Reference Card

add	rd, rs	Add	rd = rd + rs	R 0 / 000
dec	rd, rs	Decrement	rd = rs - 1	R 0 / 001
and	rd, rs	And	If (rd==rs) rd = rs & 0x0F Else rd = rs & 0x01	R 0 / 010
xor	rd, rs	Xor	rd = rd ^ rs	R 0 / 011
sdm	rd, rs	Square and Drop Middle 8-bits	rd = DM_8(rs^2)	R 0 / 100
srl	rd, rs	Shift Right Logical	rd = rd >> rs + 1	R 0 / 101
sb	rd, rs	Store Byte	M[rs] = rd	R 0 / 110
lb	rd, rs	Load Byte	rd = M[rs]	R 0 / 111
addi	rd, imm	Add Immediate	rd = rd + imm	I 0
bnez	rd, imm	Branch Not Equal Zero	if (rd !=0), pc = ([pc + 1] + imm)	I 1

old Mips

	b7	b6:b5	b4:b3	b2:b0
R-Type:	op	rd	rs	func
	b7	b6:b5	b4:b1	b0
I-Type:	op	rd	imm	func

new Mips

	b7:b6	b5:b4	b3:b1	b0
R-Type:	rd	rs	func	op
	b7:b6	b5:b2	b1	b0
I-Type:	rd	imm	func	op

old Python

	[0]	[1:3]	[3:5]	[5:8]
R-Type:	op	rd	rs	func
	[0]	[1:3]	[3:7]	[7]
I-Type:	op	rd	imm	func

new Python

	[0:2]	[2:4]	[4:7]	[7]
R-Type:	rd	rs	func	op
	[0:2]	[2:6]	[6]	[7]
I-Type:	rd	imm	func	op

Control Signals

	MemToReg	MemWrite	BnD	ALUControl	ALUSrc	RegWrite
add	0	0	xx	0010	0	1
dec	0	0	10	1010	0	1
and	0	0	00	0000	0	1
xor	0	0	00	0001	0	1
sdm	0	0	00	0101	0	1
srl	0	0	00	0111	0	1
sb	0	1	00	0100	0	0
lb	1	0	00	0100	0	1
addi	0	0	00	0010	1	1
bnez	x	0	01	1010	1	0