Basic commands

```
ADD   0000000SSSSSSSSSS000ddddd0110011
ADDI  iiiiiiiiiiiissssss000ddddd0010011
AND   0000000SSSSSSSSSS111ddddd0110011
ANDI  iiiiiiiiiiiissssss111ddddd0010011
AUIPC iiiiiiiiiiiiiiiiiiiiddddd0010111
BEQ   iiiiiiiSSSSSSSSSS000iiiii1100011
BNE   iiiiiiiSSSSSSSSSS001iiiii1100011
JAL   iiiiiiiiiiiiiiiiiiiiddddd1101111
JALR  iiiiiiiiiiiissssss000ddddd1100111
LB    iiiiiiiiiiiissssss000ddddd0000011
LUI   iiiiiiiiiiiiiiiiiiiiddddd0110111
LW    iiiiiiiiiiiissssss010ddddd0000011
OR    0000000SSSSSSSSSS110ddddd0110011
ORI   iiiiiiiiiiiissssss110ddddd0010011
SB    iiiiiiiSSSSSSSSSS000iiiii0100011
SLLI  0000000iiiiissssss001ddddd0010011
SRLI  0000000iiiiissssss101ddddd0010011
SW    iiiiiiiSSSSSSSSSS010iiiii0100011
XOR   0000000SSSSSSSSSS100ddddd0110011
```

+3

```
ANDN  0100000SSSSSSSSSS111ddddd0110011
MINU  0000101SSSSSSSSSS110ddddd0110011
XNOR  0100000SSSSSSSSSS100ddddd0110011
```

Interrupt and exception

```
CSRRC  ccccccccccccssssss011ddddd1110011
CSRRS  ccccccccccccssssss010ddddd1110011
CSRRW  ccccccccccccssssss001ddddd1110011
EBREAK 00000000000100000000000001110011
ECALL  00000000000000000000000001110011
MRET   00110000001000000000000001110011
SLTU   0000000SSSSSSSSSS011ddddd0110011
```

Supplementary

```
BLT   iiiiiiiSSSSSSSSSS100iiiii1100011
```

Control/status registers(all 32bits)

名称	编号	描述	字段1	字段1描述	字段2	字段3
mtvec	0x305	保存发生异常时处理器需要跳转到的地址	BASE[31:2]		MODE[1:0]	00则pc<-BASE,01则pc<-BASE+4*cause
mscratch	0x340	暂时存放一个字大小的数据				
mepc	0x341	指向发生异常的指令				
mcause	0x342	指示发生异常的种类	Interrupt[31]	0为异常，1为中断	Exception Code[30:0]	见下面两表
mstatus	0x300	保存全局中断使能，以及许多其他的状态	MPP[12:11]	当前权限模式。00为U,01为S,11为M		
mie	0x304	指出处理器目前能处理和必须忽略的中断	MTIE[7]			
mip	0x344	列出目前正准备处理的中断	MTIP[7]	只读		

名称	编号	描述	字段1	字段1描述	字段2	字段3
mtval	0x343	trap的附加信息：地址异常中出错的地址，非法指令异常的指令等				

Exception table

Interrupt	Exception Code	Description	详细描述
1	0	*Reserved*	
1	1	Supervisor software interrupt	
1	2	*Reserved*	
1	3	Machine software interrupt	
1	4	*Reserved*	
1	5	Supervisor timer interrupt	
1	6	*Reserved*	
1	7	Machine timer interrupt	
1	8	*Reserved*	
1	9	Supervisor external interrupt	
1	10	*Reserved*	
1	11	Machine external interrupt	
1	12–15	*Reserved*	
1	≥16	*Designated for platform use*	
0	0	Instruction address misaligned	真正跳转到的地址不是4字节对齐
0	1	Instruction access fault	读指令时出错
0	2	Illegal instruction	不合法的指令
0	3	Breakpoint	ebreak造成的断点
0	4	Load address misaligned	读内存时地址未对齐
0	5	Load access fault	读内存时出错
0	6	Store/AMO address misaligned	写内存时地址未对齐
0	7	Store/AMO access fault	写内存时出错
0	8	Environment call from U-mode	用户模式下ecall
0	9	Environment call from S-mode	监管者模式下ecall
0	10	*Reserved*	

Interrupt	Exception Code	Description	详细描述
0	11	Environment call from M-mode	机器模式下ecall
0	12	Instruction page fault	//TODO
0	13	Load page fault	//TODO
0	14	*Reserved*	
0	15	Store/AMO page fault	//TODO
0	16–23	*Reserved*	
0	24–31	*Designated for custom use*	
0	32–47	*Reserved*	
0	48–63	*Designated for custom use*	
0	≥64	*Reserved*	

Priority for exceptions

Priority	Exception Code	Description
Highest	3	Instruction address breakpoint
	12	Instruction page fault
	1	Instruction access fault
	2	Illegal instruction
	0	Instruction address misaligned
	8, 9, 11	Environment call
	3	Environment break
	3	Load/Store/AMO address breakpoint
Optionally, these may have	6	Store/AMO address misaligned
lowest priority instead.	4	Load address misaligned
	15	Store/AMO page fault
	13	Load page fault
	7	Store/AMO access fault
	5	Load access fault

MMIO registers

名称	地址	描述
mtime	0x200bff8	[63:0],表示当前时间
mtimecmp	0x2004000	[63:0],表示下次时钟中断时间

Br_comparator

信号	rdata1	rdata2	br_un	br_eq	br_lt
属性	in wire[31:0]	in wire[31:0]	in wire	out wire	out wire
描述			是否是无符号数比较	rdata1=rdata2才为1	rdata1<rdata2才为1

Decoder

sign_ext = 20{inst[31]}

sign_ext_jal = 11{inst[31]}

unsign_ext = 27{0}

信号	inst	csr_data	br_eq	br_lt	ext_op	alu_op	imm	b_select	a_select	reg_a	reg_b	reg_d	pc_select	csr	mem_wr	mem_to_reg	reg_wr	csr_reg_wr	exception	备注
属性	in wire[31:0]	in wire[31:0]	in wire	in wire	out wire[4:0]	out wire[3:0]	out wire[31:0]	out wire	out wire	out wire[4:0]	out wire[4:0]	out wire[4:0]	out wire	out wire[11:0]	out wire	out wire[1:0]	out wire	out wire	out wire[3:0]	
描述			a=b为1,否则为0	a<b为1,否则为0	执行前段操作码	alu操作码		选imm代替reg_b则为0	选pc代替reg_a则为1				pc<-pc+4为0,pc<-alu结果为1		读为0,写为1	选内存为0,选alu为1,选(PC+4)为2,选data_b为3	写为1	写为1，写alu结果	decode过程中发生的异常，报异常号	
add					OP_ADD	ADD	/	1	0	inst[19:15]	inst[24:20]	inst[11:7]	0		0	1	1	0		
addi					OP_ADD	ADD	{sign_ext,inst[31:20]}	0	0	inst[19:15]	/	inst[11:7]	0		0	1	1	0		
and					OP_AND	AND	/	1	0	inst[19:15]	inst[24:20]	inst[11:7]	0		0	1	1	0		
andn					OP_AND	ANDN	/	1	0	inst[19:15]	inst[24:20]	inst[11:7]	0		0	1	1	0		
andi					OP_AND	AND	{sign_ext,inst[31:20]}	0	0	inst[19:15]	/	inst[11:7]	0		0	1	1	0		
or					OP_OR	OR	/	1	0	inst[19:15]	inst[24:20]	inst[11:7]	0		0	1	1	0		
ori					OP_OR	OR	{sign_ext,inst[31:20]}	0	0	inst[19:15]	/	inst[11:7]	0		0	1	1	0		
xor					OP_XOR	XOR	/	1	0	inst[19:15]	inst[24:20]	inst[11:7]	0		0	1	1	0		
xnor					OP_XOR	XNOR	/	1	0	inst[19:15]	inst[24:20]	inst[11:7]	0		0	1	1	0		
minu					OP_MIN	MINU	/	1	0	inst[19:15]	inst[24:20]	inst[11:7]	0		0	1	1	0		
slli					OP_SLL	SLL	{unsign_ext,inst[24:20]}	0	0	inst[19:15]	/	inst[11:7]	0		0	1	1	0		
srli					OP_SRL	SRL	{unsign_ext,inst[24:20]}	0	0	inst[19:15]	/	inst[11:7]	0		0	1	1	0		
lui					OP_LUI	ADD	{inst[31:12],12'h000}	0	0	0	/	inst[11:7]	0		0	1	1	0		
auipc					OP_AUIPC	ADD	{inst[31:12],12'h000}	0	1	/	/	inst[11:7]	0		0	1	1	0		
lb					OP_LB	ADD	{sign_ext, inst[31:20]}	0	0	inst[19:15]	/	inst[11:7]	0		0	0	1	0		
lw					OP_LW	ADD	{sign_ext, inst[31:20]}	0	0	inst[19:15]	/	inst[11:7]	0		0	0	1	0		
sb					OP_SB	ADD	{sign_ext, inst[31:25], inst[11:7]}	0	0	inst[19:15]	inst[24:20]	inst[11:7]	0		1	/	0	0		存的值为reg_d的低8位
sw					OP_SW	ADD	{sign_ext, inst[31:25], inst[11:7]}	0	0	inst[19:15]	inst[24:20]	inst[11:7]	0		1	/	0	0		存的值为reg_d
beq			1		OP_BEQ	ADD	{sign_ext,inst[7],inst[30:25],inst[11:8],1'b0}	0	1	/	/	/	1		0	/	0	0		
			0		OP_BEQ	ADD	/	/	/	/	/	/	0		0	/	0	0		
bne			0		OP_BNE	ADD	/	/	/	/	/	/	0		0	/	0	0		
			1		OP_BNE	ADD	{sign_ext,inst[7],inst[30:25],inst[11:8],1'b0}	0	1	/	/	/	1		0	/	0	0		
blt				1	OP_BLT	ADD	{sign_ext,inst[7],inst[30:25],inst[11:8],1'b0}	0	1	/	/	/	1		0	/	0	0		
				0	OP_BLT	ADD	/	/	/	/	/	/	0		0	/	0	0		
jal					OP_JAL	ADD	{sign_ext_jal,inst[31],inst[19:12],inst[20],inst[30:21],1'b0}	0	1	/	/	inst[11:7]	1		0	2	1	0		
jalr					OP_JALR	ADD	{sign_ext,inst[31:20]}	0	0	inst[19:15]	/	inst[11:7]	1		0	2	1	0		ALU结果最低位清零
csrrc					OP_CSRR	NANDN	csr_data	0	0	inst[19:15]	/	inst[11:7]	0	inst[31:20]	0	3	1	1		
csrrs					OP_CSRR	OR	csr_data	0	0	inst[19:15]	/	inst[11:7]	0	inst[31:20]	0	3	1	1		
csrrw					OP_CSRR	ADD	csr_data	0	0	inst[19:15]	/	inst[11:7]	0	inst[31:20]	0	3	1	1		
ebreak					OP_EBREAK	ADD	/	/	/	/	/	/	0		0	/	0	0		
ecall					OP_ECALL	ADD	/	/	/	/	/	/	0		0	/	0	0		
mret					OP_MRET	ADD	/	/	/	/	/	/	0		0	/	0	0		
sltu					OP_SLT	SLTU	/	1	0	inst[19:15]	inst[24:20]	inst[11:7]	0		0	1	1	0		