

## SPU Instruction Cheat Sheet (Insomniac Games)

			pipe	latency	stalls
a	rt, ra, rb	add word	even	2	0
absdb	rt, ra, rb	absolute difference of bytes	even	4	0
addx	rt, ra, rb	add word extended	even	2	0
ah	rt, ra, rb	add halfword	even	2	0
ahi	rt, ra, s10	add halfword immediate	even	2	0
ai	rt, ra, s10	add word immediate	even	2	0
and	rt, ra, rb	and	even	2	0
andbi	rt, ra, s10	and byte immediate	even	2	0
andc	rt, ra, rb	and with complement	even	2	0
andhi	rt, ra, s10	and halfword immediate	even	2	0
andi	rt, ra, s10	and word immediate	even	2	0
avgb	rt, ra, rb	average bytes	even	4	0
bg	rt, ra, rb	borrow generate word	even	2	0
bgx	rt, ra, rb	borrow generate word extended	even	2	0
bi	ra	branch indirect	odd	4	0
bid	ra	branch indirect, disable	odd	4	0
bie	ra	branch indirect, enable	odd	4	0
bihnz	rt, ra	branch indirect if not zero halfword	odd	4	0
bihnzd	rt, ra	branch indirect if not zero halfword, disable	odd	4	0
bihnze	rt, ra	branch indirect if not zero halfword, enable	odd	4	0
bihz	rt, ra	branch indirect if zero halfword	odd	4	0
bihzd	rt, ra	branch indirect if zero halfword, disable	odd	4	0
bihze	rt, ra	branch indirect if zero halfword, enable	odd	4	0
binz	rt, ra	branch indirect if not zero word	odd	4	0
binzd	rt, ra	branch indirect if not zero word, disable	odd	4	0
binze	rt, ra	branch indirect if not zero word, enable	odd	4	0
bisl	rt, ra	branch indirect and set link	odd	4	0
bisld	rt, ra	branch indirect and set link, disable	odd	4	0
bisle	rt, ra	branch indirect and set link, enable	odd	4	0
bisled	rt, ra	branch indirect and set link on external data	odd	4	0
bisledd	rt, ra	branch indirect and set link on external data, disable	odd	4	0
bislede	rt, ra	branch indirect and set link on external data, enable	odd	4	0
biz	rt, ra	branch indirect if zero word	odd	4	0
bizd	rt, ra	branch indirect if zero word, disable	odd	4	0
bize	rt, ra	branch indirect if zero word, enable	odd	4	0
br	s18	branch relative	odd	4	0
bra	s18	branch absolute	odd	4	0
brasl	rt, s18	branch absolute and set link	odd	4	0
brhnz	rt, s18	branch if not zero halfword	odd	4	0
brhz	rt, s18	branch if zero halfword	odd	4	0
brnz	rt, s18	branch if not zero word	odd	4	0
brsl	rt, s18	branch relative and set link	odd	4	0
brz	rt, s18	branch if zero word	odd	4	0
cbd	rt, u7(ra)	generate controls for byte insertion (d-form)	odd	4	0
cbx	rt, ra, rb	generate controls for byte insertion (x-form)	odd	4	0
cdd	rt, u7(ra)	generate controls for doubleword insertion (d-form)	odd	4	0
cdx	rt, ra, rb	generate controls for doubleword insertion (x-form)	odd	4	0
ceq	rt, ra, rb	compare equal word	even	2	0
cegb	rt, ra, rb	compare equal byte	even	2	0
ceqbi	rt, ra, s10	compare equal byte immediate	even	2	0
ceqh	rt, ra, rb	compare equal halfword	even	2	0
ceghi	rt, ra, s10	compare equal halfword immediate	even	2	0
ceqi	rt, ra, s10	compare equal word immediate	even	2	0
cflts	rt, ra, scale7	convert floating to signed integer	even	7	0
cfltu	rt, ra, scale7	convert floating to unsigned integer	even	7	0
cg	rt, ra, rb	carry generate word	even	2	0
cgt	rt, ra, rb	compare greater than word	even	2	0
cgtb	rt, ra, rb	compare greater than byte	even	2	0
cgtbi	rt, ra, s10	compare greater than byte immediate	even	2	0
cgth	rt, ra, rb	compare greater than halfword	even	2	0
cgthi	rt, ra, s10	compare greater than halfword immediate	even	2	0
cgti	rt, ra, s10	compare greater than word immediate	even	2	0
cgx	rt, ra, rb	carry generate word extended	even	2	0
chd	rt, u7(ra)	generate controls for halfword insertion (d-form)	odd	4	0
chx	rt, ra, rb	generate controls for halfword insertion (x-form)	odd	4	0
clgt	rt, ra, rb	compare logical greater than word	even	2	0
clgtb	rt, ra, rb	compare logical greater than byte	even	2	0
clgtbi	rt, ra, s10	compare logical greater than byte immediate	even	2	0
clgth	rt, ra, rb	compare logical greater than halfword	even	2	0
clgthi	rt, ra, s10	compare logical greater than halfword immediate	even	2	0
clgti	rt, ra, s10	compare logical greater than word immediate	even	2	0
clz	rt, ra	count leading zeros	even	2	0
cntb	rt, ra	count ones in bytes	even	4	0
csflt	rt, ra, scale7	convert signed integer to floating	even	7	0
cuflt	rt, ra, scale7	convert unsigned integer to floating	even	7	0
cwd	rt, u7(ra)	generate controls for word insertion (d-form)	odd	4	0
cwx	rt, ra, rb	generate controls for word insertion (x-form)	odd	4	0
dfa	rt, ra, rb	double floating add	even	13	6
dfm	rt, ra, rb	double floating multiply	even	13	6
dfma	rt, ra, rb	double floating multiply and add	even	13	6
dfms	rt, ra, rb	double floating multiply and subtract	even	13	6
dfnma	rt, ra, rb	double floating negative multiply and add	even	13	6
dfnms	rt, ra, rb	double floating negative multiply and subtract	even	13	6
dfs	rt, ra, rb	double floating subtract	even	13	6
dsync		synchronize data	odd	4	0
eqv	rt, ra, rb	equivalent	even	2	0

fa	rt, ra, rb	floating add	even	6	0
fceq	rt, ra, rb	floating compare equal	even	2	0
fcgt	rt, ra, rb	floating compare greater than	even	2	0
fcmeq	rt, ra, rb	floating compare magnitude equal	even	2	0
fcngt	rt, ra, rb	floating compare greater than	even	2	0
fed	rt, ra	floating extend single to double	even	13	6
fi	rt, ra, rb	floating interpolate	even	7	0
fm	rt, ra, rb	floating multiply	even	6	0
fma	rt, ra, rb, rc	floating multiply and add	even	6	0
fms	rt, ra, rb, rc	floating multiply and subtract	even	6	0
fnms	rt, ra, rb, rc	floating negative multiply and subtract	even	6	0
frds	rt, ra	floating round double to single	even	13	6
frest	rt, ra	floating reciprocal estimate	odd	4	0
frsquest	rt, ra	floating reciprocal square root estimate	odd	4	0
fs	rt, ra, rb	floating subtract	even	6	0
fscrrd	rt	floating-point status control register read	even	13	6
fscrwr	ra	floating-point status control register write	even	7	0
fscrwr	rt, ra	floating-point status control register write	even	7	0
fsm	rt, ra	form select mask for words	odd	4	0
fsmb	rt, ra	form select mask for bytes	odd	4	0
fsmbi	rt, ul6	form select mask for byte immediate	odd	4	0
fsmh	rt, ra	form select mask for halfwords	odd	4	0
gb	rt, ra	gather bits from words	odd	4	0
gbb	rt, ra	gather bits from bytes	odd	4	0
gbh	rt, ra	gather bits from halfwords	odd	4	0
hbr	sll, ra	hint for branch (r-form)	odd	15	0
hbra	sll, s18	hint for branch (a-form)	odd	15	0
hbrp		hint for branch, prefetch (r-form)	odd	15	0
hbrr	sll, s18	hint for branch relative	odd	15	0
heq	ra, rb	halt if equal	even	2	0
heq	rt, ra, rb	halt if equal	even	2	0
heqi	ra, s10	halt if equal immediate	even	2	0
heqi	rt, ra, s10	halt if equal immediate	even	2	0
hgt	ra, rb	halt if greater than	even	2	0
hgt	rt, ra, rb	halt if greater than	even	2	0
hgti	ra, s10	halt if greater than immediate	even	2	0
hgti	rt, ra, s10	halt if greater than immediate	even	2	0
hlgt	ra, rb	halt if logically greater than	even	2	0
hlgt	rt, ra, rb	halt if logically greater than	even	2	0
hlgti	ra, s10	halt if logically greater than immediate	even	2	0
hlgti	rt, ra, s10	halt if logically greater than immediate	even	2	0
il	rt, s16	immediate load word	even	2	0
ila	rt, ul8	immediate load address	even	2	0
ilh	rt, ul6	immediate load halfword	even	2	0
ilhu	rt, ul6	immediate load halfword upper	even	2	0
iohl	rt, ul6	immediate or halfword lower	even	2	0
iret		interrupt return	odd	4	0
iret	ra	interrupt return	odd	4	0
iretd		interrupt return, disable	odd	4	0
iretd	ra	interrupt return, disable	odd	4	0
irete		interrupt return, enable	odd	4	0
irete	ra	interrupt return, enable	odd	4	0
lnop		nop operation (load)	odd	1	0
lqa	rt, s18	load quadword (a-form)	odd	6	0
lqd	rt, s14(ra)	load quadword (d-form)	odd	6	0
lqr	rt, s18	load quadword instruction relative (a-form)	odd	6	0
lqx	rt, ra, rb	load quadword (x-form)	odd	6	0
mfspr	rt, spr	move from special purpose register	odd	6	0
mpy	rt, ra, rb	multiply	even	7	0
mpya	rt, ra, rb, rc	multiply and add	even	7	0
mpyh	rt, ra, rb	multiply high	even	7	0
mpyhh	rt, ra, rb	multiply high high	even	7	0
mpyhha	rt, ra, rb	multiply high high and add	even	7	0
mpyhhaui	rt, ra, rb	multiply high high unsigned and add	even	7	0
mpyhhu	rt, ra, rb	multiply high high unsigned	even	7	0
mpyi	rt, ra, s10	multiply immediate	even	7	0
mpys	rt, ra, rb	multiply and shift right	even	7	0
mpyu	rt, ra, rb	multiply unsigned	even	7	0
mpyui	rt, ra, s10	multiply unsigned immediate	even	7	0
mtspr	spr, ra	move to special purpose register	odd	6	0
nand	rt, ra, rb	nand	even	2	0
nop		nop operation (execute)	even	0	0
nop	rt	nop operation (execute)	even	0	0
nor	rt, ra, rb	nor	even	2	0
or	rt, ra, rb	or	even	2	0
orbi	rt, ra, s10	or byte immediate	even	2	0
orc	rt, ra, rb	or with complement	even	2	0
orhi	rt, ra, s10	or halfword immediate	even	2	0
ori	rt, ra, s10	or word immediate	even	2	0
orx	rt, ra	or word across	odd	4	0
rchcnt	rt, ch	read channel count	odd	6	0
rdch	rt, ch	read channel	odd	6	0
rot	rt, ra, rb	rotate word	even	4	0
roth	rt, ra, rb	rotate halfword	even	4	0
rothi	rt, ra, s7	rotate halfword immediate	even	4	0
rothm	rt, ra, rb	rotate and mask halfword	even	4	0
rothmi	rt, ra, s6	rotate and mask halfword immediate	even	4	0
roti	rt, ra, s7	rotate word immediate	even	4	0
rotm	rt, ra, rb	rotate and mask word	even	4	0
rotma	rt, ra, rb	rotate and mask algebraic word	even	4	0
rotmah	rt, ra, rb	rotate and mask algebraic halfword	even	4	0

rotmahi	rt, ra, s6	rotate and mask algebraic halfword immediate	even	4	0
rotmai	rt, ra, s7	rotate and mask algebraic word immediate	even	4	0
rotmi	rt, ra, s7	rotate and mask word immediate	even	4	0
rotqbi	rt, ra, rb	rotate quadword by bits	odd	4	0
rotqbii	rt, ra, u3	rotate quadword by bits immediate	odd	4	0
rotqby	rt, ra, rb	rotate quadword by bytes	odd	4	0
rotqbybi	rt, ra, rb	rotate quadword by bytes from bit shift count	odd	4	0
rotqbyi	rt, ra, s7	rotate quadword by bytes immediate	odd	4	0
rotqmbi	rt, ra, rb	rotate and mask quadword by bits	odd	4	0
rotqmbii	rt, ra, s3	rotate and mask quadword by bits immediate	odd	4	0
rotqmbby	rt, ra, rb	rotate and mask quadword by bytes	odd	4	0
rotqmbbybi	rt, ra, rb	rotate and mask quadword by bytes from bit shift count	odd	4	0
rotqmbyi	rt, ra, s6	rotate and mask quadword by bytes immediate	odd	4	0
selb	rt, ra, rb, rc	select bits	even	2	0
sf	rt, ra, rb	subtract from word	even	2	0
sfh	rt, ra, rb	subtract from halfword	even	2	0
sfhi	rt, ra, s10	subtract from halfword immediate	even	2	0
sfi	rt, ra, s10	subtract from word immediate	even	2	0
sfx	rt, ra, rb	subtract from word extended	even	2	0
shl	rt, ra, rb	shift left word	even	4	0
shlh	rt, ra, rb	shift left halfword	even	4	0
shlhi	rt, ra, u5	shift left halfword immediate	even	4	0
shli	rt, ra, u6	shift left word immediate	even	4	0
shlqbi	rt, ra, rb	shift left quadword by bits	odd	4	0
shlqbii	rt, ra, u3	shift left quadword by bits immediate	odd	4	0
shlqby	rt, ra, rb	shift left quadword by bytes	odd	4	0
shlqbybi	rt, ra, rb	shift left quadword by bytes from bit shift count	odd	4	0
shlqbyi	rt, ra, u5	shift left quadword by bytes immediate	odd	4	0
shufb	rt, ra, rb, rc	shuffle bytes	odd	4	0
stop	u14	stop and signal	odd	4	0
stopd	ra, rb, rc	stop and signal with dependencies	odd	4	0
stqa	rt, s18	store quadword (a-form)	odd	6	0
stqd	rt, s14(ra)	store quadword (d-form)	odd	6	0
stqr	rt, s18	store quadword instruction relative (a-form)	odd	6	0
stqx	rt, ra, rb	store quadword (x-form)	odd	6	0
sumb	rt, ra, rb	sum bytes into halfword	even	4	0
sync		synchronize	odd	4	0
syncc		synchronize channel	odd	4	0
wrch	ch, ra	write channel	odd	6	0
xor	rt, ra, rb	xor	even	2	0
xorbi	rt, ra, s10	exclusive or byte immediate	even	2	0
xorhi	rt, ra, s10	exclusive or halfword immediate	even	2	0
xori	rt, ra, s10	exclusive or word immediate	even	2	0
xsbh	rt, ra	extend sign byte to halfword	even	2	0
xshw	rt, ra	extend sign halfword to word	even	2	0
xswd	rt, ra	extend sign word to doubleword	even	2	0

	class	num	latency	pipe	unit
load and store	LS	12	6	odd	SLS
branch hints	HB	4	15	odd	SLS
branch resolution	BR	36	4	odd	SCN
channel interface, special purpose registers	CH	4	6	odd	SSC
shuffle	SH	35	4	odd	SFS
no operation (load)	LNOP	1	0	odd	LNOP
no operation (execute)	NOP	1	0	even	NOP
single-precision floating-point	SP	6	6	even	SFP
double floating-point	DP	10	13	even	SFP
floating-point integer	FI	17	7	even	SFP
byte operations	BO	3	4	even	SFP
simple fixed-point	FX	69	2	even	SFX
word rotate and shift	WS	17	4	even	SFX

Odd:

a	absdb	addx	ah	ahi	ai	and	andbi	andc	andhi	andi	avgb	bg
bgx	ceq	ceqb	ceqbi	ceqh	ceqhi	ceqi	cflts	cfltu	cg	cgt	cgtb	cgtbi
cgth	cgthi	cgti	cgx	clgt	clgtb	clgtbi	clgth	clgthi	clgti	clz	cntb	csflt
cuflt	dfa	dfm	dfma	dfms	dfnma	dfnms	dfs	eqv	fa	fceq	fcgt	fcmeq
fcmtg	fesd	fi	fm	fms	fms	fnms	frds	fs	fscrrd	fscrwr	heq	heqi
hgt	hgti	hlgt	hlgti	il	ila	ilh	ilhu	iohl	mpy	mpya	mpyh	mpyhh
mpyhha	mpyhau	mpyhhu	mpyi	mpys	mpyu	mpyui	nand	nop	nor	or	orbi	orc
orhi	ori	rot	roth	rothi	rothm	rothmi	roti	rotm	rotma	rotmah	rotmah	rotmai
rotmi	selb	sf	sfh	sfhi	sfi	sfx	shl	shlh	shlhi	shli	sumb	xor
xorbi	xorhi	xori	xsbh	xshw	xswd							

Even:

bi	bid	bie	bihnz	bihnzd	bihnze	bihz	bihzd	bihze	binz	binzd	binze
bisl	bisld	bisle	bisled	bisledd	bislede	biz	bizd	bize	br	bra	brasl
brhnz	brhz	brnz	brsl	brz	cbd	cbx	cdd	cdx	chd	chx	cwd
cwv	dsync	frest	frsgest	fsm	fsmb	fsmbi	fsmh	gb	gbb	gbh	hbr
hbrra	hbrrp	hbrr	iret	iretd	irete	lnop	lqa	lqd	lqr	lqx	mfspr
mtspr	orx	rhcnc	rdch	rotqbi	rotqbii	rotqby	rotqbybi	rotqbyi	rotqmbi	rotqmbii	rotqmbby
rotqmbbybi	rotqmbyi	shlqbi	shlqbii	shlqby	shlqbybi	shlqbyi	shufb	stop	stopd	stqa	stqd
stqr	stqx	sync	syncc	wrch							

**Todo:** sort these by latency within each group