# pt2pt wg

# FP16

## MPI Forum Meeting
## Dec., 2018 in San Jose

Atsushi Hori
RIKEN Center for Computational Science

# Tickets and Goal

- #65 16-bit floating-point support for C/C++
  - ➡ Introducing FP16 Datatype

- #74 Datatype Naming Rule
  - ➡ Defining naming rules so that implementors can introduce new datatypes, and
  - ➡ new datatype aliases following the naming rule

★Which way to go ?

# Catch-up: Datatype Naming Rule

- Rationale of Naming Rule
  - One-to-one mapping between language datatype and MPI datatype
  - Users can easily identify MPI datatype from language datatype and vice versa.
  - Implementors can create a unique MPI datatype name from the language datatype
- Proposed Rule

  MPI_<KIND>_<TYPENAME>[__<EXTRA>]
  - Ex) MPI_C_FLOAT16, MPI_F_REAL2, …

# Proposed aliases of the current datatype names

| C datatype names | Aliases following the naming rule |
|---|---|
| MPI_CHAR | MPI_C_CHAR |
| MPI_SHORT | MPI_C_SHORT |
| MPI_SHORT_FLOAT | MPI_C_SHORT_FLOAT |
| MPI_INT | MPI_C_INT |
| MPI_LONG | MPI_C_LONG |
| MPI_LONG_LONG_INT | MPI_C_LONG_LONG |
| MPI_LONG_LONG | MPI_C_LONG_LONG |
| MPI_SIGNED_CHAR | MPI_C_SINGED_CHAR |
| MPI_UNSIGNED_CHAR | MPI_C_UNSIGNED_CHAR |
| MPI_UNSIGNED_SHORT | MPI_C_UNSIGNED_SHORT |
| MPI_UNSIGNED | MPI_C_UNSIGNED |
| MPI_UNSIGNED_LONG | MPI_C_UNSIGNED_LONG |
| constMPI_UNSIGNED_LONG_LONG | MPI_C_UNSIGNED_LONG_LONG |
| MPI_FLOAT | MPI_C_FLOAT |
| MPI_DOUBLE | MPI_C_DOUBLE |
| MPI_LONG_DOUBLE | MPI_C_LONG_DOUBLE |
| MPI_WCHAR | MPI_C_WCHAR |
| MPI_C_BOOL | MPI_C_C_BOOL |
| MPI_INT8_T | MPI_C_INT8_T |
| MPI_INT16_T | MPI_C_INT16_T |
| MPI_INT32_T | MPI_C_INT32_T |
| MPI_INT64_T | MPI_C_INT64_T |
| MPI_UINT8_T | MPI_C_UINT8_T |
| MPI_UINT16_T | MPI_C_UINT16_T |
| MPI_UINT32_T | MPI_C_UINT32_T |
| MPI_UINT64_T | MPI_C_UINT64_T |
| MPI_C_COMPLEX | - |
| MPI_C_FLOAT_COMPLEX | - |
| MPI_C_DOUBLE_COMPLEX | - |
| MPI_C_LONG_DOUBLE_COMPLEX | - |

Table 4.3: Aliases of predefined MPI datatype names in C

| Fortran datatype names | Aliases following the naming rule |
|---|---|
| MPI_CHARACTER | MPI_F_CHARACTER |
| MPI_LOGICAL | MPI_F_LOGICAL |
| MPI_INTEGER | MPI_F_INTEGER |
| MPI_REAL | MPI_F_REAL |
| MPI_DOUBLE_PRECISION | MPI_F_DOUBLE_PRECISION |
| MPI_COMPLEX | MPI_F_COMPLEX |
| MPI_DOUBLE_COMPLEX | MPI_F_DOUBLE_COMPLEX |
| (the followings are optional predefined datatypes) | |
| MPI_INTEGER1 | MPI_F_INTEGER__1 |
| MPI_INTEGER2 | MPI_F_INTEGER__2 |
| MPI_INTEGER4 | MPI_F_INTEGER__4 |
| MPI_INTEGER8 | MPI_F_INTEGER__8 |
| MPI_INTEGER16 | MPI_F_INTEGER__16 |
| MPI_REAL2 | MPI_F_REAL__2 |
| MPI_REAL4 | MPI_F_REAL__4 |
| MPI_REAL8 | MPI_F_REAL__8 |
| MPI_REAL16 | MPI_F_REAL__16 |
| MPI_COMPLEX4 | MPI_F_COMPLEX__4 |
| MPI_COMPLEX8 | MPI_F_COMPLEX__8 |
| MPI_COMPLEX16 | MPI_F_COMPLEX__16 |
| MPI_COMPLEX32 | MPI_F_COMPLEX__32 |

Table 4.4: Aliases of predefined MPI datatype names in Fortran

# Current Status on FP16

- Architectures
  - CPU (GPU)
    - AArch64 — FP16 is supported
    - X86 — FP16 conversion is supported
    - Power9 — FP16 conversion is supported
    - Nvidia — FP16 is supported
  - Network
    - Mellanox — FP16 (and others) is (are) supported
- Language Standards
  - C — _Float16 (ISO/IEC JTC 1/SC 22/WG 14 N1896)
  - C++ — _Half_float (ISO/IEC JTC1 SC22 WG14 N2017)
- Compilers
  - GCC — __fp16 (ARM only)
  - LLVM — _Float16
  - Intel — Intrinsics to convert FP16 type

# MPI Implementations

- MPICH
  MPIX_C_FLOAT16

- Open MPI
  - Fujitsu/Mellanox MPIX_SHORT_FLOAT
    - Mellanox/Uber are running Horovod (TensorFlow) over patched Open MPI
      - CPU is not involved in reductions (GPU & Network)
- others ?

# Upcoming Datatype

- The bfloat16 format is utilized in upcoming Intel AI processors, such as Nervana NNP-L1000, Xeon processors, and Intel FPGAs, Google Cloud TPUs, and TensorFlow.

  — https://en.wikipedia.org/wiki/Minifloat

# Summary of Current Status

- User demands are growing
  - MPI is considered as a communication library for AI/DL frameworks
  - not only AI/DL but also traditional HPC apps
- C language standard does not define FP16 yet
  - C working group proposal
- Not all major C compilers support FP16
- Some MPI implementations (will) have FP16 independent from the MPI standard
  - MPICH, Open MPI
- Yet another FP16 datatype (bfloat16) is on its way

# Major Arguments So Far

- #65 16-bit floating-point support for C/C++
  - We should wait until C standardize FP16

- #74 Datatype Naming Rule
  - We MAY loose portability

# My Objections

- Support FP16 *as soon as possible*
  - It can be a good message to MPI users that MPI supports what users want when users want
  - Some MPI implementations already support FP16 out of the MPI standard
  - We force users to have another programming effort to utilize FP16 unless MPI supports FP16
- Naming rule
  - As long as major MPI implementations have good manners, no portability issue arises

# My Points

- MPI supports FP16 ASAP
- Without having C standard, MPI datatype names cannot be defined
- Give implementors the freedom of introducing new datatypes following the datatype naming rule

# Straw Votes

- ~~MPI should support FP16 as soon as possible~~
  - ~~Y[ ] N[ ] A[ ] O( )~~
- MPI should not define FP16 datatype until C standard defines it
  - Y[15] N[3] A[1] O( )
- ~~MPI should have FP16 datatype according to the C working group proposal - MPI_FLOAT16 (MPI_FLOAT32 and MPI_FLOAT64)~~
  - ~~Y[ ] N[ ] A[ ] O( )~~
- ~~MPI should have the datatype naming rule~~
  - ~~Y[ ] N[ ] A[ ] O( )~~