

The FARGO_THORIN code developer's guide

Generated by Doxygen 1.8.7

Thu Jul 6 2017 10:19:43

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	pair Struct Reference	5
3.1.1	Detailed Description	5
3.1.2	Field Documentation	5
3.1.2.1	x	5
3.1.2.2	y	5
3.2	param Struct Reference	6
3.2.1	Detailed Description	6
3.2.2	Field Documentation	6
3.2.2.1	name	6
3.2.2.2	necessary	6
3.2.2.3	read	6
3.2.2.4	type	7
3.2.2.5	variable	7
3.3	planetary_system Struct Reference	7
3.3.1	Detailed Description	8
3.3.2	Field Documentation	8
3.3.2.1	acc	8
3.3.2.2	ax	8
3.3.2.3	ay	8
3.3.2.4	az	8
3.3.2.5	FeelDisk	8
3.3.2.6	FeelOthers	9
3.3.2.7	mass	9
3.3.2.8	name	9
3.3.2.9	nb	9

3.3.2.10	vx	9
3.3.2.11	vy	9
3.3.2.12	vz	9
3.3.2.13	x	9
3.3.2.14	y	10
3.3.2.15	z	10
3.4	polargrid Struct Reference	10
3.4.1	Detailed Description	10
3.4.2	Field Documentation	10
3.4.2.1	Field	10
3.4.2.2	Name	11
3.4.2.3	Nrad	11
3.4.2.4	Nsec	11
3.5	timeprocess Struct Reference	11
3.5.1	Detailed Description	12
3.5.2	Field Documentation	12
3.5.2.1	clicks	12
3.5.2.2	name	12
4	File Documentation	13
4.1	commbound.c File Reference	13
4.1.1	Detailed Description	13
4.1.2	Function Documentation	14
4.1.2.1	AllocateComm	14
4.1.2.2	CommunicateBoundaries	14
4.1.3	Variable Documentation	15
4.1.3.1	allocated_com	15
4.1.3.2	RecvInnerBoundary	15
4.1.3.3	RecvOuterBoundary	15
4.1.3.4	SendInnerBoundary	15
4.1.3.5	SendOuterBoundary	16
4.1.3.6	size_com	16
4.2	EnergySources.c File Reference	16
4.2.1	Detailed Description	18
4.2.2	LICENSE	18
4.2.3	Macro Definition Documentation	18
4.2.3.1	SOREPS	18
4.2.3.2	SORMAXITERS	18
4.2.4	Function Documentation	18
4.2.4.1	CalculateFlaring	18

4.2.4.2	CalculateQirr	19
4.2.4.3	CalculateQminus	19
4.2.4.4	ChessBoardIndexing	20
4.2.4.5	CreateTorqueMapInfile	20
4.2.4.6	DiffusionCoefs	21
4.2.4.7	EffectiveOpticalDepth	22
4.2.4.8	FluxLimiterValue	22
4.2.4.9	ImplicitRadiativeDiffusion	23
4.2.4.10	InitRadiatDiffusionFields	23
4.2.4.11	IterateRelaxationParameter	24
4.2.4.12	MidplaneVolumeDensity	25
4.2.4.13	OpacityProfile	25
4.2.4.14	SuccessiveOverrelaxation	25
4.2.4.15	SynchronizeOverlapFields	26
4.2.4.16	TemperatureGradient	27
4.2.5	Variable Documentation	27
4.2.5.1	a	27
4.2.5.2	b	28
4.2.5.3	CV	28
4.2.5.4	DiffCoefCentered	28
4.2.5.5	DiffCoefIfaceRad	28
4.2.5.6	DiffCoefIfaceTheta	28
4.2.5.7	DiscretizationCoefA	28
4.2.5.8	DiscretizationCoefB	28
4.2.5.9	omega	28
4.2.5.10	Flaring	28
4.2.5.11	GradTemperMagnitude	28
4.2.5.12	GradTemperRad	28
4.2.5.13	GradTemperTheta	29
4.2.5.14	jchess1st	29
4.2.5.15	jchess2nd	29
4.2.5.16	kappa0	29
4.2.5.17	MatrixNexttoTemperl	29
4.2.5.18	MatrixNexttoTemperlim	29
4.2.5.19	MatrixNexttoTemperlip	29
4.2.5.20	MatrixNexttoTemperljm	29
4.2.5.21	MatrixNexttoTemperljp	29
4.2.5.22	Niterbest	29
4.2.5.23	omegabest	29
4.2.5.24	Opacity	30

4.2.5.25	Qirradiation	30
4.2.5.26	RightHandSide	30
4.2.5.27	VolumeDensity	30
4.3	fargo.h File Reference	30
4.3.1	Detailed Description	31
4.4	fondam.h File Reference	31
4.4.1	Detailed Description	32
4.4.2	Macro Definition Documentation	32
4.4.2.1	AMU_CGS	32
4.4.2.2	ANGSTR_CGS	32
4.4.2.3	AU_SI	32
4.4.2.4	CPUOVERLAP	32
4.4.2.5	fac1o15	32
4.4.2.6	fac1o21	32
4.4.2.7	fac4o75	33
4.4.2.8	FLUX2CU	33
4.4.2.9	G	33
4.4.2.10	G_SI	33
4.4.2.11	GASCONST	33
4.4.2.12	GM_SI	33
4.4.2.13	MMW	33
4.4.2.14	MOLCROSSEC_CGS	33
4.4.2.15	MOLDIAMETER	33
4.4.2.16	MOLWEIGHT	33
4.4.2.17	MSOL_SI	33
4.4.2.18	OPA2CU	34
4.4.2.19	PI	34
4.4.2.20	PRESS2CGS	34
4.4.2.21	R_SI	34
4.4.2.22	R_STANDARD	34
4.4.2.23	RHO2CGS	34
4.4.2.24	SIGMA2CGS	34
4.4.2.25	SQRT2PI_INV	34
4.4.2.26	STEFANBOLTZMANN	34
4.4.2.27	STEFANBOLTZMANNCONTROL	35
4.4.2.28	SURFDENS2CGS	35
4.4.2.29	T2SI	35
4.4.2.30	TIME2SI	35
4.5	Force.c File Reference	35
4.5.1	Detailed Description	36

4.5.2	LICENSE	36
4.5.3	Function Documentation	36
4.5.3.1	ThicknessSmoothing	36
4.5.3.2	UpdateLog	37
4.5.4	Variable Documentation	37
4.5.4.1	NonReflecting	37
4.5.4.2	OpenInner	38
4.6	fpe.c File Reference	38
4.6.1	Detailed Description	38
4.6.2	Function Documentation	38
4.6.2.1	handfpe	38
4.6.2.2	setfpe	39
4.7	global.h File Reference	39
4.7.1	Detailed Description	42
4.7.2	Variable Documentation	42
4.7.2.1	AccretHeating	42
4.7.2.2	ActualizeLuminosity	42
4.7.2.3	AdvecteLabel	42
4.7.2.4	BackReaction	42
4.7.2.5	CellAbscissa	42
4.7.2.6	CellOrdinate	42
4.7.2.7	CentrifugalBalance	43
4.7.2.8	Collisions	43
4.7.2.9	CoolingTimeMed	43
4.7.2.10	CPU_Master	43
4.7.2.11	CPU_Number	43
4.7.2.12	CPU_Rank	43
4.7.2.13	Damping	43
4.7.2.14	Damplnit	44
4.7.2.15	DampVrad	44
4.7.2.16	debug	44
4.7.2.17	DiffusiveParticles	44
4.7.2.18	discard	44
4.7.2.19	DivergenceVelocity	44
4.7.2.20	DragForceRad	44
4.7.2.21	DragForceTheta	44
4.7.2.22	EnergyEq	44
4.7.2.23	EnergyMed	45
4.7.2.24	ExcludeHill	45
4.7.2.25	FakeSequential	45

4.7.2.26	fargostat	45
4.7.2.27	FeelDisk	45
4.7.2.28	GasAccelrad	45
4.7.2.29	GasAcceltheta	45
4.7.2.30	GLOBALNRAD	45
4.7.2.31	GlobalRmed	45
4.7.2.32	globcsvec	46
4.7.2.33	globpressvec	46
4.7.2.34	GotoNextOutput	46
4.7.2.35	GravAccelRad	46
4.7.2.36	GravAccelTheta	46
4.7.2.37	heatsrc	46
4.7.2.38	heatsrc_index	46
4.7.2.39	heatsrc_max	46
4.7.2.40	IMAX	46
4.7.2.41	IMIN	47
4.7.2.42	InitFromFile	47
4.7.2.43	InvDiffRmed	47
4.7.2.44	InvDiffRsup	47
4.7.2.45	invdtpeb_sq	47
4.7.2.46	invdtreb_sq	47
4.7.2.47	InvRinf	47
4.7.2.48	InvRmed	47
4.7.2.49	InvSurf	47
4.7.2.50	LogGrid	48
4.7.2.51	MassTaper	48
4.7.2.52	Max_or_active	48
4.7.2.53	MaxMO_or_active	48
4.7.2.54	Merge	48
4.7.2.55	mergers	48
4.7.2.56	MonitorIntegral	48
4.7.2.57	MonitorNPL	48
4.7.2.58	NewOutputdir	48
4.7.2.59	OmegaFrame	49
4.7.2.60	OmegaInv	49
4.7.2.61	One_or_active	49
4.7.2.62	OnlyInit	49
4.7.2.63	OverridesOutputdir	49
4.7.2.64	ParametricCooling	49
4.7.2.65	PebbleDens	49

4.7.2.66	PebbleGravAccelRad	49
4.7.2.67	PebbleGravAccelTheta	50
4.7.2.68	Pebbles	50
4.7.2.69	PebbleVrad	50
4.7.2.70	PebbleVtheta	50
4.7.2.71	PebDensInit	50
4.7.2.72	PebVradInit	50
4.7.2.73	PebVthetaInit	50
4.7.2.74	PhysicalTime	50
4.7.2.75	PhysicalTimeInitial	51
4.7.2.76	plout	51
4.7.2.77	PrescribedAccretion	51
4.7.2.78	Pressure	51
4.7.2.79	Qbalance	51
4.7.2.80	Qminus	51
4.7.2.81	Qplus	51
4.7.2.82	QplusMed	51
4.7.2.83	Radii	51
4.7.2.84	RhoInt	52
4.7.2.85	RhoStar	52
4.7.2.86	Rinf	52
4.7.2.87	Rmed	52
4.7.2.88	Rmed2	52
4.7.2.89	RocheSmoothing	52
4.7.2.90	Rsup	52
4.7.2.91	SigmaInf	53
4.7.2.92	SigmaMed	53
4.7.2.93	SloppyCFL	53
4.7.2.94	SoundSpeed	53
4.7.2.95	SQRT_ADIABIND_INV	53
4.7.2.96	StellarIrradiation	53
4.7.2.97	StokesNumber	53
4.7.2.98	StoreEnergy	53
4.7.2.99	StoreSigma	54
4.7.2.100	Surf	54
4.7.2.101	TAUPP	54
4.7.2.102	TAURP	54
4.7.2.103	TAURR	54
4.7.2.104	Temperature	54
4.7.2.105	TimeStep	54

4.7.2.106 Torque	54
4.7.2.107 TorqueDensity	55
4.7.2.108 ViscosityAlpha	55
4.7.2.109 vt1D	55
4.7.2.110 VthetaMed	55
4.7.2.111 WaveKiller	55
4.7.2.112 Write_Divergence	55
4.7.2.113 Write_Energy	55
4.7.2.114 Write_Eta	55
4.7.2.115 Write_Qbalance	55
4.7.2.116 Write_Qplus	56
4.7.2.117 Write_Temperature	56
4.7.2.118 WriteTorque	56
4.7.2.119 WriteTorqueMapFile	56
4.7.2.120 Zero_or_active	56
4.8 global_ex.h File Reference	56
4.8.1 Detailed Description	59
4.8.2 Variable Documentation	59
4.8.2.1 AccretHeating	59
4.8.2.2 ActualizeLuminosity	59
4.8.2.3 AdvecteLabel	59
4.8.2.4 BackReaction	59
4.8.2.5 CellAbscissa	59
4.8.2.6 CellOrdinate	59
4.8.2.7 CentrifugalBalance	59
4.8.2.8 Collisions	59
4.8.2.9 CoolingTimeMed	60
4.8.2.10 CPU_Master	60
4.8.2.11 CPU_Number	60
4.8.2.12 CPU_Rank	60
4.8.2.13 Damping	60
4.8.2.14 DampInIt	60
4.8.2.15 DampVrad	60
4.8.2.16 debug	61
4.8.2.17 DiffusiveParticles	61
4.8.2.18 discard	61
4.8.2.19 DivergenceVelocity	61
4.8.2.20 DragForceRad	61
4.8.2.21 DragForceTheta	61
4.8.2.22 EnergyEq	61

4.8.2.23	EnergyMed	61
4.8.2.24	ExcludeHill	61
4.8.2.25	FakeSequential	62
4.8.2.26	fargostat	62
4.8.2.27	FeelDisk	62
4.8.2.28	GasAccelrad	62
4.8.2.29	GasAcceltheta	62
4.8.2.30	GLOBALNRAD	62
4.8.2.31	GlobalRmed	62
4.8.2.32	globcsvec	62
4.8.2.33	globpressvec	62
4.8.2.34	GotoNextOutput	63
4.8.2.35	GravAccelRad	63
4.8.2.36	GravAccelTheta	63
4.8.2.37	heatsrc	63
4.8.2.38	heatsrc_index	63
4.8.2.39	heatsrc_max	63
4.8.2.40	IMAX	63
4.8.2.41	IMIN	63
4.8.2.42	InitFromFile	63
4.8.2.43	InvDiffRmed	64
4.8.2.44	InvDiffRsup	64
4.8.2.45	invdtpeb_sq	64
4.8.2.46	invdtreb_sq	64
4.8.2.47	InvRinf	64
4.8.2.48	InvRmed	64
4.8.2.49	InvSurf	64
4.8.2.50	LogGrid	64
4.8.2.51	MassTaper	64
4.8.2.52	Max_or_active	65
4.8.2.53	MaxMO_or_active	65
4.8.2.54	Merge	65
4.8.2.55	mergers	65
4.8.2.56	MonitorIntegral	65
4.8.2.57	MonitorNPL	65
4.8.2.58	NewOutputdir	65
4.8.2.59	OmegaFrame	65
4.8.2.60	OmegaInv	65
4.8.2.61	One_or_active	66
4.8.2.62	OnlyInit	66

4.8.2.63	OverridesOutputdir	66
4.8.2.64	ParametricCooling	66
4.8.2.65	PebbleDens	66
4.8.2.66	PebbleGravAccelRad	66
4.8.2.67	PebbleGravAccelTheta	66
4.8.2.68	Pebbles	66
4.8.2.69	PebbleVrad	66
4.8.2.70	PebbleVtheta	67
4.8.2.71	PebDensInit	67
4.8.2.72	PebVradInit	67
4.8.2.73	PebVthetaInit	67
4.8.2.74	PhysicalTime	67
4.8.2.75	PhysicalTimeInitial	67
4.8.2.76	plout	67
4.8.2.77	PrescribedAccretion	67
4.8.2.78	Pressure	68
4.8.2.79	Qbalance	68
4.8.2.80	Qminus	68
4.8.2.81	Qplus	68
4.8.2.82	QplusMed	68
4.8.2.83	Radii	68
4.8.2.84	RhoInt	68
4.8.2.85	RhoStar	68
4.8.2.86	Rinf	68
4.8.2.87	Rmed	69
4.8.2.88	Rmed2	69
4.8.2.89	RocheSmoothing	69
4.8.2.90	Rsup	69
4.8.2.91	SigmaInf	69
4.8.2.92	SigmaMed	69
4.8.2.93	SloppyCFL	69
4.8.2.94	SoundSpeed	70
4.8.2.95	SQRT_ADIABIND_INV	70
4.8.2.96	StellarIrradiation	70
4.8.2.97	StokesNumber	70
4.8.2.98	StoreEnergy	70
4.8.2.99	StoreSigma	70
4.8.2.100	Surf	70
4.8.2.101	TAUPP	70
4.8.2.102	TAURP	71

4.8.2.103 TAURR	71
4.8.2.104 Temperature	71
4.8.2.105 TimeStep	71
4.8.2.106 Torque	71
4.8.2.107 TorqueDensity	71
4.8.2.108 ViscosityAlpha	71
4.8.2.109 vt1D	71
4.8.2.110 VthetaMed	71
4.8.2.111 WaveKiller	72
4.8.2.112 Write_Divergence	72
4.8.2.113 Write_Energy	72
4.8.2.114 Write_Eta	72
4.8.2.115 Write_Qbalance	72
4.8.2.116 Write_Qplus	72
4.8.2.117 Write_Temperature	72
4.8.2.118 WriteTorque	72
4.8.2.119 WriteTorqueMapFile	72
4.8.2.120 Zero_or_active	73
4.9 Init.c File Reference	73
4.9.1 Detailed Description	73
4.9.2 Function Documentation	74
4.9.2.1 Initialization	74
4.9.2.2 InitLabel	74
4.9.2.3 ReadfromAsciiFile	75
4.9.2.4 ReadfromFile	76
4.9.3 Variable Documentation	76
4.9.3.1 NbRestart	76
4.9.3.2 Restart	76
4.10 Interpret.c File Reference	76
4.10.1 Detailed Description	78
4.10.2 Macro Definition Documentation	78
4.10.2.1 MAXVARIABLES	78
4.10.3 Function Documentation	78
4.10.3.1 GiveSpecificTime	78
4.10.3.2 GiveTimeInfo	78
4.10.3.3 InitSpecificTime	79
4.10.3.4 PrintUsage	79
4.10.3.5 ReadVariables	80
4.10.3.6 TellEverything	81
4.10.3.7 TellNbOrbits	81

4.10.3.8	TellNbOutputs	82
4.10.3.9	var	82
4.10.4	Variable Documentation	82
4.10.4.1	begin_i	82
4.10.4.2	Corotating	83
4.10.4.3	Current	83
4.10.4.4	CurrentUser	83
4.10.4.5	FastTransport	83
4.10.4.6	First	83
4.10.4.7	FirstStep	83
4.10.4.8	FirstUser	83
4.10.4.9	GuidingCenter	83
4.10.4.10	Indirect_Term	83
4.10.4.11	IsDisk	84
4.10.4.12	NonReflecting	84
4.10.4.13	OpenInner	84
4.10.4.14	OuterSourceMass	84
4.10.4.15	Preceeding	84
4.10.4.16	PreceedingUser	84
4.10.4.17	Restart	84
4.10.4.18	Ticks	84
4.10.4.19	VariableIndex	84
4.10.4.20	VariableSet	84
4.10.4.21	Write_Density	85
4.10.4.22	Write_Velocity	85
4.11	LowTasks.c File Reference	85
4.11.1	Detailed Description	85
4.11.2	Function Documentation	86
4.11.2.1	CreatePolarGrid	86
4.11.2.2	DumpSources	86
4.11.2.3	fopenp	87
4.11.2.4	GetGlobalFrac	88
4.11.2.5	MakeDir	88
4.11.2.6	mastererr	89
4.11.2.7	masterprint	90
4.11.2.8	message	91
4.11.2.9	MultiplyPolarGridbyConstant	91
4.11.2.10	prs_error	92
4.11.2.11	prs_exit	93
4.12	main.c File Reference	94

4.12.1 Detailed Description	95
4.12.2 Function Documentation	95
4.12.2.1 main	95
4.12.3 Variable Documentation	97
4.12.3.1 begin_i	97
4.12.3.2 Corotating	97
4.12.3.3 DumpTorqueDensNow	97
4.12.3.4 DumpTorqueNow	97
4.12.3.5 InnerOutputCounter	97
4.12.3.6 LostMass	97
4.12.3.7 NbRestart	97
4.12.3.8 OpenInner	97
4.12.3.9 Restart	97
4.12.3.10 ScalingFactor	98
4.12.3.11 StillWriteOneOutput	98
4.13 merge.c File Reference	98
4.13.1 Detailed Description	98
4.13.2 Function Documentation	98
4.13.2.1 merge	98
4.14 mpi_dummy.c File Reference	99
4.14.1 Detailed Description	100
4.14.2 Function Documentation	100
4.14.2.1 MPI_Allreduce	100
4.14.2.2 MPI_Barrier	100
4.14.2.3 MPI_Bcast	101
4.14.2.4 MPI_Comm_rank	101
4.14.2.5 MPI_Comm_size	102
4.14.2.6 MPI_Finalize	102
4.14.2.7 MPI_Init	103
4.14.2.8 MPI_Irecv	104
4.14.2.9 MPI_Isend	104
4.14.2.10 MPI_Recv	105
4.14.2.11 MPI_Send	105
4.14.2.12 MPI_Wait	106
4.15 mpi_dummy.h File Reference	106
4.15.1 Detailed Description	107
4.15.2 Macro Definition Documentation	107
4.15.2.1 MPI_CHAR	107
4.15.2.2 MPI_COMM_WORLD	107
4.15.2.3 MPI_DOUBLE	108

4.15.2.4	MPI_INT	108
4.15.2.5	MPI_LONG	108
4.15.2.6	MPI_MAX	108
4.15.2.7	MPI_MIN	108
4.15.2.8	MPI_SUM	108
4.15.3	Typedef Documentation	108
4.15.3.1	MPI_Request	108
4.15.3.2	MPI_Status	108
4.15.4	Function Documentation	108
4.15.4.1	MPI_Allreduce	108
4.15.4.2	MPI_Barrier	108
4.15.4.3	MPI_Bcast	109
4.15.4.4	MPI_Comm_rank	109
4.15.4.5	MPI_Comm_size	109
4.15.4.6	MPI_Finalize	110
4.15.4.7	MPI_Init	110
4.15.4.8	MPI_Irecv	110
4.15.4.9	MPI_Isend	111
4.15.4.10	MPI_Recv	111
4.15.4.11	MPI_Send	112
4.15.4.12	MPI_Wait	113
4.16	mpiTasks.c File Reference	113
4.16.1	Detailed Description	113
4.16.2	Function Documentation	113
4.16.2.1	mpi_make1Dprofile	113
4.17	Output.c File Reference	114
4.17.1	Detailed Description	115
4.17.2	Function Documentation	116
4.17.2.1	ActualizeQbalance	116
4.17.2.2	DumpOmegaFrame	116
4.17.2.3	EmptyPlanetSystemFile	117
4.17.2.4	GetfromPlanetFile	117
4.17.2.5	GetOmegaFrame	118
4.17.2.6	RestartPlanetarySystem	118
4.17.2.7	SendOutput	118
4.17.2.8	WriteBigPlanetFile	119
4.17.2.9	WriteBigPlanetSystemFile	120
4.17.2.10	WriteDim	120
4.17.2.11	WriteDiskPolar	121
4.17.2.12	WritePlanetFile	122

4.17.2.13 WritePlanetSystemFile	122
4.17.3 Variable Documentation	122
4.17.3.1 IsDisk	123
4.17.3.2 LostMass	123
4.17.3.3 MplanetVirtual	123
4.17.3.4 OmegaFrame	123
4.17.3.5 VXplanet	123
4.17.3.6 VYplanet	123
4.17.3.7 Write_Density	123
4.17.3.8 Write_Velocity	123
4.17.3.9 Xplanet	123
4.17.3.10 Yplanet	124
4.18 param.h File Reference	124
4.18.1 Detailed Description	126
4.18.2 Variable Documentation	126
4.18.2.1 ACCRETIONALHEATING	126
4.18.2.2 ACCRETIONRATE	126
4.18.2.3 ADIABIND	126
4.18.2.4 ADVLABEL	126
4.18.2.5 ALPHAVISCOSITY	126
4.18.2.6 ASPECTRATIO	126
4.18.2.7 BACKREACTION	126
4.18.2.8 CAVITYRADIUS	127
4.18.2.9 CAVITYRATIO	127
4.18.2.10 CAVITYWIDTH	127
4.18.2.11 COOLINGTIME	127
4.18.2.12 DAMPINGPERIODFRAC	127
4.18.2.13 DAMPINGRMAXFRAC	127
4.18.2.14 DAMPINGRMINFRAC	127
4.18.2.15 DAMPTOWARDS	127
4.18.2.16 DENSINFILE	127
4.18.2.17 DISCALBEDO	128
4.18.2.18 DISK	128
4.18.2.19 DT	128
4.18.2.20 ECCENTRICITY	128
4.18.2.21 EFFECTIVETEMPERATURE	128
4.18.2.22 ENERGYEQUATION	128
4.18.2.23 EXCLUDEHILL	128
4.18.2.24 FLARINGINDEX	128
4.18.2.25 FRAME	128

4.18.2.26 GETTORQUEFORPLANET	129
4.18.2.27 GRIDSPACING	129
4.18.2.28 HEATINGDELAY	129
4.18.2.29 HILLCUT	129
4.18.2.30 IAS15MINDT	129
4.18.2.31 IAS15PRECISION	129
4.18.2.32 IMPOSEDDISKDRIFT	129
4.18.2.33 INDIRECTTERM	129
4.18.2.34 INITIALIZEFROMFILE	129
4.18.2.35 LAMBDADOUBLING	130
4.18.2.36 MASSTAPER	130
4.18.2.37 NINTERM	130
4.18.2.38 NOUTELEMENTS	130
4.18.2.39 NRAD	130
4.18.2.40 NSEC	130
4.18.2.41 NTOT	130
4.18.2.42 OMEGAFRAME	130
4.18.2.43 OPACITYDROP	130
4.18.2.44 OPENINNERBOUNDARY	131
4.18.2.45 OUTERSOURCEMASS	131
4.18.2.46 OUTPUTDIR	131
4.18.2.47 PARAMETRICACCRETION	131
4.18.2.48 PARAMETRICOPACITY	131
4.18.2.49 PARTICLEDIFFUSION	131
4.18.2.50 PEBBLEACCRETION	131
4.18.2.51 PEBBLEALPHA	131
4.18.2.52 PEBBLEBULKDENS	132
4.18.2.53 PEBBLECOAGULATION	132
4.18.2.54 PEBBLEFLUX	132
4.18.2.55 PLANETARYDENSITY	132
4.18.2.56 PLANETCONFIG	132
4.18.2.57 PLANETSFEELDISK	132
4.18.2.58 RELEASEDATE	132
4.18.2.59 RELEASERADIUS	132
4.18.2.60 RESOLVECOLLISIONS	132
4.18.2.61 RMAX	133
4.18.2.62 RMIN	133
4.18.2.63 ROCHESMOOTHING	133
4.18.2.64 SCHMIDTNUMBER	133
4.18.2.65 SIGMA0	133

4.18.2.66 SIGMASLOPE	133
4.18.2.67 STELLARIRRADIATION	133
4.18.2.68 STELLARRADIUS	133
4.18.2.69 TARGETNPL	133
4.18.2.70 TEMPERINFILE	134
4.18.2.71 THICKNESSSMOOTHING	134
4.18.2.72 TORQUEMAPINFILE	134
4.18.2.73 TRANSITIONRADIUS	134
4.18.2.74 TRANSITIONRATIO	134
4.18.2.75 TRANSITIONWIDTH	134
4.18.2.76 TRANSPORT	134
4.18.2.77 VERTICALDAMPING	134
4.18.2.78 VISCOSITY	134
4.18.2.79 VRADINFILE	135
4.18.2.80 VTHETAFILE	135
4.18.2.81 WRITEDENSITY	135
4.18.2.82 WRITEDIVV	135
4.18.2.83 WRITEENERGY	135
4.18.2.84 WRITEETA	135
4.18.2.85 WRITEQBALANCE	135
4.18.2.86 WRITEQPLUS	135
4.18.2.87 WRITETEMPERATURE	135
4.18.2.88 WRITETORQUEFILES	136
4.18.2.89 WRITEVELOCITY	136
4.19 param_noex.h File Reference	136
4.19.1 Detailed Description	138
4.19.2 Variable Documentation	138
4.19.2.1 ACCRETIONALHEATING	138
4.19.2.2 ACCRETIONRATE	138
4.19.2.3 ADIABIND	138
4.19.2.4 ADVLABEL	138
4.19.2.5 ALPHAVISCOSITY	138
4.19.2.6 ASPECTRATIO	138
4.19.2.7 BACKREACTION	138
4.19.2.8 CAVITYRADIUS	139
4.19.2.9 CAVITYRATIO	139
4.19.2.10 CAVITYWIDTH	139
4.19.2.11 COOLINGTIME	139
4.19.2.12 DAMPINGPERIODFRAC	139
4.19.2.13 DAMPINGRMAXFRAC	139

4.19.2.14 DAMPINGRMINFRAC	139
4.19.2.15 DAMPTOWARDS	139
4.19.2.16 DENSINFILE	139
4.19.2.17 DISCALBEDO	140
4.19.2.18 DISK	140
4.19.2.19 DT	140
4.19.2.20 ECCENTRICITY	140
4.19.2.21 EFFECTIVETEMPERATURE	140
4.19.2.22 ENERGYEQUATION	140
4.19.2.23 EXCLUDEHILL	140
4.19.2.24 FLARINGINDEX	140
4.19.2.25 FRAME	140
4.19.2.26 GETTORQUEFORPLANET	141
4.19.2.27 GRIDSPACING	141
4.19.2.28 HEATINGDELAY	141
4.19.2.29 HILLCUT	141
4.19.2.30 IAS15MINDT	141
4.19.2.31 IAS15PRECISION	141
4.19.2.32 IMPOSEDDISKDRIFT	141
4.19.2.33 INDIRECTTERM	141
4.19.2.34 INITIALIZEFROMFILE	141
4.19.2.35 LAMBDADOUBLING	142
4.19.2.36 MASSTAPER	142
4.19.2.37 NINTERM	142
4.19.2.38 NOUTELEMENTS	142
4.19.2.39 NRAD	142
4.19.2.40 NSEC	142
4.19.2.41 NTOT	142
4.19.2.42 OMEGAFRAME	142
4.19.2.43 OPACITYDROP	142
4.19.2.44 OPENINNERBOUNDARY	143
4.19.2.45 OUTERSOURCEMASS	143
4.19.2.46 OUTPUTDIR	143
4.19.2.47 PARAMETRICACCRETION	143
4.19.2.48 PARAMETRICOPACITY	143
4.19.2.49 PARTICLEDIFFUSION	143
4.19.2.50 PEBBLEACCRETION	143
4.19.2.51 PEBBLEALPHA	143
4.19.2.52 PEBBLEBULKDENS	144
4.19.2.53 PEBBLECOAGULATION	144

4.19.2.54 PEBBLEFLUX	144
4.19.2.55 PLANETARYDENSITY	144
4.19.2.56 PLANETCONFIG	144
4.19.2.57 PLANETSFEELDISK	144
4.19.2.58 RELEASEDATE	144
4.19.2.59 RELEASERADIUS	144
4.19.2.60 RESOLVECOLLISIONS	144
4.19.2.61 RMAX	145
4.19.2.62 RMIN	145
4.19.2.63 ROCHESMOOTHING	145
4.19.2.64 SCHMIDTNUMBER	145
4.19.2.65 SIGMA0	145
4.19.2.66 SIGMASLOPE	145
4.19.2.67 STELLAIRRADIATION	145
4.19.2.68 STELLARRADIUS	145
4.19.2.69 TARGETNPL	145
4.19.2.70 TEMPERINFILE	146
4.19.2.71 THICKNESSSMOOTHING	146
4.19.2.72 TORQUEMAPINFILE	146
4.19.2.73 TRANSITIONRADIUS	146
4.19.2.74 TRANSITIONRATIO	146
4.19.2.75 TRANSITIONWIDTH	146
4.19.2.76 TRANSPORT	146
4.19.2.77 VERTICALDAMPING	146
4.19.2.78 VISCOSITY	146
4.19.2.79 VRADINFILE	147
4.19.2.80 VTHETAFILE	147
4.19.2.81 WRITEDENSITY	147
4.19.2.82 WRITEDIVV	147
4.19.2.83 WRITEENERGY	147
4.19.2.84 WRITEETA	147
4.19.2.85 WRITEQBALANCE	147
4.19.2.86 WRITEQPLUS	147
4.19.2.87 WRITETEMPERATURE	147
4.19.2.88 WRITETORQUEFILES	148
4.19.2.89 WRITEVELOCITY	148
4.20 Pebbles.c File Reference	148
4.20.1 Detailed Description	149
4.20.2 LICENSE	150
4.20.3 Macro Definition Documentation	150

4.20.3.1	TRAPEZEPS	150
4.20.3.2	TRAPEZMAX	150
4.20.4	Function Documentation	150
4.20.4.1	AccretePebblesOntoPlanets	150
4.20.4.2	BckpFieldsForBC	151
4.20.4.3	CorrectPebblesVtheta	152
4.20.4.4	CriticalCharTime	152
4.20.4.5	DetectCrashPebbles	153
4.20.4.6	EquilPebbleDisk	153
4.20.4.7	EtaPressureSupport	154
4.20.4.8	EvolvePebbleDisk	154
4.20.4.9	InitPebbleArrays	155
4.20.4.10	InitPebblesViaFlux	156
4.20.4.11	IntegrateColumnMass	156
4.20.4.12	ParametricAccretion	157
4.20.4.13	ParticleDiffusion	157
4.20.4.14	PebbleStokesNumbers	158
4.20.4.15	RestartPebbleDisk	159
4.20.4.16	SourceTermsPebbles	159
4.20.4.17	SubStep1Pebbles	160
4.20.4.18	SynchronizePebbleDisc	160
4.20.4.19	Trapzd	161
4.20.4.20	WritePebbles	161
4.20.5	Variable Documentation	162
4.20.5.1	AccretedMass	162
4.20.5.2	EtaCellCentered	162
4.20.5.3	EtaFaceCentered	162
4.20.5.4	FastTransport	162
4.20.5.5	PebbleAccelrad	162
4.20.5.6	PebbleAcceltheta	163
4.20.5.7	PebbleDensTemp	163
4.20.5.8	PebbleSize	163
4.20.5.9	pebbulkdens	163
4.20.5.10	Restart	163
4.21	Pframeforce.c File Reference	163
4.21.1	Detailed Description	164
4.21.2	LICENSE	164
4.21.3	Function Documentation	164
4.21.3.1	AdvanceSystemFromDisk	164
4.21.3.2	ConstructSequence	165

4.21.3.3	DampingTW04	165
4.21.3.4	FillForcesArrays	166
4.21.3.5	InitGasDensityEnergy	167
4.21.3.6	InitGasVelocity	167
4.21.4	Variable Documentation	168
4.21.4.1	AllowAccretion	168
4.21.4.2	Corotating	168
4.21.4.3	DumpTorqueDensNow	168
4.21.4.4	DumpTorqueNow	169
4.21.4.5	Indirect_Term	169
4.21.4.6	IndirectTerm	169
4.21.4.7	vt_cent	169
4.21.4.8	vt_int	169
4.22	Planet.c File Reference	169
4.22.1	Detailed Description	169
4.22.2	Function Documentation	170
4.22.2.1	AccreteOntoPlanets	170
4.22.2.2	FindOrbitalElements	170
4.23	proto.h File Reference	171
4.23.1	Detailed Description	175
4.23.2	Function Documentation	175
4.23.2.1	AccreteOntoPlanets	175
4.23.2.2	AccretePebblesOntoPlanets	175
4.23.2.3	ActualiseGas	175
4.23.2.4	ActualizeQbalance	175
4.23.2.5	AdditionalForces	176
4.23.2.6	AdvanceSystemFromDisk	176
4.23.2.7	AdvanceSystemRebound	176
4.23.2.8	AdvectSHIFT	176
4.23.2.9	AlgoGas	176
4.23.2.10	AllocateComm	176
4.23.2.11	AllocPlanetSystem	177
4.23.2.12	ApplyBoundaryCondition	177
4.23.2.13	ApplyOuterSourceMass	178
4.23.2.14	AspectRatio	178
4.23.2.15	BckpFieldsForBC	178
4.23.2.16	CalculateFlaring	178
4.23.2.17	CalculateQirr	179
4.23.2.18	CalculateQminus	179
4.23.2.19	CheckRebin	179

4.23.2.20 ChessBoardIndexing	179
4.23.2.21 ChkCloseEncWithPI	180
4.23.2.22 CommunicateBoundaries	180
4.23.2.23 ComputeAverageThetaVelocities	180
4.23.2.24 ComputeConstantResidual	180
4.23.2.25 ComputeExtQty	180
4.23.2.26 ComputeLRMomenta	180
4.23.2.27 ComputePressureField	180
4.23.2.28 ComputeResiduals	180
4.23.2.29 ComputeSoundSpeed	181
4.23.2.30 ComputeSpeQty	181
4.23.2.31 ComputeStarRad	181
4.23.2.32 ComputeStarTheta	181
4.23.2.33 ComputeTemperatureField	181
4.23.2.34 ComputeThetaElongations	181
4.23.2.35 ComputeVelocities	181
4.23.2.36 ConditionCFL	181
4.23.2.37 ConstructSequence	181
4.23.2.38 CorrectPebblesVtheta	181
4.23.2.39 CorrectVtheta	181
4.23.2.40 CreatePolarGrid	182
4.23.2.41 CreateTorqueMapInfile	182
4.23.2.42 CriticalCharTime	182
4.23.2.43 DampingBoundary	182
4.23.2.44 DampingTW04	182
4.23.2.45 DampPebbles	182
4.23.2.46 DetectCrash	182
4.23.2.47 DetectCrashPebbles	183
4.23.2.48 DiffusionCoefs	183
4.23.2.49 DiscardParticlesDist	184
4.23.2.50 DiscardParticlesUnbound	184
4.23.2.51 DivisePolarGrid	184
4.23.2.52 DumpOmegaFrame	184
4.23.2.53 DumpSources	184
4.23.2.54 EffectiveOpticalDepth	184
4.23.2.55 EmptyPlanetSystemFile	184
4.23.2.56 Energy	184
4.23.2.57 EquilPebbleDisk	184
4.23.2.58 EtaPressureSupport	184
4.23.2.59 EvolvePebbleDisk	184

4.23.2.60 FillCoolingTime	184
4.23.2.61 FillEnergy	185
4.23.2.62 FillForcesArrays	186
4.23.2.63 FillPolar1DArrays	186
4.23.2.64 FillQplus	186
4.23.2.65 FillSigma	187
4.23.2.66 FillVtheta	187
4.23.2.67 FindNumberOfPlanets	188
4.23.2.68 FindOrbitalElements	188
4.23.2.69 FluxLimiterValue	188
4.23.2.70 fopenp	188
4.23.2.71 FreePlanetary	188
4.23.2.72 FViscosity	189
4.23.2.73 GasMomentum	189
4.23.2.74 GasTotalEnergy	189
4.23.2.75 GasTotalMass	189
4.23.2.76 GetfromPlanetFile	190
4.23.2.77 GetGlobalFrac	190
4.23.2.78 GetOmegaFrame	190
4.23.2.79 GetPsysInfo	190
4.23.2.80 GetPsysInfoFromRsim	190
4.23.2.81 GiveSpecificTime	191
4.23.2.82 GiveTimeInfo	191
4.23.2.83 handfpe	191
4.23.2.84 ImplicitRadiativeDiffusion	191
4.23.2.85 ImposeKeplerianEdges	191
4.23.2.86 InitComputeAccel	191
4.23.2.87 InitCoolingTime	192
4.23.2.88 InitEuler	192
4.23.2.89 InitGas	192
4.23.2.90 InitGasDensityEnergy	192
4.23.2.91 InitGasVelocity	192
4.23.2.92 Initialization	192
4.23.2.93 InitLabel	192
4.23.2.94 InitPebbleArrays	193
4.23.2.95 InitPebblesViaFlux	193
4.23.2.96 InitPlanetarySystem	193
4.23.2.97 InitQplus	193
4.23.2.98 InitRadiatDiffusionFields	193
4.23.2.99 InitSpecificTime	194

4.23.2.100	InitTransport	194
4.23.2.101	InitVariables	194
4.23.2.102	InitViscosity	195
4.23.2.103	IntegrateColumnMass	196
4.23.2.104	IterateRelaxationParameter	196
4.23.2.105	ListPlanets	196
4.23.2.106	MakeDir	197
4.23.2.107	mastererr	197
4.23.2.108	masterprint	197
4.23.2.109	max2	198
4.23.2.110	merge	198
4.23.2.111	message	198
4.23.2.112	MidplaneVolumeDensity	198
4.23.2.113	min2	198
4.23.2.114	MinStepForRebound	199
4.23.2.115	mpi_make1Dprofile	199
4.23.2.116	MultiplyPolarGridbyConstant	199
4.23.2.117	NonReflectingBoundary	199
4.23.2.118	OneWindRad	199
4.23.2.119	OneWindRadPebbles	199
4.23.2.120	OneWindTheta	199
4.23.2.121	OneWindThetaPebbles	199
4.23.2.122	OpacityProfile	199
4.23.2.123	OpenBoundary	199
4.23.2.124	OutputElements	199
4.23.2.125	OutputNbodySimulation	200
4.23.2.126	ParametricAccretion	200
4.23.2.127	ParticleDiffusion	200
4.23.2.128	PebbleStokesNumbers	200
4.23.2.129	PrintUsage	200
4.23.2.130	prs_error	200
4.23.2.131	prs_exit	200
4.23.2.132	QuantitiesAdvection	200
4.23.2.133	QuantitiesAdvectionPebbles	200
4.23.2.134	ReadfromAsciiFile	200
4.23.2.135	ReadfromFile	200
4.23.2.136	ReadPrevDim	200
4.23.2.137	ReadVariables	201
4.23.2.138	RefillEnergy	201
4.23.2.139	RefillSigma	201

4.23.2.140ResolveCollisions	201
4.23.2.141RestartPebbleDisk	201
4.23.2.142RestartPlanetarySystem	201
4.23.2.143RestartReboundSimulation	202
4.23.2.144RotatePsys	202
4.23.2.145SendOutput	202
4.23.2.146setfpe	202
4.23.2.147SetupIntegratorParams	203
4.23.2.148SetupReboundSimulation	203
4.23.2.149SetWaveKillingZones	203
4.23.2.150Sigma	204
4.23.2.151SourceTermsPebbles	204
4.23.2.152SplitDomain	204
4.23.2.153SubStep1	205
4.23.2.154SubStep1Pebbles	205
4.23.2.155SubStep2	205
4.23.2.156SubStep3	205
4.23.2.157SuccessiveOverrelaxation	205
4.23.2.158SynchronizeFargoRebound	205
4.23.2.159SynchronizeOverlapFields	205
4.23.2.160SynchronizePebbleDisc	205
4.23.2.161TellEverything	206
4.23.2.162TellNbOrbits	207
4.23.2.163TellNbOutputs	207
4.23.2.164TemperatureGradient	207
4.23.2.165ThicknessSmoothing	207
4.23.2.166Transport	207
4.23.2.167TransportPebbles	207
4.23.2.168Trapzd	208
4.23.2.169UpdateDivVelocAndStressTensor	208
4.23.2.170UpdateLog	208
4.23.2.171UpdateVelocityWithViscousTerms	208
4.23.2.172VanLeerRadial	208
4.23.2.173VanLeerTheta	208
4.23.2.174var	208
4.23.2.175ViscousTerms	209
4.23.2.176WriteBigPlanetFile	209
4.23.2.177WriteBigPlanetSystemFile	209
4.23.2.178WriteDim	209
4.23.2.179WriteDiskPolar	209

4.23.2.180	WritePebbles	209
4.23.2.181	WritePlanetFile	209
4.23.2.182	WritePlanetSystemFile	209
4.24	Psys.c File Reference	209
4.24.1	Detailed Description	210
4.24.2	Function Documentation	210
4.24.2.1	AllocPlanetSystem	210
4.24.2.2	FindNumberOfPlanets	211
4.24.2.3	FreePlanetary	212
4.24.2.4	GetPsysInfo	212
4.24.2.5	GetPsysInfoFromRsim	212
4.24.2.6	InitPlanetarySystem	212
4.24.2.7	ListPlanets	212
4.24.2.8	RotatePsys	212
4.24.3	Variable Documentation	212
4.24.3.1	GuidingCenter	213
4.24.3.2	Xplanet	213
4.24.3.3	Yplanet	213
4.25	rebin.c File Reference	213
4.25.1	Detailed Description	213
4.25.2	Function Documentation	214
4.25.2.1	CheckRebin	214
4.25.2.2	ReadPrevDim	214
4.25.3	Variable Documentation	214
4.25.3.1	New_r	214
4.25.3.2	OldNRAD	214
4.25.3.3	OldNSEC	215
4.25.3.4	OldRadii	215
4.25.3.5	OldRmed	215
4.26	ReboundInterface.c File Reference	215
4.26.1	Detailed Description	216
4.26.2	LICENSE	216
4.26.3	Function Documentation	216
4.26.3.1	AdvanceSystemRebound	216
4.26.3.2	DiscardParticlesDist	217
4.26.3.3	MinStepForRebound	217
4.26.3.4	OutputElements	217
4.26.3.5	OutputNbodySimulation	217
4.26.3.6	ResolveCollisions	218
4.26.3.7	RestartReboundSimulation	218

4.26.3.8	SetupIntegratorParams	219
4.26.3.9	SetupReboundSimulation	219
4.26.3.10	SynchronizeFargoRebound	220
4.26.4	Variable Documentation	220
4.26.4.1	Corotating	220
4.26.4.2	OmegaFrame	220
4.27	SideEuler.c File Reference	220
4.27.1	Detailed Description	221
4.27.2	Function Documentation	222
4.27.2.1	ApplyBoundaryCondition	222
4.27.2.2	ApplyOuterSourceMass	222
4.27.2.3	CorrectVtheta	223
4.27.2.4	DampingBoundary	223
4.27.2.5	DampPebbles	223
4.27.2.6	DividePolarGrid	223
4.27.2.7	GasMomentum	223
4.27.2.8	GasTotalEnergy	224
4.27.2.9	GasTotalMass	224
4.27.2.10	InitComputeAccel	225
4.27.2.11	NonReflectingBoundary	226
4.27.2.12	OpenBoundary	226
4.27.2.13	SetWaveKillingZones	226
4.27.3	Variable Documentation	227
4.27.3.1	NonReflecting	227
4.27.3.2	OpenInner	227
4.27.3.3	OuterSourceMass	227
4.28	SourceEuler.c File Reference	227
4.28.1	Detailed Description	228
4.28.2	Macro Definition Documentation	229
4.28.2.1	CFLSECURITY	229
4.28.2.2	CVNR	229
4.28.3	Function Documentation	229
4.28.3.1	ActualiseGas	229
4.28.3.2	AlgoGas	229
4.28.3.3	ComputePressureField	230
4.28.3.4	ComputeSoundSpeed	231
4.28.3.5	ComputeTemperatureField	232
4.28.3.6	ConditionCFL	232
4.28.3.7	DetectCrash	233
4.28.3.8	FillPolar1DArrays	233

4.28.3.9	InitEuler	234
4.28.3.10	max2	235
4.28.3.11	min2	235
4.28.3.12	SubStep1	236
4.28.3.13	SubStep2	237
4.28.3.14	SubStep3	237
4.28.4	Variable Documentation	238
4.28.4.1	AlreadyCrashed	238
4.28.4.2	Corotating	238
4.28.4.3	EnergyInt	238
4.28.4.4	EnergyNew	238
4.28.4.5	FastTransport	238
4.28.4.6	GasTimeStepsCFL	238
4.28.4.7	IsDisk	238
4.28.4.8	TemperInt	238
4.28.4.9	timeCRASH	239
4.28.4.10	TimeStep	239
4.28.4.11	VradInt	239
4.28.4.12	VradNew	239
4.28.4.13	VthetaInt	239
4.28.4.14	VthetaNew	239
4.29	split.c File Reference	239
4.29.1	Detailed Description	239
4.29.2	Function Documentation	240
4.29.2.1	SplitDomain	240
4.30	Theo.c File Reference	240
4.30.1	Detailed Description	241
4.30.2	Function Documentation	241
4.30.2.1	Energy	241
4.30.2.2	FillCoolingTime	242
4.30.2.3	FillEnergy	243
4.30.2.4	FillQplus	243
4.30.2.5	FillSigma	244
4.30.2.6	FillVtheta	245
4.30.2.7	InitCoolingTime	245
4.30.2.8	InitQplus	245
4.30.2.9	RefillEnergy	245
4.30.2.10	RefillSigma	246
4.30.2.11	Sigma	246
4.30.3	Variable Documentation	246

4.30.3.1	ScalingFactor	246
4.31	TransportEuler.c File Reference	246
4.31.1	Detailed Description	247
4.31.2	Function Documentation	248
4.31.2.1	AdvectSHIFT	248
4.31.2.2	ComputeExtQty	248
4.31.2.3	ComputeLRMomenta	248
4.31.2.4	ComputeResiduals	249
4.31.2.5	ComputeSpeQty	249
4.31.2.6	ComputeStarRad	250
4.31.2.7	ComputeStarTheta	250
4.31.2.8	ComputeVelocities	250
4.31.2.9	InitTransport	251
4.31.2.10	OneWindRad	251
4.31.2.11	OneWindRadPebbles	252
4.31.2.12	OneWindTheta	252
4.31.2.13	OneWindThetaPebbles	253
4.31.2.14	QuantitiesAdvection	254
4.31.2.15	QuantitiesAdvectionPebbles	254
4.31.2.16	Transport	255
4.31.2.17	TransportPebbles	255
4.31.2.18	VanLeerRadial	256
4.31.2.19	VanLeerTheta	257
4.31.3	Variable Documentation	257
4.31.3.1	dq	257
4.31.3.2	Elongations	257
4.31.3.3	ExtLabel	257
4.31.3.4	FastTransport	257
4.31.3.5	LostMass	258
4.31.3.6	Nshift	258
4.31.3.7	OmegaFrame	258
4.31.3.8	OpenInner	258
4.31.3.9	QRStar	258
4.31.3.10	RadMomM	258
4.31.3.11	RadMomP	258
4.31.3.12	TempShift	258
4.31.3.13	ThetaMomM	258
4.31.3.14	ThetaMomP	258
4.31.3.15	TimeStep	259
4.31.3.16	VMed	259

4.31.3.17 VthetaRes	259
4.31.3.18 Work	259
4.32 types.h File Reference	259
4.32.1 Detailed Description	260
4.32.2 Macro Definition Documentation	261
4.32.2.1 ABSCISSA	261
4.32.2.2 COM_DENSITY	261
4.32.2.3 COM_VRAD	261
4.32.2.4 COM_VTHETA	261
4.32.2.5 COSINE	261
4.32.2.6 FREQUENCY	261
4.32.2.7 GET	261
4.32.2.8 HEIGHT	261
4.32.2.9 INF	261
4.32.2.10 INT	261
4.32.2.11 MARK	261
4.32.2.12 MAX1D	262
4.32.2.13 MAXPLANETS	262
4.32.2.14 NO	262
4.32.2.15 ORDINATE	262
4.32.2.16 REAL	262
4.32.2.17 SINE	262
4.32.2.18 STRING	262
4.32.2.19 SUP	262
4.32.2.20 YES	262
4.32.3 Typedef Documentation	263
4.32.3.1 boolean	263
4.32.3.2 Pair	263
4.32.3.3 Param	263
4.32.3.4 PlanetarySystem	263
4.32.3.5 PolarGrid	263
4.32.3.6 real	263
4.32.3.7 TimeProcess	263
4.33 var.c File Reference	263
4.33.1 Detailed Description	264
4.33.2 Macro Definition Documentation	264
4.33.2.1 __LOCAL	264
4.33.3 Function Documentation	264
4.33.3.1 InitVariables	264
4.34 Viscosity.c File Reference	265

4.34.1 Detailed Description	266
4.34.2 Function Documentation	266
4.34.2.1 AspectRatio	266
4.34.2.2 FViscosity	266
4.34.2.3 ImposeKeplerianEdges	267
4.34.2.4 InitViscosity	267
4.34.2.5 UpdateDivVelocAndStressTensor	268
4.34.2.6 UpdateVelocityWithViscousTerms	268
4.34.3 Variable Documentation	268
4.34.3.1 DPP	268
4.34.3.2 DRP	268
4.34.3.3 DRR	268
Index	269

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

pair	Set of two reals	5
param	The Param structure handles the parameters of the parameter file	6
planetary_system	Contains all the information about a planetary system at a given instant in time	7
polargrid	A structure used to store any scalar field on the computational domain	10
timeprocess	This structure is used for monitoring CPU time usage	11

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

commbound.c	Contains the functions used to synchronize buffer zones on all processes	13
EnergySources.c	Subroutines related to the heating/cooling source terms, numerical solver for the energy equation and radiative diffusion	16
fargo.h	Contains all the include directives requested by the code	30
fondam.h	Contains fundamental constants used thorough the code	31
Force.c	Contains the function to evaluate and write the disk torques acting on planets and also the function to get the thickness smoothing parameter	35
fpe.c	Contains two functions that specify how we handle floating point exceptions	38
global.h	Declares all global variables	39
global_ex.h	This file is created automatically during compilation from global.h	56
Init.c	Contains the functions needed to initialize the hydrodynamics arrays	73
Interpret.c	Contains the functions required to read the parameter file, and functions that provide runtime information	76
LowTasks.c	Contains many low level short functions	85
main.c	Main file of the distribution	94
merge.c	Contains the function that merges the output of different processors	98
mpi_dummy.c	Fake MPI functions library for sequential built	99
mpi_dummy.h	Declaration of fake MPI functions for sequential built	106
mpiTasks.c	Contains the function to create a 1D azimuthally-averaged array from a polar array	113
Output.c	Contains most of the functions that write the output files	114

param.h	Created automatically during compilation from var.c	124
param_noex.h	Created automatically during compilation from var.c	136
Pebbles.c	Contains functions responsible for the pebble disk initialisation, evolution due to source terms and pebble accretion	148
Pframeforce.c	Calculates the gravitational interactions between the disk and massive bodies in terms of a vertically averaged potential	163
Planet.c	Accretion of disk material onto the planets, and solver of planetary orbital elements	169
proto.h	Declaration of all the functions of the FARGO code	171
Psys.c	Contains the functions that set up the planetary system configuration	209
rebin.c	Resample the hydrodynamical fields at a restart with a different resolution	213
ReboundInterface.c	Contains the functions interfacing FARGO with the REBOUND package	215
SideEuler.c	Total mass and angular momentum monitoring, and boundary conditions	220
SourceEuler.c	Contains routines used by the hydrodynamical loop	227
split.c	Split (radially) the mesh among the different processors	239
Theo.c	A few functions that manipulate the surface density profile	240
TransportEuler.c	Functions that handle the transport substep of a hydrodynamical time step	246
types.h	Definition of the structures used in the FARGO code	259
var.c	Contains the function that connects the string of the parameter file to global variables	263
Viscosity.c	Calculation of the viscous force	265

Chapter 3

Data Structure Documentation

3.1 pair Struct Reference

Set of two reals.

```
#include <types.h>
```

Data Fields

- [real x](#)
- [real y](#)

3.1.1 Detailed Description

Set of two reals.

It is used whenever a set of two reals is needed, and it usually represents a vector in the (x,y) plane (e.g., a force), but not only: it is for instance used to store the mass inside and outside the orbit in `MassInOut()`.

Definition at line 26 of file `types.h`.

3.1.2 Field Documentation

3.1.2.1 `real pair::x`

Definition at line 27 of file `types.h`.

Referenced by `AdvanceSystemFromDisk()`, and `FillForcesArrays()`.

3.1.2.2 `real pair::y`

Definition at line 28 of file `types.h`.

Referenced by `AdvanceSystemFromDisk()`, and `FillForcesArrays()`.

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

- [types.h](#)

3.2 param Struct Reference

The Param structure handles the parameters of the parameter file.

```
#include <types.h>
```

Data Fields

- char [name](#) [80]
Name of the parameter.
- int [type](#)
Type of the parameter (e.g.
- char * [variable](#)
A pointer to the corresponding variable.
- int [read](#)
This variable is set to YES if and only if the parameter has been found in the parameter file.
- int [necessary](#)
Tell whether defining the parameter is optional or mandatory.

3.2.1 Detailed Description

The Param structure handles the parameters of the parameter file.

It allows to associate the values found in the file to the strings defining the parameters, and ultimately to the global variables associated to these strings. It is used by the function [var\(\)](#).

Definition at line 74 of file types.h.

3.2.2 Field Documentation

3.2.2.1 char param::name[80]

Name of the parameter.

This is the (case insensitive) string found in the parameter file

Definition at line 75 of file types.h.

3.2.2.2 int param::necessary

Tell whether defining the parameter is optional or mandatory.

Definition at line 79 of file types.h.

Referenced by [var\(\)](#).

3.2.2.3 int param::read

This variable is set to YES if and only if the parameter has been found in the parameter file.

Definition at line 78 of file types.h.

Referenced by [ReadVariables\(\)](#), and [var\(\)](#).

3.2.2.4 int param::type

Type of the parameter (e.g.

INT, REAL, or STRING), see [var.c](#)

Definition at line 76 of file types.h.

Referenced by [var\(\)](#).

3.2.2.5 char* param::variable

A pointer to the corresponding variable.

Definition at line 77 of file types.h.

Referenced by [var\(\)](#).

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

- [types.h](#)

3.3 planetary_system Struct Reference

Contains all the information about a planetary system at a given instant in time.

```
#include <types.h>
```

Data Fields

- int [nb](#)
Number of planets.
- real * [mass](#)
Masses of the planets.
- real * [x](#)
x-coordinate of the planets
- real * [y](#)
y-coordinate of the planets
- real * [z](#)
z-coordinate of the planets
- real * [vx](#)
x-coordinate of the planets' velocities
- real * [vy](#)
y-coordinate of the planets' velocities
- real * [vz](#)
z-coordinate of the planets' velocities
- real * [ax](#)
ax-coordinate of the planets' acceleration from the disk
- real * [ay](#)
ay-coordinate of the planets' acceleration from the disk
- real * [az](#)
az-coordinate of the planets' acceleration from the disk
- real * [acc](#)
The planets' accretion times⁻¹.
- char ** [name](#)

The planets' names.

- `boolean * FeelDisk`

For each planet tells if it feels the disk (ie migrates)

- `boolean * FeelOthers`

For each planet tells if it feels the other planet's gravity.

3.3.1 Detailed Description

Contains all the information about a planetary system at a given instant in time.

#THORIN: 3rd dimension added, acceleration from the disk added.

Definition at line 96 of file types.h.

3.3.2 Field Documentation

3.3.2.1 `real* planetary_system::acc`

The planets' accretion times⁻¹.

Definition at line 108 of file types.h.

Referenced by `AccreteOntoPlanets()`, `AllocPlanetSystem()`, and `InitPlanetarySystem()`.

3.3.2.2 `real* planetary_system::ax`

ax-coordinate of the planets' acceleration from the disk

Definition at line 105 of file types.h.

Referenced by `AllocPlanetSystem()`, `FillForcesArrays()`, and `UpdateLog()`.

3.3.2.3 `real* planetary_system::ay`

ay-coordinate of the planets' acceleration from the disk

Definition at line 106 of file types.h.

Referenced by `AllocPlanetSystem()`, `FillForcesArrays()`, and `UpdateLog()`.

3.3.2.4 `real* planetary_system::az`

az-coordinate of the planets' acceleration from the disk

Definition at line 107 of file types.h.

Referenced by `AllocPlanetSystem()`.

3.3.2.5 `boolean* planetary_system::FeelDisk`

For each planet tells if it feels the disk (ie migrates)

Definition at line 110 of file types.h.

Referenced by `AccreteOntoPlanets()`, `AllocPlanetSystem()`, and `InitPlanetarySystem()`.

3.3.2.6 boolean* planetary_system::FeelOthers

For each planet tells if it feels the other planet's gravity.

Definition at line 111 of file types.h.

Referenced by AllocPlanetSystem(), and InitPlanetarySystem().

3.3.2.7 real* planetary_system::mass

Masses of the planets.

Definition at line 98 of file types.h.

Referenced by AccreteOntoPlanets(), AdvanceSystemRebound(), AllocPlanetSystem(), FillForcesArrays(), InitPlanetarySystem(), and UpdateLog().

3.3.2.8 char planetary_system::name**

The planets' names.

Definition at line 109 of file types.h.

3.3.2.9 int planetary_system::nb

Number of planets.

Definition at line 97 of file types.h.

Referenced by AccreteOntoPlanets(), EmptyPlanetSystemFile(), FillForcesArrays(), InitPlanetarySystem(), ListPlanets(), main(), UpdateLog(), WriteBigPlanetSystemFile(), and WritePlanetSystemFile().

3.3.2.10 real* planetary_system::vx

x-coordinate of the planets' velocities

Definition at line 102 of file types.h.

Referenced by AccreteOntoPlanets(), AllocPlanetSystem(), and InitPlanetarySystem().

3.3.2.11 real* planetary_system::vy

y-coordinate of the planets' velocities

Definition at line 103 of file types.h.

Referenced by AccreteOntoPlanets(), AllocPlanetSystem(), and InitPlanetarySystem().

3.3.2.12 real* planetary_system::vz

z-coordinate of the planets' velocities

Definition at line 104 of file types.h.

Referenced by AllocPlanetSystem().

3.3.2.13 real* planetary_system::x

x-coordinate of the planets

Definition at line 99 of file types.h.

Referenced by `AccreteOntoPlanets()`, `AllocPlanetSystem()`, `FillForcesArrays()`, `GetPsysInfo()`, `InitPlanetarySystem()`, `RestartPlanetarySystem()`, and `UpdateLog()`.

3.3.2.14 `real* planetary_system::y`

y-coordinate of the planets

Definition at line 100 of file types.h.

Referenced by `AccreteOntoPlanets()`, `AllocPlanetSystem()`, `FillForcesArrays()`, `InitPlanetarySystem()`, and `UpdateLog()`.

3.3.2.15 `real* planetary_system::z`

z-coordinate of the planets

Definition at line 101 of file types.h.

Referenced by `AllocPlanetSystem()`, and `FillForcesArrays()`.

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

- [types.h](#)

3.4 polargrid Struct Reference

A structure used to store any scalar field on the computational domain.

```
#include <types.h>
```

Data Fields

- `int Nrad`
Radial size of the grid, in number of zones.
- `int Nsec`
Azimuthal size of the grid, in number of zones.
- `real * Field`
- `char * Name`
*Pointer to the array of Nrad*Nsec reals (e.g., density, etc.)*

3.4.1 Detailed Description

A structure used to store any scalar field on the computational domain.

In addition to the size (`Nrad*Nsec`) and field pointer, it also has a name, which is used by `WritePolarGrid()` to name appropriately the output file, automatically.

Definition at line 37 of file types.h.

3.4.2 Field Documentation

3.4.2.1 `real* polargrid::Field`

Definition at line 40 of file types.h.

Referenced by `AccreteOntoPlanets()`, `AccretePebblesOntoPlanets()`, `ActualizeQbalance()`, `AdvectSHIFT()`, `BckpFieldsForBC()`, `CalculateFlaring()`, `CalculateQirr()`, `CalculateQminus()`, `ComputeLRMomenta()`, `ComputePressureField()`, `ComputeResiduals()`, `ComputeSoundSpeed()`, `ComputeTemperatureField()`, `ComputeVelocities()`, `ConditionCFL()`, `CreatePolarGrid()`, `CreateTorqueMapInfile()`, `CriticalCharTime()`, `DampingTW04()`, `DiffusionCoefs()`, `DivisePolarGrid()`, `EtaPressureSupport()`, `FillForcesArrays()`, `ImplicitRadiativeDiffusion()`, `InitComputeAccel()`, `InitGasVelocity()`, `InitLabel()`, `InitPebblesViaFlux()`, `IterateRelaxationParameter()`, `MidplaneVolumeDensity()`, `NonReflectingBoundary()`, `OpacityProfile()`, `ParticleDiffusion()`, `PebbleStokesNumbers()`, `SourceTermsPebbles()`, `SubStep1()`, `SubStep1Pebbles()`, `SubStep2()`, `SubStep3()`, `SuccessiveOverrelaxation()`, `SynchronizePebbleDisc()`, `TemperatureGradient()`, `ThicknessSmoothing()`, `UpdateDivVelocAndStressTensor()`, `UpdateLog()`, `UpdateVelocityWithViscousTerms()`, `VanLeerRadial()`, `VanLeerTheta()`, and `WriteDiskPolar()`.

3.4.2.2 `char* polargrid::Name`

Pointer to the array of `Nrad*Nsec` reals (e.g., density, etc.)

Name of the `PolarGrid` (can be "dens", "vrad", "vtheta" or "label").

Definition at line 41 of file `types.h`.

Referenced by `CreatePolarGrid()`.

3.4.2.3 `int polargrid::Nrad`

Radial size of the grid, in number of zones.

Definition at line 38 of file `types.h`.

Referenced by `AccreteOntoPlanets()`, `AccretePebblesOntoPlanets()`, `ActualiseGas()`, `ActualizeQbalance()`, `ApplyOuterSourceMass()`, `BckpFieldsForBC()`, `CalculateFlaring()`, `CalculateQminus()`, `ComputeExtQty()`, `ComputeLRMomenta()`, `ComputeResiduals()`, `ComputeSpeQty()`, `ComputeStarRad()`, `ComputeStarTheta()`, `ComputeVelocities()`, `CreatePolarGrid()`, `DiffusionCoefs()`, `EtaPressureSupport()`, `FillVtheta()`, `ImposeKeplerianEdges()`, `InitComputeAccel()`, `IterateRelaxationParameter()`, `MidplaneVolumeDensity()`, `MultiplyPolarGridbyConstant()`, `OpacityProfile()`, `ParametricAccretion()`, `PebbleStokesNumbers()`, `RefillSigma()`, `SubStep1()`, `SubStep2()`, `SuccessiveOverrelaxation()`, `SynchronizePebbleDisc()`, `TemperatureGradient()`, `UpdateDivVelocAndStressTensor()`, and `UpdateVelocityWithViscousTerms()`.

3.4.2.4 `int polargrid::Nsec`

Azimuthal size of the grid, in number of zones.

Definition at line 39 of file `types.h`.

Referenced by `AccreteOntoPlanets()`, `AccretePebblesOntoPlanets()`, `ActualizeQbalance()`, `BckpFieldsForBC()`, `CalculateFlaring()`, `ConditionCFL()`, `CreatePolarGrid()`, `CriticalCharTime()`, `DampPebbles()`, `DiffusionCoefs()`, `GasMomentum()`, `GasTotalMass()`, `InitComputeAccel()`, `IterateRelaxationParameter()`, `OpacityProfile()`, `ParametricAccretion()`, `SubStep2()`, `SuccessiveOverrelaxation()`, `TemperatureGradient()`, and `ThicknessSmoothing()`.

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

- [types.h](#)

3.5 timeprocess Struct Reference

This structure is used for monitoring CPU time usage.

```
#include <types.h>
```

Data Fields

- char [name](#) [80]
- clock_t [clicks](#)

3.5.1 Detailed Description

This structure is used for monitoring CPU time usage.

It is used only if -t is specified on the command line.

Definition at line 86 of file types.h.

3.5.2 Field Documentation

3.5.2.1 clock_t timeprocess::clicks

Definition at line 88 of file types.h.

Referenced by GiveSpecificTime(), and InitSpecificTime().

3.5.2.2 char timeprocess::name[80]

Definition at line 87 of file types.h.

Referenced by GiveSpecificTime(), and InitSpecificTime().

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

- [types.h](#)

Chapter 4

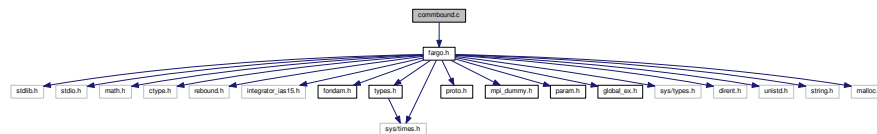
File Documentation

4.1 commbound.c File Reference

Contains the functions used to synchronize buffer zones on all processes.

```
#include "fargo.h"
```

Include dependency graph for commbound.c:



Functions

- void [AllocateComm](#) ()
- void [CommunicateBoundaries](#) ([PolarGrid](#) *Density, [PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta, [PolarGrid](#) *Energy, [PolarGrid](#) *Label)

Variables

- static [real](#) * [SendInnerBoundary](#)
- static [real](#) * [SendOuterBoundary](#)
- static [real](#) * [RecvInnerBoundary](#)
- static [real](#) * [RecvOuterBoundary](#)
- static int [allocated_com](#) = 0
- static int [size_com](#)

4.1.1 Detailed Description

Contains the functions used to synchronize buffer zones on all processes.

In addition to the main function that allows the synchronization (note that even processes first send their inner zones to the previous process, then receive their inner buffer from this process, while odd processes first receive their outer buffer from the next process, then send their outer zones to the next process. This file also contains the function that allocates the memory for the communication (once for all the run).

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [commbound.c](#).

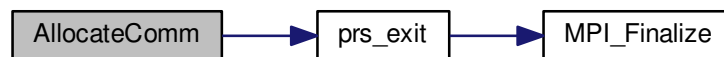
4.1.2 Function Documentation**4.1.2.1 void AllocateComm ()**

Definition at line 28 of file [commbound.c](#).

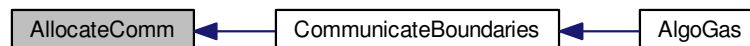
References [AdvecteLabel](#), [allocated_com](#), [CPU_Rank](#), [CPUOVERLAP](#), [EnergyEq](#), [NSEC](#), [prs_exit\(\)](#), [RecvInnerBoundary](#), [RecvOuterBoundary](#), [SendInnerBoundary](#), [SendOuterBoundary](#), [size_com](#), and [YES](#).

Referenced by [CommunicateBoundaries\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:

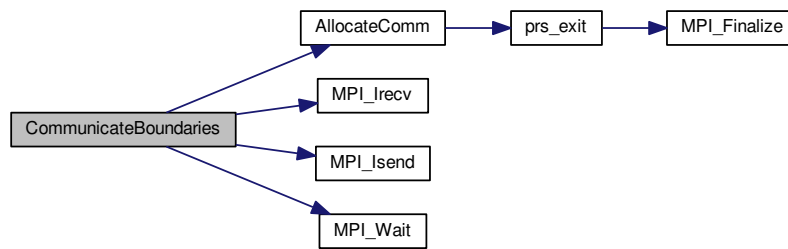
**4.1.2.2 void CommunicateBoundaries (PolarGrid * Density, PolarGrid * Vrad, PolarGrid * Vtheta, PolarGrid * Energy, PolarGrid * Label)**

Definition at line 47 of file [commbound.c](#).

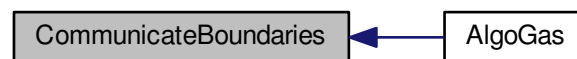
References [AdvecteLabel](#), [AllocateComm\(\)](#), [allocated_com](#), [CPU_Number](#), [CPU_Rank](#), [CPUOVERLAP](#), [EnergyEq](#), [fargostat](#), [MPI_COMM_WORLD](#), [MPI_DOUBLE](#), [MPI_Irecv\(\)](#), [MPI_Isend\(\)](#), [MPI_Wait\(\)](#), [NSEC](#), [RecvInnerBoundary](#), [RecvOuterBoundary](#), [SendInnerBoundary](#), [SendOuterBoundary](#), [size_com](#), and [YES](#).

Referenced by [AlgoGas\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3 Variable Documentation

4.1.3.1 `int allocated_com = 0` [static]

Definition at line 25 of file `commbound.c`.

Referenced by `AllocateComm()`, and `CommunicateBoundaries()`.

4.1.3.2 `real* RecvInnerBoundary` [static]

Definition at line 22 of file `commbound.c`.

Referenced by `AllocateComm()`, and `CommunicateBoundaries()`.

4.1.3.3 `real* RecvOuterBoundary` [static]

Definition at line 23 of file `commbound.c`.

Referenced by `AllocateComm()`, and `CommunicateBoundaries()`.

4.1.3.4 `real* SendInnerBoundary` [static]

Definition at line 20 of file `commbound.c`.

Referenced by `AllocateComm()`, and `CommunicateBoundaries()`.

4.1.3.5 `real* SendOuterBoundary` `[static]`

Definition at line 21 of file `commbound.c`.

Referenced by `AllocateComm()`, and `CommunicateBoundaries()`.

4.1.3.6 `int size_com` `[static]`

Definition at line 26 of file `commbound.c`.

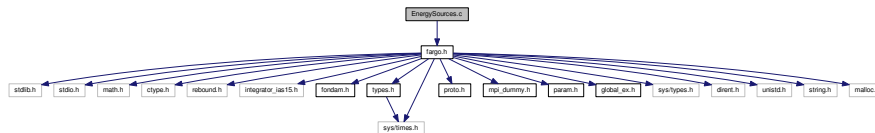
Referenced by `AllocateComm()`, and `CommunicateBoundaries()`.

4.2 EnergySources.c File Reference

Subroutines related to the heating/cooling source terms, numerical solver for the energy equation and radiative diffusion.

```
#include "fargo.h"
```

Include dependency graph for `EnergySources.c`:



Macros

- `#define SORMAXITERS` 1000
- `#define SOREPS` 1.0e-8

Functions

- void `InitRadiatorDiffusionFields` ()
Initialises the polar arrays associated with the heating/cooling processes.
- void `CalculateQminus` (`PolarGrid` *Rho)
Estimate of the heat loss due to radiation escape in the vertical direction with respect to the midplane.
- void `CalculateQirr` (`PolarGrid` *Rho)
Calculates the stellar irradiation source term.
- void `CalculateFlaring` ()
Calculates the sine of the grazing angle by reconstructing the surface from the pressure scale height.
- void `ImplicitRadiativeDiffusion` (`PolarGrid` *Rho, `PolarGrid` *EnergyInt, `PolarGrid` *EnergyNew, `real` dt)
The main numerical solver of the energy equation.
- void `IterateRelaxationParameter` ()
When solving the energy equation for the first time, the function spans through various values of the SOR parameter in order to find its best value to start with.
- int `SuccessiveOverrelaxation` (`real` omega, `boolean` errcheck)
The SOR method algorithm inspired by the one from Numerical Recipes.
- void `ChessBoardIndexing` ()
Function ensures the odd-even ordering of the SOR method when the grid is split on multiple CPUs.
- void `DiffusionCoefs` ()

Calculation of the diffusion coefficients.

- void `TemperatureGradient` ()
Finds the temperature gradients and their magnitude over the mesh.
- void `MidplaneVolumeDensity` (`PolarGrid` *Rho)
Translates the surface density into the midplane volume density using the local pressure scale height.
- void `OpacityProfile` ()
Fills the opacity polar grid, either with a fixed parametric value or using the Bell & Lin (1994) opacity table.
- `real` `FluxLimiterValue` (`real` s)
Calculates the flux limiter according to Kley (1989)
- `real` `EffectiveOpticalDepth` (`real` tau)
Calculates the effective optical depth in a simple gray model of the disk's vertical structure; see Hubeny (1990).
- void `SynchronizeOverlapFields` (`real` *field, int nr, int nsync)
For a MPI-split grid, synchronizes the values in a requested number 'nsync' of the overlapping radial rings.
- void `CreateTorqueMapInfile` (int istep, `PolarGrid` *Surfdens)
Writes an input file for the 'torquemap' code written by Bertram Bitsch.

Variables

- static `real` `kappa0` [7] = {2.0e-4, 2.0e16, 0.1, 2.0e81, 1.0e-8, 1.0e-36, 1.5e20}
- static `real` `a` [7] = {0.0, 0.0, 0.0, 1.0, 2.0/3.0, 1.0/3.0, 1.0}
- static `real` `b` [7] = {2.0, -7.0, 0.5, -24.0, 3.0, 10.0, -5.0/2.0}
- static `PolarGrid` * `GradTemperRad`
- static `PolarGrid` * `GradTemperTheta`
- static `PolarGrid` * `GradTemperMagnitude`
- static `PolarGrid` * `DiffCoefCentered`
- static `PolarGrid` * `DiffCoeflfaceRad`
- static `PolarGrid` * `DiffCoeflfaceTheta`
- static `PolarGrid` * `Opacity`
- static `PolarGrid` * `VolumeDensity`
- static `PolarGrid` * `Flaring`
- static `PolarGrid` * `Qirradiation`
- static `PolarGrid` * `DiscretizationCoefA`
- static `PolarGrid` * `DiscretizationCoefB`
- static `PolarGrid` * `MatrixNexttoTemperl`
- static `PolarGrid` * `MatrixNexttoTemperljp`
- static `PolarGrid` * `MatrixNexttoTemperljm`
- static `PolarGrid` * `MatrixNexttoTemperljp`
- static `PolarGrid` * `MatrixNexttoTemperljm`
- static `PolarGrid` * `RightHandSide`
- static `real` `CV`
- static `real` `omegabest`
- static `real` `domega`
- static int `Niterbest`
- static int `jchess1st`
- static int `jchess2nd`

4.2.1 Detailed Description

Subroutines related to the heating/cooling source terms, numerical solver for the energy equation and radiative diffusion.

Author

Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz

Calculates the individual energy source terms according to Chrenko et al. (2017). Then solves the energy equation in a linearised implicit form using the successive over-relaxation (SOR) method (see Appendix A in Chrenko et al. 2017).

4.2.2 LICENSE

Copyright (c) 2017 Ondřej Chrenko. See the LICENSE file of the distribution.

Definition in file [EnergySources.c](#).

4.2.3 Macro Definition Documentation

4.2.3.1 #define SOREPS 1.0e-8

Definition at line 23 of file EnergySources.c.

Referenced by SuccessiveOverrelaxation().

4.2.3.2 #define SORMAXITERS 1000

Definition at line 22 of file EnergySources.c.

Referenced by IterateRelaxationParameter(), and SuccessiveOverrelaxation().

4.2.4 Function Documentation

4.2.4.1 void CalculateFlaring ()

Calculates the sine of the grazing angle by reconstructing the surface from the pressure scale height.

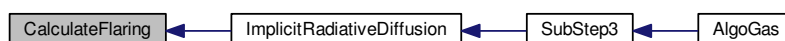
See Eq. (15) in Chrenko et al. (2017).

Definition at line 147 of file EnergySources.c.

References AU_SI, polargrid::Field, InvDiffRsup, InvRmed, polargrid::Nrad, polargrid::Nsec, OmegaInv, Rinf, Rmed, Rsup, SoundSpeed, SQRT_ADIABIND_INV, and STELLARRADIUS.

Referenced by ImplicitRadiativeDiffusion().

Here is the caller graph for this function:



4.2.4.2 void CalculateQirr (PolarGrid * Rho)

Calculates the stellar irradiation source term.

See Eq. (13) in Chrenko et al. (2017).

Definition at line 104 of file EnergySources.c.

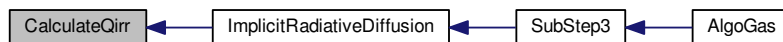
References AU_SI, DISCALBEDO, EffectiveOpticalDepth(), EFFECTIVETEMPERATURE, polargrid::Field, OPA↔CITYDROP, Rmed2, STEFANBOLTZMANN, STELLARRADIUS, and T2SI.

Referenced by ImplicitRadiativeDiffusion().

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.3 void CalculateQminus (PolarGrid * Rho)

Estimate of the heat loss due to radiation escape in the vertical direction with respect to the midplane.

See Eq. (9) in Chrenko et al. (2017).

Definition at line 76 of file EnergySources.c.

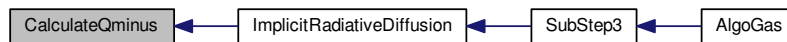
References EffectiveOpticalDepth(), polargrid::Field, polargrid::Nrad, OPACITYDROP, Qminus, STEFANBOLTZ↔MANN, and Temperature.

Referenced by ImplicitRadiativeDiffusion().

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.4 void ChessBoardIndexing ()

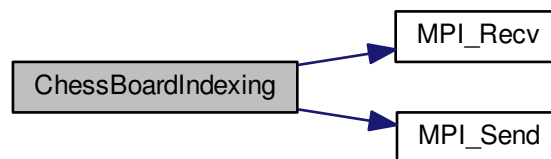
Function ensures the odd-even ordering of the SOR method when the grid is split on multiple CPUs.

Definition at line 486 of file EnergySources.c.

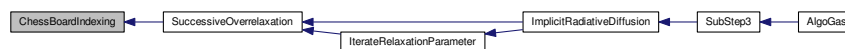
References CPU_Master, CPU_Number, CPU_Rank, CPUOVERLAP, fargostat, jchess1st, jchess2nd, MPI_CO↔MM_WORLD, MPI_INT, MPI_Recv(), MPI_Send(), and NRAD.

Referenced by SuccessiveOverrelaxation().

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.5 void CreateTorqueMapInfile (int istep, PolarGrid * Surfdens)

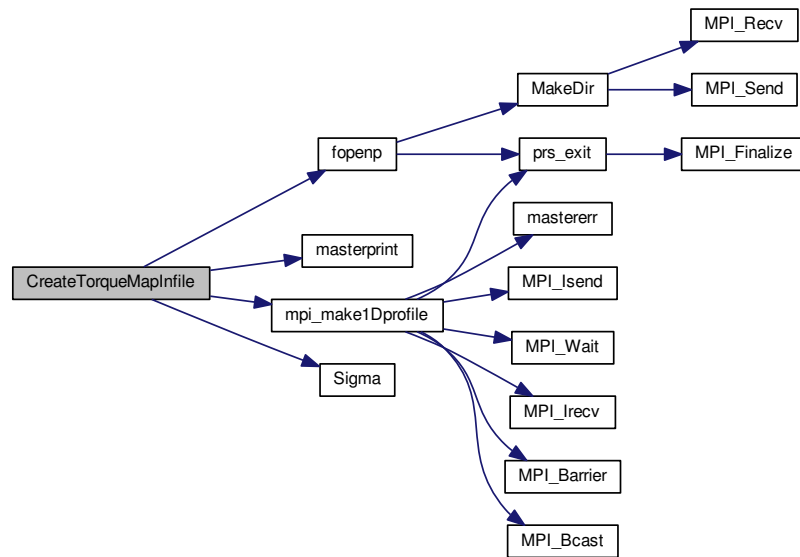
Writes an input file for the 'torquemap' code written by Bertram Bitsch.

Definition at line 816 of file EnergySources.c.

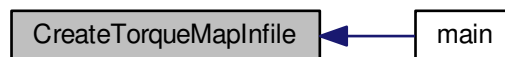
References a, ADIABIND, ALPHAVISOSITY, b, CPU_Master, polargrid::Field, fopen(), GLOBALNRAD, Global↔Rmed, kappa0, masterprint(), MAX1D, mpi_make1Dprofile(), Opacity, OUTPUTDIR, PARAMETRICOPACITY, PI, PRESS2CGS, Pressure, RHO2CGS, Sigma(), SIGMA2CGS, SoundSpeed, T2SI, Temperature, VISCOSITY, and ViscosityAlpha.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.6 void DiffusionCoefs ()

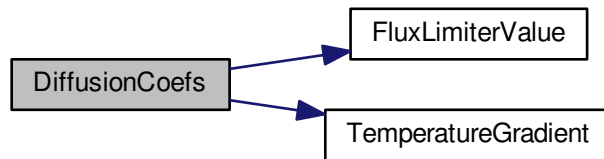
Calculation of the diffusion coefficients.

Definition at line 527 of file `EnergySources.c`.

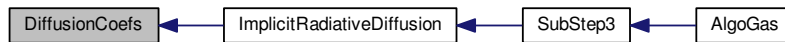
References `polargrid::Field`, `FluxLimiterValue()`, `polargrid::Nrad`, `polargrid::Nsec`, `STEFANBOLTZMANN`, `Temperature`, and `TemperatureGradient()`.

Referenced by `ImplicitRadiativeDiffusion()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.7 `real EffectiveOpticalDepth (real tau)`

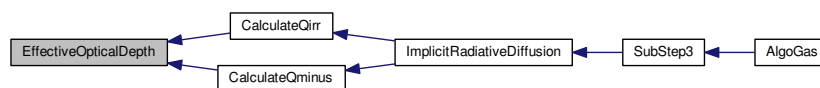
Calculates the effective optical depth in a simple gray model of the disk's vertical structure; see Hubeny (1990).

Definition at line 736 of file `EnergySources.c`.

References `StellarIrradiation`.

Referenced by `CalculateQirr()`, and `CalculateQminus()`.

Here is the caller graph for this function:



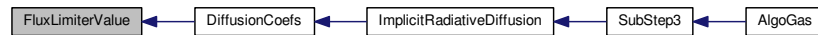
4.2.4.8 `real FluxLimiterValue (real s)`

Calculates the flux limiter according to Kley (1989)

Definition at line 719 of file `EnergySources.c`.

Referenced by `DiffusionCoefs()`.

Here is the caller graph for this function:



4.2.4.9 void ImplicitRadiativeDiffusion (PolarGrid * Rho, PolarGrid * EnergyInt, PolarGrid * EnergyNew, real dt)

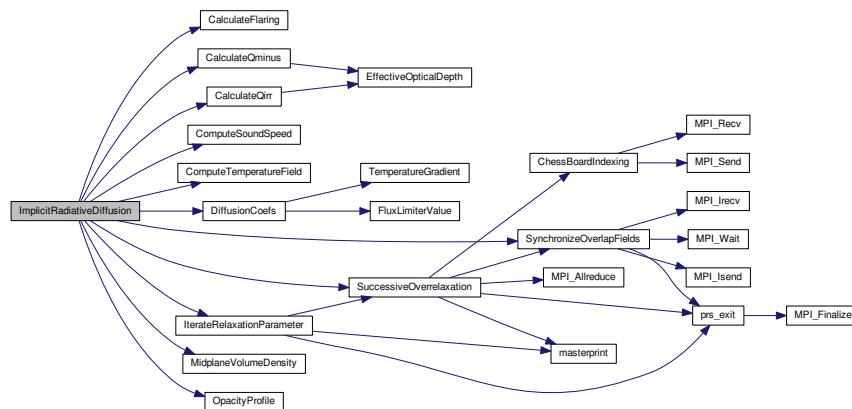
The main numerical solver of the energy equation.

Definition at line 202 of file EnergySources.c.

References AccretHeating, ADIABIND, CalculateFlaring(), CalculateQirr(), CalculateQminus(), ComputeSoundSpeed(), ComputeTemperatureField(), CPU_Number, CPU_Rank, CPUOVERLAP, CV, DiffusionCoefs(), DivergenceVelocity, domega, polargrid::Field, heatsrc, heatsrc_index, heatsrc_max, InvDiffRmed, InvDiffRsup, IterateRelaxationParameter(), MaxMO_or_active, MidplaneVolumeDensity(), Niterbest, NO, omegabest, OmegaInv, One_or_active, OpacityProfile(), PI, Qminus, Qplus, Rinf, Rmed, SoundSpeed, SQRT_ADIABIND_INV, StellarIrradiation, SuccessiveOverrelaxation(), SynchronizeOverlapFields(), Temperature, and YES.

Referenced by SubStep3().

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.10 void InitRadiatDiffusionFields ()

Initialises the polar arrays associated with the heating/cooling processes.

Definition at line 46 of file EnergySources.c.

References ADIABIND, CreatePolarGrid(), CV, GASCONST, MOLWEIGHT, NRAD, NSEC, Qbalance, Qminus, and Write_Qbalance.

Referenced by InitEuler().

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.11 void IterateRelaxationParameter ()

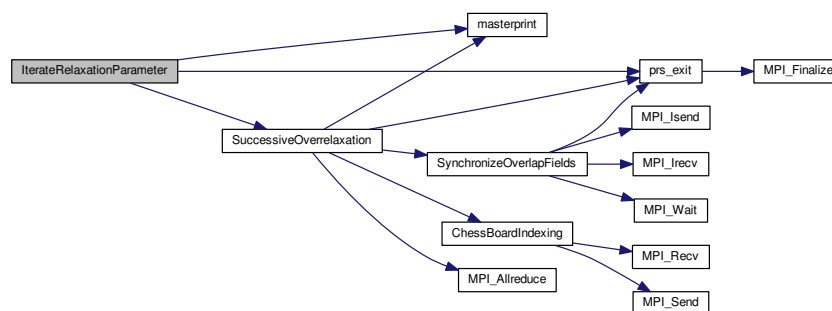
When solving the energy equation for the first time, the function spans through various values of the SOR parameter in order to find its best value to start with.

Definition at line 338 of file EnergySources.c.

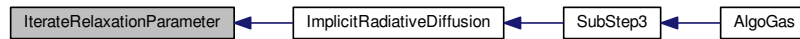
References domega, polargrid::Field, masterprint(), Niterbest, NO, polargrid::Nrad, polargrid::Nsec, omegabest, prs_exit(), SORMAXITERS, SuccessiveOverrelaxation(), Temperature, and YES.

Referenced by ImplicitRadiativeDiffusion().

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.12 void MidplaneVolumeDensity (PolarGrid * Rho)

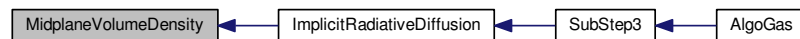
Translates the surface density into the midplane volume density using the local pressure scale height.

Definition at line 629 of file EnergySources.c.

References polargrid::Field, polargrid::Nrad, OmegaInv, SoundSpeed, SQRT2PI_INV, and SQRT_ADIABIND_INV.

Referenced by ImplicitRadiativeDiffusion().

Here is the caller graph for this function:



4.2.4.13 void OpacityProfile ()

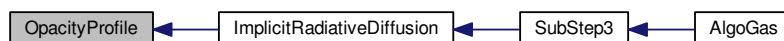
Fills the opacity polar grid, either with a fixed parametric value or using the Bell & Lin (1994) opacity table.

Definition at line 656 of file EnergySources.c.

References a, b, polargrid::Field, kappa0, NO, polargrid::Nrad, polargrid::Nsec, OPA2CU, PARAMETRICOPACITY, RHO2CGS, T2SI, Temperature, and YES.

Referenced by ImplicitRadiativeDiffusion().

Here is the caller graph for this function:



4.2.4.14 int SuccessiveOverrelaxation (real omega, boolean errcheck)

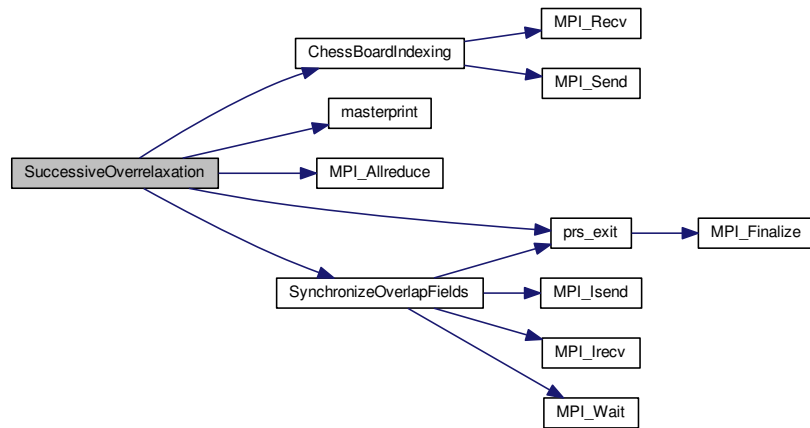
The SOR method algorithm inspired by the one from Numerical Recipes.

Definition at line 396 of file EnergySources.c.

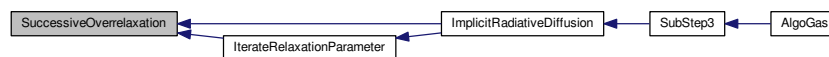
References ChessBoardIndexing(), CPU_Number, polargrid::Field, jchess1st, jchess2nd, masterprint(), MaxMO_↔ or_active, MPI_Allreduce(), MPI_COMM_WORLD, MPI_DOUBLE, MPI_SUM, NO, polargrid::Nrad, polargrid::Nsec, One_or_active, prs_exit(), SOREPS, SORMAXITERS, SynchronizeOverlapFields(), Temperature, and YES.

Referenced by `ImplicitRadiativeDiffusion()`, and `IterateRelaxationParameter()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.15 void SynchronizeOverlapFields (real * field, int nr, int nsync)

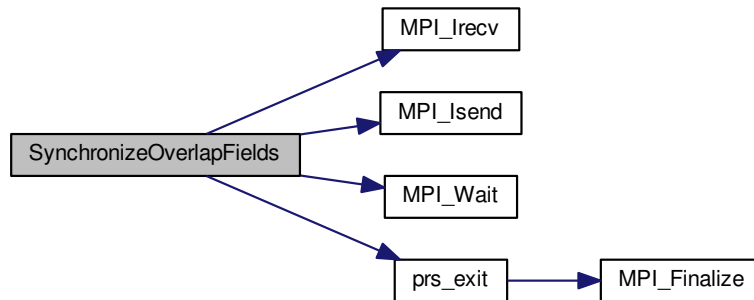
For a MPI-split grid, synchronizes the values in a requested number 'nsync' of the overlapping radial rings.

Definition at line 751 of file `EnergySources.c`.

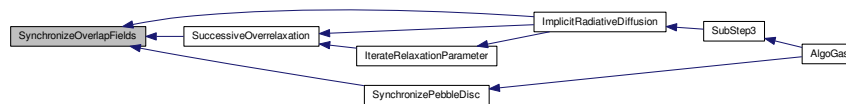
References `CPU_Number`, `CPU_Rank`, `CPUOVERLAP`, `fargostat`, `MPI_COMM_WORLD`, `MPI_DOUBLE`, `MPI_Irecv()`, `MPI_Isend()`, `MPI_Wait()`, `NO`, `NSEC`, `prs_exit()`, and `YES`.

Referenced by `ImplicitRadiativeDiffusion()`, `SuccessiveOverrelaxation()`, and `SynchronizePebbleDisc()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.4.16 void TemperatureGradient ()

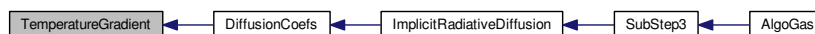
Finds the temperature gradients and their magnitude over the mesh.

Definition at line 580 of file `EnergySources.c`.

References `polargrid::Field`, `InvDiffRmed`, `polargrid::Nrad`, `polargrid::Nsec`, `PI`, `Rmed`, and `Temperature`.

Referenced by `DiffusionCoefs()`.

Here is the caller graph for this function:



4.2.5 Variable Documentation

4.2.5.1 real a[7] = {0.0, 0.0, 0.0, 1.0, 2.0/3.0, 1.0/3.0, 1.0} [static]

Definition at line 27 of file `EnergySources.c`.

Referenced by `CreateTorqueMapInfile()`, `FindOrbitalElements()`, `max2()`, `min2()`, `OpacityProfile()`, `SetupReboundSimulation()`, and `Trapzd()`.

4.2.5.2 **real** `b[7]` = {2.0, -7.0, 0.5, -24.0, 3.0, 10.0, -5.0/2.0} `[static]`

Definition at line 28 of file EnergySources.c.

Referenced by CreateTorqueMapInfile(), max2(), min2(), and OpacityProfile().

4.2.5.3 **real** `CV` `[static]`

Definition at line 39 of file EnergySources.c.

Referenced by ImplicitRadiativeDiffusion(), and InitRadiatDiffusionFields().

4.2.5.4 **PolarGrid*** `DiffCoefCentered` `[static]`

Definition at line 31 of file EnergySources.c.

4.2.5.5 **PolarGrid *** `DiffCoefffaceRad` `[static]`

Definition at line 31 of file EnergySources.c.

4.2.5.6 **PolarGrid *** `DiffCoefffaceTheta` `[static]`

Definition at line 31 of file EnergySources.c.

4.2.5.7 **PolarGrid*** `DiscretizationCoefA` `[static]`

Definition at line 35 of file EnergySources.c.

4.2.5.8 **PolarGrid *** `DiscretizationCoefB` `[static]`

Definition at line 35 of file EnergySources.c.

4.2.5.9 **real** `omega` `[static]`

Definition at line 40 of file EnergySources.c.

Referenced by AlgoGas(), ImplicitRadiativeDiffusion(), and IterateRelaxationParameter().

4.2.5.10 **PolarGrid*** `Flaring` `[static]`

Definition at line 33 of file EnergySources.c.

4.2.5.11 **PolarGrid *** `GradTemperMagnitude` `[static]`

Definition at line 30 of file EnergySources.c.

4.2.5.12 **PolarGrid*** `GradTemperRad` `[static]`

Definition at line 30 of file EnergySources.c.

4.2.5.13 PolarGrid * GradTemperTheta [static]

Definition at line 30 of file EnergySources.c.

4.2.5.14 int jchess1st [static]

Definition at line 42 of file EnergySources.c.

Referenced by ChessBoardIndexing(), and SuccessiveOverrelaxation().

4.2.5.15 int jchess2nd [static]

Definition at line 42 of file EnergySources.c.

Referenced by ChessBoardIndexing(), and SuccessiveOverrelaxation().

4.2.5.16 real kappa0[7] = {2.0e-4, 2.0e16, 0.1, 2.0e81, 1.0e-8, 1.0e-36, 1.5e20} [static]

Definition at line 26 of file EnergySources.c.

Referenced by CreateTorqueMapInfile(), and OpacityProfile().

4.2.5.17 PolarGrid* MatrixNexttoTemperl [static]

Definition at line 36 of file EnergySources.c.

4.2.5.18 PolarGrid * MatrixNexttoTemperlim [static]

Definition at line 36 of file EnergySources.c.

4.2.5.19 PolarGrid * MatrixNexttoTemperlip [static]

Definition at line 36 of file EnergySources.c.

4.2.5.20 PolarGrid * MatrixNexttoTemperljm [static]

Definition at line 37 of file EnergySources.c.

4.2.5.21 PolarGrid* MatrixNexttoTemperljp [static]

Definition at line 37 of file EnergySources.c.

4.2.5.22 int Niterbest [static]

Definition at line 41 of file EnergySources.c.

Referenced by ImplicitRadiativeDiffusion(), and IterateRelaxationParameter().

4.2.5.23 real omegabest [static]

Definition at line 40 of file EnergySources.c.

Referenced by ImplicitRadiativeDiffusion(), and IterateRelaxationParameter().

4.3.1 Detailed Description

Contains all the include directives requested by the code.

In addition, it contains a preprocessor test to know whether we should build a sequential or MPI executable.

Author

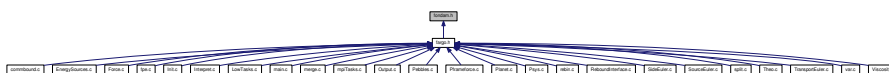
THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [fargo.h](#).

4.4 fondam.h File Reference

Contains fundamental constants used thorough the code.

This graph shows which files directly or indirectly include this file:



Macros

- `#define G 1.0`
- `#define PI 3.14159265358979323844`
- `#define CPUOVERLAP 5` /* Zeus-like overlap kernel. 2:transport; 2: source, 1:viscous stress */
- `#define GASCONST 1.0`
- `#define MOLWEIGHT 1.0`
- `#define SQRT2PI_INV (1.0/sqrt(2.0*PI))`
- `#define fac1o15 (1.0/15.0)`
- `#define fac1o21 (1.0/21.0)`
- `#define fac4o75 (4.0/75.0)`
- `#define MSOL_SI 1.98855e30`
- `#define G_SI 6.674e-11`
- `#define GM_SI 1.32712440018e20`
- `#define AU_SI 149597870700.0` /* changing this can modify basic length unit */
- `#define R_STANDARD 8.3144598`
- `#define MMW 2.4`
- `#define R_SI (R_STANDARD/(MMW*0.001))` /* R_specific = R/mu = 8.3144598 (standard gas constant) / (2.4 (fiducial mol.weight in PPDs) * 1g/mol) = 8.3144598/(2.4*0.001) in SI */
- `#define OPA2CU (MSOL_SI*1000.0/pow(AU_SI*100.0,2.0))` /* (kappa in cgs cm^2/g) * OPA2CU = (kappa in code units) */
- `#define T2SI (GM_SI/R_SI/AU_SI)` /* (T in code units) * T2K = (T in Kelvins ~ cgs or SI) */
- `#define TIME2SI (sqrt(pow(AU_SI,3.0)/GM_SI))` /* (time in code units) * TIME2S = (time in seconds ~ cgs or SI) */
- `#define RHO2CGS (MSOL_SI*1000.0/pow(AU_SI*100.0,3.0))` /* (VOLUME!!! density in code units) * RH↵O2CGS = (volume density in g/cm^3 cgs) */
- `#define STEFANBOLTZMANN (5.670367e-8*(1.0/MSOL_SI)*pow(1.0/TIME2SI,-3.0)*pow(1.0/T2SI,-4.0))`
- `#define STEFANBOLTZMANNCONTROL (5.670367e-5*pow(1.0/(R_SI*10000.0),4.0)*pow(1.0/(G_S↵l*1000.0),-2.5)*pow(1.0/(MSOL_SI*1000),-1.5)*pow(1.0/(AU_SI*100.0),-0.5))`
- `#define MOLDIAMETER 2.72` /* value in angstroms */
- `#define ANGSTR_CGS 0.00000001` /* angstrom in centimeters */

- `#define AMU_CGS 1.660538921e-24 /* atomic mass unit in grams */`
- `#define MOLCROSSEC_CGS 2.0e-15 /* molecular cross section of H2 in cm^2 */`
- `#define SURFDENS2CGS (MSOL_SI*1000.0/pow(AU_SI*100.0,2.0)) /* (surface density in code units) * SURFDENS2CGS = (surface density in g/cm^2 cgs) */`
- `#define SIGMA2CGS (MSOL_SI*1000.0/pow(AU_SI*100.0,2.0))`
- `#define PRESS2CGS (MSOL_SI*1000.0/(AU_SI*100.0*TIME2SI*TIME2SI))`
- `#define FLUX2CU (0.000003/(2.0*PI)) /* (radial flux in earth masses per year) * FLUX2CU = (radial flux in code units) */`

4.4.1 Detailed Description

Contains fundamental constants used thorough the code.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [fondam.h](#).

4.4.2 Macro Definition Documentation

4.4.2.1 `#define AMU_CGS 1.660538921e-24 /* atomic mass unit in grams */`

Definition at line 47 of file [fondam.h](#).

4.4.2.2 `#define ANGSTR_CGS 0.00000001 /* angstrom in centimeters */`

Definition at line 46 of file [fondam.h](#).

4.4.2.3 `#define AU_SI 149597870700.0 /* changing this can modify basic length unit */`

Definition at line 30 of file [fondam.h](#).

Referenced by [CalculateFlaring\(\)](#), and [CalculateQirr\(\)](#).

4.4.2.4 `#define CPUOVERLAP 5 /* Zeus-like overlap kernel. 2:transport; 2: source, 1:viscous stress */`

Definition at line 13 of file [fondam.h](#).

Referenced by [AllocateComm\(\)](#), [ChessBoardIndexing\(\)](#), [CommunicateBoundaries\(\)](#), [ImplicitRadiativeDiffusion\(\)](#), [SplitDomain\(\)](#), [SynchronizeOverlapFields\(\)](#), [SynchronizePebbleDisc\(\)](#), and [WriteDiskPolar\(\)](#).

4.4.2.5 `#define fac1o15 (1.0/15.0)`

Definition at line 24 of file [fondam.h](#).

4.4.2.6 `#define fac1o21 (1.0/21.0)`

Definition at line 25 of file [fondam.h](#).

4.4.2.7 #define fac4o75 (4.0/75.0)

Definition at line 26 of file fondam.h.

4.4.2.8 #define FLUX2CU (0.000003/(2.0*PI)) /* (radial flux in earth masses per year) * FLUX2CU = (radial flux in code units) */

Definition at line 54 of file fondam.h.

Referenced by InitPebblesViaFlux().

4.4.2.9 #define G 1.0

Definition at line 11 of file fondam.h.

Referenced by ComputeSoundSpeed(), FindOrbitalElements(), GetPsysInfo(), GetPsysInfoFromRsim(), Impose↔ KeplerianEdges(), InitGasVelocity(), OutputElements(), SetupReboundSimulation(), TellEverything(), and TellNb↔ Orbits().

4.4.2.10 #define G_SI 6.674e-11

Definition at line 28 of file fondam.h.

4.4.2.11 #define GASCONST 1.0

Definition at line 17 of file fondam.h.

Referenced by ComputeTemperatureField(), Energy(), and InitRadiatDiffusionFields().

4.4.2.12 #define GM_SI 1.32712440018e20

Definition at line 29 of file fondam.h.

4.4.2.13 #define MMW 2.4

Definition at line 32 of file fondam.h.

4.4.2.14 #define MOLCROSSEC_CGS 2.0e-15 /* molecular cross section of H2 in cm^2 */

Definition at line 48 of file fondam.h.

4.4.2.15 #define MOLDIAMETER 2.72 /* value in angstroms */

Definition at line 45 of file fondam.h.

4.4.2.16 #define MOLWEIGHT 1.0

Definition at line 18 of file fondam.h.

Referenced by ComputeTemperatureField(), Energy(), and InitRadiatDiffusionFields().

4.4.2.17 #define MSOL_SI 1.98855e30

Definition at line 27 of file fondam.h.

4.4.2.18 `#define OPA2CU (MSOL_SI*1000.0/pow(AU_SI*100.0,2.0)) /* (kappa in cgs cm2/g) * OPA2CU = (kappa in code units) */`

Definition at line 35 of file fondam.h.

Referenced by OpacityProfile().

4.4.2.19 `#define PI 3.14159265358979323844`

Definition at line 12 of file fondam.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), ComputeResiduals(), ComputeStarTheta(), ConditionCFL(), CreateTorqueMapInfile(), CriticalCharTime(), DampingTW04(), FillPolar1DArrays(), ImplicitRadiativeDiffusion(), InitComputeAccel(), InitPebblesViaFlux(), NonReflectingBoundary(), ParametricAccretion(), ParticleDiffusion(), SetupReboundSimulation(), SetWaveKillingZones(), SubStep1(), SubStep2(), TellEverything(), TellNbOrbits(), TemperatureGradient(), ThicknessSmoothing(), and VanLeerRadial().

4.4.2.20 `#define PRESS2CGS (MSOL_SI*1000.0/(AU_SI*100.0*TIME2SI*TIME2SI))`

Definition at line 53 of file fondam.h.

Referenced by CreateTorqueMapInfile().

4.4.2.21 `#define R_SI (R_STANDARD/(MMW*0.001)) /* R_specific = R/mu = 8.3144598 (standard gas constant) / (2.4 (fiducial mol.weight in PPDs) * 1g/mol) = 8.3144598/(2.4*0.001) in SI */`

Definition at line 33 of file fondam.h.

4.4.2.22 `#define R_STANDARD 8.3144598`

Definition at line 31 of file fondam.h.

4.4.2.23 `#define RHO2CGS (MSOL_SI*1000.0/pow(AU_SI*100.0,3.0)) /* (VOLUME!!! density in code units) * RHO2CGS = (volume density in g/cm3 cgs) */`

Definition at line 38 of file fondam.h.

Referenced by AccretePebblesOntoPlanets(), CreateTorqueMapInfile(), InitPebbleArrays(), OpacityProfile(), ParametricAccretion(), and SetupReboundSimulation().

4.4.2.24 `#define SIGMA2CGS (MSOL_SI*1000.0/pow(AU_SI*100.0,2.0))`

Definition at line 52 of file fondam.h.

Referenced by CreateTorqueMapInfile().

4.4.2.25 `#define SQRT2PI_INV (1.0/sqrt(2.0*PI))`

Definition at line 20 of file fondam.h.

Referenced by AccretePebblesOntoPlanets(), and MidplaneVolumeDensity().

4.4.2.26 `#define STEFANBOLTZMANN (5.670367e-8*(1.0/MSOL_SI)*pow(1.0/TIME2SI,-3.0)*pow(1.0/T2SI,-4.0))`

Definition at line 39 of file fondam.h.

Referenced by `CalculateQirr()`, `CalculateQminus()`, and `DiffusionCoefs()`.

4.4.2.27 `#define STEFANBOLTZMANNCONTROL (5.670367e-5*pow(1.0/(R_SI*10000.0),4.0)*pow(1.0/(G_SI*1000.0),-2.5)*pow(1.0/(MSOL_SI*1000),-1.5)*pow(1.0/(AU_SI*100.0),-0.5))`

Definition at line 42 of file `fondam.h`.

4.4.2.28 `#define SURFDENS2CGS (MSOL_SI*1000.0/pow(AU_SI*100.0,2.0)) /* (surface density in code units) * SURFDENS2CGS = (surface density in g/cm^2 cgs) */`

Definition at line 49 of file `fondam.h`.

4.4.2.29 `#define T2SI (GM_SI/R_SI/AU_SI) /* (T in code units) * T2K = (T in Kelvins ~ cgs or SI) */`

Definition at line 36 of file `fondam.h`.

Referenced by `CalculateQirr()`, `CreateTorqueMapInfile()`, and `OpacityProfile()`.

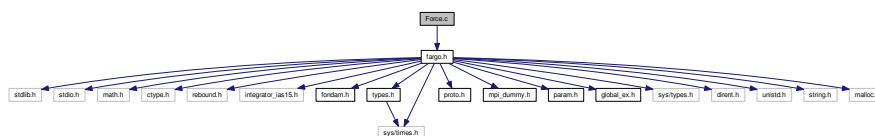
4.4.2.30 `#define TIME2SI (sqrt(pow(AU_SI,3.0)/GM_SI)) /* (time in code units) * TIME2S = (time in seconds ~ cgs or SI) */`

Definition at line 37 of file `fondam.h`.

4.5 Force.c File Reference

Contains the function to evaluate and write the disk torques acting on planets and also the function to get the thickness smoothing parameter.

```
#include "fargo.h"
Include dependency graph for Force.c:
```



Functions

- void `UpdateLog` (`PolarGrid` *Rho, `PlanetarySystem` *psys)
Calculates and writes the disk torques (both specific and normalized) acting on the planets.
- real `ThicknessSmoothing` (real x, real y)
Computes the local thickness from the sound speed and applies the thickness smoothing parameter.

Variables

- boolean `OpenInner`
- boolean `NonReflecting`

4.5.1 Detailed Description

Contains the function to evaluate and write the disk torques acting on planets and also the function to get the thickness smoothing parameter.

Author

Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz

The original ComputeForce() function was discarded, the torque computation follows from the formalism of the vertical averaging which directly provides the planet accelerations. The normalised torque is now part of the output. [UpdateLog\(\)](#) is called from [AdvanceSystemFromDisk\(\)](#) when needed.

4.5.2 LICENSE

Copyright (c) 2017 Ondřej Chrenko. See the LICENSE file of the distribution.

Definition in file [Force.c](#).

4.5.3 Function Documentation

4.5.3.1 `real ThicknessSmoothing (real x, real y)`

Computes the local thickness from the sound speed and applies the thickness smoothing parameter.

Definition at line 75 of file Force.c.

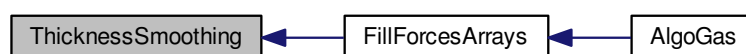
References `polargrid::Field`, `Max_or_active`, `MPI_Allreduce()`, `MPI_COMM_WORLD`, `MPI_DOUBLE`, `MPI_MAX`, `polargrid::Nsec`, `OmegaInv`, `PI`, `Rinf`, `Rsup`, `SoundSpeed`, `SQRT_ADIABIND_INV`, `THICKNESSSMOOTHING`, and `Zero_or_active`.

Referenced by `FillForcesArrays()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.3.2 void UpdateLog (PolarGrid * Rho, PlanetarySystem * psys)

Calculates and writes the disk torques (both specific and normalized) acting on the planets.

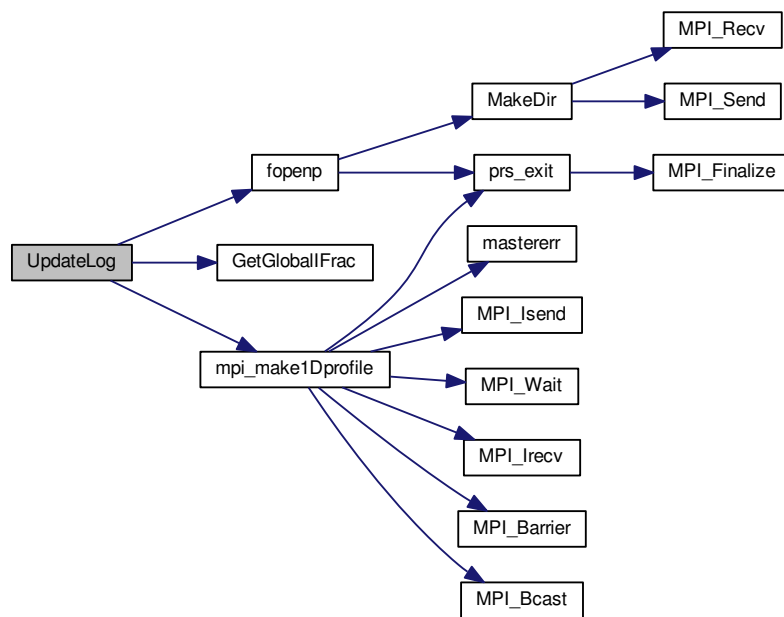
Uses the accelerations provided by the vertical averaging approach.

Definition at line 27 of file Force.c.

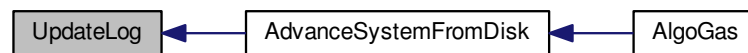
References ADIABIND, planetary_system::ax, planetary_system::ay, CPU_Number, CPU_Rank, polargrid::Field, fopenp(), GetGlobalIFrac(), planetary_system::mass, MAX1D, mpi_make1Dprofile(), planetary_system::nb, OUT←PUTDIR, PhysicalTime, SoundSpeed, SQRT_ADIABIND_INV, planetary_system::x, and planetary_system::y.

Referenced by AdvanceSystemFromDisk().

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.4 Variable Documentation

4.5.4.1 boolean NonReflecting

Definition at line 30 of file Interpret.c.

4.5.4.2 boolean OpenInner

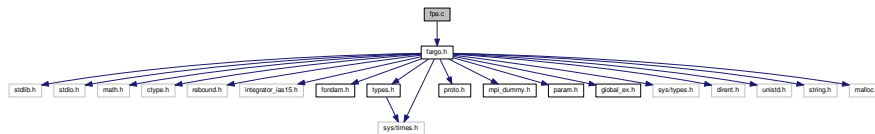
Definition at line 14 of file main.c.

4.6 fpe.c File Reference

Contains two functions that specify how we handle floating point exceptions.

```
#include "fargo.h"
```

Include dependency graph for fpe.c:



Functions

- void [handfpe](#) ()
- void [setfpe](#) ()

4.6.1 Detailed Description

Contains two functions that specify how we handle floating point exceptions.

Definition in file [fpe.c](#).

4.6.2 Function Documentation

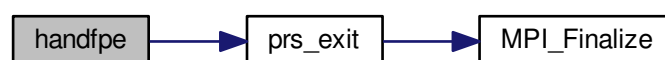
4.6.2.1 void handfpe ()

Definition at line 8 of file fpe.c.

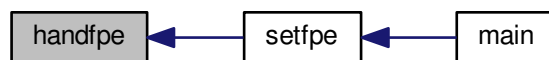
References `CPU_Rank`, and `pr_exit()`.

Referenced by `setfpe()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.6.2.2 void setfpe ()

Definition at line 15 of file fpe.c.

References handfpe().

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



4.7 global.h File Reference

Declares all global variables.

Variables

- int [CPU_Rank](#)
- int [CPU_Number](#)
- boolean [CPU_Master](#)
- int [IMIN](#)
- int [IMAX](#)

- int Zero_or_active
- int Max_or_active
- int One_or_active
- int MaxMO_or_active
- int GLOBALNRAD
- real Rinf [MAX1D]
- real Rsup [MAX1D]
- real Rmed [MAX1D]
- real Surf [MAX1D]
- real InvRmed [MAX1D]
- real InvSurf [MAX1D]
- real InvDiffRmed [MAX1D]
- real InvDiffRsup [MAX1D]
- real InvRinf [MAX1D]
- real Radii [MAX1D]
- real GlobalRmed [MAX1D]
- real SigmaMed [MAX1D]
- real SigmaInf [MAX1D]
- real MassTaper
- real OmegaInv [MAX1D]
- real Rmed2 [MAX1D]
- real EnergyMed [MAX1D]
- real globpressvec [MAX1D]
- real globcsvec [MAX1D]
- real WaveKiller [MAX1D]
- real VthetaMed [MAX1D]
- real QplusMed [MAX1D]
- real CoolingTimeMed [MAX1D]
- real PebDensInit [MAX1D]
- real PebVradInit [MAX1D]
- real PebVthetaInit [MAX1D]
- real vt1D [MAX1D]
- real invdtpeb_sq
- real invdtreb_sq
- real SQRT_ADIABIND_INV
- real OmegaFrame
- real PhysicalTime =0.0
- real PhysicalTimeInitial
- real heatsrc [MAXPLANETS]
- int heatsrc_max
- int TimeStep =0
- boolean EnergyEq
- boolean StoreEnergy
- boolean ParametricCooling
- boolean Damping
- boolean DampVrad
- boolean DampInit
- boolean StellarIrradiation
- boolean InitFromFile
- boolean Write_Temperature
- boolean Write_Energy
- boolean Write_Divergence
- boolean Write_Qplus
- boolean Write_Qbalance
- boolean Collisions

- [boolean WriteTorque](#)
- [boolean WriteTorqueMapFile](#)
- [boolean MonitorNPL](#)
- [boolean FeelDisk](#)
- [boolean Pebbles](#)
- [boolean Write_Eta](#)
- [boolean AccretHeating](#)
- [boolean BackReaction](#)
- [boolean ActualizeLuminosity](#)
- [boolean DiffusiveParticles](#)
- [boolean PrescribedAccretion](#)
- [boolean heatsrc_index \[MAXPLANETS\]](#)
- [boolean TorqueDensity](#)
- [boolean Merge](#)
- [boolean AdvecteLabel](#)
- [boolean FakeSequential](#)
- [boolean MonitorIntegral](#)
- [boolean debug](#)
- [boolean OnlyInit](#)
- [boolean GotoNextOutput](#)
- [boolean StoreSigma](#)
- [boolean ViscosityAlpha](#)
- [boolean RocheSmoothing](#)
- [boolean CentrifugalBalance](#)
- [boolean ExcludeHill](#)
- [boolean SloppyCFL](#)
- [MPI_Status fargostat](#)
- [PolarGrid * CellAbscissa](#)
- [PolarGrid * CellOrdinate](#)
- [PolarGrid * RhoStar](#)
- [PolarGrid * RhoInt](#)
- [PolarGrid * Temperature](#)
- [PolarGrid * Pressure](#)
- [PolarGrid * SoundSpeed](#)
- [PolarGrid * Qplus](#)
- [PolarGrid * Qminus](#)
- [PolarGrid * Qbalance](#)
- [PolarGrid * DivergenceVelocity](#)
- [PolarGrid * TAURR](#)
- [PolarGrid * TAURP](#)
- [PolarGrid * TAUPP](#)
- [PolarGrid * GasAccelrad](#)
- [PolarGrid * GasAcceltheta](#)
- [PolarGrid * DragForceRad](#)
- [PolarGrid * DragForceTheta](#)
- [PolarGrid * PebbleDens](#)
- [PolarGrid * PebbleVrad](#)
- [PolarGrid * PebbleVtheta](#)
- [PolarGrid * StokesNumber](#)
- [PolarGrid * GravAccelRad](#)
- [PolarGrid * GravAccelTheta](#)
- [PolarGrid * PebbleGravAccelRad](#)
- [PolarGrid * PebbleGravAccelTheta](#)
- [PolarGrid * Torque](#)
- [boolean LogGrid](#)

- [boolean OverridesOutputdir](#)
- [char NewOutputdir \[1024\]](#)
- [FILE * plout](#)
- [FILE * discard](#)
- [FILE * mergers](#)

4.7.1 Detailed Description

Declares all global variables.

Used to construct automatically the file [global_ex.h](#). The file [global.h](#) cannot contain any comment, as it would not be parsed correctly by `varparser.pl`

Definition in file [global.h](#).

4.7.2 Variable Documentation

4.7.2.1 **boolean AccreteHeating**

Definition at line 27 of file [global.h](#).

Referenced by [AccretePebblesOntoPlanets\(\)](#), [ImplicitRadiativeDiffusion\(\)](#), [ParametricAccretion\(\)](#), and [ReadVariables\(\)](#).

4.7.2.2 **boolean ActualizeLuminosity**

Definition at line 27 of file [global.h](#).

4.7.2.3 **boolean AdvecteLabel**

Definition at line 29 of file [global.h](#).

Referenced by [AllocateComm\(\)](#), [CommunicateBoundaries\(\)](#), [merge\(\)](#), [OneWindRad\(\)](#), [OneWindTheta\(\)](#), [QuantitiesAdvection\(\)](#), [ReadVariables\(\)](#), [SendOutput\(\)](#), [TellEverything\(\)](#), and [Transport\(\)](#).

4.7.2.4 **boolean BackReaction**

Definition at line 27 of file [global.h](#).

Referenced by [ReadVariables\(\)](#), [SourceTermsPebbles\(\)](#), and [SubStep1\(\)](#).

4.7.2.5 **PolarGrid* CellAbscissa**

Definition at line 33 of file [global.h](#).

Referenced by [AccreteOntoPlanets\(\)](#), [AccretePebblesOntoPlanets\(\)](#), [FillForcesArrays\(\)](#), and [InitComputeAccel\(\)](#).

4.7.2.6 **PolarGrid * CellOrdinate**

Definition at line 33 of file [global.h](#).

Referenced by [AccreteOntoPlanets\(\)](#), [AccretePebblesOntoPlanets\(\)](#), [FillForcesArrays\(\)](#), and [InitComputeAccel\(\)](#).

4.7.2.7 boolean CentrifugalBalance

Definition at line 31 of file global.h.

Referenced by InitGasVelocity(), and main().

4.7.2.8 boolean Collisions

Definition at line 26 of file global.h.

Referenced by main(), ReadVariables(), RestartReboundSimulation(), SetupIntegratorParams(), and SetupReboundSimulation().

4.7.2.9 real CoolingTimeMed[MAX1D]

Definition at line 17 of file global.h.

Referenced by FillCoolingTime(), and SubStep3().

4.7.2.10 boolean CPU_Master

Definition at line 3 of file global.h.

Referenced by AdvanceSystemFromDisk(), CheckRebin(), ChessBoardIndexing(), CreateTorqueMapInfile(), DiscardParticlesDist(), DumpOmegaFrame(), DumpSources(), EmptyPlanetSystemFile(), FillPolar1DArrays(), InitPebblesViaFlux(), InitPlanetarySystem(), ListPlanets(), main(), mastererr(), masterprint(), merge(), OutputNbodySimulation(), ReadPrevDim(), ResolveCollisions(), RestartReboundSimulation(), SendOutput(), TellEverything(), WriteBigPlanetFile(), WriteDim(), WriteDiskPolar(), and WritePlanetFile().

4.7.2.11 int CPU_Number

Definition at line 2 of file global.h.

Referenced by AdvanceSystemFromDisk(), ApplyOuterSourceMass(), ChessBoardIndexing(), CommunicateBoundaries(), DampingTW04(), FindOrbitalElements(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), ImplicitRadiativeDiffusion(), ImposeKeplerianEdges(), InitPebblesViaFlux(), main(), MakeDir(), merge(), mpi_make1Dprofile(), NonReflectingBoundary(), OutputElements(), ReadfromAsciiFile(), ReadfromFile(), SendOutput(), SplitDomain(), SuccessiveOverrelaxation(), SynchronizeOverlapFields(), SynchronizePebbleDisc(), UpdateLog(), and WriteDiskPolar().

4.7.2.12 int CPU_Rank

Definition at line 1 of file global.h.

Referenced by AllocateComm(), ApplyOuterSourceMass(), ChessBoardIndexing(), CommunicateBoundaries(), ConditionCFL(), FindOrbitalElements(), fopenp(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), handfpe(), ImplicitRadiativeDiffusion(), ImposeKeplerianEdges(), main(), MakeDir(), mpi_make1Dprofile(), NonReflectingBoundary(), OpenBoundary(), OutputElements(), ReadfromAsciiFile(), ReadfromFile(), SplitDomain(), SynchronizeOverlapFields(), UpdateLog(), and WriteDiskPolar().

4.7.2.13 boolean Damping

Definition at line 24 of file global.h.

Referenced by ApplyBoundaryCondition(), InitEuler(), ReadVariables(), and SubStep1().

4.7.2.14 boolean DampInIt

Definition at line 24 of file global.h.

Referenced by DampingBoundary(), InitEuler(), Initialization(), and ReadVariables().

4.7.2.15 boolean DampVrad

Definition at line 24 of file global.h.

Referenced by ReadVariables().

4.7.2.16 boolean debug

Definition at line 29 of file global.h.

Referenced by ConditionCFL(), main(), and SplitDomain().

4.7.2.17 boolean DiffusiveParticles

Definition at line 27 of file global.h.

Referenced by AlgoGas(), and ReadVariables().

4.7.2.18 FILE * discard

Definition at line 44 of file global.h.

Referenced by DiscardParticlesDist(), main(), RestartReboundSimulation(), and SetupReboundSimulation().

4.7.2.19 PolarGrid * DivergenceVelocity

Definition at line 36 of file global.h.

Referenced by ImplicitRadiativeDiffusion(), InitViscosity(), SendOutput(), SubStep3(), and UpdateDivVelocAndStressTensor().

4.7.2.20 PolarGrid * DragForceRad

Definition at line 37 of file global.h.

Referenced by InitPebbleArrays(), SourceTermsPebbles(), and SubStep1().

4.7.2.21 PolarGrid * DragForceTheta

Definition at line 37 of file global.h.

Referenced by InitPebbleArrays(), SourceTermsPebbles(), and SubStep1().

4.7.2.22 boolean EnergyEq

Definition at line 24 of file global.h.

Referenced by AlgoGas(), AllocateComm(), ApplyBoundaryCondition(), CommunicateBoundaries(), ComputePressureField(), ComputeSoundSpeed(), ComputeTemperatureField(), InitEuler(), Initialization(), main(), OneWindRad(), OneWindTheta(), QuantitiesAdvection(), ReadVariables(), SubStep2(), and TellEverything().

4.7.2.23 real EnergyMed[MAX1D]

Definition at line 16 of file global.h.

Referenced by DampingBoundary(), FillEnergy(), InitGasDensityEnergy(), NonReflectingBoundary(), RefillEnergy(), and SubStep3().

4.7.2.24 boolean ExcludeHill

Definition at line 31 of file global.h.

Referenced by FillForcesArrays(), and ReadVariables().

4.7.2.25 boolean FakeSequential

Definition at line 29 of file global.h.

Referenced by GasMomentum(), GasTotalEnergy(), GasTotalMass(), and main().

4.7.2.26 MPI_Status fargostat

Definition at line 32 of file global.h.

Referenced by ChessBoardIndexing(), CommunicateBoundaries(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), MakeDir(), mpi_make1Dprofile(), ReadfromAsciiFile(), ReadfromFile(), and SynchronizeOverlapFields().

4.7.2.27 boolean FeelDisk

Definition at line 26 of file global.h.

Referenced by ReadVariables(), RestartReboundSimulation(), and SetupReboundSimulation().

4.7.2.28 PolarGrid* GasAccelrad

Definition at line 37 of file global.h.

Referenced by InitPebbleArrays(), SubStep1(), and SubStep1Pebbles().

4.7.2.29 PolarGrid * GasAcceltheta

Definition at line 37 of file global.h.

Referenced by InitPebbleArrays(), SubStep1(), and SubStep1Pebbles().

4.7.2.30 int GLOBALNRAD

Definition at line 10 of file global.h.

Referenced by CheckRebin(), CreateTorqueMapInfile(), FillPolar1DArrays(), GetGlobalIFrac(), InitGasVelocity(), mpi_make1Dprofile(), SplitDomain(), and WriteDim().

4.7.2.31 real GlobalRmed[MAX1D]

Definition at line 13 of file global.h.

Referenced by CheckRebin(), CreateTorqueMapInfile(), FillPolar1DArrays(), FViscosity(), GetGlobalIFrac(), and InitGasVelocity().

4.7.2.32 **real globcsvec[**MAX1D**]**

Definition at line 16 of file global.h.

Referenced by ComputeSoundSpeed(), and FViscosity().

4.7.2.33 **real globpressvec[**MAX1D**]**

Definition at line 16 of file global.h.

Referenced by InitGasVelocity().

4.7.2.34 **boolean GotoNextOutput**

Definition at line 30 of file global.h.

Referenced by main().

4.7.2.35 **PolarGrid* GravAccelRad**

Definition at line 39 of file global.h.

Referenced by FillForcesArrays(), InitEuler(), and SubStep1().

4.7.2.36 **PolarGrid * GravAccelTheta**

Definition at line 39 of file global.h.

Referenced by FillForcesArrays(), InitEuler(), and SubStep1().

4.7.2.37 **real heatsrc[**MAXPLANETS**]**

Definition at line 21 of file global.h.

Referenced by AccretePebblesOntoPlanets(), ImplicitRadiativeDiffusion(), and ParametricAccretion().

4.7.2.38 **boolean heatsrc_index[**MAXPLANETS**]**

Definition at line 28 of file global.h.

Referenced by AccretePebblesOntoPlanets(), ImplicitRadiativeDiffusion(), and ParametricAccretion().

4.7.2.39 **int heatsrc_max**

Definition at line 22 of file global.h.

Referenced by AccretePebblesOntoPlanets(), ImplicitRadiativeDiffusion(), and ParametricAccretion().

4.7.2.40 **int IMAX**

Definition at line 5 of file global.h.

Referenced by SplitDomain().

4.7.2.41 int IMIN

Definition at line 4 of file global.h.

Referenced by FillPolar1DArrays(), InitGasVelocity(), mpi_make1Dprofile(), ReadfromAsciiFile(), ReadfromFile(), and SplitDomain().

4.7.2.42 boolean InitFromFile

Definition at line 25 of file global.h.

Referenced by Initialization(), and ReadVariables().

4.7.2.43 real InvDiffRmed[MAX1D]

Definition at line 12 of file global.h.

Referenced by ComputeStarRad(), FillPolar1DArrays(), ImplicitRadiativeDiffusion(), ParticleDiffusion(), SubStep1(), SubStep2(), TemperatureGradient(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.7.2.44 real InvDiffRsup[MAX1D]

Definition at line 13 of file global.h.

Referenced by CalculateFlaring(), FillPolar1DArrays(), ImplicitRadiativeDiffusion(), SubStep2(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.7.2.45 real invdtpeb_sq

Definition at line 19 of file global.h.

Referenced by ConditionCFL(), and CriticalCharTime().

4.7.2.46 real invdtreb_sq

Definition at line 19 of file global.h.

Referenced by ConditionCFL(), and MinStepForRebound().

4.7.2.47 real InvRinf[MAX1D]

Definition at line 13 of file global.h.

Referenced by FillPolar1DArrays(), SourceTermsPebbles(), SubStep1(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.7.2.48 real InvRmed[MAX1D]

Definition at line 12 of file global.h.

Referenced by AccretePebblesOntoPlanets(), CalculateFlaring(), ComputeResiduals(), ConditionCFL(), FillForcesArrays(), FillPolar1DArrays(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.7.2.49 real InvSurf[MAX1D]

Definition at line 12 of file global.h.

Referenced by `FillPolar1DArrays()`, and `VanLeerRadial()`.

4.7.2.50 **boolean** LogGrid

Definition at line 41 of file `global.h`.

Referenced by `FillPolar1DArrays()`, and `ReadVariables()`.

4.7.2.51 **real** MassTaper

Definition at line 14 of file `global.h`.

Referenced by `AlgoGas()`, and `FillForcesArrays()`.

4.7.2.52 **int** Max_or_active

Definition at line 7 of file `global.h`.

Referenced by `AccreteOntoPlanets()`, `AccretePebblesOntoPlanets()`, `ConditionCFL()`, `CriticalCharTime()`, `DampingBoundary()`, `DampingTW04()`, `DampPebbles()`, `FillForcesArrays()`, `GasMomentum()`, `GasTotalEnergy()`, `GasTotalMass()`, `mpi_make1Dprofile()`, `SetWaveKillingZones()`, `SplitDomain()`, and `ThicknessSmoothing()`.

4.7.2.53 **int** MaxMO_or_active

Definition at line 9 of file `global.h`.

Referenced by `ConditionCFL()`, `ImplicitRadiativeDiffusion()`, `SplitDomain()`, and `SuccessiveOverrelaxation()`.

4.7.2.54 **boolean** Merge

Definition at line 29 of file `global.h`.

Referenced by `AdvanceSystemFromDisk()`, `InitPebblesViaFlux()`, `main()`, and `SendOutput()`.

4.7.2.55 **FILE *** mergers

Definition at line 44 of file `global.h`.

Referenced by `main()`, `ResolveCollisions()`, `RestartReboundSimulation()`, and `SetupReboundSimulation()`.

4.7.2.56 **boolean** MonitorIntegral

Definition at line 29 of file `global.h`.

Referenced by `main()`.

4.7.2.57 **boolean** MonitorNPL

Definition at line 26 of file `global.h`.

Referenced by `main()`, and `ReadVariables()`.

4.7.2.58 **char** NewOutputdir[1024]

Definition at line 43 of file `global.h`.

Referenced by `main()`, and `ReadVariables()`.

4.7.2.59 `real OmegaFrame`

Definition at line 20 of file `global.h`.

Referenced by `AccreteOntoPlanets()`, `AccretePebblesOntoPlanets()`, `AlgoGas()`, `BckpFieldsForBC()`, `ComputeLRMomena()`, `ComputeVelocities()`, `DampingBoundary()`, `DampPebbles()`, `DumpOmegaFrame()`, `EtaPressureSupport()`, `FillVtheta()`, `GasMomentum()`, `GasTotalEnergy()`, `ImposeKeplerianEdges()`, `InitGasVelocity()`, `InitPebblesViaFlux()`, `main()`, `SourceTermsPebbles()`, `SubStep1()`, `WriteBigPlanetFile()`, and `WritePlanetFile()`.

4.7.2.60 `real OmegaInv[MAX1D]`

Definition at line 15 of file `global.h`.

Referenced by `AccretePebblesOntoPlanets()`, `CalculateFlaring()`, `FillForcesArrays()`, `FillPolar1DArrays()`, `ImplicitRadiativeDiffusion()`, `MidplaneVolumeDensity()`, `SourceTermsPebbles()`, `SubStep1Pebbles()`, and `ThicknessSmoothing()`.

4.7.2.61 `int One_or_active`

Definition at line 8 of file `global.h`.

Referenced by `ConditionCFL()`, `CriticalCharTime()`, `ImplicitRadiativeDiffusion()`, `SplitDomain()`, and `SuccessiveOverrelaxation()`.

4.7.2.62 `boolean OnlyInit`

Definition at line 29 of file `global.h`.

Referenced by `main()`.

4.7.2.63 `boolean OverridesOutputdir`

Definition at line 42 of file `global.h`.

Referenced by `main()`, and `ReadVariables()`.

4.7.2.64 `boolean ParametricCooling`

Definition at line 24 of file `global.h`.

Referenced by `InitEuler()`, `InitGasVelocity()`, `ReadVariables()`, `SubStep3()`, and `TellEverything()`.

4.7.2.65 `PolarGrid* PebbleDens`

Definition at line 38 of file `global.h`.

Referenced by `AccretePebblesOntoPlanets()`, `BckpFieldsForBC()`, `DetectCrashPebbles()`, `EvolvePebbleDisk()`, `InitPebbleArrays()`, `InitPebblesViaFlux()`, `ParticleDiffusion()`, `RestartPebbleDisk()`, `SourceTermsPebbles()`, `SynchronizePebbleDisc()`, and `WritePebbles()`.

4.7.2.66 `PolarGrid * PebbleGravAccelRad`

Definition at line 39 of file `global.h`.

Referenced by `FillForcesArrays()`, `InitEuler()`, and `SourceTermsPebbles()`.

4.7.2.67 PolarGrid * PebbleGravAccelTheta

Definition at line 39 of file global.h.

Referenced by FillForcesArrays(), InitEuler(), and SourceTermsPebbles().

4.7.2.68 boolean Pebbles

Definition at line 27 of file global.h.

Referenced by AlgoGas(), ConditionCFL(), FillForcesArrays(), InitEuler(), Initialization(), merge(), ReadVariables(), SendOutput(), and SubStep1().

4.7.2.69 PolarGrid * PebbleVrad

Definition at line 38 of file global.h.

Referenced by AccretePebblesOntoPlanets(), BckpFieldsForBC(), CriticalCharTime(), EvolvePebbleDisk(), InitPebbleArrays(), InitPebblesViaFlux(), ParticleDiffusion(), RestartPebbleDisk(), SourceTermsPebbles(), SubStep1Pebbles(), SynchronizePebbleDisc(), and WritePebbles().

4.7.2.70 PolarGrid * PebbleVtheta

Definition at line 38 of file global.h.

Referenced by AccretePebblesOntoPlanets(), BckpFieldsForBC(), CorrectPebblesVtheta(), CriticalCharTime(), EvolvePebbleDisk(), InitPebbleArrays(), InitPebblesViaFlux(), ParticleDiffusion(), RestartPebbleDisk(), SourceTermsPebbles(), SubStep1Pebbles(), SynchronizePebbleDisc(), and WritePebbles().

4.7.2.71 real PebDensInit[MAX1D]

Definition at line 18 of file global.h.

Referenced by BckpFieldsForBC(), DampPebbles(), and InitPebblesViaFlux().

4.7.2.72 real PebVradInit[MAX1D]

Definition at line 18 of file global.h.

Referenced by BckpFieldsForBC(), DampPebbles(), and InitPebblesViaFlux().

4.7.2.73 real PebVthetaInit[MAX1D]

Definition at line 18 of file global.h.

Referenced by BckpFieldsForBC(), DampPebbles(), and InitPebblesViaFlux().

4.7.2.74 real PhysicalTime =0.0

Definition at line 20 of file global.h.

Referenced by AccretePebblesOntoPlanets(), AdvanceSystemRebound(), AlgoGas(), AspectRatio(), DiscardParticlesDist(), FindOrbitalElements(), FViscosity(), main(), OutputElements(), ParametricAccretion(), RestartReboundSimulation(), UpdateLog(), WriteBigPlanetFile(), and WritePlanetFile().

4.7.2.75 real PhysicalTimeInitial

Definition at line 20 of file global.h.

Referenced by AspectRatio(), FViscosity(), and main().

4.7.2.76 FILE* plout

Definition at line 44 of file global.h.

Referenced by main(), OutputElements(), RestartReboundSimulation(), and SetupReboundSimulation().

4.7.2.77 boolean PrescribedAccretion

Definition at line 27 of file global.h.

Referenced by AlgoGas(), and ReadVariables().

4.7.2.78 PolarGrid * Pressure

Definition at line 35 of file global.h.

Referenced by ComputePressureField(), ComputeTemperatureField(), CreateTorqueMapInfile(), InitEuler(), InitGasVelocity(), and SubStep1().

4.7.2.79 PolarGrid * Qbalance

Definition at line 35 of file global.h.

Referenced by ActualizeQbalance(), InitRadiatDiffusionFields(), and SendOutput().

4.7.2.80 PolarGrid * Qminus

Definition at line 35 of file global.h.

Referenced by ActualizeQbalance(), CalculateQminus(), ImplicitRadiativeDiffusion(), and InitRadiatDiffusionFields().

4.7.2.81 PolarGrid * Qplus

Definition at line 35 of file global.h.

Referenced by ActualizeQbalance(), ImplicitRadiativeDiffusion(), InitEuler(), InitQplus(), SendOutput(), and SubStep3().

4.7.2.82 real QplusMed[MAX1D]

Definition at line 17 of file global.h.

Referenced by FillQplus(), and SubStep3().

4.7.2.83 real Radii[MAX1D]

Definition at line 13 of file global.h.

Referenced by CheckRebin(), FillPolar1DArrays(), and InitGasVelocity().

4.7.2.84 **PolarGrid * RhoInt**

Definition at line 34 of file global.h.

Referenced by InitEuler(), OneWindRad(), OneWindRadPebbles(), QuantitiesAdvection(), QuantitiesAdvection↵
Pebbles(), SubStep2(), VanLeerRadial(), and VanLeerTheta().

4.7.2.85 **PolarGrid* RhoStar**

Definition at line 34 of file global.h.

Referenced by InitEuler(), OneWindRad(), OneWindRadPebbles(), QuantitiesAdvection(), QuantitiesAdvection↵
Pebbles(), VanLeerRadial(), and VanLeerTheta().

4.7.2.86 **real Rinf[MAX1D]**

Definition at line 11 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), ApplyOuterSourceMass(), CalculateFlaring(),
ConditionCFL(), CriticalCharTime(), DampingTW04(), FillPolar1DArrays(), FillSigma(), GasTotalEnergy(), Implicit↵
RadiativeDiffusion(), InitGasVelocity(), NonReflectingBoundary(), ParametricAccretion(), ParticleDiffusion(), Refill↵
Sigma(), SourceTermsPebbles(), SubStep1(), ThicknessSmoothing(), UpdateDivVelocAndStressTensor(), Update↵
VelocityWithViscousTerms(), VanLeerRadial(), and VanLeerTheta().

4.7.2.87 **real Rmed[MAX1D]**

Definition at line 11 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), BckpFieldsForBC(), CalculateFlaring(),
ComputeLRMomena(), ComputeResiduals(), ComputeSoundSpeed(), ComputeStarRad(), ComputeStarTheta(),
ComputeVelocities(), ConditionCFL(), CorrectVtheta(), CriticalCharTime(), DampingBoundary(), DampPebbles(),
EtaPressureSupport(), FillCoolingTime(), FillEnergy(), FillForcesArrays(), FillPolar1DArrays(), FillQplus(), Fill↵
Sigma(), FillVtheta(), GasMomentum(), GasTotalEnergy(), ImplicitRadiativeDiffusion(), ImposeKeplerianEdges(),
InitComputeAccel(), InitGasVelocity(), InitLabel(), InitPebblesViaFlux(), NonReflectingBoundary(), Particle↵
Diffusion(), RefillSigma(), SetWaveKillingZones(), SubStep1(), SubStep2(), SubStep3(), TemperatureGradient(),
UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.7.2.88 **real Rmed2[MAX1D]**

Definition at line 15 of file global.h.

Referenced by CalculateQirr(), FillForcesArrays(), and FillPolar1DArrays().

4.7.2.89 **boolean RocheSmoothing**

Definition at line 30 of file global.h.

Referenced by ReadVariables().

4.7.2.90 **real Rsup[MAX1D]**

Definition at line 11 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), CalculateFlaring(), ConditionCFL(), Critical↵
CharTime(), DampingTW04(), FillPolar1DArrays(), GasTotalEnergy(), ParametricAccretion(), Thickness↵
Smoothing(), UpdateDivVelocAndStressTensor(), UpdateVelocityWithViscousTerms(), VanLeerRadial(), and Van↵
LeerTheta().

4.7.2.91 real SigmaInf[*MAX1D*]

Definition at line 14 of file global.h.

Referenced by FillSigma(), InitGasVelocity(), and RefillSigma().

4.7.2.92 real SigmaMed[*MAX1D*]

Definition at line 14 of file global.h.

Referenced by ApplyOuterSourceMass(), DampingBoundary(), FillSigma(), InitGasDensityEnergy(), NonReflectingBoundary(), OpenBoundary(), RefillSigma(), and SubStep3().

4.7.2.93 boolean SloppyCFL

Definition at line 31 of file global.h.

Referenced by AlgoGas(), and main().

4.7.2.94 PolarGrid * SoundSpeed

Definition at line 35 of file global.h.

Referenced by AccretePebblesOntoPlanets(), CalculateFlaring(), ComputePressureField(), ComputeSoundSpeed(), ConditionCFL(), CreateTorqueMapInfile(), DampingTW04(), FillForcesArrays(), ImplicitRadiativeDiffusion(), InitEuler(), InitGasVelocity(), MidplaneVolumeDensity(), NonReflectingBoundary(), ParametricAccretion(), ThicknessSmoothing(), and UpdateLog().

4.7.2.95 real SQRT_ADIABIND_INV

Definition at line 19 of file global.h.

Referenced by AccretePebblesOntoPlanets(), CalculateFlaring(), FillForcesArrays(), ImplicitRadiativeDiffusion(), InitEuler(), MidplaneVolumeDensity(), PebbleStokesNumbers(), ThicknessSmoothing(), and UpdateLog().

4.7.2.96 boolean StellarIrradiation

Definition at line 24 of file global.h.

Referenced by EffectiveOpticalDepth(), ImplicitRadiativeDiffusion(), and ReadVariables().

4.7.2.97 PolarGrid * StokesNumber

Definition at line 38 of file global.h.

Referenced by AccretePebblesOntoPlanets(), FillForcesArrays(), InitPebbleArrays(), InitPebblesViaFlux(), PebbleStokesNumbers(), SourceTermsPebbles(), and SubStep1Pebbles().

4.7.2.98 boolean StoreEnergy

Definition at line 24 of file global.h.

Referenced by main().

4.7.2.99 boolean StoreSigma

Definition at line 30 of file global.h.

Referenced by main().

4.7.2.100 real Surf[MAX1D]

Definition at line 11 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), FillForcesArrays(), FillPolar1DArrays(), Gas↔Momentum(), GasTotalEnergy(), GasTotalMass(), ParametricAccretion(), and VanLeerTheta().

4.7.2.101 PolarGrid * TAUPP

Definition at line 36 of file global.h.

Referenced by InitViscosity(), SubStep3(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscous↔Terms().

4.7.2.102 PolarGrid * TAURP

Definition at line 36 of file global.h.

Referenced by InitViscosity(), SubStep3(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscous↔Terms().

4.7.2.103 PolarGrid * TAURR

Definition at line 36 of file global.h.

Referenced by InitViscosity(), SubStep3(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscous↔Terms().

4.7.2.104 PolarGrid* Temperature

Definition at line 35 of file global.h.

Referenced by AlgoGas(), CalculateQminus(), ComputeTemperatureField(), CreateTorqueMapInfile(), Diffusion↔Coefs(), ImplicitRadiativeDiffusion(), InitEuler(), IterateRelaxationParameter(), OpacityProfile(), SendOutput(), SuccessiveOverrelaxation(), and TemperatureGradient().

4.7.2.105 int TimeStep =0

Definition at line 23 of file global.h.

Referenced by AdvanceSystemFromDisk(), and main().

4.7.2.106 PolarGrid* Torque

Definition at line 40 of file global.h.

Referenced by AdvanceSystemFromDisk(), FillForcesArrays(), and InitEuler().

4.7.2.107 boolean TorqueDensity

Definition at line 28 of file global.h.

Referenced by FillForcesArrays(), InitEuler(), main(), and ReadVariables().

4.7.2.108 boolean ViscosityAlpha

Definition at line 30 of file global.h.

Referenced by CreateTorqueMapInfile(), FViscosity(), InitGasVelocity(), and ReadVariables().

4.7.2.109 real vt1D[MAX1D]

Definition at line 19 of file global.h.

Referenced by AccretePebblesOntoPlanets().

4.7.2.110 real VthetaMed[MAX1D]

Definition at line 16 of file global.h.

Referenced by DampingBoundary(), and FillVtheta().

4.7.2.111 real WaveKiller[MAX1D]

Definition at line 16 of file global.h.

Referenced by DampingBoundary(), DampPebbles(), and SetWaveKillingZones().

4.7.2.112 boolean Write_Divergence

Definition at line 25 of file global.h.

Referenced by merge(), ReadVariables(), and SendOutput().

4.7.2.113 boolean Write_Energy

Definition at line 25 of file global.h.

Referenced by merge(), ReadVariables(), and SendOutput().

4.7.2.114 boolean Write_Eta

Definition at line 27 of file global.h.

Referenced by merge(), ReadVariables(), and WritePebbles().

4.7.2.115 boolean Write_Qbalance

Definition at line 25 of file global.h.

Referenced by InitRadiatDiffusionFields(), merge(), ReadVariables(), and SendOutput().

4.7.2.116 **boolean** Write_Qplus

Definition at line 25 of file global.h.

Referenced by `merge()`, `ReadVariables()`, and `SendOutput()`.

4.7.2.117 **boolean** Write_Temperature

Definition at line 25 of file global.h.

Referenced by `merge()`, `ReadVariables()`, and `SendOutput()`.

4.7.2.118 **boolean** WriteTorque

Definition at line 26 of file global.h.

Referenced by `main()`, and `ReadVariables()`.

4.7.2.119 **boolean** WriteTorqueMapFile

Definition at line 26 of file global.h.

Referenced by `main()`, and `ReadVariables()`.

4.7.2.120 **int** Zero_or_active

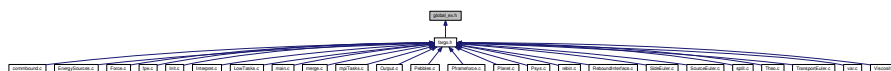
Definition at line 6 of file global.h.

Referenced by `AccreteOntoPlanets()`, `AccretePebblesOntoPlanets()`, `ConditionCFL()`, `DampingBoundary()`, `DampingTW04()`, `DampPebbles()`, `FillForcesArrays()`, `GasMomentum()`, `GasTotalEnergy()`, `GasTotalMass()`, `mpi_↔_make1Dprofile()`, `SetWaveKillingZones()`, `SplitDomain()`, and `ThicknessSmoothing()`.

4.8 global_ex.h File Reference

This file is created automatically during compilation from [global.h](#).

This graph shows which files directly or indirectly include this file:



Variables

- `int` [CPU_Rank](#)
- `int` [CPU_Number](#)
- `boolean` [CPU_Master](#)
- `int` [IMIN](#)
- `int` [IMAX](#)
- `int` [Zero_or_active](#)
- `int` [Max_or_active](#)
- `int` [One_or_active](#)
- `int` [MaxMO_or_active](#)
- `int` [GLOBALNRAD](#)

- real Rinf [MAX1D]
- real Rsup [MAX1D]
- real Rmed [MAX1D]
- real Surf [MAX1D]
- real InvRmed [MAX1D]
- real InvSurf [MAX1D]
- real InvDiffRmed [MAX1D]
- real InvDiffRsup [MAX1D]
- real InvRinf [MAX1D]
- real Radii [MAX1D]
- real GlobalRmed [MAX1D]
- real SigmaMed [MAX1D]
- real SigmaInf [MAX1D]
- real MassTaper
- real OmegaInv [MAX1D]
- real Rmed2 [MAX1D]
- real EnergyMed [MAX1D]
- real globpressvec [MAX1D]
- real globcsvec [MAX1D]
- real WaveKiller [MAX1D]
- real VthetaMed [MAX1D]
- real QplusMed [MAX1D]
- real CoolingTimeMed [MAX1D]
- real PebDensInit [MAX1D]
- real PebVradInit [MAX1D]
- real PebVthetaInit [MAX1D]
- real vt1D [MAX1D]
- real invdtpeb_sq
- real invdtreb_sq
- real SQRT_ADIABIND_INV
- real OmegaFrame
- real PhysicalTime
- real PhysicalTimeInitial
- real heatsrc [MAXPLANETS]
- int heatsrc_max
- int TimeStep
- boolean EnergyEq
- boolean StoreEnergy
- boolean ParametricCooling
- boolean Damping
- boolean DampVrad
- boolean DampInit
- boolean StellarIrradiation
- boolean InitFromFile
- boolean Write_Temperature
- boolean Write_Energy
- boolean Write_Divergence
- boolean Write_Qplus
- boolean Write_Qbalance
- boolean Collisions
- boolean WriteTorque
- boolean WriteTorqueMapFile
- boolean MonitorNPL
- boolean FeelDisk
- boolean Pebbles

- [boolean Write_Eta](#)
- [boolean AccretHeating](#)
- [boolean BackReaction](#)
- [boolean ActualizeLuminosity](#)
- [boolean DiffusiveParticles](#)
- [boolean PrescribedAccretion](#)
- [boolean heatsrc_index](#) [MAXPLANETS]
- [boolean TorqueDensity](#)
- [boolean Merge](#)
- [boolean AdvecteLabel](#)
- [boolean FakeSequential](#)
- [boolean MonitorIntegral](#)
- [boolean debug](#)
- [boolean OnlyInit](#)
- [boolean GotoNextOutput](#)
- [boolean StoreSigma](#)
- [boolean ViscosityAlpha](#)
- [boolean RocheSmoothing](#)
- [boolean CentrifugalBalance](#)
- [boolean ExcludeHill](#)
- [boolean SloppyCFL](#)
- [MPI_Status fargostat](#)
- [PolarGrid * CellAbscissa](#)
- [PolarGrid * CellOrdinate](#)
- [PolarGrid * RhoStar](#)
- [PolarGrid * RhoInt](#)
- [PolarGrid * Temperature](#)
- [PolarGrid * Pressure](#)
- [PolarGrid * SoundSpeed](#)
- [PolarGrid * Qplus](#)
- [PolarGrid * Qminus](#)
- [PolarGrid * Qbalance](#)
- [PolarGrid * DivergenceVelocity](#)
- [PolarGrid * TAURR](#)
- [PolarGrid * TAURP](#)
- [PolarGrid * TAUPP](#)
- [PolarGrid * GasAccelrad](#)
- [PolarGrid * GasAcceltheta](#)
- [PolarGrid * DragForceRad](#)
- [PolarGrid * DragForceTheta](#)
- [PolarGrid * PebbleDens](#)
- [PolarGrid * PebbleVrad](#)
- [PolarGrid * PebbleVtheta](#)
- [PolarGrid * StokesNumber](#)
- [PolarGrid * GravAccelRad](#)
- [PolarGrid * GravAccelTheta](#)
- [PolarGrid * PebbleGravAccelRad](#)
- [PolarGrid * PebbleGravAccelTheta](#)
- [PolarGrid * Torque](#)
- [boolean LogGrid](#)
- [boolean OverridesOutputdir](#)
- [char NewOutputdir](#) [1024]
- [FILE * plout](#)
- [FILE * discard](#)
- [FILE * mergers](#)

4.8.1 Detailed Description

This file is created automatically during compilation from [global.h](#).

Do not edit. See perl script "varparser.pl" for details

Definition in file [global_ex.h](#).

4.8.2 Variable Documentation

4.8.2.1 boolean AccreteHeating

Definition at line 27 of file global.h.

Referenced by `AccretePebblesOntoPlanets()`, `ImplicitRadiativeDiffusion()`, `ParametricAccretion()`, and `ReadVariables()`.

4.8.2.2 boolean ActualizeLuminosity

Definition at line 27 of file global.h.

4.8.2.3 boolean AdvecteLabel

Definition at line 29 of file global.h.

Referenced by `AllocateComm()`, `CommunicateBoundaries()`, `merge()`, `OneWindRad()`, `OneWindTheta()`, `QuantitiesAdvection()`, `ReadVariables()`, `SendOutput()`, `TellEverything()`, and `Transport()`.

4.8.2.4 boolean BackReaction

Definition at line 27 of file global.h.

Referenced by `ReadVariables()`, `SourceTermsPebbles()`, and `SubStep1()`.

4.8.2.5 PolarGrid* CellAbscissa

Definition at line 33 of file global.h.

Referenced by `AccreteOntoPlanets()`, `AccretePebblesOntoPlanets()`, `FillForcesArrays()`, and `InitComputeAccel()`.

4.8.2.6 PolarGrid * CellOrdinate

Definition at line 33 of file global.h.

Referenced by `AccreteOntoPlanets()`, `AccretePebblesOntoPlanets()`, `FillForcesArrays()`, and `InitComputeAccel()`.

4.8.2.7 boolean CentrifugalBalance

Definition at line 31 of file global.h.

Referenced by `InitGasVelocity()`, and `main()`.

4.8.2.8 boolean Collisions

Definition at line 26 of file global.h.

Referenced by `main()`, `ReadVariables()`, `RestartReboundSimulation()`, `SetupIntegratorParams()`, and `SetupReboundSimulation()`.

4.8.2.9 `real CoolingTimeMed[MAX1D]`

Definition at line 17 of file `global.h`.

Referenced by `FillCoolingTime()`, and `SubStep3()`.

4.8.2.10 `boolean CPU_Master`

Definition at line 3 of file `global.h`.

Referenced by `AdvanceSystemFromDisk()`, `CheckRebin()`, `ChessBoardIndexing()`, `CreateTorqueMapInfile()`, `DiscardParticlesDist()`, `DumpOmegaFrame()`, `DumpSources()`, `EmptyPlanetSystemFile()`, `FillPolar1DArrays()`, `InitPebblesViaFlux()`, `InitPlanetarySystem()`, `ListPlanets()`, `main()`, `mastererr()`, `masterprint()`, `merge()`, `OutputNbodySimulation()`, `ReadPrevDim()`, `ResolveCollisions()`, `RestartReboundSimulation()`, `SendOutput()`, `TellEverything()`, `WriteBigPlanetFile()`, `WriteDim()`, `WriteDiskPolar()`, and `WritePlanetFile()`.

4.8.2.11 `int CPU_Number`

Definition at line 2 of file `global.h`.

Referenced by `AdvanceSystemFromDisk()`, `ApplyOuterSourceMass()`, `ChessBoardIndexing()`, `CommunicateBoundaries()`, `DampingTW04()`, `FindOrbitalElements()`, `GasMomentum()`, `GasTotalEnergy()`, `GasTotalMass()`, `ImplicitRadiativeDiffusion()`, `ImposeKeplerianEdges()`, `InitPebblesViaFlux()`, `main()`, `MakeDir()`, `merge()`, `mpi_make1Dprofile()`, `NonReflectingBoundary()`, `OutputElements()`, `ReadfromAsciiFile()`, `ReadfromFile()`, `SendOutput()`, `SplitDomain()`, `SuccessiveOverrelaxation()`, `SynchronizeOverlapFields()`, `SynchronizePebbleDisc()`, `UpdateLog()`, and `WriteDiskPolar()`.

4.8.2.12 `int CPU_Rank`

Definition at line 1 of file `global.h`.

Referenced by `AllocateComm()`, `ApplyOuterSourceMass()`, `ChessBoardIndexing()`, `CommunicateBoundaries()`, `ConditionCFL()`, `FindOrbitalElements()`, `fopenp()`, `GasMomentum()`, `GasTotalEnergy()`, `GasTotalMass()`, `handfpe()`, `ImplicitRadiativeDiffusion()`, `ImposeKeplerianEdges()`, `main()`, `MakeDir()`, `mpi_make1Dprofile()`, `NonReflectingBoundary()`, `OpenBoundary()`, `OutputElements()`, `ReadfromAsciiFile()`, `ReadfromFile()`, `SplitDomain()`, `SynchronizeOverlapFields()`, `UpdateLog()`, and `WriteDiskPolar()`.

4.8.2.13 `boolean Damping`

Definition at line 24 of file `global.h`.

Referenced by `ApplyBoundaryCondition()`, `InitEuler()`, `ReadVariables()`, and `SubStep1()`.

4.8.2.14 `boolean DampInnit`

Definition at line 24 of file `global.h`.

Referenced by `DampingBoundary()`, `InitEuler()`, `Initialization()`, and `ReadVariables()`.

4.8.2.15 `boolean DampVrad`

Definition at line 24 of file `global.h`.

Referenced by `ReadVariables()`.

4.8.2.16 boolean debug

Definition at line 29 of file global.h.

Referenced by ConditionCFL(), main(), and SplitDomain().

4.8.2.17 boolean DiffusiveParticles

Definition at line 27 of file global.h.

Referenced by AlgoGas(), and ReadVariables().

4.8.2.18 FILE * discard

Definition at line 44 of file global.h.

Referenced by DiscardParticlesDist(), main(), RestartReboundSimulation(), and SetupReboundSimulation().

4.8.2.19 PolarGrid* DivergenceVelocity

Definition at line 36 of file global.h.

Referenced by ImplicitRadiativeDiffusion(), InitViscosity(), SendOutput(), SubStep3(), and UpdateDivVelocAndStressTensor().

4.8.2.20 PolarGrid * DragForceRad

Definition at line 37 of file global.h.

Referenced by InitPebbleArrays(), SourceTermsPebbles(), and SubStep1().

4.8.2.21 PolarGrid * DragForceTheta

Definition at line 37 of file global.h.

Referenced by InitPebbleArrays(), SourceTermsPebbles(), and SubStep1().

4.8.2.22 boolean EnergyEq

Definition at line 24 of file global.h.

Referenced by AlgoGas(), AllocateComm(), ApplyBoundaryCondition(), CommunicateBoundaries(), ComputePressureField(), ComputeSoundSpeed(), ComputeTemperatureField(), InitEuler(), Initialization(), main(), OneWindRad(), OneWindTheta(), QuantitiesAdvection(), ReadVariables(), SubStep2(), and TellEverything().

4.8.2.23 real EnergyMed[MAX1D]

Definition at line 16 of file global.h.

Referenced by DampingBoundary(), FillEnergy(), InitGasDensityEnergy(), NonReflectingBoundary(), RefillEnergy(), and SubStep3().

4.8.2.24 boolean ExcludeHill

Definition at line 31 of file global.h.

Referenced by FillForcesArrays(), and ReadVariables().

4.8.2.25 boolean FakeSequential

Definition at line 29 of file global.h.

Referenced by GasMomentum(), GasTotalEnergy(), GasTotalMass(), and main().

4.8.2.26 MPI_Status fargostat

Definition at line 32 of file global.h.

Referenced by ChessBoardIndexing(), CommunicateBoundaries(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), MakeDir(), mpi_make1Dprofile(), ReadfromAsciiFile(), ReadfromFile(), and SynchronizeOverlapFields().

4.8.2.27 boolean FeelDisk

Definition at line 26 of file global.h.

Referenced by ReadVariables(), RestartReboundSimulation(), and SetupReboundSimulation().

4.8.2.28 PolarGrid* GasAccelrad

Definition at line 37 of file global.h.

Referenced by InitPebbleArrays(), SubStep1(), and SubStep1Pebbles().

4.8.2.29 PolarGrid * GasAcceltheta

Definition at line 37 of file global.h.

Referenced by InitPebbleArrays(), SubStep1(), and SubStep1Pebbles().

4.8.2.30 int GLOBALNRAD

Definition at line 10 of file global.h.

Referenced by CheckRebin(), CreateTorqueMapInfile(), FillPolar1DArrays(), GetGlobalFrac(), InitGasVelocity(), mpi_make1Dprofile(), SplitDomain(), and WriteDim().

4.8.2.31 real GlobalRmed[MAX1D]

Definition at line 13 of file global.h.

Referenced by CheckRebin(), CreateTorqueMapInfile(), FillPolar1DArrays(), FViscosity(), GetGlobalFrac(), and InitGasVelocity().

4.8.2.32 real globcsvec[MAX1D]

Definition at line 16 of file global.h.

Referenced by ComputeSoundSpeed(), and FViscosity().

4.8.2.33 real globpressvec[MAX1D]

Definition at line 16 of file global.h.

Referenced by InitGasVelocity().

4.8.2.34 boolean GotoNextOutput

Definition at line 30 of file global.h.

Referenced by main().

4.8.2.35 PolarGrid* GravAccelRad

Definition at line 39 of file global.h.

Referenced by FillForcesArrays(), InitEuler(), and SubStep1().

4.8.2.36 PolarGrid * GravAccelTheta

Definition at line 39 of file global.h.

Referenced by FillForcesArrays(), InitEuler(), and SubStep1().

4.8.2.37 real heatsrc[MAXPLANETS]

Definition at line 21 of file global.h.

Referenced by AccretePebblesOntoPlanets(), ImplicitRadiativeDiffusion(), and ParametricAccretion().

4.8.2.38 boolean heatsrc_index[MAXPLANETS]

Definition at line 28 of file global.h.

Referenced by AccretePebblesOntoPlanets(), ImplicitRadiativeDiffusion(), and ParametricAccretion().

4.8.2.39 int heatsrc_max

Definition at line 22 of file global.h.

Referenced by AccretePebblesOntoPlanets(), ImplicitRadiativeDiffusion(), and ParametricAccretion().

4.8.2.40 int IMAX

Definition at line 5 of file global.h.

Referenced by SplitDomain().

4.8.2.41 int IMIN

Definition at line 4 of file global.h.

Referenced by FillPolar1DArrays(), InitGasVelocity(), mpi_make1Dprofile(), ReadfromAsciiFile(), ReadfromFile(), and SplitDomain().

4.8.2.42 boolean InitFromFile

Definition at line 25 of file global.h.

Referenced by Initialization(), and ReadVariables().

4.8.2.43 real InvDiffRmed[MAX1D]

Definition at line 12 of file global.h.

Referenced by ComputeStarRad(), FillPolar1DArrays(), ImplicitRadiativeDiffusion(), ParticleDiffusion(), SubStep1(), SubStep2(), TemperatureGradient(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.8.2.44 real InvDiffRsup[MAX1D]

Definition at line 13 of file global.h.

Referenced by CalculateFlaring(), FillPolar1DArrays(), ImplicitRadiativeDiffusion(), SubStep2(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.8.2.45 real invdtpeb_sq

Definition at line 19 of file global.h.

Referenced by ConditionCFL(), and CriticalCharTime().

4.8.2.46 real invdtreb_sq

Definition at line 19 of file global.h.

Referenced by ConditionCFL(), and MinStepForRebound().

4.8.2.47 real InvRinf[MAX1D]

Definition at line 13 of file global.h.

Referenced by FillPolar1DArrays(), SourceTermsPebbles(), SubStep1(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.8.2.48 real InvRmed[MAX1D]

Definition at line 12 of file global.h.

Referenced by AccretePebblesOntoPlanets(), CalculateFlaring(), ComputeResiduals(), ConditionCFL(), FillForcesArrays(), FillPolar1DArrays(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.8.2.49 real InvSurf[MAX1D]

Definition at line 12 of file global.h.

Referenced by FillPolar1DArrays(), and VanLeerRadial().

4.8.2.50 boolean LogGrid

Definition at line 41 of file global.h.

Referenced by FillPolar1DArrays(), and ReadVariables().

4.8.2.51 real MassTaper

Definition at line 14 of file global.h.

Referenced by AlgoGas(), and FillForcesArrays().

4.8.2.52 int Max_or_active

Definition at line 7 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), ConditionCFL(), CriticalCharTime(), DampingBoundary(), DampingTW04(), DampPebbles(), FillForcesArrays(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), mpi_make1Dprofile(), SetWaveKillingZones(), SplitDomain(), and ThicknessSmoothing().

4.8.2.53 int MaxMO_or_active

Definition at line 9 of file global.h.

Referenced by ConditionCFL(), ImplicitRadiativeDiffusion(), SplitDomain(), and SuccessiveOverrelaxation().

4.8.2.54 boolean Merge

Definition at line 29 of file global.h.

Referenced by AdvanceSystemFromDisk(), InitPebblesViaFlux(), main(), and SendOutput().

4.8.2.55 FILE * mergers

Definition at line 44 of file global.h.

Referenced by main(), ResolveCollisions(), RestartReboundSimulation(), and SetupReboundSimulation().

4.8.2.56 boolean MonitorIntegral

Definition at line 29 of file global.h.

Referenced by main().

4.8.2.57 boolean MonitorNPL

Definition at line 26 of file global.h.

Referenced by main(), and ReadVariables().

4.8.2.58 char NewOutputdir[1024]

Definition at line 43 of file global.h.

Referenced by main(), and ReadVariables().

4.8.2.59 real OmegaFrame

Definition at line 20 of file global.h.

4.8.2.60 real OmegaInv[MAX1D]

Definition at line 15 of file global.h.

Referenced by AccretePebblesOntoPlanets(), CalculateFlaring(), FillForcesArrays(), FillPolar1DArrays(), ImplicitRadiativeDiffusion(), MidplaneVolumeDensity(), SourceTermsPebbles(), SubStep1Pebbles(), and ThicknessSmoothing().

4.8.2.61 int One_or_active

Definition at line 8 of file global.h.

Referenced by ConditionCFL(), CriticalCharTime(), ImplicitRadiativeDiffusion(), SplitDomain(), and Successive↵ Overrelaxation().

4.8.2.62 boolean OnlyInit

Definition at line 29 of file global.h.

Referenced by main().

4.8.2.63 boolean OverridesOutputdir

Definition at line 42 of file global.h.

Referenced by main(), and ReadVariables().

4.8.2.64 boolean ParametricCooling

Definition at line 24 of file global.h.

Referenced by InitEuler(), InitGasVelocity(), ReadVariables(), SubStep3(), and TellEverything().

4.8.2.65 PolarGrid* PebbleDens

Definition at line 38 of file global.h.

Referenced by AccretePebblesOntoPlanets(), BckpFieldsForBC(), DetectCrashPebbles(), EvolvePebble↵ Disk(), InitPebbleArrays(), InitPebblesViaFlux(), ParticleDiffusion(), RestartPebbleDisk(), SourceTermsPebbles(), SynchronizePebbleDisc(), and WritePebbles().

4.8.2.66 PolarGrid * PebbleGravAccelRad

Definition at line 39 of file global.h.

Referenced by FillForcesArrays(), InitEuler(), and SourceTermsPebbles().

4.8.2.67 PolarGrid * PebbleGravAccelTheta

Definition at line 39 of file global.h.

Referenced by FillForcesArrays(), InitEuler(), and SourceTermsPebbles().

4.8.2.68 boolean Pebbles

Definition at line 27 of file global.h.

Referenced by AlgoGas(), ConditionCFL(), FillForcesArrays(), InitEuler(), Initialization(), merge(), ReadVariables(), SendOutput(), and SubStep1().

4.8.2.69 PolarGrid * PebbleVrad

Definition at line 38 of file global.h.

Referenced by `AccretePebblesOntoPlanets()`, `BckpFieldsForBC()`, `CriticalCharTime()`, `EvolvePebbleDisk()`, `InitPebbleArrays()`, `InitPebblesViaFlux()`, `ParticleDiffusion()`, `RestartPebbleDisk()`, `SourceTermsPebbles()`, `SubStep1Pebbles()`, `SynchronizePebbleDisc()`, and `WritePebbles()`.

4.8.2.70 `PolarGrid * PebbleVtheta`

Definition at line 38 of file `global.h`.

Referenced by `AccretePebblesOntoPlanets()`, `BckpFieldsForBC()`, `CorrectPebblesVtheta()`, `CriticalCharTime()`, `EvolvePebbleDisk()`, `InitPebbleArrays()`, `InitPebblesViaFlux()`, `ParticleDiffusion()`, `RestartPebbleDisk()`, `SourceTermsPebbles()`, `SubStep1Pebbles()`, `SynchronizePebbleDisc()`, and `WritePebbles()`.

4.8.2.71 `real PebDensInit[MAX1D]`

Definition at line 18 of file `global.h`.

Referenced by `BckpFieldsForBC()`, `DampPebbles()`, and `InitPebblesViaFlux()`.

4.8.2.72 `real PebVradInit[MAX1D]`

Definition at line 18 of file `global.h`.

Referenced by `BckpFieldsForBC()`, `DampPebbles()`, and `InitPebblesViaFlux()`.

4.8.2.73 `real PebVthetaInit[MAX1D]`

Definition at line 18 of file `global.h`.

Referenced by `BckpFieldsForBC()`, `DampPebbles()`, and `InitPebblesViaFlux()`.

4.8.2.74 `real PhysicalTime`

Definition at line 20 of file `global.h`.

Referenced by `AccretePebblesOntoPlanets()`, `AdvanceSystemRebound()`, `AlgoGas()`, `AspectRatio()`, `DiscardParticlesDist()`, `FindOrbitalElements()`, `FViscosity()`, `main()`, `OutputElements()`, `ParametricAccretion()`, `RestartReboundSimulation()`, `UpdateLog()`, `WriteBigPlanetFile()`, and `WritePlanetFile()`.

4.8.2.75 `real PhysicalTimeInitial`

Definition at line 20 of file `global.h`.

Referenced by `AspectRatio()`, `FViscosity()`, and `main()`.

4.8.2.76 `FILE * plout`

Definition at line 44 of file `global.h`.

Referenced by `main()`, `OutputElements()`, `RestartReboundSimulation()`, and `SetupReboundSimulation()`.

4.8.2.77 `boolean PrescribedAccretion`

Definition at line 27 of file `global.h`.

Referenced by `AlgoGas()`, and `ReadVariables()`.

4.8.2.78 PolarGrid * Pressure

Definition at line 35 of file global.h.

Referenced by ComputePressureField(), ComputeTemperatureField(), CreateTorqueMapInfile(), InitEuler(), InitGasVelocity(), and SubStep1().

4.8.2.79 PolarGrid * Qbalance

Definition at line 35 of file global.h.

Referenced by ActualizeQbalance(), InitRadiatDiffusionFields(), and SendOutput().

4.8.2.80 PolarGrid * Qminus

Definition at line 35 of file global.h.

Referenced by ActualizeQbalance(), CalculateQminus(), ImplicitRadiativeDiffusion(), and InitRadiatDiffusionFields().

4.8.2.81 PolarGrid * Qplus

Definition at line 35 of file global.h.

Referenced by ActualizeQbalance(), ImplicitRadiativeDiffusion(), InitEuler(), InitQplus(), SendOutput(), and SubStep3().

4.8.2.82 real QplusMed[MAX1D]

Definition at line 17 of file global.h.

Referenced by FillQplus(), and SubStep3().

4.8.2.83 real Radii[MAX1D]

Definition at line 13 of file global.h.

Referenced by CheckRebin(), FillPolar1DArrays(), and InitGasVelocity().

4.8.2.84 PolarGrid * RhoInt

Definition at line 34 of file global.h.

Referenced by InitEuler(), OneWindRad(), OneWindRadPebbles(), QuantitiesAdvection(), QuantitiesAdvectionPebbles(), SubStep2(), VanLeerRadial(), and VanLeerTheta().

4.8.2.85 PolarGrid* RhoStar

Definition at line 34 of file global.h.

Referenced by InitEuler(), OneWindRad(), OneWindRadPebbles(), QuantitiesAdvection(), QuantitiesAdvectionPebbles(), VanLeerRadial(), and VanLeerTheta().

4.8.2.86 real Rinf[MAX1D]

Definition at line 11 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), ApplyOuterSourceMass(), CalculateFlaring(), ConditionCFL(), CriticalCharTime(), DampingTW04(), FillPolar1DArrays(), FillSigma(), GasTotalEnergy(), ImplicitRadiativeDiffusion(), InitGasVelocity(), NonReflectingBoundary(), ParametricAccretion(), ParticleDiffusion(), RefillSigma(), SourceTermsPebbles(), SubStep1(), ThicknessSmoothing(), UpdateDivVelocAndStressTensor(), UpdateVelocityWithViscousTerms(), VanLeerRadial(), and VanLeerTheta().

4.8.2.87 real Rmed[**MAX1D**]

Definition at line 11 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), BckpFieldsForBC(), CalculateFlaring(), ComputeLRMomenta(), ComputeResiduals(), ComputeSoundSpeed(), ComputeStarRad(), ComputeStarTheta(), ComputeVelocities(), ConditionCFL(), CorrectVtheta(), CriticalCharTime(), DampingBoundary(), DampPebbles(), EtaPressureSupport(), FillCoolingTime(), FillEnergy(), FillForcesArrays(), FillPolar1DArrays(), FillQplus(), FillSigma(), FillVtheta(), GasMomentum(), GasTotalEnergy(), ImplicitRadiativeDiffusion(), ImposeKeplerianEdges(), InitComputeAccel(), InitGasVelocity(), InitLabel(), InitPebblesViaFlux(), NonReflectingBoundary(), ParticleDiffusion(), RefillSigma(), SetWaveKillingZones(), SubStep1(), SubStep2(), SubStep3(), TemperatureGradient(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.8.2.88 real Rmed2[**MAX1D**]

Definition at line 15 of file global.h.

Referenced by CalculateQirr(), FillForcesArrays(), and FillPolar1DArrays().

4.8.2.89 boolean RocheSmoothing

Definition at line 30 of file global.h.

Referenced by ReadVariables().

4.8.2.90 real Rsup[**MAX1D**]

Definition at line 11 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), CalculateFlaring(), ConditionCFL(), CriticalCharTime(), DampingTW04(), FillPolar1DArrays(), GasTotalEnergy(), ParametricAccretion(), ThicknessSmoothing(), UpdateDivVelocAndStressTensor(), UpdateVelocityWithViscousTerms(), VanLeerRadial(), and VanLeerTheta().

4.8.2.91 real SigmaInf[**MAX1D**]

Definition at line 14 of file global.h.

Referenced by FillSigma(), InitGasVelocity(), and RefillSigma().

4.8.2.92 real SigmaMed[**MAX1D**]

Definition at line 14 of file global.h.

Referenced by ApplyOuterSourceMass(), DampingBoundary(), FillSigma(), InitGasDensityEnergy(), NonReflectingBoundary(), OpenBoundary(), RefillSigma(), and SubStep3().

4.8.2.93 boolean SloppyCFL

Definition at line 31 of file global.h.

Referenced by AlgoGas(), and main().

4.8.2.94 **PolarGrid * SoundSpeed**

Definition at line 35 of file global.h.

Referenced by AccretePebblesOntoPlanets(), CalculateFlaring(), ComputePressureField(), ComputeSoundSpeed(), ConditionCFL(), CreateTorqueMapInfile(), DampingTW04(), FillForcesArrays(), ImplicitRadiativeDiffusion(), InitEuler(), InitGasVelocity(), MidplaneVolumeDensity(), NonReflectingBoundary(), ParametricAccretion(), ThicknessSmoothing(), and UpdateLog().

4.8.2.95 **real SQRT_ADIABIND_INV**

Definition at line 19 of file global.h.

Referenced by AccretePebblesOntoPlanets(), CalculateFlaring(), FillForcesArrays(), ImplicitRadiativeDiffusion(), InitEuler(), MidplaneVolumeDensity(), PebbleStokesNumbers(), ThicknessSmoothing(), and UpdateLog().

4.8.2.96 **boolean StellarIrradiation**

Definition at line 24 of file global.h.

Referenced by EffectiveOpticalDepth(), ImplicitRadiativeDiffusion(), and ReadVariables().

4.8.2.97 **PolarGrid * StokesNumber**

Definition at line 38 of file global.h.

Referenced by AccretePebblesOntoPlanets(), FillForcesArrays(), InitPebbleArrays(), InitPebblesViaFlux(), PebbleStokesNumbers(), SourceTermsPebbles(), and SubStep1Pebbles().

4.8.2.98 **boolean StoreEnergy**

Definition at line 24 of file global.h.

Referenced by main().

4.8.2.99 **boolean StoreSigma**

Definition at line 30 of file global.h.

Referenced by main().

4.8.2.100 **real Surf[MAX1D]**

Definition at line 11 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), FillForcesArrays(), FillPolar1DArrays(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), ParametricAccretion(), and VanLeerTheta().

4.8.2.101 **PolarGrid * TAUPP**

Definition at line 36 of file global.h.

Referenced by InitViscosity(), SubStep3(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.8.2.102 PolarGrid * TAURP

Definition at line 36 of file global.h.

Referenced by InitViscosity(), SubStep3(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.8.2.103 PolarGrid * TAURR

Definition at line 36 of file global.h.

Referenced by InitViscosity(), SubStep3(), UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

4.8.2.104 PolarGrid* Temperature

Definition at line 35 of file global.h.

Referenced by AlgoGas(), CalculateQminus(), ComputeTemperatureField(), CreateTorqueMapInfile(), DiffusionCoefs(), ImplicitRadiativeDiffusion(), InitEuler(), IterateRelaxationParameter(), OpacityProfile(), SendOutput(), SuccessiveOverrelaxation(), and TemperatureGradient().

4.8.2.105 int TimeStep

Definition at line 23 of file global.h.

4.8.2.106 PolarGrid* Torque

Definition at line 40 of file global.h.

Referenced by AdvanceSystemFromDisk(), FillForcesArrays(), and InitEuler().

4.8.2.107 boolean TorqueDensity

Definition at line 28 of file global.h.

Referenced by FillForcesArrays(), InitEuler(), main(), and ReadVariables().

4.8.2.108 boolean ViscosityAlpha

Definition at line 30 of file global.h.

Referenced by CreateTorqueMapInfile(), FViscosity(), InitGasVelocity(), and ReadVariables().

4.8.2.109 real vt1D[MAX1D]

Definition at line 19 of file global.h.

Referenced by AccretePebblesOntoPlanets().

4.8.2.110 real VthetaMed[MAX1D]

Definition at line 16 of file global.h.

Referenced by DampingBoundary(), and FillVtheta().

4.8.2.111 real WaveKiller[MAX1D]

Definition at line 16 of file global.h.

Referenced by DampingBoundary(), DampPebbles(), and SetWaveKillingZones().

4.8.2.112 boolean Write_Divergence

Definition at line 25 of file global.h.

Referenced by merge(), ReadVariables(), and SendOutput().

4.8.2.113 boolean Write_Energy

Definition at line 25 of file global.h.

Referenced by merge(), ReadVariables(), and SendOutput().

4.8.2.114 boolean Write_Eta

Definition at line 27 of file global.h.

Referenced by merge(), ReadVariables(), and WritePebbles().

4.8.2.115 boolean Write_Qbalance

Definition at line 25 of file global.h.

Referenced by InitRadiatDiffusionFields(), merge(), ReadVariables(), and SendOutput().

4.8.2.116 boolean Write_Qplus

Definition at line 25 of file global.h.

Referenced by merge(), ReadVariables(), and SendOutput().

4.8.2.117 boolean Write_Temperature

Definition at line 25 of file global.h.

Referenced by merge(), ReadVariables(), and SendOutput().

4.8.2.118 boolean WriteTorque

Definition at line 26 of file global.h.

Referenced by main(), and ReadVariables().

4.8.2.119 boolean WriteTorqueMapFile

Definition at line 26 of file global.h.

Referenced by main(), and ReadVariables().

4.8.2.120 int Zero_or_active

Definition at line 6 of file global.h.

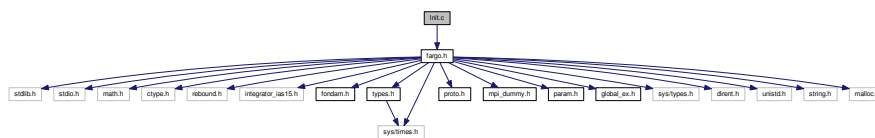
Referenced by `AccreteOntoPlanets()`, `AccretePebblesOntoPlanets()`, `ConditionCFL()`, `DampingBoundary()`, `DampingTW04()`, `DampPebbles()`, `FillForcesArrays()`, `GasMomentum()`, `GasTotalEnergy()`, `GasTotalMass()`, `mpi_↔_make1Dprofile()`, `SetWaveKillingZones()`, `SplitDomain()`, and `ThicknessSmoothing()`.

4.9 Init.c File Reference

Contains the functions needed to initialize the hydrodynamics arrays.

```
#include "fargo.h"
```

Include dependency graph for Init.c:



Functions

- void `ReadFromFile` (`PolarGrid` *array, char *fileprefix, int filenumber)
- void `InitLabel` (`PolarGrid` *array)
- void `Initialization` (`PolarGrid` *gas_density, `PolarGrid` *gas_v_rad, `PolarGrid` *gas_v_theta, `PolarGrid` *gas_↔_energy, `PolarGrid` *gas_label)
- void `ReadfromAsciiFile` (`PolarGrid` *array, char *path)

Enables to read a polar grid array from an ascii file.

Variables

- boolean `Restart`
- int `NbRestart`

4.9.1 Detailed Description

Contains the functions needed to initialize the hydrodynamics arrays.

These can be initialized by reading a given output (in the case of a restart) or by calling a function, `InitEuler()`, which contains analytic prescription for the different hydrodynamics fields. Note that this function `InitEuler()` is located in `SourceEuler.c`, which itself calls `InitGas()`, in the file `Pframeforce.c`. Also, note that the present file contains `InitLabel()`, which sets the initial value of a passive scalar.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file `Init.c`.

4.9.2 Function Documentation

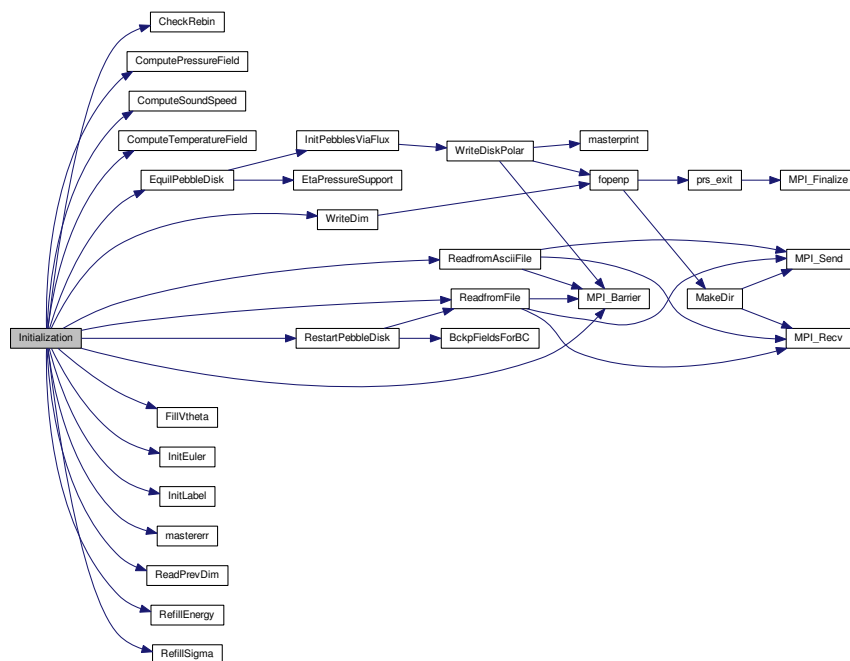
4.9.2.1 void Initialization (PolarGrid * gas_density, PolarGrid * gas_v_rad, PolarGrid * gas_v_theta, PolarGrid * gas_energy, PolarGrid * gas_label)

Definition at line 72 of file Init.c.

References ADIABIND, CheckRebin(), ComputePressureField(), ComputeSoundSpeed(), ComputeTemperatureField(), DampInit, DENSINFILE, EnergyEq, EquilPebbleDisk(), FillVtheta(), InitEuler(), InitFromFile, InitLabel(), mastererr(), MPI_Barrier(), MPI_COMM_WORLD, NbRestart, Pebbles, ReadfromAsciiFile(), ReadfromFile(), ReadPrevDim(), RefillEnergy(), RefillSigma(), Restart, RestartPebbleDisk(), TEMPERINFILE, VRADINFILE, VTHETAINFILE, and WriteDim().

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



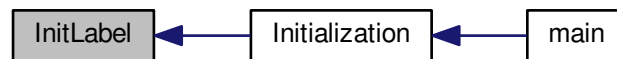
4.9.2.2 void InitLabel (PolarGrid * array)

Definition at line 56 of file Init.c.

References `polargrid::Field`, `RMAX`, `Rmed`, and `RMIN`.

Referenced by `Initialization()`.

Here is the caller graph for this function:



4.9.2.3 void ReadfromAsciiFile (PolarGrid * array, char * path)

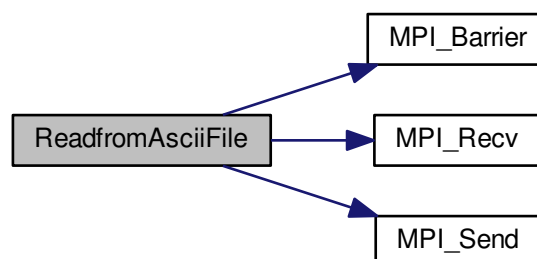
Enables to read a polar grid array from an ascii file.

Definition at line 148 of file `Init.c`.

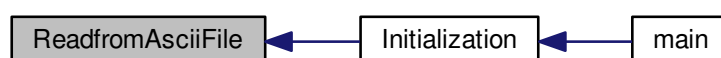
References `CPU_Number`, `CPU_Rank`, `fargostat`, `IMIN`, `MPI_Barrier()`, `MPI_COMM_WORLD`, `MPI_INT`, `MPI_Recv()`, and `MPI_Send()`.

Referenced by `Initialization()`.

Here is the call graph for this function:



Here is the caller graph for this function:



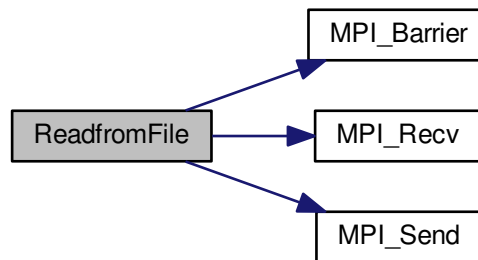
4.9.2.4 void ReadfromFile (PolarGrid * array, char * fileprefix, int filenumber)

Definition at line 23 of file Init.c.

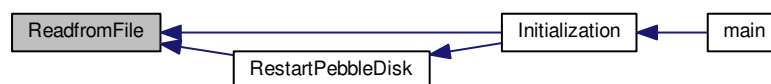
References CPU_Number, CPU_Rank, fargostat, IMIN, MPI_Barrier(), MPI_COMM_WORLD, MPI_INT, MPI_Recv(), MPI_Send(), and OUTPUTDIR.

Referenced by Initialization(), and RestartPebbleDisk().

Here is the call graph for this function:



Here is the caller graph for this function:



4.9.3 Variable Documentation

4.9.3.1 int NbRestart

Definition at line 15 of file main.c.

Referenced by Initialization(), and main().

4.9.3.2 boolean Restart

Definition at line 14 of file main.c.

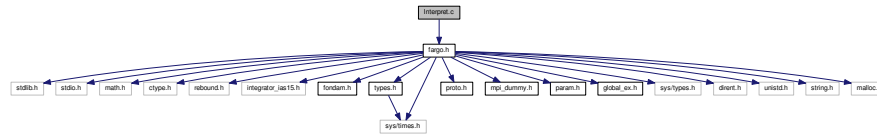
Referenced by Initialization().

4.10 Interpret.c File Reference

Contains the functions required to read the parameter file, and functions that provide runtime information.

```
#include "fargo.h"
```

Include dependency graph for Interpret.c:



Macros

- `#define MAXVARIABLES 500`

Functions

- void `var` (char *name, char *ptr, int type, int necessary, char *deflt)
- void `ReadVariables` (char *filename)
- void `PrintUsage` (char *execname)
- `real` `TellNbOrbits` (`real` time)
- `real` `TellNbOutputs` (`real` time)
- void `TellEverything` ()
- void `GiveTimeInfo` (int number)
- void `InitSpecificTime` (boolean profiling, `TimeProcess` *process_name, char *title)
- void `GiveSpecificTime` (boolean profiling, `TimeProcess` process_name)

Variables

- int `begin_i`
- boolean `OpenInner`
- boolean `Restart`
- static `Param` `VariableSet` [MAXVARIABLES]
- static int `VariableIndex` = 0
- static int `FirstStep` = YES
- static clock_t `First`
- static clock_t `Preceeding`
- static clock_t `Current`
- static clock_t `FirstUser`
- static clock_t `CurrentUser`
- static clock_t `PreceedingUser`
- static long `Ticks`
- boolean `FastTransport` = YES
- boolean `GuidingCenter` = NO
- boolean `IsDisk` = YES
- boolean `NonReflecting` = NO
- boolean `Corotating` = NO
- boolean `OuterSourceMass` = NO
- boolean `Write_Density` = YES
- boolean `Write_Velocity` = YES
- boolean `Indirect_Term` = YES

4.10.1 Detailed Description

Contains the functions required to read the parameter file, and functions that provide runtime information.

The function `var()` associates a string to a global variable. The function `ReadVariables()` reads the content of a parameter file. In addition, this file contains a function that prints the command line usage to the standard output, a function that provides verbose information about the setup (if the `-v` switch is set on the command line), and functions that act as a chronometer (if the `-t` switch is set on the command line).

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [Interpret.c](#).

4.10.2 Macro Definition Documentation

4.10.2.1 `#define MAXVARIABLES 500`

Definition at line 20 of file `Interpret.c`.

4.10.3 Function Documentation

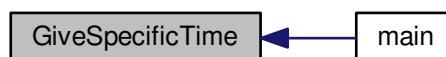
4.10.3.1 `void GiveSpecificTime (boolean profiling, TimeProcess process_name)`

Definition at line 402 of file `Interpret.c`.

References `timeprocess::clicks`, `timeprocess::name`, `NO`, and `Ticks`.

Referenced by `main()`.

Here is the caller graph for this function:



4.10.3.2 `void GiveTimeInfo (int number)`

Definition at line 358 of file `Interpret.c`.

References `begin_i`, `Current`, `CurrentUser`, `First`, `FirstStep`, `FirstUser`, `NINTERM`, `NO`, `Preceeding`, `PreceedingUser`, `Ticks`, and `YES`.

Referenced by `main()`.

Here is the caller graph for this function:



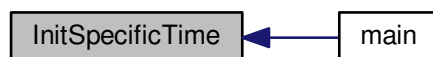
4.10.3.3 void InitSpecificTime (boolean *profiling*, TimeProcess * *process_name*, char * *title*)

Definition at line 389 of file Interpret.c.

References `timeprocess::clicks`, `timeprocess::name`, `NO`, and `Ticks`.

Referenced by `main()`.

Here is the caller graph for this function:



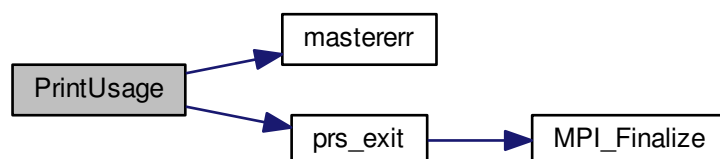
4.10.3.4 void PrintUsage (char * *execname*)

Definition at line 238 of file Interpret.c.

References `mastererr()`, and `prs_exit()`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



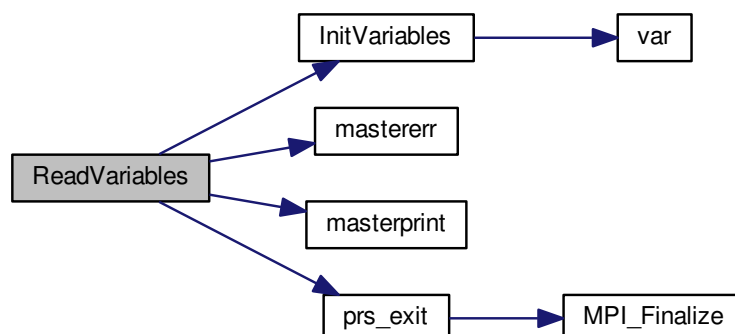
4.10.3.5 void ReadVariables (char * filename)

Definition at line 65 of file Interpret.c.

References AccretHeating, ACCRETIONALHEATING, AdvecteLabel, ADVLABEL, ALPHAVISCOSITY, Back←
 Reaction, BACKREACTION, Collisions, COOLINGTIME, Corotating, Damping, DampInit, DAMPTOWARDS,
 DampVrad, DiffusiveParticles, DISK, EnergyEq, ENERGYEQUATION, ExcludeHill, EXCLUDEHILL, FastTransport,
 FeelDisk, FRAME, GETTORQUEFORPLANET, GRIDSPACING, GuidingCenter, Indirect_Term, INDIRECTTERM,
 InitFromFile, INITIALIZEFROMFILE, InitVariables(), INT, IsDisk, LogGrid, mastererr(), masterprint(), MonitorNPL,
 NewOutputdir, NO, NonReflecting, OpenInner, OPENINNERBOUNDARY, OuterSourceMass, OUTERSOUR←
 CEMASS, OUTPUTDIR, OverridesOutputdir, PARAMETRICACCRETION, ParametricCooling, PARTICLEDIF←
 FUSION, PEBBLEACCRETION, Pebbles, PLANETSFEELDISK, PrescribedAccretion, prs_exit(), param::read,
 REAL, RESOLVECOLLISIONS, ROCHESMOOTHING, RocheSmoothing, StellarIrradiation, STELLARIRRADIA←
 TION, STRING, TARGETNPL, THICKNESSSMOOTHING, TorqueDensity, TORQUEMAPINFILE, TRANSPORT,
 VariableIndex, VISCOSITY, ViscosityAlpha, Write_Density, Write_Divergence, Write_Energy, Write_Eta, Write←
 _Qbalance, Write_Qplus, Write_Temperature, Write_Velocity, WRITEDENSITY, WRITEDIVV, WRITEENERGY,
 WRITEETA, WRITEQBALANCE, WRITEQPLUS, WRITETEMPERATURE, WriteTorque, WRITETORQUEFILES,
 WriteTorqueMapFile, WRITEVELOCITY, and YES.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



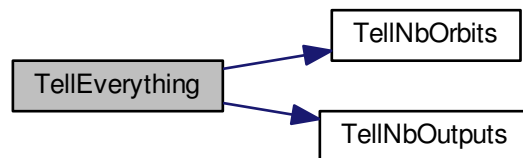
4.10.3.6 void TellEverything ()

Definition at line 281 of file Interpret.c.

References AdvecteLabel, ASPECTRATIO, CPU_Master, DT, EnergyEq, G, NINTERM, NRAD, NSEC, NTO↵T, ParametricCooling, PI, RMAX, RMIN, SIGMA0, TellNbOrbits(), TellNbOutputs(), and YES.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



4.10.3.7 real TellNbOrbits (real time)

Definition at line 269 of file Interpret.c.

References G, and PI.

Referenced by TellEverything().

Here is the caller graph for this function:



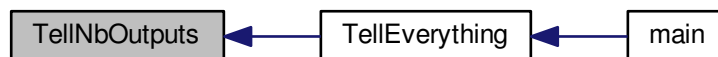
4.10.3.8 `real TellNbOutputs (real time)`

Definition at line 275 of file Interpret.c.

References DT, and NINTERM.

Referenced by TellEverything().

Here is the caller graph for this function:



4.10.3.9 `void var (char * name, char * ptr, int type, int necessary, char * deflt)`

Definition at line 34 of file Interpret.c.

References INT, param::necessary, NO, param::read, REAL, STRING, param::type, param::variable, and Variable↔Index.

Referenced by InitVariables().

Here is the caller graph for this function:



4.10.4 Variable Documentation

4.10.4.1 `int begin_i`

Definition at line 15 of file main.c.

Referenced by GiveTimeInfo(), and main().

4.10.4.2 **boolean** Corotating = NO

Definition at line 30 of file Interpret.c.

Referenced by AlgoGas(), main(), and ReadVariables().

4.10.4.3 **clock_t** Current [static]

Definition at line 27 of file Interpret.c.

Referenced by GiveTimeInfo().

4.10.4.4 **clock_t** CurrentUser [static]

Definition at line 27 of file Interpret.c.

Referenced by GiveTimeInfo().

4.10.4.5 **boolean** FastTransport = YES

Definition at line 29 of file Interpret.c.

Referenced by ComputeResiduals(), ConditionCFL(), CriticalCharTime(), OneWindTheta(), OneWindTheta↔Pebbles(), and ReadVariables().

4.10.4.6 **clock_t** First [static]

Definition at line 27 of file Interpret.c.

Referenced by GiveTimeInfo().

4.10.4.7 **int** FirstStep = YES [static]

Definition at line 26 of file Interpret.c.

Referenced by GiveTimeInfo().

4.10.4.8 **clock_t** FirstUser [static]

Definition at line 27 of file Interpret.c.

Referenced by GiveTimeInfo().

4.10.4.9 **boolean** GuidingCenter = NO

Definition at line 29 of file Interpret.c.

Referenced by GetPsysInfo(), GetPsysInfoFromRsim(), and ReadVariables().

4.10.4.10 **boolean** Indirect_Term = YES

Definition at line 31 of file Interpret.c.

Referenced by FillForcesArrays(), and ReadVariables().

4.10.4.11 **boolean** IsDisk = YES

Definition at line 30 of file Interpret.c.

Referenced by AlgoGas(), ReadVariables(), and SendOutput().

4.10.4.12 **boolean** NonReflecting = NO

Definition at line 30 of file Interpret.c.

Referenced by ApplyBoundaryCondition(), and ReadVariables().

4.10.4.13 **boolean** OpenInner

Definition at line 14 of file main.c.

Referenced by ReadVariables().

4.10.4.14 **boolean** OuterSourceMass = NO

Definition at line 30 of file Interpret.c.

Referenced by ApplyBoundaryCondition(), and ReadVariables().

4.10.4.15 **clock_t** Preceeding [static]

Definition at line 27 of file Interpret.c.

Referenced by GiveTimeInfo().

4.10.4.16 **clock_t** PreceedingUser [static]

Definition at line 27 of file Interpret.c.

Referenced by GiveTimeInfo().

4.10.4.17 **boolean** Restart

Definition at line 14 of file main.c.

4.10.4.18 **long** Ticks [static]

Definition at line 28 of file Interpret.c.

Referenced by GiveSpecificTime(), GiveTimeInfo(), and InitSpecificTime().

4.10.4.19 **int** VariableIndex = 0 [static]

Definition at line 25 of file Interpret.c.

Referenced by ReadVariables(), and var().

4.10.4.20 **Param** VariableSet[MAXVARIABLES] [static]

Definition at line 24 of file Interpret.c.

4.10.4.21 **boolean** Write_Density = YES

Definition at line 31 of file Interpret.c.

Referenced by ReadVariables(), and SendOutput().

4.10.4.22 **boolean** Write_Velocity = YES

Definition at line 31 of file Interpret.c.

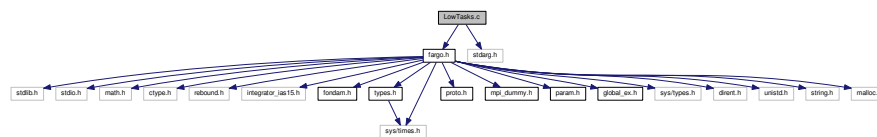
Referenced by ReadVariables(), and SendOutput().

4.11 LowTasks.c File Reference

Contains many low level short functions.

```
#include "fargo.h"
#include <stdarg.h>
```

Include dependency graph for LowTasks.c:



Functions

- [real](#) [GetGlobalFrac](#) ([real](#) r)
- void [prs_exit](#) (int numb)
- void [masterprint](#) (const char *template,...)
- void [mastererr](#) (const char *template,...)
- void [prs_error](#) (char *string)
- void [message](#) (char *msg)
- [PolarGrid](#) * [CreatePolarGrid](#) (int Nr, int Ns, char *name)
- void [MultiplyPolarGridbyConstant](#) ([PolarGrid](#) *arraysrc, [real](#) constant)
- void [DumpSources](#) (int argc, argv)
- void [MakeDir](#) (char *string)
- FILE * [fopenp](#) (char *string, char *mode)

4.11.1 Detailed Description

Contains many low level short functions.

The name of these functions should be self-explanatory in most cases. The prefix 'prs_' stands for 'personal'. The prefix 'master' means that only the process 0 executes the function [note that the architecture is not of the kind master/slaves, all processes perform similar tasks, but a minor number of tasks (like output of information on the standard output) do not need to be performed by several processes.] The function [fopenp\(\)](#) is an upper layer of [fopen\(\)](#), which should be used only in the case of writing or appending a file (and not reading a file). It tries to create the output directory if it does not exist, and it issues an error message if it fails, so that the calling function does not need to worry about these details.

Definition in file [LowTasks.c](#).

4.11.2 Function Documentation

4.11.2.1 `PolarGrid* CreatePolarGrid (int Nr, int Ns, char * name)`

Definition at line 73 of file LowTasks.c.

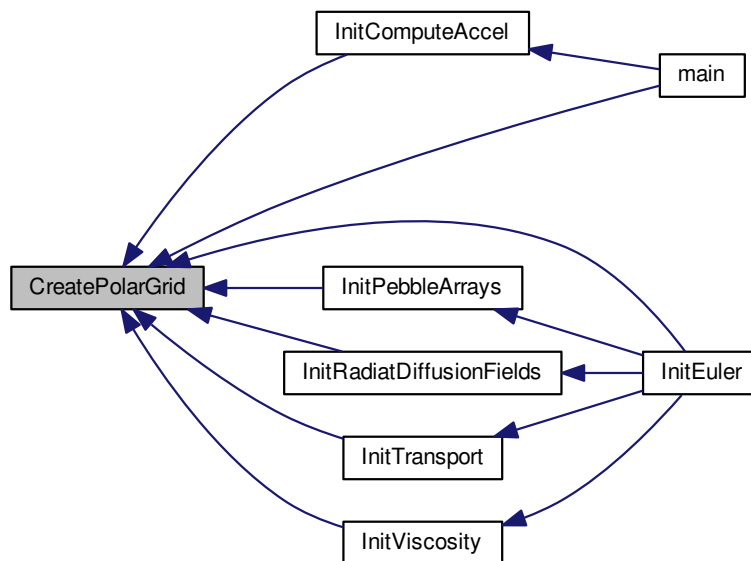
References `polargrid::Field`, `polargrid::Name`, `polargrid::Nrad`, `polargrid::Nsec`, and `prs_error()`.

Referenced by `InitComputeAccel()`, `InitEuler()`, `InitPebbleArrays()`, `InitRadiatDiffusionFields()`, `InitTransport()`, `InitViscosity()`, and `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



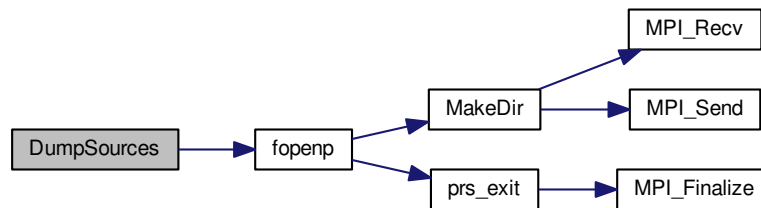
4.11.2.2 `void DumpSources (int argc, argv)`

Definition at line 123 of file LowTasks.c.

References `CPU_Master`, `fopenp()`, and `OUTPUTDIR`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



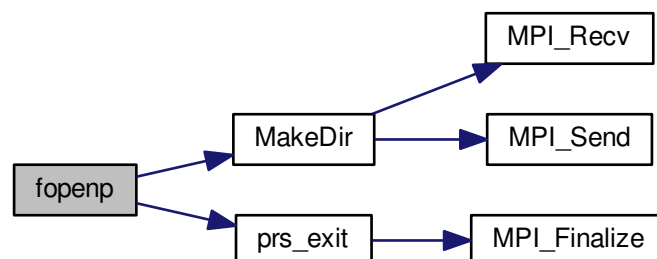
4.11.2.3 FILE* fopenp (char * *string*, char * *mode*)

Definition at line 163 of file LowTasks.c.

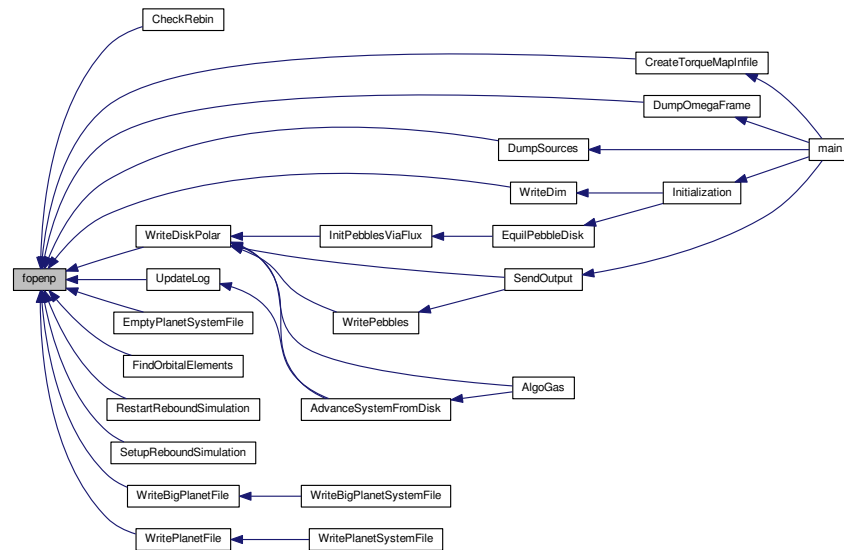
References CPU_Rank, MakeDir(), OUTPUTDIR, and prs_exit().

Referenced by CheckRebin(), CreateTorqueMapInfile(), DumpOmegaFrame(), DumpSources(), EmptyPlanet↔SystemFile(), FindOrbitalElements(), RestartReboundSimulation(), SetupReboundSimulation(), UpdateLog(), WriteBigPlanetFile(), WriteDim(), WriteDiskPolar(), and WritePlanetFile().

Here is the call graph for this function:



Here is the caller graph for this function:



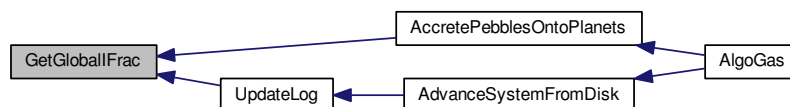
4.11.2.4 real GetGlobalFrac (real *r*)

Definition at line 21 of file LowTasks.c.

References GLOBALNRAD, and GlobalRmed.

Referenced by AccretePebblesOntoPlanets(), and UpdateLog().

Here is the caller graph for this function:



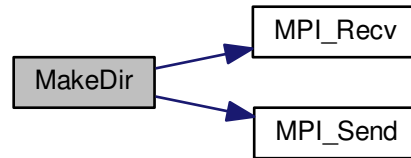
4.11.2.5 void MakeDir (char * *string*)

Definition at line 142 of file LowTasks.c.

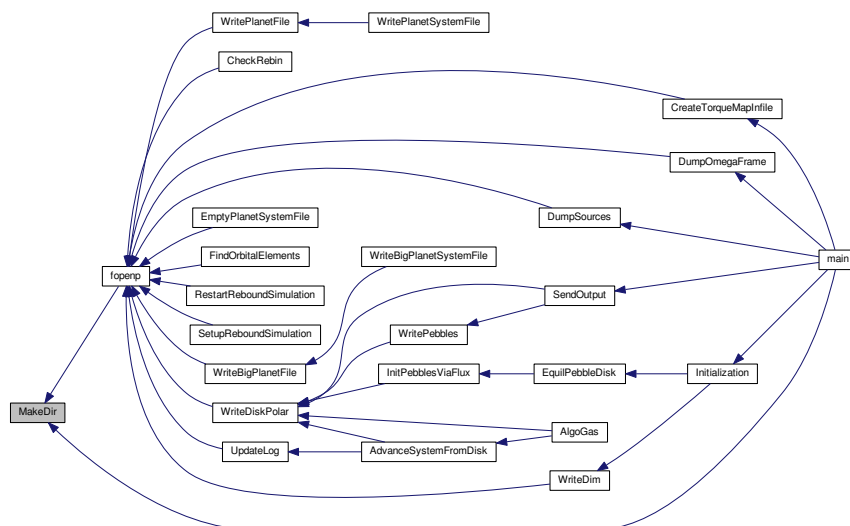
References CPU_Number, CPU_Rank, fargostat, MAX1D, MPI_COMM_WORLD, MPI_INT, MPI_Recv(), and MPI_Send().

Referenced by fopenp(), and main().

Here is the call graph for this function:



Here is the caller graph for this function:



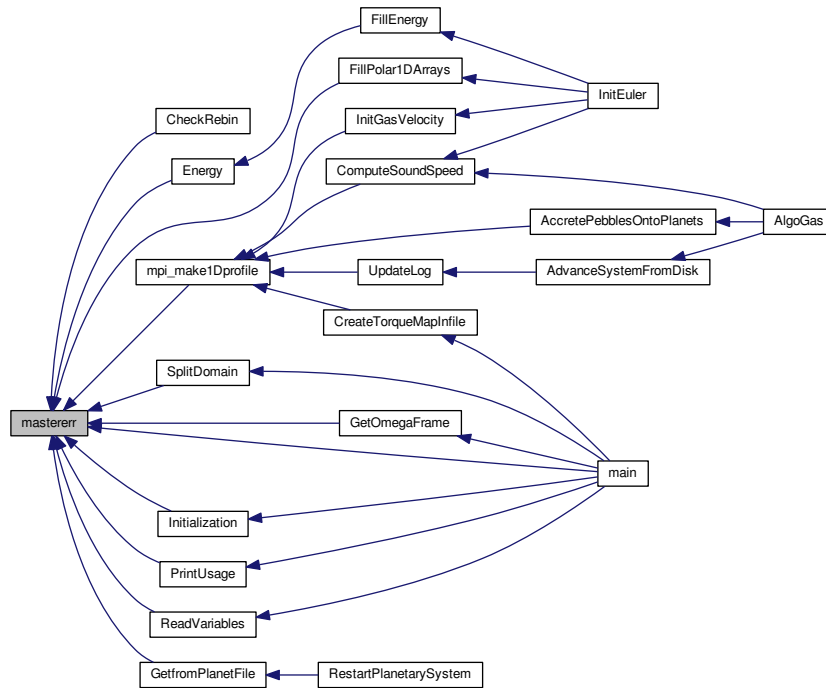
4.11.2.6 void mastererr (const char * *template*, ...)

Definition at line 49 of file `LowTasks.c`.

References `CPU_Master`.

Referenced by `CheckRebin()`, `Energy()`, `FillPolar1DArrays()`, `GetfromPlanetFile()`, `GetOmegaFrame()`, `Initialization()`, `main()`, `mpi_make1Dprofile()`, `PrintUsage()`, `ReadVariables()`, and `SplitDomain()`.

Here is the caller graph for this function:



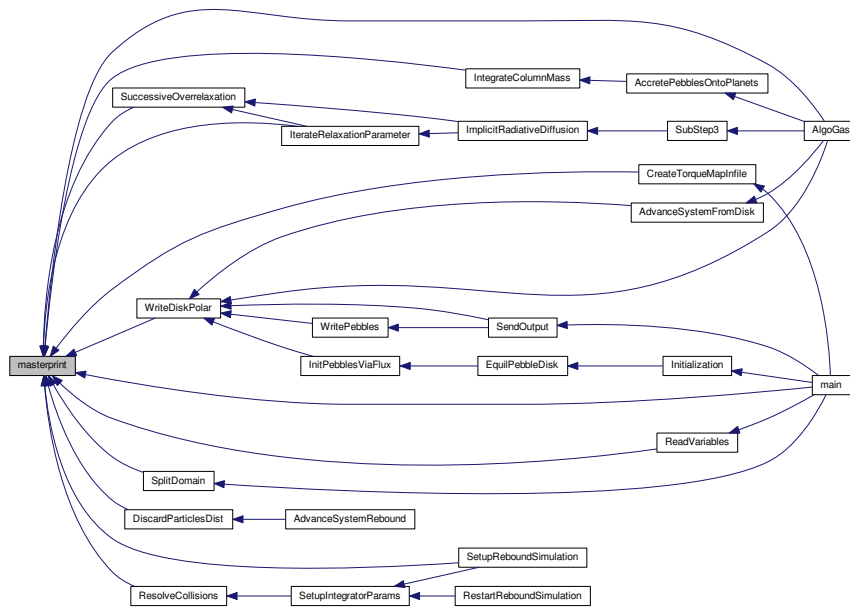
4.11.2.7 void masterprint (const char * *template*, ...)

Definition at line 40 of file `LowTasks.c`.

References `CPU_Master`.

Referenced by `AlgoGas()`, `CreateTorqueMapInfile()`, `DiscardParticlesDist()`, `IntegrateColumnMass()`, `IterateRelaxationParameter()`, `main()`, `ReadVariables()`, `ResolveCollisions()`, `SetupReboundSimulation()`, `SplitDomain()`, `SuccessiveOverrelaxation()`, and `WriteDiskPolar()`.

Here is the caller graph for this function:



4.11.2.8 void message (char * msg)

Definition at line 66 of file LowTasks.c.

Referenced by merge().

Here is the caller graph for this function:



4.11.2.9 void MultiplyPolarGridbyConstant (PolarGrid * arraysrc, real constant)

Definition at line 106 of file LowTasks.c.

References polargrid::Nrad.

Referenced by main().

Here is the caller graph for this function:



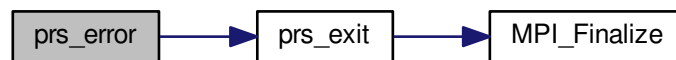
4.11.2.10 void prs_error (char * string)

Definition at line 59 of file LowTasks.c.

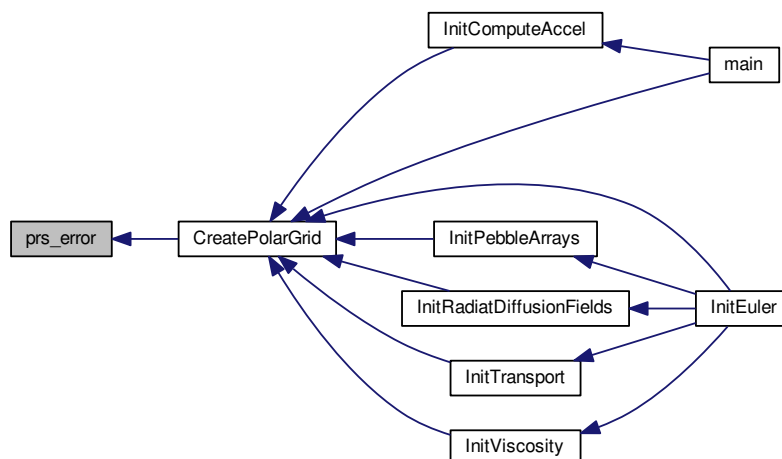
References `prs_exit()`.

Referenced by `CreatePolarGrid()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.11.2.11 void prs_exit (int numb)

Definition at line 33 of file LowTasks.c.

References `MPI_Finalize()`.

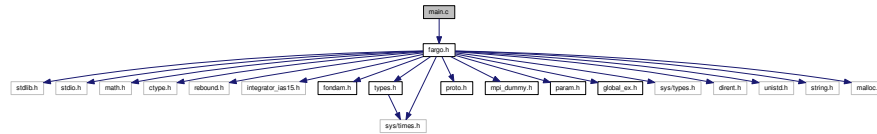
Referenced by `AllocateComm()`, `AllocPlanetSystem()`, `CheckRebin()`, `Energy()`, `FillPolar1DArrays()`, `FindNumber↵
OfPlanets()`, `fopenp()`, `GetfromPlanetFile()`, `GetOmegaFrame()`, `handfpe()`, `IntegrateColumnMass()`, `Iterate↵
RelaxationParameter()`, `main()`, `mpi_make1Dprofile()`, `PrintUsage()`, `prs_error()`, `ReadVariables()`, `SetupRebound↵
Simulation()`, `SplitDomain()`, `SuccessiveOverrelaxation()`, `SynchronizeFargoRebound()`, and `SynchronizeOverlap↵
Fields()`.

Here is the call graph for this function:




```
#include "fargo.h"
```

Include dependency graph for main.c:



Functions

- int **main** (int argc, argv)

Variables

- boolean Restart = NO
- boolean OpenInner = NO
- int begin_i = 0
- int NbRestart = 0
- static int InnerOutputCounter = 0
- static int StillWriteOneOutput
- real LostMass
- boolean Corotating
- real ScalingFactor = 1.0
- boolean DumpTorqueNow = NO
- boolean DumpTorqueDensNow = NO

4.12.1 Detailed Description

Main file of the distribution.

Manages the call to initialization functions, then the main loop.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [main.c](#).

4.12.2 Function Documentation

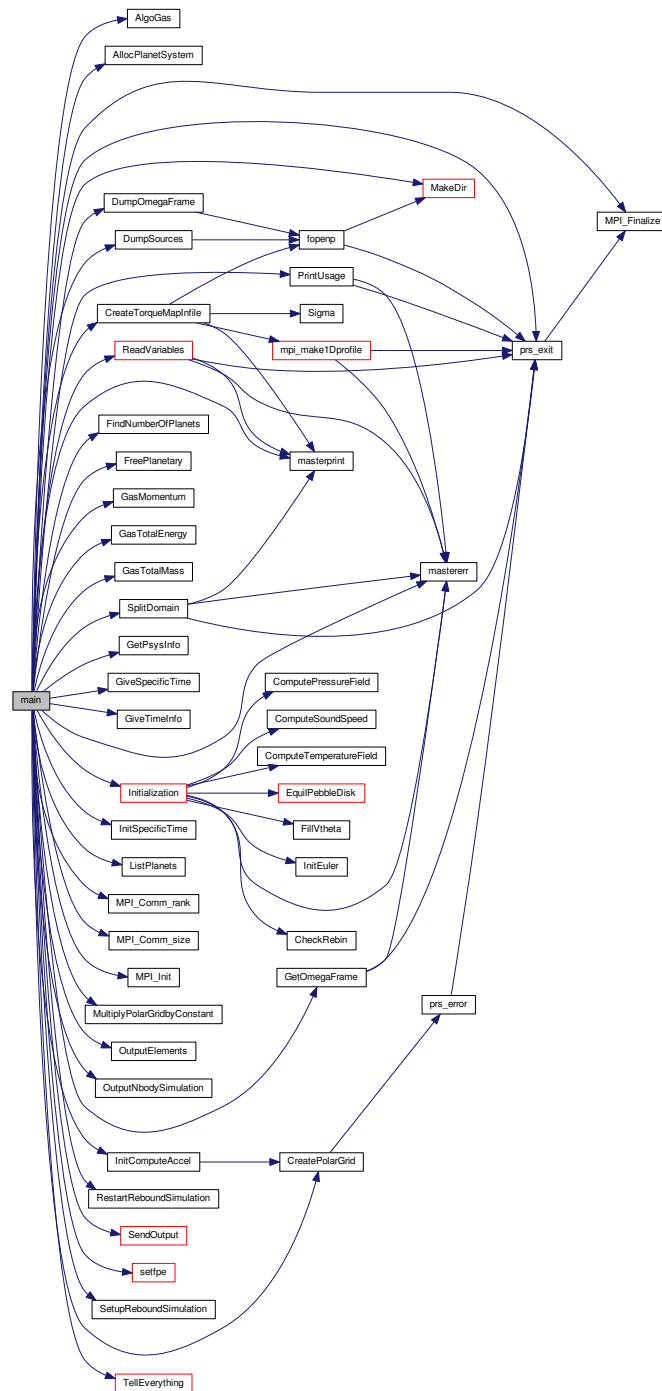
4.12.2.1 int main (int argc, argv)

Definition at line 24 of file main.c.

References AlgoGas(), AllocPlanetSystem(), begin_i, CentrifugalBalance, Collisions, Corotating, CPU_Master, CPU_Number, CPU_Rank, CreatePolarGrid(), CreateTorqueMapInfile(), debug, discard, DumpOmegaFrame(), DumpSources(), DumpTorqueDensNow, DumpTorqueNow, EnergyEq, FakeSequential, FindNumberOfPlanets(), FreePlanetary(), FREQUENCY, GasMomentum(), GasTotalEnergy(), GasTotalMass(), GetOmegaFrame(), Get↵ PsysInfo(), GiveSpecificTime(), GiveTimeInfo(), GotoNextOutput, InitComputeAccel(), Initialization(), InitSpecific↵ Time(), InnerOutputCounter, ListPlanets(), MakeDir(), mastererr(), masterprint(), Merge, mergers, MonitorIntegral,

MonitorNPL, MPI_Comm_rank(), MPI_Comm_size(), MPI_COMM_WORLD, MPI_Finalize(), MPI_Init(), MultiplyPolarGridbyConstant(), planetary_system::nb, NbRestart, NewOutputdir, NINTERM, NO, NOELEMENTS, N<RAD, NSEC, NTOT, OmegaFrame, OMEGAFRAME, OnlyInit, OUTPUTDIR, OutputElements(), OutputNbodySimulation(), OverridesOutputdir, PhysicalTime, PhysicalTimeInitial, PLANETCONFIG, plout, PrintUsage(), prs_exit(), ReadVariables(), Restart, RestartReboundSimulation(), ScalingFactor, SendOutput(), setpfe(), SetupReboundSimulation(), SloppyCFL, SplitDomain(), StillWriteOneOutput, StoreEnergy, StoreSigma, TARGETNPL, TellEverything(), TimeStep, TorqueDensity, WriteTorque, WriteTorqueMapFile, and YES.

Here is the call graph for this function:



4.12.3 Variable Documentation

4.12.3.1 `int begin_i = 0`

Definition at line 15 of file main.c.

Referenced by GiveTimeInfo(), and main().

4.12.3.2 `boolean Corotating`

Definition at line 30 of file Interpret.c.

Referenced by main().

4.12.3.3 `boolean DumpTorqueDensNow = NO`

Definition at line 21 of file main.c.

Referenced by AdvanceSystemFromDisk(), and main().

4.12.3.4 `boolean DumpTorqueNow = NO`

Definition at line 21 of file main.c.

Referenced by AdvanceSystemFromDisk(), and main().

4.12.3.5 `int InnerOutputCounter = 0` `[static]`

Definition at line 16 of file main.c.

Referenced by main().

4.12.3.6 `real LostMass`

Definition at line 32 of file TransportEuler.c.

4.12.3.7 `int NbRestart = 0`

Definition at line 15 of file main.c.

Referenced by Initialization(), and main().

4.12.3.8 `boolean OpenInner = NO`

Definition at line 14 of file main.c.

Referenced by ApplyBoundaryCondition(), ReadVariables(), and VanLeerRadial().

4.12.3.9 `boolean Restart = NO`

Definition at line 14 of file main.c.

Referenced by Initialization(), and main().

4.12.3.10 `real ScalingFactor = 1.0`

Definition at line 19 of file `main.c`.

Referenced by `main()`, and `Sigma()`.

4.12.3.11 `int StillWriteOneOutput` `[static]`

Definition at line 16 of file `main.c`.

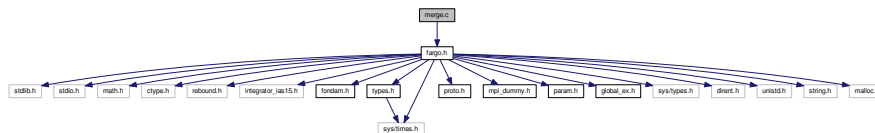
Referenced by `main()`.

4.13 `merge.c` File Reference

Contains the function that merges the output of different processors.

```
#include "fargo.h"
```

Include dependency graph for `merge.c`:



Functions

- void `merge` (int nb)

4.13.1 Detailed Description

Contains the function that merges the output of different processors.

The resulting merged file is undistinguishable from the file that would have been produced by a sequential run.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file `merge.c`.

4.13.2 Function Documentation

4.13.2.1 `void merge (int nb)`

Definition at line 15 of file `merge.c`.

References `AdvecteLabel`, `CPU_Master`, `CPU_Number`, `message()`, `NO`, `OUTPUTDIR`, `Pebbles`, `Write_Divergence`, `Write_Energy`, `Write_Eta`, `Write_Qbalance`, `Write_Qplus`, `Write_Temperature`, and `YES`.

Referenced by `SendOutput()`.

Here is the call graph for this function:



Here is the caller graph for this function:



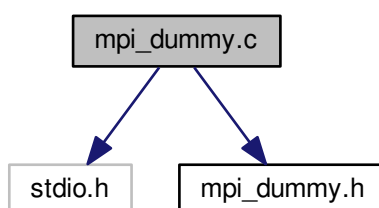
4.14 mpi_dummy.c File Reference

Fake MPI functions library for sequential built.

```
#include <stdio.h>
```

```
#include "mpi_dummy.h"
```

Include dependency graph for mpi_dummy.c:



Functions

- void [MPI_Comm_rank](#) (int a, int *b)
- void [MPI_Comm_size](#) (int a, int *b)
- void [MPI_Init](#) (int *argc, argv)
- void [MPI_Finalize](#) ()
- void [MPI_Bcast](#) ()
- void [MPI_Isend](#) ()

- void [MPI_Irecv](#) ()
- void [MPI_Send](#) ()
- void [MPI_Recv](#) ()
- void [MPI_Barrier](#) ()
- void [MPI_Wait](#) ()
- void [MPI_Allreduce](#) (void *ptr, void *ptr2, int count, int type, int foo3, int foo4)

4.14.1 Detailed Description

Fake MPI functions library for sequential built.

It is used instead of the true MPI library in the case of a sequential built (see makefile).

Definition in file [mpi_dummy.c](#).

4.14.2 Function Documentation

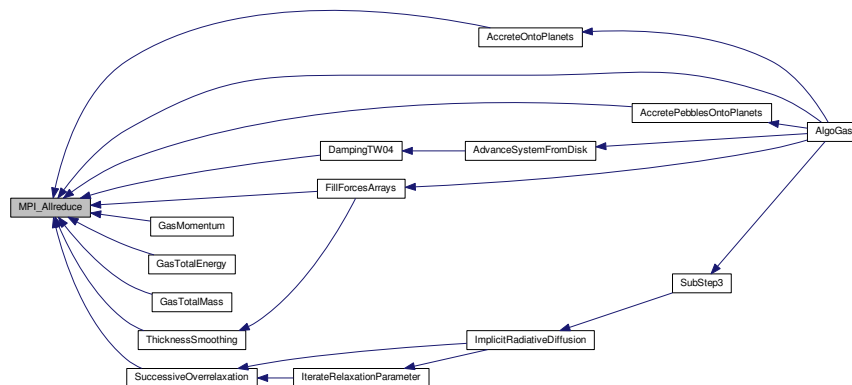
4.14.2.1 void [MPI_Allreduce](#) (void * *ptr*, void * *ptr2*, int *count*, int *type*, int *foo3*, int *foo4*)

Definition at line 72 of file [mpi_dummy.c](#).

References [MPI_DOUBLE](#), and [MPI_INT](#).

Referenced by [AccreteOntoPlanets\(\)](#), [AccretePebblesOntoPlanets\(\)](#), [AlgoGas\(\)](#), [DampingTW04\(\)](#), [FillForcesArrays\(\)](#), [GasMomentum\(\)](#), [GasTotalEnergy\(\)](#), [GasTotalMass\(\)](#), [SuccessiveOverrelaxation\(\)](#), and [ThicknessSmoothing\(\)](#).

Here is the caller graph for this function:

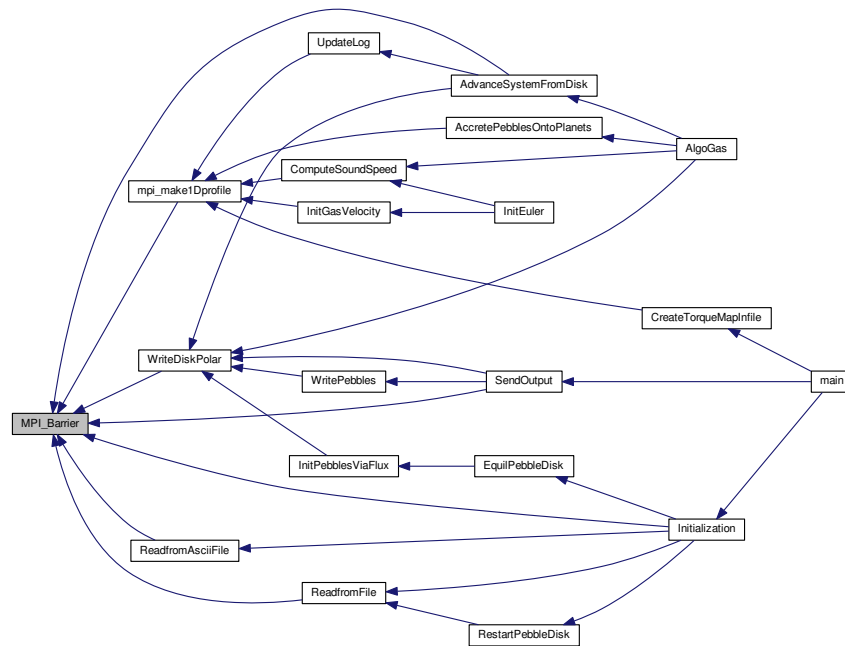


4.14.2.2 void [MPI_Barrier](#) ()

Definition at line 64 of file [mpi_dummy.c](#).

Referenced by [AdvanceSystemFromDisk\(\)](#), [Initialization\(\)](#), [mpi_make1Dprofile\(\)](#), [ReadfromAsciiFile\(\)](#), [ReadfromFile\(\)](#), [SendOutput\(\)](#), and [WriteDiskPolar\(\)](#).

Here is the caller graph for this function:

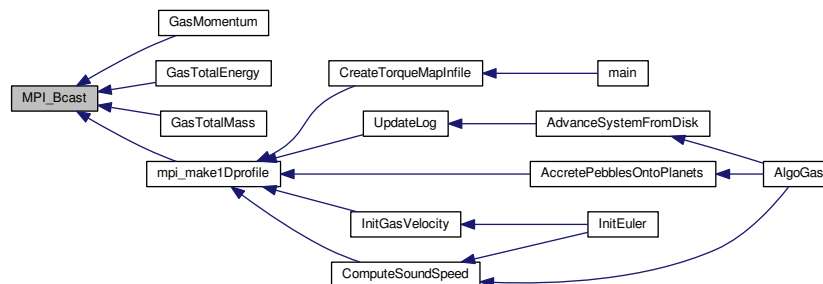


4.14.2.3 void MPI_Bcast ()

Definition at line 44 of file mpi_dummy.c.

Referenced by GasMomentum(), GasTotalEnergy(), GasTotalMass(), and mpi_make1Dprofile().

Here is the caller graph for this function:

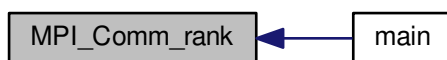


4.14.2.4 void MPI_Comm_rank (int a, int * b)

Definition at line 13 of file mpi_dummy.c.

Referenced by main().

Here is the caller graph for this function:

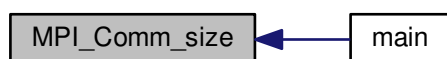


4.14.2.5 void MPI_Comm_size (int *a*, int * *b*)

Definition at line 20 of file `mpi_dummy.c`.

Referenced by `main()`.

Here is the caller graph for this function:

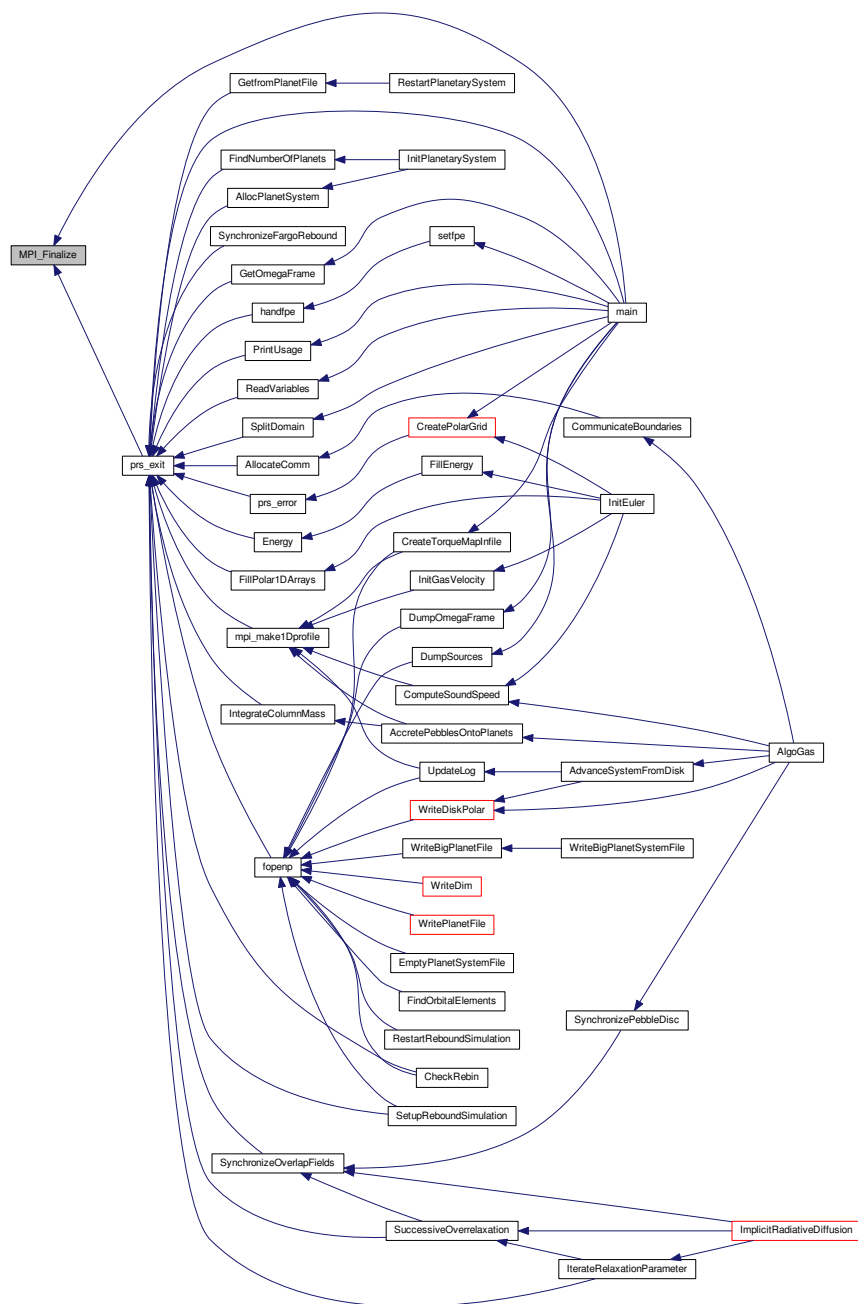


4.14.2.6 void MPI_Finalize ()

Definition at line 40 of file `mpi_dummy.c`.

Referenced by `main()`, and `prs_exit()`.

Here is the caller graph for this function:

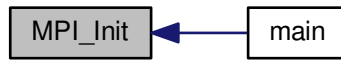


4.14.2.7 void MPI_Init (int * *argc*, argv)

Definition at line 27 of file mpi_dummy.c.

Referenced by main().

Here is the caller graph for this function:

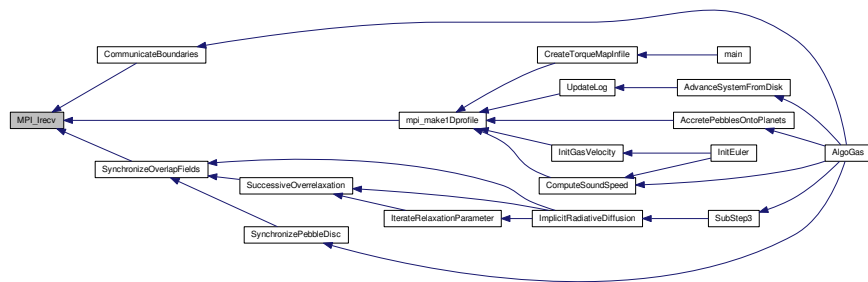


4.14.2.8 void MPI_Irecv ()

Definition at line 52 of file mpi_dummy.c.

Referenced by CommunicateBoundaries(), mpi_make1Dprofile(), and SynchronizeOverlapFields().

Here is the caller graph for this function:

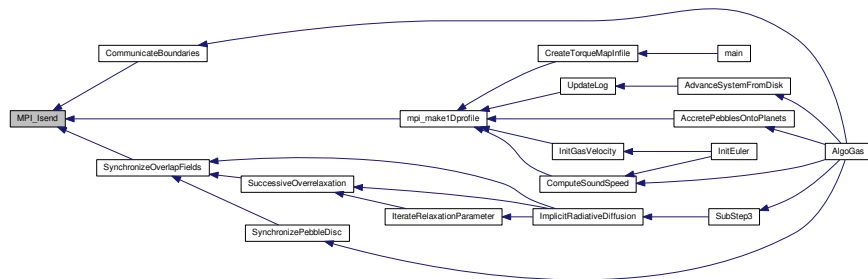


4.14.2.9 void MPI_Isend ()

Definition at line 48 of file mpi_dummy.c.

Referenced by CommunicateBoundaries(), mpi_make1Dprofile(), and SynchronizeOverlapFields().

Here is the caller graph for this function:

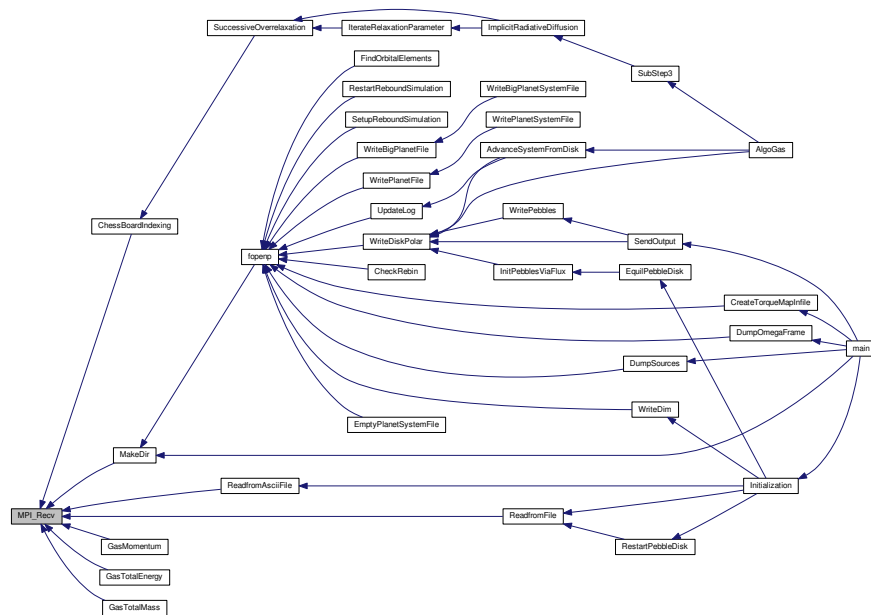


4.14.2.10 void MPI_Recv ()

Definition at line 60 of file mpi_dummy.c.

Referenced by ChessBoardIndexing(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), MakeDir(), ReadfromAsciiFile(), and ReadfromFile().

Here is the caller graph for this function:



4.14.2.11 void MPI_Send ()

Definition at line 56 of file mpi_dummy.c.

Referenced by ChessBoardIndexing(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), MakeDir(), ReadfromAsciiFile(), and ReadfromFile().

Macros

- `#define MPI_COMM_WORLD 0`
- `#define MPI_DOUBLE 2`
- `#define MPI_CHAR 1`
- `#define MPI_LONG 3`
- `#define MPI_INT 0`
- `#define MPI_MIN 0`
- `#define MPI_MAX 0`
- `#define MPI_SUM 0`

Typedefs

- `typedef int MPI_Request`
- `typedef int MPI_Status`

Functions

- `void MPI_Comm_rank ()`
- `void MPI_Barrier ()`
- `void MPI_Comm_size ()`
- `void MPI_Init ()`
- `void MPI_Finalize ()`
- `void MPI_Bcast ()`
- `void MPI_Isend ()`
- `void MPI_Irecv ()`
- `void MPI_Allreduce ()`
- `void MPI_Send ()`
- `void MPI_Recv ()`
- `void MPI_Wait ()`

4.15.1 Detailed Description

Declaration of fake MPI functions for sequential built.

There are used instead of the true MPI functions in the case of a sequential built (see makefile).

Definition in file [mpi_dummy.h](#).

4.15.2 Macro Definition Documentation

4.15.2.1 `#define MPI_CHAR 1`

Definition at line 12 of file [mpi_dummy.h](#).

4.15.2.2 `#define MPI_COMM_WORLD 0`

Definition at line 10 of file [mpi_dummy.h](#).

Referenced by [AccreteOntoPlanets\(\)](#), [AccretePebblesOntoPlanets\(\)](#), [AdvanceSystemFromDisk\(\)](#), [AlgoGas\(\)](#), [ChessBoardIndexing\(\)](#), [CommunicateBoundaries\(\)](#), [DampingTW04\(\)](#), [FillForcesArrays\(\)](#), [GasMomentum\(\)](#), [GasTotalEnergy\(\)](#), [GasTotalMass\(\)](#), [Initialization\(\)](#), [main\(\)](#), [MakeDir\(\)](#), [mpi_make1Dprofile\(\)](#), [ReadfromAsciiFile\(\)](#), [ReadfromFile\(\)](#), [SendOutput\(\)](#), [SuccessiveOverrelaxation\(\)](#), [SynchronizeOverlapFields\(\)](#), [ThicknessSmoothing\(\)](#), and [WriteDiskPolar\(\)](#).

4.15.2.3 #define MPI_DOUBLE 2

Definition at line 11 of file mpi_dummy.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), CommunicateBoundaries(), DampingTW04(), FillForcesArrays(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), MPI_Allreduce(), mpi_make1Dprofile(), SuccessiveOverrelaxation(), SynchronizeOverlapFields(), and ThicknessSmoothing().

4.15.2.4 #define MPI_INT 0

Definition at line 14 of file mpi_dummy.h.

Referenced by AccretePebblesOntoPlanets(), AlgoGas(), ChessBoardIndexing(), MakeDir(), MPI_Allreduce(), ReadfromAsciiFile(), and ReadfromFile().

4.15.2.5 #define MPI_LONG 3

Definition at line 13 of file mpi_dummy.h.

4.15.2.6 #define MPI_MAX 0

Definition at line 16 of file mpi_dummy.h.

Referenced by AlgoGas(), and ThicknessSmoothing().

4.15.2.7 #define MPI_MIN 0

Definition at line 15 of file mpi_dummy.h.

4.15.2.8 #define MPI_SUM 0

Definition at line 17 of file mpi_dummy.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), DampingTW04(), FillForcesArrays(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), and SuccessiveOverrelaxation().

4.15.3 Typedef Documentation**4.15.3.1 typedef int MPI_Request**

Definition at line 19 of file mpi_dummy.h.

4.15.3.2 typedef int MPI_Status

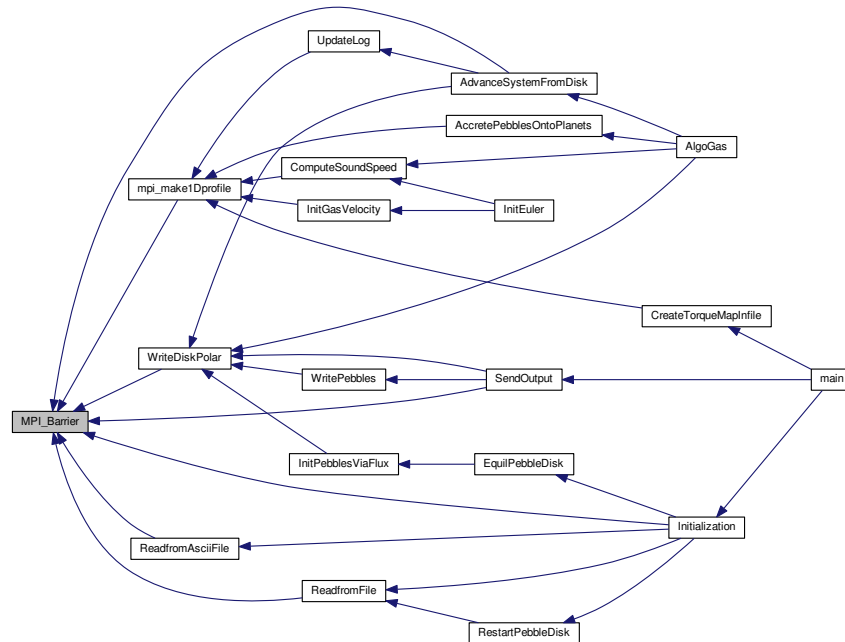
Definition at line 20 of file mpi_dummy.h.

4.15.4 Function Documentation**4.15.4.1 void MPI_Allreduce ()****4.15.4.2 void MPI_Barrier ()**

Definition at line 64 of file mpi_dummy.c.

Referenced by AdvanceSystemFromDisk(), Initialization(), mpi_make1Dprofile(), ReadfromAsciiFile(), ReadfromFile(), SendOutput(), and WriteDiskPolar().

Here is the caller graph for this function:

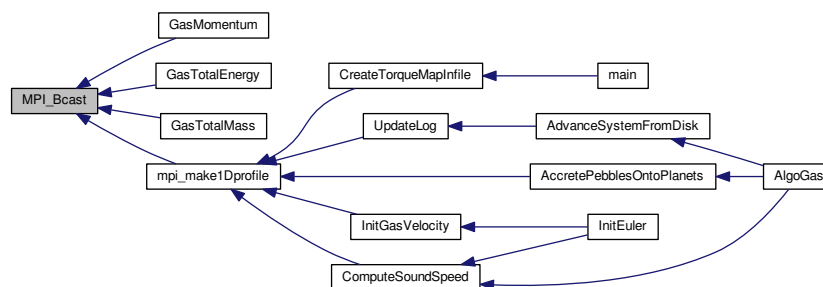


4.15.4.3 void MPI_Bcast ()

Definition at line 44 of file mpi_dummy.c.

Referenced by GasMomentum(), GasTotalEnergy(), GasTotalMass(), and mpi_make1Dprofile().

Here is the caller graph for this function:

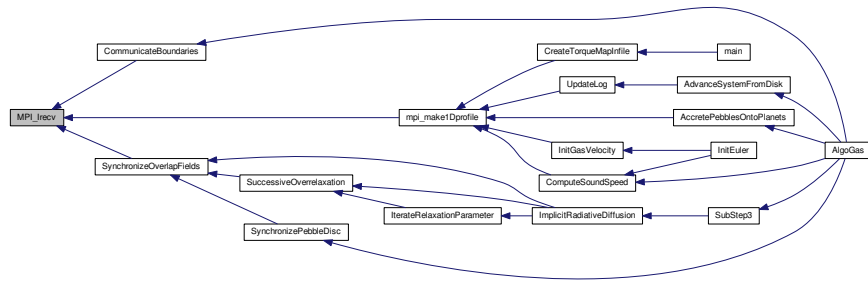


4.15.4.4 void MPI_Comm_rank ()

4.15.4.5 void MPI_Comm_size ()

Referenced by `CommunicateBoundaries()`, `mpi_make1Dprofile()`, and `SynchronizeOverlapFields()`.

Here is the caller graph for this function:

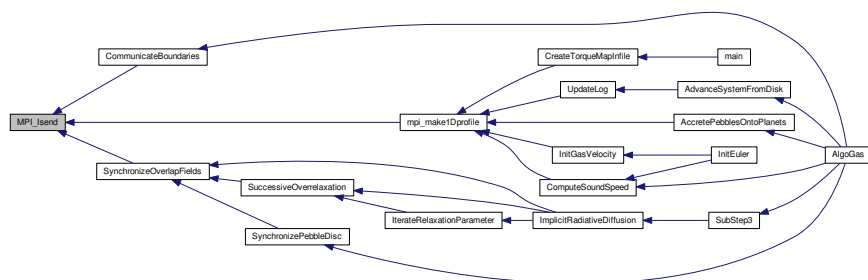


4.15.4.9 void MPI_Isend ()

Definition at line 48 of file `mpi_dummy.c`.

Referenced by `CommunicateBoundaries()`, `mpi_make1Dprofile()`, and `SynchronizeOverlapFields()`.

Here is the caller graph for this function:

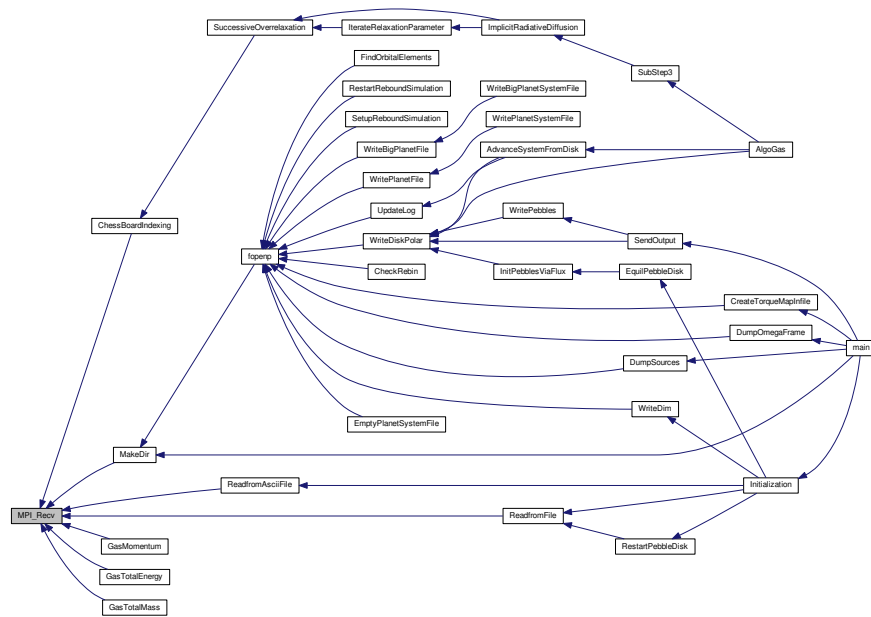


4.15.4.10 void MPI_Recv ()

Definition at line 60 of file `mpi_dummy.c`.

Referenced by `ChessBoardIndexing()`, `GasMomentum()`, `GasTotalEnergy()`, `GasTotalMass()`, `MakeDir()`, `ReadfromAsciiFile()`, and `ReadfromFile()`.

Here is the caller graph for this function:

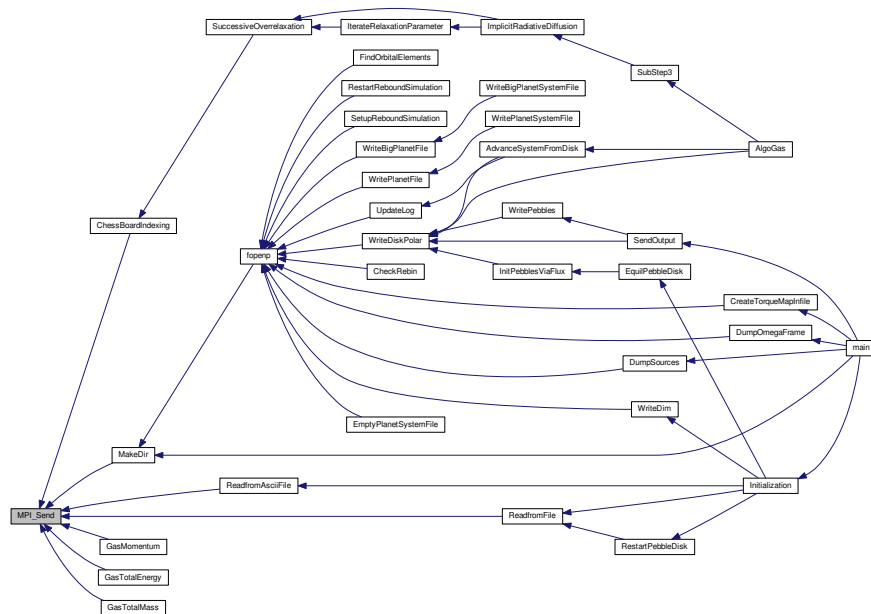


4.15.4.11 void MPI_Send ()

Definition at line 56 of file mpi_dummy.c.

Referenced by ChessBoardIndexing(), GasMomentum(), GasTotalEnergy(), GasTotalMass(), MakeDir(), ReadfromAsciiFile(), and ReadfromFile().

Here is the caller graph for this function:

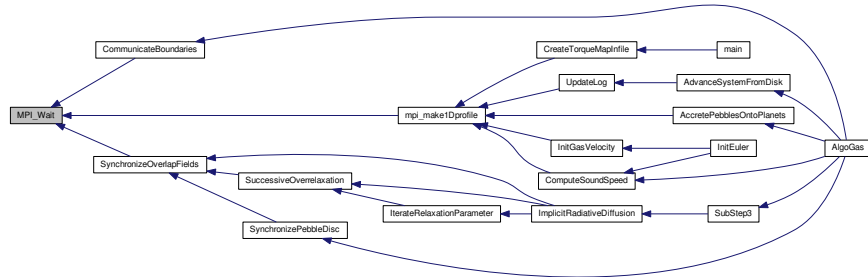


4.15.4.12 void MPI_Wait ()

Definition at line 68 of file `mpi_dummy.c`.

Referenced by `CommunicateBoundaries()`, `mpi_make1Dprofile()`, and `SynchronizeOverlapFields()`.

Here is the caller graph for this function:

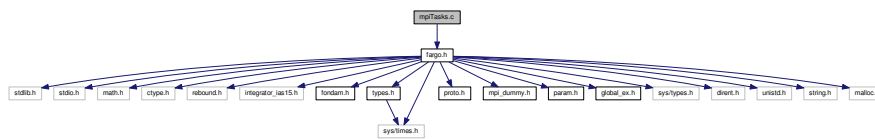


4.16 mpiTasks.c File Reference

Contains the function to create a 1D azimuthally-averaged array from a polar array.

```
#include "fargo.h"
```

Include dependency graph for `mpiTasks.c`:



Functions

- void `mpi_make1Dprofile` (`real` *gridfield, `real` *axifield)

4.16.1 Detailed Description

Contains the function to create a 1D azimuthally-averaged array from a polar array.

Works also for a MPI-split grid.

Author

Taken from FARGO-ADSG by Clément Baruteau.

Definition in file `mpiTasks.c`.

4.16.2 Function Documentation

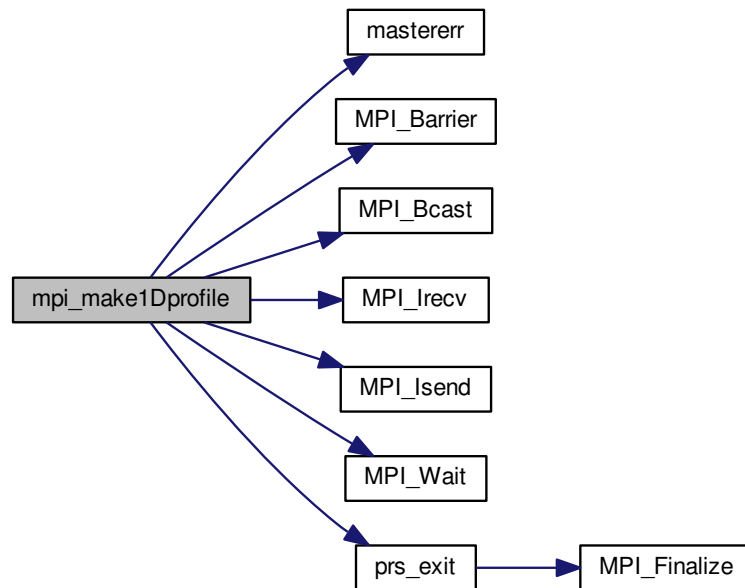
4.16.2.1 void mpi_make1Dprofile (`real`* gridfield, `real`* axifield)

Definition at line 12 of file `mpiTasks.c`.

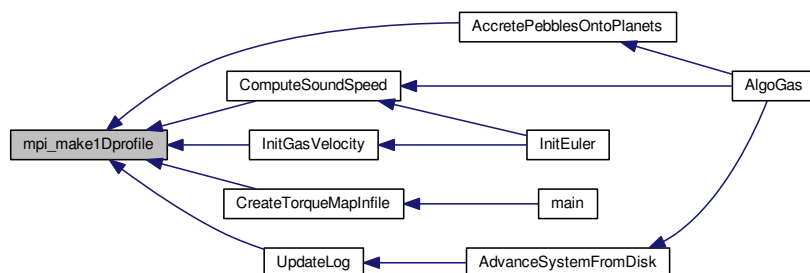
References CPU_Number, CPU_Rank, fargostat, GLOBALNRAD, IMIN, mastererr(), Max_or_active, MPI_Barrier(), MPI_Bcast(), MPI_COMM_WORLD, MPI_DOUBLE, MPI_Irecv(), MPI_Isend(), MPI_Wait(), NRAD, NSEC, prs_exit(), and Zero_or_active.

Referenced by AccretePebblesOntoPlanets(), ComputeSoundSpeed(), CreateTorqueMapInfile(), InitGasVelocity(), and UpdateLog().

Here is the call graph for this function:



Here is the caller graph for this function:

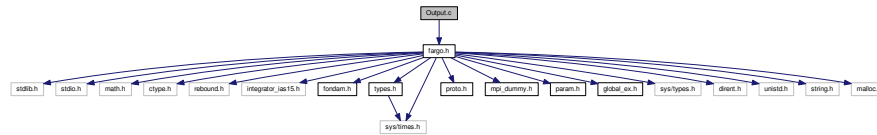


4.17 Output.c File Reference

Contains most of the functions that write the output files.

```
#include "fargo.h"
```

Include dependency graph for Output.c:



Functions

- void [EmptyPlanetSystemFile](#) ([PlanetarySystem](#) *sys)
- void [WritePlanetFile](#) (int [TimeStep](#), int n)
- void [WritePlanetSystemFile](#) ([PlanetarySystem](#) *sys, int t)
- void [WriteBigPlanetFile](#) (int [TimeStep](#), int n)
- void [WriteBigPlanetSystemFile](#) ([PlanetarySystem](#) *sys, int t)
- [real](#) [GetfromPlanetFile](#) (int [TimeStep](#), int column, int n)
- void [RestartPlanetarySystem](#) (int timestep, [PlanetarySystem](#) *sys)
- void [WriteDiskPolar](#) ([PolarGrid](#) *array, int number)
- void [WriteDim](#) ()
- void [SendOutput](#) (int index, [PolarGrid](#) *dens, [PolarGrid](#) *gasvr, [PolarGrid](#) *gasvt, [PolarGrid](#) *gasenerg, [PolarGrid](#) *label)
- void [ActualizeQbalance](#) ()
- void [DumpOmegaFrame](#) (int [TimeStep](#))
Writes the angular velocity of the coordinate system.
- [real](#) [GetOmegaFrame](#) (int [TimeStep](#))
Finds the angular velocity of the coordinate system in 'omegaframe.dat' at a given TimeStep.

Variables

- static [real](#) [Xplanet](#)
- static [real](#) [Yplanet](#)
- static [real](#) [VXplanet](#)
- static [real](#) [VYplanet](#)
- static [real](#) [MplanetVirtual](#)
- [real](#) [LostMass](#)
- [real](#) [OmegaFrame](#)
- [boolean](#) [Write_Density](#)
- [boolean](#) [Write_Velocity](#)
- [boolean](#) [IsDisk](#)

4.17.1 Detailed Description

Contains most of the functions that write the output files.

In addition to the writing of hydrodynamics files (handled by [SendOutput](#) ()), this file also contains the functions that update the planet.dat and bigplanet.dat files, and the functions that seek information about the planets at a restart.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [Output.c](#).

4.17.2 Function Documentation

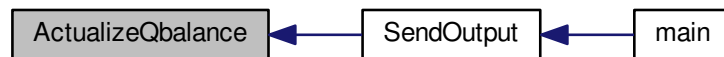
4.17.2.1 void ActualizeQbalance ()

Definition at line 239 of file Output.c.

References polargrid::Field, polargrid::Nrad, polargrid::Nsec, Qbalance, Qminus, and Qplus.

Referenced by SendOutput().

Here is the caller graph for this function:



4.17.2.2 void DumpOmegaFrame (int TimeStep)

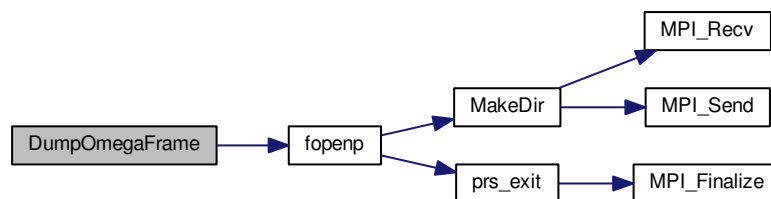
Writes the angular velocity of the coordinate system.

Definition at line 259 of file Output.c.

References CPU_Master, fopen(), OmegaFrame, and OUTPUTDIR.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:

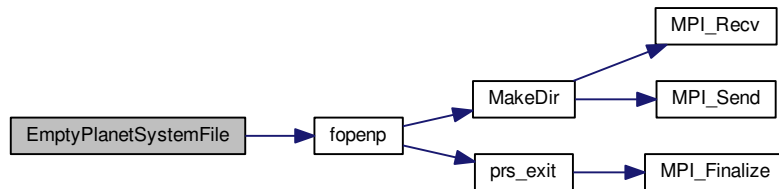


4.17.2.3 void EmptyPlanetSystemFile (PlanetarySystem * sys)

Definition at line 21 of file Output.c.

References CPU_Master, fopenp(), planetary_system::nb, and OUTPUTDIR.

Here is the call graph for this function:



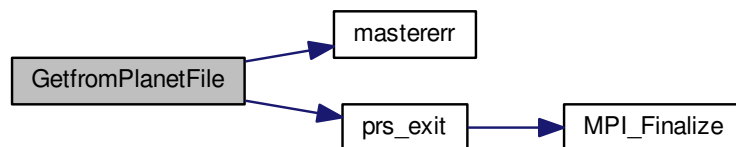
4.17.2.4 real GetfromPlanetFile (int TimeStep, int column, int n)

Definition at line 101 of file Output.c.

References mastererr(), OUTPUTDIR, and prs_exit().

Referenced by RestartPlanetarySystem().

Here is the call graph for this function:



Here is the caller graph for this function:



4.17.2.5 `real GetOmegaFrame (int TimeStep)`

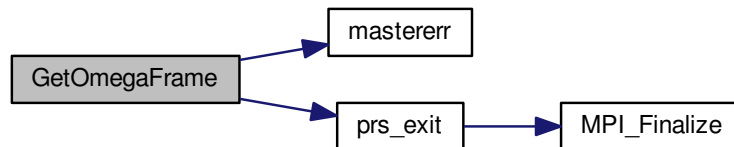
Finds the angular velocity of the coordinate system in 'omegaframe.dat' at a given TimeStep.

Definition at line 279 of file Output.c.

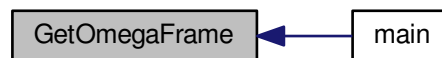
References `mastererr()`, `OUTPUTDIR`, and `prs_exit()`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.17.2.6 `void RestartPlanetarySystem (int timestep, PlanetarySystem * sys)`

Definition at line 139 of file Output.c.

References `GetfromPlanetFile()`, and `planetary_system::x`.

Here is the call graph for this function:



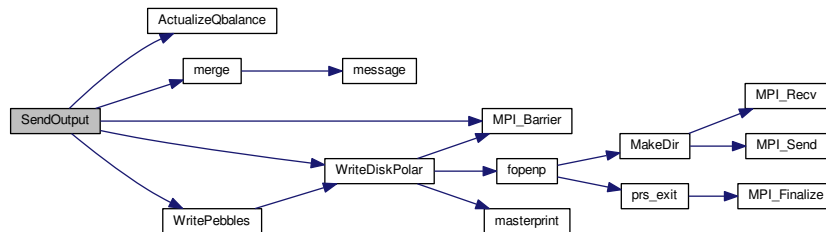
4.17.2.7 `void SendOutput (int index, PolarGrid * dens, PolarGrid * gasvr, PolarGrid * gasvt, PolarGrid * gasenerg, PolarGrid * label)`

Definition at line 202 of file Output.c.

References ActualizeQbalance(), AdvecteLabel, CPU_Master, CPU_Number, DivergenceVelocity, IsDisk, merge(), Merge, MPI_Barrier(), MPI_COMM_WORLD, NO, Pebbles, Qbalance, Qplus, Temperature, Write_Density, Write_Divergence, Write_Energy, Write_Qbalance, Write_Qplus, Write_Temperature, Write_Velocity, WriteDiskPolar(), WritePebbles(), and YES.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



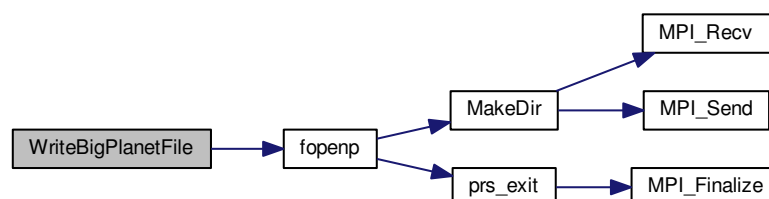
4.17.2.8 void WriteBigPlanetFile (int TimeStep, int n)

Definition at line 71 of file Output.c.

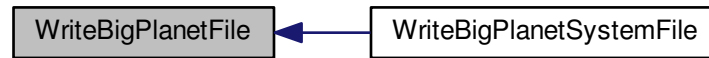
References CPU_Master, fopenp(), LostMass, MplanetVirtual, OmegaFrame, OUTPUTDIR, PhysicalTime, V_xplanet, V_yplanet, Xplanet, and Yplanet.

Referenced by WriteBigPlanetSystemFile().

Here is the call graph for this function:



Here is the caller graph for this function:

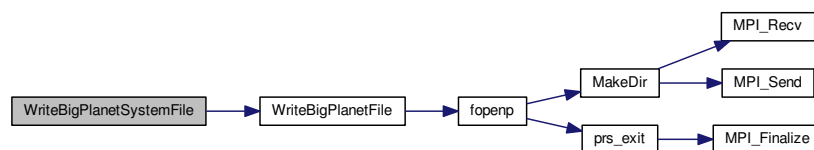


4.17.2.9 void WriteBigPlanetSystemFile (PlanetarySystem * sys, int t)

Definition at line 85 of file Output.c.

References `MplanetVirtual`, `planetary_system::nb`, `VXplanet`, `VYplanet`, `WriteBigPlanetFile()`, `Xplanet`, and `Yplanet`.

Here is the call graph for this function:



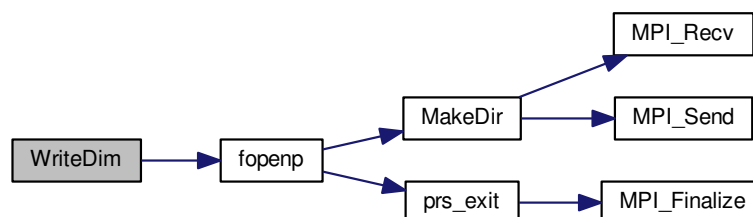
4.17.2.10 void WriteDim ()

Definition at line 191 of file Output.c.

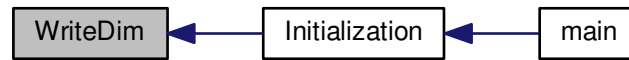
References `CPU_Master`, `fopenp()`, `GLOBALNRAD`, `NINTERM`, `NSEC`, `NTOT`, `OUTPUTDIR`, and `RMAX`.

Referenced by `Initialization()`.

Here is the call graph for this function:



Here is the caller graph for this function:



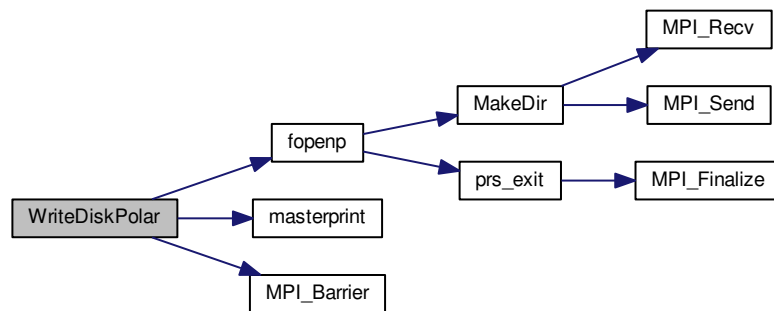
4.17.2.11 void WriteDiskPolar (PolarGrid * array, int number)

Definition at line 153 of file Output.c.

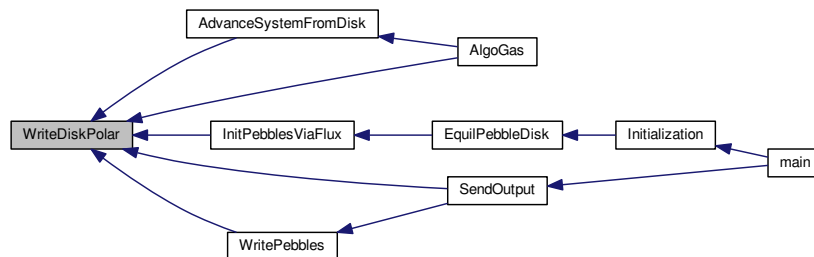
References CPU_Master, CPU_Number, CPU_Rank, CPUOVERLAP, polargrid::Field, fopenp(), masterprint(), MPI_Barrier(), MPI_COMM_WORLD, and OUTPUTDIR.

Referenced by AdvanceSystemFromDisk(), AlgoGas(), InitPebblesViaFlux(), SendOutput(), and WritePebbles().

Here is the call graph for this function:



Here is the caller graph for this function:



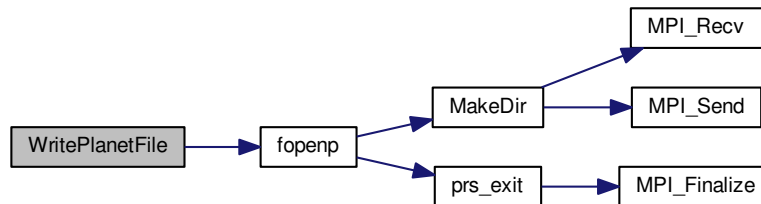
4.17.2.12 void WritePlanetFile (int *TimeStep*, int *n*)

Definition at line 36 of file Output.c.

References CPU_Master, fopenp(), LostMass, MplanetVirtual, OmegaFrame, OUTPUTDIR, PhysicalTime, $V \leftrightarrow$ Xplanet, VYplanet, Xplanet, and Yplanet.

Referenced by WritePlanetSystemFile().

Here is the call graph for this function:



Here is the caller graph for this function:

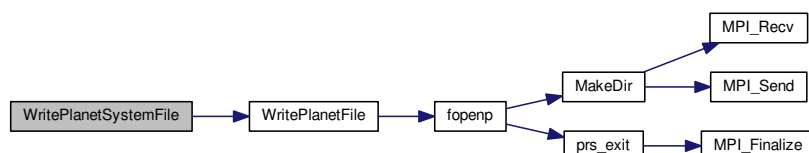


4.17.2.13 void WritePlanetSystemFile (PlanetarySystem * *sys*, int *t*)

Definition at line 54 of file Output.c.

References MplanetVirtual, planetary_system::nb, VXplanet, VYplanet, WritePlanetFile(), Xplanet, and Yplanet.

Here is the call graph for this function:



4.17.3 Variable Documentation

4.17.3.1 **boolean** IsDisk

Definition at line 30 of file Interpret.c.

Referenced by SendOutput().

4.17.3.2 **real** LostMass

Definition at line 32 of file TransportEuler.c.

Referenced by OneWindRad(), WriteBigPlanetFile(), and WritePlanetFile().

4.17.3.3 **real** MplanetVirtual [static]

Definition at line 17 of file Output.c.

Referenced by WriteBigPlanetFile(), WriteBigPlanetSystemFile(), WritePlanetFile(), and WritePlanetSystemFile().

4.17.3.4 **real** OmegaFrame

Definition at line 20 of file global.h.

Referenced by DumpOmegaFrame(), WriteBigPlanetFile(), and WritePlanetFile().

4.17.3.5 **real** VXplanet [static]

Definition at line 17 of file Output.c.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), WriteBigPlanetFile(), WriteBigPlanetSystemFile(), WritePlanetFile(), and WritePlanetSystemFile().

4.17.3.6 **real** VYplanet [static]

Definition at line 17 of file Output.c.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), WriteBigPlanetFile(), WriteBigPlanetSystemFile(), WritePlanetFile(), and WritePlanetSystemFile().

4.17.3.7 **boolean** Write_Density

Definition at line 31 of file Interpret.c.

Referenced by ReadVariables(), and SendOutput().

4.17.3.8 **boolean** Write_Velocity

Definition at line 31 of file Interpret.c.

Referenced by ReadVariables(), and SendOutput().

4.17.3.9 **real** Xplanet [static]

Definition at line 17 of file Output.c.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), ParametricAccretion(), WriteBigPlanetFile(), WriteBigPlanetSystemFile(), WritePlanetFile(), and WritePlanetSystemFile().

- [real CAVITYRADIUS](#)
- [real CAVITYRATIO](#)
- [real CAVITYWIDTH](#)
- [real TRANSITIONRADIUS](#)
- [real TRANSITIONRATIO](#)
- [real TRANSITIONWIDTH](#)
- [real LAMBDA DOUBLING](#)
- [char ENERGY EQUATION](#) [512]
- [char WRITE TEMPERATURE](#) [512]
- [char WRITE ENERGY](#) [512]
- [char WRITE DIVV](#) [512]
- [char WRITE Q PLUS](#) [512]
- [char WRITE Q BALANCE](#) [512]
- [real ADIABIND](#)
- [real COOLING TIME](#)
- [char STELLAR IRRADIATION](#) [512]
- [real OPACITY DROP](#)
- [real EFFECTIVE TEMPERATURE](#)
- [real STELLAR RADIUS](#)
- [real DISC ALBEDO](#)
- [real PARAMETRIC OPACITY](#)
- [char INITIALIZE FROM FILE](#) [512]
- [char DENS IN FILE](#) [512]
- [char VRAD IN FILE](#) [512]
- [char VTHETA IN FILE](#) [512]
- [char TEMPER IN FILE](#) [512]
- [char DAMPTOWARDS](#) [512]
- [real DAMPING R MIN FRAC](#)
- [real DAMPING R MAX FRAC](#)
- [real DAMPING PERIOD FRAC](#)
- [int NOUT ELEMENTS](#)
- [real PLANETARY DENSITY](#)
- [char RESOLVE COLLISIONS](#) [512]
- [int TARGET NPL](#)
- [real IAS15 PRECISION](#)
- [real IAS15 MINDT](#)
- [char WRITE TORQUE FILES](#) [512]
- [real HILL CUT](#)
- [real VERTICAL DAMPING](#)
- [char PLANETS FEEL DISK](#) [512]
- [real ACCRETION RATE](#)
- [char PEBBLE ACCRETION](#) [512]
- [char BACK REACTION](#) [512]
- [char ACCRETIONAL HEATING](#) [512]
- [char WRITE ETA](#) [512]
- [real PEBBLE FLUX](#)
- [real PEBBLE ALPHA](#)
- [real PEBBLE COAGULATION](#)
- [real PEBBLE BULK DENS](#)
- [real SCHMIDT NUMBER](#)
- [char PARTICLE DIFFUSION](#) [512]
- [int HEATING DELAY](#)
- [real PARAMETRIC ACCRETION](#)
- [char TORQUE MAP IN FILE](#) [512]
- [int GET TORQUE FOR PLANET](#)

4.18.1 Detailed Description

Created automatically during compilation from [var.c](#).

Do not edit. See Perl script "varparser.pl" for details.

Definition in file [param.h](#).

4.18.2 Variable Documentation

4.18.2.1 char ACCRETIONALHEATING[512]

Definition at line 84 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.2 real ACCRETIONRATE

Definition at line 81 of file param_noex.h.

Referenced by InitVariables(), RestartReboundSimulation(), and SetupReboundSimulation().

4.18.2.3 real ADIABIND

Definition at line 54 of file param_noex.h.

Referenced by ComputePressureField(), ComputeSoundSpeed(), ComputeTemperatureField(), CreateTorque↔MapInfile(), Energy(), ImplicitRadiativeDiffusion(), InitEuler(), InitGasVelocity(), Initialization(), InitPebblesViaFlux(), InitRadiatDiffusionFields(), InitVariables(), SubStep3(), and UpdateLog().

4.18.2.4 char ADVLABEL[512]

Definition at line 13 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.5 real ALPHAVISCOSITY

Definition at line 26 of file param_noex.h.

Referenced by CreateTorqueMapInfile(), FViscosity(), InitVariables(), and ReadVariables().

4.18.2.6 real ASPECTRATIO

Definition at line 24 of file param_noex.h.

Referenced by AspectRatio(), Energy(), FViscosity(), InitVariables(), and TellEverything().

4.18.2.7 char BACKREACTION[512]

Definition at line 83 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.8 real CAVITYRADIUS

Definition at line 41 of file param_noex.h.

Referenced by FViscosity(), InitVariables(), and Sigma().

4.18.2.9 real CAVITYRATIO

Definition at line 42 of file param_noex.h.

Referenced by FViscosity(), InitVariables(), and Sigma().

4.18.2.10 real CAVITYWIDTH

Definition at line 43 of file param_noex.h.

Referenced by FViscosity(), and InitVariables().

4.18.2.11 real COOLINGTIME

Definition at line 55 of file param_noex.h.

Referenced by InitCoolingTime(), InitVariables(), and ReadVariables().

4.18.2.12 real DAMPINGPERIODFRAC

Definition at line 70 of file param_noex.h.

Referenced by InitVariables(), and SetWaveKillingZones().

4.18.2.13 real DAMPINGRMAXFRAC

Definition at line 69 of file param_noex.h.

Referenced by DampingBoundary(), DampPebbles(), InitVariables(), and SetWaveKillingZones().

4.18.2.14 real DAMPINGRMINFRAC

Definition at line 68 of file param_noex.h.

Referenced by DampingBoundary(), DampPebbles(), InitVariables(), and SetWaveKillingZones().

4.18.2.15 char DAMPTOWARDS[512]

Definition at line 67 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.16 char DENSINFILE[512]

Definition at line 63 of file param_noex.h.

Referenced by Initialization(), and InitVariables().

4.18.2.17 real DISCALBEDO

Definition at line 60 of file param_noex.h.

Referenced by CalculateQirr(), and InitVariables().

4.18.2.18 char DISK[512]

Definition at line 31 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.19 real DT

Definition at line 7 of file param_noex.h.

Referenced by AccretePebblesOntoPlanets(), AlgoGas(), InitVariables(), ParametricAccretion(), TellEverything(), and TellNbOutputs().

4.18.2.20 real ECCENTRICITY

Definition at line 40 of file param_noex.h.

Referenced by InitPlanetarySystem(), and InitVariables().

4.18.2.21 real EFFECTIVETEMPERATURE

Definition at line 58 of file param_noex.h.

Referenced by CalculateQirr(), and InitVariables().

4.18.2.22 char ENERGYEQUATION[512]

Definition at line 48 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.23 char EXCLUDEHILL[512]

Definition at line 37 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.24 real FLARINGINDEX

Definition at line 39 of file param_noex.h.

Referenced by ComputeSoundSpeed(), Energy(), ImposeKeplerianEdges(), InitCoolingTime(), InitGasVelocity(), and InitVariables().

4.18.2.25 char FRAME[512]

Definition at line 32 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.26 int GETTORQUEFORPLANET

Definition at line 95 of file param_noex.h.

Referenced by FillForcesArrays(), InitVariables(), and ReadVariables().

4.18.2.27 char GRIDSPACING[512]

Definition at line 17 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.28 int HEATINGDELAY

Definition at line 92 of file param_noex.h.

Referenced by AccretePebblesOntoPlanets(), InitVariables(), and ParametricAccretion().

4.18.2.29 real HILLCUT

Definition at line 78 of file param_noex.h.

Referenced by FillForcesArrays(), and InitVariables().

4.18.2.30 real IAS15MINDT

Definition at line 76 of file param_noex.h.

Referenced by InitVariables(), and SetupIntegratorParams().

4.18.2.31 real IAS15PRECISION

Definition at line 75 of file param_noex.h.

Referenced by InitVariables(), and SetupIntegratorParams().

4.18.2.32 real IMPOSEDDISKDRIFT

Definition at line 38 of file param_noex.h.

Referenced by ApplyOuterSourceMass(), InitGasVelocity(), InitVariables(), and SubStep1().

4.18.2.33 char INDIRECTTERM[512]

Definition at line 36 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.34 char INITIALIZEFROMFILE[512]

Definition at line 62 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.35 real LAMBDA DOUBLING

Definition at line 47 of file param_noex.h.

Referenced by AspectRatio(), FViscosity(), and InitVariables().

4.18.2.36 real MASSTAPER

Definition at line 16 of file param_noex.h.

Referenced by AlgoGas(), and InitVariables().

4.18.2.37 int NINTERM

Definition at line 9 of file param_noex.h.

Referenced by GiveTimeInfo(), InitVariables(), main(), TellEverything(), TellNbOutputs(), and WriteDim().

4.18.2.38 int NOUTELEMENTS

Definition at line 71 of file param_noex.h.

Referenced by InitVariables(), and main().

4.18.2.39 int NRAD

Definition at line 18 of file param_noex.h.

Referenced by CheckRebin(), ChessBoardIndexing(), FillCoolingTime(), FillEnergy(), FillPolar1DArrays(), FillQplus(), FillSigma(), InitComputeAccel(), InitEuler(), InitPebbleArrays(), InitRadiatDiffusionFields(), InitTransport(), InitVariables(), InitViscosity(), main(), mpi_make1Dprofile(), SplitDomain(), and TellEverything().

4.18.2.40 int NSEC

Definition at line 19 of file param_noex.h.

Referenced by AllocateComm(), CheckRebin(), CommunicateBoundaries(), ConditionCFL(), FillPolar1DArrays(), InitComputeAccel(), InitEuler(), InitPebbleArrays(), InitRadiatDiffusionFields(), InitTransport(), InitVariables(), InitViscosity(), main(), mpi_make1Dprofile(), NonReflectingBoundary(), SynchronizeOverlapFields(), TellEverything(), and WriteDim().

4.18.2.41 int NTOT

Definition at line 10 of file param_noex.h.

Referenced by InitVariables(), main(), TellEverything(), and WriteDim().

4.18.2.42 real OMEGA FRAME

Definition at line 30 of file param_noex.h.

Referenced by InitVariables(), and main().

4.18.2.43 real OPACITY DROP

Definition at line 57 of file param_noex.h.

Referenced by `CalculateQirr()`, `CalculateQminus()`, and `InitVariables()`.

4.18.2.44 `char OPENINNERBOUNDARY[512]`

Definition at line 12 of file `param_noex.h`.

Referenced by `InitVariables()`, and `ReadVariables()`.

4.18.2.45 `char OUTERSOURCEMASS[512]`

Definition at line 33 of file `param_noex.h`.

Referenced by `InitVariables()`, and `ReadVariables()`.

4.18.2.46 `char OUTPUTDIR[512]`

Definition at line 11 of file `param_noex.h`.

Referenced by `AdvanceSystemFromDisk()`, `CheckRebin()`, `CreateTorqueMapInfile()`, `DumpOmegaFrame()`, `DumpSources()`, `EmptyPlanetSystemFile()`, `FillPolar1DArrays()`, `FindOrbitalElements()`, `fopenp()`, `GetfromPlanetFile()`, `GetOmegaFrame()`, `InitPebblesViaFlux()`, `InitVariables()`, `main()`, `merge()`, `OutputNbodySimulation()`, `ReadfromFile()`, `ReadPrevDim()`, `ReadVariables()`, `RestartReboundSimulation()`, `SetupReboundSimulation()`, `UpdateLog()`, `WriteBigPlanetFile()`, `WriteDim()`, `WriteDiskPolar()`, and `WritePlanetFile()`.

4.18.2.47 `real PARAMETRICACCRETION`

Definition at line 93 of file `param_noex.h`.

Referenced by `InitVariables()`, `ParametricAccretion()`, and `ReadVariables()`.

4.18.2.48 `real PARAMETRICOPACITY`

Definition at line 61 of file `param_noex.h`.

Referenced by `CreateTorqueMapInfile()`, `InitVariables()`, and `OpacityProfile()`.

4.18.2.49 `char PARTICLEDIFFUSION[512]`

Definition at line 91 of file `param_noex.h`.

Referenced by `InitVariables()`, and `ReadVariables()`.

4.18.2.50 `char PEBBLEACCRETION[512]`

Definition at line 82 of file `param_noex.h`.

Referenced by `InitVariables()`, and `ReadVariables()`.

4.18.2.51 `real PEBBLEALPHA`

Definition at line 87 of file `param_noex.h`.

Referenced by `AccretePebblesOntoPlanets()`, `FillForcesArrays()`, and `InitVariables()`.

4.18.2.52 **real** PEBBLEBULKDENS

Definition at line 89 of file param_noex.h.

Referenced by InitPebbleArrays(), and InitVariables().

4.18.2.53 **real** PEBBLECOAGULATION

Definition at line 88 of file param_noex.h.

Referenced by InitPebblesViaFlux(), and InitVariables().

4.18.2.54 **real** PEBBLEFLUX

Definition at line 86 of file param_noex.h.

Referenced by InitPebblesViaFlux(), and InitVariables().

4.18.2.55 **real** PLANETARYDENSITY

Definition at line 72 of file param_noex.h.

Referenced by AccretePebblesOntoPlanets(), InitVariables(), ParametricAccretion(), and SetupReboundSimulation().

4.18.2.56 **char** PLANETCONFIG[512]

Definition at line 15 of file param_noex.h.

Referenced by InitVariables(), and main().

4.18.2.57 **char** PLANETSFEELDISK[512]

Definition at line 80 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.58 **real** RELEASEDATE

Definition at line 29 of file param_noex.h.

Referenced by InitVariables().

4.18.2.59 **real** RELEASERADIUS

Definition at line 28 of file param_noex.h.

Referenced by InitVariables().

4.18.2.60 **char** RESOLVECOLLISIONS[512]

Definition at line 73 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.61 real RMAX

Definition at line 21 of file param_noex.h.

Referenced by DampingBoundary(), DampPebbles(), DiscardParticlesDist(), FillPolar1DArrays(), InitLabel(), InitVariables(), SetWaveKillingZones(), TellEverything(), and WriteDim().

4.18.2.62 real RMIN

Definition at line 20 of file param_noex.h.

Referenced by DampingBoundary(), DampPebbles(), FillPolar1DArrays(), InitLabel(), InitVariables(), SetWaveKillingZones(), and TellEverything().

4.18.2.63 real ROCHESMOOTHING

Definition at line 23 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.64 real SCHMIDTNUMBER

Definition at line 90 of file param_noex.h.

Referenced by InitVariables(), and ParticleDiffusion().

4.18.2.65 real SIGMA0

Definition at line 8 of file param_noex.h.

Referenced by Energy(), InitGasVelocity(), InitQplus(), InitVariables(), Sigma(), and TellEverything().

4.18.2.66 real SIGMASLOPE

Definition at line 27 of file param_noex.h.

Referenced by ApplyOuterSourceMass(), Energy(), ImposeKeplerianEdges(), InitGasVelocity(), InitQplus(), InitVariables(), Sigma(), and SubStep1().

4.18.2.67 char STELLARIRRADIATION[512]

Definition at line 56 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.68 real STELLARRADIUS

Definition at line 59 of file param_noex.h.

Referenced by CalculateFlaring(), CalculateQirr(), and InitVariables().

4.18.2.69 int TARGETNPL

Definition at line 74 of file param_noex.h.

Referenced by InitVariables(), main(), and ReadVariables().

4.18.2.70 char TEMPERINFILE[512]

Definition at line 66 of file param_noex.h.

Referenced by Initialization(), and InitVariables().

4.18.2.71 real THICKNESSSMOOTHING

Definition at line 22 of file param_noex.h.

Referenced by FillForcesArrays(), InitVariables(), ReadVariables(), and ThicknessSmoothing().

4.18.2.72 char TORQUEMAPINFILE[512]

Definition at line 94 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.73 real TRANSITIONRADIUS

Definition at line 44 of file param_noex.h.

Referenced by AspectRatio(), and InitVariables().

4.18.2.74 real TRANSITIONRATIO

Definition at line 45 of file param_noex.h.

Referenced by AspectRatio(), and InitVariables().

4.18.2.75 real TRANSITIONWIDTH

Definition at line 46 of file param_noex.h.

Referenced by AspectRatio(), and InitVariables().

4.18.2.76 char TRANSPORT[512]

Definition at line 14 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.77 real VERTICALDAMPING

Definition at line 79 of file param_noex.h.

Referenced by DampingTW04(), and InitVariables().

4.18.2.78 real VISCOSITY

Definition at line 25 of file param_noex.h.

Referenced by CreateTorqueMapInfile(), FViscosity(), InitVariables(), and ReadVariables().

4.18.2.79 char VRADINFILE[512]

Definition at line 64 of file param_noex.h.

Referenced by Initialization(), and InitVariables().

4.18.2.80 char VTHETAFILE[512]

Definition at line 65 of file param_noex.h.

Referenced by Initialization(), and InitVariables().

4.18.2.81 char WRITEDENSITY[512]

Definition at line 34 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.82 char WRITEDIVV[512]

Definition at line 51 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.83 char WRITEENERGY[512]

Definition at line 50 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.84 char WRITEETA[512]

Definition at line 85 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.85 char WRITEQBALANCE[512]

Definition at line 53 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.86 char WRITEQPLUS[512]

Definition at line 52 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.87 char WRITETEMPERATURE[512]

Definition at line 49 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.88 char WRITETORQUEFILES[512]

Definition at line 77 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.18.2.89 char WRITEVELOCITY[512]

Definition at line 35 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19 param_noex.h File Reference

Created automatically during compilation from [var.c](#).

Variables

- [real DT](#)
- [real SIGMA0](#)
- [int NINTERM](#)
- [int NTOT](#)
- [char OUTPUTDIR \[512\]](#)
- [char OPENINNERBOUNDARY \[512\]](#)
- [char ADVLABEL \[512\]](#)
- [char TRANSPORT \[512\]](#)
- [char PLANETCONFIG \[512\]](#)
- [real MASSTAPER](#)
- [char GRIDSPACING \[512\]](#)
- [int NRAD](#)
- [int NSEC](#)
- [real RMIN](#)
- [real RMAX](#)
- [real THICKNESSSMOOTHING](#)
- [real ROCHESSMOOTHING](#)
- [real ASPECTRATIO](#)
- [real VISCOSITY](#)
- [real ALPHAVISCOSITY](#)
- [real SIGMASLOPE](#)
- [real RELEASERADIUS](#)
- [real RELEASEDATE](#)
- [real OMEGAFRAME](#)
- [char DISK \[512\]](#)
- [char FRAME \[512\]](#)
- [char OUTERSOURCEMASS \[512\]](#)
- [char WRITEDENSITY \[512\]](#)
- [char WRITEVELOCITY \[512\]](#)
- [char INDIRECTTERM \[512\]](#)
- [char EXCLUDEHILL \[512\]](#)
- [real IMPOSEDDISKDRIFT](#)
- [real FLARINGINDEX](#)
- [real ECCENTRICITY](#)
- [real CAVITYRADIUS](#)

- [real CAVITYRATIO](#)
- [real CAVITYWIDTH](#)
- [real TRANSITIONRADIUS](#)
- [real TRANSITIONRATIO](#)
- [real TRANSITIONWIDTH](#)
- [real LAMBDA DOUBLING](#)
- [char ENERGY EQUATION](#) [512]
- [char WRITE TEMPERATURE](#) [512]
- [char WRITE ENERGY](#) [512]
- [char WRITE DIVV](#) [512]
- [char WRITE Q PLUS](#) [512]
- [char WRITE Q BALANCE](#) [512]
- [real ADIABIND](#)
- [real COOLING TIME](#)
- [char STELLAR IRRADIATION](#) [512]
- [real OPACITY DROP](#)
- [real EFFECTIVE TEMPERATURE](#)
- [real STELLAR RADIUS](#)
- [real DISC ALBEDO](#)
- [real PARAMETRIC OPACITY](#)
- [char INITIALIZE FROM FILE](#) [512]
- [char DENS IN FILE](#) [512]
- [char VRAD IN FILE](#) [512]
- [char VTHETA IN FILE](#) [512]
- [char TEMPER IN FILE](#) [512]
- [char DAMPTOWARDS](#) [512]
- [real DAMPING R MIN FRAC](#)
- [real DAMPING R MAX FRAC](#)
- [real DAMPING PERIOD FRAC](#)
- [int NOUT ELEMENTS](#)
- [real PLANETARY DENSITY](#)
- [char RESOLVE COLLISIONS](#) [512]
- [int TARGET NPL](#)
- [real IAS15 PRECISION](#)
- [real IAS15 MINDT](#)
- [char WRITETORQUEFILES](#) [512]
- [real HILLCUT](#)
- [real VERTICAL DAMPING](#)
- [char PLANETSFEELDISK](#) [512]
- [real ACCRETION RATE](#)
- [char PEBBLE ACCRETION](#) [512]
- [char BACK REACTION](#) [512]
- [char ACCRETIONAL HEATING](#) [512]
- [char WRITE ETA](#) [512]
- [real PEBBLE FLUX](#)
- [real PEBBLE ALPHA](#)
- [real PEBBLE COAGULATION](#)
- [real PEBBLE BULK DENS](#)
- [real SCHMIDT NUMBER](#)
- [char PARTICLE DIFFUSION](#) [512]
- [int HEATING DELAY](#)
- [real PARAMETRIC ACCRETION](#)
- [char TORQUE MAP IN FILE](#) [512]
- [int GETTORQUEFORPLANET](#)

4.19.1 Detailed Description

Created automatically during compilation from [var.c](#).

Do not edit. See Perl script "varparser.pl" for details.

Definition in file [param_noex.h](#).

4.19.2 Variable Documentation

4.19.2.1 char ACCRETIONALHEATING[512]

Definition at line 84 of file [param_noex.h](#).

Referenced by [InitVariables\(\)](#), and [ReadVariables\(\)](#).

4.19.2.2 real ACCRETIONRATE

Definition at line 81 of file [param_noex.h](#).

Referenced by [InitVariables\(\)](#), [RestartReboundSimulation\(\)](#), and [SetupReboundSimulation\(\)](#).

4.19.2.3 real ADIABIND

Definition at line 54 of file [param_noex.h](#).

Referenced by [ComputePressureField\(\)](#), [ComputeSoundSpeed\(\)](#), [ComputeTemperatureField\(\)](#), [CreateTorque↔MapInfile\(\)](#), [Energy\(\)](#), [ImplicitRadiativeDiffusion\(\)](#), [InitEuler\(\)](#), [InitGasVelocity\(\)](#), [Initialization\(\)](#), [InitPebblesViaFlux\(\)](#), [InitRadiatDiffusionFields\(\)](#), [InitVariables\(\)](#), [SubStep3\(\)](#), and [UpdateLog\(\)](#).

4.19.2.4 char ADVLABEL[512]

Definition at line 13 of file [param_noex.h](#).

Referenced by [InitVariables\(\)](#), and [ReadVariables\(\)](#).

4.19.2.5 real ALPHAVISCOSITY

Definition at line 26 of file [param_noex.h](#).

Referenced by [CreateTorqueMapInfile\(\)](#), [FViscosity\(\)](#), [InitVariables\(\)](#), and [ReadVariables\(\)](#).

4.19.2.6 real ASPECTRATIO

Definition at line 24 of file [param_noex.h](#).

Referenced by [AspectRatio\(\)](#), [Energy\(\)](#), [FViscosity\(\)](#), [InitVariables\(\)](#), and [TellEverything\(\)](#).

4.19.2.7 char BACKREACTION[512]

Definition at line 83 of file [param_noex.h](#).

Referenced by [InitVariables\(\)](#), and [ReadVariables\(\)](#).

4.19.2.8 real CAVITYRADIUS

Definition at line 41 of file param_noex.h.

Referenced by FViscosity(), InitVariables(), and Sigma().

4.19.2.9 real CAVITYRATIO

Definition at line 42 of file param_noex.h.

Referenced by FViscosity(), InitVariables(), and Sigma().

4.19.2.10 real CAVITYWIDTH

Definition at line 43 of file param_noex.h.

Referenced by FViscosity(), and InitVariables().

4.19.2.11 real COOLINGTIME

Definition at line 55 of file param_noex.h.

Referenced by InitCoolingTime(), InitVariables(), and ReadVariables().

4.19.2.12 real DAMPINGPERIODFRAC

Definition at line 70 of file param_noex.h.

Referenced by InitVariables(), and SetWaveKillingZones().

4.19.2.13 real DAMPINGRMAXFRAC

Definition at line 69 of file param_noex.h.

Referenced by DampingBoundary(), DampPebbles(), InitVariables(), and SetWaveKillingZones().

4.19.2.14 real DAMPINGRMINFRAC

Definition at line 68 of file param_noex.h.

Referenced by DampingBoundary(), DampPebbles(), InitVariables(), and SetWaveKillingZones().

4.19.2.15 char DAMPTOWARDS[512]

Definition at line 67 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.16 char DENSINFILE[512]

Definition at line 63 of file param_noex.h.

Referenced by Initialization(), and InitVariables().

4.19.2.17 **real** DISCALLEDO

Definition at line 60 of file param_noex.h.

Referenced by CalculateQirr(), and InitVariables().

4.19.2.18 **char** DISK[512]

Definition at line 31 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.19 **real** DT

Definition at line 7 of file param_noex.h.

Referenced by AccretePebblesOntoPlanets(), AlgoGas(), InitVariables(), ParametricAccretion(), TellEverything(), and TellNbOutputs().

4.19.2.20 **real** ECCENTRICITY

Definition at line 40 of file param_noex.h.

Referenced by InitPlanetarySystem(), and InitVariables().

4.19.2.21 **real** EFFECTIVETEMPERATURE

Definition at line 58 of file param_noex.h.

Referenced by CalculateQirr(), and InitVariables().

4.19.2.22 **char** ENERGYEQUATION[512]

Definition at line 48 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.23 **char** EXCLUDEHILL[512]

Definition at line 37 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.24 **real** FLARINGINDEX

Definition at line 39 of file param_noex.h.

Referenced by ComputeSoundSpeed(), Energy(), ImposeKeplerianEdges(), InitCoolingTime(), InitGasVelocity(), and InitVariables().

4.19.2.25 **char** FRAME[512]

Definition at line 32 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.26 int GETTORQUEFORPLANET

Definition at line 95 of file param_noex.h.

Referenced by FillForcesArrays(), InitVariables(), and ReadVariables().

4.19.2.27 char GRIDSPACING[512]

Definition at line 17 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.28 int HEATINGDELAY

Definition at line 92 of file param_noex.h.

Referenced by AccretePebblesOntoPlanets(), InitVariables(), and ParametricAccretion().

4.19.2.29 real HILLCUT

Definition at line 78 of file param_noex.h.

Referenced by FillForcesArrays(), and InitVariables().

4.19.2.30 real IAS15MINDT

Definition at line 76 of file param_noex.h.

Referenced by InitVariables(), and SetupIntegratorParams().

4.19.2.31 real IAS15PRECISION

Definition at line 75 of file param_noex.h.

Referenced by InitVariables(), and SetupIntegratorParams().

4.19.2.32 real IMPOSEDDISKDRIFT

Definition at line 38 of file param_noex.h.

Referenced by ApplyOuterSourceMass(), InitGasVelocity(), InitVariables(), and SubStep1().

4.19.2.33 char INDIRECTTERM[512]

Definition at line 36 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.34 char INITIALIZEFROMFILE[512]

Definition at line 62 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.35 real LAMBDA DOUBLING

Definition at line 47 of file param_noex.h.

Referenced by AspectRatio(), FViscosity(), and InitVariables().

4.19.2.36 real MASSTAPER

Definition at line 16 of file param_noex.h.

Referenced by AlgoGas(), and InitVariables().

4.19.2.37 int NINTERM

Definition at line 9 of file param_noex.h.

Referenced by GiveTimeInfo(), InitVariables(), main(), TellEverything(), TellNbOutputs(), and WriteDim().

4.19.2.38 int NOUTELEMENTS

Definition at line 71 of file param_noex.h.

Referenced by InitVariables(), and main().

4.19.2.39 int NRAD

Definition at line 18 of file param_noex.h.

Referenced by CheckRebin(), ChessBoardIndexing(), FillCoolingTime(), FillEnergy(), FillPolar1DArrays(), FillQplus(), FillSigma(), InitComputeAccel(), InitEuler(), InitPebbleArrays(), InitRadiatDiffusionFields(), InitTransport(), InitVariables(), InitViscosity(), main(), mpi_make1Dprofile(), SplitDomain(), and TellEverything().

4.19.2.40 int NSEC

Definition at line 19 of file param_noex.h.

Referenced by AllocateComm(), CheckRebin(), CommunicateBoundaries(), ConditionCFL(), FillPolar1DArrays(), InitComputeAccel(), InitEuler(), InitPebbleArrays(), InitRadiatDiffusionFields(), InitTransport(), InitVariables(), InitViscosity(), main(), mpi_make1Dprofile(), NonReflectingBoundary(), SynchronizeOverlapFields(), TellEverything(), and WriteDim().

4.19.2.41 int NTOT

Definition at line 10 of file param_noex.h.

Referenced by InitVariables(), main(), TellEverything(), and WriteDim().

4.19.2.42 real OMEGA FRAME

Definition at line 30 of file param_noex.h.

Referenced by InitVariables(), and main().

4.19.2.43 real OPACITY DROP

Definition at line 57 of file param_noex.h.

Referenced by CalculateQirr(), CalculateQminus(), and InitVariables().

4.19.2.44 char OPENINNERBOUNDARY[512]

Definition at line 12 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.45 char OUTERSOURCEMASS[512]

Definition at line 33 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.46 char OUTPUTDIR[512]

Definition at line 11 of file param_noex.h.

Referenced by AdvanceSystemFromDisk(), CheckRebin(), CreateTorqueMapInfile(), DumpOmegaFrame(), DumpSources(), EmptyPlanetSystemFile(), FillPolar1DArrays(), FindOrbitalElements(), fopenp(), GetfromPlanetFile(), GetOmegaFrame(), InitPebblesViaFlux(), InitVariables(), main(), merge(), OutputNbodySimulation(), ReadfromFile(), ReadPrevDim(), ReadVariables(), RestartReboundSimulation(), SetupReboundSimulation(), UpdateLog(), WriteBigPlanetFile(), WriteDim(), WriteDiskPolar(), and WritePlanetFile().

4.19.2.47 real PARAMETRICACCRETION

Definition at line 93 of file param_noex.h.

Referenced by InitVariables(), ParametricAccretion(), and ReadVariables().

4.19.2.48 real PARAMETRICOPACITY

Definition at line 61 of file param_noex.h.

Referenced by CreateTorqueMapInfile(), InitVariables(), and OpacityProfile().

4.19.2.49 char PARTICLEDIFFUSION[512]

Definition at line 91 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.50 char PEBBLEACCRETION[512]

Definition at line 82 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.51 real PEBBLEALPHA

Definition at line 87 of file param_noex.h.

Referenced by AccretePebblesOntoPlanets(), FillForcesArrays(), and InitVariables().

4.19.2.52 **real** PEBBLEBULKDENS

Definition at line 89 of file param_noex.h.

Referenced by InitPebbleArrays(), and InitVariables().

4.19.2.53 **real** PEBBLECOAGULATION

Definition at line 88 of file param_noex.h.

Referenced by InitPebblesViaFlux(), and InitVariables().

4.19.2.54 **real** PEBBLEFLUX

Definition at line 86 of file param_noex.h.

Referenced by InitPebblesViaFlux(), and InitVariables().

4.19.2.55 **real** PLANETARYDENSITY

Definition at line 72 of file param_noex.h.

Referenced by AccretePebblesOntoPlanets(), InitVariables(), ParametricAccretion(), and SetupReboundSimulation().

4.19.2.56 **char** PLANETCONFIG[512]

Definition at line 15 of file param_noex.h.

Referenced by InitVariables(), and main().

4.19.2.57 **char** PLANETSFEELDISK[512]

Definition at line 80 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.58 **real** RELEASEDATE

Definition at line 29 of file param_noex.h.

Referenced by InitVariables().

4.19.2.59 **real** RELEASERADIUS

Definition at line 28 of file param_noex.h.

Referenced by InitVariables().

4.19.2.60 **char** RESOLVECOLLISIONS[512]

Definition at line 73 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.61 real RMAX

Definition at line 21 of file param_noex.h.

Referenced by DampingBoundary(), DampPebbles(), DiscardParticlesDist(), FillPolar1DArrays(), InitLabel(), InitVariables(), SetWaveKillingZones(), TellEverything(), and WriteDim().

4.19.2.62 real RMIN

Definition at line 20 of file param_noex.h.

Referenced by DampingBoundary(), DampPebbles(), FillPolar1DArrays(), InitLabel(), InitVariables(), SetWaveKillingZones(), and TellEverything().

4.19.2.63 real ROCHESMOOTHING

Definition at line 23 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.64 real SCHMIDTNUMBER

Definition at line 90 of file param_noex.h.

Referenced by InitVariables(), and ParticleDiffusion().

4.19.2.65 real SIGMA0

Definition at line 8 of file param_noex.h.

Referenced by Energy(), InitGasVelocity(), InitQplus(), InitVariables(), Sigma(), and TellEverything().

4.19.2.66 real SIGMASLOPE

Definition at line 27 of file param_noex.h.

Referenced by ApplyOuterSourceMass(), Energy(), ImposeKeplerianEdges(), InitGasVelocity(), InitQplus(), InitVariables(), Sigma(), and SubStep1().

4.19.2.67 char STELLARIRRADIATION[512]

Definition at line 56 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.68 real STELLARRADIUS

Definition at line 59 of file param_noex.h.

Referenced by CalculateFlaring(), CalculateQirr(), and InitVariables().

4.19.2.69 int TARGETNPL

Definition at line 74 of file param_noex.h.

Referenced by InitVariables(), main(), and ReadVariables().

4.19.2.70 char TEMPERINFILE[512]

Definition at line 66 of file param_noex.h.

Referenced by Initialization(), and InitVariables().

4.19.2.71 real THICKNESSSMOOTHING

Definition at line 22 of file param_noex.h.

Referenced by FillForcesArrays(), InitVariables(), ReadVariables(), and ThicknessSmoothing().

4.19.2.72 char TORQUEMAPINFILE[512]

Definition at line 94 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.73 real TRANSITIONRADIUS

Definition at line 44 of file param_noex.h.

Referenced by AspectRatio(), and InitVariables().

4.19.2.74 real TRANSITIONRATIO

Definition at line 45 of file param_noex.h.

Referenced by AspectRatio(), and InitVariables().

4.19.2.75 real TRANSITIONWIDTH

Definition at line 46 of file param_noex.h.

Referenced by AspectRatio(), and InitVariables().

4.19.2.76 char TRANSPORT[512]

Definition at line 14 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.77 real VERTICALDAMPING

Definition at line 79 of file param_noex.h.

Referenced by DampingTW04(), and InitVariables().

4.19.2.78 real VISCOSITY

Definition at line 25 of file param_noex.h.

Referenced by CreateTorqueMapInfile(), FViscosity(), InitVariables(), and ReadVariables().

4.19.2.79 char VRADINFILE[512]

Definition at line 64 of file param_noex.h.

Referenced by Initialization(), and InitVariables().

4.19.2.80 char VTHETAFILE[512]

Definition at line 65 of file param_noex.h.

Referenced by Initialization(), and InitVariables().

4.19.2.81 char WRITEDENSITY[512]

Definition at line 34 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.82 char WRITEDIVV[512]

Definition at line 51 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.83 char WRITEENERGY[512]

Definition at line 50 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.84 char WRITEETA[512]

Definition at line 85 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.85 char WRITEQBALANCE[512]

Definition at line 53 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.86 char WRITEQPLUS[512]

Definition at line 52 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.87 char WRITETEMPERATURE[512]

Definition at line 49 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.88 char WRITETORQUEFILES[512]

Definition at line 77 of file param_noex.h.

Referenced by InitVariables(), and ReadVariables().

4.19.2.89 char WRITEVELOCITY[512]

Definition at line 35 of file param_noex.h.

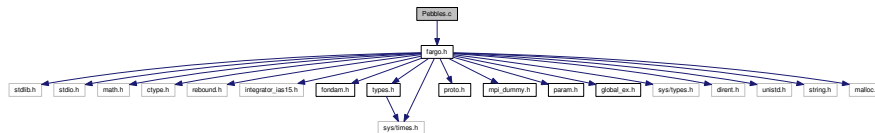
Referenced by InitVariables(), and ReadVariables().

4.20 Pebbles.c File Reference

Contains functions responsible for the pebble disk initialisation, evolution due to source terms and pebble accretion.

```
#include "fargo.h"
```

Include dependency graph for Pebbles.c:



Macros

- #define [TRAPEZMAX](#) 35
- #define [TRAPEZEPS](#) 1.0e-7

Functions

- void [InitPebbleArrays](#) ()
Initialise polar arrays associated with the pebble disk.
- void [EquilPebbleDisk](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta)
Sets up a pebble disk in a coagulation-drift equilibrium.
- void [RestartPebbleDisk](#) ([PolarGrid](#) *Rho, int index)
Reads arrays necessary to restart the pebble disk.
- void [InitPebblesViaFlux](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Vrad)
Imposing the radial mass flux of pebbles in a coagulation-drift equilibrium, initialises their surface density, flow velocity and local Stokes numbers.
- void [BckpFieldsForBC](#) ()
Backs up the initial state of the pebble disk to impose damping boundary conditions later.
- void [EtaPressureSupport](#) ([PolarGrid](#) *Vtheta)
Calculates the gas rotation parameter eta.
- void [PebbleStokesNumbers](#) ([PolarGrid](#) *Rho)
Calculates the local Stokes number using the dominant pebble size, local gas density and parametric pebble material density.
- real [Trapzd](#) (int n, real a, real b, real H2)
Trapezoidal rule to integrate the column mass of pebbles located within the accretion radius.
- real [IntegrateColumnMass](#) (real a, real b, real Hpeb)

- Primitive algorithm to find the mass in a vertical column of pebbles overlapping the accretion radius.*
- void [AccretePebblesOntoPlanets](#) ([PlanetarySystem](#) *sys, [PolarGrid](#) *Rho, [PolarGrid](#) *Energy, [PolarGrid](#) *Vtheta, [real](#) dt)
- Finds the amount of pebbles to be transfered from the pebble disk onto the planets.*
- void [CorrectPebblesVtheta](#) ([real](#) domega)
- Corrects the azimuthal flow velocity of pebbles to keep up with the frame rotation.*
- void [SourceTermsPebbles](#) ([PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta, [real](#) dt)
- Calculates the source terms acting on pebbles.*
- void [SubStep1Pebbles](#) ([PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta, [real](#) dt)
- Applies a semi-implicit method to evolve the pebbles dynamically.*
- void [ParticleDiffusion](#) ([PolarGrid](#) *Rho)
- Applies the particle diffusion term acting on pebbles.*
- void [EvolvePebbleDisk](#) ([real](#) dt)
- Calls the transport routines and applies the boundary conditions for pebbles.*
- void [SynchronizePebbleDisc](#) ()
- Synchronises pebble fluid hydrodynamic quantities among the overlapping grid zones.*
- void [CriticalCharTime](#) ([PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta)
- Restricts the time step using the CFL condition for the pebble fluid.*
- void [WritePebbles](#) ([int](#) index)
- Outputs the pebble fluid arrays.*
- [boolean](#) [DetectCrashPebbles](#) ()
- Safety check for negative pebble densities.*
- void [ParametricAccretion](#) ([PlanetarySystem](#) *sys, [real](#) dt)
- Writes filtering factors if needed.*

Variables

- [boolean](#) Restart
- [boolean](#) FastTransport
- static [PolarGrid](#) * [PebbleDensTemp](#)
- static [PolarGrid](#) * [PebbleAccelrad](#)
- static [PolarGrid](#) * [PebbleAcceltheta](#)
- static [PolarGrid](#) * [PebbleSize](#)
- static [PolarGrid](#) * [EtaFaceCentered](#)
- static [PolarGrid](#) * [EtaCellCentered](#)
- static [PolarGrid](#) * [AccretedMass](#)
- static [real](#) pebbulkdens

4.20.1 Detailed Description

Contains functions responsible for the pebble disk initialisation, evolution due to source terms and pebble accretion. Also controls several outputs.

Author

Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz

The pebble disk equilibrium model is inspired by Lambrechts & Johansen (2014). The evolution follows a standard two-fluid approximation with linear drag coupling and particle diffusion.

4.20.2 LICENSE

Copyright (c) 2017 Ondřej Chrenko. See the LICENSE file of the distribution.

Definition in file [Pebbles.c](#).

4.20.3 Macro Definition Documentation

4.20.3.1 #define TRAPEZEPS 1.0e-7

Definition at line 22 of file Pebbles.c.

Referenced by `IntegrateColumnMass()`.

4.20.3.2 #define TRAPEZMAX 35

Definition at line 21 of file Pebbles.c.

Referenced by `IntegrateColumnMass()`.

4.20.4 Function Documentation

4.20.4.1 void AccretePebblesOntoPlanets (PlanetarySystem * sys, PolarGrid * Rho, PolarGrid * Energy, PolarGrid * Vtheta, real dt)

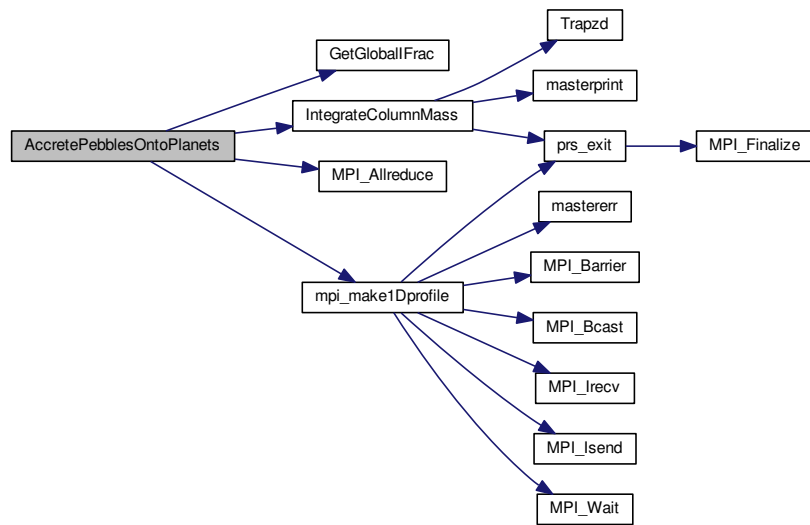
Finds the amount of pebbles to be transfered from the pebble disk onto the planets.

Definition at line 312 of file Pebbles.c.

References `AccretHeating`, `CellAbscissa`, `CellOrdinate`, `DT`, `polargrid::Field`, `GetGlobalFrac()`, `HEATINGDELA`, `Y`, `heatsrc`, `heatsrc_index`, `heatsrc_max`, `IntegrateColumnMass()`, `InvRmed`, `Max_or_active`, `MPI_Allreduce()`, `M`, `PI_COMM_WORLD`, `MPI_DOUBLE`, `MPI_INT`, `mpi_make1Dprofile()`, `MPI_SUM`, `polargrid::Nrad`, `polargrid::Nsec`, `OmegaFrame`, `OmegaInv`, `PEBBLEALPHA`, `PebbleDens`, `PebbleVrad`, `PebbleVtheta`, `PhysicalTime`, `PI`, `PLANETA`, `RYDENSITY`, `RHO2CGS`, `Rinf`, `Rmed`, `Rsup`, `SoundSpeed`, `SQRT2PI_INV`, `SQRT_ADIABIND_INV`, `StokesNumber`, `Surf`, `vt1D`, `VXplanet`, `VYplanet`, `Xplanet`, `Yplanet`, and `Zero_or_active`.

Referenced by `AlgoGas()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.2 void BckpFieldsForBC ()

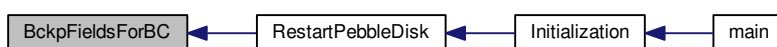
Backs up the initial state of the pebble disk to impose damping boundary conditions later.

Definition at line 168 of file Pebbles.c.

References polargrid::Field, polargrid::Nrad, polargrid::Nsec, OmegaFrame, PebbleDens, PebbleVrad, PebbleVtheta, PebDensInit, PebVradInit, PebVthetaInit, and Rmed.

Referenced by RestartPebbleDisk().

Here is the caller graph for this function:



4.20.4.3 void CorrectPebblesVtheta (real *omega*)

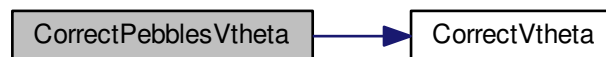
Corrects the azimuthal flow velocity of pebbles to keep up with the frame rotation.

Definition at line 516 of file Pebbles.c.

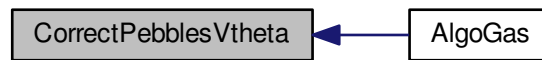
References CorrectVtheta(), and PebbleVtheta.

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.4 void CriticalCharTime (PolarGrid * *Vrad*, PolarGrid * *Vtheta*)

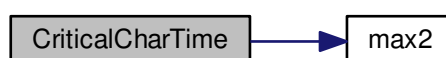
Restricts the time step using the CFL condition for the pebble fluid.

Definition at line 702 of file Pebbles.c.

References FastTransport, polargrid::Field, invdtpsb_sq, max2(), Max_or_active, polargrid::Nsec, One_or_active, PebbleVrad, PebbleVtheta, PI, Rinf, Rmed, and Rsup.

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.5 **boolean** DetectCrashPebbles ()

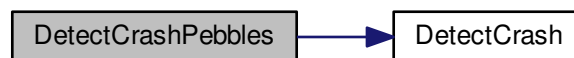
Safety check for negative pebble densities.

Definition at line 753 of file Pebbles.c.

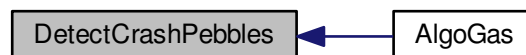
References DetectCrash(), and PebbleDens.

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.6 **void** EquilPebbleDisk (**PolarGrid *** *Rho*, **PolarGrid *** *Vrad*, **PolarGrid *** *Vtheta*)

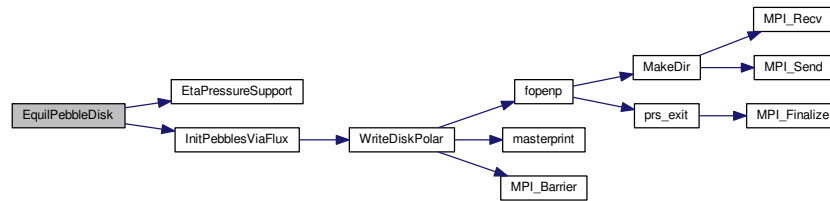
Sets up a pebble disk in a coagulation-drift equilibrium.

Definition at line 58 of file Pebbles.c.

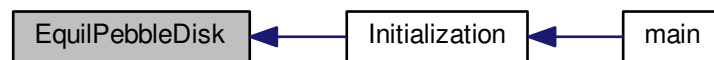
References EtaPressureSupport(), and InitPebblesViaFlux().

Referenced by Initialization().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.7 void EtaPressureSupport (PolarGrid * Vtheta)

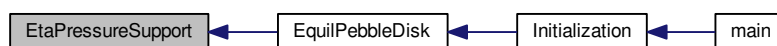
Calculates the gas rotation parameter eta.

Definition at line 195 of file Pebbles.c.

References polargrid::Field, polargrid::Nrad, OmegaFrame, and Rmed.

Referenced by EquilPebbleDisk().

Here is the caller graph for this function:



4.20.4.8 void EvolvePebbleDisk (real dt)

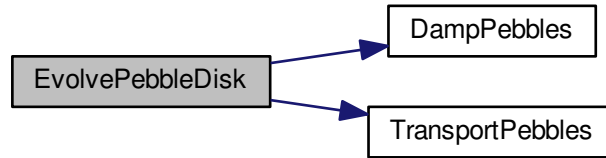
Calls the transport routines and applies the boundary conditions for pebbles.

Definition at line 675 of file Pebbles.c.

References DampPebbles(), PebbleDens, PebbleVrad, PebbleVtheta, and TransportPebbles().

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.9 void InitPebbleArrays ()

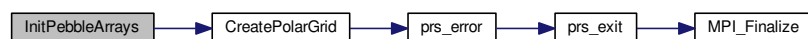
Initialise polar arrays associated with the pebble disk.

Definition at line 37 of file Pebbles.c.

References `CreatePolarGrid()`, `DragForceRad`, `DragForceTheta`, `GasAccelrad`, `GasAcceltheta`, `NRAD`, `NSEC`, `P↔`, `EBBLEBULKDENS`, `PebbleDens`, `PebbleVrad`, `PebbleVtheta`, `pebbulkdens`, `RHO2CGS`, and `StokesNumber`.

Referenced by `InitEuler()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.10 void InitPebblesViaFlux (PolarGrid * Rho, PolarGrid * Vrad)

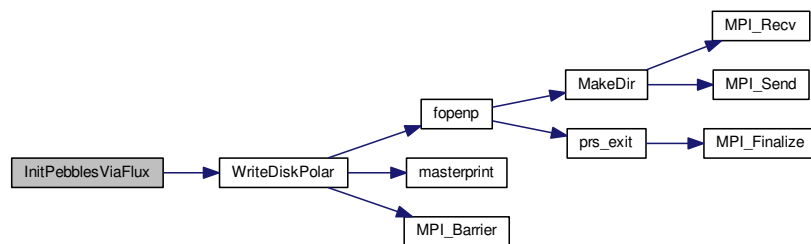
Imposing the radial mass flux of pebbles in a coagulation-drift equilibrium, initialises their surface density, flow velocity and local Stokes numbers.

Definition at line 80 of file Pebbles.c.

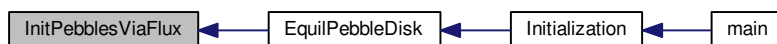
References ADIABIND, CPU_Master, CPU_Number, polargrid::Field, FLUX2CU, Merge, OmegaFrame, OUTPUT↔DIR, PEBBLECOAGULATION, PebbleDens, PEBBLEFLUX, PebbleVrad, PebbleVtheta, pebbulkdens, PebDensInit, PebVradInit, PebVthetaInit, PI, Rmed, StokesNumber, and WriteDiskPolar().

Referenced by EquilPebbleDisk().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.11 real IntegrateColumnMass (real a, real b, real Hpeb)

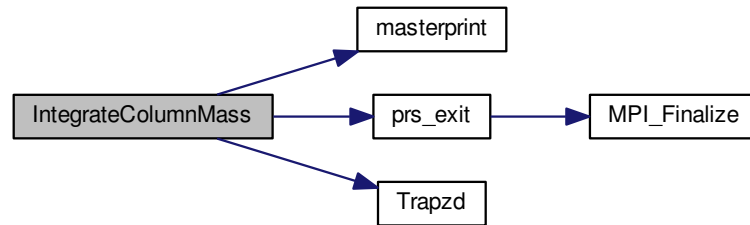
Primitive algorithm to find the mass in a vertical column of pebbles overlapping the accretion radius.

Definition at line 292 of file Pebbles.c.

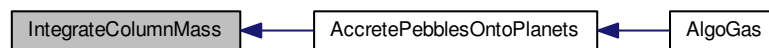
References masterprint(), prs_exit(), TRAPEZEPS, TRAPEZMAX, and Trapzd().

Referenced by `AccretePebblesOntoPlanets()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.12 void ParametricAccretion (PlanetarySystem * sys, real dt)

Writes filtering factors if needed.

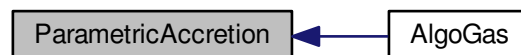
Mass accretion onto planets is provided by a parametric prescription using a given mass doubling time.

Definition at line 779 of file `Pebbles.c`.

References `AccretHeating`, `DT`, `HEATINGDELAY`, `heatsrc`, `heatsrc_index`, `heatsrc_max`, `polargrid::Nrad`, `polargrid::Nsec`, `PARAMETRICACCRETION`, `PhysicalTime`, `PI`, `PLANETARYDENSITY`, `RHO2CGS`, `Rinf`, `Rsup`, `SoundSpeed`, `Surf`, `Xplanet`, and `Yplanet`.

Referenced by `AlgoGas()`.

Here is the caller graph for this function:



4.20.4.13 void ParticleDiffusion (PolarGrid * Rho)

Applies the particle diffusion term acting on pebbles.

See Eq. (35) in Chrenko et al. (2017)

Definition at line 629 of file Pebbles.c.

References polargrid::Field, FViscosity(), InvDiffRmed, PebbleDens, PebbleVrad, PebbleVtheta, PI, Rinf, Rmed, and SCHMIDTNUMBER.

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.14 void PebbleStokesNumbers (PolarGrid * Rho)

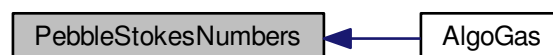
Calculates the local Stokes number using the dominant pebble size, local gas density and parametric pebble material density.

Definition at line 240 of file Pebbles.c.

References polargrid::Field, polargrid::Nrad, pebbulkdens, SQRT_ADIABIND_INV, and StokesNumber.

Referenced by AlgoGas().

Here is the caller graph for this function:



4.20.4.15 void RestartPebbleDisk (PolarGrid * *Rho*, int *index*)

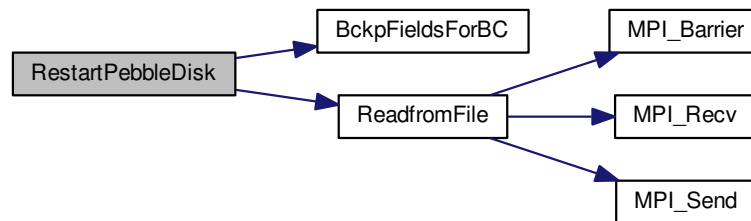
Reads arrays necessary to restart the pebble disk.

Definition at line 66 of file Pebbles.c.

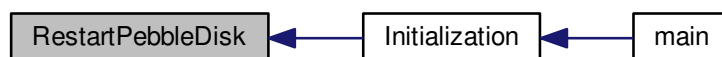
References BckpFieldsForBC(), PebbleDens, PebbleVrad, PebbleVtheta, and ReadfromFile().

Referenced by Initialization().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.16 void SourceTermsPebbles (PolarGrid * *Vrad*, PolarGrid * *Vtheta*, real *dt*)

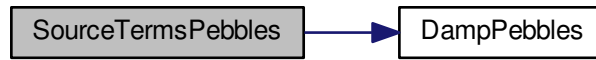
Calculates the source terms acting on pebbles.

Definition at line 523 of file Pebbles.c.

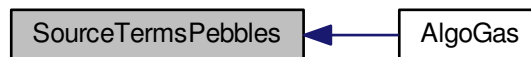
References BackReaction, DampPebbles(), DragForceRad, DragForceTheta, polargrid::Field, InvRinf, Omega↔Frame, OmegaInv, PebbleDens, PebbleGravAccelRad, PebbleGravAccelTheta, PebbleVrad, PebbleVtheta, Rinf, and StokesNumber.

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.17 void SubStep1Pebbles (PolarGrid * Vrad, PolarGrid * Vtheta, real dt)

Applies a semi-implicit method to evolve the pebbles dynamically.

See Appendix C in Chrenko et al. (2017)

Definition at line 590 of file Pebbles.c.

References polargrid::Field, GasAccelrad, GasAcceltheta, OmegaInv, PebbleVrad, PebbleVtheta, and Stokes←→Number.

Referenced by AlgoGas().

Here is the caller graph for this function:



4.20.4.18 void SynchronizePebbleDisc ()

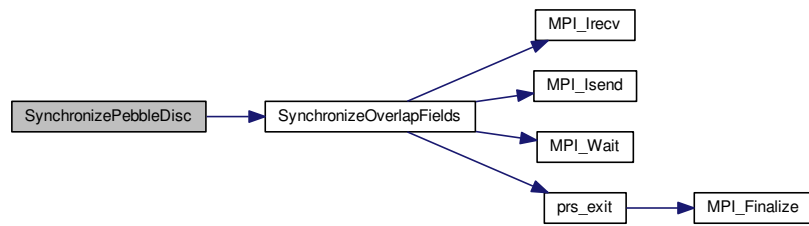
Synchronises pebble fluid hydrodynamic quantities among the overlapping grid zones.

Definition at line 685 of file Pebbles.c.

References CPU_Number, CPUOVERLAP, polargrid::Field, polargrid::Nrad, PebbleDens, PebbleVrad, Pebble←→Vtheta, and SynchronizeOverlapFields().

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.4.19 `real Trapzd (int n, real a, real b, real H2)`

Trapezoidal rule to integrate the column mass of pebbles located within the accretion radius.

Adopted from Numerical Recipes.

Definition at line 265 of file `Pebbles.c`.

References `a`.

Referenced by `IntegrateColumnMass()`.

Here is the caller graph for this function:



4.20.4.20 `void WritePebbles (int index)`

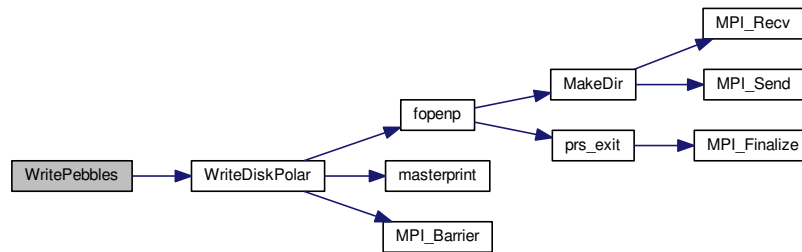
Outputs the pebble fluid arrays.

Definition at line 743 of file `Pebbles.c`.

References `PebbleDens`, `PebbleVrad`, `PebbleVtheta`, `Write_Eta`, and `WriteDiskPolar()`.

Referenced by `SendOutput()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.20.5 Variable Documentation

4.20.5.1 `PolarGrid* AccretedMass` [static]

Definition at line 29 of file `Pebbles.c`.

4.20.5.2 `PolarGrid * EtaCellCentered` [static]

Definition at line 28 of file `Pebbles.c`.

4.20.5.3 `PolarGrid* EtaFaceCentered` [static]

Definition at line 28 of file `Pebbles.c`.

4.20.5.4 `boolean FastTransport`

Definition at line 29 of file `Interpret.c`.

Referenced by `CriticalCharTime()`.

4.20.5.5 `PolarGrid * PebbleAccelrad` [static]

Definition at line 26 of file `Pebbles.c`.

4.20.5.6 **PolarGrid * PebbleAcceltheta** [static]

Definition at line 26 of file Pebbles.c.

4.20.5.7 **PolarGrid * PebbleDensTemp** [static]

Definition at line 26 of file Pebbles.c.

4.20.5.8 **PolarGrid * PebbleSize** [static]

Definition at line 27 of file Pebbles.c.

4.20.5.9 **real pebbulkdens** [static]

Definition at line 31 of file Pebbles.c.

Referenced by InitPebbleArrays(), InitPebblesViaFlux(), and PebbleStokesNumbers().

4.20.5.10 **boolean Restart**

Definition at line 14 of file main.c.

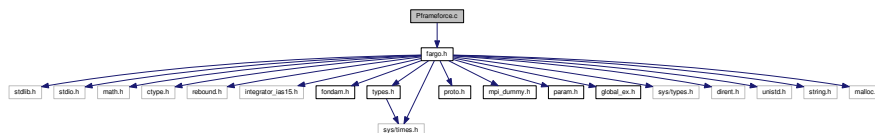
Referenced by main().

4.21 Pframeforce.c File Reference

Calculates the gravitational interactions between the disk and massive bodies in terms of a vertically averaged potential.

```
#include "fargo.h"
```

Include dependency graph for Pframeforce.c:



Functions

- void **FillForcesArrays** (**PolarGrid** *Rho, **PlanetarySystem** *sys)
Using the vertical averaging procedure of Muller & Kley (2012), calculates the acceleration in planet-disk and star-disk interactions for both gas and pebbles.
- void **AdvanceSystemFromDisk** (**PolarGrid** *Rho, **PlanetarySystem** *sys, **real** dt)
Updates the planet velocities due to disk forces.
- **real** **DampingTW04** (**PolarGrid** *Rho, **real** m, **real** x, **real** y, **real** z, **real** vz)
Artificial vertical force to damp the orbital inclinations using the Tanaka & Ward (2004) prescription (see also Morbidelli et al.
- **real** **ConstructSequence** (**real** *u, **real** *v, **int** n)
- void **InitGasDensityEnergy** (**PolarGrid** *Rho, **PolarGrid** *Energy)
Part of the initialisation.
- void **InitGasVelocity** (**PolarGrid** *Vr, **PolarGrid** *Vt)

Variables

- [boolean AllowAccretion](#)
- [boolean Corotating](#)
- [boolean Indirect_Term](#)
- static [Pair IndirectTerm](#)
- static [real vt_int](#) [MAX1D]
- static [real vt_cent](#) [MAX1D]
- [boolean DumpTorqueNow](#)
- [boolean DumpTorqueDensNow](#)

4.21.1 Detailed Description

Calculates the gravitational interactions between the disk and massive bodies in terms of a vertically averaged potential.

Author

Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz

The function [FillForcesArrays\(\)](#) computes the gravitational acceleration due to planet-disk and star-disk interactions adopting the outline of Muller & Kley (2012). Works both for the gas and pebbles. The indirect terms are included as well as a deep cubic potential with thickness smoothing (see Eq. (37) in Chrenko et al. 2017). There is also a function which provides the inclination damping for 3D planetary orbits.

4.21.2 LICENSE

Copyright (c) 2017 Ondřej Chrenko. See the LICENSE file of the distribution.

Definition in file [Pframeforce.c](#).

4.21.3 Function Documentation

4.21.3.1 void AdvanceSystemFromDisk ([PolarGrid](#) * *Rho*, [PlanetarySystem](#) * *sys*, [real](#) *dt*)

Updates the planet velocities due to disk forces.

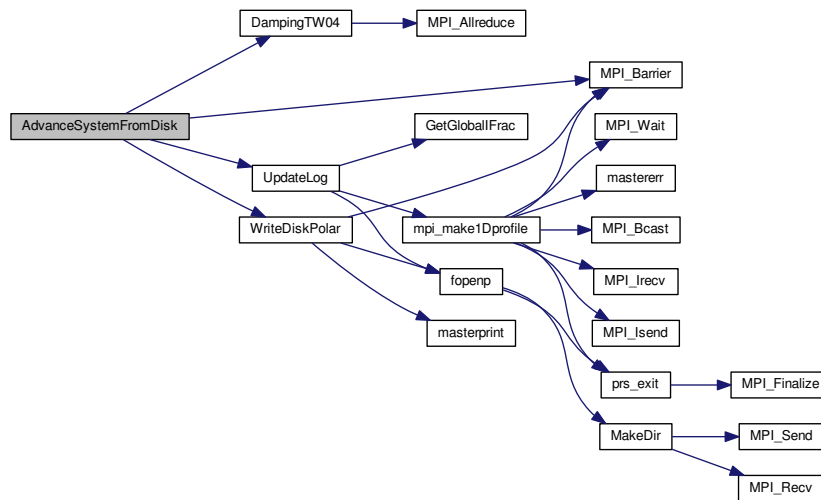
Also applies the inclination damping.

Definition at line 448 of file [Pframeforce.c](#).

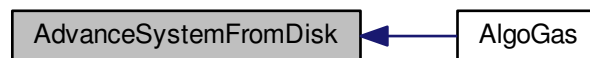
References [CPU_Master](#), [CPU_Number](#), [DampingTW04\(\)](#), [DumpTorqueDensNow](#), [DumpTorqueNow](#), [Merge](#), [M_PI_Barrier\(\)](#), [MPI_COMM_WORLD](#), [NO](#), [OUTPUTDIR](#), [TimeStep](#), [Torque](#), [UpdateLog\(\)](#), [WriteDiskPolar\(\)](#), [pair::x](#), [pair::y](#), and [YES](#).

Referenced by [AlgoGas\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:

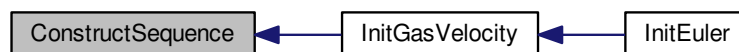


4.21.3.2 `real ConstructSequence (real * u, real * v, int n)`

Definition at line 529 of file Pframeforce.c.

Referenced by `InitGasVelocity()`.

Here is the caller graph for this function:



4.21.3.3 `real DampingTW04 (PolarGrid * Rho, real m, real x, real y, real z, real vz)`

Artificial vertical force to damp the orbital inclinations using the Tanaka & Ward (2004) prescription (see also Morbidelli et al.

2007, Pierens & Nelson 2008)

Definition at line 495 of file Pframeforce.c.

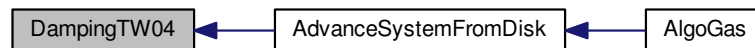
References CPU_Number, polargrid::Field, Max_or_active, MPI_Allreduce(), MPI_COMM_WORLD, MPI_DOUBLE, MPI_SUM, PI, Rinf, Rsup, SoundSpeed, VERTICALDAMPING, and Zero_or_active.

Referenced by AdvanceSystemFromDisk().

Here is the call graph for this function:



Here is the caller graph for this function:



4.21.3.4 void FillForcesArrays (PolarGrid * Rho, PlanetarySystem * sys)

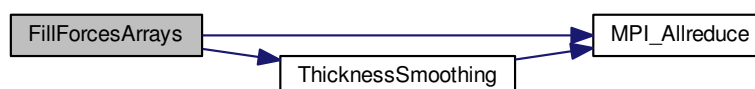
Using the vertical averaging procedure of Muller & Kley (2012), calculates the acceleration in planet-disk and star-disk interactions for both gas and pebbles.

Definition at line 38 of file Pframeforce.c.

References planetary_system::ax, planetary_system::ay, CellAbscissa, CellOrdinate, ExcludeHill, polargrid::Field, GETTORQUEFORPLANET, GravAccelRad, GravAccelTheta, HILLCUT, Indirect_Term, InvRmed, planetary_system::mass, MassTaper, Max_or_active, MAXPLANETS, MPI_Allreduce(), MPI_COMM_WORLD, MPI_DOUBLE, MPI_SUM, planetary_system::nb, NO, Omegainv, PEBBLEALPHA, PebbleGravAccelRad, PebbleGravAccelTheta, Pebbles, Rmed, Rmed2, SoundSpeed, SQRT_ADIABIND_INV, StokesNumber, Surf, THICKNESSSMOOTHING, ThicknessSmoothing(), Torque, TorqueDensity, pair::x, planetary_system::x, pair::y, planetary_system::y, YES, planetary_system::z, and Zero_or_active.

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.21.3.5 void InitGasDensityEnergy (PolarGrid * *Rho*, PolarGrid * *Energy*)

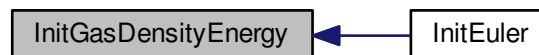
Part of the initialisation.

Definition at line 544 of file Pframeforce.c.

References `EnergyMed`, and `SigmaMed`.

Referenced by `InitEuler()`.

Here is the caller graph for this function:



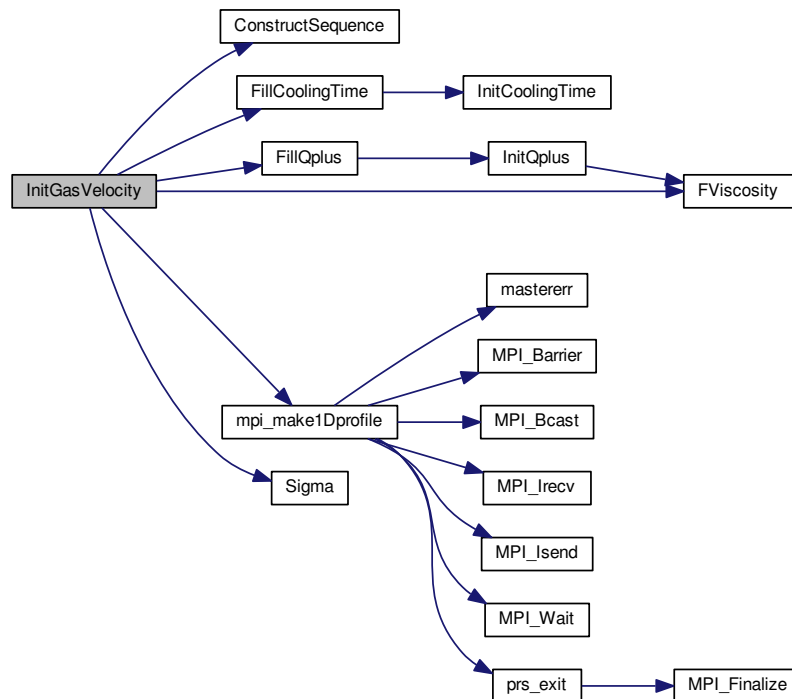
4.21.3.6 void InitGasVelocity (PolarGrid * *Vr*, PolarGrid * *Vt*)

Definition at line 568 of file Pframeforce.c.

References `ADIABIND`, `CentrifugalBalance`, `ConstructSequence()`, `polargrid::Field`, `FillCoolingTime()`, `FillQplus()`, `FLARINGINDEX`, `FViscosity()`, `G`, `GLOBALNRAD`, `GlobalRmed`, `globpressvec`, `IMIN`, `IMPOSEDDISKDRIFT`, `mpi_↔_make1Dprofile()`, `OmegaFrame`, `ParametricCooling`, `Pressure`, `Radii`, `Rinf`, `Rmed`, `Sigma()`, `SIGMA0`, `SigmaInf`, `SIGMASLOPE`, `SoundSpeed`, `ViscosityAlpha`, `vt_cent`, and `vt_int`.

Referenced by `InitEuler()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.21.4 Variable Documentation

4.21.4.1 `boolean` AllowAccretion

4.21.4.2 `boolean` Corotating

Definition at line 30 of file `Interpret.c`.

4.21.4.3 `boolean` DumpTorqueDensNow

Definition at line 21 of file `main.c`.

Referenced by `AdvanceSystemFromDisk()`, and `main()`.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [Planet.c](#).

4.22.2 Function Documentation

4.22.2.1 `void AccreteOntoPlanets (PolarGrid * Rho, PolarGrid * Vrad, PolarGrid * Vtheta, real dt, PlanetarySystem * sys)`

Definition at line 15 of file Planet.c.

References `planetary_system::acc`, `CellAbscissa`, `CellOrdinate`, `planetary_system::FeelDisk`, `polargrid::Field`, `planetary_system::mass`, `Max_or_active`, `MPI_Allreduce()`, `MPI_COMM_WORLD`, `MPI_DOUBLE`, `MPI_SUM`, `planetary_system::nb`, `polargrid::Nrad`, `polargrid::Nsec`, `OmegaFrame`, `PI`, `Rinf`, `Rmed`, `Rsup`, `Surf`, `planetary_system::vx`, `VXplanet`, `planetary_system::vy`, `VYplanet`, `planetary_system::x`, `Xplanet`, `planetary_system::y`, `YES`, `Yplanet`, and `Zero_or_active`.

Referenced by `AlgoGas()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.22.2.2 `void FindOrbitalElements (real x, real y, real vx, real vy, real m, int n)`

Definition at line 119 of file Planet.c.

References `a`, `CPU_Number`, `CPU_Rank`, `fopenp()`, `G`, `OUTPUTDIR`, and `PhysicalTime`.

- void [WriteBigPlanetSystemFile](#) ()
- [real](#) [GetfromPlanetFile](#) ()
- void [RestartPlanetarySystem](#) ()
- void [WriteDiskPolar](#) ()
- void [WriteDim](#) ()
- void [SendOutput](#) ()
- void [FillForcesArrays](#) ()
- void [AdvanceSystemFromDisk](#) ()
- [real](#) [ConstructSequence](#) ()
- void [InitGas](#) ()
- void [AccreteOntoPlanets](#) ()
- void [FindOrbitalElements](#) ()
- int [FindNumberOfPlanets](#) ()
- [PlanetarySystem](#) * [AllocPlanetSystem](#) ()
- void [FreePlanetary](#) ()
- [PlanetarySystem](#) * [InitPlanetarySystem](#) ()
- void [ListPlanets](#) ()
- [real](#) [GetPsysInfo](#) ()
- void [RotatePsys](#) ()
- [real](#) [GasTotalMass](#) ()
- [real](#) [GasMomentum](#) ()
- void [DivisePolarGrid](#) ()
- void [InitComputeAccel](#) ()
- void [OpenBoundary](#) ()
- void [NonReflectingBoundary](#) ()
- void [ApplyOuterSourceMass](#) ()
- void [ApplyBoundaryCondition](#) ()
- void [CorrectVtheta](#) ()
- [boolean](#) [DetectCrash](#) ()
- void [FillPolar1DArrays](#) ()
- void [InitEuler](#) ()
- [real](#) [min2](#) ()
- [real](#) [max2](#) ()
- void [ActualiseGas](#) ()
- void [AlgoGas](#) ()
- void [SubStep1](#) ()
- void [SubStep2](#) ()
- int [ConditionCFL](#) ()
- [real](#) [Sigma](#) ()
- void [FillSigma](#) ()
- void [RefillSigma](#) ()
- void [Transport](#) ()
- void [OneWindRad](#) ()
- void [ComputeThetaElongations](#) ()
- void [ComputeAverageThetaVelocities](#) ()
- void [ComputeResiduals](#) ()
- void [ComputeConstantResidual](#) ()
- void [AdvectSHIFT](#) ()
- void [OneWindTheta](#) ()
- void [QuantitiesAdvection](#) ()
- void [ComputeExtQty](#) ()
- void [ComputeSpeQty](#) ()
- void [InitTransport](#) ()
- void [ComputeStarRad](#) ()
- void [ComputeStarTheta](#) ()

- void [ComputeLRMomenta](#) ()
- void [ComputeVelocities](#) ()
- [real](#) [VanLeerRadial](#) ()
- void [VanLeerTheta](#) ()
- void [InitViscosity](#) ()
- void [ViscousTerms](#) ()
- void [AllocateComm](#) ()
- void [CommunicateBoundaries](#) ()
- void [handfpe](#) ()
- void [setfpe](#) ()
- void [merge](#) ()
- void [ReadPrevDim](#) ()
- void [CheckRebin](#) ()
- void [SplitDomain](#) ()
- void [InitVariables](#) ()
- [real](#) [FViscosity](#) ()
- [real](#) [AspectRatio](#) ()
- void [MakeDir](#) ()
- FILE * [fopenp](#) ()
- void [SubStep3](#) ()
- void [ComputeSoundSpeed](#) ()
- void [ComputeTemperatureField](#) ()
- void [ComputePressureField](#) ()
- [real](#) [ThicknessSmoothing](#) ()
- void [mpi_make1Dprofile](#) ()
- void [InitGasDensityEnergy](#) ()
- [real](#) [GasTotalEnergy](#) ()
- [real](#) [Energy](#) ()
- void [FillEnergy](#) ()
- void [RefillEnergy](#) ()
- void [FillVtheta](#) ()
- void [InitGasVelocity](#) ()
- [real](#) [InitCoolingTime](#) ()
- void [FillCoolingTime](#) ()
- [real](#) [InitQplus](#) ()
- void [FillQplus](#) ()
- void [UpdateDivVelocAndStressTensor](#) ()
- void [UpdateVelocityWithViscousTerms](#) ()
- void [ImposeKeplerianEdges](#) ()
- void [ReadfromAsciiFile](#) ()
- void [InitRadiatDiffusionFields](#) ()

Initialises the polar arrays associated with the heating/cooling processes.

- void [CalculateQminus](#) ()
- void [CalculateFlaring](#) ()

Calculates the sine of the grazing angle by reconstructing the surface from the pressure scale height.

- void [CalculateQirr](#) ()
- void [ImplicitRadiativeDiffusion](#) ()
- void [TemperatureGradient](#) ()

Finds the temperature gradients and their magnitude over the mesh.

- void [MidplaneVolumeDensity](#) ()
- void [OpacityProfile](#) ()

Fills the opacity polar grid, either with a fixed parametric value or using the Bell & Lin (1994) opacity table.

- [real](#) [FluxLimiterValue](#) ()
- [real](#) [EffectiveOpticalDepth](#) ()

- void [IterateRelaxationParameter](#) ()
When solving the energy equation for the first time, the function spans through various values of the SOR parameter in order to find its best value to start with.
- int [SuccessiveOverrelaxation](#) ()
- void [DiffusionCoefs](#) ()
Calculation of the diffusion coefficients.
- void [SynchronizeOverlapFields](#) ()
- void [ChessBoardIndexing](#) ()
Function ensures the odd-even ordering of the SOR method when the grid is split on multiple CPUs.
- void [SetWaveKillingZones](#) ()
Sets the wave-killing factors within the damping zones; inspired by de Val-Borro et al.
- void [DampingBoundary](#) ()
- void [ActualizeQbalance](#) ()
- struct reb_simulation * [SetupReboundSimulation](#) ()
- void [SetupIntegratorParams](#) ()
- void [AdvanceSystemRebound](#) ()
- void [AdditionalForces](#) ()
- void [OutputElements](#) ()
- void [OutputNbodySimulation](#) ()
- boolean [ChkCloseEncWithPI](#) ()
- void [DiscardParticlesDist](#) ()
- void [DiscardParticlesUnbound](#) ()
- int [ResolveCollisions](#) ()
- struct reb_simulation * [RestartReboundSimulation](#) ()
- void [SynchronizeFargoRebound](#) ()
- void [MinStepForRebound](#) ()
- real [DampingTW04](#) ()
- real [GetPsysInfoFromRsim](#) ()
- void [DumpOmegaFrame](#) ()
- real [GetOmegaFrame](#) ()
- void [InitPebbleArrays](#) ()
Initialise polar arrays associated with the pebble disk.
- void [EquilPebbleDisk](#) ()
- void [InitPebblesViaFlux](#) ()
- void [RestartPebbleDisk](#) ()
- void [PebbleStokesNumbers](#) ()
- void [AccretePebblesOntoPlanets](#) ()
- void [CorrectPebblesVtheta](#) ()
- void [EvolvePebbleDisk](#) ()
- void [WritePebbles](#) ()
- real [Trapzd](#) ()
- real [IntegrateColumnMass](#) ()
- void [EtaPressureSupport](#) ()
- void [DampPebbles](#) ()
- void [TransportPebbles](#) ()
- void [OneWindRadPebbles](#) ()
- void [OneWindThetaPebbles](#) ()
- void [QuantitiesAdvectionPebbles](#) ()
- void [SourceTermsPebbles](#) ()
- void [SubStep1Pebbles](#) ()
- boolean [DetectCrashPebbles](#) ()
Safety check for negative pebble densities.
- void [SynchronizePebbleDisc](#) ()
Synchronises pebble fluid hydrodynamic quantities among the overlapping grid zones.

- void [CriticalCharTime](#) ()
- void [ParticleDiffusion](#) ()
- void [BckpFieldsForBC](#) ()

Backs up the initial state of the pebble disk to impose damping boundary conditions later.

- void [ParametricAccretion](#) ()
- void [CreateTorqueMapInfile](#) ()

4.23.1 Detailed Description

Declaration of all the functions of the FARGO code.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [proto.h](#).

4.23.2 Function Documentation

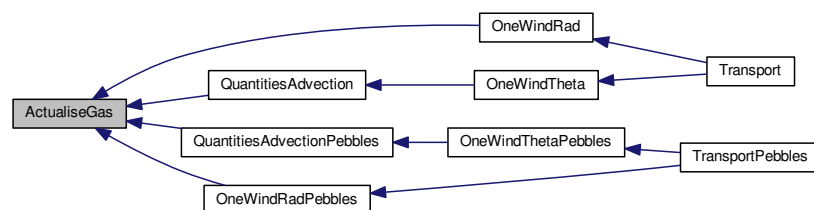
4.23.2.1 void AccreteOntoPlanets ()

4.23.2.2 void AccretePebblesOntoPlanets ()

4.23.2.3 void ActualiseGas ()

Referenced by [OneWindRad\(\)](#), [OneWindRadPebbles\(\)](#), [QuantitiesAdvection\(\)](#), and [QuantitiesAdvectionPebbles\(\)](#).

Here is the caller graph for this function:



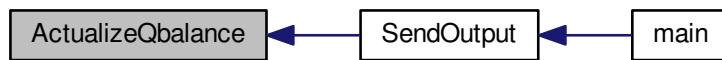
4.23.2.4 void ActualizeQbalance ()

Definition at line 239 of file `Output.c`.

References `polargrid::Field`, `polargrid::Nrad`, `polargrid::Nsec`, `Qbalance`, `Qminus`, and `Qplus`.

Referenced by `SendOutput()`.

Here is the caller graph for this function:



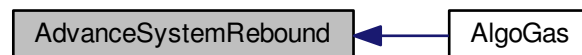
4.23.2.5 void AdditionalForces ()

4.23.2.6 void AdvanceSystemFromDisk ()

4.23.2.7 void AdvanceSystemRebound ()

Referenced by AlgoGas().

Here is the caller graph for this function:



4.23.2.8 void AdvectSHIFT ()

4.23.2.9 void AlgoGas ()

Referenced by main().

Here is the caller graph for this function:



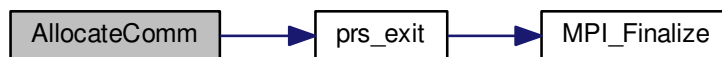
4.23.2.10 void AllocateComm ()

Definition at line 28 of file commbound.c.

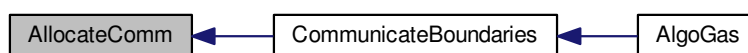
References AdvecteLabel, allocated_com, CPU_Rank, CPUOVERLAP, EnergyEq, NSEC, prs_exit(), RecvInner↔Boundary, RecvOuterBoundary, SendInnerBoundary, SendOuterBoundary, size_com, and YES.

Referenced by CommunicateBoundaries().

Here is the call graph for this function:



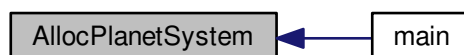
Here is the caller graph for this function:



4.23.2.11 PlanetarySystem* AllocPlanetSystem ()

Referenced by main().

Here is the caller graph for this function:



4.23.2.12 void ApplyBoundaryCondition ()

Referenced by AlgoGas().

Here is the caller graph for this function:

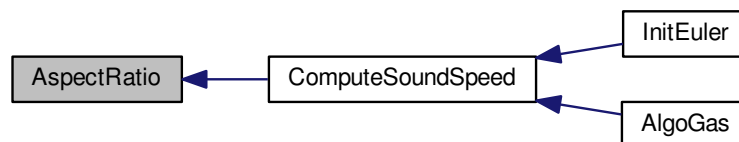


4.23.2.13 `void ApplyOuterSourceMass ()`

4.23.2.14 `real AspectRatio ()`

Referenced by `ComputeSoundSpeed()`.

Here is the caller graph for this function:



4.23.2.15 `void BckpFieldsForBC ()`

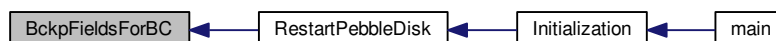
Backs up the initial state of the pebble disk to impose damping boundary conditions later.

Definition at line 168 of file `Pebbles.c`.

References `polargrid::Field`, `polargrid::Nrad`, `polargrid::Nsec`, `OmegaFrame`, `PebbleDens`, `PebbleVrad`, `PebbleVtheta`, `PebDensInit`, `PebVradInit`, `PebVthetaInit`, and `Rmed`.

Referenced by `RestartPebbleDisk()`.

Here is the caller graph for this function:



4.23.2.16 `void CalculateFlaring ()`

Calculates the sine of the grazing angle by reconstructing the surface from the pressure scale height.

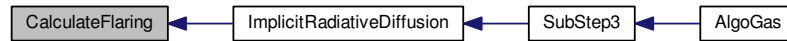
See Eq. (15) in Chrenko et al. (2017).

Definition at line 147 of file EnergySources.c.

References AU_SI, polargrid::Field, InvDiffRsup, InvRmed, polargrid::Nrad, polargrid::Nsec, OmegaInv, Rinf, Rmed, Rsup, SoundSpeed, SQRT_ADIABIND_INV, and STELLARRADIUS.

Referenced by ImplicitRadiativeDiffusion().

Here is the caller graph for this function:



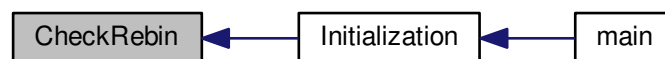
4.23.2.17 void CalculateQirr ()

4.23.2.18 void CalculateQminus ()

4.23.2.19 void CheckRebin ()

Referenced by Initialization().

Here is the caller graph for this function:



4.23.2.20 void ChessBoardIndexing ()

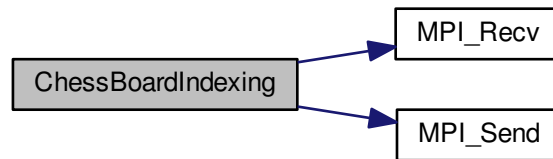
Function ensures the odd-even ordering of the SOR method when the grid is split on multiple CPUs.

Definition at line 486 of file EnergySources.c.

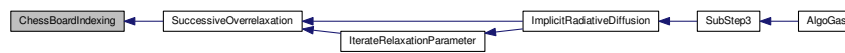
References CPU_Master, CPU_Number, CPU_Rank, CPUOVERLAP, fargostat, jchess1st, jchess2nd, MPI_CO↔MM_WORLD, MPI_INT, MPI_Recv(), MPI_Send(), and NRAD.

Referenced by SuccessiveOverrelaxation().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.21 **boolean** ChkCloseEncWithPI ()

4.23.2.22 **void** CommunicateBoundaries ()

4.23.2.23 **void** ComputeAverageThetaVelocities ()

4.23.2.24 **void** ComputeConstantResidual ()

4.23.2.25 **void** ComputeExtQty ()

4.23.2.26 **void** ComputeLRMomenta ()

4.23.2.27 **void** ComputePressureField ()

Referenced by Initialization().

Here is the caller graph for this function:

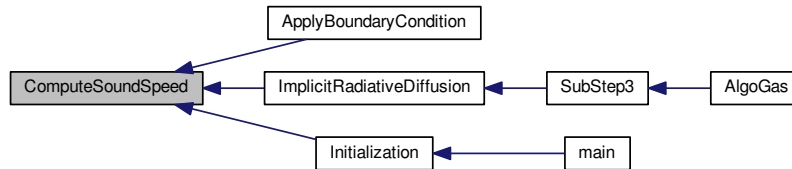


4.23.2.28 **void** ComputeResiduals ()

4.23.2.29 void ComputeSoundSpeed ()

Referenced by ApplyBoundaryCondition(), ImplicitRadiativeDiffusion(), and Initialization().

Here is the caller graph for this function:



4.23.2.30 void ComputeSpeQty ()

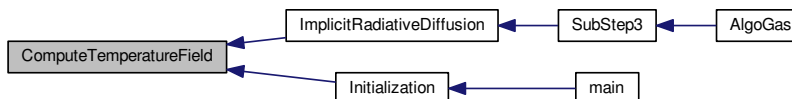
4.23.2.31 void ComputeStarRad ()

4.23.2.32 void ComputeStarTheta ()

4.23.2.33 void ComputeTemperatureField ()

Referenced by ImplicitRadiativeDiffusion(), and Initialization().

Here is the caller graph for this function:



4.23.2.34 void ComputeThetaElongations ()

4.23.2.35 void ComputeVelocities ()

4.23.2.36 int ConditionCFL ()

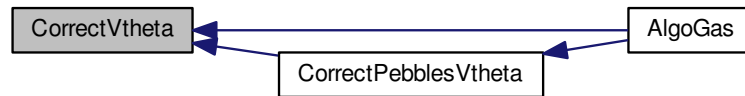
4.23.2.37 real ConstructSequence ()

4.23.2.38 void CorrectPebblesVtheta ()

4.23.2.39 void CorrectVtheta ()

Referenced by AlgoGas(), and CorrectPebblesVtheta().

Here is the caller graph for this function:



4.23.2.40 **PolarGrid*** `CreatePolarGrid ()`

4.23.2.41 `void CreateTorqueMapInfile ()`

4.23.2.42 `void CriticalCharTime ()`

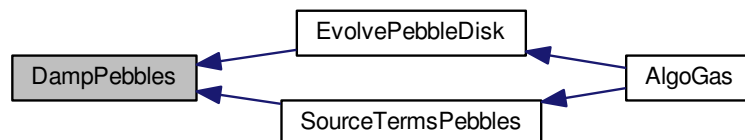
4.23.2.43 `void DampingBoundary ()`

4.23.2.44 `real DampingTW04 ()`

4.23.2.45 `void DampPebbles ()`

Referenced by `EvolvePebbleDisk()`, and `SourceTermsPebbles()`.

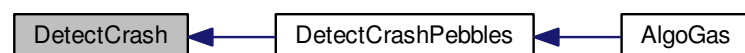
Here is the caller graph for this function:



4.23.2.46 **boolean** `DetectCrash ()`

Referenced by `DetectCrashPebbles()`.

Here is the caller graph for this function:



4.23.2.47 boolean DetectCrashPebbles ()

Safety check for negative pebble densities.

Definition at line 753 of file Pebbles.c.

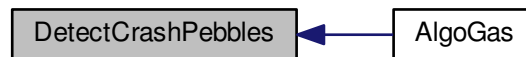
References DetectCrash(), and PebbleDens.

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:

**4.23.2.48 void DiffusionCoefs ()**

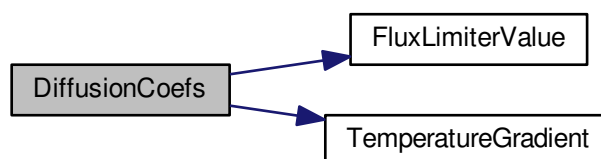
Calculation of the diffusion coefficients.

Definition at line 527 of file EnergySources.c.

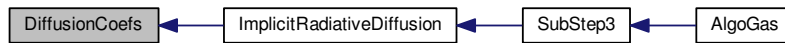
References polargrid::Field, FluxLimiterValue(), polargrid::Nrad, polargrid::Nsec, STEFANBOLTZMANN, Temperature, and TemperatureGradient().

Referenced by ImplicitRadiativeDiffusion().

Here is the call graph for this function:



Here is the caller graph for this function:



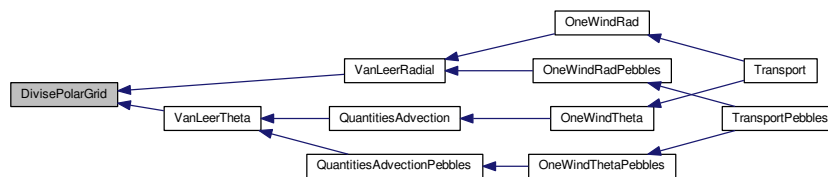
4.23.2.49 void DiscardParticlesDist ()

4.23.2.50 void DiscardParticlesUnbound ()

4.23.2.51 void DivisePolarGrid ()

Referenced by VanLeerRadial(), and VanLeerTheta().

Here is the caller graph for this function:



4.23.2.52 void DumpOmegaFrame ()

4.23.2.53 void DumpSources ()

4.23.2.54 real EffectiveOpticalDepth ()

4.23.2.55 void EmptyPlanetSystemFile ()

4.23.2.56 real Energy ()

4.23.2.57 void EquilPebbleDisk ()

4.23.2.58 void EtaPressureSupport ()

4.23.2.59 void EvolvePebbleDisk ()

4.23.2.60 void FillCoolingTime ()

Definition at line 110 of file Theo.c.

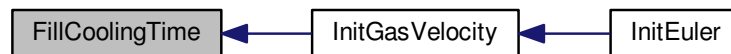
References CoolingTimeMed, InitCoolingTime(), NRAD, and Rmed.

Referenced by InitGasVelocity().

Here is the call graph for this function:



Here is the caller graph for this function:



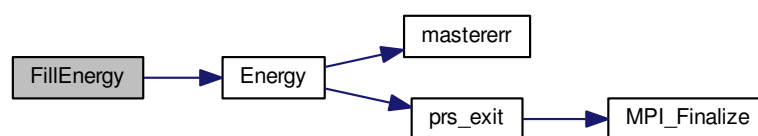
4.23.2.61 void FillEnergy ()

Definition at line 76 of file Theo.c.

References Energy(), EnergyMed, NRAD, and Rmed.

Referenced by InitEuler().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.62 void FillForcesArrays ()

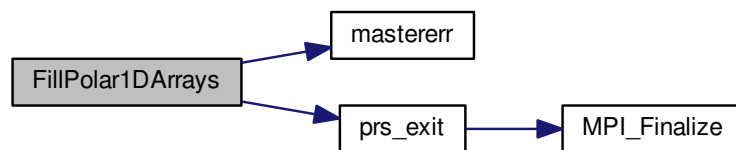
4.23.2.63 void FillPolar1DArrays ()

Definition at line 57 of file SourceEuler.c.

References CPU_Master, GLOBALNRAD, GlobalRmed, IMIN, InvDiffRmed, InvDiffRsup, InvRinf, InvRmed, InvSurf, LogGrid, mastererr(), NRAD, NSEC, OmegaInv, OUTPUTDIR, PI, prs_exit(), Radii, Rinf, RMAX, Rmed, Rmed2, RMIN, Rsup, Surf, and YES.

Referenced by InitEuler().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.64 void FillQplus ()

Definition at line 125 of file Theo.c.

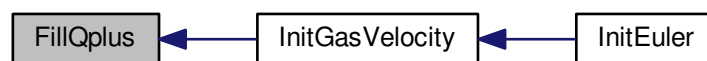
References InitQplus(), NRAD, QplusMed, and Rmed.

Referenced by InitGasVelocity().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.65 void FillSigma ()

Definition at line 25 of file Theo.c.

References NRAD, Rinf, Rmed, Sigma(), SigmaInf, and SigmaMed.

Referenced by InitEuler().

Here is the call graph for this function:



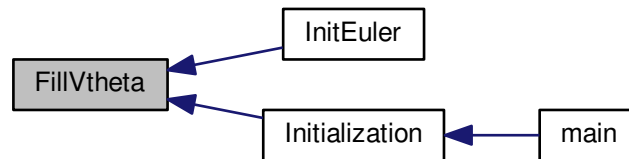
Here is the caller graph for this function:



4.23.2.66 void FillVtheta ()

Referenced by InitEuler(), and Initialization().

Here is the caller graph for this function:



4.23.2.67 `int FindNumberOfPlanets ()`

Referenced by `main()`.

Here is the caller graph for this function:



4.23.2.68 `void FindOrbitalElements ()`

4.23.2.69 `real FluxLimiterValue ()`

4.23.2.70 `FILE* fopenp ()`

4.23.2.71 `void FreePlanetary ()`

Referenced by `main()`.

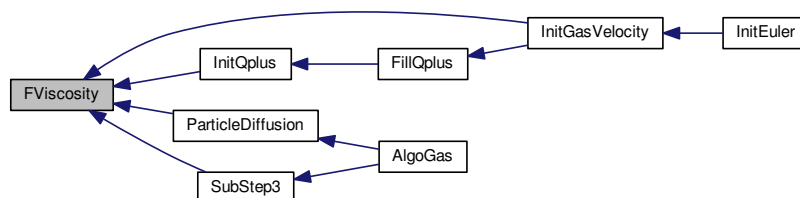
Here is the caller graph for this function:



4.23.2.72 `real FViscosity ()`

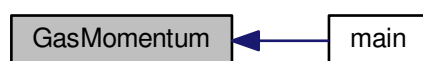
Referenced by `InitGasVelocity()`, `InitQplus()`, `ParticleDiffusion()`, and `SubStep3()`.

Here is the caller graph for this function:

4.23.2.73 `real GasMomentum ()`

Referenced by `main()`.

Here is the caller graph for this function:

4.23.2.74 `real GasTotalEnergy ()`

Referenced by `main()`.

Here is the caller graph for this function:

4.23.2.75 `real GasTotalMass ()`

Referenced by `main()`.

Here is the caller graph for this function:



4.23.2.76 `real GetfromPlanetFile ()`

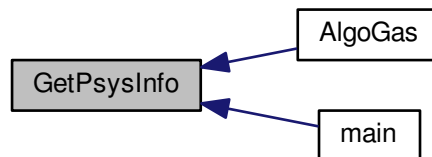
4.23.2.77 `real GetGlobalFrac ()`

4.23.2.78 `real GetOmegaFrame ()`

4.23.2.79 `real GetPsysInfo ()`

Referenced by `AlgoGas()`, and `main()`.

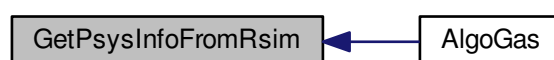
Here is the caller graph for this function:



4.23.2.80 `real GetPsysInfoFromRsim ()`

Referenced by `AlgoGas()`.

Here is the caller graph for this function:



4.23.2.81 void GiveSpecificTime ()

4.23.2.82 void GiveTimeInfo ()

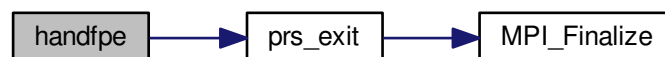
4.23.2.83 void handfpe ()

Definition at line 8 of file fpe.c.

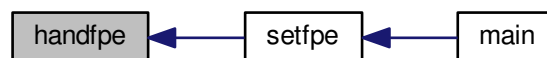
References CPU_Rank, and prs_exit().

Referenced by setfpe().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.84 void ImplicitRadiativeDiffusion ()

4.23.2.85 void ImposeKeplerianEdges ()

Referenced by SubStep1().

Here is the caller graph for this function:



4.23.2.86 void InitComputeAccel ()

Definition at line 93 of file SideEuler.c.

References CellAbcissa, CellOrdinate, CreatePolarGrid(), polargrid::Field, NRAD, polargrid::Nrad, NSEC, polargrid::Nsec, PI, and Rmed.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:

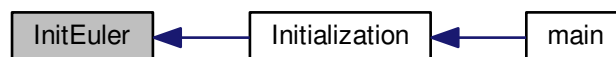


4.23.2.87 real InitCoolingTime ()

4.23.2.88 void InitEuler ()

Referenced by Initialization().

Here is the caller graph for this function:



4.23.2.89 void InitGas ()

4.23.2.90 void InitGasDensityEnergy ()

4.23.2.91 void InitGasVelocity ()

4.23.2.92 void Initialization ()

4.23.2.93 void InitLabel ()

4.23.2.94 void InitPebbleArrays ()

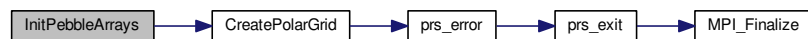
Initialise polar arrays associated with the pebble disk.

Definition at line 37 of file Pebbles.c.

References CreatePolarGrid(), DragForceRad, DragForceTheta, GasAccelrad, GasAcceltheta, NRAD, NSEC, P_←, EBBLEBULKDENS, PebbleDens, PebbleVrad, PebbleVtheta, pebbulkdens, RHO2CGS, and StokesNumber.

Referenced by InitEuler().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.95 void InitPebblesViaFlux ()

4.23.2.96 PlanetarySystem* InitPlanetarySystem ()

4.23.2.97 real InitQplus ()

4.23.2.98 void InitRadiatDiffusionFields ()

Initialises the polar arrays associated with the heating/cooling processes.

Definition at line 46 of file EnergySources.c.

References ADIABIND, CreatePolarGrid(), CV, GASCONST, MOLWEIGHT, NRAD, NSEC, Qbalance, Qminus, and Write_Qbalance.

Referenced by InitEuler().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.99 `void InitSpecificTime ()`

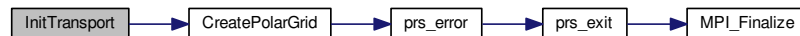
4.23.2.100 `void InitTransport ()`

Definition at line 217 of file `TransportEuler.c`.

References `CreatePolarGrid()`, `dq`, `NRAD`, `NSEC`, and `TempShift`.

Referenced by `InitEuler()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.101 `void InitVariables ()`

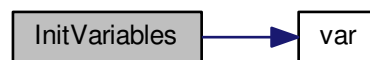
Definition at line 17 of file `var.c`.

References `ACCRETIONALHEATING`, `ACCRETIONRATE`, `ADIABIND`, `ADVLABEL`, `ALPHAVISCOSITY`, `ASPECTRATIO`, `BACKREACTION`, `CAVITYRADIUS`, `CAVITYRATIO`, `CAVITYWIDTH`, `COOLINGTIME`, `DAMPINGPERIODFRAC`, `DAMPINGRMAXFRAC`, `DAMPINGRMINFRAC`, `DAMPTOWARDS`, `DENSINFILE`, `DISCALBEDO`, `DISK`, `DT`, `ECCENTRICITY`, `EFFECTIVETEMPERATURE`, `ENERGYEQUATION`, `EXCLUDEHILL`, `FLARINGINDEX`, `FRAME`, `GETTORQUEFORPLANET`, `GRIDSPACING`, `HEATINGDELAY`, `HILLCUT`, `IAS15MINDT`, `IAS15PRECISION`, `IMPOSEDDISKDRIFT`, `INDIRECTTERM`, `INITIALIZEFROMFILE`, `INT`, `LAMBDA DOUBLING`, `MASSTAPER`, `NINTERM`, `NO`, `NOUTELEMENTS`, `NRAD`, `NSEC`, `NTOT`, `OMEGAFRAME`, `OPACITYDROP`, `OPENINNERBOUNDARY`, `OUTERSOURCEMASS`, `OUTPUTDIR`, `PARAMETRICACCRETION`, `PARAMETRICOPACITY`, `PARTICLE`

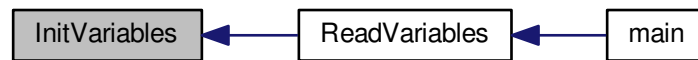
DIFFUSION, PEBBLEACCRETION, PEBBLEALPHA, PEBBLEBULKDENS, PEBBLECOAGULATION, PEBBLEFLUX, PLANETARYDENSITY, PLANETCONFIG, PLANETSFEELDISK, REAL, RELEASDATE, RELEASERADIUS, RESOLVECOLLISIONS, RMAX, RMIN, ROCHESMOOTHING, SCHMIDTNUMBER, SIGMA0, SIGMASLOPE, STELLARIRRADIATION, STELLARRADIUS, STRING, TARGETNPL, TEMPERINFILE, THICKNESSSMOOTHING, TORQUEMAPINFILE, TRANSITIONRADIUS, TRANSITIONRATIO, TRANSITIONWIDTH, TRANSPORT, var(), VERTICALDAMPING, VISCOSITY, VRADINFILE, VTHETAFILE, WRITEDENSITY, WRITEDIVV, WRITEENERGY, WRITEETA, WRITEQBALANCE, WRITEQPLUS, WRITETEMPERATURE, WRITETORQUEFILES, WRITEVELOCITY, and YES.

Referenced by ReadVariables().

Here is the call graph for this function:



Here is the caller graph for this function:



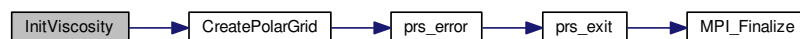
4.23.2.102 void InitViscosity ()

Definition at line 67 of file Viscosity.c.

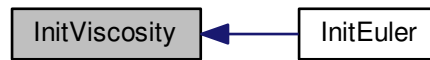
References CreatePolarGrid(), DivergenceVelocity, NRAD, NSEC, TAUPP, TAURP, and TAURR.

Referenced by InitEuler().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.103 **real** IntegrateColumnMass ()

4.23.2.104 **void** IterateRelaxationParameter ()

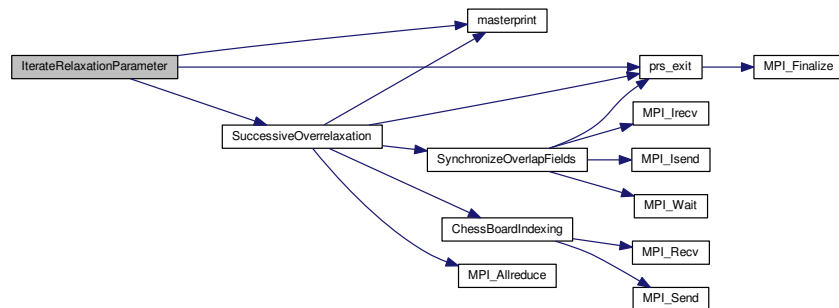
When solving the energy equation for the first time, the function spans through various values of the SOR parameter in order to find its best value to start with.

Definition at line 338 of file EnergySources.c.

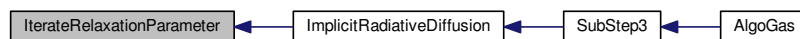
References domega, polargrid::Field, masterprint(), Niterbest, NO, polargrid::Nrad, polargrid::Nsec, omegabest, prs_exit(), SORMAXITERS, SuccessiveOverrelaxation(), Temperature, and YES.

Referenced by ImplicitRadiativeDiffusion().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.105 **void** ListPlanets ()

Referenced by main().

Here is the caller graph for this function:



4.23.2.106 void MakeDir ()

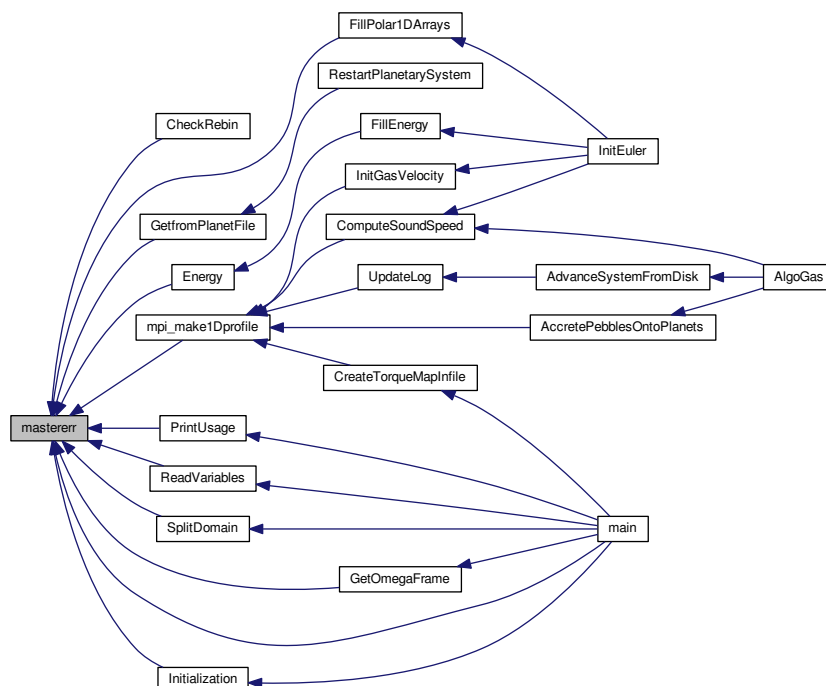
4.23.2.107 void mastererr (const char * *template*, ...)

Definition at line 49 of file LowTasks.c.

References CPU_Master.

Referenced by CheckRebin(), Energy(), FillPolar1DArrays(), GetfromPlanetFile(), GetOmegaFrame(), Initialization(), main(), mpi_make1Dprofile(), PrintUsage(), ReadVariables(), and SplitDomain().

Here is the caller graph for this function:



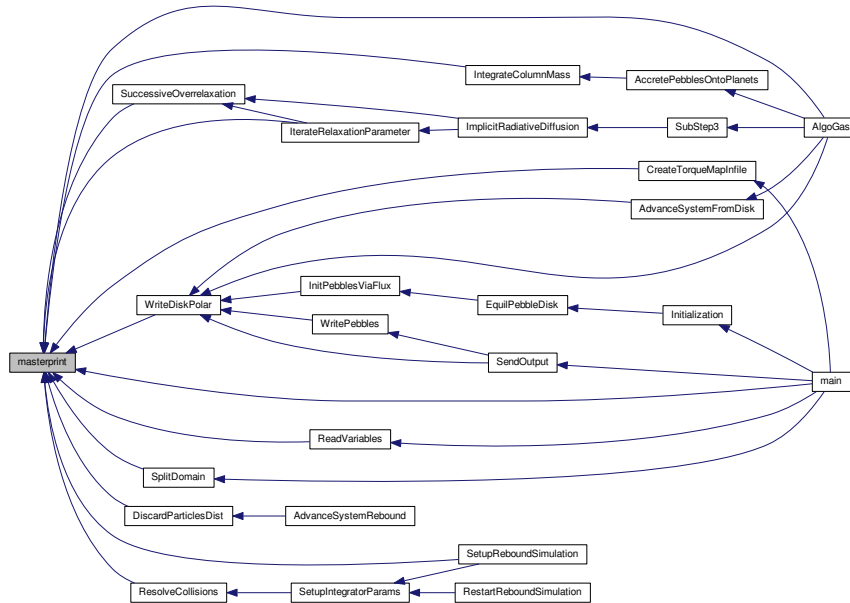
4.23.2.108 void masterprint (const char * *template*, ...)

Definition at line 40 of file LowTasks.c.

References CPU_Master.

Referenced by AlgoGas(), CreateTorqueMapInfile(), DiscardParticlesDist(), IntegrateColumnMass(), IterateRelaxationParameter(), main(), ReadVariables(), ResolveCollisions(), SetupReboundSimulation(), SplitDomain(), SuccessiveOverrelaxation(), and WriteDiskPolar().

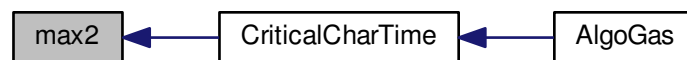
Here is the caller graph for this function:



4.23.2.109 real max2 ()

Referenced by CriticalCharTime().

Here is the caller graph for this function:



4.23.2.110 void merge ()

4.23.2.111 void message ()

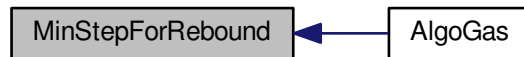
4.23.2.112 void MidplaneVolumeDensity ()

4.23.2.113 real min2 ()

4.23.2.114 void MinStepForRebound ()

Referenced by AlgoGas().

Here is the caller graph for this function:



4.23.2.115 void mpi_make1Dprofile ()

4.23.2.116 void MultiplyPolarGridbyConstant ()

4.23.2.117 void NonReflectingBoundary ()

4.23.2.118 void OneWindRad ()

4.23.2.119 void OneWindRadPebbles ()

4.23.2.120 void OneWindTheta ()

4.23.2.121 void OneWindThetaPebbles ()

4.23.2.122 void OpacityProfile ()

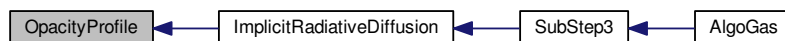
Fills the opacity polar grid, either with a fixed parametric value or using the Bell & Lin (1994) opacity table.

Definition at line 656 of file EnergySources.c.

References `a`, `b`, `polargrid::Field`, `kappa0`, `NO`, `polargrid::Nrad`, `polargrid::Nsec`, `OPA2CU`, `PARAMETRICOPACITY`, `RHO2CGS`, `T2SI`, `Temperature`, and `YES`.

Referenced by `ImplicitRadiativeDiffusion()`.

Here is the caller graph for this function:

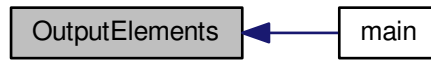


4.23.2.123 void OpenBoundary ()

4.23.2.124 void OutputElements ()

Referenced by `main()`.

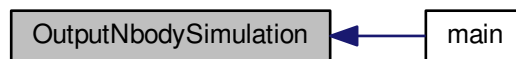
Here is the caller graph for this function:



4.23.2.125 void OutputNbodySimulation ()

Referenced by main().

Here is the caller graph for this function:



4.23.2.126 void ParametricAccretion ()

4.23.2.127 void ParticleDiffusion ()

4.23.2.128 void PebbleStokesNumbers ()

4.23.2.129 void PrintUsage ()

4.23.2.130 void prs_error ()

4.23.2.131 void prs_exit ()

4.23.2.132 void QuantitiesAdvection ()

4.23.2.133 void QuantitiesAdvectionPebbles ()

4.23.2.134 void ReadfromAsciiFile ()

4.23.2.135 void ReadfromFile ()

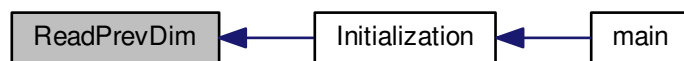
4.23.2.136 void ReadPrevDim ()

Definition at line 16 of file rebin.c.

References CPU_Master, OldNRAD, OldNSEC, OldRadii, OldRmed, and OUTPUTDIR.

Referenced by Initialization().

Here is the caller graph for this function:

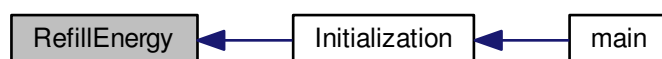


4.23.2.137 `void ReadVariables ()`

4.23.2.138 `void RefillEnergy ()`

Referenced by `Initialization()`.

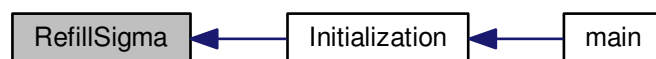
Here is the caller graph for this function:



4.23.2.139 `void RefillSigma ()`

Referenced by `Initialization()`.

Here is the caller graph for this function:



4.23.2.140 `int ResolveCollisions ()`

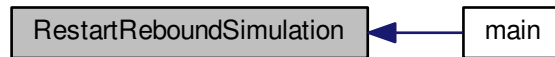
4.23.2.141 `void RestartPebbleDisk ()`

4.23.2.142 `void RestartPlanetarySystem ()`

4.23.2.143 struct reb_simulation* RestartReboundSimulation ()

Referenced by main().

Here is the caller graph for this function:



4.23.2.144 void RotatePsys ()

Referenced by AlgoGas().

Here is the caller graph for this function:



4.23.2.145 void SendOutput ()

4.23.2.146 void setfpe ()

Definition at line 15 of file fpe.c.

References handfpe().

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:

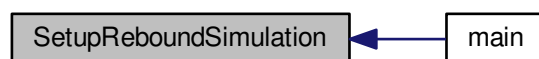


4.23.2.147 void SetupIntegratorParams ()

4.23.2.148 struct reb_simulation* SetupReboundSimulation ()

Referenced by main().

Here is the caller graph for this function:



4.23.2.149 void SetWaveKillingZones ()

Sets the wave-killing factors within the damping zones; inspired by de Val-Borro et al. (2006).

Definition at line 242 of file SideEuler.c.

References DAMPINGPERIODFRAC, DAMPINGRMAXFRAC, DAMPINGRMINFRAC, Max_or_active, PI, RMAX, Rmed, RMIN, WaveKiller, and Zero_or_active.

Referenced by InitEuler().

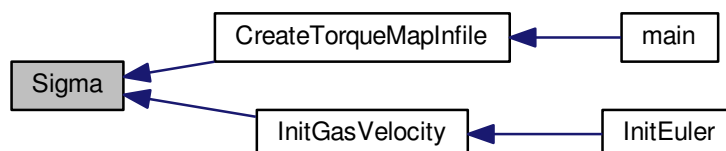
Here is the caller graph for this function:



4.23.2.150 `real Sigma ()`

Referenced by `CreateTorqueMapInfile()`, and `InitGasVelocity()`.

Here is the caller graph for this function:



4.23.2.151 `void SourceTermsPebbles ()`

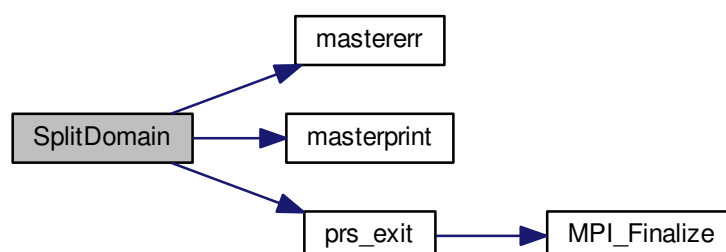
4.23.2.152 `void SplitDomain ()`

Definition at line 24 of file `split.c`.

References `CPU_Number`, `CPU_Rank`, `CPUOVERLAP`, `debug`, `GLOBALNRAD`, `IMAX`, `IMIN`, `mastererr()`, `masterprint()`, `Max_or_active`, `MaxMO_or_active`, `NRAD`, `One_or_active`, `prs_exit()`, `YES`, and `Zero_or_active`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.153 void SubStep1 ()

4.23.2.154 void SubStep1Pebbles ()

4.23.2.155 void SubStep2 ()

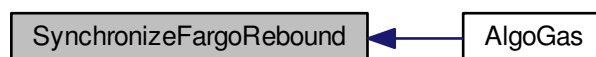
4.23.2.156 void SubStep3 ()

4.23.2.157 int SuccessiveOverrelaxation ()

4.23.2.158 void SynchronizeFargoRebound ()

Referenced by AlgoGas().

Here is the caller graph for this function:



4.23.2.159 void SynchronizeOverlapFields ()

4.23.2.160 void SynchronizePebbleDisc ()

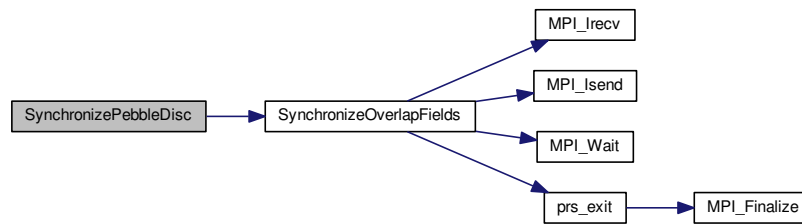
Synchronises pebble fluid hydrodynamic quantities among the overlapping grid zones.

Definition at line 685 of file Pebbles.c.

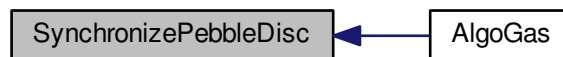
References CPU_Number, CPUOVERLAP, polargrid::Field, polargrid::Nrad, PebbleDens, PebbleVrad, PebbleVtheta, and SynchronizeOverlapFields().

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



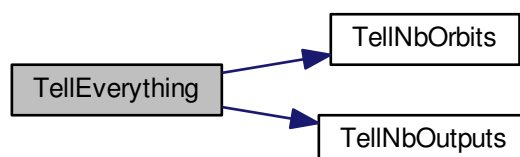
4.23.2.161 void TellEverything ()

Definition at line 281 of file Interpret.c.

References `AdvecteLabel`, `ASPECTRATIO`, `CPU_Master`, `DT`, `EnergyEq`, `G`, `NINTERM`, `NRAD`, `NSEC`, `NTOT`, `ParametricCooling`, `PI`, `RMAX`, `RMIN`, `SIGMA0`, `TellNbOrbits()`, `TellNbOutputs()`, and `YES`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.162 **real** TellNbOrbits ()

4.23.2.163 **real** TellNbOutputs ()

4.23.2.164 **void** TemperatureGradient ()

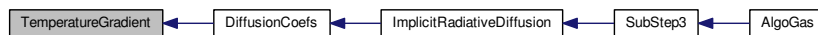
Finds the temperature gradients and their magnitude over the mesh.

Definition at line 580 of file EnergySources.c.

References polargrid::Field, InvDiffRmed, polargrid::Nrad, polargrid::Nsec, PI, Rmed, and Temperature.

Referenced by DiffusionCoefs().

Here is the caller graph for this function:



4.23.2.165 **real** ThicknessSmoothing ()

4.23.2.166 **void** Transport ()

Referenced by AlgoGas().

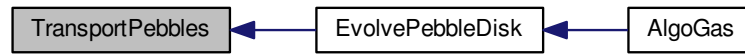
Here is the caller graph for this function:



4.23.2.167 **void** TransportPebbles ()

Referenced by EvolvePebbleDisk().

Here is the caller graph for this function:

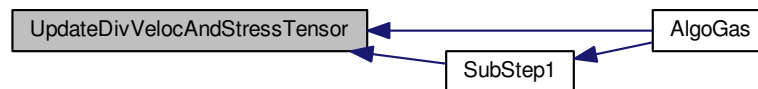


4.23.2.168 `real Trapzd ()`

4.23.2.169 `void UpdateDivVelocAndStressTensor ()`

Referenced by `AlgoGas()`, and `SubStep1()`.

Here is the caller graph for this function:



4.23.2.170 `void UpdateLog ()`

4.23.2.171 `void UpdateVelocityWithViscousTerms ()`

Referenced by `SubStep1()`.

Here is the caller graph for this function:



4.23.2.172 `real VanLeerRadial ()`

4.23.2.173 `void VanLeerTheta ()`

4.23.2.174 `void var ()`

4.23.2.175 void ViscousTerms ()

4.23.2.176 void WriteBigPlanetFile ()

4.23.2.177 void WriteBigPlanetSystemFile ()

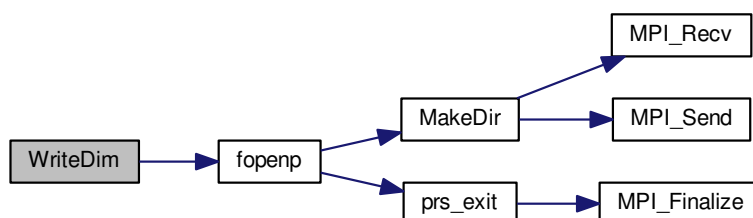
4.23.2.178 void WriteDim ()

Definition at line 191 of file Output.c.

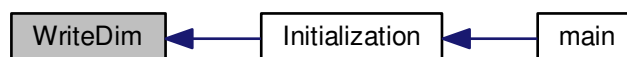
References CPU_Master, fopenp(), GLOBALNRAD, NINTERM, NSEC, NTOT, OUTPUTDIR, and RMAX.

Referenced by Initialization().

Here is the call graph for this function:



Here is the caller graph for this function:



4.23.2.179 void WriteDiskPolar ()

4.23.2.180 void WritePebbles ()

4.23.2.181 void WritePlanetFile ()

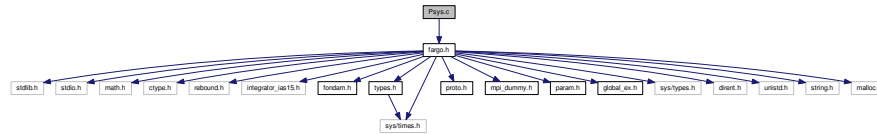
4.23.2.182 void WritePlanetSystemFile ()

4.24 Psys.c File Reference

Contains the functions that set up the planetary system configuration.

```
#include "fargo.h"
```

Include dependency graph for Psys.c:



Functions

- int [FindNumberOfPlanets](#) (char *filename)
- [PlanetarySystem](#) * [AllocPlanetSystem](#) (int nb)
- void [FreePlanetary](#) ([PlanetarySystem](#) *sys)
- [PlanetarySystem](#) * [InitPlanetarySystem](#) (char *filename)
- void [ListPlanets](#) ([PlanetarySystem](#) *sys)
- [real](#) [GetPsysInfo](#) ([PlanetarySystem](#) *sys, [boolean](#) action)

The original function was modified to handle 3D inclined orbits.
- [real](#) [GetPsysInfoFromRsim](#) (struct reb_simulation *rsim, [boolean](#) action)

An analogue of the [GetPsysInfo\(\)](#) function; here a rebound simulation structure is used as a call argument.
- void [RotatePsys](#) (struct reb_simulation *rsim, [real](#) angle)

Synchronises the planetary system with the frame.

Variables

- static [real](#) [Xplanet](#)
- static [real](#) [Yplanet](#)
- [boolean](#) [GuidingCenter](#)

4.24.1 Detailed Description

Contains the functions that set up the planetary system configuration.

In addition, the last two functions allow to track the first planet (number 0) of the planetary system, in order to perform a calculation in the frame corotating either with this planet or with its guiding-center.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [Psys.c](#).

4.24.2 Function Documentation

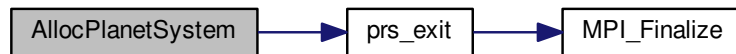
4.24.2.1 [PlanetarySystem](#)* [AllocPlanetSystem](#) (int nb)

Definition at line 40 of file [Psys.c](#).

References [planetary_system::acc](#), [planetary_system::ax](#), [planetary_system::ay](#), [planetary_system::az](#), [planetary_system::FeelDisk](#), [planetary_system::FeelOthers](#), [planetary_system::mass](#), [prs_exit\(\)](#), [planetary_system::vx](#), [planetary_system::vy](#), [planetary_system::vz](#), [planetary_system::x](#), [planetary_system::y](#), [YES](#), and [planetary_system::z](#).

Referenced by InitPlanetarySystem().

Here is the call graph for this function:



Here is the caller graph for this function:



4.24.2.2 int FindNumberOfPlanets (char * filename)

Definition at line 21 of file Psys.c.

References prs_exit().

Referenced by InitPlanetarySystem().

Here is the call graph for this function:



Here is the caller graph for this function:



4.24.2.3 void FreePlanetary (PlanetarySystem * sys)

Definition at line 98 of file Psys.c.

4.24.2.4 real GetPsysInfo (PlanetarySystem * sys, boolean action)

The original function was modified to handle 3D inclined orbits.

It is similar to the one used in fargo3D but employs tools from the REBOUND package.

Definition at line 189 of file Psys.c.

References FREQUENCY, G, GET, GuidingCenter, MARK, planetary_system::x, Xplanet, YES, and Yplanet.

4.24.2.5 real GetPsysInfoFromRsim (struct reb_simulation * rsim, boolean action)

An analogue of the [GetPsysInfo\(\)](#) function; here a rebound simulation structure is used as a call argument.

Definition at line 262 of file Psys.c.

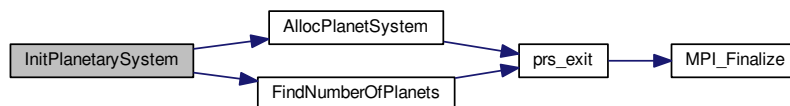
References FREQUENCY, G, GET, GuidingCenter, MARK, Xplanet, YES, and Yplanet.

4.24.2.6 PlanetarySystem* InitPlanetarySystem (char * filename)

Definition at line 117 of file Psys.c.

References planetary_system::acc, AllocPlanetSystem(), CPU_Master, ECCENTRICITY, planetary_system::FeelDisk, planetary_system::FeelOthers, FindNumberOfPlanets(), planetary_system::mass, planetary_system::nb, NO, planetary_system::vx, planetary_system::vy, planetary_system::x, planetary_system::y, and YES.

Here is the call graph for this function:



4.24.2.7 void ListPlanets (PlanetarySystem * sys)

Definition at line 155 of file Psys.c.

References CPU_Master, planetary_system::nb, and YES.

4.24.2.8 void RotatePsys (struct reb_simulation * rsim, real angle)

Synchronises the planetary system with the frame.

Affects the rebound simulation structure only, 'sys' and 'rsim' are synchronised later on.

Definition at line 330 of file Psys.c.

4.24.3 Variable Documentation

4.24.3.1 **boolean** GuidingCenter

Definition at line 29 of file Interpret.c.

Referenced by GetPsysInfo(), GetPsysInfoFromRsim(), and ReadVariables().

4.24.3.2 **real** Xplanet [static]

Definition at line 17 of file Psys.c.

Referenced by GetPsysInfo(), and GetPsysInfoFromRsim().

4.24.3.3 **real** Yplanet [static]

Definition at line 17 of file Psys.c.

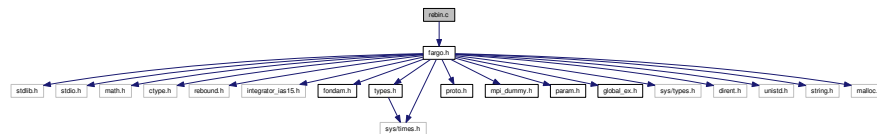
Referenced by GetPsysInfo(), and GetPsysInfoFromRsim().

4.25 rebin.c File Reference

Resample the hydrodynamical fields at a restart with a different resolution.

```
#include "fargo.h"
```

Include dependency graph for rebin.c:



Functions

- void [ReadPrevDim](#) ()
- void [CheckRebin](#) (int nb)

Variables

- static [real](#) OldRadii [MAX1D]
- static [real](#) OldRmed [MAX1D]
- static [real](#) New_r [MAX1D]
- static int OldNRAD
- static int OldNSEC

4.25.1 Detailed Description

Resample the hydrodynamical fields at a restart with a different resolution.

Note that at the restart, even if NRAD and NSEC coincide with previous values, the data is resampled if the radii do not coincide (for instance, if we switch from ARITHMETIC to LOGARITHMIC spacing).

Definition in file [rebin.c](#).

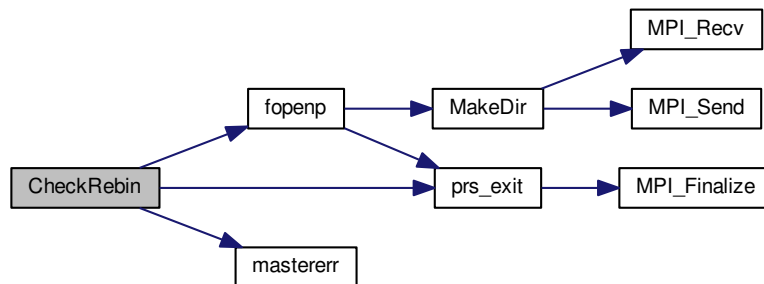
4.25.2 Function Documentation

4.25.2.1 void CheckRebin (int *nb*)

Definition at line 43 of file rebin.c.

References CPU_Master, fopenp(), GLOBALNRAD, GlobalRmed, mastererr(), New_r, NO, NRAD, NSEC, OldNRAD, OldNSEC, OldRadii, OldRmed, OUTPUTDIR, prs_exit(), Radii, and YES.

Here is the call graph for this function:



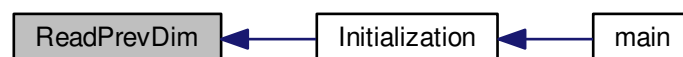
4.25.2.2 void ReadPrevDim ()

Definition at line 16 of file rebin.c.

References CPU_Master, OldNRAD, OldNSEC, OldRadii, OldRmed, and OUTPUTDIR.

Referenced by Initialization().

Here is the caller graph for this function:



4.25.3 Variable Documentation

4.25.3.1 real New_r[MAX1D] [static]

Definition at line 13 of file rebin.c.

Referenced by CheckRebin().

4.25.3.2 int OldNRAD [static]

Definition at line 14 of file rebin.c.

Referenced by `CheckRebin()`, and `ReadPrevDim()`.

4.25.3.3 `int OldNSEC` `[static]`

Definition at line 14 of file `rebin.c`.

Referenced by `CheckRebin()`, and `ReadPrevDim()`.

4.25.3.4 `real OldRadii[MAX1D]` `[static]`

Definition at line 13 of file `rebin.c`.

Referenced by `CheckRebin()`, and `ReadPrevDim()`.

4.25.3.5 `real OldRmed[MAX1D]` `[static]`

Definition at line 13 of file `rebin.c`.

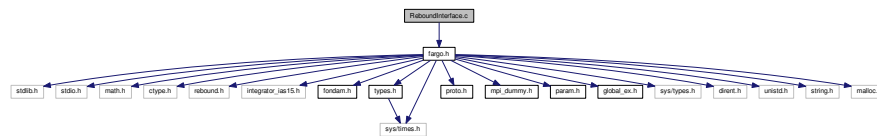
Referenced by `CheckRebin()`, and `ReadPrevDim()`.

4.26 ReboundInterface.c File Reference

Contains the functions interfacing FARGO with the REBOUND package.

```
#include "fargo.h"
```

Include dependency graph for `ReboundInterface.c`:



Functions

- `int ResolveCollisions` (`struct reb_simulation *rsim`, `struct reb_collision coll`)
If the REBOUND collision search is successful, this function merges the bodies, outputs information about the merger event and reorganises the particle list.
- `void DiscardParticlesDist` (`struct reb_simulation *rsim`, `real dt`)
A simple discard routine which looks for planets that were scattered/migrated away from the FARGO grid.
- `void SetupIntegratorParams` (`struct reb_simulation *rsim`)
Fills the rebound simulation structure with parameters inherited from FARGO.
- `struct reb_simulation * SetupReboundSimulation` (`PlanetarySystem *sys`, `char *pfile`)
Initialises a rebound simulation coupled with FARGO.
- `void AdvanceSystemRebound` (`PlanetarySystem *sys`, `struct reb_simulation *rsim`, `real dt`)
Performs an integration step of the N-body problem.
- `void SynchronizeFargoRebound` (`PlanetarySystem *sys`, `struct reb_simulation *rsim`)
Synchronises the planetary system between the REBOUND integration and the FARGO simulation.
- `void MinStepForRebound` (`struct reb_simulation *rsim`)
A simple time step restriction in order not to miss a collision.
- `void OutputElements` (`struct reb_simulation *rsim`)

Calculates and outputs the orbital elements.

- void [OutputNbodySimulation](#) (int nout, struct reb_simulation *rsim)

Stores the entire Rebound simulation in a binary file.

- struct reb_simulation * [RestartReboundSimulation](#) ([PlanetarySystem](#) *sys, int nrestart)

Part of the restart process.

Variables

- real [OmegaFrame](#)
- boolean [Corotating](#)

4.26.1 Detailed Description

Contains the functions interfacing FARGO with the REBOUND package.

Author

Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz

The default setup uses the IAS15 integrator to propagate the planets in time and employs the direct collision search with merging, if turned on. This can be easily reprogrammed if needed.

Note: This version has no test particles, rsim->N could be used directly as the loop limit. But rsim->N_active is used instead so that test particles could be easily implemented. If rsim->N is used, it indicates that a loop would in principle handle test particles as well.

4.26.2 LICENSE

Copyright (c) 2017 Ondřej Chrenko. See the LICENSE file of the distribution.

Definition in file [ReboundInterface.c](#).

4.26.3 Function Documentation

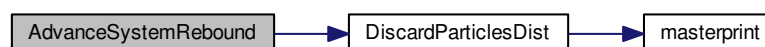
4.26.3.1 void AdvanceSystemRebound ([PlanetarySystem](#) * sys, struct reb_simulation * rsim, real dt)

Performs an integration step of the N-body problem.

Definition at line 197 of file [ReboundInterface.c](#).

References [DiscardParticlesDist\(\)](#), [planetary_system::mass](#), and [PhysicalTime](#).

Here is the call graph for this function:



4.26.3.2 void DiscardParticlesDist (struct reb_simulation * *rsim*, real *dt*)

A simple discard routine which looks for planets that were scattered/migrated away from the FARGO grid.

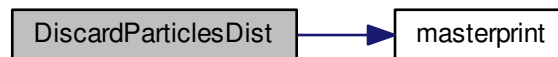
Must be called from the heliocentric frame.

Definition at line 75 of file ReboundInterface.c.

References CPU_Master, discard, masterprint(), NO, PhysicalTime, RMAX, and YES.

Referenced by AdvanceSystemRebound().

Here is the call graph for this function:



Here is the caller graph for this function:



4.26.3.3 void MinStepForRebound (struct reb_simulation * *rsim*)

A simple time step restriction in order not to miss a collision.

This should be in principle always be overridden by the IAS15 time step division.

Definition at line 281 of file ReboundInterface.c.

References invdtreb_sq.

4.26.3.4 void OutputElements (struct reb_simulation * *rsim*)

Calculates and outputs the orbital elements.

Definition at line 309 of file ReboundInterface.c.

References CPU_Number, CPU_Rank, G, PhysicalTime, and plout.

4.26.3.5 void OutputNbodySimulation (int *nout*, struct reb_simulation * *rsim*)

Stores the entire Rebound simulation in a binary file.

Useful for restarts.

Definition at line 328 of file ReboundInterface.c.

References CPU_Master, and OUTPUTDIR.

4.26.3.6 `int ResolveCollisions (struct reb_simulation * rsim, struct reb_collision coll)`

If the REBOUND collision search is successful, this function merges the bodies, outputs information about the merger event and reorganises the particle list.

Definition at line 33 of file ReboundInterface.c.

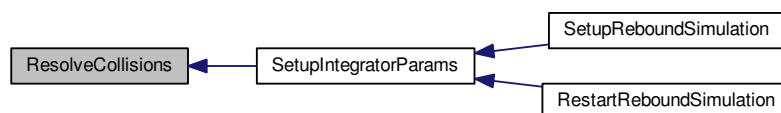
References CPU_Master, masterprint(), and mergers.

Referenced by SetupIntegratorParams().

Here is the call graph for this function:



Here is the caller graph for this function:



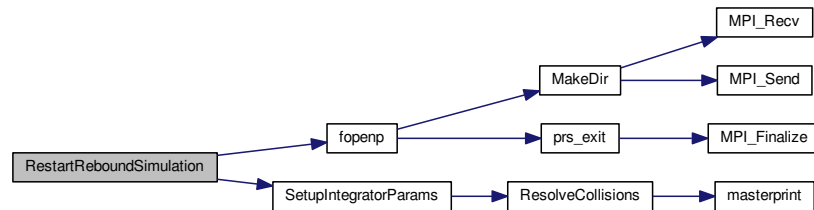
4.26.3.7 `struct reb_simulation* RestartReboundSimulation (PlanetarySystem * sys, int nrestart)`

Part of the restart process.

Definition at line 343 of file ReboundInterface.c.

References ACCRETIONRATE, Collisions, CPU_Master, discard, FeelDisk, fopenp(), mergers, OUTPUTDIR, PhysicalTime, plout, SetupIntegratorParams(), and YES.

Here is the call graph for this function:



4.26.3.8 void SetupIntegratorParams (struct reb_simulation * *rsim*)

Fills the rebound simulation structure with parameters inherited from FARGO.

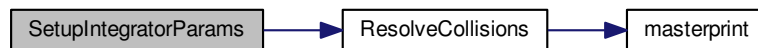
The integrator type can be easily changed from here.

Definition at line 111 of file ReboundInterface.c.

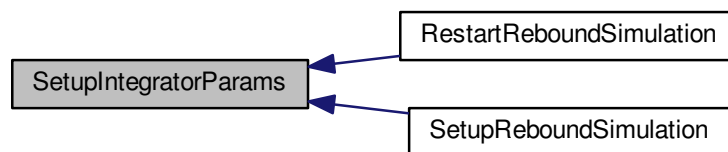
References Collisions, IAS15MINDT, IAS15PRECISION, ResolveCollisions(), and YES.

Referenced by RestartReboundSimulation(), and SetupReboundSimulation().

Here is the call graph for this function:



Here is the caller graph for this function:



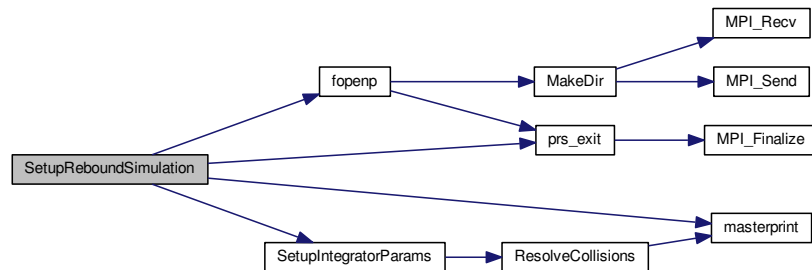
4.26.3.9 struct reb_simulation* SetupReboundSimulation (PlanetarySystem * *sys*, char * *plfile*)

Initialises a rebound simulation coupled with FARGO.

Definition at line 127 of file ReboundInterface.c.

References `a`, `ACCRETIONRATE`, `Collisions`, `discard`, `FeelDisk`, `fopenp()`, `G`, `masterprint()`, `mergers`, `OUTPUTDIR`, `PI`, `PLANETARYDENSITY`, `plout`, `prs_exit()`, `RHO2CGS`, `SetupIntegratorParams()`, and `YES`.

Here is the call graph for this function:



4.26.3.10 void SynchronizeFargoRebound (PlanetarySystem * sys, struct reb_simulation * rsim)

Synchronises the planetary system between the REBOUND integration and the FARGO simulation.

Definition at line 250 of file `ReboundInterface.c`.

References `NO`, `prs_exit()`, and `YES`.

Here is the call graph for this function:



4.26.4 Variable Documentation

4.26.4.1 boolean Corotating

Definition at line 30 of file `Interpret.c`.

4.26.4.2 real OmegaFrame

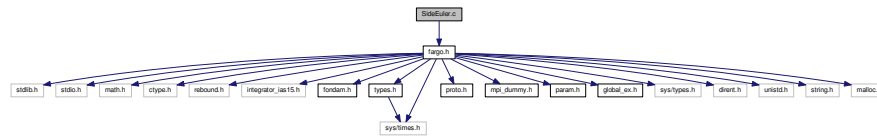
Definition at line 20 of file `global.h`.

4.27 SideEuler.c File Reference

Total mass and angular momentum monitoring, and boundary conditions.

```
#include "fargo.h"
```

Include dependency graph for SideEuler.c:



Functions

- [real GasTotalMass](#) ([PolarGrid](#) *array)
- [real GasMomentum](#) ([PolarGrid](#) *Density, [PolarGrid](#) *Vtheta)
- [void DividePolarGrid](#) ([PolarGrid](#) *Num, [PolarGrid](#) *Denom, [PolarGrid](#) *Res)
- [void InitComputeAccel](#) ()
- [void OpenBoundary](#) ([PolarGrid](#) *Vrad, [PolarGrid](#) *Rho, [PolarGrid](#) *Energy)
- [void NonReflectingBoundary](#) ([PolarGrid](#) *Vrad, [PolarGrid](#) *Rho, [PolarGrid](#) *Energy)
- [void SetWaveKillingZones](#) ()
Sets the wave-killing factors within the damping zones; inspired by de Val-Borro et al.
- [void DampingBoundary](#) ([PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta, [PolarGrid](#) *Rho, [PolarGrid](#) *Energy, [real](#) step)
Imposes the wave-killing boundary condition.
- [void DampPebbles](#) ([PolarGrid](#) *PebbleDens, [PolarGrid](#) *PebbleVrad, [PolarGrid](#) *PebbleVtheta, [real](#) dt)
Damps the pebble disk inside the wave-killing zones towards its equilibrium state.
- [void ApplyOuterSourceMass](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Vrad)
- [void ApplyBoundaryCondition](#) ([PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta, [PolarGrid](#) *Rho, [PolarGrid](#) *Energy, [real](#) dt)
- [void CorrectVtheta](#) ([PolarGrid](#) *vtheta, [real](#) domega)
- [real GasTotalEnergy](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta, [PolarGrid](#) *Energy)

Variables

- [boolean OpenInner](#)
- [boolean NonReflecting](#)
- [boolean OuterSourceMass](#)

4.27.1 Detailed Description

Total mass and angular momentum monitoring, and boundary conditions.

In addition, this file contains a few low-level functions that manipulate [PolarGrid](#) 's or initialize the forces evaluation.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [SideEuler.c](#).

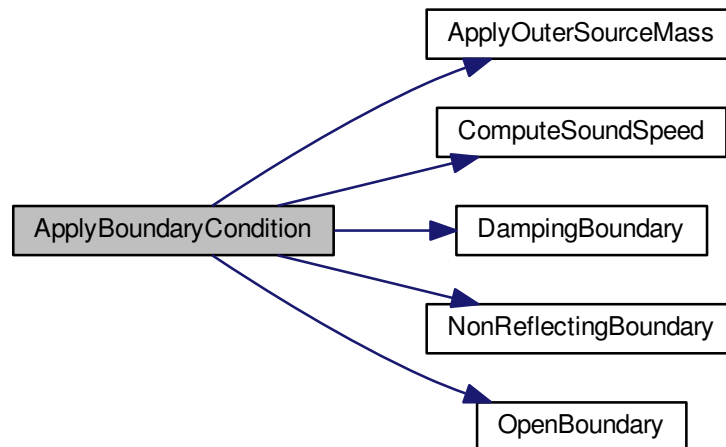
4.27.2 Function Documentation

4.27.2.1 `void ApplyBoundaryCondition (PolarGrid * Vrad, PolarGrid * Vtheta, PolarGrid * Rho, PolarGrid * Energy, real dt)`

Definition at line 384 of file SideEuler.c.

References `ApplyOuterSourceMass()`, `ComputeSoundSpeed()`, `Damping`, `DampingBoundary()`, `EnergyEq`, `NonReflecting`, `NonReflectingBoundary()`, `OpenBoundary()`, `OpenInner`, `OuterSourceMass`, and `YES`.

Here is the call graph for this function:



4.27.2.2 `void ApplyOuterSourceMass (PolarGrid * Rho, PolarGrid * Vrad)`

Definition at line 354 of file SideEuler.c.

References `CPU_Number`, `CPU_Rank`, `IMPOSEDDISKDRIFT`, `polargrid::Nrad`, `Rinf`, `SigmaMed`, and `SIGMASL`.

Referenced by `ApplyBoundaryCondition()`.

Here is the caller graph for this function:



4.27.2.3 void CorrectVtheta (PolarGrid * *vtheta*, real *omega*)

Definition at line 398 of file SideEuler.c.

References Rmed.

4.27.2.4 void DampingBoundary (PolarGrid * *Vrad*, PolarGrid * *Vtheta*, PolarGrid * *Rho*, PolarGrid * *Energy*, real *step*)

Imposes the wave-killing boundary condition.

Currently, the condition is set to always damp the radial velocity to zero. Additionally, the density, azimuthal velocity and energy can be damped to their initial values.

Definition at line 270 of file SideEuler.c.

References DAMPINGRMAXFRAC, DAMPINGRMINFRAC, DampInIt, EnergyMed, Max_or_active, OmegaFrame, RMAX, Rmed, RMIN, SigmaMed, VthetaMed, WaveKiller, and Zero_or_active.

Referenced by ApplyBoundaryCondition().

Here is the caller graph for this function:



4.27.2.5 void DampPebbles (PolarGrid * *PebbleDens*, PolarGrid * *PebbleVrad*, PolarGrid * *PebbleVtheta*, real *dt*)

Damps the pebble disk inside the wave-killing zones towards its equilibrium state.

Definition at line 317 of file SideEuler.c.

References DAMPINGRMAXFRAC, DAMPINGRMINFRAC, Max_or_active, polargrid::Nsec, OmegaFrame, PebDensInit, PebVradInit, PebVthetaInit, RMAX, Rmed, RMIN, WaveKiller, and Zero_or_active.

4.27.2.6 void DivisePolarGrid (PolarGrid * *Num*, PolarGrid * *Denom*, PolarGrid * *Res*)

Definition at line 74 of file SideEuler.c.

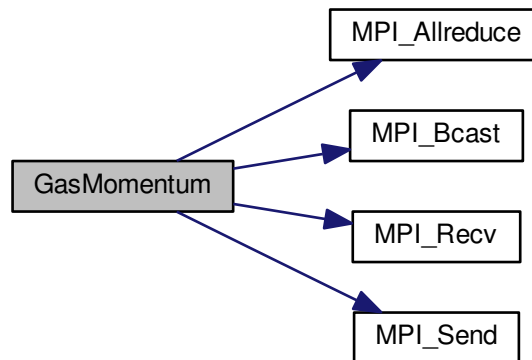
References polargrid::Field.

4.27.2.7 real GasMomentum (PolarGrid * *Density*, PolarGrid * *Vtheta*)

Definition at line 45 of file SideEuler.c.

References CPU_Number, CPU_Rank, FakeSequential, fargostat, Max_or_active, MPI_Allreduce(), MPI_Bcast(), MPI_COMM_WORLD, MPI_DOUBLE, MPI_Recv(), MPI_Send(), MPI_SUM, polargrid::Nsec, OmegaFrame, Rmed, Surf, and Zero_or_active.

Here is the call graph for this function:

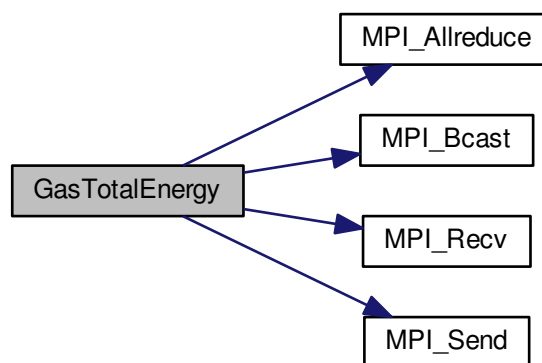


4.27.2.8 `real GasTotalEnergy (PolarGrid * Rho, PolarGrid * Vrad, PolarGrid * Vtheta, PolarGrid * Energy)`

Definition at line 416 of file SideEuler.c.

References CPU_Number, CPU_Rank, FakeSequential, fargostat, Max_or_active, MPI_Allreduce(), MPI_Bcast(), MPI_COMM_WORLD, MPI_DOUBLE, MPI_Recv(), MPI_Send(), MPI_SUM, OmegaFrame, Rinf, Rmed, Rsup, Surf, vt_cent, and Zero_or_active.

Here is the call graph for this function:

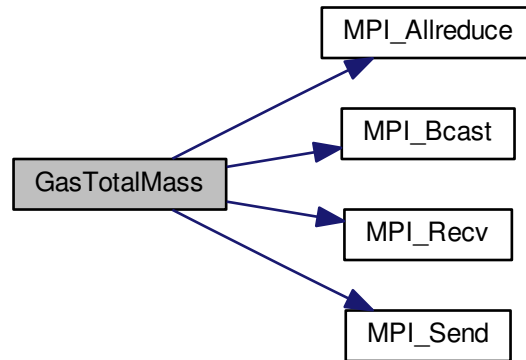


4.27.2.9 `real GasTotalMass (PolarGrid * array)`

Definition at line 18 of file SideEuler.c.

References CPU_Number, CPU_Rank, FakeSequential, fargostat, Max_or_active, MPI_Allreduce(), MPI_Bcast(), MPI_COMM_WORLD, MPI_DOUBLE, MPI_Recv(), MPI_Send(), MPI_SUM, polargrid::Nsec, Surf, and Zero_or_active.

Here is the call graph for this function:



4.27.2.10 void InitComputeAccel ()

Definition at line 93 of file SideEuler.c.

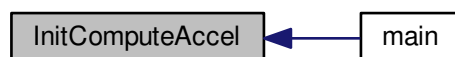
References CellAbscissa, CellOrdinate, CreatePolarGrid(), polargrid::Field, NRAD, polargrid::Nrad, NSEC, polargrid::Nsec, PI, and Rmed.

Referenced by main().

Here is the call graph for this function:



Here is the caller graph for this function:



4.27.2.11 void NonReflectingBoundary (PolarGrid * Vrad, PolarGrid * Rho, PolarGrid * Energy)

Definition at line 135 of file SideEuler.c.

References CPU_Number, CPU_Rank, EnergyMed, polargrid::Field, NSEC, PI, Rinf, Rmed, SigmaMed, and SoundSpeed.

Referenced by ApplyBoundaryCondition().

Here is the caller graph for this function:



4.27.2.12 void OpenBoundary (PolarGrid * Vrad, PolarGrid * Rho, PolarGrid * Energy)

Definition at line 112 of file SideEuler.c.

References CPU_Rank, and SigmaMed.

Referenced by ApplyBoundaryCondition().

Here is the caller graph for this function:



4.27.2.13 void SetWaveKillingZones ()

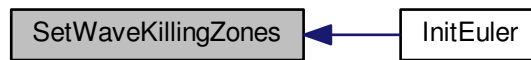
Sets the wave-killing factors within the damping zones; inspired by de Val-Borro et al. (2006).

Definition at line 242 of file SideEuler.c.

References DAMPINGPERIODFRAC, DAMPINGRMAXFRAC, DAMPINGRMINFRAC, Max_or_active, PI, RMAX, Rmed, RMIN, WaveKiller, and Zero_or_active.

Referenced by InitEuler().

Here is the caller graph for this function:



4.27.3 Variable Documentation

4.27.3.1 `boolean` NonReflecting

Definition at line 30 of file `Interpret.c`.

Referenced by `ApplyBoundaryCondition()`, and `ReadVariables()`.

4.27.3.2 `boolean` OpenInner

Definition at line 14 of file `main.c`.

Referenced by `ApplyBoundaryCondition()`.

4.27.3.3 `boolean` OuterSourceMass

Definition at line 30 of file `Interpret.c`.

Referenced by `ApplyBoundaryCondition()`, and `ReadVariables()`.

4.28 SourceEuler.c File Reference

Contains routines used by the hydrodynamical loop.

```
#include "fargo.h"
```

Include dependency graph for `SourceEuler.c`:



Macros

- `#define CFLSECURITY 0.5 /* Maximum fraction of zone size */`
- `#define CVNR 1.41 /* Shocks are spread over CVNR zones: */`

Functions

- `boolean DetectCrash (PolarGrid *array)`

- void [FillPolar1DArrays](#) ()
- void [InitEuler](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Vr, [PolarGrid](#) *Vt, [PolarGrid](#) *En)
- [real](#) [min2](#) ([real](#) a, [real](#) b)
- [real](#) [max2](#) ([real](#) a, [real](#) b)
- void [ActualiseGas](#) ([PolarGrid](#) *array, [PolarGrid](#) *newarray)
- void [AlgoGas](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta, [PolarGrid](#) *Energy, [PolarGrid](#) *Label, [PlanetarySystem](#) *sys, struct reb_simulation *rsim)
- void [SubStep1](#) ([PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta, [PolarGrid](#) *Rho, [real](#) dt)
- void [SubStep2](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Energy, [real](#) dt)
- void [SubStep3](#) ([PolarGrid](#) *Rho, [real](#) dt)
 - Numerical step responsible for the energy update.*
- int [ConditionCFL](#) ([PolarGrid](#) *Vrad, [PolarGrid](#) *Vtheta, [real](#) deltaT)
- void [ComputeSoundSpeed](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Energy)
 - Updates the sound speed over the mesh.*
- void [ComputePressureField](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Energy)
 - Updates the pressure over the mesh.*
- void [ComputeTemperatureField](#) ([PolarGrid](#) *Rho, [PolarGrid](#) *Energy)
 - Updates the temperature over the mesh.*

Variables

- static [PolarGrid](#) * [TemperInt](#)
- static [PolarGrid](#) * [VradNew](#)
- static [PolarGrid](#) * [VradInt](#)
- static [PolarGrid](#) * [VthetaNew](#)
- static [PolarGrid](#) * [VthetaInt](#)
- static [real](#) [timeCRASH](#)
- [boolean](#) [Corotating](#)
- static [PolarGrid](#) * [EnergyNew](#)
- static [PolarGrid](#) * [EnergyInt](#)
- static int [AlreadyCrashed](#) = 0
- static int [GasTimeStepsCFL](#)
- int [TimeStep](#)
- [boolean](#) [FastTransport](#)
- [boolean](#) [IsDisk](#)

4.28.1 Detailed Description

Contains routines used by the hydrodynamical loop.

More specifically, it contains the main loop itself and all the source term substeps (with the exception of the evaluation of the viscous force). The transport substep is treated elsewhere.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [SourceEuler.c](#).

4.28.2 Macro Definition Documentation

4.28.2.1 `#define CFLSECURITY 0.5 /* Maximum fraction of zone size */`

Definition at line 16 of file SourceEuler.c.

Referenced by ConditionCFL().

4.28.2.2 `#define CVNR 1.41 /* Shocks are spread over CVNR zones: */`

Definition at line 19 of file SourceEuler.c.

Referenced by ConditionCFL(), and SubStep2().

4.28.3 Function Documentation

4.28.3.1 `void ActualiseGas (PolarGrid * array, PolarGrid * newarray)`

Definition at line 181 of file SourceEuler.c.

References polargrid::Nrad.

Referenced by AlgoGas().

Here is the caller graph for this function:



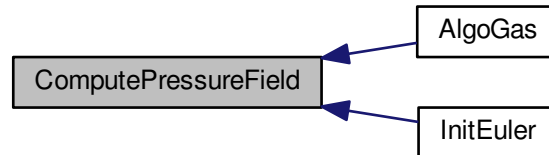
4.28.3.2 `void AlgoGas (PolarGrid * Rho, PolarGrid * Vrad, PolarGrid * Vtheta, PolarGrid * Energy, PolarGrid * Label, PlanetarySystem * sys, struct reb_simulation * rsim)`

Definition at line 199 of file SourceEuler.c.

References AccreteOntoPlanets(), AccretePebblesOntoPlanets(), ActualiseGas(), AdvanceSystemFromDisk(), AdvanceSystemRebound(), AlreadyCrashed, ApplyBoundaryCondition(), CommunicateBoundaries(), ComputePressureField(), ComputeSoundSpeed(), ComputeTemperatureField(), ConditionCFL(), Corotating, CorrectPebblesVtheta(), CorrectVtheta(), CriticalCharTime(), DetectCrash(), DetectCrashPebbles(), DiffusiveParticles, domega, DT, EnergyEq, EvolvePebbleDisk(), FillForcesArrays(), GasTimeStepsCFL, GET, GetPsysInfo(), GetPsysInfoFromRsim(), IsDisk, MARK, MassTaper, MASSTAPER, masterprint(), MinStepForRebound(), MPI_Allreduce(), MPI_COMM_WORLD, MPI_INT, MPI_MAX, NO, OmegaFrame, ParametricAccretion(), ParticleDiffusion(), Pebbles, PebbleStokesNumbers(), PhysicalTime, PrescribedAccretion, RotatePsys(), SloppyCFL, SourceTermsPebbles(), SubStep1(), SubStep1Pebbles(), SubStep2(), SubStep3(), SynchronizeFargoRebound(), SynchronizePebbleDisc(), Temperature, timeCRASH, Transport(), UpdateDivVelocAndStressTensor(), WriteDiskPolar(), and YES.

Referenced by AlgoGas(), and InitEuler().

Here is the caller graph for this function:



4.28.3.4 void ComputeSoundSpeed (PolarGrid * *Rho*, PolarGrid * *Energy*)

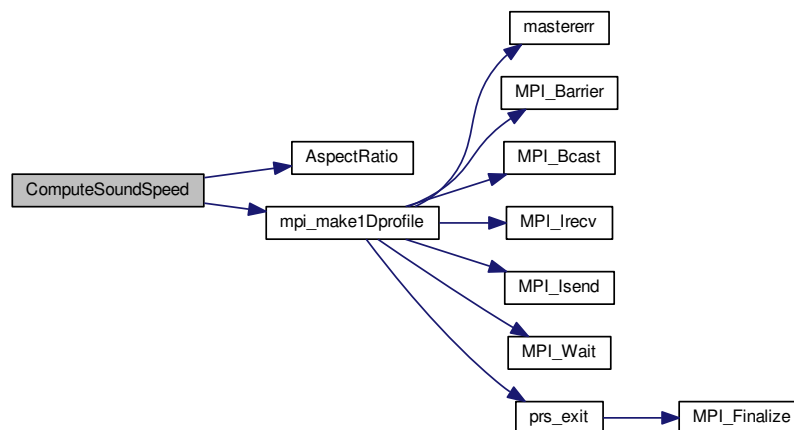
Updates the sound speed over the mesh.

Definition at line 683 of file SourceEuler.c.

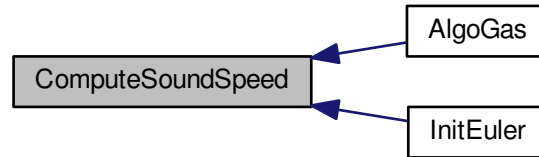
References ADIABIND, AspectRatio(), EnergyEq, polargrid::Field, FLARINGINDEX, G, globcsvec, mpi_make1Dprofile(), Rmed, and SoundSpeed.

Referenced by AlgoGas(), and InitEuler().

Here is the call graph for this function:



Here is the caller graph for this function:



4.28.3.5 void ComputeTemperatureField (PolarGrid * *Rho*, PolarGrid * *Energy*)

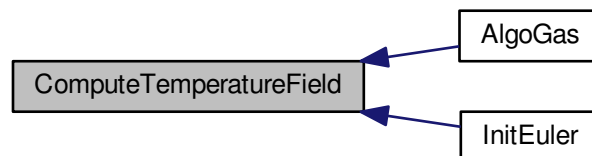
Updates the temperature over the mesh.

Definition at line 733 of file `SourceEuler.c`.

References `ADIABIND`, `EnergyEq`, `polargrid::Field`, `GASCONST`, `MOLWEIGHT`, `Pressure`, and `Temperature`.

Referenced by `AlgoGas()`, and `InitEuler()`.

Here is the caller graph for this function:



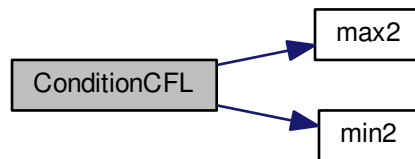
4.28.3.6 int ConditionCFL (PolarGrid * *Vrad*, PolarGrid * *Vtheta*, real *deltaT*)

Definition at line 574 of file `SourceEuler.c`.

References `CFLSECURITY`, `CPU_Rank`, `CVNR`, `debug`, `FastTransport`, `polargrid::Field`, `invdtpeb_sq`, `invdtreb_sq`, `InvRmed`, `MAX1D`, `max2()`, `Max_or_active`, `MaxMO_or_active`, `min2()`, `NSEC`, `polargrid::Nsec`, `One_or_active`, `Pebbles`, `PI`, `Rinf`, `Rmed`, `Rsup`, `SoundSpeed`, `YES`, and `Zero_or_active`.

Referenced by `AlgoGas()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.28.3.7 **boolean** DetectCrash (**PolarGrid** * *array*)

Definition at line 37 of file SourceEuler.c.

References NO, and YES.

Referenced by AlgoGas().

Here is the caller graph for this function:



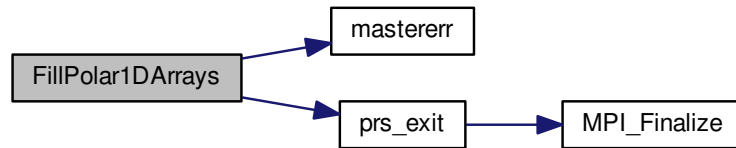
4.28.3.8 **void** FillPolar1DArrays ()

Definition at line 57 of file SourceEuler.c.

References CPU_Master, GLOBALNRAD, GlobalRmed, IMIN, InvDiffRmed, InvDiffRsup, InvRinf, InvRmed, InvSurf, LogGrid, mastererr(), NRAD, NSEC, OmegaInv, OUTPUTDIR, PI, prs_exit(), Radii, Rinf, RMAX, Rmed, Rmed2, R←MIN, Rsup, Surf, and YES.

Referenced by InitEuler().

Here is the call graph for this function:



Here is the caller graph for this function:

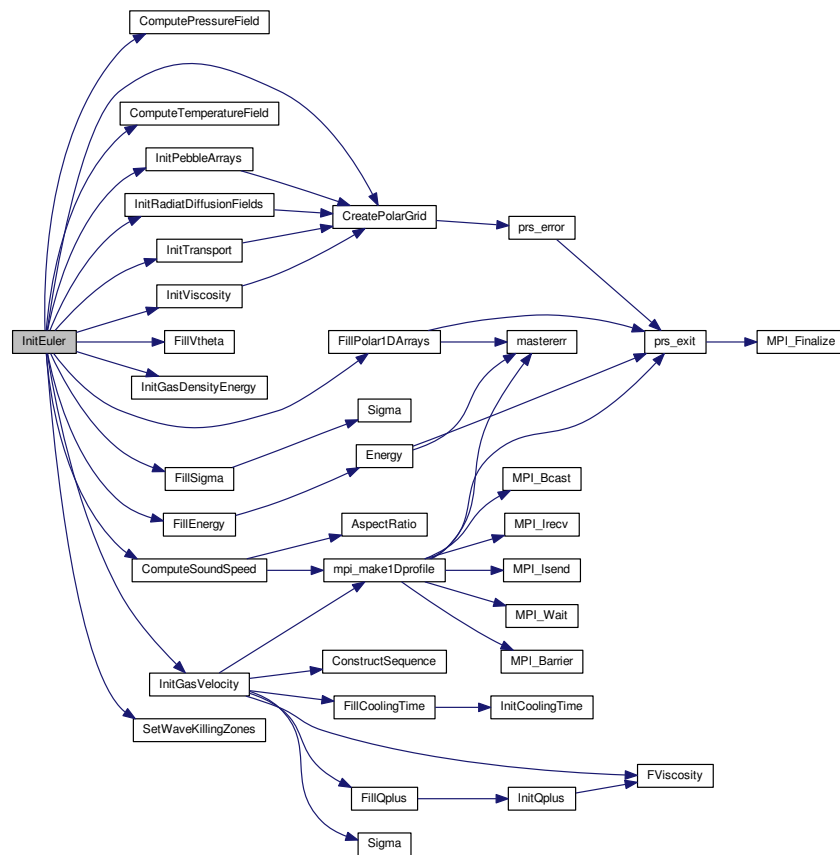


4.28.3.9 void InitEuler (PolarGrid * Rho, PolarGrid * Vr, PolarGrid * Vt, PolarGrid * En)

Definition at line 122 of file SourceEuler.c.

References ADIABIND, ComputePressureField(), ComputeSoundSpeed(), ComputeTemperatureField(), CreatePolarGrid(), Damping, DampInit, EnergyEq, FillEnergy(), FillPolar1DArrays(), FillSigma(), FillVtheta(), GravAccelRad, GravAccelTheta, InitGasDensityEnergy(), InitGasVelocity(), InitPebbleArrays(), InitRadiatorDiffusionFields(), InitTransport(), InitViscosity(), NRAD, NSEC, ParametricCooling, PebbleGravAccelRad, PebbleGravAccelTheta, Pebbles, Pressure, Qplus, RhoInt, RhoStar, SetWaveKillingZones(), SoundSpeed, SQRT_ADIABIND_INV, Temperature, Torque, and TorqueDensity.

Here is the call graph for this function:



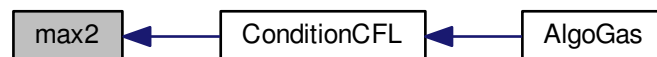
4.28.3.10 `real max2 (real a, real b)`

Definition at line 173 of file SourceEuler.c.

References `a`, and `b`.

Referenced by `ConditionCFL()`.

Here is the caller graph for this function:



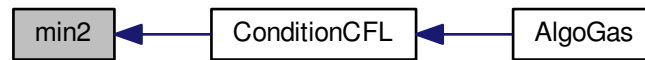
4.28.3.11 `real min2 (real a, real b)`

Definition at line 166 of file SourceEuler.c.

References a, and b.

Referenced by ConditionCFL().

Here is the caller graph for this function:



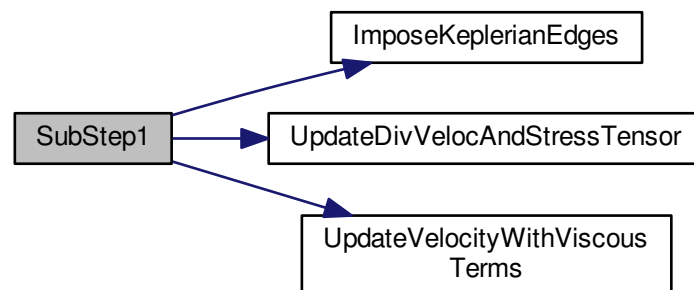
4.28.3.12 void SubStep1 (PolarGrid * Vrad, PolarGrid * Vtheta, PolarGrid * Rho, real dt)

Definition at line 304 of file SourceEuler.c.

References BackReaction, Damping, DragForceRad, DragForceTheta, polargrid::Field, GasAccelrad, GasAcceltheta, GravAccelRad, GravAccelTheta, IMPOSEDDISKDRIFT, ImposeKeplerianEdges(), InvDiffRmed, InvRinf, polargrid::Nrad, OmegaFrame, Pebbles, PI, Pressure, Rinf, Rmed, SIGMASLOPE, UpdateDivVelocAndStressTensor(), and UpdateVelocityWithViscousTerms().

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.28.3.13 void SubStep2 (PolarGrid * *Rho*, PolarGrid * *Energy*, real *dt*)

Definition at line 402 of file SourceEuler.c.

References CVNR, EnergyEq, polargrid::Field, InvDiffRmed, InvDiffRsup, polargrid::Nrad, polargrid::Nsec, $P \leftrightarrow I$, RhoInt, and Rmed.

Referenced by AlgoGas().

Here is the caller graph for this function:



4.28.3.14 void SubStep3 (PolarGrid * *Rho*, real *dt*)

Numerical step responsible for the energy update.

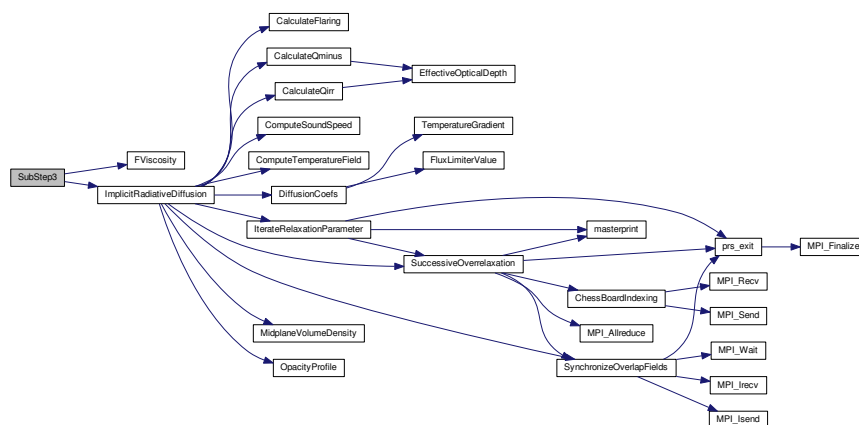
Calls the energy equation solver.

Definition at line 493 of file SourceEuler.c.

References ADIABIND, CoolingTimeMed, DivergenceVelocity, EnergyMed, polargrid::Field, FViscosity(), ImplicitRadiativeDiffusion(), ParametricCooling, Qplus, QplusMed, Rmed, SigmaMed, TAUPP, TAURP, and TAURR.

Referenced by AlgoGas().

Here is the call graph for this function:



Here is the caller graph for this function:



4.28.4 Variable Documentation

4.28.4.1 `int AlreadyCrashed = 0` `[static]`

Definition at line 31 of file SourceEuler.c.

Referenced by AlgoGas().

4.28.4.2 `boolean Corotating`

Definition at line 30 of file Interpret.c.

Referenced by AlgoGas(), and ReadVariables().

4.28.4.3 `PolarGrid * EnergyInt` `[static]`

Definition at line 29 of file SourceEuler.c.

4.28.4.4 `PolarGrid * EnergyNew` `[static]`

Definition at line 29 of file SourceEuler.c.

4.28.4.5 `boolean FastTransport`

Definition at line 29 of file Interpret.c.

Referenced by ConditionCFL().

4.28.4.6 `int GasTimeStepsCFL` `[static]`

Definition at line 31 of file SourceEuler.c.

Referenced by AlgoGas().

4.28.4.7 `boolean IsDisk`

Definition at line 30 of file Interpret.c.

Referenced by AlgoGas(), and ReadVariables().

4.28.4.8 `PolarGrid * TemperInt` `[static]`

Definition at line 23 of file SourceEuler.c.

4.28.4.9 **real timeCRASH** [static]

Definition at line 26 of file SourceEuler.c.

Referenced by AlgoGas().

4.28.4.10 **int TimeStep**

Definition at line 23 of file global.h.

4.28.4.11 **PolarGrid * VradInt** [static]

Definition at line 24 of file SourceEuler.c.

4.28.4.12 **PolarGrid* VradNew** [static]

Definition at line 24 of file SourceEuler.c.

4.28.4.13 **PolarGrid * VthetaInt** [static]

Definition at line 25 of file SourceEuler.c.

4.28.4.14 **PolarGrid* VthetaNew** [static]

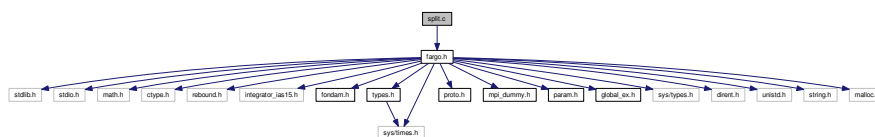
Definition at line 25 of file SourceEuler.c.

4.29 split.c File Reference

Split (radially) the mesh among the different processors.

```
#include "fargo.h"
```

Include dependency graph for split.c:



Functions

- void [SplitDomain](#) ()

4.29.1 Detailed Description

Split (radially) the mesh among the different processors.

A simple Round Robin algorithm is used to achieve a proper load balancing, assuming the cluster to be homogeneous. A ring described by a given process has its 5 (CPUOVERLAP) innermost zones that are ghost zones which are filled by communications with the previous inner process (unless it is the innermost process itself), and its 5 (CPUOVERLAP) outermost zones that are ghost zones which are filled by communications with the next outer

process (unless it is the outermost process itself). The "active" part of the submesh described by a given process is the part of this mesh that excludes the ghost zones. For each process, the (local) radial index of the first active ring is `Zero_or_active`, and the (local) radial index of the last active ring is `Max_or_active`. `MaxMO_or_active` is `Max_or_active` for all processes, except the very last one (the outermost) for which it is `Max_or_active-1` (MO stands for 'Minus One').

Definition in file [split.c](#).

4.29.2 Function Documentation

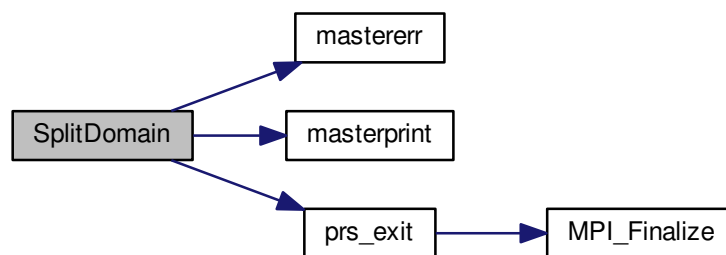
4.29.2.1 void SplitDomain ()

Definition at line 24 of file `split.c`.

References `CPU_Number`, `CPU_Rank`, `CPUOVERLAP`, `debug`, `GLOBALNRAD`, `IMAX`, `IMIN`, `mastererr()`, `masterprint()`, `Max_or_active`, `MaxMO_or_active`, `NRAD`, `One_or_active`, `prs_exit()`, `YES`, and `Zero_or_active`.

Referenced by `main()`.

Here is the call graph for this function:



Here is the caller graph for this function:

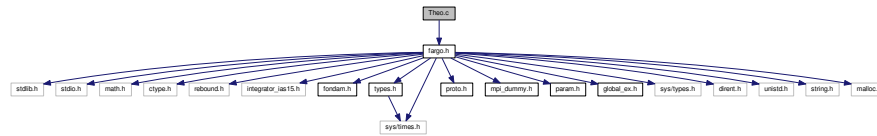


4.30 Theo.c File Reference

A few functions that manipulate the surface density profile.

```
#include "fargo.h"
```

Include dependency graph for Theo.c:



Functions

- [real Sigma](#) ([real](#) r)
- void [FillSigma](#) ()
- void [RefillSigma](#) ([PolarGrid](#) *Surfdens)
- [real Energy](#) ([real](#) r)
Initialises the energy array.
- void [FillEnergy](#) ()
- void [RefillEnergy](#) ([PolarGrid](#) *Energy)
- [real InitCoolingTime](#) ([real](#) r)
- void [FillCoolingTime](#) ()
- [real InitQplus](#) ([real](#) r)
- void [FillQplus](#) ()
- void [FillVtheta](#) ([PolarGrid](#) *Vtheta)

Variables

- [real ScalingFactor](#)

4.30.1 Detailed Description

A few functions that manipulate the surface density profile.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [Theo.c](#).

4.30.2 Function Documentation

4.30.2.1 [real Energy](#) ([real](#) r)

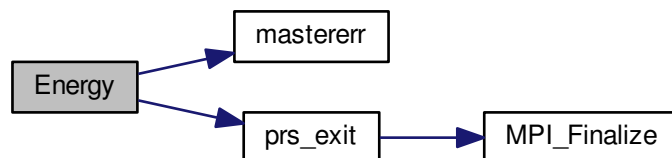
Initialises the energy array.

Definition at line 60 of file Theo.c.

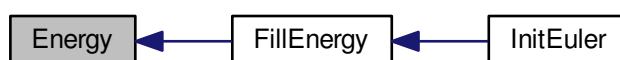
References [ADIABIND](#), [ASPECTRATIO](#), [FLARINGINDEX](#), [GASCONST](#), [mastererr\(\)](#), [MOLWEIGHT](#), [prs_exit\(\)](#), [S←IGMA0](#), and [SIGMASLOPE](#).

Referenced by [FillEnergy\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:



4.30.2.2 void FillCoolingTime ()

Definition at line 110 of file Theo.c.

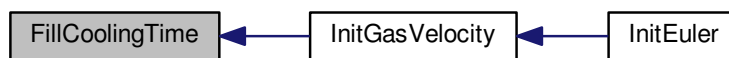
References `CoolingTimeMed`, `InitCoolingTime()`, `NRAD`, and `Rmed`.

Referenced by `InitGasVelocity()`.

Here is the call graph for this function:



Here is the caller graph for this function:



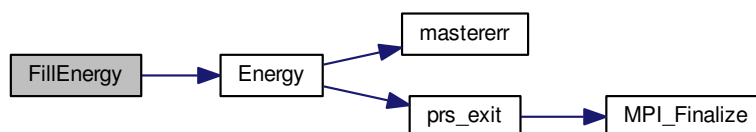
4.30.2.3 void FillEnergy ()

Definition at line 76 of file Theo.c.

References `Energy()`, `EnergyMed`, `NRAD`, and `Rmed`.

Referenced by `InitEuler()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.30.2.4 void FillQplus ()

Definition at line 125 of file Theo.c.

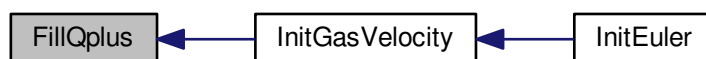
References `InitQplus()`, `NRAD`, `QplusMed`, and `Rmed`.

Referenced by `InitGasVelocity()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.30.2.5 void FillSigma ()

Definition at line 25 of file Theo.c.

References `NRAD`, `Rinf`, `Rmed`, `Sigma()`, `SigmaInf`, and `SigmaMed`.

Referenced by `InitEuler()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.30.2.6 void FillVtheta (PolarGrid * Vtheta)

Definition at line 132 of file Theo.c.

References polargrid::Nrad, OmegaFrame, Rmed, and VthetaMed.

4.30.2.7 real InitCoolingTime (real r)

Definition at line 102 of file Theo.c.

References COOLINGTIME, and FLARINGINDEX.

Referenced by FillCoolingTime().

Here is the caller graph for this function:



4.30.2.8 real InitQplus (real r)

Definition at line 116 of file Theo.c.

References FViscosity(), Qplus, SIGMA0, and SIGMASLOPE.

Referenced by FillQplus().

Here is the call graph for this function:



Here is the caller graph for this function:



4.30.2.9 void RefillEnergy (PolarGrid * Energy)

Definition at line 82 of file Theo.c.

- void `ComputeResiduals` (`PolarGrid *Vtheta`, `real dt`)
- void `AdvectSHIFT` (`PolarGrid *array`)
- void `OneWindTheta` (`PolarGrid *Rho`, `PolarGrid *Vtheta`, `PolarGrid *Energy`, `real dt`)
- void `QuantitiesAdvection` (`PolarGrid *Rho`, `PolarGrid *Energy`, `PolarGrid *Vtheta`, `real dt`)
- void `ComputeExtQty` (`PolarGrid *Rho`, `PolarGrid *Label`, `PolarGrid *ExtLabel`)
- void `ComputeSpeQty` (`PolarGrid *Rho`, `PolarGrid *Label`, `PolarGrid *ExtLabel`)
- void `InitTransport` ()
- void `ComputeStarRad` (`PolarGrid *Qbase`, `PolarGrid *Vrad`, `PolarGrid *QStar`, `real dt`)
- void `ComputeStarTheta` (`PolarGrid *Qbase`, `PolarGrid *Vtheta`, `PolarGrid *QStar`, `real dt`)
- void `ComputeLRMomenta` (`PolarGrid *Rho`, `PolarGrid *Vrad`, `PolarGrid *Vtheta`)
- void `ComputeVelocities` (`PolarGrid *Rho`, `PolarGrid *Vrad`, `PolarGrid *Vtheta`)
- `real VanLeerRadial` (`PolarGrid *Vrad`, `PolarGrid *Qbase`, `real dt`)
- void `VanLeerTheta` (`PolarGrid *Vtheta`, `PolarGrid *Qbase`, `real dt`)
- void `TransportPebbles` (`PolarGrid *Rho`, `PolarGrid *Vrad`, `PolarGrid *Vtheta`, `real dt`)
- *An alternative to `Transport()` function for the 2nd fluid.*
- void `OneWindRadPebbles` (`PolarGrid *Rho`, `PolarGrid *Vrad`, `real dt`)
- *An alternative to `OneWindRad()` function for the 2nd fluid.*
- void `OneWindThetaPebbles` (`PolarGrid *Rho`, `PolarGrid *Vtheta`, `real dt`)
- *An alternative to `OneWindTheta()` function for the 2nd fluid.*
- void `QuantitiesAdvectionPebbles` (`PolarGrid *Rho`, `PolarGrid *Vtheta`, `real dt`)
- *An alternative to `QuantitiesAdvection()` function for the 2nd fluid.*

Variables

- `real OmegaFrame`
- static `real VMed` [`MAX1D`]
- static `int Nshift` [`MAX1D`]
- static `real * TempShift`
- static `real * dq`
- static `PolarGrid * RadMomP`
- static `PolarGrid * RadMomM`
- static `PolarGrid * ThetaMomP`
- static `PolarGrid * ThetaMomM`
- static `PolarGrid * ExtLabel`
- static `PolarGrid * VthetaRes`
- static `PolarGrid * Work`
- static `PolarGrid * QRStar`
- static `PolarGrid * Elongations`
- `int TimeStep`
- `boolean OpenInner`
- `boolean FastTransport`
- `real LostMass` = 0.0

4.31.1 Detailed Description

Functions that handle the transport substep of a hydrodynamical time step.

The FARGO algorithm is implemented here. The transport is performed in a manner similar to what is done for the ZEUS code (Stone & Norman, 1992), except for the momenta transport (we define a left and right momentum for each zone, which we declare zone centered; we then transport then normally, and deduce the new velocity in each zone by a proper averaging).

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [TransportEuler.c](#).

4.31.2 Function Documentation

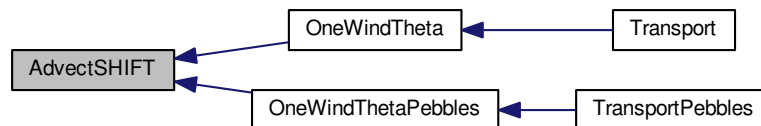
4.31.2.1 void AdvectSHIFT (PolarGrid * array)

Definition at line 113 of file TransportEuler.c.

References `polargrid::Field`, `Nshift`, and `TempShift`.

Referenced by `OneWindTheta()`, and `OneWindThetaPebbles()`.

Here is the caller graph for this function:



4.31.2.2 void ComputeExtQty (PolarGrid * Rho, PolarGrid * Label, PolarGrid * ExtLabel)

Definition at line 177 of file TransportEuler.c.

References `polargrid::Nrad`.

Referenced by `Transport()`.

Here is the caller graph for this function:



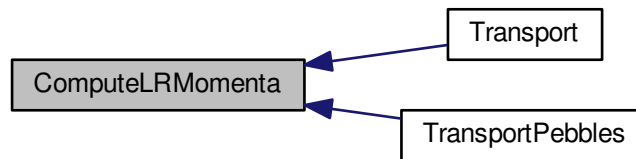
4.31.2.3 void ComputeLRMomenta (PolarGrid * Rho, PolarGrid * Vrad, PolarGrid * Vtheta)

Definition at line 318 of file TransportEuler.c.

References `polargrid::Field`, `polargrid::Nrad`, `OmegaFrame`, and `Rmed`.

Referenced by `Transport()`, and `TransportPebbles()`.

Here is the caller graph for this function:



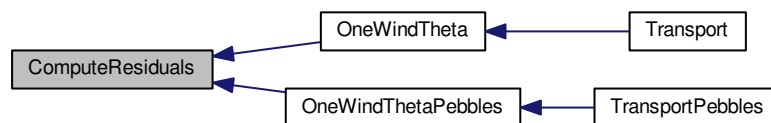
4.31.2.4 void ComputeResiduals (PolarGrid * *Vtheta*, real *dt*)

Definition at line 70 of file TransportEuler.c.

References `FastTransport`, `polargrid::Field`, `InvRmed`, `polargrid::Nrad`, `Nshift`, `PI`, `Rmed`, `VMed`, and `YES`.

Referenced by `OneWindTheta()`, and `OneWindThetaPebbles()`.

Here is the caller graph for this function:



4.31.2.5 void ComputeSpeQty (PolarGrid * *Rho*, PolarGrid * *Label*, PolarGrid * *ExtLabel*)

Definition at line 197 of file TransportEuler.c.

References `polargrid::Nrad`.

Referenced by `Transport()`.

Here is the caller graph for this function:



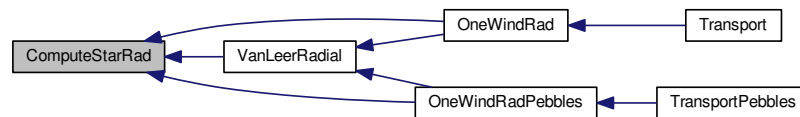
4.31.2.6 void ComputeStarRad (PolarGrid * Qbase, PolarGrid * Vrad, PolarGrid * QStar, real dt)

Definition at line 232 of file TransportEuler.c.

References dq, InvDiffRmed, polargrid::Nrad, and Rmed.

Referenced by OneWindRad(), OneWindRadPebbles(), and VanLeerRadial().

Here is the caller graph for this function:



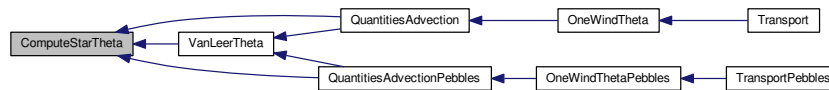
4.31.2.7 void ComputeStarTheta (PolarGrid * Qbase, PolarGrid * Vtheta, PolarGrid * QStar, real dt)

Definition at line 275 of file TransportEuler.c.

References dq, polargrid::Nrad, PI, and Rmed.

Referenced by QuantitiesAdvection(), QuantitiesAdvectionPebbles(), and VanLeerTheta().

Here is the caller graph for this function:



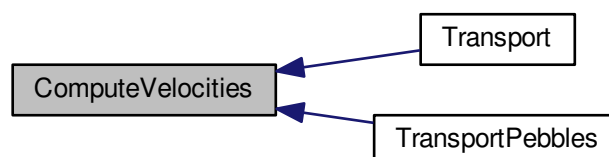
4.31.2.8 void ComputeVelocities (PolarGrid * Rho, PolarGrid * Vrad, PolarGrid * Vtheta)

Definition at line 350 of file TransportEuler.c.

References polargrid::Field, polargrid::Nrad, OmegaFrame, and Rmed.

Referenced by Transport(), and TransportPebbles().

Here is the caller graph for this function:



4.31.2.9 void InitTransport ()

Definition at line 217 of file TransportEuler.c.

References CreatePolarGrid(), dq, NRAD, NSEC, and TempShift.

Referenced by InitEuler().

Here is the call graph for this function:



Here is the caller graph for this function:



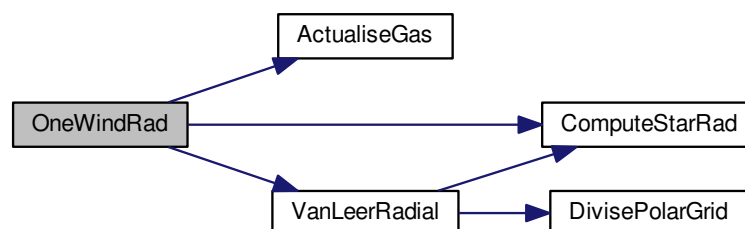
4.31.2.10 void OneWindRad (PolarGrid * Rho, PolarGrid * Vrad, PolarGrid * Energy, real dt)

Definition at line 49 of file TransportEuler.c.

References ActualiseGas(), AdvecteLabel, ComputeStarRad(), EnergyEq, LostMass, RhoInt, RhoStar, VanLeerRadial(), and YES.

Referenced by Transport().

Here is the call graph for this function:



Here is the caller graph for this function:



4.31.2.11 void OneWindRadPebbles (PolarGrid * *Rho*, PolarGrid * *Vrad*, real *dt*)

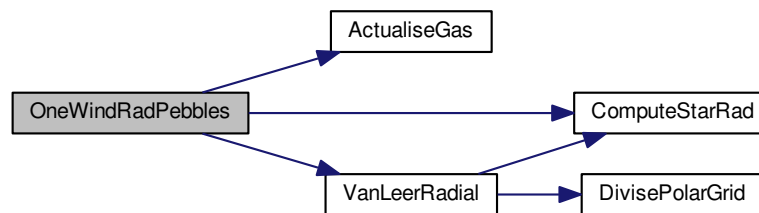
An alternative to [OneWindRad\(\)](#) function for the 2nd fluid.

Definition at line 461 of file TransportEuler.c.

References [ActualiseGas\(\)](#), [ComputeStarRad\(\)](#), [RhoInt](#), [RhoStar](#), and [VanLeerRadial\(\)](#).

Referenced by [TransportPebbles\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:



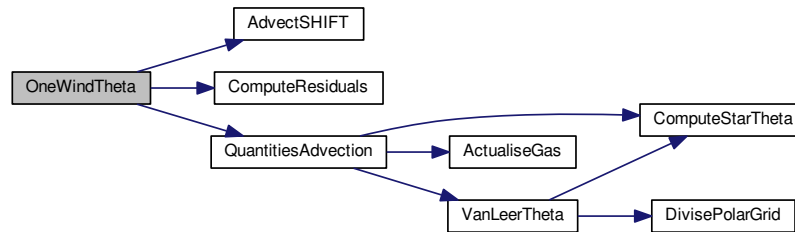
4.31.2.12 void OneWindTheta (PolarGrid * *Rho*, PolarGrid * *Vtheta*, PolarGrid * *Energy*, real *dt*)

Definition at line 138 of file TransportEuler.c.

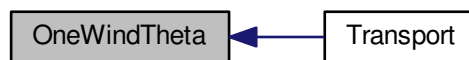
References [AdvecteLabel](#), [AdvectSHIFT\(\)](#), [ComputeResiduals\(\)](#), [EnergyEq](#), [FastTransport](#), [QuantitiesAdvection\(\)](#), and [YES](#).

Referenced by Transport().

Here is the call graph for this function:



Here is the caller graph for this function:



4.31.2.13 void OneWindThetaPebbles (PolarGrid * *Rho*, PolarGrid * *Vtheta*, real *dt*)

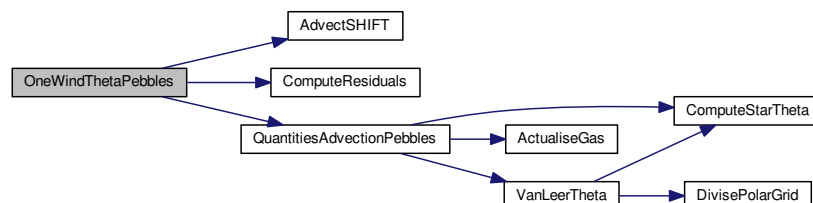
An alternative to [OneWindTheta\(\)](#) function for the 2nd fluid.

Definition at line 476 of file TransportEuler.c.

References [AdvectSHIFT\(\)](#), [ComputeResiduals\(\)](#), [FastTransport](#), [QuantitiesAdvectionPebbles\(\)](#), and [YES](#).

Referenced by [TransportPebbles\(\)](#).

Here is the call graph for this function:



Here is the caller graph for this function:



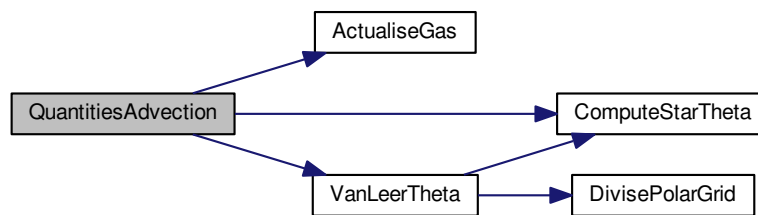
4.31.2.14 void QuantitiesAdvection (PolarGrid * *Rho*, PolarGrid * *Energy*, PolarGrid * *Vtheta*, real *dt*)

Definition at line 159 of file TransportEuler.c.

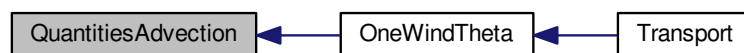
References `ActualiseGas()`, `AdvecteLabel`, `ComputeStarTheta()`, `EnergyEq`, `RhoInt`, `RhoStar`, `VanLeerTheta()`, and `YES`.

Referenced by `OneWindTheta()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.31.2.15 void QuantitiesAdvectionPebbles (PolarGrid * *Rho*, PolarGrid * *Vtheta*, real *dt*)

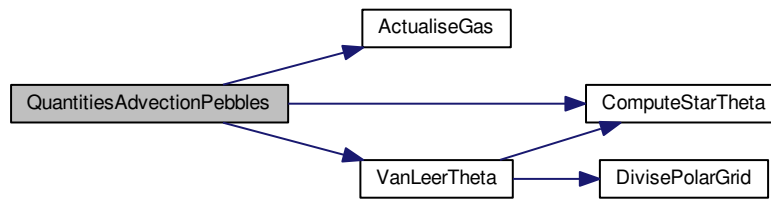
An alternative to [QuantitiesAdvection\(\)](#) function for the 2nd fluid.

Definition at line 494 of file TransportEuler.c.

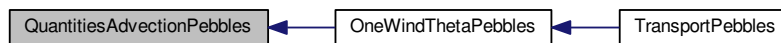
References `ActualiseGas()`, `ComputeStarTheta()`, `RhoInt`, `RhoStar`, and `VanLeerTheta()`.

Referenced by `OneWindThetaPebbles()`.

Here is the call graph for this function:



Here is the caller graph for this function:

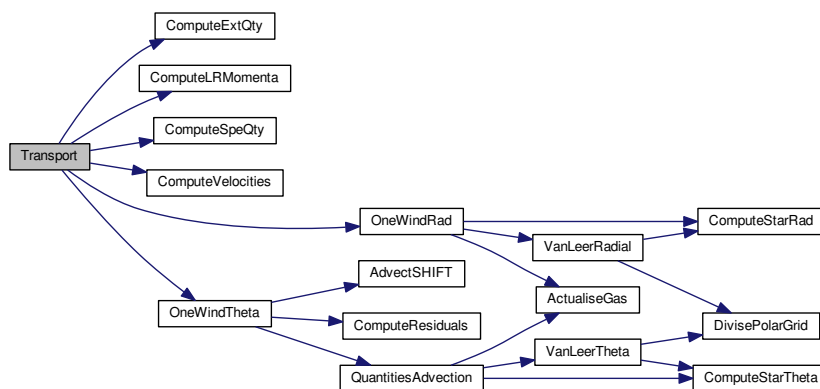


4.31.2.16 void Transport (PolarGrid * *Rho*, PolarGrid * *Vrad*, PolarGrid * *Vtheta*, PolarGrid * *Energy*, PolarGrid * *Label*, real *dt*)

Definition at line 34 of file TransportEuler.c.

References [AdvecteLabel](#), [ComputeExtQty\(\)](#), [ComputeLRMomenta\(\)](#), [ComputeSpeQty\(\)](#), [ComputeVelocities\(\)](#), [OneWindRad\(\)](#), [OneWindTheta\(\)](#), and [YES](#).

Here is the call graph for this function:



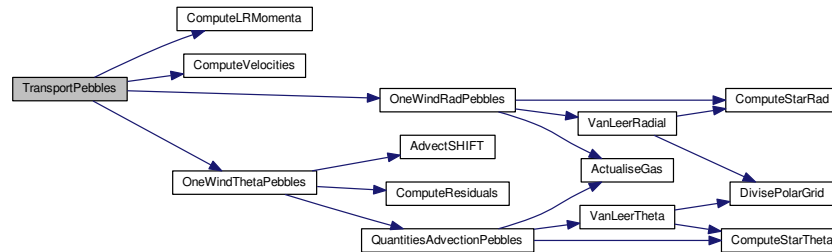
4.31.2.17 void TransportPebbles (PolarGrid * *Rho*, PolarGrid * *Vrad*, PolarGrid * *Vtheta*, real *dt*)

An alternative to [Transport\(\)](#) function for the 2nd fluid.

Definition at line 449 of file TransportEuler.c.

References ComputeLRMomenta(), ComputeVelocities(), OneWindRadPebbles(), and OneWindThetaPebbles().

Here is the call graph for this function:



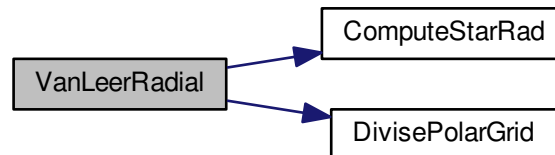
4.31.2.18 real VanLeerRadial (PolarGrid * Vrad, PolarGrid * Qbase, real dt)

Definition at line 384 of file TransportEuler.c.

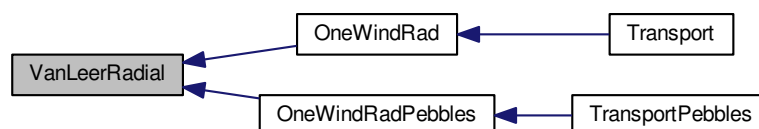
References ComputeStarRad(), DivisePolarGrid(), polargrid::Field, InvSurf, OpenInner, PI, RhoInt, RhoStar, Rinf, Rsup, and YES.

Referenced by OneWindRad(), and OneWindRadPebbles().

Here is the call graph for this function:



Here is the caller graph for this function:



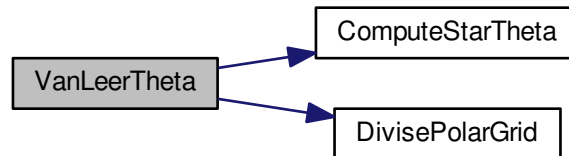
4.31.2.19 `void VanLeerTheta (PolarGrid * Vtheta, PolarGrid * Qbase, real dt)`

Definition at line 417 of file TransportEuler.c.

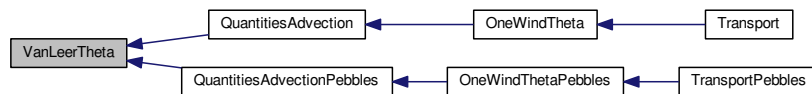
References `ComputeStarTheta()`, `DivisePolarGrid()`, `polargrid::Field`, `RhoInt`, `RhoStar`, `Rinf`, `Rsup`, and `Surf`.

Referenced by `QuantitiesAdvection()`, and `QuantitiesAdvectionPebbles()`.

Here is the call graph for this function:



Here is the caller graph for this function:



4.31.3 Variable Documentation

4.31.3.1 `real* dq [static]`

Definition at line 25 of file TransportEuler.c.

Referenced by `ComputeStarRad()`, `ComputeStarTheta()`, and `InitTransport()`.

4.31.3.2 `PolarGrid * Elongations [static]`

Definition at line 27 of file TransportEuler.c.

4.31.3.3 `PolarGrid * ExtLabel [static]`

Definition at line 26 of file TransportEuler.c.

4.31.3.4 `boolean FastTransport`

Definition at line 29 of file Interpret.c.

Referenced by `ComputeResiduals()`, `OneWindTheta()`, `OneWindThetaPebbles()`, and `ReadVariables()`.

4.31.3.5 real LostMass = 0.0

Definition at line 32 of file TransportEuler.c.

Referenced by OneWindRad(), WriteBigPlanetFile(), and WritePlanetFile().

4.31.3.6 int Nshift[MAX1D] [static]

Definition at line 22 of file TransportEuler.c.

Referenced by AdvectSHIFT(), and ComputeResiduals().

4.31.3.7 real OmegaFrame

Definition at line 20 of file global.h.

Referenced by AccreteOntoPlanets(), AccretePebblesOntoPlanets(), AlgoGas(), BckpFieldsForBC(), ComputeLR↔Momenta(), ComputeVelocities(), DampingBoundary(), DampPebbles(), EtaPressureSupport(), FillVtheta(), Gas↔Momentum(), GasTotalEnergy(), ImposeKeplerianEdges(), InitGasVelocity(), InitPebblesViaFlux(), main(), Source↔TermsPebbles(), and SubStep1().

4.31.3.8 boolean OpenInner

Definition at line 14 of file main.c.

Referenced by VanLeerRadial().

4.31.3.9 PolarGrid * QRStar [static]

Definition at line 27 of file TransportEuler.c.

4.31.3.10 PolarGrid * RadMomM [static]

Definition at line 26 of file TransportEuler.c.

4.31.3.11 PolarGrid * RadMomP [static]

Definition at line 26 of file TransportEuler.c.

4.31.3.12 real * TempShift [static]

Definition at line 23 of file TransportEuler.c.

Referenced by AdvectSHIFT(), and InitTransport().

4.31.3.13 PolarGrid * ThetaMomM [static]

Definition at line 26 of file TransportEuler.c.

4.31.3.14 PolarGrid * ThetaMomP [static]

Definition at line 26 of file TransportEuler.c.

4.31.3.15 int TimeStep

Definition at line 23 of file global.h.

Referenced by AdvanceSystemFromDisk(), and main().

4.31.3.16 real VMed[MAX1D] [static]

Definition at line 21 of file TransportEuler.c.

Referenced by ComputeResiduals().

4.31.3.17 PolarGrid * VthetaRes [static]

Definition at line 26 of file TransportEuler.c.

4.31.3.18 PolarGrid * Work [static]

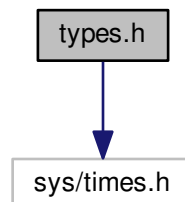
Definition at line 27 of file TransportEuler.c.

4.32 types.h File Reference

Definition of the structures used in the FARGO code.

```
#include <sys/times.h>
```

Include dependency graph for types.h:



This graph shows which files directly or indirectly include this file:



Data Structures

- struct [pair](#)
Set of two reals.
- struct [polargrid](#)

A structure used to store any scalar field on the computational domain.

- struct [param](#)

The Param structure handles the parameters of the parameter file.

- struct [timeprocess](#)

This structure is used for monitoring CPU time usage.

- struct [planetary_system](#)

Contains all the information about a planetary system at a given instant in time.

Macros

- #define [YES](#) 1
- #define [NO](#) 0
- #define [REAL](#) 1
- #define [INT](#) 0
- #define [STRING](#) 2
- #define [SINE](#) 0
- #define [COSINE](#) 1
- #define [ABSCISSA](#) 0
- #define [ORDINATE](#) 1
- #define [HEIGHT](#) 2
- #define [INF](#) 0
- #define [SUP](#) 1
- #define [GET](#) 0
- #define [MARK](#) 1
- #define [FREQUENCY](#) 2
- #define [COM_DENSITY](#) 0
- #define [COM_VRAD](#) 1
- #define [COM_VTHETA](#) 2
- #define [MAX1D](#) 16384
- #define [MAXPLANETS](#) 10

Typedefs

- typedef int [boolean](#)
The boolean type will be used mainly for the variables corresponding to the command line switches.
- typedef double [real](#)
Definition of the type 'real' used throughout the code.
- typedef struct [pair](#) [Pair](#)
- typedef struct [polargrid](#) [PolarGrid](#)
- typedef struct [param](#) [Param](#)
- typedef struct [timeprocess](#) [TimeProcess](#)
- typedef struct [planetary_system](#) [PlanetarySystem](#)

4.32.1 Detailed Description

Definition of the structures used in the FARGO code.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [types.h](#).

4.32.2 Macro Definition Documentation

4.32.2.1 `#define ABSCISSA 0`

Definition at line 53 of file types.h.

4.32.2.2 `#define COM_DENSITY 0`

Definition at line 61 of file types.h.

4.32.2.3 `#define COM_VRAD 1`

Definition at line 62 of file types.h.

4.32.2.4 `#define COM_VTHETA 2`

Definition at line 63 of file types.h.

4.32.2.5 `#define COSINE 1`

Definition at line 52 of file types.h.

4.32.2.6 `#define FREQUENCY 2`

Definition at line 60 of file types.h.

Referenced by `GetPsysInfo()`, `GetPsysInfoFromRsim()`, and `main()`.

4.32.2.7 `#define GET 0`

Definition at line 58 of file types.h.

Referenced by `AlgoGas()`, `GetPsysInfo()`, and `GetPsysInfoFromRsim()`.

4.32.2.8 `#define HEIGHT 2`

Definition at line 55 of file types.h.

4.32.2.9 `#define INF 0`

Definition at line 56 of file types.h.

4.32.2.10 `#define INT 0`

Definition at line 49 of file types.h.

Referenced by `InitVariables()`, `ReadVariables()`, and `var()`.

4.32.2.11 `#define MARK 1`

Definition at line 59 of file types.h.

Referenced by `AlgoGas()`, `GetPsysInfo()`, and `GetPsysInfoFromRsim()`.

4.32.2.12 #define MAX1D 16384

Definition at line 65 of file types.h.

Referenced by ConditionCFL(), CreateTorqueMapInfile(), MakeDir(), and UpdateLog().

4.32.2.13 #define MAXPLANETS 10

Definition at line 67 of file types.h.

Referenced by FillForcesArrays().

4.32.2.14 #define NO 0

Definition at line 47 of file types.h.

Referenced by AdvanceSystemFromDisk(), AlgoGas(), CheckRebin(), DetectCrash(), DiscardParticlesDist(), FillForcesArrays(), GiveSpecificTime(), GiveTimeInfo(), ImplicitRadiativeDiffusion(), InitPlanetarySystem(), InitSpecificTime(), InitVariables(), IterateRelaxationParameter(), main(), merge(), OpacityProfile(), ReadVariables(), SendOutput(), SuccessiveOverrelaxation(), SynchronizeFargoRebound(), SynchronizeOverlapFields(), and var().

4.32.2.15 #define ORDINATE 1

Definition at line 54 of file types.h.

4.32.2.16 #define REAL 1

Definition at line 48 of file types.h.

Referenced by InitVariables(), ReadVariables(), and var().

4.32.2.17 #define SINE 0

Definition at line 51 of file types.h.

4.32.2.18 #define STRING 2

Definition at line 50 of file types.h.

Referenced by InitVariables(), ReadVariables(), and var().

4.32.2.19 #define SUP 1

Definition at line 57 of file types.h.

4.32.2.20 #define YES 1

Definition at line 46 of file types.h.

Referenced by AccreteOntoPlanets(), AdvanceSystemFromDisk(), AlgoGas(), AllocateComm(), AllocPlanetSystem(), ApplyBoundaryCondition(), CheckRebin(), CommunicateBoundaries(), ComputeResiduals(), ConditionCFL(), DetectCrash(), DiscardParticlesDist(), FillForcesArrays(), FillPolar1DArrays(), GetPsysInfo(), GetPsysInfoFromRsim(), GiveTimeInfo(), ImplicitRadiativeDiffusion(), InitPlanetarySystem(), InitVariables(), IterateRelaxationParameter(), ListPlanets(), main(), merge(), OneWindRad(), OneWindTheta(), OneWindThetaPebbles(), OpacityProfile(), QuantitiesAdvection(), ReadVariables(), RestartReboundSimulation(), SendOutput(), SetupIntegrator

Params(), SetupReboundSimulation(), SplitDomain(), SuccessiveOverrelaxation(), SynchronizeFargoRebound(), SynchronizeOverlapFields(), TellEverything(), Transport(), and VanLeerRadial().

4.32.3 Typedef Documentation

4.32.3.1 typedef int boolean

The boolean type will be used mainly for the variables corresponding to the command line switches.

Definition at line 15 of file types.h.

4.32.3.2 typedef struct pair Pair

Definition at line 31 of file types.h.

4.32.3.3 typedef struct param Param

Definition at line 82 of file types.h.

4.32.3.4 typedef struct planetary_system PlanetarySystem

Definition at line 115 of file types.h.

4.32.3.5 typedef struct polargrid PolarGrid

Definition at line 44 of file types.h.

4.32.3.6 typedef double real

Definition of the type 'real' used throughout the code.

You can force FARGO to work in single precision by redefining real to float here. This is an untested feature of FARGO, however, and in practice it should be of little use.

Definition at line 20 of file types.h.

4.32.3.7 typedef struct timeprocess TimeProcess

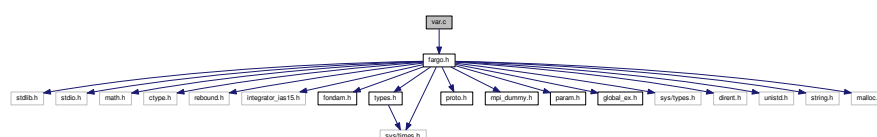
Definition at line 91 of file types.h.

4.33 var.c File Reference

Contains the function that connects the string of the parameter file to global variables.

```
#include "fargo.h"
```

Include dependency graph for var.c:



Macros

- `#define __LOCAL`

Functions

- `void InitVariables ()`

4.33.1 Detailed Description

Contains the function that connects the string of the parameter file to global variables.

The `var()` function is found in [Interpret.c](#)

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [var.c](#).

4.33.2 Macro Definition Documentation

4.33.2.1 `#define __LOCAL`

Definition at line 12 of file [var.c](#).

4.33.3 Function Documentation

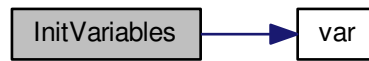
4.33.3.1 `void InitVariables ()`

Definition at line 17 of file [var.c](#).

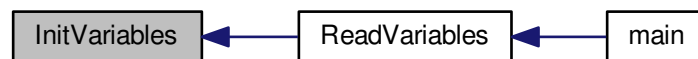
References ACCRETIONALHEATING, ACCRETIONRATE, ADIABIND, ADVLABEL, ALPHAVISCOSITY, ASPECTRATIO, BACKREACTION, CAVITYRADIUS, CAVITYRATIO, CAVITYWIDTH, COOLINGTIME, DAMPINGPERIODFRAC, DAMPINGRMAXFRAC, DAMPINGRMINFRAC, DAMPTOWARDS, DENSINFILE, DISCALBEDO, DISK, DT, ECCENTRICITY, EFFECTIVETEMPERATURE, ENERGYEQUATION, EXCLUDEHILL, FLARINGINDEX, FRAME, GETTORQUEFORPLANET, GRIDSPACING, HEATINGDELAY, HILLCUT, IAS15MINDT, IAS15PRECISION, IMPOSEDDISKDRIFT, INDIRECTTERM, INITIALIZEFROMFILE, INT, LAMBDA DOUBLING, MASSTAPER, NINTERM, NO, NOELEMENTS, NRAD, NSEC, NTOT, OMEGAFRAME, OPACITYDROP, OPENINNERBOUNDARY, OUTERSOURCEMASS, OUTPUTDIR, PARAMETRICACCRETION, PARAMETRICOPACITY, PARTICLEDIFFUSION, PEBBLEACCRETION, PEBBLEALPHA, PEBBLEBULKDENS, PEBBLECOAGULATION, PEBBLEFLUX, PLANETARYDENSITY, PLANETCONFIG, PLANETSFEELDISK, REAL, RELEASEDATE, RELEASERADIUS, RESOLVECOLLISIONS, RMAX, RMIN, ROCHESMOOTHING, SCHMIDTNUMBER, SIGMA0, SIGMASLOPE, STELLARRADIATION, STELLARRADIUS, STRING, TARGETNPL, TEMPERINFILE, THICKNESSSMOOTHING, TORQUEMAPINFILE, TRANSITIONRADIUS, TRANSITIONRATIO, TRANSITIONWIDTH, TRANSPORT, `var()`, VERTICALDAMPING, VISCOSITY, VRADINFILE, VTHETA INFILE, WRITEDENSITY, WRITEDIVV, WRITEENERGY, WRITEETA, WRITEQBALANCE, WRITEQPLUS, WRITETEMPERATURE, WRITETORQUEFILES, WRITEVELOCITY, and YES.

Referenced by `ReadVariables()`.

Here is the call graph for this function:



Here is the caller graph for this function:

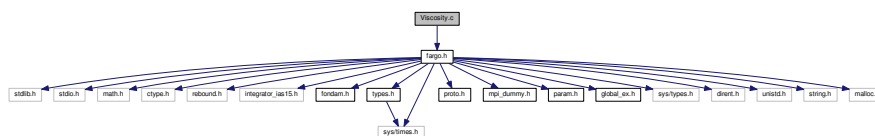


4.34 Viscosity.c File Reference

Calculation of the viscous force.

```
#include "fargo.h"
```

Include dependency graph for Viscosity.c:



Functions

- `real FViscosity (real rad)`
- `real AspectRatio (real rad)`
- `void InitViscosity ()`
- `void UpdateDivVelocAndStressTensor (PolarGrid *Vrad, PolarGrid *Vtheta, PolarGrid *Rho)`
A function derived from the original `ViscousTerms()`.
- `void UpdateVelocityWithViscousTerms (PolarGrid *Vrad, PolarGrid *Vtheta, PolarGrid *Rho, real DeltaT)`
A function derived from the original `ViscousTerms()`.
- `void ImposeKeplerianEdges (PolarGrid *Vtheta)`
A function derived from the original `ViscousTerms()`.

Variables

- static `PolarGrid * DRR`

- static [PolarGrid](#) * [DRP](#)
- static [PolarGrid](#) * [DPP](#)

4.34.1 Detailed Description

Calculation of the viscous force.

The function [FViscosity\(\)](#) returns the (kinematic) viscosity as a function of the radius (it handles all case: alpha or uniform viscosity, and inner cavity with a different viscosity). The update of the velocity is done in [ViscousTerm\(\)](#), which properly evaluate the stress tensor in 2D cylindrical coordinates. This file also contains the function [AspectRatio\(\)](#), which gives the aspect ratio as a function of the radius, in the case of a temperature jump in the disk (much in the manner as cavities arising from a viscosity jump are handled, hence the location of this function). Note that [AspectRatio\(\)](#) does not feature the FLARINGINDEX, which is taken into account by the calling function.

Author

THORIN modifications by Ondřej Chrenko chrenko@sirrah.troja.mff.cuni.cz, Copyright (C) 2017; original code by Frédéric Masset

Definition in file [Viscosity.c](#).

4.34.2 Function Documentation

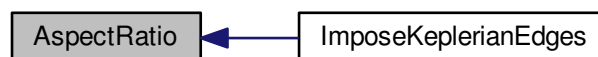
4.34.2.1 `real AspectRatio (real rad)`

Definition at line 50 of file [Viscosity.c](#).

References [ASPECTRATIO](#), [LAMBDA DOUBLING](#), [PhysicalTime](#), [PhysicalTimeInitial](#), [TRANSITIONRADIUS](#), [TRANSITIONRATIO](#), and [TRANSITIONWIDTH](#).

Referenced by [ImposeKeplerianEdges\(\)](#).

Here is the caller graph for this function:



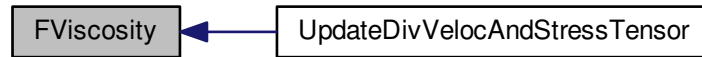
4.34.2.2 `real FViscosity (real rad)`

Definition at line 26 of file [Viscosity.c](#).

References [ALPHA VISCOSITY](#), [ASPECTRATIO](#), [CAVITYRADIUS](#), [CAVITYRATIO](#), [CAVITYWIDTH](#), [GlobalRmed](#), [globcsvec](#), [LAMBDA DOUBLING](#), [PhysicalTime](#), [PhysicalTimeInitial](#), [VISCOSITY](#), and [ViscosityAlpha](#).

Referenced by [UpdateDivVelocAndStressTensor\(\)](#).

Here is the caller graph for this function:



4.34.2.3 void ImposeKeplerianEdges (PolarGrid * Vtheta)

A function derived from the original [ViscousTerms\(\)](#).

Definition at line 215 of file Viscosity.c.

References `AspectRatio()`, `CPU_Number`, `CPU_Rank`, `FLARINGINDEX`, `G`, `polargrid::Nrad`, `OmegaFrame`, `Rmed`, and `SIGMASLOPE`.

Here is the call graph for this function:



4.34.2.4 void InitViscosity ()

Definition at line 67 of file Viscosity.c.

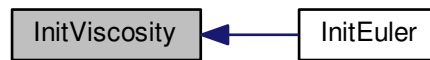
References `CreatePolarGrid()`, `DivergenceVelocity`, `NRAD`, `NSEC`, `TAUPP`, `TAURP`, and `TAURR`.

Referenced by `InitEuler()`.

Here is the call graph for this function:



Here is the caller graph for this function:



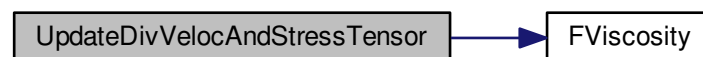
4.34.2.5 void UpdateDivVelocAndStressTensor (PolarGrid * Vrad, PolarGrid * Vtheta, PolarGrid * Rho)

A function derived from the original [ViscousTerms\(\)](#).

Definition at line 79 of file Viscosity.c.

References DivergenceVelocity, polargrid::Field, FViscosity(), InvDiffRmed, InvDiffRsup, InvRinf, InvRmed, polargrid::Nrad, Rinf, Rmed, Rsup, TAUPP, TAURP, and TAURR.

Here is the call graph for this function:



4.34.2.6 void UpdateVelocityWithViscousTerms (PolarGrid * Vrad, PolarGrid * Vtheta, PolarGrid * Rho, real DeltaT)

A function derived from the original [ViscousTerms\(\)](#).

Definition at line 159 of file Viscosity.c.

References polargrid::Field, InvDiffRmed, InvDiffRsup, InvRinf, InvRmed, polargrid::Nrad, Rinf, Rmed, Rsup, TAUPP, TAURP, and TAURR.

4.34.3 Variable Documentation

4.34.3.1 PolarGrid * DPP [static]

Definition at line 23 of file Viscosity.c.

4.34.3.2 PolarGrid * DRP [static]

Definition at line 23 of file Viscosity.c.

4.34.3.3 PolarGrid * DRR [static]

Definition at line 23 of file Viscosity.c.

Index

clicks
timeprocess, [12](#)

Field
polargrid, [10](#)

Name
polargrid, [11](#)

name
param, [6](#)
timeprocess, [12](#)

necessary
param, [6](#)

Nrad
polargrid, [11](#)

Nsec
polargrid, [11](#)

pair, [5](#)
x, [5](#)
y, [5](#)

param, [6](#)
name, [6](#)
necessary, [6](#)
read, [6](#)
type, [6](#)
variable, [7](#)

polargrid, [10](#)
Field, [10](#)
Name, [11](#)
Nrad, [11](#)
Nsec, [11](#)

read
param, [6](#)

timeprocess, [11](#)
clicks, [12](#)
name, [12](#)

type
param, [6](#)

variable
param, [7](#)

x
pair, [5](#)

y
pair, [5](#)