Task 1:

This assembly code sets up a loop that prints “Hello World” with a countdown that starts at 100 and ends at zero, then resetting to continue printing. This is done with a loop by loading the “Hello World” string into R0 and decrementing R7 by 1, starting at 101 (SUB R7, #1 will subtract 1 from R7 before printing the first value). The program checks the value of R7 against 0 with CMP R7, #0. If R7 isn’t 0, it branches to the “continue” label which prints “Hello World” (using “printf”) with the current value of R7. If R7 is 0, it resets the R7 value to 101 with MOV R7, #101, which can be seen at the end of the loop label.

```
.thumb_func
.global main
main:
        MOV R7, #101
        BL stdio_init_all
loop:
        LDR R0, =helloworld
        SUB R7, #1
        MOV R1, R7
        CMP R7, #0
        BNE continue
        MOV R7, #101
continue:
        BL printf
        B loop
.data
        .align 4
helloworld: .asciz "Hello World %d\n"
```

Task 2:

This program works by initializing each LED output and sequentially calling “BL link_gpio_put” to use each LED in an appropriate order to simulate a traffic light. The sequence is as follows: Red is turned on, followed by a wait; yellow is on, followed by a wait; red and yellow are turned off while green is turned on, followed by a wait; green is turned off while yellow is turned on, followed by a wait; finally, yellow is turned off which ends the loop to be restarted.

```
.EQU RED, 1
.EQU YELLOW, 2
.EQU GREEN, 3
.EQU GPIO_OUT, 1
.EQU sleep_time, 1000

.thumb_func
.global main
main:
        MOV R0, #RED
        BL gpio_init
        MOV R0, #RED
        MOV R1, #GPIO_OUT
        BL link_gpio_set_dir
        MOV R0, #YELLOW
        BL gpio_init
        MOV R0, #YELLOW
        MOV R1, #GPIO_OUT
        BL link_gpio_set_dir
        MOV R0, #GREEN
        BL gpio_init
        MOV R0, #GREEN
        MOV R1, #GPIO_OUT
        BL link_gpio_set_dir
loop:
        MOV R0, #RED
        MOV R1, #1
        BL link_gpio_put
        LDR R0, =sleep_time
        BL sleep_ms

        MOV R0, #YELLOW
        MOV R1, #1
        BL link_gpio_put
        LDR R0, =sleep_time
        BL sleep_ms


        MOV R0, #RED
        MOV R1, #0
        BL link_gpio_put
        MOV R0, #YELLOW
        MOV R1, #0
        BL link_gpio_put
        MOV R0, #GREEN
        MOV R1, #1
        BL link_gpio_put
        LDR R0, =sleep_time
        BL sleep_ms


        MOV R0, #YELLOW
```

```
MOV R1, #1
BL link_gpio_put
MOV R0, #GREEN
MOV R1, #0
BL link_gpio_put
LDR R0, =sleep_time
BL sleep_ms


MOV R0, #YELLOW
MOV R1, #0
BL link_gpio_put
B loop
```

Task 3:

The program shown below follows the provided instructions for task three, using three LEDs to count from binary 000 to 111 (dec0 to dec7), and down again in an infinite loop. This is done with a number of subroutines lighting up the appropriate LEDs representing each binary number (dec 1-7), and a loop which iterates through each one (0-7 and 7-0) with a 1,000 ms sleep time to allow the user to easily observe the changes in binary number represented with the three LEDs, as specified in the instructions.

```
.EQU ONE, 3
.EQU TWO, 2
.EQU THREE, 1
.EQU GPIO_OUT, 1
.EQU sleep_time, 1000

.thumb_func
.global main

main:
   MOV R0,#ONE
   BL gpio_init
   MOV R0, #ONE
   MOV R1, #GPIO_OUT
   BL link_gpio_set_dir

   MOV R0,#TWO
   BL gpio_init
   MOV R0, #TWO
   MOV R1, #GPIO_OUT
   BL link_gpio_set_dir

   MOV R0,#THREE
   BL gpio_init
   MOV R0, #THREE
   MOV R1, #GPIO_OUT
   BL link_gpio_set_dir

@ zero start
loop:
   BL one_on
   LDR R0, =sleep_time
   BL sleep_ms

   BL one_on
   LDR R0, =sleep_time
   BL sleep_ms

   BL two_on
   LDR R0, =sleep_time
   BL sleep_ms

   BL three_on
   LDR R0, =sleep_time
   BL sleep_ms

   BL four_on
   LDR R0, =sleep_time
   BL sleep_ms

   BL five_on
```

```
    LDR R0, =sleep_time
    BL sleep_ms

    BL six_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL seven_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL six_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL five_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL four_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL three_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL two_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL one_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL zero_on
    LDR R0, =sleep_time
    BL sleep_ms

    B loop
zero_on:
    PUSH {LR}
    MOV R0, #ONE
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #TWO
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #THREE
    MOV R1, #0
    BL link_gpio_put
    POP {PC}
    BX LR
@ zero end

@ one start
one_on:
    PUSH {LR}
    MOV R0, #ONE
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #TWO
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #THREE
    MOV R1, #0
```

```
    BL link_gpio_put
    POP {PC}
    BX LR

@ one end

@ two start
two_on:
    PUSH {LR}
    MOV R0, #ONE
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #TWO
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #THREE
    MOV R1, #0
    BL link_gpio_put
    POP {PC}
    BX LR

@ two end

@ three start
three_on:
    PUSH {LR}
    MOV R0, #ONE
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #TWO
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #THREE
    MOV R1, #0
    BL link_gpio_put

    POP {PC}
    BX LR
@ three end

@ four start
four_on:
    PUSH {LR}
    MOV R0, #ONE
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #TWO
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #THREE
    MOV R1, #1
    BL link_gpio_put

    POP {PC}
    BX LR

@ four end

@ five start
five_on:
    PUSH {LR}
    MOV R0, #ONE
    MOV R1, #1
    BL link_gpio_put
```

```
    MOV R0, #TWO
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #THREE
    MOV R1, #1
    BL link_gpio_put

    POP {PC}
    BX LR
@ five end

@ six start
six_on:
    PUSH {LR}
    MOV R0, #ONE
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #TWO
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #THREE
    MOV R1, #1
    BL link_gpio_put

    POP {PC}
    BX LR
@ six end

@ seven start
seven_on:
    PUSH {LR}
    MOV R0, #ONE
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #TWO
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #THREE
    MOV R1, #1
    BL link_gpio_put
    POP {PC}
    BX LR
@ seven end
```

Task 4:

This task works in essentially the same way as the previous task (three), with a higher complexity as there are more LEDs which require change. We've created subroutines for each number (0-9) which light up the appropriate LEDs to display the number on the 7 LED display, and we've created a loop to call these subroutines in the correct over (ascending and descending) which loops infinitely, with 100 milliseconds delay between each display as specified in the assignment.

```
.EQU A, 0
.EQU B, 1
.EQU C, 2
.EQU D, 3
.EQU E, 4
.EQU F, 5
.EQU G, 6
.EQU GPIO_OUT, 1
.EQU sleep_time, 1000

.thumb_func
.global main

main:
   MOV R0,#A
   BL gpio_init
   MOV R0, #A
   MOV R1, #GPIO_OUT
   BL link_gpio_set_dir

   MOV R0,#B
   BL gpio_init
   MOV R0, #B
   MOV R1, #GPIO_OUT
   BL link_gpio_set_dir

   MOV R0,#C
   BL gpio_init
   MOV R0, #C
   MOV R1, #GPIO_OUT
   BL link_gpio_set_dir

   MOV R0,#D
   BL gpio_init
   MOV R0, #D
   MOV R1, #GPIO_OUT
   BL link_gpio_set_dir

   MOV R0,#E
   BL gpio_init
   MOV R0, #E
   MOV R1, #GPIO_OUT
   BL link_gpio_set_dir

   MOV R0,#F
   BL gpio_init
   MOV R0, #F
   MOV R1, #GPIO_OUT
   BL link_gpio_set_dir

   MOV R0,#G
   BL gpio_init
```

```
    MOV R0, #G
    MOV R1, #GPIO_OUT
    BL link_gpio_set_dir

@ zero start
loop:
    BL zero_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL one_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL two_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL three_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL four_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL five_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL six_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL seven_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL eight_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL nine_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL eight_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL seven_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL six_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL five_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL four_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL three_on
    LDR R0, =sleep_time
    BL sleep_ms
```

```
    BL two_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL one_on
    LDR R0, =sleep_time
    BL sleep_ms

    BL zero_on
    LDR R0, =sleep_time
    BL sleep_ms

    B loop
zero_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #F
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #G
    MOV R1, #0
    BL link_gpio_put
    POP {PC}
    BX LR
@ zero end

@ one start
one_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #F
```

```
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #G
    MOV R1, #0
    BL link_gpio_put
    POP {PC}
    BX LR

@ one end

@ two start
two_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #F
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #G
    MOV R1, #1
    BL link_gpio_put
    POP {PC}
    BX LR

@ two end

@ three start
three_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #F
```

```
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #G
    MOV R1, #1
    BL link_gpio_put
    POP {PC}
    BX LR
@ three end

@ four start
four_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #F
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #G
    MOV R1, #1
    BL link_gpio_put
    POP {PC}
    BX LR

@ four end

@ five start
five_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #F
    MOV R1, #1
```

```
    BL link_gpio_put

    MOV R0, #G
    MOV R1, #1
    BL link_gpio_put
    POP {PC}
    BX LR
@ five end

@ six start
six_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #F
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #G
    MOV R1, #1
    BL link_gpio_put
    POP {PC}
    BX LR
@ six end

@ seven start
seven_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #F
    MOV R1, #0
    BL link_gpio_put
```

```
    MOV R0, #G
    MOV R1, #0
    BL link_gpio_put
    POP {PC}
    BX LR
@ seven end

@ eight start
eight_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #F
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #G
    MOV R1, #1
    BL link_gpio_put
    POP {PC}
    BX LR
@ eight end

@ nine start
nine_on:
    PUSH {LR}
    MOV R0, #A
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #B
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #C
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #D
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #E
    MOV R1, #0
    BL link_gpio_put

    MOV R0, #F
    MOV R1, #1
    BL link_gpio_put

    MOV R0, #G
    MOV R1, #1
```

```
    BL link_gpio_put
    POP {PC}
    BX LR
@ nine end
```