Embedded Systems Engineer Test

Read instructions carefully. In case there is information not specified you may assume the condition you feel most comfortable with and indicate it explicitly on comments on your answer.

1. Embedded C Programming

Overview

We (Smartenit) manufacture a device that is a Zigbee Load Controller. It provides the ability to control and monitor the Load, as well as measure the Energy usage of the Load. Normally the user read/write the data wirelessly through another Zigbee device.
This device has a serial port for programming new firmware into the device. One of our customer has requested the ability to read the Energy usage details through a serial port for diagnostics. So basically, the customer wants to be able to attach a serial cable to read the data directly from the device. The customer must then be able to connect to the serial port on the device via Putty or similar utility and retrieve desired data.

Requirement

To fulfill the requirement, the following diagnostics commands should be interpreted by the device when it gets the string from the serial port in ASCII format:

1. "read voltage": when the device gets this command from the serial port it should print out the actual voltage in the line.
2. "read current": when the device gets this command from the serial port it should print out the actual current demanded by the load.

Expectation

You will need to write a C module (both source and header file) for the device to achieve this new functionality to read data through serial port. The following assumptions/indications are valid:

- We have provided you some helper files, named below and source at the end.
  - **`serial_port.h`**
  - **`app.h`**
- You can assume that the serial port is already configured and initialized and that it is possible to read/write data with the functions in **`serial_port.h`**. Only those functions are available for reading/writing to the serial port.
- In case a command is formatted incorrectly, an error message should be printed: `"\nError: wrong format\r\n"`
- Every character entered by the user should be echoed back.
- A command line entered by the user is confirmed by hitting enter, after a carriage return is received the command entered by the user must be parsed and executed on success.

- The user can verify if the device is active by sending carriage return, to which the device must send the string:
  `“\nOK\r\n”`
- The “read” command must accept only 1 parameter, which is the variable meant to be read. The actual value can be extracted from the functions in **`app.h`**. User expects voltage and current data to be in a readable format:
  > read voltage
  Voltage: 123.45V
  > read current
  Current: 8.79A
- You can assume any information not explicitly detailed as you find more convenient. There is no need to compile the program, although you can use external utilities to verify syntax.
- **Bonus:** Can you provide the user the ability to erase mistyped letters before they hit enter?

```
/* From "serial_port.h" */

 /* SER_ReadByte: Reads a byte from the Serial Port Input FIFO
 * /param [OUT] pu8ReadByte: Pointer of memory where the read byte will be stored
 * /retval     bool_t    : [TRUE: A byte was read from serial port FIFO]
 *                        [FALSE: No data was read*/
bool_t SER_ReadByte(uint8_t * pu8ReadByte);

/* SER_WriteByte: writes a byte to the serial port output FIFO
 * param [IN] u8Byte: Byte to write to the port */
void SER_WriteByte(uint8_t u8Byte);
```

```
/* From "app.h" */

void APP_RelayCtl(uint8_t u8RelayNumber, uint8_t u8Action);

/* APP_ReadVoltage: Returns the current voltage in the line
 * /retval     u16Voltage: Voltage in 10mV units.
 *                         i.e: if function returns
 *                         12345, voltage is 123.45V */
uint16_t APP_ReadVoltage(void);

 /* APP_ReadCurrent: Returns the current in the line
 *
 * /retval     u16Current: Current in 10mA units.
 *                         i.e: if function returns
 *                         789, current is 7.89A*/
uint16_t APP_ReadCurrent(void);
```

2. Embedded Hardware

The following circuit will be used for sampling of an analog signal that spans from 0V to 5V. You can select a microcontroller with built-in ADC peripheral you are most familiar with and answer the following questions:

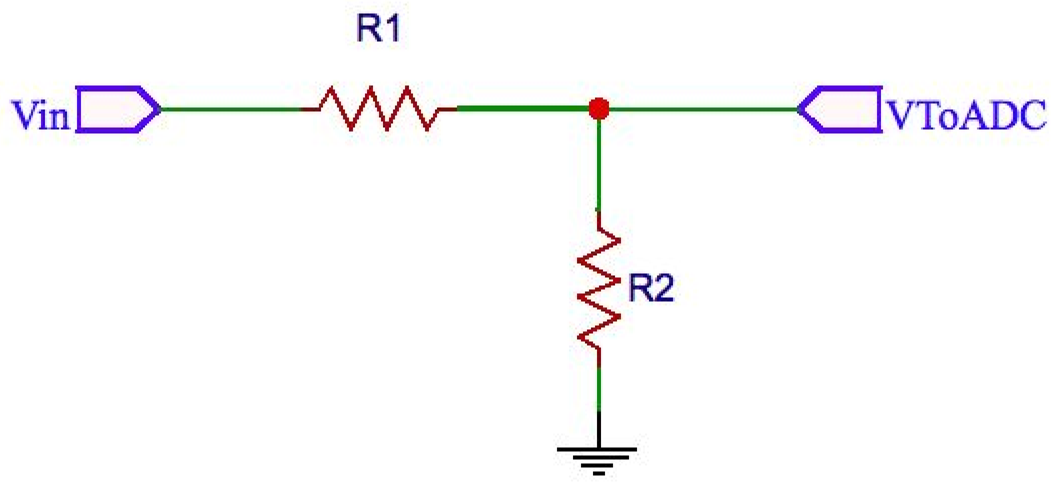

Figure 1. Circuit for signal conditioning to ADC

- What values of R1 and R2 would you select to achieve correct sampling of the input signal?
- Which limitations does the circuit on Figure 1 impose on the signal to be connected to it?
- For the selected microcontroller, what resolution will be achieved?
- What modifications would you do to the circuit to make it more robust to impedance variations in the source signal?
- What modifications would you do to the circuit to prevent damage in case of overvoltage or overcurrent?

3. Approvals and Control

Change log

Date	Prepared by	Version	Description

Approvals

Approved by	Approved version	Job Position	Date