Gameboy CPU (LR35902) instruction set

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	NOP 1 4 ----	LD BC,d16 3 12 ----	LD (BC),A 1 8 ----	INC BC 1 8 ----	INC B 1 4 Z0H-	DEC B 1 4 Z1H-	LD B,d8 2 8 ----	RLCA 1 4 000C	LD (a16),SP 3 20 ----	ADD HL,BC 1 8 -0HC	LD A,(BC) 1 8 ----	DEC BC 1 8 ----	INC C 1 4 Z0H-	DEC C 1 4 Z1H-	LD C,d8 2 8 ----	RRCA 1 4 000C
1x	STOP 0 2 4 ----	LD DE,d16 3 12 ----	LD (DE),A 1 8 ----	INC DE 1 8 ----	INC D 1 4 Z0H-	DEC D 1 4 Z1H-	LD D,d8 2 8 ----	RLA 1 4 000C	JR r8 2 12 ----	ADD HL,DE 1 8 -0HC	LD A,(DE) 1 8 ----	DEC DE 1 8 ----	INC E 1 4 Z0H-	DEC E 1 4 Z1H-	LD E,d8 2 8 ----	RRA 1 4 000C
2x	JR NZ,r8 2 12/8 ----	LD HL,d16 3 12 ----	LD (HL+),A 1 8 ----	INC HL 1 8 ----	INC H 1 4 Z0H-	DEC H 1 4 Z1H-	LD H,d8 2 8 ----	DAA 1 4 Z-0C	JR Z,r8 2 12/8 ----	ADD HL,HL 1 8 -0HC	LD A,(HL+) 1 8 ----	DEC HL 1 8 ----	INC L 1 4 Z0H-	DEC L 1 4 Z1H-	LD L,d8 2 8 ----	CPL 1 4 -11-
3x	JR NC,r8 2 12/8 ----	LD SP,d16 3 12 ----	LD (HL-),A 1 8 ----	INC SP 1 8 ----	INC (HL) 1 12 Z0H-	DEC (HL) 1 12 Z1H-	LD (HL),d8 2 12 ----	SCF 1 4 -001	JR C,r8 2 12/8 ----	ADD HL,SP 1 8 -0HC	LD A,(HL-) 1 8 ----	DEC SP 1 8 ----	INC A 1 4 Z0H-	DEC A 1 4 Z1H-	LD A,d8 2 8 ----	CCF 1 4 -00C
4x	LD B,B 1 4 ----	LD B,C 1 4 ----	LD B,D 1 4 ----	LD B,E 1 4 ----	LD B,H 1 4 ----	LD B,L 1 4 ----	LD B,(HL) 1 8 ----	LD B,A 1 4 ----	LD C,B 1 4 ----	LD C,C 1 4 ----	LD C,D 1 4 ----	LD C,E 1 4 ----	LD C,H 1 4 ----	LD C,L 1 4 ----	LD C,(HL) 1 8 ----	LD C,A 1 4 ----
5x	LD D,B 1 4 ----	LD D,C 1 4 ----	LD D,D 1 4 ----	LD D,E 1 4 ----	LD D,H 1 4 ----	LD D,L 1 4 ----	LD D,(HL) 1 8 ----	LD D,A 1 4 ----	LD E,B 1 4 ----	LD E,C 1 4 ----	LD E,D 1 4 ----	LD E,E 1 4 ----	LD E,H 1 4 ----	LD E,L 1 4 ----	LD E,(HL) 1 8 ----	LD E,A 1 4 ----
6x	LD H,B 1 4 ----	LD H,C 1 4 ----	LD H,D 1 4 ----	LD H,E 1 4 ----	LD H,H 1 4 ----	LD H,L 1 4 ----	LD H,(HL) 1 8 ----	LD H,A 1 4 ----	LD L,B 1 4 ----	LD L,C 1 4 ----	LD L,D 1 4 ----	LD L,E 1 4 ----	LD L,H 1 4 ----	LD L,L 1 4 ----	LD L,(HL) 1 8 ----	LD L,A 1 4 ----
7x	LD (HL),B 1 8 ----	LD (HL),C 1 8 ----	LD (HL),D 1 8 ----	LD (HL),E 1 8 ----	LD (HL),H 1 8 ----	LD (HL),L 1 8 ----	HALT 1 4 ----	LD (HL),A 1 8 ----	LD A,B 1 4 ----	LD A,C 1 4 ----	LD A,D 1 4 ----	LD A,E 1 4 ----	LD A,H 1 4 ----	LD A,L 1 4 ----	LD A,(HL) 1 8 ----	LD A,A 1 4 ----
8x	ADD A,B 1 4 Z0HC	ADD A,C 1 4 Z0HC	ADD A,D 1 4 Z0HC	ADD A,E 1 4 Z0HC	ADD A,H 1 4 Z0HC	ADD A,L 1 4 Z0HC	ADD A,(HL) 1 8 Z0HC	ADD A,A 1 4 Z0HC	ADC A,B 1 4 Z0HC	ADC A,C 1 4 Z0HC	ADC A,D 1 4 Z0HC	ADC A,E 1 4 Z0HC	ADC A,H 1 4 Z0HC	ADC A,L 1 4 Z0HC	ADC A,(HL) 1 8 Z0HC	ADC A,A 1 4 Z0HC
9x	SUB B 1 4 Z1HC	SUB C 1 4 Z1HC	SUB D 1 4 Z1HC	SUB E 1 4 Z1HC	SUB H 1 4 Z1HC	SUB L 1 4 Z1HC	SUB (HL) 1 8 Z1HC	SUB A 1 4 Z1HC	SBC A,B 1 4 Z1HC	SBC A,C 1 4 Z1HC	SBC A,D 1 4 Z1HC	SBC A,E 1 4 Z1HC	SBC A,H 1 4 Z1HC	SBC A,L 1 4 Z1HC	SBC A,(HL) 1 8 Z1HC	SBC A,A 1 4 Z1HC
Ax	AND B 1 4 Z010	AND C 1 4 Z010	AND D 1 4 Z010	AND E 1 4 Z010	AND H 1 4 Z010	AND L 1 4 Z010	AND (HL) 1 8 Z010	AND A 1 4 Z010	XOR B 1 4 Z000	XOR C 1 4 Z000	XOR D 1 4 Z000	XOR E 1 4 Z000	XOR H 1 4 Z000	XOR L 1 4 Z000	XOR (HL) 1 8 Z000	XOR A 1 4 Z000
Bx	OR B 1 4 Z000	OR C 1 4 Z000	OR D 1 4 Z000	OR E 1 4 Z000	OR H 1 4 Z000	OR L 1 4 Z000	OR (HL) 1 8 Z000	OR A 1 4 Z000	CP B 1 4 Z1HC	CP C 1 4 Z1HC	CP D 1 4 Z1HC	CP E 1 4 Z1HC	CP H 1 4 Z1HC	CP L 1 4 Z1HC	CP (HL) 1 8 Z1HC	CP A 1 4 Z1HC
Cx	RET NZ 1 20/8 ----	POP BC 1 12 ----	JP NZ,a16 3 16/12 ----	JP a16 3 16 ----	CALL NZ,a16 3 24/12 ----	PUSH BC 1 16 ----	ADD A,d8 2 8 Z0HC	RST 00H 1 16 ----	RET Z 1 20/8 ----	RET 1 16 ----	JP Z,a16 3 16/12 ----	PREFIX CB 1 4 ----	CALL Z,a16 3 24/12 ----	CALL a16 3 24 ----	ADC A,d8 2 8 Z0HC	RST 08H 1 16 ----
Dx	RET NC 1 20/8 ----	POP DE 1 12 ----	JP NC,a16 3 16/12 ----		CALL NC,a16 3 24/12 ----	PUSH DE 1 16 ----	SUB d8 2 8 Z1HC	RST 10H 1 16 ----	RET C 1 20/8 ----	RETI 1 16 ----	JP C,a16 3 16/12 ----		CALL C,a16 3 24/12 ----		SBC A,d8 2 8 Z1HC	RST 18H 1 16 ----
Ex	LDH (a8),A 2 12 ----	POP HL 1 12 ----	LD (C),A 2 8 ----			PUSH HL 1 16 ----	AND d8 2 8 Z010	RST 20H 1 16 ----	ADD SP,r8 2 16 00HC	JP (HL) 1 4 ----	LD (a16),A 3 16 ----				XOR d8 2 8 Z000	RST 28H 1 16 ----
Fx	LDH A,(a8) 2 12 ----	POP AF 1 12 ZNHC	LD A,(C) 2 8 ----	DI 1 4 ----		PUSH AF 1 16 ----	OR d8 2 8 Z000	RST 30H 1 16 ----	LD HL,SP+r8 2 12 00HC	LD SP,HL 1 8 ----	LD A,(a16) 3 16 ----	EI 1 4 ----			CP d8 2 8 Z1HC	RST 38H 1 16 ----

Prefix CB

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	RLC B 2 8 Z00C	RLC C 2 8 Z00C	RLC D 2 8 Z00C	RLC E 2 8 Z00C	RLC H 2 8 Z00C	RLC L 2 8 Z00C	RLC (HL) 2 16 Z00C	RLC A 2 8 Z00C	RRC B 2 8 Z00C	RRC C 2 8 Z00C	RRC D 2 8 Z00C	RRC E 2 8 Z00C	RRC H 2 8 Z00C	RRC L 2 8 Z00C	RRC (HL) 2 16 Z00C	RRC A 2 8 Z00C
1x	RL B 2 8 Z00C	RL C 2 8 Z00C	RL D 2 8 Z00C	RL E 2 8 Z00C	RL H 2 8 Z00C	RL L 2 8 Z00C	RL (HL) 2 16 Z00C	RL A 2 8 Z00C	RR B 2 8 Z00C	RR C 2 8 Z00C	RR D 2 8 Z00C	RR E 2 8 Z00C	RR H 2 8 Z00C	RR L 2 8 Z00C	RR (HL) 2 16 Z00C	RR A 2 8 Z00C
2x	SLA B 2 8 Z00C	SLA C 2 8 Z00C	SLA D 2 8 Z00C	SLA E 2 8 Z00C	SLA H 2 8 Z00C	SLA L 2 8 Z00C	SLA (HL) 2 16 Z00C	SLA A 2 8 Z00C	SRA B 2 8 Z000	SRA C 2 8 Z000	SRA D 2 8 Z000	SRA E 2 8 Z000	SRA H 2 8 Z000	SRA L 2 8 Z000	SRA (HL) 2 16 Z000	SRA A 2 8 Z000
3x	SWAP B 2 8 Z000	SWAP C 2 8 Z000	SWAP D 2 8 Z000	SWAP E 2 8 Z000	SWAP H 2 8 Z000	SWAP L 2 8 Z000	SWAP (HL) 2 16 Z000	SWAP A 2 8 Z000	SRL B 2 8 Z00C	SRL C 2 8 Z00C	SRL D 2 8 Z00C	SRL E 2 8 Z00C	SRL H 2 8 Z00C	SRL L 2 8 Z00C	SRL (HL) 2 16 Z00C	SRL A 2 8 Z00C
4x	BIT 0,B 2 8 Z01-	BIT 0,C 2 8 Z01-	BIT 0,D 2 8 Z01-	BIT 0,E 2 8 Z01-	BIT 0,H 2 8 Z01-	BIT 0,L 2 8 Z01-	BIT 0,(HL) 2 16 Z01-	BIT 0,A 2 8 Z01-	BIT 1,B 2 8 Z01-	BIT 1,C 2 8 Z01-	BIT 1,D 2 8 Z01-	BIT 1,E 2 8 Z01-	BIT 1,H 2 8 Z01-	BIT 1,L 2 8 Z01-	BIT 1,(HL) 2 16 Z01-	BIT 1,A 2 8 Z01-
5x	BIT 2,B 2 8 Z01-	BIT 2,C 2 8 Z01-	BIT 2,D 2 8 Z01-	BIT 2,E 2 8 Z01-	BIT 2,H 2 8 Z01-	BIT 2,L 2 8 Z01-	BIT 2,(HL) 2 16 Z01-	BIT 2,A 2 8 Z01-	BIT 3,B 2 8 Z01-	BIT 3,C 2 8 Z01-	BIT 3,D 2 8 Z01-	BIT 3,E 2 8 Z01-	BIT 3,H 2 8 Z01-	BIT 3,L 2 8 Z01-	BIT 3,(HL) 2 16 Z01-	BIT 3,A 2 8 Z01-
6x	BIT 4,B 2 8 Z01-	BIT 4,C 2 8 Z01-	BIT 4,D 2 8 Z01-	BIT 4,E 2 8 Z01-	BIT 4,H 2 8 Z01-	BIT 4,L 2 8 Z01-	BIT 4,(HL) 2 16 Z01-	BIT 4,A 2 8 Z01-	BIT 5,B 2 8 Z01-	BIT 5,C 2 8 Z01-	BIT 5,D 2 8 Z01-	BIT 5,E 2 8 Z01-	BIT 5,H 2 8 Z01-	BIT 5,L 2 8 Z01-	BIT 5,(HL) 2 16 Z01-	BIT 5,A 2 8 Z01-
7x	BIT 6,B 2 8 Z01-	BIT 6,C 2 8 Z01-	BIT 6,D 2 8 Z01-	BIT 6,E 2 8 Z01-	BIT 6,H 2 8 Z01-	BIT 6,L 2 8 Z01-	BIT 6,(HL) 2 16 Z01-	BIT 6,A 2 8 Z01-	BIT 7,B 2 8 Z01-	BIT 7,C 2 8 Z01-	BIT 7,D 2 8 Z01-	BIT 7,E 2 8 Z01-	BIT 7,H 2 8 Z01-	BIT 7,L 2 8 Z01-	BIT 7,(HL) 2 16 Z01-	BIT 7,A 2 8 Z01-
8x	RES 0,B 2 8 ----	RES 0,C 2 8 ----	RES 0,D 2 8 ----	RES 0,E 2 8 ----	RES 0,H 2 8 ----	RES 0,L 2 8 ----	RES 0,(HL) 2 16 ----	RES 0,A 2 8 ----	RES 1,B 2 8 ----	RES 1,C 2 8 ----	RES 1,D 2 8 ----	RES 1,E 2 8 ----	RES 1,H 2 8 ----	RES 1,L 2 8 ----	RES 1,(HL) 2 16 ----	RES 1,A 2 8 ----
9x	RES 2,B 2 8 ----	RES 2,C 2 8 ----	RES 2,D 2 8 ----	RES 2,E 2 8 ----	RES 2,H 2 8 ----	RES 2,L 2 8 ----	RES 2,(HL) 2 16 ----	RES 2,A 2 8 ----	RES 3,B 2 8 ----	RES 3,C 2 8 ----	RES 3,D 2 8 ----	RES 3,E 2 8 ----	RES 3,H 2 8 ----	RES 3,L 2 8 ----	RES 3,(HL) 2 16 ----	RES 3,A 2 8 ----
Ax	RES 4,B 2 8 ----	RES 4,C 2 8 ----	RES 4,D 2 8 ----	RES 4,E 2 8 ----	RES 4,H 2 8 ----	RES 4,L 2 8 ----	RES 4,(HL) 2 16 ----	RES 4,A 2 8 ----	RES 5,B 2 8 ----	RES 5,C 2 8 ----	RES 5,D 2 8 ----	RES 5,E 2 8 ----	RES 5,H 2 8 ----	RES 5,L 2 8 ----	RES 5,(HL) 2 16 ----	RES 5,A 2 8 ----
Bx	RES 6,B 2 8 ----	RES 6,C 2 8 ----	RES 6,D 2 8 ----	RES 6,E 2 8 ----	RES 6,H 2 8 ----	RES 6,L 2 8 ----	RES 6,(HL) 2 16 ----	RES 6,A 2 8 ----	RES 7,B 2 8 ----	RES 7,C 2 8 ----	RES 7,D 2 8 ----	RES 7,E 2 8 ----	RES 7,H 2 8 ----	RES 7,L 2 8 ----	RES 7,(HL) 2 16 ----	RES 7,A 2 8 ----
Cx	SET 0,B 2 8 ----	SET 0,C 2 8 ----	SET 0,D 2 8 ----	SET 0,E 2 8 ----	SET 0,H 2 8 ----	SET 0,L 2 8 ----	SET 0,(HL) 2 16 ----	SET 0,A 2 8 ----	SET 1,B 2 8 ----	SET 1,C 2 8 ----	SET 1,D 2 8 ----	SET 1,E 2 8 ----	SET 1,H 2 8 ----	SET 1,L 2 8 ----	SET 1,(HL) 2 16 ----	SET 1,A 2 8 ----
Dx	SET 2,B 2 8 ----	SET 2,C 2 8 ----	SET 2,D 2 8 ----	SET 2,E 2 8 ----	SET 2,H 2 8 ----	SET 2,L 2 8 ----	SET 2,(HL) 2 16 ----	SET 2,A 2 8 ----	SET 3,B 2 8 ----	SET 3,C 2 8 ----	SET 3,D 2 8 ----	SET 3,E 2 8 ----	SET 3,H 2 8 ----	SET 3,L 2 8 ----	SET 3,(HL) 2 16 ----	SET 3,A 2 8 ----
Ex	SET 4,B 2 8 ----	SET 4,C 2 8 ----	SET 4,D 2 8 ----	SET 4,E 2 8 ----	SET 4,H 2 8 ----	SET 4,L 2 8 ----	SET 4,(HL) 2 16 ----	SET 4,A 2 8 ----	SET 5,B 2 8 ----	SET 5,C 2 8 ----	SET 5,D 2 8 ----	SET 5,E 2 8 ----	SET 5,H 2 8 ----	SET 5,L 2 8 ----	SET 5,(HL) 2 16 ----	SET 5,A 2 8 ----
Fx	SET 6,B 2 8 ----	SET 6,C 2 8 ----	SET 6,D 2 8 ----	SET 6,E 2 8 ----	SET 6,H 2 8 ----	SET 6,L 2 8 ----	SET 6,(HL) 2 16 ----	SET 6,A 2 8 ----	SET 7,B 2 8 ----	SET 7,C 2 8 ----	SET 7,D 2 8 ----	SET 7,E 2 8 ----	SET 7,H 2 8 ----	SET 7,L 2 8 ----	SET 7,(HL) 2 16 ----	SET 7,A 2 8 ----

Legend:

- Misc/control instructions
- Jumps/calls
- 8bit load/store/move instructions
- 16bit load/store/move instructions
- 8bit arithmetic/logical instructions
- 16bit arithmetic/logical instructions
- 8bit rotations/shifts and bit instructions

```
Length in bytes →  | INS reg |  ← Instruction mnemonic
                   |  2  8   |  ← Duration in cycles
                   | Z N H C |  ← Flags affected
```

Duration of conditional calls and returns is different when action is taken or not. This is indicated by two numbers separated by "/". The higher number (on the left side of "/") means duration of instruction when action is taken, the lower number (on the right side of "/") means duration of instruction when action is not taken.

Instruction **STOP** has according to manuals opcode **10 00** and thus is 2 bytes long. Anyhow it seems there is no reason for it so some assemblers code it simply as one byte instruction **10**.

Flags affected are always shown in **Z H N C** order. If flag is marked by "**0**" it means it is reset after the instruction. If it is marked by "**1**" it is set. If it is marked by "**-**" it is not changed. If it is marked by "**Z**", "**N**", "**H**" or "**C**" corresponding flag is affected as expected by its function.

d8 means immediate 8 bit data
d16 means immediate 16 bit data
a8 means 8 bit unsigned data, which are added to $FF00 in certain instructions (replacement for missing **IN** and **OUT** instructions)
a16 means 16 bit address
r8 means 8 bit signed data, which are added to program counter

LD A,(C) has alternative mnemonic **LD A,($FF00+C)**
LD C,(A) has alternative mnemonic **LD ($FF00+C),A**
LDH A,(a8) has alternative mnemonic **LD A,($FF00+a8)**
LDH (a8),A has alternative mnemonic **LD ($FF00+a8),A**
LD A,(HL+) has alternative mnemonic **LD A,(HLI)** or **LDI A,(HL)**
LD (HL+),A has alternative mnemonic **LD (HLI),A** or **LDI (HL),A**
LD A,(HL-) has alternative mnemonic **LD A,(HLD)** or **LDD A,(HL)**
LD (HL-),A has alternative mnemonic **LD (HLD),A** or **LDD (HL),A**
LD HL,SP+r8 has alternative mnemonic **LDHL SP,r8**

Registers

15 . . . 8	7 . . . 0
A (accumulator)	F (flags)
B	C
D	E
H	L

15 . . . 0
SP (stack pointer)
PC (program counter)

Flag register (F) bits:

7	6	5	4	3	2	1	0
Z	N	H	C	0	0	0	0

Z - Zero Flag
N - Subtract Flag
H - Half Carry Flag
C - Carry Flag
0 - Not used, always zero