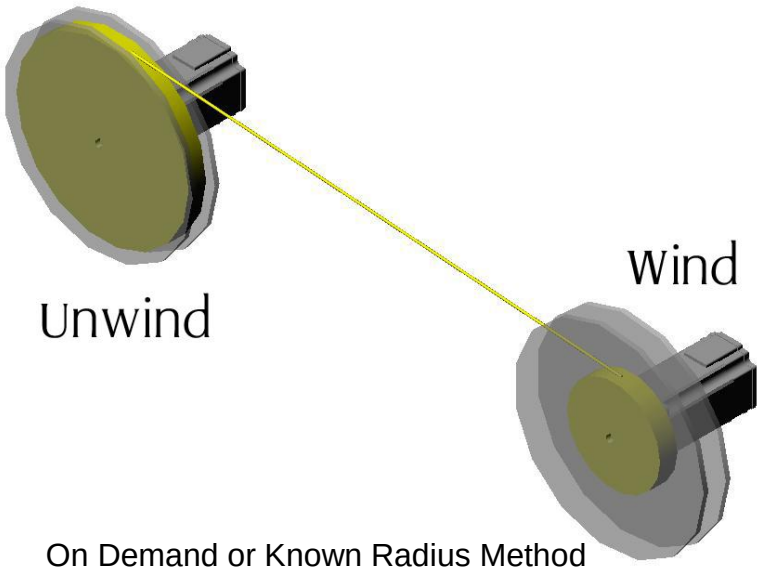
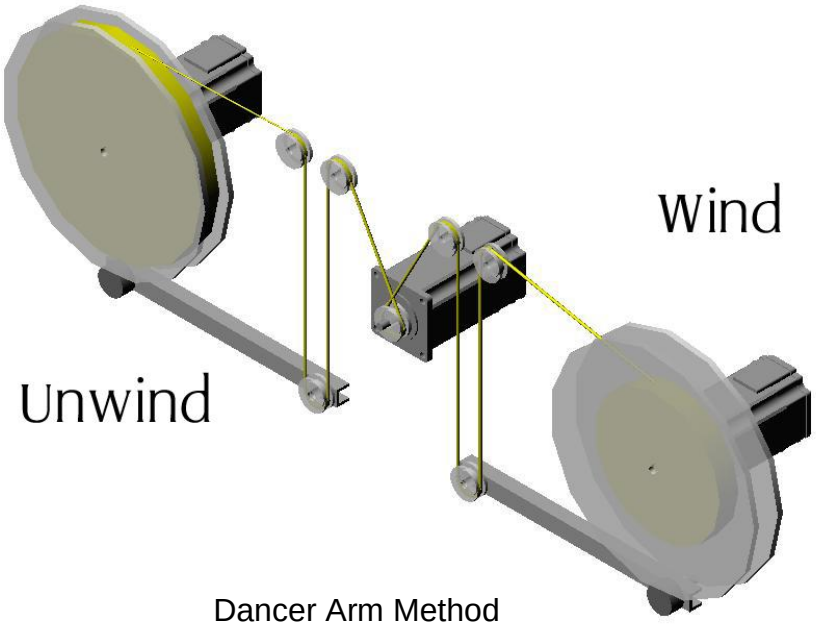
Winding / Unwind



On Demand or Known Radius Method



Dancer Arm Method

Associated QuickContro® programs included:

OnDemand_Unwind.qcp
OnDemand_Wind.qcp
KnownRadius_Unwind.qcp
KnownRadius_Wind.qcp
DancerArm.qcp

Featuring

- Traditional Dancer Arm Method
- 2 Sensorless Methods
- Bi-directional
- Direct Drive (using QCI's large inertial mismatch capability)

Prerequisites

The reader should be familiar with the following:

1. Application Note QCI-AN047 Input Mode – Joystick
2. Application Note QCI-AN023 Analog Input
3. Technical Document QCI-TD054 Servo Tuning
4. Application Note QCI-AN020 Variable Torque Clutch

Three Methods

This application note describes three methods to wind and/or unwind material. In most cases the servos directly drive a reel that continually winds or unwinds material while still providing tension.

The methods are discussed (from simple to advanced) are:

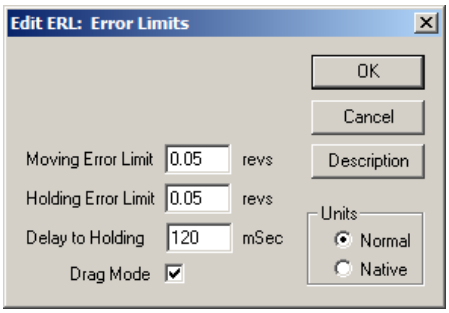
1) On Demand Method
2) Known Radius Method
3) Dancer Arm Method

Tuning

It is common for wind/unwind applications to have large inertial loads with inertial mismatches of 100:1 or greater. They will require some tuning to stabilize. For detailed instructions on tuning refer to Technical Document QCI-TD054 - Servo Tuning.

Drag Mode

Common to all the methods is the use of QuickSilver's unique slip clutch feature called Drag Mode. In Drag mode, the servo emulates a slip clutch. Drag Mode is enabled using the ERL command. ERL has already been included in all the examples. See Drag Mode in User Manual or Application Note QCI-AN020 Variable Torque Clutch for more details.

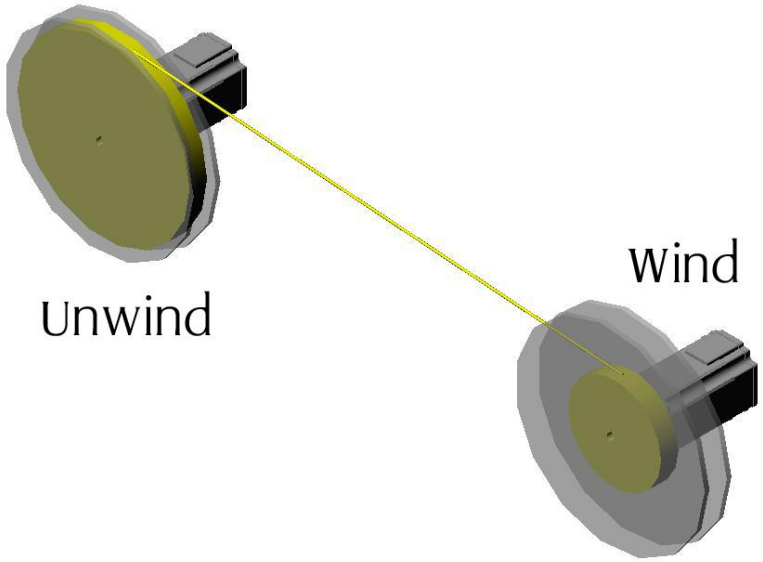


More Information

Other general information can be found in the technical documents listed above, SilverLode User Manual and the SilverLode Command Reference.

On Demand Method

The Wind servo is programmed to run at a fixed velocity. If it cannot maintain its velocity, it outputs a “Reduce Torque” signal to the Unwind servo.



Note, this method only works with sturdy material, because the Unwind servo could be set to its full reel torque with an almost empty reel for sometime while the algorithm adjusts.

OnDemand_Unwind.qcp

The maximum torque required of the Unwind servo is when it has a full reel (Tf). The minimum is for an empty reel (Te). For our example, this is 75% and 5% respectively. You will need to determine these empirically for your system.

Note, even if the motor is off, there is still some residual torque. If this is too much for your system, we suggest using a smaller motor. Alternatively you could use the Gravity Offset (GOC) command to give the servo a litter extra torque in one direction.

Torque (T) (reg 32) is first set to Tf. The servo’s torque is set by copying T into the upper word of reg 206 (Closed Loop Holding Torque).

The Reduce Torque input (I/O #110) is checked every 500ms. Once it goes HIGH, T is reduced by 1%. If Te is reached, the program ends, otherwise it loops back to start over.

Note, the 500ms delay determines how fast torque is reduced. Reduce the timer if you want the servo to respond faster to the “Reduce Torque” commands.

10:REM		Full Reel Torque (Tf)
11:WRP		Write 75 % to "Tf[30]" Register
12:REM		Empty Reel Torque (Te)
13:WRP		Write 5 % to "Te[31]" Register
14:REM		T=Tf
15:CLX		T[32] = Tf[30]
16:REM		Set servo's torque by setting holding torque register
17:CLC		Accumulator[10] = T[32]
18:CLC		High Word Closed Loop Torque Hold | Move[206] = LO(Accumulator[10])
19:REM	LOOP	Loop
20:REM		Wait for Reduce Torque signal from Wind axis Note: Decrease this delay if you want the torque to change faster
21:DLY		Delay for 500 mSec
22:JOI		Jump On Input -1 line(s) When "I/O #110" is LOW/FALSE
23:REM		Decrease current torque by 1%
24:CLD		T[32] = T[32] - 1 %
25:REM		Set servo's torque by setting holding torque register
26:CLC		Accumulator[10] = T[32]
27:CLC		High Word Closed Loop Torque Hold | Move[206] = LO(Accumulator[10])
28:REM		Stop adjusting torque when it reaches Te
29:CLX		Accumulator[10] = T[32] - Te[31]
30:JMP		Jump to "END" If Last Calc Was Negative TRUE
31:PCL		Program Call "SET TORQUE"
32:JMP		Jump to "LOOP"
33:REM	END	End Stop adjusting torque
34:END		End Program

OnDemand_Wind.qcp

The Wind servo's torque is set to the full reel torque (Tf) (see qcp).

The VMP command is used to set the velocity.

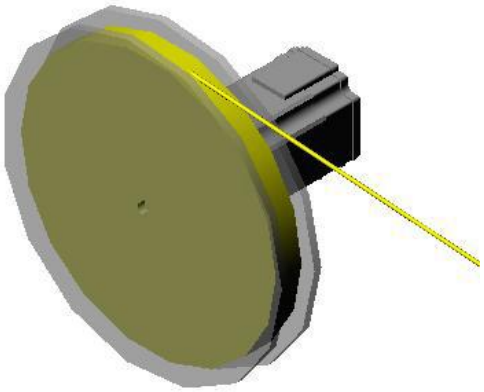
The Analog Continous Read (ACR) command continuously reads the analog velocity channel every 120 μs, passes it through a 3Hz filter and stores the result in register 30. 3Hz should be low enough to filter out any spurious velocity changes.

If the filtered velocity drops below 3.5rps, we set (HIGH) the Reduce Torque output to signal the Unwind servo to reduce its torque. If the velocity is ok, we clear (LOW) the output.

Line# Oper	Label	Command
10:REM		Set velocity
11:EMT		Enable Multi-Tasking
12:VMP		Velocity Mode: acc=15 rps/s, vel=4 rps
13:REM		Read and filter Velocity into reg 30
14:ACR		Analog Continuous Read: "Filtered Vel[30]" = Velocity 3 Hz Filter
15:REM	LOOP	If velocity falls too low, Clear #110 (Reduce Velocity) Else, maintain torque
16:JLE		Jump to "REDUCE TORQUE" When "Accumulator[10]" <= 3.9 rps
17:REM		Maintain Torque
18:COB		Clear "I/O #110"
19:JMP		Jump to "LOOP"
20:REM	REDUCE TORQUE	Reduce Torque
21:SOB		Set "I/O #110"
22:JMP		Jump to "LOOP"

Known Radius Method

The Unwind servo starts with a full reel of material with a known radius. As the material is unwound, the force on it (tension) increases. To comepensate for the increase in tension, the Unwind servo must reduce its torque in proportion to the reduction in material radius.



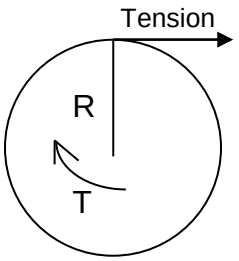
Unwind

To apply a constant Tension (F) to the material, the Torque (T) must be reduced proportionally to Radius(R).

$$T = R * F$$

We want F to be constant. We determine it emperially by determining T at a given R. For example, using an empty reel of radius R_e, use QuickControl to determine the required torque (T_e) to produce the desired tension (F).

$$F = T_e / R_e.$$



The real-time R, can be either read from an external sensor, or calculated. If the starting radius or starting amount of material is known, no sensors are necessary, otherwise a sensor is required to detect the amount of material (radius) on the reel.

Calculating Radius

R is a function of the servo's Position (P) given the following constants:

Constants:

R_e: radius of empty reel (mm)
R_f: radius of full reel (mm)
P_e: servo position counts from full to empty reel.
Note: P_e can be set to empty partial reels too.
K: servo position counts to unwind 1 mm of material

When unwinding, P will vary from zero to P_e counts.

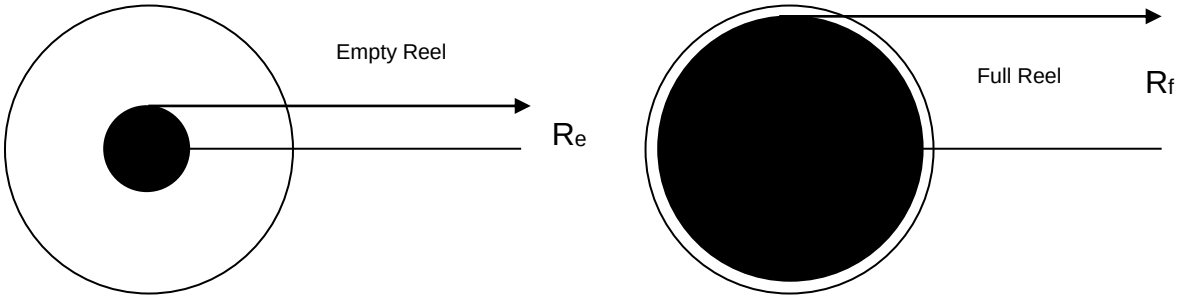
Knowing the constants P_e, K and R_e, the real-time radius (R), can be calculated knowing the actual position (P). The following equation is used to calculate R:

$$R = (P_e - P) / K + R_e$$

Therefore, the real-time torque (T), to maintain a constant material tension (F), is given by the following formula:

$$T = R * F = \{ (P_e - P) / K + R_e \} \{ T_e / R_e \}$$

KnownRadius_Unwind.qcp



Line# Oper	Label	Command
11:REM		Te=Torque to maintain desired Tension (F), for an empty reel of Radius (Re) (%)
12:WRP		Write 5 % to "Te[33]" Register
13:REM		Re=Radius of empty reel (mm)
14:WRP		Write 25 to "Re[30]" Register
15:REM		Pe=Encoder counts of rotation to empty reel.
16:WRP		Write 1600000 to "Pe[34]" Register
17:REM		K=Encoder counts of rotation per mm of radius (cnts/mm)
18:WRP		Write 60000 to "K[32]" Register
19:REM		F=Constant Tension F=Te/Re
20:CLX		F[27] = Te[33]/LO(Re[30])
21:REM	LOOP	Loop
22:REM		Pe-P P=ABSposition) ABS is used to allow for CW/CCW operation
23:CLX		P[31] = ABS(Actual Position[1])
24:CLX		Accumulator[10] = Pe[34] - P[31]
25:REM		(Pe-P)/K
26:CLC		Accumulator[10] = Accumulator[10]/K[32]
27:REM		R=(Pe-P)/K+Re
28:CLX		R[28] = Accumulator[10] + Re[30]
29:REM		T=F * R
30:CLX		T[29] = F[27] * R[28]
31:REM		Set holding torque register
32:CLC		Accumulator[10] = T[29]
33:CLC		High Word Closed Loop Torque Hold | Move[206] = LO(Accumulator[10])
34:REM		Stop when all material unwound
35:CLX		Accumulator[10] = Pe[34] - P[31]
36:JMP		Jump to "END" If Last Calc Was Negative TRUE
37:REM		Loop back
38:JMP		Jump to "LOOP"
39:REM	END	End
40:END		End Program

R_e: 25 mm
R_f: 50 mm
P_e: 1,600,000 counts
K: 60,000 counts / mm
T_e: 5% (1000 STU[1])

$F = T_e / R_e = 1000 / 25 = 40$

The main Loop first copies the absolute value of Actual Position (reg 1) into P (reg 31). ABS is used to allow for CW/CCW operation.

The rest of the loop calculates:

$T = R * F = \{ (P_e - P) / K + R_e \} \{ T_e / R_e \}$

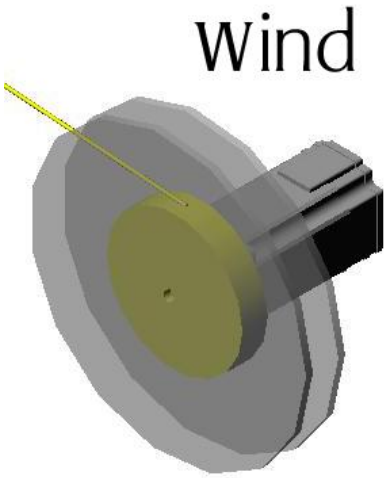
The servo's torque is set by copying T into the upper word of reg 206 (Closed Loop Holding Torque).

If $T<T_e$, the program ends.

[1] SilverLode Torque Unit (STU) – See User Manual for details.

KnownRadius_Wind.qcp

Winding material is the same as unwinding except the servo is commanded to move at some velocity.



Decoupled systems contain axes that are isolated and comprise of physical mechanisms such as gears, pulleys, springs, idle rollers, etc. To keep a constant tension on the material, both wind and unwind servos must adjust their torque accordingly.

The Wind servo uses the same algorithm as the Unwind servo. The main difference is the Wind servo torque must increase as the radius increases.

As previously mentioned, to apply a constant Tension (F) to the material, the Torque (T) must be kept proportional to Radius(R).

$$T = R * F$$

Where,

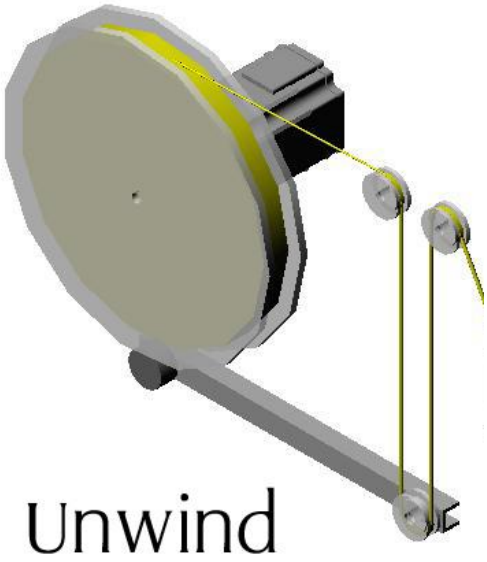
$$F = T_e / R_e.$$

$$R = P/K + R_o$$

See previous section and KnownRadius_Wind.qcp for more details.

Dancer Arm Method

The figure to the right shows a single reel with a dancer arm. The dancer arm is attached to a rotational sensor (potentiometer or hall effect) that outputs an analog signal. The dancer arm is spring loaded. Therefore, the rotational sensor's analog voltage is a measurement of the material's tension.

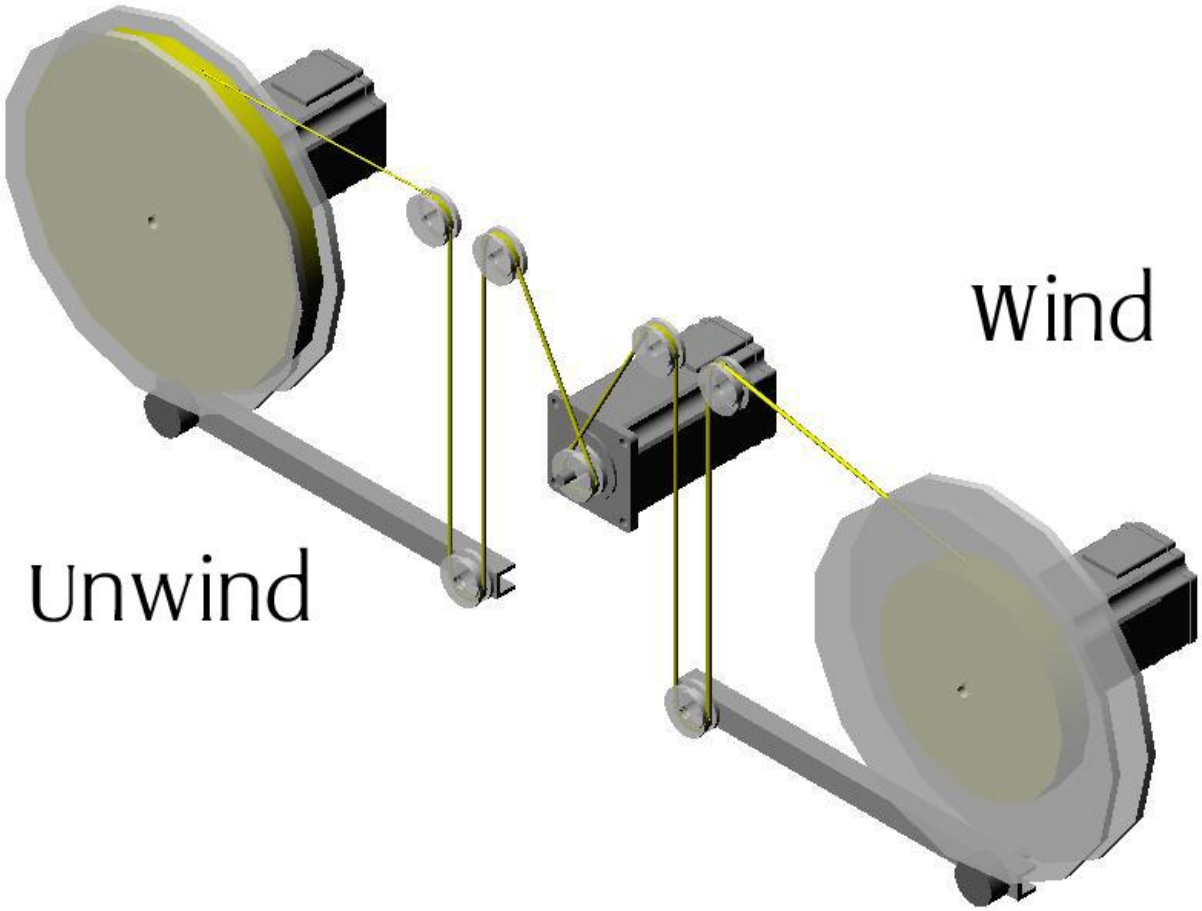


The servo uses the analog input to set its velocity. With the dancer arm at the midpoint position, the servo is configured for zero velocity.

When the dancer arm falls below the midpoint, the servo rotates counter-clockwise to increase tension. Conversely, when the dancer arm rises above the midpoint, the servo rotates clockwise to decrease tension. The velocity is determined by how far the dancer arm is from the midpoint. The logic is similar to a joystick controlling the servo's velocity. QuickSilver has a special command for this called Velocity Input Mode (VIM). See Application Note QCI-AN047 Input Mode – Joystick for details.

This method allows the servo to either wind or unwind.

Two Reels

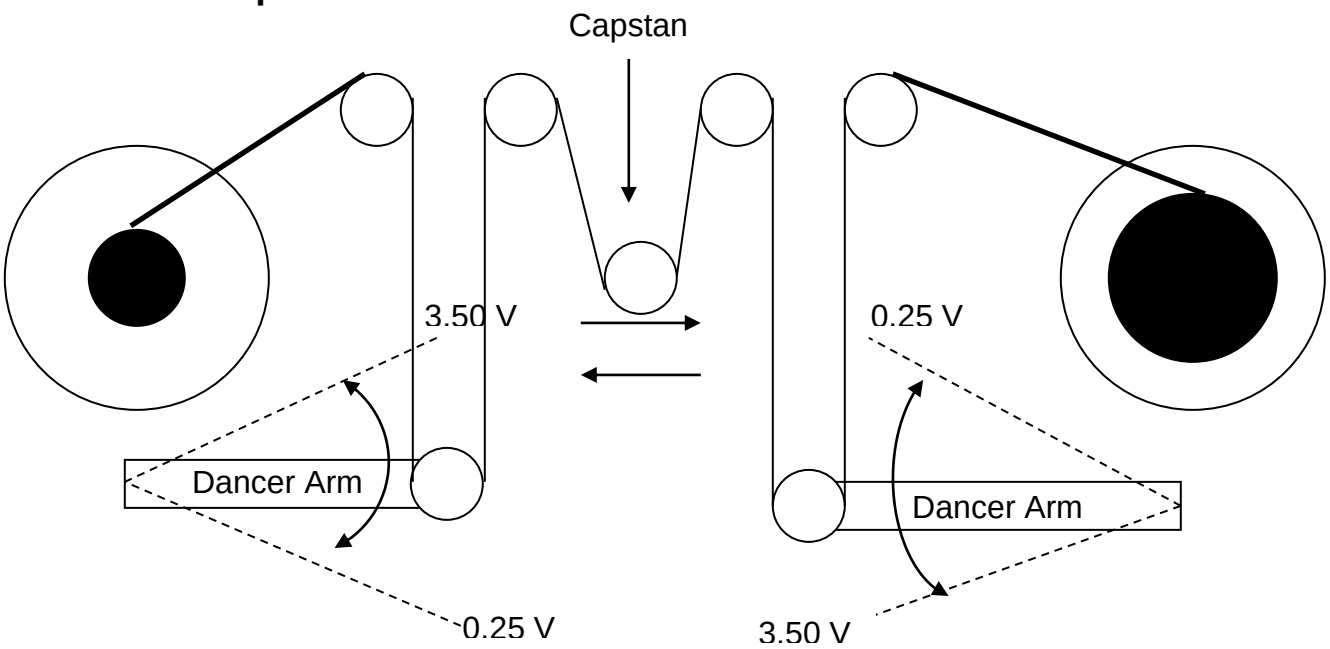


The dancer arm method is great for two decoupled reels. In this example, the user controls the speed of the middle servo, called the capstan. The other two servos automatically either wind or unwind material always keeping tension on their part of the line constant.

When the capstan moves material to the right, tension on the left side of the line increases, raising the dancer arm. The “Unwind” servo compensates by moving clockwise. On the right side, tension decreases, which lowers the dancer arm. The “Wind” servo compensates by moving clockwise to increase the tension.

The dancer arms can move much faster than the reels, for that reason, the dancer arms also act as a buffer or accumulator to give the reels enough time to match the capstan’s speed.

Hardware Setup



Application Example

This example implements a three-servo bi-directional wind/unwind reel application. That is, material can move either left or right.

The center servo (capstan) is user controlled and can run at different speeds both clockwise and counter clockwise.

In this example, ¼ turn hall effect rotary position sensors are used. Hall effect sensors are recommended because of their durability. The sensors are attached to the dancing arms. The sensors output 0-5 volts. The SilverDust and SilverNugget accepts 0-3.3 volts and 0-5 volts, respectively.

As illustrated in the above figure, the dancer arms have their dancer arm signals inverted. This enables the use of the same program for both Wind and Unwind servos.

The amount of material in the dancer arm loop determines how much the dancers can buffer speed changes from the capstan.

DancerArm.qcp

QuickSilver has a special command (VIM) to set servo velocity using an analog input. The following explains how to use VIM with a dancer arm. For more details on VIM, see Application Note QCI-AN047 Input Mode – Joystick.

The ACR command is used to read Analog Channel #1 (dancer arm) every 120 μs (in the background), convert it to ADC counts and store the result in register 12. Note, QuickControl's Register Watch tool can be used to view the contents of register 12.

Input Offset (reg 13) is the value of register 12 when the dancer arm is at its midpoint (i.e. 17000).

The dancer arm's full range determines the value of Maximum Scale/Limit (reg 15). To set reg 15, determine the value of register 12 when the dancer arm is at its full up and down positions. Subtract Input Offset (reg 13) from both values and take absolute value of results. The Maximum Scale/Limit (reg 15) equals the smaller of the two numbers.

reg 12 full up/down = 1000/32000
full up:ABS(1000-17000)=16000
full down: ABS(32000-17000)=15000
Maximum Scale | Limit [15] = 15000

Set Maximum Output Scale (reg 16) and Output Rate of Change (reg 18) to the maximum desired speed and acceleration respectively.

The following equation can be used to convert line speed to rotational speed.

$$V_m = \frac{L_m}{6.28 * R_f}$$

R_f Radius of Full Reel (in)
L_m Maximum Line Speed (in/sec)
V_m Max Motor Speed (rev/sec)

Line# Oper	Label	Command
11:REM		Read signal from dancer arm to register 12
12:ACR		Analog Continuous Read: "Dancer Arm Pos[12]" = Analog Channel #1
13:REM		Middle of dancer arm stroke
14:WRP		Write 17000 to "Input Offset[13]" Register
15:WRP		Write 0 to "Input Dead Band[14]" Register
16:REM		See application note
17:WRP		Write 15000 to "Maximum Scale|Limit[15]" Register
18:REM		Max Speed
19:WRP		Write 10 rps to "Maximum Output Scale[16]" Register
20:WRP		Write 0 rps to "Output Offset[17]" Register
21:REM		Max Acceleration
22:WRP		Write 40 rps/s to "Output Rate of Change[18]" Register
23:REM		Start Velocity Input Mode
24:VIM		Velocity Input Mode:
25:REM	LOOP	Stop motor if dancer arm goes out of range. This occurs when the material falls off.
26:JLT		Jump to "STOP" When "Dancer Arm Pos[12]" < 3500
27:JGR		Jump to "STOP" When "Dancer Arm Pos[12]" > 32000
28:JMP		Jump to "LOOP"
29:REM	STOP	Stop
30:VMP		Velocity Mode: acc=50 rps/s, vel=0 rps

Lower the VIM Filter parameter (set to 10Hz in our example) to slow down the response of the reel to the capstan. Increase it to speed up the response.

The servo is stopped if the dancer arm goes out of range (i.e. material falls off).