04:32

MobileKnowledge Technical Test

* Required

Software Engineer Q&A

2

MCU PERIPHERALS
You need to read a given value from a device connected to a GPIO on a MCU. That value changes over time and you need to keep up to date, detect when in changes and make some calculations with that value What are the two main mechanisms you can use to read that GPIO? *

Can you describe them?

There are two main mechanisms:
- interrupt: write an interrupt in the operating system that happens when the value is changed (the operating system has to provide the interrupt mechanism)
- polling: read the value at given intervals.

3

ENDIANNESS
Do you know what endianness is? What are the two main kinds of endianness? Do you know any architectures using one or the other? *

When do you think endianness has to be taken into account

Endiannes is the order in which multi-bytes variables are stored.
Two types:
- big endinan (motorola)
- little endian (intel)

In ARM architecture the endianess is configurable

4

VOLATILE
Do you know what volatile means in C? What does it do? *

How is typically volatile used in embedded systems?

It reports to the compiler to not to make optimizations on the access to the given variable, so it is accessed in the specified way. By the way "volatile" is not atomic, the aproach is different. "volatile" is commonly used in embedded system to read the memory positions that report input values from sensors and such.

5

STATIC
Do you know what static means in C? What is it used for? *

The key word "static" has two meanings in C: A "static" function, that means it is onl

6

BITWISE OPERATORS
You have the following declaration: “uint8_t a”. Please write the code to: *

Set the least significant bit in a:
Set the sixth bit:
Set the ith bit:
Unset the ith bit:

a &= 0x01
a &= (1<<6)
a &= (1<<i)
a &= ~(1 << i)

7

PREPROCESSOR MACROS
Do you know what preprocessor macros are and how they work? *

Can you see any errors in the following macro: #define MUL(a,b) a * b
How would you fix the macro?

That macro doesn't check types. So if the types don't use that * operation it would crash.
The expansion can be wrong, so it would be more precise to write: (a)*(b)

8

POINTERS AND MEMORY LOCATIONS
Read the declarations and the piece of code below: *

Can you answer to the following questions?
1) What is the value of the variable iters in function()?
2) Can you tell what will be the output in function()?
3) Can you tell the output in main()?

```c
#include "stdio.h"
#include <stdint.h>
#include <string.h>

typedef struct {
    uint8_t first_array[16];
    uint16_t value;
    uint8_t second_array[16];
} data_t;

static void function(data_t *data) {
    int i;
    uint16_t *ptr = (uint16_t *) data;
    int iters = sizeof(data_t) / sizeof(uint16_t);

    for (i = 0; i < iters; i++) {
        printf("ptr = %x\n", ptr);
        *ptr = (uint16_t) i;
        ptr++;
    }
}

int main(void) {
    data_t data;
    memset(&data, 0, sizeof(data_t));
    function(&data);
    printf("%d\n", data.value);
}
```

1) iter = 17 ;
2) function(data_t *data) is void so it not returns any output, but what it does is to give value to the first 17 uint16_t of the data structure.
3) data.value = 0x08

9

ARM PLATFORMS

Can you briefly explain what semihosting is and what is it used for? No need to explain in great detail. *

remote debugging