EC-ENG 231 (Spring 2024)

Review
Sample Questions

Fatima Anwar
fanwar@umass.edu
UMass Amherst

Midterm Logistics

- **In-person Moodle Exam** with double seating (*exceptions for students with accommodations*)
- Location: HAS 20, ELAB II
- Bring your computers to the exam location!

- **Exam duration**: 1 hour 15 minutes
  - Wednesday March 27th, 2:30 to 3:45 pm (during regular class time)
  - Arrive early to get seated
  - We will track your exam duration on Moodle so make sure you do not exceed the time limit *(except for people who require accommodations and contacted me for it)*

- At most 20 exam questions,
  - Combination of conceptual programming, short answers, and multiple choice questions
  - Be concise, but show your work

- Not allowed to take help from other people, online resources, and AI tools

Midterm Logistics

- **In-person Moodle Exam** with double seating
- Bring your computers to the exam location!

- **Location**:
  - HAS 20
  - ELABII 119

- **Seating based on Last name initial**
  - A-M (140 students) -> HAS 20
  - N-Z (90 students) -> ELABII 119

- **NOTE**:
  - Please arrive at least 15 minutes early to find your seat, open your computers, sign into Moodle, and settle-in before the exam starts
  - FULLY CHARGE YOUR COMPUTERS
  - MAKE SURE YOU ARE CONNECTED TO INTERNET and have ACCESS TO MOODLE

Exam Reading: Textbook, Datasheet and Online Resources

- Textbook: D. Molloy, “Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux, 2014”. Available in paperback (~$17 >) and Kindle (~$24) editions from Amazon. Also available at no charge on the O'Reily Safari Learning Platform: “Exploring Beaglebone 2nd Edition 2019”
  - Website for this book: `www.exploringbeaglebone.com`
- Datasheets:
  - Sitara AM335x ARM Cortex-A8 Technical Reference Manual (TRM): www.ti.com/product/am3358
  - BeagleBone Black System Reference Manual (SRM): tiny.cc/beagle104
- Navigate Beaglebone Black filesystem and software

Sample Exam Questions

C programming

Q. What is the output of this program?

1. 1 2 3 4 5
2. 1 12 3 4 5
3. 1 2 10 4 5
4. 10 2 3 4 5

```c
#include <stdio.h>

int main() {
    int numbers[] = {1, 2, 3, 4, 5};
    int *ptr;
    ptr = numbers;
    *(ptr + 2) = 10;

    for (int i = 0; i < 5; ++i) {
        printf("%d ", numbers[i]);
    }
    printf("\n");
    return 0;
}
```

C programming

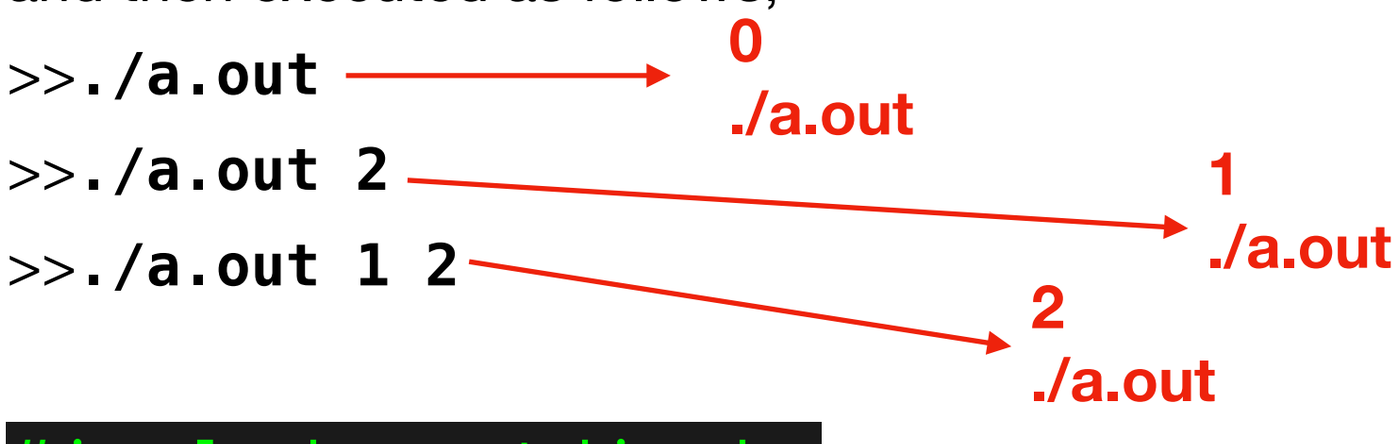
Q. The following C code is not running as expected. Explain the conceptual error in this code.

```c
int main(){
    int myArray[6] = {1, 3, 5, 2, 5, 10};
    for (int i=0; i<7; i++){
        printf("index %d has value %d\n", i, myArray[i]);
    }
    return 0;
}
```

The for loop should only iterate till index 5

C Program I/O

Q. Consider the following code. What are the outputs of this code after it is compiled, and then executed as follows,




```c
#include <stdio.h>
#include <stdlib.h>

int main (int argc, char *argv[])
{
    int x = atoi(argv[argc-1]);
    printf("%d\n", x);
    printf("%s\n", argv[0]);
    return  0;
}
```

File System Paths

Q. Write the absolute file path to the pwm directory

```
/sys/class/pwm
```

Q. Write file system navigation commands to navigate from pwm directory to home directory

```
cd ~
```

Q. The filesystem path for beaglebone GPIO is, /sys/class/gpio. Write the path to the Analog I/O directory

```
/sys/bus/iio/devices/iio:device0
```

File System Permissions

Q. Write a sample C code that opens a file “newfile.txt” in the current directory with write permission, writes integer values from 0 to 9 to the file, then closes the file.

```c
#include <stdio.h>

int main() {

    FILE *filePointer = fopen(“newfile.txt", "w");
    int i;
     for(i=0; i<10; i++){
        fprintf(filePointer, "%d\n", i);
     }
     fclose(filePointer); // Close the file when done using //fclose
    return 0;
}
```

Q. You created a new shell script “lab.sh” and when you execute it (./lab.sh), it gives you a permission denied error. Write a terminal command that will help you resolve the permission error.

```
chmod u+x lab.sh
```

Makefile

Q. Two files “hellomake.c” and “hellofunc.c” implement threads using pthread Linux API. “hellomake.h” is a header file. The following is a Makefile to compile these files. List one problem with this Makefile

```
CC=gcc
CFLAGS=-I.
DEPS = hellomake.h

hellomake.o: hellomake.c $(DEPS)
	$(CC) -c -o hellomake.o hellomake.c $(CFLAGS)

hellofunc.o: hellofunc.c $(DEPS)
	$(CC) -c -o hellofunc.o hellofunc.c $(CFLAGS)

hellomake: hellomake.o hellofunc.o
	$(CC) -o hellomake hellomake.o hellofunc.o

.PHONY: clean
clean:
	rm -f hellomake.o hellofunc.o
```

There is no -lpthread flag in the makefile to compile the threads

CPU Sharing

Q. A CPU sharing technique in which “Two or multiple programs take turns to use the CPU, and programs YIELD control voluntarily to let other programs use the CPU.” is known as:

1. Subroutines
2. Coroutines
3. Threads
4. Events

Threads

A C program implements two threads:

1. One thread waits for an input event on a pin

```
pthread_mutex_lock(&lock); // Lock during waiting
    int interrupt;
    interrupt = epoll_wait(epfd, &events, 1, -1);
    printf("Event Detected");
pthread_mutex_unlock(&lock); // Unlock after event detected
```

2. The second thread just prints the text of absence of an event to the terminal

```
   printf("Event is NOT Detected");
```

Q. What would happen if the input event never occurs?

The lock acquired by thread one will never be unlocked and the program gets stuck

Timing

Q. Write code that uses the clock_gettime(…) API and converts the time output to nanoseconds

```c
int main(){
    struct timespec now;

    clock_gettime(CLOCK_REALTIME, &now);
    long time_in_nanosec = now.tv_sec * 1000000000 + now.tv_nsec;
    return 0;
}
```

Interrupts

Q. To poll an input GPIO pin on beaglebone, we use “epoll” Linux API and monitor input events of interest. Which events should we look for when input GPIO pin value changes: EPOLLIN and/or EPOLLOUT and/or EPOLLET

`EPOLLIN and EPOLLET`

Q. A sensor data has arrived and caused hardware interrupt. At the same time, a divide by zero operation occurred in the main program and caused a software exception. Which interrupt is handled first?

Software Exception

Sensing

Q. What is the sampling resolution of an 8-bit ADC

2^8

Q. An 8-bit ADC maps to 0 to 5 V Analog input. How much voltage difference is between any two consecutive quantization levels of the 8-bit ADC

5/2^8

Sensing

Q. To prevent anti-aliasing, we need to sample the signal at a rate greater than the _____ rate

1. Euler
2. Fourier
3. Nyquist
4. Dirichlet

Multithreading versus Event-driven Approach

Q. An embedded system has two sensors (temperature and proximity sensor), two actuators (thermostat, screen), and one processor. Temperature readings are sampled periodically. A window of temperature samples are processed, based on which a thermostat actuator is controlled. The proximity sensor only generates a reading when it detects motion in vicinity. As soon as the motion is detected, screen is turned on for a duration of 10 seconds. Screen turns off in the absence of movement. Present a multi-threading approach and an event-driven approach to develop this embedded application. List drawbacks of each approach and explain which will be the optimal approach. You also have the option to come up with a hybrid approach (combination of both multi-threading and event-driven approaches)