Data Reorganization Interface (DRI)

Kenneth Cain Jr., Mercury Computer Systems, Inc.
Anthony Skjellum, MPI Software Technology
On behalf of the Data Reorganization Forum

http://www.data-re.org

HPEC – September 2002
Outline

- DRI specification highlights
  - Design/application space
  - Features in DRI 1.0
  - 2-D FFT Example*
  - Journal of development (features discussed, but not included in DRI 1.0)

- DRI forum status

- Current/future value of DRI to HPEC

* Thanks to Steve Paavola of Sky Computers for contributing to the 2-D FFT example code.
Design/Application Space

- **Generic Design Space**
  - Signal/image processing data flows
  - Data-parallel applications
  - Clique or pipeline designs
  - Stream real-time processing

- **Applications Facilitated**
  - Space-Time Adaptive Processing (STAP)
  - Synthetic Aperture Radar (SAR)
  - Others...

- **Algorithms Facilitated**
  - Lowpass filter
  - Beamforming
  - Matched filter
  - Others…
DRI Parallel Mapping

Source Processes

- DRI_Overlap[] (left, right)
- DRI_Partition[]
- Distributed Data
  - DRI_Distribution
- Local Data Info
  - DRI_Datapart

Destination Processes

- Global Data
  - DRI_GlobalData
- Distributed Data
  - DRI_Distribution
- DRI_Partition[]
- Local Data Info
  - DRI_Datapart

- Parallel Mapping
- Accessor function output
- Constructor function input
- [] array (length = # dimensions)
DRI Communication Setup

Source Processes

- Local Data Info
  - DRI_Datapart

- Distributed Data
  - DRI_Distribution

- Global Data
  - DRI_Globaledata

Destination Processes

- Local Data Info
  - DRI_Datapart

- Distributed Data
  - DRI_Distribution

- DRI_Group

- DRI_Dataspec

- DRI_Reorg

- Constructor function input

- Accessor function output

- Parallel Mapping

- Parallel communication

[] array (length = # dimensions)
DRI Parallel Communication

Source Processes

- Local Data Info
  - DRI_DATapart

- Local Block(s) Info
  - DRI_Blockinfo

Distributed Data
- DRI_Distribution

Global Data
- DRI_Globaldata

Destination Processes

- Local Data Info
  - DRI_DATapart

- Local Block(s) Info
  - DRI_Blockinfo

Distributed Data
- DRI_Distribution

DRI_Reorg
- DRI_Reorg_get DATapart
  - <produce data>
- DRI_Reorg_put DATapart

DRI_Reorg
- Constructor function input

Parallel Mapping
- Accessor function output

[] array
(length = # dimensions)
DRI In-Place Processing

Dest Processes (Reorg_1),
Source Processes (Reorg_2)

Dest Processes (Reorg_2)

Global Data
(gdo_1)
DRI_Globaldata

Distributed Data
DRI_Distribution

DRI_Reorg_process_inplace (Reorg1_dest, Reorg2_src)

DRI_Reorg
(Reorg1_dest)

DRI_Reorg
(Reorg2_src)

DRI_Reorg
(Reorg2_dest)

DRI_Reorg_get_dapart
DRI_Reorg_put_dapart

Local Data Info
DRI_Datapart

<process in-place>
DRI 2-D FFT Example

DRI_Globaldata gdo; // global data object
DRI_Distribution srcDist, dstDist; // distribution objects
DRI_Reorg srcReorg, dstReorg; // data reorg objects
DRI_Datapart dpo; // Datapart object
DRI_Blockinfo block; // block description
DRI_Group allprocs; // all processes
int gdo_dimsizes[2] = {1024, 512}; // 1024 matrix cols
int cols = gdo_dimsizes[0]; // 1024 matrix cols
int rows = gdo_dimsizes[1]; // 512 matrix rows
int my_rows, my_cols // my rows, cols
int i; // loop counter
complex *matrix;

DRI_Init (&argc, &argv);
DRI_Globaldata_create (2, gdo_dimsizes, &gdo);
DRI_Distribution_create (gdo, DRI_GROUPDIMS_ALLNOPREFERENCE, DRI_PARTITION_WB, DRI_LAYOUT_PACKED_01, &srcDist);
DRI_Distribution_create (gdo, DRI_GROUPDIMS_ALLNOPREFERENCE, DRI_PARTITION_BW, DRI_LAYOUT_PACKED_10, &dstDist);
DRI_Reorg_create (DRI_REORG_SEND, "cornerturn", DRI_COMPLEX, srcDist, allprocs, 1, 0, &srcReorg);
DRI_Reorg_create (DRI_REORG_RECV, "cornerturn", DRI_COMPLEX, dstDist, allprocs, 1, 0, &dstReorg);
DRI_Reorg_connect(&srcReorg);
DRI_Reorg_connect(&dstReorg);

DRI_Reorg_get_datapart(srcReorg, &(void *)matrix, &dpo);
DRI_Datapart_get_blockinfo(dpo, 0, &block);
my_rows = DRI_Blockinfo_length(block, 1);

for (i=0; i<my_rows; i++)
    // FFT row starting at address (complex*)matrix + i*cols;
    DRI_Reorg_put_datapart(srcReorg, &dpo);

DRI_Reorg_get_datapart(dstReorg, &(void *)matrix, &dpo);
DRI_Datapart_get_blockinfo(dpo, 0, &block);
my_cols = DRI_Blockinfo_length(block, 1);

for (i=0; i<my_cols; i++)
    // FFT col starting at address (complex*)matrix + i*rows;
    DRI_Reorg_put_datapart(dstReorg, &dpo);

// Destroy all DRI objects with DRI_Object_destroy()
DRI_Finalize();
Topics discussed, but not in DRI 1.0
- Piecemeal data production/consumption
- User-allocated memory
- Generalized buffer sharing
- Split data types (e.g., complex, RGB, …)
- Interoperation with VSIPL
- Dynamic
  - Transfer sizes (e.g., mode change support)
  - Buffer sizes (lower memory impact for multi-buffering)
  - Process sets: pick process set pair at transfer-time
- Non-CPU (device) endpoints

To address these issues, vendors/users
- Develop desired extensions (necessarily non-portable)
- Use experience to focus requirements for future standardization (if appropriate)
DRI Forum Status

DRI-1.0 Ratified in September 2002

Voting Institutions
- Mercury Computer Systems, Inc.
- MPI Software Technology, Inc.
- Sky Computers, Inc.
- SPAWAR Systems Center, San Diego
- The MITRE Corporation

Future
- Gather feedback about DRI 1.0 evaluation/use
- Further specification TBD based on community interest

See web page/email list for latest info.
- Web: http://www.data-re.org
- Email list: data-reorg@data-re.org
DRI Value to HPEC

- **DRI current value to HPEC community:**
  - Implementable with high performance
  - Packages often independently developed data flow features
  - High level of abstraction
    - Reduce source lines of code dedicated to data flow
    - Reduce development time associated with data flow

- **DRI future value to HPEC community:**
  - Common framework/terminology in which to focus community’s requirements
  - DRI 1.0 journal of development provides good starting list of expanded features
Recommendations

- Implement DRI
- Evaluate/use DRI
  - Inquire with HW/SW vendors about availability
  - Evaluate implementations, provide feedback
- Focus application requirements
  - Identify what DRI (and other middlewares in this space) must provide
  - Provide feedback to DRI, other HPEC forums
- Innovate with DRI
  - Develop reference implementations
  - Consider enhancements
    - C++ bindings
    - Integration with processing middleware