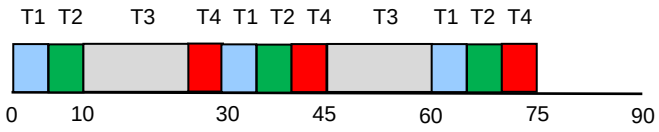
CENG 355 Midterm Solutions (2014)

1. There are many possible solutions. One of them is shown below.

```
#define PAOUT (volatile unsigned char *) 0xFFFFFFF1
#define PADIR (volatile unsigned char *) 0xFFFFFFF2
#define PBIN (volatile unsigned char *) 0xFFFFFFF3
#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();

unsigned char digit = 0;            /* Digit to be displayed */
unsigned char led = 0x1;            /* LED state: 0/1 = on/off */

int main() {
  *PADIR = 0xF1;                          /* Set Port A direction */
  *PBDIR = 0x00;                          /* Set Port B direction */
  *CTCON = 0x02;                          /* Stop Timer */
  *CTSTAT = 0x0;                          /* Clear "reached 0" flag */
  *CNTM = 100000000;                      /* Initialize Timer */
  *IVECT = (unsigned int *) &intserv;     /* Set interrupt vector */
  asm("MoveControl PSR,#0x40");           /* CPU responds to IRQ */
  *CTCON = 0x11;                          /* Enable Timer interrupts
                                           * and start counting */
  *PAOUT = 0x01;                          /* Display 0, turn LED off */
  while (1) {
    while ((*PBIN & 0x1) != 0);     /* Wait until SW is pressed */
    while ((*PBIN & 0x1) == 0);     /* Wait until SW is released */
    if (led == 0x1) led = 0x0;      /* If off, turn LED on */
    else led = 0x1;                 /* Else, turn LED off */
    *PAOUT = ((digit << 4) | led);  /* Update Port A */
    /* We can also put "*CTCON &= 0xEF;" before and "*CTCON |= 0x10;"
     * after the last statement, to make sure that intserv() is not
     * interfering with main() accessing shared digit/led/PAOUT */
  }

  exit(0);
}

interrupt void intserv() {
  *CTSTAT = 0x0;                    /* Clear "reached 0" flag */
  digit = (digit + 1)%10;           /* Increment digit */
  *PAOUT = ((digit << 4) | led);    /* Update Port A */
}
```

2. The LCM (least common multiple) of all four periods is 90; hence, we only need to figure out our RM schedule in the time interval **[0, 90)**, after which it is repeated:



RM task priorities are 1/30 for T1, 1/30 for T2, 1/45 for T3, and 1/90 for T4. Given that T1 and T2 have the same priority, we (arbitrarily) let T1 win over T2.

3.

(a) Direct-mapped: 2-bit **Block = A_{6-5}**, 3-bit **Word = A_{4-2}**; miss rate = 5/10.

Tag	Word 7	Word 6	Word 5	Word 4	Word 3	Word 2	Word 1	Word 0	
00001	[09C]	[098]	[094]	[090]	[08C]	[088]	[084]	[080]	Block 0
00100	[23C]	[238]	[234]	[230]	[22C]	[228]	[224]	[220]	Block 1
00100	[25C]	[258]	[254]	[250]	[24C]	[248]	[244]	[240]	Block 2
									Block 3

(b) 4-way set-associative: 1-bit **Set = A_5**, 3-bit **Word = A_{4-2}**; miss rate = 5/10.

Tag	Word 7	Word 6	Word 5	Word 4	Word 3	Word 2	Word 1	Word 0	
001000	[21C]	[218]	[214]	[210]	[20C]	[208]	[204]	[200]	Set 0
000010	[09C]	[098]	[094]	[090]	[08C]	[088]	[084]	[080]	Set 0
001000	[23C]	[238]	[234]	[230]	[22C]	[228]	[224]	[220]	Set 1
									Set 1

(c) Fully associative: 3-bit **Word = A_{4-2}**; miss rate = 4/10.

Tag	Word 7	Word 6	Word 5	Word 4	Word 3	Word 2	Word 1	Word 0
0000100	[09C]	[098]	[094]	[090]	[08C]	[088]	[084]	[080]
0010010	[25C]	[258]	[254]	[250]	[24C]	[248]	[244]	[240]
0010000	[21C]	[218]	[214]	[210]	[20C]	[208]	[204]	[200]
0010001	[23C]	[238]	[234]	[230]	[22C]	[228]	[224]	[220]