

## Required MIPS Assembly Instructions Set

<b>Arithmetic Instructions</b>			
<i>Instruction</i>	<i>Example</i>	<i>Meaning</i>	<i>Comments</i>
<b>add</b>	add \$1,\$2,\$3	$\$1 = \$2 + \$3$	
<b>subtract</b>	sub \$1,\$2,\$3	$\$1 = \$2 - \$3$	
<b>add immediate</b>	addi \$1,\$2,100	$\$1 = \$2 + 100$	
<b>Multiply (without overflow)</b>	mul \$1,\$2,\$3	$\$1 = \$2 * \$3$	Result is only 32 bits!

<b>Logical Instructions</b>			
<i>Instruction</i>	<i>Example</i>	<i>Meaning</i>	<i>Comments</i>
<b>and</b>	and \$1,\$2,\$3	$\$1 = \$2 \& \$3$	Bitwise AND
<b>or</b>	or \$1,\$2,\$3	$\$1 = \$2   \$3$	Bitwise OR
<b>xor</b>	xor \$1,\$2,\$3	$\$1 = \$2 \wedge \$3$	Bitwise XOR
<b>and immediate</b>	andi \$1,\$2,100	$\$1 = \$2 \& 100$	Bitwise AND with immediate value
<b>or immediate</b>	ori \$1,\$2,100	$\$1 = \$2   100$	Bitwise OR with immediate value
<b>xor immediate</b>	xori \$1,\$2,100	$\$1 = \$2 \wedge 100$	Bitwise XOR with immediate value
<b>shift left logical</b>	sll \$1,\$2,10	$\$1 = \$2 \ll 10$	Shift left by constant number of bits
<b>shift right logical</b>	srl \$1,\$2,10	$\$1 = \$2 \gg 10$	Shift right by constant number of bits

<b>Data Transfer Instructions</b>			
<i>Instruction</i>	<i>Example</i>	<i>Meaning</i>	<i>Comments</i>
<b>move</b>	move \$1,\$2	$\$1 = \$2$	<i>Pseudo-instruction</i> (provided by MARS Assembler, not processor!) Copy from register to register.
<b>load word</b>	Lw \$1,100(\$2)	$\$1 = \text{Memory}[\$2 + 100]$	
<b>store word</b>	Sw \$1,100(\$2)	$\text{Memory}[\$2 + 100] = \$1$	
<b>load upper immediate</b>	lui \$1,100	$\$1 = 100 \times 2^{16}$	Load constant into upper 16 bits. Lower 16 bits are set to zero.

<b>Conditional Branch Instructions</b>			
<i>Instruction</i>	<i>Example</i>	<i>Meaning</i>	<i>Comments</i>
<b>branch on equal</b>	Beq \$1,\$2,100	if( $\$1 == \$2$ ) go to PC+4+100	Test if registers are equal
<b>branch on not equal</b>	Bne \$1,\$2,100	if( $\$1 \neq \$2$ ) go to PC+4+100	Test if registers are not equal

<b>Comparison Instructions</b>			
<i>Instruction</i>	<i>Example</i>	<i>Meaning</i>	<i>Comments</i>
<b>set on less than</b>	slt \$1,\$2,\$3	if( $\$2 < \$3$ ) $\$1 = 1$ ; else $\$1 = 0$	Test if less than. If true, set \$1 to 1. Otherwise, set \$1 to 0.
<b>set on less than immediate</b>	Slti \$1,\$2,100	if( $\$2 < 100$ ) $\$1 = 1$ ; else $\$1 = 0$	Test if less than. If true, set \$1 to 1. Otherwise, set \$1 to 0.

<b>Unconditional Jump Instructions</b>			
<b><i>Instruction</i></b>	<b><i>Example</i></b>	<b><i>Meaning</i></b>	<b><i>Comments</i></b>
<b>jump</b>	j 1000	go to address 1000	Jump to target address
<b>jump register</b>	jr \$ra	go to return address stored in \$ra	procedure return
<b>jump and link</b>	jal 1000	\$ra=PC+4; go to procedure call which starts in address 1000	Use when making procedure call. This saves the return address in \$ra