VLSI data demodulator for a microwave landing system

A.A. Abbo

Abstract: A VLSI circuit for demodulating the microwave landing system data format is presented. The approach is based on a second-order all-digital phase locked loop. The primary goal of the work is to find a cost effective way of embedding the demodulator within an integrated navigation receiver built around a customised application specific processor (ASP). While the fully digital nature of the design increases the demodulator's immunity to analogue-related problems, the dedicated hardware approach relieves the ASP from computationally complex tasks, such as carrier acquisition and clock synchronisation.

1 Introduction

Integration of different navigation aids has recently been given much attention by the navigation community. Since no one system can fully satisfy the reliability and integrity demands of today's navigation, it is necessary to use one or more back-up systems. In addition to the functional redundancy, integration also opens an opportunity for the interoperability of different systems leading to overall performance improvement [1–3].

Even though integration can be done by simply putting together independent navigation receivers and later manipulating their outputs, a cost effective approach would require reduction of hardware redundancy. Optimal resource usage can be achieved through clever partitioning of the signal processing tasks among different implementation alternatives, i.e. analogue, digital hardware and software. In addition to cost reduction, successful partitioning can lead to a low-power and low-weight product.

An integrated navigation receiver which uses a single-chip high-performance processor has been reported in [1]. This navigation receiver, called GOLLUM, consists of four subsystems: GPS, Omega, Loran-C and the microwave landing system (MLS). The four subsystems have been chosen to cover different phases of aircraft navigation; (i) GPS, with Loran-C and Omega as back up, for en-route navigation, and (ii) GPS combined with MLS during approach, landing and take-off. To reduce the system cost, the distance measuring equipment (DME) which is needed by MLS, is replaced by the functionalities of GPS. Resources such as frequency synthesisers and analogue-to-digital converters are shared among different subsystems to achieve the design goals mentioned earlier.

In this paper, we focus on the MLS subsystem, particularly on the design of a data demodulator in the context of the GOLLUM integrated navigation receiver. The objective is to develop a demodulator that allows a greater degree of system integration. The fact that the GOLLUM design is based on an application specific processor (ASP) provides an opportunity for investigating nonconventional demodulator architectures. Previous designs were mostly based on semi-analogue techniques [4, 5], whose performance can be improved by moving to a fully digital design. One such approach is the DSP-based demodulator reported in [6]. Unfortunately, the algorithms proved to be too computationally complex to allow processor sharing among the navigation subsystems. We propose a design based on an all-digital phase locked loop (ADPLL) to reduce the number of dedicated DSPs. A demodulator which uses a second-order ADPLL has been realised on a sea-of-gates chip and costs about 23000 transistors, which is 12% of the space available for the application specific processor [7].

2 MLS receiver signal processing

The MLS, with a capacity of 200 independent guidance channels, was proposed a couple of decades ago to handle the ever increasing size of air traffic [8]. Each channel is allocated 300kHz of the 5031–5091MHz frequency band, and time multiplexed to transmit up to 15 different guidance functions. The functions are grouped into angle and data functions.

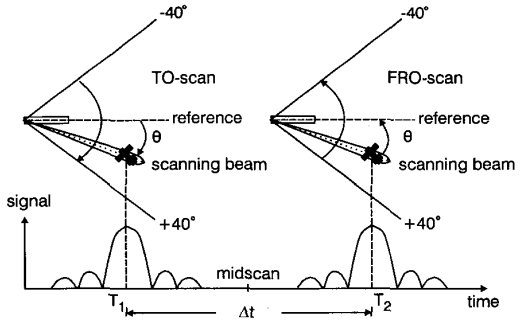

Fig. 1 *Scanning-beam pulses for angular position measurement*

The angle information, i.e. elevation, azimuth or flare, is transmitted using the time-reference scanning beam (TRSB) technique [8]. Fig. 1 shows the scanning beam operation. A vertically wide and horizontally narrow

IEE Proceedings online no. 19990112

DOI: 10.1049/ip-cds:19990112

Paper first received 23rd February and in final revised form 21st December 1998

The author was with Delft University of Technology and is now with Philips Research, P.O. Box WAY-41, Prof. Holstlaan 4, 5656 AA Eindhoven, The Netherlands

beam is used to scan the air-space in the azimuth direction. Likewise, a vertically narrow and horizontally wide beam is used in the elevation direction. The left–right (azimuth) and up–down (elevation) scanning directions are distinguished by the symbols TO and FRO, respectively.

In the TRSB technique, initially data is broadcast in all directions to synchronise the airborne receiver to the ground equipment. Once synchronisation is established, the receiver looks for the scanning beam pulses and computes the angular position from the time difference (Δt) between the TO and FRO pulses. Given the scanning speed, V [deg./μs], and the reference time difference, T_o [μs], the angular position can be computed from

$$\theta = (T_o - \Delta t)V/2 \qquad (1)$$

where T_o corresponds to the case where an aircraft is aligned with the runway.

The critical part of MLS angle processing is accurate time difference measurement. Since the quality of the received signal is degraded by noise, antenna shielding by the body of the aircraft, and reflections from nearby objects (multipath), robust algorithms are needed to handle such situations. The two well known algorithms in the MLS literature are the dwell-gate and split-gate algorithms. These algorithms and the effect of analogue-to-digital converter resolution on the algorithms' performance are discussed in [9].

Besides angle processing, the MLS receiver has to demodulate and decode the received data functions. Including the preamble bits, the length of the data packets varies from 32 bits for basic data functions to 64 bits for auxiliary data functions. Fig. 2 shows the contents of an MLS data packet.

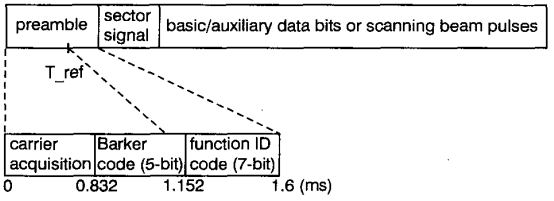

Fig. 2 *MLS data function organisation*

During the preamble period, the receiver has to complete carrier and clock synchronisation and identify the function type that is contained in the remaining part of the packet. For clock and frame synchronisation purposes, a five-bit Barker code (11101) is included ahead of the function identification code. The bits are differential phase shift keying (DPSK) encoded and have a rate of f_b = 15.625kHz.

From the power budget of the MLS specifications [8], the signal strength at the demodulator input is computed to be SNR ≥ 5dB. At this signal-to-noise ratio (SNR), the airborne receiver should guarantee a bit error rate (BER) of $BER < 10^{-4}$ to satisfy the MLS integrity requirements [10]. In the following Sections, we discuss an efficient digital demodulation technique which satisfies the BER and other receiver requirements.

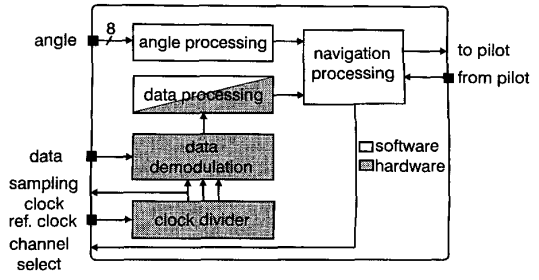

Fig. 4 *MLS receiver architecture (baseband processor)*

3 MLS receiver architecture

Figs. 3 and 4 show a functional block diagram of the MLS receiver. The front-end consists of a programmable frequency synthesiser with which one of the 200 channels can be selected for further processing. The antenna signal is mixed with the synthesiser output into an intermediate frequency (IF) signal. Further frequency down-conversion takes place in the IF stage, which also splits the incoming signal into two: first, an envelope-detected signal for angle processing, and secondly, a hard-limited signal for data demodulation.

The digital part of the receiver performs four functions: angle processing, data demodulation, data processing and control. The MOVE processor [11] is used to implement these tasks either in software or using dedicated hardware modules, which are integrated on the same chip with other functional units of the processor. The processor has a single-instruction (move) and employs a transport-triggered architecture (TTA) in which execution is initiated as a consequence of moving data into a functional unit. The processor belongs to the class of very long instruction word (VLIW) processors and achieves high-performance

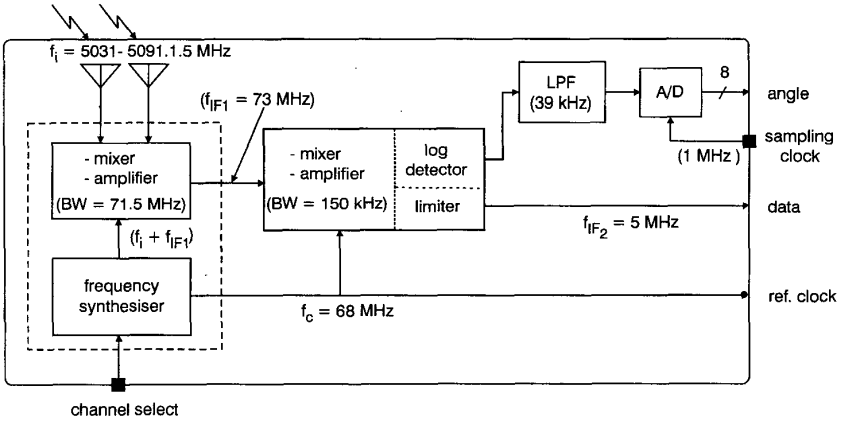

Fig. 3 *MLS receiver architecture*
Front end and intermediate frequency (IF) sections preprocess the received signal to match the capacity of the baseband processor (see Fig. 4)

through instruction-level parallelism. In embedded signal processing applications, like the GOLLUM integrated navigation receiver, the MOVE architecture provides a cost effective solution since it can be customised according to the needs of the application.

4 All-digital MLS data demodulation

Data demodulation can be done in either a coherent or an incoherent manner. While the former requires phase synchronisation between the incoming signal and the locally generated reference, the latter requires only a good frequency estimate of the input. In the case of MLS, the frequency uncertainty arises from transmitter and receiver oscillator uncertainties (±10kHz each) and Doppler shift (up to ±5kHz), which totals, in the worst case, to $\Delta f = \pm 25$kHz. Coherent data demodulation can be done by using Costas, squaring or remodulation loops [12]. These loops are typically useful in suppressed-carrier communications where no carrier component is left in the modulated data.

The hard-limited signal from the IF stage can be processed in a number of ways. Typical examples are the semi-analogue [5] and the fully software approaches [6]. In [5], the demodulation is based on a semi-analogue PLL-IC and a couple of logic gates. Even though the PLL-IC achieves a high level of integration, the loop filter being the only offchip component, the fact that the voltage controlled oscillator is sensitive to power supply and temperature variations, and component parameter changes with aging makes it less attractive.

In [6], however, the hard-limited signal is filtered and digitised into a 12-bit data at a sampling rate of 325kHz. The sampled signal is spectrum analysed using a modified fast Fourier transform (FFT) to find a coarse estimate of the carrier frequency. This is followed by an automatic frequency control (AFC) loop which further reduces the frequency uncertainty. The demodulation is then done using an incoherent technique — a dot-product detector. The computational complexity of the algorithms in this approach was evaluated using the MOVE framework [7]. It appeared that the processor buses are used exclusively by the MLS subsystem, especially during carrier acquisition, which implies the need for a dedicated MOVE processor.

The third possibility, which is the focus of this paper, is to directly process the hard-limited signal in the digital domain. This is achieved with the help of an ADPLL which operates at a high sampling rate. In addition to the ease of integration, the fully digital nature avoids the analogue-related problems mentioned earlier. A detailed discussion of ADPLL architectures and their performance analysis can be found in the literature [13–16].

Our choice of a coherent demodulation technique is based on the simplicity of the ADPLL for implementation as a dedicated functional unit on the application specific processor. Since the incoming MLS carrier suffers from frequency uncertainty, a second-order ADPLL is used to keep the output phase error small, which is important for a low BER. A second-order ADPLL, which works on a one-bit data width, can be built from a cascade of two first-order loops [17] or by combining a rate multiplier with a proportional path [16]. The one-bit data width results in a simple and area efficient very large scale integrated (VLSI) circuit implementation.

While the hardware cost of cascaded or rate multiplier-based second-order loops is nearly the same, the latter has been chosen for its lower internally generated phase jitter. Especially when the rate multiplier is an accumulating type, its output pulses are more regular which leads to low spectral noise [16]. On the other hand, a cascaded second-order loop requires a coupling counter when realising a wide hold range [17]. Consequently, one of the first-order sections cannot sufficiently average the jitter introduced by the other section and one ends up with a relatively higher output phase noise.

5 ADPLL-based data demodulation

With today's VLSI capabilities, all the phase lock loop components, i.e., the phase detector (PD), the loop filter and the tunable oscillator can be implemented in the digital domain. A simplified phase detector can be built from a multiplying type PD (XOR gate) or a sampling type PD (an RS flip-flop). A more complex sampling type PD is the phase-frequency detector (PFD), which, due to its wider linear range of operation, is attractive for most applications where the SNR is high. In the case of low SNR, a missing signal transition can be interpreted as a frequency error and the loop is steered away from its tracking state, thus degrading its performance [12, 14]. With this observation in mind and since our loop is required to work properly at $SNR_i = 5$dB, we have chosen the XOR-PD.

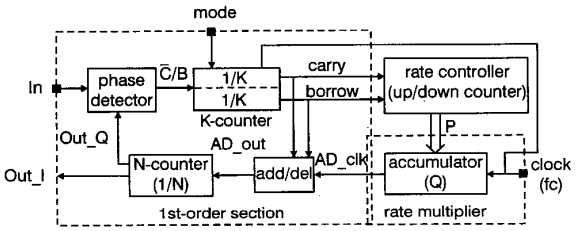

Fig. 5 *Rate multiplier-based second-order ADPLL*

Fig. 5 shows a second-order ADPLL based on a rate multiplier. The phase detector generates a count command ($\bar{C}/B$) depending on the momentary signal values of the input and the quadrature ADPLL output signals. Two divide-by-K counters realise the loop filter functionality by averaging the duty-cycle (δ) of the $\bar{C}/B$ command. When the counters overflow, carry and borrow pulses are generated which indicate in which direction and by how much the local oscillator frequency has to be changed. The loop filter response can be controlled by the mode input which selects predefined values for K during acquisition and tracking.

The carry and borrow pulses are accumulated in the rate controller to obtain the rate multiplier programming input (P). The rate multiplier is in effect a digitally controlled oscillator (DCO), with a frequency step of $\Delta f = \Delta P f_c/Q$ and output frequency given by eqn. 2 [16]. The DCO output is phase shifted (advanced or delayed) in the add–delete (add/del) unit when carry and borrow pulses are generated. The operation of the (add/del) unit is shown in Fig. 6. Finally, the irregular pulse distribution of the output of the add/del unit is averaged by the N-counter to form the inphase and quadrature ADPLL outputs.

$$f_{1D\_clk} = \frac{P}{Q} f_c \tag{2}$$

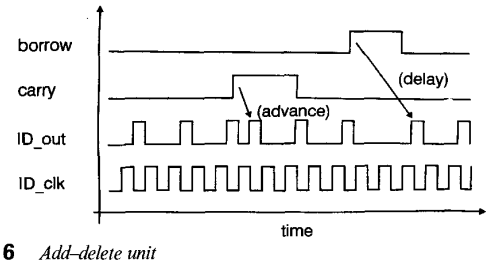

Fig. 6 *Add–delete unit*

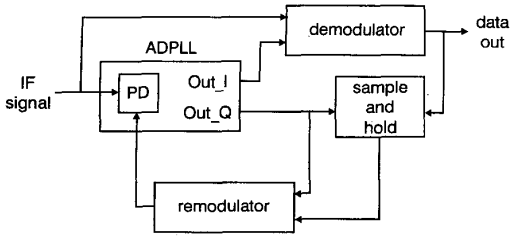

Fig. 7 *ADPLL-based binary phase shift keying demodulator*

A data demodulator can be built from the second-order ADPLL as shown in Fig. 7. A remodulation branch has been added to prevent the phase modulation in the incoming signal from passing into the phase-locked loop. The demodulator and remodulator blocks are built from simple XOR gates. The sample-and-hold unit (D-type flip-flop clocked by the quadrature ADPLL output) suppresses most of the phase jitter by taking samples away from the edges of the input signal. It is an approximation to the in-phase (I-arm) lowpass filter in the original remodulation loop [12]. In applications with moderate signal-to-noise ratios ($SNR \geq 5$dB, in our case), the performance degradation due to sub-optimal noise filtering is tolerable (see Section 5.1). On the other hand, the use of a sample-and-hold unit avoids the need for a delay unit in the quadrature (Q-) arm, which would have been necessary to compensate for the delay in the I-arm filter.

Fig. 8 shows the block diagram of the realised MLS data demodulator [7]. Additional blocks, such as a mixer, a data filter, a lock detector, a data clock synchroniser and a DPSK decoder have been added around the ADPLL to complete the demodulator functionality. The controller determines the operating modes of the ADPLL and the data filter according to the status of the demodulator, i.e. acquisition, clock synchronisation or tracking. The demodulator is interfaced to the application-specific processor after serial-to-parallel conversion, which helps to lower the interrupt rate and the associated service overheads.

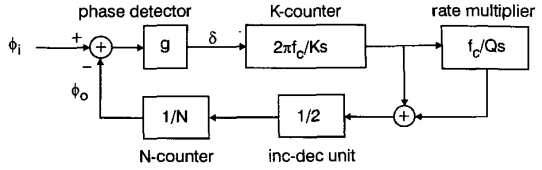

Fig. 9 *Model of a rate multiplier-based second-order ADPLL*

5.1 ADPLL parameterisation

Configuring a second-order PLL for a particular application involves determining the loop parameters, i.e. the natural frequency (ω_n) and the damping factor (ζ). The phase transfer function of a second-order ADPLL can be derived from the frequency domain model shown in Fig. 9. Such a continuous-time approximation is valid in such cases where the clock frequency (f_c) is sufficiently high compared to that of the input signal [13, 15]. This means that the phase quantisation noise in the ADPLL output signal is low enough to validate the use of a simplified linear model.

An XOR type phase detector has a triangular characteristic with gain $g = \delta/\phi_e = 2\pi$ for $-0.5\pi \leq \phi_e \leq 0.5\pi$, where δ is the duty cycle of PD output and $\phi_e = \phi_i - \phi_o$ is the phase error. The phase detector output reaches its peak value ($\delta = \pm 1$) when the phase error is $\phi_e = \pm\pi/2$ rad. Substituting the gain value, the closed loop phase transfer function and the corresponding loop parameters become

$$H(s) = \frac{\frac{2f_c}{NK}s + \frac{2(f_c)^2}{NKQ}}{s^2 + \frac{2f_c}{NK}s + \frac{2(f_c)^2}{NKQ}} \quad (3)$$

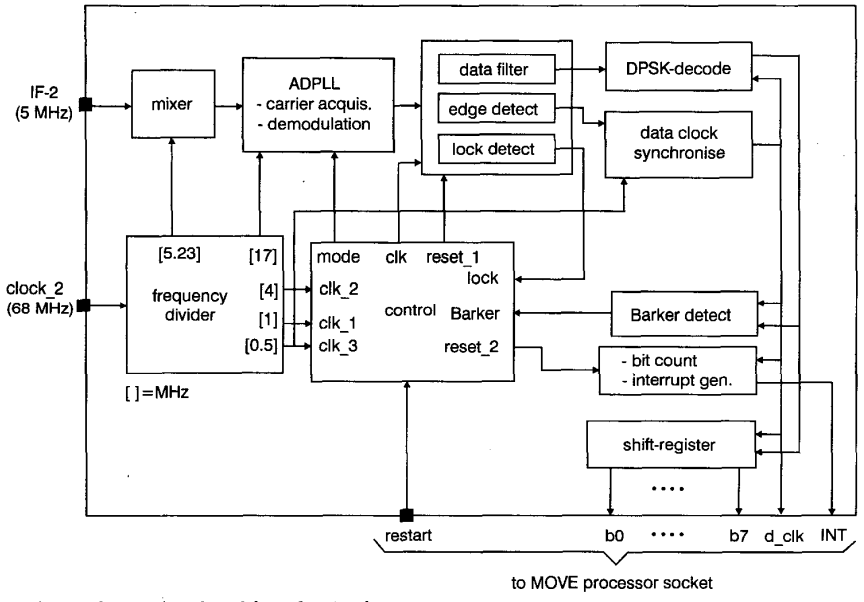

Fig. 8 *Block diagram of realised MLS data demodulator functional unit*

$$\omega_n = f_c\sqrt{\frac{2}{NKQ}} \tag{4}$$

$$\zeta = \frac{1}{2}\sqrt{\frac{2Q}{NK}} \tag{5}$$

The values of ω_n and ζ depend on the desired acquisition speed of the phase-locked loop and the allowable bit error rate of the demodulator. Acquisition can be a fast process if the input frequency lies within the capture (lock-in) range of the PLL. For an XOR phase detector the lock-in range and the lock-in time are given by [12, 16]

$$\Delta\omega_L \approx \pm\pi\zeta\omega_n \tag{6}$$

$$t_{lock} \approx 1/(\zeta\omega_n) \tag{7}$$

From eqn. 6, a lock-in range that covers the MLS carrier uncertainty of $\Delta f = \pm 25$kHz requires

$$\begin{aligned}\omega_n &> 2\pi \times \Delta f/(\pi\zeta)\\ &> 70.7 \text{ krad/s, for } \zeta = 0.707\end{aligned} \tag{8}$$

The corresponding lock-in time is $t_{lock} = 20\mu$s, which is well within the 832μs limit.

While the acquisition speed sets the lower limit on the loop natural frequency, the upper limit is set by the bit error rate requirement, $BER < 10^{-4}$. In a practical coherent receiver, a performance degradation is expected due to the influence of noise on the PLL operation. According to [18], the conditional and average BER of a DPSK signalling system with matched filtering can be expressed as

$$P_E(\phi_e) = \frac{1}{2}e^{-\gamma_b\cos^2(\phi_e)} \tag{9}$$

$$P_E = \int_{-\pi/2}^{\pi/2} p(\phi_e)P_E(\phi_e)d\phi_e \tag{10}$$

where $\phi_e = \phi_{in} - \phi_{out}$ is the phase error between the input carrier and the local reference, and $\gamma_b = E_b/N_o$ is the system parameter with E_b and N_o being the bit energy and the noise power density, respectively. The phase error probability density function, $p(\phi_e)$, is given by

$$p(\phi_e) = \frac{\exp(D\cos 2\phi_e)}{\pi I_o(D)}, \quad |\phi_e| < \pi/2 \tag{11}$$

where $I_o(D)$ is the modified Bessel function of 0th order. For a loop bandwidth of $B_L = 0.5\omega_n(\zeta + 0.25\zeta)$, input bandwidth W_i and bit rate of f_b, we have

$$D = \frac{\delta\gamma_b}{(1 + 1/(2\delta y\gamma_b))} \quad \delta = \frac{f_b}{B_L} \quad y = \frac{2B_L}{W_i} \tag{12}$$

When ϕ_e is small, which is the case for large loop SNRs, $p(\phi_e)$ converges to a Gaussian distribution with mean equal to zero and variance given by

$$\sigma^2_{\phi_e} = \frac{1}{D} = \frac{1}{2SNR_L} \tag{13}$$

$$SNR_L = S_L \times SNR_i\frac{W_i}{2B_L} \tag{14}$$

where SNR_L is the loop signal-to-noise ratio and $S_L = 1/(1 + 0.5SNR_i)$ accounts for the increase in phase noise variance due to noise products (noise × noise and signal × noise) in the carrier recovery loop [19].

Since the bit rate (f_b), the system parameter (γ_b) and the input bandwidth (W_i) are fixed, the loop noise bandwidth ($B_L << W_i$) is the only parameter that can be controlled to meet a certain BER requirement. The fixed parameters are: $f_b = 15.625$kHz and $W_i = 150$kHz, which yield $\gamma_b \geq 30$ for $SNR_i \geq 5$dB. Choosing $B_L = f_b$ yields $\sigma^2_{\phi e} = 0.083$ rad^2. Using a Gaussian phase error distribution and computing the integral numerically, we obtain an average bit error probability of 2.4×10^{-6}.

The above result is valid in the absence or rarity of cycle slippage. Otherwise, if there is a cycle slip, upon re-acquisition the remodulation loop (also Costas and squaring loops) can lock with 180° phase ambiguity. For DPSK signalling, a relock with a different phase leads to two erroneous bits localised to the cycle slip moment. The average time between cycle slips can be approximated by [12]

$$T_{AV} \approx \frac{\pi}{4B_L}\exp(4SNR_L) \tag{15}$$

To have a low bit error rate, it is necessary to keep the cycle slip rate as small as possible by choosing a high loop SNR. For instance, at $SNR_i = 3.16$ (5dB) and $B_L = f_b = 15.625$kHz, $SNR_L = 5.88$ (7.7dB) and $T_{AV} = 8.2 \times 10^5$s, which is sufficiently large compared to the maximum MLS packet length of 64 bits (= 4096μs).

Having chosen a value for the loop noise bandwidth ($B_L = f_b = 15.625$kHz), an upper bound on the loop natural frequency is obtained once the loop damping factor, ζ, is fixed. For $\zeta = 0.707$, which gives optimal loop settling, the upper bound on ω_n becomes

$$\omega_n < B_L/0.53 = 29.5 \text{ krad/s} \tag{16}$$

We see from eqns. 8 and 16 that the acquisition and BER specifications place conflicting demands on the loop bandwidth. This conflict can be resolved by providing some form of acquisition aid [12]. One such technique is bandwidth widening, which is applicable in cases where the input SNR is high enough to give stable loop operation, i.e. $SNR_L \geq 4$dB. In the ADPLL, bandwidth control is applied by changing the length of the K-counter.

We now relate ζ and ω_n to the ADPLL parameters using eqns. 4 and 5. Optimal loop settling behaviour requires $\zeta = 0.707$, which corresponds to $Q = NK$. If we choose an ADPLL clock of $f_c = 17$MHz, $N = 32$, $K = 32$, and $Q = NK = 1024$, we obtain $\omega_n = 23.5$ krad/s. We can double K to further reduce the noise bandwidth and improve the BER performance, in which case $\zeta = 0.5$, $\omega_n = 16.62$ krad/s and $B_L = 0.5\omega_n = 8.8$ kHz. The slight degradation in settling behaviour due to the change in ζ is acceptable since the MLS transmissions do not involve any frequency changes within a packet. Fast re-acquisition of new packets is handled through bandwidth expansion.

Another point of concern in using the ADPLL is the choice of the input signal frequency (f_{in}). The rate multiplier based loop can achieve lock in the frequency range $0 < f_{in} < f_{in\_max}$, where $f_{in_max} = f_cP_{max}/(2NQ) = 265.625$kHz and $P_{max} = Q$. Since the pulse irregularity (phase jitter) of the rate multiplier output has a minimum for frequencies near $f_c = 2N \times f_{in\_max}$, the input IF signal must be chosen close to f_{in_max}. To accommodate a frequency uncertainty of $\Delta f = \pm 25$kHz, the nominal ADPLL input frequency is chosen to be $f_{in_nom} = 230$kHz.

The operating range of the rate controlling counter, which sets the frequency of the rate multiplier output, is restricted such that the ADPLL output frequency stays within $[f_{in\_nom} - |\Delta f|, f_{in\_nom} + |\Delta f|]$ kHz. This is

done to avoid the ADPLL drifting to a frequency close to the integral divisions of the input signal frequency when there is no signal to lock to, for instance in between MLS packets. The ADPLL can attain false lock under such conditions, for which the demodulator output is useless.

It is also expected that the frequency acquisition will be fast when the initial state of the ADPLL is confined to the lock-in range of the loop. During the acquisition mode K is set to 8, leading to $\zeta = 1.414$, $\omega_n = 47$ krad/s and $B_L = 0.5\omega_n(\zeta + 1/4\zeta) = 37.4$kHz. According to eqn. 6 the lock-in range is $\Delta\omega_L = \pm 208.8$ krad/s, or $\Delta f_L = \Delta\omega_L/2\pi = \pm 33.23$kHz. This is just enough to cover the operating range of the ADPLL, and acquisition can be expected to be fast. Even though B_L is relatively wide and leads to reduced loop SNR, the resulting increase in phase jitter is of minor importance in the frequency acquisition mode. When lock is achieved, BL is switched to its tracking mode value (8.8kHz) for which the remaining phase acquisition is completed within a few cycles of the input signal.

5.2 Phase noise (jitter)

Phase jitter in the ADPLL output results from the phase noise contained in the input signal, and the phase quantisation due to the finite sampling clock frequency. When the ADPLL is used to build a data demodulator, the output phase jitter can also increase due to the additional signal path introduced to achieve modulation removal (see Fig. 7). A discussion of the jitter contribution of the remodulation branch is also given in this Section.

The internally generated phase noise (timing jitter) originates because the ADPLL output transitions occur at discrete time steps synchronous to the driving clock. This leads to a phase quantisation error. In the second-order loop of Fig. 5, there are two independent jitter sources. The first is the add–delete (add/del) unit. As shown in Fig. 6, the unit operates by advancing or retarding the edge of I/D_{out} by $T_c/2$, where T_c is the period of the input clock. Consequently, a timing jitter of $\pm T_c/2$ can appear at the ADPLL output. The second source of timing uncertainty in the second-order loop is the rate multiplier (RM). Depending on the ratio P/Q, its output (RM_{out}) shows some pulse irregularity. When the ADPLL is configured so that the frequency of RM_{out} is close to $f_c = 1/T_c$, the timing uncertainty is minimum and amounts to $\pm T_c/2$ [16].

Since the two timing jitter sources are independent, the overall timing uncertainty at the ADPLL output can take any of the five values: $v = [-T_c, -T_c/2, 0, T_c/2, T_c]$. From these values, one can estimate the peak phase jitter ($\Delta\phi_p$) of the output and the phase jitter variance as

$$\Delta\phi_p = 2\pi T_c/T_{in} \tag{17}$$

$$\sigma^2_{\phi_{o1}} = \sum_{i=1}^{i=5} \Pr(i) \times [2\pi v(i)/T_{in}]^2 \tag{18}$$

where, $T_{in} = 1/f_{in}$ is the input signal period, and $\Pr(i) = 1/5$ is the probability of the jitter values. For the ADPLL configuration given in the Section 5.1, i.e. $f_c = 1/T_c = 17$MHz and $f_{in,max} = 265$kHz, we obtain $\Delta\phi_p \approx 0.1$ rad and $\sigma^2_{\phi o1} \approx 0.005$ rad^2.

We now compare the internally generated phase jitter to that caused by the noise that is already present in the received signal. The phase jitter at the output of the phase locked loop depends on the input signal-to-noise ratio (SNR_i), the input bandwidth (B_i) and the loop noise bandwidth (B_L). Its variance is given by [12]

$$\sigma^2_{\phi_{o2}} = 1/2SNR_L = B_L/(2SNR_i \times B_i) \tag{19}$$

For the case where $SNR_i = 3.16$ (5dB), $B_i = 75$kHz and $B_L = 8.8$kHz (in the tracking mode), $\sigma^2_{\phi o2} \approx 0.019$ rad^2, which is about four times the jitter due to phase quantisation. The total phase noise variance at the ADPLL output can be approximated by $\sigma^2_{\phi o} = \sigma^2_{\phi o1} + \sigma^2_{\phi o2} \approx 0.024$ rad^2.

The influence of the remodulation path on the jitter performance of the ADPLL is shown in Fig. 10. Since the input phase noise ($\sigma^2_{\phi i} = 1/2SNR_i = 0.158$ rad^2) is much larger than that of the ADPLL outputs (U_{oI} and U_{oQ}), only the edges of the input (U_i) are shown to be jittery. The sampler which is active at the edges of the quadrature output helps to suppress most of the noise from demodulator output (U_d).

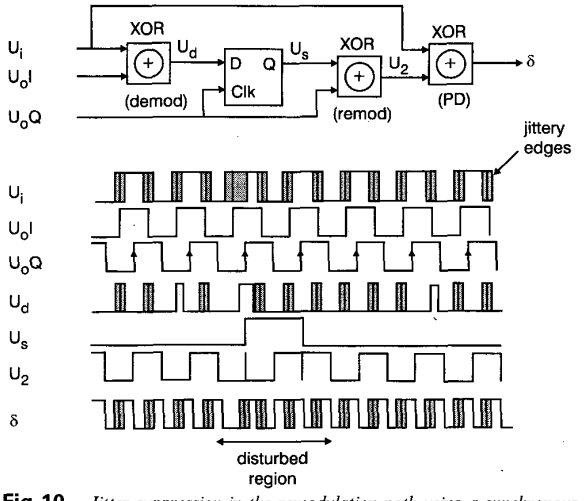

Fig. 10 *Jitter suppression in the remodulation path using a synchronous sampler*

Since the samples are taken at a distance of $\pi/2$ rad from the edges U_{oI}, and this is larger than $3\sigma_{\phi i} \approx 1.2$ rad, the probability of taking a wrong sample is very small. An erroneous sample causes a phase reversal of the remodulator output and leads to an unbalanced duty cycle (δ). Fortunately, the correct state is restored with high probability in the next cycle and the overall jitter increase is small. Actually, in more optimal carrier recovery techniques, the remodulation is done with an integrate-and-dump filtered version of U_d thereby further reducing the extra phase jitter. However, the delay of the integrate-and-dump filter (which equals, $T_b = 64\mu$s) needs to be compensated for by delaying the input to the PLL by an equal amount.

5.3 Input mixer

The 5MHz input IF signal has to be mixed down to a convenient frequency before applying it to the ADPLL. The nominal ADPLL input frequency (230kHz) can be obtained by using a second mixer input of 5.23MHz, which is easily generated from the 68MHz input clock using a divide-by-13 counter.

Since the input IF signal is hard-limited, a simple D-type flip-flop can be used as a mixer. In effect, this is a sampling process in which the mixer output is quantised to 5.23/0.23 ≈ 23 phase steps. Relative to an ideal square-wave of 230kHz, the timing uncertainty of the

transitions at the mixer output lies in the range [0, 1/5.23] μs. An estimate of the phase jitter variance at the mixer output can be obtained by assuming a uniform phase error distribution in the range [0, 2π/(5.23/0.23)] = [0, 0.276] rad. This gives a variance of $0.276^2/12 = 0.0064$ rad^2, which is negligible compared to the phase noise variance ($1/2SNR_i$ = 0.158 rad^2) that is already present in the input signal at a signal-to-noise ratio of SNR_i = 5dB.

5.4 Lock detection

A lock-detection indicator is required during acquisition to decide when to switch the ADPLL loop bandwidth to its tracking mode value. It is possible to use the output of the demodulating XOR gate for lock detection. However, since it is corrupted with phase jitter in the received signal and the ADPLL phase quantisation, some form of filtering is required for a reliable lock indication. The filter has also to be configured so that it detects lock as fast as possible and switches to the tracking mode in time.

Given the one-bit output of the demodulator, a low-cost filter can be built from a simple up/down counter which is initialised to its centre value and counts in either direction depending on the binary level of the demodulator output. The lock detection is tested by comparing the counter value with two threshold values (TH_{upper} and TH_{lower}). Two threshold levels are needed since the inputs to the demodulating XOR gate can be inphase or 180° out of phase up on lock.

Using an eight-bit up-down counter running at 1 MHz clock frequency, the thresholds are set at TH_{upper} = 225 and TH_{lower} = 31. With the counter initialised at its midvalue (128), at least 97 clock pulses (97μs) elapse from the moment lock is achieved till the lock detection flag turns on. When integration loss caused by noise is taken into account, the time taken to reach the threshold levels is slightly longer than 97μs for the case $SNR \geq 5$dB.

5.5 Data clock synchronisation

Following lock detection, the receiver starts to synchronise the local data clock (f_b = 15.625kHz) to the transitions of the incoming MLS data bits. This is done by triggering the data clock generator at one of the edges of the received synchronisation bits. For this purpose, the MLS data format provides five bits (Barker code = 11101) immediately after the carrier synchronisation interval (see Fig. 2). Due to DPSK encoding, all the 1s in the Barker code represent 180° phase transitions which can be detected in the receiver and used for triggering the local data clock.

The bit transition detection is done using the demodulator output. Filtering is necessary to reduce the effect of phase jitter and to have reliable transition detection. It is done using an up/down counter which is clocked at 4 MHz and operates in the range [0, 31]. When the counter content reaches the midvalue (16), a trigger pulse is generated. In the jitter-free case, this occurs 16/4MHz = 4μs later than the actual bit transition. Since the noisy demodulator output determines the up or down count mode, the counter occasionally loses some of its content, which delays the triggering moment beyond 4μs.

A statistical estimation based on the standard deviation of the phase jitter in the received carrier signal ($\sigma_{\phi in} \approx 0.4$ rad) showed that triggering occurs within 8μs for 99.7% of the time. This agrees well with simulation results which indicated triggering moment variation with a standard deviation of 3.2μs. Since triggering is done by loading a preset value that schedules the next data clock edge to occur after T_b = 64μs, it is possible to compensate for the integration delay by lowering the preset value. The compensation makes the timing uncertainty somewhat symmetrical (say ±4μs) and keeps it well within the MLS specification (±10μs synchronisation error).

5.6 Data filtering and DPSK decoding

Having synchronised the local data clock to MLS bit transition moments, optimal data filtering can be done using an integrate-and-dump filter. Once again, an up-down counter is used to average the one-bit output of the data demodulator. In the first version of this paper, the use of a simple integrating filter with lower and upper limits was suggested. However, later investigation showed that better BER performance can be obtained when dumping the counter content (resetting it to the midvalue) at the estimated bit transition moments. To average the demodulated data over one-bit duration (64 μs), an eight-bit counter clocked at 2MHz is used. It should be noted that the same filter hardware (counter) can be used in all the three modes of the receiver: lock-detection, clock synchronisation and bit filtering.

To complete the data extraction process, the filtered data bits are DPSK decoded by comparing the previous ($d[n-1]$) and current ($d[n]$) received bits. Since the timing of the incoming bits is known from the data clock, the DPSK decoder can be implemented with a single flip-flop (delay unit) and an XOR gate, i.e. $d_{out} = XOR(d[n], d[n-1])$.

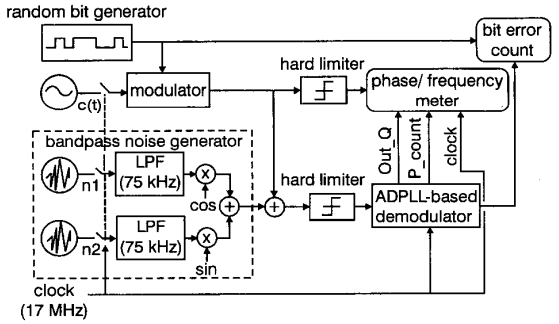

Fig. 11 *Simulation set-up for studying the performance of an ADPLL-based data demodulator*

6 Simulation results

To verify the performance of the data demodulator, simulations were conducted using a functional model of the ADPLL described in the C programming language. A block diagram of the simulation set-up is shown in Fig. 11. To account for the effect of atmospheric noise, two independent white Gaussian noise generators were applied. After lowpass filtering using second-order Butterworth filters of 75kHz bandwidth, a transformation to bandpass noise was carried out by shifting the spectrum to the frequency of the carrier signal ($c(t)$). The verification was done by measuring the phase/frequency relationships between the received signal and the ADPLL output and counting the number of bit errors for different SNRs.

The first simulation conducted was the study of the acquisition behaviour. With a nominal ADPLL fre-

quency of 230kHz and an MLS specification of ±25kHz carrier frequency uncertainty, there is a possibility that acquisition has to under go a frequency swing of up to 50kHz. Fig. 12 shows the simulation results for an input frequency change from 205kHz to 255kHz at *time* = 1.5ms. A wideband loop filter was used for the acquisition process by setting $K = 8$, which was later switched to 64 when lock was detected. The simulation also shows the situation when modulation is applied to the input carrier starting from *time* = 3ms.

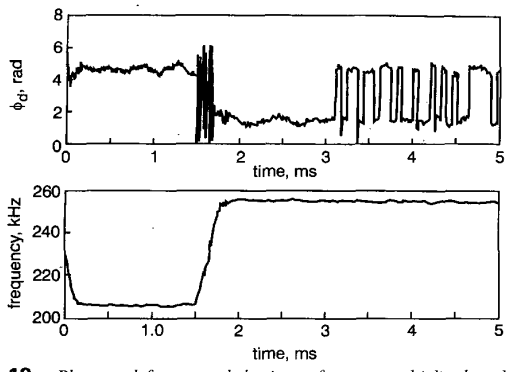

Fig. 12 *Phase and frequency behaviour of a rate multiplier-based second-order AD-PLL*
$K = 8$ and $SNR = 5$dB

The phase reading in Fig. 12 is computed from the delay between the rising edges of the noise-free reference and the quadrature ADPLL output (Out_Q), i.e., $\phi = 2\pi f_c \Delta t$. The frequency reading is obtained from the rate control signal (P_{count}) using $freq = f_c P_{count}/2NQ$, where the ADPLL parameters are $N = 32$, $Q = 1024$, and $f_c = 17$MHz. We observe that acquisition is completed within the allowed 832μs interval and the frequency transience is optimally damped. The displayed phase jitter is a superposition of the filtered phase noise from the received signal and that of the phase quantisation in the ADPLL. Simulations at very high signal-to-noise ratios ($SNR = \infty$), showed a peak tracking mode jitter of about $\pm\pi/N \approx \pm 0.1$ rad, which is attributed to phase quantisation errors.

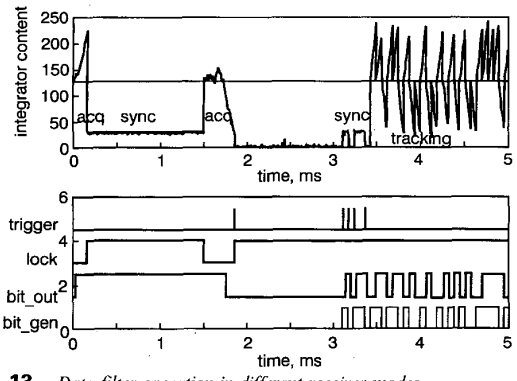

Fig. 13 *Data filter operation in different receiver modes*
Input SNR = 5dB

A second simulation studied the data filter operation and the BER. Fig. 13 shows the filter operation in the three receiver modes: lock detection (*acq*), data clock synchronisation (*sync*) and tracking. Compared with the acquisition curves, the timing of lock detection is seen to match the completion of acquisition. The trigger signal, which is used to synchronise the data clock, is shown for the five Barker bits. An estimate of the synchronisation error was obtained from 150 independent trials at $SNR = 4$dB. The standard deviation of the clock timing error was found to be about 3μs.

Finally, a BER estimation was conducted over 10000 independent simulation runs, each with 85 bits (slightly longer than the MLS packet). The results are given in Table 1 for four different SNRs. It can be seen that the MLS requirement of $BER < 10^{-4}$ is met for $SNR \geq 4$dB. The simulation was conducted using a local data clock with the synchronisation error mentioned above.

Table 1: Bit error rate performance of a second-order ADPLL based demodulator

SNR [dB]	2.0	3.0	4.0	5.0
BER	4×10^{-3}	1×10^{-4}	2.4×10^{-5}	$< 1 \times 10^{-5}$

7 Sea-of-gates implementation

The demodulator discussed in the previous Sections and a data validation functional unit were realised on a 1.6μm sea-of-gates IC. The functional units consume 22500 transistors, about 1/8th of the total number of transistors. The remaining chip area is used for other functional units of the GOLLUM integrated navigation receiver. The layout and circuit verification were done using local CAD tools like OCEAN [20]. The final metalisation of the sea-of-gates design has been completed by the Delft Institute for Microelectronics and Sub-Micron Technologies (DIMES) facilities.

8 Conclusions

A VLSI design of an ADPLL-based MLS data demodulator has been presented. The main motivation for this approach is the suitability of the design for integrated navigation receivers like GOLLUM [1]. The ADPLL hardware replaces the conventional semi-analogue approach or the computationally intensive fully software equivalent. Besides the performance improvements obtained using a fully digital design, the hardware approach relaxes the load of the central processor and increases the possibility of resource sharing among the software components of the navigation subsystems. This and the fact the proposed design is fully integrable with an application-specific processor like MOVE [11], lead to a low-cost and compact receiver implementation.

9 Acknowledgments

The author would like to thank colleagues in the Circuits and Systems Group for their direct or indirect support. Special thanks go to Prof.Dr.Ir. R.H.J.M. Otten for proposing the task and Ir. L.K. Regenbogen and Dr.Ir. R.C. den Dulk for their valuable ideas during the course of the work. I am also indebted to the anonymous reviewers whose suggestions improved the paper.

10 References

1 AARDOOM, E., and NIEUWLAND, A.: 'A single chip integrated navigation system', *Int. J. Navig.*, 1993, **46**, (1), pp. 95–104
2 ANOLL, R.: 'Integrated GPS/Loran-c receivers: a perspective on user acceptance, integration schemes and certification status'. Proceedings of the 22nd Annual technical symposium, The Wild Goose Association, 1993

3 BREEWER, E.J., MOELKER, D.J., and VAN WILLIGEN, D.: 'The MLS integrated approach system: Hybrid position calculation and initial flight tests'. Proceedings of the ION 49th Annual Meeting, The Institute of Navigation, 1993
4 HAGER, J.B., and van CLEAVE, J.R.: 'Task III report-low cost airborne MLS receiver'. Technical report, Ames Research Center, 1980
5 Bendix General Aviation Avionics Division: 'Maintenance manual ML-201A MLS', 1984
6 KOCKZO, S., and CLARK, A.W.: 'A DSP-based microwave landing system post-IF processor'. Proceedings of IEEE DASC, 1991, pp. 351–361
7 ANTENEH, A.A.: 'Design and implementation of MLS data demodulation and processing unit'. Technical report TWAI096-03, Delft University of Technology, Circuits and Systems Group, 1996
8 REDLIEN, H.W., and KELLY, R.J.: 'MLS: The new International Standard', *Adv. Electron. Electron Phys.*, 1981, **57**, pp. 311–410
9 BENEKE, J., WIGHTMAN, C.W., OFFT, A.M., and VALLONE, C.B.: 'Time reference scanning beam multimode receiver'. Technical report FAA-NO-78-84, Calspan Corporation, 1978
10 EL-ARINI, M.B., and ZELTSER, M.J.: 'Analysis of the integrity of MLS data functions', *IEEE Trans.*, 1987, **AEA–23**, (4), pp. 473–483
11 CORPORAAL, H., and MULDER, J.: 'MOVE: A frame-work for high-performance processor design'. Proceedings of Supercomputing-91, 1991,
12 GARDNER, F.M.: 'Phase lock techniques' (Wiley, New York, 1979, 2nd edn.)
13 LINDSEY, W.C., and CHIE, C.M.: 'A survey of digital phase locked loops', *Proc. IEEE*, 1981, **69**, (4), pp. 410–431
14 BEST, R.E.: 'Phase-locked loops: theory, design and applications' (McGraw–Hill, 1993)
15 IRITANI, T., KINOUCHI, Y., and USHITA, T.: 'Linear DPLLs using integrators in a pulse frequency-modulation system', *IEE Proc. F, Commun. Radar Signal Process.*, 1982, **129**, (5), pp. 352–358
16 R.C. den Dulk. 'An approach to systematic PLL design'. PhD thesis, Delft University of Technology, 1989
17 ROSINC, W.B.: 'All-digital phase-locked loops using the 74HC/HCT297 IC, designer's guide'. Philips Components, 1989
18 LINDSEY, W.C.: 'Performance of phase coherent receivers preceded by bandpass limiters', *IEEE Trans.*, 1968, **COM–16**, (2), pp. 245–251
19 PROAKIS, J.G.: 'Digital communications' (McGraw–Hill, 1989, 2nd edn.)
20 GROENEVELD, P., and STRAVERS, P.: 'OCEAN: The sea-of-gates design system. On-line users manual'. Delft University of Technology, Circuits and Systems Group, 1993