Assignment 6 – Virtual Tug-o-War Game using Direct Register Access

Connect the pushbutton switches and LEDs to the mbed module as shown in the schematic to the right.

The player for the red team will use the switch connected to p14 and the player for the green team will use the switch connected to p27. Each player will attempt to continuously press and release their switch as fast as possible. The virtual rope begins the game centered on the playfield. Whenever a player's switch changes state (from off to on or on to off) the virtual rope should move one unit closer to that player's side of the playfield. If the virtual rope has been pulled close enough to the end of the playfield, the player on that end of the playfield wins.
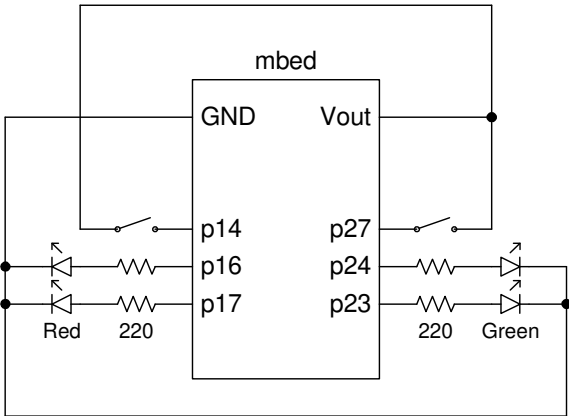
The virtual rope position can be in one of 7 positions as indicated by the external and built-in LEDs (the LEDs that are on are indicated with the starbursts):

Playfield LEDs	Description
(red on)(red on) ▬ ▬ ▬ ▬ (green)(green)	Red team has won
(red off)(red on)(blue on) ▬ ▬ ▬ (green)(green)	
(red off)(red off)(blue on)(blue on) ▬ (green)(green)	
(red off)(red off) ▬ (blue on)(blue on) ▬ (green)(green)	Rope position at the start of the game
(red off)(red off) ▬ ▬ (blue on)(blue on) (green)(green)	
(red off)(red off) ▬ ▬ ▬ (blue on)(green on)(green)	
(red off)(red off) ▬ ▬ ▬ ▬ (green on)(green on)	Green team has won

Once the virtual rope has reached one of the two winning positions, the game is over and the program should stop (go into an infinite loop). If the players want to start a new game, they should press the mbed's reset button.

Write the program to implement this game using direct access to the microcontroller's I/O registers instead of using the mbed's library (no #include "mbed.h", DigitalIn, DigitalOut, BusOut, InterruptIn, etc.). For simplicity, ignore the potential problem of switch bounce.

Submit your "main.cpp" to the appropriate dropbox on Canvas by the end of May 3rd.