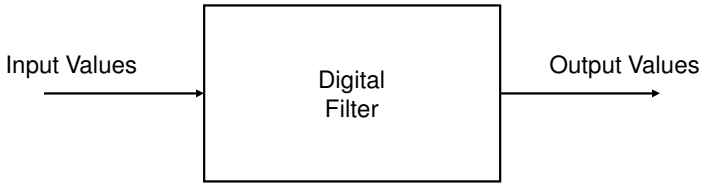
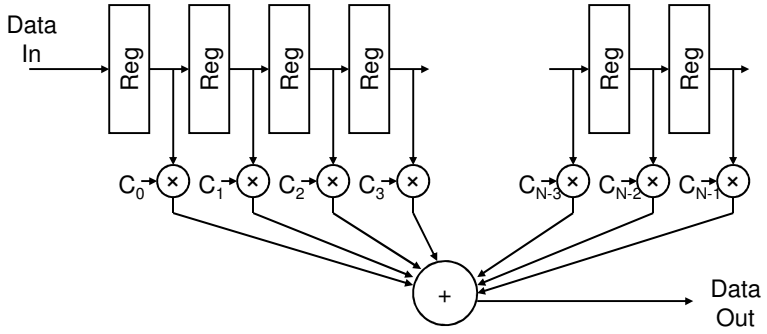
Digital Filter – Digital Domain

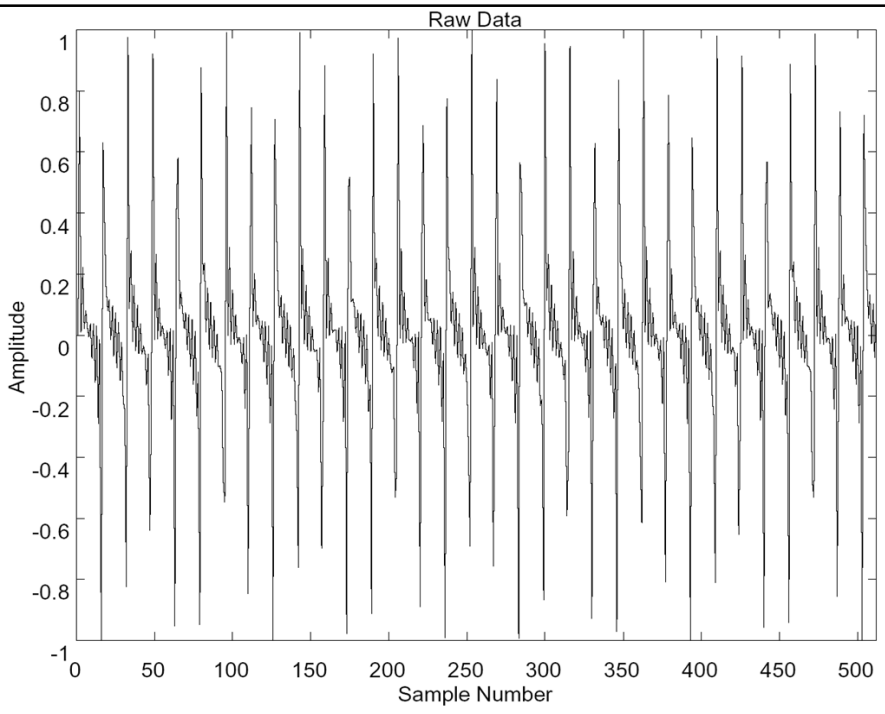


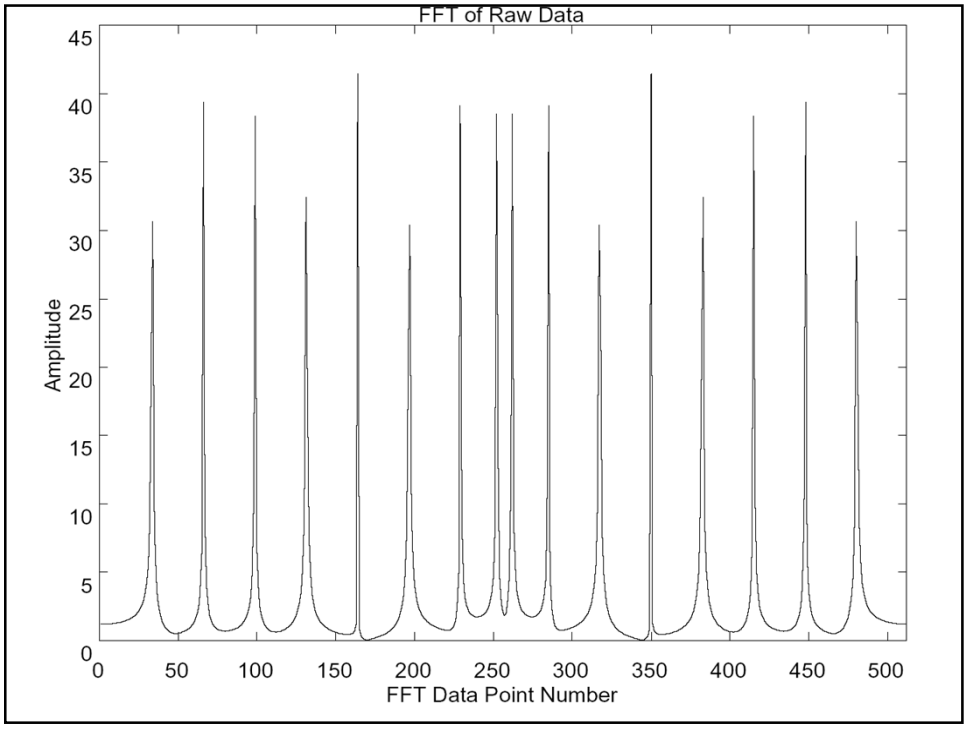
FIR System

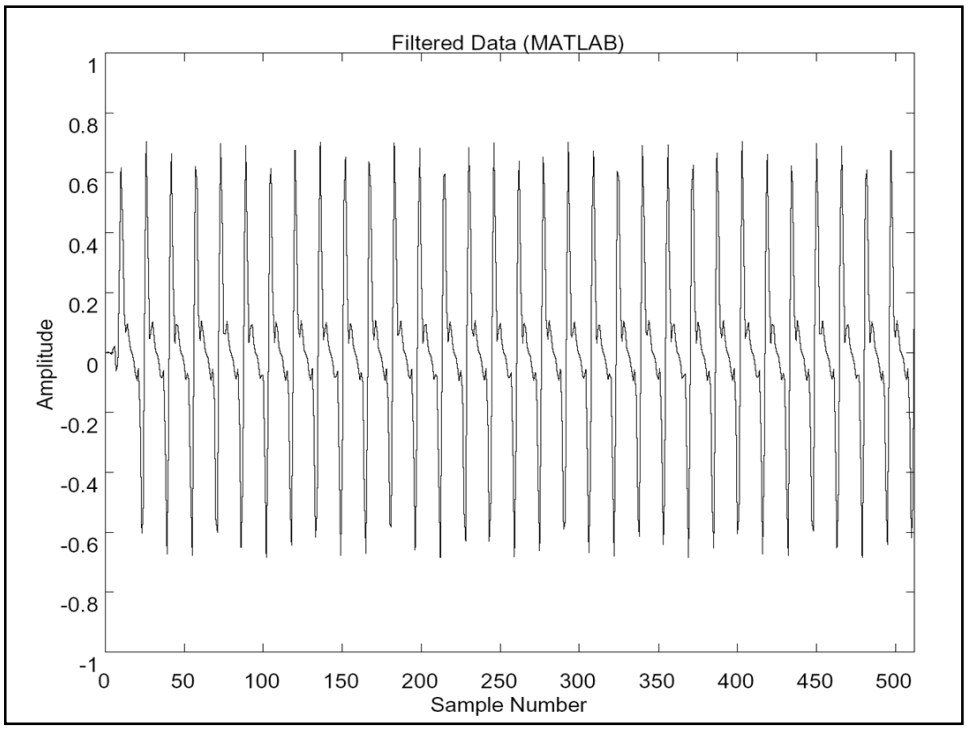
$$Output = \sum_{k=0}^{N-1} S_k \times C_k$$

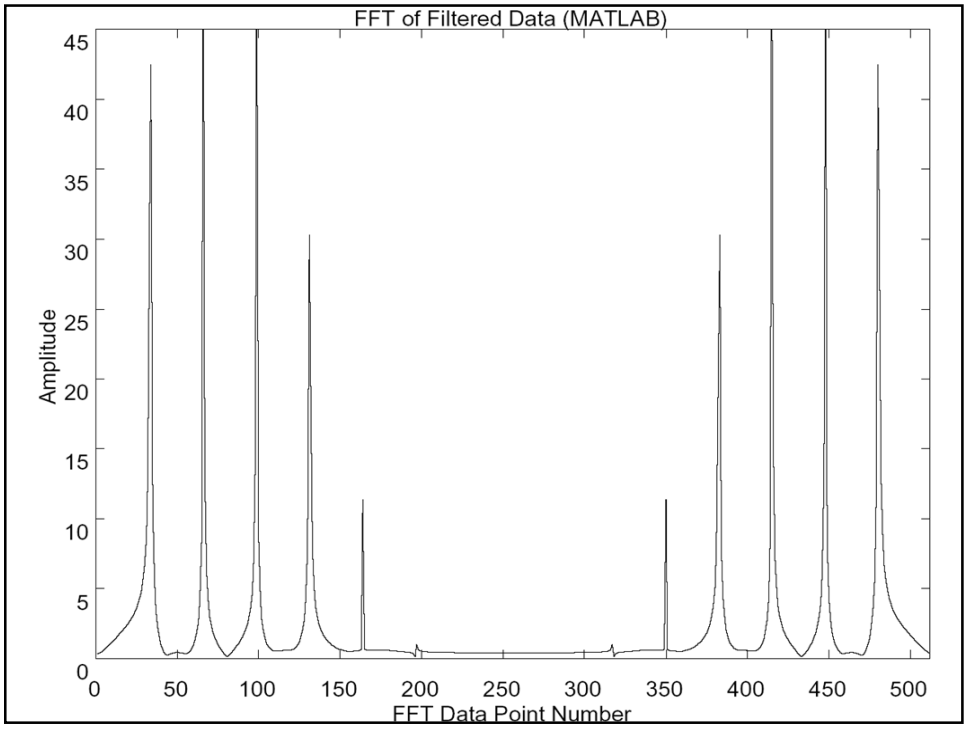
Algorithm Implementation

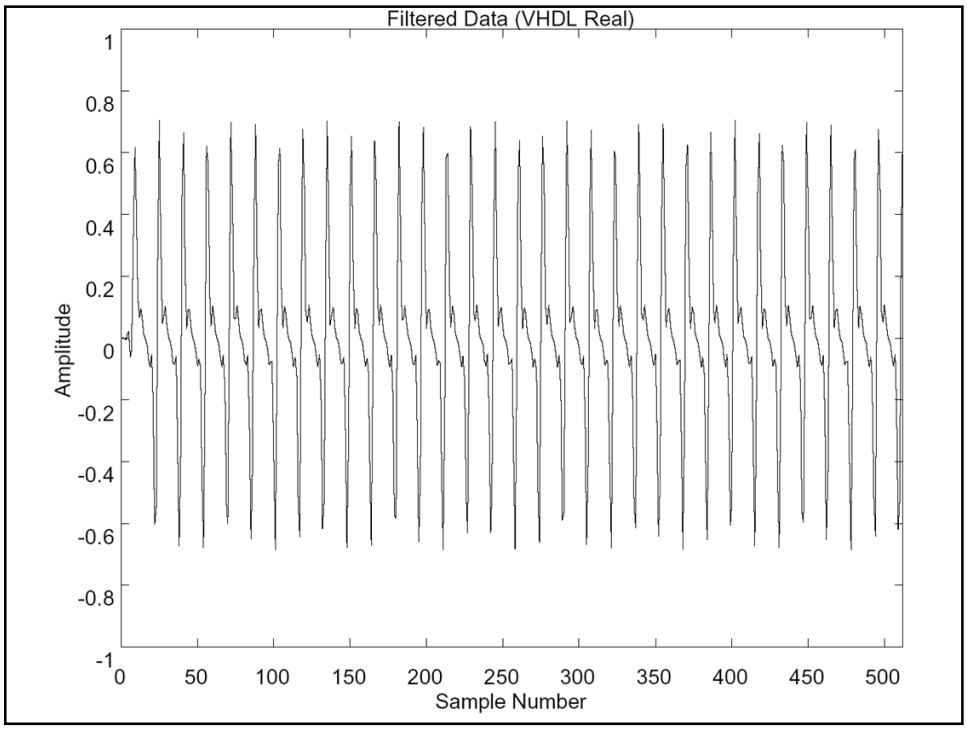


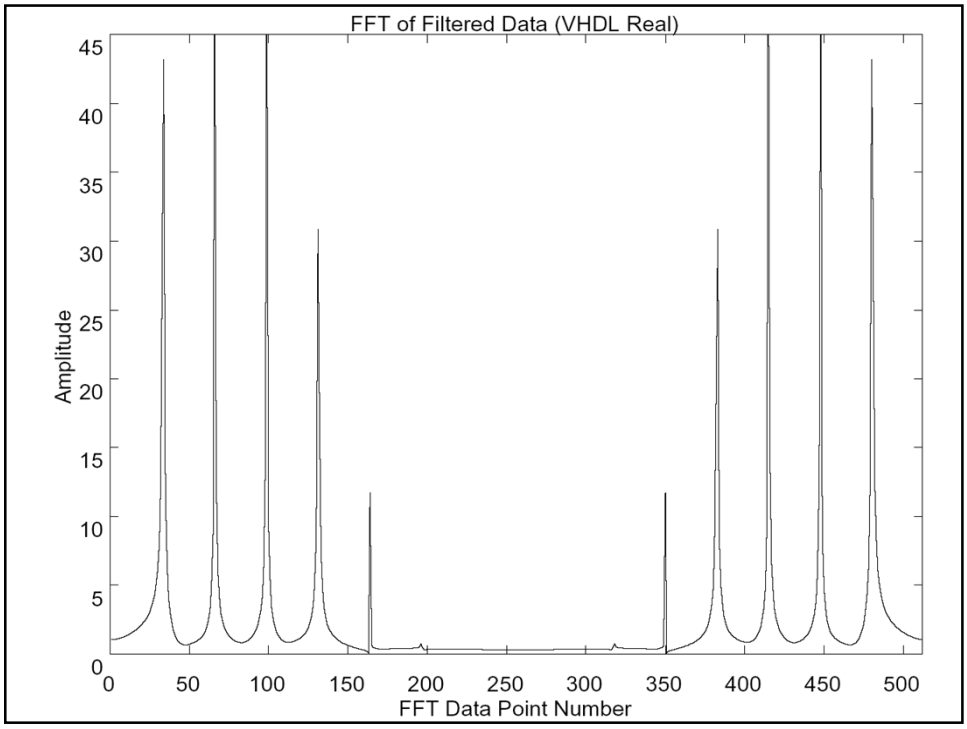


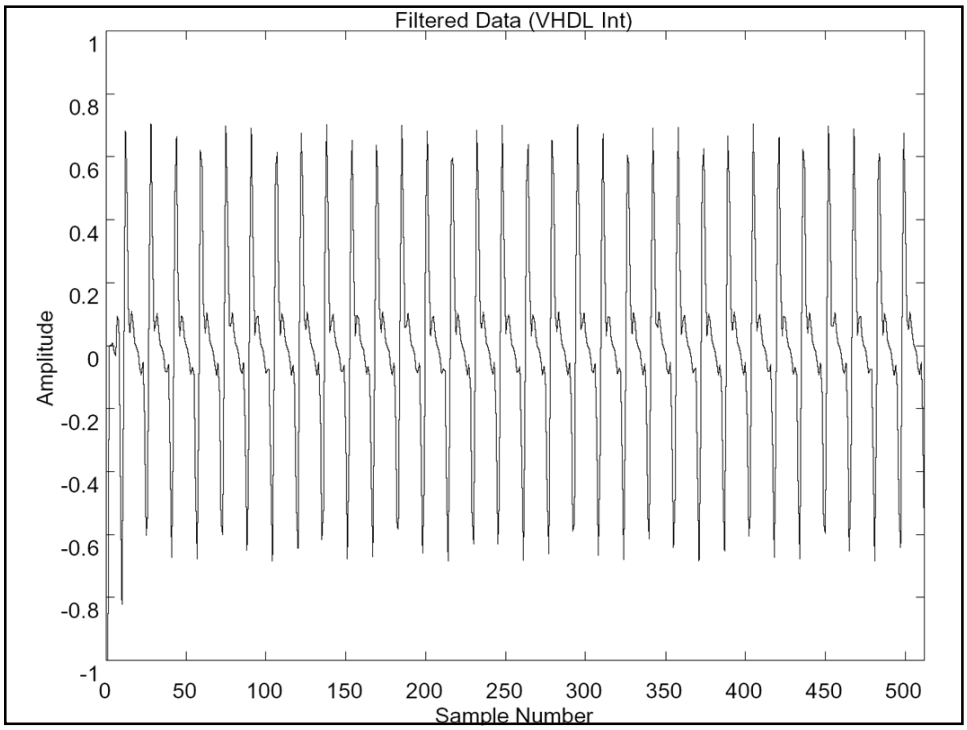

FFT of Raw Data
Amplitude
FFT Data Point Number
0
5
10
15
20
25
30
35
40
45
0
50
100
150
200
250
300
350
400
450
500

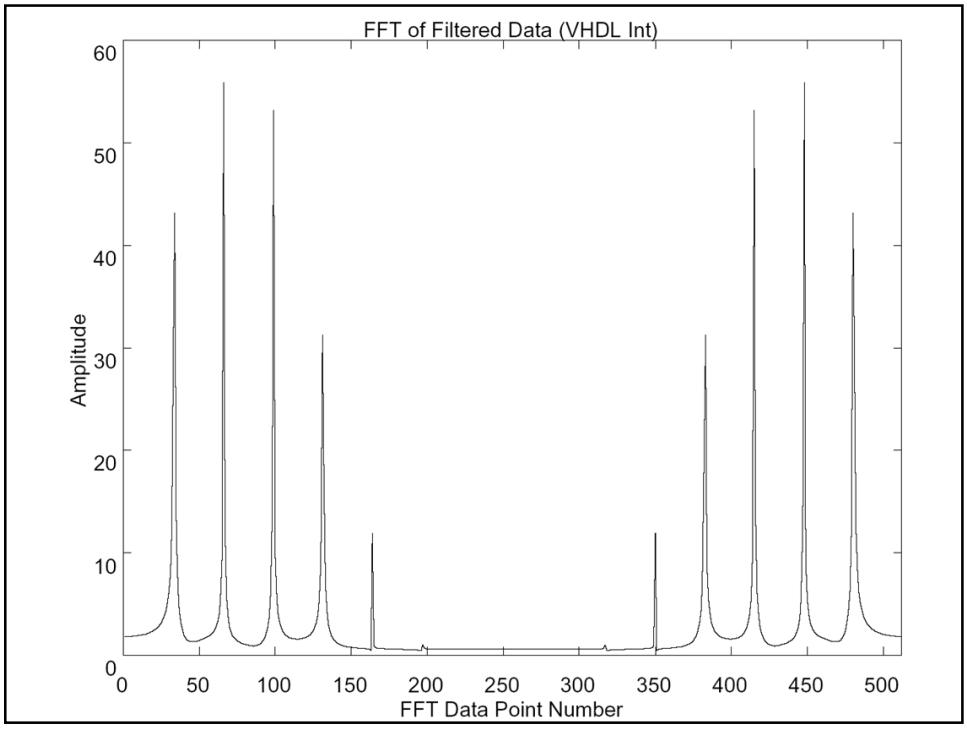

Filtered Data (MATLAB)
Amplitude
Sample Number
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
50
100
150
200
250
300
350
400
450
500

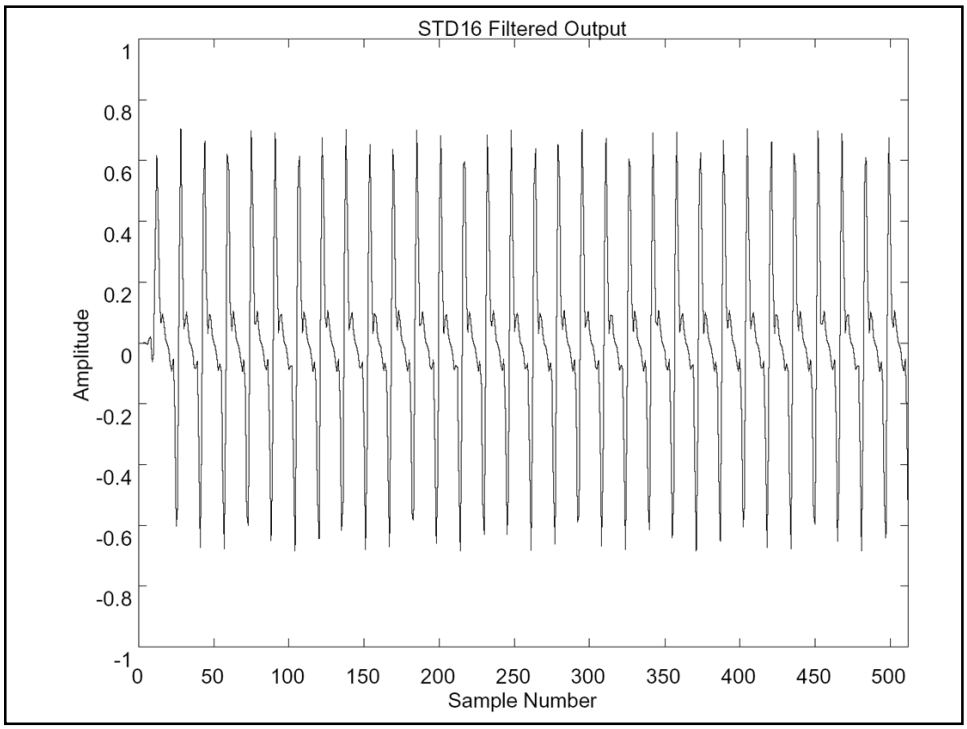

FFT of Filtered Data (MATLAB)
Amplitude
FFT Data Point Number
0
5
10
15
20
25
30
35
40
45
0
50
100
150
200
250
300
350
400
450
500

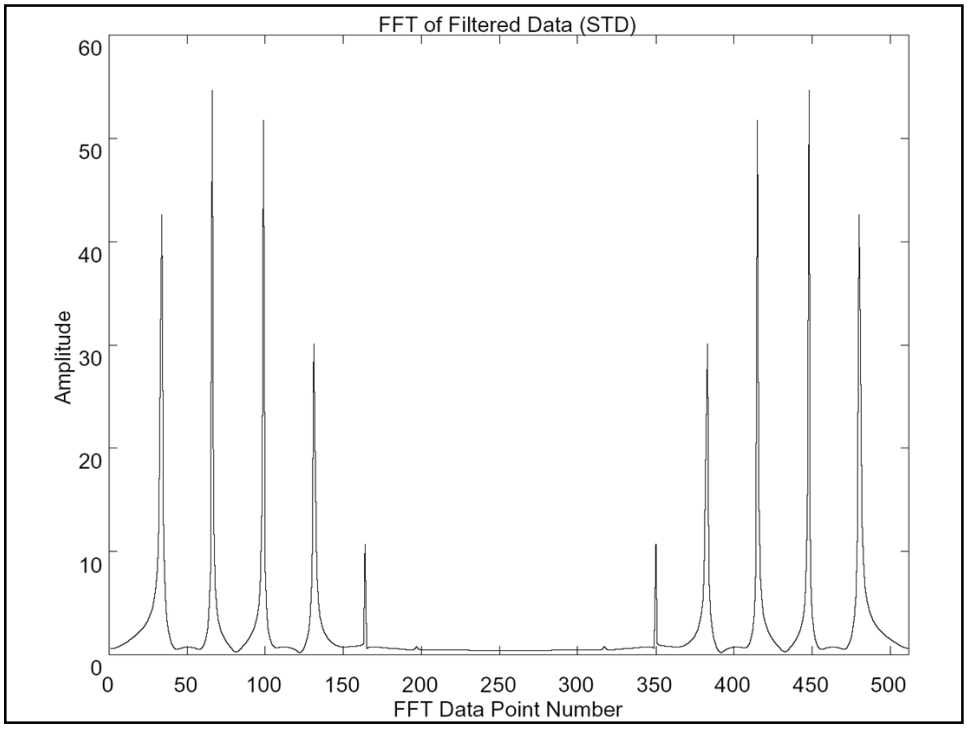

Filtered Data (VHDL Real)
Amplitude
Sample Number
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
50
100
150
200
250
300
350
400
450
500

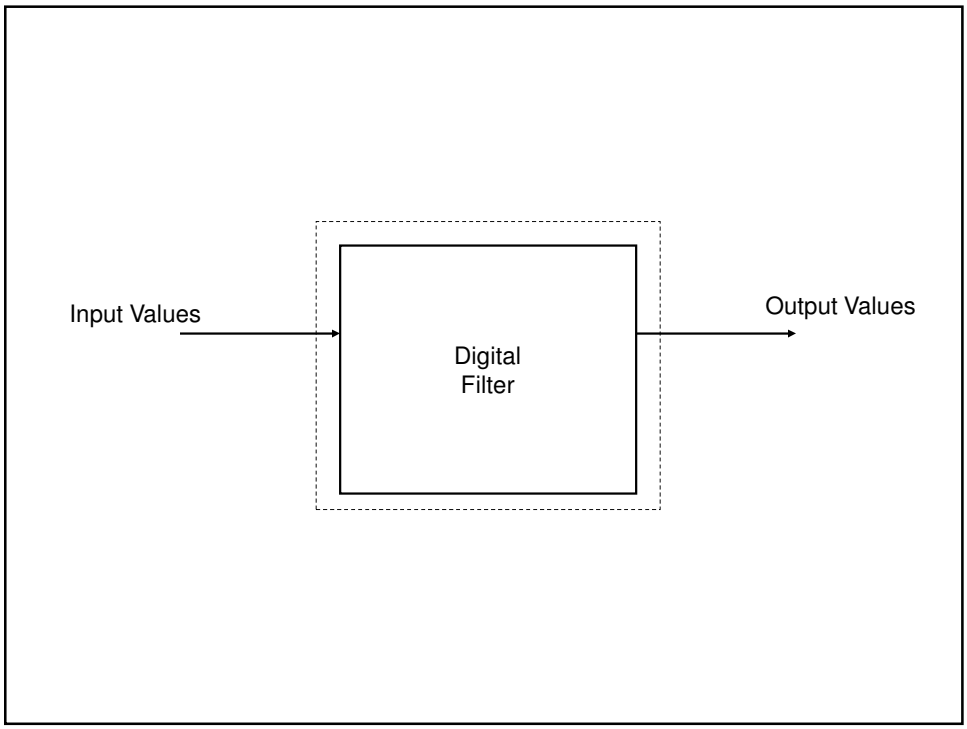

FFT of Filtered Data (VHDL Real)
Amplitude
FFT Data Point Number
0
5
10
15
20
25
30
35
40
45
0
50
100
150
200
250
300
350
400
450
500

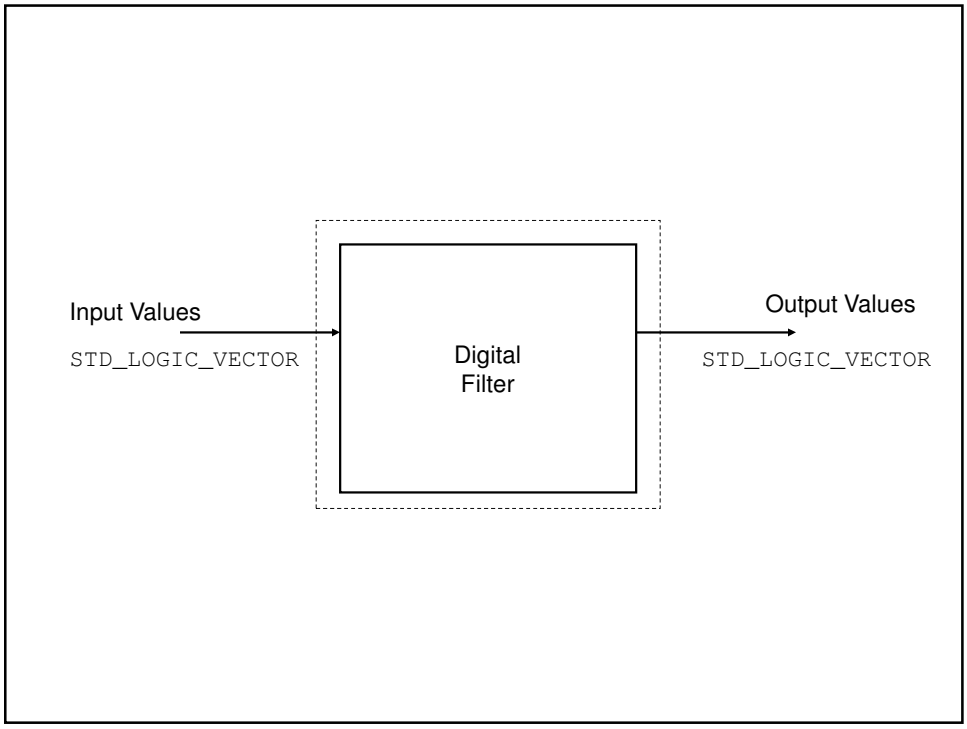

Filtered Data (VHDL Int)
Amplitude
Sample Number
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
50
100
150
200
250
300
350
400
450
500

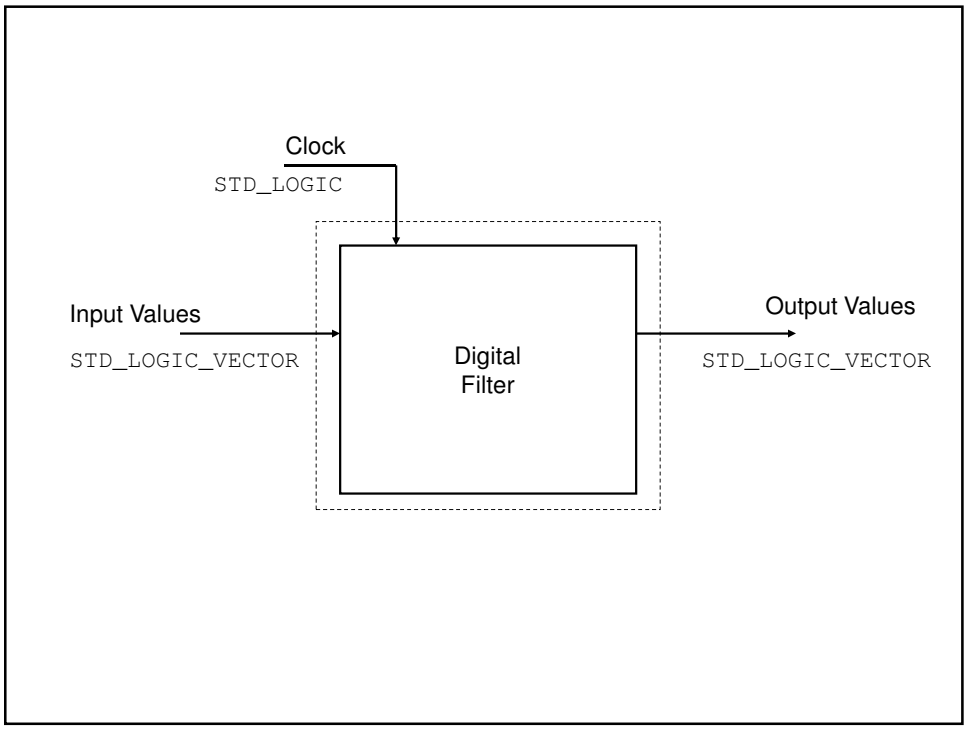

FFT of Filtered Data (VHDL Int)
Amplitude
FFT Data Point Number
0
10
20
30
40
50
60
0
50
100
150
200
250
300
350
400
450
500

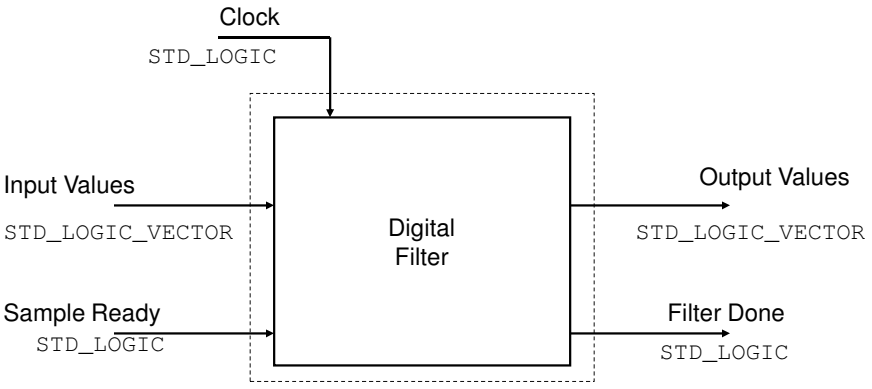

STD16 Filtered Output
Amplitude
Sample Number
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
50
100
150
200
250
300
350
400
450
500


FFT of Filtered Data (STD)
Amplitude
FFT Data Point Number
0
10
20
30
40
50
60
0
50
100
150
200
250
300
350
400
450
500

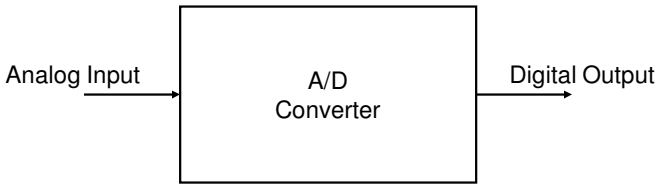

Input Values
Digital
Filter
Output Values


Input Values
STD_LOGIC_VECTOR
Digital
Filter
Output Values
STD_LOGIC_VECTOR

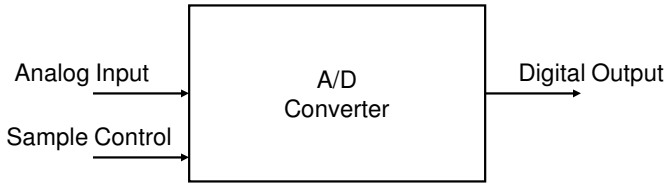

Clock
STD_LOGIC
Input Values
STD_LOGIC_VECTOR
Digital
Filter
Output Values
STD_LOGIC_VECTOR



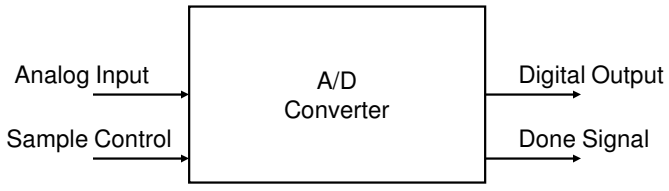
A/D Converter Behavior



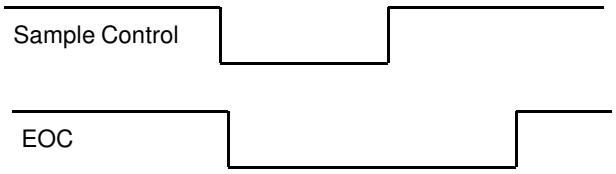
A/D Converter Behavior



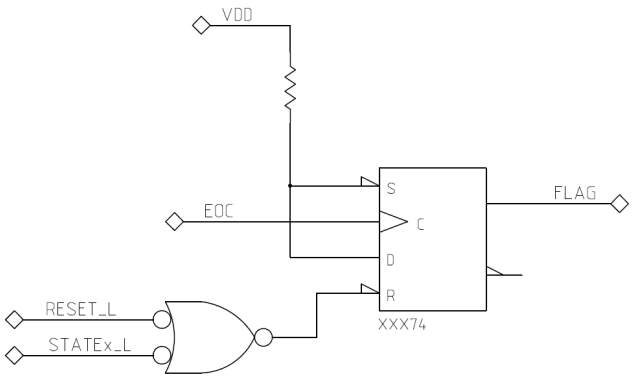
A/D Converter Behavior



Control Waveforms – A/D Converter



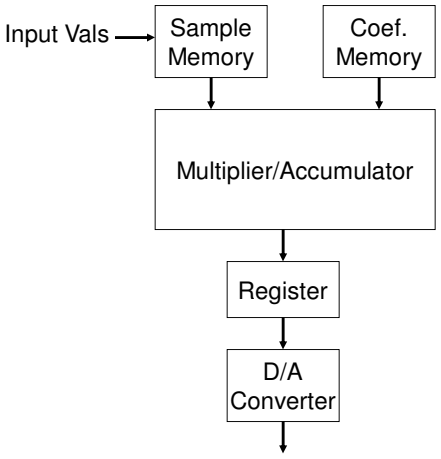
Edge Detector



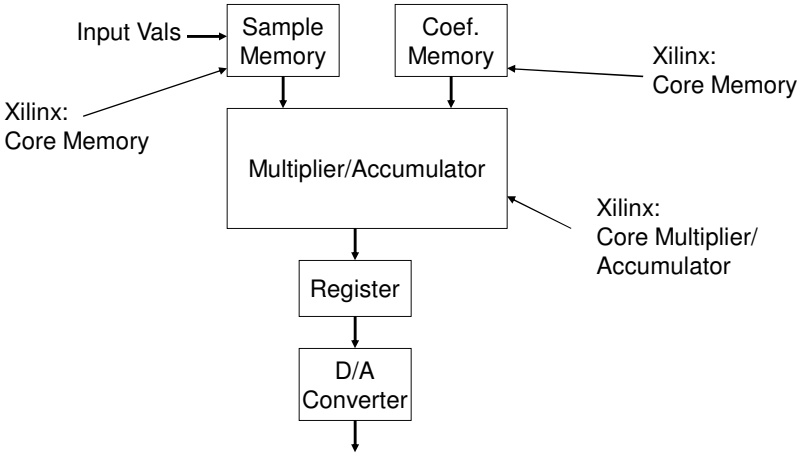
Entity Statement

```
entity FIR is
  port (
    SYS_CLK_H  : in  STD_LOGIC;
    SYS_RST_H  : in  STD_LOGIC;
    DATAIN     : in  STD_LOGIC_VECTOR ( 15 downto 0 );
    DATAOUT    : out STD_LOGIC_VECTOR ( 15 downto 0 );
    SAMPLE_H   : in  STD_LOGIC;
    DONE_H     : out STD_LOGIC
  );
end entity FIR;
```

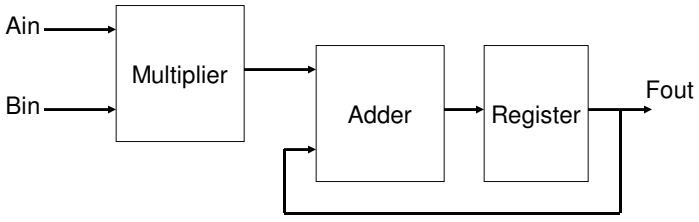
Block Diagram (1)



Block Diagram (1a)



MAC Construction



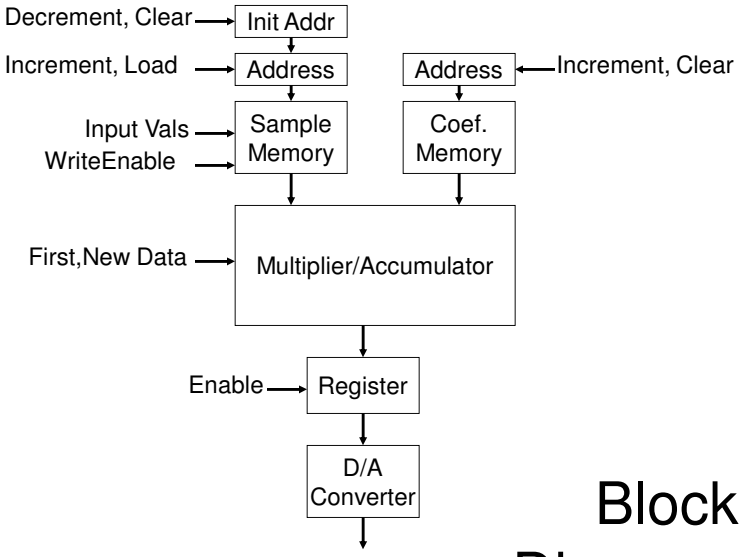
MAC Component Statement

```
component MAC23B is
  port (
    A   : in  STD_LOGIC_VECTOR ( 15 downto 0 );
    B   : in  STD_LOGIC_VECTOR ( 15 downto 0 );
    Q   : out STD_LOGIC_VECTOR ( 36 downto 0 );
    CLK : in  STD_LOGIC;
    FD  : in  STD_LOGIC;
    ND  : in  STD_LOGIC;
    RDY : out STD_LOGIC;
    RFD : out STD_LOGIC
  );
end component MAC23B;
```

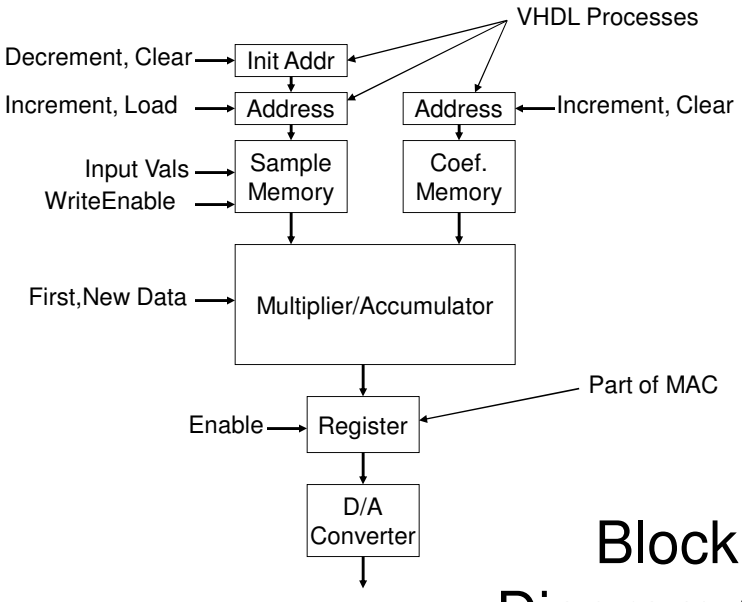
Memory Component Statements

```
component COEFMEM
  port (
    ADDR : in  STD_LOGIC_VECTOR (  5 downto 0 );
    CLK  : in  STD_LOGIC;
    DOUT : out STD_LOGIC_VECTOR ( 15 downto 0 )
  );
end component COEFMEM;

component SAMPMEM
  port (
    ADDR : in  STD_LOGIC_VECTOR (  5 downto 0 );
    CLK  : in  STD_LOGIC;
    DIN  : in  STD_LOGIC_VECTOR ( 15 downto 0 );
    DOUT : out STD_LOGIC_VECTOR ( 15 downto 0 );
    WE   : in  STD_LOGIC
  );
end component;
```



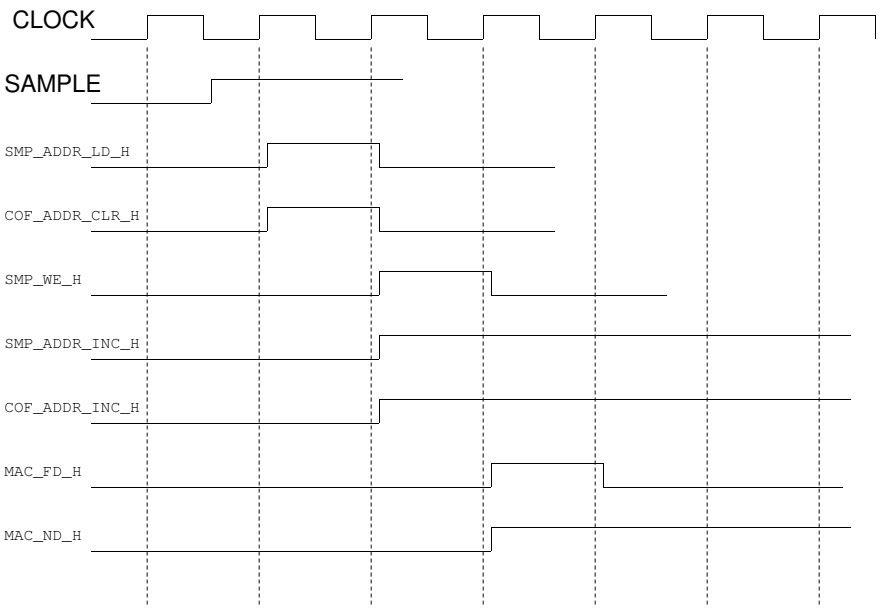
Block Diagram (2)

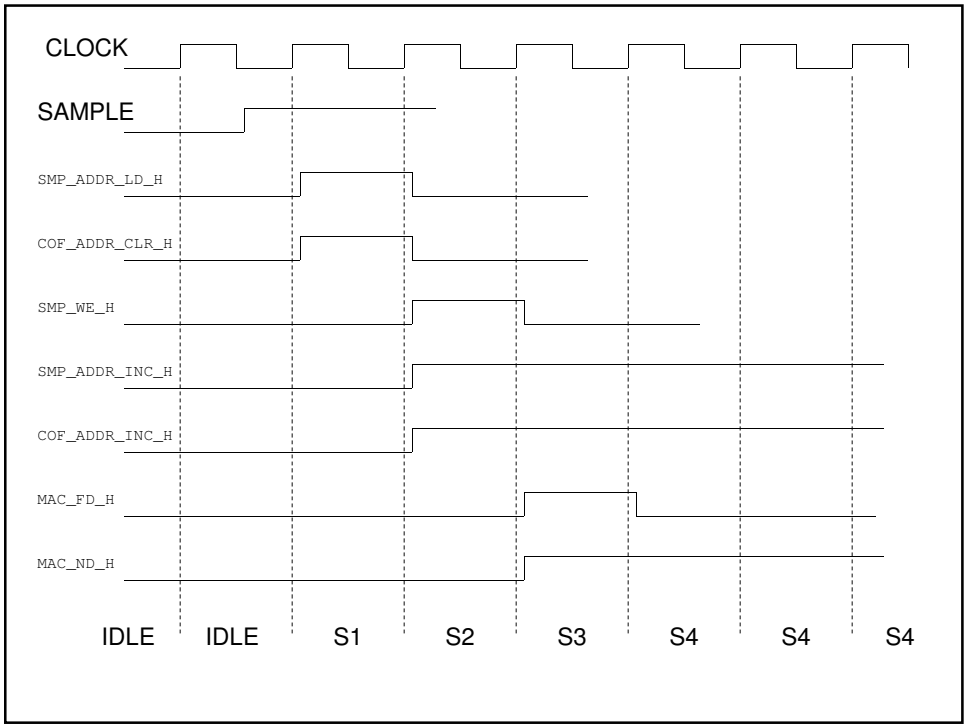


Block Diagram (2a)

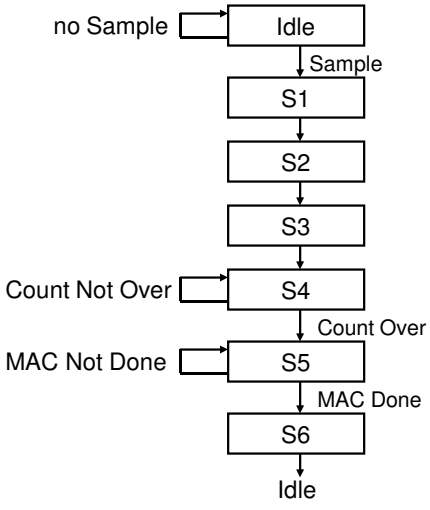
Algorithm:

1. Wait for new sample
2. Load initial address, clear Coef address
3. Write new sample to data memory
4. Load data, Coefficient to MAC (first data, new data)
   also, increment address pointers
5. Load data, Coefficient to MAC (new data)
   also, increment address pointers
   repeat N-1 times
6. Load value into output register

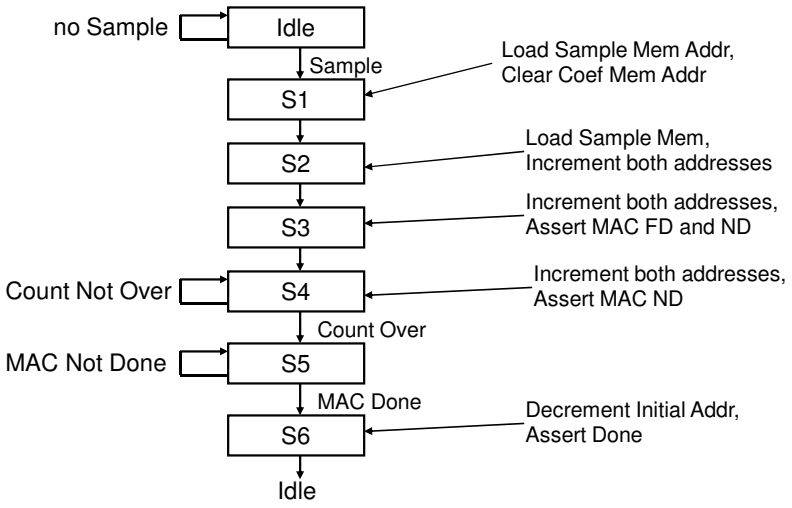




FIR State Diagram



FIR State Diagram



Clocked Sequential System Design