THE MSTC MACHINE LANGUAGE SPECIFICATION.

PART-I : REGISTERS.

The MSTC microprocessor has 16 general purpose registers, each register being 32 bit wide. They are named as r_0 to r15. The least significant bytes are named as r0q to r15q. The two least significant bytes are named from r0h [r-zero-h] to r15h.

32 — 32-bit wide — 1	
	r0
	r1
	r2
	r3
	r4
	r5
	r6
	r7
	r8
	r9
	r10
	r11
	r12
	r13
	r14
	r15

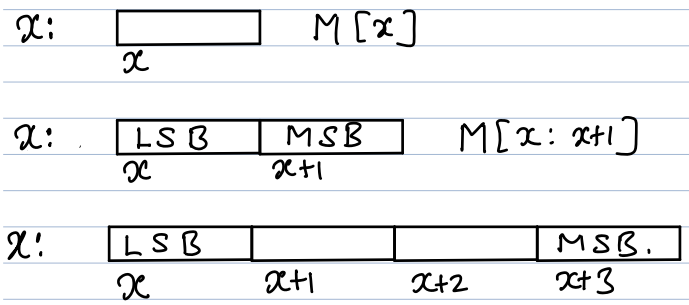
Byte-4	Byte-3	Byte-2	Byte-1	rk (where $0 \leq k \leq 15$)

Byte-2	Byte-1	rkh	Byte-1	rkq (where $0 \leq k \leq 15$)

PART-II : The MEMORY : The MSTC micro-processor has 32-bit address space. The maximum amount of memory that can be attached therefore is 4Gib. The MSTC machine under consideration will have the maximum possible memory attached viz. 4GB.

Let x be an address of a byte in memory. Notation $M[x]$

denotes one byte data element at address x. Notation $M[x:x+1]$ denotes two byte data element at addresses x and $x+1$. The byte at address x will be the least significant byte and the byte at $x+1$ denotes the most significant byte. Notation $M[x:x+3]$ denotes four byte data element at addresses, $x, x+1, x+2$ and $x+3$. The byte at address x is the least significant byte and the byte at $x+3$, being the most significant.



LSB = Least Significant Byte.

MSB = Most Significant Byte.

PART-III : The MSTC PROCESSOR INSTRUCTION FORMAT : GENERAL.

1) Unary operators:

Opcode	Operand size	Operand type	Operand
1 byte	1 byte	1 byte	1 to 4 bytes

2) Binary operator

Opcode	Operand size	Operand type -1	Operand -1	Operand type-2	Operand 2
1 byte	1 byte	1 byte	1 to 4 bytes	1 byte	1 to 4 bytes

(a) If the operand type is constant then the operand field will be the constant data number.

If the operand field is register then the operand field will be the register's number

If the operand field is memory then the operand field will be the address of starting / base memory location.

(b)

i) Constant operand. Size 1-byte.

1	constant	number
Size	type.	$0 \leq number < 2^8$

ii) constant operand two bytes

2	constant	number
size	type	$0 \leq number < 2^{16}$.

iii) Constant operand. Size = 4 bytes.

4	constant	nu	mb	e	r
size	type				

$0 \leq number < 2^{32}$

iv) Register operand: Size = 1 byte.

1	register	reg #
Size	type	

$0 \leq reg\# < 16$.

[the LSB of specified register will be used].

v) Register operand: size = 2 bytes.

2	register	reg #
Size	type	

$0 \leq reg\# < 16$.

[the first two bytes of specified register will be used].

vi) Register operand. Size = 4 bytes.

4	register	reg #
Size	type	

$0 \leq reg\# < 16$.

[the full version of specified

register will be used].

vi) Memory operand. Size = 1 byte.

1	memory	a d	d r	e s	s
Size	type	$0 \leq$ address $< 4GB$			

Let x = address. $M[x]$ will be accessed

vii) Memory operand. Size = 2 bytes.

2	memory	a d	d r	e s	s
Size	type	$0 \leq$ address $< 4GB$			

Let x = address. $M[x:x+1]$ will be accessed.

viii) Memory operand. Size = 4 bytes

4	memory	a d	d r	e s	s
Size	type	$0 \leq$ address $< 4GB$			

Let x = address. $M[x:x+3]$ will be accessed.

PART-IV : THE INSTRUCTION SET.

(1) Data Movement Instruction :

The Move Instruction : The move instruction is used to transfer 1 byte, 2 byte or 4 byte of data from memory to register, or from register to memory, or from one regsiter to another. Additionally, the move instruction can be used to move a constant to a register or a memory location.

The General Format of the move instruction.

mov(size specifier) source operand, destination operand

Size specifiers: b = byte = for 1 byte data transfer.

w = word = for 2 byte data transfer.

l = long = for 4 byte data transfer

for one byte data transfer,

mov b source, destination.

for two byte data transfer

mov w source, destination.

for four byte data transfer

mov l source, destination.

Description of the operation: After the move instruction the data at the source operand is copied into the destination operand. The data at the destination operand just before the move instruction is overwritten, the data at the source operand however remains unchanged.

Rules for combining the source and the destination operands.

1) the size of both operands must be identical.

   i.e. both the source and destination operands must be

   8 bit = 1 byte wide OR

   16 bit = 2 bytes wide OR

   32 bit = 4 bytes wide.

2) The destination operand cannot be constant.

   In other words only memory or register can be specified at destination place.

3) Both operands cannot simultaneously be memory operands.

Combining above rules, we get five valid combinations.

(a) Constant to memory.

movb constant-8, memory-8

mouw constant-16, memory-16.

mou2 constant-32, memory-32.

(b) Constant to register.

moub constant-8, register-8

mouw constant-16, register-16.

mou2 constant-32, register-32.

(c) Memory to register.

moub memory-8, register-8

mouw memory-16, register-16

mou2 memory-32, register-32.

(d) register to memory

moub register-8, memory-8

mouw register-16, memory-16

mou2 register-32, memory-32

(e) register to register

moub register-8, register-8

mouw register-16, register-16

mou2 register-32, register-32

ENCODING OF OPERAND TYPE:

1) Constant operand = $(0)_{10} = (0)_2$

2) Register operand = $(1)_{10} = (1)_2$

3) Memory operand = $(2)_{10} = (10)_2$.

THE MACHINE LANGUAGE FORMAT OF MOVE

OPCODE	SIZE	OPND Type-1	OPND-1	OPND Type-2	OPND-2
1 byte	1 byte	1 byte	1-4 bytes	1 byte	1-4 bytes
		SOURCE OPERAND		DESTINATION OPERAND	

THE OPCODE OF MOV INSTRUCTION $= (8)_{10} = (1000)_2$

(2) THE ADD INSTRUCTION.

The General Format of add instruction.

add(size specifier) source operand, destination operand.

Description of the add operation.

destination $\longleftarrow$ source + destination.

Data of source and destination just before executing the add instruction is added and stored in the destination. The destination operand is overwritten by the result of addition. Source operand remains unchanged before and after the add instruction.

Rules of combining operands is same as that of the mov instruction.

add(b/w/l) constant-8/16/32, memory/register-8/16/32
add(b/w/l) memory-8/16/32, register-8/16/32
add(b/w/l) register-8/16/32, memory/register-8/16/32

THE MACHINE LANGUAGE FORMAT OF ADD

OPCODE	SIZE	OPND Type-1	OPND-1	OPND Type-2	OPND-2
1 byte	1 byte	1 byte	1-4 bytes	1 byte	1-4 bytes
		SOURCE OPERAND		DESTINATION OPERAND	

THE OPCODE OF ADD INSTRUCTION $= (16)_{10} = (10000)_2$

(2) THE SUB INSTRUCTION.

The General Format of add instruction.

Sub.(size specifier) source operand, destination operand

Description of the add operation.

destination $\longleftarrow$ destination − source.

The data at the source operand just before the sub instruction is subtracted from the data at destination operand and the result of subtraction is stored in the destination. Source operand remains unchanged before and after the sub instruction.

Rules of combining operands is same as that of the mov instruction.

Sub (b/w/l) constant-8/16/32, memory/register-8/16/32
Sub (b/w/l) memory-8/16/32, register-8/16/32
Sub (b/w/l) register-8/16/32, memory/register-8/16/32

THE MACHINE LANGUAGE FORMAT OF SUB

OPCODE	SIZE	OPND Type-1	OPND-1	OPND Type-2	OPND-2

1 byte	1 byte	1 byte	1-4 bytes	1 byte	1-4 bytes
		SOURCE OPERAND		DESTINATION OPERAND	

THE OPCODE OF SUB INSTRUCTION $= (17)_{10} = (10001)_2$