Solution 1

1. One of possible solutions is shown below.

```
#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 */
  *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 == 0x2)        /* D = 1, E = 0 (pressed) */
      *CTCON = 0x11;          /* Start Timer, enable interrupts */
    else if (sample == 0x1)   /* D = 0 (pressed), E = 1 */
      *CTCON = 0x2;           /* Stop Timer, disable interrupts */
  }
  exit(0);
}

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

2. One of possible solutions is shown below.

```
#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 == 0x2) *CTCON = 0x1;         /* Start Timer */
  else if (sample == 0x1) *CTCON = 0x2;    /* Stop Timer */
}
```

3. Let **x** denote the I/O device activity percentage to be determined.

Maximum I/O data access rate for DMA: $\mathbf{R_{I/O}/d_{I/O\text{-}DMA} = 256}$ transfers/s.
DMA cost: $\mathbf{(x*256)(N_{DMA\text{-}start}+N_{DMA\text{-}end}) = x*230{,}400}$ cycles/s.

Maximum I/O data access rate for polling: $\mathbf{R_{I/O}/d_{I/O} = 16{,}384}$ transfers/s.
Polling cost: $\mathbf{(x*16{,}384)N_{poll\text{-}ready} + ((1-x)*16{,}384)N_{poll\text{-}not\text{-}ready} = x*3{,}276{,}800 + 1{,}638{,}400}$ cycles/s.

We know that the DMA cost is 1,000 times cheaper than the polling cost; therefore, $\mathbf{1{,}000*(x*230{,}400) = x*3{,}276{,}800 + 1{,}638{,}400}$, which yields $\mathbf{x \approx 0.0072}$, or **0.72%**.