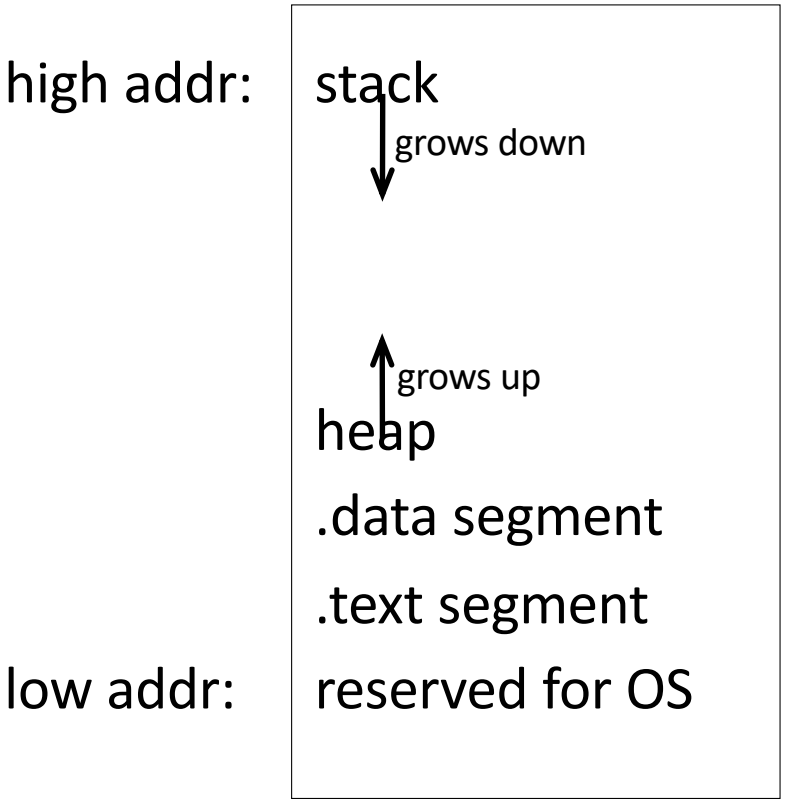
ASM function practice and binary

1/15/25

Administrivia

- HW 1 (V in assembly) due tonight

Recall: Recursion using the stack



Stack composed of "activation records" or "stack frames", each with the local variables and saved registers for one function call

Bottom of the stack is stored in $sp (stack pointer)

To reserve another frame:

$sp = $sp – (frame size)

To free the frame:

$sp = $sp + (frame size)

Computing the Fibonacci numbers

```
int fib(int n) {
        if(n < 2)
                return n;

        int val1 = fib(n-1);
        int val2 = fib(n-2);
        return val1 + val2;
}
```

Decimal numbers (base 10)

10000s digit	1000s digit	100s digit	10s digit	1s digit	
3	4	8	1	6	$= 34{,}816_{10}$

- The i^{th} digit from the right (starting with 0) is worth 10^i

Binary numbers (base 2)

16s bit	8s bit	4s bit	2s bit	1s bit	
1	0	1	1	0_2	= 16 + 4 + 2 = 22

- The i^{th} bit from the right (starting with 0) is worth 2^i
- Converting from binary: Determine value of each bit and add the results

What is the value of the binary number 1001110?

A. 42

B. 78

C. 108

D. 214

E. None of the above

What is the value of the binary number 1001110?

A. 42

B. 78

C. 108

D. 214

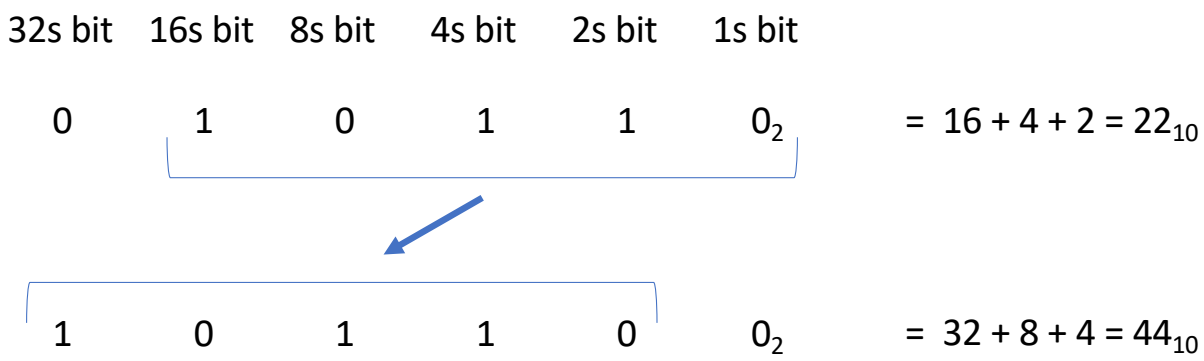
E. None of the above

Converting to binary

- Repeatedly divide by 2
  - Remainder at each step gives next least significant bit

• 214 / 2 = 107	remainder 0	0
• 107 / 2 = 53	remainder 1	10
• 53 / 2 = 26	remainder 1	110
• 26 / 2 = 13	remainder 0	0110
• 13 /2 = 6	remainder 1	10110
• 6 / 2 = 3	remainder 0	010110
• 3 / 2 = 1	remainder 1	1010110
• 1 / 2 = 0	remainder 1	11010110

Shifting operations



- Shifting left multiplies by 2
  - Each 1 contributes twice as much as before
- Shifting right divides by 2

Hexadecimal (base 16)

- Use a-f to represent 10-15
- $1ab4_{16}$ = 0x1ab4
- Each digit represents 4 bits

0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
a	10	1010
b	11	1011
c	12	1100
d	13	1101
e	14	1110
f	15	1111