HÁSKÓLINN Í REYKJAVÍK
REYKJAVIK UNIVERSITY

Benjamín Aage B. Birgisson

Exercise 9.1.1:

(a)

CPU time is wasted in executing the context switch and the interrupt handler. Device time is wasted only from the time of the interrupt until the driver issues the next I/O request.

With 4000 characters and processing each interrupt takes 50 usec, the overhead would lead to a 0.2 second delay in processing a single page. For 20 pages, the delay would be 2 seconds In overhead.

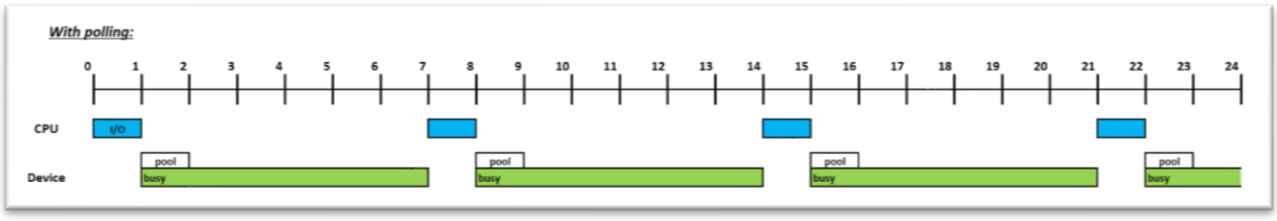
(b)

As there is only a single process (printing) is running, polling would lead to a better approach in this specific example. The CPU can busy-wait by executing a polling loop because no other computation is available to use the CPU in the meantime.

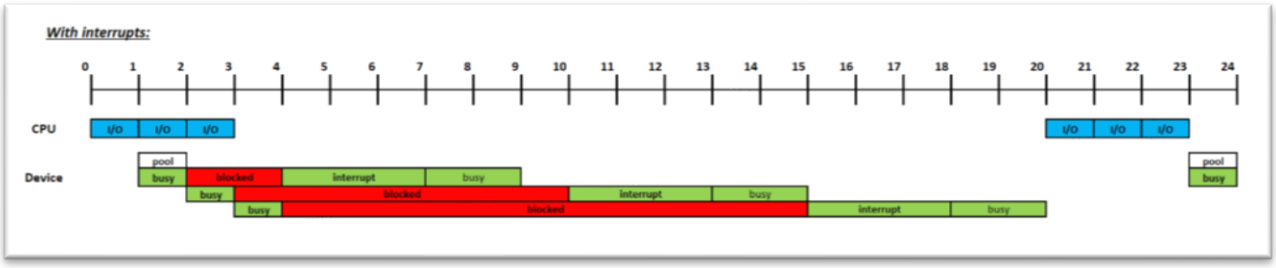
Exercise 9.1.3:

(a)

1. with polling



2. with interrupts



Exercise 9.1.4:

(a)

Programmed output with polling is analogous. When the device is not busy, the CPU copies the data from main memory to the controller buffer and issues an output request. The CPU then polls the busy flag until the operation completes. If the operation was successful, the CPU may proceed with the next output operation.

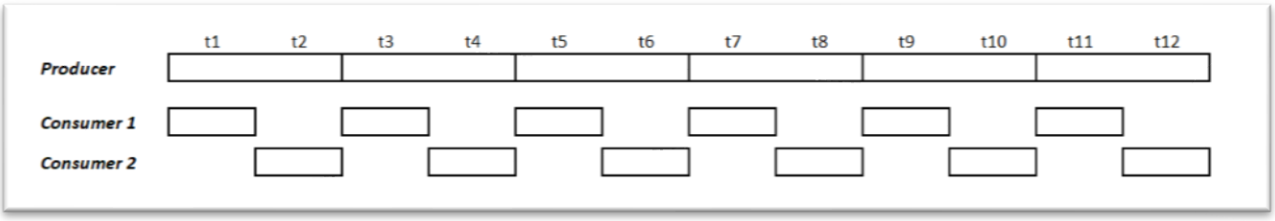
```
Repeat
    If busy == False:
            Copy data item from main memory
            Write data item to the controller buffer
            Issue an output request
            While output request:

            busy = True
        status = success
```

Exercise 9.2.1:

(a)

If the producer produces two times more than a single consumer can consume, then two consumers can be user to keep the producer busy at all times.



(b)

If C is a multiple of P, that means that C is larger than P.

So, for example if C = 4, then P would have to be 2 (2 * 2 = 4).
And if C = 9, then P would have to be 3 (3 * 3 = 9).
And if C = 16, then P would have to be 4 (4 * 4 = 16).

Thereby, the number of consumers would have to be P*2.

(c)

If C is less than P, then consumers would have to be equal or more than the number of producers.

Exercise 9.3.2:

(a)

800 tracks * 180 sectors per track = 144,000
144,000 * 512 bytes per sector = 73,728,000 bytes
= 73.728 MB

(b)

800 tracks * 180 sectors = 144,000 seek operations

(c)

5000 rpm / 60 = 83.33 rotations/sec
1 revolution takes 1 / 83.33 = 0.012 sec

73.728 MB / 800 tracks = 0.09216 MB for each track
0.09216 MB / 0.012 sec = 7.68 MB/sec

(d

The seek time between adjacent tracks is 2 ms, so 2 ms is the time it will take to read a sector on track t+1.

(e)

Sustained data rate is the rate at which the disk can transfer data continuously. The sustained data rate includes the seek times over multiple tracks and other overhead in accessing the data over time.

800 tracks * 0.012 sec (1 revolution) * 180 sectors
= 1728 / 7.68 MB
= 225 MB/sec

Exercise 9.3.4:

(a)

- FIFO
  - 143, 86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130
- SSTF
  - 143, 130, 86, 913, 948, 1022, 1470, 1509, 1750, 1774
- Scan
  - 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 130, 86
- C-Scan
  - 143, 913, 948, 1022, 1470, 1509, 1750, 1774, 86, 130

(b)

- FIFO :
  - 143 – 86 = 57
  - 86 – 1470 = 1384
  - 1470 – 913 = 557
  - 913 – 1774 = 861
  - 1774 – 948 = 826
  - 948 – 1509 = 561
  - 1509 – 1022 = 487
  - 1022 – 1750 = 728
  - 1750 – 130 = 1620
  - Total = 57 + 1384 + 557 + 861 + 826 + 561 + 487 + 728 + 1620 = 7081
- SSTF :
  - 143 – 130 = 13
  - 130 – 86 = 44
  - 86 – 913 = 827
  - 913 – 948 = 35
  - 948 – 1022 = 74
  - 1022 – 1470 = 448
  - 1470 – 1509 = 39
  - 1509 – 1750 = 241
  - 1750 – 1774 = 24
  - Total = 13 + 44 + 827 + 35 + 74 + 448 + 39 + 241 + 24 = 1745

- Scan :
  - 143 – 913 = 770
  - 913 – 948 = 35
  - 948 – 1022 = 74
  - 1022 – 1470 = 448
  - 1470 – 1509 = 39
  - 1509 – 1750 = 241
  - 1750 – 1774 = 24
  - 1774 – 130 = 1644
  - 130 – 86 = 44

  - Total = 770 + 35 + 74 + 448 + 39 + 241 + 24 + 1644 + 44 = 3319

- C-Scan :
  - 143 – 913 = 770
  - 913 – 948 = 35
  - 948 – 1022 = 74
  - 1022 – 1470 = 448
  - 1470 – 1509 = 39
  - 1509 – 1750 = 241
  - 1750 – 1774 = 24
  - 1774 – 86 = 1688
  - 86 - 130 = 44

  - Total = 770 + 35 + 74 + 448 + 39 + 241 + 24 + 1688 + 44 = 3363

Exercise 9.4.1:

(a)

1. 1 0 1 0 1 0 1 0

p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
1	1	1	1	0	1	0	0	1	0	1	0

2. 1 1 1 1 1 1 1 1

p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
1	1	1	0	1	1	1	0	1	1	1	1

3. 0 0 0 0 1 1 1 1

p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
0	0	0	1	0	0	0	0	1	1	1	1

(b)

1. 1 1 1 0 1 1 1 0 1 1 0 1

Incorrect, should be:

p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
0	0	1	0	1	1	1	1	1	1	0	1

2. 0 0 0 1 0 0 0 0 1 1 1 1

Correct.

p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
0	0	0	1	0	0	0	0	1	1	1	1

3. 1 1 1 0 0 1 0 0 1 0 1 0

Correct

p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8
1	1	1	1	0	1	0	0	1	0	1	0

(c)

2^4 = 16 data bits
so you could have 15 data bits encoded before another parity bit is needed.

Exercise 9.4.2:

(a)

Reading all blocks on the track sequentially would require up to 3 passes over the track.

The first pass reads block 0-9 and then skips forward to read the spare block number 2.

The second pass skips through blocks 0-9 as well as the damaged block at block 10, and then reads blocks 11-14, before skipping forward to read spare block number 1.

If more blocks remain, the third pass skips through blocks 0-15 (including the two damaged blocks 10 and 15), and continues reading blocks 16 to b[t-1].

(b)

The same amount of passes (revolutions) would be required, that is up to 3 passes over the track would be needed.

(c)

If sector slipping were instead used (instead of sector forwarding), all blocks could be read sequentially in a single pass over the track.
