

SECTION 12

FILTER CO-PROCESSOR

12.1	Introduction	12-3
12.2	Features.	12-4
12.3	Block Description	12-4
12.4	Programming Model	12-6
12.5	Operation Modes	12-14
12.6	Performance Analysis	12-40

12.1 INTRODUCTION

The Filter Co-Processor (FCOP) is a peripheral module designed as a general purpose fully programmable complex FIR filter, with up to 21 complex taps.

It also has dedicated modes of operation optimized for cellular Global System for Mobile Communications (GSM) base station applications, taking advantage of the special GMSK modulation scheme used in GSM.

12.1.1 FCOP Support for GSM

The FCOP assists the DSP56300 core to perform channel equalization of one radio carrier in a GSM base station, which involves both correlation and matched filtering. A matched filter is commonly used in channel equalization to maximize the Signal-to-Noise Ratio (SNR) at sampling time. In the base station receiver path (uplink), FCOP performs cross correlation between the received training sequence and a known mid-amble sequence. The result of this correlation is used to determine the impulse channel response, which is then used to calculate the coefficients for match filtering on the received data symbols (bits).

The FCOP simplifies computation of the cross correlation and match filter outputs fed into the Maximum Likelihood Sequential Estimation (MLSE) process performed by the Viterbi Co-Processor (VCOP). In GSM, while the MLSE algorithm uses the Ungerboeck scheme, the match filter output can be represented by a sequence of ordered pairs (x, y) such that x, y are real numbers.

The main features of FCOP supporting GSM are as follows:

- Special optimized mode (Mode 3) for performing cross correlation between the received training sequence and a pre-defined mid-amble
- Special optimized mode (Mode 2) for performing matched filtering required for channel equalization using the Ungerboeck scheme
- Internal data and coefficient memory banks supports GSM normal burst, as well as GSM access burst.

12.2 FEATURES

- Fully programmable complex FIR filter with 16-bit resolution
- Operates concurrently with the core processor and requires minimal CPU intervention
- Independent internal 84×16 -bit data memory bank and 42×16 -bit coefficient memory bank
- Four modes of operation with optimized performance
 - Mode 0—Real FIR machine with up to 42 real taps
 - Mode 1—Complex FIR machine with up to 21 complex taps
 - Mode 2—Complex FIR machine generating pure real and imaginary outputs alternately. In GSM, used for performing match filtering
 - Mode 3—Complex correlation of a complex data sequence with a pure real and imaginary sequence. In GSM, used for performing cross correlation between the received training sequence and a pre-defined mid-amble
- Two options for filter output decimation available in each operation mode, without loss of performance
 - No decimation
 - Decimation by 2
- I/O data transfers via core or DMA for minimal core involvement
- Four-word deep input data buffer (24-bit word) for maximal performance

12.3 BLOCK DESCRIPTION

The FCOP comprises the following main functional blocks: PMB Interface — includes the Data Input Buffer, Coefficients Input Buffer, Output Buffer, and Filter Counter (FCNT)—, FCOP Data Memory Bank (FDM), FCOP Coefficients Memory Bank (FCM), FCOP MAC machine (FMAC), Address Generator, and Control Logic.

The block diagram is presented in **Figure 12-1**.

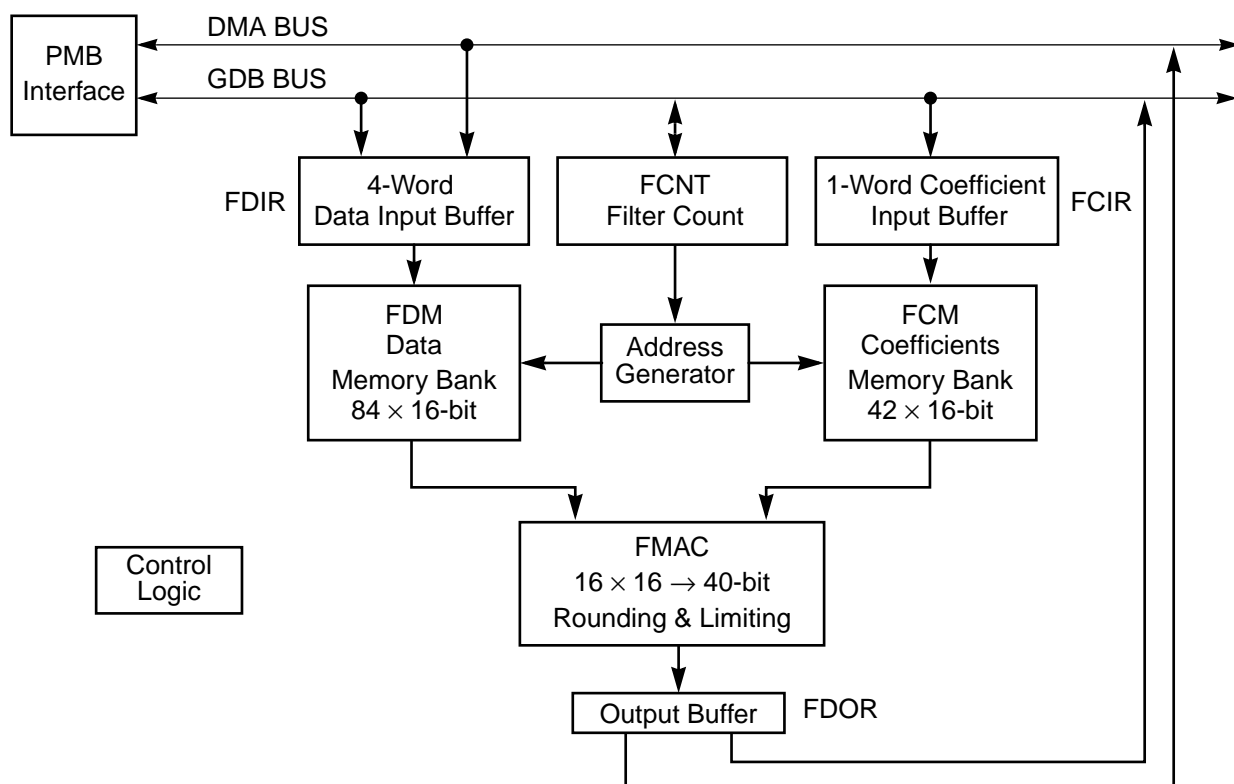


Figure 12-1 Filter Co-Processor Block Diagram

12.3.1 Peripheral Module Bus (PMB) Interface

The Peripheral Module Bus (PMB) interface block provides the control and status registers, buffers the internal bus from the coprocessor, decodes addresses, and generates and controls the handshake signals required for DMA and interrupt operations. The block generates the interrupt and DMA trigger signals, whenever data transfer is required.

The control and status registers in the PMB are described in detail in the programming model (see **Section 12.4**). The interface registers are accessible to the DSP56300 core through the PMB.

12.3.2 FCOP Memory Banks

The FCOP contains two memory banks: the FCOP Data Memory bank (FDM) and the FCOP Coefficients Memory bank (FCM).

- **FCOP Data Memory bank (FDM):**
84-word 16-bit wide memory, used to store data samples for filter processing by the FCOP. It is written via a 4-word FIFO (FDIR) and its addressing is generated by the FCOP Address Generation logic. The data samples are sequentially read from FDM into the MAC for filter processing. The FDM can be written by both the core and DMA.
- **FCOP Coefficients Memory Bank (FCM):**
42-word 16-bit wide memory, used to store filter coefficients for filter processing by the FCOP. It is written via an input buffer (FCIR) and its addressing is generated by the FCOP Address Generation logic. The filter samples are sequentially read from FDM into the MAC for filter processing. The FCM can only be written by the core.

12.3.3 Multiplier and Accumulator (FMAC)

The FCOP Multiplier and Accumulator (FMAC) machine is capable of performing a 16-bit \times 16-bit multiplication with accumulation in a 40-bit accumulator. The FMAC operates in a pipeline fashion, with the multiplication performed in one clock cycle and the accumulation in the following clock cycle. The throughput is one MAC result per clock cycle. The two MAC operands are read from the FDM and the FCM. The full 40-bit width of the accumulator is used for intermediate results during the filter calculations. After all the filter taps have been processed, the result is rounded to the nearest even and limited to the greatest (most positive) number (\$7FFF) if overflow occurred or to the least (most negative) number (\$8000) if underflow occurred. The rounding and limiting operations take two additional clock cycles after the filter processing is completed. The 16-bit result from FMAC is stored in the FDOR to be read by the DSP56300 core.

12.4 PROGRAMMING MODEL

The programming model discusses the FCOP registers available to the programmer, and interrupts and DMA access.

12.4.1 FCOP Registers

The FCOP programmable registers are listed in **Table 12-1**. The registers are described in detail in the following sections.

Table 12-1 FCOP Programming Model

Base Address Plus	Register Name	Register Abbreviation
\$0	FCOP Data Input Register	FDIR
\$1	FCOP Data Output Register	FDOR
\$2	FCOP Coefficients Input Register	FCIR
\$3	FCOP Filter Count Register	FCNT
\$4	FCOP Control/Status Register	FCSR
Note: The base address is found in the ioequ.asm file, in Appendix B. It is \$FFFFB0.		

All FCOP registers are 16 bits wide. They are of three types as listed in Table 12-2.

Table 12-2 3 Types of 16-Bit FCOP Registers

Data source	<ul style="list-style-type: none"> FCOP Data Output Register (FDOR)
Data destination	<ul style="list-style-type: none"> FCOP Data Input Register (FDIR) FCOP Coefficients Input Register (FCIR)
Non-data	<ul style="list-style-type: none"> FCOP Filter Count Register (FCNT) FCOP Control Status Register (FCSR)

Reads and writes of data and non-data are treated differently, as detailed below in Table 12-3.

Table 12-3 FCOP Register Read/Write Handling

Data Transfers	<ul style="list-style-type: none"> When a data source (FDOR) is read to a 24-bit destination (bus, register, memory, etc.), the 16-bit data value occupies the 16 Most Significant Bits (MSBs) of the 24-bit destination and the 8 Least Significant Bits (LSBs) are zeroed. When a 24-bit source (bus, register, memory, etc.) is written to a data destination (FDIR or FCIR), the 16 MSBs of the 24-bit source will be written to the 16-bit destination.
Non-Data Transfers	<ul style="list-style-type: none"> When a non-data register (FCNT or FCSR) is read to a 24-bit destination (bus, register, memory, etc.), the 16-bit value will be zero extended to a 24-bit value. When a 24-bit source (bus, register, memory, etc.) is written to a non-data register, the 8 MSBs must be written with zero for future compatibility.

12.4.2 FCOP Data Input Register (FDIR)

The FCOP Data Input Register (FDIR) is a 4-word deep 16-bit wide FIFO used for DSP-to-FCOP data transfers. Up to four data samples can be written into FDIR using the same address. Data from FDIR is transferred to the FCOP Data Memory bank (FDM) for filter processing. For proper operation, data should be written to FDIR only if the FDIBE status bit is set, indicating that the FIFO is empty. Writing to FDIR clears FDIBE. The user may use interrupt requests or DMA requests to trigger the DSP56300 core for data transfers. FDIR can be written by the DSP56300 core and DMA. FDIR is also referred to as the FCOP Data Input Buffer.

12.4.3 FCOP Data Output Register (FDOR)

The FCOP Data Output Register (FDOR) is a 16-bit wide, read-only register used for FCOP-to-DSP data transfers. Data is transferred from the FMAC to FDOR after processing of all filter taps is completed for a specific set of input samples, that is, after filter processing has been completed. For proper operation, data should be read from FDOR only if the FDOBF status bit is set, indicating that FDOR contains data. The user may use interrupt or DMA requests to trigger the DSP56300 core for data transfers. FDOR can be read by the DSP56300 core and DMA. FDOR is also referred to as the FCOP Data Output Buffer.

12.4.4 FCOP Coefficients Input Register (FCIR)

The FCOP Coefficients Input Register (FCIR) is a 16-bit write-only register used for DSP-to-FCOP coefficients transfers. The filter coefficients are written to FCIR and transferred to the FCOP Coefficients Memory bank (FCM) before starting the filter processing. The FCIR can be written by the DSP56300 core.

12.4.5 FCOP Filter Count Register (FCNT)

The FCOP Filter Count Register (FCNT) is a 16-bit read/write register used for setting the filter length (number of filter taps). The actual value written to FCNT register is the number of coefficient values minus one. The number of coefficient values is actually the number of locations used in the FCM. For a real FIR filter, the number of coefficient values is equal to the number of filter taps. For a complex FIR filter, the number of coefficient values is twice the number of filter taps. The number of taps in FCNT is used by the FCOP Address Generation logic to supply the correct addressing to the FDM and FCM memory banks.


The FCOP can only be written to by the core. Write to FCNT before enabling FCOP (by setting FEN). FCNT should be changed only when FCOP is in the FCOP individual reset state (FEN = 0), otherwise improper operation may result. The number stored in FCNT is used by the FCOP Address Generation logic to generate the correct addressing for the FDM and the FCM.

12.4.6 FCOP Control/Status Register (FCSR)

The FCOP Control/Status Register (FCSR) is a 16-bit read/write register used by the DSP56300 core to control the main operation modes and to monitor the status of the module. The FCSR bits are described in the following paragraphs. All FCSR bits are cleared after hardware and software reset.

Programming Model

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDOBF	FDIBE		FSAT	FDOIE	FDIIE		FDCM			FOM1	FOM0				FEN

 Reserved unused bit, read as zero, should be written with zero for future compatibility

 Reserved bit for internal use (such as testing), must be written with zero for proper operation

AA1119

Figure 12-2 FCOP Control/Status Register (FCSR)

12.4.6.1 FCOP Enable (FEN)—FCSR Bit 0

The FCOP Enable (FEN) read/write control bit, when set, enables FCOP operation. When FEN is cleared, the FCOP is disabled and is in the FCOP individual reset state. In this state, the FCNT contents and the FCSR control bits are unchanged, while the FCSR status bits and internal logic bits are reset to the same state produced by hardware or software reset. The status bits are cleared (for instance FDIBE, FDOIE, FSAT). The rest of the control bits in FCSR and other registers remain unchanged. The internal logic is cleared, hence the data remaining in FDIR, FDOR, and FCIR become meaningless.

12.4.6.2 FCOP Operation Mode (FOM[1:0])—FCSR Bits 4–5

The FCOP Operation Mode (FOM[1:0]) read/write control bits select the operation mode. The operation modes are shown in **Table 12-4**. FOM[1:0] should only be changed when FCOP is in the FCOP individual reset state (FEN = 0), otherwise improper operation may result. FOM[1:0] are cleared by hardware or software reset. For a detailed description of FCOP operation in the various modes, refer to **Figure 12.5**.

Table 12-4 FCOP Operation Modes

FOM1	FOM0	Mode	Mode Function
0	0	0	Real FIR Filter
0	1	1	Full Complex FIR Filter
1	0	2	Complex FIR filter with alternate Pure Real/Imaginary outputs
1	1	3	Optimized Complex Correlation

12.4.6.3 FCOP Decimation (FDCM)—FCSR Bit 8

The FCOP Decimation (FDCM) read/write control bit determines the decimation function. When FDCM is set, decimation by 2 occurs and when FDCM is cleared, no decimation occurs. No decimation is chosen for non-oversampled data while decimation by 2 is used for 2× oversampled data. FDCM should only be changed when FCOP is in the FCOP individual reset state (FEN = 0), otherwise improper operation may result. FDCM is cleared by hardware or software reset. For more details about the effect of FDCM on the module's operation, refer to **Section 12.5**.

12.4.6.4 FCOP Data Input Interrupt Enable (FDIIE)—FCSR Bit 10

The FCOP Data Input Interrupt Enable (FDIIE) read/write control bit, when set, enables the Data Input Buffer Empty Interrupt. When FDIIE is cleared, the Data Input Buffer Empty Interrupt is disabled, and the FDIBE status bit should be polled to determine if FDIR is empty. The following table describes the effect of the possible combined states of FDIIE and FDIBE:

Table 12-5 Relationship of FDIIE and FDIBE

FDIIE	FDIBE	Effect
0	0	Data Input Buffer Empty Interrupt is disabled. FCOP Data Input Register (FDIR) is not empty. Do not write to FDIR.
0	1	Data Input Buffer Empty Interrupt is disabled. FCOP Data Input Register (FDIR) is empty. DMA write to FDIR possible.
1	0	Data Input Buffer Empty Interrupt is enabled. FCOP Data Input Register (FDIR) is not empty. Do not write to FDIR.
1	1	Data Input Buffer Empty Interrupt is enabled. FCOP Data Input Register (FDIR) is empty. FCOP requests a Data Input Buffer Empty Interrupt service from the DSP56300 core. Interrupt write to FDIR is possible.

DMA transfer is enabled if a DMA channel is activated and allocated for FCOP Data Input Buffer Empty (FDIBE is set). FDIR should be written either by the interrupt routine or the DMA transfer, but not both, so it is highly recommended to enable either the interrupt or DMA, but not both.

12.4.6.5 FCOP Data Output Interrupt Enable (FDOIE)—FCSR Bit 11

The FCOP Data Output Interrupt Enable (FDOIE) read/write control bit, when set, enables the Data Output Interrupt. When FDOIE is cleared, the Data Output Interrupt is disabled and the FDOBF status bit should be polled to determine if FDOR is full. The following table describes the effect of the possible combined states of FDOIE and FDOBF:

Table 12-6 Relationship of FDOIE and FDOBF

FDOIE	FDOBF	Effect
0	0	Data Output Interrupt is disabled. FCOP Data Output Register (FDOR) is not full. Do not read from FDOR.
0	1	Data Output Interrupt is disabled. FCOP Data Output Register (FDOR) is full. DMA read from FDOR is possible.
1	0	Data Output Interrupt is enabled. FCOP Data Output Register (FDOR) is not full. Do not read from FDOR.
1	1	Data Output Interrupt is enabled. FCOP Data Output Register (FDOR) is full. FCOP requests a Data Output Interrupt service from the DSP56300 core. Interrupt read from FDOR is possible.

DMA transfer is enabled if a DMA channel is activated and allocated for FCOP Data Output Buffer Full (FDOBF is set). FDOR should be read by the interrupt routine or by the DMA transfer, but not both; thus it is highly recommended to enable either the interrupt or DMA, but not both.

12.4.6.6 FCOP Data Saturation (FSAT)—FCSR Bit 12

The FCOP Data Saturation (FSAT) read-only status bit indicates, when set, that overflow or underflow occurred in the MAC result. FSAT is a sticky status bit, set by hardware and cleared by hardware reset, software reset, or FCOP individual reset. When overflow occurs, the result will be saturated to the most positive number \$7FFF. When underflow occurs, the result will be saturated to the most negative number \$8000.

12.4.6.7 FCOP Data Input Buffer Empty (FDIBE)—FCSR Bit 14

The FCOP Data Input Buffer Empty (FDIBE) read-only status bit indicates, when set, that the FCOP Data Input Buffer (or Register, FDIR) is empty and the DSP can write data to it. FDIBE is set when all four FDIR locations are empty. For proper operation, data should be written to FDIR only if FDIBE is set. Writing to FDIR clears FDIBE. FDIBE is also cleared by hardware, software, or FCOP individual reset. When FCOP is enabled (by setting FEN), FDIBE is set indicating that FDIR is empty. When FDIBE is set and the interrupt is enabled (FDIIE is also set), FCOP generates a Data Input Buffer Empty Interrupt request to the DSP56300 core. A DMA request is always generated when FDIBE is set, but a DMA transfer only takes place if a DMA channel is activated and allocated for FCOP Data Input Buffer Empty (FDIBE is set).

12.4.6.8 FCOP Data Output Buffer Full (FDOBF)—FCSR Bit 15

The FCOP Data Output Buffer Full (FDOBF) read-only status bit indicates, when set, that the Data Output Buffer (FDOR) is full and the DSP can read data from FDOR. FDOBF is set when a result from FMAC is transferred to FDOR. For proper operation, data should be read from FDOR only if FDOBF is set. Reading FDOR clears FDOBF. FDOBF is also cleared by hardware, software, or FCOP individual reset. When FDOBF is set and the interrupt is enabled (FDOIE is also set), FCOP generates a Data Output Buffer Full Interrupt request to the DSP56300 core. A DMA request is always generated when FDOBF is set, but a DMA transfer only takes place if a DMA channel is activated and allocated for FCOP Data Output Buffer Full (FDOBF is set).

12.4.6.9 FCOP Reserved Unused Bits—FCSR Bits 1, 3, 9, 13

These bits are reserved and unused. They read as zeroes and should be written with zero for future compatibility.

12.4.6.10 FCOP Reserved Used Bits—FCSR Bits 2, 6, 7

These bits are reserved for internal testing and debugging. They must be written with zeroes for proper operation.

12.4.7 Interrupts and DMA

The FCOP interrupt vector table is listed in **Table 12-7**.

Table 12-7 FCOP Interrupt Vectors and DMA

Interrupt Address	Interrupt Vector	Priority	Interrupt Enable	Interrupt Conditions	DMA Capability
VBA + Base* + 0	Data Input Register Empty	highest	FDIIE	FDIBE = 1	Yes
VBA + Base* + 2	Data Output Register Full	lowest	FDOIE	FDOBF = 1	Yes
Note: The base address is found in intequ.asm, in Appendix A. It is \$78.					

12.5 OPERATION MODES

The FCOP operation modes are controlled by the FOM[0:1] bits in the FCSR as described in **Section 12.4.6.2**. In each mode, the FDCM bit in FCSR selects between no decimation or decimation by two. The following sections describe the operation of FCOP in each mode, either with no decimation or with decimation by two. The description includes: equation of the implemented filter, initialization and processing steps, data and coefficients input scheme, output data scheme.

12.5.1 Terminology Used in this Section

Compute	Perform all calculations to determine one filter output $F(n)$ for a specific set of input data samples.
Get	Write a data word to the FCOP Data Memory buffer (FDM) via FDIR.
$D(n)$, $DR(n)$:	Real data sample number n .
$DI(n)$	Imaginary data sample number n .
$H(n)$, $HR(n)$	Real filter coefficient number n .
$HI(n)$	Imaginary filter coefficient number n .
$F(n)$, $FR(n)$	Real output result number n .
$FI(n)$	Imaginary output result number n .
#filter_count	Number of coefficient values in the FCOP coefficients Memory bank FCM (equals contents of FCNT register plus 1).

Note: Some items on this list have two forms, for example $D(n)$, $D=R(n)$. In such cases the $D(n)$ form is used when only real values are present, and the $DR(n)$ form is used when complex values are present.

12.5.2 Input DMA Activation

The DMA for input data transfers can be activated only after enabling FCOP (FEN set) and after the core has initialized the coefficients bank through FCIR. Then, the DMA input channel can be enabled in order to start transferring data whenever there are free locations in the input FIFO, while the FCOP state machine grabs data words from that FIFO whenever required. The FCOP state machine starts computation as soon as both coefficient and data banks complete the initialization phase (according to #filter_count value).

A good practice is to program the input data DMA channel for single word transfer or line of 2, 3, or 4 word transfer (since the input buffer FIFO depth is 4), triggered by the FDIBE bit in FCSR.

12.5.3 Output DMA Activation

The DMA for output data transfers should be programmed for single word transfers triggered by the FDOBF bit. The DMA can be activated at any time (even before the FCOP is enabled).

12.5.4 Decimation by 2

FIR decimation by 2 is performed on input, when the FDCM bit is set. When FCOP decimates input, it waits until it gets two input data items (depending on the mode, the data may be real or complex) in FDM and then performs the FIR calculation on the second input data items only. This is equivalent to performing the full FIR filter but reading only every second FIR result, thus saving half the calculations.

12.5.5 FCOP Mode 0: Real FIR Filter

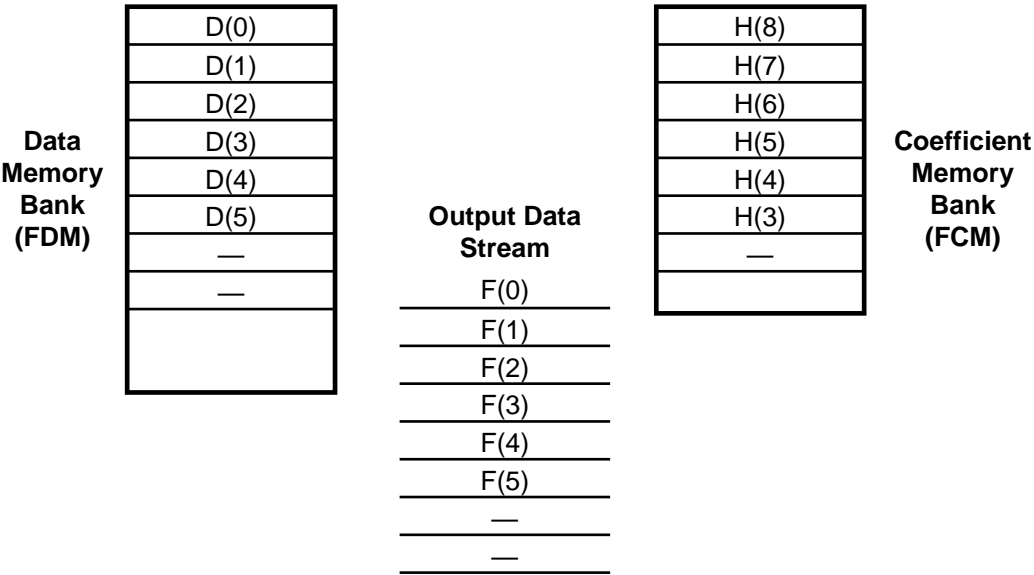
Mode 0 is selected when FOM[1:0] = 00. In this mode, calculation of filter outputs considers data and coefficient words as real numbers.

12.5.5.1 Mode 0 (Real FIR Filter), No Decimation

The following equation is implemented:

$$F(n) = \sum_{i=0}^{N-1} H(i) \cdot D(n-i)$$

Set Up	<ul style="list-style-type: none"> • Load Filter Count Register (FCNT) with (number of coefficient values – 1) • Choose operation mode (FOM[1:0], FDCM=0) and enable FCOP (FEN = 1)
DSP Initialization	<ul style="list-style-type: none"> • Core initializes coefficients in FCM in reverse order by executing #filter_count writes to FCIR • Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR
Processing	<ul style="list-style-type: none"> • Whenever FDIR is empty (FDIBE = 1), FCOP triggers core or DMA to transfer up to four new data words to FDM via FDIR • Compute F(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word • FCOP increments data memory pointer



AA1122

Figure 12-3 Input and Output Stream for Real FIR Filter without Decimation

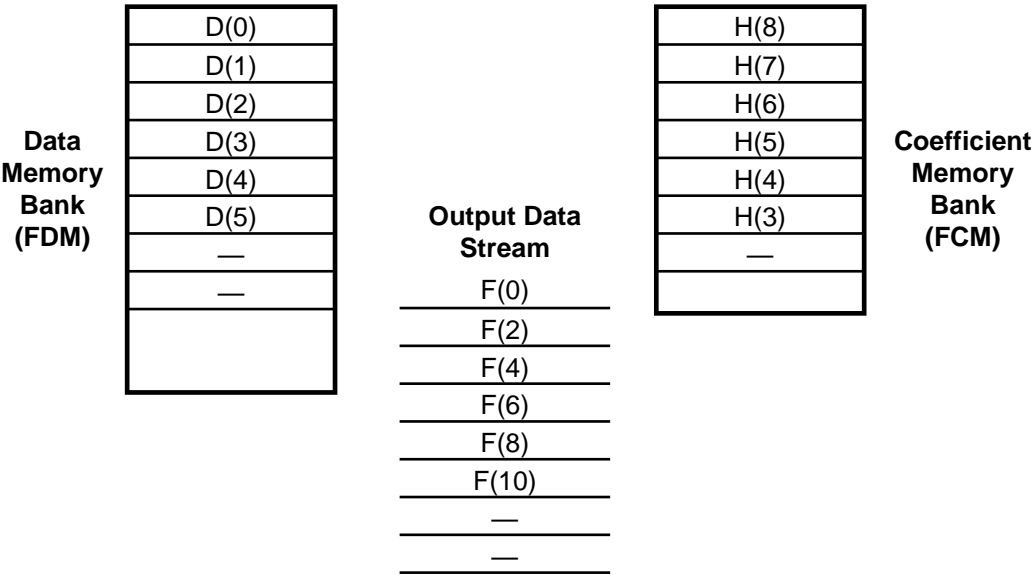
12.5.5.2 Mode 0 (Real FIR Filter), Decimation by 2

The following equation is implemented:

$$F(n|_{even}) = \sum_{i=0}^{N-1} H(i) \cdot D(n-i)$$

The set up and initialization steps are the same as in Mode 0 without decimation. Processing is also the same, except 2 ‘get’ operations are required, so that output results are calculated only for even indexes (see below).

Set Up	<ul style="list-style-type: none"> • Load Filter Count Register (FCNT) with (number of coefficient values – 1) • Choose operation mode (FOM[1:0], FDCM=0) and enable FCOP (FEN = 1)
DSP Initialization	<ul style="list-style-type: none"> • Core initializes coefficients in FCM in reverse order by executing #filter_count writes to FCIR • Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR
Processing	<ul style="list-style-type: none"> • Whenever FDIR is empty (FDIBE = 1), FCOP triggers core or DMA to transfer up to four new data words to FDM via FDIR • Compute F(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word • FCOP increments data memory pointer • Get new data word • FCOP increments data memory pointer



AA1123

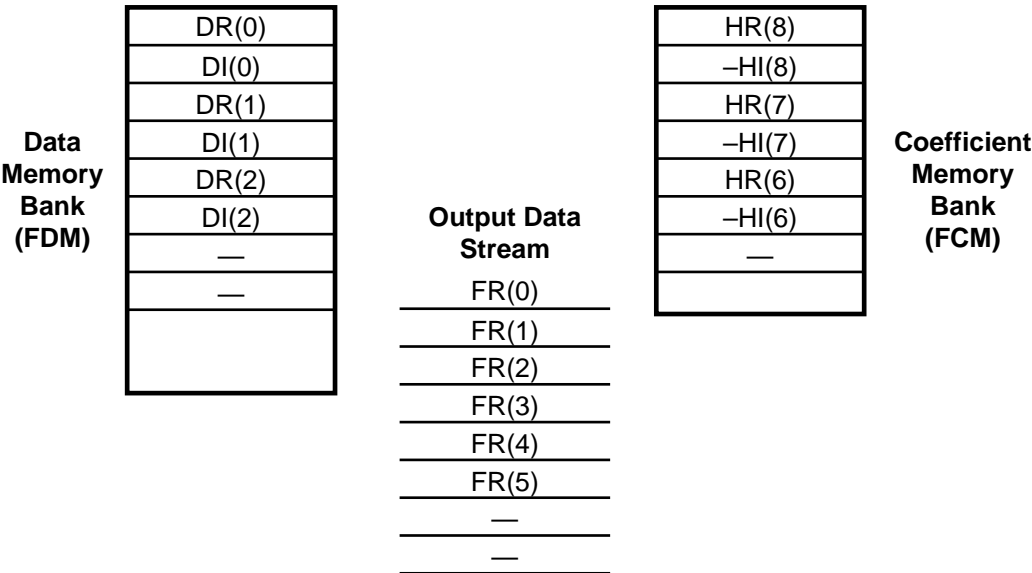
Figure 12-4 Input and Output Stream for Real FIR Filter with Decimation by 2

12.5.5.3 Mode 0 (Complex FIR Filter Generating Real Outputs Only), Decimation by 2

The following equation is implemented:

$$FR(n) = \sum_{i=0}^{N-1} (HR(i) \cdot DR(n-i)) - (HI(i) \cdot DI(n-i))$$

Set Up	<ul style="list-style-type: none"> • Load Filter Count Register (FCNT) with (number of coefficient values – 1) • Choose operation mode (FOM[1:0], FDCM=0, 0, 1) and enable FCOP (FEN = 1)
DSP Initialization	<ul style="list-style-type: none"> • Core initializes coefficients in FCM in reverse order, while imaginary coefficients are first negated, by executing #filter_count writes to FCIR • Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR
Processing	<ul style="list-style-type: none"> • Whenever FDIR is empty (FDIBE = 1), the FCOP triggers core or the DMA to transfer two or four new data words (one or two complex pairs) to the FDM via FDIR • Compute F(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word • FCOP increments data memory pointer • Get new data word • FCOP increments data memory pointer



AA1124

Figure 12-5 Input and Output Stream for Complex FIR Filter Generating Real Outputs Only with Decimation by 2

12.5.6 FCOP Mode 1: Full Complex FIR Filter

Mode 1 is selected when FOM[1:0] = 01. In this mode, for each complex input, a complex output is generated.

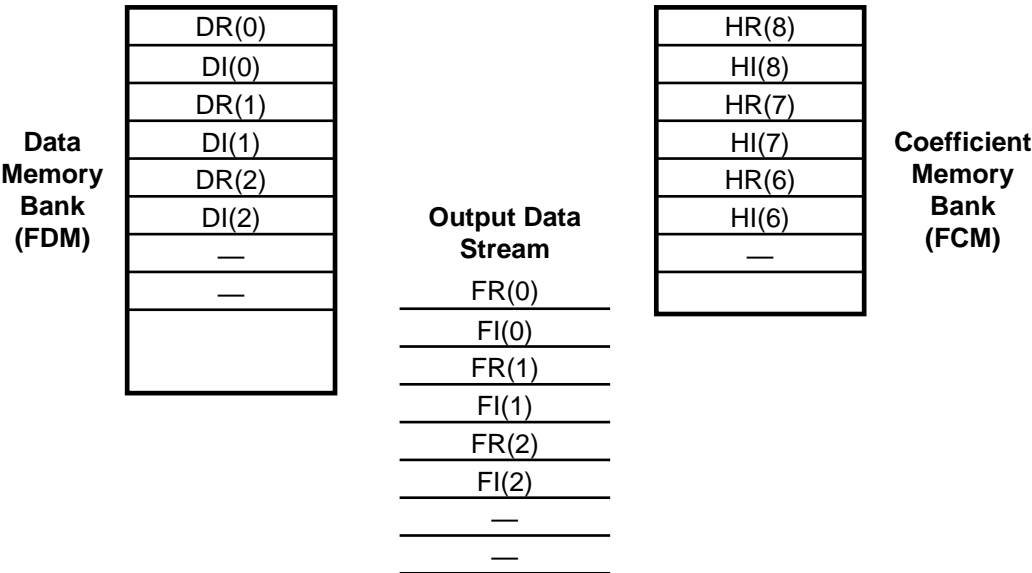
12.5.6.1 Mode 1(Full Complex FIR Filter), No Decimation

The following equations are implemented:

$$FR(n) = \sum_{i=0}^{N-1} (HR(i) \cdot DR(n-i)) - (HI(i) \cdot DI(n-i))$$

$$FI(n) = \sum_{i=0}^{N-1} (HR(i) \cdot DI(n-i)) + (HI(i) \cdot DR(n-i))$$

Set Up	<ul style="list-style-type: none"> • Load Filter Count Register (FCNT) with (number of coefficient values – 1) • Choose operation mode (FOM[1:0], FDCM = 0, 1, 0) and enable FCOP (FEN = 1)
DSP Initialization	<ul style="list-style-type: none"> • Core initializes coefficients in FCM in reverse order by executing #filter_count writes to FCIR • Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR
Processing	<ul style="list-style-type: none"> • Whenever FDIR is empty (FDIBE = 1), the FCOP triggers core or the DMA to transfer two or four new data words (one or two complex pairs) to the FDM via FDIR • Compute FR(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Compute FI(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer



AA1125

Figure 12-6 Input and Output Stream for Full Complex FIR Filter without Decimation

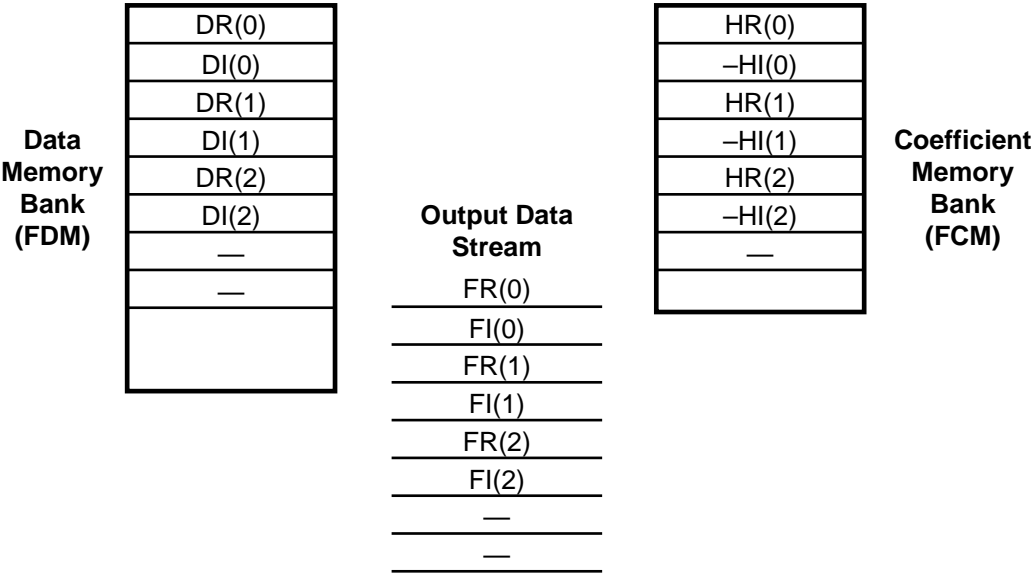
12.5.6.2 Mode 1 (Full Complex Correlation Filter), No Decimation

The following equations are implemented:

$$FR(n) = \sum_{i=0}^{N-1} (HR(i) \cdot DR(n+i)) + (HI(i) \cdot DI(n+i))$$

$$FI(n) = \sum_{i=0}^{N-1} (HR(i) \cdot DI(n+i)) - (HI(i) \cdot DR(n+i))$$

Set Up	<ul style="list-style-type: none"> • Load Filter Count Register (FCNT) with (number of coefficient values – 1) • Choose operation mode (FOM[1:0], FDCM = 0, 1, 0) and enable FCOP (FEN = 1)
DSP Initialization	<ul style="list-style-type: none"> • Core initializes coefficients in FCM in direct order, while imaginary coefficients are first negated, by executing #filter_count writes to FCIR • Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR
Processing	<ul style="list-style-type: none"> • Whenever FDIR is empty (FDIBE = 1), the FCOP triggers core or the DMA to transfer two or four new data words (one or two complex pairs) to the FDM via FDIR • Compute FR(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Compute FI(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer



AA1126

Figure 12-7 Input and Output Stream for Full Complex Correlation Filter without Decimation

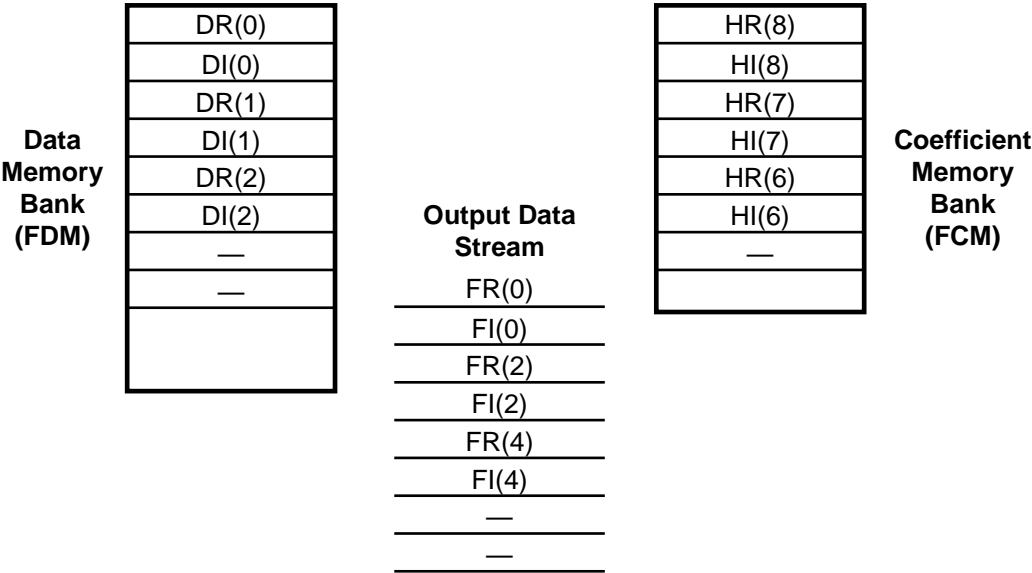
12.5.6.3 Mode 1 (Full Complex FIR Filter), Decimation by 2

The following equations are implemented:

$$FR(n|_{even}) = \sum_{i=0}^{N-1} (HR(i) \cdot DR(n-i)) - (HI(i) \cdot DI(n-i))$$

$$FI(n|_{even}) = \sum_{i=0}^{N-1} (HR(i) \cdot DI(n-i)) + (HI(i) \cdot DR(n-i))$$

Set Up	<ul style="list-style-type: none"> • Load Filter Count Register (FCNT) with (number of coefficient values – 1) • Choose operation mode (FOM[1:0], FDCM = 0, 1, 1) and enable FCOP (FEN = 1)
DSP Initialization	<ul style="list-style-type: none"> • Core initializes coefficients in FCM in reverse order, by executing #filter_count writes to FCIR • Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR
Processing	<ul style="list-style-type: none"> • Whenever FDIR is empty (FDIBE = 1), the FCOP triggers core or the DMA to transfer two or four new data words (one or two complex pairs) to the FDM via FDIR • Compute FR(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Compute FI(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer



AA1127

Figure 12-8 Input and Output Stream for Full Complex Filter with Decimation

12.5.7 FCOP Mode 2: Full Complex FIR Filter

Mode 2 is selected when FOM[1:0] = 10. Each complex input consists of two filter outputs: the first real, the second imaginary, generated alternately.

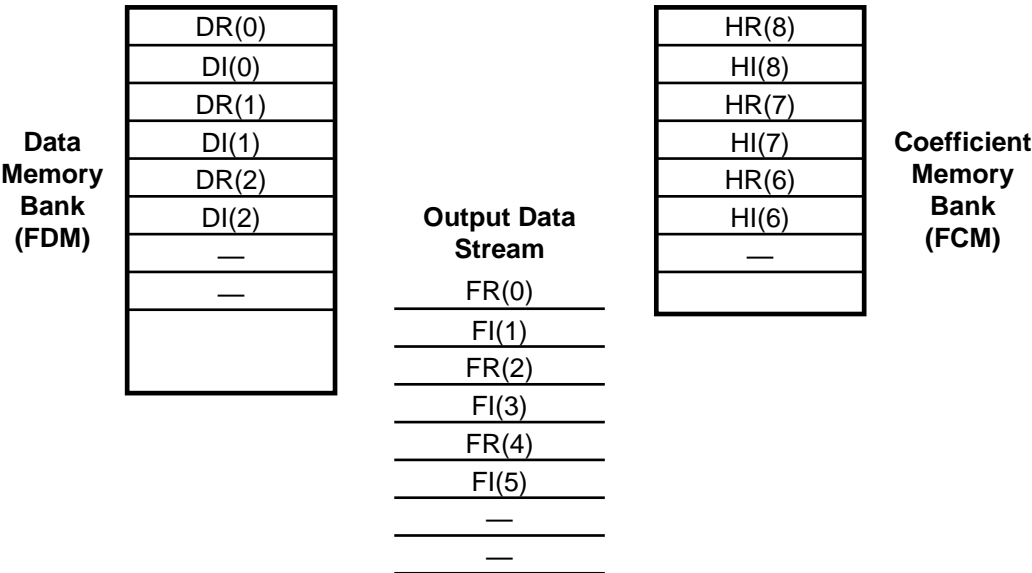
12.5.7.1 Mode 2 (Complex FIR Filter Generating Pure Real or Pure Imaginary Outputs Alternately), No Decimation

The following equations are implemented:

$$FR(n|_{even}) = \sum_{i=0}^{N-1} (HR(i) \cdot DR(n-i)) - (HI(i) \cdot DI(n-i))$$

$$FI(n|_{odd}) = \sum_{i=0}^{N-1} (HR(i) \cdot DI(n-i)) + (HI(i) \cdot DR(n-i))$$

Set Up	<ul style="list-style-type: none"> • Load Filter Count Register (FCNT) with (number of coefficient values – 1) • Choose operation mode (FOM[1:0], FDCM = 1, 0, 0) and enable FCOP (FEN = 1)
DSP Initialization	<ul style="list-style-type: none"> • Core initializes coefficients in FCM in reverse order by executing #filter_count writes to FCIR • Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR
Processing	<ul style="list-style-type: none"> • Whenever FDIR is empty (FDIBE = 1), the FCOP triggers core or the DMA to transfer two or four new data words (one or two complex pairs) to the FDM via FDIR • Compute FR(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer • Compute FI(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer



AA1128

Figure 12-9 Input and Output Stream for Complex FIR Filter Generating Pure Real or Pure Imaginary Outputs Alternately without Decimation

12.5.7.2 Mode 2 (Complex FIR Filter Generating Pure Real and Pure Imaginary Outputs Alternately), Decimation by 2

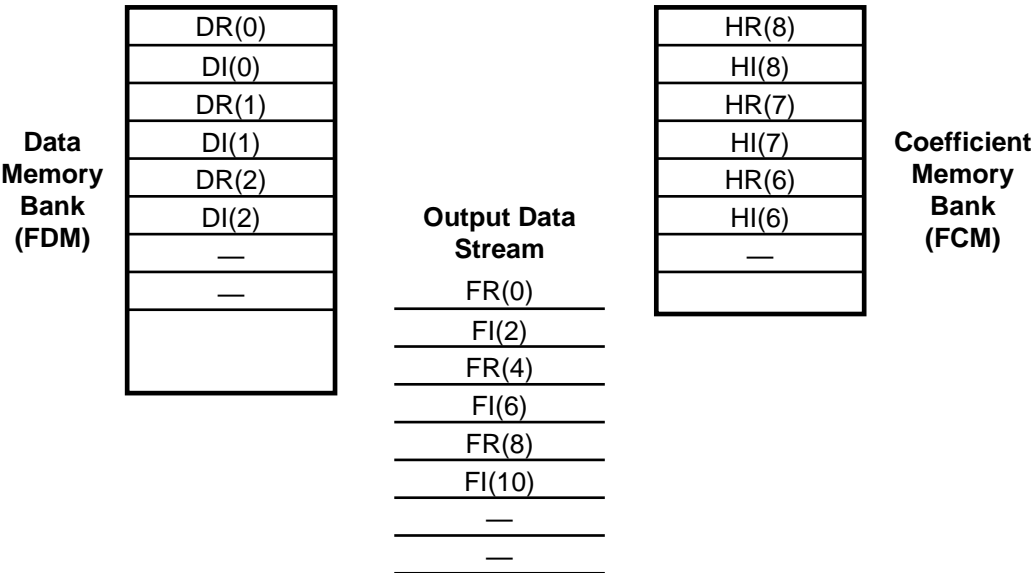
The following equations are implemented:

$$FR(n|_{0,4,8,etc}) = \sum_{i=0}^{N-1} (HR(i) \cdot DR(n-i)) - (HI(i) \cdot DI(n-i))$$

$$FI(n|_{2,6,10,etc}) = \sum_{i=0}^{N-1} (HR(i) \cdot DI(n-i)) + (HI(i) \cdot DR(n-i))$$

Operation Modes

Set Up	<ul style="list-style-type: none"> • Load Filter Count Register (FCNT) with (number of coefficient values – 1) • Choose operation mode (FOM[1:0], FDCM = 1, 0, 1) and enable FCOP (FEN = 1)
DSP Initialization	<ul style="list-style-type: none"> • Core initializes coefficients in FCM in reverse order by executing #filter_count writes to FCIR • Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR
Processing	<ul style="list-style-type: none"> • Whenever FDIR is empty (FDIBE = 1), the FCOP triggers core or the DMA to transfer two or four new data words (one or two complex pairs) to the FDM via FDIR • Compute FR(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer • Compute FI(n) and store result in FDOR • FCOP triggers core or DMA for output data transfer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer • Get new data word (DR) • FCOP increments data memory pointer • Get new data word (DI) • FCOP increments data memory pointer



AA1129

Figure 12-10 Input and Output Stream for Complex FIR Filter Generating Pure Real and Pure Imaginary Outputs Alternately with Decimation by 2

12.5.8 FCOP Mode 3: Optimized Complex Correlation Function

Mode 3 is selected when FOM[1:0] = 11. Mode 3 is capable of performing complex correlation between a complex data sequence and a pure real/imaginary sequence. In GSM, it is used to correlate between the received training sequence and a known midamble sequence.

The received training sequence can be non-oversampled (one pair of in-phase and quadrature (I&Q) samples per bit), or 2× oversampled (2 pair of I&Q samples per bit). The midamble sequence consists of alternate pure real/pure imaginary values.

The basic correlation function is:

$$FR(n) = \sum_{i=0}^{N-1} (HR(i) \cdot DR(n+i)) + (HI(i) \cdot DI(n+i))$$

$$FI(n) = \sum_{i=0}^{N-1} (HR(i) \cdot DI(n+i)) - (HI(i) \cdot DR(n+i))$$

However, since the midamble has zero components, the actual calculations are simpler, as shown in the following sections.

12.5.8.1 Mode 3 (Complex Correlation of Non-Oversampled Data), No Decimation

The received training sequence is complex (one pair of I&Q samples per bit). The midamble sequence consists of alternate pure real/pure imaginary values (one pure complex value per bit). Refer to the following table:

Table 12-8 Non-Oversampled Data Sequence

Sample (index)	Bit #	Received Input	Received Quadrature	Midamble Input	Midamble Quadrature
0	0	DR(0)	DI(0)	HR(0)	0
1	1	DR(1)	DI(1)	0	HI(1)
2	2	DR(2)	DI(2)	HR(2)	0
3	3	DR(3)	DI(3)	0	HI(3)
4	4	DR(4)	DI(4)	HR(4)	0
5	5	DR(5)	DI(5)	0	HI(5)
6	6	DR(6)	DI(6)	HR(6)	0
7	7	DR(7)	DI(7)	0	HI(7)

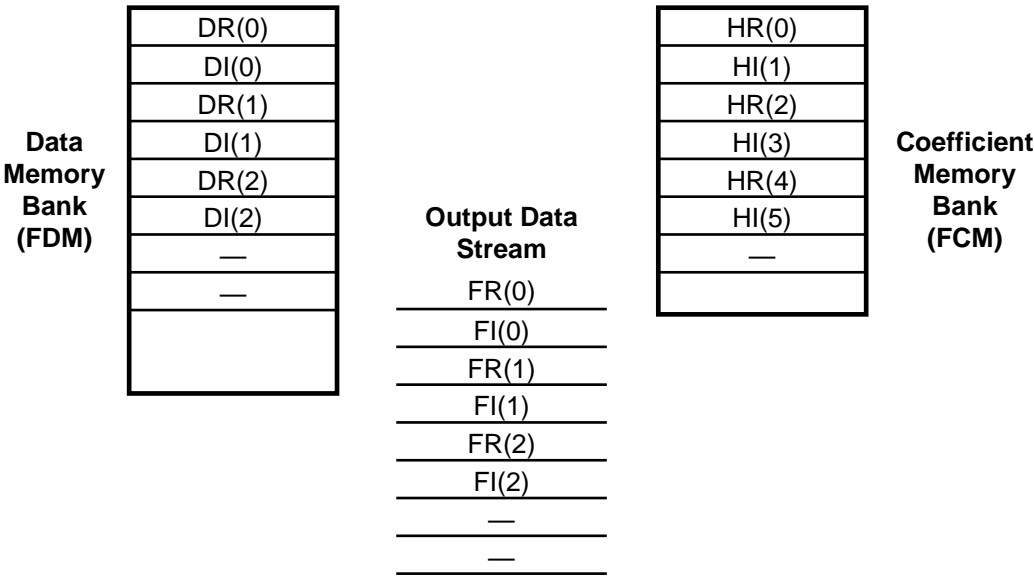
Taking advantage of the “zero” components of the midamble sequence, we get the following equations from the basic correlation equations, so that a complex output is calculated for each complex input, requiring half the MAC operations:

$$FR(n) = \sum_{i=0}^{N/2-1} (HR(2i) \cdot DR(n+2i)) + (HI(2i+1) \cdot DI(n+2i+1))$$

$$FI(n) = \sum_{i=0}^{N/2-1} (HR(2i) \cdot DI(n+2i)) - (HI(2i+1) \cdot DR(n+2i+1))$$

Operation Modes

Set Up	<ul style="list-style-type: none">• Load Filter Count Register (FCNT) with (number of coefficient values – 1).• Choose operation mode (FOM[1:0], FDCM = 1, 1, 0) and enable FCOP (FEN = 1).
DSP Initialization	<ul style="list-style-type: none">• Core initializes coefficients in FCM in direct order by executing #filter_count writes to FCIR.• Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR.
Processing	<ul style="list-style-type: none">• Whenever FDIR is empty (FDIBE = 1), the FCOP triggers core or the DMA to transfer two or four new data words (one or two complex pairs) to the FDM via FDIR.• Compute FR(n) and store result in FDOR.• FCOP triggers core or DMA for output data transfer.• Compute FI(n) and store result in FDOR.• FCOP triggers core or DMA for output data transfer.• Get new data word (DR).• FCOP increments data memory pointer.• Get new data word (DI).• FCOP increments data memory pointer.



AA1130

Figure 12-11 Input and Output Stream for Complex Correlation of Non-Oversampled Data without Decimation

12.5.8.2 Mode 3 (Complex Correlation of 2× Oversampled Data), No Decimation

The received training sequence is 2× oversampled (two pairs of I&Q samples per bit). The midamble sequence consists of alternate pure real/pure imaginary values (one pure complex value per bit). Before correlation, the midamble should be interpolated by 2. Refer to the following table:

Table 12-9 2 × Oversampled Data Sequence

Sample (index)	Bit #	Received Input	Received Quadrature	Midamble Input	Midamble Quadrature
0	0	DR(0)	DI(0)	HR(0)	0
1		DR(1)	DI(1)	0	0
2	1	DR(2)	DI(2)	0	HI(2)
3		DR(3)	DI(3)	0	0
4	2	DR(4)	DI(4)	HR(4)	0
5		DR(5)	DI(5)	0	0
6	3	DR(6)	DI(6)	0	HI(6)
7		DR(7)	DI(7)	0	0

Taking advantage of the “zero” components of the interpolated midamble sequence, we get the following equations from the basic correlation equations:

$$FR(n) = \sum_{i=0}^{N/4-1} (HR(4i) \cdot DR(n+4i)) + (HI(4i+2) \cdot DI(n+4i+2))$$

$$FI(n) = \sum_{i=0}^{N/4-1} (HR(4i) \cdot DI(n+4i)) - (HI(4i+2) \cdot DR(n+4i+2))$$

When n is even, the filter outputs are independent of the odd input samples, and when n is odd, the filter outputs are independent of the even input samples. As a result, even and odd outputs can be calculated separately, requiring half of the data memory bank size.

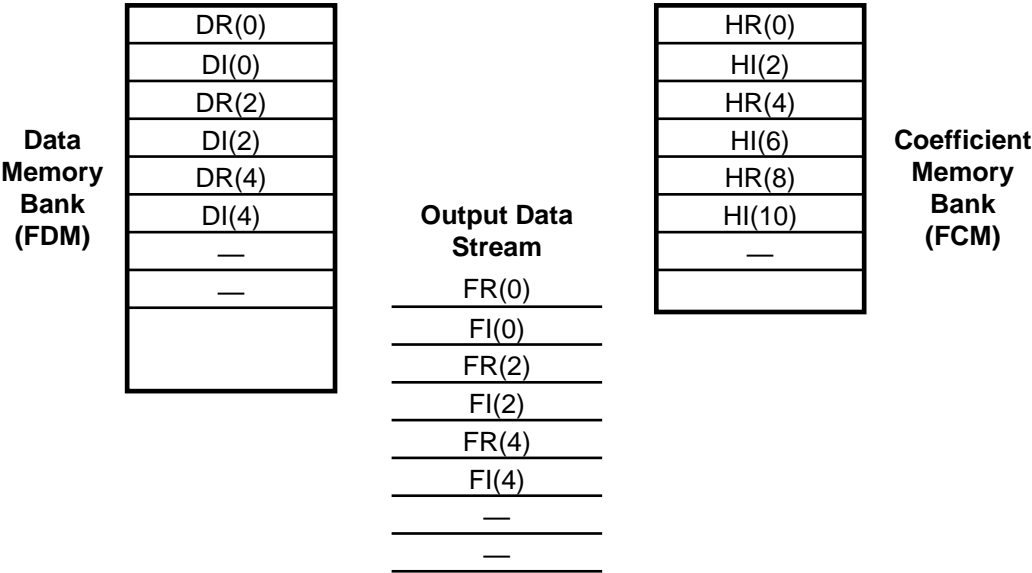
Operation Modes

For calculating the even outputs, the following equations are implemented:

$$FR(n|_{even}) = \sum_{i=0}^{N/4-1} (HR(4i) \cdot DR(n+4i)) + (HI(4i+2) \cdot DI(n+4i+2))$$

$$FI(n|_{even}) = \sum_{i=0}^{N/4-1} (HR(4i) \cdot DI(n+4i)) - (HI(4i+2) \cdot DR(n+4i+2))$$

Set Up	<ul style="list-style-type: none"> • Load Filter Count Register (FCNT) with (number of coefficient values – 1) • Choose operation mode (FOM[1:0], FDCM = 1, 1, 0) and enable FCOP (FEN = 1)
DSP Initialization	<ul style="list-style-type: none"> • Core initializes coefficients in FCM in direct order by executing #filter_count writes to FCIR. • Core or DMA initializes data in FDM in direct order by executing #filter_count writes to FDIR.
Processing	<ul style="list-style-type: none"> • Whenever FDIR is empty (FDIBE = 1), the FCOP triggers core or the DMA to transfer two or four new data words (one or two complex pairs) to the FDM via FDIR. • Compute FR(n) and store result in FDOR. • FCOP triggers core or DMA for output data transfer. • Compute FI(n) and store result in FDOR. • FCOP triggers core or DMA for output data transfer. • Get new data word (DR). • FCOP increments data memory pointer. • Get new data word (DI). • FCOP increments data memory pointer.



AA1131

Figure 12-12 Input and Output Stream for Complex Correlation of 2× Oversampled Data without Decimation

12.6 PERFORMANCE ANALYSIS

FCOP cycle count calculations are based on the following expressions. **Table 12-10** is a summary of FCOP cycle counts in a GSM base station.

Cycle count for match filter of Non-Oversampled data (Mode 2, FDCM = 0):

$$(2 \cdot Fl + 6) + (4 \cdot Fl) + N \cdot (Fl + 2) = (N + 6) \cdot (Fl + 2) - 6$$

Cycle count for match filter of 2x Oversampled data (Mode 2, FDCM = 1):

$$(2 \cdot Fl + 6) + (4 \cdot Fl) + N \cdot (Fl + 4) = (N + 6) \cdot (Fl + 4) - 18$$

Cycle count for optimized correlation (Mode 3, FDCM = 0):

$$(2 \cdot Fl + 6) + (8 \cdot Fl) + N \cdot (Fl + 1) = (N + 10) \cdot (Fl + 1) - 4$$

Where:

- *Fl* is the filter length (the number of values in the coefficient memory)
- *N* is the number of output values generated by FCOP

Table 12-10 FCOP Cycle Count in GSM Base Station

Over-sampled Data	Burst Type	Filter Mode	Data Length	Filter Length (<i>Fl</i>)	No. of Outputs (<i>N</i>)	No. of Cycles	Duration @80MHz
No	Normal	Match Filter	61 x 2	9 x 2	61	1340	16.75 μs
No	Normal	Correlation	26 x 2	26	52 x 2	3078	38.5 μs
No	Access	Match Filter	36 x 2	9 x 2	36	840	10.5 μs
No	Access	Correlation	41 x 2	41	82 x 2	7308	91.5 μs
2 x	Normal	Match Filter	61 x 2	9 x 2	61	1474	18.5 μs
2 x	Normal	Correlation	26 x 2	26	52 x 2	3078	38.5 μs
2 x	Access	Match Filter	36 x 2	9 x 2	36	924	11.6 μs
2 x	Access	Correlation	41 x 2	41	82 x 2	7308	91.5 μs

