1. **EXINU/WRT54GL: (15 points)**

a) What is WRT54GL? Mention three important components of WRT54GL.
b) What is a cross compiler? What did the cross compiler accomplish in your project 1?
c) Mention two significant differences in software development for programming projects and the XINU projects that you are working on.
d) List the differences between a realtime system and embedded system? Give an example each for a (i) realtime system,(ii) embedded system and (iii) realtime embedded system.
e) How did you realize the delay in your led-loop-driver of your project?

2. **(20 + 5 points) Finite state machine (FSM)**
(i) Craps is a popular game of dice. The rules are as follows:
- a) You roll two dice with each die having values {1-6}
- b) After the initial roll, the sum of the two is computed.
- c) If the sum is 7 or 11 in the first roll you win
- d) If the sum is 2, 3 or 12 in the first roll you lose
- e) If the sum is other than (i) and (ii) above (i.e. 4, 5, 6, 8, 9 or 10) on the first throw, the sum becomes players's points
- f) You keep rolling the dice to make the point, that is, roll the point again to win. During this roll again, if you throw a 7 or 11 you lose.
- g) Once you win or lose the game ends.

Draw a Finite State Machine (NOT A FLOWCHART) to model this problem. (what I given above is the flowchart/algorithm.)

3. **(15 points) Task creation and communication using pipes**
a) Draw the tasks and the pipes between them that are described by the following code segment.
b) Draw the task descriptor table with the relevant points and index numbers.
c) What is the output?

```
pid_t newpid;
int fd1[2], fd2[2], fd3[2];

char m0[] = "\n about to fork ..\n";
char m1[] = "message 1\n";
char m2[] = "message 2\n";
char m3[] = "message 3\n";
char m4[] = "\n done ..\n";

char rbuf1[256];
char rbuf2[256];
char rbuf3[256];
int cn1, cn2, cn3;

if ((pipe(fd1)== -1)) printf(" error \n");
if ((pipe(fd2)== -1)) printf(" error \n");
if ((pipe(fd3)== -1)) printf(" error \n");

if ((newpid = fork()) == -1) { printf("error \n"); return 0;}
```

```
 if (newpid > 0) {

   close(fd1[1]);
   close(fd2[1]);
   close(fd3[0]);
   write(8, m3, sizeof(m3));
   usleep(10000);
   cn1 = read(3, rbuf1, 256);
   cn2 = read(5,rbuf2,256);
   write(1, rbuf1, cn1);
   write(1, rbuf2, cn2);

 }
 else {
   close (fd1[0]);
   close (fd2[0]);
   close (fd3[1]);
   write(4, m1, sizeof(m1));
   usleep(10000);
   write(6,m2,sizeof(m2));
   usleep(10000);
   cn3 = read(7, rbuf3, 256);
   write(1, rbuf3,cn3);
 }
 usleep(10000);
 write(2,m4,sizeof(m4));

 return 0;
}
```

4. **(15 points) Signal and handler**
a) Write a pseudo code that defines a signal for constant 44 and a handler that counts the number of occurrences of this signal. After it counts it, it resets the signal to allow it re-occur.
b) Write the main program (pseudo code) that loops for ever: inside the loop, it prints the count, it then sleeps for random time and gets interrupted by signal 44 from the keyboard.
c) How will you interrupt (signal) the main program with a signal 44?

5. **(15 points) Memory management**
List 5 problems encountered when dealing dynamic memory. Explain with an example each.

6. **(15 points) Device driver**
a. What is device driver?
b. What are the steps in defining a device driver?
c. What are the data structures and algorithms? Explain with reference to XINU directory structure and files.

7. Hand simulate the execution of the following program (to show your understanding of the pointers) and

```
#include <stdio.h>
#include <string.h>

void  unknown(int* q, int a, int b) {

  *q = *q + a  *  b;  // #1  : what are the values of q, a and b?
  return;
}

int  main() {

  int *op1 = new int;
  *op1 = 2;
  int op2 = 145;
  int z  =   234;

  unknown (&z, *op1, op2);

  printf(" %d   %d  %d \n", *op1, op2, z);  //#2 : what is the output?

  op1 = &op2;
  unknown (&z, *op1, op2);
  printf(" %d   %d  %d \n", *op1, op2, z); //#3:   what is the output?

  return 0;
}
```

8. (10 points) Mutual exclusion & Process synchronization
Use XINU semaphores to solve the problem given below. Consider a treasure hunt game with multiple teams each with 3 participants. This game has a treasure chest located in a jungle. The chest has gold bars and can be accesses by only one participant at any time. Chest is mobile and it keeps moving around in the jungle. Members of each team have to locate the chest and retrieve the gold bars one at a time and bring it back to their home base. Whichever team brings back 10 gold bars first is the winner. Each participant will take random amount of time to reach the treasure chest depending on their survival capabilities in the jungle. Write the multi-processor pseudo code for a solution to this problem.