Assignment 2 PWM Operation and Display Driver in Zephyr RTOS (100 points)

In this assignment, you are requested to implement an application program that adds 3 shell commands in Zephyr RTOS (version 2.6.0) running on MIMXRT1050-EVKB board. In addition, you need to develop a display device driver for MAX7219-controlled led matrix and add it to Zephyr source tree. The 3 shell commands are:

1. *p2 rgb x y z* (x, y, and z are decimal integer between 0 to 100)
2. *p2 ledm r x0 x1 x2 x3 …* (x0, x1, … is a list of bytes in hexadecimal representation, i.e., with prefix 0x)
3. *p2 ledb n* (n equals 0 or 1)

where "p2" is the root command, "rgb", "ledm" and "ledb" are subcommands, and "x", "y", "z", "n", "r" and "x0-7" are command arguments.

1. p2 rgb x y z

In the example program "led_gpio_sample", red, green, and blue leds are turned on and off sequentially every second. To add intensity control, we can apply PWM signals of variant duty cycles to the leds. The SoC of MIMXRT1050-EVKB consists of 4 FlexPWM modules (please see chapter 54 of i.MX RT1050 Processor Reference Manual) and each module contains 4 submodules. The 2 PWM channels in each submodule can be programmed to put out PWM signals and its driver can be enabled in Zephyr.

We will assume the R, G, and B pins of the target led are connected to D2 (pin 3 of J22), D5 (pin 6 of J22), and D12 (pin 5 of J24), respectively. This shell command sets the duty cycles of three PWM signals to *x*, *y*, and *z,* and then apply the PWM signals to the RGB led. For instance, with "p2 rgb 20 40 60" command, three PWM signals of duty cycles 20%, 40%, and 60% are generated and applied to the target RGB led.

2. p2 ledm r x0 x1 x2 x3 …

The command reads in a list of row patterns and displays the row patters starting from *r*-th row in a MAX7219 controlled 8x8 led matrix. For instance, "*p2 ledm 4 0x00 0xff*" will turn off all leds of the 4th row and turn on all leds of the 5th row (note that the rows are numbered from 0 to 7). All other rows are not changed by this command. To drive the display operations of the led matrix, you will use the spi interface *lpspi1* of MIMXRT1050-EVKB. The pin connection to the Arduino shield of MIMXRT1050-EVKB is shown in the following table.

MAX7219	Arduino connector
DIN	D11(SPI_MOSI)
Load	D10(SPI_CS)
Clk	D13(SPI_CLK)

3. *p2 ledb n*

   When n is 1, this shell command starts the blinking mode, i.e., the display patter on 8X8 led matrix is ON and OFF every second. If n=0, the blinking mode is off.

Your application files should reside in a directory named "project_2" which contains CMakeLists.txt, prj.conf, readme.txt, a source file directory "src", and a board device tree overlay directory "boards".

For the display driver, it should at least implement three display api functions: display_blanking_on, display_blanking_off, and display_write in source file "zephyr/drivers/display/display_max7219.c". Additional modifications are needed in:

- zephyr/drivers/display/CMakeLists.txt,
- zephyr/drivers/display/Kconfig,
- zephyr/drivers/display/Kconfig.max7219, and
- zephyr/dts/bindings/display/maxim,max7219.yaml

The dts binding file maxim,max7219.yaml will be provided and should be added to the directory /(path to zephyr tree)/dts/bindings/display.

A good starting point for the driver development is to study the driver for st7735r lcd panel in Zephyr source code. You can also try to write an application to control the display of led matrix via spi interface before diving into the driver.

Please note the following items for this assignment:

- Components for the assignment will be distributed soon.
- 4 0-ohm resistors, R278-R281, are not populated on MIMXRT1050-EVK. You will need to short-circuit these DNP connections. You can use the soldering station in BYENG 217 to solder jumper wires.
- You may assume that all input shell arguments are in correct format. For instance, in the command "p2 rgb x y x", "x", "y", and "z" are integers between 0 and 100.
- The frequency of the PWM signals is fixed at 50Hz.
- IO configuration data must be extracted from board device tree and overlay. The only exception is that the IO pin names for IO multiplexing operations can be hard-coded in your main program.

Due Date

The due date is 11:59pm, March 6.

What to Turn in for Grading

- Your submission is a zip archive, named RTES-LastName-FirstInitial_02.zip, that includes
  - An application folder, named *project_2*, to include all your implementation of the assignment. The folder should contain *CMakeLists.txt, prj.conf, readme.txt*, and a source file directory *src,* a board device tree overlay directory "*boards*", and the source code of the display driver *display_max7219.c.*
  - A patch file to contain all changes in the directory /(path to zephyr tree)/driver/display. Please use the following diff command to generate the patch:

    diff –rauN /(original zephyr tree)/drivers/display /(modified zephyr tree)/drivers/display > patch_display

  - The readme text file describes all the commands you use to build and run your program.
- Note that any object code or temporary files should not be included in the submission. Submit the zip archive to the course Canvas by the due date and time.

- Please make sure that you comment the source files properly and the readme file includes a description about how to make and use your software. Don't forget to add your name and ASU id in the readme file.
- There will be 20 points penalty per day if the submission is late. Note that submissions are time stamped by Canvas. If you have multiple submissions, only the newest one will be graded. If needed, you can send an email to the instructor to drop a submission.
- The assignment must be done individually. No collaboration is allowed, except the open discussion in the forum on Canvas. The instructor reserves the right to ask any student to explain the work and adjust the grade accordingly.
- Failure to follow these instructions may cause deduction of points.
- Here are few general rule for deductions:
    - Cannot compile or compilation error -- 0 point for the assignment.
    - Must have "–Wall" flag for compilation -- 5-point deduction for each warning.
    - 10-point deduction if no compilation or execution instruction in README file.
    - Source programs are not commented properly -- 10-20-point deduction.
- ASU Academic Integrity Policy (http://provost.asu.edu/academicintegrity), and FSE Honor Code (http://engineering.asu.edu/integrity) are strictly enforced and followed