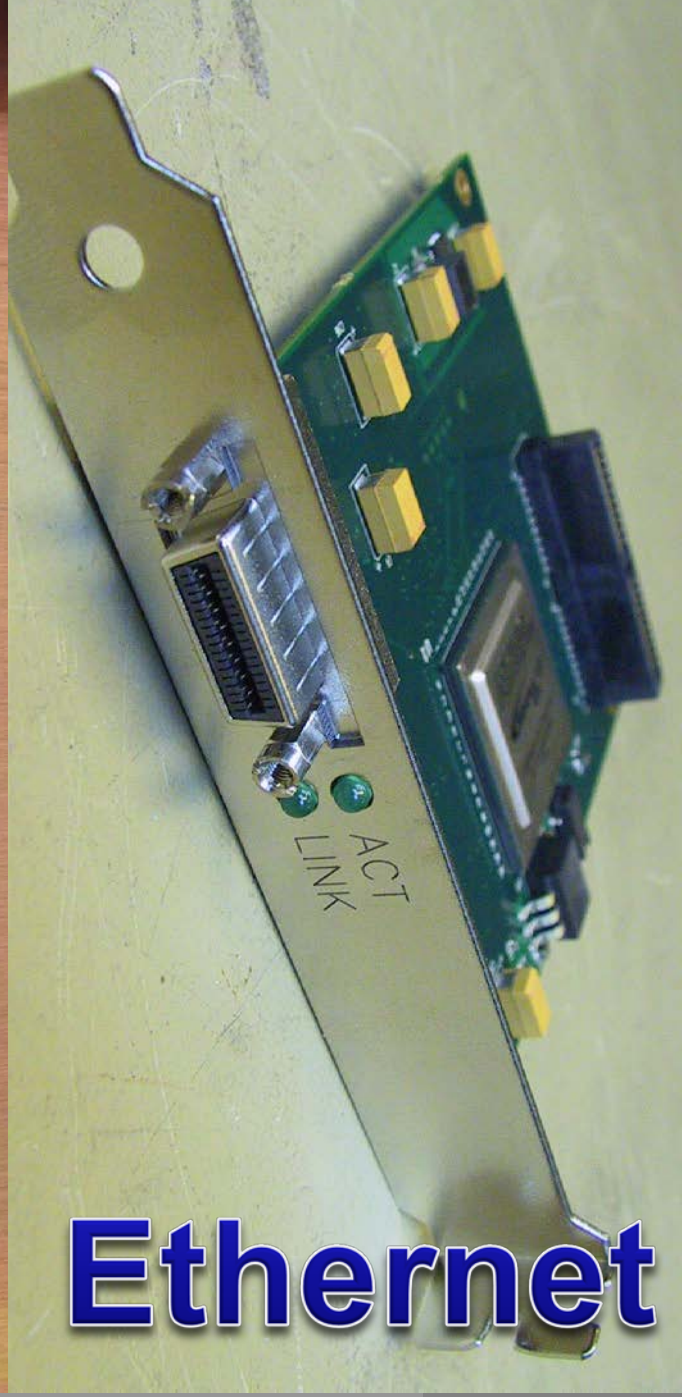
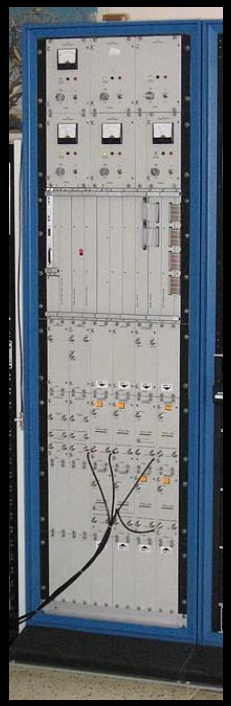
RS422

VSI/H



Ethernet

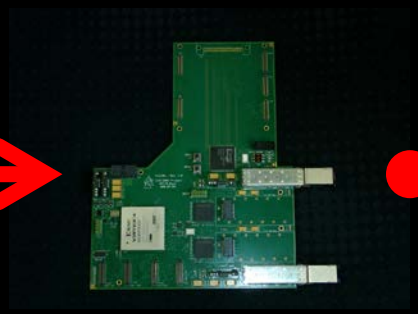

formatter
recorder



MarkIV
VLBA

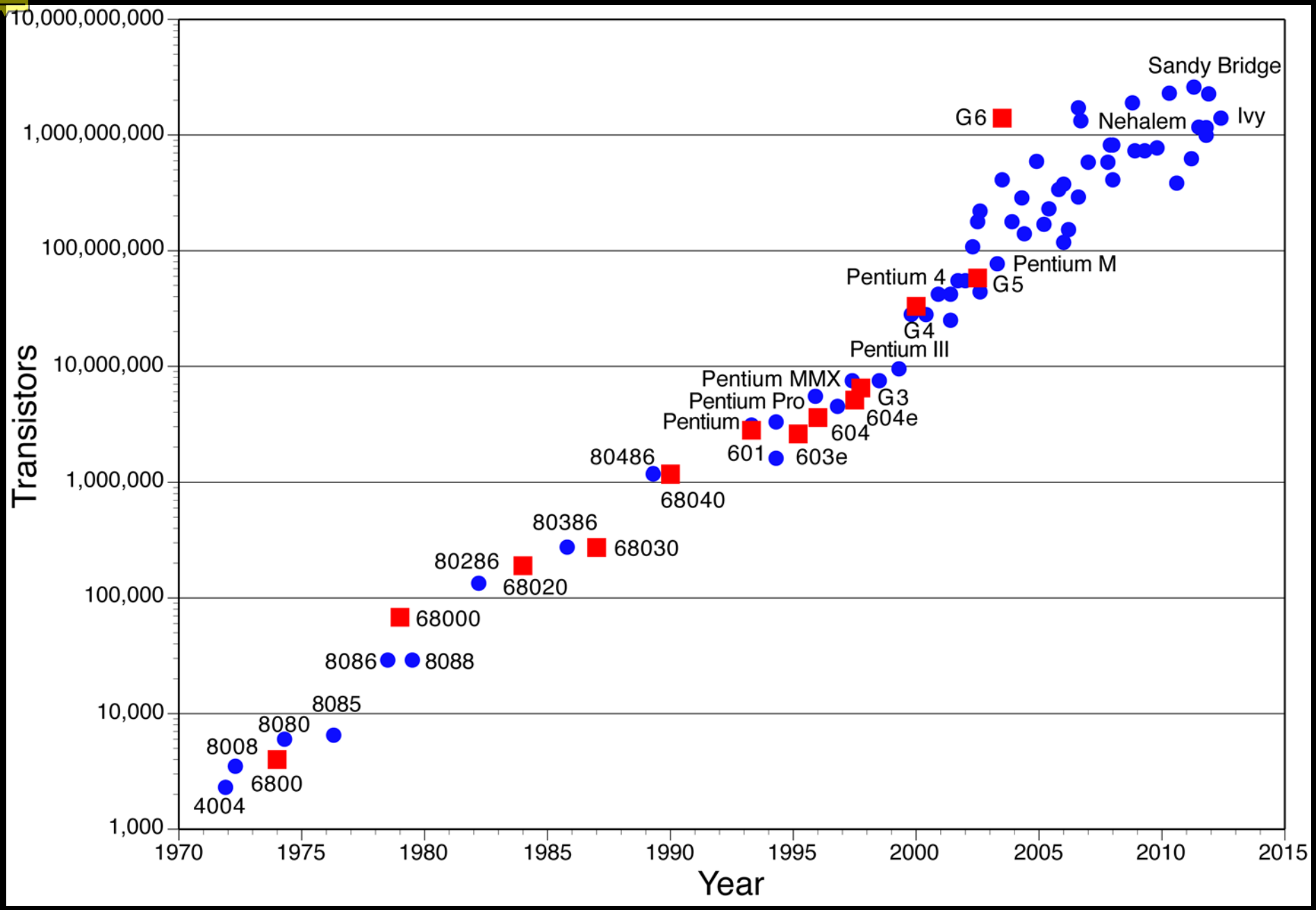
RS422

Mark5A



DBBC → VSI/H → FiLa10G → ethernet → Mark5C / Mark6 / FlexBuff

RS422	MarkIV VLBA	**Analog**
VSI/H	DBBC[2] R1002 CDAS ADS[123]000	**Digital**
Ethernet	DBBC[2]+FiLa10G R[2]DBE CDAS2 BRAS	**Digital**


Transistors
Year
10,000,000,000
1,000,000,000
100,000,000
10,000,000
1,000,000
100,000
10,000
1,000
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Sandy Bridge
Ivy
Nehalem
G6
Pentium M
Pentium 4
G5
G4
Pentium III
Pentium MMX
G3
Pentium Pro
604e
Pentium
604
601
603e
80486
68040
80386
68030
80286
68020
68000
8086
8088
8085
8080
8008
6800
4004

RS422	MarkIV VLBA	**512 Mbps**
VSI/H	DBBC[2] R1002 CDAS ADS[123]000	**2048 Mbps**
Ethernet	DBBC[2]+FiLa10G R[2]DBE CDAS2 BRAS	**8192 Mbps**

RS422	MarkIV VLBA	**Mark5A**
VSI/H	DBBC[2] R1002 CDAS ADS[123]000	**Mark5B(+)**
Ethernet	DBBC[2]+FiLa10G R[2]DBE CDAS2 BRAS	**Mark5C** **Mark6** **FlexBuff**

How to get all those bytes to disk?

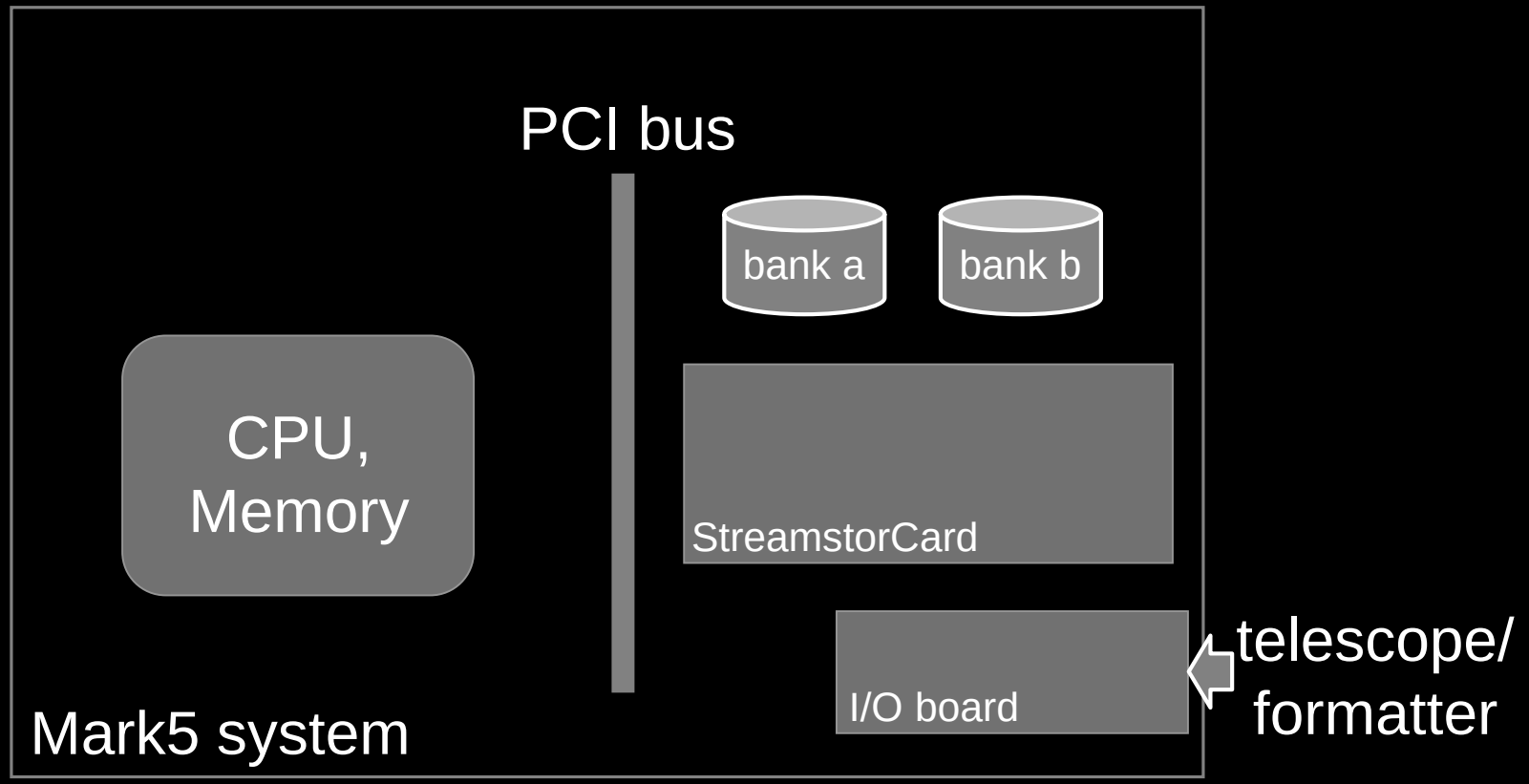
Flexing your buffs

and Marking your 6's

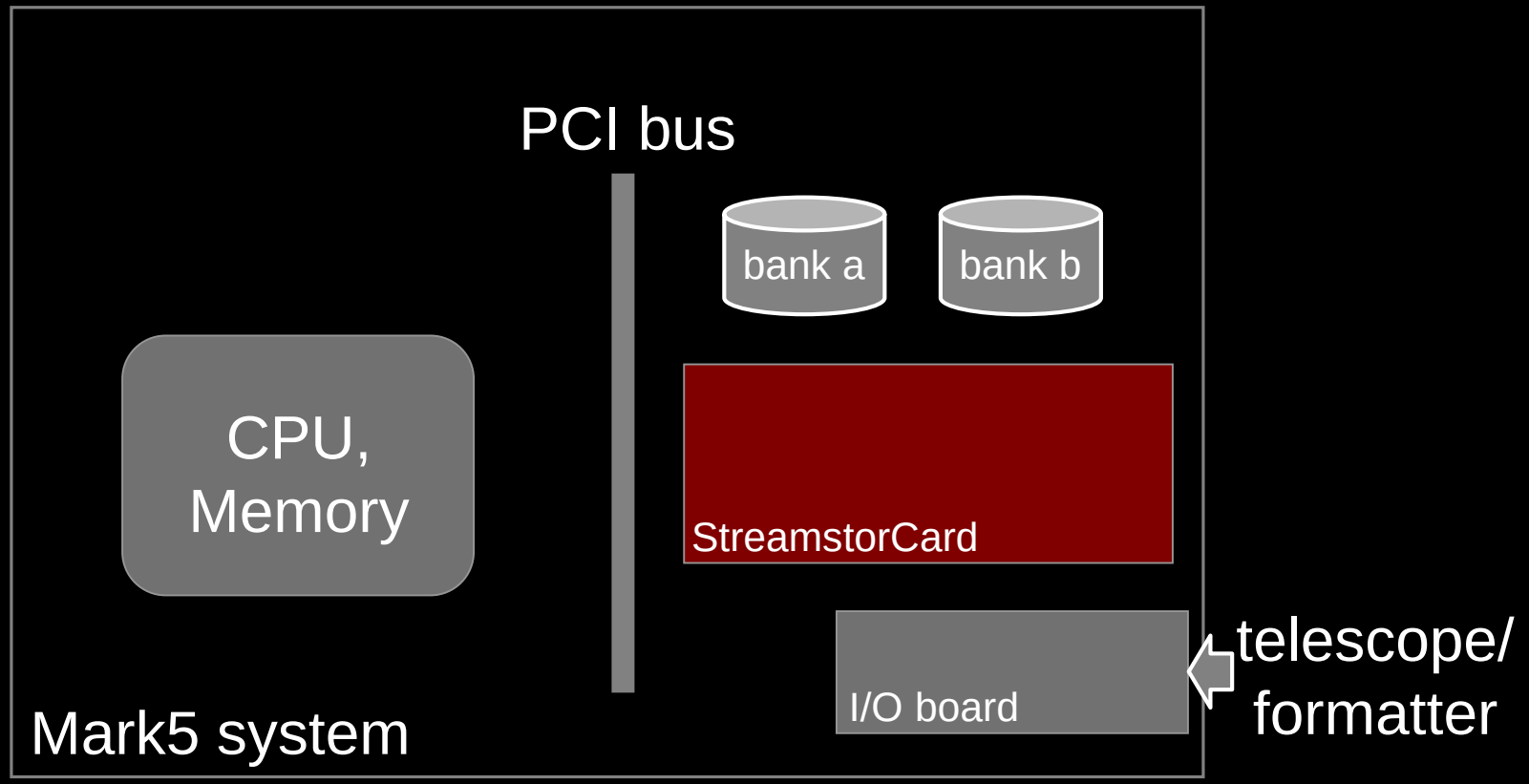
JIVE
Joint Institute for VLBI ERIC

Harro Verkouter

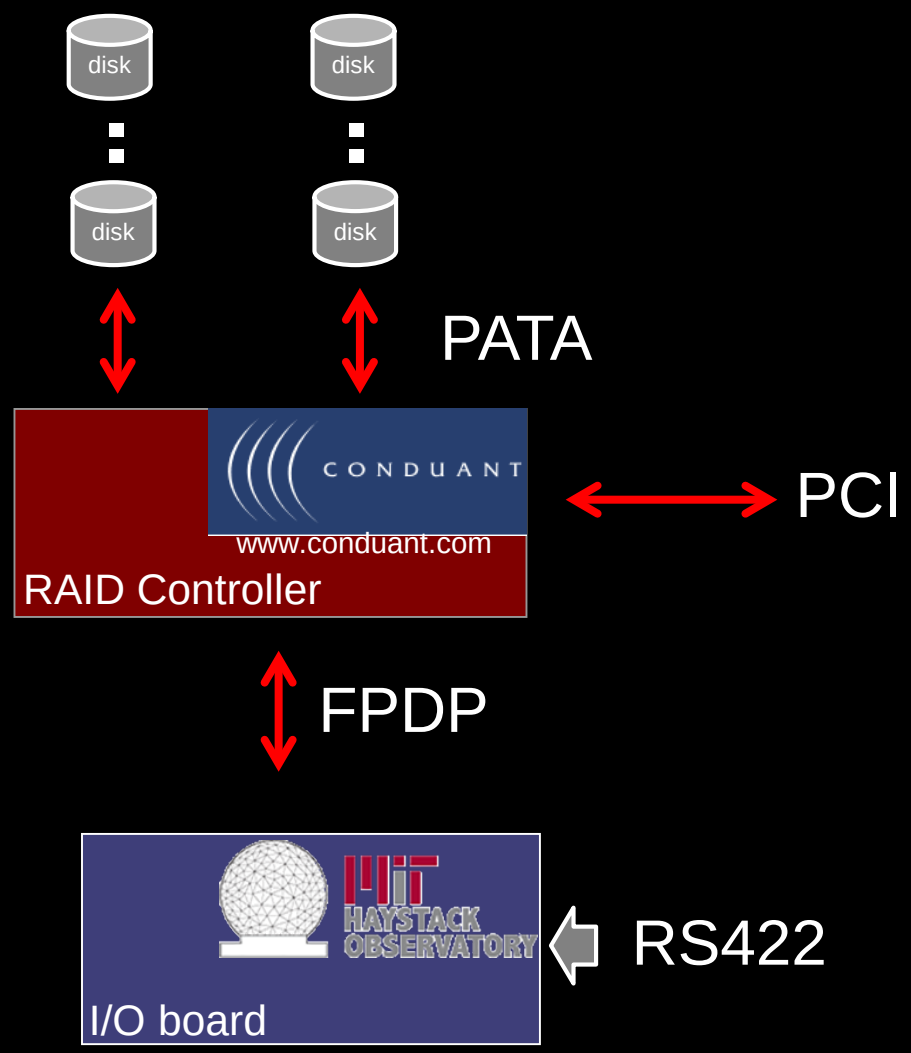
In the beginning ...



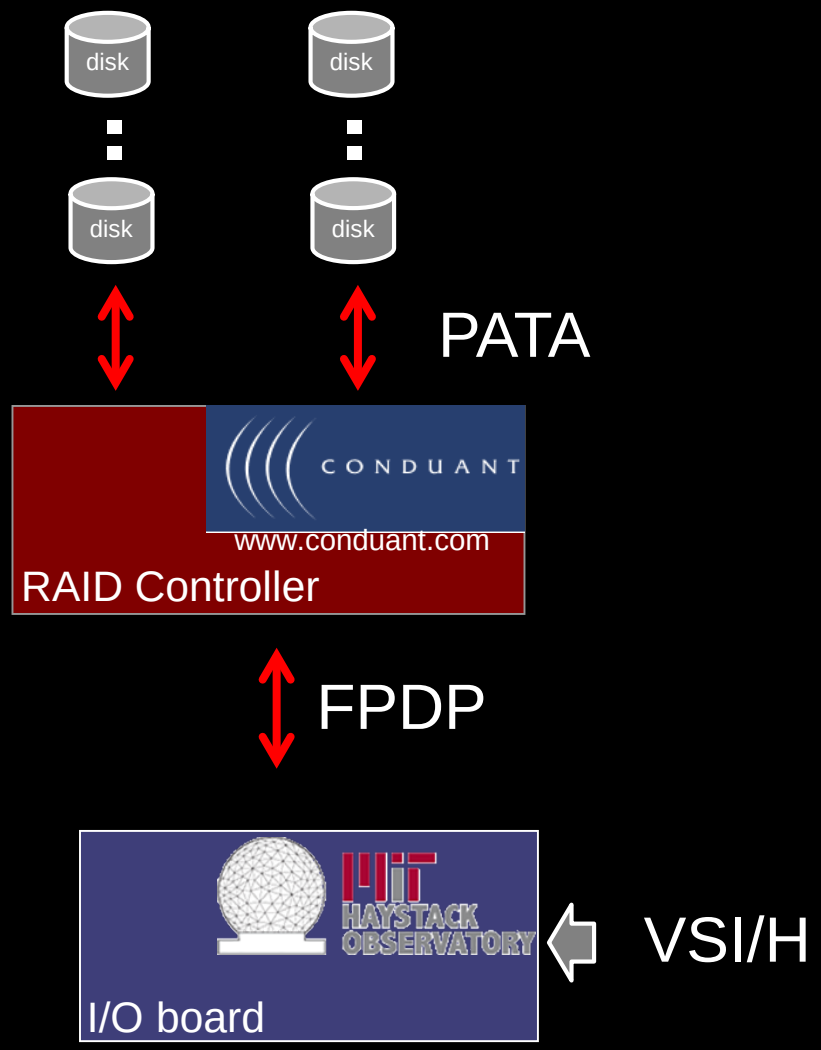
In the beginning ...



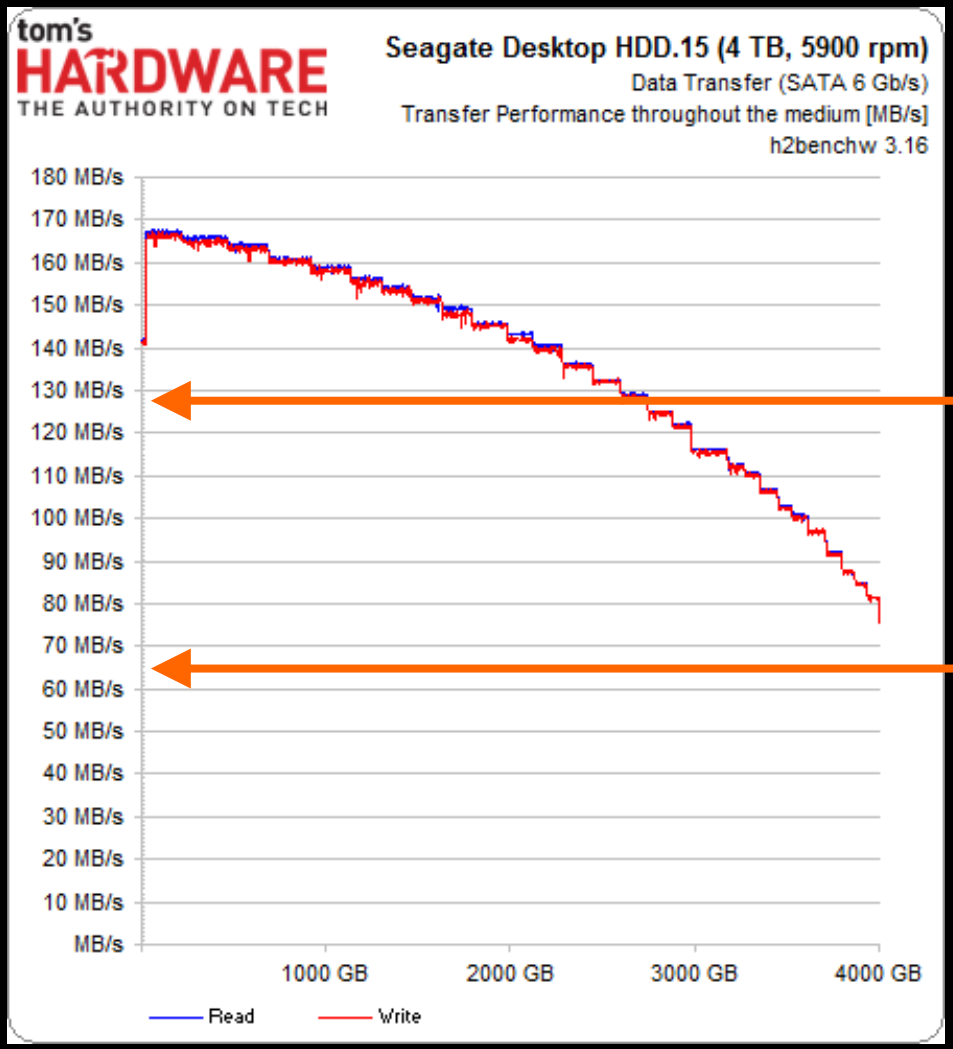
In the beginning ... (5A)



In the beginning ... (5B)



These days ...

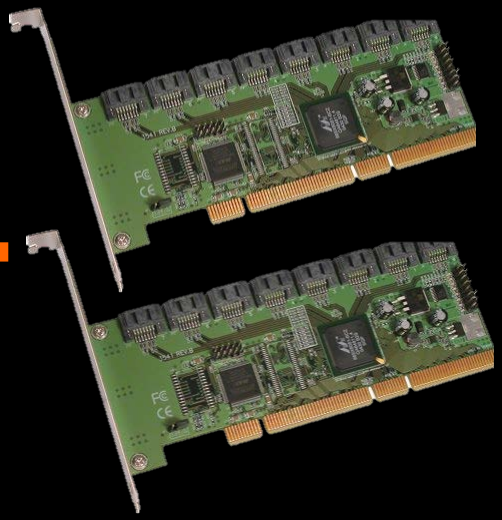


1024Mbps

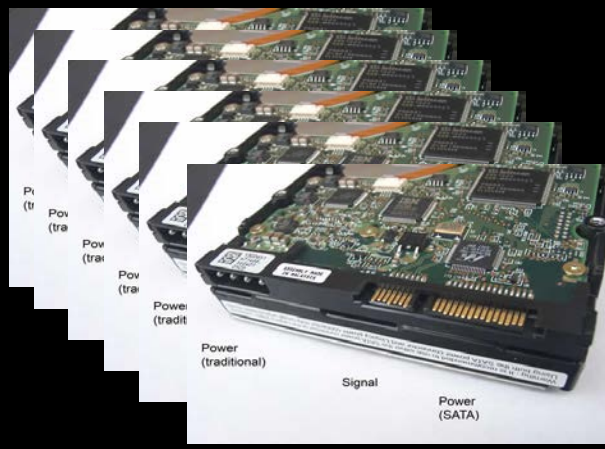
512Mbps

These days ...

+

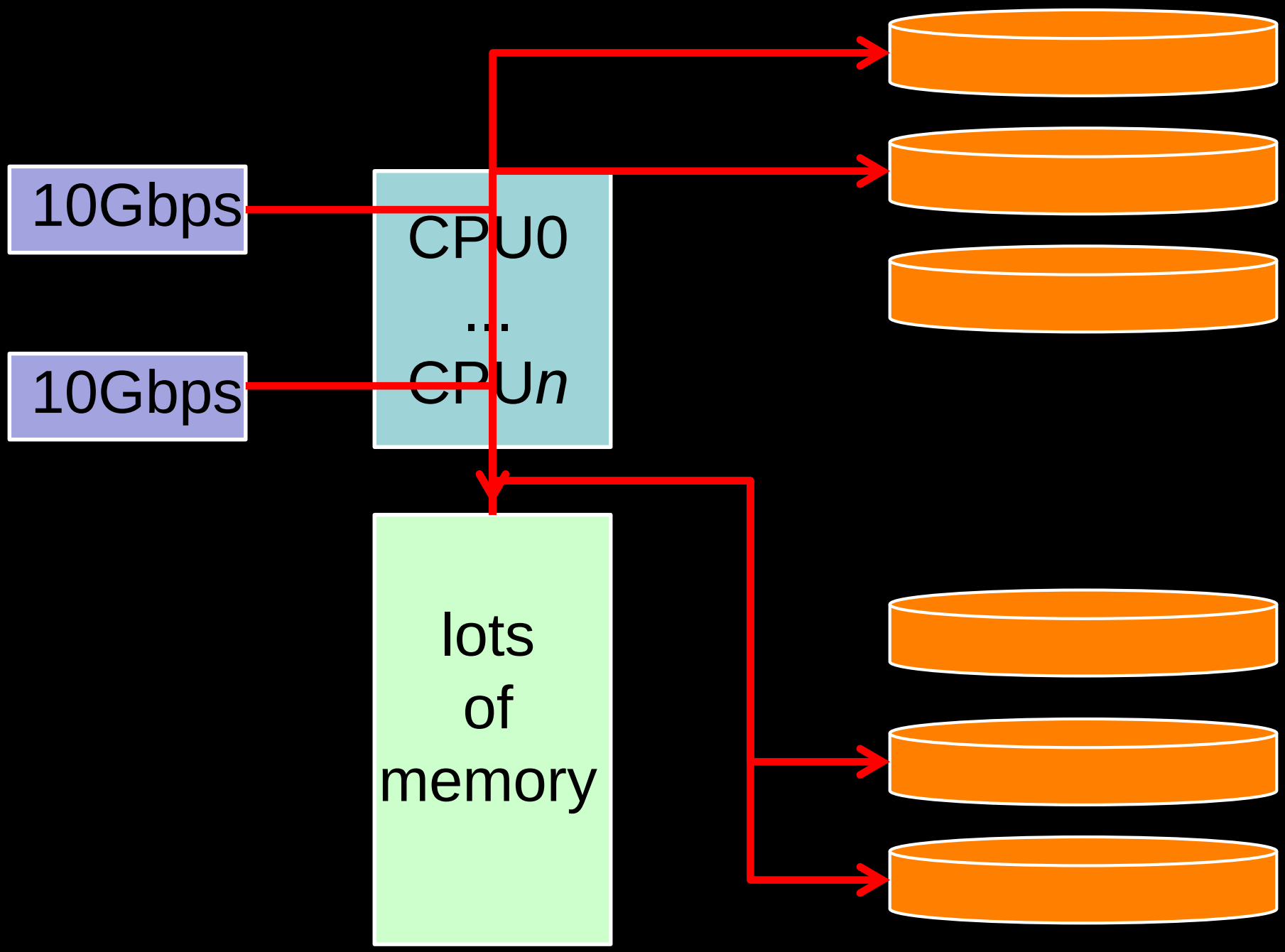


+




Mark6
CONDUANT
Module Status
Present
Power
Ready
Fault
Power
FILA 10G - SA


transtec
FlexBuff


10Gbps
10Gbps
CPU0
...
CPU*n*
lots
of
memory

What *are* the differences?

Mark6	FlexBuff
4 removable disk packs	fixed disks

All other differences are SOFTWARE!

	MIT Haystack	EVN
Mark5A	Mark5A	jive5ab
Mark5B	DIMino	jive5ab
Mark5C	drs	jive5ab
Mark6	cplane/dplane	jive5ab
FlexBuff	-	jive5ab

What *are* the differences?

Mark6	**FlexBuff**
/mnt/disks/...	/mnt/disk...
striped in single file/disk	striped in file per block
extra headers in stream	just the facts, ma’m!

Mountpoints

Mark6	FlexBuff
/mnt/disks/1/0	/mnt/disk0
... /1	/mnt/disk1
... /2	/mnt/disk2
... ...	/mnt/disk3
... /7	
/mnt/disks/2/0	
... /1	.....
... /2	
... ...	
... /7	
/mnt/disks/3/0	/mnt/disk31
... ...	

Mark6+d-plane

10Gbps

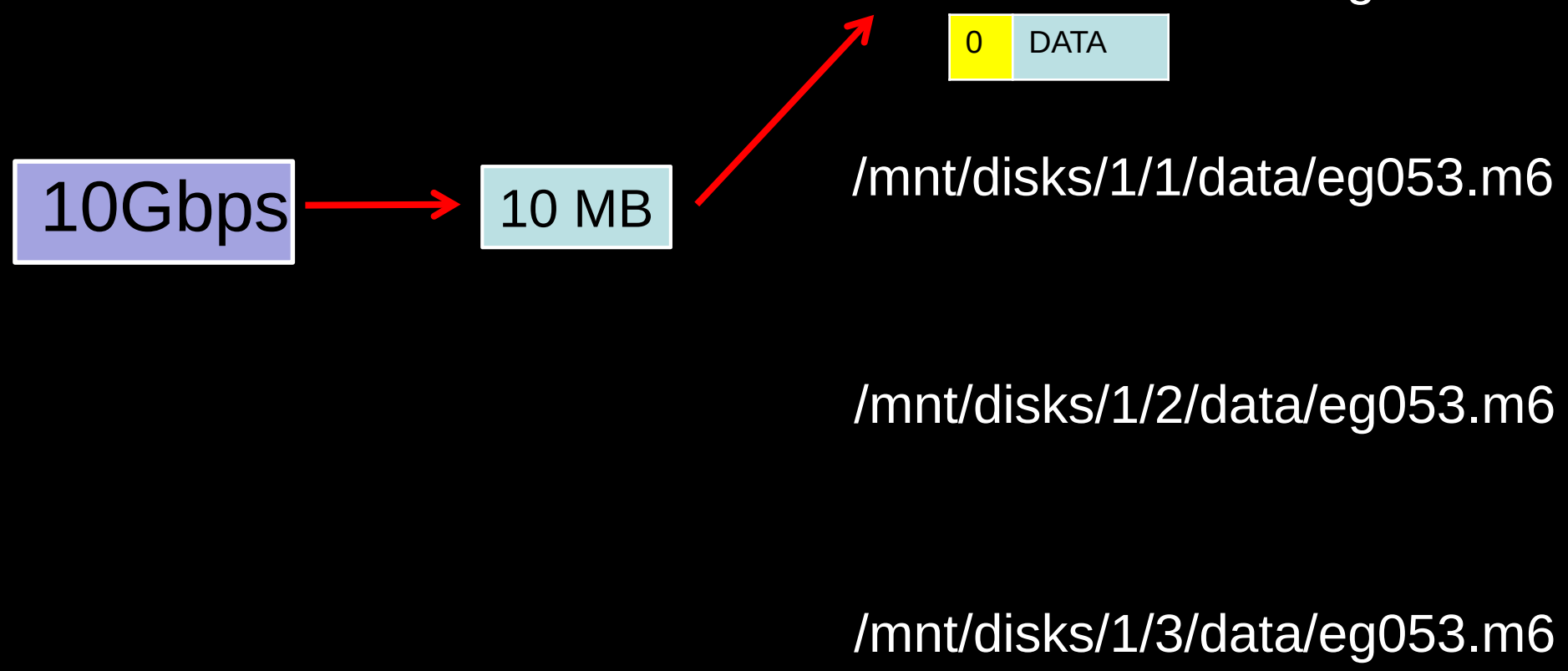
/mnt/disks/1/0/data/eg053.m6

/mnt/disks/1/1/data/eg053.m6

/mnt/disks/1/2/data/eg053.m6

/mnt/disks/1/3/data/eg053.m6

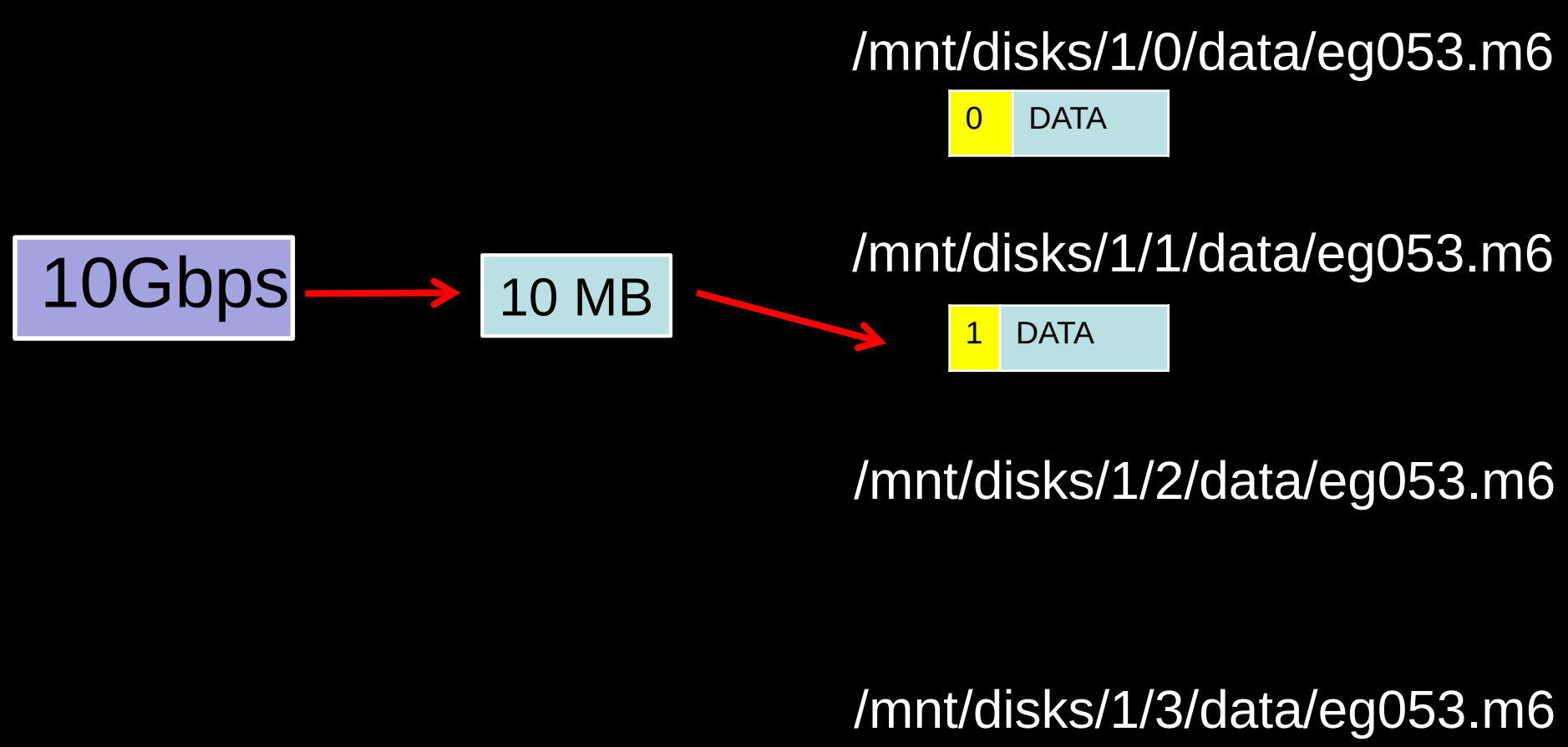
Mark6+d-plane



recording application header

VLBI data

Mark6+d-plane



recording application header

VLBI data

Mark6+d-plane

10Gbps

/mnt/disks/1/0/data/eg053.m6

0	DATA	6	DATA

/mnt/disks/1/1/data/eg053.m6

1	DATA	7	DATA

/mnt/disks/1/2/data/eg053.m6

2	DATA	5	DATA

/mnt/disks/1/3/data/eg053.m6

3	DATA	4	DATA

recording application header

VLBI data

FlexBuff+jive5ab

10Gbps

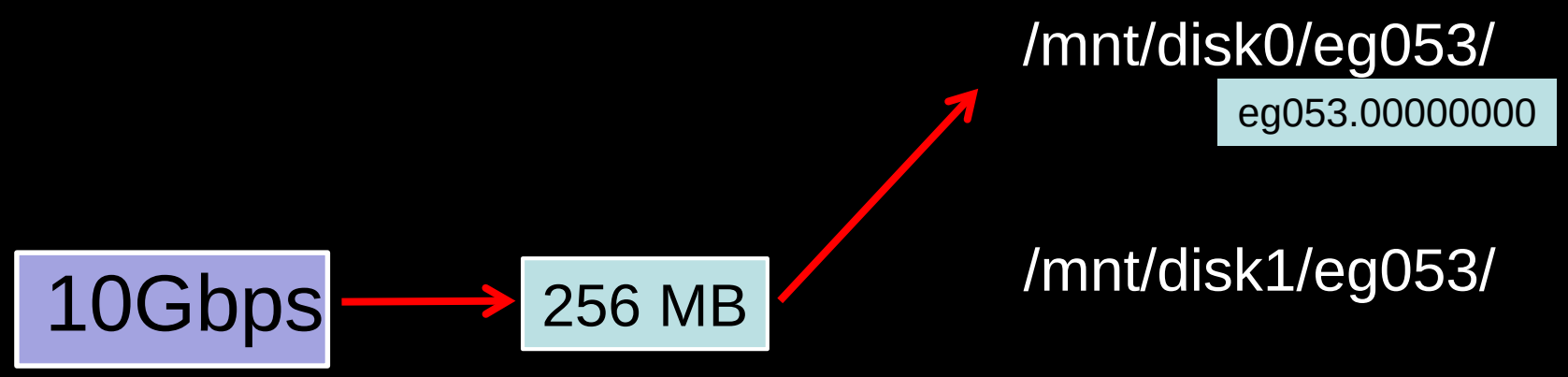
/mnt/disk0/eg053/

/mnt/disk1/eg053/

/mnt/disk2/eg053/

/mnt/disk3/eg053/

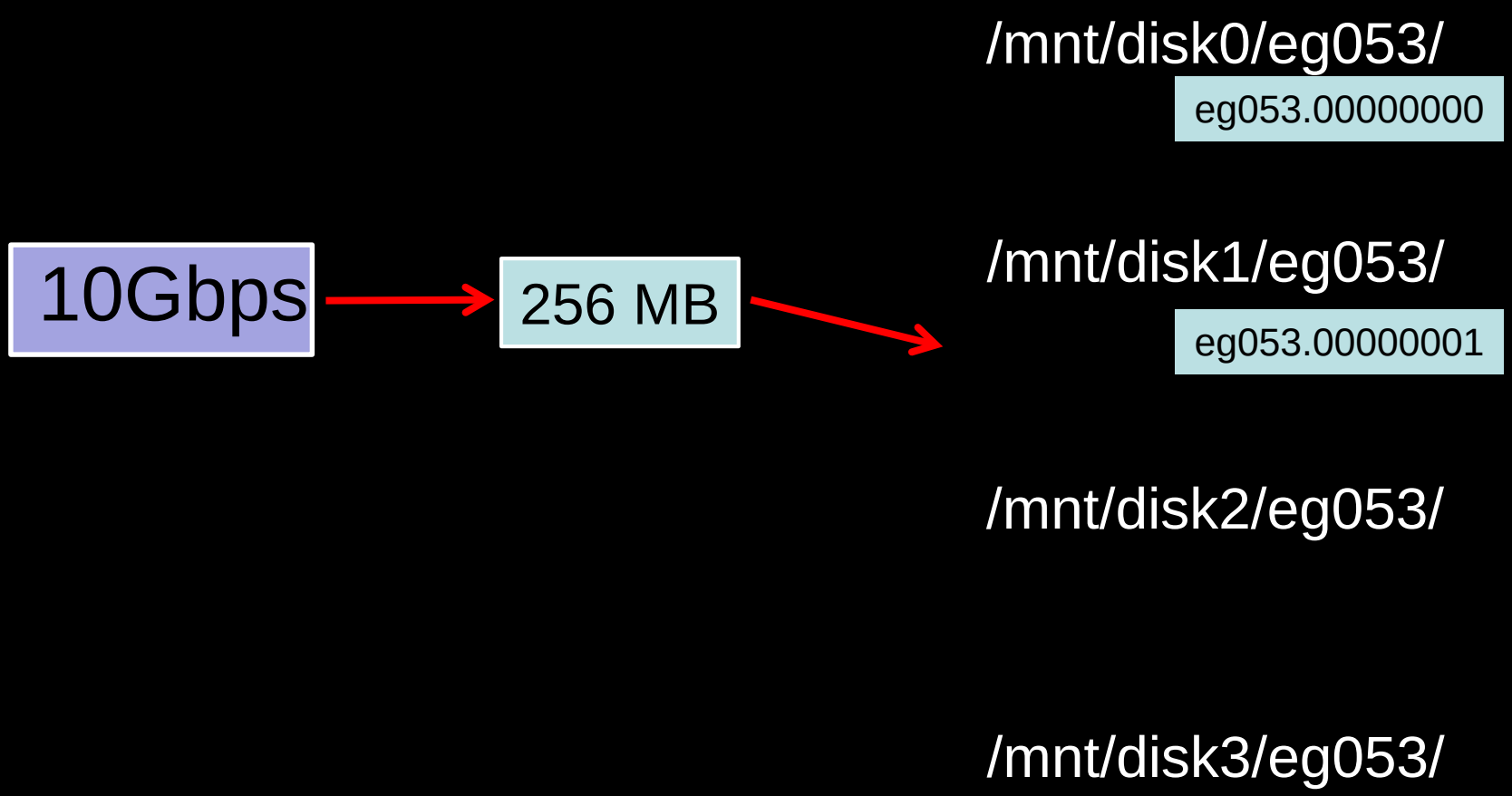
FlexBuff+jive5ab



recording application header

VLBI data

FlexBuff+jive5ab



recording application header

VLBI data

FlexBuff+jive5ab

10Gbps

/mnt/disk0/eg053/

- eg053.00000000
- eg053.00000006

/mnt/disk1/eg053/

- eg053.00000001
- eg053.00000007

/mnt/disk2/eg053/

- eg053.00000002
- eg053.00000005

/mnt/disk3/eg053/

- eg053.00000003
- eg053.00000004

recording application header

VLBI data

Still very similar!

- capture block
- assign sequence number
- write to next available disk

Still very similar!

- where to stripe data?
- in which format?

jive5ab commands for FlexBuff/Mark6

Where to stripe data?

- `set_disks = /path/to/disk* [ : more ]`
- `set_disks = [1234]+ | eMSN [ : more ]`
  - Mark6 disk module number(s) or eMSN(s)
- `set_disks = ^(foo|bar)[^0-9]$ [ : more ]`
  - full regular expression support

Set recording format

- `record = mk6 : 0` (FlexBuff `vbs` format)
- `record = mk6 : 1` (`d-plane v2` format)

How is data captured?

d-plane	jive5ab
"`tcpdump -ni ethX`" udp only, extract bytes from pkt	(IPv4:)PORT sockets all supported protocols: udp, vtp, tcp, udt

Supported data formats

d-plane	jive5ab

Assumes VDIF/Mark5B
`packet = ...(*)`
	Record explicit format:
	all supported data formats
	`mode=vdif_8000-1024-16-2`
	`mode=mark5b-2048-16-2`
	`mode=MKIV1_2-1024-16-2`
	`...`

(*) see Mark5C ‘`packet=`’ command

Multiple streams

d-plane	jive5ab
1 recording ≥ 1 streams ‘subgroups’ to split streams to different modules	1 recording = 1 stream ≥ 1 parallel recording, each indepently configurable (`set_disks=`)

Disk management

c-plane	jive5ab
disk management: mount/unmount format	no disk management: better done with shell commands/scripts(*)

(*) c-plane is Python which generates and executes shell commands for you:

```
for dsk,mp in [("/dev/sdb1", "/mnt/.."), ...]:
   subprocess.call(["mount", "-t", "xfs", dsk, mp])
```

Slamming the bytes on disk is only half the story!

- post-recording check
- recorder state
- play back at correlator
- e-transfer/conversion
- disk-shippingless operations
- ....

Useful Mark5 commands on FlexBuff/’6

```
# disk space on selected disks
rtime?      # uses recording data rate from mode=
dir_info? # recorded space and total free space

# post recording
scan_set = ...  # see Mark5 manual
scan_check?     # will recognize VDIF (heuristically)

# extract data range selected via scan_set =
disk2file = /path/to/file
disk2net = connect : host.ip.com
```

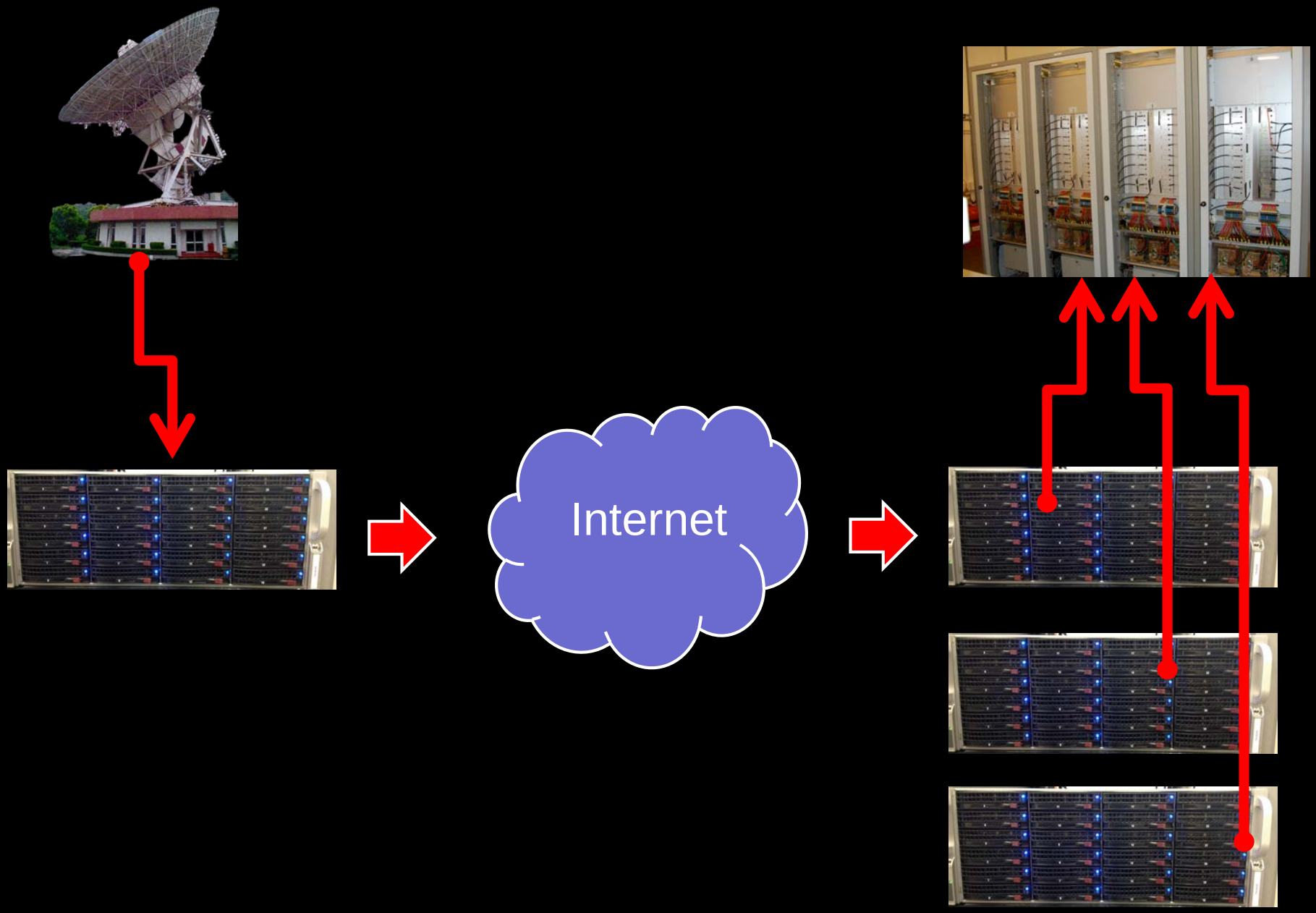
Advantages FlexBuff recording ‘format’

each file is a (small) time slice:

- only VLBI payload
- individually correlatable by DiFX and SFXC
- individually inspectable (e.g. `file_check?`)
- transferable (256MB is a manageable size)

can use UNIX utilities + scripts to locate/manipulate data

EVN goes disk shipping less operations


Internet

FlexBuff advantages

Independent of FlexBuff layout at station/JIVE

vbs ➔ vbs transfers are a sync operation

- only missing bits are transferred

Station

/mnt/disk0/eg053/

- eg053.00000000
- eg053.00000006

/mnt/disk1/eg053/

- eg053.00000001
- eg053.00000007

/mnt/disk2/eg053/

- eg053.00000002
- eg053.00000005

/mnt/disk3/eg053/

- eg053.00000003
- eg053.00000004

JIVE

/mnt/disk0/eg053/

- eg053.00000000
- eg053.00000006
- eg053.00000001
- eg053.00000007

/mnt/disk2/eg053/

- eg053.00000002
- eg053.00000005
- eg053.00000003
- eg053.00000004

Sidekicks

c-plane/d-plane	jive5a
`gather/vdifuse`	`vbs_fs`
	`m5copy`

`gather/vdifuse/vbs_fs`:

present scattered recording as one file

- `gather` copies
- `vdifuse/vbs_fs` are FUSE virtual file systems

`m5copy`:

"*copy VLBI data from somewhere to elsewhere*"

```
$> m5copy mk6://host:port/<recording> ...
$> m5copy vbs://host:port/<recording> ...
```

Note: rem source and rem destination are different

DST \ SRC		Mark5		File		FlexBuff		Mark6	
		lcl	rem	lcl	rem	lcl	rem	lcl	rem
Mark5	lcl								
	rem								
File	lcl								
	rem								
FlexBuff	lcl								
	rem								
Mark6	lcl								
	rem								

jive5ab new command line arguments

```
$> jive5ab [-6] [-f <format>]
```

-6 look for Mark6 mountpoints
(use `set_disks=` to change @runtime)

-f <format> set default recording format:
‘mk6’ record in native Mark6 mode
‘flexbuff’ record in FlexBuff mode
(use `record=mk6:[01]` to change @runtime)

Summarizing FlexBuff/Mark6

`jive5ab` 2.7.* can

- record in either format
- on either system
- read either format
- on either system

Differences wrt MIT Haystack c-plane/d-plane software:

- only record known data format (not arbitrary packet dump)
- but easy to specify the data format (libmk5access-like)
  - “VDIF_8224-8192-1-2” / “MARK5B-2048-32-2”
- c-plane: >1 network card ⇒ 1 recording
- `jive5ab`: >1 network card ⇒ >1 recording
  - per stream control where it’s recorded
  - starting a recording scripted anyway

Record recipe for FlexBuff/Mark6

```
# configure network
net_protocol = udp|pudp|tcp|udt  # which protocol
mtu = 9000 # UDP based protocols need this
net_port = 42667 # port number to listen on for data

# where to stripe data [optional: in which format?]
set_disks = .... # default: flexbuff disks
record = mk6 : 0|1 # default: vbs format

# what is the format of the data being recorded?
mode = VDIF_8192-4096-32-2

# and record it
record = on : <scanlabel>
```

Thank you for your attention

Availability

`http://www.jive.eu/~verkout/evlbi/`

`jive5ab` ".deb" installation, source code

`http://www.jive.eu/~verkout/flexbuff/`

Flexbuff scripts and documentation

`http://www.jive.eu/~verkout/evlbi/m5copy`

direct download link, always latest version

`http://www.jive.eu/~verkout/evlbi/m5copy.html`
`http://www.jive.eu/~verkout/evlbi/changelog`

changelogs of `m5copy` and `jive5ab` for inspection

Summarizing Mark6/FlexBuff

`jive5ab` will NOT:

- read FlexBuff format
- read Mark6 format

⇒ use FUSE(*) virtual file system for reading as single file

- `vbs_fs` (for FlexBuff format) – distributed by JIVE
  - `./vbs_fs [options] /path/to/dir`

- for Mark6 format either:
  - MIT Haystack (Geoff Crew?)
  - Jan Wagner's `fuseMk6`

(*) http://fuse.sourceforge.net/

Remember scan_set= ?

Purpose: Set start-scan and stop-scan pointers for data_check, scan_check, disk2file and disk2net.

Settable parameters:

Parameter	Type	Allowed values	Default	Comments
<search string>	int or ASCII	scan number | scan label | 'inc' | 'dec' | 'next'	last recorded scan	First attempts to interpret as scan number (first scan is number 1); if not numeric or no match, attempts to mat all or part of existing scan label, case insensitive (see Note 1). 'inc' increments to next scan; cycles back to first scan at end; 'dec' decrements to previous scan. 'next' finds next scan with previous value of <search string>. If null field, defaults to last fully recorded scan.
<start read>	char | time | int	s | c | e | s+ | <time> | +<time> | -<time> | +<bytes> | -<bytes>	s	s|c|e|s+: Set start scan position to 'start', 'center', 'end' (actually ~1MB before end) of scan, or specified <tim within scan; this is convenient if you want to do a subsequent 'data_check' at a prescribed position. 's+' sets t start-scan pointer to 65536 bytes past the start of the scan. <time>: time within scan: see Notes 2 & 3 +<time>: offset time from beginning of scan (i.e. '+30s' will start 30 seconds from beginning of scan) -<time>: offset time from end of scan (i.e. '-30s' will start 30 seconds before end of scan) +<bytes>: offset number of bytes from beginning of scan. -<bytes>: offset number of bytes from end of scan
<stop read>	time | int	<time> | +<time> | -<time> | +<bytes> | -<bytes>	end-of scan	<time>: Time at which to end readback; see Notes 2 & 3. If preceded by '+', indicates duration of data (in rec clock time) from <start scan> time. +<time>: offset time from <start scan> position. -<time>: offset time from end-of-scan +<bytes>: offset bytes from <start scan> position -<bytes>: offset bytes from end of scan

Remember `scan_set=` ?

In order to fully support:

```
scan_set = n15x1_o6_no0003 : 10m30s : +2s
disk2file = /path/to/file
```

Remember `scan_set=` ?

In order to fully support:

```
scan_set = n15x1_o6_no0003 : 10m30s : +2s
disk2file = /path/to/file
```

jive5ab must be able to read back the recording!

Remember scan_set= ?

In order to fully support:

```
scan_set = n15x1_o6_no0003 : 10m30s : +2s
disk2file = /path/to/file
```

jive5ab must be able to read back the recording!

```
disk2net = connect : host.ip.com
```

Upcoming m5copy capabilities

Copy FlexBuff or Mark6 recordings anywhere:

To local or remote file:

```
$> m5copy vbs://.../  file://[host.ip]/path/
$> m5copy mk6://.../  file://[host.ip]/path/
```

To remote Mark5:

```
$> m5copy vbs://.../  mk5://host.ip/path/
$> m5copy mk6://.../  mk5://host.ip/path/
```

.... etc ...

Upcoming m5copy capabilities

Resume an interrupted transfer!

```
$> m5copy --resume [...] SRC DST
```

Only works on limited set of transfers:

- DST must be file://.../
  - other endpoints do not support appending
- SRC can be mk5, file, vbs or mk6

But works on local + remote transfers!

vbs_fs FUSE file system novelties

Use to present FlexBuff style recordings as single files

```
$> vbs_fs [-6] [...] /path/to/dir
```

- Acquired command line option “-6”
  - look in Mark6 mountpoints for recordings
- Increased performance by scheduling reads by disk
- Increased stability by disabling background indexing

Summarizing FlexBuff/Mark6

`jive5ab` 2.7.0 can

- record in either format
- on either system
- read either format
- on either system

Differences wrt Haystack c-plane/d-plane software:

- only record known data format (not arbitrary packet dump)
- but easy to specify the data format (libmk5access-like)
  - “VDIF_8224-8192-1-2” / “MARK5B-2048-32-2”
- cplane: >1 network card ⇒ 1 recording
- `jive5ab`: >1 network card ⇒ >1 recording
  - per stream control where it’s recorded
  - starting a recording scripted anyway

Thank you for your attention

Availability

http://www.jive.eu/~verkout/evlbi/

jive5ab ".deb" installation, source code

http://www.jive.eu/~verkout/flexbuff/

Flexbuff scripts and documentation

http://www.jive.eu/~verkout/evlbi/m5copy
http://www.jive.eu/~verkout/evlbi/DirList.py
http://www.jive.eu/~verkout/evlbi/SSErase.py

direct download links, always latest version

http://www.jive.eu/~verkout/evlbi/m5copy.html
http://www.jive.eu/~verkout/evlbi/changelog

changelogs of m5copy and jive5ab for inspection