A Hybrid Hardware-Software Architecture for Implementation of the CCSDS File Delivery Protocol (CFDP)

Robert Klar, Charlie Howard, Christopher Mangels,
Randal Harmon, and Martin Wasiewicz
Southwest Research Institute

FSW-08, November 2008

CCSDS File Delivery Protocol

- What is it ?
  - CFDP is a protocol for transferring files to and from a spacecraft memory
    - Files can be transferred reliably, where it is guaranteed that all data will be delivered without error, or unreliably, where a best effort delivery capability is provided.
    - Files can be transmitted through at various speeds through different link types and through intermediate relay spacecraft.
    - File transfer can be initiated automatically by the spacecraft or manually by ground control.
    - We plan to initiate science data transfers from the spacecraft

CFDP Service Classes

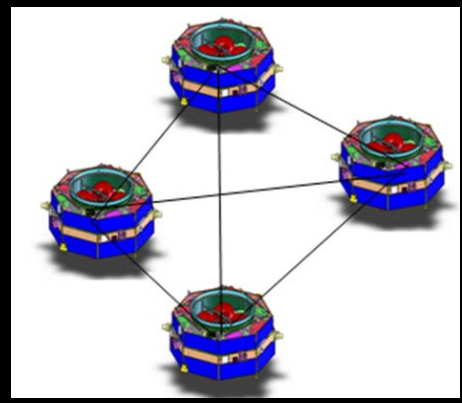
- CFDP supports multiple classes of service
  - If not all of the service classes are needed, then a subset can be implemented for a particular mission.
- Classes of Service
  - Class 1 - Unreliable transfer
  - Class 2 - Reliable transfer
  - Class 3 - Unreliable transfer via one or more waypoints in series
  - Class 4 - Reliable transfer via one or more waypoints in series
- We plan only Class 1 and 2 Service

Why use CFDP?

- Downlink is a precious resource
  - Ground contacts are limited - make the most of them
  - Lost data can mean lost opportunities for science
- Operations costs are a significant part of the mission budget
  - CFDP helps to automate routine data delivery operations to simplify operations, thereby providing some promise for reducing these costs
- Standard protocol encourages interoperability
  - Benefit realized as CFDP is adopted on more missions
- Class 3 and 4 service provides potential benefits for deep space missions
  - CFDP supports store-and-forward operation through waypoints

Motivation - MMS

- Magnetospheric MultiScale (MMS) Mission
  - Constellation of 4 identically spacecraft in variably spaced tetrahedron (1 km to several R_E)
    - Ground contacts must be multiplexed in time in order to retrieve data from all 4 spacecraft each day

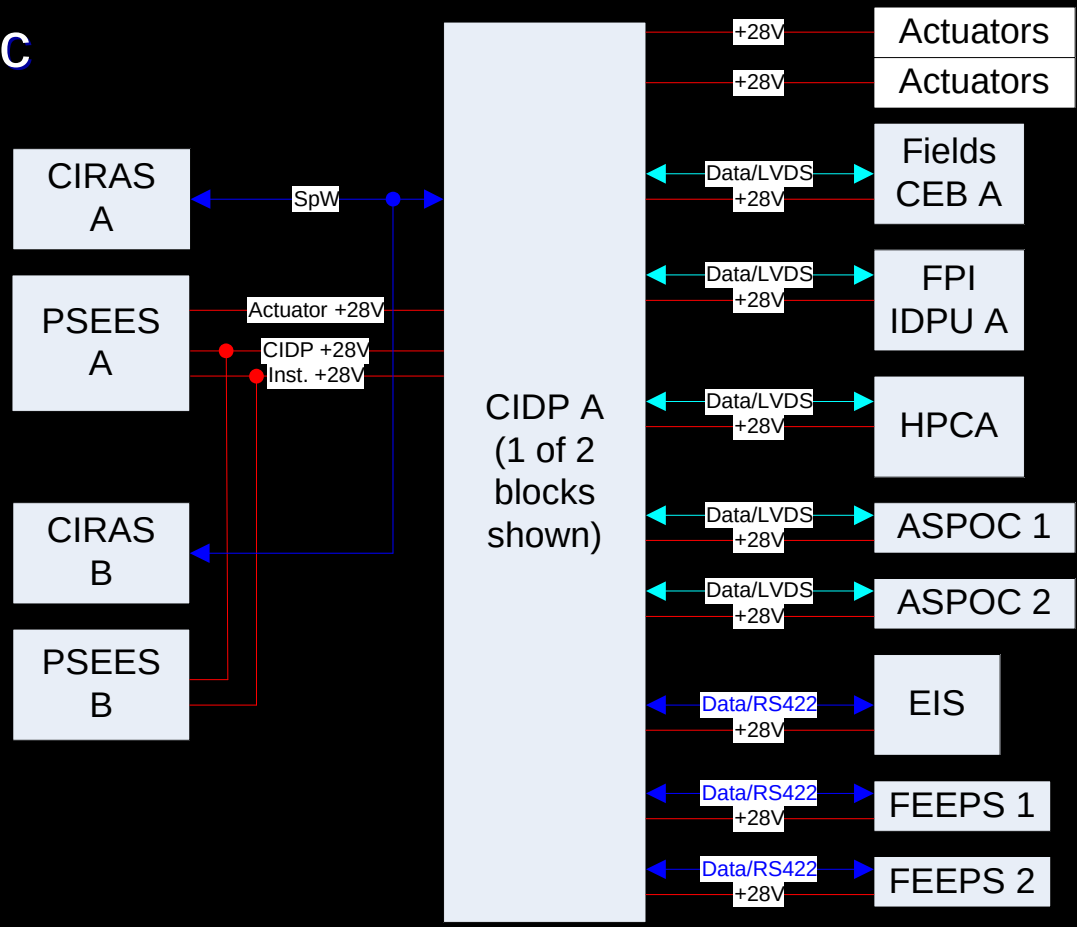


  - Overall Objectives: To discover the detailed physics of the reconnection process including its controlling factors, its spatial distribution, and its temporal behavior.

http://stp.gsfc.nasa.gov/missions/mms/mms.htm

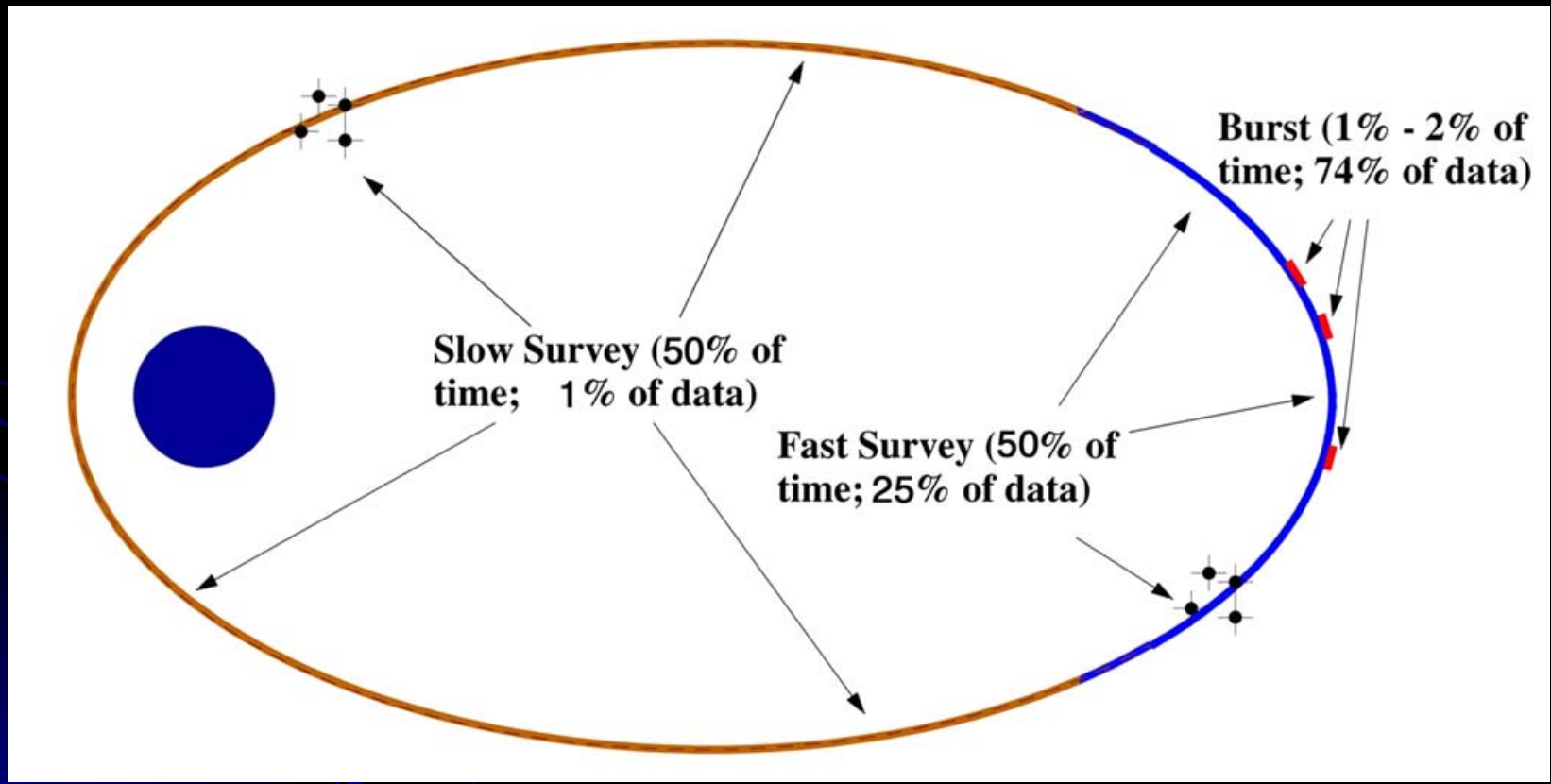
Motivation - MMS

- Solving Magnetospheric Acceleration, Reconnection, and Turbulence (SMART) Instrument Suite
  - Central Instrument Data Processor (CIDP)
    - SPARC Processor
    - Stores science data for the Instrument Suite
    - **Instruments can collect far more data than can be sent via downlink**

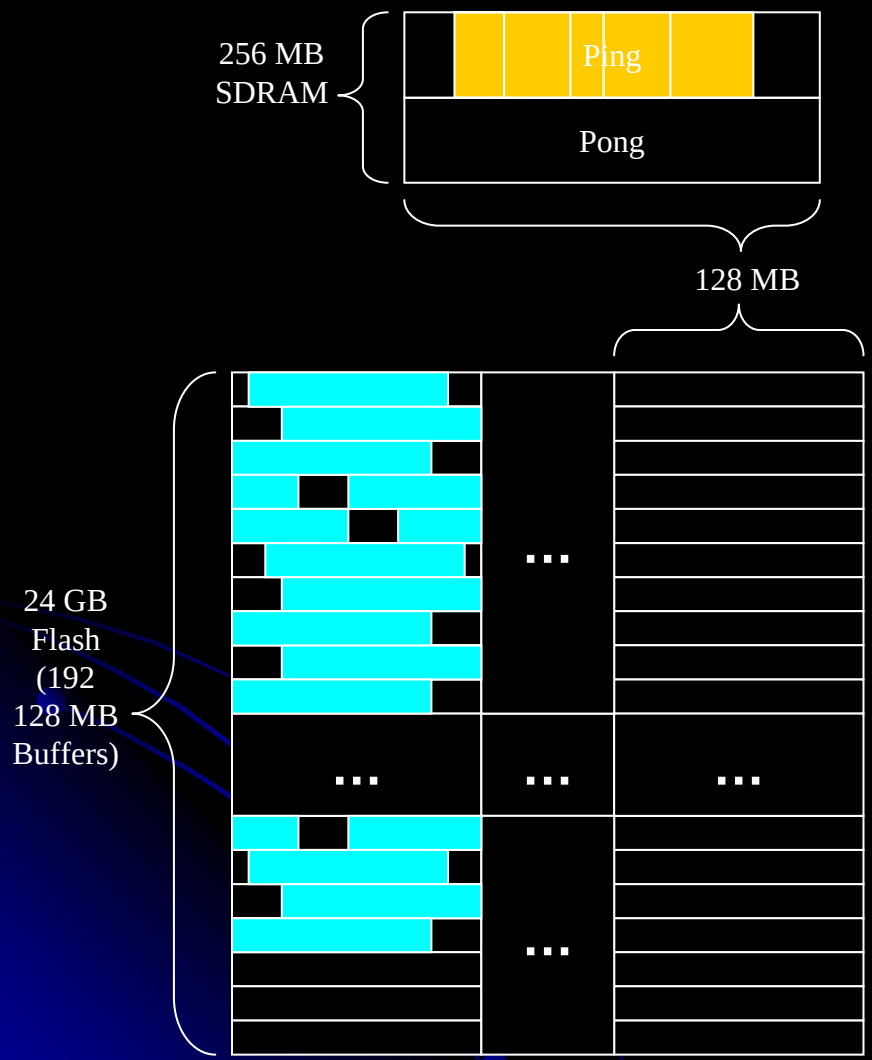


Motivation - MMS

- Highest resolution data is voluminous
  - Collect best quality data for region of interest



Implementation



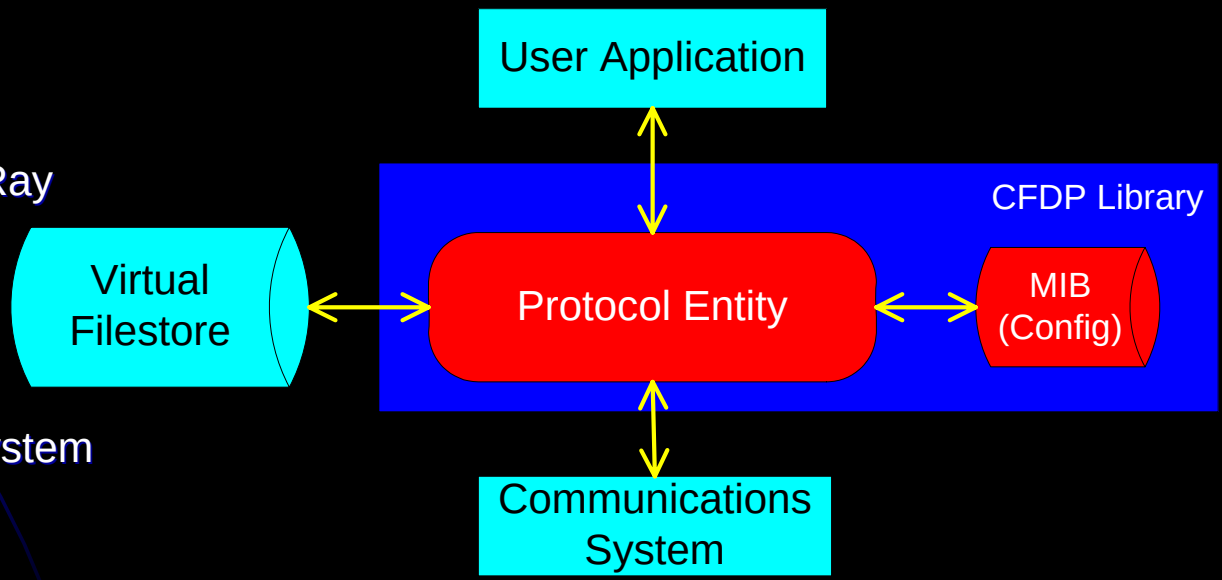
- CFDP Virtual Filestore is CIDP Mass Memory
  - Partitioned into 128 MB fixed-size buffers
    - Correspond to Files for CFDP
  - Ping-pong buffers in SDRAM
    - Each is a circular buffer
  - Buffer Replacement Algorithm
    - **Continuous competition in region of interest**
    - **Lowest quality buffers are replaced**
  - Flash buffers managed in pools
    - In-use
    - Free
    - Offline
      - **Available for downlink**
      - **Downlink may occur while in region of interest**

Context

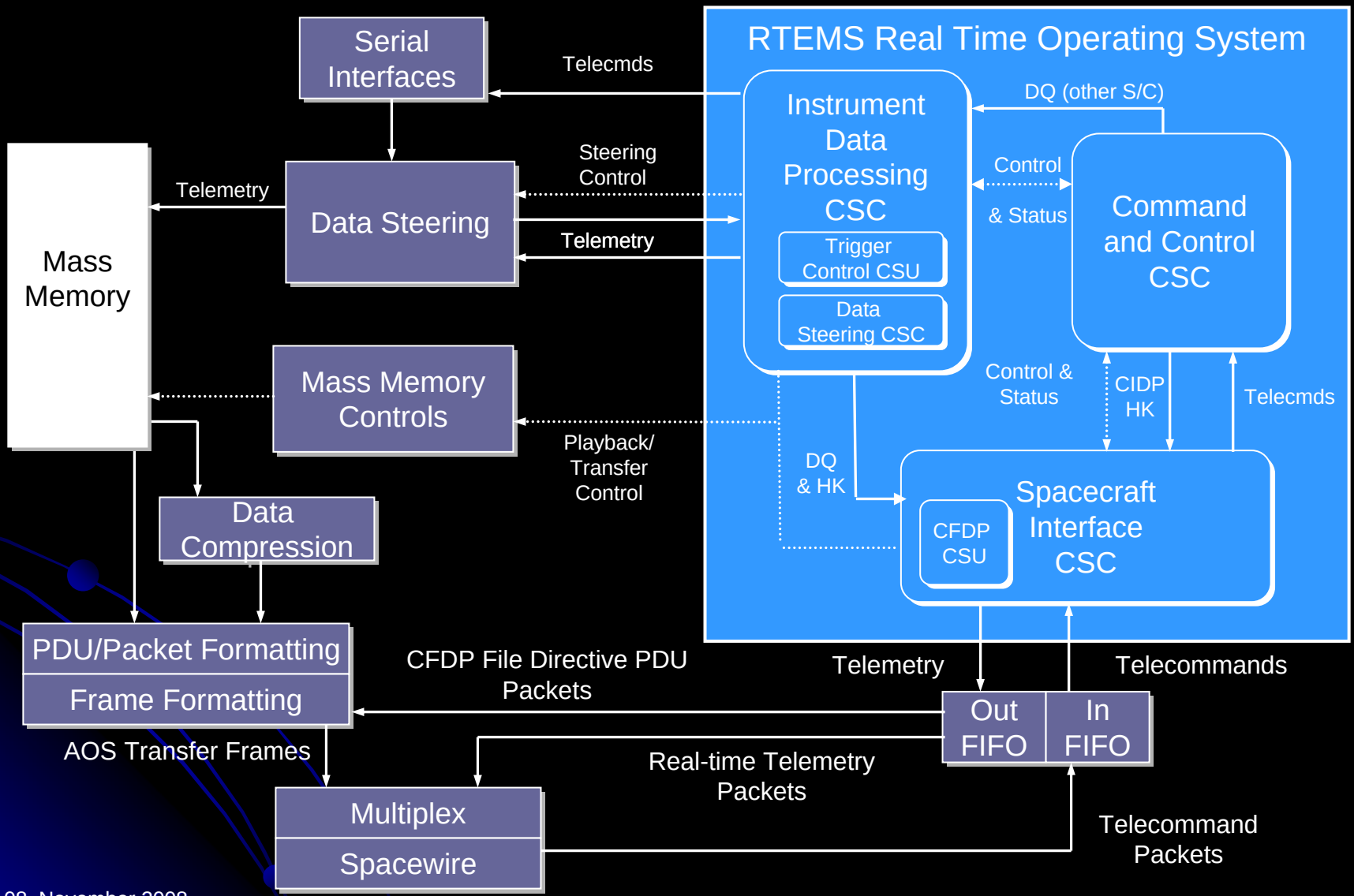
- CFDP can operate over a wide-range of underlying communications protocols
  - CCSDS Space Packet Protocol
  - **Class 2 (Reliable) with Deferred NAK**
  - Class 1 (Unreliable) is backup

CFDP Procedures				
Proximity-1	CCSDS Packet Service	SCPS-TP	TCP	UDP
	CCSDS Transfer Frame Service	SCPS-NP	IP	
Physical Communications Channel				

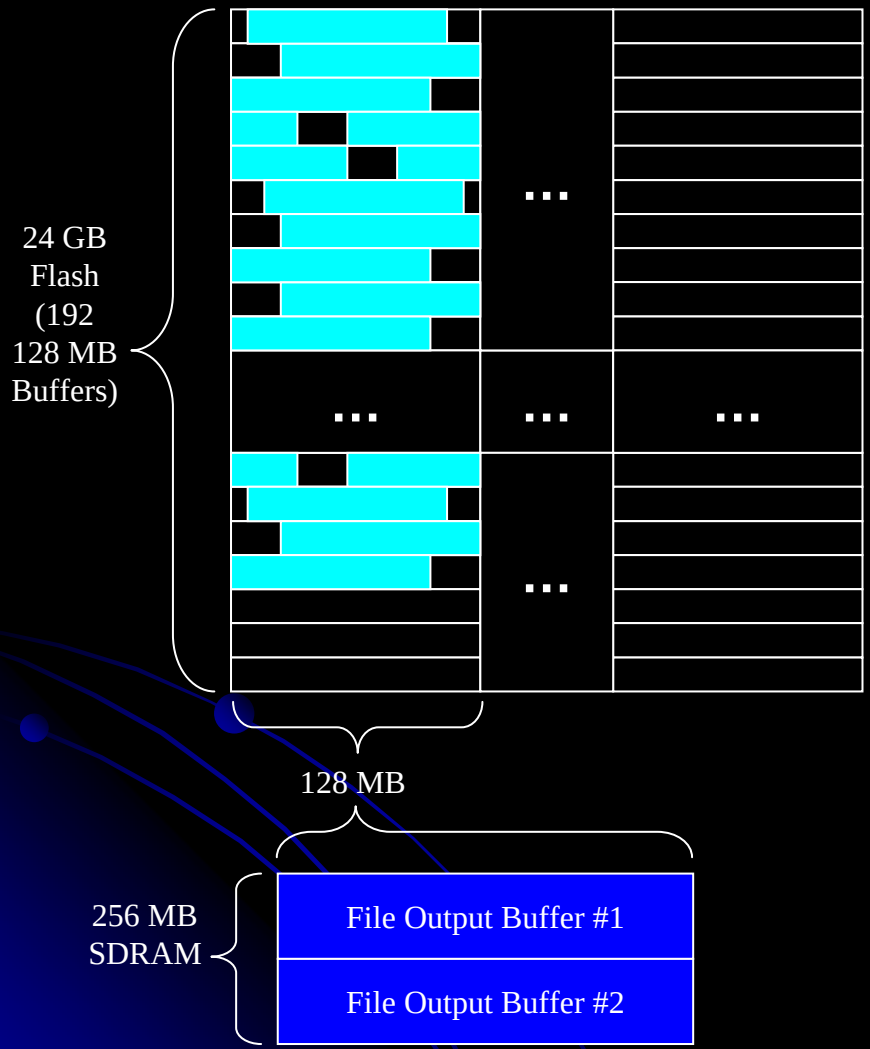
- CFDP Library
  - Implements State Machines
  - Developed by Tim Ray of NASA GSFC
  - Interfaces to User Application, Virtual Filestore and Communications System



Hardware/Software Interface



Virtual Filestore for CFDP



- Output of Buffers
  - Buffers are transferred to output buffers for CFDP formatting
    - Indexed to support retransmission of lost PDUs
    - 2 Output buffers allow for up to two files open at the same time
  - Buffers represent files
    - Compressed Files
      - File is compressed into an output buffer
    - Uncompressed Files
      - Buffer is simply copied to output buffer

Software Interfaces

- Application Interface
  - Calls to Library
    - Set Configuration (i.e. MIB) Dynamically
    - Give Request to Library
    - Give PDU to Library
    - Control delivery rate for PDUs
      - **Function invoked to send next data PDU**
    - Register callbacks for Indication
  - Callbacks from Library
    - CFDP Indications
- Virtual Filestore Interface
  - Callbacks from library
    - File I/O primitives
      - fopen(), fseek(), fread(), fwrite(), feof(), fclose()
    - Other Posix primitives
      - rename(), remove(), tmpnam()
- Communication System Interface
  - Callbacks from Library
    - Open PDU channel
    - Check Channel Ready?
    - Send PDU on channel

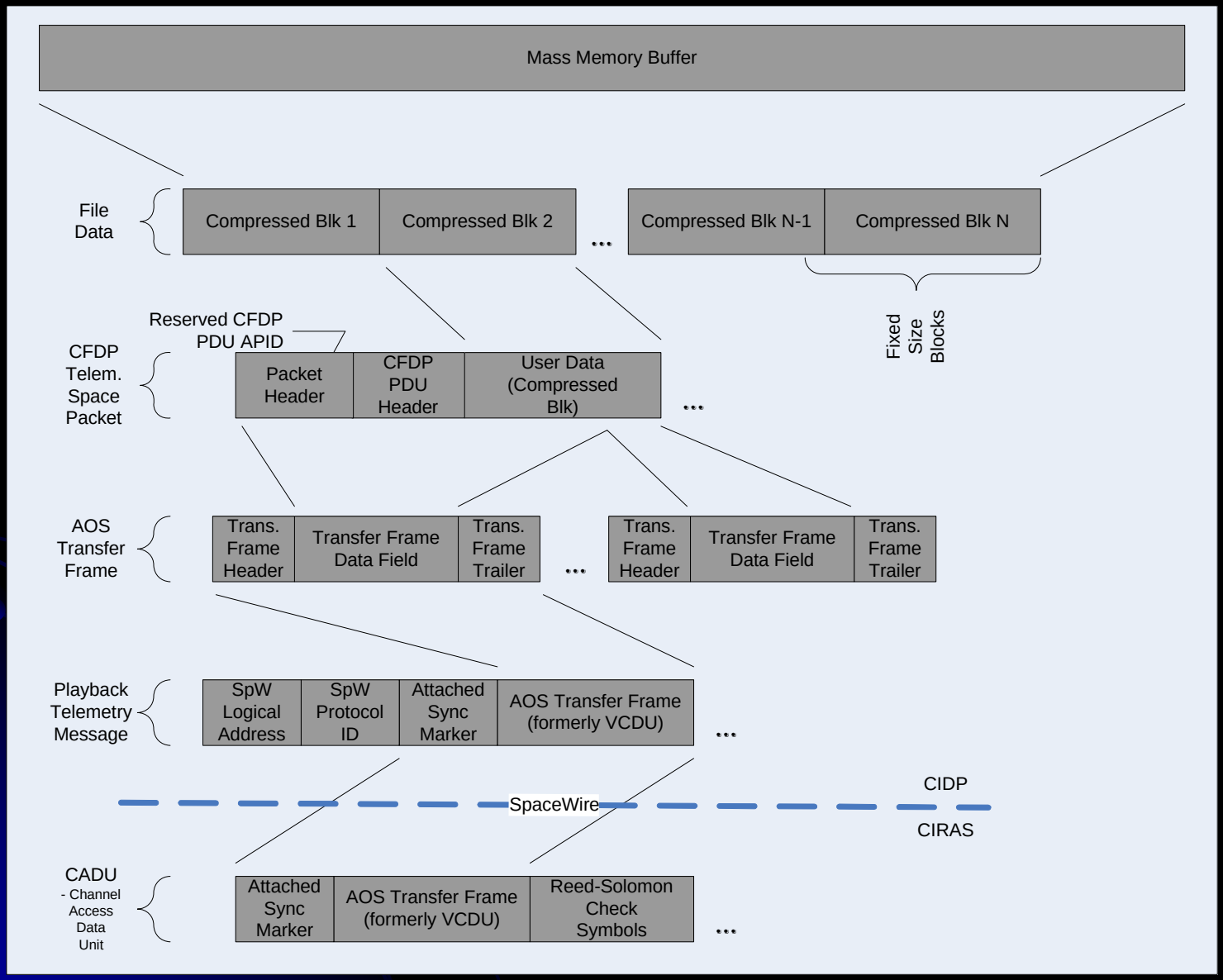
Implementation

- Virtual Filestore Interface
  - Callbacks from library
    - File I/O primitives
      - fopen() - transfer file to Output Buffer and optionally perform compression
        - Set start offset to 0
        - Return corresponding buffer index
      - fread()
        - **If first call after fopen() or fseek(), then initiate HW playback**
        - Simply advance read count
      - fseek()
        - Set HW start offset for retries
      - fwrite()
        - Not allowed
- Communication System Interface
  - Callbacks from Library
    - Open PDU channel
      - Initiate HW handshaking
    - Check Channel Ready?
      - HW ready if SpW link is up
    - Send PDU on channel
      - Send File Directive PDUs to Data Formatter to include in VC stream
      - Ignore File Data PDUs – these are being generated by the

Implementation

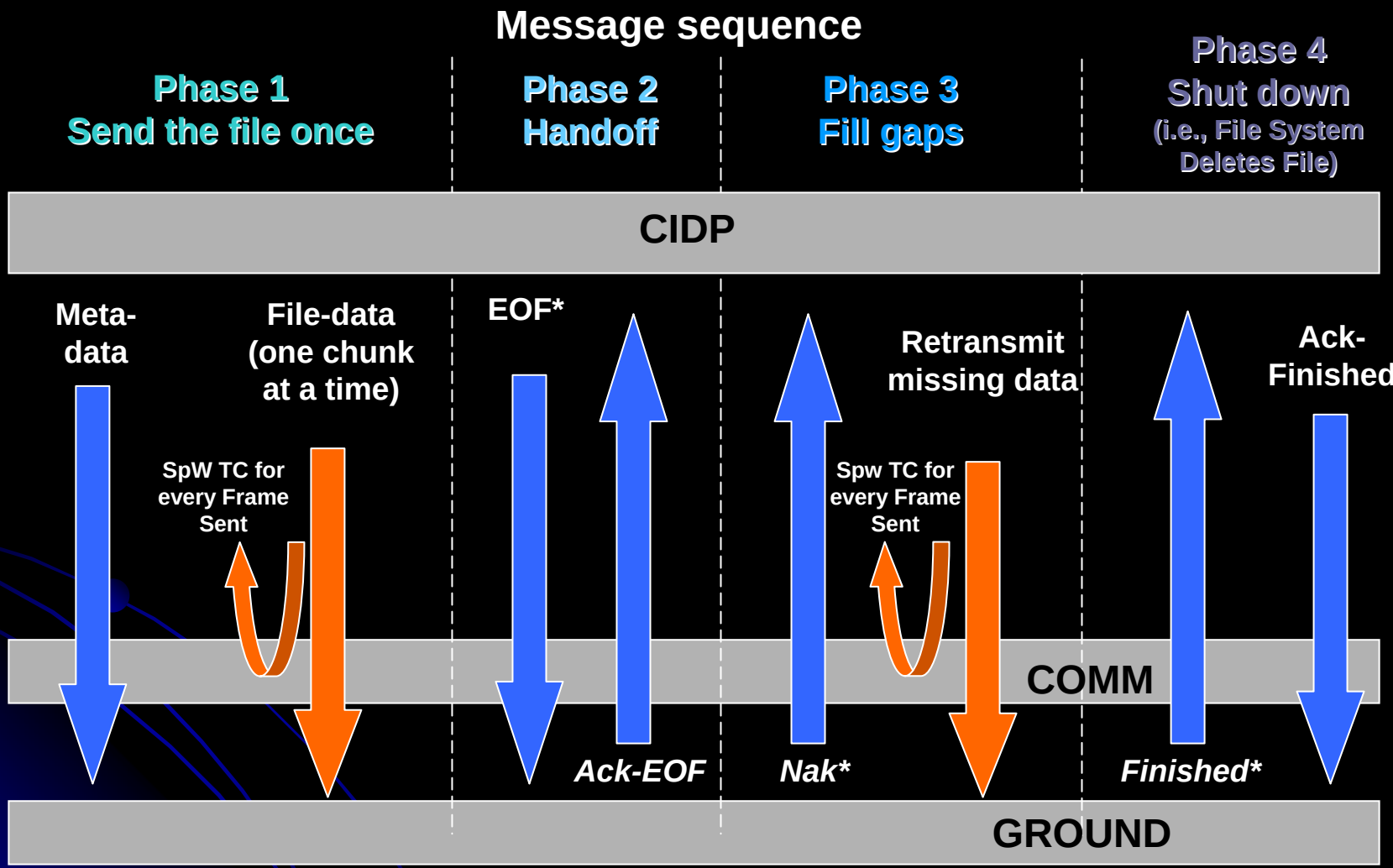
- Use Virtual Filenames
  - Mass Memory Buffer Index + Timestamp
- Hardware forms File Data PDUs
- Software forms and responds to File Directive PDUs
- Hardware/Software Synchronization
  - Hardware provides for configuration of start and end offsets in Mass Memory Buffer
    - Aligned on segment boundaries (i.e. ~ 1 KB)
  - Once playback is initiated, HW transfers File Data PDUs until completed
    - Interrupt wakes up SW task when a number of segments have been transferred
    - SW task calls CFDP Library rate control function in a loop to match up file read pointer with those segments sent by HW

Formatting



- Buffers are segmented to fit nicely into a CCSDS PDU, Telemetry Packet, and Transfer Frame

Hardware Operation



** Timers are used to ensure retransmission of the EOF, Nak, and Finished messages as required. The Nak message reports all missing data (including Metadata).*

Central Instrument Data Processor

Questions?