- What have you been able to implement and get working up until now?

I finished the part 1 and finished writing part 2 since now.

For part 1, I opened the helloworld.elf and xinu.elf files and getmem for these files before reading. Then find the section location in the section table , locate the symbol table, relocation table and string table. Different table have different element or entries, also the calculation method is different. Next, find the relocation table and conditionally find the type of the relocation, and find which section it belongs to. Then, do the relocate by using the equation given by Intel i386, and do the relocate. Before the relocate, there is one more thing, we have to find the relocate function position in the symbol table in the xinu.elf by comparing the string value in the string table of helloworld.elf and xinu.elf. Finally, after the relocation, I test the number of the main function in the helloworld.elf file and return the address of the initial location of .text file in helloworld.elf.

For part 2, the load library part is almost the same with load program. But there is some extra implementations. First, I add a hashtable in every process with hashmap. I implemented the calloc function in xinu by memset the getmem result to be 0. Then I add the hashtable_put , hashtable_get, hashtable_length and even more functions in hashmap.c in system. Also add the header in prototypes.h and process.h, and create hashmap.h file in /include. Hashmap is necessary for the implementation of library, since there would be a key and value for every function, using array for this library common container would be complicated. So I use the entry element in the process table, like number of the library functions, the table of loaded library address, the hash table of the total library functions, to make it easier for find library function. All the entry elements has been initialized in the create process function.

For find library function file, I tested whether the library number is larger than 3? Whether the library name has already existed in the library table? Whether the function name in that library has already been loaded in the hash table? If no, then it will return an error.

- What do you plan on implementing before the final due date?

After this midway, I would like to debug the part 2 first, and start to modify the shell of xinu. After that, I would do the extra credit like unloading a library if I have enough time.

- What problems have you run into while preforming your implementation? What did you do to solve those problems?

I have met a lot of problems during this project.

1. The file sever cannot by connected. There is no output come from the server. The file cannot be read. I think most of us have this problem, I still don't know how to solve this, since we are using ssh method, and there may be some bugs here or some improper implementations there. Anyway, when I go to the xinu lab, this problem has been solved. If I used ssh back home, sometimes this problem will come up again.
2. The equation for relocation in the specifications is ambiguous. If you just simply look into the datasheet or manual, you can hardly understand what they are saying. So I tried to search on the source code on web in dealing with the relocation of elf file. The relocation definitions of this remote server are still hard to understand. I asked a lot of classmates and also TAs before I finally figured out what S, A, P stand for. The process of reading file is complicated.
3. There is no calloc function implemented in xinu. The getmem function is almost the same as malloc in c. The calloc function will initialize the address of memory to be 0, but getmem would not. So, I have checked that there is a memset function in xinu, even though I don't know where it is. I tried to use this memset to set all the value in getmem address to be 0.
4. The hash table implementation is another hard problem. If trying to use a simple data structure like array or queue to implement the library, it is quite hard and almost impossible. The implementation of hash table would be easy. I found a module on web and modify some part of it to make it fit in xinu, like the type of key or value, and malloc method in c. Finally, I build a text case in main.c to simply test this function.