Lab 7 Slides

CLOSE

#include<unistd.h>

int close(int *fd*);

Returns 0 on success, or -1 on error.

The *close()* call closes an open file descriptor, freeing it for subsequent reuse by theprocess.

DUP2

#include<unistd.h>

int dup2(int *oldfd*, int *newfd*);

Returns (new) file descriptor on success, or -1 on error.

The *dup2()* call makes a duplicate of the file descriptor given in *oldfd* using the descriptor number supplies in *newfd.*

_EXIT

#include<unistd.h>

void _exit(int status);

The exit call used by the child.

These are defined in stdlib.h:

#define EXIT_SUCCESS 0

#define EXIT_FAILURE 1

EXEC

#include<unistd.h>

int execvp(const char *filename, char *const *argv*[]);

The execvp call loads a new program and environment into the process's memory. It replaces the current process image with a new process image.

Does not return on success

Returns -1 on error.

See 9-UNIX, slide 11 for an example of setting up for the exec call.

FORK

#include<unistd.h>

pid_t fork(void);

The fork call creates a new process, the *child*, which is an almost exact duplicate of the calling process, the *parent*.

In **parent**: returns process ID of child on success, or -1 on error.

In successfully created **child**: always returns a zero

OPEN

```
#include <sys/stat.h>
#include <fcntl.h>
```

int **open** (const char *pathname, int flags, .../* mode_t mode */);

Returns file descriptor on success, or -1 on error

For flags: see 6-UNIX, slide 22; Table 4-3 (in LPI, page 74).

PERROR

#include<stdio.h>

void perror(const char *_msg_);

The *perror()* function prints the string pointed to by its *msg* argument, followed b a message corresponding to the current value of *errno*.

Example: page 49 of text.

Dealing with Errors.

In handle_redir, I often used:

```
fprintf(stderr, “message”);
```

WAIT

#include<sys/wait.h>

pid_t wait(int status);

Returns process ID of terminated child*,* or -1 on error.

The *wait()* system call waits for one of the children of the calling process to terminate and returns the termination status of that child in the buffer pointed to by *status*.

Lab 7 Slides

The End