SE 3KO4: Lab 1

Simulink Model Configuration

See the "Simulink guide" PDF available on Avenue for the details of this step. You must configure your Simulink model properly after creating it in order to use the model with the K64F microcontroller used in this course.

Blinking LED Implementation

In this lab you will use Simulink to program your microcontroller to make an LED blink. First create a new blank model and configure it as described above.

Create a New Chart with Inputs and Outputs

Click in the Simulink model window and search for "chart" to add a new chart to your Simulink model. This is where you will add the logic to implement your program. You will need to configure the inputs and outputs for the chart, this can be done from the model explorer. Press **CTRL+H** to open the model explorer. Make sure that your chart is selected on the left-hand side of the model explorer, since you want these variables to exist in the scope of the chart. From here, different variables can be added.

For this chart, the inputs will be the amount of time the light should stay off between blinks, the amount of time the light should blink for. The output will be the LED state (on or off). The chart below describes the correct setup of these variables in the model explorer.

Variable	Type	I/O
DURATION_ON	Numerical type (example: int16)	Input
DURATION_OFF	Numerical type (example: int16)	Input
LED_STATE	Boolean	Output

The output variable needs to be mapped to the appropriate pin on the microcontroller so that it can actually be used to control the LED light. Do this by clicking in the Simulink model window and searching for the "Digital Write" block for the K64F. Select an LED pin to write this value to. Connect your output variable from your chart to the digital write block by clicking and dragging the output over to the digital write block.

Create States in the Chart

Double-click the chart you've created in the Simulink model window to open it. In this program, since the light can either be on or off, only 2 states are required to implement the desired functionality. Add a state by dragging it into the workspace from the toolbar on the left side, and give your state a descriptive name (for example, LED_ON or LED_OFF). You will need one state for when the LED is on and one state for when the LED is off, so create both now.

One of the states you set up will need to be defined as the starting state. In this program it is not very important which state is the starting state, but in other applications it may be important to make sure the program functions correctly. Do this by dragging a "default transition" from the toolbar on the left to the state that you want to be the starting state.

Create State Transitions

In order to get the light to actually blink, you need to set up transitions between the states in your chart. If no transitions are set up, the program will just remain in the starting state indefinitely. Transitions occur between states when their conditions are met. In this program, the transitions will be required to turn the LED on, and to turn the LED off. The table below summarizes the transitions you will need to set up.

Starting State	Ending State	Condition
LED_ON	LED_OFF	after(DURATION_ON, msec)
LED_OFF	LED_ON	after(DURATION_OFF, msec)

Click and drag from the border of the starting state to create a transition to the ending state (drag to the ending state). Click the transition arrow to modify the condition of the transition. When programming the condition you have access to the chart variables you defined earlier.

Create State Entry Procedures

You have set up your chart with input and output variables, states, and state transitions, but you still need to set up the logic for those variables to determine what happens when state transitions occur, or when the program is in a particular state. Procedures can be defined for “entry”, “during”, and “after” for each state. In this program, you only need to use the “entry” state. Click on the body of the state in your chart to define these procedures. Here you can modify variables within the scope of the chart.

The only thing you need to do in the entry procedure for each state in this program is to flip the LED_STATE variable to **true** or **false** to control the LED pin on the microcontroller. Your final state should look something like the box below when you’re finished. You will need to do this for both states so the LED can turn on and off when the program enters the appropriate state.

```
LED_ON
entry:
LED_STATE = true;
```

Define Constants to Connect with Chart Inputs

Your chart has inputs set up for the durations of the on and off portions of the blink, but no value is yet being assigned to those inputs. To set values for each of the inputs, go to the Simulink model window and click in the workspace, then search for “Constant” and add a constant block. Give your constant block a name, and set the value by clicking within the block. Click and drag the constant block output to the appropriate chart input to feed the constant value to the input. Do this for both duration inputs.

You also need to set up the value type for the constants. To do this, double-click the constant block and look in the signal attributes tab to select the type. Similar to a typed programming language, the type of the constant needs to match the type of the input you’re connecting it with. If you selected **int16** as the type for your chart inputs, select **int16** here as well.

Test the Output Using a Scope

Programming the microcontroller to run your program on the hardware can take some time. If you want to test your program using a Simulink to simulate the output instead, you can do this using a scope block. Click on the workspace in the Simulink model window and search for a “Scope” block, and add this block. Connect the chart output to the scope block.

Set the length of time for the simulation to run for (for example, 10 seconds), and run the simulation. Double-click the scope to see the changes to the output variable over time. If the output isn’t what you’d expect, go back and make sure you haven’t made a mistake.

Run the Program on Hardware

Use the “deploy to hardware” button to compile the program and flash the board. This can take some time. Once the flashing is complete you should see the LED blinking on your microcontroller.