COMP 3050-201 Computer Architecture

Homework #7 Spring, 2019

• This assignment is due no later than midnight (11:59:59 PM) **May 1**. **Hard deadline for end-of-the semester submissions is Thursday, May 2 11:59:59 PM (no exceptions).**

• All submissions must be made electronically using the submit command as described below.

- **File 1:** A short **write-up** that **first** specifies what you think your **degree of success** with a project is (**from 0% to 100%**), followed by a brief discussion of your approach to the project along with a **detailed description** of any problems that you were **not** able to resolve for this project. **Failure to specifically provide this information will result in a 0 grade** on your assignment. If you do **not disclose** problems in your write-up and problems are detected when your program is tested, you will receive a grade of 0. **Make sure that you include your email address in your write-up so that the corrector can email you your grade.**

- **File(s) 2(a, b, c, …)**: Your **complete source code**, in one or more **.c** and/or **.h** files.

- **File 3**: A make file to build your assignment. This file must be named **Makefile**.

- **File 4:** A file that includes your **resulting output** run(s) from your project. This is a simple text file that shows your output, but make sure that you annotate it so that it is self-descriptive and that all detailed output is well identified.

• The files described above should be the only files placed in one of your subdirectories, and this subdirectory should be the target of your submit command, this time specifying "hw7". See the on-line file **Assignment_Submit_Details.pdf** for specific directions.

The problem will require you to write a program on a UNIX/Linux platform (most any will do) which will:

- report a collection of **process and thread information** about itself to standard-out as described below
- create a unnamed **pipe**
- **spawn a child process** which will inherit access to the unnamed pipe
- the parent process will now **write a formatted text string into the pipe** and then do a normal exit
- the child process will report a collection of **process and thread information** about itself to standard-out as described below (just as the parent did)
- the child process will then **read the parent's string from the pipe** and write it to the standard output
- the child process will then get its **parent process ID** again and write it to the standard output and then do a normal exit

The information generated from both parent and child should look like this:

Process ID is:	PID
Process parent ID is:	PPID
Real UID is:	RUID
Real GID is:	RGID
Effective UID is:	EUID
Effective GID is:	EGID

Process initial thread priority: IT_PRIOR

All of your output must be prefixed by either PARENT: ……. or

CHILD:……… ,so that it can be readily identified as to its source.

A help file for this program can be found on Blackboard.