Code and Circuits

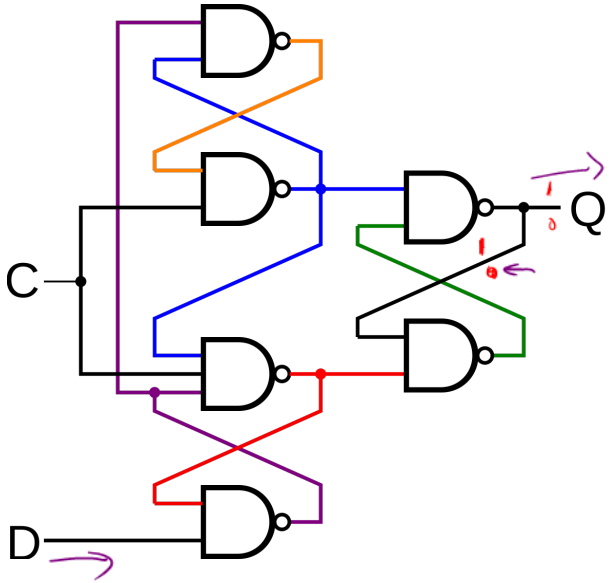
CS 2130: Computer Systems and Organization 1

September 14, 2022

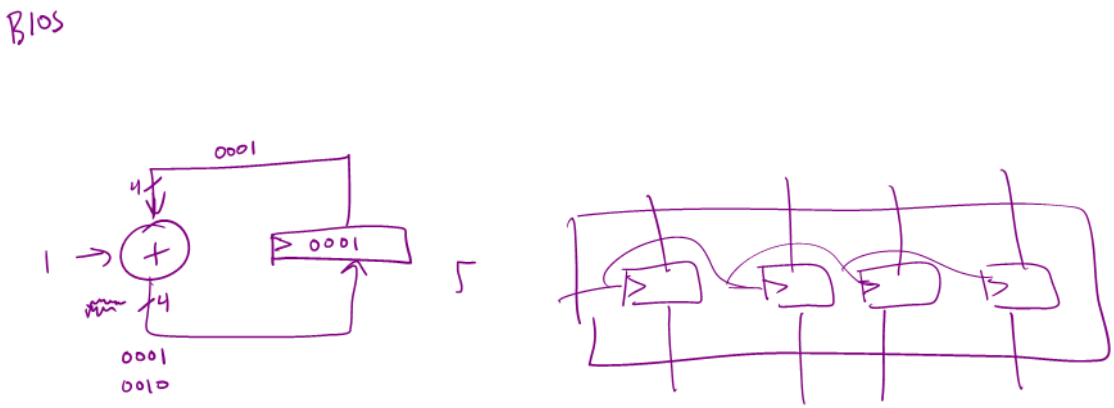
Announcements

- Homework 1 due tonight!
- Homework 2 available (due on Gradescope, 11pm Monday)

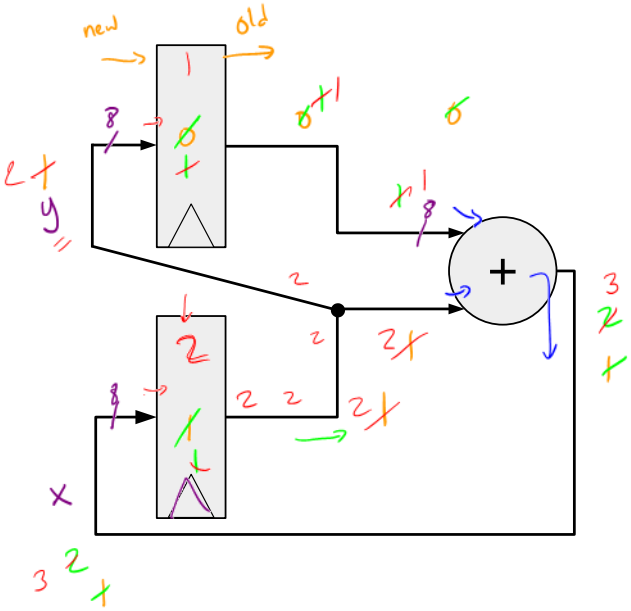
1-bit Register Circuit



Building a Counter

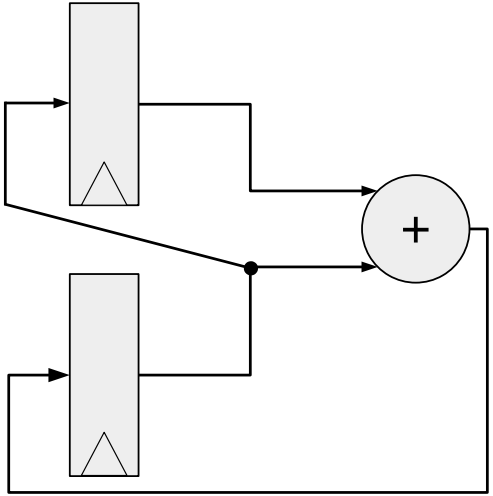


Another Circuit

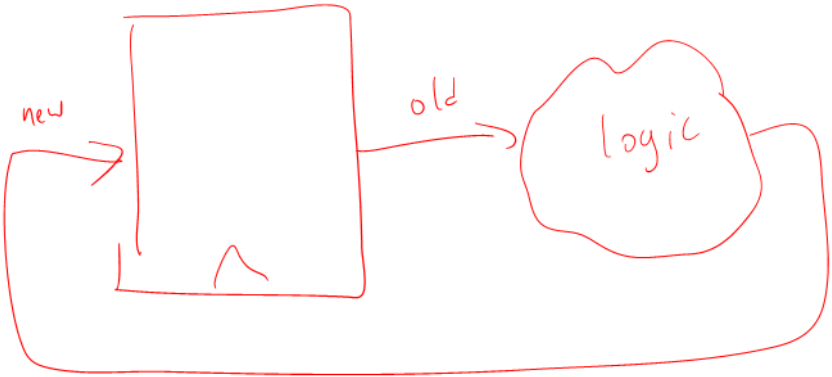


x	y	tick
0	1	0
1	0	1
1	1	2
2	1	3
3	2	4
5	3	5
8	5	6

Another Circuit



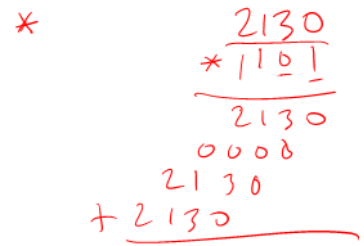
Common Model in Computers



Code to Build Circuits from Gates

Write code to build circuits from gates

- Gates we *already* know: &, |, ^, ~
- Operations we can build from gates: +, -
- Others we can build:



Code to Build Circuits from Gates

Write code to build circuits from gates

- Gates we *already* know: &, |, ^, ~
- Operations we can build from gates: +, -
- Others we can build:
- Ternary operator: ? :

z = b ? x : y

if(b)
  z = x
else
  z = y

z = (a == b ? 32 : x) * y

2x and or

Equals

Equals: =

- Attach with a wire (i.e., connect things)
- Ex: `z = x * y`

Equals

Equals: `=`

- Attach with a wire (i.e., connect things)
- Ex: `z = x * y`
- What about the following?

```
x = 1
x = 0
```

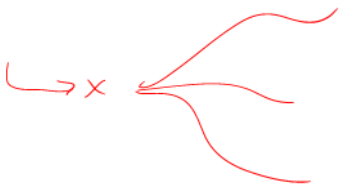
Equals

Equals: `=`

- Attach with a wire (i.e., connect things)
- Ex: `z = x * y`
- What about the following?

  ```
  x = 1
  x = 0
  ```

- **Single assignment**: each variable can only be assigned a value once



Comparisons

Each of our comparisons in code are straightforward to build:

- == - xor then nor bits of output

x == y

~(or all bits (x ^ y))

!

x = 1001

y = 1001

0000 = 0

Comparisons

Each of our comparisons in code are straightforward to build:

- `==` - xor then nor bits of output
- `!=` - same as `==` without not of output

Comparisons

Each of our comparisons in code are straightforward to build:

- `==` - xor then nor bits of output
- `!=` - same as `==` without not of output
- `<` - consider `x < 0`

$x - y < 0$

Comparisons

Each of our comparisons in code are straightforward to build:

- `==` - xor then nor bits of output
- `!=` - same as `==` without not of output
- `<` - consider `x < 0`
- `>`, `<=`, `=>` are similar