CPU Specifications

OVERVIEW

The BGC-16 is a 24-bit CPU architecture with 8 general purpose registers and with 15 banks each with 16 megabytes or 4 gigabytes of data,

Name: BGC-16
Data bus: 24-bits.
Address bus: 24-bits.

The BGC-16 CPU and a 16-bit CPU. The CPU have 16-bits of data and 24/32 bits of address meaning the CPU can address up to 16 megabytes using 24 bits of address, and 4 gigabytes with 32 bits of address. The CPU also have 45 instructions with 8 different argument modes.

INSTRUCTION SET

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
0	MOV	CMP	PUSH	POP	CALL	RET											0
1	JMP	JZ	JNZ	JS	JNS	JE	JNE	JL	JG	JLE	JGE						1
2													IN	OUT	SEF	CLF	2
3	ADD	SUB	MUL	DIV	AND	OR	NOR	NOT	SHL	SHR	ROL	ROR	INC	DEC	NEG	AVG	3
4	EXP	SQRT															4
5																	5
6																	6
7																	7
8																	8
9																	9
A																	A
B																	B
C																	C
D																	D
E																	E
F												PUSHR	POPR	INT	BRK	HELT	F
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	

MOV destination source		
CMP operand1 operand2		
PUSH source		
POP destination		
CALL address		

RET operand1
JMP address
JZ address
JNZ address
JS address
JNS address
JE address
JNE address
JL address
JG address
JLE address
JGE address
IN port destination
OUT port source
SEF flag
CLF flag
ADD destination source
SUB destination source
MUL destination source
DIV destination source
AND destination source
OR destination source
NOR destination source
XOR destination source
NOT destination
SHL destination operand1
SHR destination operand1
ROL destination operand1
ROR destination operand1
INC destination
DEC destination
NEG destination
AVG destination operand1
EXP destination operand1
SQRT destination
PUSHR
POPR
INT INTERRUPT_ROUTINE
BRK
HALT

MOV moves the value from the source into the destination
CMP Cmpare operand1 and operand2 and sets the flags in the flags register
PUSH Pushs the source onto the stack and increments the SP
POP Decrements the SP and pops the current value into the destination
CALL
RET
JMP jumps to the specified address
JZ jumps to the specified address if the zero flag is set
JNZ jumps to the specified address if the zero flag is cleared
JS jumps to the specified address if the signed flag is set
JNS jumps to the specified address if the signed flag is cleared
JE jumps to the specified address if the equal flag is set
JNE jumps to the specified address if the equal flag is cleared
JL jumps to the specified address if the less flag is set
JG jumps to the specified address if the less flag is cleared
JLE jumps to the specified address if the equal flag or the less flag is set
JGE jumps to the specified address if the equal flag is set or the less flag is cleared
IN
OUT
SEF
CLF
ADD Adds the values of the source and the destination and stores the value in destination.
SUB Subtracts the values of the source and the destination and stores the value in destination.
MUL Multiplies the values of the source and the destination and stores the value in destination.
DIV Divides the values of the source and the destination and stores the value in destination.
AND Performs a bitwise AND operation between the destination and source and stores the value in destination.
OR Performs a bitwise OR operation between the destination and source and stores the value in destination.
NOR Performs a bitwise NOR operation between the destination and source and stores the value in destination.
XOR Performs a bitwise XOR operation between the destination and source and stores the value in destination.
NOT Performs a bitwise NOT operation on the destination and stores the value in destination.
SHL Shifts the bits in the destination to the left by the number of bits specified by operand1. The {SHIFT_FLAG} is the overflowing bit, and the next bit is zero.
SHR Shifts the bits in the destination to the right by the number of bits specified by operand1. The {SHIFT_FLAG} is the overflowing bit, and the next bit is zero.
ROL Rotates the bits in the destination to the left by the number of bits specified by operand1. The {SHIFT_FLAG} is used as the next bit and the overflowing bit.
ROR Rotates the bits in the destination to the right by the number of bits specified by operand1. The {SHIFT_FLAG} is used as the next bit and the overflowing bit.
INC Increments the value at the destination by 1.
DEC Decrements the value at the destination by 1.
NEG Sets/clears the signed bit of the destination.
AVG Calculates the average of the values at the destination and operand1 and stores the value in destination.
EXP Raises the value at the destination to the power of the value specified by operand1.
SQRT Calculates the square root of destination.
PUSHR
POPR
INT Generates a interrupt routine (more in the INTERRUPTS)
BRK Generates a software interrupt (more in the INTERRUPTS)
HALT Stops the CPU

INSTRUCTION LAYOUT

XXXX_XXXX_AAAA_BBBB

X = Op code
A = argument1
B = argument2

ARGUMENT MODE

immediate

The immediate argument mode is a way to have the value in ROM, the mode is usually found together with a register or memory location, the mode only takes up 1 word of the overall instruction.

address

The addressing argument mode is a way to specific a memory location in RAM/ROM, the mode is usually found together with some of the other argument modes you will mostly find it together with the immediate mode

register

register address

indexed immediate

indexed register

indexed register immediate

CPU PINS

IRQ: interrupt request

INTERRUPTS

interrupts can be triggered be software or hardware specified by the interrupt location port

CACHES

INSTRUCTION CACHE
size: 64 kb
type: read only
notes: A

PORTS

ports are a memory mapped IO

0x00: Cartridge space
Here the cartridge is located in the ports.
the layout of the cartridge is as following
[0x000000 - 0xFFFFFD] Cartridge disk (~16 MB)
[0xFFFFFE - 0xFFFFFE] Non-markable interrupt vector (1 word)
[0xFFFFFF - 0xFFFFFF] interrupt request vector (1 word)

note:
The data from the cartridge is loaded into the 'INSTRUCTION CACHE' where the program can be executed by the CPU itself.

0x20: interrupt location port
if there is an interrupt this port will show where that interrupt came from.

MEMORY LAYOUT

BANK 0x0:
[0x000000 - 0x00FFFF] INSTRUCTION CACHE
[0x010000 - 0xFFFFFF] GENERAL PURPOSE RAM

BANK 0x1:
[0x000000 - 0x0000FF] Ports
[0x000100 - 0xFFFFFF] GENERAL PURPOSE RAM

BANK 0xF:

REGISTERS

A: (AH + AL) 16 bit general purpose register
B: (BH + BL) 16 bit general purpose register
AB: 32 bit general purpose register
The AB register is the A register and the B register combined
where A is the high part and B is the low part
C: (CH + CL) 16 bit general purpose register
D: (CH + DL) 16 bit general purpose register
CD: 32 bit general purpose register
The CD register is the C register and the D register combined
where C is the high part and D is the low part

HL: (H + L) 32 bit general purpose register
The HL register is a combination of the H and L registers

X: (XH + XL) 16 bit index register
Y: (YH + YL) 16 bit index register

PC: (PCH + PCL) 32 bit program counter
The (P)rogram (C)ounter will start at address 0 and increase through the program

MB: 4 bit memory bank register

Flags: 16: bit (F)lags register
0x0001: zero flag
0x0002: equals flag
0x0004: signed
0x0008: carry
0x0010: overflow
0x0020:
0x0040:
0x0080:
0x0100: erorr
0x0200: interrupt enable
0x0400:
0x0800:
0x1000:
0x2000:
0x4000:
0x8000: