Drum Optical Encoder Driven Level-wind Stepper Controller

Introduction

This paper describes the operation of a level-wind system for a glider winch that emulates a reversing screw behavior, such as is used in fishing reels, using a recirculating ball screw positioning stage driven by a stepper motor. Such level-wind systems have been implemented using such reversing screws driven by a chain or cogged belt from the drum shaft. This approach is an alternative approach to accomplishing this same behavior with uniformly spaced interrupts at a 100 kHz rate. In this new approach, the interrupts are generated from an optical encoder that is sensing the drum position, generally at a point further up the drive system, to realize higher position resolution and encoder pulse rates than if directly sensing the drum shaft position. First some preliminaries are covered before a block diagram illustrating the processing involved. Then some topics deferred in the main body are addressed. There is also an appendix that contains an involved mathematical analysis needed to determine some of the key parameters for this system.

Preliminaries

Readers are assumed to be familiar with the prototype winch system in development. Later editions of this paper will attempt to provide more details for readers not so intimately involved with this development. The prototype will use two (initially one) Siemens 4-pole ac motors controlled by an iFOC controller controlling their output torque. The base speed for these motors is around 3,000 RPM but we will use them in some phases of the launch to slightly over 6,000 RPM. The speed of these motors is reduced by a two-stage reduction system that will take the 3,000 RPM motor base speed down to about 500 RPM drum speed for a cable speed of about 20 m/s.

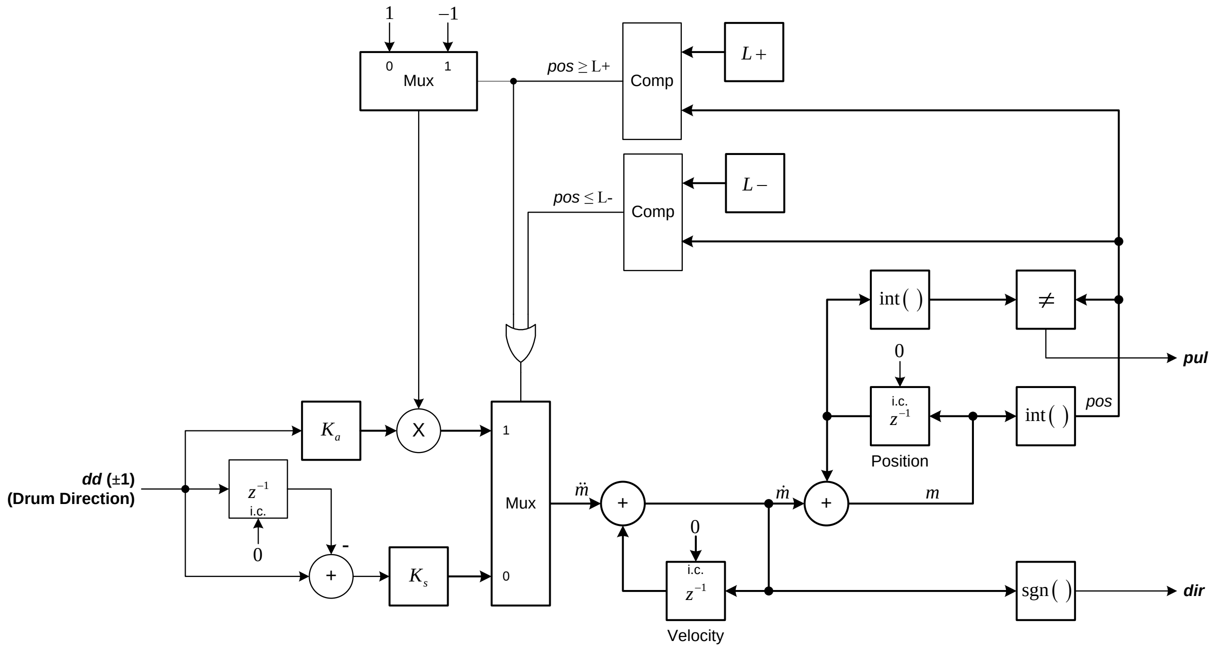
Respecting the 6,000 RPM limit of the encoder on hand, the encoder is assumed to be coupled to the shaft that drives the drum driveshaft. Its speed will be reduced by about a factor of 1.5:1 from the motor speed. This also reduces the encoder edge rates into what I believe are slightly more favorable ranges. The drive motor speed at 6,000 RPM would be geared to the drum such that the rope speed would be about 40 m/s. This intermediate shaft would be turning about 4,000 RPM here. For this discussion I will assume the encoder I have providing 360 PPR. If only one encoder edge is used to generate interrupts, the interrupt rate near maximum speed would be 24,000 interrupts per second. If 2 edges, 48,000. This rate is less than half that was being considered for the time-based approach.

For a 2X working level-wind factor, it was previously established that the sweep speed would be about 100 mm/s for a cable speed of 20 m/s. For a 40 m/s maximum rope speed, this would be about 200 mm/s. One rotation of the stepper moves the level-wind 20 mm. Assume we are using the 2000 microsteps per stepper shaft revolution driver settings. (Going forward, I will use steps and microsteps synonymously.) At this setting, one step corresponds to 10 um of level-wind motion. The number of steps per second for 200 mm/s then is about 20,000 steps per second which translated to a stepper speed of 600 RPM. A requirement is the number of steps per interrupt must be no greater than one as the ISR can only issue at most 1 step command per interrupt. This requires the maximum step rate be less than the corresponding maximum interrupt rate. That is just satisfied for 1 edge and well satisfied by 2 edge operation. If the level-wind factor needs to be increased above the current 2X baseline very much, this requirement could be violated for 1 edge encoder processing (24 kHz maximum interrupt rate). There is also some reduction in jitter realized by using a higher 2-edge interrupt rate. So for the discussion to follow I will assume we get interrupts based on 2 edge encoder processing. It would be very easy to change this during testing if we want to explore 1 edge (or 4 edge) processing.

With this choice, roughly the ratio of microsteps per encoder step is around 20,000/48,000 or about 5/12 or 0.417 steps per encoder edge/interrupt during the sweep period. (Super precision here is not needed.) The reversals take place over a relatively short distance. A typical value might be 2.5 mm. 2.5 mm corresponds to 250 microsteps. For comparison, the drum width is planned to be 165 mm. That distance corresponds to 16,500 microsteps. This concludes the preliminaries that needed to be covered.

Reversing Screw Operaton

Below is a block diagram for the processing required for this encoder triggered interrupt approach.



This discussion will initially assume no abnormal conditions, e.g., loss of synchronization, and defers discussion of initialization until the normal operational mode is covered. This system is depicted using z^{-1} elements which in DSP processing normally represent a unit sample time delay at a fixed sample rate. Here they represent a one interrupt delay but the interrupt rate can vary from zero to the maximum rates discussed above.

The rightmost digital integrator's register, termed the Position integrator, holds the current level-wind system position in integral microsteps plus a fine fractional component. This value is termed m , *m* referring to microsteps. The integral component must support the number of microsteps across the operating range so the 16,500 value developed above says it could possibly be supported by a 14-bit value. But the range may not be well centered so 15 bits would be better. Since this is so close to 16 bits and the processors we are using are generally 32-bit, the starting plan is to use the upper half of the 32-bit register for the integral component and the lower half for the fractional component. The preceding integrator, termed the Velocity integrator will generally be less than 2^{16} in magnitude but it needs a sign bit so it will employ a 32-bit word size also. The magnitude is constrained to meet the requirement that only one microstep can be issued per interrupt.

Along with each encoder edge interrupt, a binary indication of the winch drum's direction of travel is required. Drum travel in the positive direction will be considered that which pulls in cable but that will not be important for this discussion. (When odometer processing is considered it will be important. The drum direction input, *dd*, will be a binary value. To simplify the depiction of the processing, it will be assume here to be values of +1 for a forward/positive direction and -1 for travel in the reverse/negative direction. In the drawing, the lighter/finer lines and blocks are associated with binary (not necessarily 0 and 1 as just indicated) values and operations. The heavier lines represent numerical values represented with more than 1 bit.

There are 3 regions the level-wind may be in. Most of the time the level-wind rollers will be sweeping across the face of the drum at a speed proportional to the drum speed. That can be in either direction. When the sweep gets near an inner drum flange, the travel direction must reverse and the level-wind rollers start sweeping in the opposite direction. Most of the time the drum speed will be approximately constant during this reversal action (and commonly even across a sweep). In this case, the scheme to be described will reverse the travel direction over a short, fixed distance at a constant acceleration rate. As will be discussed, this reversal distance is constant independent of the drum speed or even if it varies during the reversal period. The acceleration magnitude increases (quadratically) with higher drum speed magnitudes. This is essentially the same behavior as that of a reversing screw mechanism as used in fishing reels. This system will even work just as such a reversing screw fishing reel if the drum is operating near zero speed and between positive and negative velocities, e.g., if the drum were held by hand and moved back and forth.

Two parameters determine when to start the reversal, one for each side of the drum. The first parameter is L+ and it defines the level-wind position (in integral microsteps) where the reversal action begins for the positive side of the drum. (The side that is in the direction of positive level-wind travel.) L- similarly indicates the start of the reversal region when moving in the negative direction. This integral value, the upper 16 bits of the Position integrator register, is termed *pos*. There are two comparators employed to determine which region the level-wind is currently in. One compares the current integral position value *pos* to L+ and the other to L-. The comparator output for the comparison against L+ is true if pos≥ L+. This signifies the level-wind is in the positive reversal region. Similarly, the one comparing against L- outputs true if *pos* ≤ L- when in the negative reversal region. These are ORed together to signify that the level-wind is in either of the reversal regions. When in either of these regions, the lower multiplexor selects its upper input. When in the sweep region, its lower input is used.

Operation in the Sweep Region

First the normal sweeping behavior is discussed. This is when the lower input signal is selected by the lower mux. Assume the drum is and has been drawing in rope at some reasonable and fixed speed. Then *dd* will be and has been for a while positive and its signal line is and has been at a value of +1 for some time (at least one prior interrupt). The digital differentiator on that channel will be outputting a value of 0 as *dd* has not changed recently. So the value output from the mux will be 0. This value is the velocity increment $\ddot{m}$ input into the Velocity integrator so its value will not be changing corresponding to a fixed sweep speed. (The first integrator is referred to as the Velocity integrator. But technically that is a misnomer. Velocity is the rate of change of position with respect to time. Here it is not with respect to time but to encoder edges. If the drum speed is constant, as has been assumed here, the encoder edge rate is constant and this is indeed proportion to the sweep velocity so I use this simple nomenclature. Similarly, its input, $\ddot{m}$, is not technically an acceleration unless the drum speed is constant. In both such cases, these are only proportional to velocity and acceleration scaled by the current edge rate (and direction).) To be further elaborated on later, the velocity integrator output value $\dot{m}$ is and has been a positive value K_s, (*s* for sweep) while it has been sweeping in the positive direction. This positive parameter is the number of microsteps per encoder edge value previously developed above expressed as a signed 32-bit integer with the binary point just above the 16th lsb. So, its value would be about $5/7*2^{16}$ = 27,307. (Further restrictions on actual allowable values will be added shortly but they are not relevant yet.) Every time there is another interrupt, this value increments the Position integrator. The Position integrator employs a 32-bit signed register with the same binary point location. For the assumed value of K_s, every 2 or 3 interrupts, the fractional part will overflow and increase the integer *pos* value by 1. When such an overflow occurs, a stepper micropulse is generated. This results in a pulse on the *pul* output line. This is effected in the block diagram by looking for a change in the *pos* value, again the upper half-word of the 32-bit register, between the current *pos* value and its previous value. (For travel in the negative direction, this function also detects a borrow action associated with a decrease in the *pos* value.) The positive sign of the velocity input $\dot{m}$ indicates to the stepper driver the direction this step should be in. The direction input to the stepper is name *dir*. This continues so long as the drum turns in the positive direction and the level-wind does not enter the positive reversal region. If the drum speed increases/decreases, the interrupt speed increases/decreases proportionally, and the sweep speed increases/decreases proportionally with the scale factor set by K_s. This includes the drum slowing to a complete stop. Then the encoder interrupts stop and the level-wind stops.

Now considered is when the drum speed drops to zero and then continues to decrease—meaning the drum is now rotating in the negative direction. When the drum first starts turning backward, *dd* goes to -1. The first time this happens, the current *dd* value into the differentiator is -1 and the value in the delay element is +1. But this delayed value is subtracted from the current value resulting in a value of -2. This is multiplied by the scaling factor K_s resulting in the value out of the multiplexor being $-2K_s$. This is added to the previous value, K_s, in the Velocity integrator so the velocity value $\dot{m}$ changes from K_s to $-K_s$. This negative value is integrated by the Position integrator resulting in its value decreasing. When such a subtraction results in the integral position value *pos* decreasing (by 1), the carry/borrow detector generates a stepper pulse command, *pul*. The Direction value is the sign of the velocity output, now negative, so the stepper moves in the negative direction. On the next interrupt with the drum velocity still negative, the current and prior *dd* values are both -1 and so the input differentiator produces a 0 output and $\dot{m}$ remains $-K_s$. This value remains fixed indefinitely as long as *dd* remains negative. Should the velocity return to 0 and go positive, on that transition the differentiator will output a value $2K_s$ and the velocity integrator value will toggle back to K_s. So the only two values the velocity output can take on are $\pm K_s$ so long as the level-wind remains in the sweep region. As previously noted, if the drum were moved back and forth by hand through zero, the level-wind would move back and forth proportionally with the drum deflections. This concludes the description of operation in the sweep region.

Operation in the Reversal Regions

Now the reversal behavior is described. Assume the drum is traveling in the forward direction at a constant speed. Eventually, the stepper position will increase to and become equal to L+. When this happens, the lower mux switches to the upper input. The level-wind is now in the positive reversal region with the upper comparator asserting true. That gates the -1 value through the upper mux. *dd* is still +1 and that is multiplied times a positive parameter K_a, the *a* is for acceleration. This parameter determines the acceleration (actually deceleration) magnitude during a reversal. This parameter will generally be much much smaller than K_s. The number of encoder interrupts between L+ and the farthest travel point is now shown to be K_s / K_a, call this ratio N_r. In the deferred topics section to follow, it will be shown that K_s needs to be an exact multiple of K_a so that N_r is an integer. Returning to the behavioral description during the reversal, since the drum's direction of travel is positive, the value into the multiplier is $+K_a$. The value out of the upper mux was -1 so the value into the lower mux and gated through to become the acceleration increment $\ddot{m}$ is $-K_a$. When this value is integrated in the Velocity integrator, this reduces the value of $\dot{m}$ slightly below K_s. On subsequent interrupts, this value continues to drop linearly (with interrupt edges) until it reaches 0. This will occur N_r interrupts after the reversal begins. Remember the drum speed was assumed to remain relatively constant during this reversal so the rate of encoder edge interrupts remains constant. But the decreasing value of $\dot{m}$ results in the stepper pulse rate (again with respect to encoder interrupts) dropping until the stepper velocity value reaches 0. At this time, the level-wind will have reached zero speed and the level-wind position is at its greatest value. That value will be determined shortly. But *dd* remains +1 and the level-wind position is still greater than $L+$ so the velocity increment value $\ddot{m}$ remains $-K_a$. The encoder interrupts keep occurring at the same rate so the velocity output of the first integrator continues to fall linearly but now the *dir* output has gone negative so step pulse commands, at an ever increasing rate now, are now in the reverse direction. *pos* now begins to fall and the stepper rate continues to increase until the velocity value $\dot{m}$ reaches $-K_s$. By symmetry, the distance traveled in the positive direction as $\dot{m}$ fell to zero will now have been traversed in the negative direction so the Position value will once again reach $L+$ but now traveling in the reverse direction at speed $-K_s$. The upper comparator will deassert and the system is now in the sweep region traveling at speed $-K_s$. When the sweep reaches position $L-$, the dual reversal process ensues. Here the drum travel direction is still positive but the upper mux will be passing the +1 value so the acceleration $\ddot{m}$ employed for the reversal will be $+K_a$. The velocity input $\dot{m}$ on entry was $-K_s$ so it will increase linearly reaching zero N_r interrupts later when the travel in the negative direction is at its extreme value and then become positive finally reaching $+K_s$ as the level-wind exits the negative side reversal region and is now sweeping forward again. Should the drum be rotating in the negative direction, the behavior is the same except the directions are reversed. It is the product of the *dd* and the upper multiplexor output that selects the correct sign for the acceleration value $\ddot{m}$ for all situations.

Determination of K_a, K_s

The remaining question is what is the distance traveled during the reversal period? More precisely, how much does the position value change after reaching $L+$ or $L-$ until the velocity reaches zero. This value has been determined by analysis, included as an Appendix, to be

$$\Delta P = \frac{N_r\left(N_r - 1\right)}{2} K_a = \frac{N_r - 1}{2} K_s \approx \frac{N_r}{2} K_s \qquad N_r \gg 1$$

(This is closely related to the distance traveled by an object under constant acceleration being $0.5At^2$. The ½ factor comes from this being the area under a triangle associated with the linearly increasing (or decreasing) velocity.) ΔP is properly in units of microincrements and, as shown in the Appendix, will be an integer. So this distance traveled is fixed regardless of drum speed and proportional to $K_s = N_r\, K_a$ which determines the level-wind factor. Previous analysis shows the actual acceleration employed here increases quadratically with increasing drum, hence sweep, speeds. With a little algebra, K_a can be chosen to set this distance as given by

$$K_a = \frac{K_s^2}{2\left(\Delta P + K_s\right)} \approx \frac{K_s^2}{2\Delta P} \qquad N_r \gg 1$$

N_r, the number of interrupts required to take the stepper "velocity" to zero is given by

$$N_r = 2\left(\frac{\Delta P}{K_s} + 1\right) \approx 2\frac{\Delta P}{K_s} \qquad \frac{\Delta P}{K_s} \gg 1$$

which is generally true. In practice, an approximate value for K_s is chosen based on the level-wind factor desired and the turn around distance is determined by how much acceleration of the inertial load can be supported by the stepper at the maximum drum speed. That provides an approximate value for N_r which must be rounded to yield an integer K_a value. That value multiplied by N_r is the actual value used for K_s.

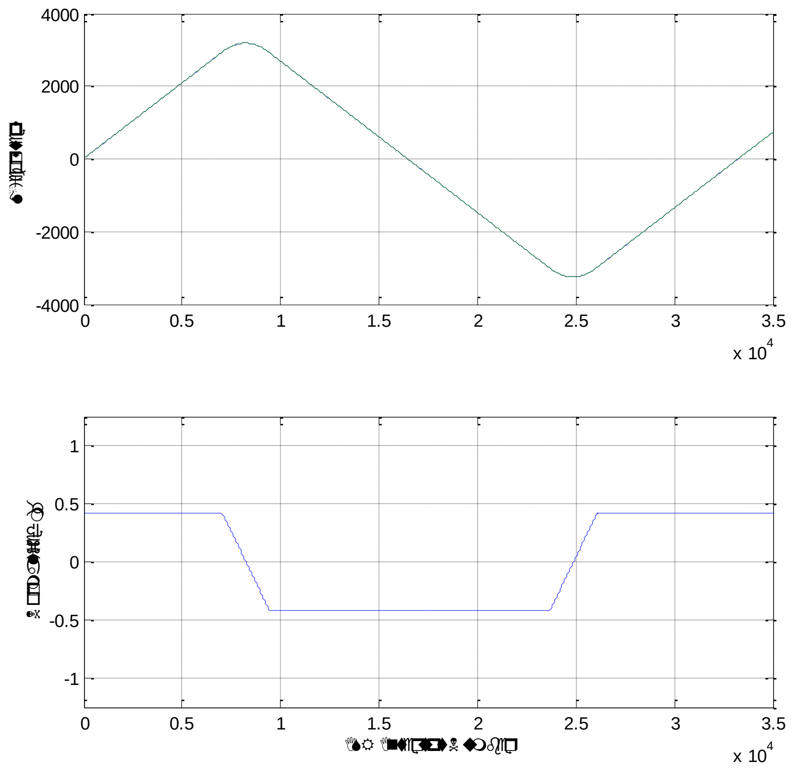
Note that the reversing period, in interrupts is actually $2\,N_r$ as N_r interrupts only get the velocity to 0, then another N_r interrupts are needed to get the speed back up to the reverse sweep speed.

Just to emphasize, these computations are not done in real time during the interrupts. They will selected and adjusted during field trials and will likely be hard coded. (It is possible there may be some way provided to set $L+$ and $L-$ for adjusting the level-wind travel for field maintenance and allowing K_s and K_a to be similarly adjusted would be simple to include.)

What is a typical value for this parameter? As noted in the preliminaries, a typical value for ΔP is 2.5 mm. But ΔP here is in units of microincrements. A microstep is 10 um but there are 2^{16} microincrements per microstpep, ΔP here is about 16,384,000 microincrements (250 microsteps). So the normalized (to 2^{16}) K_a is about 3.47E-4 as compared to about 0.417 for the normalized K_s. So N_r, the ratio of these, is about 1,200 indicating the reversal period in interrupts would span about 2,400 encoder interrupts. In terms of the 32-bit increment value for the Velocity integrator, the value used for K_a would be about 23. Comparison with the value of K_s, 21,800, demonstrates the assertion that

$K_a \ll K_s$. This is pretty coarse but a lot of accuracy is not needed here. This just goes to the distance in edge interrupts from $L+$ to the farthest point traveled during the reversal. In practice this reversal distance will be determined by trial and error during field testing as will be K_s, $L+$, and L-. If we should want finer resolution, we could shift the binary point up one, possibly 2, bits but this would complicate separating the fractional and integer parts of the Position integrator for determining when to emit a pulse. As noted, K_s must be an integer multiple of K_a, that would require here that K_s must be a multiple of 23. But the fractional change in K_s around its nominal value of 21,800 is minuscule, about 0.1%. To emphasize that high precision here should not be important, we are debating whether the level-wind factor should be closer to 2 or 3 and K_s is proportional to the level-wind factor.

The descriptions above are tedious. An example of the overall behavior over a full level-wind cycle is instructive. A Matlab script was created to simulate the above described system. The sweep distance, defined by L+ and L-, was made artificially short (about ±3,000 as compared to actual values around ±8,250) to make the reversal region behavior more evident but otherwise the response is the same as would occur using this scheme with the real hardware. Below are traces showing the key behaviors.



Operational Example

The abscissa is the edge interrupt number. The upper trace is the position in microsteps during the back and forth sweep. (The sweep distance has been artificially made very small to better portray the behavior during the reversal periods.) The quadratic trajectories during the reversal periods is normalized clearly evident. The lower trace shows the normalized $\dot{m}$ trajectory. During the sweep period, the value is approximately 5/12 = 0.417. Note that it is constant during the linear sweep periods and ramping linearly during the sweep periods.

In the deferred topics section immediately following, the need for K_s to be a multiple of is demonstrated). A similar plot is used there to show the behavior when this condition is not satisfied.

Deferred Topics

Initialization

The scheme proposed below is not the most obvious that might occur to a reader at this point but it anticipates other needs that are subsequently developed in the sequel. These include indexing and loss-of-synchronization on-the-fly recovery. Hence a scheme is proposed in anticipation of these later needs and the reasons for the values used should become more apparent as the development proceeds. Also assumed here is the level-wind rollers physical position is between L+ and L-. How to gracefully deal with cases where the position may not be within these limits is heavily dependent on the details of the physical system and this will not be addressed in this paper. It is also known that the stepper driver that is expected to be employed in the hardware realization has setup requirements on the dir level output relative to the pul pulse output. This is readily dealt with in actual implementations and is not discussed further here.

Initialization primarily means getting the system memory elements initialized but also means getting the parameters $K_s,\, K_a,\, L+, \text{and } L-$ set to the proper values. During initialization the drum is not necessarily moving so an alternate source of simulated encoder edge interrupts will be needed. If that source of interrupt's frequency can be set properly, the initialization and indexing can simply use the operational values for these values. That is assumed here but could be readily accommodated if that degree of freedom is not available, e.g., a timer for a serial port is used and its frequency choice is not free.

So without further ado, before interrupts start the velocity integrator's initial value $\dot{m}$ is set to 0 and the position integrator's initial value m is set to $L- \;-K_a\, N_r\left(N_r-1\right)/2$ microincrements. By the equation for distance traveled during the reversal given above, this is precisely the distance to the negative side of $L-$ where a normal reversal would reach its negative-most value. The differentiator's initial memory value is a don't care. What has been done here is to tell the system the level-wind rollers are at their negative-most point of travel (but again they are assumed to be between the limit switches to this point) with velocity 0 as would be the case if it was there during normal operation. This completes the initialization needed prior to indexing and we are ready to enable the encoder proxy interrupts now.

Indexing

Indexing is the process of aligning the Position integrator value m with the actual level-wind physical position. This is the most readily done using a limit switch to indicate when the level-wind rollers are at a known location. If that limit switch were physically located so as to trigger when the rollers were at the center of the sweep, conceptually what needs to be done is to sweep the rollers such that this position is passed and, when that event occurs, the Position integrator value m would be overwritten and set to 0. Conceptually this is fine but there are issues and other desired uses for the limit switches that dictate otherwise. The first is that the trigger point of a limit switch is typically dependent on the speed and direction of travel of the level-wind mechanical elements. The simplest way to deal with this is to always pass the limit switch trigger position moving in a specified direction and at a specified speed. Since all we have assumed is we know that the level-wind rollers are somewhere between where reversals would begin, it behooves us to locate the indexing limit switch trigger position somewhere very close to one of the reversal points. Let assume we have physically positioned the indexing limit switch near the point where we initiate a reversal associated with L+ and, to allow it to be used for detection of loss of synchronization (described shortly), slightly towards the sweep center from L+. But anticipating a need for a second limit switch near L-, assume it is also present and near, but slightly interior to, the negative reversal point. This completes the setup for the indexing procedure.

What we need to do now is to sweep the system in a positive direction towards this limit switch at a fixed, and typically relatively low, speed. So for this indexing operation, we employ a timer producing interrupts at a known rate and force *dd* to +1. This would simulate the drum moving in the positive direction at a known speed that results in a known level-wind sweep speed suitable for this indexing operation, i.e., relatively slow. With the drum direction set to +1 and the system initialized as described above, the system would conceptually move towards the L+ indexing limit switch at the desired known rate. Since we assumed the level-wind rollers were somewhere between the reversal points, having set m to $L\text{-} - K_a N_r (N_r - 1)/2,$ its actual physical position would correspond to a numerically greater value than the value in the position integrator. We now enable the simulated encoder interrupts. The system believes it is in the negative reversal region with a positive drum travel so the system uses the value K_a for the $\ddot{m}$ acceleration input. (Since the lower multiplexor channel is not initially used, its differentiator's memory element gets initialized here to +1 which is why its initial value was a don't care above.) The stepper speed ramps up over N_r interrupts to K_s just as m reaches $-L\text{-}$ and the system then sweeps at this rate across the drum face. Eventually, the L+ limit switch is traversed. This will occur before m reaches $L+$ because the physical system started between the limit switches. Now this limit switch event is employed to overwrite m to the known position value of the L+ limit switch and the system is now indexed. Because this limit switch was positioned near but interior to the actual reversal position, the known value used for the overwrite will be nearly equal to but less than $L+.$ But the system is still moving and we generally cannot stop instantaneously. But we are almost immediately at the positive reversal point and the system will decelerate the stepper to 0 for us. We could stop as the velocity reaches 0 and be done. But for the purposes of self-test, the system can be allowed to completely reverse and make a reverse sweep across the drum. During that transit, it will pass the L-limit switch and its proper operation can be confirmed then as it would also have a known position so the validity of the L+ indexing can be further validated. Then as the level-wind reverses in the negative reversal region we can stop when it reaches 0 velocity there. The simplest way to stop things now is to disable further simulated interrupts. Now m would once again be at the value $L\text{-} - K_a N_r (N_r - 1)/2$ but now that correctly corresponds to the level-wind mechanism's physical location.

If before the initialization/indexing operation was started, typically at the beginning of the day, the operator manually adjusted the drum position such that a wrap was coming off the drum at the flange corresponding to the negative reversal direction. The drum wraps would also be aligned for operation, at the beginning of the day this would be for the initial rope pull out to the glider area. So the level-wind system and drum would now be fully ready for operation. Interrupts for the drum encoder would now be enabled and the normal operations would commence. But given the retrieve tension is light, it is thought for retrieves it may be simpler to just center the level-wind rollers during the retrieve pull outs. Further thoughts on all the implications of this are still required.

Loss-of-Synchronization Detection and Recovery

During operation, the stepper can pull out of lock if the loads, either rope induced or inertial, are beyond the stepper's capacity to support. This is termed a loss-of-synchronization, LOS for short, event. Accelerating the inertial loads too fast is the reason the system limits the acceleration during reversals. That was also why for indexing, we employed the proposed systems intrinsic ramping velocity capability to limit accelerations as the level-wind motion was initiated with fixed rate faux drum encoder interrupts. When a stepper pulls out of lock, typically it does not slip one stepper cog (about 1.8°) but many. In particular, if it loses lock due to excessive rope side loads, it will likely slip almost to the drum face center and then stall (stop) even though pulse commands continue. When that happens, the Position integrator will continue to emulate where it thinks the level-wind mechanism is and even start the reversal process when it passes either L+ or L- (depending on the sweep direction). With the pair of limit switches placed so as to trigger just before the mechanism reaches L+ or L-, then an indication that synchronization has been lost would be that the Position integrator value *pos* indicates reversal region has been entered but the associated limit switch did register a passage. Furthermore, whenever a limit switch triggers, the value of *pos* at the trigger time can be checked to make sure it is within some tolerance of where it was expected. So a LOS event can be either 1), a limit switch did not trigger before reaching L+, when traveling in the forward direction, or L-, when traveling in the negative direction or 2) the value of *pos* when a limit switch triggered was beyond some tolerance of where it was expected to be. To simplify the description to follow, assume a type 1 LOS event occurred on the negative sweep, i.e., the pull-out occurred before the negative limit switch was passed and then the L- comparator (not limit switch) asserted true. For this discussion, we assume the mechanism was pulled back to the center and the motor stalled. The dual of the recovery behavior to be described would occur for loss during a positive sweep. Normally, to deal with a stalled motor, we would have to assume the motor mechanism is at zero speed and manually accelerate it back up to speed using simulated encoder inputs. But fortuitously, the standard reversal activity just initiated by *pos* passing L- will do this for us automatically. It ramps the Velocity integrator speed $\dot{m}$ towards 0 and then accelerates it to the positive sweep speed for the return sweep. But as $\dot{m}$ matches the physical stepper speed, assumed 0 now, the stepper will pull back into lock and start following $\dot{m}$ again. So that action neatly gets the stepper out of stall and moving in the proper direction but the system has lost index, i.e., the Position integrator position does not correspond the physical mechanism's position. As noted, under excessive side loads the mechanism will likely have slipped to near the drum's center. So the ISR algorithm thinks it is sweeping back near from the negative-most inner flange but, under the assumed scenario, the physical mechanism is starting to sweep from near the drum's center. What would happen now is the roller assembly will trigger the limit switch near the positive drum flange long before expected; based on the *pos* value. But this is simply a type 2 LOS event, the trigger occurred with *pos* not near where expected for the associated limit switch trigger point. When this occurs, we use the positive limit switch to re-index *pos*. Because the sweep speed is likely different than what is used for normal indexing, this won't be as accurate but hopefully it will be close enough to continue operations for a launch in progress. (Simple tests can be done to see how sensitive the re-indexing is to speed and possibly speed dependent corrections applied.) But at least we are traveling in the proper direction for indexing with this limit switch. But when the L- comparator event occurred without an immediately preceding L- index switch action, a sticky bit indication that we had a LOS event would be set and the operator informed. Before a subsequent launch would be allowed to be initiated, a normal indexing operation for the level-wind system would be required require to clear this sticky bit and allow launches to be initiated again. If the LOS occurred during a positive sweep, the dual of this recovery behavior would occur. Here we would approximately re-index when the L- limit switch triggers. So for LOS recovery, both limit switches are employed to detect LOS and approximately re-index the system. So, other than noting that an LOS has occurred, the only action that needs to be take during a launch is to re-index on either limit switch when the stepper is moving in the proper direction for indexing with that switch. Note the drum traveling in the negative direction glider kiting needs to be handled properly here. The logic is simple, if $\dot{m}$ is positive, re-index on the positive positioned limit switch triggering on a positive stepper direction passage. If negative, re-index on the negative positioned limit switch on a negative stepper direction passage.

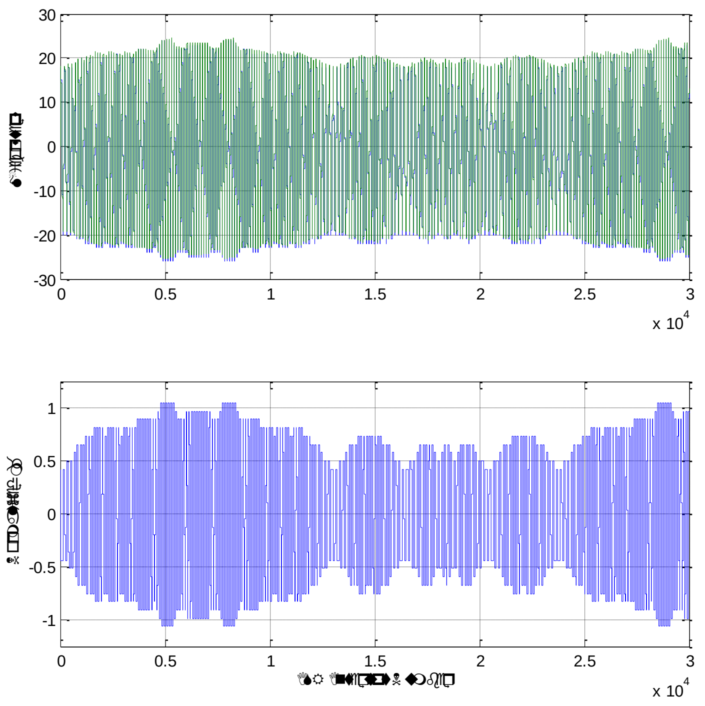
It has been asked why not use the limit switches' interrupts to initiate the reversals. There are several reasons for not doing this. The first is that it would require the limit switches to be physically moved to adjust the reversal points. This would probably make desirable some sort a method to finely adjust the limit switch trigger position with something like a screw adjustment that could be then be locked down. But with the scheme proposed above, adjusting the reversal points only requires changing software parameter values and the positioning of the limit switches can be much coarser, i.e., then limit switches do not have to be precisely set and easily mechanically tweaked. The second, and much more important reason, is this would make LOS detection and recovery much much more difficult. Once initialized and indexed, the system proposed runs essentially open loop. It would not need the limit switches at all except for indexing and LOS detection. Once indexed, it *thinks* it knows where the level-wind physical mechanism is at all times and does the reversals at those emulated positions. The limit switches' primary use (other than for indexing) is for LOS detection and recovery. If the limit switches were used to initiate the normal reversals and the stepper had stalled, the limit switch trigger event would never occur. Then we would still have to monitor the simulated position to see if the limit switch triggered near where it was expected. The third reason is my inherent bias regarding the reliability of mechanical elements like switches and potentiometers. Discussions are ongoing regarding testing for proper switch behavior but, should a switch fail during a launch, operation should be able to continue just ignoring the switches for the duration.

One thing that must be considered is limit switch debouncing. We don't want to get multiple interrupts from what is really a single transit of the switch. The switches planned to be used have both NO and NC contacts. We could implement a hardware switch debouncer using an RS flip-flop type scheme. But we should be able to implement software debouncing and avoid that hardware.

Integral Relationship Between K_s and K_a Requirement

It was asserted above that K_s should be a multiple of K_a. The simple explanation is that if this is the case, the ramping velocity will pass precisely through 0 N_r edge interrupts after the reversal is started and then reach the negative of its initial value N_r interrupts later. Then the sweep values for both directions will be precisely $\pm K_s$. If not, the extrema values of $\dot{m}$ will begin to wander away from where they should be as is now demonstrated.

A simulation similar to that above was run with non-integrally related values, K_s equaled 5 and K_a equaled 27 and assumed the position integrators fractional part is only 6 bits long. L+ and L- were 17 and -19 respectively. These values were used to accelerate a slowly divergent behavior. The results with these values are shown below.



Non-Integrally Related Parameters

There are many many sweeps here but the key behavior to observe is the normalized $\dot{m}$ values. They start and should remain transiting between ±0.422. But they diverge and wander about meaning, at a minimum, the level-wind factor is being modulated. Worse yet, around the 5000th interrupt value the value exceeds 1 which means stepper pulses are being dropped. Looking at the position trace, the reversal points start moving. It appears given long enough, around 23,000 interrupts, this whole pattern reverts and then starts to repeat. Given enough time, I'm sure a mathematical basis for what is going on could be developed but that is not pursued. As noted above, satisfying this constraint has little practical impact so its not worth the effort—the constraint will be observed.

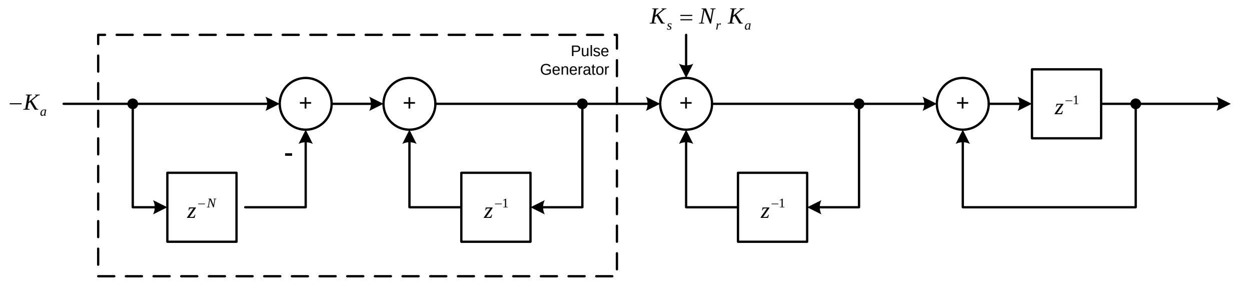
One final comment here. The normalized values are fractions but the actual computations employed use integer math. All the parameters in code are actually integers and there are no round-off issues present.

Summary

The system proposed emulates a chain driven reversing screw level-wind perfectly (so long as an LOS event does not occur) without the need for fabricating a custom reversing screw while repurposing a relatively inexpensive recirculating ball CNC stage. The same provisions used in normal operation making indexing and LOS detection and recovery very simple and straightforward.

Appendix

This appendix captures the mathematical analysis required to compute the distance traveled from the start of reversal to the zero velocity point. This will be terse as it is mainly intended to capture the math for future reference. It is based on a z transform analysis and employs its final value theorem. The model used is shown below.



Model Employed

The elements to the right represent the cascaded accumulators used for the system. The value K_a and $K_s = N_r K_a$ are all integers, i.e., not normalized. Mathematically the integrators start at 0 so this value is inpulsed into the Velocity integrator to represent the velocity integrator's initial condition at time 0. The dashed block on the left produces a pulse of amplitude $-K_a$ of duration N_r clocks in response to the impulse input of value K_a. The leading differentiator impulses the following velocity integrator with an impulse of amplitude $-K_a$ causing the integrator to step to that value. Then N_r clocks later, it produces an impulse with amplitude K_a to reset the velocity integrator to 0. With 0 into the following position integrator, it stops at a value equal to the distance traveled in microincrements.

This fixed length pulse serves to ramp down the velocity integrator from K_s to zero over a period of N_r clocks. The value that accumulates in the final system integrator is the number of microincrements. The z-transform of the response is found simply as the superposition of the responses to the individual inputs.

$$
\begin{aligned}
F(z) &= K_a \left[\frac{N_r z^{-1}}{\left(1-z^{-1}\right)^2} - \frac{z^{-1}\left(1-z^{-N_r}\right)}{\left(1-z^{-1}\right)^3} \right] \\
&= K_a \left[\frac{N_r z^{-1}\left(1-z^{-1}\right)}{\left(1-z^{-1}\right)^3} - \frac{z^{-1}\left(1-z^{-N_r}\right)}{\left(1-z^{-1}\right)^3} \right] \\
&= K_a z^{-1} \left[\frac{N_r\left(1-z^{-1}\right) - \left(1-z^{-N_r}\right)}{\left(1-z^{-1}\right)^3} \right]
\end{aligned}
$$

The term on the left in the bracket is the z-transform due to the Velocity integrator's initial value $N_r K_a$ input and the term on the right due to the $-K_a$ input.

Using the final value theorem, the resulting final value of the system is given by

$$
\begin{aligned}
\lim_{k\to\infty} f(k) &= \lim_{z\to 1}(z-1)F(z) \\
&= K_a \lim_{z\to 1}(z-1)z^{-1}\left[\frac{N_r\left(1-z^{-1}\right)-\left(1-z^{-N_r}\right)}{\left(1-z^{-1}\right)^3}\right] \\
&= K_a \lim_{z\to 1}\left[\frac{N_r\left(1-z^{-1}\right)-\left(1-z^{-N_r}\right)}{\left(1-z^{-1}\right)^2}\right] \\
&= K_a \lim_{z\to 1}\left[\frac{N_r\, z^{-2}-N_r\, z^{-N_r-1}}{2\left(1-z^{-1}\right)z^{-2}}\right] = \frac{N_r\, K_a}{2}\lim_{z\to 1}\left[\frac{1-z^{-(N_r-1)}}{2\left(1-z^{-1}\right)}\right] \\
&= \frac{N_r}{2}K_a \lim_{z\to 1}\left[\frac{\left(N_r-1\right)z^{-N_r}}{2\,z^{-2}}\right] \\
&= \frac{N_r\left(N_r-1\right)}{2}K_a
\end{aligned}
$$

Q.E.D.

Note that for any integer N_r either N_r or N_r-1 will be even and integrally divisible by 2 so the distance traveled will always be an integral number of microsteps as it should be evident it must be as the Position integrator can only represent integer microincements.