ECE 355 Assignment 2

Hai Anh Nguyen

V00894486

1.

```
#define PAOUT (volatile unsigned char *) 0xFFFFFFF1
#define PADIR (volatile unsigned char *) 0xFFFFFFF2
#define PBIN (volatile unsigned char *) 0xFFFFFFF3
#define PBOUT (volatile unsigned char *) 0xFFFFFFF4
#define PBDIR (volatile unsigned char *) 0xFFFFFFF5
#define CNTM (volatile unsigned int *) 0xFFFFFFD0
#define COUNT (volatile unsigned int *) 0xFFFFFFD4
#define CTCON (volatile unsigned char *) 0xFFFFFFD8
#define CTSTAT (volatile unsigned char *) 0xFFFFFFD9
#define IVECT (volatile unsigned int *) (0x20)
#define LEDoff (0x0)
#define LEDAon (0x4)
#define LEDBon (0x8)


interrupt void intserv();
unsigned char digitA = 0;
unsigned char digitB = 0;
unsigned char isA = 1;

int main() {
	*CTCON = 0x2; /* Stop Timer (if running) */
	*CTSTAT = 0x0; /* Clear “reached 0” flag */
	*PBDIR = 0x8F; /* Configure Port B direction*/
	*PADIR = 0xF4; /* Configure Port A direction*/
	*CNTM = 100000000; /* Initialize Timer */
	*IVECT = (unsigned int*)&intserv; /* Set up interrupt vector */
	asm(“ MoveControl PSR, #0x40 ”); /* CPU responds to IRQ */
	*CTCON = 0x11; /* Enable Timer interrupts and start counting */
	*PAOUT = LEDAon; /* Turn ON LED1 and display 0 */
	*PBOUT = LEDoff; /* Turn OFF LED2 and display 0 */

	while (1) {
		while ((*PBIN & 0x10) != 0); /* Wait until SW is pressed */
		while ((*PBIN & 0x10) == 0); /* Wait until SW is released */
		if (isA) {
			*PAOUT = (unsigned char)((digitA << 4) | LEDoff); /* Turn off LED1
*/
			*PBOUT = (unsigned char)((LEDBon << 4) | digitB); /* Turn on LED2 */
			isA = 0;
		}
		else {
			*PAOUT = (unsigned char)((digitA << 4) | LEDAon); /* Turn on LED1 */
```

```
                        *PBOUT = (unsigned char)((LEDoff << 4) | digitB); /* Turn off LED2
*/
                        isA = 1;
                }
        }
        exit(0);
}

interrupt void intserv() {
        *CTSTAT = 0x0; /* Clear “reached 0” flag */
        if (isA) {
                digitA = (digitA + 1) % 10; /* Increment digit A*/
                *PAOUT = (unsigned char)((digitA << 4) | LEDAon); /* Update port A and keep
LED1 on */
        }
        else {
                digitB = (digitB + 1) % 10;
                *PBOUT = (unsigned char)((LEDBon << 4) | digitB); /* Update port B and keep
LED2 on*/
        }
}
```

2.

```
#define PAOUT (volatile unsigned char *) 0xFFFFFFF1
#define PADIR (volatile unsigned char *) 0xFFFFFFF2
#define PBIN (volatile unsigned char *) 0xFFFFFFF3
#define PBOUT (volatile unsigned char *) 0xFFFFFFF4
#define PBDIR (volatile unsigned char *) 0xFFFFFFF5
#define CNTM (volatile unsigned int *) 0xFFFFFFD0
#define COUNT (volatile unsigned int *) 0xFFFFFFD4
#define CTCON (volatile unsigned char *) 0xFFFFFFD8
#define CTSTAT (volatile unsigned char *) 0xFFFFFFD9
#define IVECT (volatile unsigned int *) (0x20)
#define PCONT (volatile unsigned char *) 0xFFFFFFE7
#define LEDoff (0x0)
#define LED1on (0x4)
#define LED2on (0x1)


interrupt void intserv();
unsigned char digitA = 0;
unsigned char digitB = 0;
unsigned char isLED1 = 1;



int main() {
        *CTCON = 0x2; /* Stop Timer (if running) */
```

```
	*CTSTAT = 0x0; /* Clear “reached 0” flag */
	*PBDIR = 0xF5; /* Configure Port B direction*/
	*PADIR = 0x78; /* Configure Port A direction*/
	*CNTM = 100000000; /* Initialize Timer */
	*IVECT = (unsigned int*)&intserv; /* Set up interrupt vector */
	asm(“ MoveControl PSR, #0x40 ”); /* CPU responds to IRQ */
	*CTCON = 0x01; /* Start counting */
	*PCONT |= 0x10; /*enable PAIN interrupts*/
	*PAOUT = (unsigned char)((digitA << 3)); /* display 0 port A */
	*PBOUT = (unsigned char)((digitB << 4) | LED1on); /* display 0 port B and turn on
LED1*/

	while (1) {
		while ((*CTSTAT & 0x1) == 0); /* Wait until 0 is reached */
		if (isLED1) {
			digitA = (digitA + 1) % 10; /* Increment digit */
			*CTSTAT = 0x0; /* Clear “Reached 0” flag */
			*PAOUT = (unsigned char)((digitA << 3));
		}
		else {
			digitB = (digitB + 1) % 10;  /* Increment digit */
			*CTSTAT = 0x0;
			*PBOUT = (unsigned char)((digitB << 4) | LED2on);
		}
	}
	exit(0);

}


interrupt void intserv() {
	if ((*PSTAT & 0x10) == 1) { //IAIN event
		*PSTAT &= 0xFE; // Clear PASIN flag
		if (*PAIN & 0x80 == 1)  // PAIN changed and now PA[7] = 1 -> must changed
from 0 to 1
		{
			if (isLED1) {
				isLED1 = 0;
				*PBOUT = (unsigned char)((digitB << 4) | LED2on);
			}
			else {
				isLED1 = 1;
				*PBOUT = (unsigned char)((digitB << 4) | LED1on);
			}
		}
	}
	}
```