GameBoy CPU InstructionSet Sheet (GCISheet)

QuickJump Navigator:


```
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.
```

Top

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.

Top

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.

Top

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.
```

Top

```
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.
```

Top

```
CALL n        - Push address of next instruction onto
                stack and then jump to address n.



        Flags affected:

                None
```

Top

```
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

Top

CCF - Complement carry flag.

```
        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.
```

Top

```
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.)

Top

CPL - Complement A register. (Flip all bits.)

Flags affected:

Z - Not affected.

N - Set.

H - Set.

C - Not affected.

Top

DAA - Decimal adjust register A.

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.

Top

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.
```

Top

```
DEC nn        - Decrement register nn.



        nn = BC,DE,HL,SP



        Flags affected:

                None
```

Top

```
DI            - Disable interrupts.



        Flags affected:

                None
```

Top

```
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.

Top

INC nn - Increment register nn.

n = BC,DE,HL,SP

Flags affected:

None

Top

JP n - Jump to address n.

n = two byte immediate value. (LSByte first)

Flags affected:

None

Top

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
```

Top

```
JP [HL]       - Jump to address contained in HL.



        Flags affected:

                None
```

Top

```
JR n          - Add n to current address and jump to it.
```

```
        n = one byte signed immediate value.



        Flags affected:

                None
```

Top

```
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.



        Flags affected:

                None
```

Top

```
HALT          - Power down CPU until an interrupt occurs.



        Flags affected:

                None
```

Top

```
LD A,n        - Put value n into A.



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



        Flags affected:

                None
```

Top

```
LD n,A        - Put value A into n.



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



        Flags affected:
```

None

Top

LD A,[C] - Put value at address $FF00 + register C into A.

Flags affected:

None

Top

LD A,[HL+] - Same as LD A,[HLI].

Top

LD A,[HL-] - Same as LD A,[HLD].

Top

```
LD A,[HLI]    - Put value at address HL into A. Increment HL.



        Flags affected:

                None
```

Top

```
LD A,[HLD]    - Put value at address HL into A. Decrement HL.



        Flags affected:

                None
```

Top

```
LD [C],A      - Put A into address $FF00 + register C.
```

Flags affected:

None

Top

LD [HL+],A - Same as LD [HLI],A.

Top

LD [HL-],A - Same as LD [HLD],A.

Top

LD [HLI],A - Put A into memory address HL. Increment HL.

Flags affected:

None

Top

LD [HLD],A - Put A into memory address HL. Decrement HL.

Flags affected:

None

Top

LD r1,r2 - Put value r2 into r1.

Flags affected:

None

Top

LD n,nn - Put value nn into n.

n = BC,DE,HL,SP

nn = 16 bit immediate value

Flags affected:

None

Top

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.

Top

LD SP,HL - Put HL into Stack Pointer (SP).

Flags affected:

None

Top

```
LD [n],SP     - Put Stack Pointer (SP) at address n.


        n = two byte immediate address


        Flags affected:

                None
```

Top

```
LDD A,[HL]    - Same as LD A,[HLD].
```

Top

```
LDD [HL],A    - Same as LD [HLD],A.
```

Top

```
LDH [n],A     - Put A into memory address $FF00 + n.



        n = one byte immediate value



        Flags affected:

                None
```

Top

```
LDH A,[n]     - Put memory address $FF00 + n into A.



        n = one byte immediate value
```

```
        Flags affected:

                None
```

Top

```
LDHL SP,n      - Same as LD HL,[SP+n]
```

Top

```
LDI A,[HL]     - Same as LD A,[HLI].
```

Top

```
LDI [HL],A     - Same as LD [HLI],A.
```

Top

```
NOP            - No operation.
```

Flags affected:

None

Top

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.

Top

POP nn - Pop two bytes off stack into register pair nn.

Increment Stack Pointer (SP) twice.

nn = AF,BC,DE,HL

Flags affected:

None

Top

PUSH nn - Push register pair nn onto stack.

Decrement Stack Pointer (SP) twice.

nn = AF,BC,DE,HL

Flags affected:

None

Top

RES b,r - Reset bit b in register r.

b = 0-7, r = A,B,C,D,E,H,L,(HL)

Flags affected:

None

Top

RET - Pop two bytes from stack & jump to that address.

Flags affected:

None

Top

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

Top

RETI - Pop two bytes from stack & jump to that address
then enable interrupts.

Flags affected:

None

Top

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.

Top

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.

Top

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.

Top

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.

Top

RST n - Push present address onto stack.

Jump to address $0000 + n.

n = $00,$08,$10,$18,$20,$28,$30,$38

Flags affected:

None

Top

SBC A,n - Subtract n + Carry flag 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.

Top

SCF - Set Carry flag.

Flags affected:

Z - Not affected.

N - Reset.

H - Reset.

C - Set.

Top

SET b,r - Set bit b in register r.

b = 0-7, r = A,B,C,D,E,H,L,(HL)

Flags affected:

None

Top

SLA n - Shift n left into Carry. LSBit 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 7 data.

Top

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.

Top

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.

Top

STOP - ???

Flags affected:

?

Top

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.

Top

SWAP n - Swap upper & lower bits of n.

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

Flags affected:

Z - Set if result is zero.

N - Reset.

H - Reset.

C - Reset.

Top

XOR n - Logical exclusive 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.

Top

Last updated: *11.25.99 09:46*