EE 5314 Embedded Microcontroller System Design
Fall 2017 Project (revised)

1 Overview

The goal of this project is to implement a peer-to-peer communication system, supporting a carrier sense multiple access (CSMA) protocol.

The construction phase will require building a node capable of interfacing with a PC so that text commands can be entered by the user. Based on the commands, subsequent transmission on the 2-wire RS-485 bus to other nodes will occur.

The node will also extract information out of the asynchronous data stream and to control three or more devices. It will send acknowledgements back to the controller over the RS-485 physical layer to indicate successful receipt of the data. The transceiver will also send unsolicited messages back to the controller in response to changes in input device status.

A collection of most major parts will be provided to each single-person team. The pc boards, tools, and any optional items are not included in this collection of parts.

2 Two-Wire RS-485 Link

The RS-485 specification details the electrical properties of the interface. In particular, it specifies that differential signals be utilized and also specifies the voltage levels to be used for signaling a high and low logic level. The RS-485 interface allows up to 32 standard loads (devices nodes) on a shared data bus.

In the RS-485 2-wire configuration, a single differential pair (link) is provided. The controller and the device nodes share the same link for communications. The data rate will be 38400 baud. A termination of 120 ohms should be added across to the physical ends of a long wire run to allow the bus to be properly terminated.

Since the pair can be driven by more than one node at a time, a potential exists for a collision on the bus. In this project, any message sent shall be repeated at a delay calculated by a binomial exponential back-off algorithm if an acknowledgement is not received in an acceptable period of time. To prevent the possibility of duplicate messages being received, a message identifier is attached to each transmitted message to allow the recipient to determine message uniqueness.

A data is transmitted in groups of 11 bits, consisting of a start bit (‘0’), a byte, an address (‘1’) / data bit (‘0’), and a stop bit (‘1’).

3 Node Hardware

The hardware must be soldered together. No breadboards or wire-wrapping are allowed.

Microcontroller:
An ARM M4F core (TM4C123GH6PMI microcontroller) is required.

Serial interface:
If using the EK-TM4C123GXL evaluation board, then the UART0 tx/rx pair is routed to the ICDI that provides a virtual COM port through a USB endpoint.

RS-485 interface:
A SN75HVD12 is used to convert the CMOS logic level signals to/from the PIC to the differential signals used for the 2-wire RS-485 interface. The receiver enable (RE) pin of the SN75HVD12 is always enabled. Since only one node can talk on the bus at a time, the ARM asserts the driver enable (DE) pin of the SN75HVD12 only when it needs to transmit on the bus.

Power LED:
Connect a yellow LED with a 470ohm current limiting resistor to indicate power on the interface board.

Input device:
You must connect one or more input devices. At a minimum, a pushbutton must be used. An on-board pushbutton on the EK-TM4C123GXL can be used.

TX LED:
Connect a red LED with a 470ohm current limiting resistor to a GPO on the controller. Do not use the LED on the EK-TM4C123GXL board as it is too hard to distinguish multiple LEDs at the same time.

RX LED:
Connect a green LED with a 470ohm current limiting resistor to a GPO on the controller. Do not use the LED on the EK-TM4C123GXL board as it is too hard to distinguish multiple LEDs at the same time.

Output device:
You may control at least 3 devices such as a light bulb (low voltage only), servo, RGB LED, or monochrome LED (not red, green, or yellow as those are used above). Interfacing will depend on the device controlled.

Connections:
A 2-pin terminal block will be used to provide access to the RS-485 signals. You must connect this to additional nodes to test your hardware.

4 Suggested Parts List

Part	**Quantity**
EK-TM4C123GXL (rs-232 receiver/driver)	1
SN75HVD12 (rs-485 transceiver)	1
Yellow LED (power)	1
Red LED (tx)	1
Green LED (rx)	1
470ohm resistor (led current limiters)	3
0.1uF capacitor (bypassing transceiver)	1
8pin 300mil socket (sn75hvd12)	1
10x2 100mil pitch unshrouded header	2
2-pin terminal block	1
Wire (22-24 AWG solid wire, 3 colors)	1
Power supply of your choosing if not using the EK-TM4C123GXL	1
PC board	1
Device under control and interfacing	Varies
Solder, iron, needle-nose pliers, diagonal cutters, safety glasses, wire, cable, …	1 each

5 Data Packets

A packet shall consist of the following bytes in order:
DST_ADDRESS: the address of the node that will receive the packet (with the 9th bit = '1')
SRC_ADDRESS: the address of the node that sent the packet
SEQ_ID: a unique packet sequence number (can be a modulo 256 value)
COMMAND: the action being reported (the "verb")
CHANNEL: the channel of the node to which this message applies (the "noun")
SIZE: the number of data bytes being included
DATA[n]: n = 0, 1, … SIZE-1 bytes of data specific the packet being sent
CHECKSUM: the 1's compliment of the modulo-256 sum of all 6+SIZE bytes in the packet

Addresses 0-254 are designed for unicast addressing of nodes. Address 255 is a broadcast address that instructs all nodes to process the packet. Address 0 is reserved for the controller.

The following commands are defined for this project:

COMMAND & 0x7F	Action	SIZE	DATA Arguments
Control Commands			
0x00	Set	1	Value
0x01	Piecewise*	5+N	N states (1 byte), Value[n] (1 byte) where n = 0, 1, …, N-1, Dwell in 10 ms (2 bytes), Cycles (2 bytes)
0x02	Pulse	3	Value (1 byte), Duration in 10 ms (2 bytes)
0x03	Square	8	Value1 (1 byte), Value2 (1 byte), Time1 (2 bytes), Time2 (2 bytes), Cycles (2 bytes)
0x04	Sawtooth*	7	Value1 (1 byte), Value2 (1 byte), Delta (1 signed byte), Dwell per step in 10 ms (2 bytes), Cycles (2 bytes)
0x05	Triangle*	8	Value1 (1 byte), Value2 (1 byte), Delta1to2 (1 signed byte), Delta2to1 (1 signed byte), Dwell per step in 10 ms (2 bytes), Cycles (2 bytes)
Data Commands			
0x20	Data Request	0	Requests that a data value be reported
0x21	Data Report	1	Data sent unsolicited or as a response to 0x20
0x22	Report Control*	1	Delta variation in value required for a report be sent; 0 = report disabled
UI Commands			
0x40	LCD Display Text*	N	N characters to be displayed
0x48	RGB*	3	R, G, and B PWM values (brightness)
0x49	RGB Piecewise*	5+3N	N states (1 byte), RGB[n] (3 bytes) where n = 0, 1, …, N-1, Dwell in 10 ms (2 bytes), Cycles (2 bytes)
Serial Commands			
0x50	UART Data*	N	N characters to be sent
0x51	UART Control*	3	Baud rate in units of 10 bps (2 bytes), Control: | set break | parity_en | parity_type: 00=space, 01=mark 10=even 11=odd | ? | ? | tx_on | rx_on | (1 byte)

0x54	I2C Command*	2+N	Address, N bytes to send, Data[n] where n = 0, 1, …, N-1
System Commands			
0x70	Acknowledge	1	SEQ_ID of received packet
0x78	Poll Request	0	None
0x79	Poll Response	1	Currently assigned address
0x7A	Set Address	1	Address to assign to node
0x7D	Node Control*	1	Control: | ? | ? | ? | ? | ? | ? | ? | tx_enable |
0x7E	Bootload*	0	None Note: Disable all transmitters with 0x7D cmd first
0x7F	Reset	0	None

Notes: Words are sent LSB first
* optional

If the MSb of COMMAND is set, then the node shall send an Acknowledge message.

The channel is the device to be controlled on the addressed node. This byte is not defined for system-level commands and shall be set to 0.

6 Node Software

User Interface:
The controller shall provide a text-based user interface that can be accessed with a terminal emulator program running on the PC. The data format is 115200baud, 8N1. At a minimum, the following command must be supported:

All commands below should be case-insensitive.

On startup, the node should send “Ready” to the host machine on the serial interface.

The PC can send several commands to the controller:

If “reset A” is received from the PC, a reset is sent to address A.

If “cs ON” or “cs OFF” is received from the PC, carrier sense detection is enabled or disabled.

If “random ON” or “random OFF” is received from the PC, then random retransmissions is enabled or disabled.

If “set A, C, V” is received from the PC, where A is the address, C is the channel, and V is the value, a set command shall be sent.

If “get A, C” is received from the PC, where A is the address and C is the channel, a data request shall be sent.

If “poll” is received from the PC, a poll request shall be sent.

If “sa A, Anew” is received from the PC, where A is the current address and Anew is the new address, the controller shall send a set address command.

If “ack on” is received from the PC, the controller shall request an acknowledgement for each command sent.

If “ack off” is received from the PC, the controller shall not request an acknowledgement for any command sent.

If an errant command is received from the PC, an “Error” response shall be sent.

After a valid command is received from the PC, the controller shall respond with “Queuing Msg N”, where N is the unique message ID assigned to the message.

Any time a message is transmitted on the RS-485 bus, a node shall send a message “Transmitting Msg N, Attempt M” to the PC showing the message number (N) and the transmission attempt (M). If acknowledgement was requested and the maximum number of retransmissions was sent without acknowledge, an “Error Sending Msg N” message shall be sent to the PC.

Any poll responses and data reports received on the RS-485 bus shall be sent to the PC in a text format of your choosing.

Packet Transmission:
If carrier sense is enabled, then the node should determine that a carrier is not active prior to starting transmission. While this does not ensure that a subsequent transmission will be contention-free, it helps to mitigate collisions.

To address a node, the packet starts with an 8-bit node address with a 9^{th} bit set to ‘1’. Subsequent data transmissions have the 9^{th} bit cleared. For each new message (not retransmissions, the SEQ_ID field should be incremented, modulo 256.

When a message is sent, a request can be made to require an acknowledgement or response. If the sending node does not receive an acknowledgement within a period of time ($T \times 2^n$; where n = 0 initially), then it shall;

1. Increment n.
2. Wait a fixed time T0.
3. If random retransmission is enabled, wait a random period of time over an interval of 0 to $T \times 2^n$ seconds. If random retransmission is disabled, instead wait a period $T \times 2^n$ seconds (debug mode).
4. Reattempt transmission using the identical message contents (same SEQ_ID).

This procedure is repeated for subsequent timeouts for $0 < n \leq N$; where N is the maximum number of retries allowed.

Packet Reception:
If an address match is detected, then the receiver shall collect all data bytes in the packet. Based on the contents of the packet, an acknowledgement may be sent. If the message received is an acknowledgement of a message earlier transmitted, then cross-check the received SEQ_ID with pending acknowledgements so that retransmissions stop.

Upon receiving a message requesting an acknowledgement or response or an input device changing state, the peripheral shall send a packet. If the receiving node does not receive an acknowledgement within a period of time ($T \times 2^n$; where n = 0 initially), then it shall:

1. Increment n.
2. Wait a fixed time T0.
3. If random retransmission is enabled, wait a random period of time over an interval of 0 to $T \times 2^n$ seconds. If random retransmission is disabled, instead wait a period $T \times 2^n$ seconds (debug mode).
4. Reattempt transmission using the identical message contents (same SEQ_ID).

This procedure is repeated for subsequent timeouts for $0 < n \leq N$; where N is the maximum number of retries allowed.

TX LED:
The TX LED shall blink to indicate that a message has been transmitted. If the number of retries is reached, then the TX LED shall be illuminated to indicate an error (the next transmission attempt will blink and then clear this "error" condition on the LED).

RX LED:
The RX LED shall blink to indicate that a message addressing this node (broadcast or unicast) has been received by the node. This LED shall blink on power-up to indicate that the controller has booted correctly.

Output Device(s):
The software shall control one or more devices, each assigned to a channel.

Input Device(s):
Each input device should be assigned a channel. The software shall send a Value message on the reverse link when the input device status changes or when a Get command is received.

Control Commands:
When a control command is received, the output device selected shall be controlled. Some devices, such as RGB triplets, LCD displays, and serial devices can ignore these commands.

Sensor Commands:
When a Sensor Request command is received, the input device should be read and the value should be returned in a Sensor Data command.

Set Address Command:
If a Set Address command is received, then the address of the node shall be changed. The new address should be written to non-volatile memory so that the value is persistent after power loss.

Poll Request Command:
If a Poll Request command is received, the node shall transmit its address in a Poll Response message.

Reset Command:
If a reset command is received, the node shall reboot.

7 Testing

Your hardware will be tested on a bus with other nodes. Please verify operation with other nodes and the PC application prior to the defense date.

Computers and lab equipment will be provided on campus in Rm 148 NH for you to work on this project. If you do plan on plugging your project into your own machine, do so at your own risk and only after having the hardware tested. Again, you are responsible for anything that happens to your personal machine. Do not connect your project to any machines in the UTA computing labs or in other EE labs.

8 Deadlines

It is suggested that you complete construction of the hardware by 10/4. Beginning on 10/9, a 25% deduction to the hardware portion of the project will be assessed for each class period that the hardware is late. The hardware is worth 20% of the overall project grade.

Project is due at the time indicated in the syllabus. An electronic copy of you code must be received by the graders prior to 5:30pm on the day of the defense to receive points on the project. A written report (containing theory of operation and software printout) is required. You must also demonstrate your hardware and software (including compilation on site). This is an independent project.

9 Safety Issues

While far beyond the scope of this document, it is important to use tools safely. Safety goggles are a good idea, since you can cause yourself injury if a wire that is being cut flies into your eye. Similarly, if wires being unsoldered are placed under some strain, the solder can be fling toward you. Soldering entails some care to prevent burning yourself or a burning down a building if you forget to turn it off. If you choose to use solder containing lead, then care shall be taken to dispose of lead properly (don't cool off the iron in a drinking fountain, etc.). Always wash your hands after using solder to prevent the build-up of heavy metals in your body. These are a few helpful suggestions and are a very incomplete listing. Please read and understand all safety labels and exercise caution. You will be required to attend a safety orientation prior to use the 148NH lab.

Please utilize the supervised lab resources in Rm 148 NH when working on the project for your safety. You may only use the resources in Rm 148 NH when the GTA or other E.E. staff is present.

Have fun!