FMU Documentation: Required_Speed_calc

(verified 20 Jun 2025 against `req_speed_calc.fmu` *and* `req_speed_calc.slx` – revised input list)*

1 Overview

Item	Value
Model name	`req_speed_calc`
FMI version	2.0
Generated by	Simulink R2020a
Generation date (FMU)	12 Feb 2023 11:08:44 UTC
GUID	`{b9e7d4c7-1e8e-4f58-b2da-2a0d4921f96c}`
Event indicators	0 (algebraic)

Default experiment

Start time	Stop time	Recommended Δt
0 s	10 s	0.2 s

---

2 Interface definition

Cross-checking the FMU's `modelDescription.xml` with the Simulink block diagram confirms **five** scalar *Real* inputs and **one** scalar *Real* output.

2.1 Inputs

#	FMU variable	Unit	Description
1	`lat1`	°	Latitude of waypoint 1 (departure)
2	`lat2`	°	Latitude of waypoint 2 (arrival)
3	`lon1`	°	Longitude of waypoint 1 (departure)
4	`lon2`	°	Longitude of waypoint 2 (arrival)
5	`ETA`	h	Desired duration of the trip (hours from *t* = *0*)

2.2 Output

FMU variable	Unit	Description
`req_speed_kn`	kn	Vessel speed required to meet ETA along great-circle route

2.3 Independent variable

Name	Unit	Description
`time`	s	Simulation time supplied by the master algorithm

---

3 Usage notes

1. **Communication step** — The FMU suggests **0.2 s**; smaller steps ($\leq$ 0.01 s) have no effect on this algebraic calculation.
2. **Provide all inputs** — All five inputs must be set before the first `fmi2DoStep`; missing values yield `NaN` for `req_speed_kn`.
3. **Coordinate format** — Latitudes and longitudes are **decimal degrees**, positive north/east, negative south/west.
4. **ETA** — Specify the desired arrival time in **hours** counted from simulation start (`time = 0`).
5. **Thread safety** — The FMU binary is **single-threaded**; launch multiple instances for parallel simulations.
6. **Binaries** — Archive ships **Win64** only. Re-export for other platforms if needed.

---

4 Quick-start example (Python + FMU)

```
from fmpy import simulate_fmu
import numpy as np

# Example: From (34.0 N, 24.5 E) to (36.0 N, 23.0 E) with ETA = 5 h
T = [0.0]  # single algebraic call is enough

inputs = {
    'lat1': np.array([[0.0, 34.0]]),   # °
    'lat2': np.array([[0.0, 36.0]]),   # °
    'lon1': np.array([[0.0, 24.5]]),   # °
    'lon2': np.array([[0.0, 23.0]]),   # °
    'ETA':  np.array([[0.0, 5.0]])     # h
}

result = simulate_fmu('req_speed_calc.fmu',
                      start_time=0.0,
```

```
                        stop_time=0.0,
                        step_size=0.0,
                        input=inputs)

print("Required speed (knots):", result['req_speed_kn'][-1])
```

Re-generated 20 Jun 2025 — inputs updated to `**, **`, ``.