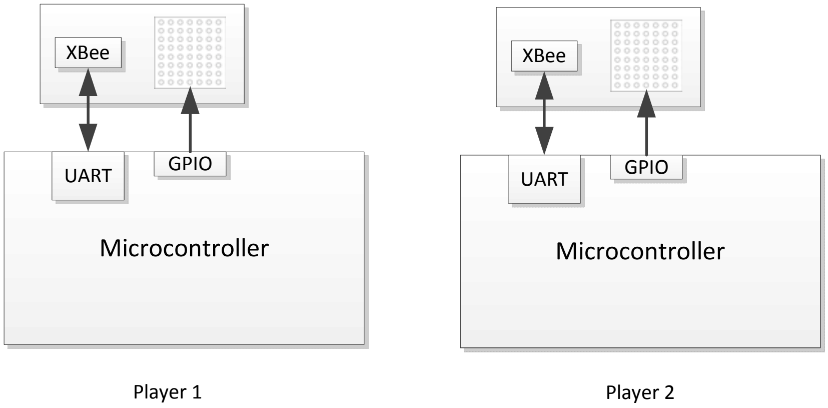
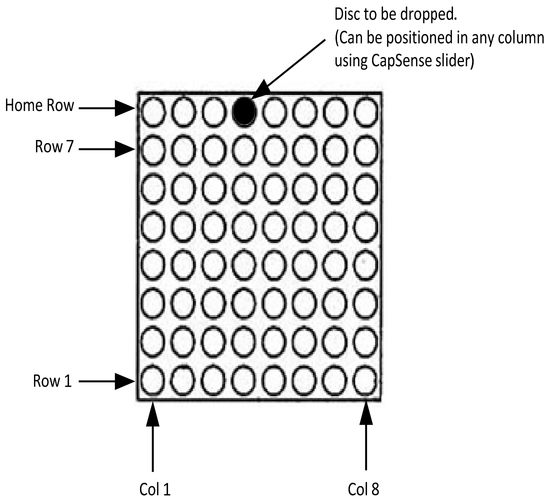
CMPE 121L: Microprocessor System Design Lab
Spring 2013
Lab Project: Connect Four Game

In this project, you will use the PSoC-5 microcontroller, an 8x8 RGB LED display, and an XBee wireless module to design a two-player Connect Four game. You will do the design independently, but will need to work in two-person teams to test and demonstrate your design.



A block diagram of the system is shown above. The row and column inputs of the 8x8 LED display are connected to the GPIO pins of the microcontroller, and the XBee module is connected to the serial pins of a UART block in the microcontroller. User input is through the CapSense button and slider on the microcontroller board.

A good description of the Connect Four game can be found in Wikipedia (http://en.wikipedia.org/wiki/Connect_Four). You need to use a grid with 8 columns and 7 rows. Each grid-point is represented by an LED in the 8x8 matrix, leaving the top row as a "home" row for dropping discs into the grid. The discs belonging to the two players will be distinguished by distinct LED colors.



When the game is being played, each player makes moves (to drop discs) using the keys on his/her microcontroller board. At any time during the game, the displays of the two players should be synchronized to show the same state of the discs. This is accomplished by the microcontroller of each player communicating each of his/her moves to the other microcontroller. That is, when Player 1 makes a move, his/her local display is updated to show the newly dropped disc. The move is also communicated to Player 2’s system, which then updates its display to show the new disc configuration.

After each move, the microcontrollers must check for a pattern of four discs along the same line, and should terminate the game if such a pattern is found.

Interfacing the 8x8 LED Matrix

A datasheet of the LED display can be found at

http://shop.emscdn.com/im/meggyjr/BL-M23B881RGB.PDF.

This display has a total of 192 LEDs (64 of each color). These are arranged in a row-column format with 8 anode pins and 24 cathode pins. Only 16 of the 24 cathode pins need to be connected to display 2 colors. The 24 pins (8 anodes and 16 cathodes) need to be connected to GPIO pins of the microcontroller. Don’t forget to use series resistors to limit the current through the LEDs (330 ohms or higher recommended). Make sure that the GPIO pins of the microcontroller can source and sink the necessary current when all the LEDs of a row/column are ON. You may use the PNP transistors in the kit to drive the anodes of the LEDs to increase the current.

To display a pattern of discs on the LED display, you will need to scan the LEDs one row at a time or one column at a time, and refresh them at a frequency high enough to eliminate flicker. This can be done using a timer to interrupt the microcontroller at periodic intervals, and refreshing the next row/column within the ISR. You may use the 74HC595 shift registers to shift in the data serially from the microcontroller.

74HC595 data sheet: https://www.sparkfun.com/datasheets/IC/SN74HC595.pdf

Using the XBee Modules

The following are two key references for designing with the XBee modules.

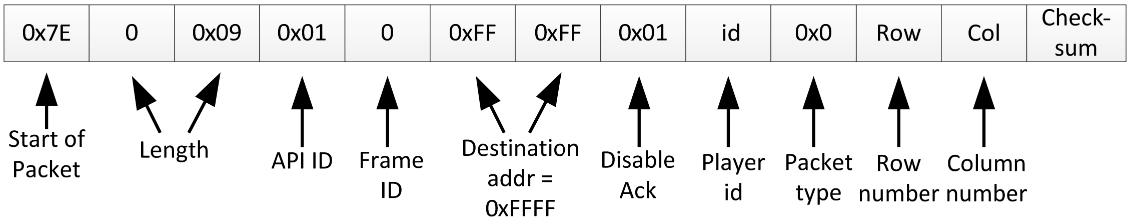
XBee Product Manual: http://ftp1.digi.com/support/documentation/90000982_H.pdf

XBee Tutorial: https://courses.soe.ucsc.edu/courses/cmpe121/Fall12/01/pages/syllabus/122-32450-xbeetutorial-v1.0.pdf

The XBee module is to be connected to the UART pins of the microcontroller. Out of the 20 pins of the XBee module, only 4 need to be connected (ground, 3.3V power, data in, data out).

The other XBee pins can be left unconnected. The 3.3V power can be generated from 5V using the regulator provided in the parts kit, or can be directly obtained from the 3.3V power rail of the microcontroller board. Be aware that the XBee device inputs are not 5 Volt-tolerant. Therefore, you will need to either set the I/O voltage on the microcontroller board to 3.3V, or use resistors to divide the voltage on the output pin to within 3.3V.

When a new move is made a player during the game, his/her microcontroller should send a packet through the XBee with the following format:



0x7E	0	0x09	0x01	0	0xFF	0xFF	0x01	id	0x0	Row	Col	Check-sum
Start of Packet	Length		API ID	Frame ID	Destination addr = 0xFFFF		Disable Ack	Player id	Packet type	Row number	Column number	

The packet id is an 8-bit identifier selected by the player originating the packet, to help the destination microcontroller identify the player. The packet type of 0 is used to indicate a packet containing information on a new move made by the player (other packet types could be assigned in the future for additional functions). The row and column numbers point to the position of the new disc inserted by the originating player.

The microcontrollers may repeat the transmission of the packets periodically to tolerate errors and lost packets. The receiver should ignore all duplicate packets.

The receiver must ignore any packet with the row and/or column number set to an out-of-range value. The transmitter may periodically send such packets to keep the wireless link alive.

CapSense Input

You must use the CapSense slider and one of the buttons to specify your moves. It should be possible to position the disc on the home row in any column by moving a finger along the slider. After positioning the disc in the desired column, pressing the button should cause the disc to be dropped into the column.

SD Card

The SD card is used to log the player moves, so that the game can be reconstructed later. The software should log the moves made by each player into a file in the order in which they were made. Each move is to be recorded in the following format:

<player-number> <row-number> <column-number>

The player number is either 1 or 2. The row and column numbers correspond to the position of the newly inserted disc. The fields must be separated by spaces or tabs, and each record must end with a newline.

Documentation on the MicroSD breakout board can be found at:

https://www.sparkfun.com/products/544?

This page has links to a number of useful design resources, including the SD card specification. SD cards use a multi-channel serial interface called SPI, which is supported in PSoC Creator.

Assembly and Testing

The project is meant to be assembled on the PCB in the parts kit. This board can then be connected to the main microcontroller board through the 40-pin connector. Avoid using the prototyping area on the microcontroller board as much possible, because of the risk of damage to the board.

The following sequence of steps is recommended for a smooth bring-up of the project.

1. Create detailed schematics of the design.
2. Determine a good placement for all the components on the PCB.
3. Solder components on the PCB. Do no solder the 8x8 display directly. Instead, solder two 16-pin female headers to serve as a socket for the display. The XBee module has a pin spacing of 2 mm, which is incompatible with the 0.1" spacing on the PCB. The adapter is needed to change the pin spacing.
4. Do not start by populating all the components. Instead, start with the display first, and populate only the components needed to make the display functional. Write a test program to display various patterns and make sure that you are able to activate all the LEDs in the matrix. Write a program to display any given 8x8 pattern (represented using an appropriate data structure) on the LED display, by scanning the rows/columns in sequence.
5. Next, populate and test the XBee module. You can first loop back the serial transmit data sent to the XBee back to the receive side of your UART and write a test program to make sure that data is received correctly. You can then work with a partner to test the wireless link by transmitting and receiving packets.
6. Test the operation of the CapSense button and slider using a test program. The program should track the slider and display its position (a number between 1 and 8) on the LCD display. This can then be used to position the disc to be dropped on top of the right column.
7. Once all the components have been tested, you can write the complete program and debug it. Try to debug as much of your program as possible without using the wireless link, to minimize your dependence on your partner's system.

What to Submit?

- Schematics of your design.
- Description of your hardware.

- Description of your software. How do the key modules of the software interact with each other? How did you make sure that an event such as the arrival of a packet or a display refresh event is serviced within an acceptable time limit?
- Discussion of key design decisions made. What factors drove your decisions on the hardware and software design for interfacing the 8x8 LED display? How did you determine the refresh rate of the display?
- A discussion of how you assembled tested your system. Include any test programs used.

You should maintain detailed notes during the design and testing of your project to help you write the documentation.

Deadlines

The project demos are due in the lab in the week of June 3rd (schedule TBA). The reports will be due on June 11th.