

This PDF is generated from: <https://www.kalelabellium.eu/Wed-23-Nov-2022-24746.html>

Title: The future prospects of energy storage power generation

Generated on: 2026-03-05 03:50:15

Copyright (C) 2026 KALELA SOLAR. All rights reserved.

For the latest updates and more information, visit our website: <https://www.kalelabellium.eu>

A future represents the result of an asynchronous operation, and can have two states: uncompleted or completed. Most likely, as you aren't doing this just for fun, you actually ...

`future (const future &) = delete; ~future (); future & operator =(const future &) = delete; future & operator =(future & &) noexcept; shared_future <R>; share () noexcept; // ...`

The class template `std::future` provides a mechanism to access the result of asynchronous operations: An asynchronous operation (created via `std::async`, ...

Checks if the future refers to a shared state. This is the case only for futures that were not default-constructed or moved from (i.e. returned by `std::promise::get_future ()`, ...

The `get` member function waits (by calling `wait ()`) until the shared state is ready, then retrieves the value stored in the shared state (if any). Right after calling this function, `valid` ...

If the future is the result of a call to `async` that used lazy evaluation, this function returns immediately without waiting. The behavior is undefined if `valid ()` is false before the call ...

The promise is the "push" end of the promise-future communication channel: the operation that stores a value in the shared state synchronizes-with (as defined in ...

Specifies state of a future as returned by `wait_for` and `wait_until` functions of `std::future` and `std::shared_future`. Constants

If the future is the result of a call to `std::async` that used lazy evaluation, this function returns immediately

The future prospects of energy storage power generation

Source: <https://www.kalelabellium.eu/Wed-23-Nov-2022-24746.html>

Website: <https://www.kalelabellium.eu>

without waiting. This function may block for longer than ...

Web: <https://www.kalelabellium.eu>

