Functional Design of the Asynchronous Pulse Blanker (Rev. 1)

Steven W. Ellingson*

May 1, 2002

Contents

*The Ohio State University, ElectroScience Laboratory, 1320 Kinnear Road, Columbus, OH 43210, USA. Email: ellingson.1@osu.edu.

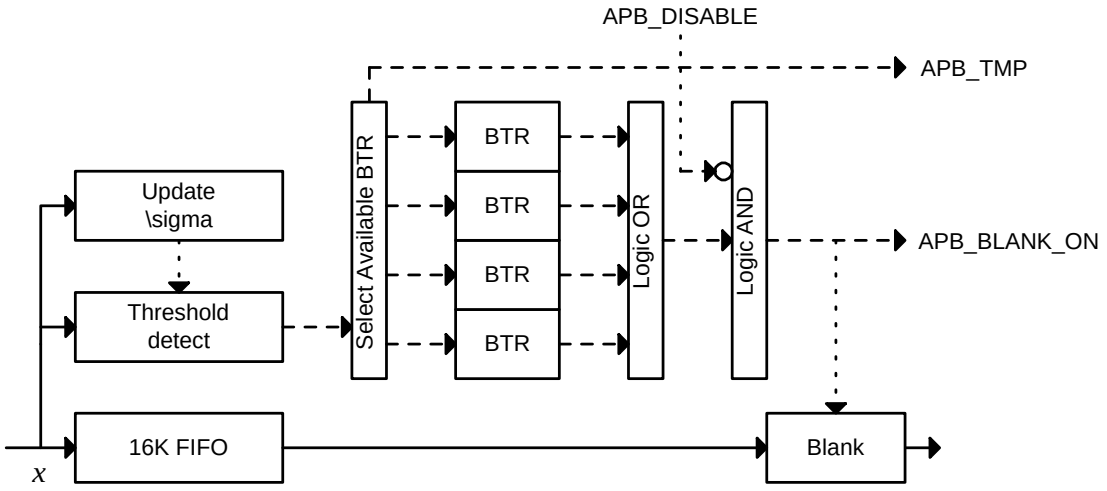

Figure 1: APB Block Diagram.

1 Introduction

This note refines the design of the IIP radiometer's Asynchronous Pulse Blanker (APB), first described in Section 2 of [1]. The concept has changed somewhat from that description; the new block diagram of the APB is shown in Figure 1. The APB now consists of a fixed-length 16K-sample buffer (163.4 μs total delay), followed by a blanker. $\|x\|^2$, the squared-magnitude of the sample x currently entering the buffer, is compared to a threshold δ. If exceeded, a user-selectable number of samples preceding and following the detected sample are set to zero. This is accomplished using one of (at least) four "blanking timing registers" (BTR's). Once triggered, a BTR waits for a predetermined number of samples, then requests blanking for a certain number of samples. The outputs of the BTRs are ORed to determine which samples are actually blanked.

This scheme is intended to accomodate the expected pulse width of the RFI source (probably a radar), plus margin to accomodate multipath, receiver recovery, and so on. The idea is to make the blanking intervals long enough to remove corrupted time samples, but no longer so as not to give away any more integration time than necessary.

δ is set to be $\beta\sigma + m$, where σ and m are the standard deviation and mean (respectively) of $\|x\|^2$. In other words, a detection is declared for any sample of $\|x\|^2$ that is greater than β standard deviations away from the mean. To avoid taking square roots, it is probably better to define the detection criterion as $(\|x\|^2 - m)^2 > \beta^2\sigma^2$; note that σ^2 is simply the variance of $\|x\|^2$. σ^2 is computed from samples taken when pulses are (nominally) not present. β sets the "aggressiveness" of the blanker: a low value (e.g., $\beta = 3$) will tend to trigger on spurious noise peaks, whereas a high value (e.g., $\beta = 100$) may allow weak pulses through. Good performance was demonstrated with $\beta = 10$ for pulse radar field data in [2].

2 Real-time Input

- 100 MSPS signal consisting of complex-valued samples, as would be provided by the front end proposed in [3]. The samples may be up to 10-bits "I" by 10-bits "Q". For notational convenience, the value of this signal at the input to the FIFO is henceforth referred to as x.

- `APB_PARM_RESET` (Reset Operating Parameters), a boolean which, when asserted, resets the values of various operating parameters using the values described in Section 3. The update is simultaneous for all parameters. Edge-triggered.

- `APB_DISABLE` (Disable APB), a boolean which, while asserted, prevents blanking. The APB continues to operate normally except samples are never actually blanked.

3 Non-Real-Time Input

Note: these are operating parameters which are held in registers and do not actually become the current values until `APB_PARM_RESET` is asserted.

- `APB_B2`, a register holding β^2.

- APB_MUMEAN, the smoothing coefficient μ_{mean} used for computing m, the mean of $||x||^2$. This is set to some number in the range [0,1], typically very close to 1.

- APB_MUVAR, the smoothing coefficient μ_{var} used for computing σ^2, the variance of $||x||^2$. This is set to some number in the range [0,1], typically very close to 1.

- APB_OLDMEAN, the "old" value of m. Used to initialize the APB to a reasonable state. (Nominally, this value will be determined as part of the system start-up procedure from observations of the output before enabling the APB.)

- APB_OLDVAR, the "old" value of σ^2. Used to initialize the APB to a reasonable state. (Nominally, this value will be determined as part of the system start-up procedure from observations of the output before enabling the APB.)

- APB_NWAIT, a register holding N_{wait}, the number of sample clocks to wait before blanking. Valid values are between 0 and the length of the FIFO buffer.

- APB_NBLANK, a register holding N_{blank}, the minimum number of samples to blank.

- APB_NSEP, a register holding N_{sep}, the minimum number of samples between attempts to trigger a BTR. This should nominally be set to be slightly longer than the longest anticipated pulse length in order to avoid multiple blank requests for a single pulse.

4 Output

- 100 MSPS signal, identical in all respects to the real-time input except for those samples which are blanked and hence set to zero.

- APB_BLANK_ON, a boolean value which is TRUE when blanking is in effect. This signal is provided for the convenience of subsequent processing blocks (e.g., the PE/S unit) and for debugging/monitoring purposes. Nominally this signal shall be made available at a square header separate from the main output, and also made available via any other available control/status/diagnostic interface. Note: It is anticipated that units downstream can detect when blanking is

being used simply by observing zero-valued samples; therefore, this use of the APB_BLANK_ON signal is not a design priority.

- APB_TMP (too many pulses), a boolean which is asserted if the APB runs out of BTR's. If asserted, it means that N_{wait} and/or N_{blank} should be shorter, or possibly that N_{sep} should be longer. APB_TMP is unasserted only when APB_PARM_RESET is asserted.

5 Operating Algorithm

Shown below is the procedure for determining when to blank (detection). This procedure runs continuously without interruption.

Per sample clock:

1. $m \leftarrow (1 - \mu_{mean})\|x\|^2 + \mu_{mean} m$
2. TEMP $\leftarrow (1 - \mu_{var})(\|x\|^2 - m)^2 + \mu_{var}\sigma^2$
3. If $(\|x\|^2 - m)^2 < \beta^2$TEMP,

   $\sigma^2 \leftarrow$ TEMP

   else (pulse detected)

   If we have not arrived here within the last N_{sep} sample clocks,

   Identify an available BTR. If none is available, assert APB_TMP.

   Trigger that BTR.
4. Go to 1.

Note that the above algorithm only updates the variance (σ^2) when pulses do not appear to be present. However, the mean (m) is updated regardless of whether pulses are detected or not.

Also, note that this algorithm must not be allowed to detect pulses or skip updates of σ^2 until it has converged to reasonable values of m and σ^2. There are two methods

by which this can be accomplished. One method is to disable blanking for a short period of time and use a processor downstream to compute reasonable initial values of m and σ^2. A second method is to begin by forcing updates of σ^2 on every iteration until the algorithm has converged, and then to re-enable selective updating of σ^2 as implied by the above algorithm. The second method shall be supported by the APB implementation.

Shown below is the operation of a BTR. This procedure only runs when launched by the above (detection) procedure. Note that each BTR is identical and runs independently of the others.

Once triggered:

1. Wait N_{wait} sample clocks.

2. Request blanking of the next N_{blank} samples emerging from the FIFO.

3. End (Allow detection procedure to request another blanking event.)

Note that N_{wait} determines the number of samples preceding a detection to blank. For example, to blank exactly one sample, N_{wait} should be set to the FIFO length and N_{blank} should be set to 1. To blank an additional sample preceding the detected sample, set N_{wait} to the FIFO length minus 1, and N_{blank} to 2. Also, note that the actual blanking may be different since it is possible for BTR blanking requests to overlap.

6 Simulation

In this section, some aspects of the algorithm proposed in the previous section are demonstated in simulation. We begin with x as 64K samples drawn from a complex circular Gaussian distribution with zero mean and magnitude variance equal to 1.* In this case, the statistics of $\|x\|^2$ are $m = 1$ and $\sigma^2 = 1$. Figure 2 shows the results using the proposed algorithm with initial values $m = \sigma^2 = 0$ and forcing σ^2 to update on every iteration. Note that the estimate of σ^2 converges within about 10K samples

*In MATLAB: `x = (randn(L,1)+j*randn(L,1))/sqrt(2).`

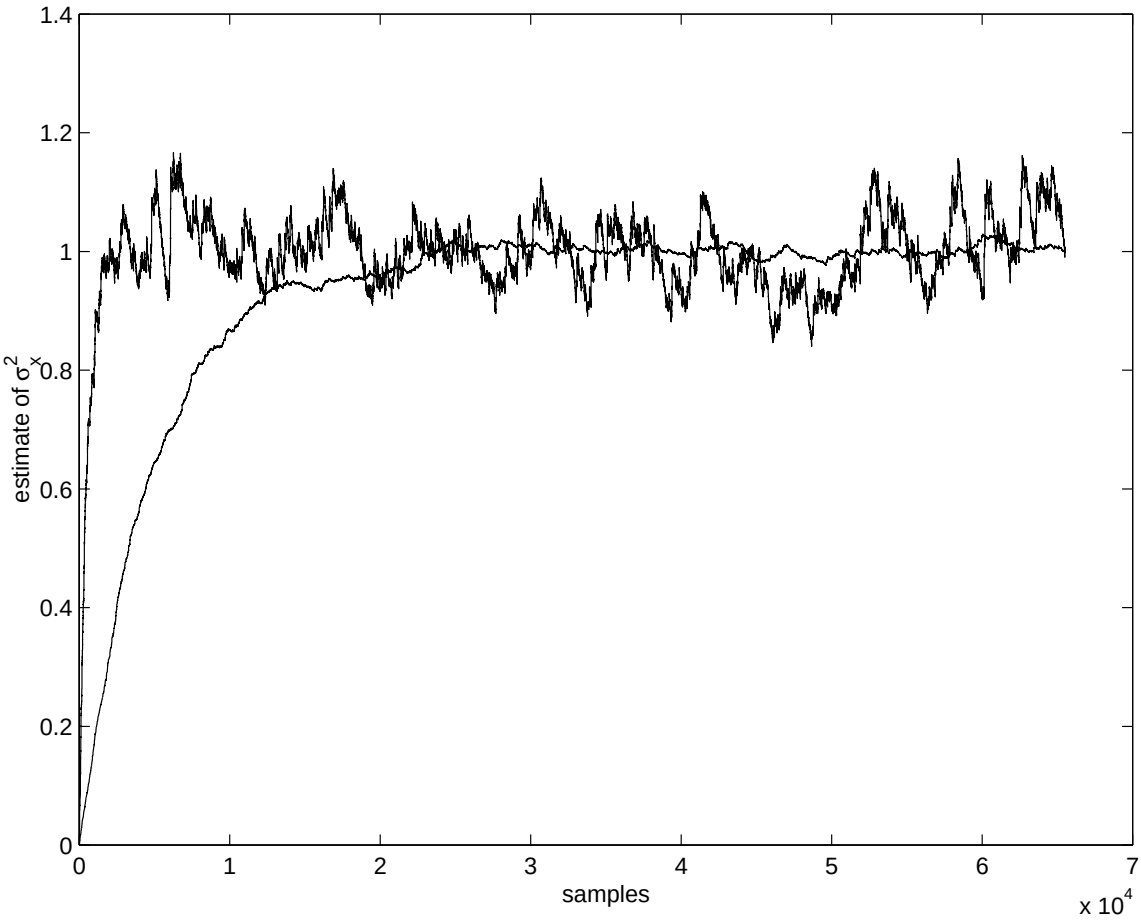

Figure 2: Convergence of the estimate of σ^2 provided by proposed algorithm in the absense of pulses. The two curves are for $\mu_{mean} = \mu_{var} = 0.999$ (fast convergence) and $\mu_{mean} = \mu_{var} = 0.9999$ (slow convergence, but smoother).

for $\mu_{mean} = \mu_{var} = 0.9999$ and within about 1K samples for $\mu_{mean} = \mu_{var} = 0.999$. The price paid for faster convergence is a steady-state variation of about 20% around the true value, as opposed to only a few percent in the other case.

Next, we add pulses to the previous input data set. Each pulse consists of exactly one sample. Twenty pulses are added, with the magnitude-squared of each pulse ranging from $1\sigma = 1$ to $20\sigma = 20$, appearing every 2000 samples beginning at the $20,000^{th}$ sample. This pulse train is evident in the top panel of Figure 3. The bottom panel shows the results using the proposed algorithm, once again with initial values $m = \sigma^2 = 0$ and forcing σ^2 to update on each iteration. Note that the estimate becomes significantly biased when pulses larger than 10σ are present. This motivates

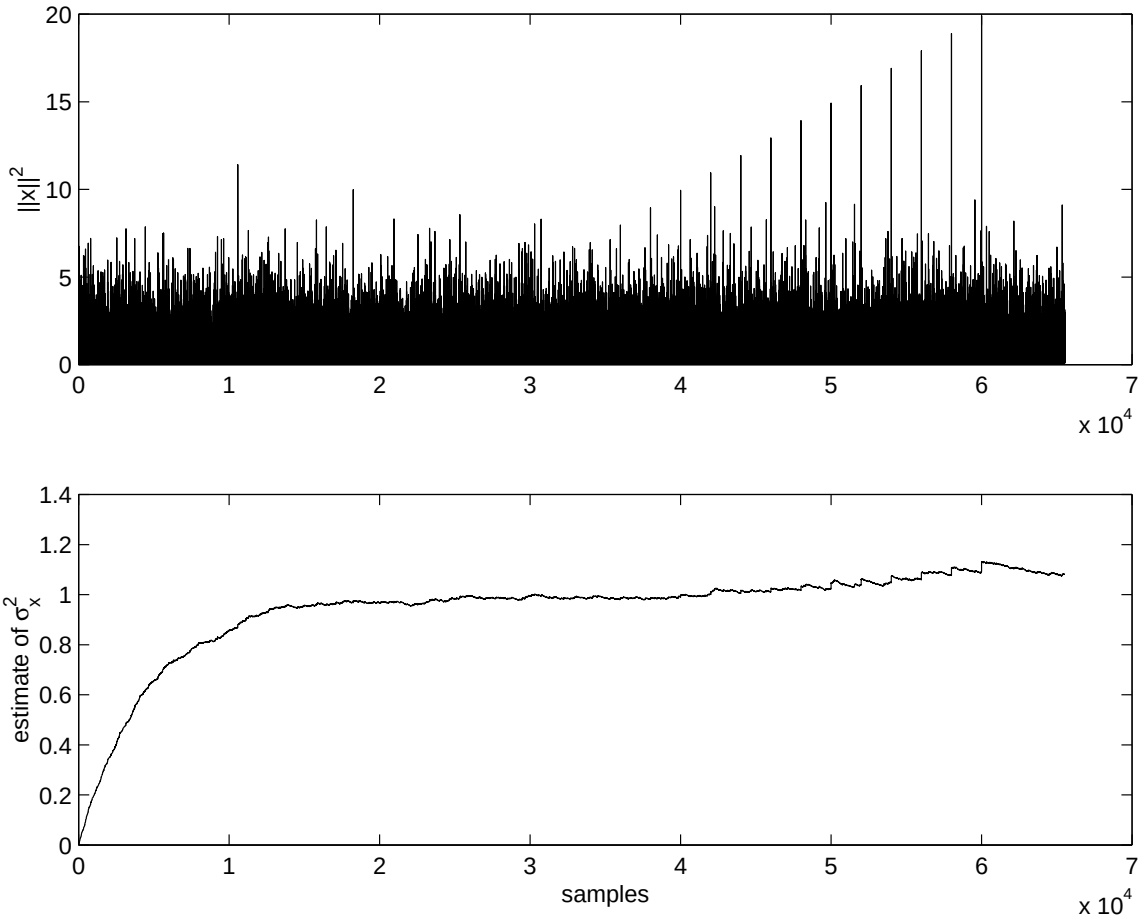

Figure 3: *Bottom Panel:* Convergence of the estimate of σ^2 provided by proposed algorithm given the data shown in the top panel. $\mu_{mean} = \mu_{var} = 0.9999$.

the feature of the full algorithm in which the estimate of σ^2 is not updated if a pulse is detected.

Figure 4 implements the full algorithm, including selective updating of σ^2. Note that this alleviates the bias problem identified above. Also note that the fact that m is updated even in the presense of pulses has no noticible effect on the performance.

Finally, we consider the relationship between β and the "false alarm" rate for the full algorithm. The input x is once again the original (noise-only) time series. In this experiment, we run the simulation for various values of β and count the samples satisfying the detection criterion. The results are as follows:

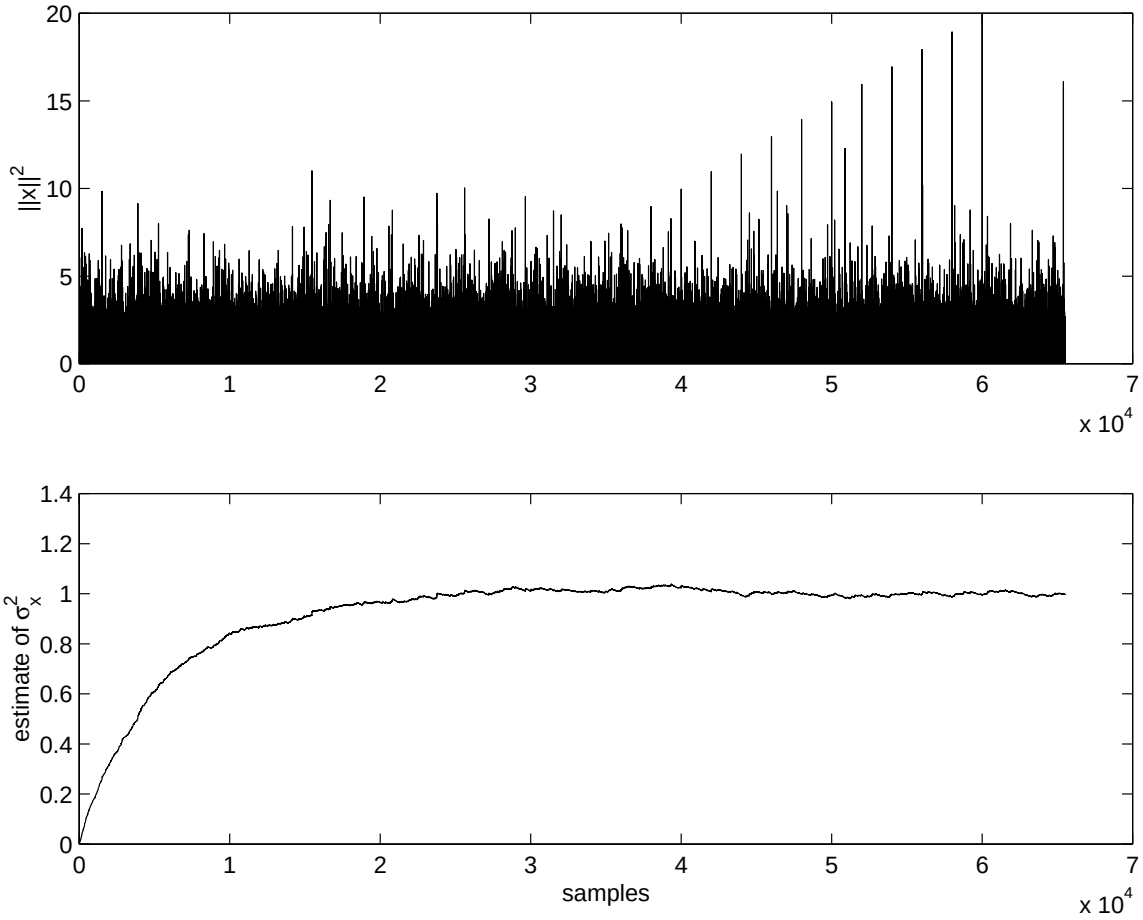

Figure 4: Same as Figure 3, except no longer updating σ^2 if a pulse is detected (as in the full algorithm). The first 19,000 samples are used to achieve convergence, during which σ^2 is updated on every iteration.

β	samples satisfying detection criterion
1.0	47 %
2.0	8.9 %
3.0	3.2 %
4.0	1.0 %
5.0	0.3 %
6.0	0.12%
7.0	0.03%
8.0	0.01%

The above table should be useful in selecting a value for β as well as as a diagnostic for the implementation.

References

[1] S.W. Ellingson, "Design Concept for the IIP Radiometer RFI Processor," informal memo, January 23, 2002.

[2] S.W. Ellingson, "Analysis and Mitigation of Radar at the RPA," informal report, January 20, 2002.

[3] S.W. Ellingson, "Design Concept for the IIP Radiometer," informal memo, January 11, 2002.