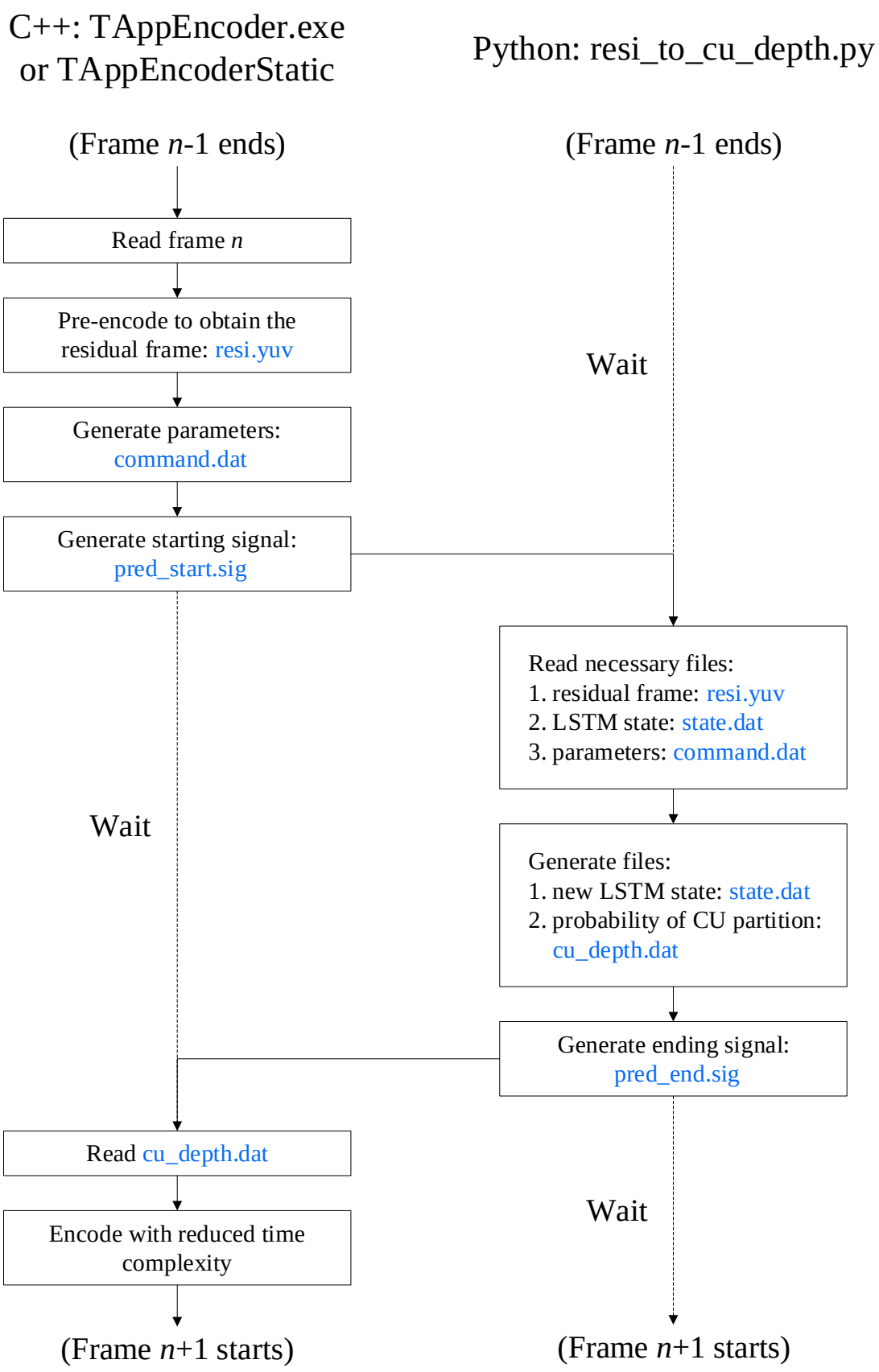
Combination of HM and CNN+LSTM



The HM encoder was written in C++, while ETH-CNN and ETH-LSTM were written in Python. In our implementation, the communication between them is realized with signaling files, as described below.

1. Before encoding each frame, a simplified configuration (maximum CU and PU sizes) is adopted in HM for pre-encoding, to obtain the residual frame save it into file “resi.yuv”.
2. HM writes some parameters into file “command.dat”
   Format: POC frame_width frame_height QP [end]
   Example: 19 416 240 22 [end]
3. HM generates a signaling file “pred_start.sig”, indicating that the pre-encoding is finished and the Python program can be started.
4. When “pred_start.sig” is detected, the Python program reads come necessary files, including “resi.yuv” and “command.dat”. In addition, the previous state of LSTM is stored in file “state.dat”, and the Python program also needs to read this file.
5. The residual frame and the previous state of LSTM are input to our networks, for generating the predicted probability of CU partition for the current frame, stored in “cu_depth.dat”. Also, the state of LSTM is updated, and the new state is stored into “state.dat” by over-writing the previous state.
6. The python program generates another signaling file “pred_end.sig”.
7. When “pred_end.sig” is detected, HM reads “cu_depth.dat” to determine the CU partition.
8. Encode according to the CU partition, with reduced time consumption than the standard HM.

Repeat the above steps, until all frames are encoded. To summarize, the communication between C++ and Python programs is achieved by two signaling files, “pred_start.sig” and “pred_end.sig”.