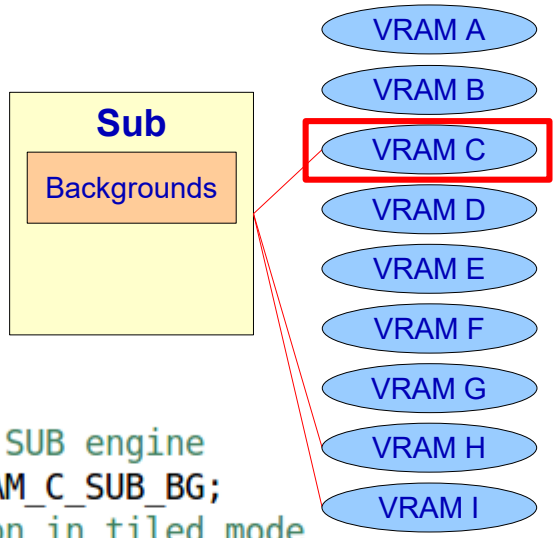
2022/2023 Mid-Term Exam
Code review

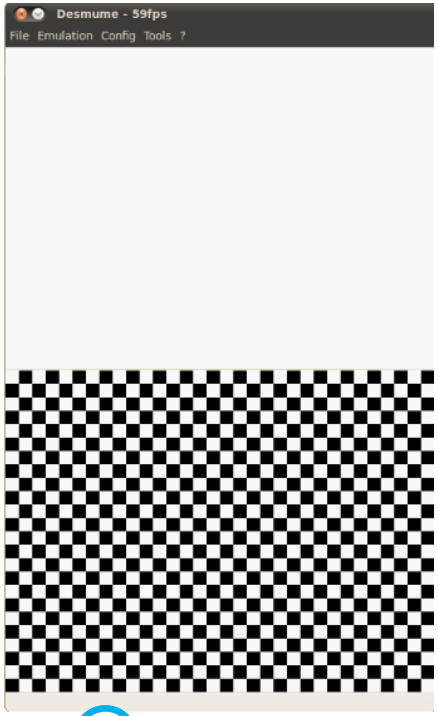
Systèmes Embarqués Microprogrammés

Prof. David Atienza Alonso, SEL-STI

Exercise 1 – Part A
Subscreen as a tiled checkered background

- Configure the subscreen in tiled mode and use the provided *WhiteTile* and *BlackTile* to fill the screen as shown in figure.





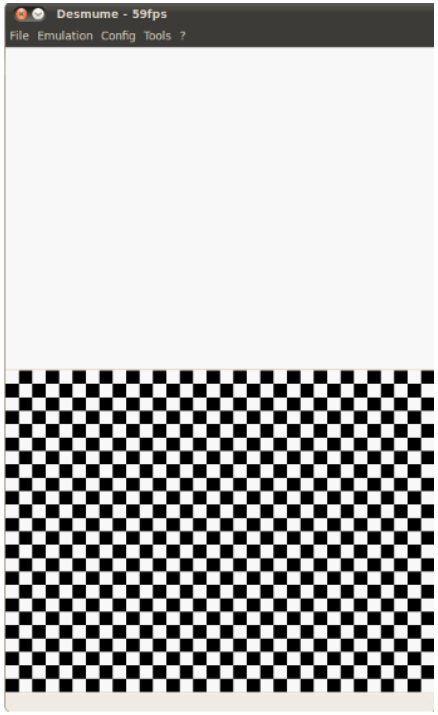

```
/*************
 * EXERCISE 1
 *************/
// 1) VRAM configuration for SUB engine
VRAM_C_CR = VRAM_ENABLE | VRAM_C_SUB_BG;
// 2) SUB engine configuration in tiled mode
REG_DISPCNT_SUB = MODE_0_2D | DISPLAY_BG0_ACTIVE;
// 3) Configure the background
BGCTRL_SUB[0] = BG_32x32 | BG_COLOR_256 | BG_MAP_BASE(1) | BG_TILE_BASE(0);
// 4) Copy the 5 tiles to the tile base
dmaCopy(RedTile,   &BG_TILE_RAM_SUB(0)[0],  64);
dmaCopy(GreenTile, &BG_TILE_RAM_SUB(0)[32], 64);
dmaCopy(BlueTile,  &BG_TILE_RAM_SUB(0)[64], 64);
dmaCopy(WhiteTile, &BG_TILE_RAM_SUB(0)[96], 64);
dmaCopy(BlackTile, &BG_TILE_RAM_SUB(0)[128],64);
```

Avoid overlap in VRAM

Mode	BG0	BG1	BG2	BG3
0	Tiled	Tiled	Tiled	Tiled

Exercise 1 – Part B
Subscreen as a tiled checkered background

- Configure the subscreen in tiled mode and use the provided *WhiteTile* and *BlackTile* to fill the screen as shown in figure.
  - Initialize the palettes with the 5 different colors
  - Generate the map to display the checkered pattern




```
// 5) Initialize the palette (5 components)
BG_PALETTE_SUB[0] = RED;
BG_PALETTE_SUB[1] = GREEN;
BG_PALETTE_SUB[2] = BLUE;
BG_PALETTE_SUB[3] = WHITE;
BG_PALETTE_SUB[4] = BLACK;

// 6) Generate the map
int row,col;
for (row=0; row<24; row++)
    for(col=0; col<32; col++)
        BG_MAP_RAM_SUB(1)[row*32+col] = (row*32+col+row%2)%2 + 3;
```

Iterate over all the tiles you have to draw

Either 3 (WHITE) or 4 (BLACK)

Exercise 2
Configure main background in rotoscale mode

- Show the provided EPFL logo image in the main screen. To do so, configure the background correctly and transfer the image information to the corresponding locations in memory.
  1. Background in rotoscale mode with 8-bit pixels palettes
  2. Transfer bitmap and palettes to the graphical memory

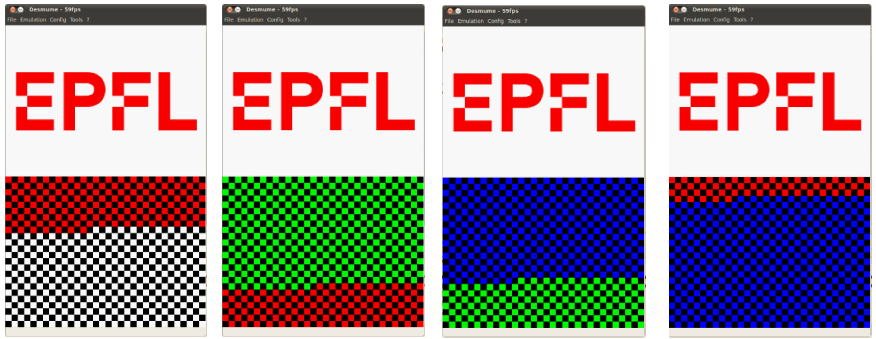
```
/*************
 * EXERCISE 2
 *************/
// 1) VRAM Configuration for MAIN engine
VRAM_A_CR = VRAM_ENABLE | VRAM_A_MAIN_BG;
// 2) Main engine configuration in rotoscale mode
REG_DISPCNT = MODE_5_2D | DISPLAY_BG2_ACTIVE;
// 3) Configure the background (Ignore warning 'large implictly truncated to unsigned type')
BGCTRL[2] = BG_MAP_BASE(0) | BgSize_B8_256x256;
// 4) Copy bitmap and palette generated by grit
swiCopy(logoPal, BG_PALETTE, logoPalLen/2);
swiCopy(logoBitmap, BG_GFX, logoBitmapLen/2);
// Uncomment this for including the Affine Matrix Transformation
bgTransform[2]->xdx = 1*256;
bgTransform[2]->ydx = 0*256;
bgTransform[2]->xdy = 0*256;
bgTransform[2]->ydy = 1*256;
bgTransform[2]->dx = 0*256;
bgTransform[2]->dy = 0*256;
```

Main engine background configuration

swiCopy transfers 16-bit words. Input data length (expressed in bytes) must be halved

Exercise 3 - A
Use timer interrupts to color the subscreen

- Use a timer to color the white tiles. Iteratively, as shown in figure, fill them with red ➔ green ➔ blue ➔ red ➔ …
  1. Use interrupts and set the color update frequency to 20Hz



```
/*************
 * EXERCISE 3
 *************/
////////////////////////////////////////////////////////////////////////////////
//IMPORTANT NOTE!!!: Do not initialize the interrupts with the call to irqInit() since
// later the touchscreen will be used
////////////////////////////////////////////////////////////////////////////////
// 1) Configure the timer to raise an interrupt 10 times per second
int frequency = 20;
TIMER_DATA(0) = TIMER_FREQ_1024(frequency);
TIMER0_CR = TIMER_ENABLE | TIMER_DIV_1024 | TIMER_IRQ_REQ;
// 2) Associate the implemented ISR to the Timer Interrupt Line
irqSet(IRQ_TIMER0, &timer_ISR);
// 3) Enable the timer interrupt
irqEnable(IRQ_TIMER0);
```

The interrupt service routine “timer_ISR” is called 20 times per second and updates the colors

Exercise 3 - B
Use timer interrupts to color the subscreen

- Use a timer to color the white tiles. Iteratively, as shown in figure, fill them with red ➔ green ➔ blue ➔ red ➔ …
  1. Use interrupts and set the color update frequency to 20Hz


```
//Timer ISR (for EXERCISE 3)
// You may want to use these variables in your ISR
int current_color = 0, timer_row = 0, timer_column = 0;
void timer_ISR()
{
    BG_MAP_RAM_SUB(1)[timer_row*32+timer_column] = current_color;

    timer_column = timer_column + 2;
    if (timer_column >= 32)
    {
        timer_row++;
        timer_column = timer_row % 2;
    }
    if(timer_row >=24)
    {
        timer_row = 0;
        timer_column = 0;
        current_color = (current_color + 1) % 3;
    }
}
```

Color current tile with current color

Move to next column

Move to next row.
Rows with odd idx must be colored starting from column 1

Go back to first tile (top left) and update the color

Exercise 4
Update the coloring frequency

- The coloring frequency is set to 20Hz in Exercise 3. KEY_UP and KEY_DOWN shall be now used to increase and decrease this frequency
  1. Pressing KEY_UP doubles the frequency
  2. Pressing KEY_DOWN halves the frequency

```
/*************
 * EXERCISE 4
 *************/
while(1){

    // 1) Scan the keypad
    scanKeys();
    // 2) Obtain the pressed keys
    int keys = keysDown();
    // 3) Check if the pressed key is the correct one and increase/decrease the frequency accordingly
    if(keys & KEY_UP)
    {
        frequency = frequency*2;
        TIMER_DATA(0) = TIMER_FREQ_1024(frequency);
    }
    if(keys & KEY_DOWN)
    {
        frequency = frequency/2;
        if(frequency == 0)
            frequency = 1;
        TIMER_DATA(0) = TIMER_FREQ_1024(frequency);
    }
```

Change frequency by updating the TIMER_DATA register

Note! The frequency is set to 20Hz. Pressing KEY_DOWN only 5 times makes frequency reach 0Hz (integer divisions). Manage this case, otherwise you cannot increase frequency anymore.

Exercise 5
Update the color with the touchscreen

- The color filling the non-black tiles can be immediately changed when touching the subscreen.
  1. If the subscreen is touched, immediately change color to update next tiles, following the same transitions used for Exercise 3.

```
while(1){
    ...
    ...
    // 4) If the touchscreen is touched, set the new color
    touchPosition touch:
    touchRead(&touch);
    if(touch.px || touch.py)
        current_color = (current_color + 1 ) % 3;

    swiWaitForVBlank();
}
```

Touch happens when px and py are not both 0.
In this case, just update the current color