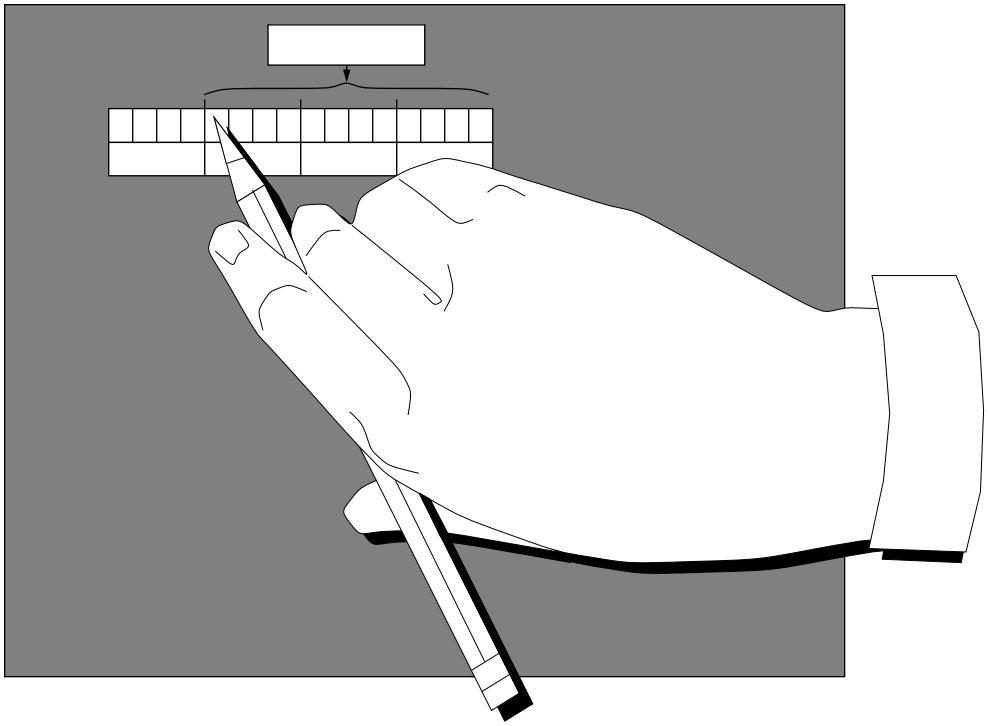
SECTION C

DSP56166 PROGRAMMING SHEETS

The following pages are a set of programming sheets intended to be copied and used to simplify programming the various programmable registers in the DSP56166. They are grouped for the core processor and each peripheral. Each register includes the name, address, reset value, and meaning of each bit. There is room to write in the value for each bit and then the hexadecimal equivalent for each register.



Section Contents

PERIPHERAL ADDRESSES

$FFFF	Reserved for on-chip emulation
$FFFE	
$FFFD	
$FFFC	
$FFFB	
$FFFA	
$FFF9	TX/RX RSSI1 TX/RX Registers
$FFF8	SR/TSR RSSI1 Status Register
$FFF7	
$FFF6	
$FFF5	
$FFF4	
$FFF3	
$FFF2	
$FFF1	TX/RX RSSI0 TX/RX Registers
$FFF0	SR/TSR RSSI0 Status Register
$FFEF	Timer Preload Register (TPR)
$FFEE	Timer Compare Register (TCPR)
$FFED	Timer Count Register (TCTR)
$FFEC	Timer Control Register (TCR)
$FFEB	
$FFEA	
$FFE9	CRX/CTX
$FFE8	COSR
$FFE7	
$FFE6	
$FFE5	HTX/HRX: Host TX/RX Register
$FFE4	HSR: Host Status Register
$FFE3	Port C Data Register (PCD)
$FFE2	Port B Data Register (PBD)
$FFE1	
$FFE0	

$FFDF	IPR: Interrupt Priority Register
$FFDE	BCR: Bus Control Register
$FFDD	IPR2: Interrupt Priority Register 2
$FFDC	PCR1
$FFDB	PCR0
$FFDA	BCR2: Bus Control Register 2
$FFD9	CRB-RSSI1 Control Register B
$FFD8	CRA-RSSI1 Control Register A
$FFD7	
$FFD6	
$FFD5	
$FFD4	
$FFD3	
$FFD2	
$FFD1	CRB-RSSI0 Control Register B
$FFD0	CRA-RSSI0 Control Register A
$FFCF	
$FFCE	
$FFCD	
$FFCC	
$FFCB	
$FFCA	
$FFC9	Reserved
$FFC8	CCR1
$FFC7	CCR0
$FFC6	
$FFC5	
$FFC4	HCR: Host Control Register
$FFC3	Port C Data Direction Register
$FFC2	Port B Data Direction Register
$FFC1	Port C Control Register (PCC)
$FFC0	Port B Control Register (PBC)

On-chip Peripherals Memory Map

INTERRUPT VECTOR ADDRESSES

Interrupt Starting Address	IPL	Interrupt Source
$0000	3	Hardware RESET
$0002	3	Illegal Instruction
$0004	3	Stack Error
$0006	3	Reserved
$0008	3	SWI
$000A	0-2	IRQA
$000C	0-2	IRQB
$000E	0-2	IRQC
$0010	0-2	RSSI0 Receive Data with Exception Status
$0012	0-2	RSSI0 Receive Data
$0014	0-2	RSSI0Transmit Data with Exception Status
$0016	0-2	RSSI0 Transmit Data
$0018	0-2	RSSI1 Receive Data with Exception Status
$001A	0-2	RSSI1 Receive Data
$001C	0-2	RSSI1 Transmit Data with Exception Status
$001E	0-2	RSSI1 Transmit Data
$0020	0-2	Timer Overflow
$0022	0-2	Timer Compare
$0024	0-2	Host DMA Receive Data
$0026	0-2	Host DMA Transmit Data
$0028	0-2	Host Receive Data
$002A	0-2	Host Transmit Data
$002C	0-2	Host Command (default)
$002E	0-2	Codec Receive/Transmit
$0030	0-2	Available for Host Command
$0032	0-2	Available for Host Command
$0034	0-2	Available for Host Command
.	.	
.	.	
.	.	
$007E	0-2	Available for Host Command

Interrupts Starting Addresses and Sources

INSTRUCTIONS

Mnemonic	Syntax	Parallel Moves	Instruction Program Words	Osc. Clock Cycles	SLEUNZVC
ABS	D	(parallel move)	1	2+mv	* * * * * * * –
ADC	S,D	(no parallel move)	1	2	- ** * * * * *
ADD	S,D	(parallel move)	1	2+mv	* * * * * * * *
AND	S,D	(parallel move)	1	2+mv	* * - - ? ? 0 -
AND(I)	#xx,D		1	2	- ? ? ? ? ? ? ?
ASL	D	(parallel move)	1	2+mv	* * * * * * ? ?
ASL4	D	(no parallel move)	1	2	- ? * * * * ? ?
ASR	D	(parallel move)	1	2+mv	* * * * * * 0 ?
ASR4	D	(no parallel move)	1	2	- * * * * * 0 ?
ASR16	D	(no parallel move)	1	2	- * * * * * 0 ?
BFCHG	#iiii,X:<aa> #iiii,X:<pp> #iiii,X:<ea> #iiii,D		2	4+mvb	- * - - - - - ?
BFCLR	#iiii,X:<aa> #iiii,X:<pp> #iiii,X:<ea> #iiii,D		2	4+mvb	- * - - - - - ?
BFSET	#iiii,X:<aa> #iiii,X:<pp> #iiii,X:<ea> #iiii,D		2	4+mvb	- * - - - - - ?
BFTSTH	#iiii,X:<aa> #iiii,X:<pp> #iiii,X:<ea> #iiii,D		2	4+mvb	- * - - - - - ?
BFTSTL	#iiii,X:<aa> #iiii,X:<pp> #iiii,X:<ea> #iiii,D		2	4+mvb	- * - - - - - ?
Bcc	xxxx ee Rn		1+ea	4+jx	- - - - - - - -
BRA	xxxx aa Rn		1+ea	4+jx	- - - - - - - -
BRKcc			1	2/8	- - - - - - - -
BScc	xxxx Rn		1+ea	4+jx	- - - - - - - -
BSR	xxxx Rn		1+ea	4+jx	- - - - - - - -
CHKAAU		(no parallel move)	1	2	- - - - ? ? ? -
CLR	D	(parallel move)	1	2+mv	* * * * * * 0 -
CLR24	D	(parallel move)	1	2+mv	- * * * * * * -
CMP	S,D	(parallel move)	1	2+mv	* * * * * * * *
CMPM	S,D	(parallel move)	1	2+mv	* * * * * * * *
DEBUG			1	4	- - - - - - - -
DEBUGcc			1	4	- - - - - - - -
DEC	D	(parallel move)	1	2+mv	* * * * * * * *
DEC24	D	(parallel move)	1	2+mv	* * * * * ? * *
DIV	S,D	(parallel move)	1	2	- * - - - - ? ?
DMAC(ss,su,uu)	S1,S2,D	(no parallel move)	1	2	- * * * * * * -

Instruction Set Summary

INSTRUCTIONS

Mnemonic	Syntax	Parallel Moves	Instruction Program Words	Osc. Clock Cycles	SLEUNZVC
DO	X:(Rn),expr #xx,expr S,expr	. . . . .	2	6/10+mv	- * - - - - -
DOFOREVER		expr.	2	6	- - - - - - - -
ENDDO		. . . . .	1	2	- - - - - - - -
EOR	S,D	(parallel move)	1	2+mv	* * - - ? ? 0 -
EXT	D	(no parallel move).	1	2	- * * * * * -
ILLEGAL		(no parallel move).	1	8	- - - - - - - -
IMAC	S1,S2,D	(no parallel move).	1	2	- * ? ? * ? ? -
IMPY	S1,S2,D	(no parallel move).	1	2	- * ? ? * ? ? -
INC	D	(parallel move)	1	2+mv	* * * * * * * *
INC24	D	(parallel move)	1	2+mv	* * * * * ? * *
Jcc	xxxx (Rn)	. . . . .	1+ea	4+jx	- - - - - - - -
JMP	xxxx (Rn)	. . . . .	1+ea	4+jx	- - - - - - - -
JScc	xxxx Rn	. . . . .	1+ea	4+jx	- - - - - - - -
JSR	xxxx	. . . . .	1+ea	4+jx	- - - - - - - -
LSL	D	(parallel move)	1	2+mv	* * - - ? ? 0 ?
LSR	D	(parallel move)	1	2+mv	* * - - ? ? 0 ?
MAC	(±)S2,S1,D S1,S2,D S1,S2,D	(one parallel move). . . . . (two parallel reads) D,X:(Rn)+NnS,D	1	2+mv	* * * * * * * -
MACR	(±)S2,S1,D S1,S2,D	(one parallel operation). . (two parallel reads)	1	2+mv	* * * * * * * -
MAC(uu,su)	S1,S2,D	(no parallel move).	1	2	* * * * * * * -
MOVE	 #xxxx,D	(one parallel operation). (double memory read) (memory access, register move)	1+ea	2+mv	* * - - - - - -
No parallel data move		(.....)	mv	mv	- - - - - - - -
Register to register data move		S,D(.....);	mv	mv	* ? - - - - - -
Address register update		(.....)ea	mv	mv	- - - - - - - -
X memory data move		(.....)X:<ea>,D. . . . . (.....)S,X:<ea>.	mv	mv mv	* ? - - - - - -
X memory data move with short displacement		(.....)X:(R2+xx),D . . . . . (.....)S,X:(R2+xx).	mv	mv mv	* ? - - - - - -
X memory data write and register data move (MPY or MAC)		D,X:(Rn)+NnS,D.	mv	mv	* ? - - - - - -
Dual X memory data read		(.....)X:<ea>,D1 X:<ea>,D2.	mv	mv	- - - - - - - -
MOVE(C)	X:<ea>,D S,X:<ea> #xxxx,D S,D X:(R2+xx),D S,X:(R2+xx)	. . . . .	1+ea	2+mvc	* ? ? ? ? ? ? ?
MOVE(I)	#xx,D	. . . . .	1	2	- - - - - - - -

Instruction Set Summary — Continued

INSTRUCTIONS

Mnemonic	Syntax	Parallel Moves	Instruction Program Words	Osc. Clock Cycles	SLEUNZVC
MOVE(M)	P:<ea>,D S,P:<ea> P:(R2+xx),D S,P:(R2+xx) P:<ea>,X:<ea> X:<ea>,P:<ea>	. . . . . 1+ea		2+mvm	* * - - - - -
MOVE(P)	X:<pp>,D X:<pp>,D S,X:<pp> X:<pp>,X:<ea>	. . . . .	1	4+mvp	* * - - - - -
MOVE(S)	X:<aa>,D S,X:<aa>	. . . . .	1	4+mvp	* * - - - - -
MPY	(±)S1,S2,D	(one parallel move)	1	2+mv	* * * * * * * -
	S1,S2,D	(two parallel reads)			
	S1,S2,D	D,X:(Rn)+Nn S,D			
MPYR	(±)S1,S2,D	(one parallel move)	1	2+mv	* * * * * * * -
	S1,S2,D	(two parallel reads)			
MPY(su,uu)	S1,S2,D	(no parallel move)	1	2	- * * * * * * -
NEG	D	(parallel move)	1	2+mv	* * * * * * * *
NEGC	D	(parallel move)	1	2	- * * * * * * *
NOP		. . . . .	1	2	- - - - - - - -
NORM	Rn,D	. . . . .	1	2	- * * * * * ? -
NOT	D	(parallel move)	1	2+mv	* * - - ? ? 0 -
OR	S,D	(parallel move)	1	2+mv	* * - - ? ? 0 -
ORI	#xx,D	. . . . .	1	2	- ? ? ? ? ? ? ?
REP	X:(Rn) #xx S	. . . . .	1	4/6+mv	- - - - - - - -
REPcc		. . . . .	1	4/6	- - - - - - - -
RESET		. . . . .	1	4	- - - - - - - -
RND	D	(parallel move)	1	2+mv	* * * * * * * -
ROL	D	(parallel move)	1	2+mv	* * - - ? ? 0 ?
ROR	D	(parallel move)	1	2+mv	* * - - ? ? 0 ?
RTI		. . . . .	1	4+rx	- ? ? ? ? ? ? ?
RTS		. . . . .	1	4+rx	- - - - - - - -
SBC	S,D	(parallel move)	1	2+mv	* * * * * * * *
STOP		. . . . .	1	n/a	- - - - - - - -
SUB	S,D	(parallel move)	1	2+mv	* * * * * * * *
	S,D	(two parallel reads)			
SUBL	S,D	(parallel move)	1	2+mv	* * * * * * ? *
SWAP	D	(no parallel move)	1	2	- - - - - - - -
SWI		. . . . .	1	8	- - - - - - - -
Tcc	(S,D)	. . . . .	1	2	- - - - - - - -
	S,D	R0,Rn			
TFR	S,D	(one parallel operation) . .	1	2+mv	- - - - - - - -
	S,D	(two memory reads)			
TFR(2)	S,D	(no parallel operation) . . .	1	2	- * - - - - - -
TFR(3)	S1,D1	X:<ea>,D2	1	2+mv	* * - - - - - -
	S1,D1	S2, X:<ea>			
TST	S	(parallel move)	1	2+mv	0 * * * * * 0 0
TST(2)	S	(no parallel move)	1	2	- * * * * * 0 0
WAIT		. . . . .	1	n/a	- - - - - - - -
ZERO	D	(no parallel move)	1	2	- * * * * * * -

Instruction Set Summary — Continued

INSTRUCTIONS

FUNCTIONAL INSTRUCTION SET SUMMARY

DUAL READ INSTRUCTIONS

DSP56166					
DATA ALU OPERATION		**DOUBLE EFFECTIVE ADDRESS**		**DOUBLE DESTINATION**	
Operation	**Registers**	**Read1**	**Read2**	**Dest1**	**Dest2**
MOVE		(Rn)+	(R3)+	~F	X0
MAC/R **MPY/R**	X1, Y1, F X1, Y0, F X0, Y1, F X0, Y0, F	(Rn)+Nn	(R3)+	Y0	X0
		(Rn)+	(R3)+N3	X1	X0
		(Rn)+Nn	(R3)+N3	Y1	X0
		n=[0,2] F = 0 → A F = 1 → B		X0	X1
ADD **SUB** **TFR**	X1, F X0, F Y1, F Y0, F			Y0	X1
				~F	Y0
				Y1	X1
ADD	~F, F				
SUB	~F, F				
TFR	~F, F				

INSTRUCTIONS

LMS INSTRUCTION

DSP56166					
DATA ALU OPERATION		DOUBLE TRANSFER			
Operation	Registers	TRANSFER1		TRANSFER2	
MAC MPY	X0, X0, F	~F	(Rn)+Nn	X1	~F
	X1, X0, F	n=[0,2] F = 0 → A F = 1 → B ~F= opposite accumulator		X0	~F
	A1, Y0, F			Y1	~F
	B1, X0, F			Y0	~F
	Y0, X1, F				
	Y1, X1, F				
	Y1, X0, F				
	Y0, X0, F				

DATA ALU INSTRUCTIONS WITH ONE PARALLEL OPERATION

DSP56166			
DATA ALU OPERATION		PARALLEL MEMORY READ or WRITE	
Operation	Registers	Effective Address	Dest/Source
MAC MPY	±X0, X0, F ±X1, X0, F ±A1, Y0, F ±B1, X0, F ±Y0, X1, F ±Y1, X1, F ±Y1, X0, F ±Y0, X0, F	(Rn)+ (Rn)+Nn (~F1) (R2+xx)	X1 X0 Y1 Y0 A0 B0 A B
		ONE ADDRESS UPDATE	
ADD SUB TFR OR/AND EOR CMP/CMPM	X1, F X0, F Y1, F Y0, F	Effective Address	
		(Rn)-	
		(Rn)+Nn	
		PARALLEL REGISTER TRANSFER	
		Source	Destination
		X0	~F

INSTRUCTIONS

DATA ALU INSTRUCTIONS WITH ONE PARALLEL OPERATION

DSP56166			
DATA ALU OPERATION		**PARALLEL MEMORY READ or WRITE**	
ADD **SUB**	X, F Y, F	X1	~F
		Y0	~F
MOVE		Y1	~F
SBC	X, F Y, F	A	X0
		A	X1
CMP/CMPM **SUBL, TFR** **ADD, SUB**	~F, F	B	Y0
		B	Y1
RND **TST** **ABS** **INC/INC24** **DEC/DEC24** **CLR/CLR24** **NEG** **ASL/ASR**		F	~F
		A0	X0
		A0	X1
		B0	Y0
		B0	Y1
NOT **ROL/ROR** **LSL/LSR** F		No Transfer	

INSTRUCTIONS

BIT FIELD MANIPULATION INSTRUCTIONS

DSP56166		
OPERATION	**OPERAND**	**COMMENTS**
BFTSTH #iiii, **BFTSTL #iiii,** **BFCHG #iiii,** **BFSET #iiii,** **BFCLR #iiii,**	X:(Rn)	n=[0,3]
	X:<aa>	First 32 words of X memory 5 bit address
	X:<pp>	Last 32 words of X memory 5 bit address
	X1, X0, Y1, Y0, R0, R1, R2, R3, N0, N1, N2, N3 M0, M1, M2, M3 A2, B2, A1, B1, A0, B0, A, B SR, OMR, SP, SSH, SSL, LA, LC	

EFFECTIVE ADDRESS UPDATE

DSP56166		
OPERATION	**SOURCE ADDRESS REGISTER**	**DESTINATION REGISTER**
LEA	(Rn) (Rn)+ (Rn)- (Rn)+Nn n=[0,3]	R0, R1, R2, R3 N0, N1, N2, N3

INSTRUCTIONS

JUMP/BRANCH INSTRUCTIONS

DSP56166		
OPERATION	**OPERAND**	**COMMENTS**
JSR **JMP** **Jcc** **JScc**	(Rn)	n=[0,3]
	$xxxx	16-bit absolute address
BSR **BRA** **Bcc** **BScc**	(Rn)	n=[0,3]
	$xxxx	16-bit absolute address
JSR	AA	8-bit absolute address [0,256]
BRA	aa	8-bit PC relative address [-128,+127]
Bcc	ee	6-bit PC relative address [-32,+31]

REP and DO INSTRUCTIONS

DSP56166		
OPERATION	**OPERAND**	**COMMENTS**
REP **DO**	X:(Rn)	n=[0,3]
	#xx	8-bit immediate short data
	X1, X0, Y1, Y0, R0, R1, R2, R3, N0, N1, N2, N3 M0, M1, M2, M3 A2, B2, A1, B1, A0, B0, A, B SR, OMR, SP, SSH, SSL, LA, LC	
REPcc	16 conditions	
DO FOREVER		

INSTRUCTIONS

SHORT IMMEDIATE MOVE INSTRUCTIONS

DSP56166		
OPERATION	**DESTINATION**	**COMMENTS**
MOVE(I) #xx,	X1 X0 Y1 Y0	Immediate short 8-bit signed data (data is put in the least significant byte)

MOVE — PROGRAM and CONTROL INSTRUCTIONS

DSP56166			
OPERATION	**Source/ Destination**	**Destination/ Source**	**COMMENTS**
MOVE(M)	P:(Rn) P:(Rn)+ P:(Rn)- P:(Rn)+Nn P:(R2+xx)	A, A0, B, B0 X0, X1, Y0, Y1	
MOVE(M)	X:(Rn)+ X:(Rn)+Nn	P:(Rn)+ P:(Rn)+Nn	
MOVE(C)	X:(Rn) X:(Rn)+ X:(Rn)- X:(Rn)+Nn X:(Rn+Nn) X:-(Rn) X:#xxxx #xxxx X:(A1) X:(B1) X:(R2+xx)	All registers	X:#xxxx**:** Long 16-bit Absolute address **#xxxx:** Long 16-bit immediate data
MOVE(C)	All registers	All registers	

INSTRUCTIONS

MOVE ABSOLUTE SHORT AND MOVE PERIPHERAL INSTRUCTIONS

DSP56166			
OPERATION	**Source/ Destination**	**Destination/ Source**	**COMMENTS**
MOVE(S)	X:<aa>	A, B, X0, Y0	First 32 words of X memory 5 bit address
MOVE(P)	X:<pp>	A, B, X0, Y0	Last 32 words of X memory 5 bit address
		X:(Rn)+ X:(Rn)+Nn	

TRANSFER WITH PARALLEL MOVE INSTRUCTION

DSP56166				
OPERATION	**REGISTER TRANSFER**		**PARALLEL MOVE**	
	Source	**Destination**	**Source/Dest.**	**Destination/ Source**
TFR(3)	A B	X0, X1, Y0, Y1	X:(Rn)+ X:(Rn)+Nn	X0,X1,Y0,Y1, A0, B0, A, B

INSTRUCTIONS

REGISTER TRANSFER WITHOUT PARALLEL MOVE INSTRUCTION

DSP56166		
OPERATION	**SOURCE**	**DESTINATION**
TFR(2)	A B	X Y

INSTRUCTIONS

REGISTER TRANSFER CONDITIONAL MOVE INSTRUCTION

DSP56166		
OPERATION	**Data ALU**	**Address Reg.**
Tcc	A, F B, F Y0, F X0, F	R0, R0
		R0, Rm

CONDITIONAL PROGRAM CONTROLLER INSTRUCTIONS

DSP56166
OPERATION
BRKcc
DEBUGcc

LOGICAL IMMEDIATE INSTRUCTIONS

DSP56166		
OPERATION	**DESTINATION**	**COMMENTS**
ORI #xx, **ANDI #xx,**	CCR MR OMR	8 bit immediate data

DOUBLE PRECISION DATA ALU INSTRUCTIONS

DSP56166		
DATA ALU OPERATION		
Operation	**sign**	**unsign**

INSTRUCTIONS

DOUBLE PRECISION DATA ALU INSTRUCTIONS

DSP56166	
DATA ALU OPERATION	
DMAC	Y1, X0, F X1, Y1, F X1, Y0, F X0, Y0, F
MPY(su,uu) **MAC(su,uu)**	

INTEGER DATA ALU INSTRUCTIONS

DSP56166	
DATA ALU OPERATION	
Operation	
IMAC **IMPY**	X0, X0, F X1, X0, F A1, Y0, F B1, X0, F Y0, X1, F Y1, X1, F Y1, X0, F Y0, X0, F

DIVISION INSTRUCTION

DSP56166	
DATA ALU OPERATION	
Operation	
DIV	X1, F X0, F Y1, F Y0, F

OTHER DATA ALU INSTRUCTIONS

DSP56166		
OPERATION		
NORM	Rn, F	n=[0,3]
TST2	X1, X0, Y1, Y0	Test data registers
ADC	X, F Y, F	
CHKAAU		Set V,N,Z according to last address ALU operation
ZERO	F	Zero F from bit 32 to 39
EXT	F	Sign extend F from bit 31 to 39
SWAP	F	Swap F1 and F0
NEGC	F	Negate with borrow
ASL4	F	
ASR4	F	
ASR16	F	Move A,A0 arithmetic

SPECIAL INSTRUCTIONS

DSP56166
OPERATION
WAIT
STOP
ENDDO
RESET
RTS
RTI
SWI
DEBUG
NOP

Application: _____ Date: _____

_____ Programmer: _____

CORE

Program Memory Wait States
Set to zero for fast memory.

Data Memory Wait States
Set to zero for fast memory.

Bus State Status — Read Only
0 = DSP **NOT** a Bus Master
1 = DSP a Bus Master

Bus Request Hold
0 = $\overline{BR}$ Asserted By External Access
1 = $\overline{BR}$ Always Asserted

Port A
Bus Control Register (BCR)
X:$FFDE Read/Write
Reset = $43FF

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RH	BS	*	*	*	*	X4	X3	X2	X1	X0	P4	P3	P2	P1	P0
		0	0	0	0	0	0	0	0	0	0	0	0	0	0
				0				0				0			

External Peripheral Wait States
Set to zero for fast memory.

Port A
Bus Control Register 2 (BCR2)
X:$FFDA Read/Write
Reset = $001F

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*	*	*	*	*	*	*	*	*	*	*	P4	P3	P2	P1	P0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
												0			

Carry
Overflow
Zero
Negative
Unnormalized
Extension
Limit
Sticky Bit
Interrupt Mask
Scaling Mode
ForeVer Flag
Loop Flag

Status Register (SR)
Read/Write
Reset = $0300

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LF	FV	*	*	S1	S0	I1	I0	S	L	E	U	N	Z	V	C
		0	0												

MR **CCR**

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

CORE

$\overline{\text{IRQA}}$ Mode

IAL1	IAL0	Enabled	IPL
0	0	No	—
0	1	Yes	0
1	0	Yes	1
1	1	Yes	2

IAL2	Trigger
0	Level
1	Neg. Edge

$\overline{\text{IRQB}}$ Mode

IBL1	IBL0	Enabled	IPL
0	0	No	—
0	1	Yes	0
1	0	Yes	1
1	1	Yes	2

IBL2	Trigger
0	Level
1	Neg. Edge

Codec IPL
0 = Lowest Level
3 = Unmaskable

Host IPL
0 = Lowest Level
3 = Unmaskable

SSI0 IPL
0 = Lowest Level
3 = Unmaskable

SSI1 IPL
0 = Lowest Level
3 = Unmaskable

Timer IPL
0 = Lowest Level
3 = Unmaskable

Interrupt Priority Register (IPR)
X:$FFDF Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TL1	TL0	S1L1	S1L0	S0L1	S0L0	HL1	HL0	CL1	CL0	IBL2	IBL1	IBL0	IAL2	IAL1	IAL0

Application: _____ Date: _____

_____ Programmer: _____

CORE

$\overline{\text{IRQC}}$ Mode			
ICL2	**ICL1**	**Enabled**	**IPL**
0	0	Not enabled	—
0	1	Level	1
1	0	Not enabled	—
1	1	Neg. Edge	1

Interrupt Priority Register 2(IPR2)
X:$FFDD Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
* 0	* 0	* 0	* 0	* 0	* 0	* 0	* 0	* 0	* 0	ICL2	ICL1	* 0	* 0	* 0	* 0
0				0								0			

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

CORE

$\overline{\text{IRQC}}$ Mode

ICL2	ICL1	Enabled	IPL
0	0	Not enabled	—
0	1	Level	1
1	0	Not enabled	—
1	1	Neg. Edge	1

Interrupt Priority Register 2(IPR2)
X:$FFDD Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*	*	*	*	*	*	*	*	*	*	ICL2	ICL1	*	*	*	*
0	0	0	0	0	0	0	0	0	0			0	0	0	0
0				0								0			

Operating Mode
00 = Boot: Byte-wide at P:$C000
01 = Boot: Host or SSI0
10 = Int. Mem; Reset at P:$E000
11 = Ext. Mem; Reset at P:$0000

Bus Arbitration Mode
0 = Slave
1 = Master

External X Memory
0 = Internal X Memory enabled
1 = Internal X Memory disabled

Saturation
0 = Disable
1 = Enable

Rounding
0 = Convergent Rounding
1 = Two's Complement Rounding

Stop Delay
0 = 524K T Stabilization
1 = 28 T Stabilization

Clock Out
0 = Clock on CLKO Pin
1 = Disable

Operating Mode Register (OMR)
Read/Write
Reset = $000x

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*	*	*	*	*	*	*	*	CD	SD	R	SA	EX	MC	MB	MA
0	0	0	0	0	0	0	0								
0				0											

* = Reserved, Program as zero

DSP56166 Phase Locked Loop Programming Sheet

Application: _____ Date: _____

_____ Programmer: _____

P.L.L.

Clockout Select

CS1	CS0	CLKOUT
0	0	PH0
0	1	Reserved
1	0	Squared F_{ext}
1	1	Squared F_{ext}÷2

Phase Select Bit
0 = ID divider bypassed
1 = ID divider used

PLL Power Down
0 = Off
1 = On

PLL Enable
0 = Disable
1 = Enable

VCO Lock – Read Only
0 = NOT Locked
1 = Locked

PLL Control Register 1(PCR1)
X:$FFDC Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Lock	PLLE	PLLD	PS	CS1	CS0	*	*	*	*	*	*	*	*	*	*
						0	0	0	0	0	0	0	0	0	0
								0				0			

Feedback Divider
Multiplies Clock Frequency by any value from 1 to 256

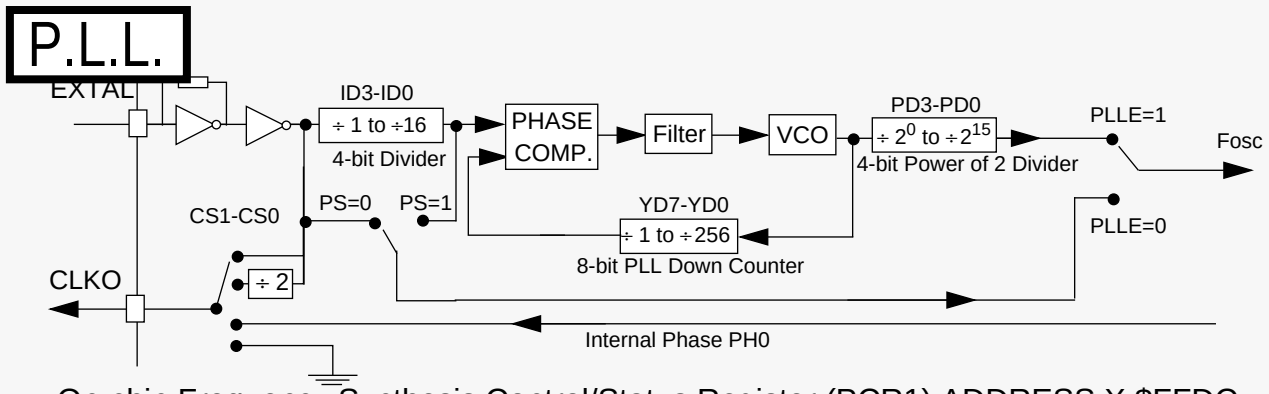
Input Divider
Divides Clock Frequency by 1 to 16

Power Down Bits
Divide by any power of 2 between 2^0 and 2^{15}

PLL Control Register 0 (PCR0)
X:$FFDB Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD3	PD2	PD1	PD0	ID3	ID2	ID1	ID0	YD7	YD6	YD5	YD4	YD3	YD2	YD1	YD0

* = Reserved, Program as zero



On-chip Frequency Synthesis Control/Status Register (PCR1) ADDRESS X:$FFDC

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOCK	PLLE	PLLD	PS	CS1	CS0	**	**	**	**	**	**	**	**	**	**

LOCK	0	PLL unlocked
	1	PLL locked
PLLE PLLD	00	PLL active but not used as Fosc
	01	PLL powered down
	10	PLL active and used as Fosc
	11	Reserved
Phase Select	0	Squared EXTAL selected as Fosc if PLLE=0
	1	Squared EXTAL/ID selected as Fosc if PLLE=0
CS1-CS0 CLKO Select	00	PH0 output to CLKO when enabled by the CD bit (bit 7) of the OMR
	01	Reserved
	10	Fext output to CLKO when enabled by the CD bit (bit 7) of the OMR
	11	Fext/2 output to CLKO when enabled by the CD bit (bit 7) of the OMR

On-chip Frequency Synthesis Control/Status Register (PCR0) ADDRESS X:$FFDB

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD3	PD2	PD1	PD0	ID3	ID2	ID1	ID0	YD7	YD6	YD5	YD4	YD3	YD2	YD1	YD0

PD3-PD0 Clock Output Divider	$0	Divide the VCO output clock by 1 (2^0)	8	Divide the VCO output clock by 256 (2^8)
	$1	Divide the VCO output clock by 2 (2^1)	9	Divide the VCO output clock by 512 (2^9)
	$2	Divide the VCO output clock by 4 (2^2)	A	Divide the VCO output clock by 1024 (2^{10})
	$3	Divide the VCO output clock by 8 (2^3)	B	Divide the VCO output clock by 2048 (2^{11})
	$4	Divide the VCO output clock by 16 (2^4)	C	Divide the VCO output clock by 4096 (2^{12})
	$5	Divide the VCO output clock by 32 (2^5)	D	Divide the VCO output clock by 8192 (2^{13})
	$6	Divide the VCO output clock by 64 (2^6)	E	Divide the VCO output clock by 16384 (2^{14})
	$7	Divide the VCO output clock by 128 (2^7)	F	Divide the VCO output clock by 32768 (2^{15})
ID3-ID0 Input Clock Divider	$0	Divide the input clock by 1	8	Divide the input clock by 9
	$1	Divide the input clock by 2	9	Divide the input clock by 10
	$2	Divide the input clock by 3	A	Divide the input clock by 11
	$3	Divide the input clock by 4	B	Divide the input clock by 12
	$4	Divide the input clock by 5	C	Divide the input clock by 13
	$5	Divide the input clock by 6	D	Divide the input clock by 14
	$6	Divide the input clock by 7	E	Divide the input clock by 15
	$7	Divide the input clock by 8	F	Divide the input clock by 16
YD7-YD0 VCO Down Counter Value	$YD	Multiplies by YD+1		

On-chip Frequency Synthesizer Programming Model

Application: _____ Date: _____

_____ Programmer: _____

Sheet 1 of 1

Timer

Timer Output Enable
000 = TOUT Disabled
001 = Compare/Overflow Pulse
010 = Overflow Pulse
011 = Compare Pulse
100 = Overflow/Compare Toggle
101 = Compare/Overflow Toggle
110 = Overflow Toggle
111 = Compare Toggle

Inverter Bit
0 = Do **NOT** Invert TIN Pin Signal
1 = Invert TIN Pin Signal

Timer Enable
0 = Disable Timer
1 = Enable Timer

Compare Interrupt Enable
0 = Disable interrupt
1 = Interrupt DSP after TCTR = TCPR

Overflow Interrupt Enable
0 = Disable interrupt
1 = Interrupt DSP when TCTR = 0

Event Select
0 = Fosc/2 is event clock
1 = TIN is event clock

Decrement Ratio
(Count Register Prescaler)

Timer Control Register (TCR)
X:$FFEC Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE	INV	TO2	TO1	TO0	CIE	OIE	ES	DC7	DC6	DC5	DC4	DC3	DC2	DC1	DC0

Count — Decremented when TPR = 0

Timer Count Register (TCTR)
X:$FFED Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Compare Value — Compare with Count Register

Timer Compare Register (TCPR)
X:$FFEE Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Number to Load into Count Register

Timer Preload Register (TPR)
X:$FFEF Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Application: _____ Date: _____

_____ Programmer: _____

Sheet 1 of 2

Codec

Input Select
0 = MIC Selected
1 = AUX Selected

Codec Enable
0 = Disabled
1 = Enable

Codec Interrupt Enable
0 = Disabled
1 = Enabled

Microphone Gain Select Bits
00 = -6dB
01 = 0dB
10 = 6dB
11 = 17dB

Codec Loop Back Bit
0 = Normal operation
1 = A/D mod. input D/A mod.

Clock Select Bit
0 = Squared Fext as Input Clock
1 = PLL output as Input Clock

Mute Bit
0 = Output Muted
1 = Output **NOT** Muted

Audio Level Control Bits

0000 = -15dB	1000 = 5dB
0001 = -10dB	1001 = 11dB
0010 = -5dB	1010 = 17dB
0011 = 0dB	1011 = 23dB
0100 = 5dB	1100 = 29dB
0101 = 11dB	1101 = 35dB
0110 = 17dB	1110 = 35dB
0111 = 23dB	1111 = 40dB

Codec Control Register 1(COCR1)
X:$FFC8 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COIE	COE	INS	MGS1	MGS0	MUT	* 0	CLS	CLB	* 0	* 0	* 0	VC3	VC2	VC1	VC0

Codec Ratio Select Bits
Select any decimation/interpolation ratio values between 65 to 128

Input Divider Bits
Divide the input clock to the codec by any value between 1 and 64

Codec Control Register 0 (COCR0)
X:$FFC7 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
* 0	* 0	CRS5	CRS4	CRS3	CRS2	CRS1	CRS0	* 0	* 0	ED5	ED4	ED3	ED2	ED1	ED0

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

Codec

Codec Transmit Data Empty
0 = Wait
1 = Write Data

Codec Transmit Underrun Error Flag
0 = OK
1 = Error

Codec Receive Data Full
0 = Wait
1 = Read Data

Codec Receive Overrun Error Flag
0 = OK
1 = Error

Codec Status Register (COSR)
X:$FFE8 Read Only
Reset = $04

7	6	5	4	3	2	1	0
*	*	*	*	CRDF	CTDE	CROE	CTUE
0	0	0	0				

Load Under Program Control

Transmit Data Register (CTX)
X:$FFE9 Write Only
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Read Under Program Control

Receive Data Register (CRX)
X:$FFE9 Read Only
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

GP I/O

Port B

Port B Control
0 = General Purpose I/O
1 = Host Interface

Port B
Control Register (PBC)
X:$FFC0 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	BC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0				0				0				0			

Port B Data Direction Control
0 = Input
1 = Output

Port B
Data Direction Register (PBDDR)
X:$FFC2 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*	DB14	DB13	DB12	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0															

Port B Data (usually loaded by program)

Port B
Data Register (PBD)
X:$FFE2 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*	PB14	PB13	PB12	PB11	PB10	PB9	PB8	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
0															

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

GP I/O

Port C

Port C Pin Control
0 = General Purpose I/O Pin
1 = Peripheral Pin

Port C
Control Register (PCC)
X:$FFC1 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
* 0	* 0	* 0	* 0	CC11	CC10	CC9	* 0	CC7	CC6	CC5	CC4	* 0	CC2	CC1	CC0
0															

Port C Data Direction Control
0 = Input
1 = Output

Port C
Data Direction
Register (PCDDR)
X:$FFC3 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
* 0	* 0	* 0	* 0	DC11	DC10	DC9	* 0	DC7	DC6	DC5	DC4	* 0	DC2	DC1	DC0
0															

Port C Data (usually loaded by program)

Port C
Data Register (PCD)
X:$FFE3 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
* 0	* 0	* 0	* 0	PC11	PC10	PC9	* 0	PC7	PC6	PC5	PC4	* 0	PC2	PC1	PC0
0															

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

HOST

Port B

Port B Control
0 = General Purpose I/O
1 = Host Interface

Port B
Control Register (PBC)
X:$FFC0 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	BC
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0				0				0				1			

HOST – DSP SIDE

Host Receive Interrupt Enable
0 = Disable 1 = Enable — Interrupt on HRDF

Host Transmit Interrupt Enable
0 = Disable 1 = Enable — Interrupt on HTDE

Host Command Interrupt Enable
0 = Disable 1 = Enable — Interrupt on HCP

Host Flags
General Purpose Read/Write Flags

Host Control Register (HCR)
X:$FFC4 Read/Write
Reset = $00

7	6	5	4	3	2	1	0
*	*	*	HF3	HF2	HCIE	HTIE	HRIE
0	0	0					

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

HOST

HOST – DSP SIDE

Host Receive Data Full
0 = Wait 1 = Read

Host Transmit Data Empty
0 = Wait 1 = Write

Host Command Pending
0 = Wait 1 = Ready

Host Flags
Read Only

DMA Status (Read Only)
0 = Disabled 1 = Enabled

Host Status Register (HSR)
X:$FFE4 Read Only
Reset = $02

7	6	5	4	3	2	1	0
DMA	* 0	* 0	HF1	HF0	HCP	HTDE	HRDF

Host Receive Data (usually Read by program)

Host Receive Data Register (HRX)
X:$FFE5 Read Only
Reset = $xxxx

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HIGH BYTE								LOW BYTE							

Host Transmit Data (usually loaded by program)

Host Transmit Data Register (HTX)
X:$FFE5 Write Only
Reset = $xxxx

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HIGH BYTE								LOW BYTE							

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

HOST

HOST – HOST PROCESSOR SIDE

Receive Request Enable
DMA Off 0 = Interrupts Disabled 1 = Interrupts Enabled
DMA On 0 = Host → DSP 1 = DSP → Host

Transmit Request Enable
DMA Off 0 = Interrupts Disabled 1 = Interrupts Enabled
DMA On 0 = DSP → Host 1 = Host → DSP

Host Flags
Write Only

Host Mode Control
00 = DMA Off 01 = Illegal
10 = 16 Bit DMA 11 = 8 Bit DMA

Initialize (Write Only)
0 = No Action 1 = Initialize DMA

Interrupt Control Register (ICR)
$0 Read/Write
Reset = $00

7	6	5	4	3	2	1	0
INIT	HM1	HM0	HF1	HF0	* 0	TREQ	RREQ

Host Vector
Executive Interrupt Routine 0-63

Host Command
0 = Idle 1 = Interrupt DSP

Command Vector Register (CVR)
$1 Read/Write
Reset = $16

7	6	5	4	3	2	1	0
HC	* 0	HV5	HV4	HV3	HV2	HV1	HV0

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

HOST – HOST PROCESSOR SIDE

Receive Data Register Full
0 = Wait 1 = Read

Transmit Data Register Empty
0 = Wait 1 = Write

Transmitter Ready
0 = Data in HI 1 = Data Not in HI

Host Flags
Read Only

DMA Status
0 = DMA Disabled 1 = DMA Enabled

Host Request
0 = $\overline{\text{HREQ}}$ Deasserted 1 = $\overline{\text{HREQ}}$ Asserted

Interrupt Status Register (ISR)
$2 Read/Write
Reset = $03

7	6	5	4	3	2	1	0
HREQ	DMA	* 0	HF3	HF2	TRDY	TXDE	RXDF

Exception vector number for use by MC68000 processor family vectored interrupts.

Interrupt Vector Register (IVR)
$3 Read/Write
Reset = $0F

7	6	5	4	3	2	1	0
IV7	IV6	IV5	IV4	IV3	IV2	IV1	IV0

Host Receive Data (usually read by program)

Receive Byte Registers
$6, $7 Read Only
Reset = $xxxx

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HIGH BYTE								LOW BYTE							

Host Transmit Data (usually loaded by program)

Transmit Byte Registers
$6, $7 Write Only
Reset = $xxxx

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HIGH BYTE								LOW BYTE							

* = Reserved, Program as zero

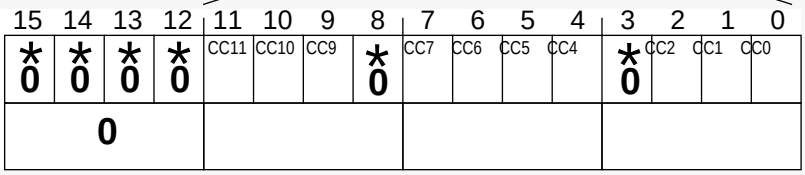
Application: _____ Date: _____

_____ Programmer: _____

RSSI

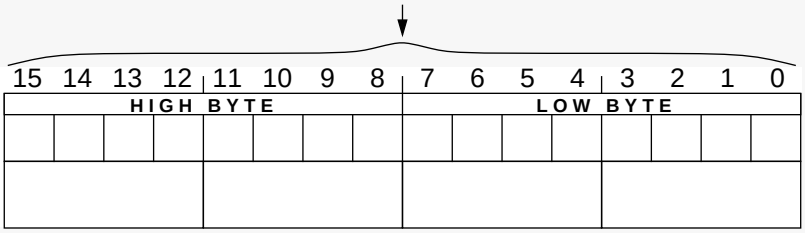
RSSI Port C Pin Control
0 = General Purpose I/O Pin
1 = RSSI Pin

PORT C
RSSI Control Register (PCC)
X:$FFC1 Read/Write
Reset = $0000



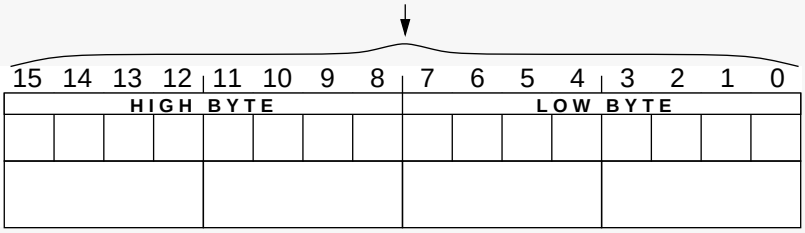
RSSI Receive Data (usually read by program)

RSSI Serial Receive Register
RSSI0 Address X:$FFF1 Read Only
RSSI1 Address X:$FFF9 Read Only
Reset = $xxxx



RSSI Transmit Data (usually loaded by program)

RSSI Serial Transmit Register
RSSI0 Address X:$FFF1 Write Only
RSSI1 Address X:$FFF9 Write Only
Reset = $xxxx



* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

RSSI

Word Length Control
00 = 8 Bits/Word
01 = Reserved
10 = 12 Bits/Word
11 = 16 Bits/Word

Prescaler Range
0 = / 1
1 = / 8

Frame Rate Divider Control
000 = 1
111 = 8

Prescale Modulus Select

RSSI Control Register A (CRA)
RSSI0 Address $FFD0 Read/Write
RSSI1 Address $FFD8 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSR	WL1	WL0	* 0	* 0	DC2	DC1	DC0	PM7	PM6	PM5	PM4	PM3	PM2	PM1	PM0

Clock Source Direction
0 = External Clock 1 = Internal Clock

Clock Polarity
0 = Data Out ↑, Data In ↓ 1 = Data Out ↓, Data In ↑

MSB Position
0 = MSB First 1 = LSB First

Frame Sync Length
0 = Word Sync 1 = Bit Sync

Frame Sync Invert
0 = Active High 1 = Active Low

Enable Bit
0 = RSSI Disable 1 = RSSI Enable

Mode Select
0 = Normal 1 = Network

Transmit Enable
0 = Disable 1 = Enable

Receive Enable
0 = Disable 1 = Enable

Transmit Interrupt Enable
0 = Disable 1 = Enable

Receive Interrupt Enable
0 = Disable 1 = Enable

FS Direction Bit:
0 = FS Input
1 = FS Output

Gated Clock Bit:
0 = Continuous Clock
1 = Gated Clock

Early FS Bit:
0 = FS on the first bit
1 = FS one bit earlier

RSSI Control Register B (CRB)
RSSI0 Address $FFD1 Read/Write
RSSI1 Address $FFD9 Read/Write
Reset = $0000

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RIE	TIE	RE	TE	MOD	SSIEN	FSI	FSL	SHFD	SCKP	SCKD	FSD	GCK	ELFS	* 0	* 0

* = Reserved, Program as zero

Application: _____ Date: _____

_____ Programmer: _____

RSSI

Transmit/Receive Frame Sync
0 = Wait 1 = Read RX

Transmitter Underrun Error
0 = OK 1 = Error

Receiver Overrun Error
0 = OK 1 = Error

Transmit Data Register Empty
0 = Wait 1 = Write

Receive Data Register Full
0 = Wait 1 = Read

RSSI STATUS REGISTER (SSISR)
RSSI0 ADDRESS $FFF0 Read Only
RSSI1 ADDRESS $FFF8 Read Only
Reset = $00

7	6	5	4	3	2	1	0
RDF	TDE	ROE	TUE	TRFS	* 0	* 0	* 0

* = Reserved, Program as zero