

RED2 Documentation

Gregg Stiesberg

September 24, 2026

Quick Start

Hardware

Software - ECat Master

The RED2 ECat Master fieldbus driver is based on the open-source SOEM C library V1.3.3, with GPL2 license. The primary purpose of the ECat driver is to manage transitions between operational states and implement high-level control as needed. The main operational state sets the drives to velocity and current(force) control modes for the 'big' and 'small' motors, respectively, using the drives to implement the high-rate low-level feedback control of the motors.

Lift Mode

The method that best fits the primary low-level control algorithm (damper emulation via zero angular velocity control on big motor, constant force spring emulation on small motor) is to change from zero velocity control to finite upward velocity control to achieve a steady-state lifting force. The force F_B applied by the big motor is proportional to the shaft angular velocity ω_B :

$$F_B = R_d^{-1} * \beta_B * \omega_B$$

where R_d is the drum radius and β_B is the equivalent damping coefficient which is the primary control parameter set for the zero velocity 'Fall Catch' mode.

For the 'Lift Mode' we most likely want to specify a maximum desired lift force and command a non-zero angular velocity consistent with that force:

$$\omega_B = F_B * R_d * \beta_B^{-1}$$

and thus implement force application seamlessly using the same velocity control mode already implemented. (To check: do we need also to dynamically change the damping coefficient β_B or can it remain fixed at the 'Fall Catch' mode value?)

Some priorities for the implementation are:

- Smooth transition to desired steady-state force is important to limit the disturbance to the patient and avoid mechanical resonances
- Computational efficiency is important to maintain consistent high-rate ECat bus performance in an embedded compute environment
- Minimizing the number of parameters eases the burden on users and high level interfaces
- Instantaneous adaptation to change in target angular velocity (force) enables anticipated higher-level control algorithm (such as height dependent force)

A compromise is needed to achieve the smoothest possible change in applied force while responding to commands as quickly as possible. As a starting point, we develop a simple algorithm equivalent to a discrete approximation to an exponential decay to the target angular velocity, with the global rate of change in velocity (and hence jerk) limited by a constant:

$$\omega_{n+1} = \omega_n + \Delta\omega$$

with details implementing approach to target angular velocity from below or from above,

$$\Delta\omega = \text{sign}(\omega_{err}) * \alpha * \min(\text{abs}(\omega_{err}), \Delta\omega_{max})$$

where the signed error is simply

$$\omega_{err} = (\omega_n - \omega_{target})$$

This algorithm responds immediately but avoids discontinuities in the force by limiting the initial slope, and has the 'infinitely smooth' property of exponential decay to the commanded steady-state force.

Note that the force prescribed by the algorithm only applies in the static case; if there is upward motion then the force is reduced by the reduction of the relative velocity error. (is this a feature or a drawback? only assist when needed, or control the target velocity in real-time to achieve prescribed force?)

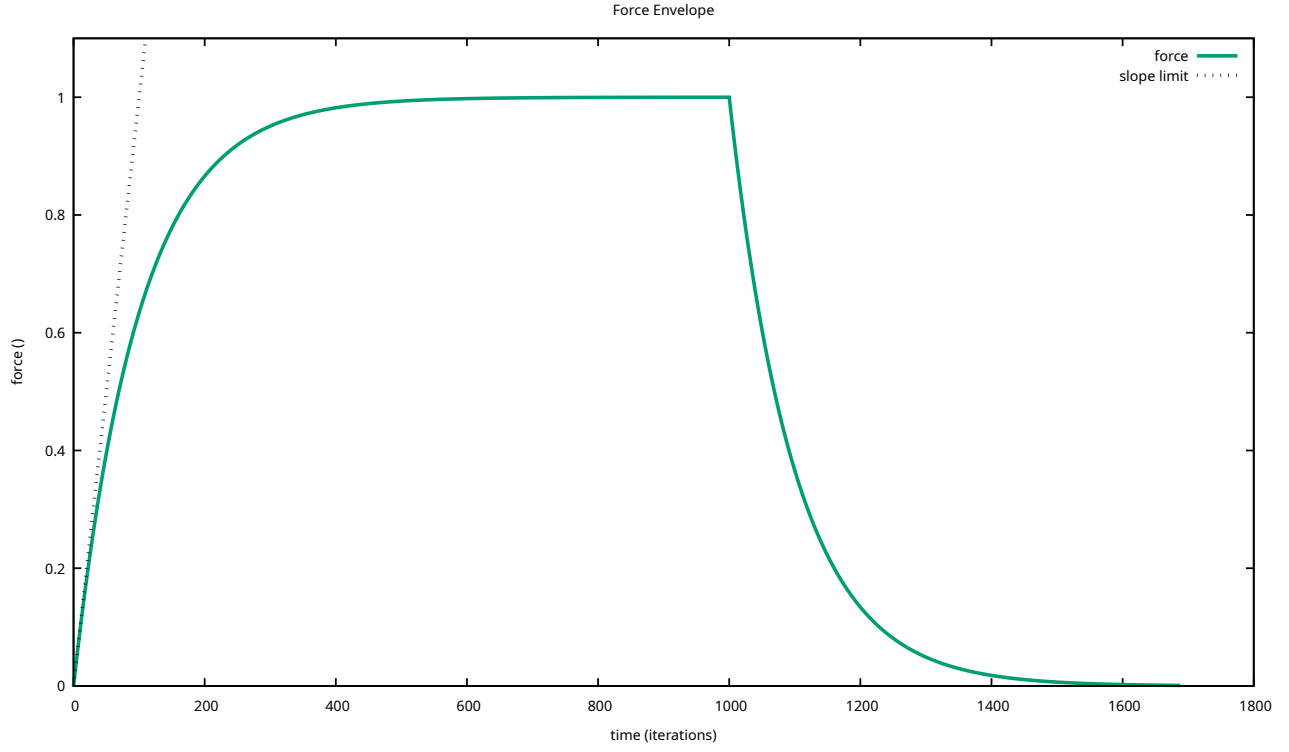


Figure 1: Example force application envelope with peak steady-state force $F = 1.0$ in arbitrary units. The target force is sufficiently small such that the initial slope of the increasing exponential is not limited by the global jerk limiting constant (shown by the dotted line).

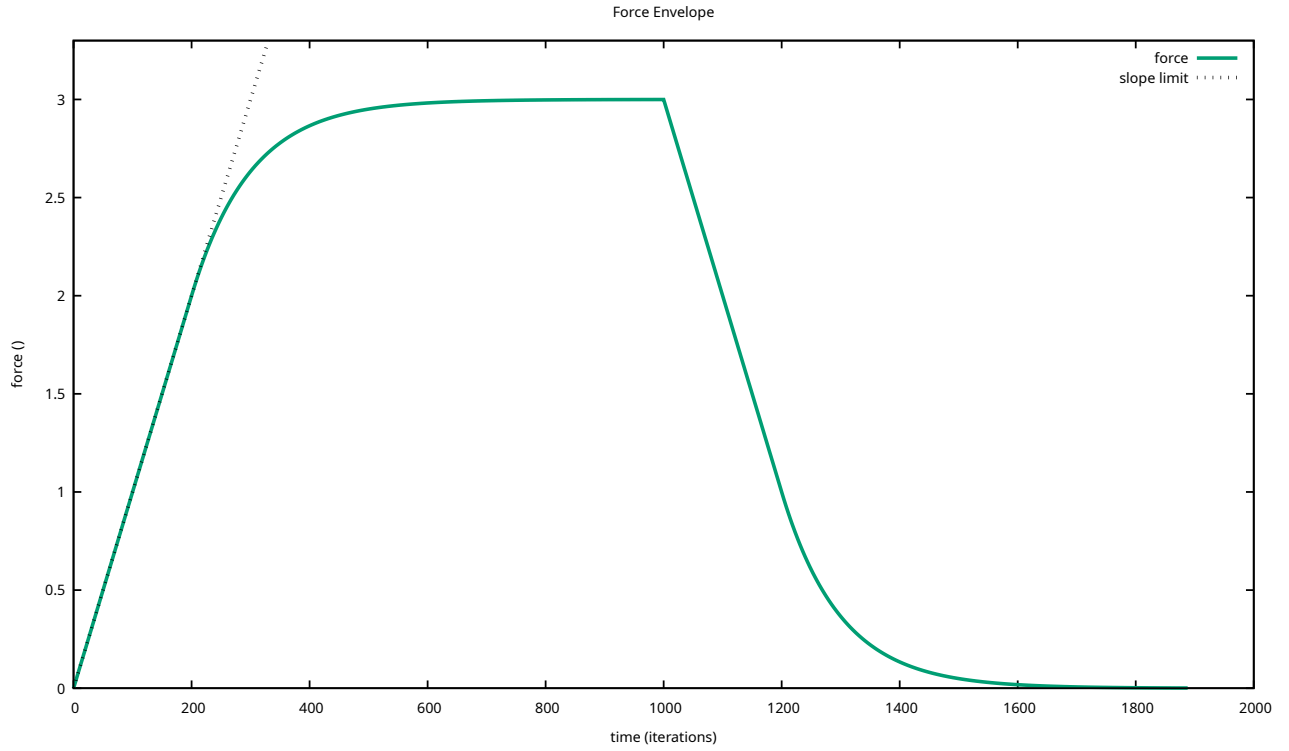


Figure 2: Example force application envelope with peak steady-state force $F = 3.0$ in arbitrary units. The target force is sufficiently large such that the initial slope of the increasing exponential is clearly limited by the global slope-limiting (jerk limiting) constant (shown by the dotted line). The slope limit also applies to the decreasing exponential.

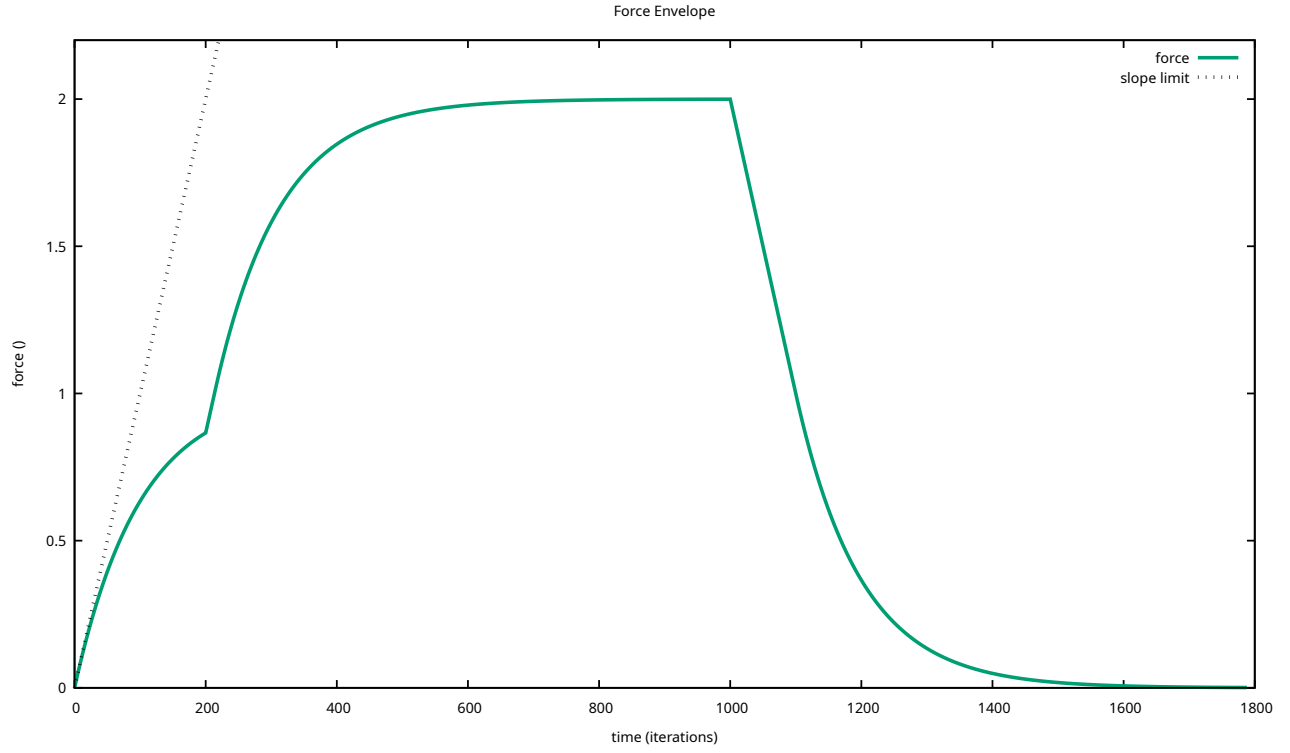


Figure 3: Example force application envelope with target steady-state force initially set to $F = 1.0$, then at iteration $n = 200$ the target is instantaneously changed to $F = 2.0$. The applied force remains continuous even when the target is discontinuously changed on-the-fly.

Regarding high-level control of the applied force, some ideas:

- **Force Specified for Static Case:** The applied force envelope is specified for the case of a non-moving patient; as the patient moves upward, the applied force is reduced (the applied force is proportional to the relative velocity error).
- **Force Specified for General Case:** The applied force envelope is maintained regardless of upward patient motion; in the context of the proposed target velocity algorithm, this requires feedback control on the target velocity to maintain desired force envelope.
- **Force Dependence on Position:** It may be desirable to reduce the force as the patient rises and stabilizes, needing less assistance.

Limitations on applied force magnitude and duration:

- **Power Available:** The continuous work done by the lifting force must not exceed power supply capabilities (6-10 kW, untested for long durations). Changes in power should be sufficiently smooth, slew rate limited by impulse response of power supply.
- **Motor Current:** The continuous motor current shall not exceed motor limits (approx. 15A via torque max / torque const)
- **Motor Temperature:** The motor temperature must be monitored and shall not exceed reasonable limits (Datasheet: up to 155degC !!! PTC sensor linearity up to 140degC.)

Maximum dead-lift thus far performed was approximately 1100kg (11000N); peak capability may be 1500kg (15000N).

Software - Bash User Interface

A simple lightweight user interface is implemented in Bash to enable user control of the system over an SSH session in a terminal.

Abbreviations and Definitions

Supplemental Material

HVDC PFC Supply Specifications

Vac_rms	:	240
Fac	:	60
Vout	:	600
Vovp	:	636.0
Vdrop	:	20
Ptot	:	1600
Fpwm	:	200040
Fsync	:	400080
Dmax	:	0.98
Vreg_vsense	:	3.0
Vreg_vinac	:	3.0
DVvao	:	3.6
Vreg_currsense	:	3.2
Vreg_currpeak	:	4.0
DVramp	:	4.0
Vref	:	6.0
Gmv	:	7e-05
Gmc	:	0.0001
Kmult	:	1.7e-05
Lx	:	0.00054
RAvs	:	3e+06
RBvs	:	15075.4
Cvs	:	1e-09
RAac	:	3e+06
RBac	:	15075.4
Cac	:	1e-09
Rrdm	:	0
Ccdr	:	0
Rimo	:	49420.4
Rsequiv	:	0.678823
Enable_Synthesis	:	0
Rsyn	:	0.0
RApk	:	6800.0
RBpk	:	13600.0
PWsyncExt	:	2.5e-07
Rrt	:	37492.5
Rdmx	:	32242.8
Cout	:	0.00112994
fracDVvao	:	0.02
Cpv	:	4.03604e-08
Fpv	:	12.0
Fzv	:	2.0
Rzv	:	3.28612e+05
Czv	:	2.42162e-07
Fzc	:	16670.0
Gpsc	:	1.80027
Rzc	:	5554.71
Czc	:	1.71879e-09
Fpc	:	66680.0
Cpc	:	4.29697e-10
Cvref	:	1e-06
Cvcc	:	1e-05
Css	:	2.22222e-06
Clock	:	0

```

Rload                :                225.0
funcPtr_Rload        : None
EnablePh1            :                1
SkipHalfMod          :                1
SkipHalf             :                0
gppPlotter           : <__main__.gnuPlotter object at 0x6ffcdecae780>
PlotPDF              :                0
PlotPrefix           : pdfplots/
SweepRload0          : <__main__.UCC28070.sweepRload object at 0x6ffce14563c0>
VLO                  : <__main__.UCC28070.VoltageLoop object at 0x6ffcde5ffa40>
CLO                  : <__main__.UCC28070.CurrentLoop object at 0x6ffcde509100>
PMO                  : <__main__.UCC28070.PowerMeter object at 0x6ffcdec61d60>
ICout at Pmax: 5.333333333333333 Vripple: 6.260094428281217
Vvinac max: 1.6970562748477143 kvff: 1.604
get_Vinac_Correction_Factor(): [1.6970562748477143, array(1.604), 1.4400000000000004, 0.8977556109725687, 1.79
IMO_max: 6.475052619743148e-05 Vimo_max: 3.2
Duty at top: 0.434314575050762 at peak voltage: 339.4112549695428
ILxTop: 4.714045207910317 dILxTop: 1.3646460982761115 ILxPeak: 5.396368257048373
dILxTop/ILxTop: 0.2894851530032406
Rsequiv: 0.6788225099390857 --> need Vreg_currpeak >= 3.663176244805185
LxMin_dILx: 0.000848358465730711 for fracilx=0.3
LxMin_CCM at Pmax/4: 0.0007198560287942411
LxMin_CCM at Pmax/3: 0.0005398920215956808
PoutMin_CCM: 533.2266879957342 with Lx: 0.00054

```