Notation:

Registers are denoted *RX* where *X* is a number between 0 and 7 inclusive.

Named registers are as follows:

- Status register: SR
- Link register: LR
- Stack Pointer: SP
- Program Counter: PC

Flag Bits Key:

- : unaffected
1 : set to 1
0 : set to 0
? : Affected but value depends on result.

Rd – Destination Register
Ra – argument register a
Rb – argument register b
Ka – Constant address
Kd – Constant data
[x] – Value contained within 'x'
{x} – Location pointed too by 'x'
M – The result of a mask operation on the lower 5 bits of the SR

Status Register:

Bit	Flag	Description
0	N	Negative Flag – The result of the last operation was negative.
1	Z	Zero Flag – The result of the last operation was zero.
2	O	Overflow – The previous operation caused an overflow.
3	L	Less than - the result of the previous compare operation
4	E	Equal to - the result of the previous compare operation
5	G	Greater than - the result of the previous compare operation
6	PR	Privilege Bits. These store the security privilege of the currently running code. They are configured by the kernel and from then on are handled only by hardware.
7		
8	S	Supervisor mode. The Privilege bits may only be configured when this is set to 1. If set then we are running in “super user” mode.
9+		Reserved for future use.

Instruction Set:

#	OPCODE	MEMONIC	DESCRIPTION	FLAGS ALTERED	FUNCTION
00	??????	BLSLI	Logical Shift Left With Immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] << Kd
01	??????	BLSLR	Logical Shift Left With Register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] << [Rb]
02	??????	BLSRI	Logical Shift Right with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] >> Kd
03	??????	BLSRR	Logical Shift Right with Register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] >> [Rb]
04	??????	BANDI	Bitwise AND with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] ^ Kd
05	??????	BANDR	Bitwise AND with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] ^ [Rb]
06	??????	BNADI	Bitwise NAND with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] AND Kd
07	??????	BNADR	Bitwise NAND with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] NAND Kd
08	??????	BIORI	Bitwise inclusive OR with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] OR Kd
09	??????	BIORR	Bitwise inclusive OR with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] OR [Rb]
10	??????	BNORI	Bitwise NOR with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] NOR Kd
11	??????	BNORR	Bitwise NOR with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] NOR [Rb]
12	??????	BXORI	Bitwise XOR with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] XOR Kd
13	??????	BXORR	Bitwise XOR with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] XOR [Rb]
14	??????	BNOTR	Bitwise NOT	N:? Z:? O:? L:- E:- G:-	Rd ← NOT [Ra]
15	??????	AASLI	Arithmetic shift left with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] < Kd
16	??????	AASLR	Arithmetic shift left with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] < [Rb]
17	??????	AASRI	Arithmetic shift left with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] > Kd
18	??????	AASRR	Arithmetic shift right with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] > [Rb]
19	??????	AADDI	Add with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] + Kd
20	??????	AADDR	Add with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] + [Rb]
21	??????	ASUBI	Subtract with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] - Kd
22	??????	ASUBR	Subtract with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] - [Rb]
23	??????	AMULI	Multiply with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] * Kd

#	OPCODE	MEMONIC	DESCRIPTION	FLAGS ALTERED	FUNCTION
24	??????	AMULR	Multiply with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] * [Rb]
25	??????	ADIVI	Integer Divide with immediate	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] / Kd
26	??????	ADIVR	Integer Divide with register	N:? Z:? O:? L:- E:- G:-	Rd ← [Ra] / [Rb]
27	??????	MLRFI	Load to register from immediate	N:? Z:? O:- L:- E:- G:-	Rd ← {Kd}
28	??????	MLIRO	Load to register from immediate with register offset	N:? Z:? O:- L:- E:- G:-	Rd ← {Kd + [Ra]}
29	??????	MLRRO	Load to register from register with register offset	N:? Z:? O:- L:- E:- G:-	Rd ← {[Ra] + [Rb]}
30	??????	MSRAI	Store register at immediate	N:- Z:- O:- L:- E:- G:-	{Kd} ← [Rd]
31	??????	MSIRO	Store register at immediate with register offset	N:- Z:- O:- L:- E:- G:-	{Kd + [Ra]} ← [Rd]
32	??????	MSRRO	Store register at register with register offset	N:- Z:- O:- L:- E:- G:-	{[Ra] + [Rb]} ← [Rd]
33	??????	BRANI	Branch to immediate address	N:- Z:- O:- L:- E:- G:-	PC ← Kd
34	??????	BRANR	Branch to address in register	N:- Z:- O:- L:- E:- G:-	PC ← {Ra}
35	??????	BRWMI	Branch with mask to immediate address	N:- Z:- O:- L:- E:- G:-	PC ← {Ra} ? M!=0
36	??????	CALLR	Call subroutine starting add address in register. Push PC to LR	N:- Z:- O:- L:- E:- G:-	LR ← [PC]; PC ← {Ra}
37	??????	CARET	Call Return – Return from Subroutine	N:- Z:- O:- L:- E:- G:-	PC ← [LR]
38	??????	STPSH	Push register to stack. Increment stack pointer.	N:- Z:- O:- L:- E:- G:-	{SP} ← Ra
39	??????	STPOP	Pop to register from stack. Decrement stack pointer.	N:- Z:- O:- L:- E:- G:-	Rd ← {SP}
40	??????	STKLD	Load stack register to gp register	N:- Z:- O:- L:- E:- G:-	Rd ← [SP]
41	??????	STLST	Load the status register to gp register.	N:? Z:? O:- L:- E:- G:-	Rd ← [SR]
42	??????	COMPR	Compare Two Registers and set the status flags accordingly	N:? Z:? O:0 L:? E:? G:?	Perform (Ra – Rd)
43	??????				
44	??????				
45	??????				
46	??????				
47	??????				
48	??????				
49	??????				
50	??????				

#	OPCODE	MEMONIC	DESCRIPTION	FLAGS ALTERED	FUNCTION
51	??????				
52	??????				
53	??????				
54	??????				
55	??????				
56	??????				
57	??????				
58	??????				
59	??????				
60	??????				
61	??????	DEBUG	Used to access debug functionality from within the SIM	N:- Z:- O:- L:- E:- G:-	
62	??????	RESET	Performs a soft reset on the processor core.	N:- Z:- O:- L:- E:- G:-	PC ← 0
63	??????	HALT	Stops all processing	N:- Z:- O:- L:- E:- G:-	