Suggested Project: Retro video gaming

Let us journey back to the early 1980's when 280x192 was considered "high resolution" and most microprocessors ran at only 1 MHz or less. With less to look at, video games required a bit more imagination and much better gameplay.

I want to give you some flexibility in this project, but I also want to try to keep everyone's project at a similar level of difficulty. From the following tables, select features that total up to at least 5 points (7 points for 3 person teams).

Game type (pick one)

Points	Implementation feature
1	Non-animated turn-based game (everything stops when waiting on the player)
2	Animated real-time game (objects continuously in motion)

Display (pick one)

Points	Implementation feature
0	Use the LPCXpresso console window for output
0.5	Use a UART to connect with the PC for output
0.5	Use LEDs scanned in a multiplexed fashion (more than one row and column)
0.5	Use a character (HD44780 style, non-graphical) LCD for output
1	Use a character (HD44780 style, non-graphical) LCD for output, with custom characters to implement pseudo graphics or sprites
2	Use a graphical LCD for output
2	Use the oscilloscope in X-Y mode as a vector graphics display.
3	Use a TV or computer monitor for output (connected directly to your circuit; no PC)

Sound (pick one or more)

Points	Implementation feature
0.5	Use a linear-feedback shift-register to generate a noise sound effect
0.5	Use PWM or timers to generate square wave beep sound effect
1	Use D-to-A to generate a sine wave based sound effect
1	Play a song with at least 8 notes
1	Play the song in the background (i.e. using interrupts)

Input hardware (pick one or more)

Points	Implementation feature
0	Use the LPCXpresso console window for input
0.5	Use a UART to connect with the PC for input
0.5	Use a keypad for input (must use at least 5 keys)
0.5	Use a joystick with an analog interface
0.5	Use a game controller with a serial interface (NES, SuperNES, Playstation 1 & 2)
0.5	Use a game controller with a multiplexed interface (Sega Genesis)
0.5	Build your own game controller (must use at least 5 switches) that uses a serial interface
0.5	Use a quadrature encoder input
1	Use a PS/2 compatible keyboard or mouse for input
1	Use an IR remove control for input (must use more than one button)

Other implementation features (optional)

Points	Implementation feature
1	Use DMA subsystem to send audio or video samples to DA or I2S subsystem for output
0.5	Use non-volatile memory (for example, EEPROM to retain high score table)
0 – 1	Awesome feature that demonstrates non-trivia digital design (point value determined by instructor)

Notes:

- The game must be non-trivial, in the instructor's opinion.
- If a range of points is given, the minimum number of points is awarded for minimally meeting the requirements. More points will be awarded for better user interfaces and configurability.
- You can use the equipment at your lab station, the parts in your lab kit, and/or other components that you acquire (all subject to the next note).
- Remember, this project should show your mastery of digital design; using a pre-built module that substantially implements a high level function will not earn you any credit.
- The lab does have some graphical LCDs, speakers, audio amplifier chips, and piezo transducers that you are welcome to borrow.
- Please be considerate of everyone in the lab by limiting the duration and volume levels of any audio during testing. In many cases you can test the generated waveforms using the oscilloscope instead of a speaker.

Example projects:

- Hangman: Turn based game with stick figure drawn using ASCII characters on PC via UART. User inputs letters guessed on PC and transferred via UART. Play a funeral dirge when the player loses. Total: 1+0.5+0.5+1 = 3 (this is less than 5, so you would get proportionally less credit)
- Freeway crossing: Player attempts to cross a freeway with multiple lanes of continuously moving traffic. Using the non-graphical LCD with custom car/truck character for output and original Nintendo controller for input. Generate noise effect for crashes and play a song for each successful crossing. Total: 2+1+0.5+0.5+1 = 5
- Space rocks: Player attempts to pulverize large flying space rocks into smaller ones without crashing their spaceship. Using oscilloscope vector display and switches connected directly to microcontroller. Ship's laser generates "pew-pew" sound effect and destroyed rocks make a explosion sound. Total 2+2+0+0.5+0.5 = 5
- Optimally packed falling shapes: Player attempts to optimally fill a space with continuously falling shapes. Using a graphical LCD for output and a homemade controller with serial interface for input. Continuously plays music for maximum player enjoyment/annoyance. Total 2+2+0.5+0.5+1+1 = 7

Project Proposal

A good project proposal should both help you convey your ideas for what you want to do as well as help organize your thoughts so that you start with a solid plan to achieve the project goals. You only need to submit this if you want to propose an alternative to the suggested project, but it’s a good idea to at least make an informal version of this for yourselves in any case.

- A general description of the project and its purpose.
    - If you want to do something other than the suggested project, include some detail of how you expect to implement your features so that I can give you feedback on how many points I think they would be worth.
- A table of the functions/features, prioritized into three categories:
    - Things that must work,
    - Things that ideally should work, but could be omitted for some loss of functionality,
    - Things that improve the value of the project, but could be omitted without loss of core functionality.
- A table of input/out devices that you expect to use (you don't need specific part numbers at this point) and how you expect to interface them (GPIO, SPI, I2C, analog input/output, etc.)
- A statement of what hardware/software you expect to build/write yourself and what hardware/software you expect to use as a pre-built module.

Grading

- 10% Preliminary hardware/software design report (due April 16th)
    - A general description of the project and its purpose
    - A complete schematic, with enough detail so that someone else could build the exact circuit you want without any ambiguity. Since this is a preliminary report, the schematic does not necessarily need to work, but you should have a reasonable expectation that it would.
    - For discrete components like resistors and capacitors, show how their values were chosen.
    - A high level description of how the software will implement the functions
- 40% Final report (due May 3rd)
    - A general description of the project and its purpose (updated, if needed)
    - Directions for use (think of it as the user manual for your project)
    - The design analysis from the preliminary report (updated, if needed)
    - The final complete schematic
    - For discrete components like resistors and capacitors, show how their values were chosen.
    - The final complete software
    - Any other supporting documentation to show that the project meets the objectives/specifications (for example, oscilloscope snapshots)
- 10% Project poster (due by end of semester)
    - Names of the team members and the project title
    - Summary of the project, its feature/specifications
    - A simple overview of how the project works (a basic block diagram is fine)
    - It's nice to have the posters visible for all to see in the next semester, but this is difficult when there are lots of large ones. Try to keep the poster size around 2 by 3 feet or less.
- 40% Demonstration (April 30th – May 3rd during regular lab/lecture time, or by appointment)
    - Each time member must participate in the project demonstration and be prepared to discuss some digital aspect of the project that they contributed to for full credit.
    - The 5 (or 7 for large teams) points of the application/function/features contributes to this portion of the grade.
    - An extra point beyond the 5 (or 7) will be allowed for bonus purposes.