COMPUTE OPTIMAL ROTOR SPEED

Filter for Wind Speed

VwOut | VwIn
fcn

lambda_opt / R_rotor

Vw_timeseries

wind_speed

wind_speed

omega star

Constant input 1&2

V_wind_const

omega *

wind_speed

PITCH ANGLE

PID Controller | Rate | Slide saturated

IF V_wind > V_rated activate pitch control
IF V_wind < V_rated no pitch control

pitch angle

wind_speed

V_rated - V_rated

delta omega

PITCH CONTROL

pitch angle

Pitch Regulated / Stall Regulated

0

delta omega

[omega]

Wind-Blades Power Transfer

wind_speed | lambda
omega | Cp
beta | Tw
WindTorque | Pw

lambda

Cp

Tw

Pw

[omega]

omega
Tw
RororDynamics | domega_dt
Tm

$\frac{1}{s}$

omega

[omega]

ROTOR DYNAMICS

Tm

Omega_c

omega - omega ref.

PID(s)

PID Stall1

DESIRED REFERENCE

[omega]

ACTUAL OMEGA

omega_0

INITIAL CONDITION

actual omega and its input

[omega]

x

out.PowerGenerated

PID(s)

```
function VwOut = fcn(VwIn,V_ci, V_co, V_rated)
% if VwIn < V_ci
%      VwOut = 0;
% elseif VwIn >= V_ci && VwIn < V_co
%      VwOut = VwIn;
% else
%      VwOut = 0;
% end

if VwIn >= V_ci && VwIn < V_co
    VwOut = VwIn;
else
    VwOut = 0;
end
```
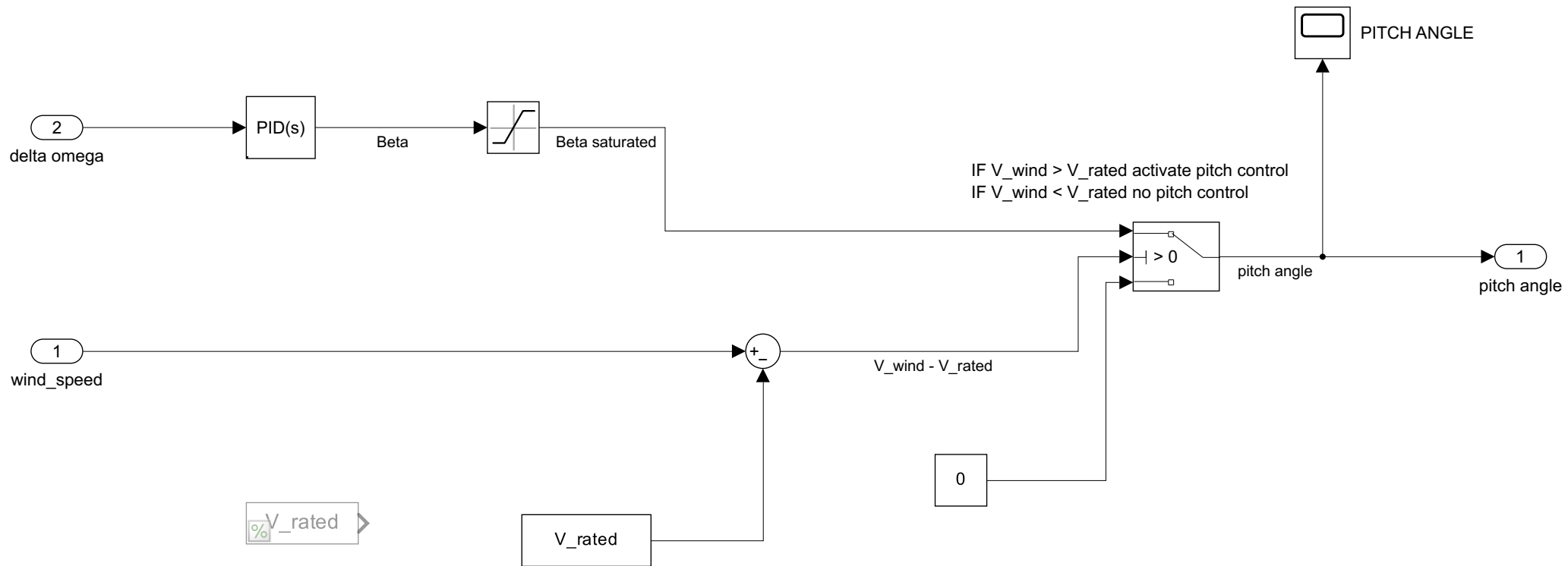
2 delta omega

PID(s) → Beta → [saturation] Beta saturated

PITCH ANGLE

IF V_wind > V_rated activate pitch control
IF V_wind < V_rated no pitch control

> 0

pitch angle

1 pitch angle

1 wind_speed

V_wind - V_rated

0

%V_rated

V_rated

```matlab
function domega_dt = RororDynamics(omega, Tw, Tm, B_r, Jeq)

% Torque balance:

% Jeq * domega_dt + B_r * omega = Tw - Tm rotor side reference frame
domega_dt = (Tw - Tm - B_r*omega) / Jeq;
```

```
function [lambda, Cp, Tw, Pw]  = WindTorque(wind_speed, omega, beta, R_rotor, rho_air, A_rotor)

fake_zero = 1e-6;

lambda = omega * R_rotor / wind_speed;
Cp = PowerFactor(lambda, beta);
if Cp < fake_zero
    Cp = fake_zero;
end

% Actual extracted power from wind
Pw = 0.5*rho_air*wind_speed^3*A_rotor*Cp;

% Actual torque acting on the rotor: P = T * w
Tw = Pw / omega;
```