CS 631-02 Parsing Binary Two's Comp

Lab 02 due tonight
Project 02 out today

Parsing

LL

LR
GLR
PEG

* speed
* left recursion

Left recursion

```
P ::= P 'A' EOT
    | ε
```

''
'A'
'AA'
'AAA'

```
parse_p ()    {
        parse_p()
        accept(EOT)
}
```

p ::= ('A')* EOT

```
X = foo;
    ↑
X = foo(1);
    ↑
```

TK_IDENT(f) TK_ASSIGN TK_IDENT TK_LPAREN
TK_INTLIT TK_RPAREN TK_SC

LL(1)
LL(2)
LL(k)

LL(*)

Conversions

atoi()

strtol()

printf("%x", x);

conv.c

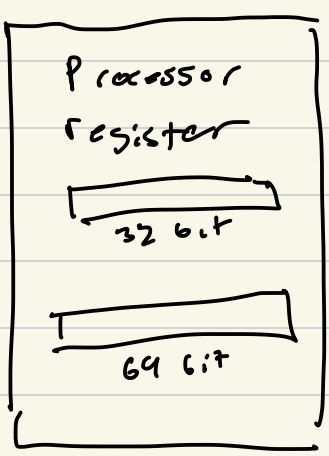
0b11011

```
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);
```

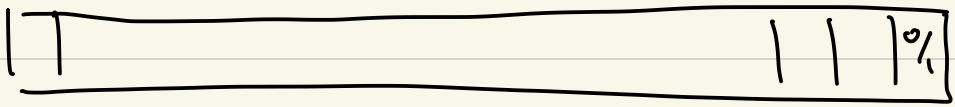
```
int x;
uint32_t y;
```

} 32 bits



32 bit value 1 byte = 8 bits

4 bytes



31 0

Most significant bit

least significant bit

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

$2^{32}-1$

$2^0=1$	$2^4=16$	$2^8=256$
$2^1=2$	$2^5=32$	$2^9=512$
$2^2=4$	$2^6=64$	$2^{10}=1024$
$2^3=8$	$2^7=128$	$2^{11}=2048$

$2^{10}=1$ KB $2^{20}=1$ MB $2^{30}=1$ GB

$\frac{2^{32}}{2^{30}}=2^2=4$ GB

int x; Signed

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

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

0

Twos Complement

3 bits bin	dec(u)	Signed magnitude	Twos 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

Problems with signed magnitude

1) two zeros

2) addition

$$\begin{array}{r} 3 \\ +\ -1 \\ \hline 2 \end{array} \qquad \begin{array}{r} 11 \\ 0011 \\ +1101 \\ \hline 0000 \end{array} \ (0)$$

Twos Complement

invert(x)+1

```
  010 (2)
   ↓
  101
+   1
-----
  110 (-2)
```

```
   ①11
   011
+  111
-----
  010 (2)
```

```
  3
+-1
---
  2
```

Generally

n bit value in twos comp

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

```
#include <stdint>
```

uint32_t uint8_t

TK_INTLIT TK_BINLIT TK_HEXLIT

uint32_t value.

nt lang -b base -w width

-b 10	31
-b 2	0b1010
-b 16	0xA3

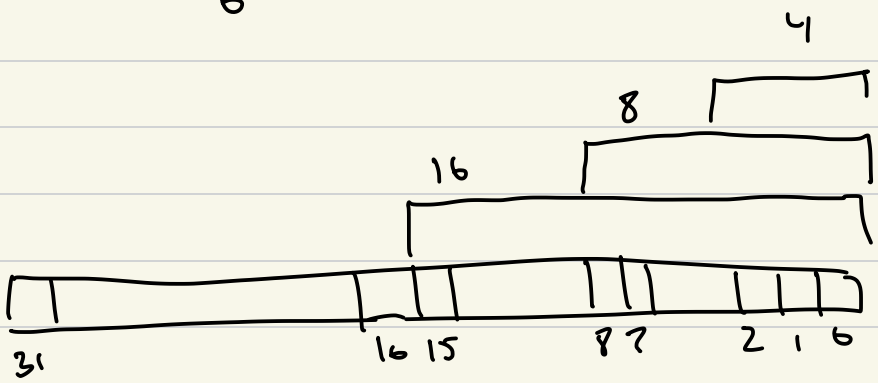
-w 4 (4 bits)

nt lang -b 16 -w 4 0xA3

0x3

(1+(2+3))

6



Bitwise Operators

~ not / invert i 0b1001 = 0b0110

& and 0b1100 & 0b0111 = 0b0100

| or 0b1100 | 0b1110 = 0b1110

^ xor 0b0101 ^ 0b0011 = 0b0110

Shifts

<< left shift

>> right shift

	x << 2	LSL logical shift left
unsigned	x >> 2	LSR logical shift right
signed int	x >> 2	ASR arithmetic shift right

ARMlang

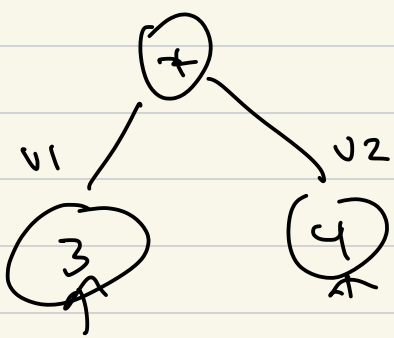
<< LSL

>> LSR

>- ASR

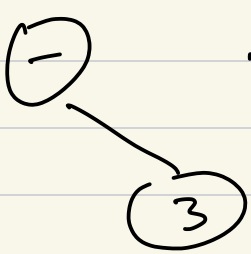
---

uint32_t value



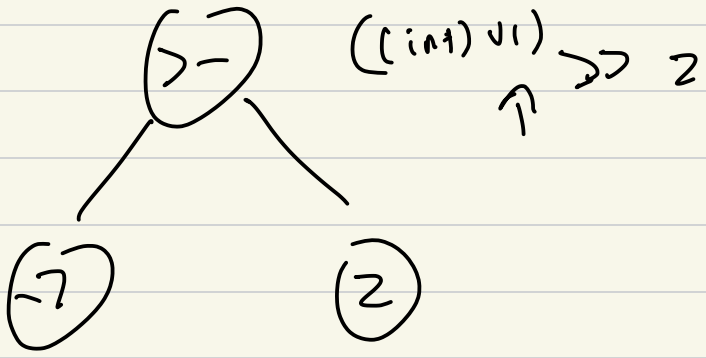
uint32_t

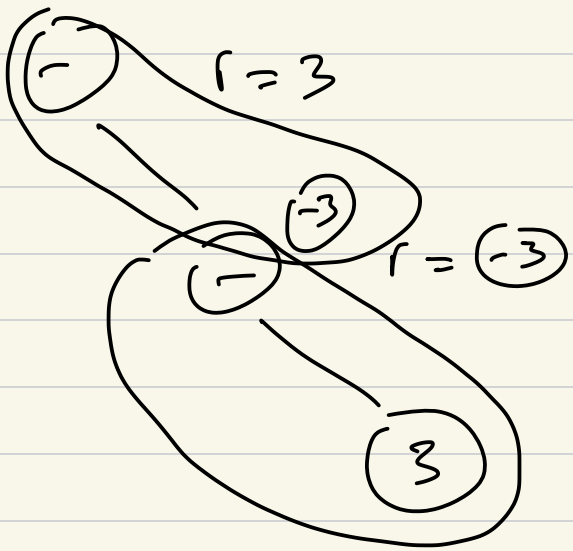
r = V1 + V2;



r = -((int)(V1))

-3





--x

---x