Chapter 8

DESIGN AND IMPLEMENTATION OF 200 Hz FILTER AND DESIGN OF MIL-STD-1553B BUS INTERFACE CIRCUIT

8.1 DESIGN AND IMPLEMENTATION OF 200 Hz DIGITAL FILTER

8.1.1 INTRODUCTION

Resolution is limited by the noise floor and the noise floor is determined by the bandwidth. It is very difficult to trim the resonance frequency of the microstructure to obtain 200 Hz bandwidth. The bandwidth of the signal has to be limited to 200 Hz.

In order to realize an integrated smart accelerometer, a digital filter, which can be easily implemented on the same silicon chip containing the sensor microstructure, has been chosen. For reducing the computational complexity an Infinite Impulse Response (IIR) Filter has been chosen.

8.1.2 DESIGN OF SECOND ORDER IIR DIGITAL FILTER

A second order digital filter is designed for a bandwidth of 200 Hz. A sampling rate of 1000 samples/second has been chosen. Digital filter is designed with 200 Hz bandwidth in analog domain; it is translated to digital domain using bilinear transformation. Due to warping effect, the digital filter will have a bandwidth of 178.25 Hz.

Warping effect:

The relation between the frequencies in analog and digital domain is given by the equation

$$\omega = (2 / T) * \tan (\Omega T/2)$$

where $\omega \rightarrow$ angular frequency in analog domain

$\Omega \rightarrow$ angular frequency in digital domain

As we increase the sampling frequency, the digital filter and analog filter characteristics become identical. The obtained digital frequency vs sampling ratio is
Given in Fig. 8.1. By selecting sampling ratio as 24, the difference in frequency can be brought down to 1.30 Hz at cut off. Hence, the sampling frequency of 4800 Hz is selected, where the warping effect is negligible (<1%).

The design of the second order IIR filter involves transformation of analog transfer function (eqn. 8.1) into digital domain.

\[ H(S) = \frac{\omega_c^2}{S^2 + 1.414 \omega_c S + \omega_c^2} \]  

Bilinear transformation is a frequency domain method of converting the analog filter transfer function \( H(S) \) into a digital one \( H(Z) \) replacing \( S \) by

\[ \frac{2/T}{(z-1)/(z+1)} \] where \( T \) is the sampling interval and \( Z = e^{j\omega} \).

\( H(Z) \) is computed as

\[ H(Z) = \frac{a_0 + a_1Z^{-1} + a_2Z^{-2}}{b_0 + b_1Z^{-1} + b_2Z^{-2}} \] , where

\[ a_0 = T^2 \cdot \omega_c^2 \quad b_0 = 4 + 2.828 \cdot \omega_c \cdot T + a_0 \]

\[ a_1 = 2 \cdot a_0 \quad b_1 = 2 \cdot a_0 - 8 \]

\[ a_2 = a_0 \quad b_2 = 4 - 2.828 \cdot \omega_c \cdot T + a_0 \]
For a bandwidth of 200 Hz and sampling rate of 4800 samples/second, the coefficients are computed as below.

\[
H(Z) = \frac{A_0 + A_1 Z^{-1} + A_2 Z^{-2}}{1 + B_1 Z^{-1} + B_2 Z^{-2}} \tag{8.2}
\]

Where \( A_0 = a_0/b_0; \ A_1 = a_1/b_0; \ A_2 = a_2/b_0; \)

\[
B_1 = b_1/b_0; \ B_2 = b_2/b_0;
\]

For a bandwidth of 200 Hz and sampling rate of 4800 samples/second, the coefficients are computed as below.

\[
A_0 = 0.014252 \quad A_1 = 0.028505 \quad A_2 = 0.014252 \tag{8.3}
\]

\[
B_1 = -1.63507 \quad B_2 = 0.692084 \tag{8.4}
\]

8.1.3 DIRECT FORM REALISATION

Fig 8.2 shows the implementation of an IIR digital filter. Using the standard notation, the input signal is referred to \( X[\cdot] \), while the output signal is denoted by \( Y[\cdot] \). Our task is to calculate the sample at location \( n \) in the output signal, i.e., \( Y[n] \). An IIR filter performs this calculation by multiplying appropriate samples from the input signal by a group of coefficients denoted by \( A_0, A_1, A_2 \), multiplying the appropriate samples from the output signal by a group of coefficients denoted by \( B_1, B_2 \) and then adding the products. For a second order filter in equation form \( Y[n] \) is found by

\[
\]

Difference equation for low pass digital filter with 200 Hz cut off is given below.

\[
Y[n] = 0.014252 X[n] + 0.028505 X[n-1] + 0.014252 X[n-2] + 1.63507 Y[n-1] - 0.692084 Y[n-2]
\]

8.1.4 COMPUTATION OF FREQUENCY RESPONSE OF THE DESIGNED FILTER

The frequency response of the designed filter is obtained by replacing \( Z \) by \( e^{j\omega} \) in eqn 8.2.

Numerator, \( N.R = A_0 + A_1 (\cos \omega - j \sin \omega) + A_2 (\cos 2\omega - j \sin 2\omega) \)

\[
= (A_0 + A_1 \cos \omega + A_2 \cos 2\omega) - j (A_1 \sin \omega + A_2 \sin 2\omega)
\]

Denominator, \( D.R = 1 + B_1 (\cos \omega - j \sin \omega) + B_2 (\cos 2\omega - j \sin 2\omega) \)

\[
= (1 + B_1 \cos \omega + B_2 \cos 2\omega) - j (B_1 \sin \omega + B_2 \sin 2\omega)
\]
Fig 8.2 IIR Digital Filter
Algorithm:

The coefficients of the digital transfer function $H(Z)$ are initialized with the computed values in (8.3) & (8.4).

Vary $\omega$ from 0 to pi. (500Hz)

Real part of N.R, $a = (A_0 + A_1 \cos \omega + A_2 \cos 2\omega)$

Imaginary part of N.R, $b = (-1)* (A_1 \sin \omega + A_2 \sin 2\omega)$

Real part of D.R, $c = (1 + B_1 \cos \omega + B_2 \cos 2\omega)$

Imaginary part of D.R, $d = (-1)* (B_1 \sin \omega + B_2 \sin 2\omega)$

Magnitude of $H(Z)$ $h = \sqrt{(a*a+b*b)/(c*c+d*d))}$

c $= 20\log_{10}(h)$. (dB)

The cut off frequency obtained from the plot is 198.70 Hz.

The designed filter has been verified with the help of the computed plots.
The flow chart is given in Fig 8.3

START

$A_0 = 0.014252$
$A_1 = 0.028505$
$A_2 = 0.014252$
$B_1 = -1.63507$
$B_2 = 0.692084$
$\omega = 0.0$

\[ a = A_0 + A_1 \cos \omega + A_2 \cos^2 \omega \]
\[ b = (-1) \times (A_1 \sin \omega + A_2 \sin 2\omega) \]
\[ c = 1 + B_1 \cos \omega + B_2 \cos^2 \omega \]
\[ d = (-1) \times (B_1 \sin \omega + B_2 \sin 2\omega) \]
\[ h = \sqrt{\frac{a^2 + b^2}{c^2 + d^2}} \]
\[ e = 20 \times 10^{\log_{10}(h)} \]

Output $\omega$, $e$

$\omega = \omega + 0.01$

Is $\omega \leq 3.14$

Yes

No

STOP

Fig 8.3 Algorithm flow chart
Magnitude in dB vs $\omega$ plot is given in fig 8.4.

Fig 8.4 Simulated Frequency response of 200 Hz digital filter

8.1.5 IMPLEMENTATION IN ADSP 21061

For implementation, the Digital Signal Processor ADSP-21061, a 32-bit floating point Super Harvard Architecture Computer (SHARC) from Analog devices has been selected. It can carry out a multiply–accumulate operation in a single clock cycle. The goal is to move the data in, perform the mathematical computation, and move the data out, before next sample is available.

There are three on-chip buses of the ADSP-21061: the PM (Program Memory), DM (Data Memory) & I/O bus. In ADSP 21061, the filter coefficients come from the program memory, data come from data memory and program instructions come from Instruction cache that contains most recent program instructions. Signal enters and exits the system through the I/O controller. The second order IIR filter uses 5 coefficients. Adding a small amount of overhead say 3 clock cycles per sample results in 8 clock cycles per sample. Hence the throughput of the algorithm works out to be 5 M samples per second at a clock speed of 40 MHz.
Block schematic of the ADSP21061 target board is shown in fig 8.5. The DSP is interfaced with A/D and D/A converter in the target board. The algorithm used for the designed filter in C language is integrated in Visual Dsp++ environment resident in the computer. The code is compiled and the executable file is downloaded to the internal memory of the ADSP 21061 through RS 232 interface between computer and the target board.

The input sine wave is fed with a gain of 5 to a 16-bit ADC in the target board. The ADC output is given to the ADSP for processing. The filtered signal from ADSP output port is directly given to a 16-bit DAC in the target board, which provides an output voltage in the range of -10 V to +10V. The frequency response is plotted in fig 8.6. The cut off frequency obtained is 202 Hz. The gain flatness is 0.19 dB. The roll off rate obtained is 40 dB/decade. The execution time for the processor is 0.52 micro second.

The selected processor uses floating-point computation units; it offers increased dynamic range and improved accuracy of processing. The effect of coefficient quantization error is negligible. For memory constraint applications, digital filter can be realized in direct form II to reduce the memory requirements.
8.6 CONCLUSION

The frequency response is simulated and verified. The cutoff frequency of the filter implemented in ADSP 21061 matches with the designed filter.
8.2 DESIGN OF MIL-STD-1553B BUS INTERFACE CIRCUIT

8.2.1 INTRODUCTION

System on-chip concept is incorporated in the newly proposed MEMS accelerometer. In order to meet the end to end requirement the accelerometer must have a digital interface with navigation processor. Although various types of interfaces are available, MIL-STD-1553B Bus interface is proposed in the new design since it is most popular for launch vehicle application. This section covers the design of circuits for MIL-STD-1553B Bus interface which is a part of ASIC to be made in the system on-chip design.

8.2.2 MIL-STD-1553B BUS

It is a serial data transmission bus standard. Only bipolar Manchester encoded data at a bit rate of 1 Mbps can be transmitted. Every data through the 1553B Bus is transmitted as words. The word format is given in figure 8.8. The transmission and reception format is given in figure 8.9. A word starts with one synch pulse of 3 \( \mu \text{s} \) followed by 16 data bits and one parity bit. Thus a word has 20 bit duration (20 \( \mu \text{s} \)) and odd parity. There are 3 types of words: command word, data word and status word. Synch pulse of command word and status word is high for the first 1.5 \( \mu \text{s} \) and low for next 1.5 \( \mu \text{s} \) and vice versa for the data word. The bus controller and status word generate the command word by the Remote Terminal (RT).

The MIL-STD-1553B Bus standard defines four hardware elements. The system configuration is given in figure 8.7. These are:

1. The transmission media.
2. Remote terminals.
The transmission media, or data bus, is defined as a twisted shielded pair transmission line consisting of the main bus and a number of stubs. The remote terminal comprises the electronics necessary to transfer data between the data bus and the subsystem. The bus controller is responsible for directing the flow of data on the data bus. A bus monitor is a terminal that listens (monitors) to the exchange of information on the data bus.
When a command word is generated from Bus Controller (BC), it is received by remote terminals of all systems connected to the bus, decode it and check whether the command is for the particular RT itself or not. This is done by checking the RT address in the command word. If the command is for the particular RT itself, then it will be executed by the RT. The command can be for receiving data by the system from the bus or for transmitting data to the bus from subsystem. If it is for reception, the RT has to transmit a status word to the bus after complete reception of data. If the command is for transmission, the RT has to transmit a status word to the bus before transmitting the data.

8.2.3 DESIGN OBJECTIVES

The aim is to design an interface between sensing part of the system and the 1553B bus with the following objectives

- Decoding of the incoming command word and identify whether it for the particular RT or not.
- Identification of Sync. (Synchronization) command in broadcast command mode and transmit command to the particular RT.

Sync command in Broadcast mode recognized by multiple RTs simultaneously and here it is used for latching data of multiple MEMS accelerometer which are part of navigation system.

- Latching of input data upon the reception of Sync command
• Transmissions of 16bit data word along with status word when appropriate transmit command received which includes the following steps

- Creation of status word
- Encoding of status word and data word
- Transmission of status word and data word to the bus in MIL-STD-1553B bus format (adding of proper sync pulse, outing of serial data at 1MBPS etc)

The overall objective is to design an interface which will transmit a status word followed by a data word to the 1553B bus controller when a transmit command word is received. Here the data word will contain the sensed acceleration information at the instant of reception of Sync Command.

8.2.4 BLOCK DIAGRAM OF MIL-STD-1553B INTERFACE

Figure 8.10 shows the block diagram for implementing an interface (remote terminal) between MIL-STD-1553B serial data bus and sensor electronics. Data transmission and reception between RT and 1553B bus is taking place through the transceiver. An isolation transformer is also provided between trans-receiver and 1553B bus for proper DC isolation. This calls for a multi-chip module packaging [18]. During reception the trans-receiver will convert the incoming high level bipolar signal from 1553B bus to digital unipolar signal for processing by the digital circuits in the RT. This is done by comparing the incoming signal with a reference threshold voltage. During transmission, the trans-receiver will convert the unipolar digital signal from the digital blocks to bipolar high level signal for transmission through the 1553B bus. Thus it acts as both level shifters as well as unipolar to bipolar and vice versa converter.

The received signal from trans-receiver will go to command decoder and RT address checking block where any received command will be decoded and checked for its RT address. The command can be a Sync command in broadcast mode or a transmit command. If the command is a Sync command, then it will give a latching pulse to the sensor electronics to latch the measured incremental acceleration at the moment. This latched data will be the input to the interface for transmission. If the command is a transmit command with matched RT address, then the command
decoder block will give a transmit data signal to the encoding and sequencing block. This signal tells the encoding and sequencing block that the current 16 bit input data has to be transmitted. In this block the 16 bit data from sensor electronics and status word from status word generator will be Manchester encoded and serially shifted out to the trans-receiver after adding appropriate sync pulse in 1553B bus format (i.e. status word followed by data word).

Fig 8.10  Block diagram of MIL-STD-1553B Interface
8.2.5 DESIGN OF DIGITAL PORTION OF 1553 INTERFACE

Figure 8.11 shows the block diagram for implementing the digital part of an interface which can transmit 16 bit data to a 1553B serial data bus at a data rate of 1 Mbps according to the received command word. In the block diagram shown, the input from the trans-receiver is given to the synchronous pulse detector and the Manchester decoder. The sync pulse detector detects whether the receiving word is a command by detecting the sync pulse. If the input contains valid command sync, the Manchester decoder will be enabled for 17 µs by the cmd enable signal from the sync pulse detector. The Manchester decoder decodes the input and produces the actual digital (binary) data and the clock used for encoding the data. The decoded data and the clock is given to Serial In Parallel Out (SIPO) and the received word will enter into the SIPO. After the reception of one word, the sync pulse detector will produce a command buffer clock to shift the word from SIPO to command buffer. The parity checker will identify any error in the parity of received word and produce a high output for any error. The command buffer is used to hold the data in command word till the next command word comes. The RT address checker will compare the RT address in the received command word with the RT address of the remote terminal and produces a high output if they are same. This output is used to start the transmission of data from the interface by enabling the clk generator, transmission logic, status word creator etc. An RT-enable equals high shows that the command word from 1553B bus is for this RT itself.

The broadcast command checker will detect sync command in broadcast mode and will give a latching pulse to sensor electronics. A status word should be transmitted at the beginning of transmission. The status word is created in the status word creator. The data should be given to the bus serially, so a Parallel In Serial Out (PISO) is used to out the data serially. The parallel input data to the PISO can be a 16 bit word from sensing part or a status word from status word creator. To select the proper input to PISO, a mux is used. The clk generator will provide 1 MHz & 50 KHz clock by dividing its 20 MHz input. The PISO will be loaded with the 16 bit status word or sensed data at the rising edge of 50 KHz clk. The 1 MHz clk is used for serial outing of data from PISO and for manchester encoding. The transmission logic will
Fig 8.11 Block diagram of digital part of interface
give selection signal for MUX and stop signal for clk_generator. The same signal which is using to stop clk_generator is using as enable signal for output logic after passing through delay circuit. The parity adder will add a 0 or 1 to the 16 bit word from PISO to make a 17 bit word of odd parity. Now the serial 17 bit word is Manchester encoded and given to the 1553B bus after adding the appropriate sync pulse (data/status sync) in the sync pulse adder. A pulse eliminator is used for removing all unwanted pulses. The output logic will give high impedance state when enable signal to output logic is low and its output will be same as the input when enable signal is high. Design of each block covered in details in subsequent sections

Sync pulse detector

Every word through the 1553B bus contains a sync pulse at the beginning of the word. The width of a sync pulse is 3 μs. The sync pulse for data/command word is different. For a command word, sync pulse is high for the first 1.5 μs and then low for the next 1.5 μs and vice versa for a data word. In this particular design, sync detector is designed only for detecting command sync. The sync pulse detector receives the input from the 1553B bus and detects any sync pulse in the receiving word. If it detects any cmd sync pulse, it will make the command enable signal high for the next 17 μs (for receiving the 17 bit associated with the sync pulse). This command enable signal is used to enable the Manchester decoder, i.e. the Manchester decoder will start decoding the input data only after the reception of a correct sync pulse.

The basic logic diagram is shown in Fig 8.12. The high and low period of input is counted in two separate counters after receiving a low to high transition in the input using a high frequency clock. If clock count is exceeding more than 1.4 μs in both counters before next raising edge, it is a valid command. This signal is used to enable a pulse width generator of 17 μs using a counter and digital comparator. Manchester decoder is enabled during this period.
Manchester Decoder

The block diagram is given in Fig 8.13. The present circuit relates to a circuit for extracting separate data and clock signal from a Manchester encoded digital communication signal. The circuit includes a pair of latch circuits which are used to detect transition of edges in the encoded signal for providing respectively set and reset signals to a third latch circuit, an output of which comprises the decoded data of the Manchester signal. The circuit also includes two delay elements, input and delay matching buffers and one or more logic gates. Logical combination of the decoded data with a delayed encoded signal provides a decoded clock.
Data signal at D input of FF1 represents the original encoded signal, delayed by $\frac{1}{4}$ of a data signal. If, when a rising edge transition in the inverted signal causes FF1 to sample the data at its D input, the sampled data is a 1, then it is known that original encoded signal has been a 1 for 2 consecutive code bit cells and that the decoded data should change from a logical 0 to a logical 1. Flip-flops FF1 accordingly outputs a 1 which is coupled to set control input of an SR flip-flop FF3. This transition causes Q output of FF3 to be set to a 1. This Q O/P is coupled back to the reset control input of FF1 to cause the Q output of FF1 to return to 0 after FF3 has been successfully set. If on the other hand data at D input of FF1 is a 0 when sampled, it is known that the original encoded signal has not been a 1 for 2 consecutive code bit cells, indicating that the encoded data has not changed from a logical 0 to 1. So FF1 remains in 0 state and FF3 is not set.
Flip-flop FF2 operates in a similar manner to reset FF3 whenever the data at its D input is a 0 when flip-flop FF2 is clocked by a rising edge in signal. Data signal at D input of FF2 represents the original encoded signal, delayed by \( \frac{1}{4} \) of a data cell. If, when a rising edge transition in the non inverted signal causes FF2 to sample the data at its D input, the sampled data is a 0, then it is known that original encoded signal has been a 0 for 2 consecutive code bit cells and that the decoded data should change from a logical 1 to a logical 0. FF2 outputs a 1 on its inverted output which is coupled to reset control input of SR flip-flop FF3. This transition causes Q output of FF3 to be set to 0. Signal at inverted Q o/p of FF3 is coupled back to set control input of FF2 to cause inverted Q o/p of FF2 to return to 0 after FF3 has been successfully rest. If on the other hand data at D input of FF2 is a 1 when sampled, it is known that the original encoded signal has not been a 0 for 2 consecutive code bit cells, indicating that the encoded data has not changed from a logical 1 to 0. So FF2 remains in 1 state and FF3 is not reset.

The Q o/p of FF3 is coupled to input of each x-or gates x1 & x2. A 2\textsuperscript{nd} i/p of x-or x1 is coupled to half clock period delay line of delay circuit by a plurality of delay matching buffer. There is a delay in the decoded data of signal relative to encoded data input. This includes a half data cell delay, which in recovering a clock signal, is compensated by delay of half clock period. There is additional delay in decoded data due to a FF1/FF2 and FF3. Delay matching buffer circuits compensate for this delay.

The clock signal at the output of XOR gate x1 is inverted with respect to original signal. This is useful because the rising edge of the inverted clock occurs in the center of each data cell of decoded data. Because the decoded data signal is coupled to XOR gate x1 to generate the recovered clock. If it is desired that the propagation delay of XOR gate x1 be cancelled, the decoded data signal can be coupled through an optimal element delay element.

For enabling or disabling the decoder a 2 input AND gate is provided before the decoder one of its input is command enable signal from sync pulse detector the other is the data from transceiver. The Manchester encoded data will be given for decoding only if the command enable signal is high.
SIPO

SIPO stands for serial in parallel out. The SIPO is 17 bit long. Every shift register is a combination of D Flip-flop which are connected serially. At every clock the input data at the shift registers are shifted to right and will come at the output lines. Thus after 17 clock cycles the whole serial data (cmd word) will be available in parallel.

Command Buffer

Command buffer is a group of 17 registers used to store the incoming command signal. Parallel in Parallel out logic is used and the command buffer clock from the sync pulse detector is given as the input clock. On the rising edge of the command clock the data is loaded into the buffer from the SIPO. The output of command buffer is given to various other logic circuits.

RT Address checker

RT stands for remote terminal. This is basically a digital comparator that compares the received remote terminal address with RT address input of particular device. This module gives a high signal if the received RT address is equal with the actual one.

Broadcast command checker

It is also a digital comparator which compares the RT address in the command with binary number 11111 which is used by broadcast commands. If the command pulse is a sync command in broadcast mode then this will give a pulse to the sensor electronics.

Parity Checker

The purpose of parity checker is to find the parity of the received word. 1553 bus uses an odd parity. The basic function of the circuit is as follows. The 17 bit data from the SIPO is given to the 17 input pins of parity checker. These 17 bits are XNOR ed to get the parity bit. This parity bit is given to the D input of DFF. The DFF will output the parity only during the rising edge of command buffer clock (after the reception of the complete command word). This will come as the output of parity checker if its enable pin is high. (i.e if rt_enable is high). The parity error bit will be high for even parity. The parity error bit from parity checker is given to the status word creator.
Status word Creator

The function of status word creator is to create the status word. It will create a status word with the first five bits as RT address and the 6th bit as parity out and remaining all as zeros. The bits from 6 to 15 indicate several additional functionalities such as broadcast, RT to RT communication etc; which are disabled here. It will give the status word only if rt_enable is high.

MUX

It is a normal 16 bit 2:1 MUX. It has two 16 bit inputs, status word from status word creator and data from input data line. The output from MUX will be either status word or data depending on the selection signal input. If the selection signal input is high then output will be status word otherwise data.

PISO and Parity adder

PISO consists of a set of 17 registers, used to store the outgoing data for status/data word and 3 additional registers used to give a 3 μs delay, for inserting the sync pulse to be transmitted. Input is loaded to the PISO at the rising edge of 50 KHz clock and data is serially transmitted using a 1 MHz clock. The 17th bit of PISO is a parity bit and stores the parity of the 16 bit data/status word being transmitted. Status word and data word are alternatively loaded in the PISO at the rising edge of 50 KHz clock on the basis of transmission logic. The parity bit is obtained by XOR ing 16 input bit to the PISO and it will become the 17th bit of data or status word to make them a 17 bit word of odd parity.

Manchester Encoder

The Manchester encoder is a XNOR gate. Here the data from PISO and 1MHz clock are given as XNOR inputs. The output will be Manchester encoded data.

Sync pulse adder

The purpose of sync pulse adder is to incorporate the sync pulse. The sync pulse is a 3 μs duration signal, with 1.5 μs high and 1.5 μs low: in which the 1st half high for status word and 1st half low for data word. The PISO for taking the 17 bit data is extended by 3 bits to 20 bits using shift key registers. This is for getting time for the sync pulse addition. The 1MHz clock from clk generator is used in conjunction with a toggle flipflop for the sync pulse production. The TFF output will go high during
power on. This will go low only after 1.5 micro seconds after the first rising edge of 50 KHz clk. For getting this 1.5 micro second duration only, the 1MHz clk is delayed by 0.5 micro second. The output of TFF will again go high after 20 micro seconds exactly. The non delayed 1 MHz clk has given to the counter, selection logic combination. The selection logic output will be high for 3 micro seconds after the rising edge of 50 KHz clk (for adding sync pulse of status word). During this period the TFF output will be selected. For the next 17 micro seconds the output of Manchester encoder is selected as the output of sync pulse adder which is the status word. Again for the next 3 micro seconds selection logic output will go high to transmit data sync for the coming data word.

**Pulse Eliminator**

It is used to remove any unwanted pulses (pulses having width less than the width of a valid Manchester encoded data i.e 0.5 micro sec) which are created due to some unavoidable gate delays. This pulse eliminator will remove pulses having width less than 0.3 micro sec. Six DFF are connected in series and with 20 MHz synchronous clock to give 0.3 micro sec delay in its input. The input and delayed output are ANDed to give clk signal to the DFF. Inverted input and inverted delayed output are ANDed to give clear signal to the DFF. The D input is connected to Vcc. Output become high at the rising edge of clk and low at the rising edge of clr. Thus this circuit can eliminate both +ve and -ve pulses.

**Output logic**

It is nothing but a tristate buffer which will give an output same as input if its enable signal is high otherwise a high impedance state. The enable pin remain high only 40 micro second for the transmission one status word and data word.

**Clock generator**

The 50 KHz and 1 MHz clocks for other blocks are given by clock generator. Basically it is a clock divider which divides a 20 MHz clock by 20 and 40 to give 1 MHz and 50 KHz clock respectively. It will start generating the clocks at the rising edge of rt_enable signal from RT address checker and stops at the falling edge of stop signal from transmission logic. The 1 MHz clock will come at the output of clock generator only after the rising edge of 50 KHz clock.
Transmission logic

The transmission logic will give selection signal for MUX and enable signal for output logic. It basically consists of a counter which counts the 1 MHz clock from clock generator. Its selection signal output will go high during the rising edge of rt_enable and the counter gets reseted. The selection signal will remain high till the counter count up to 3. The enable signal go high during the rising edge of 50 KHz clock and becomes low when the count becomes 40. Thus the enable signal remain high for 40 micro seconds during which one status word and one data word will be transmitted. The enable signal will be delayed before giving to the output logic. This is necessary because in the pulse eliminator the data to be transmitted is delayed by 0.3 micro seconds.

8.2.6 CONCLUSION

This section deals with the principle, design and working of the interface section (remote terminal) designed to convert a 16 bit digital data from the sensor part of the system to MIL-STD-1553B for transmission through 1553B serial bus when appropriate command word is received from the bus. It analyses the data from the 1553B bus, distinguishes any command word in it and responds to appropriate command word by sending a status word and one data word of Incremental Velocity from the system. Command word is decoded and directed in to the command buffer where it remains and gets executed till the arrival of the next command word. The remote terminal follows the 1553B protocols of transmission and reception and uses a Manchester encoder-decoder for encoding and decoding of data respectively. Design of logic Blocks for this interface is carried out and it is coded using VHDL. Complete simulation is carried out. Libero IDE software and hardware testing was done using Pro ASIC. The module can be integrated with other electronics to form complete ASIC with end to end design.