Compiler Register Usage

Register	Name	Function
R0	zero	Always contains 0
R1	at	Assembler temporary
R2-R3	v0-v1	Function return value
R4-R7	a0-a3	Function parameters
R8-R15	t0-t7	Function temporary values
R16-R23	s0-s7	Saved registers across function calls
R24-R25	t8-t9	Function temporary values
R26-R27	k0-k1	Reserved for interrupt handler
R28	gp	Global pointer
R29	sp	Stack Pointer
R30	s8	Saved register across function calls
R31	ra	Return address from function call
HI-LO	lo-hi	Multiplication/division results
PC	Program Counter	Points at 8 bytes past current instruction
EPC	epc	Exception program counter return address

Branch Delay Slot

There is one branch delay slot. This means that the instuction after a branch is always executed before the CPU decides to take the branch or not.

Assembly Example

Also see opcodes.asm which tests all of the opcodes.

```
   LUI   $4, 0x1234          #i = 0x12345678;
   ORI   $4, $4, 0x5678
   BLEZ  $4, NoAdd           #if (i > 0) i += 9;
   NOP                       #Branch Delay Slot
   ADDIU $4, $4, 9
NoAdd:
```

Opcodes

Opcode	Name	Action	Opcode bitfields

Arithmetic Logic Unit								
ADD rd,rs,rt	Add	rd=rs+rt	000000	rs	rt	rd	00000	100000
ADDI rt,rs,imm	Add Immediate	rt=rs+imm	001000	rs	rt	imm		
ADDIU rt,rs,imm	Add Immediate Unsigned	rt=rs+imm	001001	rs	rt	imm		
ADDU rd,rs,rt	Add Unsigned	rd=rs+rt	000000	rs	rt	rd	00000	100001
AND rd,rs,rt	And	rd=rs&rt	000000	rs	rt	rd	00000	100100
ANDI rt,rs,imm	And Immediate	rt=rs&imm	001100	rs	rt	imm		
LUI rt,imm	Load Upper Immediate	rt=imm<<16	001111	rs	rt	imm		
NOR rd,rs,rt	Nor	rd=~(rs|rt)	000000	rs	rt	rd	00000	100111
OR rd,rs,rt	Or	rd=rs|rt	000000	rs	rt	rd	00000	100101
ORI rt,rs,imm	Or Immediate	rt=rs|imm	001101	rs	rt	imm		
SLT rd,rs,rt	Set On Less Than	rd=rs<rt	000000	rs	rt	rd	00000	101010
SLTI rt,rs,imm	Set On Less Than Immediate	rt=rs<imm	001010	rs	rt	imm		
SLTIU rt,rs,imm	Set On < Immediate Unsigned	rt=rs<imm	001011	rs	rt	imm		
SLTU rd,rs,rt	Set On Less Than Unsigned	rd=rs<rt	000000	rs	rt	rd	00000	101011

SUB rd,rs,rt	Subtract	rd=rs-rt	000000	rs	rt	rd	00000	100010
SUBU rd,rs,rt	Subtract Unsigned	rd=rs-rt	000000	rs	rt	rd	00000	100011
XOR rd,rs,rt	Exclusive Or	rd=rs^rt	000000	rs	rt	rd	00000	100110
XORI rt,rs,imm	Exclusive Or Immediate	rt=rs^imm	001110	rs	rt	imm		
Shifter								
SLL rd,rt,sa	Shift Left Logical	rd=rt<<sa	000000	rs	rt	rd	sa	000000
SLLV rd,rt,rs	Shift Left Logical Variable	rd=rt<<rs	000000	rs	rt	rd	00000	000100
SRA rd,rt,sa	Shift Right Arithmetic	rd=rt>>sa	000000	00000	rt	rd	sa	000011
SRAV rd,rt,rs	Shift Right Arithmetic Variable	rd=rt>>rs	000000	rs	rt	rd	00000	000111
SRL rd,rt,sa	Shift Right Logical	rd=rt>>sa	000000	rs	rt	rd	sa	000010
SRLV rd,rt,rs	Shift Right Logical Variable	rd=rt>>rs	000000	rs	rt	rd	00000	000110
Multiply								
DIV rs,rt	Divide	HI=rs%rt; LO=rs/rt	000000	rs	rt	0000000000		011010
DIVU rs,rt	Divide Unsigned	HI=rs%rt; LO=rs/rt	000000	rs	rt	0000000000		011011
MFHI rd	Move From HI	rd=HI	000000	0000000000		rd	00000	010000
MFLO rd	Move From LO	rd=LO	000000	0000000000		rd	00000	010010
MTHI rs	Move To HI	HI=rs	000000	rs	00000000000000			010001

MTLO rs	Move To LO	LO=rs	000000	rs	0000000000000 00		010011
MULT rs,rt	Multiply	HI,LO=rs*rt	000000	rs	rt	0000000000	011000
MULTU rs,rt	Multiply Unsigned	HI,LO=rs*rt	000000	rs	rt	0000000000	011001
Branch							
BEQ rs,rt,offset	Branch On Equal	if(rs==rt) pc+=offset*4	000100	rs	rt	offset	
BGEZ rs,offset	Branch On >= 0	if(rs>=0) pc+=offset*4	000001	rs	00001	offset	
BGEZAL rs,offset	Branch On >= 0 And Link	r31=pc; if(rs>=0) pc+=offset*4	000001	rs	10001	offset	
BGTZ rs,offset	Branch On > 0	if(rs>0) pc+=offset*4	000111	rs	00000	offset	
BLEZ rs,offset	Branch On	if(rs<=0) pc+=offset*4	000110	rs	00000	offset	
BLTZ rs,offset	Branch On < 0	if(rs<0) pc+=offset*4	000001	rs	00000	offset	
BLTZAL rs,offset	Branch On < 0 And Link	r31=pc; if(rs<0) pc+=offset*4	000001	rs	10000	offset	
BNE rs,rt,offset	Branch On Not Equal	if(rs!=rt) pc+=offset*4	000101	rs	rt	offset	
BREAK	Breakpoint	epc=pc; pc=0x3c	000000	code			001101
J target	Jump	pc=pc_upper|(target<<2)	000010	target			
JAL target	Jump And Link	r31=pc; pc=target<<2	000011	target			
JALR rs	Jump And Link Register	rd=pc; pc=rs	000000	rs	00000	rd	00000	001001
JR rs	Jump Register	pc=rs	000000	rs	0000000000000 00			001000
MFC0 rt,rd	Move From	rt=CPR[0,rd]	010000	00000	rt	rd	00000000000	

	Coprocessor						
MTC0 rt,rd	Move To Coprocessor	CPR[0,rd]=rt	010000	00100	rt	rd	00000000000
SYSCALL	System Call	epc=pc; pc=0x3c	000000	00000000000000000000			001100
Memory Access							
LB rt,offset(rs)	Load Byte	rt=*(char*)(offset+rs)	100000	rs	rt	offset	
LBU rt,offset(rs)	Load Byte Unsigned	rt=*(Uchar*)(offset+rs)	100100	rs	rt	offset	
LH rt,offset(rs)	Load Halfword	rt=*(short*)(offset+rs)	100001	rs	rt	offset	
LBU rt,offset(rs)	Load Halfword Unsigned	rt=*(Ushort*)(offset+rs)	100101	rs	rt	offset	
LW rt,offset(rs)	Load Word	rt=*(int*)(offset+rs)	100011	rs	rt	offset	
SB rt,offset(rs)	Store Byte	*(char*)(offset+rs)=rt	101000	rs	rt	offset	
SH rt,offset(rs)	Store Halfword	*(short*)(offset+rs)=rt	101001	rs	rt	offset	
SW rt,offset(rs)	Store Word	*(int*)(offset+rs)=rt	101011	rs	rt	offset	