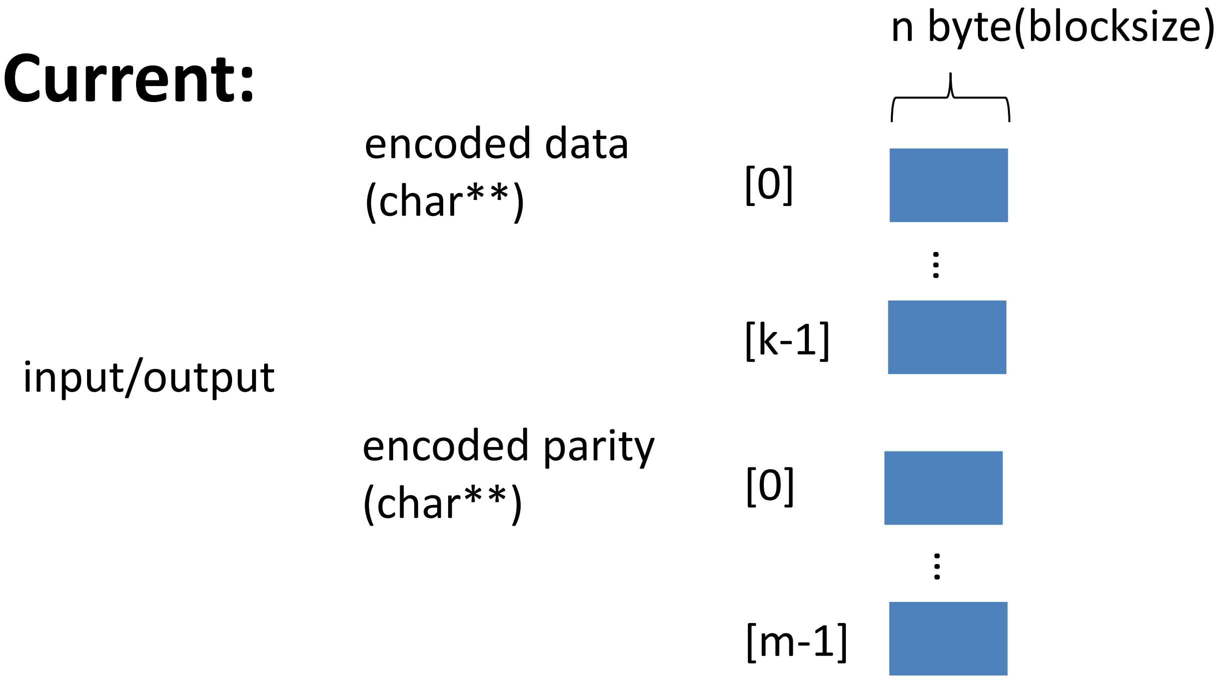
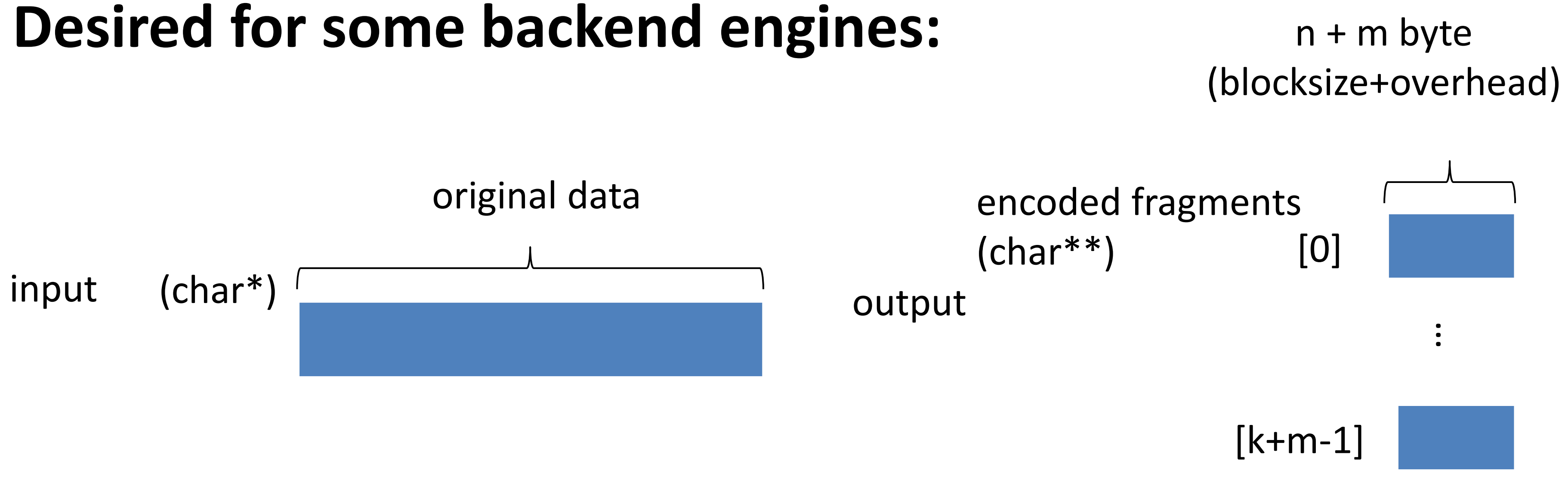
liberasurecode encode preprocess design

- prepare_fragments_for_encode() looks good to serve a list of pre-separated (and aligned) original data. Many of engines will be benefited.
- However, there is a couple of things to consider as follows:
  - The backend engine may require a natural sequential (non-separated) string for encode.
  - The backend engine may want to add a special data to each fragment. (i.e. pre-malloced buffer by prepare_fragments_for_encode() may be too small for the backend engine)
- To solve these problems, I would like to put an option to skip the preprocess.
- Note that this is the first design and it is not considered enough for eco implementation. If there is better idea, it will be great for liberasurecode.

Current:



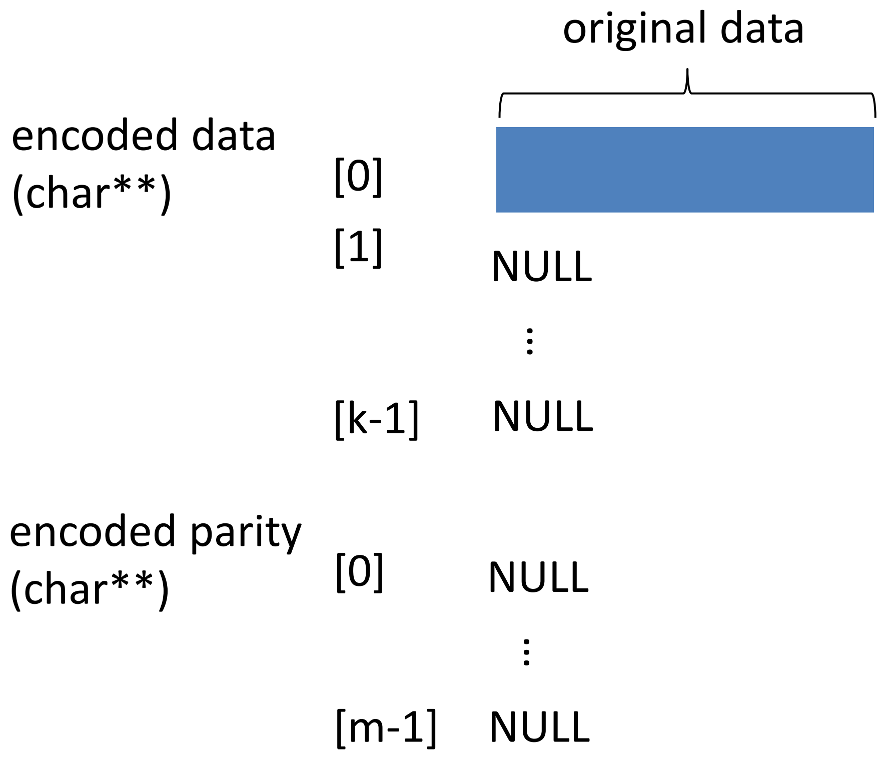
Desired for some backend engines:



Idea:

ec_backend_common.skip_preprocess = 0 or 1 // 0: false 1: true

prepare_fragments_for_encode()



NOTE: if ec_backend_common.skip_preprocess == 1, a backend engine MUST create fragments which includes fragment_header_t to ensure the fragment format.

Sample Code:

```
ec_backend_common.pass_preprocess = 0 or 1 // 0: false 1: true
prepare_fragments_for_encode(*snip*){
    *snip*
    if(instance->common.processing==1){
        encoded_data[0] = malloc(sizeof(char)*aligned_data_len);
        memcopy(encoded_data[0], orig_data, orig_data_size);

        for(i=0; i<k; i++) encoded_data[i] = NULL;
        for(i=0; i<m; i++) encoded_parity[i] = NULL;
        goto out;
    }
    *snip*
}
```