CS 631-01 Parsing Binary Two's Comp

Lab02 due tonight
Project02 out today

Lab02

Parsing

LL left lookahead

LR
GLR
PEG
* speed
* left recursion

Left recursion

P ::= P 'A' EOT
| ε

ε 'A'
'AA'
'AAA'

```
parse_p( ) {

    parse_p( );
    accept(EOT);

}
```

p ::= ('A')* EOT

```
x = foo;

x = Foo(1);
```

TK_IDENT TK_ASSIGN TK_IDENT TK_LPAREN
... TK_INTLIT TK_RPAREN

LL(1)
LL(2)
LL(K)

LL(*)

Conversions

atoi()

strtol()

printf("%x", x)

CONV.C

uint32_t binstr_to_int(char *s)

uint32_t intstr_to_int(char *s)

uint32_t hexstr_to_int(Char *s)

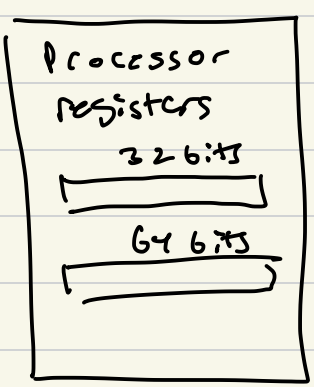
uint32_t str_to_int(char *s, int base)

---

int x;

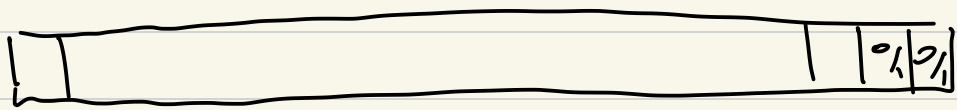
uint32_t y;

32 bits



1 byte = 8 bits

32 bit value = 4 bytes



most significant bit

least significant bit

unsigned int (uint32_t) 0 to $(2^{32}-1)$

4 billion

Memory

$2^{10} = 1024$ (K) KB

$2^9 = 512$

$2^8 = 256$

2^{20} = (M) MB 2^{30} = (G)

$2^8 = 256$

$2^3 = 8$ 1 byte

int x signed

$\frac{2^{32}}{2} = 2^{31}$

(-2^{31}) to $(2^{31}-1)$

0

Missing RPAREN

(1+2 EOT

parse_oper()

accept('(')

parse_expr()

accept_(')')

Two's Complement

3 bit values

bin	unsigned dec	signed magnitude	Two's comp
000	0	0	0
001	1	1	1
010	2	2	2
011	3	3	3
100	4	-0	-4
101	5	-1	-3
110	6	-2	-2
111	7	-3	-1

$$\begin{array}{r} {}^{1}9 \\ + \ 1 \\ \hline 10 \end{array}$$

Problems with signed magnitude

1) Two zero
2) addition

$$\begin{array}{r} 3 \\ + -1 \\ \hline 2 \end{array} \qquad \begin{array}{r} {}^{1\,1} \\ 1011 \\ + \ 101 \\ \hline 1000 \end{array}$$

Two's comp invert(x) + 1

010(2) → 101 + 1 = 110 -2

```
      1 1
  3   1 0 1 1
+ -1  + 1 1 1
      -------
        0 1 0   (2)
```

Generally,

n bits two's comp

$(-2^{(n-1)})$ to $(2^{(n-1)} - 1)$

```
#include <stdint.h>
```

uint32_t   uint8_t

TK_INTLIT   TK_BINLIT   TK_HEXLIT

↓

uint32_t value;

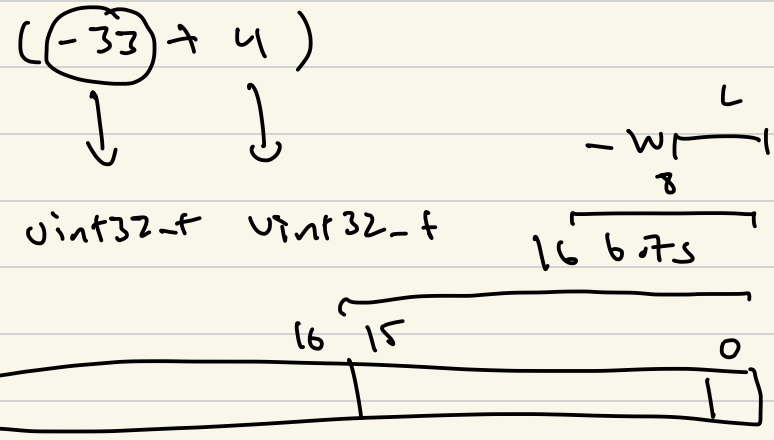
ntlang -b base -w width

-b 10 31

-8

-b 2 0b0110

-b 16 0xA3



Atlang -w 4 -e "0x3F"

0xF

-1

Bitwise operators

~ not / invert ~0b1001 = 0b0110

& and

```
  0b1100
  0b0111
---------
  0b0100
```

| or

```
  0b1100
  0b0110
---------
  0b1110
```

^ xor

```
  0b0101
  0b0011
---------
  0b0110
```

Shifts

<< left shift

>> right shift

x << 2 — LSL logical shift left

2 bits

x >> 2 — LSR logical shift right; ASR Arithmetic shift right

2 bits

Aplanar

<<	LSL
>>	LSR
>-	ASR