CPU Architecture

Digital System Design with VHDL

(Simple RISC Multi-Cycle CPU design)

1. Aim of the Task:

- System design using concurrent and sequential logic principles using advanced Simulation methods.
- Controller design based on methodology of Control and Datapath separation.
- Performing and understanding functional validation and verification of an architecture design.
- Proper analysis and understanding of architecture design.

2. Assignment definition:

In this task, you are required to design a controller-based processing machine as a Multi-cycle CPU to run a given program code. The preparation material for this lab has been learned in the prerequisites course “Central Processing Unit Architecture - theory” until lecture five and includes in addition to the self-leaning material given on a subjects of FSM methodology design and advanced functional verification.

3. Controller-based system:

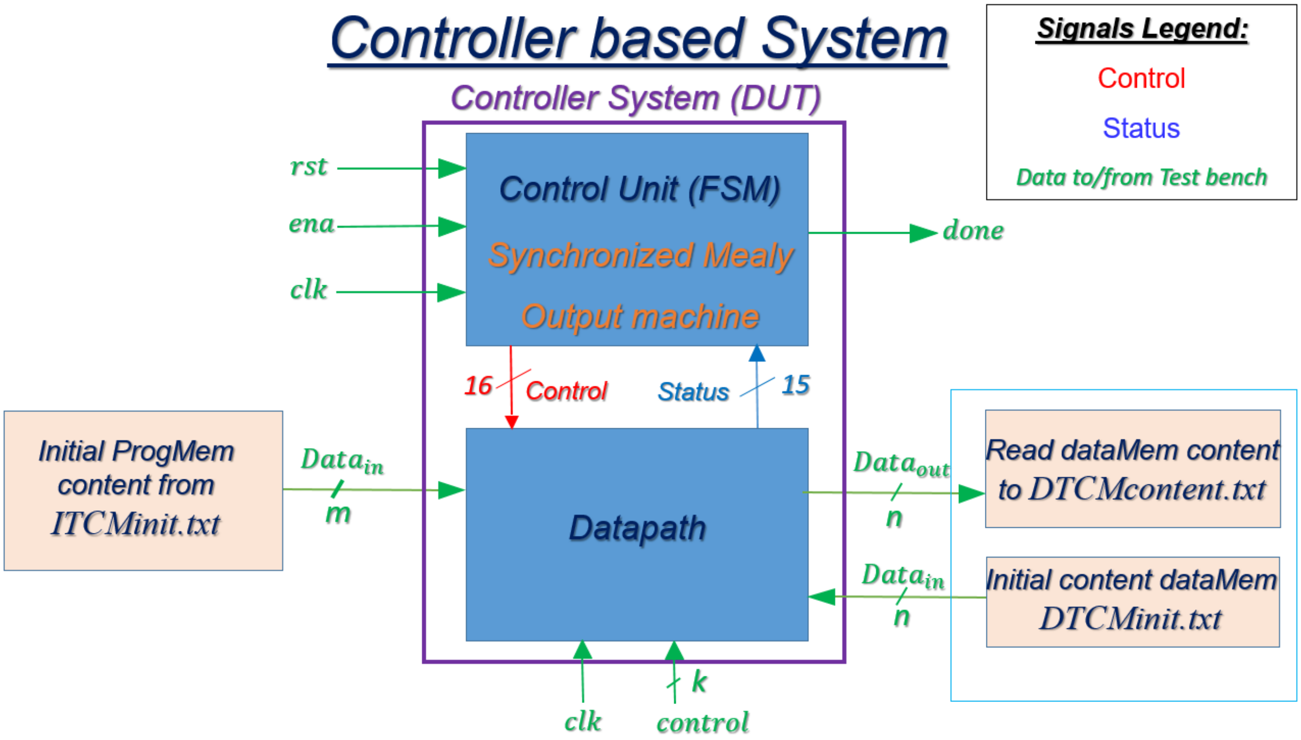


Figure 1: Overall DUT structure

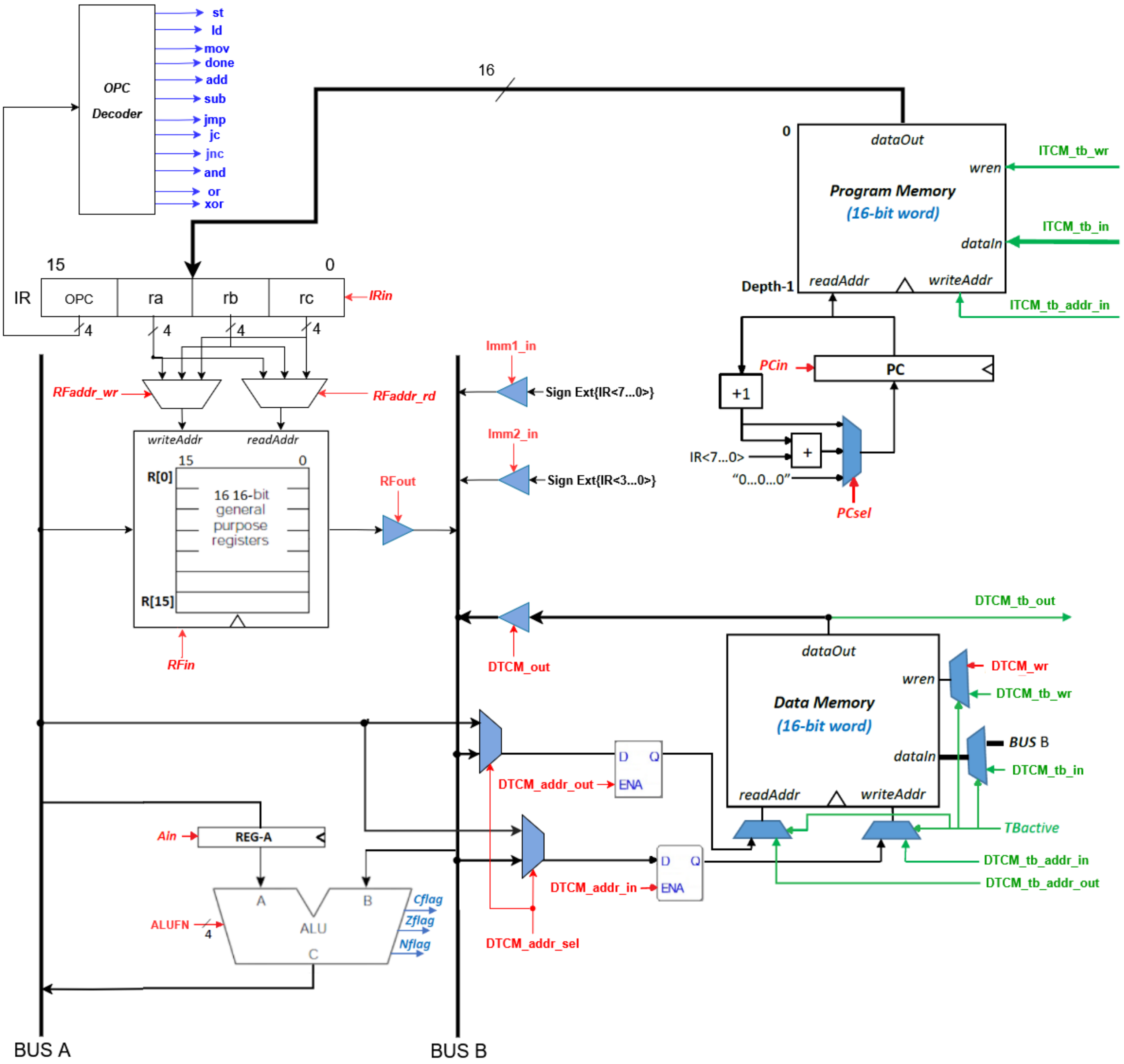


Figure 2: SRMC-I Datapath structure of a 2-BUS multicycle CPU

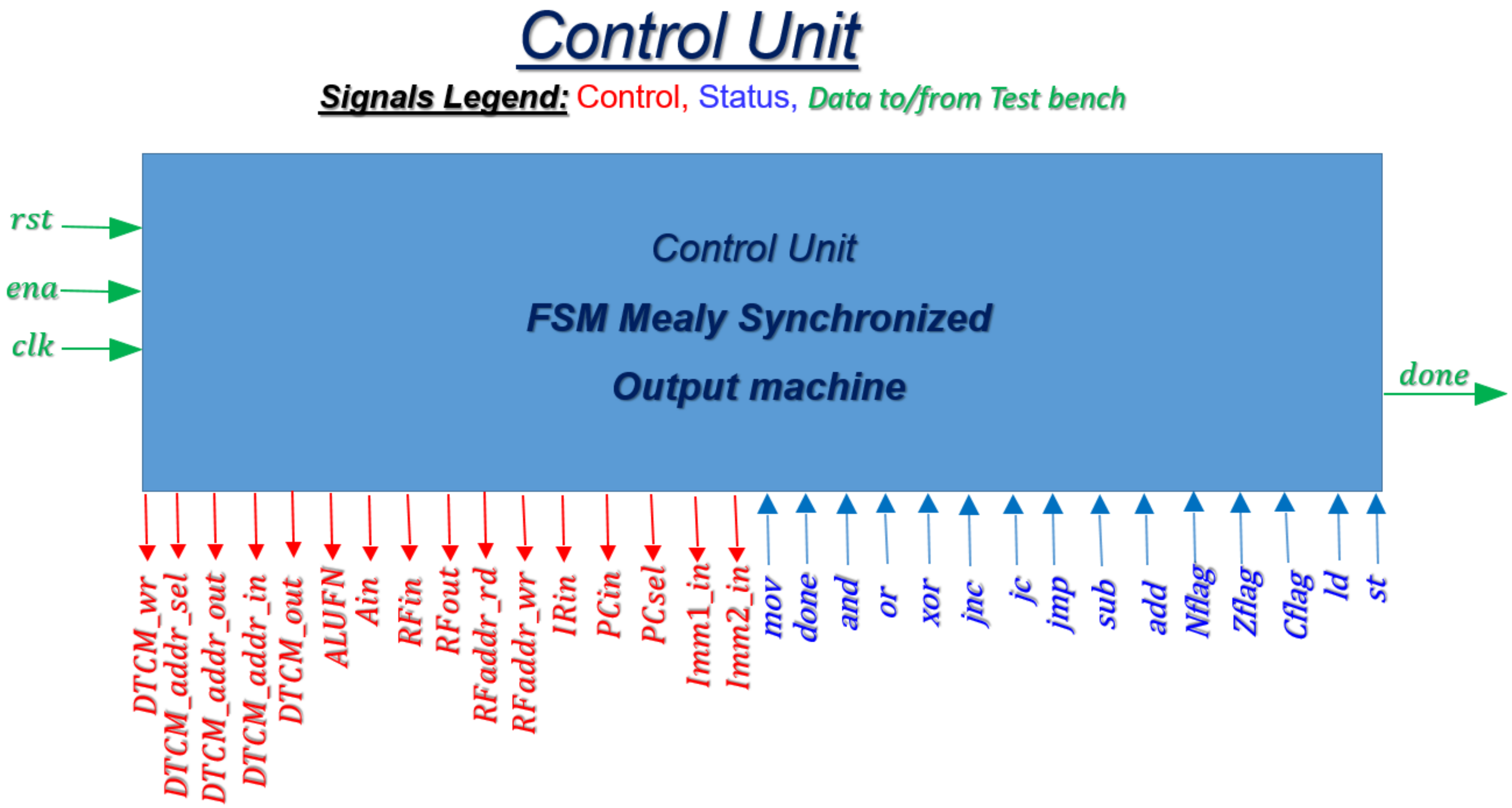
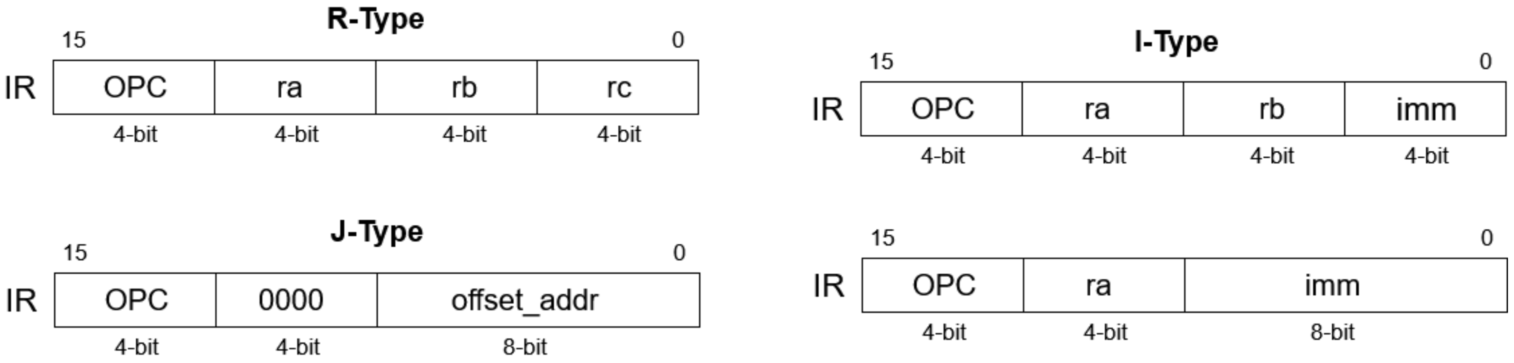


Figure 3: SRMC-I Control unit module



R-Type

	15			0
IR	OPC	ra	rb	rc
	4-bit	4-bit	4-bit	4-bit

I-Type

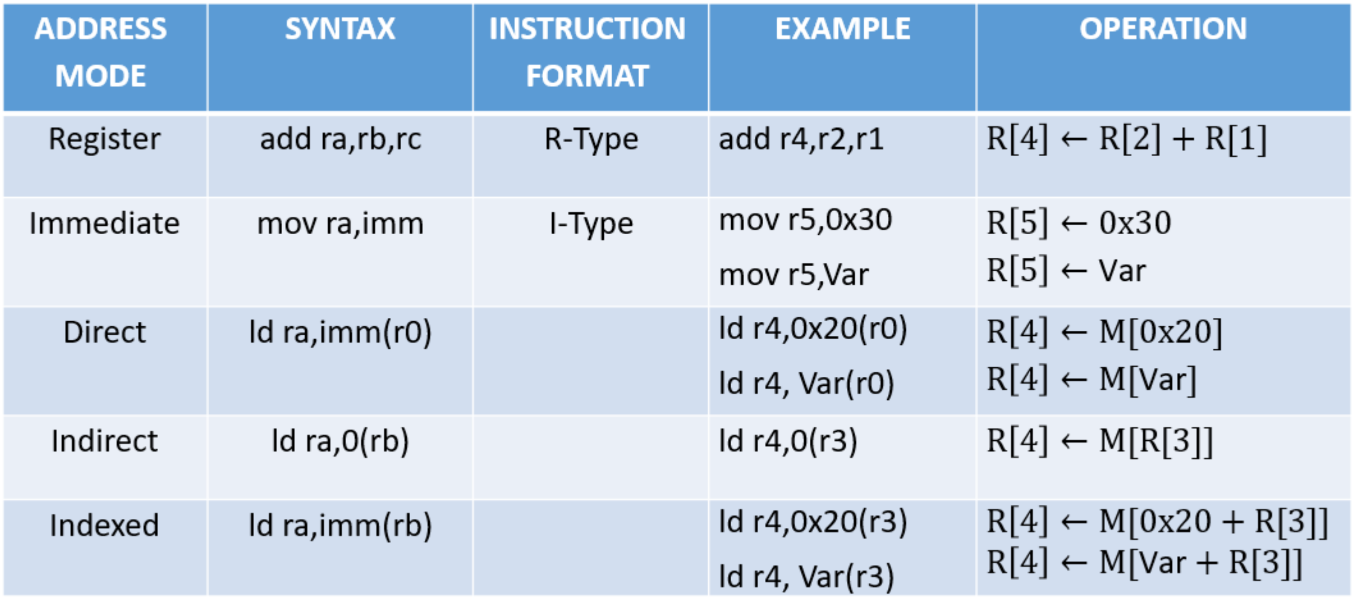
	15			0
IR	OPC	ra	rb	imm
	4-bit	4-bit	4-bit	4-bit

J-Type

	15		0
IR	OPC	0000	offset_addr
	4-bit	4-bit	8-bit

	15		0
IR	OPC	ra	imm
	4-bit	4-bit	8-bit

Figure 4: SRMC-I Instruction Formats subdivided into three types



ADDRESS MODE	SYNTAX	INSTRUCTION FORMAT	EXAMPLE	OPERATION
Register	add ra,rb,rc	R-Type	add r4,r2,r1	R[4] ← R[2] + R[1]

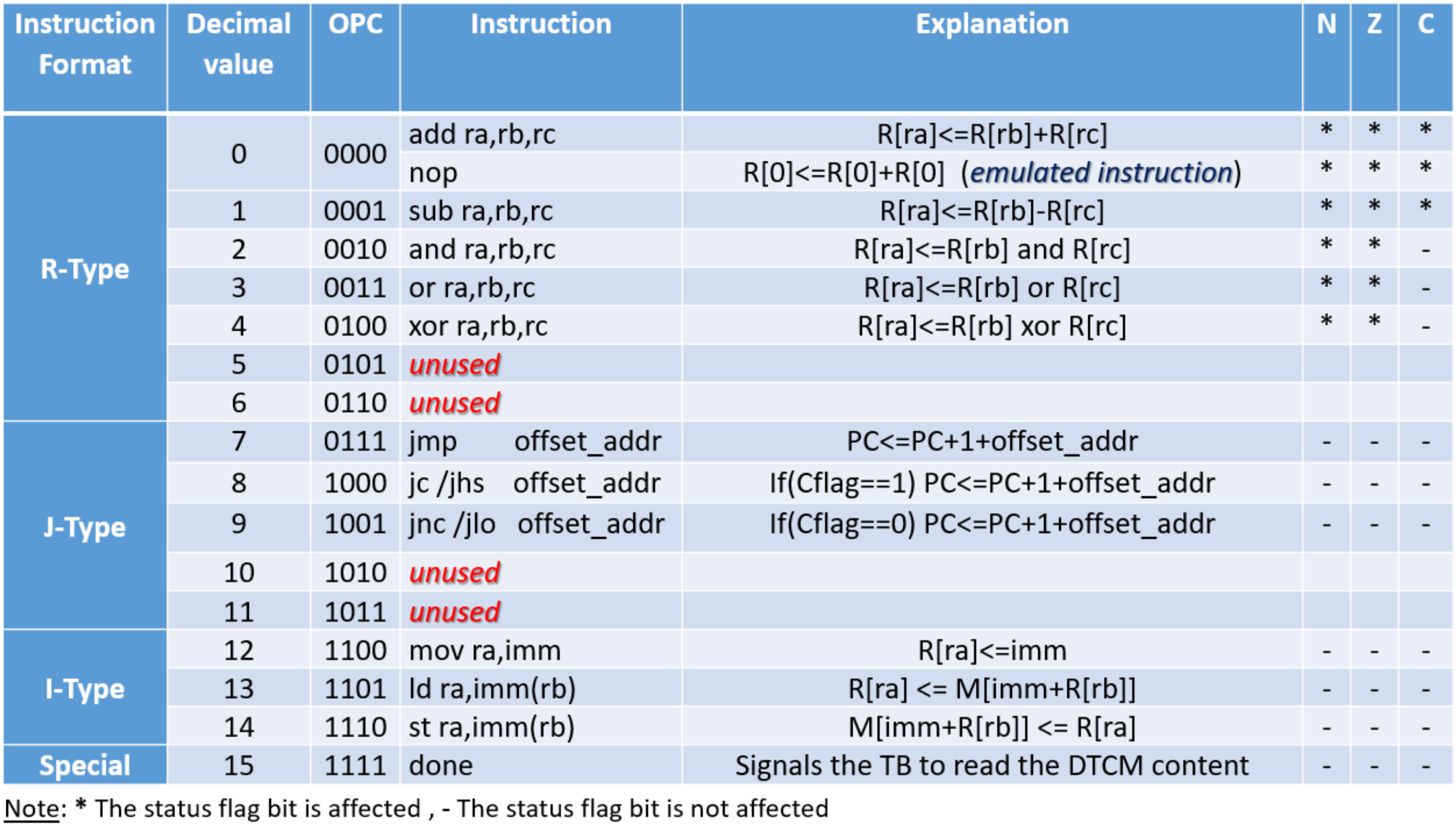
Immediate	mov ra,imm	I-Type	mov r5,0x30
			mov r5,Var
				R[5] ← 0x30
				R[5] ← Var

Direct	ld ra,imm(r0)		ld r4,0x20(r0)
			ld r4, Var(r0)
				R[4] ← M[0x20]
				R[4] ← M[Var]

Indirect	ld ra,0(rb)		ld r4,0(r3)	R[4] ← M[R[3]]

Indexed	ld ra,imm(rb)		ld r4,0x20(r3)
			ld r4, Var(r3)
				R[4] ← M[0x20 + R[3]]
				R[4] ← M[Var + R[3]]

Table 1: SRMC-I Addressing Mode



Instruction Format	Decimal value	OPC	Instruction	Explanation	N	Z	C
R-Type	0	0000	add ra,rb,rc	R[ra]<=R[rb]+R[rc]	*	*	*
			nop	R[0]<=R[0]+R[0] (*emulated instruction*)	*	*	*
	1	0001	sub ra,rb,rc	R[ra]<=R[rb]-R[rc]	*	*	*
	2	0010	and ra,rb,rc	R[ra]<=R[rb] and R[rc]	*	*	-
	3	0011	or ra,rb,rc	R[ra]<=R[rb] or R[rc]	*	*	-
	4	0100	xor ra,rb,rc	R[ra]<=R[rb] xor R[rc]	*	*	-
	5	0101	*unused*				
	6	0110	*unused*				
J-Type	7	0111	jmp offset_addr	PC<=PC+1+offset_addr	-	-	-
	8	1000	jc /jhs offset_addr	If(Cflag==1) PC<=PC+1+offset_addr	-	-	-
	9	1001	jnc /jlo offset_addr	If(Cflag==0) PC<=PC+1+offset_addr	-	-	-
	10	1010	*unused*				
	11	1011	*unused*				
I-Type	12	1100	mov ra,imm	R[ra]<=imm	-	-	-
	13	1101	ld ra,imm(rb)	R[ra] <= M[imm+R[rb]]	-	-	-
	14	1110	st ra,imm(rb)	M[imm+R[rb]] <= R[ra]	-	-	-
Special	15	1111	done	Signals the TB to read the DTCM content	-	-	-

Note: * The status flag bit is affected , - The status flag bit is not affected

Table 2: SRMC-I instruction set, recall that R[0] is a zero value (hard-wired).