Homework 1 – Mastermind

Due Date – Friday 9/22 (part one) & Friday 9/29 (finished product)

Project Statement

This first assignment is meant to grant you experience working with the Atmel Assembler as well as practice with I/O, serial transmission, and state machines. You will be designing a piece of code that emulates a variant of the game "mastermind." The game definition is as follows:

The user will take guesses at the code using the joystick with combinations of up-down-left-right. A wrong input will sound a buzzer (piezoelectric element) and reset the user's game. Completing the code will light an LED. At this point the game can be reset by pressing the push-button. All inputs will be logged over a serial interface (UART) to a computer reporting the state of the game.

The UART code will be provided for you. The rest is up to you to decide how to implement this project.

Deliverables

Part 1

For part 1, you must convince us that you have made significant headway on the project. You must submit a short report with the following

- A diagram of the state machine you intend to use for your system
- Initial Code for the solution of the system
    - Getting joystick+LED+BUZEER working to go through the game sequence

Part 2

For part 2, the entire system must be implemented. You must submit the following:

- All code you use to get the system to function
- A report with the following
    - A clear statement on the completeness of your system. Report what testing procedures you performed. This includes BAUD rates for serial, what pins you used for the LED and pushbutton, and whether or not you could complete the game.
    - Document design decisions. (i.e. What your state machine looks like, your code flow, etc…)
    - Provide a usage manual (make it clear what your system DOES not what it was intended to do but you never got working).

Grading

The grade distribution for this project will be as follows

- 15% Part 1: State machine diagram, initial code for getting joystick+LED+BUZEER working to go through the game sequence
- 85% Part 2:
  - 10% Coding style
  - 15% Reports
  - 60% Functionality

The functionality will be broken down into components. Each working component will give you credit. A fully working system will give you full credit for this portion.

Components in the functionality section:

- Joystick (up, down, left, right)
- Buzzer
- LED
- Pushbutton
- Game logic

Coding style is based on the following guidelines

- All subroutines need to be commented with:
  - Input parameters
  - Any affected registers
  - Description of what the subroutine is intended to do
- All main loop code should be documented with its intended functionality
  - i.e. A 3 line software delay loop should have documentation on what register is being used, and that those three lines represent a delay function
  - Any special purpose code written should definitely be documented
    - This includes any branching instructions, register copying, any I/O instructions
  - At least every other line of code should be documented
- Registers should be renamed with directives (e.g. Count, temp, etc…)

This section is important because assembler is very hard to follow. If you do not comment your code, it will be very difficult to assign partial credit for functionality purposes.

Game Description

The game will work as follows:

- The machine will turn on, this is the initial state
- The user will begin to make guesses at a secret code using the joystick input (up, down, left, right)
  - The secret code should be 4 in length and stored in a single register
  - The secret code should be defined using a .EQU directive where

        - UP=00, DOWN=01, LEFT=10, RIGHT=11
            - e.g. - .EQU secret = 0b00011011 = UP,DOWN,LEFT,RIGHT secret code
        - The secret code should be utilized such that no other changes have to be made to your program to switch between codes
- An incorrect guess at any state will sound the buzzer and return the user to the initial state
    - E.g. the user gets "L-R" correct before pressing "D" instead of "U", they will have to enter "L-R" again before they can proceed to guess "U"
    - Sound the buzzer for ~1/5 of a second
- A correct guess at any state will progress the user to the next state
- Getting all the way through the code will light the LED to alert the user that they have won
- The pushbutton is used to reset the game to the initial state, and can be pressed at any state
- The UART serial interface will report the following after every button push
    - The button that was pushed
    - The button that was expected to be pushed
    - The state of the game before the button was pushed
    - The state of the game after the button was pushed
    - Whether the button was correct or incorrect
- How the serial interface reports the values is up to your design. If you wish to use single ASCII characters instead of full words, this is fine. The values should be comma separated.
    - E.g. (U,D,2,0,I) could mean up was pressed, down was expected, you were in state 2, you are in state 0, and the guess was incorrect

A few notes

It is advisable that you take a state machine approach to writing the code. We will be requiring you to submit the design for week 1 report but it is recommended to stick to a state machine approach so code is more compact, concise, and reusable.

Start early. This is a hardware project, so things WILL go wrong. Get your hardware elements working with subroutines, and call them in your software state machine. If you need to test your hardware for problems, you can call your subroutines at the beginning of your code and see if it is working properly for debugging purposes.

The microcontroller clock should be set to 8MHz. The provided UART code should communicate at 4800 baud. You can use RealTerm or PuTTY to test the serial output. Set these programs to 4800 baud and 2 stop bits.

You will need the include file "m169pdef.inc"

Lastly, go to your TAs and instructor early with questions regarding the homework. Office hours are a great resource for code specific questions. Piazza is a very good resource for general questions. **DO NOT POST CODE ANYWHERE ONLINE. DO NOT SHARE CODE WITH CLASSMATES**. If you use any code you get from the web, you must **CITE** it according to citation policy. We will be using cheat check software, if

you attempt to plagiarize, you will be caught. It is better to turn in nothing than to turn in plagiarized work.