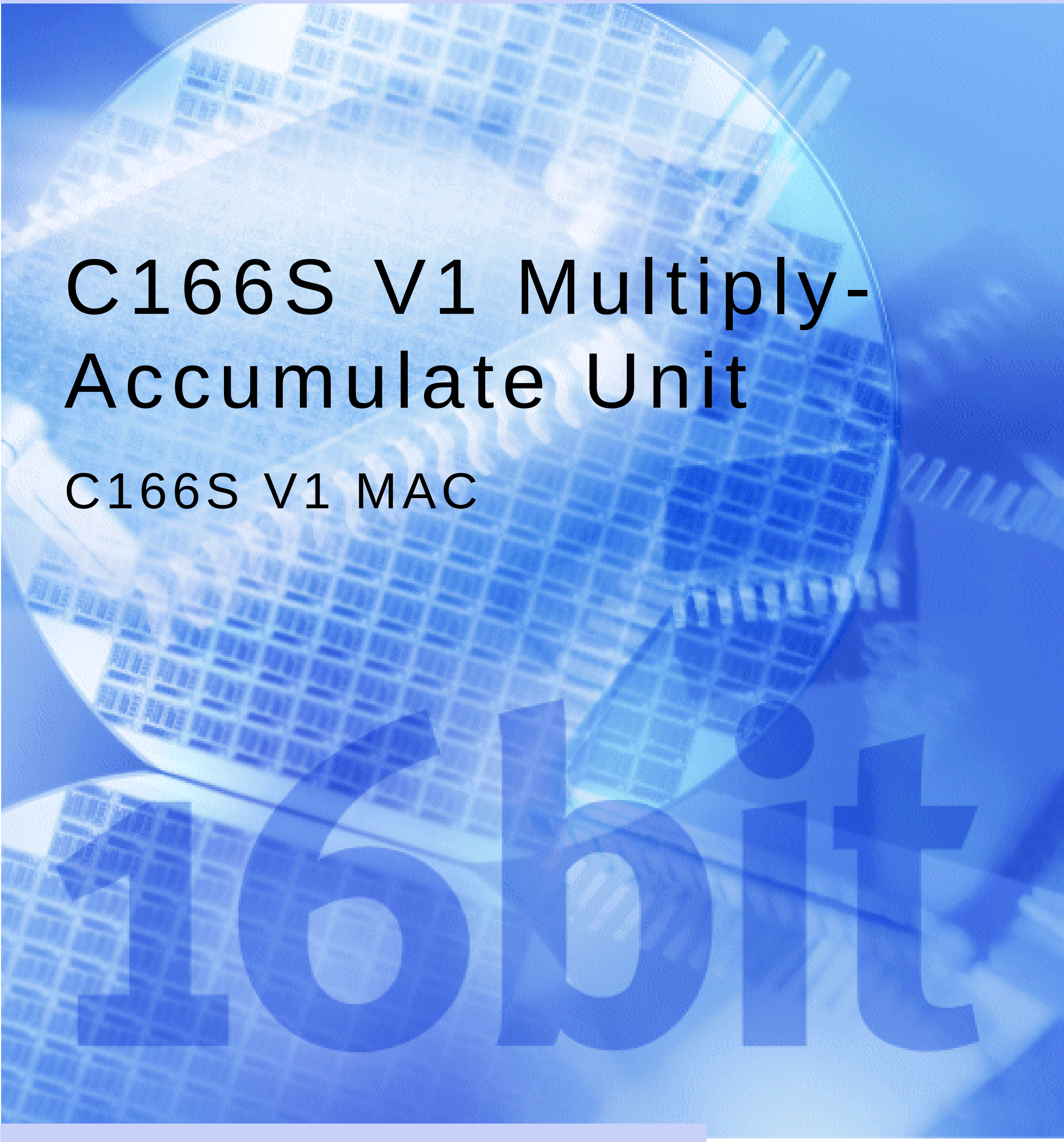
User's Manual, V 1.0, August 2001



C166S V1 Multiply-Accumulate Unit

C166S V1 MAC

Microcontrollers

Never stop thinking.

Edition 2001-08

Published by Infineon Technologies AG,
St.-Martin-Strasse 53,
D-81541 München, Germany

User's Manual, V 1.0, August 2001

C166S V1 Multiply-Accumulate Unit

C166S V1 MAC

Microcontrollers

Infineon technologies

Never stop thinking.

C166S V1 MAC

Revision History:	**2001-08**	V 1.0
Previous Version:	- -	

Page	**Subjects** (*major changes since last revision*)

Table of Contents

List of Tables

Page

1 MAC Unit Features

The Multiply-Accumulate (MAC) Unit is a specialized co-processor added to the C166S CPU core to improve the performance of signal processing algorithms. It includes:

- a multiply-accumulate unit
- an address generation unit capable of feeding the MAC Unit with 2 operands per cycle
- a repeat unit to execute a series of multiply-accumulate instructions

The architecture of the MAC Unit is outlined in **Figure 1**.

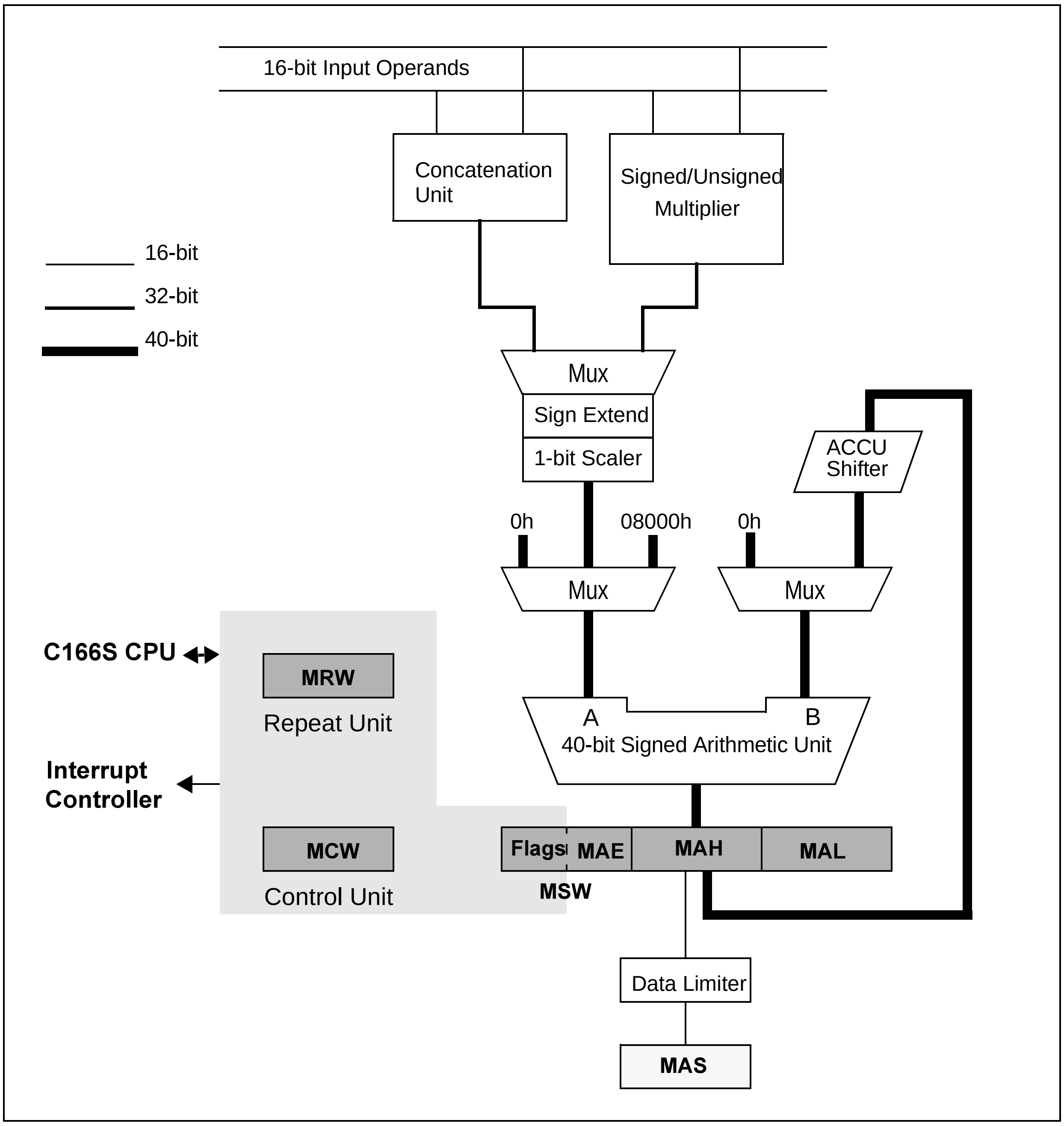


Figure 1 **MAC Unit Architecture**

The MAC Unit includes the following features.

1.1 Enhanced Addressing Capabilities

- New addressing modes, including a double indirect addressing mode with pointer post-modification.
- Parallel Data Move allows one operand move during multiply-accumulate instructions without penalty.
- New transfer instructions CoSTORE (for fast access to the MAC Unit SFRs) and CoMOV (for fast memory to memory table transfer).

1.2 Multiply-Accumulate Unit

- Execution of all MAC Unit operations within one CPU instruction cycle
- 16 x 16 signed/unsigned parallel multiplier
- 40-bit signed arithmetic unit with automatic saturation mode
- 40-bit signed accumulator
- 8-bit left/right shifter
- Scaler (one-bit left shifter)
- Data limiter
- Full instruction set with multiply and multiply-accumulate, 32-bit signed arithmetic, and compare instructions
- Three 16-bit status and control registers:
  – MSW - MAC Unit Status Word
  – MCW - MAC Unit Control Word
  – MRW - MAC Unit Repeat Word

1.3 Program Control

- Repeat Unit to allow some MAC Unit co-processor instructions to be repeated up to 8192 times. Repeated instructions may be interrupted.
- MAC Unit interrupt (implemented as Class B hardware trap) on MAC Unit condition flags.

2 MAC Unit Operation

MAC Unit operation is based on the cores instruction pipeline and extended addressing modes.

2.1 Instruction Pipelining

All MAC Unit instructions use a 4-stage pipeline. During each stage the following tasks are performed:

- **FETCH** All new instructions are double-word instructions.
- **DECODE** If required, operand addresses are calculated and the resulting operands are fetched. IDX and GPR pointers are post-modified if necessary.
- **EXECUTE** Performs the MAC Unit operation. At the cycle end the accumulator and the MAC Unit condition flags are updated if required. Modified GPR pointers are written-back during this stage, if required.
- **WRITEBACK** Operand write-back in the case of parallel data move.

Note: At least one instruction not using the MAC Unit must be inserted between two instructions that read from a MAC Unit register. This is because the accumulator and the status of the MAC Unit are modified during the Execute stage.
The CoSTORE instruction has been added to allow access to the MAC Unit registers immediately after a MAC Unit operation.

2.2 Address Generation

MAC Unit instructions can use some of the standard C166 addressing modes such as GPR direct or #data4 for immediate shift value. New addressing modes were added to supply the MAC Unit with two new operands per instruction cycle. These allow indirect addressing with **address pointer post-modification**.

Double indirect addressing requires two pointers. Any GPR can be used for one pointer, the other pointer is provided by one of two specific SFRs IDX0 and IDX1. Two pairs of offset registers QR0/QR1 and QX0/QX1 are associated with each pointer (GPR or IDX_i). The GPR pointer allows access to the entire memory space, but accesses via IDX_i are limited to the internal dual-port RAM, except for the CoMOV instruction. The various combinations of pointer post-modification for each of the two new addressing modes are shown in **Table 1**. Symbols **[$Rw_n\otimes$]** and **[$IDX_i\otimes$]** refer to these addressing modes.

Table 1 Pointer Post-modification Combinations for IDXi and Rw_n

Symbol	**Mnemonic**	**Address Pointer Operation**
[IDX_i⊗] stands for	[IDX_i]	($IDX_{i)}$ ← (IDX_i) (no-op)
	[IDX_i+]	(IDX_i) ← (IDX_i) +2 (i=0,1)
	[IDX_i -]	(IDX_i) ← (IDX_i) -2 (i=0,1)
	[$IDX_i + QX_j$]	(IDX_i) ← (IDX_i) + (QX_j) (i, j =0,1)
	[IDX_i - QX_j]	(IDX_i) ← (IDX_i) - (QX_j) (i, j =0,1)
[Rw_n⊗] stands for	[Rw_n]	(Rw_n) ← (Rw_n) (no-op)
	[Rw_n+]	(Rw_n) ← (Rw_n) +2 (n=0-15)
	[Rw_n-]	(Rw_n) ← (Rw_n) -2 (k=0-15)
	[Rw_n+QR_j]	(Rw_n) ← (Rw_n) + (QR_j) (n=0-15;j =0,1)
	[Rw_n - QR_j]	(Rw_n) ← (Rw_n) - (QR_j) (n=0-15; j =0,1)

The CoMACM class of instructions is a certain set of instructions that implement a mechanism called **Parallel Data Move**. The CoMACM instructions exclusively use double indirect addressing mode. Parallel Data Move allows the operand pointed to by IDX_i to be moved to a new location in parallel with the MAC Unit operation. The write-back address for the Parallel Data Move is calculated depending on the post-modification of IDX_i.

It is obtained by the operation "reverse" to the pointer post-modification of IDX_i, as explained in **Table 2**.

Table 2 Parallel Data Move Addressing

Instruction	**Writeback Address for Parallel Data Move**
CoMACM [IDX_i+],...	<IDX_i-2>
CoMACM [IDX_i-],...	<IDX_i+2>
CoMACM [IDX_i+QX_j],...	<IDX_i-QX_j>
CoMACM [IDX_i-QX_j],...	<IDX_i+QX_j>

The Parallel Data Move shifts a table of operands in parallel with a computation on those operands. Its specific use is for signal processing algorithms like filter computation.

An example of Parallel Data Move using the CoMACM instruction is shown in **Figure 2**.

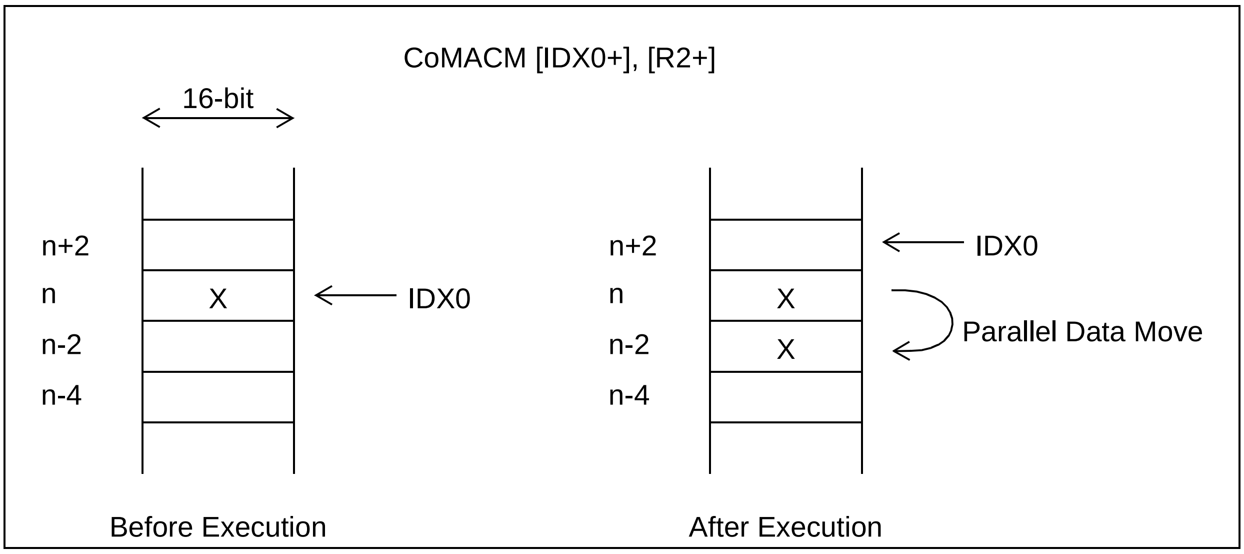


Figure 2 Example of Parallel Data Move

2.3 CoReg Addressing Mode

The MAC Unit accumulator and control registers (MAL, MAH, MSW, MCW, MRW) can be addressed by the regular instruction set as any other SFR. In addition, they can be addressed by the *CoSTORE* instruction. The CoSTORE instruction utilizes a specific 5-bit addressing mode called *CoReg* which allows the immediate storage of MAC Unit registers after an operation. Addresses of MAC Unit registers in CoReg addressing mode are shown in **Table 3**.

Table 3 CoReg 5-bit Addressing Mode

Register	Description	5-bit Address
MSW	MAC Unit Status Word	00000
MAH	MAC Unit Accumulator High Word	00001
MAS	"limited" MAC Unit Accumulator High Word	00010
MAL	MAC Unit Accumulator Low	00100
MCW	MAC Unit Control Word	00101
MRW	MAC Unit Repeat Word	00110

MAS is a **virtual** register. If MAS is specified as a source operand for CoSTORE, the MAH register is read through the data limiter. MAS cannot be addressed by regular SFR/ESFR addressing.

2.4 Number Representation and Rounding

The MAC Unit supports the **two's-complement** representation of binary numbers. In this format the sign bit is the MSB of the binary word. This is set to zero for positive numbers and set to one for negative numbers. Unsigned numbers are supported only by multiply/multiply-accumulate instructions which specify whether each operand is signed or unsigned.

In two's complement fractional format the N-bit operand is represented using the 1.[N-1] format (1 signed bit, N-1 fractional bits). This format can represent numbers between -1 and $+1-2^{-[N-1]}$ and is supported when the shift mode bit MP of register MCW is set.

The MAC Unit implements 'two's complement rounding', where one is added to the bit to the right of the rounding point (bit 15 of MAL) before truncation (MAL is cleared).

3 MAC Unit Components

The major components of the MAC Unit are shown in **Figure 1**. In the following, all of these components are described in detail.

3.1 16 x 16 Signed/Unsigned Parallel Multiplier

The multiplier executes 16 x 16-bit parallel signed/unsigned fractional and integer multiplications. The multiplier has two 16-bit input ports for the two operands and a 32-bit product output port. The result is always presented in a signed fractional or integer format.

3.2 Concatenation Unit

The concatenation unit enables the MAC Unit to perform 32-bit arithmetic operations in one CPU instruction cycle. It concatenates the two 16-bit operands to a 32-bit operand before the 32-bit operation is executed in the 40-bit arithmetic unit. The second required operand is always the current accumulator content. The concatenation unit is also used to pre-load the accumulator with a 32-bit value.

3.3 Sign Extension Unit and Scaler

Prior being fed to the 40-bit signed arithmetic unit, the result of the multiplier (or of the concatenation unit) is sign extended to a 40-bit number. This sign extension replicates the sign bit (MSB) of the word 8 times. With unsigned/unsigned instructions, (e.g. CoMULu, CoMACu) 8 zero bits are extended regardless of the MSB of the word to be extended.

The one-bit scaler can shift the sign extended result one bit to the left. Depending on the type of instruction, the scaler is controlled either by the Product Shift Mode Bit MP (bit 10 in MAC Unit Control Word - MCW) or by the instruction itself. For multiply instructions (if the MP bit is set), the product is automatically shifted left by one bit to compensate for the extra sign bit gained in multiplying two signed 2's complement numbers. The scaler is also active for instructions such as CoADD2, CoSUB2, etc., where the 32-bit operand is doubled before being fed to the arithmetic unit.

3.4 40-bit Signed Arithmetic Unit

The 40-bit signed arithmetic unit allows intermediate overflows in a series of multiply/ accumulate operations. There are two 40-bit input ports, A and B. The A-input port accepts data as 00’0000’0000h, 00’0000’8000h (round), or the sign extended and scaled result of the multiplier or of the concatenation unit.

The B-input port is the feedback of the accumulator output sent through the 8-bit left/right shifter. The B-input port can also receive 00’0000’0000h to allow direct transfer from the A-input port to the accumulator.

If, during accumulation, a 40-bit overflow of the accumulator occurs, the sticky overflow flag SV will be set in the MAC Unit Status Word (MSW).

The result of the addition/subtraction can be rounded or saturated on a 32-bit value automatically after every accumulation.

The rounding is performed by adding 00’0000’8000h to the result and clearing the Accumulator Low Word MAL. Automatic saturation is enabled by setting the saturation bit MS in the MAC Unit Control Word (MCW).

When the accumulator is in the saturation mode and a 32-bit overflow occurs, the accumulator is loaded with either the highest positive or the lowest negative value that can be represented with a 32-bit 2’s complement number, depending on the direction of the overflow. Thus the value of the accumulator upon saturation is 00’7fff’ffffh (positive) or ff’8000’0000h (negative). Automatic saturation sets the sticky limit flag (SL) in the MAC Unit Status Word (MSW).

Note: If automatic saturation and rounding is performed, MAL will be cleared on the saturated value and the result after saturating and rounding positive numbers will be 00’7fff 0000h.

Note: If the accumulator contains a value that can not be represented by a 32-bit 2-s complement number (i.e. MS was previously set to 0), then saturation can only be achieved by setting MS to 1 and one MAC Unit instruction executing. If this instruction causes a 40-bit overflow (or underflow), then the value of the accumulator upon saturation is 00’7fff’ffffh (or ff’8000’0000h).

3.5 40-bit Signed Accumulator Register

Most co-processor operations specify the 40-bit accumulator register as a source and/or destination operand. The accumulator is comprised of three SFR registers:

- MAL (MAC Unit Accumulator Low Word),
- MAH (MAC Unit Accumulator High Word) and
- MAE (ACCU Extension).

While MAH and MAL are 16-bit wide, MAE consist of only 8-bits, which are accessed as the least significant byte of the MAC Unit Status Word MSW. MAE is the most significant byte of the accumulator.

When writing to MAH by regular SFR addressing, the value in the accumulator is automatically adjusted to a sign extended 40-bit 2's complement format. MAL acquires zero value and MAE is automatically loaded with zeros in case of a positive number (MAH has 0 in the most significant bit) and with ones in case of a negative number (MAH has 1 in the most significant bit). Note that the values represented by the 32-bit number and the extended 40-bit number are the same and the MAE register does not contain significant bits. This is true whenever the highest 9 bits of the signed 40-bit result are identical.

During accumulator operations an overflow may occur and the result may not fit into 32-bits. The accumulator then exceeds the 32-bit boundary and changes the contents of MAE. Consequently there are significant (non-sign) bits in the top 8 bits of the accumulator. To indicate this extension, extension flag E, contained in the most significant byte of the MAC Unit Status Word MSW, is set to 1.

3.6 Data Limiter

Saturation arithmetic is also provided to selectively limit overflow when reading the accumulator by means of a ***CoSTORE <destination>, MAS*** instruction. If the contents of the accumulator cannot be represented by 32 bits without overflow, the limiter is enabled and MAS is modified to a limited value. Otherwise MAS is equal to MAH, as shown in **Table 4**.

Table 4 Data Limiter Output

E bit	N bit	Limiter Output (MAS)
0	x	equal to MAH
1	0	7fffh
1	1	8000h

Note: The MAS value is only readable by means of a ***CoSTORE <destination>, MAS*** *instruction. If executed, the accumulator and the status register are not affected.*

3.7 Accumulator Shifter

The accumulator shifter is a parallel shifter with 40-bit input and 40-bit output. The source operand of the shifter is an accumulator and possible shifting operations are:

- no shift (unmodified)
- up to 8-bit arithmetic left shift
- up to 8-bit arithmetic right shift

Note that left shift operations affect flags E, SV, SL, and C of the MAC Unit Status Word MSW. Therefore, if the automatic saturation mechanism is enabled (MS-bit set to 1), the behavior is similar to that of the 40-bit arithmetic unit.

Note: Certain precautions are required in cases of a left shift in conjunction with automatic saturation enabled (MS bit set to 1). If the MS bit is being set directly before the left shift instruction, correct saturation is not guaranteed under all circumstances since significant bits may have been shifted out before saturation is performed. To avoid this situation, switch on automatic saturation earlier so that the left shift instruction is already operating on a saturated value.

3.8 Repeat Unit

The MAC Unit includes a repeat unit which repeats some co-processor instructions up to 2^{13} (=8192) times. The repeat count is specified either by an immediate value (up to 31) or by the content of the repeat count (bits 12 to 0) in the MAC Unit Repeat Word (MRW). If the repeat count in MRW equals **N**, the instruction will be executed **N+1** times. At each iteration of a repeated instruction the repeat count is tested for zero. If zero, the instruction is terminated, otherwise the repeat count is decremented and the instruction is repeated. During such repeated sequences, the Repeat Flag MR (bit 15 of the MAC Unit Repeat Word (MRW)) is set until the last execution of the repeated instruction.

The syntax of repeated instructions is shown in the following examples:

```
Repeat #24 times
CoMAC[IDX0+],[R0+]          ; performed 24 times
```

In this example, the instruction is repeated according to a 5-bit immediate value. The repeat count in MRW is automatically loaded with this value minus one.

```
MOV MRW, #00FFh             ; load MRW
NOP                         ; instruction latency
Repeat MRW times
CoMACM [IDX1-],[R2+]        ; performed 256 times
```

The instruction is repeated according to the repeat count in MRW. Note that, due to the pipeline processing, at least one instruction should be inserted between MRW write and the next repeated instruction.

Repeat sequences may be interrupted. When an interrupt occurs during a repeat sequence, the sequence is stopped and the interrupt routine is executed. The repeat sequence resumes at the end of the interrupt routine. During the interrupt the repeat flag MR remains set, indicating that a repeated instruction has been interrupted and the repeat count holds the number of repetitions (minus 1) that remain to complete the sequence. If the repeat unit is used in the interrupt routine, MRW must be saved by the user and restored before the end of the interrupt routine.

Note: The MRW register must be used with caution. Except for restoring MRW after an interrupt, the MR bit should not be set by user. Otherwise correct instruction processing cannot be guaranteed.

4 MAC Unit Interrupt

The MAC Unit can generate an interrupt corresponding to the value of the status flags C (Carry), SV (Sticky Overflow), E (Extension) or SL (Sticky Limit) of the MSW register.

The MAC Unit Interrupt is globally enabled if the MIE flag in MCW is set. When enabled, the flags C, SV, E, or SL can trigger a MAC Unit Interrupt, provided that the corresponding mask flags CM, VM, EM, and LM in MCW are also set. The MIR flag in MSW is set upon the first interrupt condition. This flag must be cleared during interrupt processing. If this flag is set already, a new interrupt condition cannot trigger an interrupt.

The MAC Unit Interrupt is implemented as a Class B hardware trap (trap number A_H, trap priority I). The associated trap flag in the Trap Flag Register TFR is MACTRP (bit 6). Note that when a MAC Unit interrupt request occurs, this flag must be cleared by the user.

The layout of the Trap Flag Register TFR in the C166S V1.1 is as follows:

TFR
Trap Flag Register **SFR($FFAC_H$,$D6_H$)** **Reset value: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMI	STK OF	STK UF	0	0	0	0	0	UND OPC	MAC TRP	0	0	PRT FLT	ILL OPA	ILL INA	ILL BUS
			r	r	r	r	r	rwh	rwh	r	r				

Field	Bits	Type	Description
ILLBUS	[0]	rwh	ILLegal External BUS Access 0 No illegal external bus access detected 1 An external access has been attempted with no bus defined.
ILLINA[1)]	[1]	rwh	ILLegal INstruction Access 0 No illegal instruction access detected 1 A branch to an odd address has been attempt.
ILLOPA[1)]	[2]	rwh	ILLegal word OPerand Access 0 No illegal word operand access event detected 1 Ilegal word operand access event detected
PRTFLT[1)]	[3]	rwh	PRoTection FauLT 0 No protection fault event detected 1 Protection fault event detected

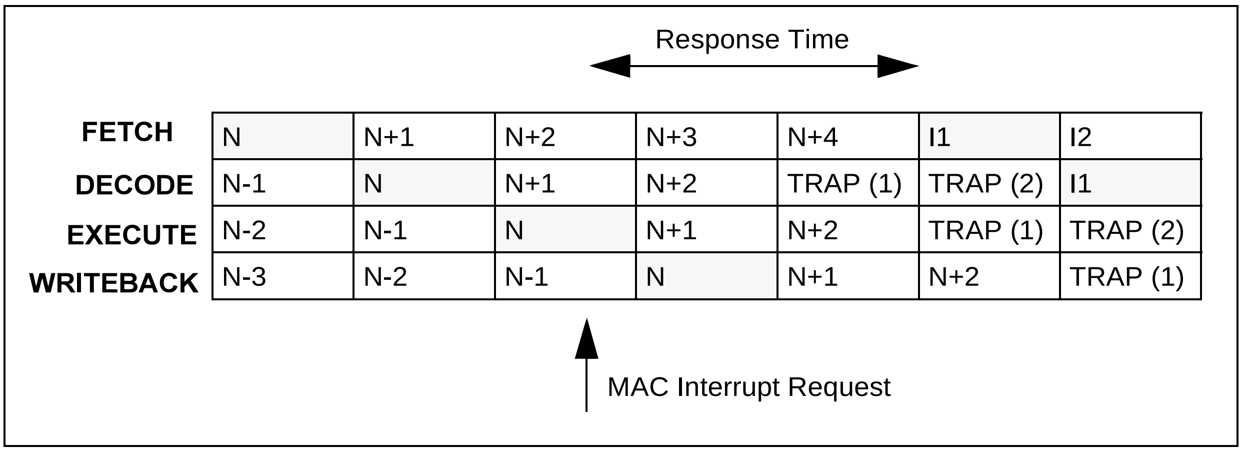
Field	Bits	Type	Description
MACTRP	[6]	rwh	MAC Unit Interrupt 0 No MAC Unit interrupt detected 1 MAC Unit interrupt detected
UNDOPC[1)]	[7]	rwh	UNDefined OPCode 0 No undefined opcode event detected 1 Undefined opcode event detected
STKUF[1)]	[13]	rwh	STacK UnderFlow flag 0 No stack underflow event detected 1 Stack underflow event detected
STKOF[1)]	[14]	rwh	STacK OverFlow flag 0 No stack overflow event detected 1 Stack overflow event detected
NMI[1)]	[15]	rwh	Non-Maskable Interrupt flag 0 No non-maskable interrupt detected 1 Non-maskable interrupt detected
0	[12, 11, 10, 9, 8, 5, 4]	r	Reserved read as '0'; writing to these bit positions has no effect.

1) This bit supports bit protection

Note: *The trap service routine must clear the respective trap flag. Otherwise, a new trap will be requested after exiting the service routine. Setting a trap request flag by software causes the same effects as if it had been set by hardware.*

As MAC Unit status flags are updated (or written by software) during the execute stage of the pipeline, the response time of a MAC Unit Interrupt Request is 3 instruction cycles, as illustrated in **Figure 3**. It is the number of instruction cycles required between the time the request is sent and the time the first instruction located at the interrupt vector location enters the pipeline.

Note: *The instruction pointer value stacked after a MAC Unit Interrupt does not point to the instruction that triggered the interrupt. Therefore, it is not possible to specify exactly the entry point of the interrupt. Furthermore, due to the interrupt response time, it may not be possible to determine the source of the interrupt request since the status of flags C, SV, E, or SL may have changed once the interrupt routine has started.*



FETCH	N	N+1	N+2	N+3	N+4	I1	I2
DECODE	N-1	N	N+1	N+2	TRAP (1)	TRAP (2)	I1
EXECUTE	N-2	N-1	N	N+1	N+2	TRAP (1)	TRAP (2)
WRITEBACK	N-3	N-2	N-1	N	N+1	N+2	TRAP (1)

Figure 3 Pipeline Diagram for MAC Unit Interrupt Response Time

5 MAC Unit Register Set

All MAC Unit registers are mapped into the SFR/ESFR memory space. Registers can be accessed using the regular instruction set and the co-processor instruction called CoSTORE. The following sections list the MAC Unit registers and their corresponding SFR/ESFR addresses.

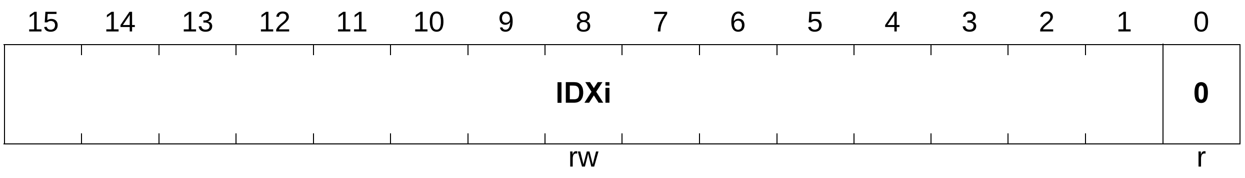
Note: With CPU Core SFRs, any write operation with the regular instruction set to a single byte of a MAC SFR, clears the non-addressed complementary byte within the specified SFR. Non-implemented SFR bits cannot be modified and always supply a read value of '0'.

5.1 MAC Unit Address Registers

The double indirect addressing modes require additional (E)SFRs: 2 address pointers IDX0/IDX1 and 4 offset registers QX0/QX1 and QR0/QR1.

The address pointer registers IDX0 and IDX1 are located in the SFR space.

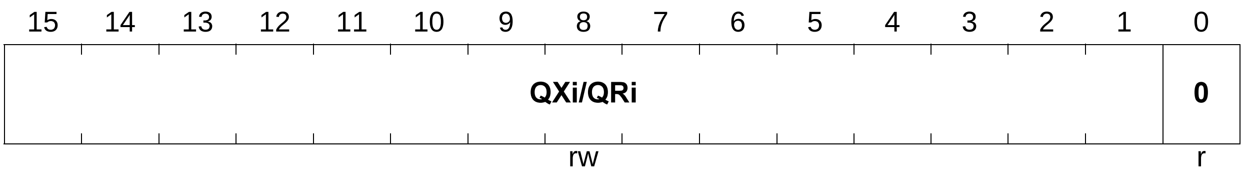
IDX0 (FF08h/84h)	**SFR**	**Reset Value: 0000h**
IDX1 (FF0Ah/85h)	**SFR**	**Reset Value: 0000h**



Field	Bits	Type	Description
IDXi	[15:1]	rw	**Modifiable portion of address pointer**
0	[0]	r	As IDXi may only contain even values, bit 0 is fixed to zero.

The offset registers QX0, QX1, QR0, and QR1 are located in the ESFR space.

QX0 (F000h/00h)	**ESFR**	**Reset Value: 0000h**
QX1 (F002h/01h)	**ESFR**	**Reset Value: 0000h**
QR0 (F004h/02h)	**ESFR**	**Reset Value: 0000h**
QR1 (F006h/03h)	**ESFR**	**Reset Value: 0000h**



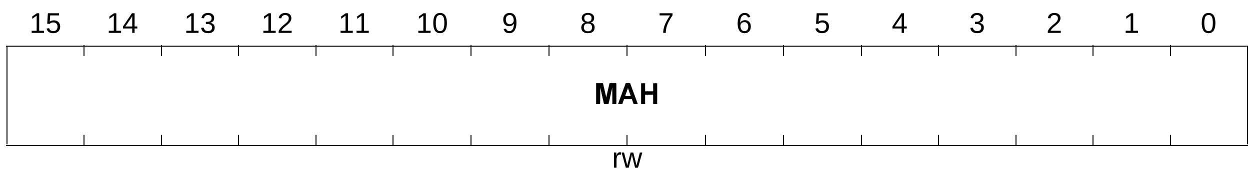
Field	Bits	Type	Description
QRi/QXi	[15:1]	rw	**Modifiable portion of offset registers** Specifies 16-bit address offset for IDXi pointers (QXi) or GPR pointers (QRi).
0	[0]	r	As MAC Unit instructions handle word operands only, bit 0 is fixed to '0'.

5.2 Accumulator Registers

The 40-bit accumulator consists of the registers MAL, MAH and the low byte of MSW, which is described in **Chapter 5.3**.

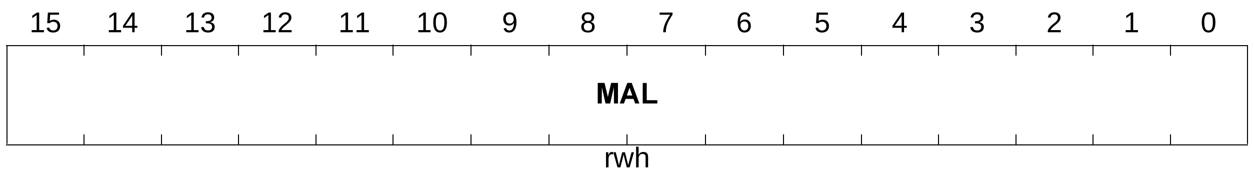
MAL and MAH are located in the non bit-addressable SFR space.

MAH (FE5Eh/2Fh) SFR Reset Value: 0000h



Field	Bits	Type	Description
MAH	[15:0]	rw	**MAC Unit Accumulator High Word** Contains bits 31 to 16 of the 40-bit MAC Unit Accumulator.

MAL (FE5Ch/2Eh) SFR Reset Value: 0000h



Field	Bits	Type	Description
MAL	[15:0]	rwh	**MAC Unit Accumulator Low Word** Contains bits 15 to 0 of the 40-bit MAC Unit Accumulator.

Note: MAL is automatically cleared when MAH is written by regular SFR addressing.

5.3 MAC Unit Status Word (MSW)

The bit-addressable register MSW reflects the current state of the MAC Unit. It is located in the SFR space and includes the 8-bit accumulator extension MAE and the 7 additional flags as shown below.

MSW (FFDEh/EFh) **SFR** **Reset Value: 0200h**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIR	**-**	**SL**	**E**	**SV**	**C**	**Z**	**N**	**MAE (ACCU Extension)**							
rwh	-	rwh	rwh	rwh	rwh	rwh	rwh	rwh							

Field	Bits	Type	Description
MAE	[7:0]	rwh	**ACCU Extension** The eight most significant bits of the 40-bit MAC Unit accumulator.
N	[8]	rwh	**Negative Result Flag** 0 MAC Unit result is negative 1 MAC Unit result is positive
Z	[9]	rwh	**Zero Flag** 0 MAC Unit result is not zero 1 MAC Unit result is zero
C	[10]	rwh	**Carry Flag** 0 No carry/borrow produced 1 Carry/borrow produced
SV	[11]	rwh	**Sticky Overflow Flag** 0 No 40-bit overflow occurred 1 40-bit overflow occurred
E	[12]	rwh	**Extension Flag** 0 MAE does not contain significant bits 1 MAE contains significant bits
SL	[13]	rwh	**Sticky Limit Flag** 0 No automatic 32-bit saturation occurred 1 Automatic 32-bit saturation occurred
MIR	[15]	rwh	**MAC Unit Interrupt Request Flag** 0 No MAC Unit interrupt is requested 1 MAC Unit interrupt is requested

ACCU Extension MAE

These 8 bits are part of the 40-bit accumulator register. The MAC Unit implicitly uses these bits during a MAC Unit operation. When writing to MAH by regular SFR addressing, MAE is automatically sign extended with the most significant bit of MAH and MAL is cleared.

Negative Result Flag N

The N flag is set if the most significant bit of the accumulator equals '1', otherwise it is cleared. With integer operations, the N Flag can be interpreted as the sign bit of the MAC Unit Accumulator (N=1 for negative, N=0 for positive). Negative numbers are always represented as the 2's complementation of the corresponding positive.

Zero Flag Z

The Z flag is set if the content of the MAC Unit Accumulator is equal to zero, otherwise it is cleared.

Carry Flag C

After a MAC Unit addition, the C flag indicates that a carry from the accumulator's most significant bit (bit 7 of MAE) has been generated. After a MAC Unit subtraction, the C flag indicates a "Borrow", which represents the logical negation of a "Carry" for the addition. During a subtraction, the C flag is set, if **no** carry from the most significant bit of the accumulator has been generated. Subtraction is performed by the MAC Unit as a 2's complement addition and the C flag is cleared when this complement addition caused a "Carry".

For left shift operations, the C flag represents the value of the bit shifted out last. Right shift operations always clear the C flag.

Sticky Overflow Flag SV

The SV flag indicates an arithmetic overflow. The SV flag is set if, during a MAC Unit operation, the accumulator exceeds the maximum range of 40-bit signed numbers. In the case of signed arithmetic, the SV flag is set if the carry into the sign bit differs from the carry out of the sign bit. With a left shift operation, the SV flag is set if the last bit shifted out is different from the new N flag.

If the SV flag is set then the result of the MAC Unit operation is invalid. Once set, other MAC Unit operations cannot affect the status of the SV flag. Only a direct write operation can clear it.

Extension Flag E

The E flag is set if the accumulator extension MAE contains significant bits, i.e., if the highest 9 bits of the accumulator are not identical.

Sticky Limit Flag SL

The SL flag is set if an automatic 32-bit saturation occurred. Automatic saturation is enabled by setting bit MS of the MAC Unit Control Word (MCW). The SL flag can also be set by a CoMIN or CoMAX operation. In such cases, the SL flag is set when the contents of the accumulator is greater than the operand of the CoMAX operation, or less than the operand of the CoMIN operation.

The SL flag is a sticky flag and, once set, is not affected by any other MAC Unit operation until cleared by a direct write operation.

MAC Unit Interrupt Request Flag MIR

The MIR flag indicates a MAC Unit Interrupt request. The interrupt mask bits of the MAC Unit Control Word (MCW) determine which flags of the MSW register can generate a MAC Unit Interrupt request. The MIR flag must be cleared during the interrupt routine.

Note: The MAC Unit status flags are modified (if needed) by executing the instruction. They are ***not affected by any instruction from the regular set*** *and thus their values may not be consistent with the accumulator content. For example, loading the accumulator with MOV instructions will not modify the status flags.*

5.4 MAC Unit Control Word (MCW)

The bit-addressable register (MCW) controls the operation of the MAC Unit and determines the functionality of the MAC Unit Interrupt. (MCW) is located in the SFR space.

MCW (FFDCh/EEh) **SFR** **Reset Value: 0000h**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MIE	**LM**	**EM**	**VM**	**CM**	**MP**	**MS**					-	-	-	-	-
rw	rw	rw	rw	rw	rw	rw									

Field	Bits	Type	Description
MS	[9]	rw	**Saturation Control** 0 Automatic saturation disabled 1 Automatic saturation enabled
MP	[10]	rw	**One-bit Scaler Control** 0 Multiplier product shift disabled 1 Multiplier product shift enabled
CM	[11]	rw	**Carry Mask** 0 C flag cannot generate interrupt request 1 C flag can generate interrupt request
VM	[12]	rw	**Overflow Mask** 0 SV flag cannot generate interrupt request 1 SV flag can generate interrupt request
EM	[13]	rw	**Extension Mask** 0 E flag cannot generate interrupt request 1 E flag can generate interrupt request
LM	[14]	rw	**Limit Mask** 0 SL flag cannot generate interrupt request 1 SL flag can generate interrupt request
MIE	[15]	rw	**MAC Unit Interrupt Enable** 0 MAC Unit interrupt globally disabled 1 MAC Unit interrupt globally enabled

Saturation Control Bit MS

If the MS bit is set, the accumulator is automatically saturated to 32 bits.

One-bit Scaler Control Bit MP

If the MP bit is set and both operands of a multiplication are signed, then the multiplier output is automatically shifted left by one bit. With multiply-accumulate operations the multiplier output is shifted before it is added to the accumulator.

Interrupt Mask Flags CM, VM, EM, LM

These bits are interrupt mask bits for the corresponding MAC Unit status bits. If a status flag and the corresponding mask bit are set, then the MIR flag of MSW will be set and a MAC Unit interrupt will be activated (provided that interrupts are enabled).

MAC Unit Interrupt Enable Bit MIE

The MIE bit globally enables or disables the MAC Unit interrupt. When set, the flags C, SV, E and SL from the MSW register can trigger an interrupt (provided that the corresponding mask flags CM, VM, EM, and LM of the (MCW) register are also set).

5.5 MAC Unit Repeat Word (MRW)

The MRW contains the number of repetitions an instruction is to be executed and is located in the SFR space.

MRW (FFDAh/EDh) **SFR** **Reset Value: 0000h**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MR	**-**	**-**	**Repeat Count**												
rwh	-		rwh												

Field	Bits	Type	Description
Repeat Count	[12:0]	rwh	**13-bit unsigned integer value** Indicates the number of times minus one a repeated instruction must be executed
MR	[15]	rwh	**Repeat Flag** Set when a repeated instruction is being executed

6 MAC Unit Instruction Set

The MAC Unit instruction set contains the following groups of instructions:

- multiply and multiply-accumulate instructions
- 32-bit arithmetic instructions
- shift instructions
- compare instructions
- transfer instructions

All MAC Unit instructions are 32-bit instructions with 4 bytes used to encode each instruction.

6.1 Syntax

Operands:

opX	specifies the immediate constant value of opX
(opX)	specifies the contents of opX
(opXn)	specifies the contents of bit n of opX
((opX))	specifies the contents of the contents of opX, i.e. opX is used as pointer to the actual operand

Operations:

(opX)← (opY)	(opY) **moved** into (opX)
+	**added** to
-	**subtracted** from
*	**multiplied** by
⇔	**compared** against
<<	logically **shifted** left
>>	logically **shifted** right
$>>_a$	arithmetically **shifted** right
(opX) || (opY)	(opX) (MSW) and (opY) (LSW) **concatenated**

Data Addressing Modes:

Rw_n or **Rw_m**	General Purpose (Word) Registers (GPRs), where "n" and "m" can be any value between 0 and 15.
[...]	indirect word memory location
Mxx	MAC Unit Register (MSW, MAH, MAL, MAS, MRW, MCW)
ACC:	MAC Unit Accumulator consisting of lowest byte of MSW, MAH and MAL.
#datax:	Immediate constant (the number of significant bits is represented by 'x').

Flag States:

-	unchanged
*	modified

Address Register Operations for double indirect addressing:

any GPR	first address pointer, allows to access the entire memory space
QR0/QR1	offset registers for first address pointer (GPR)
IDX0/IDX1	second address pointer, limited to internal dual-port RAM (except for CoMOV instruction)
QX0/QX1	offset registers for second address pointer

The symbols **[Rw_n⊗]** and **[IDX_i⊗]** refer to the various combinations of pointer post modifications, and are listed in **Table 1**.

Repeated Instruction Syntax:

Repeatable instructions, CoXXX, when repeated are expressed as follows:

```
repeat #data5 times CoXXX...
```

or

```
repeat MRW times CoXXX...
```

If MRW is invoked, the instruction is repeated (MRW[12:0]) + 1 times. Therefore, the maximum number an instruction can be repeated is 2^{13} = 8192.

The integer value #data5 specifies the number of times an instruction is repeated. Hence #data5 must be less than 32, and CoXXX can only be repeated up to 31 times.

Shift Value:

The shifter allows left/right shift operations up to 8-bit. A shift value larger than 8 invokes an 8 bit shift.

Instruction Encoding:

X	4-bit IDX addressing mode encoding
qqq	3-bit GPR offset encoding
rrrr:r	5-bit repeat field
wwww:w	5-bit CoReg address for CoSTORE instructions
ssss:	4-bit immediate shift value

6.2 List of MAC Unit Instructions

The MAC Unit instruction set is summarized in **Table 5**. Individual instructions are described in detail alphabetically.

Table 5 MAC Unit Instruction Set Summary.

Mnemonic	Addressing Modes	Rep	Mnemonic	Addressing Modes	Rep
CoMUL	Rw_n, Rw_m	No	CoMACM	[IDXi⊗], [Rw_m⊗]	Yes
CoMULu	[IDXi⊗], [Rw_m⊗]		CoMACMu		
CoMULus	Rw_n, [Rw_m⊗]		CoMACMus		
CoMULsu			CoMACMsu		
CoMUL-			CoMACM-		
CoMULu-			CoMACMu-		
CoMULus-			CoMACMus-		
CoMULsu-			CoMACMsu-		
CoMUL+rnd			CoMACM+rnd		
CoMULu+rnd			CoMACMu+rnd		
CoMULus+rnd			CoMACMus+rnd		
CoMULsu+rnd			CoMACMsu+rnd		
CoMAC	Rw_n, Rw_m	No	CoMACMR		
CoMACu	[IDXi⊗], [Rw_m⊗]	Yes	CoMACMRu		
CoMACus	Rw_n, [Rw_m⊗]		CoMACMRus		
CoMACsu			CoMACMRsu		
CoMAC-			CoMACMR+rnd		
CoMACu-			CoMACMRu+rnd		
CoMACus-			CoMACMRus+rnd		
CoMACsu-			CoMACMRsu+rnd		
CoMAC+rnd			CoADD	Rw_n, Rw_m	No
CoMACu+rnd			CoADD2	[IDXi⊗], [Rw_m⊗]	Yes
CoMACus+rnd			CoSUB	Rw_n, [Rw_m⊗]	
CoMACsu+rnd			CoSUB2		
CoMACR			CoSUBR		
CoMACRu			CoSUB2R		
CoMACRus			CoMAX		
CoMACRsu			CoMIN		
CoMACR+rnd			CoLOAD	Rw_n, Rw_m	No
CoMACRu+rnd			CoLOAD-	[IDXi⊗], [Rw_m⊗]	
CoMACRus+rnd			CoLOAD2	Rw_n, [Rw_m⊗]	
CoMACRsu+rnd			CoLOAD2-		
CoNOP	[Rw_n⊗]	Yes	CoCMP		
	[IDXi⊗]		CoSHL	#data4	No
	[IDXi⊗], [Rw_m⊗]		CoSHR	Rw_m	Yes
CoNEG		No	CoASHR	[Rw_m⊗]	
CoNEG+rnd			CoASHR+rnd		
CoRND			CoABS	Rw_n, Rw_m	No
CoSTORE	Rw_n, CoReg	No		[IDXi⊗], [Rw_m⊗]	
	[Rw_n⊗], CoReg	Yes		Rw_n, [Rw_m⊗]	
CoMOV	[IDXi⊗], [Rw_m⊗]	Yes			

CoABS Absolute Value CoABS

Group Arithmetic Instructions

Syntax **CoABS**

Source Operand(s) ACC $\rightarrow$ 40-bit signed value

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ Abs(ACC)

Description
Computes the absolute value of the 40-bit ACC contents.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	0	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if the ACC contents was 80 0000 0000H. Not affected otherwise.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Note: SV flag behavior has been changed to guarantee arithmetic correctness.

Encoding

Mnemonic	**Format**	**Repeat**
CoABS	A3 00 1A 00	no

CoABS Absolute Value CoABS

Group Arithmetic Instructions

Syntax **CoABS op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

$(ACC) \leftarrow Abs((op2) \| (op1))$

Description

Computes the absolute value of a 40-bit source operand and loads the result in the 40-bit ACC register. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW).

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	0	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoABS	Rw_n, Rw_m	A3 nm CA 00	no
CoABS	Rw_n, $[Rw_m \otimes]$	83 nm CA 0:0qqq	no
CoABS	$[IDXi \otimes]$, $[Rw_m \otimes]$	93 Xm CA 0:0qqq	no

CoADD Add CoADD

Group Arithmetic Instructions

Syntax **CoADD op1, op2**

Source Operand(s) op1, op2 → WORD

Destination Operand(s) ACC → 40-bit signed value

Operation

(tmp) ← (op2) || (op1)
(ACC) ← (ACC) + (tmp)

Description

Adds a 40-bit operand to the 40-bit ACC register contents and stores the result in the ACC register. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). This instruction is repeatable with indirect addressing modes and allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoADD	Rw_n, Rw_m	A3 nm 02 00	no
CoADD	Rw_n, [Rw_m⊗]	83 nm 02 rrrr:rqqq	yes
CoADD	[IDXi⊗], [Rw_m⊗]	93 Xm 02 rrrr:rqqq	yes

CoADD2 Add CoADD2

Group Arithmetic Instructions

Syntax **CoADD2 op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
(tmp) ← 2 * ((op2) || (op1))
(ACC) ← (ACC) + (tmp)
```

Description
Adds a 40-bit operand to the 40-bit ACC register contents and stores the result in the ACC register. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). The 40-bit operand is then multiplied by two before being added to ACC register. This instruction is repeatable with indirect addressing modes and allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoADD2	Rw_n, Rw_m	A3 nm 42 00	no
CoADD2	Rw_n, $[Rw_m\otimes]$	83 nm 42 rrrr:rqqq	yes
CoADD2	$[IDXi\otimes]$, $[Rw_m\otimes]$	93 Xm 42 rrrr:rqqq	yes

CoASHR Accumulator Arithmetic Shift Right with Round CoASHR

Group Shift Instructions

Syntax **CoASHR op1, rnd**

Source Operand(s) op1 → shift counter

Destination Operand(s) ACC → 40-bit signed value

Operation

```
(count) ← (op1)
(C) ← 0
DO WHILE (count) ≠ 0
        (ACC[n]) ← (ACC[n+1]) [n=0-38]
        (count) ← (count) -1
END WHILE
(ACC) ← (ACC) + 0000 8000h
(MAL) ← 0
```

Description

Arithmetically shifts the ACC register right by the number of bits as specified by operand op1. Then the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. To preserve the sign of the ACC register, the most significant bits of the result are filled with sign 0 if the original most significant bit was a 0 or with sign 1 if the original most significant bit was 1. Only shift values from 0 to 8 (inclusive) are allowed. op1 can be either a 4-bit unsigned immediate data or the 4 least significant bits (considered as unsigned data) of a directly or indirectly addressed operand.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated when rounding. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoASHR	#data4, rnd	A3 00 B2 0sss:s000	no
CoASHR	Rw_n, rnd	A3 nn BA rrrr:r000	yes
CoASHR	[Rw_m⊗], rnd	83 mm BA rrrr:rqqq	yes

CoASHR Accumulator Arithmetic Shift Right CoASHR

Group Shift Instructions

Syntax **CoASHR op1**

Source Operand(s) op1 $\rightarrow$ shift counter

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
(count) ← (op1)
(C) ← 0
DO WHILE (count) ≠ 0
        (ACC[n]) ← (ACC[n+1]) [n=0-38]
        (count) ← (count) -1
END WHILE
```

Description

Arithmetically shifts the ACC register right by the number of bits as specified by operand op1. To preserve the sign of the ACC register, the most significant bits of the result are filled with sign 0 if the original most significant bit was a 0 or with sign 1 if the original most significant bit was 1. Only shift values from 0 to 8 (inclusive) are allowed. op1 can be either a 4-bit unsigned immediate data or the 4 least significant bits (considered as unsigned data) of a directly or indirectly addressed operand. The MS bit of the MCW register does not affect the result.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	*	-	0	*	*	no

SL	Not affected.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoASHR	#data4	A3 00 A2 0sss:s000	no
CoASHR	Rw_n	A3 nn AA rrrr:r000	yes
CoASHR	$[Rw_m\otimes]$	83 mm AA rrrr:rqqq	yes

CoCMP Compare CoCMP

Group Compare Instructions

Syntax **CoCMP op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) none

Operation

$tmp \leftarrow (op2) \,||\, (op1)$
$(ACC) \Leftrightarrow (tmp)$

Description

Subtracts a 40-bit signed operand from the 40-bit ACC contents and updates the N, Z and C flags of the MSW register leaving the ACC register unchanged. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). The MS bit of the MCW register does not affect the result. This instruction allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	-	-	*	*	*	no

SL	Not affected.
E	Not affected.
SV	Not affected.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoCMP	Rw_n, Rw_m	A3 nm C2 00	no
CoCMP	Rw_n, [Rw_m⊗]	83 nm C2 0:0qqq	no
CoCMP	[IDXi⊗], [Rw_m⊗]	93 Xm C2 0:0qqq	no

CoLOAD — Load Accumulator — CoLOAD

Group Arithmetic Instructions

Syntax **CoLOAD op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
(tmp) ← (op2) || (op1)
(ACC) ← 0 + (tmp)
```

Description
Loads the 40-bit ACC register with a 40-bit source operand. The 40-bit source operand is the sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). This instruction allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	0	-	0	*	*	no

SL	Not affected.
E	Always cleared.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoLOAD	Rw_n, Rw_m	A3 nm 22 00	no
CoLOAD	Rw_n, [Rw_m⊗]	83 nm 22 0:0qqq	no
CoLOAD	[IDXi⊗], [Rw_m⊗]	93 Xm 22 0:0qqq	no

CoLOAD- Load Accumulator CoLOAD-

Group Arithmetic Instructions

Syntax **CoLOAD- op1, op2**

Source Operand(s) op1, op2 → WORD

Destination Operand(s) ACC → 40-bit signed value

Operation

```
(tmp) ← (op2) || (op1)
(ACC) ← 0 - (tmp)
```

Description

Loads the 40-bit ACC register with a 40-bit source operand. The 40-bit source operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). The 40-bit source operand is 2's complemented, before being stored in the ACC register. This instruction allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoLOAD-	Rw_n, Rw_m	A3 nm 2A 00	no
CoLOAD-	Rw_n, [Rw_m⊗]	83 nm 2A 0:0qqq	no
CoLOAD-	[IDXi⊗], [Rw_m⊗]	93 Xm 2A 0:0qqq	no

CoLOAD2 Load Accumulator CoLOAD2

Group Arithmetic Instructions

Syntax **CoLOAD2 op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
        (tmp) ← 2 * ((op2) || (op1))
        (ACC) ← 0 + (tmp)
```

Description

Loads the 40-bit ACC register with a 40-bit source operand. The 40-bit source operand is a sign-extended results of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). The 40-bit operand is also multiplied by two, before being stored in the ACC register. This instruction allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	0	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoLOAD2	Rw_n, Rw_m	A3 nm 62 00	no
CoLOAD2	Rw_n, [Rw_m⊗]	83 nm 62 0:0qqq	no
CoLOAD2	[IDXi⊗], [Rw_m⊗]	93 Xm 62 0:0qqq	no

CoLOAD2- Load Accumulator CoLOAD2-

Group Arithmetic Instructions

Syntax **CoLOAD2- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ 2 * ((op2) || (op1))
(ACC) $\leftarrow$ 0 - (tmp)

Description
Loads the 40-bit ACC register with a 40-bit source operand. The 40-bit source operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). The 40-bit operand is also multiplied by two and negated, before being stored in the ACC register. This instruction allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoLOAD2-	Rw_n, Rw_m	A3 nm 6A 00	no
CoLOAD2-	Rw_n, [Rw_m⊗]	83 nm 6A 0:0qqq	no
CoLOAD2-	[IDXi⊗], [Rw_m⊗]	93 Xm 6A 0:0qqq	no

CoMAC Multiply-Accumulate with Round CoMAC

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMAC op1, op2, rnd**

Source Operand(s) op1, op2 → WORD

Destination Operand(s) ACC → 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (tmp) ← ((op1) * (op2)) <<1
        (ACC) ← (ACC) + (tmp) + 00 0000 8000h
ELSE
        (tmp) ← (op1) * (op2)
        (ACC) ← (ACC) + (tmp) + 00 0000 8000h
END IF
(MAL) ← 0
```

Description

Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted, then it is added to the 40-bit ACC register contents. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared. This instruction allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMAC	Rw_n, Rw_m, rnd	A3 nm D1 00	no
CoMAC	Rw_n, [Rw_m⊗], rnd	83 nm D1 rrrr:rqqq	yes
CoMAC	[IDXi⊗], [Rw_m⊗], rnd	93 Xm D1 rrrr:rqqq	yes

CoMAC Multiply-Accumulate CoMAC

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMAC op1, op2**

Source Operand(s)	op1, op2 $\rightarrow$ WORD
Destination Operand(s)	ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (tmp) ← ((op1) * (op2)) <<1
        (ACC) ← (ACC) + (tmp)
ELSE
        (tmp) ← (op1) * (op2)
        (ACC) ← (ACC) + (tmp)
END IF
```

Description
Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted, then it is added to the 40-bit ACC register contents before being stored in the 40-bit ACC register. This instruction allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMAC	Rw_n, Rw_m	A3 nm D0 00	no
CoMAC	Rw_n, [Rw_m⊗]	83 nm D0 rrrr:rqqq	yes
CoMAC	[IDXi⊗], [Rw_m⊗]	93 Xm D0 rrrr:rqqq	yes

CoMAC- Multiply-Accumulate CoMAC-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMAC- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (tmp) ← ((op1) * (op2)) <<1
        (ACC) ← (ACC) - (tmp)
ELSE
        (tmp) ← (op1) * (op2)
        (ACC) ← (ACC) - (tmp)
END IF
```

Description
Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted, then it is subtracted from the 40-bit ACC register contents before being stored in the 40-bit ACC register. This instruction allows up to two parallel memory reads.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMAC-	Rw, Rw_m	A3 nm E0 00	no
CoMAC-	Rw_n, $[Rw_m\otimes]$	83 nm E0 rrrr:rqqq	yes
CoMAC-	$[IDXi\otimes]$, $[Rw_m\otimes]$	93 Xm E0 rrrr:rqqq	yes

CoMACM Multiply-Accumulate & Move & Round CoMACM

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACM op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (tmp) ← (((op1)) * ((op2))) <<1
        (ACC) ← (ACC) + (tmp) + 00 0000 8000h
ELSE
        (tmp) ← ((op1))*((op2))
        (ACC) ← (ACC) + (tmp) + 00 0000 8000h
END IF
(MAL) ← 0
((IDXi(-⊗))) ← ((IDXi))
```

Description
Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then, if the MP flag is set, it is one-bit left shifted, and next, it is added to the 40-bit ACC register contents. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMACM	[IDXi⊗], [Rw$_m$⊗], rnd	93 Xm D9 rrrr:rqqq	yes

CoMACM Multiply-Accumulate & Move CoMACM

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACM op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
        $(tmp) \leftarrow (((op1)) * ((op2))) <<1$
        $(ACC) \leftarrow (ACC) + (tmp)$
ELSE
        $(tmp) \leftarrow ((op1)) * ((op2))$
        $(ACC) \leftarrow (ACC) + (tmp)$
END IF
$((IDXi(-\otimes))) \leftarrow ((IDXi))$
```

Description
Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted, and next it is added to the 40-bit ACC register contents before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACM	[IDXi⊗], [Rw_m⊗]	93 Xm D8 rrrr:rqqq	yes

CoMACM- Multiply-Accumulate & Move CoMACM-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACM- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
          (tmp) ← (((op1)) * ((op2))) <<1
          (ACC) ← (ACC) - (tmp)
ELSE
          (tmp) ← ((op1)) * ((op2))
          (ACC) ← (ACC) - (tmp)
END IF
((IDXi(-⊗))) ← ((IDXi))
```

Description

Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted, and next it is subtracted from the 40-bit ACC register contents before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMACM-	[IDXi⊗], [Rw$_m$⊗]	93 Xm E8 rrrr:rqqq	yes

CoMACMR Multiply-Accumulate & Move & Round CoMACMR

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMR op1, op2, rnd**

Source Operand(s) op1, op2 → WORD

Destination Operand(s) ACC → 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (tmp) ← (((op1)) * ((op2))) <<1
        (ACC) ← (tmp) - (ACC) + 00 0000 8000h
ELSE
        (tmp) ← ((op1))*((op2))
        (ACC) ← (tmp) - (ACC) + 00 0000 8000h
END IF
(MAL) ← 0
((IDXi(-⊗))) ← ((IDXi))
```

Description

Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted, and next the 40-bit ACC register contents is subtracted from the result. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMACMR	[IDXi⊗], [Rw$_m$⊗], rnd	93 Xm F9 rrrr:rqqq	yes

CoMACMR Multiply-Accumulate & Move CoMACMR

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMR op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (tmp) ← (((op1)) * ((op2))) <<1
        (ACC) ← (tmp) - (ACC)
ELSE
        (tmp) ← ((op1)) * ((op2))
        (ACC) ← (tmp) - (ACC)
END IF
((IDXi(-⊗))) ← ((IDXi))
```

Description

Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted, and next the 40-bit ACC register contents is subtracted from the result before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMR	[IDXi⊗], [Rw_m⊗]	93 Xm F8 rrrr:rqqq	yes

CoMACMRsu Multiply-Accumulate & Move & Round CoMACMRsu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMRsu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1)) * ((op2))
(ACC) $\leftarrow$ (tmp) - (ACC) + 00 0000 8000h
(MAL) $\leftarrow$ 0
((IDXi(-⊗))) $\leftarrow$ ((IDXi))

Description
Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then the 40-bit ACC register contents is subtracted from the result before being stored in the 40-bit ACC register. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic	Format	Repeat
CoMACMRsu [IDXi⊗], [Rw_m⊗], rnd	93 Xm 79 rrrr:rqqq	yes

CoMACMRsu Multiply-Accumulate & Move CoMACMRsu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMRsu op1, op2**

Source Operand(s) op1, op2 → WORD

Destination Operand(s) ACC → 40-bit signed value

Operation

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC)
((IDXi(-⊗))) ← ((IDXi))

Description

Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then the 40-bit ACC register contents is subtracted from the result before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic	**Format**	**Repeat**
CoMACMRsu [IDXi⊗], [Rw_m⊗]	93 Xm 78 rrrr:rqqq	yes

CoMACMRu Multiply-Accumulate & Move & Round CoMACMRu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMRu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
(tmp) ← ((op1))*((op2))
(ACC) ← (tmp) - (ACC) + 00 0000 8000h
(MAL) ← 0
((IDXi(-⊗))) ← ((IDXi))
```

Description

Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended, then the 40-bit ACC register contents is subtracted from the result. Finally, the obtained result is 2’s complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMRu	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 39 rrrr:rqqq	yes

CoMACMRu Multiply-Accumulate & Move CoMACMRu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMRu op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1))*((op2))
(ACC) $\leftarrow$ (tmp) - (ACC)
((IDXi(-⊗))) $\leftarrow$ ((IDXi))

Description
Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended, then the 40-bit ACC register contents is subtracted from the result before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMRu	[IDXi⊗], [Rw_m⊗]	93 Xm 38 rrrr:rqqq	yes

CoMACMRus Multiply-Accumulate & Move & Round CoMACMRus

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMRus op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) ← ((op1)) * ((op2))
(ACC) ← (tmp) - (ACC) + 00 0000 8000h
(MAL) ← 0
((IDXi(-⊗))) ← ((IDXi))

Description

Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then the 40-bit ACC register contents is subtracted from the result. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic	**Format**	**Repeat**
CoMACMRus [IDXi⊗], [Rw_m⊗], rnd	93 Xm B9 rrrr:rqqq	yes

CoMACMRus Multiply-Accumulate & Move CoMACMRus

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMRus op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1)) * ((op2))
(ACC) $\leftarrow$ (tmp) - (ACC)
((IDXi(-⊗))) $\leftarrow$ ((IDXi))

Description

Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then the 40-bit ACC register contents is subtracted from the result before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic	Format	Repeat
CoMACMRus [IDXi⊗], [Rw_m⊗]	93 Xm B8 rrrr:rqqq	yes

CoMACMsu Multiply-Accumulate & Move & Round CoMACMsu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMsu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1)) * ((op2))
(ACC) $\leftarrow$ (ACC) + (tmp) + 00 0000 8000h
(MAL) $\leftarrow$ 0
((IDXi(-⊗))) $\leftarrow$ ((IDXi))

Description

Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then it is added to the 40-bit ACC register contents. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMsu	[IDXi⊗], [Rw$_m$⊗], rnd	93 Xm 59 rrrr:rqqq	yes

CoMACMsu — Multiply-Accumulate & Move — CoMACMsu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMsu op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1)) * ((op2))
(ACC) $\leftarrow$ (ACC) + (tmp)
((IDXi(-⊗))) $\leftarrow$ ((IDXi))

Description
Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then it is added to the 40-bit ACC register contents before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMsu	[IDXi⊗], [Rw_m⊗]	93 Xm 58 rrrr:rqqq	yes

CoMACMsu- Multiply-Accumulate & Move CoMACMsu-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMsu- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1)) * ((op2))
(ACC) $\leftarrow$ (ACC) - (tmp)
((IDXi(-⊗))) $\leftarrow$ ((IDXi))

Description

Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then it is subtracted from the 40-bit ACC register contents before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMsu-	[IDXi⊗], [Rw_m⊗]	93 Xm 68 rrrr:rqqq	yes

CoMACMu Multiply-Accumulate & Move & Round CoMACMu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1)) * ((op2))
(ACC) $\leftarrow$ (ACC) + (tmp) + 00 0000 8000h
(MAL) $\leftarrow$ 0
((IDXi(-⊗))) $\leftarrow$ ((IDXi))

Description
Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended, then it is added to the 40-bit ACC register contents. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMu	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 19 rrrr:rqqq	yes

CoMACMu Multiply-Accumulate & Move CoMACMu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMu op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

$(tmp) \leftarrow ((op1)) * ((op2))$
$(ACC) \leftarrow (ACC) + (tmp)$
$((IDXi(-\otimes))) \leftarrow ((IDXi))$

Description

Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended, then it is added to the 40-bit ACC register contents before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMu	[IDXi$\otimes$], [Rw$_m$$\otimes$]	93 Xm 18 rrrr:rqqq	yes

CoMACMu- Multiply-Accumulate & Move CoMACMu-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMu- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1)) * ((op2))
(ACC) $\leftarrow$ (ACC) - (tmp)
((IDXi(-⊗))) $\leftarrow$ ((IDXi))

Description

Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended, then it is subtracted from the 40-bit ACC register contents before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMu-	[IDXi⊗], [Rw_m⊗]	93 Xm 28 rrrr:rqqq	yes

CoMACMus Multiply-Accumulate & Move & Round CoMACMus

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMus op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1)) * ((op2))
(ACC) $\leftarrow$ (ACC) + (tmp) + 00 0000 8000h
(MAL) $\leftarrow$ 0
((IDXi(-⊗))) $\leftarrow$ ((IDXi))

Description

Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then it is added to the 40-bit ACC register contents. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMus	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 99 rrrr:rqqq	yes

CoMACMus   Multiply-Accumulate & Move   CoMACMus

Group   Multiply/Multiply-Accumulate Instructions

Syntax   **CoMACMus op1, op2**

Source Operand(s)   op1, op2 $\rightarrow$ WORD

Destination Operand(s)   ACC $\rightarrow$ 40-bit signed value

Operation

$(tmp) \leftarrow ((op1)) * ((op2))$
$(ACC) \leftarrow (ACC) + (tmp)$
$((IDXi(-\otimes))) \leftarrow ((IDXi))$

Description
Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then it is added to the 40-bit ACC register contents before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMus	[IDXi⊗], [Rw_m⊗]	93 Xm 98 rrrr:rqqq	yes

CoMACMus- Multiply-Accumulate & Move CoMACMus-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACMus- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ ((op1)) * ((op2))
(ACC) $\leftarrow$ (ACC) - (tmp)
((IDXi(-$\otimes$))) $\leftarrow$ ((IDXi))

Description

Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended, then it is subtracted from the 40-bit ACC register contents before being stored in the 40-bit ACC register. In parallel to the arithmetic operation and to the two parallel reads, the data pointed to by IDXi overwrites another data located in memory (DPRAM). The address of the overwritten data depends on the operation executed on IDXi.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACMus-	[IDXi$\otimes$], [Rw$_m$$\otimes$]	93 Xm A8 rrrr:rqqq	yes

CoMACR Multiply-Accumulate & Round CoMACR

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACR op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (tmp) ← ((op1) * (op2)) <<1
        (ACC) ← (tmp) - (ACC) + 00 0000 8000h
ELSE
        (tmp) ← (op1) * (op2)
        (ACC) ← (tmp) - (ACC) + 00 0000 8000h
END IF
(MAL) ← 0
```

Description

Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then, if the MP flag is set, it is one-bit left shifted, then the 40-bit ACC register contents is subtracted from the result. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMACR	Rw_n, Rw_m, rnd	A3 nm F1 00	no
CoMACR	Rw_n, [Rw_m⊗], rnd	83 nm F1 rrrr:rqqq	yes
CoMACR	[IDXi⊗], [Rw_m⊗], rnd	93 Xm F1 rrrr:rqqq	yes

CoMACR Multiply-Accumulate CoMACR

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACR op1, op2**

Source Operand(s)	op1, op2 $\rightarrow$ WORD
Destination Operand(s)	ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (tmp) ← ((op1) * (op2)) <<1
        (ACC) ← (tmp) - (ACC)
ELSE
        (tmp) ← (op1) * (op2)
        (ACC) ← (tmp) - (ACC)
END IF
```

Description

Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted, then the 40-bit ACC register contents is subtracted from the result before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMACR	Rw_n, Rw_m	A3 nm F0 00	no
CoMACR	Rw_n, [Rw_m⊗]	83 nm F0 rrrr:rqqq	yes
CoMACR	[IDXi⊗], [Rw_m⊗]	93 Xm F0 rrrr:rqqq	yes

CoMACRsu Mixed Multiply-Accumulate & Round CoMACRsu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACRsu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
(tmp) ← (op1) * (op2)
(ACC) ← (tmp) - (ACC) + 00 0000 8000h
(MAL) ← 0
```

Description
Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then the 40-bit ACC register contents is subtracted from the result. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACRsu	Rw_n, Rw_m, rnd	A3 nm 71 00	no
CoMACRsu	Rw_n, [Rw_m⊗], rnd	83 nm 71 rrrr:rqqq	yes
CoMACRsu	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 71 rrrr:rqqq	yes

CoMACRsu — Mixed Multiply-Accumulate — CoMACRsu

Group — Multiply/Multiply-Accumulate Instructions

Syntax **CoMACRsu op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (tmp) - (ACC)

Description

Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then the 40-bit ACC register contents is subtracted from the result before being stored in the 40-bit ACC register.

MAC Flags

N	SL	E	SV	C	Z	N
*	*	*	*	*	*	*

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACRsu	Rw_n, Rw_m	A3 nm 70 00	no
CoMACRsu	Rw_n, [Rw_m⊗]	83 nm 70 rrrr:rqqq	yes
CoMACRsu	[IDXi⊗], [Rw_m⊗]	93 nm 70 rrrr:rqqq	yes

CoMACRu Unsigned Multiply-Accumulate & Round CoMACRu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACRu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (tmp) - (ACC) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description
Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended and then the 40-bit ACC register contents is subtracted from the result. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACRu	Rw_n, Rw_m, rnd	A3 nm 31 00	no
CoMACRu	Rw_n, [Rw_m⊗], rnd	83 nm 31 rrrr:rqqq	yes
CoMACRu	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 31 rrrr:rqqq	yes

CoMACRu Unsigned Multiply-Accumulate CoMACRu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACRu op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (tmp) - (ACC)

Description

Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended and then the 40-bit ACC register contents is subtracted from the result before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACRu	Rw_n, Rw_m	A3 nm 30 00	no
CoMACRu	Rw_n, [Rw_m⊗]	83 nm 30 rrrr:rqqq	yes
CoMACRu	[IDXi⊗], [Rw_m⊗]	93 Xm 30 rrrr:rqqq	yes

CoMACRus Mixed Multiply-Accumulate & Round CoMACRus

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACRus op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (tmp) - (ACC) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description
Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then the 40-bit ACC register contents is subtracted from the result. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACRus	Rw_n, Rw_m, rnd	A3 nm B1 00	no
CoMACRus	Rw_n, [Rw_m⊗], rnd	83 nm B1 rrrr:rqqq	yes
CoMACRus	[IDXi⊗], [Rw_m⊗], rnd	93 Xm B1 rrrr:rqqq	yes

CoMACRus Mixed Multiply-Accumulate CoMACRus

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACRus op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (tmp) - (ACC)

Description
Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then the 40-bit ACC register contents is subtracted from the result before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACRus	Rw_n, Rw_m	A3 nm B0 00	no
CoMACRus	Rw_n, [Rw_m⊗]	83 nm B0 rrrr:rqqq	yes
CoMACRus	[IDXi⊗], [Rw_m⊗]	93 Xm B0 rrrr:rqqq	yes

CoMACsu Mixed Multiply-Accumulate & Round CoMACsu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACsu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (ACC) + (tmp) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description

Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then added to the 40-bit ACC register contents. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACsu	Rw_n, Rw_m, rnd	A3 nm 51 00	no
CoMACsu	Rw_n, [Rw_m⊗], rnd	83 nm 51 rrrr:rqqq	yes
CoMACsu	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 51 rrrr:rqqq	yes

CoMACsu Mixed Multiply-Accumulate CoMACsu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACsu op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (ACC) + (tmp)

Description

Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then added to the 40-bit ACC register contents before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACsu	Rw_n, Rw_m	A3 nm 50 00	no
CoMACsu	Rw_n, [Rw_m⊗]	83 nm 50 rrrr:rqqq	yes
CoMACsu	[IDXi⊗], [Rw_m⊗]	93 Xm 50 rrrr:rqqq	yes

CoMACsu- Mixed Multiply-Accumulate CoMACsu-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACsu- op1, op2**

Source Operand(s)	op1, op2 $\rightarrow$ WORD
Destination Operand(s)	ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (ACC) - (tmp)

Description
Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then subtracted from the 40-bit ACC register contents before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACsu-	Rw_n, Rw_m	A3 nm 60 00	no
CoMACsu-	Rw_n, [Rw_m⊗]	83 nm 60 rrrr:rqqq	yes
CoMACsu-	[IDXi⊗], [Rw_m⊗]	93 Xm 60 rrrr:rqqq	yes

CoMACu Unsigned Multiply-Accumulate & Round CoMACu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (ACC) + (tmp) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description

Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended and then added to the 40-bit ACC register contents. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACu	Rw_n, Rw_m, rnd	A3 nm 11 00	no
CoMACu	Rw_n, [Rw_m⊗], rnd	83 nm 11 rrrr:rqqq	yes
CoMACu	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 11 rrrr:rqqq	yes

CoMACu Unsigned Multiply-Accumulate CoMACu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACu op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (ACC) + (tmp)

Description
Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended and then added to the 40-bit ACC register contents before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACu	Rw_n, Rw_m	A3 nm 10 00	no
CoMACu	Rw_n, [Rw_m⊗]	83 nm 10 rrrr:rqqq	yes
CoMACu	[IDXi⊗], [Rw_m⊗]	93 Xm 10 rrrr:rqqq	yes

CoMACu- Unsigned Multiply-Accumulate CoMACu-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACu- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

$(tmp) \leftarrow (op1) * (op2)$
$(ACC) \leftarrow (ACC) - (tmp)$

Description

Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended and then subtracted from the 40-bit ACC register contents before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACu-	Rw_n, Rw_m	A3 nm 20 00	no
CoMACu-	Rw_n, [Rw_m⊗]	83 nm 20 rrrr:rqqq	yes
CoMACu-	[IDXi⊗], [Rw_m⊗]	93 Xm 20 rrrr:rqqq	yes

CoMACus — Mixed Multiply-Accumulate with Round — CoMACus

Group — Multiply/Multiply-Accumulate Instructions

Syntax — **CoMACus op1, op2, rnd**

Source Operand(s) — op1, op2 $\rightarrow$ WORD

Destination Operand(s) — ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (ACC) + (tmp) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description

Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then added to the 40-bit ACC register contents. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMACus	Rw_n, Rw_m, rnd	A3 nm 91 00	no
CoMACus	Rw_n, [Rw_m⊗], rnd	83 nm 91 rrrr:rqqq	yes
CoMACus	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 91 rrrr:rqqq	yes

CoMACus Mixed Multiply-Accumulate CoMACus

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACus op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op1) * (op2)
(ACC) $\leftarrow$ (ACC) + (tmp)

Description

Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then added to the 40-bit ACC register contents before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACus	Rw_n, Rw_m	A3 nm 90 00	no
CoMACus	Rw_n, [Rw_m⊗]	83 nm 90 rrrr:rqqq	yes
CoMACus	[IDXi⊗], [Rw_m⊗]	93 Xm 90 rrrr:rqqq	yes

CoMACus- Mixed Multiply-Accumulate CoMACus-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMACus- op1, op2**

Source Operand(s) op1, op2 → WORD

Destination Operand(s) ACC → 40-bit signed value

Operation

(tmp) ← (op1) * (op2)
(ACC) ← (ACC) - (tmp)

Description
Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then subtracted from the 40-bit ACC register contents before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMACus-	Rw_n, Rw_m	A3 nm A0 00	no
CoMACus-	Rw_n, [Rw_m⊗]	83 nm A0 rrrr:rqqq	yes
CoMACus-	[IDXi⊗], [Rw_m⊗]	93 Xm A0 rrrr:rqqq	yes

CoMAX Maximum CoMAX

Group Compare Instructions

Syntax **CoMAX op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op2) || (op1)
(ACC) $\leftarrow$ max((ACC),(tmp))

Description
Compares a signed 40-bit operand against the 40-bit ACC register contents. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). If the contents of the 40-bit ACC register is smaller than the 40-bit operand, then the ACC register is loaded with it. Otherwise the ACC register remains unchanged. The MS bit of the MCW register does not affect the result.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	0	*	*	no

SL	Set if the contents of ACC is changed. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMAX	Rw_n, Rw_m	A3 nm 3A 00	no
CoMAX	Rw_n, [Rw_m⊗]	83 nm 3A rrrr:rqqq	yes
CoMAX	[IDXi⊗], [Rw_m⊗]	93 Xm 3A rrrr:rqqq	yes

CoMIN Minimum CoMIN

Group Compare Instructions

Syntax **CoMIN op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
(tmp) ← (op2) || (op1)
(ACC) ← min((ACC),(tmp))
```

Description
Compares a signed 40-bit operand against the 40-bit ACC register contents. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). If the contents of the ACC register is greater than the 40-bit operand, then the ACC register is loaded with it. Otherwise the ACC register remains unchanged. The MS bit of the MCW register does not affect the result.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	0	*	*	no

SL	Set if the contents of ACC is changed. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMIN	Rw_n, Rw_m	A3 nm 7A 00	no
CoMIN	Rw_n, [Rw_m⊗]	83 nm 7A rrrr:rqqq	yes
CoMIN	[IDXi⊗], [Rw_m⊗]	93 Xm 7A rrrr:rqqq	yes

CoMOV Memory to Memory Move CoMOV

Group Data Movement Instructions

Syntax **CoMOV op1, op2**

Source Operand(s) op2 $\rightarrow$ WORD

Destination Operand(s) op1 $\rightarrow$ WORD

Operation

(op1) $\leftarrow$ (op2)

Description

Moves the contents of the memory location specified by the source operand op2 to the memory location specified by the destination operand op1. Note that, unlike for the other instructions, IDXi can address the entire memory. This instruction does not affect the MAC Flags but modify the CPU Flags as any other MOV instruction.

Note: CoMOV is the only MAC Unit instruction which affects the CPU flags. MAC Flags are not affected.

CPU Flags

E	Z	V	C	N
*	*	-	-	*

E	Set if the value of op2 represents the lowest possible negative number. Cleared otherwise. Used to signal the end of a table.
Z	Set if the value of the source operand op2 equals zero. Cleared otherwise.
V	Not affected.
C	Not affected.
N	Set if the most significant bit of the source operand op2 is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMOV	[IDXi⊗], [Rw$_m$⊗]	D3 Xm 00 rrrr:rqqq	yes

CoMUL Signed Multiply with Round CoMUL

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMUL op1, op2, rnd**

Source Operand(s) op1, op2 → WORD

Destination Operand(s) ACC → 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (ACC) ← ((op1) * (op2)) <<1 + 00 0000 8000h
ELSE
        (ACC) ← (op1) * (op2) + 00 0000 8000h
END IF
(MAL) ← 0
```

Description

Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted. Finally, the obtained result is 2's complement rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	0	*	*	yes

SL	Not affected when MP or MS are cleared, otherwise, only set in case of 8000h by 8000h multiplication.
E	Set when MP is set and MS is cleared and in case of 8000h by 8000h multiplication. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoMUL	Rw_n, Rw_m, rnd	A3 nm C1 00	no
CoMUL	Rw_n, [Rw_m⊗], rnd	83 nm C1 0:0qqq	no
CoMUL	[IDXi⊗], [Rw_m⊗], rnd	93 Xm C1 0:0qqq	no

CoMUL Signed Multiply CoMUL

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMUL op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (ACC) ← ((op1) * (op2)) <<1
ELSE
        (ACC) ← (op1) * (op2)
END IF
```

Description

Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	0	*	*	yes

SL	Not affected when MP or MS are cleared, otherwise, only set in case of 8000h by 8000h multiplication.
E	Set when MP is set and MS is cleared and in case of 8000h by 8000h multiplication. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMUL	Rw_n, Rw_m	A3 nm C0 00	no
CoMUL	Rw_n, [Rw_m⊗]	83 nm C0 0:0qqq	no
CoMUL	[IDXi⊗], [Rw_m⊗]	93 Xm C0 0:0qqq	no

CoMUL- Signed Multiply CoMUL-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMUL- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
IF (MP = 1) THEN
        (ACC) ← - ((op1) * (op2)) <<1
ELSE
        (ACC) ← - ((op1) * (op2))
END IF
```

Description

Multiplies the two signed 16-bit source operands op1 and op2. The obtained signed 32-bit product is first sign-extended, then if the MP flag is set, it is one-bit left shifted and finally it is negated before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	0	-	0	*	*	no

SL	Not affected.
E	Always cleared.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMUL-	Rw_n, Rw_m	A3 nm C8 00	no
CoMUL-	Rw_n, [Rw_m⊗]	83 nm C8 0:0qqq	no
CoMUL-	[IDXi⊗], [Rw_m⊗]	93 Xm C8 0:0qqq	no

CoMULsu — Mixed Multiply & Round — CoMULsu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMULsu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ (op1) * (op2) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description

Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	0	-	0	*	*	no

SL	Not affected.
E	Always cleared.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMULsu	Rw_n, Rw_m, rnd	A3 nm 41 00	no
CoMULsu	Rw_n, [Rw_m⊗], rnd	83 nm 41 0:0qqq	no
CoMULsu	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 41 0:0qqq	no

CoMULsu Mixed Multiply CoMULsu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMULsu op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ (op1) * (op2)

Description

Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	0	-	0	*	*	no

SL	Not affected.
E	Always cleared.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMULsu	Rw_n, Rw_m	A3 nm 40 00	no
CoMULsu	Rw_n, [Rw_m⊗]	83 nm 40 0:0qqq	no
CoMULsu	[IDXi⊗], [Rw_m⊗]	93 Xm 40 0:0qqq	no

CoMULsu- Mixed Multiply CoMULsu-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMULsu- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ - ((op1) * (op2))

Description

Multiplies the two signed and unsigned 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then negated before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	0	-	0	*	*	no

SL	Not affected.
E	Always cleared.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMULsu-	Rw_n, Rw_m	A3 nm 48 00	no
CoMULsu-	Rw_n, [Rw_m⊗]	83 nm 48 0:0qqq	no
CoMULsu-	[IDXi⊗], [Rw_m⊗]	93 Xm 48 0:0qqq	no

CoMULu — Unsigned Multiply with Round — CoMULu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMULu op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ (op1) * (op2) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description
Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended and then rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	0	*	0	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Always cleared.

Note: The behavior of E and SL flag have been changed to guarantee correct arithmetic. If two large 16-bit unsigned numbers are multiplied, then the result cannot be represented in a 32-bit signed format. In this case, either the ACC extension must be used (automatic saturation disabled, MS=0), or the result must be saturated to a 32-bit signed value (automatic saturation enabled, MS=1).

Encoding

Mnemonic		Format	Repeat
CoMULu	Rw_n, Rw_m, rnd	A3 nm 01 00	no
CoMULu	Rw_n, [Rw_m⊗], rnd	83 nm 01 0:0qqq	no
CoMULu	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 01 0:0qqq	no

CoMULu — Unsigned Multiply — CoMULu

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMULu op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ (op1) * (op2)

Description

Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	0	*	0	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Always cleared.

Note: The behavior of E and SL flag have been changed to guarantee correct arithmetic. If two large 16-bit unsigned numbers are multiplied, then the result cannot be represented in a 32-bit signed format. In this case, either the ACC extension must be used (automatic saturation disabled, MS=0), or the result must be saturated to a 32-bit signed value (automatic saturation enabled, MS=1).

Encoding

Mnemonic		**Format**	**Repeat**
CoMULu	Rw_n, Rw_m	A3 nm 00 00	no
CoMULu	Rw_n, [Rw_m⊗]	83 nm 00 0:0qqq	no
CoMULu	[IDXi⊗], [Rw_m⊗]	93 Xm 00 0:0qqq	no

CoMULu- Unsigned Multiply CoMULu-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMULu- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

$$(ACC) \leftarrow - ((op1) * (op2))$$

Description

Multiplies the two unsigned 16-bit source operands op1 and op2. The obtained unsigned 32-bit product is first zero-extended and then negated before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	-	0	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Note: The behavior of E and SL flag have been changed to guarantee correct arithmetic. If two large 16-bit unsigned numbers are multiplied, then the result cannot be represented in a 32-bit signed format. In this case, either the ACC extension must be used (automatic saturation disabled, MS=0), or the result must be saturated to a 32-bit signed value (automatic saturation enabled, MS=1).

Encoding

Mnemonic		**Format**	**Repeat**
CoMULu-	Rw_n, Rw_m	A3 nm 08 00	no
CoMULu-	Rw_n, [Rw_m⊗]	83 nm 08 0:0qqq	no
CoMULu-	[IDXi⊗], [Rw_m⊗]	93 Xm 08 0:0qqq	no

CoMULus Mixed Multiply with Round CoMULus

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMULus op1, op2, rnd**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ (op1) * (op2) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description

Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	0	-	0	*	*	no

SL	Not affected.
E	Always cleared.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMULus	Rw_n, Rw_m, rnd	A3 nm 81 00	no
CoMULus	Rw_n, [Rw_m⊗], rnd	83 nm 81 0:0qqq	no
CoMULus	[IDXi⊗], [Rw_m⊗], rnd	93 Xm 81 0:0qqq	no

CoMULus Mixed Multiply CoMULus

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMULus op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ (op1) * (op2)

Description

Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	0	-	0	*	*	no

SL	Not affected.
E	Always cleared.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMULus	Rw_n, Rw_m	A3 nm 80 00	no
CoMULus	Rw_n, [Rw_m⊗]	83 nm 80 0:0qqq	no
CoMULus	[IDXi⊗], [Rw_m⊗]	93 Xm 80 0:0qqq	no

CoMULus- Mixed Multiply CoMULus-

Group Multiply/Multiply-Accumulate Instructions

Syntax **CoMULus- op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ - ((op1) * (op2))

Description

Multiplies the two unsigned and signed 16-bit source operands op1 and op2, respectively. The obtained signed 32-bit product is first sign-extended and then negated before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	0	-	0	*	*	no

SL	Not affected.
E	Always cleared.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoMULus-	Rw_n , Rw_m	A3 nm 88 00	no
CoMULus-	Rw_n, [Rw_m⊗]	83 nm 88 0:0qqq	no
CoMULus-	[IDXi⊗], [Rw_m⊗]	93 Xm 88 0:0qqq	no

CoNEG Negate Accumulator CoNEG

Group Arithmetic Instructions

Syntax **CoNEG**

Source Operand(s) ACC $\rightarrow$ 40-bit signed value

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ 0 - (ACC)

Description

The ACC register contents is subtracted from zero before being stored in the 40-bit ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic	**Format**	**Repeat**
CoNEG	A3 00 32 00	no

CoNEG Negate Accumulator with Round CoNEG

Group Arithmetic Instructions

Syntax **CoNEG rnd**

Source Operand(s) ACC $\rightarrow$ 40-bit signed value

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(ACC) $\leftarrow$ 0 - (ACC) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description

The ACC register contents is subtracted from zero and the result is rounded before being stored in the 40-bit ACC register. The MAL register is cleared.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoNEG	rnd	A3 00 72 00	no

CoNOP No-Operation CoNOP

Group Arithmetic Instructions

Syntax CoNOP

Source Operand(s)	none
Destination Operand(s)	none

Operation

No Operation

Description

Modifies the address pointers.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	-	-	-	-	-	no

SL	Not affected.
E	Not affected.
SV	Not affected.
C	Not affected.
Z	Not affected.
N	Not affected.

Encoding

Mnemonic		**Format**	**Repeat**
CoNOP	[IDXi⊗], [Rw_m⊗]	93 Xm 5A rrrr:rqqq	yes
CoNOP	[IDXi⊗]	93 X0 5A rrrr:r000	yes
CoNOP	[Rw_m⊗]	93 0m 5A rrrr:rqqq	yes

CoRND Round Accumulator CoRND

Group Shift Instructions

Syntax **CoRND**

Source Operand(s) ACC $\rightarrow$ 40-bit signed value

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value signed value

Operation

(ACC) $\leftarrow$ (ACC) + 00 0000 8000h
(MAL) $\leftarrow$ 0

Description
Rounds the ACC register contents by adding 0000 8000h to it and stores the result in the ACC register. The MAL register is cleared.

Note: CoRND is a shortname for CoASHR #0, rnd

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic overflow occurred. Not affected otherwise.
C	Set if a carry is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic	**Format**	**Repeat**
CoRND	A3 00 B2 00	no

CoSHL Accumulator Logical Shift Left CoSHL

Group Shift Instructions

Syntax **CoSHL op1**

Source Operand(s) op1 $\rightarrow$ shift counter

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
(count) ← (op1)
(C) <- (ACC[39])
DO WHILE ((count) ≠ 0)
    (C) ← (ACC[39])
    (ACC[n]) ← (ACC[n-1]) [n=39...1]
    (ACC[0]) ← 0
    (count) ← (count) -1
END WHILE
```

Description

Shifts the 40-bit ACC register contents left by the number of times specified by operand op1. The least significant bits of the result are filled with zeros accordingly. Only shift values from 0 to 8 (inclusive) are allowed. op1 can be either a 4-bit unsigned immediate data or the 4 least significant bits (considered as unsigned data) of a directly or indirectly addressed operand.
When the MS bit of the MCW register is set and when a 32-bit overflow or underflow occurs, the obtained result becomes 00'7fff'fffh or ff'8000'000h, respectively.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if the bit shifted out last is different from the new N flag.
C	Carry flag is set according to the last most significant bit shifted out of ACC or according to the sign of ACC.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoSHL	#data4	A3 00 82 0sss:s000	no
CoSHL	Rw_n	A3 nn 8A rrrr:r000	yes
CoSHL	$[Rw_m\otimes]$	83 mm 8A rrrr:rqqq	yes

CoSHR Accumulator Logical Shift Right CoSHR

Group Shift Instructions

Syntax **CoSHR op1**

Source Operand(s) op1 → shift counter

Destination Operand(s) ACC → 40-bit signed value

Operation

```
(count) ← (op1)
(C) ← 0
DO WHILE (count) ≠ 0
        ((ACC[n]) ← (ACC[n+1]) [n=0-38]
        (ACC[39]) ← 0
        (count) ← (count) -1
END WHILE
```

Description

Shifts the 40-bit ACC register contents right the number of times as specified by the operand op1. The most significant bits of the result are filled with zeros accordingly. Only shift values from 0 to 8 (inclusive) are allowed. op1 can be either a 4-bit unsigned immediate data or the 4 least significant bits (considered as unsigned data) of a directly or indirectly addressed operand. The MS bit of the MCW register does not affect the result.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	*	-	0	*	*	no

SL	Not affected.
E	Set if the MAE is used. Cleared otherwise.
SV	Not affected.
C	Always cleared.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		Format	Repeat
CoSHR	#data4	A3 00 92 0sss:s000	no
CoSHR	Rw_n	A3 nn 9A rrrr:r000	yes
CoSHR	[Rw_m⊗]	83 mm 9A rrrr:rqqq	yes

CoSTORE — Store a MAC Unit Register — CoSTORE

Group — Data Movement Instructions

Syntax — **CoSTORE op1, op2**

Source Operand(s) — op2 $\rightarrow$ WORD

Destination Operand(s) — op1 $\rightarrow$ WORD

Operation

(op1) $\leftarrow$ (op2)

Description
Moves the contents of a MAC-Unit register specified by the source operand op2 to the location specified by the destination operand op1.

MAC Flags

SL	E	SV	C	Z	N	Sat.
-	-	-	-	-	-	no

SL	Not affected.
E	Not affected.
SV	Not affected.
C	Not affected.
Z	Not affected.
N	Not affected.

Note: Due to pipeline side effects, CoSTORE cannot be directly followed by a MOV instruction that also uses a MAC Unit register (MSW, MAH, MAL, MAS, MRW or MCW) as source operand. In such cases a NOP must be inserted between the CoSTORE and MOV instruction.

Encoding

Mnemonic		**Format**	**Repeat**
CoSTORE	Rw_n, CoReg	C3 nn wwww:w000 00	no
CoSTORE	[Rw_n⊗], CoReg	B3 nn wwww:w000 rrrr:rqqq	yes

CoSUB Subtract CoSUB

Group Arithmetic Instructions

Syntax **CoSUB op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op2) || (op1)
(ACC) $\leftarrow$ (ACC) - (tmp)

Description

Subtracts a 40-bit operand from the 40-bit ACC contents and stores the result in the ACC register. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW).

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoSUB	Rw_n, Rw_m	A3 nm 0A 00	no
CoSUB	Rw_n, [Rw_m⊗]	83 nm 0A rrrr:rqqq	yes
CoSUB	[IDXi⊗], [Rw_m⊗]	93 Xm 0A rrrr:rqqq	yes

CoSUB2 Subtract CoSUB2

Group Arithmetic Instructions

Syntax **CoSUB2 op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ 2 * (op2) || (op1)
(ACC) $\leftarrow$ (ACC) - (tmp)

Description

Subtracts a 40-bit operand from the 40-bit ACC contents and stores the result in the ACC register. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). The 40-bit operand is then multiplied by two before being subtracted from the ACC register.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoSUB2	Rw_n, Rw_m	A3 nm 4A 00	no
CoSUB2	Rw_n, [$Rw_m \otimes$]	83 nm 4A rrrr:rqqq	yes
CoSUB2	[IDXi], [$Rw_m \otimes$]	93 Xm 4A rrrr:rqqq	yes

CoSUB2R Subtract CoSUB2R

Group Arithmetic Instructions

Syntax **CoSUB2R op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

```
(tmp) ← 2 * (op2) || (op1)
(ACC) ← (tmp) - (ACC)
```

Description
Subtracts the 40-bit ACC contents from a 40-bit operand and stores the result in the ACC register. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW). The 40-bit operand is then multiplied by two before the 40-bit ACC is subtracted.

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoSUB2R	Rw_n, Rw_m	A3 nm 52 00	no
CoSUB2R	Rw_n, [Rw_m⊗]	83 nm 52 rrrr:rqqq	yes
CoSUB2R	[IDXi⊗], [Rw_m⊗]	93 Xm 52 rrrr:rqqq	yes

CoSUBR Subtract CoSUBR

Group Arithmetic Instructions

Syntax **CoSUBR op1, op2**

Source Operand(s) op1, op2 $\rightarrow$ WORD

Destination Operand(s) ACC $\rightarrow$ 40-bit signed value

Operation

(tmp) $\leftarrow$ (op2) || (op1)
(ACC) $\leftarrow$ (tmp) - (ACC)

Description

Subtracts the 40-bit ACC contents from a 40-bit operand and stores the result in the ACC register. The 40-bit operand is a sign-extended result of the concatenation of the two source operands, op1 (LSW) and op2 (MSW).

MAC Flags

SL	E	SV	C	Z	N	Sat.
*	*	*	*	*	*	yes

SL	Set if the contents of ACC is automatically saturated. Not affected otherwise.
E	Set if the MAE is used. Cleared otherwise.
SV	Set if an arithmetic underflow occurred. Not affected otherwise.
C	Set if a borrow is generated. Cleared otherwise.
Z	Set if result equals zero. Cleared otherwise.
N	Set if the most significant bit of the result is set. Cleared otherwise.

Encoding

Mnemonic		**Format**	**Repeat**
CoSUBR	Rw_n, Rw_m	A3 nm 12 00	no
CoSUBR	Rw_n, [Rw_m⊗]	83 nm 12 rrrr:rqqq	yes
CoSUBR	[IDXi⊗], [Rw_m⊗]	93 Xm 12 rrrr:rqqq	yes

7 Instruction Index

This section lists alphabetically all C166S MAC Unit instructions together with references to respective pages holding the detailed descriptions. This helps to quickly find the explanation of any specific MAC instruction.

8 Keyword Index

This section lists a number of keywords which refer to specific details of the C166S V1 Multiply-Accumulate Unit in terms of its architecture and functions. This helps to quickly find the answer to specific questions about the C166S V1 MAC.

Infineon goes for Business Excellence

“Business excellence means intelligent approaches and clearly defined processes, which are both constantly under review and ultimately lead to good operating results.
Better operating results and business excellence mean less idleness and wastefulness for all of us, more professional success, more accurate information, a better overview and, thereby, less frustration and more satisfaction.”

Dr. Ulrich Schumacher

h t t p : / / w w w . i n f i n e o n . c o m

Published by Infineon Technologies AG