Powers of Two

2^{-1}	0.5
2^{-2}	0.25
2^{-3}	0.125
2^{-4}	0.0625
2^{-5}	0.03125

2^{0}	1
2^{1}	2
2^{2}	4
2^{3}	8
2^{4}	16
2^{5}	32
2^{6}	64

2^{7}	128
2^{8}	256
2^{9}	512
2^{10}	1024
2^{11}	2048
2^{12}	4096
2^{13}	8192

2^{10}	1 K
2^{20}	1 M
2^{30}	1 G

Hex table

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

ASCII table

Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex
nul	0	00	*space*	32	20	@	64	40	`	96	60
soh	1	01	!	33	21	A	65	41	a	97	61
stx	2	02	"	34	22	B	66	42	b	98	62
etx	3	03	#	35	23	C	67	43	c	99	63
eot	4	04	$	36	24	D	68	44	d	100	64
enq	5	05	%	37	25	E	69	45	e	101	65
ack	6	06	&	38	26	F	70	46	f	102	66
bel	7	07	'	39	27	G	71	47	g	103	67
bs	8	08	(	40	28	H	72	48	h	104	68
ht	9	09	)	41	29	I	73	49	i	105	69
lf	10	0A	*	42	2A	J	74	4A	j	106	6A
vt	11	0B	+	43	2B	K	75	4B	k	107	6B
ff	12	0C	,	44	2C	L	76	4C	l	108	6C
cr	13	0D	-	45	2D	M	77	4D	m	109	6D
so	14	0E	.	46	2E	N	78	4E	n	110	6E
si	15	0F	/	47	2F	O	79	4F	o	111	6F
dle	16	10	0	48	30	P	80	50	p	112	70
dc1	17	11	1	49	31	Q	81	51	q	113	71
dc2	18	12	2	50	32	R	82	52	r	114	72
dc3	19	13	3	51	33	S	83	53	s	115	73
dc4	20	14	4	52	34	T	84	54	t	116	74
nak	21	15	5	53	35	U	85	55	u	117	75
syn	22	16	6	54	36	V	86	56	v	118	76
etb	23	17	7	55	37	W	87	57	w	119	77
can	24	18	8	56	38	X	88	58	x	120	78
em	25	19	9	57	39	Y	89	59	y	121	79
sub	26	1A	:	58	3A	Z	90	5A	z	122	7A
esc	27	1B	;	59	3B	[	91	5B	{	123	7B
fs	28	1C	<	60	3C	\	92	5C	|	124	7C
gs	29	1D	=	61	3D	]	93	5D	}	125	7D
rs	30	1E	>	62	3E	^	94	5E	~	126	7E
us	31	1F	?	63	3F	_	95	5F	*Del*	127	7F

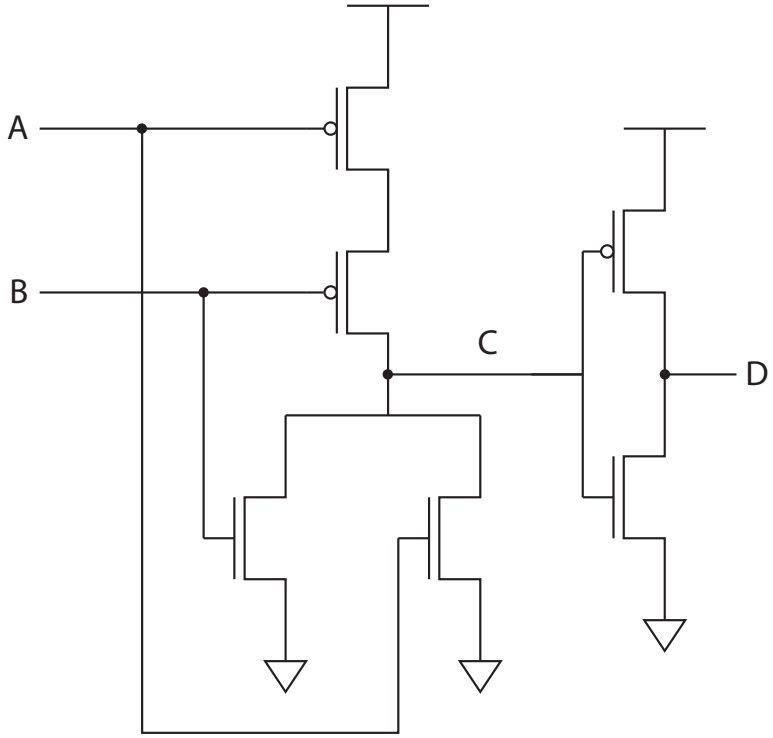
IEEE 32-bit floating point format

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
s	***exponent***								***fraction***																						

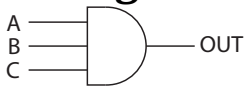
$$\mathbf{N = (-1)^{s} \times 1.}\mathit{fraction} \times 2^{\mathit{exponent}-127}, \text{ when } 1 \leq \mathit{exponent} \leq 254$$

$$\mathbf{N = (-1)^{s} \times 0.}\mathit{fraction} \times 2^{-126}, \text{ when } \mathit{exponent} = 0$$

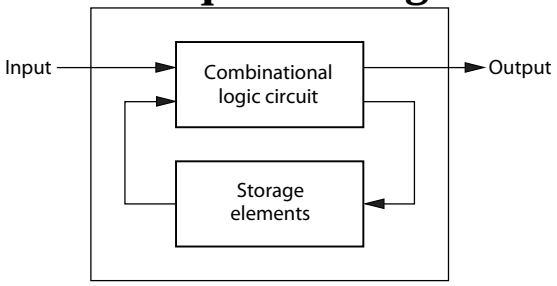
OR at transistor level



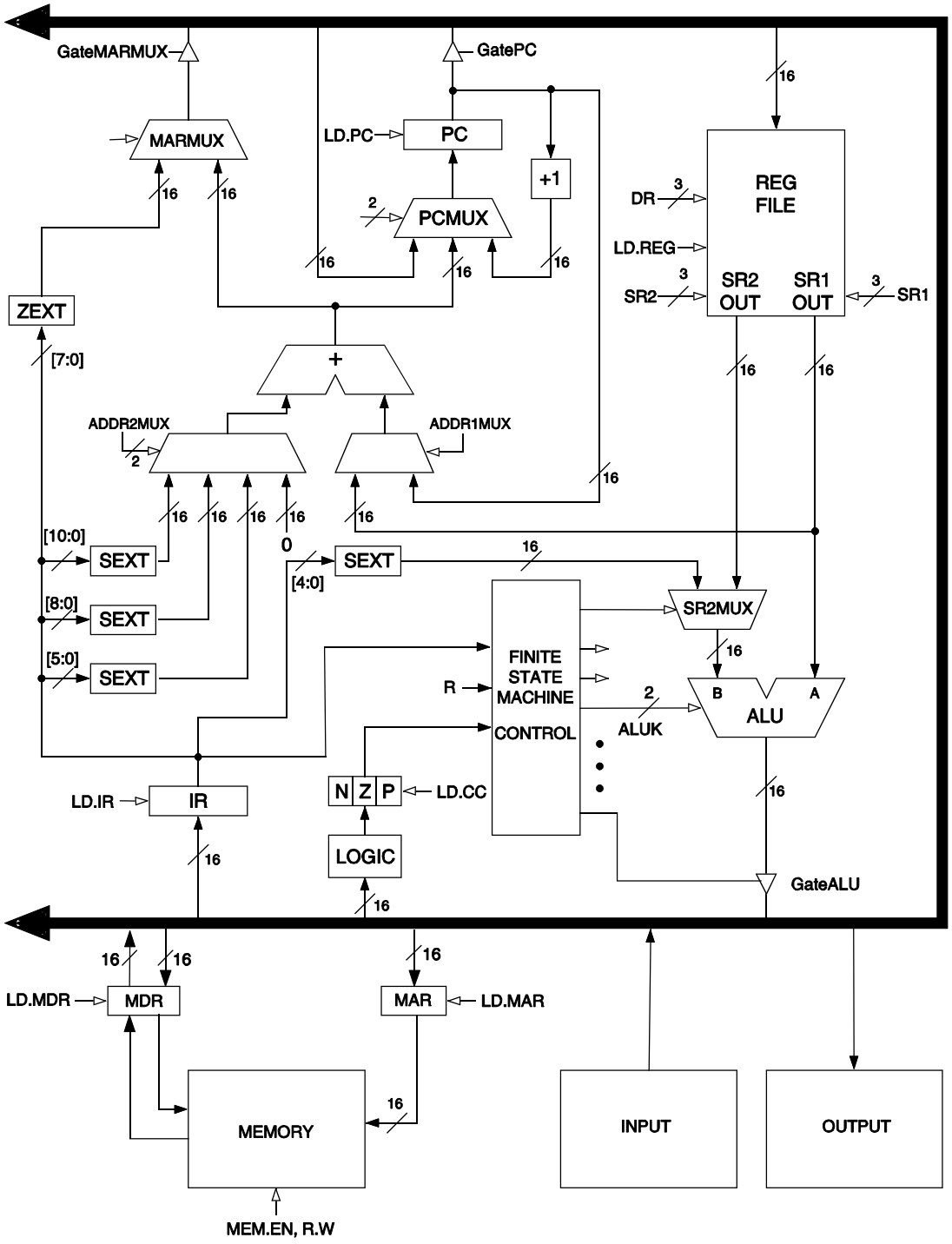
AND at gate level



Abstract sequential logic circuit



LC-3 Data Path



LC-3 instruction format

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADD	0	0	0	1	*DR*			*SR1*			0	0	0	*SR2*		
ADD	0	0	0	1	*DR*			*SR1*			1	*imm5*				
AND	0	1	0	1	*DR*			*SR1*			0	0	0	*SR2*		
AND	0	1	0	1	*DR*			*SR1*			1	*imm5*				
BR	0	0	0	0	n	z	p	*PCoffset9*								
JMP	1	1	0	0	0	0	0	*BaseR*			0	0	0	0	0	0
JSR	0	1	0	0	1	*PCoffset11*										
JSRR	0	1	0	0	0	0	0	*BaseR*			0	0	0	0	0	0
LD	0	0	1	0	*DR*			*PCoffset9*								
LDI	1	0	1	0	*DR*			*PCoffset9*								
LDR	0	1	1	0	*DR*			*BaseR*			*offset6*					
LEA	1	1	1	0	*DR*			*PCoffset9*								
NOT	1	0	0	1	*DR*			*SR*			1	1	1	1	1	1
RET	1	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0
RTI	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ST	0	0	1	1	*SR*			*PCoffset9*								
STI	1	0	1	1	*SR*			*PCoffset9*								
STR	0	1	1	1	*SR*			*BaseR*			*offset6*					
TRAP	1	1	1	1	0	0	0	0	*trapvect8*							

All instructions with a *DR* modify the condition codes.
All offsets are sign extended.
The PC is incremented *before* the offset is added.
JSR, JSRR, and TRAP save the old PC in R7.
RET is the same as JMP R7.
RTI is too hard to use in ECE 109 and CSCI 255.
BR has many flavors that specify how bits 11 to 9 are set:

NOP, BRp, BRz, BRzp, BRn, BRnp, BRnz, BRnzp, BR

Useful LC-3 pseudo-ops

`.ORIG`	Starting address for LC-3 program. Must be the first line of program.
`.FILL`	Allocates an initialized word of memory.
`.BLKW`	Allocates an uninitialized block of memory.
`.STRINGZ`	Allocates a zero-terminated string in memory.
`.END`	Must be the last line of program.

LC-3Trap Service Routines

`x20`	`GETC`	Read a character into `R0`. Do not echo the character.
`x21`	`OUT`	Write a single character stored in `R0`.
`x22`	`PUTS`	Write a zero-terminated string stored at address `R0`.
`x23`	`IN`	`PUTS` followed by `GETC`, but with echoing of the input character.
`x24`	`PUTSP`	`PUTS`, but with two characters stored in each word.
`x25`	`HALT`	Halt

LC/3 Device Registers

`KBSR` Keyboard Status Register	`xFE00`	`KBSR[15]` is one when keyboard has a new character
`KBDR` Keyboard Data Register	`xFE02`	`KBDR[7:0]` is last character typed on keyboard
`DSR` Display Status Register	`xFE04`	`DSR[15]` is one when display can accept a new character
`DDR` Display Data Register	`xFE06`	`DSR[7:0]` is the character to be displayed on screen

"Other" LC-3 Registers

`PC` Program Counter	The address of the *next* instruction to be executed. Pushed to supervisor stack on interrupt. Popped from supervisor stack on `RTI` instruction.
`PSR` Processor Status Register	`PSR[15]` is one if in user mode, `PSR[10:8]` is the priority level, `PSR[2:0]` is `NZP` bits. Pushed to supervisor stack on interrupt. Popped from supervisor stack on `RTI` instruction.
`Saved.USP` Saved User Stack Pointer	On interrupt, `R6` is stored here. On `RTI`, `R6` is set from `Saved.USP` when *new* `PSR[15]` is 1.
`Saved.SSP` Saved Supervisor Stack Pointer	On interrupt, R6 is set from `Saved.SSP`.

LC/3 Stack Example