TUTORIAL #6 Interrupts and DMA (CE/CZ1006)

**Student Solutions**

## 6.1 Interrupts

1) Consider the system diagram in Figure 1 of case study notes.
   - The processor UART peripheral in its Serial I/O module is configured to receive data with format of 1 start-bit, 7 data-bits, 1 parity-bit and 1 stop-bit.
   - Each time the UART peripheral receives a character, it'll store the data into a buffer.
   - It will then interrupt the CPU to notify CPU that there is data available.
   - The CPU will then execute the Interrupt Service Routine (ISR) to read the character received.
   - Interrupt Latency is the term used to describe the time between interrupt request and entrance to ISR.
   - The minimum interrupt latency for the CPU is 10 μs and it takes 90 μs for the CPU to execute the instructions in the ISR (read the received data from the buffer).
   - Assume that the UART data is transferred back to back i.e. no delays between each UART packets.

   What is the maximum baud rate that can be supported with this UART interface?
   [Suggested Solution]
   1/ (10+90) μs = 10000 ISRs serviced per second
   Each character transfer involved 10 bits on UART (S-7D-P-STP).
   10 bits * 10000 = 100 Kbits per second

2) The specifications of the system in Figure 1 (case study notes) requires the buttons to be asserted 400 times over a 24-hour period. Given that the processor consumes 0.1mA current when it is idling and 50mA when it is in active mode i.e. running code, answer the following.

   (a) Supposed polled IO technique is used to sample the button status. Given that the processor poll the buttons every 100ms, each poll requires the processor to be active for 10ms, what is the average current consumed over the 24-hour period?

   (b) If interrupt-driven I/O is used to sample the button status, what is the average current consumed in the same 24-hour period? Given that it takes 20ms to service each interrupt, including interrupt latency and ISR execution.

   (c) Comment on the results obtained above.

[Suggested Solution]

(a) Polled every 100ms, each poll takes 10ms

- ⇨ 90ms(idle), 10ms(active).
- ⇨ Average current = 0.9*0.1mA + 0.1*50mA = 5.09mA

(b) Interrupt mechanism.

- ⇨ Interrupted 400 times over 24-hour
- ⇨ Active time = 400*20ms = 8000ms
- ⇨ Average current consumed over 24-hour
  = [8*50mA + ((24*60*60)-8)*0.1mA] / [24*60*60]
  = 0.1046mA

(c) Interrupt mechanism only activate the processor when needed, compared to poll IO where processor needs to be active frequently. High polling frequency is required to guarantee a good system response time.

a. The results shows Interrupt mechanism keep the processor in idle mode most of the time, thereby reducing the power consumption of the system drastically compared to when polled IO is used.

b. This enable battery powered product to last much longer and reduces the need for heat dissipation measures.

## 6.2 Direct Memory Access (DMA)

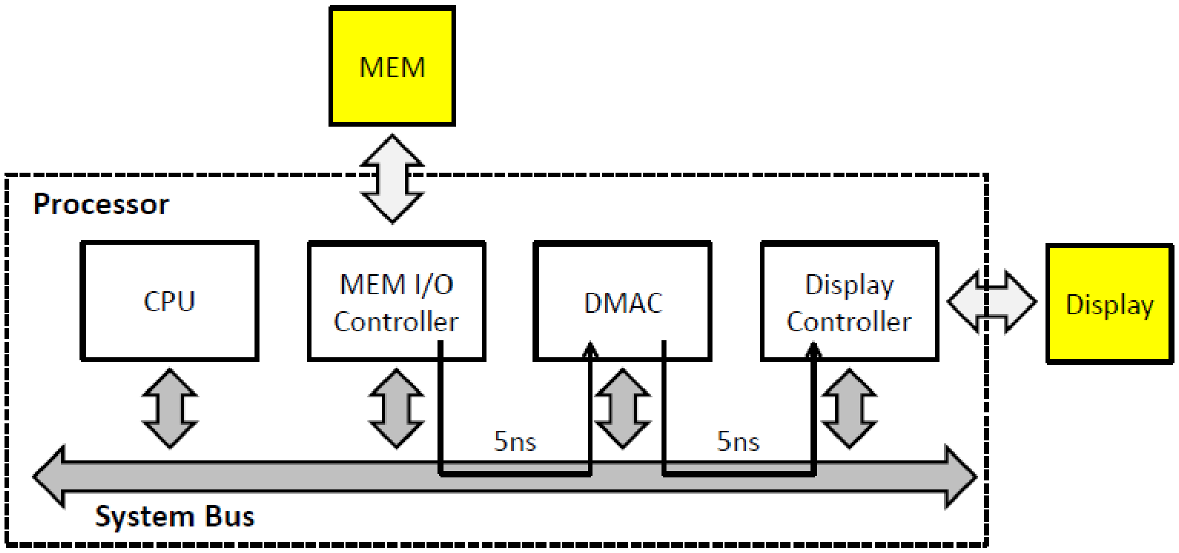
3) Although DMA does not use the CPU, the maximum transfer rate is limited by other factors. Name three possible factors that might limit the transfer rate of a block of data from the memory.

[Suggested Solution]

- CPU competes with the DMAC (DMA Controller) where system bus utilization is concerned. Higher CPU utilization rate (of the system bus) means lesser chance for DMAC to use the same bus, hence DMA data transfer rate will be affected.
- Maximum Data transfer rate is limited by the slowest device in the whole data path consisting of the source device, DMAC and destination device. If there is a memory or I/O device that is slow, it'll affect the overall DMA transfer rate.
- If cache is involved, the value of a memory location seen by the DMA controller and the processor may differ, causing coherence problem. Additional cycles needs to be used to resolve these coherence issues. E.g. Cache flushing to maintain coherency between cache and external memory.

4) Consider the system in Figure 1 (Case study notes). Given that

- The processor's system bus is capable of supporting simultaneous transfer of up to 3 bytes of data at one time.
- DMA is used to transfer video data from Memory to Display Controller module.
- Each video pixel data consist of three bytes (Red, Green, Blue) and are transferred simultaneously on the system bus on each bus cycle.
- Each transfer on the system bus takes 5ns and transfer of the bus control between the CPU and the DMAC takes 100ns.
- Assume that DMAC is using Fetch-and-Deposit DMA.
- Note that 1Kbyte = 1024 Byte.



(a) Given that the video is output at a rate of 30 frames per second and each video frame has a resolution of 1920x1080 pixels. If burst-mode was used by the DMAC to burst one frame of video data at a time, would the DMAC be able to transfer each video frame completely?

[Suggested Solution]

Each transfer of the system bus will transfer 3 bytes of video data (RGB) and take 2*5ns.

Number of transfers in 1/30 seconds = 1920*1080 = 2073600 transfers.

Total time taken

= Bus control request + actual transfer + Bus release

= 100ns + 2073600*2*5ns + 100ns = 0.0207362 sec < 1/30sec

⇨ DMAC is able to transfer each frame of video before the start of the next frame request.

(b) Repeat the calculation if the DMAC is using cycle-stealing mode, assume that DMAC needs to wait at least 5 instructions before it could request for control of the system bus again.

[Suggested Solution]

Time taken for one transfer

= Bus control request + 1 transfer + Bus release + 5*CPU Clock cycles wait

= 100ns + 2*5ns + 100ns + 5*(1/400Mhz) = 222.5ns

Total time taken for 2073600 transfers = 2073600*222.5ns = 0.461376 sec > 1/30sec

⇨ DMAC will not be able to complete the transfer of one video frame before the next frame request comes in

Note that the last 5*CPU clock cycles wait is added in the above calculation to simplify the calculation process. But it’ll not meet the requirement (>100%) even if it is taken into account.