**Code Structure – Multiscaler Image Generator**

**List of functions:**

1. *main\_multiscaler\_data\_readout.m*

Call line: Script.

Purpose: Main program that controls the data readout from multiscaler list files. Run it to open the GUI to choose the proper file or folder for readout. The output is the images listed in the list files.

2. *Multiscaler\_GUI.m*

Call line: H = Multiscaler\_GUI;

Purpose: GUI script.

3. *Multiscaler\_GUI.fig*

Call line: None.

Purpose: GUI figure script.

4. *LSTDataRead.m*

Call line: [Binary\_Data, Time\_Patch, Range] = LSTDataRead(FileName);

Purpose: Receives a filename FileName of a list file (.lst) from the multiscaler and creates a binary string char vector of this data. Also saves the timepatch and range value from the data.

5. *CreateDataVectorXXX.m*

Call line: STOP1\_Dataset = CreateDataVector32(Binary\_Data, 1, double(Range));

Purpose: Receives data from *LSTDataRead* as a long char vector, current data channel (1, 2, 6) and according to the timepatch value (that’s the XXX) it returns a table filled with all events from that data channel and their different properties (TAG input, sweep number). The events are listed in timebin units.

6. *PhotonCells.m*

Call line: [PhotonArray, NumOfLines, StartOfFrameChannel, MaxNumOfEventsInLine, TotalEvents] = PhotonCells(START\_Dataset, STOP1\_Dataset, STOP2\_Dataset, 1);

Purpose: Receives all tables with all data, and the number in which the PMT data events came in (last input number). With its subfunctions it divides the PMT data to lines (line signals are the most frequent signals that did not originate in the PMT channel). The function itself only calls its subfunctions that actually do the divisions after it detects which channel is the line channel. It returns:

PhotonArray – the list of photons with their start-of-line time (their time is zeroed for every new line).

NumOfLines – amount of line signals received.

StartOfFrameChannel – a number (1,2,6) that determines the channel from which the start of frame signal is originating.

MaxNumOfEventsInLine – a number that tells other functions the maximal number of photons found in a line.

TotalEvents – the total amounts of photons we have for this image.

7. *CreateDataList.m*

Call line: [PhotonArray, MaxNumOfEventsInLine] = CreateDataList(TotalEvents, Num\_of\_Lines, STOP1\_Dataset, START\_Dataset);

Purpose: Receives the line data and photon data, as well as the total number of photons in the list and the number of lines, and creates a two columned vector – left column is the photon arrival times, and right column is the fitting start-of-line time. Also returns the maximal number of photons one can find in a single line.

8. *CreateFrameStarts.m*

Call line: StartOfFrameVec = CreateFrameStarts(STOP2\_Dataset.Time\_of\_Arrival(:));

Purpose: Creates a vector containing the time bins of all start-of-frame signals received, after doubling them.

9. *ImageGeneratorHist3.m*

Call line: RawImagesMat = ImageGeneratorHist3(PhotonArray, SizeX, SizeY, StartOfFrameVec, NumOfLines, TotalEvents);

Purpose: Uses the hist3 MATLAB function to create an image from the two-columned vector. It returns it as a 3D matrix of intensities in RawImagesMat. It receives the photon data, size of desired image and a data vector containing times of start-of-frame signals (StartOfFrameVec).

10. *PhotonSpreadToImage2.m*

Call line: RawImagesMat(:,:,CurrentFrameNum) = PhotonSpreadToImage2(CurrentEvents, SizeX, SizeY, EdgeX, EdgeY);

Purpose: Histograms the CurrentEvents containing the relevant photons for this frame in a frame of SizeX by SizeY pixels. Edges are the ends of every line in the historgram.