MCC150-Implementation of Digital Signal Processing
Lab 3: Symbol Timing Recovery

Sundar Murugan Ramaswamy, Chalmers University of Technology.

I. Introduction

The RF transceiver which was built in lab 2 is expanded in order to add support for symbol timing recovery (STR). In the previous lab, the sample values are manually set but in this lab DSP modules are added to calculate the best sampling point. The DSP architecture is built using Simulink and hardware is generated in VHDL. The model is built and then examined and synthesized using Quartus prime.

II. PRE-LAB TASKS

1. The largest value of an N bit, unsigned number is 2^N -1. Storing L such numbers, the largest value will be L*(2^N - 1). The number of bits needed to store that value is log2(L*(2^N - 1)) < log2(L*2^N) = log2(L)+N. But an integer number will be needed, so we will need to round to the nearest higher integer ceil(log2(L) + N).

III. DSP BUILDER MODEL

The subsystems used in the building of the transmitter part of the transceiver are a PRBS Generator which generates Pseudo Random Binary Sequence or bitstream which is the data to be transmitted. The output of the Sequence Generator is then passed through a modulator. The modulator fuses the input signal with a carrier signal which is produced by the Local Oscillator. The output of the modulator is scaled and added below. The *Interpolating FIR* block increases the sampling frequency of the input signal. A *Scale* block is the added to the output of the *interpolating FIR* in order to reduce the precision which can perform rounding and saturation to provide the required output precision. The display resources block is also added to view the utilization of resources. A Control block is set up to generate the HDL code which can be used for synthesis. In the control block, generation options and clock frequency can be specified. The bandwidth used in real time communication systems is limited. Also, a square wave shaped is hard to be transmitted since its bandwidth is infinite. Hence, a low pass filter can be used to limit the bandwidth of the signal which converts the shape of the signal which will no longer be a square. Hence it is called as a pulse shaping filter. It has a longer time domain response. The Finite Impulse Response filter is a type of digital filter used for digital signal processing. The main purpose of this filter is signal conditioning. Some examples are phone line. These frequencies are limited in such a way that it cannot be heard by human beings (very low frequencies). The middle point of each pulse is the best sampling point with no Inter-Symbol Interference. The process of looking for the best sampling points is called symbol time recovery (STR). It is mandatory for every primitive subsystem to have a *Channel in* and *Channel out* block to mark the boundary of the system. Additionally, a *synthesis info* block is also required to achieve pipelining (to achieve higher clock frequency) and latency (to assume zero latency). Also, delays are added to the subsystems whenever needed to meet the clock frequency target.

3.1 Reflection Questions:

1. Considering the fact that an adder will wrap around to zero when reaching the highest value -1 as an advantage, the oversampling ratio can be limited to the powers of 2 inside the increment subsystem by removing the need of the comparison operation.

2. The result will always fit into the data type since the integer bits and overflow only are taken into consideration. Hence, we can say that the result will always fit in the data type. Since the largest value of a 12 bit signed number is 2^11 - 1 = 2047, the largest value of the magnitude will be sqrt(2047^2 + 2047^2) = 2047*sqrt(2), which is always smaller than the largest value we can fit in an unsigned 12 bit value since 2^12 - 1 = 4095 which is lot more than required.

IV. RECEIVER ARCHITECTURE

The receiver is built by adding a pulse shaping filter and a Symbol Timing Recovery unit. The *decimation* block is added to the *single rate FIR* filter. Decimation is nothing but down sampling of the signal (sample rate reduction) which is usually done to reduce the complexity in computation. The *decimation* subsystem is updated in such a way that it outputs both the selected symbol and one of its adjacent symbols which is achieved by adding an *increment* block.

V. UPDATED DECIMATION AND MAGNITUDE CALCULATION

The decimation subsystem is now updated to support STR. An increment block is added such that the subsystem calculates the adjacent index i.e., i+1. Then a subsystem is created to calculate the magnitude of the received complex I/Q samples. The I and Q signals which are the input to the magnitude

block is squared, added and then the square root of that value is passed to the output. The magnitude is calculated using the formula,

$$Magnitude = \sqrt{a^2 + b^2}$$

Where,

a- real part of the signal (I)
b- complex part of the signal (Q)

VI. AVERAGE MAGNITUDE

A subsystem to calculate the average magnitude of the selected and its adjacent symbol is created. The average is calculated for a set window(contrasting the moving average method) and the output is updated when the window is complete.

3.2 Reflection Questions:

1. In Latch_1L, the output port of the subsystem is connected to the output port of the sample delay. In case of Latch_0L, the output port of the subsystem is connected to the output port of the mux. From this we can infer that, the Latch_0L includes the feedback from the *sample delay* block at that instant. Moreover, the Latch_0L is directly connected to the output and it is dependent on whether the output from Latch_1L is high or low. If high, the data passes through and if low, the previous data is held. It cannot be removed.

2. The drawback of selecting the averaging window is that it limits the selection of possible window sizes to the powers of two. The advantages are that it reduces computational cost and time and reduces the utilization of resources.

VII. UPDATE SAMPLE INDEX

Now a system is built to update the sample index which was chosen to be the best. It compares the average magnitude of the selected symbol (*AvgMag_out*) with the average magnitude of the adjacent signal (*AvgMag_adj*). If *AvgMag_out* < *AvgMag_adj*, the index is increased one position and if *AvgMag_out* >= *AvgMag_adj*, the index is decreased by one.

3.3 Reflection Questions:

1. I would put efforts on both magnitude subsystems for the optimization as it consumes the maximum number of resources.

Analysis & Synthesis Resource Utilization by Entity

	Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits
1	MCC150_top	4281 (1)	6787 (0)	367924
1	AD9361:AD9361_inst|	2194 (7)	475 (7)	8500
2	MCC150_TransceiverBPSK:MCC150_TransceiverBPSK_inst|	1665 (0)	4970 (0)	205824
1	MCC150_TransceiverBPSK_AverageMagnitude1:theAverageMagnitude1|	43 (43)	80 (35)	0
2	MCC150_TransceiverBPSK_AverageMagnitude2:theAverageMagnitude2|	43 (43)	79 (35)	0
3	MCC150_TransceiverBPSK_Decimation:theDecimation|	8 (8)	80 (54)	0
4	MCC150_TransceiverBPSK_IndexUpdate:theIndexUpdate|	22 (22)	11 (7)	0
5	MCC150_TransceiverBPSK_InterpolatingFIR:theInterpolatingFIR|	142 (138)	546 (384)	0
6	MCC150_TransceiverBPSK_Magnitude1:theMagnitude1|	473 (471)	343 (249)	102400
7	MCC150_TransceiverBPSK_Magnitude2:theMagnitude2|	473 (471)	343 (249)	102400
8	MCC150_TransceiverBPSK_Modulator:theModulator|	10 (10)	13 (12)	0
9	MCC150_TransceiverBPSK_PRBSGenerator:thePRBSGenerator|	19 (19)	22 (18)	1024
10	MCC150_TransceiverBPSK_Scale1:theScale1|	69 (69)	13 (12)	0
11	MCC150_TransceiverBPSK_Scale2:theScale2|	72 (72)	12 (12)	0
12	MCC150_TransceiverBPSK_Scale:theScale|	63 (63)	12 (12)	0
13	MCC150_TransceiverBPSK_SingleRateFIR1:theSingleRateFIR1|	114 (114)	1706 (1680)	0
14	MCC150_TransceiverBPSK_SingleRateFIR:theSingleRateFIR|	114 (114)	1707 (1681)	0
15	dspba_delay:SampleDelay1|	0 (0)	3 (3)	0
3	sld_hub:auto_hub|	93 (1)	91 (0)	0
4	sld_signaltap:auto_signaltap_0|	328 (2)	1251 (150)	153600

Figure 1 Magnitude utilization

2. The *sqrt* block is found to consume the maximum number of resources inside the magnitude subsystem.

3. The sample index shifts up and down because of the increment and the decrement which happens just to find the best sampling point. Considering *i* as the index, if the magnitude A{i-1}>A{i}, we increment the index and if A{i+1}>A{i}, we decrement the index. So if A{i} is larger than both A{i-1} and A{i+1}, the sample index can be kept constant. This can be done by adding the decrement subsystem to the output of the increment subsystem in the *update sample index* subsystem and creating a new magnitude and average magnitude subsystem to calculate A_(i-1).

VIII. SYNTHESIS AND VERIFICATION IN QUARTUS PRIME

The HDL code of the transmitter which is generated using the DSP builder is loaded into Quartus prime for hardware synthesis. Symbol files are created for the transmitter and then the updated transceiver block is connected to the AD9361 controller. Then the design is compiled, and the compilation report is studied for design utilization, timing results and other design statistics.

IX. IN-SYSTEM SOURCES AND PROBES

The In-system sources and probes editor facilitates us to control the internal FPGA signals from Quartus Prime. The *Altera In-System Sources & Probes* IP block is then added and renamed to *Debug source* which works as a signal source and as a probe. This allows us to analyse the internal device signals. Once the design is uploaded to the FGPA, the signal tap tool can be very helpful in debugging. The signal can be tapped to study the output waveform and then later processed in Matlab and the analysis can be done.

X. POST LAB HOME ASSIGNMENT

1. The captured signals are shown in the image below.

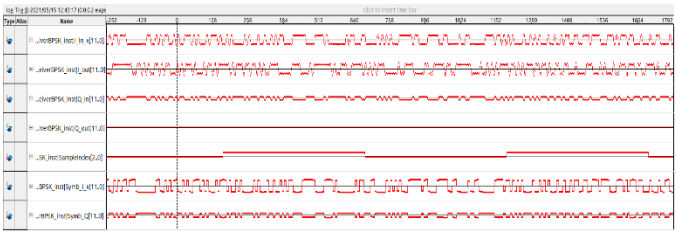

Figure 2 Signals captured in Quartus prime

The signals captured are I_in, I_out, Q_in, Q_out, Sample Index, Symb_I and Symb_Q. I_in, I_out, Q_in, Q_out represents the In-phase and quadrature signals of the data (real and complex parts) respectively. The Sample Index signal changes continuously as the sample points are updated within

a set window. The Symb_I and Symb_Q are the outputs from the decimation block.

2. The eye diagram of the received signals is as follows.

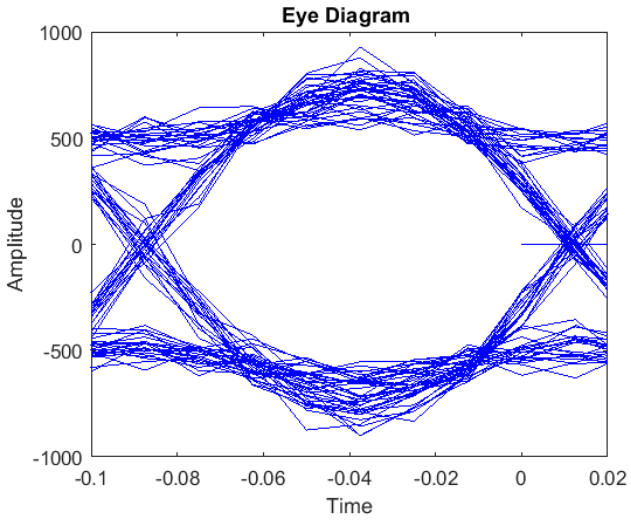

Figure 3 Eye diagram of received signals

The signals after symbol timing recovery are also added as follows.

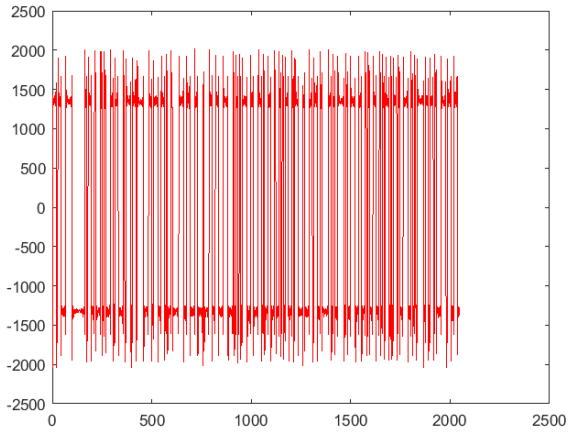

Figure 4 I_in signal as a function of time

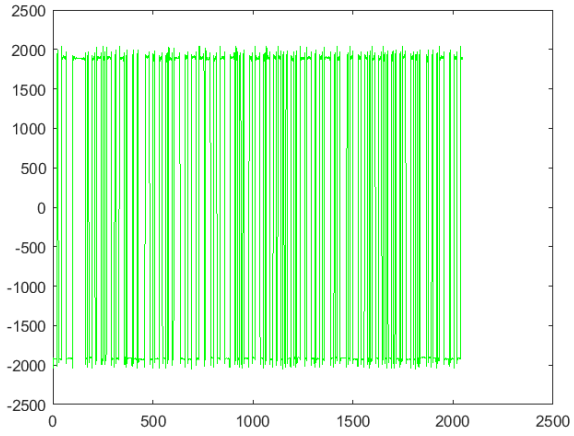

Figure 5 I_out signal as a function of time

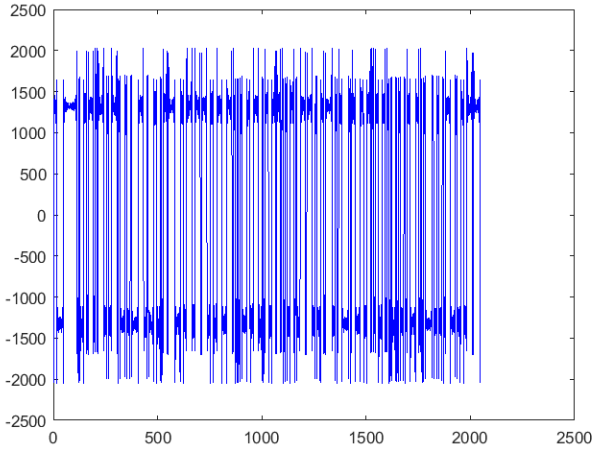

Figure 6 Q_in signal as a function of time

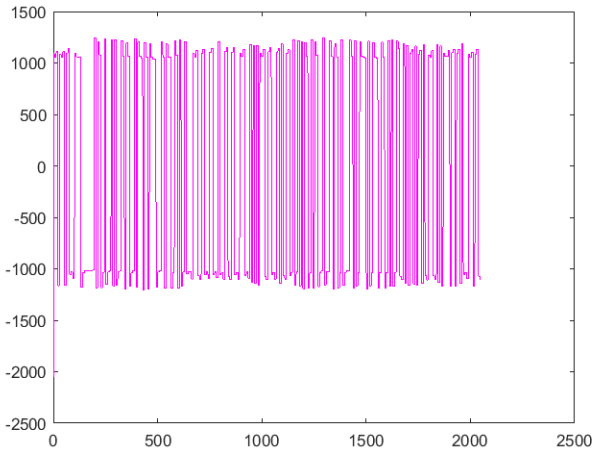

Figure 7 symb_I signal as a function of time

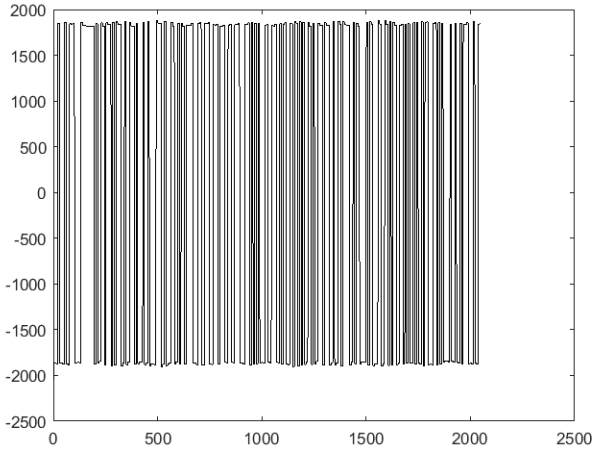

Figure 8 symb_Q signal as a function of time

3. The sample index is plotted as a function of time as follows and it is found to match the simulation.

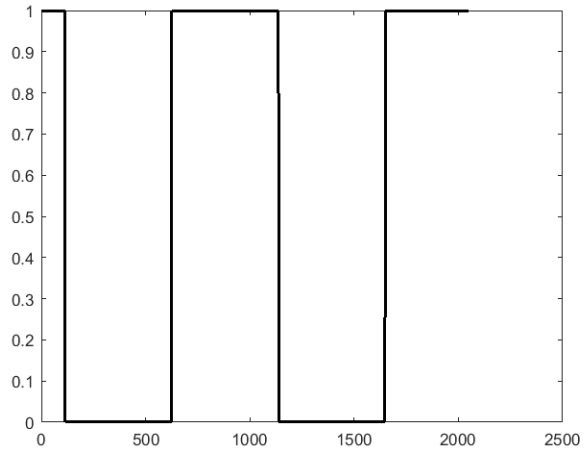

Figure 9 sample index as a function of time

4. In both the magnitude subsystems, the *sqrt* block was found to consume the highest number of resources. Hence, the *sqrt* block was replaced with *Abs* block from the primitives basic blockset. There was drastic reduction in the resource utilization (LUTs, multipliers and memory bits) as observed in Simulink.

5. If A_i is larger than both the other average magnitudes, the index is kept the same. If it is not, i-1 or i+1 is selected depending on which of them have the largest amplitude so the sample index switches value and then remains unchanged.

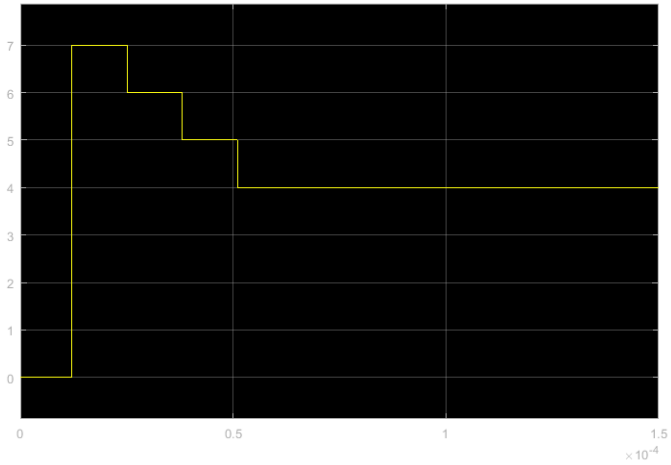

Figure 10 Sample index kept constant at 4

6. Higher the choice of the value of the length of the averaging window, the quality of the STR estimation is low and the resource utilization of the system is also low and if the averaging window length is low, both the quality of STR estimation and the resource utilization will be high, resulting in better accuracy.
Also, the SNR of the input signal affects the best choice of the averaging window length since a reduction in the value of SNR will increase the inter-symbol interference.