CHIP-8 Instruction Set

MISC

CLS	00E0	clear screen	
NOP	0000	no operation	
STOP	F000	returns program to debugger	

ADDITION

ADD	7XNN	V[X] = V[X] + NN	if carry then V[F] = 1
ADD	8XY4	V[X] = V[X} + V[Y]	if carry then V[F] = 1

SUBTRACT

SUB	8XY5	V[X] = V[X] - V[Y]	if V[X] > V[Y] then V[F] = 1
SUB	8XY7	V[X] = V[Y] - V[X]	if V[Y] > V[X] then V[F] = 1

MULTIPLY

SHL	8XYE	V[X] = V[Y], V[X] <<1	V[F] = msb	**

DIVIDE

SHR	8XY6	V[X] = V[Y], V[X] >> 1	V[F] = lsb	*

LOGIC

AND	8XY2	V[X] = V[X] & V[Y]
OR	8XY1	V[X] = V[X] | V[Y]
XOR	8XY3	V[X] = V[X] ^ V[Y]
NOT		V[Y] = 0xFF, then XOR

BCD

ANNN	Point the Index register to clear memory BEFORE using the next instruction	
BCD	FX33	store BCD equivalent of V[X] into Memory[I], M[I+1], M[I+2]
GET	FX65	load V[0] thru V[X] with Memory[I]. I=I+X+1 when finished
SET	FX29	point register I at font-number in V[X]

DISPLAY

SET	FX29	point register I at font-number in V[X]	

DRW	DXYN	I = sprite top, Draw N lines @ screen [X],[Y]
		if collision V[F] = 1 else V[F] = 0
			***

BCD	FX33	store BCD equivalent of V[X] into Memory[I], M[I+1], M[I+2]	
GET	FX65	load V[0] thru V[X] with Memory[I]. I=I+X+1 when finished	

KEYPAD

WAIT	FX0A	wait for key press, V[X] = Key
SKIP	EX9E	skip if Key == V[X]
SKIP	EXA1	skip if Key != V[X]

TIMERS

GET	FX07	V[X] = DelayTimer	each count is approx 20mS
SET	FX15	DelayTimer = V[X]	
SET	FX18	SoundTimer = V[X]	

RANDOM

RND	CXNN	V[X] = random number & 0xNN

MEMORY

BCD	FX33	store BCD equivalent of V[X] into Memory[I], M[I+1], M[I+2]
SET	FX55	store V[0] thru V[X] starting at Memory[I]. I=I+X+1 when finished
GET	FX65	load V[0] thru V[X] with Memory[I]. I=I+X+1 when finished

REGISTERS V[0] ..V[F]

COPY	8XY0	V[X] = V[Y]
SET	6XNN	V[X] = NN

INDEX REGISTER

SET	ANNN	I = NNN	
SET	FX29	point register I at font-number in V[X]	
SET	FX1E	I = I + V[X]	if carry then V[F] = 1
SET	6XNN	V[X] = NN	

SKIP

SKIP	3XNN	V[X] == NN
SKIP	4XNN	V[X] != NN
SKIP	5XY0	V[X] == V[Y]
SKIP	9XY0	V[X] != V[Y]
SKIP	EX9E	skip if Key == V[X]
SKIP	EXA1	skip if Key != V[X]
WAIT	FX0A	wait for key press, V[X] = Key

JUMP

JMP	1NNN	jump to address NNN
JMP	BNNN	Address = NNN + contents of V[0]

SUBROUTINES		**(**these instructions involve the stack and SP (stack pointer))
JSR	2NNN	jump to subroutine at NNN
RTS	00EE	return from subroutine

Notes

*	lsb = least significant bit
**	msb = most significant bit
***	Sprites bits that are SET =1 toggle each the corresponding SCREEN bits Using DXYN twice erases the sprite pattern from the screen

LINGIB
https://www.instructables.com/member/lingib/instructables/