1. Define the characteristics of a **microprocessor** and **microcontroller**. Explain the difference between a microprocessor and microcontroller. (Note: Simply stating one processes and the other controls does not suffice and will result in no credit). *(5 points)*

- Microprocessor: Purdue calculating unit, optimized for high throughput.
- Microcontroller: Calculation and electrical features together, on-chip nonvolatile memory.

2. What is meant by memory mapped I/O? *(3 points)*

   Input/Output systems accessible by writing to specific addresses.

3. Suppose you have an `uint32_t` variable `y` that you have some value in. Write some C code fragments to perform the following operations on `y`.
   a) Clear(make zero) bits 0, 2, and 31 without changing any of the other bits. *(3 points)*

   \[ y = y \& \neg (0x80000005); \]

   b) Set (make one) bits 8, 13, and 23 without changing any of the other bits. *(3 points)*

   \[ y = y | 0x00802100; \]

   c) Negate all bits (all 1's become 0's and all 0's become 1's). *(3 points)*

   \[ y = \neg y; \]

4. Explain the concept and significance of a watchdog timer. *(5 points)*

   A watchdog timer counts independent of system software and triggers an interrupt if it is ever not reset, allowing it to detect if software has hung up.

5. Why is the use of an int frowned upon, especially in embedded systems development? *(5 points)*

   The size of this data may be different depending on compiler settings, which can be a hazard to efficient operation.
6. What does the C volatile type qualifier do? **(3 points)**

   Refers to data which may be changed by other code. Also prevents tests of it from being removed by the compiler.

7. What does the C static storage class specifier do? **(3 points)**

   Keeps variables from being re-initialized.

8. Are pointers in C and references in Java the same thing? Give an example to illustrate your answer. **(3 points)**

   Not references in Java are *strongly typed* and cannot have pointer arithmetic done to them.

9. What will be output of following program? **(5 points)**

   ```c
   #include <stdio.h>
   int main()
   int a = 320;
   char *ptr;
   ptr = (char *)&a;
   printf("%d", *ptr);
   return 0;
   }
   ```

   320

10. What does the following C code print? **(5 points)**

    ```c
    uint32_t SomeNum = 0x12345678;
    uint8_t *BytePtr = (uint8_t *) &SomeNum;
    printf("0x\%x\r\n", *BytePtr);
    ```

    0x44

   Why does or does not endianness apply?
11. What does the following C code print on a big endian processor? What does it print on a little endian processor? Comment on your responses. \( (5 \text{ points}) \)

```c
uint32_t SomeNum = 0x11223344;
uint32_t NewNum = SomeNum >> 16;
printf("0x%x\n", NewNum);
```

a) Big endian processor:

b) Little endian processor:

c) Why does or does not endianness apply?

12. Write a C code fragment that shows how the memory is allocated. \( (8 \text{ points}) \)

a) stack:

b) heap:

c) global:
13. Name a typical use for each kind of storage and why it is a reasonable use of the storage. (9 points)
   a) stack:

   b) heap:

   c) global:

11. What do you use to retain the state of a register in your program for write-only register bits? (5 points)

15. When would a read modify-write operation on a register be appropriate? When would it not be appropriate? (5 points)
16. Write a `#define` for a pointer to `SOME_REGISTER` at address `0xDEADBEEF` that is 32-bits wide. (3 points)

```c
#define (uint32_t *)0xDEADBEEF
```

17. Suppose you want to use a timer interrupt. List the things required for this to work. (10 points)

- write a service routine for the timer
- copy the routine's pointer to the interrupt vector table
- enable interrupts for the timer
- set the timer's registers for length, etc.

18. Draw a block diagram of how source code is translated to object code. (5 points)

```
main
    \rightarrow \text{Precompile}
    \downarrow
main.c, headers
\rightarrow \text{Compile}
\rightarrow \text{Object code}
\rightarrow \text{Link}
\rightarrow \text{Finished code}
```

19. Draw a top-level block diagram of the RoboTank system. Only include the major functional blocks that should be used (one block is not sufficient), for this to work. (10 points)

```
Distance Sensors \rightarrow LCD \rightarrow Beaglebone \rightarrow USB Serial \rightarrow \text{Network}
\downarrow                  \downarrow                  \uparrow
\rightarrow \text{Bridge} \rightarrow \text{Wheels}
```
20. Draw a top-level UML data/control flow diagram for the RoboTank. Include the ISR’s, main(), and shared data. (10 points)

21. What is the difference between a hard, soft, and firm real time operating system? Are there any time constraints on the RoboTank and can this system meet them? (10 points)

Soft Realtime → Missed deadline is reduced performance
Firm Realtime → Missed deadline invalidates work
Hard Realtime → Missed deadline is system failure.

22. Suppose an ISR runs at a periodic rate of 1024 Hz and places one 32-bit sample in a queue each time it runs. The OS has a tick rate of 100Hz (10 ms). Meanwhile the task runs every OS tick, is the highest priority task in the system, consumes all the samples until it is empty, and then pends until the next OS tick. What is the minimum size of the queue required in bytes? (5 points)

Since it is possible to generate 17 new samples before the task that uses them runs, that is how long the buffer must be.
23. Describe how semaphores are used. What is the purpose of a semaphore? Discuss binary semaphores and counting semaphores. (10 points)

Semaphores block access to a resource so conflicting requests will not happen. Binary semaphores protect unique resources while counting semaphores manage groups of tangible resources.

*interchangeable

24. Write some pseudo-code which creates a deadlock using semaphores. Fix the deadlock. (10 points)

```
1. sub1
   acquire (resA), acquire (resB)
3. sub2
   acquire (resB), acquire (resA)
2. sub2
```

1. sub2 interrupts sub1 and acquires B
2. sub2 cannot continue until A is released
3. sub1 cannot continue until B is released

**Fix:** Reverse the order of holds in sub 2.
25. Write some pseudo-code which creates priority inversion. Fix the code using priority inheritance. (10 points)
26. What is the difference between RAM and ROM memory? (3 points)

Obvious, hopefully.

27. What is the difference between a preemptive and a non-preemptive system? (3 points)

Preemptive systems may halt tasks without their calling a sleep method.

28. What is the difference between System Specifications and System Requirements? (3 points)

X

29. What are the differences between macros and inline functions? (3 points)

X

30. What are the advantages and disadvantages of global variables? (3 points)

X