ADACS Merit Allocation Program Expression of Interest (Software Support)

Project Title: Exploring GPU Acceleration of the DiFX Software Correlator

Project Background: The Distributed FX (DiFX[1]) software correlator is a widely adopted package for the correlation of data from radio interferometers. In contrast to most radio astronomy correlators, it is general purpose and has been used by an enormous number of national and international facilities, including the Long Baseline Array and special modes of the Australian SKA Pathfinder (Australia), the Very Long Baseline Array (USA), the Event Horizon Telescope (international), virtually all major geodetic Very Long Baseline Interferometry operations world-wide, and numerous other smaller facilities. The DiFX user community has >100 members, with around 10 active coders. DiFX was recently awarded the ASA Peter McGregor Prize for astronomical instrumentation. Many future use cases for DiFX have large and expensive correlation requirements, including next-generation wideband geodetic VLBI and the wideband EHT observations capable of making a movie of the Galactic Center black hole Sgr A*; a more efficient DiFX will speed the roll-out of these new facilities.

Overview of the DiFX code: DiFX is written in C++ and employs Intel Performance Primitives for efficient use of SIMD resources. The correlation process is inherently embarrassingly parallel, and DiFX makes use of two layers of parallelisation: *MPI* to distribute large (~100ms ~= 0.1-1GB) chunks of data across multiple nodes (with double buffering to hide latency) and *pthreads* within a node to facilitate use of multiple CPU cores (which further time-slices down to chunks of ~10ms of data). While the large (~100ms) chunks currently parallelised via MPI could easily fit in modern GPU memory, the current code layout is likely not well suited to a GPU implementation, with a hierarchical object-oriented structure that separates some sequential operations into separate classes.

Project Goals: We aim to get an estimate of 1) how much code re-factoring would be necessary to facilitate a GPU implementation, and 2) how long such a GPU implementation would take. We have previously implemented a stripped-back version of the key DiFX operations and used them to validate a GPU implementation (https://github.com/XhrisPhillips/gcorr), showing that a significant speed-up is possible in principle. However, it is not clear whether a wholesale re-write of the DiFX code would be required to meet the benchmarks shown here, or whether the existing code structure could be used with minimal modifications.

Resource requirements: We already run DiFX on both ozstar (CPU only) and at Pawsey, so continued access to either of these facilities will suffice. Data volumes are minimal, and only standard/common dependencies are needed (e.g. IPP).

---

[1] https://www.atnf.csiro.au/vlbi/dokuwiki/doku.php/difx/documentation, Deller et al., 2011, PASP, 123, 275