OTAKU NO GAMEBOY

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

Gameboy Crib Sheet
V0.8 99/07/26

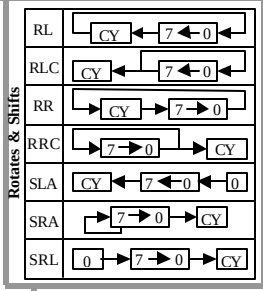
Z80 OPCODE LIST

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)
SUB x	Subtract x from A
XOR x	XOR x to A

Rotates & Shifts



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	JP C,3 JP NZ,yes

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

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

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

Z80 Flags

Z	7	Zero – Set when result of a math op. is zero, or two values match for CP op.
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
C	4	Carry – Set if carry occurred in last math op, or if A is < value for CP op.
X	3	Not Used
X	2	Not Used
X	1	Not Used
X	0	Not Used

OAM RAM Ordering

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

Z80 Registers

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

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
-112	$90
-96	$A0
-80	$B0
-64	$C0
-48	$D0
-32	$E0
-31	$E1
-16	$F0
-15	$F1
-2	$FE
-1	$FF

Video Timings

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

Interrupts

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

Clock Speeds

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

Video Sizes

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

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

ASCII Character Set

MSB / LSB	0 0000	1 0001	2 0010	3 0011	4 0100	5 0101	6 0110	7 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

Hex and Decimal Conversion

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	H/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

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	
AUD3WAVERAM		16 bytes of sound sample	R/W		FF3F

MBC3 Memory Controller Register

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 Contr

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 unused. 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.

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)
```

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
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

Cart Type	ROM	RAM	MBC	MMM0	Battery	Timer	Rumble	Camera	TAMA5	HuC
00	X									
01	X		1							
02	X	X	1							
03	X	X	1		X					
04										
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					
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			
1F								X		
FD									X	
FE										3
FF										1

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

Cart Header

Checksum LSB	014F
Checksum MSB	014E
Complement	014D
Mask ROM Ver	014C
Old Maker Code	014B
Destination Code	014A
Cart 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

RAM Size

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

ROM Sizes

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

OTAKU NO GAMEBOY

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