Kyle Jenkins

clblake Notes

main
1: get flags
2: usage if optind gt argc
3: else, action_file_(c/g)pu based on cpy flag
4: else, action_test

5: hash2str function (hash given uint8_t hash into given out char ptr)

6: action_file_cpu
    a: variables
    b: open given file
    c: blake256_init
    d: allocate mem for src and dst
    e: while bytes_read = freed, blakeTreeCPU + blake256_update
    f: close file
    g: blake256_final
    h: hash2str
    i: end

7: action_file_gpu
    a: variables
    b: open given file
    c: blake256_init
    d: while not done
        i: acquire dst
        ii: blake256_update + blakeTreeGPU_release_dst (else done = true)
        iii: while not eof and src = blakeTreeGPU_acquire_src
            1: enqueue src by bytes read
    e: close file
    f: blakeTreeGPU_close
    g: blake256_final
    h: hash2str
    i: end

8: test_gpu
    tests the gpu for the program

9: action_test
    tests cpu hash

---

blake.h

very similar to our blake2.h -- has funcs for blake2 and blake256 operation

---

blakeTreeCPU.h
hash out parallel in and out uint8_t into blake256_hash

---

blakeTreeGPU.h
1: buffer_t struct declared
2: global opencl state
3: blakeTreeGPU_init
   a: initialize all the opencl stuff (program, context, kernel, etc.)
4: blakeTreeGPU_close
   a: free up all the opencl stuff.
5: acquire_src
   a: wait for a free buffer
   b: exit if no free buffer, otherwise return new->src
6: enqueue_src
   a: enqueues the src into the knerl program
   b: set variables of new to current variables
7: acquire_dst
   a: find and return head->dst
   b: also release_dst function
8: allocate and free buffer
9: pending and wait funcs

---

blake256.cl
1: many blake256 functions translated into the cl file.
2: leads to kernel void blake256_hash_block
   a: get global/local ids
   b: set global item_in and item_out
   c: init state256 S
   d: update S using item_in and item_out
   e: final using S and item_out