codex CONFIDENTIAL MEMORANDUM

TO: List

DATE: January 14, 1985

FROM: Shahid Qureshi

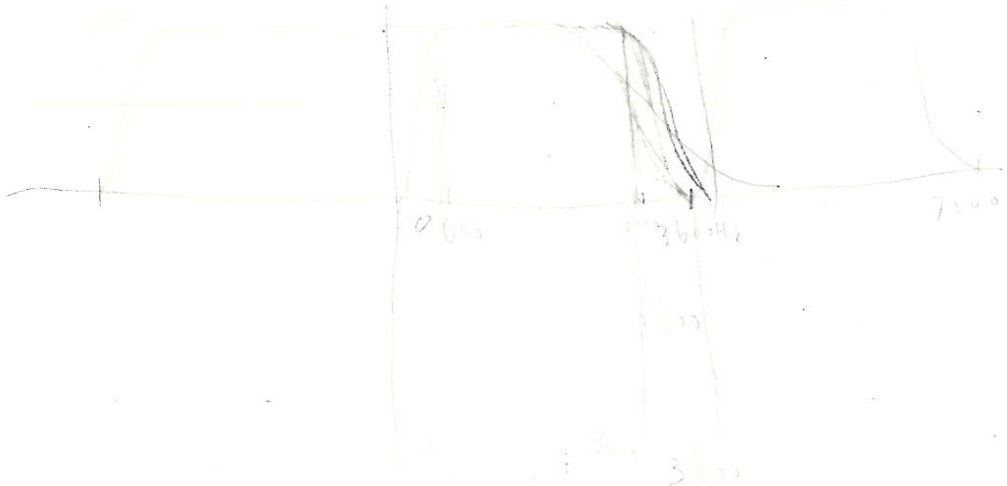
SQ: 85-02

SUBJECT: Note on Reduction of Future V.32 Analog Circuitry

Attached are handwritten notes on a subject that I have been contemplating off and on (in my spare time) for the last few months. Please let me know if you have any questions. I can review the subject with all of you if you so desire. Comments are solicited.

/llf

List: H. Chalmers
L. Gonzalez
D. Kline
F. Ling
J. Pasco-Anderson
J. Payton
B. Schissler
M. Sridhar



stdl

Jan. 13, 1985

On Reducing Future V.32 Analog Circuitry

1. TRANSMITTER

If we increase the output sample rate from 7200 samples/second to 14400 samples/s, the analog low-pass filter required to satisfy Telco specifications on out-of-band signal power is considerably simplified. Taking into account the higher sampling rate and the sample-and-hold characteristics, the repeated spectrum in the 11400 to 13800 Hz band will be at least 12 dB lower than the main band signal. A third or fourth order Chebyshev filter can then provide adequate additional rejection of the out of band signal. This in turn will help eliminate or reduce the switched-capacitor filter noise (which in the present implementation results from the two cascaded PCM switched-capacitor filters). The effect of D/A quantization noise is also reduced by 3 dB.

The proposed increased sample rate can be implemented in one of two ways. The first, straightforward, method is to simply extend the present TX

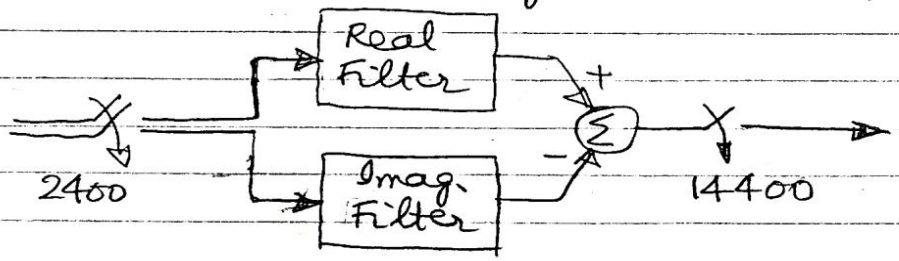


Fig. 1 TRANSMITTER (Alternative 1)

TX digital structure as shown in Fig. 1. The number of filter coefficients would then at most double from 24 to 48 and the number of multiply accumulates (MACs) for the transmit filter would increase from 48 per symbol to 96 per symbol. It may be possible to get away with a filter span of 6T or 36 coefficients (rather than 8T or 48 coefficients) when the filter is redesigned for the higher sampling rate.

An alternative approach is to use the structure shown in Fig. 2 which uses an interpolating filter to achieve the higher sample rate.

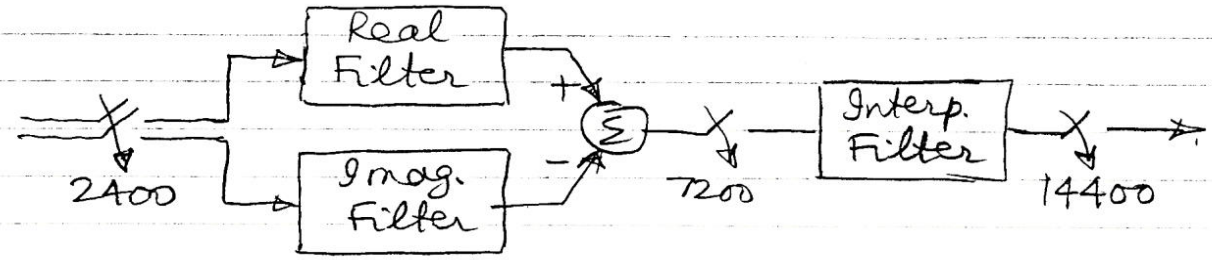


Fig. 2 (Alternative 2)

Simple rules of thumb suggest that an 8 coefficient interpolating filter at the sample rate of 14400 will be sufficient. This implies an additional computational load of 24 MACs per symbol interval and a 4 unit real delay line to store the 7200 rate samples.

Note that this 24 MAC additional computational load is only 5% of the computational load required to compute just the echo canceller outputs per symbol.

2. RECEIVER

The proposal given below for the receiver represents a more dramatic departure from our present thinking and, therefore, requires careful analysis and some confirmation by simulation before implementation is contemplated.

The idea is to eliminate <u>all</u> receive analog circuitry except a simple front-end bandpass analog filter, a sample-and-hold and an A/D converter with at least 12 to 13 bit accuracy. Fig. 3 shows a simplified block diagram. (The hybrid and DAA circuitry is not shown). Availability of low-cost audio 16-bit

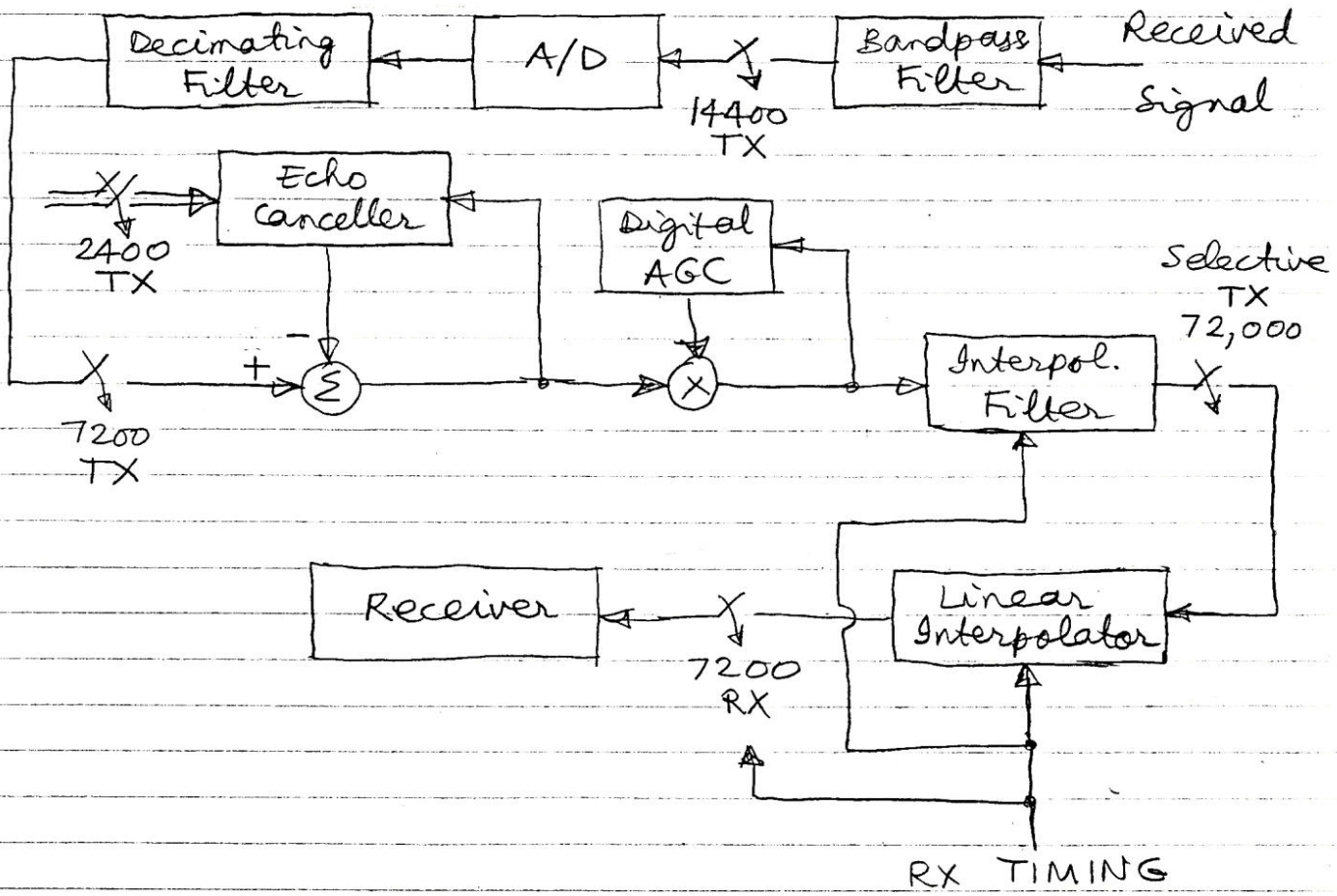


FIG. 3 RECEIVER

A/D converters is encouraging.

At first glance, the computational complexity of the proposed scheme may seem prohibitive. However, please bear with me; read on.

A/D Precision Requirement:

As in the transmitter, we propose to simplify the receive bandpass filter by increasing the sample rate to 14400. Let us assume that the maximum expected received signal level (including echo) is about 35 dB greater than the minimum received signal level at which the receiver is expected to operate properly. Let us also assume that the peak-to-

RMS ratio of the signal at the A/D input is at most 4 or 12 dB, and a $\sqrt{2}$ to 1 margin (3 dB) is desired to account for dc bias and AGC head room. Further, let us require a minimum received signal to quantization noise ratio of 30 dB. Thus, we must have an A/D converter such that its RMS quantization noise is $35+12+3+30 = 80$ dB below its peak or absolute maximum output. Assume an A/D converter which is accurate to $\pm 1/2$ LSB and has B bits. Then its RMS quantization noise is $(1/\sqrt{12})$, assuming the quantization noise is uniformly distributed between $+1/2$ and $-1/2$. The peak output of the A/D is of course:

$$2^{B-1},$$

where we have taken into account the fact that the B bits include the sign bit. Thus, the peak-signal-to-RMS-quantization noise ratio is given in dB by

$$6(B-1) + 10{\cdot}8 = 6B + 4{\cdot}8 \text{ dB},$$

which is 76·8 dB for a true 12-bit A/D. Since this is 3·2 dB lower than the desired 80 dB value, an A/D converter which is accurate to 12 bits is nearly adequate when we account for the higher sampling rate.

Decimating Filter:

The decimating filter generates a single output sample for every 2 received samples and has a nominal bandwidth of 3600 Hz, with significant rejection of noise (including quantization noise above 4000 Hz). Such a filter can be designed with 8 coefficients requiring $8 \times 3 = 24$ MACs per symbol interval. The decimation filter gives us an additional 3 dB rejection of quantization noise (assuming it is white).

Digital Subtraction:

The echo canceller uses digital subtraction to determine the residual to be used for coefficient update. This provides considerable freedom during

(half-duplex) startup to control step-size to obtain rapid convergence. Steady-state performance should also be improved because quantization of the error signal is no longer a factor.

Digital AGC:

The receiver analog AGC is substituted by a digital multiplier and, as usual, digital gain control. The placement of the receive AGC before interpolation and resampling is important to preserve accuracy.

Digital Interpolation:

Now the crux of the proposed approach, i.e. replacement of analog interpolation and resampling. The idea is as follows:

First, interpolate the 7200 sample rate sequence to produce a signal with a sample rate 10 times higher, i.e. 72000 samples/s. We estimate that for good interpolation a filter with 40 coefficients spaced at the 72000 sample rate is required. The received signal samples are obtained by linear interpolation between the two samples on each side of the desired receive sample. This is shown graphically in Fig. 4 using a 1800 Hz sine wave as an example.

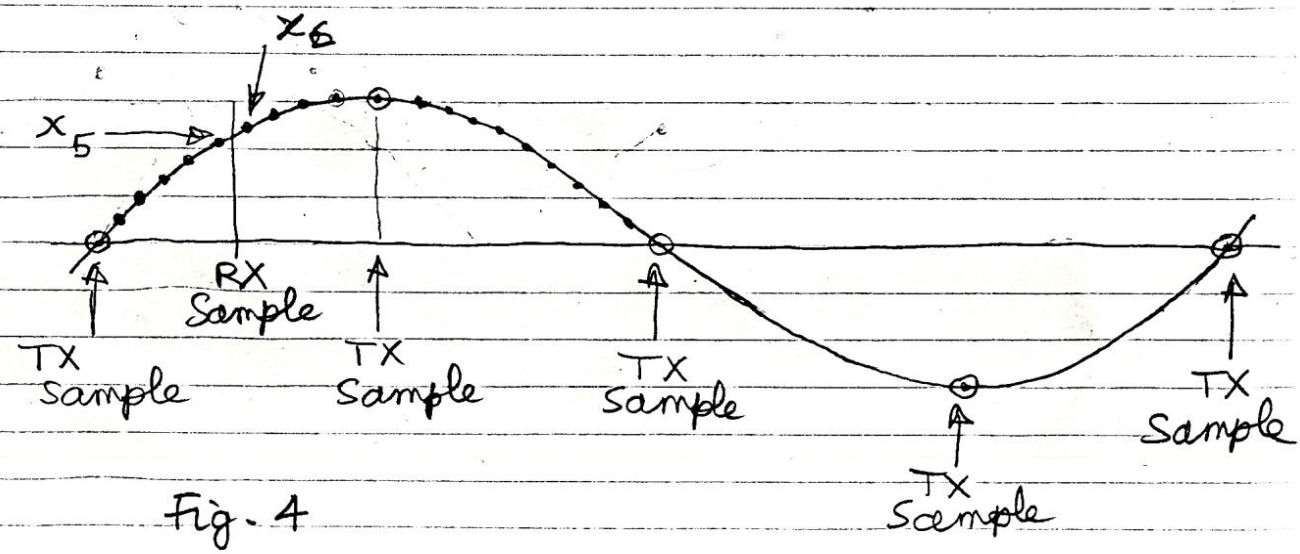


Fig. 4

Thus, in order to compute the receive sample shown in Fig. 4, we use the relation

$$\delta x_5 + (1-\delta) x_6$$

where the receive sample was to be computed at $t = (5+\delta) T_s$, where $T_s = 1/72000$, from the transmit sample. Note that in resampling the waveform, generation of all samples of the 72000 samples/s sequence are not required. Once we know the desired adjustment in the receive sampling compared to the transmit sampling, we determine which two output samples we must compute for linear interpolation of the receive sample. Generation of each 72000 rate sample requires 4 MACs. Since 6 such samples must be produced each symbol, and 2 MACs used for linear interpolation, the raw additional computational load is $4 \times 6 + 2 \times 3 = 30$ MACs per symbol. However, a number of additional steps are required to determine which 72000 rate samples to produce. Therefore, the additional load for interpolation is probably 60 steps, which is still extremely small given the benefit of replacing all that analog circuitry! The control structure for implementing this scheme is complex and needs to be looked at carefully.

We estimate that linear interpolation between adjacent 72000 rate samples to determine receive samples will introduce a noise floor about 50 dB below the received signal. This should be verified by careful analysis. (We have used a similar method in simulations without any problem.)

Finally, note that digital subtraction and digital interpolation can be implemented independently. That is digital subtraction can

be implemented with analog interpolation, and digital interpolation can be implemented with analog subtraction. In the latter case, AGC must still be before digital interpolation.

We alluded to the complexity of the control structure of digital interpolation earlier. This point needs some elaboration. In order to permit independent transmit and receive clocks, including slightly different frequencies, the shifting of new samples into the interpolating filter delay line must be controlled based on the receiver sample timing information. For instance, if the receive sample interval is on the average shorter than $10T_S$, where $10T_S$ is the transmit sample interval, then every so often a new input sample will not be required to produce an output sample. Similarly, if the receive sample interval is on the average longer than $10T_S$, then once in a while an extra input sample will be needed to produce an output sample. Of course, if a FIFO sample queue is set up at the input to the digital interpolation section, then these adjustments will be absorbed without problem. This is simply because if in fact the receive clock frequency is slightly higher than the transmit clock frequency, then in a given amount of time fewer transmit/echo canceller tasks are generated and performed resulting in fewer samples for the receiver to operate on. And vice versa.

As usual I am interested in your comments.

CONFIDENTIAL

codex MEMORANDUM

TO: List

DATE: February 11, 1985

FROM: Fuyun Ling F.L.

SUBJECT: Implementation of Digital Interpolation

This memo provides a practical design of a digital interpolater, which can be used for future V.32 modems as proposed by Shahid (Jan. 14, 1985 memo) and Dick (Jan. 17, 1985 memo).

It is shown in this memo that the digital interpolation stage (including AGC and PLL) requires about 160 - 180 SPE cycles, and hence it is feasible using the present hardware.

Please give me your components and suggestions. I wish to implement it for lab testing to see whether it will be included in Gen. 2 V.32 design.

/ej

List:

H. Chalmers
L. Gonzalez
F. Ling
J. Pasco-Anderson
J. Payton
S. Qureshi
B. Schissler
M. Sridhar

On Implementation of Digital Interpolation.

Implementation of digital interpolation for deriving receiver input (at receiver clock) from echo canceller output (at Tx clock) consists of 3 parts.

1. A second order PLL to track the timing difference between Tx and Rx clocks.
2. Digital AGC.
3. Digital interpolation.

A block diagram is given in Fig. 1. Since Part 3 is the most tricky part. I will first discuss it in detail.

1. Digital Interpolation and Control.

A. Digital interpolation filter.

The interpolation filter is a low-pass filter with a higher output rate than input rate. If we want to increase the sampling rate by L, we must interpolate L-1 new sample values between each pair of original sample values. The input signal of the interpolation filter is the original signal "filled in" L-1 zeros between each pair of its samples. The new input signal and its spectrum is given in Fig. 2.

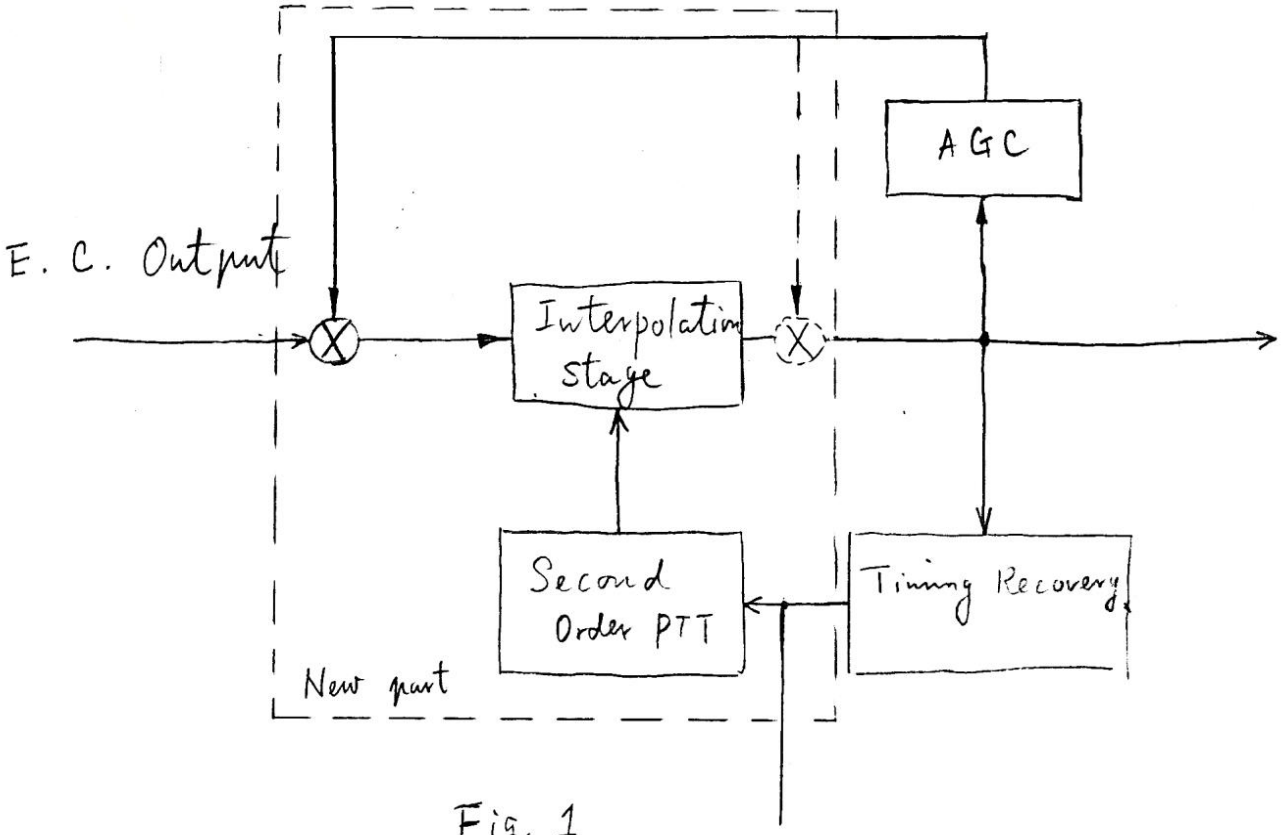

E. C. Output
AGC
Interpolation Stage
Second Order PTT
Timing Recovery
New part

Fig. 1

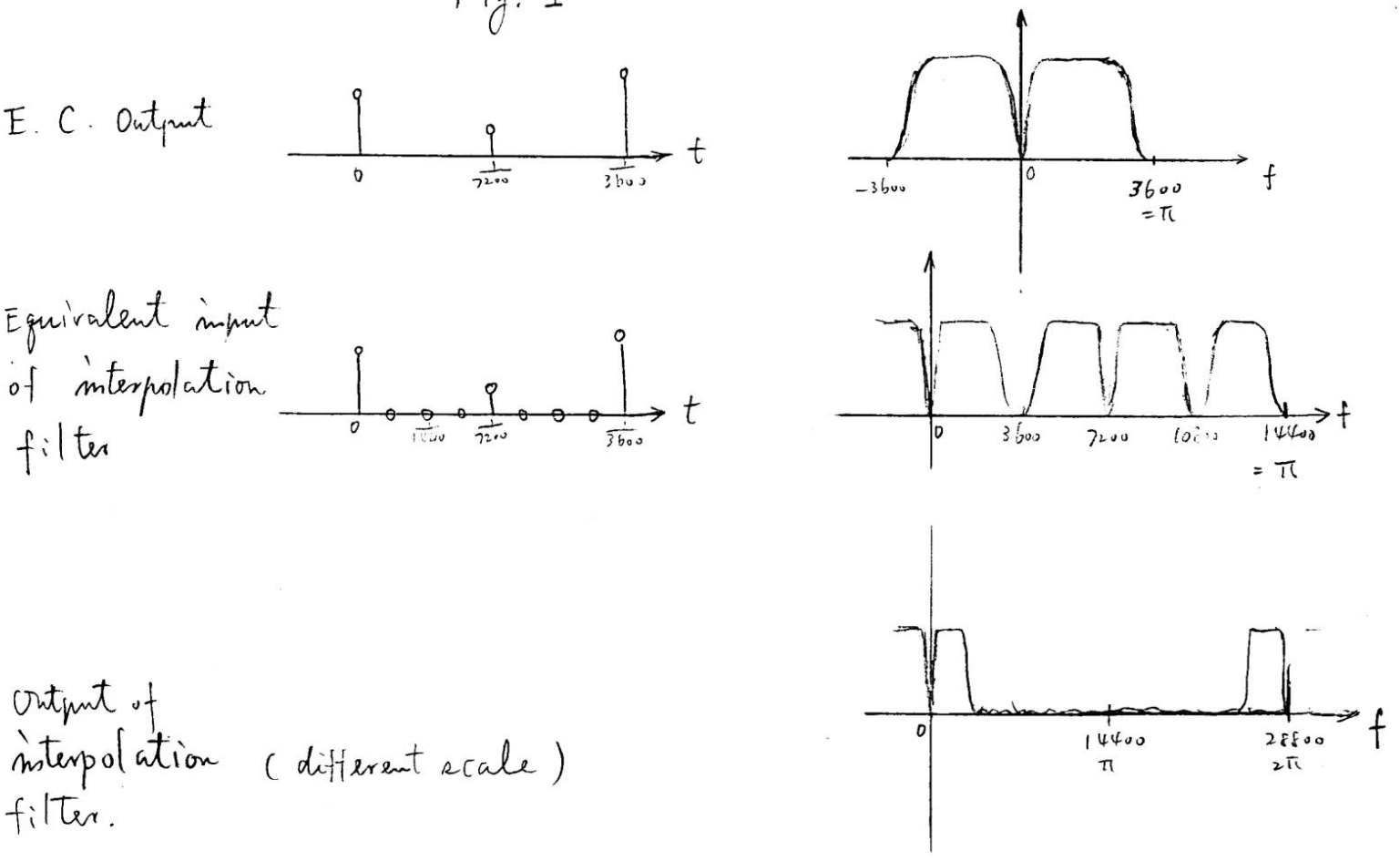

E. C. Output
Equivalent input of interpolation filter
Output of interpolation filter (different scale)
t
f
0
7200
3600
-3600
3600 = π
14400
28800
2π

Fig. 2

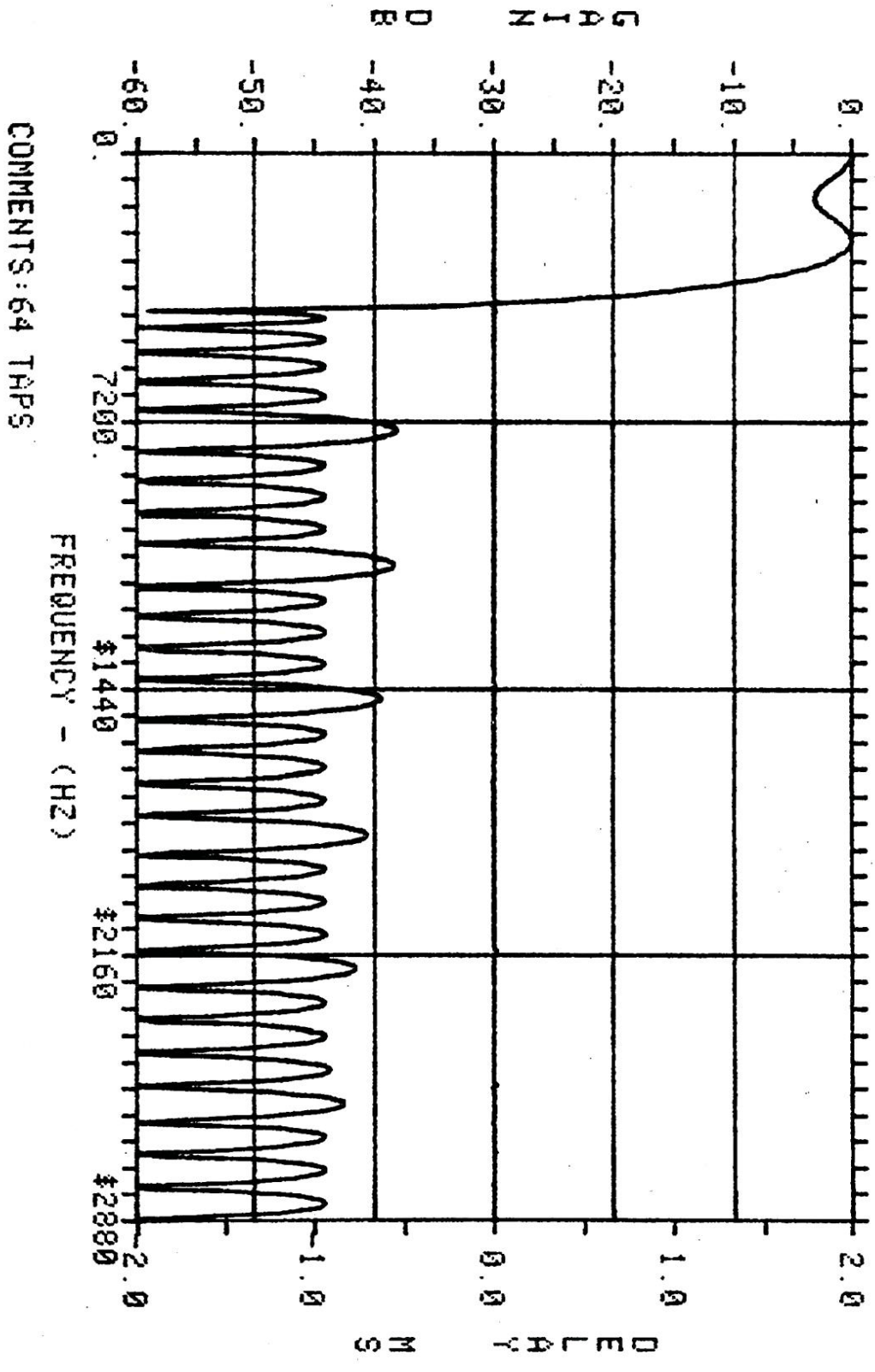

FREQUENCY RESPONSE - MAG REF 1.47 DB
GAIN DB
0.
-10.
-20.
-30.
-40.
-50.
-60.
2.0
1.0
0.0
-1.0
-2.0
DELAY MS
0.
7200.
$1440
$2160
$2880
FREQUENCY - (HZ)
COMMENTS:64 TAPS

Fig. 3c

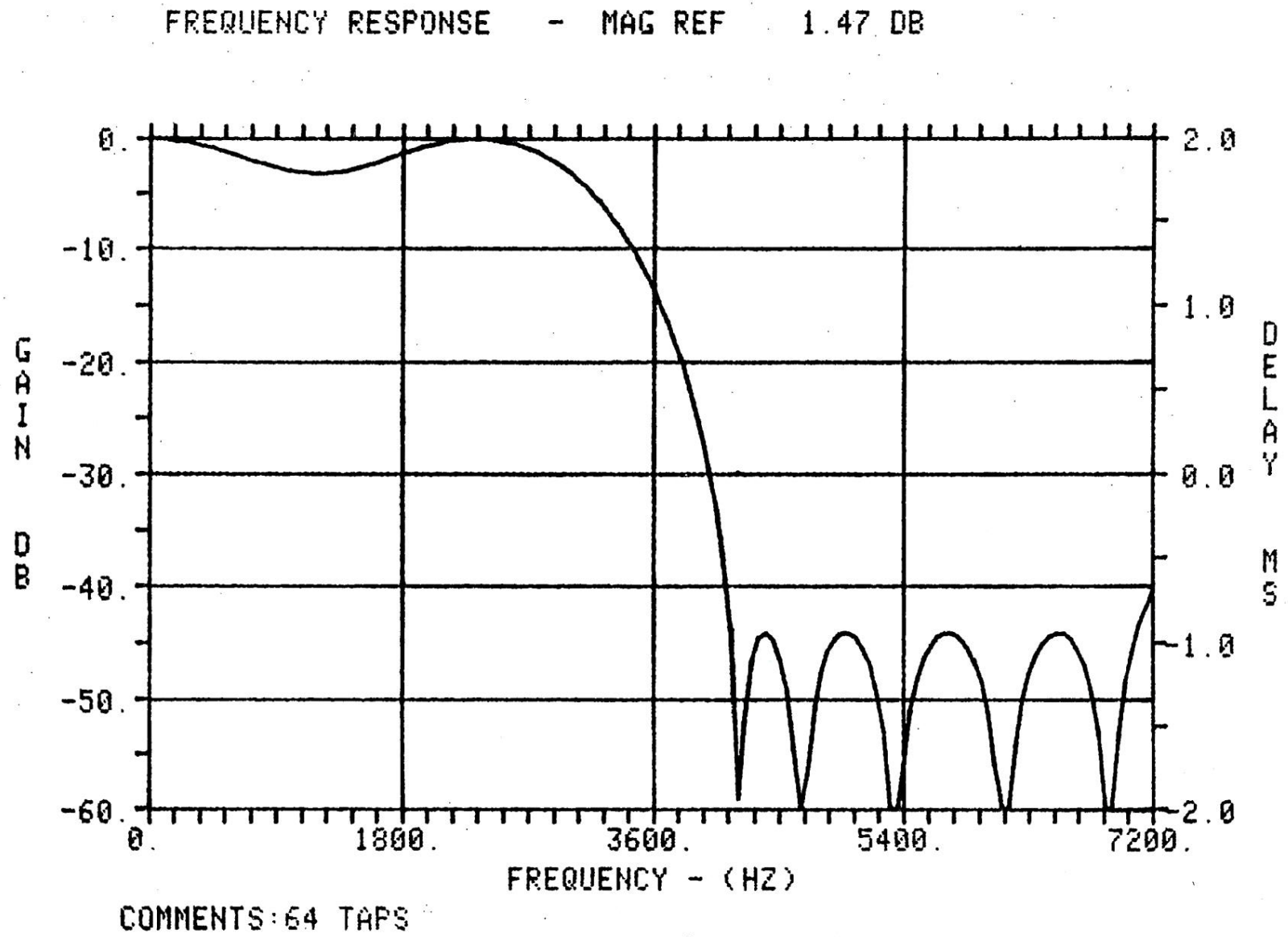

FREQUENCY RESPONSE - MAG REF 1.47 DB
0.
-10.
-20.
-30.
-40.
-50.
-60.
GAIN DB
2.0
1.0
0.0
1.0
2.0
DELAY MS
0.
1800.
3600.
5400.
7200.
FREQUENCY - (HZ)
COMMENTS:64 TAPS

Fig. 3b

The perpose of the interpolation filters is to remove signals whose frequency is beyond 3600 Hz. Hence, it is basically a low-pass filter. From the analysis of the linear interpolator given later, it is shown that $L = 8$ is enough for our perpose. Since $8 = 2^3$, we can simplify the control structure.

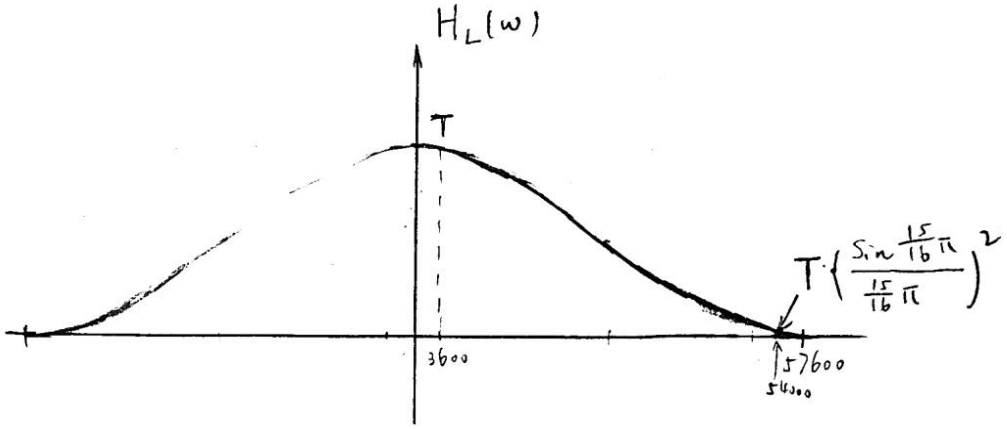
Using the FIR filter design procedure based on the Remez exchange algorithm, we designed a 64-Tap linear phase low-pass filter. Its frequency response is shown in Fig. 3. The maganitude in the stop bands is less than 44 dB. Consider the summing up effects of the multiband aliasing signals. The resulting noise may 5-8 dB higher. However a noise that is -36--39 dB below the signal will not introduce noticible effects. The filter spans 8 samples.

B. Linear interpolator.

The frequency responce of a linear interpolater is

$$H_L(\omega) = \frac{1}{T}\left(\frac{\sin \omega T/2}{\omega/2}\right)^2$$

and depicted in Fig. 4



The distortion introduced by the linear interpolater is mostly contributed by the aliasing near the frequency of 57600 Hz, and its amplitude is less than

$$\left(\frac{\sin\frac{15}{16}\pi}{\frac{15}{16}\pi}\right)^2 \doteq 0.0044 \longrightarrow -47\ dB$$

(Based on $L = 8$)

Comparing it with the aliasing introduced by the interpolation filter, its effect is negligible. This justifies our choice of $L = 8$.

(If we do not use linear interpolation but a denser filterbank, in order to obtain a similar result, we need a 160 coefficients filter).

C. Control Functions :

The samples from the echo canceller is fed into the receiver input buffer at the Tx sample rate we need to store at least 8 samples for the interpolation filter. Additional samples may also need to be stored due to the asynchronism between the transmitter and receiver operations. A 16 sample long buffer is definitely sufficient.

Normally, a pointer is incremented 3 times each baud to point the starting data of the interpolation filter. It may have to be adjusted ±1 samples due to the "sample slipping" between Tx and Rx samples. As it will be explained later.

The phase locked loop for time recovery provides a phase difference between the Tx samples and Rx samples. In the implementation I have in mind, this phase difference will be represented as a positive number (0000 → 7FFF). The MSB will always be zero. The next three bits determine which two subfilters are to be used (0 → 7). The last 13 bits is used for linear interpolation.

After the PLL modified the phase error, if the phase error is still positive, we need only to increment the pointer three times, then do the filtering and Linear interpolation as described before. If the phase error(PE) becomes negative. There are two possibilities:

1. PE is close to 7FFF before and becomes after modification.

AX AH
000 8 XXX

2. PE is close to 0000 before and becomes after modification.

AX AH
FFFF FFF

In case 1 we need to increment the delay line pointer and in the second case we need to decrement the pointer by one sample. In both cases we reset the MSB of PE.

It can be shown the first case corresponding to to receiver clock is slower than the transmitter clock, and the second case is just opposite. This implies PLL output should be positive when Tx is faster than Rx and vice versa.

A simplified flowgraph is given below

Rx Pointer RP

Phase Error PE

Pointer for interpolation filter. IFP

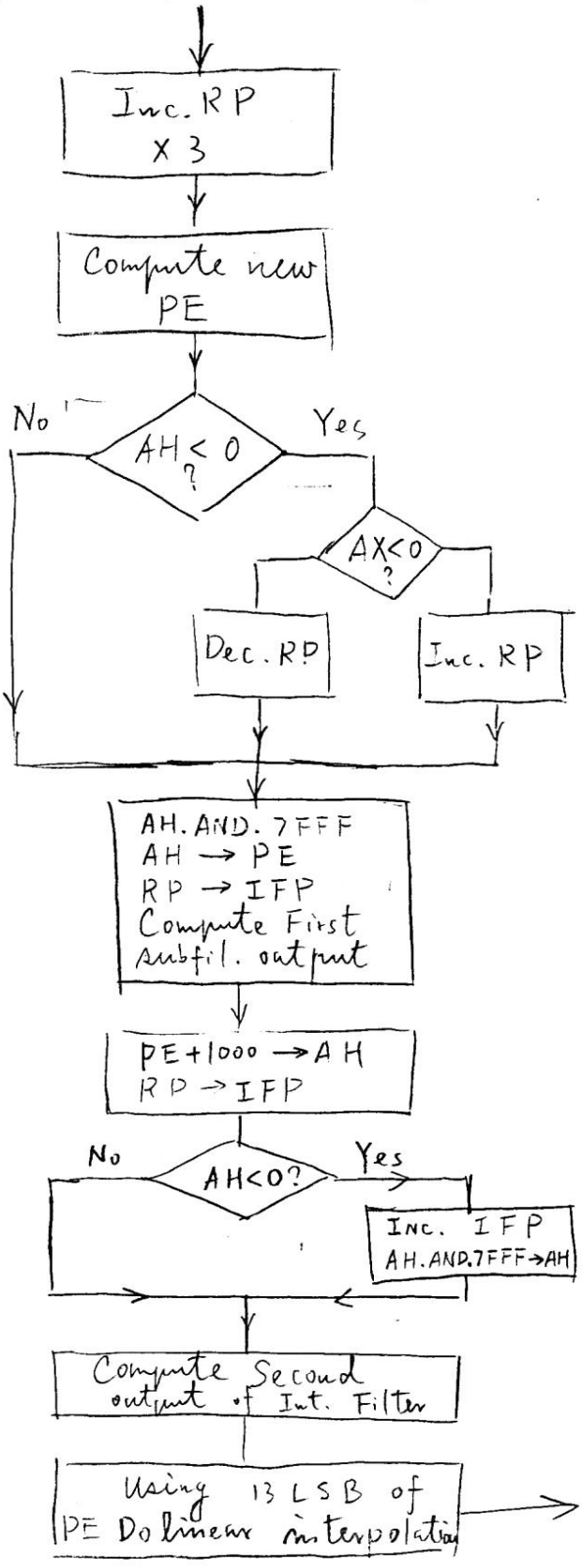


Fig. 4

The interpolation filter takes 10 cycles/coefficient to compute the two subcanceller outputs. If we use 8 coefficients for each subcanceller, 80 cycles are needed. Linear interpolation and control will use about 40 cycles. Total of 120 cycles are required for interpolation stage.

2. PLL.

The input of the timing recovery PLL is the output of the timing recovery. The PLL is a typical second order one. It fuction is.

$$F(K+1) = F(K) + C_1 \, Sgn(T(K+1))$$

$$P(K+1) = P(K) + C_2 F(K+1) + C_3 \, Sgn(T(K+1))$$

where $T(K+1)$ is the output of the timing recovery output $P(K+1)$ is PE described before. Since $0 < PE < 32768$, 300 change in PE will corresponding to 1 μs, hence. C_3 should not greater than 300 in normal operation. The value of C_2 depend on how large is the expected frequency differency between Tx and Rx clock. $C_2 = 40$ corresponds to 1 Hz difference. (We may need double precision or averaging for it).

3. AGC.

The Digital AGC will use the same input as the analog AGC. Since the multiplication always reduces the magnitude. We need to choose proper scaling factor. Shift to left may be needed after multiplication to retain the signal accuracy.

PLL ad AGC need about 30 – 35 Cycles.

4. Globel control.

Due to the different Tx and Rx sampling rates the interpolation filter may need to output more (or fewer) samples than its input. This raise the problem when to start receiver tasks. It can be solved in two ways

A If the present PTT exists, we can use the same interrupt from PTT to start Rx tasks. Since the PTT and the timing PLL should be cynchronized, if we buffer the echo canceller by one or to bands, we will not run out of input samples.

B The reicever task can run freely. It will wait when the echo canceller cannot provide more samples. In this case the output of the received has

to be buffered and a bit clock has to be generated (at remote Tx rate) to fan out the received data. In a custom design, we may simplify the receiver PTT function by using the output of the timing PLL.

If the timing recovery works properly (it should) we will not have problems in both cases.

The output of the echo canceller should send to receiver transparent to the receiver tasks is the first scheme is used.

5. Conclusions

A. Digital interporation is feasible under the present SPC hardware.

B. It will take about 160 extra cycles to the receiver SPC.

C. The noise introduce due to digital interpolation will in the order of -35 — -40 dB below the Rx signal level. It can be improved by increase the order of interp. filter (increase the number of coefficients to nine will reduce noise by 6 dB at the cost of 10 more cycles. Compare Fig. 5 with Fig. 3)

D. The performance should be justified from the real-time implementation.

E. Please review the rough design given above. If it is principally correct, we can work on more details and implement it.

FREQUENCY RESPONSE - MAG REF 1.24 DB
GAIN DB
0.
-10.
-20.
-30.
-40.
-50.
-60.
0.
2200.
$1440
$2160
$2880
FREQUENCY - (HZ)
2.0
1.0
0.0
-1.0
-2.0
MS DELAY
COMMENTS: 72 TAPS

Fig. 5a

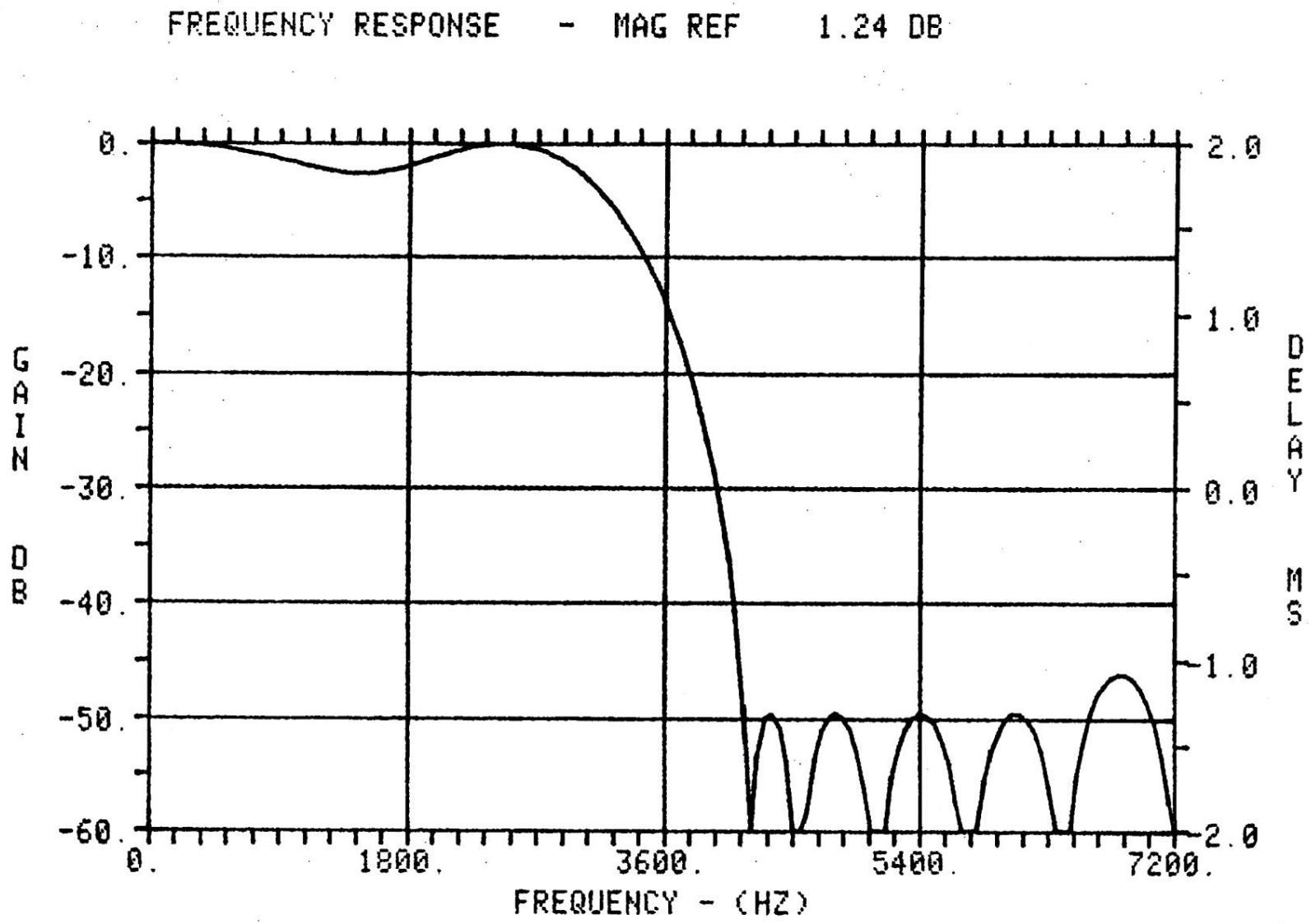

FREQUENCY RESPONSE - MAG REF 1.24 DB
0.
-10.
-20.
-30.
-40.
-50.
-60.
GAIN DB
2.0
1.0
0.0
-1.0
-2.0
DELAY MS
0.
1800.
3600.
5400.
7200.
FREQUENCY - (HZ)
COMMENTS: 72 TAPS

Fig. 5b

codex MEMORANDUM

TO: Fuyun Ling

DATE: February 22, 1985

FROM: Shahid Qureshi

SQ:85:26

SUBJECT: Timing Recovery in Conjunction with Digital Interpolation

Please review and comment on the attached note.

/llf

cc: H. Chalmers
R. Kline
J. Pasco-Anderson
J. Payton
R. Schissler
M. Sridhar

Timing Recovery in Conjunction with Digital Interpolation

I have given some more thought to the problem of timing recovery when the receive samples are obtained by digital interpolation instead of the usual S/H and A/D driven by RXSMPCLK. In this note, let us concentrate on the V.32 Gen 1 based implementation.

First, let us briefly review the problem. Samples of the echo canceller residual signals are available in the TXSPE. These samples are timed according to TXSMPCLK and can be obtained by analog or digital subtraction. If analog subtraction is used then TXSPE must also control at least a rough AGC circuit to ensure that the residual signal occupies about one half of the A/D full scale. Digital subtraction is preferable because the residual signal samples are then available as 16-bit numbers and the AGC function can be left to RXSPE. In any case, each TX symbol interval, 3 residual samples must be passed from TXSPE to RXSPE. This can, of course, be accomplished by hardware means, e.g. via a register written by TXSPE and read by RXSPE, using TXSMPCLK interrupt to RXSPE instead of RXSMPCLK interrupt. However, a software transfer through the MPU is preferable. Each TXTASK, the TXSPE can load 3 samples into the TXSPE output queue (in addition to its usual output), and the MPU can read these samples and load them into a residual sample delay line or queue in the RXSPE (this queue can replace the current RX A/D sample buffer).

The proposed method is based on the premise that it is desirable to make as little change as possible to the current system particularly the MPU state machine and software structure. Fortunately, the capabilities of the PTT allow us to do just that and still convert our system from analog to digital interpolation.

The MPU gets RXSYMCLK interrupts from the PTT and assigns an RXTASK to the RXSPE as usual. However, the MPU sends some additional information to the RXSPE through the RX task queue. This additional information tells the RXSPE the relative displacement of the receive sample time from the transmit sample time which is desired to be made. Fortunately, this information is readily available from the PTT. Each of the PTT trackers contains a so-called NETCOR counter/register. The NETCOR counter in the RX Tracker in the PTT provides a total count of adjustments made to RXSMPCLK in TRICLK periods. Each time NETCOR is read by the MPU, the count is reset to zero and the accumulation of adjustments begins again in units of $T_S/640$, where T_S is the sample interval for a sample rate of 7200.

The NETCOR counter in the TX Tracker operates in the same way. Of course, if the transmitter is operating in internal clock mode there are no adjustments made to TXSMPCLK and TX NETCOR remains zero. In this case, the MPU can obtain the relative displacement of the receive sample time from the transmit sample time by reading RX NETCOR each symbol interval. The MPU can then supply this desired adjustment to the RXSPE, through the RX task queue, for use by the RXSPE in computing the receive samples from the echo canceller residual samples by digital interpolation. If the transmitter is operating in external clock mode, then the MPU must read both RX NETCOR and TX NETCOR each symbol interval, subtract TX NETCOR value from RX NETCOR value and pass on the result to the RXSPE. The RXSPE uses this value as before to adjust the RX sample timing for digital interpolation. Further, the RX samples obtained by digital interpolation are used in the receiver as usual for equalization, timing recover, etc. The timing error computed by the timing recovery algorithm is provided to the PTT as usual via the MPU. The PTT proceeds to make adjustments to RXSMPCLK (and RXSYMCLK) keeping the count of adjustments in NETCOR. RXSYMCLK is used to interrupt the MPU which triggers RX TASK in RXSPE and supplies the net adjustment to be made to RX sample timing as described above. This closes the timing loop.

Note that in the method described above, there is no need to keep a parallel software PLL in RXSPE or to attempt to close the loop with the PTT by monitoring the depth of the residual sample delay line. The only RX PLL is in the PTT, timing adjustments made there are used to generate RXBITCLK and RXSYMCLK. These very same adjustments are also used by the digital interpolator, which in turn keeps depth of the residual sample delay line in a desired range.

Modifications and additional steps required each symbol interval are summarized below:

TXSPE

- Store echo canceller residual samples in TX output queue to MPU instead of the analog interpolation D/A buffer.

MPU

TXSYMCLK Interrupt

- Retrieve echo canceller residual samples from TX output queue in TXSPE and transfer these samples to residual sample delay line in RXSPE.

RXSYMCLK Interrupt

- Read RX NETCOR and TX NETCOR values from PTT. Subtract TX NETCOR from RX NETCOR and transfer result to RX SPE through RX task queue.

RXSPE

- Update receive sample time using the

  RX NETCOR - TX NETCOR

  value as the adjustment.

- Use digital interpolation to compute three receive samples using the sample timing computed above, and adjust each sample according to AGC gain value. (Note care must be taken here to preserve accuracy, e.g. compute interpolation filter output then shift left as appropriate before applying AGC gain value less than 1).

- Timing recovery as usual with timing error feedback to PTT via MPU as usual. Other algorithms also operate in the same way as at present. Of course, RXSPE A/D interrupts are disabled and AGC gain adjustments are performed in software.

codex MEMORANDUM

TO: List

DATE: April 18, 1985

FROM: Fuyun Ling

FL 3-85

SUBJECT: On Control of Digital Interpolator

According to the suggestion by Sri, I wrote this memo to explain how the digital interpolater works, when Tx clock and Rx clock are different. Further discussion and comments are very welcome.

/ej

List:
M. Sridhar
H. Chalmers
D. Kline
R. Schissler
J. Payton
S. Qureshi

A note on the control of digital interpolator

In this note, I try to explain how the digital interpolator (DI) works when Tx and Rx have different clock frequencies. A simplified model is used for illustration. This model is explained below.

The simplified DI operates at baud rate (the real one operate at symbol rate). It has 8 subfilter each of which has only one tap. We denote the tap value of subfilter i as $C_{i,i=0,\ldots,7}$. The input signal is $x(t)$, where t is transmitter sampling instant.

The timing PLL controls which subfilter is to be used. If Rx baud rate is faster than Tx. i will decrease from one baud to next, and vice versa.

In each receiver task, the input buffer pointer increments by one per baud, normally. If $x(t)$ was used in previous baud, $x(t+1)$ will be used in this baud. However, if i changes from 0 to 7 or 7 to 0 due to the adjustment of PLL, the input point will decrease or decrease by 1, in addition to the normal increment.

Below, we exam three possible cases. We assume $i=0$, at beginning.

① Tx and Rx has exact the same clock. There is no time slew. PLL output will not change. The same subfilter is used again and again. No problem at all.

② Tx is faster. The clock frequency of Tx is equal to 9/8 of receiver frequency. In this case, for each 9 Tx tasks, receiver only excute 8 tasks. The timming PLL makes i increase by one each baud. The output of DI is:

$C_0X_{(t)}$, $C_1X_{(t+1)}$, $C_2X_{(t+2)}$ $C_3X_{(t+3)}$ $C_4X_{(t+4)}$ $C_5X_{(t+5)}$ $C_6X_{(t+6)}$ $C_7X_{(t+7)}$ $\xrightarrow{\text{※}}$ $C_0X_{(t+9)}$.

at ※ i change from 7 to 0 (cross modulo boundary) an additional increment of input pointer occur. In this case. DI uses 9 input samples to produce 8 output

③ Tx is slower, The ration of Tx clock frequency vs. Rx clock frequency is 7/8. i decrese by 1 each baud. DI outputs are:

$C_0X_{(t)}$ $\xrightarrow{\text{※}}$ $C_7X_{(t)}$, $C_6X_{(t+1)}$ $C_5X_{(t+2)}$ $C_4X_{(t+3)}$, $C_3X_{(t+4)}$ $C_2X_{(t+5)}$ $C_1X_{(t+6)}$, $C_0X_{(t+7)}$.

at ※ i change from 0 to 7, a decrement of input pointer was made. In this case. DI outputs 8 samples by using 7 input samples.

If PTT is synchronized with timing PLL, we will have right number of Rx tasks. In the above three cases

we will have 8, 7, 9 receiver tasks, respectively. If we buffer the input samples by 1 or 2 bands, the input samples from Tx will neither be used up nor build up.

I hope I have made myself clear. If you have any questions, I am willing to discuss with you more detail.

codex MEMORANDUM

TO: **List** DATE: **July 1, 1985**

FROM: **Fuyun Ling** F. Ling

SUBJECT: **Digital Substraction/interpolation in loopback timing**

I have summarized the present implementation of digital substraction/interpolation and its problems in loopback timing mode. Since I don't have a good solution to it, and don't understand one of the problems yet, I need your ideas and comments. Please help.

/b

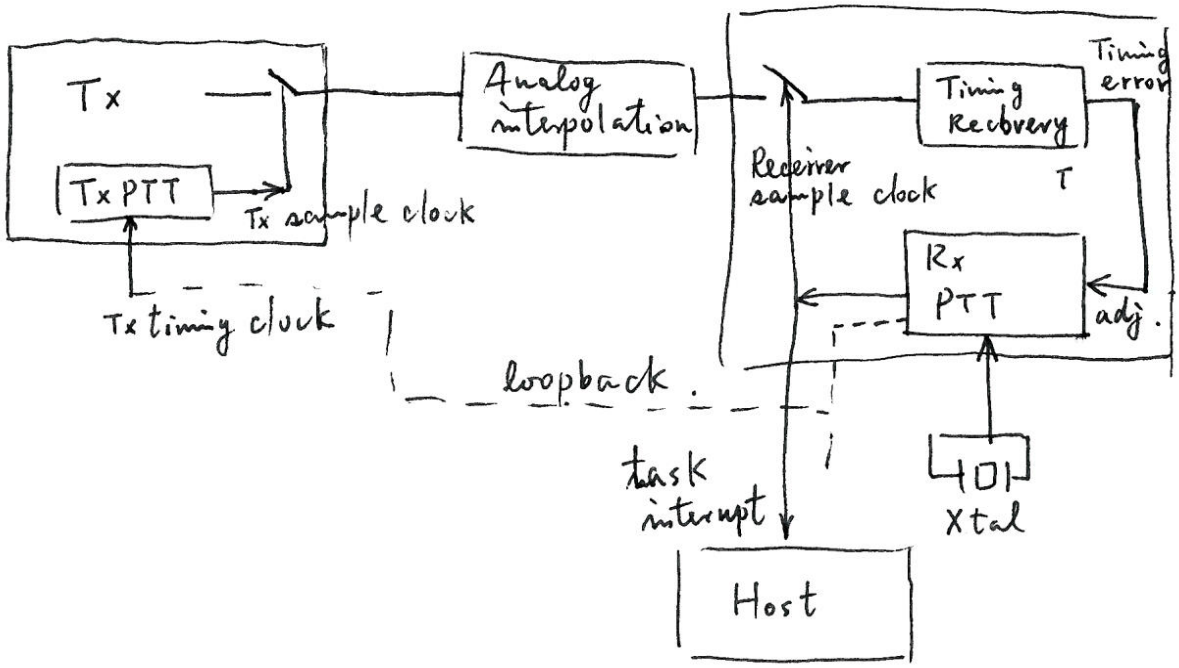
List: **A. Chorro**
J. Payton
S. Qureshi
Sri

Problems of digital interpolation — substraction in loopback mode.

Digital substraction/interpolation scheme (it will be refered as digital scheme in sequel) works fine in internal clock timing. I have not test it in external timing yet, but not foresee any problem right now. The bigest problem happens when it is in loopback timing. The implementation I am using now and the problem are described below.

1. Implementation:

A. Analog scheme (Gen 0 present structure).



2. Problems in loop-back timing.

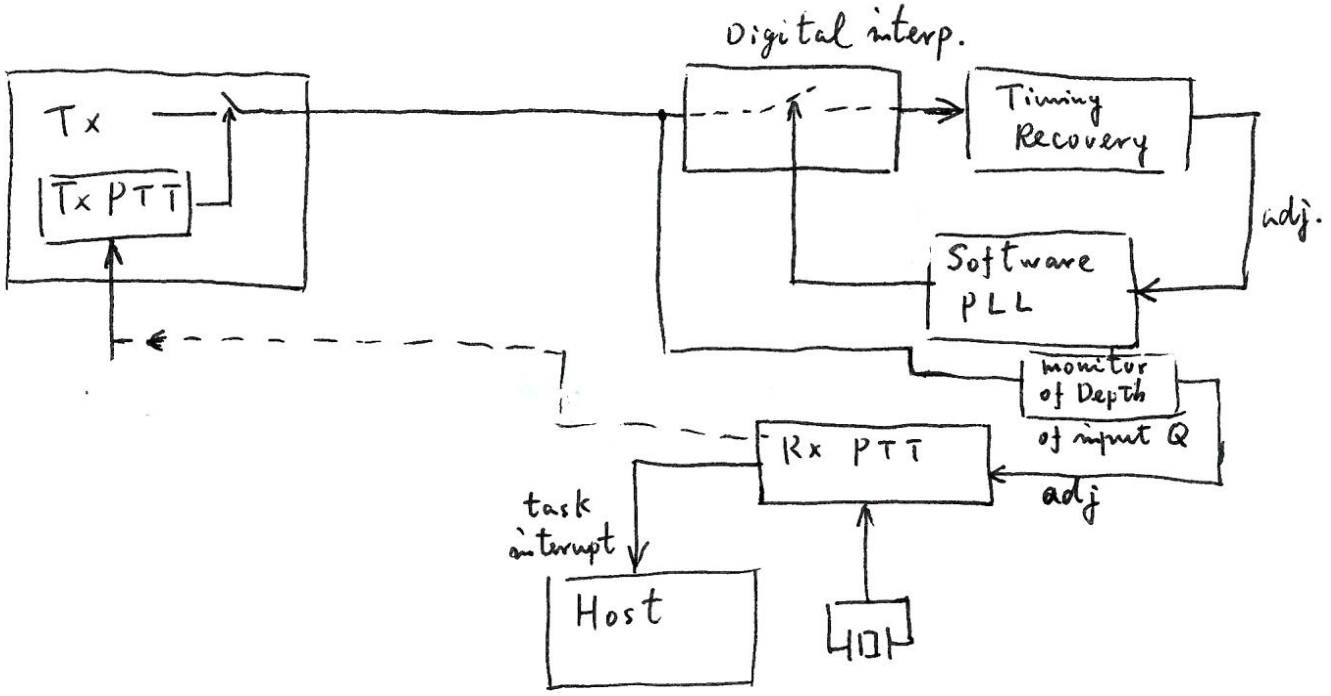
Several problems were observed.

A. In the present implementation. there is a positive feedback loop exists when loopback timing is used. Because we are using depth of input queue to adjust Rx PTT, assume receiver PTT is fast than Tx PTT, when timing is looped back. Tx PTT will run faster, as a consequence Receiver input queue has more samples coming in, which will reduce the depth of the queue, then instruct the Rx PTT even faster.

Fortunately, since the timing recovery and PLL will also detect the Tx clock speed change, and adjust the speed of PLL, then take in more Tx samples to increase the depth of the queue. This form a negative feedback loop. As a result, if the system is stable or not depends on the strength and time-constants of these two feedbacks. I have made this system stable at the cost of slowing down the adjustment of receiver PTT ever further.

B. Since the time constants of the receiver PTT loop is very long, the transmitter clock which tracks the receiver clock. will not settle down for a long time. This frequency and phase jitter affects both local and remote receiver.

B. Digital scheme (present used).



Explaination:

Since the error signal for adjust PLL is referenced to Tx sample clock, the frequency and phase that software PLL knows are all relative to the tx clock. The only absolute timing exists is the Rx PLL. However, since the error signal for adjusting Rx PTT is the Depth of input Queue. This error signal is very coarse. As a consequence, if we want to adjust Rx PTT to lock it to the remote transmitter accurately, we have to integrate this error signal, then make the time constant very long (in order of a minite). In other words we cannot expect the Rx PTT synchronized with remote signal in short time.

C. There is another mysterious problem exists, which I don't know what causes it yet.

When the modem is in answer mode. After half-duplex training, when the transmitter switched from 4800 b/s to 9600 b/s. The transmitter clock gets a phase shift. which could bee more than a sample. Both local ad remote receiver are affected badly bey this timing hit.

This phenomeon is not observed when the digital modem is in originate mode. It is not observed in the analog (Gen. 0) modems. Why?

3. Other possible schemes of implementation.

A. As John Payton originally proposed we may calculate the required adjustments of Rx PTT from the adjustments made by the timing PLL. This scheme solves problem B. However, It cannot be directly applied to external or loopback timing. In oder to use it in loopback timing, we need to modify it bey either using depth of queue control or using Tx PTT clock to

Rx PTT input (the latter cannot be done in present hardware structure). Both modifications will introduce the same positive feed-back loop as in the present implementation.

B. Open timing PLL loop in loopback timing mode. Since in the loopback timing mode the Tx and Rx PTT clocks are assumed looked together. The timing PLL may not needed when we adjust receiver PTT using timing recovery algorithm (the same as in Gen 0). The potential problem of this method is the timing adjustment may slow down some what (now the timing adjust loop is : TimRec → Rx PTT → Tx PTT → sampling phase → TimRec). On the other hand this method solves the first and second problem completely.

I have implement this scheme on the development station, the problem of slow down timing recovery responce seems not noticible. However the problem c in Part 2 still exists. In addtion, it need some softwere change, since we use different timing adjusting scheme in different timing modes.

4. I don't have a perfect solution yet. And problem C is still not clear to me. Please help and give me your ideas ad comments.

FYL → S.Q., JLP, JPA, SRI, V. Eyuboglu, H. Chalmers, D. Kline

An optimal (?) realizable (?) solution for digital subtraction / interpolation in all timing mode.

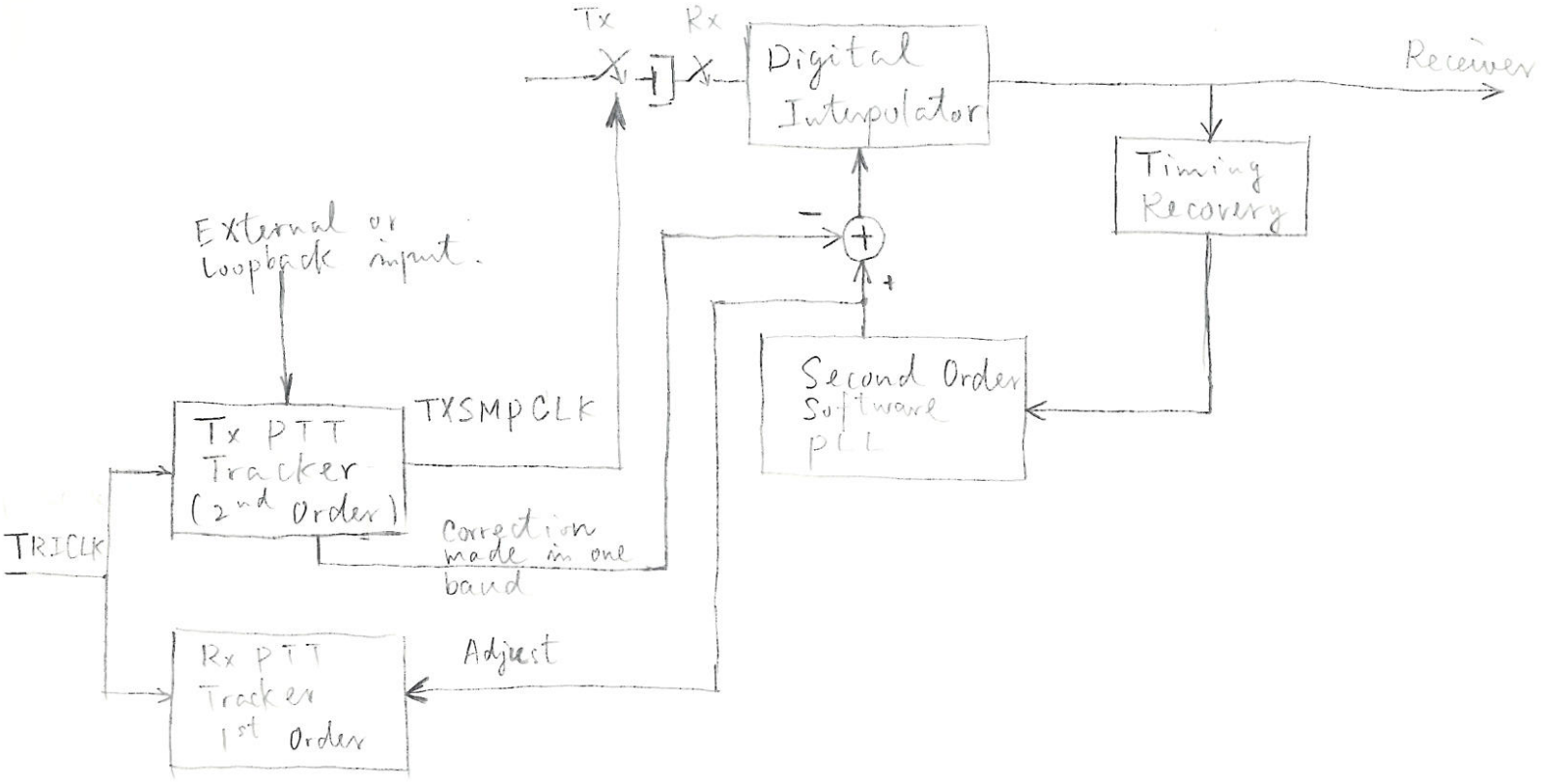
The problem occured in timing loopback mode has troubled me for more than two weeks now. There are mainly two questions need to be answered:

1. How we can solve this problem under present hardware structure (maybe not optimal).

2. Is there an optimal solution without consider the present hardware constrains.

The first question will determine if we will be able to use the digital scheme in Gen 1 V.32 products. While the second one is mainly for Gen 2.

For Gen 1, I don't have a good solution yet. John Payton / Shahid's method may work. Although it is not optimal. How good (or how poor) it works is to be seen when I implement it. In the following, I will propose another solution. It seems to me to be sound theoretically and may be realizable. I need your comments and more thought.



Explanation: The main difficulting in implementation of digital timing is that the software does not have knowledge of absolute timing. All the software timing PLL sees is relative to TxSmpClk. In internal timing mode, TXSMPLCK is directly related to the absulute timing (divide down by a fixed factor from TRICLK) hance there has no ploblem However, when in external or loopback timing mode, TXSMPCLK is no longer directly related to TRICLK,

Software PLL, as well as the Rx PTT, have no way to access absolute timing. Original we use depth of input Queue to maintain global absolute timing, however it is too coarse to accurate timing error.

In the scheme proposed above, the digital interpolator is controlled by software PLL and correction made in Tx PTT. The combined result of these two make the digital interpolator track the phase (and frequency) difference between local and remote transmitter clocks, while the software timing PLL is tracking the difference between remoted Tx clock and an absolute clock. So does in Rx PLL.

It is also easy to check that, in loopback timing, all feedbacks are negative.

The drawback of this scheme is that we have to know how much correction has been made by the Tx PTT tracker. It may be calculated by using NETCOR. The problem is how to transfer the information to Rx SPE.

codex MEMORANDUM

TO: List

DATE: August 1, 1985

FROM: Fuyun Ling F. Ling.

SUBJECT: Implementation of Digital Interpolation/Substraction

Attached is a summary of my recent work on digital interpolation/ substraction. The problem in loopback timing mode has been solved. As a byproduct the performance in external timing mode is also improved. Now I feel confident to say that it can be incorporated in Gen 1, Gen 2 V-32 or other product design. Welcome your further comments.

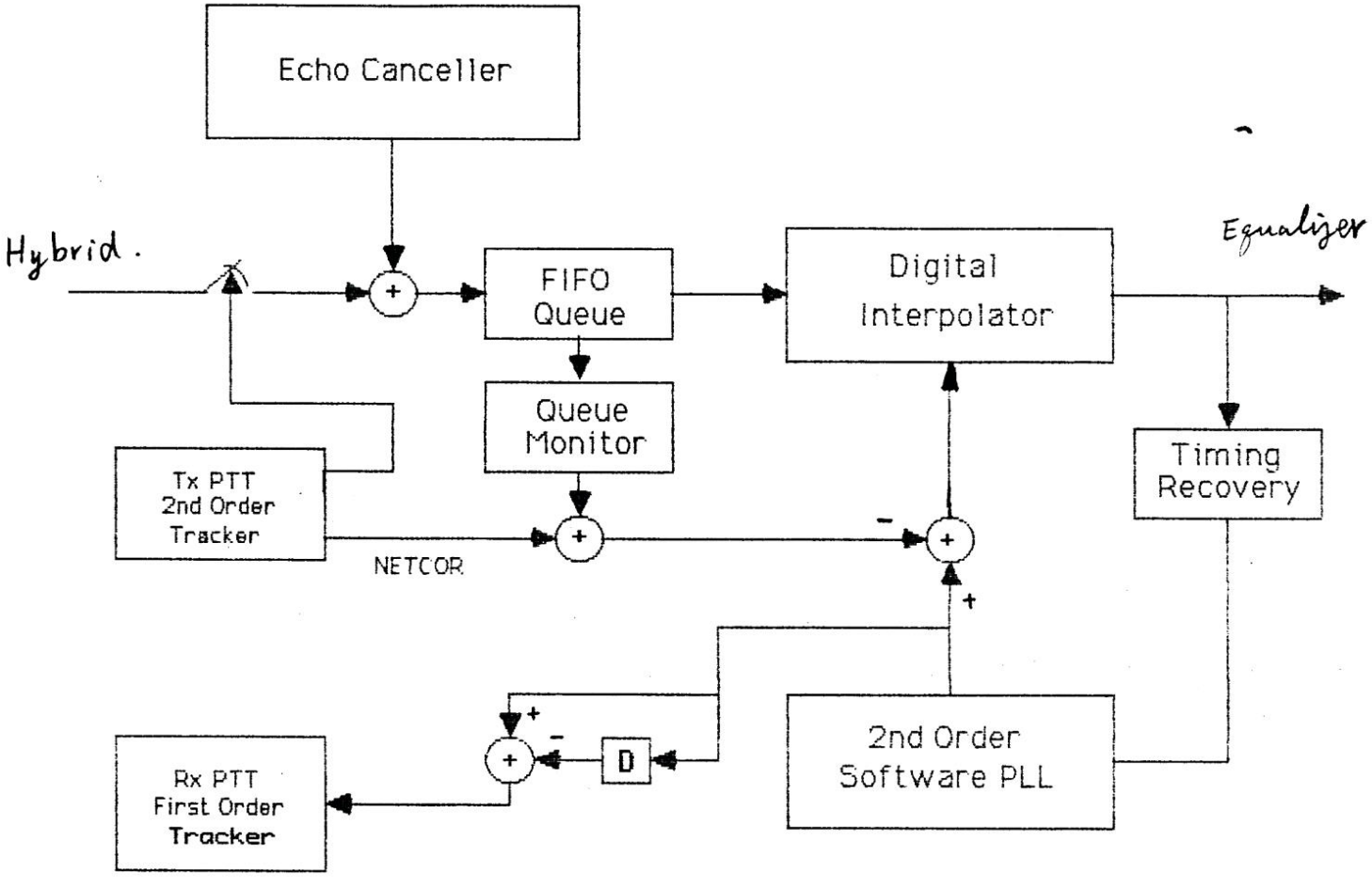
List: Harvey Chalmers C-40
Alfredo Chorro C-410
Vedat Eyuboglu C-40
Phil Giangara C-75
Mark Huntzinger C-40
Dick Kline C-40
Fuyun Ling C-420
Jay Pasco-Anderson C-410
John Payton C-410
Shadid Qureshi C-40
Robert Schissler C-410
M. Sridhar C-410

Summary Implementation of Digital Interpolation/Sabstraction in V.32 Receiver

The problem, which occurs when the modem operate in loopback timing mode, has been solved. Now we can say that the receiver uses the scheme of digital interpolation performs at least as well as the origial Gen 0 design.

1. The implementation is basically the same as the one I described in last memo. For your reference, a block diagram is given in next page.

The feature of this design is that the transmitter PTT adjustment is automatically compensated in digital interpolation block. Hence, the timing recovery will never "see" phase or frequency changes in local TX PTT. The behavior of the new receiver is almost identical to the original analog interpolation schem No degradation will be introduced in external or loopback timing modes compared to the original analog design.


Echo Canceller
Hybrid.
FIFO Queue
Digital Interpolator
Equalizer
Queue Monitor
Timing Recovery
Tx PTT 2nd Order Tracker
NETCOR
2nd Order Software PLL
D
Rx PTT First Order Tracker

From the block diagram given the previous page, it can be seen that the depth of input queue monitor is still used. This is because that although the Tx PTT adjustment in external timing mode is compensated almost pefectly, (to a degree of 99.99%, so the receiver will not notice), there still a small part that cannot be canceled out. This small error may accumulate to cause depth of input queue change, and distroy the normal operation. A theoretical analysis is given in Appendix of this memo, and discussed later. Here we want to point out that, this error is only in a order of 10^{-8} (frequency deviation of Tx signal is 0.01% = 10^{-4}, the residual error after netcor compensation is approximately this number squared namely $(10^{-4})^2 = 10^{-8}$). As a result, the depth of input queue monitor only acts very occationally. In present implementation, it may adjust every on second, if it is needed. Each adjustments is about 1μs. To avoid any positive feedback, the adjustment is first into the software

PLL, then affects RxPTT indirectly. Such a small adjustment would not degradate timing recovery and digital interpolation at all.

2. From the result obtained from Appendix. We observe some interesting conclusions.

A. The order of the relative error given in the end of Appendix is approximately the relative error of remote Tx clock or External clock squared. If these clocks have errors in order of 10^{-4}. The resulting error between RxPTT and remote transmitter clock is about 10^{-8}. It is easy to compensate.

B. if $\Delta_E = 0$ (internal timing) or $\Delta_E = \Delta$ (loopback timing) No error exists between local Rx and remote Tx. In such cases, we don't need any further compensation.

C. if $\Delta_E \neq 0$ and $\Delta_E \neq \Delta$ (external timing), depth of input queue has to be monitored. Additional adjustments to RxPTT are necessary. However, since this error is very small, an easy first order adjustment is sufficient and will not degradate receiver performance.

Appendix: An Analysis on residual error between RxPTT clock and remote transmitter clock after NETCOR Compensation.

In order to ensure normal operation of modem, the local receiver clock should be sychronized with the remote transmitter clock. any frequency difference between these two clocks will eventually distroy the receiver. For the simplesity of analysis we will consider the period of the clocks instead of frequency.

When in internal timing mode, the local transmitter clock is obtained from a Xtal, We denote its period as T_o. Assume the remote transmitter has a period equal to T_{rf}. We denote their difference Δ, and $\Delta = T_{rf} - T_o$.

In order to keep the digital interpolator point and the remote transmitter sychronized, the digital PLL has to advance by $\frac{\Delta}{T_o}$, Here we have used normalized digital phase in digital PLL, namely 1 = one Tx sample period.

This digital phase change is used to adjust Rx PTT, If we made conversion exactly, the local receiver sample clock has a period T_{r_l} equal to

$$T_{r_l} = T_0 + \frac{\Delta}{T_0} \cdot T_0 = T_0 + \Delta = T_{r_f}.$$

No frequency diffrence exist between local and Rx remote Tx.

Suppose now we use external timing. Let the period of external clock is $T_E = T_0 + \Delta_E$, or the normaliz reading from NETCOR is $\frac{\Delta_E}{T_0}$ every time.

In is case, the timing recovery ad digital PLL will still keep its pointer ad remote transmitter clock synchronized. Considering the effets of NETCOR, the normalized phase change of the digital PLL will bee.

$$\Delta\varphi = \frac{T_{r_f} - T_E}{T_E} + \frac{\Delta_E}{T_0} = \frac{T_0 + \Delta - T_0 + \Delta_E}{T_E} + \frac{\Delta_E}{T_0} = \frac{\Delta - \Delta_E}{T_E} + \frac{\Delta_E}{T_0}$$

(Since the input sample is at external clock, we have to normalized the digital phase change relative to T_E)

The adjustment made to Rx PTT is $\Delta\varphi \cdot T_0$. The local receiver clock has a period of

$$T_{r_l} = T_0 + \Delta\varphi \cdot T_0 = T_0 + \Delta_E + \frac{\Delta}{T_E} T_0 - \frac{\Delta_E}{T_E} T_0$$

Then $T_{r_l} - T_{r_f} = \left(T_0 + \Delta_E + \frac{\Delta}{T_E} T_0 - \frac{\Delta_E}{T_E} T_0\right) - (T_0 + \Delta) = \frac{(\Delta_E - \Delta)\,\Delta_E}{T_E}$

or the relative error is $\frac{(\Delta_E - \Delta)\Delta_E}{T_0 T_E}$.

Memorandum

codex
MOTOROLA

To: List

From: Fuyun Ling F.Ly

Date: October 9, 1986 FL:02-86

Subject: Digital Subtractor for Gen 1 V.32 Modem

In the past several weeks. We have done some final adjustment to the Digital Subtraction/Interpolation Software. The Gen 1 V.32 modem with digital subtractor is now fully functional. The first software freeze has begun. In this memo, I've described some of the modifications of interest. The problems revealed in the digital subtraction modem may also affect the present Gen 1 V.32 modem. Hence, it may also be necessary to check for the V.32 even V.33 modems.

"Even though you only touch a hair, the whole body may move." -Chinese Proverb.

FL/ejwg

List:	Shahid Qureshi	C2-140
	Jack Moran	C2-80
	Royal Laurent	C2-90
	Rich Posklensky	C2-80
	Vedat Eyuboglu	C2-140
	Dick Kline	C2-140
	John Payton	C2-140
	John Greszczuk	C2-140
	Jay Pasco-Anderson	C2-140
	Rashid Chaudary	C2-80
	Bob Schissler	C2-100
	M. Sridhar	C2-90
	Carol Bargoot	C2-90
cc:	Stan Ivas	C2-80
	Rich Kumpf	C2-100
	Neil Sheer	C2-100

1. AGC Parameters:

The AGC subroutine in the V.32 modem is basically the same as the one in R400/2600 modems. However, the AGC routine is called every baud in the V.32 modem instead of every other band in the R400/2600 modems. In order to obtain the same response time, some parameters were changed to accomodate this. The time constant of the AGC routine is affected by the parameters of the "leaky" integrators of the routine. Namely, a leaky integrator has the following form:

$$A(n) = w.A(n-1) + X(n)$$

The time constant of the integrater is approximately equal to $t = T/(1-w)$, when the subroutine is called every T seconds. Obviously, in order to have the same t, $1/(1-w)$ for the V.32 has to be twice as large as the $1/(1-w)$ for the R400/2600 modems, which call AGC every other baud. On the other hand, the average value of A is also proportional to the quantity $1/(1-w)$. As a result, A in the V.32 modem is also twice as large as A in the R400/2600 modems. This increase in magnitude actually causes saturation in the quantity EAGC. We have changed the value of w close to the value used in 2600/R400 modem. The time constant t is now reduced to half of the value before. Of course we have to change the adjustment threshold ETHR and adjustment step size C accordingly to ensure normal operation. (ETHR reduced and C increased). This will cause a faster responce time. But I can't think of any bad effect yet.

2. Catastrophe Detector:

The Catastrophe Detector has been described quite clearly in the July 27, 1984 memo by Mark Huntzinger. I have restored the 2600/R400 parameter to the V.32 modem. Due to the modification made in VITDEC previously, the metric in V.32 VITDEC is twice as large as

in the 2600/R400. The parameters related to the metric are LMBIAS and PHROT. Considering the larger value of the metric, LMBIAS should be twice as large and PHROT should be half as in the 2600/R400. Since the signal constellation in the V.32 Modem is not as dense as at higher rates. I tripled LMBIAS (instead of double it), while cut PHROT to a half. I also reduced AUGH to 480H from 51FH. It seems the larger LMBIAS and smaller AUGH give us better noise immunity. I believe the V.32 Gen 1 modem with analog subtractor should also change these values accordingly.

3. Discontinuity in data buffer due to "exact" timing jam.

   One problem unique to the digital subtraction V.32 modem is due to the fact that, when we do a timing jam, we move the get-pointer of the input data buffer. This causes a discontinuity in the equalizer delay line. The effect will exist until the discontinuity has passed the phase splitting filter, or at least its main tap. The problem is solved by delaying training the AA or AC equalizer by 6 Bauds. I describe this problem here because I feel it may happen in future new modem designs.

4. Call Progress:

   Call progress has been implemented. It is found that we have enough time to monitor call progress using Alfredo's software. The new more efficient version of IIR filter may be included in the future.

5. Analog Scaling:

   The statement I made before that "the analog gain should be adjusted to make the analog loopback (LAL) work properly" is not correct. The accurate statement

should be, "The analog gain should be adjusted to avoid saturation at highest recieved signal, namely back to back connection." It was shown, both analytically and experimently, a 3 dB downward adjustment is necessary to avoid saturation. The effect of this adjustment to the modem performance at lowest received signal level is still to be seen. Again, it may be necessary to check the gain setting of the V.32 modem with analog substraction. It has a gain of 6 dB higher (or 9 dB high after the 3 dB adjustment is done) than the digital subtraction modem. Will any saturation occur?

The software has been frozen. The performance test can begin. It's time for me to start something new.