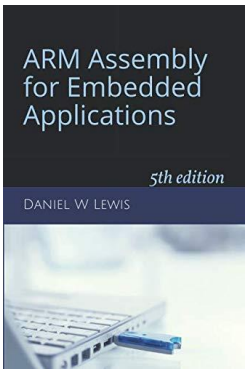


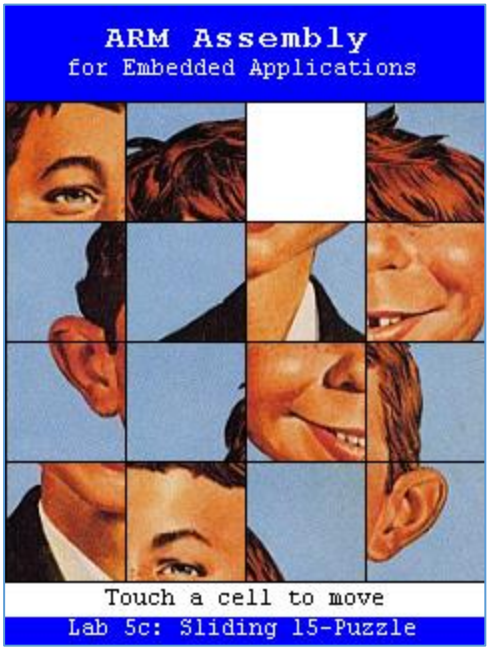
Programming Lab #5c

Sliding 15-Puzzle

Topics: Compare and branch instructions, nested loops, pointers.

Prerequisite Reading: Chapters 1-6
Revised: March 22, 2020

Background (Source: Wikipedia)**:** A "sliding" puzzle is a combination puzzle that challenges a player to move tiles on a two-dimensional array of cells so as to establish a certain end-configuration. Finding moves and the paths opened up by each move within the two-dimensional confines of the array are important parts of solving sliding block puzzles.



Assignment: In this lab, the puzzle is displayed within a $4x4$ array of 16 cells. An image has been divided into 16 tiles of $60x60$ pixels each with 15 of the tiles placed into the array. Each pixel is stored as a 32-bit value containing 8 bits each of its red, blue, green and intensity components. Create an assembly language file containing translations of the following two C functions. `CopyCell` copies all the pixels of one cell to another and `FillCell` fills a cell with white pixels. Parameters `dst` and `src` contain the address of the pixel in the top left corner of a cell.

```
void CopyCell(uint32_t *dst, uint32_t *src)
    {
    uint32_t row, col ;

    for (row = 0; row < 60; row++)
        {
        for (col = 0; col < 60; col++)
            {
            dst[col] = src[col] ;
            }
        dst += 240 ; // Move down to
        src += 240 ; // the next row
        }
    }
```

```
void FillCell(uint32_t *dst, uint32_t pixel)
    {
    uint32_t row, col ;

    for (row = 0; row < 60; row++)
        {
        for (col = 0; col < 60; col++)
            {
            dst[col] = pixel ;
            }
        dst += 240 ; // Move to next row
        }
    }
```

Test your code using the main program downloaded from here. Tiles may be moved only if they are moved to an adjacent empty cell. When the program starts, it performs a sequence of random moves and then displays the scrambled image that results. The objective is to use the fewest number of moves to unscramble the tiles so that the image is displayed correctly. Touching a tile that is adjacent to an empty cell will move the tile into the empty cell. The program records a history of up to 1000 such moves; touching the empty cell undoes the most recently recorded move. Pressing the blue pushbutton will undo all moves to display the original (unscrambled) image.