Data structure for transfer to QBiC

The most important aspects

1. The root folder is to be named **`qbicbarcode_internalname_PAEnumber`** before transfer (as before). In addition to the raw data, all other files are expected to be part of the transfer. See the QBiC Github core-utils-lib repository for the expected file structures. The underlying data structure can be seen in the data-model-lib repository.
2. If multiple samples were run on the flow cell, only one of the relevant sample barcodes will be used in the root folder and the suffix **`_pooled`** will be added to the tail of the root folder name. (e.g. with a naming schema similar to QABCD001AB_E12A345a01_PAE12345)
3. In the renamed root directory a directory is created for each run, which will host all files associated with a run. (e.g. with a naming schema similar to 20200122_1217_1-A1-B1-PAE12345_1234567a).
4. If multiple samples were run on the flow cell, Med. Micro will create (or rename the barcode) subfolders with the relevant QBiC barcodes in all four raw data folders (fastq_fail, fastq_pass, fast5_fail, fast5_pass) containing the corresponding data
   This can be done via the ***rename-nanopore.sh*** script stored in the ./bin directory on the workstation. The source code for this script is stored on the ncct-mibi github repository. Alternatively you can use the **barcode-rename_csv.py** script described below in this SOP
5. If multiple runs are contained in the root folder, both are handled the same way (no data is merged between those runs)
6. Med. Micro will merge and gzip the fastq sequencing data in each fastq folder, resulting in 2 fastq.gz files for each sample (fastq_fail and fastq_pass) per run
   This can be done via the ***cat-gz-pool.sh*** script for pooled runs or the ***cat-gz-single.sh*** script for non-pooled runs. Both scripts are stored in the ./bin directory on the workstation. The underlying script is stored on the ncct-mibi github repository.
7. Technical replicates (same barcode, 2+ flow cells) are not pooled, but sent separately
8. Permissions: make sure the permissions are correctly set before transfer - should be 660 for files and 770 for directories.
   From the root directory run the following commands:

```
find . -type f -exec chmod 660 {} \;
find . -type d -exec chmod 770 {} \;
```

Naming schema in detail:

Projekt/Root Folder:

Example Name: **“QABCD001AB_E12A345a01_PAE12345”**

Seperated by underlines:

QABCD001AB: QBiCBarCode

E12A345a01: SampleID

PAE12345: Flow Cell ID

Run Directory:

Example Name: "**20200122_1217_1-A1-B1-PAE12345_1234567a**"

Seperated by underlines:

20200122: Date when the run was performed:

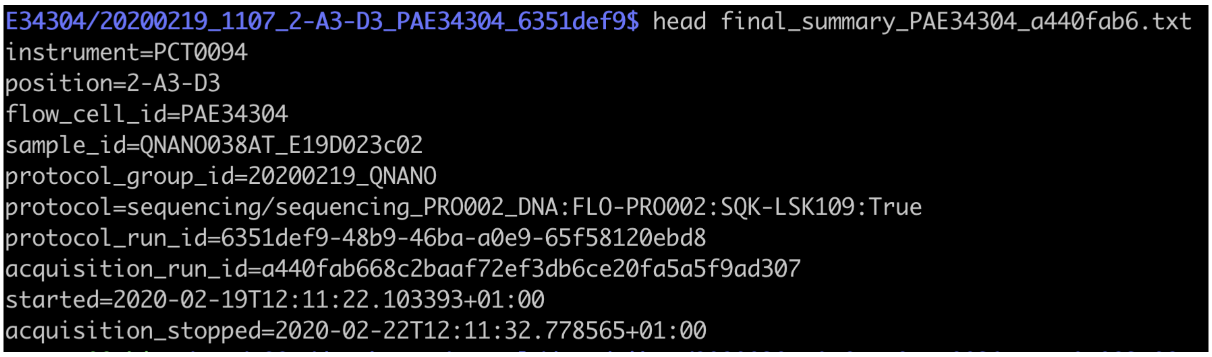
1217: Time in Hours and Minutes when the run was performed

1-A1-B1: Flow Cell Position

PAE12345: Flow Cell ID

1234567a: RunID

Most of the information can be retrieved from the *final_summary.txt* file which should look similiar to this:



```
E34304/20200219_1107_2-A3-D3_PAE34304_6351def9$ head final_summary_PAE34304_a440fab6.txt
instrument=PCT0094
position=2-A3-D3
flow_cell_id=PAE34304
sample_id=QNANO038AT_E19D023c02
protocol_group_id=20200219_QNANO
protocol=sequencing/sequencing_PRO002_DNA:FLO-PRO002:SQK-LSK109:True
protocol_run_id=6351def9-48b9-46ba-a0e9-65f58120ebd8
acquisition_run_id=a440fab668c2baaf72ef3db6ce20fa5a5f9ad307
started=2020-02-19T12:11:22.103393+01:00
acquisition_stopped=2020-02-22T12:11:32.778565+01:00
```

Useful commands to prepare data

For one sample per flow cell (no pooling)

Here, two things have to be changed in the raw output:

1. Add the QBiC barcode as prefix to the top folder:

```
QNANO028AM_E19D023a02_PAE26998
| |-- 20200219_1107_1-E3-H3_PAE26974_454b8dc6
| | |-- fast5_fail
| | | |-- ...
```

2. Merge and gzip the fastq files in fastq_pass and fastq_fail, use **`bin/qbic-prep2.sh`** from the etc repo

For many samples per flow cell (barcoding, pooling)

First, the barcode01, barcode02 etc folders have to be renamed to the corresponding QBiC barcode names. Using a csv file:
!!! Take care of the line endings of the csv file - if it is coming from Windows then change to Unix line endings with Notepad++!!!

```
while IFS="," read col1 col2; \
do echo mv fastq_pass/$col2 fastq_pass/$col1; \
done < 2005-sh-sample-rename.csv > fqp.sh
```

Then check if the script is ok and just execute it:

```
sh fqp.sh
```

After the folder are renamed, merge and gzip the fastq files there:

```
find ./ -type d -name 'QNFL*' -ls -execdir sh -c 'cat {}/*.fastq > {}/{}.fastq &&
pigz {}/{}.fastq' sh ";"
```

Get stats on the files, fast:

```
parallel seqkit stats -a ::: fastq_pass/*/*.fastq.gz | sed '/file/d'
```

Check everything is OK and delete fastq files:

```
find ./ -type d -name QNFL* -ls -execdir sh -c 'rm -v {}/*.fastq' sh ";"
```

Useful scripts for doing rename+cat+pigz for many samples per flowcell:
`bin/qbic-prep1.sh` and **`bin/qbic-prep2.sh`** from the etc repo

`barcode_rename_csv.py`, rename the barcode to the QBIC barcode

```
barcode_rename_csv.py -d /path/to/project/{fast5_pass, fast5_fail, fastq_pass,
fastq_fail} -c file.csv
```

Example for the CSV file:

```
barcode01,qbicbarcode01
barcode02,qbicbarcode02
barcode03,qbicbarcode03
```

For Illumina files

Just add "QBiCBarcodes_" before the file name.

Example:

```
2006-AH-d1-001_S1_R1_001.fastq.gz -> QHPUW298AE_2006-AH-d1-001_S1_R1_001.fastq.gz
```

Script:

```
#!/bin/bash
while IFS="," read nb ob
do
for f in "$ob"*
do
echo mv -v $f "$nb"_"$f"
done
done < rename.csv > rename.sh
```

Execute rename.sh after checkout it is OK!

Example for rename.csv file

```
2006-AH-d1-001,QHPUW298AE
2006-AH-d1-002,QHPUW299AM
2006-AH-d1-003,QHPUW300AT
```

!!!Be careful, use the following command to make the line endings of the CSV file to the UNIX format on the workstation. Or use linux line endings in Notepad++ when saving the rename.csv

```
dos2unix rename.csv rename.csv
```

Start **transfer** by copying all fastq files to /home/sysgen/datamover/data/incoming/

```
# First check daemon is running:
~/sysgen/datamover/datamover.sh status

cp -rf /path/to/fastq/*.fastq.gz /home/sysgen/datamover/data/incoming/

# to monitor the transfer you can do
tail -f home/datamover/log/datamover_log.txt

# stop tail with CTR-C
```

Tipp von Matthias Seybold: In case of problems with datamover it sometimes works to stop the program and to start it again.

check status:

```
sudo systemctl status openbis-dm
```

stop:

```
sudo systemctl stop openbis-dm
```

restart:

```
sudo systemctl start openbis-dm
```

Data transfer with rsync (if datamover fails)

```
rsync -av --progress --partial --perms --chmod=Fo-rwx --chmod=Fug+rw
--chmod=Dug+srwx --chmod=Do-rwx
```