GameBoy CPU InstructionSet Sheet (GCISheet)

QuickJump Navigator:								
ADC A,n	CCF	INC n	LD A <u>, n</u>	LD [HL+],A	LD [n],SP	<u>NOP</u>	RL n	SLA n
ADD A,n	CP n	INC nn	LD n,A	LD [HL-],A	LDD A, [HL]	OR n	IDTC n	SRA n
ADD HL, n	CPL	<u>JP n</u>	LD A, [C]	LD [HLI],A	LDD [HL],A	POP nn	libb n	SRL n
ADD SP, n	DAA	JP cc,n	LD A, [HL+]	LD [HLD],A	LDH [n],A	PUSH nn		STOP
AND n	DEC n	JP [HL]	LD A, [HL-]	LD r1, r2	LDH A, [n]	RES b,r	RST n	CIID
BIT b,r	DEC nn	JR n	LD A, [HLI]	LD n,nn	LDHL SP,n	RET	SBC A,n	CMAD n
CALL n	DI	JR cc, n	LD A, [HLD]	LD HL, [SP+n]	LDI A, [HL]	RET cc		XOR n
CALL cc, n	EI	HALT	LD [C],A	LD SP,HL	LDI [HL],A	RETI	SET b,r	AUK II

ADC A, n - Add n + Carry flag to A.

n = A,B,C,D,E,H,L,(HL),#

Flags affected:

Z - Set if result is zero.

N - Reset.

H - Set if carry from bit 3.

C - Set if carry from bit 7.

<u>Top</u>

ADD A,n - Add n to A.

```
n = A,B,C,D,E,H,L,(HL),#
        Flags affected:
                Z - Set if result is zero.
                N - Reset.
                H - Set if carry from bit 3.
                C - Set if carry from bit 7.
<u>Top</u>
ADD HL,n - Add n to HL.
        n = BC,DE,HL
        Flags affected:
                Z - Not affected
                N - Reset.
                H - Set if carry from bit 11.
```

```
C - Set if carry from bit 15.

Top

ADD SP,n - Add n to Stack Pointer (SP).

n = one byte signed immediate value
```

Flags affected:

Z - Reset.

N - Reset.

H - Set or reset according to operation.

C - Set or reset according to operation.

<u>Top</u>

AND n - Logically AND n with A, result in A.

n = A,B,C,D,E,H,L,(HL),#

```
Flags affected:
                Z - Set if result is zero.
                N - Reset.
                H - Set.
               C - Reset.
<u>Top</u>
BIT b,r - Test bit b in register r.
        b = 0-7, r = A,B,C,D,E,H,L,(HL)
        Flags affected:
                Z - Set if bit b of register r is 0.
                N - Reset.
                H - Set.
```

```
C - Not affected.
<u>Top</u>
CALL n
        - Push address of next instruction onto
                stack and then jump to address n.
        Flags affected:
                None
<u>Top</u>
CALL cc,n - Call address n if following condition
                is true:
        cc = NZ, Call if Z flag is reset.
        cc = Z, Call if Z flag is set.
        cc = NC, Call if C flag is reset.
        cc = C, Call if C flag is set.
```

```
Flags affected:
                None
<u>Top</u>
    - Complement carry flag.
CCF
        If C flag is set then reset it.
        If C flag is reset then set it.
        Flags affected:
                Z - Not affected.
                N - Reset.
                H - Reset.
                C - Complemented.
<u>Top</u>
```

```
CP n
       - Compare A with n.
        This is basically an A - n subtraction
        instruction but the results are thrown away.
        n = A,B,C,D,E,H,L,(HL),#
        Flags affected:
                Z - Set if result is zero. (Set if A = n)
                N - Set.
                H - Set if no borrow from bit 4.
                C - Set for no borrow. (Set if A < n.)
<u>Top</u>
CPL
        - Complement A register. (Flip all bits.)
        Flags affected:
```

```
Z - Not affected.
                N - Set.
                H - Set.
                C - Not affected.
<u>Top</u>
         - Decimal adjust register A.
DAA
        This instruction adjusts register A so that the
        correct representation of Binary Coded Decimal
        (BCD) is obtained.
        Flags affected:
                Z - Set if register A is zero.
                N - Not affected.
                H - Reset.
```

```
C - Set of reset according to operation.
<u>Top</u>
DEC n
      - Decrement register n.
        n = A,B,C,D,E,H,L,(HL)
        Flags affected:
                Z - Set if result is zero.
                N - Set.
                H - Set if no borrow from bit 4.
                C - Not affected.
<u>Top</u>
DEC nn - Decrement register nn.
        nn = BC, DE, HL, SP
```

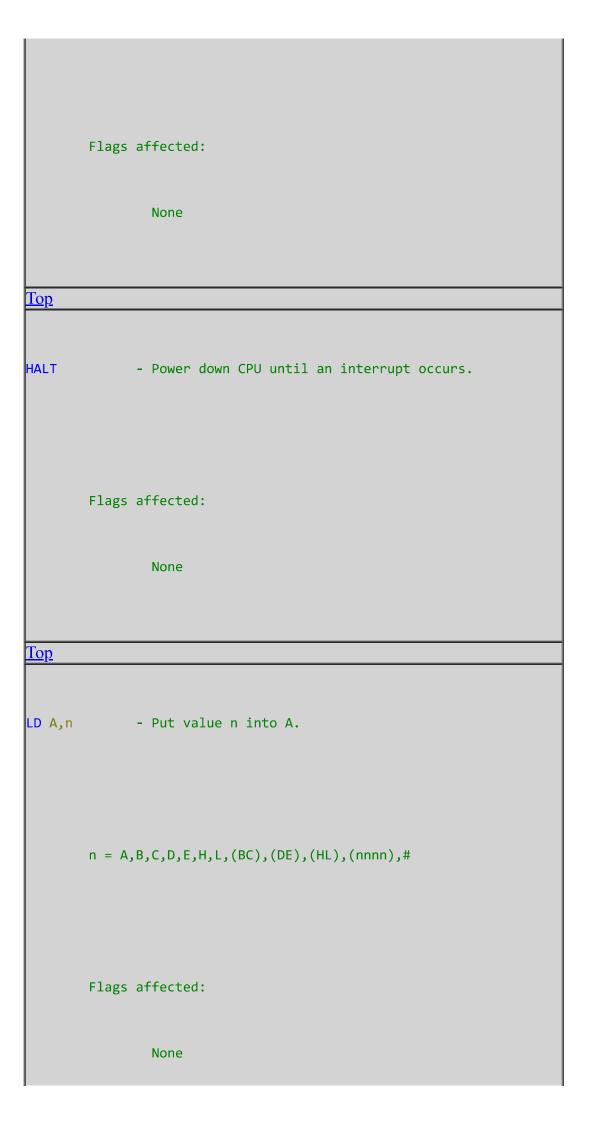
	Flags affected:
	None
<u>Top</u>	
<u>10p</u>	
DI	- Disable interrupts.
	Flags affected:
	None
<u>Top</u>	
EI	- Enable interrupts.
	This instruction enables the interrupts but not immediately.
	Interrupts are enabled after the instruction after EI is
	executed.

	Flags affected:
	None
Top	
INC n	- Increment register n.
	n = A,B,C,D,E,H,L,(HL)
	Flags affected:
	Z - Set if result is zero.
	N - Reset.
	H - Set if carry from bit 3.
	C - Not affected.
<u>Top</u>	
INC nn	- Increment register nn.

```
n = BC,DE,HL,SP
        Flags affected:
                None
<u>Top</u>
JP n
       - Jump to address n.
        n = two byte immediate value. (LSByte first)
        Flags affected:
                None
<u>Top</u>
JP cc,n - Jump to address n if following condition
                is true:
```

```
n = two byte immediate value. (LSByte first.)
        cc = NZ, Jump if Z flag is reset.
        cc = Z, Jump if Z flag is set.
        cc = NC, Jump if C flag is reset.
        cc = C, Jump if C flag is set.
        Flags affected:
                None
<u>Top</u>
JP [HL] - Jump to address contained in HL.
        Flags affected:
                None
<u>Top</u>
```

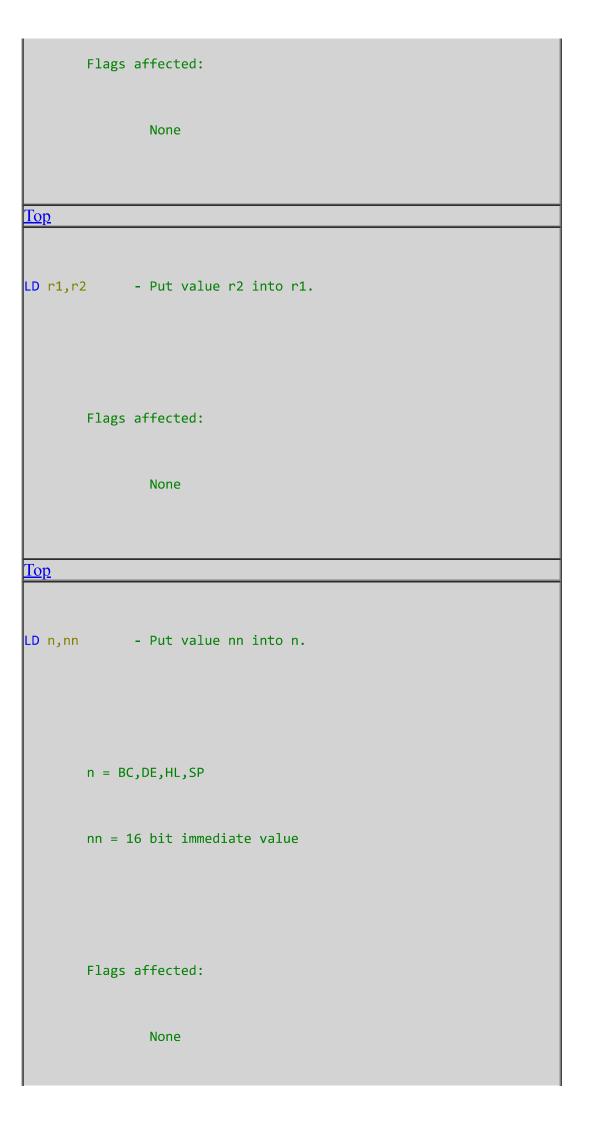
```
- Add n to current address and jump to it.
JR n
        n = one byte signed immediate value.
        Flags affected:
                None
<u>Top</u>
JR cc,n - If following condition is true then
                add n to current address and jump to it:
        n = one byte signed immediate value
        cc = NZ, Jump if Z flag is reset.
        cc = Z, Jump if Z flag is set.
        cc = NC, Jump if C flag is reset.
        cc = C, Jump if C flag is set.
```



```
<u>Top</u>
LD n,A - Put value A into n.
        n = A,B,C,D,E,H,L,(BC,(DE),(HL),(nnnn)
        Flags affected:
                None
<u>Top</u>
LD A,[C] - Put value at address $FF00 + register C into A.
        Flags affected:
                None
<u>Top</u>
LD A,[HL+] - Same as LD A,[HLI].
```

```
<u>Top</u>
LD A,[HL-] - Same as LD A,[HLD].
<u>Top</u>
LD A,[HLI] - Put value at address HL into A. Increment HL.
        Flags affected:
                None
<u>Top</u>
LD A,[HLD] - Put value at address HL into A. Decrement HL.
        Flags affected:
                None
<u>Top</u>
LD [C],A - Put A into address $FF00 + register C.
```

```
Flags affected:
                 None
<u>Top</u>
LD [HL+],A - Same as LD [HLI],A.
<u>Top</u>
LD [HL-],A - Same as LD [HLD],A.
<u>Top</u>
LD [HLI],A - Put A into memory address HL. Increment HL.
        Flags affected:
                 None
<u>Top</u>
LD [HLD],A - Put A into memory address HL. Decrement HL.
```



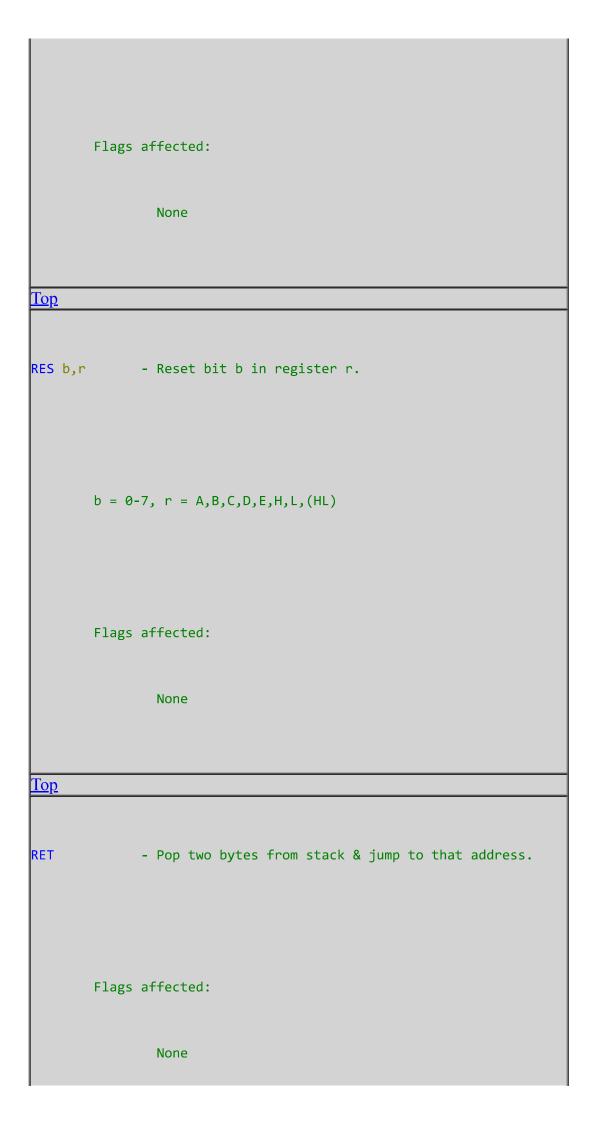
```
<u>Top</u>
LD HL,[SP+n] - Put SP + n into HL.
        n = one byte signed immediate value
        Flags affected:
                Z - Reset.
                N - Reset.
                H - Set or reset according to operation.
                C - Set or reset according to operation.
<u>Top</u>
LD SP, HL - Put HL into Stack Pointer (SP).
        Flags affected:
                None
```

```
<u>Top</u>
LD [n],SP - Put Stack Pointer (SP) at address n.
        n = two byte immediate address
        Flags affected:
                 None
<u>Top</u>
LDD A,[HL] - Same as LD A,[HLD].
<u>Top</u>
LDD [HL],A - Same as LD [HLD],A.
<u>Top</u>
LDH [n],A - Put A into memory address $FF00 + n.
```

```
n = one byte immediate value
        Flags affected:
                None
<u>Top</u>
LDH A,[n] - Put memory address $FF00 + n into A.
        n = one byte immediate value
        Flags affected:
                None
<u>Top</u>
LDHL SP,n - Same as LD HL,[SP+n]
<u>Top</u>
LDI A,[HL] - Same as LD A,[HLI].
```

```
<u>Top</u>
LDI [HL],A - Same as LD [HLI],A.
<u>Top</u>
NOP
       - No operation.
        Flags affected:
                None
<u>Top</u>
OR n - Logical OR n with register A, result in A.
        n = A,B,C,D,E,H,L,(HL),#
        Flags affected:
                Z - Set if result is zero.
                N - Reset.
```

```
H - Reset.
                C - Reset.
<u>Top</u>
POP nn - Pop two bytes off stack into register pair nn.
                Increment Stack Pointer (SP) twice.
        nn = AF,BC,DE,HL
        Flags affected:
                None
<u>Top</u>
PUSH nn - Push register pair nn onto stack.
                Decrement Stack Pointer (SP) twice.
        nn = AF,BC,DE,HL
```



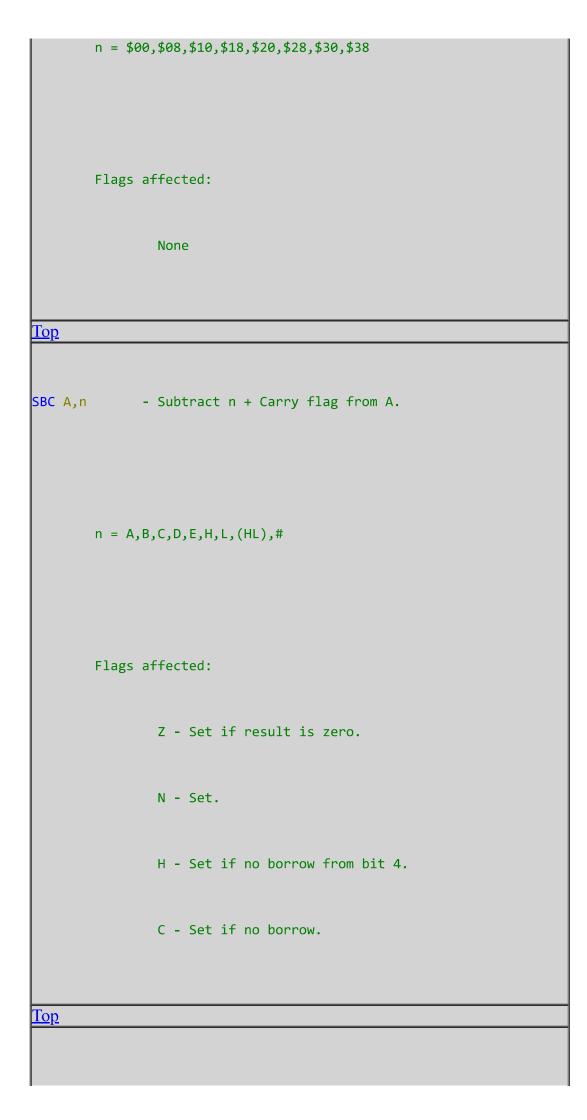
```
<u>Top</u>
RET cc - Return if following condition is true:
        cc = NZ, Return if Z flag is reset.
        cc = Z, Return if Z flag is set.
        cc = NC, Return if C flag is reset.
        cc = C, Return if C flag is set.
        Flags affected:
                None
<u>Top</u>
              - Pop two bytes from stack & jump to that address
RETI
                then enable interrupts.
        Flags affected:
```

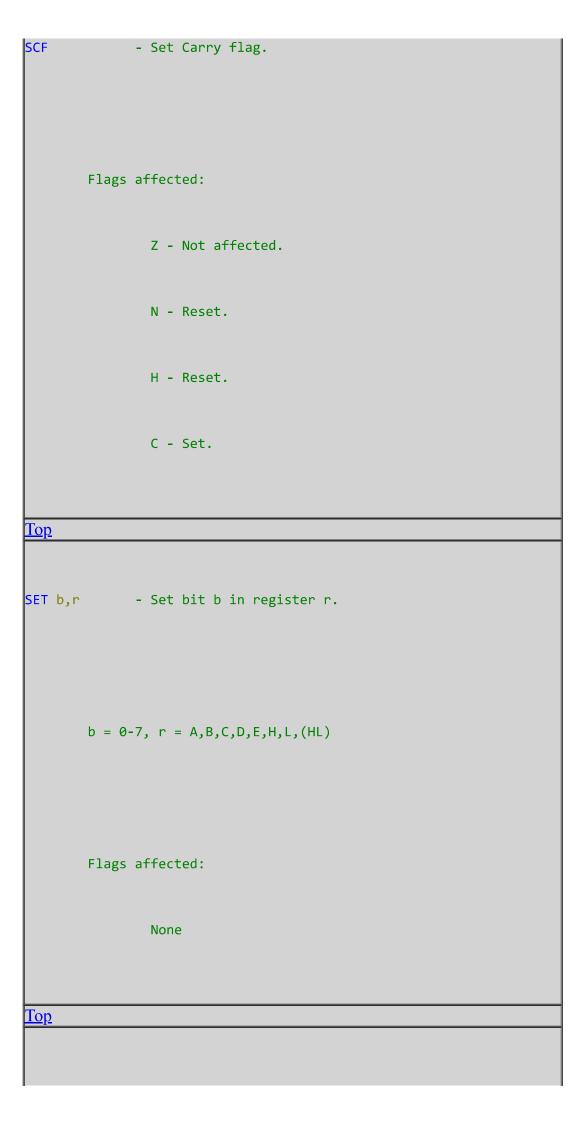
	None
<u>Top</u>	
RL n	- Rotate n left through Carry flag.
	n = A,B,C,D,E,H,L,(HL)
	Flags affected:
	Z - Set if result is zero.
	N - Reset.
	H - Reset.
	C - Contains old bit 7 data.
<u>Top</u>	
RLC n	- Rotate n left. Old bit 7 to Carry flag.

n = A,B,C,D,E,H,L,(HL)

Flags affected: Z - Set if result is zero. N - Reset. H - Reset. C - Contains old bit 7 data. <u>Top</u> RR n - Rotate n right through Carry flag. n = A,B,C,D,E,H,L,(HL)Flags affected: Z - Set if result is zero. N - Reset. H - Reset.

```
C - Contains old bit 0 data.
<u>Top</u>
RRC n
        - Rotate n right. Old bit 0 to Carry flag.
        n = A,B,C,D,E,H,L,(HL)
        Flags affected:
                Z - Set if result is zero.
                N - Reset.
                H - Reset.
                C - Contains old bit 0 data.
<u>Top</u>
              - Push present address onto stack.
RST n
                Jump to address $0000 + n.
```



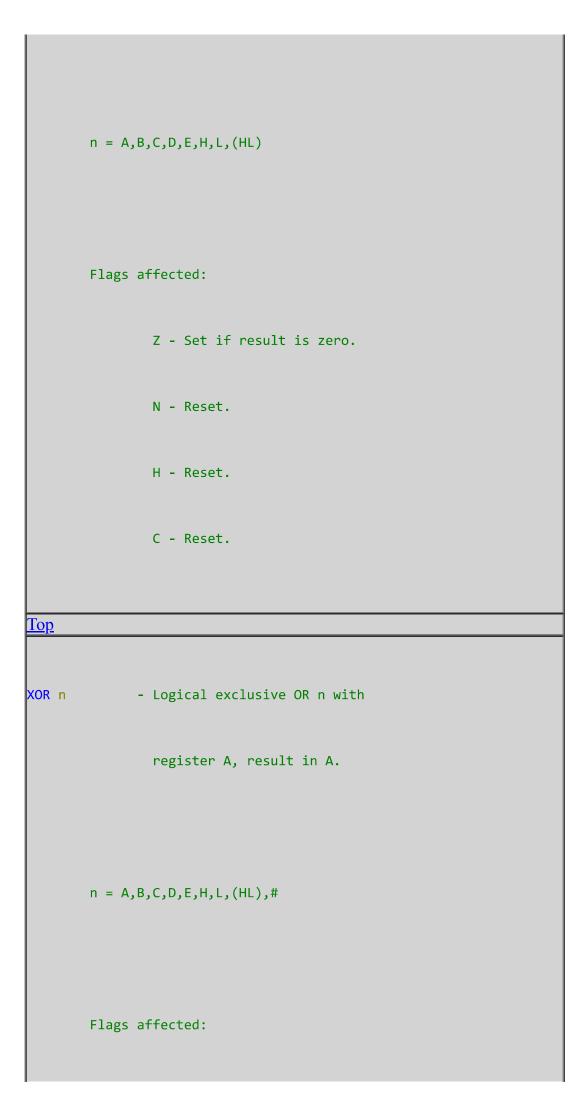


```
- Shift n left into Carry. LSBit of n set to 0.
SLA n
        n = A,B,C,D,E,H,L,(HL)
        Flags affected:
                Z - Set if result is zero.
                N - Reset.
                H - Reset.
                C - Contains old bit 7 data.
<u>Top</u>
SRA n
            - Shift n right into Carry. MSBit doesn't change.
        n = A,B,C,D,E,H,L,(HL)
        Flags affected:
                Z - Set if result is zero.
```

N - Reset. H - Reset. C - Contains old bit 0 data. <u>Top</u> SRL n - Shift n right into Carry. MSBit of n set to 0. n = A,B,C,D,E,H,L,(HL)Flags affected: Z - Set if result is zero. N - Reset. H - Reset. C - Contains old bit 0 data. <u>Top</u> STOP - ???

Flags affected: ? <u>Top</u> SUB n - Subtract n from A. n = A,B,C,D,E,H,L,(HL),#Flags affected: Z - Set if result is zero. N - Set. H - Set if no borrow from bit 4. C - Set if no borrow. <u>Top</u>

SWAP n - Swap upper & lower bits of n.



	Z - Set if result is zero.
	N - Reset.
	H - Reset.
	C - Reset.
<u>Top</u>	

All material on this page is Copyright (c) 1999 by col_deamon. All rights reserved.

Last updated: 11.25.99 09:46