Midterm Solutions

1.

```
#define PAIN (volatile unsigned char *) 0xFFFFFFF0
#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 CTCON (volatile unsigned char *) 0xFFFFFFD8
#define CTSTAT (volatile unsigned char *) 0xFFFFFFD9
#define IVECT (volatile unsigned int *) (0x20)

interrupt void intserv();

volatile unsigned char digit1 = 0;        /* DIGIT1 for display */
volatile unsigned char digit2 = 0;        /* DIGIT2 for display */
volatile unsigned char leds = 0x1;        /* LED1 on, LED2 off */

int main() {
  *PADIR = 0xF4;                          /* Set Port A direction */
  *PBDIR = 0x8F;                          /* Set Port B direction */
  *CTCON = 0x2;                           /* Stop Timer (if running) */
  *CNTM = 100000000;                      /* Initialize: 1-s timeout */
  *CTSTAT = 0x0;                          /* Clear “Reached 0” flag */
  *IVECT = (unsigned int *) &intserv;     /* Set interrupt vector */
  asm(“MoveControl PSR,#0x40”);           /* CPU responds to IRQ */
  *PAOUT = 0x0;                           /* Initialize port A */
  *PBOUT = 0x80;                          /* Initialize port B */
  *CTCON = 0x11;                          /* Start Timer
  while (1) {
    while ((*PBIN & 0x10) != 0);          /* Wait for SW press */
    while ((*PBIN & 0x10) == 0);          /* Wait for SW release */
    leds ^= 0x1;                          /* Toggle LED flag
    *PAOUT ^= 0x04;                       /* Flip LED1 state
    *PBOUT ^= 0x80;                       /* Flip LED2 state
  }
  exit(0);
}

interrupt void intserv() {
  *CTSTAT = 0x0;                          /* Clear “Reached 0” flag */
  if (leds == 0x1) {
      digit1 = (digit1+1)%10;             /* Increment DIGIT1 */
      *PAOUT = digit1 << 4;               /* Update port A, LED1 on */
    }
    else {
      digit2 = (digit2+1)%10;             /* Increment DIGIT2 */
      *PBOUT = digit2;                    /* Update port B, LED2 on */
    }
}
```

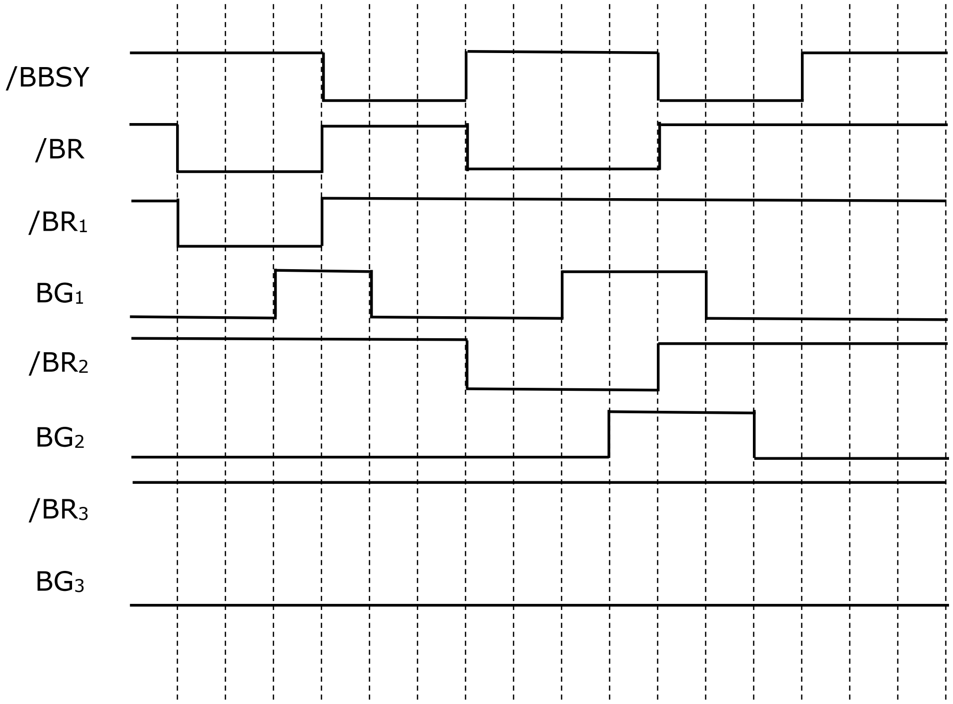
2.

The LCM (least common multiple) of all four periods is 100, i.e., we only need to determine our schedule in the time interval **[0, 100)**. RM task priorities are 1/20 for T1 arriving at (0, 20, 40, 60, 80); 1/25 for T2 arriving at (0, 25, 50, 75); 1/50 for T3 arriving at (0, 50); 1/100 for T4 arriving at (0).

<u>RM Schedule</u>

t=0:	T1
t=5:	T2
t=10:	T3
t=20:	T1
t=25:	T2
t=30:	T4
t=40:	T1 (T4 preempted)
t=45:	T4
t=50:	T2
t=55:	T3
t=60:	T1 (T3 preempted)
t=65:	T3
t=70:	Idle
t=75:	T2
t=80:	T1
t=85:	Idle
t=100:	Repeat…

3.


/BBSY
/BR
/BR$_1$
BG$_1$
/BR$_2$
BG$_2$
/BR$_3$
BG$_3$