Solution 2

1.

```
#define PBIN (volatile unsigned char *) 0xFFFFFFF3
#define PBOUT (volatile unsigned char *) 0xFFFFFFF4
#define PBDIR (volatile unsigned char *) 0xFFFFFFF5
#define PSTAT (volatile unsigned char *) 0xFFFFFFF6
#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 digit = 0;          /* digit for display */

int main() {
  unsigned char sample = 0;                /* Port B input sample */

  *PBDIR = 0xF0;                           /* Set Port B direction */
  *CTCON = 0x2;                            /* Stop Timer (if running) */
  *CTSTAT = 0x0;                           /* Clear "Reached 0" flag */
  *CNTM = 100000000;                       /* Initialize: 1-s timeout */
  *IVECT = (unsigned int *) &intserv;      /* Set interrupt vector */
  asm("MoveControl PSR,#0x40");            /* CPU responds to IRQ */
  *CTCON = 0x1;                            /* Start Timer, disable
                                              interrupts for now */
  *PBOUT = 0x0;                            /* Display 0 */
  while (1) {
    while ((*PSTAT & 0x4) == 0);           /* Wait for PBIN update */
    sample = *PBIN & 0x3;      /* Sample PBIN, isolate bits [1:0] */
    if (sample == 0x1)         /* E = 0, D = 1 */
      *CTCON |= 0x10;          /* Enable Timer interrupts */
    else if (sample == 0x2)    /* E = 1, D = 0 */
      *CTCON &= 0xEF;          /* Disable Timer interrupts */
  }
  exit(0);
}

interrupt void intserv() {
  *CTSTAT = 0x0;               /* Clear "Reached 0" flag */
  digit = (digit + 1)%10;      /* Increment digit */
  *PBOUT = digit << 4;         /* Update display */
}
```

2.

```
#define PBIN (volatile unsigned char *) 0xFFFFFFF3
#define PBOUT (volatile unsigned char *) 0xFFFFFFF4
#define PBDIR (volatile unsigned char *) 0xFFFFFFF5
#define PCONT (volatile unsigned char *) 0xFFFFFFF7
#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();

int main() {
  char digit = 0;                          /* Digit to be displayed */
  *PBDIR = 0xF0;                           /* Set Port B direction */
  *IVECT = (unsigned int *) &intserv;      /* Set interrupt vector */
  asm("MoveControl PSR,#0x40");            /* CPU responds to IRQ */
  *PCONT = 0x40;                           /* Enable PBIN interrupts */
  *CTCON = 0x2;                            /* Stop Timer */
  *CTSTAT = 0x0;                           /* Clear "reached 0" flag */
  *CNTM = 100000000;                       /* Initialize Timer */
  *PBOUT = 0x0;                            /* Display 0 */
  while (1) {
    while ((*CTSTAT & 0x1) == 0);          /* Wait until 0 is reached */
    *CTSTAT = 0x0;                         /* Clear "reached 0" flag */
    digit = (digit + 1)%10;                /* Increment digit */
    *PBOUT = digit << 4;                   /* Update display */
  }
  exit(0);
}

interrupt void intserv() {
  unsigned char sample;       /* Port B input sample */
  sample = *PBIN & 0x3;       /* Sample PBIN, isolate bits [1:0] */
  if (sample == 0x1) *CTCON = 0x1;        /* Start Timer */
  else if (sample == 0x2) *CTCON = 0x2;   /* Stop Timer */
}
```

3. The LCM (least common multiple) of all four periods is 80, i.e., we only need to determine our EDF schedule in the time interval **[0, 80)**, after which it is repeated.

EDF task priorities are: (1/16, 1/36, 1/56, 1/76) for T1 arriving at (0, 20, 40, 60); (1/32, 1/72) for T2 arriving at (0, 40); (1/35, 1/75) for T3 arriving at (0, 40); (1/70) for T4 arriving at (0).

t=0: T1
t=4: T2
t=8: T3
t=20: T1
t=24: T4
t=40: T1 (T4 preempted)
t=44: T4
t=52: T2
t=56: T3
t=60: T3
t=68: T1
t=72: Idle
t=80: Repeat…