OTAKU NO GAMEBOY

Gameboy Crib Sheet
V1.0 99/08/23

http://www.gbdev.org/
http://www.otakunozoku.com/gameboy.html

Op-code	Destination	Source	Z	N	H	C	Clk	Size
ADC A,(HL)	A	(HL)	R	0	R	R	2	1
ADC A,n8	A	8-bit integer	R	0	R	R	2	2
ADC A,r8	A	A,B,C,D,E,H,L	R	0	R	R	1	1
ADD A,(HL)	A	(HL)	R	0	R	R	2	1
ADD A,n8	A	8-bit integer	R	0	R	R	2	2
ADD A,r8	A	A,B,C,D,E,H,L	R	0	R	R	1	1
ADD HL,r16	HL	BC,DE,SP		0	R	R	2	1
ADD SP,e8	SP	8-bit offset	0	0	R	R	4	2
AND (HL)	A	(HL)	R	0	1	0	2	1
AND n8	A	8-bit integer	R	0	1	0	2	2
AND r8	A	A,B,C,D,E,H,L	R	0	1	0	1	1
BIT n3,(HL)	Zero Flag	(HL)	R	0	1		3	2
BIT n3,r8	Zero Flag	A,B,C,D,E,H,L	R	0	1		2	2
CALL cc,n16	PC	16-bit addr					6/3	3
CALL n16	PC	16-bit addr					6	3
CCF	Carry Flag			0	0	R	1	1
CP (HL)	Flags	(HL)	R	1	R	R	2	1
CP n8	Flags	8-bit integer	R	1	R	R	2	2
CP r8	Flags	A,B,C,D,E,H,L	R	1	R	R	1	1
CPL	A	A		1	1		1	1
DAA	A	A	R		0	R	1	1
DEC (HL)	(HL)	(HL)	R	1	R		3	1
DEC r16	BC,DE,HL,SP						2	1
DEC r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	1	R		1	1
DI							1	1
EI							1	1
HALT							1	1
INC (HL)	(HL)	(HL)	R	0	R		3	1
INC r16	BC,DE,HL,SP						2	1
INC r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	0	R		1	1
JP (HL)	PC	(HL)					1	1
JP cc,n16	PC	16-bit addr					4/3	3
JP n16	PC	16-bit addr					4	3
JR cc,n8	PC	8-bit integer					3/2	2
JR n8	PC	8-bit integer					3	2
LD (C),A	(C)	A					2	1
LD (HL),n8	(HL)	8-bit integer					3	2
LD (HL),r8	(HL)	A,B,C,D,E,H,L					2	1
LD (n16),A	(16-bit addr)	A					4	3
LD (n16),SP	(16-bit addr)	SP					5	3
LD (r16),A	(BC),(DE),(HL)	A					2	1
LD A,(C)	A	(C)					2	1
LD A,(n16)	A	(16-bit addr)					4	3
LD A,(r16)	A	(BC),(DE),(HL)					2	1
LD HL,(SP+e8)	HL	(SP+8-bit off)	0	0	R	R	3	2
LD r16,n16	BC,DE,HL,SP	16-bit int					3	3
LD r8,(HL)	A,B,C,D,E,H,L	(HL)					2	1
LD r8,n8	A,B,C,D,E,H,L	8-bit integer					2	2
LD r8,r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L					1	1
LD SP,HL	SP	HL					2	1
LDD (HL),A	(HL)	A					2	1
LDD A,(HL)	A	(HL)					2	1
LDH (n8),A	(8-bit off)	A					3	2
LDH A,(n8)	A	(8-bit off)					3	2
LDI (HL),A	(HL)	A					2	1
LDI A,(HL)	A	(HL)					2	1
NOP							1	1
OR (HL)	A	(HL)	R	0	0	0	2	1
OR n8	A	8-bit integer	R	0	0	0	2	2
OR r8	A	A,B,C,D,E,H,L	R	0	0	0	1	1
POP r16	AF,BC,DE,HL	(SP)					3	3
PUSH r16	(SP)	AF,BC,DE,HL					4	3
RES n3,(HL)	Bit in Memory	(HL)					3	2
RES n3,r8	Bit in Register	A,B,C,D,E,H,L					2	2
RET	PC						4	1
RET cc	PC	Condition Flag					5/2	1
RETI	PC						4	1
RL (HL)	(HL)	(HL)	R	0	0	R	4	2
RL r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	0	0	R	2	2
RLA	A	A	0	0	0	R	1	1
RLC (HL)	(HL)	(HL)	R	0	0	R	4	2
RLC r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	0	0	R	2	2
RLCA	A	A	0	0	0	R	1	1
RR (HL)	(HL)	(HL)	R	0	0	R	4	2
RR r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	0	0	R	2	2
RRA	A	A	0	0	0	R	1	1
RRC (HL)	(HL)	(HL)	R	0	0	R	4	2
RRC r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	0	0	R	2	2
RRCA	A	A	0	0	0	R	1	1
RST f	PC						4	1
SBC A,(HL)	A	(HL)	R	1	R	R	2	1
SBC A,n8	A	8-bit integer	R	1	R	R	2	2
SBC A,r8	A	A,B,C,D,E,H,L	R	1	R	R	1	1
SCF	Carry Flag			0	0	1	1	1
SET n3,(HL)	Bit in Memory	(HL)					3	2
SET n3,r8	Bit in Register	A,B,C,D,E,H,L					2	2
SLA (HL)	(HL)	(HL)	R	0	0	R	4	2
SLA r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	0	0	R	2	2
SRA (HL)	(HL)	(HL)	R	0	0	R	4	2
SRA r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	0	0	R	2	2
SRL (HL)	(HL)	(HL)	R	0	0	R	4	2
SRL r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	0	0	R	2	2
STOP							1	2
SUB (HL)	A	(HL)	R	1	R	R	2	1
SUB n8	A	8-bit integer	R	1	R	R	2	2
SUB r8	A	A,B,C,D,E,H,L	R	1	R	R	1	1
SWAP (HL)	(HL)	(HL)	R	0	0	0	4	2
SWAP r8	A,B,C,D,E,H,L	A,B,C,D,E,H,L	R	0	0	0	2	2
XOR (HL)	A	(HL)	R	0	0	0	2	1
XOR n8	A	8-bit integer	R	0	0	0	2	2
XOR r8	A	A,B,C,D,E,H,L	R	0	0	0	1	1

Instruction Descriptions (except shifts/rotates)

ADC x,y	Add Y+CY to x
ADD x,y	Add y to x
AND x	AND x to A
BIT b,x	Test bit b of x
CALL c,x	If condition c is true call subroutine at x
CALL x	Call subroutine at x (push PC and jump to x)
CCF	Complement carry flag
CP x	Compare A with x
CPL	Complement A (1's complement)
DAA	Decimal adjust A (after add/sub of BCD data)
DEC x	Decrement x by 1
DI	Disable interrupts
EI	Enable interrupts
HALT	Halt (wait for interrupt or reset)
INC x	Increment x by 1
JP c,x	If condition c is true jump to location x
JP x	Jump to location x
JR c,d	If condition c is true jump relative by d
JR d	Jump relative by d
LD x,y	Load x with y (move y to x)
LDD x,y	Load A with (HL), DEC HL
LDI x,y	Load A with (HL), INC HL
NOP	No operation
OR x	OR x to A
POP x	Pop x from top of stack updating SP
PUSH x	Push x onto top of stack updating SP
RES b,x	Reset bit b of x (to 0)
RET	Return from subroutine (POP PC)
RET c	If condition c is true return from subroutine
RETI	Return from interrupt
RST x	Call subroutine at x (1 byte instruction)
SBC x	Subtract y+CY from x
SCF	Set carry flag (to 1)
SET b,x	Set bit b of x (to 1)instruction)
STOP	Stop CPU until P1-P10 go high
SUB x	Subtract x from A
XOR x	XOR x to A

8-bit Unsigned Comp

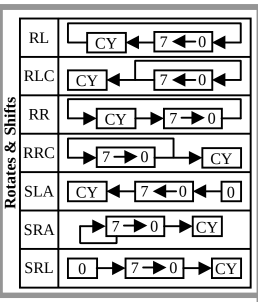
A<B	`JP C,yes`

A<=B	`JP C,yes`
	`JP Z,yes`

A==B	`JP Z,yes`
A<>B	`JP NZ,yes`
A>=B	`JP NC,yes`

A>B	`JR C,3`
	`JP NZ,yes`

Rotates & Shifts



Z80 Registers

15…8	7…0	
A	F	Accumulator/Flags
B	C	
D	E	
H	L	
SP		Stack Pointer
PC		Program Counter

Z80 Flags

Z	7	Zero – Set when result of a math operation is zero, or two values match for CP operation.
N	6	Subtract – Set if a subtraction was performed in the last math operation
H	5	Half-Carry – Set if a carry occurred from the lower nibble in the last math operation.
C	4	Carry – Set if carry occurred in last math operation, or if A is less than value for CP operation.
X	3	Not Used
X	2	Not Used
X	1	Not Used
X	0	Not Used

CE xx	ADC A,$xx	CB 70	BIT 6,B	32	LD (HLD),A	B0	OR B	C8	RET Z	CB D8	SET 3,B
8E	ADC A,(HL)	CB 71	BIT 6,C	22	LD (HLI),A	B1	OR C	D9	RETI	CB D9	SET 3,C
8F	ADC A,A	CB 72	BIT 6,D	3E xx	LD A,$xx	B2	OR D	CB 16	RL (HL)	CB DA	SET 3,D
88	ADC A,B	CB 73	BIT 6,E	FA bb aa	LD A,($aabb)	B3	OR E	CB 17	RL A	CB DB	SET 3,E
89	ADC A,C	CB 74	BIT 6,H	F0 xx	LD A,($xx)	B4	OR H	CB 10	RL B	CB DC	SET 3,H
8A	ADC A,D	CB 75	BIT 6,L	0A	LD A,(BC)	B5	OR L	CB 11	RL C	CB DD	SET 3,L
8B	ADC A,E	CB 7D	BIT 7,(HL)	F2	LD A,(C)	F1	POP AF	CB 12	RL D	CB E6	SET 4,(HL)
8C	ADC A,H	CB 7F	BIT 7,A	1A	LD A,(DE)	C1	POP BC	CB 13	RL E	CB E7	SET 4,A
8D	ADC A,L	CB 78	BIT 7,B	7E	LD A,(HL)	D1	POP DE	CB 14	RL H	CB E0	SET 4,B
C6 xx	ADD A,$xx	CB 79	BIT 7,C	3A	LD A,(HLD)	E1	POP HL	CB 15	RL L	CB E1	SET 4,C
86	ADD A,(HL)	CB 7A	BIT 7,D	2A	LD A,(HLI)	F5	PUSH AF	17	RLA	CB E2	SET 4,D
87	ADD A,A	CB 7B	BIT 7,E	7F	LD A,A	C5	PUSH BC	CB 06	RLC (HL)	CB E3	SET 4,E
80	ADD A,B	CB 7C	BIT 7,H	78	LD A,B	D5	PUSH DE	CB 07	RLC A	CB E4	SET 4,H
81	ADD A,C	CD bb aa	CALL $aabb	79	LD A,C	E5	PUSH HL	CB 00	RLC B	CB E5	SET 4,L
82	ADD A,D	DC bb aa	CALL C,$aabb	7A	LD A,D	CB 86	RES 0,(HL)	CB 01	RLC C	CB EE	SET 5,(HL)
83	ADD A,E	D4 bb aa	CALL NC,$aabb	7B	LD A,E	CB 87	RES 0,A	CB 02	RLC D	CB EF	SET 5,A
84	ADD A,H	C4 bb aa	CALL NZ,$aabb	7C	LD A,H	CB 80	RES 0,B	CB 03	RLC E	CB E8	SET 5,B
85	ADD A,L	CC bb aa	CALL Z,$aabb	7D	LD A,L	CB 81	RES 0,C	CB 04	RLC H	CB E9	SET 5,C
09	ADD HL,BC	3F	CCF	06 xx	LD B,$xx	CB 82	RES 0,D	CB 05	RLC L	CB EA	SET 5,D
19	ADD HL,DE	FE xx	CP $xx	46	LD B,(HL)	CB 83	RES 0,E	07	RLCA	CB EB	SET 5,E
29	ADD HL,HL	BE	CP (HL)	47	LD B,A	CB 84	RES 0,H	CB 1E	RR (HL)	CB EC	SET 5,H
39	ADD HL,SP	BF	CP A	40	LD B,B	CB 85	RES 0,L	CB 1F	RR A	CB ED	SET 5,L
E8 xx	ADD SP,xx	B8	CP B	41	LD B,C	CB 8E	RES 1,(HL)	CB 18	RR B	CB F6	SET 6,(HL)
E6 xx	AND $xx	B9	CP C	42	LD B,D	CB 8F	RES 1,A	CB 19	RR C	CB F7	SET 6,A
A6	AND (HL)	BA	CP D	43	LD B,E	CB 88	RES 1,B	CB 1A	RR D	CB F0	SET 6,B
A7	AND A	BB	CP E	44	LD B,H	CB 89	RES 1,C	CB 1B	RR E	CB F1	SET 6,C
A0	AND B	BC	CP H	45	LD B,L	CB 8A	RES 1,D	CB 1C	RR H	CB F2	SET 6,D
A1	AND C	BD	CP L	01 bb aa	LD BC,$aabb	CB 8B	RES 1,E	CB 1D	RR L	CB F3	SET 6,E
A2	AND D	2F	CPL	0E xx	LD C,$xx	CB 8C	RES 1,H	1F	RRA	CB F4	SET 6,H
A3	AND E	27	DAA	4E	LD C,(HL)	CB 8D	RES 1,L	CB 0E	RRC (HL)	CB F5	SET 6,L
A4	AND H	35	DEC (HL)	4F	LD C,A	CB 96	RES 2,(HL)	CB 0F	RRC A	CB FE	SET 7,(HL)
A5	AND L	3D	DEC A	48	LD C,B	CB 97	RES 2,A	CB 08	RRC B	CB FF	SET 7,A
CB 46	BIT 0,(HL)	05	DEC B	49	LD C,C	CB 90	RES 2,B	CB 09	RRC C	CB F8	SET 7,B
CB 47	BIT 0,A	0B	DEC BC	4A	LD C,D	CB 91	RES 2,C	CB 0A	RRC D	CB F9	SET 7,C
CB 40	BIT 0,B	0D	DEC C	4B	LD C,E	CB 92	RES 2,D	CB 0B	RRC E	CB FA	SET 7,D
CB 41	BIT 0,C	15	DEC D	4C	LD C,H	CB 93	RES 2,E	CB 0C	RRC H	CB FB	SET 7,E
CB 42	BIT 0,D	1B	DEC DE	4D	LD C,L	CB 94	RES 2,H	CB 0D	RRC L	CB FC	SET 7,H
CB 43	BIT 0,E	1D	DEC E	16 xx	LD D,$xx	CB 95	RES 2,L	0F	RRCA	CB FD	SET 7,L
CB 44	BIT 0,H	25	DEC H	56	LD D,(HL)	CB 9E	RES 3,(HL)	C7	RST $00	CB 26	SLA (HL)
CB 45	BIT 0,L	2B	DEC HL	57	LD D,A	CB 9F	RES 3,A	CF	RST $08	CB 27	SLA A
CB 4E	BIT 1,(HL)	2D	DEC L	50	LD D,B	CB 98	RES 3,B	D7	RST $10	CB 20	SLA B
CB 4F	BIT 1,A	3B	DEC SP	51	LD D,C	CB 99	RES 3,C	DF	RST $18	CB 21	SLA C
CB 48	BIT 1,B	F3	DI	52	LD D,D	CB 9A	RES 3,D	E7	RST $20	CB 22	SLA D
CB 49	BIT 1,C	FB	EI	53	LD D,E	CB 9B	RES 3,E	EF	RST $28	CB 23	SLA E
CB 4A	BIT 1,D	76	HALT	54	LD D,H	CB 9C	RES 3,H	F7	RST $30	CB 24	SLA H
CB 4B	BIT 1,E	34	INC (HL)	55	LD D,L	CB 9D	RES 3,L	FF	RST $38	CB 25	SLA L
CB 4C	BIT 1,H	3C	INC A	11 bb aa	LD DE,$aabb	CB A6	RES 4,(HL)	DE xx	SBC A,$xx	CB 2E	SRA (HL)
CB 4D	BIT 1,L	04	INC B	1E xx	LD E,$xx	CB A7	RES 4,A	9E	SBC A,(HL)	CB 2F	SRA A
CB 56	BIT 2,(HL)	03	INC BC	5E	LD E,(HL)	CB A0	RES 4,B	9F	SBC A,A	CB 28	SRA B
CB 57	BIT 2,A	0C	INC C	5F	LD E,A	CB A1	RES 4,C	98	SBC A,B	CB 29	SRA C
CB 50	BIT 2,B	14	INC D	58	LD E,B	CB A2	RES 4,D	99	SBC A,C	CB 2A	SRA D
CB 51	BIT 2,C	13	INC DE	59	LD E,C	CB A3	RES 4,E	9A	SBC A,D	CB 2B	SRA E
CB 52	BIT 2,D	1C	INC E	5A	LD E,D	CB A4	RES 4,H	9B	SBC A,E	CB 2C	SRA H
CB 53	BIT 2,E	24	INC H	5B	LD E,E	CB A5	RES 4,L	9C	SBC A,H	CB 2D	SRA L
CB 54	BIT 2,H	23	INC HL	5C	LD E,H	CB AE	RES 5,(HL)	9D	SBC A,L	CB 3E	SRL (HL)
CB 55	BIT 2,L	2C	INC L	5D	LD E,L	CB AF	RES 5,A	37	SCF	CB 3F	SRL A
CB 5E	BIT 3,(HL)	33	INC SP	26 xx	LD H,$xx	CB A8	RES 5,B	CB C6	SET 0,(HL)	CB 38	SRL B
CB 5F	BIT 3,A	C3 bb aa	JP $aabb	66	LD H,(HL)	CB A9	RES 5,C	CB C7	SET 0,A	CB 39	SRL C
CB 58	BIT 3,B	E9	JP (HL)	67	LD H,A	CB AA	RES 5,D	CB C0	SET 0,B	CB 3A	SRL D
CB 59	BIT 3,C	DA bb aa	JP C,$aabb	6F	LD H,A	CB AB	RES 5,E	CB C1	SET 0,C	CB 3B	SRL E
CB 5A	BIT 3,D	D2 bb aa	JP NC,$aabb	60	LD H,B	CB AC	RES 5,H	CB C2	SET 0,D	CB 3C	SRL H
CB 5B	BIT 3,E	C2 bb aa	JP NZ,$aabb	61	LD H,C	CB AD	RES 5,L	CB C3	SET 0,E	CB 3D	SRL L
CB 5C	BIT 3,H	CA bb aa	JP Z,$aabb	62	LD H,D	CB B6	RES 6,(HL)	CB C4	SET 0,H	10 00	STOP
CB 5D	BIT 3,L	18 xx	JR $xx	63	LD H,E	CB B7	RES 6,A	CB C5	SET 0,L	D6 xx	SUB $xx
CB 66	BIT 4,(HL)	38 xx	JR C,$xx	64	LD H,H	CB B0	RES 6,B	CB CE	SET 1,(HL)	97	SUB A
CB 67	BIT 4,A	30 xx	JR NC,$xx	65	LD H,L	CB B1	RES 6,C	CB CF	SET 1,A	90	SUB B
CB 60	BIT 4,B	20 xx	JR NZ,$xx	21 bb aa	LD HL,$aabb	CB B2	RES 6,D	CB C8	SET 1,B	91	SUB C
CB 61	BIT 4,C	28 xx	JR Z,$xx	F8	LD HL,SP	CB B3	RES 6,E	CB C9	SET 1,C	92	SUB D
CB 62	BIT 4,D	EA bb aa	LD ($aabb),A	2E xx	LD L,$xx	CB B4	RES 6,H	CB CA	SET 1,D	93	SUB E
CB 63	BIT 4,E	08 bb aa	LD ($aabb),SP	6E	LD L,(HL)	CB B5	RES 6,L	CB CB	SET 1,E	94	SUB H
CB 64	BIT 4,H	E0 xx	LD ($xx),A	68	LD L,B	CB BE	RES 7,(HL)	CB CC	SET 1,H	95	SUB L
CB 65	BIT 4,L	02	LD (BC),A	69	LD L,C	CB BF	RES 7,A	CB CD	SET 1,L	CB 37	SWAP A
CB 6E	BIT 5,(HL)	E2	LD (C),A	6A	LD L,D	CB B8	RES 7,B	CB D6	SET 2,(HL)	EE xx	XOR $xx
CB 6F	BIT 5,A	12	LD (DE),A	6B	LD L,E	CB B9	RES 7,C	CB D7	SET 2,A	AE	XOR (HL)
CB 68	BIT 5,B	36 xx	LD (HL),$xx	6C	LD L,H	CB BA	RES 7,D	CB D0	SET 2,B	AF	XOR A
CB 69	BIT 5,C	77	LD (HL),A	6D	LD L,L	CB BB	RES 7,E	CB D1	SET 2,C	A8	XOR B
CB 6A	BIT 5,D	70	LD (HL),B	31 bb aa	LD SP,$aabb	CB BC	RES 7,H	CB D2	SET 2,D	A9	XOR C
CB 6B	BIT 5,E	71	LD (HL),C	F9	LD SP,HL	CB BD	RES 7,L	CB D3	SET 2,E	AA	XOR D
CB 6C	BIT 5,H	72	LD (HL),D	00	NOP	C9	RET	CB D4	SET 2,H	AB	XOR E
CB 6D	BIT 5,L	73	LD (HL),E	F6 xx	OR $xx	D8	RET C	CB D5	SET 2,L	AC	XOR H
CB 76	BIT 6,(HL)	74	LD (HL),H	B6	OR (HL)	D0	RET NC	CB DE	SET 3,(HL)	AD	XOR L
CB 77	BIT 6,A	75	LD (HL),L	B7	OR A	C0	RET NZ	CB DF	SET 3,A		

Register	Purpose	Comment		Bit	Addr
P1	Read Joypad Info	P1F_5	W	5	FF00
		P1F_4	W	4	
		P1F_3	R	3	
		P1F_2	R	2	
		P1F_1	R	1	
		P1F_0	R	0	
SB	Serial Transfer Data		R/W		FF01
SC	Serial I/O Control		R/W		FF02
DIV	Timer Divider		R/W		FF04
TIMA	Timer Counter		R/W		FF05
TMA	Timer Modulo		R/W		FF06
TAC	Timer Control	Timer start/stop	R/W	2	FF07
		Timer speed	R/W	0-1	
IF	Interrupt Flag		R/W		FF0F
LCDC	LCD Control	LCD On/Off	R/W	7	FF40
		Window Addr	R/W	6	
		Window On/Off	R/W	5	
		Background Addr	R/W	3-4	
		Object Size	R/W	2	
		Object On/Off	R/W	1	
		Background On/Off	R/W	0	
STAT	LCD Status	LYCEQULY Coincidence	R/W	6	FF41
		Mode 10	R/W	5	
		Mode 01 (V-Blank)	R/W	4	
		Mode 00 (H-Blank)	R/W	3	
		Coincidence Flag	R/W	2	
		OAM/VRAM Lock	R/W	0-1	
SCY	Scroll Screen Y	Horizontal scroll	R/W		FF42
SCX	Scroll Screen X	Vertical scroll	R/W		FF43
LY	LCDC Y-Coord		R/W		FF44
LYC	LY Compare		R/W		FF45
DMA	DMA Transfer		R/W		FF46
BGP	BG Palette Data		R/W		FF47
OBP0	Obj Palette 0 Data		R/W		FF48
OBP1	Obj Palette 1 Data		R/W		FF49
WY	Window Y Pos		R/W		FF4A
WX	Window X Pos		R/W		FF4B
KEY1	CPU Speed Select	GBC only	R/W		FF4D
VBK	VRAM Bank Select	GBC only	R/W		FF4F
HDMA1	HBL General DMA	GBC only	R/W		FF51
HDMA2	HBL General DMA	GBC only	R/W		FF52
HDMA3	HBL General DMA	GBC only	R/W		FF53
HDMA4	HBL General DMA	GBC only	R/W		FF54
HDMA5	HBL General DMA	GBC only	R/W		FF55
RP	Infrared Comms	GBC only	R/W		FF56
BCPS	Bkg Colour Index	GBC only	R/W		FF68
BCPD	Bkg Colour Data	GBC only	R/W		FF69
OCPS	Obj Colour Index	GBC only	R/W		FF6A
OCPD	Obj Colour Data	GBC only	R/W		FF6B
SVBK	RAM Bank Select	GBC only	R/W		FF70
IE	Interrupt Enable	HILO Transition	R/W	4	FFFF
		Serial I/O Transfer Done	R/W	3	
		Timer Overflow	R/W	2	
		LCDC	R/W	1	
		VBL	R/W	0	
NR10	Audio Sweep	Sweep time	R/W	4-6	FF10
		Sweep increase/decrease	R/W	3	
		Sweep shift	R/W	0-2	
NR11	Audio Chan #1	Wave pattern duty	R/W	6-7	FF11
		Sound length data	R/W	0-5	
NR12	Envelope Chan #1	Initial value of envelope	R/W	4-7	FF12
		Envelope Up/Down	R/W	3	
		Number of envelope sweep	R/W	0-2	
NR13	Sound Freq #1	Frequency LSB	W		FF13
NR14	Sound Freq #1	Initialise	W	7	FF14
		Counter/consecutive	W	6	
		Frequency significant 3	W	0-2	
NR21	Audio Chan #2	Wave pattern duty	R/W	6-7	FF16
		Sound length data	R/W	0-5	
NR22	Envelope Chan #2	Initial value of envelope	R/W	4-7	FF17
		Envelope Up/Down	R/W	3	
		Number of envelope sweep	R/W	0-2	
NR23	Sound Freq #2	Frequency LSB	W		FF18
NR24	Sound Freq #2	Initialise	W	7	FF19
		Counter/consecutive	W	6	
		Frequency significant 3	W	0-2	
NR30	Audio Chan #3	Sound On/Off	R/W	7	FF1A
NR31	Sound Len #2	Sound length	R/W		FF1B
NR32	Volume #3	Select output level	R/W	5-6	FF1C
NR33	Sond Freq #3	Frequency LSB	W		FF1D
NR34	Sound Freq #3	Initialise	W	7	FF1E
		Counter/consecutive	W	6	
		Frequency significant 3	W	0-2	
NR41	Sound Len #4	Sound length	R/W	0-5	FF20
NR42	Envelope #4	Initial value of envelope	R/W	4-7	FF21
		Envelope Up/Down	R/W	3	
		Number of envelope sweep	R/W	0-2	
NR43	Audio Counter	Clock freq of polynomial	R/W	4-7	FF22
		Selection of polynomial	R/W	3	
		Selection of dividing ratio	R/W	0-2	
NR44	Audio Control	Initialise audio	R/W	7	FF23
		Counter/consecutive	R/W	6	
NR50	Channel Control	Vin→ SO2 On/Off	R/W	7	FF24
		SO2 ouput volume	R/W	4-6	
		Vin→ SO1 On/Off	R/W	3	
		SO1 ouput volume	R/W	0-2	
NR51	Sound Output	Output sound 4 to SO2	R/W	7	FF25
		Output sound 3 to SO2	R/W	6	
		Output sound 2 to SO2	R/W	5	
		Output sound 1 to SO2	R/W	4	
		Output sound 4 to SO1	R/W	3	
		Output sound 3 to SO1	R/W	2	
		Output sound 2 to SO1	R/W	1	
		Output sound 0 to SO1	R/W	0	
NR52	Sound On/Off	All Channels On/Off	R/W	7	FF26
		Channel #4 On/Off	R/W	3	
		Channel #3 On/Off	R/W	2	
		Channel #2 On/Off	R/W	1	
		Channel #1 On/Off	R/W	0	
AUD3W	AVERAM	16 bytes of sound sample	R/W		FF3F

MBC3 Memory Controller Registers

Register	Purpose	Comment	Bit	Addr Range	
RAMG	RAM/Clock write protect	Write $0A to enable		0000	1FFF
ROMB	ROM Bank Select	$00 to $7F = Rom Bank #		2000	3FFF
?	RAM Bank/Clock Select	Note 1		4000	5FFF
?	Clock latch	Note 2		6000	7FFF
SEC ($08)	Seconds Counter			4000	5FFF
MIN ($09)	Minutes Counter			4000	5FFF
HRS ($0A)	Hours Counter			4000	5FFF
DAYL ($0B)	Day Counter	LSB of Day Counter		4000	5FFF
DAYH ($0C)	Day Counter/Control	MSB of Day Counter	0	4000	5FFF
		Start/Stop Clock Counter	6	4000	5FFF
		Day Counter Carry (Note 3)	7	4000	5FFF

Note 1 : Values $00 to $03 select the RAM Bank #. Values $08 to $0C select a Clock register

Note 2 : Writing $00 and then $01 to this register latches the clock data. Another write of $00 and then $01 is required to latch updated data.

Note 3 : Bit 7 of clock register DAYH remains set until zero is written to it.

General Notes: To access the clock counter the RAM bank must first be enabled.
Due to a slow MBC3 interface 16T states are required between each register access.

MBC5 Memory Controller

Register	Purpose	Comment	Bit	Addr Range	
RAMG	External RAM Select	Write $0A to enable		0000	1FFF
ROMB0	ROM Bank Select	LSB of ROM Bank #		2000	2FFF
ROMB1	ROM Bank Select	MSB of ROM Bank #	0	3000	3FFF
RAMB	RAM Bank Select	RAM Bank # (Note 1)	0-3	4000	5FFF

General Notes: Unused bit positions in registers should be filled with zero when writing.

Note 1 : When a Rumble Pak is installed, bits 0-1 select the RAM Bank (maximum of 4 banks). Bit 3 controls the Rumble Pak. Bit 2 is unusued. A MOTOR ON (set bit 3) must be issued for 2 frames to start the Rumble Pak motor if it has not yet been started, or if it has been idle for more than 3 frames.

Cart Type	ROM	RAM	MBC	MMM01	Battery	Timer	Rumble
00	X						
01	X		1				
02	X	X	1				
03	X	X	1		X		
05	X		2				
06	X		2		X		
08	X	X					
09	X	X			X		
0B	X			X			
0C	X	X		X			
0D	X	X		X	X		
0F	X		3		X	X	
10	X	X	3		X	X	
11	X		3				
12	X	X	3				
13	X	X	3		X		
15	X		4				
16	X	X	4				
17	X	X	4		X		
19	X		5				
1A	X	X	5				
1B	X	X	5		X		
1C	X		X				X
1D	X	X	5				X
1E	X	X	5		X		X
FC	Camera						
FD	Bandai TAMA 5						
FE	HuC 3						
FF	HuC1+RAM+Batt						

OTAKU NO GAMEBOY 2

Please submit all comments/corrections to
otaku@weirdness.com

VRAM Memory Map

Tile Map 2	R/W	9C00	9FFF
Tile Map 1	R/W	9800	9BFF
Tile 00-7F (FF40, bit 4=0)	R/W	9000	97FF
Tiles 80-FF	R/W	8800	8FFF
Tiles 00-7F (FF40, bit 4=1)	R/W	8000	87FF

RGB Colour

Bit	Meaning
15	Unused
14 - 10	Blue colour value (0 to 31)
9 - 5	Green colour value (0 to 31)
4 - 0	Red colour value (0 to 31)

Memory Map (R/W)

Interrupt Enable	R/W	FFFF	FFFF
High RAM	R/W	FF80	FFFE
I/O Registers	R/W	FF00	FF7F
OAM RAM	R/W	FE00	FE9F
Low RAM	R/W	C000	DFFF
Cart RAM	R/W	A000	BFFF
Video RAM	R/W	8000	9FFF
ROM Bank 1-n	R	4000	7FFF
ROM Bank	R	0000	3FFF

Interrupts

Interrupt	Addr	Comment
Vertical Blank	$40	Occurs ~59.7 times per second, lasts ~1.1ms
LCD Control	$48	See STAT register
Timer Overflow	$50	TIMA register has changed from $FF to $00
Serial I/O Complete	$58	Serial transfer is complete
Joypad Pressed	$60	High to low transition on pins P10-P13

Video Timings

Horizontal line timing	108.7 µs
V-Blank	1.09 ms
Mode 10	19.31 µs
Mode 11	41.37 to 70.69µs
Mode 0 with 10 sprites on a scanline	18.72 µs
Mode 0 with no sprites on a scanline	48.64 µs

Clocks

CPU Clk @ 1x	4.194304 MHz
CPU Clk @ 2x	8.388608 MHz
Horiz Sync	9198 KHz
Vert Sync	59.73 Hz

Memory Map (Write Only)

RAM/ROM Select (MBC1)	W	6000	7FFF
RAM Bank Select	W	4000	5FFF
ROM Bank Select MSB (MBC5)	W	3000	3FFF
ROM Bank Select LSB	W	2000	2FFF
RAM Bank Enable	W	0000	1FFF

Gameboy Type

Value	Gameboy Type
01	DMG (SGB)
FF	MGB (SGB2)
11	CGB

The initial value in the Accumulator identifies the type of Gameboy unit.

VRAM Attributes

Byte	Bit	Purpose	Comment
0		Tile Index	
1	7	Priority	1 = Tile is in front of objects
1	6	Y Flip	1 = Tile is flipped vertically
1	5	X Flip	1 = Tile is flipped horizontally
1	4	Not Used	Should be set to 0
1	3	Tile Bank	1 = Upper tile bank (GBC only)
1	0-2	Palette Index	

OAM RAM Attributes

Byte	Bit	Purpose	Comment
0		Y Coord	
1		X Coord	
2		Tile Index	
3	7	Priority	0 = in front of background
3	6	Y Flip	1 = Sprite flipped vertically
3	5	X Flip	1 = Sprite flipped horizontally
3	4	Palette Bank	0 = OBJ0PAL / 1=OBJ1PAL
3	3	Tile Bank	0 = Lower tile bank
3	0-2	Palette Index	

Video Sizes

	Pixels	Tiles
VRAM Width	256	32
VRAM Height	256	32
Screen Width	160	20
Screen Height	144	18

Memory Map

$FFFF	Interrupt Enable Flag
$FF80	Zero Page (127 bytes)
$FF00	Hardware Registers
$FE00	OAM
$E000	Echo RAM
$D000	Game Unit WRAM Bank 1-7 Switchable 4 KBytes
$C000	Game Unit WRAM Bank 0 4 KBytes
$A000	GamePak WRAM 8 KBytes
$9C00	Background Display Data 2 Tile Indices/Attributes (Bankswitched)
$9800	Background Display Data 1 Tile Indices/Attributes (Bankswitched)
$8000	Bank 0 and 1 Character Data (Bank Switched)
$0150	User Program Area Bank 1 to n 16 KBytes User Program Area Bank 0 (fixed) 16 KBytes
$0100	ROM Registration Data Area
$0000	Interrupt Vectors RST Vectors

Cart Header

Checksum LSB	014F
Checksum MSB	014E
Complement	014D
Mask ROM Ver	014C
Old Maker Code	014B
Destination Code	014A
External RAM Size	0149
ROM Size	0148
Cart Type	0147
SGB Function	0146
Maker Code LSB	0145
Maker Code MSB	0144
Colour	0143
Game Title	0134
Nintendo Logo	0104
JP $XXXX	0101
NOP	0100

DMA Precautions

Do not switch ROM Banks if the DMA source addr is in the high ROM.	$4000-$7FFF
Do not switch RAM Banks if the DMA source addr is in the high RAM.	$D000-$DFFF
Do not switch VRAM Banks until HDMA has completed.	$8000-$9FFF
Source & Destination address must be 256-byte aligned.	$xx00
HALT cannot be used while a HDMA transfer is taking place.	
Screen must be enabled for a HDMA transfer to take place.	
HDMA must complete before another is initiated or HDMA registers altered.	
Transfer length must be correct. $80=16 bytes, $81=32 bytes, $82=48 bytes, $83 = 64bytes	
Bit 7 of HDMA 5 is clear during HDMA transfer, set on completion.	$FF55
GDMA is only reliable during VBL when LCD is enabled.	
CPU halts until GDMA completes.	
GDMA transfer time in 1xCPU mode. n = # of 16-byte blocks to transfer.	220+n*7.63µs
GDMA transfer time in 2xCPU mode. n = # of 16-byte blocks to transfer.	110+n*7.63µs

RAM Sizes

00	None		
01	16Kb	2KB	1
02	64Kb	8KB	1
03	256Kb	32KB	4
06	1Mb	128KB	16

ROM Sizes ($148)

00	256Kb	32KB	2
01	512Kb	64KB	4
02	1Mb	128KB	8
03	2Mb	256KB	16
04	4Mb	512KB	32
05	8Mb	1MB	64
06	16Mb	2MB	128
07	32Mb	4MB	256
08	64Mb	8MB	512
52	9Mb	1.1MB	72
53	10Mb	1.2MB	80
54	12Mb	1.5MB	96

Tone Conversion Table

Note	GB	KHz	Note	GB	KHz
C 0		8.176	E 5	1650	329.63
C# 0		8.662	F 5	1673	349.23
D 0		9.177	F #5	1694	369.99
D# 0		9.723	G 5	1714	391.99
E 0		10.301	G# 5	1732	415.31
F 0		10.913	A 5	1750	440.00
F# 0		11.562	A# 5	1767	466.16
G 0		12.250	B 5	1783	493.88
G# 0		12.978	C 6	1798	523.25
A 0		13.750	C# 6	1812	554.37
A# 0		14.568	D 6	1825	587.33
B 0		15.434	D# 6	1837	622.25
C 1		16.352	E 6	1849	659.26
C# 1		17.324	F 6	1860	698.46
D 1		18.354	F# 6	1871	739.99
D# 1		19.445	G 6	1881	783.99
E 1		20.601	G# 6	1890	830.61
F 1		21.826	A 6	1899	880.00
F# 1		23.124	A# 6	1907	932.32
G 1		24.499	B 6	1915	987.77
G# 1		25.956	C 7	1923	1046.5
A 1		27.500	C# 7	1930	1108.7
A# 1		29.135	D 7	1936	1174.7
B 1		30.867	D# 7	1943	1244.5
C 2		32.703	E 7	1949	1318.5
C# 2		34.648	F 7	1954	1396.9
D 2		36.708	F# 7	1959	1480.0
D# 2		38.890	G 7	1964	1568.0
E 2		41.203	G# 7	1969	1661.2
F 2		43.653	A 7	1974	1760.0
F# 2		46.249	A# 7	1978	1864.7
G 2		48.999	B 7	1982	1975.5
G# 2		51.913	C 8	1985	2093.0
A 2		55.000	C# 8	1988	2217.5
A# 2		58.270	D 8	1992	2349.3
B 2		61.735	D# 8	1995	2489.0
C 3	44	65.406	E 8	1998	2637.0
C# 3	156	69.295	F 8	2001	2793.8
D 3	262	73.416	F# 8	2004	2960.0
D# 3	363	77.781	G 8	2006	3136.0
E 3	457	82.406	G# 8	2009	3322.4
F 3	547	87.307	A 8	2011	3520.0
F# 3	631	92.499	A# 8	2013	3729.3
G 3	710	97.998	B 8	2015	3951.1
G# 3	786	103.82	C 9		4186.0
A 3	854	110.00	C# 9		4434.9
A# 3	923	116.54	D 9		4698.6
B 3	986	123.47	D# 9		4978.0
C 4	1046	130.81	E 9		5274.0
C# 4	1102	138.59	F 9		5587.7
D 4	1155	146.83	F# 9		5919.9
D# 4	1205	155.56	G 9		6271.9
E 4	1253	164.81	G# 9		6644.9
F 4	1297	174.61	D# 9		7040.0
F# 4	1339	184.99	E 9		7458.6
G 4	1379	195.99	F 9		7902.1
G# 4	1417	207.65	F# 9		8372.0
A 4	1452	220.00	G 9		8869.8
A# 4	1486	233.08	G# 9		9397.3
B 4	1517	246.94	F 9		9956.1
C 5	1546	261.63	F# 9		10548.1
C# 5	1575	277.18	G 9		11175.3
D 5	1602	293.66	G# 9		11839.8
D# 5	1627	311.13	G# 9		12543.9

Built-in Colour Palettes

Button	BG Colour	OBJ0 Colour	OBJ1 Colour
None	Green & Blue	Red	Red
Up	Brown	Brown	Brown
Up+A	Red	Green	Blue
Up+B	Dark Brown	Brown	Brown
Left	Blue	Red	Green
Left+A	Dark Blue	Red	Brown
Left+B	Grey	Grey	Grey
Down	Yellow, Red, Blue	Yellow, Red, Blue	Yellow, Red, Blue
Down+A	Yellow & Red	Yellow & Red	Yellow & Red
Down+B	Yellow	Blue	Green
Right	Green & Red	Green & Red	Green & Red
Right+A	Green & Blue	Red	Red
Right+B	Reverse	Reverse	Reverse

Hex and Decimal Conversion

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	L M
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	**0**
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	**1**
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	**2**
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	**3**
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	**4**
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	**5**
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	**6**
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	**7**
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	**8**
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	**9**
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	**A**
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	**B**
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	**C**
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	**D**
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	**E**
240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	**F**

Handling VRAM

Vertical VRAM Wrap

```
vram_addr = 0x9800 | (vram_addr & 0x0300) ;
LD A,[vram_addr_MSB] ; get msb of vram addr     (4)
AND $03              ; vram is $9800 to $9BFF   (2)
OR $98               ; add on start of vram     (2)
LD [vram_addr_MSB],A ; store msb of vram addr   (4)
```

Horizontal VRAM Wrap

```
vram_addr = (vram_addr & 0xFFE0) | (vram_addr & 0x1F)
LD A,[vram_addr_LSB] ; get lsb of vram addr     (4)
LD B,A               ; copy lsb of vram addr    (1)
AND $E0              ; mask row start addr      (2)
LD C,A               ; save result              (1)
LD A,B               ; get lsb of vram addr     (1)
AND $1F              ; calculate col offset     (1)
OR C                 ; add row start addr       (1)
LD [vram_addr_LSB],A ; store lsb of vram addr   (4)
```

Col/Row Wrap

```
col = col & 0x1F ;  // row = row & 0x1F ;
LD A,[col]           ; get column (or row)      (4)
AND $1F              ; keep it inside of vram   (2)
LD [col],A           ; store column (or row)    (4)
```

Col & Row To VRAM Addr

```
vram_addr = 0x9800 | (col | ((UWORD)(row) << 5)) ;
LD A,[row]           ; get row                  (4)
SWAP                 ; x 16                     (2)
RLC                  ; x 32                     (2)
LD C,A               ; save result for later    (1)
AND $03              ; calc msb vram row start  (2)
ADD $98              ; add start of vram        (2)
LD B,A               ; set msb of vram ptr      (1)
LD A,$E0             ; Lsb vram row start mask  (2)
AND C                ; calc lsb vram row start  (1)
LD C,A               ; save lsb vram row start  (1)
LD A,[col]           ; get column               (4)
ADD C                ; add lsb vram row start   (1)
LD C,A               ; BCcontains vram addr     (1)
```

OTAKU NO GAMEBOY

3

Please submit all comments/corrections to
otaku@weirdness.com

Powers Of Two

1	2	$0002	17	131072	$20000
2	4	$0004	18	262144	$40000
3	8	$0008	19	524288	$80000
4	16	$0010	20	1048576	$100000
5	32	$0020	21	2097152	$200000
6	64	$0040	22	4194304	$400000
7	128	$0080	23	8388608	$800000
8	256	$0100	24	16777216	$1000000
9	512	$0200	25	33554432	$2000000
10	1024	$0400	26	67108864	$4000000
11	2048	$0800	27	134217728	$8000000
12	4096	$1000	28	268435456	$10000000
13	8172	$2000	29	536870912	$20000000
14	16384	$4000	30	1073741824	$40000000
15	32768	$8000	31	2147483648	$80000000
16	65536	$10000	32	4294967296	$100000000

Two's Complement

0	$00
1	$01
15	$0F
16	$10
17	$11
31	$1F
32	$20
48	$30
64	$40
80	$50
96	$60
112	$70
126	$7E
127	$7F
-128	$80
-127	$81
-126	$82
-112	$90
-96	$A0
-80	$B0
-64	$C0
-48	$D0
-32	$E0
-31	$E1
-16	$F0
-15	$F1
-2	$FE
-1	$FF

ASCII Character Set

LSB \ MSB		0	1	2	3	4	5	6	7
		0000	0001	0010	0011	0100	0101	0110	0111
0	0000	NUL	DLE	SP	0	@	P	`	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	“	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	‘	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	i	y
A	1010	LF	SUB	*	:	J	Z	j	z
B	1011	VT	ESC	+	;	K	[	k	{
C	1100	FF	FS	,	<	L	\	l	|
D	1101	CR	GS	-	=	M	]	m	}
E	1110	SO	RS	.	>	N	^	n	~
F	1111	SI	US	/	?	O	_	o	DEL

Power Consumption

Feature	~ mA
Idle Consumption	55
Audio	15.5
No Halt	3.5
2x CPU	7.5
IR Receive	2
IR Transmit	107
Audio, No Halt, 2x CPU	83
Everything	162

```
            BC contains 16-bit unsigned value
A < const       LD    A,B                 ; get MSB of value
                CP    MSB_of_constant     ; compare with MSB of constant
                JR    NZ,is_greater       ; not equal, test for greater than
                LD    A,C                 ; get LSB of value
                CP    LSB_of_constant     ; compare with LSB of constant
            is_greater:
                JR    NC,not_less_than    ; LSB/MSB not less than, expr not equal
                CALL  condition_true
            not_less_than:
A = const       LD    A,C                 ; get LSB of value
                CP    LSB_of_constant     ; compare with LSB of constant
                JR    NZ,not_equal        ; not equal, condition failed
                LD    A,B                 ; get MSB of value
                CP    MSB_of_constant     ; compare with MSB of constant
                JR    NZ,not_equal        ; LSB/MSB not less than, expr not equal
                CALL  condition_true
            not_equal:
A <= const      LD    A,B
                CP    MSB_of_constant
                JR    NZ,is_less_than
                LD    A,C
                CP    LSB_of_constant
            is_less_than:
                JR    C,not_lt_or_eq
                CALL  condition_true
            not_lt_or_eq:
```