Freescale Semiconductor, Inc.
MOTOROLA
Consumer Systems Group

Order this document by
MCF5202UMAD/AD

MCF5202

Addendum to **MCF5202 User Manual**

April 14, 1998

This addendum to the initial release of the MCF5202UM/AD User's Manual provides corrections to the original text, plus additional information not included in the original. This document and other information on this product is maintained on the World Wide Web at http://sps.motorola.com/coldfire

Correction to DC Electrical Specifications

The units stated for Input Leakage current & Hi-Z Leakage current are incorrect; they should read uA, and not mA.

Table 1. DC ELECTRICAL SPECIFICATIONS

CHARACTERISTIC	SYMBOL	MIN	MAX	UNIT
Input high voltage	V_{IH}	2	V_{CC}	V
Input low voltage	V_{IL}	GND	0.8	V
Input leakage current @ GND, V_{CC}	I_{In}	–	20	uA
HI-Z (three-state) leakage current @GND, V_{CC}	I_{TSI}	–	20	uA
Output high voltage, I_{OH} =5mA	V_{OH}	2.4	–	V
Output low voltage , I_{OL} =5mA(	V_{OL}	–	0.5	V
Pin capacitance*	C_{In}	–	10	pF

* This specification periodically sampled but not 100% tested.

Power Consumption - Addition to Electrical Specifications

The power consumption figures stated are for 5.0 V and 50 pf loads on all pins, room temperature. The code which was used was Dhrystone 2.1. The data is as follows:

Table 2: MCF5202 Power Consumption

16MHZ	25MHz	33MHz	UNITS
302	448	589	mW

Instruction Set Architecture

The Instruction Set Architecture tables shown on pages 1-9 through 1-14 should be replaced with the following :

Notational Conventions

OPCODE WILDCARDS	
cc	Logical Condition (example: NE for not equal)
REGISTER OPERANDS	
An	Any Address Register n (example: A3 is address register 3)
Ay,Ax	Source and destination address registers, respectively
Dn	Any Data Register n (example: D5 is data register 5)
Dy,Dx	Source and destination data registers, respectively
Rn	Any Address or Data Register
Ry,Rx	Any source and destination registers, respectively
Rw	Any second destination register
Rc	Any Control Register (example VBR is the vector base register)
REGISTER/PORT NAMES	
ACC	MAC Accumulator
DDATA	Debug Data Port
CCR	Condition Code Register (lower byte of status register)
MACSR	MAC Status Register
MASK	Mask Register
PC	Program Counter
PST	Processor Status Port
SR	Status Register
MISCELLANEOUS OPERANDS	
#<data>	Immediate data following the instruction word(s)
<ea>	Effective Address
<ea>y,<ea>x	Source and Destination Effective Addresses, respectively
<label>	Assembly Program Label
<list>	List of registers (example: D3–D0)
<size>	Operand data size: Byte (B), Word (W), Longword (L)
OPERATIONS	
+	Arithmetic addition or postincrement indicator
–	Arithmetic subtraction or predecrement indicator
x	Arithmetic multiplication

Notational Conventions (Continued)

/	Arithmetic division
~	Invert; operand is logically complemented
&	Logical AND
I	Logical OR
~	Logical exclusive OR
<<	Shift left (example: D0 << 3 is shift D0 left 3 bits)
>>	Shift right (example: D0 >> 3 is shift D0 right 3 bits)
→	Source operand is moved to destination operand
←→	Two operands are exchanged
sign-extended	All bits of the upper portion are made equal to the high-order bit of the lower portion
If <condition> then <operations> else <operations>	Test the condition. If true, the operations after 'then' are performed. If the condition is false and the optional 'else' clause is present, the operations after 'else' are performed. If the condition is false and else is omitted, the instruction performs no operation. Refer to the Bcc instruction description as an example.
SUBFIELDS AND QUALIFIERS	
{}	Optional Operation
()	Identifies an indirect address
d_n	Displacement Value, n-Bits Wide (example: d_{16} is a 16-bit displacement)
Address	Calculated Effective Address (pointer)
Bit	Bit Selection (example: Bit 3 of D0)
LSB	Least Significant Bit (example: MSB of D0)
LSW	Least Significant Word
MSB	Most Significant Bit
MSW	Most Significant Word
CONDITION CODE REGISTER BIT NAMES	
P	Branch Prediction Bit in CCR
C	Carry Bit in CCR
N	Negative Bit in CCR
V	Overflow Bit in CCR
X	Extend Bit in CCR
Z	Zero Bit in CCR

Instruction Set Summary

INSTRUCTION	OPERAND SYNTAX	OPERAND SIZE	OPERATION
ADD	Dy,<ea>x <ea>y,Dx	32 32	Source + Destination → Destination
ADDA	<ea>y,Ax	32	Source + Destination → Destination
ADDI	#<data>,Dx	32	Immediate Data + Destination → Destination
ADDQ	#<data>,<ea>x	32	Immediate Data + Destination → Destination
ADDX	Dy,Dx	32	Source + Destination + X → Destination
AND	Dy,<ea>x <ea>y,Dx	32 32	Source & Destination → Destination
ANDI	#<data>,Dx	32	Immediate Data & Destination → Destination
ASL	Dx,Dy #<data>,Dx	32 32	X/C ← (Dy << Dx) ← 0 X/C ← (Dy << #<data>) ← 0
ASR	Dx,Dy <data>,Dx	32 32	MSB → (Dy >> Dx) → X/C MSB → (Dy >> #<data>) → X/C
Bcc	<label>	8,16	If Condition True, Then PC + d_n → PC

Instruction Set Summary (Continued)

INSTRUCTION	OPERAND SYNTAX	OPERAND SIZE	OPERATION
BCHG	Dy,<ea>x #<data>,<ea>x	8,32 8,32	~(<Bit Number> of Destination) → Z, Bit of Destination
BCLR	Dy,<ea>x #<data>,<ea>x	8,32 8,32	~(<Bit Number> of Destination) → Z; 0 → Bit of Destination
BRA	<label>	8,16	PC + d_n → PC
BSET	Dy,<ea>x #<data>,<ea>x	8,32 8,32	~(<Bit Number> of Destination) → Z; 1→ Bit of Destination
BSR	<label>	8,16	SP – 4 → SP; next sequential PC→ (SP); PC + d_n → PC
BTST	Dy,<ea>x #<data>,<ea>x	8,32 8,32	~(<Bit Number> of Destination) → Z
CLR	<ea>x	8,16,32	0 → Destination
CMPI	#<data>,Dx	32	Destination – Immediate Data
CMP	<ea>y,Dx	32	Destination – Source
CMPA	<ea>y,Ax	32	Destination - Source
CPUSH	(An)	32	Push and Invalidate Cache Line
DIVS	<ea>y,Dx	16 32	Dx / <ea>y → Dx {16-bit Remainder; 16-bit Quotient} Dx / <ea>y → Dx {32-bit Quotient} Signed operation
DIVU	<ea>y,Dx	16	Dx / <ea>y → Dx {16-bit Remainder; 16-bit Quotient} Dx / <ea>y → Dx {32-bit Quotient} Unsigned operation
EOR	Dy,<ea>x	32	Source ~ Destination → Destination
EORI	#<data>,Dx	32	Immediate Data ~ Destination → Destination
EXT	Dx Dx	8 → 16 16 → 32	Sign-Extended Destination → Destination
EXTB	Dx	8 → 32	Sign-Extended Destination → Destination
HALT	none	none	Enter Halted State
JMP	<ea>	none	Address of <ea> → PC
JSR	<ea>	32	SP– 4 → SP; next sequential PC → (SP); <ea> → PC
LEA	<ea>y,Ax	32	<ea> → Ax
LINK	Ax,#<data>	16	SP – 4 → SP; Ax → (SP); SP → Ax; SP + d16 → SP
LSL	Dx,Dy #<data>,Dx	32 32	X/C ← (Dy << Dx) ← 0 X/C ← (Dx << #<data>) ← 0
LSR	Dx,Dy #<data>,Dx	32 32	0 → (Dy >> Dx) → X/C 0 → (Dx >> #<data>) → X/C
MAC	Ry,Rx <shift> Ry,Rx<shift>,<ea>y,Rw	16 × 16 + 32 → 32 32 → 32	ACC + (Ry × Rx){<< 1 | >> 1} → ACC ACC + (Ry × Rx){<< 1 | >> 1} → ACC; (<ea>y{&MASK}) → Rw
MACL	Ry,Rx<shift> Ry,Rx,<shift>,<ea>y,Rw	32 × 32 + 32 → 32 32 → 32	ACC + (Ry × Rx){<< 1 | >> 1} → ACC ACC + (Ry × Rx){<< 1 | >> 1} → ACC; (<ea>y{&MASK}) → Rw
MOVE	<ea>y,<ea>x	8,16,32	<ea>y → <ea>x
MOVE from ACC	ACC,Rx	32	ACC → Rx
MOVE from CCR	Dx	16	CCR → Dx
MOVE from MACSR	MACSR,Rx MACSR,CCR	32 8	MACSR → Rx MACSR →CCR
MOVE from MASK	MASK,Rx	32	MASK → Rx
MOVE from SR	Dx	16	SR → Dx
MOVE to ACC	Ry,ACC <#<data>,ACC	32 32	Ry → ACC #<data> → ACC
MOVE to CCR	Dy,CCR #<data>,CCR	8	Dy → CCR #<data> → CCR

Instruction Set Summary (Continued)

INSTRUCTION	OPERAND SYNTAX	OPERAND SIZE	OPERATION
MOVE to MACSR	Ry,MACSR #<data>,MACSR	32	Ry → MACSR #<data> → MACSR
MOVE to MASK	Ry,MASK #<data>,MASK	32 32	Ry → MASK #<data> → MASK
MOVE to SR	Dy,SR #<data>,SR	16	Source → SR
MOVEA	<ea>y,Ax	16,32 → 32	Source → Destination
MOVEC	Ry,Rc	32	Ry → Rc
MOVEM	list,<ea>x <ea>y,list	32 32	Listed Registers → Destination Source → Listed Registers
MOVEQ	#<data>,Dx	8 → 32	Sign-extended Immediate Data→ Destination
MSAC	Ry,Rx<shift> Ry,Rx<shift>,<ea>y,Rw	32 - 16 × 16 → 32 32 → 32	ACC - (Ry × Rx){<< 1 | >> 1} → ACC ACC - (Ry × Rx){<< 1 | >> 1} → ACC, (<ea>y{&MASK}) → Rw
MSACL	Ry,Rx<shift> Ry,Rx<shift>,<ea>y,Rw	32 - 32 × 32 → 32 32 → 32	ACC - (Rw × Rx){<< 1 | >> 1} → ACC ACC - (Rw × Rx){<< 1 | >> 1} → ACC; (<ea>y{&MASK}) → Rw
MULS	<ea>y,Dx	16 x 16 → 32 32 x 32 → 32	Source × Destination → Destination Signed operation
MULU	<ea>y,Dx	16 x 16 → 32 32 x 32 → 32	Source × Destination → Destination Unsigned operation
NEG	<ea>x	32	0 – Destination → Destination
NEGX	<ea>x	32	0 – Destination– X → Destination
NOP	none	none	PC + 2 → PC; Synchronize Pipelines
NOT	<ea>	32	~ Destination → Destination
OR	Dy,<ea>x <ea>y,Dx	32	Source I Destination → Destination
ORI	#<data>,Dx	32	Immediate Data I Destination → Destination
PEA	<ea>	32	SP – 4 → SP; Address of <ea> → (SP)
PULSE	none	none	Set PST= $4
REMS	<ea>y,Dx:Dw	32	Dx/<ea>y → Dw {32-bit Remainder} Signed operation
REMU	<ea>y,Dx:Dw	32	Dx/<ea>y → Dw {32-bit Remainder} Unsigned operation
RTE	none	none	(SP+2) → SR; SP+4 → SP; (SP) → PC; SP + FormatField → SP
RTS	none	none	(SP) → PC; SP + 4 → SP
Scc	Dx	8	If Condition True, Then 1's → Destination; Else 0's → Destination
STOP	#<data>	16	Immediate Data → SR; Enter Stopped State
SUB	Dy,<ea>x <ea>y,Dx	32 32	Destination - Source→ Destination
SUBA	<ea>y,Ax	32	Destination - Source→ Destination
SUBI	#<data>,Dx	32	Destination – Immediate Data → Destination
SUBQ	#<data>,<ea>x	32	Destination - Immediate data → Destination
SUBX	Dy,Dx	32	Destination – Source – X → Destination
SWAP	Dn	16	MSW of Dn ←→ LSW of Dn
TRAP	none	none	SP – 4 → SP;PC → (SP); SP – 2 → SP;SR → (SP); SP – 2 → SP; Format → (SP); Vector Address → PC
TRAPF	none #<data>	none 16 32	PC + 2 → PC PC + 4 → PC PC + 6 → PC
TST	<ea>y	8,16,32	Set Condition Codes
UNLK	Ax	32	Ax →SP; (SP) → Ax; SP + 4 → SP

Instruction Set Summary (Continued)

INSTRUCTION	OPERAND SYNTAX	OPERAND SIZE	OPERATION
WDDATA	<ea>y	8,16,32	<ea>y →DDATA port
WDEBUG	<ea>y	2 x 32	<ea>y → Debug Module

Literature Distribution Centers:
USA/EUROPE: Motorola Literature Distribution; P.O. Box 20912, Arizona 85036.
JAPAN: Nippon Motorola Ltd.; 4-32-1, Nishi-Gotanda, Shinagawa-ku, Tokyo 141 Japan.
ASIA-PACIFIC: Motorola Semiconductors H.K. Ltd.; Silicon Harbour Center, No. 2 Dai King Street, Tai Po Industrial Estate, Tai Po, N.T., Hong Kong.

SEMICONDUCTOR PRODUCT INFORMATION

For More Information On This Product,
Go to: www.freescale.com