GRAPE-DR G6 compatibility library reference manual

Ver 0.0 — 2010/9/20

Jun Makino

November 18, 2010

Contents

1 Desctription of multi-chip interface functions **3**

1.1 g6_set_debug_level . . . 3
1.2 g6_report_timers . . . 3
1.3 g6_open . . . 3
1.4 g6_close . . . 3
1.5 g6_set_ti . . . 4
1.6 g6_set_j_particle . . . 4
1.7 g6_npipe . . . 5
1.8 g6calc_firsthalf0 . . . 5
1.9 g6calc_firsthalf . . . 6
1.10 g6calc_lasthalf0a . . . 6
1.11 g6calc_lasthalf2 . . . 7
1.12 g6calc_lasthalf . . . 8
1.13 calculate_accel_by_grape6_separate_trial_noopen . . . 8

2 KFCR-compativility interface **9**

2.1 Unimplemented KFCR functions . . . 9

3 Kokubo-proposed functions **10**

4 Compiling and Linking **10**

5 Limitations etc. **11**

6 Update history **11**

6.1 12 Nov 2011 . . . 11

1 Desctription of multi-chip interface functions

Currently, G6 library supports only the simultaneous use of four chips on a card. Also, clusterid argument, which is provided for the compatibility, currently must be 0 (will support multiple cards in future).

All functions support Fortran and C interfaces.

1.1 g6_set_debug_level

```
void g6_set_debug_level(int level);
```

Enable debug messages.

```
level 0 (default) : No message
level 1 and above : g6_open, g6_set_ti
level 2           : g6_set_j_particle
level 4           : firsthalf and lasthalf
```

levels 2 and 4 are bit flags. So, in order to get full debug message, call this function with level=7 (level=15 or higher might print some more)

1.2 g6_report_timers

```
void g6_report_timers()
```

Print out some performance information.

1.3 g6_open

```
int g6_open_(int *clusterid) (Fortran interface)

int g6_open(int clusterid)
```

Get one card and initialize it.

1.4 g6_close

```
int g6_close_(int *clusterid)

int g6_close(int clusterid)
```

Release the card.

1.5 g6_set_ti

```
void g6_set_ti_(int *clusterid, double *ti)
void g6_set_ti(int clusterid, double ti)
```

This function sets the present time (t_i) for the predictor unit.

1.6 g6_set_j_particle

```
int g6_set_j_particle_(int * clusterid,
                       int *address,
                       int *index,
                       double *tj,
                       double *dtj,
                       double *mass,
                       double a2by18[3],
                       double a1by6[3],
                       double aby2[3],
                       double v[3],
                       double x[3])

int g6_set_j_particle(int  clusterid,
                       int address,
                       int index,
                       double tj,
                       double dtj,
                       double mass,
                       double a2by18[3],
                       double a1by6[3],
                       double aby2[3],
                       double v[3],
                       double x[3])
```

This function sends the so-called j-particle data to the memory unit of GRAPE-6. x, v, aby2, a1by6, a2by18 are position, velocity, half of the acceleration, 1/6 of the first time derivative of the acceleration, and 1/18 of the second time derivative. These must all be the arrays of size three. tj, dtj and mass are the time, timestep and mass of the particle. address is the location in the GRAPE-6 memory to store the particle, starting from index 0, and index is the identifier for the particle itself, which may be different from address above.

The time and timestep must be acceptable for the blockstep algorithm, which means that the timestep must be the powers of two (negative values allowed, but smaller than the time resolution set by g6_set_tunit} should result in an error). Also the time must be an exact integer multiple of the timestep.

Note that the GRAPE-6 predictor unit can make use of the second time derivative of the force, unlike the GRAPE-4 predictor unit which can use only up to the first time derivative. If the application program cannot provide the second time derivative, it must give the pointer of the array with size three filled with zeros.

1.7 g6_npipe

```
int g6_npipes()

int g6_npipes_
```

Returns the number of pipelines.

1.8 g6calc_firsthalf0

```
void g6calc_firsthalf0_(int * clusterid,
                        int * nj,
                        int * ni,
                        int index[],
                        double x[][3],
                        double v[][3],
                        double fold[][3],
                        double j6old[][3],
                        double phiold[],
                        double eps2[],
                        double h2[],
                        int * mode)

 void g6calc_firsthalf0(int clusterid,
                        int nj,
                        int ni,
                        int index[],
                        double xi[][3],
                        double vi[][3],
                        double fold[][3],
                        double j6old[][3],
                        double phiold[],
                        double eps2[],
                        double h2[],
                        int  mode)
```

This function starts the calculation of forces to ni particles, with index, position and velocity given by index, xi, and vi. Arguments fold, j6old, phiold and h2 are

not used, and can be NULL. The mode argument controls the interpretation of eps2. If mode=0, eps2 is regarded as an array, giving different values for differnt particles. If mode=1, the value of eps2[0] is used for all particles.

1.9 g6calc_firsthalf

```
void g6calc_firsthalf_(int * clusterid,
                       int * nj,
                       int * ni,
                       int index[],
                       double xi[][3],
                       double vi[][3],
                       double fold[][3],
                       double j6old[][3],
                       double phiold[],
                       double *eps2,
                       double h2[])

void g6calc_firsthalf(int  clusterid,
                       int  nj,
                       int  ni,
                       int index[],
                       double xi[][3],
                       double vi[][3],
                       double fold[][3],
                       double j6old[][3],
                       double phiold[],
                       double eps2,
                       double h2[])
```

Same as calling g6calc_firsthalf0 with mode=1.

1.10 g6calc_lasthalf0a

```
int g6calc_lasthalf0a_(int * clusterid,
                       int * nj,
                       int * ni,
                       int index[],
                       double xi[][3],
                       double vi[][3],
                       double *eps2,
                       double h2[],
                       double acc[][3],
```

```
                    double jerk[][3],
                    double pot[],
                    int nnbindex[],
                    int *mode)

int g6calc_lasthalf0a(int clusterid,
                     int nj,
                     int ni,
                     int index[],
                     double xi[][3],
                     double vi[][3],
                     double *eps2,
                     double h2[],
                     double acc[][3],
                     double jerk[][3],
                     double pot[],
                     int nnbindex[],
                     int mode)
```

Wait the GDR card to finish the calculation and returns the result. "mode" argument is currently unused (in original GRAPE-6 library, it was used for error recovery).

1.11 g6calc_lasthalf2

```
int g6calc_lasthalf2_(int * clusterid,
                    int * nj,
                    int * ni,
                    int index[],
                    double xi[][3],
                    double vi[][3],
                    double *eps2,
                    double h2[],
                    double acc[][3],
                    double jerk[][3],
                    double pot[],
                    int nnbindex[])
```

int g6calc_lasthalf2(int clusterid,

```
                    int  nj,
                    int  ni,
                    int index[],
```

```
                    double xi[][3],
                    double vi[][3],
                    double eps2,
                    double h2[],
                    double acc[][3],
                    double jerk[][3],
                    double pot[],
                    int nnbindex[])
```

Same as g6calc_lasthalf0a with mode=1

1.12 g6calc_lasthalf

```
int g6calc_lasthalf_(int * clusterid,
                    int * nj,
                    int * ni,
                    int index[],
                    double xi[][3],
                    double vi[][3],
                    double *eps2,
                    double h2[],
                    double acc[][3],
                    double jerk[][3],
                    double pot[])

int g6calc_lasthalf(int  clusterid,
                    int  nj,
                    int  ni,
                    int index[],
                    double xi[][3],
                    double vi[][3],
                    double eps2,
                    double h2[],
                    double acc[][3],
                    double jerk[][3],
                    double pot[])
```

Same as lasthalf2 except that nnindex is not returned.

1.13 calculate_accel_by_grape6_separate_trial_noopen

```
int calculate_accel_by_grape6_separate_trial_noopen(int clusterid,
                                                    int ni,
                                                    double xi[][3],
```

```
                                        double vi[][3],
                                        int nj,
                                        double xj[][3],
                                        double vj[][3],
                                        double m[],
                                        double a[][3],
                                        double jerk[][3],
                                        double pot[],
                                        double eps2)
```

Simple interface for calculating forces from one group of particles to another. Useful for simple N-squared code and/or treecode (However, G5 compatibility library is faster for these usage).

2 KFCR-compativility interface

Thw following functions are provided as compatibility interface of KFCR-version libraries.

```
void g6_open_all(void)
void g6_close_all(void)
int g6_set_j_particle_all(int address, int index, double tj, double dtj,
                          double mass, double a2by18[3], double a1by6[3],
                          double aby2[3], double v[3], double x[3])
int g6_set_j_particle_mxonly_all(int address, int index, double mass,
void g6_set_ti_all(double ti)
void g6calc_firsthalf_all(int nj, int ni, int index[], double xi[][3],
                          double vi[][3], double fold[][3], double jold[][3],
                          double phiold[], double eps2, double h2[])
int g6calc_lasthalf_all(int nj, int ni, int index[], double xi[][3],
                        double vi[][3], double eps2, double h2[],
                        double acc[][3], double jerk[][3], double pot[])
int g6calc_lasthalf2_all(int nj, int ni, int index[], double xi[][3],
                         double vi[][3], double eps2, double h2[],
                         double acc[][3], double jerk[][3], double pot[],
                         int nnbindex[])
g6calc_lasthalf2(0, nj, ni, index, xi, vi, eps2, h2,
                 acc, jerk, pot,nnbindex);
```

2.1 Unimplemented KFCR functions

Since the neighbour list dies not exist, the following functions are not implemented.

```
int g6_read_neighbour_list_all(void)
int g6_get_neighbour_list_all(int ipipe, int maxlength,
                              int *nblen, int nbl[])
```

The following functions are obsolete. One need not use these functions.

```
void g6_set_nip_all(int nip)
void g6_set_njp_all(int njp)
void g6_set_i_particle_scales_from_real_value_all(int address, double acc[3],
                                                  double jerk[3], double phi,
                                                  double jfactor,
                                                  double ffactor)
void g6_set_i_particle_all(int address, int index, double x[3],
                           double v[3], double eps2, double h2)
int g6_get_force_all(double acc[][3], double jerk[][3], double phi[],
                     int flag[])
```

3 Kokubo-proposed functions

The followings are functions proposed by Eiichiro Kokubo.

```
void g6_set_particle(int n, int ilist[],
                     double m[], double x[][3], double v[][3],
                     double a[][3], double j[][3],
                     double t[], double dt[])
```

This function set multiple j-particles in one call.

```
void g6_calc_force(int nj, int ni, int ilist[], int nnb[],
                   double t,
                   double xp[][3], double vp[][3],
                   double a_old[][3], double j_old[][3],
                   double a_new[][3], double j_new[][3], double pot[],
                   double eps2, double h2[])
```

This function calculates forces on ni particles. Here, ni can be larger than the physical number of pipelines.

4 Compiling and Linking

The header file is grape6.h. So you need

```
#include "grape6.h"
```

and compile option

```
-I$GDRBASEDIR/lib [In Makefile, -I$(GDRBASEDIR)/lib]
```

To link, use

```
-L$GDRBASEDIR/lib -lgdrg6 -lsing -lhib
[In Makefile, -L$(GDRBASEDIR)/lib -lgdrg6 -lsing -lhib]
```

5 Limitations etc.

This version assumes 4-chip, single card. Thus, clusterid argument currently must be 0.

Neighbour list functions are not implemented yet.

Calling g6_open and g6_close many times seem to cause error. Avoid calling them more than 10 times.

6 Update history

6.1 12 Nov 2011

Bug in g6_calc_force fixed (untested yet).