

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
funct7							rs2					rs1					funct3			rd			opcode						R-type				
imm[11:0]												rs1					funct3			rd			opcode						I-type				
imm[11:5]							rs2					rs1					funct3			imm[4:0]			opcode						S-type				
imm[12 10:5]							rs2					rs1					funct3			rd			opcode						B-type				
imm[31:12]																								rd			opcode						U-type
imm[20 10:1 11 19:12]																								rd			opcode						J-type

Zbb: “Basic bit-manipulation” Extension

31	25 24							20 19					15 14			12 11		7	6						0					
0	1	0	0	0	0	0	0	rs2					rs1					1	1	1	rd		0	1	1	0	0	1	1	ANDN
0	1	0	0	0	0	0	0	rs2					rs1					1	1	0	rd		0	1	1	0	0	1	1	ORN
0	1	0	0	0	0	0	0	rs2					rs1					1	0	0	rd		0	1	1	0	0	1	1	XNOR
0	1	1	0	0	0	0	0	0	0	0	0	0	rs1					0	0	1	rd		0	0	1	0	0	1	1	CLZ
0	1	1	0	0	0	0	0	0	0	0	0	1	rs1					0	0	1	rd		0	0	1	0	0	1	1	CTZ
0	1	1	0	0	0	0	0	0	0	0	1	0	rs1					0	0	1	rd		0	0	1	0	0	1	1	CPOP
0	0	0	0	0	1	0	1	rs2					rs1					1	1	0	rd		0	1	1	0	0	1	1	MAX
0	0	0	0	0	1	0	1	rs2					rs1					1	1	1	rd		0	1	1	0	0	1	1	MAXU
0	0	0	0	0	1	0	1	rs2					rs1					1	0	0	rd		0	1	1	0	0	1	1	MIN
0	0	0	0	0	1	0	1	rs2					rs1					1	0	1	rd		0	1	1	0	0	1	1	MINU
0	1	1	0	0	0	0	0	0	0	1	0	0	rs1					0	0	1	rd		0	0	1	0	0	1	1	SEXT.B
0	1	1	0	0	0	0	0	0	0	1	0	1	rs1					0	0	1	rd		0	0	1	0	0	1	1	SEXT.H
0	0	0	0	0	1	0	0	0	0	0	0	0	rs1					1	0	0	rd		0	1	1	0	0	1	1	ZEXT.H
0	1	1	0	0	0	0	0	rs2					rs1					0	0	1	rd		0	1	1	0	0	1	1	ROL
0	1	1	0	0	0	0	0	rs2					rs1					1	0	1	rd		0	1	1	0	0	1	1	ROR
0	1	1	0	0	0	0	0	shamt					rs1					1	0	1	rd		0	0	1	0	0	1	1	RORI
0	0	1	0	1	0	0	0	0	0	1	1	1	rs1					1	0	1	rd		0	0	1	0	0	1	1	ORC.B
0	1	1	0	1	0	0	0	1	1	0	0	0	rs1					1	0	1	rd		0	0	1	0	0	1	1	REV8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
funct7								rs2				rs1				funct3			rd			opcode							R-type			
imm[11:0]												rs1				funct3			rd			opcode							I-type			
imm[11:5]								rs2				rs1				funct3			imm[4:0]			opcode							S-type			
imm[12 10:5]								rs2				rs1				funct3			rd			opcode							B-type			
imm[31:12]																						rd			opcode							U-type
imm[20 10:1 11 19:12]																						rd			opcode							J-type

Zri: “Load/Store indirect with Index” Extension

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0	0	0	0	0	0	rs2					rs1					1 1 1			rd			0	0	0	0	0	0	1	1	LB.R
0	0	0	0	0	0	0	1	rs2					rs1					1 1 1			rd			0	0	0	0	0	0	1	1	LH.R
0	0	0	0	0	0	1	0	rs2					rs1					1 1 1			rd			0	0	0	0	0	0	1	1	LW.R
1	0	0	0	0	0	0	0	rs2					rs1					1 1 1			rd			0	0	0	0	0	0	1	1	LBU.R
1	0	0	0	0	0	0	1	rs2					rs1					1 1 1			rd			0	0	0	0	0	0	1	1	LHU.R
0	0	0	0	0	0	0	0	rs3					rs1					1 1 1			rs2			0	1	0	0	0	0	1	1	SB.R
0	0	0	0	0	0	0	1	rs3					rs1					1 1 1			rs2			0	1	0	0	0	0	1	1	SH.R
0	0	0	0	0	0	1	0	rs3					rs1					1 1 1			rs2			0	1	0	0	0	0	1	1	SW.R

```

lb    rd, rs2(rs1)
lh    rd, rs2(rs1)
lw    rd, rs2(rs1)
lbu   rd, rs2(rs1)
lhu   rd, rs2(rs1)
sb    rs2, rs3(rs1)
sh    rs2, rs3(rs1)
sw    rs2, rs3(rs1)

```

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
funct7							rs2					rs1					funct3					rd					opcode					R-type			
imm[11:0]										rs2					rs1					funct3					rd					opcode					I-type
imm[11:5]							rs2					rs1					funct3					imm[4:0]					opcode					S-type			
imm[12 10:5]							rs2					rs1					funct3					rd					opcode					B-type			
imm[31:12]															rd					opcode										U-type					
imm[20 10:1 11 19:12]															rd					opcode										J-type					

Zor: "Objective RISC" Extension

Unprivileged:

31	25	24	20			19	15	14	12	11	7	6	0																			
0	0	0	0	0	0	0	0	rs2		rs1		0	0	0	rs3		0	0	0	1	0	1	1	1	SP.R	R						
0	0	0	0	0	0	0	1	rs2		rs1		0	0	0	rd		0	0	0	1	0	1	1	1	LP.R	R						
0	0	0	0	0	0	1	0	index[4:0]		frame		0	0	0	rs1		0	0	0	1	0	1	1	1	SV	R						
0	0	0	0	0	0	1	1	index[4:0]		frame		0	0	0	rd		0	0	0	1	0	1	1	1	RST	R						
0	0	0	0	0	1	0	0	zero		rs1		0	0	0	rd		0	0	0	1	0	1	1	1	QDTB	R						
0	0	0	0	0	1	0	1	zero		rs1		0	0	0	rd		0	0	0	1	0	1	1	1	QDTH	R						
0	0	0	0	0	1	1	0	zero		rs1		0	0	0	rd		0	0	0	1	0	1	1	1	QDTW	R						
0	0	0	0	0	1	1	1	zero		rs1		0	0	0	rd		0	0	0	1	0	1	1	1	QDTD	R						
0	0	0	0	1	0	0	0	zero		rs1		0	0	0	rd		0	0	0	1	0	1	1	1	QPI	R						
0	0	0	0	1	0	0	1	zero		zero		0	0	0	rd		0	0	0	1	0	1	1	1	GCP	R						
0	0	0	0	1	1	0	0	zero		frame		0	0	0	frame		0	0	0	1	0	1	1	1	POP	R						
0	0	0	1	0	0	0	1	zero		zero		0	0	0	zero		0	0	0	1	0	1	1	1	RTLIB	R						
0	0	0	1	0	0	1	0	zero		zero		0	0	0	zero		0	0	0	1	0	1	1	1	CPFC	R						
0	0	0	1	0	0	1	1	zero		zero		0	0	0	zero		0	0	0	1	0	1	1	1	CHECK	R						
imm[11:5]			rs2					rs1					0	0	1	imm[4:0]			0	0	0	1	0	1	1	SP	S					
imm[11:0]							rs1					0	1	0	rd			0	0	0	1	0	1	1	LP	I						
imm[11:0]							rs1					0	1	1	ra			0	0	0	1	0	1	1	JLIB	I						
0	0	0	0	0	0	0	0	rs2					rs1					1	0	0	rd			0	0	0	1	0	1	1	ALC	R
pi[11:0]								rs1					1	0	1	rd			0	0	0	1	0	1	1	ALCI.P	I					
dt[11:0]								rs1					1	1	0	rd			0	0	0	1	0	1	1	ALCI.D	I					
dt[6:0]				0	0	0	0	0	rd					1	1	1	pi[4:0]			0	0	0	1	0	1	1	ALCI	S				
dt[6:0]				0	0	0	0	1	0	frame					1	1	1	pi[4:0]			0	0	0	1	0	1	1	PUSHG	S			
dt[6:0]				0	0	0	0	1	1	frame					1	1	1	pi[4:0]			0	0	0	1	0	1	1	PUSH	S			

Machine Mode:

31	26 25 24			20 19			15 14		12 11		7 6		0			
1 1 1 1 1 1	0	0 0 0 0 0	0	0 0 0 0 0	0	0 0 0	rd	1 1 1 0 0 1 1	ALCB	R						
1 1 1 1 1 1	1	rs2		rs1		0 0 0	rd	1 1 1 0 0 1 1	CIOP	R						
1 1 1 1 1 1	0	1 0 0 0 0	rs1		0 0 0	rd	1 1 1 0 0 1 1	CCP	R							
1 1 1 1 1 1	0	1 0 0 0 1	rs1		0 0 0	rd	1 1 1 0 0 1 1	RPR	R							
1 1 1 1 1 1	0	1 0 1 0 0	rs1		0 0 0	rd	1 1 1 0 0 1 1	QPIR	R							
1 1 1 1 1 1	0	1 0 1 0 1	rs1		0 0 0	rd	1 1 1 0 0 1 1	QDTR	R							
1 1 1 1 1 1	0	1 0 1 1 0	rs1		0 0 0	rd	1 1 1 0 0 1 1	QPTR	R							
1 1 1 1 1 1	0	0 0 0 0 0	0	0 0 0 1 0	0 0 0	rd	1 1 1 0 0 1 1	SEAL	R							
1 1 1 1 1 1	0	0 0 0 0 0	0	0 0 0 1 1	0 0 0	rd	1 1 1 0 0 1 1	UNSL	R							

Misc:

reg	alias	reg	alias
x0	zero	x16	a6
x1	ra fix	x17	a7
x2	frame	x18	s2
x3	pcd /root/core	x19	s3
x4	ctxt	x20	s4
x5	t0	x21	s5
x6	t1	x22	s6
x7	t2	x23	s7
x8	s0	x24	s8
x9	s1	x25	s9
x10	a0	x26	s10/bm
x11	a1	x27	cnst
x12	a2	x28	t3
x13	a3	x29	t4
x14	a4	x30	t5
x15	a5	x31	t6

pseudo-instruction	implemented as
lcp rd, imm(rs1)	lp rd, imm(rs1) sp x0, imm(rs1)
lcp.r rd, imm(rs1)	lp.r rd, rs2(rs1) sp.r x0, rs2(rs1)
scp rs2, imm(rs1)	sp rs2, imm(rs1) addi rs2, x0,0
scp.r rs2, rs3(rs1)	sp.r rs2, rs3(rs1) addi rs2, x0,0
pusht pi,dt	alci frame, pi,dt

Implementation:

Instruction	rdst	rdat	rptr	raux	imm
sb/h/w	zero	ra.rix	rs1	rs2	imm
lb/bu/h/hu/w	rd	---	rs1	ra	imm
sp	zero	ra.rix	rs1	rs2	imm
lp	rd	---	rs1	ra	imm
sb/h/w.r	zero	rs3	rs1 (# frame)	rs2	---
lb/bu/h/hu/w.r	rd	rs2	rs1 (# frame)	---	---
sp.r	zero	rs3	rs1 (# frame)	rs2	---
lp.r	rd	rs2	rs1 (# frame)	---	---
sv	zero	ra.rix	frame	rs1	index
rst	rd	ra.rix	frame	bm	index
qdtx					
qpi					
gcp					
pop	frame	ra.rix	frame	---	---
jlib	ra	frame	rs1	ra	imm
jal	rd	frame	---	ra	imm
jr	rd	frame	rs1	ra	imm
rtlib	ra	ra.rix	ra	frame	---
alc	rd (# frame)	rs1	alc_params	rs2	---
alci.p	rd (# frame)	rs1	alc_params	---	pi
alci.d	rd (# frame)	rs1	alc_params	---	dt
alci	rd	ra.rix	alc_params	frame	pi & dt
pushg	rd	ra.rix	alc_params	frame	pi & dt
push	rd	ra.rix	alc_params	frame	pi & dt
alcb					
ciop	rd	rs1	---	rs2	---
rpr					
qpir					
qdtr					
qptr					
seal					
unsl					

31 30 29		3 2 1 0			
ra.rix	lib entry	rix(30:1)			
frame		frame(31:3)			color
pi	uini	pi(30:2)			color
dt	rc ri	dt(29:0)			

instruction	condition	action
jlib	ra.rix(color) != frame(color) target ptr != ra.rcd	set ra.rix(lib entry), toggle rix(color)
jal ra, ... or jr ra, ...	ra.rix(color) != frame(color)	clear ra.rix(lib entry), toggle rix(color)
pushx	ra.rix(color) = frame(color)	toggle frame(color)
pop	ra.rix(color) != frame(color)	toggle frame(color)
jr ..., 0(ra)	ra.rix(color) = frame(color)	toggle ra.rix(color) if ra.rix(lib entry) = 1 do cross code-object return else stay in this code-object

OBJECTS

Ordinary

31 30 29	2 1 0	
gc	size(29:2)	0 0
...		

Frame

31 30 29			3 2 1 0			
gc	c	size(28:4)	f	r	1	0
ra-ptr?						
fp-ix?						
fp-eop!						
ra-ix!						
fp-ptr!						
...						

Data only

31 30 29	2 1 0	
gc	size(29:2)	0 1
...		

Code

31	2 1 0
eoc(30:1)	1 1
eop(30:1)	1 1
...	

Immediate (Primitive)

31	0
integer	

Immediate (Pointer)

31	0
ptr	
ix	
attr	

POINTERS & DATA

(in memory)

	31	1 0
immediate (prim) pointer:	ptr(31:2) 0 1	

	31	3 2 1 0
ord./code/d.o.-ptr:	ptr(31:4)	0 0 1 1

	31	3	2	1	0		
immediate (ptr) pointer:	ptr(31:4)			0	1	1	1
pc pointer:							

(*immediate (ptr) pointers* shall never be present in the register-file. pc pointers shall never be stored to memory, except in the hidden ra-ptr spot of stack-frames)

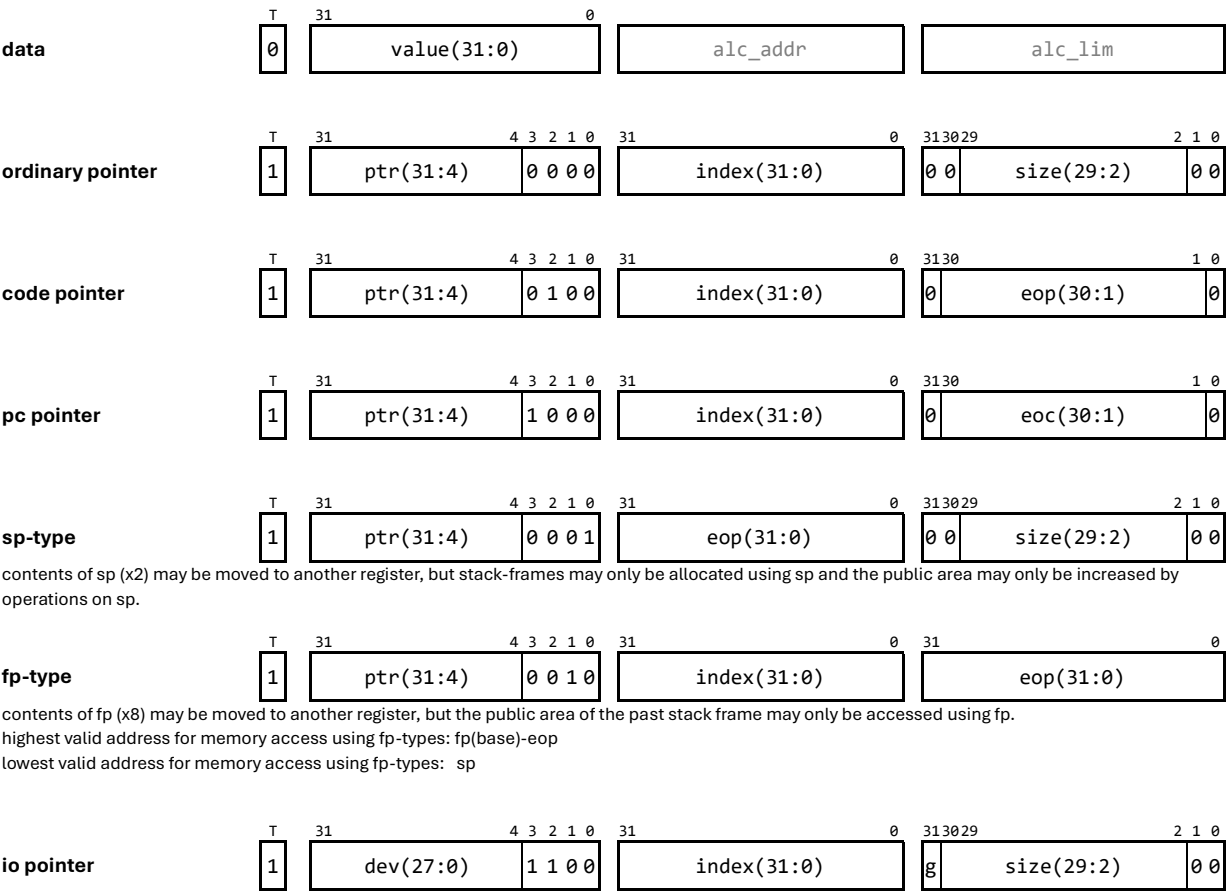
	31		4	3	2	1	0
io pointer:	dev	size	g	1	1	1	1

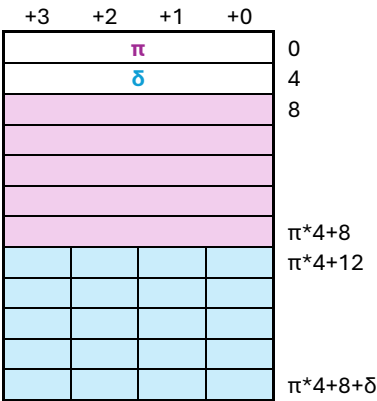
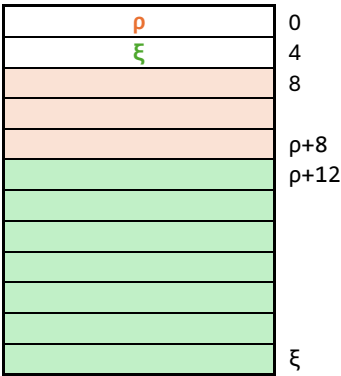
	32	31	25	24	17	16	9	8	1	0
Small Data (w):	31	int(30:0)								0
Small Data (h):	15	h1(14:0)				h0(15:0)				0
Small Data (b):	7	b3		b2		b1		b0		0

Allocate immediate primitive if:

- sw and rs(30) ≠ rs(31)
- sh at h1 and rs(14) ≠ rs(15)
- sb at b3 and (rs(7) = 1 or rs < 0)

REGISTER FILE & PIPELINE





	TAG	WERT		ATTRIBUT 1	ATTRIBUT 2
<i>Daten</i>	0	Data		null	null
<i>Datenobjektzeiger</i>	1	Pointer	000	Größe Zeigerbereich (π)	Größe Datenbereich (δ)
<i>Code-Objektzeiger</i>	1	Pointer	010	Ende öffentlicher Bereich (ρ)	Ende Code-Objekt (ξ)
<i>PC-Zeiger</i>	1	Pointer	011	Index (χ)	Ende Code-Objekt (ξ)

<i>s0</i>	0	Data		null	null
<i>s1</i>	1	Pointer	000	π	δ
<i>a0</i>	1	Pointer	010	ρ	ξ
<i>a1</i>	1	Pointer	011	χ	ξ
<i>a2</i>	0	Data		null	null
<i>a3</i>	0	Data		null	null

a4

1	Pointer	000	π	δ
---	---------	-----	-------	----------

