Meenakshi Shankar
Sr No. 22400
M.Tech EPD

<u>EMBEDDED SYSTEM DESIGN (E3-257)</u>

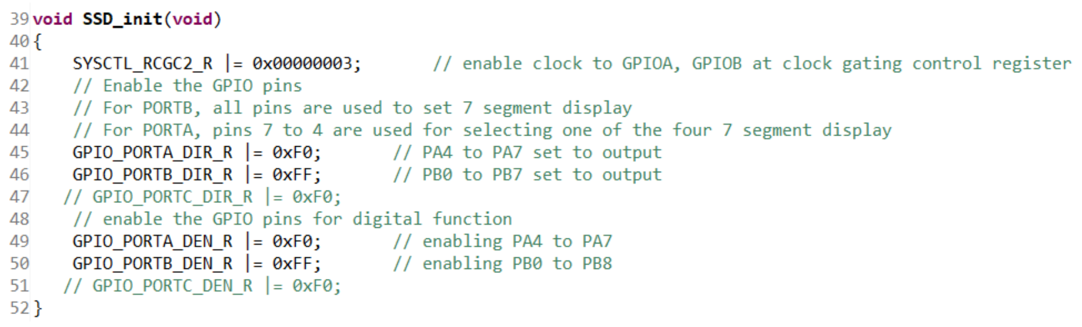
<u>LAB ASSIGNMENT – 4</u>

<u>Explanation of Code</u>

Apart from the functionalities implemented until lab 3 following new additions were made:

➔ ***<u>Initialized the seven-segment display:</u>***

Configured the port A and port B pins correctly to keep active and display the count and cycles of colour changes.



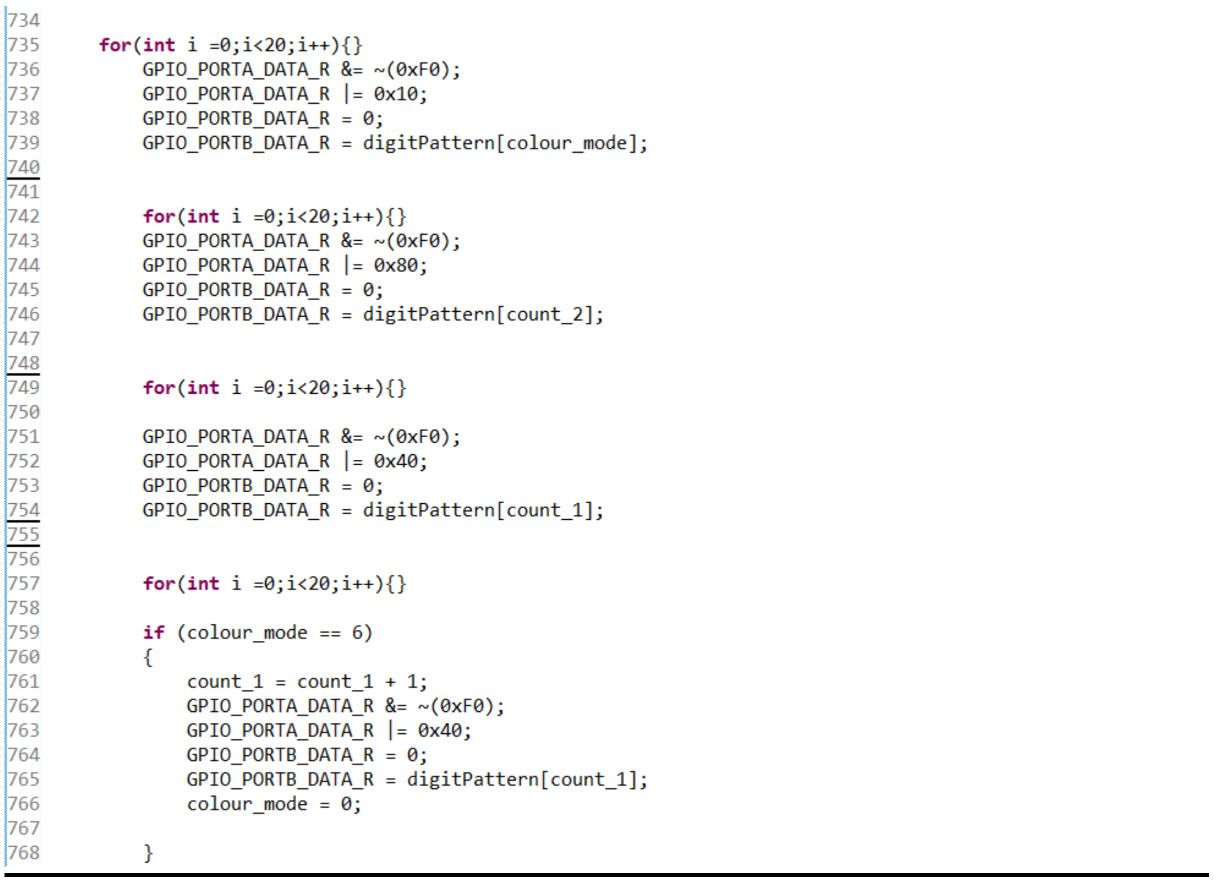
```
void SSD_init(void)
{
    SYSCTL_RCGC2_R |= 0x00000003;       // enable clock to GPIOA, GPIOB at clock gating control register
    // Enable the GPIO pins
    // For PORTB, all pins are used to set 7 segment display
    // For PORTA, pins 7 to 4 are used for selecting one of the four 7 segment display
    GPIO_PORTA_DIR_R |= 0xF0;       // PA4 to PA7 set to output
    GPIO_PORTB_DIR_R |= 0xFF;       // PB0 to PB7 set to output
   // GPIO_PORTC_DIR_R |= 0xF0;
    // enable the GPIO pins for digital function
    GPIO_PORTA_DEN_R |= 0xF0;       // enabling PA4 to PA7
    GPIO_PORTB_DEN_R |= 0xFF;       // enabling PB0 to PB8
   // GPIO_PORTC_DEN_R |= 0xF0;
}
```

➔ Kept the display steady to parallelly out 3 digits using very minimal delay between flickering to meet persistence of vision as below

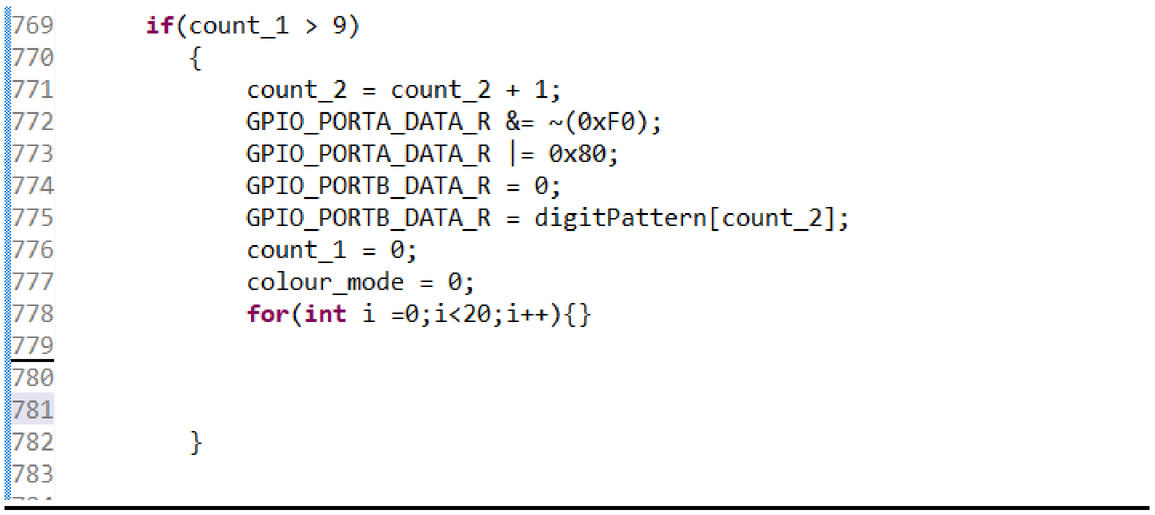
➔ Display at Zeroth place reflects the colour code going from 0 for green to 6 till white and updating the 3rd display by 1 to account for finishing one cycle.

➔ Similarly, as the cycle updates to 2-digit count 4th place is updates once 3rd display crosses 9. Same is shown below in code:

Meenakshi Shankar
Sr No. 22400
M.Tech EPD



```

    for(int i =0;i<20;i++){}
        GPIO_PORTA_DATA_R &= ~(0xF0);
        GPIO_PORTA_DATA_R |= 0x10;
        GPIO_PORTB_DATA_R = 0;
        GPIO_PORTB_DATA_R = digitPattern[colour_mode];


        for(int i =0;i<20;i++){}
        GPIO_PORTA_DATA_R &= ~(0xF0);
        GPIO_PORTA_DATA_R |= 0x80;
        GPIO_PORTB_DATA_R = 0;
        GPIO_PORTB_DATA_R = digitPattern[count_2];


        for(int i =0;i<20;i++){}

        GPIO_PORTA_DATA_R &= ~(0xF0);
        GPIO_PORTA_DATA_R |= 0x40;
        GPIO_PORTB_DATA_R = 0;
        GPIO_PORTB_DATA_R = digitPattern[count_1];


        for(int i =0;i<20;i++){}

        if (colour_mode == 6)
        {
            count_1 = count_1 + 1;
            GPIO_PORTA_DATA_R &= ~(0xF0);
            GPIO_PORTA_DATA_R |= 0x40;
            GPIO_PORTB_DATA_R = 0;
            GPIO_PORTB_DATA_R = digitPattern[count_1];
            colour_mode = 0;

        }
```



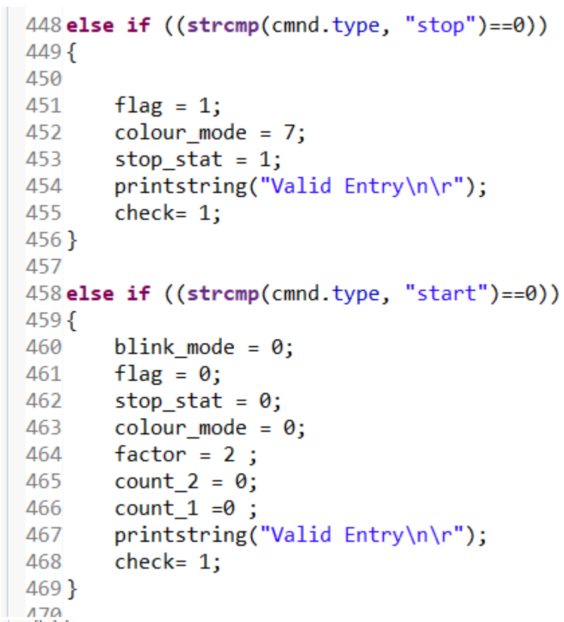
```
        if(count_1 > 9)
            {
                count_2 = count_2 + 1;
                GPIO_PORTA_DATA_R &= ~(0xF0);
                GPIO_PORTA_DATA_R |= 0x80;
                GPIO_PORTB_DATA_R = 0;
                GPIO_PORTB_DATA_R = digitPattern[count_2];
                count_1 = 0;
                colour_mode = 0;
                for(int i =0;i<20;i++){}



            }

```

➔ ***Defining the new states***

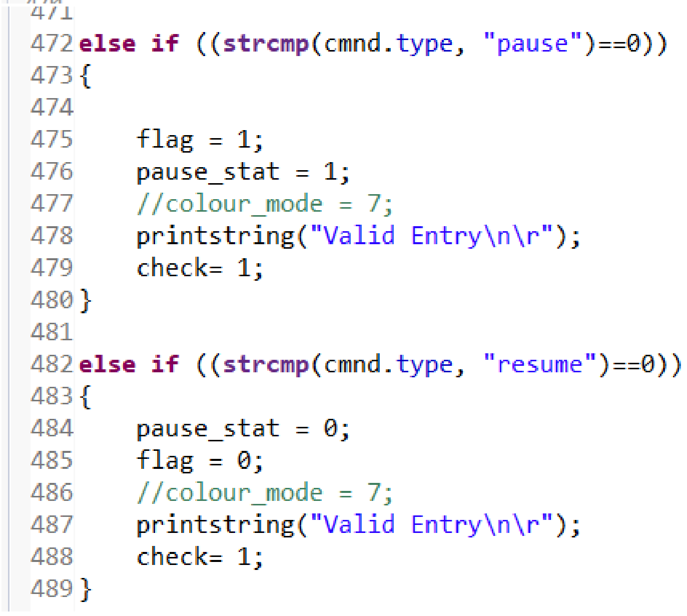
As per the task requirement the following new states were added to be reflected as console commands as well as key presses.

Meenakshi Shankar
Sr No. 22400
M.Tech EPD

➔ Given below is the addition to code to accept these states via console commands:



```c
else if ((strcmp(cmnd.type, "stop")==0))
{

    flag = 1;
    colour_mode = 7;
    stop_stat = 1;
    printstring("Valid Entry\n\r");
    check= 1;
}

else if ((strcmp(cmnd.type, "start")==0))
{
    blink_mode = 0;
    flag = 0;
    stop_stat = 0;
    colour_mode = 0;
    factor = 2 ;
    count_2 = 0;
    count_1 =0 ;
    printstring("Valid Entry\n\r");
    check= 1;
}
```

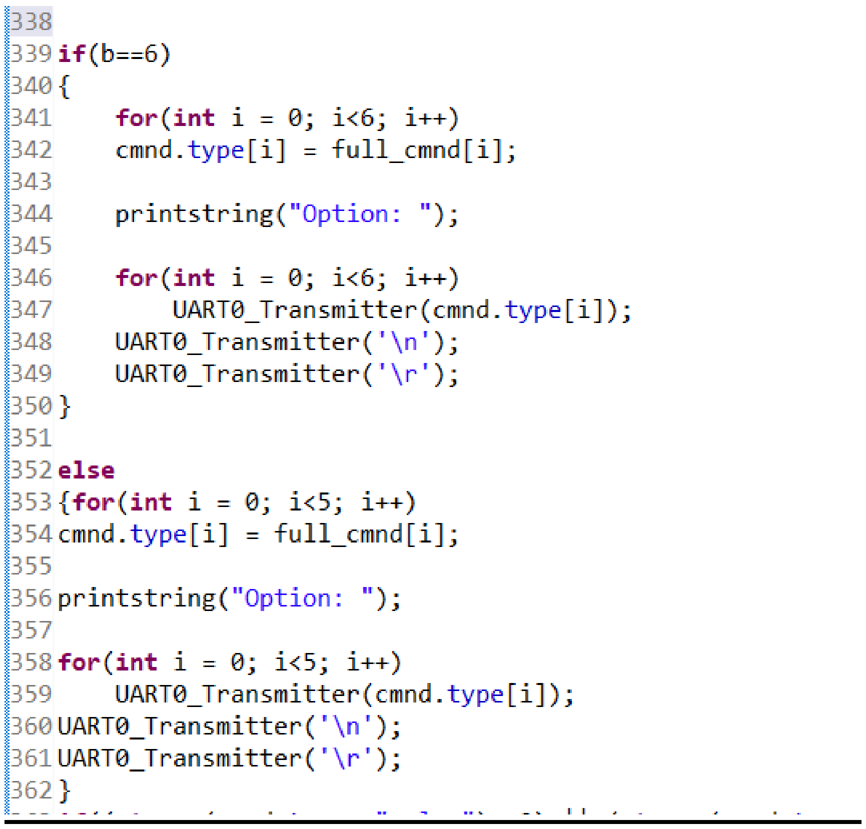


```c

else if ((strcmp(cmnd.type, "pause")==0))
{

    flag = 1;
    pause_stat = 1;
    //colour_mode = 7;
    printstring("Valid Entry\n\r");
    check= 1;
}

else if ((strcmp(cmnd.type, "resume")==0))
{
    pause_stat = 0;
    flag = 0;
    //colour_mode = 7;
    printstring("Valid Entry\n\r");
    check= 1;
}
```

➔ ***Handling the new command length and input style from console:***

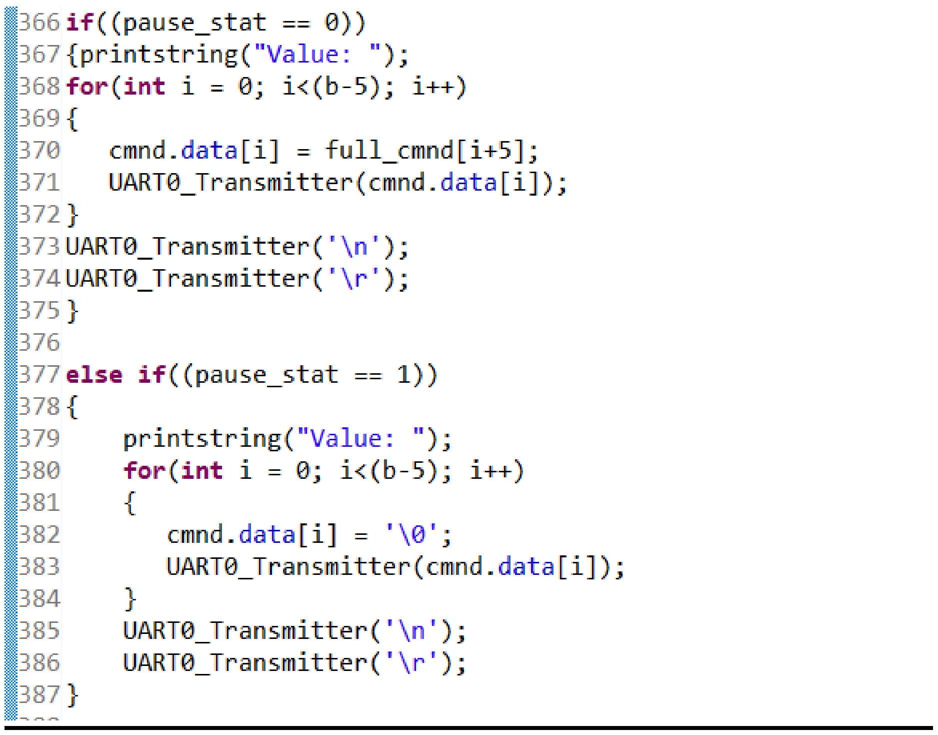
The previous UART console assignment the command lengths were fixed, but the addition of new commands required to play around with string manipulation to adjust and accept them differently as shown below:

Meenakshi Shankar
Sr No. 22400
M.Tech EPD



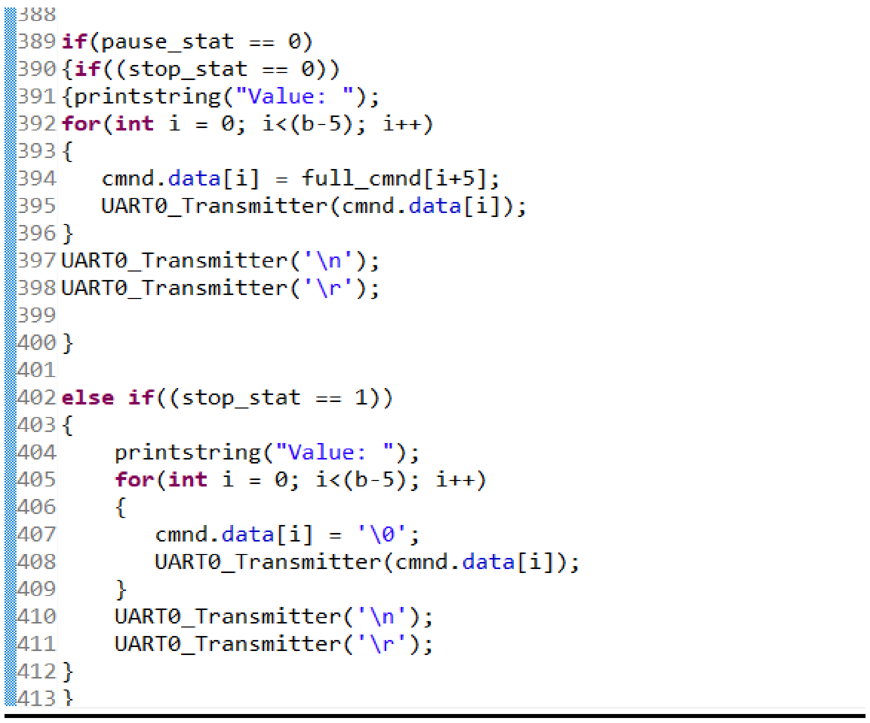
```
if(b==6)
{
    for(int i = 0; i<6; i++)
    cmnd.type[i] = full_cmnd[i];

    printstring("Option: ");

    for(int i = 0; i<6; i++)
        UART0_Transmitter(cmnd.type[i]);
    UART0_Transmitter('\n');
    UART0_Transmitter('\r');
}

else
{for(int i = 0; i<5; i++)
cmnd.type[i] = full_cmnd[i];

printstring("Option: ");

for(int i = 0; i<5; i++)
    UART0_Transmitter(cmnd.type[i]);
UART0_Transmitter('\n');
UART0_Transmitter('\r');
}
```

➔ Since the new commands “stop”, “pause”, ”start”,” resume” are not having any value to accompany as in command “color” and “blink” we have just assigned zeros to cmnd.data[] item of struct declared initially.



```
if((pause_stat == 0))
{printstring("Value: ");
for(int i = 0; i<(b-5); i++)
{
    cmnd.data[i] = full_cmnd[i+5];
    UART0_Transmitter(cmnd.data[i]);
}
UART0_Transmitter('\n');
UART0_Transmitter('\r');
}

else if((pause_stat == 1))
{
    printstring("Value: ");
    for(int i = 0; i<(b-5); i++)
    {
        cmnd.data[i] = '\0';
        UART0_Transmitter(cmnd.data[i]);
    }
    UART0_Transmitter('\n');
    UART0_Transmitter('\r');
}
```

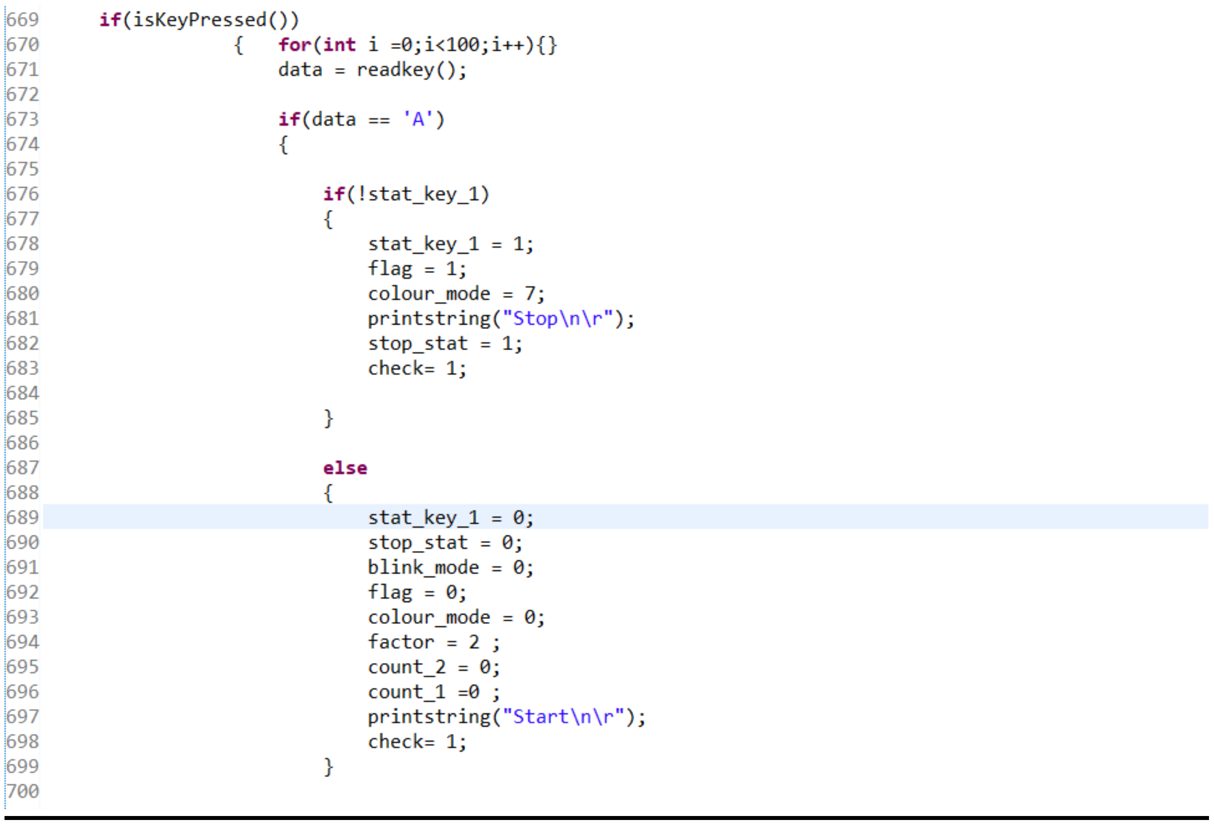
Meenakshi Shankar
Sr No. 22400
M.Tech EPD



```

if(pause_stat == 0)
{if((stop_stat == 0))
{printstring("Value: ");
for(int i = 0; i<(b-5); i++)
{
   cmnd.data[i] = full_cmnd[i+5];
   UART0_Transmitter(cmnd.data[i]);
}
UART0_Transmitter('\n');
UART0_Transmitter('\r');

}

else if((stop_stat == 1))
{
    printstring("Value: ");
    for(int i = 0; i<(b-5); i++)
    {
       cmnd.data[i] = '\0';
       UART0_Transmitter(cmnd.data[i]);
    }
    UART0_Transmitter('\n');
    UART0_Transmitter('\r');
}
}
```

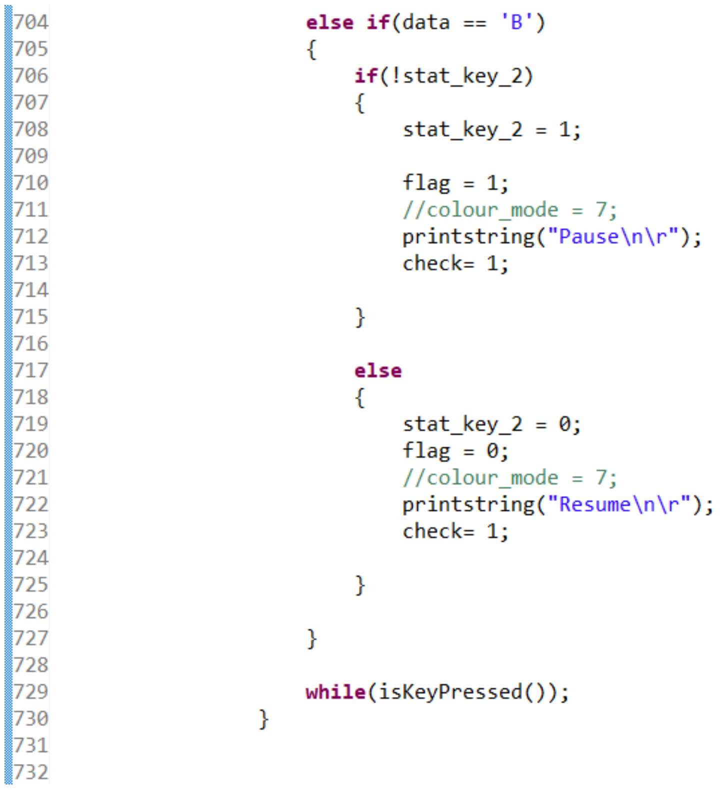
➔ ***Using keypad to toggle between states***

➔ We add the same state definitions to after keypress is checked for and toggle state is realised for either of the first two switches in the training kit:



```
    if(isKeyPressed())
                {   for(int i =0;i<100;i++){}
                    data = readkey();

                    if(data == 'A')
                    {

                        if(!stat_key_1)
                        {
                            stat_key_1 = 1;
                            flag = 1;
                            colour_mode = 7;
                            printstring("Stop\n\r");
                            stop_stat = 1;
                            check= 1;

                        }

                        else
                        {
                            stat_key_1 = 0;
                            stop_stat = 0;
                            blink_mode = 0;
                            flag = 0;
                            colour_mode = 0;
                            factor = 2 ;
                            count_2 = 0;
                            count_1 =0 ;
                            printstring("Start\n\r");
                            check= 1;
                        }

```

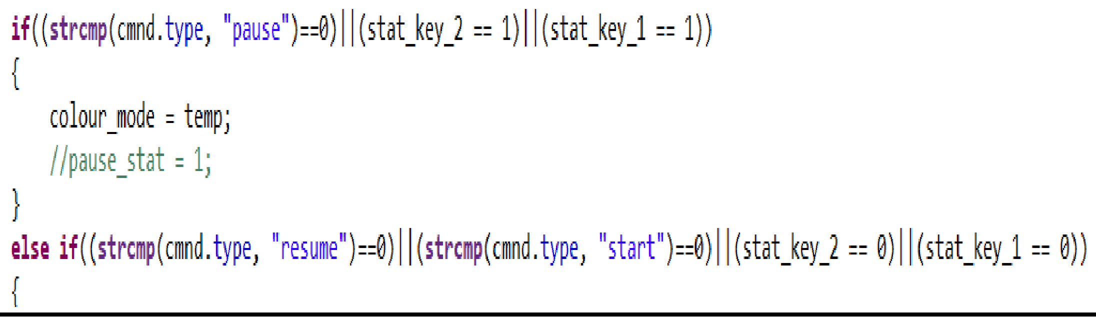
Meenakshi Shankar
Sr No. 22400
M.Tech EPD



```
                else if(data == 'B')
                {
                    if(!stat_key_2)
                    {
                        stat_key_2 = 1;

                        flag = 1;
                        //colour_mode = 7;
                        printstring("Pause\n\r");
                        check= 1;

                    }

                    else
                    {
                        stat_key_2 = 0;
                        flag = 0;
                        //colour_mode = 7;
                        printstring("Resume\n\r");
                        check= 1;

                    }

                }

                while(isKeyPressed());
            }
```

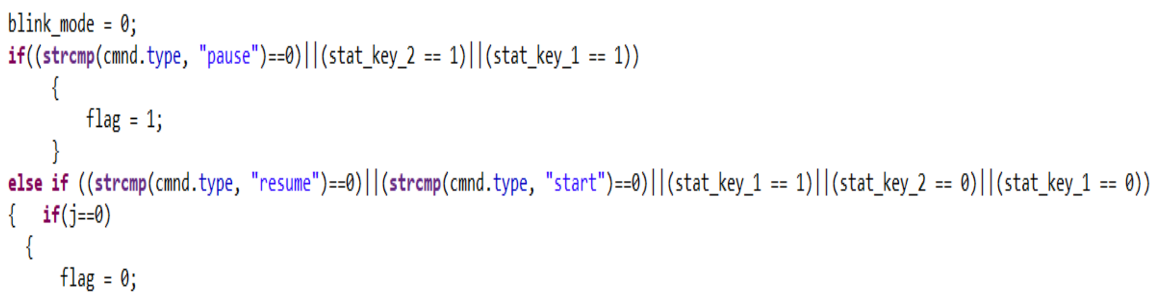
<u>→ Ensuring the States don't change by push button or console if its in "pause" or "stop" state until "start"/"resume" comes in</u>

Using certain flag assignments, we keep checking what state it is while the GPIO interrupts are happening if the flag indicating a pause/ stop state is high we try to maintain the colour_mode and blink rate or turn them off completely respectively.



```
if((strcmp(cmnd.type, "pause")==0)||(stat_key_2 == 1)||(stat_key_1 == 1))
{
    colour_mode = temp;
    //pause_stat = 1;
}
else if((strcmp(cmnd.type, "resume")==0)||(strcmp(cmnd.type, "start")==0)||(stat_key_2 == 0)||(stat_key_1 == 0))
{
```

Meenakshi Shankar
Sr No. 22400
M.Tech EPD



```
blink_mode = 0;
if((strcmp(cmnd.type, "pause")==0)||(stat_key_2 == 1)||(stat_key_1 == 1))
    {
        flag = 1;
    }
else if ((strcmp(cmnd.type, "resume")==0)||(strcmp(cmnd.type, "start")==0)||(stat_key_1 == 1)||(stat_key_2 == 0)||(stat_key_1 == 0))
{   if(j==0)
  {
      flag = 0;
```