

EZ\_PARALLEL

ALPHA 1.01

Generated by Doxygen 1.8.17



<b>1 Modules Index</b>	<b>1</b>
1.1 Modules List	1
<b>2 Data Type Index</b>	<b>3</b>
2.1 Data Types List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Module Documentation</b>	<b>7</b>
4.1 ez_parallel Module Reference	7
4.1.1 Detailed Description	8
4.2 ez_parallel_structs Module Reference	8
4.2.1 Detailed Description	8
4.2.2 Variable Documentation	8
4.2.2.1 fft_1d_1	9
4.2.2.2 fft_1d_2	9
4.2.2.3 fft_2d	9
4.2.2.4 spec_drv_1d_1	9
4.2.2.5 spec_drv_1d_2	9
<b>5 Data Type Documentation</b>	<b>11</b>
5.1 The Module Reference	11
5.1.1 Detailed Description	11
5.2 ez_parallel::create_scheme Interface Reference	11
5.2.1 Detailed Description	11
5.2.2 Member Function/Subroutine Documentation	12
5.2.2.1 create_scheme_sbr()	12
5.3 ez_parallel::create_scheme_fft Interface Reference	12
5.3.1 Detailed Description	12
5.3.2 Member Function/Subroutine Documentation	12
5.3.2.1 create_scheme_fft_sbr()	13
5.4 ez_parallel::create_scheme_spec_drv Interface Reference	13
5.4.1 Detailed Description	13
5.4.2 Member Function/Subroutine Documentation	13
5.4.2.1 create_scheme_spec_drv_sbr()	13
5.5 ez_parallel::create_scheme_zero_pad Interface Reference	14
5.5.1 Detailed Description	14
5.5.2 Member Function/Subroutine Documentation	14
5.5.2.1 create_scheme_zero_pad_dble_sbr()	14
5.6 ez_parallel::destroy_scheme Interface Reference	14
5.6.1 Detailed Description	15
5.6.2 Member Function/Subroutine Documentation	15
5.6.2.1 destroy_scheme_sbr()	15

5.7 ez_parallel::execute_scheme_fft Interface Reference . . . . .	15
5.7.1 Detailed Description . . . . .	16
5.7.2 Member Function/Subroutine Documentation . . . . .	16
5.7.2.1 execute_scheme_fft_dble_sbr() . . . . .	16
5.7.2.2 execute_scheme_fft_dcmpx_sbr() . . . . .	16
5.8 ez_parallel::execute_scheme_ifft Interface Reference . . . . .	17
5.8.1 Detailed Description . . . . .	17
5.8.2 Member Function/Subroutine Documentation . . . . .	17
5.8.2.1 execute_scheme_ifft_dble_sbr() . . . . .	17
5.8.2.2 execute_scheme_ifft_dcmpx_sbr() . . . . .	17
5.9 ez_parallel::execute_scheme_ispec_drv Interface Reference . . . . .	19
5.9.1 Detailed Description . . . . .	19
5.9.2 Member Function/Subroutine Documentation . . . . .	19
5.9.2.1 execute_scheme_ispec_drv_dble_sbr() . . . . .	19
5.9.2.2 execute_scheme_ispec_drv_dcmpx_sbr() . . . . .	20
5.10 ez_parallel::execute_scheme_izero_pad Interface Reference . . . . .	20
5.10.1 Detailed Description . . . . .	21
5.10.2 Member Function/Subroutine Documentation . . . . .	21
5.10.2.1 execute_scheme_izero_pad_dble_sbr() . . . . .	21
5.10.2.2 execute_scheme_izero_pad_dcmpx_sbr() . . . . .	21
5.11 ez_parallel::execute_scheme_spec_drv Interface Reference . . . . .	22
5.11.1 Detailed Description . . . . .	22
5.11.2 Member Function/Subroutine Documentation . . . . .	22
5.11.2.1 execute_scheme_spec_drv_dble_sbr() . . . . .	22
5.11.2.2 execute_scheme_spec_drv_dcmpx_sbr() . . . . .	23
5.12 ez_parallel::execute_scheme_zero_pad Interface Reference . . . . .	23
5.12.1 Detailed Description . . . . .	23
5.12.2 Member Function/Subroutine Documentation . . . . .	24
5.12.2.1 execute_scheme_zero_pad_dble_sbr() . . . . .	24
5.12.2.2 execute_scheme_zero_pad_dcmpx_sbr() . . . . .	24
5.13 ez_parallel::max_val Interface Reference . . . . .	24
5.13.1 Detailed Description . . . . .	25
5.13.2 Member Function/Subroutine Documentation . . . . .	25
5.13.2.1 max_val_sbr() . . . . .	25
5.14 ez_parallel::min_val Interface Reference . . . . .	25
5.14.1 Detailed Description . . . . .	26
5.14.2 Member Function/Subroutine Documentation . . . . .	26
5.14.2.1 min_val_sbr() . . . . .	26
5.15 ez_parallel_structs::scheme Type Reference . . . . .	26
5.15.1 Detailed Description . . . . .	28
5.15.2 Member Data Documentation . . . . .	28
5.15.2.1 coldcmpsizes . . . . .	28

5.15.2.2 coldcmpsizesovlp . . . . .	28
5.15.2.3 colref . . . . .	28
5.15.2.4 colspc . . . . .	28
5.15.2.5 comm . . . . .	29
5.15.2.6 commsize . . . . .	29
5.15.2.7 counts . . . . .	29
5.15.2.8 datatype . . . . .	29
5.15.2.9 displs . . . . .	29
5.15.2.10 gridsize . . . . .	30
5.15.2.11 hslabsize . . . . .	30
5.15.2.12 initfft . . . . .	30
5.15.2.13 initscheme . . . . .	30
5.15.2.14 initspecdrv . . . . .	30
5.15.2.15 norm_1d_1 . . . . .	31
5.15.2.16 norm_1d_2 . . . . .	31
5.15.2.17 norm_2d . . . . .	31
5.15.2.18 ovlp . . . . .	31
5.15.2.19 procid . . . . .	31
5.15.2.20 recv_boundaries . . . . .	32
5.15.2.21 rowdcmpsizes . . . . .	32
5.15.2.22 send_boundaries . . . . .	32
5.15.2.23 subarrays . . . . .	32
5.15.2.24 vslabint . . . . .	32
5.15.2.25 vslabsize . . . . .	33
5.15.2.26 vslabsizeovlp . . . . .	33
5.15.2.27 wsave1 . . . . .	33
5.15.2.28 wsave2 . . . . .	33
5.15.2.29 wvnmb1 . . . . .	33
5.15.2.30 wvnmb2 . . . . .	34
5.16 ez_parallel::share_subgrid_bdry Interface Reference . . . . .	34
5.16.1 Detailed Description . . . . .	34
5.16.2 Member Function/Subroutine Documentation . . . . .	34
5.16.2.1 share_subgrid_bdry_dble_sbr() . . . . .	34
5.16.2.2 share_subgrid_bdry_dcmpx_sbr() . . . . .	35
<b>6 File Documentation</b> . . . . .	<b>37</b>
6.1 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PAR↔ ALLEL_project/EZ_PARALLEL/create_scheme.f90 File Reference . . . . .	37
6.1.1 Function/Subroutine Documentation . . . . .	37
6.1.1.1 create_scheme_eh() . . . . .	37
6.1.1.2 create_scheme_sbr() . . . . .	38
6.2 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PAR↔ ALLEL_project/EZ_PARALLEL/create_scheme_fft.f90 File Reference . . . . .	39

6.2.1 Function/Subroutine Documentation	39
6.2.1.1 create_scheme_fft_eh()	39
6.2.1.2 create_scheme_fft_sbr()	39
6.2.1.3 decompose()	40
6.2.1.4 subarray()	40
6.3 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PAR↔ ALLEL_project/EZ_PARALLEL/create_scheme_spec_drv.f90 File Reference	41
6.3.1 Function/Subroutine Documentation	41
6.3.1.1 create_scheme_spec_drv_eh()	41
6.3.1.2 create_scheme_spec_drv_sbr()	41
6.4 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PAR↔ ALLEL_project/EZ_PARALLEL/create_scheme_zero_pad.f90 File Reference	42
6.4.1 Function/Subroutine Documentation	42
6.4.1.1 create_scheme_zero_pad_dble_eh()	42
6.4.1.2 create_scheme_zero_pad_dble_sbr()	43
6.4.1.3 decompose()	43
6.4.1.4 subarray()	43
6.5 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PAR↔ ALLEL_project/EZ_PARALLEL/destroy_scheme.f90 File Reference	44
6.5.1 Function/Subroutine Documentation	44
6.5.1.1 destroy_scheme_eh()	44
6.5.1.2 destroy_scheme_sbr()	45
6.6 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PAR↔ ALLEL_project/EZ_PARALLEL/execute_scheme_fft.f90 File Reference	45
6.6.1 Function/Subroutine Documentation	45
6.6.1.1 execute_scheme_fft_dble_eh()	45
6.6.1.2 execute_scheme_fft_dble_sbr()	46
6.6.1.3 execute_scheme_fft_dcmpx_eh()	46
6.6.1.4 execute_scheme_fft_dcmpx_sbr()	47
6.7 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PAR↔ ALLEL_project/EZ_PARALLEL/execute_scheme_ifft.f90 File Reference	47
6.7.1 Function/Subroutine Documentation	48
6.7.1.1 execute_scheme_ifft_dble_eh()	48
6.7.1.2 execute_scheme_ifft_dble_sbr()	48
6.7.1.3 execute_scheme_ifft_dcmpx_eh()	49
6.7.1.4 execute_scheme_ifft_dcmpx_sbr()	49
6.8 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PAR↔ ALLEL_project/EZ_PARALLEL/execute_scheme_ispec_drv.f90 File Reference	50
6.8.1 Function/Subroutine Documentation	50
6.8.1.1 execute_scheme_ispec_drv_dble_eh()	50
6.8.1.2 execute_scheme_ispec_drv_dble_sbr()	51
6.8.1.3 execute_scheme_ispec_drv_dcmpx_eh()	51
6.8.1.4 execute_scheme_ispec_drv_dcmpx_sbr()	52

6.9 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL_project/EZ_PARALLEL/execute_scheme_izero_pad.f90 File Reference . . . . .	52
6.9.1 Function/Subroutine Documentation . . . . .	52
6.9.1.1 decompose() . . . . .	53
6.9.1.2 execute_scheme_izero_pad_dble_eh() . . . . .	53
6.9.1.3 execute_scheme_izero_pad_dble_sbr() . . . . .	54
6.9.1.4 execute_scheme_izero_pad_dcmpx_eh() . . . . .	54
6.9.1.5 execute_scheme_izero_pad_dcmpx_sbr() . . . . .	55
6.9.1.6 subarray() . . . . .	55
6.10 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL_project/EZ_PARALLEL/execute_scheme_spec_drv.f90 File Reference . . . . .	56
6.10.1 Function/Subroutine Documentation . . . . .	56
6.10.1.1 execute_scheme_spec_drv_dble_eh() . . . . .	56
6.10.1.2 execute_scheme_spec_drv_dble_sbr() . . . . .	57
6.10.1.3 execute_scheme_spec_drv_dcmpx_eh() . . . . .	57
6.10.1.4 execute_scheme_spec_drv_dcmpx_sbr() . . . . .	58
6.11 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL_project/EZ_PARALLEL/execute_scheme_zero_pad.f90 File Reference . . . . .	58
6.11.1 Function/Subroutine Documentation . . . . .	58
6.11.1.1 decompose() . . . . .	59
6.11.1.2 execute_scheme_zero_pad_dble_eh() . . . . .	59
6.11.1.3 execute_scheme_zero_pad_dble_sbr() . . . . .	60
6.11.1.4 execute_scheme_zero_pad_dcmpx_eh() . . . . .	60
6.11.1.5 execute_scheme_zero_pad_dcmpx_sbr() . . . . .	61
6.11.1.6 subarray() . . . . .	61
6.12 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL_project/EZ_PARALLEL/ez_parallel.f90 File Reference . . . . .	62
6.13 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL_project/EZ_PARALLEL/ez_parallel_structs.f90 File Reference . . . . .	62
6.14 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL_project/EZ_PARALLEL/max_val.f90 File Reference . . . . .	63
6.14.1 Function/Subroutine Documentation . . . . .	63
6.14.1.1 max_val_eh() . . . . .	63
6.14.1.2 max_val_sbr() . . . . .	64
6.15 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL_project/EZ_PARALLEL/min_val.f90 File Reference . . . . .	64
6.15.1 Function/Subroutine Documentation . . . . .	64
6.15.1.1 min_val_eh() . . . . .	64
6.15.1.2 min_val_sbr() . . . . .	65
6.16 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL_project/EZ_PARALLEL/share_subgrid_bdry.f90 File Reference . . . . .	65
6.16.1 Function/Subroutine Documentation . . . . .	65
6.16.1.1 share_subgrid_bdry_dble_eh() . . . . .	66
6.16.1.2 share_subgrid_bdry_dble_sbr() . . . . .	66

6.16.1.3 share_subgrid_bdry_dcmpx_eh() . . . . .	66
6.16.1.4 share_subgrid_bdry_dcmpx_sbr() . . . . .	67

<b>Index</b>	<b>69</b>
--------------	-----------



# Chapter 1

## Modules Index

### 1.1 Modules List

Here is a list of all modules with brief descriptions:

<a href="#">ez_parallel</a>	The EZ_PARALLEL module. Contains EZ_PARALLEL subroutines and their interfaces . . . .	<a href="#">7</a>
<a href="#">ez_parallel_structs</a>	The EZ_PARALLEL structures module. Contains all EZ_PARALLEL derived datatypes . . . .	<a href="#">8</a>



## Chapter 2

# Data Type Index

### 2.1 Data Types List

Here are the data types with brief descriptions:

<a href="#">The</a> . . . . .	11
<a href="#">ez_parallel::create_scheme</a> The interface for the SCHEME creation subroutine . . . . .	11
<a href="#">ez_parallel::create_scheme_fft</a> The interface for the SCHEME FFT initialization subroutine . . . . .	12
<a href="#">ez_parallel::create_scheme_spec_drv</a> The interface for the SCHEME spectral derivative initialization subroutine . . . . .	13
<a href="#">ez_parallel::create_scheme_zero_pad</a> The interface for the SCHEME zero-padding initialization subroutine . . . . .	14
<a href="#">ez_parallel::destroy_scheme</a> The interface for the SCHEME destruction subroutine . . . . .	14
<a href="#">ez_parallel::execute_scheme_fft</a> The FFT execution interface for the EZ_PARALLEL module . . . . .	15
<a href="#">ez_parallel::execute_scheme_ift</a> The inverse FFT execution interface for the EZ_PARALLEL module . . . . .	17
<a href="#">ez_parallel::execute_scheme_ispec_drv</a> The inverse spectral derivative execution interface for the EZ_PARALLEL module . . . . .	19
<a href="#">ez_parallel::execute_scheme_izero_pad</a> The interface for the SCHEME zero-padding removal subroutine . . . . .	20
<a href="#">ez_parallel::execute_scheme_spec_drv</a> The spectral derivative execution interface for the EZ_PARALLEL module . . . . .	22
<a href="#">ez_parallel::execute_scheme_zero_pad</a> The interface for the SCHEME zero-padding execution subroutine . . . . .	23
<a href="#">ez_parallel::max_val</a> The maximum value interface for the EZ_PARALLEL module . . . . .	24
<a href="#">ez_parallel::min_val</a> The minimum value interface for the EZ_PARALLEL module . . . . .	25
<a href="#">ez_parallel_structs::scheme</a> . . . . .	26
<a href="#">ez_parallel::share_subgrid_bdry</a> The sub-grid boundary communication interface for the EZ_PARALLEL module . . . . .	34



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/create_scheme.f90 . . . . .	37
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/create_scheme_fft.f90 . . . . .	39
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/create_scheme_spec_drv.f90 . . . . .	41
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/create_scheme_zero_pad.f90 . . . . .	42
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/destroy_scheme.f90 . . . . .	44
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/execute_scheme_fft.f90 . . . . .	45
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/execute_scheme_iff.f90 . . . . .	47
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/execute_scheme_ispec_drv.f90 . . . . .	50
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/execute_scheme_izero_pad.f90 . . . . .	52
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/execute_scheme_spec_drv.f90 . . . . .	56
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/execute_scheme_zero_pad.f90 . . . . .	58
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/ez_parallel.f90 . . . . .	62
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/ez_parallel_structs.f90 . . . . .	62
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/max_val.f90 . . . . .	63
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/min_val.f90 . . . . .	64
C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALL↔ EL_project/EZ_PARALLEL/share_subgrid_bdry.f90 . . . . .	65



## Chapter 4

# Module Documentation

### 4.1 ez\_parallel Module Reference

The EZ\_PARALLEL module. Contains EZ\_PARALLEL subroutines and their interfaces.

#### Data Types

- interface [create\\_scheme](#)  
*The interface for the SCHEME creation subroutine.*
- interface [create\\_scheme\\_fft](#)  
*The interface for the SCHEME FFT initialization subroutine.*
- interface [create\\_scheme\\_spec\\_drv](#)  
*The interface for the SCHEME spectral derivative initialization subroutine.*
- interface [create\\_scheme\\_zero\\_pad](#)  
*The interface for the SCHEME zero-padding initialization subroutine.*
- interface [destroy\\_scheme](#)  
*The interface for the SCHEME destruction subroutine.*
- interface [execute\\_scheme\\_fft](#)  
*The FFT execution interface for the EZ\_PARALLEL module.*
- interface [execute\\_scheme\\_iff](#)  
*The inverse FFT execution interface for the EZ\_PARALLEL module.*
- interface [execute\\_scheme\\_ispec\\_drv](#)  
*The inverse spectral derivative execution interface for the EZ\_PARALLEL module.*
- interface [execute\\_scheme\\_izero\\_pad](#)  
*The interface for the SCHEME zero-padding removal subroutine.*
- interface [execute\\_scheme\\_spec\\_drv](#)  
*The spectral derivative execution interface for the EZ\_PARALLEL module.*
- interface [execute\\_scheme\\_zero\\_pad](#)  
*The interface for the SCHEME zero-padding execution subroutine.*
- interface [max\\_val](#)  
*The maximum value interface for the EZ\_PARALLEL module.*
- interface [min\\_val](#)  
*The minimum value interface for the EZ\_PARALLEL module.*
- interface [share\\_subgrid\\_bdry](#)  
*The sub-grid boundary communication interface for the EZ\_PARALLEL module.*

### 4.1.1 Detailed Description

The EZ\_PARALLEL module. Contains EZ\_PARALLEL subroutines and their interfaces.

Author

Jason Turner

## 4.2 ez\_parallel\_structs Module Reference

The EZ\_PARALLEL structures module. Contains all EZ\_PARALLEL derived datatypes.

### Data Types

- type [scheme](#)

### Variables

- integer, parameter, public [fft\\_1d\\_1](#) = 51  
*Flag to mark execution of 1D FFTs along the first dimension.*
- integer, parameter, public [fft\\_1d\\_2](#) = 46  
*Flag to mark execution of 1D FFTs along the second dimension.*
- integer, parameter, public [fft\\_2d](#) = 95  
*Flag to mark execution of 2D FFTs.*
- integer, parameter, public [spec\\_drv\\_1d\\_1](#) = 23  
*Flag to mark execution of 1D spectral derivatives along the first dimension.*
- integer, parameter, public [spec\\_drv\\_1d\\_2](#) = 78  
*Flag to mark execution of 1D spectral derivatives along the second dimension.*

### 4.2.1 Detailed Description

The EZ\_PARALLEL structures module. Contains all EZ\_PARALLEL derived datatypes.

Author

Jason Turner

### 4.2.2 Variable Documentation



#### 4.2.2.1 fft\_1d\_1

```
integer, parameter, public ez_parallel_structs::fft_1d_1 = 51
```

Flag to mark execution of 1D FFTs along the first dimension.

Definition at line 93 of file ez\_parallel\_structs.f90.

#### 4.2.2.2 fft\_1d\_2

```
integer, parameter, public ez_parallel_structs::fft_1d_2 = 46
```

Flag to mark execution of 1D FFTs along the second dimension.

Definition at line 96 of file ez\_parallel\_structs.f90.

#### 4.2.2.3 fft\_2d

```
integer, parameter, public ez_parallel_structs::fft_2d = 95
```

Flag to mark execution of 2D FFTs.

Definition at line 99 of file ez\_parallel\_structs.f90.

#### 4.2.2.4 spec\_drv\_1d\_1

```
integer, parameter, public ez_parallel_structs::spec_drv_1d_1 = 23
```

Flag to mark execution of 1D spectral derivatives along the first dimension.

Definition at line 102 of file ez\_parallel\_structs.f90.

#### 4.2.2.5 spec\_drv\_1d\_2

```
integer, parameter, public ez_parallel_structs::spec_drv_1d_2 = 78
```

Flag to mark execution of 1D spectral derivatives along the second dimension.

Definition at line 105 of file ez\_parallel\_structs.f90.



## Chapter 5

# Data Type Documentation

### 5.1 The Module Reference

#### 5.1.1 Detailed Description

SCHEME derived datatype contains all information about the grid and the sub-grid decomposition of the grid, as well as information needed for FFTs.

The documentation for this module was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↔\_project/EZ\_PARALLEL/[ez\\_parallel\\_structs.f90](#)

### 5.2 ez\_parallel::create\_scheme Interface Reference

The interface for the SCHEME creation subroutine.

#### Private Member Functions

- subroutine [create\\_scheme\\_sbr](#) (rowCount, colCount, colSpc, colRef, comm, mpiDatatype, ovlp, sch)

#### 5.2.1 Detailed Description

The interface for the SCHEME creation subroutine.

The CREATE\_SCHEME subroutine initializes a SCHEME which holds information about the grid and grid decomposition, as well as variables for use in FFTs. For greater detail on the SCHEME creation subroutine, see [create\\_scheme.f90](#). For greater detail on the SCHEME datatype, see [ez\\_parallel\\_structs.f90](#).

Definition at line 56 of file ez\_parallel.f90.

## 5.2.2 Member Function/Subroutine Documentation

### 5.2.2.1 create\_scheme\_sbr()

```
subroutine ez_parallel::create_scheme::create_scheme_sbr (
    integer, intent(in) rowCount,
    integer, intent(inout) colCount,
    double precision, intent(in) colSpC,
    double precision, intent(inout) colRef,
    integer, intent(in) comm,
    integer, intent(in) mpiDatatype,
    integer, intent(in) ovlp,
    type(scheme), intent(inout) sch ) [private]
```

Definition at line 59 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↔  
\_project/EZ\_PARALLEL/ez\_parallel.f90

## 5.3 ez\_parallel::create\_scheme\_fft Interface Reference

The interface for the SCHEME FFT initialization subroutine.

### Private Member Functions

- subroutine [create\\_scheme\\_fft\\_sbr](#) (sch)

### 5.3.1 Detailed Description

The interface for the SCHEME FFT initialization subroutine.

The CREATE\_SCHEME\_FFT subroutine initializes a SCHEME for future FFTs. For greater detail on the SCH↔  
EME FFT initialization subroutine, see [create\\_scheme\\_fft.f90](#). For greater detail on the SCHEME datatype, see [ez\\_parallel\\_structs.f90](#).

Definition at line 81 of file ez\_parallel.f90.

## 5.3.2 Member Function/Subroutine Documentation

### 5.3.2.1 create\_scheme\_fft\_sbr()

```
subroutine ez_parallel::create_scheme_fft::create_scheme_fft_sbr (
    type(scheme), intent(inout) sch ) [private]
```

Definition at line 83 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↔\_project/EZ\_PARALLEL/[ez\\_parallel.f90](#)

## 5.4 ez\_parallel::create\_scheme\_spec\_drv Interface Reference

The interface for the SCHEME spectral derivative initialization subroutine.

### Private Member Functions

- subroutine [create\\_scheme\\_spec\\_drv\\_sbr](#) (sch)

### 5.4.1 Detailed Description

The interface for the SCHEME spectral derivative initialization subroutine.

The CREATE\_SCHEME\_SPEC\_DERV subroutine initializes a SCHEME for future spectral derivatives. For greater detail on the SCHEME spectral derivative initialization subroutine, see [create\\_scheme\\_spec\\_drv.f90](#). For greater detail on the SCHEME datatype, see [ez\\_parallel\\_structs.f90](#).

Definition at line 99 of file ez\_parallel.f90.

### 5.4.2 Member Function/Subroutine Documentation

#### 5.4.2.1 create\_scheme\_spec\_drv\_sbr()

```
subroutine ez_parallel::create_scheme_spec_drv::create_scheme_spec_drv_sbr (
    type(scheme), intent(inout) sch ) [private]
```

Definition at line 101 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↔\_project/EZ\_PARALLEL/[ez\\_parallel.f90](#)

## 5.5 ez\_parallel::create\_scheme\_zero\_pad Interface Reference

The interface for the SCHEME zero-padding initialization subroutine.

### Private Member Functions

- subroutine [create\\_scheme\\_zero\\_pad\\_dble\\_sbr](#) (sch, schZP)

### 5.5.1 Detailed Description

The interface for the SCHEME zero-padding initialization subroutine.

The CREATE\_SCHEME\_ZERO\_PAD subroutine initializes a SCHEME to handle a zero-padded version of the global array. For greater detail on the SCHEME spectral derivative initialization subroutine, see [create\\_scheme\\_zero\\_pad.f90](#). For greater detail on the SCHEME datatype, see [ez\\_parallel\\_structs.f90](#).

Definition at line 117 of file ez\_parallel.f90.

### 5.5.2 Member Function/Subroutine Documentation

#### 5.5.2.1 create\_scheme\_zero\_pad\_dble\_sbr()

```
subroutine ez_parallel::create_scheme_zero_pad::create_scheme_zero_pad_dble_sbr (
    type(scheme), intent(in) sch,
    type(scheme), intent(inout) schZP ) [private]
```

#### Parameters

in	<i>sch</i>	Scheme associated with arr.
in, out	<i>schZP</i>	SCHEME associated with arrZP.

Definition at line 121 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↵\_project/EZ\_PARALLEL/[ez\\_parallel.f90](#)

## 5.6 ez\_parallel::destroy\_scheme Interface Reference

The interface for the SCHEME destruction subroutine.

## Private Member Functions

- subroutine [destroy\\_scheme\\_sbr](#) (sch)

### 5.6.1 Detailed Description

The interface for the SCHEME destruction subroutine.

The DESTROY\_SCHEME subroutine deallocates a SCHEME. For greater detail on the SCHEME destruction subroutine, see [destroy\\_scheme.f90](#). For greater detail on the SCHEME datatype, see [ez\\_parallel\\_structs.f90](#).

Definition at line 136 of file ez\_parallel.f90.

### 5.6.2 Member Function/Subroutine Documentation

#### 5.6.2.1 destroy\_scheme\_sbr()

```
subroutine ez_parallel::destroy_scheme::destroy_scheme_sbr (
    type(scheme), intent(inout) sch ) [private]
```

Parameters

in	<i>sch</i>	SCHEME to be destroyed.
----	------------	-------------------------

Definition at line 139 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↔  
\_project/EZ\_PARALLEL/[ez\\_parallel.f90](#)

## 5.7 ez\_parallel::execute\_scheme\_fft Interface Reference

The FFT execution interface for the EZ\_PARALLEL module.

## Public Member Functions

- subroutine [execute\\_scheme\\_fft\\_dcmpx\\_sbr](#) (subGrid, kind, sch)

## Private Member Functions

- subroutine [execute\\_scheme\\_fft\\_dble\\_sbr](#) (subGrid, kind, sch)

### 5.7.1 Detailed Description

The FFT execution interface for the `EZ_PARALLEL` module.

For greater detail on the FFT execution subroutine, see [execute\\_scheme\\_fft.f90](#).

Definition at line 180 of file `ez_parallel.f90`.

### 5.7.2 Member Function/Subroutine Documentation

#### 5.7.2.1 execute\_scheme\_fft\_dble\_sbr()

```
subroutine ez_parallel::execute_scheme_fft::execute_scheme_fft_dble_sbr (
    double precision, dimension(:, :), intent(inout) subGrid,
    integer, intent(in) kind,
    type(scheme), intent(in) sch ) [private]
```

##### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of FFT to execute, one of <code>FFT_1D_1</code> , <code>FFT_1D_2</code> , or <code>FFT_2D</code> .
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 187 of file `ez_parallel.f90`.

#### 5.7.2.2 execute\_scheme\_fft\_dcmpx\_sbr()

```
subroutine ez_parallel::execute_scheme_fft::execute_scheme_fft_dcmpx_sbr (
    double complex, dimension(:, :), intent(inout) subGrid,
    integer, intent(in) kind,
    type(scheme), intent(in) sch )
```

##### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of FFT to execute, one of <code>FFT_1D_1</code> , <code>FFT_1D_2</code> , or <code>FFT_2D</code> .
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 199 of file `ez_parallel.f90`.

The documentation for this interface was generated from the following file:

- `C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL↵_project/EZ_PARALLEL/ez_parallel.f90`



## 5.8 ez\_parallel::execute\_scheme\_ifft Interface Reference

The inverse FFT execution interface for the EZ\_PARALLEL module.

### Public Member Functions

- subroutine [execute\\_scheme\\_ifft\\_dcmpx\\_sbr](#) (subGrid, kind, sch)

### Private Member Functions

- subroutine [execute\\_scheme\\_ifft\\_dble\\_sbr](#) (subGrid, kind, sch)

### 5.8.1 Detailed Description

The inverse FFT execution interface for the EZ\_PARALLEL module.

For greater detail on the inverse FFT execution subroutine, see [execute\\_scheme\\_ifft.f90](#).

Definition at line 213 of file ez\_parallel.f90.

### 5.8.2 Member Function/Subroutine Documentation

#### 5.8.2.1 execute\_scheme\_ifft\_dble\_sbr()

```
subroutine ez_parallel::execute_scheme_ifft::execute_scheme_ifft_dble_sbr (
    double precision, dimension(:,,:), intent(inout) subGrid,
    integer, intent(in) kind,
    type(scheme), intent(in) sch ) [private]
```

##### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of FFT to execute, one of FFT_1D_1, FFT_1D_2, or FFT_2D.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 220 of file ez\_parallel.f90.

#### 5.8.2.2 execute\_scheme\_ifft\_dcmpx\_sbr()

```
subroutine ez_parallel::execute_scheme_ifft::execute_scheme_ifft_dcmpx_sbr (
    double complex, dimension(:,,:), intent(inout) subGrid,
```

```
integer, intent(in) kind,  
type(scheme), intent(in) sch )
```

## Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of FFT to execute, one of FFT_1D_1, FFT_1D_2, or FFT_2D.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 232 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL/\_project/EZ\_PARALLEL/[ez\\_parallel.f90](#)

## 5.9 ez\_parallel::execute\_scheme\_ispec\_drv Interface Reference

The inverse spectral derivative execution interface for the EZ\_PARALLEL module.

### Public Member Functions

- subroutine [execute\\_scheme\\_ispec\\_drv\\_dcmpx\\_sbr](#) (subGrid, kind, order, sch)

### Private Member Functions

- subroutine [execute\\_scheme\\_ispec\\_drv\\_dble\\_sbr](#) (subGrid, kind, order, sch)

### 5.9.1 Detailed Description

The inverse spectral derivative execution interface for the EZ\_PARALLEL module.

For greater detail on the inverse spectral derivative execution subroutine, see [execute\\_scheme\\_ispec\\_drv.f90](#).

Definition at line 285 of file ez\_parallel.f90.

### 5.9.2 Member Function/Subroutine Documentation

#### 5.9.2.1 execute\_scheme\_ispec\_drv\_dble\_sbr()

```
subroutine ez_parallel::execute_scheme_ispec_drv::execute_scheme_ispec_drv_dble_sbr (
    double precision, dimension(:,,:), intent(inout) subGrid,
    integer, intent(in) kind,
    integer, intent(in) order,
    type(scheme), intent(in) sch ) [private]
```

**Parameters**

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of spectral derivative to execute, one of SPEC_DRV_1D_1, SPEC_DERV_1D_2.
in	<i>order</i>	Order of the spectral derivative.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 294 of file ez\_parallel.f90.

**5.9.2.2 execute\_scheme\_ispec\_drv\_dcmpx\_sbr()**

```
subroutine ez_parallel::execute_scheme_ispec_drv::execute_scheme_ispec_drv_dcmpx_sbr (
    double complex, dimension(:, :), intent(inout) subGrid,
    integer, intent(in) kind,
    integer, intent(in) order,
    type(scheme), intent(in) sch )
```

**Parameters**

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of spectral derivative to execute, one of SPEC_DRV_1D_1, SPEC_DERV_1D_2.
in	<i>order</i>	Order of the spectral derivative.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 309 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↔\_project/EZ\_PARALLEL/[ez\\_parallel.f90](#)

**5.10 ez\_parallel::execute\_scheme\_izero\_pad Interface Reference**

The interface for the SCHEME zero-padding removal subroutine.

**Public Member Functions**

- subroutine [execute\\_scheme\\_izero\\_pad\\_dcmpx\\_sbr](#) (arr, sch, arrZP, schZP)

**Private Member Functions**

- subroutine [execute\\_scheme\\_izero\\_pad\\_dble\\_sbr](#) (arr, sch, arrZP, schZP)

### 5.10.1 Detailed Description

The interface for the SCHEME zero-padding removal subroutine.

The EXECUTE\_SCHEME\_IZERO\_PAD subroutine returns a non-zero-padded version of the array. For greater detail on the SCHEME zero-padding removal subroutine, see [execute\\_scheme\\_izero\\_pad.f90](#). For greater detail on the SCHEME datatype, see [ez\\_parallel\\_structs.f90](#).

Definition at line 364 of file ez\_parallel.f90.

### 5.10.2 Member Function/Subroutine Documentation

#### 5.10.2.1 execute\_scheme\_izero\_pad\_dble\_sbr()

```
subroutine ez_parallel::execute_scheme_izero_pad::execute_scheme_izero_pad_dble_sbr (
    double precision, dimension(:, :), intent(inout) arr,
    type(scheme), intent(in) sch,
    double precision, dimension(:, :), intent(in) arrZP,
    type(scheme), intent(in) schZP ) [private]
```

##### Parameters

in, out	<i>arr</i>	Array to be un-zero-padded.
in	<i>sch</i>	Scheme associated with arr.
in	<i>arrZP</i>	Allocatable array that will hold the zero-padded arr.
in	<i>schZP</i>	SCHEME associated with arrZP.

Definition at line 370 of file ez\_parallel.f90.

#### 5.10.2.2 execute\_scheme\_izero\_pad\_dcmpx\_sbr()

```
subroutine ez_parallel::execute_scheme_izero_pad::execute_scheme_izero_pad_dcmpx_sbr (
    double complex, dimension(:, :), intent(inout) arr,
    type(scheme), intent(in) sch,
    double complex, dimension(:, :), intent(in) arrZP,
    type(scheme), intent(in) schZP )
```

##### Parameters

in, out	<i>arr</i>	Array to be un-zero-padded.
in	<i>sch</i>	Scheme associated with arr.
in	<i>arrZP</i>	Allocatable array that will hold the zero-padded arr.
in	<i>schZP</i>	SCHEME associated with arrZP.

Definition at line 382 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↔\_project/EZ\_PARALLEL/ez\_parallel.f90

## 5.11 ez\_parallel::execute\_scheme\_spec\_drv Interface Reference

The spectral derivative execution interface for the EZ\_PARALLEL module.

### Public Member Functions

- subroutine [execute\\_scheme\\_spec\\_drv\\_dcmpx\\_sbr](#) (subGrid, kind, order, sch)

### Private Member Functions

- subroutine [execute\\_scheme\\_spec\\_drv\\_dble\\_sbr](#) (subGrid, kind, order, sch)

#### 5.11.1 Detailed Description

The spectral derivative execution interface for the EZ\_PARALLEL module.

For greater detail on the spectral derivative execution subroutine, see [execute\\_scheme\\_spec\\_drv.f90](#).

Definition at line 246 of file ez\_parallel.f90.

#### 5.11.2 Member Function/Subroutine Documentation

##### 5.11.2.1 execute\_scheme\_spec\_drv\_dble\_sbr()

```
subroutine ez_parallel::execute_scheme_spec_drv::execute_scheme_spec_drv_dble_sbr (
    double precision, dimension(:,:), intent(inout) subGrid,
    integer, intent(in) kind,
    integer, intent(in) order,
    type(scheme), intent(in) sch ) [private]
```

##### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of spectral derivative to execute, one of SPEC_DRV_1D_1, SPEC_DERV_1D_2.
in	<i>order</i>	Order of the spectral derivative.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 255 of file ez\_parallel.f90.

### 5.11.2.2 execute\_scheme\_spec\_drv\_dcmpx\_sbr()

```
subroutine ez_parallel::execute_scheme_spec_drv::execute_scheme_spec_drv_dcmpx_sbr (
    double complex, dimension(:, :), intent(inout) subGrid,
    integer, intent(in) kind,
    integer, intent(in) order,
    type(scheme), intent(in) sch )
```

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of spectral derivative to execute, one of SPEC_DRV_1D_1, SPEC_DERV_1D_2.
in	<i>order</i>	Order of the spectral derivative.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 270 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↔\_project/EZ\_PARALLEL/ez\_parallel.f90

## 5.12 ez\_parallel::execute\_scheme\_zero\_pad Interface Reference

The interface for the SCHEME zero-padding execution subroutine.

### Public Member Functions

- subroutine [execute\\_scheme\\_zero\\_pad\\_dcmpx\\_sbr](#) (arr, sch, arrZP, schZP)

### Private Member Functions

- subroutine [execute\\_scheme\\_zero\\_pad\\_dble\\_sbr](#) (arr, sch, arrZP, schZP)

### 5.12.1 Detailed Description

The interface for the SCHEME zero-padding execution subroutine.

The EXECUTE\_SCHEME\_ZERO\_PAD subroutine zero-pads the global array. For greater detail on the SCHEME zero-padding subroutine, see [execute\\_scheme\\_zero\\_pad.f90](#). For greater detail on the SCHEME datatype, see [ez\\_parallel\\_structs.f90](#).

Definition at line 327 of file ez\_parallel.f90.

## 5.12.2 Member Function/Subroutine Documentation

### 5.12.2.1 `execute_scheme_zero_pad_dble_sbr()`

```
subroutine ez_parallel::execute_scheme_zero_pad::execute_scheme_zero_pad_dble_sbr (
    double precision, dimension(:, :), intent(in) arr,
    type(scheme), intent(in) sch,
    double precision, dimension(:, :), intent(inout), allocatable arrZP,
    type(scheme), intent(in) schZP ) [private]
```

#### Parameters

in	<i>arr</i>	Array to be zero-padded.
in	<i>sch</i>	Scheme associated with arr.
in, out	<i>arrZP</i>	Allocatable array that will hold the zero-padded arr.
in	<i>schZP</i>	SCHEME associated with arrZP.

Definition at line 333 of file `ez_parallel.f90`.

### 5.12.2.2 `execute_scheme_zero_pad_dcmpx_sbr()`

```
subroutine ez_parallel::execute_scheme_zero_pad::execute_scheme_zero_pad_dcmpx_sbr (
    double complex, dimension(:, :), intent(in) arr,
    type(scheme), intent(in) sch,
    double complex, dimension(:, :), intent(inout), allocatable arrZP,
    type(scheme), intent(in) schZP )
```

#### Parameters

in	<i>arr</i>	Array to be zero-padded.
in	<i>sch</i>	Scheme associated with arr.
in, out	<i>arrZP</i>	Allocatable array that will hold the zero-padded arr.
in	<i>schZP</i>	SCHEME associated with arrZP.

Definition at line 345 of file `ez_parallel.f90`.

The documentation for this interface was generated from the following file:

- `C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL↵_project/EZ_PARALLEL/ez_parallel.f90`

## 5.13 `ez_parallel::max_val` Interface Reference

The maximum value interface for the `EZ_PARALLEL` module.



## Private Member Functions

- subroutine [max\\_val\\_sbr](#) (subGrid, maxValue, sch)

### 5.13.1 Detailed Description

The maximum value interface for the EZ\_PARALLEL module.

For greater detail on the maximum value subroutine, see [max\\_val.f90](#).

Definition at line 395 of file ez\_parallel.f90.

### 5.13.2 Member Function/Subroutine Documentation

#### 5.13.2.1 max\_val\_sbr()

```
subroutine ez_parallel::max_val::max_val_sbr (
    double precision, dimension(:, :), intent(in) subGrid,
    double precision, intent(inout) maxValue,
    type(scheme), intent(in) sch ) [private]
```

#### Parameters

in	<i>subGrid</i>	The local sub-grid.
in, out	<i>maxVal</i>	The variable to store the maximum value across all sub-grids.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 402 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↵\_project/EZ\_PARALLEL/[ez\\_parallel.f90](#)

## 5.14 ez\_parallel::min\_val Interface Reference

The minimum value interface for the EZ\_PARALLEL module.

## Private Member Functions

- subroutine [min\\_val\\_sbr](#) (subGrid, minVal, sch)

### 5.14.1 Detailed Description

The minimum value interface for the `EZ_PARALLEL` module.

For greater detail on the minimum value subroutine, see [min\\_val.f90](#).

Definition at line 414 of file `ez_parallel.f90`.

### 5.14.2 Member Function/Subroutine Documentation

#### 5.14.2.1 min\_val\_sbr()

```
subroutine ez_parallel::min_val::min_val_sbr (
    double precision, dimension(:, :), intent(in) subGrid,
    double precision, intent(inout) minVal,
    type(scheme), intent(in) sch ) [private]
```

##### Parameters

in	<i>subGrid</i>	The local sub-grid.
in, out	<i>minVal</i>	The variable to store the minimum value across all sub-grids.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 421 of file `ez_parallel.f90`.

The documentation for this interface was generated from the following file:

- `C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ_PARALLEL↔_project/EZ_PARALLEL/ez_parallel.f90`

## 5.15 ez\_parallel\_structs::scheme Type Reference

### Public Attributes

- integer, dimension(0:1) [gridsize](#)  
*Size in each dimension of the grid.*
- double precision [colspc](#)  
*Physical spacing between columns of the grid.*
- integer [comm](#)  
*Communicator that holds the grid.*
- integer [commsize](#)  
*Number of processes in the MPI communicator.*
- integer [datatype](#)  
*Datatype of the array.*
- integer [ovlp](#)

- Number of extra columns needed by each sub-grid to successfully step forward in time.*

  - integer, dimension(:), allocatable [rowdcmpsizes](#)
- Number of rows in each sub-grid of the horizontal-slab decomposition, excluding overlap.*

  - integer, dimension(:), allocatable [coldcmpsizes](#)
- Number of columns in each sub-grid of the vertical-slab decomposition, excluding overlap.*

  - integer, dimension(:), allocatable [coldcmpsizesovlp](#)
- Number of columns in each sub-grid of the vertical slab decomposition, including overlap.*

  - integer [procid](#)
- Processor ID.*

  - integer, dimension(0:1) [hslabsize](#)
- Sizes in each dimension of the sub-grid in the horizontal-slab decomposition of the global array.*

  - integer, dimension(0:1) [vslabsize](#)
- Sizes in each dimension of the sub-grid in the vertical-slab decomposition of the global array, excluding overlap.*

  - integer, dimension(0:1) [vslabsizeovlp](#)
- Sizes in each dimension of the sub-grid in the vertical-slab decomposition of the global array, including overlap.*

  - integer, dimension(0:1) [vslabint](#)
- Column indices of the interior of the vertical slab.*

  - double precision [colref](#)
- The physical position of the reference point in the dimension along the rows (corresponding to a column).*

  - integer, dimension(0:1) [send\\_boundaries](#)
- MPI derived datatype for sending sub-grid boundaries to neighboring sub-grids (0 = left, 1 = right).*

  - integer, dimension(0:1) [recv\\_boundaries](#)
- MPI derived datatype for receiving sub-grid boundaries from neighboring sub-grids (0 = left, 1 = right).*

  - double precision, dimension(:), allocatable [wsave1](#)
- Holds initialization info for DFFTPACK 1-D FFTS along first dimension.*

  - double precision, dimension(:), allocatable [wsave2](#)
- Holds initialization info for DFFTPACK 1-D FFTS along second dimension.*

  - double precision [norm\\_1d\\_1](#)
- The normalization coefficient for possible 1-D FFTs along the first dimension.*

  - double precision [norm\\_1d\\_2](#)
- The normalization coefficient for possible 1-D FFTs along the second dimension.*

  - double precision [norm\\_2d](#)
- The normalization coefficient for possible 2D FFTs.*

  - integer, dimension(:, :), allocatable [subarrays](#)
- Holds the datatypes necessary to perform the transposition.*

  - integer, dimension(:), allocatable [counts](#)
  - integer, dimension(:), allocatable [displs](#)
- Arrays for use in global.*

  - double complex, dimension(:), allocatable [wvnmb1](#)
- Holds coefficients for spectral derivative along the first dimension.*

  - double complex, dimension(:), allocatable [wvnmb2](#)
- Holds coefficients for spectral derivative along the second dimension.*

  - logical [initscheme](#) = .FALSE.
- Checks if SCHEME was created already.*

  - logical [initfft](#) = .FALSE.
- Checks if FFTs for SCHEME were initialized already.*

  - logical [initspecdrv](#) = .FALSE.
- Checks if spectral derivatives for SCHEME were initialized already.*

### 5.15.1 Detailed Description

Definition at line 29 of file ez\_parallel\_structs.f90.

### 5.15.2 Member Data Documentation

#### 5.15.2.1 coldcmpsizes

```
integer, dimension(:), allocatable ez_parallel_structs::scheme::coldcmpsizes
```

Number of columns in each sub-grid of the vertical-slab decomposition, excluding overlap.

Definition at line 40 of file ez\_parallel\_structs.f90.

#### 5.15.2.2 coldcmpsizesovlp

```
integer, dimension(:), allocatable ez_parallel_structs::scheme::coldcmpsizesovlp
```

Number of columns in each sub-grid of the vertical slab decomposition, including overlap.

Definition at line 42 of file ez\_parallel\_structs.f90.

#### 5.15.2.3 colref

```
double precision ez_parallel_structs::scheme::colref
```

The physical position of the reference point in the dimension along the rows (corresponding to a column).

Definition at line 55 of file ez\_parallel\_structs.f90.

#### 5.15.2.4 colspc

```
double precision ez_parallel_structs::scheme::colspc
```

Physical spacing between columns of the grid.

Definition at line 32 of file ez\_parallel\_structs.f90.

#### 5.15.2.5 comm

```
integer ez_parallel_structs::scheme::comm
```

Communicator that holds the grid.

Definition at line 33 of file ez\_parallel\_structs.f90.

#### 5.15.2.6 commsize

```
integer ez_parallel_structs::scheme::commsize
```

Number of processes in the MPI communicator.

Definition at line 34 of file ez\_parallel\_structs.f90.

#### 5.15.2.7 counts

```
integer, dimension(:), allocatable ez_parallel_structs::scheme::counts
```

Definition at line 75 of file ez\_parallel\_structs.f90.

#### 5.15.2.8 datatype

```
integer ez_parallel_structs::scheme::datatype
```

Datatype of the array.

Definition at line 35 of file ez\_parallel\_structs.f90.

#### 5.15.2.9 displs

```
integer, dimension(:), allocatable ez_parallel_structs::scheme::displs
```

Arrays for use in global.

Definition at line 75 of file ez\_parallel\_structs.f90.

#### 5.15.2.10 **gridsize**

```
integer, dimension(0:1) ez_parallel_structs::scheme::gridsize
```

Size in each dimension of the grid.

Definition at line 31 of file ez\_parallel\_structs.f90.

#### 5.15.2.11 **hslabsize**

```
integer, dimension(0:1) ez_parallel_structs::scheme::hslabsize
```

Sizes in each dimension of the sub-grid in the horizontal-slab decomposition of the global array.

Definition at line 47 of file ez\_parallel\_structs.f90.

#### 5.15.2.12 **initfft**

```
logical ez_parallel_structs::scheme::initfft = .FALSE.
```

Checks if FFTs for SCHEME were initialized already.

Definition at line 86 of file ez\_parallel\_structs.f90.

#### 5.15.2.13 **initscheme**

```
logical ez_parallel_structs::scheme::initscheme = .FALSE.
```

Checks if SCHEME was created already.

Definition at line 84 of file ez\_parallel\_structs.f90.

#### 5.15.2.14 **initspecdrv**

```
logical ez_parallel_structs::scheme::initspecdrv = .FALSE.
```

Checks if spectral derivatives for SCHEME were initialized already.

Definition at line 88 of file ez\_parallel\_structs.f90.

#### 5.15.2.15 norm\_1d\_1

```
double precision ez_parallel_structs::scheme::norm_1d_1
```

The normalization coefficient for possible 1-D FFTs along the first dimension.

Definition at line 67 of file ez\_parallel\_structs.f90.

#### 5.15.2.16 norm\_1d\_2

```
double precision ez_parallel_structs::scheme::norm_1d_2
```

The normalization coefficient for possible 1-D FFTs along the second dimension.

Definition at line 69 of file ez\_parallel\_structs.f90.

#### 5.15.2.17 norm\_2d

```
double precision ez_parallel_structs::scheme::norm_2d
```

The normalization coefficient for possible 2D FFTs.

Definition at line 71 of file ez\_parallel\_structs.f90.

#### 5.15.2.18 ovlp

```
integer ez_parallel_structs::scheme::ovlp
```

Number of extra columns needed by each sub-grid to successfully step forward in time.

Definition at line 36 of file ez\_parallel\_structs.f90.

#### 5.15.2.19 procid

```
integer ez_parallel_structs::scheme::procid
```

Processor ID.

Definition at line 46 of file ez\_parallel\_structs.f90.

#### 5.15.2.20 `recv_boundaries`

```
integer, dimension(0:1) ez_parallel_structs::scheme::recv_boundaries
```

MPI derived datatype for receiving sub-grid boundaries from neighboring sub-grids (0 = left, 1 = right).

Definition at line 59 of file `ez_parallel_structs.f90`.

#### 5.15.2.21 `rowdcmpsizes`

```
integer, dimension(:), allocatable ez_parallel_structs::scheme::rowdcmpsizes
```

Number of rows in each sub-grid of the horizontal-slab decomposition, excluding overlap.

Definition at line 38 of file `ez_parallel_structs.f90`.

#### 5.15.2.22 `send_boundaries`

```
integer, dimension(0:1) ez_parallel_structs::scheme::send_boundaries
```

MPI derived datatype for sending sub-grid boundaries to neighboring sub-grids (0 = left, 1 = right).

Definition at line 57 of file `ez_parallel_structs.f90`.

#### 5.15.2.23 `subarrays`

```
integer, dimension(:, :), allocatable ez_parallel_structs::scheme::subarrays
```

Holds the datatypes necessary to perform the transposition.

Definition at line 73 of file `ez_parallel_structs.f90`.

#### 5.15.2.24 `vslabint`

```
integer, dimension(0:1) ez_parallel_structs::scheme::vslabint
```

Column indices of the interior of the vertical slab.

Definition at line 53 of file `ez_parallel_structs.f90`.



#### 5.15.2.25 vslabsize

integer, dimension(0:1) ez\_parallel\_structs::scheme::vslabsize

Sizes in each dimension of the sub-grid in the vertical-slab decomposition of the global array, excluding overlap.

Definition at line 49 of file ez\_parallel\_structs.f90.

#### 5.15.2.26 vslabsizeovlp

integer, dimension(0:1) ez\_parallel\_structs::scheme::vslabsizeovlp

Sizes in each dimension of the sub-grid in the vertical-slab decomposition of the global array, including overlap.

Definition at line 51 of file ez\_parallel\_structs.f90.

#### 5.15.2.27 wsave1

double precision, dimension(:), allocatable ez\_parallel\_structs::scheme::wsave1

Holds initialization info for DFFTPACK 1-D FFTS along first dimension.

Definition at line 63 of file ez\_parallel\_structs.f90.

#### 5.15.2.28 wsave2

double precision, dimension(:), allocatable ez\_parallel\_structs::scheme::wsave2

Holds initialization info for DFFTPACK 1-D FFTS along second dimension.

Definition at line 65 of file ez\_parallel\_structs.f90.

#### 5.15.2.29 wvnmb1

double complex, dimension(:), allocatable ez\_parallel\_structs::scheme::wvnmb1

Holds coefficients for spectral derivative along the first dimension.

Definition at line 78 of file ez\_parallel\_structs.f90.

### 5.15.2.30 wvnmbr2

```
double complex, dimension(:), allocatable ez_parallel_structs::scheme::wvnmbr2
```

Holds coefficients for spectral derivative along the second dimension.

Definition at line 80 of file ez\_parallel\_structs.f90.

The documentation for this type was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↵\_project/EZ\_PARALLEL/ez\_parallel\_structs.f90

## 5.16 ez\_parallel::share\_subgrid\_bdry Interface Reference

The sub-grid boundary communication interface for the EZ\_PARALLEL module.

### Public Member Functions

- subroutine [share\\_subgrid\\_bdry\\_dcmpx\\_sbr](#) (subGrid, sch)

### Private Member Functions

- subroutine [share\\_subgrid\\_bdry\\_dble\\_sbr](#) (subGrid, sch)

### 5.16.1 Detailed Description

The sub-grid boundary communication interface for the EZ\_PARALLEL module.

Communicates the sub-grid boundary to neighboring sub-grids.

For greater detail on the sub-grid boundary communication subroutine, see [share\\_subgrid\\_bdry.f90](#).

Definition at line 153 of file ez\_parallel.f90.

### 5.16.2 Member Function/Subroutine Documentation

#### 5.16.2.1 share\_subgrid\_bdry\_dble\_sbr()

```
subroutine ez_parallel::share_subgrid_bdry::share_subgrid_bdry_dble_sbr (
    double precision, dimension(:,:), intent(inout) subGrid,
    type(scheme), intent(in) sch ) [private]
```

## Parameters

in, out	<i>subGrid</i>	The sub-grid belonging to the processor.
in	<i>sch</i>	SCHEME that holds grid decomposition information, etc.

Definition at line 158 of file ez\_parallel.f90.

### 5.16.2.2 share\_subgrid\_bdry\_dcmpx\_sbr()

```
subroutine ez_parallel::share_subgrid_bdry::share_subgrid_bdry_dcmpx_sbr (  
    double complex, dimension(:, :), intent(inout) subGrid,  
    type(scheme), intent(in) sch )
```

## Parameters

in, out	<i>subGrid</i>	The sub-grid belonging to the processor.
in	<i>sch</i>	SCHEME that holds grid decomposition information, etc.

Definition at line 168 of file ez\_parallel.f90.

The documentation for this interface was generated from the following file:

- C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL↔  
\_project/EZ\_PARALLEL/[ez\\_parallel.f90](#)



## Chapter 6

# File Documentation

### 6.1 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme.f90 File Reference

#### Functions/Subroutines

- subroutine `create_scheme_sbr` (rowCount, colCount, colSpc, colRef, comm, mpiDatatype, ovlp, sch)  
*The SCHEME creation subroutine.*
- subroutine `create_scheme_eh` (rowCount, colCount, ovlp, sch)  
*The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

#### 6.1.1 Function/Subroutine Documentation

##### 6.1.1.1 create\_scheme\_eh()

```
subroutine create_scheme_sbr::create_scheme_eh (  
    integer, intent(in) rowCount,  
    integer, intent(in) colCount,  
    integer, intent(in) ovlp,  
    type(scheme), intent(in) sch )
```

The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- rowCount...
  1. is not positive
- colCount...

- 1. is not positive
- 2. is too small to be divided among the processors, ignoring the overlap.
- 3. is too small to be divided among the processors, including the overlap.
- *ovlp*...
  - 1. is negative
- *sch*...
  - 1. is already initialized

Definition at line 186 of file `create_scheme.f90`.

### 6.1.1.2 `create_scheme_sbr()`

```
subroutine create_scheme_sbr (
    integer, intent(in) rowCount,
    integer, intent(inout) colCount,
    double precision, intent(in) colSpc,
    double precision, intent(inout) colRef,
    integer, intent(in) comm,
    integer, intent(in) mpiDatatype,
    integer, intent(in) ovlp,
    type(scheme), intent(inout) sch )
```

The SCHEME creation subroutine.

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in	<i>rowCount</i>	Number of rows in the grid.
in, out	<i>colCount</i>	Number of columns in the grid. Returns the number of columns in the sub-grid, including overlap.
in	<i>colSpc</i>	The spacing between columns of the grid, for reference point identification.
in, out	<i>colRef</i>	The physical position of the reference point in the dimension along the rows (corresponding to a column).
in	<i>comm</i>	MPI communicator that host the processors that contain the sub-grids.
in	<i>mpiDatatype</i>	MPI datatype corresponding to the datatype of the grid.
in	<i>ovlp</i>	Number of extra columns needed by each sub-grid to successfully step forward in time.
in, out	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 25 of file `create_scheme.f90`.

## 6.2 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_fft.f90 File Reference

### Functions/Subroutines

- subroutine `create_scheme_fft_sbr` (sch)  
*The SCHEME FFT creation subroutine.*
- subroutine `create_scheme_fft_eh` (sch)  
*The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*
- subroutine `decompose` (nelems, nparts, pidx, nelemspart, sidx)  
*Creates a decomposition of a single dimension of the 2-D array.*
- subroutine `subarray` (sizes, axis, nparts, datatype, subarrays)  
*Creates the subarray datatypes for the local array.*

### 6.2.1 Function/Subroutine Documentation

#### 6.2.1.1 `create_scheme_fft_eh()`

```
subroutine create_scheme_fft_sbr::create_scheme_fft_eh (  
    type(scheme), intent(in) sch )
```

The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  1. is not initialized
  2. is already initialized for FFTs

Definition at line 69 of file create\_scheme\_fft.f90.

#### 6.2.1.2 `create_scheme_fft_sbr()`

```
subroutine create_scheme_fft_sbr (  
    type(scheme), intent(inout) sch )
```

The SCHEME FFT creation subroutine.

#### Author

Jason Turner, University of Wisconsin-Madison

**Parameters**

<i>in, out</i>	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.
----------------	------------	--

Definition at line 11 of file create\_scheme\_fft.f90.

**6.2.1.3 decompose()**

```
subroutine decompose (
    integer, intent(in) nelems,
    integer, intent(in) nparts,
    integer, intent(in) pidx,
    integer, intent(out) nelemspart,
    integer, intent(out) sidx )
```

Creates a decomposition of a single dimension of the 2-D array.

**Parameters**

[IN]	nelems Number of elements along the dimension of the array, $\geq 0$ .
[IN]	nparts Number of parts to divide dimension into, $> 0$ .
[IN]	pidx Part index, $\geq 0$ and $< nparts$ .
[O↔ UT]	nelemspart Number of elements in part.
[O↔ UT]	sidx Start index of part.

Definition at line 115 of file create\_scheme\_fft.f90.

**6.2.1.4 subarray()**

```
subroutine subarray (
    integer, dimension(0:1), intent(in) sizes,
    integer, intent(in) axis,
    integer, intent(in) nparts,
    integer, intent(in) datatype,
    integer, dimension(0:nparts-1), intent(out) subarrays )
```

Creates the subarray datatypes for the local array.

**Parameters**

[IN]	sizes Local sizes of array.
[IN]	axis Axis to partition, $0 \leq \text{axis} < 2$ .
[IN]	nparts Number of parts, $nparts > 0$ .
[IN]	datatype MPI datatype descriptor.
[O↔ UT]	subarrays Subarray datatype descriptors.
[O↔ UT]	



Definition at line 146 of file create\_scheme\_fft.f90.

## 6.3 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_spec\_drv.f90 File Reference

### Functions/Subroutines

- subroutine [create\\_scheme\\_spec\\_drv\\_sbr](#) (sch)  
*The SCHEME zero-padding initialization subroutine (DOUBLE PRECISION).*
- subroutine [create\\_scheme\\_spec\\_drv\\_eh](#) (sch)  
*The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

### 6.3.1 Function/Subroutine Documentation

#### 6.3.1.1 create\_scheme\_spec\_drv\_eh()

```
subroutine create_scheme_spec_drv_sbr::create_scheme_spec_drv_eh (  
    type(scheme), intent(in) sch )
```

The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  1. is not initialized
  2. is not initialized for FFTs
  3. is already initialized for spectral derivatives

Definition at line 86 of file create\_scheme\_spec\_drv.f90.

#### 6.3.1.2 create\_scheme\_spec\_drv\_sbr()

```
subroutine create_scheme_spec_drv_sbr (  
    type(scheme), intent(inout) sch )
```

The SCHEME zero-padding initialization subroutine (DOUBLE PRECISION).

#### Author

Jason Turner, University of Wisconsin-Madison

## Parameters

<code>in, out</code>	<code>sch</code>	SCHEME that holds information for grid decomposition, etc.
----------------------	------------------	--

Definition at line 11 of file `create_scheme_spec_drv.f90`.

## 6.4 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_P← ARALLEL/create\_scheme\_zero\_pad.f90 File Reference

### Functions/Subroutines

- subroutine [create\\_scheme\\_zero\\_pad\\_dble\\_sbr](#) (`sch`, `schZP`)  
*The SCHEME zero-padding initialization subroutine (DOUBLE PRECISION).*
- subroutine [create\\_scheme\\_zero\\_pad\\_dble\\_eh](#) (`sch`, `schZP`)  
*The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*
- subroutine [decompose](#) (`nelems`, `nparts`, `pid`, `nelemspart`, `sid`)  
*Creates a decomposition of a single dimension of the 2-D array.*
- subroutine [subarray](#) (`sizes`, `axis`, `nparts`, `datatype`, `subarrays`)  
*Creates the subarray datatypes for the local array.*

### 6.4.1 Function/Subroutine Documentation

#### 6.4.1.1 create\_scheme\_zero\_pad\_dble\_eh()

```
subroutine create_scheme_zero_pad_dble_sbr::create_scheme_zero_pad_dble_eh (
    type(scheme), intent(in) sch,
    type(scheme), intent(in) schZP )
```

The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- `sch...`
  - is not initialized
  - is not initialized for FFTs
  - is not initialized for spectral derivatives
- `schZP...`
  - is already initialized

Definition at line 217 of file `create_scheme_zero_pad.f90`.

### 6.4.1.2 create\_scheme\_zero\_pad\_dble\_sbr()

```
subroutine create_scheme_zero_pad_dble_sbr (
    type(scheme), intent(in) sch,
    type(scheme), intent(inout) schZP )
```

The SCHEME zero-padding initialization subroutine (DOUBLE PRECISION).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in	<i>sch</i>	SCHEME that is used for arr.
in, out	<i>schZP</i>	SCHEME that is made for arrZP.

Definition at line 11 of file create\_scheme\_zero\_pad.f90.

### 6.4.1.3 decompose()

```
subroutine create_scheme_zero_pad_dble_sbr::decompose (
    integer, intent(in) nelems,
    integer, intent(in) nparts,
    integer, intent(in) pidx,
    integer, intent(out) nelemspart,
    integer, intent(out) sidx )
```

Creates a decomposition of a single dimension of the 2-D array.

#### Parameters

[IN]	nelems Number of elements along the dimension of the array, >=0.
[IN]	nparts Number of parts to divide dimension into, >0.
[IN]	pidx Part index, >=0 and <nparts.
[O↔ UT]	nelemspart Number of elements in part.
[O↔ UT]	sidx Start index of part.

Definition at line 281 of file create\_scheme\_zero\_pad.f90.

### 6.4.1.4 subarray()

```
subroutine create_scheme_zero_pad_dble_sbr::subarray (
    integer, dimension(0:1), intent(in) sizes,
```

```

integer, intent(in) axis,
integer, intent(in) nparts,
integer, intent(in) datatype,
integer, dimension(0:nparts-1), intent(out) subarrays )

```

Creates the subarray datatypes for the local array.

#### Parameters

[IN]	sizes Local sizes of array.
[IN]	axis Axis to partition, $0 \leq \text{axis} < 2$ .
[IN]	nparts Number of parts, $\text{nparts} > 0$ .
[IN]	datatype MPI datatype descriptor.
[OUT]	subarrays Subarray datatype descriptors.

Definition at line 312 of file create\_scheme\_zero\_pad.f90.

## 6.5 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/destroy\_scheme.f90 File Reference

### Functions/Subroutines

- subroutine [destroy\\_scheme\\_sbr](#) (sch)  
*The SCHEME creation subroutine.*
- subroutine [destroy\\_scheme\\_eh](#) (sch)  
*The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

#### 6.5.1 Function/Subroutine Documentation

##### 6.5.1.1 destroy\_scheme\_eh()

```

subroutine destroy_scheme_sbr::destroy_scheme_eh (
    type(scheme), intent(in) sch )

```

The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- 

1. is not initialized

Definition at line 91 of file destroy\_scheme.f90.

### 6.5.1.2 destroy\_scheme\_sbr()

```
subroutine destroy_scheme_sbr (
    type(scheme), intent(inout) sch )
```

The SCHEME creation subroutine.

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	sch	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.
---------	-----	--

Definition at line 11 of file destroy\_scheme.f90.

## 6.6 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/execute\_scheme\_fft.f90 File Reference

### Functions/Subroutines

- subroutine [execute\\_scheme\\_fft\\_dble\\_sbr](#) (subGrid, kind, sch)  
*The FFT execution subroutine (DOUBLE PRECISION).*
- subroutine [execute\\_scheme\\_fft\\_dble\\_eh](#) (kind, sch)  
*The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*
- subroutine [execute\\_scheme\\_fft\\_dcmpx\\_sbr](#) (subGrid, kind, sch)  
*The FFT execution subroutine (DOUBLE COMPLEX).*
- subroutine [execute\\_scheme\\_fft\\_dcmpx\\_eh](#) (kind, sch)  
*The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

### 6.6.1 Function/Subroutine Documentation

#### 6.6.1.1 execute\_scheme\_fft\_dble\_eh()

```
subroutine execute_scheme_fft_dble_sbr::execute_scheme_fft_dble_eh (
    integer, intent(in) kind,
    type(scheme), intent(in) sch )
```

The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- kind...
  1.
    - not supported
- sch...
  1. is not initialized
  2. is not initialized for FFTs

Definition at line 122 of file `execute_scheme_fft.f90`.

### 6.6.1.2 `execute_scheme_fft_dble_sbr()`

```
subroutine execute_scheme_fft_dble_sbr (
    double precision, dimension(0:,0:), intent(inout) subGrid,
    integer, intent(in) kind,
    type(scheme), intent(in) sch )
```

The FFT execution subroutine (DOUBLE PRECISION).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of FFT to execute, one of FFT_1D_1, FFT_1D_2, or FFT_2D.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 13 of file `execute_scheme_fft.f90`.

### 6.6.1.3 `execute_scheme_fft_dcmpx_eh()`

```
subroutine execute_scheme_fft_dcmpx_sbr::execute_scheme_fft_dcmpx_eh (
    integer, intent(in) kind,
    type(scheme), intent(in) sch )
```

The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- kind...
  1.
    - not supported

- sch...
  1. is not initialized
  2. is not initialized for FFTs

Definition at line 292 of file execute\_scheme\_fft.f90.

#### 6.6.1.4 execute\_scheme\_fft\_dcmpx\_sbr()

```
subroutine execute_scheme_fft_dcmpx_sbr (
    double complex, dimension(0:,0:), intent(inout) subGrid,
    integer, intent(in) kind,
    type(scheme), intent(in) sch )
```

The FFT execution subroutine (DOUBLE COMPLEX).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of FFT to execute, one of FFT_1D_1, FFT_1D_2, or FFT_2D.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 183 of file execute\_scheme\_fft.f90.

## 6.7 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/execute\_scheme\_ifft.f90 File Reference

### Functions/Subroutines

- subroutine [execute\\_scheme\\_ifft\\_dble\\_sbr](#) (subGrid, kind, sch)  
*The inverse FFT execution subroutine (DOUBLE PRECISION).*
- subroutine [execute\\_scheme\\_ifft\\_dble\\_eh](#) (kind, sch)  
*The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*
- subroutine [execute\\_scheme\\_ifft\\_dcmpx\\_sbr](#) (subGrid, kind, sch)  
*The inverse FFT execution subroutine (DOUBLE COMPLEX).*
- subroutine [execute\\_scheme\\_ifft\\_dcmpx\\_eh](#) (kind, sch)  
*The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

## 6.7.1 Function/Subroutine Documentation

### 6.7.1.1 `execute_scheme_ifft_dble_eh()`

```
subroutine execute_scheme_ifft_dble_sbr::execute_scheme_ifft_dble_eh (
    integer, intent(in) kind,
    type(scheme), intent(in) sch )
```

The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- kind...
  1. not supported
- sch...
  1. is not initialized
  2. is not initialized for FFTs

Definition at line 128 of file `execute_scheme_ifft.f90`.

### 6.7.1.2 `execute_scheme_ifft_dble_sbr()`

```
subroutine execute_scheme_ifft_dble_sbr (
    double precision, dimension(0:,0:), intent(inout) subGrid,
    integer, intent(in) kind,
    type(scheme), intent(in) sch )
```

The inverse FFT execution subroutine (DOUBLE PRECISION).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid.
in	<i>kind</i>	Type of FFT to execute, one of FFT_1D_1, FFT_1D_2, or FFT_2D.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 13 of file `execute_scheme_ifft.f90`.



### 6.7.1.3 execute\_scheme\_ifft\_dcmpx\_eh()

```
subroutine execute_scheme_ifft_dcmpx_sbr::execute_scheme_ifft_dcmpx_eh (  
    integer, intent(in) kind,  
    type(scheme), intent(in) sch )
```

The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- kind...
  1.  
not supported
- sch...
  1. is not initialized
  2. is not initialized for FFTs

Definition at line 304 of file execute\_scheme\_ifft.f90.

### 6.7.1.4 execute\_scheme\_ifft\_dcmpx\_sbr()

```
subroutine execute_scheme_ifft_dcmpx_sbr (  
    double complex, dimension(0:,0:), intent(inout) subGrid,  
    integer, intent(in) kind,  
    type(scheme), intent(in) sch )
```

The inverse FFT execution subroutine (DOUBLE COMPLEX).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of FFT to execute, one of FFT_1D_1, FFT_1D_2, or FFT_2D.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 189 of file execute\_scheme\_ifft.f90.

## 6.8 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/execute\_scheme\_ispec\_drv.f90 File Reference

### Functions/Subroutines

- subroutine [execute\\_scheme\\_ispec\\_drv\\_dble\\_sbr](#) (subGrid, kind, order, sch)  
*The inverse spectral derivative execution subroutine (DOUBLE PRECISION).*
- subroutine [execute\\_scheme\\_ispec\\_drv\\_dble\\_eh](#) (kind, order, sch)  
*The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*
- subroutine [execute\\_scheme\\_ispec\\_drv\\_dcmpx\\_sbr](#) (subGrid, kind, order, sch)  
*The spectral derivative execution subroutine (DOUBLE COMPLEX).*
- subroutine [execute\\_scheme\\_ispec\\_drv\\_dcmpx\\_eh](#) (kind, order, sch)  
*The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

### 6.8.1 Function/Subroutine Documentation

#### 6.8.1.1 [execute\\_scheme\\_ispec\\_drv\\_dble\\_eh\(\)](#)

```
subroutine execute_scheme_ispec_drv_dble_sbr::execute_scheme_ispec_drv_dble_eh (
    integer, intent(in) kind,
    integer, intent(in) order,
    type(scheme), intent(in) sch )
```

The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- kind...
  1. not supported
- order...
  1. negative
- sch...
  1. is not initialized
  2. is not initialized for FFTs
  3. is not initialized for spectral derivatives

Definition at line 63 of file `execute_scheme_ispec_drv.f90`.

### 6.8.1.2 execute\_scheme\_ispec\_drv\_dble\_sbr()

```
subroutine execute_scheme_ispec_drv_dble_sbr (  
    double precision, dimension(0:,0:), intent(inout) subGrid,  
    integer, intent(in) kind,  
    integer, intent(in) order,  
    type(scheme), intent(in) sch )
```

The inverse spectral derivative execution subroutine (DOUBLE PRECISION).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of spectral derivative to execute, one of SPEC_DRV_1D_1, SPEC_DRV_1D_2.
in	<i>order</i>	Order of the spectral derivative.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 15 of file execute\_scheme\_ispec\_drv.f90.

### 6.8.1.3 execute\_scheme\_ispec\_drv\_dcmpx\_eh()

```
subroutine execute_scheme_ispec_drv_dcmpx_sbr::execute_scheme_ispec_drv_dcmpx_eh (  
    integer, intent(in) kind,  
    integer, intent(in) order,  
    type(scheme), intent(in) sch )
```

The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- kind...
  1.  
not supported
- order...
  1.  
negative
- sch...
  1. is not initialized
  2. is not initialized for FFTs
  3. is not initialized for spectral derivatives

Definition at line 251 of file execute\_scheme\_ispec\_drv.f90.

#### 6.8.1.4 execute\_scheme\_ispec\_drv\_dcmpx\_sbr()

```
subroutine execute_scheme_ispec_drv_dcmpx_sbr (
    double complex, dimension(0:,0:), intent(inout) subGrid,
    integer, intent(in) kind,
    integer, intent(in) order,
    type(scheme), intent(in) sch )
```

The spectral derivative execution subroutine (DOUBLE COMPLEX).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of spectral derivative to execute, one of SPEC_DRV_1D_1, SPEC_DRV_1D_2.
in	<i>order</i>	Order of the spectral derivative.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 142 of file execute\_scheme\_ispec\_drv.f90.

## 6.9 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/execute\_scheme\_izero\_pad.f90 File Reference

### Functions/Subroutines

- subroutine [execute\\_scheme\\_izero\\_pad\\_dble\\_sbr](#) (arr, sch, arrZP, schZP)  
*The SCHEME zero-padding execution subroutine (DOUBLE PRECISION).*
- subroutine [execute\\_scheme\\_izero\\_pad\\_dble\\_eh](#) (sch, schZP)  
*The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*
- subroutine [decompose](#) (nelems, nparts, pidx, nelemspart, sidx)  
*Creates a decomposition of a single dimension of the 2-D array.*
- subroutine [subarray](#) (sizes, axis, nparts, datatype, subarrays)  
*Creates the subarray datatypes for the local array.*
- subroutine [execute\\_scheme\\_izero\\_pad\\_dcmpx\\_sbr](#) (arr, sch, arrZP, schZP)  
*The SCHEME zero-padding execution subroutine (DOUBLE COMPLEX).*
- subroutine [execute\\_scheme\\_izero\\_pad\\_dcmpx\\_eh](#) (sch, schZP)  
*The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

### 6.9.1 Function/Subroutine Documentation

### 6.9.1.1 decompose()

```
subroutine execute_scheme_izero_pad_dble_sbr::decompose (
    integer, intent(in) nelems,
    integer, intent(in) nparts,
    integer, intent(in) pidx,
    integer, intent(out) nelemspart,
    integer, intent(out) sidx )
```

Creates a decomposition of a single dimension of the 2-D array.

#### Parameters

[IN]	nelems Number of elements along the dimension of the array, >=0.
[IN]	nparts Number of parts to divide dimension into, >0.
[IN]	pidx Part index, >=0 and <nparts.
[O↔ UT]	nelemspart Number of elements in part.
[O↔ UT]	sidx Start index of part.

Definition at line 192 of file execute\_scheme\_izero\_pad.f90.

### 6.9.1.2 execute\_scheme\_izero\_pad\_dble\_eh()

```
subroutine execute_scheme_izero_pad_dble_sbr::execute_scheme_izero_pad_dble_eh (
    type(scheme), intent(in) sch,
    type(scheme), intent(in) schZP )
```

The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  1. is not initialized
  2. is not initialized for FFTs
  3. is not initialized for spectral derivatives
- schZP...
  1. is not initialized

Definition at line 128 of file execute\_scheme\_izero\_pad.f90.

### 6.9.1.3 execute\_scheme\_izero\_pad\_dble\_sbr()

```
subroutine execute_scheme_izero_pad_dble_sbr (
    double precision, dimension(0:,0:), intent(inout) arr,
    type(scheme), intent(in) sch,
    double precision, dimension(0:,0:), intent(in) arrZP,
    type(scheme), intent(in) schZP )
```

The SCHEME zero-padding execution subroutine (DOUBLE PRECISION).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in	<i>arr</i>	Sub-grid that will be zero padded.
in	<i>sch</i>	SCHEME that is used for arr.
in, out	<i>arrZP</i>	Zero-padded sub-grid.
in, out	<i>schZP</i>	SCHEME that is made for arrZP.

Definition at line 13 of file execute\_scheme\_izero\_pad.f90.

### 6.9.1.4 execute\_scheme\_izero\_pad\_dcmpx\_eh()

```
subroutine execute_scheme_izero_pad_dcmpx_sbr::execute_scheme_izero_pad_dcmpx_eh (
    type(scheme), intent(in) sch,
    type(scheme), intent(in) schZP )
```

The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  1. is not initialized
  2. is not initialized for FFTs
  3. is not initialized for spectral derivatives
- schZP...
  1. is not initialized

Definition at line 382 of file execute\_scheme\_izero\_pad.f90.

### 6.9.1.5 execute\_scheme\_izero\_pad\_dcmpx\_sbr()

```
subroutine execute_scheme_izero_pad_dcmpx_sbr (
    double complex, dimension(0:,0:), intent(inout) arr,
    type(scheme), intent(in) sch,
    double complex, dimension(0:,0:), intent(in) arrZP,
    type(scheme), intent(in) schZP )
```

The SCHEME zero-padding execution subroutine (DOUBLE COMPLEX).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in	<i>arr</i>	Sub-grid that will be zero padded.
in	<i>sch</i>	SCHEME that is used for arr.
in, out	<i>arrZP</i>	Zero-padded sub-grid.
in, out	<i>schZP</i>	SCHEME that is made for arrZP.

Definition at line 267 of file execute\_scheme\_izero\_pad.f90.

### 6.9.1.6 subarray()

```
subroutine execute_scheme_izero_pad_dble_sbr::subarray (
    integer, dimension(0:1), intent(in) sizes,
    integer, intent(in) axis,
    integer, intent(in) nparts,
    integer, intent(in) datatype,
    integer, dimension(0:nparts-1), intent(out) subarrays )
```

Creates the subarray datatypes for the local array.

#### Parameters

[IN]	sizes Local sizes of array.
[IN]	axis Axis to partition, 0 <= axis < 2.
[IN]	nparts Number of parts, nparts > 0.
[IN]	datatype MPI datatype descriptor.
[OUT]	subarrays Subarray datatype descriptors.

Definition at line 223 of file execute\_scheme\_izero\_pad.f90.

## 6.10 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_↵ PARALLEL/execute\_scheme\_spec\_drv.f90 File Reference

### Functions/Subroutines

- subroutine [execute\\_scheme\\_spec\\_drv\\_dble\\_sbr](#) (subGrid, kind, order, sch)  
*The spectral derivative execution subroutine (DOUBLE PRECISION).*
- subroutine [execute\\_scheme\\_spec\\_drv\\_dble\\_eh](#) (kind, order, sch)  
*The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*
- subroutine [execute\\_scheme\\_spec\\_drv\\_dcmpx\\_sbr](#) (subGrid, kind, order, sch)  
*The spectral derivative execution subroutine (DOUBLE COMPLEX).*
- subroutine [execute\\_scheme\\_spec\\_drv\\_dcmpx\\_eh](#) (kind, order, sch)  
*The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

### 6.10.1 Function/Subroutine Documentation

#### 6.10.1.1 [execute\\_scheme\\_spec\\_drv\\_dble\\_eh\(\)](#)

```
subroutine execute_scheme_spec_drv_dble_sbr::execute_scheme_spec_drv_dble_eh (
    integer, intent(in) kind,
    integer, intent(in) order,
    type(scheme), intent(in) sch )
```

The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- kind...
  1. not supported
- order...
  1. negative
- sch...
  1. is not initialized
  2. is not initialized for FFTs
  3. is not initialized for spectral derivatives

Definition at line 63 of file [execute\\_scheme\\_spec\\_drv.f90](#).



### 6.10.1.2 execute\_scheme\_spec\_drv\_dble\_sbr()

```
subroutine execute_scheme_spec_drv_dble_sbr (  
    double precision, dimension(0:,0:), intent(inout) subGrid,  
    integer, intent(in) kind,  
    integer, intent(in) order,  
    type(scheme), intent(in) sch )
```

The spectral derivative execution subroutine (DOUBLE PRECISION).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of spectral derivative to execute, one of SPEC_DRV_1D_1, SPEC_DRV_1D_2.
in	<i>order</i>	Order of the spectral derivative.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 15 of file execute\_scheme\_spec\_drv.f90.

### 6.10.1.3 execute\_scheme\_spec\_drv\_dcmpx\_eh()

```
subroutine execute_scheme_spec_drv_dcmpx_sbr::execute_scheme_spec_drv_dcmpx_eh (  
    integer, intent(in) kind,  
    integer, intent(in) order,  
    type(scheme), intent(in) sch )
```

The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- kind...
  1.  
not supported
- order...
  1.  
negative
- sch...
  1. is not initialized
  2. is not initialized for FFTs
  3. is not initialized for spectral derivatives

Definition at line 235 of file execute\_scheme\_spec\_drv.f90.

#### 6.10.1.4 execute\_scheme\_spec\_drv\_dcmpx\_sbr()

```
subroutine execute_scheme_spec_drv_dcmpx_sbr (
    double complex, dimension(0:,0:), intent(inout) subGrid,
    integer, intent(in) kind,
    integer, intent(in) order,
    type(scheme), intent(in) sch )
```

The spectral derivative execution subroutine (DOUBLE COMPLEX).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>kind</i>	Type of spectral derivative to execute, one of SPEC_DRV_1D_1, SPEC_DRV_1D_2.
in	<i>order</i>	Order of the spectral derivative.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 142 of file execute\_scheme\_spec\_drv.f90.

## 6.11 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_↔ PARALLEL/execute\_scheme\_zero\_pad.f90 File Reference

### Functions/Subroutines

- subroutine [execute\\_scheme\\_zero\\_pad\\_dble\\_sbr](#) (arr, sch, arrZP, schZP)  
*The SCHEME zero-padding execution subroutine (DOUBLE PRECISION).*
- subroutine [execute\\_scheme\\_zero\\_pad\\_dble\\_eh](#) (sch, schZP)  
*The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*
- subroutine [decompose](#) (nelems, nparts, pidx, nelemspart, sidx)  
*Creates a decomposition of a single dimension of the 2-D array.*
- subroutine [subarray](#) (sizes, axis, nparts, datatype, subarrays)  
*Creates the subarray datatypes for the local array.*
- subroutine [execute\\_scheme\\_zero\\_pad\\_dcmpx\\_sbr](#) (arr, sch, arrZP, schZP)  
*The SCHEME zero-padding execution subroutine (DOUBLE COMPLEX).*
- subroutine [execute\\_scheme\\_zero\\_pad\\_dcmpx\\_eh](#) (sch, schZP)  
*The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

#### 6.11.1 Function/Subroutine Documentation

### 6.11.1.1 decompose()

```
subroutine execute_scheme_zero_pad_dble_sbr::decompose (
    integer, intent(in) nelems,
    integer, intent(in) nparts,
    integer, intent(in) pidx,
    integer, intent(out) nelemspart,
    integer, intent(out) sidx )
```

Creates a decomposition of a single dimension of the 2-D array.

#### Parameters

[IN]	nelems Number of elements along the dimension of the array, >=0.
[IN]	nparts Number of parts to divide dimension into, >0.
[IN]	pidx Part index, >=0 and <nparts.
[O↔ UT]	nelemspart Number of elements in part.
[O↔ UT]	sidx Start index of part.

Definition at line 187 of file execute\_scheme\_zero\_pad.f90.

### 6.11.1.2 execute\_scheme\_zero\_pad\_dble\_eh()

```
subroutine execute_scheme_zero_pad_dble_sbr::execute_scheme_zero_pad_dble_eh (
    type(scheme), intent(in) sch,
    type(scheme), intent(in) schZP )
```

The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  1. is not initialized
  2. is not initialized for FFTs
  3. is not initialized for spectral derivatives
- schZP...
  1. is not initialized

Definition at line 123 of file execute\_scheme\_zero\_pad.f90.

### 6.11.1.3 execute\_scheme\_zero\_pad\_dble\_sbr()

```
subroutine execute_scheme_zero_pad_dble_sbr (
    double precision, dimension(0:,0:), intent(in) arr,
    type(scheme), intent(in) sch,
    double precision, dimension(:,,:), intent(inout), allocatable arrZP,
    type(scheme), intent(in) schZP )
```

The SCHEME zero-padding execution subroutine (DOUBLE PRECISION).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in	<i>arr</i>	Sub-grid that will be zero padded.
in	<i>sch</i>	SCHEME that is used for arr.
in, out	<i>arrZP</i>	Zero-padded sub-grid.
in, out	<i>schZP</i>	SCHEME that is made for arrZP.

Definition at line 13 of file execute\_scheme\_zero\_pad.f90.

### 6.11.1.4 execute\_scheme\_zero\_pad\_dcmpx\_eh()

```
subroutine execute_scheme_zero_pad_dcmpx_sbr::execute_scheme_zero_pad_dcmpx_eh (
    type(scheme), intent(in) sch,
    type(scheme), intent(in) schZP )
```

The error handling subroutine for the SCHEME creation subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  1. is not initialized
  2. is not initialized for FFTs
  3. is not initialized for spectral derivatives
- schZP...
  1. is not initialized

Definition at line 372 of file execute\_scheme\_zero\_pad.f90.

### 6.11.1.5 execute\_scheme\_zero\_pad\_dcmpx\_sbr()

```
subroutine execute_scheme_zero_pad_dcmpx_sbr (
    double complex, dimension(0:,0:), intent(in) arr,
    type(scheme), intent(in) sch,
    double complex, dimension(:,,:), intent(inout), allocatable arrZP,
    type(scheme), intent(in) schZP )
```

The SCHEME zero-padding execution subroutine (DOUBLE COMPLEX).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in	<i>arr</i>	Sub-grid that will be zero padded.
in	<i>sch</i>	SCHEME that is used for arr.
in, out	<i>arrZP</i>	Zero-padded sub-grid.
in, out	<i>schZP</i>	SCHEME that is made for arrZP.

Definition at line 262 of file execute\_scheme\_zero\_pad.f90.

### 6.11.1.6 subarray()

```
subroutine execute_scheme_zero_pad_dble_sbr::subarray (
    integer, dimension(0:1), intent(in) sizes,
    integer, intent(in) axis,
    integer, intent(in) nparts,
    integer, intent(in) datatype,
    integer, dimension(0:nparts-1), intent(out) subarrays )
```

Creates the subarray datatypes for the local array.

#### Parameters

[IN]	sizes Local sizes of array.
[IN]	axis Axis to partition, 0 <= axis < 2.
[IN]	nparts Number of parts, nparts > 0.
[IN]	datatype MPI datatype descriptor.
[OUT]	subarrays Subarray datatype descriptors.

Definition at line 218 of file execute\_scheme\_zero\_pad.f90.

## 6.12 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/ez\_parallel.f90 File Reference

### Data Types

- interface [ez\\_parallel::create\\_scheme](#)  
*The interface for the `SCHEME` creation subroutine.*
- interface [ez\\_parallel::create\\_scheme\\_fft](#)  
*The interface for the `SCHEME` FFT initialization subroutine.*
- interface [ez\\_parallel::create\\_scheme\\_spec\\_drv](#)  
*The interface for the `SCHEME` spectral derivative initialization subroutine.*
- interface [ez\\_parallel::create\\_scheme\\_zero\\_pad](#)  
*The interface for the `SCHEME` zero-padding initializtion subroutine.*
- interface [ez\\_parallel::destroy\\_scheme](#)  
*The interface for the `SCHEME` destruction subroutine.*
- interface [ez\\_parallel::share\\_subgrid\\_bdry](#)  
*The sub-grid boundary communication interface for the `EZ_PARALLEL` module.*
- interface [ez\\_parallel::execute\\_scheme\\_fft](#)  
*The FFT execution interface for the `EZ_PARALLEL` module.*
- interface [ez\\_parallel::execute\\_scheme\\_iff](#)  
*The inverse FFT execution interface for the `EZ_PARALLEL` module.*
- interface [ez\\_parallel::execute\\_scheme\\_spec\\_drv](#)  
*The spectral derivative execution interface for the `EZ_PARALLEL` module.*
- interface [ez\\_parallel::execute\\_scheme\\_ispec\\_drv](#)  
*The inverse spectral derivative execution interface for the `EZ_PARALLEL` module.*
- interface [ez\\_parallel::execute\\_scheme\\_zero\\_pad](#)  
*The interface for the `SCHEME` zero-padding execution subroutine.*
- interface [ez\\_parallel::execute\\_scheme\\_izero\\_pad](#)  
*The interface for the `SCHEME` zero-padding removal subroutine.*
- interface [ez\\_parallel::max\\_val](#)  
*The maximum value interface for the `EZ_PARALLEL` module.*
- interface [ez\\_parallel::min\\_val](#)  
*The minimum value interface for the `EZ_PARALLEL` module.*

### Modules

- module [ez\\_parallel](#)  
*The `EZ_PARALLEL` module. Contains `EZ_PARALLEL` subroutines and their interfaces.*

## 6.13 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_↔ PARALLEL/ez\_parallel\_structs.f90 File Reference

### Data Types

- type [ez\\_parallel\\_structs::scheme](#)

## Modules

- module [ez\\_parallel\\_structs](#)

*The EZ\_PARALLEL structures module. Contains all EZ\_PARALLEL derived datatypes.*

## Variables

- integer, parameter, public [ez\\_parallel\\_structs::fft\\_1d\\_1](#) = 51  
*Flag to mark execution of 1D FFTs along the first dimension.*
- integer, parameter, public [ez\\_parallel\\_structs::fft\\_1d\\_2](#) = 46  
*Flag to mark execution of 1D FFTs along the second dimension.*
- integer, parameter, public [ez\\_parallel\\_structs::fft\\_2d](#) = 95  
*Flag to mark execution of 2D FFTs.*
- integer, parameter, public [ez\\_parallel\\_structs::spec\\_drv\\_1d\\_1](#) = 23  
*Flag to mark execution of 1D spectral derivatives along the first dimension.*
- integer, parameter, public [ez\\_parallel\\_structs::spec\\_drv\\_1d\\_2](#) = 78  
*Flag to mark execution of 1D spectral derivatives along the second dimension.*

## 6.14 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/max\_val.f90 File Reference

### Functions/Subroutines

- subroutine [max\\_val\\_sbr](#) (subGrid, maxValue, sch)  
*The max value subroutine.*
- subroutine [max\\_val\\_eh](#) (sch)  
*The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

#### 6.14.1 Function/Subroutine Documentation

##### 6.14.1.1 max\_val\_eh()

```
subroutine max_val_sbr::max_val_eh (  
    type(scheme), intent(in) sch )
```

The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  1. is not initialized

Definition at line 52 of file max\_val.f90.

### 6.14.1.2 max\_val\_sbr()

```
subroutine max_val_sbr (
    double precision, dimension(0:,0:), intent(in) subGrid,
    double precision, intent(inout) maxVal,
    type(scheme), intent(in) sch )
```

The max value subroutine.

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in	<i>subGrid</i>	The local sub-grid.
in, out	<i>maxValue</i>	The variable to store the maximum value across all sub-grids.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 14 of file max\_val.f90.

## 6.15 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/min\_val.f90 File Reference

### Functions/Subroutines

- subroutine [min\\_val\\_sbr](#) (subGrid, minVal, sch)  
*The min value subroutine.*
- subroutine [min\\_val\\_eh](#) (sch)  
*The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

### 6.15.1 Function/Subroutine Documentation

#### 6.15.1.1 min\_val\_eh()

```
subroutine min_val_sbr::min_val_eh (
    type(scheme), intent(in) sch )
```

The error handling subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  - is not initialized

Definition at line 52 of file min\_val.f90.



### 6.15.1.2 min\_val\_sbr()

```
subroutine min_val_sbr (
    double precision, dimension(0:,0:), intent(in) subGrid,
    double precision, intent(inout) minValue,
    type(scheme), intent(in) sch )
```

The min value subroutine.

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in	<i>subGrid</i>	The local sub-grid.
in, out	<i>minValue</i>	The variable to store the minimum value across all sub-grids.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 14 of file min\_val.f90.

## 6.16 C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/share\_subgrid\_bdry.f90 File Reference

### Functions/Subroutines

- subroutine [share\\_subgrid\\_bdry\\_dble\\_sbr](#) (subGrid, sch)  
*The sub-grid boundary sharing subroutine (DOUBLE PRECISION).*
- subroutine [share\\_subgrid\\_bdry\\_dble\\_eh](#) (sch)  
*The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*
- subroutine [share\\_subgrid\\_bdry\\_dcmpx\\_sbr](#) (subGrid, sch)  
*The sub-grid boundary sharing subroutine (DOUBLE COMPLEX).*
- subroutine [share\\_subgrid\\_bdry\\_dcmpx\\_eh](#) (sch)  
*The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:*

### 6.16.1 Function/Subroutine Documentation

### 6.16.1.1 share\_subgrid\_bdry\_dble\_eh()

```
subroutine share_subgrid_bdry_dble_sbr::share_subgrid_bdry_dble_eh (
    type(scheme), intent(in) sch )
```

The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  1. is not initialized

Definition at line 107 of file share\_subgrid\_bdry.f90.

### 6.16.1.2 share\_subgrid\_bdry\_dble\_sbr()

```
subroutine share_subgrid_bdry_dble_sbr (
    double precision, dimension(:, :), intent(inout) subGrid,
    type(scheme), intent(in) sch )
```

The sub-grid boundary sharing subroutine (DOUBLE PRECISION).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 12 of file share\_subgrid\_bdry.f90.

### 6.16.1.3 share\_subgrid\_bdry\_dcmpx\_eh()

```
subroutine share_subgrid_bdry_dcmpx_sbr::share_subgrid_bdry_dcmpx_eh (
    type(scheme), intent(in) sch )
```

The error handling subroutine for the sub-grid boundary sharing subroutine. Aborts the program if any of the input parameters satisfy any of the following conditions:

- sch...
  1. is not initialized

Definition at line 246 of file share\_subgrid\_bdry.f90.

#### 6.16.1.4 share\_subgrid\_bdry\_dcmpx\_sbr()

```
subroutine share_subgrid_bdry_dcmpx_sbr (  
    double complex, dimension(:, :), intent(inout) subGrid,  
    type(scheme), intent(in) sch )
```

The sub-grid boundary sharing subroutine (DOUBLE COMPLEX).

#### Author

Jason Turner, University of Wisconsin-Madison

#### Parameters

in, out	<i>subGrid</i>	The local sub-grid whose boundary will be shared.
in	<i>sch</i>	SCHEME that will be used for the grid decomposition, parallel FFTs, etc.

Definition at line 151 of file share\_subgrid\_bdry.f90.



# Index

The, 11

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme.f90, 37

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_fft.f90, 39

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_spec\_drv.f90, 41

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_zero\_pad.f90, 42

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_ah.f90, 44

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_ah.f90, 45

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_fft\_ah.f90, 47

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_fft\_sbr.f90, 50

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_fft\_sbr.f90, 52

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_sbr.f90, 56

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_spec\_drv.f90, 58

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_spec\_drv\_ah.f90, 62

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_spec\_drv\_sbr.f90, 62

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_spec\_drv\_sbr.f90, 63

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_spec\_drv\_sbr.f90, 64

C:/Users/Owner/Education and Research/Graduate School/Parallelization Module Project/EZ\_PARALLEL\_project/EZ\_PARALLEL/create\_scheme\_spec\_drv\_sbr.f90, 65

coldcmpsizes

ez\_parallel\_structs::scheme, 28

coldcmpsizesovlp

ez\_parallel\_structs::scheme, 28

colref

ez\_parallel\_structs::scheme, 28

colspc

ez\_parallel\_structs::scheme, 28

comm

ez\_parallel\_structs::scheme, 28

commsize

ez\_parallel\_structs::scheme, 29

counts

ez\_parallel\_structs::scheme, 29

create\_scheme.f90

ez\_parallel\_structs::scheme, 29

create\_scheme\_ah.f90, 37

create\_scheme\_sbr, 38

create\_scheme\_ah

ez\_parallel\_structs::scheme, 29

create\_scheme.f90, 37

create\_scheme\_fft.f90

ez\_parallel\_structs::scheme, 29

create\_scheme\_fft\_ah.f90, 39

create\_scheme\_fft\_sbr, 39

decompose, 40

subarray, 40

ez\_parallel\_structs::scheme, 29

create\_scheme\_fft\_ah.f90, 39

create\_scheme\_fft\_sbr, 39

ez\_parallel::create\_scheme\_fft, 12

create\_scheme\_sbr

ez\_parallel\_structs::scheme, 29

create\_scheme\_spec\_drv.f90, 38

ez\_parallel::create\_scheme, 12

create\_scheme\_spec\_drv.f90

ez\_parallel\_structs::scheme, 29

create\_scheme\_spec\_drv\_ah.f90, 41

create\_scheme\_spec\_drv\_sbr, 41

create\_scheme\_spec\_drv\_ah

ez\_parallel\_structs::scheme, 29

create\_scheme\_spec\_drv.f90, 41

create\_scheme\_spec\_drv\_sbr

ez\_parallel\_structs::scheme, 29

create\_scheme\_spec\_drv.f90, 41

ez\_parallel::create\_scheme\_spec\_drv, 13

create\_scheme\_zero\_pad.f90

ez\_parallel\_structs::scheme, 29

create\_scheme\_zero\_pad\_dble.f90, 42

create\_scheme\_zero\_pad\_dble\_sbr, 42

decompose, 43

subarray, 43

ez\_parallel\_structs::scheme, 29

create\_scheme\_zero\_pad\_dble\_ah

ez\_parallel\_structs::scheme, 29

create\_scheme\_zero\_pad.f90, 42

create\_scheme\_spec\_drv.f90, 42

create\_scheme\_spec\_drv\_sbr

ez\_parallel\_structs::scheme, 29

create\_scheme\_spec\_drv.f90, 42

- ez\_parallel::create\_scheme\_zero\_pad, 14
- datatype
  - ez\_parallel\_structs::scheme, 29
- decompose
  - create\_scheme\_fft.f90, 40
  - create\_scheme\_zero\_pad.f90, 43
  - execute\_scheme\_izero\_pad.f90, 52
  - execute\_scheme\_zero\_pad.f90, 58
- destroy\_scheme.f90
  - destroy\_scheme\_eh, 44
  - destroy\_scheme\_sbr, 44
- destroy\_scheme\_eh
  - destroy\_scheme.f90, 44
- destroy\_scheme\_sbr
  - destroy\_scheme.f90, 44
  - ez\_parallel::destroy\_scheme, 15
- displs
  - ez\_parallel\_structs::scheme, 29
- execute\_scheme\_fft.f90
  - execute\_scheme\_fft\_dble\_eh, 45
  - execute\_scheme\_fft\_dble\_sbr, 46
  - execute\_scheme\_fft\_dcmpx\_eh, 46
  - execute\_scheme\_fft\_dcmpx\_sbr, 47
- execute\_scheme\_fft\_dble\_eh
  - execute\_scheme\_fft.f90, 45
- execute\_scheme\_fft\_dble\_sbr
  - execute\_scheme\_fft.f90, 46
  - ez\_parallel::execute\_scheme\_fft, 16
- execute\_scheme\_fft\_dcmpx\_eh
  - execute\_scheme\_fft.f90, 46
- execute\_scheme\_fft\_dcmpx\_sbr
  - execute\_scheme\_fft.f90, 47
  - ez\_parallel::execute\_scheme\_fft, 16
- execute\_scheme\_ifft.f90
  - execute\_scheme\_ifft\_dble\_eh, 48
  - execute\_scheme\_ifft\_dble\_sbr, 48
  - execute\_scheme\_ifft\_dcmpx\_eh, 48
  - execute\_scheme\_ifft\_dcmpx\_sbr, 49
- execute\_scheme\_ifft\_dble\_eh
  - execute\_scheme\_ifft.f90, 48
- execute\_scheme\_ifft\_dble\_sbr
  - execute\_scheme\_ifft.f90, 48
  - ez\_parallel::execute\_scheme\_ifft, 17
- execute\_scheme\_ifft\_dcmpx\_eh
  - execute\_scheme\_ifft.f90, 48
- execute\_scheme\_ifft\_dcmpx\_sbr
  - execute\_scheme\_ifft.f90, 49
  - ez\_parallel::execute\_scheme\_ifft, 17
- execute\_scheme\_ispec\_drv.f90
  - execute\_scheme\_ispec\_drv\_dble\_eh, 50
  - execute\_scheme\_ispec\_drv\_dble\_sbr, 50
  - execute\_scheme\_ispec\_drv\_dcmpx\_eh, 51
  - execute\_scheme\_ispec\_drv\_dcmpx\_sbr, 51
- execute\_scheme\_ispec\_drv\_dble\_eh
  - execute\_scheme\_ispec\_drv.f90, 50
- execute\_scheme\_ispec\_drv\_dble\_sbr
  - execute\_scheme\_ispec\_drv.f90, 50
- ez\_parallel::execute\_scheme\_ispec\_drv, 19
  - execute\_scheme\_ispec\_drv\_dcmpx\_eh
    - execute\_scheme\_ispec\_drv.f90, 51
  - execute\_scheme\_ispec\_drv\_dcmpx\_sbr
    - execute\_scheme\_ispec\_drv.f90, 51
  - ez\_parallel::execute\_scheme\_ispec\_drv, 20
  - execute\_scheme\_izero\_pad.f90
    - decompose, 52
    - execute\_scheme\_izero\_pad\_dble\_eh, 53
    - execute\_scheme\_izero\_pad\_dble\_sbr, 53
    - execute\_scheme\_izero\_pad\_dcmpx\_eh, 54
    - execute\_scheme\_izero\_pad\_dcmpx\_sbr, 54
    - subarray, 55
  - execute\_scheme\_izero\_pad\_dble\_eh
    - execute\_scheme\_izero\_pad.f90, 53
  - execute\_scheme\_izero\_pad\_dble\_sbr
    - execute\_scheme\_izero\_pad.f90, 53
    - ez\_parallel::execute\_scheme\_izero\_pad, 21
  - execute\_scheme\_izero\_pad\_dcmpx\_eh
    - execute\_scheme\_izero\_pad.f90, 54
  - execute\_scheme\_izero\_pad\_dcmpx\_sbr
    - execute\_scheme\_izero\_pad.f90, 54
    - ez\_parallel::execute\_scheme\_izero\_pad, 21
  - execute\_scheme\_spec\_drv.f90
    - execute\_scheme\_spec\_drv\_dble\_eh, 56
    - execute\_scheme\_spec\_drv\_dble\_sbr, 56
    - execute\_scheme\_spec\_drv\_dcmpx\_eh, 57
    - execute\_scheme\_spec\_drv\_dcmpx\_sbr, 57
  - execute\_scheme\_spec\_drv\_dble\_eh
    - execute\_scheme\_spec\_drv.f90, 56
  - execute\_scheme\_spec\_drv\_dble\_sbr
    - execute\_scheme\_spec\_drv.f90, 56
    - ez\_parallel::execute\_scheme\_spec\_drv, 22
  - execute\_scheme\_spec\_drv\_dcmpx\_eh
    - execute\_scheme\_spec\_drv.f90, 57
  - execute\_scheme\_spec\_drv\_dcmpx\_sbr
    - execute\_scheme\_spec\_drv.f90, 57
    - ez\_parallel::execute\_scheme\_spec\_drv, 23
  - execute\_scheme\_zero\_pad.f90
    - decompose, 58
    - execute\_scheme\_zero\_pad\_dble\_eh, 59
    - execute\_scheme\_zero\_pad\_dble\_sbr, 59
    - execute\_scheme\_zero\_pad\_dcmpx\_eh, 60
    - execute\_scheme\_zero\_pad\_dcmpx\_sbr, 60
    - subarray, 61
  - execute\_scheme\_zero\_pad\_dble\_eh
    - execute\_scheme\_zero\_pad.f90, 59
  - execute\_scheme\_zero\_pad\_dble\_sbr
    - execute\_scheme\_zero\_pad.f90, 59
    - ez\_parallel::execute\_scheme\_zero\_pad, 24
  - execute\_scheme\_zero\_pad\_dcmpx\_eh
    - execute\_scheme\_zero\_pad.f90, 60
  - execute\_scheme\_zero\_pad\_dcmpx\_sbr
    - execute\_scheme\_zero\_pad.f90, 60
    - ez\_parallel::execute\_scheme\_zero\_pad, 24
- ez\_parallel, 7
  - ez\_parallel::create\_scheme, 11
  - create\_scheme\_sbr, 12

- ez\_parallel::create\_scheme\_fft, 12
  - create\_scheme\_fft\_sbr, 12
- ez\_parallel::create\_scheme\_spec\_drv, 13
  - create\_scheme\_spec\_drv\_sbr, 13
- ez\_parallel::create\_scheme\_zero\_pad, 14
  - create\_scheme\_zero\_pad\_dble\_sbr, 14
- ez\_parallel::destroy\_scheme, 14
  - destroy\_scheme\_sbr, 15
- ez\_parallel::execute\_scheme\_fft, 15
  - execute\_scheme\_fft\_dble\_sbr, 16
  - execute\_scheme\_fft\_dcpx\_sbr, 16
- ez\_parallel::execute\_scheme\_ifft, 17
  - execute\_scheme\_ifft\_dble\_sbr, 17
  - execute\_scheme\_ifft\_dcpx\_sbr, 17
- ez\_parallel::execute\_scheme\_ispec\_drv, 19
  - execute\_scheme\_ispec\_drv\_dble\_sbr, 19
  - execute\_scheme\_ispec\_drv\_dcpx\_sbr, 20
- ez\_parallel::execute\_scheme\_izero\_pad, 20
  - execute\_scheme\_izero\_pad\_dble\_sbr, 21
  - execute\_scheme\_izero\_pad\_dcpx\_sbr, 21
- ez\_parallel::execute\_scheme\_spec\_drv, 22
  - execute\_scheme\_spec\_drv\_dble\_sbr, 22
  - execute\_scheme\_spec\_drv\_dcpx\_sbr, 23
- ez\_parallel::execute\_scheme\_zero\_pad, 23
  - execute\_scheme\_zero\_pad\_dble\_sbr, 24
  - execute\_scheme\_zero\_pad\_dcpx\_sbr, 24
- ez\_parallel::max\_val, 24
  - max\_val\_sbr, 25
- ez\_parallel::min\_val, 25
  - min\_val\_sbr, 26
- ez\_parallel::share\_subgrid\_bdry, 34
  - share\_subgrid\_bdry\_dble\_sbr, 34
  - share\_subgrid\_bdry\_dcpx\_sbr, 35
- ez\_parallel\_structs, 8
  - fft\_1d\_1, 8
  - fft\_1d\_2, 9
  - fft\_2d, 9
  - spec\_drv\_1d\_1, 9
  - spec\_drv\_1d\_2, 9
- ez\_parallel\_structs::scheme, 26
  - coldcmpsizes, 28
  - coldcmpsizesovlp, 28
  - colref, 28
  - colspc, 28
  - comm, 28
  - commsize, 29
  - counts, 29
  - datatype, 29
  - displs, 29
  - gridsize, 29
  - hslabsize, 30
  - initfft, 30
  - initscheme, 30
  - initspecdrv, 30
  - norm\_1d\_1, 30
  - norm\_1d\_2, 31
  - norm\_2d, 31
  - ovlp, 31
  - procid, 31
  - recv\_boundaries, 31
  - rowdcmpsizes, 32
  - send\_boundaries, 32
  - subarrays, 32
  - vslabint, 32
  - vslabsize, 32
  - vslabsizeovlp, 33
  - wsave1, 33
  - wsave2, 33
  - wvnmb1, 33
  - wvnmb2, 33
- fft\_1d\_1
  - ez\_parallel\_structs, 8
- fft\_1d\_2
  - ez\_parallel\_structs, 9
- fft\_2d
  - ez\_parallel\_structs, 9
- gridsize
  - ez\_parallel\_structs::scheme, 29
- hslabsize
  - ez\_parallel\_structs::scheme, 30
- initfft
  - ez\_parallel\_structs::scheme, 30
- initscheme
  - ez\_parallel\_structs::scheme, 30
- initspecdrv
  - ez\_parallel\_structs::scheme, 30
- max\_val.f90
  - max\_val\_eh, 63
  - max\_val\_sbr, 63
- max\_val\_eh
  - max\_val.f90, 63
- max\_val\_sbr
  - ez\_parallel::max\_val, 25
  - max\_val.f90, 63
- min\_val.f90
  - min\_val\_eh, 64
  - min\_val\_sbr, 64
- min\_val\_eh
  - min\_val.f90, 64
- min\_val\_sbr
  - ez\_parallel::min\_val, 26
  - min\_val.f90, 64
- norm\_1d\_1
  - ez\_parallel\_structs::scheme, 30
- norm\_1d\_2
  - ez\_parallel\_structs::scheme, 31
- norm\_2d
  - ez\_parallel\_structs::scheme, 31
- ovlp
  - ez\_parallel\_structs::scheme, 31

procid  
    ez\_parallel\_structs::scheme, 31

recv\_boundaries  
    ez\_parallel\_structs::scheme, 31

rowdcmpsizes  
    ez\_parallel\_structs::scheme, 32

send\_boundaries  
    ez\_parallel\_structs::scheme, 32

share\_subgrid\_bdry.f90  
    share\_subgrid\_bdry\_dble\_eh, 65  
    share\_subgrid\_bdry\_dble\_sbr, 66  
    share\_subgrid\_bdry\_dcplx\_eh, 66  
    share\_subgrid\_bdry\_dcplx\_sbr, 66

share\_subgrid\_bdry\_dble\_eh  
    share\_subgrid\_bdry.f90, 65

share\_subgrid\_bdry\_dble\_sbr  
    ez\_parallel::share\_subgrid\_bdry, 34  
    share\_subgrid\_bdry.f90, 66

share\_subgrid\_bdry\_dcplx\_eh  
    share\_subgrid\_bdry.f90, 66

share\_subgrid\_bdry\_dcplx\_sbr  
    ez\_parallel::share\_subgrid\_bdry, 35  
    share\_subgrid\_bdry.f90, 66

spec\_drv\_1d\_1  
    ez\_parallel\_structs, 9

spec\_drv\_1d\_2  
    ez\_parallel\_structs, 9

subarray  
    create\_scheme\_fft.f90, 40  
    create\_scheme\_zero\_pad.f90, 43  
    execute\_scheme\_izero\_pad.f90, 55  
    execute\_scheme\_zero\_pad.f90, 61

subarrays  
    ez\_parallel\_structs::scheme, 32

vslabint  
    ez\_parallel\_structs::scheme, 32

vslabsize  
    ez\_parallel\_structs::scheme, 32

vslabsizeovlp  
    ez\_parallel\_structs::scheme, 33

wsave1  
    ez\_parallel\_structs::scheme, 33

wsave2  
    ez\_parallel\_structs::scheme, 33

wvnubr1  
    ez\_parallel\_structs::scheme, 33

wvnubr2  
    ez\_parallel\_structs::scheme, 33