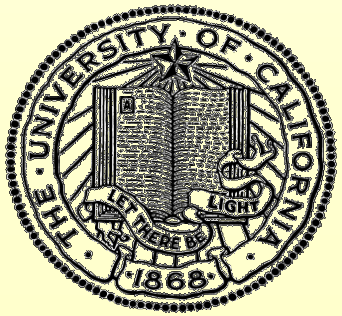
Arithmetic and Logical Operations



Midterm syllabus

- Everything before Floating Point representation
- Will announce a more formal syllabus soon
- 2 sample midterm exams already available on google drive

Logical Operations

- Operate on raw bits with 1 = true and 0 = false

		AND	OR	NAND	NOR	XOR	XNOR
In1	In2	&	|	~(&)	~(|)	^	~(^)
0	0	0	0	1	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	1	0	0	0	1

Logical Operations

- "bit-wise" logical operations are done in parallel for corresponding bits
  - Example & (AND):
    - X = 0011
    - Y = 1010
    - X AND Y = ?

Logical Operations

- "bit-wise" logical operations are done in parallel for corresponding bits
  - Example & (AND):
    - X = 0011
    - Y = 1010
    - X AND Y = X & Y = 0010

Logical Operations

- "bit-wise" logical operations are done in parallel for corresponding bits
  - Example (&) Reduction:
    - X = 0011
    - & X = ?

Logical Operations

- "bit-wise" logical operations are done in parallel for corresponding bits
  - Example (&) Reduction:
    - X = 0011
    - & X = 0&0&1&1 = 0

Shifts and Rotates: Logical Right

- **Logical right**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Introduce a 0 into the MSB

Shifts and Rotates: Logical Right Example

- **Logical right**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Introduce a 0 into the MSB
    - 00110101

→ Shift right by 1

Shifts and Rotates: Logical Right Example

- **Logical right**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Introduce a 0 into the MSB
    - 00110101
    - 00110101 ⟶

Shifts and Rotates: Logical Right Example

- **Logical right**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Introduce a 0 into the MSB
    - 00110101
    - 00110101 ⟶

Shifts and Rotates: Logical Right Example

- **Logical right**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Introduce a 0 into the MSB
    - 00110101
    - 0011010

Shifts and Rotates: Logical Right Example

- **Logical right**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Introduce a 0 into the MSB
    - 00110101
    - 00011010

Shifts and Rotates: Logical Right Example

- **Logical right**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Introduce a 0 into the MSB
    - 00110101 shift right by 1 -> 00011010

Shifts and Rotates: Logical Left

- **Logical left**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce a 0 into the LSB

Shifts and Rotates: Logical Left Example

- **Logical left**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce a 0 into the LSB

00110101

← Shift left by 2

Shifts and Rotates: Logical Left Example

- **Logical left**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce 0s into the LSB

00110101

← 00110101

Shifts and Rotates: Logical Left Example

- **Logical left**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce 0s into the LSB

00110101

← 00110101

Shifts and Rotates: Logical Left Example

- **Logical left**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce 0s into the LSB

00110101

← 110101

Shifts and Rotates: Logical Left Example

- **Logical left**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce 0s into the LSB

00110101

11010100

Shifts and Rotates: Logical Left Example

- **Logical left**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce 0s into the LSB

00110101 shift left by 2 => 11010100

Note: Logical Shift left by 1 = Multiply by 2

Note: Logical Shift left by 2 = Multiply by 4

Note: Logical Shift left by 3 = Multiply by 8

Shifts and Rotates: Arithmetic Right

- **Arithmetic right shift**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Reproduce the original MSB into the new MSB (preserve the sign!!!)

Shifts and Rotates: Arithmetic Right Examples

- **Arithmetic right shift**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Reproduce the original MSB into the new MSB

00110101 1100

⟶ Shift right by 2

Shifts and Rotates: Arithmetic Right Examples

- **Arithmetic right shift**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Reproduce the original MSB into the new MSB

00110101 1100

00110101 1100 →

Shifts and Rotates: Arithmetic Right Examples

- **Arithmetic right shift**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Reproduce the original MSB into the new MSB

00110101 1100

00110101 1100

Shifts and Rotates: Arithmetic Right Examples

- **Arithmetic right shift**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Reproduce the original MSB into the new MSB

00110101 1100

001101 11

Shifts and Rotates: Arithmetic Right Examples

- **Arithmetic right shift**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Reproduce the original MSB into the new MSB

00110101 1100

00001101 1111

Shifts and Rotates: Arithmetic Right Examples

- **Arithmetic right shift**
  - Move bits to the right, same order
  - Throw away the bit that pops off the LSB
  - Reproduce the original MSB into the new MSB

00110101 arithmetic shift right by 2 -> 00001101

1100 arithmetic shift right by 2 -> 1111

Shifts and Rotates: Arithmetic Left

- **Arithmetic left shift**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce a 0 into the LSB

Shifts and Rotates: Arithmetic Left Example

- **Arithmetic left shift**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce a 0 into the LSB

00110101

← Shift left by 2

Shifts and Rotates: Arithmetic Left Example

- **Arithmetic left shift**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce a 0 into the LSB

00110101

00110101 ←

Shifts and Rotates: Arithmetic Left Example

- **Arithmetic left shift**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce a 0 into the LSB

00110101

← 00110101

Shifts and Rotates: Arithmetic Left Example

- **Arithmetic left shift**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce a 0 into the LSB

00110101

110101

Shifts and Rotates: Arithmetic Left Example

- **Arithmetic left shift**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce a 0 into the LSB

00110101

11010100

Shifts and Rotates: Arithmetic Left Example

- **Arithmetic left shift**
  - Move bits to the left, same order
  - Throw away the bit that pops off the MSB
  - Introduce a 0 into the LSB

00110101 arithmetic shift left by 2 -> 11010100

Shifts and Rotates: Rotate Left

- **Rotate left**
  - Move bits to the left, same order
  - Put the bit(s) that pop off the MSB into the LSB
  - No bits are thrown away or lost

Shifts and Rotates: Rotate Left Examples

- **Rotate left**
  - Move bits to the left, same order
  - Put the bit(s) that pop off the MSB into the LSB
  - No bits are thrown away or lost

00110101 1100

Rotate left 1

Shifts and Rotates: Rotate Left Examples

- **Rotate left**
  - Move bits to the left, same order
  - Put the bit(s) that pop off the MSB into the LSB
  - No bits are thrown away or lost

00110101 1100

00110101 1100

Rotate left 1

Shifts and Rotates: Rotate Left Examples

- **Rotate left**
  - Move bits to the left, same order
  - Put the bit(s) that pop off the MSB into the LSB
  - No bits are thrown away or lost

00110101 1100

00110101 1100

Shifts and Rotates: Rotate Left Examples

- **Rotate left**
  - Move bits to the left, same order
  - Put the bit(s) that pop off the MSB into the LSB
  - No bits are thrown away or lost

00110101 1100

01101010 1001

Shifts and Rotates: Rotate Left Examples

- **Rotate left (**Rotate left 1**)**
  - Move bits to the left, same order
  - Put the bit(s) that pop off the MSB into the LSB
  - No bits are thrown away or lost

00110101 rotate left by 1 -> 01101010

1100 rotate left by 1 -> 1001

Shifts and Rotates: Rotate Right

- **Rotate right**
  - Move bits to the right, same order
  - Put the bit that pops off the LSB into the MSB
  - No bits are thrown away or lost

Shifts and Rotates: Rotate Right Examples

- **Rotate right (rotate right 2)**
  - Move bits to the right, same order
  - Put the bit that pops off the LSB into the MSB
  - No bits are thrown away or lost

00110101 → 1101

Shifts and Rotates: Rotate Right Examples

- **Rotate right (rotate right 2)**
  - Move bits to the right, same order
  - Put the bit that pops off the LSB into the MSB
  - No bits are thrown away or lost

00110101 1101

00110101 1101

Shifts and Rotates: Rotate Right Examples

- **Rotate right (rotate right 2)**
  - Move bits to the right, same order
  - Put the bit that pops off the LSB into the MSB
  - No bits are thrown away or lost

00110101 1101

00110101 1101

Shifts and Rotates: Rotate Right Examples

- **Rotate right (rotate right 2)**
  - Move bits to the right, same order
  - Put the bit that pops off the LSB into the MSB
  - No bits are thrown away or lost

00110101 1101

01001101 0111

Shifts and Rotates: Rotate Right Examples

- **Rotate right (rotate right 2)**
  - Move bits to the right, same order
  - Put the bit that pops off the LSB into the MSB
  - No bits are thrown away or lost

    00110101 rotate right by 2 -> 01001101

    1101 rotate right by 2 -> 0111

Binary Addition

x	0011
+y	0001

Or in tabular form…

Carry In	A	B	Sum	Carry Out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

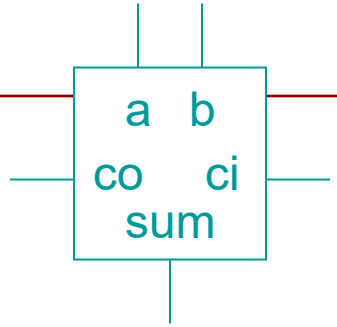
Binary Addition

```
    0 1 1
 x  0011
+y  0001
sum 0100
```

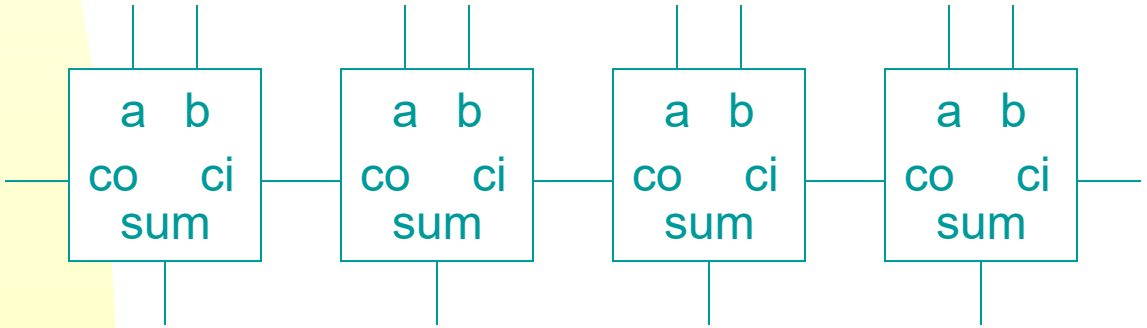
Or in tabular form…

Carry In	A	B	Sum	Carry Out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Binary Addition



- And as a full adder…
- 4-bit Ripple-Carry "adder":
  - Carry values propagate from bit to bit
  - Like pencil-and-paper addition
  - Time proportional to number of bits
  - "Lookahead-carry" can propagate carry proportional to log(n) with more space.



Addition: unsigned

- Just like the simple addition given earlier.
- Examples: (we are ignoring overflow for now)


```
  100001 (33)          00001010 (10)
+ 011101 (29)        + 00001110 (14)
```


Addition: unsigned

- Just like the simple addition given earlier.
- Examples: (we are ignoring overflow for now)


```
  0 0 0 0 1          0 1 1 1 0
    100001 (33)          00001010 (10)
+ 011101 (29)        + 00001110 (14)
  ------               --------
  111110 (62)          00011000 (24)
```

Addition: 2's complement

- Just like unsigned addition
- Assume 6-bit and observe:
  - Ignore carry-outs (overflow)
  - Sign bit is in the 2^{n-1} bit position
  - What does this mean for adding different signs?


```
 000011 (3)    101000 (-24)    111111 (-1)
+111100 (-4) +010000 (16)    +001000 (8)
```

Addition: 2's complement

- Just like unsigned addition
- Assume 6-bit and observe:
  - Ignore carry-outs (overflow)
  - Sign bit is in the 2^{n-1} bit position
  - What does this mean for adding different signs?


```
  000011  (3)     101000  (-24)     111111  (-1)
+111100   (-4)  +010000   (16)    +001000   (8)
--------        --------          --------
  111111  (-1)    111000  (-8)      000111  (7)
```

More examples: Convert to 2SC and do the addition

-20 + 15

5 + 12

-12 + -25

Do at HOME!!!

Subtraction: 2's complement

- Don't. Just use addition:
  - x – y -> x + (-y) - > x + y' + 1

Example:


```
  10110  (-10)
- 00011  (3)
--------
```

Subtraction: 2's complement

- Don't. Just use addition:
  - x – y -> x + (-y) - > x + y' + 1

Example:


```
  10110  (-10)                    10110  (-10)
- 00011  (3)      ------->      + 11100  (-3)
                                 +     1
                                 -------
```

Subtraction: 2's complement

- Don't. Just use addition:
  - x – y -> x + (-y) - > x + y' + 1

Example:


```
   10110  (-10)              1 1
 - 00011  (3)      -->       10110  (-10)
   -----                   + 11100  (-3)
                           +     1
                           -------
                         1  10011
```

Subtraction: 2's complement

- Addition and subtraction are simple in 2's complement,
  just need an adder and inverters.
- Can also flip bits of 10 and add an LSB carry in, so for -10 - 3 we get:


```
      1
  10110
+ 11100
-------
```

"add 1"

"flip bits of bottom number"

(throw away carry out)

Overflow in Addition

- **Unsigned:** When there is a carry out of the MSB

1000 (8)
+1001 (9)
1 0001 (1)

Overflow in Addition

- **2's complement:** When the signs of the addends are the same, but the sign of the result is different
- Adding 2 numbers of opposite signs never overflows.


```
  0011 (3)
+ 0110 (6)
------
  1001 (-7)
```