Prvi kolokvijum iz Operativnih sistema 1
Odsek za računarsku tehniku i informatiku
April 2014.

1. **(10 poena)**

```
IORequest *dma1Pending = 0, *dma2Pending = 0; // Currently pending requests

void startDMA1 () {
  if (ioHead!=0 && dma1Pending==0) {
    dma1Pending = ioHead; // Take the first request,
    ioHead = ioHead->next; // remove it from the list
    *dma1Address = dma1Pending->buffer; // and assign it to DMA1
    *dma1Count = dma1Pending->size;
    *dma1Ctrl = 1; // Start I/O
  }
}

void startDMA2 () {
  if (ioHead!=0 && dma2Pending == 0) {
    dma2Pending = ioHead; // Take the first request,
    ioHead = ioHead->next; // remove it from the list
    *dma2Address = dma2Pending->buffer; // and assign it to DMA2
    *dma2Count = dma2Pending->size;
    *dma2Ctrl = 1; // Start I/O
  }
}

void transfer () {
  startDMA1();
  startDMA2();
}

interrupt void dmaInterrupt () {
  if (dma1Status&1) { // DMA1 completed
    if (dma1Pending==0) return; // Exception
    if (*dma1Status&2) // Error in I/O
      dma1Pending->status = -1;
    else
      dma1Pending->status = 0;
    dma1Pending = 0;
    startDMA1();
  }
  if (dma2Status&1) { // DMA1 completed
    if (dma2Pending==0) return; // Exception
    if (*dma2Status&2) // Error in I/O
      dma2Pending->status = -1;
    else
      dma2Pending->status = 0;
    dma2Pending = 0;
    startDMA2();
  }
}
```

2.   (10 poena)

Virtual address (hex)	Mapping result (hex)
12FA0	X
C00F0	D670F0
70750	X
B0140	P
C02AB	D672AB

3.   (10 poena)

```
yield:          ; Save current context
                load  base,#cur[sp]
                add   base,base,#offsContext
                saveregs
                ; Restore new context
                load  base,#nxt[sp]
                add   base,base,#offsContext
                loadregs
                ; Return
                ret
```

4.   (10 poena)

```
int create_thread (void (*f)(void*), void* arg) {
  asm {
    load r0,#0
    load r1,#f[sp]
    load r2,#arg[sp]
    int  44h
  }
}

void wrapper (void* t) {
  if(t)((Thread*)t)->run();
}

int Thread::start () {
  if(pid)
    return pid;
  else
    return pid = create_thread(&wrapper,this);
}
```