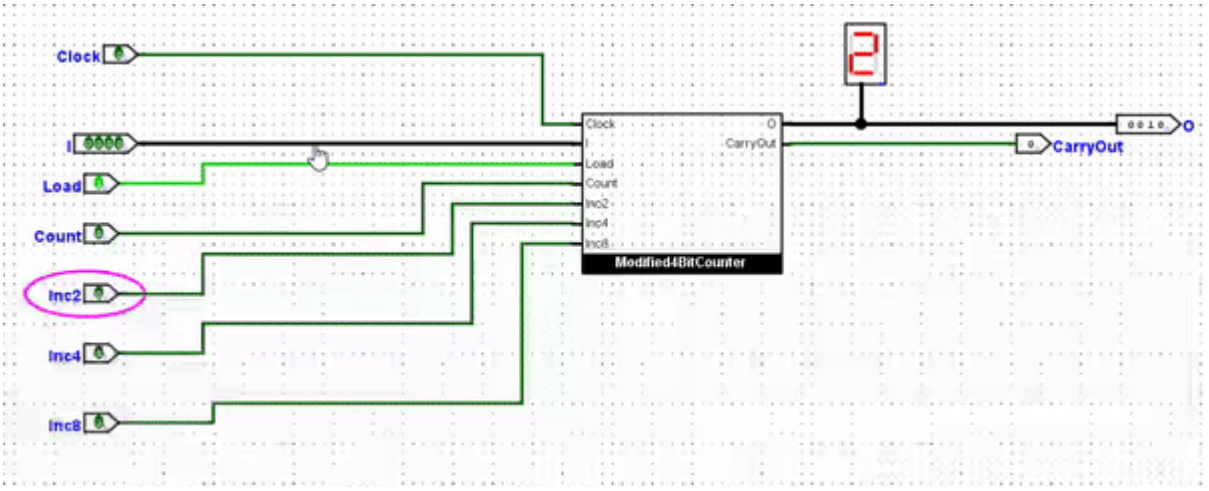
Testing Increment Counter by 4:

- Pastikan counter anda diincrement 4 ika dan hanya jika Load = 0, Count = 1, Inc8 = 0 and Inc4 = 1 terlepas dari control input Inc2.
- Example 1: Start at 0111(7), Load =0 , Count = 1, Inc2 =0, Inc4 = 1, Inc8 = 0: O = 0111(7) ⇒ 1011(B) ⇒ 1111(F) ⇒ 0011(3) ⇒ 0111(7) ⇒ Ulang
- Example 2: Start at 0010(7), Load =0 , Count = 1, Inc2 =1, Inc4 = 1, Inc8 = 0: O = 0010(2) ⇒ 0110(6) ⇒ 1010(A) ⇒ 1110(E) ⇒ 0010(2) ⇒ Ulang



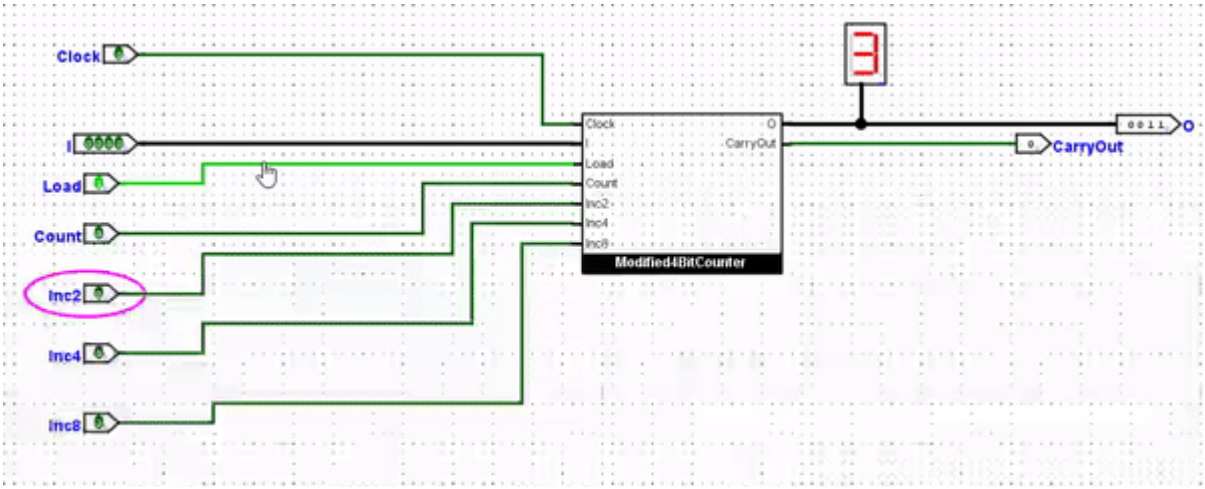
Testing Increment Counter by 2:

- Pastikan counter anda diincrement 2 jika dan hanya jika Load = 0, Count = 1, Inc8 = 0, Inc4 = 0 and Inc2 = 1
- Example 1: Start at 0000(0), Load =0 , Count = 1, Inc2 =1, Inc4 = 0, Inc8 = 0: O = 0000(0) ⇒ 0010(2) ⇒ 0100(4) ⇒ 0110(6) ⇒ 1000(8) ⇒ 1010(A) ⇒ 1100(C) ⇒ 1110(E) ⇒ 0000(0) ⇒ Ulang
- Example 2: Start at 0011(3), Load =0 , Count = 1, Inc2 =1, Inc4 = 0, Inc8 = 0: O = 0011(3) ⇒ 0101(5) ⇒ 0111(7) ⇒ 1001(9) ⇒ 1011(B) ⇒ 1101(D) ⇒ 1111(F) ⇒ 0001(1) ⇒ 0011(3) ⇒ Ulang



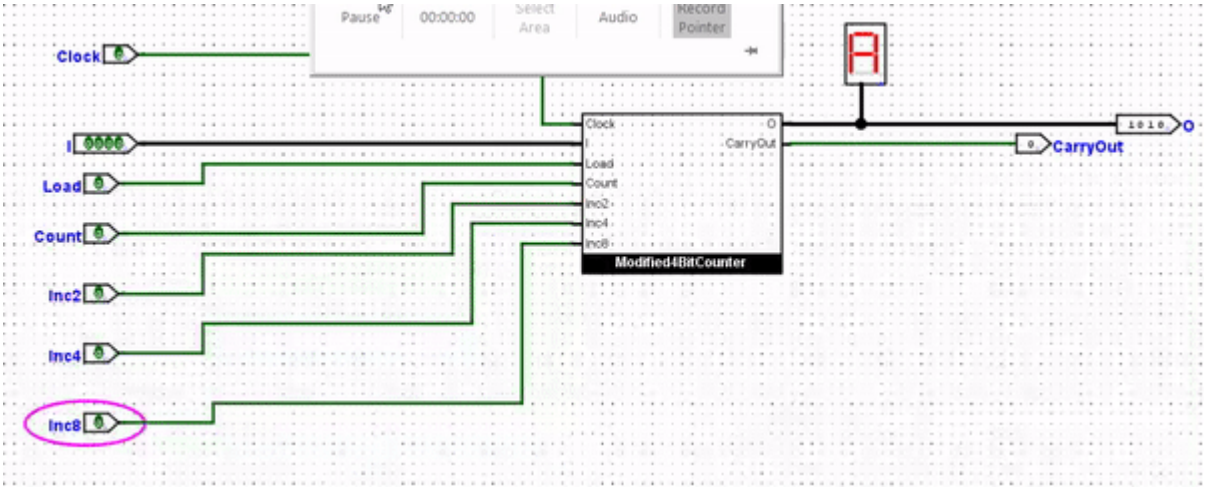
Testing Increment Counter by 1:

- Pastikan counter anda diincrement 1 jika dan hanya jika Load = 0, Count = 1, Inc8 = 0, Inc4 = 0 and Inc2 = 0
- Contoh: Start at 0000(0), Load =0 , Count = 1, Inc2 =0, Inc4 = 0, Inc8 = 0: 0 = 0000(0) ⇒ 0001(1) ⇒ 0010(2) ⇒ 0011(3) ⇒ 0100(4) ⇒ 0101(5) ⇒ 0110(6) ⇒ 0111(7) ⇒ 1000(8) ⇒ 1001(9) ⇒ 1010(A) ⇒ 1011(B) ⇒ 1100(C) ⇒ 1101(D) ⇒ 1110(E) ⇒ 1111(F) ⇒ 0000(0) ⇒ Ulang



Testing Do Nothing

- Pastikan tidak ada yang terjadi jika dan hanya jika Load = 0, Count = 0, terlepas dari nilai input lainnya.
- Contoh: Set counter menjadi 1010, I = 0000 (harus berbeda dari current count untuk memastikan counter anda tidak melakukan load dari I), Load = 0, Count = 0. Tes semua kemungkinan kombinasi dari Inc2, Inc4 and Inc8



#PSDisFun❤️

Test cases to test Modified16BitCounter:

Catatan: Di GIF/Video, subsirkuit Modified16BitCounter mempunyai Clock sebagai input. Hal tersebut dilakukan agar penulis lab bisa menyambungkan Clock ke subsirkuit untuk menunjukkan beberapa contoh berikut. Pada sirkuit Modified16BitCounter anda Clock anda harus menggunakan normal clock dari Wiring folder, BUKAN sebuah input.

Testing Load:

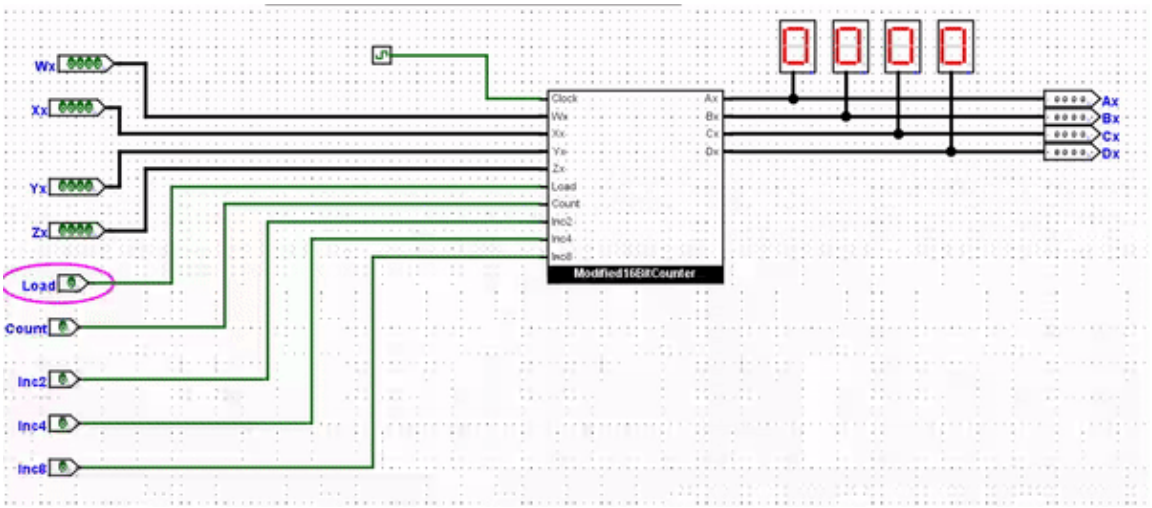
- Pastikan counter anda melakukan load Wx-Zx jika dan hanya jika Load = 1, terlepas dari nilai control inputs lainnya.

Example 1:

- Input:
  - Wx = 1010, Xx = 1011, Yx = 1100, Zx = 1101
  - Load =1 , Count = 0, Inc2 =0, Inc4 = 0, Inc8 = 0
- Hasil:
  - Ax = 1010(A), Bx = 1011(B), Cx = 1100(C) Dx = 1101(D): 0xABCD

Example 2:

- Input:
  - Wx = 1010, Xx = 1011, Yx = 1100, Zx = 1000
  - Load =1 , Count = 1, Inc2 =0, Inc4 = 1, Inc8 = 0
- Hasil:
  - Ax = 1010(A), Bx = 1011(B), Cx = 1100(C) Dx = 1000(8): 0xABC8

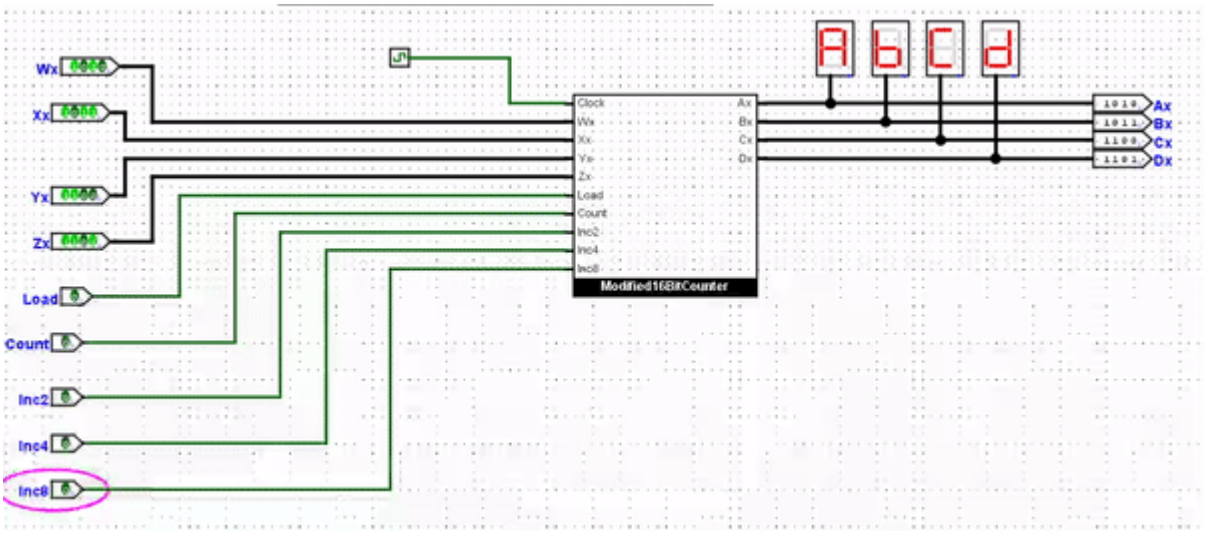


Testing Increment Counter by 8:

- Pastikan counter anda diincrement 8 jika dan hanya jika Load = 0, Count = 1 and Inc8 = 1, terlepas dari nilai control input lainnya.
- Contoh:
  - Start at 0xABCD
  - Load =0 , Count = 1, Inc2 = X, Inc4 = X, Inc8 = 1
  - 0xABCD ⇒ 0xABD5 ⇒ 0xABDD ⇒ 0xABE5 ⇒ 0xABED ⇒ 0xABF5 ⇒ 0xABFD ⇒ 0xAC05 ⇒ 0xAC0D ⇒ 0xAC15 ⇒ 0xAC1D ⇒ 0xAC25 ⇒

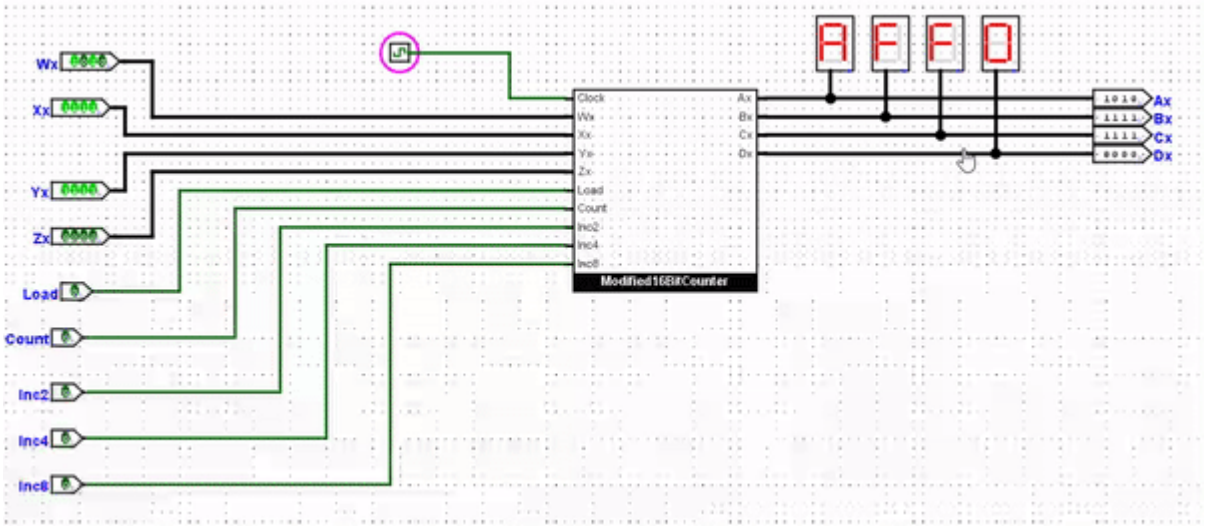
0xAC2D ⇒ 0xAC35 ⇒ 0xAC3D ⇒ 0xAC45 ⇒ 0xAC4D ⇒ 0xAC55 ⇒ 0xAC5D ⇒ …

- Di GIF/VIdeo, penulis mengubah Inc2 dan Inc4 untuk menunjukkan bahwa control input barusan tidak berpengaruh selama Load=0, Count = 1 dan Inc8 = 1



Testing Increment Counter by 4:

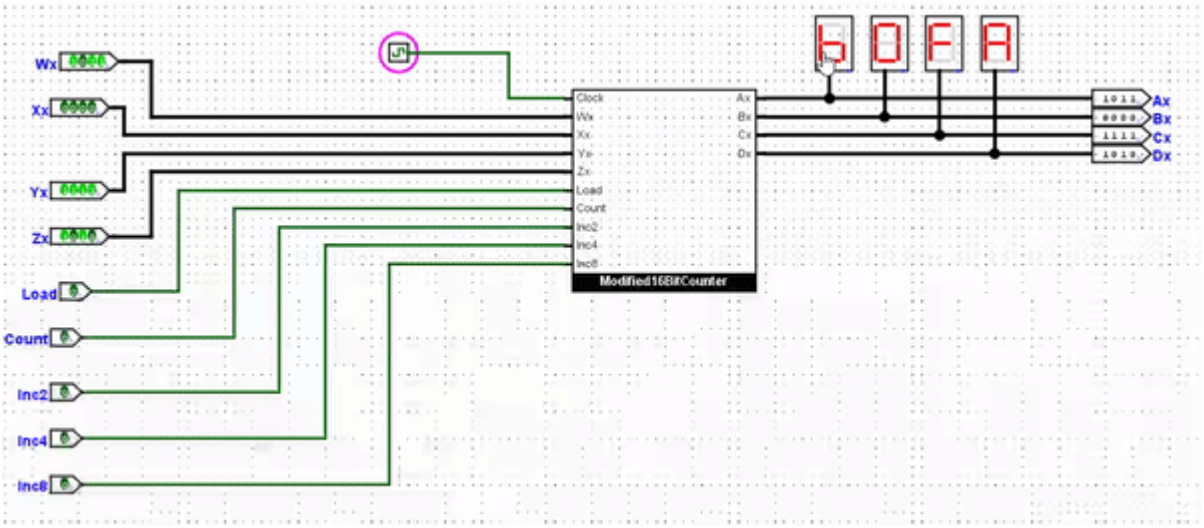
- Pastikan counter anda diincrement 4 jika dan hanya jika Load = 0, Count = 1, Inc8 = 0 dan Inc4 = 1, terlepas dari nilai Inc2.
- Contoh:
  - Mulai dari 0xAFF0
  - Load =0 , Count = 1, Inc2 = X, Inc4 = 1, Inc8 = 0
  - 0xAFF0 ⇒ 0xAFF4 ⇒ 0xAFF8 ⇒ 0xAFFC ⇒ 0xB000 ⇒ 0xB004 ⇒ 0xB008 ⇒ 0xB00C ⇒ 0xB010 ⇒ 0xB014 ⇒ 0xB018 ⇒ 0xB01C ⇒ 0xB020 ⇒ 0xB024 ⇒ 0xB028 ⇒ 0xB02C ⇒ …
- Di GIF/video, penulis mengganti-ganti Inc2 untuk menunjukkan Inc2 tidak berpengaruh selama Load=0, Count = 1 dan Inc8 = 0, Inc4 = 1



#PSDisFun❤️

Testing Increment Counter by 2:

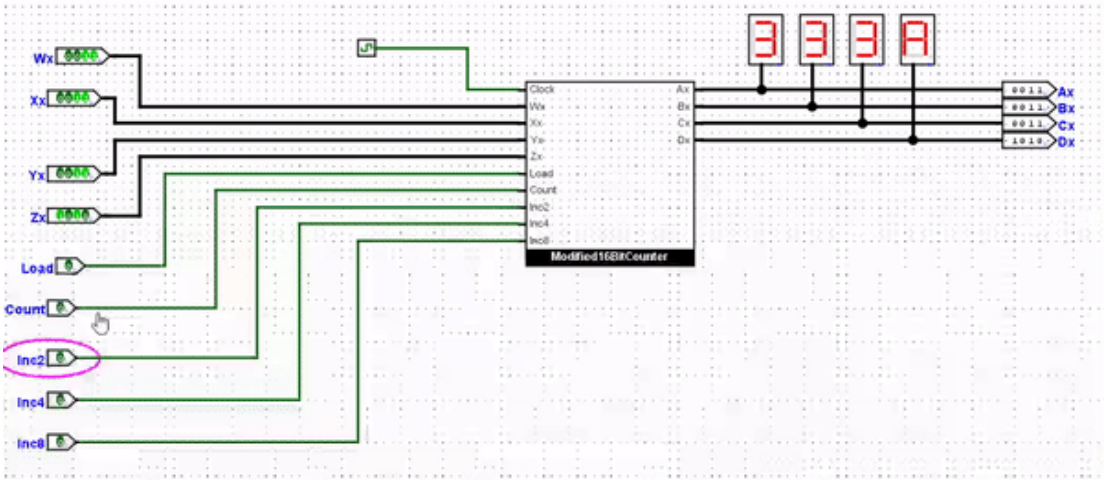
- Pastikan counter diincrement 2 jika dan hanya jika Load = 0, Count = 1, Inc8 = 0, Inc4 = 0 dan Inc2 = 1.
- Contoh:
    - Mulai dari 0xB0FA
    - Load =0 , Count = 1, Inc2 = 1, Inc4 = 0, Inc8 = 0
    - 0xB0FA ⇒ 0xB0FC ⇒ 0xB0FE ⇒ 0xB100 ⇒ 0xB102 ⇒ 0xB104 ⇒ 0xB106 ⇒ 0xB108 ⇒ 0xB10A ⇒ 0xB10C ⇒ 0xB10E ⇒ 0xB110 ⇒ …



Testing Increment Counter by 1:

- Pastikan counter diincrement 1 jika dan hanya jika Load = 0, Count = 1, Inc8 = 0, Inc4 = 0 dan Inc2 = 0.
- Contoh:
    - Mulai dari 0x333A
    - Load =0 , Count = 1, Inc2 = 0, Inc4 = 0, Inc8 = 0
    - 0x333A ⇒ 0x333B ⇒ 0x333C ⇒ 0x333D ⇒ 0x333E ⇒ 0x333F ⇒ 0x3340 ⇒ 0x3341 ⇒ 0x3342 ⇒ 0x3343 ⇒ 0x3344 ⇒ 0x3345 ⇒ 0x3346 ⇒ 0x3347 ⇒ 0x3348 ⇒ 0x3349 ⇒ 0x334A ⇒ 0x334B ⇒ 0x334C ⇒ 0x334D ⇒ 0x334E ⇒ 0x3350 ⇒ …

#PSDisFun❤️



Testing Do Nothing

- Pastikan tidak terjadi apa-apa jika dan hanya jika Load = 0, Count = 0, terlepas dari nilai control input lainnya.
- Contoh:
  - Mulai dari 0xB000
  - Load =0 , Count = 0, Inc2 = X, Inc4 = X, Inc8 = X
  - 0xB000 ⇒ 0xB000 ⇒ Ulang
- Di gif/video, penulis mengganti-ganti Inc2, Inc4 dan inc8 untuk menunjukkan control input barusan tidak berpengaruh selama Load=0 and Count = 0