Implementation notes:

File cache is maintained as a doubly linked list structure as follows.

struct file\_cache {

char filename[MAXSIZE]; // Name of the file

int pin; // {0,1} Pin value of 1 indicates file is in cache, 0 not in cache.

int dirty; // {0,1} Dirty bit. 1 indicates that buf contents are not written to local file system. 0

indicates cache and local filesystem are consistent.

char \*buf; // buffer to hold the data already read or data to be written to file system.

file\_cache\* next; // pointer to the next element in the cache list. For tail node(last node in cache list)

next is NULL.

file\_cache\* prev; // pointer to the prev element in the cache list. For head node (first node in cache

list)) prev is NULL.

};

It is assumed that there is only a single cache which client can access and it can hold a maximum of 100 entries.

Client can call file\_cache\_pin\_files() with a list of files and number of files in the list to load a list of files to cache.

I am not using file\_cache\_construct() specified in the given .h file. Hence i have removed it and a global variable max\_entries hold the no of elements in the cache. I assume its OK.

Program flow is as follows.

Compile with DEBUG set to print debug messages:

For eg: gcc -w -DDEBUG=1 –o file\_cache file\_cache.h file\_cache.c main.c

Function : file\_cache\_pin\_files() {to add a list of files to cache}

===================================================================

When the client calls file\_cache\_pin\_files() to add a list of files to cache, the cache is searched using search\_cache() with filename to check if the file is already cached or not.If the file is already, cached pin is incremented, it returns. If it is not cached already, a function called add\_to\_cache() is called . add\_to\_cache() checks if the cache is already full. If so it returns. If it is not full, it checks if the file exist in local file system or not, using exclusive open().If file exist in local file system , it is opened in read only mode and is pinned to cache. Also it reads MAXSIZE bytes to cache buffer pointed file\_cache->buf. If the file doesn't exist in local file system , it is created with mode 0777 and truncated to 10240 bytes and is added to cache list.

Function: file\_cache\_mutable\_file\_data() {to write data to cache}

==================================================================

Used to write data to cache. First function checks if file exist in cache. If it doesn’t exist in cache add\_to\_cache() is called to add the file to cache. Else if file exist in cache (i.e. if pin associated with it is 1) , check the value of dirty bit.If dirty bit is 0 , cache and local file system are consistent and data to be written is copied to buf (file\_cache->buf) associated with file\_cache. If dirty bit is 1, cache and local file system are not consistent(file\_cache->buf contains data that is not yet written to disk) and so the data already existing in buf is written to local file system using write\_to\_file and then new data is copied to buffer and dirty bit is made 1.

Function: file\_cache\_file\_data() { to read data from cache }

=============================================================

Used to read data from cache. Cache is searched using search\_cache() to check if file already exist. If yes a pointer to the data existing in buffer is returned.If file doesn't exist in cache, it is added to cache using add\_to\_cache() and MAXSIZE data is copied to buf and a pointer to the same is returned.

Function: file\_cache\_unpin\_files() { to remove a list of files from cache}

===========================================================================

A list of files is given to file\_cache\_unpin\_files() to be removed from cache. Each file in the list is searched in the cache using search\_cache() to check for existence.If file exist in cache, dirty flag is checked to see if it has some unwritten data. If dirty is 1, contents of buf are written to local file system.Pin is decremented and if pin becomes 0, file\_cache object is deleted from the list , pointers are adjusted to reflect the change and max\_entries count is decremented.

Function : file\_cache\_destroy {to flush the entire cache to disk}

================================================================

This function is used to flush all dirty cache entries to local file system. It takes a pointer to the beginning of the list and iterate through each file\_cache object. If any object is encountered with dirty bit 1 , its contents are written to disk , dirty bit is changed to 0 and buf associated with that file\_cache object is emptied.