CS 502
Project 1 Report

Jiaming Nie 369577520

Content

1 Architectural Document

This project generally implemented the functions on the following aspects:

- Process Management.
  This project will use process control blocks to store information and different Queues, timer Queue and Disk Queue to dipatch the management.
- Physical I/O operation.
  This part uses the hardware functions to utilize these constructions.
- File Management.
  Implementation based on the physical I/O operation.

1.1 Design List

1.1.1 Functions in base.c

Table 1: Functions in base.c

Type	Function Name	Description
void	Dispatcher()	Context Switch, examine whether the Ready Queue is empty or not.
void	DoOneLock(INT32)	Package of the Hardware lock READ_MODIFY().
void	DonOneUnLock(INT 32)	Package of the Hardware lock READ_MODIFY().
long	GetPIDByName(char *)	Return the process id by the process name.
void	InitializeAndStartCon text(MMIO,PCB, void)	Initialize the context and run the context.
long	OSCreateProecss(PC B*)	Create the process and return the process id.
void	Physical_Disk_Read (long,long,long)	Read the disk, the disk id and the write buffer should be determined.
void	Physical_Disk_Write (long,long,long)	Write content to the disk. The disk id and the write buffer should be determined.
void	StartTimer(long*)	Start the Hardware timer.
void	WasteTime()	This function is to let system continue when no other

1.1.2 Functions in Queue.c

Table 2: Functions in Queue.c

Type	Function Name	Description
PCBNode	DeQueueByName(PCBQueue *,PCB *)	DeQueue opertaion from the PCB Queue.
PCBNode	DeQueueFirstElement (PCBQueue, PCB*)	DeQueue the first item from the PCB Queue.
PCBNode	EnQueueByPriority (PCBQueue *, PCB *)	Add one PCB to the PCB Queue, ordering by the priority.
PCBNode	EnQueueByWakeUp Time(PCBQueue *, PCB *)	Add PCB to the timer Queue, order by the wake up time.
PCBQueue	initialQueue()	Initial the PCBQueue.
int	IsEmpty(PCBQueue *)	Determine whether the PCBQueue is empty or not.
void	TerminateSelf (PCBQueue*,PCB *)	Delete the node from the PCB Queue.

1.1.3 Functions in Functions.c

Table 3: Functions in Functions.c

Type	Function Name	Description
void	FormatDisk(Block0*,Header*)	Format the disk for the file operation.
void	InitialBlock0(Block0*,ling)	Initialize the structure block0.
void	initialHeader(Header*,Block0*)	Initialize the header structure.
int	ReadBitMap(int,int,int)	Check the BitMap. Returning the usage of the disk.
void	UpdateBitMap(int,int)	Update the usage of the BitMap.
void	WriteHeader(Header*,long)	Write the header to the disk.

Figure 1 illustrated the relationship between the PCB and Queue.

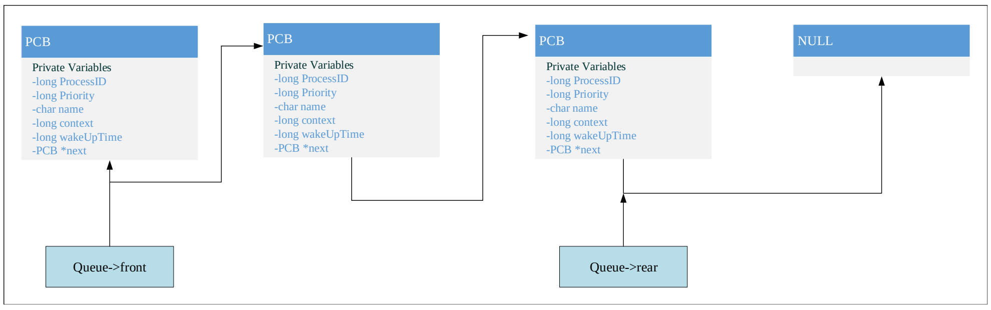
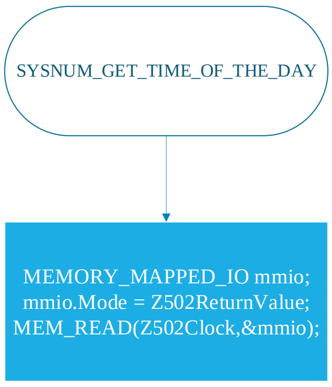


Figure 1: PCB in Queue

1.2 High Level Design

This part is about when the program runs, how it utilizes the system call.

1.2.1 SYSNUM_GET_TIME_OF_THE_DAY



```
MEMORY_MAPPED_IO mmio;
mmio.Mode = Z502ReturnValue;
MEM_READ(Z502Clock,&mmio);
```

Figure 2: SYSNUM_GET_TIME_OF_THE_DAY

1.2.2 SYSNUM_TERMINATE_PROCESS

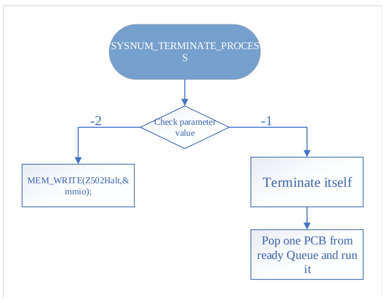

SYSNUM_TERMINATE_PROCESS
Check parameter value
-2
-1
MEM_WRITE(Z502Halt,&mmio);
Terminate itself
Pop one PCB from ready Queue and run it

Figure 3: SYSNUM_TERMINATE_PROCESS

1.2.3 SYSNUM_SLEEP

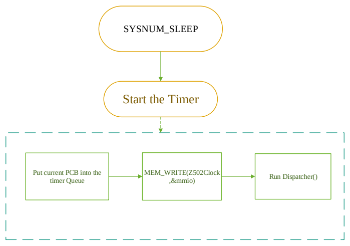

SYSNUM_SLEEP
Start the Timer
Put current PCB into the timer Queue
MEM_WRITE(Z502Clock ,&mmio)
Run Dispatcher()

Figure 4: SYSNUM_SLEEP

1.2.4 SYSNUM_CREATE_PROCESS

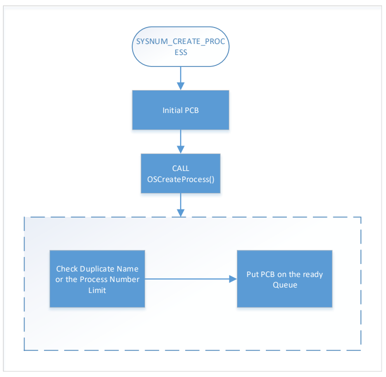

SYSNUM_CREATE_PROCESS
Initial PCB
CALL OSCreateProcess()
Check Duplicate Name or the Process Number Limit
Put PCB on the ready Queue

Figure 5: SYSNUM_CREATE_PROCESS

1.2.5 SYSNUM_GET_PROCESS_ID

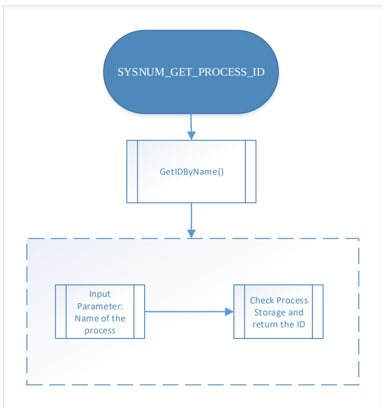

SYSNUM_GET_PROCESS_ID
GetIDByName()
Input Parameter: Name of the process
Check Process Storage and return the ID

Figure 6: SYSNUM_GET_PROCESS_ID

1.2.6 SYSNUM_PHYSICAL_DISK_WRITE

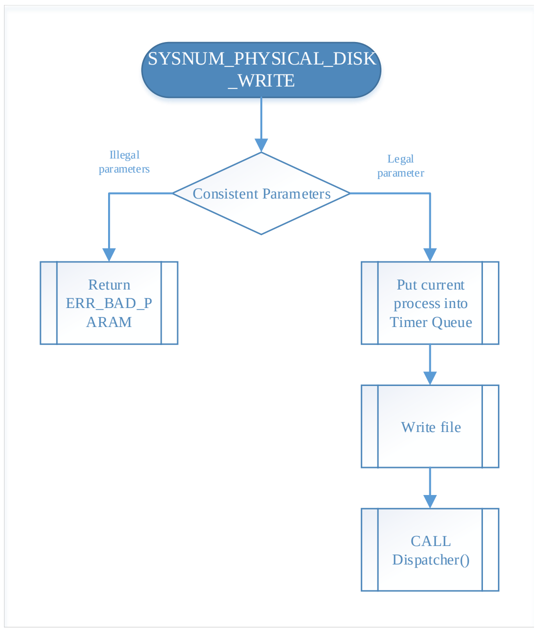


Figure 7: SYSNUM_PHYSICAL_DISK_WRITE

1.2.7 SYSNUM_PHYSICAL_DISK_READ

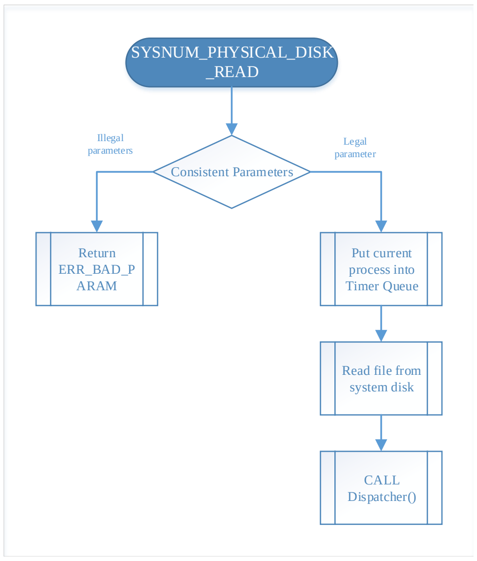


Figure 8: SYSNUM_PHYSICAL_DISK_READ

1.2.8 SYSNUM_FORMAT

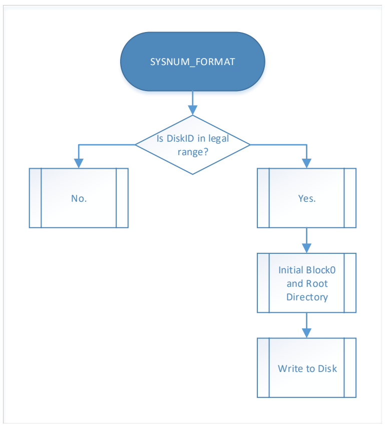


Figure 9: SYSNUM_FORMAT

1.3 Justification of High Level Design

1.3.1 Process Management

In the simulation, the operations of the process is implemented by different queues. Process is denoted by the structure of the PCB(Process control block).

a. Dispatcher in the project is used as to examine the ready queue and perform the context switch.

b. Ready Queue contains the process which waits for the system to make it run.

c. When the system call SYSNUM_SLEEP is invoked, then system will start timer and put the PCB in the timer queue, then check the ready queue and make the processes in the ready queue run.

d. Disk queue is used when the process performs the physical read and physical write operations, the PCB will be put in the disk queue, dispatcher will check if there are PCBs in the ready queue.

e. The function OSCreateProecess() is invoked when the system call SYSNUM_CREATE_PROCESS is invoked.

f. TerminateSelf() is used to release the space that the PCB of the process to be terminated, then Dispatcher() will examine the ready queue if there is no available processes then the system sits idle.

1.3.2 Physical I/O Operation

a. The Physical read and write operation used the Memory mapped IO of Z502 to perform the read/write operation.

b. When the physical read/write system call is invoked then current process will be put into disk queue.

c. Everytime the I/O operations ends, the InterruptHandler will receive one disk interrupt. This interrupt means the former I/O operation had ended and the former process should be removed from the disk queue and be put into ready queue.

d. At one time one disk can only be accessed by one process. Each disk need its own disk queue to manage the process which needs to write/read them.

1.3.3 File System

a. If there is process involving in the file management of the disk, it needs to format disk first. The format disk needs sector 0 to set the states or write content to the blocks of the disk.

b. The disk needs to initialize the sector and root directory first.

c. Once open one directory, the directory of the current directory will change. Current directory is one feature of the current process.

d. The struct header is used to determine the format of the disk or directory.

1.4 Anomalies and Bugs

1.4.1 Queue operations in the interrupt handler

When interrupt comes, in the interrupt handler different queues will input and output data to the queue.

In the uniprocessor, there will be chance that the operation will fail. The interrupt handler will need hardware lock.

1.4.2 Conflicts between Disk Interrupt and Timer Interrupt

This problem arises in test 8. Disk interrupt and timer interrupt will come by time order, sometimes all the timer interrupt will come before the disk interrupts.