LDOI MARK II instruction set

Instruction format

Jump instructions

Instruction	*Usage*	*Explanation*	*Format*			
NOP	NOP	Idle operation	*field*	*length*	*bits*	*value*
			opcode	5 bit	15-11	00000b
			don't care	11 bit	10-0	don't care
RETI	RETI	Return from interrupt	*field*	*length*	*bits*	*value*
			opcode	5 bit	15-11	00001b
			don't care	11 bit	10-0	don't care
RETC	RETC	Return from call (subroutine)	*field*	*length*	*bits*	*value*
			opcode	5 bit	15-11	00001b
			don't care	11 bit	10-0	don't care
CALL	CALL addr	Jump to subroutine at address 'addr'	*field*	*length*	*bits*	*value*
		return using RETC	opcode	5 bit	15-11	00011b
			don't care	3 bit	10-8	don't care
			addr	8 bit	7-0	addr
JMP	JMP addr	Jump to address 'addr' (for next instruction)	*field*	*length*	*bits*	*value*
			opcode	5 bit	15-11	00100b
			don't care	3 bit	10-8	don't care
			addr	8 bit	7-0	addr
JZ	JZ addr	Jump to address 'addr' (for next instruction)	*field*	*length*	*bits*	*value*
		if ZF is set	opcode	5 bit	15-11	00101b
			flag	3 bit	10-8	000b
			addr	8 bit	7-0	addr
JC	JC addr	Jump to address 'addr' (for next instruction)	*field*	*length*	*bits*	*value*
		if CF is set	opcode	5 bit	15-11	00101b
			flag	3 bit	10-8	001b
			addr	8 bit	7-0	addr

Jump instructions (continued)

Instruction	*Usage*	*Explanation*	*Format*			
JE	JE addr	Jump to address 'addr' (for next instruction)	*field*	*length*	*bits*	*value*
		if EF is set	opcode	5 bit	15-11	00101b
			flag	3 bit	10-8	010b
			addr	8 bit	7-0	addr
JG	JG addr	Jump to address 'addr' (for next instruction)	*field*	*length*	*bits*	*value*
		if GF is set	opcode	5 bit	15-11	00101b
			flag	3 bit	10-8	011b
			addr	8 bit	7-0	addr
JS	JS addr	Jump to address 'addr' (for next instruction)	*field*	*length*	*bits*	*value*
		if SF is set	opcode	5 bit	15-11	00101b
			flag	3 bit	10-8	100b
			addr	8 bit	7-0	addr

Register / memory instructions

Instruction	*Usage*	*Explanation*	*Format*			
MOVL	MOVL Rd, L	Move literal value L to register Rd	*field*	*length*	*bits*	*value*
			opcode	5 bit	15-11	01000b
			Rd	3 bit	10-8	0-7
			L	8 bit	7-0	L
MOVR	MOVR Rd, Rs	Move register Rs to register Rd	*field*	*length*	*bits*	*value*
			opcode	5 bit	15-11	01001b
			Rd	3 bit	10-8	0-7
			Rs	3 bit	7-5	0-7
			don't care	5 bit	4-0	don't care
LDR	LDR Rd, addr	Move contents of memory location 'addr'	*field*	*length*	*bits*	*value*
		to register Rd	opcode	5 bit	15-11	01010b
			Rd	3 bit	10-8	0-7
			addr	8 bit	7-0	addr

Register / memory instructions (continued)

Instruction	*Usage*	*Explanation*	*Format*			
STR	STR addr, Rs	Move register Rs to memory location 'addr'	*field*	*length*	*bits*	*value*
			opcode	5 bit	15-11	01011b
			Rs	3 bit	10-8	0-7
			addr	3 bit	7-0	addr
LDRR	LDRR Rd, Rs	Move contents of the memory location,	*field*	*length*	*bits*	*value*
		contained in Rs, to Rd	opcode	5 bit	15-11	01001b
			Rd	3 bit	10-8	0-7
			Rs	3 bit	7-5	0-7
			don't care	5 bit	4-0	don't care
STRR	STRR Rd, Rs	Move the contents of Rs to the memory	*field*	*length*	*bits*	*value*
		location, contained in Rd	opcode	5 bit	15-11	01001b
			Rd	3 bit	10-8	0-7
			Rs	3 bit	7-5	0-7
			don't care	5 bit	4-0	don't care
PUSH	PUSH Rs	Move content of register Rd to stack	*field*	*length*	*bits*	*value*
			opcode	5 bit	15-11	01110b
			Rd	3 bit	10-8	0-7
			don't care	8 bit	7-0	don't care
POP	POP Rd	Copy last stack entry to register Rd	*field*	*length*	*bits*	*value*
			opcode	5 bit	15-11	01111b
			Rs	3 bit	10-8	0-7
			don't care	8 bit	7-0	don't care

ALU operations - single operand

Instruction	*Usage*	*Explanation*	*Format*			
NOT	NOT Rds	Take bitwise complement of Rds	*field*	*length*	*bits*	*value*
		(result in Rds)	opcode	5 bit	15-11	10000b
		flags: ZF	Rds	3 bit	10-8	0-7
			Rds	3 bit	7-5	same as above
			don't care	5 bit	4-0	don't care

ALU operations - single operand (continued)

Instruction	*Usage*	*Explanation*	*Format*			
RR	RR Rds	Shift value in Rds 1 bit to the right	*field*	*length*	*bits*	*value*
		(result in Rds)	opcode	5 bit	15-11	10001b
		flags: CF, ZF	Rds	3 bit	10-8	0-7
			Rds	3 bit	7-5	same as above
			don't care	3 bit	4-0	don't care
RL	RL Rds	Shift value in Rds 1 bit to the left	*field*	*length*	*bits*	*value*
		(result in Rds)	opcode	5 bit	15-11	10010b
		flags: CF, ZF	Rds	3 bit	10-8	0-7
			Rds	3 bit	7-5	same as above
			don't care	3 bit	4-0	don't care
SWAP	SWAP Rds	Swap nibbles in Rds	*field*	*length*	*bits*	*value*
		(result in Rds)	opcode	5 bit	15-11	10011b
		flags: ZF	Rds	3 bit	10-8	0-7
			Rds	3 bit	7-5	same as above
			don't care	3 bit	4-0	don't care
INC	INC Rds	Increment Rds (+1)	Virtual instruction:			
		(result in Rds)	assembled as ADDL Rds, 01			
		flags: CF, ZF				
DEC	DEC Rds	Decrement Rds (-1)	Virtual instruction:			
		(result in Rds)	assembled as SUBL Rds ,01			
		flags: CF, ZF				
CLR	CLR Rds	Clear Rds (=0)	*Virtual instruction:*			
		(result in Rds)	assembled as MOVL Rds, 00			
		no flags affected				

ALU operaties - two operands

Instruction	*Usage*	*Explanation*	*Format*			
ANDL	AND Rd, L	Bitwise logical AND of Rd and L	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	10100b
		flags: ZF	Rd	3 bit	10-8	0-7
			L	8 bit	7-0	L

ALU operaties - two operands (continued)

Instruction	*Usage*	*Explanation*	*Format*			
ANDR	ANDR Rd, Rs	Bitwise logical AND of Rd and Rs	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	10101b
		flags: ZF	Rd	3 bit	10-8	0-7
			Rs	3 bit	7-5	0-7
			don't care	5 bit	4-0	don't care
ORL	ORL Rd, L	Bitwise logical OR of Rd and L	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	10110b
		flags: ZF	Rd	3 bit	10-8	0-7
			L	8 bit	7-0	L
ORR	ORR Rd, Rs	Bitwise logical OR of Rd and Rs	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	10111b
		flags: ZF	Rd	3 bit	10-8	0-7
			Rs	3 bit	7-5	0-7
			don't care	5 bit	4-0	don't care
XORL	XORL Rd, L	Bitwise logical XOR of Rd and L	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	11000b
		flags: ZF	Rd	3 bit	10-8	0-7
			L	8 bit	7-0	L
XORR	XORR Rd, Rs	Bitwise logical XOR of Rd and Rs	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	11001b
		flags: ZF	Rd	3 bit	10-8	0-7
			Rs	3 bit	7-5	0-7
			don't care	5 bit	4-0	don't care
ADDL	ADDL Rd, L	Addition of Rd and L	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	11010b
		flags: ZF, CF	Rd	3 bit	10-8	0-7
			L	8 bit	7-0	L
ADDR	ADDR Rd, Rs	Addition of Rd and Rs	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	11011b
		flags: ZF, CF	Rd	3 bit	10-8	0-7
			Rs	3 bit	7-5	0-7
			don't care	5 bit	4-0	don't care

ALU operaties - two operands (continued 2)

Instruction	*Usage*	*Explanation*	*Format*			
SUBL	SUBL Rd, L	Subtraction of Rd and L	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	11100b
		flags: ZF, CF	Rd	3 bit	10-8	0-7
			L	8 bit	7-0	L
SUBR	SUBR Rd, Rs	Subtraction of Rd and Rs	*field*	*length*	*bits*	*value*
		(result in Rd)	opcode	5 bit	15-11	11101b
		flags: ZF, CF	Rd	3 bit	10-8	0-7
			Rs	3 bit	7-5	0-7
			don't care	5 bit	4-0	don't care
CMPL	ADDL Rd, L	Comparison of Rd and L (<, >, =)	*field*	*length*	*bits*	*value*
		(no result, only flags)	opcode	5 bit	15-11	11110b
		flags: EF, SF, GF	Rd	3 bit	10-8	0-7
			L	8 bit	7-0	L
CMPR	ADDR Rd, Rs	Comparison of Rd and Rs (<, >, =)	*field*	*length*	*bits*	*value*
		(no result, only flags)	opcode	5 bit	15-11	11111b
		flags: EF, SF, GF	Rd	3 bit	10-8	0-7
			Rs	3 bit	7-5	0-7
			don't care	3 bit	4-0	don't care